Instituto de Computação
Universidade Estadual de Campinas

Técnicas e algoritmos de *Link Analysis* na geração de medidas de similaridade

Rodrigo Carvalho Rezende

Este exemplar corresponde à redação final da Dissertação devidamente corrigida e defendida por Rodrigo Carvalho Rezende e aprovada pela Banca Examinadora.

Campinas, 26 de Julho de 2012.

Siome Klein Goldenstein (Orientador)

Ricardo da Silva Torres (Co-orientador)

Dissertação apresentada ao Instituto de Computação, UNICAMP, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

FICHA CATALOGRÁFICA ELABORADA POR
MARIA FABIANA BEZERRA MULLER - CRB8/6162
BIBLIOTECA DO INSTITUTO DE MATEMÁTICA, ESTATÍSTICA E
COMPUTAÇÃO CIENTÍFICA - UNICAMP

Rezende, Rodrigo Carvalho, 1981-
R339t Técnicas e algoritmos de link analysis na geração de medidas de similaridade / Rodrigo Carvalho Rezende. – Campinas, SP : [s.n.], 2012.

Orientador: Siome Klein Goldenstein.
Coorientador: Ricardo da Silva Torres.
Dissertação (mestrado) – Universidade Estadual de Campinas, Instituto de Computação.

1. Recuperação da informação. 2. Algoritmos em grafos. 3. Bibliometria. 4. Redes complexas. I. Goldenstein, Siome Klein,1972-. II. Torres, Ricardo da Silva,1977-. III. Universidade Estadual de Campinas. Instituto de Computação. IV. Título.

Informações para Biblioteca Digital

Título em inglês: Link analysis techniques and algorithms for similarity measures
Palavras-chave em inglês:
Information retrieval
Graph algorithms
Bibliometrics
Complex networks
Área de concentração: Ciência da Computação
Titulação: Mestre em Ciência da Computação
Banca examinadora:
Ricardo da Silva Torres [Coorientador]
Edleno Silva de Moura
Jacques Wainer
Data de defesa: 26-07-2012
Programa de Pós-Graduação: Ciência da Computação

TERMO DE APROVAÇÃO

Dissertação Defendida e Aprovada em 26 de Julho de 2012, pela Banca examinadora composta pelos Professores Doutores:

Prof. Dr. Edleno Silva de Moura
DCC / UFAM

Prof. Dr. Jacques Wainer
IC / UNICAMP

Prof. Dr. Ricardo da Silva Torres
IC / UNICAMP

Instituto de Computação
Universidade Estadual de Campinas

Técnicas e algoritmos de *Link Analysis* na geração de medidas de similaridade

Rodrigo Carvalho Rezende

Julho de 2012

Banca Examinadora:

- Ricardo da Silva Torres (Co-orientador)
- Edleno Silva de Moura
  Departamento de Ciência da Computação (DCC), Universidade Federal do Amazonas
- Jacques Wainer
  Instituto de Computação, Universidade Estadual de Campinas (Unicamp)

Resumo

Esta dissertação estuda técnicas de *Link Analysis* para o problema de se calcular similaridade entre artigos acadêmicos organizados em uma biblioteca digital. Neste trabalho construímos um conjunto de dados e desenvolvemos um protocolo experimental para avaliar a eficácia das técnicas desenvolvidas. Para lidar com a alta complexidade dos algoritmos de similaridade para o nosso conjunto de dados, estudamos técnicas de amostragem de grafos e avaliamos objetivamente a qualidade das amostras geradas por estes métodos. A partir deste estudo, propomos um novo algoritmo de amostragem baseado na técnica *Forest Fire*. Experimentos realizados demonstram a superioridade do algoritmo de amostragem proposto. Além disso, apresenta-se uma nova meta-função de similaridade para artigos acadêmicos que considera apenas a informação de citação entre artigos, sem levar em conta o conteúdo textual e seus metadados para dizer o quanto um artigo é similar a outro. Esta meta-função transforma medidas de similaridade locais, como o coeficiente *Jaccard* e *Adamic/Adar*, em medidas recursivas, cuja similaridade depende recursivamente da similaridade de outros artigos relacionados, explorando a ideia de que dois artigos são mais similares na medida em que estão associados a artigos que também são similares. Para avaliação de eficácia do método proposto, criamos um gabarito de similaridade, que deriva da classificação hierárquica dos artigos no sistema de classificação de 1998 da *Association for Computer Machinery* (*ACM*). Este gabarito cria uma noção de similaridade tal que dois artigos são mais similares na medida em que são classificados em classes similares, isto é, que estão em classes hierarquicamente próximas. Experimentos são conduzidos no grafo de citação de artigos, extraído da biblioteca digital da *ACM*, contendo um subconjunto de 122.774 artigos e 523.699 arestas de citações, e comparam esta nova meta função de similaridade com o gabarito de similaridade e revelam que esta gera melhor eficácia que as medidas de similaridade locais consideradas. Além disso, avaliamos esta técnica na atividade prática de busca por exemplo e confirmamos que este meta-algoritmo melhora a eficácia das medidas locais consideradas.

Abstract

This work studies techniques of *Link Analysis* used to address the problem of computing the similarity between academic papers organized in a digital library. We constructed a bibliographic dataset and developed an experimental protocol to evaluate the effectiveness of these techniques. To handle the high complexity of the similarity algorithms applied to our dataset, we study graph sampling techniques and evaluate the quality of the samples generated by these methods. This study lead to the proposal of a new sampling algorithm based on an existing technique named *Forest Fire*. Experiments results demonstrate the superiority of the proposed sampling algorithm. Moreover, we present a new meta-similarity function for scholarly articles that considers only the citation information, which does not take into account their textual content and its metadata, to compute how much an article is similar to another. This meta-function transforms local similarity measures, such as the *Jaccard coefficient* and *Adamic/Adar*, into recursive measures, whose similarity score recursively depends on the similarity of other related articles, exploring the idea that two articles are more similar if they are associated with articles which are also similar. To evaluate the effectiveness of the proposed method, we constructed a *groundtruth* of similarity, which derives from a hierarchical classification system of the *Association for Computer Machinery* (*ACM*). This *groundtruth* creates a notion of similarity such that two articles are more similar if they fall into similar classes (those that are hierarchically close to each other). Experiments are conducted in the citation graph, extracted from the *ACM Digital Library*, containing a subset of 122,774 articles and 523,699 citation edges. Obtained results demonstrate that this new meta-similarity function outperforms baselines. Furthermore, this results are confirmed in other experiments concerning the use of the proposed meta-functions in similarity search tasks.

Agradecimentos

Em 2007 iniciei o processo do que hoje seria este trabalho de mestrado. Naquele momento, recém formado, fui recebido pela Unicamp como aluno especial e Dr. Siome Goldenstein, meu orientador, me ajudou oferendo uma oportunidade remunerada de trabalho em um laboratório de pesquisa. Devo bastante ao Siome por ter feito isso, pois com este emprego consegui realizar meu plano pessoal de me casar e iniciar uma pós-graduação. De 2007 até o final de 2008, estudei a maioria das disciplinas obrigatórias para o mestrado em Ciência da Computação, neste processo conheci meu futuro co-orientador Dr. Ricardo Torres. Depois da primeira, tive de repetir outra disciplina com este professor e já estava certo de que eu queria ser orientado também por ele. Em 2009, me matriculei como aluno regular de mestrado, com Siome e Ricardo como orientadores. Ainda não sei se havia sido uma boa decisão ter feito isso naquele momento, contudo, não havia como eu prever o que estava por vir. Naquele ano, após perder meu emprego, fui contratado pela Odysci, que desenvolve um mecanismo para indexar e buscar artigos científicos. Assim como no emprego anterior, no qual aprendi muito sobre tecnologias de IA de busca (obrigado aos doutores ex-colegas do Harpia), este novo emprego definiria minha área de pesquisa: Recuperação de Informação. Infelizmente naquele ano perdi minha amada Mãe, Izaura. Tive problemas de saúde e financeiro. Com estas dificuldades, este mestrado desacelerou. Naquele momento recebi muita ajuda de meus amigos, minha esposa, Eliane, e meu irmão, Alexandre. Sem eles eu não teria me reerguido e retomado este mestrado. Agradeço ao meu empregador daquela época, Reinaldo, pela sensibilidade demonstrada ao dar o tempo necessário para me reorganizar. Agradeço profundamente a minha esposa que de tanta gratidão e amor, mal consigo encontrar as palavras. Agradeço a grande lealdade de meu irmão e a todas vezes que me ajudou. Agradeço aos meu amigos e profissionais que me escutaram e me ajudaram com suas palavras e visões. Nestes últimos meses, mesmo após vários avanços técnicos e bons resultados em experimentos com o mestrado, considerei desistir. Talvez meu orientador Ricardo não saiba, mas a fé que ele demonstrou em mim e em meu trabalho foi a energia extra que precisei para terminar a escrita desta dissertação. Agradeço bastante por ser esforçar em fazer este trabalho melhor. Gostaria de agradecer a empresa Odysci por me fornecer o conjunto de dados para experimentos, a

Unicamp e ao Instituto de Computação que me ajudou aprender mais sobre computação. À CNPq, FAPESP e CAPES por financiar a infra-estrutura do laboratório RECOD. Agradeço também a banca examinadora e a todos outros que fizeram parte de meu dia a dia e que tornaram minha vida uma experiência mais rica. Dedico este mestrado em memória da minha Mãe. Ela, com sua simplicidade de quem mal estudou, me ensinou as melhores e fundamentais lições de vida. Não deixo de pensar nela sequer um dia.

Sumário

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

Este trabalho de mestrado trata do problema de se calcular similaridade entre artigos científicos para aplicações de recuperação de informação. Há várias formas de se abordar o problema de se calcular a similaridade, a depender do tipo de dados que se tem disponível. Uma abordagem possível seria considerar medidas de similaridades que levam em conta o conteúdo dos artigos e/ou seus metadados, por exemplo, seu título, resumo. Isto é, dado o conteúdo textual de dois artigos, poderíamos ser capazes de dizer o quão similares são. Outra abordagem é considerar técnicas de *Link Analysis* para o cálculo da similaridade. Neste sentido, poderíamos considerar que artigos de uma biblioteca digital são organizados em um grafo, em que citações/referências refletem arestas entre os artigos. Outras entidades como autores, conferências e revistas também poderiam ser representadas neste grafo, formando outras arestas. Nessa abordagem de *Link Analysis*, a ideia seria desconsiderar o conteúdo dos artigos e tomar como relevante para o cálculo da similaridade apenas a topologia do grafo, as arestas que envolvem artigos. Uma terceira abordagem, a híbrida, seria misturar informações da topologia do grafo da biblioteca digital com o conteúdo textual dos artigos. O foco deste trabalho é o problema de calcular a similaridade entre artigos considerando apenas a topologia do grafo da biblioteca digital, isto é, considerando técnicas de *Link Analysis*.

Este trabalho inicialmente apresenta como um grafo de uma biblioteca digital de artigos científicos, para fim dos métodos de similaridade, pode ser organizado. Em seguida, apresentam-se as principais medidas de similaridade de *Link Analysis*, da literatura, que podem ser usadas neste grafo da biblioteca digital. Como será visto adiante, a complexidade de alguns algoritmos de similaridade é alta para grafos reais (incluso aquele que usaremos em nossos experimento). Contudo, como faremos uma comparação dessas medidas para descobrir qual delas obtém melhores resultados, vamos apresentar, comparar e propor técnicas de amostragem de grafos com a finalidade de redução de escala, isto é, torná-los menores (com menos vértices e arestas), para assim viabilizar o estudo com-

parativo dessas medidas de similaridade. Nosso estudo comparativo é prático, ou seja, introduziremos: i) um método de se extrair um gabarito de similaridade a partir de como os artigos são organizados em uma biblioteca digital e ii) uma metodologia de se avaliar estas medidas de similaridade contra este gabarito. Desta forma, compararemos métodos da literatura bem como algumas propostas. Além disso, examinaremos o impacto da direção das arestas neste grafo para cada medida de similaridade avaliada. Apesar de haver várias entidades envolvidas nesse grafo da biblioteca digital (autores, artigos, conferências, etc), consideraremos em nossa comparação, apenas o subgrafo extraído que envolve artigos e suas citações/referências.

O esquema de comparação introduzido avalia as medidas de similaridade independentemente de uma aplicação. Isto é, consegue-se dizer quão promissora é uma técnica sem mesmo vincular a uma aplicação prática. Além disso, construímos um experimento focado na aplicação de busca por similaridade, segundo a qual, a partir de um artigo, encontra os outros mais similares.

1.1 Contribuições

O conteúdo desse trabalho oferece as seguintes contribuições:

1. Uma proposta para amostragem do grafo de citações de artigos;

2. Um meta algoritmo que muda o comportamento de medidas de similaridades existentes;

3. Um esquema para construir um gabarito de similaridade e metodologia de avaliação comparativa das medidas de similaridade;

4. O estudo do impacto da direção das arestas do grafo de citação na similaridade entre artigos;

5. O estudo do impacto das medidas de similaridade de *Link Analysis* na aplicação de busca *Query by Example*.

1.2 Motivação

Nesta seção são apresetados os desafios e aplicações que motivam o estudo de técnicas de *Link Analysis* para o problema da similaridade entre entidades organizadas em um grafo.

1.2.1 Desafios

Muitos dados, muitas aplicações: É crescente a pesquisa relacionada a *Link Analysis* em ciência da computação por diversos fatores: i. redes são ubíquas e há grande disponibilidade de conjuntos de dados estruturados em redes; ii. ao contínuo aumento do poder de processamento para lidar com mais dados; iii. custo decrescente no armazenamento de dados. Exemplos de dados estruturados em redes em que *Link Analysis* já foi aplicado inclui [54]: Redes sociais - contatos e amizades entre indivíduos; Redes de informação ("*information networks*") - grafos de citação acadêmica de artigos e patentes, grafos de *hyperlinks* entre páginas na WEB, ontologias, bases de dados léxicas, redes de preferências (relaciona indivíduos e objetos de sua preferência); Redes tecnológicas - redes energéticas, linhas aéreas e rodoviárias, telecomunicações, circuitos eletrônicos, Internet; Redes biológicas - vias metabólicas, rede de interação entre proteínas na transcrição e translação genética, cadeia alimentar, redes neurais, circulação sanguínea. Destes, redes de informação são objeto de muitas técnicas de *Link Analysis* com aplicações diretas em ciência da computação [28, 68]. Não diferente, o objeto de estudo deste mestrado é a investigação de técnicas e algoritmos de *Link Analysis* na geração de medidas de similaridade em uma rede de informação, no caso, o grafo de uma biblioteca digital de artigos acadêmicos. Similaridade tem um papel central na solução de problemas como classificação, clusterização, *entity resolution* [9, 56], recomendação [26, 27, 65], detecção de padrões etc., sendo, portanto, um campo fértil para aplicações do mundo real.

Grandes dados, desafios de computação: Muitos métodos de *Link Analysis*, no cálculo de similaridade em um grafo $G(V, E)$, sofrem de problemas de escalabilidade, pois são $O\left(|V|^3\right)$ em tempo e $O\left(|V|^2\right)$ em espaço [5,36,71,72], tornando-os de difícil aplicação não só em dados na escala da Web, mas também em dados de literatura cientifica, cujo grafo está na ordem de milhões de nós. Por exemplo, o algoritmo de similaridade de vértices *SimRank* [36] produz matrizes de similaridade pouco esparsas, note que são quadráticas no número de vértices e que dependendo do tamanho do grafo, a implementação deste algoritmo é desafiadora.

A aplicação de *Link Analysis* cria um campo fértil para pesquisas em otimização de algoritmos, métodos de aproximação e processamento distribuído, como aplicados em [5, 23, 24, 43]. Além disso, mesmo sem colocar os esforços nos algoritmos, é desejável examinar na prática a eficácia destes algoritmos, o que abre caminho na pesquisa de como manipular os grafos reais para reduzir a escala ou mesmo na geração de grafos sintéticos que representam situações reais.

Relações semânticas: Estamos trabalhando em um cenário em que há diferentes tipos de objetos em uma rede (autores, artigos, conferências). Assim, compará-los pelo seu

conteúdo pode ser inapropriado. Por exemplo, considere o grafo bipartido de artigos e seus autores: como usar o conteúdo destes objetos na medida de similaridades, já que são de tipos distintos? Aliás, o que seria o conteúdo de um autor? Uma alternativa é usar a informação topológica do grafo. Não obstante, em [50], os autores afirmam que o método clássico de similaridade no espaço vetorial com ponderação tf-idf [60] (que pondera a importância de uma palavra dentro de um documento em uma coleção) é extremamente ruidosa e que não se adequa à *Web*. Apesar de ser um método útil para filtrar resultados com baixa similaridade léxica, medidas baseadas somente em dados textuais parecem não colaborar bem para ranquear os demais resultados, sendo na verdade fracamente correlacionados com a similaridade semântica. Não por acaso, os sistemas de buscas para *Web* até o final da década de 90 tinham eficácia ruim se comparados aos modernos que complementam os resultados de análises textuais com algoritmos de *Link Analysis*, como o *PageRank* [11, 57]. Ainda em [50], os autores são categóricos ao dizer que a implicação deve ser a revisitação do papel de similaridade por conteúdo na *Web*. Os dados bibliográficos podem ser organizados em um grafo de biblioteca digital e arestas entre entidades expressam relações semânticas bem definidas, como: citação e autoria. Sendo assim, os algoritmos de *Link Analysis* estariam portanto trabalhando com este tipo de informação semântica.

Um laboratório para Web: Assim como os dados da WEB, a informação bibliográfica pode ser modelada por um grafo e ambas redes possuem conceitos análogos. Ambas são redes de informação e que o objeto principal é um documento (página ou artigo). Ambas têm interligações entre documentos expressando uma citação, uma fonte e possuem uma autoria. Ainda, são publicadas em sites ou anais/revistas. Evidentemente há diferenças entre as duas redes, uma é mais estática, no momento de publicação de um artigo, dificilmente seus dados se alteram no futuro. O mesmo não se pode dizer de páginas web. Contudo, poderíamos considerar os dados de bibliografia acadêmica como um laboratório para avaliar algoritmos de *Link Analysis* para a Web, dada a riqueza da analogia entre ambas redes de informação. Além disso, esta coleção representaria possivelmente um ambiente menos ruidoso, dado o controle do que é publicado e a estruturação mais padronizada em termos de organização.

1.2.2 Aplicações

Resgatando o contexto de aplicação em Recuperação de Informação, uma aplicação de grande sucesso acadêmico e econômico de técnicas de *Link Analysis* é o de "ranqueamento" de objetos, como o *PageRank* e *HITS* [39] que continuam influentes apesar de motivarem por mais de uma década novas técnicas de *Link Analysis*. Alguns outros exemplos de

aplicações são [28]:

- Classificação de objetos - neste problema de aprendizado de máquina supervisionado, objetos devem ser rotulados dentre um conjunto de categorias. O inter-relacionamento entre objetos é explorado, no que se chama de "classificação coletiva", ou seja, um conjunto de objetos correlacionados são conjuntamente rotulados [14,61];

- Agrupamento (*Clustering*) de objetos - neste problema de aprendizado de máquina não supervisionado, objetos são agrupados em grupos cujos objetos compartilham características. Em *Link Analysis* é considerado o que se chama de "*functional clustering*" [55], em que o objetivo não é estatisticamente agrupar indivíduos, mas descobrir membros de uma comunidade dinâmica com relativa similaridade funcional na rede [30];

- Resolução de entidades - é o problema de determinar a entidade no mundo real que se refere uma representação desta entidade, com aplicações em integração e "de-duplicação" de registros [4];

- *Link Prediction* - é o problema de prever a existência de uma relação entre duas entidades, baseando-se nos atributos e relações com outras entidades em uma rede. O problema de *Link Prediction* possui grande impacto comercial quando considerado em redes de preferências [30, 54], já que são o objetivo de sistemas de recomendação de propagandas, livros, filmes e amizades;

- *Subgraph discovery* - área de mineração de dados que tem por objetivo detectar padrões de subgrafos em uma rede. Combinada com o a área de classificação de grafos, cujo objetivo é rotular todo o grafo (como um único objeto) dentre um conjunto de categorias, estes padrões de subgrafos podem ser classificados e produzir interessantes aplicações, como por exemplo, análise de marketing viral, investigação em redes de terrorismo, epidemiologia, relações comerciais [30].

Antonellis, Garcia-Molina e Chang [5] consideram o problema de reescrita de busca para resultados patrocinados. Isto é, de acordo com os termos da busca que um usuário faz na *Web*, o método proposto reescreve os termos de forma a casarem melhor com um banco de dados de propagandas, com objetivo de retornar propagandas mais relevantes. Para resolver o problema, consideram o grafo bipartido entre termos de busca e propagandas. Este grafo é obtido de *logs* reais do Yahoo! de cliques de usuários em propagandas após efetuarem uma busca usando termos durante duas semanas. Consideram um subconjunto do grafo, que originalmente possuía 15 milhões de buscas distintas, 14 milhões de propagandas e 28 milhões de arestas. Estendem o modelo do *SimRank* [5] para considerar pesos nas arestas, isto é, a contagem de cliques entre termos e propaganda. Avaliam os

resultados com um corpo editorial de especialistas na área do Yahoo! e medem a precisão e a revocação de seu método. Os resultados apontam que o pior ponto da curva Precisão x Revocação é quando a precisão está próxima de 0,9 e revocação de 1,0. Isso significa que todas propagandas relevantes foram retornados ao usuário e que em torno de 10% apenas eram irrelevantes. Ainda, apontam que em 98% dos casos a busca era reescrita pelo método, isto é, que ela realmente influência nos resultados.

Liben-Nowell e Kleinberg [45] aplicam medidas de similaridade de *Link Analysis* para o problema de previsão de arestas em grafos. No caso, tentam prever se no futuro dois pesquisadores se tornarão coautores de algum artigo apenas observando a estrutura do grafo no passado. Usam diferentes coleções de artigos obtidos do *arXiv*[1] e experimentam diversas medidas de relacionamento entre coautores. Realizam experimentos usando o grafo de colaboração, isto é, os nós são autores e há uma aresta entre dois autores se, e somente se, eles publicaram juntos. Conduzem experimentos também no grafo original, bipartido, com arestas entre artigos e autores. Os resultados, apesar de acurácia baixa, mostram que ao usar medidas de similaridade para prever uma coautoria no futuro, isto é, na hipótese de que quanto mais similar são dois autores mais provável é uma coautoria ocorrer no futuro, as previsões são consideravelmente melhores que uma escolha aleatória, ou seja, comprova-se que a observação apenas da estrutura do grafo tem informações úteis para a previsão de coautoria.

White e Smyth [71] aplicam suas medidas de similaridade para encontrar as principais relações no grafo de interações entre terroristas no ataque de 11 de setembro de 2001. O grafo possui 63 nós, sendo 19 sequestradores e demais associados e 308 arestas. Em destaque aos resultados, a medida "*Markov Centrality*" responde que Mohammed Atta possui maior importância na rede e sabe-se que foi o líder dos sequestradores. Aplicam também suas técnicas para encontrar pesquisadores que estão fortemente relacionados, para tanto usam o grafo de coautoria do *Citeseer*, com 387.703 artigos entre 1991 e 2002. Percebem que várias medidas de similaridade distintas concordam nas relações mais fortes e que estão de acordo com suas expectativas.

Narayanan e Shmatikov [52] aplicam análises apenas sobre a topologia de grafos para efetuar a "de-anonimização" de agentes em redes sociais, isto é, o objetivo é re-identificar usuários anônimos. Conseguem com sucesso identificar usuários da rede social *Flickr* que também estão na rede do *Twitter*. O resultado é promissor, dado que a intersecção entre as redes é menor que 15%, ou seja, seria um desafio obter precisão nos resultados. O método toma como entrada dois grafos de relacionamento de redes sociais e uma semente de mapeamentos parciais entre usuários entre estas redes. A partir daí, o algoritmo encontra outros mapeamentos de usuários nas duas redes sociais.

[1] http://arxiv.org/ – Acessado em Julho de 2012.

Ramage, Rafferty e Manning [58] aplicam o conceito de passeios aleatórios, "*Random Walks*", no grafo léxico de termos do *WordNet*. Com isso, conseguem extrair medidas de relacionamento semântico entre palavras. No problema de reconhecimento de paráfrases o método supera métodos baseados no modelo espaço vetorial, reduzindo em 18,5% os erros. O problema de reconhecimento de paráfrases consiste em identificar quando duas frases possuem significados equivalentes.

Quanto a dados multimídia, *Link Analysis* foi aplicada no contexto de recuperação de imagem por conteúdo, sigla *CBIR* em inglês para *Content-Based Image Retrieval*. Jing e Baluja [37] criam o *VisualRank*, uma técnica que analisa o grafo de afinidade de características visuais entre imagens no estilo do *PageRank*, isto é, faz passeios aleatórios neste grafo para produzir uma importância relativa entre as imagens, esta importância é usada como critério de ordenação. Conduzem experimentos e mostram que tal técnica supera o sistema do *Google* de busca de imagens.

1.3 Abordagens

Nesta área de cálculo de medidas de relacionamento entre entidades, seja similaridade ou importância relativa, os métodos se dividem em três abordagens: a primeira diz respeito a abordagem que avalia apenas o conteúdo entre os objetos. Neste sentido, existem medidas de similaridade e modelos de recuperação de documentos por conteúdo, como o modelo espaço vetorial [6]. Incluem-se também as técnicas de recuperação de imagens por conteúdo quando observam-se as características visuais das imagens, via seus descritores. Uma segunda abordagem é a análise de um grafo de relacionamento entre objetos, o que se chama de *Link Analysis*. Neste sentido, o conteúdo dos objetos não é mais observado, mas sim a topologia da rede em que estão inseridos. O *PageRank* é um exemplo de abordagem *Link Analysis*, pois consegue avaliar a relevância global de páginas Web sem mesmo observar o conteúdo destas páginas. Finalmente, a terceira abordagem consiste em realizar uma análise combinada entre conteúdo dos objetos e o grafo de relacionamento entre eles. É um tratamento promissor para o cálculo de similaridade, pois tenta juntar o melhor dos dois mundos. Exemplos de estudos são [12, 13, 17, 20, 32, 50, 51]. Em destaque, Jing e Baluja [37] criam o *VisualRank*, um algoritmo de recuperação de imagem por conteúdo que combina a análise de características visuais de imagens (conteúdo) e análise da topologia do grafo de afinidade entre estas características ao longo de várias imagens, a partir de passeios aleatórios, assim como é feito no *PageRank*.

1.4 Descrição do Problema

Esta seção apresenta a descrição do problema abordado nesta dissertação, formalizando o problema e descrevendo o conjunto de dados que será abordado.

1.4.1 Formalização

Considere um grafo direcionado $G(V, E)$, em que V é o conjunto de vértices deste grafo e E o conjunto de arestas. Seja $\mathrm{In}_{x \in V}$ o conjunto predecessor do vértice x, isto é, o conjunto de vértices de origem, ou entrada, cujas arestas incidentes terminam em x, assim $|\mathrm{In}_x|$ é o grau de entrada de x, o número de vértices que precedem x. Análogo, $\mathrm{Out}_{x \in V}$ é o conjunto de vértices sucessores do vértice x, isto é, o conjunto de vértices terminais, ou de saída, cujas arestas incidentes partem de x, assim $|\mathrm{Out}_x|$ é o grau de saída de x, o número de vértices que sucedem x. Seja $\mathrm{Adj}_{x \in V}$ o conjunto de vértices adjacentes a x, isto é, $\mathrm{Adj}_{x \in V} = \mathrm{In}_x \bigcup \mathrm{Out}_x$.

Seja uma rede de relacionamento entre objetos/entidades modelada como um grafo direcionado $G(V, E)$ cujos vértices representam objetos do mundo real e arestas (x, y), ou $x \to y$, estabelecem uma relação, com semântica pré-estabelecida, entre um par de objetos. Ainda, um número $w_{x,y} \in \mathbb{R}$ estabelece um peso para esta relação entre os objetos. Denota-se $s_{x,y} \in \mathbb{R}$ com $x, y \in V$ uma medida, derivada somente do grafo G, para a intensidade da similaridade entre x e y. Isto é, valores $s_{x,y}$ que estejam fortemente correlacionados com um julgamento de similaridade por um gabarito.

Ao se tratar de similaridade entre entidades é observado na literatura que $s_{x,y}$ é comumente obtido de diferentes maneiras [77], usualmente a partir de uma medida formal de distância em um espaço [44]. Aqui, contudo, não teremos necessariamente esta restrição, apenas estamos preocupados em encontrar funções para $s_{x,y}$ que estejam fortemente correlacionadas a um gabarito que julga a similaridade entre os objetos x e y levando-se em conta somente as informações de $G(V, E)$.

Assim, o nosso problema se resume em encontrar uma matriz $\mathbf{S}^G_{|V(G)| \times |V(G)|}$ cujos elementos $s_{x,y} \in \mathbb{R}$ são a pontuação/intensidade para a similaridade entre dois objetos $x, y \in V(G)$ e dependem exclusivamente da estrutura do grafo G. Em nosso modelo, quanto maior $s_{x,y}$, maior a similaridade entre x e y. Para manter uma escala única entre diferentes funções avaliadas, sem perda de generalidade, restringiremos $0 \leq s_{x,y} \leq 1$.

1.4.2 Conjunto de dados: O grafo da biblioteca digital de artigos

O domínio de objetos que analisaremos é o que envolve a bibliografia científica. Exemplos de tipos de objetos são: autores, artigos, veículos de publicação, eventos, editoras, patrocinadores, etc. Estes objetos estão relacionados, através de arestas do grafo da biblioteca

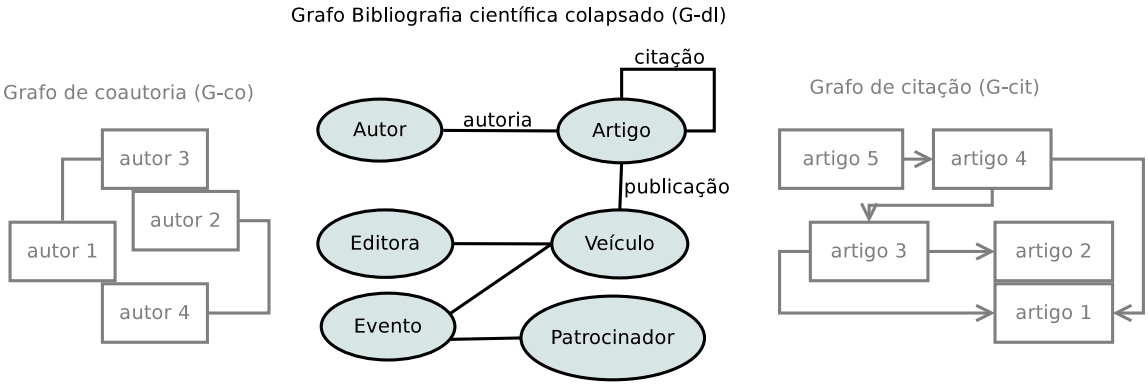


Figura 1.1: Grafo colapsado por tipos de entidades bibliográficas e grafos gerados.

digital, por semânticas bem definidas:

- Autoria – um autor escreve um artigo, formando uma aresta, não necessariamente direcionada, entre o autor e o artigo;

- Publicação – um artigo é publicado em um veículo de publicação (anais, livro, etc), de uma editora, e pode estar vinculado a um evento (conferências, *workshops*, simpósios) e eventualmente ser patrocinado. Isto inclui diferentes possibilidades de arestas: uma entre o artigo e o veículo de publicação e evento, outra entre o evento e o patrocinador e outra entre o veículo e a editora;

- Citação – um artigo cita outro em suas referências, formando uma arestas direcionada entre os artigos.

Considerando estas semânticas, podemos construir um grafo, G_{dl}, o grafo de uma biblioteca digital, que relaciona estes objetos. A Figura 1.1 mostra um grafo colapsado com as possíveis relações/arestas entre os objetos mencionados.

Nesta dissertação, estudaremos diferentes formas para calcular a matriz $\mathbf{S}^{G}_{|V(G)|\times|V(G)|}$, ou uma submatriz, que envolva apenas pares de objetos que exista um gabarito anotado para uma avaliação da correlação com a similaridade. Dado um grafo, G_{dl}, ainda, é possível gerar outros grafos que sintetizam distintas informações: podemos gerar um grafo de coautores G_{co}, o subgrafo induzido de citações G_{cit}, etc. Isto é, derivando distintos grafos podemos considerar diferentes meios para calcular a matriz de similaridade $\mathbf{S}^{\cdot}$, submatriz de $\mathbf{S}^{G_{dl}}$, em que $\cdot$ representa um grafo gerado.

1.5 Objetivos

- **Estudar técnicas de *Link Analysis* em aplicações de similaridades entre entidades:** Neste estudo, pretende-se entender como funcionam estas técnicas, seus desafios de computacionais, de implementação e eficácias. Pretende-se estudar estas técnicas existentes a fim de sabermos seus detalhes e ganhar intuições para compreender suas deficiências e vantagens. Pretende-se compreender as diferenças principais entre os métodos e em que se baseiam.

- **Construir um conjunto de dados:** Queremos construir um conjunto de dados estruturado para experimentação das técnicas e que seja grande o suficiente para ter validade nos resultados. Além de seu volume, estamos preocupados com sua qualidade. Isto se desdobra em obter dados que tenham uma origem de confiança e que sejam bem estruturados/organizados de acordo com um modelo de dados a se propor.

- **Desenvolver um protocolo experimental das técnicas estudadas:** Queremos construir experimentos objetivos e reproduzíveis que nos permitam avaliar objetivamente a qualidade e efetividade dos métodos, sendo possível, inclusive, ser reusados para futuras pesquisas.

- **Executar experimentos para aplicações das técnicas:** Pretende-se entender as possibilidades de aplicações das técnicas de *Link Analysis* em situações reais, bem como examinar sua eficácia em experimentos realizados em cenários próximos de uma aplicação real.

- **Lidar com a escala e volume de dados:** Como um dos objetivos é lidar com um bom volume de quantidade de dados, queremos também conseguir aplicar as técnicas nesta escala de dados. Para tanto, vamos estudar possibilidades em como aplicá-las, mesmo que sua complexidade computacional seja grande.

- **Propor melhorias às técnicas existentes:** Na medida em que se estudam os métodos e algoritmos existentes, observando suas diferenças e examinando em experimentos a eficácia, pretende-se ganhar intuição suficiente para entender o que poderia fazer dos métodos existentes melhores.

- **Apresentar propostas de trabalhos futuros de pesquisa:** Na medida em que se explora a área, chegaremos a outros tópicos de pesquisa que são igualmente desafiadores e promissores, todavia, fora do escopo deste trabalho.

Capítulo 2

Medidas de similaridade usando *Link Analysis*

Este capítulo apresenta as principais medidas de similaridade encontradas na literatura baseadas em *Link Analysis*.

2.1 *SimRank*

Jeh e Widom [36] exploram a propriedade de similaridade entre objetos organizados em uma rede. Para isso, expressam que dois objetos são similares se eles estão relacionados com objetos similares, ou seja, as similaridades se reforçam (do inglês "*reinforcement*"). Definem, assim, a seguinte função recursiva:

$$S_{x,y} = \begin{cases} 1 & \text{se } x = y \\ \frac{C}{|\text{In}_x||\text{In}_y|} \sum_{u,v \in \text{In}_x \times \text{In}_y} S_{u,v} & \text{se } x \neq y \end{cases}$$

O método tem uma interpretação em termos de passeios aleatórios: $s_{x,y} = E\left[c^l\right]$, em que l é a variável aleatória que diz em quantos passos dois passeios aleatórios, um partindo de x e outro de y, se encontram pela primeira vez. Os autores definem um método iterativo de cálculo que converge e cuja complexidade é $O\left(kn^2d^2\right)$, em que k é o número de iterações, n é o número de vértices no grafo e d é a esperança do grau de entrada de um vértice. No pior caso, o algoritmo é $O\left(kn^4\right)$. Quanto a espaço, o algoritmo é $O\left(n^2\right)$. Em seus experimentos, são pragmáticos e efetuam podas (sem garantias de aproximação) para executar o método em um conjunto de dados na ordem de $n = 3 \times 10^5$ documentos.

Lizorkin e Velikhov [49] melhoram o algoritmo, no pior caso, para $O\left(kn^3\right)$ e dão garantias para a quantidade de iterações necessárias. Nos experimentos, usam memória externa e executam o método em um grafo que representa os artigos do Wikipédia, com $n = 2,2 \times 10^6$ documentos.

Fogaras e Rácz [23] criaram um algoritmo probabilístico para cálculo do *SimRank* baseado no método de Monte Carlo e executam seu método em um grafo real de páginas Web, o *Stanford WebBase*, com $n = 7,9 \times 10^7$ documentos.

Li et al. [43] fazem uma abordagem completamente diferente: a equação do *SimRank* é rescrita em termos do produto de *Kronecker*, de forma que o seu cálculo não é iterativo. Além disso, fazem redução do posto da matriz de adjacência com *SVD* (*Singular value decomposition*). Nesse esquema, derivam algoritmos aproximados e conseguem responder buscas por similaridade *online* e realizar atualização incremental, isto é, adicionar novos nós no grafo sem a necessidade de recalcular o método para todo grafo. Ainda, dada esta atualização incremental, são os primeiros a abordar o *SimRank* em análise de redes dinâmicas, isto é, conseguem responder a similaridade entre objetos durante o tempo.

SimRank é um método bem influente na comunidade, haja vista os diversos métodos derivados ou semelhantes: **[47, 48, 69, 72, 73, 75, 76]**. Fica uma crítica ao *SimRank*, também observada por **[24]**, no caso em que dois nós compartilham todos seus vizinhos em suas arestas de entrada. Nesta situação o funcionamento do método é contra intuitivo, pois se dois objetos são considerados similares se eles estão relacionados com objetos similares, espera-se que a similaridade entre estes dois nós tenda a 1 na medida em que haja mais vizinhos comuns de entrada. Verifica-se, por outro lado, a partir da equação, que a similaridade tende a zero.

2.2 *Katz*

Katz [38] propõe uma técnica baseada na quantidade de caminhos entre dois vértices. O autor explora a ideia de que uma informação passa de um indivíduo x para outro indivíduo y de diversas formas, isto é, dependendo da quantidade de caminhos entre eles em um grafo de amizade. Uma aresta tem um valor associado de *efetividade* $\beta_{(x,y)}$ significando quão efetivamente ela serve como um canal de comunicação entre dois indivíduos. Com ressalvas, *Katz* assume que essa efetividade de comunicação é uma variável aleatória i.i.d, isto é, independente e identicamente distribuída. Assim, β^l é a probabilidade de que uma informação chegue após passar por l arestas, ou canais. Assim, nesse modelo, a probabilidade de que uma informação parta de x e chegue até y é $\sum_{l=1}^{\infty} \beta^l \left|\pi_{x,y}^{\langle l \rangle}\right|$, em que $\pi_{x,y}^{\langle l \rangle}$ é o conjunto de todos os caminhos de comprimento l de x até y. Ainda, *Katz* mostra que, dada uma matriz de adjacência M do grafo, pode-se calcular toda a distribuição do espaço amostral, algebricamente, por $\mathbf{p} = (I - \beta M)^{-1} - I$. Liben-Nowell e Kleinberg [45] aplicam diretamente esta técnica no contexto de *Link Prediction*. Ainda, é interessante a observação da semelhança entre este modelo e o *kernel* de Von Neumann [27,33].

Assim, define-se a similaridade como

$$S_{x,y} = \sum_{l=1}^{\infty} \beta^l \left|\pi_{x,y}^{\langle l \rangle}\right|$$

Uma alternativa de implementação à $\mathbf{S} = (I - \beta M)^{-1} - I$, que é $O(n^3)$ usando eliminação Gauss-Jordan, é um algoritmo iterativo, implementado no contexto desta dissertação:

Sabe-se que $\left|\pi_{x,y}^{\langle l \rangle}\right| = M^l$, em que M é a matriz de adjacência. Isto é, $S_{x,y}^{\langle l \rangle} = \sum_{l=1}^{\infty} \beta^l M^l$. Esta equação pode ser reescrita como:

$$\begin{aligned} S_{x,y}^{\langle l \rangle} &= \beta^1 M^1 + \beta^2 M^2 + \cdots + \beta^l M^l \\ &= \beta \left(I + \beta M + \cdots + \beta^{l-1} M^{l-1}\right) M \\ &= \beta \left(I + S_{x,y}^{\langle l-1 \rangle}\right) M \end{aligned}$$

Em que $S_{x,y}^{\langle 0 \rangle} = 0$.

Nesta versão iterativa, podemos considerar $l = 1 \ldots |\pi_{max}|$, em que π_{max} é o maior caminho entre dois vértices no grafo.

Isto é, no pior caso, este algoritmo é $O(|\pi_{max}|\, n^3)$, considerando que a multiplicação de matrizes AM é $O(n^3)$, para uma matriz $A_{n,n}$ qualquer.

Contudo, M é i) uma matriz cujos valores são zero ou um e ii) possivelmente esparsa. Estas características permitem a adoção de estratégias de otimização: Ao calcular $AM = \sum_k A_i \cdot M_j \ \forall i, j$, podemos considerar apenas os valores 1 de M, isto é, $AM = \sum_{k,j \in \text{ones}(M)} A_i \ \forall i$. Em que $\text{ones}\,(M)$ são apenas os índices cujos valores são 1 em M, em outras palavras, o conjunto de arestas do grafo que a matriz de adjacência M representa. Assim, nestas condições, AM é calculado com $O\,(nm)$ somas (sem multiplicações), em que m é o número de arestas do grafo. Portanto, o algoritmo *Katz* pode ser calculado em $O\,(|\pi_{max}|\, nm)$.

Por exemplo, em nossa aplicação, o grafo de citação tem 122.774 vértices e 523.665 arestas, bastante esparso e notavelmente $m \ll n^2$.

2.3 *Adamic/Adar*

Adamic e Adar [1] definem uma medida de similaridade semelhante ao coeficiente de *Jaccard*. Consideram que dois indivíduos estão mais fortemente relacionados na medida em que possuem itens em comum:

$$s_{x,y} = \sum_{\text{itemcomum}_{x,y}} \frac{1}{\log\left[\text{frequencia}\left(\text{itemcomum}_{x,y}\right)\right]}$$

Em que itemcomum$_{x,y}$ representa o conjunto de itens que os dois indivíduos x e y têm em comum. A técnica foi proposta com objetivo de predizer relacionamentos entre indivíduos usando informações encontradas na *Web*. Desta forma, experimentam a medida de similaridade em um conjunto de dados extraídos de páginas pessoais de cientistas na Web. Assim, consideram que um item comum é uma entidade mencionada em ambas páginas Web pessoais dos indivíduos, por exemplo: um *hyperlink*, um e-mail, etc. Já Liben-Nowell e Kleinberg [45] adaptam-na como:

$$s_{x,y} = \sum_{z \in \mathrm{Adj}_x \bigcap \mathrm{Adj}_y} \frac{1}{\log\left(|\mathrm{Adj}_z|\right)}$$

Isto é, o item em comum é um vértice que esteja na vizinhança de x e y. A frequência de um item é o número de vezes que este aparece na adjacência de algum vértice, isto é, seu grau. Note que a função não é bem definida para $|\mathrm{Adj}_z| = 1$, por exemplo nos casos de uma componente conexa C_1 (ciclo) ou $K_{m,1}$ (bipartido completo).

2.4 *Weighted Paths*

Muito similar à medida *Katz*, em [71] é definida uma técnica chamada "*Weighted Paths*", que se baseia na intuição de que: i. dois nós estão relacionados de acordo com os caminho que os ligam e ii. quanto maior é um caminho, menor é a importância conferida por este caminho na relação. Disto, define-se:

$$s_{x,y} = \sum_{\pi \in \mathcal{P}(x,y)} \beta^{-|\pi|}$$

em que $\mathcal{P}(x,y)$ é um conjunto de caminhos entre x e y. β é um parâmetro da medida de similaridade, tal que $1 \leq \beta < \infty$. Neste trabalho apresentam-se três critérios para construção do conjunto $\mathcal{P}(x,y)$. No primeiro, são considerados os menores caminhos. Apesar de ser tentador pensar que quanto menor o comprimento do caminho entre dois objetos mais relacionados estariam, ocorre um fenômeno que derruba esta intuição: o efeito "*Small World*" [54]. Isto é, a menor distância entre dois nós, também chamada de distância geodésica, é pequena, mesmo para nós não relacionados. Disto surge observações como "Seis níveis de separação", segundo a qual, a maior distância geodésica entre duas pessoas no mundo é menor ou igual a seis. Assim, não são esperados bons resultados para esta técnica. A título de curiosidade, Jean Piaget, reconhecido por fundar a Epistemologia Genética, possui o número de Erdös 3 [45], valor menor que muitos matemáticos e cientistas da computação que, supostamente, possuem trabalhos mais relacionados. Desta forma, os autores consideram também os "*K-short paths*", isto é, todos caminhos com comprimento maior ou igual a k. Por fim, ainda consideram os "*k-Short Node-Disjoint paths*" cujos

caminhos não repetem arestas ou nós (com exceção da origem e destino). Por exemplo, com $k = 3$, um conjunto válido é $\mathcal{P} = \{(1 \to 2 \to 3), (1 \to 4 \to 3), (1 \to 5 \to 7 \to 3)\}$. Já o conjunto de caminhos de vértices $\mathcal{P} = \{(1 \to 2 \to 3), (1 \to \mathbf{4} \to 3), (1 \to 7 \to \mathbf{4} \to 3)\}$ é inválido, já que o vértice 4 se repete nos dois caminhos.

2.5 *Common Neighbors*

O *Common Neighbors* [45] é uma medida de similaridade que mede quantos vizinhos comuns dois vértices possuem. Em [53] essa medida é computada no contexto de redes de colaboração e é verificada a correlação entre

$$s_{x,y} = \left|\mathrm{Adj}_x \bigcap \mathrm{Adj}_y\right|$$

e a probabilidade de que x e y irão colaborar no futuro.

2.6 *Unseen Bigrams*

Unseen bigrams explora a ideia de inferir a similaridade entre pares não observados $s_{x,y}$ a partir de valores conhecidos $s'_{x,z}$ quando x é similar a z e $s'_{x,\bullet}$ é uma outra medida de similaridade qualquer. De acordo com $s'_{x,\bullet}$, encontra-se o conjunto dos k-ésimos nós mais similares a x, denominado por $kNN^{\langle k \rangle}_{s'_{x,\cdot}}$. Assim, definem-se duas medidas de similaridade, uma versão ponderada (e outra não) a partir da similaridade de $s'_{x,z} \forall z \in kNN^{\langle k \rangle}_{s'_{x,\cdot}}$ [45]:

Na versão não ponderada, a similaridade é o número de vértices em comum entre o conjunto de vizinhos de y e os k-ésimos nós mais similares a x:

$$\overline{S_{x,y}} \;=\; \left|\left\{\forall z \in \mathrm{Adj}_y \bigcap kNN^{\langle k \rangle}_{s'_{x,\bullet}}\right\}\right|$$

Na versão ponderada, a similaridade é a soma da medida s' para os pares dos vértices vizinhos de y e aqueles mais similares a x (k-ésimos nós mais similares relativo a medida s'):

$$S_{x,y} = \sum_{\forall z \in \mathrm{Adj}_y \bigcap kNN^{\langle k \rangle}_{s'_{x,\bullet}}} s'_{x,z}$$

Quando $s'_{x,z} = 1 \forall z \in kNN^{\langle k \rangle}_{s'_{x,\cdot}}$, então $\overline{s_{x,y}} = s_{x,y}$.

2.7 *Hitting Time*

Hitting time, $H_{x,y}$, mede a esperança de passos requeridos em um passeio aleatório, do inglês *Random Walk*, para se chegar ao vértice y, na primeira vez, a partir de x. Pode ser definido como $H_{x,y} = \sum_{l=1}^{\infty} l \cdot \Pr\left(x \underset{l}{\rightarrow} y\right)$, em que l é a variável aleatória que denota o número de passos e $\Pr\left(x \underset{l}{\rightarrow} y\right)$ denota a probabilidade de que um passeio aleatório de l passos partindo de x chegue pela primeira vez a y. Esse modelo pode ser interpretado como uma cadeia de *Markov*. Como $H_{x,y}$ é uma medida de distância, em [45] é considerado o valor negativo de $H_{x,y}$:

$$s_{x,y} = -\sum_{l=1}^{\infty} l \cdot \Pr\left(x \underset{l}{\rightarrow} y\right)$$

Já White e Smyth [71] chamam de "*Markov centrality*" o recíproco de $H_{x,y}$:

$$s_{x,y} = \frac{1}{\sum_{l=1}^{\infty} l \cdot \Pr\left(x \underset{l}{\rightarrow} y\right)}$$

O nome "*Markov centrality*" é dado pois se fôssemos derivar uma importância global de x, somando $s_{x,y}$ para todos y, os nós localizados mais ao centro de massa do grafo tenderiam a ter uma importância maior.

2.8 *Preferential Attachment*

Essa medida segue da ideia do padrão "*The rich get richer*" de grafos aleatórios [54]. Isto é, a chance de uma nova aresta surgir ao acaso ser incidente ao vértice x é proporcional ao seu grau de adjacência, $|\mathrm{Adj}_x|$. Supondo isto, é tentador considerar que a chance de ao acaso uma aresta surgir incidente a x e y é proporcional a $|\mathrm{Adj}_x|\,|\mathrm{Adj}_y|$. Newman [53] e Barabási [53] observam, a partir de experimentos, esta correlação. Disto, Liben-Nowell e Kleinberg [45] definem:

$$s_{x,y} = |\mathrm{Adj}_x|\,|\mathrm{Adj}_y|$$

2.9 *Commute Time*

Dado que o *Hitting time*, $H_{x,y}$, não é simétrico, *Commute Time*, $C_{x,y} = H_{x,y} + H_{y,x}$. Assim, $s_{x,y} = -C_{x,y}$ e $s_{x,y} = \frac{1}{C_{x,y}}$ são duas possíveis medidas.

2.10 *Low-rank approximation*

Como um grafo pode ser representado por uma matriz de adjacência, não é surpresa que muitos métodos de *Link Analysis* possuem uma formulação matricial, a exemplo do estudo [43]. Uma técnica comumente aplicada em análises de matrizes grandes é a redução de seu posto criando uma aproximação da matriz original (que ainda faz redução de ruídos). Liben-Nowell e Kleinberg [45] aplicam redução de posto para o cálculo das medidas *Common Neighbors*, *Katz,* dentre outras, para predição de coautoria em artigos científicos.

2.11 *Rooted PageRank*

Liben-Nowell et al. [45] propuseram o *Rooted PageRank,* uma medida que calcula a probabilidade estacionária de se chegar a um vértice v em um passeio aleatório partindo de um vértice u. Neste passeio aleatório há duas ações: Com probabilidade $1 - \beta$, de um vértice vai para um outro vizinho (o vizinho é escolhido aleatoriamente de acordo com uma distribuição de probabilidades que deriva da matriz de adjacência); Com probabilidade β, é realizado um reinício (*restart*), isto é, de um vértice retorna para u, aquele que o passeio aleatório iniciou. Uma fórmula fechada para calcular a matriz de similaridade de acordo com a medida *Rooted PageRank* é:

$$S = (1 - \beta)\,(I - \beta T)^{-1}$$

Em que, $T = D^{-1}A$, $D_{ii} = \sum_j A_{ij}$ e A representa a matriz de adjacência.

2.12 *PropFlow*

Em [46], Lichtenwalter et al. propuseram um algoritmo denominado *PropFlow* que se baseia em um passeio aleatório (*Random Walk*) entre dois vértices de um grafo. O *PropFlow* corresponde a probabilidade de que um passeio aleatório inicie no vértice u e termine no vértice v em até l passos, usando os pesos das arestas como probabilidades de transição. De acordo com os autores, o *PropFlow* é uma medida similar ao *Rooted PageRank*, contudo, com o efeito de propagação mais localizado. A implementação do algoritmo é baseada em uma modificação de uma busca em largura com profundidade limitada em l. Lichtenwalter et al. apresentam esta medida como um método não supervisionado de predição de arestas, isto é, quanto maior a medida *PropFlow* entre dois vértices, mais provável será a criação de uma nova aresta entre estes vértices no futuro. Nesta atividade,

avaliam a técnica em um conjunto de dados, denominado "phone", que é um grafo direcionado com 712 milhões de arestas. Estas arestas relacionam indivíduos por chamadas de telefones celulares, ou seja, há uma aresta de u para v se, e somente se, u realizou uma chamada para v. Há pesos nas arestas associados a quantas ligações um indivíduo realizou para outro. Para este conjunto de dados, a tarefa é prever para quem uma pessoa irá chamar no futuro.

2.13 *Jaccard Coefficient*

Jaccard's coefficient [6] mede a similaridade de dois conjuntos. Sejam A e B dois conjuntos, então o coeficiente *Jaccard* é dado por $s = \frac{|A \bigcap B|}{|A \bigcup B|}$. No trabalho de Liben-Nowell e Kleinberg [45] consideram-se o conjuntos de vizinhos de dois vértices e, então, definem:

$$s_{x,y} = \frac{\left|\mathrm{Adj}_x \bigcap \mathrm{Adj}_y\right|}{\left|\mathrm{Adj}_x \bigcup \mathrm{Adj}_y\right|}$$

Fogaras e Racz [24] criam um algoritmo aproximado que, no momento da busca, isto é, online, estimam esta medida de similaridade com apoio de um índice.

Capítulo 3

Modelagem e amostragem de dados

3.1 Modelo e estrutura dos dados

Para conduzir os experimentos, consideramos o grafo gerado do subconjunto dos artigos científicos na área de ciência da computação e eletrônica publicados pela *ACM* (*Association for Computer Machinery*) em sua biblioteca digital[1] [21] e indexados na biblioteca digital da *Odysci*[2] [8]. A empresa *Odysci* desenvolve um sistema de busca acadêmico e para isto indexa bases de artigos da *ACM*, *IEEE*, *DBLP*, dentre outros. Deste subconjunto de artigos, extraímos da base da *Odysci* um conjunto de metadados suficientes para construção do grafo que representa uma biblioteca digital. Primeiramente reorganizamos os dados em estruturas de dados mais simples e compactas, usando o *Google Protocol Buffers*[3]. Com esta tecnologia é possível serializar dados que podem ser lidos/processados em diferentes linguagens de programação. Assim, serializamos todos dados considerados pelo esquema em um único arquivo de tamanho compacto e de rápido acesso. A seguir, apresentamos o esquema dos dados extraídos da base *Odysci*, na linguagem de descrição de dados do *Protocol Buffers*[4], ilustrado também pelo diagrama da Figura 3.1:

[1] http://dl.acm.org/– Acessado em Julho de 2012.
[2] http://www.odysci.com – Acessado em Julho de 2012.
[3] *https://developers.google.com/protocol-buffers/docs/overview* – Acessado em Julho de 2012.
[4] https://developers.google.com/protocol-buffers/docs/proto– Acessado em Julho de 2012.

```
package master;
message Dataset {
   repeated DatasetRecord record = 1;
   message DatasetRecord {
      optional ArticleData article = 1; //metadados do artigo
      optional Venue venue = 2; //veículo de publicação do artigo (coleção de artigos)
      repeated string classes = 3; //classificação ACM 1998 do artigo
      repeated ArticleData citation = 4; //conjunto de artigos citados

      message ArticleData {
         optional string doi = 1; //digital object identifier, DOI
         optional string id = 2; //identificador Odysci do artigo
         optional string title = 3; //título do artigo
         optional string url = 4; //url do artigo na biblioteca digital ACM
         repeated AuthorData author = 5; //autores do artigo
      }

      message AuthorData {
         optional string name = 1; //nome do autor
         optional string id = 2; //identificador Odysci do autor
      }

      message Venue {
         optional int32 year = 1;//ano do veículo de publicação
         optional string id = 2;//identificador do veículo de publicação
         optional string name = 3;//nome do veículo de publicação
         optional Serie serie = 4;//serie de veículos de publicação
         message Serie {
            enum SerieType{
               JOURNAL = 0;
               CONFERENCE = 1;
            }
            optional string id = 1;//identificador Odysci para a série
            optional string name = 2;//nome da série
            optional SerieType type = 3; //tipo da série: conferência ou revista
         }
      }
   }
}
```

Nosso conjunto de dados é representado por registros do tipo *DatasetRecord*. Cada registro possui metadados de um artigo científico, relação *article*, e dos artigos que este tem em suas referências bibliográficas, *citation*. Cada artigo, do tipo *ArticleData*, além de seus metadados, possui um conjunto de autores, relação *authors*, que diz quem escreveu o artigo. Cada registro ainda informa dados sobre onde o artigo foi publicado, relação *venue*. Aqui, a informação sobre o veículo de publicação é organizada em dois níveis: *Venue* e *Serie*. *Venue* é uma espécie de coleção de artigos científicos. Por exemplo, poderíamos dizer que seria o livro dos anais de uma conferência ou, ainda, um número de um volume de uma revista. Já *Serie* representa um agrupamento de *Venue*. Por exemplo, uma conferência pode ocorrer várias vezes em diferentes períodos de tempo. Cada ocorrência desta conferência pode produzir uma coleção de artigos publicados, a

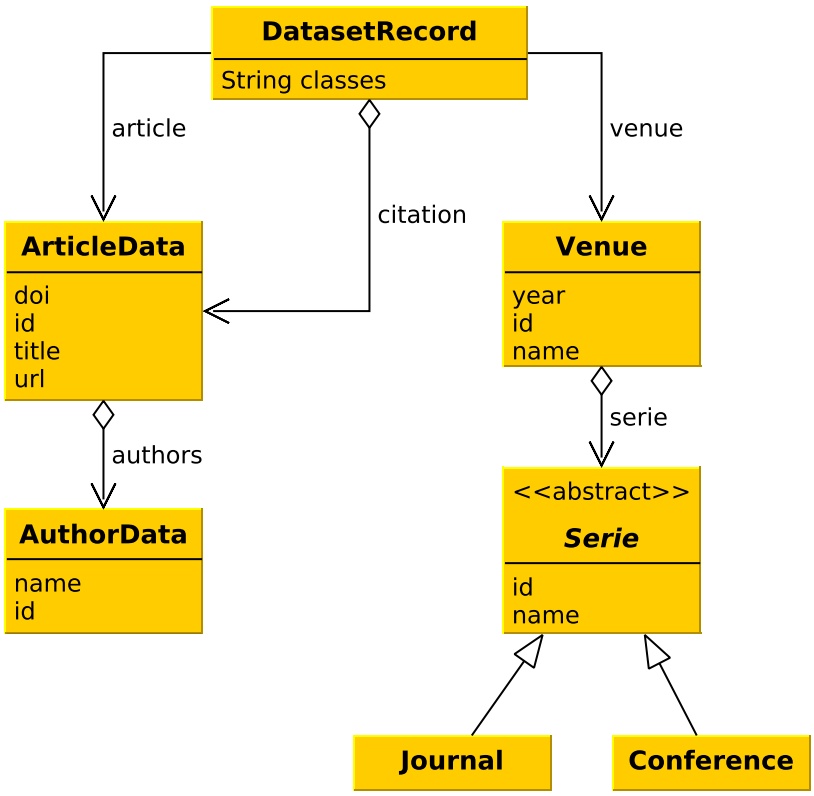


Figura 3.1: Diagrama *UML* para o esquema do conjunto de dados de uma biblioteca digital acadêmica.

Venue, mas o agrupamento de toda esta série de eventos, como uma entidade única, é a *Serie*. Ou ainda, uma revista acadêmica que é publicada mensalmente tem cada impressão representada por uma *Venue*. Já o agrupamento de todas impressões desta revista, desde seu primeiro número, é representada unicamente por uma *Serie*. Exemplificando, a revista *"Communications of the ACM"* é uma *Serie*, já uma impressão de Janeiro de 2012 é uma *Venue*. Cada artigo publicado é representado por *ArticleData* assim como suas referências bibliográficas. Este é um modelo de dados simplificado para representar dados de publicações científicas, outro modelo mais completo e expresso em termos de ontologias pode ser encontrado em [67].

Deste conjunto de dados extraídos, criamos um outro conjunto de dados, este anonimizado, isto é, removemos nomes e alteramos os identificadores das entidades para que não fosse possível saber qual entidade do mundo real as informações se referiam. Usar dados anônimos foi condição para que a *Odysci* fornecesse os dados.

Deste conjunto anônimo de dados, criamos o grafo de biblioteca digital descrito pelo

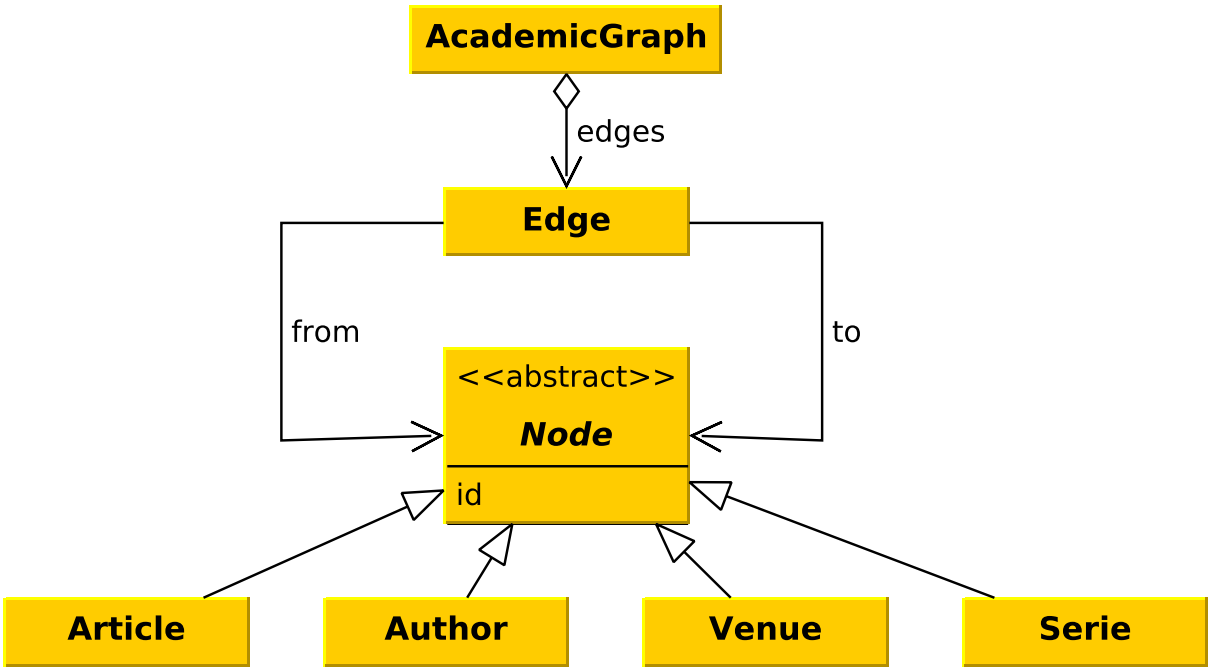


Figura 3.2: Diagrama *UML* para o esquema do grafo extraído de uma biblioteca digital acadêmica.

seguinte esquema e ilustrado pelo diagrama da Figura 3.2.

```
message AcademicGraph {
    repeated Edge edge = 1; //conjunto de arestas direcionadas
    message Edge {
        required Node from = 1; //vértice de partida
        required Node to = 2; //vértice de chegada
    }
    message Node {
        required Type type = 1; //tipo do vértice
        required int32 id = 2; //identificador do vértice
    }
}
enum Type {
    Article=0;
    Author=1;
    Venue=2;
    Serie=3;
}
```

O grafo da biblioteca digital acadêmica, representada pelo tipo *AcademicGraph*, é um conjunto de arestas direcionadas, relação *edges*. Cada aresta, do tipo *Edge*, possui dois vértices: o de partida, *from*, e o de chegada, *to*. Cada vértice, do tipo *Node*, pode

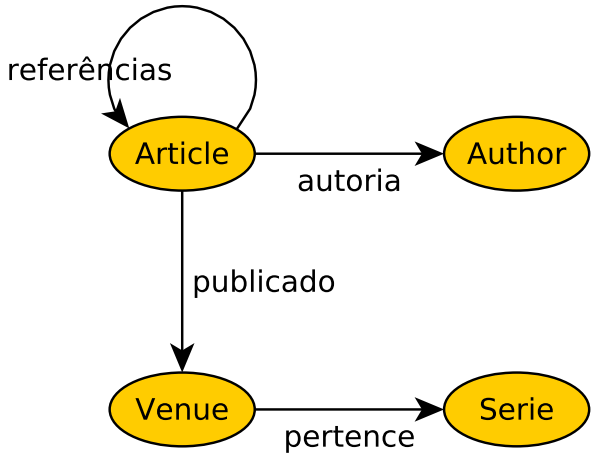


Figura 3.3: Grafo colapsado por tipo de entidades de uma biblioteca digital acadêmica.

Entidade (Vértices)	Quantidade	Relações (Arestas)	Quantidade
Artigos	122.774	Citação/Referências	523.699
Autores	123.822	Autoria	330.652
Series	7.757	Publicação	125.560
Total	**254.353**	**Total**	**979.911**

Tabela 3.1: Quantidade de entidades e relações no conjunto de dados.

representar um artigo, *Article*; um autor, *Author*, uma publicação, *Venue*; ou uma série de publicações, *Serie*. Uma topologia típica para uma instância deste grafo, colapsando pelos tipos, é representado pela Figura 3.3.

Uma alternativa é considerar o grafo gerado que associa diretamente artigos com series, conforme Figura 3.4

O grafo extraído, conforme topologia da Figura 3.4 e Tabela 3.1, envolve 122.774 artigos, 123.822 autores, 330.652 arestas de autoria, 523.699 arestas de referências/citações, 7.757 séries de veículos de publicação, 125.560 arestas de "publicado"/publicação (nada impede que um artigo seja publicado em mais de uma Serie). Em um total, são 254.353 vértices e 979.911 arestas.

Para nossos experimentos, que têm por objetivo calcular a similaridade entre artigos somente, mesmo porque nosso gabarito apenas diz respeito a artigos, trabalhamos apenas com o grafo de citação, conforme esquema expresso pela Figura 3.5.

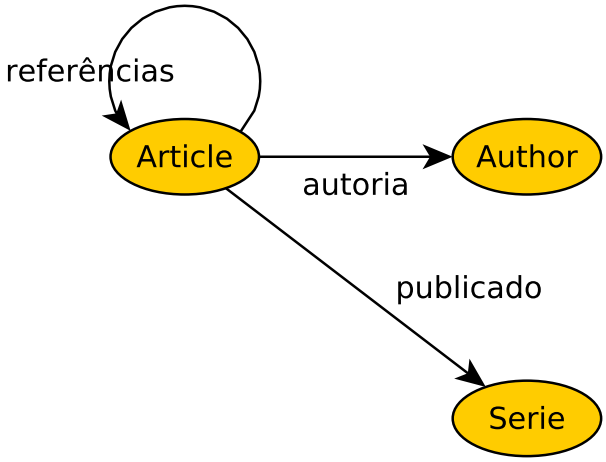


Figura 3.4: Grafo colapsado por tipo de entidades de uma biblioteca digital acadêmica omitindo series de publicação.

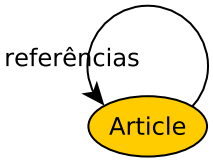


Figura 3.5: Grafo colapsado de citação acadêmica.

3.2 Amostragem do grafo de citação

A razão de usar uma amostra é que se tornou computacionalmente inviável manter, em memória, uma matriz que pode ocupar $O\left(|V|^2\right)$ de espaço e executar algoritmos de complexidade $O\left(|V|^3\right)$ e $O\left(|V|^4\right)$. Pode-se contra-argumentar em usar memória secundária para auxiliar a tarefa e processar integralmente os grafos, contudo, apesar de resolvermos a questão quanto espaço, ainda sofremos com o tempo de execução.

Diferentes algoritmos de similaridades produzem matrizes de diferentes níveis de esparsidade. Por exemplo, dependendo da topologia do grafo, o cálculo das medidas *Jaccard* e *Adamic/Adar* produzem matrizes bem esparsas, isto é, com muitos zeros. Matrizes esparsas, dependendo em como se estruturam, podem ocupar muito menos espaço em memória e, ainda, ter na prática alguns algoritmos executados mais rapidamente, apesar da análise teórica.

Isto é, seria possível para alguns algoritmos, usar o grafo de biblioteca digital integralmente. Por outro lado, alguns algoritmos como *SimRank*, em algumas situações, produzem matrizes de similaridade muito densas. Além disso, considerando sua alta complexidade teórica, $O\left(|V|^4\right)$ para o método de cálculo proposto originalmente em [36], ou mesmo $O\left(|V|^3\right)$ como no método otimizado em [49], tornou-se inviável considerar todo o grafo da biblioteca. Apenas para ilustrar, sob a ótica de espaço, considerando uma matriz densa de similaridade para um grafo com ≈100 mil vértices, sem nenhuma otimização, com precisão simples, pelo menos 37 gigabytes de memória principal seriam necessários aproximadamente.

3.3 Métodos de amostragem

Esta seção apresenta métodos de amostragem de grafos e sua utilidade de acordo com trabalhos anteriores da literatura relativa.

3.3.1 Amostragem *ForestFire*

Leskovec, Kleinberg e Foloustos [41] abordam a questão "Como grafos reais crescem com o tempo?" para estabelecer uma forma como grafos crescem/evoluem temporalmente. Ou seja, buscam um modelo de geração de grafos que mantém similar algumas propriedades consideradas ao se comparar com um grafo real. Para tanto, avaliam grafos reais de citação e afiliação em publicações científicas do *arXiv*[5] e de citações em patentes americanas. Em termos de propriedades, consideram duas: a densidade (número de vértices e arestas) e a distância mínima média entre dois vértices. Isto é, como estas duas propriedades

[5] http://arxiv.org/ – Acessado em Julho de 2012.

variam na medida em que o grafo se altera no tempo. Neste cenário, argumentam que os modelos de geração de grafos, até então na literatura, não eram capazes de captar as mesmas características destas duas propriedades quando comparadas com as encontradas em grafos reais. Estabelecer propriedades esperadas para alguns tipos de grafos, bem como conhecer como um grafo evolui tem diversas aplicações:

- Geração de grafos: para extrapolar o funcionamento de algoritmos para grafos sintéticos maiores que os reais disponíveis em diferentes cenários hipotéticos;
- Simulação: gerando grafos sintéticos quando se é impossível coletar grafos reais ou quando é muito dispendioso obter um, por exemplo, o grafo formado pelos links das páginas de toda Internet;
- Extrapolação: quando se quer saber mais sobre como seria um presente grafo no futuro;
- Na detecção de anomalias: ao avaliar se um subgrafo induzido possui as mesmas propriedades normais e esperadas, podendo ajudar a detectar fraudes, spam e ataques *DDoS* (Distributed Denial-of-service attack);
- Amostragem: há casos em que temos grafos reais disponíveis, mas, grandes o suficiente para tornar a avaliação de alguns algoritmos impraticáveis. Contudo, como saber se a amostra contém as mesmas propriedades do grafo original?

Alguns modelos podem ser preferíveis em alguns cenários específicos, dado que diferentes tipos de grafos reais têm um comportamento distinto na forma como evoluem. Por exemplo, o grafo de citação entre publicações acadêmicas tem propriedades bem características: há poucos ciclos direcionados e uma aresta, de citação, é permanente. Características bem distintas do grafo de amizade em uma rede social. Assim, pode-se esperar que estes grafos evoluam de forma distinta com o tempo e que um modelo pode servir em um caso mas não em outro.

Naquele mesmo trabalho de Leskovec et al. [41], é sugerido um novo modelo chamado *ForestFire*. A base deste modelo é definir como um novo vértice que chega com o tempo se anexa a um grafo já existente. Essencialmente, de um novo vértice v que chega, cria-se uma nova aresta para um vértice já existente w. Com isto, a partir da vizinhança de w, v passa a conhecer outros vértices para os quais ocasionalmente criam-se novas arestas de saída. Em [41], encontra-se uma boa intuição do processo com um exemplo: um autor de um artigo identifica referências para colocar na bibliografia. Primeiro encontra um artigo para citar e depois, a partir das suas referências, considera um subconjunto aleatório delas e continua a descobrir novos artigos para citar recursivamente.

Considere um vértice v se juntando à rede no instante de tempo $t > 1$, e seja G_t o grafo obtido até então, note que G_1 consiste de um grafo com um único vértice. Do vértice v formam-se arestas de saída para vértices em G_t de acordo com o seguinte processo:

1. v escolhe um vértice "embaixador" w aleatoriamente, uniformemente, e forma uma aresta para w.

2. Dois números aleatórios x, y são gerados, geometricamente distribuídos com esperanças $(1-p)^{-1}$e $(1-rp)^{-1}$ respectivamente. O vértice v escolhe x arestas de saída e y arestas de entrada ainda não visitadas, aleatoriamente. Sejam $w_1, w_2, \ldots, w_{x+y}$ os vértices terminais destas arestas selecionadas.

3. v forma arestas de saída para $w_1, w_2, \ldots, w_x$, e então aplica o passo (2) recursivamente para cada $w_1, w_2, \ldots, w_x$. Vértices não são visitados duas vezes, prevenindo ciclos.

Há dois parâmetros usados neste processo: o *forward burning probability* p e o *backward burning ratio* r. O primeiro, p, especifica a probabilidade de a partir de um vértice, seguir (o termo usado em inglês é *burn*) arestas de saída. Já o segundo parâmetro, r, é um fator da probabilidade p para seguir arestas de entrada a partir de um vértice.

3.3.2 Amostragem *RankedForestFire*

No algoritmo *ForestFire* de amostragem é usada a analogia de um pesquisador navegando através das citações e referências bibliográficas de artigos acadêmicos "embaixadores". Na medida em que um novo artigo (vértice) é descoberto nesta navegação, a partir de uma citação ou referência (aresta), este par (vértice, aresta) é adicionado ao grafo de amostra e este processo continua até que o grafo construído tenha o número de vértices ou arestas desejado, conforme descrito em detalhes na Seção 3.3.1. Dado um vértice de partida, este pesquisador escolhe aleatoriamente x arestas de saída e y arestas de entrada.

Se um pesquisador hipotético estivesse navegando em uma biblioteca digital talvez a escolha das referências ou citações a explorar não seria aleatória e, supostamente, com o viés de como estas arestas são apresentadas. Por exemplo, sistemas reais como *Odysci*[6] e *Microsoft Academic Search*[7] apresentam, na página de metadados de cada artigo, uma porção ranqueada destas arestas, isto é, uma lista dos artigos mais relevantes que citam e outra daqueles citados. A heurística destes sistemas é que o pesquisador não vai verificar todos artigos apresentados, mas possivelmente aqueles mais importantes. Isto é, se

[6] http://www.odysci.com/ – Acessado em Julho de 2012.
[7] http://academic.research.microsoft.com/ – Acessado em Julho de 2012.

este pesquisador seguisse uma porção destes artigos apresentados sua escolha não seria aleatória, mas sim dependente do ranqueamento de artigos destes sistemas.

Sendo assim, aqui é proposto um novo modelo de navegação *ForestFire* do pesquisador: o pesquisador enviesado pelo ranqueamento. Esta é uma ideia análoga a um trabalho derivado do algoritmo de ranqueamento de busca *PageRank*. Richardson e Domingos [59] estenderam o modelo de Page e Brin [11,57] para criar um viés em seu "surfista inteligente", isto é, este não escolhe aleatoriamente uma próxima página para se visitar, mas sim, escolhe uma de acordo com seu interesse definido *a priori* pelos seus termos de busca. A analogia é que o pesquisador enviesado também não escolhe referências ou citações aleatoriamente, mas sim, escolhe de acordo com o ranqueamento definido *a priori* por uma biblioteca digital. Este pesquisador tem uma "visão curta", ou seja, ele não conhece todas arestas que partem de um artigo, mas apenas aquelas que são apresentadas na ordem definida pelo sistema de uma biblioteca digital.

Portanto, o algoritmo *RankedForestFire* proposto é a alteração apenas do passo (2) do original *ForestFire*:

2. Dois números aleatórios x, y são gerados, geometricamente distribuídos com esperanças $(1-p)^{-1}$e $(1-rp)^{-1}$ respectivamente. O vértice v escolhe x arestas melhores ranqueadas de saída e y de entrada, ainda não visitadas, de acordo com a visão que se tem a partir do vértice v. Sejam $w_1, w_2, \ldots, w_{x+y}$ os vértices terminais destas arestas selecionadas.

Um parâmetro adicional ao modelo ForestFire é o esquema de ranqueamento de vértices, isto é, aqueles que são listados primeiramente para este pesquisador com "visão curta". Adiante, conduziremos experimentos para avaliar a eficácia deste algoritmo, *RankedForestFire*, usando como esquema de ranqueamento o algoritmo *PageRank*, calculado no grafo de citação "população" [8].

Acreditamos que este método de amostragem seja mais adequado para dados em que exista uma relevância distinta entre vértices do grafo e que isto cria um viés para o que seja mais "visível". Nesta noção, argumentamos a favor desse método na amostragem de grafos que representam bibliotecas digitais e grafos da Web. Aqui, avaliaremos este método para um grafo de uma biblioteca digital apenas, assim, fica como trabalho futuro verificar sua eficácia para o grafo da Web e, mais geral, para redes de informação.

[8] Observe que o algoritmo *PageRank* implementado pelo método iterativo *Power Iteration* é bastante eficiente. Já os algoritmos de similaridade de vértices são $O\left(|V|^3\right)$ ou $O\left(|V|^4\right)$, em que $|V|$ é a quantidade de vértices em um grafo. Em outras palavras, não é incoerente calcular o *PageRank* no grafo original (população) para obter amostras, já que o gargalo são os algoritmos de similaridade. Mesmo em amostras bem menores estes algoritmos exigem muito mais tempo de processamento que o *PageRank* no grafo completo.

3.3.3 Amostragem por vértices

Subgrafo induzido de vértices aleatoriamente escolhidos: Neste método de amostragem simples e direto, escolhe-se aleatoriamente um subconjunto de vértices do grafo original de forma uniformemente distribuída. Dado os vértices escolhidos, é induzido o subgrafo amostra. Isto é, seja $G(V, E)$ o grafo original e $G^s(V^s, E^s)$ o grafo amostra que se pretende obter. Dado um parâmetro $r \in \mathcal{R}$, obtém-se $V^s \subseteq V$ tal que $|V^s| \sim r\,|V|$, $\Pr(v \in V^s) = \frac{1}{|V|}$, a aresta $e \in E^s$ se, e somente se, $e \in E$ e seus vértices terminais $u, v \in V^S$.

Subgrafo induzido de vértices no tempo: Esta é uma amostragem por filtro: dados os parâmetros $y_o \leq y_f$ que representam um intervalo de anos, é construído um subconjunto de vértices $v \in V^S \subseteq V$ tal que v foi "criado" (ou conhecido) em um instante de tempo t, em que $y_o \leq t \leq y_f$. Dado V^s, é construído o subgrafo induzido $G^s(V^s \subseteq V, E^s \subseteq E)$. Esta amostragem temporal é uma forma ideal de se obter um subgrafo amostral se naquele intervalo de tempo escolhido é verdadeira a hipótese de que o grafo já possuía as mesmas propriedades da versão atual. Uma dificuldade deste método é o controle do tamanho do grafo da amostra, já que uma pequena variação do intervalo de anos pode alterar bastante a quantidade de vértices da amostra. Neste caso, uma estratégia possível seria escolher aleatoriamente um subconjunto dos vértices daquele intervalo de tempo, similar ao método da Seção 3.3.3.

3.3.4 Amostragem por arestas

Seja $G(V, E)$ o grafo original e $G^s(V^s, E^s)$ o grafo amostra que se pretende obter. Dado um parâmetro $r \in \mathbb{R}$, obtém-se $E^s \subseteq E$, tal que $|V^s| \sim r\,|V|$, $\Pr(e \in E^s) = \frac{1}{|E|}$ e $u, v \in V^s$ se, e somente se, $e\,(u, v) \in E^s$.

3.4 Avaliação dos métodos de amostragem

Em [40, 42], são apresentados diferentes métodos de amostragem de grafos e formas de avaliação para saber quão representativa é a amostra em relação ao grafo original. Considerando [42], a amostragem é o problema de encontrar um subgrafo, com menos vértices, que seja similar ao grafo original. Uma amostra é similar se mantém várias das propriedades originais do grafo. A amostragem é avaliada considerando dois objetivos, a redução de escala e o retrocesso temporal. A redução de escala consiste em simplesmente encontrar um subgrafo com menos vértices similares ao original. Já o retrocesso temporal consiste em encontrar um subgrafo similar ao original em um dado tempo, ou seja, considerando

que um grafo evolui com o tempo, como é o caso do grafo de uma biblioteca digital, o retrocesso seria derivar uma versão "antiga" do grafo sem a informação da idade de cada vértice.

Para cada um destes objetivos é definido um critério de similaridade entre amostra e o grafo dado. No caso da redução de escala, é extraída a distribuição de probabilidade de diversas propriedades dos grafos, por exemplo a distribuição do grau de entrada e saída dos vértices; distribuição do tamanho das componentes fortemente conexas; distribuição de propriedades espectrais; etc. Já para o retrocesso temporal o critério considerado é extrair em cada instante de tempo uma propriedade (um número único) do grafo amostrado. Estes valores de propriedades calculadas em cada instante de tempo formam um conjunto e então deriva-se uma distribuição de probabilidade. Várias propriedades são consideradas como por exemplo: o maior autovalor da matriz de adjacência; o tamanho (normalizado) da maior componente conexa; etc.

A ideia então é comparar as distribuições de probabilidades empíricas dos valores de diferentes propriedades obtidas a partir da amostra com as obtidas do grafo original. Para comparar estas distribuições de probabilidade é usado um método de *Goodness of Fit*, a estatística D do teste *Kolmogorov-Smirnov* [62]. Trata-se de um teste que consegue avaliar se duas amostras são obtidas de uma mesma distribuição de probabilidade ou se uma amostra foi obtida de uma distribuição de referência. Neste caso, é feito o teste para verificar se os valores da propriedade obtidos da amostragem do grafo seguem uma distribuição de probabilidade similar à distribuição dos valores da mesma propriedade calculada no grafo original. Se sim, isto é um indicador de que o grafo amostrado manteve a mesma característica comparando-se ao grafo original, isto é, são similares de acordo com a propriedade avaliada. Ao realizar o mesmo teste para diferentes propriedades, estaremos avaliando o quanto uma amostra conservou distintas características do grafo original.

A fim de concluir as melhores formas de se amostrar um grafo, os autores de [42] apresentam a metodologia mencionada e avaliam para diferentes grafos reais, dentre eles o grafo de citação acadêmica obtida da biblioteca digital do *arXiv*[9]. Isto, inclusive, reforça a relevância de suas conclusões para este trabalho, dada a avaliação em um grafo de citação. Diversos algoritmos de amostragem são avaliados, incluindo os descritos aqui. Concluem que o melhor algoritmo de amostragem é o *ForestFire* e que amostras com 15% dos vértices mantêm as características do grafo original.

[9] http://arxiv.org/ – Acessado em Julho de 2012.

3.5 Resultados dos métodos existentes

Nesta dissertação, a justificativa de se amostrar o grafo de uma biblioteca digital, conforme apresentado no início do capítulo, é uma questão de eficiência: tanto de espaço quanto em tempo. Portanto, nosso objetivo é considerar uma boa amostra sobre a ótica de redução de escala, não em retrocesso temporal. Além das conclusões e parâmetros de algoritmos que se emprestam de [41,42], reproduzimos parte da metodologia de avaliação de [42] para confirmar que o algoritmo de *ForestFire* é o mais apropriado para o nosso conjunto de dados. Avaliamos assim quatro métodos de amostragem para o critério de qualidade na redução de escala do grafo. Consideramos desta forma, as distribuições de seis propriedades: i. grau de entrada e ii. saída dos vértices; iii. o coeficiente de *cluster* [70]; iv. a distribuição gerada pelo *PageRank* [11,57]; e os valores de v. "*authority*" e vi. "*hub*" do algoritmo *HITS* [39]. O coeficiente de *cluster* de um vértice v, C_v, mede o quanto cada vértice em um grafo tende se agrupar, verificando o quão próximo os vizinhos de v estão de ser um clique, isto é, calculando razão do número de arestas entre todos vizinhos de v pelo número máximo possível de arestas entre todos vértices desta vizinhança.

As propriedades iv, v e vi não foram consideradas em [40–42] e, portanto, avaliá-las trata-se de uma contribuição de pesquisa. O especial interesse pelo *PageRank* e *HITS* como critério de avaliação é que são algoritmos reconhecidamente usados em sistemas de busca para ranqueamento de documentos interligados. Assim, é um forte sinal de qualidade das amostras se estas preservarem o *PageRank* e *HITS*, principalmente para uma aplicação de busca e ranqueamento. Argumenta-se aqui que poderíamos estender para aplicações de similaridade. Além disso, as distribuições de grau de entrada/saída dos vértices e o coeficiente de clusterização derivam-se de uma contagem local do número de arestas incidentes a eles, por outro lado, o valores de *PageRank* e *HITS* sofrem influência da topologia do grafo como um todo, vide a natureza recursiva. Portanto, argumenta-se que é um resultado possivelmente mais forte preservar também o *PageRank* e *HITS* ("*authority*" e "*hub*") do que apenas preservar a distribuição de grau de entrada/saída e o coeficiente de clusterização em uma amostra de grafo.

3.5.1 Eficácia do *baseline* de métodos de amostragem

Os quatros métodos de amostragem avaliados são: *ForestFire* [41], amostragem por Arestas, por Vértice e Temporal, conforme descrito na Seção 3.3. Para cada método foram obtidas amostras de 15%, 30%, 60% e 80% do número de vértices do grafo original. Para manter a mesma escala no número de arestas, outras amostras foram obtidas dependendo da razão aresta/vértice produzida por cada método. No caso da amostragem Temporal (para a qual não se aplica o parâmetro de porcentagem da quantidade de vértices), fo-

ram considerados quatro intervalos de tempo da publicação dos um artigos na biblioteca digital: 2000-2002, 2000-2005, 2000-2007 e 2000-2010.

Os gráficos da Figura 3.6 mostram a estatística D do teste *Kolmogorov-Smirnov*, que mede a distância entre uma distribuição amostral e a distribuição empírica da população para diferentes propriedades: coeficiente de clusterização, distribuição de grau de saída e entrada dos vértices, *PageRank* e *HITS* ("*authority*" e "*hub*"). Foi impresso $1 - D$ pois isto dá uma noção de similaridade ao invés de distância. Estes gráficos mostram o valor de $1 - D$, similaridade, de cada método de amostragem para cada amostra gerada em distintos tamanhos de amostra (caracterizada pelo número de arestas).

De acordo com estes gráficos, com exceção do *HITS-authority*, a qualidade aumenta com o aumento do tamanho da amostra, o que é de se esperar de um método de amostragem. Com exceção do *PageRank*, o método de amostragem por arestas apresenta um desempenho pior que os demais métodos, razão pela qual tal método será descartado para os demais experimento deste trabalho. Para amostras suficientemente grandes, quando o número de arestas se aproxima do grafo original, nota-se que os algoritmos apresentam uma qualidade similar, isto é, a escolha de um método de amostragem torna-se praticamente indiferente. Contudo, para amostras menores, lembrando-se da necessidade de redução de escala, percebe-se uma grande diferença entre os métodos, justificando, portanto, a eleição de um algoritmo. Nota-se, ainda, que a distribuição de grau de entrada não é uma boa propriedade para distinguir a qualidade, pois os métodos obtêm resultados similares e estáveis rapidamente. Já para as demais propriedades, com exceção do coeficiente de clusterização, o método *ForestFire* é aquele que regularmente se apresenta entre os melhores para todos tamanhos de amostra. Outro método que apresentou boa eficácia foi o Temporal, com exceção para a propriedade PageRank. Escolhemos, portano, o método *ForestFire* como algoritmo de amostragem para o restante deste trabalho, já que não requer nenhuma informação do grafo senão sua própria topologia, ao contrário do Temporal que precisa da informação de quando os vértices/arestas surgiram. Apesar de termos este dado no grafo de citação de artigos acadêmicos, a não dependência dessa informação permite uma avaliação isenta do viés da janela de tempo escolhido, assim possivelmente, obtendo resultados mais gerais.

3.5.2 Tamanho das amostras

A seguir iremos verificar o crescimento do número de vértices que cada método de amostragem produz variando o tamanho da amostra (quantidade de arestas). A Figura 3.7 mostra o número de vértices contra o número de arestas para cada amostra produzida pelos algoritmos. Tomando como referência a curva do algoritmo de amostragem temporal, nota-se a divergência do algoritmo de amostragem por arestas, isto é, suas amostras pro-

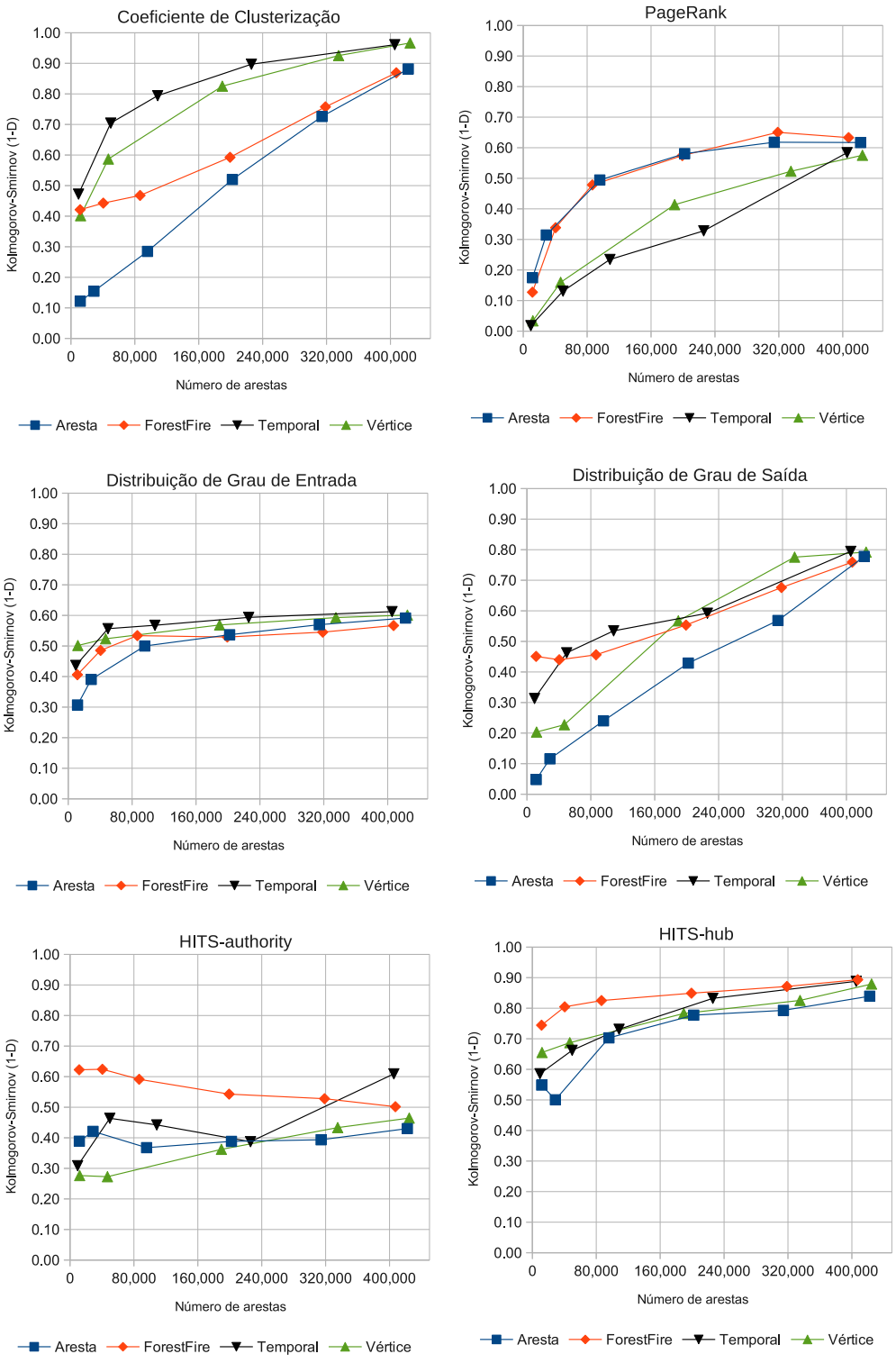


Figura 3.6: Teste Kolmogorov-Smirnov (1-D) de ajuste de distribuição na avaliação de qualidade dos algoritmos de amostragem do grafo de citação de acordo com diferentes propriedades.

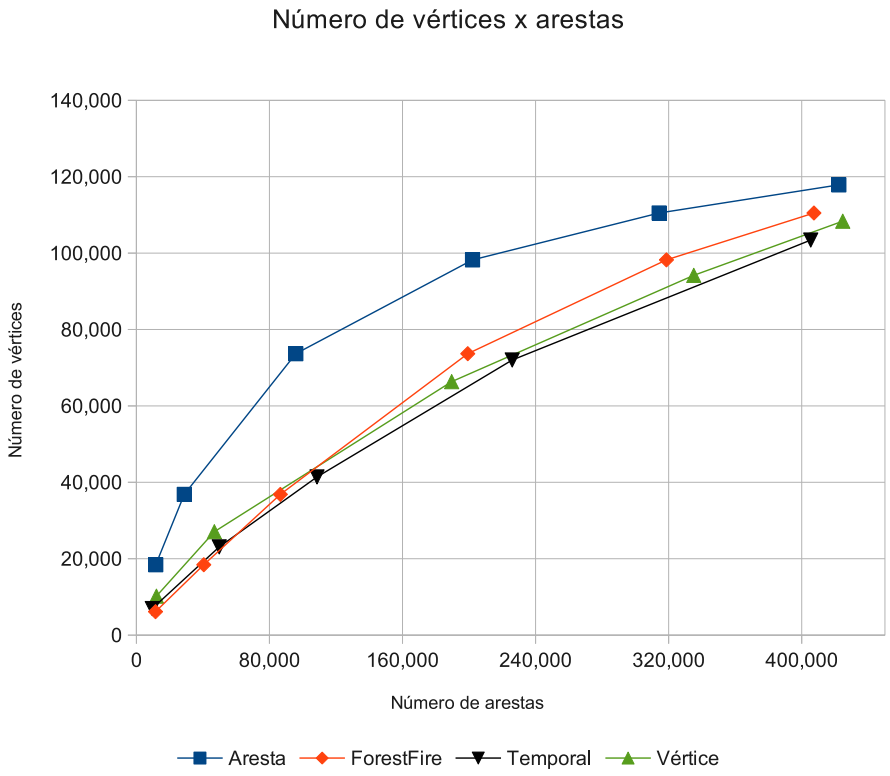


Figura 3.7: Número de vértices versus o número de arestas para amostras de cada algoritmo de amostragem.

duzem muito mais vértices para uma mesma quantidade de arestas, ou seja, grafos mais esparsos do que deveriam ser, o que mais uma vez apoia a rejeição deste algoritmo. Já as amostragem *ForestFire* e por Vértice preservam razoavelmente a razão número de arestas/vértices. A amostragem temporal é tomada como referência pois reflete perfeitamente como era o grafo em um dado instante de tempo.

3.6 Resultado do método proposto *RankedForestFire*

A mesma metodologia de avaliação descrita e executada nas seções anteriores é usada para comparar a qualidade do método proposto *RankedForestFire* contra o original *ForestFire*, que obteve melhor eficácia do *baseline*. Para tanto, considerou-se como um ranqueamento *a priori* dos artigos aquele gerado pelo algoritmo *PageRank* [11, 57]. Desta forma, um pesquisador enviesado navega de um artigo para um subconjunto de suas referências e citações com maior *PageRank* preferencialmente. Qualquer outro algoritmo de ranqueamento poderia ser considerado, até mesmo o derivado da simples contagem citações.

ForestFire versus *RankedForestFire*

Como o *ForestFire* obteve melhor eficácia dentre os métodos de amostragem aqui estudados e nosso algoritmo de amostragem proposto deriva deste método, justifica-se comparar o *RankedForestFire* contra o *ForestFire*.

A Figura 3.8 mostra diversos gráficos com o valor da estatística $1 - D$, *Kolmogorov-Smirnov*, que mede o quanto duas distribuições de probabilidade são similares. Esta estatística é calculada a partir de distribuíções de probabilidades obtidas de diversas propriedades das amostras do grafo de citação. Cada ponto destes gráficos reflete o quanto uma propriedade de uma amostra se mantém comparando-se ao grafo original. Quanto maior a estatística $1 - D$, melhor.

Analisando-se a Figura 3.8, o algoritmo *RankedForestFire* proposto apresenta um formato de curva bem similar ao *ForestFire*. Nota-se que o *RankedForestFire* supera o *ForestFire* para as propriedades clusterização, grau de saída e *HITS-authority* para tamanhos menores de amostras. Vale a pena notar que era de se esperar uma melhor eficácia no caso do *PageRank*, dado o viés introduzido pelo algoritmo de ranqueamento do *RankedForestFire*. Com exceção do *HITS-authority*, ambos métodos tendem a ter a mesma qualidade na medida em que o tamanho da amostra se aproxima do tamanho total do grafo.

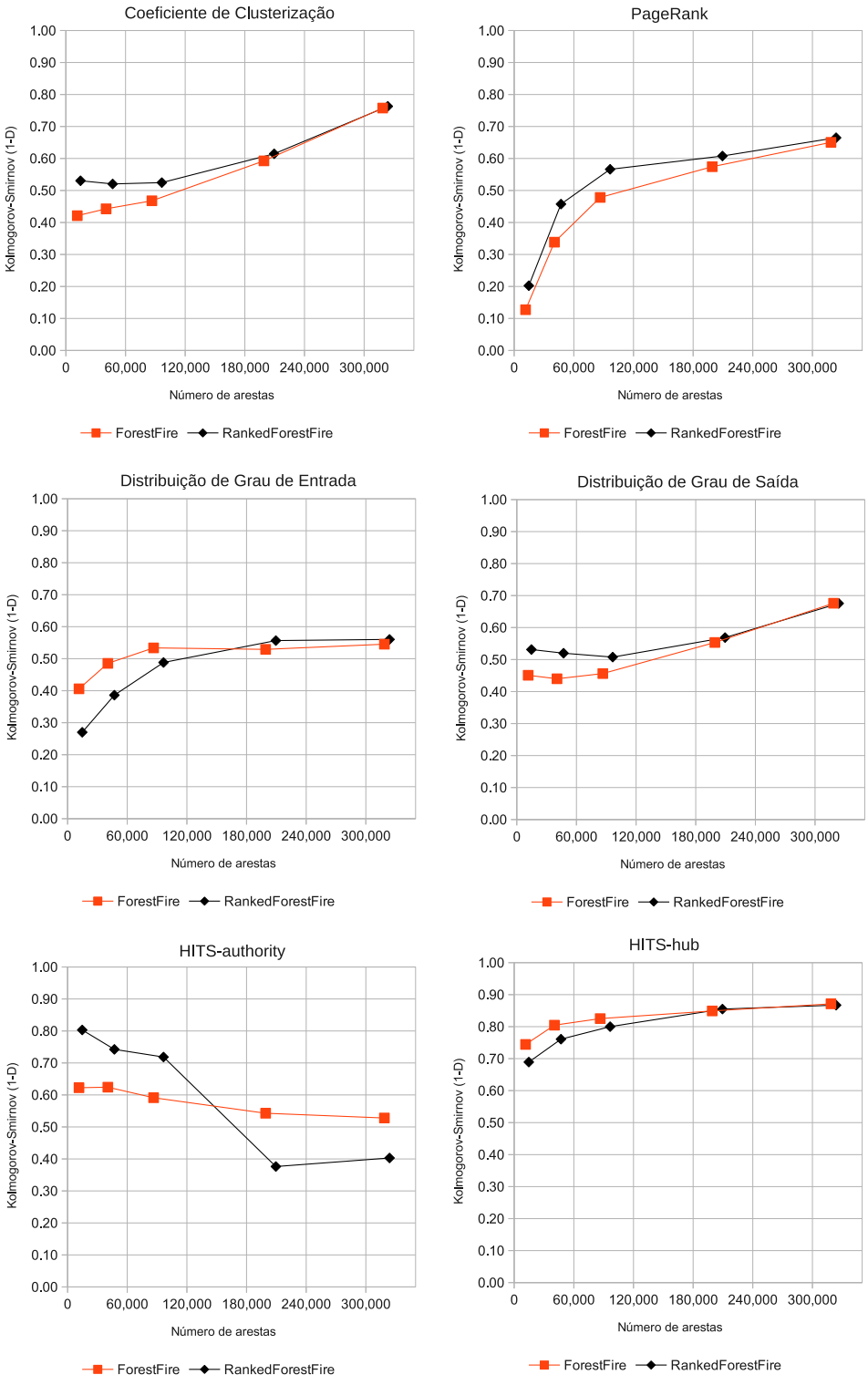


Figura 3.8: Teste Kolmogorov-Smirnov (1-D) de ajuste de distribuição na comparação de qualidade do *RankedForestFire* versus *ForestFire* para amostragem do grafo de citação.

Capítulo 4

Medidas recursivas propostas

4.1 Meta-função de similaridade

Esta dissertação de mestrado apresenta duas novas medida de similaridade denominadas *RecursiveJaccard* e *RecursiveAdamicAdar*. Tratam-se de meta algoritmos que "transformam" medidas de similaridade locais, no caso *Jaccard* e *Adamic/Adar*, em medidas recursivas, isto é, que dependem da similaridade pré-calculada da vizinhança de dois vértices. Ou seja, a similaridade entre dois vértices $x, y \in V(G)$ depende recursivamente da similaridade entre os vértices adjacentes de x e y, inspirado no algoritmo *SimRank* [36]. O argumento por trás é que a similaridade de dois vértices u, v depende recursivamente do quão similar são os vértices da vizinhança de u, v.

Na Seção 2.1, é descrito o *SimRank*, como se segue:

$$S_{x,y} = \begin{cases} 1 & \text{se } x = y \\ \frac{C}{|\text{In}_x||\text{In}_y|} \sum_{x' \in \text{In}_x} \sum_{y' \in \text{In}_y} S_{x',y'} & \text{se } x \neq y \end{cases}$$

Nesta formulação, a similaridade entre dois vértices depende, recursivamente, da similaridade entre todos pares de vértices no cartesiano quadrado dos conjuntos de vértices precedentes de x e y, respectivamente, In_x e In_y. Esta similaridade é um fator C da soma da similaridade de cada similaridade calculada entre os pares $(x', y') \in \text{In}_x \times \text{In}_y$. Reescrevendo *SimRank* com esta notação:

$$S_{x,y} = \begin{cases} 1 & \text{se } x = y \\ \frac{C}{|\text{In}_x||\text{In}_y|} \sum_{u,v \in \text{In}_x \times \text{In}_y} S_{u,v} & \text{se } x \neq y \end{cases}$$

Desta definição, é derivado um algoritmo iterativo para o cálculo:

$$
\begin{aligned}
s_{x,y}^{(0)} &= \begin{cases} 1 & \text{se } x = y \\ 0 & \text{se } x \neq y \end{cases} \\
s_{x,y}^{(k)} &= \frac{C}{|\mathrm{In}_x|\,|\mathrm{In}_y|} \sum_{u,v \in \mathrm{In}_x \times \mathrm{In}_y} s_{u,v}^{(k-1)}
\end{aligned}
$$

Em [36], é mostrado que $\lim_{k\to\infty} s_{x,y}^{(k)} = S_{x,y}$.

A ideia do meta algoritmo proposto é considerar a forma de cálculo do *SimRank*, mas com a possibilidade de "parametrizar" com uma medida de similaridade local qualquer, como se segue:

Algoritmo 1: ReccursiveMetaSimilarity

Entrada: o grafo G, diferença mínima de parada δ, número máximo iterações K e medida de similaridade local *LocalSim*

Saída : a matriz de similaridade S

1 $P \leftarrow \{(u,v) \in \mathrm{LocalSim.pairs}\,(G)\}$

2 $S_{x,y\in V(G)^2}^{(0)} \leftarrow \begin{cases} 0 & \iff x \neq y \\ \frac{1}{|V(G)|} & \iff x = y \end{cases}$

3 **for** $k \leftarrow 1, K$ **do**

4 $\quad S_{u,v}^{k} \leftarrow S_{u,v}^{k-1} + \mathrm{LocalSim.similarity}\left(S^{k-1}, u, v\right)\ \forall\,(u,v) \in P$

5 $\quad N \leftarrow \sum_{u,v\in P} S_{u,v}^{k}$

6 $\quad S_{u,v}^{k} \leftarrow \frac{S_{u,v}^{k}}{N}\ \forall (u,v) \in P$ `// Normaliza a matriz`

7 $\quad$ **if** $\max\left\{\Delta ulp\left(S_{u,v}^{(k)}, S_{u,v}^{(k-1)}\right)\ \forall\,(u,v) \in P\right\} \leq \delta$ **then**

$\quad\quad$ `// Se convergiu, pára`

8 $\quad\quad$ **stop**

9 $\quad$ **end**

10 **end**

Note que este algoritmo toma como parâmetro de entrada *LocalSim*, que se refere a um outro algoritmo de similaridade local. *LocalSim* precisa fornecer dois métodos para o meta algoritmo: $\mathrm{pairs}\,(G)$ e $\mathrm{similarity}\,(S,u,v)$. O primeiro método informa todos pares de vértices para os quais o valor da similaridade será diferente de zero (Linha 1). Trata-se apenas de um corte, isto é, uma otimização que evita que não seja calculada a similaridade para pares de vértices que já se sabe que será zero, de acordo com o método $\mathrm{similarity}\,(S,u,v)$. O segundo método é a própria função de similaridade local, que é transformada em uma função recursiva, isto é, toma como entrada os valores previamente

calculados de similaridade, a matriz S, e o par de vértice que se deseja saber o valor da similaridade, u, v (Linha 4). Nas subseções seguintes há detalhes em como este método foi implementado para as medidas de similaridade *Jaccard* e *Adamic/Adar*.

No algoritmo, há duas condições de parada: i. um limite no número de iterações K; ii. se a máxima diferença entre os elementos das duas últimas matrizes calculadas é menor que o parâmetro δ (Linha 7). Implementamos uma condição de detecção de convergência estrita, baseada na diferença de "*Unity of Least Place*" (*ulp*) [29], detalhado na Seção 5.4.4. Quando uma das duas condições é verdadeira, o algoritmo pára e retorna a última matriz calculada.

4.1.1 Método recursivo para a medida *Jaccard*

Na Seção 2.13, é apresentada a medida *Jaccard* como se segue:

$$s_{x,y} = \frac{\left|\mathrm{Adj}_x \bigcap \mathrm{Adj}_y\right|}{\left|\mathrm{Adj}_x \bigcup \mathrm{Adj}_y\right|}$$

Isto é, o coeficiente *Jaccard* de dois vértices x e y é o número de vértices adjacentes que x e y possuem em comum dividido pelo número de vértices na adjacência de x e y. Note que será zero se, e somente se, $\left|\mathrm{Adj}_x \bigcap \mathrm{Adj}_y\right| = 0$, ou seja, sempre que dois vértices não possuem algum outro vértice em comum em sua vizinhança. Assim, a implementação do método pairs (G) é simplesmente produzir todos pares de vértices que têm pelo menos um outro em comum em sua adjacência:

$$\mathrm{RecJaccard.pairs}\,(G) = \left\{(u,v) \in V\,(G) \times V\,(G)\,|\mathrm{Adj}_u \bigcap \mathrm{Adj}_v \neq \emptyset\right\}$$

Agora, o desafio é como transformar a medida *Jaccard* para considerar os valores de similaridade da vizinhança dos vértices. A ideia é calculá-la no produto cartesiano dos conjuntos formados pelos vértices vizinhos de u, v, contudo, ao invés de somar um (1) sempre que há um elemento nesta vizinhança, soma-se o valor pré-calculado de $M_{u,v}$. A medida *Jaccard* pode ser reescrita como:

$$\begin{aligned} S_{u,v} &= \frac{|\mathrm{Adj}_u \bigcap \mathrm{Adj}_v|}{|\mathrm{Adj}_u \bigcup \mathrm{Adj}_v|} \\ &= \frac{\sum\limits_{z \in (\mathrm{Adj}_u \bigcap \mathrm{Adj}_v)} 1}{\sum\limits_{z \in (\mathrm{Adj}_u \bigcup \mathrm{Adj}_v)} 1} \end{aligned} \tag{4.1}$$

Assim, redefinimos *Jaccard* para o meta algoritmo recursivo como:

$$\text{Jaccard.similarity}\left(M,u,v\right)=\frac{\sum\limits_{x,y\in\left(\text{Adj}_u\bigcap\text{Adj}_v\right)^2}M_{x,y}}{\sum\limits_{x,y\in\left(\text{Adj}_u\bigcup\text{Adj}_v\right)^2}M_{x,y}} \tag{4.2}$$

Note que esta fórmula é para o caso de vizinhança não direcionada, pois são considerados os vértices adjacentes, $\text{Adj}_\bullet$. Para o caso direcionado, usamos uma fórmula muito similar, mas que pondera entre a vizinhança dos predecessores e sucessores:

$$\begin{aligned}\text{Jaccard.similarity}\left(M,u,v\right) &= \lambda f\left(\text{In},u,v\right)+\left(1-\lambda\right)f\left(\text{Out},u,v\right)\\ f\left(\Upsilon,u,v\right) &= \frac{\sum\limits_{x,y\in\left(\Upsilon_u\bigcap\Upsilon_v\right)^2}M_{x,y}}{\sum\limits_{x,y\in\left(\Upsilon_u\bigcup\Upsilon_v\right)^2}M_{x,y}}\end{aligned}$$

Υ representa uma vizinhança de um vértice, ora os predecessores, ora os sucessores. O parâmetro λ define a importância relativa entre a vizinhança dos predecessores e sucessores.

Nesta configuração de vizinhança direcionada, o método que define os pares de vértices cujo valor será diferente de zero deve ser reescrita como:

$$\text{RecJaccard.pairs}\left(G\right)=\left\{\left(u,v\right)\in V\left(G\right)\times V\left(G\right)|\text{In}_u\bigcap\text{In}_v\neq\emptyset\text{ ou }\text{Out}_u\bigcap\text{Out}_v\neq\emptyset\right\}$$

4.1.2 Método recursivo para medida *Adamic/Adar*

Na Seção 2.3, é apresentado a medida *Adamic/Adar* como se segue:

$$s_{x,y}=\sum_{z\in\text{Adj}_x\bigcap\text{Adj}_y}\frac{1}{\log\left(\left|\text{Adj}_z\right|\right)}$$

Isto é, a similaridade *Adamic/Adar* de dois vértices x e y é a soma do inverso do logaritmo da frequência (grau) de cada vértice em comum na adjacência de x,y. Note que será zero se, e somente se, $\left|\text{Adj}_x\bigcap\text{Adj}_y\right|=0$, ou seja, sempre que dois vértices não possuem algum outro vértice em comum em sua vizinhança. Assim como ocorre com *Jaccard*, a implementação do método RecAdamicAdar.pairs (G) é simplesmente produzir todos pares de vértices que têm pelo menos um outro em comum em sua adjacência:

$$\text{RecAdamicAdar.pairs}\left(G\right) = \left\{\left(u, v\right) \in V\left(G\right) \times V\left(G\right) | \text{Adj}_u \bigcap \text{Adj}_v \neq \emptyset\right\}$$

Assim como transformamos a medida *Jaccard*, faremos com *Adamic/Adar*. A ideia é calcular no produto cartesiano da vizinhança dos vértices u, v, contudo, ao invés de somar um (1) para cada vértice na vizinhança daqueles em comum de u, v, somaremos o valor pré-calculado em M e reescalamos multiplicando por n, o número de vértices no grafo. Esta multiplicação leva o termo numa escala similar a de $|\text{Adj}_\bullet|$, isto é, proporcional ao número de vértices da vizinhança. Note que, pelo algoritmo iterativo, os valores da matriz de similaridade M são normalizados entre $[0, 1]$, por isso também, a multiplicação por n. Assim, o teto desta multiplicação tem função análoga a frequência da vizinhança.

A medida *Adamic/Adar* pode ser reescrita como:

$$\begin{aligned}
S_{u,v} &= \sum_{z \in \text{Adj}_x \bigcap \text{Adj}_y} \frac{1}{\log\left(\text{freq}\left(z\right)\right)} \\
&= \sum_{z \in \text{Adj}_x \bigcap \text{Adj}_y} \frac{1}{\log\left(|\text{Adj}_z|\right)} \\
&= \sum_{z \in \text{Adj}_x \bigcap \text{Adj}_y} \frac{1}{\log\left(\sum\limits_{w \in \text{Adj}_z} 1\right)}
\end{aligned}$$

Assim, redefinimos *Adamic/Adar* para o meta algoritmo recursivo como:

$$\begin{aligned}
\text{RecAdamicAdar.similarity}\left(M, u, v\right) &= \sum_{x,y \in \left(\text{Adj}_u \bigcap \text{Adj}_v\right)^2} \frac{1}{\log\left(\text{Freq}\left(x, y \mid M\right)\right)} \\
\text{Freq}\left(x, y \mid M\right) &= \lceil n \text{Pr}\left(x, y \mid M\right) \rceil \\
\text{Pr}\left(x, y \mid M\right) &= \sum_{z,w \in \text{Adj}_x \times \text{Adj}_y} M_{z,w}
\end{aligned} \tag{4.3}$$

Note que esta fórmula é para o caso de vizinhança não direcionada, pois são considerados os vértices adjacentes, $\text{Adj}_\bullet$. Para o caso direcionado, usamos uma fórmula muito similar, mas que pondera entre a vizinhança dos predecessores e sucessores:

$$
\begin{aligned}
\text{RecAdamicAdar.similarity}\,(M,u,v) &= \lambda f\,(\text{In},u,v) + (1-\lambda)\,f\,(\text{Out},u,v)\\
f\,(\Upsilon,u,v) &= \sum_{x,y\in(\Upsilon_u \bigcap \Upsilon_v)^2} \frac{1}{\log\left(\text{Freq}\,(x,y \mid M,\Upsilon)\right)}\\
\text{Freq}\,(x,y \mid M,\Upsilon) &= \lceil n\text{Pr}\,(x,y \mid M,\Upsilon)\rceil\\
\text{Pr}\,(x,y \mid M,\Upsilon) &= \sum_{z,w\in\Upsilon_x\times\Upsilon_y} M_{z,w}
\end{aligned}
$$

Em que Υ representa uma vizinhança de um vértice, ora os predecessores, ora os sucessores. O parâmetro λ define a importância relativa entre a vizinhança dos predecessores e sucessores.

Nesta configuração de vizinhança direcionada, o método que define os pares de vértices cujos valores de similaridade será diferente de zero deve ser reescrita como:

$$\text{RecAdamicAdar.pairs}\,(G) = \left\{(u,v) \in V\,(G) \times V\,(G)\,|\text{In}_u \bigcap \text{In}_v \neq \emptyset \text{ ou } \text{Out}_u \bigcap \text{Out}_v \neq \emptyset\right\}$$

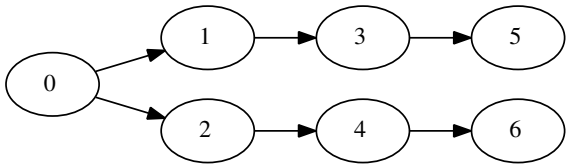
4.1.3 Redefinindo medidas de similaridade locais

Nesta seção, foram redefinidas duas medidas de similaridade locais, *Jaccard* e *Adamic/Adar*. Contudo, nada impede fazer o mesmo com outras medidas, como *Preferential Attachment*, *Common-Neighbors,* etc. A ideia que guia esta transformação é converter uma fórmula que normalmente depende de um vértice da vizinhança local de pares de vértices, para uma fórmula que depende de pares de vértices do produto cartesiano da vizinhança local de pares de outros vértices. Com este produto cartesiano, obtemos um conjunto de pares de vértices necessários para invocar a função recursiva de similaridade, isto é, o valor da matriz de similaridade da iteração anterior.

Por exemplo, na Equação 4.1, a medida *Jaccard* diz que a similaridade de u, v depende do vértice z, pertencente em uma vizinhança comum de u, v. De acordo com a Equação 4.2, a transformação desta medida faz com que a similaridade de u, v dependa agora da similaridade de outro par de vértices x, y, estes pertencentes no produto cartesiano quadrado do conjunto formado por vértices em comum de u, v. O mesmo princípio é aplicado para a medida *Adamic/Adar*. Quando se compara a versão recursiva com a não recursiva, existe um conjunto base de vizinhos de u, v para o cálculo da similaridade. Para a versão recursiva, é feito o cartesiano deste conjunto com ele mesmo, para assim produzir pares de outros vértices para invocar recursivamente a função de similaridade.

4.1.4 Exemplo

Para clarificar o método recursivo, vamos ilustrar a sua aplicação para a medida *Jaccard*. Considere o grafo da figura abaixo e tome como referência os pares de vértices $(1, 2)$ e $(0, 3)$.



Jaccard Considerando-se o método de similaridade original *Jaccard*, para calcular a similaridade destes pares basta apenas observar a quantidade de vértices na vizinhança e aqueles em comum. Neste caso, o vértice 1 tem em sua vizinhança $\{0, 3\}$ e o vértice 2 tem $\{0, 4\}$. Portanto, o par $(1, 2)$ possui em comum o vértice 0 e, ao todo, são vizinhos de $(1, 2)$ os vértices$\{0, 3, 4\}$, assim:

$$\begin{aligned} S_{1,2} &= \frac{|\{0\}|}{|\{0,3,4\}|} = \frac{1}{3} \\ S_{0,3} &= \frac{|\{1\}|}{|\{1,2,5\}|} = \frac{1}{3} \end{aligned}$$

RecursiveJaccard O método recursivo *Jaccard* não depende apenas da quantidade de vértices na vizinhança de um par, mas também do quão similar são os pares de vértices nesta vizinhança. Seguindo a definição da Equação 4.2:

$$\begin{aligned} S_{1,2} &= \frac{S_{0,0}}{S_{0,0} + S_{0,3} + S_{0,4} + S_{3,0} + S_{3,3} + S_{3,4} + S_{4,0} + S_{4,3} + S_{4,4}} \\ S_{0,4} &= \frac{S_{2,2}}{S_{1,1} + S_{1,2} + S_{1,6} + S_{2,1} + S_{2,2} + S_{2,6} + S_{6,1} + S_{6,2} + S_{6,6}} \\ S_{0,3} &= \frac{S_{1,1}}{S_{1,1} + S_{1,2} + S_{1,5} + S_{2,1} + S_{2,2} + S_{2,5} + S_{5,1} + S_{5,2} + S_{5,5}} \\ S_{1,5} &= \frac{S_{3,3}}{S_{0,0} + S_{0,3} + S_{3,0} + S_{3,3}} \end{aligned}$$

Note que para o método recursivo é feito o quadrado cartesiano dos conjuntos no numerador e denominador comparado com o método *Jaccard*. Para cada par deste produto cartesiano, a função de similaridade é re-aplicada recursivamente. Ainda, obverve que $S_{1,2}$ e $S_{0,4}$ dependem entre si.

Patamares de similaridade Para o exemplo anterior, são apresentadas as matrizes de similaridade pelo método *Jaccard* e *RecursiveJaccard*:

***Jaccard*:**

	0	**1**	**2**	**3**	**4**	**5**	**6**
0	1,00			0,33	0,33		
1		1,00	0,33			0,50	
2		0,33	1,00				0,50
3	0,33			1,00			
4	0,33				1,00		
5		0,50				1,00	
6			0,50				1,00

#2		#3		
(1,5)	(2,6)	(0,3)	(0,4)	(1,2)
0,50	0,50	0,33	0,33	0,33

RecursiveJaccard:

	0	**1**	**2**	**3**	**4**	**5**	**6**
0	1,00			0,23	0,23		
1		1,00	0,26			0,41	
2		0,26	1,00				0,41
3	0,23			1,00			
4	0,23				1,00		
5		0,41				1,00	
6			0,41				1,00

#2		#3	#4	
(1,5)	(2,6)	(1,2)	(0,3)	(0,4)
0,41	0,41	0,25	0,23	0,23

Note que o algoritmo *Jaccard* gera apenas 3 patamares de similaridade, os pares com similaridade 1,0; 0,5 e 0,33. Já o algoritmo *RecursiveJaccard* gera 4 patamares de similaridade, diferenciando a similaridade de $(1, 2)$ da similaridade dos pares $(0, 3)$ e $(0, 4)$. Independente se isso vai gerar bons resultados ou não, o algoritmo recursivo faz um "ajuste fino", diferenciando melhor a similaridade entre os pares. No Capítulo 5 avaliamos se isto gera um benefício.

4.2 Direção da vizinhança dos vértices

Em [45], as medidas de similaridade de *Link Analysis* são calculadas desconsiderando a direção das arestas do grafo. Por exemplo, a medida *Jaccard* é proposta com a fórmula:

$$s_{x,y} = \frac{\left|\mathrm{Adj}_x \bigcap \mathrm{Adj}_y\right|}{\left|\mathrm{Adj}_x \bigcup \mathrm{Adj}_y\right|}$$

Ou seja, consideram-se todos vizinhos de um vértice, sem distinguir predecessores e sucessores. Talvez esta estratégia tenha sido tomada pelos autores deve-se ao fato de que trabalharam com o grafo de coautoria de artigos científicos. Neste grafo de coautoria, previamente mencionado na Seção 1.4.2, os vértices representam autores e arestas informam que dois autores já escreveram algum artigo juntos. Nesta situação, a semântica da

direção das arestas não é clara, por isso, possivelmente tenham desconsiderado a direção das arestas e as medidas de similaridade são calculados na vizinhança $\text{Adj}_\bullet$.

Por outro lado, aqui estamos lidando com o grafo de citação, em que vértices são artigos e arestas representam que um artigo cita outro em suas referências. A Figura 4.1 mostra um exemplo deste grafo de citação, em que os vértices enumerados representam artigos e as arestas dizem qual é o artigo que faz a citação (citante) e aquele que recebe uma citação (citado).

Dada a natureza direcionada deste grafo de citação, acreditamos ser importante também verificar o comportamento das medidas de similaridade considerando a direção das arestas. Por isso, redefinimos todas as medidas de similaridade implementadas neste trabalho em três configurações de vizinhança dos vértices (considere a Figura 4.1 e o vértice 4 como referência):

- Vizinhança dos predecessores (In): Os vizinhos predecessores são os vértices {1,2}.
- Vizinhança dos sucessores (Out): Os vizinhos sucessores são os vértices {5,6}.
- Toda vizinhança (Adj), quando é desconsiderada a direção das arestas: Os vizinhos são os vértices {1,2,5,6}.

Dadas estas três possíveis configurações da vizinhança, a formulação da medida *Jaccard* pode ser desdobrada nas seguintes versões:

$$
\begin{aligned}
s_{x,y}^{\text{in}} &= \frac{|\text{In}_x \bigcap \text{In}_y|}{|\text{In}_x \bigcup \text{In}_y|} \\
s_{x,y}^{\text{out}} &= \frac{|\text{Out}_x \bigcap \text{Out}_y|}{|\text{Out}_x \bigcup \text{Out}_y|} \\
s_{x,y}^{\text{undirected}} &= \frac{\left|\text{Adj}_x \bigcap \text{Adj}_y\right|}{\left|\text{Adj}_x \bigcup \text{Adj}_y\right|}
\end{aligned}
$$

É direto e simples fazer o mesmo desdobramento para as demais medidas de similaridade, em especial aquelas avaliadas nesta dissertação: *Jaccard*, *Adamic/Adar*, *SimRank*, *Katz* e o método recursivo proposto no início deste capítulo. Será mostrado no Capítulo 5 que estas configurações distintas de uma mesma medida de similaridade apresentam eficácias bem distintas, confirmando a importância de se adotar a direção das arestas no cálculo das medidas de similaridade.

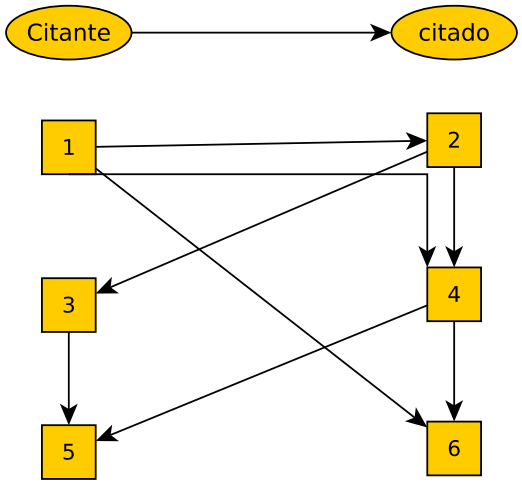


Figura 4.1: Exemplo de grafo de citação e a semântica da direção da aresta.

Capítulo 5

Avaliação das medidas de similaridade

5.1 O *Ground Thruth* de similaridade

Um passo fundamental do método científico é a avaliação dos resultados. Contudo, a avaliação da qualidade de medidas de similaridade pode ser uma tarefa difícil, pois não é incomum a falta de um gabarito, *ground truth,* principalmente por existir um custo alto (em tempo e subjetividade) para anotar dados. Realizar avaliação de medidas de similaridade com usuários reais pode não ser uma tarefa prática. Além de laboriosa, dependendo de como é executado, pode se tornar um trabalho difícil de ser reaproveitado para contínua melhoria e investigação dos algoritmos, já que a avaliação de usuários pode ter sido obtida com respeito a uma configuração de parâmetros do sistema, isto é, com um viés. Suponha a tarefa de busca e que resultados são apresentados aos usuários a fim de classificarem se cada resultado retornado é relevante ou não. O que foi retornado para o usuário classificar, em muitas construções de sistemas de busca, é parte do que de fato seria relevante. Assim, este usuário estaria classificando apenas uma parte do que este sistema selecionou, introduzindo um viés da configuração do sistema. A questão é: seria possível considerar esta anotação para um outro sistema de busca que tem outras configurações? Talvez este outro sistema revelaria outros resultados melhores ranqueados sem anotação prévia. Ainda, seria possível realizar uma nova anotação com os mesmos usuários ou novos, mas mantendo os mesmos critérios de avaliação que possivelmente são subjetivos? Deve-se considerar a hipótese de que uma nova anotação de usuários pode não estar reproduzindo as mesmas condições iniciais de outro experimento, dificultando, assim, a conclusão se um sistema é estatisticamente superior a outro. Pode-se argumentar a favor da anotação completa de relevância de todos resultados para um conjunto pré-determinado de buscas, contudo, quando lidamos com conjunto de dados enormes, por exemplo a Web ou bibliotecas digitais, existem restrições de recursos humanos, custo e tempo para completar a tarefa.

Esta é uma situação enfrentada por muitos pesquisadores nesta tarefa, e para contornar estas dificuldades, Haveliwala et al. [32] propuseram um método de avaliação de buscas por similaridades de páginas Web sem um *ground truth* direto. Na verdade, no lugar de um *ground truth*, utilizam diretórios Web (por exemplo, o *Dmoz* "*Open Directory*"[1], o antigo "*Google Directory*"[2] e "*Yahoo! Directory*"[3]) para avaliar a qualidade dos resultados de uma busca por similaridade. A metodologia deles se apoia na ideia de que há uma noção de similaridade implícita na hierarquia destes diretórios e, a partir disto, induzem um *ground truth* com a ordenação "correta" da relevância dos documentos dado um outro de busca. A ordenação produzida pela medida de similaridade é então comparada, estatisticamente, com este *ground truth* induzido. Esta metodologia, que faz uma clara separação entre a avaliação da medida de similaridade da avaliação em um sistema final, foi usada também em [23–25, 32].

Considerando a Figura 5.1, o *ground truth* induzido da hierarquia de diretório corresponde a uma ordenação parcial de similaridade dos documentos em relação ao documento de busca como se segue: os documentos no mesmo diretório/classe são mais similares do que aqueles em classes irmãs, que por sua vez, são mais similares do que os nas classes primas e assim por diante. Disto, deriva-se a seguinte ordenação parcial: classe do documento $\prec$ classe irmã $\prec$ classe prima $\cdots$ etc. Trata-se de uma ordenação parcial pois a partir dessa definição não é possível estabelecer uma ordem entre todos pares de documentos, haja vista aqueles que estão em um mesmo diretório. Desta ordenação deriva-se a noção de Distância Familiar [32], isto é, a distância dependente do grau de parentesco entre dois objetos na hierarquia em que estão contidos, podendo estar na mesma classe ou serem irmãos, primos ou não relacionados, quando estão suficientemente distantes ao longo da hierarquia. O que está implícito neste *ground truth* é que documentos em uma mesma classe são, em geral, mais similares que aqueles em outras classes e que esta similaridade é decrescente na medida em que o grau de parentesco de duas classes é menos "forte", de acordo com a definição adiante.

[1] http://www.dmoz.org/– Acessado em Julho de 2012.
[2] http://directory.google.com/– Acessado em Julho de 2009.
[3] http://dir.yahoo.com/– Acessado em Julho de 2012.

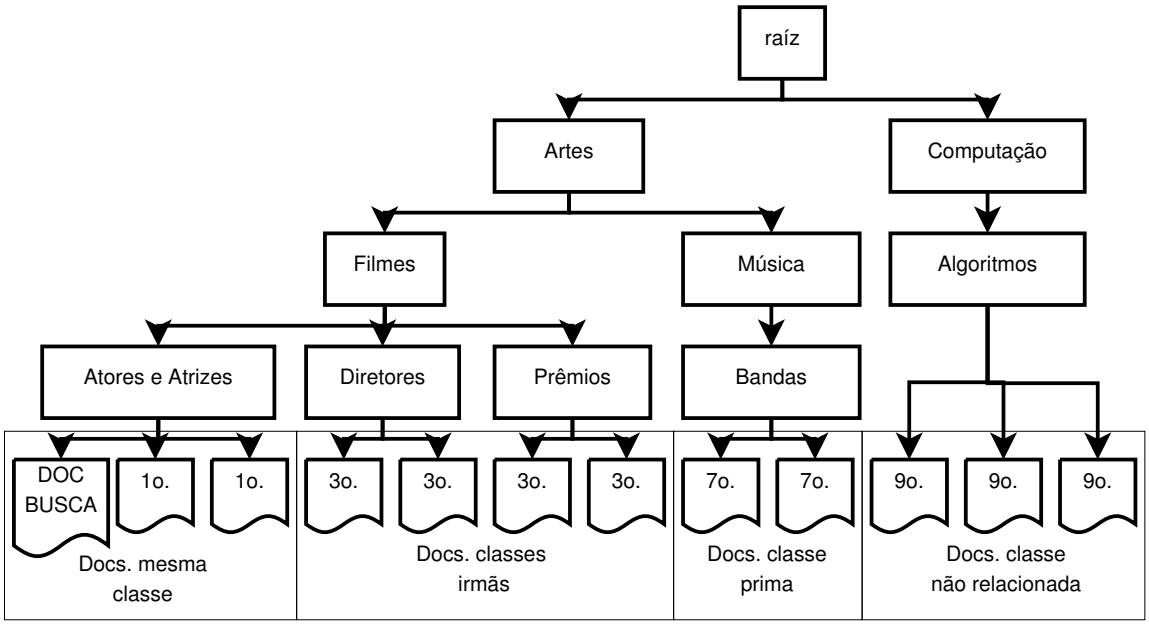


Figura 5.1: Mapeamento de uma hierarquia de diretórios para uma ordenação parcial de similaridade dado um documento de busca.

5.1.1 Definição formal da distância familiar

No trabalho de Haveliwala et al. [32], é formalizada esta noção de distância a partir de um documento de busca e outro na hierarquia de diretório como se segue:

> A distância familiar $d_f(s, d)$ de um documento de origem s e um outro documento d em uma classificação hierárquica é a distância da classe s até a classe mais específica que domina ambos s e d.

Em uma hierarquia de classe em que um documento pode estar classificado em níveis distintos desta árvore, a definição anterior apresenta o problema que $d_f(s, d) \neq d_f(d, s)$. Suponha o caso, tomando como base a Figura 5.1, um documento s na classe "Atores e Atrizes" e outro na classe "Música". Assim, a classe mais específica que domina ambas classes dos documentos é "Artes". Desta forma, $d_f(s, d) = 2$ ("Atores e Atrizes" $\rightarrow$ "Filmes" $\rightarrow$ "Artes"), mas $d_f(d, s) = 1$ ("Música" $\rightarrow$ "Artes"). Naquele trabalho, colapsam a hierarquia para um nível fixo e igual para todos documentos, garantindo a comutatividade.

Aqui, entretanto, propõe-se uma definição mais genérica e que garante esta propriedade comutativa, independente do nível em que um documento está:

> A distância comutativa familiar $D_f(a, b)$ de um documento a e um outro documento b em uma classificação hierárquica é a distância máxima da classe de a e da classe de b até a classe mais específica que domina ambos

a e b. Em outras palavras, é a distância máxima entre ambas classes até o "*Lowest common ancestor*" (LCA) [2] de a e b.

Esta função é comutativa, $D_f(a,b) = D_f(a,b)$, pois $D_f(a,b) = \max\{d_f(a,b), d_f(b,a)\}$, em que d_f é a função original de Haveliwala et al. [32]. Em termos práticos, para dois documentos em uma mesma altura da árvore, $D_f(a,b) = d_f(a,b) = d_f(b,a)$. Ou seja, considerando que Haveliwala colapsou a árvore para que todos documentos estivessem no mesmo nível, esta proposta não altera o resultado final para aquele trabalho. Já para este trabalho faz toda diferença, como será visto adiante, em nosso *ground truth*, dois documentos podem estar, e é bem frequente, em diferentes níveis da árvore.

A depender da altura desta árvore H_a, há diferentes valores possíveis para a distância, mais formalmente, $H_a + 1$ valores possíveis. Estes valores são nomeados, seguindo nomenclatura daquele trabalho:

Distância 0*: Mesma* — Documentos que estão na mesma classe;

Distância 1: *Irmã* — Documentos que estão em classes irmãs;

Distância 2: *Prima-1* — Documentos que estão em classes primas de primeiro grau;

Distância 3: *Prima-2* — Documentos que estão em classes primas de segundo grau;

⋮

Distância H_a*: Prima-*$(H_a - 2)$ — Documentos que estão em classes primas de grau $(H_a - 2)$;

Distância $H_a + 1$: *Não-Relacionada* — Documentos não relacionados. Estão hierarquicamente bem distantes, isto é, o LCA destes dois documentos é raiz da árvore e pelo menos um deles está no último nível da árvore.

No caso do trabalho de Haveliwala, $H_a = 3$, então há apenas um nível de primos e quatro valores possíveis. Já para este trabalho, como será visto adiante, $H_a = 4$, assim há primos de primeiro e segundo grau e cinco valores possíveis de distância entre documentos.

5.1.2 Definição da ordenação parcial familiar

Pretende-se usar a distância familiar D_f para criar uma ordenação parcial de similaridade de artigos dado um de referência. O princípio geral, já dito anteriormente, é que na média, a real similaridade dos documentos em relação a um de referência decresce monotonicamente de acordo com a distância familiar.

A ordenação parcial familiar, $\prec_{D_f}$, de todos documentos com relação a um documento de referência s é definida como $\prec_{D_f(s)} = \{(a,b) \mid D_f(s,a) < D_f(s,b)\}$.

Observe que esta ordenação de similaridade é bem fraca, haja vista que D_f produz apenas $H_a + 1$ valores. Isto produz poucas distinções na ordem, sendo que é mais notável a separação do conjunto de documentos mais similares dos demais. Não há distinção com os documentos não relacionados, possivelmente a maioria.

5.2 Sistema de classificação em computação *ACM* 1998

Nossa pesquisa envolve entidades acadêmicas, isto é, autores, artigos, conferências, revistas, etc. Neste caso, para estabelecer uma ordenação parcial de similaridade entre os pares de artigos, utilizamos os dados da biblioteca digital da *ACM* [21] e a categorização de cada artigo de acordo com o sistema de classificação em computação da *ACM* (*ACM Computing Classification System — CCS*)[4] [16]. Esta classificação do artigo é realizada pelo próprio autor no momento de publicação de seu artigo[5]. Este sistema de classificação foi definido em 1982 e foi revisado por um comitê em 1983, 1987 e 1991, sendo que a versão atual é uma revisão de 1998 e válida até 2012. O *CCS* é uma árvore e cada categoria, até o terceiro nível, é codificada. O código é composto por letras (para o nível mais alto) e sequência de números (que refletem um caminho até uma categoria na árvore). O quarto nível da árvore, não codificado, trata do assunto específico de uma publicação. Estas categorias são permanentes, contudo, na medida em que o sistema é revisado, algumas são aposentadas e outras expandidas, mas a categorização feita em um artigo permanece constante. A Figura 5.2 mostra todo o nível mais alto desta classificação, codificada por letras.

A Figura 5.3 mostra a expansão do segundo nível para duas categorias (***H.Information Systems*** e ***I.Computing Methodologies***). Já as Figuras 5.4,5.5 mostram a expansão do segundo até o quarto nível de duas subcategorias (***H.3.Information Storage and Retrieval*** e ***I.5.Pattern Recognition***).

[4] http://www.acm.org/about/class/1998/ – Acessado em Julho de 2012.
[5] http://www.acm.org/about/class/how-to-use – Acessado em Julho de 2012.

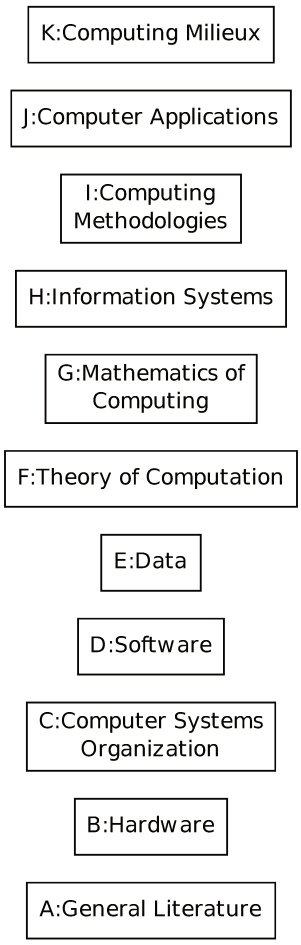


Figura 5.2: Categorias de mais alto nível da classificação *ACM-1998*.

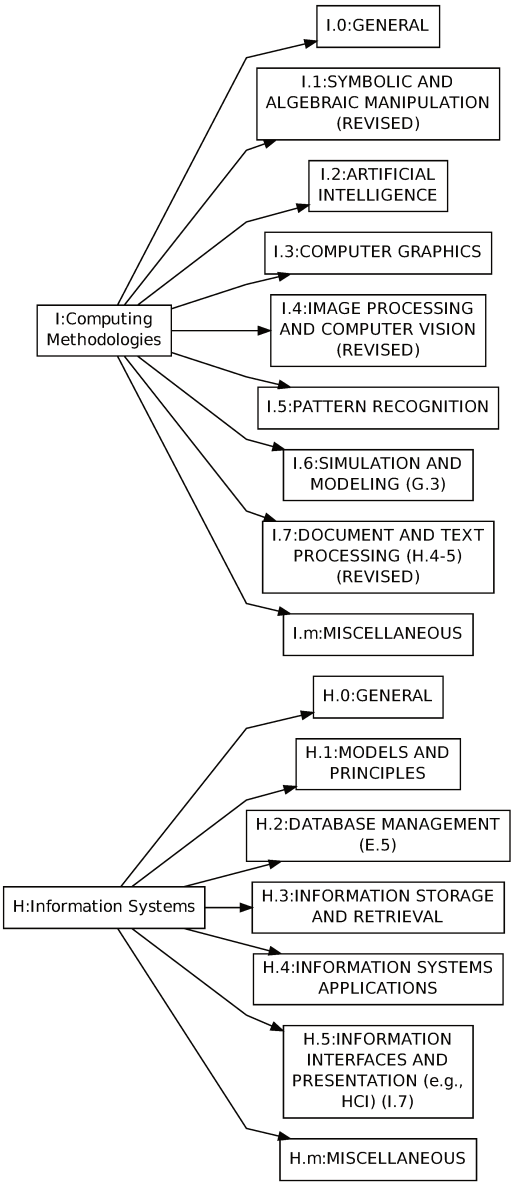


Figura 5.3: Categorias I e H, até terceiro nível, da classificação *ACM-1998*.

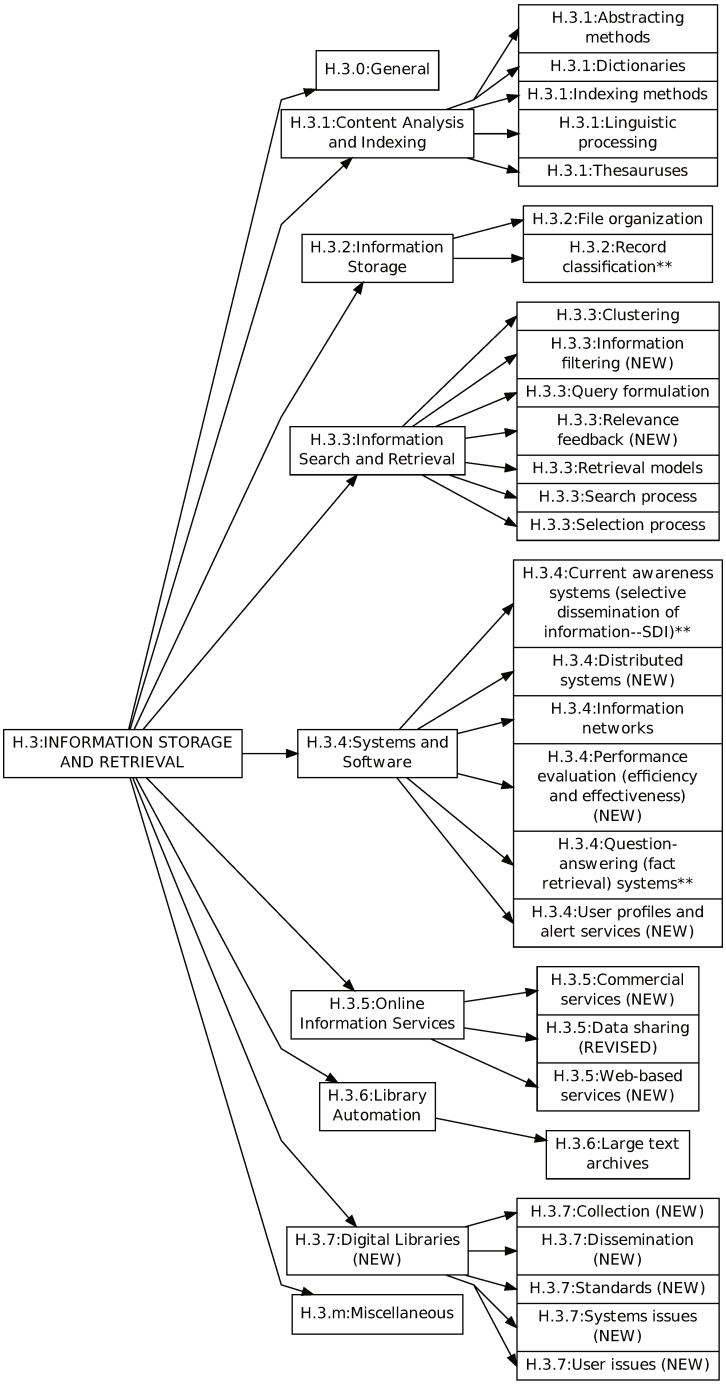


Figura 5.4: Sub-categoria H.3 da classificação *ACM-1998*.

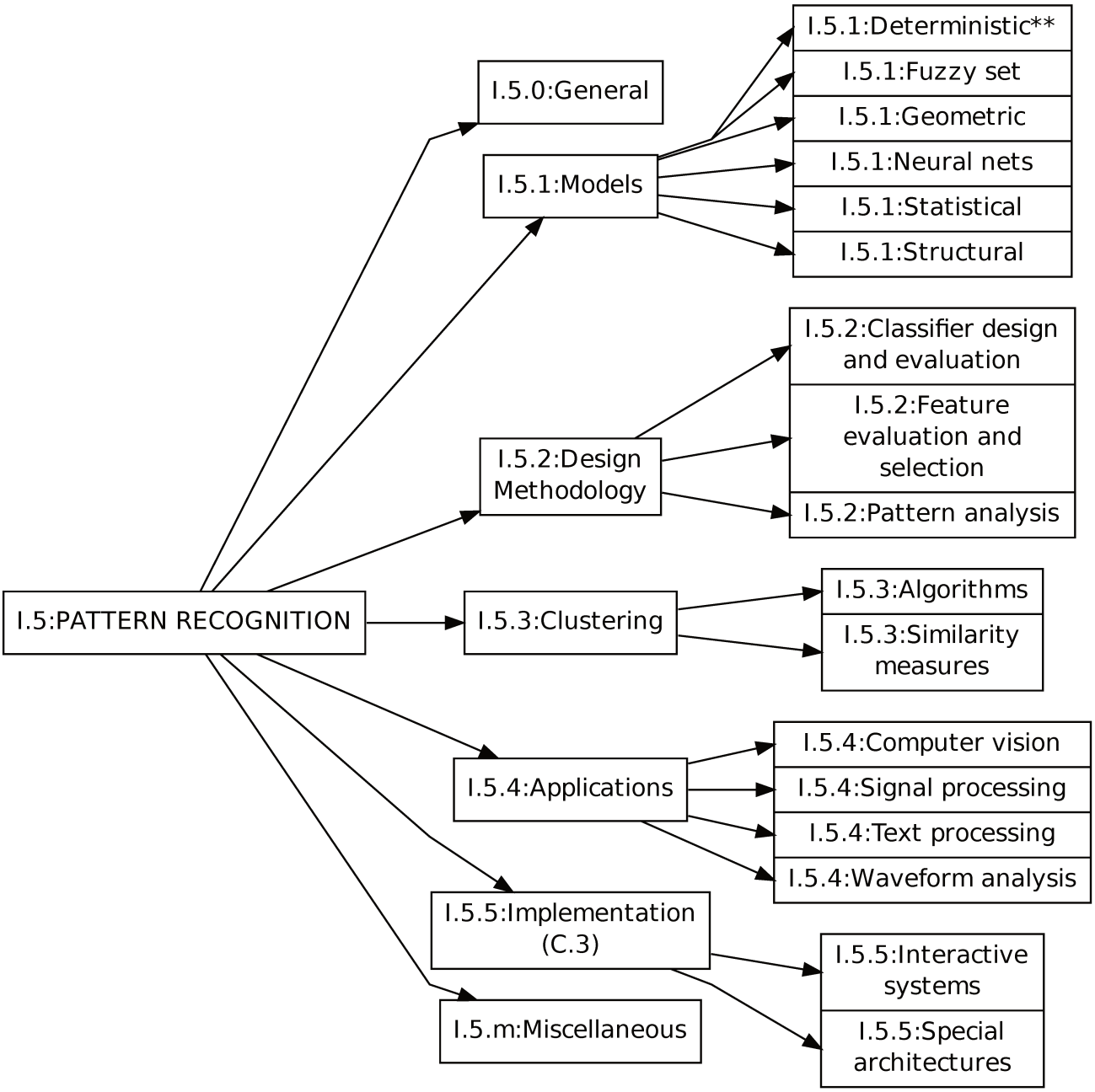


Figura 5.5: Sub-categoria I.5 da classificação *ACM-1998*.

Acreditamos que essa classificação desempenha função análoga ao diretório Web para artigos acadêmicos. Isto é, o que Haveliwala executou com páginas na WEB usando o diretório *DMOZ*, nós executamos com artigos científicos usando o sistema de classificação *ACM* 1998. Ou seja, com dados de categorização de cada artigo científico na hierarquia da *ACM* construímos um *ground truth* induzido para a similaridade entre artigos. Há certamente casos em que uma hierarquia não reflete precisamente a similaridade em documentos, assim como Haveliwala percebeu utilizando o *DMOZ*. Por exemplo, aqueles documentos em ***H.Information Systems/H.3.Information Storage and Retrieval/H.3.3.Information Search*** **and Retrieval/Clustering** são provavelmente mais similares àqueles em ***I.Computing Methodologies/I.5.Pattern Recognition/I.5.3.Clustering*** do que os em ***H.Information Systems/H.3.Information Storage and Retrieval/H.3.3.Information Search and Retrieval/H.3.3.Query Formulation***. A Figura 5.6, todavia, exemplifica acertadamente esta noção de similaridade familiar para dois artigos muito relacionados (observe que compartilham das categorias H.3.3 e G.2.2).

Por fim, imaginamos que uma ordenação parcial é ideal como um gabarito de similaridade se compararmos a uma ordenação total ou mesmo a uma matriz gabarito com valores de similaridade. Em ambos casos, estes gabaritos devem resolver casos difíceis de ordem e escala da similaridade entre todos pares, incluso para aqueles pares que a similaridade estariam empatadas. Em uma anotação de gabarito, ao comparar dois pares de similaridade, a e b, um poderia dizer que não se sabe o quanto a similaridade de a é maior que b, ou até mesmo dizer se a similaridade de a é maior o suficiente para caracterizar uma diferença significante. Um "juiz" hipotético diria $S(a) > S(b)$, já outro que $S(a) \approx S(b)$. Não necessariamente há discordância, apenas o segundo considerou que a diferença é de pequena escala. No caso de uma matriz gabarito, o problema é mais crítico: Deve-se quantificar a similaridade e, consequentemente, ter de responder o quanto $S(a) > S(b)$, em termos absoluto, $S(a) - S(b)$, ou relativo $\frac{S(a)}{S(b)}$. Resumindo, assim como alertado em [32], também temos a opinião de que um gabarito completo e correto de similaridade é suspeito.

Já o nosso gabarito é uma ordenação parcial de similaridade e possui apenas $H_a + 1$ níveis de similaridade, em que H_a é a altura da árvore de classificação dos artigos. No caso da classificação da *ACM*, $H_a = 4$, portanto há apenas 5 níveis de similaridade entre pares de artigo. Este número reduzido de intensidade de similaridade reflete, ao nosso ver, bem a noção de como um gabarito de similaridade deveria ser anotado: distinguindo níveis de similaridade que certamente são bem distintos e mantendo em um mesmo nível diferenças mais finas. Ou seja, com poucas distinções, contudo, quando existe, é precisa e significante.

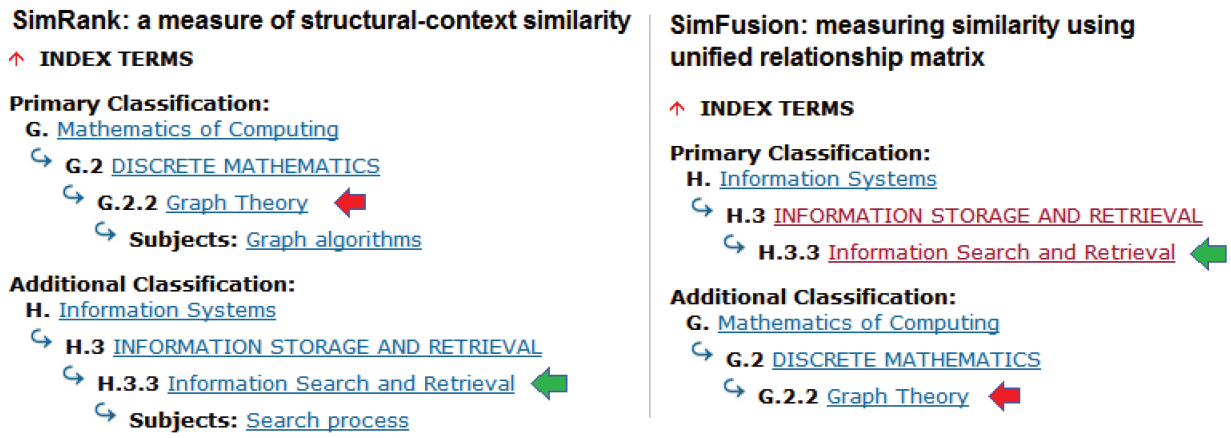


Figura 5.6: Dois artigos relacionados na classificação *ACM-1998*.

5.3 Comparando a similaridade

Como definido anteriormente, de uma classificação hierárquica é possível extrair uma noção de similaridade dos documentos classificados/organizados nela. No nosso caso, documentos são artigos acadêmicos e a hierarquia é o sistema de classificação em computação da *ACM*. O problema agora é como usar esta similaridade induzida da classificação como um gabarito para a similaridade calculada dos métodos de *Link Analysis* avaliados e propostos. A solução desse problema é derivada de [32]. Tanto da similaridade induzida da classificação, quanto na similaridade calculada dos métodos, é possível extrair uma ordenação de documentos tomando como base um de referência. Como já mostrado na Seção 5.1.2, sabendo a classificação de documentos na ACM-1998, podemos calcular uma ordenação parcial familiar, $\prec_{D_f}$, de todos documentos dado um de referência. Já no caso da similaridade calculada pelos métodos a se avaliar, é direto o cálculo desta ordem (total), basta ordenar todos documentos pelo valor da similaridade calculada com o documento de referência.

A solução do problema então é comparar duas ordenações, a derivada da hierarquia de classificação (o *ground truth*) e a outra do cálculo das medidas de similaridade em *Link Analysis*, para vários documentos de referência. Em [32], é sugerida a medida associação *Kruskal-Goodman* Γ [31]. Argumenta-se ser uma melhor alternativa em relação às medidas de correlação de *rank Spearman* e *Kendall*, pois a ordenação parcial derivada do *ground truth* produz muitos dados com mesmo posto, do inglês "*ties*".

Dado duas ordenações $\prec_a$ e $\prec_b$,

$$\Gamma(\prec_a, \prec_b) = 2\Pr[\prec_a, \prec_b \text{ concordam em } (x,y) \mid \prec_a, \prec_b \text{ ordenam } (x,y)] - 1.$$

Seja C o número de pares concordantes — (x,y) ranqueados na mesma ordem tanto em

$\prec_a$ quanto em $\prec_b$, isto é, ($x \prec_a y$ e $x \prec_b y$) ou ($y \prec_a x$ e $y \prec_b x$). Seja D o número de pares discordantes — (x, y) ranqueados em ordem distinta em $\prec_a$ e $\prec_b$, isto é, ($x \prec_a y$ e $y \prec_b x$) ou ($y \prec_a x$ e $x \prec_b y$). Então

$$\Gamma(\prec_a, \prec_b) = \frac{C - D}{C + D}$$

Os valores de Γ vão de 1 (associação perfeita) a -1 (associação perfeitamente inversa), sendo que o valor 0 significa que não há associação entre as ordenações.

Considera-se que, dado um artigo de referência s, se duas ordenações $\prec_{a(s)}$ e $\prec_{b(s)}$ diferem no valor de Γ com respeito a uma ordenação gabarito $\prec_{D_f(s)}$, então a ordenação com maior valor de Γ produziu uma melhor medida de similaridade dos documentos com respeito a s. Repetindo este teste para vários artigos de referência estaremos comparando, portanto, a medida de similaridade em geral.

Sendo assim, considerando a classificação hierárquica da *ACM*, um artigo de referência r e uma medida de similaridade S é possível construir duas ordenações dos artigos, uma derivada do gabarito da ordenação parcial familiar $\prec_{D_f(r)}$ e outra da medida de similaridade $\prec_{S(r)}$. O valor de $\Gamma\left(\prec_{D_f(r)}, \prec_{S(r)}\right)$ reflete a nossa medida de eficácia da medida de similaridade S para o artigo r. Para ter uma avaliação da eficácia de S globalmente, consideramos um subconjunto amostral A de artigos e agregamos a quantidade de pares concordantes C^{aggreg} e discordantes D^{aggreg}, quando comparamos $\prec_{D_f(r)}$ e $\prec_{S(r)}$ $\forall r \in A$. Então calculamos:

$$\Gamma^{agreg} = \frac{C^{aggreg} - D^{aggreg}}{C^{aggreg} + D^{aggreg}}$$

Esta fórmula reflete, portanto, nossa medida de eficácia da medida de similaridade S.

5.4 Avaliação segundo a ordenação parcial da classificação hierárquica de artigos

O objetivo do experimento aqui considerado consiste em avaliar a eficácia, medida por Γ^{agreg}, de várias medidas de similaridade em *Link Analysis*. Assim, selecionamos um subconjunto aleatório de teste T com 20% dos artigos da amostra do grafo de citação. Para cada artigo t de T, selecionamos um subconjunto aleatório de comparação X com 30% dos artigos da amostra do grafo. É então calculada a similaridade entre cada par de artigos, $s(t, x \in X)$. Esta similaridade é então convertida para uma distância (tomando seu valor negativo). Este conjunto de distâncias calculadas definem uma ordenação parcial de X com relação ao artigo t para a medida de similaridade S, $\prec_{S(t)}$. Do *ground truth*, é obtida a distância familiar destes pares, $D_f(t, x \in X)$ (veja Seção 5.1.1), o que define a ordenação parcial de X com relação ao artigo t de acordo com o *ground truth*, $\prec_{\text{ground truth}(t)}$.

	Número de Artigos	Número de Citações
1	18417	41568
2	36833	88286
3	61387	158506
4	85942	253355

Tabela 5.1: Tamanho de cada amostra do grafo de citação.

Então, calcula-se $\Gamma\left(\prec_{S(t)}, \prec_{\text{ground truth}(t)}\right)$, de acordo com a Seção 5.3. O valor de Γ^{agreg} é obtido para todos artigos $t \in T$, que é nossa medida de eficácia.

O conjunto de dados para o experimento é o mesmo descrito na Seção 3, o grafo de citação, que possui 122.774 artigos e 523.699 arestas de citação. Deste grafo são feitas quatro amostras, pelo algoritmo *ForestFire* (veja Seção 3.2). A Tabela 5.1 revela o tamanho de cada amostra. Considerando a matriz de similaridade calculada, esta avaliação compara 6% dos pares parcialmente ordenados desta matriz com relação ao *ground truth*. Esta matriz tem dimensão $|V(G)| \times |V(G)|$. Assim, se calculássemos para o grafo completo, estaríamos avaliando Γ^{agreg} para aproximadamente 904 milhões de pontos. Por essa enorme quantidade de pontos, argumentamos em favor da significância estatística.

Na Seção 3.2, avaliamos alguns métodos de amostragem de grafos existentes e propomos um novo. Esta proposta, *RankedForestFire*, apresentou uma melhora em relação ao *baseline* para amostras menores, isto é, conseguiu produzir amostras menores que mantêm melhor as propriedades do grafo original. Contudo, seguiremos nossos experimentos com amostras geradas pelo algoritmo *ForestFire*. A razão disto é que o algoritmo *ForestFire* já é reconhecido na literatura. Além disso, não queremos dar margem a hipótese de que os resultados de eficácia dos algoritmos de similaridade, em especial a medidas propostas, foram obtidos graças ao método de amostragem proposto e que poderia ser diferente se amostrado por métodos já aceitos pela literatura.

5.4.1 Eficácia do *baseline*

Agora avaliaremos o *baseline* de algoritmos de *Link Analysis* para calcular a similaridade entre artigos acadêmicos, de acordo com a metodologia e *ground truth* apresentados nas seções anteriores. Para isto, usaremos como conjunto de dados o grafo de citação, conforme descrito no Capítulo 3.

Selecionamos algumas medidas de similaridade do Capítulo 2, a saber: *Jaccard*, *Katz*, *SimRank*[6] e *Adamic/Adar*. Estas medidas foram escolhidas pois representam uma boa

[6]Não foi possível calcular as medidas SimRank e Katz para o caso não direcionado para a maior amostra do grafo. Para a implementação dessas medidas, precisava-se além dos 48GB de memória principal disponível.

miscelânea das estratégias de algoritmos de similaridade. *Jaccard* e *Adamic/Adar* são medidas que consideram apenas dados mais locais, isto é, a vizinhança local dos vértices. Além disso, *Adamic/Adar* pondera a importância dos vértices nesta vizinhança local, tal que um vértice "raro" em comum deve contribuir mais para a similaridade que outro mais frequente. Além disto, apresentou melhor eficácia na tarefa de predição não supervisionada de coautoria em [45]. Já o *Katz* é um algoritmo que usa dados de caminhos entre vértices e também é muito similar a métodos de *Kernel* em Grafos [27, 33]. *SimRank* é um algoritmo recursivo e que pode ser considerado como um *Random Walk* [36].

A Figura 5.7 mostra o desempenho das medidas de similaridade do *baseline*. O eixo-x é o número de arestas para cada amostra do grafo de citação considerada. A figura mostra o valor da associação (*Kruskal-Goodman* Γ) com o gabarito de cada medida de similaridade na medida em que a amostra do grafo aumenta. Quanto maior Γ, maior é a associação este a medida de similaridade e o gabarito de ordenação de similaridade. Percebe-se uma clara distinção da qualidade dos resultados entre as medidas *Jaccard*, *AdamicAdar* e as demais. Em seguida, há um grupo intermediário que compreende a medida *SimRank* (grafo não direcionado) e *Katz*. É notável também a diferença de qualidade entre o *SimRank* direcionado e não direcionado, sendo este último melhor. Por fim, os piores resultados é do *SimRank* direcionado. Isto é, do *baseline* considerado, destaca-se o desempenho do *Jaccard* e *AdamicAdar*, ambos calculados na vizinhança de vértices predecessores.

Dentre estas medidas consideradas, no aspecto de direção das arestas, parece ser uma exceção o caso do *SimRank*: calculado sob uma vizinhança não direcionada é melhor que direcionada. Para as demais medidas de similaridade, ocorre exatamente o inverso, isto é, medidas que consideram direção da aresta tendem a ter resultados melhores que aquelas que ignoram a direção da aresta.

No trabalho [45], é desconsiderado o efeito da direção das arestas na vizinhança dos vértices na aplicação proposta de predição de coautoria. Aqui, exploramos esta propriedade e verificamos que mesmo no *baseline* há grandes diferenças de resultado em uma mesma medida de similaridade se calculadas em diferentes configurações de vizinhança.

Esta análise sobre como a direção das arestas do grafo de citação afeta os resultados continua também para os métodos propostos e aplicações desta dissertação. Será evidenciado que esta propriedade é relevante e que esta percepção é uma contribuição deste trabalho.

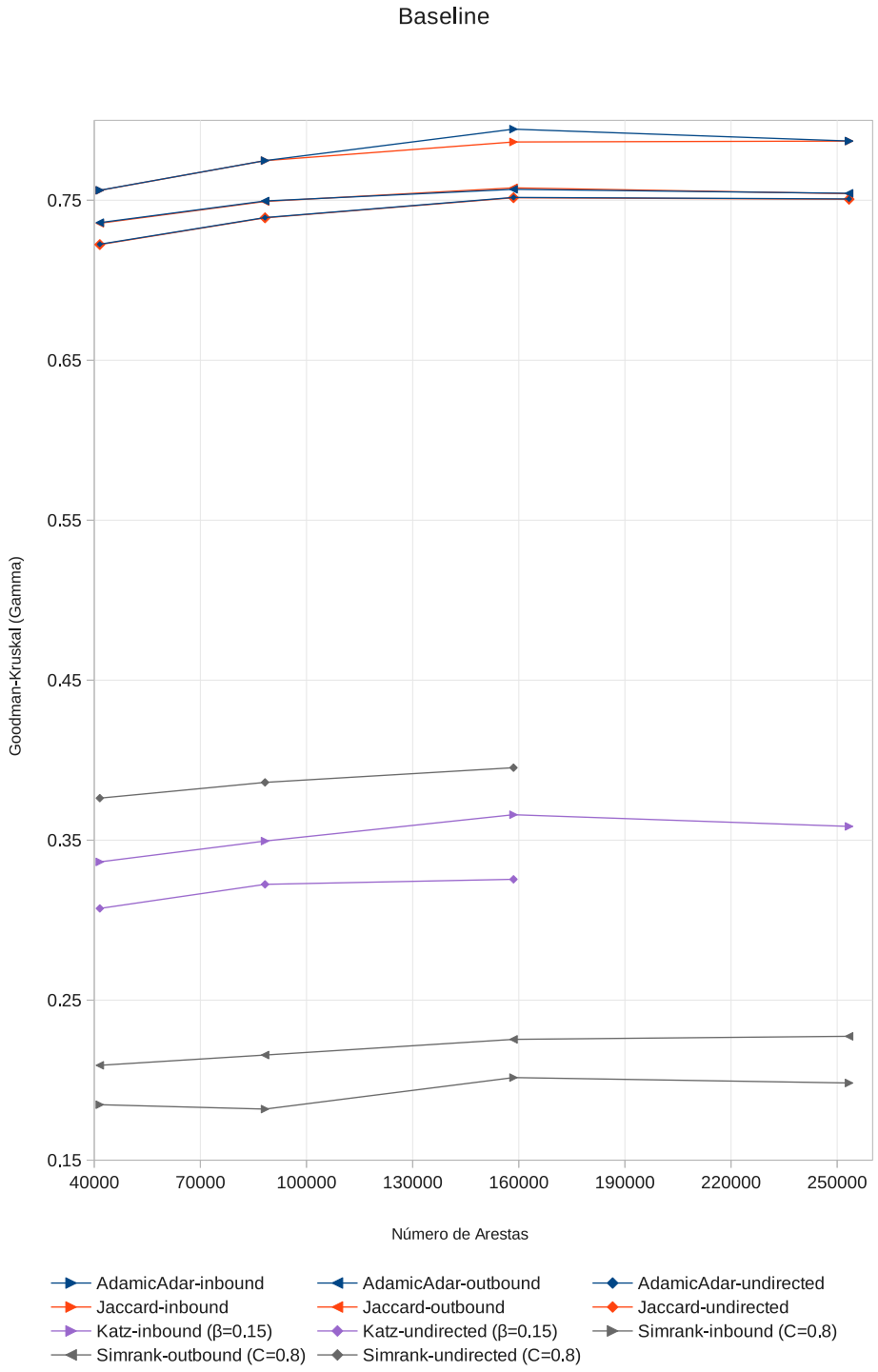


Figura 5.7: Avaliação medidas de similaridade do *baseline*.

5.4.2 Eficácia das medidas recursivas de similaridade

Esta seção compara a nossa proposta de medidas recursivas de similaridade contra suas versões originais, isto é, a medida *AdamicAdar* versus *RecursiveAdamicAdar* e *Jaccard* versus *RecursiveJaccard*. Cada uma destas medidas tem uma configuração de como considerar a direção das arestas do grafo de citação. Portanto, comparamos sempre cada par de medida de similaridade com a mesma configuração.

5.4.2.1 Melhor parâmetro

As medidas recursivas, com arestas direcionadas, possuem um parâmetro λ que define o peso relativo entre a importância da vizinhança de entrada e de saída de pares de vértices, conforme definições da Seção 4.1. A fim de estudar o comportamento destas medidas de similaridade com diferentes valores de λ, calculamos a associação Γ destas medidas com o gabarito para diferentes valores de λ. A Figura 5.8 mostra como Γ varia de acordo com λ para as medidas *RecursiveJaccard* e *RecursiveAdamicAdar*. O gráfico (A) mostra $\lambda \in [0, 1]$, já o gráfico (B) mostra $\lambda \in (0, 1)$. A ideia do gráfico (B) é mostrar em detalhes o que ocorre fora das bordas, quando $\lambda = \{0, 1\}$. No gráfico (A), percebe-se que há três patamares de Γ: Quando $\lambda = 0$, isto é, a configuração que considera apenas a vizinhança dos sucessores dos vértices, temos um desempenho intermediário. Para qualquer valor pouco maior de λ, isto é, quando é considerado também a vizinhança dos predecessores dos vértices, há um decaimento de Γ. Na medida em que λ aumenta, isto é, quando a importância da vizinhança dos predecessores dos vértices aumenta, não há alteração de Γ. Quando $\lambda = 1$, há um aumento drástico de Γ. Disto conclui-se que, para ambas medidas propostas *RecursiveJaccard* e *RecursiveAdamicAdar*, direcionadas, a melhor configuração é considerar apenas a vizinhança dos predecessores dos vértices.

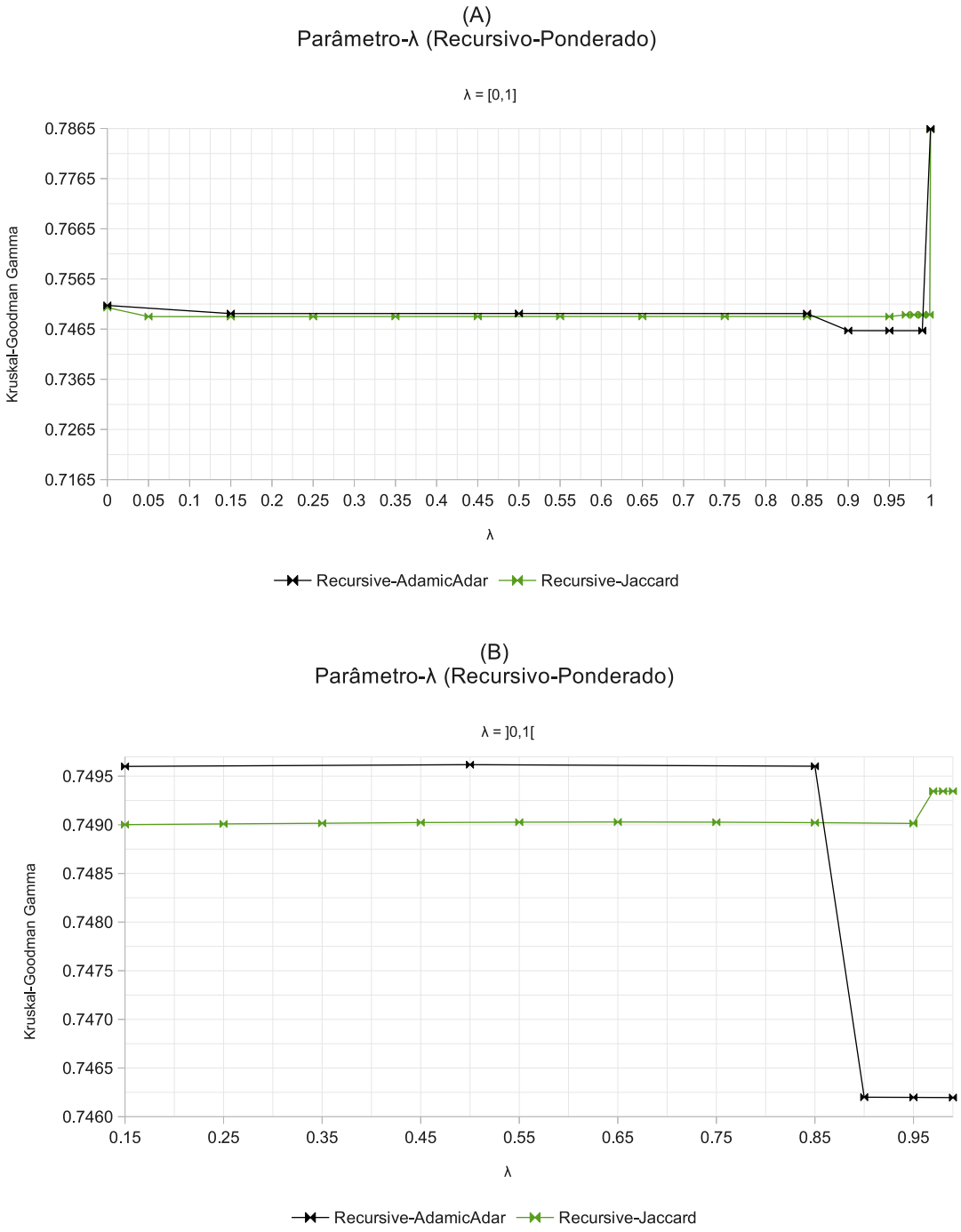


Figura 5.8: Qualidade da similaridade pelo parâmetro do *RecursiveJaccard*.

5.4.2.2 Direção da vizinhança dos vértices

As medidas recursivas de similaridade também são avaliadas em suas configurações de vizinhança direcionada ou não. No caso não direcionado, não há parâmetro λ. A Figura 5.9 compara os métodos recursivos com e sem direção das arestas. Para o caso direcionado, considera-se os parâmetro $\lambda = 1$ (vizinhança dos predecessores) e $\lambda = 0$ (vizinhança dos sucessores). Nesta figura, nota-se que para ambas medidas recursivas, a direção das arestas é importante e que quando calculadas considerando a vizinhança dos predecessores obtêm os melhores resultados. Fenômeno similar ocorreu com as medidas de similaridade do *baseline* selecionadas, como apresentado na Seção 5.4.1.

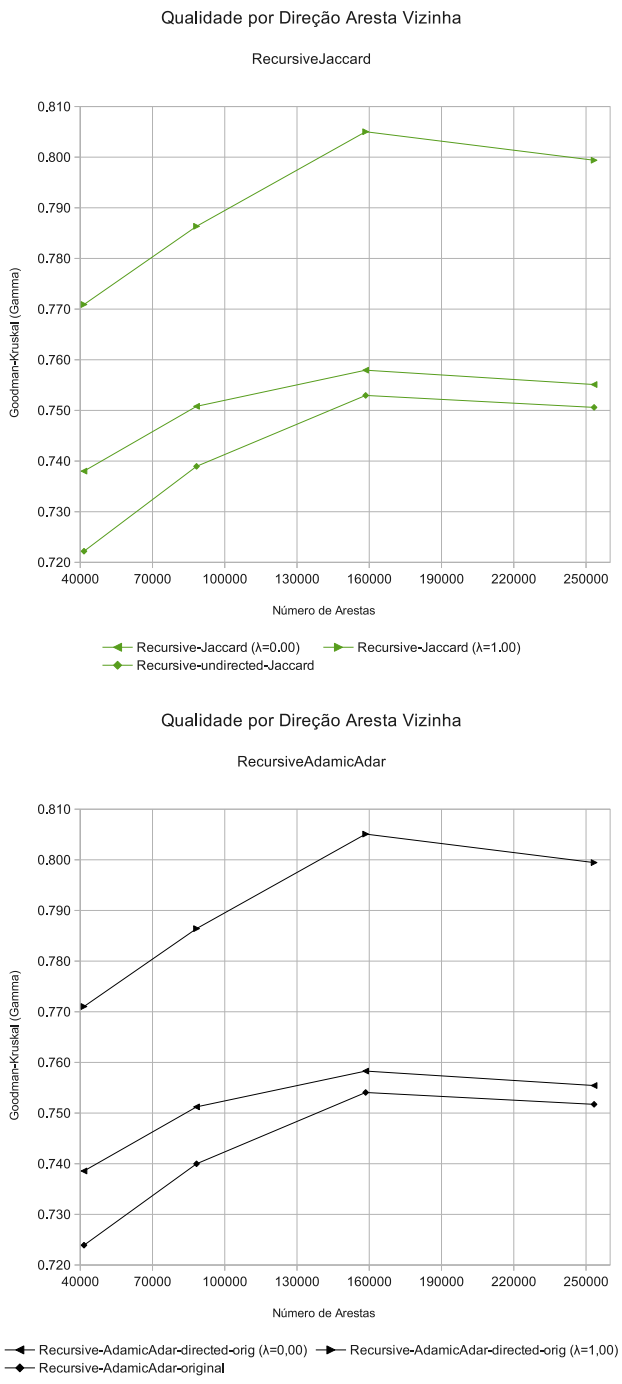


Figura 5.9: Comparação de medidas de similaridade recursivas dependendo da construção de vizinhança dos vértices.

5.4.3 Comparação do método recursivo contra as medidas locais *Jaccard* e *AdamicAdar*

Esta seção compara as duas melhores medidas de similaridade do *baseline, Jaccard* e *AdamicAdar*, contra as medidas propostas desta dissertação, que "transforma" estas medidas locais existentes em um esquema recursivo. Esta comparação avalia o comportamento das medidas para diferentes configurações de direção das arestas na vizinhança dos vértices.

5.4.3.1 Vizinhança não direcionada

No caso de não considerar a direção das arestas, toma-se como a vizinhança de um vértice todos aqueles que são sucessores ou predecessores. A Figura 5.10 mostra a associação das medidas *Jaccard*, *AdamicAdar* e suas versões recursivas com o gabarito, usando-se a medida *Goodman-Kruskal* Γ, para diferentes tamanhos de amostras do grafo de citação. Quanto maior Γ, melhor a associação com o gabarito. Nesta figura, percebe-se que as versões recursivas superam suas correspondentes do *baseline* para amostras de aproximadamente 160 mil arestas, o que corresponde a uma amostra de 50% do número de vértices em relação ao grafo original de citações. Esta diferença se torna menor para uma amostra de 70% do número de vértices, com aproximadamente 250 mil arestas de citação. Contudo, nota-se que a medida *RecursiveAdamicAdar* obtém o melhor resultado para esta configuração de vizinhança não direcionada.

5.4.3.2 Vizinhança dos sucessores

Esta seção avalia as medidas de similaridade, considarando-se a direção das arestas de citação. Toma-se como a vizinhança de um vértice apenas aqueles que são seus sucessores. A Figura 5.11 mostra a associação das medidas *Jaccard*, *AdamicAdar* e suas versões recursivas com o gabarito, usando-se a medida *Goodman-Kruskal* Γ, para diferentes tamanhos de amostras do grafo de citação. Quanto maior Γ, melhor a associação com o gabarito. Nesta figura, percebe-se que as versões recursivas superam suas correspondentes do *baseline* para quase todas amostras consideradas. No caso da amostra de 50% do número de vértices ($\sim$160 mil arestas de citação), a medida *Jaccard* se aproxima bastante da sua proposta recursiva. Assim como ocorreu para o caso não direcionado, a medida *RecursiveAdamicAdar* obtém os melhores resultados.

5.4.3.3 Vizinhança dos predecessores

Esta seção apresenta experimentos que consideram a direção das arestas de citação. Toma-se como a vizinhança de um vértice apenas aqueles que são seus predecessores. A Figura 5.12 mostra a associação das medidas *Jaccard*, *AdamicAdar* e suas versões recursi-

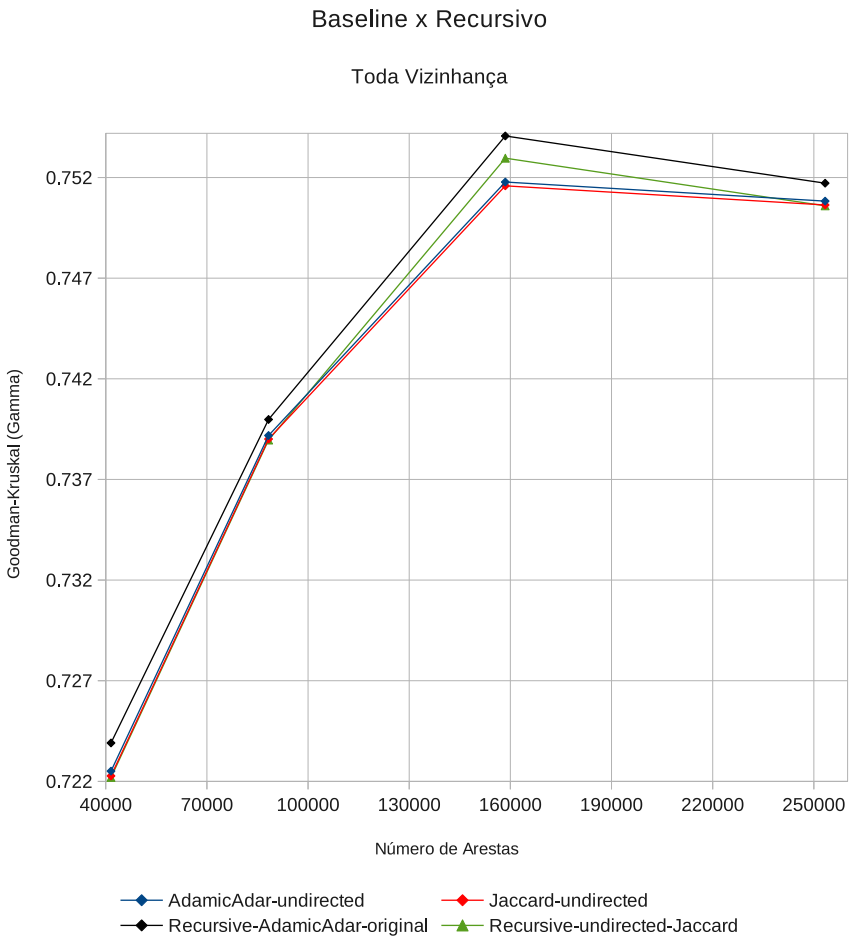


Figura 5.10: Comparação das medidas de similaridade recursivas vs. locais para a vizinhança não direcionada.

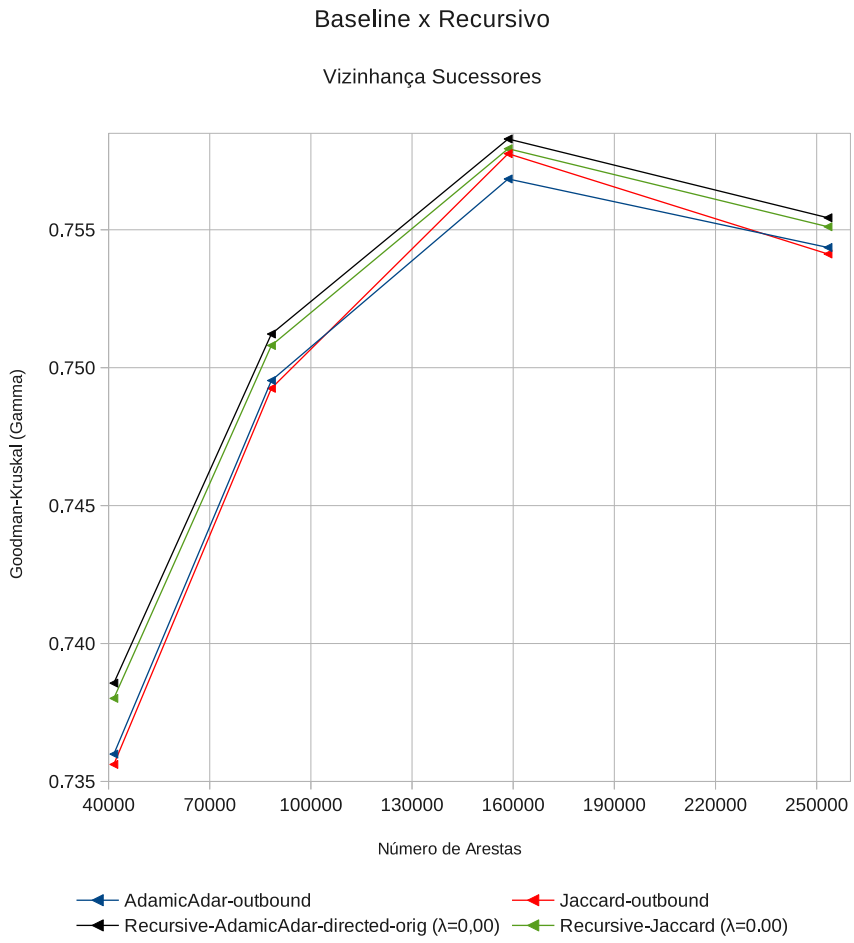


Figura 5.11: Comparação das medidas de similaridade recursivas vs. locais para a vizinhança dos sucessores.

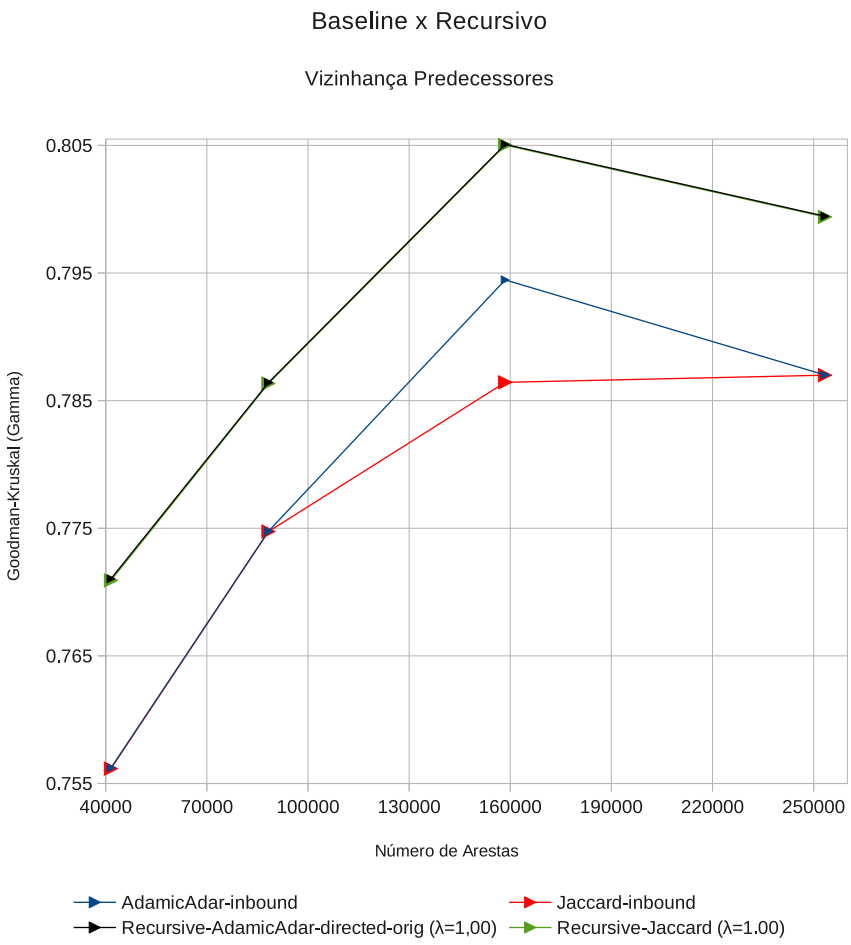


Figura 5.12: Comparação das medidas de similaridade recursivas vs. locais para a vizinhança dos predecessores.

vas com o gabarito, usando-se a medida *Goodman-Kruskal* Γ, para diferentes tamanhos de amostras do grafo de citação. Quanto maior Γ, melhor a associação com o gabarito. Nesta figura, percebe-se que as versões recursivas superam bastante suas correspondentes do *baseline* para todas amostras consideradas. Mais ainda, esta é a configuração de vizinhança que o *baseline* apresenta os melhores resultados e portanto, o método recursivo apresentado aqui supera todo o *baseline*, independente da direção das arestas.

Para as três configurações possíveis de vizinhança, dado um grafo direcionado ou não, as medidas de similaridade propostas superam o *baseline*. No caso de uma vizinhança não direcionada, a diferença dos resultados é menor, mas aumenta no caso da vizinhança dos sucessores. Para o caso da vizinhança dos predecessores, a diferença foi maior.

Amostra		**AdamicAdar**		**Jaccard**		**Katz**	**SimRank**
#Arestas	#Vértices	**predecessores**		**predecessores**		**pred.** ($\beta = 0,15$)	**não direc.** ($C = 0,80$)
		Baseline	Recursiva	Baseline	Recursiva	Baseline	Baseline
41568	18417	0,756	**0,771**	0,756	**0,771**	0,336	0,376
88286	36833	0,775	**0,786**	0,775	**0,786**	0,350	0,386
158506	61387	0,795	**0,805**	0,786	**0,805**	0,366	0,395
253355	85942	0,787	**0,800**	0,787	0,799	0,359	Não calculado.*

* Para a implementação, precisava-se além dos 48GB de memória principal disponível.

Tabela 5.2: Associação, Goodman-Kruskal (Γ), do gabarito de similaridade com a melhor configuração de cada medida de similaridade avaliada para diferentes amostras do grafo de citação.

5.4.3.4 Confrontando cada medida de similaridade em sua melhor configuração

A Figura 5.13 e Tabela 5.2 mostram o resultado das medidas de similaridade em suas melhores configurações, isto é, com exceção do *SimRank*, todas configuradas para considerar apenas a vizinhança dos predecessores e, no caso do *SimRank*, considera-se o grafo não direcionado. Fica claro que a versão recursiva dos algoritmos *Jaccard* e *AdamicAdar* superam todos algoritmos e que há uma grande diferença em relação aos algoritmos *SimRank* e *Katz*. Nossa intuição para explicar os resultados é que os algoritmos *SimRank* e *Katz* têm piores resultados porque introduzem "ruídos" ao propagar demais a similaridade. Isto é, o valor da similaridade de um par de vértices propaga-se bastante, influenciando o cálculo de similaridade de outros pares que não necessariamente estão "fortemente" relacionados, característica de medidas de similaridade globais. Por outro lado, as medidas *AdamicAdar* e *Jaccard* consideram apenas a vizinhança imediata de um par de vértices, isto é, apenas seus predecessores e sucessores, tornando-as medidas locais. Acreditamos que, para nosso experimento, estas medidas locais de similaridade não sofrem os ruídos introduzidos por pares de vértices não relacionados. Nesta hipótese, se justifica a diferença de qualidade de medidas locais versus globais. As versões recursivas melhoram as medidas locais pois, sob nosso ponto de vista, propagam localmente a influência da similaridade entre pares de vértices relacionados. Isto é, o que as medidas globais fazem no grafo como um todo, propagando a similaridade de um par para muitos outros pares de todo grafo, os algoritmos recursivos propagam apenas localmente, controlando assim o efeito "ruído". Desta forma, mostrou-se produtivo propagar o valor de similaridade calculado entre os pares, contudo, deve-se ter cuidado quanto ao alcance disto. Fica aqui a proposta para, em trabalhos futuros, se controlar este efeito para medidas globais como *SimRank* e *Katz*, por exemplo, reduzindo, respectivamente, seus parâmetros β e C.

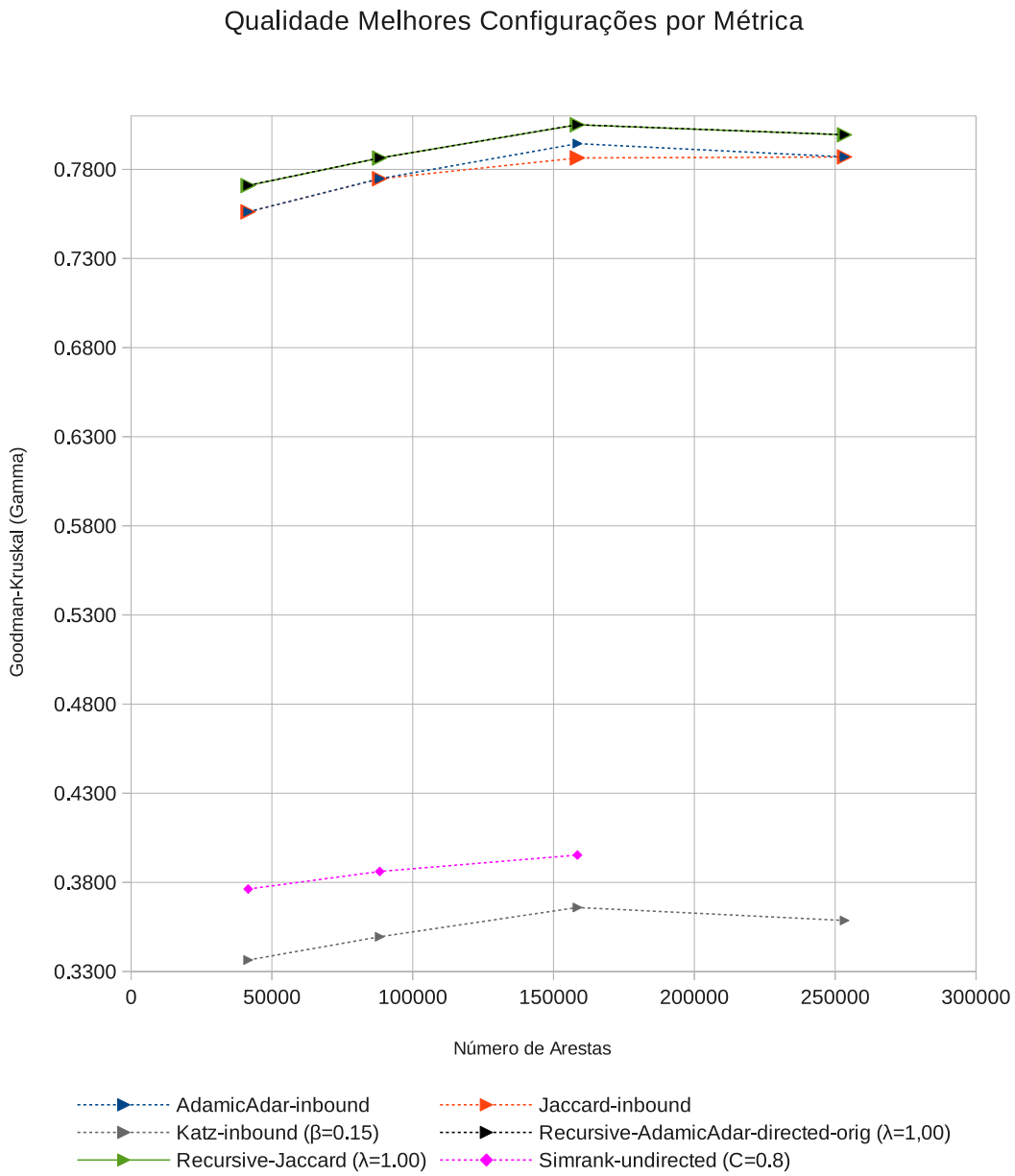


Figura 5.13: Comparação da melhor configuração de cada medida de similaridade avaliada.

Amostras		AdamicAdar				
Arestas	Vértices	Γ Local × Gabarito	Γ Recursiva × Gabarito	Γ Recursiva × Local	n	$\neq$
41.568	18.417	0,756	**0,771**	0,9995	20.357.784	**Sim**
88.286	36.833	0,775	**0,786**	0,9997	81.405.350	**Sim**
158.506	61.387	0,795	**0,805**	0,9997	226.123.926	**Sim**
253.355	85.942	0,787	**0,800**	0,9998	443.183.987	**Sim**

Tabela 5.3: Significância estatística da diferença da eficácia do algoritmo local × recursivo para a medida Adamic/Adar de acordo com o teste Z de Steiger.

Amostras		Jaccard				
Arestas	Vértices	Γ Local × Gabarito	Γ Recursiva × Gabarito	Γ Recursiva × Local	n	$\neq$
41.568	18.417	0,756	**0,771**	0,9998	20.357.784	**Sim**
88.286	36.833	0,775	**0,786**	0,9998	81.405.350	**Sim**
158.506	61.387	0,786	**0,805**	0,9999	226.123.926	**Sim**
253.355	85.942	0,787	**0,799**	0,9999	443.183.987	**Sim**

Tabela 5.4: Significância estatística da diferença da eficácia do algoritmo local × recursivo para a medida Jaccard de acordo com o teste Z de Steiger.

Significância Observa-se que em alguns casos a diferença de eficácia entre dois métodos é numericamente pequena, portanto, resta a hipótese se estes valores são ou não significantemente distintos, sob o ponto de vista estatístico. A medida de eficácia que usamos, Γ, é uma medida de correlação entre a ordenação de vértices obtida da similaridade calculada e a ordenação parcial do gabarito. Para testar se dois valores (de correlação) são significantemente distintos, usamos o teste Z de Steiger [63], para verificar se os resultados do nosso algoritmo recursivo proposto são significantemente distintos, portanto superiores, aos comparados com sua versão local. A Tabela 5.3 e 5.4 apresentam o resultado da análise de significância para as variações recursiva e local dos algoritmos Adamic/Adar e Jaccard, respectivamente, calculados considerando a vizinhança dos predecessores (cujos resultados foram os melhores). Além da medida Γ de correlação entre cada algoritmo e o gabarito, apresentam-se as correlações Γ entre as duas medidas de similaridade (local × recursiva) e a quantidade total de elementos (n) de todas ordenações produzidas para esta avaliação. Para todas amostras do grafo de citação, a diferença da eficácia entre o algoritmo recursivo proposto e o local é estatisticamente significante, portanto, confirmando a superioridade do método proposto.

5.4.4 Análise de convergência

O algoritmo das medidas propostas é iterativo, isto é, para cada iteração é calculada uma matriz de similaridade entre os vértices. A fim de verificar sua convergência, anotamos as diferenças entre as matrizes produzidas durante as iterações. Esta diferença é calculada em termos de "*Unity of Least Place*" (*ulp*) [29]. A diferença de dois números em termos de *ulp* representa quantos números, em ponto-flutuante, um programa pode representar no intervalo destes dois números. Isto é, $a + 1ulp$ é o menor número maior que a que um programa conseguiria representar. A escolha da diferença em *ulp* em relação à diferença relativa ε, é que se obtém um resultado mais rigoroso e, segundo, é que conseguimos comparar diferenças entre matrizes produzidas por diferentes algoritmos em uma mesma escala.

Define-se a diferença, em *ulp*, entre duas matrizes $M_{n,n}^{(k)}$ e $M_{n,n}^{(k+1)}$ como:

$$\Delta ulp\left(M_{n,n}^{(k)}, M_{n,n}^{(k+1)}\right) = \max\left\{\left|M_{i,j}^{(k)} - M_{i,j}^{(k+1)}\right|_{ulp} \forall i,j \in (n \times n)\right\}$$

Em que $|a-b|_{ulp}$ é a quantidade de números, ponto-flutuante, que podem ser representados por um programa no intervalo $(a, b]$. Em resumo, é o maior Δulp obtido comparando todos pares de valores, para cada posição, da matriz atual e a da iteração anterior.

A Figura 5.14 mostra como convergiram as matrizes de similaridades para o algoritmo iterativo proposto, para diferentes valores do parâmetro λ, tanto para a versão *RecursiveJaccard*, quanto *RecursiveAdamicAdar*. No eixo x está o número da iteração, já no eixo y, em escala logarítmica, está o Δulp. Assim, um ponto (x, y) representa o valor de $\Delta ulp\left(M_{n,n}^{(x)}, M_{n,n}^{(x-1)}\right)$ para na iteração x. A figura mostra, com apenas uma exceção, que a diferença entre as matrizes diminui na medida em que iteramos no algoritmo e que esta diferença se torna muito pequena em até 14 iterações. A exceção está na medida *RecursiveAdamicAdar*, que a partir da terceira iteração, não gera matrizes que diminuem a diferença com relação a anterior.

5.4.4.1 Método recursivo que converge para *Adamic/Adar*

Neste trabalho propomos uma versão recursiva do *AdamicAdar* que converge, chamada *RecursiveAdamicAdar-surprisal*.

A Figura 5.14 mostra que a proposta da medida recursiva para *Adamic/Adar*, vide Equação 4.3, não convergiu. Assim propomos uma alteração na fórmula recursiva, que usa o conceito de "surpresa", de *surprisal* em inglês, ou *Self Information*, da Teoria da Informação [18].

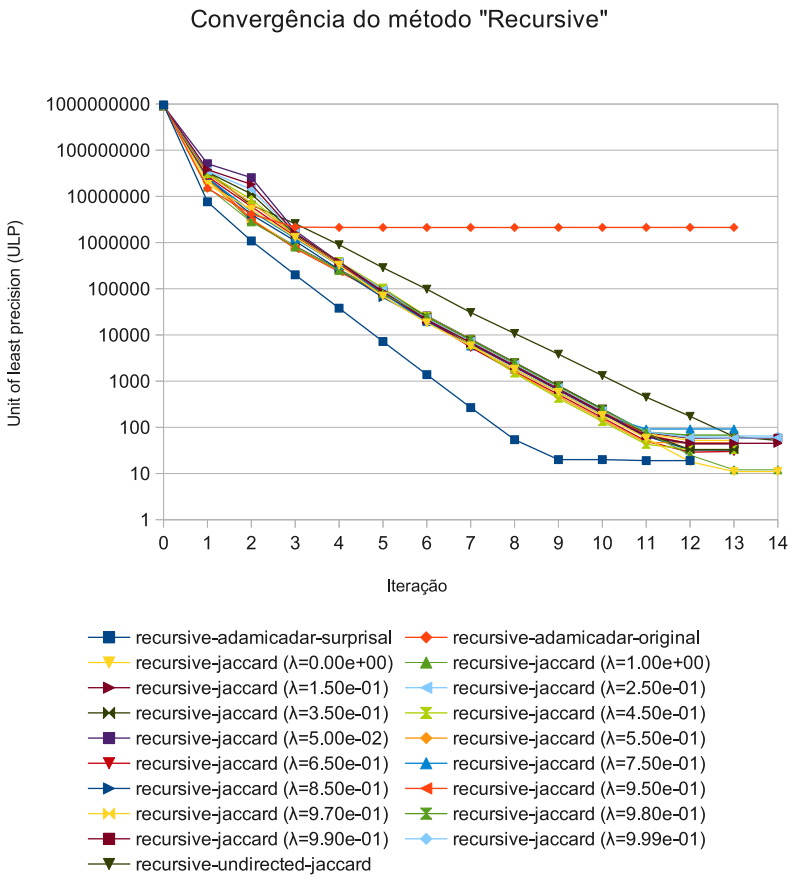


Figura 5.14: Convergência do método recursivo proposto.

# Arestas da Amostra	Goodman-Kruskal Γ	
	Recursive-AdamicAdar-surprisal	Recursive-AdamicAdar-original
41568	0,7239	0,7239
88286	0,7399	0,7400
158506	0,7540	0,7541
253355	0,7517	0,7517

Tabela 5.5: Eficácia da medida *RecursiveAdamicAdar* convergente.

$$\begin{aligned} \text{RecAdamicAdar.similarity}\,(M,u,v)^{surprisal} &= -\sum_{x,y\in(\text{Adj}_u \bigcap \text{Adj}_v)^2} \frac{1}{\log\left(\Pr\left(x,y \mid M\right)\right)} \\ \Pr\left(x,y \mid M\right) &= \sum_{z,w\in \text{Adj}_x \times \text{Adj}_y} M_{z,w} \end{aligned}$$

A Figura 5.14 mostra ainda que esta alteração produziu o algoritmo que mais rapidamente convergiu, com nove iterações. Em termos de qualidade de resultados, a Tabela 5.5 mostra que a versão original (*RecursiveAdamicAdar-original*) e a alterada (*RecursiveAdamicAdar-surprisal*) possuem resultados praticamente idênticos, portanto, a alteração trouxe o benefício de convergência (para os dados avaliados) sem comprometer a qualidade do método.

5.5 Avaliação segundo uma aplicação de *Query by Example*

A tarefa de *Query by Example* consiste em dado uma submissão de um ou mais documentos, um sistema de busca recupera outros documentos mais similares àqueles submetidos pelo usuário. Em nosso contexto, a partir de um artigo acadêmico de busca, nossa aplicação recupera outros mais similares. Nesta atividade não são consideradas palavras-chaves ou conteúdo textual, apenas a informação de referências/citações dos artigos. A ideia é que o grafo formado pelas citações/referências oferece informação semântica suficiente para concluir se dois artigos são similares ou não e em que nível, para assim, ser realizado um ranqueamento de similaridade. Assim como um sistema de busca, nem todos resultados relevantes são apresentados ao usuário, mas sim uma pequena porção daquilo que é mais relevante. A heurística disto é que um usuário não vai observar todos resultados para encontrar o que se deseja, pois isto consome muito tempo. Portanto, é importante apresentar uma lista resumida daquilo que é possivelmente relevante. A ideia de se usar

	Número de Artigos	Número de Citações
1	18417	41568
2	36833	88286
3	61387	158506
4	85942	253355

Tabela 5.6: Tamanho de cada amostra do grafo de citação.

um documento como busca é que um usuário possa querer ver mais documentos similares àquele que se está observando, ou então, quando é difícil de expressar uma busca a partir de uma linguagem ou critérios. O primeiro caso pode ser visto inclusive como uma recomendação (já que o usuário expressou interesse por um documento, que tal oferecer outros similares). Já o segundo caso é latente para buscas em dados multimídia, como busca de imagens e músicas por conteúdo. Assim, com uma aplicação de *Query by Example* para artigos acadêmicos, somos capazes de oferecer outros artigos relacionados, possivelmente de seu interesse, ao que o usuário está observando, sem a necessidade de uma busca explícita.

5.5.1 Experimento

Assim, construímos um experimento para validar se usar apenas o grafo de citação/referências e medidas de similaridade de artigos neste grafo é eficaz na atividade de busca de artigos mais similares. Nosso experimento funciona da seguinte forma: São selecionados aleatoriamente 15% dos artigos presentes em uma amostra do grafo de citação, formando o conjunto de busca. Para cada artigo deste conjunto de busca é retornada, de acordo com uma medida de similaridade de *Link Analysis*, uma lista ordenada dos $n = 20$ artigos mais similares. Então, cada artigo retornado é avaliado como relevante ou não relevante, ou seja, é considerado relevante se aquele retornado está, em relação ao artigo de busca, na mesma classe ou em uma classe irmã de acordo com o sistema de classificação em computação da *ACM* (*ACM Computing Classification System — CCS*) [16] (Detalhes sobre esta classificação e a definição de mesma classe ou classe irmã é encontrado na Seção 5.1). Por fim, para este conjunto de busca, é então avaliado o *Mean Average Precision* (*MAP*) e o *Normalized Discounted Cumulative Gain* (*nDCG*) [35], que revelam a eficácia do sistema para diversas buscas.

O conjunto de dados para o experimento é o mesmo descrito no Capítulo 3, o grafo de citação, que possui 122.774 artigos e 523.699 arestas de citação. Deste grafo, são feitas quatro amostras, pelo algoritmo *ForestFire* (veja Seção 3.2). A Tabela 5.6 revela o tamanho de cada amostra. Então, é calculado o *MAP* e o *nDCG* para cada medida de similaridade para todas amostras.

5.5.1.1 MAP

MAP é definido da seguinte forma:

$$\begin{aligned} \text{MAP} &= \frac{\sum_{q=1}^{Q} \text{AveP}\,(q)}{Q} \\ \text{AveP}(q) &= \frac{\sum_{k=1}^{n} \text{Precisão}\,(k,q)}{n} \\ \text{Precisão}\,(k,q) &= \frac{\sum_{j=1}^{k} r_q\,(j)}{k} \end{aligned}$$

Em que $r_q\,(j) = \{0,1\}$, 1 se o artigo retornado na posição j é relevante para a busca q, 0 caso contrário, isto é, irrelevante. Em outras palavras, para uma única busca q, podemos definir Precisão como:

$$\text{Precisão} = \frac{\text{número de artigos relevantes retornados}}{\text{número de artigos retornados}}$$

Sendo assim, AveP (q) é, para uma única busca q, a média/expectativa da Precisão considerando cada um dos n artigos retornados, neste caso, $n \leq 20$. Portanto, o MAP de um sistema de busca é a média de AveP para todas Q buscas realizadas no sistema, no caso $Q = 0,15\,|V\,(G_{\text{amostra}})|$. Quanto maior o MAP, $0 \leq \text{MAP} \leq 1$, mais eficaz é o sistema de busca.

5.5.1.2 nDCG

nDCG é definido da seguinte forma:

$$\begin{aligned} \text{nDCG} &= \frac{\sum_{q=1}^{Q} \text{nDCG}\,(q)}{Q} \\ \text{nDCG}(q) &= \frac{\text{DCG}(q)}{\text{DCG Ideal}(q)} \\ \text{DCG}(q) &= \sum_{i=1}^{min(20,n)} \frac{2^{\text{rel}_i} - 1}{\log_2\,(1+i)} \end{aligned}$$

Em que DCG Ideal(q) é o valor máximo de DCG que se pode calcular com os resultados retornados da busca e rel_i é o grau de relevância de um resultado de busca na posição i. São cinco graus de relevância, definidos de acordo com o parentesco, na classificação hierarquica da ACM, entre o artigo de busca e o artigo retornado na posição i, Seção 5.1.1:

Parentesco	rel_i
Mesma Classe	4
Irmãos	3
Primos-1	2
Primos-2	1
Não Relacionados	0

Para nosso experimento, consideramos $\text{nDCG}(q) = 0$ se $n = 0$, isto é, quando não há um resultado para uma busca. Outra possibilidade seria considerar $\text{nDCG}(q) = 1$, contudo, acreditamos que apesar do sistema não retornar nenhum resultado para uma busca provavelmente exista um artigo relevante, isto é, o sistema não foi capaz de encontrá-lo.

A medida de eficácia então é a média do nDCG para todas Q buscas realizadas no sistema, no caso $Q = 0,15\,|V(G_{\text{amostra}})|$. Quanto maior o nDCG, $0 \leq \text{nDCG} \leq 1$, mais eficaz é o sistema de busca.

5.5.2 Eficácia do *Baseline*

Adotamos o mesmo *baseline* de medidas de similaridade do Capítulo 5, Seção 5.4.1. Isto é: *Jaccard*, *Adamic/Adar*, *Katz* e *SimRank*[7]. A Figura 5.15 mostra a eficácia, em termos de *MAP*, de cada medida de similaridade do *baseline*, para cada amostra do grafo de citação na tarefa de *Query by Example*. Desta figura, nota-se que a melhor eficácia é obtida pelas medidas *Jaccard* e *Adamic/Adar* para a configuração de vizinhança não direcionada. Ainda, percebe-se que os métodos configurados para vizinhança não direcionada obtiveram maior *MAP*, seguido por aqueles configurados para vizinhança dos sucessores e, por fim, os piores resultados, configurados para a vizinhança dos predecessores. A Figura 5.16 mostra a eficácia, em termos de *nDCG*, para as mesmas medidas de similaridade. Os resultados são bem parecidos se comparados com a avaliação do *MAP*, contudo, percebe-se que o *SimRank*, pelo nDCG, supera o *Jaccard* e *Adamic/Adar*.

Apesar deste resultado concordar com o Capítulo 5, para as medidas *Jaccard* e *Adamic/Adar*, nota-se uma completa discordância em termos da relação entre eficácia e a configuração de vizinhança das medidas de similaridade. Naquele capítulo, constatou exatamente o contrário, que o melhor resultado se obtém na vizinhança dos predecessores, seguido pelos resultados considerando a vizinhança dos sucessores e, por fim, para o caso não direcionado. Contudo, deve-se alertar que a natureza destas avaliações são bem distintas. A do Capítulo 5 confronta cada medida de similaridade com o gabarito para um

[7]Não foi possível calcular as medidas SimRank e Katz para o caso não direcionado para a maior amostra do grafo. Param implementação, precisava-se além dos 48GB de memória principal disponível.

conjunto aleatório de posições da matriz de similaridade. Estas posições podem referenciar pares de artigos tanto similares quanto pouco similares, sendo que aquela avaliação se concentra em quão boa é a ordenação de similaridade em relação ao gabarito. Agora, todavia, estamos concentrando em obter, para cada busca, uma lista dos 20 artigos mais similares, isto é, estamos lidando com pares de artigos bastante similares apenas e avaliamos se cada artigo retornado é relevante ou não para a busca. Portanto, nenhum dos resultados invalida o outro, na verdade são complementares: enquanto o primeiro é uma avaliação mais ampla, que verifica o quão boa é uma ordenação de similaridade, tanto para elementos similares quanto os dissimilares, já o segundo método de avaliação é uma verificação mais fina na região dos artigos considerados mais similares de acordo com uma medida de similaridade.

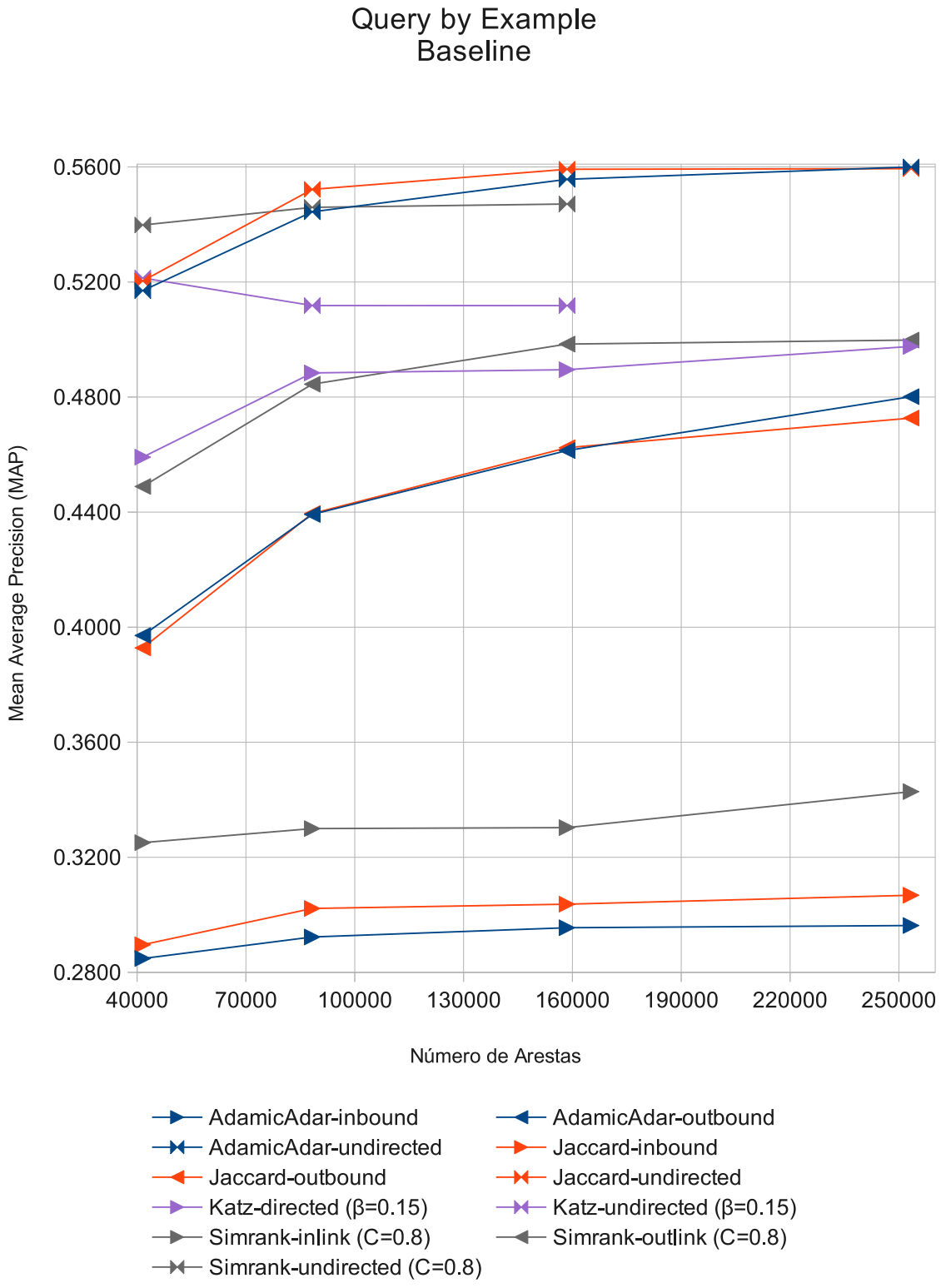


Figura 5.15: Desempenho (*MAP*) do baseline na tarefa de *Query by Example*.

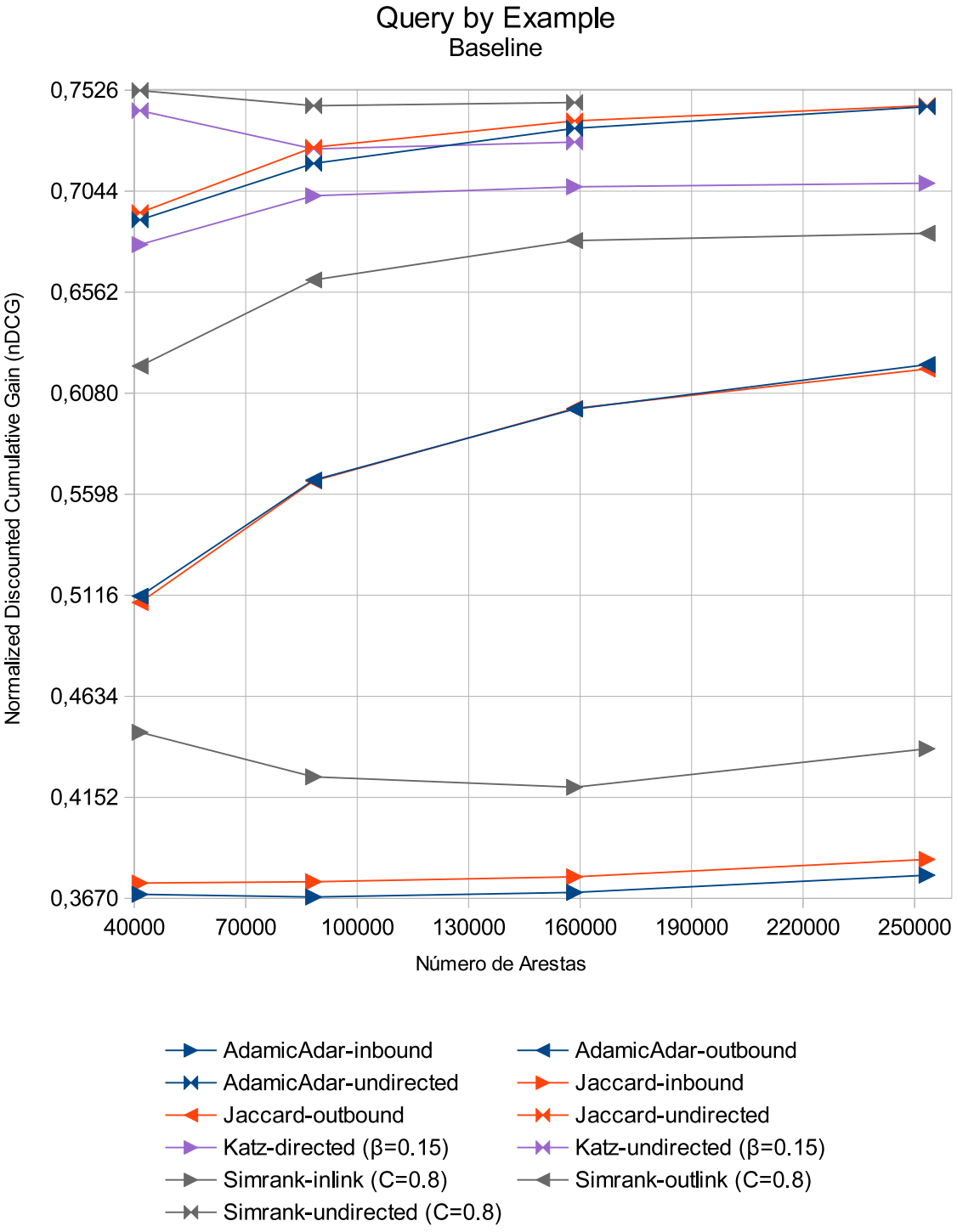


Figura 5.16: Desempenho ($nDCG$) do baseline na tarefa de *Query by Example*.

5.5.3 Medidas recursivas de similaridade

Esta seção tem como objetivo analisar o comportamento das medidas recursivas para diferentes valores de parâmetros e configurações de vizinhança do vértices no grafo direcionado.

5.5.3.1 Melhor parâmetro

As medidas recursivas, considerando a direção das arestas do grafo de citação, possuem um parâmetro λ que define o peso relativo entre a importância da vizinhança de predecessores e de sucessores de um vértice, conforme definições da Seção 4.1. A fim de estudar o comportamento destas medidas de similaridade para diferentes valores de λ, na aplicação de *Query by Example*, calculamos a eficácia *MAP (Mean Average Precision)* destas medidas variando λ. A Figura 5.17 mostra como o valor do *MAP* varia de acordo com λ para as medidas *RecursiveJaccard* e *RecursiveAdamicAdar*. O gráfico (A) mostra $\lambda \in [0, 1]$, enquanto o gráfico (B) mostra $\lambda \in (0, 1)$. A ideia do gráfico (B) é mostrar em detalhes o que ocorre fora das bordas, quando $\lambda \neq \{0, 1\}$ (lembrando que $\lambda = 0$, é considerada apenas a vizinhança dos sucessores e $\lambda = 1$ apenas os sucessores). No gráfico (A), percebe-se que nas bordas, foi obtido a pior eficácia, isto é, para a vizinhança dos predecessores e sucessores. Para a medida *RecursiveAdamicAdar*, o parâmetro próximo da borda também tem pior eficácia. Já para *RecursiveJaccard*, quando $\lambda \to 1$ é obtida a maior eficácia. O gráfico (B) mostra melhor o comportamento estável da eficácia de ambas as medidas de similaridade quando $\lambda \in (0,1; 0,9)$, mas aponta também um valor máximo do *MAP* para a medida *RecursiveAdamicAdar* quando $\lambda = 0,5$. Portanto, os melhores parâmetros, no caso direcionado, são: *RecursiveJaccard*: $\lambda \to 1$ (mas não $\lambda = 1$); *RecursiveAdamicAdar*: $\lambda = 0,5$.

Todavia, de acordo com a Seção 5.5.3.2, para estas medidas recursivas o melhor *MAP*, para a tarefa de *Query by Example*, é obtido para o caso do grafo não direcionado, sendo portanto de pouca utilidade este parâmetro λ. Note que na avaliação do Capítulo 5, que comparava a ordenação gerada por uma medida de similaridade e o *ground truth*, o parâmetro λ foi relevante.

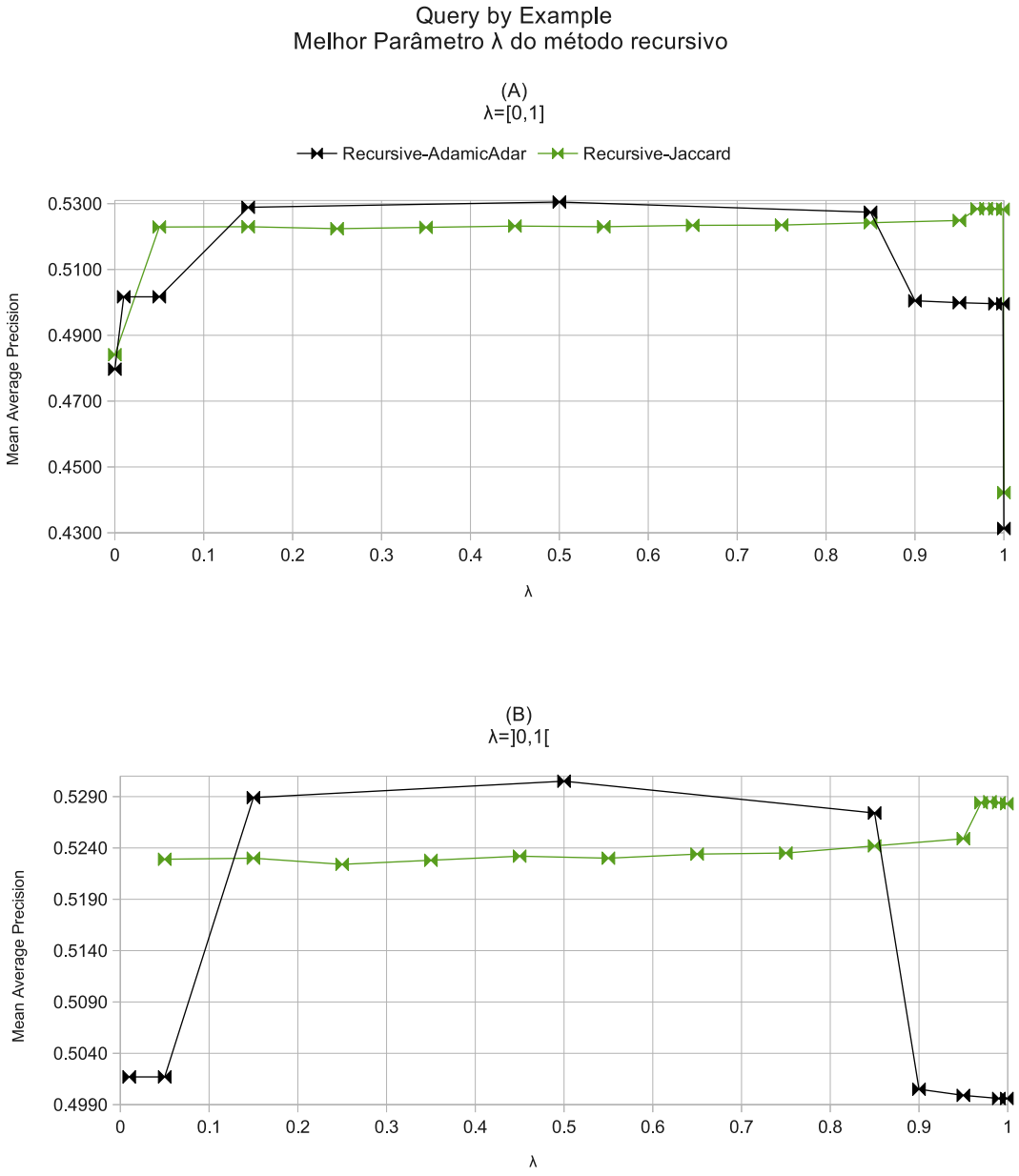


Figura 5.17: Qualidade da similaridade pelo parâmetro do *RecursiveJaccard*.

5.5.3.2 Direção da vizinhança dos vértices

Assim como ocorreu na avaliação de eficácia da ordenação de similaridade, Seção 5.4.2.2, consideramos avaliar a hipótese de que a direção das arestas influencia a eficácia na aplicação de *Query by Example* usando as medidas de similaridade propostas. Assim, as medidas recursivas são avaliadas em suas configurações de vizinhança direcionada ou não. No caso não direcionado, não há parâmetro λ. A Figura 5.18 mostra a eficácia, medida pelo *MAP* (*Mean Average Precision*), dos resultados de busca usando os métodos recursivos como medida de similaridade considerando diferentes configurações da direção das arestas do grafo de citação. Para o caso direcionado, considera-se o parâmetro $\lambda = 1$ (vizinhança dos predecessores) e $\lambda = 0$ (vizinhança dos sucessores). Nesta figura, nota-se que para ambas as medidas recursivas, a direção das arestas influencia os resultados, contudo, negativamente. Isto é, as medidas *RecursiveJaccard* e *RecursiveAdamicAdar* obtêm desempenho bem superior se for desconsiderada a direção das arestas, para esta aplicação de *Query by Example*. Fenômeno similar observou-se na avaliação das medidas de similaridade do *baseline*, Seção 5.5.2.

Este resultado é bastante interessante em termos práticos, já que em uma biblioteca digital de artigos acadêmicos, a informação de arestas predecessoras do grafo de citação, isto é, os artigos citantes, não está disponível imediatamente quando um artigo é recém publicado. Ou seja, se um sistema de busca dependesse principalmente destas arestas, os artigos recentemente publicados dificilmente estariam bem ranqueados em uma busca da aplicação *Query by Example*, em outras palavras, seria um limitador de como esta busca poderia ser usada em um sistema real. Portanto, com este resultado, pode-se esperar em um ranqueamento de similaridade também artigos acadêmicos recentemente publicados, além daqueles publicados há mais tempo.

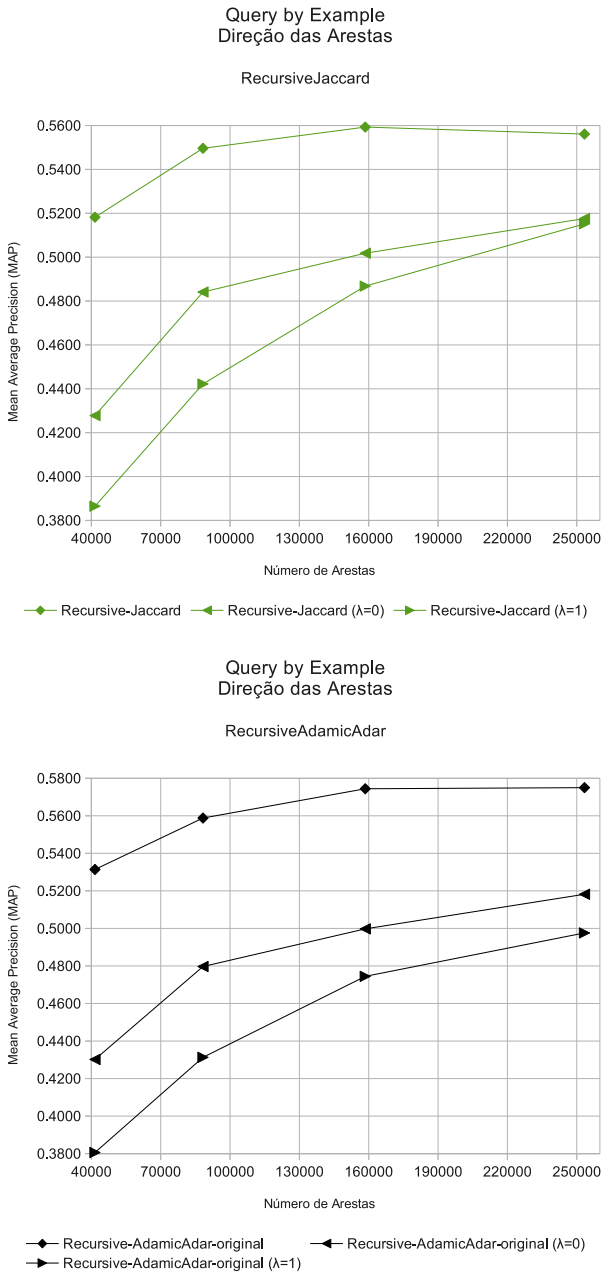


Figura 5.18: Eficácia (*MAP*) das medidas de similaridade "Recursive" dependendo da configuração de vizinhança dos vértices, na aplicação *Query by Example*.

5.5.4 Comparação do método recursivo contra as medidas locais *Jaccard* e *AdamicAdar*

O objetivo aqui é comparar as duas melhores medidas de similaridade do *baseline, Jaccard* e *AdamicAdar*, contra as medidas recursivas propostas na aplicação de *Query by Example*. Esta comparação avalia o comportamento das medidas de similaridade para diferentes configurações de direção das arestas na vizinhança dos vértices.

5.5.4.1 Vizinhança dos sucessores

Considerando o grafo de citação direcionado, são comparadas as medidas de similaridade recursivas contra sua versão não recursiva, calculadas apenas para a vizinhança dos sucessores de um vértice. A Figura 5.19 mostra a eficácia, medida pelo *MAP* (*Mean Average Precision*), das medidas *Jaccard*, *AdamicAdar* e suas versões recursivas para a aplicação de *Query by Example*. Nesta figura, percebe-se que as versões recursivas superam bastante suas correspondentes do *baseline* para todas amostras consideradas.

5.5.4.2 Vizinhança dos predecessores

Considerando o grafo de citação direcionado, são comparadas as medidas de similaridade recursivas contra sua versão não recursiva, calculadas apenas para a vizinhança dos predecessores de um vértice. A Figura 5.20 mostra a eficácia, medida pelo *MAP* (*Mean Average Precision*), das medidas *Jaccard*, *AdamicAdar* e suas versões recursivas para a aplicação de *Query by Example*. Nesta figura, percebe-se que as versões recursivas superam bastante suas correspondentes do *baseline* para todas amostras consideradas.

5.5.4.3 Vizinhança não direcionada

Considerando o grafo de citação não direcionado, são comparadas as medidas de similaridade recursivas contra sua versão não recursiva, calculadas para toda adjacência de um vértice. A Figura 5.21 mostra a eficácia, medida pelo *MAP* (*Mean Average Precision*), das medidas *Jaccard*, *AdamicAdar* e suas versões recursivas para a aplicação de *Query by Example*. Nesta figura, percebe-se que a medida recursiva *RecursiveAdamicAdar* supera bastante as demais. Mais ainda, esta é a configuração de vizinhança que o *baseline* apresenta os melhores resultados e portanto, o método recursivo baseado na medida *AdamicAdar* supera todas outras avaliadas, para todas possíveis configuração de vizinhança, seja direcionada ou não. A Figura 5.22 apresenta o mesmo resultado, mostrando as demais medidas de similaridade do *baseline* em suas melhores configurações.

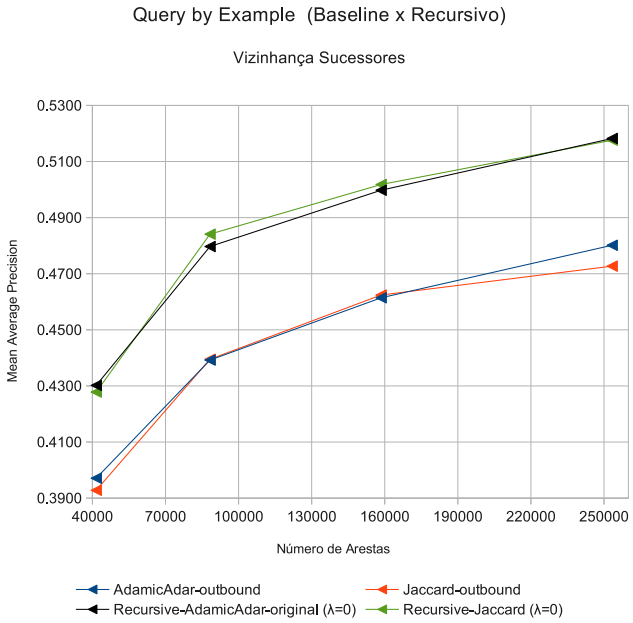


Figura 5.19: Eficácia (MAP) das medidas recursivas, para a vizinhança dos sucessores, comparadas com a versão não recursiva do baseline na aplicação de *Query by Example*.

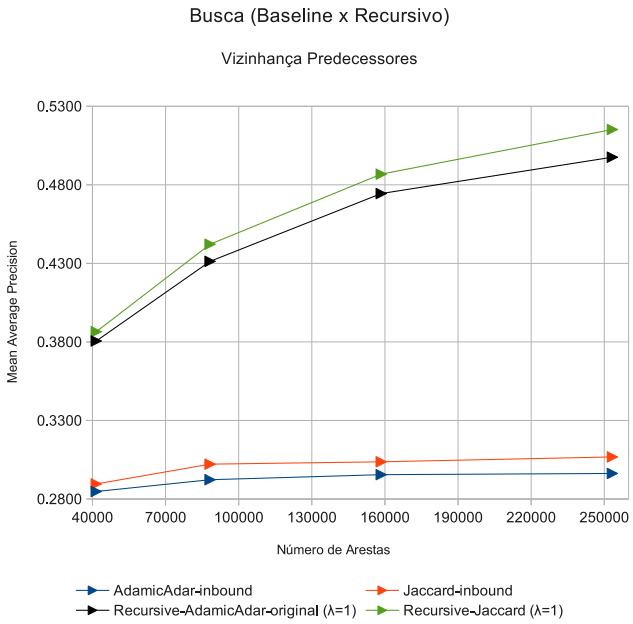


Figura 5.20: Eficácia (MAP) das medidas recursivas, para a vizinhança dos predecessores, comparadas com a versão não recursiva do baseline na aplicação de *Query by Example*.

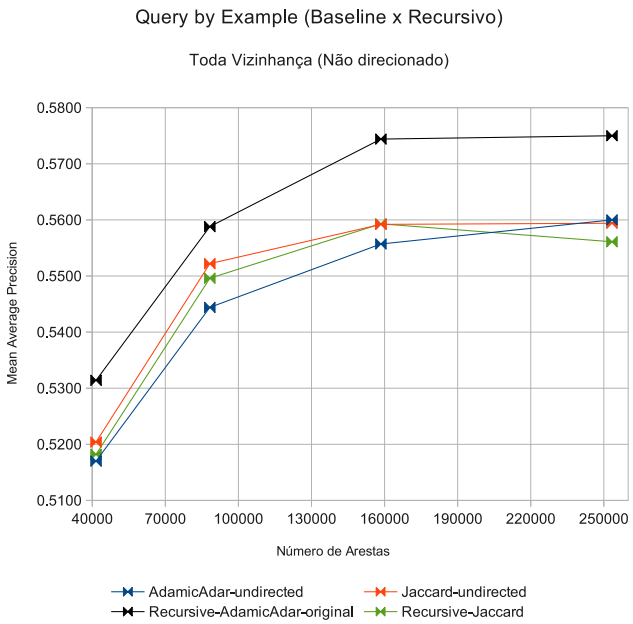


Figura 5.21: Eficácia (*MAP*) das medidas recursivas, para grafo não direcionado, comparadas com a versão não recursiva do baseline na aplicação de *Query by Example*.

5.5.4.4 Confrontando cada medida de similaridade em sua melhor configuração

As Figuras 5.22 e 5.23 e as Tabelas 5.7 e 5.8 mostram as eficácias, medidas respectivamente pelo *MAP* e *nDCG*, das medidas de similaridade em suas melhores configurações, isto é, todas configuradas para considerar o grafo de citação não direcionado, ou seja, toda adjacência de um vértice é considerada no cálculo de similaridade. Como pode ser observado, a versão recursiva do algoritmo *Adamic|Adar* supera todos algoritmos, contudo, o mesmo não acontece com a versão recursiva da medida *Jaccard*.

Intervalo de confiança Com o objetivo de avaliar estatisticamente se a eficácia do método recursivo é diferente da sua versão não recursiva (baseline) nesta avaliação de *Query by Example*, a Figura 5.24 mostra os intervalos de confiança de 95% para os valores do *MAP* e *nDCG* para as medidas de similaridade *Adamic/Adar* e *Jaccard*. Nota-se que no caso do *Adamic/Adar* a versão recursiva é significamente superior a versão não recursiva, por outro lado, a diferença numérica da eficácia do *Jaccard* recursivo não é significativa se comparada com a versão não recursiva.

Amostra		**AdamicAdar**		**Jaccard**		**Katz**	**SimRank**
#Arestas	#Vértices	**não dir.**		**não dir.**		**não dir.** ($\beta = 0,15$)	**não direc.** ($C = 0,80$)
		Baseline	Recursivo	Baseline	Recursivo	Baseline	Baseline
41568	18417	0,5170	0,5314	0,5204	0,5182	0,5214	**0,5398**
88286	36833	0,5444	**0,5588**	0,5522	0,5496	0,5118	0,5459
158506	61387	0,5557	**0,5744**	0,5592	0,5593	0,5118	0,5471
253355	85942	0,5600	**0,5750**	0,5594	0,5561	*	*

* Para a implementação, precisava-se além dos 48GB de memória principal disponível.

Tabela 5.7: Eficácia, *Mean Average Precision*, da melhor configuração de cada medida de similaridade na tarefa de *Query by Example* para diferentes amostras do grafo de citação.

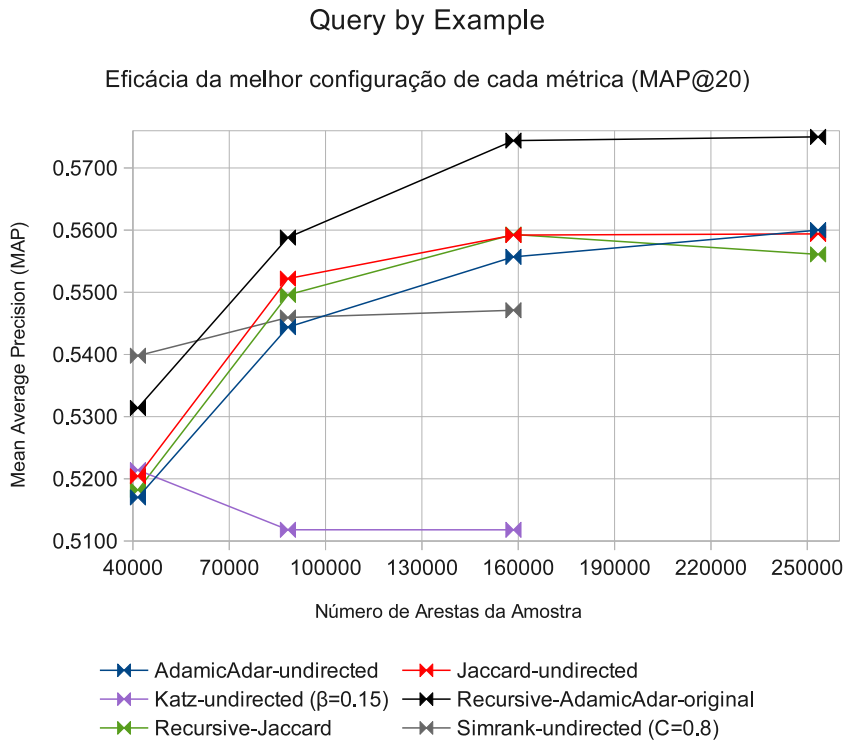


Figura 5.22: Eficácia (*MAP*) da melhor configuração de cada medida de similaridade na tarefa de *Query by Example*.

Amostra		**AdamicAdar**		**Jaccard**		**Katz**	**SimRank**
#Arestas	#Vértices	**não dir.**		**não dir.**		**não dir.** ($\beta = 0,15$)	**não direc.** ($C = 0,80$)
		Baseline	Recursivo	Baseline	Recursivo	Baseline	B$nDCG$aseline
41568	18417	0,6907	0,7044	0,6942	0,6926	0,7428	**0,7524**
88286	36833	0,7176	0,7313	0,7253	0,7241	0,7245	**0,7452**
158506	61387	0,7344	**0,7543**	0,7379	0,7430	0,7278	0,7467
253355	85942	0,7447	**0,7562**	0,7452	0,7435	*	*

* Para a implementação, precisava-se além dos 48GB de memória principal disponível.

Tabela 5.8: Eficácia, *nDCG*, da melhor configuração de cada medida de similaridade na tarefa de *Query by Example* para diferentes amostras do grafo de citação.

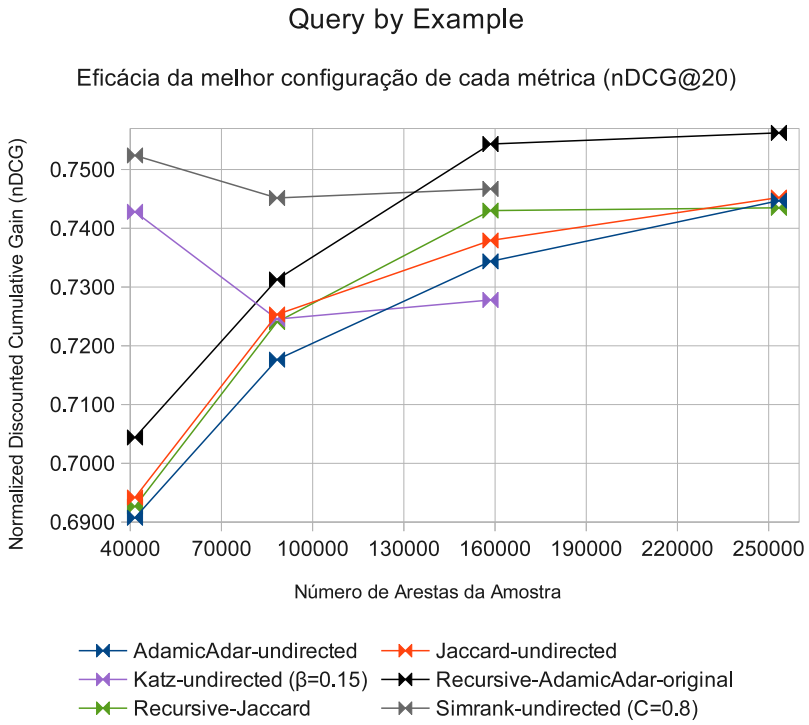
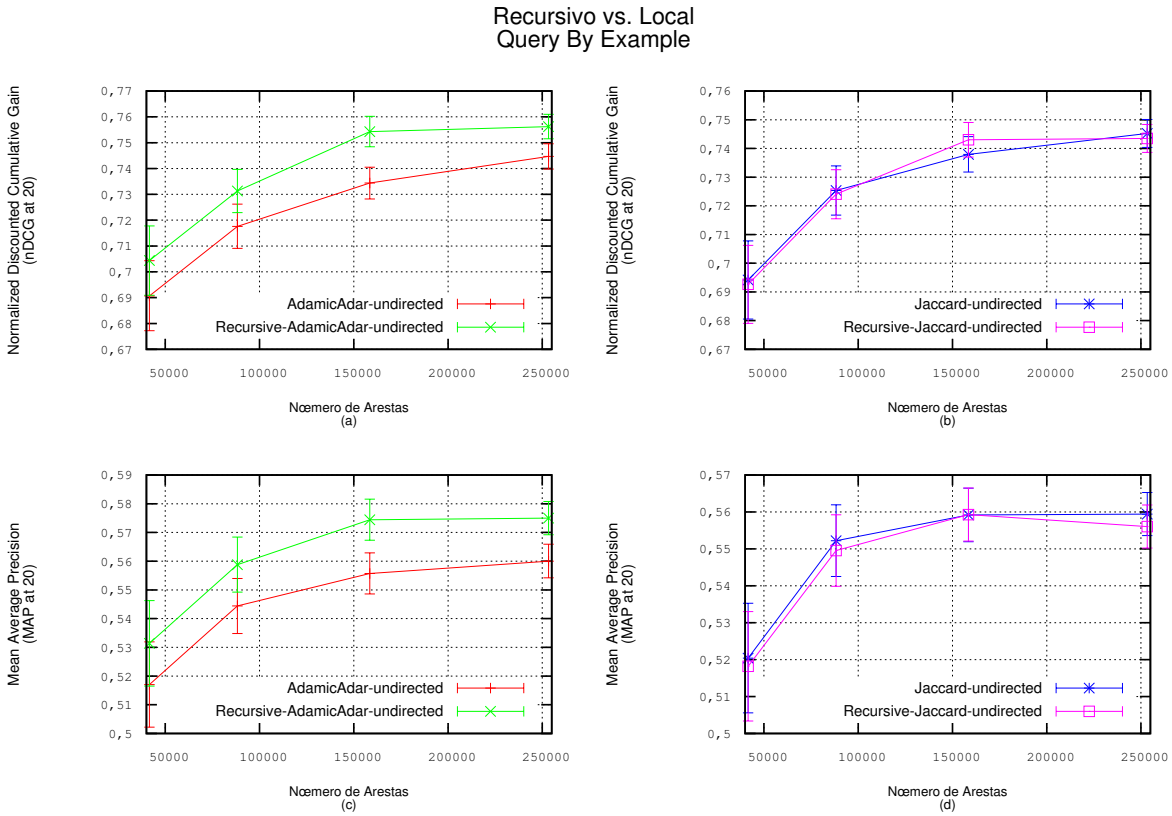


Figura 5.23: Eficácia (*nDCG*) da melhor configuração de cada medida de similaridade na tarefa de *Query by Example*.



d

Figura 5.24: Intervalo de confiança da eficácia da versão local e recursiva do *Adamic/Adar* e *Jaccard*.

Capítulo 6

Conclusões

6.1 Contribuições e discussão dos resultados

Neste trabalho, foi realizado um estudo comparativo de diversas medidas de similaridades de *Link Analysis*. No Capítulo 2, estudamos a formulação de diferentes medidas de similaridade e, no Capítulo 5, realizamos experimentos para descobrir a eficácia de cada medida em um conjunto de dados de bibliografia acadêmica. No Capítulo 3, modelamos o conjunto de dados como um grafo, construído a partir dos metadados obtidos da biblioteca digital da *Association for Computer Machinery* (*ACM*) [21] fornecidos da empresa *Odysci* [8].

Com o objetivo de calcular a similaridade entre artigos acadêmicos a partir do grafo de citação, desenvolvemos um protocolo experimental em duas partes: a primeira avaliava quão bem uma matriz de similaridade calculada, cujos elementos é o valor da similaridade entre dois artigos, segue uma ordenação parcial de similaridade de um gabarito calculado. Este gabarito que deriva da classificação hierárquica dos artigos no sistema de classificação de 1998 da *ACM*[1][2] [16], que é realizada pelos próprios autores dos artigos no momento da publicação. Este gabarito cria uma noção de similaridade tal que dois artigos são mais similares na medida em que são classificados em classes similares, isto é, que estão em classes hierarquicamente próximas. Já a segunda parte foi dedicada à avaliação dos algoritmos em uma tarefa de busca, *Query by Example*, avaliando se um ranqueamento dos resultados pela similaridade retornavam artigos relevantes ou não dado um outro artigo de busca.

Dada a complexidade de tempo e, principalmente, de espaço dos algoritmos de similaridade selecionados, tornou-se inviável a execução em todo grafo de citação. Desta forma, abordamos o problema de escala reduzindo o tamanho do grafo, com técnicas de

[1] http://www.acm.org/about/class/how-to-use – Acessado em Julho de 2012.
[2] http://www.acm.org/about/class/1998 – Acessado em 1998

amostragem. Assim, na Seção 3.2, estudamos diferentes formas de se criar amostras de um grafo que mantêm as propriedades do grafo original.

Ao estudar as técnicas de *Link Analysis*, entendemos que para o cálculo de similaridade entre objetos a partir da análise de um grafo em que se organizam, há algoritmos que se baseiam em diferentes estratégias: i) na análise dos caminhos formados entre objetos (*Katz* e *Weighted Paths*); ii) na análise da vizinhança local (*Common Neighbors*, *Adamic/Adar*, *Jaccard* e *Preferential Attachment*); iii) passeios aleatórios (*SimRank*, *Hitting Time*, *Commute Time*, *Rooted PageRank* e *PropFlow*); iv) outros métodos, como exemplo aqueles que se baseiam na fatoração da matriz de adjacência (*Low-rank approximation*). Selecionamos algumas destas medidas de similaridade de diferentes estratégias e observamos a partir de experimentos que aquelas que analisam a vizinhança local, além da simplicidade de implementação, obtêm resultados excelentes em termos de eficácia e eficiência. Neste sentido, destacam-se as medidas *Jaccard* e *Adamic/Adar* que em ambos experimentos (avaliação da ordenação de similaridade e o de busca) conseguiram obter a melhor eficácia dentre as medidas de similaridade selecionadas.

Dada a natureza direcionada do grafo de citação, percebemos que poderíamos reescrever estas medidas de similaridade para considerar a direção das arestas. Com isto, percebemos que a direção das arestas influencia bastante nos resultados de eficácia. Por exemplo, na avaliação da ordenação de similaridade, a eficácia aumentou consideravelmente se usássemos apenas a vizinhança dos predecessores dos vértices. Já na avaliação de busca por artigos similares, as medidas de similaridade configuradas para considerar apenas a vizinhança dos sucessores conseguiram os melhores resultados.

Nosso protocolo experimental (Capítulo 5), inspirado nas ideias de Haveliawala et al. [32], permite a avaliação da eficácia dos métodos objetivamente e de forma reproduzível, isto é, sem a necessidades de atividades manuais para julgar a similaridade entre artigos (no nosso ponto de vista é laboriosa e subjetiva). De fato, produzimos um gabarito utilizando dados que derivam de uma anotação humana (a classificação em categorias hierárquicas [16]), mas que é uma atividade obrigatória para a publicação e organização de um artigo na biblioteca digital da *ACM* [16, 21]. Isto é, com o tempo, nosso gabarito continuará sendo enriquecido sem nossa intervenção manual, fornecendo um ambiente de experimentação, que por construção, irá evoluir.

Ao lidar com o volume de dados, percebemos que várias técnicas não escalavam, pois tinham complexidade $O\left(|V|^2\right)$ de espaço e pelo menos $O\left(|V|^3\right)$ de tempo, em que V é o conjunto de vértices do grafo de citação (lembrando que $|V| = 122.774$). Há várias estratégias para resolver este desafio: processamento distribuído (como feito em [5] usando o MapReduce [19]), otimização de algoritmos (realizado em [49]), algoritmos de aproximação e probabilísticos (como em [23–25, 43]). Contudo, tomamos uma via não tomada antes para este problema, pelo menos ao que sabemos. Decidimos reescalar o conjunto de

dados e manter os algoritmos como foram propostos. Assim, no Capítulo 3.2 avaliamos várias técnicas de amostrar o grafo de citação, verificando se as amostras conservavam propriedades do grafo original. Percebemos que o método *ForestFire* [41] foi o que mais se destacou em termos de eficácia comparado a outros métodos da literatura. Ainda, derivamos um novo algoritmo do *ForestFire*, denominado *RankedForestFire*, que cria um viés na amostragem de acordo com um ranqueamento de artigos. Este novo algoritmo proposto obteve melhor eficácia que o *ForestFire* e, dado o tipo de viés que cria, acreditamos que seja ideal na amostragem de grafos de bibliotecas digitais e do grafo da *Web*, aquele formado por hiperlinks entre páginas.

Além desta contribuição, propomos uma melhoria para as técnicas existentes do cálculo de similaridade. Assim, no Seção 4, propomos uma nova meta-função de similaridade que transforma medidas locais, como *Jaccard* e *Adamic/Adar*, em medidas recursivas, cuja similaridade depende recursivamente da similaridade de outros artigos relacionados, explorando a ideia de que dois artigos são mais similares na medida em que estão associados a artigos que também são similares. Esta ideia foi inspirada nas justificativas do desenvolvimento do *SimRank* [36]. Nosso novo algoritmo foi aplicado para duas medidas de similaridade existes, o *Adamic/Adar* e *Jaccard*, assim produzimos outras duas medidas de similaridade: *RecursiveAdamicAdar* e *RecursiveJaccard*. Nos experimentos conduzidos no Capítulo 5, a primeira conseguiu a melhor eficácia que todos métodos avaliados nos dois experimentos que executamos. Já a segunda, *RecursiveJaccard*, conseguiu eficácia superior a sua correspondente não recursiva, *Jaccard*, para o experimento de ordenação de similaridade, contudo, no experimento de busca, não superou o *Jaccard*, obtendo resultado similar. Este resultado continuou consistente ao avaliar estas medidas para diferentes configurações de vizinhança dos vértices, isto é, levando-se em conta a direção das arestas do grafo de citação. Portanto, a medida de similaridade mais eficaz avaliada foi a nossa proposta *RecursiveAdamicAdar*. Este método recursivo é implementado por um algoritmo iterativo e que se mostrou convergir para o nosso conjunto de dados.

Ainda, dada a atenção que a literatura na área dá ao método *SimRank*, ficamos surpresos com a baixa eficácia deste método em nossos experimentos. Este conseguiu eficácia comparável aos métodos recursivos apenas no experimento de busca, *Query by Example*, quando configurada para desconsiderar a direção das arestas de citação. Todavia, no outro experimento de ordenação de similaridade, o *SimRank* obteve a pior eficácia dentre os métodos avaliados.

Várias ideias e hipóteses foram levantadas durante o desenvolvimento deste trabalho, contudo, estavam à margem do escopo deste mestrado. Entretanto, na Seção 6.2 a seguir, levantamos as principais ideias e hipóteses para que possam ser exploradas em trabalhos futuros. Em destaque, algumas ideias podem contribuir ainda mais na eficácia no cálculo de similaridade como por exemplo: i) considerar pesos na arestas do grafo de citação,

ponderando a importância de uma aresta de acordo com sua idade; ii) a combinação de várias medidas de similaridade existentes para compor outra talvez melhor.

Por fim, acreditamos que alcançamos todos objetivos levantados na Seção 1.5, principalmente porque usamos dados bem estruturados e de qualidade, desenvolvemos um protocolo de experimentação objetivo, reescalamos os grafos de citação com técnicas de amostragem e propomos novos algoritmos de similaridade que melhoram a eficácia dos métodos existentes selecionados.

6.2 Trabalhos futuros

Muitos assuntos relacionados e igualmente interessantes/promissores, todavia, além do que se propôs com este trabalho foram estudados. Destacamos aqui o que julgamos importante para serem tratados em trabalhos futuros.

6.2.1 Aplicações

Diversas aplicações que poderiam usar a similaridade entre entidades organizadas em um grafo e se beneficiar das técnicas pesquisadas nesta dissertação:

- **Agrupamento e Classificação de artigos:** Nesta atividade, o conjunto de dados explorado neste trabalho e o gabarito produzido contêm informações suficientes para o desenvolvimento e avaliação experimental de algoritmos de agrupamento (*clustering*). Por exemplo, um algoritmo poderia explorar as arestas de autoria e de publicação de um artigo, ver Capítulo *3,* para construir um vetor de característica e então aplicar técnicas existentes de agrupamento, como *K-means* [34]. Não é incomum que um mesmo autor publique artigos de assuntos relacionados. Ainda, percebe-se que conferências em geral estão relacionadas a temas de pesquisa específicos, isto é, artigos publicados em uma mesma conferência podem tratar de assuntos relacionados. Portanto, explorar estas arestas pode ser promissor para algoritmos de agrupamento de artigos, quando quer se agrupar artigos de temas relacionados. Note o potencial deste agrupamento, que possivelmente irá explorar um relacionamento semântico entre os artigos, possivelmente mais interessante que a similaridade apenas léxica de algoritmos que se baseiam do conteúdo textual destes artigos. No caso de classificação de artigos, além destas características que também podem ser usadas, existe um gabarito para avaliação experimental, isto é, o sistema de classificação da ACM de 1998 [16].

- **Agrupamento e Classificação de autores:** Análogo ao caso da aplicação para artigos, podem-se clusterizar ou classificar autores. Observar arestas de autoria,

tanto no caminho entre autores que determina uma coautoria, quanto correferência e co-citação. Coautoria é um caminho no grafo da biblioteca digital entre dois autores quando estes publicaram um artigo em comum, isto é derivado da aresta de autoria. Correferência é quando dois autores foram referenciados em um mesmo artigo, isto é, deriva-se esta informação das arestas de citação. Co-citação é o caso análogo, quando dois autores citam um mesmo artigo. Estas relações, coautoria, correferência e co-citação, mostram a relação entre autores usando como "*pivot*" um artigo. Podem-se criar outras possíveis relações trocando o objeto "*pivot*", como usar uma conferência, isto é, explorando o quão relacionados estão dois autores de acordo com o padrão de onde publicam seus artigos. Mais uma vez, estas relações extraídas do grafo da biblioteca digital podem ser consideradas para computar vetores de características e serem usados em atividades de agrupamento e classificação de autores. Estas relações citadas podem, inclusive, produzir matrizes que cruzam Atores x Artigos e Autores x Conferências e técnicas de fatoração de matrizes, como *Latent Semantic Analysis* (*LSA*) [22], poderiam ser abordadas. Um possível gabarito para a terefa de classificação seria a informação de afiliação dos autores que, infelizmente, não está contida em nosso conjunto de dados.

- **Predição de coautoria e citação:** Nosso conjunto de dados possuem informações da citação de artigos e a informação da data de publicação dos artigos. Por isso, contém informação suficiente para avaliar técnicas que tentam predizer se dois autores irão trabalhar juntos no futuro, isto é, se serão coautores, ou então, tentar prever quantas citações um artigo receberia no futuro. O primeiro caso foi explorado por Liben-Nowell et al. [45]. O segundo, de prever o número futuro de citações de um artigo, poderia ser explorado como uma série temporal, isto é, observando quantas citações um artigo recebe ao longo do tempo. Para explorar as contribuições de Link Analysis neste caso, não é incomum perceber que citações ocorrem entre artigos relacionados, entre grupos de autores relacionados ou entre artigos publicados em conferências relacionadas, portanto, poderiamos reusar a similaridade calculada entre estas entidades para aumentar a precisão das predições. Recomenda-se ao leitor interessado, os artigos [66, 74] relacionados a este tema.

- **Recomendação de artigos:** Uma possível aplicação do cálculo de similaridade entre artigos é a recomendação de artigos relacionados. Se um usuário está observando um artigo em uma biblioteca digital, talvez queira ver mais artigos relacionados. Este tipo de abordagem, explorado em nosso experimento de *Query by Example*, não explora os interesses que um usuário pode ter *a priori*, não diferenciando dois autores que estão observando o mesmo artigo. Nos trabalhos de Sugiyama et al. [64,65], apresentam-se técnicas que exploram a construção de um perfil de um

usuário e, ainda, disponibilizam abertamente um conjunto de dados anotados para outros pesquisadores[3]. Ainda, recomenda-se ao leitor a leitura de [26] que faz aplicação de técnicas de similaridade de *Link Analysis*, contudo, usando um conjunto de dados que relaciona pessoas e filmes, organizados em um grafo bipartido. Supõe-se que o mesmo poderia ser aplicado para o subgrafo induzido da biblioteca digital que relaciona artigos e autores, também bipartido.

6.2.2 Amostragem na Web

Na nossa proposta de um método de amostragem do grafo de citação, o *RankedForestFire*, mencionamos que seria apropriado sua aplicação no grafo da Web, aquele formado por arestas de *hyperlinks* entre páginas. Assim, fica uma proposta para trabalho futuro avaliar esta hipótese e, sugere-se explorar os projetos *Snap* (*Stanford Network Analysis*)[4] e *WebBase* (*The Stanford WebBase Project*) [15], ambos provêm acesso a um grafo *Web*[5][6].

6.2.3 Escalabilidade

Um sério problema enfrentado nos métodos de *Link Analysis* é escalar bem em volumes grandes de dados. Muitos dos métodos, no cálculo de similaridade ou outras medidas de relacionamento em um grafo $G(V, E)$, sofrem de problemas de escalabilidade, pois são $O\left(|V|^3\right)$ em tempo e $O\left(|V|^2\right)$ de espaço [5, 36, 71, 72]. Para lidar com a questão, observa-se na literatura a utilização de algoritmos probabilísticos e aproximados, como em [23, 24, 43]. Outros fazem redução de dimensionalidade dos dados, isto é, a redução do posto da matriz de adjacência, como [43, 45]. Há menção de que os algoritmo são paralelizados no esquema de processamento distribuído *MapReduce* [19], conforme mencionado em [5] (mas não mostrado o algoritmo). Neste caso, falta na literatura o desenho de algoritmos de *Link Analysis* nesse *framework* de *MapReduce*, sendo uma lacuna interessante a se preencher. Quanto à complexidade de espaço, há dois problemas: o da representação do grafo de relações entre os objetos, normalmente esparso, e a representação da matriz de similaridade entre os objetos. Quanto à representação de grafos, foram observados estudos relevantes na literatura. Muitos confirmam que grafos do mundo real, que representam interações sociais, não só são extremamente esparsos como apresentam uma distribuição "*power-law*" nos graus de saída dos nós [54]. Barabási [7] cria um modelo teórico baseado na intuição de "*preferential attachment*" para explicar essa distribuição.

[3] http://www.comp.nus.edu.sg/~sugiyama/SchPaperRecData.html – Acessado em Julho de 2012.

[4] http://snap.stanford.edu/index.html – Acessado em Julho de 2012.

[5] http://dbpubs.stanford.edu:8091/~testbed/doc2/WebBase/webbase-pages.html– Acessado em Julho de 2012.

[6] http://snap.stanford.edu/data/web-Google.html– Acessado em Julho de 2012.

Algumas técnicas de representação de grafos foram desenvolvidas de forma a capturar tais propriedades, como no projeto *WebGraph* [10]. Esta pesquisa, na linha de teoria da informação, cria um código instantâneo para compactação de grafos que tira proveito de uma distribuição empírica observada dos "*gaps*" entre nós em uma lista de adjacência. Aplicam o esquema em grafos reais da WEB e, em seus *benchmarks*, gastam de 2 a 4 bits apenas para representar cada aresta. Isto é um grande feito, ainda mais porque o acesso é tanto sequencial quanto randômico. O *WebGraph* oferece abertamente a biblioteca com código fonte aberto, inclusive, com vários algoritmos implementados para o processamento, como por exemplo o *PageRank* que é de grande relevância para este mestrado.

No nosso trabalho, abordamos esta questão realizando amostragem no grafo de citação, contudo, trabalhos futuros poderiam focar na implementação de algoritmos distribuídos de *Link Analysis*, representações compactas e eficientes do grafo de citação e a otimização de algoritmos existentes, bem como algoritmos de aproximação e probabilísticos.

6.2.4 Similaridade

Aqui sugerimos trabalhos futuros relacionados a melhoria da eficácia das medidas de similaridades.

6.2.4.1 Composição

Estudar o quanto uma medida de similaridade está correlacionada com outra, como feito em [45]. Disto podemos entender se estratégias relacionadas de medidas de similaridade produzem ou não algoritmos cujos resultados estão correlacionados. Por exemplo, estariam correlacionados todos algoritmos que consideram apenas a vizinhança local dos vértices? Estariam correlacionados aqueles baseados em caminhos do grafo? E os baseados em *Random Walk*? O que se pode dizer da correlação dos algoritmos que têm estrategias distintas?

Estas informações podem permitir a especificação de métodos combinados, isto é, permitir combinar duas ou mais medidas de similaridade para produzir uma outra mais eficaz. Uma possibilidade seria usar algoritmos de aprendizado de máquina para encontrar funções que combinam as medidas de similaridade existentes.

6.2.4.2 Outros dados no grafo

Outra possibilidade para melhorar a eficácia dos métodos de similaridade seria explorar outras informações contidas no grafo que representa a biblioteca digital.

Por exemplo, usar pesos nas arestas do grafo: Uma possibilidade seria considerar tanto a diferença temporal entre vértices dado uma aresta, quanto a idade da aresta. Isto é,

quando um autor publica um artigo ou um artigo é citado, sabemos quando isto ocorreu, pelo ano de publicação de um artigo. Ou seja, sabemos a idade de uma aresta. Ainda, no caso da citação, podemos extrair a diferença de anos de publicação entre os dois artigos nesta aresta. Ambas informações poderiam ser usadas para criar pesos nas arestas para informar o quão relevante ela é atualmente. Poderíamos levantar a hipótese que arestas antigas não são tão importantes quanto as mais jovens ao se calcular a similaridade entre os vértices. Outra possibilidade consiste em considerar que arestas de artigos jovens para artigos mais antigos revelam importantes arestas, pois, supõe-se que revelariam a base de um estudo. Para ambas hipóteses, poderíamos definir funções de pesos para aresta que consideram estas propriedades temporais e observar se isto influenciaria a eficácia dos métodos de similaridade.

Outra possibilidade seria julgar que certas aresta são mais importantes que outras a depender da semântica que têm. Por exemplo, de acordo com a Figura 3.3, seriam arestas de citação mais importantes que arestas de autoria ou de publicação? Sendo assim, podem-se explorar pesos entre arestas que dependem do tipo da relação e avaliar os algoritmos de similaridade considerando todo grafo da biblioteca digital. Note que neste trabalho foi considerado apenas arestas de citação.

Neste sentido de considerar todas arestas do grafo da biblioteca digital, mesmo que de semântica distinta, um recente estudo de Sun et al. [66] exploram a ideia de Meta-Caminhos, isto é, alguns caminhos do grafo colapsado, Figura 3.3, podem produzir relações importantes para se considerar em algoritmos de *Link Analysis*. Por exemplo, o Meta-Caminho Autor–Artigo–Autor estabelece uma relação de coautoria entre dois autores, já o Meta-Caminho Autor–Artigo–Venue–Artigo–Autor revelam as conferências/periódicos que dois autores publicaram em comum. Estes autores propõem técnicas para converter estes Meta-Caminhos entre entidades em características e aplicam na tarefa de predizer *links* ou interações entre estas no futuro. Acreditamos que este tipo de abordagem pode contribuir para o cálculo de similaridade entre artigos, pois explora de forma genérica, várias possibilidades de relações existentes no grafo de uma biblioteca digital.

6.2.5 Complexidade

Fica como proposta de trabalho futuro o entendimento da complexidade de tempo e espaço para algoritmos de *Link Analysis* aplicados para grafos reais, em especial, o de citação. Apesar de se saber a complexidade teórica de vários algoritmos, algumas propriedades particulares do grafo de citação pode permitir encontrar funções assintóticas mais justas. No caso de citação, trata-se de um grafo muito esparso, livre de escala e com uma distribuição "*Power-law*" de arestas de entrada [3]. Estas propriedades devem influenciar bastante na análise de complexidade destes algoritmos. Em especial, o estudo mais

detalhado destas características deve facilitar o cálculo da complexidade dos algoritmos propostos *RecursiveAdamicAdar* e *RecursiveJaccard*.

Referências Bibliográficas

[1] Lada A. Adamic and Eytan Adar. Friends and neighbors on the web. *Social Networks*, 25(3):211 – 230, 2003.

[2] A. V. Aho, J. E. Hopcroft, and J. D. Ullman. On finding lowest common ancestors in trees. In *Proceedings of the fifth annual ACM symposium on Theory of computing*, STOC '73, pages 253–265, New York, NY, USA, 1973. ACM.

[3] Y. An, J. C. M. Janssen, and E. E. Milios. Characterizing and mining the citation graph of the computer science literature. *Knowledge and Information Systems*, 6(6):664–678, 2004. http://www.odysci.com/article/1010112991413578.

[4] Rohit Ananthakrishna, Surajit Chaudhuri, and Venkatesh Ganti. Eliminating fuzzy duplicates in data warehouses. In *VLDB '02: Proceedings of the 28th international conference on Very Large Data Bases*, pages 586–597. VLDB Endowment, 2002.

[5] Ioannis Antonellis, Hector Garcia-Molina, and Chi-Chao Chang. Simrank++: query rewriting through link analysis of the click graph. *Proceedings of the VLDB Endowment*, 1(1):408–421, 2008.

[6] Ricardo A Baeza-Yates and Berthier A Ribeiro-Neto. *Modern Information Retrieval*. ACM Press / Addison-Wesley, 1999.

[7] Albert-László Barabási and Réka Albert. Emergence of Scaling in Random Networks. *Science Magazine*, 286(5439):509–512, 1999.

[8] Reinaldo Alvarenga Bergamaschi. Announcement: introducing odysci. *ACM SIGACCESS Accessibility and Computing*, (101):20–21, 2011.

[9] Indrajit Bhattacharya and Lise Getoor. A latent dirichlet model for unsupervised entity resolution. In *Sixth SIAM International Conference on Data Mining*, pages 47–58, 2006. http://www.odysci.com/article/1010112988295835.

[10] P Boldi and S Vigna. The webgraph framework I: compression techniques. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 595–602, New York, NY, USA, 2004. ACM.

[11] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, 1998.

[12] Pável Calado, Marco Cristo, Marcos André Gonçalves, Edleno Moura, Berthier Ribeiro-Neto, and Nivio Ziviani. Link-based similarity measures for the classification of web documents. *Journal of the American Society for Information Science and Technology*, 57(2):208–221, 2006. http://www.odysci.com/article/1010112991043026.

[13] Pável Calado, Marco Cristo, Edleno Moura, Nivio Ziviani, Berthier Ribeiro-Neto, and Marcos André Gonçalves. Combining link-based and content-based methods for web document classification. In *CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management*, pages 394–401, New York, NY, USA, 2003. ACM.

[14] Soumen Chakrabarti, Byron Dom, and Piotr Indyk. Enhanced hypertext categorization using hyperlinks. In *SIGMOD '98: Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, pages 307–318, New York, NY, USA, 1998. ACM.

[15] J. Cho and H. Garcia-Molina. Webbase and the stanford interlib project. In *Kyoto International Conference on Digital Libraries 2000*, pages 178–178, 2000. http://www.odysci.com/article/1010112987348875.

[16] N. S. Coulter. Acm's computing classification system reflects changing times. *Communications of the ACM*, 40(12):111–112, 1997.

[17] Thierson Couto, Marco Cristo, Marcos André Gonçalves, Pável Calado, Nivio Ziviani, Edleno Moura, and Berthier Ribeiro-Neto. A comparative study of citations and links in document classification. In *JCDL '06: Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries*, pages 75–84, New York, NY, USA, 2006. ACM.

[18] Thomas M. Cover and Joy A. Thomas. *Elements of information theory*. Wiley-Interscience, New York, NY, USA, 1991.

[19] Jeffrey Dean and Sanjay Ghemawat. MapReduce: simplified data processing on large clusters. *Communications ACM*, 51(1):107–113, 2008.

[20] Jeffrey Dean and Monika Rauch Henzinger. Finding related pages in the world wide web. *Computer Networks*, 31(11-16):1467–1479, 1999.

[21] Peter J. Denning. The acm digital library goes live. *Communications of the ACM*, 40(7):28–29, 1997.

[22] Susan T. Dumais. Latent semantic analysis. *Annual Review of Information Science and Technology*, 38(1):188–230, 2004.

[23] Dániel Fogaras and Balázs Rácz. A scalable randomized method to compute link-based similarity rank on the web graph. In *Current Trends in Database Technology - EDBT 2004 Workshops*, volume 3268 of *Lecture Notes in Computer Science*, pages 361–363. Springer Berlin / Heidelberg, 2005.

[24] Dániel Fogaras and Balázs Rácz. Scaling link-based similarity search. In *WWW '05: Proceedings of the 14th international conference on World Wide Web*, pages 641–650, New York, NY, USA, 2005. ACM.

[25] Dániel Fogaras and Balázs Rácz. Practical algorithms and lower bounds for similarity search in massive graphs. *Knowledge and Data Engineering, IEEE Transactions on*, 19(5):585 –598, may 2007.

[26] Francois Fouss, Alain Pirotte, Jean-Michel Renders, and Marco Saerens. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *Knowledge and Data Engineering, IEEE Transactions on*, 19(3):355 –369, march 2007.

[27] Francois Fouss, Luh Yen, Alain Pirotte, and M. Saerens. An experimental investigation of graph kernels on a collaborative recommendation task. In *6th IEEE International Conference on Data Mining (ICDM 2006)*, pages 863 –868, dec. 2006.

[28] Lise Getoor and Christopher P Diehl. Link mining: a survey. *ACM SIGKDD Explorations Newsletter*, 7(2):3–12, 2005.

[29] David Goldberg. What every computer scientist should know about floating-point arithmetic. *ACM Computing Surveys*, 23(1):5–48, mar 1991.

[30] Anna Goldenberg, Alice X. Zheng, Stephen E. Fienberg, and Edoardo M. Airoldi. A survey of statistical network models. *Foundations and Trends in Machine Learning*, pages 129–233, Dec 2009.

[31] Leo A. Goodman and William H. Kruskal. Collective dynamics of small-world networks. *Journal of the American Statistical Association*, 49(268):732–764, December 1954.

[32] Taher H Haveliwala, Aristides Gionis, Dan Klein, and Piotr Indyk. Evaluating strategies for similarity search on the web. In *WWW '02: Proceedings of the 11th international conference on World Wide Web*, pages 432–442, New York, NY, USA, 2002. ACM.

[33] Takahiko Ito, Masashi Shimbo, Taku Kudo, and Yuji Matsumoto. Application of kernels to link analysis. In *KDD '05: Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 586–592, New York, NY, USA, 2005. ACM.

[34] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[35] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems*, 20(4):422–446, 2002. http://www.odysci.com/article/1010112992295166.

[36] Glen Jeh and Jennifer Widom. SimRank: a measure of structural-context similarity. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 538–543, New York, NY, USA, 2002. ACM.

[37] Yushi Jing and S. Baluja. Visualrank: Applying pagerank to large-scale image search. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(11):1877 – 1890, nov. 2008.

[38] L. Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953.

[39] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

[40] V. Krishnamurthy, M. Faloutsos, M. Chrobak, L. Lao, J. Cui, and A. G. Percus. Reducing large internet topologies for faster simulations. In *NETWORKING 2005: Networking Technologies, Services*, volume 3462, pages 328–341, 2005. http://www.odysci.com/article/1010112987700831.

[41] J. Leskovec, J. M. Kleinberg, and C. Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. In *Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 177–187, 2005. http://www.odysci.com/article/1010112987289034.

[42] Jure Leskovec and Christos Faloutsos. Sampling from large graphs. In *Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 631–636, 2006. http://www.odysci.com/article/1010112987290170.

[43] Cuiping Li, Jiawei Han, Guoming He, Xin Jin, Yizhou Sun, and Yintao Yu. Fast computation of simrank for static and dynamic information networks. In *EDBT 2010, 13th International Conference on Extending Database Technology*, volume 426, pages 465–476, Lausanne, Switzerland, March 2010.

[44] Ming Li, Xin Chen, Xin Li, Bin Ma, and P.M.B. Vitanyi. The similarity metric. *IEEE Transactions on Information Theory*, 50(12):3250–3264, dec. 2004. http://www.odysci.com/article/1010112992190572.

[45] David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology*, 58(7):1019–1031, 2007.

[46] Ryan Lichtenwalter, Jake T. Lussier, and Nitesh V. Chawla. New perspectives and methods in link prediction. In *16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 243–252, 2010.

[47] Zhenjiang Lin, Irwin King, and Michael R Lyu. PageSim: A Novel Link-Based Similarity Measure for the World Wide Web. In *WI '06: Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 687–693, Washington, DC, USA, 2006. IEEE Computer Society.

[48] Zhenjiang Lin, Michael R Lyu, and Irwin King. MatchSim: a novel neighbor-based similarity measure with maximum neighborhood matching. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, pages 1613–1616, New York, NY, USA, 2009. ACM.

[49] Dmitry Lizorkin, Pavel Velikhov, Maxim Grinev, and Denis Turdakov. Accuracy estimate and optimization techniques for SimRank computation. *Proceedings of the VLDB Endowment*, 1(1):422–433, 2008.

[50] Ana Gabriela Maguitman, Filippo Menczer, Fulya Erdinc, Heather Roinestad, and Alessandro Vespignani. Algorithmic computation and approximation of semantic similarity. *World Wide Web*, 9(4):431–456, 2006.

[51] Benjamin Markines, Ciro Cattuto, Filippo Menczer, Dominik Benz, Andreas Hotho, and Gerd Stumme. Evaluating similarity measures for emergent semantics of social

tagging. In *WWW '09: Proceedings of the 18th international conference on World wide web*, pages 641–650, New York, NY, USA, 2009. ACM.

[52] Arvind Narayanan and Vitaly Shmatikov. De-anonymizing social networks. In *SP '09: Proceedings of the 2009 30th IEEE Symposium on Security and Privacy*, pages 173–187, Washington, DC, USA, 2009. IEEE Computer Society.

[53] M. E. J. Newman. Clustering and preferential attachment in growing networks. *Physical Review Letters E*, 64(2):25102, 2001.

[54] M. E. J. Newman. The Structure and Function of Complex Networks. *SIAM Review*, 45(2):167–256, 2003.

[55] M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*, 74(3):36104, 2006.

[56] S. Oyama, K. Hayashi, and H. Kashima. Cross-temporal link prediction. In *2011 IEEE 11th International Conference on Data Mining*, pages 1188–1193, 2011. http://www.odysci.com/article/1010113017133232.

[57] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.

[58] Daniel Ramage, Anna N. Rafferty, and Christopher D. Manning. Random walks for text semantic similarity. In *TextGraphs-4: Proceedings of the 2009 Workshop on Graph-based Methods for Natural Language Processing*, pages 23–31, Morristown, NJ, USA, 2009. Association for Computational Linguistics.

[59] Matthew Richardson and Pedro Domingos. The intelligent surfer: Probabilistic combination of link and content information in pagerank. In *NIPS*, pages 1441–1448, 2001.

[60] Gerard Salton, Edward A. Fox, and Harry Wu. Extended boolean information retrieval. *Communications of the ACM*, 26(11):1022–1036, November 1983. http://www.odysci.com/article/1010112982792987.

[61] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93–106, 2008.

[62] Nikolai Vasilyevich Smirnov. On the Estimation of the Discrepancy Between Empirical Curves of Distribution for Two Independent Samples. *Moscow University Mathematics Bulletin*, 2:3–14, 1939.

[63] James H. Steiger. Tests for comparing elements of a correlation matrix. *Psychological Bulletin*, 87(2):245–251, 1980.

[64] K. Sugiyama and M. Kan. Scholarly paper recommendation via user's recent research interests. In *2010 Joint International Conference on Digital Libraries, JCDL 2010*, pages 29–38, 2010.

[65] K. Sugiyama and M. Kan. Serendipitous recommendation for scholarly papers considering relations among researchers. In *Proceeding of the 11th annual international ACM/IEEE joint conference on Digital libraries*, pages 307–310, 2011. http://www.odysci.com/article/1010113015005944.

[66] Y. Sun, J. Han, C. C. Aggarwal, and N. V. Chawla. When will it happen?: relationship prediction in heterogeneous information networks. In *Fifth ACM International Conference on Web Search and Data Mining*, pages 663–672, 2012.

[67] Y. Sure, S. Bloehdorn, P. Haase, J. Hartmann, and D. Oberle. The swrc ontology - semantic web for research communities. In *Progress in Artificial Intelligence, 12th Portuguese Conference on Artificial Intelligence, EPIA 2005*, volume 3808, pages 218–231, 2005. http://www.odysci.com/article/1010112984452944.

[68] Mike Thelwall. *Link Analysis: An Information Science Approach*. Academic Press, first edition, January 2005.

[69] Xuanhui Wang, Jian-Tao Sun, and Zheng Chen. Shine: search heterogeneous interrelated entities. In *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 583–592, New York, NY, USA, 2007. ACM.

[70] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of small-world networks. *Nature*, 393(6684):440–442, June 1998.

[71] Scott White and Padhraic Smyth. Algorithms for estimating relative importance in networks. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 266–275, New York, NY, USA, 2003. ACM.

[72] Wensi Xi, Edward A Fox, Weiguo Fan, Benyu Zhang, Zheng Chen, Jun Yan, and Dong Zhuang. SimFusion: measuring similarity using unified relationship matrix. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 130–137, New York, NY, USA, 2005. ACM.

[73] Wensi Xi, Benyu Zhang, Zheng Chen, Yizhou Lu, Shuicheng Yan, Wei-Ying Ma, and Edward Allan Fox. Link fusion: a unified link analysis framework for multi-type interrelated data objects. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 319–327, New York, NY, USA, 2004. ACM.

[74] D. Yin, L. Hong, and B. D. Davison. Structural link analysis and prediction in microblogs. In *20th ACM international conference on Information and knowledge management*, pages 1163–1168, 2011.

[75] Xiaoxin Yin, Jiawei Han, and Philip S Yu. LinkClus: efficient clustering via heterogeneous semantic links. In *VLDB '06: Proceedings of the 32nd international conference on Very large data bases*, pages 427–438. VLDB Endowment, 2006.

[76] Peixiang Zhao, Jiawei Han, and Yizhou Sun. P-Rank: a comprehensive structural similarity measure over information networks. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, pages 553–562, New York, NY, USA, 2009. ACM.

[77] Justin Zobel and Alistair Moffat. Exploring the similarity space. *ACM SIGIR Forum*, 32(1):18–34, 1998. http://www.odysci.com/article/1010112991881939.