David José Machado Ferreira

Procura Estruturada de Textos para Perfis de Utilizadores

Universidade da Beira Interior

Departamento de Informática

Agosto 2009

David José Machado Ferreira

Procura Estruturada de Textos para Perfis de Utilizadores

Tese submetida à Universidade da Beira Interior para o preenchimento dos requisitos para a concessão do grau de Mestre efectuada sob a supervisão do Doutor Gaël Harry Adélio André Dias, Universidade da Beira Interior, Covilhã, Portugal

Universidade da Beira Interior
Departamento de Informática
Agosto 2009

Agradecimentos

Primeiramente, começo por agradecer ao meu orientador, Gaël Dias, pelo apoio e ajuda incontestáveis. Foi um prazer realizar o trabalho desta tese na sua companhia, onde me foi possibilitado atingir um novo nível de experiência com toda a discussão de ideias e explicações dadas sobre assuntos relacionados com o trabalho da tese mas, também, sobre todo um outro conjunto variadíssimo e interessante de temas.

O meu obrigado também aos membros do HULTIG, em especial ao Tiago que me ajudou e fez companhia pelo laboratório durante este longo período.

Agradeço também a todos os meus amigos que sempre me ouviram e apoiaram nos momentos mais complicados durante todos estes anos.

Não menos importante, quero deixar aqui um agradecimento especial à minha família pelo apoio incondicional que sempre me deu. O meu muito obrigado, pois sem vocês não conseguiria e por isso vos dedico este projecto.

Dizem que os últimos são sempre os primeiros, eu concordo, e por isso agradeço finalmente à minha russa de olhinhos azuis. Esta linda namorada que me faz muito feliz, acompanhou-me sempre em todos os momentos, dando-me o ânimo e a força necessária para continuar com o trabalho mesmo quando passava por fases menos boas da vida.

O meu muito obrigado!

Resumo

A dimensão cada vez mais colossal e a falta de estruturação da informação existente na Web, está a fazer com que os sistemas de Information Retrieval enfrentem graves dificuldades no cumprimento dos objectivos para os quais foram criados. Torna-se cada vez mais difícil para estes sistemas encontrarem um conjunto limitado e valioso de informação procurada pelo utilizador. A consciência deste facto ao longo dos anos tem proporcionado um aumento da comunidade de investigadores que se debatem na exploração de diversos temas e conceitos capazes de solucionarem os problemas.

Das diversas propostas, as mais promissoras são o clustering dos resultados e a personalização. O clustering permite que a informação devolvida seja organizada em categorias, dando ao utilizador a possibilidade de restringir a sua área de procura com base na escolha das categorias mais apelativas. Por sua vez, a personalização procura escolher automaticamente os resultados que são mais próximos do perfil do utilizador, efectuando uma reordenação da informação de acordo com os interesses de cada um.

A criação de perfis passa geralmente pela obtenção das categorias de interesse através de uma análise a todo o conteúdo dos documentos lidos e classificados como relevantes para o utilizador. Este tipo de metodologia leva a que seja necessário analisar um grande conjunto de informação. Daqui surgem dois problemas. O tempo de computação e o armazenamento dos dados necessários. Se por um lado é possível carregar o documento da Web sem que seja necessário o seu armazenamento, o tempo dispendido nesta operação para cada documento leva a que o tempo total de processamento seja excessivamente elevado. Por outro lado ao guardar os documentos, o tempo de computação é deveras reduzido mas faz com que para um conjunto alargado de utilizadores, o conjunto de informação a armazenar se torne muito grande. Certas metodologias ultrapassam este problema ao proporem modelos pré-definidos, contudo este tipo de abordagem é sempre mais limitado pois as possibilidades

dadas nem sempre contemplam todas as particularidades de cada utilizador.

O trabalho desta tese propõe uma nova abordagem à criação dos perfis para tentar ultrapassar estes problemas. Criar os perfis dos utilizadores a partir da análise do histórico das pesquisas efectuadas pelos mesmos num motor de pesquisa capaz de efectuar categorização dos resultados. Fazer uso das categorias associadas a cada pesquisa para extrair conhecimento oculto e auxiliar à criação dos perfis. Ou seja, em vez de analisar todos os documentos para extrair as categorias que mais se sobressaem para um utilizador em questão, o sistema analisa a estrutura das queries bem como o conjunto de categorias que estão associados. Esta metodologia reduz muito o conjunto de dados e torna a computação bastante mais rápida. Além do mais, ao não fazer uso de categorias pré-definidas, é possível criar um modelo bastante específico para cada utilizador. Tendo como objectivo a construção de um sistema completamente autónomo e independente, é também proposta uma nova metodologia para efectuar a categorização de Web snippets baseada no cálculo do valor de importância das palavras e como tal, sem fazer uso de listas de palavras vazias. Com a categorização associada à criação de perfis de utilizadores, torna-se possível para um sistema identificar o conteúdo que está mais associado aos interesses de cada pessoa, potencializando-se assim a interactividade entre o sistema e o utilizador.

Conteúdo

Lista de Figuras

Lista de Tabelas

Acrónimos

IR Information Retrieval

STC Suffix Tree Clustering

SVM Support Vector Machine

Web World Wide Web

FPF Furthest Point First

DHC Divisive Hierarchical Clustering

ODP Open Directory Project

LLSF Linear Least Squares Fit

CBL Clustering by Labels

Capítulo 1

Introdução

Dwight D. Eisenhower quando iniciou o projecto ARPA e a criação da rede ARPANET com certeza não imaginou a revolução a que estaria a dar início. Os protocolos e conhecimentos extraídos deste projecto deram lugar ao que hoje é conhecido por internet, a rede que mudou o mundo. Inicialmente, esta rede mapeava um único directório e praticamente todos os servidores eram conhecidos mas, expandiu-se num instante e sendo o conteúdo editado manualmente tornou-se óbvia a dificuldade em manter uma hierarquia desta estrutura. Como resultado, foi necessário adoptar técnicas de recuperação de informação desenvolvidas para outras áreas e adaptá-las à Web.

Os primeiros sistemas de pesquisa que surgiram não mantinham informação relativa ao conteúdo das páginas Web, apenas indexavam os títulos. Em 1994 deu-se mais um passo. Passou a indexar-se todo o conteúdo dos documentos, incluindo os títulos, de modo a que o utilizador pudesse procurar dentro do conteúdo das páginas e não apenas nos títulos. O mercado expandiu-se, tornando empresas como a Yahoo [20], Altavista [14] referências no sector. Em 1998, aconteceu aquilo que Bill Gates mais temia [1] o Google [19] apareceu e veio mudar tudo. Era um motor de pesquisa criado no anonimato e revolucionário, capaz de proporcionar resultados bem melhores que os obtidos pelos motores de busca existentes. Este novo sistema considerava a estrutura de hiperligações ente os documentos na Web e não apenas o seu conteúdo. O algoritmo denominado de PageRank, introduziu o conceito de citação na Web: quanto mais citações um documento da Web tenha, maior importância lhe é dada. Além disso, quanto maior importância tiver a fonte citadora, mais importante se torna este documento alvo.

Nos tempos correntes os motores de pesquisa estão massificados e o seu uso e importância não pode ser negado. Estudos no mercado dos Estados Unidos [13] revelam terem sido efectuadas 9.4 biliões pesquisas nos principais motores de busca apenas no mês de Maio 2009 e mostram existir uma crescente afluência por parte dos utilizadores aos motores de pesquisa a cada ano que passa na ordem dos 20%.

1.1 Problemática dos motores de pesquisa na Web

A dimensão cada vez mais colossal e a falta de estruturação da informação está a fazer com que os sistemas de recuperação de informação (Information Retrieval IR) enfrentem graves dificuldades no cumprimento dos objectivos para os quais foram criados. A consciência deste facto ao longo dos anos tem proporcionado um aumento da comunidade de investigadores que se debatem na exploração de diversos temas e conceitos capazes de solucionarem os problemas.

O objectivo puro de um motor de pesquisa é devolver os documentos considerados relevantes para uma sequência de palavras-chave (query) fornecida por um utilizador. Por documento, considera-se qualquer tipo de ficheiro que possa ser particionado num conjunto de palavras ou expressões regulares sendo tipicamente uma página Web, um PDF, um DOC, ou um ficheiro Postscript. A query geralmente é representada por um conjunto de palavras pertencentes a uma língua e com uma forte relação com a informação pretendida. Os motores de busca podem classificar-se como sendo globais, quando devolvem resultados provenientes de uma escolha face a um conjunto muito disperso e numeroso de documentos existentes na internet. Acabam por funcionar como portais para qualquer página Web. Caso estejam indexados num domínio específico como um site, são denominados por locais visto que a sua área de informação é restrita.

Ponderando um pouco sobre os biliões de páginas existentes acessíveis na rede e a complexidade da linguagem, é fácil perceber que os problemas com que se deparam os sistemas de WebIR não são poucos nem triviais. Algoritmos eficientes de crawling, capacidade de processamento distribuído, capacidade de indexação distribuída, adaptação ao ritmo alucinante de novos protocolos Web e conteúdos multimédia, algoritmos de filtragem

linguísticos, algoritmos de reconhecimento de spam, são apenas alguns exemplos do quanto é exigente e complexo montar um sistema de recuperação de informação.

Parte de todos estes componentes serve para resolver problemas de outra ordem. Os próximos cinco parágrafos apresentam os principais desafios, segundo Gulli [45], com que se deparam os sistemas de WebIR.

O primeiro, está na própria definição da palavra ”relevante”. Por muito tempo se estudaram diversas abordagens e muito esforço foi dispendido na criação de algoritmos capazes de providenciar uma selecção de documentos da Web mais relevantes para um determinado conjunto de palavras-chave. A mais potente metodologia foi introduzida com a adopção do conceito de redes sociais e análise de hiperligações como o caso dos algoritmos PageRank e Hits. Contudo, em muitas situações esta não é a solução mais perfeita, pois não consegue ultrapassar certos problemas inerentes à complexidade da estrutura da Web. Para queries específicas, onde o número de documentos relacionados existentes na rede é reduzido, torna-se um problema encontrá-los. Para queries mais genéricas, que apontam para diversos assuntos, acabam por existir milhões de documentos relevantes e torna-se extremamente complexo saber qual a melhor ordenação mesmo recorrendo a estes algoritmos. Um dos outros aspectos relevantes para o processo de ranking é que este se baseia numa metodologia. Tal efeito leva a que muitas entidades (empresas cujo comércio depende muito da visibilidade das suas páginas num motor de busca) explorem o comportamento dos algoritmos de modo a encontrarem a técnica que lhes possibilite um melhor ranking das suas páginas. Este tipo de comportamento é conhecido por optimização para os motores de busca e é legal. Infelizmente, existe quem se aproveite destas técnicas para dar ênfase a páginas cujo conteúdo não é digno de interesse mas que, dada a sua estrutura, aos olhos dos motores de pesquisa podem ser classificadas como relevantes, deteriorando os resultados da ordenação.

O segundo desafio resulta do já referido tamanho da informação disponível na internet. A qualidade de um motor de pesquisa depende em muito da completude e frescura do seu índice de páginas que deve conter poucos documentos desactualizados e hiperligações quebradas. Infelizmente, a explosão de informação digital existente na rede está a tornar esta tarefa impraticável. Se por um lado começa a ser impossível indexar todas as páginas, por outro, o número de documentos relevantes devolvidos pelo sistema para uma pesquisa também aumenta (milhões de páginas relevantes não é um bom indicador de eficiência),

visto ser proporcional ao tamanho da rede. O problema da indexação de todos os documentos, pode ser resolvido com a recorrência a um meta-motor de pesquisa [25] que explora um conjunto de resultados provenientes de múltiplos motores e como tal a área de cobertura da rede também aumenta. Contudo, isto leva a um outro problema que é a reordenação por grau de interesse de todo o conjunto de documentos e o tamanho ainda maior deste.

A terceira dificuldade resulta do facto da relevância de um documento ser subjectivo e dependente do contexto pretendido. O mesmo conjunto de palavras-chave pode representar interesses diferentes conforme os interesses de cada utilizador e pode até, ser dependente do tempo. Imagine-se o seguinte caso: dois utilizadores introduzem a palavra programação. Porém um deles é completamente alheio ao termo e procura simplesmente a definição. Um outro utilizador, já mais familiarizado pode pretender conhecer linguagens de programação e até num outro momento do espaço temporal simplesmente querer dicas sobre programação.

O quarto desafio centra-se no interesse por parte do utilizador em obter informação actual. Basta dar-se um acontecimento invulgar, que suscite o interesse das pessoas e estas tendem instantaneamente a colocar no motor de pesquisa uma query relativa ao assunto com o intuito de obter informação actual. Neste caso, o interesse é colocado em artigos noticiosos, que para poderem ser dados como relevantes não podem ser classificados com as mesmas técnicas de análise por relação entre hiperligações, visto que, muito normalmente uma notícia recente aponta para praticamente nenhum outro documento. Um caso bastante recente que confirma este tipo de situação foi a da morte do cantor Michael Jackson, cujo número de pesquisas efectuadas ao mesmo tempo foi tão elevado que um dos mais famosos motores de busca foi obrigado a ficar indisponível por uns minutos [5].

Por fim, um outro entrave relevante, está relacionado com os sinónimos (diferentes palavras podem ter o mesmo significado) e palavras polissémicas (uma palavra pode ter diferentes significados) presentes nas diversas linguagens, o que faz com que nem sempre uma query possa ser bem definida e explícita. Por exemplo, a palavra sol pode ser relativa à estrela principal do nosso sistema solar, ou a um jornal conhecido. Os documentos devolvidos para uma query deste tipo abordarão um pouco de todas as possibilidades, resultando numa lista, algo confusa, para um utilizador que procure apenas um destes assuntos.

1.2 A nova geração de sistemas WebIR

Face aos problemas correntes e previamente referidos com que os sistemas de recuperação de informação se deparam, torna-se fulcral encontrar novas metodologias capazes de responder e satisfazer os objectivos básicos de qualquer motor de pesquisa: devolver os resultados mais relevantes para o utilizador. Uma análise pela literatura actual e somos capazes de constatar uma grande contribuição neste âmbito, por parte de variadíssimos autores, onde múltiplas teorias e experiências parecem fazer crer que os sistemas de WebIR serão capazes de respirar novamente e fazer aquilo para o qual foram construídos. Apesar de diversas, as abordagens recaem sobre três tópicos que se interligam: ordenação (ranking), categorização (clustering) e personalização [36] [52] [31] [43] [42] [45] [37] [39] [23]. Muitas das soluções passam pela colaboração entre pares destes tópicos, ordenação e categorização [31] [43] [42], ordenação e personalização [36] [52] [37] [39], mas poucas fazem o uso colaborativo de todas elas [45]. A nossa aposta vai para esta última abordagem, na qual pensamos estar a chave para o sucesso da nova era dos sistemas WebIR, ainda que haja um longo caminho a percorrer.

A categorização pode ser definida como o processo de agrupamento de objectos existentes num conjunto inicial em novos subconjuntos que partilham propriedades semelhantes. No caso concreto da categorização de documentos para uma pesquisa Web (Web search clustering), isto significa que, após a recepção de um conjunto de documentos relevantes para uma query, estes são agrupados em novos conjuntos tendo por base a semelhança que os documentos têm entre si. Esta geração é derivada de um processo automático e não deve ser confundido com a classificação de documentos para a qual, são pré-definidas categorias e aos documentos são atribuídas categorias cujas propriedades lhe são mais próximas. A categorização, é um processo fulcral quando se trata de estruturar e organizar amplos conjuntos de informação heterogénea (a Web é um claro exemplo disso). A natureza navegável da hierarquia criada dinamicamente quando aliada a uma boa descrição dos conteúdos de cada grupo, ajuda o utilizador a identificar mais rapidamente os resultados mais relevantes para o seu interesse temporal. A pessoa tem a possibilidade de consultar num espaço compactado os vários assuntos inicialmente dispersos por entre os resultados, navegar por aqueles que lhe são mais chamativos e analisar um conjunto mais restrito, mas bem mais representativo do tópico seleccionado. No fundo, a categorização pode actuar como um catalisador que ajuda o utilizador a resolver em tempo real os problemas derivados

da polissemia e dos sinónimos, para além de extrair conhecimento inicialmente oculto.

A literatura apresenta uma diversidade de algoritmos capazes de gerar uma boa categorização dos resultados (provenientes de uma primeira fase de extracção dos documentos mais relevantes para uma query) baseados em múltiplas ideologias [31] [43] [42] [45] [22] [21] [27] [46]. Estas passam por efectuar primeiro os agrupamentos e depois ver qual a melhor expressão que representa cada conjunto, a analisar primeiro todo o conjunto de documentos para extrair as melhores expressões e depois agrupar os documentos com proximidade a essas expressões. Facto que praticamente todas as abordagens impõem é mesmo o processo de categorização da informação, como sendo uma boa alternativa à actual lista de resultados devolvida pelos sistemas WebIR.

A personalização é todo o processo efectuado para adaptar um qualquer conteúdo a uma entidade específica. A personalização no contexto de um sistema WebIR, significa realçar todos os resultados da pesquisa que estejam mais relacionados com o utilizador ou, como utopia, devolver exclusivamente os conteúdos que no seu imaginário ele pretende. Isto só é passível de ser efectuado quando o sistema possui conhecimento sobre a pessoa para a qual está a devolver os resultados. Tal conhecimento é representado através de um modelo ou perfil de utilizador, expressão regularmente utilizada por diversas entidades como psicólogos, sociólogos, investigadores criminais entre outros de forma a caracterizarem um grupo ou individualidade de acordo com aquilo que lhes é mais específico. Um psicólogo caracterizará certamente um paciente com um registo da sua infância, dos seus hábitos sociais, dos seus problemas de saúde ou familiares, para poder mais tarde tomar uma medida capaz de reequilibrar esse ser. Um modelo de utilizador para um sistema de WebIR é todo este registo de informação que permite identificar aquilo que o usuário é de modo a que o sistema possa, com o auxílio desta informação, ultrapassar os problemas da ambiguidade e subjectividade, apresentando os resultados com real interesse. Imaginem-se, dois utilizadores apaixonados pelos seus respectivos clubes FC Porto e SL Benfica, ao introduzirem a mesma query ”futebol”. Sem a interacção com os modelos de utilizador, o sistema retornará os mesmos resultados para ambos os interessados. Porém, recorrendo aos perfis de cada utilizador, o sistema será capaz de apresentar resultados diferentes a cada utilizador indo ao encontro das suas pretensões e facilitando a descoberta mais rápida de documentos de interesse para qualquer uma das partes.

Este tipo de personalização no campo das ciências computacionais enfrenta dois grandes desafios.

O primeiro depreende-se com a dificuldade em criar perfis de forma automática e com um grau de precisão considerável. Diversas metodologias [41] [33] [40] [32] aspiram a resolver o problema recorrendo a múltiplos recursos de carácter explícito e não explícito para o utilizador. Se [39] [50] recorrem à indexação das páginas Web referentes ao histórico de navegação dos utilizadores para extraírem a informação representativa dos interesses com base em técnicas de categorização, outros preferem fazê-lo através da análise e extracção de conhecimento a partir da estrutura das queries [51]. De modo a tentar melhorar os seus resultados, muitas das abordagens permitem ao utilizador modelar e interagir com os perfis propostos, mas num mundo onde são poucos os utilizadores que perdem tempo a passar à próxima página de resultados, será algo estranho observá-los a optimizar o seu próprio perfil.

O segundo deve-se a questões éticas mais concretamente à privacidade, motivo pelo qual cada vez mais movimentos de protesto são realizados pelo mundo [12].

Actualmente, no âmbito comercial os principais motores de WebIR devolvem o mesmo resultado independentemente da pessoa que efectuou a pesquisa. Os sistemas de categorização como o Vivíssimo [18] dão provas das potencialidades desta tecnologia como forma de ajudar o utilizador a navegar pela quantidade imensa de resultados. Sistemas capazes de modelar o comportamento e associá-lo a um motor de pesquisa global começam a surgir, contudo ainda que numa fase de protótipo. Os seus resultados porém mostram melhorias relativas ao nível da ordenação dos documentos, quando se tem em conta o perfil dos utilizadores. Normalmente, o processo de categorização serve para gerar e mostrar as categorias inicialmente ocultas por entre os resultados ou, no caso da personalização para gerar os modelos de utilizador através da sua sobreposição aos documentos que o utilizador achou importantes. Nós pensamos haver uma outra forma de utilizar a categorização. Fazer dela uma ponte de ligação entre tudo aquilo que o navegador pesquisa na Web (histórico das pesquisas) e que é considerado relevante, de forma a criar um modelo de utilizador construído simplesmente a partir deste histórico e das categorias associadas a estas pesquisas. Este modelo de utilizador pode ser utilizado para diversos fins, desde pesquisas na Web ao comércio centralizado.

1.3 Objectivos

O objectivo desta tese é o de melhorar os resultados devolvidos por um sistema de recuperação de informação fazendo uso da unificação das duas soluções actualmente mais em voga no que toca a ultrapassar as barreiras impostas pela ambiguidade das queries. São elas a categorização automática dos resultados mais frequentemente denominada por clustering e a personalização.

Ambas as soluções estão ainda numa fase muito embrionária de adaptação aos sistemas de IR comerciais porém a categorização leva uma ligeira vantagem no que toca à implementação nesta área onde o poderoso motor de busca Vivíssimo [18] é referência neste sector. Por outro lado a integração de um sistema de personalização no site de comércio Amazon [15] demonstra bem as capacidades da personalização na Web.

Os motores de pesquisa de uma forma geral passam por quatro fases até devolverem os resultados ao utilizador. Elas são o crawling, a indexação, a procura e a ordenação dos resultados. A informação relativa aos interesses dos utilizadores poderia muito bem ser acoplada em qualquer uma destas fases porém o custo de implementação seria muito elevado. A forma mais simples e mais abordada [37] [39] [23] é a de adaptar este conhecimento do utilizador aos resultados finais provenientes de um motor (ou meta-motor) de pesquisa, fazendo uma reordenação dos resultados tendo em conta o perfil do utilizador.

Frequentemente as duas soluções não são usadas homogeneamente. Muitas das contribuições quando focam a utilização da categorização para a melhoria dos resultados permitindo ao utilizador socorrer-se da escolha das categorias que mais lhe são chamativas, não utilizam a personalização para tentarem reordenar essas categorias. Do nosso ponto de vista a aglomeração destas duas soluções pode trazer vantagens visto que uma auxilia a outra.

De modo a resolver o problema de imparcialidade em relação ao utilizador e aos seus interesses por parte dos WebIR, pretendemos criar um sistema capaz de categorizar os resultados provenientes de múltiplos motores de pesquisa, melhorando assim a experiência de utilização ao possibilitar a restrição do número de documentos face á escolha da categoria mais apelativa. Este sistema ao contrário de muitos outros será completamente independente da linguagem e de enriquecimento externo.

Com vista a introduzir conhecimento sobre os utilizadores num sistema de recuperação de informação, pretendemos criar automaticamente modelos de utilizadores a partir do armazenamento do seu historial de pesquisas onde para além das queries introduzidas também são guardados os clusters que lhes são relativos. Desta forma com o uso destes modelos de utilizador, os sistemas de pesquisa poderão reordenar os resultados e as categorias de modo a colocar em destaque todo o conteúdo que tem mais interesse para o utilizador.

1.4 Organização da tese

No capítulo 2 será efectuada uma análise à literatura relacionada com a área de WebIR e a sua optimização. Serão estudados os sistemas de IR e o seu funcionamento geral, passando depois para uma revisão mais profunda dos processos de clustering e de criação de perfis. É aqui também apresentada a proposta de trabalho.

O capítulo 3 refere-se ao trabalho desenvolvido ao longo da investigação onde serão descritas as metodologias e os algoritmos utilizados para a categorização de snippets proposta.

O capítulo 4 abordará o processo de construção dos modelos de utilizador criados com o auxílio da categorização.

No capítulo 5 serão apresentados e discutidos os resultados do trabalho.

Finalmente o capítulo 6 da tese apresenta as conclusões e algum, possível, trabalho futuro.

Capítulo 2

Estado da arte

2.1 Sistemas de recuperação de informação

Durante muitas décadas as pessoas tornaram-se conscientes da importância de encontrar e armazenar informação. Com o surgimento da era dos computadores, tornou-se possível guardar grandes quantidades de documentos e como tal, encontrar informação útil dentro destas bases de conhecimento tornou-se uma necessidade. O campo dos sistemas de recuperação de informação foi criado em 1945, fruto destas necessidades [24]. Em 1960, Gerard Salton desenvolveu o sistema SMART, protótipo de um sistema IR que serviu de teste a algoritmos que automaticamente indexavam e devolviam documentos. Muita teoria de Text Mining estava lá: análises estatísticas de texto, extracção da raiz das palavras, remoção de palavras que não representam conhecimento e teoria da informação, tudo foi experimentado. Como em todos os sistemas desenvolvidos até esta data, os conjuntos de informação analisados pertenciam a um ambiente controlado onde o conteúdo existente era realmente importante e bem estruturado, tinha dimensões grandes mas nada de extraordinário e era bastante estático no sentido de que o conteúdo dos documentos não variava.

O surgimento da Web mudou a situação por completo. A multiplicidade de conteúdos heterogéneos, dinâmicos, objectivos e subjectivos, aliados ao crescimento exponencial das fontes que os disponibilizam, fez com que uma nova realidade fora de controlo emergisse. De modo a existir uma adaptação a esta nova situação, foi criada uma nova disciplina: os sistemas de recuperação de informação na Web (Web Information Retrieval ou WebIR).

Esta tem por base muitos conceitos tradicionais de recuperação de informação, mas foi capaz de introduzir muitos mais engenhos inovadores capazes de se adaptarem ao meio e às circunstâncias.

2.1.1 Os sistemas WebIR

Os sistemas de recuperação de informação tradicionais propuseram diversos modelos capazes de representar o conteúdo dos documentos [24]: booleanos, probabilísticos, de inferência e de espaço vectorial. Os últimos três são os mais conhecidos devido a permitirem calcular e atribuir um valor de interesse para cada documento face a uma query, levando a que no final os resultados a devolver possam ser ordenados com base nesta qualificação. Os modelos booleanos apesar de também conseguirem ordenar resultados por uma certa característica, não têm esta visão de quantificação da proximidade entre documento e query. É contudo, o modelo de espaço vectorial (vector space model) que merece mais destaque face à sua maior aceitação por parte dos sistemas IR.

Actualmente a Web é a maior fonte de informação do mundo. Notícias, vídeos, música, fóruns, jogos, informação militar, análises e explicações detalhadas de todo o tipo de material existente à face da Terra, tudo está lá. Difícil mesmo é encontrar o que se pretende numa rede que não pára de crescer e de se actualizar. Os modernos sistemas de recuperação de informação na Web permitiram com a exploração de algumas das metodologias de IR clássicas, desenvolver metodologias inovadoras capazes de encontrar informação relevante nesta rede tão complexa. Um estudo recente [7] mostra que 80% dos utilizadores que navegam na Web encontram os sites, que visitam depois com frequência, a partir dos motores de pesquisa na Web como o Google [19], Yahoo [20], Bing [17] ou Ask [16]. Se um utilizador pretende saber algo sobre um local ou uma tecnologia ou simplesmente encontrar o restaurante mais próximo dele, simplesmente introduz uma query no motor de busca e espera numa fracção de segundos obter as suas respostas.

Tal facto faz com que esta disciplina se tenha estabelecido como uma das mais importantes áreas de pesquisa e interesse da comunidade científica, cujo intuito é desenvolver sistemas capazes de providenciar cada vez mais melhores resultados. Nos parágrafos que se seguem, a arquitectura geral destes sistemas bem como algumas das suas tecnologias serão abordadas se bem que de uma forma algo leve dado não ser directamente o âmbito desta

tese. Assim propomos uma leitura de [47] [48] [24] para se familiarizar com os sistemas de WebIR.

2.1.2 Estrutura de um motor de pesquisa Web

O processo de execução de um sistema WebIR pode ser visto como um aglomerar de quatro fases: extração de documentos ou páginas Web, indexação, procura e ordenação. (ver figura 2.1)

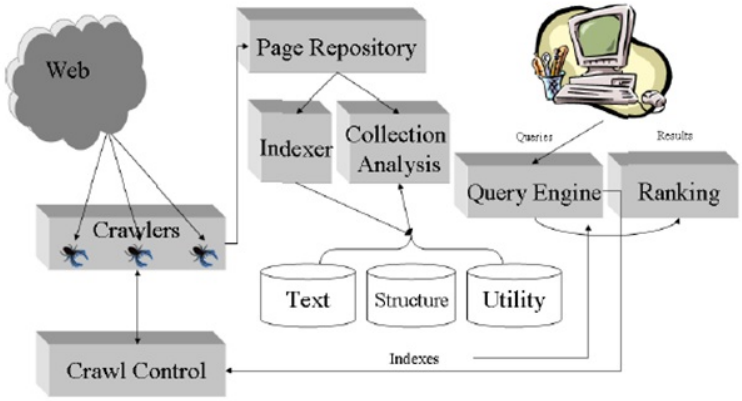

Figura 2.1: Arquitectura de um motor de pesquisa.

A extração de documentos Web, tarefa talvez mais conhecida por ”crawling”passa por percorrer a rede ou parte dela e reunir o conteúdo dos documentos guardando-os num repositório. Tal trabalho só é possível graças à computação distribuída. Este trabalho é feito por agentes controlados que operam de diversas formas conforme os requisitos impostos pela entidade que os controla. Esta pode definir a maneira como os agentes percorrem a Web segundo certas políticas. Em certos intervalos de tempo, o repositório é processado sendo os documentos analisados e indexados num grafo de forma a permitir um rápido acesso à estrutura e ao conteúdo dos documentos. Neste grafo os documentos são os vértices e as hiperligações as arestas. Esta estrutura é depois analisada tanto do ponto de vista relacional como contextual, permitindo fazer o ranking de cada documento. É na análise da parte relacional que se encontram muitas das inovações face aos tradicionais sistemas de pesquisa. Além de considerar os termos que o documento contém e a sua importância relativa ao conteúdo do documento (considerar para o efeito como este sendo o conjunto A), são considerados também os termos que representam os textos correspondentes às hiperligações que apontam para este mesmo documento (considerar para o efeito este como sendo o

conjunto B). Tal deve-se ao facto de muitas vezes as descrições associadas às hiperligações, serem boas representações do conteúdo do documento para as quais se referem. Esta teoria está já bem cimentada como se pode constatar em [47]. Desta forma o conteúdo que representa cada documento na estrutura é proveniente da reunião dos conjuntos A e B. Por razões de eficiência, parte do processo de ranking é feito off-line, antes da query ser submetida ao motor de pesquisa.

Actualmente os motores de pesquisa indexam biliões de páginas através do recurso a plataformas de computação distribuída onde os índices da estrutura são replicados por vários clusters de servidores autónomos. Esta distribuição é necessária para sustentar e dar resposta aos milhões de queries introduzidas pelos utilizadores dia após dia.

2.2 Categorização

O termo categorização ”clustering”denota um alargado número de metodologias para identificar estruturas comuns escondidas em grandes conjuntos de objectos. Este tipo de metodologias é aplicado nas mais diversas áreas como a física, biologia, estatística, análise numérica, mineração de dados entre muitas outras. Um cluster é um grupo de objectos cujos membros são mais similares entre si, que os elementos de outros clusters. Pode-se por isso dizer que a similaridade intra-grupo é alta mas a similaridade inter-grupos é baixa. No caso especifico da área de IR, os objectos referidos são os documentos da Web na sua completude ou parcialmente. Salientamos o facto de esta categorização de objectos ser um processo todo automático, não podendo ser confundido com classificação onde existem à partida diversas categorias pré-definidas. Também é de realçar que a categorização que vamos abordar é relativa ao conteúdo Web e como tal foca-se no trabalho com texto. Uma particularidade deste tipo de categorização é que não tem só como objectivo fazer uma boa geração de grupos mas também atribuir a estes grupos um nome que represente bem o seu conteúdo (Labeling).

2.2.1 Métodos de categorização

Os métodos de categorização podem ser classificados atendendo a vários aspectos como a estrutura, a unidade de indexação, a duração e o algoritmo utilizado.

A estrutura pode ser plana (flat clustering) onde só existe um primeiro nível de grupos apresentado sob a forma de uma lista. No caso de os grupos serem organizados numa estrutura em árvore denomina-se por hierárquica (hierarchical clustering). Os elementos da raiz da árvore, são os grupos de nível 1. Cada um desses grupos pode ainda estar subdividido em mais grupos e assim sucessivamente. Qualquer uma destas estruturas pode ser designada de overlap caso os elementos possam pertencer a mais que um grupo.

A unidade de indexação refere-se á representação dos objectos. Nesta área em particular os documentos são representados por um conjunto de palavras, vulgarmente referido como saco de palavras ou por um conjunto de frases onde a ordem das palavras é tida em conta.

Quanto à duração, a categorização pode ser efectuada sobre um conjunto de documentos persistente, cujas propriedades (o conteúdo) não variam ao longo do tempo. Ou, sobre um conjunto que existe por breves momentos como o caso dos documentos devolvidos por um motor de pesquisa na Web para uma query fornecida. Denomina-se este último caso por categorização efémera.

O algoritmo é o tipo de sequência utilizada para efectuar a categorização dos objectos. Esta pode ser por divisão (o processo é iniciado a partir de um conjunto de objectos e efectua-se a subdivisão deste em múltiplos grupos, podendo alguns dos elementos pertencer a diferentes conjuntos) ou por Aglomeração (começa-se individualmente por cada um dos elementos e efectua-se a junção destes em grupos existindo a mesma possibilidade de overlap).

Nos sistemas de recuperação de informação tradicionais o processo de categorização persistente era considerado o mais comum visto ”os documentos existirem num ambiente controlado e bastante estático sendo necessário categorizar a informação apenas uma vez e depois manter a estrutura”como refere Salton em [30]. Correntemente tal não acontece pois ainda que existam múltiplas estruturas de documentos estáticas, a Web não o é e como tal todo o seu conteúdo é muito dinâmico. Os motores de pesquisa na Web de modo a mostrar os resultados em categorias navegáveis muito rapidamente, utilizam a categorização efémera sobre os resultados onde a durabilidade da estrutura é proporcional ao tempo da sessão com o utilizador.

2.2.2 Categorização de Web snippets

No decorrer deste capítulo temos vindo a analisar o processo de categorização, mais concretamente, a categorização de documentos da Web. Contudo existe ainda um tipo de categorização mais específico, de Web snippets. Neste caso, o conteúdo original de uma página Web é substituído por um conjunto de expressões representativas do documento, sendo esse conjunto de texto muito mais reduzido, dado como entrada para o algoritmo de categorização. A categorização de snippets foi primeiramente introduzida no sistema Northern Light [10] (funcionava mais como um classificador) mas tornou-se famosa com o motor de pesquisa Vivíssimo. Tudo o que se pretende de um sistema de categorização de documentos Web, pretende-se também na categorização de snippets. Contudo, as dificuldades deste tipo de categorização são ainda mais acentuadas visto que o conjunto de informação com que se trabalha é muito mais reduzido.

O termo snippet pode não ser muito conhecido para quem não seja da área, porém praticamente todas as pessoas que já utilizaram um motor de pesquisa, já olharam para conteúdo deste tipo. O snippet é todo o conjunto de expressões que existem no título e descrição, associados a um url. Ou seja, é a informação associada a cada url existente na lista de resultados apresentada, para uma qualquer query introduzida, em um qualquer motor de pesquisa.

Uma das vantagens associadas ao uso dos snippets para efectuar o processo de categorização está relacionada com a rapidez de execução do algoritmo. Qualquer sistema que faça uso dos resultados (sem categorização) de um motor de pesquisa e pretenda efectuar a devida categorização, recebe normalmente estes snippets numa lista ordenada por relevância. O sistema pode então efectuar dois processamentos: carregar para cada url o conteúdo completo da página que este representa ou, trabalhar directamente com esta informação mais reduzida. O primeiro caso implica que sejam descarregados da Web o conteúdo de cerca de 100 a 500 páginas para só depois ser efectuada a categorização. Mesmo que já exista uma grande quantidade de páginas indexadas, a quantidade de informação, torna o sistema demasiado lento para poder trabalhar em tempo real e satisfazer milhares de queries por dia. Outra das vantagens dos snippets relaciona-se com o facto de um documento normalmente abordar múltiplos tópicos. Os snippets extraídos dos documentos, estão normalmente mais relacionados com o tema procurado e devido a isso não é necessário segmentar o texto à

procura do tópico de interesse.

O trabalho com os snippets permite a criação de sistemas capazes de categorização em tempo real, contudo e como já referido a informação é muito menor, existindo dificuldade acrescida na criação de bons clusters e boas expressões identificadoras do conteúdo destes.

Para resolver os problemas inerentes a este tipo de categorização, muitas das abordagens existentes passam não só pelo cálculo da similaridade entre o conteúdo dos documentos para depois formar grupos (document-derived clustering), mas também através da valorização de termos ou expressões compostas que façam parte do conjunto de snippets (label-derived clustering, como referido em [45]). Estes termos mais fortes servem depois para aglomerar documentos que as contenham, gerando assim clusters. O algoritmo para a categorização de snippets proposto nesta tese segue este segundo tipo de metodologia.

2.2.3 Trabalho relacionado

O trabalho relacionado pode ser dividido em duas categorias: comercial e académico. Tal acontece porque a categorização de Web snippets já é uma metodologia bastante implementada no mundo comercial e apesar de muitas ideias surgirem do mundo académico, é importante realçar o esforço que algumas entidades fazem em explorar este potencial e disponibilizá-lo aos utilizadores.

No mundo comercial o primeiro sistema a efectuar a categorização de snippets foi o Northern Light [10], o qual, para cada documento considerado, atribuía um conjunto de tags identificadores de categorias que pudessem estar associadas a esse documento. Esta atribuição era feita manualmente, tornando este sistema mais como um classificador. Os resultados eram devolvidos e consoante as tags associadas a cada documento, também eram apresentadas as categorias que representavam esses documentos. O sistema foi desactivado em 2002.

Desde esta data, diversos motores de categorização de Web snippets foram surgindo. Funcionam na sua globalidade como meta-motores de busca visto que os resultados relevantes para uma query são obtidos a partir de outros motores de pesquisa. Entre os mais conhecidos estão o Kartoo [8], Mooter [9], iboogie [6], Vivíssimo [18], Grokker

[4] e Carrot2 [2]. É contudo o Vivíssimo que apresenta melhores resultados tendo até sido distinguido com diversos prémios na área. Infelizmente pouco se conhece sobre a metodologia implementada neste sistema que a cada dia que passa apresenta resultados melhores.

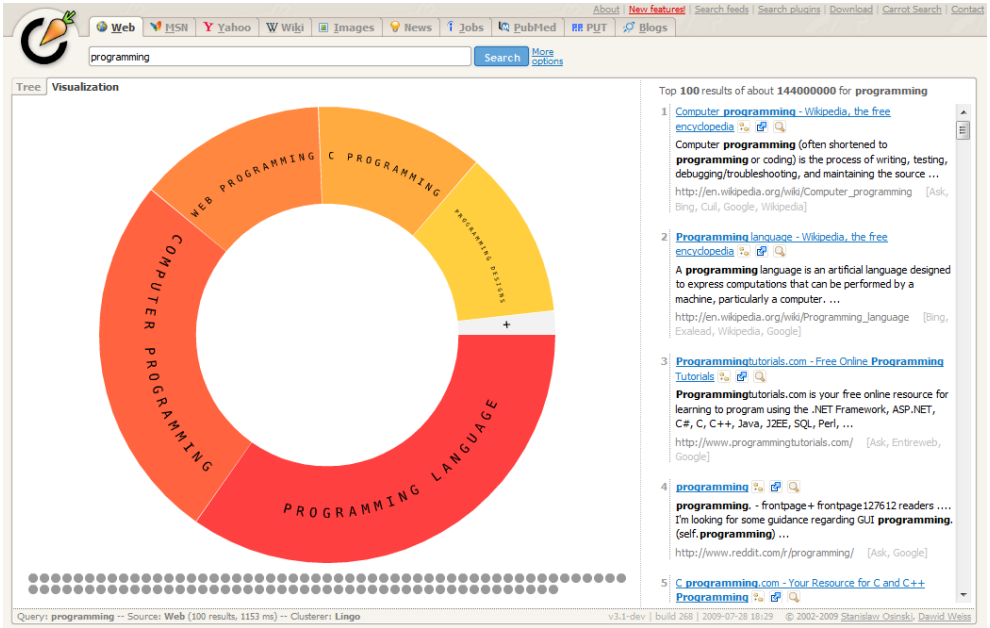

Figura 2.2: Carrot, visualização em círculo.

Figura 2.3: Carrot, visualização em lista.

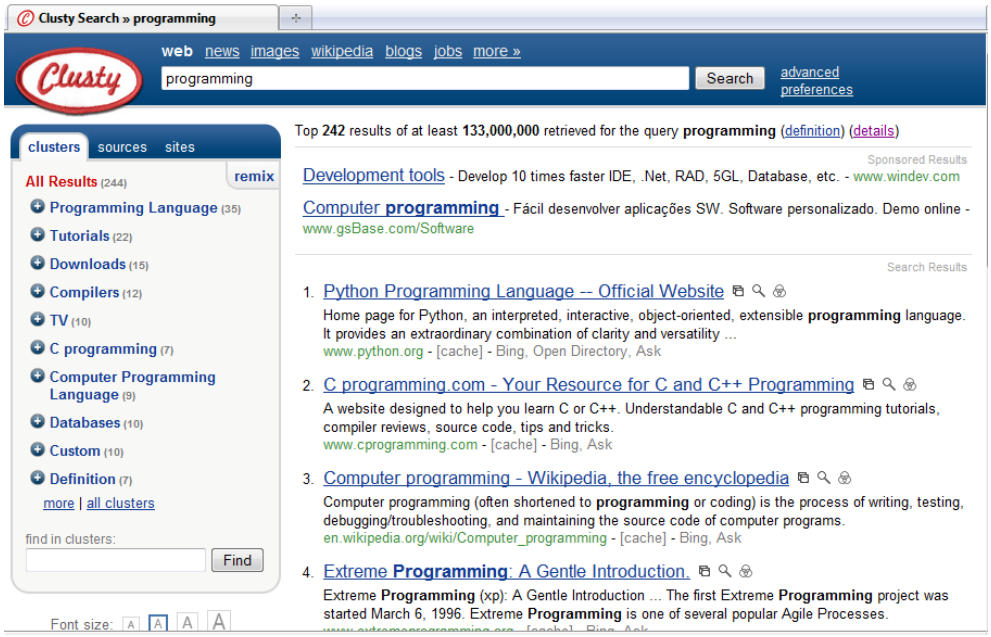

Figura 2.4: Clusty (Vivíssimo).

O Carrot2, é também um dos meta-motores de pesquisa com categorização que apresenta resultados relevantes e de uma forma visualmente atractiva (quando se visualizam as categorias no modo gráfico). Possibilita a geração dos clusters com base na escolha entre

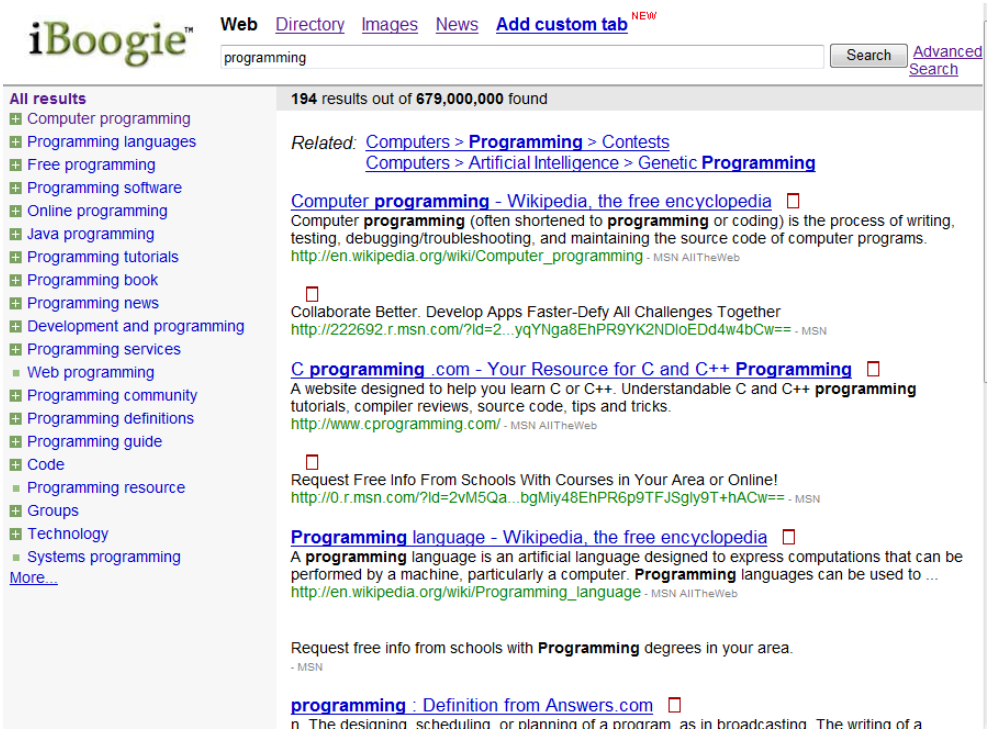

Figura 2.5: iBoogie.

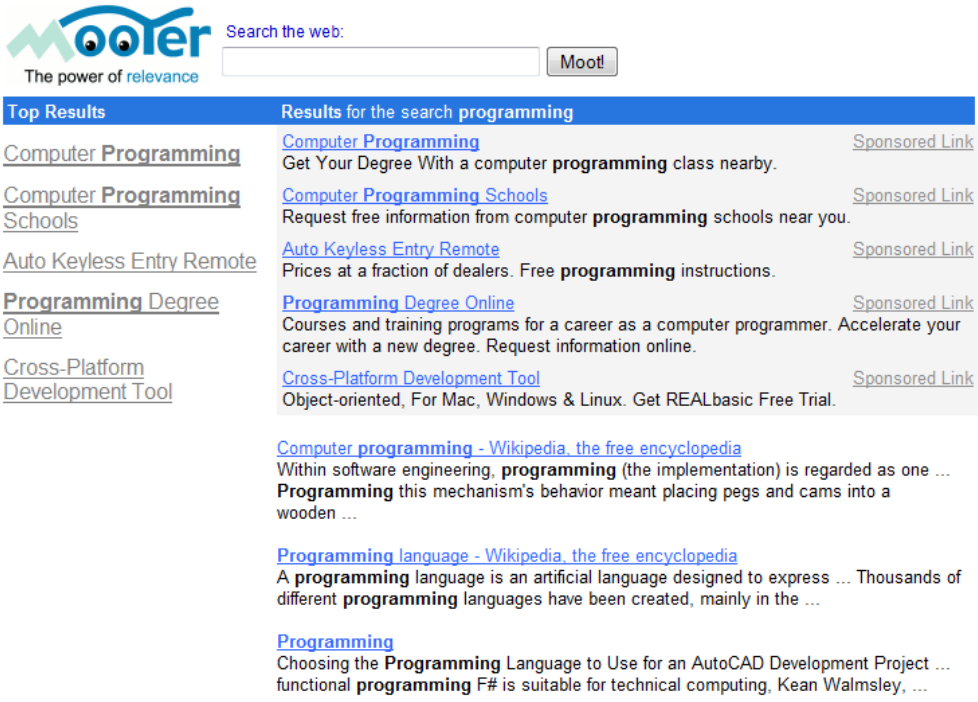

Figura 2.6: Mooter.

um algoritmo de árvore de sufixos (STC) ou com o uso do Lingo [49], um algoritmo que também efectua a geração de clusters com base na descoberta das frases mais fortes.

No mundo académico, são muitas as propostas existentes que visam conseguir o melhor processo de categorização tendo em conta um ou vários factores mais importantes como a qualidade, o tempo de execução, a dependência de fontes externas de conhecimento entre outros. Segue-se uma análise a algumas das abordagens mais bem sucedidas.

Ferragina P. e Gulli A. [45] propõem um sistema de categorização hierárquica de snippets denominado Snaket. Seguem uma abordagem do tipo label-derived dado fazerem uso da selecção dos melhores termos existentes nos snippets para efectuarem a geração dos clusters. De modo a ultrapassar o problema da normalmente pouca informação existente nos snipptes, fazem o enriquecimento destes com o uso de duas bases de conhecimento.

Estas bases de conhecimento são construídas offline e são utilizadas em conjunto com o $TFIDF$ no processo de atribuição de um valor de importância dos termos. Uma, guarda os títulos das hiperligações extraídos a partir de mais de 200 milhões de páginas Web e a outra indexa a base de dados $DMOZ$ que classifica $3,500,000$ sites em mais de $460,000$ categorias (é com o auxílio desta base de dados que é dado um valor de importância aos termos, consoante a sua ocorrência nas diversas categorias).

Ao adicionarem informação a cada snippet, estes passam a ser mais completos visto geralmente os títulos das hiperligações serem bons descritores do conteúdo dos sites para onde apontam. Contudo ao fazerem uso das bases de conhecimento, estão a limitar a área de acção visto que não podem obter informação para todos os documentos existentes na Web.

Após este enriquecimento dos snippets, são extraídas aquilo que os autores chamam de ”gapped sentences”i.e. expressões compostas por termos não consecutivos. No final, a cada url é associado, se possível, um conjunto de expressões compostas não consecutivas. A geração dos clusters parte destas expressões. Os snippets que partilham as mesmas ”gapped sentences”candidatas, falam de um mesmo tema e portanto são agrupados.

Para a geração de uma estrutura hierárquica, são extraídos do conjunto de urls de cada grupo expressões compostas secundárias que ocorram em 80% dos membros de cada grupo. Passa assim, cada conjunto a ser representado por uma expressão composta primária, ao qual se segue um conjunto de outras expressões compostas que representam um conjunto de documentos mais específicos.

Estando a estrutura hierárquica construída, é feito o ranking dos documentos de cada

cluster com base no ranking das suas expressões.

O sistema Snaket ambiciona ainda ser um sistema personalizado e como tal, permite que o utilizador após receber a estrutura hierárquica das categorias associadas à query pesquisada, possa seleccionar um conjunto de títulos para assim filtrar a lista de documentos devolvidos cujo conteúdo pertença somente a esses títulos. Salienta-se ainda que este foi um dos trabalhos que foi fonte de inspiração ao trabalho proposto nesta tese. Tal facto deve-se ao agrupamento dos urls com base na força das suas expressões e não na simples similaridade entre termos.

Geraci F. et al. [27] efectuam também eles a categorização de Web snippets com o seu sistema Armil [3]. Este é um sistema com o intuito de ser muito rápido e para tal, é apenas dependente do conteúdo dos snippets. A obtenção dos documentos relevantes para uma query é feita a partir de um meta-motor de pesquisa. Ao contrário de Ferragina e Gulli que propõem a criação dos clusters com base na força das expressões (sendo a sua descrição já efectuada á priori), os autores seguem uma abordagem document-derived, pois tomam como sendo mais importante a fase de agrupamento dos documentos e só depois a fase de procura dos melhores descritores do conteúdo de cada cluster.

O agrupamento dos documentos tendo por base os snippets, é feito através do uso de um algoritmo "furthest-point-first" (FPF) para um k número de clusters. Os snippets são por isso representados num modelo de espaço vectorial e é calculada a sua similaridade com base numa métrica não referida pelos autores. É depois aplicada sobre estes documentos uma versão modificada do FPF denominada $M - FPF$ que para um k dado, constrói k conjuntos de documentos. Cada documento só pode existir num cluster, classificando-se assim o algoritmo como "non-overlap".

A escolha dos nomes representativos dos clusters é feita com base na medida de ganho de informação atribuída a cada termo que maximiza a existência desse termo dentro do cluster e a desvaloriza conforme a sua existência em outros grupos. A expressão atribuída como descritivo do grupo é não um destes termos, mas sim, uma expressão retirada de um dos títulos que exista no conjunto e que contenha um ou mais dos melhores termos.

Lipai [22] [21], apresenta a construção de uma interface completa que permite a categorização dos resultados com base no algoritmo k-means. Contudo para a geração das categorias, todo o conteúdo dos documentos é considerado, sendo por isso o algoritmo algo exigente em termos de tempo e processamento. Os documentos são representados

no modelo de espaço vectorial, sendo depois calculados os centroides correspondentes ao número de clusters definido previamente. Após sucessivas iterações, obtêm-se os conjuntos finais que são associados a um identificador, também este simples, resultante da escolha dos termos com mais peso.

Campos R e Dias G [46] propõem tal como Lipai um sistema de categorização de documentos da Web fazendo uso da totalidade do seu conteúdo e de técnicas de mineração de dados. A metodologia usada é simplista apesar de ter uma forte componente teórica. É descrita em quatro fases: (1) selecção das páginas mais relevantes de entre o conjunto de páginas devolvidas pelo motor de pesquisa; (2) a extracção de multipalavras dos documentos com o uso do software SENTA; (3) detecção dos termos mais relevantes representativos de um documento fazendo o uso do WEBSPY e (4) apresentação dos documentos numa estrutura hierárquica através do uso do POBOC.

O primeiro passo do algoritmo tem em vista remover documentos com base na análise da frequência do seu domínio por entre os resultados. Para os que são seleccionados (frequência do domínio muito baixa), são obtidos através de uma nova pesquisa os n documentos mais relacionados com esse domínio. Estes documentos são adicionados ao conjunto de resultados.

No segundo passo, são identificadas multipalavras por entre os documentos com o uso do SENTA, um software de extracção de n-grams puramente estatístico e independente da linguagem. Estas multipalavras de cada documento servem de entrada ao software WEBSPY de forma a serem encontrados os termos mais relevantes à query para cada documento. O WEBSPY faz uso da análise de 12 características entre as palavras ou frases existentes no documento em análise, devolvendo no final um conjunto de termos associado a uma probabilidade de relevância para a query. Segue-se a geração da matriz de similaridade entre documentos, fazendo o uso da medida de Cosine para o cálculo da similaridade entre cada par de documentos representados pelos termos mais fortes escolhidos para cada um.

Finalmente procede-se à geração da estrutura hierárquica com base no uso do algoritmo POBOC e da matriz de similaridade criada no passo anterior. O identificador de cada cluster é obtido escolhendo para cada cluster a palavra ou multipalavra mais frequente.

Também Zamir e Etzioni [43] propõem a categorização de snippets mas com recurso a metodologias bastante diferentes. Os clusters são gerados através de um algoritmo de árvore de sufixos denominado ”Suffix Tree Clustering”(STC) que possibilita a sua construção com

base nas frases comuns a cada documento. Uma frase segundo o contexto dos autores, é uma sequência ordenada de uma ou mais palavras.

O algoritmo processa-se em três fases: limpeza do documento, identificação dos clusters base e combinação de clusters base.

Na primeira fase, é efectuada a remoção de certas tags e caracteres especiais, bem como é aplicado o processo de stemming a cada palavra.

A segunda fase, passa pela criação da árvore representativa das frases e da ligação entre as palavras. Esta árvore apresenta as seguintes propriedades:

- A árvore é enraizada e direccionada;
- Cada nodo interno contém pelo menos 2 filhos;
- Cada aresta é nomeada com uma sub-string de S;
- Nenhuma das arestas que ligam a cada nodo, podem conter palavras que estejam nas arestas acima do nodo;

Onde S, é uma string representativa das palavras de uma frase.

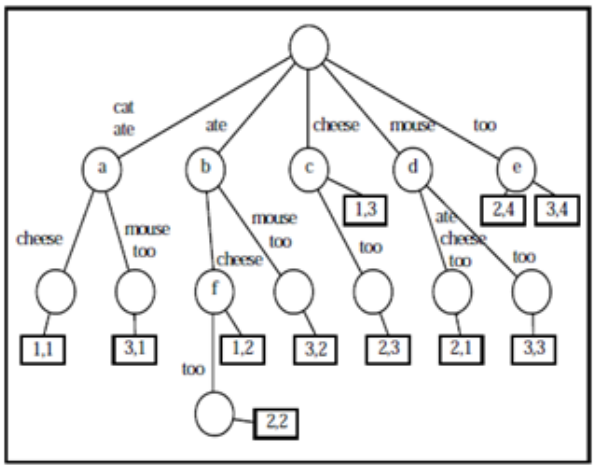

Figura 2.7: A árvore de sufixos para as strings: “cat ate cheese”, “mouse ate cheese” e “cat ate mouse too”.

Cada nodo na árvore representa um conjunto de documentos e uma frase comum a todos eles. É por isso considerado que cada nodo na árvore é um cluster base. A cada um destes clusters é atribuído um valor de importância consoante o número de documentos que contem e as palavras que compõe a frase.

Na última fase, os clusters base são aglomerados conforme uma função definida que agrupa conjuntos cuja grande parte dos documentos que pertencem a cada um, esteja presente em ambos os conjuntos. A escolha dos clusters finais é feita através da selecção dos 10 melhores classificados.

2.3 Modelos de utilizador

O constante e alucinante crescimento da informação disponível na internet impõe altos requisitos no que toca a conceitos e tecnologias que proporcionem aos utilizadores uma filtragem relevante dessa informação por forma a satisfazer os seus interesses. Ao longo da secção 2.1, foram abordados os sistemas de recuperação de informação na Web, motores de pesquisa que permitem através de um conjunto de metodologias actualmente muito baseadas em redes sociais, devolver informação relativa a um tópico requerido pelo utilizador através da introdução de uma query. Estes sistemas conseguem de facto devolver documentos cuja relação com a query não se questiona. Porém também é um facto que as queries são muitas vezes ambíguas e aliado ao facto de serem devolvidos muitas vezes milhares de resultados relacionados faz com que o utilizador possa não encontrar nas primeiras páginas aquilo que pretende. De forma a tentar suavizar esta lacuna, o processo de categorização destes resultados foi implementado em muitos sistemas académicos e comerciais. É assim extraído conhecimento oculto por entre os resultados, que é colocado de forma visível ao utilizador, tendo apenas este de navegar na estrutura para o tópico que mais se adequa às suas necessidades. A geração e aplicação de categorização automática nos documentos Web, abordada na secção 2.2, apesar de trazer grandes vantagens ao utilizador apresenta ainda alguns problemas. Muitas vezes, os tópicos apresentados não são suficientemente fortes para que o utilizador possa saber exactamente qual a categoria que pretende. Ao navegar por esta estrutura acaba por perder o mesmo tempo que a navegar pela lista de resultados inicial. A consciência deste facto levou os investigadores a adaptar os modelos de utilizador à personalização dos resultados dos motores de pesquisa de forma a dar relevo aos que vão mais de encontro ao perfil do utilizador.

2.3.1 Modelos de utilizador e WebIR

"Todos diferentes, todos iguais" é uma frase bem passível de ser adaptada à filosofia dos utilizadores de um sistema WebIR. Todos têm as suas convicções, interesses e objectivos o

que os torna diferentes mas quando frente a um motor de pesquisa, todos aspiram a encontrar a informação que mais lhes convém de forma rápida e concisa. Os modelos de utilizador focam-se na primeira parte da frase. Cada um de nós sem excepção tem preferências, gostos, hábitos, que em toda a sua extensão fazem com que tenhamos comportamentos diferentes. Um modelo de utilizador pretende mapear estas características únicas de cada um.

Casos muito frequentes de utilização de modelos de utilizador, estão presentes no nosso quotidiano. Quando vamos a um médico pela primeira vez, ele faz-nos um inquérito sobre o nosso historial de doenças, doenças na família, sintomas actuais, registando estes dados numa possível ficha clínica ou base de dados do computador. Este registo permite depois numa outra consulta, saber quem eu sou sem ter de me estar a pedir novamente essas informações. Este registo é um modelo ou perfil de utilizador.

Um modelo de utilizador pode não ser só referente a uma individualidade. Neste caso o modelo serve para abranger um conjunto de pessoas que têm um interesse mútuo em certas áreas. Esta mutualidade de interesses entre utilizadores permite optimizar a eficácia dos motores de pesquisa integrando a pesquisa colaborativa. Se uma pessoa A tem os mesmos interesses que outra B e está a procurar informação relativa a assuntos (expressos sobre a forma de uma query) que já foram procurados por B e B sabe de alguns documentos D relevantes sobre esse mesmo assunto, então é possível indicar a A que os documentos D provavelmente lhe interessarão.

A adaptação dos modelos de utilizador aos sistemas de WebIR, tem como objectivo, introduzir conhecimento nos resultados devolvidos. Sabendo aquilo que o utilizador é, pode reordenar a listagem inicial de documentos similares com a query de modo a colocar no topo da lista os resultados mais próximos dos interesses do utilizador. Imaginemos dois utilizadores distintos. Ambos introduzem num motor de pesquisa a query “madonna”. Um dos utilizadores tem por hábito ir a concertos musicais e outra tem interesse em saber as líricas de diversas músicas. Com base nos seus perfis, onde estão presentes estas particularidades, é possível reordenar os resultados por base no cálculo de uma distância desses resultados aos interesses de cada utilizador.

2.3.2 Construção de um modelo de utilizador

Muita pesquisa foi feita para reunir e reconhecer os interesses de um utilizador. Os sistemas construídos para efectuar tal tarefa podem recorrer a indicadores de carácter implícito ou explícito. Explícito, quando os utilizadores indicam ao sistema quais os documentos ou outro tipo de informação é de interesse. Implícito, se de alguma forma é extraído conhecimento das acções do utilizador sem que este se aperceba disso.

No exemplo dado anteriormente referente ao questionário efectuado por um médico para a construção de um perfil do utilizador, o utilizador provavelmente não se mostrará relutante quanto ao facto de ter de responder às perguntas. No contexto da Web o caso é bastante diferente pois a maior parte das pessoas quando lhes é proposta a participação em algum tipo de inquérito não está disposta a perder tempo com isso ou, simplesmente não confia na autoridade que pede a informação levantando problemas de privacidade. É por isso necessário encontrar uma metodologia que seja capaz de indicar automaticamente que tipos de documentos são ou não relevantes para um utilizador. Quando estes documentos são encontrados, é possível analisá-los e extrair informação que sendo comum a um conjunto de documentos ou outro tipo de informação, representa os prováveis interesses do utilizador. Imaginemos uma pessoa que consulta diversas páginas sobre notícias relativas ao mundo do desporto e uma grande fatia dessas páginas é relativa ao futebol. Ao analisar estes dados, será muito provável que os termos desporto e futebol tenham uma grande frequência por entre o conjunto de documentos, o que realmente é representativo dos interesses do utilizador. De forma a extrair este tipo de conhecimento dos documentos lidos ou de um histórico de queries pesquisadas por um utilizador, são utilizadas diversas metodologias que serão descritas nesta mesma subsecção.

Reconhecer a informação relevante para o utilizador automaticamente implica fazer uma análise aos seus padrões de interacção com o sistema. Só desta forma é possível reconhecer que informação é relevante para ele. Para tal é necessário a criação de uma ferramenta que lhe permita a realização normal das suas tarefas mas que consiga também efectuar um registo das suas acções. No caso concreto da análise do comportamento do utilizador dentro do mundo Web, é necessário um browser que seja capaz de permitir ao utilizador navegar para qualquer página e efectuar as suas pesquisas mas, que também consiga indicar que documentos o utilizador leu e achou relevantes. Saber que pesquisas efectuou com

resultados úteis, é também muito importante para a obtenção de dados que permitam depois a construção de um modelo de utilizador.

De modo a saber que documentos e consequentemente, pesquisas, foram relevantes, existem muitas variáveis que podem ser tidas em conta:

- O tempo que decorre desde a abertura de um documento ao seu fecho,
- A movimentação do cursor sobre um documento,
- O conteúdo seleccionado num documento,
- O scroll efectuado no documento,
- O número de cliques dado no rato,
- A impressão do documento,
- A adição aos favoritos do documento,

Ao relacionar os valores de algumas destas variáveis é possível saber que documentos foram de facto interessantes (ou não) para o utilizador. De salientar o facto de que uma delas usualmente é muito mais representativa do interesse de um utilizador num determinado documento. Esta variável é o tempo. Como se poderá constar na abordagem a alguns dos estudos referentes ao assunto apresentados de seguida [41][33][40][32][44], todos referem o tempo como sendo o principal indicador do interesse de um utilizador num documento.

Morita e Shinoda [41] focam-se nos documentos do Usenet News Articles com o intuito de demonstrar o interesse dos utilizadores nestes mesmos artigos. Concluem que apenas o tempo disponibilizado pelo leitor na leitura dos documentos é um indicador desse interesse. Salienta-se o facto dos resultados apresentados mostrarem que os tempos mais indicadores de documentos não relevantes (menor tempo gasto), quando confrontados com a leitura percentual do documento, não representam uma leitura total (indicada pelo scroll ate ao final do documento) o que é algo intrigante. Deixa de o ser quando é referido um conceito bastante interessante: normalmente um utilizador consegue saber se um documento é interessante ou não pela leitura do texto presente nos primeiros parágrafos, não necessitando de observar todo o documento quando este não é relevante.

Kim, Oard e Romanik [33] analisam a relevância do tempo de leitura de um documento como forma de previsão para preferência de leitura de artigos de jornais profissionais ou académicos. Também é estudada a relação entre o tempo dispendido a consultar um documento e a acção de impressão desse mesmo documento. Concluem que os utilizadores tendem a perder mais tempo na leitura dos documentos relevantes para o estudo efectuado do que os outros documentos. Porém é referido que este tempo varia consoante o tipo de artigo, se noticioso, académico ou de um jornal profissional.

Claypool, Le, Waseda e Brown [40] estudam a correlação entre a indicação explícita de interesse e diversos factores implícitos depreendidos do comportamento de navegação do utilizador num browser denominado Curious Browser. São tomados em conta o tempo total dispendido na visualização de um documento, o tempo perdido a mover o rato, o tempo de scroll e o número de cliques nesse documento. Neste estudo é mostrado que o tempo perdido na leitura e o tempo de scroll são bons indicadores de interesse, o número de cliques no rato não é um bom indicador e que existe uma boa correlação entre o tempo de movimentação do rato e o valor de importância dado ao documento. Contudo, os movimentos do rato sozinhos parecem ser só relevantes para a determinação dos documentos que têm menos importância para o utilizador. O contrário não acontece.

Goecks e Shavlik [32] desenharam uma rede neuronal para aprender os interesses dos utilizadores a partir dos cliques no rato, do scroll pelos documentos e da movimentação do rato. Concluíram apenas que a monitorização da actividade do rato não era suficiente para existir uma correlação com o interesse do utilizador em algum documento.

Chan [44], introduziu algumas métricas para estimar o interesse para cada página consultada. O interesse de um utilizador numa determinada página é calculado através de uma função que recebe 5 parâmetros de entrada:

- Frequência de visitas a essa mesma página;
- Booleano a indicar se a página foi marcada como sendo favorita;
- O tempo gasto na página normalizado pelo seu tamanho;
- O tempo passado desde que a página foi visitada pela última vez;

- O número de hiperligações visitadas sobre o número de hiperligações existente;

De seguida é preciso saber como são construídos e aplicados os modelos de ulilizadores no contexto WebIR.

2.3.3 Trabalho relacionado

Aktas, Nacar e Menczer [38] propõem a criação de um sistema de personalização baseado na alteração do algoritmo PageRank. Este passa por introduzir no cálculo do algoritmo informação relativa aos interesses do utilizador por certos domínios específicos. Focam-se por isso especificamente na análise das características do url de cada página. Na sua implementação foram escolhidas nove categorias das quais três são geográficas e as restantes seis relativas aos tópicos comercial (.com), militar (.mil), governamental (.gov), organizações sem fins lucrativos (.org), organizações da rede (.net) e educacional (.edu). Os utilizadores especificam os seus interesses sob a forma de um vector binário correspondente ao interesse ou não numa determinada categoria. Dado um vector de entrada, o sistema processa o valor PageRank para cada página baseado na comparação do domínio do url desta com os do vector representativo do perfil. Visto existirem nove categorias, existe a possibilidade de existirem 2^9 perfis diferentes que são computados off-line. Quando o utilizador utiliza o sistema, este indica em que categorias está interessado e o perfil daí resultante é utilizado para a escolha do PageRank personalizado já calculado que depois irá proporcionar a selecção dos documentos mais relevantes para a query do utilizador.

Tamine, Boughanem e Zemirli [37] inferem os interesses do utilizador a partir do histórico das suas procuras. Do seu ponto de vista, um perfil de utilizador expressa os seus interesses de longa duração e estes estão presentes no histórico das suas pesquisas sobre a forma das palavras mais utilizadas por entre os documentos considerados relevantes para cada pesquisa feita. A construção do perfil é um processo de duas etapas: primeiro, existe um intervalo de tempo em que é armazenada alguma informação. Após este intervalo de tempo, é possível inferir os interesses a partir de uma análise estatística dos termos mais frequentes entre os documentos e as queries. A segunda etapa, consiste na monitorização de uma possível actualização do perfil através de uma comparação entre os termos das queries que vão sendo introduzidas e dos termos que representam o perfil. Quando existe um afastamento considerável, o perfil é actualizado.

Masayasu Atsumi [39] recorre a uma metodologia completamente diferente para extrair os interesses do utilizador a partir das páginas Web que este consulte. Faz o uso de Algoritmos Genéticos. Representa um conjunto de documentos como um vector de palavras representam num vector space model [29] onde a cada termo é atribuída a frequência no documento. O interesse é representado como um cromossoma de tamanho variável cujos genes são pares dos termos e dos seus valores. A extracção dos interesses do utilizador pode ser vista como uma procura genética no espaço de documentos por um cromossoma óptimo, capaz de aproximar todos os documentos ao julgamento de uma porção destes por parte do utilizador.

Gulli e Ferragina [45] propõem um tipo de personalização que não faz uso de registo de documentos relevantes para o utilizador e consequentemente, que se afasta do actual problema de privacidade. A sua forma de personalização passa por permitir ao utilizador uma selecção dos tópicos existentes na árvore de conceitos gerada no processo de clustering por forma, a gerar uma nova lista (ordenada por relevância) de resultados cujos documentos estão mais associados a esses tópicos. Desta forma, é permitido ao utilizador adaptar a escolha dos tópicos de interesse de acordo com a sua subjectividade e interesses temporais dependentes.

Sieg, Mobasher e Burke [23] propõem uma procura na Web personalizada através do uso de ontologias como representação dos interesses do utilizador. Cada ontologia do perfil de utilizador é inicialmente uma instância de uma ontologia de referência constituída por múltiplos conceitos. A cada conceito no perfil do utilizador é atribuído um valor de interesse que inicialmente é 1. À medida que o utilizador vai interagindo com o sistema através da selecção ou visualização de documentos, a ontologia deste é actualizada e os valores para o grau de interesse de cada conceito são modificados através do desencadeamento de uma acção de propagação. Desta forma, o contexto com o utilizador é garantido e actualizado de acordo com o comportamento deste. A informação sobre os verdadeiros interesses do utilizador é colectada com a intervenção mínima nos padrões de navegação deste. Recorrem para isso a uma observação passiva do seu comportamento: a frequência de visitas a uma determinada página, o tempo dispendido na sua leitura, acções de impressão ou adição aos favoritos. A escolha dos conceitos para a criação da ontologia de referência ficou-se no uso do Open Directory Project, que é organizado numa hierarquia de tópicos e páginas relacionadas com esses mesmos tópicos. A ligação de termos a conceitos na ontologia, foi

feita a partir do cálculo dos termos mais relevantes existentes nas páginas de cada tópico tendo por base a medida $TFIDF$. Os resultados provenientes da procura são devolvidos ao utilizador por ordem crescente de um valor de importância dado a cada documento. Este valor é calculado com base da multiplicação de 3 outros valores: a distância do documento à query (Cosine), o valor de interesse para o melhor conceito encontrado para o documento e o valor da distância entre o conceito e a query.

Singh, Murthy e Gonsalves [51] propõem não um sistema de procura personalizado, mas sim, uma forma de encontrar o interesse do utilizador em tempo real. O objectivo passa por descobrir para uma determinada sessão de pesquisa os termos (palavras) de interesse para o utilizador, baseando-se numa análise aos snippets devolvidos e na supervisão dos documentos consultados pelo utilizador. Os autores referem que, se um utilizador introduz a query "Jaguar" e visualiza os documentos referentes a "car" então é porque muito provavelmente está interessado em "Jaguar car". Ao descobrir este tipo de informação é possível utilizá-la de várias formas como no auxílio à criação de um perfil de utilizador, ou em reordenar os resultados se a pessoa avançar para os próximos resultados da query.

A metodologia proposta para encontrar os referidos termos baseia-se em dividir o conjunto de resultados (T) em dois subconjuntos T_C e $T_{\overline{C}}$, que representam respectivamente os documentos visualizados e os não visualizados. O próximo passo, é encontrar o termo w que ocorre mais frequentemente nos resultados clicados $(P_c(w))$ e menos frequentemente nos resultados não visualizados $(P_{\overline{C}}(w))$.

$$d(w) = |P_c(w) - P_{\overline{C}}(w)| * Log\frac{2 - P_{\overline{C}}(w)}{2 - P_c(w)} \tag{2.1}$$

Calculando $d(w)$ através da equação , obtêm-se um valor entre $[-1, 1]$, sendo que quanto mais próximo de 1 estiver este valor, mais força tem o termo no conjunto de documentos que o utilizador consultou e mais representativo do interesse do utilizador.

Tanner [52] aponta para a criação de uma hierarquia de interesses do utilizador por forma, a melhorar a personalização dos motores de pesquisa. A sua abordagem para a reunião dos documentos que mostram os interesses do utilizador passa pela consulta do histórico das páginas favoritas deste. Após reunir o conteúdo de cada página, efectua uma limpeza do seu conteúdo através da remoção de palavras sem significado e posteriormente, efectua um processo de stemming. Cada documento passa a ser representado por frases cujos termos

são todos significativos. Destas frases são retirados todos os possíveis pares de palavras com um algoritmo próprio denominado ”Divisive Hierarchical Clustering”(DHC), cuja entrada é um grafo constituído por todas as palavras extraídas dos documentos e as relações entre elas (obtidas dos pares de palavras). A partir deste grafo é feita a criação da árvore de interesses. A raiz da árvore é um cluster com todas as palavras. Os elementos de cada nodo vão também eles ser clusters (filhos), os quais contêm subconjuntos de termos pertencentes ao seu pai. De modo a determinar os clusters filhos, inicia-se a parte divisiva do algoritmo que é a eliminação de algumas ligações entre os elementos com base no cálculo da pertença a um intervalo definido. O valor (para reordenar os resultados de uma pesquisa) para cada documento é calculado através do somatório do valor dado a cada termo existente na árvore e no documento.

Liu, Yu e Meng [28] efectuam a personalização das pesquisas na Web através do mapeamento das queries do utilizador em categorias. O processo desenvolve-se em três fases: primeiro, modelar e reunir a informação proveniente de uma sessão de pesquisa; segundo, construir o perfil de utilizador baseado no histórico das pesquisas e num perfil genérico construído a partir da hierarquia de categorias existentes no Open Directory Project; terceiro, deduzir as categorias que estarão associadas e uma nova query introduzida pelo utilizador. A primeira fase implica reconhecer o que é realmente relevante para o utilizador, a nível de documentos. São utilizadas as técnicas mais comuns como o tempo passado a ver o texto, o scroll e a movimentação do rato. Para cada query introduzida são guardadas duas categorias relacionadas bem como os documentos lidos e que pertencem a estas categorias. A informação sobre estas categorias pode ser obtida em alguns motores de pesquisa que relacionam certos documentos a categorias. A segunda fase, transforma esta informação numa árvore de interesses, que implica organizar as relações entre queries e documentos, bem como, categorias e documentos. Para tal, é feito o uso de duas matrizes (DT e DC), onde as linhas são os documentos e as colunas as palavras da query em DT, ou as categorias em DC. A cada posição da matriz DT, o valor atribuído tem por base o cálculo do $TFIDF$ da palavra. A partir de DT e DC computa-se a matriz M, cujos valores das linhas representam a relação entre os termos e as categorias. O mesmo tipo de processo é realizado para a construção da matriz M_g, representativa do perfil genérico obtida a partir das relações entre documentos e categorias existentes no ODP. A forma como são criadas estas matrizes, M e M_g, passa pela utilização do algoritmo Linear Least Squares Fit (LLSF), sobre as matrizes DT e DC de modo a aproximar $DT * M^T$ a DC.

Para uma aprendizagem adaptativa, é utilizado como alternativa o método de Rocchio. Por fim, de forma a calcular as categorias mais próximas de uma nova query introduzida pelo utilizador, é calculada a distancia com base na função de Cosine entre a query e cada categoria de M.

2.4 Proposta de trabalho

Como já referido neste capítulo, o processo de categorização permite agrupar em conjuntos elementos com propriedades semelhantes. Esta metodologia é aplicada em diversas disciplinas e a pesquisa na Web é uma das áreas em que o futuro do clustering parece mais promissor. O estado da arte propõe a categorização automática como solução para resolver os problemas relativos à ambiguidade das queries e ao crescente número de resultados relevantes devolvidos por um sistema IR. O motor de pesquisa recebe a query, encontra os resultados relevantes e numa fase final, processa estes de modo a encontrar similaridades entre eles de tal forma que possam ser colocados em vários grupos, devidamente identificados. Desta forma, o utilizador recebe uma lista de resultados sob a forma de uma estrutura com a qual pode interagir e assim centralizar a sua atenção num conjunto de documentos.

Com o mesmo objectivo de melhorar a performance dos sistemas WebIR, o estado da arte refere também a criação de modelos de utilizador. Estes modelos são adaptados aos sistemas WebIR e permitem fazer com que possa existir uma personalização dos resultados devolvidos. A construção de um modelo de utilizador baseia-se na extracção do conhecimento implícito existente nos documentos consultados pela pessoa ao qual o modelo se destina. Ao submeter estes documentos a um algoritmo de categorização, as particularidades destes permitem encontrar muita informação oculta e comum, muitas vezes representativa dos interesses do utilizador. As categorias geradas representam normalmente estas áreas de interesse, e a transformação da hierarquia num modelo de utilizador quantificado é feita de forma simples.

O processo referido anteriormente necessita do armazenamento dos documentos que são relevantes para o utilizador, fazendo com que passado algum tempo já seja necessária uma base de dados gigantesca para se poder efectuar a criação ou a actualização dos modelos de utilizador. Este tipo de metodologia talvez não seja a mais indicada para a personalização

dos resultados para os milhões de utilizadores que efectuam pesquisas na Web. Deste modo e após ter estudado profundamente o assunto decidimos seguir um caminho diferente para a criação dos modelos de utilizador.

Sabendo que os documentos guardados apenas servem para encontrar informação que seja comum entre os documentos, normalmente representativa dos interesses do utilizador, porque não efectuar o mesmo mas a partir de uma fonte diferente. Ao efectuar a categorização dos resultados para as pesquisas, é possível encontrar conhecimento que não era visível de forma explícita por entre os resultados. Estas categorias são úteis para o utilizador porque lhe podem ser chamativas e quando o são, normalmente deve-se ao facto de irem ao encontro dos seus interesses. Podemos então dizer que as categorias resultantes de uma pesquisa podem ser consideradas como áreas de interesse do utilizador. Resta agora encontrar quais as categorias que são realmente interessantes para cada pessoa. Num sistema trivial, poderíamos simplesmente gerar estas categorias, propô-las ao utilizador e as que este consultasse seriam os seus interesses. Contudo, numa situação real muitas vezes o utilizador não faz uso destas categorias, pois ele próprio não sabe muito bem a categoria onde está o que pretende. Cabe ao sistema encontrar estes interesses automaticamente.

Ao analisar o histórico de pesquisas do utilizador constituído apenas pelas queries que este introduziu e que lhe providenciaram documentos realmente de interesse, é possível construir uma árvore relacional de pesquisas. Esta árvore só tomará em conta pesquisas cujos termos ocorrem com alguma frequência em termos de introdução e de tempo, visto serem estes os factores mais representativos dos interesses reais do utilizador. Uma pesquisa feita por impulso pode ser feita de longe a longe, mas as pesquisas que são feitas de modo a encontrar informação relativa a interesses deste são muito mais frequentes. Quem está interessado em linguagens de programação, provavelmente irá pesquisar muito sobre vários tipos de linguagem ao longo do tempo, mas se necessitar de saber onde fica um determinado local, provavelmente só fará pesquisas relativas a isso num único dia. O tipo de estrutura daqui resultante irá certamente gerar um vasto número de ramificações, pois muitas pesquisas podem ser feitas com recorrência a palavras diferentes apesar de recaírem sobre uma mesma categoria. É nesta fase que pensamos ser possível dar um outro uso à categorização dos resultados das pesquisas. Ao termos ramificações de termos na árvore de pesquisas do utilizador, guardamos também para cada uma dessas pesquisas as categorias que lhe estão relacionadas. Deste modo, podemos encontrar as categorias mais comuns

entre as pesquisas que pertencem a cada ramo. Ao fazer isto para cada ramo, obteremos as categorias que melhor representam cada um (as mais frequentes e mais comuns a todos os elementos) e que muito provavelmente representarão os reais interesses do utilizador.

Também muito provavelmente, existirão repetições de categorias devido ao facto já referido de que pesquisas que têm termos diferentes muitas vezes recaem sobre a mesma categoria. Ao acontecer isto, estas categorias terão ainda mais importância no que toca a quantificar os interesses do utilizador.

Digamos por exemplo, (caso muito simples) um utilizador pesquisa bastante por shimano xt, shimano brakes, por mavic, mavic wheels e por truvativ, truvativ firex, truvativ brakes. A estrutura resultante irá ser do género apresentado na figura 2.8.

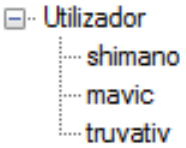

Figura 2.8: Perfil do utilizador sem categorização.

Ao ser incluída a informação sobre as categorias, a estrutura resultante será algo que já contem um dado mais específico que demonstra que todas estas pesquisas recaem sobre a categoria de bicicletas de montanha, ou seja, o utilizador está interessado neste tipo de modalidade (ver figura 2.9).

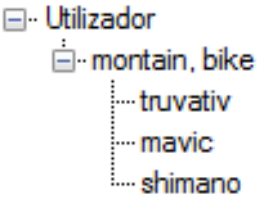

Figura 2.9: Perfil do utilizador com categorização.

A criação deste tipo de modelos de utilizador, leva a que o trabalho tenha de ser dividido em duas partes: a categorização dos resultados das pesquisas e a criação dos respectivos modelos de utilizador.

Capítulo 3

Clustering de Web snippets

A literatura mostra que a categorização de Web snippets resultantes de uma pesquisa é um processo que acresce dificuldades às correntes metodologias de categorização. Tal deve-se ao facto do conjunto de informação ser muito mais reduzido e apresentar particularidades muito interessantes como o conteúdo dos snippets ser muito dependente da query ou as frases existentes serem alvo de grande personalização como a repetição de palavras para dar ênfase ao conteúdo que muitas vezes nem é apelativo. É por isso, difícil classificar as palavras com recorrência à simples medida de $TFIDF$ e sem recorrer a outros serviços que providenciam uma avaliação da categoria morfológica das palavras. Em [45] Gulli e Ferragina propõem um sistema capaz de efectuar a categorização dos resultados de uma pesquisa com o recurso a algumas medidas que a nosso ver são realmente interessantes. Dentro do universo académico, esta é também a abordagem que conduz a melhores resultados. Dizemos dentro do universo académico porque se olharmos para o mundo comercial, existe um sistema que está na vanguarda e os seus resultados revelam-se de alta qualidade. Falamos do motor de pesquisa Vivíssimo (também referido como Clusty), um sistema já existente há alguns anos mas que ao longo do tempo tem vindo a evoluir muito positivamente. Infelizmente não existe literatura que faça uma análise mais profunda da metodologia utilizada por este sistema. Parte da metodologia proposta por Gulli e Ferragina serviu de inspiração para a criação do nosso algoritmo pois a sua proposta tem em conta para a criação das categorias, não o cálculo da similaridade entre documentos, mas sim a procura das melhores e mais frequentes expressões que existem entre os documentos de forma a criarem os grupos em torno destas mesmas expressões. O nosso algoritmo de certa forma também ambiciona efectuar o mesmo. Encontrar os termos mais fortes por entre o conteúdo dos documentos para depois agrupar os documentos que contemplam estas expressões. A

diferença principal é que não introduzimos conhecimento no nosso algoritmo como o uso de bases de dados classificativas (o Snaket usa o $DMOZ$) ou remoção de stop-words, nem adicionamos informação ao snippet de cada url. Dada a natureza do nosso algoritmo, que se baseia na selecção de termos fortes para a construção dos clusters e atribuição dos nomes ou ”labels”dos clusters também com base nestes termos, este foi denominado por ***CBL***, Clustering by Labels. É claramente um algoritmo do tipo label-derived.

3.1 Obtenção dos documentos relevantes para uma query a partir de um meta-motor de pesquisa

Não sendo o objectivo deste trabalho melhorar os motores de pesquisa relativamente à recolha dos documentos que mais estão relacionados com uma query, o processo de obtenção destes documentos passou pela criação de um meta-motor de pesquisa [25]. Um meta-motor de pesquisa é um sistema que faz uso de vários motores de pesquisa existentes para obter um conjunto de documentos mais forte no que toca à relação com a query. Isto acontece porque ao recolher os resultados de cada motor, o conjunto resultante irá conter uma maior variedade de assuntos visto que cada motor de pesquisa cobre uma área da Web que não é exactamente a mesma dos outros motores. Estes diferentes motores de pesquisa também aplicam diferentes algoritmos para a classificação dos documentos, logo os resultados também são muitas vezes diferentes. É também importante referir que a representação dos documentos deste meta-motor de pesquisa é feita através de snippets. Ou seja, cada documento é representado pelo seu respectivo url, o título e algumas frases mais importantes do documento (Ver figura 3.1).

Figura 3.1: Exemplo do snippet de um documento pertencente ao conjunto de resultados no motor de pesquisa Google para a query “apple”.

O nosso meta-motor de pesquisa recorre ao uso de três dos mais conhecidos motores de pesquisa: o Google, o Yahoo e o MSN Live cuja última remodelação levou à alteração do seu nome para Bing. Para cada motor de pesquisa são obtidos os primeiros 50 a 100 resultados,

conforme o valor escolhido pelo utilizador. Foi estabelecido este intervalo pois achamos que os resultados a partir da posição 100 já não são úteis para o que se pretende visto que é muito raro um utilizador navegar até ao resultado número 100 para verificar se existe algum conteúdo de interesse. O mínimo imposto deve-se ao facto de ser necessário uma quantidade relativa de documentos para que possa existir uma boa geração de categorias.

Visto ser possível que os diferentes motores de pesquisa possam devolver alguns documentos iguais, é necessário calcular o valor de importância que estes vão ter relativamente aos outros documentos. Essa importância é calculada a partir do ranking dado ao documento por cada motor. O valor de ranking atribuído pelo nosso meta-motor de pesquisa para cada documento dá mais importância aos resultados do Google tal como definido na equação 3.1.

$$R(d) = 0.4 * R_g(d) + 0.3 * R_m(d) + 0.3 * R_y(d) \tag{3.1}$$

Onde $R(d)$ representa o ranking dado ao documento d, $Rg(d)$ é o ranking dado pelo Google ao documento, $Rm(d)$ é o ranking dado pelo MSN ao documento e $Ry(d)$ é o ranking dado pelo Yahoo ao documento.

Quando um ou mais motores não devolvem o mesmo documento, o seu peso é dividido para o peso de cada motor que possa ter devolvido o resultado de forma equitativa como na equação 3.2 onde o documento d não é devolvido pelo motor de busca Google.

$$R(d) = 0.5 * R_m(d) + 0.5 * R_y(d) \tag{3.2}$$

A informação relativa ao título e às frases mais importantes de um documento que é devolvido por vários motores de pesquisa é unida numa estrutura que representa o conteúdo do documento. Ou seja, os vários títulos são unidos numa única representação de texto bem como numa outra variável são unidas todas as frases mais importantes desse documento. Mais para a frente iremos explicar porque não faz diferença nesta fase verificar se a informação relativa ao mesmo documento e proveniente de diferentes motores, não é igual.

3.2 Aglomeração dos snippets por domínio e pré-processamento dos dados

Esta é uma das fases mais importantes para que o processo de categorização possa ter como entrada boas palavras. Para tal é necessário primeiramente agrupar todos os resultados de acordo com o domínio do url. Isto vai ser útil numa fase posterior para poder analisar as várias frases que possam existir em documentos que pertençam ao mesmo domínio.

O pré-processamento dos dados implica a remoção de tudo o que é lixo numa frase. Por lixo entende-se as tags utilizadas por certos motores de pesquisa para realçar algumas palavras, o uso de caracteres especiais para representar conteúdo html, simbologias que muitos utilizadores usam para personalizar as suas frases mas que não introduzem nenhum conhecimento e alguns caracteres especiais cuja remoção não altera a estrutura das frases.

Após esta primeira filtragem efectua-se uma análise à estrutura do texto onde, com recurso a uma etiquetagem definida por nós, são marcadas as secções onde uma frase acaba, onde existe uma vírgula ou onde existem cortes (frases que dada a sua extensão não são apresentadas na sua totalidade no snippet, sendo substituído o resto da frase por três pontos). Cada sequência de caracteres é analisada e conforme a sua constituição é-lhe atribuído um marcador definido na tabela 3.1.

Marcador	**Descrição**
%p	Final de frase
%d	Delimitador na frase
≺o≻	Conteúdo da palavra analisável
≺a≻	Acrónimo
≺n≻	Nome
≺t≻	Conteúdo descartável
≺d≻	Data
≺m≻m	Interrupção da frase

Tabela 3.1: Descrição dos marcadores utilizados no pré-processamento dos snippets.

Estes marcadores servem para que numa fase seguinte, certas propriedades que decidimos avaliar nas palavras possam estar marcadas nos snippets. Algumas destas propriedades são descritas de seguida:

- *Classificação de uma palavra como nome:* uma palavra é classificada com sendo representativa de um nome quando esta começa com uma maiúscula, sendo todos os restantes caracteres minúsculos, quando não está no início de uma frase, e quando numa janela de p palavras em torno da que está em avaliação, não existem mais que n % de palavras também elas podendo ser classificadas como nomes. Tal restrição deve-se ao facto do conteúdo dos snippets ser muitas vezes alvo de uma má escrita onde por exemplo uma frase é composta toda ela por palavras que começam por maiúsculas.

- *Classificação de uma palavra como acrónimo:* o processo é mesmo do que para a classificação de nomes e com as mesmas restrições. A única diferença é que a palavra tem de ser composta na sua totalidade por letras maiúsculas.

- *Classificação de uma palavra ou expressão como não útil:* todas as sequências de caracteres que tenham mais do que um hífen, que representem hiperligações ou cujo seu conteúdo esteja representado por caracteres não alfa-numéricos com a excepção dos hífenes.

Na figura 3.2, vemos o tratamento de um snippet após o pré processamento e já preparado para ser analisado numa fase posterior.

```
Snippet sem pré-processamento:

The <b>apple</b> is the pomaceous fruit of the <b>apple</b> tree, species
Malus domestica in <br>  the rose family Rosaceae. It is one of the most
widely cultivated tree fruits <br>  <b>...</b>


Snippet com pré-processamento:

<o>the <o>apple <o>is <o>the <o>pomaceous <o>fruit <o>of <o>the <o>apple
<o>tree %d <o>species <n>malus <o>domestica <o>in <o>the <o>rose <o>family
<n>rosaceae %p <o>it <o>is <o>one <o>of <o>the <o>most <o>widely
<o>cultivated <o>tree <o>fruits <m>m
```

Figura 3.2: Comparação entre um snippet com e sem pré-processamento.

3.3 Extracção de características das palavras

Com o conteúdo dos snippets já devidamente etiquetado e agrupado por domínios passa a ser possível processar todos os textos e reunir a informação relativa a cada palavra existente. Esta informação vai ser utilizada para calcular o nível de importância de cada palavra. Além de ser guardada a frequência com que cada palavra ocorre, o número de vezes que é considerada um nome, um acrónimo, o número de urls em que ocorre, são efectuadas contagens mais relevantes para o cálculo da sua importância. Estas têm a ver com a co-ocorrência da palavra com outras palavras situadas imediatamente adiante ou atrás (caso existam claro). Após muito tempo a analisar as características dos snippets chegámos à conclusão de que poderíamos obter a qualidade de uma palavra a partir do número de palavras diferentes que ocorrem com a palavra em questão. As palavras que não introduzem conhecimento relacionam-se com um número muito variado de palavras (por ex: "o carro", "o gato", "ganhou o", "o sol"). Pelo contrário, palavras que induzam conhecimento na generalidade não se associam a um número tão grande de outras palavras. Ao sabermos o número total de palavras que ocorreram nas laterais da palavra em análise e o número de palavras diferentes deste conjunto, é possível quantificar o nível de interesse que essa palavra terá.

Dado que muitas vezes os motores de pesquisa utilizados no nosso meta motor devolvem resultados pertencentes ao mesmo domínio, sendo o seu conteúdo integralmente (ou mesmo exactamente) igual apenas variando o url, existiu a necessidade de adaptar o nosso algoritmo a esta realidade. A ignorância deste facto iria viciar os resultados no sentido de que ao serem repetidas muitas frases iria acontecer que as palavras iriam ocorrer com um conjunto muito pouco variado de palavras e com uma frequência elevada. Isto resultaria numa boa classificação da palavra aos olhos do nosso algoritmo algo que não seria verídico. O excerto que se segue na figura 3.3 de uma lista de resultados para a query "cars" demonstra bem este caso.

```
New &amp; Used Cars for Sale, Auto Dealers, Car Reviews and Car Finance Advice
Search 2.6 million new &amp; used car listings, price a new car, get a dealer quote, read
expert reviews, or sell your car at thousands over trade-in.
4 - http://www.cars.com/go/index.jsp?aff=ithaca

New &amp; Used Cars for Sale, Auto Dealers, Car Reviews and Car Finance Advice
Search 2.6 million new &amp; used car listings, price a new car, get a dealer quote, read
expert reviews, or sell your car at thousands over trade-in.
5 - http://www.cars.com/go/index.jsp?aff=dmoines

New &amp; Used Cars for Sale, Auto Dealers, Car Reviews and Car Finance Advice
Search 2.6 million new &amp; used car listings, price a new car, get a dealer quote, read
expert reviews, or sell your car at thousands over trade-in.
7 - http://www.cars.com/go/index.jsp?aff=jconline

New &amp; Used Cars for Sale, Auto Dealers, Car Reviews and Car Finance Advice
Search 2.6 million new &amp; used car listings, price a new car, get a dealer quote, read
expert reviews, or sell your car at thousands over trade-in.
8 - http://www.cars.com/go/index.jsp?aff=thetimes

New &amp; Used Cars for Sale, Auto Dealers, Car Reviews and Car Finance Advice
Search 2.6 million new &amp; used car listings, price a new car, get a dealer quote, read
expert reviews, or sell your car at thousands over trade-in.
9 - http://www.cars.com/go/index.jsp?aff=wfmy

New &amp; Used Cars for Sale, Auto Dealers, Car Reviews and Car Finance Advice
Search 2.6 million new &amp; used car listings, price a new car, get a dealer quote, read
expert reviews, or sell your car at thousands over trade-in.
11 - http://www.cars.com/go/index.jsp?aff=batcreek
```

Figura 3.3: Conjunto de snippets para a query “cars” cujo conteúdo é exactamente igual.

É este o motivo pelo qual os resultados foram agrupados por domínios, para que o algoritmo efectue a extracção da informação através da análise das sequências de palavras que são distintas entre os urls de cada domínio. Cada uma destas sequências é também adicionada a uma lista que depois servirá para a extracção de expressões compostas. Na tabela seguinte 3.2 são apresentadas todas as características que são guardadas para cada palavra. Na figura 3.4 está o pseudo-código do algoritmo que permite a extracção desta informação por entre todos os resultados do meta-motor de pesquisa.

```
Criar lista de expressoes (E);

Por cada dominio
{
    Criar lista de expressoes_temp (Et);

    Por cada url de um domínio
    {
        Por cada expressao (exp) do url
        {
            se(singular) -> incrementar F e S;

            se(composta && !existe em Et) -> processar cada palavra (w) e analisar as associadas;
                                             incrementar F(w), WIL(w) ou WIR(w), WDL(w) ou WDR(w);
                                             se(w) = <n> -> incrementar F_Name(w);
                                             se(w) = <a> -> incrementar A_Name(w)
                                             adicionar exp a Et;
        }
    }

    Adicionar Et a E;

}
```

Figura 3.4: Pseudo-código para a extracção de informação das palavras.

Característica	Descrição
W	Representação em caracteres da palavra
query_Word	Indicação se a palavra existe na query
F_Name	Número de ocorrências em que a palavra é considerada um nome
F_Acron	Número de ocorrências em que a palavra é considerada um acrónimo
S	Número de ocorrências em que a palavra está isolada
F	Frequência da palavra
U	Número de Urls em que a palavra ocorre
WIL	Número de palavras contadas imediatamente à esquerda da palavra
WIR	Número de palavras contadas imediatamente à direita da palavra
WDL	Número de palavras diferentes contadas imediatamente à esquerda da palavra
WDR	Número de palavras diferentes contadas imediatamente à direita da palavra
W_Class	Raiz da palavra (quando é utilizado stemming)

Tabela 3.2: Características analisadas numa palavra.

Visto que durante a execução do algoritmo é necessário estar constantemente a aceder ao objecto que contem a informação relativa a cada palavra, foi criada uma estrutura que permite encontrar muito rapidamente esse objecto. Esta estrutura consiste numa árvore de nível 3 (ver figura 3.5) onde cada ramo e sub-ramo contem os caracteres do alfabeto. No final de cada ramo da árvore, existe um apontador para uma lista que contem as referências dos objectos que contêm a informação das palavras cujo nome começa pelas letras desta ramificação. É assim possível saber muito mais rapidamente para um palavra onde está a estrutura que a representa e deste modo poder processar todos os dados em tempo real (obrigatoriedade para os sistemas de WebIR).

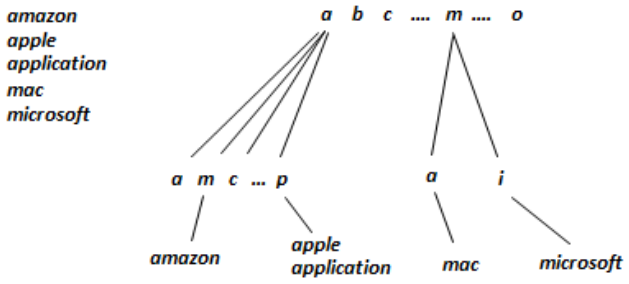

Figura 3.5: Estrutura de indexação das palavras em árvore.

3.4 Cálculo do valor de importância de uma palavra

Após nas fases anteriores terem sido reunidos todos os dados relativos a cada palavra, o valor de importância final dado a cada uma destas é calculado com base no valor das seguintes quatro propriedades:

Propriedade W_1***:*** Se um termo aparece sozinho num segmento de texto, quer seja separado dos restantes termos por uma vírgula, um ponto ou outro separador, então é muito provável que esse termo tenha significado.

$$W_1(w) = \frac{A(w)}{ln(F(w))} \tag{3.3}$$

Onde w é qualquer termo, $A(w)$ é o número de ocorrências em que w aparece sozinha e $F(w)$ é a frequência do termo w.

Propriedade W_2***:*** Quanto maior for o número de termos que co-ocorrem com qualquer termo w tanto no contexto do lado esquerdo ou do lado direito, então menos importante esse termo será.

$$W_2(w) = \frac{WIL(w) + WIR(w)}{2 * F(w)} \tag{3.4}$$

Onde w é o termo, $WIL(w)$ e $WIR(w)$ são o número de termos que co-ocorrem imediatamente nos lados esquerdo e direito do termo w respectivamente e $F(w)$ é a frequência do termo w.

Propriedade W_3***:*** Quanto maior for o número de termos diferentes que co-ocorrem com o termo w em ambos os seus lados esquerdo e direito comparativamente ao número total de termos existentes nos seus lados esquerdo e direito respectivamente, então, provavelmente menos importância terá essa palavra.

$$W_3(w) = \left[(\frac{WDL(w)}{WIL(w)} + \frac{WDL(w)}{FH(w)}) * \frac{WIL(w)}{F(w)}\right] + \left[\frac{WDR(w)}{WIR(w)} + \frac{WDR(w)}{FH(w)}) * \frac{WIR(w)}{F(w)}\right] \tag{3.5}$$

Onde w é qualquer termo, $WDL(w)$ e $WDR(w)$ são respectivamente o número de termos diferentes imediatamente ao lado do termo w e $FH(w)$ é igual ao $Max[F(w)]$, para todos os termos w.

Propriedade W_4***:*** Se um termo aparece designado pelo processo de pré-filtragem como sendo um nome ou um acrónimo com uma certa frequência num conjunto de texto, então é muito provável que esse termo tenha significado.

$$W_4(w) = \frac{I(w)}{ln(F(w))} \tag{3.6}$$

Onde w é qualquer termo, $I(w)$ é o valor da melhor representação do termo como sendo acrónimo ou nome e $F(w)$ é a frequência do termo w.

Finalmente, baseado nestas quatro propriedades é possível atribuir um valor de importância a w.

$$\begin{cases} W_2(w) * W_3(w), & \text{se} \quad W_1(w) < 0.5 \\ \frac{W_2(w)*W_3(w)}{1+W_1(w)}, & \text{se} \quad W_1(w) \geq 0.5 \wedge W_4(w) < 0.5 \\ \frac{W_2(w)*W_3(w)}{1+W_1(w)+W_4(w)}, & \text{se} \quad W_1(w) \geq 0.5 \wedge W_4(w) \geq 0.5 \end{cases}$$

Onde $W(w)$ é o valor da importância do termo que quanto mais baixo for, mais importante é o termo.

3.5 Geração das categorias com base nas palavras mais fortes

Após todas as palavras importantes terem sido identificadas, elas vão, como já havia sido referido, representar um papel crucial no processo de categorização dos resultados visto que este tem por base a escolha destas melhores palavras. O algoritmo é executado nos três seguintes passos: criação dos pólos, unificação e absorção, escolha de um nome identificador para o conteúdo do cluster.

Criação dos pólos: é necessário inicializar o algoritmo para que sejam escolhidos os termos mais representativos. Com esse propósito, todas as palavras que se situem entre as

α primeiras posições da lista ordenada de palavras mais importantes para cada url e que existam em mais de dois urls, são propostas para centros iniciais de clusters (pólos). A cada pólo, é associado uma lista de urls. Um url é adicionado à lista se contém a palavra que está no pólo entre as β primeiras posições da lista de palavras ordenada por grau de importância para esse url. Particularmente, o valor β permite controlar o número de urls que é adicionado a cada pólo visto que um β baixo vai produzir clusters mais pequenos e um β maior irá levar à existência de mais resultados dentro dos clusters. Na corrente implementação do algoritmo o valor de β é igual a 3 pois é um valor que permite um número aceitável de resultados por cada categoria e também que um url só possa existir no máximo em 3 categorias. O valor de α é igual a 1 pois desta forma só são criados pólos em torno das palavras mais importantes para cada url. No capítulo 5 (discussão dos resultados) são apresentados os resultados obtidos com valores diferentes para α.

União e Absorção: o próximo passo pretende iterativamente unir clusters que contenham urls similares. Para este propósito definimos dois tipos de aglomeração: (1) a união, quando dois clusters contêm um número significante de urls comuns e partilham de um tamanho semelhante em termos do número de urls, (2) a absorção, quando partilham muitos urls mas o tamanho dos clusters é bastante diferente, sendo um bem mais numeroso que o outro. Como consequência definimos duas proporções: P_1, o número de urls comuns aos clusters dividido pelo número de urls do cluster mais pequeno e P_2, o número de urls do cluster mais pequeno dividido pelo número de urls do cluster maior. O algoritmo que se segue é iterado.

Por cada cluster, P_1 é calculado relativamente a todos os outros clusters. Depois para cada par de clusters, se P_1 é mais elevado do que o valor de uma constante μ, então avalia-se o valor de P_2 entre esses dois clusters. Se P_2 é maior do que uma constante θ, então o par de clusters é adicionado à lista de união, caso contrário é integrado na lista de absorção. Assim que todos os clusters sejam mapeados, ambas as listas de união e absorção são tratadas. A lista de uniões é tratada primeiro como se descreve no paragrafo seguinte. Os melhores valores encontrados para μ e θ através de experiências realizadas são respectivamente 0.7 e 0.8. Novamente, no capítulo 5 são mostrados resultados com diferentes valores para μ e θ.

Para cada par de clusters na lista de uniões, os dois clusters pertencentes as estes pares são juntos no cluster com o maior valor de importância da palavra $W(w)$ que é representativa do seu centro. A cada passo deste processo, os índices dos clusters são substituídos e os

clusters unidos são removidos da lista de uniões de forma a manter uma lista actualizada de clusters. Após todos os elementos desta lista terem sido processados, trata-se da lista de absorções.

Iterativamente selecciona-se o par de clusters que contém um qualquer cluster que não possa ser absorvido por nenhum outro cluster existente na lista. Quando encontrado, este cluster absorve o cluster que faz par com ele. Os índices dos clusters são novamente actualizados e os clusters inutilizáveis são removidos. Ambas as listas são actualizadas e o processo repete-se, dando a possibilidade da geração de uma categorização do tipo plana (primeiro passo do algoritmo), ou a uma categorização hierárquica (todos os passos do algoritmo). De realçar que o algoritmo permite a um url existir em diversos clusters, podendo assim o algoritmo ser classificado de Soft Clustering permitindo o overlap.

Escolha de um nome identificador para o conteúdo do cluster: através da união e absorção, cada cluster pode conter mais do que uma potencial palavra para descrever o seu conteúdo. Contudo, também pode acontecer que os urls do cluster contenham um outro tipo de palavras (expressões compostas) que providenciem um nome mais interessante para o cluster. Como consequência, estas expressões, extraídas após a fase de cálculo do grau de importância das palavras com o uso do algoritmo proposto em [35], são comparadas às frequências das palavras simples que representam o cluster. No caso da frequência de uma palavra composta ser mais elevada que um valor ϱ imposto de proporcionalidade com a frequência da melhor palavra simples candidata a nome do cluster, então será a expressão composta a identificar o conteúdo do cluster. Caso contrário será a palavra simples candidata a mais forte.

3.6 Comparação com o estado da arte

O CBL foi desenvolvido com o intuito de ser uma metodologia completamente independente e capaz de agrupar os documentos com base nas palavras mais fortes. Ao ser aprofundado o conhecimento sobre o clustering de documentos, podemos constatar que as abordagens do tipo document-derived como em Lipai [22] e Geraci [27], tendem a produzir categorias cujo seu conteúdo diversas vezes está afastado do nome atribuído à categoria. Tal acontece porque são algoritmos que se baseiam à priori no cálculo da similaridade entre

documentos e depois no agrupamento dos que são mais similares. Contudo neste tipo de abordagem, por forma a criar conjuntos com um número considerável de documentos, existe uma menor restrição ao nível da similaridade necessária para adicionar um novo elemento a um conjunto. Isto faz com que certos elementos sejam adicionados por terem similaridade com um conjunto de termos existente realmente no conjunto mas que nem é interessante ou tão frequente quanto outros que depois são seleccionados para representarem o conjunto. Quando a similaridade necessária é mais exigente, os grupos formados são realmente fortes mas regra geral muito pequenos o que pode levar á existência de muitos grupos pequenos que quando apresentados a um utilizador acabarão por fazer com que este se perca à procura do pretendido.

Ao estudarmos o trabalho de Gulli e Ferragina [45] achamos que a sua ideia ia mais de encontro ao que idealizamos como um bom sistema de categorização: agrupar os resultados em torno de expressões que são consideradas fortes. A sua abordagem é contudo muito dependente de informação adicional para o enriquecimento dos snippets e para a classificação da qualidade dos termos. Além disso aplicam filtros que removem as palavras vazias pré-estabelecidas. Esta dependência de diversos outros sistemas, foi algo que sempre ambicionamos ultrapassar e por isso desta metodologia serviu de orientação no que tocou à escolha da metodologia label-derived. A fase final de agrupamento dos resultados com base nas palavras mais importantes, absorção e união de categorias, foi também algo influenciada por esta metodologia.

Contudo a fase mais importante do nosso algoritmo está na atribuição de um valor de importância às palavras que compõe os snippets. Um trabalho bastante similar ao que propomos é o de Ventura J. e Silva J. [34] que também qualifica as palavras presentes num conjunto de documentos com um valor de relevância. Este valor é como no nosso algoritmo, obtido através da análise estatística dos sucessores e precedentes para cada palavra. A ideia é a mesma, palavras que co-ocorrem com um conjunto mais limitado de palavras são normalmente mais indicadoras de conhecimento e qualidade. O cálculo da importância de uma palavra é dada pela média do valor obtido para os valores de ocorrência com os seus sucessores e precedentes. Contudo no caso específico do português, aplicam uma medida diferente pois a estrutura da língua sugere ser normal uma maior variação nas palavras que precedem a avaliada e deste modo isso é levado em consideração. Para reforçar o valor de importância dado a uma palavra, sugerem também a introdução de um peso conforme

o número de sílabas que a palavra apresente. Palavras com uma sílaba ou muitas sílabas são consideradas como menos importantes. Nesta abordagem também é descrito uma nova metodologia denominada de "Ilha" que ambiciona calcular a fronteira entre as palavras que se podem considerar relevantes e não relevantes.

A nossa implementação para o cálculo do valor de importância de uma palavra difere um pouco da abordagem referida. A forma como relacionamos o número de palavras diferentes e o número de palavras existentes para as palavras que precedem e sucedem a palavra em avaliação é diferente. A ideia com que ficamos das nossas experiências foi que uma palavra pode ser importante se apenas o número de palavras diferentes que esteja num dos lados da palavra em análise seja baixo. Se a palavra ocorrer com poucas palavras nas suas fronteiras, e estas variarem bastante, a palavra deve mesmo assim obter uma certa importância. É isso que pretendemos quantificar com as funções propostas. Ao longo do desenvolvimento do nosso algoritmo também pensamos introduzir um peso que penaliza-se palavras com um número de sílabas muito baixo contudo verificamos que existem bastantes palavras (em qualquer linguagem) que seriam penalizadas injustamente. Decidimos por isso acrescentar como factores indicativos da importância de uma palavra a sua ocorrência isoladamente, como acrónimo e como nome. De salientar que a palavra não é qualificada por exemplo como acrónimo apenas por ter todas as palavras em maiúsculas. É também necessário que numa certa janela de palavras existam algumas sem estar neste formato. A não verificação deste caso, levaria a que muitos títulos ou outras porções dos snippets escritos totalmente em maiúsculas, fossem considerados acrónimos, quando não o eram. O algoritmo proposto é por isso uma mistura de diversos conceitos com o objectivo principal de agrupar informação em torno de palavras fortes e sem dependências de outras ferramentas à excepção das fontes para recolha de resultados em forma de snippets.

Capítulo 4

Criação de modelos de utilizador

Para que possa ser criado um modelo de utilizador é necessário reunir informação que seja realmente relevante para este. Existem duas formas de reunir esta informação, implicitamente ou explicitamente. O problema de reunir dados que sejam de interesse para uma pessoa sob forma automática, ou seja implicitamente, reside na escolha dos indicadores que permitem saber que informação é realmente relevante para este. Por outro lado o problema de reunir informação que o utilizador indica explicitamente como sendo relevante, é o de conseguir a sua participação.

A nossa escolha passou pelo uso das duas mas, com muito mais uso da informação recolhida sem percepção por parte do utilizador de modo a não o afastar do sistema. Como já referido na proposta de trabalho a metodologia escolhida para o nosso algoritmo de criação de modelos de utilizador passa, numa primeira fase, pela criação de uma árvore relacional das queries relevantes introduzidas pelo utilizador. Dizemos relevantes, porque é muito frequente fazermos uma pesquisa para a qual nenhum dos resultados nos é chamativo, dando origem a uma das duas situações: a reformulação da query, ou a desistência. Devido a isto existe a necessidade de saber que queries resultaram numa experiência positiva para o utilizador. Muitos estudos [41] [33] [40] [32] [44] revelam que os factores implícitos indicadores do interesse de um utilizador num documento são o tempo que este passa a ver o documento e as acções de movimentação do cursor. A nossa opção para saber se uma query foi ou não produtiva para um utilizador passou pelos seguintes indícios:

- Abertura de um ou mais documentos;

- Em pelo menos um dos documentos abertos o utilizador passou mais de s segundos na sua leitura.

Se uma query introduzida por um utilizador satisfaz estes requisitos, então é adicionada ao histórico da sessão de pesquisa do utilizador.

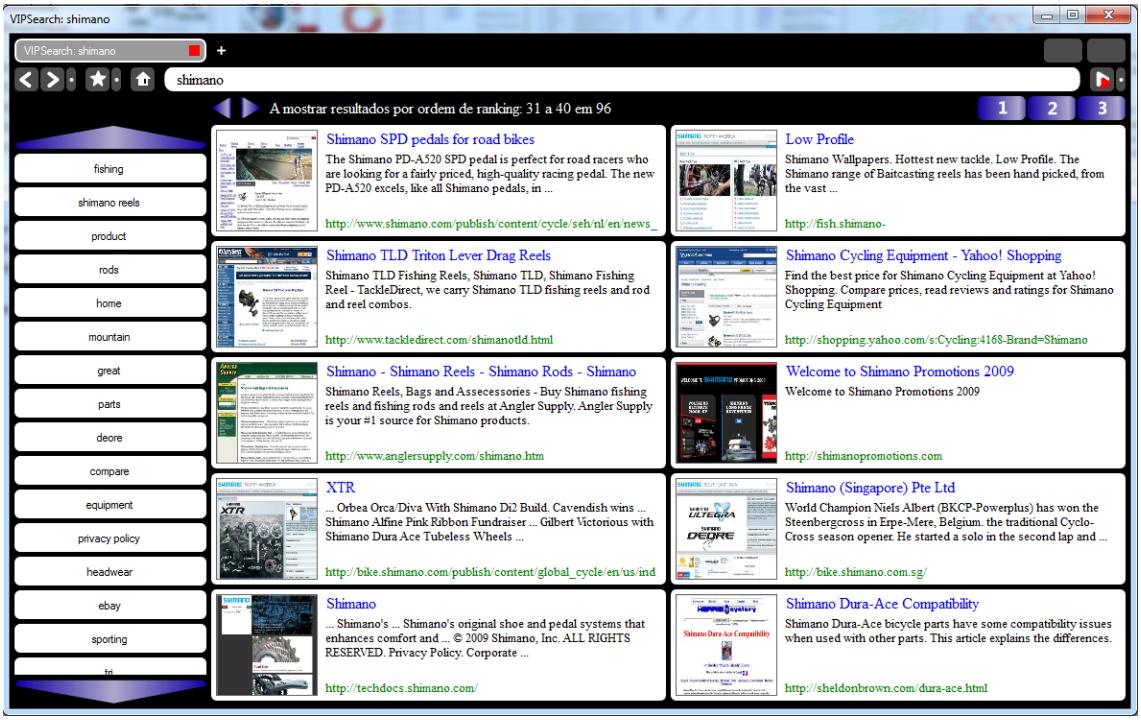

Figura 4.1: Aplicação VIPWeb.

O termo sessão que ainda não havia sido referido anteriormente, pretende denominar todo o conjunto de pesquisas efectuado desde a abertura até ao encerramento da aplicação criada para servir de plataforma ao desenvolvimento deste trabalho. Esta aplicação tem as funcionalidades básicas de um browser visto que permite ao utilizador navegar por entre o conteúdo dos sites mas, também providencia o acesso ao meta-motor de pesquisa desenvolvido que permite obter os documentos mais relevantes para uma query bem como a listagem das categorias que estão associadas a estes resultados. A aplicação foi denominada de VIPWeb e é apresentada na figura 4.1. Para além da aplicação VIPWeb, é também possível obter dados a partir do VIPAccess [26], uma aplicação para mobile que visa unificar diversas funcionalidades de pesquisa e apresentar a informação sob forma estruturada e clarificada (ver figura 4.2 ponto 1). Em cada sessão são guardadas as queries relevantes do utilizador bem como as categorias associadas a estas queries e que foram obtidas a partir

do algoritmo de categorização descrito no capítulo anterior. São guardadas também as categorias explicitamente escolhidas pelo utilizador para consultar documentos que revelaram terem sido de interesse para ele (ver figura 4.2 ponto 2). No final de uma sessão toda a informação é enviada a um servidor criado para este efeito e onde existem tabelas para armazenar os diferentes dados (ver figura 4.2 ponto 3).

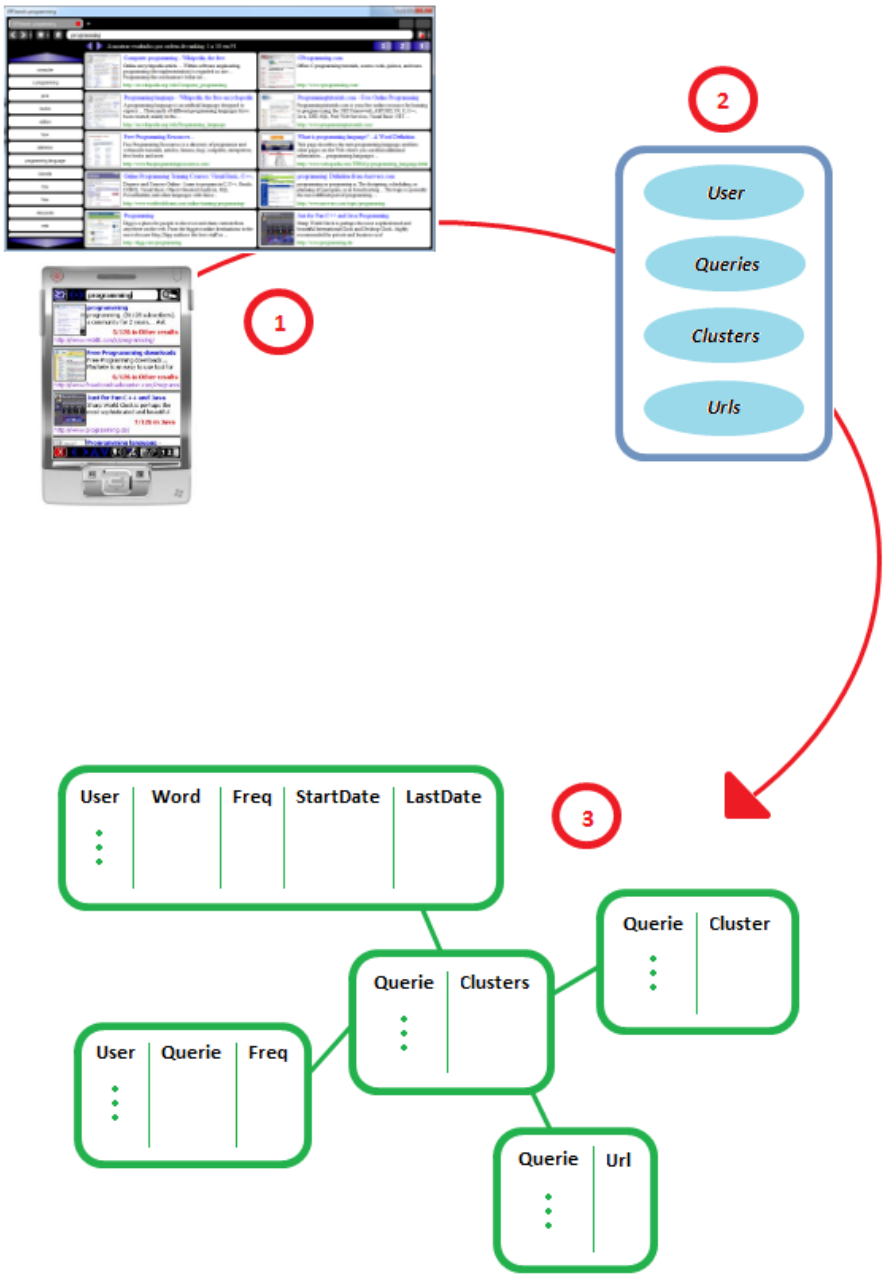

Figura 4.2: Interação entre os componentes do sistema.

4.1 Criação da árvore relacional

Como já havia sido referido, a metodologia escolhida para a criação dos modelos de utilizador passa pela extracção do conhecimento existente nas queries. Na fase anterior foi reunida toda a informação possível sobre estas queries mas, de modo a encontrar os interesses de longo prazo do utilizador, é necessário filtrar este conjunto. Ou seja só são consideradas para a criação da árvore as queries que satisfaçam as seguintes propriedades:

- A frequência de pelo menos um dos seus termos seja maior que fTm (frequência mínima do termo). Tomamos $fTm = 10$ como valor de base.
- O intervalo de tempo decorrido entre o uso pela primeira vez de um termo e o uso pela última vez dele seja maior que iTm (intervalo mínimo de uso do termo). Tomamos $iTm = 15$ dias como valor de base.
- O número de termos da query não exceda o nTm (número máximo de termos). Foi necessária esta restrição pois uma query com mais de um certo número de termos é muito específica e não ajudaria em muito na criação de um perfil que pretende ser algo abrangente. Este número é o limitador do nível de profundidade da árvore. Definimos $nTm = 4$ sabendo que a maioria das queries tem entre 2 a 3 palavras.

De modo a tornar mais eficiente a selecção das queries que servirão para a construção da árvore de interesses, é criada uma tabela para cada utilizador onde são guardadas as queries aceites e para cada termo utilizado, são guardadas a frequência, data de início e última utilização desse termo bem como os clusters e urls que lhe estão associados.

Com o uso desta tabela é possível filtrar as queries que servirão de entrada ao algoritmo que as ramificará. Contudo, existem ainda dois problemas. A árvore que se pretende criar para servir de modelo de utilizador visa nos seus primeiros níveis de profundidade conter os termos mais genéricos e só depois apresentar os mais específicos. É por isso necessário que os termos sejam classificados com um valor representativo do grau de generalidade. Este é um dos problemas. O outro, prende-se com a própria definição da palavra generalidade. Em certos casos, é fácil saber que um termo é mais genérico do que outro. Por exemplo consideremos os termos: “automóvel” e “audi”. É fácil saber que o termo “automóvel” é mais genérico do que “audi”. Agora adicionemos à lista destes termos a palavra “análise”. Sabemos que “análise” pode estar associado a todos os termos que se relacionam com “automóvel” e ainda a outras categorias sendo por isso mais genérico. Contudo na definição

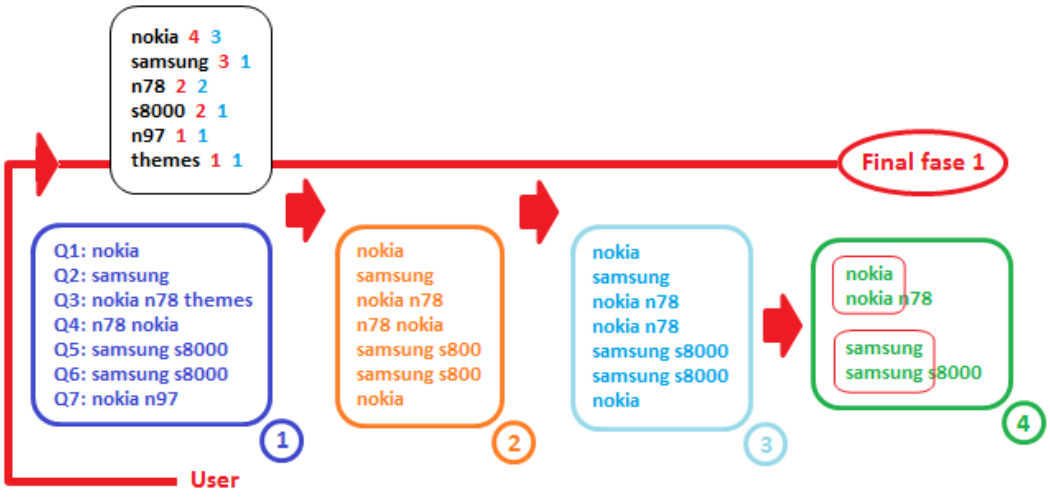

Figura 4.3: Estruturação dos termos das queries.

de um perfil de utilizador, será uma palavra que deva estar nos primeiros níveis a representar os seus interesses mais genéricos, ou será uma palavra representativa de uma particularidade do utilizador? Na implementação deste trabalho, ao calcular a generalidade das palavras, esta obterá um grau indicativo de uma forte generalidade. Contudo ao refinar a árvore gerada esta palavra passará para os níveis mais internos da árvore visto que o conhecimento que introduz sobre o utilizador só é relevante se estiver associada a outras palavras.

O método utilizado para calcular a generalidade dos termos, baseia-se na interligação entre as palavras que compõem as queries. Ao longo da utilização do sistema uma matriz quadrada de n termos (em que n está em constante actualização devido ao número de termos usados aumentar ao longo do tempo) marca a existência de um ligação entre cada par. Esta interligação é marcada sempre que uma query é composta por mais de um termo e se a query contém por exemplo: “A”, “B” e “C” então, são marcadas na matriz as ligações “A - B”, “A - C” e “B - C”. A generalidade de um termo pode ser calculada com base no número de termos diferentes a que este está interligado, permitindo assim saber se um termo é mais ou menos genérico que outro termo da matriz.

A primeira etapa da criação da árvore relacional é efectuar a remoção de termos das queries cujos valores impostos de fTm e nTm não são atingidos. Ou seja para cada query existente na tabela criada anteriormente, só são mantidos os termos aceites (ver figura 4.3 ponto 1 para 2).

Com o uso da matriz de generalidade, passa a ser possível efectuar-se a ordenação dos

termos de cada query pelo seu grau de generalidade (ver figura 4.3 ponto 2 para 3). Depois, todas as queries são ordenadas por ordem alfabética numa lista (ver figura 4.3 ponto 3 para 4), procedendo-se a uma execução iterativa dos seguintes passos: retirar o conjunto de queries que estejam no topo da lista e cujo primeiro termo de todas as queries seja o mesmo (D) (ver figura 4.4 passo 1). Se o tamanho desse conjunto for maior do que um valor κ imposto, então é criado um primeiro nível na árvore com esse termo. Do conjunto inicial D (ver figura 4.4 passo 2), obter o sub-conjunto sD que contenha termos de segunda ordem iguais. A esse conjunto reaplica-se novamente o conjunto de passos descrito até agora. E assim sucessivamente. O valor de κ depende do nível de profundidade na estrutura de tal modo que $\kappa_0 = 10$ e $\kappa_i = (\kappa_{i-1})/2$.

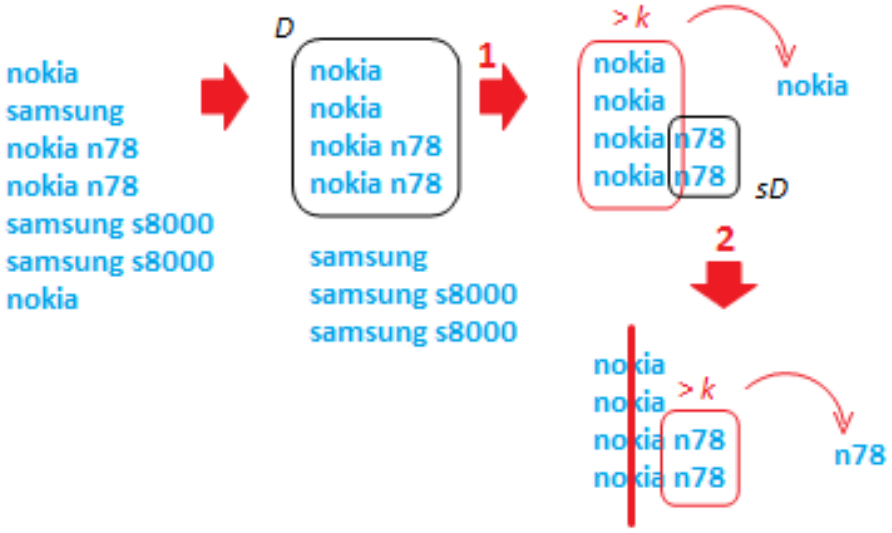

Figura 4.4: Geração da hierarquia de termos.

O algoritmo termina quando a lista de queries ordenadas estiver vazia, obtendo-se como resultado uma árvore da estrutura relacional dos termos das queries (ver figura 4.3 ponto 4).

4.2 Refinamento da árvore representativa dos interesses do utilizador com o uso da categorização

A árvore de interesses do utilizador obtida a partir da ramificação dos termos das queries já permite obter algum conhecimento sobre este. É contudo incompleta visto que somente a partir das queries não é possível extrair informação sobre todas as categorias de interesse para o utilizador. Um utilizador menos habituado ao uso de motores de pesquisa,

se pretender encontrar informação sobre bicicletas provavelmente introduzirá uma query com este termo. Ao ver o conjunto de resultados e as inúmeras categorias apresentadas irá com certeza numa próxima pesquisa efectuar uma query mais forte como "bicicletas montanha". Neste caso o sistema será capaz de pela ramificação das queries, obter as categorias de interesse do utilizador. Contudo utilizadores mais avançados, cujas queries utilizadas são mais específicas, não permitem ao sistema saber que pesquisas por "ktm", "trek", "shimano", "berg", pretendem obter informações sobre o mesmo tipo de interesse do utilizador menos experiente (bicicletas).

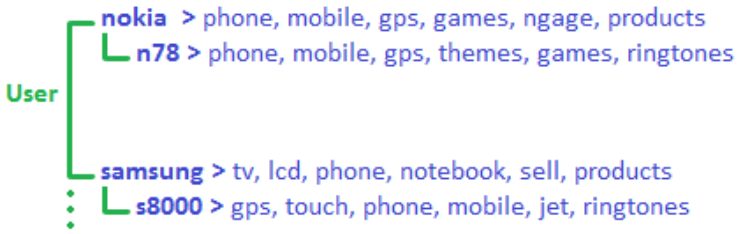

Figura 4.5: Associação de clusters a termos.

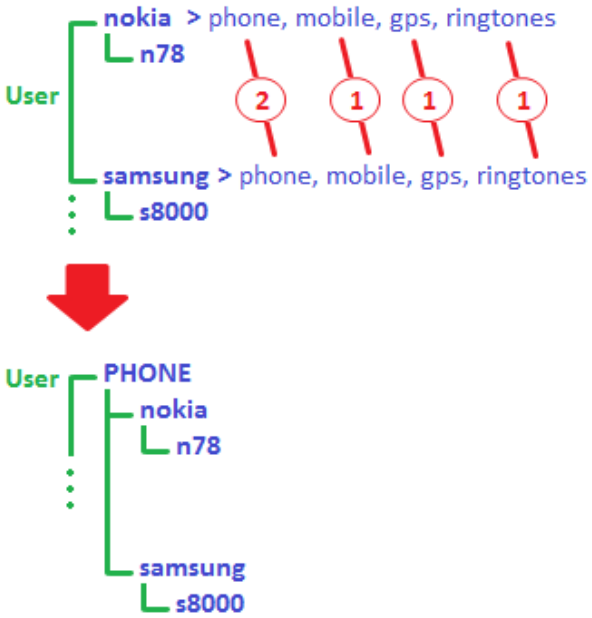

Figura 4.6: Relação de termos através dos clusters associados.

O uso das categorias associadas a cada query permite descobrir esta informação (ver figura 4.5). Se um utilizador efectua muitas pesquisas sobre um determinado tema, é provável que esse tema seja uma das categorias encontradas pelo algoritmo de categorização

dos resultados. Ou seja, um utilizador vai introduzindo as suas queries. A estas queries estão associados um conjunto de tópicos. Os tópicos que muito provavelmente são relativos ao utilizador, são aqueles que são mais frequentes de aparecer no conjunto das queries do utilizador.

De forma a seleccionar as categorias que melhor representarão os interesses, o algoritmo para cada sub-árvore no nível 1 da árvore de interesses de utilizador, extrai as categorias que são mais frequentes por entre o conjunto de queries pertencente a essa sub-árvore 4.6. O sistema dispõe de uma estrutura que permite guardar informação sobre cada categoria seleccionada. São guardados os identificadores das sub-árvores de nível 1 em que a categoria ocorre e a frequência com que essa categoria ocorre dentro de cada sub-árvore. Com estes dados, é possível saber que sub-árvores se relacionam (pelas categorias) e o objectivo passa por aglomerar as sub-árvores que se interligam por um maior número de categorias.

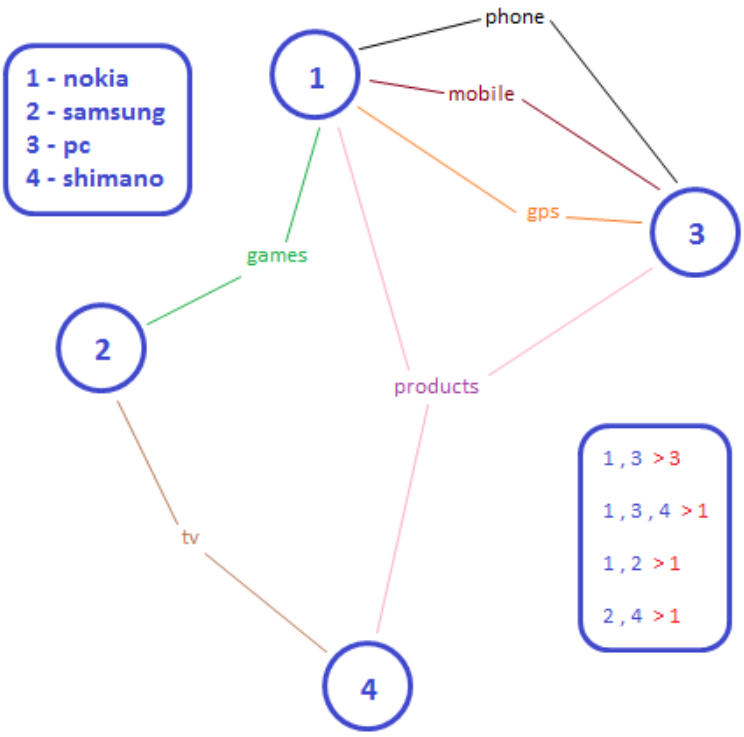

Figura 4.7: Grafo relacional de termos de nível 1.

Para alcançar o objectivo, é criado virtualmente um grafo em que as diversas sub-árvores são os vértices e os clusters as arestas (ver figura 4.7). São computados todos os grupos possíveis de interligação entre vértices com base na junção de vértices que são unidos

exactamente pelas mesmas categorias (se tal não fosse efectuado o número de grupos seria elevadíssimo). A cada grupo associa-se numa estrutura as categorias que interligam os seus elementos e para cada uma delas é guardado o valor de ligação mais fraca com os vértices do grupo. Este valor serve para numa fase final decidir qual o nome de categoria que irá descrever o grupo, sendo que o escolhido é aquele cujo valor de ligação mais fraca é maior.

De todos os grupos encontrados, é feita uma filtragem com a imposição de um valor mínimo de interligações entre os elementos constituintes. Os grupos que passam esta imposição entram para o cálculo da média do número de interligações entre os diversos grupos. No final só são aceites os grupos cujo número de interligações entre os seus elementos é maior que a média.

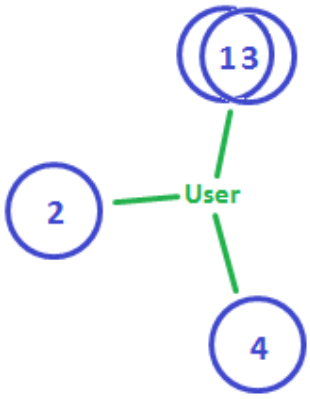

Figura 4.8: Interesses do utilizador.

Por fim, os descritores de cada grupo são analisados e se existirem os mesmos descritores para grupos diferentes, esses grupos são aglomerados. A árvore é actualizada, passando certos elementos do nível 1 da árvore criada na primeira fase (somente com as queries), a serem descendentes das categorias seleccionadas como interesses de utilizador e que aglomeraram estes elementos (ver figura 4.8).

4.3 Comparação com o estado da arte

Como referido na proposta de trabalho, o objectivo da metodologia apresentada nesta tese para a criação de modelos de utilizador foi de afastar a necessidade do uso do conteúdo dos documentos relevantes para efectuar a construção destes mesmos modelos. Uma imposição

sempre presente para a criação de modelos de utilizador é que estes devem ser gerados a partir de conteúdo que seja realmente interessante ou apelativo ao utilizador. Os estudos feitos por Morita e Shinoda [41], Kim, Oard e Romanik [33] entre outros, foram importantes no sentido de mostrarem a melhor forma de reconhecer o interesse do utilizador em certos documentos. O tempo foi o factor mais discriminatório referido por ambos e foi o que usamos para reconhecer o interesse num determinado documento. Poderia-mos ter usado mais factores mas este aspecto não era um dos objectivos da tese.

De todo o trabalho relacionado mencionado, o único que tem certos aspectos em comum com o nosso é o de Tamine, Boughanem e Zemirli [37] pois a construção dos modelos baseia-se no histórico das pesquisas e nos documentos consultados a partir daí. Contudo ao invés de fazer uso da informação contida nos documentos para extrair conhecimento sobre o utilizador, o nosso processo faz uso da categorização dos documentos efectuada aquando da devolução dos resultados da pesquisa. Visto que a categorização é um processo automático e completamente dependente do conteúdo dos resultados, as categorias utilizadas não são restringidas a tópicos pré-definidos como Sieg, Mobasher e Burke [23] fazem para a criação das ontologias propostas. Desta forma é possível obter um conjunto de interesses mais direccionado e específico a cada utilizador e sem a necessidade de consultar o conteúdo dos documentos.

Capítulo 5

Discussão dos resultados

Os dois temas abordados na tese apresentam um grande problema no que diz respeito à análise dos resultados. É muito difícil arranjar uma medida que possa quantificar o nível da qualidade dos resultados. Tal deve-se ao facto da categorização de Web snippets e a construção de perfis de utilizadores serem processos automáticos e não supervisionados. Não existem resultados pré-definidos como sendo bons resultados para que se possa fazer uma comparação. Cabe assim a avaliação dos resultados ser fruto de uma observação subjectiva entre o conteúdo dos resultados e a informação que um utilizador esperaria obter. A comparação entre o sistema desenvolvido e os sistemas já existentes é uma outra forma de avaliar os resultados. No nosso caso concreto, vamos comparar os resultados do processo de categorização implementado, aos resultados do motor de categorização referência no sector: o Vivíssimo. No que toca aos modelos de utilizador, serão apenas mostrados os resultados obtidos.

5.1 Resultados obtidos pelo CBL

A figura 5.1 exemplifica a forma de mostrar a informação na aplicação VIPWeb criada de propósito para o trabalho desta tese. As categorias são listadas no lado esquerdo e na área do lado direito, são apresentados os urls e a sua respectiva informação. A aplicação conforme o seu redimensionamento mostra mais ou menos resultados sem que seja necessário efectuar scroll. Existem três modos de apresentação dos resultados: 1 - ver o primeiro url de cada cluster; 2 - ver todos os resultados ordenados por ranking; 3 - ver resultados de um cluster. Os resultados mostrados na figura correspondem à pesquisa por “programming”.

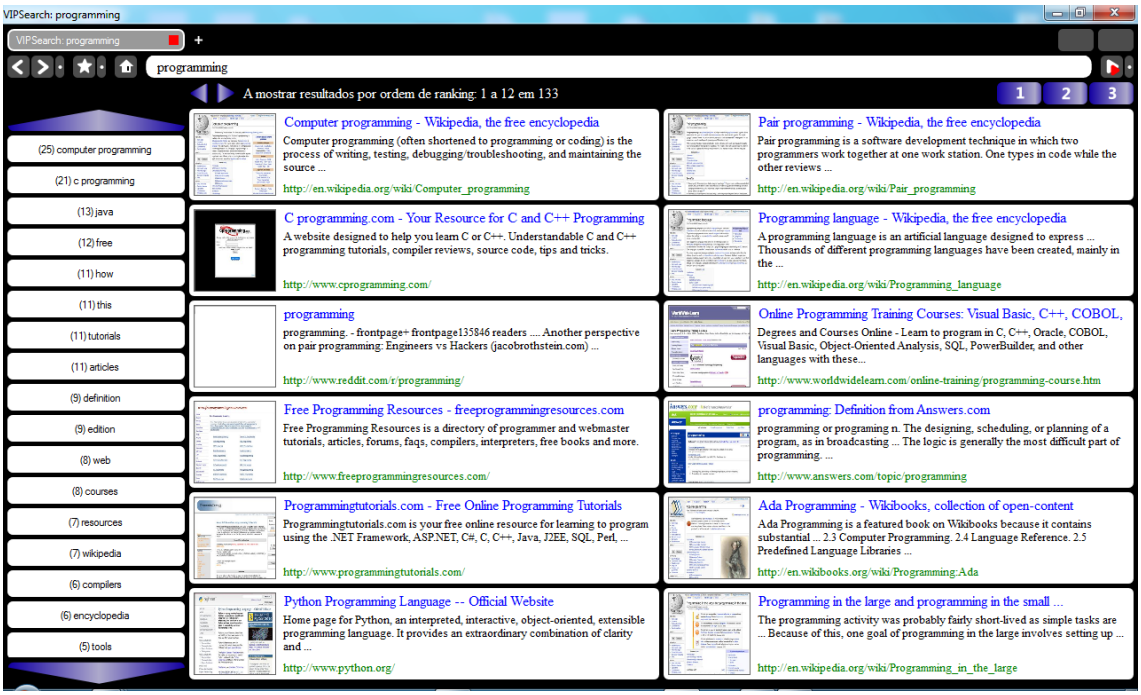

Figura 5.1: query: “programming”.

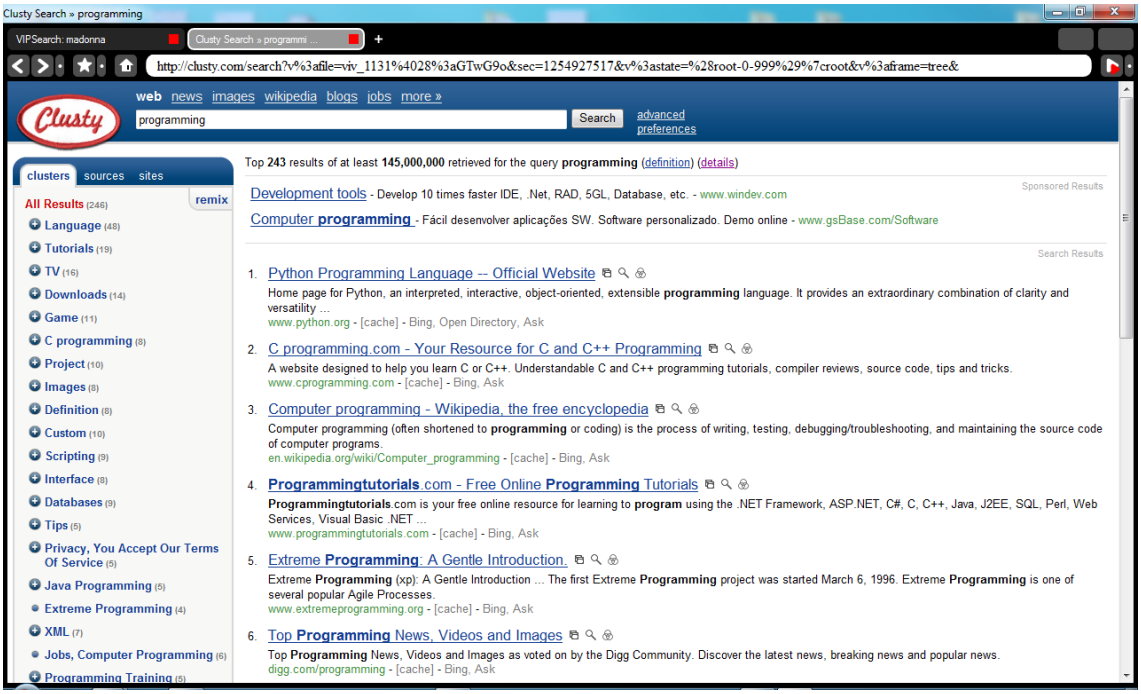

Figura 5.2: Clusty query: “programming”.

Na figura 5.2 são mostrados os resultados do Vivíssimo para a mesma query. O Vivíssimo neste caso desambigua melhor a palavra ao encontrar termos como “tv” ou “games”. Tal

facto deve-se ao uso de uma maior diversidade de motores de pesquisa. No nosso caso, ao longo da lista de termos extraídos dos snippets dos urls, a palavra “game” só ocorreu com uma frequência de 2.

Segue-se na figura 5.3 uma listagem dos termos seleccionados após análise das várias propriedades w descritas no capítulo 3. É a partir de alguns destes termos que cada grupo é construído sendo que o seu label ou descrição pode ser apenas um dos termos ou então uma sequência de termos (expressões compostas).

```
sql - W: 0,00455154919837343
wikibook - W: 0,0121549468211516
perl - W: 0,0126544879440278
programming - W: 0,0151090276558331
cgi - W: 0,0306623495222967
video - W: 0,0391178116858215
tools - W: 0,0398862912336226
book - W: 0,0490771624479124
faq - W: 0,0638562530166495
articles - W: 0,0661191633511354
computer - W: 0,068880239862401
c - W: 0,0773402209091494
interactive - W: 0,0795464237991807
media - W: 0,0795464237991807
subject - W: 0,0798985403796926
practice - W: 0,098528162720882
content - W: 0,116984399098164
posix - W: 0,133580930636747
compiler - W: 0,134416535032952
object-oriented - W: 0,135363130350085
introduction - W: 0,14278319278765
java - W: 0,143801157805175
window - W: 0,206100605896839
such - W: 0,228632478632479
network - W: 0,270197361351629
college - W: 0,275255712313168
forum - W: 0,286583533653846
tip - W: 0,307692307692308
thread - W: 0,322780585105484
linux - W: 0,323277170621031
object - W: 0,338141025641026
procedure - W: 0,338141025641026
design - W: 0,346153846153846
selection - W: 0,350210336538462
how - W: 0,363986459968603
example - W: 0,383413461538462
proces - W: 0,385817307692308
visual - W: 0,393874643874644
science - W: 0,402911324786325
rule - W: 0,452991452991453
mathematical - W: 0,452991452991453
dish - W: 0,457264957264957
custom - W: 0,457264957264957
please - W: 0,457264957264957
browse - W: 0,457264957264957
wide - W: 0,457417582417582
find - W: 0,478632478632479
collection - W: 0,479266826923077
ruby - W: 0,482271634615385
top - W: 0,482271634615385
more - W: 0,491286057692308
intelligence - W: 0,498461538461538
encyclopedia - W: 0,5
debugging - W: 0,50801282051282
distributed - W: 0,50801282051282
time - W: 0,512019230769231
open - W: 0,5125
directory - W: 0,516025641025641
community - W: 0,525412087912088
tutorials - W: 0,550289459675273
system - W: 0,572884615384615
code - W: 0,577991452991453
course - W: 0,618461538461538
topic - W: 0,64453125
today - W: 0,64453125
freelance - W: 0,64453125
web - W: 0,649188182060154
extreme - W: 0,650240384615385
microsoft - W: 0,650390625
language - W: 0,657448611994066
reference - W: 0,664615384615385
home - W: 0,679487179487179
2008 - W: 0,679487179487179
then - W: 0,679487179487179
task - W: 0,681623931623932
full - W: 0,685897435897436
downloads - W: 0,714921652421653
site - W: 0,740769230769231
definition - W: 0,749607535321821
basic - W: 0,751536185707783
pragmatic - W: 0,752884615384615
graphic - W: 0,771634615384615
just - W: 0,775240384615385
resource - W: 0,792330450614474
using - W: 0,796875
link - W: 0,803365384615385
guide - W: 0,806790865384615
machine - W: 0,806790865384615
service - W: 0,814102564102564
programmer - W: 0,830959164292498
provide - W: 0,852029914529915
writing - W: 0,870726495726496
degree - W: 0,880096153846154
artificial - W: 0,904447115384615
unix - W: 0,905982905982906
before - W: 0,905982905982906
live - W: 0,905982905982906
learning - W: 0,906593406593407
many - W: 0,91025641025641
data - W: 0,91025641025641
music - W: 0,91025641025641
including - W: 0,91025641025641
learn - W: 0,918626827717737
software - W: 0,929142878605769
art - W: 0,938076923076923
this - W: 0,951388888888889
new - W: 0,951923076923077
text - W: 0,966346153846154
work - W: 0,966346153846154
beginner - W: 0,996923076923077
```

Figura 5.3: Lista de termos seleccionados para a query “programming”.

O algoritmo neste momento trabalha com palavras cujo nível de importância esteja abaixo de 1, um número talvez algo elevado e que muitas vezes possibilita que exista a passagem de termos não desejados. A escolha deste valor é mais por uma questão de equilíbrio visto

que, com um número próximo de 1, podem existir termos mais fracos, mas também são mantidos praticamente todos os bons termos. Se a constante for definida com um valor mais aproximo de 0, o conjunto de termos vai sendo reduzido, eliminando todas as palavras não desejadas, mas também removendo muitas que poderiam ser interessantes.

A lista de palavras compostas (ver figura 5.4) é completamente dependente da qualidade da lista de termos que são aceites. Isto porque se a lista de termos mais importantes contiver palavras sem interesse e muito comuns, é provável que estas venham a fazer parte das expressões compostas, reduzindo a sua qualidade. Como se pode verificar na figura, as palavras compostas seleccionadas para as queries "programming" (dir.) e "batata" (esq.) são quase todas boas expressões. Tal facto deve-se á maioria dos termos seleccionados introduzirem conhecimento e aqueles que não o fazem, geralmente ocorrem com uma frequência muito baixa para serem considerados nas expressões compostas. No caso da query "heart" (centro) ao serem aceites os termos "your" e "local", fazem com que se encontrem algumas expressões fracas.

```
batata inglesa - 4
batata doce - 19
batata recheada - 3
batata frita - 14
batata vada - 11
batata vada recipe - 3
batata palha - 4
batata wada - 4
sweet potato - 10
la batata - 6
salgado valdez - 3
information about batata - 3
```

```
heart disease - 46
heart disease prevention - 4
about heart disease - 6
information about - 3
heart information - 3
more about - 4
learn about heart - 5
about heart - 14
congenital heart disease - 8
congenital heart - 13
heart failure - 17
heart health - 6
american heart association - 9
heart attack - 19
heart music - 5
heart rhythm - 3
human heart - 8
muscular organ - 4
your heart - 13
your local - 3
circulatory system - 8
```

```
computer programming - 21
programming language - 21
free programming - 5
encyclopedia article - 4
pair programming - 3
c programming - 12
c sharp - 3
object-oriented programming - 4
web programming - 5
top programming - 3
extreme programming - 6
php programming - 3
java programming - 7
erlang programming - 3
mathematical programming - 5
```

Figura 5.4: Lista de palavras compostas para as queries "batata", "heart" e "programming".

Para uma melhor ideia dos resultados obtidos pelo processo de categorização seguem-se figuras com uma comparação ao Vivíssimo de diversas queries, das quais algumas pertencem à lista das top 500 mais procuradas pelo mês de Junho de 2009 [11].

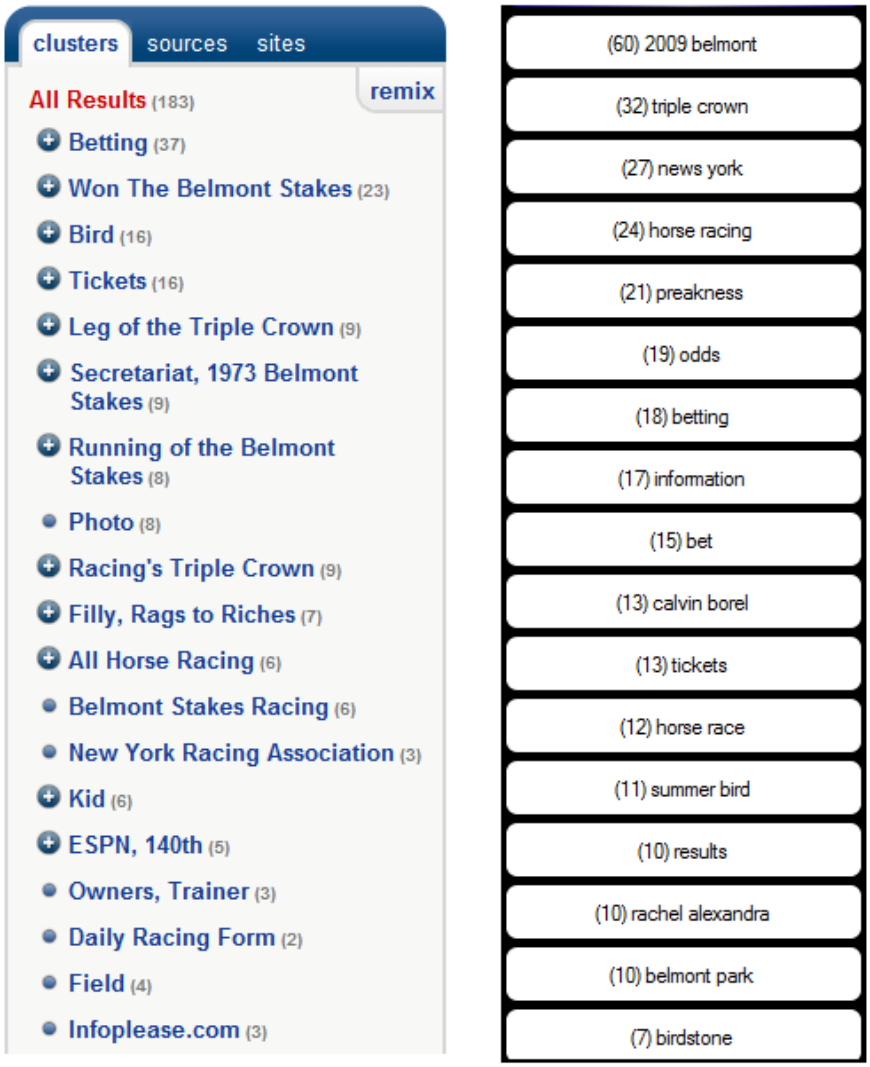

Figura 5.5: CBL vs Vivíssimo: “belmont stakes”.

Figura 5.6: CBL vs Vivíssimo: “culinária”.

Figura 5.7: CBL vs Vivíssimo: “land of the lost”.

Figura 5.8: CBL vs Vivíssimo: “porto”.

Figura 5.9: CBL vs Vivíssimo: “batata”.

Como se pode observar pelos resultados, o algoritmo CBL é capaz de encontrar muitas das categorias que o Vivíssimo encontra, sendo isto um bom indicador de qualidade. Perde na não remoção de palavras que não introduzem conhecimento, ao propor por exemplo a categoria “this‘”. Contudo, ao não remover este tipo de palavras, podem-se manter termos que por vezes são informativos. Tomemos o caso do “it”. Normalmente é considerada uma stop-word mas no caso da pesquisa por “hp” pode-se observar na figura 5.10 (e 5.11 no ponto 3 para a query “network”) que a categoria “it” apresentada, refere-se na sua globalidade a “information technology”. O termo “how” também pode ser interessante no caso de alguém que pretenda saber como fazer algo (ver figura 5.11 no ponto 2). Os

sistemas que fazem uso de stop-words removem sempre este tipo de palavras, mesmo nos casos em que elas apresentam significado devido ao contexto onde estão inseridas. Para além disso o uso de stop-words implica a existência de um conjunto de stop-words para cada linguagem. O nosso algoritmo não usa stop-words. Procura desvalorizar as palavras que normalmente são consideradas stop-words mas somente quando realmente estas não produzem conhecimento relativamente ao contexto. Ao realizar esta tarefa de forma autónoma, a generalidade dos resultados é satisfatória e prova que é possível criar um sistema de categorização sem o uso de stop-words. Infelizmente em certos casos ainda existe algum ruído.

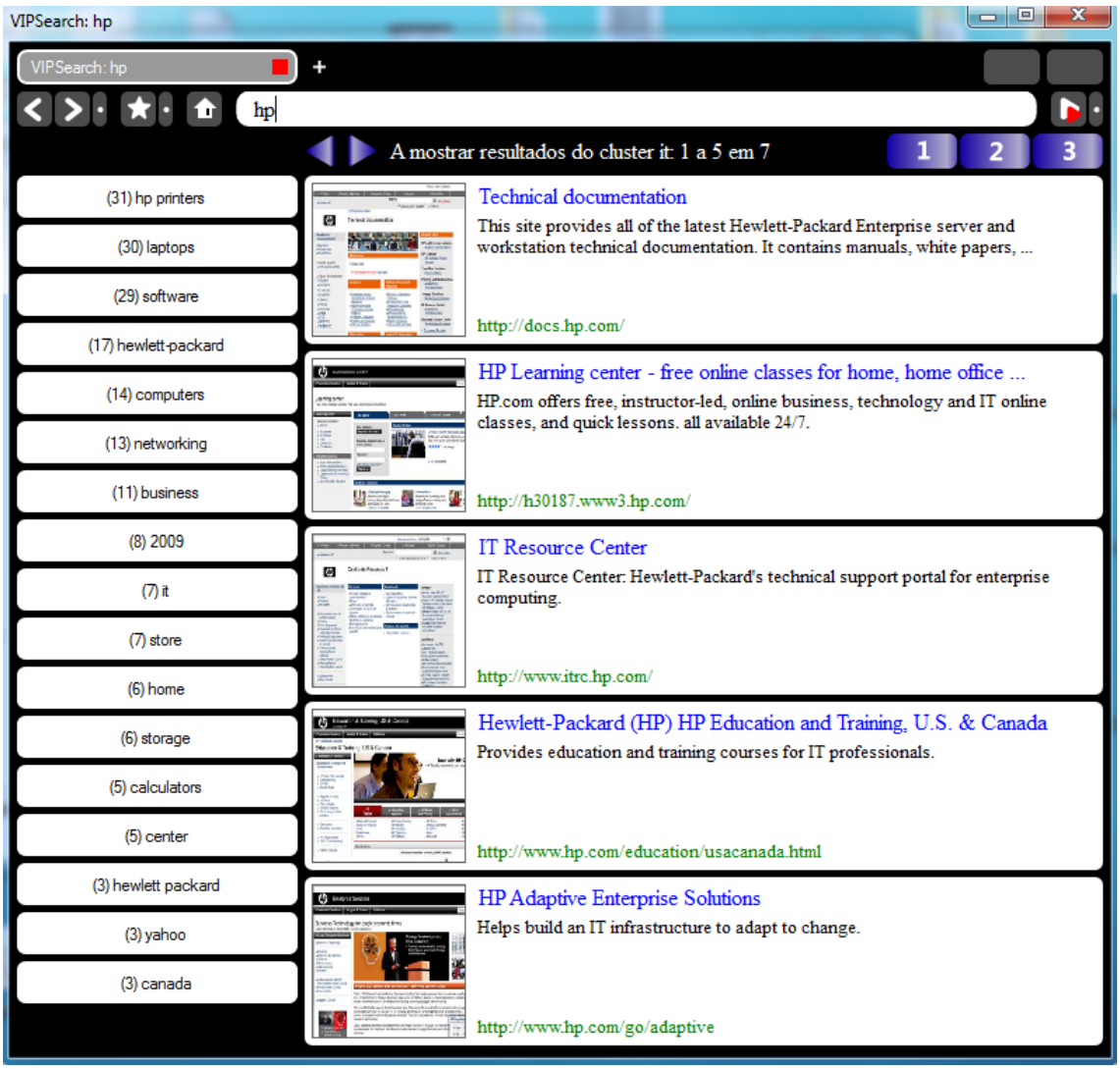

Figura 5.10: VIPWeb: “hp”.

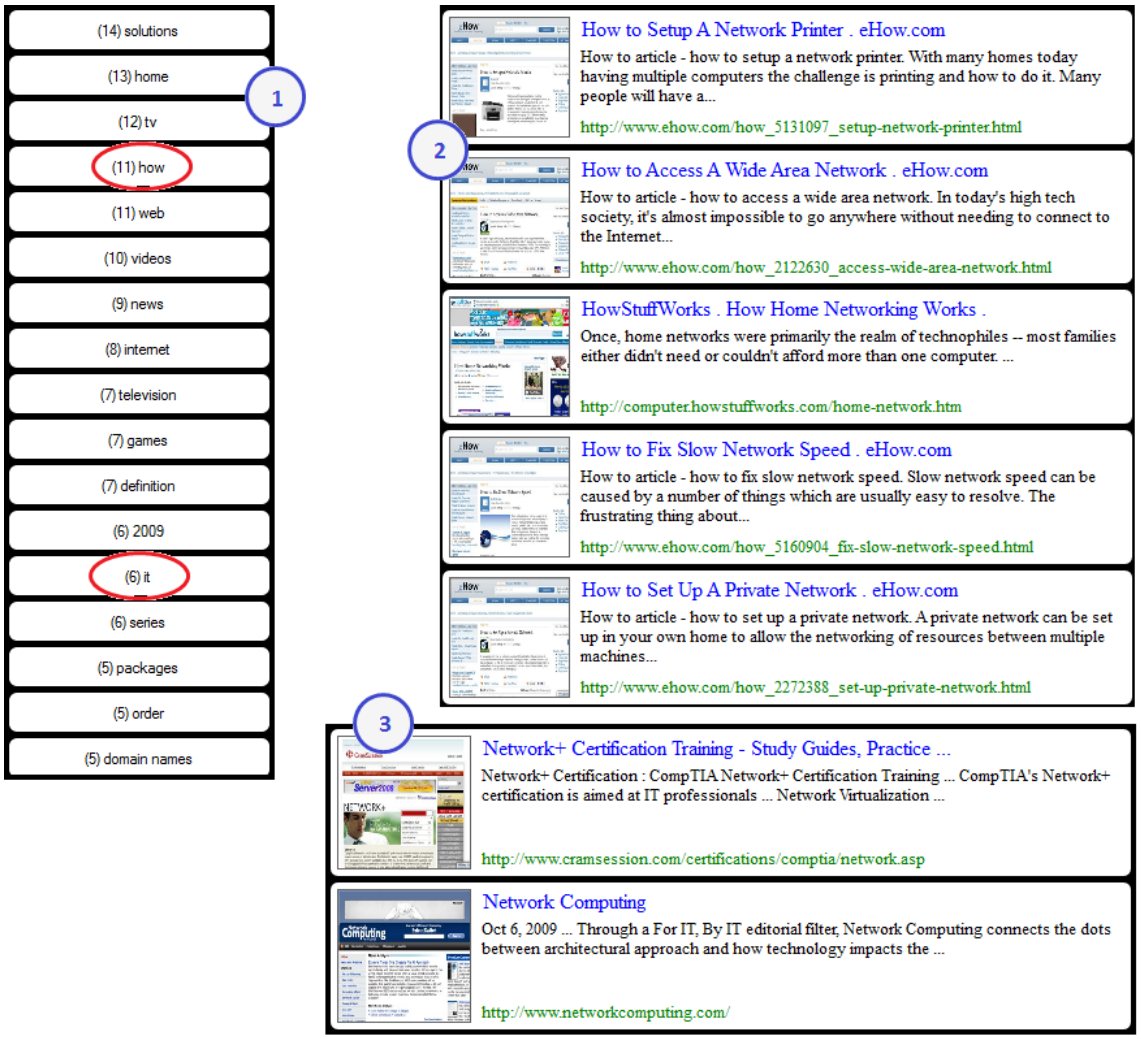

Figura 5.11: VIPWeb: “network”.

Salienta-se ainda que a qualidade do processo de categorização vai-se deteriorando com a especificidade da query. Isto vai de encontro à utilidade do processo de categorização que é realmente importante no caso das queries serem pouco específicas e serem ambíguas. Como se pode observar na figura 5.12, um utilizador que introduza a query ”air”irá certamente beneficiar do processo de categorização já que o termo é bastante ambíguo. A lista de categorias apresentadas no ponto 1 é representante de diversos temas. O utilizador pode estar a procura de tópicos relativos à qualidade do ar, a procurar por sapatilhas específicas da marca Nike (ponto 3) ou por poluentes do ar (ponto 2) entre outros assuntos.

No caso das queries serem muito específicas, significa que o utilizador sabe muito bem aquilo que procura, reduzindo o número de categorias passíveis de serem encontradas. Neste caso, podem surgir muitas palavras que não representam conhecimento. Tal facto deve-se

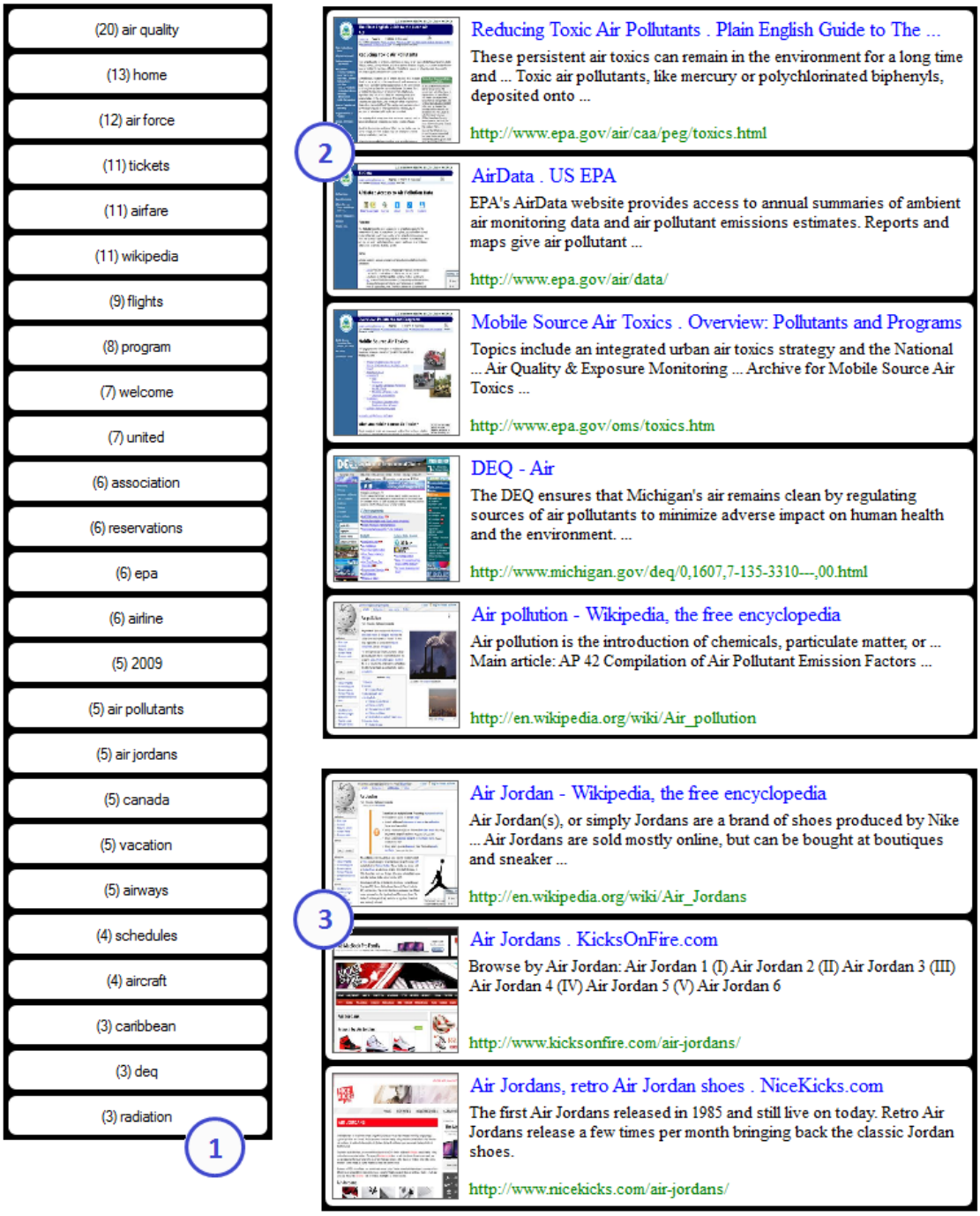

Figura 5.12: VIPWeb: “air”.

à existência de muitas frases repetidas por entre os diversos resultados, fazendo com que as palavras co-ocorram com poucas palavras diferentes mas com uma frequência elevada, sendo classificadas pelo algoritmo como sendo relevantes.

5.2 Variação dos parâmetros

Ao descrever o algoritmo de clustering foi referido o uso de vários parâmetros que influenciam os resultados finais. Nas próximas figuras, são variados esses parâmetros e podem-se observar as diferenças nos resultados.

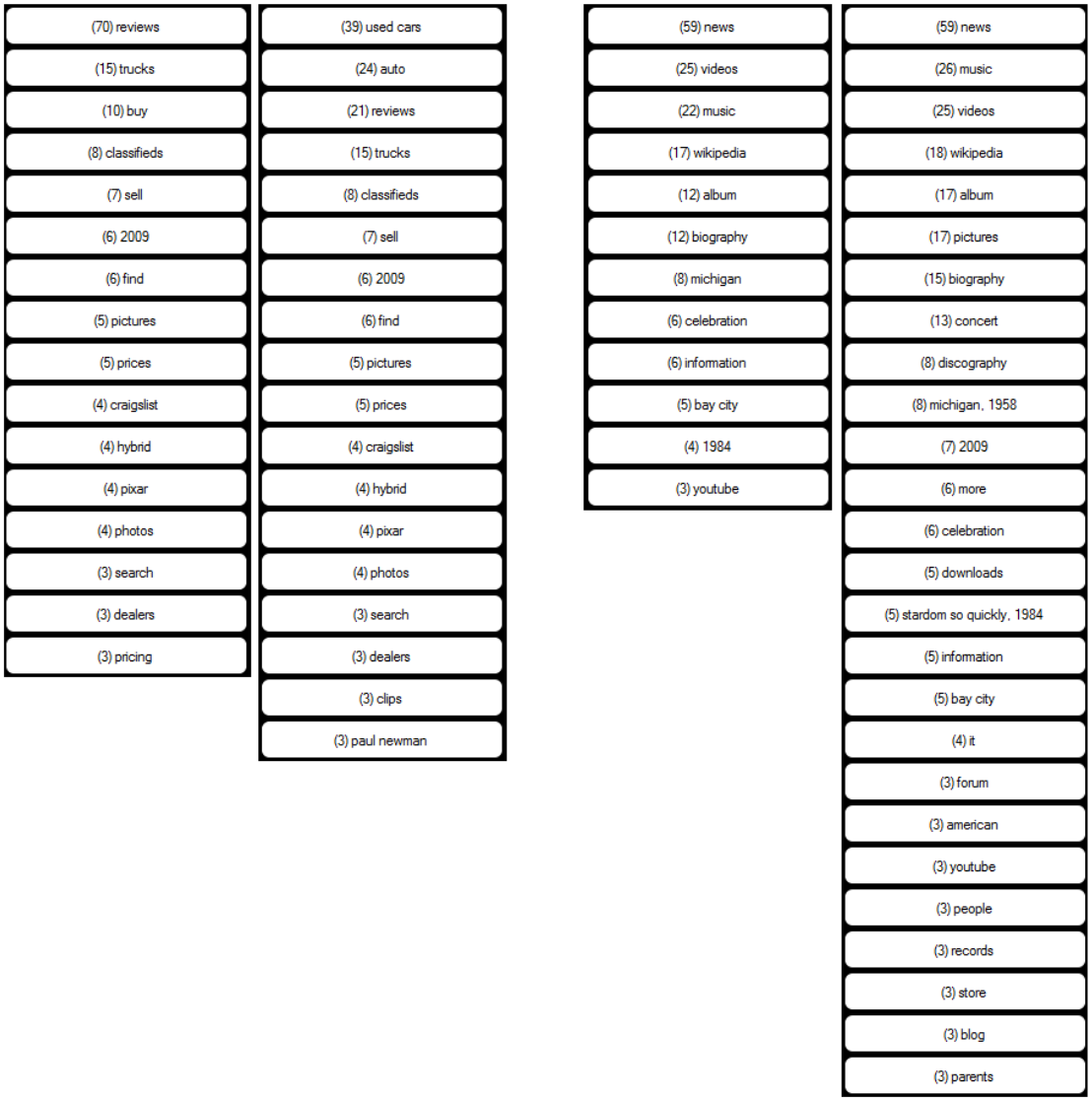

Figura 5.13: VIPWeb, variação de parametros para as queries “cars” e “madonna”.

A figura 5.13 mostra a variação do valor α (número de primeiros termos seleccionados de cada url para criação de pólos de clusters). Foram utilizadas duas queries ”cars”(duas listas do lado esq.) e ”madona”(duas listas do lado dir.) em que para cada query no lado esquerdo o valor de α é igual a 1. No lado direito o valor de α é igual a 2. Como se pode

observar ao aumentar α o número de categorias aumenta devido a geralmente o número de termos seleccionados para pólos também aumentar. No caso da query ”madonna”o número de clusters é bastante maior possivelmente porque os termos existentes na segunda posição da lista ordenada para cada url são bastante diferentes dos que estão na primeira posição.

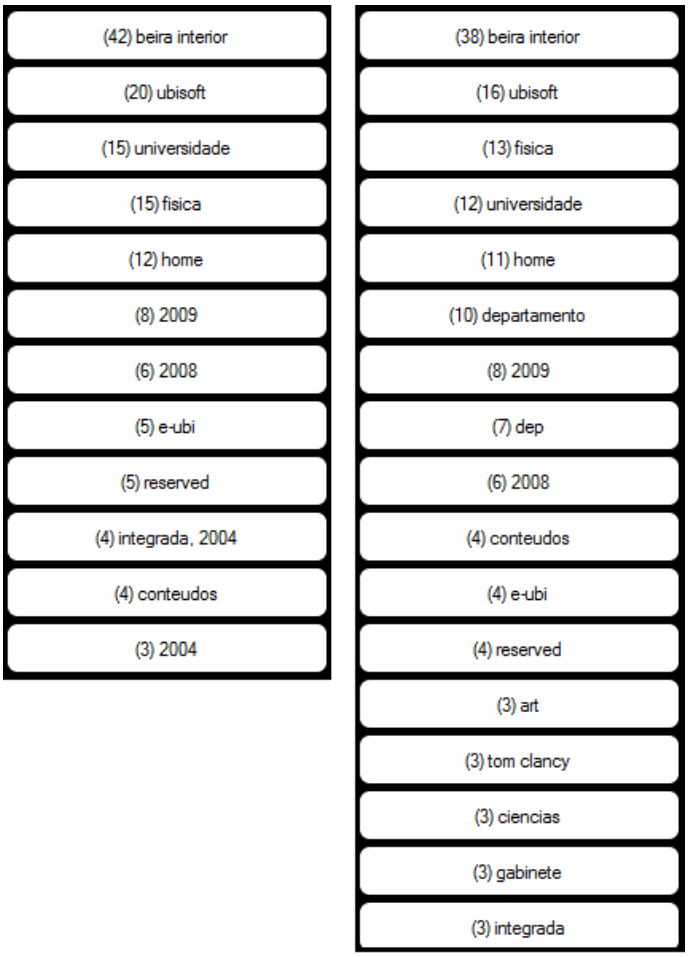

Figura 5.14: VIPWeb, variação de parametros para a query “ubi”.

Outros parâmetros passíveis de variação são os valores μ e θ que controlam a absorção e a união entre clusters respectivamente. Na figura 5.14 as categorias obtidas para a query ”ubi”em português no lado esquerdo foram obtidas com valores menos restritivos no que toca à absorção ($\mu = 0.6$) mas com um valor exigente para a união ($\theta = 0.9$). Deste modo obtém-se um número menor de categorias visto que cada uma pode conter muitas subcategorias. No lado direito as categorias foram obtidas com valores de $\mu = 0.75$ e $\theta = 0.8$. Estes valores permitem um maior número de categorias visto ser menos fácil uma categoria ser incorporada dentro de outra. A união continua a ser algo restritiva. De salientar

que os valores em uso normal do algoritmo são $\alpha = 1$, $\mu = 0.7$ e $\theta = 0.8$.

5.3 Estrutura hierárquica e tempos de execução

No capítulo 3, ao ser descrito o algoritmo implementado CBL, foi referido que este é capaz de gerar uma categorização hierárquica. Os resultados apresentados não vão de encontro a esta definição. Tal facto deve-se a termos seleccionado apenas os grupos do primeiro nível, ocultando os sub-níveis e por isso apresentando uma lista de categorias plana. O motivo para a escolha tomada deve-se ao uso deste algoritmo para auxiliar à construção do modelo de utilizador onde as subcategorias já seriam demasiado específicas para o tipo de perfil de utilizador que ambicionamos construir. Na figura 5.15 é exposta a estrutura hierárquica gerada pelo algoritmo para a query "madonna".

```
* news, photos
    - merchandise
    - gossip
* music
* videos
* wikipedia
* pictures
    - wallpapers
* biography
* albums
* concert
* songs
* michigan
* celebration
* information
* bay city
* 1984
* youtube
* people
```

Figura 5.15: Estrutura hiérarquica.

O algoritmo proposto ao funcionar como meta-motor de pesquisa impõe que os seus resultados tenham de ser obtidos em tempo real. Na tabela 5.1 são mostrados os tempos totais da pesquisa bem como o tempo de recepção dos resultados dos motores de pesquisa. É fácil verificar que grande parte do tempo dispendido em todo o processo vai para o tempo de recepção dos resultados. A média de execução do processo de categorização anda em volta de 1 segundo num processador Core 2 Duo a 2Ghz.

query	Tempo total	Tempo de recepção motores busca
“cars”	2331	1905
“madonna”	3614	2646
“apple”	3585	2562
“windows”	4077	3084
“lorus”	5027	3837
“heart”	4469	3344
“programming”	3936	2961
“shimano”	5524	4572
“culinária”	5470	4845
“jogos”	5782	4934
“belmont stakes”	3361	2403
“land of the lost”	3780	2764
Média	4246	3253

Tabela 5.1: Tempos de processamento.

5.4 Geração de modelos de utilizador

Relativamente aos modelos de utilizador, optamos por não efectuar comparações a outros sistemas por diversos motivos. Primeiramente porque normalmente os modelos de utilizador são elementos de fases pré-finais, ou seja, servem para ajudar a chegar a certos resultados e por isso, os sistemas não mostram a sua informação. Outro aspecto a ter em conta é o da maior parte dos estudos na área se centrar num conjunto fechado de documentos, ao contrário do nosso caso que pode ser toda a Web. Aqueles que também o fazem usam aplicações específicas e para poder existir uma comparação seria necessário uma mesma pessoa utilizar ambos os sistemas e efectuar as mesmas acções, trabalho algo moroso de ser efectuado e por isso posto de parte nesta tese. Dado a implementação dos modelos de utilizador estar ainda numa fase experimental no decorrer do trabalho, não foi possível também reunir um número significativo de testes que pudessem comprovar a eficiência da metodologia proposta.

Deste modo para a avaliar o sistema de criação de modelos de utilizador e saber se os resultados são representativos ou não dos seus interesses, recorremos à avaliação subjectiva

por parte de um utilizador. Este utilizou o sistema por um período de tempo ao fim do qual o modelo gerado na primeira fase do processo está apresentado na figura 5.16 na árvore apresentada à esquerda. Nesta fase, a árvore gerada apenas é constituída por termos da query.

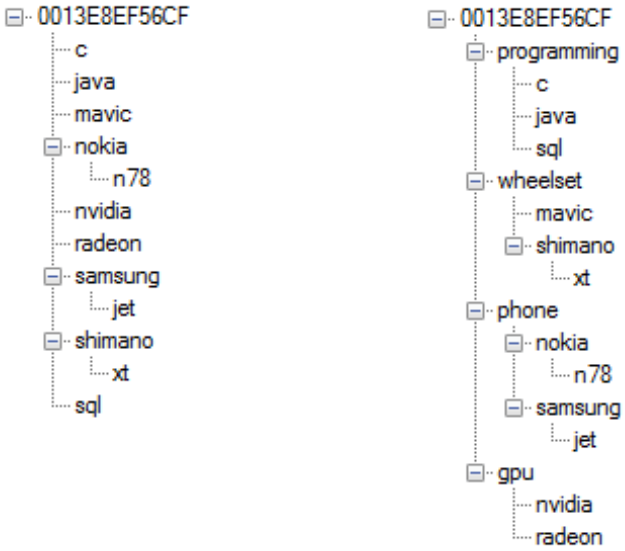

Figura 5.16: Árvores representativas de um modelo de utilizador.

Após o uso da categorização associada às queries, foi possível obter o modelo mostrado na figura 5.16 na árvore apresentada á direita. Como se pode observar, ao serem utilizadas as categorias associadas às queries, foi possível extrair conhecimento representativo dos seus interesses e que permitiu aglomerar diversos tópicos propostos na árvore da fase inicial.

Os resultados apresentados neste perfil de interesses, vão de encontro à auto-avaliação feita pelo utilizador alvo da experiência onde indicou que os seus interesses recaíam sobre as categorias: ciclismo, programação, tecnologia. Num futuro este perfil poderá ser utilizado para adaptar os resultados a uma pesquisa efectuada pelo utilizador, sendo escolhidos como resultados mais relevantes, aqueles cujos termos se encontrarem mais nas categorias que definem o utilizador.

Capítulo 6

Conclusão e trabalho futuro

6.1 Conclusão

Quando iniciámos o trabalho tínhamos como objectivo melhorar a experiência de pesquisa para o utilizador. Para tal foi aprofundado o conhecimento relativo ao modo de operar dos motores de pesquisa bem como sobre todo o conjunto de metodologias existentes ou em fase de implementação, que visam melhorar os resultados destes sistemas. Deparamo-nos com dois caminhos algo distintos: a categorização da informação e a personalização. Se de algum modo a categorização e a personalização estão associados, isto é porque simplesmente na personalização a categorização é um dos meios utilizados para construir os modelos de utilizador como foi referido no capítulo 2. Se não for por este motivo, os sistemas ou fazem a categorização dos resultados e adicionam esta informação aos resultados ou então devolvem uma lista de resultados personalizada mas sem mostrarem nenhum tipo de categorias associadas a essa informação. Ao descobrirmos isto, iniciámos uma investigação com o intuito de juntar o que há de melhor nos dois mundos. Efectuar uma categorização dos resultados e construir modelos de utilizador auxiliados da categorização que depois pudessem ser utilizados para a personalização dos resultados da pesquisa e da estrutura resultante da categorização.

Dada a extensão da ideia, optámos por impor como objectivo a categorização dos resultados e a criação dos modelos de utilizador que, como referido na introdução e na proposta de trabalho, fazem uso da categorização mas de uma maneira diferente de todos os artigos estudados relativos ao assunto. No nosso caso particular a categorização dos resultados

serve para adicionar ao histórico das queries do utilizador a informação relativa aos assuntos que essas mesmas queries abordam. Esta foi uma ideia inovadora que permite a construção de um modelo de utilizador representado sobre a forma de uma árvore, simplesmente a partir de uma estruturação das palavras da query e da interligação das categorias que são comuns a certos ramos da árvore. Sendo assim o trabalho de investigação dividiu-se em duas fases que no final se interligam. A primeira, implementar um algoritmo que permitisse uma categorização dos resultados. A segunda, fazer o registo das queries introduzidas e relacioná-las com as categorias geradas na primeira fase de modo a poder construir a partir daí um modelo de utilizador. Podemos deste modo retirar conclusões para as duas etapas independentemente.

Relativamente à fase de categorização, ao optarmos por também aí inovar no sentido de que pretendíamos estudar até que ponto era possível afastarmo-nos da introdução de conhecimento nos resultados provenientes dos vários motores de pesquisa utilizados, podemos concluir que realmente é possível encontrar grupos cujo conteúdo tem por base documentos que estão associados a palavras fortes encontradas através da nossa metodologia. Ao compararmos os resultados com a referência incontestável do sector, o Vivíssimo, é possível ver que para a mesma query, muitas das categorias deste poderoso motor se encontram entre as categorias devolvidas pelo nosso sistema. A grande vantagem do nosso algoritmo ao não introduzir conhecimento como o enriquecer dos snippets com recurso a ferramentas externas (caso do Snaket), a não remoção de palavras vazias e a não utilização de algoritmos de análise morfológica é que faz com que seja possível obter resultados para qualquer tipo de linguagem dos continentes europeu e americano (certas particularidades do nosso algoritmo não se adaptam à língua chinesa por exemplo). Ao ser independente de informação externa, o desempenho do algoritmo também é bastante bom, gerando categorias em menos de um segundo após obter os resultados de todos os motores de pesquisa utilizados. Grande parte dos sistemas para os quais existe documentação sobre as metodologias utilizadas, consegue apenas categorizar um número limitado de idiomas, sendo que o Vivíssimo consegue trabalhar com muitos mais mas não existe literatura que descreva as metodologias utilizadas. A não remoção de palavras vazias é a principal causa da existência de algum lixo que por vezes faz com que existam nomes de categorias pouco apelativos quando são introduzidas queries mais extensas. Mesmo assim é compensatório verificar que a metodologia proposta para o cálculo da importância das palavras funciona num conjunto tao restrito de informação e que é alvo das maiores incorrecções ao nível sintáctico. Contudo na maior parte dos casos o

algoritmo consegue filtrar com sucesso a maior parte das palavras que não são portadoras de conhecimento ou quando as mantém é porque elas estão muito ligadas a certas expressões como o caso de ”c programming”.

Sobre a fase de construção de modelos de utilizador, podemos referir que este é o tipo de área para o qual existe menos trabalho relacionado, devido talvez à frescura do assunto no que toca à personalização dos sistemas de pesquisa. A abordagem implementada foi mais uma vez fruto da tentativa de criar algo de novo e ultrapassar certas barreiras encontradas em metodologias existentes como a rapidez de processamento e o conjunto de informação necessária. Sabendo que só a informação relativa ao historial de pesquisas (queries) do utilizador não seria suficiente para a criação de um modelo de utilizador forte, decidimos jogar com as categorias geradas para cada query de modo a encontrar conhecimento oculto no historial de pesquisas. Desta forma foi possível encontrar informação que nunca havia sido introduzida nas queries. De um modo geral os modelos de utilizador criados automaticamente pelo nosso algoritmo representam os reais interesses destes. Contudo a estrutura dos modelos criados talvez não seja ainda a mais correcta pois quando o histórico de pesquisas é bastante extenso e muito variado, faz com que exista uma modelo de utilizador com bastantes áreas de interesse. Outro dos problemas é de ordem morfológica e relacionado com o saber que termos são mais genéricos. Excluindo estes problemas, o algoritmo proposto consegue identificar sem recorrer ao conteúdo dos textos lidos pelo utilizador os interesses explícitos e implícitos deste, fazendo com que a teoria fosse comprovada na prática.

Com este trabalho, demonstrou-se que ainda existem muitas áreas a explorar no que toca à melhoria dos sistemas de informação. As metodologias propostas são a prova disso pois ficou demonstrado que é possível criar um sistema de categorização independente de conhecimento com resultados satisfatórios e gerar modelos de utilizador a partir do bom uso da categorização dos resultados e do histórico de pesquisa dos utilizadores. Tal como em todos os trabalhos de investigação, nunca existe o alcance da perfeição. É sempre possível melhorar aquilo que já foi feito e o nosso trabalho não é excepção. No tópico seguinte são descritas as possíveis melhorias ao trabalho bem como as ideias para a continuidade deste.

6.2 Trabalho futuro

A motivação para este trabalho foi a de melhorar a experiência de utilização dos sistemas de pesquisa. Um dos objectivos propostos, a categorização, já é um passo nesse sentido pois com a sua aplicação permite ao utilizador centrar-se numa categoria de resultados que lhe seja mais chamativa de entre as diversas categorias apresentadas. O modelo de utilizador só passa a ser útil quando aplicado ao motor de pesquisa. Desta forma o sistema passa a possuir conhecimento sobre o utilizador e pode reordenar os resultados e categorias de acordo com aquilo que é mais do seu interesse.

Deste modo, o primeiro passo a dar na continuação do trabalho é adaptar este modelo de utilizador aos resultados iniciais da pesquisa. Isto pode ser feito através de heurísticas simples como em [28] onde é feita apenas uma soma dos termos que aparecem no snippet e num dos ramos da árvore, sendo dada maior relevância aos documentos que mais termos têm em comum com os que descrevem o perfil do utilizador.

O processo de categorização implementado tal como em muitas outras abordagens, padece por vezes de um número excessivo de categorias. Quando isso acontece as vantagens do seu uso desaparecem para o utilizador que se perde por entre as categorias. Para contornar este problema uma outra forma de agrupamento de snippets já está em estudo. Este tem por base um algoritmo do tipo Global K-Means com o uso da medida de similaridade InfoSimba [29] que pretende limitar as diferentes categorias aos diferentes significados dos termos da query.

Ao longo do desenvolvimento do trabalho muitas novas ideias foram surgindo e nem todas foram passíveis de ser utilizadas e experimentadas. Para optimizar a importância dada a cada palavra, podemos também efectuar para cada uma, um cálculo do tipo PageRank [47]. O objectivo seria para cada palavra guardar a lista de palavras com que ela ocorre numa janela para ambos os lados juntamente com as suas frequências. Para cada uma das palavras calcular-se-ia o seu peso tendo por base o nível de co-ocorrência que tem com outras palavras.

A extracção de palavras compostas efectuada de momento é feita sem poder existir saltos entre as palavras. Como foi visto na literatura, alguns algoritmos fazem a extracção de palavras compostas com a possibilidade de salto [45]. Se introduzirmos esta funcionali-

dade e dermos mais relevância às palavras compostas no nosso algoritmo de categorização, poderemos melhorar os resultados até ao nível dos nomes das categorias.

Relativamente à geração dos modelos de utilizador, para uma melhor filtragem dos interesses deste, podem ser utilizados os termos dos snippets dos documentos relevantes, cuja informação pode permitir tomar melhores decisões quanto à geração dos ramos da estrutura. Para ultrapassar os problemas relativos à posição que certas palavras devem ter na árvore, um inquérito pode ser efectuado e consoante isso ser tomada uma decisão mais consensual.

Bibliografia

[1] Artigo t3, google: O passado, o presente, o futuro. agosto 2009.

[2] http://search.carrot2.org/stable/search.

[3] http://ubi8.imc.pi.cnr.it.

[4] http://www.grokker.com.

[5] http://www.hollyscoop.com/michael-jackson/michael-jacksons-death-crashes-google.aspx.

[6] http://www.iboogie.tv/.

[7] http://www.internetnews.com/stats/article.php/1363881.

[8] http://www.kartoo.com/en/kartoo.html.

[9] http://www.mooter.com/.

[10] http://www.nlsearch.com/home.php.

[11] http://www.searchengineguide.com/wordtracker/top-500-search-engine-keywords-of-the-we-7.php.

[12] http://www.washingtonpost.com/wp-dyn/content/article/2007/07/23/-ar2007072301543.html.

[13] Nielsen announces may u.s. search share rankings, with total searches increasing 20 percent-over-year. *www.nielsen-online.com*.

[14] www.altavista.com.

[15] www.amazon.pt.

[16] www.ask.com.

[17] www.bing.com.

[18] www.clusty.com.

[19] www.google.com.

[20] www.yahoo.com.

[21] Lipai A. Web metasearch result clustering system. *Informatica Economica Journal*, pages 113–116, 2008.

[22] Lipai A. World wide web metasearch clustering algorithm. *Informatica Economica Journal*, pages 5–11, 2008.

[23] Sieg A., Mobasher B., and Burke R. Ontological user profiles for personalized web search. *Conference on Information and Knowledge Management Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 525–534, 2008.

[24] Singhal Amit. Modern information retrieval: A brief overview. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 24(4):35–43, 2001.

[25] Dreilinger D. and Howe A.E. Experiences with selecting search engines using metasearch. *Journal of ACM Transaction on Information Systems*, 15(3):195–222, 1997.

[26] Machado D., Barbosa T., Pais S., Martins B, and Dias G. Universal mobile information retrieval. *13th International Conference on Human Computer Interaction (HCII 2009)*, July 19-24 2009.

[27] Geraci F., Pellegrini M., Maggini M., and Sebastiani F. Cluster generation and cluster labelling for web snippets: A fast and accurate hierarchical solution. *String Processing and Information Retrieval, Web Clustering and Text Categorization*, pages 25–36, 2006.

[28] Liu F., Yu C., and Meng W. Personalized web search by mapping user queries to categories. *In Proceedings of the eleventh international conference on Information and knowledge management (CIKM)*, pages 558–565, 2002.

[29] Dias G., Alves E., and Lopes. Topic segmentation algorithms for text summarization and passage retrieval: An exhaustive evaluation. *22nd Conference on Artificial Intelligence (AAAI 2007)*, pages 1334–1340.

[30] Salton G. and McGill M. Introduction to modern information retrieval. *McGraw Hill*, 1983.

[31] Zeng H., He Q., Chen Z., Ma W., and Ma J. Learning to cluster web search results. *Annual ACM Conference on Research and Development in Information Retrieval Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 210–217, 2004.

[32] Goecks J. and Shavlik J. Learning users interests by unobtrusively observing their normal behavior. *In Proc. 5th international conference on Intelligent User Interfaces*, pages 129–132, 2000.

[33] Kim J., Oard D.W., and Romanik K. Using implicit feedback for user modeling in internet and intranet searching. *Tech. Rep., College of Library and Information Services, University of Maryland, College Park*, 2000.

[34] Ventura J. and Silva J. New techniques for relevant word ranking and extracting. *Progress in Artificial Intelligence, Springer verlag*, LNAI series 4874:691–702.

[35] Chunyu K. and Yorick W. The virtual corpus approach to deriving ngram statistics from large scale corpora. *Proceedings of 1998 International Conference on Chinese Information Processing*, 1998.

[36] Hyoung R. Kim and Philip K. Chan. Personalized ranking of search results with learned user interest hierarchies from bookmarks. *In WEBKDD Workshop*, SIGKDD Conf 2005.

[37] Tamine L., Boughanem M., and Zemirli N. Inferring the user interests using the search history. *LWA*, 1:108–110, 2006.

[38] Aktas M., Nacar M., and Menczer F. Using hyperlink features to personalize web search. *Advances in Web Mining and Web Usage Analysis*, pages 104–115, October 17, 2006.

[39] Atsumi M. Extraction of user's interests from web pages based on genetic algorithm. *In IPSJ SIG Notes (Information Processing Society of Japan, The Special Interest Groups Notes)*, 1997.

[40] Claypool M., Le P., Waseda M., and D. Brown. Implicit interest indicators. *In Proc. International Conference on Intelligent User Interfaces*, 2001.

[41] Morita M. and Shinoda Y. Information filtering based on user behaviour analysis and best match text retrieval. *In Proc. 17th ACM Annual International Conference on Research and Development in Information Retrieval*, pages 272–281, July 1994.

[42] Radovanovic M. and Ivanoic M. Cats: A classification-powered meta-search engine. *Advances in Web Intelligence and Data Mining*, (11):191–200, August 2006.

[43] Zamir O. and Etzioni O. Web document clustering: a feasibility demonstration. *Annual ACM Conference on Research and Development in Information Retrieval Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 46–54, 1998.

[44] Chan P. A non-invasive learning approach to building web user profiles. *In Proc. ACM SIGKDD International Conference*, pages 7–12, 1999.

[45] Ferragina P. and Gulli A. A personalized search engine based on web-snippet hierarchical clustering. *In Proceedings of WWW05, 14th International World Wide Web Conference*, pages 801–810, 2005.

[46] Campos R. and Dias G. Automatic hierarchical clustering of web pages. *In Proceedings of the ELECTRA Workshop associated to 28th Annual International ACM SIGIR Conference*, pages 83–85, August 19 2005.

[47] Brin S. and Page L. The anatomy of a large-scale hypertextual web search engine. *In Proceedings of 7th International World Wide Web Conference*, pages 107–117, 1998.

[48] Chakrabarti S. Mining the web: Analysis of hypertext and semi structured data. *Morgan Kaufmann*, 2003.

[49] Osinski S., Stefanowsky J., and Weiss D. Lingo: Search results clustering algorithm based on singular value decomposition. *Advances in Soft Computing, Intelligent Information Processing and Web Mining, Proceedings of the International IIS: IIPWM´04 Conference*, pages 359–368, 2004.

[50] Schockaert S., De Cock M., Cornelis C., and Kerre E.E. Clustering web search results using fuzzy ants. *Lecture notes in computer science*, 3172:342–349, 2004.

[51] Singh S., Murthy H., and Gonsalves T. Determining user's interest in real time. *International World Wide Web Conference Proceeding of the 17th international conference on World Wide Web*, pages 1115–1116, 2008.

[52] Tanner and Chris. Adaptive web personalization: Improving web personalization via user interest hierarchy and scoring techniques. Dec 2006.