Trabalho Prático 3
Recuperação de Informação

Luís E. O. Lizardo[1]

[1]Departamento de Ciência da Computação – Universidade Federal de Minas Gerais
Caixa Postal 702 – 30.123-970 – Belo Horizonte – MG – Brasil

lizardo@dcc.ufmg.br

Resumo. *Máquinas de busca revolucionaram a Internet ao possibilitarem que usuários tenham acesso aos conteúdos mais diversos e de forma rápida. Devido ao grande tamanho da Web, essas máquinas precisam ser eficientes e ainda precisas em relação as respostas retornadas aos usuários. O objetivo deste trabalho é melhorar os resultados das respostas do TP2 expandindo o índice do trabalho anterior para suportar a indexação de textos âncoras e com ánalise de links usando PageRank. Uma avaliação experimental mostrou que a inclusão de textos âncoras melhorou os resultados em relação aos trabalhos anteriores.*

1. Introdução

Máquinas de busca são mecanismos importantes na era da Internet. Elas permitem procurar por páginas Web tendo como base palavras chaves ou expressões, e retornam de forma eficiente uma lista de páginas candidatas a resposta da consulta. A forma mais simples de se realizar ou processar uma consulta, é retornar todas as páginas que contém os termos pesquisados, porém, desta forma nem todas as páginas retornadas podem ser relevantes para o usuário, mesmo que elas possuam o termo pesquisado. Ocorre também de inúmeras páginas serem encontradas, e aquela que o usuário realmente precisa, aparecer entre as últimas exibidas, dificultando assim sua pesquisa.

Identificar páginas relevantes a uma consulta em um índice é uma tarefa difícil e incerta, pois uma página considerada relevante para um usuário, pode não ser para outro. O que as máquinas de busca fazem é utilizar modelos de ranqueamento para predizer uma ordem de relevância das páginas encontradas. O objetivo deste trabalho é melhorar os resultados das respostas do trabalho prático anterior adicionando suporte a indexação de textos âncoras e também incorporando ánalises de links usando PageRank [Brin and Page 1998]. Textos âncoras são os textos visíveis em hiperlinks de páginas HTML. Estes textos são importantes pois normalmente descrevem as páginas de destino dos links. PageRank é um algoritmo que mede a importância de uma página contabilizando a quantidade e qualidade dos links apontando para ela.

Neste trabalho a indexação de textos âncoras e a ánalise de links usando o PageRank foram combinados aos modelos desenvolvidos no TP2: BM25 e Cosseno.

2. Textos âncoras

Textos âncoras são os textos visíveis e clicáveis em um hiperlink. As palavras contidas em um texto âncora são muito relevantes para algoritmos de busca, pois o texto que faz referência ao link normalmente descreve a página de destino. Neste trabalho os termos retirados dos textos âncoras são inseridos no mesmo índice que o conteúdo das páginas, porém associado ao documento de destino.

3. PageRank

O PageRank representa a probabilidade de uma pessoa visitar uma página navegando aleatoriamente. Ele é modelado por meio de uma Cadeia de Markov. O PageRank de cada documento é dado pela seguinte fórmula:

$$PR(a) = \frac{q}{T} + (1 - q) \sum_{i=1}^{n} \frac{PR(p_i)}{L(p_i)}$$

onde T é o número total de páginas da coleção e q é o fator de *dumping* ou amortecimento. O Algorítmo 1 descreve a implementação do PageRank.

Algorithm 1 PageRank

G: grafo das páginas e seus relacionamentos
d: fator de dumping
maxIterations: número limite de iterações
maxDelta: maior diferença tolerável para convergência

1: **for all** Documento $d_i \in G$ **do**
2: $PR(d_i) \leftarrow 1/\left|G\right|$;
3: $delta \leftarrow \infty$;
4: **while** $t < maxIterations \wedge delta >= maxDelta$ **do**
5: $delta \leftarrow 0$;
6: **for all** Documento $d_i \in G$ **do**
7: $sum \leftarrow 0$;
8: **for all** Vizinho $d_j \in d_i$ **do**
9: $sum \leftarrow sum + PR(d_j)/L(d_j)$;
10: $rank = d/\left|G\right| + (1 - d) * sum$;
11: $delta \leftarrow \max(delta, abs(PR(d_i) - rank))$;
12: $PR(d_i) \leftarrow rank$;
13: $t \leftarrow t + 1$

O grafo das páginas é mantido em memória principal e computado durante a fase de indexação. A complexidade de tempo do algoritmo é dada pelo número de iterações vezes o número de nós do grafo $O(maxIterations \times E)$ e a complexidade de espaço é dada pelo tamanho do grafo $O(E)$.

4. Adaptação do índice

Algumas ações foram necessárias para extrair os textos âncoras e adicioná-los ao índice, assim como para montar o grafo de conexão de páginas do PageRank.

A primeira ação foi a criação de uma tabela em memória com as URLs de cada documento. Esta tabela é utilizada, após o *parsing*, para verificar se os links encontrados fazem referência a algum documento da coleção. A complexidade de espaço desta tabela é $O(N)$, onde N é o número de documentos da coleção.

Para reduzir o tamanho das URLs, foi utilizado a Função Hash MD5 [Rivest et al. 1992] nas URLs dos documentos e nos links das páginas. O MD5 transforma *strings* de qualquer tamanho em outra *string* de 32 bytes. O MD5 gera sempre para

duas *strings* iguais, a mesma saída. No entanto, uma pequena modificação na *string* de entrada, provoca uma grande alteração nos bytes de saída. O MD5 é suficiente para transformar todas as URLs da coleção sem colisão, pois, segundo o Paradoxo do Aniversário, a primeira colisão é esperada depois de transformar 2^{64} itens, que é muito maior que a coleção. A complexidade de tempo do MD5 é linear no tamanho da *string* de entrada[1].

Outra ação realizada foi a normalização dos links e URLs dos documentos. Esta ação foi tomada para aumentar a taxa de *matching* dos links das páginas com as URLs dos documentos. Além de passar a URL para *lower case*, a normalização também remove *scripts* Javascript, MailTo, FTP etc, e ainda adapta endereços relativos, conforme exemplos da Tabela 1. A complexidade de tempo da normalização de links é $O(T + C)$, onde T é o tamanho do link da página e C é o tamanho do caminho relativo do link, caso exista.

Tabela 1. Tabela com exemplos de modificações realizadas nos links e URLs

URL	Caminho relativo	URL Normalizada
http://www.example.com/index.html	-	http://www.example.com/
www.example.com/search.php#comment	-	http://www.example.com/search.php
www.example.com/page/public/index.html	/form.html	http://www.example.com/form.html
www.example.com/page/public/index.html	./form.html	http://www.example.com/page/public/form.html
www.example.com/page/public/index.html	../form.html	http://www.example.com/page/form.html
www.example.com/page/public/index.html	form.html	http://www.example.com/page/public/index.html
www.example.com/page/public/index.html	google.com.br/search?q=ufmg	https://www.google.com.br/search?q=ufmg

Os links e os textos âncoras das páginas são extraídos durante o *parsing* e armazenados em um arquivo temporário em disco. Após o término do *parsing* de todos os documentos da coleção, estes links são processados. Para cada link do arquivo temporário, é verificado se ele faz referência (casa) com alguma URL dos documentos da coleção. Este casamento é verificado após a normalização do link e a transformação utilizando a função MD5. Se o link casa com a URL de alguma página da coleção, e se esta página não é a mesma da do link (auto referência), o texto âncora deste link é adicionado ao arquivo temporário de triplas e o link ao grafo do PageRank.

Assim como o conteúdo das páginas, os textos âncoras também passam pelo *parsing* e seus termos são inseridos ao final do arquivo temporário de triplas (termo, documento, frequência), porém com o documento de referência do link. O problema dessa abordagem é que ela pode criar triplas duplicadas no arquivo temporário de triplas, ou seja, triplas de mesmo termo e documento. Para corrigir o problema, durante a fase de ordenação do arquivo temporário, as triplas duplicadas são identificadas e unificadas com suas frequências somadas. Como esta operação é feita durante a ordenação, na saída do Heap, ela não altera a complexidade do algoritmo.

5. Modelos

Nesta seção são apresentados os modelos Cosseno, BM25 e um modelo resultante da combinação dos dois, chamado neste trabalho de Custom.

5.1. Vetorial

O Modelo Vetorial ou Modelo de Espaço Vetorial representa cada documento como um vetor de termos e, cada termo possui um valor associado que indica seu grau de importância (peso) para o documento.

[1] http://daoyuan.li/category/projects/9-md5/

A representação dos documentos dj e da consulta q são vetores t dimensionais dados por:

$$\vec{d_j} = (w_{1,j}, w_{2,j}, ..., w_{t,j})$$
$$\vec{q} = (w_{1,q}, w_{2,q}, ..., w_{t,q})$$

onde

$$w_{i,j} = (1 + \log f_{i,j}) \times \log\left(\frac{N}{n_i}\right)$$
$$w_{i,q} = (1 + \log f_{i,q}) \times \log\left(\frac{N}{n_i}\right)$$

O modelo vetorial avalia o grau de similaridade entre um documento d_j e uma consulta q como uma correlação entre os vetores d_j e q. A correlação é quantificada pelo cosseno do ângulo entre esses dois vetores:

$$sim(d_j, q) = \frac{\vec{d_j} \bullet \vec{q}}{\left|\vec{d_j}\right| \times |\vec{q}|}$$

onde $\left|\vec{d_j}\right|$ e $|\vec{q}|$ são as normas dos vetores que representam a norma do documento e da consulta. Como $|\vec{q}|$ é o mesmo para todos os documentos, ele não afeta o *ranking*, e neste trabalho foi considerado sempre igual a 1.

Nos experimentos apresentados na Seção **??**, foram utilizados 3 tipos diferentes de normalização dos documentos:

- Normalização vetorial, dada por:

$$\vec{d_j} = \sqrt{\sum_{i}^{t} w_{i,j}^2}$$

- Normalização pelo número de termos no documento; e
- Normalização igual a 1.

O algoritmo para calcular as similaridades dos documentos processa as palavras da consulta em ordem e mantem um conjunto de acumuladores para cada documento encontrado. Os acumuladores armazenam o somatório do TF-IDF. Por fim, o valor de cada acumulador e divido pela norma do documento. Neste trabalho os acumuladores são estruturas *Hash*, então a complexidade de tempo e espaço deste modelo é $O(n)$, onde n é o número de documentos encontrados pelos termos pesquisados.

Mais informações sobre o Modelo Vetorial podem ser encontradas em [Baeza-Yates et al. 1999].

5.2. BM25

O modelo BM25 foi criado como resultado de uma série de experimentos sobre variações da fórmula probabilistica clássica:

$$sim(d_j, q) \sim \sum_{k_i \in q \wedge k_i \in d_j} \log\left(\frac{N - n_i + 0.5}{n_i + 0.5}\right)$$

Esses experimentos foram motivados pela observação de que, como no modelo vetorial clássico, uma boa ponderação de termos é baseada em três princípios: (1) frequência inversa de documentos, (2) frequência dos termos e (3) normalização pelo tamanho dos documentos. A formula probabilistica clássica cobre apenas o primeiro princípio.

O BM25 foi motivado pela combinação de fatores de frequência de termos das fórmulas de ranqueamento BM11 e BM15, como segue:

$$ß_{i,j} = \frac{(K_1 + 1) f_{i,j}}{K_1 \left[(1 - b) + b \frac{len(dj)}{avg\_doclen} \right] + f_{i,j}}$$

onde b é uma constante introduzida com valores no intervalo [0,1]. Se $b = 0$, a equação acima é reduzida ao fator de frequência de termos usado na BM15, Se $b = 1$, ela é reduzida ao fator de TF da BM11. Para valores de b entre 0 e 1, a equação fornece uma combinação da BM11 com BM15.

A equação de ranqueamento do modelo BM25 pode ser escrita como:

$$sim_{BM25}(d_j, q) \sim \sum_{k_1[q,d_j]} ß_{i,j} \times log \left(\frac{N - n_i + 0.5}{n_i + 0.5} \right)$$

onde K_1 e b são constantes empíricas na expressão para $ß_{i,j}$. Neste trabalho, o BM25 foi avaliado para $k_1 = 1$ e b iguais a 0, $0,25$, $0,50$, $0,75$ e $1,00$.

O algoritmo para calcular as similaridades deste modelo é similar ao do modelo vetorial. Sendo sua complexidade $O(n)$ em tempo e espaço.

Mais informações sobre o BM25 podem ser encontradas em [Baeza-Yates et al. 1999].

5.3. Custom

Este modelo foi criado por meio de uma combinação dos modelos Vetorial e BM25. Ele é o modelo vetorial normalizado com um peso b, como no modelo BM25. Sua similaridade é dada por:

$$sim(d_j, q) = \frac{\vec{d_j} \bullet \vec{q}}{\left[(1 - b) + b \frac{len(dj)}{avg\_doclen} \right] + f_{i,j}}$$

O algoritmo para calcular as similaridades deste modelo é similar ao do modelo vetorial, linear em tempo e espaço em relação ao número de documentos encontrados.

5.4. Combinação dos modelos com o PageRank

Neste trabalho foi utilizada uma combinação linear dos modelos com o PageRank. Como exemplo, suponha um conjunto de páginas p que satisfazem uma determinada consulta Q. Então, a classificação $R(p, Q)$ das páginas p em relação a consulta Q pode ser computada por:

$$R(p, Q) = \alpha BM25(p, Q) + (1 - \alpha) PR(p)$$

onde $\alpha \in [0, 1]$.

6. Implementação

Este trabalho foi implementado em *C++11*. Os arquivos com cabeçalho e códigos fontes estão organizados em pastas dentro do diretório *src*. Os arquivos principais de execução dos programas estão na raiz desta pasta. Os arquivos de códigos da interface Web estão na pasta *www*.

As bibliotecas de terceiros necessárias para a compilação dos programas são fornecidas juntamente com o código fonte. Elas estão na pasta *lib* e seus cabeçalhos na pasta *include*. As bibliotecas são: Gumbo Parser, OneURL, riCode, CMPH e a biblioteca zlib[2].

6.1. Programas

Os seguintes programas estão disponíveis para execução:

- *index_writer_sorter*: Faz o *parsing* da coleção e cria o arquivo temporário de triplas ordenado.
- *index_query*: Constrói o índice comprimido e processa as consultas escritas em um arquivo passado como parâmetro da execução. Cada linha do arquivo é processada como uma consulta independente. Os resultados de cada consulta são salvos em arquivos separados.
- *index_server*: Constrói o índice comprimido e espera por requisições via *socket*. Cada requisição processada tem os resultados respondidos no formato JSON. Quando este programa é executado, ele primeiramente constrói o índice, que pode ser um processo demorado, para somente depois atender as requisições.

6.2. Compilação

Para compilar os programas e fazer o *link* as bibliotecas execute no Linux:

```
make all
```

6.3. Execução

Para executar um teste dos programas utilizando a coleção *toyExample*, execute no Linux:

```
make run
```

Para executar o *index_server*, execute:

```
make server
```

Para executar os programas individualmente, os seguintes argumentos devem ser passados:

```
./index_writer_sorter [-d -m -n -i -c]
./index_query [-d -t -q -n]
./index_server [-d -t -n]
```

onde as opções são,

-d: diretório de saída, onde os arquivos gerados pela execução serão criados.
-t: nome do arquivo temporário de triplas.

[2]zlib: http://www.zlib.net/

-m: tamanho da memória em MB, se não especificado, 1 MB será utilizado. O tamanho da memória durante a execução tem que ser o mesmo para todos os programas.
-n: número máximo de documentos a serem indexados, se não especificado, no máximo 1 milhão de documentos serão indexados.
-i: diretório da coleção comprimida de documentos.
-c: índice da coleção comprimida de documentos.
-q: caminho completo do arquivo contendo as consultas. Uma consulta por linha.

7. Avaliação Experimental

Os testes foram realizados no sistema operacional Ubuntu 13.10, rodando um notebook com processador 3rd Generation Intel® Core™ i7-3630QM (2.40GHz 6MB Cache), 8GB RAM, HHD 1 TB S-ATA (5,400 rpm). Foram feitos 5 execuções para cada teste e retirado a média. A coleção de documentos utilizada é a mesma do TP2. Os dados de Precisão x Revocação foram calculados com base na lista de documentos relevantes para 34 consultas disponibilizados na especificação do TP2. A interpolação utilizada é a mesma definida em [Baeza-Yates et al. 1999].

7.1. Dados gerais de indexação

Na Tabela 2 são apresentadas estatísticas gerais da indexação com a inclusão dos textos âncoras. Os dados são comparados com os dados obtidos no TP2. É possível notar na tabela que o maior impacto da inclusão dos textos âncoras foi no tempo de construção do arquivo temporário, que teve aumento de 120%. Este aumento é explicado pelo tempo necessário para normalizar e casar as URLs, além do tempo de *parser* dos textos âncoras.

Tabela 2. Estatísticas gerais de indexação com e sem texto âncora

Descrição	Sem texto âncora	Com texto âncora
Número de documentos	945642	945642
Número de termos	3908119	3908437
Tempo médio de construção do arquivo temporário	3600 s	8032 s
Tamanho do arquivo temporário	2.1 GB	2.8 GB
Tempo médio de construção do índice invertido	350 s	371 s
Tamanho do índice invertido comprimido	411.9 MB	412.5 MB

7.2. Dados gerais dos hiperlinks

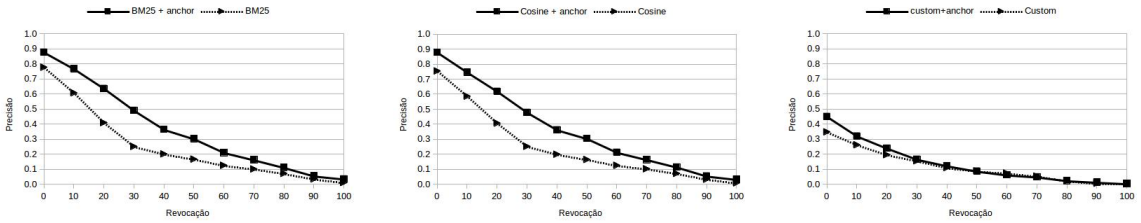
Na Tabela 3 são apresentados alguns dados sobre os hiperlinks extraídos da coleção. Do total de hiperlinks apenas 3,9% foram utilizados no cálculo do PageRank, pois são aqueles que apontam para documentos da coleção e não fazem referência a própria página.

7.3. Textos âncoras

Nesta seção, são comparados os resultados obtidos pelos modelos BM25 (com constante $b = 0$), Cosseno e Custom, desenvolvidos e apresentados no trabalho prático anterior, com a inclusão dos textos âncoras. Conforme esperado e apresentado no gráfico da Figura 1, a inclusão dos textos âncoras melhorou os resultados de Precisão x Revocação dos três modelos.

Tabela 3. Estatísticas gerais dos hiperlinks

Número total de hiperlinks na coleção	52.772.373
Número de hiperlinks que apontam para documentos da coleção	2.743.800 5,2%
Número de hiperlinks que apontam para documentos da coleção excluídos aqueles que apontam para o próprio documento	2.084.412 3,9%



(a) BM25 (b) Cosseno (c) Custom

Figura 1. Precisão x Revocação dos modelos BM25, Cosseno e Custom com a adição de textos.

7.4. PageRank

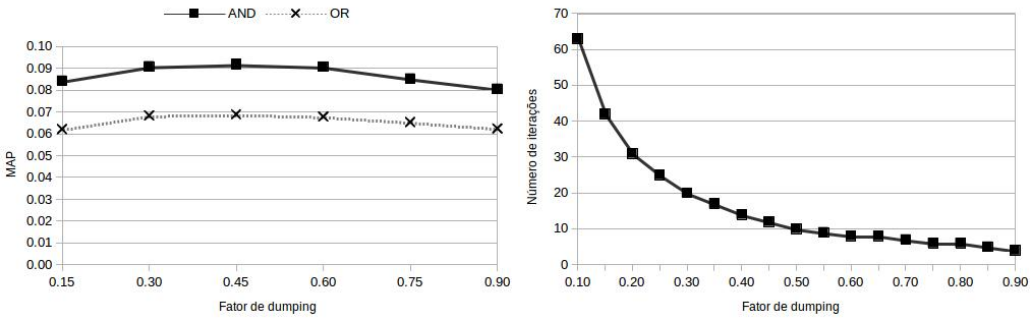
Nesta seção são apresentados os resultados obtidos com as análises do PageRank. Em todos os testes foi utilizado 1^{-9} como maior difereça, delta, para verificar a convergência.

7.4.1. Fator de dumping

Nesta seção é avaliado o impacto do fator de *dumping* ou amortecimento nos resultados do PageRank e também na convergência das iterações. No gráfico (a) da Figura 2 é apresentado a precisão média (MAP) do PageRank para valores de *dumping* variando de 0.15 até 0.90, com incrementos de 0.15. AND e OR representam respectivamente testes feitos com a interseção e união dos documentos recuperados no índice. Conforme o gráfico, o melhor valor para o fator de *dumping* é de 0.45, diferente dos 0.15 sugerido por [Baeza-Yates et al. 1999]. A Figura 2(b) apresenta o número de iterações necessárias para o PageRank convergir em função do fator de *dumping*. Neste experimento o fator varia de 0.10 até 0.90 com incrementos de 0.05. Pelo gráfico apresentado é possível observar que a convergência tem um decaimento quadratico em relação ao fator de *dumping*. Dessa forma, um fator maior faz o PageRank convergir mais rápido.

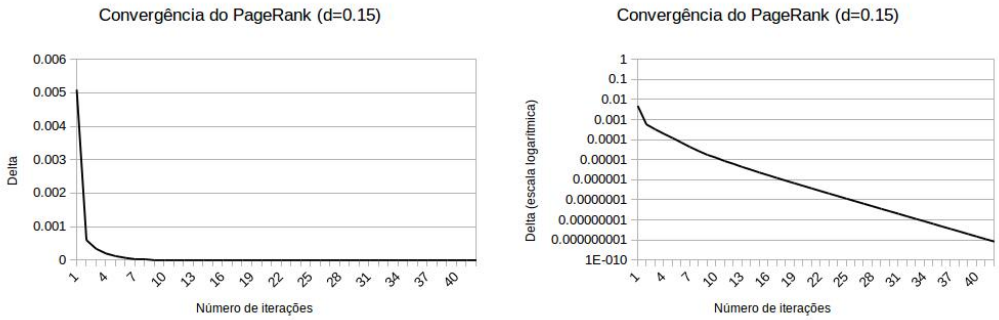
7.4.2. Convergência

Os gráficos da Figura 3 apresenta a taxa de convergência do algoritmo de PageRank. Nesse experimento foi utilizado um fator de *dumping* igual a 0.15. Com pouco mais de 7 iterações o algoritmo se estabiliza e a maior diferença fica abaixo de 1^{-5}.



(a) Fator de dumping (b) Iterações para convergência

Figura 2. Análise do fator de dumping do PageRank.



(a) Escala linear (b) Escala logarítmica

Figura 3. Convergência do PageRank para fator de dumping = 0.15.

7.4.3. Melhores documentos

A Tabela 4 mostra os 10 documentos com maiores valores de PageRank da coleção.

7.4.4. Qualidade das respostas

O gráfico da Figura 4 apresenta a Precisão x Revocação média do PageRank. Dois testes foram realizados, com a interseção (AND) e a união (OR) dos documentos recuperados no índice. O fator de *dumping* utilizado foi de 0.45. Conforme é possível observar no gráfico, o PageRank sozinho apresentou resultados muito ruins quando comparados com os dos demais modelos.

7.5. Combinação dos Modelos

Nesta seção é avaliado o PageRank quando combinado com os modelos BM25 e Cosseno. Para os experimentos realizados, foi utilizado, para o BM25, constante b igual a 0, conforme definido como melhor no TP2, fator de *dumping* do PageRank igual a 0.45.

Tabela 4. Documentos no topo do PageRank

#	Página	PageRank
1	http://www.blogger.com/	0.00601856
2	http://wordpress.com/	0.00453986
3	http://www.blogger.com/features	0.00382695
4	http://www.statcounter.com/	0.00286233
5	http://blogsofnote.blogspot.com/	0.00236366
6	http://openid.net/	0.00195074
7	http://openid.net/2008/12/	0.00165828
8	http://diythemes.com/thesis/	0.000956689
9	http://www.via6.com/sobre.php	0.000705323
10	http://www.via6.com/user_form2.php	0.000703304

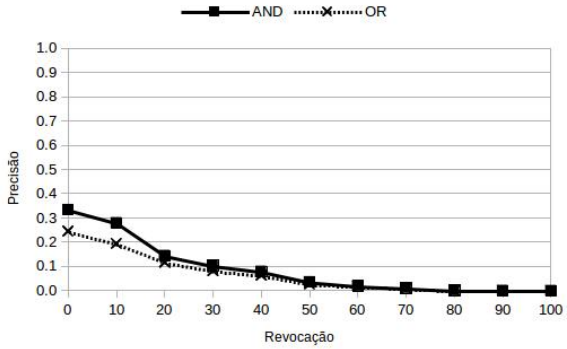


Figura 4. Precisão x Revocação do PageRank considerando interseção (AND) e união (OR) de documentos.

7.5.1. Avaliação do parâmetro Alfa

O gráfico da Figura 5 apresenta uma análise do parâmetro *Alfa* na combinação linear dos modelos BM25 e Cosseno com o PageRank. Conforme pode ser observado no gráfico, um Alfa mais baixo, que valoriza o PageRank, apresenta resultados ligeiramente melhores para o BM25, apesar da variação ter sido pouco significativa. Nos experimentos realizados, foi calculado o MAP para valores de Alfa variando de 0.10 até 0.90, com incrementos de 0.10.

7.5.2. Qualidade das respostas

Na Figura 6 são apresentados dois gráficos com a Precisão x Revocação média obtida pelos modelos BM25 e Cosseno na evolução com a adição do PageRank e do texto âncora. Nos gráficos podemos observar que os dois modelos apresentaram resultados muito parecidos, sendo o BM25 um pouco superior. Outro ponto importante de observação é que as curvas de Precisão x Revocação se mantiveram iguais após a adição do PageRank, que não provocou melhoria nos resultados obtidos anteriormente com a adição do texto âncora.

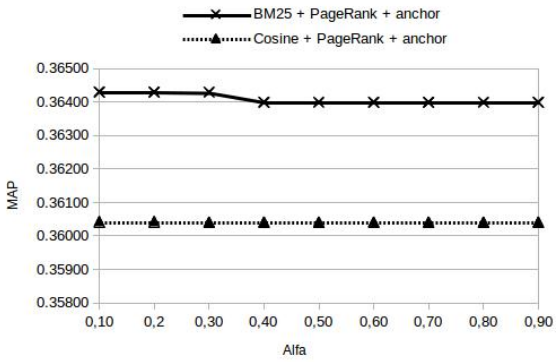
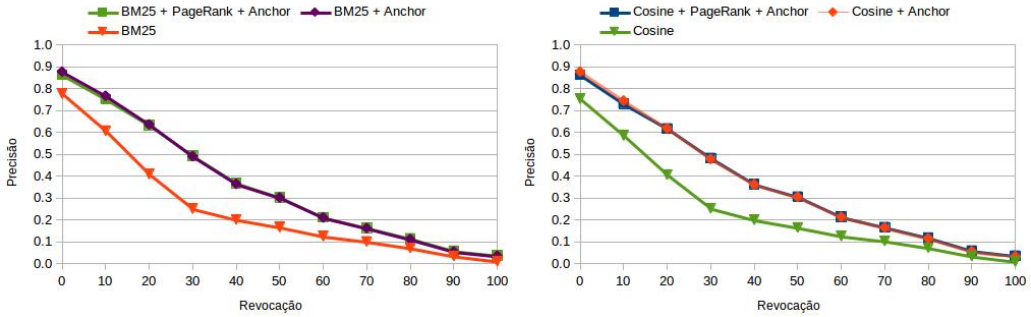


Figura 5. Avaliação do parâmetro Alfa.



(a) BM25 (b) Cosseno

Figura 6. Precisão x Revocação média e evolutiva dos modelos.

As Figuras 7 e 8, no final do documento, apresentam os gráficos de Precisão x Revocação dos modelos BM25 e Cosseno, com texto âncora e PageRank, para cada um dos 34 termos consultados durante os testes. A consulta pelo termo 'pânico' foi a que apresentou os melhores resultados.

8. Conclusão

Neste trabalho foram estudadas e implementadas novas fontes de evidências para processadores de consultas. As evidências estudadas foram o PageRank e textos âncoras. Os modelos BM25 e Cosseno, desenvolvidos no trabalho anterior, foram combinados com essas novas fontes de evidências, e seus resultados analisados.

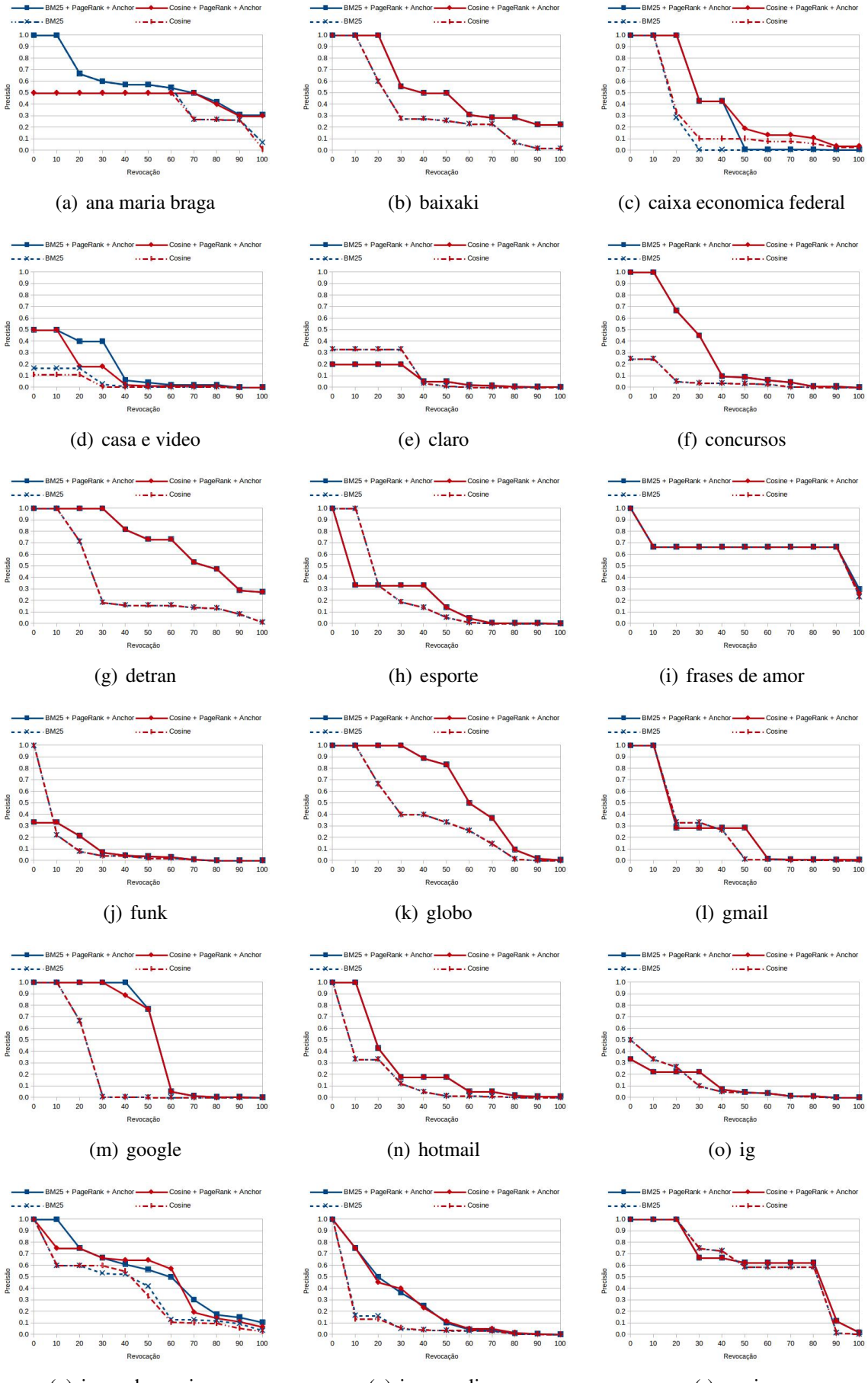
Uma avaliação experimental mostrou que a adição de textos âncoras melhorou significativamente as respostas das consultas. No entanto, o PageRank apresentou pouca melhoria.

Referências

Baeza-Yates, R., Ribeiro-Neto, B., et al. (1999). *Modern information retrieval*, volume 463. ACM press New York.

Brin, S. and Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1):107–117.

Rivest, R. L. et al. (1992). RFC 1321: The MD5 message-digest algorithm. *Internet activities board*, 143.



(a) ana maria braga (b) baixaki (c) caixa economica federal

(d) casa e video (e) claro (f) concursos

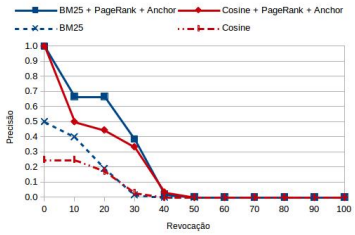
(g) detran (h) esporte (i) frases de amor

(j) funk (k) globo (l) gmail

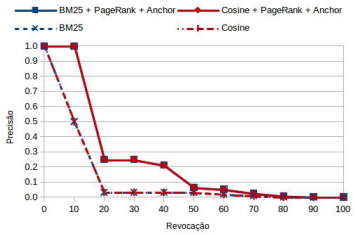
(m) google (n) hotmail (o) ig

(p) jogos de meninas (q) jogos online (r) mario

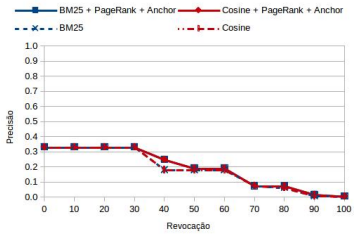
Figura 7. Gráficos de Precisão x Revocação.



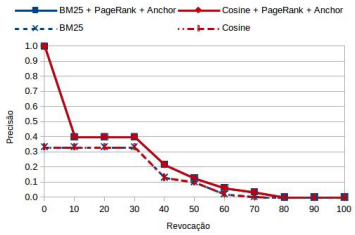
(a) mercado livre



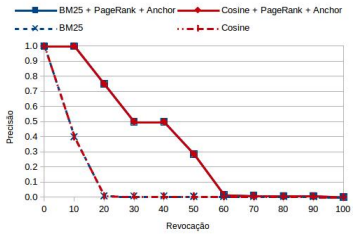
(b) msn



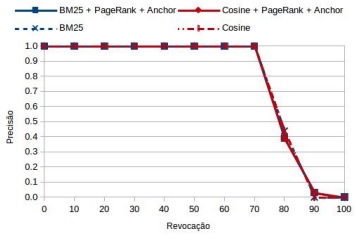
(c) naruto



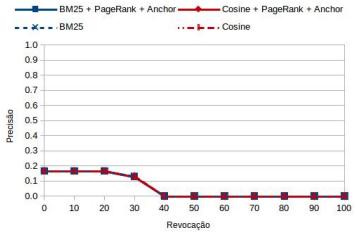
(d) oi



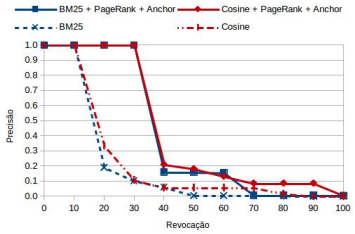
(e) orkut



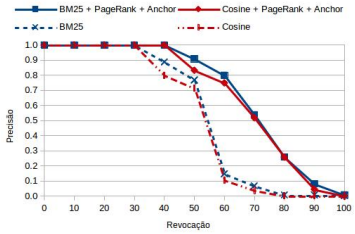
(f) panico



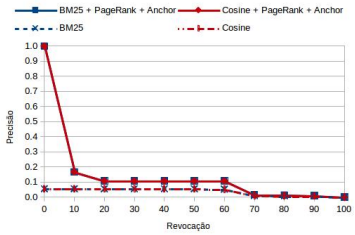
(g) poquer



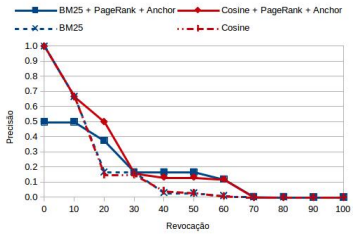
(h) previsao do tempo



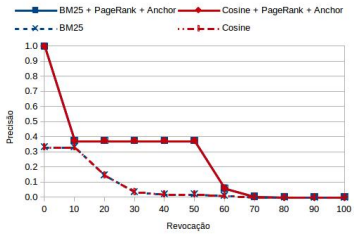
(i) receita federal



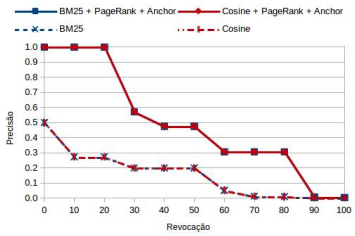
(j) record



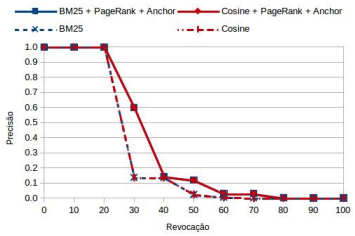
(k) rio de janeiro



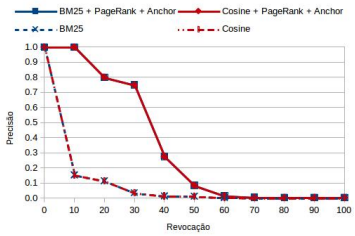
(l) terra



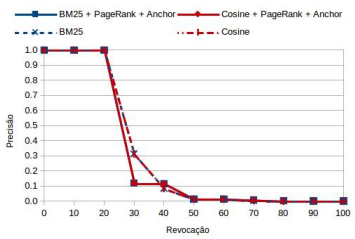
(m) uol



(n) vivo



(o) yahoo



(p) youtube

Figura 8. Gráficos de Precisão x Revocação. (Continuação)