Organização de dados multimídia para busca conceitual baseada em ontologias*

Luciano B. de Paula
Department of Computer Engineering and Industrial Automation (DCA)
School of Electrical and Computer Engineering (FEEC)
State University of Campinas (UNICAMP)
P.O. Box 6.101 - 13083-970 - Campinas - SP - Brazil
luciano@dca.fee.unicamp.br

Rodolfo S. Villaça
Department of Engineering and Computing (DECOM/CEUNES)
Federal University of Espírito Santo (UFES)
Rod. BR101N, km 60 - São Mateus - ES - Brazil
rodolfovillaca@ceunes.ufes.br

Maurício F. Magalhães
Department of Computer Engineering and Industrial Automation (DCA)
School of Electrical and Computer Engineering (FEEC)
State University of Campinas (UNICAMP)
P.O. Box 6.101 - 13083-970 - Campinas - SP - Brazil
mauricio@dca.fee.unicamp.br

ABSTRACT
The current available large volume of multimedia data, semantically annotated, from several different sources, generates new issues about storing and retrieval of those data. In this scenario, it is common to use simple ontologies to classify data, creating a relationship between them, that relationship may be measured by a similarity index. The contribution of this paper is to propose a way to organize multimedia data by its conceptual classification, using LSH (Locality Sensitive Hashing) functions, facilitating the conceptual search in P2P networks.

RESUMO
Atualmente, a existência de um grande volume de dados multimídia, semanticamente anotados, provenientes de diversas fontes, gera a necessidade de novas formas para armazená-los e obtê-los. Neste cenário, é comum a utilização de ontologias simples na classificação de dados, criando uma relação conceitual entre eles, que pode ser medida por um índice de similaridade. A contribuição deste trabalho é propor uma forma de organizar dados multimídia conceitualmente, utilizando funções LSH (*Locality Sensitive Hashing*) baseadas na classificação conceitual do dado, facilitando a busca conceitual em redes P2P.

Organization of multimedia data for conceptual search based on ontologies

Categories and Subject Descriptors
H.3.3 [**Information Systems**]: Information Search and Retrieval; E.2 [**Data**]: Data Storage Representations

General Terms
Funções LSH

Keywords
Semântica, Web Semântica, busca conceitual

1. INTRODUÇÃO
O crescimento da Web Semântica proporciona, a cada dia, o aumento no volume de dados semanticamente anotados disponíveis na Internet [2]. Fotos, desenhos, vídeos, filmes, músicas, sons, etc, são alguns dos exemplos dos diversos tipos de mídia possíveis de serem obtidos por intermédio de uma classificação semântica [1]. Por meio dessa classificação, definida utilizando-se padrões como RDF, OWL, entre outros, é possível fazer com que as máquinas entendam os dados, não apenas por seus tipos e conteúdos, mas também pelo que representam (foto de uma praia, vídeo de uma partida de futebol, texto de uma notícia sobre política brasileira, página *web* de uma empresa, etc).

Vários trabalhos encontrados na literatura procuram relacionar os dados levando em consideração somente seus conteúdos. Dessa forma, imagens são relacionadas de acordo com suas características (histograma, formas geométricas presentes) [7, 11], textos são relacionados de acordo com suas palavras-chaves [18, 10], etc. Acima desse nível de similaridade, há um outro, no qual os dados, independentemente de suas naturezas (imagens, vídeos, textos, etc), podem ser agrupados de acordo com suas classificações dentro de um domínio de conhecimento, geralmente definido por um conjunto de conceitos. Por exemplo: um texto sobre futebol, um vídeo de uma partida de futebol e uma foto de um lance em um jogo de futebol podem estar agrupados, a partir de uma classificação de domínio, em um mesmo conceito, por exemplo, *"Futebol"*. Isso possibilita a *Busca Conceitual* [14, 12, 5], cujo objetivo é recuperar dados classificados sob um mesmo conceito pertencente a um domínio de conhecimento. Esse

tipo de busca se mostra mais abrangente que a busca, comumente utilizada, por palavras-chaves, como indicado em [14]. Na busca por palavras-chaves é necessária a existência de tais palavras, ou sinônimos, no próprio dado (por exemplo palavras em um texto, anotações em uma imagem, etc), para que esse seja recuperado. Na busca por um conceito, isso não é necessário, já que, independente do conteúdo do dado, esse pode estar classificado em um determinado conceito de domínio, tornando-o alvo da busca.

Uma das maneiras de organizar conceitos para classificação semântica de conteúdos é utilizar ontologias simples (também conhecidas na literatura por ontologias *ligthweight*, hierarquia de conceitos *IS-A*, entre outros nomes) para descrever domínios de conhecimentos. Seguindo o exemplo anterior, o conceito *"Futebol"* pode ser uma subcategoria do conceito *"Esportes"*, e ambos podem estar contidos em uma ontologia que define o domínio *"Entretenimento"*.

Na literatura, são encontrados vários trabalhos que analisam a similaridade entre conceitos de uma ontologia, como os mostrados em [4] e [9], apresentando maneiras de determinar um índice que os relacione. Essa análise é feita sem levar em consideração a forma com a qual os dados classificados pela ontologia são armazenados e obtidos, atividades que poderiam ser facilitadas ao se utilizar esse índice de similaridade na busca por dados semanticamente similares.

Por outro lado, há trabalhos que se importam exatamente com o armazenamento e recuperação de dados [18, 7], sem se importar com a sua classificação, mas tão somente com o seu conteúdo. Por exemplo, imagens são consideradas similares pelo seu conteúdo e não pelo que retratam. Esses trabalhos apresentam melhorias na busca de dados similares sem se preocupar com a relação conceitual entre eles.

Este trabalho utiliza aspectos das duas abordagens anteriores visando a organização de um sistema para *Busca Conceitual*. A contribuição deste trabalho é, em um primeiro instante, apresentar como criar identificadores para conjuntos de dados multimídia, utilizando funções *hash* sensíveis à localidade, que mantenham a classificação conceitual e possibilite a relação com outros dados similares no nível conceitual. Em seguida, é proposta a utilização desses identificadores na indexação dos dados em uma rede P2P estruturada, de forma que a busca por elementos similares, do ponto de vista da classificação conceitual, seja facilitada.

O restante deste artigo está organizado da seguinte forma: a Seção 2 define alguns conceitos utilizados neste trabalho e apresenta o cenário de aplicação. A Seção 3 apresenta como é possível extrair, de uma ontologia, um índice de similaridade entre os seus conceitos. A Seção 4 introduz as funções de *hash* sensíveis à localidade (LSH), apresentando dois exemplos, enquanto que, na Seção 5, é proposta uma função LSH que utiliza a similaridade entre conceitos de uma ontologia. A Seção 6 mostra resultados obtidos em alguns testes para validação da proposta e a Seção 7 conclui o artigo.

2. CONCEITOS E CENÁRIO

As próximas subseções apresentam os conceitos básicos deste trabalho e o seu cenário de aplicação.

2.1 Conceitos básicos

Antes de apresentar uma proposta de organização de dados conceitualmente classificados, é preciso definir, para o escopo deste trabalho, alguns elementos:

- **dado**: corresponde a qualquer estrutura que possa ser armazenada e recuperada em um sistema computacional. Imagens, textos, áudio, páginas *web*, etc, são exemplos de dados;
- **metadado**: anotações atreladas aos dados que definem a sua classificação a partir de uma hierarquia de conceitos, taxonomia, ontologia, etc. Geralmente, são escritos em RDF, RDF-S ou OWL. Neste texto, será usado o termo dado tanto para o dado em si, quanto para o seu metadado, salvo quando indicado;
- **repositório de dados**: toda máquina (ou conjunto de máquinas) que contenha dados ou metadados, de diversos tipos, semanticamente anotados segundo uma determinada classificação com a intenção de compartilhá-los. Exemplos possíveis: máquinas pessoais com músicas e vídeos, grupos de servidores de dados de uma universidade com dados multimídia, repositórios de arquivos RDF da Web Semântica, etc.
- **ontologia**: no contexto deste artigo, ontologia consiste em uma estrutura de dados que define um domínio formado por conceitos e suas relações, possibilitando o raciocínio e a inferência sobre certo termo da ontologia em relação a outros termos [13]. Por meio de uma ontologia, é possível a agregação de termos similares ou mesmo a especialização de um termo em outros. Utilizaremos ontologias simples, também conhecidas na literatura por ontologias *ligthweight*, hierarquia de conceitos *IS-A*, dentre outros nomes, na qual as relações entre conceitos são do tipo *"é um"* e *"parte de"*. No decorrer do texto, a palavra *ontologia* fará referência a esse tipo de estrutura.

Esses elementos são utilizados no cenário de aplicação apresentado na próxima subseção.

2.2 Cenário de aplicação

A Figura 1 ilustra a proposta de organização dos dados apresentada neste trabalho visando um esquema de busca conceitual. Essa proposta é descrita a seguir.

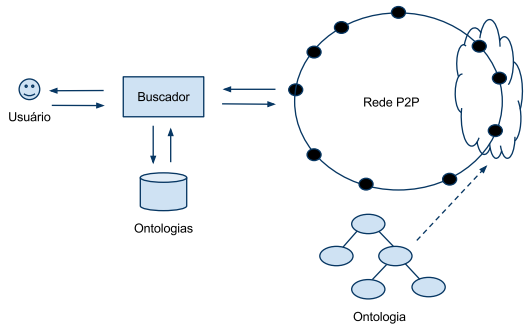

Figura 1: Organização do sistema

Diversos repositórios disponibilizam dados multimídia (imagens, textos, áudio, etc) sobre assuntos variados. Cada repositório emprega ontologias simples para a classificação dos dados que possui. São utilizadas pequenas ontologias, com dúzias ou centenas de termos, para facilitar a gerência e uso destas. Essas ontologias descrevem domínios de conhecimento que definem aqueles dados.

A indexação é feita em uma rede P2P estruturada utilizando-se DHT (*Distributed Hash Table*), geralmente empregada no compartilhamento de grandes volumes de dados. A indexação é feita de maneira que a similaridade entre os conceitos das ontologias seja levada em consideração. Dados similares são indexados próximos uns aos outros no espaço de endereçamento da DHT, por meio de chaves de conceito, facilitando a obtenção de dados relacionados pela redução do número de saltos necessários para obtê-los na DHT.

Um cenário para uso da busca conceitual, como proposto neste artigo, pode ser ilustrado por um usuário que, ao fazer uma consulta no sistema, por meio de um portal de busca, indica o conceito sobre o qual ele queira pesquisar. A determinação desse conceito pode ser feita: (i) de forma explícita, na qual o usuário indica, a partir de uma ontologia, qual o conceito de interesse para a busca; ou (ii) de forma implícita, na qual o conceito buscado e a ontologia associada são determinados por meio de analisadores de linguagem natural e raciocinadores. Essas ontologias são obtidas em um repositório de ontologias. A forma implícita está além do escopo deste trabalho. Ainda no momento da consulta, o usuário pode também escolher um nível de similaridade, entre 0 e 1, o qual ele admite que seja usado na sua busca (ou esse valor pode ser intrínseco ao sistema).

O conceito buscado é traduzido em uma chave de conceito indexada na rede P2P, retornando ao usuário informações armazenadas com aquela chave, como o localizador dos repositórios que possuem dados classificados naquele conceito ou o próprio metadado indicando a URI de onde obtê-lo. Dados inseridos com chaves de conceitos similares também são retornados, respeitando o índice de similaridade pedido pelo usuário ou intrínseco ao sistema. É importante citar que nem todo dado relacionado aos conceitos buscados deve necessariamente ser retornado ao usuário. O resultado da busca pode ser utilizado por um outro serviço, responsável pela filtragem dos dados obtidos, para melhor atender a requisição do usuário.

Uma das formas mais comuns de criação de chaves para redes P2P é utilizar uma função de *hash* (MD5, SHA-1, etc). O valor obtido pelo *hash* passa a ser identificador daquele conteúdo na rede P2P, por meio do qual o dado é obtido, em um esquema de *put* e *get*. Os dados são inseridos pela primitiva *put(k, v)* e obtidos pela primitiva *get(k)*, onde k é o identificador do dado e v é um valor atrelado ao identificador (o dado propriamente dito, um metadado, um localizador de onde o dado pode ser obtido, etc). Geralmente, as redes P2P estruturadas são construídas sobre DHTs (*Distributed Hash Tables*), as quais podem ser organizadas de diversas formas do ponto de vista topológico. Neste trabalho, foram consideradas DHTs baseadas no *Chord*[15], que organizam os nós em um anel virtual, topologia comum dentre as redes P2P estruturadas disponíveis na literatura. A extensão da proposta apresentada neste artigo para outras topologias de DHTs será tema de trabalhos futuros da nossa proposta.

Uma característica decorrente da utilização de funções comuns de *hash* na criação dos identificadores dos dados é a total falta de relação semântica entre o valor obtido pelo *hash* e o dado que o originou devido à natureza aleatória de tais funções. Uma alternativa é a utilização de funções *hash* sensíveis à localidade (*Locality Sensitive Hash* - LSH)[8, 3]. Essas funções, sempre atreladas a uma função de similaridade, geram valores que carregam as características dos dados que os originaram. Dessa forma, dados classificados de forma similar possuem grande probabilidade de gerarem valores de *hash* próximos em uma determinada métrica ou, até mesmo, idênticos.

Em diversos trabalhos na literatura [18, 7], há a preocupação da manutenção da relação entre dados semanticamente parecidos quando armazenados em alguma estrutura de indexação, seja centralizada (bases de dados) ou distribuída (redes P2P, DHTs, etc). Um problema com a abordagem desses trabalhos é que utilizam funções LSH atreladas à similaridade do conteúdo dos dados para gerar os identificadores para cada dado, e não para um grupo, dificultando a busca conceitual.

Neste trabalho, visando exatamente a busca conceitual, é utilizada a classificação conceitual para gerar uma função LSH atrelada à similaridade entre os conceitos. Dessa forma, conceitos similares são armazenados próximos no espaço virtual de uma rede P2P, facilitando a busca por dados relacionados. A Figura 1 mostra que os identificadores de conceitos de uma ontologia são armazenados em uma mesma região do espaço virtual.

A próxima seção apresenta como é possível extrair similaridade entre conceitos de uma ontologia simples. Essa similaridade é a base da função LSH proposta neste artigo.

3. SIMILARIDADE BASEADA EM ONTOLOGIAS

Vários trabalhos encontrados na literatura, tais como apresentados em [4] e [9], mostram formas de se extrair similaridade de ontologias. Entretanto, a abordagem utilizada é baseada no conteúdo dos dados. Por exemplo, [16] utiliza ontologias para descrever textos, baseando-se na *WordNet*[1], uma ontologia de substantivos da língua inglesa.

Este trabalho aborda a busca conceitual, por meio da qual o dado é classificado de acordo com o que representa, e não com o seu conteúdo, sendo essa classificação feita por meio de uma ontologia *lightweight*. Seguindo a proposta da Web Semântica, podemos citar, como exemplo, vários projetos que utilizam classificação semântica por hierarquia de conceitos para diversos domínios diferentes. Por exemplo, *MusicMoz*[2], *Musicbrainz*[3], *Yahoo!*[4] são alguns sites que possuem uma classificação conceitual para definir o domínio *"Música"*. A Figura 2 apresenta uma ontologia baseada em

[1] http://wordnet.princeton.edu
[2] http://musicmoz.org - hierarquia de conceitos
[3] http://musicbrainz.org - tags relacionadas
[4] http://dir.yahoo.com - diretório de categorias

alguns dos conceitos encontrados no diretório de categorias do *Yahoo!*. Essa classificação pode ser utilizada para a indexação de textos, vídeos, imagens sobre música ou até mesmo arquivos de áudio. Neste trabalho, essa hierarquia de conceitos é utilizada na determinação da similaridade entre seus conceitos. Essa extração visa a criação de uma função LSH que relacione os conceitos e facilite a indexação e recuperação.

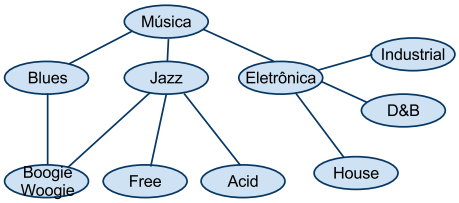

Figura 2: Ontologia *lightweight* para o domínio *"Música"*

Neste trabalho utilizou-se o mesmo algoritmo encontrado em [13] para gerar vetores de conceitos para cada um dos conceitos de uma ontologia. A similaridade de cada conceito pode ser medida pelo cálculo do cosseno entre cada par de vetores. Quanto maior o valor do cosseno, mais relacionados são os conceitos. A maneira de se extrair os vetores de conceito é mostrada a seguir.

Para todo conceito folha de uma ontologia, como a apresentada na Fig. 2, o vetor de conceito $\overrightarrow{\tau_i} = \{\tau_{i,1}, \tau_{i,2}, ..., \tau_{i,k}\}$, onde k é o número total de conceitos da ontologia, é calculado da seguinte maneira:

$$\tau_{i,k} = \begin{cases} 1 & \text{se } \tau_k \in \text{ ao caminho de } \tau_i \text{ até a raiz ou } i = k \\ 0 & \text{caso contrário} \end{cases}$$

Para todo conceito interno da ontologia, o vetor de conceito é calculado da seguinte forma:

$$\overrightarrow{\tau_i} = |\sum \overrightarrow{\tau_s}|$$

onde $\overrightarrow{\tau_s}$ são todos os vetores de conceitos dos filhos diretos de τ_i. Para cada conceito da ontologia da Fig. 2, os vetores obtidos normalizados e aproximados são:

$\overrightarrow{Musica}$ = (0,72, 0,28, 0,46, 0,26, 0,28, 0,1, 0,1, 0,08, 0,08, 0,08)
$\overrightarrow{Blues}$ = (0,5, 0,5, 0,5, 0, 0,5, 0, 0, 0, 0, 0)
$\overrightarrow{Jazz}$ = (0,642, 0,194, 0,642, 0, 0,194, 0,224, 0,224, 0, 0, 0)
$\overrightarrow{Eletronica}$ = (0,654, 0, 0, 0,654, 0, 0, 0, 0,22, 0,22, 0,22)
$\overrightarrow{Industrial}$ = (0,577, 0, 0, 0,577, 0, 0, 0, 0, 0, 0,577)
$\overrightarrow{BW}$ = (0,5, 0,5, 0,5, 0, 0,5, 0, 0, 0, 0, 0, 0, 0)
$\overrightarrow{Free}$ = (0,577, 0, 0,577, 0, 0, 0,577, 0, 0, 0, 0)
$\overrightarrow{Acid}$ = (0,577, 0, 0,577, 0, 0, 0, 0,577, 0, 0, 0)
$\overrightarrow{House}$ = (0,577, 0, 0, 0,577, 0, 0, 0, 0,577, 0, 0)
$\overrightarrow{DB}$ = (0,577, 0, 0, 0,577, 0, 0, 0, 0, 0,577, 0)

Para esse exemplo, algumas similaridades calculadas pelo cosseno do ângulo entre os vetores são mostradas na Tabela 1.

Tabela 1: *Cosseno entre alguns dos vetores de conceito*

	$\overrightarrow{Musica}$	$\overrightarrow{Blues}$	$\overrightarrow{Jazz}$	$\overrightarrow{Elet}$	$\overrightarrow{BW}$	$\overrightarrow{Free}$
$\overrightarrow{Musica}$	1	0,899	0,855	0,736	0,899	0,738
$\overrightarrow{Blues}$	0,899	1	0,848	0,635	1	0,717
$\overrightarrow{Jazz}$	0,855	0,848	1	0,625	0,848	0,872
$\overrightarrow{Elet}$	0,736	0,635	0,625	1	0,635	0,567
$\overrightarrow{BW}$	0,899	1	0,848	0,635	1	0,717
$\overrightarrow{Free}$	0,738	0,717	0,872	0,567	0,717	1

Dessa forma, é possível identificar quais conceitos são mais ou menos similares entre si, em ontologias *lightweight*. Isso possibilita que, por exemplo, nas buscas por dados classificados como *"Jazz"*, se for necessária uma resposta exata, essa deve conter somente elementos classificados por esse conceito. Entretanto, se houver uma tolerância, por exemplo de 0,8 ou mais de similaridade, a busca pode ser respondida com elementos classificados como *"Acid"*, *"Free"*, *"Blues"*, *"Boogie Woogie"* e até mesmo *"Música"*, se forem admitidos conceitos mais amplos. Esse índice de similaridade pode ser fornecido pelo usuário ou intrínseco ao sistema, inclusive podendo ser usado como fator de classificação dos resultados. É importante notar que, pela forma com a qual a extração dos vetores de conceito de ontologias *lightweight* é feita, para a ontologia da Figura 2, o conceito *"Blues"* e seu subconceito *"Boogie Woogie"* possuem similaridade igual a 1. Isso acontece devido ao fato de o conceito mais genérico, no caso *"Blues"*, possuir somente um subconceito, o que os relaciona totalmente (considerando ontologias *lightweight* com relações simples como *"é um"*).

A próxima seção apresenta as funções de *hash* do tipo LSH. Essas funções permitem manter, com certa probabilidade, a similaridade dos dados nos valores dos identificadores gerados, as quais utilizaremos como índices no armazenamento e na busca em uma DHT.

4. FUNÇÕES HASH SENSÍVEIS À LOCALIDADE (LSH - *LOCALITY SENSITIVE HASH*)

Em [8], é definido que uma família de funções de *hash* F é classificada como sensível à localidade (LSH) correspondente a uma função de similaridade *sim(a, b)*, se, para toda função $h \in F$, temos:

$$\mathbf{Pr}_{h \in F}[h(a) = h(b)] = sim(a, b)$$

onde **Pr** é a probabilidade e *sim(a, b)* é uma função de similaridade que varia entre 0 e 1, sendo 1 para a e b completamente relacionados e 0 caso contrário.

As próximas subseções apresentam alguns exemplos de funções de similaridade e funções de *hash* LSH correspondentes.

4.1 *Permutações independentes Min-wise*

As permutações independentes *Min-wise* (*Min-wise independent permutations*) [6][18] provêem uma família de funções

LSH correspondente à função de similaridade de Jaccard, definida como:

$$sim(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

onde A e B são conjuntos de inteiros.

Sendo π uma permutação randômica no universo de inteiros I, $A = \{a_1, a_2, ..., a_n\} \subseteq I$, e $B = \{b_1, b_2, ..., b_n\} \subseteq I$, a função de *hash* h_π é definida como:

$$h_\pi(A) = min\{\pi(a_1), \pi(a_2), ..., \pi(a_n)\}$$

onde $h_\pi(A)$ aplica a permutação π em cada elemento de A e considera somente o menor dos resultados. Para dois conjuntos A e B, temos que $x = h_\pi(A) = h_\pi(B)$ somente se $\pi^{-1}(x) \in (A \cap B)$. Portanto, para dois conjuntos A e B:

$$\mathbf{Pr}_{h \in F}[h(A) = h(B)] = sim(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

4.2 *Funções Hash Random Hyperplane*

Outra função de similaridade que pode ser empregada é o cosseno do ângulo entre dois vetores. Sendo θ o ângulo, em radianos, entre os vetores $\overrightarrow{u}$ e $\overrightarrow{v}$, se $\cos\theta(\overrightarrow{u}, \overrightarrow{v}) = 1$, ambos os vetores são completamente relacionados e, se $\cos\theta(\overrightarrow{u}, \overrightarrow{v}) = 0$, não há relação entre eles.

Uma família de funções LSH que correspondem à similaridade de cosseno é a função *Random Hyperplane Hash* (RHH) [3]. Dada uma coleção de vetores em R^d, um vetor aleatório $\overrightarrow{r}$ é escolhido a partir de uma distribuição gaussiana d-dimensional (cada coordenada é sorteada de uma distribuição gaussiana 1-dimensional - uma distribuição Normal). Para este vetor $\overrightarrow{r}$, a função de *hash* é definida como:

$$h_{\overrightarrow{r}}(\overrightarrow{u}) = \begin{cases} 1 & \text{se } \overrightarrow{r}.\overrightarrow{u} \geq 0 \\ 0 & \text{se } \overrightarrow{r}.\overrightarrow{u} < 0 \end{cases} \quad (1)$$

Para vetores $\overrightarrow{u}$ e $\overrightarrow{v}$, [3] prova que

$$\mathbf{Pr}[h_r(\overrightarrow{u}) = h_r(\overrightarrow{v})] = 1 - \frac{\theta(\overrightarrow{u}, \overrightarrow{v})}{\pi} \quad (2)$$

mostrando a relação entre (2) e $\cos\theta(\overrightarrow{u}, \overrightarrow{v})$. A função RHH produz um único bit. Portanto, para criar chaves mais longas, é necessário produzir vários vetores $\overrightarrow{r}$ e concatenar os resultados, compondo uma sequência binária. Para cada posição na sequência binária, vetores similares têm alta probabilidade de produzirem um resultado idêntico.

A próxima seção apresenta como gerar uma função LSH baseada na similaridade entre conceitos de uma ontologia, combinando as funções acima apresentadas.

5. LSH BASEADA EM ONTOLOGIA

A criação dos identificadores é feita basicamente em dois passos. Primeiro, para uma ontologia *lightweight*, são extraídos os vetores de conceito utilizando a técnica apresentada na Seção 3. Em seguida, cada vetor de conceito é utilizado em uma função LSH, gerando um valor de *hash* para cada conceito da ontologia. Conceitos similares possuirão valores de *hash* próximos. Mais detalhes serão dados nas próximas subseções.

5.1 Criação dos identificadores

A geração de chaves de conceito é feita extraindo vetores de conceito para cada conceito da ontologia e aplicando a função RHH (Seção 4.2) nesses vetores. Para o RHH, são gerados n vetores $\overrightarrow{r}$, sendo cada coordenada destes vetores sorteada de uma distribuição Normal com média 0 e variação 1. Uma chave de n bits é obtida para cada conceito pelo produto escalar dos vetores $\overrightarrow{r}$ com os vetores de conceito. Os n resultados são concatenados compondo a chave.

Como apresentado na Seção 4.2, duas chaves de conceito têm probabilidade p de possuírem o mesmo valor para os bits em uma mesma posição, sendo p determinada pela equação (2). Esse fato faz com que, mesmo para duas chaves de conceitos muito similares, exista a probabilidade de serem armazenadas distantes uma da outra no espaço virtual de uma DHT. Isso acontecerá se pares de bits, em uma mesma posição, da parte mais significativa dos identificadores não possuírem o mesmo valor. Tentando reduzir esse impacto, nossa estratégia consiste em gerar várias chaves para cada conceito e usar a que possui o menor valor lexicográfico. Dessa forma, escolhemos a chave que possui o maior prefixo com "0"s como chave do conceito. A probabilidade é alta de que, para conceitos similares, a menor chave criada seja a mesma ou com valores muito próximos. Isso pode ser explicado pela similaridade de Jaccard usada no *Min-wise*: cada grupo de m identificadores de conceito pode ser visto como um conjunto de inteiros, fazendo com que a probabilidade dos conceitos similares possuírem o mesmo menor valor seja a mesma explicada na Seção 4.1.

Vários trabalhos na literatura, como [18, 6], ao utilizarem funções LSH, indexam cada dado várias vezes na DHT. Isso é feito na tentativa de aumentar a probabilidade de dois identificadores de conceitos similares serem armazenados próximos. Considerando o aumento do volume de dados semanticamente anotados na Web, acreditamos que essa abordagem não é a melhor, devido à sobrecarga de identificadores no sistema. Nossa proposta consiste em criar vários identificadores para cada conceito, mas utilizar somente o menor deles na indexação. Dessa forma, não há sobrecarga no sistema, à medida que o número de conceitos indexados aumenta. Nossa proposta de função de *hash* LSH pode ser definida da seguinte forma:

- para cada conceito de uma ontologia *lightweight*, é extraído um vetor de conceito;
- m grupos de n vetores $\overrightarrow{r}$ são criados, cada coordenada sorteada de uma distribuição Normal;
- para cada vetor de conceito, é aplicada a função (1) para cada vetor $\overrightarrow{r}$ de um grupo, gerando uma chave

de n bits. Esse processo é feito m vezes, uma para cada grupo de m vetores.

- A menor chave dentre as m geradas é escolhida como chave de conceito.

Quanto maior o valor de m, mais próximos os conceitos são armazenados uns dos outros. Dependendo das necessidades do sistema de indexação, m pode ser ajustado para agregar mais ou menos os dados indexados. Mais detalhes na Seção 6.

Um problema dessa abordagem é que ela leva em conta somente a topologia da ontologia, fazendo com que duas ontologias que possuam a mesma topologia, mas que definam domínios totalmente diferentes, gerem chaves de conceito idênticas. A próxima subseção apresenta como adicionar o vocabulário da ontologia na criação dos identificadores, de modo a evitar que se use somente a topologia das ontologias na determinação do índice de conceito.

5.2 Criação dos prefixos

O uso de prefixos nos identificadores dos conceitos distancia a indexação de ontologias que possuam a mesma topologia, mas que definem domínios diferentes. Ao mesmo tempo, aproxima, no espaço virtual, duas ontologias com topologias diferentes, mas que definem o mesmo domínio. Para essa finalidade, propomos compor a chave de conceito em duas partes. A primeira é um prefixo gerado a partir do vocabulário da ontologia, também criado por uma função LSH. O sufixo é obtido como apresentado na Subseção 5.1.

O prefixo é criado utilizando a função LSH Min-Wise (Seção 4.1). Essa função é aplicada a um vetor de vocabulário, criado para cada ontologia, composto pelos nomes de todos os conceitos. Duas ontologias similares terão grandes chances de possuírem o mesmo prefixo, dada a similaridade entre seus vocabulários. A geração do prefixo é feita da seguinte maneira:

- Para cada ontologia, um vetor de vocabulário v é criado, composto pelos nomes de todos os conceitos presentes na ontologia;
- v é convertido em um vetor de inteiros v' por meio de uma função de *hash* tradicional(MD-5, SHA-1, etc.) para cada elemento de v;
- O vetor v' é utilizado em uma função Min-wise, processo descrito na Seção 4.1;
- O resultado do Min-wise de v', ou seja, o menor inteiro gerado após as permutações, é usado como prefixo em todos os identificadores de conceitos daquela ontologia.

Dessa forma, ontologias similares, que definam o mesmo domínio, por exemplo *"Música"*, gerarão o mesmo prefixo e serão armazenadas na mesma região do espaço virtual da DHT. Ontologias que definam outros domínios, por exemplo *"Esporte"*, serão armazenadas em outras regiões do espaço virtual, balanceando o espaço de endereçamento da DHT. A

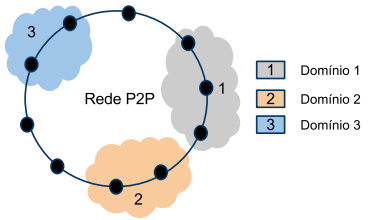

Figura 3: Distribuição dos domínios na rede P2P

Figura 3 mostra uma possível distribuição de três domínios diferentes em uma rede P2P.

A próxima seção apresenta alguns testes feitos e os resultados obtidos. Os testes visam mostrar o comportamento da agregação dos identificadores gerados pela função LSH no espaço virtual de uma DHT e os ganhos obtidos com essa abordagem.

6. AVALIAÇÃO

Os testes feitos objetivam mostrar o impacto da agregação e os ganhos proporcionados pela função LSH atrelada a uma ontologia, em um espaço de indexação de uma DHT baseada no *Chord*. Nesse tipo de DHT, os nós são organizados em um anel virtual, em que cada nó é responsável pelo armazenamento de identificadores com valores entre o identificador do próprio nó e o identificador do vizinho anterior no anel. Os testes utilizaram identificadores de 128 bits e a ontologia mostrada na Figura 2. Devido à falta de trabalhos na literatura que abordam a indexação de dados multimídia de forma conceitual, utilizando funções LSH, este artigo não contém testes comparativos.

O primeiro teste consistiu em, variando-se o valor de m (Seção 5.1), constatar o seu impacto na agregação das chaves. A Figura 4 mostra, para m variando em 1, 5, 10, 20 e 50, a faixa de distribuição dos identificadores em um espaço virtual de 128 bits em forma de anel, dividido em 2000 partes iguais, cada parte representando um nó em uma DHT. A faixa é calculada pelo número de nós existentes entre o primeiro e o último que armazenam identificadores no espaço virtual. Nesse teste, o prefixo mostrado na Seção 5.2 não foi utilizado, já que o intuito é mostrar o impacto na agregação somente pela variação do valor de m.

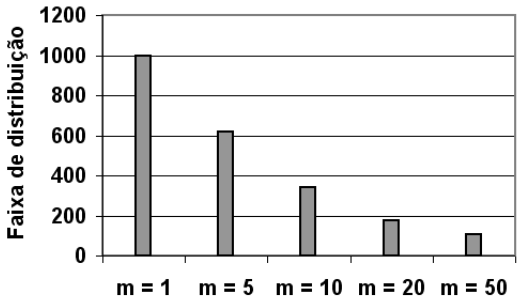

Figura 4: Impacto do m na faixa de distribuição

A Figura 4 mostra a média de 10 testes em cada coluna. Como pode ser visto, quanto maior o valor de m, menor é

a faixa de distribuição dos identificadores, ou seja, os identificadores são armazenados mais próximos uns dos outros. Isso mostra que m é um fator que pode ser ajustado de acordo com a necessidade do sistema, caso seja necessário uma maior ou menor agregação dos identificadores. Os intervalos de confiança de 95% são, para m com valores iguais a 1, 5, 10, 20 e 50, respectivamente: 180,48, 188,53, 95,51, 64,52 e 36,94.

A Figura 5 mostra a relação da agregação dos identificadores em relação ao número de nós existentes na DHT e a variação do tamanho do prefixo utilizado nos identificadores. Para esses testes, m foi fixado em 20.

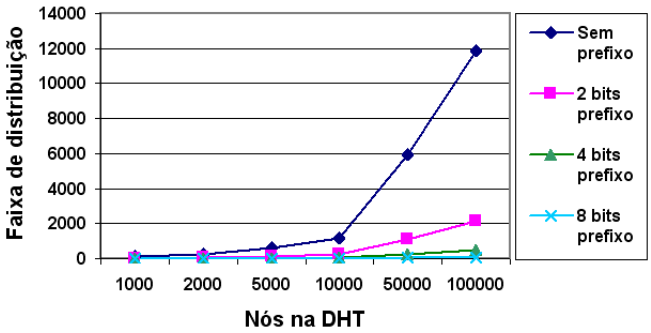

Figura 5: Variação no tamanho da DHT

É possível ver que, quanto maior o tamanho do prefixo, mais agregados os identificadores se encontram e, quanto maior o tamanho da DHT, mais nós compartilham os identificadores de uma mesma ontologia. Para comparação, o mesmo teste feito com uma função tradicional de *hash* (MD5) possui um crescimento linear, ocupando praticamente todo o espaço do anel. Por motivos de escala, este resultado não foi colocado no gráfico.

Esse resultado mostra que o prefixo atua na agregação da seguinte forma: quanto mais bits são usados no prefixo, o espaço virtual da DHT é dividido em um número maior de partes para a indexação de diversas ontologias. Por exemplo, um prefixo de 1 bit divide o espaço virtual em duas partes (a primeira parte para chaves com prefixo igual a "0" e a segunda parte com prefixo igual a "1"). Um prefixo de 2 bits divide o espaço em 4 partes (chaves com prefixo igual a "00", "01", "10" e "11"), e assim por diante. Dessa forma, quanto maior o prefixo usado nas chaves de conceito de uma ontologia, menor será a parte do espaço virtual na qual as chaves dessa ontologia estarão indexadas. O número de nós existentes em cada parte varia de acordo com a distribuição e a quantidade de nós existentes na DHT.

Devemos lembrar que, se, a princípio o resultado obtido com a função LSH pode sinalizar um desbalanceamento no espaço virtual da DHT, o sistema proposto deve indexar diversas ontologias, definindo diversos domínios de conhecimento. Como mostrado na Figura 3, a inserção de várias ontologias no sistema resulta no preenchimento do anel pelas regiões que essas ocuparão.

A Figura 6 mostra a distância, na média, entre o identificador de um dos conceitos em relação a todos os identificadores dos outros conceitos, variando-se o tamanho do prefixo e mantendo o número de nós no espaço virtual em 2000 e o valor de m igual a 20. Nesse teste, o conceito escolhido foi o *"Free"*. Estes valores são comparados também com o mesmo teste feito com o MD5.

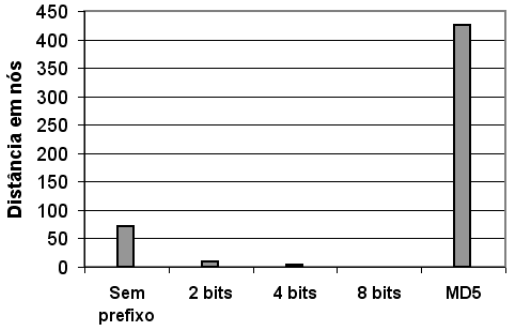

Figura 6: Distância do conceito *"Free"*

Novamente, é possível ver que, quanto maior o tamanho do prefixo utilizado, mais próximo os identificadores dos conceitos são armazenados. A média para prefixos com 8 bits, ou mais, torna-se praticamente 0, o que significa que praticamente todos os conceitos foram armazenados no mesmo nó da DHT. Para os identificadores gerados pela função de *hash* tradicional, a distância entre os conceitos é maior, o que resulta em um maior balanceamento da DHT, porém, em um maior custo na busca por conceitos relacionados, como mostrado no próximo teste. Na Figura 6, os seguintes intervalos de confiança de 95%, para os testes "Sem Prefixo", "2 bits", "4 bits", "8 bits" e "MD5" são, respectivamente: 6,09, 2,41, 0,3, 295,94.

A Figura 7 mostra o número de saltos necessários para, a partir do nó que armazena o identificador de um conceito, atingir todos os outros conceitos da ontologia. Como no teste anterior, o termo escolhido foi o *"Free"* e foi variado o tamanho do prefixo. Neste teste foi empregado o simulador de rede P2P *Oversim*[5], utilizando 2000 nós no anel da DHT. Esse teste foi feito em duas DHTs diferentes: no *Chord* e em uma baseada no fenômeno do *Small World* [17]. A diferença entre as duas DHTs se refere à criação dos *fingers* entre os nós. No *Chord*, o estabelecimento dos *fingers* é feito de forma que não há preferência para sua criação, enquanto que a DHT baseada no fenômeno do *Small World* privilegia a criação de *fingers* entre nós que estejam próximos no espaço virtual. Por esse motivo, DHTs baseadas no fenômeno *Small World* são mais aptas para o armazenamento e recuperação de chaves agregadas, como mostrado em [17].

Devido ao roteamento do *Chord* ser feito no sentido horário, para uma maior justiça nos testes, as chaves foram ordenadas e buscadas na ordem crescente. Ou seja, para uma chave de conceito C_k qualquer, se o valor da chave conceitual de *"Free"* é menor que C_k, os saltos necessários a partir de *"Free"*, com destino à C_k, foram contabilizados. Caso contrário, contabilizaram-se os saltos a partir de C_k com destino à chave do conceito *"Free"*.

Como pode ser visto na Figura 7, em ambas DHTs, o número

[5]http://www.oversim.org

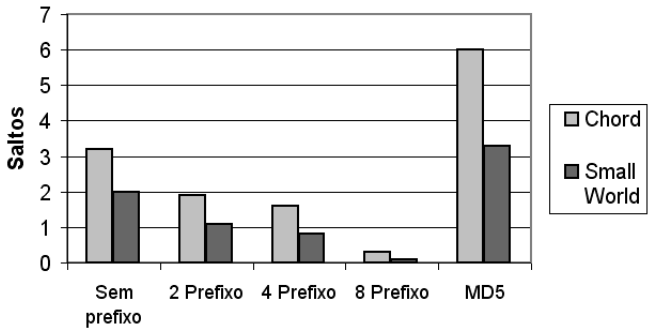

Figura 7: Saltos em relação a *"Free"*

de saltos necessários para atingir todos os identificadores de conceitos a partir do identificador do conceito *"Free"* é menor se comparado com número de saltos necessários para o mesmo teste, mas considerando identificadores criados pelo MD5. Os intervalos de confiança de 95% são, para o *Chord* "Sem Prefixo", "2 bits de Prefixo", "4 bits de Prefixo", "8 bits de Prefixo" e "MD5", respectivamente: 1,77, 1,03, 0,89, 0,59 e 0,86. Para os testes utilizando a DHT *Small World* com "Sem Prefixo", "2 bits de Prefixo", "4 bits de Prefixo", "8 bits de Prefixo" e "MD5", os intervalos de confiança de 95%, respectivamente, são: 0,8, 0,7, 0,44, 0,2 e 0,93.

7. CONCLUSÃO

Este trabalho apresentou uma proposta de organização de repositórios de dados conceitualmente classificados, visando facilitar a busca conceitual. Foi feita uma proposta de criação de funções LSH atrelada à similaridade entre os conceitos de uma ontologia *lightweight* para criar identificadores de conceitos que mantenham essa similaridade. Esses identificadores foram indexados em DHTs e foi mostrado que houve um ganho na recuperação de conceitos similares.

Para trabalhos futuros, pretende-se aprimorar a proposta para novos cenários e investigar outras formas de se criar os identificadores, inclusive avaliando novas formas de relacionar ontologias similares. Pretende-se investigar também novas formas de organização em redes P2P para facilitar ainda mais o armazenamento e busca de dados relacionados.

8. REFERÊNCIAS

[1] Batista, C.E.C.F., Schwabe, D.: Linkedtube - informações semânticas em objetos de mídia na internet. In: Webmedia 2009. Fortaleza, CE (2009)

[2] Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web: A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. In: Scientific American (May 2001)

[3] Charikar, M.S.: Similarity estimation techniques from rounding algorithms. In: STOC '02: Proceedings of the thiry-fourth annual ACM symposium on Theory of computing. pp. 380–388. ACM Press, New York, NY, USA (2002), `http://dx.doi.org/10.1145/509907.509965`

[4] Cordi, V., Lombardi, P., Martelli, M., Mascardi, V.: An ontology-based similarity between sets of concepts. In: WOA'05 (2005)

[5] Formica, A.: Concept similarity in formal concept analysis: An information content approach. Know.-Based Syst. 21(1), 80–87 (2008)

[6] Gupta, A., Agrawal, D., Abbadi, A.E.: Approximate range selection queries in peer-to-peer systems. In: CIDR (2002)

[7] Haghani, P., Michel, S., Cudré-Mauroux, P., Aberer, K.: Lsh at large - distributed knn search in high dimensions. In: WebDB (2008)

[8] Indyk, P., Motwani, R.: Approximate nearest neighbors: towards removing the curse of dimensionality. In: STOC '98: Proceedings of the thirtieth annual ACM symposium on Theory of computing. pp. 604–613. ACM, New York, NY, USA (1998)

[9] K., S.U., N., N.: Ontology based similarity measure in document ranking. International Journal of Computer Applications 1(26) (2010)

[10] Köhler, J., Philippi, S., Specht, M., Rüegg, A.: Ontology based text indexing and querying for the semantic web. Know.-Based Syst. 19(8), 744–754 (2006)

[11] Kulis, B., Jain, P., Grauman, K.: Fast similarity search for learned metrics. IEEE Transactions on Pattern Analysis and Machine Intelligence 31, 2143–2157 (2009)

[12] Lyte, V., Jones, S., Ananiadou, S., Kerr, L.: UK Institutional Repository Search: Innovation and Discovery. Ariadne Issue 61 (2009), `http://www.ariadne.ac.uk/issue61/lyte-et-al/`

[13] Polyvyanyy, A.: Evaluation of a Novel Information Retrieval Model: eTVSM. Master's thesis, Hasso Plattner Institut, Univeristat Potsdm (2007)

[14] Ratinov, L., Roth, D., Srikumar, V.: Conceptual Search And Text Categorization. Technical Report UIUCDCS-R-2008-2932, UIUC, CS Dept. (2008)

[15] Stoica, I., Morris, R., Karger, D., Kaashoek, M., Balakrishnan, H.: Chord: A scalable peer-to-peer lookup service for internet applications. Proceedings of ACM SIGCOMM'01 pp. 149–160 (2001)

[16] Varelas, G., Voutsakis, E., Raftopoulou, P., Petrakis, E.G., Milios, E.E.: Semantic similarity methods in wordnet and their application to information retrieval on the web. In: WIDM '05: Proceedings of the 7th annual ACM international workshop on Web information and data management. pp. 10–16. ACM, New York, NY, USA (2005)

[17] Villaça, R., de Paula, L., Magalhães, M., Verdi, F.L.: Avaliação de redes p2p baseadas no fenômeno small world para o compartilhamento de conteúdos similares gerados por funções lsh. In: VI Workshop de Redes Dinâmicas e Sistemas P2P (WP2P 2010). Gramado, RS (2010)

[18] Zhu, Y.: Enhancing Search Performance in Peer-to-Peer Networks. Ph.D. thesis, Computer Science and Engineering, University of Cincinnati (2005)