

Desenvolvimento de um Sistema para a Centralização e Consulta de Informações da Estação SONDA/OES

Miriam Pizzatto Colpo, Adriano Petry, Fabrício Viero, Márcio Ceconi

Centro Regional Sul de Pesquisas Espaciais – CRS/CCR/INPE – MCT em colaboração com o Laboratório de Ciências Espaciais de Santa Maria – LACESM/CT – UFSM, Santa Maria, Rio Grande do Sul, Brasil.

*mpcolpo@inf.ufsm.br, adriano.petry@crs.inpe.br,
fabricioviero@hotmail.com, marcioceconi@gmail.com*

Resumo. *Este artigo descreve o desenvolvimento de um banco de dados, de um software de carregamento de dados e de um sistema de consulta a dados, com a finalidade de centralizar as informações solares e eólicas obtidas por sensores pertencentes a uma subestação do projeto SONDA, instalada no Observatório Espacial do Sul em São Martinho da Serra no Rio Grande do Sul.*

1. Introdução

Com o intuito de melhorar a base de dados dos recursos de energia solar e eólica no Brasil, o Instituto Nacional de Pesquisas Espaciais (INPE), criou o Sistema de Organização Nacional de Dados Ambientais (SONDA), um projeto que tem como objetivo a construção de infra-estrutura física e desenvolvimento de recursos humanos que permitam o levantamento dessas informações ao longo do território nacional. Atualmente, a rede de dados SONDA possui treze estações próprias, incluindo a instalada no Observatório Espacial do Sul (OES), em São Martinho da Serra no Rio Grande do Sul, capacitada com equipamentos para a obtenção de dados eólicos e solares, que são armazenados em arquivos texto com estrutura pré-definida.

Para facilitar o acesso e a visualização dos dados presentes nesses arquivos textuais, foi construído um banco de dados, um software para carregar esses dados para o banco criado e um sistema de consulta. Esse artigo apresenta inicialmente o Projeto SONDA e sua subestação SONDA/OES, juntamente com os dados oriundos dessa estação, que objetivaram todo o trabalho desenvolvido. Após, é mostrada a construção do banco de dados para a centralização dessas informações, juntamente com o sistema de carregamento de dados. E, por fim, é apresentada a construção do sistema de consulta aos dados, responsável por abstrair o processo de acesso e consulta ao banco de dados para o usuário.

2. O Projeto SONDA e a Estação SONDA/OES

O SONDA [SONDA 2011] é um projeto coordenado e executado pelo INPE, através do seu Centro de Ciências do Sistema Terrestre (CCST), que tem como objetivo o levantamento e melhoria da base de dados dos recursos de energia solar e eólica no Brasil. A rede de dados SONDA possui, atualmente, treze estações próprias e duas

estações colaboradoras distribuídas pelo território brasileiro em sítios selecionados por suas características climáticas e ambientais.

Dentre as estações próprias do projeto SONDA, encontra-se a instalada no OES, equipada com sensores para medida de dados ambientais, como radiação, iluminância, pressão atmosférica, temperatura e umidade do ar, além de anemômetros para determinação da velocidade e direção do vento. Os sensores da Estação SONDA/OES, suas metodologias de medida e de calibração, são padronizados internacionalmente, o que possibilitou a inclusão da estação em redes internacionais de dados, como a *Baseline Surface Radiation Network* (BRSN), um projeto da Organização Mundial de Meteorologia (WMO), e a *Aerosol RObotic NETwork* (AERONET), rede coordenada pelo *Goddard Space Flight Center* (GSFC) da *National Aeronautics and Space Administration* (NASA).

2.1. Informações Produzidas pelos Sensores da Estação SONDA/OES

Os dados gerados pelos equipamentos da Estação SONDA/OES são coletados e armazenados em arquivos texto no formato CSV (*Comma Separated Values*), separados por vírgulas, por um sistema automático que retém as informações em uma memória de espaço limitado, até que alguém os transfira para outro local de armazenamento. Esses arquivos seguem uma estrutura pré-definida, sendo que cada linha do arquivo contém um código identificador para o tipo de dados (solarimétrico ou anemométrico), além das informações referentes à data e ao horário em que as medições foram realizadas e dos valores obtidos pelos sensores para os dados medidos, como mostra a figura 1.

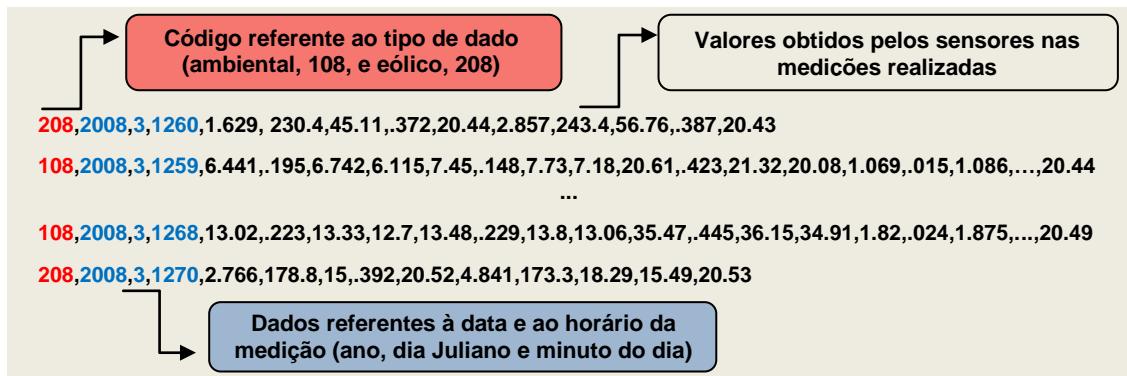


Figura 1. Disponibilidade dos dados nos arquivos texto

3. Desenvolvimento do Banco de Dados para os Equipamentos da Estação SONDA/OES

Com a finalidade de possibilitar a absorção dos dados produzidos pelos sensores da Estação SONDA/OES, foi desenvolvido um sistema de banco de dados relacional, chamado “Banco_Measures”, cuja implantação fez uso do Sistema de Gerenciamento de Banco de Dados (SGBD) PostgreSQL [PostgreSQL 2011], escolhido por ser um banco de dados *open source* e altamente escalável, tanto na quantidade de dados a ser gerenciada, quanto no número de usuários concorrentes a serem acomodados. A escalabilidade, principalmente para a quantidade de dados gerenciáveis, é uma necessidade nessa aplicação, visto que as informações são coletadas com freqüência, fazendo com que a base de dados cresça de forma considerável diariamente.

3.1. Modelagem do Banco_Measures

O banco de dados foi organizado em quatro tabelas relacionadas, sendo que cada arquivo com dados de medições deve equivaler a um registro na tabela “*measures_file*”, que identifica o arquivo de origem dos dados. Esses arquivos podem conter informações a cada minuto e a cada dez minutos. Cada linha com dados de medições desses arquivos representará um registro da tabela “*measures_data*” e diversos registros associados (uma para cada medida) na tabela “*measures_values*”, que armazena os valores das medições. Além disso, cada registro da tabela “*measures_values*” terá um registro associado na tabela “*measures_name*”, relacionando, assim, o valor da medição à sua respectiva medida.

3.2. Carga dos Dados

Para realizar o carregamento dos dados dos arquivos textuais para o banco de dados criado, foi desenvolvido um software escrito em linguagem Java [Java 2011], escolhida por ser uma linguagem multiplataforma, *open source*. Também foi utilizado o *framework* Hibernate [Elliott e Obrie 2009], usado no programa para realizar o mapeamento objeto relacional das tabelas, facilitando o acesso ao banco de dados.

O carregamento de dados pode ser feito a partir do endereço de um arquivo, ou de uma pasta contendo arquivos, que pode também conter outras sub-pastas, gerando uma busca recursiva por arquivos válidos. O software é responsável por ler o arquivo linha a linha, identificando o tipo das informações a serem inseridas (anemométricas ou anemométricas) e carregando-as, juntamente com seus respectivos dados, para o local determinado no banco.

4. Sistema de Consulta aos Dados

Com a finalidade de disponibilizar esses dados de forma prática e proporcionar ao usuário uma maior abstração do processo de pesquisa, tornando desnecessário o uso de conhecimentos prévios, como o de linguagem SQL (*Structured Query Language*) [Damas 2007], foi desenvolvido um sistema de consulta utilizando uma interface Web, que pode ser acessado através do site <http://200.132.24.196/site/>.

The figure shows a screenshot of a web-based data query system. On the left, a sidebar displays the following text:

Tela inicial do sistema, onde o usuário estabelece o período (dias e horários iniciais e finais) e o tipo de dados a ser pesquisado: anemométricos (10 min) ou solarimétricos (1 min). Na próxima tela, o sistema listará as medidas referentes ao tipo de dados escolhido. Para a opção "1 min", serão listadas as medidas mostradas em "a" e para a opção "10 min" as medidas mostradas em "b", logo ao lado.

The main interface has two main sections labeled 'a)' and 'b)' under the heading 'Sensors SONDA - BD / Measures'. Section 'a)' lists measures for 1-minute resolution: Case_Temp, Cos_Angle, Diffuse_AVG, Diffuse_MAX, Diffuse_MIN, Diffuse_STD, Diffuse_Temp, Direct_AVG, Direct_MAX, Direct_MIN, Direct_STD, Direct_Temp, Domo_Temp, Global_AVG, Global_MAX, Global_MIN, Global_STD, Global_Temp, Humidity, LUX_AVG. Section 'b)' lists measures for 10-minute resolution: Temp_25, Temp_50, Wd25_AVG, Wd25_STD, Wd50_AVG, Wd50_STD, Ws25_AVG, Ws25_STD, Ws50_AVG, Ws50_STD. A tooltip for 'Temp_50' in section 'b)' indicates it corresponds to 'Air Temperature at 50 meters'. At the bottom, there are buttons for 'Next', 'Back', and 'Display'.

Figura 2. Interface do sistema de consulta.

Para realizar uma consulta no sistema, o usuário deve estabelecer seus requisitos, informando período, tipo de dados, medidas a serem pesquisadas e a forma que as informações retornadas pela consulta deverão ser exibidas, como mostrado na figura 2.

Os dados consultados podem ser armazenados em um arquivo-texto com formato CSV, que poderá ser salvo no computador do usuário, permitindo o posterior uso desses dados em outros softwares científicos. Além disso, há a possibilidade das informações serem apresentadas textualmente, em forma de tabela, ou graficamente na tela. Para a implementação dessa última opção, foi usado o *framework* JFreeChart [JFreeChart 2011], uma biblioteca de código livre para a linguagem de programação Java que facilita o processo de criação de gráficos. A figura 3 mostra o resultado de uma consulta, com a opção de visualização gráfica selecionada.

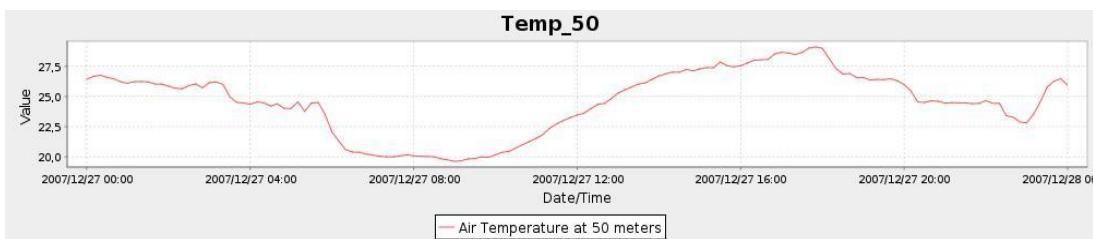


Figura 3. Gráfico da temperatura do ar a 50 metros de altura para o dia 27 de dezembro de 2007

5. Conclusão

Este trabalho apresentou inicialmente o Sistema de Organização Nacional de Dados Ambientais e sua subestação, instalada no Observatório Espacial do Sul. Em seguida, descreveu o banco de dados relacional criado para armazenar as informações eólicas e solares oriundas dessa subestação, além do desenvolvimento de um software para efetuar o carregamento desses dados para o banco criado e de um sistema de consulta. O trabalho realizado possibilitou não só a centralização dos dados, até então inexistente, em um banco de dados único, mas também uma maior abstração do processo de pesquisa, tornando o acesso às informações o mais intuitivo possível para o usuário. A visualização gráfica dos dados, oferecida pelo sistema de consulta desenvolvido, permitirá o monitoramento dos dados coletados, ajudando a identificar possíveis anomalias no comportamento das informações medidas, o que acelerará o estudo do fenômeno ou o processo de reparo dos sensores, caso o problema seja técnico.

Referências

- Damas, L. (2007) “SQL: Structured Query Language”, LTC, 6^a edição.
Elliott, J. e Obrie, T. (2009) “Dominando Hibernate”, Alta Books, 1^a edição.
Java (2011) <<http://www.oracle.com/technetwork/java/>>, fevereiro.
JFreeChart (2011) <<http://www.jfree.org/jfreechart/>>, fevereiro.
PostgreSQL (2011) <<http://www.postgresql.org/>>, fevereiro.
SONDA – Sistema de Organização Nacional de Dados Ambientais (2011)
<<http://sonda.ccst.inpe.br/>>, fevereiro.

Um Modelo de Grupos para MANETs

Henrique S. dos Santos¹, Raqueline R. M. Penteado¹,

Luiz Carlos P. Albini², Carmem Hara²

¹Departamento de Informática
Universidade Estadual de Maringá (UEM) – Maringá, PR – Brasil

²Departamento de Informática
Universidade Federal do Paraná (UFPR) – Curitiba, PR – Brasil
henrique.soares.santos@gmail.com, raque@din.uem.br,
albini@inf.ufpr.br, carmem@inf.ufpr.br

Abstract. Managing a mobile ad hoc network (MANET) is a complex task given the mobility of nodes and resource limitation of the devices. One strategy for simplifying the development of applications in this context is by providing group management services to abstract the network topology and assist the distributed data management. This paper proposes a group management service based on the “Friend of a Friend” (FOAF) ontology. We implemented an NS-2 agent based on the proposed model and simulate the system in a urban scenario in which vehicle drivers exchange information about the traffic. We considered two metrics for validating the system: energy consumption and traffic load.

Resumo. O gerenciamento de redes ad hoc sem fio (MANETs) é uma tarefa complexa devido à mobilidade dos nós e à limitação dos recursos dos dispositivos envolvidos. Uma estratégia para simplificar o desenvolvimento de aplicações neste contexto é fazer uso de serviços de gerenciamento de grupos para abstrair a topologia da rede e auxiliar o gerenciamento de dados distribuídos. Neste trabalho é proposto um modelo de grupo para MANETs baseado na ontologia “Friend of a Friend” (FOAF). Para sua análise foi implementado um agente no NS-2 e simulados alguns cenários em um contexto de troca de informações entre motoristas de uma cidade. As métricas consideradas foram o consumo de energia e troca de pacotes.

1. Introdução

As redes sem fio datam do início da década de 70, quando as universidades das ilhas do Havaí montaram uma rede utilizando microondas para se interligarem, chamada de ALOHAnet [Abramson 1985]. Entretanto, foram nos últimos anos que as redes sem fio se difundiram e ganharam atenção. Para o usuário, a principal vantagem dessas redes é a mobilidade. Utilizando redes sem fio, os usuários podem se mover livremente sem perder a conexão com a rede.

As redes sem fio podem ser divididas em dois grupos: com infra-estrutura e sem infra-estrutura. As redes sem fio com infra-estrutura são caracterizadas pela presença de estação base. Essas estações base são conhecidas nas Redes Locais Sem Fio (WLANS)

– *Wireless Local Area Networks*) como pontos de acesso (APs – *Access Points*). Todas as comunicações efetuadas na rede devem passar pela estação base.

As redes sem fio sem infra-estrutura são o mais comumente chamadas de redes ad hoc móveis (MANETs - *Mobile Ad hoc Networks*). Elas são geralmente formadas de modo dinâmico como um sistema autônomo, por unidades móveis (nós) que se comunicam através de enlaces sem fio. Esses nós se comunicam diretamente uns com os outros, não existindo uma estação base na rede, ou nenhum outro tipo de unidade centralizadora. Cada nó deve ser capaz de se comunicar com os nós que estão dentro do raio de alcance da sua antena. Devido ao raio de alcance limitado das transmissões via rádio, as redes ad hoc são normalmente multi-saltos e os proprietários dos nós devem agir como roteadores das mensagens dos outros nós [Sesay; Yang 2004]. Essas características tornam estas redes propícias para aplicações em diversos cenários, tais como aplicações militares e locais recém-atingidos por catástrofes como furacões e terremotos. Uma característica comum nestes cenários é a troca de informações dentro de “grupos de interesse”. Por exemplo, após uma catástrofe, podem ser formados grupos de prospecção de danos na infra-estrutura, ajuda a feridos e distribuição de alimentos.

A topologia dinâmica e a limitação dos recursos nos dispositivos em uma MANET, tais como limitações de comunicação, armazenamento, processamento e energia, podem ocasionar partícipes na rede e dificultar o gerenciamento de dados distribuídos entre seus nós [HARA 2005] [PADMANABHAN et. al. 2008]. Serviços como o de gerenciamento de grupos podem simplificar e facilitar o desenvolvimento de aplicações para uma MANET, permitindo a troca de informação entre grupos específicos. Dessa forma, quando um nó retorna à rede depois de um particionamento, ele deve ser atualizado somente de acordo com os nós do seu grupo, não dependendo dos demais nós da rede. Essa técnica pode economizar tempo e energia dos dispositivos. Os nós devem trocar informações somente com outros que pertençam ao(s) mesmo(s) grupo(s), aumentando a segurança e reduzindo o tráfego na rede.

Uma das maiores dificuldades da criação de serviços para MANETs é a abstração da topologia da rede. Em [BOULKENAFED; SACCHETTI; ISSARNY 2005] um serviço de gerenciamento de grupos genérico é proposto. Neste modelo, os serviços gerenciam o dinamismo de uma rede detectando a entrada e saída voluntárias ou involuntárias de nós em grupos utilizando um modelo de usuário específico. Este modelo é baseado em uma lista de atributos como localização dos nós, grau de confiança entre os nós, proximidade em relação a um ponto específico e distância em número de saltos. Ao contrário deste modelo, no qual o gerenciamento de grupos utiliza um modelo específico, o presente trabalho propõe um modelo que toma como base algumas classes e atributos de uma ontologia padrão, a “*Friend of a Friend*” (FOAF). Essa ontologia tem sido utilizada atualmente em diversas redes de relacionamento para gerenciar perfis, grupos e listas de amigos. Sendo assim, se um nó já possuir uma instância dessa ontologia, o gerenciamento de grupos poderá utilizá-la automaticamente em seus serviços. As contribuições do artigo são listadas abaixo.

- Proposta de um modelo de usuário baseado na ontologia FOAF para o gerenciamento de grupos em MANETs ;
- Implementação do modelo e realização de simulações de cenários na plataforma NS (*Network Simulator*) versão 2.34. Foi considerado um contexto no qual motoristas em movimento trocam informações sobre a fluidez do trânsito de

uma cidade. Para garantir a consistência dos dados distribuídos, os cenários consideram replicação total e atualização síncrona e assíncrona entre réplicas;

- Análise dos resultados, considerando as métricas de consumo de energia e troca de pacotes entre os nós de um grupo.

O restante deste artigo está organizado da seguinte forma: a Seção 2 apresenta características da ontologia FOAF e introduz conceitos sobre gerenciamento de dados distribuídos; a Seção 3 descreve o modelo proposto, os cenários simulados e analisa os resultados obtidos; a Seção 4 conclui o artigo e cita direções futuras.

2. Definições Preliminares

Nesta seção é apresentada a ontologia FOAF (*Friend of a Friend* – Amigo de um Amigo), sobre a qual o modelo de grupos deste trabalho foi baseado. São abordados também alguns conceitos de gerência de dados distribuídos.

2.1 Ontologia *Friend of a Friend*

A ontologia FOAF foi criada com o objetivo de facilitar o compartilhamento e a utilização de informações de pessoas e suas atividades. Ela permite que grupos de pessoas criem redes sociais sem a necessidade de um banco de dados centralizado [Brickley; Miller 2010].

O projeto FOAF iniciou-se no ano de 2000 como um projeto experimental de informações interligadas. Atualmente, o projeto é mantido pelo consórcio W3C e é utilizado em diversos sítios da Internet para armazenar dados de usuários e gerenciar perfis, grupos, listas de amigos e transferência de informações entre usuários. Vários exemplos são citados por Golbeck e Rothstein (2008), como o blog *LiveJournal* que possui atualmente mais de 28 milhões de contas criadas e 2 milhões de contas ativas. Redes sociais em destaque atualmente, como o Twitter e o Facebook, disponibilizam ferramentas para que o perfil de um usuário seja convertido para FOAF automaticamente (http://wiki.foaf-project.org/w/DataSources#Sites_and_tools_that_export_FOAF).

A ontologia pode ser instanciada por meio de um vocabulário e documentos que podem ser processados de forma automatizada. Estes documentos contêm dados sobre pessoas, suas ligações e atividades ou objetos que elas criam, fazem e possuem interesse. Os participantes da rede podem utilizar os perfis FOAF para encontrar, por exemplo, todas as pessoas que vivem no Brasil, ou ainda uma lista de todas as pessoas que um amigo conhece. Isso é realizado através da definição de ligações entre as pessoas.

A ontologia FOAF define classes, propriedades e associações entre classes. Exemplos de classes incluem: *Agent*, *Person*, *Document*, *Group*, *Image* e *Organization*. Dentre as propriedades de uma pessoa (instância da classe *Person*), encontram-se o nome (*firstName* e *surname*), imagem (*img*) e interesses (*interest*). Há também associações entre classes. A Figura 1 apresenta um exemplo de documento que segue esta ontologia. Na linha 2, a classe *Group* é instanciada criando o grupo “*dois*”. Na linha 4, uma instância da classe *Person*, “*henrique*”, é associada ao grupo por meio da propriedade *member* da classe *Group*. Nas linhas que seguem é definida a associação no sentido inverso, entre a pessoa “*henrique*” e o grupo “*dois*”.

Trabalhos anteriores, tais como *Transhumance* [Paroux et. al. 2007] e *AdhocFS*

[Boulkenafed; Issarny 2003] consideram grupos para gerenciamento de dados. Entretanto, eles não deixam explícitos os atributos considerados para o gerenciamento de grupos. O principal objetivo do modelo proposto neste trabalho e apresentado na Seção 3, é distinguir quais nós de uma rede podem ou não trocar mensagens entre si por meio de agrupamentos de nós definidos através da ontologia FOAF.

```

1 <!-- http://xmlns.com/foaf/0.1/dois -->
2 <Group rdf:about="&foaf;dois">
3   <rdf:type rdf:resource="&owl;NamedIndividual"/>
4   <member rdf:resource="&foaf;henrique"/>
5 </Group>
6 <!-- http://xmlns.com/foaf/0.1/henrique -->
7 <Person rdf:about="&foaf;henrique">
8   <rdf:type rdf:resource="&owl;NamedIndividual"/>
9   <member rdf:resource="&foaf;dois"/>
10</Person>
```

Figura 1: Exemplo na ontologia FOAF

2.2 Roteamento e Replicação

Os nós de uma MANET devem ser capazes de se comunicar mesmo quando estão em movimento. Isso cria um novo desafio à computação distribuída, que é a possibilidade de acessar informações em qualquer tempo e lugar. Sendo assim, aplicações criadas para esses ambientes devem definir quais parâmetros deverão ser levados em consideração para o gerenciamento de dados. Além disso, devem ser consideradas as limitações dos dispositivos, bem como a disponibilidade de banda, as frequentes alterações na topologia e o consumo de energia. As aplicações devem ser estruturadas de forma a lidar com a rápida mudança de dados e a variação de desempenho e confiabilidade da conectividade.

Visto que a comunicação entre os nós é ponto-a-ponto, o roteamento de pacotes pode ser do tipo *unicast* ou *multicast*. De uma forma geral, segundo Kant e Awasthi (2010), se um nó quer enviar um pacote para um grupo com três nós, no *unicast* o emissor envia o pacote três vezes, uma para cada receptor. Já no *multicast* o emissor envia somente um pacote para todos os membros do grupo. Entretanto, para usar comunicação *multicast*, diversos outros parâmetros são necessários aumentando a sobrecarga da rede. Dentre os protocolos do tipo *unicast* pode ser citado o AODV [Perkins; Royer; Das, 1999], sendo o PUMA [Garcia; Vaishampayan 2004] um exemplo de protocolo de roteamento *multicast*.

Segundo Hara (2005), a replicação de dados é um ponto importante no gerenciamento de dados em sistemas distribuídos e pode solucionar problemas importantes, como o particionamento de uma rede. Por exemplo, considere que o nó A está executando uma transação utilizando dados que estão alocaados no nó B. Caso o nó se movimente, desconectando-se da rede, se não houver réplicas dos dados de B em outros nós da rede, A não poderá continuar a transação. Havendo réplicas, o nó A continuará seu processamento normalmente usando uma réplica. Posteriormente, o nó B deverá ser atualizado para garantir a integridade dos dados distribuídos. Isso leva a um outro ponto importante que é a consistência entre réplicas. A cada alteração dos dados nos nós, todas as réplicas devem ser atualizadas, garantindo a sua consistência.

O método de replicação pode ser total ou parcial. Na replicação total, os dados

são mantidos em todos os nós integrantes de uma rede. Essa abordagem possui um protocolo de propagação simples e não exige um gerenciamento complexo, uma vez que todas as alterações dos nós são enviadas a todos os integrantes da rede. Já na abordagem de replicação parcial, os dados não são replicados para todos os nós da rede, sendo criados subconjuntos de réplicas. Isso poupa espaço na memória dos dispositivos e diminui a quantidade de mensagens para sincronização das réplicas. Porém, ela necessita de um protocolo de propagação mais complexo que o método de replicação total.

Os mecanismos para controle de consistência entre réplicas podem ser classificados de acordo com dois parâmetros: local (quais réplicas podem ser atualizadas) e tempo (quando as réplicas são atualizadas) [Martins; Pacitti; Valduriez 2006]. Quanto ao primeiro, os protocolos de replicação podem ser classificados como *single-master* ou *multi-master*. Nos protocolos *single-master* um item de dado pode ter várias réplicas, porém somente uma delas pode aceitar operações de leitura e escrita. As demais réplicas aceitam somente operações de leitura. Já nos protocolos *multi-master* todas as réplicas aceitam leitura e escrita. Protocolos *multi-master* permitem a concorrência de dados, porém tornam o gerenciamento mais complexo que nos protocolos *single-master*.

Quanto ao parâmetro de tempo, a atualização das réplicas pode ser classificada como síncrona ou assíncrona. Na atualização síncrona, todas as réplicas de um item de dado são sincronizadas antes que o dado seja efetivamente atualizado. Já na assíncrona, as réplicas são sincronizadas periodicamente. Nas duas metodologias existem vantagens e desvantagens. Na atualização síncrona, pode-se garantir que em um determinado momento, todas as réplicas de um item de dado estão com os mesmos valores, enquanto que na assíncrona isso não é possível. Por outro lado, a atualização síncrona pode se tornar um gargalo na execução do sistema, o que pode ser aliviado com a utilização do método assíncrono.

3. *GroupModel*: Um Modelo de Gerenciamento de Grupos

O principal objetivo do modelo proposto neste trabalho, o *GroupModel*, é prover a funcionalidade de definir grupos de interesse, que podem ser constituídos por membros relativamente estáveis ou transitórios. Estes agrupamentos são definidos através da ontologia FOAF e determinam quais nós (membros) da rede podem trocar mensagens entre si, restringindo assim o número de nós nos quais determinados dados são compartilhados. Os estudos experimentais realizados mostram que esta funcionalidade pode proporcionar uma economia no consumo de energia de até 42% em dispositivos móveis.

Para a definição do modelo, algumas classes e atributos da ontologia FOAF foram considerados:

- O atributo *Group* foi definido com base nas classes *Group* e *Person* da ontologia FOAF. *Person* representa um nó que pode ser incluído em um grupo existente por meio da propriedade *member* da classe *Group*. A classe *Group* representa os membros dos grupos na ontologia. Esse atributo é usado no momento do envio de um pacote. O pacote é enviado somente para os membros de um grupo. Portanto o nó já tem conhecimento prévio dos nós que poderão receber o pacote, seguindo a constituição do grupo.
- O atributo *Interest* do modelo foi definido com base na classe *Person* e no

atributo *Interest* da classe *Agent*. A classe *Agent* representa uma superclasse de *Person* na ontologia FOAF. Com a utilização desse atributo, um nó indica qual o interesse, de sua lista, que é considerado para cada pacote enviado. O pacote é então enviado para todos os pares da rede e recebido por aqueles que possuem o mesmo interesse que aquele estabelecido pelo nó origem. O pacote é descartado pelos nós que não tem interesse no pacote.

Sendo assim, um gerenciador de grupos que segue o modelo proposto pode considerar dois métodos para o envio de mensagens: por grupo ou por interesse. O primeiro considera o atributo *Group* e sugere a formação de grupos cujos membros são conhecido a priori e, portanto, com participação “estável”. O segundo considera o atributo *Interest*, o que indica uma participação transitória; ou seja, um determinado nó pode, em um determinado instante, ter interesse em receber informações a respeito de um determinado assunto, mas deixar de tê-lo a qualquer momento. Cada nó pode participar de vários grupos e possuir vários interesses.

Para o roteamento de pacotes, foram considerados dois protocolos de roteamento: o protocolo *unicast* AODV e protocolo *multicast* PUMA. Quando o grupo é formado através do atributo *Interest*, é utilizado o protocolo AODV e todos os nós da uma rede recebem o pacote. Quando o envio é feito considerando o atributo *Group*, o protocolo PUMA viabiliza uma entrega direcionada de pacotes para os membros de um determinado grupo.

3.1 Simulações

Para a análise do *GroupModel* foi implementado um agente no simulador de redes NS-2.34, chamado de *GroupAgent*. Seis cenários foram simulados na ferramenta, considerando uma aplicação do mundo real, que permite a motoristas de uma metrópole trocar informações sobre a situação do tráfego nas regiões nas quais já estiveram. Foram gerados 10 nós (representados pelos carros), distribuídos em três grupos (representados por bairros): Zona Oeste, Centro e Zona Leste. A Figura 2 ilustra o cenário da simulação.

Para todas as simulações, foi adotada a replicação total dos dados, ou seja, os nós replicam seus dados para todos os pares envolvidos. Conforme o cenário, a atualização das réplicas é síncrona ou assíncrona, considerando ou não o modelo *GroupModel*. Além disso, a política de atualização adotada é que todos os nós tem permissão para alterar as réplicas.

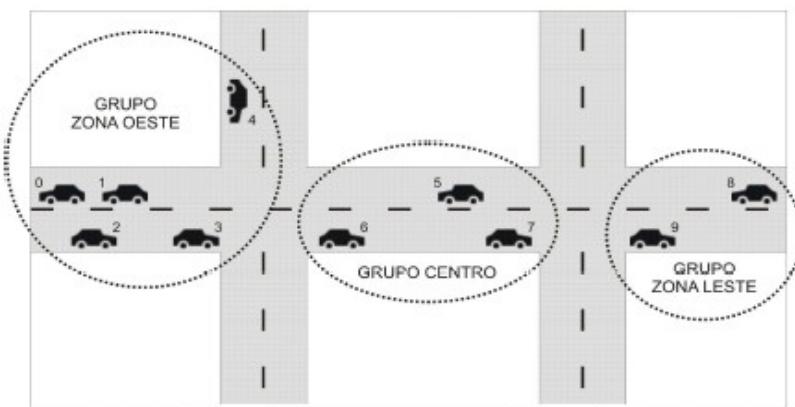


Figura 2. Cenário das simulações

Durante a simulação, os nós movimentam-se arbitrariamente seguindo o modelo *Random Waypoint Model*. Em determinados instantes, alguns nós alteram dados armazenados e replicam suas informações para os demais. Na atualização assíncrona, a propagação é realizada a cada 5 segundos. Já na atualização síncrona, a cada alteração todas as réplicas são atualizadas.

Os cenários considerados nas simulações são descritos abaixo:

- não considerando o *GroupModel*, com atualização síncrona entre as réplicas;
- não considerando o *GroupModel*, com atualização assíncrona entre as réplicas;
- considerando o atributo *Group*, com atualização síncrona entre as réplicas;
- considerando o atributo *Group*, com atualização assíncrona entre as réplicas;
- considerando o atributo *Interest*, com atualização síncrona entre as réplicas;
- considerando o atributo *Interest*, com atualização assíncrona entre as réplicas.

Quando um cenário considera o atributo *Group*, a comunicação entre os nós só pode ocorrer se eles fizerem parte de um mesmo grupo. Por exemplo, entre nós do grupo '*Zona Leste*'. Quando o atributo é *Interest*, os nós devem ter um interesse em comum em seu perfil. Por exemplo, dois nós podem trocar informações se planejam ir para a região oeste da cidade e expressam interesse na '*Região Oeste*'.

Com relação ao modelo utilizado, é possível fazer as seguintes observações a respeito da utilização dos atributos *Group*, *Interest* e sem a utilização de qualquer tipo de agrupamento:

- Nas simulações que não utilizam o *GroupModel*, quando um dado é alterado em algum nó, a sincronização das réplicas é feita para todos os nós da rede. Nesse método, existe um alto tráfego na rede e não existem restrições de acesso aos dados entre os integrantes da rede, uma vez que os nós conhecem todas as informações de todos os pares da rede.
- Quando utilizado o atributo *Group* do modelo, o nó tem conhecimento, no momento do envio dos pacotes, para quais nós ele deve enviar um pacote. Esse método aumenta a segurança na rede, pois somente os membros de um mesmo grupo podem trocar mensagens entre si. Outra vantagem é que o tráfego na rede é diminuído, pois a replicação de dados não é feita para todos os nós, mas somente para os membros do grupo. A diminuição no tráfego de dados resulta em economia de energia para os nós.
- Com a utilização do atributo *Interest*, o nó necessita enviar um pacote a todos os pares da rede, pois ele desconhece quais pares compartilham o mesmo interesse. Com isso, essa simulação inicialmente se assemelha às simulações que não utilizam o modelo de grupo. Porém, nela um grupo pode ser formado espontaneamente de acordo com o interesse dos nós.

Com relação à sincronização das réplicas, é possível observar que a atualização síncrona pode diminuir a inconsistência dos dados, uma vez que a sincronização é realizada sempre que ocorre uma alteração em um nó. Porém, os recursos da rede são utilizados com mais freqüência, aumentando o tráfego e o consumo de energia dos nós. Muitas vezes ocorrem sincronizações que não precisariam ser feitas, pois nenhum outro

nó necessita da informação replicada. Nos cenários simulados utilizando a sincronização assíncrona, ela é realizada a cada 5 segundos. Mesmo quando ocorrem várias atualizações em um mesmo dado durante esse período, somente a última ocorrida antes do momento da sincronização é propagada a seus pares. Esse método tem por objetivo a redução do tráfego de pacotes na rede, resultando em economia de energia para os nós. Porém, esta abordagem aumenta a possibilidade de ocorrerem leituras de dados inconsistentes, uma vez que um determinado nó pode estar utilizando valores antigos até que a sincronização seja feita.

Para analisar os cenários, primeiramente foi comparado o consumo de energia dos nós e, em seguida, a quantidade de pacotes trocados pelo *GroupAgent*. A análise foi pontual, ou seja, somente o comportamento do agente implementado foi considerado.

- Consumo de energia: cada nó iniciou com energia igual a 100 unidades. A cada transmissão de pacote, ele gastou 0,6 unidades de energia e a cada recepção foram gastos 0,2 unidades de energia. Conforme analisado, a energia utilizada pelos nós foi a mesma não considerando grupos ou grupos formados por *Interest*. Isso aconteceu nos dois modos de atualização de réplicas. Já considerando o atributo *Group* a energia utilizada pelos nós foi aproximadamente 42,72 % menor em relação ao atributo *Interest*.
- Tráfego de pacotes: o número de pacotes enviados/recebidos foi diretamente proporcional ao consumo de energia; ou seja, quanto maior o número de pacotes enviados e recebidos, maior foi o gasto de energia. A troca de pacotes foi menor com o atributo *Group* e maior quando um agrupamento não é considerado ou o atributo *Interest* é considerado. Isso é devido ao fato do atributo *Group* utilizar um protocolo de roteamento *multicast*, enquanto os dois últimos métodos utilizam um protocolo de roteamento *unicast*. Logo, no *multicast* um pacote é enviado uma única vez por um nó e todos os nós de um grupo o recebem. No *unicast* um nó envia um pacote para cada nó destinatário.

Para finalizar a análise, foi gerado um cenário maior e foi realizada uma breve simulação com 50 nós e 20 atualizações de dados replicados em momentos aleatórios. Foram formados dois grupos com 15 nós cada e um terceiro com 20 nós. Nesse cenário foi possível observar uma economia de energia de aproximadamente 94,45% com a utilização do atributo *Group* em relação à utilização do atributo *Interest*. O número de pacotes enviados e recebidos pelo *GroupAgent* foi diretamente proporcional ao consumo de energia. Considerando esse cenário, ficou ainda mais evidente a vantagem da utilização do atributo *Group*, no qual o aumento do número de nós e a disposição dos grupos resultou em um grande ganho de desempenho na rede.

4. Conclusão

Redes sociais populares vêm empregando a ontologia FOAF para definir os perfis de seus usuários. Sendo assim, os milhares de usuários que já possuem um perfil neste padrão poderão utilizar o serviço proposto neste artigo para realizar trocas de informações em uma MANET, sem a necessidade de criar um novo perfil.

Considerando as métricas consumo de energia e troca de pacotes na análise dos cenários que utilizam o *GroupModel*, foi possível concluir que a redução no tráfego da rede e a consequente diminuição do consumo de energia se deram principalmente com a utilização do atributo *Group* do modelo. Minimizar o consumo de energia é sempre uma

preocupação nas aplicações em uma MANET, uma vez que isso implica em menos partições de rede e desconexões de nós. A estabilidade da rede, por consequência, facilita o gerenciamento de dados distribuídos. Com o atributo *Group*, nós de uma rede podem ser agrupados em “sub-redes” facilitando o gerenciamento de dados por parte de sistemas distribuídos.

Com a utilização do atributo *Interest*, as simulações se mostraram equivalentes ao uso de modelos tradicionais. Porém, o uso do atributo *Interest* apresenta vantagens porque grupos podem ser formados espontaneamente de acordo com um interesse de um conjunto de nós. Isto é interessante pois redes podem ser formadas entre usuários desconhecidos que possuem interesses em comum em locais arbitrários.

Um outro ponto que merece destaque é que a segurança em uma rede pode ser mais alta com a utilização do atributo *Group*, uma vez que apenas nós que pertençam à um mesmo grupo podem se comunicar. Já com a utilização do atributo *Interest* ou no modelo tradicional, o envio é feito para todos os nós da rede, dificultando assim a manutenção da segurança.

Embora o artigo apresente resultados preliminares da utilização do conceito de grupos em MANETs, os resultados obtidos com a utilização do atributo *Group* demonstram o potencial da abordagem. Dentre as direções futuras deste trabalho podem ser citados:

- Eleição de um nó líder para cada grupo estável: para que o atributo *Group* seja utilizado por um gerenciador de grupos, todos os nós devem ter seus grupos atualizados. Sendo assim, existe a necessidade de um nó líder para aceitar inscrições de nós em grupos e refletir atualizações necessárias para os nós que pertençam aos grupos alterados. Esse nó líder deve ter uma carga de energia satisfatória para gerenciar alterações de grupos e também deve ter uma alta disponibilidade. Vários protocolos de sistemas distribuídos tratam de eleição de líderes considerando características como nível de energia e partição de redes.
- Controle de vocabulário: dois nós podem ter um mesmo interesse, porém, com palavras-chave de interesse diferentes. Uma extensão interessante seria considerar sinônímia no processo de verificação de interesses em comum entre usuários. Trabalhos na área de processamento semântico de texto tratam deste tipo de detecção.
- Replicação e sincronização: um estudo considerando todas as camadas de uma MANET seria interessante para analisar o modelo proposto como um todo. O estudo de formas alternativas de replicação e sincronização, tais como a replicação parcial e sincronização híbrida [Padmanabhan et. al. 2008] serão investigadas no futuro.

Referências

- Abramson, N. (1985). Development of the alohanet. IEEE Transactions on Information Theory, 31.
- Boulkenafed, M.; Issarny, V. (2003) “AdHocFS: Sharing Files in WLANS”, Second IEEE International Symposium on Network Computing and Applications, Massachusetts (USA).

- Boulkenafed, M.; Sacchetti, D.; Issarny, V. (2005) "Using Group Management to Tame Mobile Ad Hoc Networks", Proceedings of the 6th International Conference on Mobile Data Management, Nicosia (Cyprus), pp. 192-199.
- Brickley, D.; Miller, L. (2010) "FOAF Vocabulary Specification 0.98", Namespace Document 9 August 2010, FOAF Project, <http://xmlns.com/foaf/spec/>.
- Garcia, J., J.; Vaishampayan, R. (2004) "Efficient and Robust Multicast Routing in Mobile Ad Hoc Networks", Proceedings of the IEEE International Conference on Mobile Ad Hoc and Sensor Systems, Florida (USA), pp. 304-313.
- Golbeck, J.; Rothstein, M. (2008) "Linking Social Networks on the Web with FOAF", Proceedings of the 17th International World Wide Web Conference (WWW2008), Beijin (CN).
- Hara, T. (2005) "Data Replication Issues in Mobile Ad Hoc Networks", Proceedings of the 16th International Workshop on Database and Expert Systems Applications (DEXA'05), Copenhagen (DNK).
- Kant, K.; Awasthi, L. K. (2010) "Unicast and Multicast Routing Protocols for Manets: A Comparative Survey", International Journal of IT & Knowledge Management (IJITKM), ISSN: 0973-4414, Special issue, January.
- Martins, V.; Pacitti, E.; Valduriez, P. (2006) "Survey of data replication in P2P systems", Technical Report, Institut National de Recherche en Informatique et en Automatique, Nantes (FRA).
- Padmanabhan, P. et. al. (2008) "A Survey of Data Replication Techniques for Mobile Ad Hoc Network Database", The VLDB Journal — The International Journal on Very Large Data Bases, Volume 17 Issue 5, August.
- Paroux, G. et al. (2007) "Transhumance: A power sensitive middleware for data sharing on mobile ad hoc networks", International Workshop on Applications and Services in Wireless Networks, Santander (ESP).
- Perkins, C.; Royer, E.; Das, S. (1999) "Ad hoc on demand distance vector (AODV) routing", Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, New Orleans (USA), pp. 90-100.
- Sesay, S.; Yang, Z.; He, J. (2004) "A Survey on Mobile Ad Hoc Wireless Network", Information Technology Journal 3(2), pp. 168-175.

Uma Ferramenta para Geração Automática de Código Java a partir do Modelo de Dados

Jansser Ribeiro¹, Renato Lima Novais¹

¹Grupo de Informática Aplicada - Instituto Federal da Bahia – Campus Santo Amaro
CEP 44.200-000 – Santo Amaro – BA – Brasil

jansser_costa@hotmail.com, renatonovais@gmail.com

Abstract. *Development of information systems usually implies in codification of basic and repetitive activities that manipulate the data of the tables in the database. This paper presents a tool that automates these steps, thus contributing to increased productivity in the software development process.*

Resumo. *Construção de sistema de informação normalmente se dá pela realização de atividades básicas e repetitivas de cadastros das tabelas do banco de dados. Este trabalho apresenta uma ferramenta que automatiza esses passos, contribuindo assim para o aumento da produtividade no processo de desenvolvimento de software.*

1. Introdução

Durante o processo de desenvolvimento de software de Sistema de Informação (SI), os desenvolvedores se deparam cotidianamente com tarefas de codificação básicas. Como exemplos, podem ser citadas tarefas de inclusão, exclusão, atualização e seleção das informações das tabelas básicas do sistema. Considerando o contexto de orientação a objetos, devem ser criadas, para cada uma dessas entidades, tanto as classes de persistência que manipulam o banco de dados quanto às classes que implementam as interfaces com o usuário.

Essas tarefas básicas consomem um tempo considerável do processo de desenvolvimento de software (MENIN, 2005, p.15). Por outro lado, seria interessante se este tempo pudesse ser dedicado principalmente às tarefas de implementação das funcionalidades mais importantes dos sistemas. É possível observar que existe uma repetição no desenvolvimento dessas tarefas. Da mesma forma, existe um padrão de codificação das mesmas, permitindo que elas possam ser automatizadas. Isto pode ser feito, baseando-se nas informações dos metadados das tabelas, os quais definirão como serão montados os SQLs e as telas das tabelas básicas.

Tendo em vista esse contexto, esse trabalho, apresenta uma ferramenta que proporciona uma alternativa automatizada para a realização das tarefas de codificação citadas anteriormente. Essa ferramenta comunica-se com dois bancos de dados bastante utilizados na atualidade – MySQL e PostgreSQL –, recupera os metadados das tabelas. Com posse dos metadados ela gera, de maneira eficaz e simples, toda estrutura básica do software na linguagem JAVA. A ferramenta foi desenvolvida como *plug-in* de uma das IDEs (*Integrated Development Environment*) mais utilizadas atualmente: o Eclipse (CARDOSO, 2009).

A ferramenta traz diversos benefícios, dentre os quais, podem ser citados: i) aumento da produtividade da equipe de desenvolvimento; ii) ganho de qualidade interna do projeto, uma vez parte do código fonte é gerado automaticamente; iii) redução de erros, pois, por mais que as atividades são básicas, desenvolvimento de software sempre está sujeito a inclusão de defeitos, etc.

2. A Ferramenta

A ferramenta foi desenvolvida utilizando a linguagem Java e integrada ao IDE Eclipse sob a forma de *plug-in*. Na figura 1 são apresentados os principais passos do funcionamento da ferramenta: primeiramente a ferramenta conecta com o banco de dados escolhido (i), extrai os metadados (ii) e por fim, gera as classes do programa (iii).

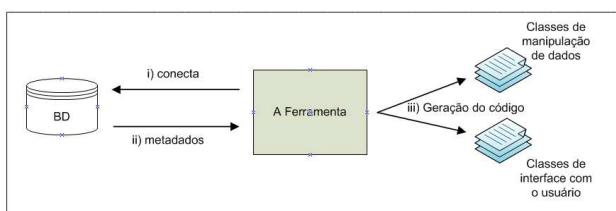


Figura 1. Funcionamento geral da ferramenta

Tendo em vista as particularidades estruturais de cada SGBD (Sistema Gerenciador de Banco de Dados) utilizado, procurou-se desenvolver a ferramenta de uma maneira que pudesse ser estendida para outros SGBDs. Para tal, foi utilizado o padrão de projeto Strategy (GAMMA et al, 1995). A ferramenta foi inicialmente desenvolvida para MySQL e PostgreSQL, que será detalhada a seguir.

2.1. Extrairando metadados do MySQL e PostgreSQL

Para que um sistema seja desenvolvido a partir de um modelo de dados faz-se necessário o conhecimento dos metadados desse modelo. A extração dos metadados é o passo fundamental para tais ferramentas, uma vez que ela extrai as informações necessárias para a geração do código fonte.

Nesta ferramenta foi realizada a extração dos metadados através da utilização de APIs (*Application Programming Interface*) do JDBC (*Java Database Connectivity*). As principais informações a serem extraídas pela ferramenta são: a) os nomes das tabelas contidas em um banco de dados; b) as colunas de uma determinada tabela e os seus tipos.

A extração dos metadados foi feita utilizando as classes *DatabaseMetaData* e a *ResultSetMetaData* da API JDBC. Tais classes possuem métodos que facilitam a captura de metadados, como por exemplo, o método “*getSchemas();*” que retorna todos os *schemas* de um servidor de banco de dados.

Cada SGBD possui especificidades que os diferenciam. Como exemplo pode ser citado o MySQL, que identifica um banco de dados como um “*catalog*”. Por sua vez, o PostgreSQL utiliza o termo “*schemas*”. Além disso, existem nomenclaturas diferentes para tipos de dados nesses dois SGBDs, as quais devem ser tratadas pela ferramenta. Tais diferenças vão influenciar diretamente na implementação das APIs do JDBC.

2.2. Tipo de dados

Uma questão fundamental para se desenvolver sistemas de geração automática a partir do modelo de dados é o mapeamento dos tipos de dados do SGBD para a linguagem escolhida. A tabela 1 apresenta um mapeamento, construído a partir das observações desse trabalho, dos tipos de dados do MySQL para tipos de dados Java, componentes Swing e ainda componentes HTML.

Tipo do Banco de dados	Tipo Java	Componente Swing	Componente Web (HTML)
VARCHAR	String	Text Field	Input Text
TEXT	String	Text Area	Input Text
BIT	Boolean	Radio Button ou CheckBox	Radio Button ou CheckBox
SmallInt	Short	Text Field	Input Text
Decimal	Java.math.BigDecimal	Text Field	Input Text
Int	Int	Text Field	Input Text
Integer	Int	Text Field	Input Text
Float	Double	Text Field	Input Text
Double	Double	Text Field	Input Text
Real	Float	Text Field	Input Text
Binary	Byte[]	Text Field	Input Text
Date	Java.sql.Date	Text Field	Input Text
Time	Java.sql.Time	Text Field	Input Text

Tabela 1: Mapeamento dos tipos de dados do MySQL para o Java (Fonte Própria).

2.3. Geração de código

Após a extração dos metadados, é feita a transformação das informações extraídas em código-fonte. Para a geração automática de código foram utilizadas APIs fornecidas pelo JDT (*Java Development Tools*), que permitem acessar e manipular código-fonte Java (VOGELLA, 2010). Uma classe no JDT é chamada de “ICompilationUnit”. A figura 2 mostra a criação de uma classe de maneira programática.

```
String sourceCode = "public Class BD ()"; Código fonte da classe.
ICompilationUnit ClassDB = pacote.createCompilationUnit("BD.java", sourceCode, true, null);
↑
Pacote Nome do arquivo Código fonte
```

Figura 2. Gerando uma classe com JDT.

3. Exemplo de uso

Na figura 2 é apresentada a ferramenta em funcionamento para uma tabela de banco de dados simples. No topo da Figura 2, têm-se, da esquerda para direita, as seguintes telas: conexão com o banco de dados, seleção do banco de dados e seleção das tabelas do banco de dados, para as quais serão gerados os códigos fontes correspondentes.

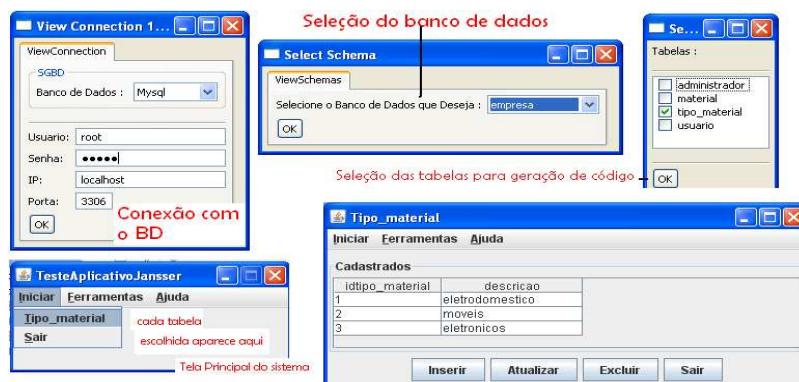


Figura 3. Funcionamento geral da ferramenta

Na parte de baixo da figura 3, é possível observar as telas do programa gerado pela ferramenta já em execução. A primeira tela representa a tela principal do sistema. A segunda tela representa a tela principal que é gerada para cada tabela, a qual permite o usuário fazer as operações básicas sobre os dados da tabela.

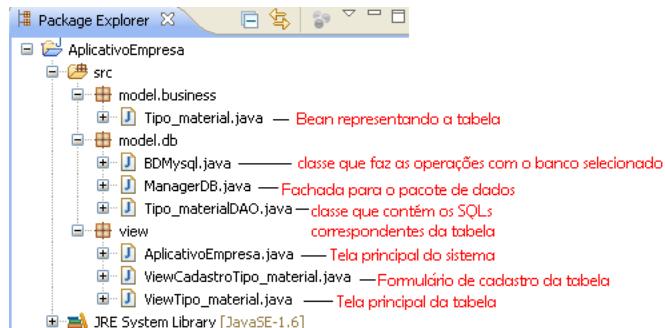


Figura 4. Classes do programa gerado

Na figura 4 é apresentada da estrutura do projeto Java gerado no Eclipse, mostrando cada uma das classes geradas para a tabela selecionada.

4. Conclusão

Este trabalho teve como objetivo principal o desenvolvimento de uma ferramenta para geração automática de código a partir de um modelo de dados. Tal objetivo foi alcançado com êxito, de modo que a ferramenta construída atendeu adequadamente as expectativas.

Um ponto a se destacar do trabalho é a integração da ferramenta com o IDE Eclipse, um dos mais populares ambientes de desenvolvimento da atualidade, possibilitando a geração de código inserida no mesmo ambiente de trabalho, sem a necessidade de utilizar programas externos. A facilidade na adequação de outros SGBDS por colaboradores, devido à estrutura na qual a ferramenta foi codificada é outra característica que deve ser enfatizada como um ponto positivo.

A ferramenta possui algumas limitações, dentre elas, não possuir suporte a chaves estrangeira e tratamento de dados. Como trabalhos futuros, pretende-se estendê-la para suporte às limitações apresentadas, além de suporte a outros SGBDs.

Referências

- CARDOSO, Luis. Desenvolvimento de um Plug-in para Extração de Estruturas de Código Fonte - Regras de Negócio. UFBA. 2009.
- ECLIPSE. Disponível em <<http://www.eclipse.org/>>. Acesso em: 20 outubro de 2010.
- MENIN, Juliane. GERADOR DE CÓDIGO JSP BASEADO EM PROJETO DE BANCO DE DADOS MYSQL. Universidade Regional de Blumenau, 2005.
- VOGELLA, Eclipse JDT - Abstract Syntax Tree (AST) and the Java Model – Tutorial. Disponível em: <<http://www.vogella.de/>> Acesso em: 10 de outubro 2010.
- GAMMA, E., HELM, R., JOHNSON, R. & VLISSIDES, J. (1995). Design patterns: Elements of reusable object-oriented software. Reading, MA: Addison Wesley.

Implementação de um Simulador de Consultas em Álgebra Relacional

Larissa R. Lautert¹, Deise de B. Saccò¹

¹Curso de Ciência da Computação – Universidade Federal de Santa Maria
(UFSM) – Santa Maria – RS – Brasil

{llautert,deise}@inf.ufsm.br

Abstract. *A Relational Database is represented by a collection of tables, where each tuple represents a relation between a set of values. To submit queries in relational databases it can be used Relational Algebra. Because it is a formal language, few tools interpret queries in relational algebra format, making students learning harder. This work proposes a relational algebra query simulator software to support relational algebra operations learning. We analyzed existent software needs and implemented a tool based on these requirements.*

Resumo. *Um Banco de Dados Relacional é representado por uma coleção de tabelas, onde cada linha representa uma relação entre um conjunto de valores. Para efetuar consultas em bancos de dados relacionais, pode-se utilizar a Álgebra Relacional. Por se tratar de uma linguagem formal, poucas ferramentas interpretam consultas em álgebra relacional, dificultando a aprendizagem dos alunos. O presente trabalho propõe um programa simulador de consultas em álgebra relacional para auxiliar na aprendizagem das operações fundamentais. Foram exploradas as carências dos softwares existentes e implementada uma ferramenta com base nessas deficiências.*

1. Introdução

O estudo e a compreensão da álgebra relacional são fundamentais para estudantes de cursos de Banco de Dados. Para facilitar seu aprendizado, é conveniente ter um ambiente para testes de consultas nessa linguagem. Existem alguns *software* que permitem tais testes, dentre os quais destacam-se *LEAP* [1998] e *DBTools 2000* [2000]. Ambos deixam a cargo do usuário a instalação e configuração do banco de dados, sendo que esta atividade é mais trabalhosa no *DBTools 2000*. O primeiro suporta diversas operações além das fundamentais da álgebra relacional, como criação, edição e remoção de tabelas e até do banco de dados. Porém, possui uma interface não intuitiva e não utiliza os símbolos e a sintaxe adotados pela maioria dos livros de banco de dados. Em oposição, o segundo possui interface bastante amigável para criação de consultas e apresentação de seus resultados. No entanto, durante a execução do programa percebe-se que algumas exceções não foram tratadas quando, por exemplo, o usuário digita uma consulta incorreta.

As ferramentas disponíveis analisadas são pouco utilizadas devido aos pontos negativos relatados. Constatou-se a necessidade de um *software* estável, funcional, intuitivo e que siga o padrão de simbologia dos livros. Com ele, os alunos poderão utilizar os conceitos aprendidos em aula para criar consultas e verificar se os resultados correspondem aos esperados. Por isso, este trabalho propõe a implementação de um interpretador de consultas em álgebra relacional com esses requisitos.

2. A Ferramenta *SimAlg*

A ferramenta *SimAlg* tem como finalidade a simulação de consultas em álgebra relacional com as operações de seleção, projeção, união, intersecção, diferença, produto cartesiano e junção natural. O programa já vem com um banco de dados pronto para a execução de consultas, assim como nos outros *software* analisados.

Para simular a execução de consultas em álgebra relacional, a ferramenta realiza uma conversão para SQL, executa no banco de dados e o resultado é mostrado na tabela presente na interface gráfica. O programa possui um tratamento de erros bem completo com mensagens para guiar o usuário no caso de utilização incorreta.

Sua interface gráfica é composta por quatro abas. As três primeiras são destinadas a realização de consultas e a quarta, a visualização das tabelas presentes no banco de dados. Contém ainda um menu de ajuda para auxílio em caso de dúvidas quanto à utilização do programa e à álgebra relacional. As seções a seguir explicam a utilização da ferramenta.

2.1. Operação de Seleção

Na interface de seleção, o usuário tem possibilidade de executar uma consulta com até duas condições. Deve-se compor a consulta através dos menus *dropdown* e campo de texto para valor de comparação. Após selecionar a(s) condição(ões) desejada(s), o usuário deve clicar no botão “Executar consulta” para conferir o resultado na tabela abaixo. O SQL equivalente aparecerá abaixo do botão. O sistema permite salvar a tabela resultante da consulta, possibilitando a execução de consultas em que o resultado de uma operação é a entrada de outra.

A figura 1 mostra a execução de uma consulta que seleciona os alunos que ingressaram no grupo PET no ano de 2009. A execução de consultas, visualização do resultado e opção de salvar a tabela resultante são iguais para as três abas de operações.

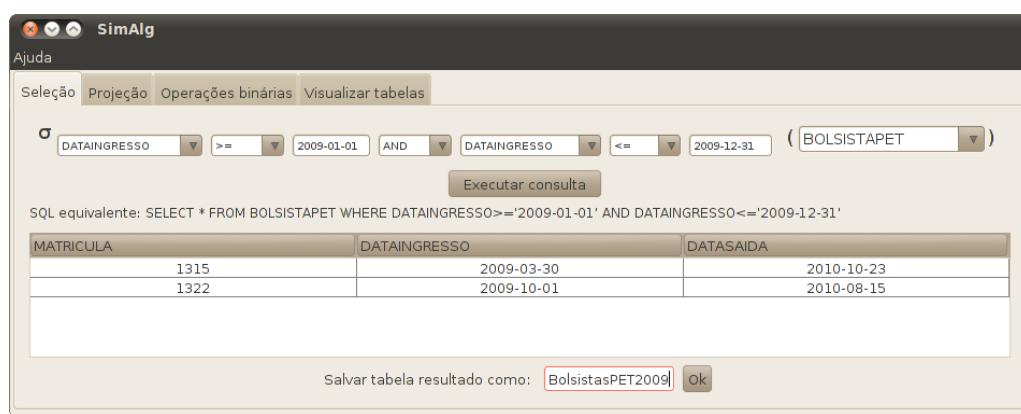


Figura 1. Seleção dos alunos que ingressaram no grupo PET no ano de 2009.

2.2. Operação de Projeção

A segunda aba é destinada a operações de projeção, onde o usuário pode fazer uma consulta que projete até três colunas. A figura 2 mostra o fragmento da interface onde as consultas de projeção são compostas.



Figura 2. Interface para construção de consultas de projeção.

2.3. Operações Binárias

A terceira aba possui interface para as operações binárias de união, intersecção, diferença, produto cartesiano e junção natural. O usuário seleciona duas tabelas nos menus *dropdown* e a operação entre elas, conforme figura 3.



Figura 3. Interface para construção de consultas com operações binárias.

2.4. Visualização de Tabelas

A fim de facilitar a construção das consultas, a quarta aba do programa permite saber quais tabelas e dados estão presentes no banco de dados. Basta selecionar no menu *dropdown* a tabela que deseja-se visualizar e seu conteúdo é exibido. Tabelas resultado de outras operações salvas durante a execução do programa também podem ser visualizadas.

3. Implementação

A ferramenta foi desenvolvida em linguagem Java e utiliza o Sistema Gerenciador de Banco de Dados HSQLDB 2.0 [2010] por não precisar de instalação no computador em que o programa será executado, tornando o uso da ferramenta mais prático.

O programa possui dez classes, dentre as quais destacam-se as cinco a seguir:

- GUI: contém todos os componentes da interface gráfica. Invoca métodos da classe GUIDAO para a execução das operações, obtenção do SQL equivalente e criação de tabelas.
- GUIDAO: apresenta uma abstração para as operações realizadas na classe GUI envolvendo consultas e tabelas. Possui três métodos para conversão da consulta de álgebra relacional para SQL, sendo um para cada aba de operações do sistema.
- DBConnection: possui os métodos que conectam com o banco de dados HSQLDB e realizam as consultas. Cria as tabelas do banco de dados que vem com a aplicação e insere tuplas nestas. Se o usuário desejar alterar a base de dados para as consultas, deve alterar os códigos SQL presentes nesta classe.
- BancoDados: implementa uma lista com todas as estruturas das tabelas utilizadas no banco de dados. Seu principal método, utilizado para compor os menus *dropdown* da interface gráfica, retorna o nome de todas as tabelas.
- Tabela: apresenta uma abstração para a estrutura das tabelas armazenadas no banco de dados. Possui métodos que retornam os tipos e os nomes das colunas de uma tabela, importantes para a composição dos menus *dropdown* da interface e geração do SQL de criação de tabela.

Os códigos fonte e a aplicação estão disponíveis em <http://goo.gl/qcDaV>.

4. Comparação entre *LEAP*, *DBTools 2000* e *SimAlg*

A ferramenta *SimAlg* não necessita de instalação nem configuração relativa ao banco de dados, de forma que sua execução constitui uma vantagem sobre o *LEAP* e o *DBTools 2000*. A interface gráfica do programa, com espaço para visualização do conteúdo existente no banco de dados, permite analisar se o resultado obtido corresponde ao esperado. Assim como o *DBTools 2000*, o *SimAlg* segue a sintaxe e a simbologia utilizadas nos livros, facilitando a associação das consultas executadas no programa com o conteúdo de álgebra relacional aprendido. A fim de auxiliar também na aprendizagem de SQL, o *SimAlg* mostra qual é o SQL equivalente à consulta realizada em álgebra relacional. O tratamento de erros, juntamente com os tópicos de ajuda, constituem outra vantagem, pois guiam o usuário. No entanto, o *SimAlg* não permite a execução de consultas mais complexas como nas outras ferramentas, nem a criação e troca de banco de dados, possível no *LEAP*. A tabela 1 compara as três ferramentas levando em conta os pontos citados.

Tabela 1. Comparação entre ferramentas que simulam consultas em álgebra relacional.

	LEAP	DBTools 2000	SimAlg
Instalação e execução fáceis	×		×
Interface intuitiva		×	×
Liberdade para construção de consultas	×	×	
Sintaxe e simbologia tradicionais		×	×
Visualização do SQL equivalente		×	×
Visualização do conteúdo existente no BD			×
Tratamento de erros	×		×
Estabilidade	×		×
Criação de BD durante a execução	×		
Menu de ajuda dentro do programa	×		×

5. Considerações Finais

No decorrer do ano de 2011, a aplicação será testada pelos alunos de Fundamentos de Banco de Dados da UFSM para analisar principalmente a usabilidade. Testes com as operações suportadas e potenciais casos de erro já foram realizados pelas autoras e, pela análise destes e comparação entre as funcionalidades dos programas *LEAP*, *DBTools 2000* e *SimAlg*, concluiu-se que a ferramenta atingiu com sucesso seu principal objetivo de suprir as necessidades dos programas disponíveis existentes. Desta forma, espera-se a propagação de seu uso como ferramenta de ensino nos cursos de Banco de Dados.

Sugere-se, para trabalhos futuros, a implementação de uma interface gráfica que permita criação e alternância de banco de dados para a realização das consultas. No momento, estas operações apenas podem ser realizadas alterando o código da aplicação.

Referências

- HSQLDB 2.0 (2010). Disponível em: <http://hsqldb.org>. Acesso em: setembro de 2010.
- DBTools 2000 (2000). Disponível em: <http://www.cc.gatech.edu/computing/Database/dbTools>. Acesso em: agosto de 2010.
- LEAP (1998). Disponível em: <http://leap.sourceforge.net>. Acesso em: agosto de 2010.

Proposta de um WebSIG para Visualização de Informações sobre a Qualidade da Água Superficial de um Sistema de Informações Ambientais

Rejane M. Basso¹, Helena G. Ribeiro¹, Marcio Bigolin¹

¹Universidade de Caxias do Sul (UCS)

Rua Francisco Getúlio Vargas, 1130 - CEP 95070-560 – Caxias do Sul – RS - Brasil

{rmbasso, hgribeiro, marcio.bigolin}@ucs.br

Abstract. This article describes the initial process of developing the query module of a WebSIG application for disclosure information of an Environmental Information System. The prototype developed allows the display of the environmental monitoring points installed in Taquari-Antas Watershed through the web and in a georeferenced way.

Resumo. Este artigo descreve o processo inicial de desenvolvimento do módulo de consultas de uma aplicação WebSIG para divulgação de informações de um Sistema de Informação Ambiental. O protótipo WebSIG desenvolvido permite a exibição dos pontos de monitoramento ambiental instalados na Bacia Hidrográfica Taquari-Antas através da web e de forma georreferenciada.

1. Introdução

Com o crescente avanço da ocupação humana, houve um aumento dos problemas relacionados à degradação dos recursos naturais. Diante dessa situação, faz-se necessária a utilização de novas tecnologias, que permitam conhecer, gerenciar e controlar as mudanças que ocorrem no meio ambiente, buscando minimizar os impactos ambientais resultantes deste processo [MENDES and CIRILO 2001].

Os Sistemas de Informações Geográficas consistem em ferramentas computacionais que possibilitam a disponibilização de um conjunto de informações de forma georreferenciada em um mapa interativo, servindo como instrumento de gestão e suporte a tomada de decisões [SAUDE 2002] em problemas urbanos, rurais e ambientais [CAMARA and MEDEIROS 1998]. Eles permitem a realização de análises complexas integrando dados de diversas fontes em bancos de dados georreferenciados. Os SIGs que operam sobre a WWW são conhecidos por WebSIGs. A popularização dessa tecnologia é uma tendência mundial, e a crescente utilização da ferramenta deve-se a sua versatilidade, robustez e facilidade de utilização. Na crescente demanda por informações geográficas, os SIGs representam o avanço da tecnologia da informação e o advento da passagem da cartografia clássica para a digital permitindo uma disponibilização mais eficaz da informação espacial.

O Instituto de Saneamento Ambiental, da Universidade de Caxias do Sul, atua no projeto **Gestão dos Recursos Hídricos na Bacia Hidrográfica Taquari-Antas**

trecho Antas: disponibilidade, qualidade, usos e impactos decorrentes, cujo objetivo é avaliar os usos, a qualidade, a disponibilidade dos recursos hídricos na Bacia Hidrográfica Taquari-Antas - Trecho Antas, bem como os impactos ambientais decorrentes das atividades humanas nesta área [SANEAMENTO AMBIENTAL 2009].

O projeto compreende um sistema de informações, denominado Sistema de Informação Ambiental (SIA), que permitirá o cruzamento de grande quantidade de dados para a construção de cenários que servirão de subsídio ao entendimento da região que compreende a Bacia Hidrográfica Taquari-Antas. Os dados e informações georreferenciados gerenciados pelo sistema são oriundos de pesquisas e programas de monitoramento ambiental instalados na Bacia Hidrográfica, além de dados de diferentes instituições que também atuam na região.

O SIA contempla o estudo do uso de um WebSIG para relacionar as informações de monitoramento georreferenciadas através de um mapa interativo, com limitação de acesso definida por níveis de usuário. Atualmente existem sete tipos de usuários (Administrador, Hidrelétrica, Técnico, FEPAM, Especial, e Usuários Comuns). O protótipo desenvolvido nesse trabalho contempla os Usuários Comuns, que visualizam as informações de acesso ao público em geral.

A seguir, na seção 2, são apresentados os principais componentes e ferramentas utilizados no desenvolvimento do protótipo. A interface e principais recursos do protótipo são apresentados na seção 3. Na seção 4 tem-se as principais conclusões do trabalho realizado.

2. Desenvolvimento do WebSIG

De uma forma geral, o WebSIG deve atuar como um facilitador para visualização das informações monitoradas através de marcadores sobre um mapa, além de permitir a identificação dos limites da bacia hidrográfica, seus recursos hídricos, barramentos e os limites territoriais dos municípios que a compõe através de camadas vetoriais.

Entre as tecnologias para desenvolvimento de WebSIGs, destacam-se as Interfaces de Programação de Aplicativos (APIs) e os Servidores de Mapas. Neste trabalho utilizou-se a API do Google Maps V3, por ser bastante conhecida e de fácil utilização. A implementação do protótipo do WebSIG compreende:

- Página HTML: onde são organizados e apresentados os elementos que compõe a aplicação, como a lista de pontos e camadas, consulta e o mapa;
- Arquivo Javascript: implementa as funcionalidades de interação com o usuário, com a API do Google Maps e com o Servidor Web da aplicação.
- Arquivo PHP: realiza o acesso ao banco de dados para retornar as informações para a aplicação. Optou-se pelo PHP para manter as mesmas características de linguagens de programação do SIA e por ser uma linguagem amplamente utilizada e com suporte a inúmeros bancos de dados.

2.1. API do Google Maps

A API do Google Maps é uma interface de programação para acessar um serviço gratuito fornecido pela empresa Google para visualização e pesquisa de mapas. Ela é

uma biblioteca remota baseada em classes Javascript, que é uma linguagem que roda diretamente no navegador do usuário em uma página HTML.

Ao importar a API do Google Maps na aplicação, suas funcionalidades ficam disponíveis através de seus métodos, propriedades e eventos. As funcionalidades da API utilizadas no WebSig foram os marcadores (pontos) com balões informativos e as camadas vetoriais (linhas e polígonos).

2.2. Banco de Dados

O SIA utiliza um banco de dados PostgreSQL para armazenar as informações. O PostgreSQL possui um módulo específico para tratamento dos dados geográficos, o PostGIS. O PostGIS suporta diferentes tipos de dados geográficos entre eles: POINT (coordenadas que representam um ponto), LINESTRING (representam uma linha) e POLYGON (representam áreas). O banco de dados do SIA foi modelado utilizando o tipo de dado geográfico POINT, nele são armazenadas as coordenadas geográficas referentes à localização de cada ponto de monitoramento.

2.3. Camadas Vetoriais

As camadas vetoriais representam sobreposições que são carregadas sobre o mapa. Elas foram armazenadas em arquivos KML. As sobreposições utilizadas foram os polígonos e as linhas, sendo os polígonos para representar os limites das bacias e os limites territoriais dos municípios inseridos na bacia, e as linhas para representar os recursos hídricos que compõem a bacia.

Um arquivo KML contém um conjunto de coordenadas geográficas que compõe o elemento vetorial a ser sobreposto ao mapa. Criado pela Google, recentemente foi adotado como padrão internacional para compartilhamento de mapas.

3. Protótipo Desenvolvido

O protótipo consiste em uma aplicação de fácil utilização que serve de apoio à análise espacial das informações e como um banco de dados geográfico, com funções de recuperação de informação espacial.

O protótipo possibilita, além da visualização dos pontos de monitoramento, exibir/ocultar os limites da bacia, os limites municipais, os recursos hídricos e barramentos, auxiliando na análise das informações, além da possibilidade de realizar consultas por palavras específicas. Na figura 1 é possível visualizar o protótipo desenvolvido. Sobre o mapa são visualizados alguns pontos de monitoramento que podem ser detalhados clicando sobre o marcador no mapa.

4. Conclusões

O protótipo desenvolvido está disponível em <http://vbaco01.ucs.br>, nele qualquer usuário pode conhecer a aplicação, interagir e explorar seus diferentes recursos. O trabalho desenvolvido é o marco inicial de um WebSIG que deve crescer em funcionalidades na medida em que for sendo desenvolvido para o acesso dos demais níveis de usuários. Fazendo um comparativo com outros WebSIGs, como por exemplo o WebSIG do Instituto de Gestão das Águas e Clima (INGA), do Governo do Estado da

Bahia, que é um WebSIG bastante completo, composto de um conjunto de dados espaciais que envolvem a gestão das águas, hidrografia, dados de monitoramento de estações fluviométricas, pluviométricas e climatológicas, percebe-se que a ferramenta tem muito a evoluir pois há uma grande quantidade de recursos que podem ser explorados tornando a ferramenta mais completa.

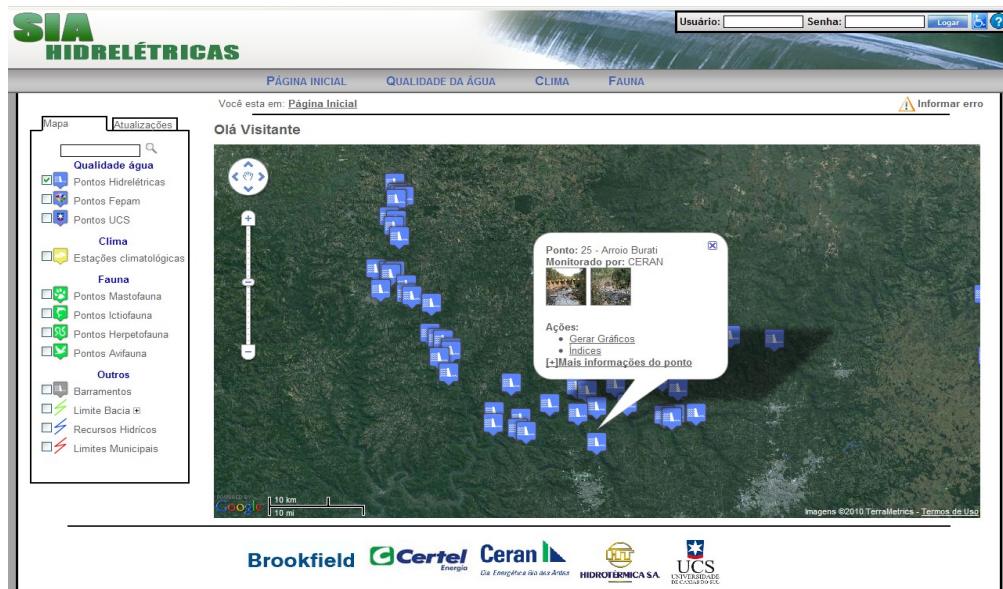


Figura 1. Protótipo desenvolvido.

Utilizar um banco de dados geográfico para armazenar as informações geográficas é um mecanismo eficiente pois permite trabalhar com a informação espacial junto com os atributos descritivos de um dado, ampliando as possibilidades de análise espacial da informação.

A partir deste trabalho percebeu-se a importância da criação e aperfeiçoamento de técnicas para divulgação das informações de forma a facilitar a compreensão do que está sendo representado. A utilização de um SIG como meio de divulgação de informações representa um avanço nesse sentido, pois ajuda a responder questões e a resolver problemas fornecendo-nos uma visão da informação que nos permite rapidamente compreendê-la.

Referências

- CAMARA, G.; MEDEIROS, J. S. de. (1998). Geoprocessamento para Projetos Ambientais. 2.ed. São José dos Campos: Instituto Nacional de Pesquisas Espaciais.
- MENDES, C. A. B.; CIRILO, J. A. (2001). Governabilidade dos Recursos Hídricos no Brasil: a implementação dos instrumentos de gestão na Bacia do Rio Paraíba. Porto Alegre: Associação Brasileira de Recursos Hídricos.
- SANEAMENTO AMBIENTAL, I. de. (2009). Sistemas de Informações para Programas de Monitoramento Ambiental - Hidroelétricas da Bacia Hidrográfica Taquari-Antas. Caxias do Sul: Universidade de Caxias do Sul.
- SAUDE, O. P. da. (2002). Sistemas de Informação Geográfica em Saúde: conceitos básicos. Brasília (DF): Organização Mundial da Saúde.

Reallog: Uma Ferramenta para Monitorar Fluxo Contínuo de Dados

Marcelo Preis Ferreira¹, Carla Diacui Medeiros Berkenbrock¹

¹Universidade do Estado de Santa Catarina - UDESC

Departamento de Ciência da Computação
Joinville, SC, Brasil

preis.udesc@gmail.com, diacui@joinville.udesc.br

Resumo. Um arquivo de log armazena mensagens sobre os eventos produzidos por sistemas computacionais. Esse armazenamento permite fazer um rastreamento dos eventos produzidos pelos sistemas. Geralmente, os arquivos de log são volumosos devido ao grande número de eventos que podem ocorrer nos sistemas. Este trabalho propõe uma ferramenta para monitorar a geração de log. Essa ferramenta, intitulada RealLog, poderá ser configurada através de métodos que colaboram na análise de arquivos de logs. O RealLog estará integrado a um sistema gerenciador de banco de dados com características especiais para tratar fluxo contínuos de dados.

1. Introdução

O aumento da capacidade de processamento dos computadores bem como das redes de computadores propiciaram uma evolução para sistemas distribuídos. Assim esses sistemas permitiram o acesso a qualquer hora e lugar. Isso obrigou os sistemas computacionais a serem equipados de mecanismos para tratar problemas de acessibilidade, segurança e disponibilidade.

Neste contexto, existem vários sistemas operando de forma remota, produzindo cada um seus arquivos de log. Assim, surgem problemas como analisar, armazenar, consultar de forma simples e em tempo real os arquivos produzidos por esses sistemas. Com isso, tem-se também um alto consumo de tempo para analisar arquivos de log com grandes volumes.

A utilidade dos arquivos de log é a de armazenar mensagens de eventos produzidos por sistemas computacionais, o que permite fazer um rastreamento dos eventos produzidos por um sistema computacional [Sah 2002]. Esse rastreamento permite aumentar a segurança, identificar perfis dos usuários dos sistemas, bem como promover ações que tornem os sistemas mais úteis no contexto onde estão inseridos [Girardin and Brodbeck 1998]. Porém, essas análises geralmente consomem recursos computacionais e humanos para fazer a interpretação e análise dos dados.

Elencando algumas características que estão associadas à análise de arquivos de log, podemos mencionar: (i) a demora em produzir os resultados das análises, (ii) o armazenamento que consome muitos recursos, (iii) a falta de formas eficientes de fazer consultas em arquivos que já possuem mensagens e continuam recebendo novas mensagens. Essas são características que podem ser associadas a fluxos contínuos de dados, onde os sistemas de fluxos dados se propõem a resolver vários desses problemas.

Este trabalho propõe uma ferramenta que poderá colaborar na resolução desses problemas gerando um ganho de produtividade em análise de arquivos de log. As mensagens em tais arquivos poderão ser vistas em tempo real através de tabelas, proporcionando a diminuição dos tempos de análise. O protótipo proposto será chamado de RealLog. Este protótipo possuirá características onde o usuário será capaz configurar e definir filtros através de uma interface. Assim, o usuário poderá configurar a ferramenta para realizar ações como descartar as mensagens que não são úteis ao contexto definido. O usuário também poderá extrair as informações produzidas por um determinado período e compará-las com as mensagens que estão chegando e sendo processadas pela ferramenta.

O artigo está organizado da seguinte forma. A Seção 2 fornece uma breve visão sobre sistemas gerenciadores de fluxos de dados. Na Seção 3 descrevem-se a ferramenta Reallog. Finalmente, na Seção 4, apresentam-se as considerações finais.

2. Sistemas Gerenciadores de Fluxos Dados

Fluxo contínuo de dados são mensagens ou sinais de dados que chegam continuamente a um determinado local para serem processadas ou armazenadas. Esses fluxos geralmente são produzidos através de sistemas computacionais que recebem requisições, processam e produzem os resultados para quem os solicitou.

Sistemas Gerenciadores de Fluxos de Dados (SGFD) se propõem a tratar fluxos contínuos de dados. Esses sistemas ajudam a fazer consultas, armazenar e extrair determinadas mensagens de fluxos contínuos e relacioná-los com os dados que estão chegando a todo instante.

Verificando que um arquivo de log recebe fluxo contínuo de dado, pode-se inserir um SGFD para fazer o controle desse fluxo. O fluxo produzido por eventos de sistemas chegarão o tempo todo para o SGFD processar.

Os SGFDs usam técnicas de consultas, exclusão e inserção dos dados semelhantes às usadas nos banco de dados tradicionais, porém os contextos que esses sistemas operam podem exigir um alto grau de desempenho, devido aos grandes volumes de dados que são produzidos pelos fluxos contínuos nos quais eles operam.

Os SGFDs oferecem recursos de processamento de fluxo e garantem resultados previsíveis e repetíveis. Isto leva a importância da perspectiva de tolerância a falhas e recuperação, como repetir e reprocessar os fluxos de entrada, produzindo o mesmo resultado, independentemente do tempo de execução [Stonebraker et al. 2005]. Essa característica representa a escalabilidade dos sistemas, pois pode-se alocar várias linhas de execução (*threads*) para processamento paralelo. Contudo, isto pode comprometer os recursos de hardware onde os sistemas operam, ao ponto que cause uma parada completa do funcionamento do sistema. Portanto, deve-se tomar ações que previnam os sistemas para que isso não venha a acontecer.

3. Projeto Reallog

O RealLog é uma ferramenta para a análise de arquivos de log. Ele tem como objetivo: (i) diminuir o tempo e custo necessário para fazer a análise de um arquivo desse tipo, (ii) promover formas eficientes de fazer consultas nesses arquivos, e (iii) fazer o armazenamento das mensagens recebidas por esses arquivos de forma eficiente. Esses objetivos definem

um caminho a ser percorrido, onde existem vários sistemas operando em um contexto. Esses sistemas recebem eventos que podem gerar mensagens de log, e essas mensagens são colocadas em um canal de comunicação. Como existem várias mensagens trafegando ao mesmo tempo sem um intervalo definido, é gerado um fluxo contínuo de dados no canal de comunicação, o que exige uma análise desse fluxo para que as informações sejam extraídas conforme determinadas pelas consultas e armazenadas em local para serem analisadas posteriormente.

As extrações de informações dos arquivos de log geram custos como consumo de tempo de uma pessoa e também um grande consumo de hardware para processar as consultas feitas nesses arquivos. Portanto, para diminuir essas demandas serão proposta formas de consultas dinâmicas em fluxos.

Nos requisitos funcionais da ferramenta tem-se que a ferramenta deve ler qualquer fluxo de dados, seja esse fluxo entrando em um arquivo, ou chegando através de uma determinada porta TCP/IP. O RealLog deve ser executado em uma Java Virtual Machine (JVM), para atender os requisitos de portabilidade da ferramenta. Para ler um determinado fluxo contínuo de dados, o usuário vai configurar qual arquivo, ou porta na rede deseja fazer a leitura.

A ferramenta lerá somente um fluxo por vez. Caso o usuário deseje mais do que um canal de fluxo, lido pela aplicação, ele terá que direcionar todos os fluxos para um mesmo canal, onde a aplicação esta fazendo a leitura do fluxo. Quando o RealLog ler o fluxo ele irá executar os filtros que estão armazenados no Bancos de Consultas (BC).

A armazenagem dos filtros no BC será feita pelo usuário. Quando o usuário tiver cadastrando as mensagens a ferramenta oferecerá as formas para caracterizar cada consulta. Depois de executado esses filtros, as mensagens que não passarem nesses filtros, serão eliminadas pela aplicação SGFD.

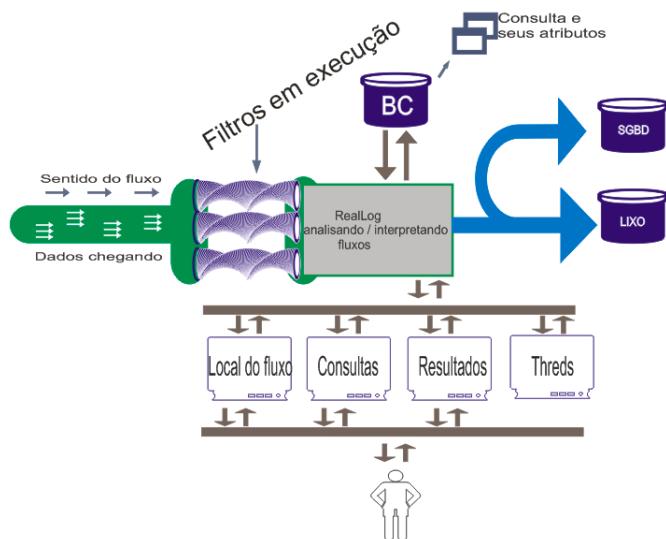


Figura 1. Ferramenta RealLog processando fluxos e interagindo com usuário

Na Figura 1 é apresentada uma descrição das funcionalidades da ferramenta, supondo que uma aplicação esteja sendo executada. BC corresponde a região onde estão armazenadas as consultas. Essas consultas equivalem aos filtros que podem estar sendo

executados na chegada do fluxo. A execução de cada filtro depende da programação feita pelo usuário de quando será executado determinado filtro na linha do tempo. A execução na linha do tempo será definida pelos atributos contidos em cada consulta. O tempo de execução dos filtros é definido pelos atributos nas consultas inseridas pelo usuário. Os dados que passarem no filtro serão armazenados no SGBD, os demais serão descartados. Todas as consultas serão cadastradas pelo usuário. O papel principal do usuário será cadastrar as consultas, definir o fluxo que será analisado, e visualizar os resultados. Os sentidos das flechas entrando e saindo entre os componentes significa que o usuário ou a ferramenta está inserindo, alterando ou excluindo aquele item.

O armazenamento dos dados se dará integrando a ferramenta a um SGBD. A ferramenta também usa um sistema embutido em seu funcionamento para tratar o fluxo contínuo de dados. Integrando esses itens, a ferramenta deverá propiciar uma forma dinâmica de analisar um número grande de tipos de arquivos de log, onde os usuários configurarão quantos contextos e situações desejarem, somente caracterizando suas consultas.

4. Considerações Finais

O RealLog é uma ferramenta que está sendo desenvolvida para análise de arquivos de log. Os logs poderão ser lidos de arquivos ou direto de uma porta TCP/IP. Com a implantação de uma ferramenta com as características do RealLog, o usuário terá flexibilidade para analisar seus arquivos de log, podendo diminuir os custos em análise, visto que esse processo pode ser aplicado a uma grande variedade de arquivos, moldados através de consultas.

O RealLog está sendo projetado para possibilitar a configuração do arquivo a ser analisado, permitindo que se escolha os dados a serem extraídos desses arquivos de modo a facilitar sua análise. Contudo, a configuração e criação das consultas para a ferramenta é um pouco oneroso para o usuário, visto que ele precisa ter noções sobre SQL, análise de arquivos de log, bem como conhecer o contexto em que a ferramenta irá atuar.

O RealLog está sendo desenvolvido em Java para aproveitar a portabilidade oferecida pela JVM, e integrando um SGFD e um SGBD para facilitar sua implantação. As questões relacionadas à segurança não serão tratadas. Ainda, serão consideradas operações realizadas localmente. Contudo, no futuro esses itens podem ser mudados por meio de ajustes na ferramenta.

Referências

- Girardin, L. and Brodbeck, D. (1998). A visual approach for monitoring logs. In *Proceedings of the 12th USENIX conference on System administration*, pages 299–308. USENIX Association.
- Sah, A. (2002). A new architecture for managing enterprise log data. In *USENIX Systems Administration (LISA XVI) Conference Proceedings*, pages 121–132.
- Stonebraker, M., Çetintemel, U., and Zdonik, S. (2005). The 8 requirements of real-time stream processing. *ACM SIGMOD Record*, 34(4):42–47.

Projeto e Implementação da Interface com Usuário do Sistema *BInXS*

Douglas T. Machado, Fabíola W. Jaeger, Ronaldo dos Santos Mello

Departamento de Informática e Estatística – Centro Tecnológico – Universidade Federal
de Santa Catarina (UFSC)
Caixa Postal 476 – 88.040-900 – Florianópolis – SC – Brasil.
{goga, binha, ronaldo}@inf.ufsc.br

Resumo. Este artigo descreve o desenvolvimento da interface gráfica com o usuário para um sistema de integração de esquemas XML denominado *BInXS*. Este sistema tem como peculiaridade a abstração conceitual de esquemas XML e a resolução de conflitos de integração inerentes à representação de dados XML em nível conceitual, visando a geração de um esquema conceitual global. O processo de integração é semi-automático, exigindo a intervenção de um usuário especialista em diversas tarefas. Assim sendo, uma interface com usuário bem projetada se faz necessária para viabilizar o processo. O projeto da interface está concluído e a implementação está em finalização.

1. Introdução

A interoperabilidade de dados utilizando XML é uma realidade dada a popularidade da representação e troca de informações neste formato entre pessoas e sistemas. O sistema *BInXS* (*Bottom-up Integration of XML Schemata*) é uma proposta de solução para esta problemática, sendo responsável pela integração semântica de esquemas XML de fontes de dados heterogêneas em um mesmo domínio e geração de um esquema global que serve de base para a realização de consultas unificadas, sem a necessidade de se conhecer a origem e a estruturação específica de cada fonte [Mello and Heuser 2005]. Seu diferencial em relação a trabalhos relacionados é a abstração de cada esquema XML em um esquema conceitual e a resolução de conflitos de integração inerentes à representação semi-estruturada de dados XML em nível conceitual.

Por se tratar de um processo de integração de dados em nível conceitual, intervenção de um usuário especialista é necessária para determinar a semântica adequada na conversão de elementos e atributos XML em classes ou relacionamentos, bem como na resolução de conflitos e ajustes no esquema global. Este artigo apresenta uma estratégia de projeto de interface com usuário adotada para o *BInXS* [Jaeger 2007] e a implementação do primeiro protótipo desta interface [Machado 2011]. As Seções seguintes descrevem as tarefas do sistema que exigem interação com o usuário, o desenvolvimento da interface gráfica e as considerações finais.

2. Interação com o Usuário no *BInXS*

O sistema *BinXS* integra esquemas XML heterogêneos segundo uma abordagem *bottom-up* que segue duas etapas. Na primeira etapa, chamada *Conversão*, o esquema XML de cada fonte é convertido em um esquema conceitual que melhor representa a

intenção semântica dos dados. Na segunda etapa, chamada *Integração Semântica*, um conjunto de esquemas conceituais locais são unificados, definindo um esquema conceitual global passível de consulta. Bases de dados terminológicas apóiam a descoberta de conceitos semanticamente similares [Mello and Heuser 2005].

Intervenção com um usuário especialista é esperada em ambas as etapas. Na *Conversão*, antes do mapeamento para um esquema conceitual, pode-se remover elementos e atributos semanticamente irrelevantes, como *tags* que mantêm agrupamentos de outros elementos, bem como definir novos nomes semanticamente mais relevantes para os mesmos. Durante o mapeamento, quando um elemento composto X possui elementos componentes Y que se repetem, o usuário deve indicar se existe um (1) tipo de relacionamento com cardinalidade “n” entre os conceitos que representam X e Y, ou se existem vários relacionamentos com semânticas diferentes entre eles. Ainda, quando existe recursividade entre elementos, deve-se definir o papel do auto-relacionamento conceitual que os representa. Após o mapeamento, o usuário valida o esquema conceitual preliminar gerado, podendo alterar, por exemplo, nomes de conceitos, bem como nomes, cardinalidades e tipos de relacionamentos.

Na etapa de *Integração Semântica*, o usuário inicialmente confirma se a similaridade semântica entre conceitos locais, determinada automaticamente, é válida. A partir destas similaridades, um esquema conceitual global preliminar é gerado, sendo permitido ao usuário definir novos relacionamentos de especialização relevantes entre conceitos provenientes de esquemas locais distintos e ainda renomear conceitos, relacionamentos e definir restrições de exclusão mútua entre relacionamentos.

3. Desenvolvimento da Interface Gráfica do BInXS

A interação do usuário com o sistema BinXS caracteriza-se pelo controle da execução de um processo sequencial que realiza várias tarefas dentro das suas duas etapas. Algumas abordagens para projeto de interface com o usuário centradas na modelagem da interação para execução de tarefas foram analisadas [Wood, 1997; Larry and Lockwood, 1999]. A abordagem *The Bridge* [Wood, 1997] foi escolhida pela sua maior eficiência/simplicidade na definição do projeto, realizando menos transformações entre modelagens dos requisitos do sistema para modelagens dos objetos de interface.

A abordagem *The Bridge* basicamente: (i) define *fluxos de tarefas* para cada necessidade de interação do usuário, sendo cada fluxo composto por uma sequência de ações e um resultado; (ii) identifica *classes de objetos* (entidades do sistema) necessárias à execução da tarefa; e (iii) associa estes objetos com *objetos gráficos da interface* com o usuário. Por exemplo, para a etapa de *Conversão* projetou-se um fluxo de tarefa denominado “*Remoção de componentes irrelevantes do esquema XML*”. Este fluxo (a) analisa elementos/atributos do esquema XML; (b) identifica aqueles que são relevantes e solicita a remoção; (c) o sistema verifica se as remoções são possíveis e as executa; (d) como resultado, tem-se o esquema XML modificado. Identificou-se, neste caso, que esta tarefa manipula uma classe “*Esquema XML*”, com atributos que descrevem seus elementos/atributos, e que possui métodos de remoção de elementos e de atributos. Por fim, conforme ilustrado na Figura 1 a seguir, um objeto desta classe foi representado graficamente como uma tabela aninhada que mostra a hierarquia de elementos do esquema XML, sendo a tarefa executada através de seleção direta de elementos e

atributos pelo usuário. A abordagem *The Bridge* possui uma notação gráfica para o descrição de suas três fases, que não é detalhada neste artigo por limitações de espaço.

Pelo fato do BInXS aplicar um processo sequencial com tarefas bem definidas, optou-se ainda por um estilo *Tutorial* ou *Wizard* de interação, onde o usuário executa uma sub-etapa de cada etapa do processo por vez, com a previsão de mensagens de erro e tarefas adicionais, dependendo da ação tomada. A Figura 1 mostra o primeiro protótipo da interface gráfica do BInXS onde este estilo de projeto é adotado.

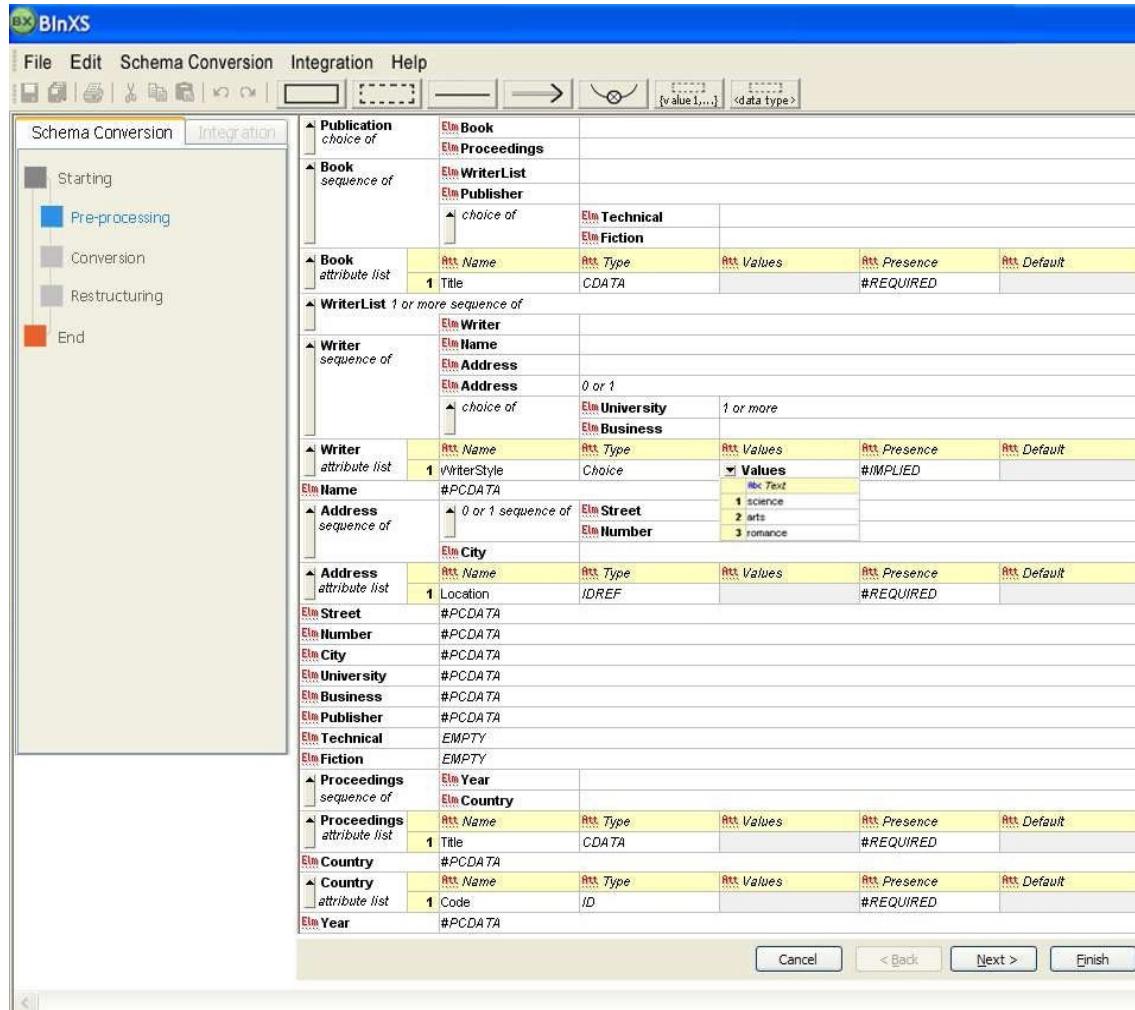


Figura 1. Interface com o Usuário do Sistema BInXS – Exemplo 1.

A Figura 1 indica que o usuário encontra-se na sub-etapa de *Pré-processamento* da etapa de *Conversão* do BInXS, que ocorre antes do mapeamento do esquema XML para um esquema conceitual. Esta etapa exibe o esquema XML, sendo permitido ao usuário alterá-lo, conforme descrito na Seção 2. O estilo *Tutorial* da interface permite ao usuário avançar, retroceder, cancelar ou encerrar cada tarefa (botões no rodapé direito), indicando, além disso, a etapa/sub-etapa em que o usuário está (abas à esquerda).

A Figura 2 mostra outra tela de interação do usuário com a interface do BInXS para a sub-etapa de *Reestruturação* da etapa de *Integração Semântica*. Nesta tela é possível perceber que a interface projetada possui uma área de visualização/edição de esquemas. Neste exemplo é mostrado um esquema conceitual global, formado por conceitos (retângulos) e relacionamentos de associação (setas). O usuário pode alterar

ou incluir conceitos e relacionamentos se valendo de alguns ícones presentes na parte superior da interface que representam componentes do esquema conceitual.

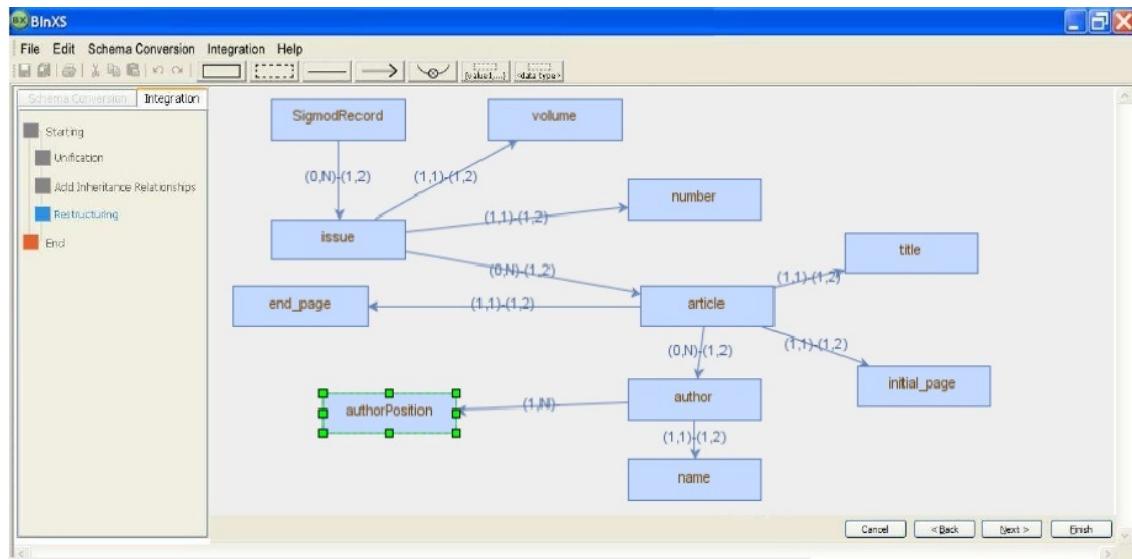


Figura 2. Interface com o Usuário do Sistema BInXS – Exemplo 2.

4. Considerações Finais

Este artigo relata sucintamente o projeto e a implementação da interface com o usuário para um sistema de integração de esquemas XML. Dado o grande número de tarefas semi-automáticas neste sistema, o desenvolvimento da interface exigiu esforços de pesquisa e desenvolvimento para permitir uma interação adequada e fácil com o usuário durante processos de integração. Diversas propostas de sistemas similares existem na literatura, porém elas não descrevem a utilização de técnicas de projeto de interface com o usuário [Jaeger 2007]. O projeto da interface está concluído. A implementação está sendo desenvolvida em *Java*, utilizando o pacote gráfico *JGraph*, encontrando-se em fase de finalização. Atividades futuras incluem testes e eventuais aprimoramentos do interface para que o sistema possa se tornar operacional.

References

- Jaeger, F. W. (2007) “Projeto de uma Interface Gráfica para o Ambiente BInXS”, Trabalho de Conclusão de Curso, Departamento de Informática e Estatística, Universidade Federal de Santa Catarina.
- Larry, C. and Lockwood, L. A. D. (1999), Software for Use: A Practical Guide to the Models and Methods of Usage, Addison Wesley.
- Machado, D. T. (2011) “Desenvolvimento de uma Interface com o Usuário para o Sistema de Integração de Esquemas XML BInXS”, Trabalho de Conclusão de Curso, Departamento de Informática e Estatística, Universidade Federal de Santa Catarina.
- Mello, R. S. and Heuser, C. A. (2005) “BInXS: A Process for Integration of XML Schemata. In: Conference On Advanced Information Systems Engineering (CAISE), 17th edition, LNCS 3520, p. 151-166.
- Wood, L. E. (1997), User Interface Design: Bridging the Gap from User Requirements to Design, Taylor & Francis, Inc.

Aplicando tuning no PostgreSQL

Marcelo Josué Telles¹, Renata Galante¹, Carina F. Dorneles²

¹Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – CEP 91.501-970 – Porto Alegre, RS - Brazil

²Dpto. de Informática e Estatística – Universidade Federal de Santa Catarina (UFSC)
Caixa Postal 476 – CEP 88049-900 – Florianópolis, SC - Brazil

marcelojtelles@gmail.com, galante@inf.ufrgs.br, dorneles@inf.ufsc.br

Abstract. This paper presents initials results about experiments that analyze the performance of the PostgreSQL DBMS. The experiments has been conducted over two different installations: the standard one and an installation using tuning techniques. Operations such as UPDATE, INSERT, DELETE and SELECT have been executed in order to simulate their use in both environments.

Resumo. Este artigo apresenta resultados iniciais de um experimento realizado sobre o SGBD Postgres. Os experimentos foram realizados em dois ambientes: instalação padrão do SGBD e outra usando técnicas de tuning. Foram realizadas operações de inserção, alteração, deleção e seleção a fim de simular seu comportamento, em termos de performance, nos dois ambientes instalados.

1. Introdução

Os SGBD têm por princípio armazenar e disponibilizar informações de forma conveniente e eficiente [Silberschatz et al. 2006]. Geralmente estas informações são acessadas por um número significativo de usuários. Tais acessos geram processamento ocasionando consumo de recursos de *hardware* e aumento no tempo de resposta. O conceito de *tuning* em SGBD, está relacionado à realização de uma ou mais tarefas no menor tempo possível sem que isso tenha consequências na estabilidade do serviço do Banco de Dados [PostgreSQL 2009].

Nos SGBDs os problemas relacionados a velocidade (relacionado ao tempo de realização das operações) se dividem em [Silberschatz et al. 2006]: (i) problemas no Sistema Operacional (sistema de arquivos, serviços concorrentes etc.); (ii) problemas no *hardware* (barramento incompatível, recursos mal dimensionados e ou compartilhados etc.); (iii) problemas na elaboração das consultas, tais como ausência de limites inferiores e superiores (limitar a quantidade de resultados) nas consultas, ausência de índices, junções desnecessárias etc.); (iv) problemas nos ajustes e configuração do SGBD.

Neste trabalho, são investigados problemas nos ajustes e configuração do SGBD, tais como: ajustes na quantidade de memória utilizada pelos *buffers* de transações, intervalos de acesso ao disco, quantidade de cache e *logs* etc. Como objeto de análise para este artigo serão utilizadas tabelas e operações(*insert*, *delete*, *update* e *select*) simulando uma necessidade do mundo real. O SGBD escolhido foi o PostgreSQL versão 8.3 32bit.

Este artigo está organizado como segue. A Seção 2 apresenta trabalhos relacionados, a Seção 3 descreve a abordagem proposta, ambiente, *hardware* e *software* utilizados,

a Seção 4 descreve os experimentos realizados e a comparação dos resultados. Finalmente, a Seção 5 apresenta trabalhos futuros.

2. Trabalhos Relacionados

O gerenciamento de *buffers* permite melhorar o desempenho de tarefas típicas, como [Milanés et al. 2004] consultas e atualizações. O redimensionamento dos recursos de *hardware* para o SGBD é feito pelos DBAs e exige amplo conhecimento sobre o SGBD e *hardware*. O trabalho de [Carneiro et al. 2007] argumenta que o redimensionamento dos recursos a serem configurados, para determinados usos do banco de dados, deve ser avaliado. Alguns trabalhos buscam obter uma visão global das tarefas realizadas pelos SGBDs, para identificar quais as exigências de processamento e outros recursos de *hardware* [Milanés and Lifschitz 2005] devem ser alocadas a cada momento. O objetivo do trabalho descrito neste artigo é identificar estes ajustes, colaborando no desenvolvimento de algoritmos estatísticos para definição da alocação dos recursos de *hardware*. Tal prática irá colaborar para implementação de *tuning* automático ou *self-tuning*, como propõe [Milanés et al. 2004]. Desta forma este trabalho oferece suporte para a definição de quais recursos merecem atenção na implementação de *self-tuning*.

Alguns SGBDs comerciais estão dando atenção para a ideia de *tuning* automático, como no caso do **Oracle 10g** e **Oracle 11g** [Dias et al. 2005]. O **Oracle 11g** realiza análises no uso do SGBD em produção para identificar a carga de trabalho e assim redimensionar convenientemente os recursos disponíveis [Athreya 2007].

3. Configuração do Ambiente e Método de pesquisa

O sistema operacional utilizado foi o Debian versão 2.6.26 i386 de 32bit [Debian 2009], pois este SO apresenta características de servidores. A configuração de *hardware* adotado: Processador Core 2 Quad 2.4GHz e memória de 2GB DDR2 1066MHz, caracterizando um *hardware* fácil aquisição para eventuais novos testes. O esquema criado no SGBD consiste em uma tabela simples, mas muito comum em sistemas que utilizam SGBD, uma chave primária, um texto, um número e uma data com hora. Este esquema recebeu 20.000 registros. Os experimentos foram realizados com o uso de instruções SQL, para cada uma delas, foi registrado o tempo necessário para sua realização. Em seguida o “tune” foi aplicado e experimentos realizados novamente. Por fim os tempos foram comparados.

Instalação usando “tuning”

As seguintes configurações foram definidas no arquivo “postgresql.conf”, para ajustar o funcionamento do PostgreSQL:

- max_connections = 300: indica o número de conexões aceitas pelo servidor. Este parâmetro foi definido em 300, para simular uma situação real mais exigente; por padrão, max_connections = 100.
- shared_buffers = 512MB: define a quantidade de memória que o SGBD irá utilizar. As alterações aqui devem obedecer os limites impostos pelo SO. O mínimo recomendado é de 128kB ou 16kB por conexão, definido no parâmetro “max_connections”.
- temp_buffers = 64MB: indica o número máximo de *buffers* temporários utilizados por cada sessão de banco de dados. O padrão é de 8MB.

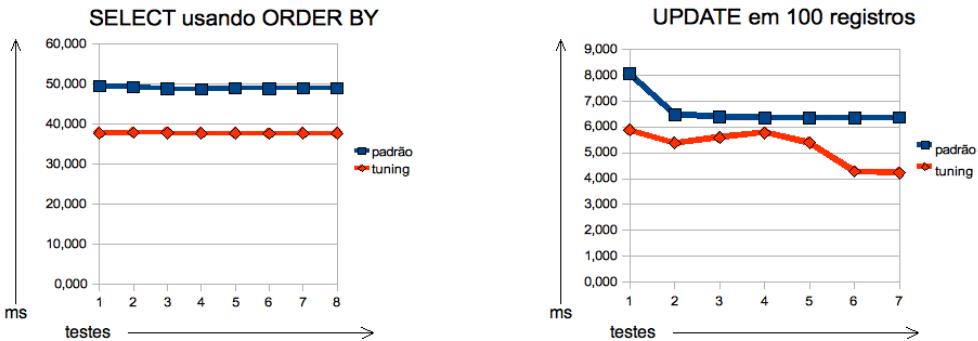


Figura 1. Primeira análise entre instalação padrão versus instalação com “tune” e “update” em 100 registros

- work_mem = 4MB: indica quantidade de memória por operações internas de classificação antes de alternar para arquivos temporários em disco. O valor padrão é de 1MB.
- maintenance_work_mem = 512MB: especifica a quantidade máxima de memória a ser utilizada em operações de manutenção, tais como vacuo, criação de índice, alteração de estruturas de tabelas e adição de chaves estrangeiras.

4. Experimento Realizado

O experimento simula uma situação do mundo real e compara as reduções de tempo para a realização das operações de: *insert*, *delete*, *update* e *select* na instalação padrão e na instalação com “tune”. As configurações feitas, alocaram recursos de *hardware* para o banco de dados, em especial *work_mem*, definindo que o acesso ao disco só será feito após a operação exigir 4MB ou mais e *shared_buffers* que indica a quantidade de memória que o SGBD irá utilizar. Foram feitas modificações no SO, possibilitando que mais de 512MB fosse utilizado, no entanto o SO ficou instável. No primeiro experimento foi feita a seguinte consulta:

`EXPLAIN ANALYSE SELECT * FROM template ORDER BY numero`

A consulta foi realizada sete vezes utilizando a instalação padrão, a média (soma dos sete tempos, dividido por sete) de tempo foi de 48,955ms. Em seguida o arquivo “*postgresql.conf*” foi configurado conforme descrito na Seção 3.1. A média de tempo na instalação utilizando técnicas de “tune” foi de 37,642ms. A Figura 1 mostra o gráfico da operação de *select* envolvendo os 20.000 registros e o gráfico da análise do desempenho para operação de *update* envolvendo 100 registros. A Figura 2 mostra o gráfico da análise do desempenho para a operação de *insert* de apenas 1 registro e *delete* envolvendo 100 registros.

O *script* de criação da tabela e inserção dos 20.000 registros necessitou de 12,40 segundos para ser realizado na instalação padrão, enquanto que na instalação “tune” o tempo necessário foi de 11,70 segundos, representando 5,65 % de redução no tempo para realização do *script*. Os cálculos dos percentuais tomaram como base o tempo necessário para realização do *script* na instalação padrão.

Este artigo apresentou os benefícios que podem ser obtidos ao adotar técnicas de *tuning*. As modificações no arquivo “*postgresql.conf*” permitem diminuição signifi-

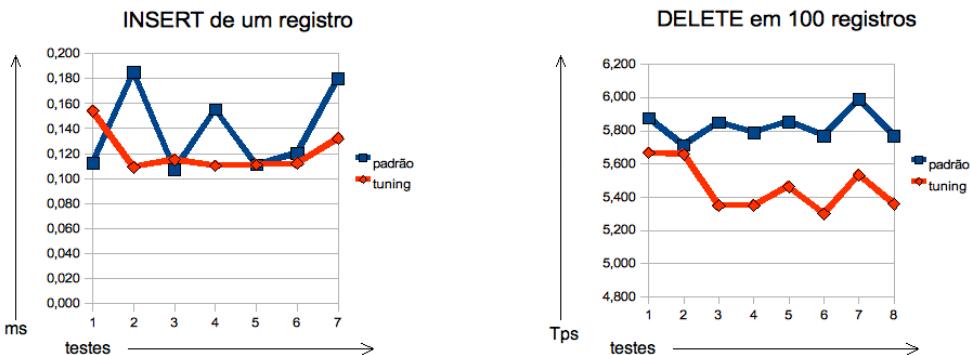


Figura 2. “insert” de um registro e “delete” de 100 registros

cativa nos tempos de processamento consumidos pelas operações no SGBD. Os ganhos mais significativos foram nas operações de busca com ordenação, em alguns experimentos chegando a 28 %, sendo que a média de ganho nos experimentos, foi de 24 %. Nas operações de inserção o tempo sofreu muitas oscilações, em alguns experimentos houve ganhos de 41 %. Na exclusão o ganho chegou a 10 % e alteração os ganhos chegaram a 33 %.

5. Trabalhos Futuros

Para futuros trabalhos: (i) realizar experimentos em um *hardware* com mais de 2GB de memória RAM, usando sistema operacional e SGBD de 64bit, desta forma podendo ampliar os valores de work_mem e shared_buffers; (ii) implementação de “tuning” dinâmico pois as operações variam, realizando gravação/leitura em disco e memória e realizando processamento comparando e ordenando valores.

Referências

- Athreya, J. R. (2007). Banco de dados oracle 11g: Visão geral do real application testing e da capacidade de gerenciamento. *White paper Oracle*, pages 07–11.
- Carneiro, A. P., Moreira, J. L., and de Freitas, A. L. C. (2007). Tuning - técnicas de otimização de banco de dados um estudo comparativo: Mysql e postgresql. *Monografia de conclusão de curso*.
- Debian (2009). <http://www.debian.org>. Acessado em novembro.
- Dias, K., M, Ramacher, Shaft, U., Venkataramani, V., and Wood, G. (2005). Performance diagnosis and tuning in oracle. In *Online Procs Biennial Conf. on Innovative Data Systems Research (CIDR)*, pages 84–94.
- Milanés, A. Y. and Lifschitz, S. (2005). Design and implementation of a global self-tuning architecture. *Brazilian Symposium on Database*.
- Milanés, A. Y., Lifschitz, S., and Salles, M. A. V. (2004). Estado da arte em auto-sintonia de sistema de bancos de dados relacionais. *Monografia de conclusão de curso*.
- PostgreSQL (2009). <http://www.postgresql.org.br/sobre>. Acessado em outubro.
- Silberschatz, A., Korth, H. F., and Sudarshan, S. (2006). *Database System Concepts*. McGraw-Hill.

Um Panorama do Uso do Tempo em Motores de Busca Web*

Edimar Manica^{1,3}, Carina F. Dorneles², Renata Galante¹

¹Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brasil

²Departamento de Informática e Estatística - Universidade Federal de Santa Catarina (UFSC)
Campus Universitário Trindade - 88049-900 - Florianópolis - SC - Brasil

³Campus Avançado Ibirubá - Instituto Federal do Rio Grande do Sul (IFRS)
Rua Nelsi Ribas Fritsch, nº 1111, Bairro Esperança – 98200-000 – Ibirubá – RS – Brasil

{emanica,galante}@inf.ufrgs.br, dorneles@inf.ufsc.br

Resumo. A Web recebe todos os dias um turbilhão de páginas. Com isso, ela acaba se tornando um repositório de informações temporais. Diante dessa fonte de informação, os usuários podem estar interessados em informações de um período de tempo específico. Ao perceber o valor do tempo para a recuperação de informação, os pesquisadores começaram a incorporar a dimensão temporal aos motores de busca. Este trabalho apresenta a evolução dos motores de busca com relação à exploração da informação temporal, bem como aponta as direções de pesquisa e perspectivas futuras, sob o ponto de vista dos autores.

Abstract. The web can be considered a huge repository of temporal information, once daily receives a huge amount of new pages. Generally, users are interested in information related to a specific temporal period. Considering the information retrieval area, researches have preliminarily incorporated the temporal dimension to the search engines. This paper presents a study that describes the evolution of search engines on the exploitation of temporal information. Research directions and future perspectives are also presented, considering the authors' point of view.

1. Introdução

As páginas Web descrevem inúmeros tópicos, tais como conferências, esportes, política e entretenimento. Dentro desses tópicos, encontram-se inúmeros eventos que se repetem ao longo do tempo. O SBD ocorre todo ano. A Copa do Mundo ocorre a cada 4 anos. As eleições no Brasil ocorrem a cada 2 anos¹. Capítulos de novelas são exibidos quase todo dia. Com isso, o processamento das consultas sobre esses eventos deve levar em conta o intervalo de tempo desejado pelo usuário.

Muitas pessoas que realizam buscas Web estão interessadas em informações atuais. Por exemplo, Quem foi eleito presidente do Brasil na última eleição? No entanto, a cada dia páginas e mais páginas são postadas na Web. Muitas dessas páginas permanecem disponíveis formando um conjunto de informações

*Este trabalho é parcialmente financiado pelo INCT (processo 573871/2008-6).

¹Os mandatos são de 4 anos, porém há um período de 2 anos entre as eleições estaduais e federais e as eleições municipais.

históricas. Diante dessa fonte de informação, os usuários podem estar interessados em informações passadas. Por exemplo, Quais os artigos publicados no SBBD em 2009?, Qual seleção venceu a Copa do Mundo em 2002? e Quem era o presidente em 1998?. Além disso, os usuários podem estar interessados em informações futuras. Por exemplo, Qual a previsão do tempo para a próxima segunda-feira? e O que acontecerá na novela no capítulo da próxima terça-feira?

O tempo pode ajudar na reconstrução de um período histórico particular ou descrever o contexto de um documento ou coleção. Com isso, o tempo pode ser útil para propostas de ranqueamento de resultados por relevância [Alonso et al. 2007]. Com a percepção do valor da informação temporal para a recuperação de informação, os motores de busca começaram a fazer uso dessa dimensão. A primeira iniciativa foi ordenar os resultados considerando o tempo, de modo que os resultados mais recentes fossem colocados no topo. Esses motores de busca são referenciados neste trabalho como motores de busca temporais de **Primeira Geração**. Após, foram disponibilizadas formas de filtrar os resultados em intervalos de tempo de acordo com a data de coleta das páginas. Esses motores de busca são referenciados neste trabalho como motores de busca temporais de **Segunda Geração**. Finalmente, trabalhos recentes propõem a exploração do tempo presente no conteúdo das páginas Web e nas palavras-chave da consulta. Estes motores de busca são referenciados neste trabalho como motores de busca temporais de **Terceira Geração**.

O objetivo deste trabalho é apresentar um panorama da utilização do tempo em motores de busca Web, apresentando a evolução de seu uso no passado e presente, bem como descrevendo as prováveis direções para o futuro. A principal contribuição deste trabalho é apresentar a evolução da utilização da informação temporal em motores de buscas Web, apontando lacunas ainda presentes neste cenário, bem como direções de pesquisa e perspectivas futuras, sob o ponto de vista dos autores.

Este artigo está organizado como segue. A Seção 2 discute os motores de busca de primeira geração. Na Seção 3 são apresentados os motores de busca de segunda geração. A Seção 4 discute os motores de busca de terceira geração. A Seção 5 discute as direções futuras. Finalmente, na Seção 6 são apresentadas as considerações finais.

2. Primeira Geração

Os motores de busca de primeira geração utilizam o tempo para ordenar os resultados, colocando mais ao topo as páginas mais recentes. A informação temporal utilizada é a data de coleta da página ou a data de criação/última alteração da página. Os motores de busca tradicionais utilizam essa estratégia de ordenação, além de outras métricas, como, por exemplo, a popularidade.

A estratégia da primeira geração assume que as páginas que foram postadas mais recentemente são as mais relevantes para o usuário. Essa assertiva é válida tipicamente para notícias, onde o usuário geralmente deseja encontrar as novidades e páginas mais antigas podem já ter sido lidas pelo usuário. No entanto, essa estratégia não beneficia os usuários interessados em informações sobre determinada ocorrência de um evento. Por exemplo, considere as páginas Web X e Y. A página X possui informações sobre a última eleição e foi criada e coletada no dia 10/01/2011. A página Y possui uma análise da eleição de 2006 e foi criada e coletada no dia 11/01/2011. A consulta “eleição”, com

o objetivo de retornar informações sobre a última eleição, submetida a um motor de busca da primeira geração retorna a página Y mais ao topo que a página X². No entanto, a página X é mais relevante para a consulta do que a página Y.

3. Segunda Geração

Os motores de busca de segunda geração exploram o tempo de coleta das páginas Web, ou seja, permitem a realização de filtros temporais sobre a data em que a página foi inserida na base de dados do motor de busca. Google³, Chronica [Efendioglu et al. 2006] e InfoSeek⁴ são exemplos de motores de busca dessa geração.

Extrair a data de coleta das páginas Web é uma tarefa trivial, basta ao armazenar a página na base de dados obter o *timestamp* atual. No entanto, geralmente há uma lacuna entre a data de coleta da página e a data a que a informação nela contida refere-se. Esta lacuna ocorre em três situações principais: (i) quando a informação contida na página refere-se a data de postagem e a coleta da página ocorre em dias posteriores; (ii) quando a página contém informações históricas; e (iii) quando a página contém previsões futuras.

Considere uma página X postada em 07 de março de 2010 com informações sobre os jogos realizados nesse dia. A página X possui poucos *links* importantes que apontam para ela. Com isso, essa página pode ser coletada dias após a data de postagem. Por exemplo, a página pode ser coletada no dia 10 de março de 2010. Neste caso, “10 de março de 2010” é a informação temporal associada a página X pelos motores de busca de segunda geração, embora essa não seja a informação temporal real da página.

Considere uma página Y postada em 14 de dezembro de 2010 com informações sobre a Copa do Mundo de 2002. A página Y possui vários *links* importantes que apontam para ela. Com isso, a página Y foi coletada no mesmo dia da postagem. A informação temporal associada com a página será “14 de dezembro de 2010”. No entanto, a página contém informações sobre um evento que ocorreu em 2002.

Considere uma página Z postada em 20 de maio de 2010 com informações sobre a previsão do tempo para o dia 25 de maio de 2010. A página Z possui vários *links* importantes que apontam para ela. Com isso, a página Z foi coletada no mesmo dia da postagem. A informação temporal associada com a página será “20 de maio de 2010”. No entanto, a página contém informações referentes ao dia 25 de maio de 2010.

4. Terceria Geração

Os motores de busca temporais de terceira geração exploram o tempo presente no conteúdo dos documentos Web para melhorar a qualidade dos resultados das buscas. As informações temporais presentes no conteúdo de documentos texto, tais como: um ponto no tempo, um evento ou um período de tempo, são descritas em um nível conceitual por uma **entidade temporal**. Uma sequência de *tokens* que representa uma instância de uma entidade temporal é denominada **expressão temporal**. Existem três categorias de expressões temporais [Alonso et al. 2007]:

²Desconsiderando as demais métricas utilizadas pelos motores de busca.

³<http://www.google.com/>

⁴<http://www.infoseek.co.jp/>

- **Explícitas** - expressões temporais que descrevem diretamente uma entrada em uma linha de tempo, tal como uma data exata ou ano específico. Por exemplo, a expressão “dezembro de 2004” ou “12 de janeiro de 2006” em um fragmento de texto são expressões temporais explícitas e podem ser mapeadas diretamente para pontos em uma linha de tempo.
- **Implícitas** - são expressões temporais que precisam de conhecimento predefinido (ontologias de tempo, por exemplo) para serem mapeadas para uma entrada em uma linha de tempo. Nome de feriados e eventos específicos são típicos exemplos de expressões temporais implícitas. Por exemplo, a expressão “Natal de 2005” precisa ser mapeada para “25 de dezembro de 2005”.
- **Relativas** - são expressões temporais que representam entidades temporais que apenas podem ser mapeadas para uma entrada em uma linha de tempo em referência à uma expressão temporal explícita, implícita ou ainda ao momento em que o texto foi escrito. Por exemplo, a expressão “ontem” só pode ser mapeada com base no momento em que o texto foi escrito.

O processo de mapear as expressões temporais para um formato único, representando de forma padronizada um ponto na linha de tempo, é denominado **normalização de expressões temporais**. Os motores de busca temporais indexam as expressões temporais normalizadas para permitir um acesso rápido e consistente dos dados temporais.

Os motores de busca temporais de terceira geração possuem dois grandes desafios: extraír as expressões temporais e definir como a informação temporal será utilizada pelo motor de busca. Portanto, antes de apresentar os motores de busca propriamente ditos, serão descritas algumas ferramentas de extração de expressões temporais, pois são peças-chave para esses motores de busca. A Subseção 4.1 apresenta as principais ferramentas de extração de expressões temporais. A Subseção 4.2 apresenta os principais motores de busca temporais de terceira geração.

4.1. Ferramentas de Extração de Expressões Temporais

As ferramentas atuais de extração de expressões temporais em documentos Web usam técnicas de extração de entidades nomeadas para identificar as expressões temporais. Alguns trabalhos propõem o uso de um documento XML para armazenar essas expressões anotadas [Alonso et al. 2007]. TimeML [TimeML 2010] é um padrão emergente para anotação de expressões temporais e eventos. No entanto, existem várias ferramentas de anotação de expressões temporais que definem sua própria forma de anotação. A seguir são apresentadas as principais ferramentas de extração de expressões temporais encontradas na literatura pesquisada, bem como um breve comparativo entre elas.

ANNIE [ANNIE 2010] é uma ferramenta de código aberto para extração de informação que faz parte do *framework* GATE [Cunningham et al. 2002]. Além de informação temporal, ANNIE extrai informações de localização, pessoas, organizações, esportes, etc. A extração é feita a partir de entidades nomeadas. Algumas entidades pré-definidas estão disponíveis e, se necessário, é possível definir novas entidades através da linguagem de regras que vem embutida na ferramenta. Como resultado, tem-se um arquivo XML com as anotações das entidades identificadas em uma linguagem específica da ferramenta. A manipulação desse XML pode ser realizada através de uma API própria, que é oferecida junto com o *framework*. ANNIE anota expressões temporais explícitas, implícitas e relativas. Porém, não realiza a normalização dessas expressões temporais.

GUTime [GUTime 2010] é uma ferramenta de código aberto especificada para anotação temporal. Junto com essa ferramenta é necessário instalar o TreeTagger [TreeTagger 2010]. O TreeTagger é um POS (*Part-of-Speech*) tagger que rotula as palavras de um texto com suas funções morfossintáticas, como verbo e substantivo. GUTime utiliza esses rótulos nas suas regras de inferência de expressões temporais. Como resultado, tem-se um documento XML com as expressões temporais anotadas de acordo com a linguagem TimeML. Essas anotações podem ser manipuladas por uma linguagem de processamento de consulta XML como XQuery. GUTime anota e normaliza expressões temporais explícitas, implícitas e relativas. Para normalizar expressões relativas, tais como: hoje, amanhã, ontem, próximo mês e ano passado, GUTime verifica o contexto local a fim de identificar o ponto de referência temporal no qual esses tempos são relativos. Normalmente, o ponto de referência temporal é a data de publicação do documento.

PorTexto [Craveiro et al. 2009] é uma ferramenta de reconhecimento de entidades temporais em textos de língua portuguesa. O processamento dos documentos é efetuado frase a frase. A identificação das expressões temporais é feita usando padrões de expressões, criados a partir de co-ocorrências existentes em referências temporais. A determinação das co-ocorrências é realizada utilizando um conjunto de palavras temporais de referência (PTR). A lista das PTR é criada manualmente e deve ser constituída por todas as palavras temporais que aparecem em expressões com, no mínimo, duas palavras, para que realmente haja a necessidade de determinar as palavras que ocorrem conjuntamente. Os meses do ano e dias da semana são exemplos de PTR, pois ocorrem em expressões como “em janeiro” e “na próxima segunda-feira”. Os padrões extraídos são definidos por expressões regulares e armazenados em um arquivo. É possível alterar os padrões existentes e até mesmo incluir novos padrões. Como resultado, tem-se um documento XML com as anotações temporais que seguem as diretrivas gerais e de tempo do HAREM [Santos et al. 2008]. O HAREM é uma avaliação conjunta na área do reconhecimento de entidades nomeadas em português. PorTexto anota expressões temporais explícitas, implícitas e relativas. Porém, não as normaliza. Além disso, não está disponível para utilização.

Chronos [Negri e Marseglia 2004] é uma ferramenta projetada para realizar o reconhecimento e normalização de expressões temporais. O processamento do texto em Chronos envolve a extração de *tokens*, um processamento linguístico e o reconhecimento de multi-palavras baseado em uma lista de 5.000 entradas recuperadas da WordNet⁵. Em seguida, o texto é processado por um conjunto de aproximadamente 1.000 regras básicas que reconhecem construções temporais e extrai informações sobre elas que são úteis para o processo de normalização. Em seguida, são executadas regras de composição que resolvem ambiguidades quando múltiplos rótulos são possíveis. Como resultado, tem-se um documento XML com as anotações temporais na linguagem TIMEX2⁶ [Ferro et al. 2001]. Chronos anota expressões temporais explícitas e relativas.

A Tabela 1 apresenta um resumo comparativo das ferramentas de anotação temporal discutidas nessa seção. Nessa tabela são considerados os seguintes itens de comparação:

- POS - indica se as ferramentas utilizam processamento linguístico;

⁵<http://wordnet.princeton.edu/>

⁶TimeML é uma extensão de TIMEX2.

- **Linguagem** - indica qual linguagem é utilizada para anotação das expressões temporais;
- **Disponível** - indica se as ferramentas estão disponíveis para utilização;
- **Tipos** - indica quais os tipos de expressões temporais (explícitas (E), implícitas (I) e relativas (R)) são tratados pelas ferramentas;
- **Normalização** - indica se as ferramentas normalizam as expressões temporais;
- **Isolamento** - indica se cada expressão é avaliada isoladamente.

Tabela 1. Comparativo entre as ferramentas de anotação de expressões temporais

	ANNIE	GUTime	PorTexto	Chronos
POS	Sim	Sim	Não	Sim
Linguagem	Própria	TimeML	Diretivas HAREM	TIMEX2
Disponível	Sim	Sim	Não	Não
Tipos	E, I, R	E, I, R	E, I, R	E, R
Normalização	Não	Sim	Não	Sim
Isolamento	Sim	Sim	Sim	Sim

Legenda: E: explícita I: implícita R: relativa

Constata-se que a maioria das ferramentas utilizam processamento linguístico. Essa característica aumenta o custo de processamento. Cada ferramenta utiliza uma linguagem de anotação das expressões temporais diferente, embora todas as linguagens sejam baseadas em XML. Apenas GUTime e ANNIE estão disponíveis para utilização. A maioria das ferramentas tratam expressões temporais explícitas, implícitas e relativas. Apenas GUTime e Chronos realizam a normalização das expressões temporais. Todas as ferramentas avaliam cada expressão temporal de forma isolada.

Estão disponíveis na Web os mais diversos tipos de documentos, tais como páginas Web, documentos XML, documentos PDF, etc. As técnicas de extração de informações temporais podem não ser adequadas para todos os tipos de documentos. Por exemplo, as ferramentas apresentadas não são adequadas para documentos XML orientados a dados. Documentos XML orientados a dados são documentos XML estruturados, geralmente oriundos de bases relacionais, onde tipicamente o nome dos nodos representa anotação semântica. Esses documentos não contêm frases compostas por vários elementos morfossintáticos. Geralmente, tem-se nodos apenas com substantivos. Com isso, as ferramentas que utilizam processamento linguístico são prejudicadas. Além disso, o fato dessas ferramentas avalarem cada expressão temporal de forma isolada gera o descarte de informações importantes para o processo de normalização das expressões temporais. Agrupar as expressões de acordo com o caminho que as contém e analisar em conjunto todas as expressões de um mesmo caminho fornece maiores informações para o processo de normalização. Por exemplo, no documento XML apresentado na Figura 1, o formato da expressão temporal presente na linha 04 é ambíguo, pois não é possível descobrir se a expressão temporal refere-se a 12 de março de 2010 ou a 03 de dezembro de 2010. No entanto, ao verificar o outro valor do mesmo caminho XML (`pessoas/pessoa/nascimento`) é possível descobrir que o formato do caminho é dia/mês/ano.

TPI [Manica et al. 2010] é um motor de busca temporal de terceira geração específico para documentos XML orientados a dados. Esse motor de busca agrupa as expressões

```

01. <pessoas>
02.   <pessoa>
03.     <nome>XYZ</nome>
04.     <nascimento>12/03/2010</nascimento>
05.   </pessoa>
06.   <pessoa>
07.     <nome>TZY</nome>
08.     <nascimento>25/03/2010</nascimento>
09.   </pessoa>
10. </pessoas>

```

Figura 1. Exemplo de documento XML contendo dados temporais

de acordo com o caminho que as contém para obter informações para o processo de normalização. Além disso, esse motor de busca possui heurísticas para identificação de datas estruturadas em vários elementos e intervalos temporais.

4.2. Motores de Busca

A seguir são apresentados os principais motores de busca de terceira geração encontrados na literatura pesquisada, bem como um breve comparativo entre eles.

TISE [Jin et al. 2008] indexa uma expressão temporal por página, selecionando a expressão temporal que descreve mais apropriadamente os eventos da página Web. O predicado temporal é especificado na forma de um intervalo, que deve ser definido em um campo diferente do campo utilizado para as demais palavras-chave da consulta. O predicado temporal da consulta é aplicado na expressão temporal indexada para a página. A informação temporal é representada como um intervalo.

TERN [Vicente-Díez e Martínez 2009] indexa todas as expressões temporais de cada página. O predicado temporal é expresso no mesmo campo onde são expressas as demais palavras-chave da consulta. O predicado temporal da consulta é aplicado em qualquer expressão temporal indexada para a página. A informação temporal é representada como um instante.

Pasca [Pasca 2008] propõem um motor de busca temporal utilizado quando o usuário deseja descobrir quando determinado evento ocorreu. Com isso, o usuário não informa um predicado temporal e sim recebe um valor temporal como resultado. Esse motor de busca indexa uma expressão temporal para cada *nugget* temporal, formando um pseudo-documento. Um *nugget* temporal é um fragmento de uma sentença que informa fatos de domínio aberto associados com alguma entidade. A informação temporal é representada como um instante.

A Tabela 2 apresenta um resumo comparativo entre os motores de busca temporais discutidos nessa seção. Nessa tabela são considerados os seguintes itens de comparação:

- **Predicado** - verifica se o predicado temporal é embutido na consulta ou se é informado em um campo separado;
- **Rótulo** - indica se a informação temporal indexada é representada como um instante ou como um intervalo;
- **Índice Temporal** - indica qual informação temporal é indexada;

- Tipo de Consulta - indica qual o tipo de consulta pode ser especificada:
 - (i) Seleção Temporal, consultas nas quais utiliza-se um predicado temporal para filtrar a consulta e (ii) Saída Temporal, consultas onde o usuário está interessado em saber qual o tempo em que um determinado evento ocorreu.

Tabela 2. Comparativo entre motores de busca temporais de terceira geração

	Predicado	Rótulo	Índice Temporal	Tipo de Consulta
TISE	Separado	Intervalo	1 timex por página	Seleção Temporal
TERN	Embutido	Instante	Todas as timex para página	Seleção Temporal
Pasca	NA	Instante	1 timex por <i>nugget</i> temporal	Saída Temporal

Legenda: NA: Não se aplica Timex: Expressão temporal

Constata-se que TERN é o único motor de busca que permite que o predicado temporal seja inserido no mesmo campo que as demais palavras-chave da consulta. Essa característica exige uma etapa de identificação e normalização das expressões temporais presentes na consulta. No entanto, simplifica e agiliza a criação da consulta, uma vez que não é necessário preencher vários campos. TISE é o único motor de busca que representa a informação temporal indexada como um intervalo. A representação como intervalo é mais abrangente uma vez que permite, por exemplo, para a informação “presidiu o senado entre 2001 e 2003”, expressar que em 2002 a informação também era válida. Cada motor de busca indexa uma informação temporal diferente. Indexar apenas uma expressão temporal para a página (TISE), ocasiona o descarte de várias informações temporais relevantes. Indexar cada *nugget* temporal associado a uma expressão temporal (Pasca), ao invés de indexar todas as expressões temporais para a página (TERN), tem a vantagem de associar a uma expressão temporal apenas os termos que estão próximos no conteúdo da página. A maioria dos motores de busca permitem consultas de seleção temporal. TISE, TERN e Pasca utilizam técnicas das ferramentas de extração de expressões temporais para identificar e normalizar as expressões temporais a fim de indexar os dados temporais de forma consistente e padronizada.

5. Direções Futuras

Há vários desafios e direções para pesquisas futuras com relação à utilização do tempo em motores de busca temporais:

1. **Peso da Informação Temporal** - a informação temporal é uma característica importante a ser considerada no ranqueamento das páginas Web. Porém, existem outras métricas a serem utilizadas, como, por exemplo, a popularidade. A questão de pesquisa futura é qual o peso da informação temporal com relação às demais métricas. Além disso, é necessário considerar a proximidade temporal. Por exemplo, em uma consulta onde o usuário deseja informações dos últimos 3 dias, uma informação de 4 dias atrás pode ser relevante, pois em muitas consultas Web os usuários não sabem exatamente o que estão procurando e em outras não sabem expressar corretamente a sua necessidade;
2. **Normalização de Expressões Temporais** - embora muitos trabalhos realizem a identificação, extração e normalização de expressões temporais, ainda existem lacunas, tais como desambiguação de formatos e identificação de períodos temporais. Considere a expressão temporal “12/03/2010” contida em uma página

Web. Essa expressão temporal possui formato ambíguo, uma vez que seu formato pode ser dia/mês/ano ou mês/dia/ano. Como descobrir o formato correto? Considere a frase “Em 2004, Ciclano tomou posse como diretor da empresa XYZ, dirigindo-a até 2007”. Essa frase possui um período temporal que inicia em 2004 e termina em 2007. Com isso, é necessário ter o conhecimento que em 2005 Ciclano era o diretor da empresa XYZ. Como identificar períodos temporais em páginas Web?;

3. **Consulta Temporal Embutida** - a maioria dos motores de busca temporais possui um campo adicional onde o usuário especifica o período de tempo desejado. No entanto, quando a informação temporal é adicionada no mesmo campo que as demais palavras-chave da consulta surgem novos desafios, como, por exemplo, a manipulação de operadores temporais. Os operadores temporais definem relações temporais, tais como depois e antes. Esses operadores podem estar explícitos. Por exemplo, a consulta “presidente depois 2000”. Esses operadores, também, podem estar implícitos. Por exemplo, a consulta “regime militar 64 84”. Nessa consulta, a relação temporal pode ser igualdade, ou seja, o usuário deseja encontrar páginas sobre os regimes militares que iniciaram em 1964 e terminaram em 1984. Por outro lado, a relação temporal pode ser intersecção, ou seja, o usuário deseja encontrar páginas sobre qualquer regime militar que ocorreu entre 1964 e 1984. A identificação de operadores temporais implícitos exige a análise da semântica e do contexto da consulta;
4. **Versões** - devido a característica dinâmica da Web, várias páginas são modificadas constantemente. Com isso, uma consulta temporal pode ter como objetivo recuperar uma versão passada de uma página;
5. **Índice** - o índice dos motores de busca temporais devem considerar a dimensão temporal. Um motor de busca tradicional utiliza um índice invertido para armazenar os termos. Usualmente, é utilizada uma lista de *postings* (identificadores de documentos e escores pré-computados) por termo. Essa técnica é escalável, o que permite sua utilização pelos motores de busca Web. Entretanto, inserindo a dimensão temporal, o índice invertido conteria os *postings* para todas as versões dos documentos em todo o período de tempo da coleção. Consequentemente, pesquisas por instantes ou intervalos curtos são penalizadas por esta explosão no espaço da versão. Esse problema é agravado em consultas multi-palavra. A menos que seja restringida a flexibilidade da consulta em multi-palavras de interesse pré-identificadas, necessita-se de um índice posicional. Nesse índice, cada lista por palavra contém *postings* para cada ocorrência da palavra em cada documento. Agregar este tipo de índice com a dimensão temporal de forma otimizada é um desafio [Weikum et al. 2011].

6. Considerações Finais

O tempo é uma dimensão importante para qualquer aplicação. Percebendo o valor da informação temporal para a recuperação de informação, os pesquisadores começaram a incorporar essa dimensão nos motores de busca Web para aprimorar seus mecanismos de ranqueamento. A primeira iniciativa, e ainda utilizada atualmente, foi colocar as páginas mais recentes no topo dos resultados. Em seguida, surgiram as opções de filtrar os resultados, considerando intervalos de tempo de acordo com a data de coleta das páginas. Finalmente, foram propostos motores de busca que exploram a informação temporal

presente no conteúdo das páginas Web. Este artigo mostrou que o tratamento do tempo não está resolvido. Ao contrário, com a Web, os desafios de detectar diversos formatos de tempo, considerar o peso da informação temporal e definir uma boa estratégia de indexação são pontos cruciais a serem considerados pelos motores de busca.

Referências

- Alonso, O.; Gertz, M. e Baeza-Yates, R. A. (2007). On the value of temporal information in information retrieval. *SIGIR Forum*, 41(2):35–41.
- ANNIE (2010). Open source information extraction. Disponível em: <<http://www.aktors.org/technologies/annie/>>. Acesso em: 11 nov. 2010.
- Craveiro, O.; Macedo, J. e Madeira, H. (2009). Use of co-occurrences for temporal expressions annotation. In *SPIRE*, volume 5721 of *LNCS*, pages 156–164. Springer.
- Cunningham, H.; Maynard, D.; Bontcheva, K. e Tablan, V. (2002). A framework and graphical development environment for robust nlp tools and applications. In *ACL*, pages 168–175.
- Efendioglu, D.; Faschetti, C. e Parr, T. J. (2006). Chronica: a temporal web search engine. In *ICWE*, pages 119–120. ACM.
- Ferro, L.; Mani, I.; Sundheim, B. e Wilson, G. (2001). Tides temporal annotation guidelines - version 1.0.2. MITRE Technical Report MTR 01W0000041. McLean, Virginia: The MITRE Corporation. June 2001.
- GUTime (2010). Adding timex3 tags. Disponível em: <<http://www.timeml.org/site/tarsqi/modules/gutime/index.html>>. Acesso em: 12 jun. 2010.
- Jin, P.; Lian, J.; Zhao, X. e Wan, S. (2008). Tise: A temporal search engine for web contents. In *IIITA '08*, pages 220–224, Washington, USA. IEEE Computer Society.
- Manica, E.; Dorneles, C. F. e Galante, R. (2010). Supporting temporal queries on xml keyword search engines. *JIDM*, 1(3):471–486.
- Negri, M. e Marseglia, L. (2004). Recognition and normalization of time expressions: Itc-irst at tern 2004. Technical report, ITC-irst, Trento.
- Pasca, M. (2008). Towards temporal web search. In *SAC*, pages 1117–1121. ACM.
- Santos, D.; Freitas, C.; Oliveira, H. G. e Carvalho, P. (2008). Second harem: New challenges and old wisdom. In *PROPOR*, volume 5190 of *LNCS*, pages 212–215. Springer.
- TimeML (2010). Markup language for temporal and event expessions. Disponível em: <<http://www.timeml.org>>. Acesso em: 10 nov. 2010.
- TreeTagger (2010). A language independent part-of-speech tagger. Disponível em: <<http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/>>. Acesso em: 10 jun. 2010.
- Vicente-Díez, M. T. e Martínez, P. (2009). Temporal semantics extraction for improving web search. In *DEXA Workshops*, pages 69–73. IEEE Computer Society.
- Weikum, G. et al. (2011). Longitudinal analytics on web archive data: Its about time! In *5th Biennial Conference on Innovative Data Systems Research (CIDR2011)*.

Gerência de Informações de Mapeamento no Sistema de Integração de Esquemas XML *BInXS*

Eduardo Vaz de M. T. Bozzi, Ronaldo dos Santos Mello

Departamento de Informática e Estatística – Centro Tecnológico – Universidade Federal de Santa Catarina (UFSC)
Caixa Postal 476 – 88.040-900 – Florianópolis – SC – Brasil.

{vaz, ronaldo}@inf.ufsc.br

Abstract. *XML is a standard format for representation and exchange of data, specially on the Web. Given its semistructured and flexible nature for organizing the same data, XML allows the definition of several heterogeneous schemata for different data sources in a same domain. Such a situation makes complex the definition of a single query over these data sources. The BInXS system is a solution to this problem, generating a global conceptual schema based on a set of local XML schemata in a same domain. This paper presents a system module that manages mappings between components of the conceptual schema to each local XML schema, allowing the easy translation of global queries. This module is a contribution to BInXS development.*

Resumo. *XML é um formato popular para a representação e intercâmbio de dados, principalmente na Web. Dada a sua natureza semi-estruturada e flexível para organizar um mesmo dado, XML permite a definição de vários esquemas de dados para diferentes fontes de dados em um mesmo domínio. Esta situação torna complexa a formulação de uma única consulta sobre estas fontes de dados. O sistema BInXS é uma solução para este problema, gerando um esquema conceitual global a partir de um conjunto de esquemas XML locais em um mesmo domínio. Este artigo apresenta um módulo para a gerência dos mapeamentos de componentes do esquema conceitual para cada esquema XML local, facilitando a tradução de uma consulta em nível global. Este módulo é uma contribuição para o desenvolvimento do BInXS.*

1. Introdução

XML é um formato popular para representação de dados, principalmente no contexto da Web, assim como para a interoperabilidade de dados e informações entre pessoas e sistemas de aplicação ou de gerência de dados. O sistema *BInXS* (*Bottom-up Integration of XML Schemata*) é uma proposta de solução para esta problemática, sendo responsável pela integração semântica de esquemas XML de fontes de dados heterogêneas em um mesmo domínio e pela geração de um esquema global que serve de base para a realização de consultas unificadas, sem a necessidade de se conhecer a origem e a estruturação específica de cada fonte [Mello 2002]. A contribuição do BInXS em relação a trabalhos relacionados (citados na Seção 6) é a abstração de cada esquema semi-estruturado XML em um esquema conceitual e a resolução de conflitos de integração inerentes à representação de dados XML em nível conceitual.

O processo de integração realizado pelo BInXS ocorre em duas etapas, conforme descreve a Seção 2. Durante a primeira etapa, esquemas XML de fontes locais são convertidos em esquemas conceituais, visando facilitar a compreensão do usuário que acessa um esquema de dados para fins de consulta, bem como facilitar uma integração de esquemas em nível semântico. Esta conversão respeita regras de mapeamento que definem como um elemento, atributo ou relacionamento hierárquico entre elementos no nível XML é traduzido para uma entidade ou relacionamento entre entidades no nível conceitual. Desta forma, uma consulta formulada no nível conceitual pode ser traduzida para consultas locais respeitando a estrutura XML. Por exemplo, uma entidade chamada *Autor* no esquema conceitual pode se referir ao elemento *Escritor* em uma fonte de dados XML e ao atributo *Criador* em outra fonte.

Este trabalho apresenta o módulo do sistema BInXS responsável pela geração e catalogação de informações de mapeamento. Ele define expressões de caminho *XPath* [XPath 2011] que descrevem o mapeamento de cada entidade e de cada relacionamento no esquema conceitual e registra estas informações de mapeamento na definição do esquema conceitual, que é mantida em documentos OWL [OWL 2011]. Este módulo controla também o cadastro e posterior acesso a catálogos relacionais particulares do BInXS responsáveis pelo registro de transformações ocorridas durante o pré-processamento de esquemas XML. Este pré-processamento, que ocorre antes da conversão XML→conceitual, realiza ações como a remoção de elementos e atributos semanticamente irrelevantes - um elemento cuja finalidade é apenas encapsular um conjunto de outros elementos (*lista de autores* de um *livro*, por exemplo) -, bem como a renomeação de elementos ou atributos, visando dar nomes semanticamente mais relevantes para os mesmos. A catalogação destas atualizações é necessária para a posterior geração de expressões de mapeamento válidas.

Este artigo possui mais seis Seções. A Seção 2 detalha o processo realizado pelo BInXS. A Seção 3 apresenta a estratégia de gerência de mapeamentos e a Seção 4 detalha o módulo respectivo. A Seção 5 descreve sucintamente um estudo de caso e as Seções 6 e 7 são dedicadas a trabalhos relacionados e conclusão, respectivamente.

2. BInXS

O sistema BinXS integra esquemas XML heterogêneos segundo um processo *bottom-up*, conforme mostra a Figura 1 (a) [Mello 2002]. O esquema XML de cada fonte é convertido em um esquema conceitual que melhor representa a intenção semântica dos dados (etapa de *Conversão*) e, na sequência, um conjunto de esquemas conceituais locais são unificados, definindo um esquema conceitual global passível de consulta (etapa de *Integração Semântica*). Esta segunda etapa conta com o suporte de bases de dados terminológicas que apóiam a descoberta de conceitos semanticamente similares [Silva and Mello 2006]. A etapa de *Conversão* considera ainda a análise de instâncias de dados XML (documentos XML) para determinar, com melhor acurácia, certos aspectos da modelagem conceitual, como por exemplo, cardinalidades de relacionamentos e tipos de dados de elementos e atributos. O processo adotado pelo BInXS é *semi-automático*, pois a intervenção de um usuário especialista se faz necessária durante a execução de certas tarefas [Mello 2002].

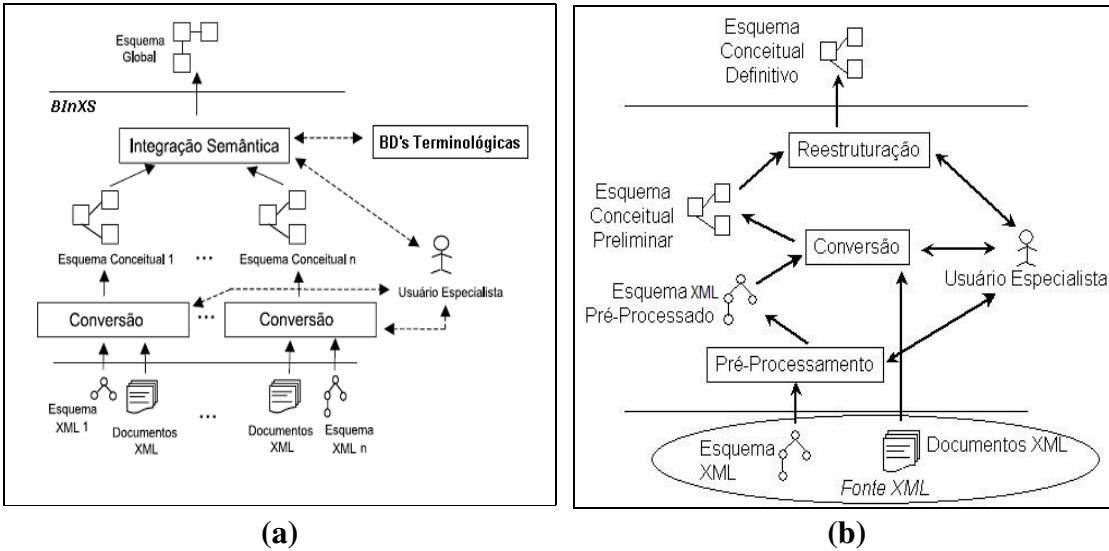


Figura 1. (a) Processo seguido pelo BInXS; (b) Sub-etapas da Etapa de Conversão.

O foco deste trabalho está na etapa de *Conversão*, pois é nela que ocorre a definição de informações de mapeamento. A Figura 1 (b) apresenta as suas sub-etapas. A sub-etapa de *Pré-processamento* recebe como entrada um esquema XML descrito em *XML Schema* (XSD) [XSD 2011] (*esquema XML original*). Ela refina o esquema XML original com o objetivo de preservar no esquema XML somente dados relevantes para a abstração em nível conceitual. Exemplos de tarefas realizadas por esta sub-etapa são: substituição da definição de tipos abstratos na definição dos elementos/atributos que os referenciam; remoção de elementos componentes e atributos semanticamente irrelevantes; e alteração de nomes de elementos/atributos.

Na sub-etapa de *Conversão*, o *esquema XML pré-processado* é transformado, através de regras de conversão pré-definidas e eventual auxílio de um usuário especialista para a definição da intenção semântica dos dados XML, em um *esquema conceitual preliminar*¹. Durante essas conversões são geradas informações de mapeamento para cada definição de entidade ou de relacionamento. Por fim, a sub-etapa de *Reestruturação* registra o *esquema conceitual definitivo* e as informações de mapeamento em um documento OWL (*Ontology Web Language*), que é o resultado da etapa de *Conversão* do BInXS. OWL é uma linguagem baseada em XML para a descrição de esquemas conceituais e ontologias. O esquema definitivo pode ser idêntico ao esquema preliminar ou ser o resultado de validações do usuário em relacionamentos com o objetivo de melhor expressar a intenção semântica do esquema. Um exemplo é a transformação de um relacionamento de associação entre entidades *Publicação* e *Livro* em especialização, pois este melhor expressa a semântica do relacionamento.

2.1 Catálogos de Apoio ao Mapeamento

Três catálogos relacionais de apoio à geração de informações de mapeamento são populados na sub-etapa de *Pré-Processamento*: (i) catálogo de renomeações; (ii) catálogo de elementos removidos e; (iii) catálogo de elementos virtuais. O *catálogo de*

¹ Detalhes sobre o modelo conceitual e regras estão na Seção 3 e em [Mello 2002], respectivamente.

renomeações registra alterações realizadas pelo usuário na nomenclatura de elementos e atributos. Este catálogo é composto de uma tabela com as colunas *Fonte*, *Nome* e *Nome Original*. A coluna *Fonte* contém a localização e o nome do esquema XML através de uma URL ou caminho na unidade de armazenamento local. *Nome* registra o novo nome do elemento/atributo e *Nome Original* mantém o nome antigo.

O *catálogo de elementos removidos* mantém elementos que não tem relevância semântica e foram removidos do esquema XML original. Uma tabela com quatro colunas cataloga os elementos removidos: *Fonte*, *Elemento_Removido*, *Elemento_Pai* e *Elemento_Filho*. A coluna *Fonte* tem função idêntica a do catálogo anterior. A coluna *Elemento_Removido* armazena o nome do elemento que foi removido do esquema. Os campos *Elemento_Pai* e *Elemento_Filho* mantêm o nome dos elementos pai e filho, respectivamente, do elemento removido.

O *catálogo de elementos virtuais* armazena os elementos que representam estruturas aninhadas anônimas internas a um determinado elemento *Ex* do esquema XML. Por exemplo, a definição de um elemento *Pessoa* pode conter uma escolha dentre duas estruturas alternativas (*Universidade* e *Empresa*), tendo, cada uma delas, elementos componentes distintos, supondo que elas representem duas formas diferentes de filiação de uma pessoa. Essas estruturas, se puderem se repetir dentro de *Ex* – por exemplo, uma pessoa que possui várias Universidades como filiação –, são complexas de serem representadas no nível conceitual, pois exigiria uma restrição do tipo “todos os elementos componentes da estrutura interna devem ocorrer *N* vezes juntos”.

A solução empregada para resolver esse problema é a definição de um *elemento virtual* *Ev*, que encapsula esse conjunto de elementos componentes de uma estrutura aninhada. Desta forma, a modelagem conceitual é facilitada, gerando um relacionamento com cardinalidade *N* entre *Ex* e *Ev*. A tabela gerada para o armazenamento desse tipo de elemento tem a seguinte estrutura: *Fonte* e *Nome* do elemento virtual definido nesta fonte de dados.

3. Estratégia de Mapeamento de Esquemas XML-Conceitual

A conversão de um esquema XML em um esquema conceitual requer que correspondências entre componentes do esquema conceitual e do esquema XML sejam mantidas para que consultas formuladas sobre um esquema conceitual possam alcançar os dados XML respectivos na(s) fonte(s) de dados que respeita(m) o(s) esquema(s) XML. Para tanto, uma estratégia de mapeamento XML-conceitual se faz necessária.

O modelo adotado pelo BInXS para a representação de esquemas conceituais é o modelo ORM (*Object with Roles Model*) [Halpin 1998]². Ele foi escolhido devido a sua flexibilidade para a modelagem da associação entre propriedades simples e estruturadas, comum em esquemas XML na associação de elementos simples/atributos e elementos compostos. A Figura 2 mostra uma modelagem ORM.

Entidades conceituais em ORM são chamadas conceitos não-léxicos e conceitos léxicos. Um *conceito não-léxico* modela um fato que é composto por outros conceitos, sendo representado por uma elipse contínua. *Livro* é um exemplo de conceito não-

² Vale observar que OWL é o modelo de armazenamento de esquemas conceituais ORM. Maiores detalhes encontram-se na Seção 5.

léxico, sendo composto por título, preço e autores. Um *conceito léxico* modela um conteúdo simples, sendo representado por uma elipse tracejada, como é o caso de *Preço*. ORM suporta também dois tipos de relacionamento: *associação* e *herança*. Um exemplo de associação ocorre entre *Livro* e *Preço*, possuindo restrições de cardinalidade. Um exemplo de herança ocorre entre *Publicação* e *Livro*.

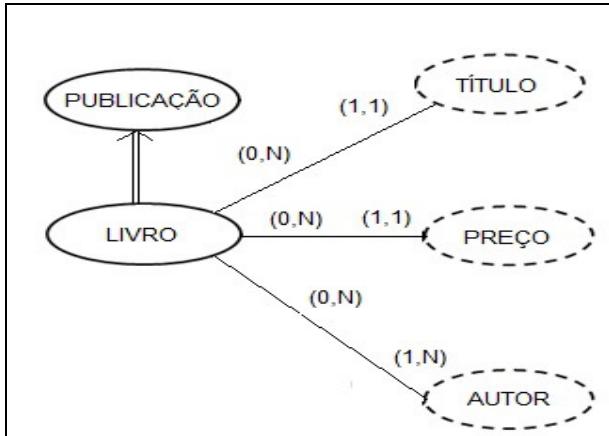


Figura 2. Exemplo de Modelagem ORM.

A linguagem *XPath* é utilizada para a descrição de mapeamentos. Ela foi escolhida porque define, de forma simples, expressões de caminho que indicam como acessar um elemento ou atributo em uma estrutura hierárquica XML. A estratégia geral de mapeamento de esquemas ORM para esquemas XML é a seguinte:

- *Mapeamento de conceitos:* expressão de caminho *absoluta* para o elemento ou atributo correspondente. Uma expressão absoluta *XPath* tem origem no elemento raiz do esquema XML e descreve a hierarquia de elementos a percorrer até o elemento/atributo. Caso haja mais de um caminho a partir do elemento raiz, todos devem ser indicados. Por exemplo, a expressão absoluta “/Acervo/Livro/ListaAutores/Autor” poderia estar associada ao conceito *Autor*, indicando que esta sequência de elementos deve ser percorrida para alcançar o elemento *Autor* correspondente no esquema XML de uma dada fonte de dados.
- *Mapeamento de relacionamentos:* expressão de caminho *relativa* para o elemento ou atributo correspondente. Uma expressão relativa *XPath* indica como se alcança um elemento/atributo no esquema XML a partir de outro elemento/atributo. BInXS mantém o mapeamento de relacionamentos em ambos os sentidos, visando facilitar a tradução de consultas. Supondo o exemplo dado no item anterior, o mapeamento do relacionamento entre *Livro* e *Autor* no sentido direto seria “*ListaAutores/Autor*” e no sentido inverso seria “*..*”.

Detalhes sobre as regras de geração de expressões de mapeamento para elementos ou atributos do esquema XML alvo são dados a seguir.

3.1 Mapeamento de Elementos e Atributos

O mapeamento para um elemento ou atributo XML é dado por uma expressão absoluta *XPath*, lembrando que um atributo na *XPath* é denotado por “@*nome_atributo*”. Os catálogos de mapeamento são consultados no momento da definição destes

mapeamentos para a geração de expressões de caminho válidas para o esquema XML original. O *catálogo de renomeações* é consultado para verificar se o nome do conceito correspondente foi alterado, de forma a recuperar o nome original do elemento ou do atributo. O *catálogo de elementos removidos* é consultado para verificar se elementos presentes no caminho até o elemento/atributo alvo foram removidos do esquema XML original e devem agora ser considerados. Ainda, caso o elemento seja virtual (esteja cadastrado no *catálogo de elementos virtuais*), não há definição de mapeamento pois o elemento não existe no esquema XML.

3.2 Mapeamento de Relacionamentos

O mapeamento típico de um relacionamento entre elementos ou entre elemento-atributo é dado por expressões relativas *XPath* que percorrem os dois sentidos do relacionamento. Os catálogos de mapeamento são consultados para considerar eventuais elementos renomeados, cujo nome original deve ser recuperado e inserido na expressão de mapeamento, e elementos removidos no caminho que conecta os elementos correspondentes aos conceitos relacionados, que devem também ser inseridos na expressão de mapeamento para definir o caminho correto no esquema XML original.

Caso o relacionamento entre dois conceitos *X* e *Y* envolva um elemento virtual (*Y*, por exemplo), define-se expressões de mapeamento relativas entre o elemento/atributo correspondente a *X* e todos os elementos componentes da estrutura anônima aninhada que *Y* encapsula, pois são estes elementos componentes que existem concretamente no esquema XML. Ainda, caso exista um *relacionamento de herança* entre um conceito genérico *X* e um conceito especializado *Y* no esquema conceitual, devem ser definidos mapeamentos de todos os conceitos associados à *X* com *Y*, uma vez que estes conceitos estão associados também a *Y* por herança.

4. Módulo Gerenciador de Mapeamentos para o Sistema BInXS

O módulo gerenciador de mapeamentos (ou *Módulo de Mapeamento* - MM) do BInXS [Bozzi 2009] interage com os módulos correspondentes às sub-etapas da etapa de *Conversão* da seguinte forma (ver Figura 3 (a)): um documento XSD de entrada é submetido ao *Pré-Processamento* e, neste momento, o MM popula os catálogos de apoio. Na sub-etapa seguinte, o MM consulta os catálogos para gerar expressões de mapeamento *XPath* para todos os conceitos e relacionamentos do esquema conceitual preliminar. Na sub-etapa *Reestruturação*, o MM verifica as alterações no esquema conceitual preliminar, atualiza expressões de mapeamento, se for o caso, e cadastra as mesmas no documento OWL que mantém o esquema conceitual definitivo.

O MM está codificado em Java por esta ser a linguagem adotada na implementação do BInXS. A API *JDOM* é utilizada para a manipulação da árvore DOM do documento XSD, em particular os métodos que utilizam a tecnologia SAX, dado o seu melhor desempenho em termos de gerência de memória. A implementação do MM estende a arquitetura de classes definida em [Garcia 2005], que implementou a etapa de *Conversão* do BInXS e adota o padrão *Model-View-Control* (MVC), bastante difundido na área de projeto de software [Gamma 2004]. O padrão de projeto MVC está definido da seguinte forma no BInXS (ver Figura 3 (b)): as classes *Acesso Dados* e *Evento* pertencem ao padrão *Modelo*. A primeira realiza o acesso a dados (esquema XML ou

conceptual preliminar) e a segunda corresponde a um evento de leitura de um dado de um esquema. As classes do padrão *Controle* são as sub-etapas da *Conversão* e a classe *Main*. A classe *Main* coordena a execução de cada sub-etapa sobre os dados lidos. Por fim, a classe *Visão Observador* é responsável por gerar resultados a partir de tarefas realizadas pelas classes de sub-etapas. No caso do BInXS, tem-se as classes de construção do arquivo XSD pré-processado e dos arquivos OWL (classe *ConstrutorOWL*) com os esquemas conceituais preliminar e definitivo.

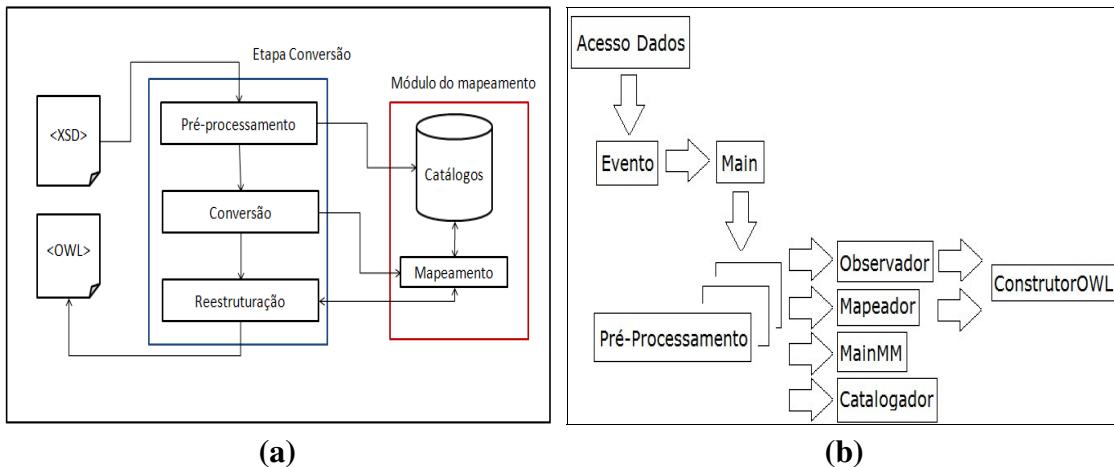


Figura 3. (a) Interação do MM com a Etapa de Conversão e (b) Principais Classes responsáveis pelo Fluxo de Execução da Conversão XML→Conceitual.

Na interação com a sub-etapa de *Pré-Processamento*, o MM define uma classe visão *Catalogador* que recebe notificações referentes a mudanças realizadas em elementos/atributos e cadastrá-las nos catálogos. O banco de dados MySQL foi utilizado para o armazenamento dos catálogos, sendo acessado através de classes responsáveis pelo acesso a cada uma de suas tabelas.

O MM define as expressões de mapeamento no final da sub-etapa de *Conversão* e início da sub-etapa de *Reestruturação*, pois é neste momento do processo uma estrutura de árvore DOM está disponível com informações sobre os conceitos definitivos e representativos de elementos e atributos, bem como seus relacionamentos. O MM, através de uma classe *Controle* chamada *MainMM*, realiza uma varredura nesta árvore DOM, gerando inicialmente os caminhos absolutos para cada elemento e atributo (dado XML) nesta árvore. A medida que cada dado XML é acessado, faz-se o acesso aos catálogos para descobrir refinamentos que o mesmo possa ter sofrido.

Após a produção dos mapeamentos, a sub-etapa de *Reestruturação* gera o documento OWL de saída. Durante esta geração, o MM se vale de uma classe *Visão Mapeador* que observa o trabalho da sub-etapa de *Reestruturação* e passa informações de mapeamento para ela. Assim, para cada classe que define um conceito no documento OWL, o *Mapeador* gera uma propriedade de mapeamento (*ConceptMapping*), onde se registra a expressão de mapeamento. Analogamente, gera-se, nas classes de relacionamento, uma propriedade de mapeamento (*RelationshipMapping*), onde são descritos os mapeamentos direto e inverso. Um exemplo de documento OWL e estas propriedades é mostrado na próxima Seção.

5. Estudo de Caso

Um estudo de caso foi aplicado à fonte de dados do periódico *SIGMOD Record* [Sigmod 2011], que mantém um repositório XML com informações sobre edições e um esquema XML associado. A estrutura parcial do esquema é mostrada na Figura 4 (a).

O esquema XML original, ao ser submetido à etapa de *Conversão* do BInXS, passa inicialmente pelo *Pré-processamento*. Neste momento, refinamentos podem ser feitos no esquema para melhor capturar a semântica dos dados. Neste estudo de caso realizaram-se algumas alterações, ilustradas na Figura 5 (a): (i) o elemento raiz *SigmodRecord* foi renomeado para *Journal*. Para tanto, o MM registra a alteração no catálogo de *Renomeações*; (ii) o elemento *IssueType* foi removido por ser considerado irrelevante. Neste caso, o MM faz o registro no catálogo de *Elementos Removidos*.

Na sequência, o esquema XSD pré-processado passa para a sub-etapa de *Conversão*, onde são gerados os conceitos correspondentes aos elementos e atributos do esquema de acordo com regras de conversão que não fazem parte do escopo deste trabalho. O esquema conceitual parcial gerado é ilustrado na Figura 5 (b). O MM analisa então esta estrutura e define expressões de mapeamento de acordo com a abordagem descrita na Seção 3 e a consulta aos catálogos. Os mapeamentos resultantes para os conceitos são os seguintes: */SigmodRecord* (*Journal*), */SigmodRecord/Issues* (*Issues*), */SigmodRecord/Issues/IssueType/volume* (*Volume*), */SigmodRecord/Issues/IssueType/number* (*Number*), */SigmodRecord/Issues/IssueType/articles* (*Articles*). No caso dos relacionamentos (*Direto/Inverso*), gera-se: “*Issue*” (D) | “..” (I) (*Journal-Issues*), “*IssueType/volume*” (D) | “..../” (I) (*Issues-Volume*), “*IssueType/number*” (D) | “..../” (I) (*Issues-Number*), “*IssueType/articles*” (D) | “..../” (I) (*Issues-Articles*).

Por fim, supondo que nenhuma modificação do esquema foi realizada na sub-etapa de *Reestruturação*, esta sub-etapa gera o documento OWL de saída. Parte deste documento é apresentado na Figura 4 (b). Neste documento, cada conceito e cada relacionamento é definido como uma subclasse de uma metaclasse do modelo ORM. Na definição do conceito *Number* é possível verificar a catalogação do seu mapeamento na propriedade *ConceptMapping*. A catalogação do mapeamento do seu relacionamento com o conceito *IssueType* também é apresentada na propriedade *RelationshipMapping*.

6. Trabalhos Relacionados

Informações de mapeamento entre esquemas são geralmente registrados em catálogos específicos de bancos de dados ou através de predicados de seleção que indicam (e eventualmente restringem) a posição do dado no esquema XML. Este trabalho adota a segunda abordagem, utilizando a linguagem *XPath* para definir expressões de mapeamento e documentos OWL para a sua catalogação. Esta estratégia é mais simples porque evita o gerenciamento de um catálogo específico para mapeamentos.

Apesar da desvantagem recém-salientada, alguns trabalhos clássicos utilizam catálogos [Madhavan et. al. 2001, Jensen et. al. 2003, Yuliana and Chittayasothorn, 2005]. Já outros trabalhos baseados em predicados de seleção empregam notações mais complexas para a definição de mapeamentos que este trabalho [McBrien and Poulovassilis 2001, Lóscio and Salgado 2003]. Isto dificulta a definição de mapeamentos, especialmente se ela for realizada de forma manual.

<pre> <?xml version="1.0" encoding="UTF-8"?> <xss:schema xmlns:xss="http://www.w3.org/2001/XMLSchema"> <xss:element name="SigmodRecord"> <xss:complexType> <xss:sequence> <xss:element name="issues"> <xss:complexType> <xss:sequence> <xss:element maxOccurs="unbounded" . . name="issue" type="IssueType"/> </xss:sequence> </xss:complexType> </xss:element> </xss:sequence> </xss:complexType> </xss:element> <xss:complexType name="IssueType"> <xss:sequence> <xss:element name="volume" type="xs:integer"/> <xss:element name="number" type="xs:integer"/> <xss:element name="articles"> <xss:complexType> . . </xss:complexType> </xss:sequence> </xss:complexType> </xss:schema> </pre>	<pre> . . <Class ID="number"> <subClassOf resource="#LexicalConcept"/> . . <restriction> . . <onProperty resource="#ConceptMapping"> <toClass><UnionOf parseType="collection"> <Thing about="#NumberSigmodRecord"/> </UnionOf> . . </subClassOf></Class> . . <Class ID="IssueNumber"> <subClassOf resource="#AssociationRelationship"/> <subClassOf> <restriction toClass="Issue"> . . <restriction toClass="Number"> . . <restriction hasValue="(1,1)"> . . <restriction hasValue="(1,n)"> . . <onProperty resource="#RelationshipMapping"> <toClass><UnionOf parseType="collection"> <Thing about="#IssueNumberSigmodRecord"/> </UnionOf> . . </subClassOf></Class> . . <ConceptMapping ID="#NumberSigmodRecord"> <Source>http://www.acm.org/sigmod/record/xml</Source> <PathExpression>/SigmodRecord/Issues/IssueType/number </PathExpression> </ConceptMapping> . . <RelationshipMapping ID="#IssueNumberSigmodRecord"> <Source>http://www.acm.org/sigmod/record/xml</Source> <Direct>IssueType/number</Direct> <Inverse>...</Inverse> </RelationshipMapping> . . </pre>
--	--

(a)

(b)

Figura 4. Documentos (a) XSD e (b) OWL parciais para o Estudo de Caso.

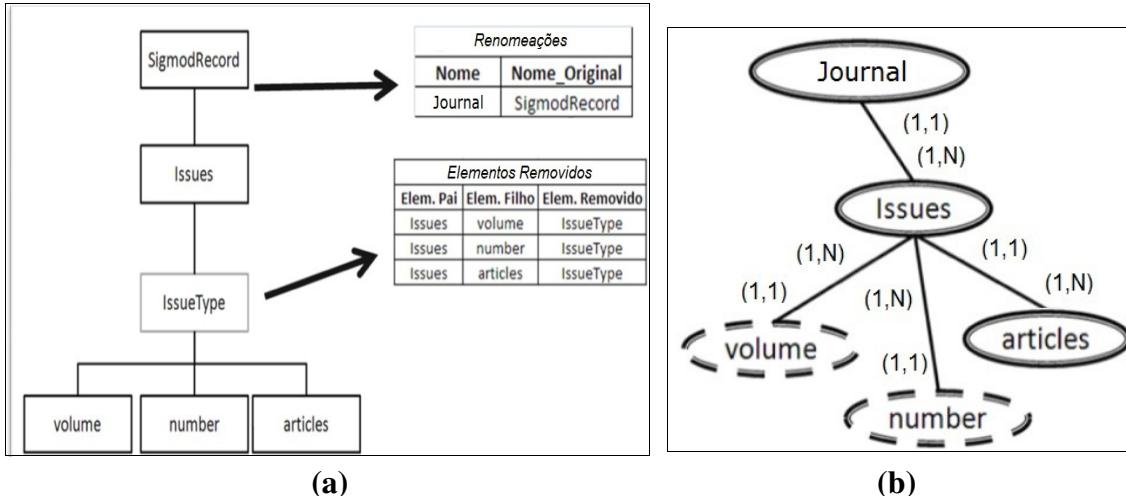


Figura 5. (a) Exemplo de Ações de Pré-Processamento e (b) Esquema Conceitual Parcial gerado para o Estudo de Caso.

7. Conclusão

Este trabalho apresenta o módulo automático de gerência de mapeamentos entre esquemas para o sistema de integração de esquemas XML BInXS. O módulo é responsável pela definição e persistência de informações de mapeamento, sendo necessário neste sistema para identificar, de forma não-ambígua, o elemento/atributo de um esquema XML local que cada conceito de um esquema conceitual global representa.

Este trabalho contribui não apenas com a implementação da estratégia de mapeamento proposta em [Mello 2002], mas também com a adaptação do BInXS a tecnologias mais recentes para a representação de esquemas XML e conceitual, no caso, XSD e OWL.

Trabalhos futuros incluem a realização de testes de validação com diversos esquemas XML de grande volume e o suporte ao mapeamento de relacionamentos não-hierárquicos em esquemas XSD, como é o caso de atributos do tipo *ID/IDREF*.

Referências

- Bozzi, E. V. M. T. (2009) “Geração de Informações de Mapeamento no Sistema de Integração de Esquemas XML BInXS”, Trabalho de Conclusão de Curso, Departamento de Informática e Estatística, Universidade Federal de Santa Catarina.
- Gamma, E. et. al. (2004) Padrões de Projeto, Bookman, 364 p.
- Garcia, L. G. (2006) “Uma Ferramenta para Engenharia Reversa de Esquemas XML em Esquemas Conceituais no Ambiente BInXS”, Trabalho de Conclusão de Curso, Departamento de Informática e Estatística, Universidade Federal de Santa Catarina.
- Halpin, T. (1998) “Object-Role Modeling (ORM/NIAM)”, Handbook on Architectures of Information Systems, Springer-Verlag, p. 81-102.
- Jensen et. al. (2003) “Converting XML DTDs to UML Diagrams for Conceptual Data Integration”, Data & Knowledge Engineering, v.44, n.3. p. 323-346.
- Lóscio, B. F. and Salgado, A. C. (2003) “Generating Mediation Queries for XML-based Data Integration Systems”, In: Simpósio Brasileiro de Banco de Dados, p. 99-113.
- Madhavan et. al. (2001) “Generic Schema Matching with Cupid”, In: 27th Conference on Very Large Data Bases, Morgan Kaufmann, p. 49-58.
- McBrien, P. and Poulovassilis, A. (2001) “A Semantic Approach to Integrating XML and Structured Data Sources”, In: Conference on Advanced Information System Engineering, Springer-Verlag, p. 330-345.
- Mello, R. S (2002) “Uma Abordagem Bottom-Up para a Integração Semântica de Esquemas XML”. Tese de Doutorado, Instituto de Informática, Universidade Federal do Rio Grande do Sul.
- Silva, F. S. and Mello, R. S. (2006) “BDTerm: Um Sistema de Gerenciamento de Bases de Dados Terminológicas”, In: Escola Regional de Banco de Dados, p. 93-98.
- OWL. (2011) “Web Ontology Language”, <http://www.w3.org/2004/owl>, Abril.
- Sigmod. (2011) “SIGMOD Record”, <http://www.acm.org/sigmod/record/xml>, Abril.
- XPath. (2011) “XML Path Language”, <http://www.w3.org/TR/xpath>, Abril.
- XSD. (2011) “W3C XML Schema”, <http://www.w3.org/XML/Schema>, Abril.
- Yuliana, O. Y. and Chittayasothorn, S. (2005) “XML Schema Re-Engineering Using a Conceptual Schema Approach”, In: International Conference on Information Technology, IEEE Computer Society, p. 255-260.

Uma Ferramenta para Consulta Estruturada na Web

Poline Lottin, Carina F. Dorneles

Depto. de Informática e Estatística - INE

Universidade Federal de Santa Catarina (UFSC)

Florianópolis, Santa Catarina – Brazil, 88.049-900

{poline, dorneles}@inf.ufsc.br

Resumo. *Sistemas de busca na Web são voltados ao suporte de consultas por palavras-chave que permitem pouca expressividade do usuário. Este artigo apresenta um novo modelo de ferramenta de busca, que utiliza alguns princípios de consultas a dados estruturados para indexar documentos Web e interpretar consultas do usuário.*

1. Introdução

A popularização da *Web* proporcionou o aumento de dados pelo mundo em documentos digitais, em sua maioria, descritos em arquivos textuais. Hoje, a forma mais popular de recuperar estes dados é a busca através de palavras-chave, que se mostra pouco expressiva, apresentando problemas com sinonímia e polissemia, por exemplo. A necessidade de encontrar informações mais relevantes em milhares de documentos na *Web* estimula pesquisas de novas técnicas mais eficazes para consultas textuais.

Atualmente o método de pesquisa mais conhecido é o utilizado por máquinas de busca, tais como *Google*¹ e *Yahoo!*², em que seu mecanismo de consulta é baseado no uso de palavras-chave. Este mecanismo procura colecionar o maior número possível de recursos através de softwares robôs, também chamados de *spiders* ou *crawlers*, que varrem páginas da *Web* e indexam seu conteúdo baseado em termos encontrados nos documentos.

Um dos primeiros motores baseados em robôs foi o *WebCrawler*, lançado em abril de 1994 [CENDON, 2001]. Atualmente, a maioria dos serviços que utilizam este mecanismo traz como resultado da consulta os links que são considerados mais relevantes em relação a um conjunto de palavras-chave informado pelo usuário. Os resultados são ordenados através de sua relevância, que pode ser definida de formas diferentes em cada serviço de buscas. A relevância de um link pode ser definida, por exemplo, a partir do número de repetições dos termos pesquisados, se ele possui algum termo no título da página e até mesmo pela popularidade do site.

Porém, estes serviços de consulta limitam o usuário a apenas buscar páginas através de palavras-chave, obrigando-o a deduzir quais termos devem ser inseridos na consulta para se obter a página desejada como resultado. Existe a necessidade de atribuir significado às palavras ou frases informadas nas consultas, desta forma seria possível contextualizar e relacionar conteúdos da web, obtendo resultados mais significativos.

¹ www.google.com

² www.yahoo.com

Este trabalho propõe uma nova ferramenta de busca, o *Find Me*, que tem por finalidade auxiliar consultas estruturadas na Web. A consulta é feita através de uma interface *Web* simples, que facilita a construção da consulta até mesmo para usuários que não tenham familiaridade com o uso de linguagens de consulta estruturadas. A ferramenta possui um motor de busca, que varre a Web em busca de páginas HTML. Este motor possui um certo conhecimento semântico associado a respeito de padrões de texto em arquivos HTML e é capaz de capturar dados em textos e tabelas do documento, identificando sua relevância e os relacionando com as demais informações. Com os dados relacionados entre si, estas informações podem ser indexadas de forma estruturada facilitando a consulta. O relacionamento entre páginas HTML é feito através dos links que se encontram em *tags* do tipo <A HREF> e fazem referência a outro documento através de URL.

Este artigo está organizado com segue. A Seção 2 apresenta uma descrição dos trabalhos relacionados. A Seção 3 descreve a idéia básica do funcionamento da ferramenta, sua arquitetura e as heurísticas usadas para a indexação dos dados presentes em páginas HTML. A implementação da ferramenta é apresentada na Seção 4. Finalmente, na Seção 5, são descritos os trabalhos futuros e a conclusão.

2. Trabalhos Relacionados

A ferramenta Mesa, desenvolvida por [MERGEN, FREIRE, HEUSER, 2008] é um motor de busca que permite acesso aos dados estruturados em tabelas HTML de documentos *Web*. Mesa possibilita a consulta de dados previamente indexados, tanto com palavras-chave quanto com *queries* formatadas em SQL. Quando a busca é feita por palavras-chave, o Mesa simplesmente retorna às páginas que possuem tabelas *Web* com o termo. Os dois tipos de consulta ainda podem ser agrupados, buscando tabelas que possuem os termos do SQL, em que a palavra-chave está presente. O resultado da consulta é exibido em um sumário com todas as tabelas resultantes e a URL de onde esta tabela foi extraída. Ao disponibilizar o link para a página original do conteúdo, o sistema permite que o usuário navegue pelo real contexto da tabela podendo alcançar novas informações que possuem relação com a busca. A consulta permite cláusulas de condição, como *ano > 1990*, onde a busca resulta em tabelas que possuam pelo menos uma linha com este valor. Os autores utilizaram um *crawler* específico para acessar páginas que possuem tabelas em um contexto de filmes. A ferramenta indexa dados específicos de filmes armazenados no domínio do *Wikipedia*, portanto, como regra para a indexação das páginas, o documento deve possuir pelo menos uma tabela com as três colunas: *title*, *film* e *movie*. A ferramenta também utiliza um documento de índice que auxilia a consulta por palavras-chave. Algumas técnicas de otimização foram utilizadas, como o *merge-join* que relaciona dados de diferentes tabelas. A necessidade de formulação de uma consulta em SQL foi apontada como limitação da ferramenta, visto que informar palavras-chave seria muito mais fácil para o usuário, porém, pouco expressiva. Como trabalhos futuros, os autores sugerem a pesquisa de novas interfaces de consulta que equilibrem expressividade e simplicidade de uso.

Propostas de linguagens estruturadas para *Web* já surgiram, tais como Squeal [SPERTUS, STEIN, 2000] e SPARQL [PRUD'HOMMEAUX, SEABORNE, 2008]. A linguagem *Squeal* possui uma sintaxe similar à linguagem SQL, com construções efetuadas sobre os elementos HTML. Esta linguagem permite o acesso aos elementos e

atributos das *tags* do HTML. Porém, não é possível efetuar consultas usando a semântica do domínio ao qual se está consultando. O *Squeal* difere da linguagem SPARQL (*SPARQL Protocol and RDF Query Language*) cujo objetivo principal é ser a linguagem padrão para a *Semantic Web*, e é considerada a peça chave nas tecnologias desenvolvidas na *Semantic Web*. Toda a linguagem foi projetada para ser executada sobre documentos RDF. Portanto, o domínio a ser consultado deve estar descrito através de RDF para que consultas SPARQL possam ser executadas.

O presente trabalho difere dos trabalhos apresentados acima nos seguintes pontos: ao contrário do *Mesa*, o *Find Me* está sendo desenvolvido para efetuar busca no documento HTML como um todo, não apenas em estruturas de elementos *<table>*. Em relação às linguagens estruturadas Squeal e SPARQL, a proposta apresentada neste artigo não constrói suas consultas sobre a estrutura dos elementos HTML, como a Squeal, nem faz uso de um descritor semântico como o RDF, como se propõe a SPARQL. O objetivo é efetuar consultas estruturadas sobre objetos, e atributos, encontrados em documentos HTML e que são definidos através de certas heurísticas (apresentadas a seguir).

3. Find Me

Esta seção apresenta o funcionamento do *Find Me*, uma ferramenta de busca que permite consultas estruturadas em documentos HTML. A principal contribuição da ferramenta é permitir a construção de consultas mais expressivas sobre documentos em HTML na *Web*, utilizando princípios de consultas estruturadas com dados não necessariamente encontrados em algum tipo de estrutura.

A Figura 1 representa a arquitetura geral do sistema. O *Find Me* é composto por três componentes principais:

- **Crawler:** este componente é encarregado de percorrer a World Wide Web automaticamente, identificar documentos em HTML e encaminhá-los para o Parser. O ponto inicial da varredura do Crawler na Web é dado através de uma ou mais páginas Web que são denominadas sementes.
- **Parser:** recebe do Crawler cada documento que utiliza um XML Parser para reconhecer o documento, suas características e indexá-las de acordo com as regras definidas nas Seções 3.1. e 3.2.
- **Índice:** é encarregado de armazenar informações sobre os documentos em uma estrutura simples, utilizando poucos campos em uma única tabela, o que facilita o processo de consulta.

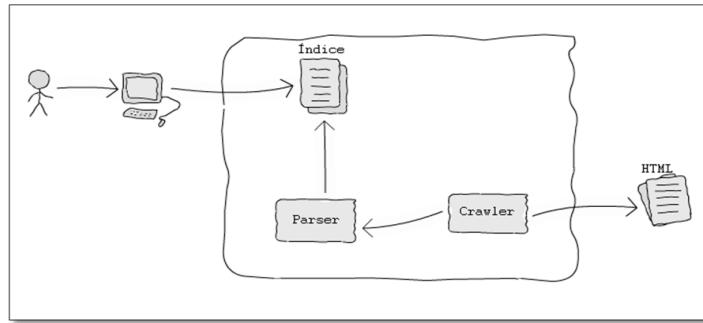


Figura 1 - Arquitetura da ferramenta Find Me

O processo de indexação do documento parte do princípio de que todo documento HTML é um objeto na *Web* e cada palavra encontrada pelo *parser* é identificada como um atributo do objeto. Partindo desta definição, é possível que um atributo contido em uma *tag* `<a>` (que refencie outro documento) seja também um objeto.

Para que os documentos sejam encontrados através de consultas estruturadas, é necessário identificar o tipo deste objeto e as relações entre os atributos da página. Por exemplo, para que a consulta “*SELECT * FROM cidades WHERE população > 5.000*” consiga recuperar documentos sobre cidades que possuem população maior que 5 mil, é preciso primeiro identificar que o documento a ser indexado possui informações sobre uma cidade, ou seja, identificar que o tipo deste objeto é cidade. Em seguida, também é necessário identificar que o documento possui o atributo população com um valor maior que 5 mil.

Encontrar relações do tipo “atributo-valor” no conteúdo de um documento é o principal desafio do *Find Me*. Para isto, foram criadas algumas heurísticas que identificam tipos de documentos e atributos. Estas heurísticas são descritas respectivamente nas Seções 3.1. e 3.2.

3.1. Heurísticas para determinação do tipo de um objeto

Inicialmente, foi determinado um escopo de páginas similares entre si, para que o *crawler* percorra. O motor de busca inicia a visita de páginas a partir de uma semente definida como uma página da Wikipedia, em que todos os estados e municípios do Brasil são listados e referenciados em *hyperlinks* (isso significa que, no momento, o protótipo funciona para este conjunto de páginas HTML; o padrão será generalizado para sua versão final).

A Figura 2 mostra o código fonte HTML do documento apresentado na Figura 3. Tanto o *layout*, quanto o código HTML foram simplificados para facilitar a compreensão dos exemplos.

```

<table class="infobox">
  <tbody>
    <tr><td>Município de Florianópolis</td></tr>
    <tr>
      <th>Distância até a capital</th>
      <td>1 692 km</td>
    </tr>
    <tr>
      <th><a href="http://...>Área</a></th>
      <td>433,317 km2</td>
    </tr>
    <tr>
      <th><a href="http://...>População</a></th>
      <td>421 203 hab.</td>
    </tr>
    <tr>
      <th><a href="http://...>Mesorregião</a></th>
      <td><a href="http://...>Grande Florianópolis</a></td>
    </tr>
  </tbody>
</table>

```

Figura 2 - Exemplo de código HTML. FONTE: adaptado de <http://pt.wikipedia.org/wiki/Florian%C3%B3polis>



Figura 3 - Exemplo de padrão de layout. FONTE: adaptado de <http://pt.wikipedia.org/wiki/Florian%C3%B3polis>

Para este domínio de páginas, a heurística para identificar o tipo do documento é determinada a partir do valor encontrado na primeira linha da primeira tabela do documento HTML. Segundo o caminho na árvore DOM, deve-se recuperar o valor encontrado na seguinte tag: `/body/table[@class='infobox']//tr//td`. O texto encontrado geralmente significa o contexto da página. Por exemplo, no caso do documento da Figura 3, a primeira linha da tabela possui o texto Município de Florianópolis.

3.2. Heurísticas para identificação de atributo

Como descrito anteriormente, as palavras de um documento, identificadas pelas heurísticas definidas neste trabalho, são consideradas atributos do objeto. Neste caso, entende-se como objeto o documento que está sendo verificado. O desafio principal é conseguir identificar relações entre a palavra identificada como atributo e a palavra que corresponde ao valor do atributo. Por exemplo, na frase “*a capital do Brasil é Brasília*”, o desafio principal é determinar que para este objeto, o atributo “*capital*” possui valor igual a “Brasília”. O *layout* apresentado na Figura 3 foi encontrado no domínio da *Wikipedia* e é seguido como padrão em grande parte dos documentos do site. Por este motivo, uma heurística é determinada para a identificação de atributos de um objeto nestas páginas.

No código fonte esboçado na Figura 2, cada característica pode ser encontrada seguindo o caminho na árvore DOM: `/body/table[@class='infobox']//tr/th`. Consultando a tag irmã da tag `<th>`, será encontrada uma tag `<td>`, onde o valor da característica é encontrado. Neste exemplo, é possível observar que nas tags `<th>`, o texto encontrado é o nome de uma característica, como a palavra “População”. Na tag seguinte à `<th>` encontra-se o valor “421 203 hab.”. Com base no apresentado, o *Find Me* pode concluir que a população de Florianópolis é igual a 421.203 habitantes.

Como cada atributo pode estar referenciando um novo documento (possuir um *hyperlink* associado), uma página que descreve a cidade de Florianópolis, por exemplo, tem atributos primitivos como “Distância até a capital” e atributos que serão outros objetos como “População” que faz referência à página principal sobre a palavra “população” no *Wikipedia*. Este cenário cria a necessidade de definir dois tipos de atributos: atributos-primitivos e atributos-objetos.

Tabela 1 - Descrição dos campos da tabela de índice

Campo	Descrição
ID	Código identificador único.
ID_PAI	Código que identifica o documento ao qual o atributo pertence.
TIPO	É preenchido somente quando a linha for um objeto. Representa o tipo do objeto.

NOME	Pode ser tanto o nome de uma página, quanto o nome do atributo.
VALOR	É preenchido somente quando a linha for um atributo, representa o valor do atributo.
NOME_URL	No caso da indexação de objetos, é a URL do documento. Para a indexação de um atributo, o campo representa a URL a qual o atributo faz referência, no caso de atributos contidos em <i>hyperlinks</i> .
VALOR_URL	Representa a URL a qual o valor do atributo faz referência, caso a palavra esteja contida em um <i>hyperlink</i> .

4. Implementação

As etapas de indexação foram desenvolvidas utilizando a linguagem de programação Java³. A mesma tecnologia foi utilizada para a implementação da interface de consulta através de JSP⁴ (*Java Server Page*).

Um banco de dados relacional foi utilizado para o armazenamento do índice. Devido ao pobre desempenho, já conhecido em bancos de dados relacionais, principalmente ao efetuar joins entre tabelas, o índice foi desenvolvido em apenas uma tabela. Porém, para suportar consultas mais aprimoradas, esta tabela ainda conta com um auto-relacionamento, onde um atributo pode referenciar sua origem (documento a quem pertence). A tabela e seu relacionamento podem ser vistos na Figura 4 e a descrição dos campos da tabela de índice são descritos na Tabela 1.

³java.sun.com

⁴<http://java.sun.com/products/jsp/docs.html>

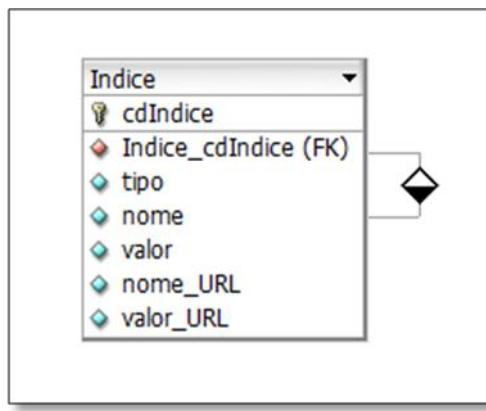


Figura 4 - Modelo lógico da tabela de índice

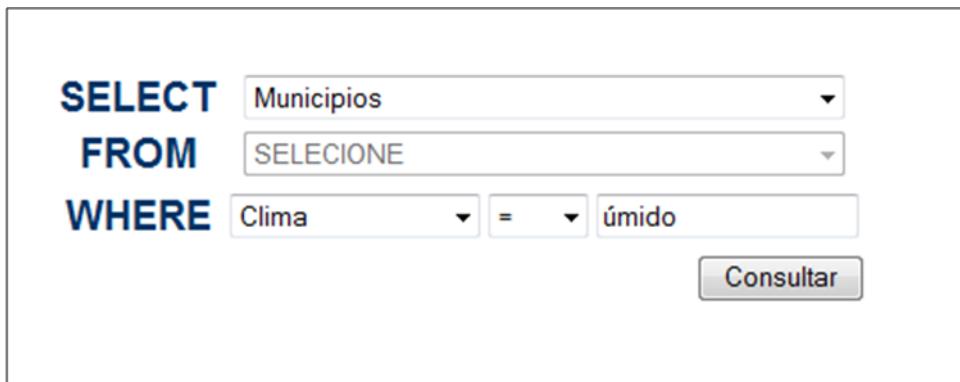
A interface final com o usuário é representada na Figura 5, onde os campos necessários para a consulta são pré-definidos de acordo com o conteúdo encontrado na tabela de índice. O campo *SELECT* possui uma lista de todos os nomes dos atributos encontrados nos documentos indexados e também uma lista com todos os tipos de documentos identificados. O campo *FROM* apresenta os tipos de documentos em que a consulta deverá ser efetuada. Os valores possíveis no campo da cláusula *WHERE*, assim como o campo *SELECT*, são também os atributos já indexados. Apenas o campo referente ao valor a ser comparado na cláusula *WHERE* é aberto para digitação livre do usuário.

Figura 5 - Interface de consulta

A consulta informada nos campos da Figura 5 significa que deverão ser listados todos os prefeitos de municípios que possuem população menor que 20.000 habitantes. Em outras palavras, serão recuperados todos os índices com o nome “Prefeito(a)” pertencentes a um documento do tipo “Cidade” que também possua a característica (atributo) “População” com um valor menor que 20.000.

Outra possibilidade de consulta é representada pela Figura 6, onde se deseja buscar todos os municípios com clima úmido. Note que neste caso a cláusula *FROM* deve ficar desabilitada. A consulta recupera as linhas do índice em que o tipo do

documento é “Cidade” e que possua um atributo chamado “Clima”, com valor igual a “úmido”. Os resultados serão representados através da URL de cada documento.

A interface de consulta é uma janela com um formulário para construção de consultas SQL. O formulário contém os seguintes campos:

- SELECT:** Um dropdown menu com a opção "Municípios".
- FROM:** Um dropdown menu com a opção "SELEÇÃO".
- WHERE:** Um campo com o critério "Clima = úmido".
- Consultar:** Um botão cinza para executar a consulta.

Figura 6 - Interface de consulta

A consulta “montada” pelo usuário no estilo do SQL é utilizada apenas para permitir maior expressividade e não é efetuada diretamente ao índice para obter os resultados esperados. Na verdade, todas as consultas informadas no *Find Me* precisam ser interpretadas e adaptadas para o contexto da tabela de índice. Esta etapa deve ser transparente ao usuário. Um exemplo de adaptação de consulta é representado a seguir.

Consulta informada pelo usuário:

```
SELECT Prefeito  
FROM Municípios  
WHERE população < 20000;
```

Consulta efetuada à tabela de índice

```
SELECT *  
FROM Indice objeto  
JOIN Indice atributo  
ON objeto.ID = atributo.ID_PA  
WHERE objeto.TIPO = "C"  
AND atributo.NOME like "populacao"  
AND atributo.VALOR = 20000;
```

5. Conclusões e Trabalhos Futuros

O artigo apresentou o *Find Me*, uma ferramenta para busca estruturada na Web. O objetivo é efetuar consultas estruturadas em busca de objetos, e atributos, encontrados

em documentos HTML e que são definidos através de certas heurísticas definidas para a determinação de objetos e para a determinação de atributos.

Como trabalhos futuros, pretende-se desenvolver experimentos para avaliação do sistema, testando a confiabilidade dos resultados retornados pelo sistema. Estes testes serão efetuados através das medidas de Precisão, Revocação e F-value [YATES e NETO, 1999]. Onde a medida de precisão exprime quantos registros relevantes foram recuperados, enquanto a de revocação mede a proporção de registros relevantes recuperados.

Além disso, as heurísticas para a categorização de um documento ainda precisam ser estudadas para aumentar o escopo, considerando padrões mais genéricos do que os atuais. Caso algum problema seja identificado durante os testes, a correção deverá ser efetuada nesta etapa. Poderão ser implementados métodos que convertam os valores dos atributos em unidades de medida padrão. O valor para o atributo área, por exemplo, pode ser representado tanto por 25 km² ou 25.000 m². Atualmente o sistema apenas captura o valor independente de sua unidade de medida, podendo indexar o número 25 e 25.000 para um mesmo valor. Essa diferença gera inconsistências em consultas utilizando a cláusula *WHERE*

6. Referências

- Cendon, Beatriz Valadares. Ferramentas de busca na Web. Ci. Inf. [online]. 2001, vol.30, n.1, pp. 39-49. ISSN 0100-1965.
- Madhavan, Jayant et al. Harnessing the Deep Web: Present and Future. Biennial Conference on Innovative Data Systems Research - CIDR, 2009. 6 p.
- Mergen, Sergio; FREIRE, Juliana; HEUSER, Carlos. MESA: A Search Engine for QueryingWeb Tables. Salt Lake City - U.S [s.n.], 2008. 6 p.
- Prud'hommeaux, E.; Seaborne, A. SPARQL Query Language for RDF. W3C Recommendation 15 January 2008.
- Spertus, E.; Stein, L.A. Squeal: A structured query language for the web. In Proceedings of the 9th International World Wide WebConference (WWW9), pages 95– 103, Amsterdam, The Netherlands, 2000.
- Wives, Leandro Krug. Tecnologias de descoberta de conhecimento em textos aplicadas à inteligência competitiva. Porto Alegre – RS. [s.n.], 2002. 116 p.
- Yates, Ricardo B.; NETO Berthier R. Modern Information Retrieval. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1999.

Análise comparativa dos recursos de alta disponibilidade entre os bancos de dados Oracle e SQL Server

Eduardo de O. Gomes, Fábio Silva Lopes

Faculdade de Computação e Informática – Universidade Presbiteriana
Mackenzie Rua da Consolação, 930 – 01302-907 – São Paulo – SP – Brasil

duduardo1@gmail.com, flopes@mackenzie.br

Abstract. This article aims to evaluate technologies to ensure high availability to server database, through comparison of features available in two database management systems, Oracle and SQL Server. The study was conducted by the acquisition of features available to both DBMS, create a comparative table contents the research results and their analysis. Was observed that the Oracle presents major high availability than SQL Server, however, SQL Server showed higher robustness in the points of failure analysis as well as to attend demands of disaster recovery.

Resumo. Este artigo objetiva avaliar as tecnologias para garantir alta disponibilidade para servidores de banco de dados, por meio da comparação de características disponíveis em dois sistemas gerenciadores de banco de dados, Oracle e SQL Server. O estudo foi conduzido pela aquisição das características disponíveis em ambos os SGBD's, elaboração de um quadro comparativo contendo os resultados da pesquisa e respectivas análises. Observou-se que o gerenciador Oracle apresenta mais recursos de alta disponibilidade que o SQL Server, no entanto, este outro apresentou maior robustez na análise de pontos de falha e recursos para atender demandas de recuperação em caso de desastre.

1. Introdução

Assegurar a disponibilidade dos recursos de computação é uma das tarefas que mais exigem esforços dos profissionais de informática, um bom sistema depende da eficiência do software e da disponibilidade dos recursos aos usuários.

Mesmo com necessidades claras e requisitos funcionais bem definidos, muitas soluções são implementadas sem a devida preocupação com critérios de alta disponibilidade relativos ao acesso a dados. Fatores como a falta de capacitação profissional e deficiência na compreensão de como tornar ambientes mais disponíveis, pode proporcionar tempo de indisponibilidade trazendo prejuízos às organizações.

Uma solução de alta disponibilidade trata eventuais falhas de hardware e software, mantendo a aplicação disponível para o usuário final (MSDN, 2010). Portanto, no que tange a compreensão de técnicas de implantação de alta disponibilidade, a necessidade de aprofundar os conhecimentos relacionados aos gerenciadores de banco de dados é

evidente e está presente na pauta das discussões sobre arquiteturas mais capazes de atender demandas dos usuários.

No entanto, observa-se a recorrente dificuldade de classificar os elementos a serem comparados em uma eventual análise para subsidiar a escolha da ferramenta mais adequada. Assim, o objetivo deste estudo é avaliar duas aplicações de alta disponibilidade e comparar as características disponíveis a fim de obter melhor clareza de opções em eventuais escolhas desta natureza.

Sistemas gerenciadores de banco de dados como o Oracle e o SQL Server são largamente utilizados no mercado e disponibilizam aplicações para atender as demandas de alta-disponibilidade: o RAC (Real Application Clusters) e o *Database Mirroring* respectivamente. Sendo assim, este estudo foi conduzido no sentido de levantar características destas ferramentas com vistas a obter um quadro comparativo destes produtos, bem como, uma lista de itens a serem observados em um eventual processo de aquisição.

Desta forma, estas tecnologias, configuradas de forma eficiente e acompanhadas de um bom projeto de contingência garantem a disponibilidade do acesso a dados em diversos cenários de armazenamento. Portanto, são relevantes para a composição da arquitetura de sistemas de informação.

As próximas seções apresentarão a descrição das aplicações, seguido da comparação elaborada entre os produtos.

2. *Database Mirroring*

Database Mirroring é uma solução de alta disponibilidade no nível de banco de dados disponível para o SQL Server, que opera através do espelhamento do banco de dados em duas cópias residentes em instâncias separadas do SQL Server, geralmente em computadores diferentes. Em determinado momento, somente uma cópia do banco de dados estará disponível para as transações dos usuários enquanto a outra cópia permanecerá em espera (MICROSOFT, 2010).

O servidor que armazenará a cópia do banco de dados disponível é conhecido como servidor *Principal*. Já o servidor que armazenara a cópia do banco de dados que fica em espera é conhecido como *Mirroring*. Para manter a réplica atualizada, o SQL Server envia as transações efetuadas no arquivo LOG do servidor *Principal* para o arquivo de LOG do servidor *Mirroring*. Os servidores envolvidos neste processo são considerados parceiros, já que ambos os servidores podem, a qualquer momento, assumir o papel de *Principal* ou *Mirroring*.

É possível configurar também um servidor *Witness*. Este servidor é utilizado para automatizar a troca de papéis entre os servidores *Principal* e *Mirroring* em caso de falha. O servidor *Witness* monitora ambos servidores e, caso ele note uma falha de conexão com o servidor *Principal*, ele altera o papel do servidor *Mirroring* para *Principal*.

3. Oracle RAC

O Oracle RAC é uma aplicação disponível desde a versão 9i de 2001. Trata-se de um software para *clustering* e alta disponibilidade em ambientes de banco de dados Oracle. A solução se baseia em prover a múltiplos computadores acesso simultâneo a um único banco de dados. Um banco de dados *Real Application Clusters* é altamente disponível e escalável.

A falha de um dos nós do *cluster* não afeta as sessões de clientes nem a disponibilidade do próprio cluster até o último nó no cluster falhar; o único impacto que um nó perdido tem sobre o cluster é uma leve degradação no tempo de resposta, dependendo do número total de nós no cluster (ORACLE, 2010).

4. Comparações

O estudo identificou que a falha em um dos nós no *cluster* do banco de dados RAC não afeta as sessões de clientes nem a disponibilidade dos dados até que o último nó do *cluster* falhe. Porém, os nós possuem apenas um SGBD configurado. Os dados são armazenados em uma única *Storage*. Se ocorrer uma falha de hardware ou conexão com a *Storage* todo o ambiente de banco de dados tornar-se-á indisponível.

No caso do *Database Mirroring* havendo uma falha na conexão entre o servidor *Principal* e o servidor *Mirroring* a cópia em *Stand-By* ficará desatualizada. Ocorrendo uma falha na conexão entre as instâncias *Principal* e *Witness* o sistema troca os papéis, mesmo que o servidor *Principal* não apresente falha.

Com relação ao desempenho, o Oracle RAC apresenta vantagem sobre o SQL Server *Database Mirroring*, por permitir vários nós em seu *clustered*. O Oracle RAC, além de garantir alta disponibilidade faz o balanceamento de carga entre os nós melhorando o processamento das operações.

Em casos de situações de *Disaster Recover*, a solução SQL Server *Database Mirroring* apresenta melhor resposta, pois, garante alta disponibilidade em casos de falha de hardware o *Database Mirroring* e em casos de desastres naturais. O banco de dados espelhado pode ser armazenado em um site diferente do *Principal*, ou seja, se for garantida a conexão entre as instâncias do SQL Server, a réplica poderá ser armazenada em outro site. O Oracle necessita de outra ferramenta, o DataGuard, para prover esta mesma característica.

A capacidade de ocultar um problema para o usuário final é outro ponto que o Oracle RAC trata com mais eficiência. Quando ocorre falha em um dos nós, outro nó assume a carga de trabalho, de maneira transparente para a aplicação e os usuários.

Sobre manutenção e complexidade, é necessário analisar o cenário de implantação, pois, caso o ambiente seja configurado com o Oracle RAC, a quantidade de nós envolvidos determinará o dimensionamento da complexidade e manutenção, e caso o ambiente seja configurado com o SQL Server *Database Mirroring*, o modo de operação, tipo de licença e quem vão determinar a complexidade.

O tempo de indisponibilidade em caso de falha é semelhante, no caso do RAC a indisponibilidade não será percebida, pois um nó assumirá o papel do nó que falhou, já a solução SQL Server terá indisponibilidade de cinco milissegundos.

A síntese de comparações realizada entre as soluções é apresentada na Tabela 1.

Tabela 1 – Síntese de Comparações entre as soluções avaliadas:

	ORACLE RAC	DATABASE MIRRORING
Ponto de Falha	Falha de hardware ou falha de conexão com a <i>storage</i> onde estão os dados.	Falha no link entre o principal e <i>witness</i> ou entre principal e <i>mirroring</i>
Desempenho	RAC faz balanceamento de carga (mais rápido)	Dependendo da edição do SQL Server o modo de sincronismo pode ser síncrono ou assíncrono podendo impactar no desempenho
Disaster Recovery	O RAC precisa que os nós do <i>cluster</i> estejam próximos para atender requisitos de velocidade.	O banco de dados espelhado pode estar localizado em outro site, em outra cidade (solução em desastres naturais)
Transparência	A carga de trabalho é distribuída entre os nós de maneira transparente	Precisa de suporte da linguagem de programação para ser transparente a troca de papéis
Custos	A licença é exigida para cada nó.	A licença é exigida para o principal, <i>mirror</i> e <i>witness</i> (se houver).
Manutenção e complexidade	Depende do cenário, quanto mais nós mais manutenções e mais complexidade	Depende do cenário, se usar <i>witness</i> três servidores precisaram de manutenções senão apenas dois.
Tempo de indisponibilidade	Não há	Cinco milissegundos.

5. Considerações Finais

Alta disponibilidade não acontece por acaso. Ela é alcançada através do fortalecimento da combinação de pessoas, processos, e tecnologia. Nesta linha, a opção correta por aspectos arquitetônicos do sistema podem fazer diferença no sucesso da aplicação.

Os resultados deste estudo sugerem atenção especial aos itens analisados, visando minorar dificuldades no alcance de níveis elevados de disponibilidade. Preparar a plataforma apropriada de hardware e software são pontos relevantes. Contudo, manter alta disponibilidade do sistema é resultado de bom planejamento e assimilação de práticas obtidas por meio do uso adequado das tecnologias existentes.

6. Referências

MICROSOFT. Comparação de Funcionalidade do SQL Server, 2005. Disponível em: <<http://www.microsoft.com/portugal/sql/prodinfo/features/compare-features.mspx>> Acesso em: 20 ago.2010

MSDN. Visão geral das soluções de alta disponibilidade. Biblioteca MSDN. Disponível em: <<http://msdn.microsoft.com/pt-br/library/ms190202.aspx>> Acesso em: 15 jun. 2010.

ORACLE. Configurações disponíveis para o Oracle RAC, 2004. Disponível em: <<http://www.oracle-base.com/articles/10g/OracleDB10gR2RACInstallationOnLinuxUsingNFS.php>> Acesso em: 01 jun. 2010.

O Uso de Descoberta de Conhecimento em Bases de Dados Associado a Sistemas de Informações Geográficas no Auxílio à Definição de Políticas de Saúde Pública

Suián Boff Menegás¹, Marta Rosecler Bez¹, Guillermo Nudelman Hess²

¹Intituto de Ciências Exatas e Tecnológicas – Universidade Feevale
RS 239, 2755 – 93352-000 - Novo Hamburgo – RS

²Centro de Formação e Aperfeiçoamento em Informática – Universidade Feevale
RS 239, 2755 – 93352 – 000 – Novo Hamburgo - RS
suián.bm@canela.com.br, {martabez, hess}@feevale.br

Abstract. This paper presents a proposal for using tools to aid decision making in public health. The proposal is to implement the process of Knowledge Discovery in Databases on the DATASUS database and to integrate the gathered results into a Geographic Information System. This proposal focuses mainly on the distribution of knowledge acquired through the process of Knowledge Discovery in Databases in space and time seeking to understand the events and patterns of health, on a regional and temporalized basis.

Resumo. O presente artigo apresenta uma proposta de uso de ferramentas de auxílio à tomada de decisão na saúde pública. A proposta baseia-se na aplicação do processo de Descoberta de Conhecimento em Bases de Dados sobre as bases de dados do DATASUS, e a posterior integração das informações e conhecimentos obtidos a um Sistema de Informações Geográficas. Tal proposta tem como foco principal a distribuição do conhecimento adquirido através do processo de Descoberta de Conhecimento em Bases de Dados no espaço e no tempo buscando o entendimento dos eventos e padrões de saúde, de forma regionalizada e temporalizada.

1. Introdução

Os problemas de gestão da saúde já são conhecidos no Brasil desde a época da industrialização. Nesta época, não se sabia onde nem como aplicar os poucos recursos destinados à saúde por falta de informações disponíveis. Como agravante desta situação, a grande mudança no sistema de saúde brasileiro, iniciada no período de 1993 a 1996, com a criação da Norma Operacional Básica (NOB/93), quando houve, principalmente, a municipalização do Sistema Único de Saúde (SUS) (SANTOS, 1998), fez com que os municípios herdassem a enorme carga de responsabilidade de gerenciar e promover políticas próprias de saúde pública, com o objetivo de apurar, melhorar e manter a qualidade dos serviços prestados.

Nesse contexto, surgiram demandas crescentes de meios e métodos que possibilitem a criação de métricas para apontar a qualidade da saúde pública, bem como de ferramentas que auxiliem os gestores na tomada de decisão, principalmente no que diz respeito à distribuição de recursos em geral.

Nesta situação, o presente artigo pretende apresentar uma forma de auxiliar na solução para os problemas de distribuição e aplicação dos recursos da saúde pública, integrando as informações geradas no processo de Descoberta de Conhecimentos em Bases de Dados sobre as bases do DATASUS, a um Sistema de Informações Geográficas. Tal proposta pretende disponibilizar aos gestores de saúde pública, não somente as informações sobre a saúde de seus municípios traduzidas em relatórios gerenciais, como também permitir que tais informações sejam distribuídas no espaço e no tempo, permitindo a esses gestores avaliar se os recursos destinados a saúde estão sendo distribuídos de forma condizente com a realidade, associar eventos a aspectos de meio ambiente, criar projeções epidemiológicas, projeções de consumo de medicamentos por região, e ainda, fazer inferências sobre a população de cada região do município.

O restante do artigo está estruturado como segue: na seção 2 é apresentado o DATASUS, que é a base de dados pública de saúde. Na seção 3 são discutidos os conceitos de descoberta de conhecimento e banco de dados e de sistemas de informações geográficas relevantes no contexto deste artigo. Ao final da seção, alguns trabalhos relacionados são apresentados. A seção 4 apresenta a abordagem proposta neste artigo, bem como os resultados iniciais obtidos. Por fim, conclusões e trabalhos futuros são discutidos na seção 5.

2. A base de dados DATASUS

O Brasil, até pouco tempo atrás, definia suas políticas de saúde pública através de indicadores de saúde obtidos com o uso da epidemiologia como ferramenta de diagnóstico da saúde. Durante este período, tudo o que os municípios faziam era seguir tais indicadores e tais políticas. Porém, tais parâmetros já não são mais consoantes com a realidade, pois sabe-se hoje que os indicadores de saúde de um município ou local, na maioria das vezes, não servem à outro (Drumond Jr., 1993). Partindo desta visão, e da visão da estrutura atual do Sistema Único de Saúde (SUS) do Brasil, cada município precisa criar processos de avaliação do estado da própria saúde pública, a fim de garantir um direcionamento dos recursos de forma constante e condizente com a realidade local.

Neste cenário, o SISTEMA DATASUS – sistema de bases de dados do SUS – tem por objetivo manter e divulgar informações estatísticas e financeiras de saúde dos municípios brasileiros, tais como indicadores de saúde; assistência à saúde; epidemiologias e morbilidades; rede assistencial; informações demográficas e sociais; inquéritos e pesquisas e saúde suplementar (DATASUS, 2010). Tais informações são alimentadas pelos municípios, em caráter obrigatório, através dos softwares disponibilizados. Tais softwares seguem relacionados na tabela 1, juntamente com as informações que gerenciam e os tipos de bases de dados que utilizam.

Tabela1. Softwares disponibilizados pelo DATASUS

APLICATIVO	SIGLA	DADOS GERENCIADOS	TIPO DE BASE DE DADOS
Gerenciador de Informações Locais	GIL	Dados dos pacientes, prontuários médicos, produção médica.	FireBird
Cadastrador de Usuários do Sistema Único de Saúde	CADSUS	Informações individuais sobre os pacientes e seus domicílios.	Pode ser conectado a vários tipos de bases. Porém as usuais são FireBird e PostgreSQL.
Sistema de Informação da Atenção Básica	SIAB	Informações Cadastrais e de Produção das Equipes de Saúde da Família.	Bases DBF.
Sistema de Cadastramento e Acompanhamento de Hipertensos e Diabéticos	HIPERDIA	Informações cadastrais de hipertensos e diabéticos.	Interbase.
Sistema de Acompanhamento de Gestantes	SISPRENATAL	Cadastramento e acompanhamento de gestantes.	FireBird
Sistema de Controle de Internações Hospitalares	SISAIH	Informações de controle de internações e tratamentos hospitalares.	FireBird
Sistema de Informação sobre Mortalidade	SIM	Informações sobre mortalidades e suas causas.	DBF
Sistema de Informação de Nascidos Vivos	SINASC	Informações cadastrais e de acompanhamento de gestantes e recém nascidos.	FireBird e arquivos de texto.

A partir deste ponto, pode-se supor que os municípios, necessariamente, mantém a maioria dos dados sobre saúde pública, porém não conseguem visualizá-los por estarem distribuídos em bases de dados heterogêneas, conforme a Tabela 1, constituindo assim, para os gestores municipais de saúde, somente um conjunto de dados que não expressa informação alguma.

Além disso, algumas informações inerentes às necessidades locais, como abrangência das unidades de saúde, distribuição de profissionais de saúde, distribuição de medicamentos, distribuição sócio-econômica da população, abrangência de equipes do Programa da Saúde da Família (PSF) não estão disponíveis.

3. Descoberta de Conhecimento em Banco de Dados e Sistemas de Informações Geográficas aplicados à gestão de saúde pública

Existem duas ferramentas que podem ser empregadas na solução dos problemas expostos na seção anterior: Descoberta de Conhecimento em Banco de Dados (DCBD) e Sistemas de Informações Geográficas (SIG).

A Descoberta de Conhecimento em Banco de Dados constitui-se de um processo formado de várias etapas sucessivas e iterativas sobre os dados existentes. O processo de DCBD é “o processo não trivial de extração de informações implícitas, previamente desconhecidas e potencialmente úteis a partir dos dados armazenados em um banco de dados” (FAYYAD, 1996). Tal processo permite extrair conhecimento (padrões, tendências, relações entre eventos, associações entre dados, etc.) anteriormente desconhecido e que não está explicitamente apresentado ou que é de difícil inferência. A figura 1 mostra um modelo do processo de DCBD.

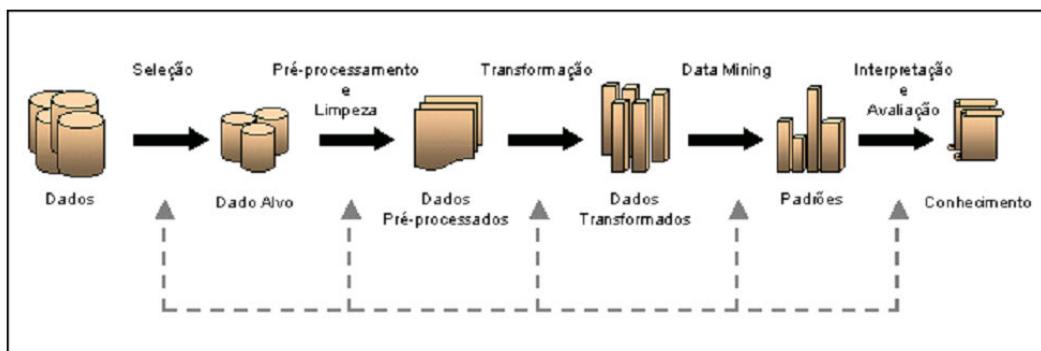


Figura 1 - O processo de DCBD (FAYYAD, 1996)

Um Sistema de Informações Geográficas (SIG) é “um conjunto de tecnologias de coleta, tratamento, manipulação e apresentação de informações espaciais” (COSTA, 2002). Estes sistemas têm uma capacidade de reunir grandes quantidades de dados sobre descrição geométrica, posicionamento espacial e características geográficas (dados espaciais), integrá-los e estruturá-los de tal forma que são vistos como ferramenta essencial na manipulação de dados espaciais (LEÓN, 2007). A figura 2 ilustra uma estrutura de um SIG, juntamente com um esquema representativo de sua organização em camadas.

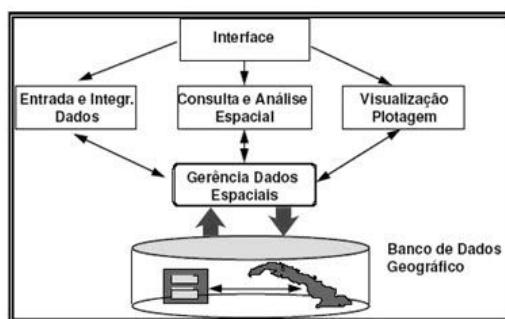


Figura 2 - Modelo de SIG (Qeuiroz; Ferreira, 2006 apud Ruiz, 2010)

Um SIG permite que dados sejam consultados com base em sua localização geográfica ou com base em seus atributos descritivos, ou ainda pela combinação deles. Além disso, dados espaciais permitem algumas operações adicionais àquelas possíveis com dados ditos convencionais, tais como proximidade, distância, vizinhança, entre outras. Um SIG permite, ainda, que o resultado de uma consulta seja visualizado espacialmente, plotando os dados em um mapa. Assim, a distribuição geográfica de um determinado elemento, como, por exemplo, uma doença, pode ser facilmente visualizada.

Desta forma, a aplicação de um processo de DCBD, aliado a uma ferramenta de geoprocessamento, poderia dar aos municípios a chance de entender a distribuição de população, inferir sobre distribuição de recursos, avanços epidemiológicos, áreas de risco, visualizar o perfil dos pacientes, bem como da população, enfim, todas as informações necessárias à gestão de saúde de forma eficiente e eficaz. Por exemplo, poderia se observar séries temporais de informações de hipertensos e diabéticos (extraídas a partir do processo de DCBD) distribuídos em áreas demarcadas em mapas gerados através de um SIG. Ou, através do mesmo processo, poderia ser visualizada a distribuição de atendimentos médicos.

Além disso, o uso conjunto de tais sistemas pode auxiliar na previsão de demandas futuras de diversos materiais e serviços, bem como na projeção de necessidades e de recursos.

3.1 Trabalhos relacionados

Alguns trabalhos já foram elaborados em outras áreas usando um processo semelhante ao proposto neste artigo, como o exposto em (SILVA; DAVIS JR; DAVIS, 2010), que propõe a utilização de etapas do processo de DCBD, mais especificamente integração e mineração de dados com o intuito de criar um conjunto de dados hidrográficos que contenha referências no tempo-espacó (dados geográficos) da cidade de Governador Valadares, com o objetivo de prevenir enchentes.

Também em (SKABA et al., 2004) é realizado um estudo dos eventos de epidemiologia unindo dados do Sistema de Informações de Agravos de Notificações (SINAN) com dados georreferenciados no registro de setores censitários do IBGE, visando uma análise dos eventos de saúde pública em centros urbanos.

4. Abordagem proposta

Para atingir os objetivos propostos neste artigo, o processo como um todo foi dividido em duas etapas. Primeiramente, foi executado um processo de DCBD sobre o conjunto de dados dos SISs do DATASUS, principalmente nos registros correspondentes à cadastro nacional, produção ambulatorial, epidemiologia, produção hospitalar, internações hospitalares e sobre dos dados de mortalidade infantil, nascidos vivos e ocorrência de óbitos. Sendo que os três últimos itens podem ser obtidos no *site* do DATASUS (www.datasus.gov.br). Nesta fase, foram obtidos os padrões, classificações, associações e inter-relações entre fatos e eventos que darão o retrato da saúde pública de um modo geral.

No que diz respeito ao pré-processamento, como os dados se originam de bases heterogêneas, e considerando o fato de que é impossível obter-se acesso direto às bases, e por isso existe a necessidade de geração de arquivos de exportação de dados, tornou-se indispensável a integração de dados.

4.1 Integração dos Dados

Para a tarefa de integração de dados a técnica escolhida foi Data Warehouse o que se justifica pela necessidade descrita anteriormente de se gerar arquivos de exportação. Estes arquivos são gerados pelos softwares DATASUS sempre em um formato de arquivo de dados denominado DBF. O Sistema de Gerenciamento de Banco de Dados (SGBD) utilizado foi o MySQL.

4.2 Limpeza dos Dados

No processo de limpeza dos dados, dadas as características e forma de organização dos mesmos, somente foi necessária a exclusão de alguns registros onde a maioria das informações que compõem o objetivo de trabalho, estavam ausentes. O motivo de não terem sido empregadas outras técnicas como Agregação, Seleção de Subconjunto de Características, Discretização e Transformação de Variáveis, é que os dados dispostos nas bases deste sistema (DATASUS) já seguem uma estrutura de categorização e escala. Tal estrutura está organizada de tal forma que nos registros armazenados nessas bases, somente pode-se encontrar números que correspondem a códigos de bairro, de municípios e de categorias como faixa etária, grau de instrução e outras. As informações reais estão disponibilizadas na forma de Arquivos de Conversão (.CNV), que são disponibilizados pelo site do DATASUS ou gerados pelos próprios softwares que compõem o sistema.

4.3 Mineração de Dados

Nesta etapa foram utilizadas as técnicas de classificação supervisionada e não-supervisionada para obter-se informações como distribuição de faixa etária dos pacientes das unidades de saúde, grau de instrução, sexo, estado nutricional, número de atendimentos a gestantes, turno de maior movimento (manhã, tarde), especialidade mais procurada, entre outras. Para tanto utilizou-se o software livre weka. Abaixo, a figura 3 mostra um exemplo de resultado obtido.

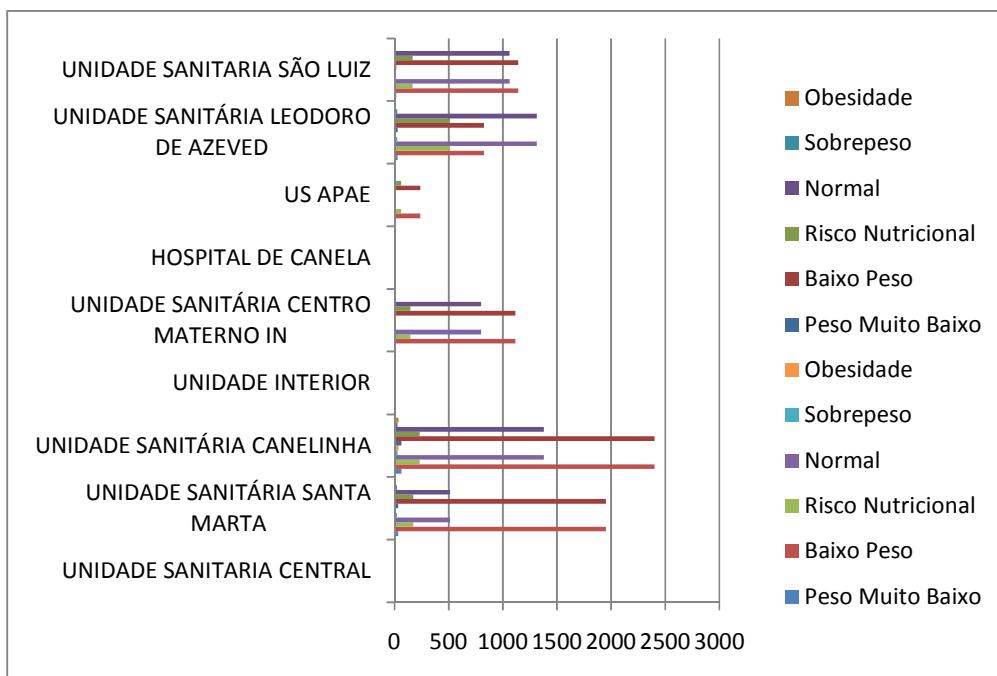


Figura 3 - Gráfico com informações sobre o estado nutricional por UBS.

Na figura acima, observa-se, além do perfil nutricional dos pacientes de cada unidade sanitária, prováveis falhas no registro de informações por parte da Unidade Sanitária Central e do Hospital de Canela.

Na segunda etapa, o conjunto de dados resultante passou pelo processo de integração com dados georreferenciados. Tal integração gerou um novo conjunto de dados composto pelas informações resultantes da etapa anterior distribuídas de forma espacial, tornando-se passíveis de processamento por SIGs. Foi nesta etapa em que foram construídos mapas necessários à visualização da distribuição no espaço dos recursos, necessidades e eventos de saúde pública de uma população, bem como a própria distribuição populacional. Segue abaixo, na figura 4, uma mapa construído a partir da sobreposição das áreas de abrangência das unidades de saúde (mapa construído através de informações do município) com os setores censitários (construído com informações disponibilizadas pelo IBGE na URL ftp://ftp.ibge.gov.br/Censos/Censo_Demografico_2000/Dados_do_Universo/Agregado_por_Setores_Censitarios) e aplicação de um geoprocessamento denominado União Espacial, que cruzou os dados entre os referidos mapas usados de base.

Para a execução desta fase do processo foi utilizado o software de geoprocessamento gvSIG distribuído de forma gratuita através do portal gvSIG em <http://www.gvsig.org>. Este software pode conectar-se a várias bases de dados, inclusive PostgreSQL e ao seu complemento PostGIS suportando diferentes tipos de banco de dados espaciais. Entretanto, neste trabalho a forma de armazenamento de dados utilizada foi o armazenamento dos atributos da geometria (polígonos, linhas, pontos, etc.) em *shape file*, enquanto os atributos de dados foram armazenados em arquivos do tipo DBF. Tal escolha deu-se para se evitar conversões desnecessárias para outras bases de dados, visto que os softwares do sistema DATASUS somente exportam seus dados para arquivos DBF.

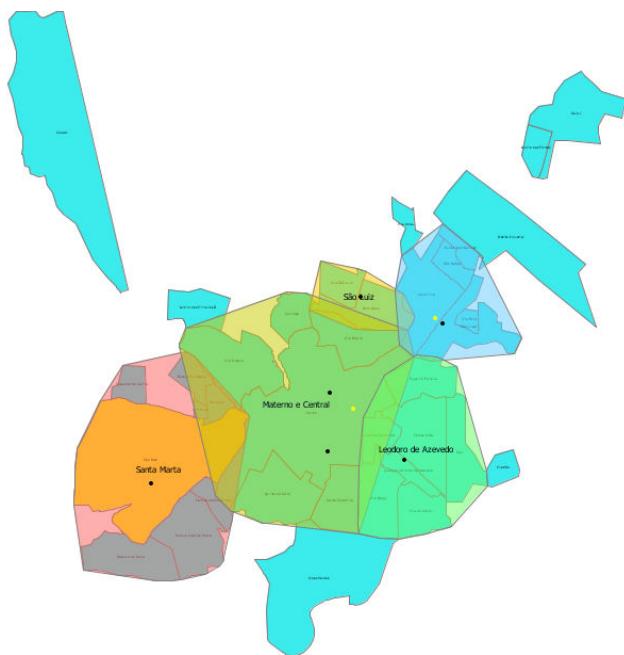


Figura 4 - Mapa de Abrangência das Unidades de Saúde do Município de Canela

Na figura acima, além da visão da abrangência das unidades de saúde (pontos pretos) com relação aos bairros, ficou claro que alguns bairros estão sem cobertura e/ou estão situados muito distantes da unidade de saúde mais próxima. Como mencionado acima, a operação de União Espacial gerou uma gama enorme de informações populacionais pertinentes a várias áreas do conhecimento. Abaixo encontra-se a Tabela 2 com um pequeno fragmento de tais informações, sendo que o significado de cada uma das variáveis constantes (V0036, V0037, V0048, V0051, V0052 e V0054, encontram-se na Tabela 3.

Tabela 2. Dados de domicílios produzidos através de geoprocessamento.

Nome	V0036	V0037	V0048	V0051	V0052	V0054
Santa Marta	1539	1536	232	3	2	3
Canelinha	1557	1548	232	9	2	2
Leodoro de Azevedo	3514	3507	466	7	0	0
Materno e Central	5993	5980	778	13	1	0
São Luiz	1703	1695	247	8	2	2
Total	14306	14266	1955	40	7	7

Tabela 3. Descrição das variáveis da Tabela 2.

Nome da	Descrição

Variável	
V0036	Domicílios particulares permanentes sem banheiro, nem sanitário
V0037	Domicílios particulares permanentes com banheiro
V0048	Domicílios particulares permanentes com lixo coletado
V0051	Domicílios particulares permanentes com lixo queimado na propriedade
V0052	Domicílios particulares permanentes com lixo enterrado na propriedade
V0054	Domicílios particulares permanentes com lixo jogado em rio, lago ou mar

Alguns dos dados necessários ao georreferenciamento como as áreas de cobertura das unidades básicas de saúde, bem como identificação das mesmas juntamente com dados de capacidade, puderam ser obtidos também nos registros e documentos do próprio município.

5. Considerações Finais

O Brasil, tanto por sua extensão, quanto pelas diferentes características de cada estado, região e até mesmo entre os vários municípios de uma mesma região, possui muitas divergências em termos de necessidades em saúde. Por exemplo, no Acre, pode-se dizer que a prioridade em termos de saúde pública é combater e controlar doenças como a malária e a dengue. Por outro lado, no caso do Rio Grande do Sul, não é possível afirmar o mesmo. Se fosse realizada uma investigação detalhada, provavelmente se chegaria à conclusão de que municípios vizinhos como Canela e Gramado (região serrana do Rio Grande do Sul) têm necessidades de saúde diferentes, devido ao perfil populacional.

Desta forma, fica explícita a importância da racionalização no uso dos recursos de saúde pública de forma a distribuí-los visando eficiência, eficácia e abrangência. Pois tal racionalização, além dos benefícios gerados à população como planos de prevenção e promoção de saúde coletiva, pode gerar economias em nível municipal, estadual e federal.

Como exposto anteriormente, o presente artigo propõe uma forma de se atingir tais objetivos. Entretanto, alguns fatores devem ser levados em consideração, como o custo, principalmente no que diz respeito aos SIGs, ao levantamento dos dados necessários do georreferenciamento. Outro fato importante a ser considerado quando se pretende implantar ferramentas como estas, é que elas produzem resultados a longo prazo, e devem ser entendidas como um processo contínuo e infinito de atualização de informações.

É importante mencionar também que neste trabalho foi utilizado como exemplo apenas uma pequena parcela dos dados, análises e informações possíveis, pois a gama de informações geradas neste tipo de processo pode ser extremamente variada.

REFERÊNCIAS

COSTA, Giseli Fernandes da. **Geoprocessamento: Uso e Aplicação na Saúde Pública e na Saúde Ambiental.** I Encontro Associação Nacional de Pós Graduação e Pesquisa em Ambiente e Sociedade. Indaiatuba – SP. 2002. Disponível em:

<http://www.anppas.org.br/encontro_anual/encontro1/gt/sustentabilidade_cidades/Gisel%20Fernandes%20da%20Costa.pdf>. Acesso em: 26/11/2010.

DATASUS. **Departamento de Informática do SUS.** Disponível em <<http://www2.datasus.gov.br/DATASUS/index.php>> . Acesso em 01/09/2010.

DRUMOND, Marcos Jr. **Epidemiologia nos Municípios Muito Além das Normas.** São Paulo, SP: Hucitec, 2003.

LEÓN, Maria Eliane dos Santos. Dissertação (Mestrado do Programa de Pós-Graduação em Geomática) Universidade Federal de Santa Maria. Santa Maria, RS, Brasil, 2007. Disponível em: <http://cascavel.cpd.ufsm.br/tede/tde_arquivos/21/TDE-2007-05-18T142757Z-584/Publico/leon.pdf>. Acesso em 26/11/2010.

PRASS, Fernando Sarturi. **Processo de Descoberta de conhecimento em Bancos de Dados.** Grupo de Interesse em Engenharia de Software. Florianópolis. 2004. v. 1, p. 10 – 14. Disponível em <http://www.google.com.br/#hl=pt-BR&biw=1345&bih=574&q=Fernando+Sarturi+Prass+%2B+KDD&aq=f&aqi=&aqi=l&oq=&gs_rfai=&fp=111e00860681258d>. Acesso em: 26/11/2010.

REDE INTERNACIONAL DE INFORMAÇÕES PARA A SAÚDE. **Sistemas de Informação Geográfica e a Gestão da Saúde no Município.** [S.l.], [200-]. Disponível em <<http://www.bvsde.paho.org/bvsacd/cd29/sig-saudade.pdf>>. Acesso em: 30/11/2010.

RUIZ, Luis Fernando Chimelo. **Estudo de Caso: Uma Aplicação de Sistemas de Informação Geográfica em Imobiliárias.** Universidade Federal de Santa Maria. 2010. Disponível em <<http://tecgeoprocessamento.blogspot.com>>. Acesso em 12/11/2010.

SANTOS, Fausto Pereira dos. **Sistema Único de Saúde em Belo Horizonte: Reescrevendo o público.** Organizada por Cezar Rodrigues Campos; Deborah Carvalho Malta; Afonso Teixeira dos Reis; Aleneir de Fátima dos Santos; Emerson Elias Merhy. 1. ed. São Paulo, SP: Xamã, 1998.

SILVA, Petrônio C. de L. e; DAVIS JR., Clodoveu A.; DAVIS, Elizabeth G. **Descoberta de Conhecimento em Bancos de Dados Espaço-Temporais para Previsão de Risco Hidrológico.** Anais do XXX Congresso da Sociedade Brasileira de Computação. Belo Horizonte, MG. 2010. Disponível em <http://www.inf.pucminas.br/sbc2010/anais/pdf/wcama/st02_01.pdf>. Acesso em: 26/11/2010.

SKABA et al. **Geoprocessamento de dados da saúde: o tratamento dos endereços.** Caderno de Saúde Pública. Rio de Janeiro. 2004. v. 20, n. 6. Disponível em <<http://www.scielo.br/pdf/csp/v20n6/37.pdf>>. Acesso em: 26/11/2010.

Uma Ferramenta Para Análise Quantitativa da Produção Científica de Pesquisadores

Jardel Gugel¹, Cristiano R. Cervi^{1,2}, Renata Galante², José Palazzo M. de Oliveira²

¹Instituto de Ciências Exatas e Geociências – Universidade de Passo Fundo (UPF)
Caixa Postal 611 – 99001-970 – Passo Fundo – RS – Brasil

²Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brasil

jardel_internacional@hotmail.com, cervi@upf.br, {galante, palazzo}@inf.ufrgs.br

Resumo. As redes de colaboração científica têm despertado o interesse da comunidade científica mundial. Elas conseguem representar problemas de maneira objetiva, oportunizando um estudo dos diversos tipos de relacionamentos entre grupos, entidades e pessoas. No campo da ciência da computação as redes de colaboração científica vêm sendo utilizadas para analisar e identificar as tendências sobre as publicações de um pesquisador, quantidade de publicações ou pessoas vinculadas a pesquisadores. Este trabalho apresenta uma ferramenta que possibilita a análise quantitativa da produção científica de pesquisadores. Essa análise é proveniente das informações disponibilizadas pela DBLP, através de um documento XML.

1. Introdução

Atualmente, sistemas podem ser representados e problemas podem ser tratados através da abordagem de rede. Um grupo de pessoas em uma organização trocando e-mails pode ser interpretado como uma rede social, onde cada pessoa passa a ser um ator e as mensagens eletrônicas trocadas passam a ser os laços da rede. O entendimento das redes, de sua estrutura, características e comportamento são fundamentais para a compreensão das diversas classes de sistemas e problemas que podem ser por elas modelados e tratados (Brandão, Parreiras e Silva 2007).

Uma das áreas de aplicação de redes sociais pode ser a descoberta das vinculações existentes entre autores de artigos, ou ainda alguma correlação que os mesmos possuem sobre algum trabalho publicado. Dessa forma é possível identificar pesquisadores em potencial ou mesmo sugerir pesquisadores que possuem uma mesma área de interesse. Neste contexto, uma rede dessa natureza pode ser chamada de rede de colaboração científica. Um exemplo deste tipo de rede, onde existem informações sobre artigos publicados na área da Ciência da Computação é a biblioteca digital DBLP¹ (DBLP 2010). A DBLP disponibiliza informações sobre as publicações de pesquisadores, sendo possível encontrar informações relacionadas a cada publicação cadastrada, como título, conferência ou periódico onde foi publicada, nome de autores e coautores, bem como ano de publicação.

Este artigo aborda o processo de desenvolvimento de uma ferramenta e verificação de seus resultados para a análise quantitativa da produção científica de pesquisadores. Além disso, mostra o processo de transformação das informações brutas em informações resultantes de consultas realizadas.

O trabalho está dividido como segue. A seção 2 apresenta os trabalhos relacionados. A abordagem proposta, com a metodologia, critérios de classificação, forma de aquisição dos dados e a ferramenta desenvolvida é detalhada na seção 3. A seção 4 apresenta os experimentos realizados e as análises dos resultados obtidos. Finalmente, na seção 5, são apresentadas as considerações finais e esboçados trabalhos futuros.

¹ DBLP – Digital Bibliography & Library Project é uma biblioteca digital que armazena informações sobre as principais publicações da área da ciência da computação, tanto de periódicos como de eventos.

2. Trabalhos Relacionados

Diversos trabalhos existentes na literatura abordam questões sobre produção científica de pesquisadores, identificação de perfis, extração de dados da web para modelagem destes perfis, identificação de especialistas baseados em dados científicos ou simplesmente análise de produção para obter estatísticas relevantes. Nesse contexto se destacam as redes sociais de colaboração científica. Estas redes mostram como os pesquisadores se relacionam através do desenvolvimento de trabalhos e publicações conjuntas.

O modelo proposto por Hope, Nishimura e Takeda (2006) visa integrar dados obtidos de várias redes sociais de pesquisadores através de três abordagens: 1) utilização de técnicas de mineração, onde se podem criar redes iniciais automaticamente através de informações disponíveis na web; 2) utilização da interação do usuário com o mundo real, com a obtenção de modelos de redes sociais com a captura das interações do usuário; e 3) utilização da descoberta de interações do usuário em sistemas web. A descoberta de redes sociais científicas é discutida em alguns trabalhos (TANG et al., 2008a; TANG et al., 2008b; TANG, ZHANG e YAO, 2007), no qual a descoberta é realizada principalmente através da extração de informações da web e de bibliotecas digitais sobre a produção científica de pesquisadores. Outros trabalhos utilizam dados sobre a produção científica para encontrar especialistas em determinada área do conhecimento (ZHANG, TANG e LI, 2007; MIMNO e MCCALLUM, 2007; FU et al., 2007; LI et al., 2007). Todos esses trabalhos usam informações coletadas na web, em páginas pessoais dos pesquisadores e em diferentes bibliotecas digitais, como DBLP e CiteSeer (CITESEER 2010).

Redes de colaboração científica entre pesquisadores são apresentadas e discutidas em Liu et al (2005), onde são agrupados pesquisadores e verificados como é a relação entre eles. Nesse mesmo contexto, trabalhos como os de Chen et al. (2008) e Huang et al. (2008) analisam redes de coautoria através da definição de grafos e suas relações. Menezes et al. (2009) estuda redes de coautoria na área de ciência da computação através de 30 programas de pós-graduação em computação (8 no Brasil, 16 na América do Norte e 6 na Europa). Neste trabalho foram coletados e analisados dados de, aproximadamente, 170.000 pesquisadores, 350.000 publicações e 2.000 veículos de publicação. Os autores analisaram também a evolução temporal de redes sociais dos pesquisadores nestas diferentes regiões. Os dados foram obtidos da DBLP tendo-se como intervalo um período de 12 anos.

Em uma abordagem para análise estatística, a produção em computação também é apresentada em alguns trabalhos, como Medeiros (2008), Vardi (2009) e Wainer et al. (2009). Wainer, Xavier e Bezerra (2009) apresentam dados da produção científica de pesquisadores mediante publicações em periódicos e conferências. Utilizam dados indexados por ISI² (*Institute for Scientific Information*) e Scopus³ para comparar as publicações de pesquisadores brasileiros com pesquisadores de outros países, separados por área. Arruda et al. (2009) analisam a produção de pesquisadores brasileiros mediante dados de seus currículos Lattes com o objetivo de classificação regional e gênero.

3. Ferramenta Desenvolvida

As subseções abaixo descrevem como ocorreu o desenvolvimento da ferramenta, objetivos e as tecnologias empregadas no seu desenvolvimento.

3.1. Objetivo e Funcionalidades da Ferramenta

O objetivo da ferramenta é promover consultas e cruzamento de dados sobre informações disponibilizadas pela DBLP por meio de um arquivo XML. As informações são utilizadas para análise quantitativa da produção científica de pesquisadores. Essa base de dados armazena informações referentes a artigos publicados nas principais conferências e periódicos da área de ciência da computação.

Através das mais variadas formas de busca disponibilizadas pela ferramenta é possível encontrar diversas informações sobre os autores cadastrados na biblioteca digital DBLP. As informações permitem a

² Disponível em <http://isiwebofknowledge.com>

³ Disponível em <http://www.scopus.com>

visualização da lista contendo as publicações de um autor, os coautores destas publicações, bem como o ano e o veículo da publicação. Uma das formas disponíveis de pesquisa é pelo ano, onde as publicações do ano pesquisado são mostradas pela ferramenta, juntamente com os nomes dos autores envolvidos com as mesmas. Ainda, é possível buscar dados através do título de uma publicação. Se a mesma for encontrada, é apresentada uma lista de autores da publicação e seu ano.

Os métodos de pesquisa disponibilizados não visam saber os nomes das publicações nem nomes de coautores. O objetivo é disponibilizar formas de visualização gráfica sobre o número de publicações, coautores e, consequentemente, visualizar a evolução temporal quantitativa que os autores cadastrados obtiveram ao longo da sua vida acadêmica, verificando ano a ano o número de publicações e coautores envolvidos.

Também há métodos para ver a evolução da quantidade de publicações e do aumento do número de coautores envolvidos ao longo dos anos, independentemente de um determinado autor. O próprio usuário é quem define o intervalo dos anos em que o gráfico será construído e apresentado.

A ferramenta disponibiliza, ainda, métodos de comparações para definir os autores que mais publicaram artigos na DBLP, sendo que o valor é definido pelo próprio usuário. Subentende-se que os autores selecionados são os mais prestigiados autores devido ao grande número de publicações. Também é possível comparar autores que publicaram mais que um determinado número de publicações em determinado ano.

Existem ainda métodos de busca para comparar os autores em que o total de coautores seja maior que um determinado valor. Subentende-se dessa forma que os coautores possuem uma maior rede de colaboração, sendo possível pesquisar autores que utilizaram um maior número de colaboradores dentro de determinado ano.

3.2. Tecnologias Utilizadas

Para consultas simples optou-se por um Sistema Gerenciador de Banco de Dados XML Nativo (SGBDXN) de código livre chamado Sedna, desenvolvido e mantido pela ISPRAS⁴. Este possui todas as características de um SGBDXN, que vão desde o armazenamento das informações, segurança de acesso, transações ACID⁵, triggers, indexação, controle de concorrência, dentre outros. Além destas importantes características, o Sedna foi escolhido por conseguir gerenciar um grande volume de dados, requisito essencial para este trabalho.

Para efetuar as consultas das informações contidas no banco de dados XML utilizou-se o Xquery⁶. Para o desenvolvimento da interface da ferramenta foi utilizada a linguagem de programação JAVA⁷, juntamente com a API DOM⁸. Como os dados retornados de cada consulta estão em uma estrutura de *tags* XML foi necessária a utilização da API DOM para remoção destas *tags*. A API define uma forma padronizada para manipulação de documentos XML e transforma o documento em uma estrutura de árvores, que pode ser acessada utilizando-se de um conjunto de objetos disponíveis pela API. Isto permite a leitura dos dados contidos no retorno da consulta, sendo possível a extração dos dados das *tags*, sua validação e apresentação ao usuário.

Para consultas quantitativas optou-se por um Sistema Gerenciador de Banco de Dados Objeto-Relacional (SGBDOR), o PostgreSQL⁹. Ele suporta o padrão SQL 2003 e pode implementar linguagens procedurais como o PL/pgSQL, que possibilita a criação de funções para cálculos, verificações, validações, dentre outras. As *procedures* podem ser implementadas diretamente no banco de dados e podem ser executadas antes ou depois da leitura ou escrita de informações dentro do banco de dados.

⁴ Disponível em <http://www.modis.ispras.ru/sedna/>

⁵ São propriedades fundamentais nas transações de que garantem a consistência das informações

⁶ Disponível em http://www.w3schools.com/xquery/xquery_reference.asp

⁷ Disponível em <http://www.oracle.com/technetwork/java/index.html>

⁸ Disponível em <http://www.w3.org/DOM/>

⁹ Disponível em <http://www.postgresql.org/>

3.3. Descrição da Ferramenta

As informações disponibilizadas na biblioteca digital estão em um único arquivo no formato XML. Não há ordenação do arquivo e alguns dos registros estão inconsistentes, ou seja, faltam dados para utilização. Devido a este problema, antes da utilização dos dados foi necessária uma verificação dos mesmos, eliminando-se itens inválidos, nulos ou vazios. O arquivo XML é constituído por uma série de *tags* que organizam e armazenam as informações referentes a cada publicação, como o título, autores envolvidos, ano, veículo de publicação, dentre outros. Algumas *tags* armazenam o endereço *web* onde o trabalho pode ser encontrado. O Quadro 1 apresenta um pequeno exemplo da disposição dos dados dentro do arquivo XML e suas respectivas *tags*.

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no"?>
<!DOCTYPE dblp SYSTEM "dblp.dtd">
<dblp>
    <proceedings mdate="2004-07-30" key="conf/3dica/1998">
        <editor>Richard N. Ellison</editor>
        <editor>Joseph H. Nurree</editor>
        <title>Proceedings of the Conference on Three-Dimensional Image Capture and Applications, San Jose, CA, USA, January 27-28, 1998</title>
        <year>1998</year>
    </proceedings>
    <inproceedings mdate="2004-07-30" key="conf/3dica/AzumaUM99">
        <author>Takeo Azuma</author>
        <author>Kenya Uomori</author>
        <author>Atsushi Morimura</author>
        <title>Real-time Active Range Finder Using Light Intensity Modulation.</title>
        <year>1999</year>
    </inproceedings>
    <article mdate="2005-11-15" key="journals/4or/WerraH05">
        <author>Dominique de Werra</author>
        <author>Pierre Hansen</author>
        <title>Variations on the Roy-Gallai theorem.</title>
        <year>2005</year>
        <journal>4OR</journal>
    </article>
</dblp>
```

Fonte: DBLP

Quadro 1. Estrutura do arquivo XML da DBLP.

Por meio do arquivo XML é feita a transposição dos dados que nele estão contidos para o banco de dados Sedna. Desta forma, a ferramenta conecta-se à base de dados, realiza consultas, faz o cruzamento de dados e mostra os resultados ao usuário.

Sobre as consultas no banco XML da ferramenta, o usuário as realiza utilizando-se de uma interface que se comunica com o banco de dados. A partir desta interface o usuário pode realizar as consultas disponibilizadas pela ferramenta ou estruturar as suas próprias. A Figura 1 representa o funcionamento da ferramenta para a pesquisa em XML.

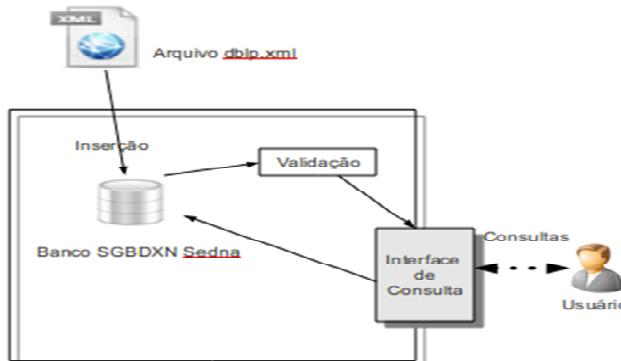


Figura 1. Funcionamento da ferramenta.

Quando são efetuadas consultas na base XML, os dados retornados estão contidos entre as *tags* XML, sendo assim, há a necessidade destes dados passarem por uma etapa de extração e de validação.

A etapa de extração e de validação dos dados é definida em um método. Ele recebe um trecho de código em XML, que é o resultado da consulta aos dados. Após, é criado um documento em memória, onde o mesmo é manipulado pela API DOM. Desta forma, é efetuada a extração dos dados contidos nas *tags*. Após a retirada dos dados é realizada uma validação sobre os mesmos, ou seja, se não há itens nulos ou vazios. Na seqüência, os dados são adicionados a uma classe chamada *Publicacao*, que contém todos os dados relativos àquela publicação, como o nome da publicação, a lista de autores e o ano.

Para a realização das consultas quantitativas foram transportados os dados contidos na base XML para o banco de dados relacional PostgreSQL, tendo em vista que as consultas efetuadas diretamente sobre esta base XML seriam muito onerosas.

O cadastramento dos dados no banco relacional ocorre da seguinte forma: (i) são selecionados todos os autores cadastrados no banco de dados XML e cadastrados na tabela *author*, inicializando-se o número total de publicações de cada autor com o valor zero; (ii) são cadastrados os dados na tabela *ano*; (iii) é pesquisado cada autor cadastrado no banco de dados onde o total de publicações é igual a zero. Com o resultado dessa pesquisa o nome do autor é pesquisado na base de dados XML. O retorno das publicações é armazenado em uma lista da classe *Publicacao*; (iv) são realizadas as contagens de publicações e de coautores para serem cadastradas no ano em que ocorreram; (v) são realizados ajustes do total de publicações de cada autor, onde sabe-se que os autores em que o total de publicações era diferente de zero já tinham sido pesquisados na base de dados XML, estando devidamente cadastrados no banco relacional.

4. Experimentos e Análise de Resultados

Para verificar se a ferramenta atendia os princípios especificados diversos experimentos de consultas foram realizados. Entre as consultas, algumas retornam alguns dados estatísticos, outras são consultas diretas. Todas elas podem ser visualizadas no Quadro 2.

- 1) Quantos artigos cada pesquisador publicou.
- 2) Quantidade de artigos por ano cada pesquisador possui.
- 3) Quantidade de autores existente.
- 4) Somatório de publicações por ano.
- 5) Título da publicação e nome dos autores de um determinado ano.
- 6) Título da publicação e colaboradores de um autor.
- 7) Evolução do número total de publicações ao longo dos anos.
- 8) Evolução quantitativa de cada pesquisador.
- 9) Média quantitativa de publicações e colaboradores ao longo dos anos.
- 10) Quantidade de autores que publicaram um mesmo número de publicações.
- 11) Percentual de autores que publicaram um mesmo número de publicações.
- 12) Quais autores tiveram os maiores números de publicação.
- 13) Quais autores tiveram os maiores números de publicação por ano.
- 14) Quais autores tiveram as maiores redes sociais.
- 15) Quais autores tiveram as maiores redes sociais por ano.

Quadro 2. Perguntas a serem respondidas pela ferramenta.

A Figura 2 apresenta o resultado referente à pergunta 5 do Quadro 2. Ao informar o ano e executar a opção *Pesquisar*, são mostradas as publicações daquele ano, com seu título e autores.

Seleciona Publicações do ano	
Digite o ano da publicação ex: 1999	2000
Titulo	Autores
A TECHNIQUE FOR COMPUTING WATERMARKS FR...	Chin-Chen Chang, Chwei-Shyong Tsai,
LIFTEZ LES SYLOWSI UNE SUITE "SOUS-GROUPES" P...	Bruno Poizat, Frank O. Wagner,
NEWLINES AND LEXER STATES.	Chris Clark,
LIST SCHEDULING OF GENERAL TASK GRAPHS UN...	Tomasz Kalinowski, Iordan Kort, Denis Trystram,
BINARY SEARCHING WITH NONUNIFORM COSTS A...	Gonzalo Navarro, Ricardo A. Baeza-Yates, Eduar...
SAN FRANCISCO PERFORMANCE: A CASE STUDY I...	A. Ralph Christ, Steven L. Halter, Kenton Lynne, St...
PANORAMIC IMAGE PROCESSING AND ROUTE NA...	Thomas Rfer,
HIGH-PERFORMANCE ROUTING IN NETWORKS OF...	Federico Silia, Jos Duato,
THE WEIGHT HIERARCHY OF PRODUCT CODES	Hans Georg Schaathun,
DESCRIPTIONAL COMPLEXITY OF DETERMINISTIC ...	Martin Kappes,
EDITORIAL: SPECIAL ISSUE IN HONOR OF JOHN RIC...	Ronald F. Boisvert, Wayne R. Dyksen, Elias N. Hou...
IMPROVING THE ODDS IN DISCRIMINATING "DRUG..."	Thomas M. Frimurer, Robert Bywater, Lars Nrum, ...
ESTIMATING DRUG/PLASMA CONCENTRATION LE...	Kristin M. Tolle, Hsinchun Chen, Hsiao-Hui Chow,
TOP 10 HARDWARE PRODUCTS OF 1999	Carl Machover,
AN ALGEBRAIC PROGRAMMING STYLE FOR NUME...	T. B. Dinesh, Magne Haveraaen, Jan Heering,
POLYNOMIAL TIME APPROXIMATION OF DENSE WE...	Wenceslao Fernandez de la Vega, Marek Karpinski,
UNIVERSAL DATA COMPRESSION BASED ON TIE D...	Dernhard Dalkenhol, Stefan Kuntz,
EFFECTIVE IMAGE SEGMENTATION WITH FLEXIBLE...	Odemir Martinez Bruno, Luciano da Fontoura Cost...
GRBNER BASES APPLIED TO FINITELY GENERATED...	Jrn Mller-Quade, Rainer Steinwandt,
EVALUATION OF ABSTRACTS BASED ON SURFACE I...	Yuji Ichioka, Tsuyoshi Yamamura, Yuji Sagawa, No...
WEIGHTED ERROR ESTIMATES FOR FINITE ELEM...	Heribert Rium, Franz-Theo Suttmeier

Foram encontradas 43379 publicações no ano de 2000

Figura 2. Lista de publicações por ano, com título da publicação e autores.

A Figura 3 apresenta o resultado da pergunta 6 do Quadro 2. Ao informar o nome do autor e executar a opção Pesquisar, são mostradas as publicações do autor, o título da publicação, seu ano e o nome dos coautores (se existirem).

Publicações do Autor	
Autor	Pesquisar
Jehoshua Bruck	
colaboradores -- Michael Langberg -- Alexander Sprintson	
Titulo = DEPTH-EFFICIENT NEURAL NETWORKS FOR DIVISION AND RELATED PROBLEMS., Ano de Publicação = 1993	
colaboradores -- Kai-Yeung Siu -- Thomas Kailath -- Thomas Hofmeister	
Titulo = X-CODE: MDS ARRAY CODES WITH OPTIMAL ENCODING., Ano de Publicação = 1999	
colaboradores -- Liqao Xu	
Titulo = LOW-DENSITY MDS CODES AND FACTORS OF COMPLETE GRAPHS., Ano de Publicação = 1999	
colaboradores -- Liqao Xu -- Vasken Bohossian -- David G. Wagner	
Titulo = CONSTRAINED CODES AS NETWORKS OF RELATIONS., Ano de Publicação = 2008	
colaboradores -- Moshe Schwartz	
Titulo = OPTIMAL UNIVERSAL SCHEDULES FOR DISCRETE BROADCAST., Ano de Publicação = 2008	
colaboradores -- Michael Langberg -- Alexander Sprintson	
Titulo = SHORTENING ARRAY CODES AND THE PERFECT 1-FACTORIZATION CONJECTURE., Ano de Publicação = 2009	
colaboradores -- Vasken Bohossian	

Figura 3. Publicações de um determinado pesquisador.

Outra forma de visualização das informações resultantes da pesquisa anterior é apresentada na Figura 4. Respondendo a pergunta 8 e a pergunta 1 do Quadro 2, é apresentada graficamente a evolução da quantidade de publicações e do número de colaboradores ao longo dos anos, sendo que no título é mostrado o total de publicações.

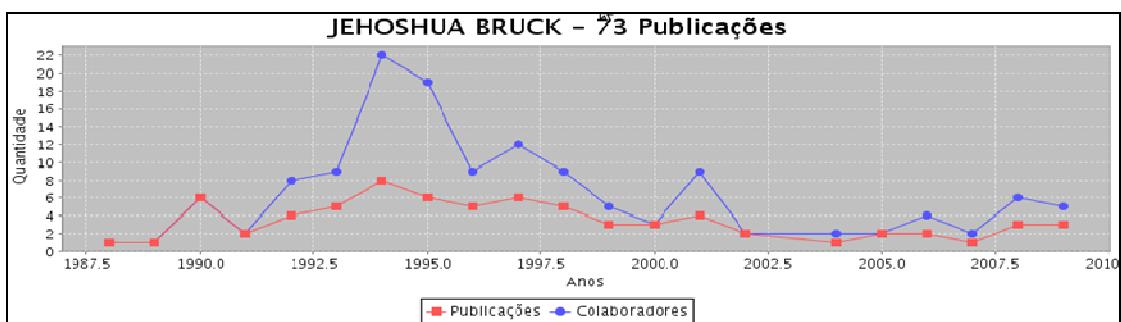


Figura 4. Evolução temporal quantitativa de um pesquisador.

Respondendo à pergunta 7 do Quadro 2 é apresentado graficamente na Figura 5 a evolução quantitativa do total de publicações e colaboradores ao longo dos anos. Esta consulta é independente de autor e apenas é selecionado o intervalo de anos que o gráfico será construído.

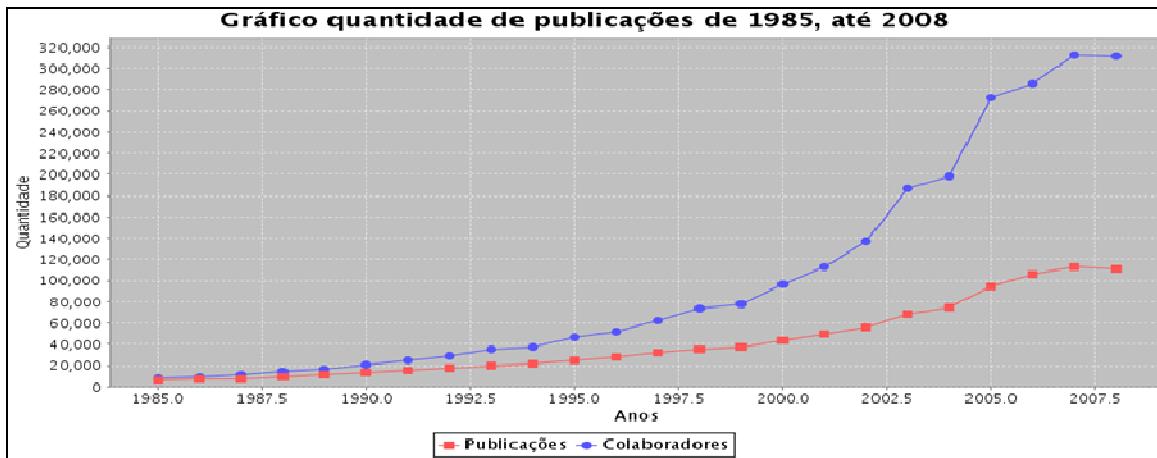


Figura 5. Evolução temporal quantitativa em um período.

A Figura 6 apresenta o resultado referente à pergunta 10 do Quadro 2. É realizado o somatório de autores que publicaram um mesmo número de publicações. Pode-se observar que 220.113 autores publicaram apenas um artigo, o que corresponde a, aproximadamente, 50% do número total de pesquisadores da DBLP.

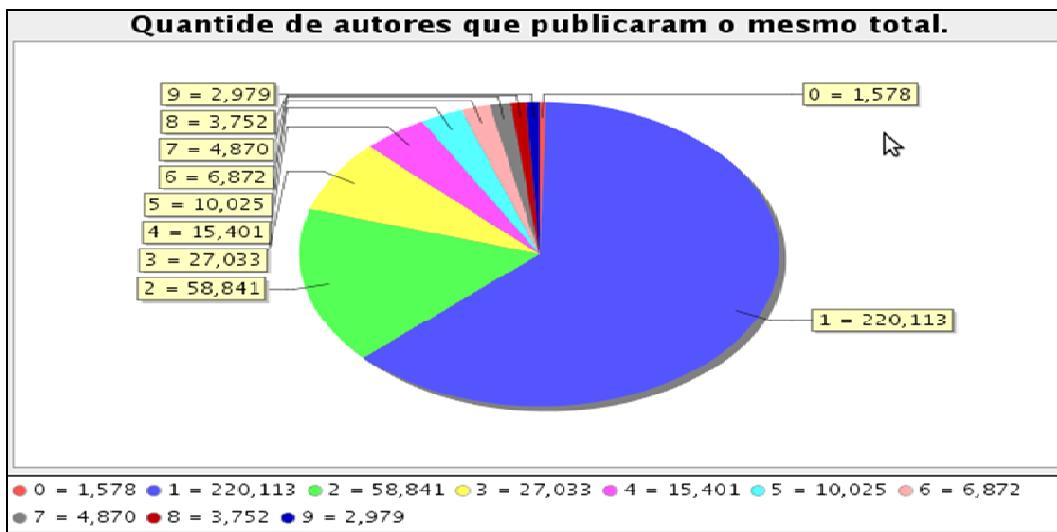


Figura 6. Somatório de autores que publicaram o mesmo número de artigos.

Na Figura 7 é apresentado o resultado referente à pergunta 9 do Quadro 2. Os cálculos são feitos da seguinte forma: é somada a quantidade de autores que publicaram no ano e dividida pela quantidade de autores, assim é obtida a média de autores. Da mesma forma, ocorre para descobrir a média de colaboradores: selecionam-se todos os colaboradores e divide-se pelo total de publicações. Assim, é verificado um pequeno aumento da média de publicações por ano. Em relação aos colaboradores, foi constatado um grande aumento ao longo dos anos, crescendo o tamanho da rede de colaboração.

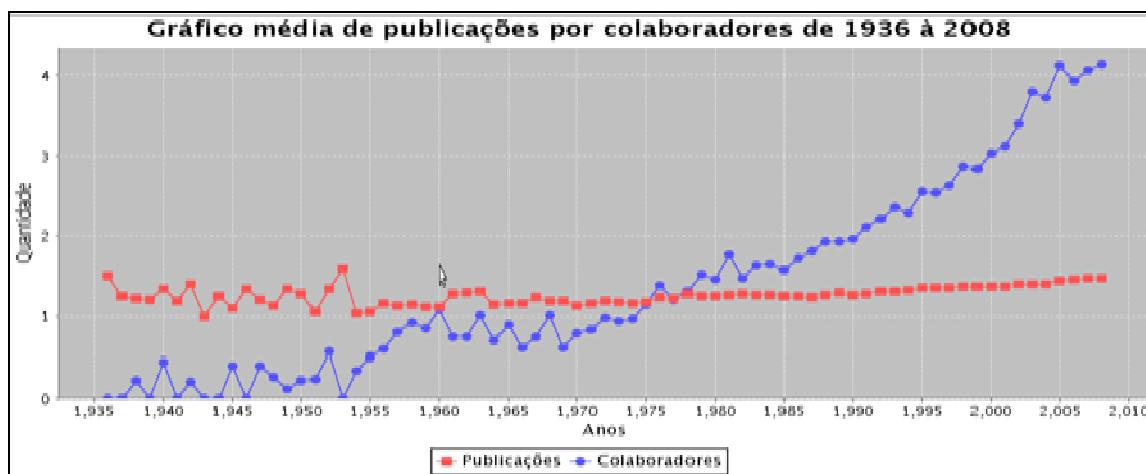


Figura 7. Média quantitativa de publicações e colaboradores ao longo dos anos.

Na Figura 8 é apresentado o resultado referente à pergunta 15 do Quadro 2. São selecionados todos os autores que publicaram com mais de 30 coautores em um determinado ano. Além do nome dos autores, é apresentado o total de publicações do ano, quantidade de coautores e o ano em que o autor publicou com mais que 30 coautores. Ao final da tabela é apresentado o somatório de vezes.

Maior Rede de Colaboradores				
Author	Total de Publicações	Total de Colaborad...	Ano	
Peter Willett 0002	8	34	2006	▲
Gad Ariav	3	35	1994	▼
Jie Chen	5	43	2005	▼
Suraj Peri	2	43	2004	▼
Allan Dickerman	1	42	2007	▼
J. Dana Eckart	1	42	2007	▼
Bruno W. S. Sobral	1	42	2007	▼
Roberto H. Higa	4	33	2004	▼
Roberto C. Togawa	4	33	2004	▼
Paula R. Kuser	4	33	2004	▼
F. A. Kolpakov	3	35	1999	▼
Olga A. Podkolodnaya	3	40	1999	▼
T. N. Goryachkovskaya	2	34	1999	▼
Irina L. Stepanenko	2	34	1999	▼
Nikolay L. Podkolodny	3	40	1999	▼
Denis G. Vorobiev	3	32	1999	▼
Michael Hucka	1	42	2003	▼
Andrew Finney	1	42	2003	▼
H. Bolouri	1	42	2003	▼
John C. Doyle	1	42	2003	▼

Figura 8. Autores que publicaram com mais de 30 coautores em determinado ano.

Na Figura 9, é apresentado o resultado referente à pergunta 12 do Quadro 2, onde são selecionados todos os autores que publicaram mais de 120 publicações ao longo dos anos. Além do nome dos autores também é apresentado o total de publicações, bem como uma mensagem contendo a quantidade de autores encontrados.

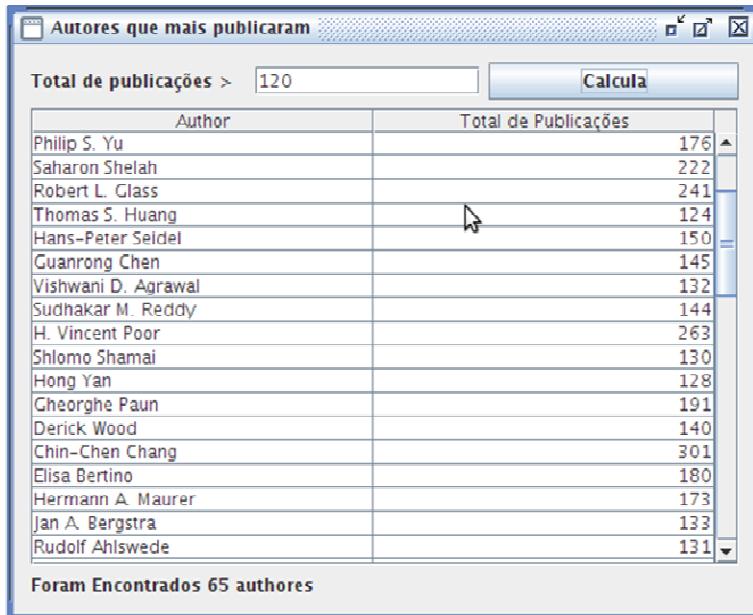


Figura 9. Autores onde o número total de publicações é maior que 120.

Percebe-se, através dos dados da Figura 9 que poucos pesquisadores possuem um número elevado de publicações.

5. Considerações Finais

O trabalho teve como objetivo demonstrar todas as etapas da construção de uma ferramenta para a análise quantitativa da produção dos pesquisadores, demonstrando através de experimentos o funcionamento da ferramenta proposta.

O estudo sobre redes de colaboração científica possibilitou uma análise nos diversos tipos de relacionamentos entre pesquisadores, desde as tendências das relações ao longo dos anos até a mudança de comportamento que as redes sofrem com o passar do tempo. Foi possível, ainda, verificar que a maioria dos pesquisadores existentes na DBLP possui um pequeno número de artigos publicados.

Pode-se observar que, aproximadamente, 220 mil pesquisadores publicaram apenas um artigo, o que corresponde a 50% do total de pesquisadores cadastrados na DBLP. Outro dado interessante é que dos 440.000 pesquisadores da DBLP, apenas 65 publicaram mais do que 120 artigos. Também foi possível identificar e cruzar dados entre pesquisadores identificando autores e coautores em determinado período do tempo, analisando se sua rede de colaboração aumentou ou diminui com o passar do tempo.

Como trabalho futuro pretende-se identificar a(s) subárea(s) de cada artigo, definindo um perfil para cada pesquisador. Isto pode indicar com foi o comportamento do pesquisador ao longo de sua trajetória, em que áreas atuou, bem como quantos trabalhos possui em cada área. Com isso, pode-se fazer recomendação de trabalhos para que pesquisadores aumentem sua rede de colaboração científica.

Referências

- Arruda, D., Bezerra, F., Neris, V. A., Rocha de Toro, P., Wainer, J. Brazilian Computer Science Research: Gender and Regional Distributions. *Scientometrics*, 2009.
- Brandão, W. C.; Parreiras, F. S.; Silva, A., B. O. (2007). Redes em Ciência da Informação: Evidências Comportamentais dos Pesquisadores e Tendências Evolutivas das Redes de Co-autoria. *Informação & Informação*, Londrina, v. 12, n. 0, p. 00-00, jan./jun. 2007.

- Chen, C., Song, I., Yuan, X., and Zhang, J. (2008). The Thematic and Citation Landscape of Data and Knowledge Engineering (1985-2007). *Data & Knowledge Engineering*. v.67, n.2 (Nov. 2008), p. 234-259.
- CiteSeer (Scientific Literature Digital Library). Disponível em <<http://citeseer.ist.psu.edu>>. Acesso em maio, 2010.
- DBLP (Digital Bibliography & Library Project). University of Trier. Disponível em <<http://dblp.uni-trier.de>>. Acesso em abril, 2010.
- Fu, Y.; Xiang, R.; Liu, Y.; Zhang, M.; Ma, S. Finding Experts Using Social Network Analysis. Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence. Silicon Valley, USA, 2007.
- Hope, T.; Nishimura, T.; Takeda, H. An Integrated Method for Social Network Extraction. Proceedings of the 15th International Conference on World Wide Web. Edinburgh, Scotland, 2006.
- Huang, J., Zhuang, Z., Li, J., and Giles, C. L. 2008. Collaboration Over Time: Characterizing and Modeling Network Evolution. In Proceedings of the International Conference on Web Search and Web Data Mining. Palo Alto, California, USA, 2008.
- Li, J.; Tang, J.; Zhang, J.; Luo, Q.; Liu, Y.; Hong, M. EOS - Expertise Oriented Search Using Social Networks. In Proceedings of the 16th International Conference on World Wide Web. Banff, Canada, 2007.
- Liu, X., Bollen, J., Nelson, M. L., and Van de Sompel, H. (2005). Co-authorship Networks in the Digital Library Research Community. *Information Processing and Management: An International Journal*. V.41, n.6 , p.1462-1480, 2005.
- Medeiros, C. M. B. (2008). Grand Research Challenges in Computer Science in Brazil. *Computer* v.41, n.6, p.59-65, 2008.
- Menezes, G. V., Ziviani, N., Laender, A. H., and Almeida, V. (2009). A Geographical Analysis of Knowledge Production in Computer Science. In Proceedings of the 18th International Conference on World Wide Web. Madrid, Spain, 2009.
- Mimno, D.; McCallum, A. Expertise Modeling for Matching Papers With Reviewers. In Proceedings of the 13th International Conference on Knowledge Discovery and Data Mining. San Jose, USA, 2007.
- Tang, J., Zhang, J., Yao, L., Li, J. Extraction and Mining of an Academic Social Network. In Proceedings of the 17th International Conference on World Wide Web. Beijing, China, 2008.
- Tang, J.; Zhang, D.; Yao, L. Social Network Extraction of Academic Researchers. In Proceedings of 7th International Conference on Data Mining. Omaha, USA, 2007.
- Tang, J.; Zhang, J.; Yao, L.; Li, J.; Zhang, L.; Su, Z. ArnetMiner – Extraction and Mining of Academic Social Networks. In Proceeding of the 14th International Conference on Knowledge Discovery and Data Mining. Las Vegas, USA, 2008.
- Vardi, M. Y. (2009). Conferences vs. Journals in Computing Research. *Communications of the ACM* v.52, n.5 p.5-5, 2009.
- Wainer, J., Novoa Barsottini, C. G., Lacerda, D., and Magalhães de Marco, L. R. (2009). Empirical Evaluation in Computer Science Research Published by ACM. *Information and Software Technology*. V.51, n.6, p.1081-1085, 2009.
- Wainer, J., Xavier, E. C., Bezerra, F. Scientific Production in Computer Science: A Comparative Study Between Brazil and Other Countries. *Scientometrics*, 2009.
- Zhang, J.; Tang, J.; Li, J. Expert Finding in a Social Network. Proceedings of 12th International Conference on Database Systems for Advanced Applications. Bangkok, Thailand, 2007.

Banco de Dados em Nuvem: Conceitos, Gerenciamento e Desafios

Darlan Florêncio de Arruda¹, José A. F. de Moura Júnior^{1,2}

¹Faculdade de Ciência e Tecnologia de Caruaru - Universidade de Pernambuco (UPE)

Caruaru – PE- Brasil

²Instituto Federal de Pernambuco (IFPE)

Belo Jardim – PE - Brasil

darlanarruda250@gmail.com, almir.moura@belojardim.ifpe.edu.br

Abstract. *Cloud computing has emerged as a new paradigm in the deployment of applications in which computing resources are provided as a service via a network connection. These services are provided in the form of software, infrastructure and platforms. With the use of these services are offering users' information, often confidential unknown geographically remote servers, or databases in the clouds. This paper presents overview of this type of database, as well as features and advantages, besides dealing with the challenges encountered in managing the data stored in this environment.*

Resumo. *Computação em Nuvem surge como um novo paradigma na implantação de aplicações em que os recursos computacionais são fornecidos como um serviço através de uma conexão de rede. Esses serviços são disponibilizados em forma de software, infraestruturas e plataformas. Com a utilização desses serviços usuários estão disponibilizando informações, muitas vezes confidenciais em servidores remotos desconhecidos geographicamente, ou seja, os bancos de dados em nuvem. Este artigo apresenta uma visão geral desse tipo de banco de dados, assim como características e vantagens, além de tratar dos desafios encontrados no gerenciamento dos dados armazenados nesse ambiente.*

1. Introdução

Com o avanço tecnológico, a ideia de ter, armazenar e distribuir informações de forma rápida vem chamando a atenção de muitos usuários, fazendo com que aumente a busca por esse tipo de serviço, principalmente em grandes corporações. Diante disso, a *Cloud Computing* surge como um novo paradigma na implantação de aplicações em que os recursos computacionais são fornecidos como um serviço através de uma conexão de rede. Esses serviços são disponibilizados em forma de software, infraestruturas e plataformas, onde usuários podem adquiri-los sob demanda.

Mas, como toda nova tecnologia, a *Cloud Computing* também preocupa os usuários quanto a diversos pontos relacionados ao gerenciamento e segurança dos dados que estarão armazenados na nuvem. Sistemas de gerenciamento de dados são fortes candidatos para implantação em nuvem. Pois com o uso dos serviços disponibilizados, há também o uso de uma grande quantidade de dados que precisa ser armazenada. Sistemas de gerenciamento de dados em nuvem possuem custo reduzido e maior facilidade de acesso aos dados, entretanto existem desafios relacionados à consistência e segurança dos dados, que são importantes em ambientes *Cloud computing* e deste modo precisam ser tratados cautelosamente (Sousa et al., 2010).

Este artigo busca enfatizar os principais desafios enfrentados no gerenciamento de informações que são armazenadas em banco de dados em nuvem. A próxima seção apresenta a metodologia utilizada no artigo. A seção 3 mostra alguns trabalhos relacionados ao tema. A seção 4 trata do conceito de Computação em Nuvem. Na seção 5 são abordadas conceitos e características de Banco de Dados em Nuvem, já a seção 6 discorre a respeito dos conceitos gerais de Gerenciamento de Dados em ambientes de *Cloud Computing*, a seção 7 mostra os principais Sistemas de Gerenciamento de Dados em Nuvem. A seção 8, por sua vez, apresenta alguns dos principais desafios que são encontrados no Gerenciamento de Dados em Nuvem, e, por fim, a seção 9 traz as conclusões do estudo realizado e sugestões de trabalhos futuros.

2. Metodologia

Para Silva (2001) a metodologia científica é o conjunto de processos e operações mentais que se deve empregar nas investigações. É a linha de raciocínio adotado no processo da pesquisa. Ou seja, para que uma pesquisa seja efetuada é necessário um conjunto de procedimentos intelectuais e técnicos. Uma pesquisa pode ser classificada por diversas características, que podem ser, por exemplo, quantos aos fins e quanto aos meios.

Quanto aos fins, a presente pesquisa pode ser considerada descritiva, pois de acordo com Gil (1999, p.46) este tipo de pesquisa adota como objetivo primordial as descrições das características de uma determinada população ou determinado fenômeno ou o estabelecimento de relações entre variáveis.

Quanto aos meios, esse trabalho possui características de pesquisa bibliográfica. Segundo Silva (2001), pesquisa bibliográfica é elaborada a partir de material já publicado, constituído principalmente de livros, artigos de periódicos e atualmente com material disponibilizado na internet.

3. Trabalhos Relacionados

Algumas publicações encontradas na literatura tratam de gerenciamento de dados em nuvem, bem como suas limitações, desafios e as oportunidades que surgem como agentes impulsionadores para a solução desses desafios.

Abadi (2009), por exemplo, apresenta uma discussão sobre as limitações e possibilidades de implantação de técnicas de gerenciamento de dados em plataformas de *Cloud Computing* emergentes, como a *Amazon Web Services*, por exemplo. No referido trabalho também são apresentadas algumas características que um Sistema de

Gerenciamento de Dados em Nuvem deve possuir quando se é projetado para o armazenamento de dados em grande escala.

Wei et al. (2009), por sua vez, enfatiza a importância da escalabilidade e da alta disponibilidade em serviços de dados em nuvem. Entretanto para garantir essas características ele cita que ocorrem alguns problemas como consistência de dados e segurança dos dados, que devem ser resolvidos, porém no referido artigo não são explanadas soluções para a resolução dos mesmos.

Em seu trabalho, Sousa et al. (2010) aborda os principais conceitos de computação em nuvem com ênfase para os desafios no gerenciamento de dados nesse tipo de ambiente, entre os quais são citados escalabilidade e consistência, qualidade dos serviços de dados, e segurança dos serviços de dados.

Além de explanar problemas similares, os trabalhos supracitados têm em comum o fato de apontarem desafios, sem todavia, fazerem propostas efetivas de soluções para a resolução dos mesmos.

A seção seguinte traz uma visão geral do conceito de Computação em Nuvem.

4. Computação em Nuvem

Segundo Nist (2009), Computação em nuvem é um modelo que possibilita acesso, de modo conveniente e sob demanda, a um conjunto de recursos computacionais configuráveis (por exemplo, redes, servidores, armazenamento, aplicações e serviços) que podem ser rapidamente adquiridos e liberados com mínimo esforço gerencial ou interação com o provedor de serviços. Este modelo de nuvem promove disponibilidade e é composto por cinco características essenciais, três Modelos de Serviços (SaaS, PaaS e IaaS), e quatro Modelos de Implantação (privada, pública, comunitária e híbrida)¹.

Diante disso, a Computação em Nuvem apresenta-se como uma maneira bastante eficiente de maximizar e flexibilizar os recursos computacionais, diante da multiplicidade de serviços prestados pelos provedores de Computação em Nuvem, tais como, armazenamento de dados, desenvolvimento de aplicativos personalizados e gestão de infraestrutura, por exemplo. Cada parte desta infraestrutura é provida como um serviço e, estes são normalmente alocados em centros de dados, utilizando hardware compartilhado para computação e armazenamento (BUYYA et al., 2008).

Na seção a seguir, serão explanadas as principais características dos Banco de Dados em Nuvem.

5. Banco de Dados em Nuvem (BDN)

Banco de Dados em Nuvem (BDN) estão começando a ser utilizados e têm o potencial de atrair clientes de diversos setores do mercado, desde pequenas empresas com o objetivo de reduzir o custo total, por meio da utilização de infraestrutura e sistemas de terceiros, até grandes empresas que buscam soluções para gerenciar milhares de máquinas e permitir o atendimento de um aumento inesperado de tráfego (Abadi, 2009).

Segundo Cottman (2011), há uma palavra que resume bem a origem dos BDN: volume. A evolução da Internet tem gerado grandes volumes de dados (terabytes, petabytes, e assim por diante) que surpreendem até mesmo a mente de um profissional de TI, em contrapartida os bancos de dados relacionais convencionais não se mostraram

¹ Para maiores detalhes sobre as cinco características essenciais, os três modelos de serviços e os quatro Modelos de Implantação, ver (Nist, 2009)

suficientemente preparados para lidarem da melhor forma possível com um quantitativo de dados dessa magnitude. Não é por acaso que quando se pergunta como portais bem estabelecidos como o *Google*, o *Facebook* e o *Twitter* poderiam armazenar, processar e arquivar essa vasta quantidade de dados, a resposta é uma só: através de BDN.

Apesar do uso cada vez mais generalizado do termo BDN, não foi encontrada na literatura uma definição formal para o mesmo. Não obstante, em linhas gerais pode-se defini-lo como uma coleção de dados inter-relacionados que estão armazenados na web, e que podem ser gerenciados e manipulados através de Sistemas de Gerenciamento de Dados em Nuvem (SGDN), softwares estes que segundo Cottman (2011) são especialmente concebidos para atingir o processamento massivamente paralelo – na ordem das centenas de milhares de tarefas simultâneas -, utilizando os recursos distribuídos de computação em *grid*².

Para Barros (2011), conexões de internet mais rápidas favorecem a adoção de BDN. A mesma possibilitará aos sistemas estarem hospedados na web. Com um banco de dados totalmente *online*, operações como *backups*, *restores*, pesquisas, inserções e deleções poderão ser realizadas facilmente por intermédio de qualquer *browser*.

Barros (2011) aponta ainda um crescimento da procura por BDN tanto a nível acadêmico quanto profissional. Uma pesquisa realizada pela Embarcadero Technologies³ reforça esta tese. Segunda a pesquisa - realizada com 1230 profissionais da área de banco de dados - com o objetivo de recolher informações sobre as tendências de banco de dados, desafios, principais iniciativas e ferramentas atuais que estão sendo utilizadas, pôde-se verificar que a tecnologia de BDN será a que mais afetará o setor de banco de dados (EMBARCADERO, 2010).

33,6% dos entrevistados citaram BDN como a tecnologia que exercerá maior impacto na comunidade. Também foram citadas outras tecnologias como virtualização⁴, discos de estado sólido e *tuning* visual, conforme ilustrado na figura 1.



Figura 1. Tecnologias impactantes na comunidade de banco de dados

(Fonte: EMBARCADERO, 2010)

² Computação em grid é o termo utilizado para se referir a uma técnica computacional que utiliza os recursos de diferentes computadores.

³ A Embarcadero Technologies, Inc. é uma empresa que fornece ferramentas de bancos de dados de nível profissional que as empresas utilizam para desenhar, desenvolver e gerenciar bancos de dados.

⁴ Virtualização é uma forma de esconder as características físicas de uma plataforma computacional dos usuários, mostrando outro hardware virtual, emulando um ou mais ambientes isolados.

Tecnologais que envolvem BDN possuem um alto grau de escalabilidade, fazendo com que as aplicações crescam de acordo com a sua demanda de utilização. Esse crescimento se dá uniformemente - a escalabilidade em BDN implica diretamente no desempenho do sistema utilizado.

Deste modo, pode-se dizer que a utilização de BDN possui algumas vantagens em relação aos Banco de Dados convencionais. Dentre essas podem-se destacar a flexibilidade/escalabilidade, a tecnologia de ponta utilizada, o acesso à informação em qualquer lugar (informação distribuída) e o custo.

A seção seguinte aponta a importância do Gerenciamento de Dados em Nuvem, bem como algumas de suas principais características.

6. Gerenciamento de Dados em Nuvem

O gerenciamento de dados é um fator muito importante dentro do contexto de computação em nuvem, uma vez que a segurança desses dados é fator crucial em ambientes de *Cloud Computing* e deve ser tratado com certa atenção. Outro fator relevante é que os Sistemas de Gerenciamento de Banco de Dados (SGBDs) relacionais não possuem escalabilidade quando existe uma grande quantidade de dados armazenados por eles. Assim, aspectos de armazenamento de dados, processamento de consultas e controle transacional têm sido flexibilizados por algumas abordagens para garantir a escalabilidade, mas ainda não existem soluções que combinem estes aspectos de forma a melhorar o desempenho sem comprometer a consistência dos dados (ABADI, 2009).

A escalabilidade do sistema deve ser transparente para os usuários, podendo estes armazenar seus dados na nuvem sem a necessidade de saber a localização dos dados ou a forma de acesso (Sousa et al., 2010). A escalabilidade do sistema ocorre quando o mesmo possui a capacidade de crescer conforme a demanda de uso.

Já a disponibilidade do serviço possibilita aos usuários acesso aos dados quando e onde quiserem ou precisarem, os Sistemas de Gerenciamento de Dados em Nuvem (SGDN) devem dispor de uma alta disponibilidade, visto que o meio de comunicação entre o usuário e o sistema é a internet, e que pode ocorrer atrasos e indisponibilidade do sistema.

Outro item relevante no contexto de gerenciamento de dados em nuvem é a consistência de dados, onde todos os nós devem ter a mesma visão dos dados ao mesmo tempo, fazendo assim, com que todos os dados armazenados sejam iguais para todos os nós do sistema, por exemplo, se um determinado dado for atualizado, então ele deverá ser atualizado para todos os nós do sistema, mantendo assim os dados íntegros, coerentes e consistentes.

Existem diversas ferramentas que viabilizam o gerenciamento de dados em nuvem, algumas das mais relevantes são apresentadas a seguir.

7. Sistemas de Gerenciamento de Dados em Nuvem (SGDN)

Em linhas gerais, um Sistema de Gerenciamento de Dados em Nuvem (SBDN) pode ser definido como um ou mais programas que possibilita(m) a criação, manipulação e gerenciamento de Bancos de Dados em Nuvem.

Segundo Sousa (2010) a infraestrutura de SGDN possui várias vantagens para os usuários, como: previsibilidade e custos mais baixos, proporcional à qualidade do serviço (QoS) e cargas de trabalho reais, complexidade técnica reduzida, graças a interfaces de acesso unificado e a delegação de *tuning* e administração de SGBDs e a elasticidade⁵ e escalabilidade, proporcionando a percepção de recursos quase infinitos.

Existem diversos sistemas para o gerenciamento de dados em nuvem. Em uma enquete realizada pela Dzone puderam-se verificar quais são os sistemas de gerenciamento de dados em nuvem que estão sendo utilizados ou despertam interesse para adoção. O resultado dessa enquete pode ser visto na figura 2.

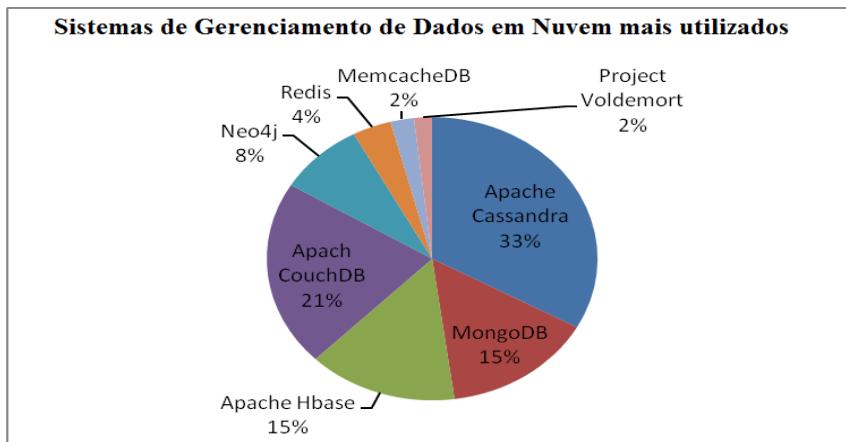


Figura 2. Sistemas de Gerenciamento de Dados em Nuvem mais utilizados

(Fonte: Dzone, 2010)

Estes dados são referentes a um universo de 736 pessoas que responderam a pesquisa, não representando a mesma proporção entre o uso dos SGDN.

Conforme evidencia a figura 2, dentre os SGDN mais comuns destacam-se o Cassandra, o CouchDB, o MongoDB e o Microsoft SQL Azure. A seguir são apresentadas de forma resumida algumas das principais características desses sistemas.

7.1 Cassandra

O *Apache Cassandra* é um sistema de armazenamento distribuído para o gerenciamento de grandes quantidades de dados espalhados por centenas de máquinas. Ele pode funcionar em *hardware* de baixo custo e lida com alta taxa de escrita sem sacrificar a eficiência na leitura, é um sistema tolerante a falhas, fazendo com que aumente sua confiabilidade. Recentemente ganhou popularidade por ser utilizado, entre outros, pelo Twitter e pelo Facebook (CASSANDRA,2011).

7.2 CouchDB

O *Apache CouchDB* é um SGDN orientado a documento⁶ que possui uma série de características que torna sua utilização viável em servidores que possuem *hardware* de baixo

⁵ Elasticidade diz respeito ao fato de os recursos poderem ser adquiridos de forma elástica, em alguns casos, automaticamente escalados com o aumento da demanda e liberados, na sua retração.

⁶ Na abordagem orientada a documento, os documentos são as unidades básicas de armazenamento e estes não utilizam necessariamente qualquer tipo de estruturação pré-definida, como é o caso das tabelas do Modelo Relacional.

desempenho e utilizam técnicas de armazenamento e controle de concorrência baseadas na estrutura do documento (COUCHDB, 2011).

7.3 Mongo DB

O *MongoDB* é um SGDN escalável e de alto desempenho, de código aberto, orientado a documento, escrito em C++ e que não possui transações ou junções (MONGODB, 2011).

7.4 Microsoft SQL Azure

O *Microsoft SQL Azure* é um SGDN composto por um conjunto de serviços para o armazenamento e processamento de dados em nuvem. Juntamente com o *Windows Azure Storage* formam o SGDN da *Microsoft*. O *SQL Azure* implementa alta disponibilidade, tolerância a falhas e o conceito de multi-inquilino⁷. O SQL Azure foi projetado com base em tecnologia de SGBD relacional do *SQL Server* onde seu principal componente é o *SQL Azure Database* (AZURE, 2011).

7.5 Análise Comparativa

O quadro 1 mostra os resultados de uma análise comparativa - realizada pelos autores do presente artigo - entre os SGDN explorados no trabalho. Os dados presentes na análise são oriundos de pesquisas realizadas nas páginas oficiais dos sistemas supracitados. No comparativo foram levadas em consideração algumas características essenciais para esses ambientes, como distribuição, modelo de dados, armazenamento de dados, escalabilidade, entre outras.

Quadro 1. Comparativo entre os Sistemas de Gerenciamento de Dados em Nuvem

Características	Apache Cassandra	Couch DB	Mongo DB	Microsoft SQL Azure
Distribuição dos dados	Distribuído	Distribuído	Distribuído	Distribuído
Modelo dos dados	Orientado a coluna	Orientado a documento	Orientado a documento	Relacional
Armazenamento dos dados	Indices	Arvore B+	Indices	Tabelas
Custo	Proprietário	Open Source	Open Source	Proprietário
Escrito em	JAVA	Erlang OTP	C++	-
Escalabilidade	Forte	Eventual	-	Eventual
Quem Usa	Facebook,e Twitter,Digg	Meebo e BBC	Engine Yard	-

Pelo quadro acima nota-se nitidamente que a escalabilidade ainda não atingiu um grau satisfatório em pelo menos metade dos SGDN analisados, apontando a escalabilidade como um desafio importante a ser superado na área de BDN. Por outro lado, algumas outras características relevantes não puderam estar presentes no quadro 1 tais quais *segurança* e *qualidade de serviço*, visto que ainda não se mostraram

⁷ Gerenciamento de dados multi-inquilino é uma técnica para consolidar aplicações de múltiplos inquilino em um único sistema.

suficientemente maduras no contexto de gerenciamento de BDN, e por isso também se apresentam como desafios relevantes para a referida área.

A seção seguinte aponta quais seriam os principais desafios para o gerenciamento de BDN nos dias atuais.

8. Desafios para o Gerenciamento de Banco de Dados em Nuvem

A Computação em Nuvem proporciona diversas vantagens à aqueles que a utiliza, porém mediante a tantas vantagens, surgem diversos desafios que devem ser superados nesse tipo de ambiente, entre os quais podem-se destacar a *segurança da informação*, a *escalabilidade*, a *consistência de dados*, a *qualidade do serviço de dados*, entre outros.

8.1 Segurança

A segurança tem tido um papel muito importante no impedimento do desenvolvimento da Computação em Nuvem. Questões de segurança tradicionais já bastante conhecidas e tratadas em sistemas distribuídos, como controle de acesso, confidencialidade, integridade de dados, tolerância a falhas, também fazem parte desta nova tecnologia.

Por utilizar a internet como meio para a disponibilização dos serviços, isto se torna um tanto complexo, diante da diversidade de recursos que são utilizados, como por exemplo, sistemas operacionais distintos, *softwares*, domínio de redes, políticas de segurança da informação. A percepção de que a Nuvem é um aglomerado de informações pode caracterizá-la como sendo um alvo propício a ataques. Ameaças como estas podem afetar diretamente os pilares da segurança da informação: disponibilidade, confidencialidade, integridade e não-repúdio, e consequentemente comprometer toda a Nuvem.

A segurança da informação deve ser tratada como peça-chave no desenvolvimento de BDN, uma vez que proporciona a autenticidade, integridade e confidencialidade ao sistema.

8.2 Escalabilidade e consistência dos dados

As soluções em nuvem focam muito em escalabilidade e com isso oferecem uma fraca consistência dos dados armazenados. Os SGDN devem ser altamente escaláveis, pois não se pode definir uma quantidade de dados a serem armazenados, e a escalabilidade proporcionará o crescimento do sistema de acordo com a demanda de uso, porém não garante a consistência dos dados por eles armazendados.

A maioria das soluções de BDN utiliza a consistência eventual, que é um tipo de consistência fraca. Este tipo de consistência não permite a construção de uma ampla gama de aplicações, tais como serviços de pagamento e leilões *online*, que não podem trabalhar com dados inconsistentes (Wei et al. 2009).

Dessa maneira, a consistência dos dados torna-se assunto crucial a ser analisado e tratado quando se for projetar aplicações e soluções de BDN, sendo visto como um desafio que deve ser superado para que sejam projetados SGDN mais eficientes quanto ao seu uso, e sobretudo com maior grau de integridade e corretude dos dados.

8.3 Qualidade do serviço de dados

A qualidade do serviço é característica essencial em ambientes de *Cloud computing*, proporcionando aos usuários algumas garantias como disponibilidade e desempenho. A disponibilidade de serviços permite aos usuários acessarem e utilizarem a nuvem onde e quando desejarem. Dessa forma, os ambientes de Computação em Nuvem devem prover

alta disponibilidade, que pode acontecer através de garantias de *QoS* (*Quality of Service*).

Os sistemas em nuvem apesar de possuirem algumas limitações em termos de segurança e de rede, devem fornecer desempenho elevado, e ser muito flexíveis para se adequar a uma grande quantidade de requisições. O desempenho passa a ser um desafio considerável em SGBN, pois como alguns são de acesso público não há como dimensionar a quantidade de requisições realizadas, aumentando a dificuldade em fazer estimativas e obter garantias de *QoS*. Deste modo, o desempenho se apresenta como mais um grande e importante desafio a ser superado para que esses sistemas possam ser executados em sua amplitude.

9. Conclusões e Trabalhos Futuros

Este trabalho abordou os principais aspectos e características dos Bancos de Dados em Nuvem, enfatizando os desafios que são encontrados no gerenciamento dos dados que são armazenados em ambientes deste tipo.

Foi visto ainda que a tecnologia de BDN é a que trará maior impacto na comunidade de banco de dados. Embora essa abordagem traga diversas vantagens para os usuários, ainda existem diversas barreiras e desafios que precisam ser estudados e superados. Dentre os mesmos destacam-se a *segurança*, a *consistência dos dados*, a *escalabilidade* e as garantias de *qualidade do serviço*, que são essenciais para o bom funcionamento de um sistema que faça uso de um BDN.

Neste cenário, é importante fazer uma análise profunda das vantagens oferecidas por essa tecnologia e as ameaças que podem ser geradas quando se migra para esse tipo de ambiente para que, no futuro, essas vantagens não se tornem prejuízos irrecuperáveis.

Como sugestões de trabalhos futuros, destaca-se a necessidade de ampliar o estudo através de análises mais profundas sobre segurança da informação em BDN, também verificar e expandir métodos que garantam a consistência dos dados armazenados mantendo a qualidade do serviço, além de garantir a escalabilidade do sistema, visto que esta última se trata de uma das principais características e necessidades de aplicações em nuvem. Além de analisar outros desafios que por hora não são os mais importantes para o desenvolvimento da área, mas, que possam contribuir efetivamente para uma melhora de qualidade dos Sistemas de Gerenciamento de Dados em Nuvem.

Referências Bibliográficas

ABADI, D. J. (2009). *Data management in the cloud: Limitations and opportunities*. Disponível em: <http://www.cs.yale.edu/homes/dna/papers/abadi-cloud-ieee09.pdf>. Acessado em: 07 de dezembro de 2010

AZURE. *Microsoft Azure*. Disponível em: <http://www.microsoft.com/azure/>. Acessado em: 05 de janeiro de 2011.

BARROS, L. E. B. *Banco de Dados em Nuvem*. Disponível em: <http://pesquompile.wikidot.com/banco-de-dados-em-nuvem>. Acessado em: 10 de Fevereiro de 2011.

BUYYA, R.; YEO C. S.; VENUGOPAL, S. *Market-Oriented Cloud Computing: Vision, Hype, and Reality for Delivering IT Services as Computing Utilities*. Grid Computing and Distributed Systems (GRIDS) Laboratory Department of Computer Science and Software Engineering. The University of Melbourne, Australia, 2008.

CASSANDRA. *Cassandra* Disponível em: <http://cassandra.apache.org/>. Acessado em: 05 de Janeiro de 2011.

COTTMAN, G. *Why Cloud Databases*. Disponível em: http://wiki.toadforcloud.com/index.php/Why_cloud_databases. Acessado em: 12 de Fevereiro de 2011.

COUCHDB. *The CouchDB Project*. Disponível em: <http://couchdb.apache.org>. Acessado em: 05 de Janeiro de 2011.

DZONE. *NoSQL DZone Poll Results*. Disponível em: <http://java.dzone.com/articles/nosql-dzone-poll-results>. Acessado em: 05 de Janeiro de 2011.

EMBARCADERO. *Database Trends Survey Report*. Disponível em: <http://www.embarcadero.com/images/dm/technical-papers/database-survey-report.pdf>. Acessado em: 18 de dezembro de 2010.

GIL, Antonio C. *Metodos e técnicas de pesquisa social*. 5.ed. São Paulo: Atlas, 1999.

MONGODB. *Mongo DB*. Disponível em: <http://www.mongodb.org/>. Acessado em: 05 de Janeiro de 2011.

NIST. *The NIST Definition of Cloud Computing*. 2009. Disponível em: <<http://csrc.nist.gov/groups/SNS/cloud-computing/>>. Acessado em: 23 de novembro 2010.

SILVA, Edna Lúcia. *Metodologia da pesquisa e Elaboração de dissertação*. 3º Ed. Florianópolis: 2001.

SOUZA, Flávio R. C.; MOREIRA, Leonardo O.; MACÊDO, José Antônio de; MACHADO, Javam C. *Gerenciamento de Dados em Nuvem: Conceitos, Sistemas e Desafios*. Disponível em: http://www.es.ufc.br/~flavio/files/Gerenciamento_Dados_Nuvem.pdf. Acessado em 20 de dezembro de 2010.

WEI, Z., PIERRE, G., CHI, C.-H. (2009). *Scalable transactions for web applications in the cloud*. Disponível em: http://www.globule.org/publi/STWAC_europar2009.pdf. Acessado em 23 de dezembro de 2010

PGSimilarity: executando consultas aproximadas em um SGBDR

Euler Taveira de Oliveira¹

¹Instituto de Informática
Universidade Federal do Rio Grande do Sul (UFRGS)
Av. Bento Gonçalves, 9500 – Porto Alegre – RS – Brazil

euler@timbira.com.br

Abstract. Although it has already detected the need to support approximate queries in a RDBMS, the current SQL standard didn't define it yet. This paper describes additions to SQL standard needed for approximate queries (including user-defined operators and session variables) and presents the PGSimilarity tool that uses the described infrastructure and implements a set of similarity operators. Furthermore, indexing support was developed for some similarity functions.

Resumo. Embora já tenha sido detectada a necessidade de suporte a consultas aproximadas em um SGBDR, o atual padrão SQL não o definiu ainda. O presente trabalho descreve o suporte SQL necessário as consultas aproximadas (incluindo operadores e variáveis de sessão definidas pelo usuário) e apresenta uma ferramenta que utiliza a infra-estrutura definida e implementa um conjunto de operadores de similaridade. Além disso, suporte à indexação foi desenvolvido para algumas funções de similaridade.

1. Introdução

Hoje em dia a grande maioria das aplicações utilizam SGBDs baseados no Modelo Relacional de Dados [Codd 1970], que são os chamados *Sistemas Gerenciadores de Bancos de Dados Relacionais* (SGBDR) ou seus variantes denominados de *Sistemas Gerenciadores de Bancos de Dados Objeto-Relacionais* (SGBDOR). As consultas suportadas por esses SGBDs são baseadas em expressões condicionais que permitem que valores sejam comparados utilizando operadores tais como $=$, \neq , $>$, \geq , $<$ e \leq . Entretanto, não conseguimos inferir que duas informações referem-se a mesma entidade apesar de estarem representadas de maneira diferente.

Algumas das consultas requeridas por aplicações de integração de informação são normalmente baseadas em uma noção de aproximação ou similaridade, que é específica para cada domínio de dados. Esse tipo de consulta, que é denominado de *consulta por similaridade* ou *consulta aproximada*, retorna um conjunto de dados que sejam similares a um *objeto de consulta* (cadeia de caracteres, número, subsequências de DNA, imagens, dentre outros) de acordo com um certo critério de similaridade. Como as comparações por similaridade necessitam de um meio para mensurar o grau de aproximação entre dois objetos utilizamos o que denominamos de *função de similaridade* ou *função de distância* [Cohen et al. 2003].

Embora já tenha sido detectado a necessidade de suporte a consultas por similaridade em um SGBDR, o atual padrão da linguagem *SQL* (*Structured Query Language*) não apresenta recursos para que tais operações sejam possíveis. Para fornecer esse suporte, um mecanismo de definição de novos operadores deve ser incorporado a linguagem *SQL*. Assim, algoritmos de similaridade podem ser codificados como funções definidas pelo usuário¹ e serem associadas a novos operadores.

Neste artigo, apresentamos a sintaxe de operadores definidos pelo usuário e o *PGSimilarity*, que contém um conjunto de operadores definidos pelo usuário para suportar consultas aproximadas. Exploramos toda infra-estrutura de um SGBDR para manter a sintaxe e a performance das consultas aproximadas semelhantes as consultas tradicionais.

Este artigo está organizado como segue: a seção 2 descreve a extensão ao SQL que provê a infra-estrutura necessária para implementação de operadores de similaridade. A seção 3 apresenta o *PGSimilarity*, uma extensão de código aberto para o SGBDR PostgreSQL. Nas seções 4 e 5 são descritos os trabalhos relacionados e considerações finais, respectivamente.

2. Extensão ao SQL

A linguagem amplamente utilizada nos SGBDRs é a *SQL*. Apesar da riqueza de recursos que a linguagem *SQL* dispõe, não temos construções para especificar consultas por similaridade ou mesmo definir novos operadores para consultas aproximadas. Para permitir a manipulação de dados por similaridade em um SGBDR é necessário incorporar à linguagem *SQL* construções que possibilitem:

- *DDL*: definir operadores de similaridade que especifiquem a função a ser utilizada e os tipos de dados dos operandos. Os tipos de dados são os tradicionais (ou seja, aqueles definidos na linguagem *SQL*) mas que podem ser estendidos a tipos definidos pelo usuário também. Definir variáveis de sessão que controlem quanto flexível é o casamento entre os operandos;
- *DML*: permitir a especificação de consultas aproximadas de maneira similar a consultas utilizando operadores tradicionais (incluindo operações de seleção e junção).

2.1. Operadores

Um dos pontos fundamentais para consultas por similaridade em *SQL* está relacionado à definição de medidas de similaridade como operadores. Como não existe o conceito de definição de operadores em *SQL*, é necessário criar novos comandos para tratar essa questão. Desse modo, são necessários três comandos para manipulação desses operadores definidos pelo usuário: *CREATE OPERATOR*, *ALTER OPERATOR* e *DROP OPERATOR*. Alguns SGBDRs apresentam esses comandos de definição de operadores tais como *Oracle* [Oracle 2011] e *PostgreSQL* [PostgreSQL 2011]. No entanto, eles não permitem associar variáveis definidas pelo usuário a um operador. A sintaxe sugerida para o comando *CREATE OPERATOR* é descrita na sintaxe 1.

¹a maioria dos SGBDRs comerciais dispõem desse recurso

```
CREATE OPERATOR <nome_de_operador>
( { <operando_a_esquerda> | NONE } ,
{ <operando_a_direita> | NONE } )
USING <funcao_de_operador>
```

Sintaxe 1. Criar operador definido pelo usuário

O *nome_de_operador* não nomeia o operador; ele juntamente com os operandos à esquerda e à direita fazem esta identificação. Assim, podemos definir o mesmo nome de operador para tipos de dados diferentes (se este for o caso). Por isso, as sintaxes dos comandos *ALTER OPERATOR* e *DROP OPERATOR* precisam conter além do nome de operador, os operandos à esquerda e à direita. As sintaxes sugeridas dos comandos *ALTER OPERATOR* e *DROP OPERATOR* são, respectivamente, sintaxes 2 e 3.

```
ALTER OPERATOR <nome_de_operador>
( { operando_a_esquerda | NONE } ,
{ operando_a_direita | NONE } )
[ USING <funcao_de_operador> ]
[ RENAME TO <novo_nome_de_operador> ]
```

Sintaxe 2. Modificar operador definido pelo usuário

```
DROP OPERATOR <nome_de_operador>
( { operando_a_esquerda | NONE } ,
{ operando_a_direita | NONE } )
```

Sintaxe 3. Remover operador definido pelo usuário

A cláusula **USING** permite a especificação da função de similaridade associada ao operador. Esta função deve receber dois parâmetros (do mesmo tipo dos operandos à esquerda e à direita) e retornar um booleano que é *verdadeiro*, caso os valores sejam similares; *falso*, caso contrário. Geralmente, a função de similaridade retorna um valor numérico então, ela compara o valor obtido com uma variável de sessão pré-definida para decidir se retorna verdadeiro ou falso. A cláusula **RENAME TO** define um outro nome de operador para um operador de similaridade pré-existente.

Um novo operador de similaridade $\sim==$, que representa a métrica de similaridade *Levenshtein* pode ser definido como no exemplo 4.

```
CREATE OPERATOR ~== (character varying , character varying)
USING levenshtein(character varying , character varying)
```

Exemplo 4. Criar operador de similaridade Levenshtein

Da mesma forma, os comandos *ALTER OPERATOR* e *DROP OPERATOR* podem ser utilizados como descrito no exemplo 5.

```
ALTER OPERATOR ~== (character varying , character varying) RENAME TO @!@
DROP OPERATOR ~~~ (character varying , character varying)
```

Exemplo 5. Alterar e remover operadores definidos pelo usuário

Sejam $col1$ e $col2$ colunas do tipo *character varying* e WHERE $col1 \sim== col2$ o predicado de uma consulta . O SGBDR deve ser capaz de identificar que $\sim==$ é um operador (definido pelo usuário) e percorrer o mesmo caminho de um operador pré-definido.

O registro dos operadores definidos pelo usuário é feito no catálogo do SGBDR junto com os operadores pré-definidos. Não há necessidade de ter dois caminhos de execução para o processamento das operações especificadas, já que o registro dos operadores e funções definidas pelo usuário (UDFs) são feitos no mesmo local (catálogo) e da mesma forma (syntax) sem distinguir se um objeto é pré-definido ou não. O processamento de uma consulta que utiliza operadores no predicado fará a leitura do catálogo para associar cada operador com a UDF que executa aquela operação.

Após a definição do operador, o identificador $\sim==$ pode ser utilizado para denotar a função *levenshtein*($col1$, $col2$) onde $col1$ e $col2$ são do tipo *cadeia de caracteres*².

2.2. Variáveis de Sessão

Uma sessão SQL está associada a uma conexão. Uma sessão SQL se estende ao longo da execução de uma sequência de consecutivos comandos SQL. O tempo de vida de alguns itens tais como variáveis de sessão, tabelas temporárias e comandos preparados estão ligados a duração da sessão.

Variáveis de sessão são parâmetros associados a uma sessão. O padrão SQL utiliza o comando SET para permitir a mudança dessas variáveis. Em uma sessão SQL podemos querer definir fuso horário local (SET TIME ZONE), usuário (SET ROLE) ou esquema (SET SCHEMA). Estas variáveis podem assumir valores diferentes ao longo da duração da sessão. Isso permite o controle de certas operações do SGBDR incluindo alterar o funcionamento de módulos tais como planejador, otimizador, fuso horário ou caminho de busca de objetos para suprir a necessidade do usuário.

```
SET SCHEMA 'foo';
SELECT a,b FROM bar; — referencia foo.bar
SET SCHEMA 'baz';
SELECT a,b FROM bar; — referencia baz.bar
```

Exemplo 6. Definindo valor da variável de sessão

Além das variáveis de sessão definidas pelo padrão SQL, alguns SGBDRs permitem que novas variáveis de sessão sejam definidas pelo usuário. Esta funcionalidade permite que variáveis possam ser definidos pelos módulos adicionais (extensões) do SGBDR.

No exemplo 7 definimos um novo limiar para o operador de similaridade *Levenshtein* com o comando SET em tempo de execução.

```
SET levenshtein_threshold TO 0.8;
SELECT a,b FROM bar WHERE a ~== b; — retorna 10 registros
SET levenshtein_threshold TO 0.5;
SELECT a,b FROM bar WHERE a ~== b; — retorna 10 ou mais registros
```

Exemplo 7. Definindo valor para variável de sessão

²No padrão SQL:2008, os tipos cadeia de caracteres são *character* e *character varying*

3. PGSimilarity

O *PGSimilarity* foi desenvolvido para validar a infra-estrutura proposta na seção 2. O protótipo implementa uma biblioteca contendo operadores de similaridade que fazem interface com o SGBDR PostgreSQL para dar suporte a consultas aproximadas. O *PGSimilarity* é constituído basicamente por quatro componentes (figura 1):

- **Funções de Similaridade.** Um conjunto de funções que implementam os algoritmos de similaridade. Estas funções podem ser utilizadas como UDF em comandos SQL e, servirão como base para a implementação de operadores de similaridade;
- **Operadores.** Um conjunto de operadores definidos para as funções de similaridade disponíveis. Este componente utiliza as funções de similaridade disponíveis para fazer o cálculo da similaridade e, para isso, utiliza um limiar, que está disponível na sessão. O limiar é uma variável de sessão que pode ser alterada em tempo de execução;
- **Variáveis de Sessão.** Variáveis que controlam parâmetros das funções de similaridade. Essas variáveis controlam, por exemplo, o quanto flexível é o casamento de *strings* e qual é a função utilizada para produzir os *tokens* (se este for o caso). Todas as variáveis podem ser alteradas em tempo de execução;
- **Interface a Método de Acesso.** Um conjunto de definições que especificam como um determinado operador pode utilizar um método de acesso para acelerar as consultas aproximadas.

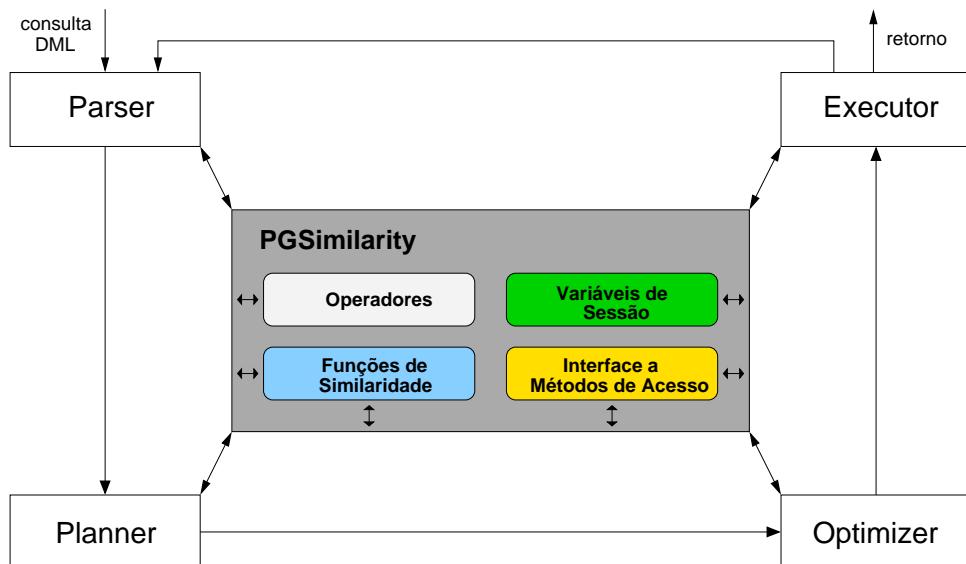


Figura 1. Arquitetura do *PGSimilarity*

3.1. Funções de Similaridade

As funções de similaridade são o principal componente da arquitetura do *PGSimilarity*; são elas que determinam: (i) como os dados serão comparados, (ii) como os atributos serão armazenados e consultados no método de acesso e (iii) quais os dados serão retornados. Todos os comandos SQL enviados ao SGBDR são analisados pelo *parser* e, caso haja alguma cláusula que envolva operações de similaridade, as funções de similaridade correspondentes serão executadas.

As funções de similaridade disponibilizadas pelo *PGSimilarity* são funções definidas pelo usuário (*UDF*). Elas podem ser classificadas em três categorias:

- **funções baseadas em distância de edição.** Funções de distância produzem um valor real r para cada par de *strings* $\langle s, t \rangle$, onde um valor pequeno indica que há grande semelhança entre s e t . Estas funções associam um custo às operações de edição que convertem s em t . Estas operações são: inserção, remoção, substituição e inversão. A distância (ou custo final) é dada pela soma dos custos das operações realizadas para conversão de s em t .
- **funções baseadas em tokens.** Duas *strings* s e t podem ser consideradas conjuntos de *tokens* (palavras, *substrings*, expressões, entidades nomeadas, dentre outras). O cálculo da similaridade consiste em considerar o número de *tokens* em comum (ou não) e utilizar algum critério pré-estabelecido.
- **funções híbridas.** São funções que utilizam as duas abordagens definidas acima. Geralmente estas funções separam os *tokens* das *strings*, aplicam um algoritmo de distância de edição em cada par de *tokens* e depois realizam algum cálculo envolvendo os valores obtidos pelo algoritmo de distância de edição nos pares de *tokens*.

Os algoritmos utilizados no *PGSimilarity* foram selecionados dentre os mais abordados na literatura [Cohen et al. 2003]. Os algoritmos implementados no *PGSimilarity* estão descritos na tabela 1.

Block Distance	Cosine	Dice Coefficient
Euclidean Distance	Hamming	Jaccard
Jaro	Jaro-Winkler	Levenshtein
Matching Coefficient	Monge-Elkan	Needleman-Wunsch
Overlap Coefficient	Q-Gram	Smith-Waterman
Smith-Waterman-Gotoh		

Tabela 1. Algoritmos de similaridade utilizados no PGSimilarity

3.2. Variáveis de Sessão

O *PGSimilarity* possui várias variáveis de sessão que controlam particularidades das funções de similaridade. Todas as funções necessitam de pelo menos dois parâmetros. Uma variável que permite alterar o limiar no qual consideramos casamentos ou não e outra variável que alterna o tipo de saída da função de similaridade (normalizada ou não).

3.3. Interface a Método de Acesso

Alguns algoritmos de similaridade possibilitam a utilização de um índice para acelerar as buscas. Esses algoritmos geralmente são aqueles cujas funções são baseadas em *tokens* e funções híbridas [Arasu et al. 2006] [Chaudhuri et al. 2003] [Sarawagi and Kirpal 2004] [Xiao et al. 2008] [Xiao et al. 2009] [Gravano et al. 2001].

O *PGSimilarity* escolheu um método de acesso flexível e eficiente para as métricas de similaridade utilizadas. Este método de acesso utiliza uma estrutura de dados que combina árvore B+ e lista invertida [Faloutsos and Oard 1995].

A combinação de árvore B+ e lista invertida (figura 2) é uma estrutura de dados no molde de uma árvore B+ cujas chaves (números) são o vocabulário da lista invertida e os dados (folhas) são as ocorrências da lista invertida. As ocorrências de cada chave ficam armazenadas em uma lista encadeada.

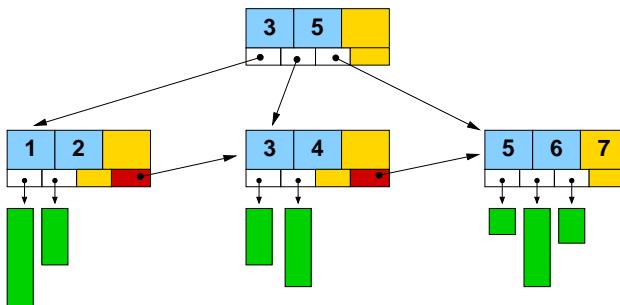


Figura 2. Combinando árvore B+ e lista invertida

3.4. Outros Aspectos Relacionados à Implementação

O protótipo *PGSimilarity* foi desenvolvido em C como uma extensão do SGBDR PostgreSQL. A versão atual do protótipo conta com suporte a 16 funções de similaridade (tabela 1). A instalação, uso e os dados utilizados nos exemplos são apresentados na página do projeto ³.

Os novos comandos *CREATE OPERATOR*, *ALTER OPERATOR* e *DROP OPERATOR* não precisaram ser definidos já que o SGBDR utilizado (PostgreSQL) implementa esses comandos (com uma sintaxe ligeiramente diferente da que apresentamos na seção 2 mas a funcionalidade é idêntica).

Para cada algoritmo de similaridade, definimos duas UDFs: (i) uma que faz o cálculo da similaridade e (ii) outra que utiliza a primeira mas normaliza o valor de saída ($[0, 1]$), e testa se a saída normalizada é maior do que o limiar definido na variável de sessão (retornando verdadeiro, caso contrário, retornando falso).

Para cada algoritmo de similaridade, definimos duas UDFs: (i) uma que faz o cálculo da similaridade e (ii) outra (idêntica a primeira) mas que retorna um valor booleano. A segunda UDF é utilizada pelo operador de similaridade. O valor booleano retornado por ela é o resultado da comparação do valor calculado com o limiar definido na variável de sessão. Caso o retorno seja verdadeiro, a tupla fará parte do resultado da consulta.

Haverá pelo menos duas variáveis de sessão para cada algoritmo de similaridade. Essas variáveis controlam aspectos como normalização do valor de saída, limiar de casamento de *strings* e função que separa os *tokens* da *string*.

O exemplo 8 ilustra o uso do operador Levenshtein (`==`) e a mudança do limiar em tempo de execução. Observe que além do operador `==`, utilizamos a função Levenshtein para atestar que somente pares de *strings* cujo valor de saída é superior ao limiar são retornados. Note que o exemplo também apresenta uma junção aproximada entre as tabelas *foo(a)* e *bar(b)*.

³<http://pgsimilarity.projects.postgresql.org/>

```

mydb=# show pg_similarity.levenshtein_threshold;
pg_similarity.levenshtein_threshold

```

```

0.7
(1 row)

mydb=# select a, b, lev(a,b) from foo, bar where a ~== b;
      a           |       b       |   lev
-----+-----+-----+
 Euler          | Euller      | 0.833333
 Euler Taveira de Oliveira | Euler T. de Oliveira | 0.76
(2 rows)

```

```

mydb=# set pg_similarity.levenshtein_threshold to 0.5;
SET
mydb=# select a, b, lev(a,b) from foo, bar where a ~== b;
      a           |       b       |   lev
-----+-----+-----+
 Euler          | Euller      | 0.833333
 Oiler          | Euller      | 0.5
 Euler Taveira de Oliveira | Euler T. de Oliveira | 0.76
(3 rows)

```

Exemplo 8. Uso do operador Levenshtein

O exemplo 9 ilustra o uso de vários operadores (Jaro-Winkler, QGram e Levenshtein) com o mesmo valor de limiar (o valor padrão é 0.7).

```

mydb=# select * from bar where b ~@@ 'euler'; — jaro-winkler operator
      b
-----+
 Euler T. de Oliveira
 Euller
(2 rows)

mydb=# select * from bar where b ~~~ 'euler'; — qgram operator
      b
-----+
(0 rows)

mydb=# select * from bar where b ~== 'euler'; — levenshtein operator
      b
-----+
 Euller
(1 row)

```

Exemplo 9. Uso dos operadores Jaro-Winkler e QGram

4. Trabalhos Relacionados

Muitos trabalhos foram dedicados ao desenvolvimento de soluções eficientes para o problema de casamento de *strings* aproximadas em um SGBDR. Um excelente levantamento dos trabalhos nesta área é apresentado em [Navarro 2001].

[Cohen 1998] apresenta um framework para a integração de banco de dados heterogêneos baseado em similaridade de *strings*. Além disso, ele propõe o WHIRL que foi construído utilizando o modelo vetorial e é uma ferramenta externa ao SGBDR. O objetivo é o mesmo mas a proposta é bem diferente da descrita neste artigo.

[Gravano et al. 2001] explora a construção de junções aproximadas sem a necessidade de modificar o SGBDR. Ele utiliza a própria infra-estrutura do SGBDR para armazenar dados intermediários (q-grams) e uma série de filtros na consulta para diminuir o número de execuções da UDF envolvida.

[Gravano et al. 2003] utiliza uma estratégia de junção aproximada baseada em amostras sem a utilização de recursos adicionais (UDFs); nesse caso, apenas SQL é utilizado para processar a junção.

[Borges 2006] propõe algumas funções de similaridade no PostgreSQL mas diferente da proposta deste artigo, só há uma variável de sessão e ela é utilizada para controlar o limiar de todas as funções de similaridade. Essas funções foram disponibilizadas sob o nome de PgSimilar.

[Ferreira and Traina Jr. 2008] apresenta operadores de similaridade para implementar operações de seleção e junção. O foco deste trabalho são dados complexos (imagens e áudio) enquanto o deste artigo são dados primitivos (texto).

5. Considerações Finais

Este artigo apresentou a ferramenta *PGSimilarity*, uma extensão à um SGBDR, que permite a representação e execução de consultas aproximadas sobre dados primitivos armazenados em um banco de dados relacional. A ferramenta permite executar operações, tanto de seleção quanto de junção, baseadas na similaridade dos atributos. Além disso, o suporte ao uso de um método de acesso para acelerar as consultas também está disponível para alguns operadores de similaridade.

Esta ferramenta possibilita o desenvolvimento de vários trabalhos futuros. Dentre eles podemos destacar: a implementação de novos algoritmos de similaridade, o comparativo com outras propostas, a implementação de indexação para funções baseadas em distância de edição e a implementação de funções que calculam a seletividade para operadores e junções.

Referências

- Arasu, A., Ganti, V., and Kaushik, R. (2006). Efficient exact set-similarity joins. In *VLDB '06: Proceedings of the 32nd international conference on Very large data bases*, pages 918–929. VLDB Endowment.
- Borges, Eduardo Nunes; Dorneles, C. F. (2006). Pgsimilar: Uma ferramenta open source para suporte a consultas por similaridade no postgresql. In *Simpósio Brasileiro de Bancos de Dados*, pages 1–6.
- Chaudhuri, S., Ganjam, K., Ganti, V., and Motwani, R. (2003). Robust and efficient fuzzy match for online data cleaning. In *SIGMOD '03: Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 313–324, New York, NY, USA. ACM Press.

- Codd, E. F. (1970). A relational model of data for large shared data banks. *Commun. ACM*, 13(6):377–387.
- Cohen, W. W. (1998). Integration of heterogeneous databases without common domains using queries based on textual similarity. In Haas, L. M. and Tiwary, A., editors, *SIGMOD Conference*, pages 201–212. ACM Press.
- Cohen, W. W., Ravikumar, P. D., and Fienberg, S. E. (2003). A comparison of string distance metrics for name-matching tasks. In Kambhampati, S. and Knoblock, C. A., editors, *IIWeb*, pages 73–78.
- Faloutsos, C. and Oard, D. W. (1995). A survey of information retrieval and filtering methods. Technical Report CS-TR-3514, University of Maryland.
- Ferreira, M. R. P. and Traina Jr., C. (2008). Suporte a consultas por similaridade unárias em sql. Msc, Institute of Mathematics and Computer Sciences (ICMC) / University of São Paulo (USP).
- Gravano, L., Ipeirotis, P. G., Jagadish, H. V., Koudas, N., Muthukrishnan, S., and Srivastava, D. (2001). Approximate string joins in a database (almost) for free. In *Proceedings of the 27th International Conference on Very Large Databases (VLDB 2001)*, pages 491–500.
- Gravano, L., Ipeirotis, P. G., Koudas, N., and Srivastava, D. (2003). Text joins in an rdbms for web data integration. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 90–101, New York, NY, USA. ACM Press.
- Navarro, G. (2001). A guided tour to approximate string matching. *ACM Comput. Surv.*, 33(1):31–88.
- Oracle (2011). Oracle database sql language reference 11g release 1. http://download.oracle.com/docs/cd/B28359_01/server.111/b28286/statements_6004.htm. Acessado em 25 de fevereiro de 2011.
- PostgreSQL (2011). Postgresql 9.0 documentation. <http://www.postgresql.org/docs/9.0/static/sql-createoperator.html>. Acessado em 25 de fevereiro de 2011.
- Sarawagi, S. and Kirpal, A. (2004). Efficient set joins on similarity predicates. In *SIGMOD '04: Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 743–754, New York, NY, USA. ACM.
- Xiao, C., 0011, W. W., Lin, X., and Shang, H. (2009). Top-k set similarity joins. In *ICDE 2009: Proceedings of the 25th International Conference on Data Engineering*, pages 916–927. IEEE.
- Xiao, C., Wang, W., and Lin, X. (2008). Ed-join: an efficient algorithm for similarity joins with edit distance constraints. *Proc. VLDB Endow.*, 1(1):933–944.