28TH BRAZILIAN SYMPOSIUM ON DATABASES

SHORT PAPERS

PROCEEDINGS

September 30th – October 3rd, 2013
Recife, Pernambuco, Brazil

Promotion

Brazilian Computer Society – SBC
SBC Special Interest Group on Databases

Organization

Universidade Federal de Pernambuco – UFPE

Realization

Centro de Informática (CIn)

Short Papers Chairs

Renato Fileto, UFSC and Marco Cristo, UFAM

Editorial

A scientific society needs appropriate venues to discuss ideas and accomplishments. SBBD is the official database event of the Brazilian Computer Society (SBC), and the largest venue in Latin America for presentation and discussion of research results in the database domain. It joins researchers, students and practitioners, from Brazil and abroad, for discussing problems related to the main topics in modern database technologies. SBBD papers submissions have been organized into two independent tracks: full papers and short papers. Short papers should present interesting and exciting recent results or novel thought-provoking ideas that are not quite ready in any subject on databases and related areas.

The 28th edition of SBBD accepted 29 short papers, out of the 69 submitted. Each paper was evaluated by at least three members of the Program Committee, according to the criteria of originality, technical quality, presentation, relevance to SBBD and contributions. The Short Papers Program Committee was composed by 51 distinguished researchers working actively in the database area, and 5 additional reviewers appointed by some of these researchers. All accepted short papers will be presented as posters in an interactive setting, published electronically, and indexed by BDBComp (http://www.lbd.dcc.ufmg.br/bdbcomp). Additionally, some selected short papers will be invited for oral presentation in a technical session during the event, and to be re-submitted in an extended English version for a fast-track revision for publication in the Journal of Information and Data Management (JIDM).

Many people have contributed for the success of the SBBD 2013 short papers track. We would like to thank the authors for submitting their papers, and the anonymous reviewers who generously donated their expertise and time to help select and improve these papers. Special thanks to Renata Galante, who chaired the short papers track last year and shared her experience kindly to support our work; Mirella Moro, for her wise advisement and prompt last minute reviews; and Vanessa Braganholo, for her attentive assistance in the selection of papers for JIDM. We would also like to express our gratitude to the Local Organization Committee, in particular to Bernadette Farias Lóscio, for taking care of the logistics behind SBBD; and Cristina Dutra de Aguiar Ciferri, for her important integration role in the General Program Committee. We are grateful for their competent service and proud of these graciously capable women, whose leadership has been vital for the proper functioning of our community.

We hope you enjoy this short paper edition, and that its contents prove to be useful.

Renato Fileto
SBBD 2013, Short Paper Chair

Marco Cristo
SBBD 2013, Short Paper Chair

28TH BRAZILIAN SYMPOSIUM ON DATABASES

September 30th – October 3rd, 2013

Recife, Pernambuco, Brazil

Promotion

Brazilian Computer Society – SBC
SBC Special Interest Group on Databases

Organization

Universidade Federal de Pernambuco – UFPE

Realization

Centro de Informática (CIn)

SBBD Steering Committee

Marco A. Casanova, PUC-Rio (Chair)
Ana Carolina Salgado, CIn-UFPE
Cristina Dutra de Aguiar Ciferri, USP
José Palazzo Moreira de Oliveira, UFRGS
Mirella M. Moro, DCC-UFMG

SBBD 2013 Committee

Steering Committee Chair
Marco A. Casanova, PUC-Rio
Local Organization Chair
Bernadette Farias Lóscio, CIn-UFPE
Program Committee Chair
Cristina Dutra de Aguiar Ciferri, USP
Short Papers Chairs
Renato Fileto, UFSC and Marco Cristo, UFAM
Demos and Applications Chairs
Damires Souza, IFPB and Daniel Kaster, UEL
Thesis and Dissertation Workshop Chairs
Karin Becker, UFRGS and André Santanchè, UNICAMP
Tutorials Chair
Mirella M. Moro, DCC-UFMG
Lectures Chair
João Eduardo Ferreira, IME-USP

Local Organization Committee

Bernadette Farias Lóscio, CIn-UFPE (Chair)
Ana Carolina Salgado, CIn-UFPE
Agenor Marinho de Souza Neto, CIn-UFPE

Carlos Eduardo Santos Pires, DSC-UFCG
Danusa Ribeiro Cunha, CIn-UFPE
Priscilla Vieira, CIn-UFPE
Robson Fidalgo, CIn-UFPE

Short Papers Program Committee

Alexandre de Assis Bento Lima, UFRJ
Ana Carolina Salgado, UFPE
André Luiz Costa Carvalho, UFAM
Carlos Eduardo Santos Pires, UFCG
Carmem Hara, UFPR
Clodoveu Davis, UFMG
Daniel Kaster, UEL
Daniel Lichtnow, UFSM
Daniela Barreiro, UFBA
Deise Saccol, UFRGS
Denio Duarte, UFFS
Eduardo Ogasawara, CEFET/RJ
Eli Cortez, Neemu Technologies
Fabricio Benevenuto, UFMG
Fernanda Araujo Baião, NP2Tec/UNIRIO
Geraldo Zimbrao, UFRJ
Gilberto Z. Pastorello, University of Alberta
Giseli Rabello Lopes, PUC-Rio
Helena Graziottin Ribeiro, UCS
Humberto Luiz Razente, UFU
Jonice Oliveira, UFRJ
Jose Macedo, UFC
Jose Rodrigues Jr., ICMC-USP
Jucimar Souza, IFAM
Jussara Almeida, UFMG
Klessius Berlt, UFAM
Leandro Krug Wives, UFRGS
Luciana Alvim Santos Romani, Embrapa
Luiz André P. Paes Leme, UFF
Luiz Merschmann, UFOP
Marcela Xavier Ribeiro, UFSCar
Marcos Rodrigues Vieira, IBM Research Rio
Maria Camila Nardini Barioni, UFU
Maria Luiza Machado Campos, UFRJ
Mirella M. Moro, UFMG
Monica R. P. Ferreira, ICMC-USP
Nádia Puchalski Kozievitch, IFPR
Rebeca Schroeder, UFPR

Renata Galante, UFRGS
Ricardo Torres, UNICAMP
Robson Cordeiro, ICMC-USP
Robson Fidalgo, UFPE
Sandra de Amo, UFU
Sergio Lifschitz, PUC-Rio
Sergio Manuel Serra Da Cruz, UFRRJ
Stanley Oliveira, Embrapa
Valeria Cesario Times, UFPE
Valéria Gonçalves Soares, UFPB
Vânia Vidal, UFC
Vasco Furtado, UNIFOR
Viviane P. Moreira, UFRGS

External Reviewers

Josiel Figueiredo, UFMT
Sergio Mergen, Unipampa
Demetrio Gomes Mestre, UFCG
Valeria Pequeno
Lucio F. D. Santos, ICMC-USP

28TH BRAZILIAN SYMPOSIUM ON DATABASES

SHORT PAPERS

Table of Contents

Projeto de banco de dados de simulações numéricas

Ramon G. Costa, Fábio Porto, Bruno Schulze, Hermano Lustosa

LNCC - Laboratório Nacional de Computação Científica, Brasil
{ramongc, fporto, schulze, hlustosa}@lncc.br

Resumo. Com a rápida evolução dos sistemas computacionais, simulações numéricas baseadas em modelagem computacional têm alcançado soluções cada vez mais realistas. Ainda assim, o processo de simulação é complexo, exigindo grande capacidade computacional e produzindo muitos arquivos auxiliares com os resultados das simulações. Uma grande quantidade de arquivos, como os produzidos durante o processo de varredura de parâmetros, torna a gerência de experimentos uma tarefa bastante complexa, que associado ao cálculo da simulação, analisam seus resultados com programas específicos que precisam ser construídos e que fazem acesso ineficiente aos arquivos, muitas vezes em texto bruto. Por outro lado, a adoção de sistemas de gerência de bancos de dados (SGBD) em apoio à simulações numéricas traz seus próprios desafios. A representação de domínios espaço-temporal e de variáveis que representam quantidades físicas se mostra adequada para o armazenamento e consulta a esses tipos de dados. Neste contexto, este trabalho visa preencher uma lacuna existente na direção da adoção de SGBDs para o tratamento de dados de simulações numéricas. O focos deste trabalho está no processo de projeto do bancos de dados. Identifica-se, a partir do processo de projeto tradicional de banco de dados, extensões necessárias que permitem mapear o modelo conceitual de uma aplicações de simulação numérica em um projeto que combina modelo de dados relacional com modelo de matrizes multidimensionais. Apresenta-se um exemplo de aplicação de simulação do sistema cardiovascular humano e o processo de projeto de banco de dados implementado no sistema SciDB. Este trabalho tem como objetivo definir as fases de um projeto para o armazenamento de dados gerados por simulações numéricas, demonstrando a necessidade do armazenamento destes dados em array multidimensionais e o uso de técnicas tradicionais durante o processo de projeto do banco de dados. O conjunto de dados é representado através do uso do SciDB, que utiliza um modelo de dados em matrizes multidimensionais como a base de sua representação.

Categories and Subject Descriptors: G.1.10 [**Numerical Analysis**]: Applications; H.2.8 [**Database Management**]: Scientific databases; I.6.7 [**Simulation and Modelling**]: Simulation Support Systems

Keywords: Database Management, Information Storage and Retrieval, Numerical Analysis, Simulation and Modelling

1. INTRODUÇÃO

A simulação numérica de fenômenos naturais tem sido favorecida com os grandes avanços nas plataformas de computação de alto desempenho, obtendo representações mais realistas dos fenômenos modelados. No processo de modelagem computacional de fenômenos naturais, tais como o do sistema cardiovascular humano (SCH), utiliza-se um conjunto de equações diferenciais para representar as variações das quantidades no espaço-tempo. O modelo matemático é discretizado em um modelo computacional usando algum método numérico disponível. Do ponto de vista computacional, o estado da arte para o processamento de simulações numéricas utiliza arquivos texto para o armazenamento de dados. É fácil imaginar que a gerência de milhares de arquivos, como os produzidos durante o processo de varredura de parâmetros, torna a gerência de experimentos uma tarefa bastante complexa para o pesquisador. Ainda pior, uma vez calculada a simulação, cientistas analisam seus resultados, com programas de análise específicos que precisam ser construídos para o acesso ineficiente e não padronizado aos arquivos em formato de texto bruto.

O caráter científico de aplicações de simulações numéricas exige igualmente que os resultados obtidos possam ser reproduzidos e o processo de cálculo seja auditável. Neste contexto, dados de proveniência [Buneman et al. 2001] descrevem o processo experimental apoiando o cientista no processo de análise de seus resultados. Em particular, simulações numéricas requerem, além da gerência de dados de simulações propriamente dita, o tratamento de dados de proveniência.

Neste contexto, estender os benefícios do gerenciamento de dados em bancos de dados à aplicações de simulação numérica se torna essencial, uma vez que leva a esse domínio as vantagens já consagradas

na adoção de SGBDs em outros domínios, como na astronomia [Szalay et al. 2000]. Esta adoção, todavia, traz seus próprios desafios. A representação de domínios espaço-temporal determinando as variações admissíveis dos valores de quantidades físicas requer um modelo de dados mais apropriado do que simples relações n-árias [Stonebraker 2012]. Dentro deste contexto, o modelo de dados de matrizes multidimensionais implementado no sistema SciDB torna-se uma boa solução [Costa et al. 2012]. Naquela proposta apresentada, malhas espaciais tridimensionais e sua variação no tempo são mapeadas para dimensões em matrizes multidimensionais, enquanto as varíaveis são representadas em células. Neste trabalho, avançamos nesta proposta propondo um método para a derivação do projeto de bancos de dados para simulações numéricas a partir da representação conceitual do domínio, conforme adotado no projeto tradicional de bancos de dados.

Para ilustrar a utilização deste modelo no problema de simulações numéricas, considere a simulação do SCH desenvolvida pelo projeto Hemolab no LNCC [Blanco et al. 2009]. Nesta aplicação constrói-se uma representação geométrica das artérias do corpo humano na forma de uma malha multidimensional. Ao longo da artéria simulada, modelos com malhas em dimensões 1D e 3D são acoplados, segundo o interesse de pesquisa e a capacidade de processamento. Um modelo de simulação calcula, para cada instante de tempo e ponto da malha, valores de interesse como pressão e velocidade.

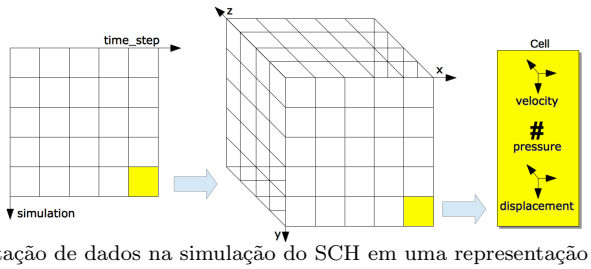


Fig. 1: Representação de dados na simulação do SCH em uma representação multidimensional

A Fig. 1 ilustra um modelo multidimensional para representar os dados calculados pela simulação do SCH. A ilustração apresenta as dimensões *simulation* e *time_step* e os pontos de uma malha tridimensional, referentes à simulação s e a um instante de tempo t. Na célula indicada, aparecem os valores das quantidades: velocidade, pressão e deslocamento. Assim sendo, o mapeamento da representação ilustrada na Fig. 1 para um modelo de dados de matrizes multidimensionais fica facilitado. As dimensões e os valores das células encontram correspondência direta nas estruturas de dimensão e células do modelo. A partir desta representação, podem-se elaborar consultas com base no espaço multidimensional construído, tal como obter os valores de velocidade e pressão em uma simulação s_i e tempo t_j para uma área da malha nas coordenadas $< x, y, z >$. Com isto, ao prover uma linguagem de alto-nível para expressões de consulta tem-se uma contribuição importante para cientistas que utilizam arquivos de texto bruto como resultado de suas simulações, obtendo ganhos maiores quando se utilizam estruturas de armazenamento e indexação eficientes [Zhang, Yi et al. 2011] para otimizar o acesso em disco.

O exemplo acima ilustra, de forma preliminar, a adequação do modelo de dados de matrizes multidimensionais no gerenciamento de dados de simulações numéricas. Uma questão fundamental, não explorada na literatura, é: *Como projetar um banco de dados para o modelo de matrizes multidimensionais, considerando-se as técnicas de projetos de bancos de dados*?

Neste contexto, este trabalho apresenta um método para o projeto de bancos de dados de simulações numéricas, tendo como alvo sua representação em modelos de dados de matrizes multidimensionais, bem como o tratamento de dados de proveniência produzidos durante o cálculo da simulação.

2. TRABALHOS RELACIONADOS

Vários sistemas têm surgido oferecendo suporte ao modelo de dados de matrizes multidimensionais [Stonebraker 2012] [Baumann et al. 1998], expondo as vantagens e os desafios no armazenamento e manipulação de *arrays* de forma nativa pelo sistema. Alguns trabalhos evocam a necessidade do uso de SGBDs para o armazenamento e gerenciamento de dados de simulação numérica, mas ainda utilizam

o modelo de dados relacional como forma de representação, como é o caso do *Turbulence Database Cluster* [Perlman et al. 2007]. Nosso trabalho considera que a utilização de matrizes multidimensionais são adequadas para o armazenamento e gerência de bancos de dados de simulações numéricas.

Trabalhos recentes discutem a necessidade de um tratamento mais eficiente para o armazenamento, indexação e processamento de dados científicos [Ogasawara et al. 2011] e [Zhang, Yi et al. 2011]. Stonebraker [Stonebraker et al. 2009] afirma que as atuais tecnologias de gerenciamento de dados são claramente incapazes de lidar com as exigências dos cientistas. Conhecer os requisitos destacados em projetos bem sucedidos se torna importante. Dentre estes projetos, podemos destacar o SciDB: um SGBD Orientado à Ciência [Stonebraker 2012] e os esforços de seus pesquisadores para especificar o conjunto de requisitos para a criação de um sistema de bancos de dados para a área científica. Este trabalho utiliza o SciDB para ilustrar o projeto físico do banco de dados de matrizes multidimensionais, demonstrando como um modelo conceitual pode ser mapeado para um esquema em *array*.

3. O PROJETO DE BANCOS DE DADOS PARA SIMULAÇÕES NUMÉRICAS

Para desenvolver o raciocínio que segue para a criação de bancos de dados de simulações numéricas, o processo será descrito a partir da análise dos requisitos que definem o ambiente do sistema, passando pelas fases de projeto conceitual até o projeto físico. Esta abordagem, chamada de processo de projeto *Top-down*, será utilizada neste trabalho para descrever o processo de projeto de bancos de dados de simulação numérica (Fig. 2). As fases do projeto são detalhadas nas subseções que se seguem.

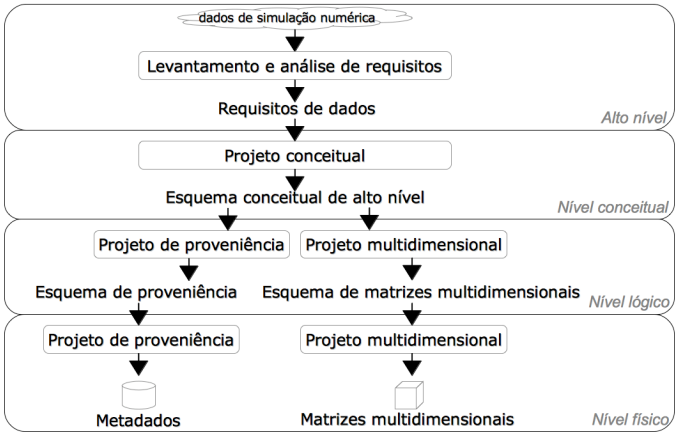


Fig. 2: Processo de projeto Top-Down para os dados gerados por simulações numéricas

3.1 O minimundo: dados de simulações numéricas

As simulações numéricas, nas quais o movimento de um fluido pode ser descrito pela especificação completa das propriedades a serem computadas em função de coordenadas espaço-temporais, são escopo de nossa pesquisa. Nas simulações numéricas de interesse, obtém-se informações do escoamento em função do que acontece em pontos fixos do espaço, conhecido como método de descrição euleriana do movimento [Batchelor 2000] e suas propriedades computadas referidas como quantidades físicas eulerianas.

Neste trabalho, os dados de simulação numérica são interpretados como sendo compostos por dois conjuntos de variáveis: variáveis dimensionais e quantidades físicas eulerianas. O primeiro conjunto define um sistema de coordenadas multidimensional, enquanto o segundo informa sobre as quantidades físicas calculadas em cada ponto de referência. Tipicamente, as variáveis dimensionais incluem espaço e tempo, além da dimensão identificadora de simulação. A dimensão espacial refere-se a uma malha que representa a topologia do domínio físico por meio de uma composição de simples objetos geométricos (por exemplo, um tetraedro). Uma malha é representada por um conjunto de pontos, referindo-se aos vértices dos objetos geométricos, e um conjunto de arestas que ligam os pontos e as faces do modelo, assim como em um grafo não direcionado.

No âmbito do projeto HeMoLab [HeMoLab 2013], a simulação do SCH pode se valer de um modelo tridimensional (3D) para representação espacial de seu domínio. Os modelos 3D são utilizados para obter um maior nível de detalhe sobre o comportamento do fluxo de sangue ao longo da artéria para uma estrutura especificada [Blanco et al. 2009].

A simulação computacional de um modelo do escoamento de sangue em uma artéria é feita através da resolução de um sistema de equações de Navier Stokes que descreve o movimento de fluidos [Formaggia et al. 2001]. Este processo é realizado por softwares informalmente referidos como *resolvedores numéricos*. Este último, recebe um conjunto de parâmetros que descrevem: a representação geométrica do domínio e parâmetros próprios da simulação. Cada período de simulação (geralmente um período refere-se a uma batida do coração) é subdividido em passos de tempo (um valor típico é 640) e para cada passo de tempo, um resolvedor numérico atualiza o arquivo que armazena os resultados da simulação com os valores de deslocamento, velocidade e pressão em cada ponto do volume. Por fim, durante o cálculo da simulação, armazenam-se os dados de proveniência informando sobre os programas utilizados, os parâmetros de entrada e os resultados obtidos.

3.2 O projeto do banco de dados de matrizes multidimensionais e de proveniência

Na abordagem adotada, inicia-se o projeto do banco de dados de simulações numéricas com o projeto conceitual dividido em duas partes. A primeira representa as dimensões envolvidas e as quantidades físicas associadas. A segunda diz respeito aos dados de proveniência associados ao cálculo da simulação.

Vários sistemas têm surgido oferecendo suporte ao modelo de dados de matrizes multidimensionais [Stonebraker 2012] [Baumann et al. 1998]. Como exemplo específico de implementação, podemos representar o esquema multidimensional usando o SciDB. Como as matrizes multidimensionais são a base de representação de dados no SciDB, um usuário pode especificar estruturas multidimensionais, fornecendo intervalos de valores para cada dimensão e uma lista de atributos para compor uma célula. Neste contexto, a estratégia de mapeamento a seguir, foi definida: *(i)* define-se o conjunto Δ de dimensões envolvidas; *(ii)* especifica-se a lista de quantidades eulerianas Π a serem computadas (cada conjunto de quantidades eulerianas correspondem a um fenômeno simulado em uma dada escala); *(iii)* cria-se uma matriz multidimensional contendo as dimensões Δ e os atributos Π.

Neste contexto, pode-se adotar a estratégia de anotação na representação conceitual Entidade-Relacionamento (ER) [Parent et al. 2006] para distinguir tais atributos neste modelo. Assim, na primeira parte, estende-se a representação conceitual para caracterizar cada atributo quanto ao seu propósito no ambiente de simulação numérica. Para cada atributo ou papéis de um relacionamento recursivo, deve-se identificar quais devem ser modelados como Metadados [M], Dimensões [D] ou Quantidades físicas eulerianas [Q]. Outros atributos ou papéis não devem conter identificação. A Fig. 3 apresenta um Diagrama ER de nível conceitual com as anotações [M], [D] e [Q].

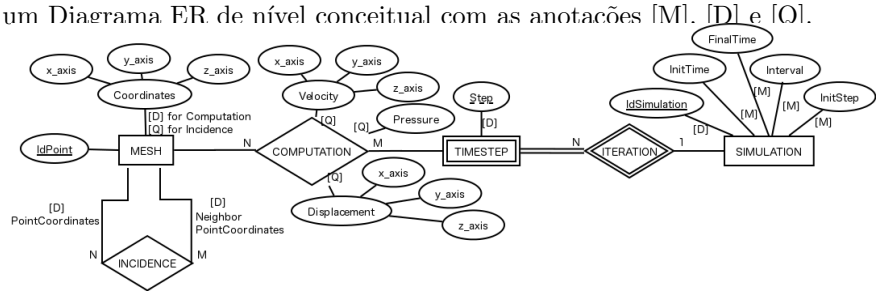


Fig. 3: Diagrama ER de Peter Chen com as anotações que direcionam o mapeamento para matrizes multidimensionais

No mapeamento do diagrama ER (Fig. 3) para uma representação lógica usando hipercubos (Fig. 4), as seguintes observações devem ser feitas: *(i)* identifique os dois grupos de informacões: atributos dimensionais [D] e atributos referentes às quantidades físicas eulerianas [Q]; *(ii)* as quantidades físicas eulerianas são mapeadas em células, onde os atributos correspondem às anotações [Q]; *(iii)* a dimensões espacial, temporal e identificadora das simulações devem ser representadas como dimensões do hipercubo; *(iv)* a matriz de adjacência entre os pontos da malha pode ser representada como um novo hipercubo.

A distinção entre atributos dimensionais e quantidade físicas requer a definição de conjunto de atributos determinantes funcionais. Para isso adota-se a representação de dependência funcional [Sadri and Ullman 1980] entre o conjunto de atributos. Sendo assim, Seja $A = \{a_1, a_2, \ldots, a_n\}$, tal que $a_i \subset d_i$, para todo $1 \leq i \leq n$, e d_i um domínio qualquer de valores, onde A é o conjunto de atributos envolvidos no cálculo de uma simulação numérica. Determinam-se os subconjuntos $D \subset A$ e $Q = A - D$, tais que D determina funcionalmente Q [Sadri and Ullman 1980]. O conjunto de atributos em D corresponde às dimensões do problema, enquanto o conjunto de atributos em A identifica as quantidade eulerianas. Neste caso:

`{Simulation, TimeStep, Point3D}` $\rightarrow$ `{velocity3D, pressure, displacement3D}`
`{Simulation, Point3D, NeighborPoint3D}` $\rightarrow$ `NeighborCoordinate3D`

A Fig. 4 apresenta a estrutura multidimensional resultante deste mapeamento.

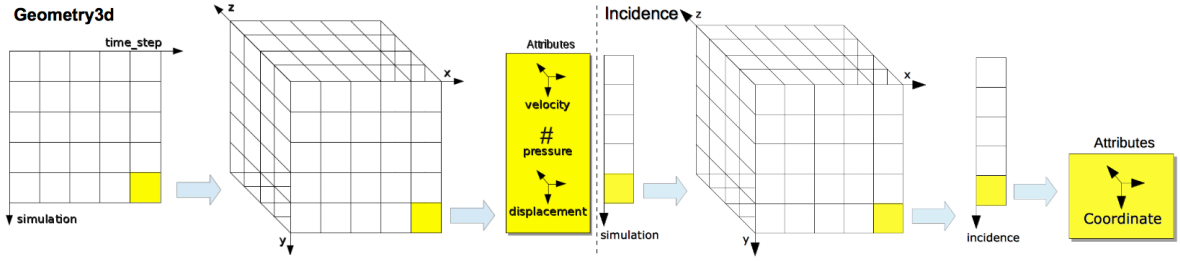


Fig. 4: Representação do modelo 3D do SCH em hipercubos

Através das linguagens AQL e AFL é possível seguir para o projeto físico e implementar o esquema interno que armazena dados em matrizes multidimensionais. A Tab. I apresenta o código para a criação do esquema interno do hipercubo *Geometry3d* apresentada na Fig. 4.

Tab. I: AQL utilizada para representar a matriz multidimensional *Geometry3d*

```
create empty array Geometry3d
<velocity_x:float, velocity_y:float, velocity_z:float, pressure:float,
displacement_x:float, displacement_y:float, displacement_z:float>
[simulation_number=0:9,1,0, time_step=0:30720,1921,0, x_axis(float)=155000,155000,0,
y_axis(float)=155000,155000,0, z_axis(float)=155000,155000,0];
```

Para o projeto do banco de dados de proveniência, deve-se levar em consideração as anotações referentes aos metadados [M] representados no diagrama da Fig. 3, e combinados aos dados de proveniência encontrados no problema, modelar um diagrama respeitando o domínio adotado. A Fig. 5(a) apresenta um diagrama, modelado para os dados de proveniência, usando o PROV-DM [Moreau et al. 2012].

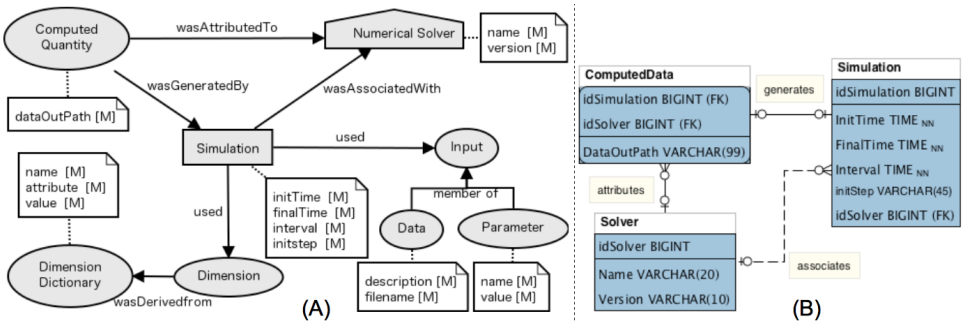


Fig. 5: (a) Diagrama PROV-DM; e (b) Diagrama *Crow's foot* para parte do conjunto de dados de proveniência.

No mapeamento do diagrama conceitual de proveniência (Fig. 5(a)) para o diagrama de nível lógico (Fig. 5(b)), as Entidades, os Agentes e as Atividades [Moreau et al. 2012] podem ser mapeadas como tabelas e as anotações [M] são mapeadas como atributos em cada tabela.

A Tab. II apresenta a implementação do esquema lógico representado na Fig. 5(b).

Tab. II: SQL utilizada para representar o catálogo de metadados

```
create database Cardio_catalog;
create table Cardio_catalog.Solver (
  idSolver serial primary key, name varchar, version varchar );
create table Cardio_catalog.Simulation (
  idSimulation serial primary key, InitTime time, FinalTime time,
  Interval time not null, initStep bigint,
  foreign key (idSolver) references Cardio_catalog.Solver(idSolver) );
create table Cardio_catalog.Computed_data (
  idSimulation bigint, idSolver bigint, DataOutPath varchar,
  primary key (idSimulation, idSolver),
  foreign key (idSimulation) references Cardio_catalog.Simulation(idSimulation),
  foreign key (idSolver) references Cardio_catalog.Solver(idSolver) );
```

4. CONCLUSÃO

Este trabalho propôs um processo de projeto *top-down* para o desenvolvimento de bancos de dados de simulação numérica. Dados desta natureza têm a característica de mudarem com o tempo e espaço e de serem envolvidos em operações próprias, tais como: multiplicação de matrizes, transposição de matrizes e estudo de convergências de valores. Devido a estas características, se torna adequado o uso de sistemas de bancos de dados que permitem o armazenamento de matrizes multidimensionais de forma nativa. Este trabalho demonstrou que os métodos tradicionais de projeto de bancos de dados podem ser utilizados em um processo que culmina na criação de matrizes multidimensionais para o armazenamento de dados de simulação numérica envolvidos durante o cálculo da simulação.

Muito se tem discutido a respeito do uso de estratégias diferentes das tradicionais para o armazenamento de dados científicos, mas nenhuma delas aborda o fato de definir um método para a especificação de bancos de dados de simulações numéricas. O resultado apresentado neste trabalho é uma proposta até então inexistente na área.

REFERENCES

Batchelor, G. K. *An Introduction to Fluid Dynamics*, 2000.

Baumann, P., Dehmel, A., et al. The multidimensional database system rasdaman. In *Proceedings ACM SIGMOD International Conference on Management of Data*. Seattle, USA, pp. 575–577, 1998.

Blanco, P. J., Pivello, M. R., Urquiza, S. A., and Feijóo, R. A. On the potentialities of 3d-1d coupled models in hemodynamics simulations. *Journal of Biomechanics* 42 (7): 919–930, 2009.

Buneman, P. et al. Why and where: A characterization of data provenance. In *ICDT*. pp. 316–330, 2001.

Costa, R. G., Porto, F., and Schulze, B. Towards analytical data management for numerical simulations. In *AMW*. pp. 210–214, 2012.

Formaggia, L., Gerbeau, J.-F., et al. On the Coupling of 3D and 1D Navier-Stokes Equations for Flow Problems in Compliant Vessels. *Computer Methods in Applied Mechanics and Engineering* 191 (6-7): 561–582, 2001.

HeMoLab. Hemodynamics modeling laboratory, 2013. `http://hemolab.lncc.br`.

Moreau, L., Missier, P., et al. Prov-dm: The prov data model. Tech. rep., 2012.

Ogasawara, E., Dias, J., de Oliveira, D., Porto, F., Valduriez, P., and Mattoso, M. An algebraic approach for data-centric scientific workflows. In *Proceedings of the VLDB Endowmt*. Vol. 4. Seattle, USA, pp. 1328–1339, 2011.

Parent, C., Spaccapietra, S., and Zimányi, E. *Conceptual modeling for traditional and spatio-temporal applications - the MADS approach*. Springer, 2006.

Perlman, E., Burns, R., Li, Y., and Meneveau, C. Data exploration of turbulence simulations using a database cluster. In *Proceedings of the IEEE conference on Supercomputing*. Vol. 23. ACM, New York, USA, pp. 1–11, 2007.

Sadri, F. and Ullman, J. D. The interaction between functional dependencies and template dependencies. In *SIGMOD Conference*. pp. 45–51, 1980.

Stonebraker, M. Scidb: An open-source dbms for scientific data. *ERCIM News* 2012 (89), 2012.

Stonebraker, M., Becla, J., Dewitt, D., Lim, K.-T., Maier, D., Ratzesberger, O., and Zdonik, S. Requirements for science data bases and scidb. In *Conference on Innovative Data Systems Research (CIDR)*. Asilomar, USA, 2009.

Szalay, A. S., Kunszt, P. Z., et al. Designing and mining multi-terabyte astronomy archives: the sloan digital sky survey. In *Proceedings of the ACM SIGMOD Conf. on Management of data*. New York, USA, pp. 451–462, 2000.

Zhang, Yi et al. Storing matrices on disk: Theory and practice revisited. In *VLDB*. Seatle, USA, 2011.

Análise de Conformidade de Processos de Negócios : Experiência com os Dados do MPTCM-PA

Müller Miranda[1], Fábio Bezerra[2]

[1] Instituto de Ciências Exatas e Naturais
Universidade Federal do Pará – Belém, PA – 66.075-110
mullergsm@ufpa.br
[2] Instituto Ciberespacial
Universidade Federal Rural da Amazônia – Belém, PA – 66.077-901
fabio.bezerra@ufra.edu.br

Resumo.

Técnicas de *Process Mining* são utilizadas para extração automática de informações sobre processos de negócio a partir de logs de dados, assim, pode-se averiguar a conformidade entre o processo planejado e o executado. A fim de ilustrar a utilidade de técnicas dessa natureza, este trabalho apresenta um relato de experiência com os dados gerados pelo sistema de tramitação de processos do MPTCM-PA. No caso, foi possível revelar diferenças de execução do processo de negócio, que estavam relacionadas à própria utilização do sistema ou erros de migração dos dados provenientes do sistema legado.

Abstract.

Process Mining techniques are used for automatic extraction of information about business processes from logs of information systems, so you can verify the conformity between the planned process and the actual process. To illustrate the usefulness of such techniques, this paper presents an experience report with the data generated by the process-aware system of MPTCM-PA. It was possible to reveal differences in the implementation of the business process, which were related to erroneous use of the system or data migration from the legacy system.

Categories and Subject Descriptors: H.2 [**Database Management**]: Miscellaneous; H.3 [**Information Storage and Retrieval**]: Miscellaneous; D.4 [**Operating Systems**]: Process Management

Keywords: Conformance, Business Processes, MPTCM-PA, Data Migration, Process Mining

1. INTRODUÇÃO

Um processo de negócio é um conjunto de atividades e recursos organizados para produzir valor para seus clientes e a eficiência como são conduzidos contribui para o sucesso das organizações. As organizações precisam se adaptar constantemente, mudando seus processos de negócio para atender suas necessidades, exigindo conhecimento e análises de como são atualmente executados. No entanto, esta é uma tarefa custosa e complexa, que exige uma grande quantidade de tempo e recursos e, na maioria das vezes, não alcança os resultados esperados [Rozinat 2005; Belluzzo 2007].

Neste cenário, *Process Mining* apresenta-se como uma alternativa eficiente e eficaz para extrair informações de forma automática, viabilizando a análise e descoberta objetiva dos processos de negócios [van der Aalst 2011]. Tata-se de uma técnica de gestão de processos que permite a análise de processos de negócios com base em registros de eventos (*log* de dados). No Brasil, apesar de pouco adotada pela indústria, tem despertado o interesse de alguns grupos de pesquisa. Temos, como exemplo, os trabalhos [Esposito et al. 2011] e [Francisco and Santos 2011]. Então, esperamos que este trabalho seja uma oportunidade para ilustrar a riqueza de informações que o uso de técnicas e ferramentas de *Process Mining* pode revelar de forma clara e objetiva, permitindo verificar o grau de conformidade entre o planejado e o atualmente executado.

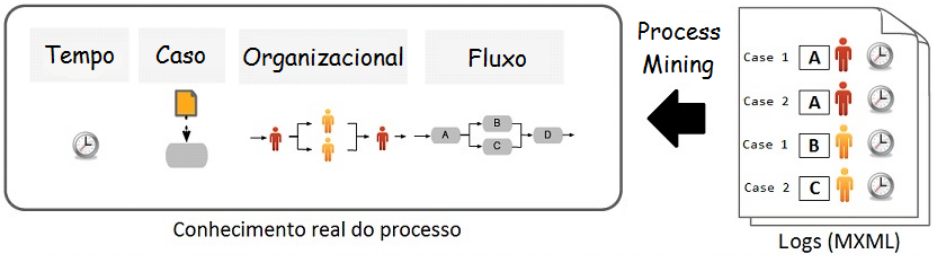


Fig. 1. As quatro perspectivas de *Process Mining*.

Diferente da abordagem *top-down*, que se inicia com a definição de um modelo de processo de negócio, *Process Mining* apresenta-se como uma abordagem *bottom-up*, ou seja, a partir do *log* de dados é possível construir o modelo de processo de negócio. O objetivo de se usar ferramentas dessa natureza é descobrir, verificar conformidade, analisar o desempenho e promover melhorias aos processos de negócios [van der Aalst 2011]. No caso, é necessário obter um *log* de dados proveniente de algum banco de dados controlado por um sistema de informação, de tal forma que cada evento represente uma atividade e está relacionado a um caso particular. Além disso, técnicas de *Process Mining* poderão considerar outros tipos de informação, tais como a pessoa que executou determinado evento, a data e hora em que este ocorreu, entre outros. Assim, diferentes perspectivas[1] podem ser analisadas.

Existem três diferentes tipos de *Process Mining* [Hornix 2007]. O primeiro é a **descoberta**, quando é possível obter o modelo do processo de negócio a partir do *log* de dados sem conhecimento prévio. O segundo é a **conformidade**, utilizada para analisar se o processo observado no *log* de dados está em conformidade com o modelo previsto. Por fim, a **extensão**, que trata de complementar ou enriquecer com informações um modelo de processo de negócio. De acordo com [van Dongen 2007], para cada tipo de *Process Mining* é possível associar quatro diferentes perspectivas (ver Figura 1), de forma que cada uma delas destaca características importantes de um processo. A **perspectiva de fluxo** tem como finalidade caracterizar todas as sequências de atividades presentes no *log*. O objetivo desta perspectiva é encontrar uma boa caracterização de todas as sequências possíveis, resultando um fluxo de atividades. A **perspectiva organizacional** tem como principal objetivo compreender os padrões de socialização, a forma como as pessoas, sistemas ou departamentos interagem entre si. A **perspectiva de caso** refere-se a descoberta dos dados ou informações manipuladas pelo processo. Mais recentemente, considera-se também, isoladamente e separada da perspectiva de caso, a **perspectiva de tempo**, que refere-se a análise do desempenho da execução dos processos de negócio [van der Aalst 2011].

Este trabalho tem por objetivo apresentar alguns resultados da análise de conformidade, ressaltando as perspectivas de fluxo, organizacional e tempo, dos dados do *Prizeus* [Encarnação et al. 2012], que é o atual Sistema de Gerenciamento e Controle dos Processos tramitados no Ministério Público junto ao Tribunal de Contas dos Municípios do Estado do Pará (MPTCM-PA), por meio da ferramenta chamada *ProM Framework* [van Dongen et al. 2005], que incorpora algoritmos de *Process Mining*. A existência de inconformidades, poderá indicar que a organização está funcionando de forma diferente do planejado ou que o modelo de negócio está desatualizado e que já não satisfaz a lógica atual.

Vale observar que grande parte dos dados do sistema *Prizeus* são oriundos de um sistema legado, e que, após migração desses dados, nenhuma validação ou teste foi aplicado de forma a identificar possíveis erros, sejam eles pelo processo de migração ou pela insuficiência nas informações. Erros nos dados podem ser registros incompletos ou valores atípicos, por exemplo. A forma como esses processos são revelados para serem corrigidos, se dá por meio de queixas de usuários que utilizam o sistema, portanto, este trabalho também tem por interesse verificar automaticamente os processos errôneos, que podem ser resultados do processo de migração e adaptação dos dados para o novo sistema (*Prizeus*).

[1]Mais detalhes no *Process Mining Manifesto*, disponível em http://www.win.tue.nl/ieeetfpm

Durante revisão bibliográfica, não identificamos artigos que apresentem este aspecto de migração relacionadas a área de *Process Mining*. Portanto, espera-se que estudos relacionados a conformidade de dados possam contribuir como instrumento interessante para o apoio ao processo de validação de migração entre bases de dados.

O restante deste trabalho está organizado como segue: na Seção 2 é apresentado o método para desenvolvimento deste trabalho, o período de extração dos dados, os algoritmos utilizados e o processo de negócio, objeto da experiência relatada na Seção 3. Finalmente, na Seção 4 são apresentadas algumas considerações finais dessa experiência prática e indicações para esforços futuros.

2. ANÁLISE DE PROCESSOS BEM ESTRUTURADOS

Processos estruturados possuem uma execução clara, costumam ser executados de uma maneira pré-estabelecida, admitem execuções excepcionais e seus usuários possuem um bom entendimento do processo de negócio [van der Aalst 2011]. Para realizar a análise desses processos há o modelo **L* life-cycle model** [van der Aalst 2011], que é formado por cinco etapas: planejamento, extração dos dados, criação de modelos, criação de modelos integrados e apoio operacional. Esse modelo será aplicado na análise do processo de negócio do MPTCM-PA, que apresenta pelo menos três evidências de que é um processo bem estruturado: **(i)** possui uma estrutura de execução clara, além de ser uma estrutura aderente a requisitos legais; **(ii)** os participantes do processo possuem um bom conhecimento geral do esquema de tramitação dos processos; e **(iii)** as responsabilidades dos participantes do processo é bem conhecida.

Este trabalho apresenta um relato de experiência do uso desse modelo de análise, com os dados gerados pelo sistema de tramitação de processos do MPTCM-PA. No caso do MPTCM-PA esse sistema é o *Prizeus*[Encarnação et al. 2012], que dá suporte à tramitação de processos de contas auditados pelo Tribunal de Contas dos Municípios do Estado do Pará (TCM-PA), pois o MPTCM-PA tem a função de fiscal da correta aplicação da lei desses processos [Moraes 2003]. No caso do MPTCM-PA, são tramitados anualmente aproximadamente 3500 processos, um volume de instâncias muito grande para uma análise manual, especialmente se considerarmos um legado de pelo menos 10 anos. Portanto, o cenário de aplicação do estudo realizado neste trabalho é motivador.

A Figura 2 ilustra o modelo de extração e análise dos dados que foi adotado neste trabalho. Inicialmente, identificamos quais os dados ou período de extração dos dados nos interessava. Nesta etapa foram verificadas as características do sistema de informação e a estrutura dos dados que são manipulados por este sistema. Além disso, procuramos conhecer o processo de trabalho e seus participantes, a fim de desenharmos uma visão geral do processo de negócio, ou seja, o modelo planejado. O período de registros extraído para a criação do *log* de dados foi de 01 de Janeiro de 2009 à 01 de janeiro de 2013, resultando em 12190 instâncias de processos. Esta primeira etapa é referente às duas primeiras etapas do modelo **L* life-cycle model** como visto anteriormente, no caso o planejamento e extração dos dados.

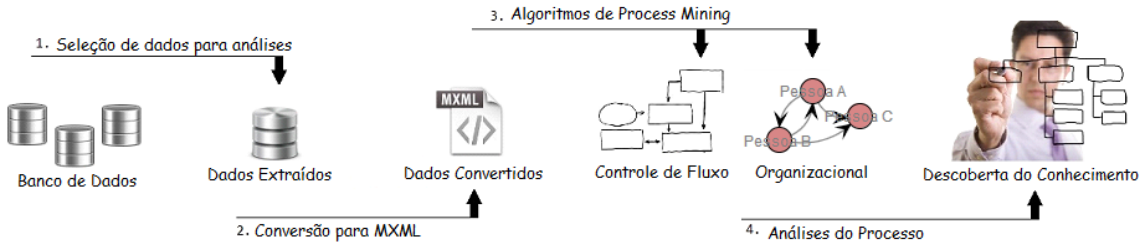


Fig. 2. Modelo de extração e análise dos dados.

Após a coleta dos dados, iniciamos o processo de conversão, que foi apoiado pela ferramenta *ProM Import*[2]. Esta conversão é necessária para carregar o *log* de dados no formato adequado (XML) para leitura no *ProM Framework*, ferramenta livre que incorpora algoritmos em todas as perspectivas de *Process Mining*. Seguindo com a terceira etapa do modelo **L* life-cycle model**, utilizamos determinados algoritmos de *Process Mining* que resultaram o modelo de controle de fluxo e organizacional. Finalmente, realizamos várias análises acerca dos modelos descobertos, possibilitando a geração do conhecimento, referente a quarta e quinta etapa do modelo **L* life-cycle model**.

Aplicamos os algoritmos *Heuristics Miner*, *Performance Analysis With Petri net* e *Performance Sequence Diagramas Analysis* para a análise do fluxo de controle do processo. O algoritmo *Heuristics Miner* [Weijters et al. 2006] é útil para lidar com ruídos no *log* de dados, resultando apenas o comportamento principal presente no processo, Figura 3. O algoritmo *Performance Analysis With Petri net* apresenta a média temporal de conclusão de atividades, ajudando-nos a avaliar o desempenho de cada participante. Finalmente, o algoritmo *Performance Sequence Diagramas Analysis* indica a frequência e as varias formas de execução do processo.

Para análise organizacional utilizamos os algoritmos: *Organizational Miner*, *Role Hierarchy Miner* e *Social Network Miner* [Song and van der Aalst 2006; da Costa Alves 2010]. O algoritmo *Organizational Miner* revela grupos de trabalho ou papéis a partir da identificação dos participantes que executam a mesma atividade. O algoritmo *Role Hierarchy Miner* [Guo et al. 2008] apresenta as relações hierárquicas entre os papéis descobertos, e apresenta uma matriz de frequência com que cada participante executa uma atividade. O algoritmo *Social Network Miner* revela a forma como as pessoas ou departamentos interagem entre si por meio de sociogramas.

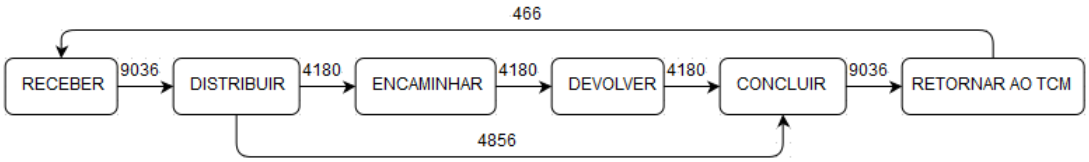


Fig. 3. Controle de fluxo encontrado por meio do algoritmo *Heuristics Miner*

No processo de negócio apresentado acima, início do processo se da pela atividade "RECEBER", enquanto que a atividade final é a "RETORNAR AO TCM". Todo processo deve ser concluído por um procurador, mas em alguns casos, isto não se da de forma direta, ou seja, o procurador encaminha o processo a um assessor para que o mesmo faça suas análises, devolva, e assim, poder ser concluído pelo procurador e por fim, retornado ao TCM. O objetivo é agilizar a conclusão de processos. A relação entre as atividades de "RETORNAR AO TCM" e "RECEBER", é justificado pela necessidade de algumas vezes os processos concluídos outrora requererem novos pareceres, satisfazendo novas condições, por exemplo.

3. RESULTADOS E DISCUSSÃO

A fim de avaliarmos as diferenças entre o processo planejado e o processo revelado pelos dados analisados, realizamos entrevistas com mais de cinco pessoas que conhecem bem o processo de negócio para, assim, comparar o que foi revelado com o que era esperado pela organização.

As análises relacionadas ao controle de fluxo e tempo, revelaram que aproximadamente 51% dos processos são encaminhados, ou seja, pouco mais da metade dos processos passam por assessores e que somente 5,4% retornaram para novo parecer, fato visto com bastante interesse por especialistas.

[2]Mais detalhes sobre a ferramenta no sítio www.promtools.org/promimport

Também foi observado que a média de duração para se concluir um processo, quando o mesmo é feito diretamente por um procurador, ou seja, da atividade "DISTRIBUIR" para "CONCLUIR", é de aproximadamente 13 dias. No entanto, quando o processo é encaminhado a um assessor, essa média aumenta para 43 dias. Num primeiro momento, poderíamos julgar a eficiência dos assessores, considerando o aumento substancial no tempo de conclusão, porém após avaliação com os especialistas no MPTCM-PA, encontramos uma explicação: processos encaminhados geralmente não são prioridades, podem aguardar mais tempo para ser concluído, e em alguns casos o procurador não concorda com a manifestação do assessor e o processo é reenviado informalmente para alterações sem utilizar o sistema para registrar o reenvio para o assessor. Portanto, porque o *log* de dados não registra o reenvio, o tempo médio descoberto não revela uma interação iterativa entre procuradores e assessores. Após análises e entrevistas, foi revelado o motivo pelo qual os procuradores não utilizam o sistema para o reenvio, alegam que o processo de tramitação se da de forma trabalhosa e complexa.

Após análises foi possível detectar outro *loop* invisível no *log* de dados. Este *loop* é a relação entre procuradores e secretários, pois nem todo processo pode ser despachado ou concluído por qualquer procurador, por exemplo, quando há conflito de interesses. Nestes casos, o processo deveria ser "REDISTRIBUÌDO", ou seja, o procurador deveria devolver o processo para que a secretaria pudesse redistribuir a outro procurador, porém esta prática não consta no *log* de dados, apesar de ser sabido que eventos deste tipo tenham sido ocorridos.

Considerando a perspectiva organizacional, observamos a ocorrência de 25.957 tarefas executadas para apenas um secretário, equivalente a 43% de todo o *log* de dados. Porém, os avaliadores de nossa análise afirmam que esta informação não é verdadeira. Após análises, foi revelado que o motivo pelo qual este secretário teve uma aparente sobrecarga de serviço está relacionado a erros no processo de migração de bases de dados do sistema legado. Além disso, após identificação automática dos grupos existentes na organização, revelamos um acordo informal entre secretários sobre divisão de tarefas, ou seja, nem todos estavam executando todas as atividades relacionadas a secretaria, no caso, "RECEBER", "DISTRIBUIR" e "RETORNAR AO TCM". Outra informação importante revelada com a análise foi que participantes de outros setores realizaram atividades relacionadas à secretaria. Isto ocorreu pela insuficiência em determinados momentos de secretários, seja por motivo de férias, licença médica, faltas, entre outras razões. Notamos ainda que tal exceção não foi observada com os grupos *Assessores* e *Procuradores*, pois são grupos que exigem habilidades e competências que não são facilmente substituíveis. Revelamos também as relações de subcontratação entre os participantes, assim, foi possível identificar quais procuradores encaminham mais processos aos assessores e quais apresentam maior desempenho quanto à conclusão direta de processos. Outro ponto interessante foi a identificação de participantes "invisíveis" no *log* de dados.

Especialistas no negócio ficaram satisfeitos com os resultados apresentados, principalmente na identificação de processos errôneos no *log* de dados advindos de erros do processo de migração dos dados do sistema legado para o novo. A identificação desses processos anômalos minimizou esforços e melhorou o planejamento do setor de informática do órgão, que poderia ser acionado pelos usuários do sistema quando os mesmos precisassem corrigir essas instâncias. Além disso, corrigir essas ocorrências inconsistentes no banco de dados melhorou a confiabilidade dos dados persistidos, requisito fundamental para diminuir a rejeição e aumentar a confiabilidade do sistema, e ainda, um panorama do processo de negócio tal como executado na realidade foi revelado e o setor de informática do MPTCM-PA passou a tomar decisões com base em resultados reais, por exemplo, orientando os usuários quanto ao registro obrigatório da atividade "REDISTRIBUIR", melhorias quanto à interface do sistema para promoção da utilização da funcionalidade de reenviar um processo, e controle de exceções, pois identificamos participantes invisíveis no *log*.

4. CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho relatou a experiência da aplicação de técnicas de *Process Mining* como instrumento automático de extração de informações em processos bem estruturados para análise de conformidade. Para tanto, usamos o *log* de dados gerado pelo sistema de informação do MPTCM-PA, onde foi possível observar algumas exceções quanto ao fluxo de execução e as relações de trabalho entre os participantes. A identificação prévia dessas exceções promoveu maior suporte para a adoção do novo sistema, pois os erros identificados foram corrigidos antes de reportado por usuários.

Apesar de relatos de instâncias de processos excepcionais, observamos um elevado grau de conformidade nos dados. Mesmo em caso de total alinhamento, esta avaliação se torna extremamente valiosa, ao garantir confiança no modelo. Além disso, aplicar *Process Mining* evita o esforço do uso das complexas consultas ao banco de dados, que levariam semanas para extrair informações semelhantes a respeito do processo de negócio. Finalmente, a experiência com os dados do MPTCM-PA foi importante para notarmos a aplicabilidade da análise de conformidade como instrumento de avaliação da migração de dados entre sistemas. Desta forma, esforços futuros poderão ser inclinados em definir um procedimento, método ou algoritmo de validação de dados migrados entre sistemas de informação, para tanto, utilizando como base técnicas e ferramentas de *Process Mining*. A incerteza sobre este processo de definição de um procedimento à testabilidade científica é instigador, árduo e necessário ao avanço do campo teórico.

Agradecimentos

Agradecemos a Dra. Elisabeth Salame, Procuradora-chefe do MPTCM-PA na época do desenvolvimento deste trabalho, que gentilmente autorizou a publicação deste trabalho, por acreditar que outras instituições públicas poderiam se beneficiar dos resultados apresentados aqui. Agradecemos também a todos do setor de informática que colaboraram com análises e sugestões.

REFERENCES

Belluzzo, R. Inteligencia, informação e conhecimento em corporações. *Revista Brasileira de Biblioteconomia e Documentação* 3 (1), 2007.

da Costa Alves, C. S. *Social Network Analysis for Business Process Discovery*. M.S. thesis, Instituto Superior Técnico - Universidade Técnica de Lisboa, 2010.

Encarnação, I., Junior, E., Lopes, C., and Bezerra, F. Gerenciamento e controle da tramitação de processos de contas: Experiencia do mptcm-pa. In *Escola Regional de Informática Norte 2*, 2012.

Esposito, P. M., Vaz, M. A., de Souza, J. M., and Terres, L. Uma abordagem para a mineração dos processos de uma universidade. In *V Workshop on Business Process Management*. Salvador, BA, pp. 485 – 492, 2011.

Francisco, R. and Santos, E. A. P. Aplicação da mineração de processos como uma prática para a gestão do conhecimento. In *V Workshop on Business Process Management*. Salvador, BA, pp. 447 – 484, 2011.

Guo, Q., Vaidya, J., and Atluri, V. The role hierarchy mining problem: Discovery of optimal role hierarchies. In *Annual Computer Security Applications Conference*, 2008.

Hornix, P. *Performance Analysis of Business Processes through Process Mining*. M.S. thesis, Eindhoven University of Technology, 2007.

Moraes, A. *Constituição do Brasil Interpretada e Legislação Constitucional*. Editora Atlas, 2003.

Rozinat, A. Conformance testing: Measuring the alignment between event logs and process models, 2005.

Song, M. and van der Aalst, W. Towards comprehensive support for organizational mining. In *Eindhoven University of Technology*. Eindhoven, Netherlands, 2006.

van der Aalst, W. *Process Mining: Discovery, Conformance and Enhancement of Business Processes*. Springer, 2011.

van Dongen, B. *Process Mining and Verification*. Ph.D. thesis, Eindhoven University of Technology, 2007.

van Dongen, B., de Medeiros, A., Verbeek, H., A.J.M.M.Weijters, and van der Aalst., W. The prom framework: A new era in process mining tool support. In *6th International Conference on Applications and Theory of Petri Nets*, 2005.

Weijters, A., van der Aalst, W. M., and de Medeiros, A. A. Process mining with the heuristics miner-algorithm. BETA Working Paper Series WP 166, Eindhoven University of Technology, Eindhoven, Netherlands, 2006.

Descoberta de ruído em páginas da *web* oculta através de uma abordagem de aprendizagem supervisionada

João A. F. Lutz, Carlos A. Heuser

Universidade Federal do Rio Grande do Sul, Brazil
{jaflutz, heuser}@inf.ufrgs.br

Abstract.

Um dos problemas da extração de dados na *web* é a remoção de ruídos existentes nas páginas. Esta tarefa busca identificar todos os elementos não informativos em meio ao conteúdo, como por exemplo cabeçalhos, menus ou propagandas. A presença de ruídos pode prejudicar seriamente o desempenho de motores de busca e tarefas de mineração de dados na *web*. Este trabalho aborda o problema da descoberta de ruídos em páginas da *web* oculta, a parte da *web* que é acessível apenas através do preenchimento de formulários. No processamento da *web* oculta, a extração de dados geralmente é precedida por uma etapa de inserção de dados, na qual os formulários que dão acesso às páginas ocultas são automaticamente ou semi-automaticamente preenchidos. Durante esta fase, são coletados dados do domínio em questão, como os rótulos e valores dos campos. A proposta deste trabalho é agregar este tipo de dados com informações sintáticas dos elementos que compõem a página. É mostrado empiricamente que esta combinação atinge resultados melhores que uma abordagem baseada apenas em informações sintáticas.

Categories and Subject Descriptors: H.3.3 [**Information Search and Retrieval**]: Information filtering

Keywords: *Web* oculta, Recuperação de Informações, Eliminação de Ruídos *Web*

1. INTRODUÇÃO

O crescimento exponencial da *web* gerou uma divisão em diferentes segmentos dentro desta. Denomina-se "*web* visível" a parte da *web* cujo conteúdo é possível acessar diretamente, seja através de motores de busca ou *URLs (Uniform Resource Locators)* estáticas. Porém existe outra porção da *web* chamada "*web* oculta", que é composta de páginas geralmente acessíveis apenas através do preenchimento de formulários, e são renderizadas dinamicamente com informações provenientes de um banco de dados. [Bergman 2001] estima que a *web* oculta é 550 vezes maior do que a *web* visível. A exploração dessa parte oculta é dividida em três fases distintas [Raghavan and Garcia-Molina 2000]: primeiro, é necessário descobrir os pontos de entrada para este conteúdo, ou seja, encontrar formulários que, quando preenchidos corretamente, direcionam o usuário para páginas com conteúdo que normalmente não é acessível via *URLs* ou motores de busca. A segunda fase envolve o uso de técnicas para preencher tais formulários, visto que diferentes preenchimentos levam a diferentes resultados. Após a submissão do formulário, a última fase compreende a extração dos dados da página resultante.

Este trabalho está inserido no contexto da terceira fase, onde os resultados de uma busca devem ser extraídos. Busca-se neste trabalho eliminar o ruído nas páginas *web* resultantes. Ruído em páginas *web* é todo conteúdo que não é informativo nem de interesse do usuário final [Yi et al. 2003]. Entre estes elementos não informativos, podem ser citados banners, propagandas, painéis de navegação, informações de contato, e até mesmo políticas de privacidade. [Gibson et al. 2005] estima que até 50% de todo conteúdo da *web* é composto de algum tipo de ruído. A importância da remoção de ruídos

está no fato de que estas porções de conteúdo prejudicam seriamente a performance de sistemas de recuperação de informações na *web*.

Diversos métodos foram desenvolvidos para alcançar este objetivo, e é possível classificá-los em grupos relativos à maneira como o ruído é identificado e removido. [Cai et al. 2003], [Fernandes et al. 2007], [Li and Ezeife 2006], [Song et al. 2004], [Kovacevic et al. 2002] e [Burget and Rudolfova 2009] utilizam algoritmos baseados em visão, já que todos estes trabalhos classificam o conteúdo da página *web* em ruído ou não baseado nos seus atributos visuais. O segundo grupo, composto pelos trabalhos de [Bar-Yossef and Rajagopalan 2002], [Debnath et al. 2005], [Lin and Ho 2002], [Chen et al. 2006] e [Wang et al. 2008], tenta identificar o ruído procurando por *templates* nas páginas *web*. Este procedimento é feito procurando-se por repetição de conteúdo nestas páginas através de heurísticas bem definidas. O outro grupo, com os trabalhos de [Yi et al. 2003] e [Vieira et al. 2006], tenta identificar o ruído baseado na similaridade estrutural da árvore *DOM (Document Object Model)* das páginas *web*. Por fim, existe um grupo que busca identificar o ruído baseado em propriedades textuais, como [Kohlschütter et al. 2010], [Weninger et al. 2010] e [Sun et al. 2011]. Este último será usado neste trabalho como *baseline*. O grande problema da maior parte destes trabalhos é que todos precisam assumir que as páginas *web* possuem uma estrutura específica, ou precisam procurar por elementos *HTML* especiais.

O objetivo principal deste trabalho, dada sua inserção no contexto da *web* oculta, é melhorar os resultados da remoção de ruídos baseando-se em uma técnica que utiliza a densidade de texto e *links* em uma página *web* para encontrar o ruído [Sun et al. 2011]. O trabalho referido assume que elementos que representam conteúdo informativo possuem uma densidade de texto puro mais alta que o ruído. Elementos ruidosos, por sua vez, possuem um número alto de elementos de formatação, além de muito mais *links* em seu conteúdo. Após a implementação do trabalho como *baseline*, foi possível observar que pode-se obter resultados melhores utilizando dados de fases anteriores da exploração da *web* oculta. Este objetivo é atingido utilizando-se um método de aprendizagem supervisionado para classificar elementos de texto em ruído ou não.

O trabalho está organizado da seguinte maneira: no capítulo 2 é mostrada uma técnica para remoção de ruído utilizada como *baseline*; no capítulo 3, é explicado como é possível melhorar a remoção de ruídos e obter melhores resultados utilizando um método de aprendizagem supervisionado; no capítulo 4, são descritos e comparados os experimentos realizados com ambas as técnicas, utilizando-se diferentes conjuntos de dados; no capítulo 5, são discutidos os resultados encontrados e possibilidades de trabalhos futuros.

2. ELIMINAÇÃO DE RUÍDOS ATRAVÉS DE PROPRIEDADES TEXTUAIS

Diversos trabalhos focaram-se na criação de técnicas automáticas de remoção de ruídos e, através destes, diferentes metodologias foram desenvolvidas. Esta diversidade impede a definição de um padrão claro para este tipo de extração de dados. Um grupo de métodos, composto entre outros por [Kohlschütter et al. 2010] e [Weninger et al. 2010], busca identificar o ruído baseado em propriedades textuais contidas nas páginas *web*. Neste trabalho, foi implementada e utilizada como *baseline* a técnica descrita em [Sun et al. 2011]. Ela assume que o conteúdo principal da página possui mais elementos de texto do que elementos de formatação. Já um trecho ruidoso, por sua vez, seria composto de um número maior de elementos de formatação, além de possuir textos mais curtos e mais *links*. São definidas então duas medidas, chamadas de Densidade de Texto (DT_i) e Densidade de Texto Composta (DTC_i). Estas medidas funcionam da seguinte maneira: após realizar o parsing dos elementos do código *HTML* e representar a página *web* como uma árvore, todos os comentários, scripts e elementos de estilo são removidos. Então, para cada nodo da árvore resultante, são contados o número de caracteres abaixo deste nodo, juntamente com o número de *tags*. A proporção entre estes dois parâmetros é chamada de Densidade de Texto, ou seja, quanto maior o número de caracteres existentes em uma ramificação da árvore *DOM*, maior é a probabilidade desta sub-árvore representar

conteúdo relevante. Se o número de *tags* for muito alto, a Densidade de Texto possuirá um valor baixo, o que significa que a taxa de formatação é alta e a probabilidade desta ramificação ser ruído também será alta.

Apesar de ser um bom indicativo, esta medida pode ser acrescida de mais informações, como a quantidade de *links*, por exemplo. Isto é possível devido ao fato da maior parte de ruídos em páginas *web* serem consituídos de *links* para outras páginas. Assim, outra medida é criada, a Densidade de Texto Composta, que incorpora o número de caracteres de *links* (CL_i) em cada sub-árvore, juntamente com o número de elementos de *links* (TL_i) existentes em cada ramificação. A Densidade de Texto Composta pode ser calculada através da fórmula abaixo:

$$DTC_i = \frac{C_i}{T_i} \log_{\ln(\frac{C_i}{\neg CL_i} CL_i + \frac{CL_b}{C_b} C_i + e)} (\frac{C_i}{CL_i} \frac{T_i}{TL_i})$$

Além do número de *links*, a fórmula é composta também pelo número de caracteres que não são *links* ($\neg CL_i$), pelo número total de caracteres de *links* abaixo da *tag* <body> (CL_b), e pelo número de caracteres de texto abaixo de <body> (C_b). Quando qualquer denominador é zero, tal valor é ajustado para um. Nesta fórmula, observa-se a existência de uma proporção entre caracteres de texto e caracteres de *links* ($\frac{C_i}{CL_i}$), assim como uma proporção entre *tags* comuns e *tags* de *links* ($\frac{T_i}{TL_i}$). Contudo, após o cálculo da Densidade de Texto Composta, alguns nodos ruidosos podem receber erroneamente altos valores de densidade, assim como nodos de conteúdo podem receber valores baixos. Desta maneira, nodos de conteúdo acabariam sendo descartados na fase de extração. Para resolver o problema, os autores propuseram outra medida, chamada de Soma das Densidades, que é representada pela soma de todas as Densidades de Texto Compostas dos descendentes de um determinado elemento.

3. INCORPORANDO EVIDÊNCIAS OBTIDAS EM FASES ANTERIORES DA EXPLORAÇÃO DA *WEB* OCULTA

Visto que este trabalho está inserido no contexto da *web* oculta, é possível melhorar os resultados da remoção de ruídos fazendo uso de dados obtidos em fases anteriores da exploração da *web* oculta. Para isto, são coletados todos os rótulos e valores utilizados para preencher os campos de todos os formulários submetidos. Este conjunto de termos representa uma grande *query* de um domínio para ser buscada nas páginas *web* renderizadas dinamicamente. Estes novos dados são coletados a partir do trabalho de [Kantorski et al. 2012], que explora técnicas de preenchimento automatico de formulários, os quais representam pontos de entrada da *web* oculta.

Assim, para cada nodo da árvore *DOM* da página *web* resultante, são criadas novas evidências baseadas nos dados previamente coletados. Uma destas evidências é a *QueryScore*, que representa a soma do número de vezes que um termo da *query* é encontrado em algum dos nodos descendentes, divido pela distância relativa. Essa distância é o número de nodos existente entre o nodo descendente que contém o texto até o nodo ancestral que está sendo calculado. Esta medida será relevante para todos os nodos que contém não apenas texto, mas para nodos contendo texto e outras sub-árvores. Também são utilizadas evidências mais simples, como o número simples de correspondências entre o texto diretamente abaixo do nodo e a *query* (gerada a partir dos rótulos e valores), e a distância absoluta entre o nodo sendo processado e a *tag* <body> do documento.

A fim de resolver o problema de agregação entre as medidas criadas na *baseline* (Densidade de Texto Composta e Soma das Densidades) e estas novas evidências, pode ser utilizada uma abordagem de aprendizagem de máquina. Empregando um classificador do tipo árvore de decisão, é possível classificar os nodos de texto em ruído ou não. Desta forma, para cada conjunto de dados é criado um arquivo contendo 6 atributos: Densidade de Texto Composta, Soma das Densidades, *QueryScore*, correspondências simples e distância absoluta. A última coluna do arquivo indica se um nodo é ruído ou não, verificado juntamente à avaliação manual de cada código *HTML* resultante.

Tabela I. Resultados da *baseline* e da abordagem supervisionada (TP) com conjunto de dados da *web* oculta

Conjunto de dados	Precisão-baseline	Revocação-baseline	Precisão-TP	Revocação-TP	F1-TP
e4s	38.09%	47.75%	98.3%	98.4%	98.3%
missing	60.28%	90.42%	99.7%	99.7%	99.6%
pubs	23.34%	56.64%	99.8%	99.8%	99.8%
drinkgenius	41.82%	65.32%	99.8%	99.8%	99.8%
onlinerace	25.33%	17.06%	99.2%	99.2%	99.2%
posteritati	00.34%	23.46%	99.7%	99.7%	99.6%
book	89.05%	88.97%	97.5%	98.3%	97.5%
career	56.85%	83.55%	98.7%	98.5%	98.5%
mldb	14.91%	36.53%	96.6%	95.6%	95.9%

Para classificar os nodos em ruído ou não, é utilizado um classificador de árvore de decisão, implementado no algoritmo C4.5. Esse método possui duas fases, a fase de treinamento e a fase de teste. A entrada do algoritmo, o arquivo com os atributos previamente apresentado, é dividido em dois grupos. O primeiro grupo é empregado na fase de treinamento, e a partir deste conjunto é construída uma árvore onde cada folha é um possível valor de classe gerado pelo classificador. Um caminho completo da árvore descreve as decisões feitas baseadas nos atributos de cada nodo para, depois de seguir o caminho, chegar a um resultado, neste caso ruído ou não. Na segunda fase, cada nodo do grupo de teste é experimentado contra esta árvore, e verificado se o resultado é correto ou não. Desta maneira é possível obter a precisão e revocação do método proposto.

4. EXPERIMENTOS

Para validar a hipótese de que o uso de informações da *web* oculta pode auxiliar a identificação de ruído em páginas *web*, implementou-se a solução proposta em [Sun et al. 2011]. Primeiramente, utilizou-se os conjuntos de dados fornecidos pelos autores da técnica em seu trabalho, e obteve-se os valores indicados no trabalho em questão. O conteúdo das páginas deste conjunto foi extraído e avaliado manualmente pelos autores da *baseline*, e todos estes dados foram coletados do *website* dos autores. Introduziu-se então neste trabalho outros conjuntos de dados, um grupo restrito dos dados utilizados em [Kantorski et al. 2012]. Os conjuntos de dados utilizados são: *e4s*, uma ferramenta de busca de empregos para estudantes; *missing*, um formulário para busca de pessoas desaparecidas; *pubs*, um motor de busca de bares; *drinkgenius*, um formulário para procura de receitas e drinques; *onlinerace*, uma busca de resultados de corridas de carro; *posteritati*, uma ferramenta para procura de cartazes de filmes; *book*, uma busca de livros à venda; *career*, uma ferramenta para procura de empregos e *mldb*, uma busca em um banco de dados de música e letras. Nestes formulários de entrada para a *web* oculta, diversas combinações possíveis de preenchimento são realizadas, já que cada preenchimento leva a uma página diferente, resultante de uma consulta em um banco de dados. Após a submissão de cada uma destas combinações, foi possível executar a implementação da *baseline* com o código *HTML* resultante. Como estes dados não estavam avaliados, selecionou-se um grupo de páginas resultantes após a submissão dos formulários para a avaliação manual, definindo quais elementos fazem parte do conteúdo e quais são ruído. O conteúdo correto foi então extraído e armazenado em um arquivo separado, para os testes a seguir.

Optou-se por utilizar uma abordagem de aprendizagem de máquina, de maneira que fosse possível combinar as medidas criadas na *baseline* (Densidade de Texto Composta e Soma das Densidades) com as evidências derivadas dos termos coletados no preenchimento dos formulários. Tais evidências são constituídas pelo que foi chamado de *QueryScore*, que leva em consideração o número de correspondências entre termos da *query* e termos abaixo de um nodo, junto com todos seus descendentes e a distância relativa ao nodo sendo calculado. Além disso, também adicionou-se o número de correspondências entre a *query* e termos diretamente abaixo do nodo sendo analisado. Por fim, adicionou-se a distância absoluta do nodo da árvore até o elemento <body>.

Para cada conjunto de dados, agrupou-se todos os nodos de todas as páginas *HTML* resultantes, juntamente com o valor resultante da avaliação manual que indica se tal nodo é conteúdo ou ruído. Com esta informação, foi possível utilizar uma abordagem supervisionada de aprendizagem para determinar se um nodo é ruído ou não. Para esta tarefa, utilizou-se a ferramenta de mineração Weka [Hall et al. 2009], que permite ao usuário realizar uma limpeza no conjunto de dados e possui diversos algoritmos de aprendizagem de máquinas para serem aplicados. Utilizando o algoritmo de árvore de decisão J48 (implentação Java do algoritmo de classificação C4.5), os dados foram divididos em dois grupos: o primeiro, que possui 66% dos nodos *HTML* e é utilizado para a fase de treinamento do algoritmo, e o segundo, contendo o restante dos nodos a serem utilizados para a fase de teste. Como existiam muito mais nodos ruído, o resultado da fase de teste é uma média ponderada dos resultados de cada categoria (ruído ou não), levando-se em consideração os resultados de precisão e revocação.

Para avaliar os resultados da implementação da *baseline*, foi utilizado um subconjunto das métricas contidas em [Sun et al. 2011], entre elas a Precisão, Revocação, e F1. O conteúdo extraído é comparado ao conteúdo correto, da seguinte maneira: para calcular a precisão, encontra-se a proporção entre o tamanho da substring comum mais longa ($LCS(a, b)$) de ambos os resultados e do tamanho do conjunto de texto extraído. Para calcular a revocação, busca-se a proporção entre a substring comum mais longa entre os resultados e o tamanho do conjunto de texto correto. Desta maneira, é possível calcular a F1, precisão média e revocação média para cada conjunto de dados. Testando a implementação da *baseline* com os conjuntos de dados dos autores, obteve-se resultados semelhantes aos originais descritos no artigo *baseline*.

Entretanto, ao executarmos a implementação da *baseline* com o conjunto de dados da *web* oculta, os resultados foram muito abaixo dos resultados fornecidos pelos autores em [Sun et al. 2011]. Isto provavelmente acontece pelo fato da técnica não estar preparada para lidar com páginas contendo dados tabulares em sua grande maioria. Então torna-se clara a necessidade de uma outra abordagem para lidar com este tipo de páginas *web*, contendo grandes porções de dados de forma tabulada. A segunda metade da tabela I mostra a execução do algoritmo de árvore de decisão utilizando dados da *web* oculta incorporados como evidência, e deixa claro a melhora obtida. Observando-se a matriz de confusão resultante do algoritmo de classificação, pode-se perceber que os resultados da detecção de ruídos foram positivos, já que foi possível classificar nodos ruidosos com altos valores de precisão e revocação. Todos os conjuntos de dados obtiveram valores de precisão e revocação acima de 95%. A matriz de confusão mostra que a técnica desenvolvida atinge bons resultados para detectar ruído, que é o objetivo principal deste trabalho, mas deixa a desejar para classificar nodos de conteúdo em si.

5. CONCLUSÃO

Neste trabalho, aborda-se o problema da eliminação de ruídos em páginas *web*. Este ruído, composto por todos os tipos de elementos não informativos ao usuário final, pode afetar negativamente o desempenho de motores de busca, índices *web*, e mineração de dados, como classificação e clusterização. Propõe-se neste trabalho a incorporação de novas medidas às já existentes em [Sun et al. 2011], utilizando dados obtidos a partir da segunda fase da exploração da *web* oculta, como os rótulos e valores dos preenchimentos dos formulários que dão acesso às páginas renderizadas dinamicamente. Acrescendo estas novas evidências, é possível observar uma grande melhora dos resultados para as páginas da *web* oculta através da utilização de um algoritmo de aprendizagem de máquina. Com uma abordagem supervisionada, um algoritmo classificador de árvore de decisão pode obter resultados médios de precisão e revocação acima de 95%, mostrando a melhoria gerada.

Após os experimentos executados este trabalho, é possível identificar espaço para muitas melhorias nos resultados. A melhor opção seria focar na melhoria da descoberta do conteúdo, não apenas do ruído, já que os resultados da descoberta de ruído foram suficientemente bons, mas não a busca pelo conteúdo em si. Para atingir este objetivo, seria possível testar muitos algoritmos de classificação diferentes com múltiplas escolhas de parâmetros, levando a um melhor entendimento das mudanças

dessas propriedades na descoberta do ruído. Ainda poderiam ser explorados outros tipos de conjuntos de dados, não apenas aqueles resultantes de submissão de formulários. Estas experimentações parecem vitais para a descoberta de novas possibilidades em relação ao tema abordado.

REFERENCES

Bar-Yossef, Z. and Rajagopalan, S. Template detection via data mining and its applications. In *Proceedings of the 11th international conference on World Wide Web*. WWW '02. ACM, New York, NY, USA, pp. 580–591, 2002.

Baroni, M., Chantree, F., Kilgarriff, A., and Sharoff, S. Cleaneval: a competition for cleaning webpages.

Bergman, M. K. The Deep Web: Surfacing Hidden Value. *JEP: The Journal of Electronic Publishing* 7 (1), 2001.

Burget, R. and Rudolfova, I. Web page element classification based on visual features. In *Intelligent Information and Database Systems, 2009. ACIIDS 2009. First Asian Conference on*. pp. 67 –72, 2009.

Cai, D., Yu, S., rong Wen, J., ying Ma, W., Cai, D., Yu, S., rong Wen, J., and ying Ma, W. 1 vips: a vision-based page segmentation algorithm, 2003.

Chen, L., Ye, S., and Li, X. Template detection for large scale search engines. In *Proceedings of the 2006 ACM symposium on Applied computing*. SAC '06. ACM, New York, NY, USA, pp. 1094–1098, 2006.

Debnath, S., Mitra, P., and Giles, C. L. Automatic extraction of informative blocks from webpages. In *Proceedings of the 2005 ACM symposium on Applied computing*. SAC '05. ACM, New York, NY, USA, pp. 1722–1726, 2005.

Fernandes, D., de Moura, E. S., Ribeiro-Neto, B., da Silva, A. S., and Goncalves, M. A. Computing block importance for searching on web sites. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*. CIKM '07. ACM, New York, NY, USA, pp. 165–174, 2007.

Gibson, D., Punera, K., and Tomkins, A. The volume and evolution of web page templates. In *Special interest tracks and posters of the 14th international conference on World Wide Web*. WWW '05. ACM, New York, NY, USA, pp. 830–839, 2005.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. The weka data mining software: an update. *SIGKDD Explor. Newsl.* 11 (1): 10–18, Nov., 2009.

Kantorski, G. Z., Moraes, T. G., Moreira, V. P., and Heuser, C. A. Choosing values for text fields in web forms. In *ADBIS (2)*. pp. 125–136, 2012.

Kohlschütter, C., Fankhauser, P., and Nejdl, W. Boilerplate detection using shallow text features. In *Proceedings of the third ACM international conference on Web search and data mining*. WSDM '10. ACM, New York, NY, USA, pp. 441–450, 2010.

Kovacevic, M., Diligenti, M., Gori, M., and Milutinovic, V. Recognition of common areas in a web page using visual information: a possible application in a page classification. In *Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on*. pp. 250 – 257, 2002.

Kushmerick, N. Learning to remove internet advertisements. In *Proceedings of the third annual conference on Autonomous Agents*. AGENTS '99. ACM, New York, NY, USA, pp. 175–181, 1999.

Li, J. and Ezeife, C. Cleaning web pages for effective web content mining. In *Database and Expert Systems Applications*, S. Bressan, J. Kng, and R. Wagner (Eds.). Lecture Notes in Computer Science, vol. 4080. Springer Berlin / Heidelberg, pp. 560–571, 2006.

Lin, S.-H. and Ho, J.-M. Discovering informative content blocks from web documents. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. KDD '02. ACM, New York, NY, USA, pp. 588–593, 2002.

Raghavan, S. and Garcia-Molina, H. Crawling the hidden web. Technical Report 2000-36, Stanford InfoLab, 2000.

Song, R., Liu, H., Wen, J.-R., and Ma, W.-Y. Learning block importance models for web pages. In *Proceedings of the 13th international conference on World Wide Web*. WWW '04. ACM, New York, NY, USA, pp. 203–211, 2004.

Sun, F., Song, D., and Liao, L. Dom based content extraction via text density. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. SIGIR '11. ACM, New York, NY, USA, pp. 245–254, 2011.

Vieira, K., da Silva, A. S., Pinto, N., de Moura, E. S., Cavalcanti, J. a. M. B., and Freire, J. A fast and robust method for web page template detection and removal. In *Proceedings of the 15th ACM international conference on Information and knowledge management*. CIKM '06. ACM, New York, NY, USA, pp. 258–267, 2006.

Wang, Y., Fang, B., Cheng, X., Guo, L., and Xu, H. Incremental web page template detection by text segments. *Semantic Computing and Systems, IEEE International Workshop on* vol. 0, pp. 174–180, 2008.

Weninger, T., Hsu, W. H., and Han, J. Cetr: content extraction via tag ratios. In *Proceedings of the 19th international conference on World wide web*. WWW '10. ACM, New York, NY, USA, pp. 971–980, 2010.

Yi, L., Liu, B., and Li, X. Eliminating noisy information in web pages for data mining. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. KDD '03. ACM, New York, NY, USA, pp. 296–305, 2003.

Efficient Entity Matching over Multiple Data Sources with MapReduce

Demetrio Gomes Mestre, Carlos Eduardo Pires

Universidade Federal de Campina Grande, Brazil
demetriogm@gmail.com, cesp@dsc.ufcg.edu.br

Abstract. The execution of data-intensive tasks such as entity matching on large data sources has become a common demand in the era of Big Data. To face this challenge, cloud computing has proven to be a powerful ally to efficient parallel the execution of such tasks. In this work we investigate how to efficiently perform entity matching over multiple large data sources using the MapReduce programming model. We propose MSBlockSlicer, a MapReduce-based approach that supports blocking techniques to reduce the entity matching search space. The approach utilizes a preprocessing MapReduce job to analyze the data distribution and provides an improved load balancing by applying an efficient block slice strategy as well as a well-known optimization algorithm to assign the generated match tasks. We evaluate our approach against an existing one that addresses the same problem on a real cloud infrastructure. The results show that our approach increases significantly the performance of distributed entity match task by reducing the amount of generated data from the map phase and minimizing the execution time.

Categories and Subject Descriptors: H.2.4 [**Systems**]: Distributed Databases; H.3.4 [**Systems and Software**]: Distributed Systems

Keywords: Entity Matching, LoadBalancing, MapReduce, Multiple Data Sources

1. INTRODUCTION

Distributed computing has received a lot of attention lately to perform high data-intensive tasks. Extensive powerful distributed hardware and service infrastructures capable of processing millions of these tasks are available around the world. Programming models have being created to make efficient use of such cloud environments. In this context, MapReduce (MR) [Dean and Ghemawat 2008], a well-known programming model for parallel processing on cloud infrastructures, emerges as a major alternative for the efficient distributed data-intensive tasks.

Entity Matching (EM) (also known as entity resolution, deduplication, or record linkage) is a data-intensive and performance critical task and demands studies on how it can benefit from cloud computing. EM is applied to determine all entities (duplicates) referring to the same real world object given a set of data sources [Kopcke and Rahm 2010]. The task has critical importance for data cleaning and integration, e.g., to find duplicate product descriptions in databases.

Two common situations can be found when dealing with EM over data sources, the single and the multiple data sources matching. The first one refers to find all the duplicates in a single data source (traditional) and the second one, focus of this work, refers to the special case of find the duplicates between two or more data sources [Kopcke and Rahm 2010]. Both situations share the main problem that makes EM heavy to perform, the need of applying matching techniques on the Cartesian product of all input entities (naive) leading to a quadratic complexity of $O(n^2)$. For large datasets, the application of such approach is very ineffective.

To minimize the workload caused by the Cartesian product execution and to maintain the match quality, techniques like blocking [Baxter et al. 2003] become necessary. Such techniques work by partitioning the input data into blocks of similar entities and restricting EM to entities of the same block. For instance, it is sufficient to compare entities of the same manufacturer when matching product offers.

Nevertheless, even using blocking techniques, EM remains hard to process for large datasets [Kolb et al. 2012a]. Therefore, EM is an ideal problem to be treated with a distributed solution. The execution of blocking-based EM over single and multiple sources can be done in parallel with the MR model by using several map and reduce tasks. More specifically, the map tasks can read the input entities in parallel and redistribute them among the reduce tasks according to the blocking key. Entities sharing the same blocking key are assigned to a common reduce task and several blocks can be matched in parallel.

However, this simple MR implementation, known as **Basic**, has vulnerabilities. Severe load imbalances can occur due to large blocks (skew problem) occupy a node for a long time and leave the other nodes idle. This is not interesting since there is an urgency to complete the EM process as quickly as possible.

The load imbalance problem when dealing with single sources has been addressed by **BlockSplit** [Kolb et al. 2012a], a general load balancing MR-based approach that takes the size of blocks into account. More details about this work will be shown in the Related Work section. However, this solution has load imbalances and excessive entity replication issues that undermine its running time. Consequently, **BlockSplit**'s extension that address the multiple data sources problem, also found in [Kolb et al. 2012a], extends the same issues of the single source solution. To overcome these problems, we make the following contributions:

—We propose **MSBlockSlicer** (Multiple Source BlockSlicer), an extension of our **BlockSlicer** approach [Mestre and Pires 2013] (proposed to address the single source problem) that provides a load balancing improvement by applying an efficient block slice strategy over multiple data sources. The approach takes the size of blocks into account and generates match tasks of entire blocks only if this does not violate the load balancing constraints. Larger blocks are sliced into several match tasks to enable a fewer number of comparisons respecting the Cartesian product. A greedy optimization is used to assign match tasks of entire blocks and sliced ones to the proper reduce tasks aiming to optimize the load balancing parallel matching.

—We evaluate **MSBlockSlicer** against **BlockSplit** extension for multiple data sources and show that our approach provides a better load balancing strategy by reducing the amount of data generated from the map phase and diminishing the overall execution time. The evaluation is performed on a real cloud environment and uses real-world data.

2. RELATED WORK

Entity Matching is a very studied research topic. Many approaches have been proposed and evaluated as described in the recent survey [Kopcke and Rahm 2010]. However there are only a few approaches that consider parallel entity matching. The first steps in order to evaluate the parallel Cartesian product of two sources is described in [Kim and Lee 2007]. [Kirsten et al. 2010] proposes a generic model for parallel entity matching based on general partitioning strategies that take memory and load balancing requirements into account.

Few MR-based approaches address the load balancing and skew handling problem. [Okcan and Riedewald 2011] applied a static load balancing mechanism, but it is not suitable due to arbitrary join assumptions. Similarly to our work, the authors employ a previous analysis phase to determine the datasets' characteristics (using sampling) and thereafter avoid the evaluation of the Cartesian

product. This approach focus on data skew handle in the map process output, which leads to an overhead in the map phase and large amount of map output.

MapReduce has already been employed for EM (e.g., [Wang et al. 2010]) but load balancing was not the main focus and only one mechanism of near duplicate detection by the PPjoin paradigm adapted to the MapReduce framework can be found. [Kolb et al. 2012b] studies load balancing for Sorted Neighborhood (SN). However, SN follows a different blocking approach (fixed window size) that is by design less vulnerable to skewed data. [Vernica et al. 2010] shows another approach for parallel processing entity matching on a cloud infrastructure and an extension that address the multiple data source problem. This study explains how a single token-based string similarity function performs with MR. However, this approach suffers from load imbalances because some reduce tasks process more comparisons than the others.

As mentioned in the introduction, [Kolb et al. 2012a] addresses our problem. The authors present two approaches, **BlockSplit** and **PairRange**. **BlockSplit** is an algorithm that processes small blocks within single match tasks. The larger blocks are split according to the m input partitions into m sub-blocks obeying an appropriate scheme of entity replication (based on m input partitions). Each sub-block is processed by a single match task. **PairRange** on the other hand implements a virtual enumeration of all entities and the relevant comparisons (pairs) based on the data distribution provided by the so-called BDM, Block Distribution Matrix, aiming to send entities to all reduce tasks according to the relevant comparisons belonging to ranges previously established based on the average workload. **PairRange** neither consider input partitions nor blocks, but instead ranges of comparisons. Despite **PairRange** provide a little more uniform distribution than **BlockSplit**, it generates too much entity replication (for load balancing purposes) which leads to an extra overhead. According to [Kolb et al. 2012a], this overhead leads **PairRange** to perform equals or worse than **BlockSplit**. In addition, such overhead can lead to lack of memory problems in the cloud infrastructure when executing **PairRange** for huge datasets. For this reason, we have only implemented **BlockSplit** and compare it with our work in an experimental evaluation.

3. GENERAL MR-BASED ENTITY MATCHING WORKFLOW FOR LOAD BALANCING

To perform our load balancing optimization for EM processing over multiple data sources, we need two MR jobs.

3.1 First Job: Block Distribution Matrix

The Block Distribution Matrix (BDM) is a simple preprocessing step to determine some datasets' characteristics. As described in [Kolb et al. 2012a], the BDM consists in a b x m matrix that specifies the number of entities of b blocks across m input partitions.

3.2 Second Job: Improving Block-based Load Balancing

The second job, denoted by **MSBlockSlicer**, performs our load balancing optimization approach for EM over multiple data sources. Since the EM of multiple data sources problem can be simplified to the pairwise comparisons of all envolved data sources, this section describes an extension of **BlockSlicer** [Mestre and Pires 2013] for matching two sources R and S.

As shown in the example data of Figure 1, we utilize the entities A-N and the blocking keys w-z. Each entity belongs to one of the two sources R and S. Source R is stored in the partition Π_0 and source S is stored in the partitions Π_1 and Π_2. The sources are automatic partitioned according to the number of available map tasks.

The BDM computation is the same but adds a source tag to the map output key aiming to identify

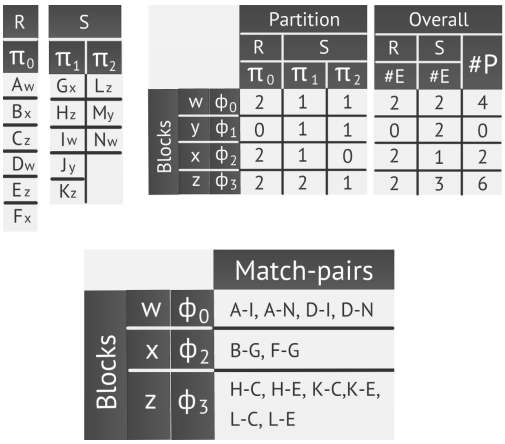


Fig. 1. Example data (up-left), BDM (up-right), and the expected match-pairs (down-center).

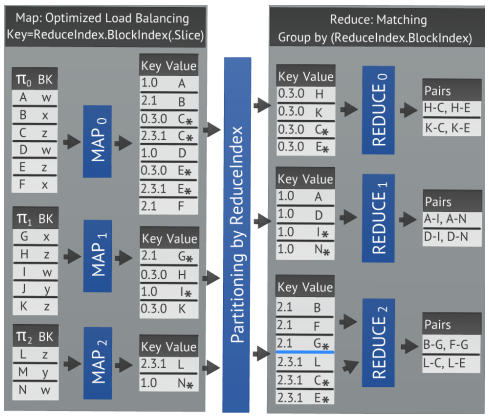


Fig. 2. Example BlockSlicer dataflow for 2 sources.

blocks with the same key in different sources, i.e., $\Phi_{i,R}$ and $\Phi_{i,S}$. The BDM has the same structure for the one-source case but distinguishes between the two sources for each block (see Figure 1).

3.2.1 *MSBlockSlicer Load Balancing.* The **BlockSlicer** approach for two sources (**MSBlockSlicer**) follows the same scheme as for one source, but has two main differences. Firstly, given a block B, the sliced-block sb consists in allowed entities $B_{allowed}$ belonging to the same source (R or S) and the not allowed blocks $B_{notAllowed}$ belonging to the other source. $B_{allowed}$ is extracted from the source that contains the largest number of entities from B. Secondly, the entities belonging to $B_{allowed}$ will no longer iterate comparisons among themselves, only with the entities belonging to $B_{notAllowed}$. In this case, if the permutation of $B_{allowed}$ entities with $B_{notAllowed}$ entities generates a number of pairs above the average workload, then sb is sliced into two new sliced-blocks to enable load balancing. The slicing is processed only on entities belonging to $B_{allowed}$ and the entities belonging to $B_{notAllowed}$ are emitted for each sliced-block generated. The exact slicing is calculated as $\mid B_{allowed} \mid = \frac{\lceil T/r \rceil}{\mid B_{notAllowed} \mid}$, where T is the number of the comparisons generated and r is the number of reduce tasks available.

After the slicing phase, the remaining entities of $B_{allowed}$ are submitted to a new average workload constraint verification. If the number of pairs is still above the average workload, the remaining entities of $B_{allowed}$ are submitted to a new slicing process (recursive procedure).

Figure 2 shows the workflow for the example data of Figure 1. The BDM indicates 12 overall pairs so that the average workload is 4 pairs. The largest block Φ_3 is therefore subject to a slice process due to the number of pairs that must be processed (6 pairs). The slice phase results in two new sliced-blocks (the match tasks 3.0 and 3.1). All match tasks are ordered by the number of pairs: 3.0 (4 pairs, $reduce_0$), 0 (4 pairs, $reduce_1$), 2 (2 pairs, $reduce_2$) and 3.1 (2 pairs, $reduce_2$). The dataflow is shown in Figure 2. Partitioning is based on the reduce task index only, for routing all data to the reduce tasks whereas sorting is done based on the entire key. The reduce function is called for every match task $k.(i)$ and compares entities considering only pairs from different entity types (allowed against not allowed). For illustration, Figure 2 shows that H is an allowed entity from match task 3.0 and is only compared with the not allowed entities (marked with an "*"), e.g., the entities C and E.

4. EVALUATION

In the following, we evaluate[2] **MSBlockSlicer** against **BlockSplit**'s extension approach, which was implemented according to the pseudo-code available in [Kolb et al. 2012a], regarding one performance critical factor: the number of available nodes (n) in the cloud environment. In each experiment we evaluate the algorithms aiming to investigate their behavior when dealing with the resources consumption caused by the use of many map and reduce tasks and how they can scale with the number of available nodes.

We ran our experiments on a 10-node HP Pavilion P7-1130 cluster (with Hadoop 0.20.2) and utilized one real-world dataset. The dataset DS1 (DBLP) contains about 1.46 million publication records and was divided into two data sources (R and S) containing about 730,000 entities each. The first three letters of the publication title form the default blocking key. Two entities were compared by computing the Jaro-Winkler [Cohen et al. 2003] distance of their comparing attributes and those pairs with a similarity ≥ 0.7 were regarded as matches.

4.1 Scalability: Number of nodes

As mentioned in the introduction, scalability is important for many reasons and one of them is the financial. The number of nodes should be carefully estimated since distributed infrastructure suppliers usually charge per hired machines even if they are underutilized. To analyze the scalability of the two approaches, we vary the number of nodes from 1 up to 10. Following the Hadoop's documentation, for n nodes, the number of map tasks is set to $m = 2 \cdot n$ and the number of reduce tasks is set to $r = 4 \cdot n$, i.e., adding new nodes leads to additional map and reduce tasks. The resulting execution times values are shown in Figure 3 (DS1) and the respectively number of generated key-value pairs are illustrated in Figure 4.

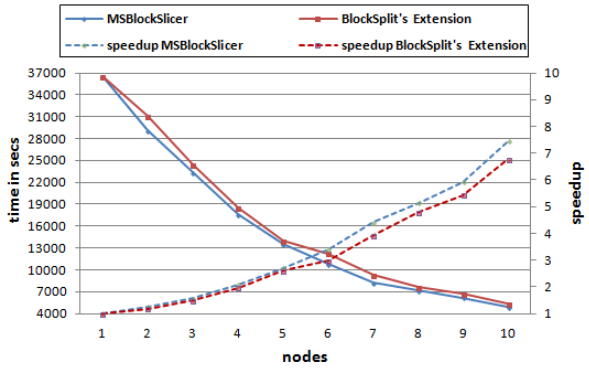


Fig. 3. Execution times and speedup for both approaches using DS1.

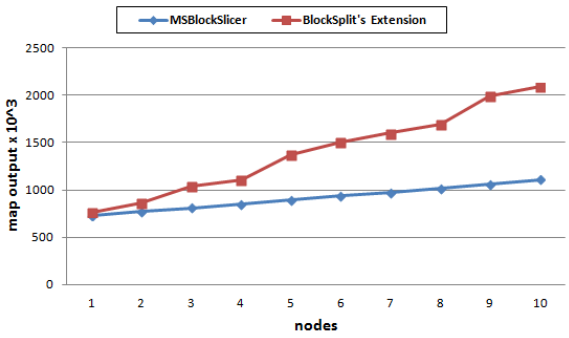


Fig. 4. Number of generated key-value pairs by map for DS1 varying the number of nodes n (m=2·n, r=4·n).

We can note that both **MSBlockSlicer** and **BlockSplit**'s extension scale almost linearly showing their ability to evenly distribute the workload across reduce tasks and nodes. However, due to the difficulties already discussed, i.e., high dependency of the input partitions, **BlockSplit**'s execution time is compromised by the deficiency of properly split larger blocks with few input partitions as we can see in Figure 3 when we vary n from 5 to 6. This deficiency is the main reason of **BlockSplit**'s extension approach to perform around 1000 seconds slower than **MSBlockSlicer** on experiments with 10 nodes (m=20 and r=40). The difference is highlighted by the speedup (it's like **MSBlockSlicer** has almost one extra node working). Furthermore, Figure 4 shows, by the map output generation,

[2]The datasets and codes are available in https://sites.google.com/site/demetriomestre/activities

an overgrowth of output entities during the **BlockSplit**'s map phase. This overgrowth leads to the associated overhead already discussed earlier, i.e, unnecessary data transfer (network resources), sorting large partitions and OS memory management when processing reduce tasks, and thereby causes the deterioration of the execution time. Besides, depending on the entities length or dataset size, such situation can cause serious problems of lack of memory. Note that, for 10 nodes, the amount of output generated by **BlockSplit**'s extension approach is almost twice the amount generated by the **MSBlockSlicer** one. This limits **BlockSplit**'s extension to be used with huge datasets sustainably.

5. SUMMARY AND OUTLOOK

We proposed an improved load balancing approach, **MSBlockSlicer**, for distributed blocking-based entity matching over multiple data sources using a well-known MapReduce framework. The solution is by design efficient to provide load balancing to an entity matching process of huge datasets without depending on the data distribution or order of the input partitions with a significant improved reduction of the information emitted from map to reduce phases (map output). It is able to deal optimally with skewed data distributions and workload among all reduce tasks by slicing large blocks. Our evaluation on a real cloud environment using real-world data demonstrated that **MSBlockSlicer** scale with the number of available nodes without relying on any extra architecture configuration. We compared our approach against an existing one (**BlockSplit**'s extension) and we verified that **MSBlockSlicer** overcomes **BlockSplit**'s extension in performance terms.

In future work, we will investigate how we can improve our solution to also address the horizontal skew problem (entities with disproportionate lengths). Also, we will further investigate how our load balancing approach can be adapted to address other MapReduce-based techniques for different kinds of data-intensive tasks, such as join processing or data mining.

REFERENCES

Baxter, R., Christen, P., and Churches, T. A comparison of fast blocking methods for record linkage. In *ACM SIGKDD '03 Workshop on Data Cleaning, Record Linkage, and Object Consolidation*. pp. 25–27, 2003.

Cohen, W. W., Ravikumar, P., and Fienberg, S. E. A comparison of string distance metrics for name-matching tasks. In *IIWeb*. pp. 73–78, 2003.

Dean, J. and Ghemawat, S. Mapreduce: simplified data processing on large clusters. *Commun. ACM* 51 (1): 107–113, Jan., 2008.

Kim, H.-s. and Lee, D. Parallel linkage. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*. CIKM '07. ACM, New York, NY, USA, pp. 283–292, 2007.

Kirsten, T., Kolb, L., Hartung, M., Gross, A., Kopcke, H., and Rahm, E. Data Partitioning for Parallel Entity Matching. In *8th International Workshop on Quality in Databases*, 2010.

Kolb, L., Thor, A., and Rahm, E. Load balancing for mapreduce-based entity resolution. In *Proceedings of the 2012 IEEE 28th International Conference on Data Engineering*. ICDE '12. IEEE Computer Society, Washington, DC, USA, pp. 618–629, 2012a.

Kolb, L., Thor, A., and Rahm, E. Multi-pass sorted neighborhood blocking with mapreduce. *Comput. Sci.* 27 (1): 45–63, Feb., 2012b.

Kopcke, H. and Rahm, E. Frameworks for entity matching: A comparison. *Data Knowl. Eng.* 69 (2): 197–210, Feb., 2010.

Mestre, D. G. and Pires, C. E. Improving load balancing for mapreduce-based entity matching. In *Proceedings of the Eighteenth IEEE Symposium on Computers and Communications*. ISCC '13. IEEE Computer Society, 2013.

Okcan, A. and Riedewald, M. Processing theta-joins using mapreduce. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*. SIGMOD '11. ACM, New York, NY, USA, pp. 949–960, 2011.

Vernica, R., Carey, M. J., and Li, C. Efficient parallel set-similarity joins using mapreduce. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*. SIGMOD '10. ACM, New York, NY, USA, pp. 495–506, 2010.

Wang, C., Wang, J., Lin, X., Wang, W., Wang, H., Li, H., Tian, W., Xu, J., and Li, R. Mapdupreducer: detecting near duplicates over massive datasets. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*. SIGMOD '10. ACM, New York, NY, USA, pp. 1119–1122, 2010.

Análise de Fatores Impactantes na Recomendação de Colaborações Acadêmicas Utilizando Projeto Fatorial

Michele A. Brandão, Mirella M. Moro, Jussara M. Almeida

Universidade Federal de Minas Gerais, Brasil
{micheleabrandao, mirella, jussara}@dcc.ufmg.br

Resumo. Sistemas de recomendação têm sido muito utilizados em comércio eletrônico e redes sociais de amizade ou trabalho. Dos vários desafios de construir tais sistemas, um ainda pouco explorado é como parametrizar esses sistemas e suas avaliações. Geralmente, cada estratégia de recomendação tem seus parâmetros e fatores que podem ser variados. Neste artigo, propomos avaliar o impacto dos parâmetros principais de duas funções do estado-da-arte que recomendam colaborações acadêmicas e sua forma de avaliação. Nossos resultados mostram que os fatores afetam a revocação, novidade, diversidade e cobertura das recomendações de colaboração de formas diferentes. Por fim, tal avaliação mostra a importância de estudar os fatores e as interações entre eles no contexto de recomendações de colaboração acadêmica.

Categories and Subject Descriptors: H.2 [**Database Management**]: Miscellaneous

Palavras-chave: recomendação de colaboração, estratégia de divisão, projeto fatorial 2^k, fatores

1. INTRODUÇÃO

Sistemas de recomendação têm sido utilizados em comércio eletrônico e redes sociais de amizade ou trabalho. O foco deste trabalho é nas recomendações em redes sociais acadêmicas de coautoria, onde os nós representam pesquisadores e as arestas entre eles são coautoria em publicações. Aqui, a recomendação de colaborações entre pesquisadores é bastante relevante para ajudar pesquisadores a formar novos grupos e a buscar colaborações para projetos de pesquisa, para melhorar a qualidade de comunicação na rede e para investigar comunidades de pesquisa [Brandão et al. 2013]. Ademais, um trabalho recente mostra que grupos de pesquisa com uma rede social de coautoria bem conectada tendem a ser mais produtivos [Lopes et al. 2011]. Dos vários desafios de construir tais sistemas, nosso trabalho foca na avaliação das recomendações.

Sistemas de recomendação podem ser avaliados via mecanismos de feedback do usuário. Porém, no contexto acadêmico, tal feedback é complexo, pois um pesquisador pode avaliar uma recomendação como ruim devido a questões subjetivas tais como não gostar do recomendado, falta de afinidade ou competição. Outra opção é a partir das preferências dos usuários, modelar seu comportamento e usar essa modelagem para avaliar as recomendações [Shani and Gunawardana 2011]. Essa estratégia também não é adequada no cenário acadêmico, pois não há dados públicos com informações das preferências de um pesquisador em relação a outro.

Assim, as recomendações de colaboração têm sido avaliadas dividindo a rede social em duas partes [Brandão et al. 2013; Lopes et al. 2010]. A primeira parte é utilizada para aplicar as funções de recomendação e gerar uma lista com as recomendações. A segunda é utilizada para estabelecer um comparativo entre as colaborações efetivamente realizadas durante aquele período (*ground truth*) e as recomendações. A avaliação das recomendações verifica se as colaborações recomendadas estão presentes no *ground truth*. Dentre os vários desafios para gerar tais recomendações, um ainda pouco explorado é como parametrizar as funções de recomendação assim como a estratégia de avaliação adotada (particularmente no que tange à divisão dos dados em duas partes). É importante analisar como a variação no tamanho dessas divisões assim como nos valores dos parâmetros das funções de recomendação impactam no resultado das recomendações de colaboração.

Para realizar essa análise, consideramos duas funções de recomendação do estado-da-arte: a *Affin* de Brandão et al. [2013] e a *CORALS* de Lopes et al. [2010]. Essas funções de recomendação possuem pesos que podem impactar nas recomendações de colaboração. Entretanto, não foi feita nenhuma análise teórica ou experimental da sensibilidade das funções de recomendação a tais parâmetros e possíveis interações entre eles.

Este trabalho visa analisar o impacto das divisões dos dados definidas assim como dos pesos explorados pelas funções de recomendação. Essa análise consiste da realização de um projeto experimental fatorial 2^k em que cada parâmetro sob análise (chamado fator) pode assumir dois valores (ou níveis). Esse projeto permite a avaliação do impacto relativo de cada fator bem como da interação de múltiplos fatores na variável resposta sendo estudada (a eficácia da recomendação no nosso caso) [Jain 1991]. Note que há interação entre fatores quando o efeito de um fator na variável resposta depende do valor do outro [Lima et al. 2010]. Aqui, consideramos um projeto 2^k com $k = 3$ fatores: a estratégia de divisão do número total de publicações e os dois pesos explorados pelas funções de recomendação. Realizamos projetos separados para as funções *Affin* e *CORALS*, a fim de estudar o impacto relativo de cada fator na eficácia de cada uma das funções.

As recomendações de colaboração são avaliadas em relação a quatro métricas: revocação, novidade, diversidade e cobertura, que são próprias para tal contexto conforme descrito por Brandão and Moro [2012]. Também foram avaliados o número total de recomendações e o número de recomendações corretas retornadas como métricas de avaliação, pois estão relacionadas com a revocação. Ou seja, para cada função de recomendação, foram realizados projetos experimentais 2^k para cada uma destas seis métricas. A análise realizada com o projeto fatorial mostra que os fatores impactam o resultado dessas métricas de diferentes formas. Tal avaliação mostra o potencial de utilizar o projeto fatorial na avaliação de sistemas de recomendação e de outras aplicações dependentes da definição de parâmetros de dados. Este artigo segue com os seguintes tópicos: trabalhos relacionados (Seção 2); funções de recomendação de colaboração do estado-da-arte (Seção 3); fatores e especificação do projeto fatorial (Seção 4); identificação e análise dos fatores impactantes (Seção 5); e conclusões e trabalhos futuros (Seção 6).

2. TRABALHOS RELACIONADOS

É possível recomendar itens e pessoas em redes sociais. Por exemplo, Freyne et al. [2010] e He and Chu [2010] recomendam itens explorando as preferências e interações dos usuários. Já a recomendação de pessoas deve considerar aspectos das conexões sociais [Lopes et al. 2010; Symeonidis et al. 2010]. Em redes sociais acadêmicas, Brandão et al. [2013] e Lopes et al. [2010] apresentam novas funções de recomendação de colaborações. Porém, ambos fixaram os valores dos parâmetros (pesos) e não estudaram como tais valores impactam a eficácia dos métodos, que é o objetivo deste trabalho, conforme detalhado na Seção 3.

Avaliar a qualidade de recomendações não é trivial [Fouss and Saerens 2008]. Pode-se avaliar as recomendações com estratégia *online*, na qual usuários avaliam quão boa elas são, ou *offline*, onde elas são comparadas com um *ground truth*, sem interação dos usuários [Shani and Gunawardana 2011]. Por exemplo, a estratégia *online* tem sido utilizada pela Netflix e Amazon, e a *offline* para avaliar a diversidade na recomendação de livros [Ziegler et al. 2005] e de colaborações acadêmicas [Brandão et al. 2013; Lopes et al. 2010]. A estratégia de avaliação adotada é um dos parâmetros que pode influenciar no resultado das recomendações.

Este trabalho visa quantificar o impacto de diferentes fatores nas colaborações sugeridas pelas funções de recomendação propostas por Brandão et al. [2013] e Lopes et al. [2010]. São considerados como fatores tanto a estratégia de avaliação *offline* adotada (divisão da rede em duas partes) quanto parâmetros específicos das duas funções. Essa análise é feita com uma aplicação do projeto fatorial 2^k [Jain 1991]. Essa técnica tem sido utilizada, por exemplo, como diretriz para o desenvolvimento de um portal *e-government* [Marzoughi et al. 2010], para parametrizar algoritmos de programação genética [Lima et al. 2010] e para investigar a influência da concentração de Fe(III)-oleate, temperatura de reação, tempo e taxa de aquecimento sobre o núcleo de partículas, tamanho das distribuições hidrodinâmicas e magnetização [Lak et al. 2013]. Até onde sabemos, o estudo aqui realizado ainda não foi feito no contexto de recomendação de colaborações acadêmicas.

3. FUNÇÕES DE RECOMENDAÇÃO DE COLABORAÇÕES ACADÊMICAS

Considere as funções de recomendação acadêmica *Affin* [Brandão et al. 2013] e *CORALS* [Lopes et al. 2010], as quais recomendam pares de pesquisadores i e j a colaborarem, conforme as Equações 1 e 2, respectivamente.

$$r_{i,j} = \begin{cases} Iniciar, & \text{if } (Cp_{i,j} = 0) \wedge \\ & (\textit{Affin\_Sc}_{i,j} > \textit{limiar}); \end{cases} \quad (1)$$

$$r_{i,j} = \begin{cases} Iniciar, & \text{if } (Cp_{i,j} = 0) \wedge \\ & (Cr\_Sc_{i,j} > \textit{limiar}); \end{cases} \quad (2)$$

$Cp_{i,j}$ é o peso para quanto o pesquisador i colaborou com outro j, onde valor *zero* indica que tal par ainda não colaborou. O peso $\textit{Affin\_Sc}_{i,j}$ (Equação 1) é uma média ponderada entre os pesos afiliação *Affin* e proximidade social Sc, onde *Affin* quantifica o quanto um pesquisador i colaborou com pessoas da instituição do pesquisador j, e Sc mede o menor caminho entre pares de pesquisadores i e j na rede de colaboração. O peso $Cr\_Sc_{i,j}$ (Equação 2) é uma média ponderada entre os pesos correlação Cr e proximidade social, onde Cr quantifica o quanto pares de pesquisadores i e j têm publicado em áreas de pesquisa em comum.

A função *Affin* recomenda pares de pesquisadores a iniciar colaboração quando $Cp_{i,j}$ é zero e $\textit{Affin\_Sc}_{i,j}$ é maior que um limiar predefinido. Esse limiar é definido de acordo com intervalos que podem seguir uma escala linear, tal como $baixo < 33\%$ e $alto > 66\%$. O valor "baixo" foi escolhido como limiar. Já a *CORALS* recomenda pesquisadores a iniciar colaboração quando $Cp_{i,j}$ é zero e $Cr\_Sc_{i,j}$ é maior que "baixo" (limiar escolhido). O limiar também é um parâmetro das funções de recomendação, mas não foi incluído no projeto fatorial 2^k pelo fato dos pesos terem um maior impacto nas recomendações de colaborações acadêmicas.

4. PROJETO EXPERIMENTAL

O principal objetivo de um projeto experimental é obter a máxima quantidade de informação com o número mínimo de experimentos [Jain 1991]. Existem diversos tipos de projetos experimentais, tais como projeto simples, projeto fatorial completo, projeto fatorial fracionado, entre outros. Neste trabalho, realizamos um projeto fatorial 2^k, onde k representa o número de fatores e 2 o número de níveis que cada fator possui[1]. Os fatores são os parâmetros que afetam o resultado dos experimentos, e os níveis são os valores que cada fator pode assumir. O projeto fatorial 2^k foi escolhido por permitir o estudo do impacto dos fatores e interações entre eles nas variáveis resposta. É importante notar que realizamos o projeto fatorial sem replicação. Em outras palavras, apenas um resultado é produzido para cada configuração definida pelos níveis dos fatores. Isso foi feito porque os algoritmos das funções de recomendação são determinísticos[2]. Assim, apresentamos o conjunto de dados e detalhamos a aplicação do projeto fatorial 2^k no nosso contexto.

Em relação ao conjunto de dados, os experimentos utilizam dados reais obtidos da biblioteca digital DBLP[3]. A rede social acadêmica construída a partir da DBLP contém 629 pesquisadores de 45 instituições brasileiras e suas 13.867 publicações do ano de 1973 a 2012 em periódicos e eventos. Essa rede foi construída com nós dados pelos autores e arestas pela coautoria nas publicações, conforme descrito por Brandão et al. [2013]. Além disso, como o conjunto de dados é dividido em duas partes, então uma rede social acadêmica é construída para cada parte. Nessa divisão, a primeira parte possui publicações de anos inferiores ou iguais à segunda parte, e é utilizada para gerar as recomendações. Já a segunda parte é utilizada para avaliá-las.

Em relação ao projeto fatorial 2^k, os fatores que podem impactar na recomendação são: a divisão do número total de publicações em duas partes e os pesos das funções de recomendação, conforme definido a seguir.

Estratégia de divisão. A rede social acadêmica de coautoria é dividida em duas partes diferentes, conforme amostras na Tabela I, e cada divisão representa um nível possível para este fator. Por exemplo, no segundo nível, 20% dos dados são utilizados para gerar recomendações e possuem publicações com ano inferior ou igual aos 80% dos dados da segunda parte. Como esse fator está diretamente relacionado à estratégia de avaliação, analisar seu impacto significa analisar o impacto da estratégia. Para o projeto fatorial 2^k, escolhemos evitar extremos, tendo os níveis 2 e 8 como representantes dos níveis menor e maior, conforme Tabela II.

[1] Os dois níveis de cada fator são aqui chamados de maior e menor.
[2] No caso de processos aleatórios, um projeto 2^k com replicação poderia ser adotado.
[3] DBLP: http://www.informatik.uni-trier.de/∼ley/db

Tabela I. Estratégia de Divisão

		Primeira Parte		Segunda Parte	
Descrição	**Nível**	**% Dado**	**# Publicação**	**% Dado**	**# Publicação**
menor	1	10%	1.386	90%	12.481
	2	**20%**	**2.773**	**80%**	**11.094**
	3	30%	4.160	70%	9.707
	4	40%	5.546	60%	8.321
igual	5	50%	6.933	50%	6.933
maior	6	60%	8.321	40%	5.546
	7	70%	9.707	30%	4.160
	8	**80%**	**11.094**	**20%**	**2.773**
	9	90%	12.481	10%	1.386

Tabela II. Definição dos Níveis

Fator	**Nível**	**Valor**
Estratégia de Divisão	menor	20% - 80%
	maior	80% - 20%
Pesos	menor	10
	maior	100

Tabela III. Porcentagem de Variação (%)

Variável Resposta	***Affin***	***CORALS***
Revocação	AB: 86,88	A/AB: 96,47
Total	B/C/BC: 69,37	A/BC: 77,59
Corretas	A/B/C/BC: 99,81	A/BC/ABC: 82,23
Novidade	B/C/AB: 91,08	B/C/AB: 78,38
Diversidade	B/C/BC: 81,32	A/AB/AC/BC: 98,89
Cobertura	B/C/BC: 98,55	A/B/C: 96,24

Pesos das funções de recomendação. As funções de recomendação possuem pesos associados que quantificam a relação entre pares de pesquisadores. A *Affin* possui os pesos afiliação e proximidade social, e a *CORALS* possui os pesos correlação e proximidade social (proximidade social é comum às duas). Cada peso representa um fator que pode impactar nas recomendações de colaboração. Esses fatores são numéricos e podem ter diversos níveis. Os valores adotados como menor e maior para cada peso são mostrados na Tabela II.

Especificação do modelo. O projeto fatorial 2^k é realizado para cada função e com três fatores (estratégia de divisão, peso proximidade social e peso afiliação/correlação), ou seja, $k = 3$ sendo necessários $2^3 = 8$ experimentos. Cada fator é representado pelas variáveis x_A, x_B e x_C. Essas variáveis assumem os valores -1 e 1 representando os níveis inferior e superior de cada fator. Intuitivamente, a ideia por trás do projeto é que o valor da variável resposta pode ser regredido nas variáveis x_A, x_B e x_C usando um modelo aditivo não linear apresentado na Equação 3. O y representa a variável resposta do projeto fatorial (revocação, novidade, diversidade, cobertura, total de recomendações e recomendações corretas), cada q é o efeito do fator na variável resposta, q_0 é o comportamento médio da função de recomendação independente do fator. Especificamente, $q_0 = \frac{1}{2^3}(y_1 + y_2 + y_3 + y_4 + y_3 + y_4 + y_5 + y_6 + y_7 + y_8)$, onde cada y_i é obtida pela combinação dos níveis dos fatores. Por exemplo, y_1 é obtido quando os três fatores estão no nível negativo. Por exemplo, q_A é o efeito do fator A em y e q_{AB} é o efeito da interação entre os fatores A e B (Jain [1991] descreve o cálculo desses efeitos).

$$y = q_0 + q_A x_A + q_B x_B + q_C x_C + q_{AB} x_A x_B + q_{AC} x_A x_C + q_{BC} x_B x_C + q_{ABC} x_A x_B x_C \quad (3)$$

O efeito dos fatores e das interações entre eles podem ser positivos ou negativos nas variáveis resposta (um efeito positivo indica correlação positiva, e um efeito negativo indica correlação negativa). Um fator/interação com efeito positivo tem maior impacto na variável resposta quando o nível do fator é o valor superior. Já um fator/interação com efeito negativo afeta significativamente a variável resposta y quando o nível do fator é o valor inferior. Os efeitos são utilizados para calcular a porcentagem de variação dos dados que é explicada por cada fator. A porcentagem de variação captura a importância de cada fator na variável resposta. É importante notar que consideramos os fatores impactantes quando a porcentagem de variação explicada por eles é superior a 10%. Outro limiar pode ser escolhido, dependendo da aplicação e do custo para analisar os fatores.

5. ANÁLISE DOS RESULTADOS

Nesta seção, apresentamos uma análise dos resultados obtidos do projeto fatorial 2^k com o objetivo de quantificar o impacto relativo dos diversos fatores considerados nas funções de recomendação.

Função de Recomendação *Affin*. Nessa função, os fatores/interações possuem efeitos negativos e positivos. Por exemplo, os fatores estratégia de divisão e peso proximidade social possuem um efeito positivo na revocação. Já a interação entre os fatores peso proximidade social e peso afiliação possuem um efeito negativo sobre a diversidade. Esses efeitos são utilizados para calcular as porcentagens de variação explicadas por cada fator/interações nas variáveis resposta, conforme ilustrado na Figura 1.

A Figura 1 mostra que a interação entre os fatores estratégia de divisão (A) e peso proximidade social (B) é responsável pela maior parte da variação na revocação (acima de 80%). Os fatores peso proximidade social (B), peso afiliação (C) e sua interação BC são significativos para o total de recomendações, e os fatores estratégia de divisão (A), peso proximidade social (B), peso afiliação (C) e a interação ABC para a variável resposta recomendações corretas. Para a variável novidade, os fatores peso proximidade social (B), peso afiliação (C) e

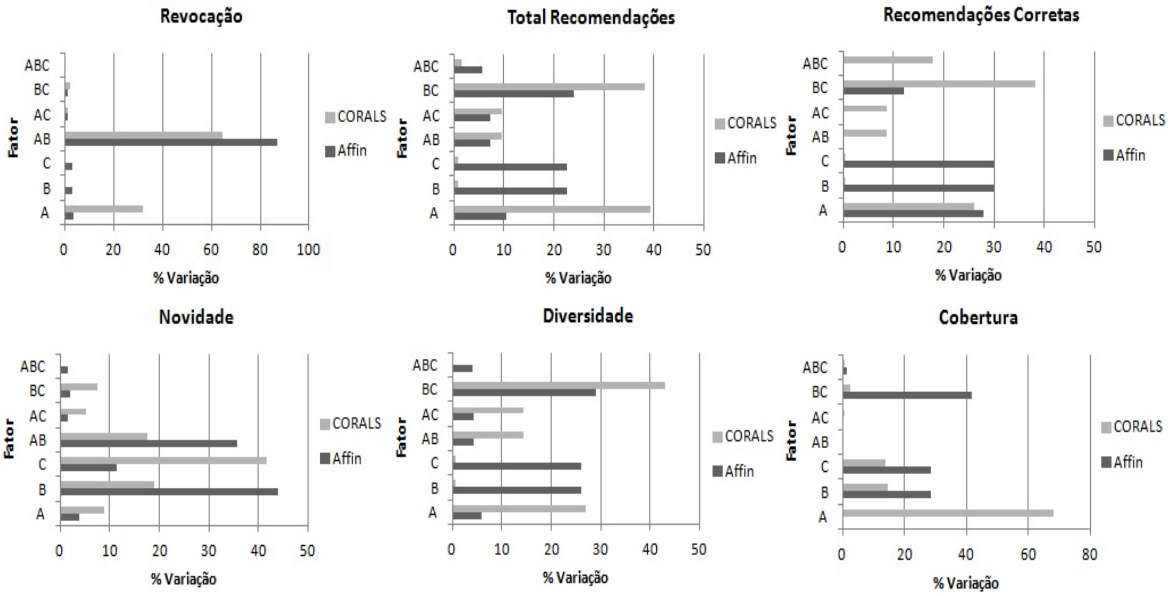


Fig. 1. Porcentagem de variação explicada por cada fator na variável resposta - *Affin* versus *CORALS*

a interação AB explicam praticamente toda a sua variação (91,08%). Já a diversidade e cobertura são afetados principalmente pelos fatores peso proximidade social (B), peso afiliação (C) e sua interação BC.

Em todas as variáveis resposta, a interação entre diferentes fatores (notavelmente entre os dois pesos BC) e a interação entre a estratégia de avaliação e o peso proximidade social (AB), têm impacto significativo. Essa interação existe quando diferenças em um fator depende do nível do outro fator. Por exemplo, na revocação, o fator estratégia de divisão (A) não deve ser considerado em separado do fator peso proximidade social (B), pois o impacto de ambos depende dos seus níveis específicos. Em outras palavras, o impacto do fator estratégia de divisão (A) na revocação depende fortemente do nível do fator peso proximidade social (B) utilizado. Se o nível do peso proximidade social for alterado, o impacto da estratégia de divisão pode mudar.

Função de Recomendação *CORALS*. Existem também efeitos positivos e negativos para as diversas variáveis resposta das recomendações geradas pela *CORALS*. A Figura 1 apresenta as porcentagens de variação dos dados explicadas por cada fator. Os fatores estratégia de divisão (A) e a interação AB são os principais responsáveis pela variação na revocação. Os fatores estratégia de divisão (A) e BC são relevantes no total de recomendações e os fatores estratégia de divisão (A), BC e ABC explicam as maiores variações no número de recomendações corretas. Em relação às variações na novidade, os fatores peso proximidade social (B), peso correlação (C) e AB são os responsáveis. A diversidade é variada pelos fatores estratégia de divisão (A), AB, AC e BC. Por fim, os fatores estratégia de divisão (A), peso proximidade social (B) e peso correlação (C) explicam a variação na cobertura, ou seja, os fatores podem ser considerados independentes na análise dessa variável resposta.

Antes de comparar os resultados obtidos, é importante notar que a realização de um projeto 2^k exige a validação de algumas premissas, em particular, que os efeitos de diferentes fatores sejam *aditivos* [Jain 1991]. Um erro comum é a realização de um projeto 2^k usando fatores cujos efeitos se multiplicam (e não se adicionam). Nesse caso, é necessário fazer uma transformação nos dados, a partir da aplicação da função logaritmo, antes de fazer o projeto 2^k, gerando assim um modelo multiplicativo[4] [Jain 1991]. Enfatizamos que todas as premissas foram validadas nos nossos experimentos. Em particular, para testar se a premissa de fatores aditivos poderia estar influenciando nossos resultados, foram realizados também projetos com modelos multiplicativos, obtendo resultados qualitativamente iguais. Logo, nossas conclusões estão consistentes.

Análise Comparativa das Funções de Recomendação. A Tabela III apresenta os fatores que são relevantes para explicar as variações nas variáveis resposta e a porcentagem total de variação capturada pelos fatores.

[4]Isto porque se $y = a * b$, então, $log(y) = log(a) + log(b)$.

Observa-se que todas porcentagens são superiores a 69%, sendo várias acima de 90%. Então, os fatores apresentados na Tabela III explicam significativamente a maior parte das variações nas variáveis resposta.

Uma análise comparativa entre os fatores relevantes para as variáveis resposta em cada função de recomendação mostra que existem fatores semelhantes em ambas funções. Observa-se que a interação entre a estratégia de divisão e o peso de proximidade social influenciam a revocação de ambas. Além disso, o total de recomendações, a diversidade e a cobertura da *Affin* não são muito impactados pela estratégia de divisão adotada (nem isolado e nem interagindo com outro fator). Já na *CORALS*, todas as variáveis resposta são impactadas por tal fator. Como consequência, observa-se que a forma de avaliação das recomendações de colaboração por meio da divisão da rede social em duas partes influencia nas variáveis resposta, principalmente, na *CORALS*. Outro comparativo mostra que a *CORALS* é mais sensível aos fatores do que a *Affin*, i.e. ao usá-la é preciso dedicar maior atenção na configuração dos fatores. Essa diferença entre as funções é devido aos pesos afiliação e correlação. Esses pesos diferenciam as recomendações de colaboração feitas por cada função.

6. CONCLUSÕES

Foram avaliados os fatores que impactam em funções de recomendação utilizando a técnica de projeto fatorial 2^k. Esses fatores foram representados pela estratégia de divisão e pesos das funções de recomendação. A novidade está em descobrir como esses fatores e suas interações afetam as recomendações de colaborações acadêmica em relação a revocação, novidade, diversidade, cobertura, total de recomendações e recomendações corretas. Os resultados mostram que a *CORALS* é muito mais sensível aos fatores, principalmente à estratégia de avaliação, em relação a *Affin*. A avaliação realizada permite focar nos parâmetros que podem influenciar os resultados das recomendações. No futuro, pretende-se considerar mais níveis no estudo dos fatores/interações para aperfeiçoar as funções de recomendação e configurar adequadamente a estratégia de avaliação.

Agradecimentos. Este trabalho foi financiado por: CAPES, CNPq, Fapemig e InWeb, Brasil.

REFERÊNCIAS

BRANDÃO, M. A. AND MORO, M. M. Recomendação de colaboração em redes sociais acadêmicas baseada na afiliação dos pesquisadores. In *SBBD*. São Paulo, Brasil, 2012.

BRANDÃO, M. A., MORO, M. M., LOPES, G. R., AND DE OLIVEIRA, J. P. M. Using Link Semantics to Recommend Collaborations in Academic Social Networks. In *WWW Workshops*. Rio de Janeiro, Brasil, 2013.

FOUSS, F. AND SAERENS, M. Evaluating Performance of Recommender Systems: An Experimental Comparison. In *WI-IAT*. Sydney, Australia, pp. 735–738, 2008.

FREYNE, J., BERKOVSKY, S., DALY, E. M., AND GEYER, W. Social networking feeds: recommending items of interest. In *RecSys*. New York, USA, pp. 277–280, 2010.

HE, J. AND CHU, W. W. A social network-based recommender system (snrs) data mining for social network data. Annals of Information Systems, vol. 12. Springer US, Boston, MA, 4, pp. 47–74, 2010.

JAIN, R. *The Art of Computer Systems Performance Analysis: techniques for experimental design, measurement, simulation, and modeling*. Wiley, 1991.

LAK, A., LUDWIG, F., SCHOLTYSSEK, J., DIECKHOFF, J., FIEGE, K., AND SCHILLING, M. Size distribution and magnetization optimization of single-core iron oxide nanoparticles by exploiting design of experiment methodology. *IEEE Transactions on Magnetics* 49 (1): 201–207, 2013.

LIMA, E. D., PAPPA, G., ALMEIDA, J. D., GONÇALVES, M., AND MEIRA, W. Tuning genetic programming parameters with factorial designs. In *CEC*. Shanghai, China, pp. 1–8, 2010.

LOPES, G. R., MORO, M. M., DA SILVA, R., BARBOSA, E. M., AND DE OLIVEIRA, J. P. M. Ranking Strategy for Graduate Programs Evaluation. In *ICITA*. Sydney, Australia, 2011.

LOPES, G. R., MORO, M. M., WIVES, L. K., AND DE OLIVEIRA, J. P. M. Collaboration Recommendation on Academic Social Networks. In *ER Workshops*. Vancouver, Canada, pp. 190–199, 2010.

MARZOUGHI, F., FARHANGIAN, M., AHMADIZADEH, E., CHAREJOO, F., AND AGHASIAN, E. Modeling an e-government portal of tourism industry using two level factorial design. In *ICEBE*. Shanghai, China, pp. 421–427, 2010.

SHANI, G. AND GUNAWARDANA, A. Evaluating recommendation systems. In *Recommender Systems Handbook*. pp. 257–297, 2011.

SYMEONIDIS, P., TIAKAS, E., AND MANOLOPOULOS, Y. Transitive node similarity for link prediction in social networks with positive and negative links. In *RecSys*. Barcelona, Spain, pp. 183–190, 2010.

ZIEGLER, C.-N., MCNEE, S. M., KONSTAN, J. A., AND LAUSEN, G. Improving recommendation lists through topic diversification. In *WWW*. Chiba, Japão, pp. 22–32, 2005.

Técnicas de Filtragem para Persistência de Dados de Redes de Sensores Ópticos FBG

A. D. F. Santos, M. J. Sousa, C. S. Sales, M. F. M. Silva, C. S. Fernandes e J. C. W. A. Costa

Universidade Federal do Pará, Laboratório de Eletromagnetismo Aplicado, Brasil
adam.santos@itec.ufpa.br moises.silva@icen.ufpa.br
{marcojsousa, cssj, cindystella, jweyl}@ufpa.br

Resumo. Sensores ópticos FBG ocupam um papel de destaque no monitoramento estrutural devido as suas características únicas. Taxas de aquisição cada vez mais elevadas têm sido possíveis utilizando interrogadores ópticos mais recentes, o que dá origem a um grande volume de dados cuja manipulação e armazenamento tornam-se dispendiosos em termos de processamento e também em termos de espaço de armazenamento em disco. Este artigo propõe duas técnicas de filtragem para reduzir a quantidade de registros no banco de dados ao mesmo tempo em que garante a integridade e a utilidade das informações armazenadas. As técnicas de filtragem propostas foram incorporadas como funcionalidades no sistema de aquisição e persistência e testadas em uma rede de sensores ópticos FBG composta de dois sensores de temperatura e dois sensores de aceleração, implantados para monitorar uma passarela de pedestres de metal com aproximadamente 40 metros de comprimento, situada no campus da UFPA (Belém, Pará). Será mostrado que as técnicas de filtragem propostas oferecem um grande impacto na redução do volume de dados necessários para o monitoramento da passarela.

Categorias e Descritores de Assunto: H.2 [**Database Management**]: Miscellaneous

Palavras-chave: Monitoramento estrutural, Redes de sensores ópticos FBG, Técnicas de filtragem

1. INTRODUÇÃO

A tecnologia para monitoramento estrutural utilizando sensores ópticos FBG (*Fiber Bragg Gratings*) já é considerada madura [Yang 2010]. A vantagem desta tecnologia deriva do fato dos sensores serem criados diretamente na fibra óptica e não precisarem de alimentação de energia. Devido às características favoráveis da fibra óptica como canal de comunicação, os sensores podem ser colocados distantes da infraestrutura de aquisição e processamento de dados. As fibras, assim como os sensores, são bastante duráveis e inertes, podendo ser imersos dentro do concreto da construção ou nos compósitos plásticos que constituem desde asas de avião até pás de aerogeradores eólicos [Tosi et al. 2009].

O monitoramento estrutural é suportado por dados obtidos em tempo real relacionados com a resposta da estrutura, fornecendo indicadores sobre eventuais danos na estrutura que podem condicionar a sua integridade e desempenho. Pode ser registrado um grupo variado de parâmetros, incluindo mudanças de propriedades físicas, químicas ou elétricas, corrosão e fadiga. Estes parâmetros estão relacionados com propriedades físicas da estrutura, tais como deformação, temperatura ou aceleração [Antunes et al. 2010]. Estes dados são obtidos através de interrogadores ópticos com altas taxas de aquisição, que basicamente monitoram os deslocamentos de comprimento de onda dos sensores e disponibilizam essa informação geralmente por intermédio de uma interface de rede Ethernet. Contudo, um grande volume de dados é gerado cuja manipulação e o armazenamento em banco de dados tornam-se tarefas dispendiosas em termos de processamento e espaço de armazenamento em disco.

Com o grande volume de dados gerados do monitoramento estrutural se torna de fundamental importância a filtragem desses dados de maneira a evitar uma sobrecarga de processamento no sistema de aquisição, assim como uma sobrecarga de dados que devem ser persistidos no banco de dados. As tarefas de aquisição, filtragem e persistência podem ser centralizadas em um sistema de *software* que possa gerenciar este processamento por completo e interagir com o banco de dados em tempo real.

Pelo conhecimento dos autores não existem aplicações similares a abordada neste artigo, mas dois artigos estão relacionados por tratarem de modelos de armazenamento de dados. Em [Costa and Cugnasca 2010] é mostrado o uso de *datawarehouse* para gerenciamento de dados obtidos de sensores sem fio no monitoramento do *habitat* de abelhas. O artigo propõe um modelo para extrair, transformar e normalizar dados de redes de sensores e carregá-los em um *datawarehouse*, de forma que possam ser facilmente analisados por especialistas para auxiliar no processo de tomada de decisão. No entanto, esse artigo não mostra nenhuma técnica de filtragem de dados gerados pelos sensores, que podem ser em grande volume e prejudicar o processo de obtenção de informação. Parte-se do pressuposto que os dados das bases a serem integradas já estão consolidados, mas na prática o tratamento de volumes elevados de dados acaba sendo a parte mais onerosa do sistema. Em [Gonçalves et al. 2012] é proposto um modelo dinâmico que considera as vantagens de diferentes modelos de armazenamento, como o modelo local, externo e *data-centric*. A estratégia é baseada em três passos, sendo que o último passo é o condicionamento na frequência de transmissão de mensagens de dados a serem armazenados baseado em limiares. O gerenciamento de transmissão de mensagens de dados é crítico em sensores sem fio, pois tem alto impacto no consumo de energia global. Assim, o artigo avalia diferentes limiares para determinar a frequência na qual um sensor atualiza suas leituras no repositório. Apesar deste último passo ser semelhante à abordagem proposta aqui, o interesse é a economia de energia e não há preocupação com a geração de dados excessivos, que podem sobrecarregar os repositórios de dados. Em uma rede de sensores sem fio com muitos nós sensores medindo diferentes grandezas, assim como em uma rede de sensores ópticos usando multiplexação, o potencial de geração de um grande volume de dados que pode se tornar intratável do ponto de vista computacional é bastante plausível.

Com objetivo de reduzir de forma significativa a quantidade de registros que devem ser armazenados em um banco de dados, todavia garantindo a utilidade das informações armazenadas, este trabalho apresenta duas técnicas de filtragem de dados aplicadas nas amostras geradas pelos sensores FBG que monitoram uma passarela de pedestres. O sistema InterAB (*Interrogator Abstraction*) é o *software* que realiza a aquisição de dados enviados pelo interrogador, aplica as técnicas de filtragem e realiza a persistência em banco de dados das amostras filtradas. A técnica de filtragem por variação no comprimento de onda é proposta para dados de sensores FBG de temperatura e a técnica de filtragem por atividade é sugerida para dados de sensores FBG aceleração.

O artigo está organizado como segue: na Seção 2 é apresentado o funcionamento do sistema InterAB, na Seção 3 são apresentadas e discutidas as técnicas de filtragem de dados propostas, na Seção 4 são mostrados os resultados de cada técnica em um cenário real de aplicação e as considerações finais e os trabalhos futuros são apresentados na Seção 5.

2. AQUISIÇÃO, FILTRAGEM E PERSISTÊNCIA: SISTEMA INTERAB

O sensoriamento óptico está relacionado ao monitoramento de deslocamentos do comprimento de onda de Bragg com as mudanças no mensurando (e.g., temperatura, deformação, aceleração) devido à variação periódica do índice de refração do núcleo da fibra óptica [Yang 2010]. Os interrogadores são as unidades leitoras de medidas que extraem informação medida a partir dos sinais de luz provenientes das cabeças sensoras [Tosi et al. 2009]. Portanto, os interrogadores são destinados a medir os deslocamentos de comprimento de onda de Bragg e converter estes resultados para dados medidos que são disponibilizados para *softwares* clientes que estabeleçam comunicação com interrogador de acordo com seu protocolo. O sistema InterAB é uma aplicação baseada na plataforma Java responsável por esta comunicação, além de possuir outras funcionalidades.

O funcionamento do sistema InterAB é apresentado de forma simplificada na Figura 1. O sistema interage através de *socket* TCP/IP com o interrogador óptico, que está coletando as amostras geradas pela rede de sensores, com o objetivo de receber os dados do monitoramento em tempo real. As amostras são desvios de comprimentos de onda que indicam mudanças na grandeza medida pelo sensor. O InterAB então processa os dados que são transmitidos pelo interrogador em um formato

SCPI (*Standard Commands for Programmable Instruments*) e aplica a técnica de filtragem adequada dependendo do tipo de grandeza medida pelo sensor. Após a etapa de filtragem de dados, o sistema comunica-se com a base de dados por intermédio de conexão JDBC (*Java Database Connectivity*) para persistir as amostras filtradas de cada sensor.

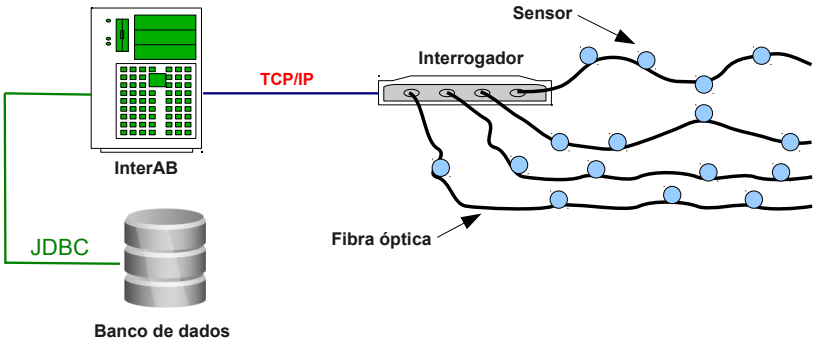


Fig. 1. Funcionamento do sistema InterAB.

3. TÉCNICAS DE FILTRAGEM

As técnicas de filtragem de dados têm por objetivo diminuir a quantidade de dados, provenientes do monitoramento estrutural com redes de sensores FBG, que deve ser armazenada em um banco de dados. Estas técnicas baseiam-se em critérios que proporcionam a redução de registros persistidos sem a perda de qualidade da informação, ou seja, com um número menor de amostras armazenadas ainda é possível identificar o comportamento da grandeza que está sendo medida pelo sensor dentro de uma faixa aceitável de erro.

Considerando um interrogador óptico com taxa de amostragem de 100 S/s [1] com apenas um sensor óptico, a Tabela I apresenta o crescimento da quantidade de registros armazenados no banco de dados caso todas as amostras do sensor tivessem que ser persistidas durante o tempo de monitoramento de uma determinada estrutura. A quantidade de dados pode sobrecarregar o armazenamento na base de dados. Por outro lado, as técnicas de filtragem tem por objetivo reduzir de forma considerável as amostras geradas pelos sensores que serão persistidas.

Tabela I. Crescimento linear da quantidade de registros no banco de dados

Tempo	1 hora	12 horas	1 dia	1 semana	1 mês	1 ano
Quantidade de dados	360.000	4.320.000	8.640.000	60.480.000	264.000.000	3.140.000.000

3.1 Filtragem por variação no comprimento de onda

A filtragem por variação no comprimento de onda é apropriada para grandezas de variação lenta, particularmente, a temperatura. Em se tratando de monitoramento estrutural, a temperatura geralmente não apresenta mudanças significativas em um curto espaço de tempo, salvo em casos de eventos danosos à estrutura. Portanto, na grande maioria das vezes não se faz necessário armazenar um grande volume de dados de medições que não representa variações interessantes de temperatura.

[1] S/s: amostras por segundo

O processo é realizado comparando-se a última amostra s_i coletada com a última amostra que de fato foi inserida no banco de dados s_j. Se $|s_i - s_j| > \Delta T$, então s_i é inserida no banco de dados e substitui o valor de s_j. O parâmetro ΔT define a precisão do sistema e a intensidade da filtragem. Quando maior ΔT menos amostras serão inseridas no banco de dados. Para sensores de temperatura ópticos baseados em FBG, a precisão em graus Célsius é da ordem de $0,1$ graus, portanto este constitui um valor mínimo para ΔT.

3.2 Filtragem por atividade

Os sensores FBG de aceleração apresentam rápidas variações em seu comprimentos de onda de Bragg. Logo, é requerida uma taxa de amostragem adequada por parte do interrogador com o objetivo de possibilitar verificações de variações desta grandeza. Diferentemente da filtragem por variação no comprimento de onda, neste caso não há redundância nas medidas e todas as amostras são importantes.

Considere o caso do monitoramento de uma ponte ferroviária. Normalmente as medidas de aceleração são desprezíveis a menos que um trem venha a atravessar a ponte. Fora esta eventualidade, apenas fenômenos naturais pouco comuns tem a capacidade de despertar algum interesse em termos estruturais. O evento da passagem do trem representaria poucos minutos e não seria eficiente registrar sistematicamente cada segundo ao longo de 24 horas e sim monitorar a ponte apenas durante o evento, quando os dados precisarão ser coletados continuamente usando a taxa de amostragem nominal dos acelerômetros sem perdas de amostras. Portanto, um elemento essencial desta técnica é a identificação de um evento relevante ou de atividade. Este trabalho propõe o uso da diferença absoluta entre a aceleração registrada pela última amostra e a média de n últimas amostras:

$$\Delta s_i = \left| |s_i| - \sum_{k=i-n}^{i-1} |s_k| \right| \tag{1}$$

onde Δs_i representa a diferença absoluta e s_i representa a última amostra coletada pelo interrogador. O subscrito indica a ordem de coleta da amostra, quanto mais recente, maior o subscrito.

Uma vez que pelo menos n_d dentre n_s amostras consecutivas satisfaça $\Delta s_i > s_d$, onde s_d representa um valor de "disparo", então n_0 amostras anteriores e pelo menos n_1 amostras posteriores serão adicionadas no banco de dados. Caso a condição de disparo seja novamente identificada dentre as n_1 amostras posteriores, então mais n_1 amostras a partir do ponto de identificação serão adicionadas. Este processo garante que enquanto houver suficientes ocorrências da condição $\Delta s_i > s_d$, as amostras subsequentes serão apropriadamente adicionadas no banco de dados até que os eventos parem de ocorrer e o sistema monitorado volte para um estado de repouso.

4. RESULTADOS

4.1 Cenário de testes

As técnicas de filtragem propostas foram testadas em um cenário caracterizado por uma passarela de pedestres de metal com aproximadamente 40 metros de comprimento, situada no campus da UFPA (Belém, Pará). O monitoramento desta estrutura é realizado através de uma rede de sensores ópticos FBG composta de dois sensores de temperatura e dois sensores de aceleração, e de um interrogador óptico industrial *BraggMeter* com taxa de amostragem de 100 S/s. A passarela, exemplos de sensores instalados e o esquema de monitoramento da estrutura são mostrados na Figura 2. Um cabo óptico de aproximadamente 500 metros conecta a rede de sensores instalada na passarela com o interrogador localizado em um laboratório para medições.

Fig. 2. Cenário de testes: passarela de pedestre da UFPA.

Como forma de testar as técnicas propostas foram realizadas coletas durante 27 horas, buscando filtrar as amostras que representassem variações de aproximadamente 1°C e 2°C. Um comparativo entre o sinal original e os sinais filtrados com variação de 1°C e 2°C é mostrado na Figura 3. O sinal original possui 10148765 amostras, o sinal filtrado para variações de 1°C possui 47 amostras e o sinal filtrado para variações de 2°C possui 11 amostras, o que implica em uma redução de registros na base de dados de aproximadamente $99,9995\%$ e $99,9998\%$, respectivamente. Contudo, é possível verificar uma grande redução no número de registros que foram persistidos no banco de dados após a filtragem sem ao mesmo tempo perder a qualidade da informação, pois através das amostras armazenadas é possível observar o comportamento da temperatura na passarela e estimar de forma aproximada o sinal original. Entretanto, é importante destacar que a redução na quantidade de registros é aceitável até certo ponto, em outras palavras, quanto menor o número de registro no banco de dados menor é a possibilidade de precisão na identificação do comportamento da grandeza medida pelo sensor.

Um comparativo entre o sinal filtrado com o sinal completo para um acelerômetro uniaxial no eixo z (normal ao piso da passarela) é apresentado na Figura 4. Foram utilizados os seguintes parâmetros: $n = 100$, $s_d = 0,05\ g$, $n_s = 100$, $n_d = 5$, $n_0 = 1000$ e $n_1 = 2000$. As amostras registradas no banco de dados são aquelas que formam a curva preta e as amostras em cinza não foram registradas. Observa-se que as amostras registradas estão agrupadas em torno de pontos de atividade com picos que superam $0,1\ g$. O mérito desta técnica de filtragem aplicada a acelerômetros é que ela permite ignorar eventuais pontos fora da curva originados por erros de sincronismo da unidade de interrogação devido ao ruído. O valor n_d exige que pelo menos 5 eventos ocorram dentro de um certo intervalo de tempo e dessa forma um evento autêntico pode ser distinguido de um breve ruído.

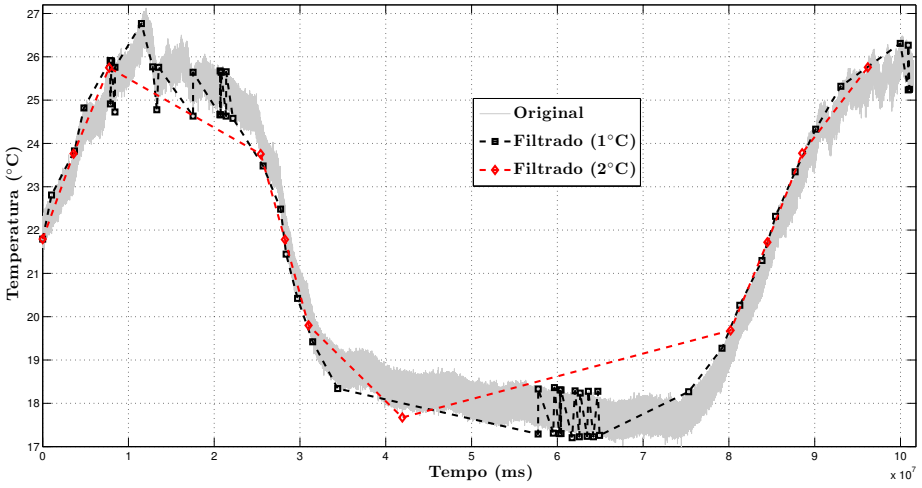


Fig. 3. Filtragem de variações de 1°C e de 2°C.

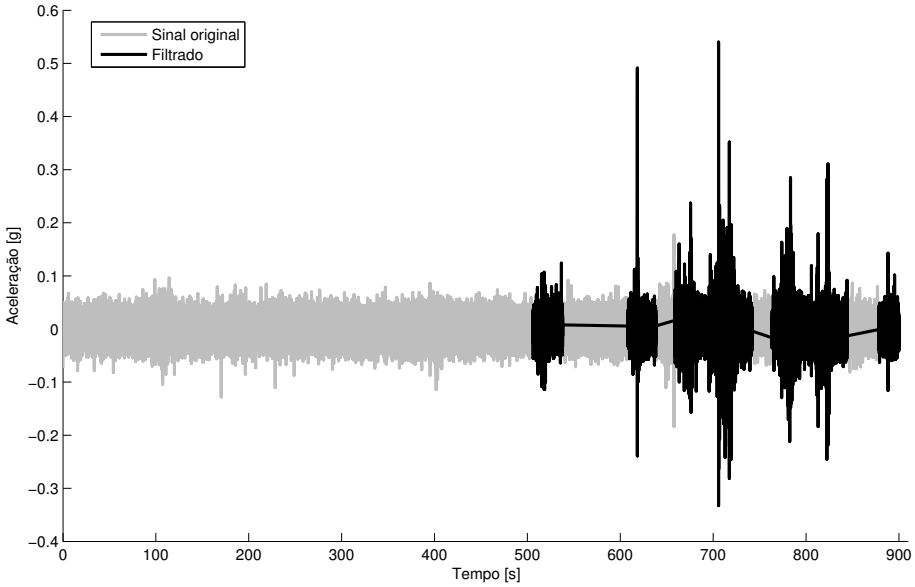


Fig. 4. Filtragem de dados de sensores FBG de aceleração.

5. CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS

Neste artigo foram apresentadas duas técnicas de filtragem de dados propostas para redes de sensores ópticos FBG, particularmente sensores de temperatura e aceleração. As técnicas, que foram incorporadas no sistema InterAB, tiveram como objetivo a redução de forma considerável de registros persistidos no banco de dados, mas sem alterar de forma significativa a qualidade da informação processada. Os resultados de aplicações das técnicas de filtragem por comprimento de onda e por atividade no cenário real de uma passarela de pedestre mostraram as vantagens da utilização de uma etapa de filtragem de dados antes do armazenamento, reduzindo a quantidade de registros em mais de 90% do volume original. Para os próximos passos pretende-se desenvolver novas técnicas baseadas em outros critérios de filtragem, estender a técnica de filtragem por atividade para sensores ópticos FBG de deformação mecânica e ampliar a abordagem proposta neste trabalho para outros tipos de redes de sensores.

REFERÊNCIAS

Antunes, P., Lima, H., Varum, H., and Andre, P. Static and dynamic structural monitoring based on optical fiber sensors. In *12th International Conference on Transparent Optical Networks (ICTON)*. pp. 1–4, 2010.

Costa, R. A. and Cugnasca, C. E. Use of Data Warehouse to Manage Data from Wireless Sensors Networks That Monitor Pollinators. In *Eleventh International Conference on Mobile Data Management (MDM)*. pp. 402–406, 2010.

Gonçalves, N. M. F., dos Santos, A. L., and Hara, C. S. DYSTO - A Dynamic Storage Model for Wireless Sensor Networks. *Journal of Information and Data Management* 3 (3): 147–162, 2012.

Tosi, D., Olivero, M., Perrone, G., Vallan, A., and Arcudi, L. Simple fiber Bragg grating sensing systems for structural health monitoring. In *IEEE Workshop on Environmental, Energy, and Structural Monitoring Systems (EESMS)*. pp. 80–86, 2009.

Yang, N. Technologies for structural test and monitoring: The modern approach. In *IEEE AUTOTESTCON*. pp. 1–5, 2010.

A Redundancy-Free Approach to Schema Evolution

Claudiomar Desanti[1], Marcos Bernardelli[1], Márcio Fuckner[1], Raquel Kolitski Stasiu[1,2]

[1] Pontifícia Universidade Católica do Paraná, Brazil
[2] Universidade Tecnológica Federal do Paraná, Brazil
{c.desanti, m.bernardelli, marcio.fuckner, raquel.stasiu}@pucpr.br, raquel@utfpr.edu.br

Abstract. Changes in relational database schemas are part of the continuous improvement process of information systems. In general, iterative and incremental approaches for software development produce frequent releases in order to improve the operational environment and also attend to business or legal requirements. As an effect, such dynamic produces significant amounts of work to database administrators when applying those changes in production. The risk also increases as cumulative changes need to be manually executed in the environment. Having those issues in mind we propose an approach to automate the schema evolution, using a redundancy-free algorithm to merge cumulative changes, reducing downtimes and improving the software availability. When executed in a production environment, the proof-of-concept demonstrated a significant reduction of the amount of time to process the update, resulting in the same schema as applied by known tools.

Categories and Subject Descriptors: H.2.1 [**Database Management**]: Logical Design—*Schema and subschema*

Keywords: Diff Algorithms, Evolving Database Systems, Schema Evolution

1. INTRODUCTION

Scope changes are part of the continuous improvement process of information systems. Several software development teams use iterative and incremental development approaches in order to accelerate the requirements gathering process, in contrast to traditional waterfall approaches. The agile approach leads to frequent changes of software artifacts and requires an effective change management process. As a consequence, those changes affect database structures and their current data [Batini et al. 2009] [Roddick 2009]. Also, several companies acquire off-the-shelf solutions, which aim at attending different configurations and customer needs. It leads companies to update their environment with small, but frequent releases generated from their suppliers. Risks and downtime rates increase when cumulative and interdependent scripts are applied. As a result, customers decrease the frequency of updates, adopting cumulative "big bang" update strategies, which demands complex, interdependent and error-prone tasks.

According to [Fowler 2002], refactoring is a quality improvement process to rewrite source-code without affecting the resulting product. Changes could improve readability, which has impact on future maintenance. Moreover, architectural changes could allow software modularization and improve testability. [Ambler and Sadalage 2006] have adapted Fowler's concept to fit within the database reality. Such changes aim at attending an indirect business requirement or improve the integration between the software and the database. These improvements could generate new database objects such as indexes, columns, relationships and constraints.

Schema evolution is the ability for a database schema to evolve without the loss of existing information. Due to its great significance and practical importance, many research efforts have been made to propose approaches for schema evolution. The work of [Papastefanatos et al. 2008] proposes a

graph-based framework for capturing changes in database schemas and generating annotation based on a proposed extension to the SQL language, specifically tailored for the management of evolution. Using a similar approach, the work of [Curino et al. 2009] presents a web-based framework, which uses several semantic techniques for query rewriting. Techniques for mapping the differences between models can also be used as a key element for the schema evolution problem, even if they address different contexts such as ontologies and object-oriented models. For example, [Fagin et al. 2011] presents two fundamental operators on schema mappings, namely composition and inversion to address the mapping adaptation problem in the context of schema evolution. [Carvalho et al. 2013] proposes an evolutionary approach to find complex matches between schemas of semantically related data repositories using only the data instances. The work of [dos Santos Mello et al. 2002] describes a unification method for heterogeneous XML schemata.

Differently from other works, this approach proposes a schema evolution mechanism that minimizes possible redundancies that exists in a set of schema updates. A schema update in this particular scenario could be interpreted as a group of SQL sentences, arranged in a labeled version. The proposed mechanism allows the user to apply the set of updates in only one step. For the sake of clarity, the solution could be decomposed in two main processes: (i) Generation of a delta version with the difference between two schemas and (ii) merging a set of delta versions. In (i) the inputs come from the development team, whereas in (ii), the tasks are conducted by database administrators who want to refresh their environments with the last stable version of the software. The results show that our approach reduced the amount of time to process the update, resulting in the same target schema as applied by known tools.

2. A REDUNDANCY-FREE APPROACH TO SCHEMA EVOLUTION

A method was developed to address the problem of finding differences between two schemas using a diff algorithm. This type of algorithm aims at finding the shortest distance between two scripts, which can be represented by t_i and t_{i-1}, where t represents the schema and i is its version number. The result is a delta version, represented by d, which has directives to allow the construction of one of the t versions using the previous or a subsequent version [Cobena et al. 2001]. In this particular case, both scripts and deltas are sets of SQL commands. The generated prototype was developed to recognize PostgreSQL objects [PostgreSQL Global Development Group 2013]. However, a clear separation between core and interface layers demands few changes to adapt the prototype to work with other relational databases. The next section describes the process to obtain the delta version.

2.1 FINDING THE DELTA VERSION BETWEEN TWO SCHEMAS

A database schema is a hierarchical structure of objects. Hence, a tree could be used as its representation. Figure 1 presents an illustrative and non-exhaustive tree that fits to the natural taxonomy of a schema. To generate the delta d, we transform schemas t_i and t_{i-1} into trees and use them as an input to the diff algorithm. To discover the differences, the algorithm needs to identify which node was changed (e.g., attribute, table or index). To provide such functionalities the algorithm traverses both trees from the upper nodes to the leaf, matching each one with their corresponding node in the opposite tree.

In order to avoid visiting all nodes during the comparison, the MD5 (*Message-Digest Algorithm 5*) [1] hash function is executed for each node, using the associated SQL sentence as an input parameter. Figure 2 shows two trees with their calculated hash values. When traversing the tree (a), the algorithm ignores the node 1 because both hashes from trees (a) and (b) are the same. By the other hand, the same example shows *node*2 with different hash values, which justifies further verification.

[1] A complete specification of the MD5 algorithm can be found at *http://tools.ietf.org/html/rfc1321*

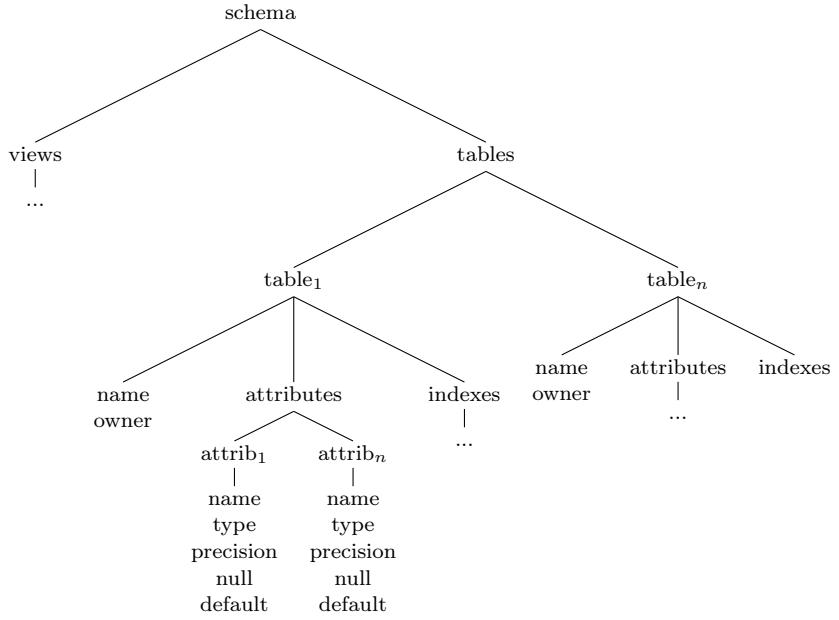


Fig. 1. A tree representing the schema

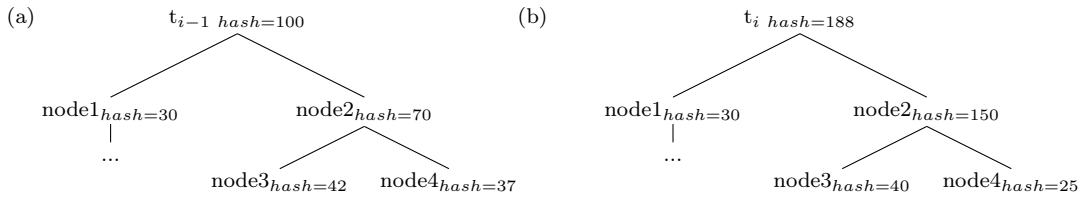


Fig. 2. A tree representing the schema

The algorithm evaluates each node from the versions t_i and t_{i-1} using three possible patterns: *drop* operations indicating that one or more objects in t_i were removed from t_{i-1}, *create* operations indicating that one or more new objects in t_i were identified, and *modify* operations selecting objects belonging to both trees, with different values on at least one common attribute in the node.

At the end of the comparison process, a delta version is generated in XML format (see Figure 3). A root node called *version* contains the attribute that indicates the version number. The subsequent nodes represent structural changes by object type. The implemented prototype recognizes structural changes in tables, views, sequences and indexes. Each one has specific metadata, providing sufficient information to change the schema. There is a common attribute called *op*, which identifies the type of difference (*drop*, *create* or *modify*). One special node called *scripts* can be used to allow administrators to insert user-defined scripts, commonly used for conversion process purposes.

2.2 MERGING A SET OF DELTA VERSIONS

This section presents an approach to evaluate a set of delta versions, identify and remove their redundancies, and finally apply those changes in the target schema. The approach is organized as follows: (i) recovery of delta versions from the repository, (ii) merging the delta versions, removing redundancies, and applying the changes in the target schema. In (i) the current schema and the set of deltas are selected from one repository. Schemas should be located in a configuration control infrastructure with support to artifact versioning and tracking, which is out of the scope of this work.

```
<version id="3">
   <tables>
      <table id="invoice" operation="modify">
         <attributes>
            <attr id="number" op="create"><name>number</name><type>integer</type> ...
            <attr id="comments" op="modify"><name>remarks</name> <type>varchar</type> ...
         </atributes>
      </table>
   <tables>
   <scripts>...</scripts>
</version>
```

Fig. 3. Generated delta example

In (ii), the algorithm receives a list of delta scripts, identify and merge those changes in one redundancy-free script. Three types of changes could occur in one database object: *drop* operation, *create* operation, and *modify* operation. For illustration purposes, the following scenario with a delta script in the version v_1 can be considered. This delta script from version 1 creates a new field in the table *invoice* called *trackingNumber*. The field *trackingNumber* is an integer field that accepts null values and has a precision of 8 digits. In version v_2 the field is changed in order to forbid null values. Finally, in the version v_3, the precision is changed to support 10 digits instead of 8.

The Algorithm 1 was created in order to mimic the decisions typically made by one database administrator when merging two scripts. The function called $schemaMerge$ requires an input parameter called D, which has the set of deltas to merge. Lines from 12 to 20 are responsible for evaluate each sentence from the delta, using a rule-based algorithm detailed in Algorithm 2 called *mergeSentences*. The rules guide the merging process, deciding what is the most economic operation. Redundancies are removed as a consequence of a series of override operations in the underlying object. For example, taking two SQL sentences s_1 = *"create table foo (field1 integer)"* and s_2 = *"alter table foo add field2 char(1)"*. The first sentence ($s1$) is a *create operation*, while the second ($s2$) is a *modify operation*. According to the rule 3 described in the algorithm 2, the merging process will generate only one merged sentence S_r = *"create table foo (field1 integer, field2 char(1))"*.

Algorithm 1 Schema merging algorithm

```
Require: D = d_1, d_2, ..., d_n : delta bundle
 1: function schemaMerge
 2: Map ← empty map {Map of schema objects with their changes}
 3: M ← empty set {Set of merged changes}
 4: for each delta d_i ∈ D do
 5:    for each change c ∈ d_i do
 6:       add c in Map[c.object name].values
 7:    end for
 8: end for
 9: for each object name n_i ∈ Map.keys do
10:    current change ← null
11:    for each change c ∈ Map[n_i].values do
12:       current change ← mergeSentences(current change, c)
13:    end for
14:    if current change ≠ null then
15:       M.add(current change)
16:    end if
17: end for
18: return M
```

3. EXPERIMENTAL RESULTS

This section presents the results of o the first experiment, which evaluates the performance and reliability of this approach. A set of releases of a real ERP called *Auditor*, implemented by the *Method Informatics* company were used as a input for the experiment, resulting in 20 schema-changing scripts

Algorithm 2 Sentence merging functions

```
1: function mergeSentences(current,new)
2: if current = null then
3:     return new
4: end if
5: if current.operation ≡ create and new.operation ≡ drop then
6:     return null
7: end if
8: if current.operation ≡ create and new.operation ≡ modify then
9:     return mergeScript(create, current.sentence, new.sentence)
10: end if
11: if current.operation ≡ drop and new.operation ≡ create then
12:     return new
13: end if
14: if current.operation ≡ modify and new.operation ≡ drop then
15:     return new
16: end if
17: if current.operation ≡ modify and new.operation ≡ modify then
18:     return mergeScript(modify, current.sentence, new.sentence)
19: end if
20:
21: function mergeScript(type,current,new)
22: if type ≡ create then
23:     return current + new
24: else
25:     return current ∪ new
26: end if
```

delivered from 2011 to 2012. The samples allowed the detection of different scenarios, from simple inclusion of objects to more complex tasks such as column merging and splitting. The scripts were applied in two databases namely A and B, with the same schema but varying in size (170mb and 410mb). Splitting and merging operations must be added in the delta script, specifically inside the tag called *scripts* as shown in Figure 3. Table I also shows relevant numbers about the experiment.

Table I. Experiment numbers

Information	#
# of tables at the baseline version	317
# of schema-changing scripts	20
# of new tables	38
# of modified tables	21
# of modifications (attribute insertions, deletes and updates)	239
# of tables at the 20th version	355

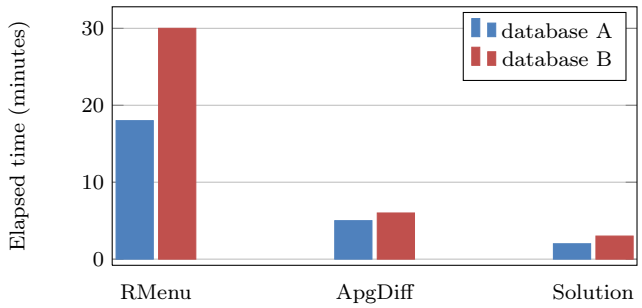


Fig. 4. Comparison of approaches (execution time).

Rmenu [Genexus 2013] and ApgDiff [StartNet 2013] were selected to make a preliminary comparison, with focus on response time and reliability. After the evolution, all schemas must be exactly the same. Both Rmenu and ApgDiff were selected because they are mature and multi-function tools, used in several environments. As it can be seen in Figure 4, the prototype was able to reduce the time spent

during the execution, since there is no extra work in order to process modifications that may be superposed. Compiled versions optimizes the DDL script generated to modify the database schema. For this optimization, all the detected redundancies were removed, reducing the number of DDL instructions and, consequently, decreasing the execution time.

4. CONCLUSIONS AND FUTURE WORK

This paper deals with the problem of database evolution maintenance using an approach to integrate the schema versioning during the database refactoring process, avoiding redundancies during the execution of several schema evolutions. The first proof-of-concept allows the automation and optimization of potential set of changes applied in one database schema. The applicability and efficiency of this approach has been tested using real world examples. A significant time reduction was observed in order to process the update, resulting on the same target schema as applied by known tools.

This work presents the initial results of experiments, where several improvements were detected. One of those improvements includes the evaluation of the approach on different databases as well as adding other perspectives to the benchmark, such as the readability of the generated scripts and identification of schema-changing patterns. There are three others directions of investigation: to identify the dependence objects in time of modification, to eliminate the redundancy during data update by changing the order of DDL script execution and to study techniques to identify the actual version of database schema.

REFERENCES

Ambler, S. W. and Sadalage, P. J. *Refactoring Databases: Evolutionary Database Design*. Addison-Wesley, 2006.

Batini, C., Cappiello, C., Francalanci, C., and Maurino, A. Methodologies for data quality assessment and improvement. *ACM Computing Surveys (CSUR)* 41 (3): 16:1–16:52, July, 2009.

Carvalho, M. G., Laender, A. H. F., Gonçalves, M. A., and Da Silva, A. S. An evolutionary approach to complex schema matching. *Information Systems* 38 (3): 302–316, May, 2013.

Cobena, G., Abiteboul, S., and Marian, A. Detecting changes in XML documents. In *Proceedings of the 2001 IEEE International Conference on Data Engineering*. ICDE '01. pp. 41–52, 2001.

Curino, C. A., Moon, H. J., Ham, M., and Zaniolo, C. The PRISM workbench: Database schema evolution without tears. In *Proceedings of the 2009 IEEE International Conference on Data Engineering*. ICDE '09. IEEE Computer Society, Washington, DC, USA, pp. 1523–1526, 2009.

dos Santos Mello, R., Castano, S., and Heuser, C. A. A method for the unification of XML schemata. *Information & Software Technology* 44 (4): 241–249, 2002.

Fagin, R., Kolaitis, P. G., Popa, L., and Tan, W. C. Schema mapping evolution through composition and inversion. In *Schema Matching and Mapping*. pp. 191–222, 2011.

Fowler, M. *Refactoring: Improving the Design of Existing Code*. Addison-Wesley Professional, 2002.

Genexus. *RMenu*. Genexus, 2013. Available at http://www.genexus.com.

Papastefanatos, G., Vassiliadis, P., Simitsis, A., Aggistalis, K., Pechlivani, F., and Vassiliou, Y. Language extensions for the automation of database schema evolution. In *ICEIS 2008 - Proceedings of the Tenth International Conference on Enterprise Information Systems*. pp. 74–81, 2008.

PostgreSQL Global Development Group. PostgreSQL, 2013. Available at http://www.postgresql.org.

Roddick, J. F. Schema evolution. In *Encyclopedia of Database Systems*. pp. 2479–2481, 2009.

StartNet. *Another PostgreSQL Diff Tool (apgdiff) free database schema diff tool*. StartNet, 2013. Available at http://www.apgdiff.com.

Um sistema de recomendação para informações tecnológicas agrícolas

Flavio M. M. de Barros[1], Stanley R. M. Oliveira[1,2], Leandro H. M. Oliveira[2]

[1] Universidade Estadual de Campinas, Brasil
flavio.barros@feagri.unicamp.br
[2] Embrapa Informática Agropecuária
{stanley.oliveira, leandro.oliveira}@embrapa.br

Abstract. The aim of this work was to design, develop and deploy a web recommender system based on association rules to automatically make recommendations of agricultural content, according to the profile of a user community. The data used in this study were extracted from a database of accesses to the site of Embrapa Information Agency. The user visits were stored in a structure of access lists, and from such lists, association rules between pages were generated. The set of rules led to a knowledge base that was used to make content recommendations to users. The system was evaluated using a metric called bounce rate, so that by means of statistical tests it was possible to evaluate the impact of these recommendations. The results revealed that through recommendations, users find relevant information associated with their visits and increase their time spent on the site. The system was validated for the sugarcane crop, but it can be easily extended to provide recommendations regarding other crops.

Categories and Subject Descriptors: H.3.5 [**Online Information Services**]: Web-based services; H.4.0 [**Information Systems Applications**]: General; H.2.8 [**Database Applications**]: Data mining

Keywords: Agricultural Information Systems, Association Rule Mining, Recommender Systems

1. INTRODUÇÃO

Aplicativos modernos, desenvolvidos para o ambiente Web, possuem mecanismos para aprender as preferências de seus usuários. Esses aplicativos são conhecidos como sistemas de recomendação e seus usuários estão cada vez mais acostumados com suas recomendações. Esses sistemas são bem diversos, com diferentes objetivos, desde indicações de produtos até sugestões de leitura, filmes e lugares turísticos. No entanto, no domínio agrícola, nota-se uma carência de tais sistemas já que não se conhece registros na literatura.

No Brasil, devido a uma demanda por informações técnicas agrícolas de qualidade, a Empresa Brasileira de Pesquisa Agropecuária (Embrapa) investiu em um projeto denominado Agência de Informação Embrapa, com o objetivo de organizar, tratar, armazenar e divulgar informações técnicas e conhecimentos gerados ao longo de 40 anos de pesquisa. Por meio do endereço eletrônico `http://www.agencia.cnptia.embrapa.br`, o usuário tem acesso a todo o conteúdo do site na forma de textos, artigos, livros, arquivos de imagem, arquivos de som e planilhas eletrônicas. Em particular, a Agência apresenta as principais informações de cadeias produtivas, como aspectos socioeconômicos e ambientais, planejamento, manejo, colheita, processamento e gestão industrial. O conteúdo foi organizado para atender pesquisadores, produtores rurais, extensionistas e, diariamente, o site recebe milhares de acessos que são registrados em uma base de dados.

O portal da Agência Embrapa disponibiliza milhares de páginas de conteúdo individual, mesmo

para uma única cultura como a cana-de-açúcar. Essa oferta de informações em grande quantidade pode confundir e dificultar o acesso pelos usuários [Yang and Tang 2003]. Assim, a recomendação de conteúdo é uma alternativa viável para auxiliar usuários devido ao volume elevado de informações [Kumar, A; Thambidurai 2010]. A recomendação personalizada de conteúdo aumenta a usabilidade dos sistemas, agrega valor aos serviços, poupa tempo e fideliza usuários.

Uma forma de produzir recomendações é inferir o comportamento dos usuários baseado nos padrões de uso das informações. Em particular, no domínio agrícola, onde há informações em quantidade elevada, armazenadas digitalmente em bancos de dados, as ferramentas de mineração de dados apresentam recursos para análise que podem fornecer padrões de uso do site para fazer recomendações. Esta é uma das melhores alternativas dentre as técnicas utilizadas para identificar o comportamento de uso de sites e oferecer recomendações aos seus usuários [Han et al. 2011].

Assim, considerando a importância da agricultura para o Brasil e a existência de grandes repositórios de informações agrícolas disponíveis na Web, o objetivo deste trabalho foi projetar e desenvolver um sistema de recomendação web, baseado em regras de associação, que ofereça recomendações sobre a cultura da cana-de-açúcar de acordo com os padrões de acessos de uma comunidade de usuários. No Brasil, essa é uma das primeiras iniciativas de aplicações de sistemas de recomendação no domínio agrícola. Esse sistema pode ser facilmente estendido para outros portais, com outras culturas agrícolas.

2. TRABALHOS RELACIONADOS

Existem experiências bem-sucedidas com sistemas de recomendação de livros, CDs, e outros produtos na Amazon.com [Linden et al. 2003], filmes pela MovieLens [Miller et al. 2003] e notícias pela VERSIFI Technologies [Billsus et al. 2002]. Na implementação desses sistemas, as técnicas mais utilizadas são a filtragem colaborativa e a filtragem baseada em conteúdo [Ricci et al. 2011]. Existem também técnicas híbridas que combinam as duas abordagens [Burke 2000].

Existem também casos de aplicação de regras de associação em sistemas de recomendação para descobrir relações entre páginas visitadas. Por exemplo, em [Kazienko 2009] foram determinadas regras de associação entre páginas para criar listas de recomendações. Em [Yang and Parthasarathy 2003] e [Jorge et al. 2002] foram implantados sistemas de recomendação baseados exclusivamente em listas de recomendação, obtidas por meio de mineração de regras. Mais recentemente regras de associação foram utilizadas no contexto de classificação em um sistema de recomendação aplicado ao turismo [Lucas et al. 2013] e em [Paranjape-Voditel and Deshpande 2013] foram utilizadas regras de associação para a criação de um sistema de recomendação de portfólios no mercado de ações.

Esse trabalho se diferencia dos demais em três aspectos fundamentais: a) escopo da aplicação: no domínio agrícola, aplicações de sistemas de recomendação são raramente encontradas; b) extensibilidade: o sistema proposto pode ser facilmente estendido para diversas culturas agrícolas; c) método de avaliação das recomendações: a avaliação do sistema proposto foi realizada do ponto de vista da usabilidade do site, por meio da taxa de rejeição. Conforme [Sculley et al. 2009], a taxa de rejeição pode ser utilizada para avaliar se os usuários encontram a informação desejada em um site.

3. ARQUITETURA DO SISTEMA DE RECOMENDAÇÃO

A arquitetura do sistema é basicamente dividida em duas partes: servidor e navegador, conforme Figura 1. O servidor compreende o servidor apache, o módulo PHP, o banco de dados e o módulo Recommender, implementado em R. Quando um usuário acessa uma das páginas da Agência Embrapa, o servidor apache responde enviando dois arquivos: uma página em HTML e o script responsável pelas consultas às recomendações e, depois, atualiza a página visitada. Assim que o navegador carrega a página, o script é inicializado e envia uma requisição de consulta ao servidor com o endereço da

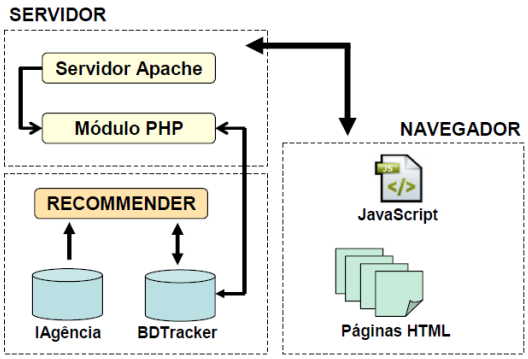


Fig. 1. Arquitetura do sistema de recomendação para informações tecnológicos agrícolas.

página visitada. Em seguida, é acionado o módulo PHP que realiza a consulta ao banco de dados "BDTtacker". Caso haja recomendações, o servidor envia as informações ao script que atualiza a página com os links (sugestões de páginas a ser visitadas). Caso contrário, a página fica inalterada.

A inferência das regras envolve a interação entre o Recommender e os bancos "BDTracker" e "IAgência". No primeiro, o Recommender consulta as informações sobre os históricos de navegação dos usuários e, no segundo, os títulos das páginas que serão apresentados nas recomendações. As regras são recalculadas uma vez por semana, assim os novos acessos dos usuários, incluindo os cliques em recomendações, atualizam o banco de dados contribuindo para o aparecimento de novas regras. Um dos resultados mais importantes desse trabalho é a base de conhecimento gerada. Essa base, representada pelas regras de associação entre páginas, além de ser utilizada para recomendações também pode ser utilizada para adaptar o site de acordo com o perfil de uso dos usuários [Perkowitz and Etzioni 2000].

4. RESULTADOS

O conteúdo do portal da Agência Embrapa está estruturado em árvores de conhecimento. Como os acessos à árvore do conhecimento da cana-de-açúcar representam quase metade do total de acessos à Agência Embrapa, o sistema proposto foi validado para essa cultura agrícola. As visitas dos usuários à árvore da cana-de-açúcar foram estruturadas em duas tabelas: clientes e tracker, sendo que na tabela clientes são armazenadas informações de cada usuário que entrou na Agência e iniciou uma sessão e na tabela tracker são armazenados dados relativos a cada visualização de página associada a um usuário.

Sempre que uma requisição é feita, são registrados os seguintes atributos do usuário na tabela clientes: idsessao (identificador único com 32 caracteres), ip, tempo de permanência, latitude, longitude, cidade, pais e estado. Cada linha na tabela clientes representa um usuário. Na tabela tracker são registrados os seguintes atributos: idtracker (identificador único de cada sessão), idsessão (o mesmo da tabela clientes), página visitada, árvore (neste caso a árvore do conhecimento é a da cana-de-açúcar), data do servidor, hora do servidor e tempo da sessão.

A tabela clientes, no banco de dados, possui 2.574.763 linhas, que representam o número de usuários distintos que acessaram conteúdos sobre a cana-de-açúcar no período compreendido entre outubro de 2010 a janeiro de 2013. A tabela tracker possui 5.223.003 linhas, onde cada linha contém a informação de cada requisição individual de uma página do sistema, também relativo ao mesmo período. Cada linha da tabela tracker representa uma requisição de página. Logo, um mesmo usuário pode aparecer em várias linhas, pois este pode ter requisitado mais de uma página em sua sessão.

A base de conhecimento gerada para a árvore da cana-de-açúcar é composta por vinte e oito regras de associação entre as páginas (Tabela I), relacionando páginas de conteúdo textual e páginas com recursos eletrônicos, como arquivos em pdf e vídeos. Mesmo com o suporte e a confiança baixos

($sup = 0.0005$ e $conf = 0,51$) e com o volume elevado de sessões de usuário (mais de 2 milhões), não emergiram padrões de acesso que relacionassem páginas que já não estavam ligadas entre si. Conforme Tabela II, somente seis páginas possuem mais de uma recomendação. Além das páginas apresentarem muitos links (não são recomendações, mas hiperlinks para outras páginas), estas apresentam de 1 a 4 recomendações. Esse resultado é particularmente importante, pois mostra que a base de conhecimento tem um potencial de resumo para direcionar o usuário aos links mais importantes.

Table I. Base de conhecimento com 28 regras de associação entre páginas da cultura da cana-de-açúcar.

Regra	Antecedente	Consequente	Sup	Conf
1	Fabricação do açúcar	Produção dos açúcares líquido	0,007%	83%
2	Extração	Moendas	0,015%	83%
3	Processamento da cana	Açúcar e álcool: o combustível do Brasil [vídeo]	0,010%	80%
4	Variedades	3ª geração de variedades CTC	0,007%	79%
5	Custos e rentabilidade	Planilha geral de custos e rentabilidade	0,013%	77%
6	Cachaça	Fábrica de aguardente de cana-de-açúcar	0,007%	75%
7	Cachaça	O perfil da cachaça	0,005%	72%
8	Processamento da cana	Açúcar e álcool: a tecnologia sucroalcooleira [vídeo]	0,007%	72%
9	Queima	Exigências	0,012%	70%
10	Variedades	Variedades RB de cana-de-açúcar	0,007%	68%
11	Processamento da cana	Modelo de otimização para o planejamento em usinas	0,010%	64%
12	Processamento da cana	Açúcar e álcool: a produção do álcool [vídeo]	0,012%	63%
13	Plantio	Mudas	0,119%	63%
14	Açúcar	Mercado	0,018%	62%
15	Correção e adubação	Adubação e calagem em cana-de-açúcar	0,006%	62%
16	Doenças	Outras doenças	0,042%	60%
17	Qualidade de matéria-prima	Produção de etanol de cana-de-açúcar	0,007%	59%
18	Abertura	Cana-de-açúcar	0,006%	59%
19	Cachaça	A arte de produzir cachaça [vídeo]	0,005%	57%
20	Diagnose nutricional	Expectativa da produtividade	0,006%	56%
21	Plantio	Recomendações técnicas em Rondônia	0,008%	55%
22	Análise de solo	Interpretação da análise	0,032%	54%
23	Preparo do solo	Plantio direto	0,035%	53%
24	Implicações	Exigências	0,009%	53%
25	Abertura	Pré-produção	0,147%	52%
26	Doenças fúngicas	Outras doenças	0,036%	51%
27	Meio ambiente	Diagnóstico agroambiental	0,010%	51%
28	Meio ambiente	Impactos	0,017%	51%

De acordo com [Sculley et al. 2009], quando a taxa de rejeição de um site (conjunto de páginas) ou de uma única página é alta, isto pode ter dois significados: os usuários encontram exatamente o que estavam procurando ou eles não encontraram a informação desejada e não acham o site atrativo para explorá-lo. Assim, para verificar o efeito do sistema de recomendação foram calculadas as taxas de rejeição para as páginas da cana-de-açúcar que receberam recomendações. Todas as sessões foram consideradas no período de 25 de novembro de 2012 até 16 de janeiro de 2013.

Para a verificação da significância estatística das variações das taxas de rejeição, antes e depois da implantação do sistema, foram utilizados dois testes estatísticos: teste Z e o teste qui-quadrado para diferenças entre proporções. Os dois testes são paramétricos, sendo que o segundo é o teste estatístico mais comumente utilizado para testes de homogeneidade [Devore 2006]. A hipótese para utilização dos testes paramétricos foi que os dados tinham uma distribuição Binomial, pois foi considerado nesse trabalho que cada usuário que entrou na Agência Embrapa no período da pesquisa, ao acessar uma das páginas, tomou uma decisão de forma independente dos demais. É importante salientar que no caso de a hipótese alternativa ser a simples diferença entre as taxas de rejeição, isto é $H_1 : p_1 \neq p_2$, os dois testes são equivalentes. No entanto, o teste Z foi utilizado pois é possível testar a hipótese $H_1 : p_1 > p_2$. Assim, nas linhas destacadas da Tabela III, tem-se as páginas que apresentaram variação na taxa de rejeição e que, em pelo menos um dos testes estatísticos, essa variação foi significativa.

Table II. Regras com suas respectivas páginas antecedentes, número total de recomendações e o total de links na página.

Regras	Antecedente	Recomendações	Total de Links
1	Fabricação do açúcar	1	44
2	Extração	1	41
3,8,11,12	Processamento da cana-de-açúcar	4	49
4,1	Variedades	2	45
5	Custos e rentabilidade	1	39
6,7,19	Cachaça	3	49
9	Queima	1	46
13,21	Plantio	2	44
14	Açúcar	1	37
15	Correção e adubação	1	52
16	Doenças	1	49
17	Qualidade de matéria-prima	1	41
18,25	Abertura	2	25
20	Diagnose das necessidades nutricionais	1	49
22	Análise de solo	1	48
23	Preparo do solo	1	43
24	Implicações	1	44
26	Doenças fúngicas	1	49
27,28	Meio ambiente	2	42

Observando-se os dados da Tabela III, vê-se que em algumas páginas o sistema de recomendação não teve impacto na taxa de rejeição. Para as páginas de Extração, Processamento da cana-de-açúcar, Cachaça, Queima, Plantio, Doenças, Análise do solo, Preparo do solo e Meio ambiente, não houve variação estatisticamente significativa para ambos os testes. Para o caso de algumas páginas, como a de "Diagnose das necessidades nutricionais", "Queima" e "Implicações", houve um número pequeno de sessões onde estas eram a primeira página visitada.

Table III. Estatísticas sobre a resposta ao sistema de recomendação para as recomendações.

Página	Taxa Rejeição (antes)	Taxa Rejeição (depois)	Qui-Quadrado (p-valor)	Z (p-valor)
Fabricação do açúcar	**0.6780**	**0.7757**	**0.0000**	**0.0000**
Extração	0.8984	0.9153	0.2922	0.1461
Processamento da cana-de-açúcar	0.7747	0.7743	0.9775	0.4887
Variedades	**0.9392**	**0.9003**	**0.0002**	**0.0001**
Custos e rentabilidade	**0.7463**	**0.4832**	**0.0000**	**0.0000**
Cachaça	0.9437	0.9414	0.7278	0.3639
Queima	0.8949	0.8727	0.6026	0.3013
Plantio	0.8421	0.8492	0.4904	0.2452
Açúcar	**0.8455**	**0.9078**	**0.0013**	**0.0007**
Correção e adubação	**0.9580**	**0.9250**	**0.0010**	**0.0005**
Doenças	0.5497	0.5776	0.3602	0.1801
Qualidade de matéria-prima	**0.9624**	**0.9152**	**0.0000**	**0.0000**
Abertura	**0.2849**	**0.2373**	**0.0000**	**0.0000**
Diagnose das necessidades nutricionais	0.6268	0.5000	0.3005	0.1503
Análise de solo	0.9257	0.8849	0.0160	0.0080
Preparo do solo	0.7031	0.7537	0.0113	0.0056
Implicações	**0.9552**	**0.8571**	**0.0911**	**0.0456**
Doenças fúngicas	**0.9718**	**0.9514**	**0.0674**	**0.0337**
Meio ambiente	0.9638	0.9810	0.3484	0.1742

Por outro lado, aproximadamente metade das páginas apresentaram variações estatisticamente significativas (p-valor $< 0{,}05$, em pelo menos um dos testes). Destas, a maioria teve a taxa de rejeição diminuída com exceção das páginas "Fabricação do açúcar" e "Açúcar". Especialmente, no caso das páginas de "Fabricação do açúcar" e "Açúcar", a diferença positiva pode ser interpretada à luz do volume de acessos: como estas são duas das páginas mais acessadas e com altas taxas de rejeição, isso

pode indicar que os usuários já encontraram as informações desejadas nas próprias páginas. Outra hipótese é que a quantidade de dados usada antes da disponibilização do sistema foi muito maior do que a quantidade de dados utilizada após a implantação do sistema, para essas páginas. Assim, essa variação pode ser resultado dessa desproporção, que pode diminuir com a coleta de mais dados.

5. CONCLUSÃO

Neste trabalho, foi apresentado um sistema de recomendação para informações tecnológicas agrícolas. Sua validação se deu por meio de um estudo de caso relacionado à cultura da cana-de-açúcar. Os resultados demonstraram que a partir da implantação desse sistema, houve um impacto positivo na usabilidade do portal da Agência Embrapa. O sistema implantado permite aos usuários acessar mais informações sem a necessidade de realizar novas buscas, o que é muito interessante para usuários não especialistas e menos experientes (ex.: produtores rurais e extensionistas).

Vinte páginas relacionadas à cultura da cana-de-açúcar receberam recomendações, e destas, mais da metade tiveram a taxa de rejeição diminuída com significância estatística. A base de conhecimento, composta de 28 regras, atua como uma forma de resumo, indicando quais são os links mais importantes nas páginas do portal sobre a cana-de-açúcar.

REFERENCES

Billsus, D., Brunk, C., and Evans, C. Adaptive interfaces for ubiquitous web access. *Communications of the ACM - The Adaptive Web* 45 (5): 34–38, 2002.

Burke, R. Knowledge-based recommender systems. *Encyclopaedia of Library and Information Systems* 69 (32), 2000.

Devore, J. L. *Probabilidade e Estatística para Engenharia e Ciências*. Thomson Learning, São Paulo, SP, Brasil, 2006.

Han, J., Kamber, M., and Pei, J. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2011.

Jorge, A., Alves, M. A., and Azevedo, P. Recommendation with association rules: A web mining application. In *in Proceedings of Data Mining and Warehousing, Conference of Information Society 2002, Eds. D. Mladenic and M. Grobelnik, Josef Stefan Institute*, 2002.

Kazienko, P. Mining indirect association rules for web recommendation. *Int. J. Appl. Math. Comput. Sci.* 19 (1): 165–186, Mar., 2009.

Kumar, A; Thambidurai, P. Collaborative Web Recommendation Systems -A Survey Approach. *Global Journal of Computer Science and Technology* 9 (5), 2010.

Linden, G., Smith, B., and York, J. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing* 7 (1): 76–80, Jan., 2003.

Lucas, J. P., Luz, N., Moreno, M. N., Anacleto, R., Figueiredo, A. A., and Martins, C. A hybrid recommendation approach for a tourism system. *Expert Systems with Applications* 40 (9): 3532 – 3550, 2013.

Miller, B. N., Albert, I., Lam, S. K., Konstan, J. A., and Riedl, J. MovieLens unplugged. In *Proceedings of the 8th international conference on Intelligent user interfaces - IUI '03*. ACM Press, New York, New York, USA, pp. 263, 2003.

Paranjape-Voditel, P. and Deshpande, U. A stock market portfolio recommender system based on association rule mining. *Applied Soft Computing* 13 (2): 1055 – 1063, 2013.

Perkowitz, M. and Etzioni, O. Towards adaptive web sites: Conceptual framework and case study. *Artificial Intelligence* 118 (1–2): 245 – 275, 2000.

Ricci, F., Rokach, L., Shapira, B., and Kantor, P. B., editors. *Recommender Systems Handbook*. Springer, 2011.

Sculley, D., Malkin, R. G., Basu, S., and Bayardo, R. J. Predicting bounce rates in sponsored search advertisements. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '09*. ACM Press, New York, New York, USA, pp. 1325, 2009.

Yang, H. and Parthasarathy, S. On the use of constrained associations for web log mining. In *WEBKDD 2002 - Mining Web Data for Discovering Usage Patterns and Profiles*, O. Zaïane, J. Srivastava, M. Spiliopoulou, and B. Masand (Eds.). Lecture Notes in Computer Science, vol. 2703. Springer Berlin Heidelberg, pp. 100–118, 2003.

Yang, H.-L. and Tang, J.-H. A three-stage model of requirements elicitation for Web-based information systems. *Industrial Management & Data Systems* 103 (6): 398–409, 2003.

UDRB: Uma Nova Heurística Eficaz para Deduplicação de Referências Bibliográficas

Sérgio Canuto[1], Guilherme Dal Bianco[2], Marcos André Gonçalves[1], Jussara Almeida[1], Thierson Couto[3]

[1] Universidade Federal de Minas Gerais, Brasil
{sergiodaniel, mgoncalv, jussara}@dcc.ufmg.br
[2] Universidade Federal do Rio Grande do Sul, Porto Alegre, RS, Brasil
gbianco@inf.ufrgs.br
[3] Universidade Federal de Goiás, Goiânia, GO, Brasil
thierson@inf.ufg.br

Abstract. Publicações científicas normalmente contêm referências bibliográficas a trabalhos anteriores. Tais referências são usadas como fonte de informação para bibliotecas digitais, contribuindo com recursos de busca, navegação e estimativa de qualidade das obras. Neste contexto, frequentemente ocorre um problema que consiste em identificar se duas referências representam uma mesma publicação, conhecido como deduplicação de referências bibliográficas (DRB). Soluções para DRB podem ser divididas em supervisionadas (dependem de um conjunto de treinamento) e não supervisionadas (baseados em heurísticas). Com objetivo de evitar o acentuado custo manual de criação de um conjunto de treinamento, propomos neste trabalho uma heurística não supervisionada para DRB, denominada UDRB. Os experimentos em bases reais mostraram que a heurística proposta alcançou ganhos de mais de 7% em relação ao método não supervisionado estado-da-arte, e eficácia similar as de métodos supervisionados na maioria dos casos, sem a necessidade da dispendiosa tarefa de rotulação manual.

Categories and Subject Descriptors: H.3.3 [**Information Search and Retrieval**]: Information Search and Retrieval

Keywords: Deduplicação de Referências, Desambiguação

1. INTRODUÇÃO

O surgimento da Web tem permitido a coleta de uma grande quantidade de informações bibliográficas sobre artigos científicos. Entre as várias informações disponíveis, destacam-se as referências bibliográficas que aparecem geralmente nas últimas seções dos artigos. Quando consideradas coletivamente, as referências bibliográficas permitem mensurar a importância de uma determinada obra, por exemplo, por meio da contagem do número de trabalhos de uma coleção que possuem referência bibliográfica a tal obra. Alguns índices de reputação de obras científicas baseiam-se nesta contagem, como é o caso do *fator de impacto*, muito utilizado para medir a importância de revistas indexadas no meio científico[1].

As métricas usadas para avaliar o impacto de uma publicação são de grande importância prática, contudo, existe um problema que antecede o seu cálculo, o qual consiste em identificar quais referências bibliográficas distintas se referem a uma mesma obra científica. No contexto de bibliotecas digitais, esse problema é denominado *deduplicação de referências bibliográficas* - DRB.

É comum encontrarmos em uma biblioteca digital centenas de milhares de referências bibliográficas e, nesse contexto, uma solução manual não escala. É necessário o desenvolvimento de soluções automáticas que possam ser executadas por computadores. A dificuldade de se conseguir soluções automáticas para DRB se deve às seguintes razões: (i) referências a uma mesma obra podem estar escritas de formas muito diferentes entre si, normalmente por questões de estilo. Por exemplo, abreviações de iniciais nos nomes de autores e a presença (ou ausência) de numeração de páginas são fatores que podem diferenciar referências que, na verdade, correspondem à mesma obra; (ii) referências a publicações distintas mas que possuem o texto muito parecido. Um exemplo típico ocorre quando um autor publica uma versão preliminar de seu trabalho em uma conferência e posteriormente publica uma versão estendida do mesmo trabalho em um periódico.

[1] http://thomsonreuters.com/journal-citation-reports

Para lidar com esses problemas, dois grupos de métodos para DRB foram propostos: *supervisionados* e *não-supervisionados*. Os métodos DRB supervisionados [Bilenko and Mooney 2003; Borges et al. 2012] dependem de um conjunto de treinamento informativo para a identificação dos padrões presentes na base de dados. Se o conjunto de treinamento não for suficientemente informativo, o método provavelmente não atingirá a eficácia esperada. Já os métodos não-supervisionados [Borges et al. 2011; McCallum et al. 2000; Lawrence et al. 1999] utilizam heurísticas para identificar os padrões de duplicatas, como por exemplo, medidas de similaridade entre os campos autores, título e ano de duas referências [Borges et al. 2011]. A principal vantagem das técnicas não supervisionadas é a capacidade de atingir uma eficácia muitas vezes comparável a de métodos supervisionados sem a necessidade da dispendiosa tarefa de criação do conjunto de treinamento.

Neste trabalho propomos um método não supervisionado para DRB: o UDRB. Tal abordagem adiciona um conjunto de melhorias na heurística não-supervisionada proposta em [Borges et al. 2011], considerada estado-da-arte, aprimorando assim sua eficácia. Nossa experimentação mostrou que o UDRB atinge uma eficácia equivalente a dos melhores métodos supervisionados recentemente descritos em [Borges et al. 2012], sem a necessidade da criação de uma amostra manualmente rotulada. Mostramos também que a estratégia UDRB sempre é competitiva, mesmo quando as estratégias supervisionadas são aprimoradas com um novo conjunto de atributos aqui propostos. Portanto, a segunda contribuição do trabalho é a melhoria nos métodos supervisionados de [Borges et al. 2012] através da extensão do conjunto de seus atributos com diversas medidas de similaridade, como Levenshtein, Jaro-Winkler, Jaccard, *softTFIDF* [Cohen et al. 2003], bem como medidas utilizadas em trabalhos de deduplicação anteriores [McCallum et al. 2000; Lawrence et al. 1999]. Com essas medidas, é possível capturar evidências de similaridade que consideram abreviações, erros tipográficos, inversões de termos, e especificidades de cada campo das referências.

A última contribuição deste trabalho consiste na análise do impacto na eficácia da deduplicação de quatro fatores normalmente considerados em métodos de deduplicação: a segmentação das referências em campos [Isaac Councill 2008], a blocagem [McCallum et al. 2000] (que reduz a quantidade de comparações entre pares), a identificação de referências associadas à mesma publicação, e a geração de uma solução consistente (i.e., que lida com o problema da transitividade [Huang et al. 2006], que surge se o método classifica tanto a e b quanto b e c como correferentes, mas a e c como não correferentes).

Este artigo é organizado da seguinte forma. A Seção 2 apresenta trabalhos relacionados. A Seção 3.1 apresenta o método não supervisionado para DRB do estado-da-arte seguido pela nossa proposta. O conjunto de atributos propostos para deduplicação supervisionada é apresentado na Seção 4. Experimentos e resultados são discutidos na Seção 5, enquanto a Seção 6 conclui o artigo.

2. TRABALHOS RELACIONADOS

Até pouco tempo, as soluções para a DRB utilizavam diretamente o texto das referências no processo de deduplicação. Entretanto, esta abordagem leva a resultados que não são satisfatórios devido à grande quantidade de ruído existente nos textos. Recentemente, técnicas de extração de informação [Isaac Councill 2008] têm sido empregadas para separar o texto de cada referência bibliográfica em seus campos constituintes, tais como: autores, título, veículo de publicação, data, etc. Desse modo, as soluções para DRB podem trabalhar com campos correspondentes de duas referências bibliográficas, obtendo mais precisão no processo de deduplicação [McCallum et al. 2000].

As técnicas supervisionadas (que utilizam um conjunto de dados de treinamento) apresentam bons resultados na deduplicação de referências. Em tal abordagem, uma função de deduplicação é gerada através de estratégias como SVM treinado com atributos baseados na co-ocorrência de termos [Bilenko and Mooney 2003], bem como *Naive Bayes*, SVM, e Árvores de Decisão (AD) usando similaridades entre campos das referências como atributos [Borges et al. 2012]. Neste trabalho, aprimoramos os resultados recentes dos métodos supervisionados descritos em [Borges et al. 2012]. Para tal, empregamos abordagens como SVM, AD e Florestas Randômicas (FR), e estendemos o conjunto

de atributos explorado para incluir também métricas relacionadas com co-ocorrência de termos e similaridades entre campos das referências.

As soluções não supervisionadas [Borges et al. 2011; McCallum et al. 2000; Lawrence et al. 1999] identificam se referências estão associadas à mesma publicação através de heurísticas. A melhor heurística proposta em [Lawrence et al. 1999] consiste em quantificar termos em comum entre referências, ignorando a estrutura de campos da referência. Para utilizar essa estrutura de campos, [McCallum et al. 2000] combina linearmente as similaridades entre os campos das referências. Recentemente, o trabalho [Borges et al. 2011] propõe um conjunto de regras para combinar similaridades específicas correspondentes a cada campo das referências, apresentando resultados bastante positivos. Entretanto, tal heurística considera apenas os campos ano, autores e título, e falha ao deduplicar os casos em que algum dos campos da referência não está presente. Para superar esse problema, o método não supervisionado aqui proposto complementa o método em [Borges et al. 2011] utilizando, em adição aos campos, o texto completo das referências bibliográficas. O método proposto melhora sensivelmente a deduplicação quando os campos não são corretamente identificados ou estão ausentes.

3. HEURÍSTICAS PARA DEDUPLICAÇÃO NÃO-SUPERVISIONADA

Nesta seção são apresentados os métodos *MetadataMatch* e UDRB que correspondem, respectivamente, ao estado-da-arte para deduplicação de referências bibliográficas de modo não supervisionado, e à heurística não supervisisionada proposta.

3.1 Baseline: Deduplicação MetadataMatch

MetadataMatch propõe a utilização de um conjunto de regras, definidas de acordo com o conteúdo de cada campo, para identificar correferências de forma automática. Tal heurística considera que um pré-processamento foi otimamente realizado para segmentar em campos. Mais especificamente, o Algoritmo 1 detalha a heurística utilizando três medidas para verificar se duas referências a e b descrevem a mesma publicação. A primeira medida consiste na diferença entre o ano de publicação de tais referências, denotada por $|Year(a) - Year(b)|$. Tal diferença deve ser menor que um limiar t_Y para que a e b sejam consideradas correferentes (Linha 2). A segunda medida, denotada pela função *NameMatch*, mensura a similaridade entre os autores de a e b (Linha 3). Essa função compara as iniciais de todos autores de a com as iniciais de todos autores de b, considerando que tais iniciais podem aparecer invertidas. Por fim, o título das referências é comparado usando a medida de similaridade Levenshtein normalizada [Borges et al. 2011]. Caso essas três medidas respeitem os seus respectivos limiares t_Y, t_N e t_L, as referências a e b são consideradas correferentes (Linha 5).

Algorithm 1 Heurística MetadataMatch proposta por [Borges et al. 2011].

```
1: function MetadataMatch(a, b, t_Y, t_N, t_L)
2:     if |Year(a) − Year(b)| ≤ t_Y then
3:         if NameMatch(Authors(a), Authors(b)) ≥ t_N then
4:             if Levenshtein(Title(a), Title(b)) ≥ t_L then
5:                 return 1
6:     return 0
```

3.2 Nova Heurística: Deduplicação UDRB

A heurística de deduplicação UDRB, proposta neste trabalho, visa estender o método *MetadataMatch* ao considerar também todo o texto das referências, isso é, o texto original não segmentado. Como o pré-processamento para a extração automática das citações pode falhar, ou os campos podem nem mesmo existir no texto da referência, a heurística *MetadataMatch* pode atingir uma qualidade aquém da desejada. Neste contexto, a heurística UDRB utiliza o texto completo das referências, e considera os casos em que não é possível extrair os campos de uma referência com precisão.

O Algoritmo 2 difere do *MetadataMatch* por incluir os conectivos lógicos *OR* para que as medidas de similaridade sobre os campos ano, autores e título sejam desconsideradas caso não tenham sido extraídos. Além disso, foi incluída a medida *Jaccard* [Cohen et al. 2003] para comparar todo o texto de duas referências (Linha 5). Logo, todas as palavras presentes no texto original da referência a podem

ser comparadas as da referência b sem considerar a divisão em campos. Para efeito de comparação, foi criado um *baseline* denominado Jaccard-DRB que corresponte ao uso da Linha 5 somente.

Algorithm 2 UDRB: nova heurística de deduplicação.

```
1: function UDRB(a, b, t_Y, t_N, t_L, t_F)
2:     if |Year(a) − Year(b)| ≤ t_Y  OR  Year(a) = ∅ OR  Year(b) = ∅  then
3:         if Namematch(Authors(a), Authors(b)) ≥ t_N  OR  Authors(a) = ∅ OR  Authors(b) = ∅ then
4:             if Levenshtein(Title(a), Title(b)) ≥ t_L  OR  Title(a) = ∅ OR  Title(b) = ∅ then
5:                 if Jaccard(FullText(a), FullText(b)) ≥ t_F then
6:                     return 1
7:     return 0
```

4. ATRIBUTOS PARA DEDUPLICAÇÃO SUPERVISIONADA

Nesta seção é proposto um conjunto de atributos específicos para deduplicação de referências. Tais atributos podem ser usados com estratégias supervisionadas conhecidas de aprendizagem de máquina, como SVM, Árvores de Decisão (AD) e Florestas Randômicas (FR). Recentemente, as estratégias SVM e AD foram usadas em [Borges et al. 2012]. Entretanto, tais métodos de aprendizagem foram treinados com um conjunto reduzido de 8 atributos, destacados em negrito na Tabela I. Tais atributos consideram a quantidade de autores de cada referência, a diferença entre essas quantidades, e as medidas usadas pela heurística *MetadataMatch* (em conjunto com suas versões normalizadas).

Neste trabalho, estendemos essas medidas com um conjunto mais robusto de atributos capazes de extrair informações relevantes para estratégias supervisionadas de DRB. Vale ressaltar que não é do nosso conhecimento que o conjunto completo de atributos proposto neste trabalho tenha sido utilizado em algum trabalho anterior. Mais especificamente, a Tabela I ilustra o conjunto de atributos que propomos para complementar o conjunto usado em [Borges et al. 2012]. A medida softTFIDF [Cohen et al. 2003] foi usada na maioria dos campos, visto que foi criada para lidar com o peso TFIDF das palavras considerando erros tipográficos e abreviações. Para considerar a ordem das palavras no texto das referências, foi utilizada a estratégia *Jaccard2gram*, como sugerido em [Lawrence et al. 1999]. Essa medida fornece um índice proporcional à quantidade de pares de palavras consecutivas que aparecem em ambas referências. As medidas *Jaro-Winkler* [Cohen et al. 2003] e *SameType* foram usadas especificamente para lidar, respectivamente, com os campos título e veículo de divulgação. *Jaro-Winkler* lida com o fato de que um título pode aparecer incompleto, e *SameType* retorna 1 caso ambas referências são pertencentes ao mesmo tipo de veículo (isto é, ambas foram publicadas em um periódico, por exemplo) e 0 caso contrário.

Campo	Medida
Ano	**|Year(a)-Year(b)|**
Páginas	Levenshtein(Pages(a), Pages(b))
Instituição	softTFIDF(Instituition(a), Instituition(b))
Localização	softTFIDF(Location(a), Location(b))
Veículo	SameType(Venue(a), Venue(b)), softTFIDF(Venue(a), Venue(b))
Autores	**NameMatch(Authors(a),Authors(b)),Norm_NameMatch(Authors(a),Authors(b)),Num(Authors (a)), Num(Authors(b)), |Num(Authors(a))-Num(Authors(b))|**,softTFIDF(Authors(a),Authors(b))
Título	**Levenshtein(Title(a), Title(b)), Norm_Levenshtein(Title(a), Title(b))**, Jaccard(Title(a), Title(b)), softTFIDF(Title(a), Title(b)), Jaccard2gram(Title(a), Title(b)), Jaro-Winkler(Title(a), Title(b))
Texto todo	Jaccard(FullText(a),FullText(b)),Jacard2gram(FullText(a),FullText(b)),SoftTFIDF(FullText(a),FullText(b))

Tabela I. Conjunto de medidas de similaridade para DRB. As medidas em negrito foram usadas em [Borges et al. 2012].

5. RESULTADOS EXPERIMENTAIS

Nesta seção, avaliamos as abordagens não supervisionadas UDRB e *MetadataMatch*, bem como os métodos supervisionados SVM, AD e FR executados com os atributos de [Borges et al. 2012], denominados respectivamente, SVM-Borges, AD-Borges e FR-Borges, ou executados com o conjunto de atributos propostos, denominados SVM-DRB, AD-DRB e FR-DRB. A eficácia dos resultados foi mensurada pela medida F1 [Borges et al. 2011]. Todos os experimentos foram feitos usando o procedimento de validação cruzada em 4 partições (*folds*), e portanto, apresentamos o F1 médio destas 4 execuções. A significância estatística foi confirmada por meio de um teste-t pareado, com 95% de confiança. Foram utilizadas duas bases de dados reais: a coleção CiteSeer, com 1563 citações a 906 publicações [Lawrence et al. 1999], e a coleção Cora com 2191 citações a 305 publicações.

5.1 Parametrização

Foi adotada a implementação LIBLINEAR do classificador SVM. O parâmetro de regularização C foi escolhido entre onze valores, de 2^{-5} até 2^{15} através da estratégia de validação cruzada em 4 partições. As árvores dos métodos FR e AD foram geradas sem poda. Como sugerido na literatura para o método FR, utilizamos a quantidade de árvores $T > 100$ e a quantidade de atributos considerados a cada divisão $M = log_2 F$, em que F é a quantidade total de atributos.

Quanto a heurística *MetadataMatch*, foram testados os valores 0, 1 e 2 para o parâmetro t_Y, e valores de 0,4 a 0,7, variando de 0,05 em 0,05, para os parâmetros t_L e t_N. Para a nova heurística UDRB (e para Jaccard-DRB), foram testados também valores de 0,2 a 0,6 para o parâmetro adicional t_F. Foi feito um projeto fatorial completo com quatro replicações, considerando todas as possíveis combinações de valores considerados para os parâmetros. A Tabela II apresenta os parâmetros que levaram aos melhores resultados no treino.

Abordagem	Parâmetros
MetadataMatch	Para Cora, t_L=0,6, t_Y=1 e t_N=0,5. Para CiteSeer, t_L=0,55, t_Y=0 e t_N=0,5
UDRB	Para Cora, t_L=0,65, t_Y=1, t_N=0,5 e t_F=0,35. Para CiteSeer, t_L=0,5, t_Y=1, t_N=0,5 e t_F=0,35
Jaccard-DRB	Para as coleções CiteSeer e Cora, t_F=0,5
SVM-DRB e SVM-Borges	Para as coleções CiteSeer e Cora $C = 2$, Kernel linear
FR-DRB e FR-Borges	Para as coleções CiteSeer e Cora, $T = 500$, $M = log_2 F$
AD-DRB e AD-Borges	Para as coleções CiteSeer e Cora, árvore gerada sem poda

Tabela II. Parâmetros adotados nos experimentos.

5.2 Eficácia das Abordagens de Deduplicação

Nos experimentos apresentados na Tabela III foi realizado um pré-processamento para corrigir casos em que o campo correspondente ao ano da publicação não existe nos metadados. Para *MetadataMatch**, esse pré-processamento foi feito como descrito em [Borges et al. 2011]. Para lidar com as demais abordagens da Tabela III, propomos extrair o ano através da busca por um número entre 1950 e 2013 em todo o texto da referência. Com essa pequena modificação, *MetadataMatch* superou em 4% sua versão *MetadataMatch** com pré-processamento original.

O método UDRB supera MetadataMatch com significância estatística, com ganhos de 7% e 9% nas coleções Cora e CiteSeer, respectivamente. Tais melhorias advêm da exploração do texto completo das referências e da consideração dos casos em que campos das referências não foram devidamente segmentados. Em geral, UDRB também mostrou uma eficácia comparável (e em alguns casos superior) à dos métodos supervisionados, que exigem a dispendiosa tarefa de rotulação de pares de referências. Especificamente, UDRB apresentou ganhos de 3% a 6% sobre métodos supervisionados SVM-Borges, AD-Borges e FR-Borges na coleção Cora e empatou com os mesmos na coleção Citeseer. UDRB também empatou com SVM-DRB, AD-DRB e FR-DRB na coleção Citeseer, mas na coleção Cora os métodos AD-DRB e FR-DRB apresentaram ganhos de cerca de 3% sobre UDRB. Tais ganhos são pequenos, tendo em vista o custo adicional de rotulamento.

Abordagens da Literatura	Cora	CiteSeer	Abordagens Propostas	Cora	CiteSeer
MetadataMatch*	$83,3 \pm 2,0 \downarrow$	$84,2 \pm 2,0 \downarrow$	Jaccard-DRB	$88,2 \pm 4,0 \downarrow$	$85,9 \pm 2,6 \downarrow$
MetadataMatch	$86,9 \pm 0,9 \downarrow$	$85,0 \pm 3,1 \downarrow$	UDRB	$94,9 \pm 1,1$	$91,2 \pm 3,3$
SVM-Borges	$89,2 \pm 4,1 \downarrow$	$91,5 \pm 5,9 \updownarrow$	SVM-DRB	$94,7 \pm 2,2 \updownarrow$	$92,5 \pm 7,1 \updownarrow$
AD-Borges	$91,7 \pm 2,6 \downarrow$	$88,6 \pm 10,3 \updownarrow$	AD-DRB	$97,0 \pm 1,2 \uparrow$	$94,0 \pm 4,2 \updownarrow$
FR-Borges	$90,4 \pm 4,5 \downarrow$	$92,0 \pm 5,0 \updownarrow$	FR-DRB	$97,7 \pm 1,0 \uparrow$	$95,0 \pm 5,6 \updownarrow$

Tabela III. Eficácia média, em F1, das diferentes abordagens para deduplicação (e intervalos com 95% de confiança). $\downarrow$, $\uparrow$, $\updownarrow$ indicam, respectivamente, perdas, ganhos e empates com UDRB. *Executada com pré-processamento da literatura.

Como esperado, houve ganhos significativos dos métodos SVM-DRB, AD-DRB e FR-DRB sobre suas respectivas versões da literatura SVM-Borges, AD-Borges e FR-Borges na coleção Cora, pois os métodos da literatura não utilizam diversos dos atributos descritos na Tabela I. Na coleção CiteSeer, os ganhos não foram significativos devido à grande variação dos resultados, que reflete nos largos intervalos de confiança (cerca de três vezes maiores que os da Cora). Tal variação deve-se ao fato que a coleção não foi dividida aleatoriamente, como a Cora. Ao invés disso, as referências foram particionadas por área, o que causa diferença na distribuição dos dados.

5.3 Análise dos Fatores que Influenciam a Eficácia da Deduplicação

Com o objetivo de avaliar a importância e a interação entre quatro fatores que influenciam a eficácia da deduplicação, executamos um projeto fatorial $2^k r$ [Jain 1991] para analisar o impacto de cada um deles na variação dos dados. Cada fator pode assumir um de dois níveis, que correspondem às possíveis variações consideradas para tal fator, conforme especificado na Tabela IV.

Fator	Níveis do fator	
Segmentação	**Seg. Parscit**	Experimentos com a segmentação automática ParsCit [Isaac Councill 2008].
	Seg. Original	Experimentos usando a segmentação original da coleção.
Blocagem	**Com Bloc.**	Experimentos usando o método de blocagem *canopy* [McCallum et al. 2000].
	Sem Bloc.	Experimentos sem usar estratégia de blocagem.
Deduplicação	**MetadataMatch**	Experimentos usando *MetadataMatch* para deduplicação.
	UDRB	Experimentos usando a heurística UDRB proposta.
Transitividade	**Com Trans.**	Experimentos resolvem o problema de transitividade [Huang et al. 2006].
	Sem Trans.	Experimentos sem lidar com o problema de transitividade

Tabela IV. Variação dos fatores que impactam na eficácia da deduplicação.

De acordo com o projeto fatorial desenvolvido, cujos resultados são mostrados na Tabela V, os fatores que mais impactam a eficácia da deduplicação são a estratégia de deduplicação adotada, responsável por 60% da variação nos dados, e o método de segmentação adotado, responsável por 8% da variação. Em geral, a abordagem UDRB se saiu muito melhor que o método *MetadataMatch* para deduplicação, o que explica o papel central da estratégia usada na eficácia dos resultados. O fator segmentação também se mostrou importante, pois a utilização da segmentação automática ParsCit introduz erros que se mostraram, em geral, prejudiciais aos métodos. Além destes dois fatores primários, a interação entre a deduplicação e os métodos de blocagem usados também se mostrou importante, explicando 10% da variação nos dados. Isto se deve ao fato de que a eficácia do método *MetadataMatch* sempre é aprimorada com a utilização da estratégia de blocagem, pois a blocagem é baseada na comparação de todo texto das referências com a medida de similaridade Jaccard. Logo, a inclusão do blocagem no MetadataMatch de certa forma inclui também essa medida de similaridade no método. Os demais fatores e interações entre fatores, quando significativos, foram responsáveis por variações desprezíveis nos resultados.

		Seg. ParsCit		Seg. Original	
		Com bloc.	Sem bloc.	Com bloc.	Sem bloc.
UDRB	Com trans.	$91,7 \pm 1,1$	$92,3 \pm 1,1$	$94,9 \pm 1,1$	$94,4 \pm 1,3$
	Sem trans.	$92,0 \pm 1,0$	$91,8 \pm 1,0$	$93,8 \pm 1,2$	$93,4 \pm 1,0$
MetadataMatch	Com trans.	$88,6 \pm 3,2$	$85,5 \pm 1,5$	$90,3 \pm 4,9$	$86,9 \pm 0,9$
	Sem trans.	$90,5 \pm 1,8$	$85,6 \pm 1,2$	$91,5 \pm 3,0$	$86,3 \pm 0,9$

Tabela V. Eficácia dos fatores que impactam na deduplicação da coleção Cora (e intervalos com 95% de confiança).

6. CONCLUSÃO E TRABALHOS FUTUROS

Neste artigo propomos um novo método não-supervisionado para deduplicação de referências, que se mostrou significativamente superior à heurística de deduplicação do estado-da-arte e com eficácia equivalente aos melhores métodos supervisionados considerados. A avaliação da performance dos métodos usando diferentes estratégias de segmentação é um trabalho atualmente em andamento.

REFERÊNCIAS

Bilenko, M. and Mooney, R. Adaptive duplicate detection using learnable string similarity measures. In *KDD*, 2003.

Borges, E. N., Becker, K., Heuser, C. A., and Galante, R. A classification-based approach for bibliographic metadata deduplication. *IADIS*, 2012.

Borges, E. N., de Carvalho, M., Galante, R., Gonçalves, M. A., and Laender, A. An unsupervised heuristic-based approach for bibliographic metadata deduplication. *Inf. Process. Manage.*, 2011.

Cohen, W. W., Ravikumar, P. D., and Fienberg, S. E. A comparison of string distance metrics for name-matching tasks. In *IIWeb*, 2003.

Huang, J., Ertekin, S., and Giles, C. L. Efficient name disambiguation for large-scale databases. In *PKDD*, 2006.

Isaac Councill, C. Lee Giles, K. Parscit: An open-source crf reference string parsing package. In *LREC*, 2008.

Jain, R. K. *The Art of Computer Systems Performance Analysis*. Wiley, 1991.

Lawrence, S., Giles, C. L., and Bollacker, K. D. Autonomous citation matching. In *AGENTS*, 1999.

McCallum, A., Nigam, K., and Ungar, L. H. Efficient clustering of high-dimensional data sets with application to reference matching. In *KDD*, 2000.

*Open*SBBD: Usando *Linked Data* para Publicação de Dados Abertos sobre o SBBD

Mateus Gondim Romão Batista, Bernadette Farias Lóscio

Centro de Informática, Universidade Federal de Pernambuco, Brasil
{mgrb, bfl}@cin.ufpe.br

Abstract. The Semantic Web was designed to expand the current Web we know, enabling the large amount of data available on the Web to be processed not only by humans, but also by machines. Along this process, the Semantic Web gathered new technologies which allow the modeling and description of data embebbed with semantic definitions. In addition to the adoption of these technologies, the concept of Linked Data emerged. Linked Data is a set of principles and techniques created with the purpose of interlinking data from distinct data sources. On the other hand, the Open Data Movement has encouraged data publishing on the Web, without restrictions from copyright, patents or other mechanisms of control. The combination of Semantic Web technologies, the principles of Linked Data and the Open Data movement has played a major role in data publishing. In this context, this work proposes: I) the creation of an interlinked open dataset, available in RDF and consistent with the principles of Linked Data, about the 27 editions of the Brazilian Symposium on Databases; II) the creation of a SPARQL Endpoint available to be queried using SPARQL; III) The development of a Web application with the purpose of providing more user-friendly views about this dataset.

Resumo. A Web Semântica foi projetada para expandir a Web que conhecemos atualmente, possibilitando que a imensa quantidade de dados disponíveis na Web possa ser compreendida não só por pessoas, mas também por máquinas. Neste processo, a Web Semântica agregou novas tecnologias que permitem a modelagem e descrição de dados com definições semânticas associadas. Em conjunto com a adoção dessas tecnologias, surgiu o conceito de *Linked Data*, um conjunto de princípios e técnicas para publicação de dados estruturados na Web, criado com o objetivo de possibilitar a interligação de dados de fontes de dados distintas. Por outro lado, o movimento de Dados Abertos (Open *Data*) tem incentivado a disponibilização de dados na Web sem restrições de copyright, patentes e outros mecanismos de controle. A combinação das tecnologias da Web Semântica, os princípios de *Linked Data* e o movimento de Dados Abertos tem desempenhado um papel importante na publicação de dados na Web. Nesse contexto, este artigo propõe: i) a criação de um conjunto de dados abertos, disponível em RDF e de acordo com os princípios de *Linked Data*, sobre as 27 edições do Simpósio Brasileiro de Banco de Dados; ii) a disponibilização de um *SPARQL Endpoint* para a execução de consultas SPARQL sobre os dados e metadados do SBBD; iii) o desenvolvimento de uma aplicação Web com o intuito de oferecer visões mais amigáveis destes dados.

Keywords: Semantic Web, Linked Data, Open Data

1. INTRODUÇÃO

A Web Semântica é a extensão da *World Wide Web* que permite às pessoas compartilharem conteúdo além dos limites de aplicações e *websites* [10]. Encorajando a inclusão de conteúdo semântico em páginas Web, a

Web Semântica visa converter a Web atual dominada por documentos estruturados e semi-estruturados em uma "Web de dados". Para tornar a Web Semântica uma realidade, uma série de novas tecnologias foram adotadas, como RDF, OWL e XML. Conectado com estes novos atores da Web, surgiu um conjunto de princípios e tecnologias chamado *Linked Data*, que de uma maneira geral, refere-se a empregar o RDF e o *Hypertext Transfer Protocol* (HTTP) para publicar dados estruturados na Web, interligando dados de diferentes fontes de dados [4]. O movimento de Dados Abertos (Open Data), que incentiva a disponibilização de dados a todos, também contribui para a disponibilização de conjuntos de dados que podem ser interligados no formato *Linked Data*.

Seguindo estes conceitos, este trabalho aborda o problema de criação de uma base de dados e metadados sobre o Simpósio Brasileiro de Banco de Dados (SBBD), o maior evento na América Latina para a apresentação e discussão de resultados de pesquisa relacionados à área de Banco de Dados [11]. A cada edição do SBBD são gerados diversos dados sobre a comunidade brasileira de Banco de Dados, os quais podem ser de grande utilidade para a obtenção de informações relacionadas ao perfil dos participantes, às áreas de pesquisa, às instituições participantes, dentre outras. Entretanto, na maioria das vezes, estes dados são disponibilizados em documentos textuais ou até mesmo apenas de forma impressa, não permitindo o seu processamento direto por algum aplicativo. Dessa forma, este trabalho tem como objetivo a criação de um conjunto de dados abertos, denominado *Open*SBBD, com os dados mais relevantes sobre todas as 27 edições já realizadas do SBBD. Os dados do *Open*SBBD são estruturados de acordo com o modelo RDF seguindo os princípios de *Linked Data*.

Como principais contribuições deste trabalho, destacam-se: i) A criação de um conjunto de dados seguindo os princípios de *Linked Data* com alto potencial para ser reutilizado por futuros trabalhos, em conjunto com a criação de uma ontologia, denominada SBC*Event*, com novos termos sobre o domínio de conferências e com reuso de vocabulários existentes; ii) Disponibilização de um SPARQL *Endpoint* para a realização de consultas SPARQL sobre o conjunto de dados criado e iii) Criação de visualizações dos dados do SBBD em formatos amigáveis, como gráficos e tabelas. O restante do artigo está estruturado da seguinte forma. A Seção 2 aborda a contextualização deste trabalho. A Seção 3 descreve a criação da ontologia SBC*Event*. A Seção 4 aborda a criação do conjunto de dados *Open*SBBD. A Seção 5 descreve a criação de um SPARQL *Endpoint* e o desenvolvimento de uma aplicação para a visualização dos dados do *Open*SBBD. Por fim, a Seção 6 expõe a conclusão deste trabalho, além de sugestões para projetos futuros.

2. CONTEXTUALIZAÇÃO

Nesta seção, apresentamos alguns conceitos importantes para o entendimento do restante do trabalho.

2.1 Dados Abertos

O termo Dados Abertos refere-se a dados que podem ser livremente usados, reusados e distribuídos por qualquer pessoa – sujeito apenas, e no máximo, ao requisito de atribuição e compartilhamento pela mesma licença [7]. Os princípios básicos da abertura de dados são [8]: i) *Disponibilidade e acesso*: os dados devem estar disponíveis como um todo, preferencialmente via download na internet; ii) *Reuso e redistribuição:* os dados devem ser fornecidos sob termos que permitam reuso e redistribuição, incluindo a interligação com outros conjuntos de dados e suporte a processamento por máquinas e iii) *Participação universal:* qualquer pessoa deve ser capaz de usar, reusar e redistribuir os dados.

Os principais formatos utilizados para publicar dados abertos são JSON, XML, CSV e RDF. Este último é o formato recomendado pelo W3C[1], e tem atraído grande interesse daqueles que publicam dados de forma

[1] http://www.w3.org/

aberta. Um exemplo disso é o grande número de conjuntos de dados publicados em *Linked Data* atualmente disponíveis no portal de dados governamentais abertos do Reino Unido[2]. A principal motivação é a facilidade de combinar dados neste formato com múltiplas fontes de dados, interligando-se a outras iniciativas *Open Data* na Web [6].

2.2 Considerações de projeto de conjuntos de dados *Linked Data*

Na publicação de dados em *Linked Data*, existem aspectos que devem ser levados em conta para que o projeto possa tornar-se eficiente e alcançar seus objetivos. Dentre esses aspectos, destacaremos dois: a escolha do formato das URIs (*Uniform Resource Identifier*) e o reuso de vocabulários. A escolha do formato das URIs, identificadores únicos de recursos na Web, é um ponto que merece destaque em um projeto de um conjunto de dados em *Linked Data*. As principais recomendações nesta escolha são: i) Utilizar URIs HTTP, pois o esquema "http://" é o único esquema de URIs amplamente suportado pelas infraestrutura dos dias atuais; ii) Definir URIs em um *namespace* HTTP sob controle, onde se pode implementar o que for necessário para torná-las dereferenciáveis, ou seja, prover conteúdo quando as URIs forem acessadas e iii) Tentar manter as URIs estáveis e persistentes, considerando que trocar as URIs em um momento posterior irá quebrar todos os *links* existentes.

Outro elemento importante no que diz respeito à criação de um conjunto de dados em *Linked Data* são os vocabulários. Vocabulários são descrições de conceitos de um domínio (geral ou específico), materializadas em um conjunto de termos. Os termos de um vocabulário, juntamente com os seus relacionamentos, podem ser descritos por meio de ontologias e seus componentes (classes e propriedades), os quais, assim como os outros recursos, possuem uma URI, o que possibilita o seu reuso. De uma maneira geral, os vocabulários são usados para a descrição dos metadados de um conjunto de dados. Com o objetivo de tornar a tarefa de processamento de dados o mais simples possível, o reuso de termos de vocabulários conhecidos e estabelecidos como um padrão é fortemente recomendado. É importante notar que novos termos devem ser criados apenas no caso dos vocabulários já existentes não fornecerem os termos desejados [3]. É uma prática comum incluir termos de diferentes vocabulários na criação de uma nova ontologia para descrever todos os conceitos relacionados a um conjunto específico de dados.

3. ONTOLOGIA SBC*EVENT*

O primeiro passo para a criação do conjunto de dados *Open*SBBD foi a criação da ontologia que representasse os termos usados para a descrição dos dados do SBBD. Para a criação desta ontologia, inicialmente, foram avaliados quais conceitos referentes às conferências promovidas pela SBC (Sociedade Brasileira de Computação) seriam importantes para uma modelagem coerente do domínio. Desta forma, a ontologia poderia ser utilizada para representar não só o SBBD, mas também outros simpósios promovidos pela SBC. Durante a criação da ontologia, duas etapas principais foram realizadas, como descrito a seguir.

- *Análise e reuso de vocabulários:* após a definição dos principais conceitos da ontologia, buscou-se analisar vocabulários conhecidos e estáveis com o objetivo de reaproveitar o maior número de termos possível. Consultando tanto vocabulários de caráter geral como vocabulários com um foco específico no mundo acadêmico (eventos, publicações, etc.), foi possível reutilizar 13 classes e 17

[2] http://www.data.gov.uk

propriedades. Os vocabulários reusados foram os seguintes: FOAF[3], DUBLIN CORE[4], EVENT[5], GEONAMES[6], BIBO[7], AKT[8], SWC[9], SWRC[10].

- *Criação de novos termos:* Apesar dos vocabulários existentes cobrirem uma ampla parte dos termos necessários para a criação da ontologia SBC*Event*, ainda foi necessário definir alguns novos termos, já que em alguns casos não foi encontrado nenhum termo existente refletindo a semântica do conceito que se desejava representar. Desta forma, um conjunto de termos foi criado com o objetivo de complementar o esquema da ontologia.

Utilizando o *Protégé*[11], a ontologia foi criada em OWL. Todos os novos termos foram ligados a outros vocabulários, utilizando *tags* de RDF *Schema*. Uma classe criada que merece destaque é a *sbc:Participation*. Esta classe foi criada para representar um relacionamento entre três conceitos: *Pessoa*, *Organização* e *Conferência* (equivalente a um relacionamento ternário em modelagens Entidade Relacionamento). Alguns vocabulários possuem propriedades que ligam uma organização a uma pessoa, e uma pessoa a uma conferência. No entanto, a afiliação de participantes em simpósios não é definitiva, ou seja, ao longo de diferentes edições do SBBD, autores, por exemplo, podem estar afiliados a diferentes universidades. Desta forma, a modelagem da ontologia deveria oferecer suporte à representação deste tipo de situação de forma consistente. Seguindo uma recomendação do W3C [5], uma classe (*sbc:Participation*) foi criada para representar essa relação ternária. Com isso, uma instância desta classe representa uma instância da relação entre indivíduos das três classes relacionadas.

4. CRIAÇÃO DO CONJUNTO DE DADOS *OPEN*SBBD

A criação do conjunto de dados *Open*SBBD baseou-se em duas linhas de ação: a conversão de dados em bases relacionais para o modelo RDF e a extração manual de dados dos *websites* e anais das edições passadas do SBBD. Para o mapeamento entre o modelo relacional e o modelo RDF, foi utilizada a plataforma D2RQ[12], que fornece um ambiente integrado com múltiplas opções para acessar dados relacionais como grafos RDF virtuais de leitura. Entre os métodos de acesso possíveis, destacam-se os Dumps RDF(*RDF/XML*), as APIs RDF (para aplicações Java) e um SPARQL *Endpoint* [1]. A base relacional utilizada é proveniente de um trabalho sobre uma rede de coautoria baseado em artigos publicados no SBBD, publicado no próprio SBBD em 2010[9]. Para adequá-la a este trabalho, foram realizadas algumas modificações. Um exemplo foi a padronização de valores literais referentes ao sexo de uma pessoa. No banco original os valores eram "M" e "F", mas depois da observação de outros conjuntos de dados, concluiu-se que o padrão mais largamente utilizado é "*male"* e "*female"*.

[3] http://xmlns.com/foaf/spec/
[4] http://purl.org/dc/terms/
[5] http://purl.org/NET/c4dm/event.owl#
[6] http://www.geonames.org/ontology/ontology_v3.1.rdf
[7] http://purl.org/ontology/bibo/
[8] http://www.aktors.org/publications/ontology/portal#
[9] http://data.semanticweb.org/ns/swc/ontology#
[10] http://ontoware.org/swrc/swrc/SWRCOWL/swrc_updated_v0.7.1.owl#
[11] http://protege.stanford.edu/
[12] http://d2rq.org/

Um dos componentes da plataforma D2RQ é o *generate-mapping*. A partir da análise do esquema do banco de dados, esta ferramenta cria um arquivo de mapeamentos no formato *Turtle*[13], no qual é possível configurar os mapeamentos de tabelas e colunas do banco para classes e propriedades de ontologias. Assim, os mapeamentos foram definidos utilizando tanto os termos criados como os termos reusados, previamente definidos, além do mapeamento para dados de outros conjuntos de dados, como o GeoNames, interligando os respectivos conjuntos de dados. A partir deste arquivo, um dump RDF do banco de dados relacional foi gerado, utilizando a serialização *RDF/XML*, através de outro componente da plataforma D2RQ, a ferramenta *dump-rdf*. Após executar o *dump-rdf*, um *dump* RDF foi criado de acordo com o esquema definido na Seção 3. Além destes dados, outros dados foram adicionados ao conjunto de dados manualmente. A partir de uma análise dos *websites* e anais dos eventos passados, foram criados *scripts* SQL para a inserção desses dados no banco que, por sua vez, foram convertidos para RDF através do D2RQ.

5. BUSCA E VISUALIZAÇÃO DOS DADOS DO *OPEN*SBBD

Para facilitar a busca dos dados disponíveis no *Open*SBBD, foi utilizada uma ferramenta chamada *D2R-Server*, da plataforma D2RQ, que permite a publicação do conteúdo de bases relacionais na Web Semântica. Além disso, disponibiliza um SPARQL *Endpoint* para a submissão de consultas SPARQL. A partir desta interface, foram realizadas consultas explorando as classes e propriedades definidas na ontologia. Como exemplo, temos a consulta *Q1*: *Quais instituições de ensino já publicaram no SBBD e quantos autores publicaram por cada uma delas (ordem descendente)?* A consulta SPARQL correspondente é mostrada na Figura 1 e os dez primeiros resultados são apresentados na Figura 2.

Para a visualização de dados foi desenvolvida uma aplicação com o objetivo de fornecer informações relevantes sobre o SBBD, por meio de gráficos e tabelas. As principais tecnologias utilizadas foram a linguagem de programação *Python*[14], e o framework para desenvolvimento Web *Django*[15], e a API *SPARQL Endpoint interface to Python*[16], utilizada para submeter consultas SPARQL ao *Endpoint* do *D2R-Server*. As figuras 3 e 4 ilustram exemplos de gráficos gerados na aplicação a partir de consultas realizadas. O esquema final da ontologia, o conjunto de dados e o SPARQL *Endpoint* podem ser acessados através do link http://www.cin.ufpe.br/~mgrb/opensbbd/.

```
SELECT DISTINCT ?name (COUNT(?person) as ?authors) WHERE {
  ?institution a akt:Learning-Centred-Organization.
  ?institution foaf:name ?name.
  ?person a foaf:Person.
  ?article dcterms:creator ?person.
  ?participation sbbd:participationPerson ?person.
  ?participation sbbd:participationOrganization ?institution.
  ?participation sbbd:participationConference ?conference.
  ?article bibo:presentedAt ?conference.
}
GROUP BY ?institution ?name
ORDER BY DESC(?authors)
```

Fig. 1. - Consulta SPARQL Q1

?name	?authors
Universidade Federal do Rio Grande do Sul	100
Universidade Federal do Rio de Janeiro	76
Pontifícia Universidade Católica (RJ)	71
Universidade Federal de Pernambuco	64
Universidade Estadual de Campinas	47
Universidade Federal da Paraíba	45
Universidade Federal de Minas Gerais	43
Universidade de São Paulo	28
Instituto Militar de Engenharia	26
Universidade Federal de São Carlos	13

Fig. 2. - Resultado da Consulta Q1

[13] http://www.w3.org/TR/turtle/
[14] http://www.python.org/
[15] https://www.djangoproject.com/
[16] http://sparql-wrapper.sourceforge.net/

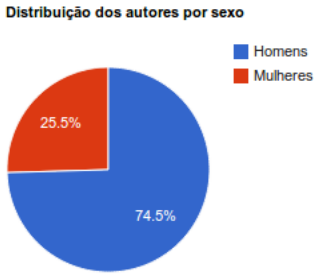


Fig. 3. Gráfico de distribuição de autores por sexo

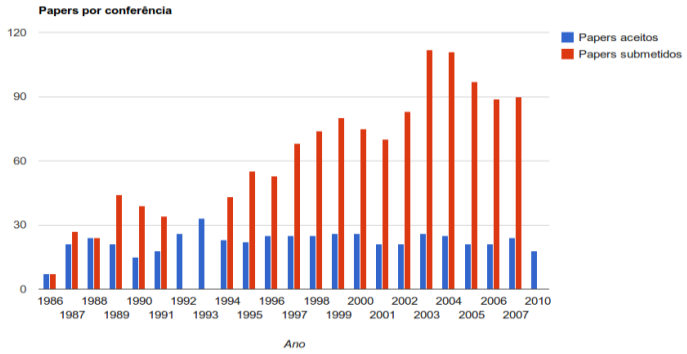


Fig. 4. Gráfico de artigos por conferência ao longo das edições do SBBD

6. CONCLUSÃO

Este trabalho dividiu-se essencialmente em três partes: a criação de um conjunto de dados em RDF, interligado com grandes conjuntos de dados conhecidos; a publicação deste conjunto de dados em um *SPARQL Endpoint*, disponível para a execução de consultas; o desenvolvimento de uma aplicação Web para a visualização de gráficos e tabelas gerados a partir do conjunto de dados criado. Com este conjunto de dados sobre o SBBD, as informações das edições passadas deste grande evento passam a fazer parte da Web de dados. Além disso, aplicações relacionadas ao SBBD agora têm a possibilidade de utilizar este conjunto como uma de suas fontes de dados. Este material também pode incentivar futuros trabalhos na área de Web Semântica e *Linked Data* sobre eventos da SBC, ou de forma mais genérica, sobre eventos acadêmicos em geral. Entre linhas de ações futuras, podemos destacar as seguintes: i) Ampliar o conjunto de dados com mais dados que não foram coletados durante a realização deste trabalho; ii) Mudança de ferramenta para disponibilização do *SPARQL Endpoint*, uma vez que um dos pré-requisitos para utilizar o *SPARQL Endpoint* oferecido pelo *D2R-Server* é a existência de uma base relacional, para que os mapeamentos sejam feitos em tempo real; iii) Enriquecimento da aplicação de visualização de dados.

REFERÊNCIAS

[1] BIZER, C.; CYGANIAK, R. "D2RQ-lessons learned." *W3C Workshop on RDF Access to Relational Databases*. 2007.

[2] BIZER, C.; CYGANIAK, R.; HEATH, T. "How to publish linked data on the web." (2008).

[3] BIZER, C.; HEATH, T.; BERNERS-LEE, T. "Linked data-the story so far." *International Journal on Semantic Web and Information Systems (IJSWIS)* 5.3 (2009): 1-22.

[4] BIZER, C.; HEATH, T.; IDEHEN, K.; BERNERS-LEE, T. Abril, 2008. Linked data on the web (LDOW2008). In *Proceeding of the 17th international conference on World Wide Web* (pp. 1265-1266). ACM.

[5] NOY, N.; et al. "Defining n-ary relations on the semantic web." *W3C Working Group Note* 12 (2006): 4.

[6] OPEN DATA HANDBOOK. File Formats, 2013. Disponível em <http://opendatahandbook.org/en/appendices/file-formats>. Acesso em: 10 jul. 2013.

[7] OPEN DEFINITION, 2013. Disponível em: <http://opendefinition.org/>. Acesso em: 4 jul. 2013.

[8] OPEN KNOWLEDGE FOUNDATION, 2013. Disponível em: <http://okfn.org/opendata/>. Acesso em: 4 jul. 2013.

[9] PROCÓPIO, P. S.; LAENDER, A. H. F.; MORO, M. M. "Análise da rede de coautoria do simpósio brasileiro de bancos de dados." SIMPÓSIO BRASILEIRO DE BANCO DE DADOS, Florianópolis, 2011. Proceedings... Florianópolis (2011).

[10] SEMANTIC WEB WIKI, 2013. Disponível em: <http://semanticweb.org/wiki/Main_Page>. Acesso em: 9 jan. 2013.

[11] SIMPÓSIO BRASILEIRO DE BANCO DE DADOS. SBBD, 2012. Disponível em: <http://sws2012.ime.usp.br/sbbd/>. Acesso em: 25 jan. 2013.

FPSMining: A Fast Algorithm for Mining User Preferences in Data Streams

Jaqueline A. J. Papini, Sandra de Amo, Allan Kardec S. Soares

Federal University of Uberlândia, Brazil
jaque@comp.ufu.br, deamo@ufu.br, allankardec@gmail.com

Abstract. The traditional preference mining setting, referred to here as the *batch setting*, has been widely studied in the literature in recent years. However, the dynamic nature of the problem of mining preferences increasingly requires solutions that quickly adapt to change. The main reason for this is that frequently user's preferences are not static and can evolve over time. In this article, we address the problem of mining *contextual* preferences in a data stream setting. *Contextual Preferences* have been recently treated in the literature and some methods for mining this special kind of preferences have been proposed in the batch setting. As main contribution of this article, we formalize the contextual preference mining problem in the stream setting and propose an algorithm for solving this problem. We implemented this algorithm and showed its efficiency through a set of experiments over real data.

Categories and Subject Descriptors: H.2.8 [**Database Management**]: Database Applications—*data mining*

Keywords: context-awareness, data mining, data streams, preference mining

1. INTRODUCTION

A data stream may be seen as a sequence of relational tuples that arrive continuously in variable time. Some typical fields of application for data streams are: the financial market, credit card transaction flow, web applications and sensor data. Traditional approaches for data mining cannot successfully process the data streams mainly due to the potentially infinite volume of data and its evolution over time. Consequently, several data stream mining techniques have emerged to deal properly with this new data format [Domingos and Hulten 2000; Bifet and Kirkby 2009; Gama 2010].

Most of the research on preference mining has focused on scenarios in which the mining algorithm has a set of static information on user preferences at its disposal [Jiang et al. 2008; de Amo et al. 2012]. However, in most situations, user preferences are dynamic. For instance, consider an *online news site* that wants to discover the preferences of its users regarding news and make recommendations based on that. Note that due to the dynamic nature of news, it is plausible that user's preferences would evolve rapidly with time. In times of elections, a user can be more interested in *politics* than in *sports*. In times of Olympic Games, the other way round.

The most crucial issues which makes the process of mining in a stream setting much more challenging than in a batch setting are the followings: (1) data are not stored and are not available whenever needed; each tuple must be accepted as it arrives and once inspected or ignored it is discarded with no possibility to be inspected again; (2) the mining process has to cope with both limited workspace and limited amount of time; (3) the mining algorithm should be able to produce the best model at any moment it is asked to and using only the training data it has observed so far. For more details on these challenges see [Rajaraman and Ullman 2011].

This work focus on a particular kind of preferences, the *contextual preferences* ([de Amo et al.

2012]). The preference model is a set of *contextual preference rules*.

Proposals for suitable algorithms to solve the problem of mining user preferences in streams have been little explored in the literature. [Jembere et al. 2007] presents an approach to mine user preferences in an environment with multiple context-aware services, but uses incremental learning only for the context, and not for the user's preferences. [Somefun and La Poutré 2007] presents an online method that aims to use aggregated knowledge on the preference of many customers to make recommendations to individual clients. Finally, [Shivaswamy and Joachims 2011] proposes an online learning model that learns through preference feedback, and is especially suitable for web search and recommendation systems. None of them specifically addresses the problem we address.

The main goal of this article is to propose an algorithm for mining contextual preferences from a *preference stream* sample. We also present the results of a set of experiments for this algorithm executed on real data.

2. BACKGROUND ON CONTEXTUAL PREFERENCE MINING IN THE BATCH SETTING

In this section we briefly introduce the problem of mining contextual preferences in a batch setting. Please see [de Amo et al. 2012] for more details on this problem.

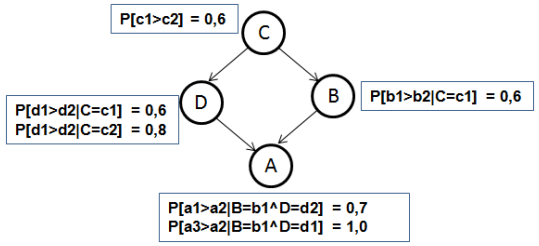
A *preference relation* on a finite set of objects $A = \{a_1, a_2, ..., a_n\}$ is a strict partial order over A, that is a binary relation $R \subseteq A \times A$ satisfying the irreflexivity and transitivity properties. We denote by $a_1 > a_2$ the fact that a_1 is preferred to a_2. A *Preference Database* over a relation R is a finite set $\mathcal{P} \subseteq \text{Tup}(R) \times \text{Tup}(R)$ which is *consistent*, that is, if $(u, v) \in \mathcal{P}$ then $(v, u) \notin \mathcal{P}$. The pair (u, v), usually called bituple, represents the fact that the user prefers *the tuple u to the tuple v*. Fig. 1 (b) illustrates a preference database over R, representing a sample provided by the user about his/her preferences over tuples of I (Fig. 1 (a)).

Id	A	B	C	D
t_1	a_1	b_1	c_1	d_1
t_2	a_1	b_1	c_1	d_2
t_3	a_2	b_1	c_1	d_2
t_4	a_1	b_2	c_1	d_2
t_5	a_2	b_1	c_2	d_1
t_6	a_3	b_1	c_1	d_1

(a)

(t_1,t_2)
(t_1,t_3)
(t_4,t_5)
(t_4,t_2)
(t_5,t_6)
(t_3,t_5)
(t_4,t_1)

(b)



(c)

Fig. 1. (a) An instance I, (b) A Preference Database $\mathcal{P}$, (c) Preference Network $\mathbf{PNet}_1$

The problem of mining contextual preferences in the batch setting consists in extracting a *preference model* from a preference database provided by the user. The preference model is specified by a *Bayesian Preference Network* (BPN), specified by (1) a directed acyclic graph G whose nodes are attributes and the edges stands for attribute dependency and (2) a mapping θ that associates to each node of G a finite set of conditional probabilities. Fig. 1(c) illustrates a BPN $\mathbf{PNet}_1$ over the relational schema $R(A, B, C, D)$. Notice that the preference on values for attribute B depends on the context C: if $C = c1$, the probability that value b_1 is preferred to value b_2 for the attribute B is 60%. A BPN allows to infer a *preference ordering* on tuples. The following example illustrates how this ordering is obtained.

Before defining the precision and recall of a preference network, we need to define the *strict partial order* inferred by the preference network.

Example 2.1 *Preference Order.* Let us consider the BPN $\mathbf{PNet}_1$ depicted in Fig. 1(c). In order to compare the tuples $u_1 = (a_1, b_1, c_1, d_1)$ and $u_2 = (a_2, b_2, c_1, d_2)$, we proceed as follows: (1) Let $\Delta(u_1, u_2)$ be the set of attributes where the u_1 and u_2 differ. In this example, $\Delta(u_1, u_2) = \{A, B, D\}$; (2) Let $\min(\Delta(u_1, u_2)) \subseteq \Delta(u_1, u_2)$ such that the attributes in $\min(\Delta)$ have no ancestors in Δ (according to graph G underlying the BPN $\mathbf{PNet}_1$). In this example $\min(\Delta(u_1, u_2)) = \{D, B\}$. In order to u_1 be

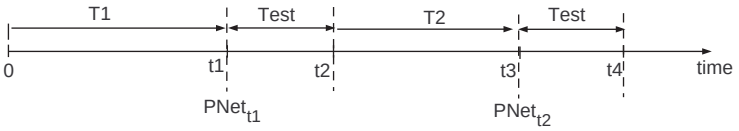


Fig. 2. The dynamics of the mining and testing processes through time

preferred to u_2 it is necessary and sufficient that $u_1[D] > u_2[D]$ and $u_1[B] > u_2[B]$; (3) Compute the following probabilities: p_1 = probability that $u_1 > u_2 = P[d_1 > d_2|C = c_1] * P[b_1 > b_2|C = c_1] = 0.6$ * $0.6 = 0.36$; p_2 = probability that $u_2 > u_1 = P[d_2 > d_1|C = c_1] * P[b_2 > b_1|C = c_1] = 0.4$ * $0.4 = 0.16$. In order to compare u_1 and u_2 we select the higher between p_1 and p_2. In this example, $p_1 > p_2$ and so, we infer that u_1 is preferred to u_2. If $p_1 = p_2$ we conclude that u_1 and u_2 are incomparable.

A BPN is evaluated by considering its *Precision* and *Recall* with respect to a *test* preference database $\mathcal{P}$. The *recall* is defined by Recall(**PNet**,$\mathcal{P}$)= $\frac{N}{M}$, where M is number of bituples in $\mathcal{P}$ and N is the amount bituples $(t_1, t_2) \in \mathcal{P}$ compatible with the preference ordering inferred by **PNet** on the tuples t_1 and t_2. The *precision* is defined by Precision(**PNet**,$\mathcal{P}$)= $\frac{N}{K}$, where K is the number of elements of $\mathcal{P}$ which are comparable by **PNet**.

3. PROBLEM FORMALIZATION IN THE STREAM SETTING

The main differences between the batch and the stream settings concerning the preference mining problem we address in this article may be summarized as follows:

—*Input data*: to each sample bituple (u, v) collected from the stream of clicks from a user on a site, is associated a timestamp t standing for the time the user made this implicit choice. Let T be the infinite set of all timestamps. So, the input data from which a preference model will be extracted is a *preference stream* defined as a (possibly) infinite set $P \subseteq Tup(R) \times Tup(R) \times T$ which is *temporally consistent*, that is, if $(u, v, t) \in P$ then $(v, u, t) \notin P$. The triple (u, v, t) that we will call *temporal bituple*, represents the fact that the user prefers tuple u over tuple v at the time instant t.

—*Output*: the preference model to be extracted from the preference stream is a *temporal BPN*, that is, a sequence of BPNs $< PNet_t : t \in \mathbb{N} >$. At each instant t the algorithm is ready to produce a preference model $PNet_t$ which will be used to predict the user preferences at this moment.

—*The preference order induced by a BPN at each instant t*: At each instant t we are able to compare tuples u and v by employing the Preference Model $PNet_t$ updated with the elements of preference stream until the instant t. The preference order between u and v is denoted by $>_t$ and is obtained as illustrated in example 2.1.

—*Precision and Recall at instant t*: The quality of the preference model $PNet_t$ produced by the algorithm at instant t is measured by considering a finite set $Test$ of preference samples arriving at the system after instant t, that is, by considering a finite set $Test$ whose elements are of form (u, v, t') with $t' \geq t$. Let $\mathcal{P}$ be the (non temporal) preference database obtained from $Test$ by removing the timestamp t' from its elements. The precision and recall of the preference model $PNet_t$ obtained at instant t is evaluated according to the equations given in the previous section applied to the (static) BPN $PNet_t$ and the non temporal preference database $\mathcal{P}$. The precision and recall of the algorithm are measured repeatedly over time. Fig. 2 illustrates this dynamic process of mining and testing the preference models from the preference stream. In fact, as we will see in the following section, the mining algorithm is able to produce the preference model by using only a reduced set of *statistics* extracted from the training sets at the instant t it is requested to act. The entire sets T_i of the preference samples stream are not needed in the mining process.

Now we are ready to state the problem of Mining Contextual Preferences from a Preference Stream:

Input: a relational schema $R(A_1, A_2, ..., A_n)$, and a preference stream S over R.

Output: whenever requested, return a BPN_t over R having good precision and recall, where t is the time instant of request.

		c_1>c_2	c_2>c_1	c_4>c_5
A	a_1	3	1	-
	a_2	-	-	2
B	b_3	1	-	1
	b_5	1	-	1

c_1>c_2	4
c_2>c_1	2
c_4>c_5	3

(a)

		c_1>c_2	c_2>c_1	c_4>c_5
A	a_1	3	2	-
	a_2	-	-	2
B	b_3	1	-	1
	b_5	1	-	1
	b_6	-	1	-

c_1>c_2	4
c_2>c_1	3
c_4>c_5	3

(b)

	u_1			u_2		
T	A	B	C	A	B	C
t_1	a_1	b_3	c_1	a_1	b_3	c_2
t_2	a_1	b_3	c_2	a_1	b_5	c_1
t_3	a_2	b_5	c_2	a_1	b_3	c_1
t_4	a_2	b_3	c_4	a_2	b_6	c_5
t_5	a_1	b_5	c_1	a_1	b_5	c_2
t_6	a_2	b_3	c_4	a_2	b_3	c_5
t_7	a_1	b_3	c_1	a_1	b_5	c_2
t_8	a_2	b_5	c_1	a_1	b_6	c_2
t_9	a_1	b_5	c_4	a_2	b_5	c_5
t_{10}	a_1	b_6	c_2	a_1	b_6	c_1

(c)

Fig. 3. (a) Sufficient statistics for attribute C at the time instant t_9. (b) Sufficient statistics for attribute C at the time instant t_{10}. (c) Preference stream S until time instant t_{10}.

4. THE FPSMINING ALGORITHM

In this article we propose an algorithm for mining user contextual preferences in the stream setting: the *FPSMining algorithm*.

In order to save processing time and memory, in this algorithm we do not store the elements of the preference stream processed so far, just collect *sufficient statistics* from it in an online way. Example 4.1 illustrates the sufficient statistics collected from a preference stream S.

Example 4.1 *Sufficient Statistics.* Let $R(A, B, C)$ be a relational schema with a_1, $a_2 \in \text{dom}(A)$, b_3, b_5, $b_6 \in \text{dom}(B)$ and c_1, c_2, c_4, $c_5 \in \text{dom}(C)$. Let S be a preference stream over R as shown in Fig. 3(c), where T column shows the timestamp that the temporal bituple was generated, and $u_1 >_{t_i} u_2$ (u_1 is preferred to u_2 at t_i) for every temporal bituple (u_1, u_2, t_i) in the preference stream, with $1 \leq i \leq 10$. Consider the sufficient statistics for attribute C shown in Fig. 3(a) collected from the preference stream S until time instant t_9. The table on top of Fig. 3(a) shows the *context counters* regarding the preferences over the attribute C, and the table on the bottom shows the *general counters* over C. Context counters account for the possible causes for a particular preference over an attribute, and general counters stores the number of times that a particular preference over an attribute appeared in the stream. With the arrival of a temporal bituple at the time instant t_{10} - call it l, the sufficient statistics are updating as follows (see Fig. 3(b)): (1) Compute $\Delta(u_1, u_2)$, which is the set of attributes where u_1 and u_2 differ in l. In this example, $\Delta(u_1, u_2) = \{C\}$, then only the attribute C will have their statistics updated according to l; (2) Increment context counters a_1 and b_6 regarding the preference c_2>c_1 (table top of the Fig. 3(b)). Note that in the temporal bituple l values a_1 and b_6 are possible contexts (causes) for the preference c_2>c_1, just because they are equal in both tuples. As until now we had no context b_6, it is inserted in the statistics; (3) Increment general counter of preference c_2>c_1 (table down of the Fig. 3(b)).

The main idea of the FPSMining algorithm is to create a preference relation from the most promising dependencies between attributes of a preference stream. In order to measure the dependence between a pair of attributes, we use the concept of *degree of dependence*. The degree of dependence of a pair of attributes (X, Y) with respect to a snapshot Q of the sufficient statistics from the preference stream S at the time instant t (computed by Alg. 1) is a real number that estimates how preferences on values for the attribute Y are influenced by values for the attribute X. In Alg. 1, for each pair (y, y') appearing in the *general counters* over Y of the sufficient statistics Q (line 1), $T^{time}_{yy'}$ denotes the set of temporal bituples $(u_1, u_2, t) \in S$, such that $t \leq time$, $(u_1[Y] = y \wedge u_2[Y] = y')$ or $(u_1[Y] = y' \wedge u_2[Y] = y)$; $\text{support}((y, y'),Q) = \frac{|T^{time}_{yy'}|}{n}$, where n is the number of elements of the preference stream processed until the time instant $time$ and $|T^{time}_{yy'}|$ is the sum of the values of two general counters of Y from Q. We say that a pair (y, y') in the general counters over Y is *comparable* (line 1) if $\text{support}((y, y'),Q) \geq \alpha_1$, for a given threshold α_1, $0 \leq \alpha_1 \leq 1$. For each $x \in \mathbf{dom}(X)$ (line 2), we denote by $S^{time}_{x|(y,y')}$ the subset of $T^{time}_{yy'}$ containing the temporal bituples (u_1, u_2, t) such that $u_1[X] = u_2[X] = x$; $\text{support}(S^{time}_{x|(y,y')},Q) = \frac{|S^{time}_{x|(y,y')}|}{|\bigcup_{x' \in \mathbf{dom}(X)} S^{time}_{x'|(y,y')}|}$, where $|S^{time}_{x|(y,y')}|$ is obtained by the sum of the values of two context counters

of Y from Q: fixing the line for $X = x$, and sum the column values $y > y'$ and $y' > y$. We say that x is a *cause for* (y, y') *being comparable* (line 2) if support($S^{time}_{x|(y,y')}$,Q) $\geq \alpha_2$, for a threshold α_2, $0 \leq \alpha_2 \leq 1$.

Algorithm 1: The degree of dependence of a pair of attributes

Input: Q: a snapshot of the sufficient statistics from the preference stream S at the time instant *time*; (X, Y): a pair of attributes; two thresholds $\alpha_1 > 0$ and $\alpha_2 > 0$.
Output: the degree of dependence of (X, Y) with respect to Q at the time instant *time*.

1 **for** *each pair* $(y, y') \in$ ***general counters*** *over* Y *from* Q, $y \neq y'$ *and* (y, y') *comparable* **do**
2 **for** *each* $x \in$ ***dom***(X) *where* x *is a cause for* (y, y') *being comparable* **do**
3 Let $f_1(S^{time}_{x|(y,y')}) = \max\{N, 1 - N\}$, where
$$N = \frac{|\{(u_1, u_2, t) \in S^{time}_{x|(y,y')} : u_1 >_t u_2 \wedge (u_1[Y] = y \wedge u_2[Y] = y')\}|}{|S^{time}_{x|(y,y')}|}$$
4 Let $f_2(T^{time}_{yy'}) = \max\{f_1(S^{time}_{x|(y,y')}) : x \in \mathbf{dom}(X)\}$
5 Let $f_3((X, Y), Q) = \max\{f_2(T^{time}_{yy'}) : (y, y') \in$ **general counters** over Y from Q, $y \neq y'$, (y, y') comparable$\}$
6 **return** $f_3((X, Y), Q)$

Given this, our algorithm is straightforward and builds a BPN_t from sufficient statistics extracted from the preference stream using the Alg. 2.

Algorithm 2: The FPSMining Algorithm

Input: $R(A_1, A_2, ..., A_n)$: a relational schema; S: a preference stream over R.
Output: whenever requested, return a BPN_t over R, where t is the time instant of request.

1 Take a snapshot Q of the sufficient statistics from S at the time instant t.
2 **for** *each pair of attributes* (A_i, A_j), *with* $1 \leq i, j \leq n, i \neq j$ **do**
3 Use Alg. 1 for calculate the degree of dependence between the pair (A_i, A_j) according to Q
4 Let Ω be the resulting set of these calculations, with elements of type (A_i, A_j, dd), where dd is the degree of dependency between the pair (A_i, A_j)
5 Eliminate from Ω all elements whose $dd < 0.5$ (indicates a weak dependence between a pair of attributes)
6 Order the elements (A_i, A_j, dd) in Ω in decreasing order according to their dd
7 Start the graph G_t of BPN_t with a node for each attribute of R
8 **for** *each element* $(A_i, A_j, dd) \in$ *ordered set* Ω **do**
9 Insert the edge (A_i, A_j) in the graph G_t only if the insertion does not form cycles in G_t
10 Once the graph G_t of BPN_t was created, estimate the conditional probabilities tables θ_t of BPN_t, using the Maximum Likelihood Principle (see [de Amo et al. 2012] for details) over Q.
11 **return** BPN_t

5. EXPERIMENTAL RESULTS

The data used in the experiments contains preferences related to films collected by GroupLens Research [1] from the MovieLens web site [2]. We simulated a preference stream from these data, as follows: we stipulated a time interval λ (measured in days or hours), and each tuple in the dataset was compared to all others in a radius λ relative to its timestamp, thus generating temporal bituples. The resulting preference stream has five attributes (director, genre, language, star and year), and its elements correspond to preferences about movies determined by a particular user.

In order to evaluate our algorithm, we adapted the sampling technique proposed by [Bifet and Kirkby 2009], based on holdout for data stream, for the preference scenario. The sampling technique used takes input from three parameters: n_{train}, n_{test} and n_{eval}. The n_{train} and n_{test} variables represent, respectively, the number of elements in the stream to be used to train and test the model at each evaluation. The variable n_{eval} represents the number of evaluations desired along the stream.

[1] Available at `http://www.grouplens.org/taxonomy/term/14`
[2] Available at `http://movielens.umn.edu`

In the tests, we vary all parameters of the algorithm. Fig. 4(a), (b), (c), (d) show the results for varying the parameters *user* (each *user* has a different stream), λ (each λ produces a different stream), n_{train} and n_{test}, α_1 and α_2, respectively. In each figure, except for the parameter that was varied, the default values used were: $user$=$U1$, λ=1 day, n_{train}=1,000 and n_{test}=100, α_1=0.2 and α_2=0.1. The n_{eval} values were calculated according to the total size of the stream.

Fig. 4(a) shows that the more movies the user evaluated (tuples), the better the recall and precision of our algorithm with respect to their preferences. Fig. 4(b) shows that until λ=1 day the quality values increase as the number of elements in the preference stream increases. Fig. 4(c),(d) show that our algorithm was stable to the variation of the parameters n_{train}, n_{test}, α_1 and α_2 in these data.

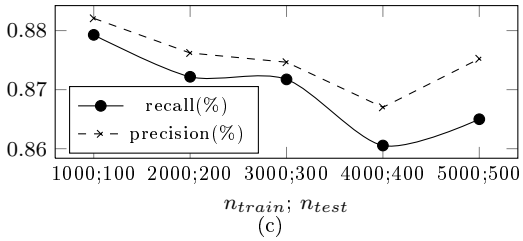
In the tests carried out the recall and precision values were satisfactory, showing that in most cases our algorithm compared correctly temporal bituples submitted to it, thereby proving to be efficient for mining user preferences in a stream setting. The execution time for the generation of the model was 16 ms and the time to complete one cycle of the holdout was 31 ms.

User	Tuples	Bituples	Recall	Precision
$U1$	7359	1580710	0.871	0.874
$U2$	6047	1658450	0.794	0.801
$U3$	4483	563419	0.778	0.784
$U4$	4449	198618	0.769	0.785

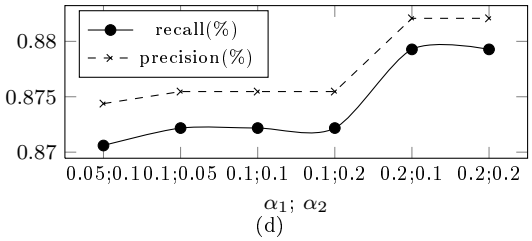
(a)

Stream	λ	Bituples	Recall	Precision
$S1$	1h	227100	0.800	0.814
$S2$	12h	645319	0.846	0.853
$S3$	1d	1048228	0.879	0.882
$S4$	7d	1580710	0.871	0.874
$S5$	15d	1759753	0.875	0.876

(b)



(c)



(d)

Fig. 4. Experimental Test Set

6. CONCLUSION AND FURTHER WORK

In this article we proposed the FPSMining algorithm to solve the problem of mining user contextual preferences in the stream setting. We show that it is fast and produces satisfactory results on real data. As immediate future work, we intent to implement two different learning technique based on FPSMining (an incremental and an ensemble algorithms) as well as a synthetic preference stream generator for tests with a huge amount of data.

REFERENCES

Bifet, A. and Kirkby, R. Data stream mining: a practical approach. Tech. rep., The University of Waikato, 2009.

de Amo, S., Bueno, M. L. P., Alves, G., and Silva, N. F. Cprefminer: An algorithm for mining user contextual preferences based on bayesian networks. In *IEEE 24th International Conference on Tools with Artificial Intelligence*. pp. 114–121, 2012.

Domingos, P. and Hulten, G. Mining high-speed data streams. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2000.

Gama, J. *Knowledge Discovery from Data Streams*. Chapman & Hall/CRC, 2010.

Jembere, E., Adigun, M. O., and Xulu, S. S. Mining context-based user preferences for m-services applications. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence*, 2007.

Jiang, B., Pei, J., Lin, X., Cheung, D. W., and Han, J. Mining preferences from superior and inferior examples. In *KDD*. ACM, pp. 390–398, 2008.

Rajaraman, A. and Ullman, J. D. *Mining of Massive Datasets*. Cambridge University Press, 2011.

Shivaswamy, P. K. and Joachims, T. Online learning with preference feedback. *CoRR* vol. abs/1111.0712, 2011.

Somefun, D. J. A. and La Poutré, J. A. A fast method for learning non-linear preferences online using anonymous negotiation data. In *Proceedings of the AAMAS workshop and TADA/AMEC conference*. pp. 118–131, 2007.

On defining metrics for elasticity of cloud databases

Rodrigo F. Almeida[1], Flávio R. C. Sousa[1], Sérgio Lifschitz[2] and Javam C. Machado[1]

[1] Universidade Federal do Ceará - Brasil
rodrigo.felix@lsbd.ufc.br, {sousa,javam}@ufc.br
[2] Departamento de Informática, PUC-Rio - Brasil
sergio@inf.puc-rio.br

Abstract. In the database area, elasticity of cloud computing has required data systems to increase and decrease their resources on demand. However, traditional benchmark tools for data systems are not sufficient to analyze some specificities of these systems in a cloud. New metrics for elasticity are needed to provide an indicator both from consumer and provider perspective. In this work we present a set of metrics for elasticity of cloud data systems. In addition, we evaluate our model performing experiments and analyzing their results.

Categories and Subject Descriptors: H.2 [**Database Management**]: Miscellaneous; H.3 [**Information Storage and Retrieval**]: Miscellaneous

Keywords: Benchmarking, Elasticity, Databases, Cloud

1. INTRODUCTION

Scalability, elasticity, and pay-per-use pricing model are the major reasons for the successful and widespread adoption of cloud infrastructures. Since the majority of cloud applications are data-driven, database management systems (DBMSs) powering these applications are critical components in the cloud software stack [Elmore et al. 2011]. Elasticity is the degree to which a system is able to adapt to workload changes by provisioning and deprovisioning resources in an autonomic manner, such that at each point in time the available resources match the current demand as closely as possible. [Herbst et al. 2013] Stateful systems, such as DBMSs, are hard to scale elastically because of the requirement of maintaining consistency of the database that they manage [Minhas et al. 2012]. Some DBMSs implement elasticity [HBase 2013] [Cassandra 2013] [MongoDB 2013]. However, consumers and providers want to measure the elasticity of a given database system. On one hand, consumers want to compare some cloud data systems to choose one that fits better to their needs. On the other hand, providers want to meet the Service-Level Agreement (SLA) of database systems with the minimum resources and cost. There are some studies on how to measure the elasticity [Cooper et al. 2010] [Islam et al. 2012] [Dory et al. 2011]. However, these studies do not address important issues like considering DBMS-specific aspects, analyzing over-provisioning scenario and measuring elasticity when the number of clients varies during workloads. Changing the number of clients during workload illustrates a more realistic scenario for many applications whose number of users goes up and down. Particularly, reducing the number of clients is a more challenging aspect, that is not addressed for most related works.

The major contributions of this article are (i) definition of a model with metrics to measure the elasticity of cloud database systems, (ii) presentation of two different perspectives (consumer and provider) for elasticity, and, finally, (iii) evaluation of our model through some experiments. In order to perform these experiments, we developed BenchXtend [Almeida 2013b].

2. OUR PROPOSED APPROACH

2.1 BenchXtend

We propose a tool called BenchXtend [Almeida 2013b] which is based on YCSB [Cooper et al. 2010] to provide a way to change the number of clients while running a workload, as well as to calculate the metrics proposed in this work. Varying the number of clients accessing a cloud data system is essential to properly evaluate the elasticity of that system, since the load will vary and then the system will have to act (adding or removing resources) to keep up the response time in an acceptable range that satisfies the SLA. In our tool, the number of clients is changed automatically according to a timeline defined in an input file. In our context, a timeline is simply a list of pairs $< time, number of clients >$ explicitly defined in a file, that describes the expected number of clients in such a moment.

Our tool sends queries to, what we call, a Cloud Database System that is composed of (i) an Instance Manager (composed of a Monitor and a Decision Taker), (ii) a Database Manager and (iii) a pool of instances (virtual machines) that are running or available to be started. The Instance Manager is responsible for monitoring the pool gathering statistics, as well as for taking decision on starting or on stopping instances of the pool. The Database Manager is the access point to where queries are sent. Depending on the DBMS, there is no central node to receive queries and distributed. Thus, queries are sent all over the pool and the Database Manager is a regular node. The pool of instances is only a set of pre-configured instances that can be started or stopped by the Instance Manager. Since our focus is not on how well (or badly) designed the cloud database system itself is, but on the benchmark tool, for a matter of simplicity, we implemented separately our own Instance Manager [Almeida 2013a] in Ruby calling Amazon EC2 API. Other authors [Konstantinou et al. 2012] [Cruz et al. 2013] propose more robust solutions to manage instances, even though they are not developed for Amazon EC2.

2.2 Elasticity metrics

Our approach to define metrics for elasticity extends the work proposed on a previous work [Almeida 2012]. We use a penalty model approach to measure imperfections in elasticity for database systems. Similarly to [Islam et al. 2012], our elasticity model is composed of two parts: penalty for over- and under-provisioning. Unlike [Islam et al. 2012], we explore database system features, like response times, and present both the consumer and provider perspectives. We consider a scenario where a consumer accesses a database service in a Platform-as-a-Service (PaaS) provider. In a PaaS scenario, database systems are usually exposed as services and the applications deployed by the consumers can execute queries on them, abstracting the complexity of replication, consistency, and so on.

2.3 Consumer perspective

Due to the large number of PaaS providers and to the so-claimed buzzword elasticity, consumers need to have a model to evaluate and compare elasticity of database systems. From a PaaS-consumer perspective, a database system is elastic if, regardless the number of queries submitted to the system, it satisfies the SLA. We assume that upper and lower bounds for response times are defined by *operation type* in the SLA. Queries whose response times are between these bounds are not considered either under- nor over-provisioned. YCSB provides five operation types (read, insert, update, delete, scan) but, due to paper space restrictions, we analyze in our experiments only scan and insert.

Under-provisioning penalty (*underprov*): The strategy for *underprov* is compliant to the concept of penalties for database services in the cloud. In this case, the resources allocated to the consumer are under-provisioned, i.e. insufficient to keep up the quality, generating a response time increase and consequently causing a penalty. This penalty is for violating the upper bound of SLA. Penalties due to under-provisioning are directly related to bad elasticity, since, if the system had a good one, it would have scaled up to address the increase of demand to avoid penalties. We define this metric

(shown in Equation 1) as the average of the ratio execution time by expected time of those n queries whose response times are greater than the upper bound defined in the SLA and that are not outliers. In order to remove discrepant values, we use interquartile range analysis to identify extreme outliers on the response times data distribution and then we remove them from the calculation of all metrics of our model. Outliers are those response times that are out of the range $[Q_1 - 3 * R, Q_3 + 3 * R]$, where Q_1 is the 1^{st} quartile, Q_3 is the 3^{rd} quartile and R is the interquartile range. Expected response time ($expected_{rt}$) is defined by operation type in the SLA. This time represents the maximum time (or upper bound) a query should take without disrespecting the SLA. The violated execution time ($violated_{et(i)}$) represents time spent by a query i that did not meet the SLA and that is not an outlier.

$$underprov = \frac{\sum_{i=1}^{n} \frac{violated_{et(i)}}{expected_{rt(i)}}}{n} \quad (1)$$

The higher *underprov* is, the less elastic the database system is, because more queries violates the SLA. $\frac{violated_{et(i)}}{expected_{rt(i)}}$ is always greater than 1, since it is applied only for violated queries.

2.4 Provider perspective

From a customer perspective, we presented *underprov* metric. For a PaaS provider, besides measuring that, it is also essential to evaluate how efficient the database system is to use only the minimum amount of resources to meet the SLA. Thus, from a provider perspective, our approach proposes *underprov* exactly as showed in the last section, based on observed SLA, and *overprov*, based on the charged level of resources. Finally, we put *underprov* and *overprov* together to get a single dimensionless value for elasticity of cloud database systems.

Over-provisioning penalty (*overprov*): When there is over-provisioning, the provider offers more resources than necessary to meet a demand. Thus, the provider may be subject to a higher operating cost than the necessary to meet the SLA. In this situation, the database system has a number of resources (i.e. nodes) that are running a given workload, but this amount may be higher than necessary. Unlike *underprov*, this metric does not make sense from a consumer perspective, since for a PaaS consumer there is no problem to have more available resources if that does not imply in a cost increase. *overprov* considers the execution time of queries performed when the database system is over-provisioned. We define this metric (shown on Equation 2) as the average of the ratio lower bound time by execution time for those m queries that are considered over-provisioned. *overprov* moves the $expected_{rt}$ to the numerator since in an over-provisioning scenario the execution time is supposed to be lower than the expected one. Similarly to *underprov* outlier values are disconsidered. The query execution time ($query_{et}$) is the time spent by a query i that is considered in the *overprov* calculation.

$$overprov = \frac{\sum_{i=1}^{m} \frac{expected_{rt(i)}}{query_{rt(i)}}}{m} \quad (2)$$

Elasticity for Cloud Database System ($elasticity_{db}$): Since the provider is affected both on under- and over-provisioning scenario, from his perspective, elasticity can be given by the weighted average of metrics from both scenarios. Worthwhile noticing that the penalty due to over-provisioning should have a lower weight than the under-provisioning one, since that affects only one of the parties. To provide a dimensionless metric that quantifies elasticity from the provider perspective, we suggest the formula 3 and provide the variable x to set how bigger *underprov* weight is when compared to *overprov* weight (fixed to 1, for a matter of simplicity). Numerical domain of x is $[1, \infty)$.

$$elasticity_{db} = \frac{x * underprov + overprov}{x + 1} \quad (3)$$

For instance, x weight could be defined based on costs. In this case, x could be defined by the cost of paying for underprovisioning penalties and *overprov* weight (set to 1) by the cost that could be saved if some resources had been released when environment was overprovisioned. Lower values of $elasticity_{db}$ indicates a more elastic cloud database system, since it makes better use of resources while meeting the SLA. Our metric meets tests of reasonableness, such as (i) elasticity is non-negative and (ii) elasticity captures both over- and under-provisioning.

3. EXPERIMENTS

3.1 Environments

Since our goal is to validate our metrics and not compare elasticity among database systems, we considered one database system in our experiments and compared metrics in two scenarios: (i) with elasticity, where the Decision Taker scales in and out and (ii) without elasticity, where no machine is added or removed during the workload. Decision Taker is the module responsible for adding or removing a node depending on monitored CPU usage [Almeida 2013a]. Cassandra (version 1.1.5) was chosen as the database system, due its wide adoption both by academy and industry. The following machines were used in our experiments: 1 EC2 instance (m1.medium) to run BenchXtend tool; 1 EC2 instance (m1.small) to run the Instance Manager. In addition, we set up 2 instances (m1.large) as seeds and 2 instances (m1.large) for data nodes. Seeds are started before the workload execution while data nodes keep turned off until the Decision Taker starts one of them.

3.2 Execution

We run YCSB Core Workload E (Short ranges - 95% scan, 5% insert), populating the database with 4 millions 1-KB 10-field records. Workload was executed for 4 hours, number of clients was changed according to a timeline (Figure 1) and a Linear function was chosen to interpolate timeline entries. We start Cassandra seeds and then perform the BenchXtend load phase. After that, we restart seeds and then start Monitor and Decision Taker. Before starting the run phase, we check if no compaction of Cassandra SSTables is being performed. During compaction [DataStax 2013] there is a temporary spike in disk space usage and disk I/O that may affect our results. Each started instance had Cassandra already configured, but the database is empty. As soon as we add or remove an instance, Instance Manager updates the *hosts* property of the YCSB workload file in order to let it know about the cluster change. In the SLA file, we set expected response time by operation type. For insert operations we set 200000μs and 100000μs as upper and lower bounds, respectively. For scan operations the values were 700000μs and 350000μs. These values were defined after some experiments in our environment and there is no rule of thumb to define them. x weight was empirically set to 5.

3.3 Results

The first chart of Figure 1 plots when machines are actualy added or removed and compare them with client variation. The time to add (bootstrap) a Cassandra node may be considerably high and may vary a lot. 5min were taken to add the 1^{st} machine, while 35min to the 2^{nd} and 14min to the 3^{rd} one. A reason for this is that Amazon EC2 presents performance issues depending on the time the experiments are executed and on the CPU model of the physical machines where the VMs are placed. Another reason is the cost of data streaming among Cassandra nodes. To remove (decommission) Cassandra nodes the variation was lower (about 2min30s for each of 3 decommission operations).

As we can see on Figures 1 and 2, response times increase as number of clients goes up and reduces as number of clients goes down. Thus, it is fairly reasonable to assert that changing number of clients directly affects response time. Although the number of clients is pretty much the same on first and second peaks, on the second one (from 7000s to 11000s) we can notice on Figure 1 higher values of

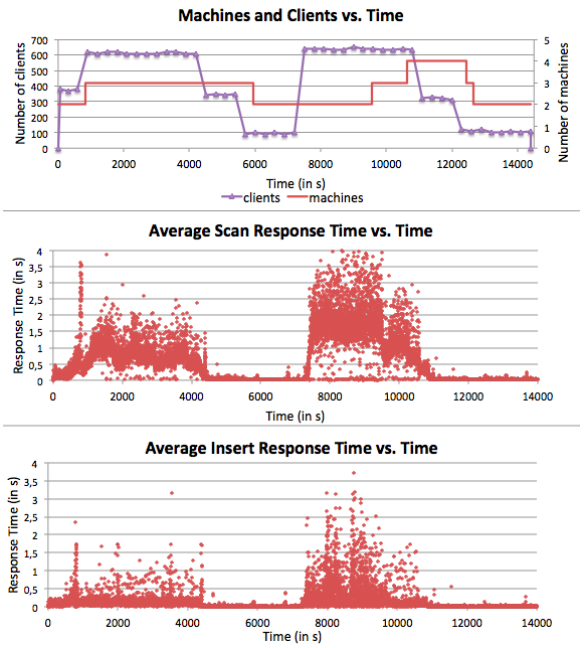


Fig. 1. Experiments with elasticity

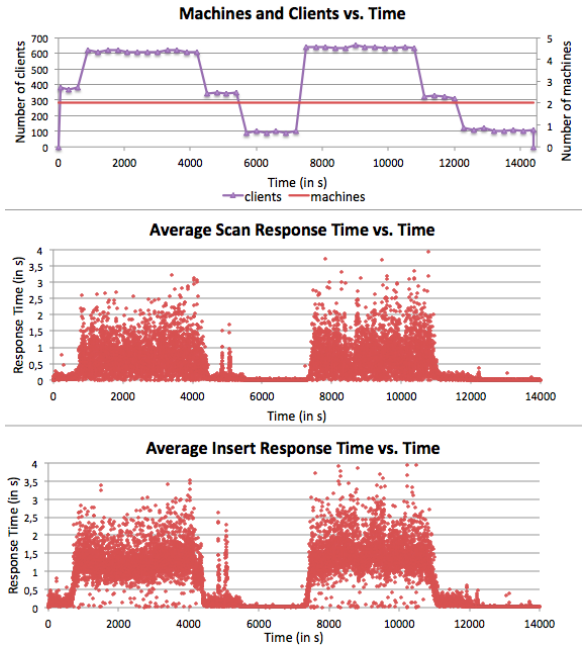


Fig. 2. Experiments without elasticity

	Scan		**Insert**	
	with elasticity	**w/o elasticity**	**with elasticity**	**w/o elasticity**
Total of underprov queries	514719	987528	16302	33116
Underprov metric	**2.019751**	**3.084927**	**1.439694**	**2.613020**
Total of overprov queries	2426529	1767091	137090	93563
Overprov metric	**16.859536**	**15.631292**	**22.129353**	**23.521127**
Elasticitydb metric	**4.493048**	**5.175988**	**4.887970**	**6.097704**

Table I. Metrics gathered in both experiments

response time, due to the long bootstrap (that is an I/O-intensive operation) time, about 35min, to add the 2^{nd} machine. Scan operations require multiple I/Os that are considerably impacted by bootstrap. Insert operations performed well since Cassandra is write-optimized. In addition, we can see that insert operations perform better when new Cassandra nodes are added.

Metrics results are shown on Table I. For *underprov* of scan operations our model could capture the improvement on adding machines when system was overloaded. In this case, the number of violated queries is 52% lower than number of non-elastic experiment and *underprov* had also a lower value. Unlike what was expected, *overprov* of scan operations showed a higher value for the elastic experiment. This may suggest the strategy adopted by Decision Taker should be improved to drop earlier machines when the system is overprovisioned. $elasticity_{db}$ of scan operations is lower in elastic scenario (4.493048) than in the non-elastic one (5.175988), even with a higher value of *overprov*. For insert operations, metrics values showed lower values on elastic experiment, as expected.

4. RELATED WORK

[Klems et al. 2012] discuss about elastic scalability but do not present a metric for that and mention some metrics that do not consider SLA rules. [Shawky and Ali 2012] provide metrics inspired on elasticity definition from physics, focus on network bandwidths instead of database-specific aspects and did not use real data in their experiments. [Islam et al. 2012] [Dory et al. 2011] [Cruz et al. 2013]

discuss elasticity but do not propose metrics to compare their results with other works. [Cruz et al. 2013] presents improvements when compared to [Konstantinou et al. 2012] but no common indicator is used to analytically measure and compare elasticity of cloud data systems. [Islam et al. 2012] propose ways to quantify the elasticity concept in a cloud. They define a measure that reflects the financial penalty to be paid to a consumer, due to under- or over-provisioning. However, it does not take into account DBMSs features like response time. [Dory et al. 2011] provide definitions of elasticity for database and a methodology to evaluate the elasticity. However, these definitions deal only with under-provisioning scenarios and do not consider aspects like SLA, penalties, and resources. [Cooper et al. 2010] present *elastic speedup* and *scaleup*. The first metric illustrates the latency variation as new machines are instantiated. The second one is a traditional metric and does not encompass elasticity aspects. Even though these metrics are useful, they do not illustrate the consumer perspective.

5. CONCLUSION AND FUTURE WORK

In this work, we presented metrics to measure the elasticity of cloud databases. Our model presented metrics based on SLA and from two different perspectives. According to the analysis of experiments, we could validate our metrics since our model could capture the correct variation of elasticity between a scenario with elasticity and another without this. These metrics can now be used to compare elasticity of cloud database systems and to help providers on tuning strategies to add or remove resources on demand. As future work, we intend to execute experiments to compare elasticity of MongoDB and Cassandra and perform a statistical analysis to try to reduce the number of parameters of our model.

REFERENCES

Almeida, R. Benchxtend: a tool to benchmark and measure elasticity of cloud databases. In *Simpósio Brasileiro de Bancos de Dados - SBBD 2012 - Workshop de Teses e Dissertações*. pp. 93–98, 2012.

Almeida, R. F. rodrigofelix/benchxtend-monitor. `https://github.com/rodrigofelix/benchxtend-monitor`, 2013a.

Almeida, R. F. rodrigofelix/YCSB. `https://github.com/rodrigofelix/YCSB/`, 2013b.

Cassandra. The apache cassandra project. http://cassandra.apache.org/, 2013.

Cooper, B. F., Silberstein, A., Tam, E., Ramakrishnan, R., and Sears, R. Benchmarking cloud serving systems with ycsb. In *SoCC*. New York, NY, USA, pp. 143–154, 2010.

Cruz, F., Maia, F., Matos, M., Oliveira, R., Paulo, J., Pereira, J., and Vilaca, R. Met: Workload aware elasticity for nosql. In *EuroSys 2013*, 2013.

DataStax. About Writes in Cassandra. `http://www.datastax.com/docs/1.1/dml/about_writes`, 2013.

Dory, T., Mejías, B., Roy, P. V., and Tran, N.-L. Measuring elasticity for cloud databases. In *Proceedings of the The Second International Conference on Cloud Computing, GRIDs, and Virtualization*. Berlin, Heidelberg, 2011.

Elmore, A. J., Das, S., Agrawal, D., and El Abbadi, A. Zephyr: live migration in shared nothing databases for elastic cloud platforms. In *SIGMOD '11*. New York, NY, USA, pp. 301–312, 2011.

HBase. Apache hbase. http://hbase.apache.org/, 2013.

Herbst, N. R., Kounev, S., and Reussner, R. Elasticity in Cloud Computing: What it is, and What it is Not. In *Proceedings of the 10th International Conference on Autonomic Computing (ICAC 2013), San Jose, CA, June 24–28*, 2013. Preliminary Version.

Islam, S., Lee, K., Fekete, A., and Liu, A. How a consumer can measure elasticity for cloud platforms. In *ICPE'12 - Second Joint WOSP/SIPEW International Conference on Performance Engineering*. New York, NY, USA, 2012.

Klems, M., Bermbach, D., and Weinert, R. A runtime quality measurement framework for cloud database service systems. In *Proceedings of the 8th International Conference on the Quality of Information and Communications Technology*. IEEE, Conference Publishing Services (CPS), pp. 38–46, 2012.

Konstantinou, I., Angelou, E., Tsoumakos, D., Boumpouka, C., Koziris, N., and Sioutas, S. Tiramola: elastic nosql provisioning through a cloud management platform. In *Proceedings of the 2012 ACM SIGMOD*. New York, NY, USA, pp. 725–728, 2012.

Minhas, U. F., Liu, R., Aboulnaga, A., Salem, K., Ng, J., and Robertson, S. Elastic scale-out for partition-based database systems. In *SMDB '12, ICDE Workshops*. Washington, DC, USA, pp. 281–288, 2012.

MongoDB. Mongodb. http://www.mongodb.org/, 2013.

Shawky, D. and Ali, A. Defining a measure of cloud computing elasticity. In *Systems and Computer Science (ICSCS), 2012 1st International Conference*. Lille, FR, pp. 1–5, 2012.

A Live Migration Approach for Multi-Tenant RDBMS in the Cloud

Leonardo O. Moreira, Flávio R. C. Sousa, José G. R. Maia,
Victor A. E. Farias, Gustavo A. C. Santos, Javam C. Machado

Universidade Federal do Ceará (UFC), Brazil
{leoomoreira, sousa, gilvanm, gsantos, javam}@ufc.br, victorfarias@lia.ufc.br

Abstract. Cloud computing is a trend of technology aimed at providing on-demand services with payment based on usage. To improve the use of resources, providers adopt multi-tenant approaches, reducing the operation costs of services. Moreover, tenants have irregular workload patterns, impacting in the guarantees of quality of service, mainly due to interference between the tenants. This paper proposes an approach to improve quality of service for multi-tenant RDBMS. This approach uses migration techniques of tenants, system monitoring and benefits of cloud infraestructure to improve performance and reduce costs of the provider. We carried out experiments on performance and resource usage in order to evaluate our approach.

Categories and Subject Descriptors: H.2 [**Database Management**]: Miscellaneous; H.3 [**Information Storage and Retrieval**]: Miscellaneous; I.7 [**Document and Text Processing**]: Miscellaneous

Keywords: Cloud Computing, Database Live Migration, Multi-tenancy, Quality of Service, Service Level Agreement

1. INTRODUCTION AND MOTIVATION

Cloud computing is a paradigm of service-oriented computing and has revolutionized the way computing infrastructure is abstracted and used. Elasticity, pay-per-use pricing, and economies of scale are the major reasons for the successful and widespread adoption of cloud infrastructures. We consider a cloud computing enviroment as a set of Virtual Machines (VMs) which may be assigned to different Physical Machines (PMs). Since the majority of cloud applications are data-driven, Database Management Systems (DBMSs) are critical components in the cloud software stack [Elmore et al. 2011]. When users purchase computing time from a cloud provider, typically both sides, user and provider, establish the Quality of Service (QoS) via a Service Level Agreement (SLA). It is crucial that cloud providers offer SLAs based on performance for their customers [Schad et al. 2010]. For many systems, most of the time consumed by requests is related to the DBMS, thus it is important that quality is applied to the DBMS to control the consumed time [Sousa et al. 2012].

To improve the resource management, increase resource utilization and reduce costs, providers implement resource sharing among tenants [Barker et al. 2012]. Multi-tenant is a technique for consolidating various tenant applications on a single system, and is often used to eliminate the need for separate systems for each tenant. Sharing of resources at different levels of abstraction and distinct isolation levels result in various multi-tenant models; the shared VM, shared DBMS, shared database and shared table are well known [Lang et al. 2012]. Such sharing improves providers' profits. Multi-tenant DBMSs have been used to host multiple tenants (databases) into a single system, allowing efficient resource sharing at different levels of abstraction and isolation [Elmore et al. 2011].

Database migration is a technique to move one or more tenants to another environment when

changes in the performance of this tenant are detected [Barker et al. 2012]. This technique can be used to decrease the load on the host environment, and also to consolidate multiple tenants in an environment that they may share resources with reduced interference. Thus, in a way, it becomes possible to establish a scenario allocation for a set of tenants in a set of resources, where the main objective is to reduce interference between such tenants. In the cloud, the migration brings up some costs, for example, SLA costs and related costs to human intervention. However, to reduce these costs, the database migration technique must make fair decisions that involve the following variables: "when" determines the instant of time that the tenant must be migrated, "which": specifies what tenants should be targets of migration, "where": determines the tenant destiny and "how" that determines the manner, process or procedure used to migrate the tenant [Barker et al. 2012].

According to [Chaudhuri 2012], an interesting challenge is to develop techniques to guarantee the performance of multi-tenant DBMS. Dealing with unpredicted load patterns and elasticity is a challenging but critical task to ensure that tenant's SLAs are met [Elmore et al. 2011]. Moreover, before develop new techniques, it is necessary to understand how the workload of a tenant influences other tenant's behavior and even the isolation provided by a DBMS to prevent interference between tenants. This problem is addressed in some works [Elmore et al. 2011] [Ahmad and Bowman 2011] [Xiong et al. 2011] [Lang et al. 2012] [Hatem A. Mahmoud and El-Abbadi 2012] [Schiller et al. 2013] [Moon et al. 2013]. However, the studies do not address aspects of performance, they are resources-oriented because they focus on consolidating as many tenants in same hardware or VM, and/or do not check the isolation of each tenant.

This paper presents an approach to improve the QoS for multi-tenant RDBMS (*Relational Database Management System*) in the cloud. Our approach aims to reduce SLA violations through the migrating tenants when they need a resource with greater capacity. Furthermore, a performance experiment was made to show the effectiveness of our approach, in which techniques for migration, monitoring and QoS are used to achieve these goals. The major contributions of this paper are as follows:

—We present an approach to improve the QoS for multi-tenant RDBMS in the cloud by the migration techniques. While other approaches to improve QoS of resources have been proposed, our approach is to tackle aspects of performance, checking the interference between each tenant in the same RDBMS by monitoring techniques and SLA violations.
—We present a database migration strategy for cloud-based multi-tenant environments. Our strategy migrates the tenant quickly according to the current workload.
—We present a full prototype implementation and experimental evaluation of our end-to-end system. We demonstrate that this strategy is effective in managing databases in scenarios from data-driven cloud applications.

Organization: This paper is organized as follows: Section 2 explains theoretical aspects and implementation of our approach. Evaluation results are presented in Section 3. Section 4 surveys related work and, finally, Section 5 presents the conclusions.

2. OUR APPROACH

In this section we describe our approach to improve the performance of multi-tenant database systems. We use monitoring techniques to check the status of tenants according to the workload to ensure the SLA. To develop our approach, we extended our QoSDBC (*Quality of Service for Database in the Cloud*) architecture for managing databases in the cloud [Sousa et al. 2012].

We use shared DBMS model, because it is used more than shared table and shared database models [Elmore et al. 2011], and shared VM has lower interference among tenants. Thus, according to the model used, which has a database corresponding to a tenant on the system, in our approach, each VM can run multiple RDBMSs. Despite using more resources, this approach reduces interference among

tenants as it works with fewer tenants on the same RDBMS. We use the definitions of QoS from our previous work [Sousa et al. 2012]. We use the SLA metric of response time: The value of maximum response time expected for processing the input workload. In our approach, each service has a SLA associated with it (e.g. Wiki$_{SLA}$= 0.5s).

To achieve our goals, some components had to be changed, while others had to be implemented from scratch. An overview of the architecture of the QoSDBC is shown in Figure 1.

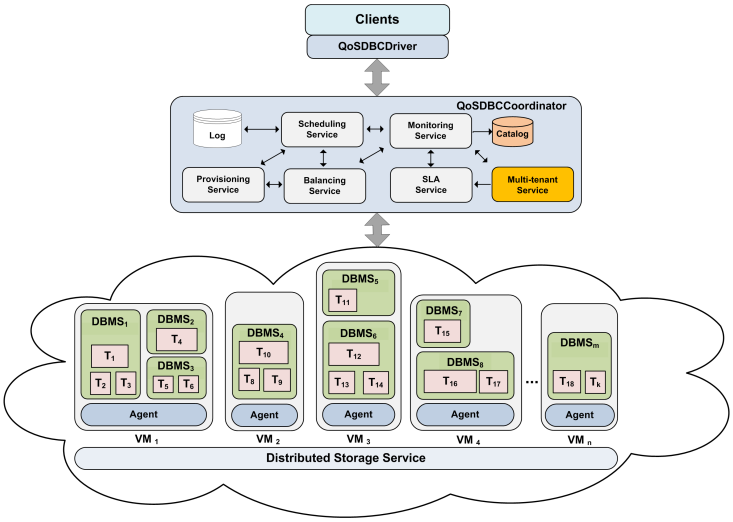


Fig. 1. QoSDBC architecture.

The *QoSDBCDriver* component is used by Java applications that want to use our platform of data management in the cloud. This component implements all the features of a common JDBC (*Java Database Connectivity*) driver. In this sense, any application that uses the JDBC standard to interact with the database, can migrate to QoSDBC. The *QoSDBCCoordinator* is the component responsible for receiving requests for connections through *QoSDBCDriver*. Moreover, it is the component that acts as an orchestrator of all the actions that are performed by QoSDBC. The component that has the responsibility of maintaining information about the cloud environment for QoSDBC is the *Agent*. It provides information from the environment through the *Catalog* component. The *Agent* knows the location of the *Catalog* and uses a JDBC connection to send the information collected in the VM. The pieces of information being monitored for a VM are: free CPU percentage, free space of main memory and secondary memory size available. Also the pieces of information that are tracked for each DBMS are: number of active connections for each database and their sizes they occupy in secondary memory. To find out what queries are running on tenants, the *QoSDBCCoordinator* intercepts the query, executes it, extracts both its local and global response times. The local response time is the time it takes for the query to run in the tenant, while the global response time is the time it takes for a request to come to the QoSDBC and its response returns to the tenant. All this information is stored in the *Catalog* component. The *Catalog* component keeps some information about the state of VMs, the RDBMS and tenants. The *Catalog* is a distributed database that is available in the cloud. Thus, characteristics of fault tolerance, availability and redundancy are achieved by the same cloud infrastructure where QoSDBC is operating. This database stores information about what tenants that are active in each VM, number of connections in each RDBMS, the size of each tenant, state of memory, disk and CPU for each VM. The *Log* is the component responsible for maintaining information about the queries that are being sent and executed in tenants. However, only update informations are kept in this database. The purpose of this database is to keep this information to assist the migration process and restore of the tenant.

Our migration strategy is tightly coupled to the components of QoSDBC. For each application, we associate an SLA described by our quality model. When you start QoSDBC, the agents begin monitoring the tenants of RDBMSs and VM to know the state of the environment. When *QoSDBCCoordinator* identifies an SLA violation of a tenant, it explores the monitored data and starts, in parallel, the backup of the tenant, provisioning a new VM to receive the tenant. After that, the collection of queries that came after the backup starts are executed in the old copy of tenant and stored in QoSDBC sytem. Until the completion of the backup process and provisioning of a VM, the tenant is restored in the new VM. When the restore process is completed, to ensure consistency the update queries that were executed in the old copy are propagated to the new VM. At this time, there is a lockout period that the tenant is inactive. When the tenant is fully migrated, the *QoSDBCCoordinator* warns *QoSDBCDriver* for direct connection to the new address of the tenant. After that, the old tenant is deleted.

3. EVALUATION

To evaluate the efficiency of our approach, we use the results of experiments carried out in one of our previous works [Moreira et al. 2012]. In this work, we show that some tenants could generate interference that reduce the QoS of others. In this regard, we will apply our approach to decrease the number of SLA violations during the occurrence of such interference. Our experiments aim to show how our approach can minimize the SLA violations between tenants, thereby improving the QoS for tenants. In our experiment we used the RDBMS MySQL 5.5 and OS Ubuntu 12.04.1 LTS with InnoDB engine and 128MB of buffer. We use OLTPBenchmark [OLTPBenchmark 2013] to simulate different workloads to the RDBMS in the experiments. OpenNebula [OpenNebula 2013] was the provider used to perform the experiments. In this work we used only VM instances that have 2 CPUs, 2GB RAM, 30GB of disk. For these experiments, we used the quality model that considers the response time metric. For our experiment environment, we create a VM image with the setup described at the beginning of this section. This will be used in cases where our quality strategy requires a provisioning of a VM. To simulate the tenants, we use the OLTPBenchmark [OLTPBenchmark 2013]. The OLTPBenchmark is a framework to evaluate the performance of different RDBMSs facing settings of OLTP workloads. The framework has several benchmarks representing different applications, such as TPC-C, Twitter, YCSB, AuctionMark, Wikipedia. As tenants we use TPC-C (400MB), YCSB (800MB) and Wikipedia (600MB). For this experiment were specified the following SLAs: TPC-C_{SLA}= 100.0ms, $YCSB_{SLA}$= 50.0ms, $Wikipedia_{SLA}$= 50.0ms. All tenants were placed in a VM and OLTPBenchmark in another VM.

3.1 Experimental Results

The objetive of this experiment is to analyze the number of SLA violations in a situation of increasing workload in only one tenant, the TPC-C. In order to bring the experiments execution closer to that of a real environment, only values after the first 120 seconds of obtained measures were considered to avoid that delays in the startup of tenants could impact the results. In our experiment was performed varying the rate of transactions over a period of time. In the this experiment, the rates of YCSB and Wikipedia were fixed at 60, and the rate of the TPC-C was changed according to the sequence (50, 100, 150, 200, 250), every 300 seconds. Figure 2 show the variation of the average response time for the experiment without QoSDBC system. Already the Figure 3 show the variation of the average response time for the experiment with QoSDBC system.

In [Moreira et al. 2012] we show that these three tenants, TPC-C is what causes more interference. Therefore, we increased the rate of the TPC-C to that had SLA violation and thus show how our approach behaves. In our experiment, the TPC-C tenant, after the first SLA violation, has been migrated to a new VM. As shown in the experiments, we can see that our approach reduces the number of SLA violations for tenant TPC-C. However, during the migration process has been SLA

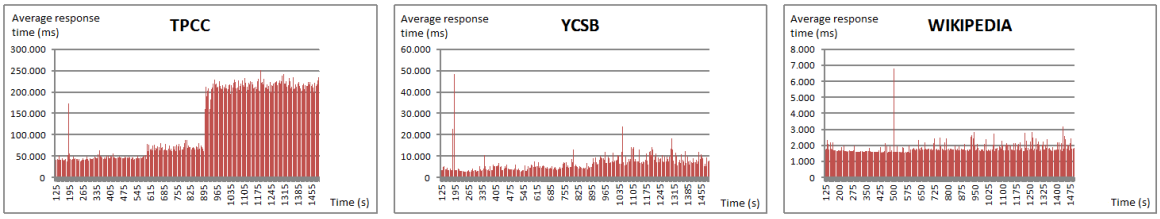


Fig. 2. Average response time without QoSDBC system

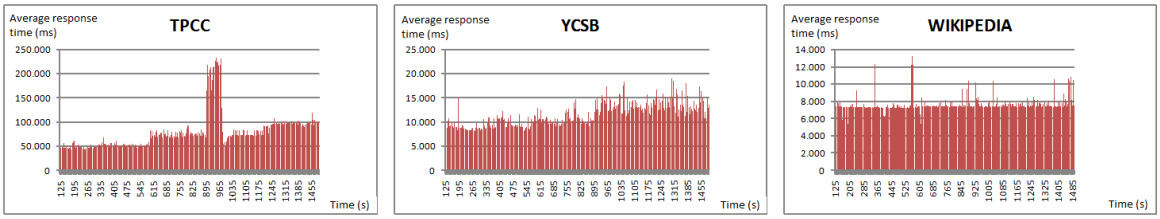


Fig. 3. Average response time with QoSDBC system

violations. The best way would be to anticipate the migration process that had not SLA violations. Furthermore, it was found that there was an increase in the average response time of the transactions. A justification for this is that our approach intercepts queries and performs monitoring tasks, like adding an increase in transaction execution.

4. RELATED WORK

In [Elmore et al. 2011] various multi-tenant models are presented (shared PM, VM, OS, DBMS, database, database schema and database table) for the use of database in virtualized environments. However, only problems achieved when a DBMS has several databases are presented and, still, no evaluation or results of the study are shown. [Barker et al. 2012] presents a solution to databases migration that satisfies some workload constraints. It employs the shared DBMS model and aims to minimize the impact of decreased performance during the migration process of the databases. The relationship between the DBMS and the database is one to one. In this sense, a DBMS can only have one database. However, the focus of this work is the migration of databases, and it does not support other multi-tenant models. The work presented by [Xiong et al. 2011] discusses a solution for managing resources for shared DBMS in the cloud. However, the model presented in this work can not be used on public providers, such as Amazon, since it requires physical access to infrastructure resources.

[Schiller et al. 2013] presents an approach that implements live migration designed for multi-tenant RDBMS, where applications running OLTP workloads execute in tenants. This approach uses a better management of data pages and the buffer pool in the RDBMS. However, this approach is tightly integrated with the used RDBMS, workload type, concurrency control etc. Finally, one of the disadvantageous aspects of this approach is that it not ensures serializability criteria and does not prevent all concurrent access anomalies. In [Moon et al. 2013] it is shown a method to manage replicated multi-tenant databases. To reduce the workload of a given replica that is violating SLA, this work uses a load balancing strategy that finds a subset of replicas where the workload can be directed to. However, the load balancing strategies are appropriated to solve performance problems in replicated environments. Furthermore, it does not eliminate SLA violations when a replica of the tenant is overloaded, because this work does not perform provisioning. [Hatem A. Mahmoud and El-Abbadi 2012], on the other hand, aims to minimize the necessary amount of resources for a set of tenants, satisfying an SLA for each tenant. From the viewpoint of multi-tenant strategies, this work

uses shared DBMS model. Besides, the presented solution is oriented to OLAP applications, and its focus is not to exceed the resources limits of the environment. Therefore, it does not aim the QoS through the SLA.

5. CONCLUSION

The work presents an approach to improve QoS for multi-tenant RDBMS in the cloud by migration techniques, covering its architecture and performance aspects. According to the results, our approach reduces the number of SLA violations through migration and provisioning of new resources. In future work, we intend to anticipate the provisioning and therefore further reduce the number of SLA violations. For this purpose, our prediction work that forecasts workload based on a short window of observation of the environment [Santos et al. 2013] will be added to our infrastructure. In addition, we intend to improve the allocation of tenants in the environment to avoid unnecessary provisioning. The monitored information are not being used in our work. We intend to add this information to our prediction work to improve its effectiveness.

REFERENCES

Ahmad, M. and Bowman, I. T. Predicting system performance for multi-tenant database workloads. In *Proceedings of the Fourth International Workshop on Testing Database Systems*. DBTest '11. ACM, New York, NY, USA, pp. 6:1–6:6, 2011.

Barker, S., Chi, Y., Moon, H. J., Hacigümüş, H., and Shenoy, P. "cut me some slack": latency-aware live migration for databases. In *Proceedings of the 15th International Conference on Extending Database Technology*. EDBT '12. ACM, New York, NY, USA, pp. 432–443, 2012.

Chaudhuri, S. What next?: a half-dozen data management research goals for big data and the cloud. In *Proceedings of the 31st symposium on Principles of Database Systems*. PODS '12. ACM, New York, NY, USA, pp. 1–4, 2012.

Elmore, A., Das, S., Agrawal, D., and Abbadi, A. E. Towards an elastic and autonomic multitenant database. In *NetDB 2011 - 6th International Workshop on Networking Meets Databases Co-located with SIGMOD 2011* , 2011.

Elmore, A. J., Das, S., Agrawal, D., and El Abbadi, A. Zephyr: live migration in shared nothing databases for elastic cloud platforms. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*. SIGMOD '11. ACM, New York, NY, USA, pp. 301–312, 2011.

Hatem A. Mahmoud, Hyun Jin Moon, Y. C. H. H. D. A. and El-Abbadi, A. Towards multitenancy for io-bound olap workloads. Tech. Rep. UCSB-2012-02, CS Department, University of California, Santa Barbara. May, 2012.

Lang, W., Shankar, S., Patel, J. M., and Kalhan, A. Towards multi-tenant performance slos. *Data Engineering, International Conference on* vol. 0, pp. 702–713, 2012.

Moon, H. J., Hacümüş, H., Chi, Y., and Hsiung, W.-P. Swat: a lightweight load balancing method for multitenant databases. In *Proceedings of the 16th International Conference on Extending Database Technology*. EDBT '13. ACM, New York, NY, USA, pp. 65–76, 2013.

Moreira, L. O., Sousa, F. R. C., and Machado, J. C. Analisando o desempenho de banco de dados multi-inquilino em nuvem. In *XXVII Simpósio Brasileiro de Banco de Dados, SBBD 2012*. pp. 161–168, 2012.

OLTPBenchmark. *OLTPBenchmark*, 2013. `http://www.oltpbenchmark.com`.

OpenNebula. *OpenNebula - Open Source Data Center Virtualization*, 2013. `http://www.opennebula.org`.

Santos, G., Maia, J., Moreira, L., Sousa, F., and Machado, J. Scale-space filtering for workload analysis and forecast. In *Proceedings of the 6th IEEE International Conference on Cloud Computing*. IEEE CLOUD '13. IEEE, Santa Clara Marriott, CA, USA, pp. 677–684, 2013.

Schad, J., Dittrich, J., and Quiané-Ruiz, J.-A. Runtime measurements in the cloud: observing, analyzing, and reducing variance. *Proc. VLDB Endow.* 3 (1-2): 460–471, Sept., 2010.

Schiller, O., Cipriani, N., and Mitschang, B. Prorea: live database migration for multi-tenant rdbms with snapshot isolation. In *Proceedings of the 16th International Conference on Extending Database Technology*. EDBT '13. ACM, New York, NY, USA, pp. 53–64, 2013.

Sousa, F. R. C., Moreira, L. O., Santos, G. A. C., and Machado, J. C. Quality of service for database in the cloud. In *CLOSER*, F. Leymann, I. Ivanov, M. van Sinderen, and T. Shan (Eds.). SciTePress, pp. 595–601, 2012.

Xiong, P., Chi, Y., Zhu, S., Moon, H. J., Pu, C., and Hacigumus, H. Intelligent management of virtualized resources for database systems in cloud environment. In *Proceedings of the 2011 IEEE 27th International Conference on Data Engineering*. ICDE '11. IEEE Computer Society, Washington, DC, USA, pp. 87–98, 2011.

Translating Natural Language into Ontology

Ryan Ribeiro de Azevedo[1,2], Fred Freitas[2], Rodrigo G. C. Rocha[1,2], Daniel S. Figueredo[1], Silas Cardoso de Almeida[2] and Gabriel de França Pereira e Silva

[1] Computer Science - UAG/Federal Rural University of Pernambuco, Brazil

[2] CIn/Federal University of Pernambuco, Brazil

{rra2,fred,rgcr,gfps}@cin.ufpe.br, silas.sca@gmail.com, jdanielll@icloud.com

Abstract. In this paper, we present a natural language translator for ontologies and ensure that it is a viable solution to the automated acquisition of ontologies and complete axioms, constituting an effective solution for automating the expressive ontology building Process. The translator is based on syntactic and semantic text analysis. The viability of our approach is demonstrated through the generation of descriptions of complex axioms from concepts defined by users and glossaries found at Wikipedia. We evaluated our approach in an experiment with entry sentences enriched with hierarchy axioms, disjunction, conjunction, negation, as well as existential and universal quantification to impose restriction of properties.

Categories and Subject Descriptors: H.2 [**Database Management**]: Languages;

Keywords: Description Logic (DL), Ontology, Ontology Learning, PLN

1. INTRODUCTION

One of the subfields of Ontology that has been standing out the most along the last decade is Ontology Engineering. Its purpose is to create, represent and model knowledge domains, most of which are not trivial, such as Bioinformatics and e-business, among others. However, as pointed out by [Simperl, E and Tempich, C 2009], the task of Ontology Engineering still consumes a big amount of resources even with the exertion of principles, processes and methodologies to create ontologies, which makes it an arduous and onerous task, besides expensive [Gómez-Pérez, A et al 2004]. Thus, new technologies, methods and tools capable of dealing with the technical and economic challenges regarding the construction of ontologies have been made necessary in order to minimize the need of highly specialized personnel and manual efforts required.

As a consequence, a research line that has been increasingly important through the past two decades is the extraction of domain models from text written in natural language, using Natural Language Processing (NLP) techniques. The process of acquiring of a domain model from text and the automated creation of ontologies, for example, by means of making an analysis of a set of texts using NLP techniques is known as Ontology Learning and was first proposed by [Mädche, A. and S. Staab 2001]. Even so, as affirmed by [Zouaq, A 2011], in spite of the increasing interest and efforts taken towards the improvement of Ontology Learning methods based in NLP techniques [Völker, J *et al*., 2010] [Buitelaar, P and Cimiano, P 2008] [Cimiano, P and Völker, J 2005] [Buitelaar, P et al 2005], the notable potential of the techniques and representations available to the learning process of expressive ontologies and complex axioms has not yet been completely exploited, leaving gaps and unanswered questions that need viable and effective solutions. Among them, these stand out [Pease, A 2011][Völker, J et al 2010]:

- There is a considerable amount of tools and *frameworks* of *Ontology Learning* that have been developed aiming at the automatic or semi-automatic construction of ontologies based on structured, semi-structured or unstructured data. Nonetheless, although useful, the majority of these tools used in Ontology Learning are only capable of creating informal or unexpressive ontologies.

- Evaluating the consistency of ontologies automatically: it is necessary that the automatically created ontologies be assessed by the time of their development, minimizing the amount of errors committed by the ones involved in the development phase and verify whether or not the ontology is contradictory and free of inconsistencies.

All the questions and issues abovementioned justify the approach hereby proposed. It is based in a translator, which consists in the utilization of a hybrid method that combines syntactic and semantic text analysis both in superficial and in-depth approaches of NLP. Demonstrating that a translator for creating ontologies that formalizes and codifies knowledge in OWL DL [Horrocks, I. et al 2007] from sentences provided by users is a viable and effective solution to the process of automatic construction of expressive ontologies and complete axioms.

2. THE APPROACH AND EXAMPLE

One of the goals of this work consists in demonstrating that a translator, through the processing of sentences in natural language provided by users, is capable of creating – automatically and according to the discourse interpreted $\mathcal{ALC}$ [Horrocks, I. et al 2007] ontologies with minimal expressivity. An overview of the translator's architecture and function flow diagram are depicted in Figure 1 and described as follows.

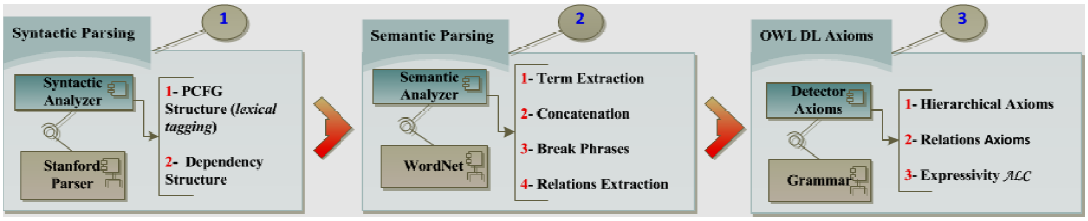


Fig. 1. Translator's architecture and function flow diagram

The architecture of our approach is composed by 3 modules: the **Syntactic Parsing Module (1)**, **Semantic Parsing Module (2)** and the **OWL DL Axioms Module (3)**. The activities executed in the respective modules and their functions are presented in the following sections.

2.1 Módulo Syntactic Parsing

The syntactic analysis of the sentences inserted by users takes place in the **Syntactic Parsing Module (1)**. Two activities are executed by this module, the lexical tagging and the dependence analysis. The results obtained by this module are shown in Fig. 2 and 3**.** We used the sentence **(S1)**: "A self-propelled vehicle is a motor vehicle or road vehicle that does not operate on rails" to illustrate the results obtained by the translator's modules.

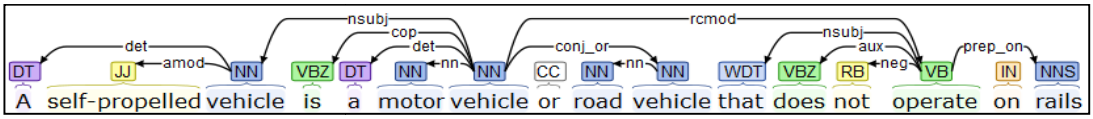


Fig. 2. Lexical tagging and dependence structure

Each word of the sentence (S1) above (Fig. 2) is grammatically classified according to their lexical categories and the dependence between them is attributed.

(**NP** (DT A) (JJ self-propelled) (NN vehicle)) | (**VP** (VBZ is) (NP (DT a) (NN motor) (NN vehicle) | (CC or) (NN road) (NN vehicle)) | (**SBAR** (**WHNP** (WDT that)) | (**VP** (VBZ does) (RB not) (**VP** (**VB** operate) (**PP** (IN on) (**NP** (NNS rails))

Fig. 3. Classification in syntagmatic or sentential categories

Syntagmatic categories are in red and, in black, the lexicon to which each category pertains (See Fig. 3).

2.2 Módulo Semantic Parsing

The results of the activities carried out by the systems of the **Syntactic Parsing Module (1)** are used by the systems of the **Semantic Parsing Module (2)**, which carries out the activities shown in Figure 4 and are detailed as follows.

Fig. 4. Activities carried out in the Semantic Parsing Module

This module initiates its activities by assessing the entry sentence and the referred result of the syntactic analysis obtained in the previous module and then starts the extraction of terms (Term Extraction) that are fit to be concepts of the ontology (Activity (1)). In this phase, terms classified as prepositions (IN), conjunctions (CC), numbers (CD), articles (IN, CC, RB, DT+PDT+WDT) and verbs (EX+MD+VB+VBD+VBG+VBN+VBP+VBZ) are discarded, and the terms classified as nouns (NN+NNS+NNP+NNPS) and adjectives (JJ+JJR+JJS) are indicated as possible concepts of the ontology, therefore, the terms extracted, who are fit to be concepts were: motor/NN, vehicle/NN, road/NN, vehicle/NN, self-propelled/JJ, vehicle/NN e rails/NNS as presented in Fig. 5.

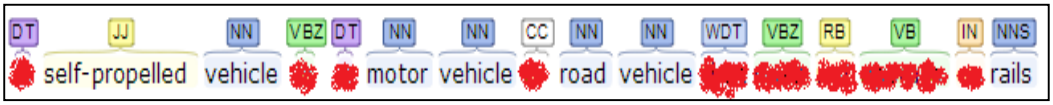


Fig. 5. Result of the extraction of terms

After the term extraction activity is done, the Activity (2), called **Concatenation** is enabled. This activity uses the results of the dependences between the terms (See Figures 2 e 3) and makes the junction of NPs composed by two or more nouns and/ or adjectives inside the analyzed sentence and which, in fact, are related. In the example sentence (*S*1), the concatenation results in the junction of the terms (self-propelled/JJ ↔ vehicle/NN), (motor/NN ↔ vehicle/NN) e (road/NN ↔ vehicle/NN) into an only term, because they are dependent of one another, resulting in just 3 terms: *self-propelled-vehicle*, *motor-vehicle* e *road-vehicle*, and no longer 6 terms, as in the initial phase of the **Term Extraction** activity.

In Activity (3), **Break Phrases,** every time terms or punctuation marks like comma (,), period (.), *and*, *or*, *that, who* or *which* (what we call sentence breakers) are found, the sentences are divided into subsentences and analyzed separately, the result for (S1) was:

A self-propelled-vehicle is a motor-vehicle **| or** road-vehicle **| that** does not operate on rails

The last activity to take place in the **Semantic Parsing Module** is Activity (4), **Relations Extraction**. The relations between the terms are verified and validated through verbs found in the sentences and patterns observed in the translator's inner grammar. The verbs are separated and the terms dependent on verbs are extracted, resulting in:

self-propelled-vehicle **is a** motor-vehicle | self-propelled-vehicle **is a** road-vehicle | self-propelled-vehicle **operate** on rails

This module detects the terms and the relations between them, both hierarchical and nonhierarchical. However, this module neither extracts disjunctions, conjunctions nor generates OWL code

corresponding to the result obtained. The activity of this module is exclusively for detecting terms, their relations and validity.

2.3 Módulo OWL DL Axioms

The function of the **OWL DL Axioms Module** is to symbolically find/learn axioms that prevent ambiguous interpretations and limit the possible interpretations of the discourse, enabling systems to verify and disregard inconsistent data. The process of discovering the axioms is the hardest part of the process of creating ontologies. Here, the axioms discovered correspond to $\mathcal{ALC}$ expressivity. The module recognizes coordinating conjunctions (*OR* and *AND*), labeled CC, indicating the union (disjunction) and intersection (conjunction) respectively for concepts and/or properties, recognizes linking verbs followed by negations, like *does not* and *is not* for negation axioms (¬), besides generating universal quantifiers (∀) and existential quantifiers (∃). It also recognizes *is* and *are* as taxonomic relations (⊑ - hierarchical). The transformations occur in four steps and make use of the results obtained by the previous modules:

Step (1): construction of taxonomic/hierarchical relations. The pattern used here is **<NPs> <VP> <NPs> where <VP> in this case is a (*is a/an*, *is* or *are*).** For all the transformations, the patterns are automatically chosen by the translator.

A self-propelled-vehicle **is a** motor-vehicle → *self-propelled-vehicle* ⊑ *motor-vehicle*
self-propelled-vehicle **is a** road-vehicle → *self-propelled-vehicle* ⊑ *road-vehicle*

Step (2): construction of nonhierarchical relations. The pattern used here is **<NPs> <VP> <NPs> where <VP>** in this case is a verb other than **(is a/an, is or are)**.

self-propelled-vehicle **operate** on rails → self-propelled-vehicle ≡ ∃*operate.rails*

Step (3): verification of conjunctions and disjunctions. The conjunctions OR and AND are verified and analyzed. They can be associated with concepts and/or properties. The pattern **<NPs> *is a/an*** or ***are*** **<NPs> <CC> <NPs>**, where **<CC>** is the conjunction *Or* or *And* that links two or more **<NPs>** is one of the patterns associated with union and intersections of concepts, and is chosen by the translator resulting in:

*A self-propelled-vehicle is a motor-vehicle **or** road-vehicle* → *self-propelled-vehicle* ⊑ *(motor-vehicle* ⊔ *road-vehicle)*

Step (4): detection of negations. The fourth analysis detects the negations, its dependences and classifies the sentence to apply the patterns. Two negations are possible: negations and disjunctions of concepts and negations of properties. Two patterns or a junction of these patterns are taken into consideration in the process of extraction of negation axioms for hierarchies: **<NPs> is not <NPs>** and the pattern **<NP>does not<VP><NP>** for negation of properties. For (S1), the following result was obtained:

self-propelled-vehicle that does not operate on rails → *self-propelled-vehicle* ⊓ ¬∃*operate.rails*

The final result, after the integration of the partial results obtained by the three modules, for (S1) in OWL 2 code, was:

(S1): *A self-propelled vehicle is a motor vehicle or road vehicle that does not operate on rails* → *self-propelled-vehicle* ≡ *(motor-vehicle* ⊔ *road-vehicle)* ⊓ ¬∃*operate.rails*

Our approach generated 4 axioms, 2 of which being hierarchical axioms, 1 being the union between concepts and 1 other of negation of properties. The approach proposed by us is effective in patterns like this and makes correct or approximately correct interpretations of what the user desires.

3. EXPERIMENTS, PRELIMINARY RESULTS AND DISCUSSION

In order to validate the translator, sentences from various knowledge domains were used. The set of data utilized in the experiments contains a total of 120 sentences and in all of them there were negation axioms and in all of them there were negation, conjunction or disjunction axioms and/or two and/or three types of axioms in the same sentence, as well as axioms with definition of terms hierarchy. We opted for sentences found in *Wikipedia* glossaries because they offered in principle a controlled language without syntactic and semantic errors, besides providing a great opportunity for automatic learning. For discussion and comparison purposes, some of the sentences analyzed were extracted from the work of [Völker, J et al 2010], which were also obtained from *Wikipedia*. Some examples of sentences having negation, union and conjunction axioms used in the experiments and the respective results generated by the translator, along with a discussion on these results are shown as follows.

Processed Sentence (1): Juvenile is an young fish or animal that has not reached sexual maturity.
Result: Juvenile ≡ (young-fish ⊔ animal) ⊓ ¬∃hasReached.Sexual-maturity

Discussion: the result of the analysis of the sentence is different from the results of the processing performed by the LExO system [Völker, J et al 2010]: Juvenile ≡ (young ⊓ (Fish ⊔ Animal) ⊓ ¬∃reached.(Sexual ⊓ Maturity). The compared system (LExO) classifies *young*, *fish* and *animal* as distinct terms, however, by the interpretation in natural language of the sentence in analysis, the word *young* is an adjective of the *fish* concept, thus, our approach classifies and represents '*young fish'* as a composite noun, that is, composing a single concept (*young-fish*). The same occurs for s*exual maturity,* being interpreted by LExO as distinct concepts when they are not, whereas in our approach these two terms are classified as a single concept in the same way as the classification of the previous concept (*young-fish*). We can also observe the creation of two axioms, one of union of concepts and one of negation of property..

Processed Sentence (2): A Vector is any agent (person, animal or microorganism) that carries and transmits an infectious pathogen
Result: Vector ≡ Agent ⊓ (∃carriesPathogen.InfectiousPathogen ⊓ ∃transmitsPathogen.InfectiousPathogen)

Discussion: the result obtained in **(2)** was also compared with resulted generated by LExO: Vector ≡ (Organism ⊓ (*carries* ⊔ ∃*transmit.Pathogen*)). The verb *to carry* was not correctly classified as an existential restriction when analyzed by LExO, whereas in our approach, the sentence was coherently classified, the existential quantifier was created and the disjunction of the relations created was performed, where *carriesPathogen* and *transmitsPathogen* are disjoint, which evidences the accurate interpretation of the sentence in natural language.

Sentença Processada (3): The Budget is a list planned expenses and is a list planned revenues.
Resultado: Budget ≡ (List-plannedexpenses ⊓ List-plannedrevenues)

Discussão: : In this sentence, the concept *budget* forms a hierarchy with the other two concepts (*list-plannedrevenues* and *list-plannedexpenses*). Besides, it generates an intersection of both terms, meaning that individuals pertaining to the concept *budget* pertain to the set of individuals of both concepts at the same time.

4. VALIDATION

The objective of planning and performing the experiments was to verify the model constructed and test it in order to answer the following research questions: is the translator capable of constructing minimal-expressivity $\mathcal{ALC}$ ontologies and represent knowledge? And, is the translator capable of identifying rules and axioms inherent to $\mathcal{ALC}$ expressivity based on the texts produced from dialogues

with users? In order to answer these questions, the following hypothesis was formulated to assess the translator's performance during the experiment:

- $H_{a,1}$: The translator constructs the ontology correctly in more than half of the cases;
- $H_{0,1}$: The translator does not construct the ontology in more than half of the cases.

Analyzing all the 120 sentences, we obtained the following results: in 75% of the sentences analyzed (90 sentences), the translator detected and created coherently the axioms, whereas in 30 sentences (25%, also including the ones the translator could not possibly solve in any way), the translator did not detect the axioms coherently and committed errors; however, in 24 out of those 30 sentences, the translator created axioms and made possible the recreation of the ontologies through the proceeding of inserting new definitions. The right unilateral binomial test was applied and the following results obtained.

Limiar → 50% | Level of significance α (5%) | p-valor = 1.886e-08

Sustention: As p-value is lower than the significance of the null hypothesis ($H_{0,1}$: The translator does not construct the ontology in more than half of the cases), it is not accepted, that is, the translator statistically constructs the ontology correctly in more than half of the cases. Using the same binomial test we confirm that this success ratio is statistically superior to 67% (0.67 < success ratio < 1; p-value=0.03636). Therefore, we conclude that the success ratio presented by our approach, 75%, is statistically higher than 67% and, in fact, significant.

5. CONCLUSIONS AND FUTURE WORKS

In this paper, we describe an approach to automatic development of expressive ontologies from definitions provided by users. The results obtained through the experiments evidence the need of automatic creation of expressive axioms, sufficient to creating ontologies with $\mathcal{ALC}$ expressivity, besides the success in the identification of rules and axioms pertaining to $\mathcal{ALC}$ expressivity. We also conclude that the translator can aid both experienced ontology engineers and developers and inexperienced users just starting to create ontologies. As future works, we include the integration of our approach with other existing approaches in the literature, the creation of a module for automatic inclusion of unprecedented patterns in the translator and one module for automatic insertion of individuals for terms of ontologies created by the translator.

REFERENCES

BUITELAAR, P and CIMIANO, P. *Ontology learning and population: Bridging the gap between text and knowledge*. In: Frontiers in Artificial Intelligence and Applications Series (Vol. 167). IOS Press, 2008.

BUITELAAR, P., CIMIANO, P. and MAGNINI, B. *Ontology learning from text: An overview*. In: Buitelaar, P., Cimiano, P., & Magnini, B.(Eds.), *Ontology learning from text: Methods, applications and evaluation* , pp. 3–12. IOS Press, 2005.

CIMIANO, P and VÖLKER, J.,. Text2Onto. *NLDB*, 227–238. 2005.

GÓMEZ-PÉREZ, A, FERNÁNDEZ-LÓPEZ, M and CORCHO, O. *Ontological Engineering with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web*. In: Advanced information and knowledge processing, Springer, London, 2004.

MADCHE, A. and STAAB, S. *Ontology learning for the semantic web*. In: IEEE Intelligent Systems, 16(2):72-79, 2001.

PEASE, A. *Ontology: A Practical Guide*. Published by. Articulate Software Press. USA. 2011.

SIMPERL, E and VTEMPICH, C. *Exploring the Economical Aspects of Ontology Engineering synthetic*. In: Handbook on Ontologies, International Handbooks on Information Systems. Springer, Berlin, Germany, pp. 337–358, 2009.

VÖLKER, J. *Learning Expressive Ontologies*. No. 002. In: Studies on the Semantic Web, AKA Verlag / IOS Press, ISBN: 978-3-89838-621-0. 2009.

ZOUAQ, A. *An Overview of Shallow and Deep Natural Language Processing for Ontology Learning*. Chapter 2. Pg 16-37. In: Ontology Learning and Knowledge Discovery Using the Web: Challenges and Recent Advances. IGI Global. EUA . ISBN 978-1-60960-625-1 (hardcover). 2011.

HORROCKS, I. et al. *OWL: a Description-Logic-Based Ontology Language for the Semantic Web*. In: The Description Logic Handbook: Theory, Implementation and Applications. P458-486. Cambridge University Press. 2ed. 2007

XprefRec: minimizando o problema de cold-start de item com mineração de preferências

Cleiane Gonçalves Oliveira[1,2], Sandra de Amo[1]

[1] Universidade Federal de Uberlândia, Brasil
[2] Instituto Federal do Norte de Minas Gerais - Campus Januária, Brasil
cleiane.oliveira@ifnmg.edu.br, deamo@ufu.br

Abstract. Sistemas de recomendação de filtragem colaborativa encontram diversos desafios para alcançar acuradas recomendações. Um deles é o denominado *cold-start* de item que é o fato do sistema não ser capaz de recomendar itens que nunca foram avaliados por outros usuários. Neste artigo apresentamos uma proposta para minimizar esse problema, o ***XPrefRec***. Trata-se de um sistema de recomendação híbrido, seguindo os princípios gerais das abordagens clássicas de *filtragem colaborativa* e *recomendação baseada em conteúdo*, e incorporando técnicas de *mineração de preferências contextuais* e técnicas de *extração de consenso*. Testes realizados em dados reais de filmes mostram resultados bastante promissores de precisão, revocação e cobertura quando comparados aos resultados obtidos utilizando-se um método clássico de recomendação (*Content-boosted collaborative filtering (CBCF)*), também baseado em uma abordagem híbrida.

Categories and Subject Descriptors: H.2.8 [**Database Management**]: Data Mining; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; H.4 [**Information Systems Applications**]: Miscellaneous

Keywords: Mineração de dados, Mineração de preferências, Regra de preferência contextual, Sistemas de recomendação

1. INTRODUÇÃO

Cada vez mais sistemas de recomendação vêm sendo utilizados em diversos cenários devido a sua capacidade de predizer sobre o que os clientes/usuários têm preferência. A técnica mais popular é a denominada *filtragem colaborativa* (CF) introduzida em [Resnick et al. 1994]. Esta técnica recomenda itens a um determinado usuário a partir do histórico de compras/avaliações de outros usuários que tenham gostos semelhantes a ele, isto é, que tenham um histórico semelhante de avaliações de produtos. Os sistemas de recomendação CF partem da ideia de que se um usuário gostou das mesmas coisas que outro no passado, ele tende a continuar gostando das mesmas coisas no futuro. Para prover recomendações é necessário também que o usuário ao qual se deseja recomendar (usuário ativo) possua algum histórico de avaliações para receber recomendações.

Porém, a técnica CF enfrenta alguns desafios. Os usuários do sistema, geralmente, avaliam poucos itens, fazendo com que os dados disponíveis sobre avaliações sejam esparsos. Para que um item seja recomendado é necessário que ele tenha sido avaliado pelos usuários similares ao usuário ativo. Desta forma, um item que nunca recebeu nenhuma avaliação não é recomendado. Da mesma forma, um item novo, recém inserido no sistema e que ainda não possui avaliação, também não é recomendado até que um número expressivo de usuários o avalie. Esta dificuldade é denominada como *cold-start* de item [Adomavicius and Tuzhilin 2005].

Entre os diversos trabalhos que tentam tratar esse problema tem-se o *Content-boosted collaborative filtering* (CBCF), introduzido em [Melville et al. 2002]. Este trabalho apresenta um sistema de recomendação híbrido de CF com *recomendação baseada em conteúdo* (CB). Um sistema de recomendação CB recomenda itens semelhantes ao que o usuário já avaliou. Por exemplo, se o usuário avaliou bem um filme de comédia, o sistema CB recomendará outros filmes de comédia. O CBCF aplica a técnica CB para cada usuário do sistema:

We thank the Brazilian Research Agencies CNPq and FAPEMIG for supporting this work.

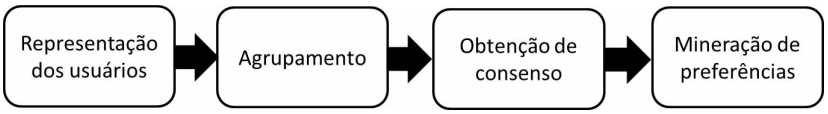


Fig. 1. Fase de processamento *offline* do ***XprefRec***

a partir dos itens avaliados pelo usuário, são preditas avaliações para os itens não avaliados. Desta forma, a técnica CB elimina o problema da esparsidade dos dados fazendo com que todos os usuários tenham avaliado todos os itens. A técnica CF é então aplicada: encontra-se quais são os usuários mais similares ao usuário ativo e a partir de seus históricos são feitas as recomendações. Observa-se que o CBCF consegue tratar o problema de *cold-start* de item, uma vez que todos os itens conseguem ser recomendados porque possuem avaliações de todos os usuários. Porém, quando um novo item é inserido no sistema repete-se todo o processo da técnica CB para predição da avaliação dos usuários àquele novo item, para que assim, em um momento próximo, ele possa ser recomendado.

Neste artigo apresentamos o ***XprefRec***, um sistema de recomendação híbrido CF e CB que utiliza técnicas de *mineração de preferências contextuais* e técnicas de *extração de consenso* entre usuários, a fim de otimizar a questão da recomendação de itens novos para os usuários. O sistema é capaz de tratar o problema de *cold-start* de item de forma mais eficiente que o CBCF (não necessitando refazer o modelo de recomendação a cada inserção de um novo item) sem prejudicar a acurácia da recomendação.

Este artigo é organizado como se segue: na Seção 2 apresentamos o sistema ***XprefRec***, na Seção 3 os resultados dos experimentos em dados reais e na Seção 4 a conclusão e trabalhos futuros.

2. O SISTEMA *XPREFREC*

O ***XprefRec*** apresenta uma abordagem diferenciada de um sistema de recomendação híbrido CF e CB a partir da *mineração de preferências contextuais* e *extração de consenso* entre os usuários. São definidos grupos entre os usuários do sistema que possuam gostos semelhantes. Para cada grupo define-se um gosto consensual que por sua vez é submetido a um minerador de preferências contextuais. Esse minerador apresenta como resultado um conjunto de *regras de preferências contextuais* que será utilizado para predizer preferências de usuários que se enquadram naquele grupo. Dados dois itens, uma *regra de preferência contextual* é capaz de dizer qual item é preferido, ou se não é possível compará-los, a partir das características dos itens. O ***XprefRec***, no momento da recomendação ao usuário ativo, em vez de compará-lo com todos os usuários do sistema, faz a comparação somente com os grupos, sendo que estes são em número bem menor em relação à quantidade de usuários. As regras daquele grupo são então aplicadas sobre os itens para predizer recomendações. As características da técnica CB estão implícitas no minerador de preferências, já que o modelo de preferências extraído envolve o conteúdo dos itens. Assim, para que um item seja recomendado basta que ele atenda às regras associadas ao usuário ativo. Quando um novo item é inserido no sistema este já pode ser recomendado a um usuário ativo, simplesmente aplicando o modelo de preferências relativo ao grupo no qual este usuário se enquadra.

O ***XprefRec*** pode ser dividido na fase de processamento *offline* e na fase *online*, que consiste na atividade de recomendação. Uma visão geral da fase *offline* é apresentada na Figura 1.

2.1 Representação dos usuários

Diferente da representação dos usuários normalmente utilizada em sistemas de recomendação que representam cada usuário por um vetor de notas (como na Tabela I), apresentamos o conceito de *matriz de preferências* (*MPref*) como uma nova maneira de representar os usuários em sistemas de recomendação. Uma *MPref* é uma matriz de dimensão $n \times n$, sendo n a quantidade de itens no sistema. A cada usuário está associada uma *MPref*. Cada posição (a, b) da *MPref* de um usuário u contém um valor entre 0 e 1 que representa o quanto o usuário u prefere o item i_a ao item i_b. Este valor é calculado utilizando-se uma *relação de preferência fuzzy* ([Chiclana et al. 2001]) dada pela fórmula: $r(n_a, n_b) = \frac{n_a}{n_b}$, onde n_a (resp. n_b) é a nota que o usuário associou ao item i_a (resp. n_b). Esta razão é normalizada utilizando-se a função $h(x) = \frac{x}{x+1}$ que além de produzir um número no intervalo $[0, 1]$ também verifica algumas propriedades importantes quando se trata de *graus de preferência*,

Table I. Avaliações do usuário ***u***

i_1	5
i_2	3
i_3	*
i_4	1

Table II. *MPref* relativa ao usuário ***u***

	i_1	i_2	i_3	i_4
i_1	0.50	0.63	*	0.83
i_2	0.37	0.50	*	0.75
i_3	*	*	0.50	*
i_4	0.17	0.25	*	0.50

dentre elas $h(\frac{1}{x}) = 1 - h(x)$[1]. Assim, o valor final normalizado é dado por $f(a, b) = h(r(n_a, n_b))$. Como exemplo, a Tabela I apresenta um conjunto de avaliações dadas por um usuário ***u***. O símbolo * representa o fato que o usuário ***u*** não avaliou o item correspondente, o que consequentemente não permitirá a comparação desse item com nenhum outro. Calculando a relação de preferência fuzzy entre cada par de item, chega-se à *MPref* apresentada na Tabela II. O valor $f(1, 4) = 0,83$, por exemplo, representa o fato de que o usuário ***u*** prefere o item i_1 ao item i_4 com um grau de preferência $0,83$.

As matrizes de preferências são usadas para determinar a semelhança entre os usuários. Um usuário é semelhante a outro se suas matrizes de preferência forem similares (de acordo com alguma noção de similaridade a ser discutida na Seção 2.2). Assim, usuários semelhantes gostam dos mesmos itens com graus de preferência semelhantes.

2.2 Agrupamento de usuários

Os usuários, representados por suas respectivas matrizes de preferência, são organizados em grupos. Para esta tarefa aplica-se o algoritmo de clusterização DBScan introduzido em [Ester et al. 1996] e a distância do cosseno como medida de similaridade entre as matrizes. Para cada grupo é calculada uma *MPref* consensual. Esta tarefa é discutida na Seção 2.3.

De uma certa maneira, pode-se dizer que a *MPref* consensual de um determinado grupo agrega tanto as preferências comuns de seus usuários quanto preferências específicas de cada um, o que resulta na fase de mineração, em uma maior variedade de regras possibilitando melhores recomendações. Além disso, no momento da recomendação as amostras de preferência do usuário ativo (uma matriz bem esparsa) precisará ser comparada com as matrizes consensuais de cada grupo, cujo número será menor que a quantidade de usuários do sistema, como é feito na técnica CF.

2.3 Técnicas alternativas para a *extração do consenso*

A obtenção do consenso constitui uma etapa muito importante para alcançar boas recomendações. Para tanto, propôs-se o estudo de cinco alternativas para obtenção do consenso que são aplicadas ao conjunto de valores de cada posição da matriz consensual. Cada posição (i, j) da *Mpref* consensual só é calculada se mais da metade dos usuários do grupo informaram algum valor para esta posição, caso contrário tal posição permanece com o símbolo "*".

A primeira alternativa (Alternativa (1)) aplica simplesmente a média aritmética entre os valores das correspondentes posições (i, j) de todas as *Mpref* do grupo. A segunda alternativa (Alternativa (2)) aplica a média ponderada sendo o peso de cada usuário a medida do seu coeficiente de silhueta. Este coeficiente mede o quanto o usuário está coeso ao seu grupo e separado dos outros grupos. Desta forma o usuário que possuir maior coeficiente de silhueta possuirá peso maior no cálculo do consenso.

As alternativas (3), (4) e (5) consistem na aplicação dos quantificadores *fuzzy most*, *at least half*, *as many as possible* respectivamente. Quantificadores *fuzzy* expressam termos da linguagem natural e definem quais valores, dentro de um conjunto de valores, serão considerados no cálculo do consenso ([Chiclana et al. 2001]). Vamos ilustrar estas 3 alterativas através de um exemplo. Suponha que temos 4 usuários dentro de um grupo e desejamos calcular o valor de uma certa posição da matriz de consenso deste grupo. Os valores desta posição nas 4 matrizes de preferência constitui um vetor $V = (0.4, 0.6, 0.5, 0.3)$. A aplicação de um quantificador

[1]Outras funções de normalização podem ser utilizadas, desde que verifiquem certas propriedades especificadas em [Chiclana et al. 2001], dentre elas a propriedade $h(\frac{1}{x}) = 1 - h(x)$.

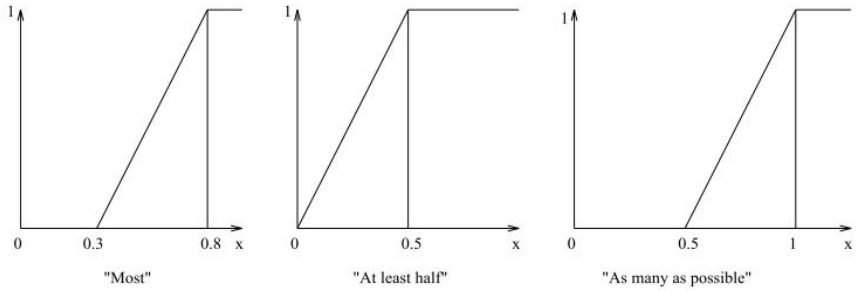


Fig. 2. Quantificadores *fuzzy* e seus parâmetros

fuzzy para obter o valor consensual envolve os seguintes passos: (1) ordenação dos valores do vetor de forma decrescente; (2) cálculo do peso de cada posição do vetor a partir dos parâmetros a e b de cada quantificador (por exemplo, na Figura 2 os parâmetros do quantificador *most* são a = 0.3 e b = 0.8); (3) cálculo da média ponderada dos valores do vetor com seus respectivos pesos. O peso de cada posição i ($i = 1, ..., 4$) é calculado por

$$w_i = Q(i/n) - Q((i-1)/n), \text{ onde } Q(r) = \begin{cases} 0 \text{ se } r < a \\ \frac{r-a}{b-a} \text{ se } a \leq r \leq b \\ 1 \text{ if } r > b \end{cases}$$

Assim, o consenso dos valores do vetor V utilizando-se os quantificadores *most*, *at least half* e *as many as possible* são 0.43, 0.55 e 0.35 respectivamente.

2.4 Mineração de preferências

Sobre a *MPref* consensual de cada grupo é aplicado o minerador de preferências contextuais *CPrefMiner* ([de Amo et al. 2012], [de Amo et al. 2013]). Este minerador explora as características dos itens sobre os quais o grupo tem preferência e retorna um conjunto de *regras de preferências contextuais* que permitem dizer, dados dois itens, qual deles o usuário prefere.

As *regras de preferências contextuais* são mineradas no formato contexto->preferência, exemplo: Diretor="Woody Allen"→ (Categoria=comédia) > (Categoria=drama). A parte à esquerda de "→"é o contexto da regra e a parte à direita é a preferência neste contexto. Esse exemplo, em outras palavras, diz que o usuário prefere filmes de comédia a filmes de drama quando o diretor é Woody Allen.

O uso das regras de preferências contextuais se torna benéfico no tratamento do *cold-start* de item. Ao associar o conjunto de regras do grupo ao usuário ativo, um novo item adicionado ao sistema de recomendação poderá ser comparado, mesmo nunca tendo sido avaliado por outro usuário, se ele puder ser ordenado pelo modelo de preferências (as regras) do grupo.

2.5 Recomendação

A fase *online* do ***XprefRec*** é a tarefa de recomendação. Dado um usuário ativo e seu histórico de avaliação, constrói-se sua matriz de preferência e a compara com as matrizes consensuais de cada grupo do ***XprefRec***. Definido o grupo mais semelhante, o seu respectivo conjunto de regras de preferências contextuais é aplicado sobre os itens para gerar as recomendações. Como as regras são aplicadas a pares de itens, são formados pares dos itens ainda não avaliados pelo usuário ativo para saber quais são recomendados. Se for desejado um *ranking* de recomendações em ordem decrescente de relevância, pode-se aplicar algoritmos que extraem um *ranking* a partir de um conjunto de pares de itens fornecido como entrada (por exemplo, o algoritmo proposto em [Cohen et al. 1999]).

3. RESULTADOS EXPERIMENTAIS

A fim de avaliar o ***XprefRec*** sobre dados reais usamos a base disponível no projeto GroupLens [2] que apresenta dados de avaliações de filmes no formato (IdUsuario,AtributosFilme, Nota). Para compor a base de teste foram escolhidos usuários que avaliaram muitos itens e itens que foram avaliados por muitos destes usuários selecionados. Assim, selecionou-se uma base de 296 usuários, 262 itens e todas as avaliações entre estes usuários e itens, no total de 67.971 avaliações. Este banco é o denominado BD_100. Os itens são representados pelos atributos dos filmes (AtributosFilme): gênero, ator principal, categoria, diretor, ano e idioma. Para comparar o comportamento do sistema em bancos mais esparsos, o conjunto de avaliações foi sendo decrementado de forma estratificada, construindo-se mais 5 bancos de testes. Assim, o banco BD_90 possui 10% a menos de avaliações em relação ao banco original, e assim sucessivamente. A Tabela III apresenta as características de cada banco.

Para validar o sistema proposto usamos a técnica de *k-cross-validation*, para $k = 5$: primeiramente dividimos o conjunto de usuários em 5 partes, sendo que a cada iteração quatro partes são usadas para treinamento e a outra para teste. Para cada usuário de teste também foi realizado o *5-cross-validation* dividindo o conjunto de itens em 5 partes, de tal forma que os itens avaliados por esse usuário estavam divididos de forma estratificada em relação às notas recebidas.

Como as regras de preferências contextuais são aplicadas sobre pares de itens, são definidos sobre o conjunto dos 262 itens todos os pares que se sabe que o usuário avaliou. Como exemplo da Tabela I seriam gerados os pares $(i_1, i_2), (i_1, i_4), (i_2, i_4)$ de forma que o primeiro termo é preferido ao segundo. Estes pares são denominados *pares reais*.

Após o processamento *offline* do ***XprefRec***, as regras são aplicadas sobre os pares reais. Elas podem acertar indicando a ordem correta do par; errar, quando invertem a ordem; ou ser indiferente quando não é capaz de definir a ordem entre os dois itens. A partir disso são definidas as medidas de validação *precisão*(p) e *revocação*(r) bem como a média harmônica entre tais medidas:

$$\text{p} = \frac{acertos}{acertos+erros}; \text{ r} = \frac{acertos}{|paresreais|}; \text{ F1} = \frac{2}{\frac{1}{r}+\frac{1}{p}}.$$

Para verificar como o ***XprefRec*** se comporta em relação ao *cold-start* de item definimos a medida de *cobertura*. Esta medida calcula a capacidade do sistema para recomendar itens que não foram avaliados ainda, tendo somente suas características. Assim, para cada usuário teste foram selecionados os itens que ele avaliou e que não estavam entre os 262 selecionados. Entre estes itens foram escolhidos 20% de forma estratificada em relação as notas dadas. Com esses itens foram definidos pares e sobre eles aplicadas as regras geradas pelo sistema. As regras podem acertar (acertos_cobertura), errar (erros_cobertura) ou serem indiferentes. A medida de cobertura é definida de forma semelhante à precisão: a razão entre os acertos de cobertura e a soma entre os acertos e erros de cobertura.

O primeiro teste teve como objetivo apresentar quais das alternativas de *extração de consenso* consegue melhor resultado sobre o banco BD_100. A Tabela IV apresenta os resultados obtidos. Como pode ser observado, a alternativa (1) e (2) alcançaram melhores resultados tanto na acurácia da recomendação (precisão, revocação e f1), como na cobertura. Entre as duas alternativas, por ser levemente superior, a alternativa (1) é a escolhida para os testes de comparação com o baseline CBCF apresentados na Tabela V e Tabela VI. Nestas tabelas também são apresentados as medidas de desvio padrão σ_p, σ_r e σ_c para a precisão, revocação e cobertura respectivamente.

Em comparação com o *baseline CBCF* o ***XprefRec*** apresenta resultados superiores confirmando ser esssa uma abordagem promissora para sistemas de recomendação. Observa-se também que a precisão não decresce muito à medida que a esparsidade dos dados aumenta. Quanto à cobertura, verifica-se que o ***XprefRec*** apresentou mesmo um aumento no valor da cobertura à medida que a esparsidade aumentou, o inverso acontecendo com o *baseline*.

[2] http://www.grouplens.org/

Table III. Bancos de validação

BD	| Avaliações |	Esparsidade
BD_100	67.971	12,35%
BD_90	61.143	21,16%
BD_80	54.423	29,82%
BD_70	47.464	38,80%
BD_60	40.831	47,35%
BD_50	33.344	57,00%

Table IV. Teste das alternativas de obtenção de consenso

Alternativa	Precisão	Revocação	F1	Cobertura
(1)	**76.06%**	**73.80%**	**74.91%**	**63.25%**
(2)	76.04%	73.79%	74.90%	63.17%
(3)	63.58%	60.12%	61.80%	56,92%
(4)	63.31%	60.02%	61.62%	57.83%
(5)	63.83%	60.53%	62.13%	59.05%

Table V. Resultados CBCF

BD	Precisão	σ_p	Revogação	σ_r	F1	Cobertura	σ_c
BD_100	70.64%	7.41%	70.64%	7.41%	70.64%	61.19%	5.35%
BD_90	69.59%	7.18%	69.59%	7.18%	69.59%	61.60%	5.39%
BD_80	64.88%	5.13%	64.88%	5.13%	64.88%	61.25%	4.99%
BD_70	58.46%	4.43%	58.46%	4.43%	58.46%	61.36%	4.65%
BD_60	52.07%	4.75%	52.07%	4.75%	52.07%	60.89%	4.65%
BD_50	50.71%	5.94%	50.71%	5.94%	50.71%	58.55%	4.68%

Table VI. Resultados ***XPrefRec***

BD	Precisão	σ_p	Revogação	σ_r	F1	Cobertura	σ_c
BD_100	76.06%	6.46%	73.80%	8.86%	74.91%	63.22%	4.63%
BD_90	75.38%	6.70%	71.43%	10.65%	73.35%	63.64%	4.73%
BD_80	74.01%	4.73%	65.28%	5.31%	69.37%	67.81%	4.17%
BD_70	71.80%	3.87%	60.80%	4.12%	65.84%	68.26%	4.96%
BD_60	73.94%	3.52%	60.19%	3.61%	66.36%	71.14%	4.02%
BD_50	72.98%	3.24%	59.27%	3.48%	65.41%	74.36%	4.04%

4. CONCLUSÃO

O ***XprefRec*** apresenta uma nova abordagem híbrida de sistema de recomendação trazendo como diferencial alternativas de *extração de consenso* e aplicação de um *minerador de preferências contextuais*. Os testes realizados apontam esta proposta como promissora ao obter resultados de precisão, revocação e cobertura superiores ao *baseline CBCF* utilizado. Como continuação deste trabalho propõe-se testes com outros algoritmos de clusterização bem como uma otimização do algoritmo de mineração de preferências. A otimização, já em fase de implementação, se baseia na aplicação da técnica de *Range Voting* ([Woodall 1994]) após a mineração das preferências. Tal técnica permite uma melhora sensível na revocação de algoritmos de mineração de preferências. Desta forma, acreditamos que melhorando a revocação do *Cprefminer* possamos melhorar a revocação do ***XprefRec***.

REFERENCES

ADOMAVICIUS, G. AND TUZHILIN, A. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. and Data Eng.* 17 (6): 734–749, jun, 2005.

CHICLANA, F., HERRERA, F., AND HERRERA-VIEDMA, E. Integrating multiplicative preference relations in a multipurpose decision-making model based on fuzzy preference relations. *Fuzzy Sets and Systems* 122 (2): 277–291, 2001.

COHEN, W. W., SCHAPIRE, R. E., AND SINGER, Y. Learning to order things. *J. Artif. Intell. Res. (JAIR)* vol. 10, pp. 243–270, 1999.

DE AMO, S., BUENO, M. L. P., ALVES, G., AND DA SILVA, N. F. F. Cprefminer: An algorithm for mining user contextual preferences based on bayesian networks. In *ICTAI*. pp. 114–121, 2012.

DE AMO, S., BUENO, M. L. P., ALVES, G., AND DA SILVA, N. F. F. Mining user contextual preferences. *JIDM* 4 (1): 37–46, 2013.

ESTER, M., KRIEGEL, H.-P., SANDER, J., AND XU, X. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*. pp. 226–231, 1996.

MELVILLE, P., MOONEY, R. J., AND NAGARAJAN, R. Content-boosted collaborative filtering for improved recommendations. In *Eighteenth national conference on Artificial intelligence*. American Association for Artificial Intelligence, Menlo Park, CA, USA, pp. 187–192, 2002.

RESNICK, P., IACOVOU, N., SUCHAK, M., BERGSTROM, P., AND RIEDL, J. Grouplens: An open architecture for collaborative filtering of netnews. In *CSCW*. pp. 175–186, 1994.

WOODALL, D. R. Properties of preferential election rules, 1994.

A Query-by-Similarity Indexing Strategy for Web Forms

Willian Ventura Koerich, Ronaldo dos Santos Mello

Universidade Federal de Santa Catarina, Brazil
{willian.vkoerich, ronaldo}@inf.ufsc.br

Abstract. Search engines do not provide specific searches for Web forms related to the Deep Web, in particular, similarity search. To deal with this lack, we propose a query-by-similarity system called WF-Sim, and this paper presents the indexing strategy adopted by WF-Sim for querying-by-similarity Web forms. It is centered on suitable index structures to the main kinds of queries posed on Web forms, as well as some optimizations in order to reduce the number of index entries. To evaluate the indexes performance, we ran experiments on two WF-Sim persistence strategies: file system and database. We also compare the performance of our indexes against the traditional keyword-based index, and the results were promising.

Categories and Subject Descriptors: H.2 [**Database Management**]: Miscellaneous; H.3.1 [**Content Analysis and Indexing**]: Indexing Methods; H.3.3 [**Information Search and Retrieval**]: Miscellaneous

Keywords: Deep Web, Web form, indexing, query-by-similarity

1. INTRODUCTION

The increasing volume of data available on the Deep Web [Madhavan et al. 2009] increases, in turn, the need for accessing this kind of information by users. The problem of finding, integrating and organizing Web forms that serve as entry points to hidden databases has been received substantial attention by the database community [Barbosa and Freire 2007; Barbosa et al. 2007; Fang et al. 2007; Madhavan et al. 2008; Chuang and Chang 2008; Wang et al. 2009; Hong et al. 2010; dos Santos et al. 2012]. However, there is a lack of solutions regarding query support for exploring Web form collections. Some relevant efforts on this direction are *Deque* [Shestakov et al. 2005] and *DeepPeep* [Barbosa et al. 2010], but they do not have support for similarity search. This is a real drawback, considering that Deep Web data sources in a same domain, in particular, data presented in Web form fields, usually are heterogeneous in terms of the definition of their *labels* and *values*. Some few existing work that proposes query-by-similarity over Deep Web data are limited in terms of query capabilities [Qiu et al. 2003; Dong and Halevy 2007].

These limitations had motivated the development of a more efficacy solution called *WF-Sim*. WF-Sim is a system that provides query-by-similarity for Web form data [Goncalves et al. 2011]. Searching at WF-Sim occurs over indexed form fields, which are called *elements*. Index structures are defined for data clusters generated by an element matching process, allowing the retrieval of forms with similar elements. These indexes have been properly designed to facilitate the most frequent types of queries posed on Web forms.

This paper presents the indexing-by-similarity strategy for Web form data developed for WF-Sim, aiming at accessing data hold on two types of persistence schemata supported by the project: file system and database. We evaluate not only the performance of the index structures for each type of persistence, but also which index(es) structure(s) had obtained better performance.

This paper contains more five sections. Section 2 presents the WF-Sim system. Section 3 details the *Indexing* module and the proposed index structures. Section 4 describes an experimental evaluation. Section 5 comments related work and section 6 is dedicated to the conclusion.

2. WF-SIM SYSTEM

Searching methods for Web forms usually performs the exact matching between terms specified in the query input and existing terms on the forms, like labels and values' contents. Aiming at adding query-by-similarity capabilities to Web forms, we had designed the *WF-Sim* system. It is inspired by the assumption that data available in Deep Web are relevant for users who want to find Web forms that meet your needs in terms of online services for several purposes.

WF-Sim is a query-by-similarity processing system for Web forms based on its elements. Given an input query, i.e., a set of elements called *form template*, the system returns a set of Web forms that have similar elements. The system has modules for searching, indexing and clustering. The *Clustering* module was the focus of a previous work [Goncalves et al. 2011]. It applies similarity metrics to group similar elements into clusters. The *Search* module allows the definition of a form template, the submission of the request for elements similar to the template elements (to be searched in the indexes), and the generation of the ranked list of Web forms that are similar to the form template. Details about how these modules work are out of the scope of this paper.

The *Indexing* module is the focus of this paper. It is responsible for creating and updating the index structures, as well as for providing access to them in order to retrieve the relevant forms. Given a form template, the search process for similar forms accesses a synonyms database that directs the form template elements' labels to synonyms elements called *centroids*. A centroid is the element label that represents a cluster of similar elements, as generated by the *Clustering* module. For example, an element with label *"Make"* could be the centroid of a cluster composed by elements with labels *"Make"*, *"Brand"* and *"Select a Make"*. All the centroids are indexed, and each index entry indicates the list of Web forms that have exact or similar elements. Next section details the indexing module and the proposed index structures. In fact, the proposed indexes here are an upgrade of the work of [Goncalves et al. 2011], which had defined a single index structure with a larger index entry composed by an element label and all of its values.

3. INDEXING MODULE

Three index structures have been defined to access Web form data by similarity: *keyword*, *context* and *metadata*. The notion of these structures was borrowed from a previous work [dos Santos Mello et al. 2010] and adapted to the problem of similarity search for Web forms. These structures, in particular, context and metadata indexes, are suitable to the frequent types of queries posed on Web forms.

Context queries search for Web forms based on the contextualization of their fields, i.e., they filter forms that have specific values for a given element label. *Metadata queries* retrieve Web forms that hold a specific metadata content, i.e., a content that acts as a field label or a field value.

Figure 1 shows examples of entries for the developed indexes. The keyword index has entries for any term that may appear in a Web form element, i.e., an existing value or label. The context index defines entries for each combination of a label and one of its allowed values, with a *value###label* format. The metadata index is based on the type of metadata the user is interested in, with entries in the format *term###VALUE* or *term###LABEL*. We prioritize element components whose content variation is higher, like label values in the context index, at the beginning of the index entry string in order to provide an index entry ordering more suitable to the searching. This ordering is particular relevant for inequality or range queries, which require sequential scanning of the index structure[1].

[1]We intend to provide inequality and range queries in a next version of the WF-Sim system.

The synonyms index is an auxiliary structure that allows the searching for similar terms. At label indexing time, a synonym database is queried to check if the label is a synonym of a centroid. If so, a record is inserted in the synonym index containing label name and centroid name. For future queries, if an element label in the template is not found in the indexes, but it is a synonymous of a centroid, it is possible to find out similar forms by searching for synonyms, ensuring, in this way, a similarity search. The synonyms index is important in our indexing strategy because it allows that only labels of centroid elements be indexed in the other index structures, reducing their number of entries.

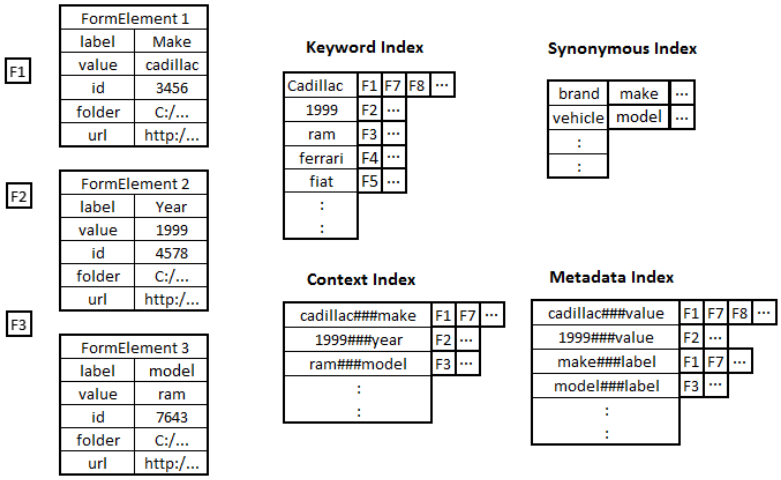


Fig. 1. Indexing structures and considered element information.

Our indexing strategy also supports data cleaning by running a parser that makes term standardization. It converts all letters to lowercase and removes undesirable variations through the process of stemming and removal of stop words. Removal of stop words reduces the size of the indexes and the stemming process reduces the amount of entries, since only the radical terms of desirable element labels are indexed. The parser is also executed during the searching process for similar Web forms. In this case, the template labels are also cleaned before their checking in the indexes.

The access to the indexes is based on the traditional boolean model of information retrieval [Baeza-Yates and Ribeiro-Neto 1999]. In order to illustrate this process, suppose a user wants to find forms that have *GM* brand and a label *year*, i.e., a form template with these two elements' features. The WF-Sim *Search* module then generates the following query: *GM###brand* `AND` *year###LABEL*. Each generated filter is then passed to the *Indexing* module. Considering the filter *GM###brand*, the *Indexing* module recognizes it as a context filter (formed by two terms: a label and a possible value for it), verifies if data cleaning is necessary and then access the index of synonyms. Assuming that the term *"brand"* has only one centroid as a synonym (*"make"*, for example), the filter is converted to "*GM###make*" and the entry corresponding to this filter is accessed in the context index. If there is more than one synonym for *"brand"*, the index entries corresponding to all the synonyms are accessed and it is made a union of the Web form URLs accessed by each entry. The same reasoning applies to the filter *year###LABEL*, and the Web form resulting set of each filter is further processed by the indexing module according to the logical operators defined in the query, being the final Web form result set returned to the Search module.

The index structures were implemented as inverted indexes [Baeza-Yates and Ribeiro-Neto 1999] through the *Lucene* tool [luc 2013]. In the context of this paper, the inverted indexes map a term, for example, a value of an element, to the Web forms containing that particular term. As shown in Figure 1, WF-Sim system stores the following element properties: labels, allowed values, Web

form URL, and a database/file system ID. The ID is the identification of the centroid that holds the element. The indexes point to the location of these elements stored in the file system or database.

File system was the first persistence strategy adopted by the WF-Sim system, being initially conceived for Deep Web domains with a reduced number of Web forms, like *Biology*. It maintains a data file for each cluster of similar elements, being the centroid ID the file name (the file system ID). A data file holds a serialized set of element records, and each record stores the element properties previously described. A relational database was further adopted to keep the element clusters. The database schema is basically composed by a *cluster*, *element* and *value* tables. The element table contains an ID (database ID), a label, a Web form URL and a reference to a cluster. The value table basically holds a description and a reference to an element. The cluster table maintains the cluster ID and the data settings used to configure the similarity metrics that generate the cluster.

Besides *Lucene*, we use *WordNet* [wor 2013] as the basis for the creation of the synonym database. It was used to determine the similarity between elements as well as to build the synonyms index.

4. EXPERIMENTAL EVALUATION

Our indexing strategy was evaluated through preliminary experiments that measure the query processing performance accessing the proposed index structures. Two versions of the indices were created, aiming at accessing Web form elements stored in the file system and database. The experiments were conducted on a computer desktop with an *Athlon II X2 2.9 GHz*, 4 GB memory, 500 GB hard drive, operating system *Windows 7 Home Basic* SP1 64-bit, *MySQL* database, and a sample with 1090 Web forms and 6157 elements.

Figure 2 shows the average runtime (in miliseconds) of the same set of query filters, specific for each kind of index, accessing file system or the database. Figure 3 shows the results of the experiments with an incremental number of query predicates in the *Auto* domain, i.e., filters over one, two and five elements in order to access the context index. We focus on context index because it maintains the query filters most commonly used in Web form querying. In all tests, we ran each query 10 times and got the average processing time.

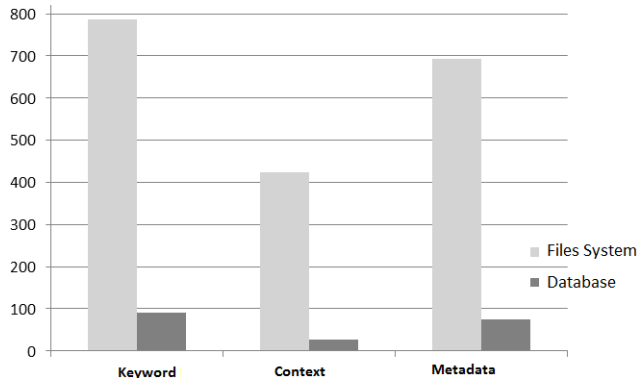


Fig. 2. Average processing time of queries that access the index structures for data in file system and database.

Figure 2 highlights a large difference between file system and database performance. In fact, the execution of the queries spent much more time (8 times slower on average) to access indexed data in the file system. This behavior is due to the fact that file systems do not provide efficient access optimization strategies like database management systems do.

An important contribution of this work, in the context of queries over Web forms, is the reduced processing time for queries accessing context and metadata indexes if compared to the performance

using traditional keyword index, as also shown in Figure 2. This better performance is justified by the fact that, during indexes creation, data about elements are analyzed and filters by context and metadata are automatically defined and stored in the corresponding indices. These two structures are very suitable for further queries over Web forms since they ensure a simplified mapping from a template to these indexes'entries.

With respect to the tests shown in Figure 3, we note that both database and file system query processing time increase were proportional to the increase in the number of conjunctive filters. However, the processing time increase was smaller for the file system, including a processing time decrease from 2 to 5 query filters. This situation needs to be better evaluated through new experiments over larger volumes of data. Even so, processing time difference between file system and database had remained very high, being database access 10 to 30 times faster than file system access for conjunctive filters. Also, we have only a linear increase in database processing time with respect to the number of filters, demonstrating a good index performance.

CONTEXT	# WF	Files system Time	Database Time
jeep###make	355	381 ms	11 ms
corolla###model	155	118 ms	10 ms
jeep###make AND corolla###model	86	486 ms	19 ms
jeep###make OR corolla###model	424	484 ms	21 ms
jeep###make AND NOT corolla###model	269	484 ms	20 ms
acura###make AND audi###make AND chevrolet###make AND 1000###price AND maverick###vehicle"	0	459 ms	36 ms
acura###make OR audi###make OR bentley###make OR 1964###year OR 1999###year	464	490 ms	49 ms
acura###make OR audi###make AND bentley###make OR 1964###year AND NOT 1999###year	355	488 ms	53 ms

Fig. 3. Performance of a set of query predicates that requires access to the context index.

5. RELATED WORK

As far as we know, indexing solutions for Web form query-by-similarity is a research topic not so much addressed in the literature. The closest related work is the approach of [Qiu et al. 2003], which also extracts and stores hidden database information in a relational database. The database maintains a table for each Web form field that holds its values and pointers to clusters of similar values. We see several limitations in comparison to our proposal: *(i)* A great number of indexes must be created if we want to index values of several fields; *(ii)* They assume that a schema matching for Web forms in a same domain was previously accomplished to generate a relational global schema using a mediation system. Such a support increases the complexity of the approach; *(iii)* They are able to answer only queries over field values. Our approach has a broader scope, being able to query field labels by similarity (and their values), as well as terms that acts as a specific metadata for a given Web form.

Other relevant approach is the work of [Dong and Halevy 2007], which deals with the more general problem of indexing dataspaces and supports only field-value (context) queries. Our approach gives additional index support to query form metadata.

6. CONCLUSION

This paper presents and evaluates an indexing strategy for supporting Web form query-by-similarity in the context of the WF-Sim system. This strategy considers the indexing of element data on suitable index structures called metadata and context, besides the traditional keyword indexing. The first two structures provide access optimizations since they index, in a straightforward way, the more frequent query filters over Web forms. If a keyword index would be used for processing a context search (a

"label = value" predicate), for example, we would have to get the Web form result set that contain the term corresponding to the label, then the Web form result set containing the desired value, and compute an intersection of the two result sets. This overhead is unnecessary with the introduction of the context index. The same reasoning holds for metadata search and the metadata index.

Few related work in the literature define indexing structures for similarity search over Web forms, and the existing solutions have limitations in terms of query capabilities and/or seems to be not so effective. Our experimental evaluation has shown the good performance of our specific index structures for Web form data, being the main contribution of this paper. Even so, we intend to compare our solution with related work in terms of processing time, if we have access to their source codes.

Besides that, we intend to evaluate performance for a larger sample of Web forms. Through a partnership with New York University, a sample of approximately 40,000 forms in several domains will be soon available for new experiments. Another future work is to consider searching for Web forms that hold dependencies between fields, such as a field *"Make"* whose values determine the values of a field *"Model"*, assuming an *Auto* domain. The intention is to consider filters that test the existence of such dependencies and define indexing structures that facilitate queries-by-similarity. This kind of query is relevant if the user wants to access Web forms that have a chain of dependencies between fields.

REFERENCES

Apache Lucene. http://lucene.apache.org/core/, 2013.

Wordnet. http://wordnet.princeton.edu/, 2013.

Baeza-Yates, R. A. and Ribeiro-Neto, B. A. *Modern Information Retrieval*. ACM Press / Addison-Wesley, 1999.

Barbosa, L. and Freire, J. An adaptive crawler for locating hidden web entry points. In *Proceedings of the International World Wide Web Conferences*. Banff, Canada, pp. 441–450, 2007.

Barbosa, L., Freire, J., and da Silva, A. S. Organizing hidden-web databases by clustering visible web documents. In *Proceedings of the IEEE International Conference on Data Engineering*. Istambul, Turkey, pp. 326–335, 2007.

Barbosa, L., Nguyen, H., Nguyen, T. H., Pinnamaneni, R., and Freire, J. Creating and exploring web form repositories. In *Proceedings of the ACM SIGMOD International Conference on Management of Data Conference*. Indianapolis, IN, USA, pp. 1175–1178, 2010.

Chuang, S.-L. and Chang, K. C.-C. Integrating web query results: Holistic schema matching. In *Proceedings of the International Conference on Information and Knowledge Engineering*. Napa Valley, CA, USA, pp. 33–42, 2008.

Dong, X. and Halevy, A. Y. Indexing dataspaces. In *SIGMOD Conference*. Beijing, China, pp. 43–54, 2007.

dos Santos, L. B., Dorneles, C. F., and dos Santos Mello, R. An approach for extracting web form labels based on distance analysis of html components. In *Proceedings of IADIS International Conference WWW/Internet*. Madrid, Spain, 2012.

dos Santos Mello, R., Pinnamaneni, R., and Freire, J. Indexing web form constraints. *Journal of Information and Data Management* 1 (3): 343–358, 2010.

Fang, W., Cui, Z., and Zhao, P. Ontology-based focused crawling of deep web sources. In *Proceedings of International Conference on Knowledge Science, Engineering and Management*. Melbourne, Australia, pp. 514–519, 2007.

Goncalves, R., Dagostini, C. S., Silva, F. R., Dorneles, C. F., and dos Santos Mello, R. A similarity search method for web forms. In *Proceedings of IADIS International Conference WWW/Internet*. Rio de Janeiro, RJ, Brazil, 2011.

Hong, J., He, Z., and Bell, D. A. An evidential approach to query interface matching on the deep web. *Information Systems* 35 (2): 140–148, 2010.

Madhavan, J., Afanasiev, L., Antova, L., and Halevy, A. Y. Harnessing the deep web: Present and future. In *Proceedings of International Biennial Conference on Innovative Data Systems Research*. Asilomar, CA, USA, 2009.

Madhavan, J., Ko, D., Kot, L., Ganapathy, V., Rasmussen, A., and Halevy, A. Y. Google's deep web crawl. *Proceedings of the VLDB Endowment* 1 (2): 1241–1252, 2008.

Qiu, J., Shao, F., Zatsman, M., and Shanmugasundaram, J. Index structures for querying the deep web. In *Proceedings of the Workshop on Web and Databases*. San Diego, CA, USA, pp. 79–86, 2003.

Shestakov, D., Bhowmick, S. S., and Lim, E.-P. Deque: Querying the deep web. *Data & Knowledge Engineeering* 52 (3): 273–311, 2005.

Wang, Y., Lu, J., and Chen, J. Crawling deep web using a new set covering algorithm. In *Proceedings of International Conference on Advanced Data Mining and Applications*. Beijing, China, pp. 326–337, 2009.

SGProv: Mecanismo de Sumarização para Múltiplos Grafos de Proveniência[1]

Daniele El-Jaick, Marta Mattoso e Alexandre A. B. Lima

Universidade Federal do Rio de Janeiro, Brazil
{deljaick, marta, assis}@cos.ufrj.br

Resumo. Os Sistemas de Gerência de Workflows Científicos (SGWfC) têm o objetivo de automatizar a construção e execução de experimentos científicos. Várias execuções de workflows são necessárias para realizar um experimento. O rastro de proveniência, coletado pelos SGWfC durante estas execuções, é importante para que os cientistas possam compreender, reproduzir e analisar seus experimentos. Um rastro de proveniência contém o histórico da derivação dos dados, assim, pode ser representado sob a forma de um grafo direcionado e acíclico. Cada execução de um workflow gera um grafo de proveniência. Após várias execuções, por exemplo, explorando parâmetros, inúmeros grafos são gerados. A base de proveniência, portanto, requer um espaço de armazenamento considerável e consultá-la envolve a manipulação de um grande volume de grafos. Consultas típicas de proveniência percorrem os diversos grafos para obter o caminho de derivação (linhagem) dos dados da consulta. Este trabalho propõe um mecanismo de sumarização para grafos de proveniência (SGProv), usando um banco de dados de grafos para armazenar e consultar esses grafos. O objetivo é gerar um único grafo sumário que represente todos os grafos de proveniência gerados durante um experimento, mas com tamanho reduzido e eliminando dados repetidos. Esta abordagem de sumarização visa reduzir o tempo de processamento de consultas de proveniência utilizando apenas o grafo sumário para respondê-las sem precisar reconstruir os grafos originais. Consultas típicas sobre dados de proveniência de execução de workflows mostraram o potencial da nossa solução.

Categories and Subject Descriptors: [H. **Information Systems**]: [H.2. **Database Management**]

1. INTRODUÇÃO

Os experimentos científicos assistidos por computadores são baseados em simulações realizadas pelo encadeamento de programas. Este encadeamento é comumente representado por meio de workflows científicos, que podem ser definidos como uma especificação formal dos passos executados em um experimento científico. A natureza exploratória dos experimentos científicos faz com que um mesmo workflow seja executado sob diversas concepções. Um cientista pode precisar analisar os resultados da execução com diferentes combinações de parâmetros ou com a substituição de programas. Assim, para um único experimento, inúmeras execuções de workflows são realizadas [Mattoso *et al.* 2010].

Cada execução de um workflow gera um rastro de proveniência com a descrição de todas as transformações de dados que ocorreram durante a execução: dados de entrada e resultados intermediários e finais. Os rastros de proveniência são baseados em objetos (dados e programas) e seus relacionamentos (dependências) e são representados na forma de um grafo direcionado acíclico (*directed acyclic graph* – DAG) [Aggarwal e Wang 2010]. Independente da complexidade do workflow, o grafo de proveniência resultante será um DAG. A presença de execuções condicionais e paralelismo, por exemplo, afeta a complexidade do workflow, mas não do DAG. Uma base de dados

[1] Esse artigo foi parcialmente financiado pelo CNPq, Capes e Faperj.

de proveniência é, portanto, composta por vários DAG cujos nós possuem estruturas variáveis e que, muitas vezes, não são previamente conhecidas. Os nós possuem também muitas dependências entre si. Estas características fazem com que o armazenamento e a consulta aos dados de proveniência ainda sejam desafios significativos [Anand *et al.* 2010]. Consultas típicas de proveniência incluem obter a "linhagem" (caminho de derivação) de resultados (finais ou intermediários) de uma execução do workflow. Executá-las eficientemente (curto tempo de resposta) e obter um grafo, ou um conjunto de grafos, como resposta são desafios [Woodman *et al.* 2011].

A abordagem mais usada na literatura para tratar o problema de armazenamento e consulta de dados de proveniência é utilizar um banco de dados relacional como em [Huahai e Singh 2008] e [Anand *et al.* 2012]. O principal problema desta abordagem é a complexidade da especificação e o tempo de processamento das consultas que, na maioria das vezes, envolvem muitas junções entre relações, especialmente em grafos muito volumosos e com muitas arestas entre seus nós. O esquema (descrição da base de dados) rígido do modelo relacional também representa um problema para a sua utilização, uma vez que os dados de proveniência, geralmente, possuem esquema flexível [Davidson e Freire 2008].

Este artigo analisa o uso de um banco de dados de grafos (BDG) para armazenar e consultar grafos de proveniência, pois seu modelo de dados é adequado para as características de consultas desta aplicação. Este modelo é composto por nós, arestas e atributos, que podem ser associados tanto aos nós quanto às arestas. As arestas possuem tipos, sendo possível a existência de várias arestas de tipos diferentes entre quaisquer pares de nós. Não existe esquema rígido que defina previamente os atributos relativos a cada nó ou aresta, tornando o armazenamento de dados bastante flexível. Os BDG utilizam operações clássicas da teoria dos grafos para processar os grafos armazenados, tais como travessias, busca em largura e em profundidade e determinação do menor caminho entre dois nós [Angles e Gutierrez 2008]. Desta forma, usar um BDG para percorrer e recuperar dados de vários grafos de proveniência torna as consultas mais simples e eficientes em relação a um banco de dados relacional, pois não envolve o uso de junções entre relações.

Mesmo com a utilização de um BDG, persiste o problema da manipulação do grande volume de dados dos grafos de proveniência, que pode comprometer o tempo de processamento das consultas. Uma abordagem para tratar grafos volumosos é a sumarização [Yu *et al.* 2010]. [Navlakha *et al.* 2008] propõem um grafo sumário gerado através da sumarização de áreas densas do grafo, de subgrafos completos e bipartidos e de cliques. [Yu *et al.* 2010] propõem um grafo sumário onde: nós de um super-nó devem ter os mesmos atributos com os mesmos valores; uma super-aresta é criada se cada nó de um super-nó tiver pelo menos uma aresta com um nó de outro super-nó. [Liu e Yu 2011] propõem um grafo sumário onde: nós de um super-nó devem possuir atributos em comum, com os mesmos valores (ou valores similares) e o mesmo número (ou número similar) de arestas para super-nós vizinhos. Entretanto, as soluções existentes se dedicam a sumarizar um único grafo que tenha um grande volume de nós, enquanto as consultas de proveniência são realizadas sobre um grande volume de grafos, não necessariamente com muitos nós.

Este artigo propõe um mecanismo de sumarização para múltiplos grafos de proveniência (SGProv) resultantes de execuções de workflows durante um experimento científico computacional. Seu objetivo é gerar um único grafo sumário que represente todos os grafos gerados durante um experimento, de tal modo que caminhos e nós repetidos em mais de um grafo sejam armazenados uma única vez e suas diferenças sejam evidenciadas. O volume de dados do grafo sumário é reduzido em relação ao conjunto de grafos original e as consultas podem ser aplicadas somente a ele, melhorando o tempo de processamento. Para avaliar o SGProv, foram analisados os tempos de processamento de consultas típicas de proveniência aplicadas somente ao grafo sumário e aos grafos da base original.

Este artigo está organizado da seguinte forma: a seção 2 descreve o mecanismo de sumarização proposto, a seção 3 apresenta os resultados obtidos com o SGProv e a seção 4 apresenta as conclusões.

2. SGPROV

Esta seção descreve o modelo de dados e como o grafo sumário é gerado a partir dos grafos de proveniência.

2.1 Modelo de Dados

Inicialmente, é definido o conceito de grafo adotado pelo SGProv.

Definição 1 (Grafo): Um grafo é definido como G=(V, E, A, T), onde V={v_1, v_2,..., v_n} é seu conjunto de nós, E={e_1, e_2,..., e_m} ⊂ (VxV) é seu conjunto de arestas direcionadas, A={a_1, a_2,..., a_p} é o conjunto de atributos associados aos nós ou arestas e T={t_1, t_2,..., t_q} é o conjunto de tipos de arestas. Cada nó $v_i \in V$ tem pelo menos um atributo $a_x \in A$. O valor de a_x no nó v_i é representado por Val(v_i, a_x). Cada aresta $e_j \in E$ possui um único tipo t_k definido por Tipo(e_j), onde $t_k \in T$. O valor do atributo a_y pertencente a uma aresta e_j é representado por Val(e_j, a_y). Como as arestas são direcionadas, Origem(e_j) e Destino(e_j) representam, respectivamente, seus nós de origem e de destino.

Seguindo o modelo apresentado, é definido o conceito de grafo de proveniência.

Definição 2 (Grafo de proveniência): Um grafo de proveniência é definido como G_p=(V_p, E_p, A_p, T_p), onde os nós de V_p representam programas ou dados e as arestas de E_p representam a "linhagem" dos nós. Um atributo "tipo" $\in A_p$ é obrigatório para todos os nós e arestas. Se um nó v_{pi} representa um programa, Val(v_{pi}, tipo) = "programa". Se este nó representa um dado, Val(v_{pi}, tipo) = "dado". O conjunto A_p contém também os atributos encontrados na coleta de proveniência, como, por exemplo, os nomes dos parâmetros de entrada dos programas. Nós que representam programas possuem ainda um atributo "nome" $\in A_p$. O conjunto T_p representa tipos de arestas.

2.2 Grafo Sumário

O SGProv baseia-se no fato de que nós que representam um mesmo programa tendem a ocorrer com frequência em diferentes grafos de proveniência, pois, geralmente, um programa é executado repetidas vezes com diferentes combinações de parâmetros. Os nós que representam os dados apresentam grande variação nestes grafos, pois correspondem às entradas e saídas dos programas. Com base nestas características, o SGProv agrupa os nós e as arestas do conjunto de grafos de proveniência para gerar o grafo sumário, que contém todos os nós e arestas dos grafos originais, mas sem redundâncias. Para processar os grafos armazenados, o SGProv utiliza a busca em largura, que visita os nós do grafo nível a nível, ou seja, todos os nós a uma distância *k* do nó inicial são visitados antes de qualquer nó a uma distância *k+1* do inicial [Haichuan e Kitsuregawa 2013].

Definição 3 (Grafo Sumário): Um grafo sumário é definido por G_S=(V_S, E_S, A_S, T_S). Cada nó $v_s \in V_S$ é chamado de super-nó e cada aresta $e_s \in E_S$ é chamada de super-aresta.

Definição 4 (Super-nó): Um super-nó v_{si}={v_{p1}, v_{p2},..., v_{pn}} ⊂ (V_{p1} U V_{p2} U...U V_{px}), onde Val(v_{p1}, tipo) = Val(v_{p2}, tipo)=...= Val(v_{pn}, tipo) = "programa" e Val(v_{p1}, nome) = Val(v_{p2}, nome) =...= Val(v_{pn}, nome).

Definição 5 (Super-aresta): Uma super-aresta liga dois super-nós se existir pelo menos uma aresta de um nó de um super-nó para um nó do outro super-nó. Uma super-aresta e_{si}={e_{p1}, e_{p2},..., e_{pm}}⊂ (E_{p1} U E_{p2} U...U E_{py}), onde {Origem(e_{p1}), Origem(e_{p2}),..., Origem(e_{pm})} ⊂ Origem(e_{si}), {Destino(e_{p1}), Destino(e_{p2}),..., Destino(e_{pm})} ⊂ Destino(e_{si}) e Tipo(e_{p1}) = Tipo(e_{p2}) =...= Tipo(e_{pm}).

Os nós e as arestas presentes em um único grafo de proveniência são inseridos diretamente no sumário, fora de qualquer super-nó ou super-aresta. Isto ocorre quando um programa é usado em uma

única execução do workflow e, portanto, pertence a um único grafo G_p. Alguns super-nós e super-arestas do grafo sumário possuem um atributo "exec", que consiste em uma lista de identificadores das execuções dos workflows de que fizeram parte. Este atributo torna possível recuperar os caminhos dos grafos originais e responder as consultas de proveniência com base apenas no grafo sumário. Quando todos os super-nós e super-arestas do grafo sumário possuem um mesmo valor no seu atributo "exec" significa que fizeram parte de uma mesma execução do workflow. A ausência do atributo "exec" em um super-nó (super-aresta) indica que ele representa um programa (dependência) presente em todos os grafos de proveniência.

Na figura 1 é apresentado um exemplo do mecanismo proposto. A partir dos grafos de proveniência G_{p1} e G_{p2} é gerado o grafo sumário G_s, onde: o super-nó v_{s1} é criado, pois Val(v_{10}, tipo) = Val(v_{20}, tipo) = "programa" e Val(v_{10}, nome) = Val(v_{20}, nome) = "P2"; o super-nó v_{s2} é criado, pois Val(v_{11}, tipo) = Val(v_{21}, tipo) = "programa" e Val(v_{11}, nome) = Val(v_{21}, nome) = "P3"; e a super-aresta e_{s12} é criada, pois existem arestas (e_1, e_2) entre nós de v_{s1} e v_{s2}, onde {Origem(e_1), Origem(e_2)} ⊂ Origem(e_{s12}), {Destino(e_1), Destino(e_2)} ⊂ Destino(e_{12}) e Tipo(e_1) = Tipo(e_2).

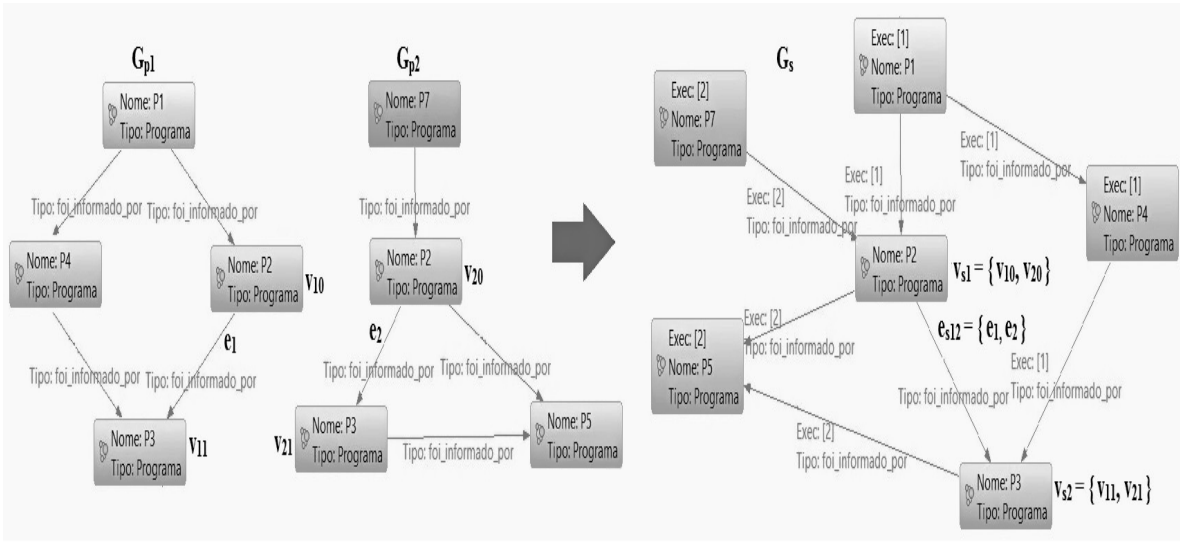


Figura 1. Exemplo do mecanismo de sumarização.

3. AVALIAÇÃO EXPERIMENTAL

O SGProv foi implementado sobre o BDG Neo4J[2] e para as consultas foi utilizada sua linguagem Cypher[3]. Os experimentos foram realizados em uma máquina com processador Intel Core i7, com 2.4 GHz e 8GB de RAM. Para realizar uma avaliação inicial da contribuição do SGProv, foi criada uma base de testes com 1000 grafos de proveniência gerados a partir de execuções de workflows científicos sintéticos. Alguns grafos possuem estruturas similares, mas não iguais, e outros possuem estruturas bem distintas entre si. Eles estão no formato Prov-DM[4] modelo proposto pela W3C[5] (Consórcio *World Wide Web*) como padrão para proveniência. Na base de testes, os nós dos grafos representam programas utilizados por um cientista no seu experimento, onde um programa pode pertencer a um ou mais grafos. Foram definidos 30 programas distintos para workflows sintéticos que

[2] http://www.neo4j.org/
[3] http://www.neo4j.org/learn/cypher
[4] http://www.w3c.org/TR/2012/WD-prov-dm-20120202
[5] http://www.w3.org/TR/prov-aq/

possuem, em média, 15 programas e, em cada execução, ocorre a substituição de programas por outros similares ou a inclusão de novos programas. A tabela 1 mostra características da base de testes, que foi armazenada em um banco de dados, e do grafo sumário produzido pelo SGProv, que foi armazenado em outro banco de dados. Os bancos de dados são gerenciados pelo Neo4J. Como foram definidos 30 programas que se repetem nos grafos de proveniência, o grafo sumário é composto por 30 nós. Isso porque o mecanismo de sumarização agrupa os nós do grafo pela igualdade entre os nomes dos programas. Como o fluxo de dados entre os programas (arestas) pode se repetir nos grafos gerados, o mecanismo de sumarização agrupa as arestas que possuem o mesmo programa de origem e de destino produzindo um número reduzido de arestas no sumário.

Tabela 1. Características da base de testes e do grafo sumário produzido após a aplicação do SGProv.

	Base de Testes	Grafo Sumário
Número de nós	8490	30
Número de arestas	10.149	305

No contexto de experimentos científicos, os cientistas precisam consultar a base de proveniência para obter informações como: a relação entre a produção e o consumo de conjuntos de dados pelos processos, histórico da derivação dos dados e resultados intermediários obtidos. Assim, é possível interpretar, avaliar e validar seus experimentos e detectar possíveis erros [Gadelha e Mattoso 2011].

Foram executadas consultas típicas de proveniência, definidas em [Woodman *et al.* 2011], sobre a base de testes e sobre o grafo sumário para avaliar a variação dos tempos de processamento nos dois casos. As consultas foram realizadas no Neo4J e foram repetidas 100 vezes. A tabela 2 mostra a média dos tempos obtidos no processamento de cada consulta.

Tabela 2. Tempos médios de processamento das consultas de proveniência.

Consultas	Base de Teste	Sumário
C1. Consultar todos os programas do grafo e seus descendentes.	110 ms	15 ms
C2. Consultar os descendentes diretos e os descendentes dos descendentes (descendentes não diretos) do nó que representa o programa cujo atributo “nome” = P5.	93 ms	2 ms
C3. Consultar o menor caminho entre os nós que representam os programas cujo atributo “nome” = P1 e cujo atributo “nome” = P4 e procurar dependências entre os programas cujo atributo “nome” = P4 e cujo atributo “nome” = P5.	-	1 ms
C4. Consultar todos os programas que produziram dados de entrada para o programa cujo atributo “nome” = P10 (consulta aos ascendentes).	35 ms	1 ms

A redução obtida com as consultas realizadas com base no grafo sumário evidenciou o potencial da sumarização. A consulta C3 aplicada à base de testes ultrapassou o tempo limite de processamento. Isso pode ser explicado pelo grande volume de grafos na base de testes e grande quantidade de nós que representam os programas P1, P4 e P5. Uma forma de minimizar este problema seria efetuar a consulta individualmente para cada grafo da base de testes. Entretanto, o tempo de processamento de cada consulta a um grafo da base de testes foi, em média, 5000 ms.

4. CONCLUSÕES

O mecanismo de sumarização reduziu consideravelmente o volume de grafos da base de testes no exemplo analisado. No exemplo, quanto mais os programas e suas dependências com outros programas se repetem nos grafos, melhor é o resultado obtido na sumarização. O custo extra corresponde à introdução do atributo ”exec”, que será medido em trabalhos futuros. Métricas para avaliar a qualidade do sumário e sua prova de correção estão em desenvolvimento. Uma forma de avaliar a qualidade é analisar a taxa de participação dos nós dos super-nós de origem e de destino em cada super-aresta. Taxa de participação maior do que 50% indica uma super-aresta forte, caso

contrário é considerada fraca. O sumário ideal deve possuir o maior número possível de super-arestas fortes [Yu *et al.* 2010]. Uma forma de verificar a correção seria reconstruir os grafos originais a partir do sumário e analisar os resultados das consultas de proveniência aplicadas à base de testes e ao grafo sumário para identificar se estas retornam os mesmos valores sem perdas nem repetições.

Os resultados obtidos nos testes mostraram uma grande redução no tempo de processamento das consultas de proveniência quando aplicadas ao grafo sumário. Por outro lado, as consultas aplicadas aos grafos da base de testes tiveram um desempenho bem menos eficiente. Entretanto, se os grafos da base de testes forem muito distintos entre si, em relação aos programas e suas dependências, o mecanismo de sumarização pode produzir um grafo sumário tão volumoso quanto os da base de testes comprometendo o tempo de processamento das consultas. Neste caso, é preciso avaliar uma forma de fazer a sumarização com base em outras características de nós e arestas dos grafos. Uma forma pode ser agrupar nós que possuem um mesmo conjunto de nós vizinhos [Liu e Yu 2011]. No entanto, a característica principal de grafos de proveniência é a grande similaridade entre eles. Há quase sempre uma grande interseção entre os programas de simulação que fazem parte da modelagem dos workflows explorados nos experimentos científicos.

Os Neo4J apresentou bom desempenho para armazenamento, recuperação e consulta dos grafos da base de dados. Entretanto, uma análise mais ampla sobre a utilização dos BDG para gerenciar grafos de proveniência requer testes com outros sistemas de gerência de banco de dados orientados a grafos.

O mecanismo de sumarização considerou programas repetidos para agrupar os nós e arestas. Entretanto, os grafos de proveniência no formato PROV-DM também possuem outros tipos de nós, como agentes e entidades (dados), e mais quatro tipos de arestas que precisam ser avaliados. Além disso, os programas podem ter vários atributos com valores distintos. Estas questões serão abordadas em trabalhos futuros.

REFERÊNCIAS

AGGARWAL, C., HAIXUN, W. *Managing and Mining Graph Data*. Springer, 2010.

ANAND, M., SHAWN B., LUDASCHER, B. Provenance Browser: Displaying and Querying Scientific Workflow Provenance Graphs. In *Proceedings of the 26th IEEE International Conference on Data Engineering*. Long Beach, USA, pp. 1201-1204, 2010.

ANAND, M., SHAWN B., LUDASCHER, B. Database Support for Exploring Scientific Workflow Provenance Graphs. In *Proceedings of the 24th International Conference on Scientific and Statistical Database Management*. Crete, Greece, pp. 343-360, 2012.

ANGLES, R., GUTIERREZ, C. Survey of Graph Database Models. Journal ACM Computing Surveys, volume 40, issue 1, article 1, 2008.

DAVIDSON, S., FREIRE, J. Provenance and Scientific Workflows: Challenges and Opportunities. In *Proceedings of the of the ACM SIGMOD International Conference on Management of Data*. New York, USA, pp. 1345-1350, 2008.

GADELHA, L., MATTOSO, M. Provenance Query Patterns for Many-Task Scientific Computing. In *Proceedings of the* 3rd *Usenix Workshop on the Theory and Practice of Provenance*. Greece, 351-370, 2011.

HAICHUAN, S., KITSUREGAWA, M. Efficient Breadth-First Search on Large Graphs with Skewed Degree Distributions. In *Proceedings of the 16th International Conference on Extending Database Technology*. Italy, pp. 311–322, 2013.

HUAHAI, H., SINGH, A. Graphs-at-a-time: Query Language and Access Methods for Graph Databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*. New York, USA, pp. 405-418 , 2008.

LIU, Z., YU, J. On Summarizing Graph Homogeneously. In *Proceedings of the Database Systems for Advanced Applications*. Hong Kong, China, pp. 299-310, 2011.

MATTOSO, M., WERNER, C., TRAVASSOS, G., BRAGANHOLO, V., MURTA, L., OGASAWARA, E., OLIVEIRA, F., CRUZ, S. Towards Supporting Large Scale in Silico Experiments Life Cycle. International Journal of Business Process Integration and Management, v. 5(1), pp. 79-92, 2010.

NAVLAKHA, S., RASTOGI, R., SHRIVASTAVA, N. Graph Summarization with Bounded Error. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*. Vancouver, Canada, pp. 419-432 2008.

WOODMAN, S., HIDEN, H., WATSON, P. Achieving Reproducibility by Combining Provenance with Service and Workflow Versioning." In *Proceedings of the 6th Workshop on Workflows in Support of Large-scale Science*. New York, USA, pp. 127–136, 2011.

YU, P., HAN J., FALOUTSOS, C., TIAN, Y., PATEL, J. *Link Mining: Models, Algorithms, and Applications*. Springer. 2010.

OPIS: Um Método para a Identificação e a Busca de Páginas-Objeto

Miriam Pizzatto Colpo[1], Edimar Manica[1,2], Renata Galante[1]

[1] Universidade Federal do Rio Grande do Sul, Brasil
[2] Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul - Campus Ibirubá, Brasil
{mpcolpo, edimar.manica, galante}@inf.ufrgs.br

Abstract. Este artigo propõe um novo método, denominado OPIS, para a identificação e a busca de páginas-objeto, que são páginas que representam um único objeto do mundo real na web. A motivação para este trabalho se encontra no fato de que os motores de busca convencionais não conseguem responder a buscas por páginas-objeto de forma satisfatória atualmente, já que a quantidade de páginas-objeto recuperada é bastante limitada. OPIS caracteriza-se por adotar técnicas de pré-processamento de texto e de aprendizagem de máquina na classificação de páginas. Quando integrado a um motor de busca convencional, ele permite que somente páginas classificadas como páginas-objeto sejam recuperadas pelas consultas do usuário, ao invés de todas as páginas que contêm os termos da consulta. Experimentos preliminares mostram que o OPIS melhora em média 56% da precisão dos resultados de busca por páginas-objeto, quando comparado a um motor de busca convencional.

Categories and Subject Descriptors: H.2 [**Database Management**]: Miscellaneous; H.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; I.7 [**Document and Text Processing**]: Miscellaneous

Keywords: busca-objeto, classificação de páginas web, páginas-objeto

1. INTRODUÇÃO

`Objetos da web` são unidades de dados sobre as quais informações da web são coletadas, indexadas e ordenadas. Esses objetos são conceitos usualmente reconhecidos (como autores, artigos, conferências ou revistas), relevantes a um domínio de aplicação e que podem ser representados por um conjunto de atributos, os quais dependem do domínio do objeto [Nie et al. 2007]. `Páginas-objeto` são páginas que representam exatamente um objeto inerente na web. Isso significa que páginas que listam diversos objetos não são consideradas páginas-objeto por não representarem um objeto em particular. A busca por páginas-objeto é feita através de consultas restringidas por atributos de domínio e pode ser chamada de `busca-objeto` [Pham et al. 2010]. Uma consulta desse tipo é "*professor de banco de dados da UFRGS*", que restringe a área e a instituição de atuação de um objeto professor e tem como objetivo recuperar páginas que descrevam esse objeto.

Os motores de busca convencionais da web (do inglês, *General Search Engines* – GSEs) são programas que visam recuperar informações da web e apresentá-las, de forma organizada e eficiente, aos usuários. Basicamente, um GSE recebe um conjunto de palavras-chave e, analisando apenas o texto não estruturado, gera uma lista de páginas que contenham essas palavras [Baeza-Yates and Ribeiro-Neto 1999]. Embora os GSEs consigam atender à maioria das consultas realizadas atualmente, eles se mostram inadequados para recuperar páginas-objeto [Pham et al. 2010]. Os resultados esperados para a busca-objeto apresentada anteriormente, por exemplo, são páginas institucionais, pessoais ou de currículos, etc. que descrevam um objeto professor e considerem as restrições (de instituição e área

Este trabalho é parcialmente financiado pelo Instituto Nacional de Pesquisa da Web, pelo CNPq e pela CAPES.

de atuação) estabelecidas. Porém, na realidade, muitas páginas relacionadas a notícias, projetos e concursos para professores serão retornadas, dificultando para o usuário a localização de informações de seu interesse.

Este artigo propõe um novo método para a identificação e a busca de páginas-objetos, denominado OPIS (acrônimo para ***O**bject **P**age **I**dentifying and **S**earching*), que adota técnicas de pré-processamento de texto e de aprendizagem de máquina aplicadas na classificação baseada em conteúdo de páginas web. O OPIS envolve a integração de um classificador a um motor de busca, de modo a permitir que somente páginas identificadas (classificadas) como páginas-objeto sejam indexadas e posteriormente recuperadas pelas consultas dos usuários. A principal contribuição deste método é a melhoria na precisão de buscas-objeto, fornecendo aos usuários finais resultados que melhor atendam a suas necessidades de informação. Experimentos preliminares no domínio real de pesquisadores mostram que o OPIS superou o motor de busca convencional *Lucene*[1] com aumentos de precisão de 65% (0.600 vs. 0.364) nos primeiros cinco resultados e de 56% (0.453 vs. 0.289) nos primeiros 20.

O restante desse artigo está organizado da seguinte forma. Na Seção 2, são apresentados trabalhos relacionados. Na Seção 3, o OPIS é detalhadamente especificado. Na Seção 4, é apresentada a implementação e a experimentação de um caso de estudo no domínio de pesquisadores e, na Seção 5, o artigo é concluído e direções futuras são apontadas.

2. TRABALHOS RELACIONADOS

Muitos esforços têm sido feitos para melhorar os resultados recuperados pelos GSEs. A maioria deles propõe a criação de motores de busca verticais [Ji et al. 2009][Lee et al. 2011][Luo 2009], que são motores de busca específicos a determinados domínios, levando em consideração as particularidades de cada tópico. Além disso, outros trabalhos [Bennett et al. 2010][Geng et al. 2009][Pham et al. 2010] usam funções de *ranking* específicas para recuperar páginas relacionadas a domínios específicos. Em geral, esses trabalhos usam técnicas de processamento de texto e aprendizagem de máquina para extrair o conteúdo das páginas e, com base nisso: aprender um determinado domínio e filtrar apenas páginas consideradas pertencentes a esse domínio no processo de coleta; ou determinar as características do domínio a serem usadas em uma função de *ranking* específica. O OPIS se difere desses trabalhos à medida que não deseja aprender apenas o tópico das páginas, mas também seu tipo funcional (se a página é ou não uma página-objeto).

Blanco [Blanco et al. 2008] propõe um método para coletar automaticamente páginas da web que publicam dados relacionados à instancias de entidades conceituais com um esquema implícito. Esse método assume que o usuário fornece exemplos de páginas de entidades a partir de sites distintos e percorre cada um desses sites procurando por páginas que apresentam *templates* e caminhos similares aos respectivos exemplos. Esse trabalho difere do OPIS por focar na coleta de páginas de entidades com *templates* similares, enquanto o OPIS busca identificar páginas-objeto, sem considerar *templates* específicos, para melhorar a busca-objeto. Também com relação à coleta de páginas, Assis [Assis 2008] propõe um coletor focado para tópicos de interesse que possam ser representados por características de gênero e de conteúdo. Quando o usuário deseja buscar por páginas de planos de ensino de disciplinas de banco de dados, por exemplo, um conjunto de características (termos) que descreva o gênero (planos de ensino) e outro que descreva o conteúdo (banco de dados) devem ser informados por um usuário especializado, de modo que o coletor possa analisar cada página através da sua similaridade com os termos de ambos os aspectos. No OPIS, o conteúdo e o gênero das páginas não são considerados separadamente, o que reduz o nível de especialidade do usuário, uma vez que ele não precisa discernir entre esses dois aspectos e nem selecionar termos manualmente para caracterizá-los.

Para Pham [Pham et al. 2010], cuja proposta esta mais próxima do OPIS, o problema de busca-objeto se assemelha ao de aprendizagem de *ranking*, em que o principal objetivo é aprender uma função

[1] Apache Lucene, http://lucene.apache.org/core

de *ranking* através de uma função de aprendizagem, com base em um conjunto de características relevantes. A solução proposta consiste em desenvolver diversos motores de busca verticais para suportar a busca por páginas-objeto em diferentes domínios. Para isso, uma função de *ranking* deve ser aprendida para cada domínio específico. O desenvolvedor[2] deve submeter consultas por palavras-chave e anotar um conjunto de treinamento a partir das páginas recuperadas. Esse conjunto de treinamento tem suas características extraídas automaticamente e usadas em uma função de aprendizagem. O OPIS também adota o conteúdo das páginas para melhorar a busca-objeto através da classificação funcional (páginas-objeto ou não) das páginas. Porém, ele não considera a busca-objeto como um problema de aprendizagem de *ranking*. Ao invés disso, o processo de *ranking* fica a cargo do motor de busca ao qual acoplamos o método. Isso torna desnecessária a análise das informações estruturadas embutidas nas páginas, durante o processo de busca, para casá-las com as características que integram a função de *ranking* aprendida (por exemplo, "a palavra professor aparece no título").

3. OPIS: OBJECT PAGE IDENTIFYING AND SEARCHING

Esta seção descreve o método para identificação e busca de páginas-objeto, denominado OPIS (acrônimo para *Object Page Identifying and Searching*). O OPIS caracteriza-se por adotar técnicas de pré-processamento de texto e de aprendizagem de máquina na classificação de páginas baseada em conteúdo, a fim de permitir que somente páginas classificadas como páginas-objeto possam ser indexadas e recuperadas por consultas de usuários. O OPIS abrange a `construção de um classificador` e a `integração deste a um motor de busca convencional`, permitindo que os resultados de buscas-objeto se tornem mais precisos e, dessa forma, mais adequados às necessidades dos usuários.

Fig. 1. Visão geral do OPIS: Integração das atividades de identificação e busca

Na Figura 1 é apresentada uma visão geral do OPIS. Note que as páginas coletadas passam através de um processo de classificação e apenas as páginas classificadas como páginas-objeto seguem para a tarefa de indexação, permitindo que apenas estas sejam recuperadas em futuras consultas nesse domínio. Considerando o processo convencional realizado pelos GSEs, a única diferença é a introdução do classificador (identificador), após a tarefa de coleta, para a filtragem das páginas-objeto.

O classificador do OPIS é construído com base em um conjunto de atividades. Inicialmente, o `conjunto de treinamento`, que é a base do processo de aprendizagem, deve ser criado, sendo ne-

[2]Pessoa responsável por guiar o treinamento da função de *ranking* para um domínio específico, permitindo que usuários possam, então, submeter consultas relacionadas a esse domínio.

cessária a definição de um domínio, a coleta (que pode ser simplificada com o auxílio de um coletor focado [Chakrabarti et al. 1999][Torkestani 2012]) de um conjunto de páginas relacionadas a esse domínio e a rotulação (quanto a ser ou não uma página-objeto) dessas páginas de treinamento. Após, devido a maioria do conteúdo das páginas web ser textual e não estruturado, algumas atividades de `pré-processamento`, como a remoção de *tags* HTML, a tradução de termos estrangeiros, a remoção de *stopwords* e a representação do conteúdo através do Modelo de Espaço Vetorial (do Inglês, *Vector Space Model* – VSM) [Manning et al. 2008], são aplicadas às páginas coletadas. Por fim, a `criação do modelo de classificação`, para a identificação de páginas-objeto, pode ser iniciada. Esse passo envolve a escolha e parametrização de um algoritmo de classificação, o seu treinamento através de um conjunto de páginas e a análise de desempenho do modelo gerado.

4. CASO DE ESTUDO

Nesta seção, é apresentada a implementação e a avaliação do OPIS em um caso de estudo a fim de demonstrar a aplicação e a viabilidade do método em um domínio real. O objetivo desse caso de estudo é permitir a realização de buscas-objeto, tanto em um motor de busca convencional quanto no OPIS integrado a esse motor, e mostrar, através da avaliação dos resultados, que o OPIS melhora a precisão de buscas-objeto. O domínio escolhido foi o de pesquisadores, por possuir um número representativo de páginas-objeto disponíveis e por abranger diferentes tipos de páginas-objeto (como institucionais, pessoais e currículos).

4.1 Implementação

Inicialmente, um `conjunto de páginas` foi coletado, de modo que um subconjunto (25% dessa coleção) fosse considerado no treinamento do classificador e restante usado como coleção a ser indexada pelo motor de busca. Para isso, foram selecionadas páginas-*hub*, que se caracterizam por apresentarem uma lista de pesquisadores e *links* para suas páginas-objeto, relacionadas ao corpo docente de cursos de graduação em Ciência da Computação, Matemática e Estatística de 10 universidades brasileiras. Essas páginas-*hub* foram automaticamente analisadas, por meio de uma aplicação desenvolvida com o auxílio da biblioteca *HTML Parser*[3], e as páginas-objeto indicadas pelos *links* presentes nelas foram coletadas. A biblioteca *Google Custom Search*[4] foi utilizada para coletar exemplos negativos (páginas não objeto), através da submissão de consultas gerais, relacionadas às universidades consideradas, e do armazenamento das páginas resultantes. Após, todas as páginas armazenadas foram manualmente classificadas entre as classes `páginas-hub`, `página-objeto` e `páginas não objeto`, tendo cada uma obtido a quantidade de 114, 939 e 2012 páginas, respectivamente. A classe *hub* foi considerada nesse domínio na tentativa de melhorar a aprendizagem do classificador, uma vez que suas páginas podem conter diversas semelhanças com as páginas-objeto.

Durante o `pré-processamento`, a biblioteca *HTML Parser* foi utilizada na extração do conteúdo textual (sem *tags* HTML) das páginas web. Como a coleção é composta por páginas de universidades brasileiras e, no domínio de pesquisadores, é frequente a ocorrência de termos em Inglês, usou-se também a biblioteca *Web Translator Java*[5] para traduzir esses termos para o Português. Para representar o conteúdo textual das páginas através do modelo VSM, usou-se o filtro *StringToWordVector*, considerando TF-IDF [Manning et al. 2008] como forma de ponderação dos termos e o uso de uma lista de *stopwords* em Português, fornecida pelo *Snowball*[6], em sua parametrização. Esse filtro pertence à biblioteca de mineração *Weka*[7], usada neste trabalho por ser de código aberto e apresentar diversos algoritmos tanto para o pré-processamento dos dados quanto para o processo de aprendizagem.

[3]HTML Parser, http://htmlparser.sourceforge.net
[4]Google Custom Search, https://developers.google.com/custom-search
[5]Web Translator Java API, http://sourceforge.net/projects/webtranslator
[6]Snowball Portuguese Stop Word List, http://snowball.tartarus.org/algorithms/portuguese/stop.txt
[7]Weka Data Mining Software API, http://www.cs.waikato.ac.nz/ml/weka

Para a `criação do modelo de classificação` foram testados três algoritmos: *J48* (Árvore de Decisão), *NaiveBayes* (Probabilístico) e *LibSVM* (Máquinas de Vetores de Suporte). Embora não tenham apresentado grandes diferenças quanto à precisão da classificação, o primeiro algoritmo se mostrou mais lento e o terceiro obteve melhor precisão que os demais. Dessa forma, o *LibSVM* [Chang and Lin 2013] foi utilizado, com núcleo linear (por ser de rápida execução), através do *Weka* e treinado com 25% da coleção criada anteriormente. O motor de busca convencional *Lucene* foi usado na `integração do classificador`, de modo a permitir que somente páginas classificadas como páginas-objeto sejam indexadas e recuperadas. O *Lucene*, assim como as ferramentas já mencionadas, fornece uma biblioteca livre em Java, o que garante a portabilidade do sistema. Para a integração, uma aplicação em Java, que usa as bibliotecas *Lucene* e *Weka* para classificar, indexar e buscar páginas, foi desenvolvida. Quando o método de indexação é chamado, o documento a ser indexado passa por uma atividade adicional, onde é testado pelo classificador, e somente passa pelo processo de indexação se for classificado como uma página-objeto.

4.2 Avaliação Experimental

Experimentos foram realizados, considerando o caso de estudo desenvolvido, com o objetivo de avaliar a influência exercida pelo OPIS nos resultados de busca-objeto, em relação a um motor de busca convencional. Os experimentos foram executados em um PC padrão, usando as ferramentas mencionadas anteriormente, e consistiram na submissão de buscas-objeto do domínio de pesquisadores e na avaliação da relevância dos resultados recuperados por essas consultas. Participaram desses experimentos 10 usuários, que submeteram e avaliaram cinco consultas cada. A fim de tornar o processo menos exaustivo, os usuários foram divididos em dois grupos, de modo que cada grupo realizasse suas consultas em apenas um dos métodos, ou seja, no motor de busca com a adição do classificador (OPIS) ou sem (*baseline*). Após, essas consultas foram reproduzidas e avaliadas no método oposto. Para evitar o uso de diferentes critérios de relevância durante a reprodução das consultas, os usuários especificaram o objetivo e os tipos de resultados que consideraram relevantes em cada uma de suas consultas através de uma área de texto adicional presente na interface de usuário, que foi especialmente desenvolvida para a realização desses experimentos.

Como não era viável solicitar que os usuários avaliassem todos os resultados recuperados pelas consultas, apenas os 20 primeiros foram apresentados para serem analisados. Para medir a precisão das páginas apresentadas e ter um indicativo de suas posições no *ranking*, a métrica de precisão em n (p@n)[Manning et al. 2008], que considera somente os primeiros n resultados recuperados pelo sistema, foi usada com n de 5, 10, 15, e 20.

As médias para todas as consultas em ambos os casos (com e sem o OPIS) são apresentadas na Tabela I. Considerando as 20 primeiras páginas retornadas, o OPIS apresenta uma melhora de 56% na precisão. Isso acontece devido ao fato de muitas páginas irrelevantes (páginas não objeto) retornadas pelo motor de busca convencional para buscas-objeto serem descartadas pelo OPIS, o que torna os resultados mais precisos. Com base no Teste-T, é possível afirmar que o OPIS melhora significativamente (valor-p do Teste-T < 0.01) a precisão para buscas-objeto, em todos os pontos de corte considerados, quando comparado ao motor de busca convencional.

Table I. Resultados para todas as consultas em ambos os métodos

	Média das 50 consultas dos usuários			
	p@5	p@10	p@15	p@20
OPIS	0.600	0.526	0.489	0.453
Baseline	0.364	0.342	0.312	0.289

5. CONSIDERAÇÕES FINAIS

Neste artigo, foi proposta uma solução para o problema de busca-objeto em motores de busca convencionais, através de um novo método, denominado OPIS, para a identificação e a busca de páginas-objeto. O OPIS usa técnicas aplicadas na classificação baseada em conteúdo da web, como atividades de pré-processamento de texto e algoritmos de aprendizagem de máquina, para permitir que apenas páginas classificadas como páginas-objeto sejam indexadas e recuperadas pelas consultas dos usuários.

O OPIS teve sua viabilidade e aplicação demonstrada através da implementação de um caso de estudo no domínio de pesquisadores. Experimentos preliminares foram realizados nesse caso de estudo, contando com a contribuição de usuários, que submeteram e avaliaram buscas-objeto tanto em um motor de busca convencional quanto no OPIS integrado a esse motor. Os resultados mostraram que o OPIS forneceu um ganho de aproximadamente 56%, considerando a média das precisões das 20 primeiras páginas recuperadas por todas as consultas.

Como trabalhos futuros, pretende-se simplificar a construção do classificador através do uso de realimentação de relevância e reduzir o problema de ambiguidade das palavras-chave por meio da expansão automática de consultas. Uma experimentação mais exaustiva, considerando um domínio adicional e o trabalho de busca-objeto desenvolvido por Pham [Pham et al. 2010], apresentado nos trabalhos relacionados, como *baseline*, também faz-se necessária.

REFERENCES

Assis, G. T. *Uma Abordagem Baseada em Gênero para Coleta Temática de Páginas da Web*. M.S. thesis, Universidade Federal de Minas Gerais (UFMG), Brazil, 2008.

Baeza-Yates, R. A. and Ribeiro-Neto, B. A. *Modern Information Retrieval*. ACM Press / Addison-Wesley, 1999.

Bennett, P. N., Syore, K., and Dumais, S. T. Classification-Enhanced Ranking. In *Proceedings of the International World Wide Web Conferences*. Raleigh, EUA, pp. 111–120, 2010.

Blanco, L., Crescenzi, V., Merialdo, P., and Papotti, P. Supporting the Automatic Construction of Entity Aware Search Engines. In *Proceedings of the ACM Workshop on Web Information and Data Management*. Napa Valley, EUA, pp. 149–156, 2008.

Chakrabarti, S., Berg, M., and Dom, B. Focused Crawling: a New Approach to Topic-Specific Web Resource Discovery. In *Proceedings of the International World Wide Web Conferences*. Toronto, Canada, pp. 1623–1640, 1999.

Chang, C. and Lin, C. LIBSVM: A Library for Support Vector Machines, 2013. http://www.csie.ntu.edu.tw/ cjlin/libsvm.

Geng, B., Yang, L., Xu, C., and Hua, X. Ranking Model Adaptation for Domain-Specific Search. In *Proceedings of the Conference on Information and Knowledge Management*. Hong Kong, China, pp. 197–206, 2009.

Ji, L., Yan, J., Liu, N., Zhang, W., Fan, W., and Chen, Z. ExSearch: A Novel Vertical Search Engine for Online Barter Business. In *Proceedings of the Conference on Information and Knowledge Management*. Hong Kong, China, pp. 1357–1366, 2009.

Lee, H., Nazareno, F., Jung, S., and Cho, W. A Vertical Search Engine for School Information Based on Heritrix and Lucene. In *Proceedings of the International Conference on Convergence and Hybrid Information Technology*. Daejeon, Korea, pp. 344–351, 2011.

Luo, G. Design and Evaluation of the iMed Intelligent Medical Search Engine. In *Proceedings of the IEEE International Conference on Data Engineering*. Shanghai, China, pp. 1379–1390, 2009.

Manning, C. D., Raghavan, P., and Schütze, H. *An Introduction to Information Retrieval*. Cambridge University Press, 2008.

Nie, Z., Ma, Y., Shi, S., Wen, J., and Ma, W. Web Object Retrieval. In *Proceedings of the International World Wide Web Conferences*. Banff, Canada, pp. 81–90, 2007.

Pham, K. C., Rizzolo, N., Small, K., Chang, K. C., and Roth, D. Object Search: Supporting Structured Queries in Web Search Engines. In *Proceedings of the NAACL HTL - Workshop on Semantic Search*. Los Angeles, EUA, pp. 44–52, 2010.

Torkestani, J. A. An Adaptive Focused Web Crawling Algorithm Based on Learning Automata. *Applied Intelligence* vol. 37, pp. 586–601, 2012.

Caracterização do uso de *hashtags* do Twitter para mensurar o sentimento da população *online*: Um estudo de caso nas Eleições Presidenciais dos EUA em 2012

Glívia A. R. Barbosa, Pedro H. F. Holanda, Geanderson E. dos Santos, Conrado C. da Costa. Ismael S. Silva, Adriano Veloso e Wagner Meira Jr.

Pontifícia Universidade Católica de Minas Gerais, Brasil
{pedro.holanda, geanderson.santos, conrado.correa}@sga.pucminas.br

Universidade Federal de Minas Gerais, Brasil
{gliviaangelica, ismael.silva, adrianov, meira}@dcc.ufmg.br

Resumo. Neste artigo, descrevemos os resultados parciais e as direções futuras de uma pesquisa em andamento, cujo objetivo principal é analisar se a opinião expressa através de *hashtags* do Twitter pode contribuir para identificar e monitorar o sentimento da população *online* em relação a diferentes tópicos. Uma vez confirmada essa contribuição, esse tipo de *hashtag* pode ser utilizada como insumo para aplicações que visam detectar e monitorar automaticamente o sentimento das pessoas, sobre eventos ou tópicos específicos, expresso a partir do grande fluxo de dados proveniente das mídias sociais.

Abstract. *In this paper we describe the partial results and future directions of a research in progress, which main goal is to analyze whether the opinion expressed through Twitter hashtags can help to identify and track online sentiment. Once confirmed this contribution, this type of hashtag can be used as input for applications that aim to detect and track automatically online population sentiment about different events.*

Categories and Subject Descriptors: H.Information Systems [H.m. Miscellaneous]: Databases

Keywords: *Hashtag*, Análise de Sentimento *Online*, Descoberta de Conhecimento, Twitter

1. INTRODUÇÃO

Considerando os benefícios da análise de sentimento *online*, referenciados em [Diakopoulos & Shamma 2010][Chew & Eysenbach 2010][Jansen et al. 2009] e [Silva et al. 2011], e o potencial dos softwares sociais como fonte de conteúdo opinativo, muitas técnicas de classificação automática têm sido propostas para automatizar esse tipo de análise. No processo de classificação automática é preciso utilizar um conjunto de treinamento previamente rotulado para que o algoritmo seja capaz de aprender e classificar as mensagens que chegam sem rótulos (i.e., para que o classificador execute a predição). Quando essa classificação ocorre em fluxo de sentimentos, para que as mensagens que chegam sejam bem representadas, novos exemplos rotulados devem ser inseridos ao conjunto de treinamento e assimilados pelo modelo de classificação com o passar do tempo. Logo, esses exemplos devem ser extraídos a partir do fluxo corrente [Silva et al. 2011].

Entretanto, não existe um consenso sobre a melhor forma de geração desse conjunto de treino. Na maioria dos casos, ele é gerado por um usuário especialista no domínio do problema. Contudo, este processo pode ter um custo proibitivo, principalmente no cenário de análise de fluxo de sentimentos. Isso porque o conjunto de treino deve ser atualizado em tempo real [Silva et al. 2011][Wu et al. 2012]. Para enfrentar os desafios relacionados a rotulagem de dados durante a análise do fluxo de sentimento *online*, pesquisadores têm utilizado técnicas de "aprendizado ativo" e "aprendizado semi-

supervisionado", onde o algoritmo aprende sobre o fluxo de mensagens, a partir de um pequeno conjunto de treino, que é atualizado automaticamente com o passar do tempo [Silva et al. 2011].

Além dessas técnicas, conforme apresentado em [Barbosa et al. 2012], as *hashtags* atribuídas às mensagens podem ser promissoras nesse processo de geração de dados de treinamento para classificação automática do sentimento. Isso porque, uma vez confirmada a eficácia das *hashtags* para representar e agrupar um grande volume de mensagens que expressam sentimento, esse metadado pode ser utilizado como insumo para treinar algoritmos que visam detectar e monitorar automaticamente a opinião das pessoas, expressa a partir do grande fluxo de dados proveniente dos softwares sociais *online* [Barbosa et al. 2012]. Diante dessa hipótese, os autores afirmam que para verificar a efetividade das *hashtags* nesse contexto é preciso caracterizar e apreciar o uso desse recurso em diferentes cenários, como em eleições, eventos esportivos e propagação de doenças [Barbosa et al. 2012].

Com o intuito de estender a investigação iniciada por [Barbosa et al., 2012] e motivados pelos benefícios que a detecção de sentimento dos usuários oferece para diversas áreas, neste artigo, descrevemos os resultados obtidos em um estudo de caso que buscou analisar se a opinião expressa através de *hashtags* do Twitter pode contribuir para representar, agrupar e propagar o sentimento da população *online* no contexto das Eleições Presidenciais dos Estados Unidos da América (EUA) em 2012. As observações obtidas até o momento convergem para os resultados apresentados em [Barbosa et al. 2012], porém, em outra base de dados. Dessa forma, nossos resultados sustentam a hipótese e relevância dessa pesquisa, uma vez que apontam para a eficácia das *hashtags* como um recurso promissor para retratar e monitorar o sentimento da população *online.*

Este trabalho está organizado da seguinte forma, na próxima seção apresentamos os trabalhos relacionados. Posteriormente, descrevemos a coleção de dados utilizada, bem como a metodologia adotada. Em seguida, as análises realizadas e os resultados obtidos são apresentados e, finalmente, a última seção apresenta as conclusões obtidas até o momento e as direções futuras dessa pesquisa.

2. TRABALHOS RELACIONADOS

No que se refere a pesquisas que tratam de *hashtags*, [Romero et al. 2011] direciona sua pesquisa para verificar a hipótese de que diferentes tipos de informações se propagam de formas distintas em redes sociais. Para isto, eles analisam a difusão de *hashtags*, isto porque o padrão de propagação de uma *hashtag* utilizada para categorizar um tópico, reflete na propagação deste tópico [Romero et al. 2011]. Em [Cunha et al. 2011] é apresentado um estudo inspirado na linguística para verificar como as *hashtags* são criadas, utilizadas e disseminadas. O objetivo foi investigar e constatar o impacto dessas *hashtags* na inovação linguística. Os resultados revelaram que as *hashtags* podem servir efetivamente como modelo para caracterizar a propagação de formas linguísticas.

Finalmente, em [Barbosa et al. 2012] são apresentados os resultados iniciais sobre a efetividade das *hashtags* do Twitter para representar o sentimento da população *online* no contexto das Eleições Presidenciais no Brasil em 2010. Os resultados reportados indicaram que a população *online* fez uso desse recurso em suas mensagens (i.e., incluíram *hashtags* nos *tweets*) para expressar sentimentos negativos e positivos em relação à candidata Dilma Rousseff. A análise mostrou também que, no contexto das eleições no Brasil, existe uma equivalência entre o sentimento *online* expresso com as *hashtags* e o sentimento real da população [Barbosa et al. 2012].

Contudo, os autores destacam que para confirmar a eficácia das *hashtags* nesse contexto, é necessário avaliar seu uso para expressar sentimento em relação a diferentes eventos. Sendo assim, o presente trabalho visa estender a pesquisa iniciada por [Barbosa et al. 2012] caracterizando o uso dessas *hashtags* nas eleições presidenciais dos EUA. Uma vez confirmada a sua eficácia em diferentes cenários, para representar o sentimento das mensagens que elas acompanham, será possível

avaliar a acurácia das *hashtags* que expressam sentimento como insumo para treinar algoritmos que visam detectar e monitorar automaticamente a opinião da população *online*.

3. COLEÇÃO E METODOLOGIA PARA ANÁLISE DE DADOS

Considerando o objetivo desse trabalho, a análise proposta foi realizada a partir dos *tweets* relacionados à campanha eleitoral do atual presidente dos Estados Unidos da América (EUA), Barack Obama, durante as Eleições Presidências em 2012. Isto porque, esse foi um dos principais eventos ocorridos no ultimo ano e alcançou alta popularidade no Twitter em 2012 [Twitter, 2012].

Os dados para geração da nossa coleção foram obtidos a partir da *API de streaming* do Twitter, onde foram buscados *tweets* que continham em seu texto a palavra "Obama" e que foram publicados entre os dias 16 de outubro de 2012 e 12 de novembro de 2012 (i.e., período que compreende a campanha eleitoral e a votação – que aconteceu em 06 de novembro de 2012).

No total, 16.779.113 *tweets* foram coletados. Dentre as mensagens selecionadas, 7.630.067 (45%) continham algum tipo de *hashtag* que poderia expressar sentimento ou não. Dado que uma mesma *hashtag* poderia estar sendo utilizada mais de uma vez, computamos a quantidade de *hashtags* distintas. No total foram identificadas 321.579 *hashtags* diferentes.

Nossa apreciação se concentrou, inicialmente, em analisar o conteúdo de cada *hashtag* sem relacioná-lo ao *tweet* ao qual a mesma foi atribuída. Dessa forma poderíamos ver a expressividade da *hashtag*. Para essa análise três autores deste trabalho classificaram, manualmente, as 30.000 *hashtags* distintas mais frequentes (i.e, top-*hashtags*), iniciando da mais frequente para a menos frequente. A classificação ocorreu a partir da leitura de cada *hashtag* e essa por sua vez era classificada como: positiva ou negativa, quando expressava sentimento explicitamente; ambígua, quando o sentimento era implícito; ou neutra, quando a *hashtag* não expressava sentimento. A Tabela 1 descreve alguns exemplos de *hashtags* identificadas para cada categoria de sentimento.

Tabela I. Exemplo de *Hashtags* Classificadas

Positivas	**Negativas**	**Ambíguas**	**Neutras**
#TeamObama	#CantAfford4More	#Romney	#Election2012
#voteObama	#Nobama	#barackobama	#Debate

Finalizada a etapa da classificação, foi possível verificar que das 30.000 top-*hashtags* distintas classificadas, 1.931 expressam algum sentimento positivo, negativo ou ambíguo. Dessas *hashtags*: 562 (29%) foram classificadas como negativas; 779 (40%) como positivas; e 590 (31%) como ambíguas. É importante ressaltar que, embora o número de *hashtags* distintas que expressam sentimento seja menor, quando comparado com as *hashtags* neutras, a partir das 1.931 top-*hashtags* positivas, negativas ou ambíguas, 2.256.519 *tweets* contidos na base de dados poderiam ser classificados quanto ao seu sentimento. A seguir descrevemos as análises realizadas, bem como os resultados obtidos.

4. CARACTERIZAÇÃO DO USO DAS *HASHTAGS* PARA EXPRESSAR SENTIMENTOS NAS ELEIÇÕES PRESIDENCIAIS DOS EUA EM 2012

Para caracterizar como os usuários do Twitter têm utilizado as *hashtags* para expressar, compartilhar e propagar suas opiniões sobre as eleições nos EUA em 2012, buscamos analisar dois aspectos: (1) se a opinião reportada a partir das *hashtags* reflete no sentimento real da população ao longo do tempo e (2) como tem sido a popularidade e propagação dessas *hashtags* no Twitter.

Em relação a distribuição temporal do sentimento reportado a partir das *hashtags*, conforme demonstrado pelo gráfico da Figura 1(a), é possível verificar que tanto as positivas, quanto as negativas foram utilizadas durante todo o período entre 16 de Outubro e 12 de Novembro de 2012 (i.e., 6 dias depois do fim das eleições, onde Barack Obama venceu).

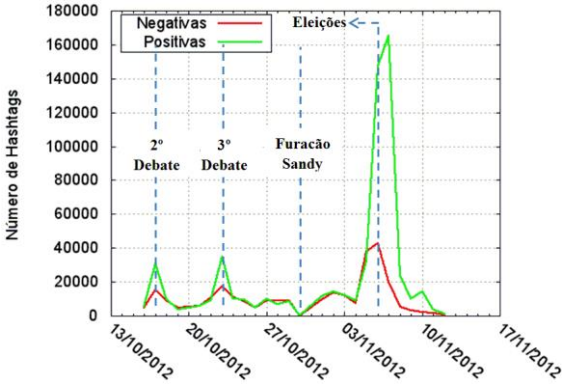


Fig 1 (a) Frequência diária de *hashtags* positivas e negativas em relação a Barack Obama no período das Eleições

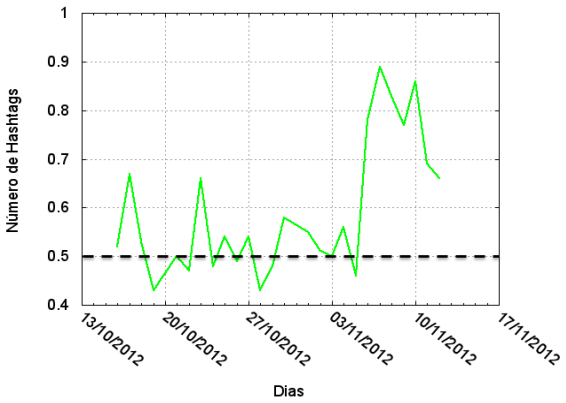


Fig 1. (b) Positividade diária dos *tweets* que mencionam Barack Obama

Fig. 1. Distribuição temporal do sentimento reportado a partir das *hashtags*.

Além disso, observou-se que, na maior parte do tempo, existe uma equivalência entre a frequência de *hashtags* positivas e negativas em relação ao então candidato a presidência Barack Obama. Contudo, esse comportamento não ocorre nos dias 16 e 22 de Outubro, que correspondem respectivamente ao 2º e 3º debates presidenciais [CNN Politics, 2012], bem como, a partir do dia 06 de Novembro, dia em que de fato Obama venceu as eleições [The New York Times, 2012 (b)]. Nas datas mencionadas nota-se uma frequência maior de *hashtags* positivas para Obama. Esses indicadores sugerem uma provável vitória desse candidato antes das eleições, sobretudo pelo sentimento reportado no dia dos debates.

Outra comparação interessante entre o mundo real e o sentimento da população *online* retratado no gráfico da Figura 1(a), refere-se a frequência das *hashtags* no dia 30 de Outubro nos EUA. Nota-se que nessa data, não houve nenhuma mensagem contemplando *hashtags* positivas ou negativas para Obama. Essa ausência de *hashtags* relacionadas as eleições pode ser reflexo da ascensão de outro tópico de discussão que repercutiu mundialmente na mesma data, o Furacão Sandy. Uma vez que, neste dia, o furacão que atingia o leste dos EUA chegou a cidades dos estados de New York e New Jersey provocando grande devastação [The New York Times, 2012 (a)] e gerando grande repercussão no Twitter em todo mundo [Twitter, 2012].

Para melhor evidenciar a correspondência entre o sentimento *online* sobre as eleições nos EUA, reportado a partir das *hashtags* do Twitter, e o sentimento real da população, o gráfico da Figura 1 (b) descreve a positividade diária dos *tweets* que mencionam Obama. Esse indicador foi calculado diariamente, dividindo a quantidade de mensagens que contemplavam *hashtags* positivas pela soma de mensagens que continham *hashtags* positivas e negativas. Através desse gráfico é possível notar que a positividade se mantém, na maioria das vezes, acima de 50%. Se compararmos esse indicador com o sentimento real da população, é possível notar uma relação, isso porque, no mundo real, Obama venceu as eleições. De acordo com os dados da [CNN Politics, 2012] Obama obteve apoio de 332 votos do colégio eleitoral, contra 206 de seu oponente e, além disso, alcançou 51% dos votos populares.

A próxima etapa da caracterização consistiu em verificar a popularidade das *hashtags* e seu processo de propagação no Twitter. Em redes sociais, a difusão de um conteúdo acontece em forma de cascata, onde as pessoas, conscientemente ou não, fazem suas escolhas, baseadas no comportamento e escolhas feitas por outras. Avaliar a popularidade das *hashtags* que expressam sentimento, bem como a maneira como elas têm sido utilizadas na rede podem indicar a ocorrência ou não desse fenômeno de propagação [Easley & Kleinberg. 2010].

O gráfico da Figura 2 apresenta o ranking de popularidade de cada *hashtag*, em outras palavras, indica quantas vezes cada *hashtag* que expressa sentimento foi utilizada, estabelecendo um ranking ordenado a partir da primeira *hashtag* mais popular para cada tipo de sentimento.

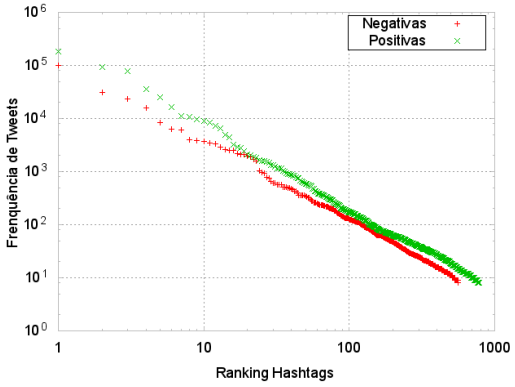


Fig. 2. Ranking de Frequência das *Hashtags* em *tweets* que mencionam Obama

Tabela II. Distribuição das *Hashtags* mais populares que expressam sentimento

% de *Hashtags* que aparecem em pelo menos *j* *Tweets*				
	***j* = 100.000**	***j* = 10.000**	***j* = 1.000**	***j* = 100**
Negativas	0,18%	0,71%	4,27%	21,53%
Positivas	0,13%	1,03%	4,62%	17,97%

Através de uma análise comparativa da Figura 2 é possível verificar que a frequência das *hashtags* positivas é superior em relação às negativas em todo o ranking. Além disso, para ambos os sentimentos, existe uma alta popularidade para um número reduzido de *hashtags* (e.g., a *hashtag* positiva mais popular, apareceu em 185.211 *tweets*, já a negativa classificada como 1ª no ranking foi atribuída a 100.535 mensagens). Isto indica a existência do fenômeno: "ricos ficam mais ricos", onde poucas *hashtags* que expressam sentimento – as mais populares – são usadas na maioria dos *tweets*, enquanto que uma vasta quantidade delas são usadas apenas em poucas mensagens [Easley & Kleinberg. 2010]. Para uma melhor compreensão desse resultado a Tabela II, apresenta a distribuição das *hashtags* positivas e negativas mais populares. Observa-se que menos de 10% das *hashtags*, que expressam ambos os sentimentos, aparecem em mais de 1.000 *tweets*.

Os resultados apresentados até o momento demonstraram que as *hashtags* estão sendo utilizadas pelos usuários não apenas para categorizar conteúdo, mas também para expressar sentimento. Além disso, reforçam os resultados publicados por [Barbosa et al. 2012], por indicar que a intensidade do sentimento expresso a partir das *hashtags* é similar ao sentimento da real população. Entretanto, até o momento, as *hashtags* foram analisadas sem associá-las aos *tweets* que elas acompanham. Diante disso, procuramos investigar se a *hashtag* era utilizada apenas como uma forma de agrupar e reafirmar o sentimento expresso na mensagem ou se os usuários têm utilizado esse metadado para atribuir sentimento ao conteúdo que sem a *hashtag* poderia ser inexpressivo.

Para realizar essa análise 3.000 *tweets* que continham as *hashtags* classificadas como positivas ou negativas foram selecionados aleatoriamente. Durante a avaliação foi preciso verificar: (1) se o sentimento contido na mensagem correspondia ao sentimento expresso pela *hashtag* e (2) se o conteúdo do *tweet* expressava sentimento ou não sem a *hashtag* associada. A avaliação revelou que

96% dos *tweets* classificados expressavam o mesmo sentimento das *hashtags* associadas a eles. Além disso, 83% dos *tweets* analisados eram expressivos sem a *hashtag*.

Uma vez que esses resultados também foram observados na base analisada por [Barbosa et al. 2012] – os autores indicaram que as *hashtags* representavam o sentimento de 97% dos *tweets* de sua amostra -, eles reforçam a hipótese de que esse metadado pode contribuir para a classificação automática de grande volume de dados. Isso porque, as *hashtags* de fato representam os *tweets* que as utilizam, descrevem o sentimento real da população e, além disso, uma mesma *hashtag* representa o conteúdo de milhares de *tweets*, conforme demonstrado no gráfico da frequência (Figura 2). Logo, utilizá-las como recurso para treino dos classificadores, pode aumentar a eficácia do classificador, além de reduzir o custo para a criação do conjunto de treino para classificação.

5. CONCLUSÕES E DIREÇÕES FUTURAS

Os resultados apresentados nesta pesquisa convergem para aqueles apresentados em [Barbosa et al. 2012] em outra base de dados. Nesse sentido, nossos resultados reforçam a hipótese relacionada a eficácia das *hashtags* como um recurso promissor para monitorar o sentimento da população *online*.

Como etapas posteriores, esse trabalho ainda prevê a caracterização do uso das hashtags, relacionadas a sentimento em outros tópicos de interesse (e.g., "Epidemias" e "Eventos Esportivos"), bem como a análise da correlação e acurácia das mesmas para descrever o sentimento do tweet ao qual a mesma foi atribuída. Com essas diferentes dimensões, buscamos verificar se existem ou não padrões no uso e propagação desse tipo de hashtag, de tal forma que seja possível utilizá-las como insumo para treinar aplicações que visam detectar e monitorar automaticamente a opinião das pessoas, expressa a partir do grande fluxo de dados proveniente dos softwares sociais *online.*

Vale ressaltar que, este tipo de aplicação tem se tornado cada vez mais útil para usuários interessados em acompanhar o sentimento da população *online* sobre determinados acontecimentos, sobretudo, para aqueles usuários ou órgãos responsáveis por tomadas de decisão em relação ao assunto sobre o qual o sentimento está sendo monitorado.

REFERÊNCIAS

BARBOSA , G. A. R., SILVA, I. S., ZAKI, M. J., MEIRA JR. W., PRATES, R. O., AND VELOSO, A Characterizing the effectiveness of twitter *hashtags* to detect and track *online* population sentiment. In Proc. CHI, pp. 2621-2626, 2012.

CHEW, C. AND EYSENBACH, G. Pandemics in the Age of Twitter: Content Analysis of Tweets during the 2009 H1N1 Outbreak. PLoS ONE, 5(11):e14118+, 2010.

CNN POLITICS. Presidential Debates. Election Center. 2012. Disponível em: < http://goo.gl/3Apw1 >.

CUNHA, E., MAGNO, G., COMARELA, G., ALMEIDA, V., GONÇALVES, M. A., AND BENEVENUTO, F. Analyzing the dynamic evolution of *hashtags* on twitter: a language-based approach. In Proc. of the Workshop on LSM, pp. 58-65, 2011.

DIAKOPOULOS, N. A. AND SHAMMA, D. A.. Characterizing debate performance via aggregated twitter sentiment. In Proc. CHI, pp. 1195-1198, 2010.

EASLEY, D. AND KLEINBERG, J. Networks, Crowds, and Markets: Reasoning About a Highly Connected World. Cambridge University Press, 2010.

JANSEN, B. J., ZHANG, SOBEL, M., K., AND CHOWDURY, A.. Micro-blogging as *online* word of mouth branding. In EA CHI, pp. 3859-3864, 2009.

ROMERO D. M., MEEDER, B., AND KLEINBERG, J. Diferences in the mechanics of information difusion across topics: idioms, political *hashtags*, and complex contagion on twitter. In Proc. WWW, pp. 695-704, 2011.

SILVA, I. S., GOMIDE, J., VELOSO, A., MEIRA JR, W., AND e FERREIRA, R. Effective Sentiment Stream Analysis with Self-Augmenting Training and Demand-Driven Projection. In Proc of. SIGIR, 2011.

THE NEW YORK TIMES (a). Hurricane Sandy: Covering the Storm. 2012. Disponível em: < http://goo.gl/X7lA9 >.

THE NEW YORK TIMES (b). President Results. Election 2012. 2012. Disponível em: < http://goo.gl/BtM3J >.

TWITTER. 2012 Year On Twitter. 2012. Disponível em: < https://2012.twitter.com/en/trends.html>.

WU, X., LI, P. AND HU, X. 2012. Learning from concept drifting data streams with unlabeled data. Neurocomput. 92 (September 2012), pp. 145-155.

On Using an Automatic, Autonomous and Non-Intrusive Approach for Rewriting SQL Queries

Arlino H. M. de Araújo[1,2], José Maria Monteiro[2], José Antônio F. de Macêdo[2], Júlio A. Tavares[3], Angelo Brayner[3], Sérgio Lifschitz[4]

[1] Federal University of Piauí, Brazil
[2] Federal University of Ceará, Brazil
{arlino,monteiro,jose.macedo}@lia.ufc.br
[3] University of Fortaleza, Brazil
brayner@unifor.br
[4] Pontifical Catholic University of Rio de Janeiro
sergio@inf.puc-rio.br

Abstract. Database applications have become very complex, dealing with a huge volume of data and database objects. Concurrently, low query response time and high transaction throughput have emerged as mandatory requirements to be ensured by database management systems (DBMSs). Among other possible interventions regarding database performance, SQL query rewriting has been shown quite efficient. The idea is to rewrite a new SQL statement equivalent to the statement initially formulated, where the new SQL statement provides performance gains w.r.t. query response time. In this paper, we propose an automatic and non-intrusive approach for rewriting SQL queries. The proposed approach has been implemented and its behavior was evaluated in three different DBMSs using TPC-H benchmark. The results show that the proposed approach is quite efficient and can be effectively considered by database professionals.

Categories and Subject Descriptors: H.2 [**Database Management**]: Miscellaneous

Keywords: Query processing, Database tuning, Query rewriting

1. INTRODUCTION

Database queries are expressed by means of high-level declarative languages, such as SQL (Structured Query Language), for instance. Such queries are submitted to the query engine, which is responsible for processing queries in database mangement systems (DBMSs). Thus, query engines should implement four main activities: query parsing, logical execution plan (LEP) generation, physical execution plan (PEP) generation and PEP execution. LEP and PEP generation and PEP execution are often called the query optimization. The cost models implemented by current query optimizers are very complex and rely on factors which may induce the optimizers to produce inefficient PEPs. In order to overcome such an issue, there are strategies to help optimizers, such as to provide query hints and query rewriting [Garcia-molina et al. 2000].

The rewriting technique consists in writing a new SQL statement Q_b equivalent to the statement Q_a initially formulated. Two queries are equivalent if and only if their execution produces the same result [Ramakrishnan and Gehrke 2002]. Thus, the executions of Q_a and Q_b return the same result. However, Q_b execution provides performance gains w.r.t. Q_a execution. In order to illustrate performance gains achieved by means of query rewriting, consider the queries Q_a and Q_b depicted in Figures 1 and 2, respectively. In order to execute those queries, the TPC-H benchmark's database has been created in a PostgreSQL server.

Looking more closely to Figures 1 and 2, one can observe that Q_a and Q_b are equivalent. Nonetheless, Q_a execution lasts $13.685ms$, while Q_b is executed in $1.825ms$.

Therefore, we may formulate the following hypothesis, the way a query Q is formulated in terms of SQL commands induces the query optimizer to produce a given PEP for Q. To show the veracity of our hypothesis, consider the execution plans for queries Q_a and Q_b depicted in Figures 3 and 4. Those plans have been yielded by PostgreSql's query engine. The PEP showed in Figure 4 is more efficient (1,825 ms) than PEP depicted in Figure 3 (13,685 ms). This fact stems from the least amount of data materialized in the "Materialize" operation, which is performed after the "Aggregate" operation (in the PEP illustrated in Figure 4).

```
SELECT *
FROM orders
WHERE o_totalprice < ALL (SELECT o_totalprice
                          FROM orders
                          WHERE o_orderpriority = '2-HIGH')

Execution time = 13,685 ms
```

Fig. 1. SQL expression for Q_a.

```
SELECT *
FROM orders
WHERE o_totalprice < (SELECT MIN(o_totalprice)
                      FROM orders
                      WHERE o_orderpriority = '2-HIGH')

Execution time = 1,825ms
```

Fig. 2. SQL expression for Q_b.

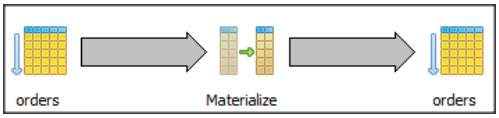


Fig. 3. PEP P_{Q_a} yielded by PostgreSQL for Q_a

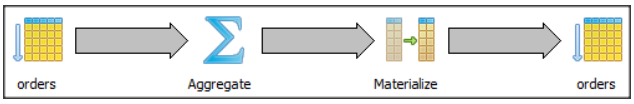


Fig. 4. PEP P_{Q_b} yielded by PostgreSQL for Q_b

Database query tuning explores these aspects in order to help query optimizers to produce better PEPs. In fact, there are tools that aim at tuning database query performance by providing rewriting recommendations. However, they adopt an offline approach, since they provide recommendations only when an human (e.g., DBA) requires such recommendations. Most of them are intrusive, which means that they are DBMS-specific. Additionally, they present a strong issue, for a given query they may provide several rewriting recommendations. In this case, the DBA is in charge to choose one of them without any indication (from the tool) that the chosen recommendation will provide the best performance gains regarding response time.

In this paper, we propose an automatic, autonomous and non-intrusive approach for rewriting SQL queries, denoted ARe-SQL. The proposed approach provides the necessary support to implement two different strategies for SQL-query rewriting. In the first strategy, ARe-SQL autonomously captures SQL queries submitted by users or applications and identify which queries should be rewritten. Thereafter, queries are sent to the database query engine. The second strategy works as an autonomous advisor, which analyze previously executed SQL statements in order to recommend (through alerts, reports and wizards) SQL tuning opportunities. In both strategies, the rewritten queries profit from performance gains as we show in Section 4. We have implemented the proposed approach and evaluated its behavior in three different DBMSs considering the TPC-H benchmark. The results show that our approach can be effectively considered by database professionals.

The rest of the paper is structured as follows. Section 2 discuss most relevant related work. The proposed approach is presented and analyzed in Section 3. In turn, Section 4 brings and analyzes experimental results. Finally, Section 5 concludes this paper.

2. RELATED WORK

Bruno *et al* propose in [Bruno et al. 2009] a framework, called Power Hints, which enables the creation and use of hints. In the proposed framework, hints are created by means of regular expressions, making it easy and flexible to create restrictions for query execution plans, allowing more precise tuning. The tool IBM Optim Development Studio [Studio 2010] collects query performance metrics in DB2. The metrics are query execution frequency, cost and time. Thus, with such metrics, it is possible to identify queries, for which query rewriting recommendation are worthwhile. It is important to mention that this tool collects metrics for a given period of time (defined by the DBA). Embarcadero DB Optimizer XE [Optimizer 2010] has the functionality of identifying hints which should be encoded in a given SQL statement, i.e., it does not propose query rewritings. Its goal is mainly to eliminate unnecessary outer joins and cartesian products. Additionally, Embarcadero may provide recommendation for improving index configuration. Quest SQL Optimizer for Oracle [Quest 2010] provides semantically equivalent SQL statements for a given query. In this case, the DBA should pass to the tool which query should be analyzed. Moreover, it is up to the DBA to choose one of the several recommended SQL statements.

Therefore, all the investigated SQL tuning tools (Automatic SQL Tuning Advisor [Dageville and Dias 2006], IBM Optim Development Studio [Studio 2010], Embarcadero DB Optimizer XE [Optimizer 2010] and Quest SQL Optimizer for Oracle [Quest 2010]) adopt an offline approach. In this sense, they transfer to the DBA the responsibility for defining the set of queries to be evaluated for choosing one of the several alternatives provided by them. Observe that DBAs work in a reactive way, i.e., they only trigger a tool or an advisor when the problem already exists. Furthermore, after identifying the problem (a time consuming query) and a possible solution, a DBA should rewrite the SQL statement, test and send it to a programmer to change the application code which is using the SQL statement. This whole process is rather time consuming.

3. ARE-SQL: AN AUTOMATIC, AUTONOMOUS AND NON-INTRUSIVE APPROACH FOR REWRITING SQL QUERIES

ARe-SQL's main functionality is to influence query optimizer to choose an effectual query execution plan. ARe-SQL proactively rewrites SQL statements which will induce query engine to produce better plans. In order to achieve its goal, ARe-SQL works directed by a set of heuristics. The heuristics consist of rules to identify potential opportunities for tuning SQL statements and the ways to rewrite the statements. Table I brings the eleven heuristics applied by ARe-SQL. Furthermore, it indicates whether or not each heuristic is currently implemented by three major DBMSs: PostgreSQL 8.3, Oracle 11g and SQL Server 2008.

Table I. Heuristics for SQL Tuning.

	Heuristcs for SQL Tuning	PostgreSQL	Oracle	SQL Server
H1	Transform queries which create and use temporary table into an equivalent sub-query.	No	No	No
H2	Eliminate unnecessary GROUP BY.	No	No	No
H3	Remove having clause whose predicates do not have any aggregate function. The predicates should be moved to a WHERE clause.	No	No	No
H4	Change query with disjunction in the WHERE to a union of query results.	No	Yes	No
H5	Remove ALL operation with greater/less-than comparison operators by including a MAX or MIN aggregate function in the subquery.	No	Yes	No
H6	Remove SOME operation with greater/less-than comparison operators by including a MAX or MIN aggregate function in the subquery.	No	Yes	No
H7	Remove ANY operation with greater/less-than comparison operators by including a MAX or MIN aggregate function in the subquery.	No	Yes	No
H8	Replace IN set operation by a join operation.	No	Yes	No
H9	Eliminate unnecessary DISTINCT.	No	No	No
H10	Move function applied to a column index to another position in the expression.	No	No	No
H11	Move arithmetic expression applied to a column index to another position in the expression.	No	No	No

3.1 Implementing ARe-SQL

In order to implement ARe-SQL, two different approaches were employed: assisted and automatic. For each approach, a different tool was implemented. However, both approaches present the following features: *(i)* Non-intrusive: completely decoupled from the source code of the DBMS. This allows that the conceived solution can be used with any DBMS and *(ii)* Independent of location: it can run on a machine different than that used to host the DBMS, not consuming server resources where the DBMS is hosted.

3.1.1 *Assisted Approach.* A query-rewriting tool based on assisted approach consists of an advisor which has the ability: *(i)* to capture the previously SQL statements executed by DBMS; *(ii)* to analyze these statements, and; *(iii)* to recommend (through alerts, reports or wizards) SQL tuning opportunities. Thus, the advisor can identify SQL statements that could be rewritten. Additionally, such a type of tool allow the DBA to interact with the tuning process. For instance, a DBA may select a subset of available heuristics to be applied for the SQL-query tuning process. In other words, the DBA can specify that some heuristics are unnecessary or inappropriate for a given database system or a given statement. We have implemented a tool, called ARe-SQL Advisor, which implements an assisted approach for ARe-SQL. The main components of the architecture illustrated in Figure 5 are the following:

—Agent for Workload Obtainment (AWO): This agent observes the operations submitted to the DBMS and retrieves the SQL statements and then stores them in the local metabase (Local MetaData - LM). This agent can be configured to run continuously (On-the-fly).

—Local Meta Data (LM): Database that stores workloads captured by AWO.

—Driver for Workload Access (DWA): This component enables ARe-SQL Advisor to access metabase (catalog) of a given DBMS.

—Agent for Statistics Obtainment (ASO): This component is in charge of accessing statistics information of the target DBMS, such as table cardinality, the amount of disk pages required to store a database table, the height of (B^+ tree) index structures and so forth.

—Driver for Statistics Access (DSA): driver that allows the ASO to retrieve statistics for a specific DBMS.

—Heuristic Set (HS): set of heuristics used by the agents to identify SQL statements with oportunities of tuning and in order to rewrite them. HS is composed of 11 heuristics, which are depicted in Table I. However, new heuristics can be defined and inserted into HS.

—Agent for SQL Tuning (AST): Component responsible for tuning a particular SQL statement using the set of heuristics (HS).

—Tuning Settings (TS): This component is preference file containing pairs <SQL statement Q, subset of heuristics H> defined by the DBA. Each pair indicates a heuristic subset chosen by DBA from HS that he/she wants to be applied for tuning Q. If there isn't an entry in this file for a given statement Q, all 11 heuristics are applied for tuning Q (in both, Assisted and Automatic Approaches).

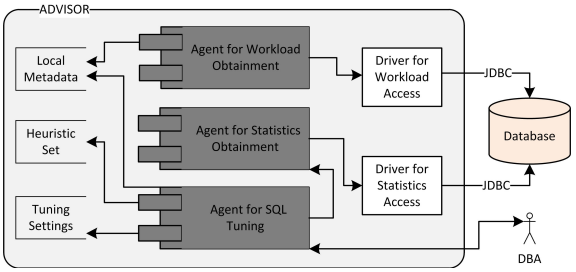


Fig. 5. ARe-SQL Advisor's Architecture.

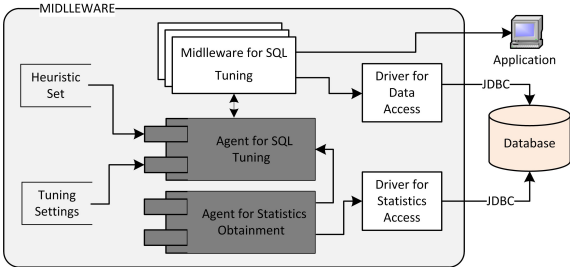


Fig. 6. Architecture of ARe-SQL's Automatic Approach.

3.1.2 *Automatic Approach.* The automatic approach consists of a middleware, ARe-SQL Mid, that works between the application and the DBMS. This middleware is responsible for: *(i)* automatically receiving SQL statements sent by applications; *(ii)* analyzing and rewriting them whenever necessary, and; *(iii)* submitting the statements (rewritten or not) to the DBMS. The architecture designed for the automatic approach is illustrated in Figure 6. The main components of this architecture, in addition to components that are also present in Figure 5, are:

—Middleware for SQL Tuning (MST): is responsible for receiving SQL statements from the applications; send them to the agent AST examine them; and receive the statements rewritten (or not) and send them to the DBMS; receive the result of executing these queries and send them to the applications.

—Driver for Data Access (DDA): driver that allows the engine of the middleware (Middleware for SQL Tuning) to send the SQL statements to the target DBMS.

4. EXPERIMENTAL RESULTS

4.1 Simulation Setup

We have evaluated ARe-SQL in three different scenarios. In the first scenario, TPC-H benchmark has been used, including its database and workload, which is composed of 23 queries. In the second scenario, a workload of 30 synthetic queries has been executed on TPC-H database. In both scenarios, we have used TPC-H benchmark with scale factor of 2 GB. Finally, in the third scenario, we exploit the database of a system used in several brazilian universities, called Integrated Management System (IMS), and another synthetic workload. Each synthetic workload is formed by 30 SQL queries. Each SQL query was written in order to enable the application of one of the heuristics presented in the Table I.

Table II. Execution times of the TPC-H queries 18 and 20 in their original formats and after being rewritten.

	PostgreSQL		SQL Server	
	Original	**Rewritten**	**Original**	**Rewritten**
Query TPC-H 18	134.807ms	111.933ms	19s	14s
Query TPC-H 23	912ms	712ms	15s	11s

All experiments were run on a Core i3-2100 (3.10GHz) server, with 4GB RAM and 500 GB HD. PostgreSQL 8.1, Oracle 11g and SQL Server 2008 have been used as database systems. For each experimentation scenario, three different experiments have been performed: i) The workload containing the original queries has been submitted to a database system. The results of this experiment have been be used as baseline; ii) The workload submitted to ARe-SQL advisor (assisted approach). Thereafter, the workload with tuned SQL queries were manually submitted to a database system, and; iii) Each query from original workload has been sent to the automatic approach of ARe-SQL. For each test we have executed the set of queries belonging to used workload once, twice, 4, 8, 16, and 32 times. For each different number of executions (iterations), the sum of execution time of the whole set of queries was computed.

4.2 Scenario 1: Full TPC-H Benchmark

Figures 7 and 8 bring the result of using the assisted and automatic approach of ARe-SQL running on PostgreSQL and SQL Server, respectively. It is important to observe that from the 23 queries that comprise the TPC-H benchmark, two (18 and 23) have been rewritten by ARe-SQL advisor (see Section 3.1.1). Regarding the results presented in Figures 7 and 8, the assisted approach had a small decrease in workload runtime. This can be explained by the fact that only two TPC-H queries have presented opportunities for tuning. Table II shows the execution time of TPC-H queries 18 and 23 in their original formats and after being rewritten.

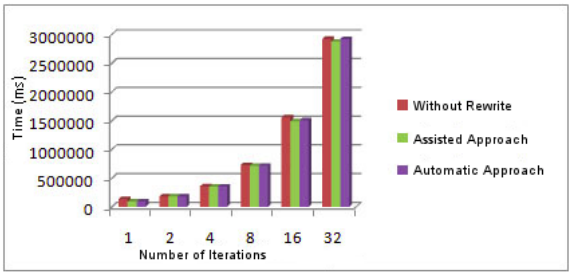


Fig. 7. Benchmark TPC-H on PostgreSQL

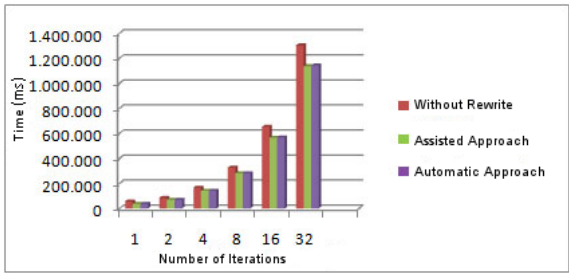


Fig. 8. Benchmark TPC-H on SQL Server

On Oracle, however, TPC-H queries 18 and 23 were not rewritten by ARe-SQL advisor, since Oracle implements heuristic H8. So, the ARe-SQL automatic approach had a slight worsening compared to baseline (24.95%). This is because the automatic approach had the overhead of trying to rewrite both queries.

4.3 Scenario 2: TPC-H Database with Synthetic Workload

Figures 9, 10 and 11 show that using ARe-SQL advisor ensures a significant reduction in query response. On the other hand, ARe-SQL's automatic approach presents smaller benefits than ARe-SQL advisor, since it involves the overhead of tuning the SQL statements received in runtime.

It's important to note that in the test on Oracle (Figure 11) automatic approach presents a small decrease in the execution time of the workload. This is explained by the fact that from the total of eleven heuristics five are already implemented by Oracle (H4, H5, H6, H7 and H8). Besides, from the six remaining heuristics, three of them (H9, H10 and H11) make use of statistical information, and, therefore, require additional access to DBMS, which significantly increased the overhead to rewrite SQL queries.

In the experiment with TPC-H Database and Synthetic Workload on Oracle, the runtime for the 32 iterations of the original workload (without rewriting) took 673.28s, while the runtime for the 32 iterations of the workload generated after applying the assisted approach lasted 49.43 seconds. In this case, assisted approach yielded a gain of 623 seconds (92.5%).

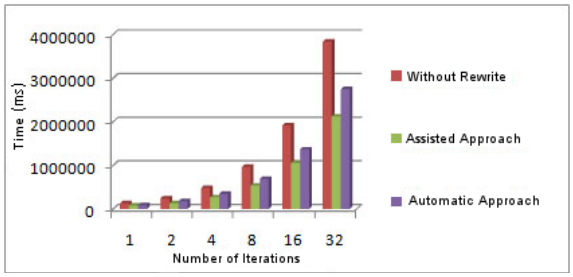


Fig. 9. Synthetic queries in TPC-H database on PostgreSQL

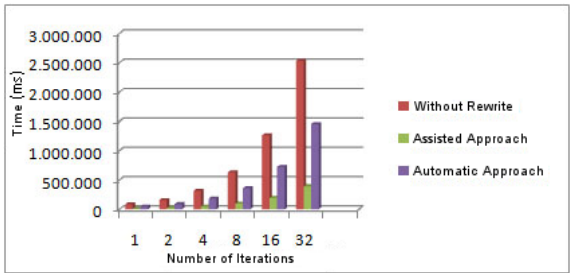


Fig. 10. Synthetic queries in TPC-H database on SQL Server

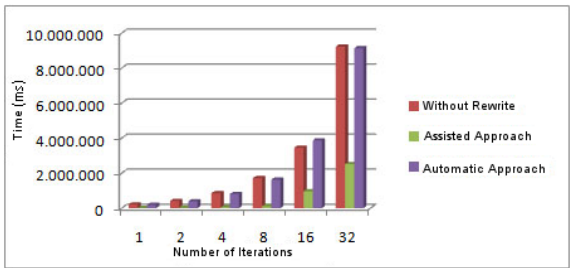


Fig. 11. Synthetic queries in TPC-H database on Oracle

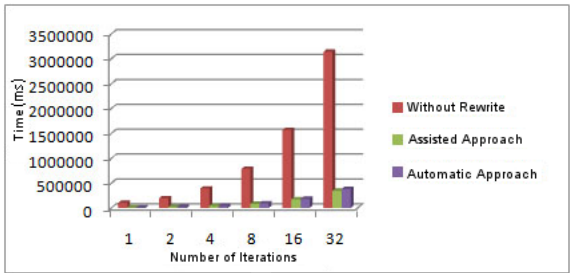


Fig. 12. Synthetic queries in IMS database on PostgreSQL.

4.4 Scenario 3: IMS Database with Synthetic Workload

For this scenario, only PostgreSQL has been used, as the IMS database is only available for that DBMS. Again, the proposed approaches have provided a high reduction in time execution of the submitted workload (Figure 12). In the experiment with IMS Database and Synthetic Workload on PostgreSQL, the runtime for the 32 iterations of the original workload (without rewriting) took 314.32s, while the runtime for the 32 iterations of the workload generated after applying the assisted approach lasted 34.72 seconds. In this case, assisted approach yielded a gain of 280 seconds (89%).

5. CONCLUSIONS

In this paper, we propose an automatic, autonomous and non-intrusive approach for rewriting SQL queries, denoted ARe-SQL. The key goal of ARe-SQL is to assist optimizers in building efficient query physical execution plan. For this, ARe-SQL rewrites SQL statements, using a set of 11 heuristics. Based on ARe-SQL, two different strategies for SQL-query rewriting, denoted assisted and automatic, were implemented. These strategies were evaluated in three different DBMSs considering three different scenarios. The experimental results indicate that both strategies can provide performance gains. In some cases, such performance gains reach 92.5%. The proposed solutions are applicable to situations where the query optimizer cannot produce optimal plans, even using access methods and assessment strategies supported by the DBMS.

REFERENCES

BRUNO, N., CHAUDHURI, S., AND RAMAMURTHY, R. Power hints for query optimization. In *ICDE '09 Proceedings of the 2009 IEEE International Conference on Data*. IEEE Computer Society, Washington, DC, USA, pp. 469–480, 2009.

DAGEVILLE, B. AND DIAS, K. Oracle's self-tuning architecture and solutions. In *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*. IEEE Computer Society, Oracle, USA, 2006.

GARCIA-MOLINA, H., ULLMAN, J., AND WIDOM, J. *Database System Implementation*. Prentice Hall, 2000.

OPTIMIZER, E. D. Embarcadero db optimizer xe, 2010. Available: http://www.embarcadero.com/products/db-optimizer-xe. June 2011.

QUEST, S. O. F. O. Quest sql optimizer for oracle, 2010. Available: http://www.quest.com/SQL-Optimizer-for-Oracle. June 2011.

RAMAKRISHNAN, R. AND GEHRKE, J. *Database Management Systems*. McGraw Hill, 2002.

STUDIO, I. O. D. Ibm optim development studio, 2010. Available: http://www-01.ibm.com/software/data/optim/development-studio. June 2011.

Implementing an Architecture for Semantic Search Systems for Retrieving Information in Biodiversity Repositories

Flor Karina Mamani Amanqui[1], Kleberson Junio Serique[1], Franco Lamping[1], José Laurindo Campos dos Santos[2], Andréa Corrêa Flôres Albuquerque[2], Dilvan de Abreu Moreira[1]

[1] University of São Paulo, Computer Science Dept, São Carlos, Brazil
{flork, serique, lamping, dilvan}@icmc.usp.br
[2] National Institute for Amazonian Research, Manaus, Brazil
{andreaa, lcampos}@inpa.gov.br

Abstract. Biological diversity is of essential value to life sustainability on Earth and motivates many efforts to collect data about species, giving rise to a large amount of information. Biodiversity data, in most cases, is stored in relational databases. Researchers use this data to extract knowledge and share their new discoveries about living things. However, nowadays the traditional search approach, based on keywords, is not appropriate to be used in large amounts of heterogeneous biodiversity data. In addition, the search by keyword has low precision and recall in this kind of data. In this paper, we present a novel architecture, for ontology based semantic search systems, and test results of a prototype system, implemented using state of the art free semantic web tools, using a set of representative data about biodiversity from INPA (consisting of specimens of fish and insects). This test results show that the prototype had better recall and precision than keyword based methods (for the same dataset). In the semantic web, ontologies allow knowledge to be organised into conceptual spaces in accordance to its meaning. For that reason, for semantic search to work, a key point is to create mappings between the data, stored in relational databases, and the ontologies describing this data. This work also developed such a mapping.

Categories and Subject Descriptors: H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

Keywords: Biodiversity, Ontology, Data Integration, Semantic Search

1. INTRODUCTION

The Web became one of the most common ways to get scientific data for the acquisition of new knowledge. However, this wide data availability generates large amounts of information in all areas of knowledge. Due to the huge size of the Web, it has become difficult for users to find the information they need.

In the biodiversity field, it is not different. There is a large quantity of online data generated by research institutions that have collections of specimens collected in Brazil. Efforts, like the SpeciesLink network or the Global Biodiversity Information Facility (GBIF)[1], integrate specie and specimen data available in natural history museums, herbaria and culture collections, making it openly and freely available on the Internet. They integrate organisations like the National Institute for Amazonian Researsh (INPA)[2], Pará's Emílio Goeldi Museum (MPEG)[3], the Amazon Biotechnology Center (CBA)[4],

[1]http://www.gbif.org/
[2]INPA http://www.inpa.gov.br
[3]http://www.museu-goeldi.br
[4]http://www.suframa.gov.br/cba/

the Reference Center for Environmental Information (CRIA)[5], and The New York Botanical Garden (NYGB)[6], among other important organisations for dissemination of biological data collections. That has led to discussions about the best way to arrange this data and provide tools and environments that stimulate and facilitate information sharing.

This proliferation of information from different sources means that the search for information could be met by a variety of available resources, which store data about the same domains but have different characteristics. Therefore, the need for integration and analysis of data from these sources becomes more evident.

For biodiversity data to be analysed and retrieved efficiently, it is necessary to use new technologies to improve human and computer collaboration on the Web. Interestingly, despite improvements in search engine technology, the difficulties remain essentially the same. It seems that the amount of Web content outpaces technological progress. Fortunately, Semantic Web technologies enable explicit declaration of knowledge embedded in many Web-based applications, integrating information in an intelligent way, providing semantic-based access to the Web.

Berners-Lee, Hendler and Lassila introduced the concept of the Semantic Web in 2001 and helped to stimulate innovative ideas and new technologies from different computing areas [Berners-Lee et al. 2001]. There are a number of important issues related to the Semantic Web: ontologies, languages for the Semantic Web, semantic search, semantic markup of pages and services.

In this work, we present an architecture for a Semantic Search system that supports mapping between the data (INPA's biodiversity data) stored in relational databases, and the biodiversity ontology (OntoBio). OntoBio was developed by INPA. Its main objective is to provide a clear and precise conceptualisation of the information presented in biodiversity data collections. The complete ontology is presented in details in [Albuquerque 2011].

Futhermore, the mapping was implemented using free state of the art Semantic Web tools and tested on a set of representative data about biodiversity from INPA. The contributions of this work are as follows: (i)A proposal for a new Semantic Search Architecture for biodiversity data; (ii)A mapping between the INPA's biodiversity data (stored in relational databases), and the ontology OntoBio from INPA and (iii) finally, experimental results on collected data from INPA showing improvements in precision and recall when compared to keyword based methods.

The remainder of this article proceeds as follows: Section 2 presents our Semantic Search Architecture. Section 3 presents the experiments and evaluations. Section 4 describes related works, and in Section 5 the conclusions, discussions and future works are presented.

2. RELATED WORK

This section focuses on the analysis of search mechanisms for integrating information about biodiversity. In Brazil, there are three important web tools that manage biological collections: SpeciesLink[7], Specify[8] and SinBiota[9]. In our approach, we have analyzed these systems to understand the needs and requirements of biodiversity experts. SpeciesLink is used for the comparison between keyword based search with semantic based search because it stores information from INPA collections.

In the context of semantic web technologies applied to biodiversity data, there is a large number of projects implemented and published that supports research on biodiversity conservation on the Web. A number of techniques have been developed for use ontologies to retrieve relevant documents

[5]http://www.cria.org.br/
[6]http://www.nybg.org
[7]http://splink.cria.org.br/
[8]http://informatics.biodiversity.ku.edu/
[9]http://sinbiota.cria.org.br/

in response to a query as [Kara et al. 2012], [Freitas et al. 2012], [de S. Fedel et al. 2012], [dos Santos et al. 2011]. These systems use relational databases to store the biodiversity data and ontologies. However, none of the work focused on the problem of storage and retrieval of Resource Description Framework (RDF)[10] triples. Also, an additional limitation in many of the existing approaches is the lack of an an evaluation of the quality of results.

3. ARCHITECTURE FOR SEMANTIC SEARCH

The Semantic Search Architecture for the biodiversity domain created by us is shown in Figure 1 and its components are detailed next.

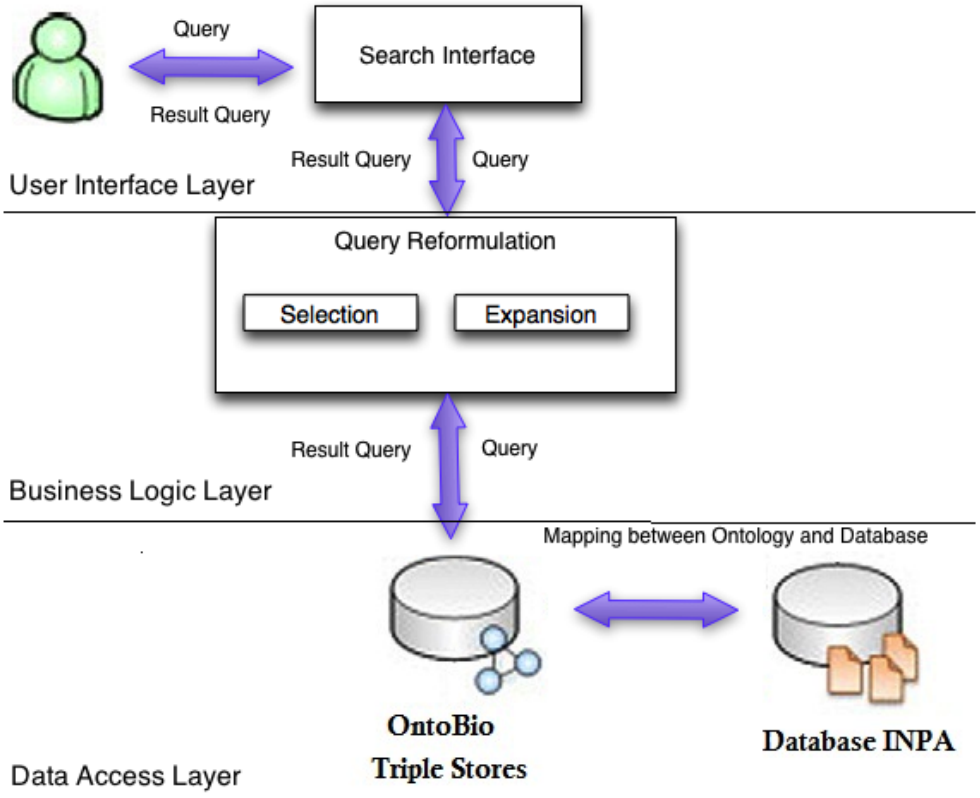


Fig. 1. Architecture for Semantic Search

The User Interface layer is responsible for the interaction between user and system. The search process begins with an initial keyword list, entered by the biodiversity specialist, which represents his/her search intentions.

The Business logic layer processes the information retrieval. Its components are detailed as follow.

The Query Reformulation Component receives the input of search terms from the user, selects and expands keyword lists by adding semantically related terms, using the techniques of expansion

[10]http://www.w3.org/RDF/

and semantic similarity. Querying the semantic data is simplified because of the use of triples. These triples are generated when the database records are mapped into ontology concepts.

The Data Access layer provides access to the RDF triples stored in the Triple Store (in our prototype, Virtuoso) for the layers above and also to machine clients in the web through a SPARQL Endpoint.

The Mapping Component The mapping of the database records into RDF triples is done offline. It generates the triples that will be stored in the triple store (in our case, Virtuoso) and queried during user searches. The Mapping Component loads the domain ontologies, taxonomic information and the collection database and transforms them in a set of RDF triples. We used Ontop, a platform to query databases as Virtual RDF Graphs using SPARQL, to do the mapping between the relational databases records and the OWL ontologies. Ontop has two tools: (i)**OntopPro**, which is a Protege 4 plugin that implements a graphic mapping editor to allow the definition of data sources (databases) for ontologies and the mapping of records, from this data sources, into ontology entities using an intuitive mapping language [Mariano R and Calvanese 2012]. (ii)**Quest**, which is a SPARQL query engine/reasoner that supports RDFS and OWL 2 QL entailment regimes and SPARQL-to-SQL query rewriting. Using OntopPro, it is possible to export the RDF triples generated by the Quest tool to a file.

3.1 Semantic Search Algorithm

The basic idea of our algorithm is to compare input keyword with OntoBio resources (subject, predicate and object) in Virtuoso triple store. Virtuoso supports the SPARQL query language and provides a faceted browser user interface for querying the RDF store. The use of triple stores makes our semantic search architecture more versatile due to the fact that they allow great volumes of data and work directly with the SPARQL language. The algorithm used is shown below:

```
Algorithm:  SemanticSearch (String Query)
Step 1:  Establish connection with Virtuoso Triple Store and
OntoBio ontology
Step 2:  Extract OntoBio information (hierarchy)
Step 3:  For each Query in OntoBio
(a) Construct SPARQL query with Query as object, subject or
predicate
(b) Calculate similarity of Query with subject, object and
predicate from OntoBio
Step 4:  Go to Step 3 until no string is left in Triples OntoBio
or no more text are to be considered
Step 5:  Arrangue the results according to order of similarity
Step 6:  Send and display the results
```

4. EXPERIMENTS

We implement a prototype using Java, Google Web Toolkit 2.5.1 (to create a web application), Jena RDF framework (to process SPARQL queries) and Virtuoso Server, as triple store. In order to validate our architecture, researchers from our group and biodiversity scientists (from INPA) were interviewed to categorise important information from the INPA dataset (e.g. genus, family, specie, location description). We then defined use cases with features and scenarios to identify important user tasks. These use cases are:

USE CASE 01: Statistics relating to fishing and use of resources

INITIATOR: Mary Smith, Biologist, 29 years-old **GOAL:** To determine the best areas for aqua-

culture development, considering different types of species and time of year. Aquaculture is an activity ecologically sustainable that generates income and can propel the economy of a region. **BEST SCENARIO** Mary is working in her laboratory and needs to search fish species suitable for aquaculture in the Amazon. She needs to specify the name of the fish species she is interested and determine the fish size, viable habitats, municipalities (where these habitats are present).

USE CASE 02: Environmental impact study

INITIATOR: John Rubin, Environmental Engineer, 37 years. **GOAL:** To determine if specimens sampled in a swamp are endemic or cosmopolitan. **BEST SCENARIO** To discover, in all collections available, the records of specimens belonging to the same species of the ones found in the swamp; find out the geographical location and the habitat of each collected specimen and then determine if all species found in the swamp are found in different habitats and/or distributed geographic locations.

Using to the previous use cases, we identified the user requirements (for each case) and created samples of the kind of queries they would do. Using just fish and insect data we tested this queries using keyword and semantic search. Experts in biodiversity identified, in each query, what they judged as relevant and non relevant (precision) and if the query returned all relevant data (recall).

We compared the result returned by two search system: our semantic search prototype and the keyword based search from the SpeciesLink [11] tool with data from INPA. The efficiency of the result for each approach was evaluated using the average precision and recall. This evaluation is shown in Figure 2. We used a total of 14 queries (7 for each system). We would like to have done more queries, but the analysis of each query represents a lot of work for the biodiversity experts from INPA, they have to determine, from a big dataset, which records are relevant or not for each query to analyse the quality of the query results. Unfortunately, the time these experts had to dedicate to our project was very limited.

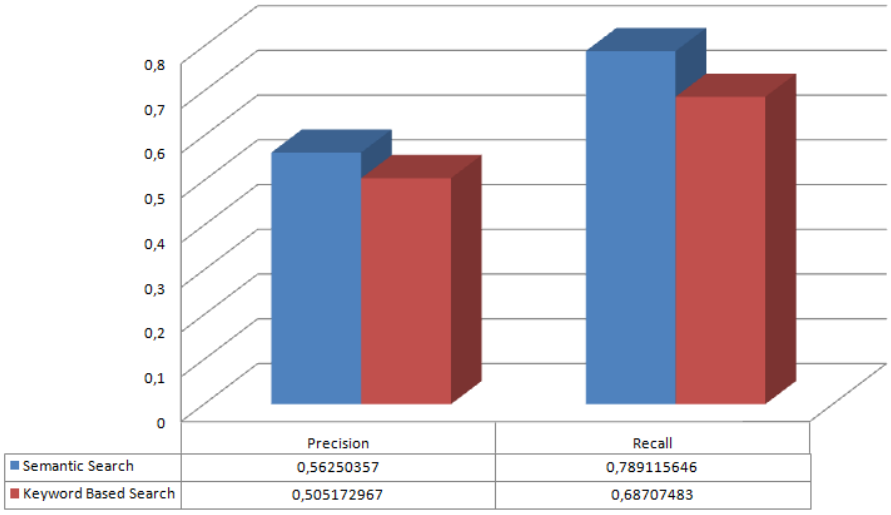


Fig. 2. Evaluation of semantic search with keyword based search

The semantic search architecture had the highest precision (on average), 11.3 % better than using the keyword search. The semantic search recall (on average) was 14.8% better. These results show the advantage of the semantic approach. It is important to note that this tests do not cover cases where

[11] http://www.splink.org.br/

queries need information from different data sources, something almost impossible to do automatically with web based keyword search systems and very easily done using semantic search in SPARQL Endpoints. Figure 3 shows graphic interface to support the user queries.

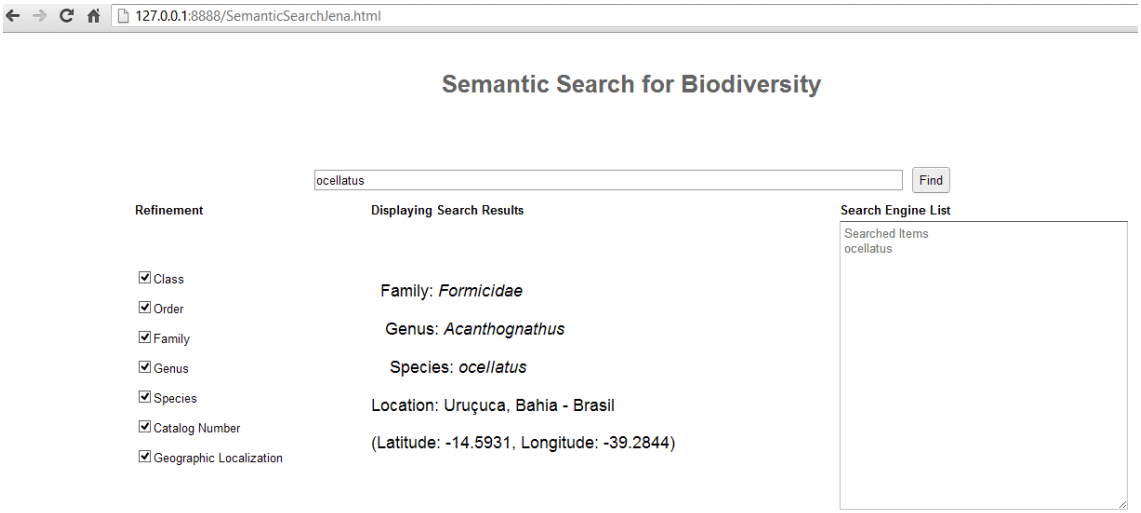


Fig. 3. Screen copy of the prototype using mapping implementations

5. CONCLUSIONS AND FUTURE WORKS

In this article we presented a semantic search architecture that provides a new document retrieval process by exploring a domain knowledge (OntoBio) and semantic search to support scientists in the process of discovery and integration biodiversity data. Our goals were to specify and developed semantic search approaches that allow biodiversity researchers to easily find and access relevant data sources. These approaches can improve precision and recall compared to the existing keyword-based queries offered by tools such as SpeciesLink using the same data from INPA. As future work, we plan to refine the results from our use cases. We intend to reuse geoSPARQL ontology terms to describe georreferenced data. We also intend to extend our current implementation with more advanced structured searches in collaboration with researches from INPA.

The authors would like to thank CNPq for funding this work.

REFERENCES

Albuquerque, A. *Desenvolvimento de uma Ontologia de Domínio para Modelagem de Biodiversidade*. Dissertação de Mestrado. Universidade Federal do Amazonas, 2011.

Berners-Lee, T., Hendler, J., and Lassila, O. The semantic web. *Scientific American* 284 (5): 34–43, May, 2001.

de S. Fedel, G., Medeiros, C. B., and dos Santos, J. A. Sinimbu- multimodal queries to support biodiversity studies. In *Proceedings of the 12th international conference on Computational Science and Its Applications - Volume Part I*. ICCSA12. Springer-Verlag, Berlin, Heidelberg, pp. 620–634, 2012.

dos Santos, V., Baiao, F., and Tanaka, A. An architecture to support information sources discovery through semantic search. pp. 276 –282, 2011.

Freitas, A., Curry, E., and O'Riain, S. A Distributional Approach for Terminology-Level Semantic Search on the Linked Data Web. In *27th ACM Symposium On Applied Computing (SAC 2012)*. ACM Press, 2012.

Kara, S., Alan, O., Sabuncu, O., Akpnar, S., Cicekli, N. K., and Alpaslan, F. N. An ontology-based retrieval system using semantic indexing. *Information Systems* 37 (4): 294–305, June, 2012.

Mariano R, M. and Calvanese, D. *Quest, an OWL 2 QL Reasoner for Ontology-Based Data Access*. KRDB Research Centre for Knowledge and Data, Free University of Bozen-Bolzano, Bolzano, Italy, 2012.

K-medoids LSH: a new locality sensitive hashing in general metric space

Eliezer S. Silva, Eduardo Valle

RECOD Lab – DCA/FEEC, University of Campinas, Brazil
{eliezers, dovalle}@dca.fee.unicamp.br

Abstract. The increasing availability of multimedia content poses a challenge for information retrieval researchers. Users want not only have access to multimedia documents, but also make sense of them. The ability of finding specific content in extremely large collections of textual and non-textual documents is paramount. At such large scales, Multimedia Information Retrieval systems must rely on the ability to perform search by similarity efficiently. However, Multimedia Documents are often represented by high-dimensional feature vectors, or by other complex representations in metric spaces. Providing efficient similarity search for that kind of data is extremely challenging. In this article, we explore one of the most cited family of solutions for similarity search, the Locality-Sensitive Hashing (LSH), which is based upon the creation of hashing functions which assign, with higher probability, the same key for data that are similar. LSH is available only for a handful distance functions, but, where available, it has been found to be extremely efficient for architectures with uniform access cost to the data. Most of existing LSH functions are restricted to vector spaces. We propose a novel LSH method for generic metric space based on K-medoids clustering. We present comparison with well established LSH methods in vector spaces and with recent competing new methods for metric spaces. Our early results show promise, but also demonstrate how challenging is to work around those difficulties.

Categories and Subject Descriptors: H.2 [**Database Management**]: Miscellaneous; H.3.1 [**Content Analysis and Indexing**]: Indexing methods; H.3.3 [**Information Search and Retrieval**]: Miscellaneous

Keywords: hashing, metric space indexing, nearest neighbor search, similarity search

1. INTRODUCTION

Content-based Multimedia Information Retrieval (CMIR) is an alternative to keyword-based or tag-based retrieval, which works by extracting *features* based on distinctive properties of the multimedia objects. Those features are organized in *multimedia descriptors*, which are used as surrogates of the multimedia object, in such a way that the retrieval of similar objects is based solely on that higher level representation, without the need to refer to the actual low-level encoding of the media. The descriptor can be seen as a compact and distinctive representation of multimedia content, encoding some invariant properties of the content. For example, in image retrieval the successful *Scale Invariant Feature Transform* (SIFT) [Lowe 2004] encodes local gradient patterns around Points-of-Interest, in a way that is (partially) invariant to illumination and geometric transformations.

The descriptor framework also allows to abstract the media details in multimedia retrieval systems. The operation of looking for similar multimedia documents becomes the more abstract operation of looking for multimedia descriptors which have small distances. The notion of a "feature space", that organizes the documents in a geometry, putting close-by those that are similar emerges. Of course, CMIR systems are usually much more complex than that, but nevertheless, looking for similar descriptors often plays a critical role in the processing chain of a complex system.

Although the operation of finding descriptors which have small distances seems simple enough,

E.S. Silva is supported by a CAPES/PROEX grant.

performing it fast for multimedia descriptors is actually very challenging, due to the scale of the collections, the dimensionality of the descriptors and the diversity of distance functions [Akune et al. 2010]. The literature on the subject is very extensive, but in this article, we focus on one of the most cited family of solutions, the *Locality-Sensitive Hashing* (LSH) [Indyk and Motwani 1998; Gionis et al. 1999; Datar et al. 2004], proposing *K-medoids LSH* as an extension to metric space.

2. LOCALITY SENSITIVE HASHING (LSH)

The LSH indexing method relies on a family of locality-sensitive hashing function H, [Indyk and Motwani 1998], to map objects from a metric domain U in a D-dimensional space (usually $\mathbb{R}^d$) to a countable set C (usually $\mathbb{Z}$), with the following property: nearby points in high dimensional space are hashed to the same value with high probability.

DEFINITION 1. *Given a distance function $d : U \times U \to \mathbb{R}^+$, a function family $H = \{h : U \to C\}$ is (r, cr, p_1, p_2)-sensitive for a given data set $S \subseteq U$ if, for any points $p, q \in S$, $h \in H$:*

—If $d(p, q) \leq r$ then $Pr_H[h(q) = h(p)] \geq p_1$ (probability of colliding within the ball of radius r),
—If $d(p, q) > cr$ then $Pr_H[h(q) = h(p)] \leq p_2$ (probability of colliding outside the ball of radius cr)
—$c > 1$ and $p_1 > p_2$

The basic scheme [Indyk and Motwani 1998] provided locality-sensitive families for the Hamming distance on Hamming spaces, and the Jacquard distance in spaces of sets. For several years those were the only families available, although extensions for L_1-normed (Manhattan) and L_2-normed (Euclidean) spaces were proposed by embedding those spaces into a Hamming space [Gionis et al. 1999].

The practical success of LSH, however, came with the E2LSH[1] (Euclidean LSH) [Datar et al. 2004], for L_p-normed space, where a new familiy of LSH functions was introduced:

$$H = \{h_i : \mathbb{R}^D \to \mathbb{Z}\} \tag{1}$$

$$h_i(\mathbf{v}) = \left\lfloor \frac{\mathbf{a}_i.\mathbf{v} + b_i}{w} \right\rfloor \tag{2}$$

$\mathbf{a}_i \in \mathbb{R}^D$ is a random vector with each coordinate picked independently from a Gaussian distribution $N(0, 1)$, b_i is an offset value sampled from uniform distribution in the range $[0, \ldots, w]$ and w is a scalar for the quantization width. Applying h_i to a point or object $\mathbf{v}$ corresponds to the composition of a projection to a random line and a quantization operation, given by the quantization width w and the floor operation.

A function family G is constructed by concatenating M randomly sampled functions $h_i \in H$, such that each $g_j \in G$ has the following form: $g_j(\mathbf{v}) = (h_1(\mathbf{v}), ..., h_M(\mathbf{v}))$. The use of multiple h_i functions reduces the probability of false positives, since two objects will have the same key for g_j only if their value coincide for all h_i component functions. Each object $\mathbf{v}$ from the input dataset is indexed by hashing it against L hash functions $g_1, \ldots, g_L$. At the *search phase* a query object $\mathbf{q}$ is hashed using the same L hash functions and the objects stored in the given buckets are used as the candidate set. Then, a ranking is performed among the candidate set according to their distance to the query, and the k closest objects are returned.

A few recent works approach the problem of designing LSH indexing schemes using only the distance information: M-Index [Novak et al. 2010], DFLSH (Distribution Free Locality-Sensitive Hashing) [Kang and Jung 2012] and [Tellez and Chavez 2010].

[1] *LSH Algorithm and Implementation (E2LSH)*, accessed in 22/09/2013. `http://www.mit.edu/~andoni/LSH/`

K-means LSH. [Paulevé et al. 2010] present a comparison between *structured* (regular random projections, lattice) and *unstructured* (K-means and hierarchical K-means) quantizers in the task of searching high dimensional SIFT descriptors, resulting on the proposal of a new LSH family based on the latter (Equation 3). Results indicate that the LSH functions based on unstructured quantizers perform better, as the induced Voronoi partitioning adapts to the data distribution, generating more uniform hash cells population than the structured LSH quantizers. However, K-means and hierarchical K-means are clustering algorithms for vector spaces, restricting the application of this approach. In order to overcome this limitation we turn to a clustering algorithm designed to work in generic metric spaces – *K-medoids clustering*.

3. K-MEDOIDS LSH

We propose a novel method for locality-sensitive hashing in the metric search framework and compare them with other similar methods in the literature. Our method is rooted on the idea of partitioning the data space with a distance-based clustering algorithm (K-medoids) as an initial quantization step and constructing a hash table with the quantization results. *K-medoids LSH* follows a direct approach taken from [Paulevé et al. 2010]: each partition cell is a hash table bucket, therefore the hashing is the index number of the partition cell.

Definition 2. *Given a metric space (U, d) (U is the domain set and d is the distance function), the set of cluster centers $C = \{c_1, \ldots, c_k\} \subset U$ and an object $x \in U$:*

$$\begin{gathered} h_C : U \to \mathbb{N} \\ h_C(x) = \text{argmin}_{i=1,\ldots,k}\{d(x, c_1), \ldots, d(x, c_i), \ldots, d(x, c_k)\} \end{gathered} \tag{3}$$

PAM (Partitioning Around Medoids) [Kaufman and Rousseeuw 1990] is the classic algorithm for K-medoids clustering. In that work a medoid is defined as the most centrally located object within a cluster. The algorithm initially selects a group of medoids at random (or using some heuristics), then iteratively assigns each non-medoid point to the nearest medoid and update the medoid set, looking for the optimal set of medoid points minimizing the quadratic sum of distance. PAM features a prohibitive computational cost, since it performs $O(k(n-k)^2)$ distance calculation for each iteration. There are a few methods designed to cope with that complexity constraint. Park and Jun [Park and Jun 2009] propose a simple approximate algorithm based on PAM with significant performance improvements. Instead of searching the entire dataset for a new optimal medoid, the method restricts the search for points within the cluster. We choose to apply this method for its simplicity and performance.

Therefore, our baseline method consists in the following steps:

(1) Generate L lists of k cluster centers ($C_i = \{c_{i1}, \ldots, c_{ik}\}, \forall i \in \{1, \ldots, L\}$) in L runnings of the Park and Jun fast k-medoid algorithm (as in In Paulevé *et al.* [Paulevé et al. 2010], this clustering is done over a sampling of the dataset).
(2) Index each point x of the dataset using $h_{C_i}(x)$ as the bucket key for each table ($i \in \{1, \ldots, L\}$).
(3) Given a query point q, hash it using $h_{C_i}(q)$ ($i \in \{1, \ldots, L\}$) and retrieve all colliding points in a candidate set of kNN[2] points. Then perform a linear scan over that candidate list.

Initialization. K-means clustering results exhibit a strong dependence on the initial cluster selection. K-means++ [Ostrovsky et al. 2006; Arthur and Vassilvitskii 2007] solve the $O(\log k)$ approximate K-means problem[3] simple by carefully designing a good initialization algorithms. That raises the question of how the initialization could affect the final result of the kNN search for the proposed methods. The

[2]kNN - k nearest neighbors
[3]the exact solution is NP-Hard

K-means++ initialization method is based on metric distance information and sampling, thus can be plugged into a K-medoids algorithm without further changes. [Park and Jun 2009] propose also a special initialization for the fast K-medoids algorithm. We implemented both of those initialization, and the random selection as well and empirically evaluated their impact on the similarity search task.

3.1 Experiments and analysis

Datasets:. for the initial experiments we resorted to a dataset called APM (*Arquivo Público Mineiro* – The Public Archives in Minas Gerais) which was created from the application of various transformation (15 transformation of scale, rotation, etc) to 100 antique photographs of the Public Archives. The SIFT descriptors of each transformed image were calculated, resulting in a dataset of 2.871.300 feature vectors (SIFT descriptor is a 128 dimensional vector). The queries dataset is build from the SIFT descriptors of the original images (a total of 263.968 feature vectors). Each query point is equiped with its set of true nearest neighbors – the ground-truth. For these experiments we used 500 points uniformly sampled from the query dataset and performed a 10-NN search.

Metrics:. We are concerned especially with the relations between the recall and selectivity metric given the variation in the parametric space of the methods. The *recall* metric is the fraction of total correct answers over the number of true relevant answers. The *selectivity*[4] metric is the fraction of the dataset selected for the shortlist processing. Selectivity is an important metric for those methods being considered, since the shortlist processing is a bottleneck in the whole query processing [Paulevé et al. 2010].

K-means LSH, K-medoids LSH and DFLSH:. In order to evaluate the feasibility of K-medoids LSH we compared it with the K-means LSH *et al.* [Paulevé et al. 2010] and the DFLSH [Kang and Jung 2012]. We implemented all the methods in the same framework, namely Java and *The Apache Commons^{TM} Mathematics Library*[5]. K-means LSH is used as a baseline method, since its performance and behavior are well studied and presented in the literature and DFLSH is as an alternative method for the same problem. Figure 1 shows recall, query time and selectivity statistics averaged over 500 queries and using a single hash function[6].

Figure 1a presents results relating recall with selectivity. Notice that high selectivity means that a large part of the dataset is being processed in the final sequential ranking – that is not a desirable point of operation, since the main performance bottleneck on the query processing time is located at that stage(1b). For a selectivity as low as 0.3%, both methods' recall metric is approximately 65% and with 1% selectivity it grows up to a 80%. These results could be improved by using more than one hash function. Clearly K-means LSH presents the best result on the recall metric, however it is important to notice that both DFLSH and K-medoids LSH are not exploiting any vector coordinate properties, relying solely on distance information.

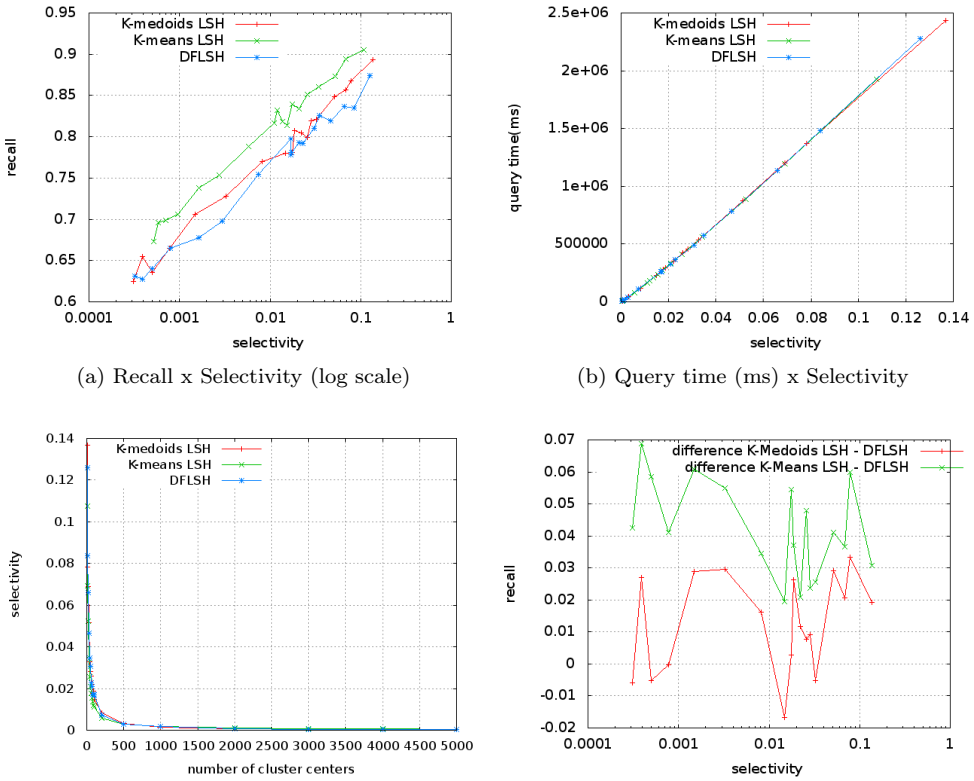
Figure 1b depicts a strong linear correlation between query time and selectivity. Other noticeable strong correlation appears between selectivity and number of cluster centers (Figure 1c). Theoretically it is possible to show that, given an approximate uniform population of points in the hash buckets, the selectivity for a t number of cluster centers is $O(t^{-1})$. The plot shows that the experimental data follows a power law curve.

---

[4]the term selectivity has distinct use in the database and image retrieval research communities. Here we adopt the latter, following *Paulève et al.*

[5]*Commons Math: The Apache Commons Mathematics Library*, accessed in 22/09/2013. `http://commons.apache.org/proper/commons-math/`

[6]Including more than one hash function would have approximately the same effect in the recall and querying time metric for all methods, nevertheless the preprocessing time for K-means LSH and K-medoids LSH (envolving extra rounds of the optimization procedure) would increase much more then for DFLSH (sampling more points)



(a) Recall x Selectivity (log scale)

(b) Query time (ms) x Selectivity

(c) Selectivity x Number of cluster centers

(d) Difference of Recall x Selectivity (log scale)

Fig. 1: Comparison of K-means LSH, K-medoids LSH and DFLSH for 10-NN

Using DFLSH as a baseline, we plot the difference on the recall from K-means LSH and K-medoids LSH to DFLSH (Figure 1d. The recall difference can be up to 0.07 positive (11%) for K-means LSH and 0.03 (5%) for K-medoids LSH. The trend on the differences is sustained along the curves for different values of selectivity. There is a clear indication that K-medoids LSH is a viable approach with equivalent performance to K-means LSH and DFLSH.

Initialization effect:. First of all it is important to notice that the initial segment of the curves in Figure 2 are not much informative – those points corresponds to number of clusters up to 20, implying in an almost brute-force search over the dataset (notice the explosion on the selectivity in Figure 1c). Figure 2 depicts the difference on the recall metric for the two initialization algorithm, using the random selection as a baseline. The K-means++ initialization affects positively the results with average gain of 3%. In the other hand the Park and Jun initialization does not presents great gain over the random baseline (in fact, the average gain is negative). Further experiments with more datasets are needed in order to achieve statistical confidence on these results, but the initial analysis indicates that the K-means++ initialization can contribute to better results in the recall metric, while the Park and Jun method presents a null effect (or negative).

4. CONCLUSION

Efficient large-scale similarity search is a crucial operation for Content-based Multimedia Information Retrieval (CMIR) systems. But because those systems employ high-dimensional feature vectors, or other complex representations in metric spaces, providing fast similarity search for them has been a persistent research challenge. LSH, a very successful family of methods, has been advanced as a

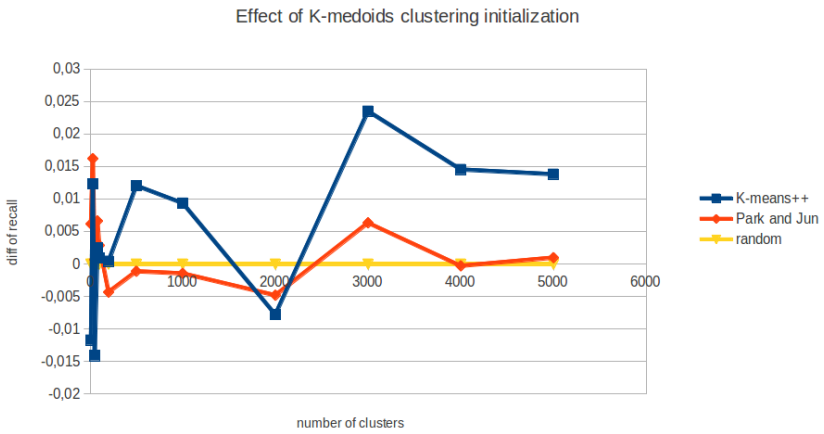


Fig. 2: Effect of the initialization procedure for the K-medoids clustering in the quality of nearest-neighbors search

solution to the problem, but it is available only for a few distance functions. In this article we propose to address that limitation, by extending LSH to general metric spaces, using K-medoids clustering as basis for a LSH family of functions. We show in our experiments that the K-medoids LSH improves the results over the random choice of sample of DFLSH, while keeping the advantage of relying only on distance information. As expected, K-medoids LSH performance is slightly worse than K-means LSH, but it is important to note that K-means relies heavily on the vector-space structure, and many data types of interest do not offer such structure.

REFERENCES

Akune, F., Valle, E., and Torres, R. MONORAIL: A Disk-Friendly Index for Huge Descriptor Databases. In *2010 20th International Conference on Pattern Recognition*. IEEE, pp. 4145–4148, 2010.

Arthur, D. and Vassilvitskii, S. k-means++: the advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. SODA '07. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, pp. 1027–1035, 2007.

Datar, M., Immorlica, N., Indyk, P., and Mirrokni, V. S. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the twentieth annual symposium on Computational geometry - SCG '04*. ACM Press, New York, New York, USA, pp. 253, 2004.

Gionis, A., Indyk, P., and Motwani, R. Similarity search in high dimensions via hashing. In *Proceedings of the 25th International Conference on Very Large Data Bases*. VLDB '99. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 518–529, 1999.

Indyk, P. and Motwani, R. Approximate nearest neighbors. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing - STOC '98*. ACM Press, New York, New York, USA, pp. 604–613, 1998.

Kang, B. and Jung, K. Robust and Efficient Locality Sensitive Hashing for Nearest Neighbor Search in Large Data Sets. In *NIPS Workshop on Big Learning (BigLearn)*. Lake Tahoe, Nevada, pp. 1–8, 2012.

Kaufman, L. and Rousseeuw, P. J. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley-Interscience, 1990.

Lowe, D. G. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision* 60 (2): 91–110, Nov., 2004.

Novak, D., Kyselak, M., and Zezula, P. On locality-sensitive indexing in generic metric spaces. *Proceedings of the Third International Conference on SImilarity Search and APplications - SISAP '10*, 2010.

Ostrovsky, R., Rabani, Y., Schulman, L., and Swamy, C. The Effectiveness of Lloyd-Type Methods for the k-Means Problem. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*. Vol. 59. IEEE, pp. 165–176, 2006.

Park, H.-S. and Jun, C.-H. A simple and fast algorithm for K-medoids clustering. *Expert Systems with Applications* 36 (2): 3336–3341, 2009.

Paulevé, L., Jégou, H., and Amsaleg, L. Locality sensitive hashing: A comparison of hash function types and querying mechanisms. *Pattern Recognition Letters* 31 (11): 1348–1358, Aug., 2010.

Tellez, E. S. and Chavez, E. On locality sensitive hashing in metric spaces. In *Proceedings of the Third International Conference on SImilarity Search and APplications*. SISAP '10. ACM, New York, NY, USA, pp. 67–74, 2010.

A Machine Learning Approach for SQL Queries Response Time Estimation in the Cloud

Victor A. E. Farias, José G. R. Maia, Flávio R. C. Sousa
Leonardo O. Moreira, Gustavo A. C. Campos, Javam C. Machado

Universidade Federal do Ceará (UFC), Brazil
{gilvanm, sousa, leoomoreira, gsantos, javam}@ufc.br, victorfarias@lia.ufc.br

Abstract. Cloud computing provides on-demand services with pay-as-you-go model. In this sense, customers expect quality from providers where QoS control over DBMSs services demands decision making, usually based on performance aspects. It is essential that the system be able to assimilate the characteristics of the application in order to determine the execution time for its queries' execution. Machine learning techniques are a promising option in this context, since they offer a wide range of algorithms that build predictive models for this purpose based on the past observations of the demand and on the DBMS system behavior. In this work, we propose and evaluate the use of regression methods for modeling the SQL query execution time in the cloud. This process uses different techniques, such as bag-of-words, statistical filter and genetic algorithms. We use K-Fold Cross-Validation for measuring the quality of the models in each step. Experiments were carried out based on data from database systems in the cloud generated by the TPC-C benchmark.

Categories and Subject Descriptors: H.Information Systems [**H.m. Miscellaneous**]: Databases

Keywords: Machine Learning, QoS, Cloud Computing

1. INTRODUCTION

Cloud computing is an extremely successful paradigm of service-oriented computing and has revolutionized the way computing infrastructure is abstracted and used. Scalability, elasticity, pay-per-use pricing, and economies of scale are the major reasons for the successful and widespread adoption of cloud infrastructures. Since the majority of cloud applications are data-driven, database management systems (DBMSs) powering these applications are critical components in the cloud software stack [Elmore et al. 2011].

Many companies expect cloud providers to guarantee quality of service (QoS) using service level agreement (SLAs) based on performance aspects. It is crucial that providers offer SLAs based on performance to their customers. Thus, it is important that quality is applied to the DBMS to control the consumed time [Sousa et al. 2012]. Controlling the QoS in DBMSs often needs an estimation for time and resources required to perform a query. Thereby, knowledge about the execution time of an incoming query provides a valuable variable for decision making in the cloud.

In this work, we propose and evaluate the use of regression methods for modeling the SQL query execution time in the cloud. This process is carried out as follows. First, tokens from SQL statement samples are represented using a Bag-of-Words (BoW). Then model selection is performed along with parameter tuning in order to improve accuracy of the regression method. At this point, we perform feature selection using both a statistical filter and a genetic algorithm to reduce dimensionality of input vectors. Since each sample contains a myriad of features from BoW, K-Fold Cross-Validation is used for measuring model quality all along this process. Experiments were carried out in order to evaluate our approach based on data from database systems in the cloud generated by the benchmark TPC-C [TPC 2013]. To the best of our knowledge, this is the first paper to formulate and explore the problem of how to work the bag-of-word of SQL queries with the goal of estimating response time for queries.

This paper is organized as follows: Section 2 explains theoretical aspects of this paper. The proposed approach is detailed in Section 3. The implementation and evaluation of our solution is described in Section 4. Section 5 surveys related work and, finally, Section 6 presents the conclusions.

2. BACKGROUND

- BoW is a widely used simple model for text processing. Given a set of text samples, BoW first extracts each word (token) from all samples, thus building a "dictionary". Hence, a specific text is represented as a "token histogram", i.e., by a vector whose each component corresponds to the frequency of a given token in the dictionary for that text;
- Genetic algorithm is an optimization technique for finding approximate solutions inspired by the evolutionary theory. Such technique is typically used when the search space is very large, perhaps infinite. Evolution theory simulation is carried out over a population of individuals based on natural selection, inheritance, crossover and mutation operations. In this context, each individual represents a solution for the problem that is evaluated using a known fitness function;
- K-fold cross-validation (K-fold CV) provides a measure of accuracy for a particular model from a dataset. The generalization capacity of a model is evaluated by partitioning the dataset into k mutually exclusive sets with equal sizes. Thereby, each partition is used as test set and the other k-1 partitions are used as training set. A model is trained and evaluated for each of these k settings. Finally, the mean of errors found in each setting is used as the accuracy measure;
- For regression tasks, we used three supervised learning algorithms: (1) Gradient Boosting Regression (GBR) [Friedman 2002] [Friedman 2000] builds a set of weak learners, one by one, and each predictor is built using the residual of the last predictor built, and this error is optimized by using gradient descent algorithm on the differentiable loss function; (2) Random Forest Regression (RFR) [Breiman 2001] [Breiman 1998], on its turn, builds simple regression trees over random subsets of the dataset in order to generate a more complex predictor. Thus, the RFR analyses the contribution of each simple predictor to compute the final result; and (3) Support Vector Regression (ϵ-SVR) [Vapnik 1998] in an adaptation of the Support Vector Machine method, which was developed as a binary classifier, for dealing with regression based on the ϵ-loss function foundation. It makes an ϵ-tube around the training samples so that the samples within the ϵ-tube are not counted as errors, while samples outside of the ϵ-tube become support vectors that will be used for test;
- F-test is a statistical test, in which f-distribution is used. In this context, f-distribution is used as the null distribution, assuming the null hypothesis is true, i.e., there is no relationship between two measured phenomena. Disproving such hypothesis implies that a given model is well suited for explaining a dataset. The null hypothesis is that two independent normal variances are equal, and the observed sums of some appropriately selected squares are then examined to see whether their ratio is significantly incompatible with this null hypothesis. The p-value is the probability of rejecting the null hypothesis when it is true.

3. OUR APPROACH

The proposed approach builds on top of different techniques to address the predictive query time execution problem before its execution. This problem can be treated as a machine learning problem, in particular, as a regression problem, since the phenomenon to be estimated can be described as a continuous variable. Regression models have the ability to model a target function by means of the application of an algorithm on a training set. In our context, such model is able to estimate the execution time of a query that has not yet been submitted to the system. Regression models demand the application of a feature extraction algorithm, to generate feature vectors from an SQL query.

In this work, we focused on analyzing only the text of the SQL query. Such approach can generate less accurate models, but in general, the process of obtaining the vectors is less costly than an approach

using data from the catalogs of the DBMS, because there is an additional cost of I/O. Thus, SQL statements issued to the DBMS and their corresponding execution times are used for building feature vectors comprising the dataset. An overview of our approach is depicted in Figure 1. The Feature Extraction (A) phase consists in extracting feature vectors from a mass of SQL data. For this purpose, we used the technique of vector representation BoW.

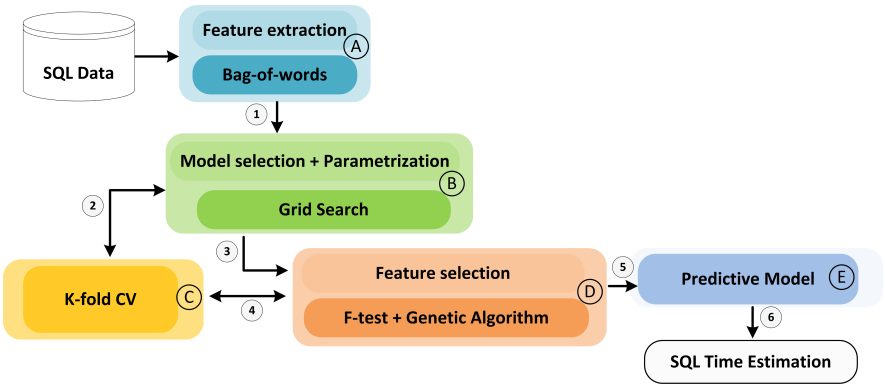


Fig. 1. Our Approach

Several regression methods were tested in our experiments, but we will focus only in those that had the best performance, which were: SVR, RFR and GBR. These methods have a large number of parameters. In addition, the change of parameters causes uncontrolled, non-intuitive effects to the model. To circumvent this problem, we use the technique of grid search to find a parameter configuration for each method, thus optimizing its accuracy. This technique consists of setting manually a set of parameters configuration, and exhaustively training a model for each of them and evaluating this model with a performance metric.

In order to tune the parameters, the Model Selection phase (B) receives such data set (1) and performs a grid search over the most relevant, method-specific parameters. Each trained model is then evaluated using K-fold CV (2) in order to estimate its associated error. For K-fold CV method (C), the designated error metric is the mean absolute error from the expected value to the estimated target values of each partition. Experiments performed with BoW vectors generated from SQL queries resulted in very high number of components. Training a regression model with such dimensionality can be computationally unfeasible and, moreover, there may exist features that do not explain the phenomenon which may mislead the regression method producing inaccurate predictive models. Therefore a process of dimensionality reduction is applied, by selecting a subset of features that best describe the phenomenon.

The Feature Selection (D) uses the models with tuned parameters received from the Model Selection (3). And it is done in two stages: First, a statistical filter is applied. Let X be the set of all features in the dataset, the p-value score is computed for every features in relation to the target feature, thus iteratively we generate sets of features $V_1, V_2, ...$ in 3 steps:(I) We start the iteration with set of features V_1 containing only the feature with the highest p-value and $i = 0$; (II) $i = i + 1$, generate a new set V_i such that $V_i = V_{i-1} \cup u$ where u and is the feature with highest p-value in $V_{i-1} - X$; (III) Measure the error for the newly trained model with this new configuration using K-fold CV; (IV) if $i \leq 50$ and if the error associated for model with features V_i has decreased compared with the model with features V_i, stop. Otherwise, go to (II). At the end, we select the model that obtained the lowest error; Second stage, using genetic algorithm, let $X = x_1, x_2, \ldots, x_k$ be the features selected by statistical filter. Each individual in a population represents a subset of X, thus an individual I in the population is given by a binary string S of size k. Thereby, x_i belongs to the subset of X represented by I if and only if $S_i = 1$ and otherwise if $S_i = 0$, $\forall i \quad 1 \leq i \leq k$. The fitness function designated to measure the quality

of each individual is the K-fold CV (4). Therefore, this step outputs the best model generated (5). This way, it is produced an accurate and not computationally expensive composed predictive model (E), which can predict the response time for a unseen SQL query that has not yet been executed in the system.

4. EXPERIMENTS

4.1 Environment

We deploy a prototype of our approach and conduct experiments on a virtual machine on OpenNebula infrastructure at Federal University of Ceara [OpenNebula 2013]. This machine has a 1.0 GHz Xeon processor, 1.7 GB memory and 160 GB disk capacity. We use the Ubuntu 12.04 operating system and MySQL 5.5. According to [OLTPBenchmark 2013], the following database was generated based on TPC-C with 400 MB. Regarding the regression methods, we used libsvm [Chang and Lin 2011] for Support Vector Regression, Scikit-learn [Pedregosa and et al. 2011] for Random Forest Regression and Gradient Boosting Regression and Pyevolve [Perone 2009] for the genetic algorithm implementation.

The SQL data was generated using this environment and a large amount of SQL queries was generated. Therefore, for purposes of experiments, we generated four data dets. Three of them are composed by the last 2000 executed queries in the benchmark of a single type - Select, Insert or Update - and the fourth is composed only by Delete queries, and it has only 400 queries due to the low occurrence of Delete queries in TPC-C.

4.2 Experimental Results

The full experiment was executed for each data set (Select, Insert, Update and Delete), using each regression method (SVR, RFR and GBR) as a model provider for k-fold CV. The tests were made using 4-fold CV. Firstly, the feature extraction step is executed on each dataset. The four sets of vectors are directly given as input for Model Selection step. The grid search is performed over each pair of Regression Method and dataset, the Mean absolute error (MAE) obtained and the number of features (#) is presented in Table I.

	Select		**Insert**		**Update**		**Delete**	
Regression Methods	MAE	#	MAE	#	MAE	#	MAE	#
Support Vector Regression	73.62	968	14.96	5182	162.61	2021	6.56	34
Gradient Boosting Regression	74.21	968	14.90	5182	154.36	2021	6.54	34
Random Forest Regression	85.18	968	19.79	5182	167.31	2021	9.78	34

Table I. Model Selection (MAE in milliseconds)

Insert and Update queries add information on the Data Base, so they tend to have a wider variety of word, and thus, they have more words than the other types. Delete queries have so few features because of the number of delete queries in TPC-C and, in addition, the actions of the queries are always over the same table. Delete and Insert queries have a lower MAE because their response times are low and do not vary much. Differently from Select and Update queries, which can reach a response time of 6000 ms. In general, all the queries types have a large dimensionality. For this reason, we search for a smaller set of explanatory features. The Statistical Filter is then applied as we can in Table II.

Note that we can describe most of the events with few features, and with an acceptable trade-off between number of features and obtained MAE. Furthermore, in some cases the MAE has decreased. Now, we run the genetic algorithm with 30 individuals and 30 generations. At each generation, the best individual is preserved, and we maintain the same sets of features obtained in the previous

	Select		Insert		Update		Delete	
Regression Methods	MAE	#	MAE	#	MAE	#	MAE	#
Support Vector Regression	75.95	34	14.96	305	162.12	3	6.56	5
Gradient Boosting Regression	70.53	7	14.90	48	161.55	39	6.54	5
Random Forest Regression	84.34	51	24.93	20	171.63	22	11.56	4

Table II. Statistical Filter (MAE in milliseconds)

generation. We illustrate, with two cases, the convergence of the algorithm in Figure 2, showing the evolution of the MAE along the generations, we show the minimum, average and maximum in each generation.

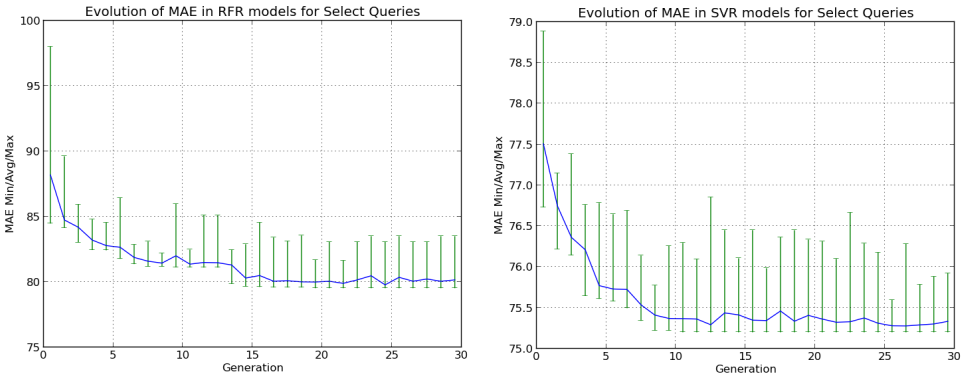


Fig. 2. Evolution of MAE in RFR and SVR models for Select Queries

	Select		Insert		Update		Delete	
Regression Methods	MAE	#	MAE	#	MAE	#	MAE	#
Support Vector Regression	75.23	24	14.96	133	162.12	3	6.56	5
Gradient Boosting Regression	70.53	7	14.90	22	160.59	13	6.54	5
Random Forest Regression	79.51	30	24.01	8	172.58	11	11.56	4

Table III. Genetic Algorithm (MAE in milliseconds)

In Table III we have the complete result of this step, which was not executed for all the cases, it was executed only for cases where there are more than 20 features. This model still performs poor predictions for Update queries, due to the their response time variations, and BoW cannot represent the complexity of the environment. But for the other types, we have a model with few features to quickly predict the response time for a query with an acceptable MAE. Besides this, given that it is a cloud environment, the aim is guarantee the QoS, so there is a comfortable gap for errors in prediction. For example, for a new query Select with SLA set at 800 ms, the response time provided by our approach will be between 730 ms and 870 ms. This information assists the provider in resource allocation and the elasticity, ensuring the QoS. A direct comparison with other works is not possible due to the different bechmarks employed in tests, as TPC-DS in [Ganapathi et al. 2009] and TPC-H in [Wu et al. 2013] and [Akdere et al. 2012]. Moreover, they used another error metric, the relative error.

5. RELATED WORK

A different approach using learning algorithms was proposed in [Akdere et al. 2012]. Their representations for the queries are based on the information provided by the query optimizer of the DBMS and by the cost estimation of each operator in the query execution plan. They developed modeling in different levels of granularity. They offer also a on-line approach to train model in the execution time with features captured from the newly received queries. The prediction of other metrics such as CPU, memory and disk usage was studied in [Ganapathi et al. 2009]. This approach also uses query optimizer information to extract feature for the queries. [Singhal 2012] presents a framework to estimate the query response time for database systems. This framework uses database concurrency model, CPU and, I/O to predict response time. [Wu et al. 2013] proposes a model to predict query execution time based on query plan using machine learning. These four approaches introduce an I/O overhead in the DBMS to produce features. In contrast, our approach uses only the text representation of queries to accomplish the prediction.

6. CONCLUSION

This work presented a machine learning approach for SQL queries response time estimation in the cloud. It combines different techniques to address each step of the predictive estimation time problem. We evaluated this work considering the cloud environment. According to the analysis of the obtained results, our approach uses few features to quickly predict the response time for a query with an acceptable error. As future work, we intend to conduct experiments with different benchmarks and make a deeper investigation of the reason for the poor perfomance of the model for Update queries. Other important issues to be addressed is to use a small set of information from the DBMS query optimizer or query execution plan to improve our approach. Finally, we intend to propose a new regression method specific to queries response time estimation.

REFERENCES

Akdere, M., Cetintemel, U., Riondato, M., Upfal, E., and Zdonik, S. Learning-based query performance modeling and prediction. In *ICDE '12*. pp. 390–401, 2012.

Breiman, L. *Arcing Classifiers*, 1998.

Breiman, L. Random forests. *Mach. Learn.* 45 (1): 5–32, 2001.

Chang, C.-C. and Lin, C.-J. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* vol. 2, pp. 27:1–27:27, 2011.

Elmore, A. J., Das, S., Agrawal, D., and El Abbadi, A. Zephyr: live migration in shared nothing databases for elastic cloud platforms. In *SIGMOD '11*. pp. 301–312, 2011.

Friedman, J. H. Greedy Function Approximation: A Gradient Boosting Machine. In Annals of Statistics. *Annals of Statistics* vol. 29, pp. 1189–1232, 2000.

Friedman, J. H. Stochastic gradient boosting. *Comput. Stat. Data Anal.*, 2002.

Ganapathi, A., Kuno, H., Dayal, U., Wiener, J. L., Fox, A., Jordan, M., and Patterson, D. Predicting multiple metrics for queries: Better decisions enabled by machine learning. In *ICDE '09*. pp. 592–603, 2009.

OLTPBenchmark. *OLTPBenchmark*, 2013. `http://www.oltpbenchmark.com`.

OpenNebula. *OpenNebula - Open Source Data Center Virtualization*, 2013. `http://www.opennebula.org`.

Pedregosa, F. and et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* vol. 12, pp. 2825–2830, 2011.

Perone, C. S. Pyevolve: a python open-source framework for genetic algorithms. *SIGEVOlution* 4 (1): 12–20, 2009.

Singhal, R. A framework for predicting query response time. In *14th IEEE International Conference on High Performance Computing and Communication*. pp. 1137–1141, 2012.

Sousa, F. R. C., Moreira, L. O., Santos, G. A. C., and Machado, J. C. Quality of service for database in the cloud. In *CLOSER '12*. pp. 595–601, 2012.

TPC. *TPC-C*, 2013. `http://www.tpc.org/tpcc/`.

Vapnik, V. *Statistical learning theory*. Wiley, 1998.

Wu, W., Chi, Y., Zhu, S., Tatemura, J., Hacigumus, H., and Naughton, J. F. Predicting query execution time: Are optimizer cost models really unusable? *ICDE '13*, 2013.

Tracking Sentiment Evolution on User-Generated Content: A Case Study on the Brazilian Political Scene

Diego Tumitan, Karin Becker

Instituto de Informatica - Universidade Federal do Rio Grande do Sul, Brazil
{dctumitan, karin.becker}@inf.ufrgs.br

Abstract. Opinion mining, which aims the automatic processing of subjective information, has become a key field. In the political context, the awareness of people's sentiment towards their representatives, public organizations, political parties or politicians, can support political decisions, campaign moves, government policies, marketing strategies, etc. This paper describes a case study over opinions expressed about politicians as a reaction to news. Our ultimate goal was to detect whether user comments on an on-line newspapers reflect external indicators of public acceptance (e.g. vote intention). The paper outlines the approach used to identify and classify sentiment in news comments written in Portuguese language and to correlate it to external indicators, and discusses the main results for this case study.

Categories and Subject Descriptors: I.7 [**Document and Text Processing**]: Miscellaneous

Keywords: news, opinion mining, sentiment analysis, user-generated content

1. INTRODUCTION

Web users are no longer mere consumers of information. They interact with other users, expressing opinions and sentiment about various entities, such as products, brands, political figures, etc. This rich content can influence others and therefore, opinion mining, which aims at automatically processing subjective content, has become a very active field [Tsytsarau and Palpanas 2012].

Early work on opinion mining concentrated on products/services reviews [Tsytsarau and Palpanas 2012]. More recent works focus on specific entities (e.g. politicians, brands) on social networks [Pak and Paroubek 2010; Guerra et al. 2011] and news [Godbole et al. 2007]. The overall goal is to capture and track the general sentiment over the time, as represented by some metric, towards a target entity supporting many types of application.

In the political context, the awareness of people's opinion about their representatives, public organizations, political parties or politicians, can support political decisions, campaign moves, government policies, marketing strategies, etc. Traditional approaches involve expensive (and therefore infrequent) polls for detecting politicians' popularity, government approval, vote intention, etc. The potential of opinion mining for more up-to-date and broad opinion perspective has been demonstrated with regard to social medias such as Twitter or Facebook, and many commercial tools are available (e.g. TweetSentiments[1], Sentimonitor[2]). Less attention has been paid to user-generated content over news.

This paper describes a case study over opinions expressed in Portuguese about politicians as a reaction to news. Our ultimate goal was to detect whether user comments on on-line news reflects

[1]http://twittersentiment.appspot.com
[2]http://www.sentimonitor.com

external indicators of public acceptance (e.g. vote intention). We analyzed data referring to the 2012 mayoral elections of São Paulo, expressed as comments on a major on-line newspaper (Folha on-line), and used the external indicators provided by Datafolha polls. Our findings for this specific case study were: a) people do not tend to comment about the specific news content, but rather express their feelings in general about politics, politicians and their parties; b) there was an overall frustration over the current state of affairs, with a majority of negative comments; c) unlike other medias (e.g. Twitter, Facebook), very few people use this media support candidates, and d) considering the metrics developed, the sentiment has moderate correlation with vote intention for the candidates ellected for the second round. This case study is part of an on-going research about mechanisms for detecting and predicting sentiment evolution, based on user-generated comments written in Portuguese language.

The remainder of this paper is organized as follows. Section 2 contains related work. We outline the adopted approach in Section 3, and describe the case study in Section 4. Conclusions and future work are addressed in Section 5.

2. RELATED WORK

Opinion mining involves detecting subjective content, classifying its polarity, and summarizing the overall sentiment. Polarity classification relies on dictionary-based, machine-learning or statistical methods [Tsytsarau and Palpanas 2012]. The former is the most common one, but its results are dependent on the quality of sentiment lexicons. Classification can be at document or sentence-level. The latter is appropriate when a same document express opinions on several entities.

Many works have addressed the identification of sentiment about entities. Sentiment expressed in news towards an specific entity is analyzed and tracked in the system discussed in [Godbole et al. 2007]. The approach is based on an expansion technique over WordNet [Fellbaum 2010], which is not available for Portuguese. User-generated content on tweets are addressed in [Pak and Paroubek 2010; Narr et al. ; Guerra et al. 2011]. The former two propose a language-independent machine-learning approach, but which requires a training corpus. To eliminate that need, a transfer-knowledge approach is proposed in [Guerra et al. 2011], in which the sentiment is derived from the social relations between known pro/against opinion holders. On the political context, a study analyzes the results of elections in Germany with regard to the emotion expressed in tweets with mentions to political parties and candidates [Tumasjan et al. 2010]. The approach uses a linguistic tool that is not available for the Portuguese language.

One of the few works addressing user-generated content related to news in Portuguese language is [Sarmento et al. 2009], in which the authors create a set of lexico-syntactic patterns to identify the polarity of sentences. All used sentences are from comments related to a political newspaper, but the authors' goal is to create a reference corpus for political opinion mining.

Our work differs from the previous ones in that we address comments expressed as a reaction to news, in Portuguese language, and verify whether they correlate to external indicators of public acceptance. Towards this end, we develop a case study using the mayoral election of 2012.

3. APPROACH OUTLINE

In this paper, we describe a case study that tracks the general public sentiment towards political figures over the time, based on the perception of comments extracted from news regarding the Brazilian political scene. Our ultimate goal was to detect whether user comments on on-line newspapers reflect and correlate with external indicators of public acceptance (e.g. vote intention). This is a preliminary result of a broader on-going research, in which techniques for forecasting changes of attitude are under investigation. Figure 1 shows an overview of the proposed analysis approach. Notice that the process is highly iterative, in which returns to previous steps are necessary to improve results.

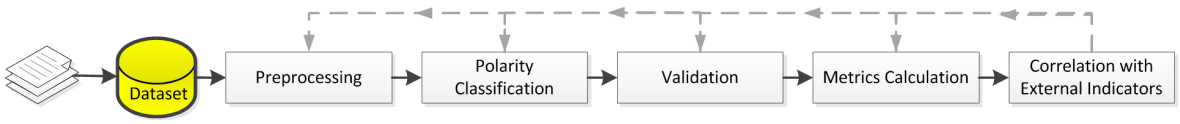


Fig. 1. Overview of the iterative proposed analysis approach.

Preprocessing: considering a dataset composed of news and their respective user comments, this step involves tasks to improve or discard user-generated comment, such as removing duplicated or excessively short comments, identification of slangs or domain-specific vocabulary, unification of all names and aliases for observed candidates (e.g nicknames, mean expressions), identification of misspelled or disguised words (e.g. cursing), among others.

Polarity Classification: encompasses breaking comments into sentences, identifying the ones with mentions to candidates, and classifying sentences' polarity. Sentence-level classification is necessary for most comments that involve more than one entity. We used a Portugal Portuguese sentiment lexicon, enriched with Brazilian and domain-specific terms. Sentiment words are identified, summarized (positive terms are added, and negative terms are subtracted), and the resulting sentiment is assigned to the target. Each target is thus associated with a set of positive, negative and neutral sentences.

Validation: In the lack of an annotated corpus, and considering the extent of the content to be analyzed, we adopted a sample-based validation. We randomly selected a set of sentences, and 3 different people annotated them using the same set of instructions. Only sentences with at least 2 agreements were used to validate.

Metrics Calculation: the polarized sentences are finally summarized to compose different metrics. We developed and experimented with different metrics, displayed in the first two columns of Table I. A suitable metric should reflect the general public sentiment, and a possible way to evaluate the best metric is through their correlation of with external indicators of public acceptance.

Correlation with External Indicators: Each domain may have different indicators that express overall sentiment, and which can be used to assess the results. In this case study, we adopted a typical election indicators such as voting intention and rejection rate. Job approval, popularity, economical indexes are other possible examples.

4. CASE STUDY

4.1 Dataset and Data Preprocessing

Dataset. We composed the dataset with news and comments extracted from the on-line version of Folha de São Paulo, one of the main newspapers in Brazil. We extracted news on the Brazilian Mayoral Elections of São Paulo, related mostly with the candidates, their parties, campaign moves, etc. We used Google Reader[3] as the indexer of Folha.com's political news section, called *Poder* (Power). These news cover the period from September 1st, to October 7th, 2012, which corresponds to the first round of the election. We extracted 583 news and 36,108 respective user's comments. We only considered the three leading candidates: Celso Russomanno, Fernando Haddad and José Serra [Datafolha 2012].

Data Preprocessing. we removed 3,808 duplicated comments (detected using the Cosine Similarity index) and 7,185 unreasonably short comments (less than 3 words or empty). After manual inspection, we realized other issues to be handled, such as transformation of words that were disguised by special characters (e.g. c@n@lh@ - scoundrel); misspelling or bad use of accentuation; use of regional (e.g. "petralha", "malufista") and idiomatic expressions ("é o cara" for the expression "he's the man") denoting sentiment. We also identified variations on candidate mentions (e.g. Serra, Zehserra),

[3]http://google.com/reader

Table I. Proposed metrics and correlation between sentiment metrics and vote intention/rejection rate.

Description	Formula	Haddad	Serra	Russomanno
Ratio of positive sentiment of an entity to the negative sentiment of the same entity	$s_d = \frac{pos_e}{neg_e}$ (1)	0.57/0.44	0.54/-0,14	0.12/-0.04
Ratio of positive sentiment to the total sentiment	$s_d = \frac{pos_e}{pos_e + neg_e}$ (2)	0.56/0.40	0.54/-0.16	0.08/-0.04
Ratio of negative sentiment to the total sentiment	$s_d = \frac{neg_e}{pos_e + neg_e}$ (3)	-0.56/-0.40	-0.54/0.16	-0.08/0.04
Ratio of positive sentiment of an entity to the positive sentiment of all entities	$s_d = \frac{pos_e}{pos_{entities}}$ (4)	0.09/0.07	0.22/0.29	0.39/-0.29
Ratio of negative sentiment of an entity to the negative sentiment of all entities	$s_d = \frac{neg_e}{neg_{entities}}$ (5)	-0.35/-0.23	0.16/0.34	0.16/-0.04

some of them with implied sentiment (e.g. Vampiserra, Malhaddad, as mean aliases), which were handled using regular expressions based on the candidates' names. Domain-specific sentiment vocabulary was added to the used lexicon along the process. Lastly, comments were broken into 79,752 sentences.

4.2 Target Identification and Sentiment Classification

Each sentence containing a mention to a candidate and sentiment words (9,758 sentences) was then polarized. We adopted SentiLex-PT [Silva et al. 2012], which contains 7,014 lemmas e 32,347 inflected forms for Portugal Portuguese. Each entry has a polarity (1, -1 and 0). To improve our results, over the time we added regional and domain-dependent terms to this dictionary. We tried different approaches for handling negations (e.g. "not good"), but no experiment yielded good results yet.

4.2.1 *Approach Validation.* To validate the sentence classification performance, we randomly selected 600 sentences that contained mentions to the candidates. The annotators were three graduate students majored in computer science, with no previous experience on corpus annotation. They were instructed to base their classification on what was explicitly written, disregarding any assumption about political entities or parties [Sarmento et al. 2009], so that their political background would not interfere in their judgment. The resulting gold-standard contained 482 sentences classified as negative, 72 as positive, and 46 as neutral. We discarded 3% of the sentences due to the lack of agreement.

4.2.2 *Performance Assessment.* Considering our gold-standard, we developed different experiments to classify the sentiment of the sentences, including attempts to handle negation adverbs (e.g. not good, never excelled). Only the best 3 results are discussed here due to space limitations. Results are summarized in Table II, which does not display the results for the neutral class.

In the *"Baseline"* experiment, we straightforwardly applied the co-occurrence method, with no significant terms preprocessing. We obtained fairly good results in terms of precision for the negative sentences, but we were not satisfied neither with the precision of the positive sentences, nor with the recall for both positive and negative classes. The next two experiments report two attempts developed for improving positive sentences classification and recall.

In the *"Modified Lexicon"* experiment, we manually analyzed the top 1,000 more frequent words that were not in the SentiLex-PT. As a result, we selected and added to the lexicon 268 new words and idiomatic expressions. Our results for the positive class significantly improved.

The *"Without Accentuation"* experiment adopts the refined lexicon and addresses users' accentuation typos. We removed all the accentuation from both comments and sentiment lexicon. We significantly improved the recall for negative sentences, while maintaining the same precision for both classes. Thus,

Table II. Experiments results according to Accuracy (P), Micro-Averaged F1 (Mi-A), Macro-Average F1 (Ma-A), Precision (P), Recall (R) and F-score (F).

Variation	**A(%)**	**Mi-A(%)**	**Ma-A(%)**	**Polarity**	**P (%)**	**R (%)**	**F(%)**
Baseline	35.54	46.17	39.19	Positive	17.35	21.79	19.32
				Negative	89.22	37.76	53.06
Modified Lexicon	43.21	50.05	46.17	Positive	26.02	**65.38**	**37.23**
				Negative	**90.52**	39.63	55.12
W/O Accentuation	**52.14**	**58.52**	**51.02**	Positive	**26.99**	56.41	36.51
				Negative	90.18	**51.45**	**65.52**

according to Accuracy, Micro-averaged F1 and Macro-Averaged F1, the results of this experiment constitute a significant improvement with regard to all previous attempts. This is thus the approach used for tracking sentiment in the remaining of this paper.

4.2.3 *Polarity Classification Error Analysis.* In general, our method did not perform well in classifying positive sentences. User-generated content is full of typographic errors, and thus the lexicon may contain the sentiment word, but the term is not recognized. We minimize this issue at some extent by disregarding accentuation. We also observed the importance of sentiment words related to an specific context [Godbole et al. 2007], with many terms specific to the city of São Paulo.

An important issue is that many sentences express comparative opinions (e.g. *"X is better than the candidate Y, because he is less involved in corruption"*, where the negative polarity of *"corruption"* will be related to both candidates). Finally, irony and sarcasm is a hard problem.

4.3 Sentiment Tracking

Considering the 9,758 polarized sentences resulting from the previous steps, we calculated the value for each metric of Table I per candidate and per day. Finally, we examined whether they correlated over the time with election external indicators (vote intention and rejection rate), as provided by poll results provided by Datafolha [Datafolha 2012] using Pearson correlation.

In order to visually analyze the trend and compare their evolution more easily, we applied z-score for both data since they are at different scales. We also distributed the polls values linearly due to the different data granularity (eight polls exist in the considered period - x axis of Figure 2). This choice represents the assumption that no change occurred in public opinion in between the poll results' publications, which may not be correct. Figure 2 shows the overlap of the sentiment ratio (dashed line) and vote intention (solid line) for the two candidates elected for the second round. The peak of sentiments observed on Sept. 25^{th} and Oct. 3^{rd} correspond to comments about news on the tie between Serra and Haddad. The correlation of each metric for both vote intention and rejection rate for each candidate is displayed in the last three columns of Table I. Considering the vote intention, the first three metrics worked fairly well for both Serra and Haddad, with moderate correlation. This means that vote intention increases along with positive comments and decrease of negative comments. However, no metric presented a consistent result for Russomanno. Actually, we observed that people just quited commenting about him near the election day, a fact that may have influenced these results. As for rejection rate, we observed almost no correlation, meaning that either these are not good metrics, or rejection rates reveal intrinsic feelings that are harder to influence with comments on news.

5. CONCLUSION AND FUTURE WORK

This paper described a case study over opinions expressed about politicians as a reaction to news. Our ultimate goal was to detect whether user comments on an on-line newspapers reflect external indicators of public acceptance. We presented the method used, the best results of the experiments

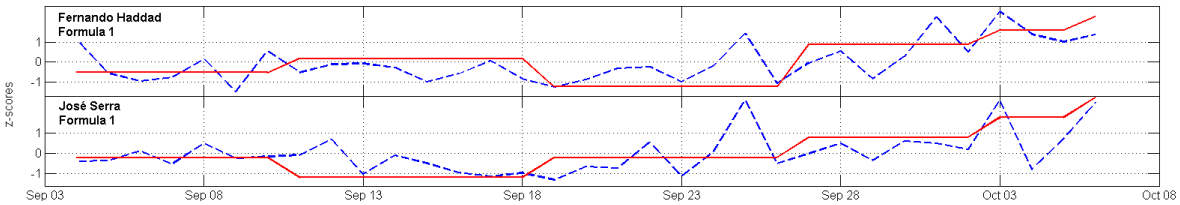


Fig. 2. Overlap of positive sentiment ratio (dashed) and vote intention (solid), normalized using z-score

developed, and the results for the correlation according to several metrics, which has shown a moderate correlation with vote intention for the candidates elected for the second round.

Our primarily expectation was that user-generated comments would reveal the authors' opinion about the respective news post. We realized that most comments refers to frustration about politics in general; transference of opinion by the candidate's association to other corrupt politicians or parties; political scandals; poor previous administration; etc.

We also expected that authors would support or make opposition to candidates. Support was very rare, and the authors would rather debate which candidates were less worst. Thus, this media seems to present a different role and impact if compared to Twitter or other social networks [Tumasjan et al. 2010; Pak and Paroubek 2010; Guerra et al. 2011].

Another challenge was the presence of sentiment words and idiomatic expression that are exclusive to the Brazilian language, political context and even a city. Reactions to Russomanno were very different from the ones towards the other candidates, due to his association to religion, a situation which was unique to the city of São Paulo. Replicating this experiment to other cities or elections involves the hard task of identifying contextual, regional and domain-dependent terms. The process of acquiring domain-specific vocabulary is laborious, and subject to errors. New approaches need to be considered for improving the classification results.

This work is part of an on-going broader research on deriving models that detects changing patterns on the attitude towards a subject. To overcome limitations of the present study, we need to consider other cities and candidates, as well as other on-line news and social medias, thus extending the type of comments addressed and their scope. We are currently experimenting techniques to develop a predictive model for public acceptance of political figures based on the sentiment expressed.

REFERENCES

Datafolha. Intenção de voto para prefeito de são paulo, 2012. Retrieved November 19, 2012.

Fellbaum, C. *WordNet*. Springer, 2010.

Godbole, N., Srinivasaiah, M., and Skiena, S. Large-scale sentiment analysis for news and blogs. In *Proceedings of the International Conference on Weblogs and Social Media (ICWSM)*. Vol. 2, 2007.

Guerra, P. H. C., Veloso, A., Jr., W. M., and Almeida, V. From bias to opinion: a transfer-learning approach to real-time sentiment analysis. In *Proceedings of the KDD*. pp. 150–158, 2011.

Narr, S., Hülfenhaus, M., and Albayrak, S. Language-independent twitter sentiment analysis.

Pak, A. and Paroubek, P. Twitter as a corpus for sentiment analysis and opinion mining. In *LREC*, 2010.

Sarmento, L., Carvalho, P., Silva, M., and de Oliveira, E. Automatic creation of a reference corpus for political opinion mining in user-generated content. In *Proceedings of the 1st international CIKM workshop on Topic-sentiment analysis for mass opinion*. ACM, pp. 29–36, 2009.

Silva, M., Carvalho, P., and Sarmento, L. Building a sentiment lexicon for social judgement mining. *Computational Processing of the Portuguese Language*, 2012.

Tsytsarau, M. and Palpanas, T. Survey on mining subjective data on the web. *Data Mining and Knowledge Discovery*, 2012.

Tumasjan, A., Sprenger, T., Sandner, P., and Welpe, I. Predicting elections with twitter: What 140 characters reveal about political sentiment. In *Proceedings of the Fourth International aaai conference on weblogs and social media*. pp. 178–185, 2010.

Evolução dos Temas de Interesse do SBBD ao Longo dos Anos

Anderson Kauer, Viviane Moreira

Instituto de Informática – UFRGS, Brazil
{aukauer, viviane}@inf.ufrgs.br

Abstract. A mineração temporal de textos (temporal text mining - TTM) é uma técnica cujo objetivo é descobrir a estrutura latente e padrões temporais em coleções de texto. Estas características são importantes em coleções em que os tópicos de interesse mudam frequentemente com o passar do tempo. Além disso, a mineração temporal de textos é útil em ferramentas de sumarização e descoberta de tendências. Neste trabalho, aplicamos TTM para analisar a evolução dos temas abordados pelos artigos publicados no Simpósio Brasileiro de Bancos de Dados (SBBD) ao longo dos anos. Para tal, coletamos os abstracts dos artigos publicados entre 1989 e 2012. Os resultados mostram tendências interessantes na distribuição dos temas ao longo do tempo.

Categories and Subject Descriptors: H.3.3 [**Information Storage and Retrieval**]: Clustering

Keywords: clustering, evolução temporal, mineração de textos

1. INTRODUÇÃO

Em diferentes domínios de aplicações, existe a necessidade de se conhecer os principais temas discutidos em coleções de texto. Um cenário bastante comum consiste em bases de artigos científicos, onde em determinado ponto no tempo os temas de interesse são modificados. Isto faz com que alguns assuntos tornem-se mais expressivos, sejam descontinuados, ou até mesmo se especializem em novos temas. A mineração temporal de textos (*temporal text mining* – TTM) é uma técnica que consiste em identificar estas estruturas latentes como responsáveis por definir o assunto ou modelo gerador dos documentos [Mei and Zhai 2005], assim é possível identificar como o estudo de um tópico influenciou o estudo de um outro tópico em épocas diferentes.

Neste trabalho, aplicamos o método para TTM que foi introduzido por [Mei and Zhai 2005] como forma de identificar automaticamente a evolução dos temas abordados no Simpósio Brasileiro de Bancos de Dados (SBBD) ao longo dos anos. O método utilizado consiste na criação de um modelo probabilístico que visa descobrir os temas latentes nas coleções em cada edição e identificar as relações entre estes temas ao longo dos anos.

O restante deste trabalho está organizado da seguinte maneira: na Seção 2 apresentamos uma breve descrição dos trabalhos que compõem o estado da arte. Na Seção 3 descrevemos os detalhes do método implementado e na Seção 4 são avaliados os resultados obtidos nos experimentos. Finalmente, na seção 5 são discutidos os resultados e apontamos algumas direções para trabalhos futuros.

2. TRABALHOS RELACIONADOS

Muitos aspectos da TTM têm sido utilizados em outras áreas de pesquisa. Inicialmente, a TTM estava fortemente relacionada à sumarização temporal [Allan et al. 2001]. Este artigo, embora relacionado com sumarização, tem como diferencial a identificação das características implícitas no texto.

A detecção e o rastreamento de temas (*topic detection and tracking*–TDT) [Kaur and Gupta 2012] concentra-se em descobrir e extrair informações comuns a partir de um conjunto de documentos. A relação com este trabalho está na identificação da estrutura latente da coleção, entretanto a TDT não considera os padrões temporais e evolucionários da coleção.

Sipos et. al. [Sipos et al. 2012] desenvolveram um trabalho bastante semelhante ao nosso propósito, entretanto, o objetivo é identificar a influência de documentos e autores com o passar dos anos, enquanto que não são consideradas as transições entre temas implícitos na coleção.

Por fim, algoritmos de agrupamento para coleções de texto têm ganhado bastante atenção nos últimos anos devido à capacidade de identificar estruturas latentes dos documentos [Zhai et al. 2004]. Esta abordagem é interessante por identificar características comuns e raras em diferentes coleções. Adicionalmente é possível identificar características comuns entre os temas conforme descrito em [Mei and Zhai 2005] como forma de identificar as transições e intensidade em cada período de tempo.

3. TTM SOBRE O SBBD

Para o desenvolvimento deste trabalho, seguiu-se a metodologia abordada em [Mei and Zhai 2005]. Onde, por definição, existe conjunto de coleções $\{C_1, C_2, \ldots, C_n\}$, onde cada coleção $C_i = \{d_1, d_2, \ldots, d_n\}$ e cada documento d_j é representado por uma sequência de palavras que compõem o vocabulário $V = \{w_1, w_2, \ldots, w_{|V|}\}$, onde cada w_i é uma palavra.

Neste contexto, um *tema* θ em uma coleção C_i segue uma distribuição probabilística de palavras que caracterizam um tópico. Esta distribuição permite que palavras que compartilham um determinado tema estejam próximas.

Assim, cada coleção C_i é composta por um conjunto de temas $\Theta_i = \{\theta_1, \theta_2, \ldots, \theta_n\}$, que também podem ser considerados os modelos responsáveis por gerar os documentos. Cada documento d_j tem probabilidade $\pi_{j,k}$ de ter sido gerado pelo tema θ_k e portanto $\sum_{k=0}^{n} \pi_{j,k} = 1$.

3.1 Extração dos temas

A extração dos temas em cada coleção C_i seguiu um modelo probabilístico de mistura que foi descrito em [Zhai et al. 2004] e, posteriormente, adaptado para TTM [Mei and Zhai 2005]. Este método calcula a distribuição das palavras em todas as coleções (Eq. 1), gerando o modelo conhecido como *background text* (θ_B), para então estimar um conjunto de modelos responsáveis pela distribuição das palavras em cada documento.

$$\hat{p}(w|\theta_B) = \frac{\sum_{i=1}^{m} \sum_{d \in C_i} c(w,d)}{\sum_{i=1}^{m} \sum_{d \in C_i} \sum_{w' \in V} c(w',d)} \tag{1}$$

onde $c(w,d)$ indica o número de ocorrências da palavra w no documento d, m é o número de coleções e w' é uma palavra pertencente ao vocabulário V.

O ajuste dos parâmetros pode ser feito utilizando uma variante do algoritmo *Expectation Maximization* (EM) [Dempster et al. 1977]. A utilização de λ_B como uma constante tem como objetivo tornar as palavras mais discriminativas em relação à coleção, reduzindo assim a influência de palavras pouco informativas.

Seguindo a definição do algoritmo EM, cada iteração ocorre em dois passos: no passo E (*expectation*) são estimadas as variáveis ocultas $\{z_{d,w}\}$ e suas probabilidades $p(z_{d,w} = j)$ indicando que a palavra w no documento d foi gerada a partir do tema j:

$$p(z_{d,w} = j) = \frac{\pi_{d,j}^{(n)} p^{(n)}(w|\theta_j)}{\sum_{j'=1}^{k} \pi_{d,j'}^{(n)} p^{(n)}(w|\theta_{j'})} \tag{2}$$

$$p(z_{d,w} = B) = \frac{\lambda_B p(w|\theta_B)}{\lambda_B p(w|\theta_B) + (1 - \lambda_B) \sum_{j=1}^{k} \pi_{d,j}^{(n)} p^{(n)}(w|\theta_j)} \tag{3}$$

onde $p(z_{d,w} = B)$ indica a probabilidade da palavra w seguir a distribuição das palavras na coleção e n é o número da iteração.

Em seguida, no passo M (*maximization*) os parâmetros são atualizados:

$$\pi_{d,j}^{(n+1)} = \frac{\sum_{w \in V} c(w,d) p(z_{d,w} = j)}{\sum_{j'=1}^{k} \sum_{w \in V} c(w,d) p(z_{d,w} = j')} \tag{4}$$

$$p^{(n+1)}(w|\theta_j) = \frac{\sum_{i=1}^{m} \sum_{d \in C_i} c(w,d)(1 - p(z_{d,w} = B)) p(z_{d,w} = j)}{\sum_{w' \in V} \sum_{i=1}^{m} \sum_{d \in C_i} c(w',d)(1 - p(z_{d,w'} = B)) p(z_{d,w'} = j)} \tag{5}$$

onde $p(w|\theta_j)$ indica a probabilidade da palavra w ser gerada a partir do tema j.

Uma característica deste método é a garantia de convergência que maximiza a verossimilhança dos dados. Assim, o conjunto de parâmetros $\Lambda = \{\theta_j, \pi_{d,j} | d \in C_i, 1 \leq j \leq k\}$ pode ser avaliado ao final de cada iteração como:

$$\mathcal{L} = \sum_{d \in C_i} \sum_{w \in V} \left[c(w,d) \log(\lambda_B p(w|\theta_B) + (1 - \lambda_B) \sum_{j=1}^{k} (\pi_{d,j} p(w|\theta_j))) \right] \tag{6}$$

A utilização deste método exige que seja definido o número de temas em cada coleção C_i. Para isto, foram analisadas, de maneira empírica, as probabilidades $\pi_{d,j}$ em cada coleção de forma que cada um dos documentos pertencesse a um determinado tema com uma alta probabilidade (i.e., maior do que 0.7). Também é necessária a definição da constante λ_B, que na prática serve como o fator de discriminação entre as palavras. Neste contexto, λ_B controla o ruído existente na coleção. [Zhai et al. 2004] sugerem valores de λ_B entre 0.9 e 0.95 . Em nossos experimentos utilizou-se $\lambda_B = 0.95$ juntamente com remoção de palavras comuns (*stop-words*) e *Light Stemming* que consiste em remover os sufixos indicativos de plural, gerúndio e passado (*s, ing, ed*). Também não foram utilizadas *keywords*, pois este trabalho visa identificar as características implícitas nos documentos.

3.2 Análise das Transições Evolucionárias

A análise de transições consiste em identificar a proximidade entre os temas de coleções diferentes. No caso do SBBD, seria a proximidade entre temas de edições diferentes. Considerando que cada tema em uma coleção pode ser representado por $\gamma = \langle \theta_j, C_i \rangle$, uma transição evolucionária ocorre quando a distância entre γ_1 e γ_2 for menor que um determinado limiar. A medida utilizada segue a distância de Kullback-Leibler [Cover and Thomas 1991] ajustada ao modelo de mistura [Mei and Zhai 2005] apresentado anteriormente. Assim a proximidade entre dois temas $D(\theta_2 \| \theta_1)$ é calculada como segue.

$$D(\theta_2 \| \theta_1) = \sum_{i=1}^{|V|} p(w_i|\theta_2) \log \frac{p(w_i|\theta_2)}{p(w_i|\theta_1)} \tag{7}$$

onde V representa a união das palavras pertencentes a θ_1 e θ_2. Temas diferentes compartilham poucas palavras, assim utilizou-se a metodologia proposta em [Pinto et al. 2007] para evitar que temas com poucas palavras em comum obtivessem uma distância muito baixa.

Finalmente, são consideradas como transições evolucionárias todos os pares de temas que atingiram distância inferior ao limiar ξ. Assim, a definição deste limiar depende de análise de cada uma das combinações possíveis. Em nossos experimentos definiu-se empiricamente a utilização de $\xi = 60$.

4. EXPERIMENTOS E RESULTADOS

Inicialmente foram coletados os *abstracts* de artigos completos de todas as edições do SBBD (de 1986 a 2012). Assim, foram obtidas as coleções C_i necessárias. Um dos problemas encontrados nesta etapa foi a heterogeneidade entre os documentos: muitas edições antigas estão disponíveis somente na versão impressa; alguns artigos possuem resumo somente em português ou espanhol. Os artigos impressos foram digitalizados manualmente. Como focamos a análise nos *abstracts* em inglês, as edições de 1986 a 1988 foram descartadas. Portanto, a análise foi realizada sobre 475 *abstracts* publicados entre 1989 e 2012. Este conjunto de documentos foi particionado em 23 intervalos de tempo, onde cada intervalo representa uma edição do SBBD. Cada edição é composta por um número diferente de palavras e de documentos, este fato implica na utilização de um número variável de temas para cada uma destas coleções.

Para definir o número de temas foi avaliada a probabilidade $\pi_{d,j}$. Valores muito baixos para todos os documentos em um determinado tema implicam que um número maior de temas é necessário. Outro aspecto analisado foi o conjunto de temas final: visto que temas muito semelhantes indicam que existe um excesso de temas. A Tabela I apresenta as cinco primeiras palavras com maior probabilidade em cada um dos temas identificados. Devido ao grande número de coleções, são apresentados somente quatro temas em cada década.

Inicialmente, nota-se que cada um dos temas trata de algum aspecto específico da área de banco de dados. Em 1989 os principais temas eram modelagem em Redes de Petri, e bases de conhecimento. Em 1990 são identificados mais temas relacionados operações sobre visões (*views*), estruturas de janelas de transações, representação de regras em banco de dados, expressão de consultas e armazenamento de imagens. Em 1991 são apresentados trabalhos relacionados ao modelo ER Estendido, hierarquias tolerantes a falhas, banco de dados temporal e técnicas declarativas para otimização de consultas. Em 1992 são abordados modelagem e design de dados, expressão e otimização de consultas, e representação de objetos complexos.

Nove anos depois, em 1999 são apresentados trabalhos sobre banco de dados distribuídos, modelos multi-dimensionais, técnicas para criação de índices e persistência. Em 2000 os principais tópicos estão relacionados a dados temporais, transferência de dados, *data warehouse*, processamento e recuperação de e-mails, processamento paralelo de consultas e restrições de integridade. Em 2001 os trabalhos abordaram técnicas para criação de índices, dados temporais, *data mining*, integração de dados e versionamento de esquemas. Em 2002 os temas foram criptografia de dados, performance de consultas, dados semi-estruturados, banco de dados orientado a objetos, recuperação de informação multilíngue e ferramentas de auxílio à tomada de decisão.

Nos anos mais recentes, os temas mais relevantes em 2009 estão relacionados a consultas utilizando inferência probabilística, motores de busca e base de dados centralizadas, revisões no estado da arte, aprendizagem de máquina, mineração de dados e computação em nuvem. Em 2010, os temas mais relevantes estão relacionados à ordenação e recuperação de consultas, *clickthrough information*, identificação de *phrasal terms*, interpretação de palavras-chave, web semântica, classificação e recuperação de vídeo. Em 2011 o trabalhos abordam recuperação e processamento de imagens, algoritmos de classificação e ordenação, *data-intensive workflows* e ontologias. Finalmente, em 2012, as principais contribuições estão relacionadas a banco de dados orientado a grafos, algoritmos de *clustering*, *hardware*, eficiência de consumo de energia, identificação de referências em artigos e identificação de usuários maliciosos.

A partir dos temas extraídos, utilizou-se a distância de KL, conforme descrito na seção 3.2, para identificar as transições evolucionárias. A Figura 1 apresenta o grafo com as principais transições mostrando o fluxo entre os temas ao abordados ao longos dos anos. Praticamente não há transições significativas até 1996. Os temas tratados nas diferentes edições do SBBD destes anos são mais dissimilares entre si. A maior parte das transições aparece a partir de 1996. O Tema 5 (dados

Tabela I. Temas extraídos dos abstracts do SBBD

	Tema 1	Tema 2	Tema 3	Tema 4	Tema 5	Tema 6
1989	modell 0,155 nets 0,124 petri 0,124 entit 0,112 informalit 0,062	eiti 0,075 base 0,062 oms 0,050 prolog 0,049 datalog 0,049	knowledg 0,100 coupl 0,048 kee 0,034 dbs 0,034 krisy 0,033			
1990	view 0,240 updat 0,102 operation 0,054 constraint 0,048 directl 0,038	window 0,159 featur 0,057 pixel 0,046 pyrami 0,046 insid 0,044	rule 0,091 languag 0,087 rdl 0,054 program 0,075 variabl 0,050	imag 0,106 eds 0,038 cod 0,033 manager 0,032 express 0,027		
1991	eer 0,117 typ 0,096 lead 0,066 entit 0,065 extend 0,047	hierarch 0,096 answer 0,091 cooperativ 0,048 cobas 0,044 fault 0,044	temporal 0,113 stat 0,076 condition 0,056 disk 0,042 time 0,042	optimizer 0,085 dbms 0,048 commit 0,035 cope 0,033 purpos 0,021		
1992	application 0,155 aristot 0,055 composit 0,053 target 0,051 designer 0,045	catalog 0,106 standar 0,088 design 0,082 assessment 0,053 purpos 0,037	imag 0,086 class 0,070 complexit 0,066 languag 0,055 circumvent 0,036	step 0,058 appropriat 0,041 attach 0,035 workbench 0,033 cooperation 0,031	object 0,069 iql 0,052 dbm 0,034 interactiv 0,027 formulation 0,026	lock 0,044 expert 0,027 relat 0,026 tabl 0,026 transient 0,024
1999	migration 0,086 execution 0,068 network 0,060 ral 0,052 dimension 0,052	mdbs 0,141 transaction 0,098 local 0,088 multidatabas 0,087 failur 0,068	activ 0,065 data-driven 0,063 essentiall 0,062 rul 0,049 event-action 0,042	modul 0,074 snep 0,069 dimensional 0,045 dimension 0,042 dvr 0,042	index 0,059 exampl 0,034 dedy 0,033 hierarch 0,032 conventional 0,030	persistenc 0,044 action 0,035 reflection 0,027 uncertaint 0,026 fuzz 0,020
2000	temporal 0,155 kess 0,046 program 0,038 metadata 0,031 biodiversit 0,029	client 0,103 broadcast 0,081 mobil 0,064 protocol 0,051 listen 0,032	acces 0,077 spatial 0,057 warehous 0,054 factor 0,037 internet 0,032	mbe 0,071 visual 0,065 email 0,051 interfac 0,038 not 0,035	polylin 0,057 pair 0,043 approximation 0,035 parallel 0,029 intersect 0,028	attribut 0,046 constraint 0,045 corporation 0,035 formula 0,032 dimension 0,025
2001	index 0,108 tabl 0,107 join 0,102 vector 0,094 bitmap 0,051	temporal 0,134 join 0,129 r-tre 0,085 deby 0,044 semistructur 0,044	imag 0,145 itemset 0,082 pelican 0,052 maximal 0,050 frequent 0,043	mediator 0,094 juridical 0,043 repartition 0,043 generation 0,041 thesauru 0,029	medical 0,040 evolution 0,040 gis 0,038 collection 0,024 diseas 0,023	nam 0,041 attribut 0,035 olap 0,029 word 0,022 file 0,020
2002	protocol 0,086 extraction 0,078 encrypt 0,062 decipher 0,047 ontolog 0,046	mart 0,062 sourc 0,061 optimization 0,049 dig 0,045 pre-model 0,045	mediat 0,079 accurac 0,056 reliabl 0,050 xmls 0,047 correspondenc 0,042	odbm 0,063 dsmio 0,063 prism 0,050 revers 0,036 association 0,028	parallel 0,034 join 0,034 cross-languag 0,031 evaluat 0,028 odmg 0,027	mdbc 0,049 decision 0,039 prioritization 0,031 vsm 0,029 host 0,029
2009	probabilistic 0,196 inferenc 0,079 sql 0,048 safe 0,048 offlin 0,032	engin 0,090 cost 0,089 olap 0,055 pargr 0,052 centraliz 0,048	solution 0,078 challeng 0,048 section 0,040 art 0,039 dimensional 0,039	record 0,082 block 0,060 exampl 0,053 oracl 0,050 genetic 0,043	survival 0,040 solap 0,039 climat 0,032 xbrl 0,031 smtm 0,024	tutorial 0,045 subsumption 0,029 cover 0,023 googl 0,021 functional 0,019
2010	rank 0,101 relevanc 0,091 function 0,085 query-level 0,075 query-sensitiv 0,075	wclr 0,133 collection 0,114 clickthrough 0,106 featur 0,070 discriminativ 0,053	phrasal 0,204 term 0,164 ontolog 0,069 stream 0,057 detect 0,047	search 0,087 ontolog 0,071 form 0,059 keywor 0,043 keyword-bas 0,040	credibilit 0,064 classification 0,050 spatial 0,035 preferenc 0,035 lsh 0,029	video 0,079 cliqu 0,049 sentiment 0,048 youtub 0,028 cooperativ 0,028
2011	imag 0,179 descriptor 0,118 attribut 0,082 effectivenes 0,054 lazy 0,035	k-nearest 0,102 neighbor 0,102 operator 0,086 predicat 0,048 rewrit 0,041	rank 0,108 aggregation 0,062 learn 0,057 iets 0,048 keyword 0,043	workflow 0,109 grasp 0,071 data-intensiv 0,048 aco 0,030 pbs 0,030	ontolog 0,087 effect 0,045 classifier 0,036 temporally 0,029 adc 0,027	trajector 0,047 bias 0,044 ontolog 0,025 question 0,020 sla 0,019
2012	graph 0,360 fingerprint 0,107 kernel 0,089 comput 0,044 min-hash 0,035	cluster 0,175 densit 0,091 graph 0,090 internal 0,056 metric 0,045	memor 0,183 flash 0,076 devic 0,055 comput 0,054 file 0,040	etl 0,122 consumption 0,043 location 0,038 repositor 0,034 dysto 0,033	prediction 0,032 author 0,030 pim 0,030 sport 0,030 busines 0,022	crim 0,026 maliciou 0,026 pais 0,026 pplocator 0,026 trac 0,026

geográficos) especializa-se em três diferentes temas: Temas 4 (mapeamento de objetos espaciais) e 5 (recuperação de imagens), ambos de 1997, e Tema 6 (*hypermedia*) em 1998. Os temas 5 (indexação) e 6 (persistência) de 1999 estão próximos ao Tema 5 (integração de dados) de 2001. Observa-se a formação de um fluxo a partir dos Temas 5 e 6 de 2001 para o Tema 5 (recuperação de informação multilíngue) em 2002, após, para o Tema 3 (*data mining*) em 2004 e posteriormente especializa-se nos Temas 2 (similaridade de objetos), 3 (mineração de dados temporais) e 5 (regras de associação e classificação) em 2005. Também há um fluxo do Tema 5 (dados temporais) em 2004 para o Tema 3 em 2005. Em 2006, os temas 4 (versionamento) e 5 (representação de objetos) deslocam-se para o Tema 5 (banco de dados geo-temporais) em 2007, mais tarde influenciando o Tema 5 (similaridade e recuperação de imagens) em 2008, e por fim o Tema 5 (duplicação e redundância de dados) em 2009. Em 2010, o Tema 4 (ontologias) especializa-se nos temas 3 (ordenação e seleção) e 5 (classificação e ontologias) em 2011. O Tema 5 (classificação) em 2010 passa pelo Tema 5 em 2011 e finaliza no Tema 4 (eficiência e alocação de recursos).

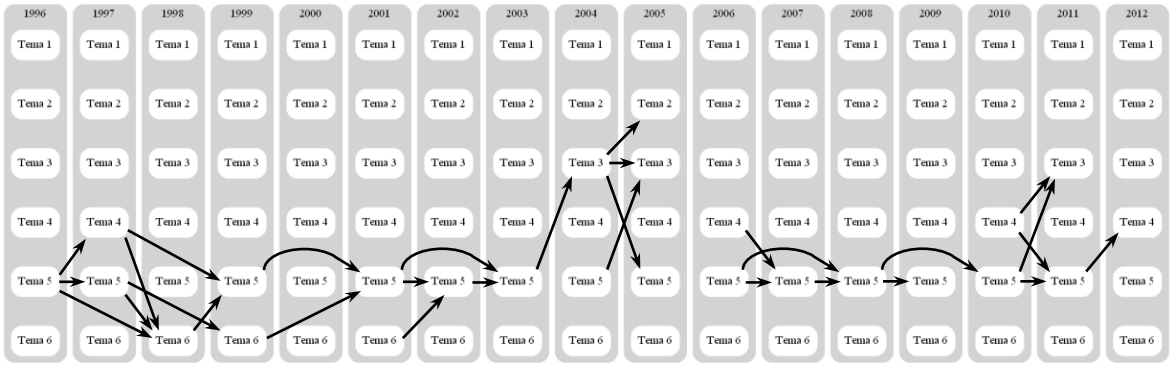


Figura 1. Transições entre os temas ao longo dos anos

5. CONCLUSÃO

A mineração temporal de dados é uma técnica que permite identificar as estruturas latentes em coleções de texto, facilitando o entendimento dos principais assuntos abordados em cada coleção. O método utilizado consiste formar agrupamentos (*clusters*) de palavras, identificando assim os temas da coleção. Neste trabalho, utilizamos os *abstracts* dos artigos publicados pelo SBBD para compor nossas coleções. Na etapa de classificação, identificou-se que os temas formados são bastante coerentes com o propósito do SBBD, isto permitiu uma análise das transições evolucionárias entre estes temas com o passar dos anos.

Os resultados mostram que, nos anos iniciais, os temas eram mais diferentes entre si a cada edição. Isto pode ser explicado pelo fato de que nos anos iniciais os artigos tratavam de tecnologias emergentes. Com o passar do tempo, essas tecnologias foram sendo combinadas gerando as transições evolucionárias.

Como trabalho futuro, pretendemos mensurar a intensidade dos temas em cada período. Assim será possível identificar os principais temas abordados. Seria interessante comparar os temas do SBBD com os de outro evento da área de banco de dados, como o VLDB, por exemplo.

Agradecimentos: O primeiro autor é aluno bolsista do CNPq. Este trabalho foi parcialmente financiado pelos projetos CNPq 478979/2012-6 e 480283/2010-9.

REFERÊNCIAS

Allan, J., Gupta, R., and Khandelwal, V. Temporal summaries of news topics. In *24th ACM SIGIR conference on Research and development in information retrieval*. ACM, New Orleans, Louisiana, USA, pp. 10–18, 2001.

Cover, T. M. and Thomas, J. A. *Elements of information theory*. Vol. 1. Wiley Online Library, New York, 1991.

Dempster, A. P., Laird, N. M., and Rubin, D. B. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)* vol. 39, pp. 1–38, 1977.

Kaur, K. and Gupta, V. A survey of topic tracking techniques. *International Journal of Advanced Research in Computer Science and Software Engineering* 2 (5): 384–393, 2012.

Mei, Q. and Zhai, C. Discovering evolutionary theme patterns from text: an exploration of temporal text mining. In *ACM SIGKDD international conference on Knowledge discovery in data mining*. KDD '05. ACM, Chicago, Illinois, USA, pp. 198–207, 2005.

Pinto, D., Benedí, J.-M., and Rosso, P. Clustering narrow-domain short texts by using the kullback-leibler distance. In *International Conference on Computational Linguistics and Intelligent Text Processing*. CICLing '07. Springer-Verlag, Mexico City, Mexico, pp. 611–622, 2007.

Sipos, R., Swaminathan, A., Shivaswamy, P., and Joachims, T. Temporal corpus summarization using submodular word coverage. In *ACM International Conference on Information and Knowledge Management*. CIKM '12. ACM, Maui, Hawaii, USA, pp. 754–763, 2012.

Zhai, C., Velivelli, A., and Yu, B. A cross-collection mixture model for comparative text mining. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '04. ACM, Seattle, WA, USA, pp. 743–748, 2004.

Processamento de consultas na Web de Dados: uma abordagem para busca de fontes de dados relevantes

Alberto Trindade Tavares, Bernadette Farias Lóscio

Centro de Informática - Universidade Federal de Pernambuco, Brasil
{att, bfl}@cin.ufpe.br

Abstract. The adoption of Linked Data principles has contributed towards the creation of a Web of Data, allowing the development of applications and tools which run queries over available information. One of the main challenges for the query processing over the Web is the selection of relevant sources, i.e., sources which could contribute significantly to the result of a query. In this paper, we discuss this problem and present an approach for identifying Web sources that may potentially be relevant to the processing of a set of queries. A distinct issue of our work is that the process of searching for sources employs the user requirements expressed in SPARQL queries.

Resumo. A adoção dos princípios do *Linked Data* tem contribuído para a construção de uma Web de Dados, permitindo o desenvolvimento de aplicativos e ferramentas que executam consultas sobre as informações disponibilizadas. Diante do crescente volume de dados desta natureza, um dos principais desafios para o processamento de consultas sobre a Web é a seleção de fontes relevantes, ou seja, aquelas capazes de contribuir de maneira significativa com os resultados de uma determinada consulta. Neste artigo, discutimos este problema e propomos uma abordagem para identificar fontes da Web de Dados que possam, potencialmente, contribuir com os resultados de um conjunto de consultas. Uma característica importante da abordagem apresentada é que o processo de busca de fontes faz uso dos requisitos de usuário expressos em consultas SPARQL.

Categories and Subject Descriptors: H.4 [**Information System Applications**]: Miscellaneous; H.2 [**Database Management**]: Miscellaneous

Keywords: Semantic Web, Linked Data, Web Crawling.

1. INTRODUÇÃO

A Web, por meio da publicação de documentos de hipertexto, estabeleceu um espaço global de informações. No entanto, a atual estrutura da mesma apresenta diversas limitações quanto ao processamento automático do seu conteúdo, pois os documentos são especificados em linguagens de marcação que fornecem, essencialmente, descrições sintáticas. Para lidar com esta restrição, diversas iniciativas foram desenvolvidas para facilitar o compartilhamento e processamento dos dados publicados. Dentre essas iniciativas, destaca-se a Web Semântica (*Semantic Web*), uma extensão da Web atual, onde os dados estão associados a um significado compreensível por máquinas. No contexto da Web Semântica, o termo *Linked Data* é utilizado para descrever um conjunto de práticas para a publicação de dados estruturados na Web, utilizando o modelo RDF para a representação do conhecimento.

A adoção dos padrões de *Linked Data* tem levado à construção de uma Web de Dados, caracterizado como um grafo global de dados formado por bilhões de triplas RDF. As informações disponibilizadas neste ambiente cobrem uma vasta gama de tópicos, tais como localizações geográficas, pessoas, empresas, livros, publicações científicas, dados estatísticos, entre outros [Bizer et al. 2009]. A publicação de dados interligados tem motivado o desenvolvimento de aplicações e ferramentas, pois a viabilidade de consultar

esta nuvem de dados, como se eles constituíssem um grande banco de dados distribuído, oferece diversas possibilidades. Entretanto, executar consultas sobre a Web de Dados ainda é um desafio para os desenvolvedores [Hartig 2013].

Neste trabalho, estamos interessados nas aplicações que acessam fontes de dados RDF, publicados de acordo com os princípios de *Linked Data*, onde este acesso é feito por meio de consultas escritas na linguagem SPARQL. Especificamente, este artigo aborda o problema da identificação de fontes, disponíveis na Web de Dados, relevantes para o processamento de um conjunto de consultas SPARQL. São consideradas fontes de dados relevantes, aquelas que podem contribuir com informações úteis do ponto de vista do usuário [Oliveira et al. 2012], ou seja, atendem aos requisitos dos usuários. Na abordagem proposta, consideramos que os requisitos de usuário são expressos nos padrões de triplas das consultas. O uso desta abordagem permite a detecção de fontes de dados inicialmente não conhecidas, mas que, potencialmente, irão contribuir com os resultados de consultas de uma aplicação.

O restante deste artigo é organizado como se segue. Na Seção 2, são apresentados os principais conceitos e termos relacionados às tecnologias da Web Semântica, fornecendo a fundamentação teórica para este trabalho. A Seção 3 descreve a abordagem proposta para a busca de fontes RDF na Web. A seção 4 mostra um exemplo prático de utilização da abordagem. Por fim, a Seção 5 conclui o artigo, citando contribuições deste trabalho e indicando pesquisas futuras.

2. FUNDAMENTAÇÃO TEÓRICA

Nesta seção, são apresentados alguns conceitos e terminologias que serão utilizados ao longo deste artigo.

2.1 URI e RDF

URI[1] é uma sequência de caracteres que identifica unicamente um recurso Web. O mecanismo básico para acessar recursos Web, segundo os padrões de *Linked Data*, se dá através de um processo chamado de dereferenciamento de URIs, que consiste no acesso via HTTP a uma URI, obtendo-se um conjunto de descrições RDF [Bizer et al. 2009]. O RDF[2], por sua vez, é um modelo de dados que permite descrever recursos na Web por meio de triplas, as quais podem ser organizadas como grafos direcionados. Os três componentes de uma tripla são: *Sujeito*, *Predicado* e *Objeto*.

2.2 SPARQL

SPARQL[3] é uma linguagem de consulta para dados RDF, permitindo a recuperação de informação contida em grafos. As principais partes de uma consulta SPARQL são [Pérez et al. 2009]: o *padrão da consulta*, que é composto por um conjunto de padrões de triplas (*Triple Patterns*), constituindo o denominado BGP (*Basic Graph Pattern*) da consulta, responsável por descrever os padrões com os quais as triplas resultantes devem estabelecer correspondência; os *modificadores de solução*, que permitem reorganizar o resultado da consulta e a *saída*, que especifica o formato do resultado. Fontes de dados interligados tipicamente disponibilizam um SPARQL *endpoint*[4], um serviço Web que permite ao usuário submeter consultas SPARQL sobre os dados RDF armazenados na fonte.

[1]http://www.w3.org/TR/uri-clarification/
[2]http://www.w3.org/RDF/
[3]http://www.w3.org/TR/rdf-sparql-query/
[4]http://semanticweb.org/wiki/SPARQL_endpoint

2.3 Web Crawler

A extração de dados na Web pode ser realizada através de Web *crawlers*, agentes de software que acessam a Web de maneira automatizada, navegando entre os recursos por meio de *links* [Castillo 2005]. A tarefa realizada por este agente é chamada de *crawling*. Um *crawler* inicia sua busca de dados na Web a partir de um conjunto de recursos de origem, denominados s*eeds* [Tavares et al. 2012a].

3. A ABORDAGEM PROPOSTA

Diante do número crescente de fontes de dados interligados que estão disponíveis na Web, se torna um desafio a execução de consultas sobre a Web de Dados. Uma das dificuldades nesta tarefa é a seleção das fontes que são capazes de responder a essas consultas. Neste trabalho, propomos uma abordagem que realiza um filtro nos conjuntos de dados, disponíveis na Web de Dados, usando como critério de seleção os requisitos de usuários que podem ser extraídos de consultas SPARQL. Especificamente, os requisitos de usuário considerados nesta abordagem são aqueles que podem ser extraídos a partir do BGP de uma consulta SPARQL e correspondem às URIs que representam um recurso (sujeito ou objeto) ou um predicado presente nos padrões de triplas do BGP.

A abordagem de busca de fontes proposta pode ser dividida em três etapas: i) Extração de recursos mais relevantes; ii) Realização de um *crawling* na Web para detectar fontes de dados interligados que descrevam os principais recursos extraídos na etapa anterior e iii) Classificação das fontes detectadas de acordo com a taxa de cobertura dos predicados das consultas da aplicação.

O Algoritmo *DatasetsSearch* (Algoritmo 1) apresenta o processo de busca proposto, descrevendo as etapas listadas acima. O algoritmo recebe, como entrada, um conjunto *Q* de consultas SPARQL e fornece, como saída, uma lista de fontes de dados, seguindo uma ordem de relevância, que são potencialmente capazes de responder as consultas em *Q*. Especificamente, a saída é composta por uma lista de SPARQL *endpoints*. O Algoritmo *DatasetsSearch* é detalhado nas subseções a seguir.

Algorithm DatasetsSearch
Input Q: A set of SPARQL queries
  k: Number of relevant resources to be considered during the crawling
Output *DE:* A list of SPARQL endpoints of fetched datasets
Begin
1. $RR \leftarrow \boldsymbol{ExtractRelevantResources}(Q)$
2. $Seeds \leftarrow SelectResources(RR, k)$
3. $Predicates \leftarrow \{sameAs,\ seeAlso,\ equivalentClass\}$
4. $FetchedTriples \leftarrow ExecuteCrawling(Seeds, Predicates)$
5. $ProvenanceTriples \leftarrow ExtractProvenance(FetchedTriples)$
6. $Endpoints \leftarrow \emptyset$
7. $\boldsymbol{For\ each}\ p \in ProvenanceTriples\ \boldsymbol{do}$
8. $\quad Endpoints \leftarrow Endpoints \cup RetrieveSparqlEndpoint(p)$
9. $\boldsymbol{End\ for}$
10. $QueryPredicates \leftarrow ExtractPredicates(Q)$
11. $DE \leftarrow \boldsymbol{RankDatasetsByPredicates}(Endpoints, QueryPredicates)$
12. $\boldsymbol{Return}\ DE$

End

Algoritmo 1. Algoritmo para Busca de Fontes de Dados

3.1 Extração de Recursos Relevantes

O primeiro passo da busca de fontes de dados é a identificação dos recursos mais relevantes a partir das consultas fornecidas como entrada. Consideramos, neste trabalho, que os recursos mais relevantes são os

mais frequentes no BGP da consulta. Estes recursos irão guiar a busca por fonte de dados na Web. A identificação é realizada através da função *ExtractRelevantResources*, apresentada no Algoritmo 2, que recebe como entrada um conjunto de consultas *Q* e retorna a lista dos recursos mais frequentes de *Q*. Especificamente, a função *ExtractRelevantResources* recupera o BGP de cada uma das consultas em *Q* e, para cada padrão de tripla de um dado BGP, seus elementos (sujeito, predicado e objeto) são processados por um *Visitor*. Ao longo da navegação desses elementos, temos a construção de uma lista de recursos, que estão presentes nos padrões de triplas das consultas, e de suas respectivas quantidades de ocorrência. A partir dessa lista, é gerada uma segunda lista, *RR* (*RelevantResources*), que armazena os recursos ordenados pela frequência, de forma decrescente.

3.2 Web Crawling para a Busca de Fontes

Uma vez que os recursos mais frequentes são identificados, o próximo passo é a execução de um *crawling* na Web de Dados, com o objetivo de buscar fontes relevantes para a execução das consultas. O processo de *crawling* considera como *seeds* os *k* primeiros recursos da lista *RR*, constituída por URIs que representam os recursos relevantes do conjunto de consultas *Q*. O conjunto de predicados {*rdfs:seeAlso*, *owl:sameAs* e *owl:equivalentClass*} é definido como outro parâmetro do *crawling*, indicando os *links* a serem seguidos pelo agente. O uso desses predicados permite a obtenção de novos recursos da Web que são similares aos recursos *seeds* [Ding et al. 2010].

Algorithm ExtractRelevantResources

Input Q: A set of queries
Output *RR:* A sorted list by frequency of query resources
Begin
1. $FrequencyList \leftarrow \emptyset$
2. **For each** $q \in Q$ **do**
3. $BGP \leftarrow ExtractBGP(q)$
4. **For each** $triplePattern \in BGP$ **do**
5. $Resources \leftarrow VisitTriplePattern(triplePattern)$
6. $FrequencyList \leftarrow FrequencyList \cup Resources$
7. **End for**
8. **End for**
9. $RR \leftarrow DecreasingElementsList(FrequencyList)$
10. **Return** RR

End

Algoritmo 2. Algoritmo para Extração de Recursos Relevantes

Algorithm RankDatasetsByPredicates

Input SE: A set of SPARQL endpoints
QP: A list of predicates
Output DE: A sorted list of SPARQL endpoints
Begin
1. $PredicatesRateByEndpoint \leftarrow NewMap()$
2. **For each** $e \in SE$ **do**
3. $PR \leftarrow RetrievePredicatesRate(e, QP)$
4. $PutKeyValueInMap(e, PR, PredicatesRateByEndpoint)$
5. **End for**
6. $DE \leftarrow GetKeysSortedByValue(PredicatesRateByEndpoint)$
7. **Return** RR

End

Algoritmo 3. Algoritmo para Classificação das Fontes Recuperadas

Ao final do *crawling*, temos um conjunto de triplas extraídas da Web de Dados que descrevem os recursos selecionados como *seeds*. O próximo passo do algoritmo é a construção da lista das fontes descobertas ao longo da busca. Para esta tarefa, são extraídas as proveniências das triplas coletadas pelo Web *crawler*. A informação de proveniência de uma tripla é representada por uma URI que indica a localização do seu arquivo RDF de origem. Para cada URI de proveniência, é utilizada a função *RetrieveSparqlEndpoint*, responsável por recuperar a URI do SPARQL *endpoint* das fontes de dados de procedência das triplas. O número de fontes, identificadas nesta etapa, é determinado pela quantidade de *links rdfs:seeAlso*, *owl:sameAs* e *owl:equivalentClass* subsequentes existentes a partir das fontes de origem do *crawling*, as quais são procedentes do dereferenciamento das URIs *seeds*.

3.3 Classificação das Fontes Detectadas

A última etapa do algoritmo é a classificação das fontes detectadas na busca de acordo com a taxa de cobertura dos predicados das consultas. Para uma determinada fonte de dados, esta taxa mede a porcentagem dos predicados presentes nos padrões de triplas das consultas *Q* que aparecem em triplas RDF da respectiva fonte. Essa métrica é utilizada por permitir a avaliação das fontes quanto à capacidade de satisfazer os padrões de correspondência definidos nas consultas. A classificação é realizada pela função *RankDatasetsByPredicates*, apresentada no Algoritmo 3. A função recebe, como parâmetros, o conjunto de SPARQL *endpoints* obtidos na etapa anterior e os predicados das consultas *Q*, que são extraídos por meio da função *ExtractPredicates*.

A função *RankDatasetsByPredicates* aplica, para cada um dos *endpoints*, a função *RetrievePredicatesRate* que irá calcular a taxa de cobertura de predicados de *Q* para a fonte associada ao *endpoint*. Esta função executa uma consulta SPARQL, através do *endpoint* dado como parâmetro, para cada um dos predicados das consultas, verificando se existe alguma tripla que possua o respectivo predicado. Por fim, os *endpoints* são ordenados segundo a taxa de cobertura de predicados, de forma decrescente, gerando a lista *DE* (*Dataset Endpoints*) que é fornecida como saída do Algoritmo 3. A lista de SPARQL *endpoints* retornada pela função *RankDatasetsByPredicates* é o resultado final do algoritmo *DatasetsSearch*. Os *endpoints* provêm à aplicação o acesso a fontes da Web de Dados, permitindo a execução das consultas SPARQL sobre as mesmas.

4. EXEMPLO DE UTILIZAÇÃO

Para ilustrar a abordagem proposta neste trabalho, esta seção apresenta um exemplo de utilização sobre o domínio bibliográfico. Para este domínio, diversas fontes estão disponíveis na Web de Dados fornecendo informações a respeito de autores, conferências, artigos, livros, entre outros tópicos relacionados. Considere três consultas SPARQL sobre este domínio, selecionando informações sobre o autor *Alon Halevy* e o evento *International Semantic Web Conference* (ISWC). Essas consultas são mostradas nas figuras 1, 2 e 3.

Q1. Retorne o título de artigos que foram publicados na ISWC 2012	*Q2. Retorne o nome dos autores que tiveram artigos publicados na ISWC 2012*	*Q3. Retorne o título de artigos que foram escritos por Alon Y. Halevy*
SELECT ?tituloArtigo WHERE *{* *{ ?artigo akt:cites-publication-reference id:conf/iswc/2012 . }* *{ ?artigo akt:has-title ?tituloArtigo . }* *}*	*SELECT DISTINCT ?nomeAutor WHERE* *{* *{ ?artigo akt:cites-publication-reference id:conf/iswc/2012 . }* *{ ? artigo akt:has-title ?tituloArtigo . }* *{ ?artigo akt:has-author ?autor . }* *{ ?autor akt:full-name ? nomeAutor . }}*	*SELECT ?tituloArtigo WHERE* *{* *{ ?artigo akt:has-author id:people-a86143 . }* *{ ? artigo akt:has-title ?tituloArtigo . }* *}*
Figura 1. Consulta #1	**Figura 2. Consulta #2**	**Figura 3. Consulta #3**

Para buscar fontes na Web de Dados que possam responder a essas consultas, vamos aplicar as três etapas da abordagem. A primeira tem o objetivo de selecionar os recursos relevantes dessas consultas, ou seja, URIs que representam um sujeito ou objeto e que fazem parte de algum dos três padrões de consultas considerados. A Tabela 1 apresenta o resultado desta fase, onde dois recursos são extraídos dos padrões de triplas das consultas e classificados de acordo com a frequência. A etapa seguinte consiste no *crawling* na

Web a partir dos dois recursos selecionados. Como resultado deste processo, são retornadas quatro fontes de dados, apresentadas na Tabela 2, juntamente com a URI do respectivo SPARQL *endpoint*. Entre essas fontes, a *DBLP RKBExplorer* é a única fonte procedente dos recursos *seeds*, sendo obtida no primeiro passo do processo de *crawling*. Por outro lado, as outras três fontes são descobertas ao longo da navegação entre os *links* RDF.

Tabela 1. Recursos Relevantes

URI do Recurso	# de Ocorrências
id:conf/iswc/2012	2
id:people-a86143	1

Tabela 2. Fontes Retornadas pelo Web *Crawler*

Nome da Fonte	URI do SPARQL *Endpoint*
DBLP RKBExplorer	http://dblp.rkbexplorer.com/sparql/
DBLP L3S	http://dblp.l3s.de/d2r/sparql
IEEE	http://ieee.rkbexplorer.com/sparql/
BibSonomy	-

Para a última etapa da abordagem, são consideradas somente as fontes de dados que disponibilizam um SPARQL *endpoint*, pois a interface para a execução de consultas sobre a base é fornecida por este serviço. Consequentemente, o *BibSonomy* é descartado, por não oferecer um *endpoint*, enquanto as outras três fontes são classificadas. A classificação das fontes é dada de acordo com a taxa de cobertura dos seguintes predicados: *akt:cites-publication-reference*, *akt:has-title*, *akt:has-author* e *akt:full-name*. Como resultado, as três fontes são igualmente classificadas, pois as mesmas possuem, entre as suas triplas RDF, todos os quatro predicados. O SPARQL *endpoint* dessas bases de dados são retornadas como a saída do Algoritmo, permitindo, a uma aplicação, a execução das consultas sobre tais fontes.

5. CONCLUSÃO

Neste artigo, apresentamos uma abordagem para a busca de fontes da Web de Dados que são, potencialmente, capazes de contribuir com os resultados de consultas de uma determinada aplicação. Essa abordagem faz uso dos requisitos de usuário expressos nas próprias consultas, que estabelecem o escopo da busca a ser executada. Como trabalhos futuros, gostaríamos de destacar algumas direções:

(i) Extensão de um protótipo desenvolvido para a busca de fontes [Tavares et al. 2012b], dando suporte ao uso de predicados de consultas para a classificação das fontes.

(ii) Realização de experimentos utilizando a nova versão do protótipo, para fins de validação da abordagem.

REFERÊNCIAS

BIZER, C., HEATH, T. and BERNERS-LEE, T. 2009. Linked data - the story so far. In *Proceedings of the International Journal on Semantic Web and Information Systems*, 5(3), 1-22.

HARTIG, O. 2013. An Overview on Execution Strategies for Linked Data Queries. In *Datenbankspektrum*, 13(2).

PÉREZ, J., ARENAS, M. and GUTIERREZ, C. 2009. Semantics and complexity of SPARQL. In *Proceedings of the ACM Transactions on Database Systems*, TODS 2009, Nova York, NY, USA, 34(3).

OLIVEIRA, H. R., TAVARES, A. T. and LÓSCIO, B. F. 2012. Feedback-based Data Set Recommendation for Building Linked Data Applications. In *Proceedings of the International Conference on Semantic Systems*, I-SEMANTICS 2012, Graz, Austria.

TAVARES, A. T., OLIVEIRA, H. R. and LÓSCIO, B. F. 2012. RDFMat – Um serviço para criação de repositórios de dados RDF a partir de crawling na Web de dados. In *I Escola Regional de Informática de Pernambuco*, 2012, Recife, Brasil.

CASTILLO, CARLOS. 2005. Effective Web Crawling. In ACM SIGIR Forum, Vol.39, Nova York, NY, USA.

DING L, SHINAVIER J, SHANGGUAN Z, MCGUINNESS DL. 2010. SameAs networks and beyond: analyzing deployment status and implications of owl: sameAs in linked data. In *Proc of the 9th International Semantic Web Conference*, ISWC 2010, Shanghai, China.

TAVARES, A. T., OLIVEIRA, H. R. and LÓSCIO, B. F. 2012. *Buscando Fontes de Dados Relevantes para Aplicações Linked Data. XVIII Simpósio Brasileiro de Sistemas Multimídia e Web*, WebMedia 2012, São Paulo. IX Workshop de Trabalhos de Iniciação Científica, 2012, p. 119-122.

Mecanismo de Encadeamento de Notícias por Reconhecimento de Implicação Textual

Phillipe S. Cavalcante, Wallace A. Pinheiro

Instituto Militar de Engenharia, Rio de Janeiro - RJ - Brasil
{cavalcante.phillipe, wallaceapinheiro}@gmail.com

Abstract. Atualmente, o acesso à informação relevante é feito com uma simples consulta a um sistema de busca na Web. Em se tratando de notícias, além de um sistema de consulta para o acesso à informação, sites especializados sugerem ao usuário notícias relacionadas para leitura complementar. No entanto, estas notícias relacionadas nem sempre complementam o conteúdo da primeira notícia lida pelo usuário. Os principais problemas desta abordagem são a existência de conteúdo redundante e a falta de ordenação segundo um critério que proporcione uma leitura textual coerente para o usuário, isto é, uma leitura que forneça notícias organizadas de tal modo que proporcione ao usuário uma progressão do conteúdo que está sendo lido. Este trabalho apresenta um mecanismo de encadeamento textual capaz de organizar notícias relacionadas segundo um critério de implicação textual, proporcionando ao usuário uma leitura textual coerente. Para isso, o mecanismo utiliza um sistema de reconhecimento de implicação textual para o encadeamento de notícias, e um sistema de similaridade para identificação de notícias com conteúdo semelhante. Os experimentos realizados mostram que o mecanismo produz encadeamentos textuais semelhantes aos encadeamentos organizados pelos usuários.

Categories and Subject Descriptors: H.3.3 [**Information Search and Retrieval**]: Information filtering, Retrieval models; I.7 [**Document and Text Processing**]: Miscellaneous; I.2.7 [**Natural Language Processing**]: Language parsing and understanding

Keywords: Encadeamento de Notícias, Reconhecimento de Implicação Textual, Notícias Relacionadas

1. INTRODUÇÃO

Sistemas de busca têm proporcionado ao usuário acesso à informação relevante e atualizada na *Web*. Contudo, estes sistemas recuperam tanta informação relacionada que escolher por onde começar ou continuar uma leitura tem se tornado uma tarefa trabalhosa para o usuário [Mulder et al. 2006; Kobayashi and Takeda 2000]. Essa tarefa é bastante comum em sites de notícias e tem sido tratada com o uso de sistemas de recomendação [Adomavicius and Tuzhilin 2005].

Sistemas de recomendação de notícias, em geral, sugerem notícias de acordo com um critério de similaridade textual [Lv et al. 2011]. No entanto, estas sugestões nem sempre complementam a notícia inicialmente lida pelo usuário. Os principais problemas encontrados nesta abordagem são: (*i*) redundância de conteúdo, uma vez que o critério de similaridade agrupa notícias semelhantes; e (*ii*) a falta de ordenação das notícias, de modo a proporcionar ao usuário uma progressão do conteúdo que está sendo lido.

O sistema proposto por [Rodrigues et al. 2010] aborda os dois problemas apresentados e utiliza o conceito de sabedoria das multidões para a sugestão de notícias em um encadeamento causal e temporal, mas, tem a necessidade de solicitar *feedback* do usuário para realizar o encadeamento. Outra solução é apresentada por [Shahaf and Guestrin 2011; 2012], que propõe um método de encadeamento textual entre duas notícias previamente selecionadas pelo usuário, porém, este método se torna inviável para a recomendação de notícias, já que a única informação diponível é a notícia selecionada pelo usuário.

Este artigo apresenta um mecanismo de encadeamento de notícias capaz de fornecer notícias organizadas segundo um critério de implicação textual, proporcionando ao usuário uma progressão do conteúdo que está sendo lido. Para isso, o mecanismo utiliza um sistema de reconhecimento de implicação textual, para o encadeamento das notícias, e um sistema de similiaridade textual, para identificação de notícias semelhantes.

Os resultados obtidos com este mecanismo foram comparados com o sistema proposto por [Rodrigues 2011; Rodrigues et al. 2010] e mostraram que o mecanismo de encadeamento de notícias por implicação textual tem uma acurácia maior do que a apresentada pelo encadeamento de notícias segundo um critério causal e temporal.

2. RECONHECIMENTO DE IMPLICAÇÃO TEXTUAL E SIMILARIDADE TEXTUAL

O reconhecimento de implicação textual identifica quando existe inferência semântica entre dois textos. Dados dois textos, t e h, t implica textualmente em h quando o significado de h é provável de ser inferido do significado de t [Tatar et al. 2009; Zanzotto et al. 2009].

Por exemplo, entre os textos t e h, a seguir, ocorre implicação textual:

t: O Brasil foi a sede da copa do mundo de 2014, e
h: O Brasil participou da copa do mundo de 2014.

Neste exemplo, h pode ser inferido semanticamente de t; se o Brasil foi a sede da copa do mundo de 2014, então o Brasil participou da copa do mundo de 2014. Portanto, dizemos que existe uma implicação textual entre t e h. Observa-se que não existe implicação textual no sentido contrário; a participação do Brasil na copa do mundo de 2014 não quer dizer que ele tenha sido o anfitrião da copa do mundo de 2014, por mais que isso seja verdade. A implicação textual não é tão formal quanto a implicação lógica, pois ela aceita casos em que a verdade de h seja plausível, em vez de certa.

O TF-IDF é um coeficiente de similaridade textual que indica o quão importante é uma palavra para um documento, em relação a um conjunto de documentos [Zhang et al. 2008]. Este coeficiente é utilizado no módulo de similaridade (ver figura 1) do mecanismo para a formação do índice de similaridade textual. O índice de similaridade é uma matriz de ordem $n \times n$, onde n é a quantidade de notícias do sistema. Neste trabalho, as notícias foram transformadas em vetores onde cada dimensão indica o TF-IDF da palavra que está presente no documento. Cada componente do índice de similaridade é obtida através do cosseno entre cada par de notícias [Zhang et al. 2008].

Outra medida de similaridade utilizada neste artigo é o coeficiente de Jaccard: $J(A, B) = \frac{|A \cap B|}{|A \cup B|}$, onde A e B são dois conjuntos de termos, os documentos a serem comparados. O coeficiente de Jaccard avalia a similaridade e a diversidade das palavras presentes nos documentos [Rodrigues 2011]. Este coeficiente é utilizado no extrator de características do módulo de implicação textual, apresentado na figura 2.

3. MECANISMO DE ENCADEAMENTO NOTÍCIAS

O mecanismo de encadeamento de notícias é capaz de ordenar as notícias de acordo com um critério de implicação textual. Para isso, o mecanismo utiliza um sistema de reconhecimento de implicação textual e um sistema de identificação de similaridade textual. A figura 1 ilustra o mecanismo proposto.

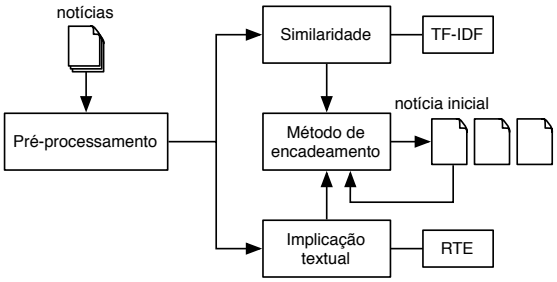


Fig. 1. Mecanismo de Encadeamento de Notícias.

O mecanismo é formado por dois módulos principais: (*i*) módulo de similaridade, responsável por gerar um índice de similaridade; e (*ii*) módulo de implicação textual, responsável por treinar a máquina de reconhecimento de implicação textual e gerar um índice de implicação textual (ver figura 2).

O módulo de *pré-processamento* é responsável pela limpeza textual das notícias do sistema. Algumas funções deste módulo são: remoção de *stopwords* e pontuações, conversão de caracteres para UTF-8 e transformação para caracteres minúsculos, e obtenção de lema das palavras.

O encadeamento textual é gerado por um algoritmo de estratégia gulosa, localizado no módulo *método de encadeamento*. Este algoritmo busca no módulo de *implicação textual* uma notícia que implique semanticamente na *notícia inicial*, escolhida pelo usuário, e a adiciona à lista encadeada de notícias. O primeiro elemento da lista encadeada é a própria notícia inicial.

Caso exista mais de uma notícia implicante, o algoritmo compara o valor de similaridade entre as notícias implicantes - esses valores são encontrados no módulo de *similaridade* - e escolhe aquela notícia com o menor valor de similaridade, adicionando-a à lista encadeada. Este processo se repete para a escolha da próxima notícia implicante: a notícia inicial passa a ser a notícia implicante até que não haja notícia implicante à notícia anterior.

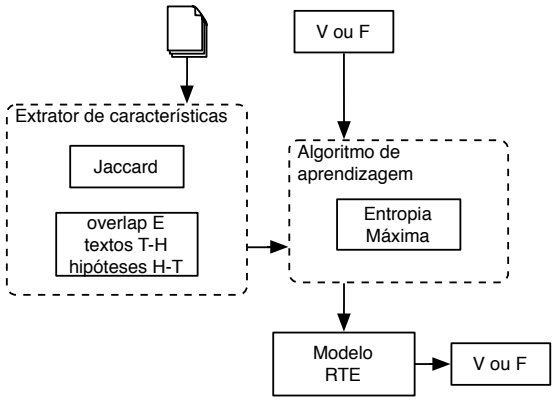


Fig. 2. Módulo de Implicação Textual.

A figura 2 apresenta a arquitetura do módulo de *implicação textual*. Este módulo contém um extrator de características da base de treinamento de Reconhecimento de Implicação Textual (RTE), esta base foi obtida de [Dagan et al. 2005] e traduzida para o português, e um algoritmo de aprendizagem supervisionada, baseado em Entropia Máxima [Malouf 2002]. O primeiro é responsável pela extração de características de similaridade baseadas no coeficiente de Jaccard. O segundo, gera um classificador que recebe como entrada pares de notícias e classifica o relacionamento entre elas como sendo de implicação textual ou não. Este componente identifica relações entre o fim de uma notícia e o início de outra, para a formação do encadeamento [Rodrigues 2011].

As características extraídas pelo módulo de implicação textual foram, dado duas notícias t e h:

(1) O conjunto de termos que existem em t, mas não existem em h;

(2) O conjunto de termos que existem em h, mas não existem em t;

(3) O conjunto de termos que t e h têm em comum;

3.1 Avaliação do Encadeamento

A avaliação do encadeamento gerado pelo mecanismo é feita através da observação do coeficiente de correlação de Spearman (ρ) [Rodrigues 2011]. Este coeficiente é um indicador da relação de ordem entre duas variáveis, X e Y. Para a obtenção do cálculo do coeficiente, dada uma amostra de tamanho n, primeiro as variáveis X e Y são ordenadas e transformadas em x e y. O coeficiente de Spearman é obtido pela equação: $\rho = 1 - \frac{6 \cdot \sum d_i^2}{n \cdot (n^2 - 1)}$, em que $d_i = x_i - y_i$ é a diferença entre as posições de cada observação das variáveis.

Os valores deste coeficiente variam entre -1 e 1. Quando o valor tende a 1, significa que a ordenação da variável Y segue a ordenação da variável X. Quando o valor tende a -1, significa que Y segue a ordenação invertida da variável X. Para este experimento, por exemplo, supondo que Y seja o encadeamento do mecanismo e que X seja o encadeamento feito pelo usuário, um valor de $\rho = 1$ (ou ρ tendendo a 1) diz que o encadeamento feito pelo mecanismo é similar ao encadeamento do usuário (ou tende ao encadeamento do usuário).

A escolha da avaliação usando o coeficiente de Spearman e a quantidade de usuários para o experimento é justificada em [Rodrigues 2011]. A principal razão para a escolha deste coeficiente é que ele avalia a ordenação entre duas variáveis. Com relação a amostragem de notícias feita para o experimento, a quantidade de notícias utilizadas foi de 5, todas relacionadas a um mesmo tópico. As notícias foram extraídas do Google News (news.google.com) sobre o tópico Crise na Coréia do Norte. Elas foram selecionadas considerando o tempo de publicação, a similaridade, e se juntas formavam um encadeamento textual. As notícias continham, em média, 300 palavras. Experimentos iniciais com 10 notícias causaram desconforto no usuário, quanto a ordenação das notícias segundo o critério de encadeamento textual. Isto é, a partir da sexta notícia o usuário já apresentava dificuldades de lembrar o conteúdo presente nas 5 notícias anteriores para poder ordenar as próximas 5 notícias.

A quantidade de participantes escolhida para o experimento foi de 10. Os participantes tinham entre 21 e 28 anos, e grau de instrução mínimo de Mestrado em Ciência da Computação. O artigo de [Rodrigues 2011] mostra que 9 é um número suficiente para a realização do experimento, baseando-se na proporção populacional do Brasil. A fórmula utilizada para o cálculo da quantidade de usuários é a seguinte: $n = \frac{Z_{\alpha/2}^2 \cdot p \cdot q}{E^2}$, onde: n é o número desejado de usuários (10); $Z_{\alpha/2}$ é o valor crítico que corresponde ao grau de confiança desejado (80%); p é a proporção populacional pertencente a categoria que se deseja estudar (50%); q é a proporção populacional que não pertence à categoria que se deseja estudar ($q = 1 - p$); e E é a margem de erro, que indica a diferença máxima entre a proporção da amostra utilizada e a verdadeira proporção populacional (20%).

4. RESULTADOS

Os resultados mostraram que o encadeamento de notícias, gerado pelo mecanismo, seguiu a ordenação das sequências encadeadas pelos usuários, conforme a figura 3. Na figura, o eixo horizontal representa o encadeamento realizado por cada usuário (de 0 a 9), e o eixo vertical representa o valor de Spearman entre o encadeamento gerado pelo mecanismo e o encadeamento de notícias de cada usuário. O coeficiente de Spearman obtido foi, em média, de $\rho = 0.84$, o que indica que as notícias geradas pelo mecanismo tiveram uma ordenação próxima da ordenação realizada pelo usuário.

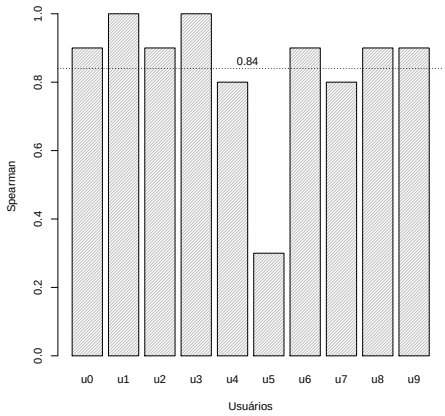


Fig. 3. Comparação do encadeamento do Mecanismo com o encadeamento dos usuários.

Além disso, o mecanismo produziu resultados melhores que os apresentados em [Rodrigues 2011] ($\rho = 0.3$) para um conjunto de notícias relacionadas a apenas um tópico.

Uma análise feita com os dados utilizados para o experimento, mostrou que a mediana do coeficiente de Spearman foi de 0.9 e que o valor mínimo foi de 0.3. Este valor mínimo é um *outlier*, por ser o único resultado fora do padrão encontrado.

Na tabela I é apresentado o encadeamento de notícias formado pelos usuários e pelo mecanismo. Na tabela, as notícias estão enumeradas de 1 a 5. Na primeira linha, encontra-se a lista de usuários submetidos ao experimento, enumerados de 0 a 9, e o mecanismo. Da primeira à décima coluna tem-se o encadeamento feito por cada usuário e a distância entre o encadeamento feito pelo mecanismo e por cada usuário, separados pelo símbolo /. Dessa forma, como o usuário 0 e o mecanismo escolheram a notícia 1 para ser a primeira notícia do encadeamento, a distância entre a posição da primeira notícia sugerida pelo mecanismo e a primeira notícia escolhida pelo usuário é igual a 0, ou seja, 1/0. Já no caso do usuário 5, a notícia escolhida para ser a primeira do encadeamento foi a notícia 4. Assim, a distância entre a posição da primeira notícia sugerida pelo mecanismo e a primeira notícia escolhida pelo usuário é igual a 3, ou seja, 4/3.

Table I. Encadeamento de Notícias: Usuário × Mecanismo

u_0	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8	u_9	Mecanismo
1/0	1/0	1/0	1/0	2/1	4/3	1/0	2/1	1/0	1/0	1
2/0	2/0	2/0	2/0	1/1	1/1	2/0	1/1	2/0	2/0	2
3/0	3/0	4/1	3/0	3/0	3/0	3/0	3/0	3/0	4/1	3
5/1	4/0	3/1	4/0	5/1	2/2	5/1	5/1	5/1	3/1	4
4/1	5/0	5/0	5/0	4/1	5/0	4/1	4/1	4/1	5/0	5

Uma análise sobre os valores da tabela I mostra que o número de acertos feito pelo mecanismo é de aproximadamente 58%, considerando que a base de cálculo para isso seja $1 - \frac{e}{c}$, onde e é a quantidade total de erros (21) e c é a quantidade total de comparações realizadas para cada posição do encadeamento (50, número de usuários por quantidade de notícias).

5. CONCLUSÃO E TRABALHOS FUTUROS

O principal objetivo do mecanismo proposto é o encadeamento de notícias relacionadas segundo um critério de implicação textual. Para isso, o mecanismo utiliza técnicas de reconhecimento de implicação textual e de similaridade entre textos.

Os resultados obtidos mostraram que o mecanismo é capaz de fornecer encadeamentos de notícias mais próximos da necessidade do usuário, solucionando os problemas de redundância textual e ordenação, apresentados no início deste trabalho, de uma forma plausível. Além disso, a tarefa de encadeamento se mostrou capaz de ordenar as notícias no tempo, proporcionando uma leitura textual coerente. Sendo assim, um passo importante para a geração automática de textos para a contagem de uma história, do início ao fim.

O uso de técnicas para geração de árvores sintáticas sobre os textos durante o processo de implicação textual é uma abordagem que pode trazer melhorias à geração do encadeamento textual, bem como a utilização de lógica de primeira ordem, para a computação semântica das expressões de linguagem natural. Tais técnicas podem ser implementadas em trabalhos futuros.

REFERENCES

Adomavicius, G. and Tuzhilin, A. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Trans. on Knowl. and Data Eng.* 17 (6): 734–749, June, 2005.

Dagan, I., Glickman, O., and Magnini, B. The PASCAL Recognising Textual Entailment Challenge. In *Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*, 2005.

Kobayashi, M. and Takeda, K. Information retrieval on the web. *ACM Comput. Surv.* 32 (2): 144–173, June, 2000.

Lv, Y., Moon, T., Kolari, P., Zheng, Z., Wang, X., and Chang, Y. Learning to model relatedness for news recommendation. In *Proceedings of the 20th international conference on World wide web*. WWW '11. ACM, New York, NY, USA, pp. 57–66, 2011.

Malouf, R. A comparison of algorithms for maximum entropy parameter estimation. In *proceedings of the 6th conference on Natural language learning - Volume 20*. COLING-02. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 1–7, 2002.

Mulder, I., de Poot, H., Verwij, C., Janssen, R., and Bijlsma, M. An information overload study: using design methods for understanding. In *OZCHI '06: Proceedings of the 20th conference of the computer-human interaction special interest group (CHISIG) of Australia on Computer-human interaction: design: activities, artefacts and environments*. ACM, New York, NY, USA, pp. 245–252, 2006.

Rodrigues, T. S. *WhySearch: Um Mecanismo Causal e Temporal de Encadeamento de Notícias*. M.S. thesis, Universidade Federal do Rio de Janeiro, 2011.

Rodrigues, T. S., Pinheiro, W. A., Souza, J. M., and Xexeo, G. Relacionando Notícias Web: Uma Abordagem Causal e Temporal. In *9th International Information and Telecommunication Technologies Symposium*, 2010.

Shahaf, D. and Guestrin, C. Connecting the dots between news articles. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence - Volume Volume Three*. IJCAI'11. AAAI Press, Barcelona, Catalonia, Spain, pp. 2734–2739, 2011.

Shahaf, D. and Guestrin, C. Connecting Two (or Less) Dots: Discovering Structure in News Articles. *ACM Trans. Knowl. Discov. Data* 5 (4): 24:1–24:31, Feb., 2012.

Tatar, D., Serban, G., and Andreea, M. Textual Entailment as a Directional Relation. *Journal of Research and Practice in Information Technology - ACJ*, 2009.

Zanzotto, F. M., Pennacchiotti, M., and Moschitti, A. A Machine Learning Approach to Textual Entailment Recognition. *Natural Language Engineering* vol. 15, pp. 551–582, 10, 2009.

Zhang, W., Yoshida, T., and Tang, X. TFIDF, LSI and Multi-word in Information Retrieval and Text Categorization. In *Systems, Man and Cybernetics, 2008. SMC 2008. IEEE International Conference on*. pp. 108–113, 2008.

Orbit: Efficient Processing of Iterations

Douglas Ericson M. de Oliveira, Fabio Porto

Extreme Data Lab (DEXL), Laboratório Nacional de Computação Científica
ericson@lncc.br, fporto@lncc.br

Abstract. Various scientific applications compute iterations on a huge set of input data. Examples include parameter sweep, which processes the same model through hundreds or thousands of input data, scientific visualization and numeric methods to solve hyperbolic equations. The state of the art for executing such applications is to model them as scientific workflows and to run them in HPC infrastructure. The execution model strives to combine pipeline parallelism, among activities of the workflow, with intra-workflow parallelism over data partitions, both of them subjected to workflow characteristics. In such scenario, the scientific workflow execution model can be approximated to that of parallel query processing. In fact, database groups at LNCC and COPPE have implemented parallel workflow engines, QEF and Chiron, under this assumption. In order to full integrate iterations within a data processing execution model, an extension is required to treat the loop control operator. In this paper we investigate the problem of efficiently executing scientific workflows that include iterations. We introduce Orbit which is a generic operator to manage the data flow in an iterative procedure. An execution model centered into Orbit is proposed including a centralized and parallelized modes. Additionally, two new execution strategies are investigated: first-tuple-first, first-iteration-first. We have obtained initial results that evaluate the different execution strategies in both centralized and parallel modes.

Categories and Subject Descriptors: H.Information Systems [**H.m. Miscellaneous**]: Databases

General Terms: Scientific Workflows, Execution Model

Keywords: Orbit, Control Operator, Query Processing, iteration

1. INTRODUCTION

The adoption of scientific workflow model to implement large scale science experiments and data analysis applications is the state of the art in eScience. Systems designed to process scientific workflows are known as workflow engines and a handful of them are available for the community to use. Among the different workflow engines, a particular architecture that is of interest to this paper evaluates huge volume of data in parallel computers. Applications such as parameter sweep, wherein a scientific model is evaluated against a huge parameter value space; and scientific visualization, in which the results of numeric simulations are browsed in time-space bringing a realistic view of a phenomenon, are examples of scientific applications that iterate over a number of steps or over input instances.

In previous work [Porto et al. 2007; Ogasawara et al. 2011], we have highlighted the similarities between the execution model of scientific workflows and that of query processing. Under this perspective, a workflow model is a generalization of data processing models of which query processing is a simple case. In particular, the execution of queries is governed by query execution plans represented as trees, left deep or bushy [Ozsu and P.Valduriez 2011]. This restricted topology is extended in workflow models. The reason why this analogy is relevant is that it may lead to the application of cost based query optimization strategies to workflow models, in addition to the reuse of query processor machinery to run scientific workflow. That is exactly the path that the QEF system [Porto et al. 2007] has followed. Thus, it is instructive to identify workflow models whose execution behavior would not be supported by workflow engines and would limit the applicability of query optimization techniques.

In this context, the applications referred above are examples of such category. Indeed, the work

on query optimization has developed on the relational algebra [Codd 1970] whose semantics does not include program flow control structures, such as iterations. A line of work explored the execution of transitive closure [Sippu and Soisalon-Soininen 1988] over tuples of a relation that basically consider a special case of iteration, one that appears in self-joins. Note that we are interested in more general iterations, in which tuples may iterate over a fragment of the workflow.

In order to illustrate a typical application, consider the virtual particle trajectory (VPT) visualization application. At the Hemolab Laboratory [1], a computational model of the cardio-vascular system simulates the flow of blood through arteries. The VPT application receives a mesh, geometrically representing parts of a human artery, and, for each point in the mesh, the velocity of the flow at each time instant. Moreover, VPT receives a set virtual particle representations of the blood with a initial position in space-time. The workflow includes three operations: a spatial join, that matches each particle position within a mesh geometrical figure; a join in space-time that for a given set of points look for their velocity in a time instant; and a trajectory computing program that calculates the next position for the virtual particle. Thus, this fragment of the workflow must be iterated the number of times corresponding to the time-range of the simulation. The iterative evaluation of a fragment of a workflow introduces a cycle into graph representation of the workflow.

In this context, this paper contributes to the execution model of scientific workflows with a data processing operator, named *Orbit*, that implements data iteration through a fragment of a workflow. Orbit introduces cycles into workflow execution while retaining the generic behavior of query processing operators bringing forth the integration of query processing strategies with scientific workflow execution models. An experiment has been conducted comparing two execution models: $First\_Tuple\_First$ and $First\_Iteration\_First$, and two distribution models: Master and Remote.

2. RELATED WORK AND CONCEPTS

Processing huge amount of data is currently a hot topic, sometimes encapsulated around the buzzword *Big Data*. An important initiative in these lines is to extend databases with user defined functions (UDF) [Stonebraker and Rowe 1986], which has been adopted within the Sloan Digital Sky Survey project [2], or adding full execution environments, such as [Liae et al. 2008], which deals with iteration within the package. Other approaches, such as Oracle pl/sql add full fledged programming language to be used in stored-procedures and UDFs. None of these initiatives deal with the introduction of iteration such that it can be taken into consideration during query optimization.

Another area of research with results over time is related to the computation of the transitive closure operation on relations, with special attention given by the expression of recursion in Datalog [Abiteboul et al. 1994]. In both cases, the operation is evaluated as a self-join. A more recent initiative Haloop [Bu et al. 2010] implements iterations as a sequence of joins with aggregates using the Hadoop system [Had 2013], focusing in collocating at the same node operations that communicate data.

In the workflow scenario, such approaches need to be extended to cope with data iteration through a fragment of the workflow. Orbit has been inspired by the n-ary Eddy operator [Avnur and Hellerstein 2000]. Eddy has been proposed to introduce adaptivity to a query execution plan. In fact, the operator achieves an iterative behavior almost by chance, while breaking the full ordering of operators in a query execution plan. Thus, in the perspective of this paper, Eddy could be considered to implement two different behaviors:iteration and adaptivity. In this line, the present work clarifies this issue in a way that Orbit could be used in association with another operator to deal with adaptation. More importantly, Orbit implements iteration irrespectively of an adaptive or fixed execution strategy.

[1] http://macc.lncc.br
[2] http://www.sdss.org

2.1 QEF

Orbit has been conceived as an operator following the known *iterator* interface [Graefe 1990], thus it can be integrated in any modern query processor. Our implementation, nevertheless, was done using the QEF (*Query Engine Framework*).QEF is a data processing engine designed as an extension based on query processor technology. In QEF a data processing application is implemented by extending three main structures: Data Unit, Algebraic Operators and Control Operators.

A QEF workflow defines a DAG, where nodes correspond to operators and directed edges represent the flow of data from a producer to a consumer operator. In this work, we have extended QEF model by supporting cyclic workflows with the introduction of the Orbit control operator.

3. THE ORBIT OPERATOR

In this section, we introduce the Orbit operator. We initially give some basic definitions used during the operator specification.

Definition 3.1. A workflow model is a partial ordered set of operations. Each operation consumes and produces a Data Unit. An operator op_j succeeds op_i, $op_j \succ op_i$, in a workflow model w, if there is a data item d that is produced by op_i and consumed by op_j, directly or indirectly.

Definition 3.2. A Fragment of Workflow Model ϕ is a subset of operators in a workflow model w, such that for each pair of operators op_i and op_j, either $op_i \succ op_j$ or $op_j \succ op_i$ in ϕ. A fragment of workflow is limited by a bottom and top operators. A bottom operator in a fragment of workflow ϕ, $op_b \in \phi$, is such that op_b directly $\succ op_l$ and $op_l \notin \phi$. The bottom operator directly succeeds the top operator, in a fragment of a workflow.

Definition 3.3. Cyclic fragment of a workflow (CFW) - is such that exist two operators $op_i, op_j \in \phi$, with $op_j \succ op_i$ and $op_i \succ op_j$.

Given a cyclic fragment of a workflow ϕ, the Orbit operator is placed in ϕ such that the bottom operator in ϕ directly $\succ$ Orbit, and the latter directly $\succ$ top.

3.1 Semantics

Orbit is a quaternary control operator defined as *Orbit(Tuple inpipeline, Tuple outpipeline, Tuple inorbit, Tuple outorbit)*, such that *inpipeline* and *outpipeline* correspond to the tuples coming from *producer* and leaving to *consumer* operators, respectively (see below). Conversely, within the CFW, the *inorbit* and *outorbit* correspond to tuples feeding, and returning from, the internal loop, respectively. The operator is placed in a workflow to implement the cyclic behavior in a fragment of a workflow model. A cyclic workflow model enables the iterative evaluation of tuples through the operators taking part in the workflow fragment.

Figure 1 presents Orbit components, also showing the relationship among its respective producers/consumers. Each one of these components, together with their corresponding responsibilities in the cyclic execution model implemented with Orbit are described next.

—Producer: is responsible for providing the tuples to be iterated by the cyclic fragment of the workflow (CFW). Orbit implements the iterator operator interface [Graefe 1990] and issues *getNext* calls to the producer operator. An independent thread is charged to call this function and to store the tuples into the Orbit buffer. This thread is controlled by a semaphore that warns when a tuple can or cannot be loaded into the buffer. Therefore, a new tuple cannot be loaded until old tuples have not finished their execution (i.e. either finished their iterations or are eliminated by one operator within

the fragment), otherwise the thread is blocked. In this paper, such thread is called a feeding thread. The buffer stores all tuples generated by the *Producer* operator that have not yet been consumed by the *Consumer* operator and that are not being processed by the fragment of the workflow;

—Consumer: is an operator $op_l \notin \phi$, such that op_l directly $\succ$ Orbit. It obtains tuples that have achieved the desired number of iterations.

—CFW Producer Operator: corresponds to the top operator of a CFW. It is succeeded by the Orbit operator. Orbit evaluates whether the tuple produced by the CFW Producer Operator is subject to a new iteration, in which case it is stored in the buffer that feeds the CFW Consumer Operator;

—CFW Consumer Operator: conversely, it corresponds to the bottom operator in a CFW. It consumes tuples that are in the Orbit buffer, according to the Iterator model. It is worth observing that the consumed tuples are indeed eliminated from the buffer, but they may eventually be returned to it during Orbit evaluation;

—Functions: as already described, Orbit main goal is to continuously evaluate a tuple until it reaches a certain requirement that eliminates it from the cycle. Orbit may implement any end of iteration criteria using boolean functions, which are specific to each application. For example, considering the TCP application, each processed tuple is sent to the Orbit operator that increments the number of iterations performed by the tuple and determines whether it will be re-evaluated by the CFW operators, or if it will leave the execution environment. In the latter case, the tuple is consumed by the Consumer operator, which is responsible for performing the final procedures according to this application. Besides this tuple controlling task, Orbit also includes other two important functionalities: the first, called iteration, is a property assigned to each tuple when it is sent to the buffer for the first time, before its execution by the CFW operator. This property aims at identifying the number of iterations performed by each tuple, and its initial value is set to zero. The second is to make a copy of each tuple before sending it to the Consumer operator, in case Orbit decides the tuple must be re-executed by CFW.

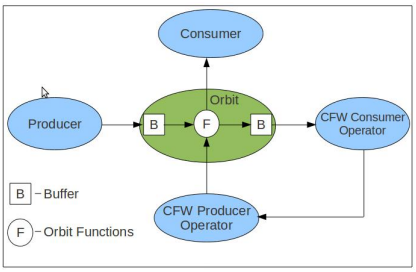


Fig. 1. Orbit Architecture and Relationships among its Operators.

4. PARALLELISM USING ORBIT

Orbit can be executed using three different approaches: (1) Centralized,(2) Parallel with Orbit as a master node and (3) Parallel with Orbit on each remote node. In the (1) all operators in the fragment of the workflow run in the same machine. The cyclic behavior is implemented by introducing the Orbit operator into the workflow model defining the CFW.

In (2) there is a master node and set of remote nodes. The remote nodes are used to enable the parallel execution of a fragment of the workflow. In particular, each remote node may run an instance of the CFW enabling the fragment parallel execution. When Orbit is run in the master node, the whole CFW is placed in the remote nodes. Data consumed by the bottom operator is sent in blocks through the network. Once the tuples in a block have been processed by the CFW they are packed back into a block and sent to the Orbit operator.

In (3) the complete CFW, including Orbit are placed at the remote nodes. The *Producer* operator communicates with the CFW through the Control operators dealing with block transfer. Comparing to the *Orbit in the master* model, in this model, Orbit loses its environment adaptive functions described previously. The execution behavior on each remote node follows that of the Centralized model. A great advantage of this model is in reducing the number of block transfers through the network.

5. EXPERIMENTS

In this section the proposed execution methods will be evaluated. Experimental tests have been carried out within the purpose of evaluating the Orbit operator performance in iterative applications. In this context, we chose the computational technique over particle's trajectory (VPT) to elaborate these tests, in which 1000 virtual particles were processed over 10 iterations. Experiments were performed within a set of seventeen nodes (sixteen nodes dedicated for execution and a single one as a master) and each machine was composed of 2 Quad Core Intel Xeon E5520 @ 2.27GHz processors, 12GB memory, Ubuntu OS 10.04 and 1 TB hard disk.

For each Orbit execution model three implementation types were performed: First Iteration First (FIF), First Tuple First (FTF) and FREE. In all of them, the Orbit buffer responsible for the tuple storage was implemented according to a priority queue data structure. The consumption mode of the stored tuples inside the Orbit buffer determines the execution model to be applied. In the FREE model tuples are taken randomly.

In FIF tuples are synchronized according to its iterations. The purpose here is to ensure all tuples will always perform the same iteration, i.e., they will not execute the next iteration until all of them have finished executing the current one. Therefore, even if a tuple has executed its iteration faster than others, it will have to wait for the others before executing the next iteration. Hence, this execution model is characterized by not eliminating tuples until the last iteration has been executed.

The FTF model prioritizes tuples taking part of higher level iterations. Therefore, if a tuple has executed its iteration faster than others, it does not need to wait for them to execute the next iteration. Thus, it is possible to have tuples processing all their iterations, while others are still executing previous ones. The purpose here is to provide the user with a faster final answer time, ensuring that, if for any reason the tuple does not return the expected result, this occurrence can be identified during its execution and not at the end, as in FIF model.

5.1 Time Execution Evaluation

Figure 2 presents the execution time (seconds) for the model combination (FIF, FTF and FREE) and parallel execution mode (master - OM and remote Orbit). It is possible to notice that execution considering Orbit in each remote node is faster than its execution using Orbit only in the master node. This is due to the fact that in the OM, more time is spent with communication between the master and each remote node. At each iteration the Orbit in the master node splits its data among each remote node. Then, for each iteration, the following procedure is performed: the remote machine executes an iteration over its data part and returns the processed data to the master node. This continuous communication data flow between the master and remote nodes finishes by directly influencing the final application execution time.

When Orbit executes at each remote node, data are read, split and sent a single time by the master to each remote node, according to a parallel processing similar to MapReduce. Thus, each machine executes all the tuples received for iteration. When this process ends, data are sent back from the remote node to the master, which concludes the execution. Hence, time in this latter execution model is much shorter than the first one, when Orbit is in the master node. Nevertheless, there exists an advantage of executing Orbit in the master node. The larger scalability provided by execution in remote nodes reduces their capacity of reacting to fluctuations, and thus decreases time performance.

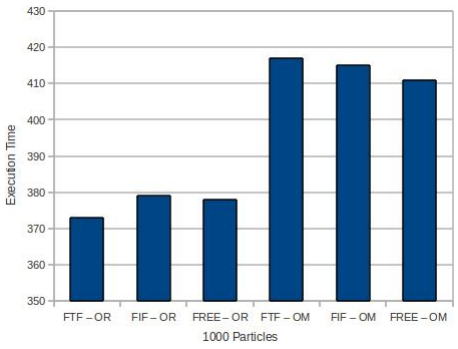


Fig. 2. Execution Time considering each Orbit Model

6. CONCLUSION AND FUTURE WORKS

This paper introduces Orbit an operator for the implementation of iterative behavior in data processing systems. A data processing system equipped with Orbit may produce different execution models, including centralized and distributed modes and different tuple scheduling strategies. We have implemented Orbit in QEF, a data processing system, and experimented with a real application. These first results privilege the allocation of Orbit, and the CFW, in remote nodes. More experiments may be needed to define its proper allocation in a more varied execution scenario.

7. ACKNOWLEDGEMENTS

This work was partially funded by CNPq, Research Productivity - RP - 2012, process:309494/2012-5. The authors would also like to express their gratitude to the ComCiDIS laboratory for allowing us to run our experiments on their cluster environment.

REFERENCES

Hadoop http://hadoop.apache.org, Last access July 2013, 2013.

Abiteboul, S., Hull, R., and Vianu, V. *Foundations of Databases:The Logical Level*. Addison-Wesley, 1994.

Avnur, R. and Hellerstein, J. M. Eddies: continuously adaptive query processing. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*. SIGMOD '00. ACM, New York, NY, USA, pp. 261–272, 2000.

Bu, Y., Howe, B., Balazinska, M., and Ernst, M. D. Haloop: efficient iterative data processing on large clusters. *Proc. VLDB Endow.* 3 (1-2): 285–296, Sept., 2010.

Codd, E. A relational model for large shared data banks. *Communication of the ACM* 13 (6, 1970.

Graefe, G. Encapsulation of parallelism in the volcano query processing system. *SIGMOD Rec.* 19 (2): 102–111, 1990.

Liae, Y., Perlmanb, E., Wana, M., Yanga, Y., C. Meneveaua, R. B., S. Chena, A. S., and Eyinkd, G. A public turbulence database cluster and applications to study lagrangian evolution of velocity increments in turbulence. *Journal of Turbulence* 9 (2, Oct., 2008.

Ogasawara, E. S., de Oliveira, D., Valduriez, P., Dias, J., Porto, F., and Mattoso, M. An algebraic approach for data-centric scientific workflows. *PVLDB* 4 (12): 1328–1339, 2011.

Ozsu, T. and P.Valduriez. *Principles of Distributed Database Systems*. Springer, 2011.

Porto, F., Tajmouati, O., Silva, V. F. V. D., Schulze, B., and Ayres, F. V. M. Qef - supporting complex query applications. In *CCGRID '07: Proceedings of the Seventh IEEE International Symposium on Cluster Computing and the Grid*. IEEE Computer Society, Washington, DC, USA, pp. 846–851, 2007.

Sippu, S. and Soisalon-Soininen, E. A generalized transitive closure for relational queries. In *PODS*. pp. 325–332, 1988.

Stonebraker, M. and Rowe, L. A. The design of postgres. *SIGMOD Rec.* 15 (2): 340–355, June, 1986.

Distribuição de Bases de Dados de Proveniência na Nuvem *

Edimar Santos[1], Vanessa Assis[1], Flavio Costa[1], Daniel de Oliveira[2] e Marta Mattoso[1]

[1]Programa de Engenharia de Sistemas e Computação – COPPE / UFRJ
[2]Instituto de Computação - Universidade Federal Fluminense - UFF
{ebabilon,vmarques,flscosta,marta}@cos.ufrj.br, danielcmo@ic.uff.br

Resumo. Dados de proveniência no contexto de *workflows* científicos são peças fundamentais, pois, por meio deles, os experimentos são passíveis de reprodução e validação. O histórico da execução dos *workflows* é fundamental também para a gerência da execução de novos *workflows* uma vez que possibilitam às máquinas de *workflow* realizar predições sobre desempenho ou custo financeiro de nuvens de computadores. *Workflows,* com dados em larga escala, executados em nuvens, são com frequência alocados em máquinas virtuais distribuídas fisicamente. As soluções existentes coletam os dados de proveniência de forma distribuída e os armazenam de modo centralizado em único repositório, após o término da execução do workflow. Além da capacidade de reprodução, dados de proveniência permitem um acompanhamento refinado por parte do cientista, quando disponibilizados à medida que são gerados, durante a execução do workflow. Porém, quando os dados de proveniência só estão disponíveis para consulta após a execução do workflow, seu uso fica limitado. Para permitir consultas durante a execução do workflow, o acesso ao banco de dados de proveniência deve estar em sintonia com a máquina de execução distribuída de workflows. Este artigo discute aspectos de projeto de distribuição de dados de proveniência, levando em consideração o esquema de representação de proveniência do W3C, aspectos de processamento distribuído de consultas em nuvens de computadores e considerando a execução distribuída do workflow. A estratégia aqui adotada trouxe melhoria de desempenho para as consultas que submetemos em tempo de execução dos *workflows* aumentando assim a eficiência dos *workflows* científicos testados.

Palavras-chave: *Workflow*, Proveniência de dados, projeto de distribuição

1. INTRODUÇÃO

Os experimentos científicos baseados em simulações computacionais são comumente realizados por múltiplas combinações de programas, onde cada um deles possui um conjunto de parâmetros e dados de entrada. Esses experimentos podem ser modelados como *workflows* científicos (Deelman *et al.* 2009). Um *workflow* científico é uma abstração que representa o encadeamento de programas formando um fluxo coerente que produz um resultado final. Tais *workflows* são modelados e executados para validar hipóteses levantadas pelos cientistas, e por muitas vezes são complexos demandando a utilização de recursos computacionais de capacidade de alto desempenho. Como os experimentos científicos necessitam ter sua reprodução garantida, a utilização da proveniência se torna uma questão fundamental (Freire *et al.* 2008). Os dados relativos à execução do *workflow* devem ser coletados e passíveis de consulta. Essa gerência dos dados de proveniência se torna mais complexa quando os *workflows* precisam ser executados em paralelo em ambientes de alto desempenho, como as nuvens de computadores (Vaquero *et al.* 2009). É fundamental que a proveniência registre qual atividade gerou qual dado e onde esse dado está armazenado.

Nas máquinas de *workflows* atuais, os dados de proveniência são coletados durante a execução do *workflow*, mas não são disponibilizados para consulta enquanto o *workflow* não termina de executar. A disponibilização dos dados de proveniência para consulta, durante a execução do *workflow*, pode prover uma série de vantagens para o cientista. Com a proveniência consultada em tempo real, é possível realizar o escalonamento adaptativo de *workflows* (Oliveira *et al.* 2011), a supervisão da execução do *workflow* (*i.e. workflow steering*) (Dias *et al.* 2011) e a gerência de falhas das atividades (Costa *et al.* 2012).

* Esse trabalho foi financiado parcialmente pela Capes, Faperj e CNPq

Entretanto, a literatura sobre sistemas e máquinas de *workflows* não apresenta soluções para consultas a dados de proveniência distribuídos durante a execução do workflow. Os repositórios de proveniência são disponibilizados somente após as execuções do workflow, de forma centralizada, ou seja, armazenam em uma mesma base todos os dados das execuções de todos os *workflows* independentemente de onde e por quem foram executados. Este cenário torna-se ainda mais complexo quando há equipes geograficamente dispersas trabalhando nos mesmos *workflows* cuja análise dos resultados é colaborativa e em tempo real.

Alguns poucos trabalhos tratam da distribuição de dados de proveniência. Allen *et al.* 2011, Freire *et al.* 2008, Zhou *et al.* 2011. Nenhum destes trabalhos tem o objetivo de disponibilizar dados de proveniência durante a execução distribuída do workflow. Entretanto, alguns desses trabalhos possuem um grau de semelhança com a proposta apresentada neste artigo. O artigo de Zhou *et al.* (2011) apresenta uma abordagem de proveniência para sistemas distribuídos. Entretanto, Zhou *et al.* fazem justamente o contrário do que é proposto aqui. Os autores assumem que os dados estão distribuídos em diversos sítios e realizam consolidação para que se possa analisar o desempenho de execuções dinâmicas no sistema como um todo. No trabalho de Allen *et al.* (2011) é apresentada uma arquitetura para o compartilhamento de proveniência, mas assume que os dados já estão distribuídos. A arquitetura proposta poderia ser utilizada como complemento à estratégia de fragmentação que é analisada no presente artigo.

O problema de distribuição de dados já foi amplamente discutido e diversas soluções são apresentadas por Özsu e Valduriez (2011). Entretanto, pouco se sabe sobre o comportamento de bancos de dados distribuídos em nuvens de computadores, foco de deste trabalho, por se tratar do ambiente utilizado por muitos grupos de pesquisa para execução de seus experimentos. A complexidade se torna maior quando o esquema de fragmentação precisa estar de acordo com um padrão, no caso o W3C, e acompanhar o dinamismo da máquina distribuída de execução de workflows em nuvens.

O objetivo deste artigo é avaliar o comportamento de consultas a dados de proveniência que são gerados em tempo real, de modo distribuído, por máquinas de execução de *workflows* em nuvens, de modo a propor um projeto de distribuição que atenda ao cenário apresentado. Ao trabalharmos com distribuição de dados na nuvem, diferente de outros ambientes de computação paralela, a transferência de dados entre os nós é um ponto que deve ser sempre investigado. Este artigo apresenta uma proposta de execução distribuída de consultas sobre uma fragmentação de dados de proveniência e discute o comportamento de consultas submetidas durante uma execução real de workflow na nuvem. Para tal, partiu-se do esquema de dados de proveniência utilizado pela máquina de execução distribuída de *workflows* SciCumulus (Oliveira *et al.* 2011). O esquema do SciCumulus segue o PROV-Wf (Costa *et al.* 2013) que estende o modelo PROV do W3C para a representação de proveniência em *workflows*.

Além da introdução, este artigo possui 3 seções. A Seção 2 apresenta o projeto de distribuição dos dados de proveniência. A Seção 3 avalia o processamento distribuído das consultas e a Seção 4 conclui.

2. PROJETO DE DISTRIBUIÇÃO E CONTROLE DE ACESSO AOS DADOS DE PROVENIÊNCIA

Um projeto de distribuição realiza duas decisões, a fragmentação e a alocação (Özsu e Valduriez 2011). Além disso, podemos destacar também o controle de acesso aos fragmentos. Para o projeto de fragmentação, são analisadas as consultas mais frequentes e a alocação determina onde os fragmentos e eventuais réplicas serão alocados. Durante a execução do workflow, o cientista deverá analisar apenas as tuplas associadas ao *workflow* que ele está acompanhando, assim, a fragmentação horizontal, por *workflow*, se aplica de forma adequada para esse cenário. Por outro lado, a proveniência é usada pela máquina de execução do *workflow* para funções como o escalonamento de uma determinada atividade. Nesse caso, apenas os tempos de execução da atividade e os indicativos de falha são suficientes. Essas consultas à base de proveniência acessam poucos atributos, mas necessitam de todas as tuplas, já que irão calcular a média de tempo de execução, desvio-padrão, etc. Neste cenário, a fragmentação vertical parece ser a mais adequada, mesmo porque, somente os tempos estariam acessíveis, sem apresentar os resultados da pesquisa que podem estar armazenados no mesmo esquema de proveniência.

Seja o seguinte cenário (com nomes propositalmente fictícios, mas com um *workflow* real): Pedro é um cientista, localizado em São Paulo, que executa em seus experimentos o *workflow* SciPhy (Ocaña *et al.* 2011), que realiza análises filogenéticas. Em sua base de proveniência, já existe uma grande quantidade de informações referentes a execuções do SciPhy. Um colega de Pedro que reside em Dublin, chamado Robert, está se familiarizando com o *workflow* SciPhy e pretende utilizá-lo para experimentos de análise filogenética em um futuro próximo. Assim, Pedro pretende executar experimentos em conjunto com Robert de forma que possam realizar uma análise colaborativa. Entretanto, existem outros cientistas executando o *workflow SciPhy* que Pedro não deseja compartilhar informação, como Dr. Kobayashi do Japão. Robert deve poder usar os dados de Pedro para que sua máquina de *workflow* possa realizar um escalonamento com base no histórico ou possa descobrir pontos de falha de maneira eficiente. Esse é um cenário comum em que Pedro pode compartilhar seus dados com Robert sem que terceiros possam acessá-los, e assim Robert possa obter esses dados de proveniência da maneira mais eficiente possível.

No cenário apresentado, pode-se fragmentar horizontalmente a base de dados de proveniência de acordo com o nome do *workflow* em execução, pois essa é uma distribuição natural, na qual os fragmentos representam os conjuntos de *workflows* executados em uma determinada localidade. Nesse caso, Robert faria sua consulta apenas no fragmento que contém as informações das execuções do *workflow* SciPhy executado em Sao Paulo. Posteriormente, usaria seu próprio sítio para armazenar seus dados, mas continuaria utilizando a massa de dados do fragmento de São Paulo para compor suas análises. Se um mesmo *workflow* pode ser executado tanto na Irlanda quanto em Tóquio, ou mesmo no Brasil como no exemplo apresentado, a próxima decisão a ser tomada seria de alocação com replicação dos dados relativos ao *workflow*.

Na fragmentação horizontal, os fragmentos possuem todos os atributos da relação original, mas há uma redução na quantidade de tuplas a serem armazenadas e consultadas, o que favorece a diminuição do tempo de resposta. Considerando que os atributos mais frequentes para as consultas, são o nome do *workflow* e a sua localidade, e tendo em vista que o cientista tende a fazer um número maior de consultas em experimentos executados por seu grupo de pesquisas, a escolha por esses atributos para a fragmentação horizontal também acaba se tornando uma decisão natural. Assim, os atributos utilizados no projeto de fragmentação foram *tag* (nome do *workflow* que se encontra na classe *Workflow* do Prov-Wf) e localidade. Tal projeto só foi possível devido à estratégia de se adotar um modelo único derivado do PROV. Aplicou-se então o algoritmo de fragmentação horizontal baseado em mintermos, conforme em (Özsu e Valduriez 2011). Os predicados simples utilizados são:

P1. *tag*="SciPhy"; P2. *tag*="SciPhylomics"; P3. *tag*="SciEvol"; P4. *tag*="ExpX"; P5. *tag*="exp"; P6. *tag*="Scimult"; P7. localidade="Tóquio"; P8. localidade="Dublin"; P9. localidade="São Paulo".

Foram criados os fragmentos correspondentes a cada um dos mintermos finais para os *workflows* com as combinações das *tags* "SciPhy", "SciPhylomics", "SciEvol", "ExpX", "Exp" e "Scimult" para cada uma das localidades "Dublin", "São Paulo" e "Tóquio". Além disso, para garantir a completude, existe também um fragmento chamado *wk_outros*, para cada uma das localidades, onde devem ser armazenados *workflows* com *tags* diferentes das apresentadas anteriormente que foram elencadas a partir das consultas mais frequentes. Um total de 21 fragmentos foram gerados: sete fragmentos chamados *wk_sciphy, wk_sciphylomics, wk_scievol, wk_expx, wk_exp, wk_simultaneous* e *wk_outros* para cada uma das três localidades Dublin, São Paulo e Tóquio.

Além disso, criamos um repositório central com todos os fragmentos em questão, de modo que para todo fragmento criado, existe uma réplica em Singapura que possui os dados relativos a todo o conjunto de *workflows* executados. Os alvos de alocação foram então os três sítios: São Paulo, Dublin e Tóquio (utilizamos o serviço da Amazon AWS que permite criar máquinas virtuais especificando a localização física de servidores em várias partes do mundo). Para aumentar a disponibilidade dos dados, todos os fragmentos foram enviados para Singapura onde formaram um repositório centralizado sendo replicados após o projeto de distribuição, aumentando a disponibilidade em caso de falhas.

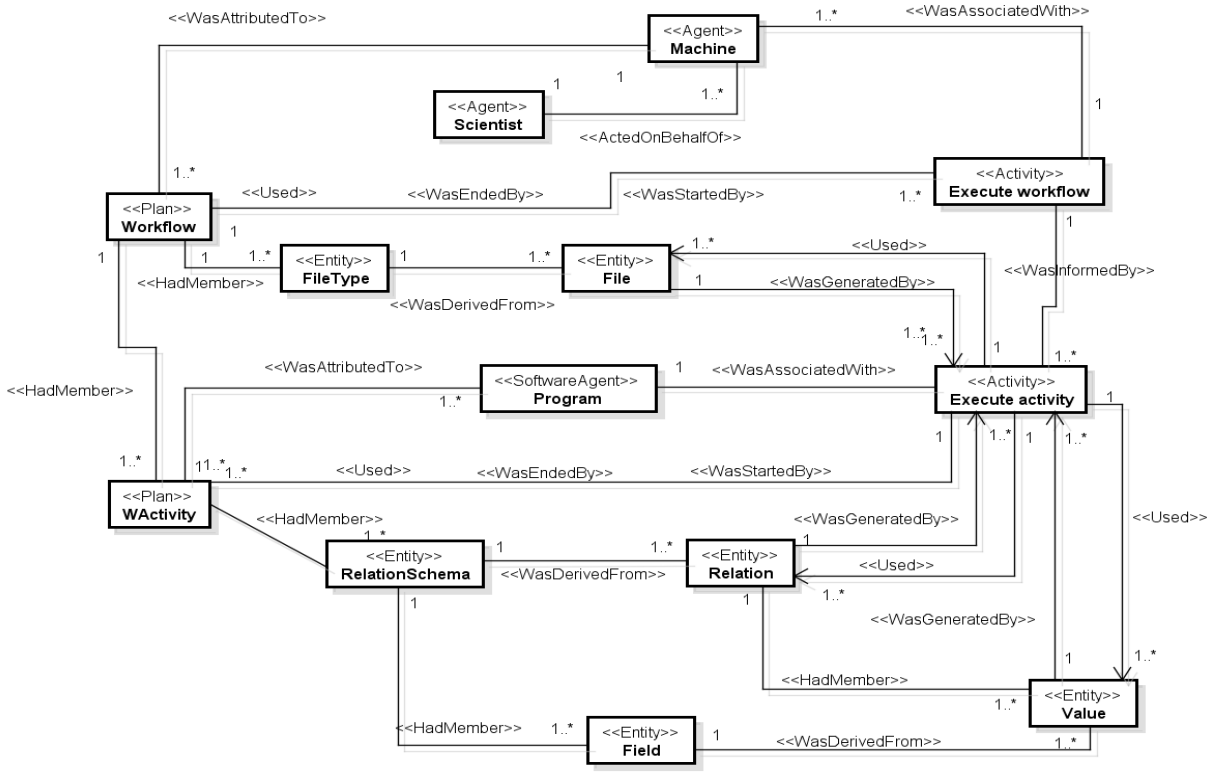


Figura 1 O esquema de proveniência PROV-Wf como proposto por Costa *et al.* (2013)

Como estamos trabalhando com fragmentos de dados (mesmo no caso de Singapura onde esses fragmentos são agrupados) foi possível criar um controle de acesso em nível de fragmento, ou seja, para cada equipe de pesquisa, eu tenho um conjunto de fragmentos disponíveis. Esses fragmentos podem estar localizados no sítio da equipe, no sítio de uma outra equipe colaboradora ou no sítio central em Singapura.

3. AVALIAÇÃO DE CONSULTAS SOBRE DADOS DE PROVENIÊNCIA FRAGMENTADOS

Para realizar uma análise de desempenho de consultas sobre os fragmentos gerados, foram consideradas consultas que são frequentemente utilizadas pelos cientistas dos workflows de análise filogenética. Em nossos experimentos, os dados da base de proveniência vão sendo disponibilizados para consulta, à medida que o workflow vai sendo executados. Utilizamos então algumas consultas que são submetidas durante a execução do workflow (sendo que nestes casos, para que os resultados fossem sempre os mesmos e fosse possível fazer a média de 10 execuções, as consultas foram submetidas de modo desacoplado da execução, sem prejuízo à análise dos resultados). Para a avaliação, foram selecionadas nove consultas consideradas frequentes, descritas a seguir: (1) Listar os rótulos e as descrições de todas as atividades de um determinado *workflow*; (2) Listar todas as execuções de cada uma das atividades de um determinado *workflow;* (3) Listar todas as atividades com estado diferente de "*finished*" de um determinado *workflow* em São Paulo, ou seja, todas as atividades que ainda não foram executadas ou não finalizaram sua execução; (4) Listar a duração de todas as atividades com estado igual a "*finished*" de um determinado *workflow* de Dublin; (5) Listar o horário de início de cada uma das atividades com status igual a "*running*" de um determinado *workflow* em Tóquio; (6) Buscar a duração de cada uma das execuções em workflows que contêm uma determinada atividade; (7) Buscar a mensagem de erro, obtida com o atributo *terr*, de todas as execuções de atividades com status de saída diferente de zero (*i.e.* sem erro) de um determinado *workflow*; (8) Listar todos os *workflows* que contêm uma determinada atividade especificada e (9) Listar, por ordem crescente de execuções de *workflows*, as datas e horas de início e término das ativações, *tags*

dos *workflows* e suas descrições, bem como o nome de todas as atividades associadas a todas as execuções dos *workflows* que foram executados sem erro. Cada uma das nove consultas apresentadas foi executada no ambiente distribuído e no ambiente centralizado, ambos como máquinas virtuais na nuvem da Amazon. O ambiente distribuído não usou a base centralizada para as consultas, assim, quando a consulta envolvia fragmentos não locais, consultas remotas eram submetidas nas máquinas virtuais correspondentes e os resultados consolidados localmente. Os tempos de resposta, obtidos com a execução das consultas (Tabela 1), foram comparados a fim de analisar o desempenho atingido utilizando o projeto de distribuição apresentado e como essa estratégia se comporta em relação ao ambiente centralizado em termos de tempo de resposta das consultas no ambiente de nuvens computacionais.

Tabela 1. Resultados das consultas sobre a base fragmentada e centralizada

Consulta	Tuplas acessadas	Tempo em ms (centralizado)	Tempo em ms (distribuído)
C1	105	826	405
C2	5.828	8.408	1.852
C3	15	391	208
C4	23	374	209
C5	11	406	199
C6	1.614	1.278	1.352
C7	125	1.743	773
C8	42	231	722
C9	15.259	16.973	26.052

Para a realização dessa análise de desempenho foram efetuadas cerca de dez execuções de cada uma das consultas descritas anteriormente, obtendo assim uma média dos tempos de execução. Por terem apresentado um desvio-padrão desprezível, as variações não são apresentadas. A escolha das consultas levou em consideração que a maioria dos cientistas realiza pesquisas buscando um *workflow*, em particular, o que ele está executando naquele momento. Podemos observar, conforme o esperado, que ocorreu um ganho significativo nos tempos de execução para as consultas realizadas sobre fragmentos específicos locais, como foi o caso das consultas C3, C4 e C5. Esse resultado era esperado, pois essas consultas utilizam como restrição a pesquisa por um determinado *workflow* e sua localidade, assim acessam um só fragmento local. Entretanto, mesmo em consultas com acesso remoto, mas direcionadas a fragmentos específicos, houve ganho significativo, como foi o caso de C1, C2 e C7. Cabe ressaltar que C2 acessa um número significativo de tuplas e obteve uma redução também importante, C2 sobre a base centralizada foi cerca de 4 vezes mais lenta que a distribuída. Esse resultado mostra que o paralelismo em consultas frequentes desse tipo podem compensar perdas em outros casos como C8 e C9. Cabe observar que se as consultas 8 e 9 forem as mais frequentes, uma fragmentação não seria recomendada e talvez uma replicação total da base nos sítios mais ativos poderia ser considerada. Podemos citar também o caso particular de C6, que não envolve a pesquisa por um *workflow* específico, e que busca a duração das execuções de uma determinada atividade. O tempo de resposta para essa pesquisa apresentou resultados equivalentes na base centralizada e distribuída, isso ocorre devido ao acesso específico à tabela atividade em cada sítio distribuído em paralelo. Ou seja, o processamento em paralelo sobre os fragmentos distribuídos acabou por tornar a execução distribuída, competitiva com a centralizada. As consultas que requerem percorrer todos os sítios acabam sendo penalizadas pelo tempo de comunicação e transferência de dados, levando em consideração a distância física entre os sítios. É importante lembrar que esse tipo de influência da distância reflete a realidade, tendo em vista que os centros de pesquisa estão espalhados pelo mundo, porém, considerando um conjunto de consultas analíticas, espera-se que os ganhos superem as perdas, ou seja, bastam duas execuções da C2 para compensar uma execução da C9. Por outro lado, se considerarmos um projeto de replicação que contemple um sítio contendo todos os fragmentos, como o de Singapura, o processamento distribuído de consultas poderia direcionar essa classe de consultas diretamente à base centralizada. Como as alterações da base de proveniência é baseada em inserções sem atualização de tuplas, o custo de manutenção da consistência não é muito significativo.

4. CONCLUSÃO

Os *workflows* científicos apresentam-se como uma abordagem fundamental para a modelagem de experimentos científicos baseados em simulações. A gerência da proveniência desses *workflows* é uma questão fundamental, pois além da análise do experimento, auxilia a reprodução, peça fundamental do processo científico. Além da reprodução, os dados de proveniência podem ajudar a máquina de execução do workflow no escalonamento, detecção de falhas, dimensionamento do ambiente, etc., pois contêm informações acerca das atividades executadas, as diferentes ativações utilizadas, os tempos de execução, os resultados gerados, possíveis erros ocorridos dentre outros. Obter essas informações traz grandes benefícios para os cientistas, pois a partir delas é possível abortar uma execução que não caminha na direção almejada e, em caso de sucesso, reproduzir a execução de um *workflow*, além de realizar um melhor gerenciamento dos recursos necessários e análises de desempenho, bem como detectar erros com maior facilidade e em tempo real. Como os dados de proveniência são utilizados frequentemente para tomar decisões em tempo real, é importante ter consultas com um tempo de resposta pequeno, para que essas não afetem o desempenho do *workflow*. Entretanto, as máquinas de *workflows* de hoje utilizam repositórios de proveniência centralizados, o que impõe pontos de falha e problemas de segurança e desempenho. Tendo isso em mente, foi proposto neste artigo uma estratégia de fragmentação e alocação dos dados de proveniência de maneira a minimizar o tempo necessário para que as consultas sejam realizadas. Essa estratégia levou em consideração dois fatores principais: o fato de que as consultas realizadas por um cientista em geral giram em torno do *workflow* em que ele está trabalhando; e que, no geral, os dados gerados por aquele cientista são armazenados nos sítios mais próximos a ele. Assim, foi realizada uma fragmentação horizontal sobre os atributos *tag* e localidade, que representam respectivamente o rótulo de identificação do *workflow* e o local onde o mesmo foi executado. Uma série de consultas, frequentemente utilizadas por cientistas, foi selecionada e avaliada tanto no ambiente distribuído, como no ambiente centralizado. Os resultados obtidos indicam uma melhora significativa no ambiente distribuído em consultas realizadas em um único fragmento, mostrando ganhos significativos sobre o tempo de execução da consulta na base centralizada mesmo considerando sítios distantes fisicamente. Como trabalho futuro será realizado uma análise da fragmentação vertical e híbrida.

REFERENCES

Allen, M., Chapman, A., Blaustein, B., Seligman, L., (2011), "Getting It Together: Enabling Multi-organization Provenance Exchange". In: *TaPP 2011*, Athens, Greece.

Costa, F., Oliveira, D., Ocaña, K., Ogasawara, E., Mattoso, M., (2012), "Enabling Re-Executions of Parallel Scientific Workflows Using Runtime Provenance Data. In: 4th International Provenance and Annotation Workshop"

Costa, F., Silva, V., Oliveira, D., Ocaña, K., Dias, J., Ogasawara, E., Mattoso, M., (2013), "Capturing and Querying Workflow Runtime Provenance with PROV: a Practical Approach". In: *BigProv'13*, Genova, Italy.

Deelman, E., Gannon, D., Shields, M., Taylor, I., (2009), "Workflows and e-Science: An overview of workflow system features and capabilities", *Future Generation Computer Systems*, v. 25, n. 5, p. 528–540.

Dias, J., Ogasawara, E., Oliveira, D., Porto, F., Coutinho, A., Mattoso, M., (2011), "Supporting Dynamic Parameter Sweep in Adaptive and User-Steered Workflow". In: *6th WORKS*, p. 31–36, Seattle, WA, USA.

Freire, J., Koop, D., Santos, E., Silva, C. T., (2008), "Provenance for Computational Tasks: A Survey", *Computing in Science and Engineering, v.10*, n. 3, p. 11–21.

Missier, P., Belhajjame, K., Cheney, J., (2013), "The W3C PROV family of specifications for modelling provenance metadata". In: *Proceedings of the 16th EDBT*, p. 773–776, New York, NY, USA.

Ocaña, K. A. C. S., Oliveira, D., Ogasawara, E., Dávila, A. M. R., Lima, A. A. B., Mattoso, M., (2011), "SciPhy: A Cloud-Based Workflow for Phylogenetic Analysis of Drug Targets in Protozoan Genomes", BSB 2011: Springer, p. 66–70.

Oliveira, D., Ogasawara, E., Ocaña, K., Baião, F., Mattoso, M., (2011), "An Adaptive Parallel Execution Strategy for Cloud-based Scientific Workflows", *Concurrency and Computation: Practice and Experience*, v. (online)

Özsu, M. T., Valduriez, P., (2011), *Principles of Distributed Database Systems*. 3 ed. New York, Springer.

Vaquero, L. M., Rodero-Merino, L., Caceres, J., Lindner, M., (2009), "A break in the clouds: towards a cloud definition", *SIGCOMM Comput. Commun. Rev.*, v. 39, n. 1, p. 50–55.

Zhou, W., Ding, L., Haerberlen, A., Ives, Z. G., Loo, B. T., (2011), "TAP: Time-aware Provenance for Distributed Systems". In: *Proc. of TaPP'11*, Greece.