Joaquim Melo Henriques de Macedo

Recuperação de Informação Textual Distribuída por Fontes Autónomas com Sobreposição

Tese de Doutoramento

Universidade do Minho
Escola de Engenharia
Departamento de Informática

Braga — Julho 2001

Joaquim Melo Henriques de Macedo

Recuperação de Informação Textual Distribuída por Fontes Autónomas com Sobreposição

Tese submetida à Universidade do Minho
para obtenção do grau de Doutor em Informática

Universidade do Minho
Escola de Engenharia
Departamento de Informática

Braga — Julho 2001

Título:

Recuperação de Informação Textual Distribuída por Fontes Autónomas com Sobreposição

Autor:

Joaquim Melo Henriques de Macedo

Tese de Doutoramento em Informática, Departamento de Informática, Escola de Engenharia, Universidade do Minho

Candidatura a Doutoramento aceite pelo Conselho Científico da Escola de Engenharia da Universidade do Minho em 19 de Janeiro de 1994

Orientador:

Vasco Luís Barbosa de Freitas, Professor Catedrático do Departamento de Informática da Escola de Engenharia, Universidade do Minho

Conclusão: Julho de 2001

Página Web: http://marco.uminho.pt/~macedo/tese.html

Abstract

Network Information Retrieval is a recent research area, intended for rethinking the established Information Retrieval field in the context of WWW and the increasing digitalization and Internet availability of conventional Libraries information sources. The main goal is to provide to the end-user with a *virtual centralized IR system*, building automatic tools to select from a large number of existing information sources, the relevant ones. Furthermore, a unique ranked result set must be returned to the user.

However, before distributed information retrieval systems can be of real use, solutions must be found for some existing problems.

A first issue is the selection of relevant sources for a given user query — **the database selection problem**. This selection is based up on source content summaries (or gathered information using past queries) and must be introduced instead of the simple broadcast of the user query, to scale when faced with a potential large number of information sources.

As different search engines may have particular query interfaces, the usual approach is the use of a minimum common functionalities as the query interface of the overall system. In such context, the mapping of queries at the broker level is a trivial task. A more complex approach but also more efficient one is to try and use all the query language facilities mapping queries to target search engines. In the last case, the **query mapping problem** must be solved.

Another important issue, is the **merging/fusion problem**, ie, the need of merging/fusion of returned results from potentially heterogeneous sources with different models, ranking algorithms and heuristics. The merging problem can also be envisaged in conventional IR, when two or more retrieval strategies are combined to achieve a better answer for a given query. Also possible different metadata fields must be converted to a given system established format. This last issue is the **metadata mapping problem**.

As in any distributed application, in distributed IR the replication is used to decrease response time and increase system reliability and availability. So, the replication transparency must be ensured at document level (in search engine or result fusion) and at collection level (in the broker when selecting databases, merging results or processing summaries). In other words, **the overlapping problem** must be solved.

The main goal of this thesis work is the study of overlapping on network information retrieval, mainly its relation with database selection and fusion of results. So, algorithms for allocation of documents, database selection and data fusion, considering the overlapping between the different component databases, are proposed and evaluated using standard methods and merit figures.

A relevant outcome of this work was the engineering of an architecture to be used as a testbed for hot network information retrieval issues. The resulting system, named Webtrieve (WEB reTRIEVE), is based on Vector Space Model and may be used as starting point for research, teaching or operational information retrieval initiatives.

Keywords: Distributed Information Retrieval, Uncoordinated Replication, Results Merging and Database Selection.

Resumo

A Recuperação de Informação em Rede é uma área recente de investigação, definida com a intenção de repensar o campo já estabelecido da Recuperação da Informação (RI) no contexto do WWW e da crescente digitalização e acesso na Internet de fontes de informação das Bibliotecas convencionais.

O principal objectivo é disponibilizar ao utilizador final um *sistema de RI centralizado virtual*, construindo programas que permitam seleccionar, de um largo número de fontes de informação, aquelas que são relevantes para uma determinada necessidade de informação do utilizador. Adicionalmente, deve ser devolvido ao utilizador um único conjunto ordenado de documentos pontuados.

Para que a recuperação da informação em rede se torne uma verdadeira alternativa à recuperação de informação centralizada, é necessário encontrar soluções eficientes e eficazes para um conjunto de problemas bem definidos.

O primeiro aspecto é a selecção das fontes relevantes para uma dada interrogação do utilizador — **o problema da selecção das Bases de Dados**. Esta selecção é feita com base em resumos dos conteúdos das fontes (disponibilizados pelas próprias fontes ou construídos com base em interrogações com um único termo), e deve ser introduzida para substituir a simples difusão da interrogação, não escalável na presença dum grande número de fontes de informação.

Como diferentes motores de busca podem ter linguagens de interrogação diferentes, a abordagem comum é usar um conjunto reduzido das funcionalidades comuns como linguagem de interrogação do sistema distribuído. Nesse contexto, a correspondência de interrogações é uma tarefa trivial. Uma abordagem mais eficiente, mas também mais complexa, é usar ao máximo a capacidade de cada motor. Neste último caso, **o problema da correspondência da interrogação** tem que ser resolvido.

Outra questão pertinente é **o problema da fusão dos resultados** devolvidos por fontes

potencialmente heterogéneas que usam modelos, heurísticas e algoritmos de pontuação diferentes. Este problema já se colocava na recuperação de informação tradicional quando se combinavam diferentes estratégias de recuperação de informação para melhorar a qualidade da resposta. Também é possível que os items de informação devolvidos por cada documento no conjunto de resposta variem de sistema para sistema, colocando-se **o problema da correspondência da meta-informação**.

Como em qualquer aplicação distribuída, na recuperação distribuída de informação a replicação é usada para diminuir tempos de respostas e aumentar a fiabilidade e disponibilidade do sistema. Portanto, a transparência da replicação tem que ser assegurada ao nível de cada documento e de cada fonte. Dito por outras palavras, **o problema da sobreposição** tem que ser resolvido.

O principal objectivo desta tese é o estudo da sobreposição na recuperação da informação em rede, principalmente no seu relacionamento com a selecção de base de dados e a fusão de resultados. Assim, são propostos algoritmos de selecção de bases de dados e fusão de resultados que consideram a sobreposição como parâmetro de entrada. Esses algoritmos são avaliados usando métodos e figuras de estilo normalizados. A introdução deste parâmetro permite melhorar a eficácia do resultado.

Um resultado colateral importante do trabalho realizado é a concepção e concretização duma arquitectura que pode ser usada como bancada para investigação e desenvolvimento dos tópicos mais em foco na recuperação de informação em rede, apresentados nos parágrafos anteriores. O sistema resultante, designado por Webtrieve (Web reTRIEVE), baseado no modelo do espaço vectorial, pode também ser usado para iniciativas pedagógicas ou de prestação de serviço de recuperação de informação.

Palavras-chave: Recuperação Distribuída de Informação, Replicação Descoordenada, Fusão de Resultados e Selecção de Bases de Dados.

À memória da minha mãe Amélia.

A todos os pés descalços de criança que, em campos minados ou no asfalto quente das cidades, escrevem a tese da sobrevivência...

Agradecimentos

Queria começar por agradecer ao Professor Vasco Freitas, pelo apoio e orientação deste trabalho. Como supervisor da tese, não poupou esforços na criação das condições necessárias para a sua realização.

O Professor Alan Smeaton, da Dublin City University, funcionou, ainda que informalmente, como um verdadeiro co-supervisor do trabalho, participando e de que maneira na sua orientação. Foi também inexcedível no apoio prestado e na disponibilidade demonstrada durante os três meses e uma semana (em três visitas) da minha estadia em Dublin, quando estive a trabalhar no Laboratório do Grupo *Multimedia IR*, sob a sua orientação.

Ao Professor Fernando Moura Pires da Universidade de Évora, pela disponibilidade e pelas sugestões dadas ao longo deste trabalho. Ao Professor Alexandre Santos, pelas inúmeras revisões, pelo apoio e incentivo constante. O Professor Pedro Henriques teve a gentileza de dedicar uma parte do seu tempo a rever e a comentar esta tese.

Não queria deixar de destacar a contribuição do meu colega António Costa que foi um autêntico cúmplice na realização deste trabalho... Desde uma ajuda preciosa na concepção e concretização do Webtrieve até as discussões acaloradas sobre os algoritmos, conceitos e mesmo a escrita da dissertação! Aos meus outros colegas do Grupo de Comunicações por Computador (em que realço a Maria João) e do Departamento de Informática pelas revisões do texto, apoio técnico e constante incentivo.

Ao meu pai Flávio e à minha filha Liliana pelo constante apoio e principalmente pelas prolongadas ausências suportadas durante os últimos anos. Aos meus familiares e amigos pelo constante apoio e incentivo. A Clara, a Fernanda e o Xandinho ainda tiveram que rever o *pretuguês*...

À Universidade do Minho, em particular ao Departamento de Informática, pelos meios postos à disposição. À Fundação Calouste Gulbenkian pelo suporte financeiro (sempre a tempo e horas!) concedido para as deslocações a Dublin.

Índice

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Motivação

Com o crescimento da Internet em tamanho e popularidade, os seus serviços de informação, com o WWW em posição de grande destaque, tornaram-se na principal fonte de informação para muitos utilizadores. Ao mesmo tempo que a sua utilização pelos mais diversos sectores da população se vai banalizando, os tipos de informações e serviços disponilizados vão transformando o Web cada vez mais num serviço de informação de cobertura universal. Uma outra vertente importante é o aumento da sua acessibilidade: computadores pessoais e de bolso, televisão interactiva e mais recentemente os telefones portáteis.

Ao mesmo tempo, assiste-se a um verdadeiro frenesim quer da parte do sector público e privado quer das pessoas individualmente, para disponibilização de informação e serviços através do WWW. Todos os dias aparecem milhões de novas páginas. Esta tarefa é facilitada pela existência de geradores de páginas cada vez mais sofisticados e também gateways que permitem a aplicações (Bases de Dados, principalmente) disponibilizar conteúdos dinâmicos. A Internet permitiu passar de uma situação em que o acesso à informação era extremamente difícil e limitado pela proximidade geográfica às fontes, para uma outra em que todo tipo de informação está potencialmente disponível a qualquer pessoa. Com a ligação à Internet, o utilizador fica sujeito a uma *sobrecarga de informação* .

A sobrecarga de informação põe na ordem do dia a necessidade de ferramentas e aplicações que permitam a cada utilizador descobrir rapidamente a informação que necessita em face da grande quantidade disponível. Essas ferramentas ou aplicações devem ser eficientes e eficazes para permitirem ao utilizador satisfazer a sua necessidade de informação em tempo útil.

A recuperação de informação (RI) [1] [SM83, vR95] é uma área das ciências da computação que se preocupa com a selecção, num universo de documentos disponíveis, do conjunto de documentos relevantes para uma necessidade específica de informação do utilizador.

Durante décadas, o desenvolvimento de sistemas de recuperação e armazenamento de informação foi feito para aplicações de âmbito bastante específico. Nos últimos anos, a comunidade que se dedica à investigação neste domínio de conhecimento, está confrontada com um novo e estimulante desafio: a concepção e concretização de soluções para um espaço de informação gigantesco, dinâmico e heterogéneo como a Internet.

A abordagem mais imediata e mais natural foi a disponibilização dos motores de busca tradicionais na Internet. Para esse efeito foi usado o paradigma cliente/servidor. Pelo facto do espaço de informação indexado ser o WWW e pela sua popularidade, o HTTP [BLFF96] foi a escolha quase obrigatória para protocolo de acesso. Nos dias de hoje, o quotidiano dos utilizadores passa por uma ou mais interrogações a serviços de busca na Internet.

Porém, com o passar dos anos, tanto os vários fornecedores de serviços como a própria comunidade de investigadores nesta área constataram, em face dos grandes volumes de informação existentes no WWW (da ordem dos terabytes), do seu dinamismo e do elevado ritmo de chegada de interrogações, a necessidade de migração[2] dos sistemas de RI convencionais para sistemas distribuídos de RI. Esta constatação não contraria o facto de actualmente os serviços de busca de informação mais populares da Internet serem baseados em sistemas de RI centralizados.

Para efectivar esta migração, impõe-se a procura de soluções eficazes e eficientes para uma série de novos problemas colocados pela distribuição. O principal objectivo é atingir níveis de eficácia, em termos de precisão e cobertura, comparáveis com os sistemas de RI centralizados. Por outro lado, torna-se imprescindível a obtenção de ganhos significativos em eficiência, escalabilidade e disponibilidade, que são afinal a grande motivação para a utilização dos sistemas distribuídos.

Com a importância crescente das *fontes de informação* acessíveis na Internet que inclui os meios de difusão massiva (rádio, jornais e televisão), bibliotecas digitais, discotecas e videotecas, a quantidade e diversidade das fontes vai continuar a aumentar, trazendo atrás uma larga população de utilizadores.

[1] Information Retrieval, em inglês

[2] a médio e longo prazo, pelo menos

Os sistemas de RI centralizados convencionais não conseguem responder em tempo útil quer a mudanças do espaço de informação indexado (novos documentos, documentos modificados), quer a interrogações de milhões de utilizadores. Para volumes de informação da ordem dos terabytes, o tamanho dos índices é também gigantesco.

Mesmo os serviços mais populares da Internet estão longe de indexar todo o espaço de informação WWW, como é indicado pelas estatísticas: HotBot 34%, Altavista 28%, Northern Light 20% e Excite 14% (fonte http://www.nua.net,1998). Apareceu mais recentemente um novo serviço, O Google, que se tem transformado na coqueluche da Internet, exibindo um elevado grau de cobertura. Não se dispõe, para já, das respectivas estatísticas.

Para além do volume, a diversidade da informação disponível vai exigir uma maior especialização dos motores de busca na representação e indexação da variedade dos conteúdos disponíveis. A diversidade não pode ser capturada por um único modelo de RI, que tem de usar uma abordagem generalista, reduzindo a sua utilidade. É necessária a utilização de modelos e esquemas de meta-informação que capturem as peculiaridades dos recursos de informação existentes.

Não é possível gerar índices de qualidade automaticamente, a menos que se usem heurísticas locais ou anotações manuais. Um exemplo é a indexação de frases baseadas em nomes locais (pessoas, organizações e lugares). Este problema complica-se bastante num ambiente com muitas línguas. Várias organizações, como as Bibliotecas por exemplo, dispõem de mão de obra para criação e/ou enriquecimento local de índices. Esta mão de obra não é utlizável por serviços de busca com cobertura total da Internet.

Algumas organizações possuem informação mais sensível, pelo que só permitem uma exportação controlada da sua informação. Esta situação vai-se tornar cada vez mais comum, com a introdução de mecanismos que salvaguardem os direitos de autor e com um maior número de empresas com serviços baseados na rede.

Uma abordagem possível para solucionar muitos dos problemas apresentados, é a distribuição dos serviços de recuperação de informação na Internet, isto é, vários servidores que indexam e respondem a interrogações de partições do espaço de informação total. O critério de partição pode ser geográfico, temático, por tipo de documento, etc...

Para responder ao problemas de escala da Internet, que é constituída por centenas de milhar[3] de organizações autónomas interligadas em rede, faz todo sentido a utilização de

[3] Milhões?

arquitecturas sem recursos partilhados [4] com a interligação em rede de vários (muitos) nós completamente autónomos. Este tipo de arquitecturas usam o mesmo tipo de filosofia que a própria Internet.

Uma outra vantagem desta arquitectura é a possibilidade da utilização da base instalada, nomeadamente o facto de sistemas centralizados já existentes poderem ser integrados no sistema distribuído, acrescentando apenas o módulo de software para comunicação, concretizado de acordo com o(s) protocolo(s) utilizado(s).

Cada nó da arquitectura distribuída funciona, por si só, como um serviço de busca autónomo, acedido quer pelos utilizadores da organização para pesquisar a sua Intranet como por utlizadores da mesma região ou que usem a mesma língua. Por outro lado, cada nó, em função do volume de informação a indexar, pode ter necessidade de usar técnicas de paralelização e estar mesmo disperso por vários computadores ou ser suportado por uma máquina multiprocessador.

Todavia, a distribuição deve ser *transparente* para o utilizador. A transparência da distribuição é conseguida se o serviço distribuído for visto pelo utilizador como *virtualmente centralizado*.

Com a distribuição dos serviços de busca, os índices são colocados em diferentes subcolecções. Dada uma interrogação, o sistema pode difundi-la prontamente para todas as sub-colecções. Esta solução é muito dispendiosa e ineficiente se existirem centenas ou milhares de sub-colecções. Nessas circunstâncias, impõe-se uma diminuição significativa do número de fontes a interrogar. O mecanismo responsável por esta diminuição do número de sub-colecções a interrogar é designado por selecção de bases de dados. Dada uma interrogação, o sistema deve seleccionar as sub-colecções relevantes para a construção da resposta. Trata-se dum problema tradicional de RI, embora em lugar de documentos, estejam as diferentes sub-colecções. Para que esta selecção seja possível, confronta-se a interrogação com os representantes das diversas sub-colecções.

A seguir, passa-se a interrogar cada colecção seleccionada. Em presença de vários servidores de interrogação, é necessário uma linguagem comum de interrogação. Caso não seja possível, podem-se usar mecanismos de conversão da interrogação original nas linguagens de interrogação suportadas pelos diferentes servidores [CCGMP96].

Finalmente, resta construir uma lista única com os documentos de resposta. Esta lista

[4] shared-nothing architecture

apresenta os documentos por ordem descrescente de pontuação, que é uma medida de similaridade entre cada documento e a interrogação. Esta tarefa pode ser bastante complexa uma vez que a pontuação atribuída varia de sistema para sistema e com o algoritmo usado. Mesmo usando o mesmo algoritmo, como existem parâmetros que dependem da colecção, as pontuações atribuídas podem não ser comparáveis.

Este último passo é chamado fusão ou junção de resultados. Existem várias abordagens limitadas pela quantidade de informação associada a cada item de resposta devolvida por cada engenho [MCF00a, BKFS95, SC97, Lee97, VGJL95].

A fusão ou concatenação dos dados já era usada nos sistemas centralizados de RI, para combinação da evidência de várias estratégias de interrogação. Neste contexto, usam-se vários motores de busca a indexar precisamente os mesmos documentos e obtém-se uma resposta mais eficaz do que a que seria obtida usando uma única estratégia de recuperação de informação.

Nos sistemas distribuídos esta função é indispensável para se poder devolver uma resposta única ao utilizador, que possa potencialmente incluir documentos de todo o espaço de informação.

Entenda-se por controlo da interrogação num sistema distribuíde RI, o conjunto de acções que precisam de ser desencadeadas em diversos componentes do sistema para responder a uma interrogação do utilizador.

Mesmo com a existência de um índice distribuído, o controlo da interrogação pode ser centralizado, sendo feito pelo utilizador ou por um agente seu. O controlo centralizado da interrogação pode constituir um entrave para o desempenho em arquitecturas distribuídas em larga escala. O controlo descentralizado da interrogação exige um algoritmo mais complexo para detecção de ciclos, sincronização da resposta, etc...

O uso de mecanismos importantes da recuperação de informação como a realimentação de relevância e outras técnicas de modificação da interrogação e mesmo o uso de medidas de eficácia como a precisão e cobertura, tornam-se mais difíceis com o controlo descentralizado da interrogação.

A distribuição dos índices torna necessária a introdução de protocolos de transferência de índices e mecanismos para caching de índices. Num ambiente bastante dinâmico como a Internet, são também úteis os algoritmos para actualização incremental de índices.

Com o tamanho actual e a taxa de crescimento do Web, são necessárias ferramentas eficientes para colecta distribuída dos dados. A colecta distribuída dos dados tem uma série de requisitos nomeadamente: um formato normalizado para a meta-informação e índices dos documentos, regras estabelecidas para indexação do conteúdos, critérios de partição do espaço de informação pelos diferentes colectores e protocolos de transferência da informação para os diferentes motores de busca. Há alguns progressos nesta direcção... Os URCs[SM94, Sol98] do IETF, os Metadados Dublin Core[WM97], os SOIFs do Harvest[HMW+95] ou os RDFs do W3C[LS99], são bons pontos de partida para formatos normalizados de meta-informação de documentos. Outra proposta de norma a considerar é o PICS[RM96] pela sua versatilidade.

Uma insuficiência importante da maioria dos formatos propostos é a falta de suporte à frequência de ocorrência dos termos, necessária para ponderação dos termos na pontuação dos documentos. Estas insuficiências podem ser resolvidas com as propostas feitas no âmbito do STARTS[GCGM+97].

O gatherer/broker do Harvest[HMW+95] é um modelo possível para partição do espaço de informação e interacção entre o colector e o indexador. A carga dos robots sobre os servidores de informação é apenas regulada por uma norma de exclusão de robots. Para minimizar esta carga, precisam de ser desenvolvidos e normalizados um conjunto de protocolos de registo e notificação.

Outros aspectos mais genéricos de sistemas distribuídos como nomeação, replicação, segurança, entre outros, precisam de ser estudados no contexto particular dos sistemas de recuperação de informação.

Há trabalho importante e significativo já realizado com o objectivo da distribuição dos sistemas de RI. Os trabalhos da comunidade da investigação em redes e aplicações distribuídos são orientados prioritariamente para os aspectos arquitecturais e de engenharia da distribuição de RI e o seu suporte na Internet [WFS96a, FSW96, HMW+95, PFG+94, Kun95].

A contribuição da comunidade de RI prende-se com os aspectos algorítimicos resultantes da distribuição, nomeadamente os problemas da fusão e selecção e os modelos para representação das sub-colecções e respectiva comunicação [GGMT94, GGM95, VF95, Fuh96, Bau97, Bau99, CCGMP96, GCGM+97, MZ95, CLC95, CPV98, CPC+99].

1.2 Contexto da tese

Num espaço de informação cooperativo e de grande dimensão como a Internet, é natural a existência de sistemas de interrogação diferentes, cada um deles concebido para um determinado tipo de informação e audiência.

Pretende-se, entretanto, que o conjunto de motores de busca cooperem de alguma forma no sentido de poupar o utilizador da complexidade resultante da distribuição. A transparência da distribuição é conseguida pela introdução dum novo componente, o encaminhador de interrogações (EI), que recebe a interrogação do utilizador, selecciona as fontes que considera relevantes, interroga-as, junta os respectivos resultados numa única resposta que devolve ao utilizador. Este ambiente pode ser visto na Figura 1.

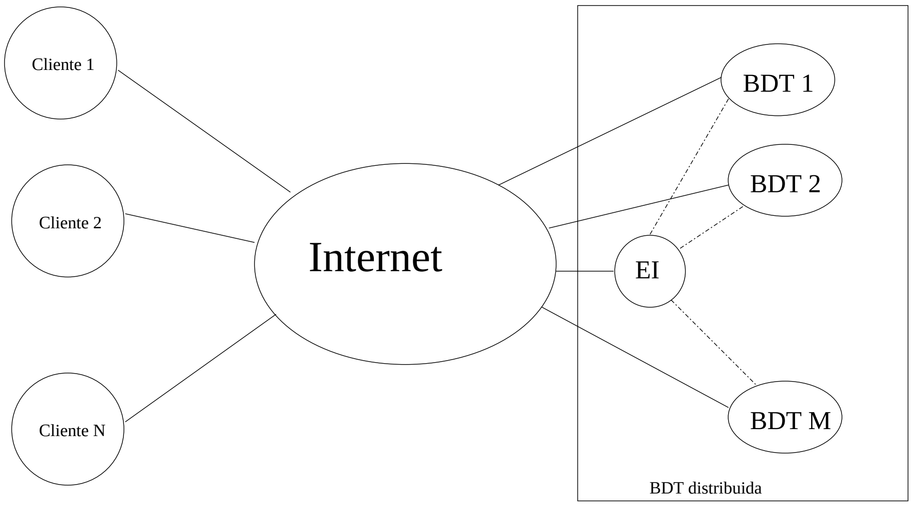


Figura 1: O contexto da tese: EI(Encaminhador de Interrogações) BDT(Base de Dados de Texto)

O espaço de informação a indexar usa a replicação de documentos para diminuir a sobrecarga da sua transferência através da rede. Os próprios motores, pelo facto de serem geridos autonomamente, podem indexar partições sobrepostas do espaço de informação. Pode assim existir sobreposição entre as diversas fontes com que o sistema de RI tem que lidar no sentido de oferecer a transparência da replicação ao utilizador e a alguns dos seu componentes.

O trabalho realizado nesta tese enquadra-se na problemática da RI em ambiente com múltiplas fontes, preocupando-se quer com os aspectos arquitecturais de concepção e concretização deste tipo de sistema, quer com os aspectos mais algorítmicos das funções a realizar no EI, considerando a existência de sobreposição entre as diferentes fontes.

O trabalho foi realizado tanto do ponto de vista da componente mais teórica dos algoritmos e dos mecanismos a usar como da sua concretização num protótipo, cuja eficácia foi avaliada experimentalmente usando as medidas tradicionais de precisão e cobertura.

1.3 Objectivos

Que seja do conhecimento do autor, não há nenhum trabalho anterior da área da recuperação distribuída da informação em que se considere a existência da replicação não coordenada. Considera-se, isso sim, uma arquitectura de replicação que permita a utilização das réplicas pelo algoritmo de selecção [LM99].

Com o presente trabalho, pretende-se estudar a influência da sobreposição entre as diferentes fontes indexadas no desempenho global do sistema de RI distribuído, nomeadamente nas funções de selecção e fusão.

A hipótese colocada é que se a sobreposição entre as diferentes BDTs for usada como parâmetro dos algoritmos de fusão de resultados e selecção de bases de dados, se consegue melhorar a eficácia da resposta devolvida ao utilizador pelo sistema distribuído de RI.

Sabe-se da RI centralizada que a fusão de resultados de vários motores que indexam precisamente os mesmos documentos é mais eficaz que um engenho simples [BKFS95, Lee97]. Existem também, disponíveis na literatura, diversos algoritmos concebidos para a fusão de resultados de motores que indexam espaços de informação completamente disjuntos ou em que não se utiliza como parâmetro a sobreposição entre o conjunto de resultados sobrepostos [CHT99, CLC95]. Pretende-se nesta tese usar essa informação no algoritmo de fusão para melhorar a eficácia dos resultados obtidos.

Na selecção das BDTs relevantes para uma dada interrogação, as BDTs são seriadas por ordem de importância para a interrogação. Nos diversos algoritmos de selecção mais citados na literatura, não se considera a possibilidade de existência de sobreposição entre as diferentes BDTs indexadas, devido a replicação não coordenada [Bau99, Pow01, GGMT94, CPC+99]. Usando como único critério o conteúdo das BDTs, pode acontecer que por exemplo duas

BDTs sejam seleccionadas precisamente pelos mesmos documentos, degradando o desempenho do algoritmo. Neste tipo de cenários, a utilização da informação de sobreposição pode prevenir esta diminuição da eficácia da selecção.
Para que a sobreposição possa ser usada pelo algoritmo de selecção, é indispensável o seu conhecimento antecipado ou pelo menos ter uma ideia da sua ordem de grandeza. Esse conhecimento corresponde apenas à sua estimação, porque não se pode conhecer à priori que sobreposição existirá nas respostas do subconjunto de BDT interrogadas. Por esse facto, é necessário estudar técnicas de estimação da sobreposição das BDTs com a informação disponível no componente do sistema distribuído de RI responsável pela selecção que é o EI.

1.4 As contribuições desta tese

Como já foi referido anteriormente, os sistemas de recuperação distribuída de informação têm ainda um longo caminho a percorrer para que se constituam como uma verdadeira alternativa aos tradicionais sistemas centralizados. Esse percurso inclui quer o estudo de aspectos mais teóricos relacionados com algoritmos e mecanismos para usar nos componentes introduzidos pela necessidade da distribuição, quer os aspectos mais práticos da engenharia e concretização dum sistemas distribuído de RI.
O trabalho realizado tem como ponto de partida a engenharia dum sistema distribuído de busca e os problemas decorrentes da sua concepção e concretização. Usando este sistema, obtém-se a sensibilidade suficiente para tentar encontrar soluções próprias para alguns dos problemas mais importantes colocados pela distribuição, nomeadamente a selecção de bases de dados e a fusão de resultados.
Estes problemas são resolvidos considerando a existência de sobreposição entre as diferentes bases de dados de texto (BDT) e que o nível de sobreposição é um parâmetro importante nos algoritmos de selecção e fusão. Ao contrário do trabalho já existente em que propõe arquitecturas de replicação [LM99], aqui parte-se do princípio que a quantidade de réplicas e a sua localização é descoordenada.
As contribuições deste trabalho são as seguintes:

1. É apresentada uma arquitectura completa para busca distribuída utilizável no WWW, concretizada num protótipo [MCF98, MCF00b];

2. São apresentadas diferentes estratégias para a estimação da sobreposição estática entre as BDs;
3. É proposto um algoritmo de fusão de resultados que usa a sobreposição entre os diferentes conjuntos de resultados para melhorar a eficácia da fusão [MCF00a];
4. É proposta a utilização da informação de sobreposição como parâmetro de algoritmos de selecção, melhorando a sua eficácia [MCF01];
5. É proposto um algoritmo de selecção que usa como pressuposto uma distribuição normal da similaridade dos documentos com a interrogação nas diferentes BDs; e,
6. É introduzida uma referência de base para avaliar a selecção, considerando a sobreposição [MCF01].

Os vários mecanismos e algoritmos propostos são avaliados usando as metodologias normalizadas e comparados com outros apresentados na literatura.

1.5 A estrutura do documento

A estrutura desta dissertação está apresentada na Figura 2.
No capítulo 1 apresenta-se sumariamente a temática da RI distribuída, o domínio de conhecimento onde se insere esta tese. Introduz-se igualmente o contexto, objectivos, contribuições e estrutura da tese.
No capítulo 2 definem-se, de uma forma simples mas precisa, os conceitos básicos e a terminologia desta área das ciências da computação, apresentando-se posteriormente, de forma sumária, o trabalho relacionado.
No capítulo 3, apresenta-se uma arquitectura que se entende cumprir os requisitos para constituir uma verdadeira bancada de experimentao e estudo dos problemas mais importantes que se colocam na recuperação distribuída de texto. Esta arquitectura corresponde à especificação funcional do que foi concretizado como protótipo para a realização das experiências feitas neste trabalho.
As funções específicas da distribuição estão destacadas e apresentadas com maior detalhe em capítulos próprios. Como são tópicos com relativa autonomia, a opção tomada foi a transformação desses capítulos em auto-contidos, com o seu próprio enquadramento, trabalho relacionado, algoritmos propostos, concretização, resultados e discussão. Evita-se

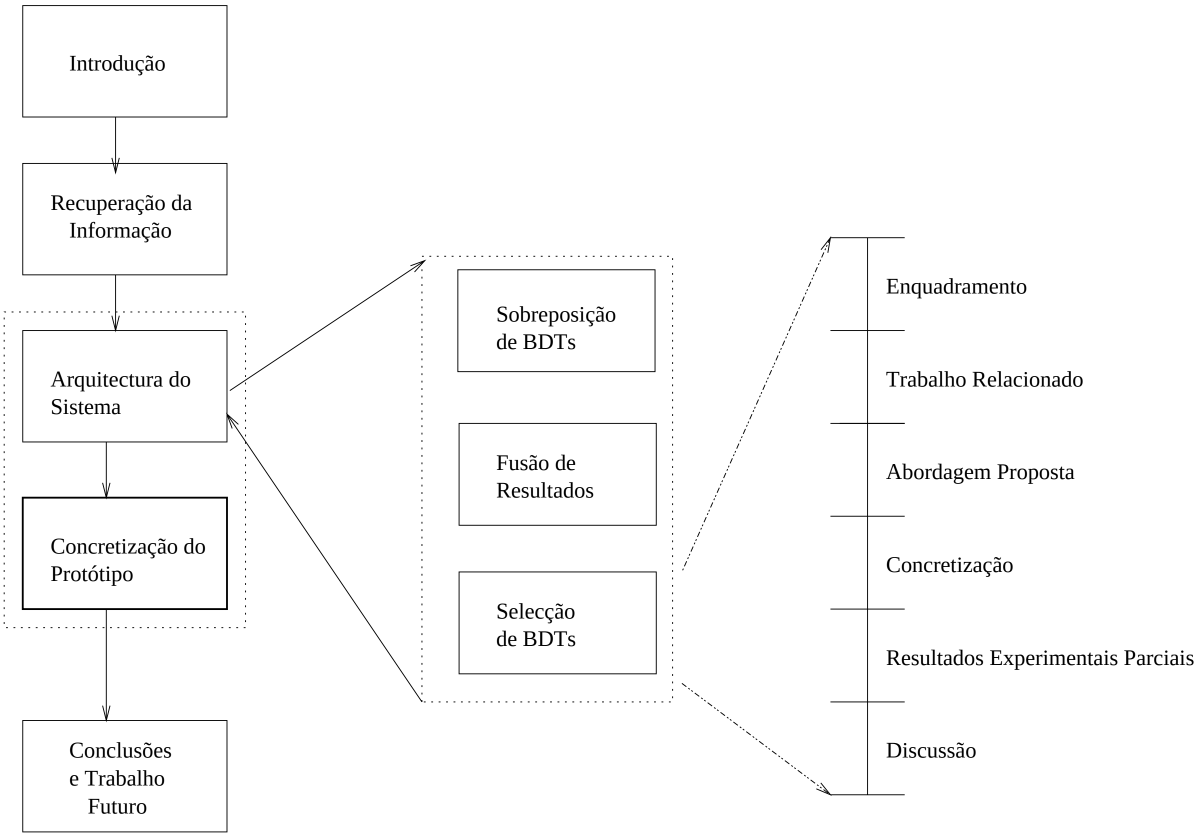


Figura 2: A estrutura da tese

desta forma que o leitor tenha que ter presente o que foi apresentado noutros capítulos, para conseguir uma visão completa de cada tópico. Por esta razão, os capítulos sobre a concretização do protótipo e conclusões estão com um enquadramento mais global.

No capítulo 4, define-se a sobreposição entre as diferentes fontes, distinguindo-se a sobreposição estática entre as fontes da dinâmica, que ocorre entre as respostas à cada interrogação. Apresentam-se alguns métodos que permitem a estimação da sobreposição estática em ambiente operacional, devido à indisponibilidade da informação necessária no Encaminhador de Interrogações. Incluem-se algumas estratégias de alocação e replicação cuja única finalidade é criar os cenários de experimentação indispensáveis neste trabalho. Por fim, apresenta-se alguns resultados experimentais e e também alguns comentários a respeito dos métodos de alocação e replicação utilizados.

No capítulo 5, aborda-se a problemática da fusão, definido-a formalmente e apresentando as estratégias usadas em ambientes de total sobreposição ou de completa disjunção entre as fontes. Apresenta-se o algoritmo de fusão proposto neste trabalho e compara-se a sua eficácia com os métodos convencionais.

O capítulo 6 trata da selecção de bases de dados. Define-se o problema e propõe-se um algoritmo que é avaliado de acordo com a metodologia usual. Posteriormente, modifica-se um algoritmo bem conhecido, no sentido de introduzir a informação de sobreposição estática, e mostra-se que se consegue melhoramentos na eficácia dos resultados.

Nos capítulos 7 e 8 apresentam-se os aspectos de concretização mais gerais, as conclusões e perspectivas de trabalho futuro.

No final da tese, apresenta-se um glossário de terminologia usada, no sentido de precisar a sua utilização na língua portuguesa (que muitas vezes varia de acordo com o grupo de investigação) .

Chama-se particularmente a atenção para a Página Web da tese[5] onde, para além do resumo, se disponibilizam a versão electrónica da tese (em .ps e .pdf), alguns artigos relacionados e mais informação a respeito da sua componente experimental. Alguma desta informação era tradicionalmente apresentada como anexo da tese.

[5] http://marco.uminho.pt/~macedo/tese.html

Capítulo 2

Recuperação da Informação

Sumário

Neste capítulo, não se pretende fazer uma apresentação exaustiva da recuperação da informação que, para além de maçadora, não substituiria a consulta de alguns bons livros que existem sobre o assunto. Para além duma apresentação sucinta dos vários modelos e métodos usados em RI introduz-se, de forma simples mas precisa, os conceitos básicos e a terminologia que serão usados ao longo deste trabalho. Na parte final, introduz-se a problemática da recuperação distribuída da informação que é temática principal desta tese. São incluídas algumas das referências bibliográficas mais importantes para permitir aos interessados aprofundar, se necessário, alguns dos assuntos apresentadas.

2.1 Introdução

A recuperação de informação[1] (RI) é um termo muito vago e de largo espectro, usado em muitos contextos. O objecto de estudo desta tese é a teoria subjacente aos sistemas automáticos de recuperação de informação [SM83, BYN99, vR95], que permitem encontrar os documentos relevantes, face a uma necessidade específica de informação do utilizador.
O utilizador tem que traduzir a sua necessidade de informação para a linguagem suportada pelo sistema automático de RI. Conforme se trate dum sistema de filtragem de informação ou dum motor de busca, a necessidade de informação transforma-se num *perfil de informação* ou numa *interrogação*.

[1] Information Retrieval, na terminologia inglesa

Sistema de RI	Interface	Necessidade de Informação	
		Tipo	Formulação
Pesquisa	Interactiva	Esporádica	Interrogação
Filtragem	Em deferido	Permanente	Perfil de Informação
Navegação	Interactiva	Indefinida	Percurso

Tabela 1: Sistemas de Recuperação de Informação

O tipo de formalização exigida para a necessidade de informação, depende também do tipo de sistema de recuperação de informação que estiver a ser utilizado.

Se se tratar de um sistema de filtragem de informação, o sistema mantém o registo do perfil de informação de cada utilizador, notificando-o cada vez que detecta no seu fluxo algum documento considerado relevante. O sistema de filtragem de informação ou de disseminação selectiva da informação necessita, para esse efeito, de confrontar todos os perfis de utilizadores registados com cada um dos documentos do seu fluxo. Um perfil de informação é uma forma razoavelmente completa e complexa de especificar uma necessidade de informação e adequa-se, tal como o modelo de funcionamento de um sistema de dessiminação selectiva da informação, a ser utilizado para necessidades de informação permanentes.

Um motor de busca [2] selecciona, de entre os documentos registados, aqueles que são considerados relevantes para a interrogação do utilizador. Pela simplicidade das interrogações (frequentemente menos do que meia dúzia de palavras) e pelo modelo de operação dos motores de busca, este tipo de sistema é usado para satisfazer necessidades de informação mais esporádicas.

Estes dois tipos de sistemas, embora tenham modos de funcionamento diferentes, tratam o mesmo problema que é a determinação, num universo de documentos, do conjunto de documentos relevantes para uma determinada necessidade de informação do utilizador [BC92].

Existe um terceiro tipo de sistemas que oferece uma interface mais adequada à navegação[3], em que os de documentos são agrupados em categorias ou hierarquias de categorias. Apesar de muitos dos serviços de classificação disponíveis serem semi-automáticos ou por vezes manuais, a construção de sistemas automáticos de classificação é baseada nos mesmos pressupostos e conceitos dos restantes sistemas de recuperação de informação. As diferenças

[2] search engines, na terminologia inglesa
[3] browsing, na terminologia inglesa

Visceral	sentido de preocupação
Consciente	área definida de indecisão
Formalizada	explicitada em termos concretos
Executável pelo sistema	traduzida para a linguagem do sistema

Tabela 2: Estados de desenvolvimento de uma necessidade de informação

essenciais entre esses sistemas estão ilustrados na Tabela 1.

Não é fácil aos utilizadores explicitar a sua necessidade de informação, que muitas vezes é apenas visceral (ver Tabela 2) [Tay62]. Isto é, um utilizador sabe que precisa de informação mas não consegue sequer a sua definição em linguagem natural. Há ainda a dificuldade acrescida de traduzi-la para a linguagem suportada pelo sistema que, por vezes, não é suficiemente amigável ou simplesmente não se adequa. Normalmente, a tradução feita pelos utilizadores da sua necessidade de informação é imprecisa. A utilidade de um sistema de recuperação de informação é, por isso, muito dependente da facilidade de uso da sua linguagem de interrogação.

A dificuldade da concretização destes sistemas automáticos começa pela subjectividade da noção de relevância. A relevância varia de pessoa para pessoa. Quantas vezes duas pessoas concordam na relevância de um documento para uma dada interrogação? Por vezes a relevância varia de ocasião para ocasião.

Uma outra dificuldade pode resultar do grande número de documentos e de interrogações ou perfis não permitirem a comparação directa, por questões de eficiência da resposta. Por esse facto, é necessário encontrar representantes tanto para os documentos como para as interrogações. Os representantes podem e devem ser mais compactos (mais fáceis de armazenar) e mais fáceis de processar que os objectos por si representados e são dependentes do modelo subjacente ao sistema de recuperação de informação. Uma analogia para o representante de um documento são as fichas usadas tradicionalmente nas Bibliotecas para os livros e usadas pelos bibliotecários para satisfazer a nossa necessidade de informação. Teríamos de esperar bastante mais tempo se o objecto de consulta directa fossem os livros...

Embora a noção de relevância seja binária (um documento é relevante ou não), se o sistema

encontrar muitos documentos relevantes precisa de os ordenar de alguma forma, devolvendo os mais importantes ao utilizador. Para isso precisa de os pontuar. Muitos sistemas pontuam todos documentos e consideram como relevantes os mais pontuados.
Na secção seguinte, apresentam-se os modelos usados para construção de sistemas de recuperação de informação. Esses modelos determinam quer a linguagem de interrogação utilizada e os representantes usados (para documentos e interrogações), quer os pressupostos conducentes à pontuação dos documentos, quando existirem.

Antes de passar à secção seguinte, vamos distinguir a recuperação de dados, objecto da disciplina de Bases de Dados, da recuperação de informação uma vez que são confundidas muitas vezes. Esta confusão resulta da fronteira entre estes dois tipos de sistemas ser bastante ténue.
A recuperação de dados, quando usada no contexto da recuperação de informação, consiste simplesmente em determinar quais os documentos, de entre os disponíveis na colecção, que contêm as palavras chave incluídas na interrogação do utilizador. Frequentemente, este modelo de funcionamento não é suficiente para satisfazer as necessidades de informação do utilizador. De facto, um utilizador de um sistema de recuperação de informação está mais interessado em encontrar informação relevante sobre determinado assunto do que documentos que satisfaçam determinada interrogação. A recuperação de dados usa uma linguagem precisa baseada numa expressão regular ou expressão algébrica e tem como objectivo encontrar todos os documentos que cumprem as condições especificadas. Na recuperação de informação, são tipicamente usadas interrogações em linguagem natural nem sempre bem estruturadas e por vezes semanticamente ambíguas confrontadas com documentos nas mesmas circunstâncias. Os documentos devolvidos têm normalmente um grau de incerteza. Esta distinção leva a considerar a recuperação de dados como determinística e a recuperação de informação como probabilística. Não é por acaso que a teoria das probabilidades ocupa um lugar relevante na recuperação de informação, o que não acontece nas bases de dados convencionais. Por exemplo, experimente apresentar uma mesma ideia através de um parágrafo em duas ocasiões diferentes e observe o resultado.

2.2 Modelos

Os vários modelos de recuperação de informação baseiam-se na comparação directa entre os documentos (ou os seus representantes) e a interrogação.

	d_1	d_2	$\ldots$	d_j	$\ldots$	d_N
t_1	$r_{1,1}$	$r_{1,2}$	$\ldots$	$r_{1,j}$	$\ldots$	$r_{1,N}$
t_2	$r_{2,1}$	$r_{2,2}$	$\ldots$	$r_{2,j}$	$\ldots$	$r_{2,N}$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
t_i	$r_{i,1}$	$r_{i,2}$	$\ldots$	$r_{i,j}$	$\ldots$	$r_{i,N}$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
t_M	$r_{M,1}$	$r_{M,2}$	$\ldots$	$r_{M,j}$	$\ldots$	$r_{M,N}$

Tabela 3: Matriz termos (t) x documentos (d); (r_{ij}) é um registo de ocorrência

Tradicionalmente, essa comparação é feita usando estruturas de dados onde estão representados os termos dos documentos e das interrogações.

Os termos representados nessas estruturas de dados são derivados das palavras que ocorrem nos documentos ou interrogações. Das palavras que ocorrem nos documentos, são retiradas aquelas que não têm nenhuma semântica associada, isto é as palavras funcionais como artigos, preposições, advérbios. As palavras funcionais são bastante comuns, não tendo nenhuma capacidade de descriminação entre os documentos. As restantes (substantivos, adjectivos e verbos) são normalmente radicalizadas (transformadas na palavra de que são derivadas) e o resultado dessa operação constitui os termos.

Os termos podem ser ainda qualificados pelo número de ocorrências, posições no documento, componente do documento em que ocorrem, variando a informação armazenada de acordo com as operações disponibilizadas pelo sistema.

Uma outra operação que pode ser realizada sobre os termos é a utilização de sinónimos (simultânea ou alternativamente) e conceitos subjacentes, que podem ser extraídos de um simples dicionário ou de um thesaurus.

Perde-se bastante informação quando se passa dum documento para o seu representante ou mesmo duma interrogação para o seu representante. Por este facto e, pelo que já foi dito a respeito da imprecisão das interrogações, não devemos ficar surpreendidos se um motor de busca nos devolver só documentos irrelevantes.

Independentemente do modelo utilizado, o conjunto de documentos que constitui uma colecção é processado previamente para se construir um ficheiro invertido [HFBYL92], que não é mais que uma forma compacta de representação duma matriz termos versus documentos (ver Tabela 9). Para cada termo, existe uma lista de ocorrências[4] com um registo para cada

[4]Posting List, na terminologia inglesa

	Bíblia	GNUlib	TREC
Nºde documentos	31 102	64 267	742 358
Nºde termos	884 988	2 570 939	333 856 749
Termos distintos	9 020	47 064	538 244
Tamanho(Mb)	4,33	14,05	2054,52

Tabela 4: Estatísticas de algumas colecções

documento em que o termo ocorre. O registo contém uma identificação interna do documento, o número de ocorrências e opcionalmente informação de localização (no documento) de cada ocorrência.

A construção do ficheiro invertido pode ser feita usando diversos métodos. Pode, para além disso, ser baseada na intercalação de vários ficheiros parciais, correspondendo cada um deles à inversão dum lote de documentos. A grande quantidade de termos e documentos existentes numa colecção exige alguns cuidados na gestão dos recursos de disco e memória necessários, em face dos disponíveis [WMB94]. Apresenta-se, para exemplificar, as estatísticas de algumas colecções na Tabela 4. A vírgula é usada para separar os milhares e o ponto as casas decimais.

O problema central da recuperação de informação é distinguir os documentos relevantes dos não relevantes para uma dada interrogação. Conforme já foi dito, isso é feito com base numa função de pontuação que ordena os documentos de acordo com a sua relevância. As premissas conducentes ao método de pontuação dos documentos varia de acordo com o modelo utilizado.

Os modelos clássicos de recuperação de informação são o booleano, o vectorial e o probabilístico.

O Modelo Booleano [War92] é um modelo simples de recuperação da informação baseado na teoria dos conjuntos e na álgebra booleana. Uma interrogação consiste num conjunto de termos ligados por operadores booleanos. Um documento é considerado relevante se for verdadeiro para a interrogação. Devido à sua simplicidade e pelo facto de ser intuitivo, o modelo booleano foi o primeiro a ser usado em sistemas de pesquisa bibliográficos. Apesar desta simplicidade, apresenta um certo número de problemas [VGB61]. A decisão binária a respeito da relevância dos documentos não permite atribuir-lhes uma pontuação. Assim, não é possível controlar a importância dos termos na interrogação ou nos documentos.

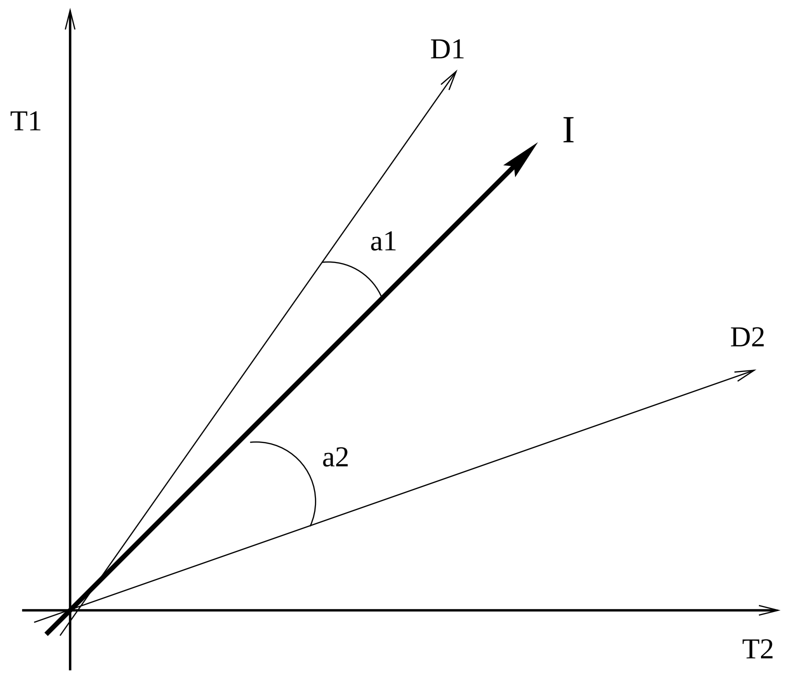


Figura 3: Ilustração gráfica do modelo vectorial

Uma outra dificuldade resulta do conhecimento informal, da parte dos utilizadores, dos operadores booleanos o que não lhes permite, em muitos casos, expressar correctamente a sua interrogação.

O Modelo do Espaço Vectorial [SL68, SY73] tem como princípio básico considerar interrogações e documentos como vectores. As interrogações são frases em linguagem natural e são também consideradas documentos. A transposição destes dois tipos de objectos para um espaço vectorial é conseguida com base em algumas simplificações nomeadamente a que assume que os termos são linearmente independentes. A independência dos termos é uma simplificação pois pode existir uma associação semântica ou sintáctica entre dois termos diferentes. No entanto, é esta simplificação que permite tirar partido das técnicas da álgebra vectorial.

Uma ilustração gráfica do Modelo do Espaço Vectorial (MEV) com 2 termos é apresentada na Figura 3. A cada termo no documento ou na interrogação, pode ser associado um peso que pode traduzir a sua presença, importância ou representatividade. O modelo não faz qualquer pressuposto a respeito da forma de estabelecer a importância (ou peso) dos termos. As escolhas mais comuns são a binária (termo ausente ou presente), a frequência no documento ou o produto da frequência do termo[5] [SL68] com a potência de descriminação[6] (inversamente proporcional ao número de documentos em que ocorre) [Jon72].

O MEV constitui uma larga classe de métodos de recuperação de informação, cada um dos quais com um método de indexação e com uma função de unificação. O método de indexação gera vectores de descrição (a partir dos documentos e interrogações) e a função de unificação compara os vectores de descrição de cada documento com o da interrogação e pontua cada documento. Por cada interrogação, os documentos são apresentados por ordem decrescente de pontuação ao utilizador [Har92a].

A função de unificação pode ser traduzida como inversamente proporcional à distância entre o documento e a interrogação no espaço vectorial. Para traduzir distância em pontuação pode-se pensar numa relação do tipo $p = b^{-d}$ [7] ou então no ângulo entre os vectores, a tradicional medida do coseno [SM71]. Se o ângulo for aproximadamente igual a zero, significa que a importância relativa dada aos termos é aproximadamente a mesma. O sistema de RI baseado no modelo vectorial mais conhecido é o SMART [Buc85]. Grande parte da investigação realizada no modelo vectorial foi suportada por ele [SM71, YS76].

Conforme já foi dito, existe uma grande imprecisão quer na formulação da interrogação quer na representação dos documentos, e a relevância de um documento relativamente a uma interrogação depende de muitos factores, pelo que é difícil encontrar um modelo exacto. Por este facto, não causa nenhuma surpresa que a abordagem probabilística seja uma das mais bem sucedidas.

A abordagem probabilística [RJ76] consiste na aplicação directa da teoria das probabilidades aos sistemas de RI. Baseia-se no pressuposto de que a distribuição de ocorrências dos termos da interrogação nos documentos nos diz alguma coisa a respeito da sua relevância. O Modelo Probabilístico (MP) tem a vantagem de usar uma teoria bem formalizada para o desenvolvimento dos algoritmos para estes sistemas. A utilização da teoria das probabilidades é uma escolha natural já que se baseia no raciocínio por evidências. Isto é resumido pelo princípio da ponderação probabilística introduzido por Cooper [Coo94]:

Se a resposta dum sistema de RI a cada interrogação é um conjunto de documentos por

[5] acrónimo tf(term frequency)
[6] acrónimo idf(inverse document frequency)
[7] sendo p a pontuação, b uma constante e d a distância

ordem decrescente de probabilidade de relevância, e se as probabilidades são calculadas com tanta precisão quanto possível com os dados disponibilizados pelo sistema para esse efeito, então a eficácia geral do sistema será a melhor que se pode conseguir com esses dados.

Dada uma interrogação i, o modelo probabilístico atribui a cada documento d uma pontuação proporcional a $\frac{P(d-relevante-para-i)}{P(d-irrelevante-para-i)}$. Essa pontuação minimiza a probabilidade de serem devolvidos documentos irrelevantes ao utilizador.

A forma como é calculada essa probabilidade [CH79] não vai ser detalhada, mas baseia-se no teorema de Bayes e nas probabilidades de ocorrência dos termos. Mais uma vez, o facto de se assumir a independência dos termos facilita o cálculo da obtenção da fórmula de pontuação global com a remoção das probabilidades condicionadas.

O modelo booleano é considerado o mais débil dos três. Embora haja quem diga que o modelo probabilístico é o mais eficaz dos vários modelos, parece haver um quase consenso na comunidade de RI sobre o melhor desempenho do modelo do espaço vectorial, se o indicador usado for a popularidade.
Há uma série de novos modelos, desenvolvidos a partir destes três, que tentam, na maior parte dos casos, ultrapassar as suas limitações e acrescentar novas funcionalidades.
A interpretação clássica dos operadores booleanos é muito restritiva. O modelo booleano clássico não é muito eficaz nem fácil de usar. Foram propostos uma série de modelos baseados na generalização dos operadores booleanos no sentido de aumentar a eficácia dos resultados. Como exemplo, pode apresentar-se a utilização da teoria dos conjuntos fuzzy [OMK91] e o modelo booleano estendido [SFW83] que interpreta as operações booleanas como distâncias algébricas. As experiências demonstraram que cada um destes modelos tem melhor desempenho que o modelo booleano clássico [OMK91, SFW83].
O Latent Semantic Index (LSI) [GSD$^+$88] é um modelo baseado no modelo do espaço vectorial, que se afirma com maior desempenho. Da colecção completa dos documentos, forma-se uma matriz termo versus documento, com cada elemento a representar o número de ocorrências de cada termo no documento. A matriz é decomposta em valores próprios, sendo os mais pequenos eliminados. O vector próprio da interrogação e as matrizes de vectores próprios são então usados para fazer corresponder aos vectores de ocorrências dos termos num sub-espaço, no qual as relações semânticas de documentos versus termos se mantêm, ao mesmo tempo que se eliminam as variações da utilização dos termos. Os documentos podem então ser ordenados por similaridade decrescente em relação à interrogação, utilizando a

medida do coseno obtida com o produto interno normalizado destes vectores.

Um outro desenvolvimento do Modelo do Espaço Vectorial é o modelo do espaço vectorial generalizado [WZRW85, WZRW87], em que os vectores de termos de índice são considerados linearmente independentes mas não ortogonais dois a dois. Esta interpretação é menos restritiva que a convencionalmente adoptada, em que os vectores de termos de índices para além de lineramente independentes são ortogonais. A pontuação resultante da aplicação deste modelo, combina a pontuação tradicional dos termos nos documentos com factores de correlação entre os termos. Contudo, para além da sobrecarga de processamento introduzida, não são muito claras as situações em que este modelo tem maior desempenho que o modelo do espaço vectorial clássico.

Um modelo importante e com larga utilização, baseado no desenvolvimento no modelo probabilístico clássico, é o das redes de inferência [TC91, TC90]. As redes de inferência de Bayes são modelos probabilísticos de raciocínio por inferência usados há já alguns anos. É um grafo acíclico directo cujos nós representam proposições (variáveis associadas a) e os arcos dependências. O valor do nó folha é uma função dos valores das variáveis de que o nó depende. Os nós folha tipicamente representam proposições cujos valores podem ser determinados por observação. Os outros nós representam proposições cujos valores devem ser determinados por inferência. A funcionalidade a realçar das redes de Bayes é o facto das dependências não precisarem de ser absolutas. As probabilidades podem ser representadas por pesos nos arcos.

O Inquery[CCH92] é um sistema de recuperação de informação, baseado num tipo de rede de Bayes chamada rede de inferência para pesquisa de documentos que inclui duas redes componentes: uma para os documentos e outra para as interrogações. Os nós na rede de inferência são verdadeiros ou falsos. Os valores atribuídos aos arcos variam entre 0 e 1 e são interpretados como crenças. A rede da interrogação é ligada à rede de documentos por arcos entre os nós dos conceitos e os nós de representação dos conteúdos. A pontuação de um documento (d) em face de uma interrogação (i) é interpretada como uma relação de unificação de conceitos e reflecte o grau de cobertura fornecida pelo conceito i ao conceito d.

Os modelos de recuperação de informação baseados nas redes de Bayes, constituem variantes modernas do raciocínio probabilístico cuja maior adequabilidade (para a recuperação de informação) deriva da possibilidade de combinação simples e elegante de fontes de evidência

distintas para suportar o julgamento de relevância (uma pontuação numérica) para um dado documento. Para além de permitirem a incorporação da informação usada nos modelos booleanos e do espaço vectorial, podem ser estendidas com naturalidade para incoporporação de informação obtida na realimentação de relevância (ver definição na secção 2.3) e históricos de sessões do utilizador.
Os custos computacionais dos sistemas baseados em redes de inferência são similares aos dos sistemas baseados no modelo do espaço vectorial.

Por vezes, estamos interessados em encontrar documentos em que determinados tópicos aparecem em componentes particulares da estrutura do documento. Modelos de recuperação de informação em que se combina a especificação de conteúdo com a estrutura do documento são designados por modelos de recuperação estruturada de informação [BYN96]. Exemplos deste tipo de modelos são o das *listas não sobrepostas* [Bur92b] e o dos *nós próximos* [NBY97].
A proposta no primeiro modelo é dividir o texto dos documentos em regiões não sobrepostas de texto que são suportadas por listas [Bur92b, CCB94]. Uma vez que há múltiplas formas de dividir um texto em regiões não sobrepostas são geradas listas múltiplas. Podemos ter por exemplo a lista de todos os capítulos de todas as secções ou todas de um livro. Essas listas são mantidas em estruturas de dados distintas. Embora as regiões duma mesma lista não se sobreponham, as regiões de texto de listas diferentes podem sobrepor-se. Para permitir a recuperação de termos em determinadas regiões de texto, basta tratar os vários componentes estruturais como termos de índice normais e registar informação de posição no texto de todos os termos de índice. Utilizando esta informação como base, é proposta uma álgebra simples [Bur92a], que permite a recuperação de informação estruturada.
O segundo modelo permite a definição de estruturas hierárquicas independentes de índices para o mesmo documento textual [NBY95, NBY97]. Cada uma dessas estruturas hierárquicas é composta por capítulos, secções, parágrafos, frases e linhas que são designadas por nós. Cada um desses nós tem uma região de texto associada. Para além disso, duas hierarquias diferentes podem referenciar regiões sobrepostas de texto. No ficheiro invertido, cada termo inclui a posição do texto em que ocorre e na hieraquia cada nó indica a posição no texto do seu componente estrutural. A linguagem de interrogação permite a especificação de expressões regulares para pesquisa de palavras, a referência a componentes estruturais e ambos. Este modelo suporta interrogações muito mais complexas que as que podem ser formuladas no modelo anterior. Contudo, para acelerar o processamento, só os nós próximos

Modelos clássicos	Booleano	Espaço Vectorial	Probablístico
Derivações dos clássicos	Fuzzy	Vector Generalizado	Redes de Inferência
	Booleano estendido	Latent Semantic Index	Redes de Crenças
		Redes Neuronais	
Estruturados	Listas não sobrepostas	Nós próximos	

Tabela 5: Taxonomia dos modelos usados em RI

(pertencentes à mesma hierarquia) são usados para restringir o conjunto de documentos do resultado.

Foram entretanto propostos uma série de novos modelos para recuperação estruturada da informação sendo os mais recentes, como não podia deixar de ser, baseados no XML [FGK00].

O facto do modelo vectorial considerar as interrogações também como documentos, transforma o processamento da interrogação num problema de comparação de documentos. Isto conduz ao estudo da proximidade e similaridade entre documentos, bem como à introdução de técnicas de aglomeração de documentos.

A ideia básica da aglomeração é que documentos similares devem ser agrupados para formar aglomerados. A importância dada à aglomeração resulta do estabelecido pela hipótese de aglomeração: *Documentos fortemente associados tendem a ser importantes para as mesmas interrogações* [vR95].

Tomando como ponto de partida a hipótese de aglomeração, pode-se pensar na construção de sistemas de recuperação de informação, onde em vez de se confrontar a interrogação com cada um dos documentos se compara com um representante de cada aglomerado de documentos. Encontrado o aglomerado mais importante, passa-se a fazer uma comparação da interrogação com cada documento desse aglomerado.

A aglomeração atraiu muita atenção na RI e na bibliotecnia assim como no reconhecimento de padrões. Embora o ênfase do reconhecimento dos padrões não seja dado à aglomeração de documentos, os métodos e conceitos usados são bastante similares. Actualmente, a aglomeração está ser objecto de trabalho para classificação automática ou semi-automática de documentos.

A aglomeração de documentos é normalmente de dois tipos: global e local [XC96].

Na estratégia de aglomeração global [Voo86, QF93], os documentos são agrupados de acordo com as suas ocorrências na colecção como um todo. Na estratégia local [AF77, XC96], os documentos são agrupados de acordo com o contexto definido pela interrogação e o conjunto de documentos passíveis de serem relevantes para essa interrogação.

A aglomeração pode ser aplicada aos termos em vez de aos documentos. Os termos co-ocorrentes são normalmente relevantes uns para os outros e por vezes sinónimos. O agrupamento dos termos é util na construção automática de thesaurus [JY92] e na redução da dimensão do dicionário de termos. A construção automática de thesaurus é baseada em critérios estatísticos e é conceptualmente idêntica aos métodos de algomeração de documentos. O thesaurus de termos pode ser usado na realimentação de relevância e na expansão de interrogações.

Para sintetizar, apresenta-se na Figura 5 uma taxonomia para os modelos de recuperação de informação retirada de [BYN99]. Alguns dos modelos apresentados na Tabela não foram aqui descritos mas podem ser estudados em [BYN99]. Os modelos derivados dos modelos clássicos estão colocados na coluna do modelo clássico correspondente.

2.3 Explicitação iteractiva de necessidades de informação

Face a um sistema automático de recuperação de informação, um utilizador pode não ser capaz de explicitar a sua necessidade de informação numa única interacção. Ele pode preferir um processo iterativo em que vá refinando a sua interrogação com base na informação que lhe é devolvida pelo sistema. O tipo de informação usada na reformulação da interrogação pode ser a mais diversa. Se um termo é muito frequente na base de dados de texto, não é uma boa escolha para se incluir numa interrogação. São necessários termos mais específicos. Se uma interrogação devolve muitos documentos, necessita de ser refinada para diminuir o número de documentos potencialmente irrelevantes. O sistema pode sugerir também sinónimos dos termos usados na interrogação para sua posterior inclusão. O processo que permite modificar a interrogação, com base nos resultados devolvidos, pode ser visto como um processo de realimentação.

A realimentação é utilizada para descrever um mecanismo que permite ao sistema melhorar a sua competência a realizar as suas tarefas, tomando em consideração a sua competência no passado. Trata-se de um conceito bem estabelecido dos sistemas de controlo, que tem sido usado com muito sucesso nos sistemas de recuperação de informação.

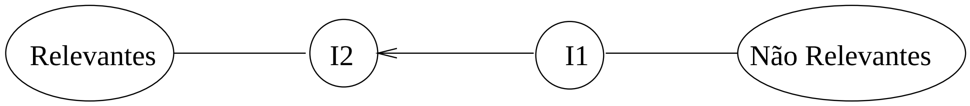


Figura 4: Processo de modificação da interrogação

Quer em sistemas de filtragem de informação quer em engenhos de pesquisa existem mecanismos de *realimentação de relevância* [Har92b], que permitem ao utlizador ir melhorando o seu perfil ou interrogação, com base nas respostas do sistema, até chegar a uma especificação que traduza mais fielmente a sua necessidade de informação. No processo de realimentação de relevância, a importância dos termos pode ser alterada, conforme ocorram em documentos considerados relevantes ou não relevantes pelo utilizador.

Existe um mecanismo que é uma simplificação da realimentação de relevância, sem qualquer intervenção do utilizador. Este mecanismo realiza duas iteracções no processamento duma interrogação: a primeira, usando a interrogação original do utilizador, devolve os documentos mais pontuados; a segunda que expande a interrogação original com os termos mais importantes dos documentos devolvidos e volta a processá-la. Só os documentos incluídos na resposta à segunda interrogação são devolvidos ao utilizador. Este mecanismo, que é totalmente transparente ao utilizador, é designado por *expansão de interrogação*.

O processo de modificação da interrogação usado nestes dois mecanismos está ilustrado na Figura 4. Com a modificação da interrogação, pretende-se torná-la mais similar, mais próxima dos documentos relevantes afastando-a dos documentos irrelevantes. A experiência mostra que tanto a realimentação de relevância como a expansão da interrogação permitem melhorar a eficácia da resposta.

2.4 Avaliação

Foi realizada muita investigação e feito muito esforço para resolver o problema da avaliação de sistemas de RI [Sal91]. Contudo, é provavelmente correcto dizer que a maioria dos investigadores desta área ainda sente que o problema está longe de estar resolvido. No que resta da secção, introduzem-se as métricas mais comuns e a metodologia utilizada na avaliação deste tipo de sistemas.

O objectivo da avaliação é saber se o sistema funciona com eficácia e eficiência. Ser eficaz

significa devolver respostas correctas e ser eficiente significa responder com rapidez, em face dos recursos computacionais utilizados.
Em 1966, Cleverdon [CMK66] apresentou seis quantidades mensuráveis para avaliar a competência dum sistema de RI:

1. A *importância da colecção*, isto é, a extensão em que o sistema inclui material relevante e de interesse para a população de utilizadores;

2. A forma de *apresentação* dos resultados [Mar92];

3. O *esforço* envolvido por parte do utilizador para obter respostas aos seus pedidos;

4. O *tempo de atraso*, isto é o intervalo médio entre o instante em que é feito o pedido e devolvida a resposta;

5. A *cobertura* do sistema, isto é, a proporção do material devolvido relativamente ao relevante existente; e,

6. A *precisão* do sistema, isto é, a proporção do material devolvido que é relevantei.

Enquanto as três primeiras medidas se preocupam com a utilidade e a facilidade de uso, a quarta preocupa-se com a eficiência do sistema e a as duas últimas com a eficácia ou correcto funcionamento do sistema. Admite-se que quanto mais eficaz for o sistema maior será a satisfação do utilizador.
A eficácia do sistema é directamente proporcional à qualidade da resposta e inversamente proporcional ao esforço do utilizador para obter essa resposta. Se se assumir que o esforco do utilizador é constante, a eficácia será proporcional à qualidade da resposta. A qualidade da resposta ou eficácia dum sistema de RI pode ser medida com base na relevância dos documentos incluídos na resposta.
Houve muita controvérsia no passado a respeito da precisão e cobertura serem as medidas mais adequadas para medir a eficácia. Uma alternativa seria a cobertura e desacerto[8] (proporção de documentos não relevantes devolvidos). Contudo, todas as alternativas precisam de alguma forma da determinação de relevância.
A utilização do par precisão e cobertura é intuitiva. A ideia é obter o máximo de documentos possível (máxima cobertura) mas ao tempo minimizar o número de documentos

[8] fall-out, na terminologia inglesa

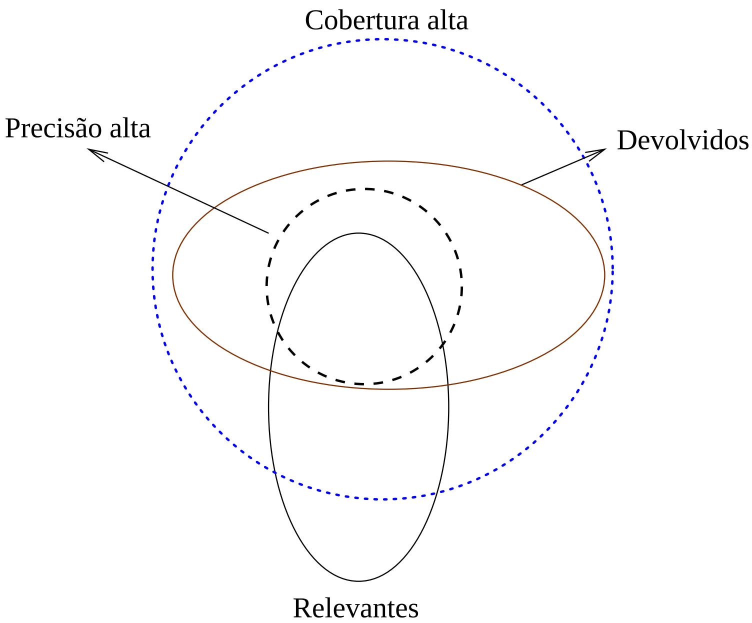


Figura 5: Maximização da precisão e cobertura

	relevante	não relevante	
devolvidos	$A \cap B$	$\overline{A} \cap B$	B
não devolvidos	$\overline{A} \cap B$	$\overline{A} \cap \overline{B}$	$\overline{B}$
	A	$\overline{A}$	

Tabela 6: Tabela de contigência para medidas de eficácia

irrelevantes (máxima precisão). A maximização duma das medidas corresponde normalmente à minimização da outra, como se pode observar na Figura 5. É preciso encontrar um equilíbrio [BG94]. A importância relativa dada a cada uma pelo utilizador, que pode variar com a situação concreta, pode ser um aspecto a considerar.

As várias medidas de eficácia podem ser facilmente obtidas da Tabela 6, onde A é o conjunto de documentos relevantes e B o conjunto de documentos devolvidos.

$$\begin{aligned} \text{precisão (P)} &= \frac{|A \cap B|}{|B|} \\ \text{cobertura (C)} &= \frac{|A \cap B|}{|A|} \\ \text{desacerto (D)} &= \frac{|\overline{A} \cap B|}{|\overline{A}|} \end{aligned}$$

O facto das várias medidas de eficácia se basearem na relevância dificulta a avaliação deste tipo de sistemas.

A relevância é uma noção subjectiva. Dois utilizadores podem diferir na atribuição de relevância ou não a documentos particulares. O mesmo utilizador pode diferir no seu julgamento em contextos diferentes. Apesar disso, a diferença não é suficientemente grande que invalide a existência de colecções de documentos, dispondo de um conjunto de interrogações de teste para as quais existam julgamentos de relevância. Os julgamentos de relevância são atribuídos por um painel de peritos no assunto, com ou sem a ajuda de um conjunto de sistemas de RI considerados de boa qualidade. O conjunto de colecções de teste mais usadas são disponibilizadas pela iniciativa TREC [Har93, Jon95].

Para cada interrogação submetida ao sistema, pode ser construída uma tabela similar à Tabela 6. Com base nessas tabelas, são calculados os valores de precisão e cobertura. Se o resultado da interrogação depende de um parâmetro como o número de documentos devolvidos, ele pode ser modificado para a obtenção de uma tabela diferente e, portanto, um par diferente de precisão e cobertura. Se λ for parâmetro então $P(\lambda)$ designa precisão, $C(\lambda)$ a cobertura e a precisão-cobertura é designada pelo par ordenado $(P(\lambda), C(\lambda))$. O conjunto de pares ordenados define o gráfico precisão-cobertura. Para medir o desempenho geral combinam-se as curvas de alguma forma para obtenção duma curva média. Esta abordagem é designada por *micro-avaliação*.

Um abordagem alternativa para determinar médias é a *macro-avaliação*, que pode ser calculada independentemente de qualquer parâmetro tal como o número de documentos devolvidos. A curva média é obtida especificando um conjunto de valores de cobertura normalizados para os quais se calculam os valores de precisão médios. Os valores globais são obtidos calculando a média das interrogações individuais. Muitas vezes não correspondem a um único valor, pelo que é necessário interpolar.

2.5 Recuperação distribuída da informação

A distribuição das Bases de Dados de Texto lança novos desafios que se traduzem na adaptação a um ambiente distribuído, dos mecanismos e facilidades disponíveis num sistema de pesquisa centralizado.

Entretanto, a necessidade de tornar *transparente* ao utilizador a distribuição das BDTs, introduz fundamentalmente duas novas funções:

1. Seleccionar o subconjunto de BTDs relevantes para uma dada interrogação — o problema da *selecção* das Bases de Dados;

2. Distribuir as interrogações pelas fontes seleccionadas e esperar pela devolução das respostas; e,

3. Juntar as respostas devolvidas pelas BDTs seleccionadas numa única, a devolver ao utilizador— o problema da *fusão de respostas.*

Tanto a fusão como a selecção são realizadas num novo componente, o Encaminhador de Interrogações, que se confronta com uma série de problemas adicionais:

1. A possibilidade da sobreposição entre as diferentes BDTs — o problema da *sobreposição*;

2. A necessidade de correspondência entre as interrogações de BDTs diferentes — o problema da *correspondência de interrogações* [CCGMP96]; e,

3. A necessidade de correspondência entre a meta-informação das respectivas respostas — o problema da *correspondência da meta-informação* dos documentos.

Com múltiplas fontes de informação disponíveis, para satisfazer uma necessidade de informação do utilizador, pode-se usar uma de duas abordagens. A primeira é interrogar todas as fontes de informação, partindo do princípio que todas contêm documentos relevantes para a sua satisfação. A segunda é seleccionar apenas o subconjunto das fontes consideradas relevantes para a necessidade de informação.

A segunda abordagem é a única que é viável se houver um grande número de fontes disponíveis, tornando os custos de comunicação e processamento decorrentes da difusão incomportáveis. O problema que se coloca é que critério usar para a selecção das fontes.

A abordagem utilizada é arranjar um representante para cada fonte da mesma maneira que se representam os documentos num engenho de pesquisa convencional e pontuar as fontes em função duma fórmula de similaridade com a interrogação. As fontes são ordenadas pela pontuação, sendo interrogadas apenas as k fontes mais pontuadas. Cada colecção é representada na maior parte dos casos por um resumo com as estatísticas associadas a cada termo do dicionário (nomeadamente o número de ocorrências e o número de documentos da colecção em que ocorre). Para além disso, está disponível o total de documentos de cada colecção.

Teremos cada colecção representada no ficheiro invertido do encaminhador de interrogações como um *mega-documento* e estatísticas correspondentes. A ideia é usar as fórmulas convencionais de pontuação de documentos (nomeadamente a fórmula do coseno) para pontuar o mega-documento e atribuir essa pontuação à colecção.
Coloca-se aqui um problema de granularidade. A pontuação elevada dum mega-documento pode estar diluída em muitos documentos da colecção e haver um mega-documento mal pontuado que represente uns poucos documentos com pontuação elevada. Uma solução para este problema é a o uso de blocos (B) com um certo número de documentos. Se o tamanho de B for 1 temos um engenho de pesquisa convencional. Se B tiver o tamanho da colecção teremos a situação anterior. A ideia é encontrar um valor de B adequado para resolver o problema da escala (tamanho da BD do EI) e ao mesmo tempo minimizar estas situações com pontuações enganadoras [MZ95].
Embora possa ser baseada nos resultados da RI convencional, o problema da selecção de BDs tem as suas especificidades e existe já trabalho realizado e algumas propostas para a sua solução [Fuh96, Pow01, CPV98, CPC+99, Bau97, Bau99, GGMT94, GGM95].
Uma vez recebidas as respostas das várias fontes interrogadas, coloca-se o problema da fusão das respostas numa única, a devolver ao utilizador. Se se ignorar a pontuação atribuída a cada item de resposta, trata-se simplesmente de juntar os resultados parciais. Em vez disso pretende-se, com base em várias listas ordenadas por pontuação, obter uma lista única também ordenada por pontuação. As pontuações atribuídas pelas várias fontes podem não ser comparáveis pelo facto de se usarem fórmulas diferentes e também porque as estatísticas de cada sub-colecção (que são usadas nas diferentes fórmulas) são diferentes. Existem várias abordagens para solucionar este problema que vão desde a repontuação dos documentos com base nas estatísticas associadas (quando são devolvidas), repontuação com base nas pontuações parciais, na própria informação de selecção ou outras estatísticas disponíveis no EI.

2.5.1 Trabalho Relacionado

O trabalho mais referenciado é o apresentado em GLOSS[GGMT94] e gGLOSS [GGM95]. Esta proposta resolve o problema da ordenação das fontes de acordo com a relevância para uma dada interrogação. Não calcula o número de documentos a pedir a cada fonte. Foi baseado inicialmente no modelo booleano [GGMT94] para posteriormente ter subjacente o modelo vectorial e usar uma estrutura hierárquica para resolver os problemas de escala

[GGM95]. A simplificação subjacente aos sumários disponíveis é que a ocorrência das palavras se distribui uniformente em todos documentos em que ocorre. Existem duas fórmulas diferentes para ordenação de fontes, pressupondo dois cenários de correlação dos diferentes termos. No cenário de alta correlação, parte-se do príncipio que os termos de maior peso ocorrem nos documentos conjuntamente com todos os outros termos da interrogação. No cenário de dijunção, os termos da interrogação ocorrem em documentos diferentes. A pontuação de cada fonte é calculada como sendo a soma de todos os documentos cuja similaridade com a interrogação ultrapassa um determinado valor de corte. Além disso, o trabalho do gGLOSS [GGM95] suporta uma hierarquia de índices em que a informação mantida nos níveis superiores é a identificação da fonte que contêm determinada palavra, o respectivo número de documentos e a soma do número de ocorrências da palavra no conjunto das fontes. Neste âmbito, a disseminação das estatísticas é global e também se utiliza o mesmo dicionário de termos e modelo de pontuação de documentos. A avaliação deste sistema foi feita usando mensagens das News e obteve-se uma eficácia bastante boa em face do elevado nível de compactação conseguido com os sumários de índices (2% da informação original). Todavia, o número de fontes usado é muito limitado e não se usaram interrogações com julgamentos de relevância.

O objectivo da abordagem apresentada em [LCC96, CLC95] é o uso de redes de inferência, que têm subjacente o modelo probabilístico para resolver a pesquisa em colecções múltiplas (selecção de BDTs e fusão de resultados). A implementação de suporte usada é o INQUERY [CCH92]. Usa uma estrutura hierárquica em que a informação mantida nos níveis superiores é o número de BDTs que contêm uma dada palavra e a soma do número de documentos em cada colecção. Mantém um vocabulário de índices global, a estatística da colecção é disseminada e todas as BDTs usam o mesmo esquema de representação de documentos e de pontuação. A informação mantida no EI relativa a cada BD é para cada termo o número de documentos que contêm esse termo e a soma dos respectivos pesos. No modelo suportado por redes de inferência para colecções, o texto associado (documentos, passagens e palavras) são representados como nós na rede da colecção. A rede da interrogação inclui um nó para a necessidade de informação, nós para interrogações e nós para os conceitos. Com esta abordagem, há um único modelo para pontuação dos documentos e da colecções. Os resultados obtidos por este sistema mostram que não há grande diferença de eficácia entre a situação centralizada e a distribuída, quando se usam em média metade da colecção distribuída, isto é 50% das fontes. Contudo, este resultado pode ter sido prejudicado pelo facto de usar um pequeno número de fontes (17).

No trabalho apresentado em [Voo95] pretende-se resolver o problema da pesquisa de colecções múltiplas usando o modelo de espaço vectorial e usando o SMART [Buc85] como BDT. Nesta proposta, utiliza-se uma estrutura plana e a informação usada para seleccionar é informação obtida com interrogações passadas. Portanto, a informação usada para selecção é obtida usando um conjunto de interrogações de treino feitas a cada uma das BDTs. São usadas duas técnicas para ordenar as BDTs e calcular os documentos a devolver. A primeira baseia-se na aglomeração das interrogações e a segunda na modelação da distribuição dos documentos relevantes. Os resultados, obtidos usando a colecção TREC, mostram que se consegue no caso distribuído resultados 10% inferiores à situaçãocentralizada, embora se use apenas 5 sub-colecções. O problema desta abordagem é a necessidade de julgamentos de relevância para as interrogações de treino, o que é muito difícil de conseguir em ambientes operacionais.

O trabalho apresentado em [Bau97] tem como objectivo contribuir para um enquadramento teórico fundamentado para a pesquisa distribuída de BDTs. Usa o modelo probabilístico como suporte e encara-se a possibilidade de usar o sistema SPIDER [Sch93] para concretizá-lo. Baseia-se numa estrutura hierárquica e apenas um subconjunto das características de índice do documento são exportadas para a indexação da colecção no nível superior. Todavia, não é apresentado qualquer método para construção de índices [9]. Reclama a inexistência de um dicionário de termos global mas apenas usa dicionários locais a cada sub-colecção. Os parâmetros que esta abordagem propõe que se utilize de cada base de dados é a distribuição da função de RSV[10]. O cálculo do número de documentos de cada BDT é efectuado com base pela área sobre a curva de RSV estimada. Fazem-se apenas algumas considerações a respeito da sobreposição e existe a possibilidade de expansão de interrogações e realimentação de relevância local e global. Os resultados experimentais, que são apresentados em trabalho subsequente [Bau99], com 81 BDs usando a colecção de teste TREC, aproximam-se bastante da situação centralizada. A diferenças de precisão média são da ordem das centésimas.

A abordagem apresentada em [Fuh96], tem subjacente uma estratégia baseada em custos para selecção óptima das BDTs. Tem como suporte o modelo probabilístico de pesquisa, mas reclama poder ser usado como um enquadramento genérico de todas as abordagens que

[9] indicado para trabalho futuro
[10] Retrieval Status Value

lidem com o problema da seleçao de BDTs. Esta abordagem não faz qualquer pressuposto a respeito dos métodos usados na construção de índices embora possa ser usado em ambiente hierárquico. Também não indica qual é a fórmula usada para pontuação os documentos. Os parâmetros usados de cada BDT são as curvas esperadas de precisão e totalidade, o número esperado de documentos relevantes e o custo para processamento de interrogações e acesso aos documentos. Como é usual, o custo aqui pode significar dinheiro, tempo de computação, tempo de resposta ou o tempo despendido pelo utilizador. As fórmulas para ordenação das BDTs e determinação do respectivo número de documentos são baseados na minimização das funções de custo (usando multiplicadores de Lagrange) relativamente ao vector do número de documentos devolvidos por cada BDT. Existe uma concretização baseada neste modelo no âmbito do Projecto Medoc [DGH+97].

O trabalho reportado em [MZ95], já referenciado, usa um índice centralizado de blocos de documentos. Assim, pesquisa-se inicialmente o índice de blocos e posteriormente cada bloco. A eficácia depende bastante do tamanho do bloco e há uma degradação significativa quando o bloco tem um tamanho superior a 1000 documentos .

Finalmente, o trabalho apresentado em [XC98], utiliza a expansão de interrogações e a indexação de frases para melhorar a eficácia da pesquisa distribuída. A expansão é feita usando uma colecção de treino, concebida para o efeito. Os resultados apresentados mostram uma melhoria na eficácia das respostas devido a estes dois mecanismos.

Capítulo 3

Arquitectura do Webtrieve

Sumário

Neste capítulo, é relatado trabalho que tem como objectivo a concepção e concretização do Webtrieve, uma arquitectura distribuída para Bases de Dados de Texto. A principal utilização do Webtrieve é servir de bancada de teste a novos modelos e algoritmos em sistemas distribuídos de recuperação de informação. Apresentam-se as funcionalidades esperadas num sistema de RI distribuído. Introduzem-se os principais componentes e a sua interacção usando protocolos normalizados como o HTTP e o STARTS. Posteriormente, são apresentadas com algum detalhe as funcionalidades dos diversos componentes. Descreve-se o processo que vai desde a colecta dos documentos até a disponibilidade para interrogação do motor de busca distribuído. No final do capítulo abordam-se aspectos relacionados com avaliação em Laboratório do Webtrieve.

3.1 Descrição geral da arquitectura

O Webtrieve é um sistema distribuído de Bases de Dados de Texto interligadas em rede, cuja arquitectura geral é apresentada na Figura 6. O modelo subjacente ao Wetrieve é o Modelo do Espaço Vectorial [SL68, SY73, Buc85].

Na Figura 6 e nas restantes ao longo deste capítulo, os valores entre parêntisis (X) indicam uma sequenciação temporal entre os vários fluxos de dados. A notação usada nessas figuras é baseada na usada nos Diagramas de Fluxos de Dados (DFDs).

O espaço de informação indexado pelo Webtrieve é constituída por um conjunto de servidor de documentos (SD). Na Figura 6, apresenta-se o espaço de informação (tipicamente o

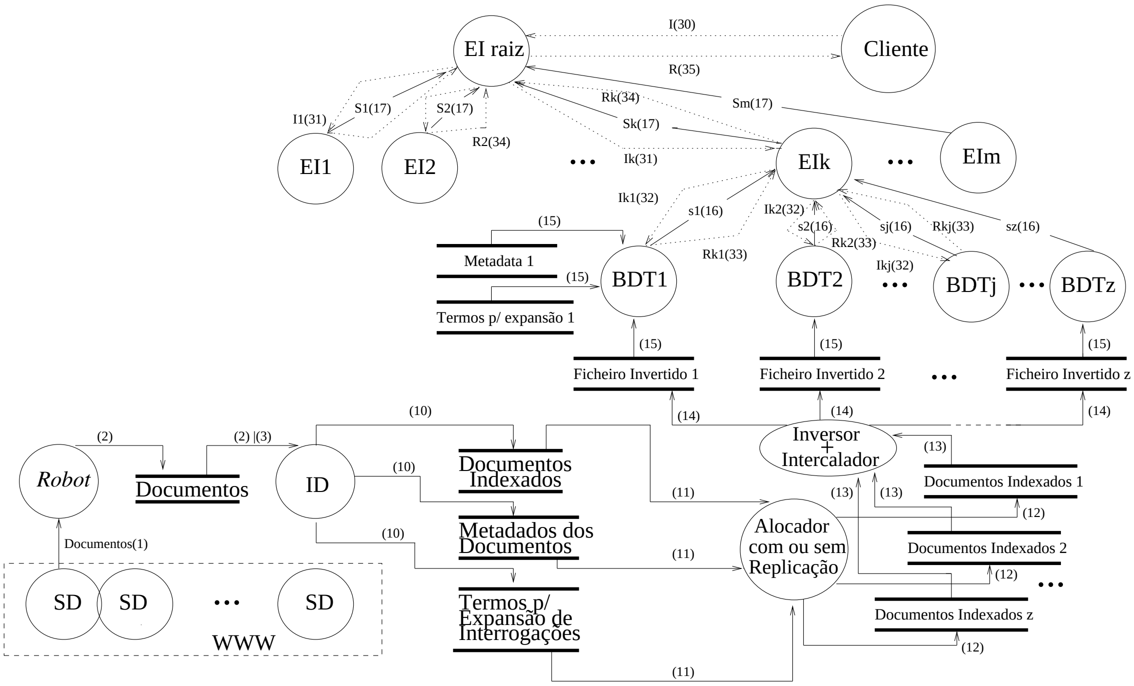


Figura 6: Arquitectura Geral do Webtrieve

WWW) e os componentes principais do Webtrieve. Os componentes do Webtrieve são o Encaminhador de Interrogações (EI), a Base de Dados de Texto (BDT), o Alocador, o Robot e um conjunto de programas auxiliares para indexação. O *Robot* recolhe documentos de um conjunto de servidores de documentos (SDs) seleccionados colocando-os em disco. Esses documentos são posteriormente processados pelo *Indexador de Documentos* (ID).
Na Figura 7 estão apresentados os documentos no formato Wall Street Journal da TREC usados para alguns testes preliminares ao protótipo do Webtrieve. Os documentos foram gerados manualmente para permitir uma verificação simples do funcionamento do protótipo.

O ID processa os documentos produzindo três tipos de informação de saída: documentos indexados, metadados dos documentos [WM97] e termos para expansão.
O facto de se separar os termos a usar para expansão, foi uma opção simples mas que acarreta necessidade de maior espaço em disco. Esta opção justifica-se pelo facto de, para já, não se pretender tratar do problema no âmbito desta tese. É um tópico importante para trabalho futuro quer na sua concretização em cada motor de busca quer na RI distribuída.
Um documento indexado corresponde a uma tabela com os termos e respectiva informação de occorrência. O registo correspondente a cada termo mantém o número de ocorrências, posição de ordem das ocorrências e até, caso se aplique, em que componente[1] estrutural do documento ocorreu. O componente pode ser representado por uma etiqueta. Os componentes de um documento indexado são apresentados a seguir: o identificador interno do documento (docid), um conjunto de uma ou mais entradas correspondentes aos termos que ocorrem nesse documento (tp). Associado a cada termo, tem-se a respectiva palavra (term), o número de ocorrências (freq) e a informação associada a cada ocorrência (le). Para cada ocorrência temos a identificaçãcao da posição no texto em que ocorre e uma etiqueta que identifica o componente do texto em que ocorre. Essa etiqueta pode ser obtida da árvore de reconhecimento sintático do documento. Apresenta-se na definição 1 a gramática BNF do ficheiro invertido.

$$\begin{array}{llllll} di & ::= & docid & (tp)+ & \\ tp & ::= & term & freq & (le)+ \\ le & ::= & position & tag & \end{array} \tag{1}$$

Um exemplo de documento invertido está apresentado na Figura 8. Desligou-se a informação

[1]Exemplos de componentes são o corpo, título, autor, etc...

```
<DOC> <DOCNO> WSJ870324-0001 </DOCNO>
     <DD> 03/23/87</DD>
     <TEXT> Industrial production in Italy declined declined 3.4% in January
             from a year earlier, the government said.  </TEXT> </DOC>
<DOC> <DOCNO> WSJ870323-0181 </DOCNO>
<DD> 03/23/87</DD>
<TEXT> Industrial Industrial production production production production in </TEXT>
</DOC>
<DOC> <DOCNO> WSJ870323-0180 </DOCNO>
<DD> 03/23/87</DD>
<TEXT> Italy Italy Italy declined declined 3.4% in January </TEXT> </DOC>
<DOC> <DOCNO> WSJ870323-0179 </DOCNO>
<DD> 03/23/87</DD>
<TEXT> from a year year earlier earlier earlier earlier, </TEXT> </DOC>
<DOC> <DOCNO> WSJ870323-0178 </DOCNO>
<DD> 03/23/87</DD>
<TEXT> the government government government  government said. said.  </TEXT>
</DOC>
<DOC> <DOCNO> WSJ870323-0177 </DOCNO>
<DD> 03/23/87</DD>
<TEXT> Industrial Industrial Industrial production production in
   Italy Italy declined declined 3.4% in January </TEXT> </DOC>
<DOC> <DOCNO> WSJ870323-0176 </DOCNO>
<DD> 03/23/87</DD>
<TEXT> Industrial Industrial Industrial Industrial production production
      production in from a year year earlier earlier earlier, </TEXT> </DOC>
 <DOC> <DOCNO> WSJ870323-0175 </DOCNO>
<DD> 03/23/87</DD>
<TEXT> Industrial Industrial production in the government government
       government said. said.  </TEXT> </DOC>
<DOC> <DOCNO> WSJ870323-0172 </DOCNO>
<DD> 03/23/87</DD>
<TEXT> from a year year earlier, the government said. said.  </TEXT> </DOC>
<DOC> <DOCNO> WSJ870323-0171 </DOCNO>
<DD> 03/23/87</DD>
<TEXT> Industrial production production in Italy Italy Italy Italy Italy
  declined declined 3.4% in January from a year year earlier, </TEXT> </DOC>
```

Figura 7: Documentos usados na colecção de teste

```
<DOC docid=0 pos=false stem=true stop=true>
<METADOC TYPE=webt DOCID=0>
<FIELD Name=document-size Value=201>
<FIELD Name=docnumber Value=WSJ870324-0001>
<FIELD Name=document-count Value=8>
<FIELD Name=document-md5 Value=d41d8cd98f00b204e9800998ecf8427e>
<FIELD Name=story-date> 03/23/87 </FIELD>
<FIELD Name=linkage Value=/usr1/webt/webtrieve/adm/input/wsj_eval:0>
</METADOC>
<TP term=januari number=1>
<TP term=industri number=1>
<TP term=product number=1>
<TP term=govern number=1>
<TP term=earlier number=1>
<TP term=declin number=2>
<TP term=itali number=1>
</DOC>
```

Figura 8: Exemplo de documento invertido

posicional e da estrutura uma vez que não foram usados ao longo deste trabalho.

Colocou-se informação a respeito da estrutura dos documentos (a etiqueta) apenas com objectivo de manter a compatibilidade com o STARTS, o protocolo usado na comunicação entre os diversos componentes. Trata-se de uma solução provisória, que precisa de ser reequacionada tomando como ponto de partida trabalho já existente para busca estruturada de documentos [BYN96, CCB94].

Os metadados do documento correspondem a um registo de meta-informação incluindo autor, data, tamanho, URL, sumário criptográfico (MD5 [Riv92]), descrição e outra informação. Os atributos possíveis dependem do tipo de documento que está a ser tratado.

Os termos para expansão correspondem aos termos que possuem um maior número de ocorrências no documento (em número esse que é parametrizável) e que podem ser usados quer para expansão automática de interrogações quer na realimentação de relevância [Har92b].

A informação mantida para cada documento resume-se aos k termos mais pontuados. Para cada entrada, inclui-se o termo propriamente dito e o respectivo número de ocorrências, de acordo com a gramática apresentada na definição 2.

```
<PL term=declin idf=6> <PE docid=11 num=2> <PE docid=9 num=2> <PE docid=8 num=2>
<PE docid=5 num=2> <PE docid=2 num=2> <PE docid=0 num=2> </PL>
<PL term=earlier idf=6> <PE docid=3 num=4> <PE docid=6 num=3> <PE docid=8 num=2>
<PE docid=11 num=1> <PE docid=10 num=1> <PE docid=0 num=1> </PL>
<PL term=govern idf=5> <PE docid=4 num=4> <PE docid=7 num=3> <PE docid=9 num=2>
 <PE docid=10 num=1> <PE docid=0 num=1> </PL>
<PL term=industri idf=6><PE docid=6 num=4> <PE docid=5 num=3> <PE docid=7 num=2>
<PE docid=1 num=2> <PE docid=11 num=1> <PE docid=0 num=1> </PL>
<PL term=itali idf=6> <PE docid=11 num=5> <PE docid=8 num=4> <PE docid=2 num=3>
<PE docid=5 num=2> <PE docid=9 num=1> <PE docid=0 num=1> </PL>
<PL term=januari idf=6> <PE docid=11 num=1> <PE docid=9 num=1>
<PE docid=8 num=1> <PE docid=5 num=1><PE docid=2 num=1> <PE docid=0 num=1> </PL>
<PL term=product idf=6><PE docid=1 num=4> <PE docid=6 num=3> <PE docid=11 num=2>
 <PE docid=5 num=2> <PE docid=7 num=1> <PE docid=0 num=1> </PL>
```

Figura 9: Ficheiro invertido para a coleccção de teste

$$\begin{array}{lcll} exp & ::= & docid & (tf)+ \\ tf & ::= & term & freq \end{array} \tag{2}$$

A informação relevante extraída de de todos documentos é tratada por um alocador que divide os documentos por grupos, usando um conjunto de critérios pré-estabelecidos. O número de grupos depende da quantidade de Bases de Dados de Texto (BDT) disponíveis. O alocador é usado em ambiente laboratorial para dividir *artificialmente* os documentos em grupos mas pode também ser usado em ambiente operacional para permitir carregar BDTs temáticas [RMCF95]. As estratégias de alocação de documentos podem ser de natureza aleatória, baseadas na meta-informação dos documentos, perfil dos servidores ou no conteúdo dos documentos, usando algoritmos de aglomeração[2] [Exp97]. Neste último caso, o encaminhador de interrogações pode ser usado como alocador. Para esse efeito, usam-se os representantes [Exp97] das BDTs como sumários e o documento a alocar como interrogação. A BDT com maior pontuação será a escolhida para colocar o documento.

A alocação dos documentos pode ser feita com replicação para permitir estudar a influência da sobreposição de BDs neste tipo de sistemas. As estratégias de replicação podem ser aleatórias, de acordo com níveis pré-definidos de sobreposição de documentos ou mesmo de acordo com a popularidade do documento.

[2] *clustering algorithms*, na terminologia inglesa

Constituídos os grupos de documentos indexados, os documentos de cada grupo são processados por um inversor para construção do ficheiro invertido. Por limitação de recursos[3] de memória, principalmente, pode ser necessário construir ficheiros invertidos parciais que são posteriormente intercalados num único.
A estrutura do ficheiro invertido mantém as listas de ocorrências (pl^4) para cada termo t. Cada lista de ocorrências mantém um registo por cada documento em que o termo ocorre (pe^5). Por sua vez, cada pe contém uma lista de localizações le^6, incluindo cada uma a posição (pos) e uma etiqueta (tag^7) correspondente ao componente do documento em que ocorre. Apresenta-se na definição 3, a gramática em BNF do ficheiro invertido.

$$\begin{array}{lll} infvile & ::= & term \quad pl+ \\ pl & ::= & pe+ \\ pe & ::= & docid \quad le+ \\ le & ::= & pos \quad tag \end{array} \tag{3}$$

O ficheiro invertido para colecção de teste está apresentado na Figura 9. Mais uma vez não há informação posicional nem estrutural. Pode-se ver uma lista de ocorrência por linha, embora haja algumas com mais que uma....
Posteriormente, carregam-se as BDTs com os respectivos ficheiros invertidos, ficheiros de metadados e termos para expansão.
Para que as BDTs possam ser integradas numa hierarquia, precisam de exportar sumários (S_i e s_i na Figura 6) dos seus conteúdos para encaminhadores de interrogações (EIs). Estes poderão ter necessidade de fazer o mesmo, exportando também sumários de informação para o nível superior da hierarquia.
O conteúdo do sumário, dependente do algoritmo de selecção usado, corresponde a uma síntese da forma como cada termo ocorre nos documentos da colecção. No protocolo STARTS [GCGM+97] envia-se, como síntese, o número de documentos em que ocorre o termo e a soma das frequências. No Webtrieve, para possibilitar a utilização de algoritmos em que se usa uma função de distribuição de probabilidade que necessitam adicionalmente do desvio padrão (ver Capítulo 6), inclui-se, para além disso, a soma do quadrado das ocorrências.

[3] Os erros *Out of Memory* da nossa implementação Java
[4] abreviatura com base na terminologia utilizada *posting list*
[5] acrónimo resultante da da terminologia inglesa usada (*posting entry* -pe)
[6] location entry, da terminologia inglesa
[7] do termo em inglês

Uma vez construída a hierarquia de indexação, está tudo pronto para se responder a interrogações (I_i na Figura 6). As interrogações podem ser originadas por um cliente normal (caso de funcionamento operacional) ou por um cliente em lotes[8] (usado em Laboratório).

Embora as interrogações possam ser colocadas pelo cliente a qualquer EI, elas começam sempre, ainda que virtualmente, no topo da hierarquia. A seguir, são encaminhadas pelos EIs que as encaminham para um subconjunto das BTDs que supostamente contêm os documentos mais relevantes. As respostas (R_i na Figura 6) das BDTs são fundidas [BKFS95], devolvendo-se uma única resposta ao utilizador.

Uma descrição mais pormenorizada dos componentes mais importantes da arquitectura, é feita nas secções seguintes.

3.2 Clientes

Os clientes são aplicações muito simples que permitem ao utilizador interagir com o sistema de RI. Iniciam quase todas as tarefas realizadas no sistema mas realizam muito pouco processamento. Existem vários tipos de comandos que podem ser efectuados pelo utilizador, como pode ser visualizado na Figura 10.

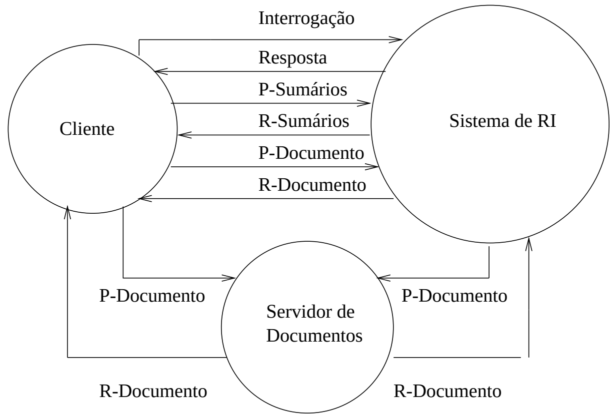


Figura 10: Tipo de interacções do cliente com o sistema de RI

[8] Tradução de cliente *batch* ne terminologia inglesa

O comando típico é uma interrogação. A interrogação é feita usando a linguagem de interrogação do sistema. É composta por um conjunto de termos e de operadores suportados pelo sistema.

Uma vez devolvida a resposta com uma listagem de documentos, o utilizador pode pretender visualizar informação resumida do conjunto de documentos (P-Sumário)) e pode também pretender visualizar totalmente algum documento (P-Documento). Em alguns casos, o servidor de RI não intervém no serviço de documentos.

O utilizador pode pretender refinar a sua interrogação indicando no conjunto de documentos devolvidos, um ou mais relevantes. Os restantes são considerados não relevantes. Adicionalmente, podem ser sugeridos ao utilizador os termos que o sistema pretende adicionar ou remover da interrogação. Este processo é designado por realimentação de relevância.

No caso de um sistema de RI distribuída, o utilizador pode especificar algumas restricções no processamento da interrogação. Uma delas é eliminar um subconjunto das fontes pesquisáveis. Tambem pode limitá-las em quantidade ou mesmo em tempo de resposta.

No caso do Webtrieve, e como se implementa a interface STARTS [GCGM$^+$97], estão disponíveis uma expressão de pontuação[9] e um filtro booleano.

Na expressão de pontuação é especificado um conjunto de termos que podem ter atribuída uma pontuação que especifica a sua importância relativa. Essa pontuação pode ser usada por cada motor na ponderação relativa dos diversos termos da interrogação. Numa lista simples ou lista com pontuação pode-se, inclusivamente, especificar em que componentes do documento (documento semi-estruturado) se espera que ocorram os termos. Os termos podem também estar ligados por operadores de proximidade. As várias expressões de pontuação podem estar ligadas por operadores booleanos.

Como exemplo de expressão de pontuação apresenta-se

```
list((body-of-text "databases") and ( "selection"))
```

Os documentos são pontuados de acordo com a ocorrência das palavras indicadas da lista.

No filtro boleano são especificadas algumas restricções que devem ser cumpridas por todos documentos da resposta. Os filtros são expressões booleanas com termos simples ou termos ligados com operadores de proximidade. Pode-se também especificar em que componentes do documento o termo pode ocorrer.

[9]ranking expression na terminologia inglesa

Como exemplo de filtro booleano pode apresentar-se

```
((author "Joaquim Macedo") and (title "overlapping"))
```

Neste caso, só são devolvidos documentos com este author e com a palavra indicada no título.
O cliente usado é baseado em formulários Web que geram um comando HTTP com a sintaxe STARTS. Como veremos adiante, há também um *cliente por lotes* para ser usado em experiências laboratoriais.

3.3 Robot

O robot, cujo digrama funcional é apresentado na Figura 11, percorre exaustivamente o espaço de informação e colecta documentos para disco local.
Esses documentos são processados posteriormente pelo Indexador de Documentos.
O robot utiliza normalmente como pontos de partida um conjunto de endereços documentos bem conhecidos desginados por *endereços semente*. Posteriormente , com base em algoritmos recursivos (usando as referências existentes em cada documento), percorre todo o espaço de informação indexável.
O percurso pelos documentos do espaço de informação, pode ser efectuado usando estratégias *primeiro em largura*[10] ou *primeiro em profundidade*[11].
Antes da transferência de qualquer documento, verifica-se se este já foi processado, a validade da referência e se cumpre os requisitos de domínio e de formato[12]. Para documentos já indexados, a transferência só é efectuada se tiver havido alguma alteração.
Fazendo periodicamente o percurso pelo espaço de informação, o robot pode obter dados históricos a respeito da chegada de novos documentos, modificações de documentos já existentes, disponibilidade e tempos de resposta dos servidores. Com esses históricos, podem-se obter curvas experimentais com o ritmo de criação de novos documentos e modificações nos existentes. Essas curvas podem ser usadas para avaliação da eficiência do sistema, como se verá mais adiante na secção referente à avaliação da arquitectura.

[10] percurso infixo
[11] percurso pós-fixo
[12] Tipicamente, apenas são processados os documentos no domínio de nomes a processar (por exemplo uminho.pt) e só os formatos donde é possível extrair texto

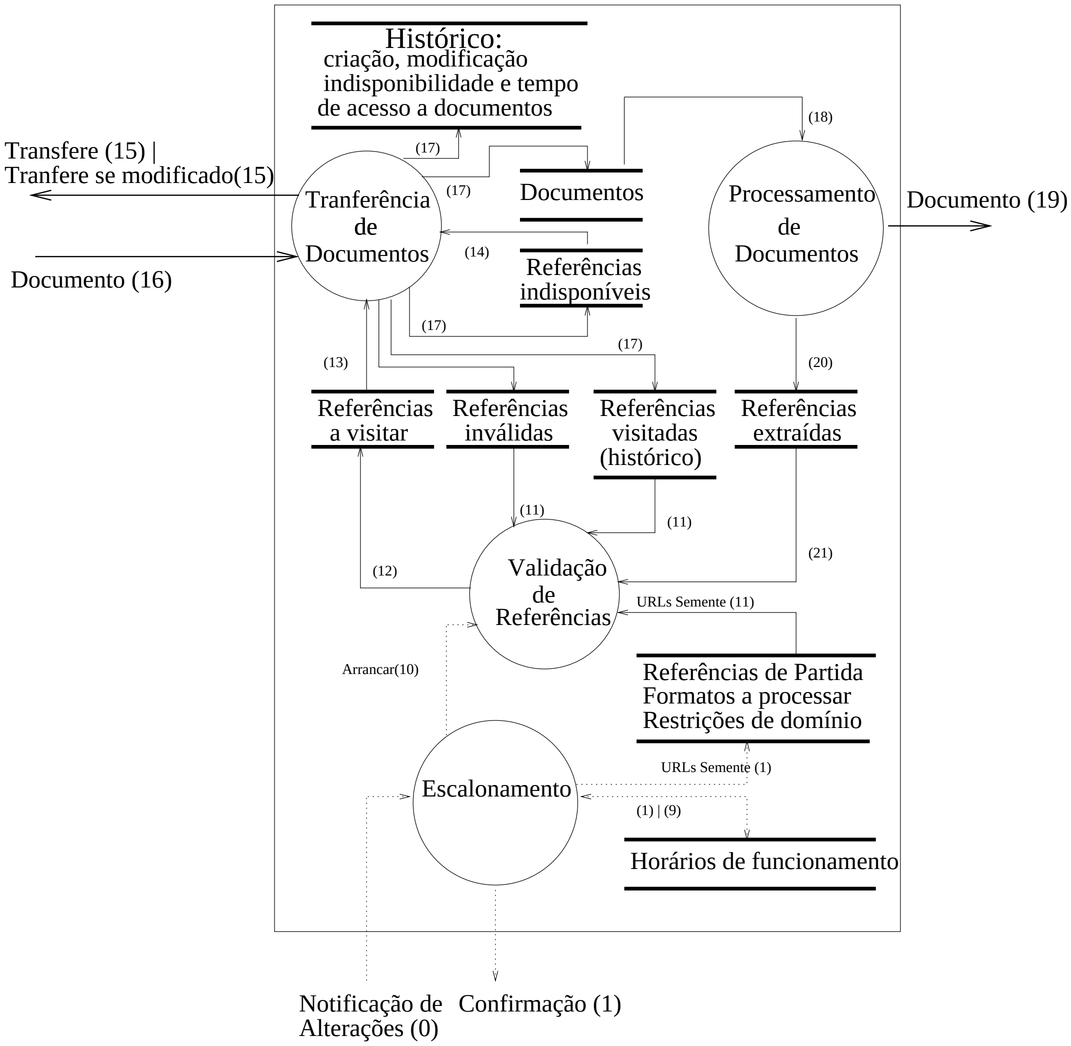


Figura 11: Diagrama funcional do Robot

Embora o robot seja normalmente invocado regularmente por um componente responsável pelo seu escalonamento (ver Figura 11), pode ser notificado pelos servidores de informação ou pelos seus agentes, de alterações registadas no espaço de informação. Nesse caso, pode ser invocado extraordinariamente.

Embora seja funcionalmente muito simples, a complexidade deste componente cresce se houver a pretensão de colectar documentos num espaço de informação muito grande que é o que acontece com os serviços de busca Internet. Com o aumento da escala, é necessário aumentar a eficiência de diversos componentes, nomeadamente o que faz consultas ao serviço de nomes, o que detecta a transferência de informação duplicada, os protocolos de acesso e utilização da largura de banda disponível [HN99, Goo, Int].

Uma outra fonte de complexidade pode ser a pretensão de indexar as partes de Web escondido [RM01] e personalizado [MB98]. Para alcançar este objectivo, são necessários módulos que usem algoritmos e mecanismos de aprendizagem computacional.

Os robots utilizados estão bastante mal documentados na literatura principalmente por questões comerciais[13].

De qualquer forma, por mais complexo que seja o robot há sempre um subconjunto de páginas que ficam por indexar. Há uma parte do web dinâmica em que se executam programas do lado do cliente e que não são normalmente visitadas pelos robots.

Dispõe-se para já de um robot bastante simples, porque no âmbito desta tese utilizaram-se apenas documentos da colecção TREC. Os aspectos de colecta de informação serão objecto de trabalho futuro [CMBS01].

3.4 Indexador de Documentos

A estrutura do indexador de documentos é apresentado na Figura 12. Os seus componentes principais são o reconhecedor sintático e o contador de ocorrências e registo de posição e etiquetagem. O indexador de documentos processa os documentos fonte e constrói ficheiros de documentos indexados, metadados e termos para expansão. A sua concepção é baseada no trabalho descrito em [HS93]. As únicas diferenças têm a ver com os componentes dependentes da língua.

As capacidades do indexador são determinadas pelo tipo de documentos e línguas que é capaz de manipular. Idealmente, deve ser fácil acrescentar um novo tipo de documento ou

[13] O segredo é a alma do negócio

uma nova língua ao indexador.

Para facilitar a introdução de vários tipos de documento, o reconhecedor sintático é gerado quase automaticamente. Grande parte do processo está automatizado (com base no DTD[14]), embora haja ainda necessidade de trabalho manual. O mesmo acontece no detector de tipos que é principalmente baseado em regras heurísticas. Uma das heurísticas mais usadas é a extensão do ficheiro.

O reconhecimento da língua é feito usando *anagramas*[15] e dicionários negativos [Gre95]. Trata-se de um passo prévio indispensável para invocação dos componentes dependentes da língua. Este componente não está para já a ser usado porque neste trabalho são manipulados apenas documentos em inglês.

Para processar um determinado documento, o indexador começa por determinar o seu tipo. Caso o tipo seja suportado, invoca o respectivo reconhecedor sintático. Uma vez extraídas as palavras é necessário detectar a língua utilizada. Conhecida a língua, podem-se remover as palavras muito frequentes que não têm qualquer potência de descriminação dos documentos e reduzir as palavras ao seu radical morfológico. Conseguida essa informação, resta contar o número de ocorrências de cada termo e registar as posições no texto em que ocorre e também associar etiquetas que identificam os componentes do documento em que ocorre.

Processado o documento, está disponível a informação necessária para ser armazenada nos ficheiros de saída.

3.5 Base de Dados de Texto

A Base de Dados de Texto[16] é composta pelo Engenho de Pesquisa, que é descrito mais pormenorizadamente na Figura 14, uma cache de resultados, os termos para expansão e a meta-informação dos documentos. O seu diagrama funcional esta apresentado na Figura 13.

A BDT (esquematicamente representada na Figura 13) devolve os documentos considerados relevantes para uma dada interrogação.

Chegada uma interrogação, começa por consultar a cache de resultados anteriores para verificar se já foi feita uma interrogação idêntica. Se for este o caso, é devolvida a mesma resposta ao utilizador. Se por questões de poupança de espaço em cache, só se mantiver o

[14] O Document Type Definition do SGML

[15] *ngrams*, em inglês

[16] Que basicamente tem as funções de search engine

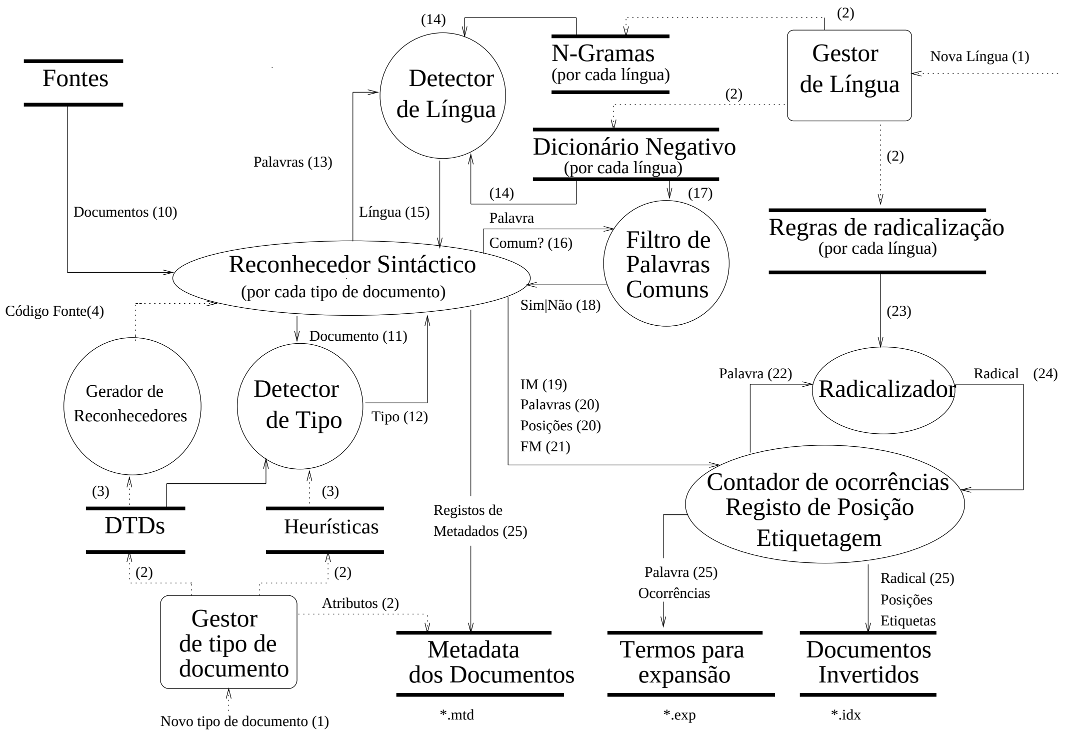


Figura 12: Indexador de Documentos

identificador dos documentos, a resposta a devolver ao utilizador precisa de ser composta consultando a base de dados com a meta-informação dos documentos.
Caso contrário, a interrogação é pré-processada com o objectivo da extração dos termos e respectiva informação de ocorrência (*tf* na Figura 14). Outro tipo de informação relevante a extrair, depende naturalmente do formato da interrogação mas pode incluir a língua, número máximo de documentos pretendidos e outras restrições a aplicar aos documentos a devolver.

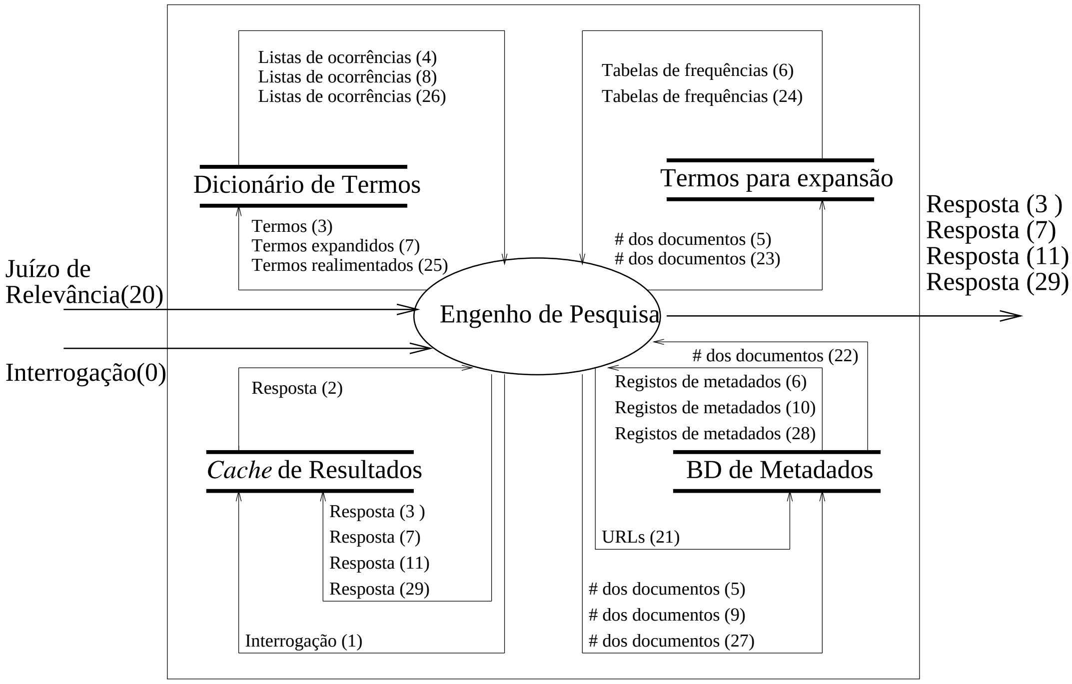


Figura 13: Base de Dados de Texto

Segue-se o acesso ao dicionário de termos[17] no sentido de obter as listas de ocorrências dos termos da interrogação. As listas de ocorrências são ordenadas por ordem inversa da quantidade dos documentos em que ocorrem (*idf* na Figura 14), isto é pela potência de descriminação dos respectivos termos. Os registos correspondentes a cada documento podem por sua vez ser ordenados por ordem crescente de frequência.

[17] Objecto que usa uma estrura de dados o mais eficiente possível para permitir o acesso ao ficheiro invertido

Esta ordenação pode permitir a utilização de algoritmos de corte[18] [KS96], para evitar o processamento quer de listas de ocorrência correspondentes a termos com fraca potência de descriminação quer a registos de documentos em que o termo seja pouco frequente.

Para determinar a similaridade entre a interrogação e os documentos da BDT, executa-se um procedimento que usa um conjunto limitado de acumuladores, um por documento. Os termos da interrogação são processados pela ordem de importância e, acumulados os valores parciais de similaridade com os documentos nos acumuladores respectivos. O processamento termina quando forem processadas todas as listas de ocorrência ou uma fracção delas considerada relevante pelos algoritmos de corte.

Quando se atinge o número máximo de acumuladores, processa-se apenas as ocorrências de documentos já existentes.

Uma vez obtida a pontuação[19] dos documentos mais relevantes para a interrogação (em número determinado pelos acumuladores disponíveis), é necessário ordená-los por ordem decrescente de pontuação.

Do conjunto de documentos disponíveis, apenas um subconjunto dos M documentos mais pontuados são considerados para fazer parte da resposta. O valor M é normalmente um múltiplo de N (número de documentos pedidos pelo utilizador).

Se não houver expansão automática da interrogação, uma resposta de N items é prontamente devolvida ao utilizador (Resposta(20) na Figura 14). Ao mesmo tempo a resposta com M items é mantida em cache (Resposta(21) na Figura 14). A manutenção em cache de M items, salvaguarda futuros pedidos de mais items de resposta pelo utilizador.

Caso a expansão esteja activada, é composta uma nova interrogação usando os termos para expansão de entre um subconjunto dos documentos mais pontuados (ver Figura 14). A interrogação expandida é então submetida ao sistema. Este procedimento é similar ao usado para processar interrogações sob a forma de julgamentos de relevância (JRs), como se mostra na Figura 14.

Os julgamentos de relevância não são mais que uma resposta devolvida pelo sistema com os documentos julgados relevantes pelo utilizador, já assinalados.

---

[18] *Thresholding algorithms*, na terminologia inglesa

[19] Com base na similaridade com a interrogação

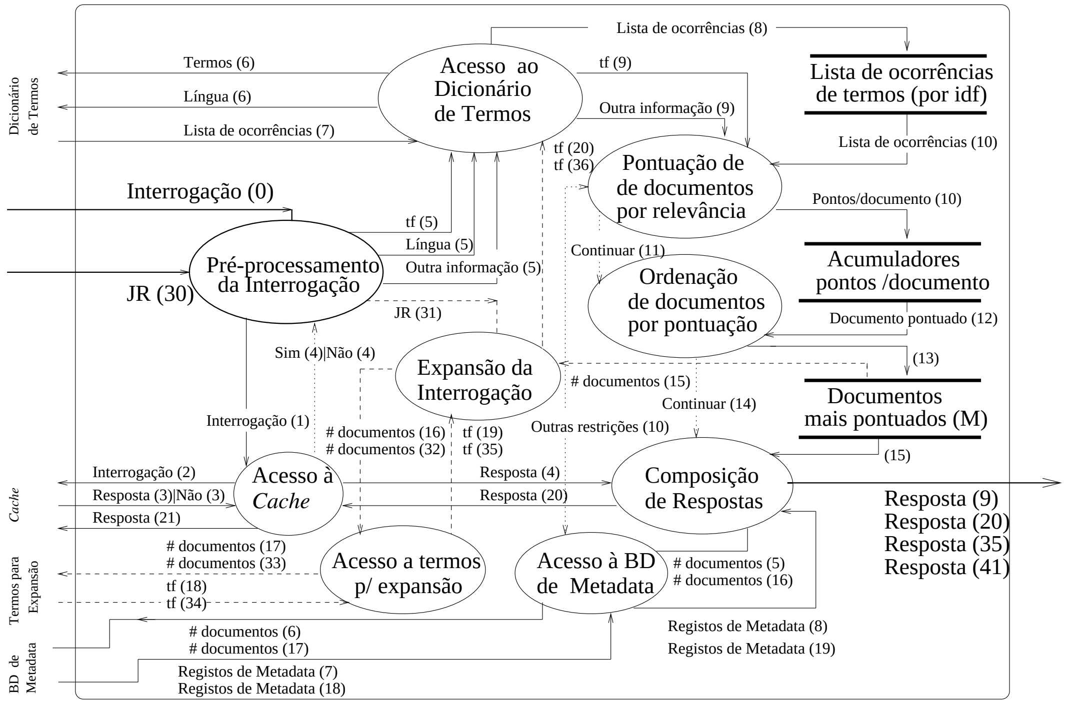


Figura 14: Engenho de Pesquisa, componente principal da BDT

3.6 Encaminhador de Interrogações

O Encaminhador de Interrogações, cujo esquema funcional é apresentado na Figura 15 tem como principais componentes o Selector, o Fusor, o Detector de Sobreposição e módulos que representam, ao nível local, o cliente e as BDT remotas com os quais existe interacção.
O encaminhador de interrogações (ver Figura 15) selecciona as BDTs relevantes para uma dada interrogação (I) e junta as respectivas respostas (R_i) numa única (R), que devolve ao utilizador.
Este componente é o mais importante da arquitectura distribuída, uma vez que as suas funções são quase todas justificadas pela necessidade de distribuição. É nos seus módulos que são colocados os algoritmos de selecção, fusão, sobreposição e correspondência de interrogações e resultados.
O encaminhador pode funcionar em dois modos de operação distintos designados por *local* e *total*, uma analogia com o modo *chaining* e *referral* do X.500 [ITU93], indicados através de um sinal de controlo do utilizador ou por configuração.
No modo local, constrói a resposta (R(3) na Figura 15) com a informação que dispõe. Essa resposta pode ser uma indicação quantitativa[20] ou pode devolver como referência BDTs ou EIs com documentos relevantes.
No modo total, selecciona as BDs relevantes para uma dada interrogação (BDT_1 a iBDT_k na Figura 15), encaminhando-a em conformidade. A selecção é determinada pelos sumários dos conteúdos das BDTs e pela informação de sobreposição disponível. Caso a selecção indique um grande número de BDTs com importância próxima, o sistema pode pedir ao utilizador para refinar a sua interrogação ou inclusive construir uma resposta parcial. A resposta parcial é construída com base numa selecção aleatória das fontes a interrogar. Neste modo de funcionamento, a utilização da cache pode diminuir bastante o tempo de resposta.
No caso da realimentação de relevância (JR na Figura 15), o sistema selecciona as BDTs apenas pelo URL dos documentos.
Seleccionadas as BDTs relevantes, a interrogação é feita a cada uma delas individualmente ($I_1(5)$ a $I_k(5)$ na Figura 15) que é suposto devolver uma resposta diferente ($R_1(6)$ a $R_k(6)$ na Figura 15). A fusão das respostas numa única (R(9) na Figura 15) para devolver utilizador, pode ser feita das mais diversas formas. Os algoritmos são baseados em regras de fusão

[20] Do tipo a BD_1 tem 2 documentos com o autor Salton

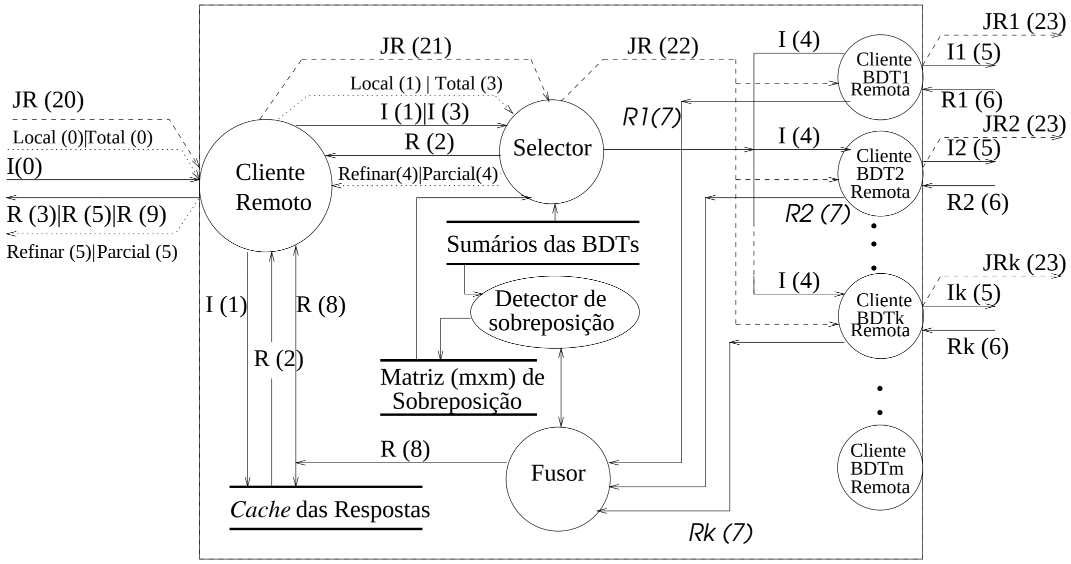


Figura 15: Encaminhador de Interrogações

heurísticas [BKFS95], com mais ou menos informação devolvida pela BDT.

A sobreposição entre diferentes BDTs pode ser usada quer para aumento da eficácia quer da eficiência dos mecanismos de selecção de BDTs e fusão de resultados. Esta sobreposição pode ser calculada com base na sobreposição das respostas devolvidas por cada uma das BDTs. Como essa informação não pode ser conhecida a priori, podem ser usados métodos para a sua estimação. Esses aspectos serão discutidos com maior detalhe no Capítulo 4.

Um factor de complexidade adicional presente no encaminhador de interrogações reside na necessidade de fazer correspondências ou traduções de interrogações [CCGMP97, CCGMP96], metadados e respostas.

3.7 Realimentação de relevância

A realimentação de relevância [Har92b] é uma técnica estabelecida para melhorar a interrogação inicialmente colocada pelo utilizador que muitas vezes desconhece a colecção de documentos subjacente. Existe uma variante deste mecanismo em que não intervem o utilizador e é designada por expansão da interrogação.

Também nos sistemas distribuídos de RI se melhora a eficácia da resposta com a utilização da realimentação da relevância e expansão da interrogação [XC98]. O problema que se coloca é o custo destas operações. Esse custo resulta do facto dos documentos não estarem disponíveis no EI.

Se se usar a mesma abordagem que na BDT centralizada. Nesse caso, o EI selecciona as BDTs, interroga-as e funde as respostas. Utiliza os os documentos mais pontuados da resposta (assinalados pelo utilizador ou automaticamente) para refazer a interrogação e recomeçar o processo. Alternativamente poderia usar apenas o subconjunto das BDTs mais rápidas a responder para este processo de modificação da interrogação ou mesmo uma colecção de treino local [XC98], construída previamente a partir das BDTs disponíveis, usando um método de sonda[21].

3.8 Actualizações incrementais dos ficheiros invertidos

Muitos dos sistemas de RI actuais assumem um corpo de documentos relativamente estático. O ficheiro invertido é construído a partir do nada, usando o conjunto de documentos. Periodicamente (fim de semana ou uma vez por mês) com a chegada de novos documentos, o ficheiro invertido é completamente reconstruído. Trata-se de uma operação cara mas que é amortizada com os vários dias de operação.

Em muitos ambientes de RI, em que o corpo de documentos é bastante dinâmico, essa abordagem não é praticável. Se se indexar mensagens de correio electrónico ou notícias, a informação mais recente é a mais importante. Nessas circunstâncias, é desejável actualizar os índices à medida que vão chegando novos documentos. Na realidade, a actualização é feita por lotes de novos documentos.

Uma outra razão para as actualizações incrementais é a necessidade de manter os sistemas permanentemente disponíveis, 24 horas por dia, 7 dias por semana. Junta-se a este facto o custo da operação de construção de indíces que é bastante cara, em termos de consumo de recursos.

Este assunto foi tratado exaustivamente em [TGMS94], onde se propõe uma estrutura de índices dual, em se trata de maneira diferente as listas de ocorrência longas (palavras mais frequentes) e a listas de ocorrência curtas (palavras menos frequentes). As listas de ocorrências curtas são mantidas em *buckets* que são blocos em disco com tamanho fixo.

[21] probing, em inglês

A localização de uma palavra em particular é determinada por uma função de *hash*. Quando uma lista de ocorrência curta cresce em tamanho e deixa de caber no respectivo *bucket* é transformada uma lista de ocorrência longa que é mantida numa sequência de *chunks* em disco. Os *chunks* são blocos contínuos de disco de tamanho variável.

Para as listas longas é mantida uma directoria em memória central com os respectivos termos e apontadores para os *chunks* em disco respectivos.

Quando se faz uma actualização incremental são construídas listas de ocorrências em memória central, mantendo-se o sistema já a responder também com os novos documentos. A forma de actualizar os índices mantidos em disco com a nova informação mantida em memória pode ser determinada por diferentes políticas.

Essas políticas podem ser definidas de forma a beneficiar o desempenho da actualização dos índices em detrimento dos acessos a disco no processamento de interrogações e vice-versa. Além disso, quanto maior for o espaço em disco para as listas longas, mais rápido é tempo de acesso para leitura. Para se encontrar uma política aplicável há necessidade de encontrar uma solução de compromisso.

Embora este assunto esteja fora do âmbito deste trabalho, optou-se por usar os resultados apresentados em [TGMS94] na concretização do dicionário de termos que é usado tanto na BDT como no EI. Esta opção permite abrir uma porta para as actualizações em linha no Webtrieve (para já) e a investigação da actualização incremental de índices em sistemas distribuídas de RI em trabalho futuro. Acredita-se que pelo facto do sistema ser distribuído permite maior flexibilidade na definição das políticas de actualização. Uma outra vantagem num sistema distribuído de RI é a maior flexibilidade para o suporte da tolerância a faltas, indicada em [TGMS94] como trabalho futuro.

3.9 Interacção entre diferentes componentes

A interacção dos vários componentes é feita usando o protocolo STARTS [GCGM+97]. O protocolo é não orientado ao estado e não usa sessões. Trata-se de um protocolo muito mais simples que o Z39.50[Kun95, PFG+94] que define apenas uma interface de interrogação e que informação a seu respeito deve ser exportada pelas diversas fontes.

De acordo com a terminologia definida no STARTS, o sistema é constituído por um conjunto de recursos. Cada recurso consiste numa ou mais fontes ou simplesmente exporta informação de contacto para as suas fontes. Na arquitectura do Webtrieve, cada recurso

será representado por um servidor que hospeda uma ou mais EIs ou BDTs. Nas interacções com o servidor é sempre identificada a fonte (ou fontes) pretendidas.
Uma fonte é um conjunto de documentos planos com um motor de busca associado que aceita interrogações de clientes e devolve resultados. De acordo com a terminologia usada no Webtrieve, uma fonte é um ou mais servidores de documentos (SD), um robot e uma BDT.
O cliente pode ser um EI que funciona como intermediário entre os clientes e as fontes, isto é um meta-motor de busca.
O STARTS é um protocolo especialmente concebido para permitir o pleno funcionamento de um meta-motor de busca, um componente idêntico ao encaminhador de interrogações (EI), prevendo para esse efeito a sua interacção com os clientes e as várias BDTs, e disponibilizando toda a informação necessária para a selecção das BDTs e para fusão das respectivas respostas.
Uma das componentes mais importantes do STARTS é a sua interface de interrogação, usada no processamento de qualquer interrogação pelos diversos componentes. Cada interrogação STARTS é composta por uma expressão para pontuação[22] e uma expressão para filtragem[23]. A expressão para ordenação é usada para pontuar os documentos e devolvê-los por ordem decrescente de pontuação. A expressão de filtragem é uma expressão boolena que indica que condições devem satisfazer os documentos para integrar a resposta. Estão incluídos os operadores mais comuns que permitem a elaboração de expressões de pontuação e filtragem complexas.
De acordo com o STARTS, na resposta devem ser incluídas estatísticas que permitam ao EI, se for necessário repontuar cada documento devolvido em cada uma das respostas.
São disponibilizados comandos para importação de sumários de conteúdo gerados tanto pelas BDT como pelos EIs dos níveis intermédios da hierarquia. Como vai ser mais tarde comprovado, há uma ligeira diferença entre o formato do sumário de conteúdo STARTS e do Webtrieve (ver Capítulo 6), embora sejam compatíveis.
No Webtrieve, há uma série de facilidades adicionais que permitem uma maior prosmicuidade entre os diversos motores, com as seguintes vantagens:

1. Partilha dos documentos fonte;

2. Existência de um identificador único para cada documento no domínio;

[22] ranking expression
[23] filter expression

Nº	Conteúdo	Nº	Conteúdo
901	declined	906	January
902	earlier	907	production
903	government	908	Industrial production
904	Industrial	909	production Italy
905	Italy	910	Italy declined

Tabela 7: Interrogações de teste para a colecção de teste

3. A utilização de algoritmos de alocação para decidir em que BDT do domínio colocar os índices para um dado documento. Esta abordagem pode permitir a existência de servidores temáticos e mesmo hierarquias temáticas; e,

4. A existência de replicação da informação para permitir tolerância a faltas com a disponibilidade prévia dos níveis de sobreposição, ou a simulação em laboratório de ambientes de replicação coordenada ou não coordenada.

Estas facilidades fazem sentido para motores na mesma organização ou em organizações que mantenham algum tipo de coordenação técnica. E, naturalmente, são facilidades bastante úteis para experiências laboratoriais de recuperação distribuída de informação.

3.10 Avaliação

O esquema usado para avaliação da arquitectura está ilustrado na Figura 16, onde estão representados os vários tipos de estímulos exteriores a que pode estar sujeito um sistema de RI: interrogações ou actualizações do espaço de informação indexado. É a qualidade da reacção do sistema a esses estímulos que importa avaliar. Associado a cada estímulo existe o respectivo ritmo de chegada.

O primeiro passo na avaliação da arquitectura é verificar se o sistema devolve uma resposta correcta. Isto significa ter alguma medida da *eficácia do sistema*.

As diferenças básicas entre o funcionamento para avaliação, apresentado na Figura 16 e o funcionamento operacional, está na utilização pelo sistema de interrogações de teste e colecções de teste [Har93].

Esta abordagem, tradicionalmente usada por investigadores deste domínio, permite a avaliação da arquitectura do sistema em ambiente laboratorial, antes mesmo de ser usada em

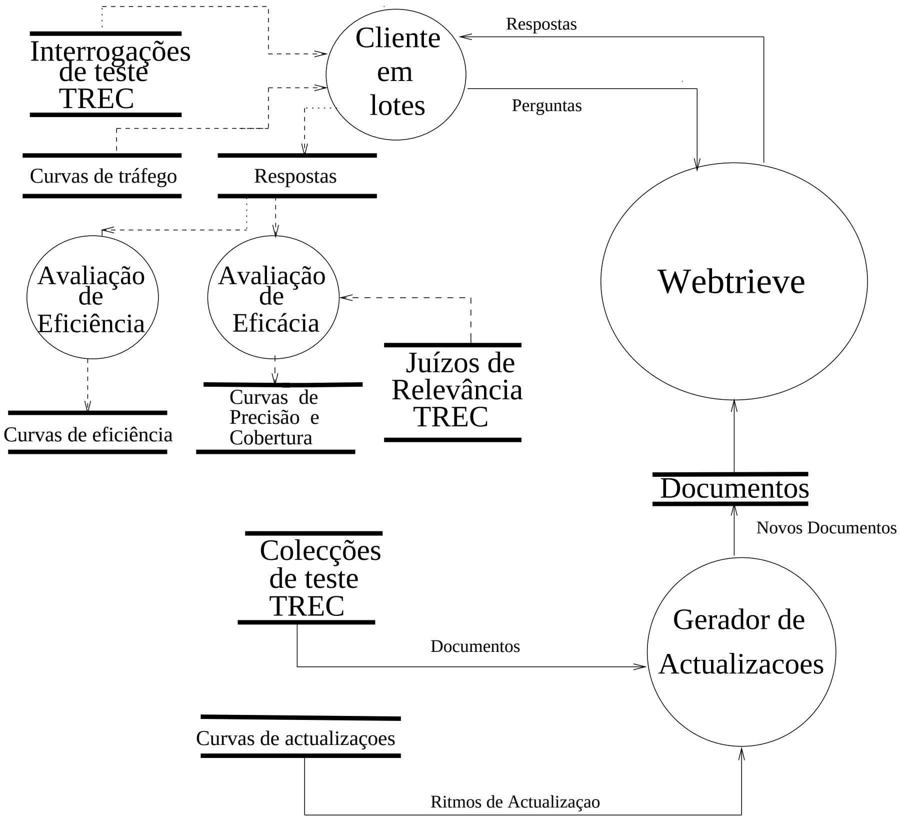


Figura 16: Esquema para avaliação da arquitectura

ambiente operacional.

Na Tabela 7 são apresentadas as interrogações usadas para avaliação da colecção de teste. Na Tabela 8 estão apresentados os julgamentos de relevância para as interrogações de teste para essa mesma colecção. A primeira coluna é o número da interrogação, a segunda um campo usado no formato dos julgamentos de relevância do TREC que não é aqui usado, a terceira a identificação do documento Wall Street Journal e a quarta contém um valor binário que indica a relevância ou não para a interrogação.

O cliente em lotes usa como fonte as interrogações disponibilizadas em colecções de teste [Har93]. As respostas, devolvidas pelo sistema a essas interrogações, são comparadas com os juízos de relevância correspondentes, também incluídos na colecção de teste, para avaliar a eficácia do sistema. A eficácia é apresentada sob a forma de curvas de precisão e cobertura[24] [Sal91, BG94].

O passo seguinte da avaliação é a verificação da eficiência do sistema na resposta a interrogações com diferentes ritmos de chegada.

No caso do cliente em lotes, as interrogações podem ser desencadeadas de acordo com ritmos de chegada típicos ou obtidos experimentalmente. Estes últimos são obtidos analisando registos históricos de servidores *proxy* para interrogações aos serviços de busca bem conhecidos da Internet. Neste caso, a eficiência do sistema pode ser confrontada também com os tempos de resposta desses serviços.

Finalmente, é preciso avaliar como reage o sistema a alterações do espaço de informação indexado. Isto significa medir os tempos de actualização em linha do sistema, quando confrontado com diferentes ritmos (típicos ou experimentais) de chegada de actualizações. As curvas experimentais para chegada de actualizações podem ser obtidas, como já foi descrito anteriormente, com auxílio o do robot. Uma vez aprovada no exame laboratorial, a arquitectura está pronta para ser usada e avaliada em ambiente operacional.

3.11 Discussão

A arquitectura implementada cumpre em termos conceptuais os requisitos necessários para servir de bancada de ensaios para a avaliação de modelos e algoritmos dos componentes indispensáveis à distribuição dos serviços de busca, nomeadamente o encaminhador de interrogações. Esta convicção já foi efectivada com a sua utilização na obtenção de dados

[24] *recall*, na terminologia inglesa

NºInt	X	Identificação do documento	Rel
901	0	WSJ870324-0001	1
901	0	WSJ870323-0180	1
902	0	WSJ870323-0172	1
903	0	WSJ870324-0001	1
903	0	WSJ870323-0178	1
903	0	WSJ870323-0175	1
904	0	WSJ870323-0171	1
905	0	WSJ870323-0180	1
905	0	WSJ870323-0181	1
905	0	WSJ870323-0177	1
906	0	WSJ870323-0174	1
906	0	WSJ870323-0171	1
907	0	WSJ870323-0171	1
907	0	WSJ870324-0001	1
907	0	WSJ870323-0181	1
908	0	WSJ870324-0001	1
908	0	WSJ870323-0177	1
908	0	WSJ870323-0175	1
909	0	WSJ870323-0174	1
909	0	WSJ870324-0001	1
909	0	WSJ870323-0177	1
910	0	WSJ870323-0176	1
910	0	WSJ870323-0174	1
910	0	WSJ870323-0173	1
910	0	WSJ870323-0171	1

Tabela 8: Julgamentos de relevância para as interrogações da colecção de teste

experimentais para novos algoritmos de fusão e selecção, como poderá ser verificado nos Capítulos 5 e 6.

Entretanto e como a arquitectura foi concebida de forma a cobrir os aspectos mais relevantes da recuperação distribuída da informação, alguns dos seus componentes não foram sequer considerados para a primeira fase de concretização. Esses aspectos (expansão de interrogações, realimentação de relevância, concretização de filtros booleanos, caching, actualizações incrementais) terão que ser estudados e concretizados na fase seguinte deste projecto.

Capítulo 4

Sobreposição de Bases de Dados

Sumário

A autonomia de diferentes BDTs que colaboram para responder a uma interrogação do utilizador pode conduzir à existência de replicação não coordenada. Podem existir documentos e índices replicados, provocando a existência de sobreposição entre as diferentes BDTs. Neste capítulo, apresentam-se métodos que permitem ao EI estimar a sobreposição estática entre as suas diferentes BDTs. Esta estimativa permite obter um indicador da sobreposição dinâmica, isto é nas respostas a cada interrogação. É introduzido um método de alocação com replicação para simular em Laboratório uma replicação não coordenada. Para um cenário particular de sobreposição, obtém-se os valores experimentais para a sobreposição dinâmica e estática. É feita uma análise comparativa dos diversos valores obtidos.

4.1 Introdução

A replicação é a chave para conseguir um bom desempenho, elevada disponibilidade e tolerância a faltas em sistemas distribuídos [Lu99, Com95, BMS95]. Entretanto, o sistema replicado deve ser acedido duma maneira consistente pelos seus utilizadores. Dito de outra maneira, a replicação deve ser *transparente* ao utilizador.

A replicação é já bastante usada na Internet. As News da Usenet [Hor83, KL86] e o DNS [Moc87a, Moc87b] são bons exemplos de sistemas replicados.

Existem duas estratégias de replicação: a estática e a dinâmica. Na replicação estática estão pré-definidos o conjunto de recursos a replicar bem como a fonte e o destino da réplica. Na replicação dinâmica, tanto os recursos a replicar como a fonte e o destino são determinados pelos pedidos do utilizador. Este tipo de replicação é normalmente designada por *caching*.

A replicação estática pode ser feita ao nível duma organização tendo como objectivo resolver problemas de tolerância a faltas e sobrecarga do servidor, ou mesmo ultrapassar restrições de acesso. Para esse efeito, cada servidor é replicado totalmente num servidor secundário, de preferência mantido numa outra organização. Como exemplo deste tipo de replicação pode-se apresentar o *Domain Name System*.

A replicação total da informação é usada também nas Usenet News mantendo-se, por príncipio, toda a informação disponível no servidor de cada organização. Os *mirrors* ftp e web são também exemplos de replicação estática embora com maior selectividade feita com base na popularidade da informação ou software. Neste último caso, a replicação é usada no sentido de diminuir a carga do servidor hospedeiro e o tempo de resposta ao utilizador.

O *caching* é uma abordagem dinâmica à replicação, já bastante usada no DNS e outros sistemas distribuídos tal como bases de dados e sistemas operativos. Esta abordagem é agora bastante popular nos serviços de informação Internet com destaque especial para o Web [BBM+97, LG98, MNRZ98, ASF98]. O recurso remoto, uma vez acedido pelo utilizador, é guardado na *cache* do cliente ou dum servidor *proxy* da organização, ficando à disposição para pedidos posteriores. Com esta abordagem, só os recursos mais populares têm réplicas durante mais tempo em servidores da Rede.

Uma outra estratégia de replicação, que pode ser interessante principalmente para sistemas de pesquisa distribuída, é a baseada no conteúdo do recurso. Este critério pode ser usado em associação com outras estratégias nomeadamente a baseada nos pedidos do utilizador. Nesta estratégia de replicação, o recurso pode ser exportado de servidores com cobertura geográfica para servidores temáticos usando algum tipo de algoritmos de aglomeração. Com a replicação baseada em algoritmos de aglomeração, o servidor original pode calcular os servidores onde colocar réplicas para um dado recurso, evitando a necessidade de mensagens de difusão e tendo maior controlo sobre a localização das cópias. Esta estratégia de replicação pode permitir um funcionamento mais eficiente do sistema distribuído replicado de RI pela hipótese de aglomeração (documentos relacionados com a interrogação têm tendência a ser colocados no mesmo servidor) e mecanismos mais eficientes de verificação de consistência (o servidor que mantém o original pode descobrir mais facilmente a localização das suas cópias).

A existência da replicação faz com que exista sobreposição entre diferentes BDTs mesmo

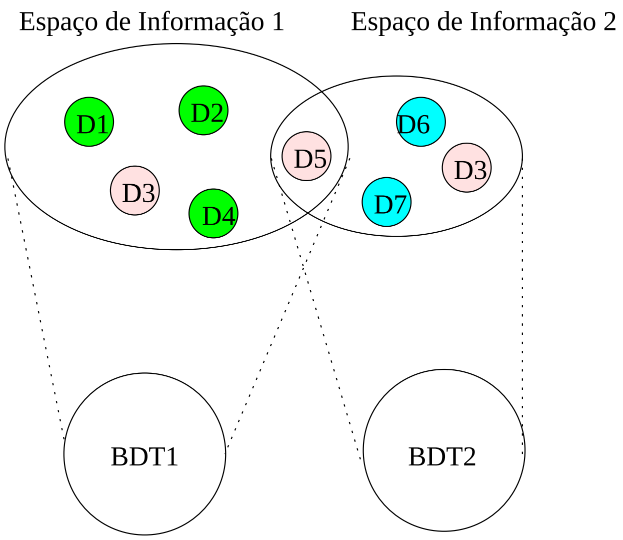


Figura 17: Sobreposição entre BDTs

quando indexam espaços de informação disjuntos. Uma outra situação comum é o espaço de informação indexado por cada BDT já estar sobreposto. Assim, na pesquisa distribuída é muito comum existir algum tipo de sobreposição entre BDTs diferentes.

A sobreposição em BDTs está ilustrada na Figura 17, onde D_i pode ser um documento ou um servidor de documentos[1].

A intenção de esconder do utilizador a existência de componentes do sistema replicados, requer que o sistema de pesquisa distribuído seja capaz de manipular a sobreposição. A sobreposição deve ser detectada no sentido de evitar a devolução de itens idênticos ou bastante similares (em termos de conteúdo), como resposta a uma interrogação. Para além disso e para conseguir eficiência no seu funcionamento, o sistema distribuído deve ser capaz de esconder a replicação de alguns dos seus componentes.

[1] Um servidor WWW, por exemplo

4.2 O que é a sobreposição?

Vai-se definir, para já, a noção de sobreposição que, no âmbito dum ambiente de BDT distribuída, resulta imediatamente de haver réplicas no espaço de informação. Também pode haver sobreposição resultante de duas BDTs indexarem o mesmo espaço de informação, como ilustrado na Figura 17.
Supõe-se a existência de duas BDTs X e Y, cuja cardinalidade é representada por $|X|$ e por $|Y|$. Considere-se também a existência de operações nos conjuntos de documentos.

Definição 1 *A sobreposição da BDT X na BDT Y é definida por*

$$O(X/Y) = \frac{|X \cap Y|}{|X|}$$

Esta definição tem as seguintes propriedades

- Se $X \cap Y = \{\}$, $O(X/Y) = O(Y/X) = 0$
- Se $X = Y$, $O(X/Y) = O(Y/X) = 1$
- Se $X \subset Y$, $O(X/Y) = 1 \wedge 0 < O(Y/X) < 1$
- Se $(X \cap Y) \subset Y \wedge (X \cap Y) \subset X$, $(0 < O(X/Y) < 1) \wedge (0 < O(Y/X) < 1)$

Sejam X e Y duas BDTs que indexam respectivamente o conjuntos de documentos D_X e D_Y.

$$D_X = \{d_1, d_2, d_3, d_4, d_5\}$$

$$D_Y = \{d_1, d_5, d_6, d_7, d_8, d_9, d_{10}\}$$

Para estas duas BDTs tem-se que $X \cap Y = D_X \cap D_Y = \{d_1, d_5\}$. Assim teremos $O(X/Y) = \frac{2}{5} = 0.4$ e $O(Y/X) = \frac{2}{7} = 0.286$.
Pretende-se, a seguir, generalizar a Definição 1 para N BDTs.

Definição 2 *Se existirem n BDTs $X_1, X_2, ..., X_N$, a sobreposição de X_1 nas restantes BDTs é dada pelos documentos de X_1 que existem nas restantes BDTs*

$$O(X_1) = \frac{|X_1 \cap (X_2 \cup X_3 \cup .... \cup X_N)|}{|X_1|}$$

isto é

$$O(X_1) = \frac{|X_1 \cap (\cup_{i=2}^{i=N} X_i)|}{X_1}$$

Observe-se que $O(X_1) \neq \sum_{i=2}^{i=N} O(X_1/X_i)$ se cada documento puder ter mais que uma cópia. No caso de de ter só uma cópia, as duas fórmulas são equivalentes. Uma outra situação simples é cada documento estar replicado em todas as restantes BDTs. Nesse caso $O(X_1) = O(X_1/X_i)$, isto é uma BDT está igualmente sobreposta com todas as restantes BDTs.

Para concretizar, introduz-se um exemplo. Suponha-se a existência de três BDTs (X,Y e Z) que indexam os conjuntos de documentos D_X,D_Y e D_Z.

$$D_X = \{d_1, d_2, d_3, d_4\}$$

$$D_Y = \{d_3, d_4, d_5, d_6\}$$

$$D_Z = \{d_3, d_4, d_7, d_8\}$$

Se se usar a soma das sobreposição um-a-um acima apresentada a sobreposição de X é dada por $O(X) = 0,5 + 0,5$ que cria a impressão errada que os documentos de X estão todos replicados no conjunto das BDTs em Y e Z! Deste modo não é possível calcular a sobreposição de um-para-muitos, com base na sobreposição de um-para-um, se cada documento poder ter mais que uma réplica.

Pode interessar, dada uma sequência ordenada de BDs, $X_1X_2X_3........X_N$, determinar a sobreposição da BDT X_k com as anteriores isto é $O(X_k|X_1, X_2, ..., X_{k-1})$. Claro que se $k = 1$ só há uma BDT e se k=2, basta conhecer a sobreposição com a primeira BDT. Se $k = N$, trata-se de calcular a sobreposição com todas restantes BDTs, isto é $O(X_N|X_1, X_2, ..., X_{N-1}) = O(X_N)$

Seja $O_{max(k)}$ o maior valor de sobreposição que X_k tem com as BDs que a antecedem na sequência. Se $O(X_k)$ for a sobreposição de X_k com as restantes N-1 BDs, teremos

$$O_{max(k)} \leq O(X_k|X_1, X_2, ..., X_{k-1}) \leq O(X_k)$$

Desta forma, temos um intervalo de valores possíveis para o valor que pretendemos calcular. Sabemos também que este valor aumenta com k. Se quisermos saber a máxima influência da sobreposição, basta-nos usar o seu majorante.
Se a definição 2 fosse levada à letra implicaria manter informação de sobreposição bastante detalhada nomeadamente a intersecção de X_1 com todas as combinações possíveis dos outros conjuntos. Essa abordagem seria demasiado cara em termos das estruturas de dados que seria necessário manter para calcular a sobreposição de uma BDT relativamente às restantes. Mesmo para um número pequeno de BDTs (20) a quantidade de sequências possíveis impossibilita o cálculo prévio e o armazenamento destes valores. Resta, por esse facto, a alternativa de calcular ou estimar esses valores em tempo de processamento da interrogação.
Em determinados ambientes mais controlados pode-se conhecer à priori, a técnica usada na distribuição de réplicas dos documentos pelas diferentes BDTs.
Suponhamos, por exemplo, que sabemos que existe no máximo um cópia de cada documento. Neste caso podemos determinar a sobreposição da BDT X_k conhecida a sua sobreposição um-para-um com as BDTs melhores pontuadas. Basta-nos nesse caso, somar essas sobreposições.
Uma outra situação de solução simples é aquela em que todos documentos quando são replicados, possuem replicas em todas as BDTs. Trata-se do caso em que a sobreposição duma BDT é sempre a mesma, independente da sua posição na sequência, com excepção da posição 0, naturalmente.
Mesmo em casos ligeiramente mais complexos, é possível determinar uma estimativa da sobreposição que dependa da posição da sequência da BDT. Por esta razão, dada um conjunto de N BDTs, pode-se apenas manter valores de sobreposição de um-para-um e o valor de sobreposição de um para todos os restantes. Os diferentes valores possíveis ou são calculados em tempo de processamento da interrogação ou então são estimados com base nos anteriores. Esta estimação terá que ser feita usando informação adicional a respeito do método de replicação utilizado.

4.2.1 Sobreposição dinâmica

Nas várias definições apresentadas até agora, pretendeu-se definir e determinar a sobreposição estática entre as diferentes BDTs.
Essas fórmulas e definições podem ser usadas para determinar a sobreposição entre as

respostas devolvidas pelas BDTs a uma dada interrogação. Neste último caso estamos em presença da *sobreposição dinâmica* entre as BDTs, já que varia de interrogação para interrogação.
Observe-se que neste caso duas BDTs idênticas que não usem modelos e/ou fórmulas de pontuação iguais podem ser consideradas diferentes!
Trata-se do tipo de sobreposição que é mais relevante no contexto da RI distribuída. Apresenta-se a definição 3 da sobreposição dinâmica entre duas BDTs.

Definição 3 *Para obter a sobreposição de duas BDTs com base na resposta à interrogação i utiliza-se a seguinte expressão*

$$O(BTD_k/BDT_j) = \frac{r_i(k) \cap r_i(j)}{r_i(k)}$$

em que i é uma interrogação

Consideremos a colecção distribuída, usada como exemplo, em que os documentos estão distribuídos da seguinte forma pelas diferentes BDTs:

$$BDT_0 = \{d_7, d_{10}\}$$
$$BDT_1 = \{d_3, d_7, d_8, d_9\}$$
$$BDT_2 = \{d_1, d_2, d_5\}$$
$$BDT_3 = \{d_0, d_1, d_4, d_6, d_9, d_{11}\}$$

Usando a colecção de teste, obteve-se as respostas parciais para as interrogações apresentadas na Tabela 9.
Isto conduz aos valores para as interrogações da colecção de teste apresentados na Tabela 10.

4.2.2 Sobreposição entre documentos

Nas definições apresentadas apenas se consideram documentos iguais. Considerou-se apenas uma definição binária de igualdade dos documentos.
Podem existir documentos parecidos ou várias versões do mesmo documento. Do ponto de vista da RI, interessa detectar documentos que sejam suficientemente similares para satisfazer a mesma necessidade de informação do utilizador.
Se se quiser usar funções de sobreposição entre documentos diferentes e, desde que o valor da sobreposição varie de 0 até 1, o seu valor pode ser usado também nessas fórmulas.

NºInt.	BDT_0	BDT_1	BDT_2	BDT_3
901		d_9, d_8	d_2, d_5	d_0, d_9, d_{11}
902	d_{10}	d_3, d_8	d_2, d_5	d_0, d_6, d_{11}
903	d_{10}, d_7	d_7, d_9		d_0, d_4, d_9
904	d_7	d_7	d_1, d_5	d_0, d_1, d_6, d_{11}
905		d_8, d_9	d_2, d_5	d_0, d_9, d_{11}
906		d_8, d_9	d_2, d_5	d_0, d_9, d_{11}
907	d_7	d_7	d_1, d_5	d_0, d_1, d_6, d_{11}
908	d_7	d_7	d_1, d_5	d_0, d_1, d_6, d_{11}
909	d_7	d_7, d_8, d_9	d_1, d_2, d_5	$d_0, d_1, d_6, d_9, d_{11}$
910		d_8, d_9	d_2, d_5	d_0, d_9, d_{11}
históricos	d_7, d_{10}	$d3, d_7, d_8, d_9$	d_1, d_2, d_5	$d_0, d_1, d_4, d_6, d_9, d_{11}$

Tabela 9: Documentos incluídos nas respostas parciais das BDTs do exemplo às interrogações

	BDT 0			BDT 1			BDT 2			BDT 3		
Int.	1	2	3	0	2	3	0	1	3	0	1	2
901	0,0	0,0	0,0	0,0	0,0	0,5	0,0	0,0	0,0	0,0	0,33	0,0
902	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0
903	0,5	0,0	0,0	0,5	0,0	0,5	0,0	0,0	0,0	0,0	0,33	0,0
904	1.0	0,0	0,0	1,0	0,0	0,0	0,0	0,0	0,5	0,0	0,0	0,25
905	0,0	0,0	0,0	0,0	0,0	0,5	0,0	0,0	0,0	0,0	0,33	0,0
906	0,0	0,0	0,0	0,0	0,0	0,5	0,0	0,0	0,0	0,0	0,33	0,0
907	1,0	0,0	0,0	1,0	0,0	0,0	0,0	0,0	0,5 0	0,0	0,0	0,25
908	1,0	0,0	0,0	1,0	0,0	0,00	0,0	0,0	0,50	0,0	0,0	0,25
909	1,0	0,0	0,0	0,33	0,0	0,33	0,0	0,0	0,33	0,0	0,2	0,2
910	0,0	0,0	0,0	0,0	0,0	0,5	0,0	0,0	0,0	0,0	0,33	0,0

Tabela 10: Sopreposição por interrogação, calculada com as respostas da colecção usada como exemplo

Trata-se dum aspecto bastante importante, porque podem existir documentos que embora não sejam rigorosamente idênticos têm um grande nível de sobreposição em termos de conteúdo. Este assunto já foi suficientemente bem tratado na literatura [SGM94, BDGM95], pelo que o objectivo, neste âmbito, é integrar coerentemente o trabalho já realizado.
Todavia, no sentido de simplificar as experiências a realizar, utilizam-se por defeito valores binários para a sobreposição entre dois documentos.

4.3 Objectivos

O objectivo deste trabalho é estimar a sobreposição dinâmica entre as diferentes BDTs para uma interrogação dada, isto é a sobreposição das respostas das diferentes BDTs à mesma interrogação.
Trata-se de trabalho integrado no objectivo mais geral desta tese, em que se assume que o conhecimento antecipado do nível de sobreposição permite melhorar a eficácia e eficiência do sistema de RI distribuído.
As funções típicas dum sistema de RI distribuído são a selecção das BDTs relevantes para uma dada interrogação e a fusão das respostas devolvidas numa única a devolver ao utilizador. Essas funções são realizadas no Encaminhador de Interrogações (EI) (ver Figura 6).
Existem vários níveis de sobreposição (igualdade, inclusão, disjunção) que é de todo interesse determinar entre duas ou mais BDTs. Esta informação, se confirmada, permite-nos evitar desperdício de comunicação e processamento no sistema de pesquisa distribuído.
Consideremos para começar a função de selecção de BDTs. Existem situações em que são óbvios os benefícios do conhecimento prévio da sobreposição entre as diferentes BDTs.
Se se conseguir determinar que duas BDTs são iguais, devem ser seleccionadas para interrogação com base em outros critérios que não o conteúdo (por exemplo, distribuição de carga ou mesmo tolerância a faltas). Se tiverem o mesmo conteúdo e também a mesma estratégia de busca, não haverá vantagem em serem seleccionadas conjuntamente. Se tiverem o mesmo conteúdo mas estratégias de busca diferentes, os seus resultados podem ser fundidos mas com objectivos de melhorar a eficácia da resposta. Se a BDT X estiver incluída na BDT Y é, em príncipio, preferível interrogar Y em lugar de X. Para que isso aconteça é necessário que se cumpra a condição $(O(X/Y) = 1) \cap (O(Y/X) = \frac{|X|}{|Y|})$.
Voltemos agora a nossa atenção para a função de fusão de resultados. Se a BDT X e a BDT

Y forem disjuntas, não vale a pena comparar os respectivos conjuntos de resultados para detectar sobreposição. Para que isto aconteça é necessário que $(O(X/Y) = 0) \cap (O(Y/X) = 0)$.

Se nenhuma das condições acima se verificar, as duas BDTs têm apenas um valor de sobreposição que de qualquer maneira deve ser considerado nas funções de fusão e selecção. Nessa situação $0 < O(X/Y) < 1$ e $0 < O(Y/X) < 1$.

O primeiro objectivo é verificar em que medida a sobreposição estática entre diferentes BDTs nos dá uma estimativa da sobreposição dinâmica. Posto isto, interessa propor os algoritmos e as técnicas que permitam a estimação da sobreposição estática entre as diferentes BDTs com base nos respectivos sumários e nos históricos das respostas às interrogações que lhes vão sendo colocadas.

A constatação de que sobreposição não tem sido abordada de forma consistente na maior parte do trabalho existente sobre pesquisa distribuída [Bau97, Bau99, LCC96, GGM95, GGMT94, MZ95] é a principal motivação desta tese.

4.4 Cálculo da sobreposição estática

Não é possível determinar à priori a sobreposição da respostas das várias BDTs a cada interrogação, pelo que é necessáro estimar esses valores.

Em particular, vão-se usar como indicadores da sobreposição dinâmica os valores de sobreposição estática. Serão apresentadas várias técnicas para estimação da sobreposição estática em ambiente operacional onde, ao contrário do Laboratório, não é possível calcular o seu valor exacto.

Para determinar a sobreposição é necessário encontrar uma forma para determinar a igualdade dos documentos. Embora para as experiências laboratoriais se possam usar os identificadores internos para cada documento (simples inteiros), em ambiente operacional esses identificadores precisam de ser substituídos por sumários criptográficos gerados a partir dos URLs. A utilização dos sumários criptográficos permite poupar na quantidade de memória utilizada. Se estiverem disponíveis os URLs referenciados por cada documento podem ser utilizadas heurísticas para detecção de sobreposição de documentos e mesmo de BDTs.

O EI é o componente da BDT distribuída responsável pela selecção e fusão. Para o EI poder estimar a sobreposição entre as diferentes BDTs, precisa de colectar a informação necessária.

Essa informação pode ser colectada das várias fontes de várias formas. Uma das formas é através da informação exportada pelas várias BDTs nos seus sumários de conteúdo. Embora não seja previsto no STARTS, na exportação de qualquer sumário da meta-informação dos documentos de cada fonte, faz sentido que se inclua um campo na meta-informação de cada fonte onde estejam os URLs de todos os seus documentos, e se for o caso também todos os URLs de documentos referenciados. Esta informação é também importante para se poder explorar as hiperligações do WWW na RI distribuída.

Uma outra forma é através de históricos de interrogações, de onde podem ser extraídos os identificadores para os documentos devolvidos nas respostas das várias fontes. O próprio EI pode desencadear um conjunto de interrogações de treino construídas com os termos obtidos do ficheiro invertido que foi construído com base nos sumários dos conteúdos das várias BDTs indexadas. Se os sumários de conteúdo não estiverem disponíveis é necessário um dicionário de termos externo a partir do qual seja possível extrair os termos para a construção de interrogações. Uma vez obtida essa informação, o EI está em condições de calcular a sobreposição entre as diferentes BDTs.

Se houver um grande número de documentos colectados, pode-se usar, em vez de todos os documentos colectados da BDT, apenas uma amostra aleatória dos seus documentos, poupando-se tanto memória como tempo de processamento. A precisão do resultado depende do tamanho da amostra.

Como já foi referido anteriormente, na estimação de sobreposição utilizam-se algoritmos apenas baseados na igualdade dos documentos. Este tipo de abordagem não nos permitiria detectar, por exemplo a sobreposição entre uma BDT que indexa apenas os sumários dos artigos publicados pelo Departamento de Informática, com outra BDT que indexa os artigos completos. Por esta razão, os métodos usados devem, idealmente, usar a sobreposição resultante quer de documentos idênticos quer de documentos que tenham apenas algum grau de sobreposição, como o definido pelo COPS [BDGM95] e SCAM [SGM94].

4.4.1 Matriz de sobreposição entre BDTs

Uma vez colectada a informação necessária, pode-se calcular a matriz de sobreposição para as BDTs indexadas pelo EI. Na matriz de sobreposição (NxN) ficam armazenados os valores de sobreposição um-para-um O(Xk/Xj) e utiliza-se a diagonal principal da matriz para manter a sobreposição de cada BDT com todas as outras.

Quando a matriz é calculada de forma exacta e se dispõe do tamanho das várias bases de

	0	1	2	3
0	50,0	50,0	0,0	0,0
1	25,0	50,0	0,0	25,0
2	0,0	0,0	33,33	33,33
3	0,0	16,6	16,6	33,33

Tabela 11: Matriz de sobreposição para o EI exemplo com 4 BDTs

$$H = \begin{bmatrix} & \sum\frac{|r_1 \cap r_2|}{|r_1|} & \sum\frac{|r_1 \cap r_3|}{|r_1|} & \cdots & \sum\frac{|r_1 \cap r_N|}{|r_1|} \\ \sum\frac{|r_1 \cap r_2|}{|r_2|} & & \sum\frac{|r_2 \cap r_3|}{|r_2|} & \cdots & \sum\frac{|r_2 \cap r_N|}{|r_2|} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \sum\frac{|r_1 \cap r_N|}{|r_N|} & \sum\frac{|r_2 \cap r_N|}{|r_N|} & \sum\frac{|r_3 \cap r_N|}{|r_N|} & & \end{bmatrix}$$

Tabela 12: Históricos da intersecção do conjuntos de resultados devolvidos pelas BDTs

dados, pode-se manter apenas na matriz os tamanhos dos conjuntos intersecção e usar o tamanho de cada BD para a partir dai determinar a sobreposição de um-para-um.

Estes valores podem ser calculados na fase de arranque, de forma a serem prontamente utilizados em tempo de processamento de interrogação.

Como exemplo, apresenta-se na Tabela 11 uma matriz de sobreposição para 4 BDTs indexadas. Os valores, são apresentados em percentagem e foram calculados usando a definição anterior.

4.4.2 Estimação com base nas respostas às interrogações

Para obter a sobreposição entre duas BDTs diferentes com base nos resultados devolvidos para as interrogações, pode ser mantida uma matriz H_{NxN} (sendo N o número de BDTs indexadas pelo EI) com a intersecção dos documentos devolvidos pelas BDTs, duas a duas, às diferentes interrogações.

A informação mantida nessa matriz é apresentada na Tabela 12.

Definição 4 *Para obter a sobreposição de duas BDTs com base na resposta às interrogações usa-se a seguinte expressão*

$$O(BTD_k/BDT_j) = \frac{1}{|I|} \sum_{i \in I} (\frac{|r_i(k) \cap r_i(j)|}{|r_i(k)|})$$

	BDT 0			BDT 1			BDT 2			BDT 3		
	1	2	3	0	2	3	0	1	3	0	1	2
S_m	0,45	0,0	0,0	0,38	0,0	0,28	0,0	0,0	0,18	0,0	0,18	0,10
S_h	0,5	0,0	0,0	0,25	0,0	0,25	0,0	0,00	0,33	0,0	0,17	0,17

Tabela 13: Sobreposição com base nos históricos (S_h) e sobreposição média das interrogações (S_m)

em que i é uma interrogação do conjunto I de interrogações realizadas.

Trata-se do valor médio da sobreposição das interrogações.

Uma outra estratégia consiste em manter para cada uma das BDTs históricos dos URLs das respostas devolvidas. Posto isto, pode-se calcular a sobreposição O(X/Y), tomando aleatóriamente URLs da lista de X e verificando se pertencem à lista de Y. Esta abordagem é similar à usada em [BB98].

Relativamente à formula apresentada anteriormente, obtém-se esta estimativa fazendo uma ligeira modificação na fórmula que se traduz contudo numa grande diferença em termos do volume de informação que precisa de ser mantida.

Foi incluída para comparação uma estimativa da sobreposição entre as BDTs calculada com base nos históricos de interrogaçes usando para esse efeito a fórmula seguinte:

$$O(BTD_k/BDT_j) = \frac{|(\sum_{i\in I} r_i(k)) \cap (\sum_{i\in I} r_i(j))|}{|\sum_{i\in I} r_i(k)|}$$

Tomando como referência a colecção usada como exemplo e as interrogações realizadas, teremos os valores apresentados na Tabela 13.

Como se pode observar (comparando os valores apresentados nas Tabelas 11 e 13) existe uma grande diferença entre os valores de sobreposição calculada com os históricos e os calculados por interrogação. Apenas em casos extremos de sobreposição total ou sobreposição nula, os valores do histórico nos permitem concluir alguma coisa a respeito da sobreposição das interrogações. Neste exemplo, quer o valor médio da sobreposição das respostas quer o valor exacto da sobreposição entre as BDTs (ver Tabela 11) são bastante aproximados.

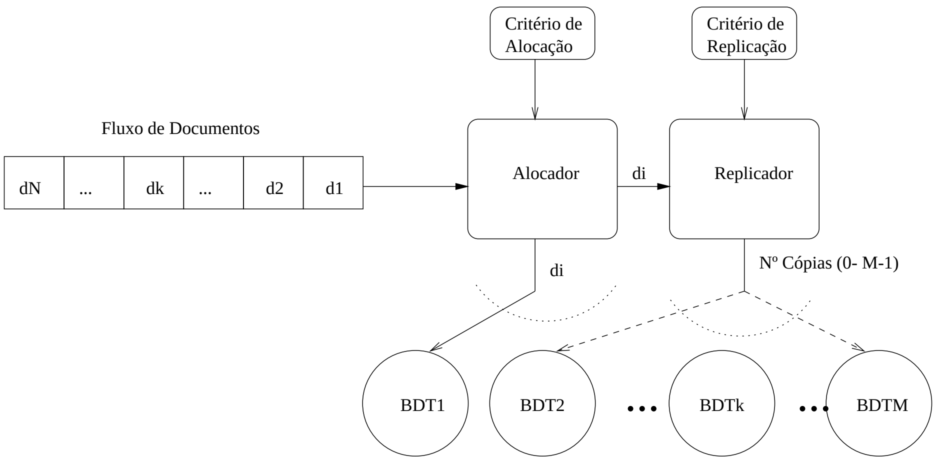


Figura 18: Diagrama simplificado do alocador

4.5 Alocação de Documentos com sobreposição

Num ambiente operacional, em que os vários engenhos de pesquisa são administrados autonomamente, não é possível definir centralmente uma estratégia de alocação dos documentos pelos diferentes engenhos. Uma divisão natural é manter um serviço de pesquisa por cada organização. Nesse caso a localização do representante do documento é determinado pela sua localização geográfica ou por qualquer outro critério.

A situação é diferente se os vários engenhos pertencerem à mesma organização, ou se houver uma coordenação técnica conjunta entre as várias organizações que os hospedam. O mesmo acontece num ambiente laboratorial.

As estratégias de alocação vão ser normalmente combinadas com a estratégia de replicação e não faz sentido que sejam de tipo diferente. Caso isso acontecesse, o algoritmo de alocação seria uma mistura de dois critérios de alocação diferentes. Um diagrama simplificado do alocador é apresentado na Figura 18.

Inicialmente, usaram-se apenas critérios muito simples, no sentido de criar os cenários de experimentação indispensáveis para esta tese. Os algoritmos de aglomeração de documentos [Exp97] serão usados em trabalho futuro.

4.5.1 Concretização

O módulo do Webtrieve correspondente está concretizado com base em 5 entidades principais que correspondem aos destinos da alocação, o fluxo de objectos, o critério de alocação e o critério de replicação. O parâmetros de entrada do alocador são o conjunto de destinos e os dois critérios. Essas entidades correspondem ao diagrama simplificado do alocador, apresentado na Figura 18.

O alocador vai colocando os objectos (que correspondem a documentos neste caso) do fluxo nos destinos (as diferentes sub-colecções) de acordo com os critérios de alocação e replicação.

Os diferentes métodos de alocação e replicação atrás descritos são concretizações de entidades abstractas, pelo que é fácil introduzir novos métodos de alocação e replicação. Também se pode combinar com facilidade diferentes métodos de alocação e replicação embora se tenham que ter alguns cuidados.

Os documentos integrados no fluxo, já correspondem na realidade a documentos invertidos com o vector de registo de ocorrências dos termos e correspondente meta-informação, radicalizados e sem palavras comuns. Uma vez alocados, estão prontos a ser incluídos no ficheiro invertido e BD de meta-informação dos engenhos respectivos. Esta opção é justificada pela existência de documentos que podem ser replicados e no caso de serem alocados em bruto, teriam que ser processados de acordo com o número de cópias.

Uma alternativa que deverá ser estudada no sentido de tornar o processo de alocação e indexação mais eficiente é a utilizar apenas os identificadores dos documentos em vez dos seus representantes. Desta forma , poder-se-ia posteriormente filtrar o ficheiro invertido ou a BD de meta-informação com os resultados da alocação para obter os ficheiros invertidos e meta-informação para cada sub-colecção.

4.5.2 Replicação aleatória com parâmetro de sobreposição

Partimos do príncipio que temos um determinado nível de sobreposição (s) entre os servidores que varia de 0 a 1.0. Isto significa que dado um determinado documento a probabilidade dele nao ser replicado é $p = 1 - s$.

Dado o valor do parâmetro p, podemos estabelecer que a probabilidade de um documento ter uma réplica é q e a probablidade de ter i réplicas é q/i.

Portanto se houver um total de N servidores

$$\text{número de cópias} = \begin{cases} 0 & \text{se } X < p \\ 1 & \text{se } p \leq X < p + q \\ 2 & \text{se } p + q \leq X < p + q + \frac{q}{2} \\ \dots & \dots \\ i & \text{se } p + \sum_{k=1}^{k=i-1} \frac{q}{k} \leq X < p + \sum_{k=1}^{k=i} \frac{q}{k} \\ \dots & \dots \\ N-1 & \text{se } p + \sum_{k=1}^{k=N-1} \frac{q}{k} \leq X < p + 1 \end{cases}$$

Figura 19: Modelo de replicação aleatória com parâmetro de sobreposição ($s = 1 - p$);

$$p + \sum_{i=1}^{i=N-1} \frac{q}{i} = 1$$

Desta forma podemos calcular o valor de q.
Gera-se um valor aleatório X entre 0 e 1. Se X for inferior a p, o documento não tem réplicas. Se tiver o valor entre p e p+q tem 1 réplica, se tiver o valor entre p+q e p+1.5 q tem 2 réplicas e assim sucessivamente.
Conhecido o número de copias (original + réplicas=n) gera-se n valores aleatórios entre 0 e N-1, determinado-se assim em que servidores colocar uma cópia do índice do documento. Posto isto pode-se variar o valor de p para se obter valores de sobreposição dos servidores entre 0 e 1.

4.5.3 Variância na sobreposição

Existe entretanto uma variante deste método no sentido de assegurar uma determinada variância na sobreposição entre as várias BDTs. Para esse efeito, são definidas diferentes classes de sobreposição, suponhamos por exemplo $\{o_1, o_2, o_3\}$.
As diferentes BDTs são distribuídas equitativamente pelas diferentes classes de sobreposição e utiliza-se a mesma estratégia de replicação que a anterior com as seguintes diferenças:

1. Uma vez alocado o documento a determinado destino, o número de cópias é determinado pelo parâmetro o_i da respectiva classe e não por um parâmetro geral; e

2. Os destinos elegíveis para as cópias só podem ser da mesma classe

	5		10		15		20		25		30	
	$\overline{D}$	ΔD	$\overline{D}$	ΔD	$\overline{D}$	ΔD	$\overline{D}$	ΔD	$\overline{D}$	ΔD	$\overline{D}$	ΔD
10	20026	103,2	22882	154,4	25570	116,6	28324	75,4	31077	183,8	33852	172,4
20	10904	90,9	13216	106,8	15548	103,0	17932	97,9	20186	120,6	22252	144,5
40	6302	61,4	8222	63,6	10331	85,6	12334	70,6	14244	69,9	16275	82,0
80	3853	52,3	5537	57,6	7314	64,8	9031	72,1	10777	79,1	12535	87,3

Tabela 14: Número médio ($\overline{D}$) e desvio padrão (ΔD) para documentos em cada BDT

Para experiências no algoritmo de selecção criou-se um cenário de distribuição designado por webt80.var com duas classes, uma com parâmetro 5% e outra 30%. Das 40 BDs metade pertence a cada classe.
As estatísticas obtidas para a colecção distribuída foram as seguintes: média para sobreposição um-a-um 13,96 % e para sobreposição um-para-muitos 58,27 %. Desvio padrão da sobreposição um-para-um 16,12 % e para sobreposição 22,84 %. No que toca à distribuição do número de documentos a média é 5603,6 e o desvio padrão 2425,4. Como se pode perceber por estatísticas, este método permite-nos garantir uma certa variância na sobreposição das várias BDTs.
Como veremos mais adiante, a existência de variância na sobreposição das diferentes BDTs é relevante para o algoritmo de selecção de BDTs.

4.6 Experiências realizadas

Nas experiências vai-se usar como referência a sobreposição dos vários conjuntos de resultados devolvidos pelas várias BDs. Na RI distribuída isso é que é relevante estimar.
Vão ser usadas as interrogações de teste derivadas do TREC. A ideia é difundir as interrogações para todas as BDTs e detectar a sobreposição dos documentos devolvidos pelas diferentes BDTs.
Comecemos por estudar os efeitos do método de alocação com replicação utilizado. Para um determinado valor do parâmetro de sobreposição (s) que valor se obtém para a sobreposição entre as diferentes BDTs? Como varia esse valor quando se altera o número de BDTs? O número de documentos alocados é de 172,837. Vamos verificar as estatísticas obtidas para os vários valores do parâmetro de sobreposição e número de BDTs. Estas estatísticas ajudar-nos-ão a interpretar os resultados obtidos na fusão e na selecção e mesmo os resultados

	5		10		15		20		25		30	
	$\overline{S_g}$	ΔS_g	$\overline{S_g}$	ΔS_g	$\overline{S_g}$	ΔS_g	$\overline{S_g}$	ΔS_g	$\overline{S_g}$	ΔS_g	$\overline{S_g}$	ΔS_g
10	17,93	0,14	32,13	0,21	42,56	0,23	51,16	0,18	58,31	0,23	64,24	0,22
20	24,65	0,24	41,09	0,28	52,78	0,26	61,43	0,29	67,88	0,28	73,19	0,31
40	34,84	0,40	52,68	0,41	64,49	0,38	72,02	0,33	77,23	0,35	81,39	0,31
80	46,74	0,56	64,85	0,50	74,88	0,42	80,89	0,42	84,94	0,33	87,97	0,30

Tabela 15: Média ($\overline{S_g}$) e desvio padrão (ΔS_g) para sobreposição de cada BDT com todas as restantes

globais do sistema. Nas Tabelas 14, 15 e 16 as colunas indicam os valores do parâmetro de sobreposição (em percentagem) e as linhas o número de BDTs em que os documentos são alocados.

Quanto ao número de documentos, ver Tabela 14, os valores médios aumentam com o parâmetro de sobreposição e diminuem com o número de bases de dados. O desvio padrão é inferior a 1% do número médio de documentos na maior parte dos cenários. Isto significa que, em cada um dos cenários, as BDTs são aproximadamente do mesmo tamanho. Conforme se vai aumentando o parâmetro de sobreposição, a taxa de aumento relativo de número de documentos vai descrescendo. O mesmo acontece com a redução do número de documentos quando se aumenta o número de BDTs.

Os valores de sobreposição global (S_g na Tabela 15) aumentam com o número de BDTs e naturalmente com o parâmetro de sobreposição. O desvio padrão é inferior a 1% em valor absoluto. O mesmo acontece na sobreposição de uma para uma, como se ilustra na Tabela 16.

Na comparação entre os dois tipos de sobreposição, verifica-se que a sobreposição global é aproximadamente o dobro da sobreposição de um-para-um. O coeficiente de correlação linear entre os pares de valores de sobreposição dá 0,99.

Posteriormente foi usado o cenário com 10 Bases de Dados e 0.15 de parâmetro de sobreposição (webt10.15) para análise em ambiente mais realista dos valores de sobreposição, e em que medida os diversos métodos permitem a sua estimação. Optou-se por este cenário devido ao elevado número de resultados.

A primeira experiência realizada foi para verificar em que medida a sobreposição estática entre BDTs permite determinar a sobreposição nas interrogações que usam as mesmas fórmulas de pontuação de documentos.

	5		**10**		**15**		**20**		**25**		**30**	
	$\overline{S}$	ΔS	$\overline{S}$	ΔS	$\overline{S}$	ΔS	$\overline{S}$	ΔS	$\overline{S}$	ΔS	$\overline{S}$	ΔS
10	9,13	0,14	16,28	0,21	21,63	0,23	26,05	0,18	29,55	0,23	32,66	0,22
20	11,81	0,24	19,93	0,28	25,50	0,26	29,95	0,29	33,13	0,28	35,55	0,31
40	16,94	0,40	25,34	0,41	31,27	0,38	35,17	0,33	37,63	0,35	39,92	0,31
80	22,52	0,56	31,51	0,50	36,49	0,49	39,22	0,46	41,59	0,40	43,07	0,40

Tabela 16: Média ($\overline{S}$) e desvio padrão (ΔS) para sobreposição de um-para-um nos vários cenários

Embora os diferentes engenhos usem a mesma fórmula de pontuação, o facto das estatísticas serem diferentes pode fazer com que não sejam devolvidos os mesmos documentos pelas diferentes BDTs. Como o número de documentos são limitados, a sobreposição pode ser diferente da efectiva. Uma outra razão para esse facto é cada BDT devolver apenas os documentos que são considerados relevantes para a interrogação.

Usando 10 interrogações da colecção TREC (151-160) obtiveram-se os resultados apresentados nas Tabelas 17 a 19. Para um total de 200 documentos a devolver ao utilizador, foram pedidos 72 documentos a cada BDT. Cada BDT devolveu em todos os casos o total de documentos pedidos. Por esta razão, podemos verificar que os valores de sobreposição entre duas BDTs para a mesma interrogação são sempre simétricos.

Comparemos para já os valores obtidos nestas tabelas com os apresentados na Tabela 20. Tomemos a BDT 0 como exemplo. Embora a sobreposição estática da BDT 0 com as restantes seja mais ou menos a mesma, existe uma ligeira variação da sobreposição com cada BDT para diferentes interrogações.

Os valores da sobreposição estática da BDT 0 com cada uma das restantes, toma valores entre 21,35% e 22,18% com uma variação máxima de 0,83 %, isto é menos de 1%. A sobreposição por interrogação varia entre 9,72% (interrogação 160 com a BDT 4) e 30,56%(interrogação 158 com a BDT 3) o que corresponde a uma variação máxima de 20,84%. No que toca à sobreposição da BDT 0 com todas as restantes no caso estático temos um valor de 42,69% e no caso dinâmico temos uma variação de 12,5% desde 31,94% (interrogação 152) e 44,44% (interrogação 157).

A sobreposição estática dá apenas uma indicação que se pode considerar directamente se tivermos valores perto de 0% e perto de 100%. Já se sabe por trabalho relacionado que a sobreposição estática pode ser calculada com quanta precisão quanta a desejável usando

		151	152	153	154	155	156	157	158	159	160
	1	18,06	20,83	20,83	18,06	23,61	20,83	19,44	20,83	20,83	13,89
	2	15,28	16,67	19,44	16,67	20,83	22,22	20,83	26,39	26,39	18,06
B	3	18,06	20,83	19,44	18,06	25,00	19,44	22,22	30,56	25,00	13,89
D	4	25,00	18,06	26,39	19,44	22,22	15,27	20,83	23,61	23,61	9,72
T	5	23,61	19,44	22,22	18,06	25,00	18,06	22,22	22,22	18,06	12,50
	6	16,67	18,06	20,83	11,11	18,06	13,89	15,28	20,83	20,83	13,89
0	7	25,0	16,67	25,00	20,83	22,22	20,83	15,28	20,83	26,39	13,89
	8	27,78	15,28	20,83	22,22	25,00	25,00	22,22	22,22	18,06	15,27
	9	23,61	19,44	25,00	16,67	20,83	15,28	23,61	23,61	19,44	18,06
	T	**41,67**	**31,94**	**41,67**	**38,89**	**38,89**	**38,89**	**44,44**	**43,06**	**41,67**	**33,33**
	0	18,05	20,83	20,83	18,06	23,61	20,83	19,44	20,83	20,83	13,89
	2	25,00	20,83	23,61	19,44	26,39	18,06	20,83	23,61	23,61	16,67
B	3	20,83	19,44	20,83	18,06	23,61	12,5	23,61	18,06	18,06	13,89
D	4	19,44	18,06	22,22	20,83	22,22	18,06	22,22	16,67	20,83	15,28
T	5	23,61	18,06	15,28	18,06	29,17	16,67	19,44	18,06	18,06	16,67
	6	20,83	19,44	26,39	15,28	20,83	9,72	19,44	13,89	18,06	16,67
1	7	29,16	18,06	23,61	16,67	20,83	13,89	19,44	15,28	18,06	16,67
	8	19,44	13,89	19,44	22,22	25,00	19,44	19,44	26,39	11,11	18,06
	9	26,39	20,83	23,61	12,5	16,67	18,06	20,83	12,5	16,67	16,67
	T	**43,05**	**30,55**	**47,22**	**43,06**	**45,83**	**34,72**	**43,05**	**31,94**	**31,94**	**37,50**
	0	15,28	16,67	19,44	16,67	20,83	22,22	20,83	26,39	26,39	18,06
	1	25,00	20,83	23,61	19,44	26,39	18,05	20,83	23,61	23,61	16,67
B	3	15,28	19,44	20,83	20,83	20,83	16,67	22,22	23,61	26,39	19,44
D	4	20,83	19,44	22,22	26,39	25,00	11,11	23,61	19,44	27,78	20,83
T	5	20,83	18,06	15,28	22,22	31,94	18,06	13,89	25,00	23,61	12,50
	6	18,05	20,83	26,39	11,11	25,00	15,28	23,61	22,22	23,61	20,83
2	7	27,78	20,83	23,61	16,67	27,78	19,44	22,22	22,22	22,22	22,22
	8	16,67	15,28	19,44	22,22	20,83	26,39	20,83	25,00	22,22	20,83
	9	27,78	22,22	23,61	19,44	22,22	16,67	22,22	22,22	22,22	16,67
	T	**41,67**	**40,28**	**44,44**	**40,27**	**51,39**	**40,27**	**45,83**	**43,06**	**48,61**	**44,44**
	0	18,06	20,83	19,44	18,06	25,00	19,44	22,22	30,56	25,00	13,89
	1	20,83	19,44	22,22	18,06	23,61	12,50	23,61	18,06	18,06	13,89
B	2	15,28	19,44	20,83	20,83	20,83	16,67	22,22	23,61	26,39	19,44
D	4	20,83	13,89	19,44	22,22	25,00	16,67	2,.61	23,61	25,00	12,50
T	5	20,83	23,61	29,17	20,83	20,83	22,22	23,61	22,22	18,06	15,28
	6	15,28	20,83	26,39	16,67	18,06	18,06	25,00	23,61	23,61	19,44
3	7	25,00	18,06	22.22	18,06	25,00	19,44	18,06	26,39	22,22	16,67
	8	18,06	13,89	23,61	20,83	29,17	19,44	20,83	19,44	22,22	23,61
	9	22,22	20,83	23,61	23,61	26,39	20,83	23,61	23,61	20,83	16,67
	T	**38,89**	**31,94**	**38,89**	**44,44**	**43,06**	**40,28**	**50,0**	**48,61**	**44,44**	**40,28**

Tabela 17: Sobreposição real das respostas parciais no cenário webt10.15 (BDTs 0-3)

		151	152	153	154	155	156	157	158	159	160
	0	25,00	18,06	26,39	19,44	22,22	15,28	20,83	23,61	23,61	9,72
	1	19,44	18,06	22,22	20,83	22,22	18,06	22,22	16,67	20,83	15,28
B	2	20,83	19,44	22,22	26,39	25,00	11,11	23,61	19,44	27,78	20,83
D	3	20,83	13,89	19,44	22,22	25,00	16,67	23,61	23,61	25,00	12,50
T	5	23,61	15,28	29,17	23,61	23,61	19,44	22,22	23,61	25,00	11,11
	6	25,00	16,67	26,39	12,50	20,83	12,5	19,44	18,06	22,22	19,44
4	7	30,55	15,27	22,22	16,67	27,78	23,61	19,44	20,83	26,39	18,06
	8	33,33	11,11	23,61	22,22	29,17	18,06	20,83	19,44	19,44	16,67
	9	26,39	16,67	23,61	16,67	22,22	18,06	22,22	22,22	20,83	11,11
	T	**48,61**	**29,16**	**45,83**	**44,44**	**50,0**	**33,33**	**41,67**	**38,89**	**44,44**	**33,33**
	0	23,61	19,44	22,22	18,06	25,00	18,06	22,22	22,22	18,06	12,50
	1	23,61	18,06	23,61	18,06	29,17	16,67	19,44	18,06	18,06	16,67
B	2	20,83	18,06	15,28	22,22	31,94	18,06	13,89	25,00	23,61	12,50
D	3	20,83	23,61	25,00	20,83	20,83	22,22	23,61	22,22	18,06	15,28
T	4	23,61	15,28	26,39	23,61	23,61	19,44	22,22	23,61	25,00	11,11
	6	16,67	25,00	25,00	13,89	22,22	19,44	20,83	19,44	18,06	13,89
5	7	30,55	23,61	20,83	20,83	29,17	20,83	15,28	20,83	23,61	11,11
	8	23,61	11,11	22,22	20,83	25,00	22,22	22,22	26,39	13,89	16,67
	9	29,17	22,22	22,22	19,44	20,83	13,89	23,61	18,06	20,83	11,11
	T	**38,88**	**38,88**	**43,06**	**43,06**	**56,94**	**40,27**	**45,83**	**43,06**	**41,67**	**29,16**
	0	16,67	18,06	20,83	11,11	18,06	13,89	15,28	20,83	20,83	13,89
	1	20,83	19,44	25,00	15,28	20,83	9,72	19,44	13,89	18,06	16,67
B	2	18,06	20,83	26,39	11,11	25,00	15,28	23,61	22,22	23,61	20,83
D	3	15,28	20,83	23,61	16,67	18,06	18,06	25,00	23,61	23,61	19,44
T	4	25,00	16,67	26,39	12,50	20,83	12,50	19,44	18,06	22,22	19,44
	5	16,67	25,00	25,00	13,89	22,22	19,44	20,83	19,44	18,06	13,89
6	7	26,39	23,61	22,22	15,28	26,39	18,06	19,44	22,22	20,83	23,61
	8	18,06	13,89	25,00	11,11	18,06	13,89	16,67	16,67	18,06	22,22
	9	22,22	20,83	25,00	13,89	19,44	15,28	16,67	16,67	16,67	15,28
	T	**38,89**	**38,89**	**50,00**	**30,55**	**43,06**	**30,55**	**44,44**	**38,89**	**40,28**	**43,06**
	0	25,00	16,67	25,00	20,83	22,22	20,83	15,28	20,83	26,39	13,89
	1	29,16	18,06	27,78	16,67	20,83	13,89	19,44	15,28	18,06	16,67
B	2	27,78	20,83	23,61	16,67	27,78	19,44	22,22	22,22	22,22	22,22
D	3	25,00	18,06	20,83	18,06	25,00	19,44	18,06	26,39	22,22	16,67
T	4	30,56	15,28	22,22	16,67	27,78	23,61	19,44	20,83	26,39	18,06
	5	30,56	23,61	20,83	20,83	29,17	20,83	15,28	20,83	23,61	11,11
7	6	26,39	23,61	22,22	15,28	26,39	18,06	19,44	22,22	20,83	23,61
	8	25,00	13,83	20,83	22,22	20,83	23,61	19,44	23,61	15,28	15,28
	9	30,56	23,61	25,00	16,67	23,61	19,44	16,67	15,28	25,00	19,44
	T	**54,17**	**40,28**	**46,22**	**34,72**	**48,61**	**44,44**	**41,67**	**43,06**	**45,83**	**43,06**

Tabela 18: Sobreposição real das respostas parciais no cenário webt10.15 (BDTs 4-7)

		151	152	153	154	155	156	157	158	159	160
	0	27,78	15,28	20,83	22,22	25,00	25,00	22,22	22,22	18,06	15,28
	1	19,44	13,89	25,00	22,22	25,00	19,44	19,44	26,39	11,11	18,06
B	2	16,67	15,28	19,44	22,22	20,83	26,39	20,83	25,00	22,22	20,83
D	3	18,06	13,89	19,44	20,83	29,17	19,44	20,83	19,44	22,22	23,61
T	4	**33,33**	11,11	23,61	22,22	29,17	18,06	20,83	19,44	19,44	16,67
	5	23,61	11,11	22,22	20,83	25,00	22,22	22,22	26,39	13,89	16,67
8	6	18,06	13,89	25,00	11,11	18,06	13,89	16,67	16,67	18,06	22,22
	7	25,00	13,89	20,83	22,22	20,83	23,61	19,44	23,61	15,28	15,28
	9	25,00	15,28	22,22	18,06	23,61	23,61	19,44	19,44	11,11	18,06
	T	**44,44**	**25,0**	**47,22**	**36,11**	**50,0**	**45,83**	**38,89**	**44,44**	**34,72**	**43,06**
	0	23,61	19,44	25,00	16,67	20,83	15,28	23,61	23,61	19,44	18,06
	1	26,39	20,83	25,00	12,50	16,67	18,06	20,83	12,5	16,67	16,67
B	2	27,78	22,22	23,61	19,44	22,22	16,67	22,22	22,22	22,22	16,67
D	3	22,22	20,83	16,66	23,61	26,39	20,83	23,61	23,61	20,83	16,67
T	4	26,39	16,67	23,61	16,67	22,22	18,06	22,22	22,22	20,83	11,11
	5	29,17	22,22	22,22	19,44	20,83	13,89	23,61	18,06	20,83	11,11
9	6	22,22	20,83	25,00	13,89	19,44	15,28	16,67	16,67	16,67	15,28
	7	30,56	23,61	25,00	16,67	23,61	19,44	16,67	15,28	25,00	19,44
	8	25,00	15,28	22,22	18,06	23,61	23,61	19,44	19,44	11,11	18,06
	T	**50,00**	**41,67**	**47,22**	**36,11**	**43,06**	**43,06**	**43,06**	**41,67**	**41,67**	**40.28**

Tabela 19: Sobreposição real das respostas parciais no cenário webt10.15 (BDTs 8-9)

	0	1	2	3	4	5	6	7	8	9
0	**42,69**	21,46	21,62	21,52	21,35	21,74	21,82	22,17	21,71	21,85
1	21,64	**42,43**	21,81	21,38	21,52	21,50	21,57	22,00	21,50	21,71
2	21,65	21,66	**42,50**	21,71	21,39	21,70	21,57	22,06	21,62	21,72
3	21,63	21,32	21,79	**42,38**	21,47	21,75	21,30	21,70	21,58	21,50
4	21,46	21,46	21,48	21,48	**42,54**	21,64	21,36	21,87	21,47	21,56
5	21,83	21,41	21,76	21,74	21,61	**42,61**	21,29	21,88	21,54	21,93
6	21,95	21,51	21,66	21,31	21,37	21,32	**42,73**	22,06	21,69	21,59
7	22,11	21,75	21,96	21,52	21,69	21,72	21,86	**42,97**	21,74	22,14
8	21,54	21,16	21,42	21,30	21,19	21,28	21,40	21,64	**42,25**	21,57
9	21,91	21,59	21,74	21,5	21,50	21,89	21,51	22,27	21,79	**42,50**

Tabela 20: Sobreposição entre as BDTs calculada pela definição para o cenário webt10.15

históricos de respostas. Nesse caso também é simples calcular a sobreposição de um para muitos.

Incluiu-se também a média da sobreposição para as várias interrogações das várias BDTs entre si e de cada uma com todas as restantes (ver Tabela 21). Se tomarmos novamente a BDT 0 como referência os valores de sobreposição variam entre 16,95% e 21,39%. A ideia que se obtém graficando os valores é que a média por interrogação se aproxima mais da sobreposição das respostas embora também não permita a sua estimação.

A sobreposição nas várias interrogações da BDT_0 com as cada uma restantes BDTs e a sua sobreposição global apresentam uma média de coeficiente de correlação de 0,9 com a sobreposição estática e uma correlação de 0,92 com a média de sobreposição das várias interrogações. Concluímos que quer a sobreposição da média das interrogações como a sobreposição estática dão uma boa indicação da sobreposição das interrogações.

Por falta de tempo não foram realizadas algumas experiências que vão ser entretando aqui apresentadas. Uma delas é usar fórmulas de pontuação diferentes para as várias BDTs. A ideia é verificar se o facto de se usar fórmulas distintas altera ou não os valores de sobreposição entre os resultados. Para esse efeito as várias BDTs podem ser divididas em grupos que utilizam as mesmas fórmulas.

Uma outra experiência importante é determinar qual o tamanho mínimo de amostra necessária para um determinado nível de confiança da estimação da sobreposição estática, usando por exemplo os históricos de interrogações.

	0	1	2	3	4	5	6	7	8	9
0	**39,45**	19,72	20,28	21,25	20,42	20,14	16,94	20,69	21,39	20,56
1	19,72	**38,89**	21,81	19,03	19,58	20,14	17,92	19,58	20,00	18,61
2	20,28	21,81	**44,03**	20,56	21,67	20,14	20,70	22,50	20,97	21,52
3	21,25	19,03	20,56	**42,08**	20,28	21,25	20,42	20,97	20,69	21,53
4	20,42	19,58	21,67	20,28	**40,97**	21,67	19,31	22,0	21,39	20,00
5	20,14	20,14	20,14	21,25	21,67	**42,08**	19,44	21,67	20,41	20,14
6	16,94	17,92	20,69	20,41	19,31	19,44	**39,86**	21,80	17,36	18,19
7	20,69	19,58	22,50	20,97	22,08	21,67	21,81	**44,31**	20,00	21,53
8	21,39	20,00	20,97	20,69	21,39	20,42	17,36	20,00	**40,97**	19,58
9	20,56	18,61	21,53	21,53	20,00	20,13	18,19	21,52	19,58	**42,78**

Tabela 21: Sobreposição entre as BDTs calculada com as médias das respostas às interrogações 151-160 em webt10.15

4.6.1 Resultados para sequências

Apresenta-se como exemplo os resultados obtidos usando um cenário com 40 BDs em que se calculou a sobreposição entre BDs numa sequência de 12 BDs. Apresenta-se nas colunas a posição de ordem, o valor real de sobreposição, o máximo dos anteriores e o valor de sobreposição total. Os valores estão apresentados em percentagem.
São apenas valores ilustrativos para uma determinada BDT distribuída e usando uma técnica de replicação particular. De qualquer forma e como a sobreposição penaliza a BDT se o valor de sobreposição for aumentando com a posição, não vai haver modificação da posição de ordem de nenhuma BDT se os valores crescerem da forma apresentada na tabela.

4.7 Trabalho relacionado

Embora a facilidade de publicação e distribuição da informação seja a principal vantagem dos sistemas de informação Internet, muitos problemas precisam ainda de ser resolvidos se possam substituir totalmente os circuitos tradicionais de publicação e distribuição da informação em papel. Um dos problemas mais importantes a resolver é, sem dúvida, a protecção dos direitos de autor [SGM95].
A maior parte do trabalho existente na detecção da sobreposição é no sentido da detecção de sobreposição de documentos para protecção de direitos de autor ou de publicacção

k	real	máximo	total
0	0,0	0,0	64,94
1	31,11	31,11	64.,3
2	42,02	31,74	64,29
3	47,65	31,74	64,36
4	51,52	32,51	65,01
5	52,75	31,59	64,06
6	54,71	31,83	64,41
7	56,54	32,06	64,47
8	56,14	31,14	63,81
9	58,11	31,66	64,63
10	58,76	31,53	64,89
11	59,04	31,80	64,36
12	59,99	32,08	65,36

Tabela 22: Dados ilustrativos no cálculo de sobreposição duma sequência com posição de ordem (k), valor real, máximo e total

[SGM94, BDGM95]. Nesse sentido são usadas estratégias de registo ou assinaturas de documentos para detecção de duplicados publicados noutros locais ou modificados abusivamente por outros autores. Este tipo de abordagem pode ser usada em RI, tal como o proposto pelos autores para evitar que o utilizador seja confrontado com documentos que têm precisamente a mesma informação. A abordagem utilizada não se limita a detectar documentos idênticos mas também documentos com o mesmo conteúdo. A detecção apenas de documentos idênticos trata convenientemente situações em que o mesmo documento é publicado em vários formatos ou mesmo quando apenas algumas partes são publicadas separadamente. Devido ao tempo de processamento associado, este tipo de mecanismo é mais adequado para sistemas de filtragem de informação.

Existe trabalho mais recente [BB98] no sentido de se calcular a sobreposição e tamanho relativo entre os diferentes engenhos de pesquisa mais populares da Internet. A técnica utilizada é calcular a sobreposição de URLs obtidos usando interrogações aleatórias. Trata-se da abordagem mais próxima da que se pretende, embora existam algumas diferenças.

A primeira diferença é que, para além de ser importante a sobreposição de um-para-um, também pode ser importante a sobreposição de um-para-muitos. Isto faz com que o número de valores possíveis seja incomportável, quando se aumenta o número de BDs. Todavia, os números podem diminuir drasticamente se se conseguir fazer este cálculo em tempo de processamento. Diga-se que embora se possa ter acesso previlegiado aos engenhos [BB98] existem limitações de memória e ou de tempo de processamento que impossibilitam o cálculo exacto da sobreposição.

Uma outra diferença importante é que se pretende estimar mais precisamente a sobreposição nas respostas com base na sobreposição estática. O objectivo é precisamente o contrário.

A informação que pode estar disponível no componente que precisa da informação de sobreposição é para além do histórico de interrogações os sumários dos conteúdos das BDTs, que é exportado se se estiver a usar o protocolo STARTS.

Um outro trabalho importante nesta área [LM99], prende-se com a selecção e utilização de réplicas parciais das BDTs permitindo aumentar a eficiência do sistema com a distribuição de carga e utilização de réplicas mais próximas do utilizador (um tipo de caching) para minimizar a latência da rede. Ao mesmo tempo que se aumenta a eficiência do sistema consegue-se manter a eficácia.

Neste trabalho, como será referido em capítulos posteriores o objectivo é aumentar a eficácia do sistema usando como parâmetro a sobreposição, num ambiente de replicação não coordenado.

4.8 Discussão

As técnicas de replicação usadas foram concebidas única e simplesmente para as experiências a realizar nesta tese. Não podem ser usadas em ambientes operacionais. Para esse efeito é necessário combiná-las com algoritmos de aglomeração ou com técnicas de caching de acordo com os pedidos dos utilizadores [LM99].

Não se fez qualquer experiência em que a localização da réplica fosse determinada pelo conteúdo do documento. Trata-se duma abordagem que, embora esteja fora do âmbito deste trabalho, deve ser considerada em ambientes operacionais para minimizar o número de BDTs interrogadas.

Embora as BDTs usadas tivessem uma sobreposição estática bastante similar, a sobreposição obtida em várias interrogações varia bastante mais. Conclui-se que a sobreposição estática entre as BDTs e a média da sobreposição das interrogações passadas embora não possam ser usadas para estimar a sobreposição entre as diferentes BDTs para a nova interrogação nos permitem ter uma indicação a seu respeito.

Conseguiu-se, com os valores obtidos na sobreposição estática, apenas um indicador da

sobreposição que pode existir nas respostas às interrogações. Para se estimar esse valor é necessário usar abordagens que passam pela construção de uma pequena colecção representativa da colecções em presença, usando técnicas de amostragem.
Como se verá no capítulo 6 (que trata da problemática da selecção), a sobreposição estática, embora seja uma indicação grosseira da sobreposição dinâmica, permite melhorar o desempenho do algoritmo de selecção.

Capítulo 5

Fusão de Resultados

Sumário

Este capítulo tem como objectivo estudar o problema da fusão de resultados e, em particular, a variação da eficácia (em termos de precisão e cobertura) com o nível de sobreposição entre as diferentes BDTs. Para esse efeito, é proposto um algoritmo que usa o nível de sobreposição para ordenar o conjunto de resultados a fundir. As diferentes lógicas de fusão propostas na literatura são usadas para calcular a pontuação dos documentos duplicados, sendo usada uma aproximação linear para os documentos não replicados. Para fusão de conjuntos de resultados disjuntos, usam-se técnicas de intercalação simples ou ponderada. A eficácia do algoritmo de fusão é comparada com um algoritmo idêntico que não considera a sobreposição. Estuda-se, para além disso, a variação da eficácia do algoritmo proposto com o nível de sobreposição e com o número de bases de dados.

5.1 Introdução

Uma das funções que é necessário realizar na pesquisa distribuída de BDTs é, uma vez devolvidas as respostas parciais por cada uma das BDT, juntar as respostas numa única a devolver ao utilizador.

Como os itens de resultado são ordenados por ordem de relevância, o problema a resolver é juntar as várias sequências de resultados numa única. Assim, a dificuldade do problema está em ordenar os itens de resultados devolvidos pelas diferentes BDTs.

Trata-se de um problema que já se colocava nas BDT centralizadas mas por outras razões e num contexto ligeiramente diferente. Nesse caso a fusão de dados é enquadrada num problema mais geral da combinação de evidência [BKFS95] em que se inclui também a combinação de diferentes interrogações correspondentes a diferentes formulações da mesma necessidade de informação com o objectivo de aumentar a eficácia dos resultados. A justificação do aumento da eficácia do resultado assenta na imprecisão na formulação das interrogações e no modelo probabilístico subjacente aos vários motores de busca [BKFS95].

É de esperar que a combinação de interrogações formuladas de forma diferente se aproxime mais da necessidade de informação do utilizador. Por outro lado, a soma da resposta de dois sistemas de RI independentes tem tendência a previlegiar os documentos relevantes e ser mais imune ao ruído [Lee97] (os documentos não relevantes[1]).

Enquanto na RI centralizada a fusão de dados pode ser usada opcionalmente para efeitos de aumento da eficácia, na RI distribuída ela é imprescindível para se conseguir tornar transparente ao utilizador a distribuição do sistema.

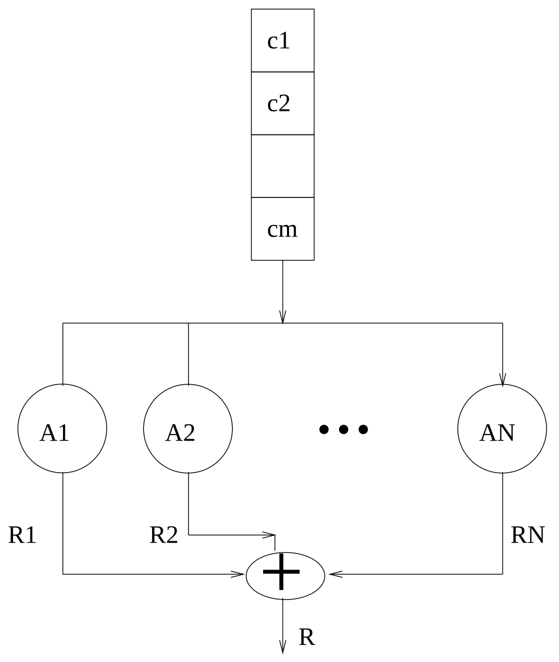
Para facilitar a compreensão das diferenças entre os dois casos, vamos introduzir uma analogia. Suponhamos que temos um concurso e vários avaliadores $\{A1, A2, ..., A_n\}$ para uma determinada população de candidatos C. Como a população de candidatos é muito grande, os avaliadores só devolvem uma seriação dos melhores candidatos. Podemos construir dois cenários de avaliação do conjunto de candidatos C.

No primeiro cenário, cada candidato é avaliado por todos avaliadores. Nesse caso, cada avaliador faz uma seriação dos candidatos e a seriação final é a combinação das seriações de cada avaliador. Este cenário corresponde à problemática da fusão em sistemas com BDT centralizadas, onde todos os documentos são seriados por todas as estratégias de RI. Este cenário está ilustrado na Figura 20.

No segundo cenário, o conjunto de candidatos é dividido num número de partições igual ao número de avaliadores, isto é em $\{C1, C2, ..., C_n\}$. Neste caso, a partição de candidatos C_i é avaliada pelo avaliador A_i. Pretende-e na fusão fazer uma seriação única dos candidatos com base na seriação das partições feita por cada um dos avaliadores. Este cenário corresponde à problemática da fusão em ambientes de RI distribuída. Este cenário está ilustrado na Figura 21. Assuma-se, na Figura 21, que $\sum_{i=1}^{n} l_i = m$, isto é que o total de candidatos é m. l_i é o cardinal de cada subconjunto de candidatos.

Para tornar a analogia mais próxima da realidade, vamos pensar que alguns candidatos podem ser seriados por vários avaliadores, isto é, que pode existir sobreposição entre as

[1] em analogia com a junção de vários canais independentes para transporte do mesmo sinal



ci - candidato i Ai- Avaliador i Ri - Seriação de candidatos por Ai
R - Seriação final dos candidatos

Figura 20: N avaliadores avaliam todos candidatos

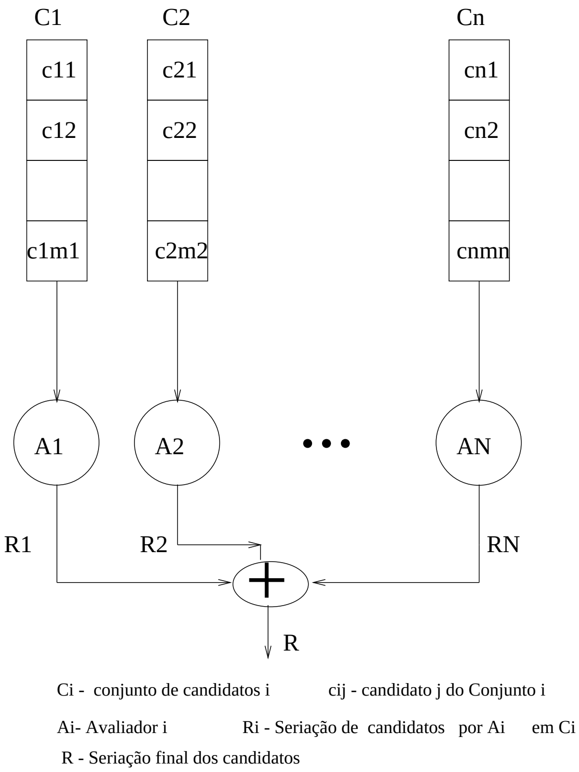


Figura 21: Cada avaliador avalia um subconjunto dos candidatos

diferentes partições. Nessa situação $\sum_{i=1}^{n} l_i \geq m$.

O primeiro cenário exige muitos recursos pois todos os avaliadores têm de pontuar todos os candidatos. Este requesito é bastante problemático quando a população de candidatos é muito grande. Se o algoritmo de pontuação for $O(m)^2$, teremos o tempo de resposta a variar com $O(nm)$.

Na segunda situação, cada avaliador pontua apenas um subconjunto de candidatos. Se a população de candidatos for distribuída equitativamente pelos avaliadores teremos um algoritmo de pontuação $O(\frac{m}{n})$. Neste caso, quando o número de candidatos aumenta, basta aumentar o número de avaliadores, para manter um mesmo nível de tempo de resposta.

Contrariamente ao primeiro cenário, é de esperar que a qualidade da avaliação global piore

[2] Está a usar o O como uma medida da complexidade temporal do algoritmo

com a quantidade de avaliadores. A razão principal desta degradação é a potencial disparidade entre as diferentes avaliações. Os candidatos sobrepostos permitem comparar os diferentes avaliadores e estabelecem pontos de referência.

Regressemos ao caso concreto em análise. Uma diferença importante entre as duas situações resulta do facto de no caso das BDTs centralizadas, os vários *avaliadores* seriarem precisamente os mesmos documentos. É comum assumir-se que, na combinação de resultados, os documentos que sejam devolvidos simultaneamente pelas várias fontes devem ser valorizados em detrimento dos restantes [Lee97]. Também o facto de um documento não ser devolvido por um dos avaliadores significa que é considerado menos relevantes que os restantes. Neste caso, a pontuação ou a posição de um documento pode não estar disponível, mas existe sempre um limite superior, que é o do documento que aparece no final da lista de resultados.

No caso distribuído, supõe-se que os diferentes avaliadores usem um conjunto disjunto de documentos embora possam existir sobreposições. De qualquer forma, o facto de um documento aparecer em várias resultados, pode não significar que seja mais relevante, mas unicamente que está mais vezes replicado! Se determinado documento não é devolvido por determinada BDT, tanto pode significar que não é considerado relevante como que não foi sequer avaliado porque não existe.

Assim é fácil concluir que o problema da fusão de resultados é muito mais complexo em BDTs distribuídas.

No caso distribuído, pode-se justificar uma reavaliação global dos candidatos devolvidos por cada um dos avaliadores para minimizar os efeitos das diferenças de avaliação (ver Figura 22). Esta abordagem pode ser usada em alternativa à simples intercalação com base nos resultados apresentados pelos diferentes avaliadores. Entretanto, para que isto seja possível, é indispensável a disponibilidade da informação pormenorizada sobre os candidatos escolhidos pelos diferentes avaliadores em tempo de fusão. Um outro problema são os custos adicionais de computação.

Uma técnica que permite a normalização dos resultados devolvidos é a reavaliação pelos restantes avaliadores (ou apenas por um outro avaliador) dos melhores e piores candidatos apresentados por cada um dos avaliadores.

Este trabalho tem como objectivos o estudo da relação entre a eficácia de fusão com o nível de sobreposição entre as diferentes BDTs de BDTs , como uma contribuição para a resolução da problemática mais geral da fusão em sistemas de RI distribuída.

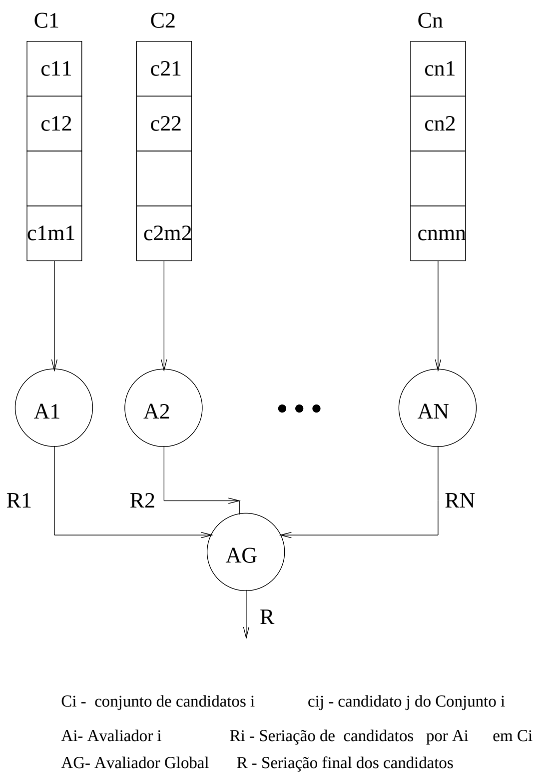


Figura 22: Reavaliação completa dos candidatos na fusão

5.2 Definição formal do problema

Comecemos por introduzir algumas noções preliminares para posteriormente podermos definir o problema em termos matemáticos.
O problema pode ser colocado da seguinte forma [CSS98]:
Dado um conjunto de objectos X cuja cardinalidade pode ser considerada infinita

$$X = \{x_1, x_2, ..., x_m\}$$

Sejam u, v duas instâncias de X, define-se função de preferência PREF como sendo a função de dois argumentos

$$PREF : X^2 \Rightarrow [0,1]$$

Se $PREF(u, v)$ tiver o valor próximo de 1 (de 0) significa uma forte recomendação de colocar o objecto u primeiro que o (depois do) objecto v. Um valor próximo de $\frac{1}{2}$ significa uma abstenção da função de preferência, isto é, que não faz qualquer recomendação.
No contexto de interesse, é possível definir que as funções de preferência são derivadas da existência de uma determinada ordem que se define para os objectos de X [CSS98].
Seja S um conjunto totalmente ordenado e seja $>$ um operador de comparação. Seja f uma função de ordem em S

$$f : X \Rightarrow S$$

A função f induz uma função de preferência R_f tal que

$$R_f(u,v) \begin{cases} 1 & se & f(u) > f(v) \\ 0 & se & f(u) < f(v) \\ \frac{1}{2} & qualquer & outro\ caso \end{cases}$$

Algumas vezes é conveniente permitir que uma função de ordem se abstenha e não dê preferência a um par (u,v). Seja $\perp \notin S$ um símbolo especial tal que

$$f : X \Rightarrow S \cup \{\perp\}$$

$f(u) = \perp$	significa que a ordem de u não está definida para f
$R(u, v) = \frac{1}{2}$	se $f(u) = \perp$ ou $f(v) = \perp$ ou ainda $f(u) = f(v)$.

5.2.1 Exemplos

Considere uma BDT que devolve uma resposta com uma Lista L ordenada de documentos. Suponhamos que está disponível apenas a ordem dos documentos nessa lista.
Podemos definir f como sendo

$f(u) = -k$	para o documento que aparece na posição k na lista L
$f(u) =\perp$	para um documento que não conste da lista L

Se estiver disponível a pontuação do documento relativamente à interrogação colocada, podemos fazer a ordem de precedência coincidir com essa pontuação.
A pontuação normalmente é uma medida da similaridade entre o documento e a interrogação. Trata-se de um valor positivo que aumenta geralmente com a relevância do documento.

$f(u) = sim(u, i)$	onde sim é função de pontuação da BDT
$f(u) =\perp$	para um documento que não conste da lista L
i é a interrogação submetida	

De qualquer forma, assegura-se que todos os elementos de X que não fazem parte da lista L têm uma pontuação abaixo (ou um número de ordem acima) do pior elemento de L.

5.2.2 Realimentação de preferência

A *realimentação de preferência* [CSS98] é definida como sendo o juízo do utilizador relativamente a determinada preferência especificada pelo sistema.
A preferência do utilizador relativamente aos itens da resposta devolvida pelo sistema pode ser devolvida explicitamente pelo utilizador ou indirectamente pelo tempo dispendido pelo utilizador na visualização de cada item de resposta.
A realimentação de preferência pode ser usada para avaliar a qualidade de uma função de preferência e para aumentar a sua eficácia.
É introduzida a medida para avaliar a função de preferência como sendo [CSS98]

$$AGREE(\rho, PREF) = \sum_{u,v:\rho(u)>\rho(v)} PREF(u, v)$$

Onde ρ é a função de preferência do utilizador.
A função AGREE é o somatório do valor da função de preferência quando esta última está de acordo com os juízos de preferência do utilizador.

5.2.3 O problema

No caso da fusão de resultados, o objectivo é a construção de uma função de preferência global, com base em funções de preferência de cada uma das BDTs, isto é $R_1, R_2, ..., R_N$.
Neste caso é possível também assumir que as funções de preferência R_i são induzidas por funções f_i de ordem de precedência[3].
Para a construção da função de preferência global R é necessário obter um método de combinação das funções de preferência elementares.
Especifiquemos agora formalmente o problema que temos entre mãos:

Dado um conjunto de objectos X, um conjunto S, onde $|S| \geq 3$; uma colecção de N funções de precedência f_i
O conjunto de objectos X está dividido em várias partições com ou sem sobreposição $X_1, X_2, X_3, ..., X_N$.

$$X = X_1 \cup X_2 \cup X_3 \cup ... \cup X_N$$

Cada uma das funções de preferência f_i está definida para objectos na correspondente partição X_i. Isto é

$$f_i : X_i \Rightarrow S$$

Encontrar uma função de preferência global PREF para X

$$PREF : X^2 \Rightarrow [0, 1],$$

tal que

$$AGREE(PREF, \rho) > k$$

Em que k é um valor dado.

[3]ranking order, no original em inglês

Podemos pensar em obter a função de preferência como uma combinação linear das funções de preferência elementares [CSS98].

$$PREF(u,v) = \sum_{i=1}^{N} w_i R_{f_i}(u,v)$$

onde os componentes do vector $\vec{w} = \{w_1, w_2, ..., w_N\}$ são valores fraccionários tal que $\sum_{i=1}^{N} w_i = 1$.

Repare-se que, relativamente à situação em que não existe distribuição [CSS98], o problema é colocado de forma ligeiramente diferente, pois o universo de objectos X não é considerado constituído por partições. Nesse caso, cada função de ordem f_i está definida implicita ou explicitamente[4] para todos os objectos de X. Então se o problema nesse caso já é NP-completo [CSS98], nesta situação a a complexidade é aparentemente dum grau superior pelo facto das diferentes funções de ordem f_i estarem definidas para apenas um subconjunto de objectos. Por este facto ainda se justificam mais a utilização de algoritmos aproximados baseados em heurísticas.

Vamos ilustrar mais uma vez a situação com um exemplo.

Se pensarmos num ambiente distribuído com N BDTs em função da informação disponível teremos N funções de preferência com cada uma das f_i definidas da forma acima mencionada, sendo L_i a lista de resultados devolvidos pela BDT_i.

Nesta situação, justifica-se ainda mais a existência de um valor indefinido para a função de precedência, pois um determinado documento u que não exista na lista de resultados L_i terá valor indefinido para f_i, isto é $f_i(u) = \perp$.

Porém, no caso particular em análise, é legítimo pensar que se dispõe relativamente a cada documento:

1. As estatísticas de ocorrência dos termos da interrogação no documento e na BDT original;

2. A pontuação atribuída pelo motor a esse documento e/ou a sua posição de ordem na lista de resultados; e

---

[4] Como é dito na introdução

3. Sumários do conteúdo da respectiva BDT.

Esta informação adicional facilita a determinação de uma função de preferência geral para todos os documentos.

A utilização da informação de sobreposição, que nos interessa estudar, adequa-se, particularmente, quando não se dispõe de informação tão detalhada, mas apenas da pontuação ou posição ordem dos documentos nas várias respostas parciais. É isso que vamos considerar a partir de agora.

5.3 Trabalho relacionado

Relativamente ao trabalho relacionado, convém definir uma série de aspectos que contextualizam as várias propostas apresentadas.

Comecemos por distinguir entre ambientes *homogéneos* de BDTs e ambientes *heterogénos*. Num ambiente homogéneo as várias BDTs utilizam o mesmo modelo para a representação de interrogações e documentos e esquemas equivalentes de pontuação de documentos, face a uma determinada interrogação. Nestes ambientes, o único problema a resolver é a normalização dos resultados para evitar a disparidade da pontuação atribuída pelas diferentes BDTs.

Nos ambientes heterogéneos não é possível fazer nenhum pressuposto a respeito dos modelos e algoritmos subjacentes a cada uma BDTs intervenientes. Nesse caso, é necessário arranjar mecanismos que permitam a comparação das diferentes funções de ordem. Torna-se necessário começar por garantir a compatibilidade da informação trocada entre a BDT e o EI. Essa compatibilidade é necessária tanto em termos da informação que deve ser disponibilidade como da respectiva sintaxe.

Nesse sentido, existe uma proposta norma para um protocolo a usar [GCGM^{+}97] para as interrogações, respostas e outra informação a respeito das fontes. Esse protocolo está concebido no sentido das diferentes BDTs disponibilizarem toda a informação indispensável à fusão de resultados.

5.3.1 Ordem versus pontuação

Os métodos de combinação de resultados que se apresentam nas secções que se seguem baseiam-se na utilização quer da ordem dos documentos na lista de resultados quer da pontuação atribuída pelas várias BDTs. Nesse caso a função de ordem vai ser derivada desses valores. Comecemos pela caracterização e comparação destes dois tipos de medidas. A pontuação, a menos da normalização, é uma medida que fornece um valor absoluto da similaridade ou proximidade entre a interrogação e um documento dado. Desta forma, pode ser usado para comparar dois documentos devolvidos por diferentes BDs.

O valor atribuído como pontuação do documento é obtido com base numa fórmula que depende de determinadas estatísticas da colecção. Por esse facto, o mesmo valor de pontuação devolvido por BDTs diferentes pode referir-se a documentos completamente diferentes. Isto é possível se se tiver estatísticas díspares dos respectivos termos. Por exemplo, o termo computador é muito menos frequente em textos de economia do que em textos de informática. Poderá ter assim uma maior potência de descriminação na BDT de economia.

A única forma de resolver esse problema, é a disseminação das estatísticas totais da colecção para todos os locais [VF95]. Alternativamente, pode-se pontuar cada documento unicamente em função das suas próprias estatísticas de ocorrência dos termos da interrogação, ignorando as estatísticas da sub-colecção em que se encontra integrado.

No caso de se tratarem de colecções com um determinada grau de homogeneidade em termos das estatísticas dos termos, basta normalizar os valores das diferentes BDTs. A normalização corresponde normalmente a fazer cair esses valores no intervalo $[0, 1]$[5].

Um forma simples de fazer essa normalização é [Lee97]

$$p_{norm} = \frac{p_{orig} - p_{min}}{p_{max} - p_{min}}$$

Onde

p_{norm}	é a pontuação normalizada
p_{orig}	é a pontuação original
p_{max}	é a pontuação máxima
p_{min}	é a pontuação mínima

Para esse efeito os valores máximo e mínimo para a pontuação devem poder estar disponíveis. Uma aproximação que pode ser usada, caso isso não aconteça, é o valor máximo e mínimo no actual conjunto de resultados.

Por outro lado, a posição de ordem atribuída a um documento é uma medida relativa que depende não só do documento em questão, mas também da restante população de

[5] ou se estiver em percentagem no intervalo $[0, 100]$

documentos da colecção. É assim mais difícil comparar documentos com base nas posições de ordem em que são devolvidos.

Existe a possibilidade de transformar a posição de ordem numa pontuação do documento, e usar um dos métodos, propostos para combinação de pontuações, para combinar posições de ordem [Lee97]

$$p = 1 - \frac{o-1}{n}$$

Onde

p	valor calculado da pontuação
o	ordem do documento na lista de resultados
n	total de documentos devolvidos

Esta relação mostra que a pontuação aumenta quando a ordem diminui e vice-versa.

Uma diferença básica entre a posição de ordem e a pontuação é a disponibilidade desses valores no conjunto de resultados. Enquanto a posição de ordem é implicita para listas de resultados ordenadas a pontuação pode não estar disponível.

5.3.2 Combinação em BDTs sobrepostas

Apresentam-se a seguir alguns métodos que se baseiam na combinação da pontuação [BKFS95, Lee97] devolvida pelos vários motores em que são usados os mesmos documentos para as várias BDTs.

De acordo com nossa terminologia apresentada em 5.2, consiste em determinar uma função de ordem f com base nas f_i, consideradas para uma das BDT como sendo o valor da pontuação atribuída a cada documento. Essa pontuação traduz uma medida de similaridade entre a interrogação e cada documento. Pode-se assumir, sem perda de generalidade, que se trata de um valor positivo e tanto maior quanto maior for a *proximidade* entre o documento e a interrogação. O requisito para utilização destes métodos é a disponibilidade dos valores de pontuação normalizados.

$CombMIN$ corresponde a uma função em que a pontuação atribuída a um documento é o mínimo da pontuação nas diferentes BDTs.

$$CombMIN : f(u) = \min\{f_1(u), f_2(u), ..., f_N(u)\}$$

Pretende-se com este método salvaguardar situações em que a pontuação atribuída a um documento não relevante possa ser *inflacionado* pela pontuação atribuída por um dos motores.

$CombMAX$ corresponde a uma função em que a pontuação atribuída a um documento é o máximo da pontuação nas diferentes BDTs.

$$CombMAX : f(u) = \max\{f_1(u), f_2(u), ..., f_N(u)\}$$

Com este método salvaguardam-se situações em que a pontuação atribuída a um documento relevante possa ter sido *deflacionada* pela pontuação baixa atribuída por um dos motores.

$CombMED$ corresponde a uma função em que a pontuação atribuída a um documento é a mediana da pontuação nas diferentes BDTs.

$$CombMED : f(u) = mediana\{f_1(u), f_2(u), ..., f_N(u)\}$$

Com este último método, evita-se as situações anómalas que podem ocorrer com os métodos anteriores.

$CombSUM$ corresponde a uma função em que a pontuação atribuída a um documento é a soma ou a média da pontuação nas diferentes BDTs.

$$CombSUM : f(u) = \sum_{i=1}^{N} f_i(u)$$

Algumas variantes permitem evitar que os documentos não devolvidos por alguns dos motores, sejam penalizados pelo $CombSUM$ ou que não sejam suficientemente destacados. $CombANZ$ corresponde a uma função em que se consideram para a média apenas os valores diferentes de 0 e $CombMNZ$ corresponde a uma uma função em que se multiplica a soma com o número de valores diferentes de 0.
Existe ainda $CombGMNZ$ que combina o $CombSUM$ e o $CombMNZ$, com auxílio de um parâmetro λ.

$$CombGMNZ : f(u) = (\sum_{i=1}^{N} f_i(u))(valores \neq o)^{\lambda}$$

Assim

$$CombGMNZ = \begin{cases} CombSUM & se \quad \lambda = 0 \\ CombMNZ & se \quad \lambda = 1 \end{cases}$$

Embora na maior parte dos casos se use a pontuação atribuída aos documentos pelas BDTs, nada impede que se usem as posições de ordem nos resultados devolvidos por cada uma das BDTs.

5.3.3 Combinação em BDT disjuntas

Neste caso, abordaremos a situação de fusão de resultados na pesquisa distribuída de BDTs, em que supostamente os conjuntos de documentos são disjuntos, podendo haver entretanto uma sobreposição marginal.

Vamos apresentar um conjunto de abordagens que se baseiam na combinação da posição de ordem dos objectos na lista de resultados. Todavia, poder-se-ia também usar nesses exemplos a pontuação dos documentos desde que fossem normalizados.

Relativamente aos métodos anteriormente apresentados, tem a vantagem de necessitar de menos informação, isto é, é apenas necessário conhecer a posição de cada documento no item de resultados. Assim considera-se para a BDT_i a função de ordem de precedência como sendo

$$f_i(u) = -k$$

partindo do princípio que u aparece na k-ésima posição da lista L_i.

Uma abordagem possível, agora usando posições de ordem, é a intercalação simples de resultados [CLC95]. Esta abordagem parte do princípio que as diferentes BDTs contribuem com documentos relevantes em igual proporção. Neste caso, basta considerar que cada posição de ordem

$$f(u) = f_i(u) + N * f_i(u) - (i - 1)$$

Com

$f_i(u)$	é a função de ordem de precedência do documento na respectiva BD
N	é o número total de BDs interrogadas
i	é número de ordem da BD

Se existirem 2 BDTs teremos os valores $\{-1, -3, -5, ...\}$ para os documentos de L_1 e, os valores $\{-2, -4, -6, ...\}$ para os documentos da L_2.
Outra abordagem usada é a da *intercalação ponderada* em que se multiplica a pontuação de cada documento com a importância dada à colecção. Uma fórmula para a ponderação é por exemplo [CLC95]

$$p = 1 + |C|.\frac{s - \widehat{s}}{\widehat{s}}$$

$$\text{Onde} \quad \begin{array}{lcl} |C| & = & \text{o número de BDTs interrogadas} \\ \text{s} & = & \text{a pontuação da colecção} \\ \widehat{s} & = & \text{a pontuação média das colecções} \end{array}$$

Genericamente o peso atribuído a cada BDT pode ser o atribuído pelo mecanismo de selecção [Voo95].
No caso de se disporem das estatísticas de cada documento devolvido poder-se-á também no caso de BDTs disjuntas normalizar a pontuação dos documentos devolvidos, usando a repontuação do primeiro e do último classificado.

5.3.4 Discussão

Como seria de esperar, nas BDT sobrepostas os resultados melhoram quando aumenta o número de BTDs. Não existem dados na literatura para BDT disjuntas mas é de supor que os resultados piorem.
Tipicamente, a combinação da pontuação conduz a melhores resultados que a combinação das posições de ordem. Todavia se a pontuação atribuída não estiver disponível ou se os resultados não forem comparáveis, a combinação das posições de ordem pode ser uma alternativa [BKFS95].
Há resultados experimentais que indicam que a normalização das pontuações e sua ponderação com base numa *importância* atribuída a cada BD interrogada [CLC95] também melhora os resultados da fusão.
Os resultados encontrados sugerem que o CombSUM é a melhor técnica de combinação de pontuação [BKFS95]. De qualquer maneira os resultados da combinação são melhores que as respectivas entradas. Trata-se entretanto de combinações de BDs sobrepostas.
Uma outra restrição a estes resultados é que foram encontrados para ambientes homogénos.
Um outro resultado ainda que colateral é que a disseminação das estatísticas da colecção aumenta a eficácia dos resultados [VF95]. Uma outra conclusão importante do mesmo

trabalho [VF95] é que a disseminação é tanto mais necessária quanto maior for o grau de afinidade dos conteúdos dos documentos colocados nas várias BDTs.

5.4 Abordagem proposta

Para avançar com o nosso objectivo de propor estratégias de fusão em ambientes distribuídos comecemos por discutir as vantagens e desvantagens das duas abordagens possíveis:

1. Seriação dos resultados com base nos resultados parcelares
2. Reavaliação dos documentos devolvidos pelas várias BDTs

A primeira abordagem tem a vantagem de necessitar de menor capacidade de computação e comunicação. Entretanto, as BDTs devem devolver informação suficiente para a fusão. Contudo, coloca-se sempre o problema da comparabilidade da pontuação atribuída por diferentes BDTs. Obviamente permite, na maior parte dos casos, resultados menos eficazes que os da reavaliação.

A segunda abordagem necessita que as BDTs devolvam a informação para reavaliação completa dos documentos. A principal vantagem é a comparabilidade da pontuação atribuída aos diferentes documentos. A desvantagem vem do facto de ser necessária mais computação.

A abordagem que vai ser usada é uma abordagem mista. Usa o facto de se poder estabelecer, em certa medida, a comparabilidade das diferentes funções de similaridade com base na pontuação atribuída pelas diferentes funções aos mesmos documentos replicados em diferentes conjuntos de resultados. Caso não exista duplicação e se informação necessária estiver disponível, é sempre possível repontuar alguns documentos de cada motor (o melhor e o pior) no sentido de normalizar os resultados com base na pontuação original. Se isso não for possível, resta a intercalação simples.

Em ambientes homogéneos pode-se usar uma série de medidas que permitam de certa forma a *comparabilidade* entre as funções de ordem das diferentes BDTs. Como normalmente as funções de ordem dependem das estatísticas da colecção, um dos passos nessa direcção é a disponibilidade das estatísticas globais da colecção.

Nesse sentido é conveniente que seja feita a disseminação das estatísticas das várias BDTs. Suponhamos que o termo t_i tem o peso w_i na interrogação original. Se antes de enviar para

BDT_k esse peso for afectado de um factor $\frac{idf_k}{idf}$[6], temos os resultados devolvidos pelas diferentes BDTs normalizados. Os valores idf e idf_k são respectivamente a frequência inversa no documento desse termo para toda a colecção e para a BDT_k. Se essas estatísticas estiverem disponíveis no fusor, a manipulação dos pesos associados aos termos da interrogação pode permitir resultados normalizados de BDT díspares [CLC95].

Se os documentos forem alocados aleatoriamente pelas diferentes BDTs, a disseminação das estatísticas pode ser feita de forma mais relaxada [VF95]. O relaxamento corresponde à intensidade e globalidade necessária na disseminação das estatísticas.

Os resultados anteriormente apresentados foram obtidos em ambientes homogéneos.

Num ambiente de RI distribuída, é de esperar que seja necessário por vezes fundir respostas de BTDs heterogéneas. Nessa situação, a informação disponibilizada pelas várias fontes pode ser diferente. Pode haver fontes que disponibilizem só posição de ordem e outras devolvam conjuntamente a pontuação atribuída aos documentos. Como se tratam de BDTs que usam métodos e modelos diferentes a pontuação (ou número de ordem) atribuída aos vários documentos é bastante mais difícil de comparar. Uma técnica no sentido da comparabilidade é possuir valores das diferentes funções de ordem f_i para o mesmo conjunto documentos em resposta as mesmas interrogações. Aqui assume uma particular importância o nível de *sobreposição entre as diferentes BDTs.*

Para BDTs com elevado nível de sobreposição, uma grande parte dos documentos sobrepostos devolvidos no resultado vai ter pontuação atribuída por várias funções de ordem. Relativamente aos documentos que são apenas devolvidos por alguns motores, a pontuação, embora não conhecida, é inferior a um valor conhecido. Entretanto, podem existir funções ordem contraditórias. A forma de resolução destes conflitos é a ponderação das funções de ordem.

A ideia subjacente ao método proposto é usar os documentos duplicados como pontos de referência. Pode-se, desta forma, estabelecer um certo grau de comparabilidade entre as funções de pontuação usadas pelos diferentes motores.

5.4.1 Descrição do algoritmo

Suponha que existem l BDTs $X_1, X_2, ..., X_l$. Pretende-se fundir os resultados devolvidos pelas diferentes BDTs isto é fundir os conjuntos $R_1, R_2, R_3, ..., R_l$.

[6]inverse document frequency (idf)

A pontuação atribuída aos diferentes documentos devolvidos é dada por $P_i(D_k)$. A respectiva posição de ordem no conjunto de resultados é dada por $O_i(D_k)$.
Numa primeira fase, determina-se o número de documentos sobrepostos nos conjuntos de resultados (dois a dois), obtendo-se a correspondente matriz simétrica de sobreposição. Nessa altura, atribui-se uma numeração sequencial aos documentos, atribuindo-se naturalmente a documentos idênticos em diferentes $R_i s$ a mesma identificação.
Posteriormente, fundem-se os conjuntos de resultados dois a dois, até obter um único conjunto de resultados. Começa-se sempre pelos pares (R_i, R_j) com o maior número de sobreposições. Para os documentos sobrepostos, utiliza-se uma das lógicas de fusão apresentadas para BDT sobrepostas, usando-se uma aproximação linear para os restantes documentos. O processo de fusão será apresentado com detalhe mais adiante.
Este processo é repetido até obter um único conjunto de resultados ou vários conjunto disjuntos de resultados. Neste último caso, utiliza-se um método de intercalação simples ou ponderada.
Sintetizando

1. Detectar os resultados duplicados entre cada par de fontes e representar essa sobreposição numa matriz que é simétrica e tem diagonal zero;

2. Remover os conjuntos de resultados sem sobreposição e colocá-los no conjunto de disjuntos;

3. Ordenar a matriz de sobreposição pela soma das linhas;

4. Tomar o resultado cuja linha tem maior soma e fundir com aquele que constitui a maior parcela nessa soma (identificado pela coluna);

5. Actualizar a matriz de sobreposição;

6. Se a matriz tiver mais que um elemento regressar ao princípio ; senão colocar o resultado no conjunto de disjuntos; e,

7. Interacalar os resultados disjuntos.

Fusão de dois conjuntos de resultados

Para exemplificar, suponha que se dispõe de 2 BDTs, cujas respostas à mesma interrogação X_1 e X_2 têm dois documentos sobrepostos D_1 e D_2. Seja f_1 e f_2 as escalas de pontuações

Lógica	Máximo	Mínimo
CombSUM	$Max(f_1) + Max(f_2)$	$Min(f_1, f_2)$
CombMAX	$Max(f_1, f_2)$	$Min(f_1, f_2)$
CombMIN	$Max(f_1, f_2)$	$Min(f_1, f_2)$
CombMED	$\frac{Max(f_1)+Max(f_2)}{2}$	$Min(f_1, f_2)$

Tabela 23: Cálculo do máximo e mínimo de f para várias lógicas

atribuída aos documentos em X_1 e X_2, respectivamente. Seja f a pontuação atribuída no conjunto $X = X_1 \cup X_2$, resultado da fusão.

Suponha que a função f_i devolve resultados entre Min_i e Max_i.

O primeiro passo é combinar os valores máximos e mínimos parciais para calcular os limites da função f. Os limites de f (Min, Max), são calculados de acordo com a lógica de fusão utilizada. Observe-se que se parte do princípio que se utiliza a lógica para os elementos não duplicados, se se tratarem de extremos (ver Tabela 23). Essa heurística permite-nos determinar um intervalo possível dos resultados da fusão.

Para os documentos sobrepostos $f(D_i)$ é calculada usando uma das lógicas de fusão usadas para as BDT sobrepostas descritas na secção anterior correspondente. f pode ser o mínimo, a média, o máximo, etc...

A pontuação em X dos restantes documentos (os não sobrepostos) em X_1 e X_2 vai ser obtida usando uma aproximação linear. Vai-se apresentar aqui a formula de transformação para os documentos de X_1.Para X_2 será precisamente similar.

$$f(u) = \begin{cases} \frac{f(D_1)-NewMin}{f_1(D_1)-Min_1} * (f_1(u) - Min_1) + NewMin, \\ \qquad\qquad\qquad\qquad \text{se } Min_1 \leq f_1(u) < f_1(D_1) \\ \frac{f(D_2)-f(D_1)}{f_1(D_2)-f_1(D_1)} * (f_1(u) - f_1(D_1)) + f(D_1), \\ \qquad\qquad\qquad\qquad \text{se } f_1(D_1) \leq f_1(u) \leq f_1(D_2) \\ \frac{NewMax-f(D_2)}{Max_1-f_1(D_2)} * (f_1(u) - f_1(D_2)) + f(D_2), \\ \qquad\qquad\qquad\qquad \text{se } f_1(D_2) < f_1(u) \leq Max_1 \end{cases}$$

As fórmulas de transformação podem ser generalizadas com facilidade para o caso de m documentos sobrepostos.

Os novos máximos e mínimos ($NewMax$ e $NewMin$ na fórmula) são calculados usando a tabela 23.

fonte 1		fonte 2		fonte 3		fonte 4	
A	0,170	F	0,239	I	0,251	L	0,331
B	0,161	G	0,234	J	0,245	A	0,153
C	0,141	H	0,229	A	0,144	B	0,145
D	0,114	A	0,147	B	0,131	C	0,134
E	0,113	C	0,131	K	0,119	K	0,118

Tabela 24: Pontuação original de conjunto de resultados para fusão

Há situações especiais em que um documento tem precisamente a mesma pontuação, no conjunto de partida do que outro para o qual existe sobreposição. Para usar essa referência com o mesmo valor basta aumentar ou diminuir ligeiramente a pontuação desse documento.

Um exemplo

Imagine-se que ao interrogar quatro fontes diferentes, são devolvidos os conjuntos de resultados apresentados na Tabelas 24 e 25. Os documentos estão identificados por letras e dispõe-se, para além disso, da pontuação atribuída a cada documento pelos motores de busca remotos (ver Tabela 24) ou caso esta não esteja disponível uma pontuação derivada da posição (ver Tabela 25).

Comeca-se por apresentar na Figura 23 uma visualização gráfica da posição dos documentos em cada conjunto de resultados. O documento A aparece nos quatro resultados (na 1ª, 2ª, 3ªe 4ªposições). O documento B aparece em três dos resultados (na a 2ª, 3ªe 4ªposições). O documento C aparece também em 3 dos resultados (3ª, 4ªe 5ªposições) enquanto o documento K aparece duas vezes na quinta e última posição.

O resultado da fusão, usando um algoritmo convencional e como dados de entrada quer a pontuação original quer uma pontuação derivada da posição é apresentado nas Tabelas 26 e 27. Como seria de esperar, a pontuação final tem um nível maior de descriminação usando a pontuação original do que quando se usa apenas a ordem, onde os documentos nos extremos (início e fim) têm todos a mesma pontuação.

Segue-se a apresentação da fusão usando o algoritmo proposto. A ideia é comecar por fundir os resultados com maior sobreposição. No exemplo apresentado, as fontes 1 e 4 são as primeiras. Posteriormente funde-se resultado com a fonte 3 e seguidamente a fonte 2. Os resultados da fusão (usando as várias lógicas e para os valores de pontuação de entrada)

fonte 1		fonte 2		fonte 3		fonte 4	
A	1	F	1	I	1	L	1
B	0,8	G	0,8	J	0,8	A	0,8
C	0,6	H	0,6	A	0,6	B	0,6
D	0,4	A	0,4	B	0,4	C	0,4
E	0,2	C	0,2	K	0,2	K	0,

Tabela 25: Pontuação deduzida da ordem de conjunto de resultados para fusão

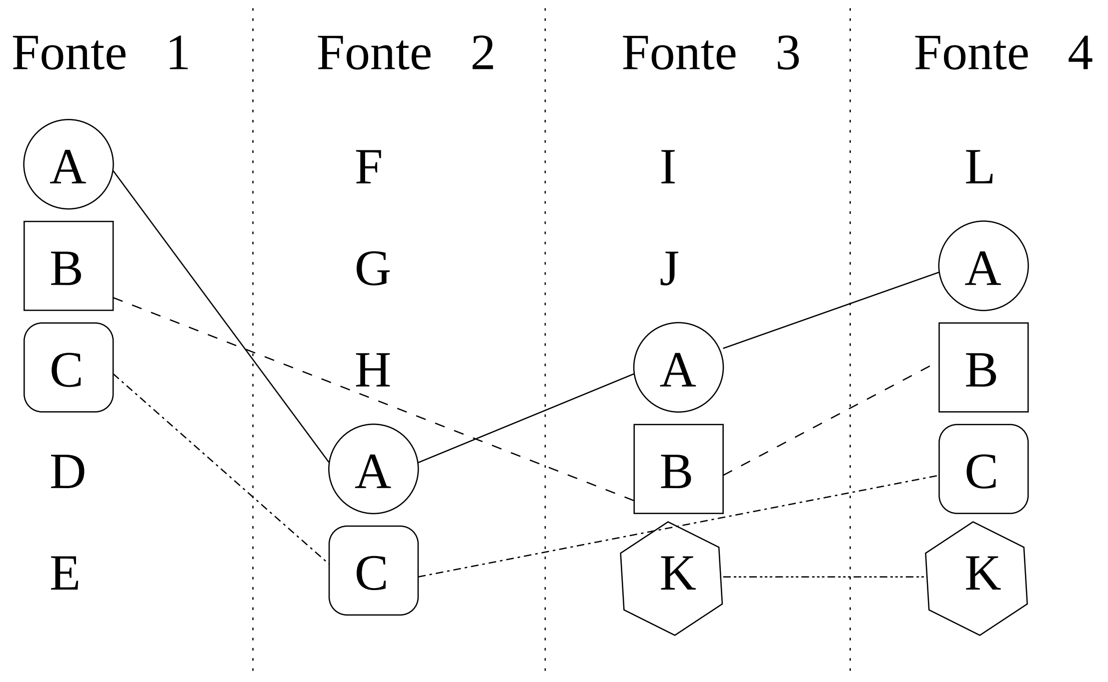


Figura 23: Diferença de posição dos mesmos documentos em fontes diferentes

máximo		média		mínimo		soma	
L	0,331	L	0,331	L	0,331	A	0,614
I	0,251	I	0,251	I	0,251	B	0,437
J	0,245	J	0,245	J	0,245	C	0,406
F	0,239	F	0,239	F	0,239	L	0,331
G	0,234	G	0,334	G	0,334	I	0,251
H	0,229	H	0,229	H	0,229	J	0,245
A	0,170	A	0,154	A	0,144	F	0,239
B	0,161	B	0,146	B	0,131	K	0,237
C	0,141	C	0,135	C	0,131	G	0,234
K	0,119	K	0,119	K	0,119	H	0,229
D	0,114	D	0,114	D	0,114	D	0,114
E	0,113	E	0,113	E	0,113	E	0,113

Tabela 26: Pontuação final usando algoritmo convencional para as diferentes lógicas e a pontuação original

máximo		média		mínimo		soma	
A	1	F	1	F	1	A	2,8
F	1	I	1	I	1	B	1,8
I	1	L	1	L	1	C	1,2
L	1	G	0,8	G	0,8	F	1
B	0,8	J	0,8	J	0,8	I	1
G	0,8	A	0,7	H	0,6	L	1
J	0,8	B	0,6	A	0,4	G	0,8
C	0,6	H	0,6	B	0,4	J	0,8
H	0,6	D	0,4	D	0,4	H	0,6
D	0,4	C	0,4	C	0,2	D	0,4
E	0,2	E	0,2	E	0,2	K	0,4
K	0,2	K	0,2	K	0,2	E	0,2

Tabela 27: Pontuação final usando algoritmo convencional para as diferentes lógicas e a pontuação deduzida da ordem

máximo		média		mínimo		soma	
I	0,501	I	0,251	I	0,501	F	0,991
F	0,501	L	0,25	H	0,501	I	0,752
L	0,5	G	0,246	F	0,501	H	0,752
J	0,484	J	0,2459	L	0,5	L	0,751
G	0,483	F	0,245	J	0,4825	J	0,7453
H	0,465	H	0,241	G	0,4815	G	0,7452
A	0,17	A	0,159	A	0,143	A	0,627
B	0,161	B	0,145	B	0,131	B	0,596
C	0,1415	C	0,136	C	0,1248	C	0,545
D	0,1192	D	0,1161	D	0,1161	D	0,239
E	0,1191	E	0,116	E	0,116	E	0,2321
K	0,119	K	0,113	K	0,113	K	0,232

Tabela 28: Pontuação final usando algoritmo proposto para as diferentes lógicas e a pontuação original

são apresentado nas Tabela 28. Fundiu-se primeiramente os resultados da fonte 1 com os da fonte 4, depois os da fonte 3 e finalmente os da fonte 2.

Como seria de esperar, o algoritmo proposto promove bastante mais os documentos F, G, H, I, J e L que o algoritmo convencional pelo facto de ocorrerem acima do documento sobrepostos A,B e C. Pelo facto de ocorrerem acima dos documentos replicados que são usados como referência pelo algoritmo, estes documentos aparecem em todos os casos nos primeiros lugares com ligeiras mudanças de posição. Pela mesma razão, os documentos D,E e K aparecem sempre nos últimos 3 lugares. Como se pode observar, com a utilização deste exemplo, os documentos replicados têm um papel chave na pontuação final dos restantes documentos.

5.4.2 Arquitectura do Fusor

A arquitectura do fusor pode ser visualizada na Figura 24. O fusor tem cinco componentes básicos: o normalizador de interrogações, o normalizador de pontuações, o detector de duplicados, o intercalador de respostas e o reavaliador de documentos.

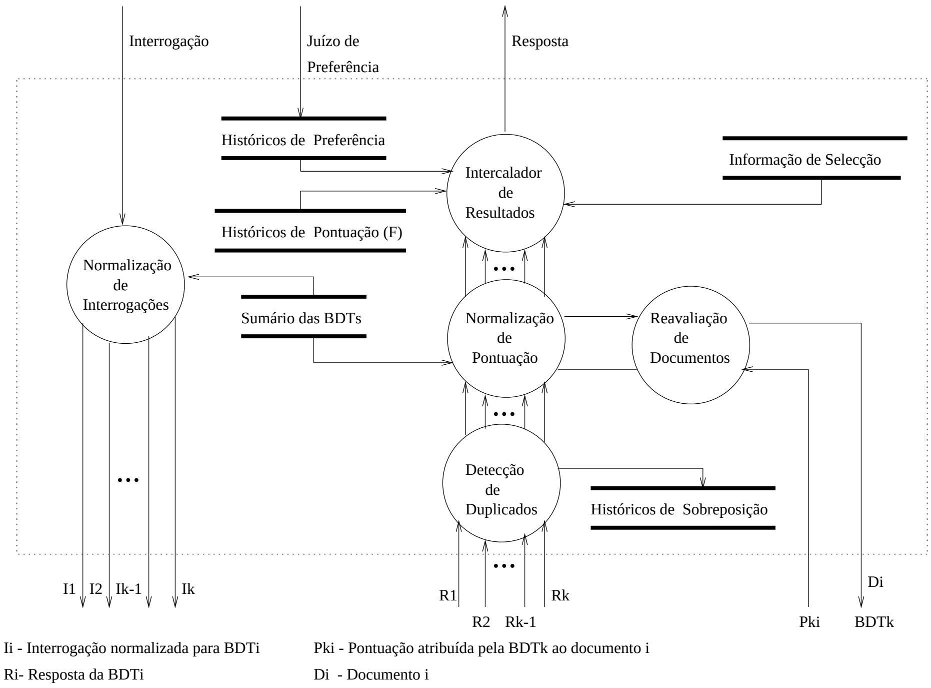


Figura 24: Arquitectura do Fusor

Normalizador de Interrogações

A interrogação vinda do utilizador é normalizada por forma a incorporar informação da estatística da colecção global, no sentido da potência da descriminação dos termos (quantificada pelo *idf*) ser relativa a toda a informação e não a cada BDT em particular. A normalização dos termos é feita com base na informação disponibilizada pelos sumários das várias BDTs. Dessa forma a interrogação enviada para a BDT_i (I_i) é modificada pela relação entre a estatística de cada termo na BDT_i e na colecção global.
Sintetizando:

Dada a interrogação $I = (i_1, i_2, ..., im)$ em que i_j corresponde ao termo t_j.
Para cada BDT_k para $k = 1, ...N$, o objectivo é obter a versão da interrogação normalizada associada I_k.
Para cada termo t_j da interrogação obtemos cada componente de I_k multiplicando o correspondente valor pela relação $\frac{idf_{kj}}{idf_j}$. Uma vez normalizada I_k é enviada para a BDT_k.

Detector de Duplicados

Devolvidas as respostas pelas diferentes BDTs, há um detector de duplicados que identifica os itens de resposta (documentos) que aparecem em mais do que uma resposta. Essa informação é registada num histórico de informação de sobreposição entre diferentes BDTs.

Admita-se que cada BDT_k devolve o respectivo conjunto de resultados $R_k = (r_{1k}, r_{2k}, ..., r_{mk})$.
Cada r_{ik} constitui um item da resposta correspondendo a um documento.
O conjunto de itens duplicados nos dois conjuntos de respostas R_k e R_ji corresponde à intersecção $R_k \cap R_j$, detectada com base na comparação dos campos de meta-informação MD5, URL ou título e autor.
O número de duplicados entre cada par de BDTs deve ser registado num histórico de sobreposições.
Este módulo deve devolver também a matriz de sobreposição dois a dois.

Normalizador de pontuações

Após o detector de duplicados, entra em jogo um normalizador de pontuações (ou número de ordem) atribuída aos documentos para posterior fusão. Caso se justifique, o normalizador usa os serviços de um reavaliador de documentos para obter pontuações de alguns documentos nas várias BDT envolvidas. O normalizador pode necessitar de usar informação estatística das colecções disponibilizadas nos sumários. O normalizador pode eventualmente ter necessidade de traduzir posições de ordem em pontuações.

Para cada item de resposta da BDT_k se estiver disponível a pontuação máxima (p_{max}) e mínima (p_{min}) atribuíveis, calcular o valor normalizado usando a fórmula $p_n = \frac{p-p_{min}}{p_{max}-p_{min}}$. Se não estiverem disponíveis esses valores utilizar os valores máximo e mínimo do conjunto de resultados como aproximação.
Se a pontuação não estiver disponível, determiná-lo com base na fórmula da posição de ordem $p = 1 - \frac{o-1}{n}$ em que o é o posição e n o conjunto de resultados.

Intercalador de respostas

Por fim, as respostas passam por um intercalador de respostas que, usando um determinado algoritmo de fusão, devolve uma resposta com itens seriados e com a respectiva pontuação (se possível) para ser entregue ao utilizador. O algoritmo usado já foi anteriormente descrito. O utilizador pode ou não devolver um julgamento de preferência (JP na Figura 24) que pode ser usado para realimentar o sistema. Este método de realimentação vai ser explorado apenas em trabalho futuro.
Para seu funcionamento, o intercalador de respostas pode usar dados provenientes da selecção das BDTs[7] para ponderar os resultados.

5.5 Resultados experimentais

Para se obter resultados experimentais foi usada a coleção WSJ completa. Embora as várias técnicas de fusão já tenham sido comparadas noutros contextos [BKFS95, Lee97, SC97], a ideia aqui é verificar o desempenho dessas técnicas num ambiente ligeiramente distinto. Assim, vamos usar alternativamente algumas das técnicas de fusão disponibilizadas (6) para comparar o seu desempenho.
Prepararam-se diversas experiências variando o número de BDTs total e o nível de sobreposição entre as BDTs. A variação do nível de sobreposição exige que se preparem diferentes conjuntos de dados. As estatísticas da colecção utilizada estão apresentadas na Tabela 29. Os vários documentos são distribuídos pelos várias BDTs, usando o alocador de documentos, que ja foi apresentado no Capítulo 4. A distribuição dos documentos é feita variando o nível de sobreposição entre as diferentes BDTs já que é a motivação fundamental do nosso

[7] Caso ela seja feita, naturalmente!

Colecção usada	Wall Street Journal
Tamanho da colecção(Mb) (compactada com gzip)	201,032
Número de documentos	172,873
Número de termos	264,276
Número de registos de ocorrência	6 872 095
Número of Bases de Dados	10, 20, 40, 80
Parâmetro de sobreposição na alocação	0.05, 0.15, 0.2, 0.25, 0,3

Tabela 29: Estatísticas do cenário para experiências

estudo. Embora do ponto de vista do alocador se faça depender o número de cópias de cada documento do nível de sobreposição, talvez se justifique, no entanto, variar o número de cópias dos documentos fixando o nível de sobreposição. A única razão para não explorar esta via, é a de restringir o conjunto de experiências a realizar.
Para esse efeito colocou-se o encaminhador de interrogações a funcionar sem selecção, em modo *broadcast.*
O número de documentos pedidos a cada fonte está especificado na expressão seguinte:

$$\widehat{r} = \frac{k}{(1-\Phi_m)*N + \Phi_m * N} \tag{4}$$

Onde k é número de documentos pedidos pelo utilizador e Φ_m é a média da sobreposição de uma BD com todas as restantes. A justificação para essa fórmula é pensar-se que se Φ_m é a média da sobreposição entre BDTs em média uma BDT com N documentos tem $N*(1-\Phi_m)$ documentos únicos e $iN*\Phi_m$ documentos duplicados. De qualquer forma, e mesmo que os documentos ultrapassem o valor k, a resposta só inclui os k documentos melhor pontuados, sendo truncados os documentos a mais.
Esta abordagem é diferente da usada em alguns artigos da literatura em que pede k documentos de cada BD (num total de $k*N$) para devolver os k considerados melhores ao utilizador. Para efeitos de avaliação os resultados são os mesmos desde que se consiga determinar o número de documentos pedidos a cada BDT. De qualquer forma e relativamente à fórmula utilizada, poder-se-ia mesmo usar o nível de sobreposição médio de cada BD e não pedir um igual número de documentos de cada BD.
Paralelamente, foi construída uma base de dados centralizada exactamente com os mesmos

	Central.	Soma		Média		Mínimo		Máximo	
		S	W	S	W	S	W	S	W
P_m	0,2057	0,06990	0,1872	0,1163	0,1937	0,1164	0,1848	0,1157	0,1865
P_r	0,2756	0,1505	0,2518	0,1860	0,2530	0,1866	0,2522	0,1853	0,2464

Tabela 30: Variação da precisão com a lógica de fusão, fixando o número de BDTs (40) e o parâmetro de sobreposição (0.15)

documentos para servir de referência. A extração da informação dos documentos foi feita com radicalização e remoção de palavras comuns.

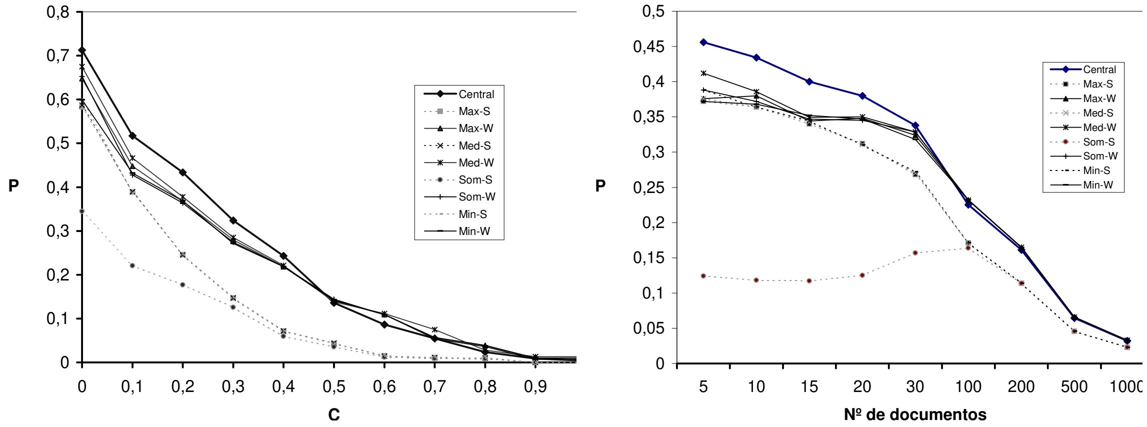
5.5.1 Compararação das lógicas de fusão

Começou-se por variar as lógicas de fusão usando dois algoritmos. Por um lado o método de fusão proposto (aqui designado por Webt (W)) e por outro um método mais convencional (aqui designado por Simples (S)) em que não se condera se considera o nível de sobreposição. Para limitar as experiências usou-se apenas as lógicas da soma, média, máximo e mínimo. Nesta experiência utilizaram-se 40 bases de dados e como valor do parâmetro de sobreposição 0.15. Este cenário experimental é sucintamente desigando como webt40.15.

Apresenta-se em primeiro lugar a Tabela 30, que inclui a precisão média (P_m) e a precisão após serem devolvidos o número de documentos relevantes existentes (P_r). Como referência é usada a BDT centralizada . Os valores de P_m e P_r para o algoritmo proposto aproximam-se bastante mais dos valores da BDT centralizada que o algoritmo convencional. Os ganhos de precisão são bastante significativos em qualquer das lógicas. Se tomarmos como referência o valor de precisão média da BDT centralizada temos melhorias que vão desde 58% a um mínimo de 33%.

Na Figura 25 apresentam-se os gráficos que relacionam a precisão, a cobertura e o número de documentos para as várias lógicas e para os dois métodos de fusão, o convencional e o proposto. De um modo geral, estes gráficos confirmam a superioridade do algoritmo proposto relativamente ao convencional e essa superioridade é mais acentuada nos valores de menor cobertura e quando se pede ao sistema menor número de documentos. A partir dos 100 documentos há menos diferenças na eficácia, obtendo-se mesmo valores bastante aproximados aos da abordagem centralizada.

Observe-se em particular o facto da lógica da soma usando o algoritmo convencional ser



(a) Precisão e Cobertura (b) Precisão e Numero de documentos

Figura 25: Variação da lógica de fusão, fixando o número de BDTs (40) e o parâmetro de sobreposição (0.15)

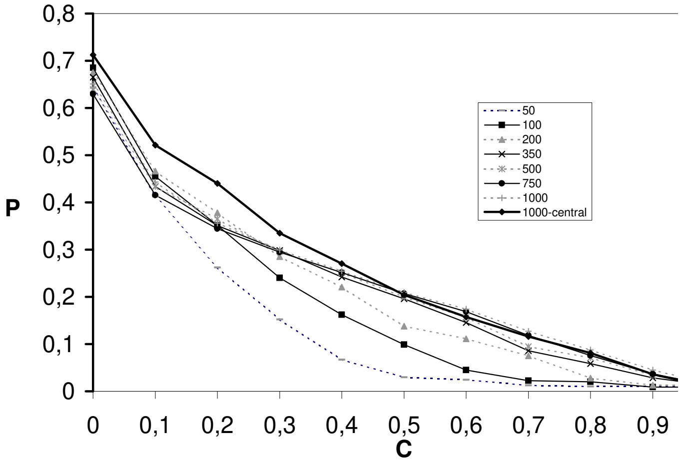
bastante penalizada relativamente ao que é usual. A explicação desse facto é, em nosso entender, não fazer sentido neste contexto que os documentos replicados sejam favorecidos relativamente aos restantes porque o facto de aparecerem nos vários conjuntos de resultados significa apenas que estão duplicados e não que são mais relevantes que os outros.

5.5.2 Variação do número de documentos

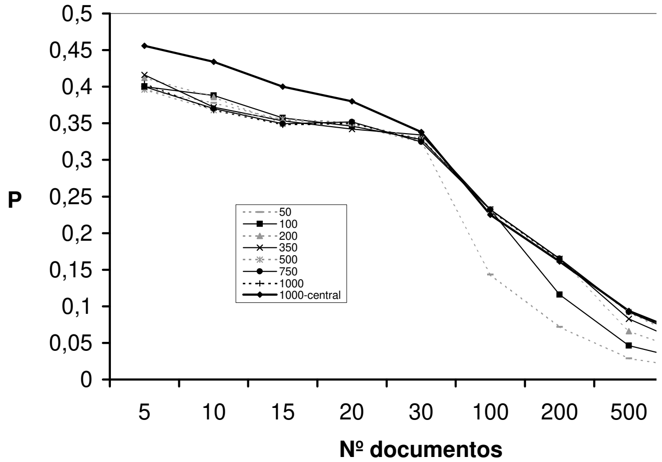
O passo seguinte (usando o cenário webt40.15) foi variar o número de documentos pedidos pelo utilizador (k) . Os valores usados para k foram 50, 100, 200, 350, 500, 750 e 1000.

Um primeiro resultado que é apresentado na Figura 26 é o número de documentos relevantes devolvidos no total das interrogações tanto na estratégia distribuída (usando o algoritmo proposto) como centralizada. Como se pode observar a quantidade de documentos relevantes apresentados na resposta são praticamente os mesmos. E supreendentemente, pelo menos para este cenário experimental, os relevantes devolvidos na estratégia distribuída são ligeiramente superiores ao da estratégia centralizada.

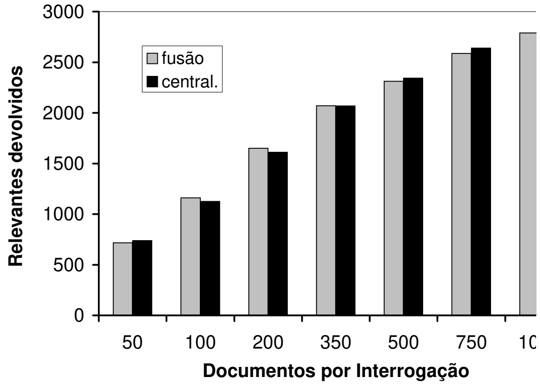
Embora os documentos relevantes sejam praticamente os mesmos, a abordagem centralizada apresenta uma maior eficácia que a distribuída. De qualquer forma e como pode ser visualizado nos gráficos da Figura 26 , a eficácia não se altera mais do que seria de esperar com



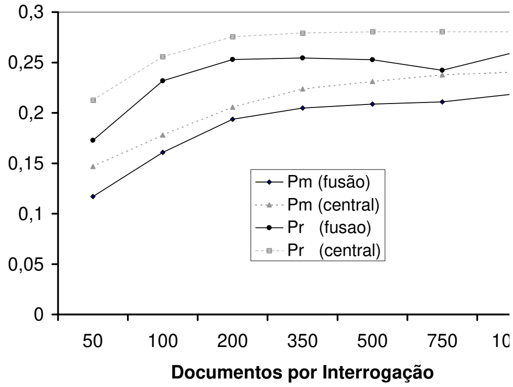
(a) Precisão e cobertura



(b) Precisão e número de documentos



(c) Relevantes devolvidos



(d) Precisão média e com total relevantes

Figura 26: Variação do número documentos pedidos pelo utilizador mantendo fixo o número de BDs (40) e o parâmetro de sobreposição (0.15) a variação dos documentos pedidos, isto é apenas quando se chega a parte correspondente ao número de documentos pedidos. Isto é realçado nas curvas de precisão média ou para o número de documentos correspondentes ao total de relevantes em que há uma melhoria com o número de documentos pedidos (ver na Figura 26).

A referência centralizada utilizada foi a obtida para 1000 documentos embora se tenham também feito as experiências os valores para os restantes valores de k. Os valores de precisão e cobertura são sempre superiores, com o comportamento relativo idêntico (para os vários valores de k) aos da abordagem distribuída.

5.5.3 Variação com o nível de sobreposição e número de BDTs

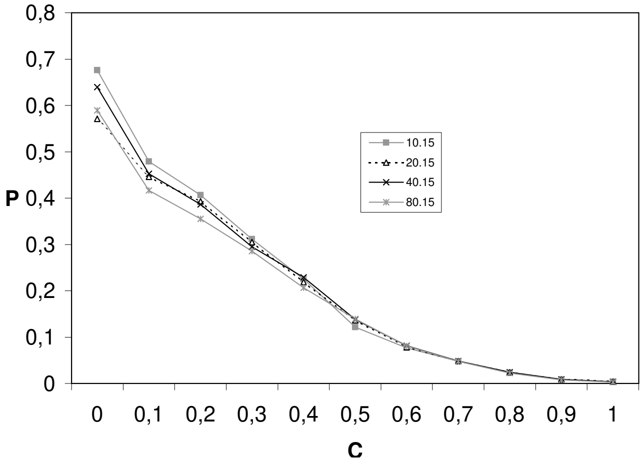
Para um mesmo nível de sobreposição, quando se aumenta a quantidade de BDTs diminui a sobreposição entre cada par de BDTs. Esta situação faz supor que exista uma quantidade menor de documentos sobrepostos entre cada par de resultados, reduzindo as respectivas referências e comparabilidade. Por este facto espera-se que nestas experiências o algoritmo proposto diminua a sua eficácia com aumento de BDTs num mesmo nível de sobreposição. Ao mesmo tempo, quando se mantém o número de BTDs, a eficácia deve melhorar com o aumento do nível de sobreposição.

Vejamos se estas suposições se confirmam nos resultados experimentais. Comecemos por analisar a Tabela 31. Vamos para já ignorar os valores para níveis de sobreposição superiores a 15%. Nota-se uma tendência dos dois tipos de precisão melhorarem com a diminuição do número de BDTs. Existe uma excepção que é o cenário 20.15 que mesmo assim foi corrido duas vezes. A mesma tendência se verifica quando se mantém o número de BDTs e aumenta o o nível de sobreposição. Não são resultados conclusivos relativamente ao que se previa, é necessário um maior número de experiências.

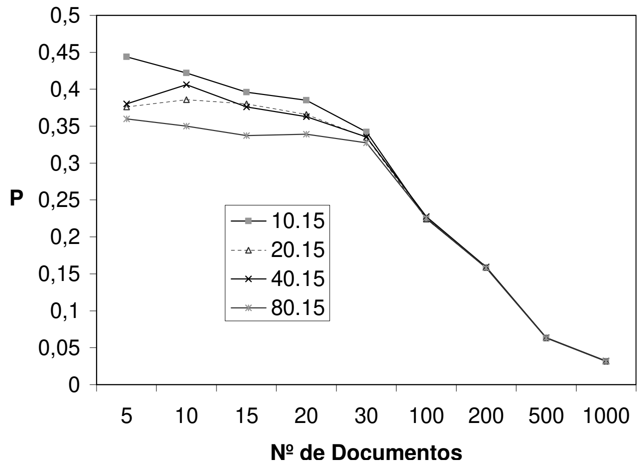
5.6 Discussão

A hipótese colocada neste trabalho confirmou-se plenamente no que se refere à fusão de resultados. A utilização da sobreposição pode melhorar a eficácia dos algoritmos de fusão. No cenário utilizado melhorou-se em 33% a a precisão média das respostas.

Este problema da fusão dos resultados só por si poderia ser objecto de um trabalho mais extenso. Muitos aspectos foram deixados de fora e muitas experiências ficaram por realizar.

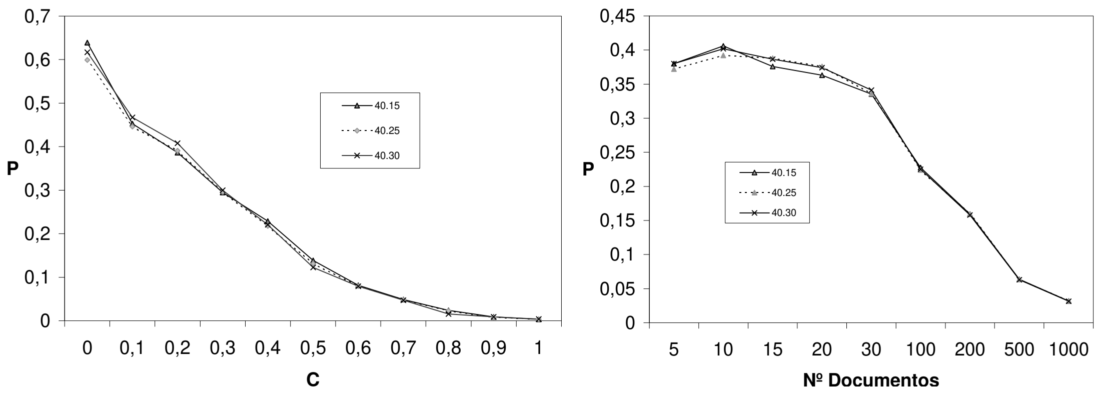


(a) Precisão e cobertura



(b) Precisão e número de documentos

Figura 27: Variação do número de BDs (20,40,80) mantendo fixo o número de de documentos pedidos pelo utilizador e o parâmetro de sobreposição (0.15)



(a) Precisão e cobertura

(b) Precisão e número de documentos

Figura 28: Variação do parâmetro de sobreposição mantendo fixo o número de de documentos pedidos pelo utilizador e o e o número de Bases de Dados (40)

Cenário	P_m	P_r
10.15	0,1952	0,2558
20.15	0,1843	0,2451
40.15	0,1852	0,2518
40.25	0,181	0,2464
40.30	0,1861	0,2589
80.15	0,172	0,2568

Tabela 31: Precisão média (P_m) e precisão ao final dos relevantes (P_r) mantendo fixo o número de de documentos pedidos pelo utilizador, variando o parâmetro de sobreposição e também o número de BDTs.

Embora alguns deles sejam de concretização simples, a correspondente realização das experiências e a análise de resultados prolongariam bastante o tempo necessário para realização deste trabalho.

A normalização do peso das interrogações usando as estatísticas da colecção distribuída é simples de concretizar usando o Webtrieve. Este mecanismo está relacionado ou pode ser usado alternativamente à ponderação da pontuação dos documentos com a pontuação atribuída a respectiva base de dados pelo mecanismo de selecção.

Um outro aspecto importante é a utilização de mecanismos de repontuação do máximo e mínimo de cada conjunto de resultados para aumentar a comparabilidade de diferentes funções de pontuação. A repontuação pode ser importante em situações em que a ausência de documentos sobrepostos pode conduzir à intercalação simples dos resultados.

Também se poderia estudar em que medida a eficácia da fusão varia com o número de fórmulas diferentes usadas pelos diferentes motores para pontuação dos documentos. Para além das fórmulas poderia-se-ia usar modelos de RI subjacentes diferentes.

A utilização das hiperligações entre documentos e a sua influência no algoritmo de fusão é importante é vai ser objecto de trabalho futuro quando se usar as colecções Web da TREC.

Um aspecto que precisa de ser ainda bastante estudado é a utilização de mecanismos de realimentação de preferência na fusão de resultados. Trata-se de trabalho a realizar a mais médio prazo. Este mecanismo consiste em usar simplesmente as preferências do utilizador para refinar o algoritmo de fusão.

Capítulo 6

Selecção de Bases de Dados

Sumário

A selecção de BDTs é uma das funções mais importantes dum sistema distribuído de RI. Se houver replicação descoordenada, algumas BDTs podem alguns documentos da sua resposta sobrepostos com documentos devolvidos por BDTs melhor pontuadas. Isto pode provocar a degradação do desempenho dos algoritmos de selecção. O trabalho apresentado neste capítulo, tem como objectivo principal a utilização da informação de sobreposição na selecção, como forma de evitar essa degradação.

6.1 Introdução

A Internet resulta da interconexão de redes pertencentes a organizações com políticas próprias de utilização de recursos e com pouca coordenação técnica ou administrativa. Por este facto, não surpreende a existência de replicação descoordenada em aplicações distribuídas, nomeadamente em sistemas distribuídos de RI.

No capítulo 5, já se mostrou que a utilização da informação de sobreposição pode, em determinados contextos, melhorar a eficácia dos algoritmos de fusão convencionais. O principal objectivo deste capítulo é estudar os algoritmos de selecção em contextos de sobreposição entre diferentes BDTs.

Se tomarmos como base a informação necessária para a função de selecção, podemos agrupar os algoritmos de selecção em duas classes: uma primeira que precisa por cada termo do total de ocorrências e o número de documentos em que ocorre; uma segunda em que, para além disso, é necessário conhecer a variância da distribuição de ocorrências.

O mecanismo de selecção proposto neste trabalho, adiante designado por WEBT, usa como ponto de partida a exportação de sumários pelas BDTs usada no gGLOSS e GLOSS [GGMT94, GGM95]. A diferença está no facto de se pretender explorar no algoritmo de selecção a aproximação da similaridade dos documentos com a interrogação a uma distribuição normal, quando a interrogação tem muitos termos. Essa aproximação, baseada no teorema do limite central, pode ser crítica quando se diminui o número de termos. Neste último caso, pode-se aumentar *artificialmente* o número de termos usando a expansão automática da interrogação.

Com base no conhecimento dos parâmetros das distribuições para os diferentes termos da interrogação, é possível conhecer a distribuição da similaridade da interrogação com os documentos de cada fonte. A partir daí, e usando uma de várias heurísticas alternativas para estimar a população de documentos com base no número de documentos em que ocorre cada termo (informação também disponível nos sumários), é possível estimar o número esperado de documentos com similaridade superior a um dado valor de corte. O valor de corte para a similaridade escolhido é o maior que for possível para a soma das contribuições das várias fontes ser igual ao número de documentos pedidos pelo utilizador.

O protocolo usado para na obtenção de informação das fontes apresenta, por esse facto, algumas diferenças relativamente ao STARTS [GCGM+97]. Para além da soma das ocorrências, é necessária a soma dos quadrados das ocorrências para ser possível calcular a variância.

Um outro aspecto inovador é o estudo da influência da sobreposição entre as diferentes BDTs na eficácia dos algoritmos de selecção e, por essa via, a sua inclusão como um parâmetro de entrada. O conhecimento do nível de sobreposição com as restantes fontes interrogadas, permite ajustar o número de documentos a pedir de uma determinada fonte. Desta forma, tenta-se prever as dificuldades causadas pelos documentos que aparecem replicados em fontes diferentes.

As métricas para avaliação da selecção tiveram também que ser revistas para poderem ser usadas em ambientes sobrepostos. Trata-se de uma das contribuições desta tese.

Nesse sentido, são redefinidas as métricas de avaliação no sentido da consideração da possibilidade da existência de sobreposição entre as diferentes BDTs.

Antes de apresentar o algoritmo de selecção proposto, define-se sucintamente o problema a resolver pela selecção, usando como modelo subjacente o Modelo do Espaço Vectorial.

O método proposto calcula os parâmetros da distribuição normal para cada BDT, usando como base a informação disponibilizada nos sumários correspondentes. O algoritmo tem duas variantes conforme considere ou não como parâmetro de entrada a sobreposição entre as várias BDTs.
No final do capítulo, são apresentados resultados experimentais, usando métricas propostas na literatura, para avaliação dos algortimos de selecção. Para além dos resultados experimentais com o algoritmo proposto, avalia-se o algoritmo CORI, bem conhecido da literatura, com um factor dependente da sobreposição.

6.2 O problema da selecção de BDTs

De acordo com o modelo vectorial, cada documento ou interrogação é representado por um vector, cuja dimensão é dada pelo dicionário de termos. As posições no vector correspondentes aos termos têm um valor real de acordo com as ocorrências, tendo 0 nos restantes termos do dicionário.
O problema pode ser definido da seguinte forma:
dada uma interrogação $I = [i_1, i_2, .., i_k]$, em que i_j são os termos, que requer resposta com n documentos,

- Seleccionar l BDTs de um total de N, contendo os n documentos mais relevantes
- Calcular o nºde documentos, r_i, a devolver por cada BDT

$$\vec{r} = [r_1, r_2, ..., r_l] \qquad r_i > 0 \qquad \sum_{i=0}^{l} r_i = n$$

Corresponde a maximizar a função de similaridade

$$S_t(n) = \max_{\vec{r}} \sum_{i=1}^{l} S_i(r_i)$$

Onde
$S_t(n)$ é a soma da similaridade com a interrogação dos n documentos devolvidos
$S_i(r_i)$ é a soma da similaridade com a interrogação dos r_i documentos devolvidos pela BDT_i
Observe-se que não se torna necessário conhecer exactamente o valor das funções $S_i(r_i)$ mas apenas a sua relação de ordem.
Também é importante explicitar algumas propriedades da funções $S_i(r_i)$

1. A similaridade para um conjunto de 0 documentos é sempre 0

$$S_i(0) = 0$$

2. A similaridade do conjunto de documentos devolvidos pela BDT_i é a soma das similaridades dos documentos individuais com a interrogação

$$S_i(r_i) = \sum_{k=1}^{r_i} s_i(I, D_{ik})$$

3. São funções crescentes: sempre que se acrescenta um documento ao resultado a função de similaridade no pior dos casos fica igual, embora seja suposto aumentar

$$S_i(r_i + 1) \geq S_i(r_i)$$

4. Suponhamos que r_i e r_j são respectivamente o número de documentos que devem ser pedidos à BDT_i e BDT_j. Se se diminuir um nºde documentos u numa das BD e se aumentar noutra a soma das funções de similaridade não aumenta. Suponha um valor qualquer inteiro $0 < u < r_i$, $i, j = 1, ..., l$, $i \neq j$

$$S_i(r_i) + S_j(r_j) \geq S_i(r_i - u) + S_j(r_j + u)$$

A função de similaridade de cada documento D da BDT_i com a interrogação I pode ser explicitado pela seguinte expressão.

$$s_i(I, D) = \sum_{j=1}^{k} \Phi_i(i_j, idf_j, d_j)$$

Relativamente à função de similaridade das BDTs, depende dos seguintes parâmetros:

i_j	peso do termo j na interrogação
idf_j	inverso do número de documentos em que o termo j ocorre
Φ_i	função de pontuação da BDT_i
d_j	peso do termo j no documento

6.3 Utilização da meta-informação

Associada a cada documento, para além do seu conteúdo, existe informação que o descreve como título, autor, datas, formato, localização na rede, língua, etc... Essa descrição é normalmente designada como *meta-informação do documento.*

Alguma dessa informação, como por exemplo título e autor é indexada pelo que a selecção das BDTs já a considera. Eventualmente, ao serem incluídos no conjunto dos termos indexáveis a sua frequência pode ser inflancionada de acordo com a importância.

Há outra informação que embora não faça sentido indexar (datas, língua, localização geográfica, entre outras), pode ser usada, única e simplesmente, para limitar o espaço de pesquisa. Isto significa limitar o conjunto de BDTs seleccionáveis e, eventualmente, o número de documentos a pedir a cada BDT.

Para além da matriz de frequências, pode existir uma matriz de atributos de meta-informação onde para cada documento é mantida informação como tamanho, datas, formato, domínio de origem, etc...

O problema novo que se coloca no caso distribuído é como sintetizar a meta-informação existente em cada servidor e usá-la conjuntamente com os termos no encaminhamento das interrogações.

Relativamente à meta-informação, a forma que é usada para sintetizar os valores que tomam os vários atributos é a utilização dos *centroids*, isto é manter para cada atributo o conjunto de valores e também os respectivo número de ocorrências [RMF97].

Esta informação pode ser mantida no encaminhador de interrogações, separadamente para cada servidor, por forma restringir os servidores a seleccionar no caso da interrogação conter atributos de meta-informação que sejam obrigatórios.

O número de ocorrências também pode ser usado para ajustar o número de documentos esperado de cada um dos servidores.

Um exemplo pode ser manter a síntese das datas de documentos duma BDT de acordo com categorias pré-definidas, como pode ser observado na tabela 32.

Se o utilizador quiser apenas documentos de 1998, à BDT_1 não poderam ser pedidos mais que 7.

Data	BDT_1	BDT_2	BDT_3	....	BDT_n
$<$ 1985	100	50	20	....	30
1985-1990	30	50	10	....	50
1991-1995	55	48	23	...	74
1996-1997	4	23	39	...	90
1998	7	8	3	...	75
$>$ 1998	200	300	90	...	88

Tabela 32: Número de ocorrências de datas mantidas por categoria

6.4 Métricas para avaliação da selecção

Antes da apresentação dos resultados experimentais, é preciso explicar as métricas que são usadas para avaliação dos algoritmos de selecção [CPV98, CPC+99, GGM95].

A primeira componente é a referência de base que é usada para comparação com a estimação calculada pelos algoritmos proposto. A referência de base define uma figura de mérito que vai ser usada para atestar a qualidade da estimativa conseguida com o algoritmo de selecção.

A primeira referência é a seriação baseada na relevância, cujo acrónimo em inglês é RBR[1]. Esta seriação é obtida usando os julgamentos de relevância disponibilizados pelo TREC. Neste caso as BDTs são ordenadas por ordem decrescente do número total de documentos relevantes que contêm para a interrogação.

Uma outra referência, também utilizada, é a seriação das BDTs pelo seu tamanho, cujo acrónimo em inglês é SBR[2].

Como pretendemos avaliar os algoritmos em ambientes onde pode existir sobreposição, vamos derivar duas referências para as adequar a esses ambientes.

A referência ORBR[3] é a seriação das BDs por ordem decrescente de documentos relevantes não duplicados em BDs mais pontuadas. Isto é um documento relevante só é considerado para a BD_k se não tiver já sido contabilizado numa $BD_i, i < k$.

A referência OSBR[4] é a seriação das BDs por ordem decrescente do número de documentos não replicados em BDs melhor pontuadas. Isto é um documento só é contabilizado para a BD_k se não existir numa $BD_i, i < k$.

---

[1] Relevance Based Ranking
[2] Size Based Ranking
[3] Overlapping Relevance Based Ranking
[4] Overlapping Size Ranking

Observe-se que tanto a ORBR como a OSBR se reduzem nas referências iniciais para ambientes com BDTs disjuntas.

Dada uma seriação de base, B e uma seriação estimada E, teremos um determinado mérito para cada BD usando a seriação estimada (E_i) e usando a seriação de base (B_i). O mérito será definido de acordo com a semântia definida para a referência de base.

Usando essas duas seriações são definidas métricas análogas às tradicionais precisão e cobertura para avaliação em RI [GGM95].

A cobertura (C_n), usando como referência RBR, é o total de documentos relevantes existentes nas n BDs com E_i mais elevado a dividir pelo número documentos das n BDs melhores classificadas usando o RBR.

$$C_n = \frac{\sum_{i=1}^{i=n} E_i}{\sum_{i=1}^{i=n} B_i}$$

A cobertura alternativa $(\overline{C}_n)$ usa como denominador a soma de todos os documentos relevantes existentes nas BDs. n_* é o valor de n a partir do qual $B_i = 0$.

$$\overline{C}_n = \frac{\sum_{i=1}^{i=n} E_i}{\sum_{i=1}^{i=n_*} B_i}$$

Da mesma forma a precisão é obtida com a fracção de BDs nas primeiras n seleccionadas pelo algoritmo $Top_n(E_i)$ em que o mérito seja diferente de 0. Se a a referência for RBR, será o número de BDs em que o número de documentos relevantes seja diferente de 0.

$$P_n = \frac{|bd_i, i < n, merito(db_i) \neq 0|}{|Top_n(E_i)|}$$

As várias métricas podem ser usadas tomando como referência ORBR, e é o que será feito nos resultados experimentais.

Para concretizar as diferença entre o RBR e o ORBR, calculou-se o valor óptimo de cobertura alternativa usando essas duas referências. Como se pode observar na Figura 29, pelo facto de existirem documentos duplicados não se obtém os mesmos valores, sendo as medidas RBR prejudicadas pelo facto de se considerar como denominador um número de documentos superior aos realmente relevantes.

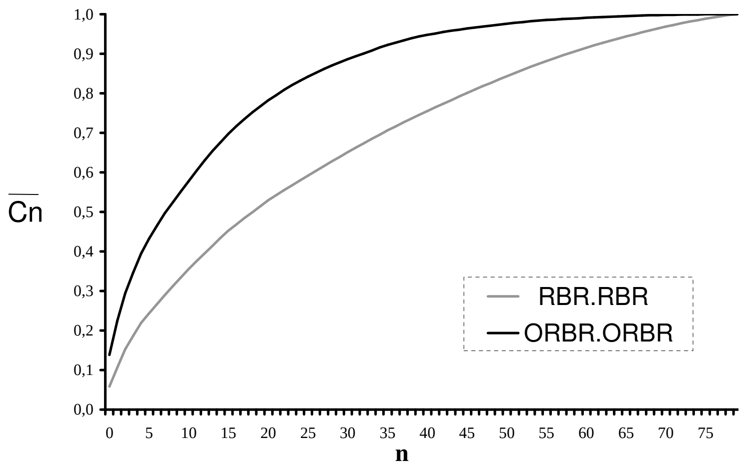


Figura 29: Comparação dos valores de cobertura alternativa óptima usando como referência o RBR e o ORBR para o cenario webt80.15.

6.5 Utilização da informação de sobreposição

Tipicamente, quando se aborda o problema da selecção de BDTs, supõe-se que se está na presença de BDTs disjuntas, ou seja sem qualquer tipo de sobreposição. Como foi referido anteriormente, um dos objectivos deste trabalho é estudar a influência da sobreposição no mecanismo de selecção de BDTs para uma determinada interrogação. Analisemos para já os seguintes cenários:

- *Cenário A*

  Se duas BTDs, por exemplo X_1 e X_2, são idênticas, isto é contêm precisamente os mesmos documentos cumpre-se que

  $$O(X_1/X2) = O(X_2/X_1) = 1.0$$

  Nesse caso, pode-se pensar que apenas uma delas deve ser considerada para selecção já que é suposto responderem precisamente com os mesmos documentos.

  Um pressuposto na afirmação anterior será a existência duma mesma estratégia de pesquisa pois noutra situação podem devolver respostas diferentes à mesma interrogação. Nesse caso, pode-se justificar interrogar as duas com objectivos de melhorar a eficácia da resposta.

A escolha da BDT a interrogar pode ser determinado por um outro critério que não o seu conteúdo. Pode se usar o tempo de resposta, por exemplo.

- *Cenário B*

  Se determinada BDT, por exemplo X_1, inclui completamente uma segunda (X_2), apenas X_1 deve ser considerada para interrogação já que poderá contribuir com uma resposta mais rica. Neste caso X_2 só deverá ser incluída se a X_1 estiver indisponível.

  Considera-se, obviamente, a mesma simplificação que no item anterior.

- *Cenário C*

  Pode acontecer que a BDT X_1 esteja completamente sobreposta não numa mas num subconjunto das restantes. Neste caso, só se pode excluir X1 se as BDTs que a incluiem fizerem parte do conjunto das seleccionadas e tiverem maior pontuação que X_1.

  Mais uma vez se considera a utilização da mesma estratégia de busca.

- *Cenário D*

  Caso uma determinada BDT tenha uma determinada sobreposição com as restantes, a sua pontuação em face duma interrogação deve ser penalizada por esse facto.

  Um método simples de penalização pode ser afectar a população de potenciais documentos relevantes com o número de possíveis sobreposições com as BDTs com pontuação superior.

$$\widehat{R} = \widehat{R}_{orig}(1 - O)$$

  Observe-se que se a BDT estiver totalmente sobreposta, isto é $O = 1$ os documentos úteis são 0. Uma outra observação importante é que para poder contribuir com $\widehat{R}$ documentos úteis a BDT tem que devolver na mesma $\widehat{R}_{orig}$ documentos.

Para que seja possível considerar a sobreposição, pressupõe-se a existência duma matriz de sobreposição estática entre os pares de BDTs ou de um método que permita estimar a sobreposição dinâmica entre as respostas das várias BDTs às interrogações.

Na primeira experiência realizada, utilizou-se o cenário webt10.15 para o qual se dispõe dos valores de sobreposição por interrogação e entre as diferentes BDs (um-a-um e um-para-todos) cujas tabelas foram apresentada no capítulo 4.

O que se pretende verificar é em que medida se melhora a selecção das interrogações, afectando a seriação das BDTs com a informação de sobreposição.

Suponhamos que a pontuação da BDT_j para a interrogação I_k é dada por P_{kj}. Em que medida se pode conseguir uma pontuação $\overline{P_{kj}} = P_{kj} * \alpha_{kj}$, em que α_{kj} é um factor que dependa da sobreposição estática O_{kj} da BDT_j com as restantes BDTs e da posição de ordem da BDT_j na seriação das BDTs para I_k com maior eficácia?

Suponhamos que o selector calcula para a BDT_k o valor P_{kj}, isto é a pontuação atribuída independentemente da sobreposição. Suponhamos para facilitar que dispõe inclusivé do valor exacto do número de documentos relevantes existentes para cada interrogação que é usada no RBR.

Em que medida se obtém uma melhor seriação, afectando essa pontuação óptima com a informação de sobreposição? Isto significa obter uma melhor correlação com o valor óptimo considerando a sobreposição, isto é com o ORBR.

Nesta experiência considerou-se como sendo

$$\alpha_{kj} = (1 - \frac{R_{kj} - 1}{N - 1} * O_j)$$

Em que

R_{jk}	é a posição de ordem da BDT_j para I_k
	(neste caso um valor de 1 a 10)
N	é o número de BDTs (10)
O_k	sobreposição estática (valor de 0.0 a 1.0)

Trata-se duma fórmula linear de penalização da pontuação da BDT que é proporcional ao nível de sobreposição.

Para esta experiência, utilizaram-se as interrogações geradas automáticamente para os tópicos TREC 151 a 200.

Obteve-se para cada uma das 10 BDTs o número de documentos relevantes para cada interrogação e (RBR) e também o número de documentos relevantes não sobrepostos com BDTs melhor pontuadas (ORBR). Usando os valores correspondentes de sobreposição, obtiveram-se os valores $\overline{P_{kj}}$ usando para valor de P_{kj} o RBR respectivo e como R_{jk} a seriação obtida com base no RBR. Os valores de correlação linear obtida para os quinhentos valores (10

RBR_{kj} versus $ORBR_{kj}$	0,943
$\overline{RBR_{kj}}$ versus $ORBR_{kj}$	0,977

Tabela 33: Correlação entre o ORBR e o RBR simples e com factor de sobreposição $\overline{RBR}$

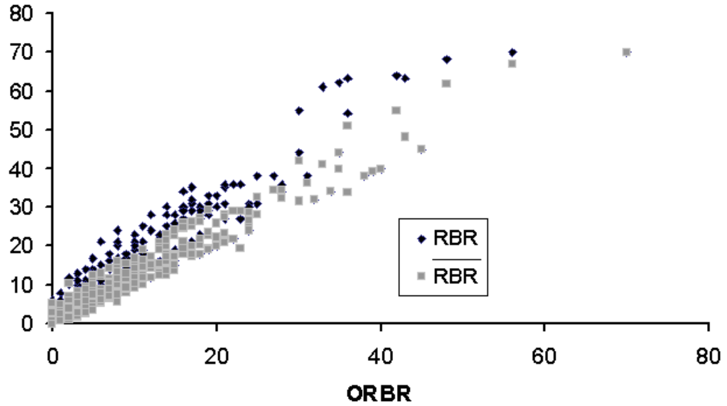


Figura 30: Diagrama de dispersão do ORBR com o RBR afectado e não afectado com a sobreposição

BDTs*50 interrogações) são apresentados na tabela 33.
Os resultados desta tabela, permitem concluir que, neste cenário experimental, a informação de sobreposição estática pode ser usada para melhorar os algoritmos de selecção. Isto apesar de ser uma estimativa muito grosseira da sobreposição efectiva entre as interrogações como se viu no capítulo 4. O diagrama de dispersão correspondente, pode ser observado na figura 30. Como se pode observar os valores de $\overline{RBR}$, são mais próximos da recta $y = x$.
Nas experiências seguintes, o factor de ajustamento α_{ij} para a sobreposição foi melhorado por forma a introduzir a sobreposição máxima com as BDTs melhor pontuadas (como mínimo) e a sobreposição com todas as restantes BDTs (como valor máximo). A fórmula do factor de sobreposição usado é a apresentada em 5.

$$\alpha_{ij} = \begin{cases} 1 & \text{if } R_{ij} = 1 \\ (1 - O_{max(j)} + (O_j - O_{max(j)}) * (\frac{R_{ij}-2}{N-2} * O_j) & \text{if } R_{ij} > 1 \end{cases} \quad (5)$$

Soma da ocorrências (em todos documentos)	o_{ki}
Soma do quadrado das ocorrências (em todos documentos)	v_{ki}
Número de documentos (em que o termo ocorre)	m_{ki}

Tabela 34: Valores mantidos no sumário relativos ao termo k para a BD_i (s_{ik})

$O_{max(j)} = max\{O(DB_j/DB_k) : R_{ik} < R_{ij}\}$ e O_j é a sobreposição global.
Isto dá o máximo da sobreposição um-para-um com as BDTs melhor pontuadas.

6.6 Abordagem Proposta

Dada uma interrogação I com k termos sendo $k << T$, e sendo T o tamanho do dicionário. Para além disso, há o número n de documentos pretendidos pelo utilizador.
Pretende-se estimar o valor de $\vec{r} = (r_1, r_2, ..., r_l)$ de documentos a pedir dum subconjunto l BDTs dum total de N disponíveis. Parte-se obviamente do princípio que $r_i = 0$ para $i > l$.
Começa-se por especificar a informação disponível para o processo de selecção. Os sumários das BDTs consistem numa matriz TxN, em que T é o tamanho do dicionário de termos e N o total de BDTs disponíveis. Cada elemento da matriz é formado por um conjunto de valores (de dimensão 3) mantidos por termo em cada BDT. Esses valores constituem a síntese feita à lista de ocorrências de cada termo na BDT.
Contrariamente ao que é proposto no GLOSS [GGMT94], propõe-se neste trabalho a exportação tanto da soma de ocorrências e total de documentos em que ocorre como da soma dos quadrados do número de ocorrências. Os valores exportados por cada termo da BDT são apresentados na tabela 34. Para além disso, cada BDT exporta o seu número total de documentos. Exceptuando a soma dos quadrados, todos os restantes valores são especificados no protocolo STARTS [GCGM$^+$97].
Seja T o tamanho do dicionário de termos do Encaminhador de Interrogações, resultante da união dos dicionários de termos das BDT individuais indexadas.

$$\vec{T} = \bigcup_{j=1}^{N} \vec{T_j}$$

Seja X_{ji} o número de ocorrências do termo i em cada documento relevante da BDT_j. Então a similaridade com a interrogação de qualquer documento relevante da BDT_j é dado por

$$S_j = \sum_{i=1}^{m} \alpha_{ji} X_{ji}$$

α_{ji} depende de estatísticas para o termo i na colecção e na interrogação. Para esse efeito, a fórmula usada aqui [Har92a] vai ser

$$\alpha_{ji} = idf_{ji}^2 * i_i$$

com

$$idf_{ji} = \frac{\log N_j}{\log n_{ji}}$$

em que

i_i	peso do termo i na interrogação
N_j	é tamanho da BDT_j
n_{ji}	é o número de documentos da BDT_j em que ocorre o termo i

X_{ji} tem distribuição normal com média μ_{ji} e desvio padrão σ_{ji}. Como S_j é a soma de m variáveis aleatórias pelo teorema do limite central, para m suficientemente grande, tem-se que S_j é uma variável aleatória com distribuição normal com os seguintes parâmetros:

$$\mu_j = \sum_{i=1}^{m} \alpha_{ji} \mu_{ji}$$

$$\sigma_j^2 = \sum_{i=1}^{m} \alpha_{ji} \sigma_{ji}^2$$

Interessa maximizar a similaridade com a interrogação do conjunto r_j de documentos devolvidos pela BDT_j

$$S(r_j) = \sum_{k=1}^{r_j} S_{jk}$$

Pode-se estimar o valor de documentos r_j com o valor $\widehat{r_j}$, calculando a área sobre a curva da distribuição normal para cada BDT. Seja $\widehat{R_j}$ uma estimativa dos documentos relevantes na BDT_j. Seja S_λ um valor de similaridade mínimo para os documentos. O valor r_j pode ser calculado usando a seguinte expressão

$$r_j = P(S > S_\lambda) * R_j$$

O valor $P(S > S_\lambda)$ é a cauda superior da distribuição normal com parâmetros μ_j e σ_j, determinados usando o sumário para a BDT_j.
O cálculo de $P(S > S_\lambda)$ é efectuada usando as expressões 6 e 7, aproximações da curva sobre a distribuição normal retiradas de [AS72].

$$P(x) \leq \begin{cases} P_1(x) = \frac{1}{2} + \frac{1}{2}(1 - e^{\frac{2*x^2}{\Pi}})^{\frac{1}{2}} & x > 0 \\ P_2(x) = 1 - \frac{(4+x^3)^{\frac{1}{2}} - x}{2} * (2 * \Pi)^{-\frac{1}{2}} e^{-\frac{x^2}{2}} & x > 1.4 \end{cases} \tag{6}$$

$$P(x) \geq \begin{cases} P_3(x) = \frac{1}{2} + \frac{1}{2}(1 - e^{\frac{2*x^2}{\Pi}} - \frac{2*(\Pi-3)}{3*\Pi^2} * x^4 * e^{\frac{x^2}{2}})^{\frac{1}{2}} & x > 0 \\ P_4(x) = 1 - \frac{1}{x} * (2 * \Pi)^{-\frac{1}{2}} * e^{\frac{x^2}{2}} & x > 2.2 \end{cases} \tag{7}$$

Para estes efeitos importa calcular $Q(x) = 1 - P(x)$ e vai-se usar apenas $P_2(x), 0 \leq x < 1.65$ e $P_3(x), x > 1.65$, que se verificou usando uma simples folha de cálculo serem a melhor escolha.
Está-se a considerar para simplificar que:

- Todos documentos devolvidos por cada BDT são relevantes; e,
- O número de ocorrências de cada termo na interrogação se distribui normalmente nos documentos devolvidos pela BDT. Isto é uma aproximação que é só é necessário fazer se se usar o método aqui apresentado para seleccionar BDTs para interrogações com poucos termos (m pequeno).

Uma outra aproximação é e forma como se calcula a população de documentos potencialmente relevantes em cada BDT (R_j).
Seja n_{ji} o número de documentos em que ocorre termo i na BDT_j. Tem-se, nesse caso, um vector de ocorrências $\vec{n_j}$ para cada BDT.

$$\vec{n_j} = (n_{j1}, n_{j2}, ..., n_{jm})$$

Pode-se usar várias lógicas para estimar o valor (R_j).

$$\widehat{R_j} = \quad \max \vec{n_j} \quad | \quad \min \vec{n_j} \quad | \quad \frac{1}{m} * \sum_i^m n_{ji} \quad | \quad \sum_i^m n_{ji}$$

Cada uma das lógicas parte de determinados pressupostos para a correlação dos termos nos documentos devolvidos.

A lógica da soma, parte do princípio que os termos ocorrem em documentos diferentes. A do máximo e a do mínimo partem do princípio que os termos são fortemente correlacionados, pressupondo entretanto que os documentos relevantes são aqueles em que ocorre pelos menos um termo (MAX) ou aqueles em que ocorrem todos os termos (MIN).

Uma outra questão importante é determinar o valor S_λ correcto. Toma-se como ponto de partida um valor de similaridade mínima especificado pelo utilizador S_{min} .

O outro valor S_{max} é calculado da seguinte forma. Ordena-se as fontes por $\mu + 3 * \sigma$. Para a fonte em primeiro lugar determina-se o valor de similaridade para o qual era devolvido pelo menos um documento.

Faz-se uma pesquisa binária no intervalo $[S_{min}, S_{max}]$ até encontrar o maior valor de similaridade para o qual o conjunto das BDTs dispõe de um número igual a k documentos com valor similaridade superior. Esse valor é o S_λ.

6.6.1 Um exemplo

Passa-se a apresentar um exemplo para ilustrar este método. Trata-se dum exemplo apenas ilustrativo já que o método só é aplicável para valores de m suficientemente grandes.

Suponha a existência de 3 BDTs com o seguinte dicionário de termos

$$\vec{T} = (a, b, c, d, e, f)$$

A informação mantida nos sumários é a apresentada na tabela 35.

Suponha que está em presença da interrogação I=(a,b) que corresponde a $i_1 = a$ e $i_2 = b$.

Para a BDT_1

Retira-se directamente da tabela que $\mu_{1a} = 4$, $\sigma_{1a} = 2$, $\mu_{1b} = 7,5$ e $\sigma_{1b} = 3$.

$$idf_{1a} = log\frac{1000}{300} = 0,522$$

$$idf_{1b} = log\frac{1000}{400} = 0,398$$

$$\mu_1 = \alpha_{1a} * \mu_{1a} + \alpha_{1b} * \mu_{1b}$$

termo	BDT_1			BDT_2			BDT_3		
	nºocorr		nºdocs	nºocorr		nºdocs	nºocorr		nºdocs
	$\sum$	σ^2		$\sum$	σ^2		$\sum$	σ^2	
a	1200	4	300	3000	9	400	4000	16	500
b	3000	9	400	1600	1	800	1400	1	700
c	1000	4	200	4800	25	400	3000	4	600
d	500	9	100	400	1	200	900	4	300
e	400	16	50	270	25	30	1800	90	20
f	800	25	80	700	36	100	50	0	50
Total docs	1000			1500			2000		

Tabela 35: Sumário das BDTs para exemplo

	Total	$> \mu$	$> \mu + \sigma$	$> \mu + 2 * \sigma$	$> \mu + 3 * \sigma$
		$> 10,14$	$> 13,03$	> 15.91	$> 17,29$
R_{max}	400	200	74,4	9,1	0,54
R_{med}	350	175	65,1	7,96	0,47
R_{min}	300	150	55,8	6,82	0,41

Tabela 36: Documentos estimados para BDT_1 com $\mu_1 = 10,1451$ e $\sigma_1^2 = (2,381)^2$

$$\mu_1 = idf_{1a} * idf_{1a} * i_a * \mu_{1a} + idf_{1b} * idf_{1b} * i_b * \mu_{1b}$$

$$\mu_1 = 2 * 0,522 * 4 + 2 * 0,398 * 7,5 = 10,1451$$

$$\sigma_1^2 = \alpha_{1a} * \sigma_{1a} + \alpha_{1b} * \sigma_{1b}$$

$$\sigma_1^2 = (2,381)^2$$

Também se pode estimar os valores dos documentos relevantes usando as várias lógicas: $R_{1min} = 300$; $R_{1med} = 350$; $R_{1max} = 400$.
Os número de documentos aparecem estimados na tabela 36. Recorde-se que os valores são $\mu_1 = 10,1451$

	Total	$> \mu$	$> \mu + \sigma$	$> \mu + 2 * \sigma$	$> \mu + 3 * \sigma$
		$> 6,83$	$> 10,12$	$> 13,41$	$> 16,72$
R_{max}	800	400	148,2	18,2	1,08
R_{med}	600	300	111,6	13,7	0,81
R_{min}	400	200	74,4	9,1	0,54

Tabela 37: Documentos estimados para BDT_2 com $\mu_2 = 6,83$ e $\sigma_2^2 = (3,29)^2$

	Total	$> \mu$	$> \mu + \sigma$	$> \mu + 2 * \sigma$	$> \mu + 3 * \sigma$
		$> 33,8$	$> 41,85$	$> 49,9$	$> 57,95$
R_{max}	700	350	129,5	14,87	0,945
R_{med}	600	300	111	13,7	0,81
R_{min}	500	250	92,5	10,5	0,675

Tabela 38: Documentos estimados para BDT_3 com $\mu_3 = 33.8$ e $\sigma_3^2 = (8,05)^2$

Os documentos estimados para a BDT_2 e BDT_3 aparecem nas tabelas 37 e 38, respectivamente. Os cálculos foram feitos da mesma forma com base na tabela dos sumários.
Se o utilizador pretende 40 documentos basta interrogar a BDT_3, pois de acordo com os cálculos (comparar tabelas 36, 37 e 38) apresenta os 40 documentos mais pontuados.

6.6.2 Síntese do algoritmo

Dada uma interrogação $I = (i_1, i_2, i_3, ..., i_m)$ e um valor k que indica o número de documentos pedidos pelo utilizador.
O objectivo é calcular um subconjunto de tamanho l de BDTs que devem se interrogadas dum total de N e estimar o vector $\vec{r} = (r_1, r_2, ..., rl)$ de documentos a pedir a cada BDT seleccionada.
Determinar numa primeira fase se existem BDTs completamente sobrepostas nas restantes. Eliminá-las a não ser que se justifique pelo facto de usarem processos de pontuação diferentes e se pretenda usá-las conjuntamente para melhor a eficácia da resposta. Neste último caso, isto deve ser considerado para efeitos de fusão. A detecção da sobreposição completa pode ser feita usando a matriz de sobreposição.
Processar as listas de ocorrências correspondente aos termos da interrogação e determinar para as BDTs correspondentes os parâmetros da distribuição normal nomeadamente a média

e o desvio padrão. Para todas as BDTs com ocorrências,

1. $\mu_j = 0$; $\sigma_j^2 = 0$
2. Para cada termo da interrogação (i=1 até m)
   (a) calcular $idf_{ji} = \log \frac{N_j}{n_{ji}}$
   (b) calcular $\alpha_{ji} = i_i * idf_{ji} * idf_{ji}$
   (c) calcular $\mu_j = \mu_j + \alpha_{ji} * \mu_{ji}$
   (d) calcular $\sigma_j^2 = \sigma_j^2 + \alpha_{ji} * \sigma_{ji}^2$

Uma vez determinados os parâmetros da distribuição de similaridade, as diversas BDTs são ordenadas usando o valor correspondente a $\mu + 3 * \sigma$.
Uma vez ordenadas as BDTs, determina-se o valor de corte da similaridade fazendo uma pesquisa binária começando com os limites de 0 e $\max(\mu + 3 * \sigma)$. O valor S_λ é o valor mínimo de similaridade para o qual o número de documentos é k.
Determinado o valor de S_λ determina-se o valor de l, isto é o número de bases de dados para o qual a soma dos documentos pedidos a cada base é o total de documentos pedidos pelo utilizador. Estima-se o número de documentos de cada BDT usando o valor de corte de similaridade e uma estimativa da população calculada com o auxílio duma heurística.
As heurísticas utilizadas já foram descritas anteriormente e são a soma, a média, o máximo e o mínimo.
No cálculo da população pode-se usar ou não a estimativa da sobreposição com as BDTs melhor posicionadas.
Se R corresponde à população , $\widehat{R} = R(1 - O)$ corresponde à população de documentos não sobrepostos. O número de documentos a pedir a BDT_i é $k_i = \widehat{R} * P(S > S_\lambda)$.

6.7 Arquitectura do selector

Os componentes principais do selector estão apresentados na figura 31.
Chegada a interrogação já pre-processada, as listas de ocorrência correspondentes aos seus termos são processadas. Deste processamento podem ser excluídas as BDTs que não cumprem as restrições do filtro booleano da interrogação (quer em termos da ocorrência de termos quer mesmo com base em sumários da meta-informação respectiva). Um outro tipo

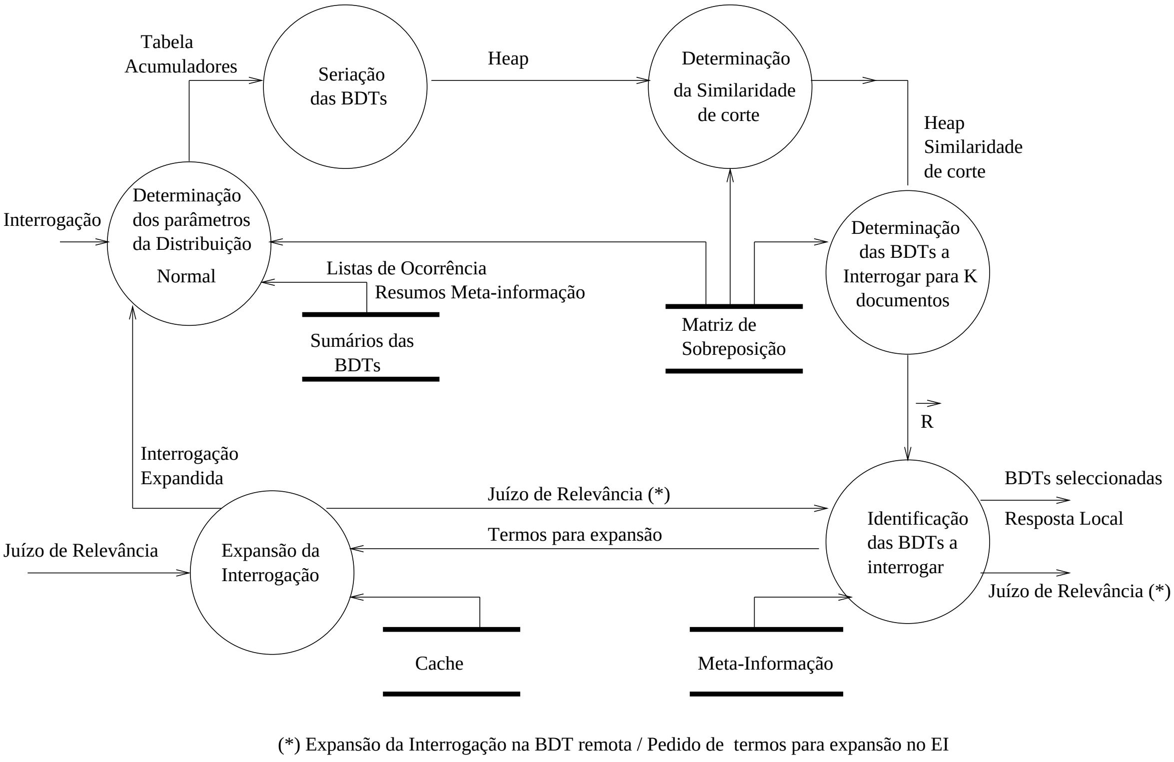


Figura 31: Diagrama funcional do selector

de restrição pode ser o facto de alguma da BD estar completamente sobreposta numa ou em várias das restantes. Nesse caso pode também não se justificar processá-la.

Processadas as listas de ocorrência, os acumuladores para as BDTs com esses termos contêm os parâmetros da distribuição normal da similaridade dos seus documentos relativamente a essa interrogação. Usando como chave o $\mu + 3 * \sigma$, os acumuladores são ordenados numa *heap*.

Usando essa *heap* e a lógica de cálculo da população em uso, determina-se por pesquisa binária entre $[S_{min}, Smax]$ o valor de similaridade de corte S_λ máximo que permita recuperar os k documentos pedidos pelo utilizador do conjunto de fontes disponíveis. Pode aqui ser usado um parâmetro dado pelo utilizador que indique o número máximo de fontes que podem ser interrogadas. O cálculo da população é desde já ajustado com a sobreposição.

Se o utilizador pretender uma resposta com base na informação dos sumários, é-lhe devolvida uma resposta local. Senão, as restricções de meta-informação são processadas para verificar se não é possível eliminar desde já algumas BDTs do espaço de pesquisa para esta interrogação.

Conhecida S_λ, determina-se o número de fontes que devem ser interrogadas isto é o $\vec{R}_i$. A identicação interna das BDTs e a pontuação de cada BDT permitem desde logo construir um esqueleto da resposta intermédia devolvida pelo selector. Este esqueleto fica preenchido consultando os registos de meta-informação das BDTs seleccionadas, é suposto existir o respectivo URL para posterior interrogação ou uma resposta com as referência se o modo *chaining* estiver desactivado.

Se a interrogação for submetida na forma dum juízo de relevância, contém a identificação dos documentos o que permite saber que BDT os contém (mantêm-se um registo dessa informação no EI). É preciso obter dessas BDTs os termos para expansão para construir uma interrogação e recomeçar o processo.

6.8 Resultados Experimentais

Nesta secção, apresentam-se as medidas obtidas para as várias métricas. Para esse efeito colocou-se o encaminhador de interrogações a funcionar em modo de selecção apenas, e nesse caso o resultado da selecção é logo devolvido para o cliente.

Os resultados da selecção não são mais que o número de documentos pedidos a cada BDT em cada uma das interrogações de teste.

Uma vez devolvidos os resultados da selecção, esses resultados são usados como ficheiros de entrada do programa *seval* (ver Capítulo 7) que produz as médidas para as métricas usadas. O programa utiliza como parâmetro de entrada os julgamentos de relevância para as interrogações usadas.

O funcionamento do programa *seval* é muito simples. Usando os julgamentos de relevância para cada interrogação e os ficheiro com os identificadores TREC de cada fonte calcula o número de documentos relevantes (RBR) ou relevantes não sobrepostos (ORBR) em cada uma das BDTs e ordena-as por ordem descrescente. Esses valores são usados como referências de base (B_i nas fórmulas das métricas). Os resultados directos da selecção são também ordenados por ordem decrescente do número de documentos pedidos, constituindo assim os E_i das mesmas fórmulas. Com base nesses valores são usadas as fórmulas para obter os valores C_n, $\overline{C}_n$ e P_n. O parâmetro n é a variável independente e constitui o número de BDTs seleccionadas.

6.8.1 Método proposto versus CORI

Pelo facto de se usar o CORI para comparar com o algoritmo proposto, e também para avaliar o efeito da sobreposição, vai-se incluir a fórmula utilizada para seriar as várias BDTs disponíveis.

$$T = \frac{df_{ij}}{df_{ij} + k.(1-b) + b\frac{cw_i}{ave_cw}}$$

$$I = \frac{log(\frac{|N|}{c_f} + 0.5)}{log(|N| + 1.0)}$$

$$P(t_j|C_i) = \alpha + (1-\alpha).T.I$$

Onde

df_{ij}	número de documentos que contêm o termo t_j na colecção C_i
cw_i	número de palavras na colecção C_i
$ave_c w$	média do número de palavras nas diferentes colecções
N	número de colecções
c_f	número de colecções em que ocorre o termo r_j
k	constante que controla o valor de df (o valor por omissão é 200)
b	constante que varia entre 0 e 1 que controla a sensibilidade ao valor actual de cw (o valor por omissão é 0.75)
α	probablidade por defeito (colocada a 0.4)

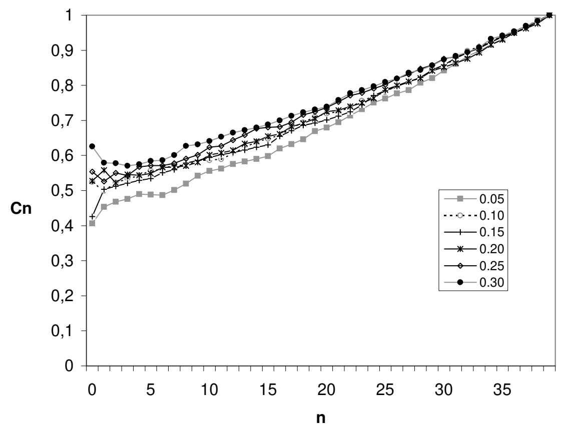
A pontuação atribuída a cada BDT depende da estrutura da interrogação, mas é normalmente apenas a média das probabilidades dos termos da interrogação.

O algoritmo proposto WEBT é prometedor, embora precise de mais informação das fontes que o CORI ou o método proposto pelo GLOSS. Este razão deste facto é não fazer pressupostos tão simplistas a respeito da distribuição dos termos pelos documentos de cada fonte. Esta abordagem vai permitir, como já foi apresentado anteriormente determinar com maior precisão o número de documentos a pedir a cada fonte. Entretanto o método proposto precisa de ser testado mais exaustivamente e comparado considerando outro tipo de métricas, nomeadamente as que medem o desempenho geral do sistema. O seguir essa via, seria entretanto, prolongar o trabalho e não nos traria conclusões a respeito da hipótese central da tese que é a influência da sobreposição nos algoritmos de fusão e seleção. Por este facto, vamos apenas comparar o desempenho do algoritmo proposto com um bem conhecido na literatura como o CORI e testar as nossas hipóteses sobre a sobreposição usando o CORI.

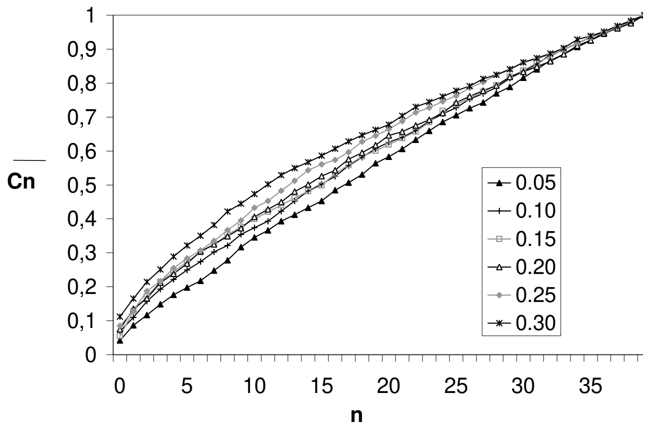
Nas figuras 32 e 33, são apresentadas as medidas obtidas para os vários cenários webt40.05 a webt40.30, usando o algoritmo proposto (WEBT) e o algoritmo CORI. Pelos valores obtidos, pode-se concluir que os resultados são da mesma ordem de grandeza. Isto significa que, pelo menos para os vários contextos utilizados, o algoritmo proposto pode ser usado sem grandes penalizações na eficácia.

6.8.2 CORI e sobreposição

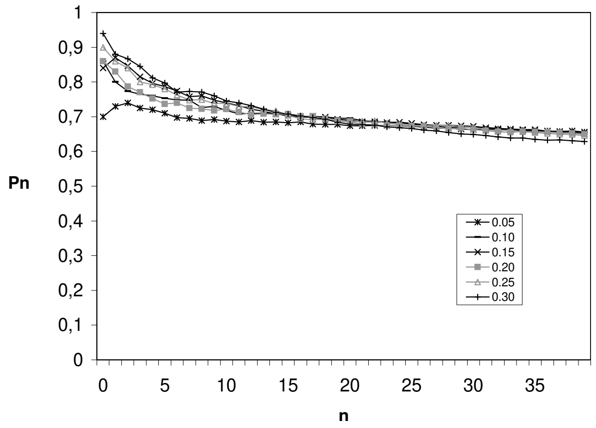
No conjunto de experiências seguintes, consideradas as mais importantes no que toca à selecção para a hipótese colocada nesta tese, apresenta-se o comportamento do algoritmo CORI, quando se introduz um factor dependente da sobreposição de acordo com a fórmula 5, introduzida anteriormente neste Capítulo.



(a) Cobertura

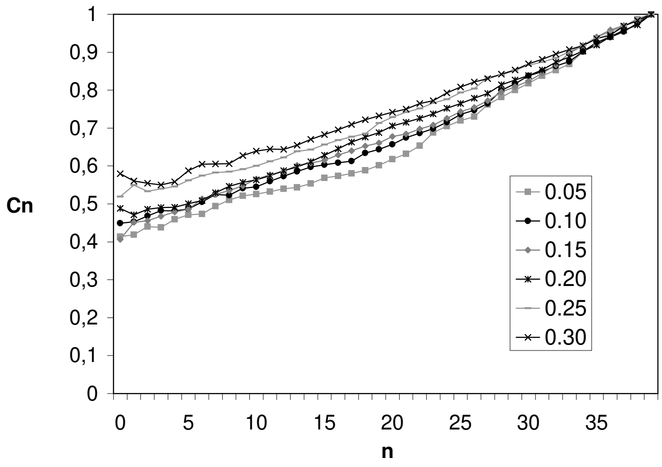


(b) Cobertura alternativa

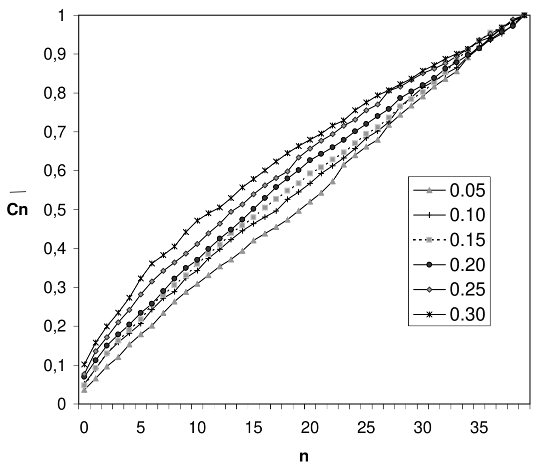


(c) Precisão

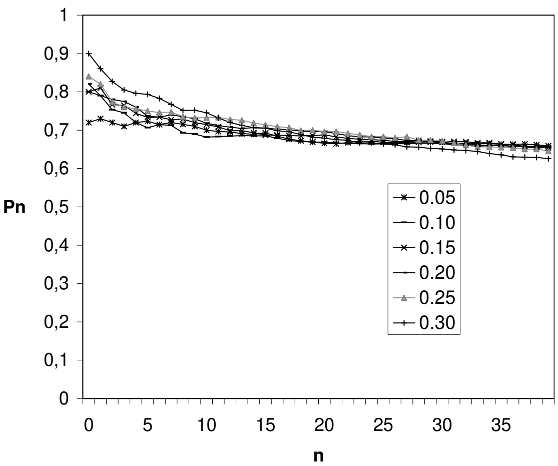
Figura 32: Medidas de eficácia para o algoritmo WEBT para cenários com 40 BDTs e variando a sobreposição



(a) Cobertura



(b) Cobertura alternativa



(c) Precisão

Figura 33: Medidas de eficácia para o algoritmo CORI para cenários com 40 BDTs e variando a sobreposição

Inicialmente, tentou-se usar os cenários originais apresentados nas tabelas 15 e 14, mas as diferenças nos resultados não eram significativas e resolveu-se avançar com cenários alternativos. A razão dessas diferenças insignificantes está no facto de não existirem variações significativas quer na quantidade de documentos quer na sobreposição. Com base nessa constatação, criaram-se cenários alternativos em que se tentou obter alternativamente uma maior variação no número de documentos e na sobreposição.

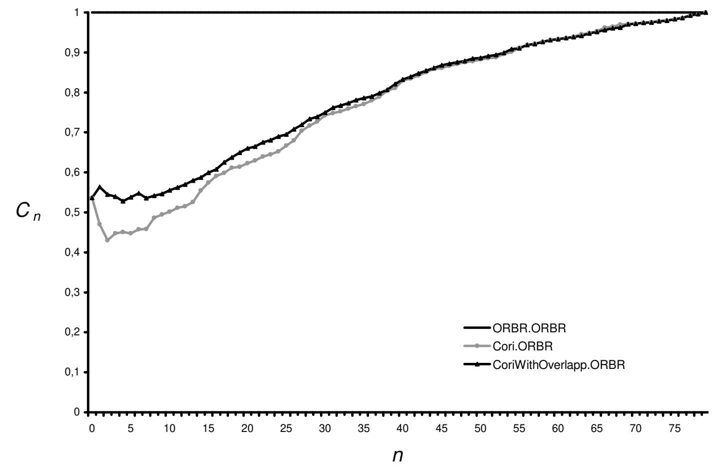
O primeiro desses cenários, webt80.xx, foi criado com base em BDTs dos vários cenários originais com 80 BDTs, mas com níveis diferentes de sobreposição, obtendo-se assim uma maior variação no número de documentos e na sobreposição. Os valores obtidos foram $\overline{S_g} = 93.06\%$, $\Delta S_g = 2.31\%$, $\overline{S} = 9.57\%$, $\Delta S = 11.73\%$ para a sobreposição (S_g é a sobreposição de um-para-todos e S é a sobreposição de um-para-um). e $\overline{D} = 8192.4$ e $\Delta D = 3001.7407$ para os documentos. Conseguiu-se assim alguma variância quer no nível de sobreposição quer no tamanho das BDTs.

O cenário seguinte também com 80 BDTs, designado por webt80.zz, foi criado com BDTs de cenários originais com 10, 20, 40 e 80 BDTs, obtendo-se uma maior variação no número de documentos embora se aumente apenas ligeiramente a variação na sobreposição. Os valores obtidos foram $\overline{S_g} = 99.25\%$, $\Delta S_g = 0.29\%$, $\overline{S} = 8.12\%$, $\Delta S = 6.9\%$ para a sobreposição e $\overline{D} = 11785.6$ e $\Delta D = 6497.2$.

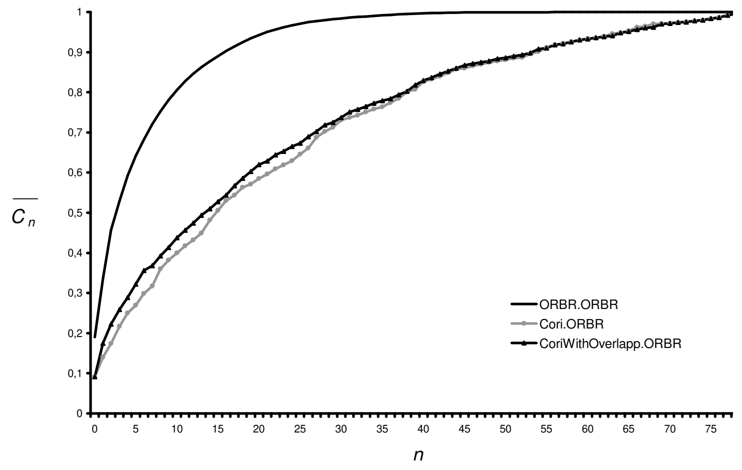
Por fim, usando o método de alocação com duas classes de sobreposição (uma de 5% e outra de 30%), criou-se o cenário webt80.var também com 80 BDTs em que pretendeu principalmente aumentar a variação na sobreposição. Os valores obtidos foram $\overline{S_g} = 58.3\%$, $\Delta S_g = 22.8\%$, $\overline{S} = 13.98\%$, $\Delta S = 16.12\%$ para a sobreposição e $\overline{D} = 5603.6$ e $\Delta D = 2425,4$.

Para o cenário webt80.xx, são visíveis na Figura 34 que se melhora os valores obtidos para as várias métricas usando a informação de sobreposição. Esta melhoria é mais visível até 50% das BDTs, altura em que é realmente mais importante. No cenário webt80.zz, as melhorias são quase imperceptíveis mas podem ser verificadas se se analisarem os valores obtidos. De qualquer forma não se conseguem melhorias significativas no cenário webt80.zz. No cenário webt80.var as melhorias são evidentes.

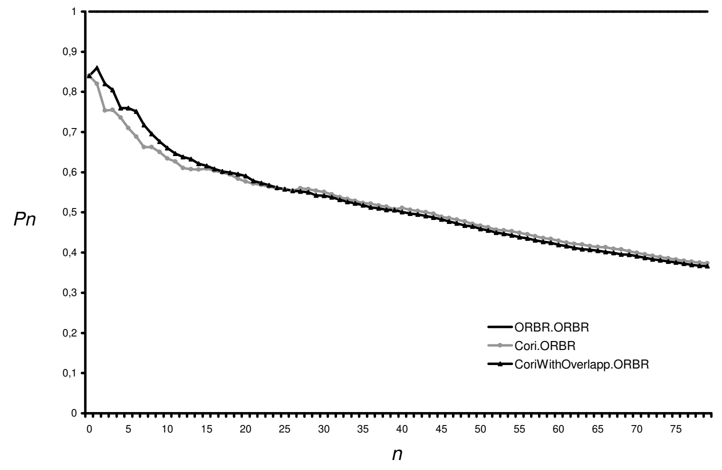
Os resultados apresentados nas figuras 34, 35 e 36 mostram que o factor dependente da sobreposição melhora a eficácia dos resultados e essa melhoria é mais efectiva quando a variância da sobreposição entre as diferentes BDTs aumenta.



(a) Cobertura

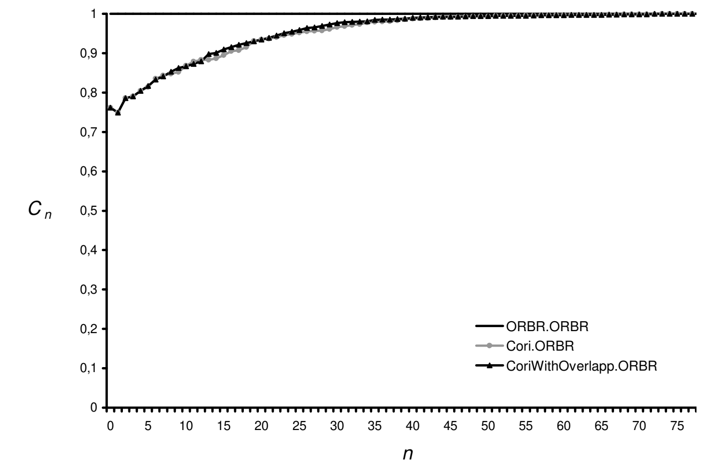


(b) Cobertura Alternativa

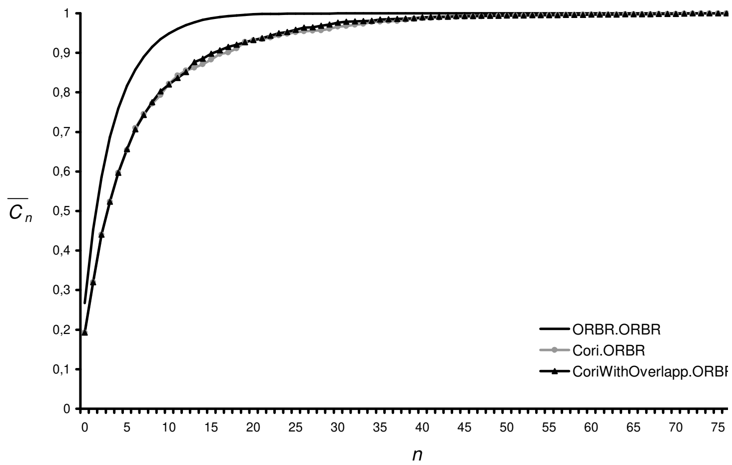


(c) Precisão

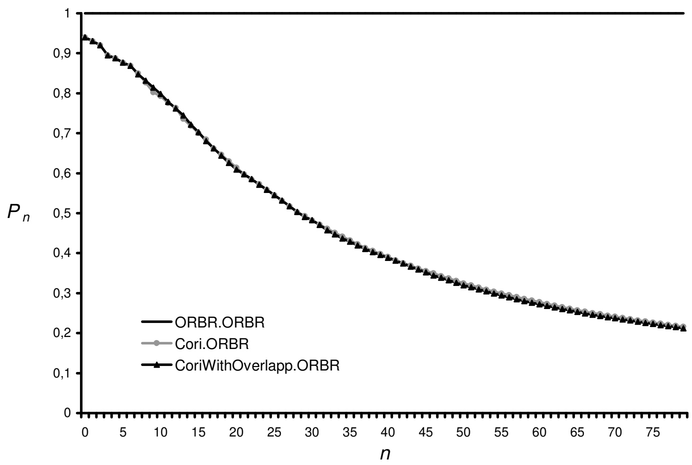
Figura 34: Resultados da selecçção para webt80.xx



(a) Cobertura

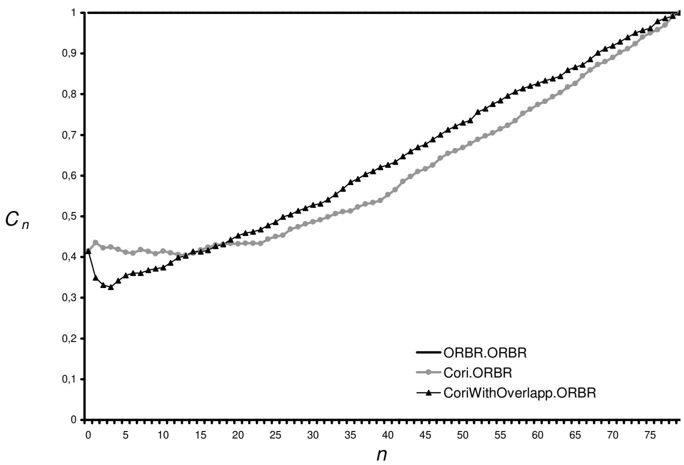


(b) Cobertura Alternativa

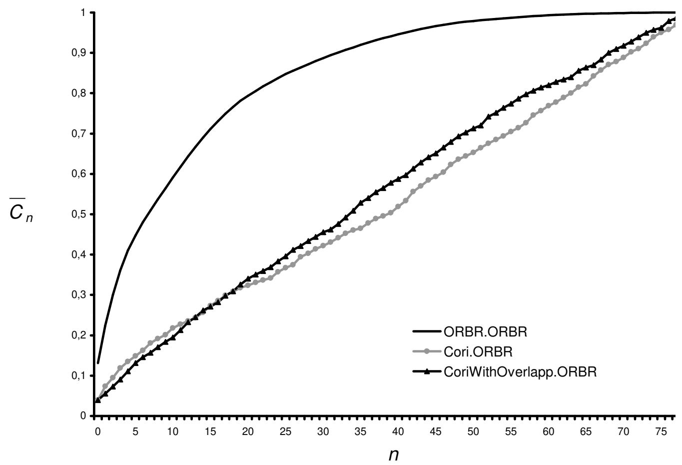


(c) Precisão

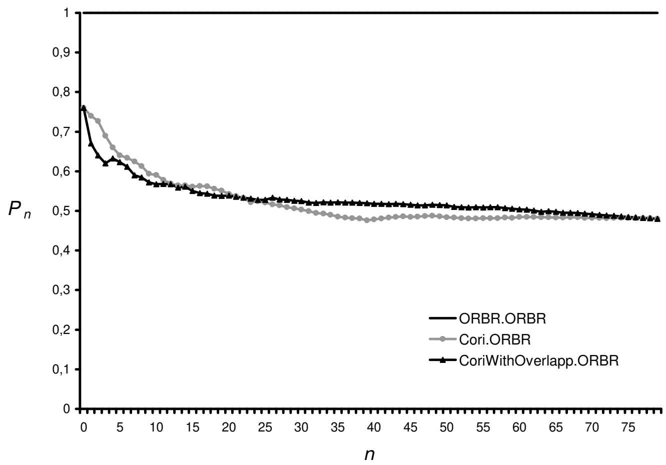
Figura 35: Resultados da seleccção para webt80.zz



(a) Cobertura



(b) Cobertura Alternativa



(c) Precisão

Figura 36: Resultados da seleccção para o cenário webt80.var

6.9 Trabalho relacionado

Conforme já se disse o método de selecção aqui proposto é baseado no Modelo do Espaço Vectorial tal como o proposto no GLOSS e gGLOSS.

O GLOSS usa, para cada termo, a soma de ocorrência e o número de documentos para cada sub-colecção. Com essa informação, ordena-se as colecções por ordem de importância e calcula-se a quantidade de documentos potencialmente relevante em cada colecção. O método de selecção introduzido também tem subjacente o modelo do espaço vectorial, embora precise, para além disso, da variância da ocorrências de cada termo nos vários documentos. Embora apresente uma eficácia da ordem de grandeza do CORI e GLOSS precisa de mais informação, pelo facto de usar a distribuição normal como modelo da ocorrência de cada termos nos documentos de cada BDT. Embora ainda não tenha sido completamente comprovado neste trabalho, pressupõe-se que o método tire vantagem desta informação adicional para calcular com mais precisão o número de documentos a pedir de cada BDT seleccionada.

Para além disso foi realizado um estudo sobre a influência da utilização da informação de sobreposição o no desempenho do algoritmo CORI[CPC+99, CLC95, LCC96]. Nesse estudo provou-se que se pode melhorar a eficácia do algoritmo, mesmo com uma indicação grosseira da sobreposição entre as interrogações dada pela sobreposição estática entre as diferentes BDTs. Com objectivo da avaliação, as métricas usadas [CPC+99] foram refinadas considerando a informação de sobreposição.

Um outro trabalho importante relacionado [LM99] propõe uma arquitectura de replicação para a RI distribuída, na qual um algoritmo de selecção de réplicas permite o balanceamento de carga, provocada pelas interrogações dos utilizadores, entre o servidor original e das réplicas, sem degradação da eficácia da resposta. O algoritmo de selecção de réplicas é também baseado no CORI. O objectivo deste trabalho não é propor uma arquitectura de replicação, mas sim usar a informação de distribuição de réplicas quando existe um determinado nível de sobreposição entre as BDTs elegíveis. A diferença básica vem de um caso se tratar de um ambiente de replicação coordenada e no caso aqui tratado de replicação não coordenada.

6.10 Discussão

O trabalho realizado dá uma forte indicação de que se pode conseguir melhorar a eficácia dos algoritmos de selecção, em ambientes de replicação não coordenada. No trabalho aqui realizado utilizou-se a sobreposição estática, que é uma aproximação muito grosseira da sobreposição entre os resultados das interrogações devolvidas pelas diferentes BDTs e mesmo assim se conseguiu melhorar a eficácia, ainda que de forma não muito significativa. Quanto maior a variância nos valores da sobreposição entre as diferentes BDTs, mais importante se torna o uso da informação de sobreposição.

Com um estimador adequado para a sobreposição um-a-um e um-para-vários, calculado por interrogação, os melhoramentos introduzidos por esta abordagem seriam, provavelmente , mais evidentes. Mas isto precisa de ser explorado em trabalho futuro.

A ideia a explorar é a utilização de métodos apropriados para construir uma BDT local em que exista uma amostra de documentos cada BDT remota (duplicados são permitidos). Usando uma heurística apropriada, essa BDT pode-nos permitir estimar a sobreposição entre as BDTs para uma dada interrogação. Os seus custos seriam partilhados pela possibilidade de expansão da interrogação.

O método de selecção proposto, embora simples, já permite seleccionar BDTs com uma eficácia da mesma ordem de grandeza de algoritmos apresentados na literatura, embora precise de mais informação.

Capítulo 7

Concretização do Protótipo

Sumário

Este capítulo tem como objectivo a descrição da concretização do protótipo usado na parte experimental desta tese. Embora a arquitectura usada já tenha sido descrita anteriormente, vale a pena descrever como foram concretizados os vários elementos da arquitectura e justificar as opções feitas durante essa concretização.

7.1 Introdução

Foi apresentada no Capítulo 3 , uma arquitectura para uma BDT distribuída para ser usada em ambiente operacional e laboratorial e cuja concretização em protótipo é o objectivo da parte experimental desta tese.

Ao longo desta tese e nos locais mais adequados já foram sendo apresentados resultados experimentais obtidos com este protótipo. Nesses vários capítulos, foram também detalhadas as funções mais importantes a realizar por este sistema distribuído de RI e especificados os respectivos componentes e algoritmos.

Para além das funções específicas de motor de busca distribuído, foram também introduzidas funções de serviço de documentos para permitir a sua utilização plena em ambiente laboratorial, usando apenas colecções de documentos de teste.

Tomando como base estes pressupostos, estabeleceu-se como objectivo concretizar um motor de busca distribuído com os seguintes componentes e funcionalidades:

- Robot WWW para indexação de documentos acessíveis na rede;
- Reconhecedor de documentos HTML [BLC95] e formatos TREC [Har93] para utilização de colecções de teste em ambiente laboratorial;
- Filtro de palavras comuns e radicalizador para Inglês;
- Alocador de documentos com ou sem replicação com as funcionalidades descritas no Capítulo 4;
- Motor de busca em conformidade com a interface STARTS [GCGM+97] e com possibilidade de actualizações em linha;
- Um Encaminhador de Interrogações[1] para selecção de BDTs e fusão de resultados e também em conformidade com o STARTS;
- Exportação de sumários de conteúdos pelas BDTs e EIs para EIs de nível superior;
- Uso do HTTP [BLFF96] e HTML para resposta as interrogações e pedidos documentos;
- Cliente baseado em formulários HTML;
- Cliente em lotes para interrogações de teste;
- Programas para avaliação da eficácia do sistema alinhados com o TREC;
- Programas para avaliação da eficácia da selecção de acordo com metodologias apresentadas na literatura, adicionando as funcionalidades e métodos descritos no Capítulo 6; e,
- Programas para avaliação da eficiência do sistema de RI e respectivos componenentes.

No resto deste capítulo passar-se-á a descrever como foram concretizados os diferentes módulos do Webtrieve e que ferramentas foram usadas.

[1] Vulgo broker na terminologia inglesa

7.2 As razões de algumas opções

Foram feitas opções que interessa justificar pela importância que tiveram na concretização do protótipo.
O objectivo desta tese é estudar os problemas específicos introduzidos pela distribuição das BDT. A primeira questão que se pode colocar é a justificação de se ter concretizado um sistema de RI completo, em vez de se usar um motor de busca já existente. Com essa opção havia, quando muito, necessidade de concretizar o EI, o alocador de documentos e os restantes programas.
Em termos históricos, a opção de usar um motor já existente foi a primeira escolha. Para esse efeito foram feitos contactos com o Grupo de MMIR da Dublin City University (DCU) e analisados mesmo outros pacotes. Naturalmente que o motor teria que sofrer algumas modificações, mesmo que ligeiras para poder ser usado. Isto exigia tanto dispor do código fonte como compreendê-lo para poder fazer as alterações. O código dos pacotes disponíveis estava muito pouco documentado e na maior parte dos casos nem sequer comentários havia. Infelizmente é o que se passa com muito software disponibilizado por equipas de investigação: estão bastante mal comentados e a sua utilização exige canais de comunicação privilegiados com os seus autores. No caso da BDT da DCU até existiam. Mas a pessoa em questão mudou-se para os Eire Telecom...
As vantagens resultantes da concretização de um sistema de RI de raiz são, para além da experiência adquirida, dispor de software que pode ser usado em projectos futuros de I&D, ensino e prestação de serviços.

Outra opção que importa justificar é a escolha do Java [Lim96].
Por se tratar da concretização dum sistema complexo e com bastantes componentes, mesmo com os atrasos introduzidos pela curva de aprendizagem duma linguagem nova, são inúmeras as vantagens de usar uma linguagem orientada por objectos: a possibilidade do encapsular essa complexidade nas diferentes classes de objectos ; a facilidade de modificação, extensibilidade, manutenção e configuração.
As razões acima descritas justificam a opção por uma linguagem orientada por objectos. Pelo facto de se tratar dum sistema distribuído há grande vantagem em usar uma linguagem multi-plataforma. Isto permite que uma única versão corra nas plataformas mais populares como o Windows, Linux, Unix e OS/2.
Em muitos programas, a eficiência pode ser melhorada com a introdução de múltiplos fios de

execução. Os motores de busca são um exemplo paradigmático. Por se tratar de um motor de pesquisa distribuído as possibilidades são ainda maiores. O Java permite manipular, com certa facilidade, os múltiplos fios de execução [Lea97].
Outra facilidade disponibilizada no Java é o suporte embebido de protocolos como o FTP e HTTP e a linguagem HTML. Ora como o principal objecto de trabalho é o WWW, as vantagens são óbvias. Uma vantagem acrescida é o fácil acesso a bases de dados relacionais. Um aspecto que não convém esquecer é o suporte de várias línguas que pode ser um aspecto relevante na indexação e recuperação de documentos no Web. Desde a versão 1.1 que o Java suporta múltiplas línguas.
Para além de se tratar duma linguagem bastante popular actualmente, uma das direcções de investigação é a utilização da mobilidade de código e os conceitos subjacentes às redes activas na recuperação e filtragem distribuída de informação.

7.3 Cenários de utilização

Existem diversos cenários de utilização do sistema de RI distribuído que implicam a existência e ou a importância dada aos diversos componentes do sistema. Vejamos, para concretizar, o diagrama apresentado na Figura 37.
No cenário operacional mais genérico existem M servidores de documentos e N BDTs colocadas em computadores diferentes. Essas N BDTs integram uma hierarquia de indexação com vários níveis de EIs. Se houver replicação coordenada (ver Figura 37), eles podem ser geridos centralmente, existindo unicamente um robot para colecção de documentos e um alocador para distribuição dos documentos ou seus representantes (rd na Figura 37) pelas diferentes BDTs. Caso a colecção e recolha de documentos e alocação de representantes seja feita centralmente pode-se justificar que a indexação dos documentos seja feita também centralmente evitando que documentos replicados em várias BDTs sejam indexados várias vezes. No caso duma distribuição em larga escala, cada organização mantém um nó ou uma sub-árvore da hierarquia de indexação.
Se existir replicação descoordenada, como se pode ver na Figura 38, não existe nenhum alocador e os diferentes robots podem descarregar dos mesmos servidores de documentos. Este cenário é o que está mais de acordo com os objectivos desta tese.
Os servidores de documentos (SDs) que são tipicamente servidores WWW estão dispersos por várias organizações e justifica-se a existência dum robot que recolha os documentos e

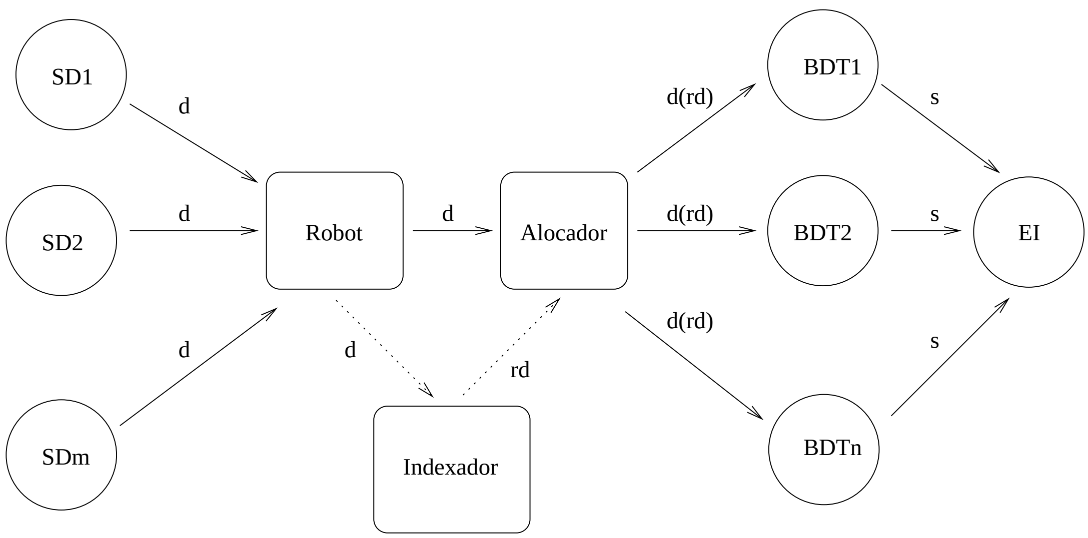


Figura 37: Cenário de utilização operacional com replicação coordenada. d - documento; rd - representante do documento; s - sumário

os coloque em disco local para posterior indexação.

O indexador pode ser bastante simplificado se os documentos a indexar estiverem todos na mesma língua e documentos do mesmo tipo.

Se existir uma única BDT não é necessário nenhum alocador. Nesse caso todos os documentos colectados são colocados na mesma BDT e também não é necessário nenhum EI.

Com várias BDTs é necessário a existência de pelo menos um EI que responda com documentos do todo o espaço de informação indexado. Nesse caso, cada BDT deve exportar previamente o sumário do seu conteúdo para o EI de nível superior.

No cenário laboratorial definido para esta tese, ilustrado na Figura 39, foram usados apenas documentos da colecção Wall Street Journal (WSJ) da TREC. A disponibilidade das fontes ao indexador permite a utilização dum robot simples para descarregar do sistema de ficheiros. O facto de se usar apenas documentos em inglês e duma única colecção do TREC também permite desligar no indexador os mecanismos de detecção de língua e de tipo de documento. Neste ambiente, a replicação descoordenada pode ser simulada usando

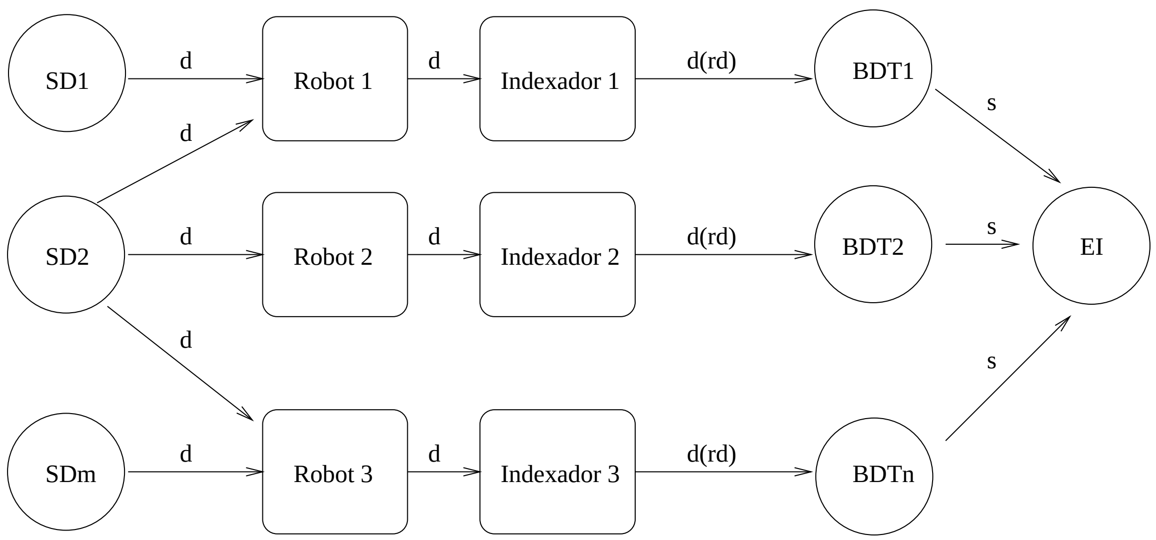


Figura 38: Cenário genérico de utilização operacional com replicação descoordenada. d - documento; rd - representante do documento; s - sumário

um método adequado de replicação, como o descrito no Capítulo 4.

Por limitações de recursos, a distribuição das várias BDTs e dum único EI é simulada por diferentes processos numa mesma máquina, pelo que se dispensa os mecanismos protocolares de suporte à distribuição de índices e exportação de sumários. De qualquer forma, manteve-se a utilização do protocolo STARTS encapsulado no HTTP, para troca de informação com o cliente e entre os diversos servidores durante o processamento da interrogação.

Pelo facto de se pretender neste trabalho apenas o estudo da influência da sobreposição nos algoritmos de fusão e selecção, os componentes da arquitectura concretizados foram apenas os indispensáveis para obtenção dos resultados experimentais, como enumerado na secção anterior.

7.4 O Ciclo de vida de uma fonte

Por fonte, entenda-se uma BDT centralizada ou distribuída que permita a recuperação de informação e o acesso a uma colecção de documentos. O conjunto de documentos é agrupado em colecção por qualquer razão administrativa, geográfica, política ou outra qualquer.

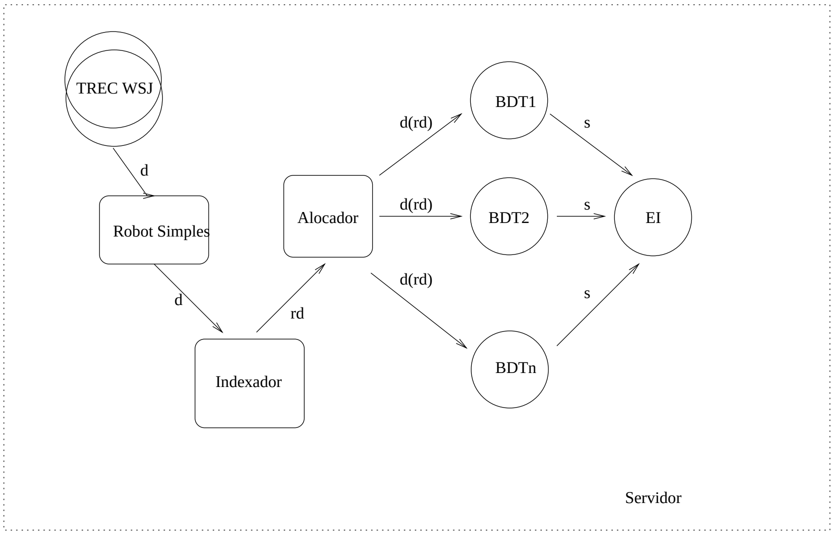


Figura 39: Cenário de experimentação laboratorial. d - documento; rd - representante do documento; s - sumário

Para identificação dos módulos necessários para o sistema de recuperação distribuída, usado em ambiente laboratorial, comecemos por apresentar o ciclo de vida duma fonte apresentada na Figura 40.

Este ciclo de vida é usado como diagrama de estados do programa principal do protótipo que permite a gestão de fontes.

O programa principal vai processando a fonte e actualizando o ficheiro respectivo. A título ilustrativo apresenta-se um ficheiro fonte nas Figuras 41 e 42.

A primeira mudança de estado ocorre para colecta de documentos (ver Figura 40). A fonte passa do estado inicial para o estado colectado. Uma vez armazenados em disco, os documentos precisam de ser reconhecidos sintaticamente e é extraída toda a informação considerada relevante para criação de índices. O processo de extracção corresponde a extrair todas as palavras e as estatísticas de ocorrência bem como informação de localização. Nesse processo obtém, para além disso, um registo de meta-informação para cada documento.

Após o reconhecimento sintático, o processo pode seguir dois percursos consoante se queira ou não manter todos os documentos da colecção num único motor de busca. Se for este o

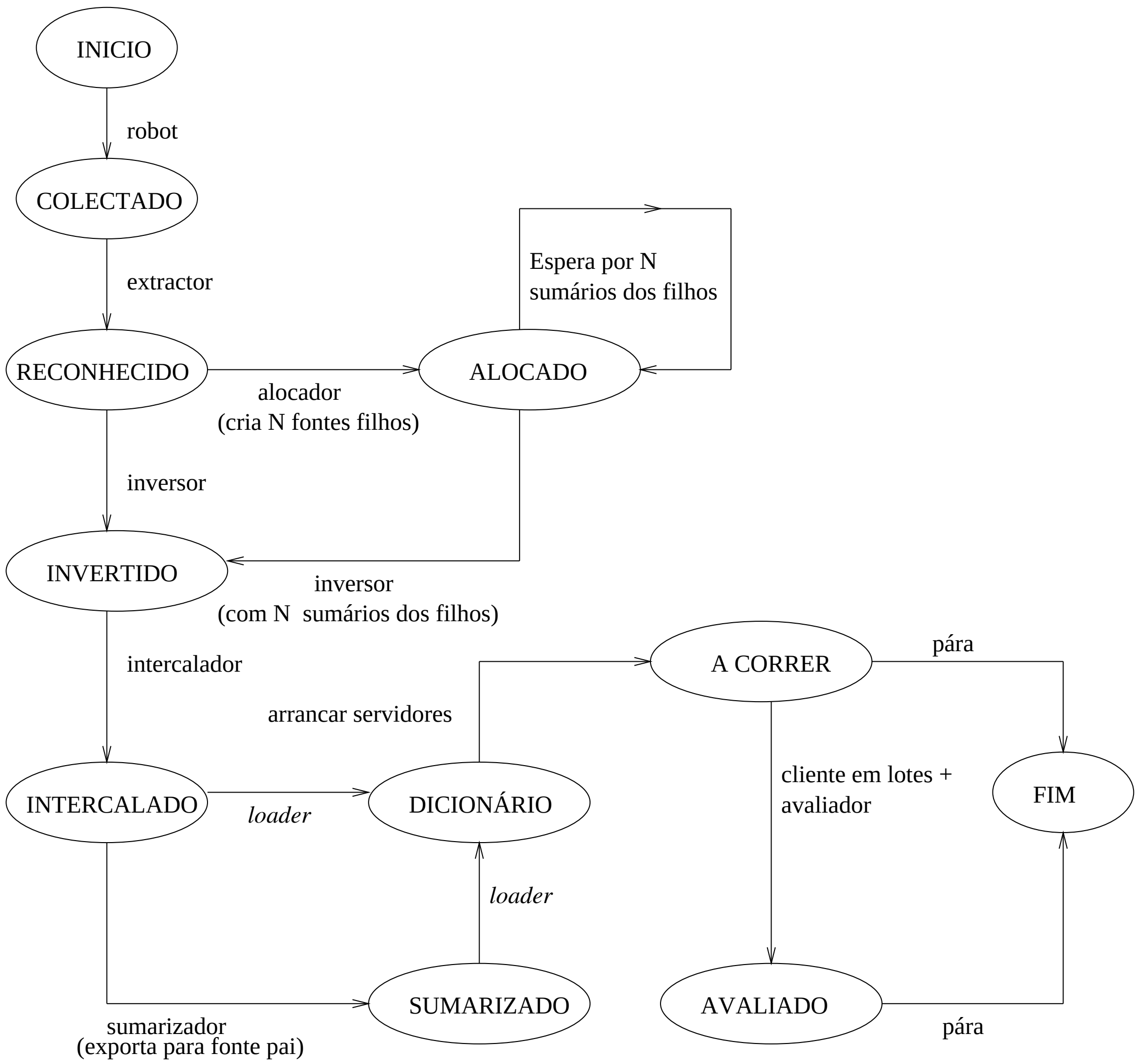


Figura 40: O ciclo de vida duma fonte

```
// Source file for source 'wsj_broker'...
// Last modified on Wed Jun 07 11:12:15 GMT+01:00 2000
// source current state
//   INVALID_STATE[-1], INIT[0], PARSE[1], CLONESOURCE[2], ALLOC[3],
//   WAITCHILDS[4], COPYCHILDSUMMARIES[5], INVERT[6], MERGE[7], SUMMARIZE[8],
//   LENGTHNORMALIZE[9], LOADMETA[10], LOADDICTIONARY[11], CONFIGURESERVER[12],
//   DOCNUMBERS[13], UPDATEMETADBINFO[14], ALLDONE[100]
currentState: 100
// source identification
sourceNumber: 1
sourceName: wsj_broker
description: Webtrieve Source
contact: macedo@uminho.pt
language: en
parentSource:
childSources: { wsj_broker0, wsj_broker1, wsj_broker2, wsj_broker3 }
isBroker: true
stemming: true
stopping: true
positioning: false
// source access info
serverName: polo.uminho.pt
serverPort: 5555
searchUrl: http://polo.uminho.pt:5555/
metadataUrl: http://polo.uminho.pt:5555/
// update policy
updatePolicy: { IN_PLACE_IF_ENOUGH_SPACE, NEW, PROPORTIONAL }
// engine behavior
// termWeightSchema(TCFNFX[0],NXXBPX[1],BFXBFX[2],BXXBPX[3],TXCTXX[4],BXXBXX[5]
)
termWeightSchema: 0
maxAccumulators: 10000
queryExpansion: 0
threshold: 0
normalizationSchema: 0
// broker behavior
// selectionPopulationLogic( MIN[0], MAX[1], AVG[2], SUM[3] )
// fusionLogic( COMBMIN[1], COMBMAX[2], COMBMED[3], COMBSUM[4], COMBANZ[5], COMB
MNZ[6], COMBGMNZ[7] )
selectionPopulationLogic: 3
selectionSchema: 0
maxSourceAccumulators: 1000
fusionSchema: 1
fusionLogic: 4
fusionUseSourceWeigth: 0
fusionOrderType: 1
fusionScoreType: 0
fusionNormalizationType: 1
```

Figura 41: Ficheiro de descrição da fonte (Parte 1 de 2)

```
// source inputs & outputs...
sourceInputs: { /usr1/webt/webtrieve/adm/input/ }
  ...
zipout: true
serial: false
// source counters...
firstDocid: 0
nextDocid: 715
numberOfDocs: 4
numberOfTerms: 14195
numberOfPostings: 31637
...

// allocation related data: only for broker sources!
numberOfDestinations: 4
allocationCriterion: random
replicationCriterion: overlap
overlappingWanted: 0.2
overlappingMatrixFilename: /usr/local/webt/adm/wsj_broker/output/wsj_broker.ovr.
dimf
// document set: all document numbers (TREC docnumber String) in this source
docNumbersFile: /usr/local/webt/adm/wsj_broker/output/wsj_broker.did.dimf.gz

// allocation related data: can be available on all sources!
overlappingValue: 0.0
// inversion related data: dxi2inv and merge
...
// (meta & dict) loader counters
numberOfMetaInformationRecords: 4
...
```

Figura 42: Ficheiro de descrição da fonte (Parte 2 de 2) caso, passa-se a informação extraída para o inversor. Senão, os documentos são manipulados por um alocador que os divide, usando um critério definido, por um conjunto de N sub-fontes.

Posteriormente, os documentos para cada fonte são invertidos e intercalados (se necessário) para constituir o ficheiro invertido [HFBYL92], que vai posteriormente carregado no dicionário de termos.

Se por acaso esse motor for indexado por um encaminhador de interrogações de nível superior, o ficheiro invertido precisa de ser resumido num sumário cujo conteúdo será exportado para esse EI (a sua fonte pai). É a partir dos sumários das suas bases de dados de texto componentes que, por sua vez, o EI constrói o seu ficheiro invertido.

Uma vez carregado o dicionário de termos, a BDT está pronta a ser interrogada. Uma vez interrogada, a BDT com base no processamento das listas de ocorrências devolve o conjunto de documentos que considera relevantes para a interrogação.

Por seu lado, o EI quando interrogado com base na informação mantida no seu dicionário de termos, selecciona as BDTs mais relevantes para essa interrogação. No seu modo de funcionamento normal, interroga as suas sub-fontes, junta os resultados num único que devolve ao utilizador. Para além disso, pode funcionar em modo só de selecção, caso em que devolve as fontes seleccionadas ao utilizador.

Em ambiente laboratorial para além da interrogação em modo interactivo, há a possibilidade de interrogação em lotes, usando interrogações de teste, para posterior avaliação da eficácia do sistema. A avaliação é feita usando um programa que compara as respostas devolvidas pelo sistema com um conjunto de documentos relevantes pré-definidos, medindo a precisão e a cobertura das respostas devolvidas.

O ciclo de vida duma fonte, descrito com a ajuda da Figura 40, permite a identificação dos módulos de software mais relevantes para este o pacote Webtrieve. Os módulos componentes do Webtrieve estão apresentados resumidamente na Tabela 39 com uma pequena descrição.

No que resta deste capítulo, vamos passar a descrever os aspectos mais importantes da concretização dos vários módulos apresentados na Tabela 39.

7.5 Formatos para troca de informação

Como se trata dum protótipo bastante complexo onde é necessário trocar informação em texto (por ficheiro ou não) entre vários módulos, a especificação cuidada dos formatos é um

http:	:	interface http para clientes e servidores
robot	:	robot WWW
parser	:	reconhecedores TREC concretizados com o JavaCC
stop	:	dicionário negativo para inglês
stem	:	radicalizador para inglês
allocator	:	alocador de documentos com replicação
basic	:	classes básicas
fusion	:	fusão de resultados
metadata	:	Manipulação de meta-informação
		Suportados os conjuntos de atributos
		dublin-core, basic-1, mbasic-1, webt-1
textdb	:	Dicionário de Termos (DT) com
	:	listas de ocorrência curtas e longas e com
	:	possibilidade de actualizações dinâmicas
loader	:	carregamento do DT e da meta-informação
engine	:	motor de busca (vários fórmulas de pontuação)
broker	:	EI com vários modos (difusão ou selecção, devolução
	:	de referência ou interrogação de sub-fontes)
starts	:	protocolo STARTS (baseado em
	:	código do Carl Lagoze, Cornell) e
	:	servidor http STARTS
eval	:	Código de avaliação alinhado com o TREC e
		cliente em lotes
overlap	:	cálculo e estimação da sobreposição
		entre diferentes BDTs
inversion	:	inversão e interlação de documentos
summarizer	:	sumarização de ficheiros invertidos
util	:	utilitários

Tabela 39: Módulos componentes do Webtrieve

aspecto bastante importante.

Optou-se pela existência de um formato binário e um formato em texto. Para este último, optou-se por uma linguagem de etiquetas especificada por uma gramática. Desta forma, existe uma linguagem sem ambiguidades para descrever os vários tipos de objectos cujas instâncias são susceptíveis de ocorrer nesses ficheiros.

Essa linguagem é designada como sendo DIMF[2] e todos os ficheiros descritos nesta linguagem têm a extensão dimf. Os ficheiros *.dimf podem também ser lidos com facilidade pelo programador permitindo uma maneira fácil mas precisa de visualização de resultados intermédios. Todos os ficheiros que não tenham a extensão DIMF estão em formato binário. Por outro lado, os reconhecedores sintácticos podem ser gerados automaticamente utilizando as ferramentas adequadas[3].

A DIMF é uma linguagem de etiquetas muito simples mas com a complexidade suficiente para descrever os objectos e suas instâncias transferidos entre programas neste pacote. Exemplos de objectos mais complexos descritos nesta linguagem são documentos invertidos, ficheiros invertidos, meta-informação de documentos, sumários de bases de dados, etc.... Como exemplos de objectos mais simples há um registo de ocorrência, um registo de localização, uma lista de ocorrência, um item de resultado, etc...

Para distinguir o tipo de objectos existentes num determinado ficheiro também se convencionou usar extensões típicas para cada tipo de objectos. Por exemplo wsj.inv.dimf é um ficheiro invertido no formato DIMF enquanto wsj.sum.gz é um ficheiro com um sumário no formato binário com o conteúdo compactado.

Assim todos os objectos persistentes dispõem duma interface que permita a sua escrita ou leitura em formato DIMF e formato série. Qualquer desses ficheiros pode ser ou não compactado tendo ou não uma extensão gz. Existem programas para compactar e descompactar todas as sub-fontes duma BDT distribuída.

7.6 Extração de Informação

Uma larga porção dos documentos acessíveis através da Internet, estão descritos em HTML. Isto significa que um documento é normalmente dividido em vários ficheiros HTML (nós hipertexto) que não se encontram no mesmo servidor.

[2] acrónimo de Distributed Index Markup Language

[3] Como por exemplo o JavaCC

Por este facto, acontece-nos frequentemente quando interpelamos os serviços de pesquisa mais populares da Internet, temos várias páginas do mesmo documento em diferentes itens do resultado.

A constatação é que não é fácil agrupar as diferentes páginas HTML que integram o mesmo documento. A situação complicou-se ainda mais com a utilização das *frames* no HTML. A abordagem seguida neste trabalho é tomar como unidade indexável o ficheiro fonte HTML. Para possibilitar um pós-processamento para agrupar as várias páginas dum mesmo documento, as hiperligações HTML são extraídas cuidadosamente do ficheiro fonte para o registo de meta-informação associada. Esta informação pode ser usada, quer para heurísticas que identifiquem páginas dum mesmo documento ou então para incluir apenas uma dum conjunto de páginas altamente relacionadas como item de resposta. Esta solução pode ser aceitável uma vez que as páginas omitidas estão apenas a distância dum accionar do rato.

Para extração da informação é necessário um conjunto de filtros. Um filtro é uma classe usada numa cadeia para processamento de texto. Neste contexto, os filtros podem ser acoplados para extrair os termos indexáveis e a meta-informaçãdum documento dado.

Um filtro aceita um certo formato na sua entrada e produz um determinado formato de saída.

No processo de indexação, um documento é processado pelos seguintes filtros, pela ordem indicada:

1. *Reconhecedor sintáctico*[4]: aceita um documento TREC ou HTML, devolvendo as palavras que ele contém. As etiquetas HTML ou TREC são removidas. Quando se pretende extrair informação de estrutura do documentos, como é o caso, para além da palavra propriamente dita, é registado o o componente do documento em que a palavra ocorreu (cabeçalho, corpo, título, subtítulo, autor, etc...).

2. *Dicionário negativo*[5]: o reconhecedor tem um parâmetro (stop) que reconhece elimina as palavras muito frequentes, sem qualquer potência de descriminação entre os documentos.

3. *Radicalizador*[6]: aplica uma série de regras de radicalização e devolve os radicais correspondentes aos documentos.

[4] *Vulgo parser*
[5] *Stopper, em inglês*
[6] *Stemmer,em inglês*

4. *Contador de ocorrências e registo de posição:* dada a sequência de palavras que constituem o texto de entrada, devolve uma tabela, com uma palavra por entrada e respectiva informação de ocorrência e localização no texto.

Para este processo, os reconhecedores são concretizados usando o JavaCC [Met00]. Tanto o Dicionário Negativo como o Radicalizador foram derivados da classe StreamTokenizer do Java. Com base no StreamTokenizer, foi desenvolvido um SimpleTextTokenizer que depois pode ser acoplado com um NoStopTokenizer e um StemTokenizer.

O dicionário negativo usa uma abordagem baseada num autómato finito determinista que permite detectar as palavras comuns à medida que estas vão sendo lidas [Fox92]. A sua utilização com outras línguas depende apenas de ter uma lista de palavras vulgares para cada língua.

O radicalizador para inglês é o baseado no algoritmo de Porter [Por80] que vem descrito em muitos artigos da literatura. Como um exemplo referencia-se [Fra92].

Relativamente aos componentes do texto em que as palavras aparecem, são atribuídas etiquetas geradas pela própria árvore de reconhecimento sintático, existindo uma classe onde se mantêm as etiquetas comuns para componentes similares, em vários tipos de documentos.

Uma vez obtidos os vários documentos invertidos, o programa de inversão agrupa as estatísticas dos vários radicais num ou vários ficheiros invertidos que depois precisam de ser intercalados.

Pode ser necessário algum pós-processamento nos ficheiros de índices, nomeadamente o seu carregamento no dicionário de termos.

Para o caso do processamento distribuído, um processamento adicional é, dado o ficheiro invertido para uma dada fonte, obter o sumário correspondente a ser exportado para os EIs que indexam a fonte.

7.7 Dicionário de Termos

O dicionário de termos é uma das componentes mais importantes e complexas que foram desenvolvidas. É ela que mantém quase toda a informação útil extraída dos documentos duma determinada fonte.

Por ser complexa, esta componente está dividida num conjunto de módulos mais pequenos, apresentados na Tabela 40.

plists	:	listas de ocorrências para documentos
ptables	:	tabelas usadas para armazenar as listas de ocorrências
summary	:	classes especiais usadas para gerar sumários de BDTs
util	:	classes usadas para *buckets* e *chunks*

Tabela 40: Os módulos do Dicionário de Termos

Duma forma simplista, a informação disponível no dicionário é um conjunto de palavras associadas às respectivas listas de ocorrências. A lista de ocorrência tem um registo de ocorrência por documento em que a palavra ocorre.

Como não podia deixar de ser, na prática é tudo um pouco mais complexo. O dicionário foi concretizado utilizando uma abordagem em que se trata de forma diferente as palavras mais frequentes e as palavras mais raras. As listas de ocorrência longas são armazenadas em cadeias de *chunks*, enquanto as listas de ocorrências curtas são mantidas em *buckets*. Um *chunk* é um bloco de tamanho fixo ou variável e um *bucket* é um bloco de tamanho fixo que contém várias listas de ocorrência. Os termos a manter nas listas longas são descobertos dinamicamente por transbordo das listas curtas mantidas em *buckets*. Quando transborda um *bucket*, a sua lista com maior comprimento é transformada numa lista longa, passando a ser armazenada em *chunks*.

Para além disso, podem existir listas de ocorrência em memória, originadas por actualizações incrementais, que ainda foram armazenadas em disco.

O diagrama funcional do dicionário de termos pode ser visto na Figura 43. Toda a interface com o exterior é feita através do Gestor do Dicionário que por sua vez utiliza os serviços do Gestor de Buckets e do Gestor de Chunks, consultando se necessário as listas de ocorrências em memória, resultantes das actualizações em linha.

Desta forma, para verificar se uma palavra ocorre no dicionário, é necessário verificar se existe alguma lista de ocorrência longa ou curta para esse termo. Eventualmente, pode haver também uma lista de ocorrência em memória.

Em tempo de arranque a frio do sistema, dado o ficheiro invertido, é necessário determinar o tamanho dos *buckets* e o número de *buckets*. O melhor tamanho da estrutura de *buckets* (o ficheiro em disco ocupado pelos *buckets*) depende da quantidade de informação indexada na BDT original. A utilização de dicionário negativo, radicalização e a granularidade do índice (documento, palavra ou frase) influencia bastante este tamanho. A granularidade

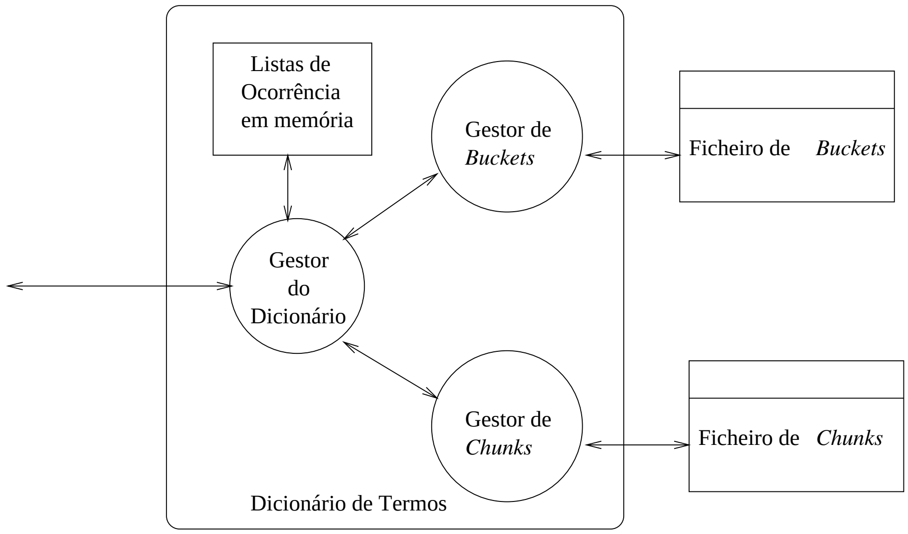


Figura 43: Diagrama funcional do Dicionário de Termos

indica o detalhe em é dada a informação de ocorrência. Se o ficheiro de *buckets* for muito pequeno conduz a um grande número de transbordos de listas de ocorrências pequenas em listas longas tornando o processo bastante ineficiente. Se o ficheiro for grande conduz a perda de espaço em disco e a ineficiência na leitura e escrita de *buckets* grandes.

Como ponto de partida, consideramos como palavras frequentes as 5 % de todas palavras com maior número de ocorrências. A partir deste pressuposto é possível determinar, com base em estatísticas (um histograma de tamanhos das várias listas de ocorrências), a dimensão ideal para o bucket e o número total de buckets necessário para armazenar esses 5%.

Apresenta-se na Tabela 41 um exemplo dos cálculos feitos para um EI (no cenário webt80.zz) e no cenário centralizado, de um única BD com toda a informação. Observe-se que não se está a usar informação de posição, e o tamanho das listas de ocorrência é proporcional ao número de documentos ou BDTs (no caso do EI) em que ocorre a palavra em questão. De qualquer forma, como se utiliza no programa o tamanho em bytes das listas de ocorrência, o presença da informação de posição não modificaria a abordagem utilizada.

Partiu-se do princípio que o tamanho mínimo dos *chunks* e *buckets* seria 8192 bytes que correspondem ao tamanho de bloco utilizado pelo sistema operativo.

No caso do EI (como existem apenas 80 BDTs), os sumários têm apenas listas de ocorrências com um tamanho máximo de 80 registos de ocorrência e um tamanho mínimo de 1 registo. Nesse caso, toda a informação é armazenada em 19024 *buckets* que ocupam aproximadamente 156 Megabytes e 4426 *chunks* que ocupam 36,3 Megabytes. No ficheiro invertido existem 262671 listas de ocorrência que se convertem em 12475 (4,75 %) listas de ocorrência longa (LOL) e 250196 (95,25 %) listas de ocorrência curtas (LOC). As primeiras armazenam 19% dos registos de ocorrência enquanto as últimas mantêm 80% dos registos de ocorrência.

As estatísticas obtidas no dicionário correspondente ao EI foram 25778 LOLs, cada uma delas com um único *chunk*. A percentagem de espaço utilizado é de 31,4 %. No caso do EI esta baixa percentagem de ocupação é provocada pelo facto dos chunks não serem dimensionados de acordo com o número máximo de BDTs existentes. Uma outra alternativa seria terem tamanho variável. Para além disso, existem *buckets* com uma ocupação de espaço de 80,3 % com 236893 listas de ocorrência. O espaço ocupado pelos *chunks* é 66,3 Mb.

No caso da BDT centralizada existem 264276 listas de ocorrência. O número previsto de LOLs é 13164 (5% do total) mantendo 94,1% do total de registos de ocorrência. As LOC são 251112 correspondendo a 95 % do total e mantendo 5,9 do total de registos de ocorrência. O número de *buckets* é 2334 ocupando 19,1 Megabytes. Quanto aos *chunks* são previstos 23042 ocupando 188,75 Megabytes.

No dicionário correspondente obtiveram-se 15114 LOLs com 33511 *chunks* a ocupar 188 MB com uma ocupação de espaço de 69%. O número de *buckets* é o previsto com uma percentagem de ocupação de espaço de 94,4%.

A abordagem dual à concretização do dicionário foi feita tendo em vista a existência de actualizações incrementais e primariamente para as BDTs, não para os EIs. Conforme se pode ver pelos valores previstos e obtidos há uma maior aproximação para o caso da BDT centralizada, como seria de esperar. No caso do EI o algoritmo de previsão é pouco preciso, pelo facto de existirem muitos termos com 80 registos de ocorrência. Neste caso, justifica-se que o programa de previsão use como parâmetro o facto de se tratar dum EI. Trata-se dum pormenor que é fácil resolver, pelo facto de saber à partida qual o tamanho máximo das listas de ocorrência. Um outro problema que tem o programa de dimensionamento é não considerar a utilização duma função de *hash* que faz com que os *buckets* nunca possam ter uma percentagem de ocupação de 100 %. Este facto pode ser usado como parâmetro. Relativamente à baixa percentagem de ocupação de *chunks*, resolve-se usando *chunks* de

Descripção	*EI*		*BDT*	
	P	E	P	E
cenário	webt80.zz		centralizada	
espaço para os termos		2829427		2847597
tamanho médio dos termos		10,78		10,78
n°de LO		262671		264276
n°de RO		5126107		25033156
max. registos em LO		80		135116
min. registos em LO		1		1
med. registos LO		19,5		94,8
n°de LOCs	250176	236983	251130	249162
% LOCs em LOs	95,3	90,18	95	94,3
n°de LOLs	12475	25778	13146	15141
% LOLs em LOs	4,7	9,92	95	5,7
n°ROs em LOCs	4128107	3283324	1482442	1354692
n°RO em LOLs	998000	1842783	23550714	23678464
tamanho *bucket* (bytes)		8192		8192
n°*buckets*		19024		2334
tamanho *chunk* (bytes)		8192		8192
n°*chunks*	4426	25778	23042	33511
tamanho ficheiro *buckets* MB		155,8		19,1
% ocupação de *buckets*		80,3		94,4
Mínimo de LOCs/*bucket*		2		76
Média de LOCs/*bucket*		12,45		106,8
Máximo de LOCs/*bucket*		25		139
tamanho ficheiro *chunks* MB	36	66,3	188	274
% ocupação de *chunks*		31,4		69
Mínimo de *chunks*/LOL		1		1
Média de *chunks*/LOL		1		2,2
Máximo de *chunks*/LOL		1		133

Tabela 41: Dimensionamento e estatísticas da estrutura dual no dicionário: P (Previsto) e E (efectivo).

Engenho	Encaminhador	Código Comum
SortedDocument	SortedSource	SortedItem
DocumentAccumulator	SourceAccumulator	Accumulator
EngineAccumulatorTable	BrokerAccumulatorTable	AccumulatorTable
EngineExpression	BrokerExpression	Expression
EngineRanking	BrokerRanking	Ranking
EngineFilter	BrokerFilter	Filter
EngineRankedFilter	BrokerRankedFilter	RankedFilter

Tabela 42: Partilha de código e semelhanças entre o Encaminhador de Interrogações e o Engenho de Pesquisa

tamanho variável, mais adequados para colecções de teste que são estáticas.
As actualizações incrementais são feitas com lotes de documentos numa periodicidade diária, por exemplo. Dado o conjunto de novos documentos, cria-se o ficheiro invertido respectivo. Posteriormente, é adicionado ao dicionário. Mantêm-se em memória central as várias listas de ocorrência que são posteriormente escritas em disco de acordo com a política utilizada. Existem duas políticas extremas de optimização respectivamente das actualizações e das interrogações. Para optimizar as interrogações a lista é lida do disco, acrescenta-se a actualização e escreve-se o resultado em disco. Para optimizar as actualizações acrescenta-se a actualização directamente no fim da lista em disco. Entre essas duas políticas extremas, há uma série de políticas intermédias que podem ser definidas.
As actualizações incrementais conduzem também à necessidade de expansão do tamanho dos *buckets*.

7.8 Motor de Busca e Encaminhador de Interrogações

Estes dois componentes que são baseados no Dicionário de Termos, partilham bastante código e são bastante similares uma vez que têm ambos como função o processamento das interrogações feitas pelo cliente. Uma ideia da parilha de código é dada na Tabela 42. Enquanto o motor de busca pontua documentos processando as listas de ocorrência dos termos da interrogação, no processo de selecção o EI faz precisamente para pontuar as BDTs. Por esse facto partilham bastante código e o componente mais importante que é o Dicionário de Termos.
Enquanto no caso da BDT cada termo mantém registos de ocorrência por cada documento

indexado, no EI cada termo do dicionário mantém registos de ocorrência por cada base de dados indexada.

Comecemos pelo Encaminhador de Interrogações. O seu modo de operação é determinado por duas variáveis: *chaining* e *broadcast*.

Se estiver a funcionar em modo de difusão (brodcast) o módulo de selecção é desligado e a interrogação é difundida para todas as BDTs ou EIs indexadas. Caso contrário, são seleccionadas as BDTs mais relevantes para a interrogação corrente.

A interrogação só é reenviada para as várias BDTs (todas ou as seleccionadas) se o modo *chaining* estiver activo. Caso contrário, as BDTs seleccionadas constituem a resposta a devolver ao utilizador. Trata-se da devolução dum conjunto de referências ao utilizador para serem interrogadas posteriormente. Este modo é também útil para avaliar a eficácia da selecção em Laboratório.

No modo *chaining*, o EI tem-se que invocar um cliente para cada BDT a ser interrogada. São acrescentados alguns campos que são necessários para posterior processamento depois de chegada a resposta. Como exemplo dum desses campos pode-se apresentar o MD5 [Riv92] que permite detectar documentos sobrepostos. Em ambiente laboratorial, o MD5 pode ser substituído pelo identificador TREC de cada documento.

Cada cliente é executado num fio de execução distinto, aguardando a resposta correspondente. Para obviar situações de falha, é arrancado um temporizador que dispara uma vez passado o tempo, definido por configuração, para esperar pela resposta das BDTs interrogadas.

As respostas devolvidas são registadas em ficheiros históricos para posterior actualização da estimativa de sobreposição entre as diferentes Bases de Dados.

Devolvidas as respostas elas são fundidas numa única resposta a devolver ao utilizador, de acordo com o algoritmo de fusão apresentado oportunamente.

O processamento da interrogação em cada BDT tem duas partes. A primeira correspondente ao filtro booleano que devolve o conjunto de documentos para os quais o filtro é verdadeiro (F). A segunda processa a expressão de ranking apenas para documentos que estejam incluídos no conjunto que satisfazem o filtro. Se o número de documentos devolvidos pelo processamento da expressão de pontuação (R) não for suficiente, o conjunto de resultados finais é preenchido com elementos de F que não pertençam a R, com pontuação zero.

Interessa agora referir pormenores de configuração do motor de busca. Disponibiliza seis

designação	**Fórmula**	**Descrição**
TCFNFX	$\sum_{i=1}^{m} \frac{tf_i(D)*\log\frac{N}{n_i}}{\sqrt{\sum_{k=1}^{m} tf_k(D)*\log\frac{N}{n_k}}} * (0.5 + \frac{0.5*tf_i(Q)}{\max tf(Q)})$	Melhor ponderação total
NXXBPX	$\sum_{i=1}^{m}(0.5 + \frac{0.5*tfi_i(D)}{\max(tf_i}) * \log\frac{N-n_i}{n_i}$	Mellhor pontuação com ponderação probabilística
BFXBFX	$\sum_{i=1}^{m}(\log\frac{N}{n_i})^2$	pontuação idf clássica
BXXBPX	$\sum_{i=1}^{m}\log\frac{N-n_i}{n_i}$	independência de termo binária
TXCTXX	$\sum_{i=1}^{m} \frac{tf_i(D)*tf_i(I)}{\sqrt{\sum_{i=1}^{m} tf_i(D)^2}}$	peso tf normalizado
BXXBXX	$\sum_{i=1}^{m} 1$	nível de co-ordenação

Tabela 43: Fórmulas usadas para pontuação no motor

fórmulas diferentes para pontuação dos documentos como se pode observar na Tabela 43. A terminologia usada e as fórmulas apresentadas vêm descritas em [SB88]. Cada fórmula de pontuação é descrita por duas triplas representando respectivamente, a frequência do termo, a frequência na colecção e factores de normalização para os termos do documento (1ªtripla) e termos da interrogação (2ªtripla). São usados três componentes diferentes para a frequência do termo nomeadamente o peso binário (B), o normal (T) e o normalizado (N). São usados três componentes para a frequência da colecção que representam multiplicadores de 1(X) um factor convencional proporcional à frequência inversa (F) e um factor probabilístico(P). Finalmente, o factor de normalização de comprimento pode estar presente (C) ou ausente(X). É esta a terminologia que se apresenta na Tabela 43 e no ficheiro de configuração do motor de busca. São incluídas fórmulas de pontuação que vão desde a co-ordenação até à do coseno.

Estão também disponíveis várias alternativas para normalização de documentos nomeadamente a do coseno (norma do vector de termos do documento) e a baseada no tamanho em bytes [SSMB94].

No processamento das interrogações, é comum existir um acumulador por cada documento que conste das listas de ocorrências dos termos da interrogação. Nesse acumulador, vai sendo somada a pontuação parcial devido a cada termo da interrogação que ocorra nesse documento. Uma vez processados todos os termos, o acumulador tem a pontuação atribuída ao documento.

Se os termos forem ordenados por ordem crescente do tamanho das suas listas de ocorrência, são processados primeiro os termos com maior potência de descriminação (ou idf). Nessa situação, faz sentido limitar o número máximo de acumuladores, pois os documentos que ocorrem apenas nos termos com menor peso vão provavelmente ter uma pequena pontuação. Os acumuladores são mantidos numa Tabela que é uma árvore binária balanceada. Esta estrutura de dados foi implementada de raiz, para se obter eficiência na pesquisa. nessa altura ainda não existia o módulo *collections* do Java, pelo que não havia alternativa. A chave usada na árvore é o identificador interno do documento.
Um dos problemas críticos no processamento das interrogações é a ordenação. É necessário dada uma interrogação, ordenar os termos de acordo com o seu peso. Os acumuladores tambríe precisam de ser ordendaos de acordo com a pontuação atribuída ao documento correspondente. Como, pelos menos no último caso, pode corresponder a milhares de elementos, o tempo de ordenação pode contribuir sobremaneira para o tempo de resposta. Para esse efeito foi concretizada uma heap (e derivada uma mini-heap), isto é uma fila de prioridade baseada em vectores de acordo com algoritmo proposto em [GBY90]. Algum do código foi mesmo adaptado do código em Pascal original.
Adicionalmente , e embora não esteja ainda totalmente implementado é possível activar a expansão automática de interrogações [Har92b] e a existência de algoritmos de corte (*thresholding*) nas listas de ocorrência [KS96].

7.9 Meta-informação

Para tratamento da meta-informação, são suportados registos com campos definidos em três conjuntos de atributos fundamentais: o BASIC-1, o DublinCore e o MBASIC-1.
O DublinCore e o BASIC-1 são usados para suportar a meta-informação dos documentos e o MBASIC-1 para a meta-informação das fontes. Trata-se de atributos que são normalizados de facto ou de *jure*. Esta utilização foi imposta pela opção feita a favor do STARTS.
Foram ainda definidos alguns atributos específicos do Webtrieve, Webt-1, quer ditados pelo formato dos documentos TREC, quer por outros usados internamente no processamento das interrogações e na normalização dos resultados.
Para cada documento é mantida em memória central um Meta-índice com toda a informação por documento, necessária para o processamento da interrogação, e ainda um apontador para o ficheiro em que se encontra o registo de meta-informação do documento. Para

http	:	classes para colocação de headers http
parsing	:	reconhecimento de expressões de ranking e filtros booleanos usados por interrogações STARTS
query	:	manipulação da interrogação
regexp	:	manipulação de expressões regulares
resource	:	especificação das fontes interrogáveis
results	:	modulo para manipulação de resultados
soif	:	encapsulamento e desempacotamento de SOIFs (Standard Object Interchange Format)
wais	:	classes de ligação ao WAIS
webt	:	classes específicas de ligação ao Webtrieve (usando o modulo wais como exemplo)

Tabela 44: Os módulos do STARTS

esse efeito são usadas tabelas de *hash*. No caso do EI, é usada abordagem idêntica para manipular a meta-informação associada a cada BDT.

Podem ser introduzidos com facilidade novos conjuntos de atributos e também esquemas de registos para suportar novos tipos de documentos. Existem mecanismos de tradução de campos para registos que usem conjuntos de atributos diferentes.

Embora a meta-informação seja para já mantida num ficheiro de acesso directo, esta estratégia também pode ser alterada com alguma facilidade. E deverá sê-lo no futuro.

7.10 Comunicação entre os componentes

O protocolo usado para a interacção cliente/servidor foi o protocolo STARTS encapsulado no HTTP.

A parte do código correspondente ao STARTS foi gentilmente cedida pelo Carl Lagoze da Universidade de Cornell, EUA. Entretanto precisou de algumas modificações, para poder ser usado conjuntamente com o Webtrieve.

Trata-se, como não podia deixar de ser, de um componente duma certa complexidade e que está também dividido nos módulos apresentados na Tabela 44.

Nesse sentido, foi concretizado um servidor HTTP dedicado capaz de interpretar os pedidos HTTP e tambem os pedidos STARTS. Esta opção foi feita principalmente para evitar a dependência dum servidor HTTP em particular e a utilização de CGIs [W3C96]. Também

é possível, se se justificar, comunicar usando o STARTS directamente sobre o TCP. Está assim facilitada a automizatição da comunicação entre servidores no futuro.
O servidor manipula directamente HTTP e STARTS. Fica à escuta numa porta e arranca um fio de execução para atender cada novo pedido.
O servidor manipula os pedidos HTTP para servir ficheiros e também pedidos STARTS. Um interacção STARTS pode ser:

1. Interrogação
2. Pedido de sumário de conteúdo da fonte
3. O conteúdo da fonte
4. A meta-informação da fonte
5. A meta-informação do recurso (que corresponde a um conjunto de fontes)

O processamento de uma interrogação é o mais complexo. Esse processamento começa pela interrogação de entrada e termina com os resultados da interrogação. A abordagem usada pelo Webtrieve é manter no objecto *Query* toda a informação de processamento. Começa-se por colocar lá a interrogação original. Esta interrogação é objecto de pré-processamento para extracção do filtro booleano e expressão de pontuação que são colocados nas estruturas de dados adequadas para processamento. No processamento da expressão de pontuação, primeiro obtém-se os termos e ordena-se pela sua importância. Posteriormente, processa-se as listas de ocorrência de cada termo, colocando os resultados parciais numa tabela de acumuladores, um por documento. Processadas as listas tem-se nos acumuladores a pontuação de cada documento. A seguir os documentos filtrados usando o filtro booleano. Os que satisfazerem o filtro são ordenados por ordem decrescente de pontuação e os M mais pontuados são devolvidos ao utilizador pela mesma ordem. O número M é o total de documentos pedidos pelo utilizador. Se não se conseguir os M documentos por este processo são obtidos os restantes do conjunto de documentos da BDT a que se atribui a pontuação zero. Obtido o resultado, coloca-se também num campo da *Query*. Por fim, resultado é ordenado de acordo com os campos indicados, devolvendo-se mais uma vez apenas os campos pedidos.
Para processamento da interrogação propriamente dita, é invocado o código do EI se estiver configurado como *Broker* ou pelo código da BDT se estiver configurado como *Engine*. O

processamento é o descrito anteriormente, que corresponde no caso do EI apenas à selecção das fontes que depois são interrogadas e o resultados fundido. No caso do EI, constrói-se posteriormente à selecção uma nova *Query* por fonte a interrogar. Os respectivos resultados são depois fundidos e constituiem a resposta a colocar na *Query* original.

Antes do processamento, a interrogação pode ter que ser traduzida de acordo com correspondências especificadas na fonte. Os componentes documento podem ter nomes diferentes e pode inclusive haver operadores que não são suportados.

Esta explicação repete em certa medida a que já foi feita para o funcionamento do EI e da BDT.

Os restantes pedidos resumem-se à devolução de ficheiros já construídos pelo que podem ser devolvidos pelo servidor, sem grande processamento adicional.

7.11 Avaliação da eficácia

O software de avaliação está alinhado com o usado para o mesmo fim pelo TREC, que aliás lhe serviu de base.

Existe um cliente em lotes capaz de usar os ficheiros de tópicos TREC, isto é as interrogações de teste, e submetê-las ao sistema.

Como se pode verificar pela Figura 44, o cliente em lotes pode ser usado em diferentes cenários de avaliação.

O modo de difusão (*broadcast* $(-b)$) permite avaliar apenas a fusão, desligando-se a selecção de BDs.

O modo de selecção $(-e)$ devolve apenas os resultados da selecção, permitindo por exemplo a sua avaliação em separado. Nesse caso, o conjunto de resultados devolvido é constuído pelas bases de dados consideradas mais relevantes para a interrogação.

As interrogações podem ser geradas em duas variantes: curtas e longas. Na variante longa usa-se todo o texto do tópico TREC, enquanto na curta se usa apenas o texto de alguns campos bem definidos. Também se indica que subconjunto de interrogações se vai usar (os parâmetros $-l$ e $-f$) e quantos documentos são devolvidos em cada resposta $(-n)$.

Para além disso, o cliente em lotes não precisa de estar hospedado na mesma máquina em que está o servidor, permitindo que se indique a sua localização na rede (o parâmetro $-u$, o *url*). Como podem existir várias fontes no mesmo servidor é necessário para além disso indicar que fonte se pretende interrogar (o parâmetro $-s$).

```
polo.uminho.pt% java webtrieve.eval.bc
 Must specify arguments!
 usage: bc [ -b -e -q -u url -s source -f # -l # -n # -o outputPath]
        InputPath

         -b    request broadcast mode to a broker
         -e    request DB selection results only
         -q    uses short queries instead of long
         -u url    server url to send request to
         -s src    source name to query
         -f #  number of first TREC topic to send
         -l #  number of last TREC topic to send
         -n #  number of results to retrieve
         -o path      output dir to store results
         InputPath    TREC topics directory
```

Figura 44: Cliente em lotes para Webtrieve

As respostas recebidas incluem todos os campos necessários para a execução do programa de avaliação o *trec_eval*, disponibilizado pelo TREC. Trata-se duma classe derivada do ResultSet e do Document que contêm os métodos necessários para imprimir os ficheiros em formato compatível com o *trec_eval*, isto é as *trec_top_files*.

Como os documentos são distribuídos por diferentes motores, é necessário existir um pequeno programa que filtra o ficheiro com julgamentos de relevância, mantendo apenas os existem numa determinada fonte.

O *trec_eval* é (ver Figura 45) posteriormente utilizado para gerar os dados necessários para construir as curvas de precisão e cobertura tanto para uma interrogação como para um conjunto especificado de interrogações anteriormente realizado. Esses dados são obtidos consultando os julgamentos de relevância disponibilizados pelo TREC para cada uma das interrogações.

O programa *trec_eval* (ver Figura 45) precisa dos ficheiros com os documentos devolvidos por resposta e dos julgamentos de relevância. Os outros parâmetros (q, a, d) permitem obter resultados mais ou menos detalhados.

Para avaliar a fusão usa-se o procedimento a seguir apresentado. Suponhamos que vamos

```
 polo% trec_eval
Usage: trec_eval [-q] [-a] [-d] [-t] trec_rel_file trec_top_file
```

Figura 45: Programa para avaliação da eficácia da BDT ou EI

usar o cenário *wsjeval_broker*.

1. Editar o ficheiro *wsjeval_broker.src* e colocar os parametros fusionSchema, fusionLogic, fusionScoreType, overlapMethod de acordo com o cenario de fusão que se pretende avaliar O EI deve ser configurado para funcionar em modo chaining.

2. Arrancar o servidor EI

   ```
   java webtrieve.starts.aloneServer wsjeval_broker
   ```

3. Arrancar as BDTs indexadas pelo EI. Essas BDTs estão indicadas no ficheiro fonte do EI como sendo os seus filhos.

   ```
   java webtrieve.starts.aloneServer wsjeval_broker0 \
         wsjeval_broker1  wsjeval_broker2 wsjeval_broker2
   ```

   em modo brodcast (-b)

   ```
   java webtrieve.eval.bc -b -u http://polo.uminho.pt:6666 -s \
    wsjeval_broker -f 901 -l 910 -n 20 -o resultados-fusao  \
      /usr1/webt/webtrieve/adm/eval/queries-wsjeval
   ```

4. As *trec_top_files* para cada uma das interrogações são armazenadas na directoria resultados-fusao. Resta invocar o programa $trec_eval$

   ```
   trec_eval qrels.wsjeval resultados-fusao > fusao.evl
   ```

```
java webtrieve.eval.seval
Must specify arguments!
usage:  seleval -f # -l # -r qrelsFilename -e estimationFilename \
                -o outputPath  source
```

Figura 46: Programa para avaliação da selecção

No ficheiro fusao.evl temos os resultados da eficácia da fusao.

Para avaliação completa da BDT distribuída, foram desenvolvidos um conjunto de programas para avaliação do módulo de selecção. Interessa realçar o programa *selval* (ver Figura 46) que produz precisamente, para a selecção, o mesmo tipo de resultados que os produzidos pelo *trec_eval* para o caso centralizado. As métricas usadas são a precisão (P_n), a cobertura (C_n) e cobertura alternativa ($\widehat{C}_n$). Para além disso, são usados o erro quadrático médio e o coeficiente de correlação de Spearman.
O programa necessita das interrogações inicial e final (-f e -l), o ficheiro com os juízos de relevância e o ficheiro com os resultados da selecção numa dada fonte. O ficheiro com os resultados da selecção é obtido usando o cliente de lotes com o sistema configurado apenas para o modo de selecção.
Única e simplesmente por razões de eficiência, o programa *seval* foi dividido em dois. Inicialmente ele produzia directamente a selecção óptima considerando ou não a sobreposição. A selecção óptima considerando a sobreposição é um programa muito demorado uma vez que é necessário:

1. Ordenar as N BDs pelo número de documentos relevantes;
2. Para $k = 1, N-1$, remover os documentos relevantes da BD_k, na sub-sequência $BD_{k+1}, .., BD_N$; e
3. Ordenar o conjunto $BD_{k+1}, .., BD_N$.

Por esse facto, e porque cada cenário ($webtX.Y$) poder ser avaliado variando uma série de parâmetros, há um programa *baseline* que produz ficheiros com extensão *rbr* (não considerando a sobreposição) e *orbr* (considerando a sobreposição), conforme se use a métrica base usada para avaliação da selecção, usando como entrada os julgamentos de relevância

e os ficheiros dos identificadores dos documentos para cada um dos motores do EI. Posteriormente esses ficheiros são lidos pelo programa *seleval*, que necessita de qualquer forma dos julgamentos de relevância quando se usa a *orbr*, por causa da remoção dos documentos das BDTs com maior valor estimado de pontuação.
Desta maneira, para avaliar um algoritmo de selecção, é necessário usando por exemplo o EI *wsjeval_broker*

1. Editar o ficheiro da fonte, *wsjeval_broker.src* e modificar o parâmetro *selectionSchema* para o método de selecção a avaliar;

2. Arrancar o servidor como broker

```
java webtrieve.starts.aloneServer wsjeval_broker
```

3. Interrogá-lo com o cliente em modo de selecção

```
java webtrieve.eval.bc -e -u http://polo.uminho.pt:6666 -s \
  wsjeval_broker -f 901 -l910 -n 20 -o .  \
    /usr1/webt/webtrieve/adm/eval/queries-wsjeval
```

   No final do programa dispõe-se do ficheiro com os resultados da selecção, *wsjeval_broker.sel*, para as várias interrogações realizadas, 901 a 910.

4. Posteriormente, é necessário criar os ficheiros com as referências de base, isto é a selecção óptima considerando ou não a sobreposição.

```
java webtrieve.eval.baseline -rbr -f901 -l 910 -q \
    qrels.wsjeval -o . wsjeval_broker

java webtrieve.eval.baseline  -f901 -l 910 -q   \
    qrels.wsjeval -o . wsjeval_broker
```

Estes programas criam nos directorios apropriados da fonte os ficheiros *wsjeval_broker.rbr* e *wsjeval_broker.orbr*.

5. Por fim, é necessário usar o programa *seval* que cria os ficheiros $wsjeval\_broker - sel - rbr.evl$ e $wsjeval\_broker - sel - orbr.evl$ com as medidas usadas para a avaliação. Como entrada, utiliza os ficheiros criados anteriormente

```
java webtrieve.eval.seval  -f901 -l 910 -q \
 qrels.wsjeval -e wsjeval_broker.sel  \
   -o. wsjeval_broker
```

As métricas e medidas que foram concretizadas estão explicadas em detalhe no capítulo da selecção de BDs.

7.12 Avaliação de eficiência

Não foi uma preocupação ao longo deste trabalho a optimização dos algoritmos e técnicas utilizadas no sentido de minimizar espaço e tempo necessário para sua execução. Este facto é aliás confirmado pela utilização dum única máquina para manter as diferentes BDTs, o EI e mesmo o cliente. Este ambiente de experimentação contraria, em termos de desempenho, as premissas tomadas como base para a distribuição. Embora o programa já tenha corrido em várias plataformas, é bastante mais simples ter todo o sistema a funcionar na mesma máquina. Como nesta tese o objecto principal de estudo é a eficácia e não a eficiência, esta opção não comprometeu de forma nenhuma o trabalho a realizar embora com a penalização resultante dos tempos de execução.
Nesta secção, faz-se uma abordagem preliminar à avaliação da eficiência dos principais componentes do sistema. Esta abordagem, embora preliminar, tem como objectivo apresentar uma metodologia adequada para avaliação da eficiência dum sistema distribuído de RI. O objectivo fundamental é mostrar como se avalia a eficiência e obter alguns resultados preliminares. Vamos concentrar-nos para já nos recursos em disco, memória e tempos de processamento. Esses recursos serão avaliados para a BDT centralizada e também para o EI.
Para avaliar a eficência dum sistema distribuído com muitos clientes, servidores e redes é necessário compreender e definir a carga de trabalho a que vai estar submetido [MA98].

É muito difícil caracterizar cargas de trabalho em sistemas distribuídos reais. Há um grande número de parâmetros em jogo! Assim, é necessário simplificar e capturar as características mais importantes da carga de trabalho. Essas características mais importantes dependem naturalmente do objectivo do estudo a realizar.
Embora cada sistema precise duma abordagem específica para analisar e especificar a sua carga, há um procedimento genérico que se adequa a todos os estudos. Os passos a seguir são [MA98]:

1. Especificação do ponto de vista segundo o qual a carga deve ser analisada;
2. Escolha dos parâmetros que capturam as características mais importantes da carga para o estudo a realizar;
3. Monitorização do sistema para obtenção de históricos de desempenho;
4. Análise e síntese dos dados de desempenho; e,
5. Construção do modelo de carga.

Por se tratar dum sistema de RI, o tipo de estímulos a que está submetido são interrogações, pedidos de sumários, pedidos de documentos e actualizações dos seus índices. Importa pois estabelecer a que quantidade de transacções e a que ritmo de chegada, vai estar submetido cada componente do sistema no período de tempo em estudo. Para além disso, existe um outro conjunto de operações que são feitas em tempo de indexação ou colecta de documentos. Num sistema distribuído de RI, os componentes mais importantes são os clientes, BDTs, EIs e robots.
Neste trabalho, o estudo de desempenho que se realizado. é preliminar com o objectivo simples da monitorização do sistema para obtenção de algumas medições. Essas medições permitem dar uma ideia da do desempenho do sistema nas principais operações e detectar eventuais pontos de estragulamento. Desta forma pode-se afinar a eficiência do Webtrieve. Não se vão estudar, para já, os ritmos de chegada dos diferentes comandos e como eles se misturam [CM00]. Isto será feito em trabalho futuro.

7.12.1 Ambiente utilizado

A máquina utilizada é servidor SUN Enterprise 250, com um único processador (UltraSPARC-II a 296MHz), o sistema operativo Solaris (SunOS 5.6) e 512 Mbytes de memória central.

O servidor tem 1 disco de 9Gb e 2 discos de 18Gb com um tempo médio de posicionamento de 9 mseg e uma velocidade de transferência de 15.2 MB/seg.

A versão da máquina virtual Java utilizada é a Java 1.3.

Esta informação pode ser retirada da sequência de comandos e respectivas respostas a seguir transcritas.

```
polo% dmesg |more
SunOS Release 5.6 Version Generic_105181-26 [UNIX(R) System V Release 4.0]
Copyright (c) 1983-1997, Sun Microsystems, Inc.
mem = 524288K (0x20000000)
avail mem = 516579328
Ethernet address = 8:0:20:a2:89:58
root nexus = Sun (TM) Enterprise 250 (UltraSPARC-II 296MHz)
....

PCI-device: scsi@3,1, glm #1
pci1000,f1 is /pci@1f,4000/scsi@3,1
sd0 at pci1000,f0: target 0 lun 0
sd0 is /pci@1f,4000/scsi@3/sd@0,0
        <SUN18G cyl 7506 alt 2 hd 19 sec 248>
sd6 at pci1000,f0: target 6 lun 0
sd6 is /pci@1f,4000/scsi@3/sd@6,0
sd7 at pci1000,f0: target 8 lun 0
sd7 is /pci@1f,4000/scsi@3/sd@8,0
...

polo%java -version
java version "1.3.1"
Java(TM) 2 Runtime Environment, Standard Edition (build 1.3.1-b24)
Java HotSpot(TM) Client VM (build 1.3.1-b24, mixed mode)
```

7.12.2 Abordagem utilizada

Um dos problemas que foi preciso contornar foi o facto do protótipo ter sido concretizado em Java. Por este facto, todos os vários programas concretizados correm sobre a máquina virtual Java [LY99]. Como a máquina virtual Java é por si só um ambiente com múltiplos fios de execução é necessário distinguir entre os recursos gastos pela própria JVM e pelo programa. Uma forma possível para se obter estes dados é a inclusão de código de monitorização para esse efeito. A abordagem é boa se se tratar dum programa com um único fio de execução numa linguagem compilada bastar-nos-ia colocar pedaços de código dentro do próprio programa. No caso presente e quando existirem vários fios de execução, é necessário distinguir entre os recursos gastos por cada um deles.

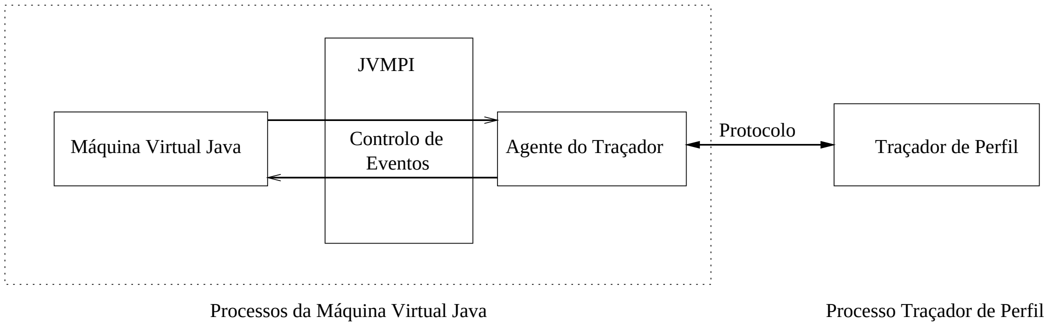


Figura 47: Arquitectura de traçador de perfis da máquina Virtual Java

Por este facto, resolveu-se usar a JVMPI [VL00], que permite traçar o perfil de utilização de recursos dum programa Java. Trata-se simplesmente de um mecanismo que permite obter informações directamente da máquina virtual Java, com base na notificação da ocorrência dum conjunto de eventos pré-definidos (entrada e saída de métodos, alocação de memória, *garbage collection*, etc...).

A JVMPI, cuja arquitectura está apresentada na Figura 47, tem como principal objectivo criar uma interface genérica e transportável para a máquina virtual Java. A ideia é suportar os mais variados tipos de necessidades de levantamento de perfis e também poder ser usada em várias concretizações da máquina virtual Java.

Os autores da JVMPI [VL00] sugerem que se oferece como granularidade mínima o método

```
   polo% java -Xrunhprof:help
         Hprof usage: -Xrunhprof[:help]|[:<option>=<value>, ...]

Option Name and Value   Description              Default
---------------------   -----------              -------
heap=dump|sites|all     heap profiling           all
cpu=samples|old         CPU usage                off
format=a|b              ascii or binary output   a
file=<file>             write data to file       java.hprof(.txt for ascii)
net=<host>:<port>       send data over a socket  write to file
depth=<size>            stack trace depth        4
cutoff=<value>          output cutoff point      0.0001
lineno=y|n              line number in traces?   y
thread=y|n              thread in traces?        n
doe=y|n                 dump on exit?            y

Example: java -Xrunhprof:cpu=samples,file=log.txt,depth=3 FooClass
```

Figura 48: Opções do HPROF

pois, segundo eles, os programadores facilmente distribuem os custos pelas várias instruções do mesmo método. Para além disso, é possível distinguir as invocações dos métodos em fios de execução diferentes, permitindo assim distinguir o respectivo tempo de processador e alocação da *heap*. A *heap* é importante porque tem associado o mecanismo de *garbage collection*. Há, para além disso, os custos de monitores que são usados para sincronização em Java. Importa também reportar a possibilidade de obtenção dos dados de perfil por amostragem estatística diminuindo a carga da monitorização mas com os custos de precisão associados.

Como o HPROF [VL00] é um traçador de perfis já disponível que faz praticamente tudo o que se pretende, optou-se pela sua utilização como ponto de partida. O HPROF está escrito em C.

O HPROF permite obter uma série de informações da máquina virtual Java de acordo com opções indicadas no arranque da máquina virtual (ver Figura 48). A opção da sua utilizaąo foi de, pelo facto de permitir uma monitorização genérica de eventos, se ajustar quase perfeitamente às nossas necessidades. As várias hipóteses para os parâmetros são explicados na própria Figura 48.

Nas Figuras 49, 50 e 51 apresentam-se exemplos de ficheiros de registo gerados pelo HPROF.

Na Figura 49, é apresentado o número de vezes que o método é chamado, o método chamador e o tempo de CPU e de corrida. Na Figura 50 apresentam-se os *traces* com uma pilha de métodos que são registados e a respeito dos quais se apresentam dados de tempos de processamento e de alocação de memória. Os dados de alocação de memória estão apresentados na Figura 51, com número de objectos vivos e alocados e uma pontuação atribuída aos pontos de alocação (*traces*).

Um dos problemas com que nos confrontamos foi o facto de, em termos de tempos, a única informação disponibilizada pelo HPROF ser o tempo de processamento (tempo de CPU) e não haver nenhuma colecta de dados de tempo total de corrida de programa [7]. Para esse efeito, tiveram que ser feitas algumas alterações na biblioteca dinâmica disponibilizada. Foram alterações pontuais no sentido de colectar para cada método para além do tempo de CPU o tempo de corrida. Uma outra modificação que teve que ser realizada foi na forma de apresentação dos resultados.

Há alguns problemas na obtenção de tempo de CPU e de corrida de programa num ambiente com vários fios de execução. Se existirem vários CPUs, o tempo de CPU pode ser superior ao tempo de corrida do programa. Há para além disso falta de precisão na forma como é distribuído o tempo de CPU total pelos vários fios de execução pela máquina virtual Java. Por limitações de precisão dos valores de tempo de CPU devolvidos pela máquina virtual Java, os tempos obtidos só são fiáveis a partir de dezenas de microsegundos. Como nos importa, para já, colectar dados de processamento das operações principais, com tempos de duração da ordem dos milisegundos, optamos por não fazer mais modificações no HPROF ou usar outro programa. Os registo de tempo de CPU e de corrida são apresentados na Figura 49.

Nas Figuras seguintes apresenta-se uma pontuação de acordo com percentagem de ocupação da heap, em que os pontos de alocação (método e linha onde ocorreu uma criação de objecto) são identificados por um número de *trace* na Figura 51 e depois a pilha de invocações para cada um desses traces na Figura 50.

Os dados assim apresentados permitem apenas identificar pontos críticos na alocação de memória. De qualquer forma, os mesmos dados colectados podem ser manipulados e apresentados doutra forma mais adequada à avaliação da eficiência. Podem, por exemplo, agrupar-se todas as alocações efectuadas por método Java, obtendo assim uma utilização de memória com a mesma granularidade que se obtém na medição do tempo de execução.

[7] *elapsed time*, na terminologia inglesa

```
 count callee caller time elapsed_time
157978 java/lang/Character.toLowerCase(C)C java/lang/String.toLowerCase(Ljava/ut
il/Locale;)Ljava/lang/String; 680 786
48990 java/io/ByteArrayInputStream.read()I java/io/DataInputStream.readUnsignedB
yte()I 250 273
48990 java/io/DataInputStream.readUnsignedByte()I java/io/DataInputStream.readUT
F(Ljava/io/DataInput;)Ljava/lang/String; 490 577
22272 java/lang/System.arraycopy(Ljava/lang/Object;ILjava/lang/Object;II)V java/
lang/String.getChars(II[CI)V 110 142
19741 java/lang/String.getChars(II[CI)V java/lang/StringBuffer.append(Ljava/lang
/String;)Ljava/lang/StringBuffer; 220 280
16094 java/lang/String.toLowerCase(Ljava/util/Locale;)Ljava/lang/String; java/la
ng/String.toLowerCase()Ljava/lang/String; 1100 1161
```

Figura 49: Tempos de CPU e de corrida devolvidos

```
 TRACE 1:
        <empty>
TRACE 3530:
        java/util/Vector.contains(Vector.java:Unknown line)
        java/lang/ClassLoader.loadLibrary0(ClassLoader.java:Unknown line)
TRACE 610:
        java/lang/Class.newInstance0(Class.java:Native method)
        java/lang/Class.newInstance(Class.java:Compiled method)
         ...
        java/net/URLClassLoader.defineClass(URLClassLoader.java:Compiled method)
        java/net/URLClassLoader.access$1(URLClassLoader.java:Compiled method)
        java/net/URLClassLoader$1.run(URLClassLoader.java:Compiled method)
TRACE 2513:
        webtrieve/metadata/MetaInformationRecord.<clinit>(MetaInformationRecord.java:Unknown line)
        ....
        webtrieve/metadata/MetaData.readMtdRecordAt(MetaData.java:Compiled method)
        webtrieve/starts/AConnection.run(aloneServer.java:Compiled method)
TRACE 2409:
        java/math/BigInteger.remainder(BigInteger.java:Compiled method)
        ...
```

Figura 50: Sites para os traces do HPROF

```
HEAP DUMP END
SITES BEGIN (ordered by live bytes) Wed Jun 20 23:55:17 2001
          percent          live       allocated  stack class
 rank   self  accum    bytes objs   bytes objs trace name
    1  6.79%  6.79%    16388    1   16388    1   106 [C
    2  6.79% 13.58%    16388    1   16388    1    97 [C
    3  4.82% 18.40%    11628    3   11628    3   291 [C
    4  3.40% 21.79%     8196    1    8196    1   104 [B
    5  3.40% 25.19%     8196    1    8196    1    95 [B
    6  3.34% 28.53%     8068    1    8068    1   294 [S
    7  3.01% 31.54%     7256    4    7256    4   288 [C
    8  1.91% 33.45%     4620   33    4620   33   681 java/lang/Class
    9  1.33% 34.79%     3220   23    3220   23   985 java/lang/Class
   10  1.29% 36.08%     3120   98    3120   98   791 [C
   11  1.28% 37.36%     3080   22    3080   22  1335 java/lang/Class
   12  1.12% 38.47%     2692   45    2692   45   600 [C
   13  1.05% 39.52%     2532    1    2532    1   295 [I
```

Figura 51: Sites com ocupação da HEAP

Por forma a alterar o formato dos dados disponibilizados pelo HPROF, houve que introduzir algumas alterações no HPROF para processar os ficheiros de registos históricos, para além, claro, duma metodologia complementar manual para obtenção dos valores pretendidos.

Um dos passos prévios para obtenção dos resultados experimentais desejados é a identificação dos métodos que no pacote Webtrieve concretizam as diversas operações desejadas. Posteriormente e para cada caso, são processadas as linhas dos ficheiros de registos históricos correspondentes.

Começaremos por identificar os métodos associados às operações principais realizadas no Webtrieve, que podem ser vistas na Figura 40, apresentada no Capítulo 7. Essas operações estão apresentadas na Tabela 45. A colecta do documento é realizada pelo robot, embora seja um processo simples em ambiente laboratorial. O reconhecimento do documento, pode ser feito usando ou não a remoção de palavras comuns e com ou sem radicalização. No que toca à alocação de documentos é necessário distinguir entre a existência ou não de replicação.

Uma vez os documentos reconhecidos são criados e os respectivos ficheiros com documentos invertidos (extensão dxi). Os documentos são alocados pelas diferentes BDTs e passa-se à

Operação	Método(s) Chamado(s)	Método Chamador
Colecta de documento	webtrieve.robot	
Reconhecimento de documento com ou sem stem com ou sem stop	parser.parse.parseFile	basic.WebtrieveSourceManager./ doParse
Alocação de documentos sem replicação com replicação	allocator.ObjectAllocator.allocateIt	basic.WebtrieveSourceManager./ doAllocate
Criação do ficheiro ivertido	inversion.dxi2inv./ doInvert	basic.WebtrieveSourceManager./ doInvert
Estatísticas para normalização	LengthNormalizer./ computeNormalizationFactors LengthNormalizer./ dumpResults	basic.WebtrieveSourceManager ./ doLoadMetadata
Estatísticas para dimensionamento buckets e chunks	InvStats.computeStats InvStats.processHistogram	basic.WebtrieveSourceManager ./ doCumputeInvStats
Geração de sumários	summmarizer.sumit.doSummarize	basic.WebtrieveSourceManager./ doSummarize
Construção do dicionário	loader.DictionaryLoader.addPath	basic.WebtrieveSourceManager./ .doLoadDictionary
Construção da meta-informação	loader.MetaData.addPath	basic.WebtrieveSourceManager./ doLoadMetadata

Tabela 45: Métodos invocados para construção da fonte

Operação	Método(s) Chamado(s)	Método Chamador
Carregamento do dicionário	textdb.DicitionaryOfTerms	starts.webt./ WebtrieveSourceDescription
Carregamento da meta-informação	metadata.MetaData	starts.webt./ WebtrieveSourceDescription
Interrogação BDT centralizada BDT distribuída	starts.AConection.HandleQuery	starts.aloneServer./ handleHTTPRequest
Acesso a páginas	starts. HTTPStartsHandler./ handleGetDocument	starts.aloneServer./ handleHTTPRequest

Tabela 46: Métodos invocados para interacção com o servidor

criação do ficheiro invertido. Este processo é realizado em dois passos, inversão de ficheiros parciais e posterior intercalação desses ficheiros num único ficheiro invertido. A medição é realizada para todo esse processo que cria único ficheiro com extensão inv, com base num conjunto de ficheiros com extensão dxi.

De seguida, processa-se o ficheiro invertido e inclui-se informação suficiente nos registos de meta-informação de cada documento da colecção para sua posterior normalização em tempo de interrogação. O tempo gasto nestes cálculos está apresentado na linha correspondente a estatísticas de normalização.

BDTs ou EIs que sejam indexados por um nível superior precisam de exportar seus sumários. Esse sumário é gerado com base no ficheiro invertido. O tempo de geração desse sumário também esta incluído no tempo de criação duma BDT componente.

As estruturas de dados mais importantes tanto do EI como da BDT é o dicionário de termos e a base de dados de meta-informação. Os tempos seguintes referem-se à criação das estruturas de dados usadas pelo dicionário, nomeadamente os ***chunks***, ***buckets*** e os respectivos índices.

Na tabela 46 comeca-se por apresentar o tempo de carregamento das estruturas do dicionário, em tempo de arranque. O mesmo tratamento é dado aos registos de meta-informação para os quais é necessário manter também índices em memória central. São ainda registados os tempos correspondentes aos principais tipos de interacções com o servidor: a interrogação (para BDT centralizada ou distribuída) e também a transferência de páginas estáticas.

O passo seguinte é a identificação das operações principais numa BDT para processamento duma interrogação. A Figura 52 apresenta sumariamente o esquema duma BDT. Nesse esquema prioriza-se a interacção com o exterior nomeadamente as interrogações que chegam com determinado ritmo e também actualizações dos documentos indexados. A BDT processa cada pedido e devolve a respectiva resposta. São apresentadas filas de espera para chegada de interrogações e actualizações e para envio de respostas. Nestas filas de espera são colocados os pedidos pendentes a processar pelo engenho ou pelo actualizador e as respostas para envio pelos representantes locais dos clientes. A existência de filas permite atrasos no processamento de pedidos e no envio das respostas por sobrecarga da rede.

O tempo de processamento de cada interrogação pode envolver a sua expansão e depende fundamentalmente da leitura das listas de ocorrência, do pré-processamento da interrogação e da ordenação dos documentos da resposta. São esses tempos parciais cujos métodos estão identificados na Tabela 47.

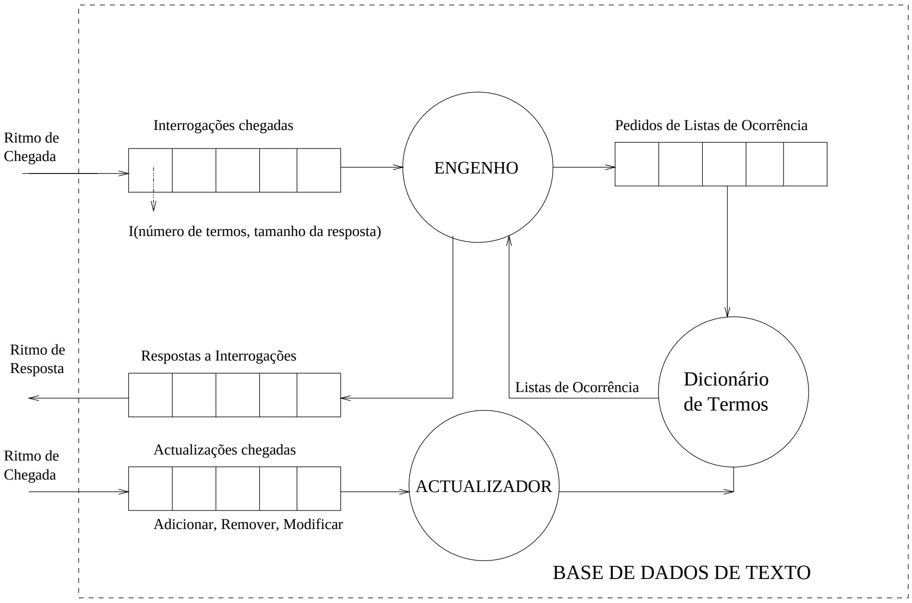


Figura 52: Avaliação da eficiência da BDT

Relativamente ao EI, está identificado o tempo de selecção que inclui os tempos componentes similares aos indicados na BDT, e posteriormente o tempo de difusão da interrogação para cada BDT componente seleccionado, o tempo de espera e o tempo de fusão das respostas parciais. Os vários tempos parciais estão apresentados na Tabela 48. Um esquema simplificado do EI orientado para as suas interacções com o exterior, está apresentado na Figura 53. O modelo inclui filas de espera para envio e recepção na interacção com os clientes ou com as BDTs.

O tempos envolvidos, do ponto de vista do cliente, estão ilustrados na Figura 54. Os vários tempos de interrogação da BDT ou EI terão que ser afectados com tempos de propagação do cliente ao servidor e vice-versa. Digamos que são estes os tempos que medem a eficiência do sistema do ponto de vista do utilizador.

Para identificar a memória central gasta na concretização das várias operações importa identificar as estruturas de dados correspondentes e o objecto ou método Java onde essas estruturas são mantidas. Com base nisso, e usando a identificação do ponto de aloacação

Operação	Método(s) Chamado(s)	Método Chamador
Pre-rocessamento da interrogação	starts.aloneServer.QueryPreProcess	starts.aloneServer./ HandleQuery
Ordenação de termos da interrogação	engine.Ranking./ sort	engine.Ranking./ processTerms
Leitura de listas de ocorrência	engine.Ranking./ processTerms	engine.Ranking./ processRanking
Ordenação de documentos	query.results.SortResults	aloneServer.HandleQuery
	engine.AccumulatorTable./ topRanked	engine.Ranking./ processRanking
Acesso a Meta-informação	engine.AccumulatorTable./ topRanked.MetaInformationRecord./ getMetaInformationRecord	engine.Ranking./ processRanking
Expansão da interrogação	query.expansion	aloneServer.HandleQuery

Tabela 47: Principais métodos usados para processamento da interrogação na BDT

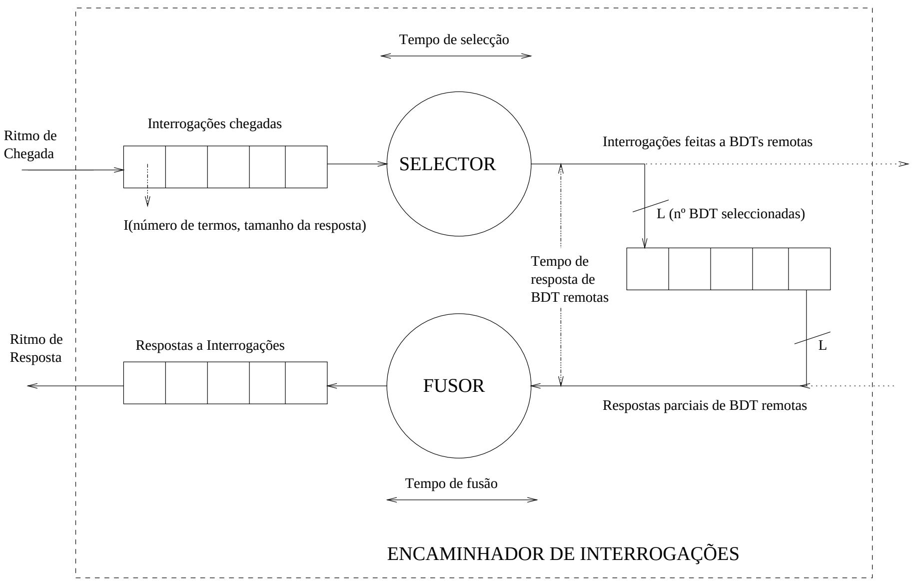


Figura 53: Avaliação da eficiência do EI

Operação	Método(s) Chamado(s)	Método Chamador
Selecção das BDTs	broker.BrokerRankedFilter./ select	.stars.aloneServer./ query.source.DoSourceSearch
Fusão das respostas	broker.BrokerRankedFilter./ fusion	.stars.aloneServer./ query.source.DoSourceSearch
Interrogação de BDT remotas	broker.BrokerRankedFilter./ chain	.stars.aloneServer./ query.source.DoSourceSearch
Expansão da interrogação	query.expansion	aloneServer.HandleQuery

Tabela 48: Principais métodos usados para processamento da interrogação no EI

(ver Figura 50), pode-se obter o total da *heap* ocupada e calcular a percentagem em relação ao total para cada método, bem como a quantidade de objectos e de bytes alocados e instanciados (ver Figura 49). Para além dos objectos mais genéricos onde são mantidas as principais estruturas de dados, importa registar o número de instâncias e objectos criados para ter uma ideia como se distribuem os gastos pelos vários tipos de objectos.

7.12.3 Resultados obtidos

Uma vez explicada a metodologia e os programas usados para avaliação da eficiência, passemos à apresentação de resultados. Um parâmetro importante na avaliação dos resultados é o tamanho da entidade que vai ser objecto de processamento. No que toca ao pré-processamento de cada documento e mesmo da colecção, interessa o respectivo tamanho expresso quer em número de documentos quer em Kbytes ou Megabytes. Quando se trata de colecção distribuída, também é relevante o número de BDTs e se existir replicação, o nível de sobreposição pretendido. Quando se processam interrogações, é importante saber o número de termos a processar. Os tempos podem ser apresentados para além dos valores absolutos, médias por unidade (M/U) para além das médias por invocação.

Apresentam-se na Tabela 50 algumas estatísticas e objectos persistentes para o cenário utilizado para avaliação da eficiência. Usou-se uma colecção pequena para viabilizar a invocação de vários processos da JVM que foi invocada em alguns casos com a biblioteca dinâmica derivada do HPROF. Tratou-se duma medida de precaução pelo facto de se usar uma única máquina para simular o ambiente distribuído. A invocação dos métodos do HPROF são muito intrusivos e aumentam bastante o tempo de corrida dos vários programas. Apresentam-se números para a BDT centralizada, EI e valores médios para as BDT

Operação	Estrutura de Dados	Objecto/Método
inversão de documento	basic.Text.MetasInformationRecord basic.Text.TermFrequency	basic.Text
inversão de ficheiro	inversion.dxi2inv.TextSet inversion.inversion.TextSet	inversion.dxi2inv
cálculo de estatísticas	basic.InvStats.int[]	basic.invStats
cálculo de sumário	summarizer.sumit./ WebtFileInputStream WebtFileOutputStream	summarizer.sumit
construção do dicionário	loader.DictionaryLoader./ DictionaryOfTerms	loader.DictionaryLoader
construção da meta-informação	loader.DictionaryLoader./ metadata.Metadata metadata.MetaIndex	loader.metaloader
índices de chunks		
processamento da interrogação	engine.Ranking./ AccumulatorTable	engine.Ranking engine.BrokerRanking
gestão de buckets	textdb.utils./ ChunkManager	textdb.utils./ ChunkManager
gestão de chunks	textdb.utils./ BucketManager	textdb.utils./ BucketManager
índices dos chunks		
índices da meta-informação		
carregamento do dicionário de termos	basic.DictionaryOfTerms./ DictionaryMainIndex ChunkManager BucketManager	basic.DictionaryOfTerms./

Tabela 49: Métodos e/ou objectos responsáveis pela alocação de memória

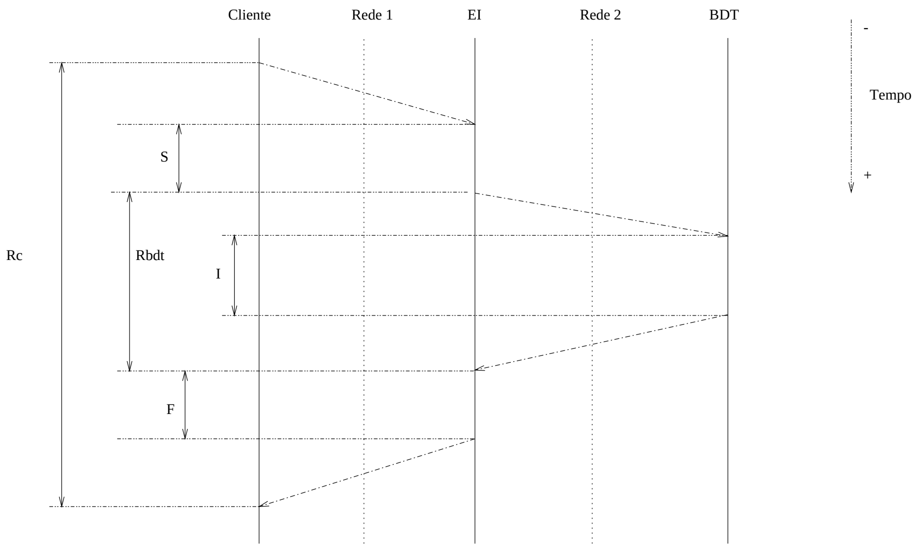


Figura 54: Avaliação da eficiência do lado do cliente

componentes. Usaram-se um subconjunto dos ficheiros de um dos anos da colecção Wall Street Journal da TREC.

No tamanho de alguns ficheiros pelo facto de se registado apenas a versão usando a linguagem de marcas DIMF (para aumentar a sua legibilidade) não é explícita a compactação conseguida com o processo de indexação. Essa redução de tamanho pode ser vista apenas nos ficheiros compactados.

O processo de alocação aleatório dividiu equatatiavemente os documentos pelas BDTs componentes. A soma dos documentos nas várias BDTs é superior ao número total de documentos originais pela existência de replicação. Os documentos registados para o EI são as BDTs.

Os indíces da meta-informação são mantidos em memória central para uma rápida localização da metadata dum dado documento dado o seu número. Mantêm simplesmente o ficheiro onde se encontra a meta-informação para uma gama de número de documentos.

Os índices do dicionário indicam os termos a que correspondem listas de ocorrências longa e a localização da respectiva lista de *chunks*.

Também se pode observar na tabela que embora haja um bom aproveitamento do espaço dos

Item	BDT Centralizada		EI		Média das BDTs Componentes	
	Plano	Compact.	Plano	Compact.	Plano	Compact.
Tamanho da colecção	9821	3505				
N°documentos		2947		8		478
N°termos		31750		31750		12154
N°registos de ocorrência		438479		97323		68778
Tamanho de total de documentos invertidos	12553	2137			319	2040
Tamanho do ficheiro invertido	10656	1416	9541	582	1685	263
Tamanho do ficheiro de meta-informação	1848	184	6	1	314	31
Tamanho de índices para meta-informação	93	51	1	1	15	9
Número de buckets		236		564		73
Tamanho do bucket		8192		8192		8192
Tamanho ficheiro de buckets	1904	616	4288	592	601	177
% Ocupação no ficheiro de buckets		96,8		92,3		98,5
Número de chunks		1801		1061		660
Tamanho do chunk		8192		8192		8192
Tamanho ficheiro de chunks	14424	840	8424	70	5402	107
% Ocupação no ficheiro de chunks		16,7		3,5		5,0
Tamanho de índices para dicionário	23	59	11	37	8	22
Tamanho do sumário					96	1411

Tabela 50: Algumas estatísticas e recursos de disco necessários para objectos persistentes planos e compactados do cenário usado para testes de eficiência

buckets, o ficheiro de chunks é bastante desaproveitado, apresentando índices de ocupaąo abaixo dos 10 %. Trata-se dum aspecto que vai ser objecto de trabalho futuro e em que deverão ser avaliadas diferentes políticas para os chunks face ao compromisso a estabelecer entre ritmos de actualização e interrogação e o desempenho do dicionário nessas operações. Uma heurística particular deve ser estudada para o EI, definida em função do número de BDTs indexadas.

Para além do cenário atrás apresentado, com uma BDT centralizada e uma BD distribuiída usaram-se as interrogações correspondentes aos tópicos 151 a 161 do TREC. O número de interrogações foi também limitado pelas razões atrás explicitadas.

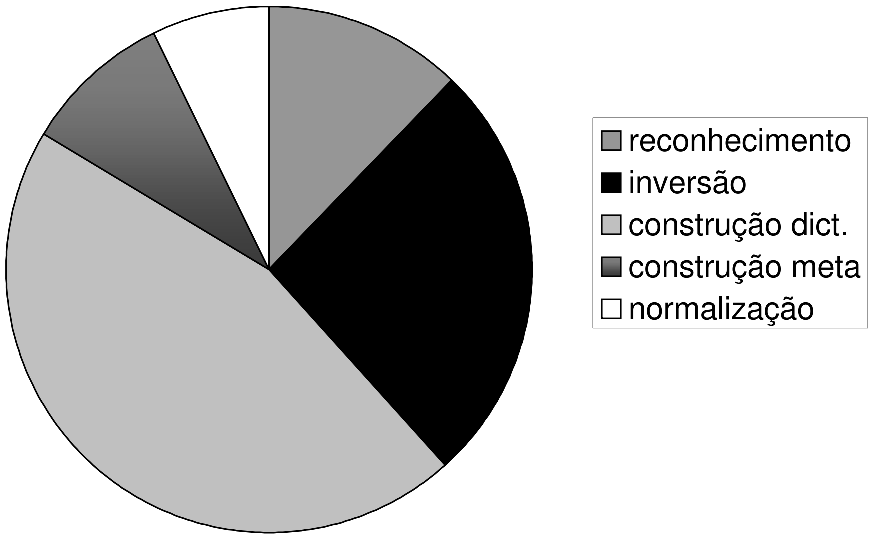
Um problema a resolver tem a ver com que tempo usar para registar o tempo de corrida dos vários programas. Como seria de prever o tempo obtido com o traçador de perfis derivado do HPROF devido à invocação dos métodos da biblioteca de monitorização introduz atrasos substanciais. So para exemplificar o tempo médio de interrogação ao EI sem HPROF é 20,468 segundos contra 17 minutos invocando a JVM com HPROF! A dúvida que a seguir se coloca é se poderá usar o HPROF para avaliar os tempos de corrida relativos mas o coeficiente de correlação obtido foi de 0,895. Por este facto optou-se por registar os tempos de corrida sem a invocação do HPROF. Isto só é possível porque se faz uma utilização limitada de vários fios de execução neste protótipo. Esta problemática bem como a contabilização do tempo de CPU será objecto de trabalho futuro.

Apresenta-se na Tabelas 51 e 52 os tempos de corrida dos programas para criação das fontes necessárias para o cenário de teste. Os valores obtidos podem ser avaliados quando confrontados com os tamanhos dos ficheiros de informação processados que já foram apresentados anteriormente. A construção do EI é muito mais rápida (cerca de um terço do tempo) que a construção da BDT centralizada, por duas razões fundamentais. O processo de inversão no EI é bastante agilizado pelo facto do ficheiro invertido ser construído com base no sumário gerado pelas BDT componentes. A construção do dicionário é também dependente do número de documentos que diminui sobremaneira e não se está a incluir o tempo de reconhecimento dos documentos, já efectuado para a BDT centralizada.

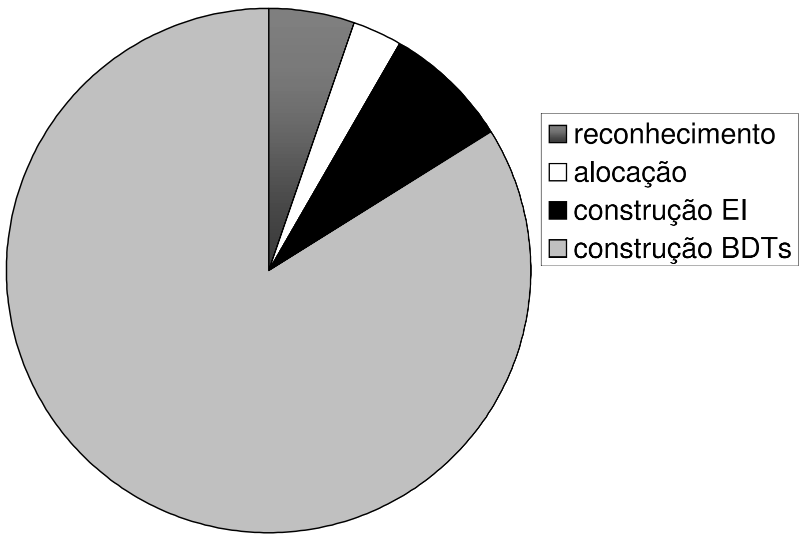
Na Figura 55 dá-se uma imagem visual de como é partilhado o tempo de construção pelas várias operações na construção duma BDT centralizada e duma forma mais grossa numa BDT distribuída. Neste último caso, se se usasse um ambiente distribuído real, o tempo de construção das várias BDT componentes poderia ser dividido pelas máquinas em presença. Na construção da BDT centralizada que dá uma ideia também da construção de cada componente duma BDT distribuída, a maior parte do tempo é usado para construção do dicionário, metadata e para a inversão. Isto para além do tempo de reconhecimento e do cálculo das estatísticas de normalização.

Avaliado o tempo de construção das BDTs passemos a olhar para o processamento das interrogações. Também se registaram tempos para a BDT centralizada e distribuída.

Na Tabela 53 disponibilizam-se os valores em segundos das operações no EI. Para detecção de possível relação está indicado o número de termos em cada interrogação. Uma imagem visual da distribuição de tempos está apresentada na Figura 57. O tempo de selecção mantêm-se bastante baixo comparativamente aos restantes. Este tempo deve ser determinado quer pelo tamanho da interrogação (número de termos) e tamanho das respectivas



(a) Centralizada



(b) Distribuída

Figura 55: Partilha do tempo de processamento entre as principais operações na construção das BDTs

Operação	**Tempo (s)**
Reconhecimento	269
Inversão	577
Construção do dicionário	995
Construção da meta-informação	201
Normalização	161
Total	**2202**

Tabela 51: Duração das principais operações na construção da BDT centralizada

Operação ou Componente	**Tempo (s)**
Reconhecimento	269
Alocação	145
Construção do EI	385
Construção de 8 BDTs	4192
Total	**4991**

Tabela 52: Duração das principais operações na construção da BDT distribuída

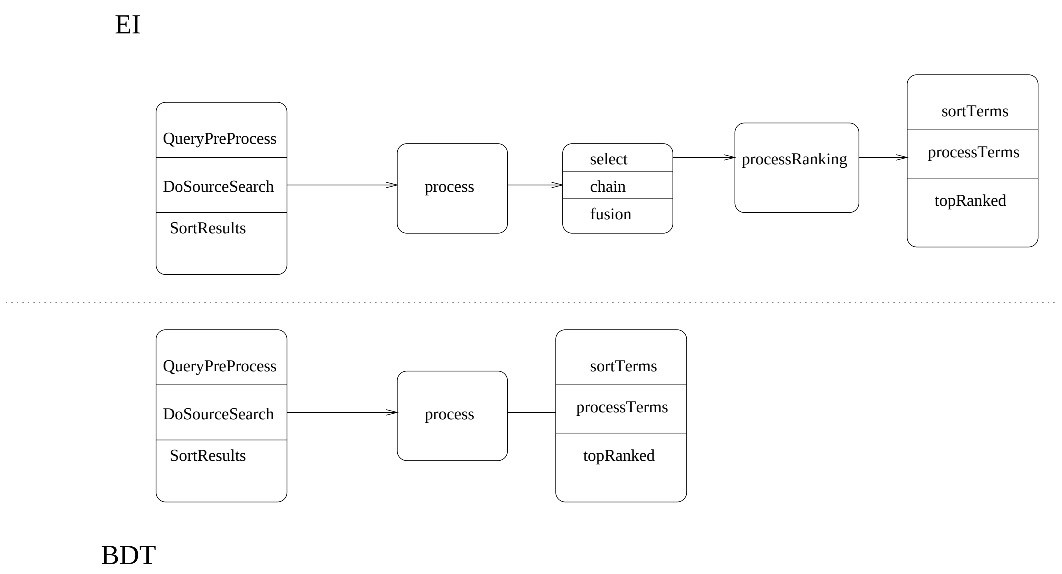


Figura 56: Aninhamento dos métodos analisados no EI e na BDT

Nºda Interrog.	151	152	153	154	155	156	157	158	159	160	161	Média
Nºde Termos	24	45	18	35	44	33	48	43	24	25	45	33,9
Tempo Cliente	41,8	29,9	9,4	23,6	38,7	10,3	17,2	16,3	6,1	47,6	12,4	24,0
Tempo EI	40,0	27,7	8,6	22,4	37,9	9,5	16,3	15,4	5,6	46,9	11,7	23,8
Transferência	1,91	1,30	0,79	1,21	0,83	0,79	0,91	0,89	0,47	0,69	0,72	0,98
Pré-processam.	0,33	0,11	0,04	0,08	0,11	0,76	0,11	0,17	0,05	0,06	0,09	0,11
Selecção	2,35	1,32	1,02	1,35	1,73	0,84	1,05	1,04	0,53	0,63	0,85	1,21
Difusão	21,7	14,2	5,14	8,87	12,9	6,78	8,66	8,10	4,5	11,0	6,9	10,2
Fusão	15,2	12,0	2,3	12,0	23,1	1,76	6,42	6,06	4,71	35,1	3,79	11,5

Tabela 53: Tempos parciais em segundos para processamento da interrogação no EI

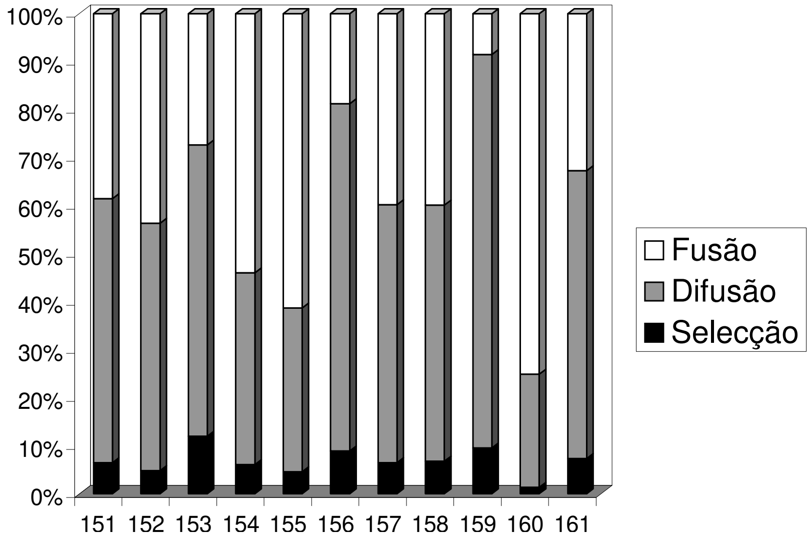
Nºda Interrog.	151	152	153	154	155	156	157	158	159	160	161	Média
Nºde Termos	24	45	18	35	44	33	48	43	24	25	45	33,9
Tempo Cliente	7,07	3,98	3,35	3,57	2,04	1,55	2,41	1,52	1,67	1,63	1,82	2,88
Tempo BDT	4,98	2,64	1,77	2,02	1,20	0,80	1,63	0,68	0,89	0,58	0,85	1,72
Transferência	2,09	1,34	1,58	1,56	0,85	0,75	0,79	0,84	0,78	0,11	0,98	1,16
Pré-processam.	0,35	0,12	0,06	0,18	0,11	0,16	0,17	0,09	0,13	0,06	0,08	0,14
Ord. termos	0,36	0,34	0,04	0,27	0,33	0,09	0,32	0,05	0,12	0,05	0,08	0,20
Process. LO	0,80	1,09	0,89	0,87	0,44	0,19	0,64	0,20	0,30	0,13	0,29	0,55
Docs. topo	1,87	0,88	0,49	0,45	0,25	0,14	0,35	0,29	0,28	0,26	0,19	0,52

Tabela 54: Tempos parciais para processamento da interrogação na BDT

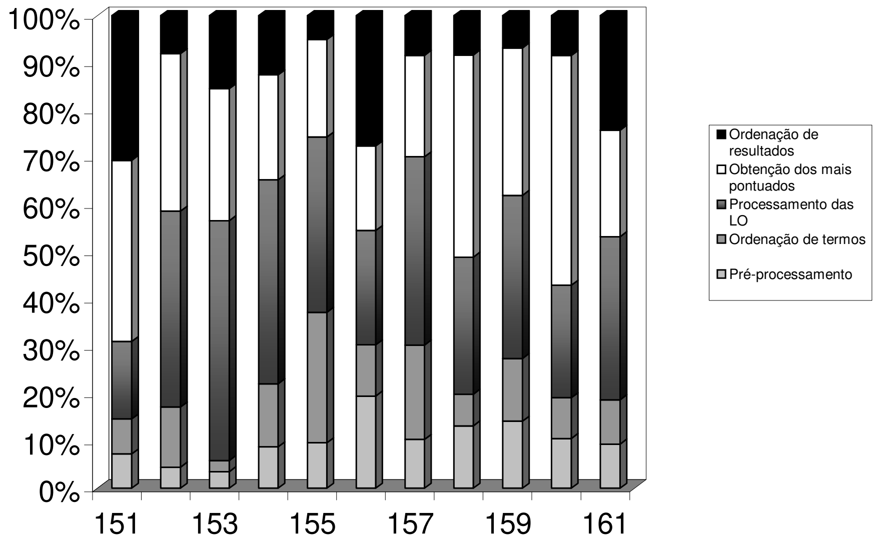
listas de ocorrência que precisam de ser lidas de disco durante o processamento. É também influenciado pelo número de BDTs elegíveis. Quer o tempo de fusão como o de difusão são bastante grandes e da mesma ordem de grandeza. Como o tempo de difusão é determinado pelos tempos de resposta das BDT componentes, resta dizer que o algoritmo de fusão necessita de mais trabalho. Esta constação pode ser feita com base no algoritmo que depende do número de BDTs a fundir e também dos documentos em presença.

Segue-se uma análise dos tempos parciais no processamento duma interrogação na BDT. Nota-se um tempo de transferência demasiado grande sendo as porções de tempo mais significativas dispendidas no processamento das listas de ocorrência e na ordenação da lista de acumuladores para obtenção dos documentos mais pontuados.

Uma outra indicação que vai no sentido da necessidade da afinação e remoção de alguns problemas do código é o facto de não haver uma boa correlação entre os tempos das várias operações e das variáveis de que teoricamente deviam depender.



(a) EI



(b) BDT

Figura 57: Partilha do tempo entre as principais operações para processamento da interrogação

Relativamente a ocupação de memória principal, apresenta-se nas Figuras 58 59 a quantidade de memória necessária para as principais operações no EI e na BDT.
Existe informação mais pormenorizada para cada método que não é aqui incluída mas que pode ser usada para explicar os gastos em memória nas várias operações. Tomemos como exemplo o EI.
Na fusão, grande parte dos bytes são usados para manter os decimais de vírgula flutuante que mantém a informação de pontuação atribuída aos vários documentos. Esta informação é também usada no processo de fusão para a obtenção das novas pontuações para os documentos a apresentar no resultado final. Os valores da pontuação são obtidos de strings mantidas nos resultados parciais. Existe também um grande número de cadeias de caracteres que mantêm os a restante informação dos documentos que são os componentes dos resultados parciais a fundir.
Na difusão existe uma grande quantidade de objectos do tipo cadeia de caracter que são os buffers usados na comunicação com as BDTs interrogadas. Esses buffers mantêm os SOIFs usados pelo protocolo STARTS para transportar as respostas e os itens de documentos. O reconhecimento desses SOIFs permite a obtenção dos conjuntos de resultados parciais que posteriormente serão objecto de fusão.
Relativamente à selecção existe uma grande quantidade de inteiros das listas de ocorrência processadas, cadeias de bytes lidas de disco como chunks e buckets donde são obtidas as listas de ocorrência. Existem ainda algumas strings para manter os termos em processamento e a meta-informação associada a cada fonte que foi seleccionada.
Este tipo de observações permitem localizar as zonas do código que devem ser trabalhadas para se conseguir poupanças de memória.

7.13 Discussão

O facto de se ter optado pela concepção e concretização dum sistema de raiz teve como vantagem permitir a familiarização com a componente mais prática dos conceitos e algoritmos da RI centralizada. Uma outra vantagem importante é o conhecimento adquirido para tratar problemas como recuperação de texto semi-estruturado, várias línguas, dicionários negativos, radicalizadores, actualizações em linha, estrutura de índices, etc...
Uma outra constatação da concretização é que a maior parte do código Java usado para a BDT foi reaproveitado para o EI. Por este facto grande parte do tempo dispendido na

```
M: METHOD FILTERED DATA [BEGIN]
M: 1: webtrieve.broker.BrokerRankedFilter chain  TRACES=429
M:      TRACE Alloced Alloced-Bytes Live Live-Bytes CLASS
M:        516 230512 303754840   1956 23821024 TOTAL
M:       7047 728128 975974912  32061 71501920 BIG TOTAL
M:        516 31.66%   31.12% 6.10%   33.32% PERCENTAGE
M: 2: webtrieve.broker.BrokerRankedFilter fusion  TRACES=160
M:      TRACE Alloced Alloced-Bytes Live Live-Bytes CLASS
M:        202  87019 556753888    755 31122672 TOTAL
M:       7047 728128 975974912  32061 71501920 BIG TOTAL
M:        202 11.95%   57.05% 2.35%   43.53% PERCENTAGE
M: 3: webtrieve.broker.BrokerRankedFilter select  TRACES=890
M:      TRACE Alloced Alloced-Bytes Live Live-Bytes CLASS
M:       1222 155831 48150720   1058  5808192 TOTAL
M:       7047 728128 975974912  32061 71501920 BIG TOTAL
M:       1222 21.40%    4.93% 3.30%    8.12% PERCENTAGE
```

Figura 58: Alocação de memória no EI às principais operações

concretização da BDT, constituiu na realidade um investimento que facilitou bastante a concretização da BDT. Esta grande percentagem de reaproveitamento justificou a opção pelo Java.

Para além da facilidade de desenvolvimento devido à modularidade e legibilidade imposta pela linguagem Java, foi possível usar uma série de ferramentas de auxílio e geração do código e mesmo para sua validação e teste. O protótipo já foi executado em Linux, Windows NT e o Solaris (que foi usado para o desenvolvimento) com apenas alguns ajustes na interface com o sistema operativo e sistemas de ficheiros. Pode-se com facilidade gerar documentação para o código. Importa realçar também que, com algum esforço, se comentou todo o código em inglês. A utilização do Java também torna simples a introdução de novos módulos para tratar novos problemas ou a substituição de algum módulo já existente que entretanto se considere obsoleto ou desadequado.

Uma outra boa opção foi o STARTS. Para além do acesso ao código do Carl Lagoze, que facilitou bastante o desenvolvimento, colocaram-se logo à partida o tipo de problemas típicos duma BDT distribuída que foram tratadas a preceito usando os conceitos, objectos e estruturas de dados adequadas. Em suma, estas vantagens resultam de se ter baseado o desenvolvimento numa proposta de norma.

A estrutura dual do dicionário permitiu dotar o protótipo dos mecanismos necessários para a

```
M: 9: webtrieve.engine.AccumulatorTable topRanked  TRACES=243
M:      TRACE Alloced Alloced-Bytes Live Live-Bytes CLASS
M:        340  19275 15138336    861  2347856 TOTAL
M:       5523 310783 104981416  80389 20477064 BIG TOTAL
M:        340  6.20%   14.42%  1.07%   11.47% PERCENTAGE
M: 11: webtrieve.engine.Ranking processTerms  TRACES=340
M:      TRACE Alloced Alloced-Bytes Live Live-Bytes CLASS
M:        432  79187 35304688   1070  3827976 TOTAL
M:       5523 310783 104981416  80389 20477064 BIG TOTAL
M:        432 25.48%   33.63%  1.33%   18.69% PERCENTAGE
M: 13: webtrieve.metadata.MetaData getMetaInformationRecord  TRACES=137
M:      TRACE Alloced Alloced-Bytes Live Live-Bytes CLASS
M:        220  14160 10870400    362  1817072 TOTAL
M:       5523 310783 104981416  80389 20477064 BIG TOTAL
M:        220  4.56%   10.35%  0.45%    8.87% PERCENTAGE
M: 12: webtrieve.engine.Ranking sort  TRACES=183
M:      TRACE Alloced Alloced-Bytes Live Live-Bytes CLASS
M:        233  28315  8990576    249   764040 TOTAL
M:       5523 310783 104981416  80389 20477064 BIG TOTAL
M:        233  9.11%    8.56%  0.31%    3.73% PERCENTAGE
M: 15: webtrieve.starts.AConnection QueryPreProcess  TRACES=665
M:      TRACE Alloced Alloced-Bytes Live Live-Bytes CLASS
M:        939  10602  2605848   1033   317992 TOTAL
M:       5523 310783 104981416  80389 20477064 BIG TOTAL
M:        939  3.41%    2.48%  1.29%    1.55% PERCENTAGE
```

Figura 59: Alocação de memória na BDT às principais operações realização de actualizações em linha e a possibilidade da sua futura utilização em ambientes dinâmicos como o Web. Foram também criadas as condições para tirar partido duma maior promiscuidade com acesso ao disco, utilização de vários discos, curto-circuitando se necessário o sistema de ficheiros.

Uma outra opção que facilitou o desenvolvimento sem comprometer os objectivos, embora aumentando o tempo das experiências, foi o facto de se usar um único servidor para suportar todo o sistema de RI distribuído. Embora simples, houve uma série de mecanismos e correspondente código necessário para a distribuição que se pôde omitir.

A utilização do TREC para avaliação da eficácia do sistema e dos seus componentes, facilitou sobremaneira a concretização dos módulos de avaliação. Para além das colecções de teste e algum código que foi directamente usado, tomou-se como ponto de partida uma metodologia de avaliação que se pode considerar uma norma de facto e isso possibilitou a comparação directa com outros resultados apresentados na literatura.

A abordagem usada na avaliação da eficiência seguiu também uma metadologia estabelecida, usada em muitos trabalhos reportados na literatura. Embora se tenha feito uma incursão preliminar a esta temática, criaram-se bases sólidas para trabalho futuro.

Na concretização do protótipo, houve que fazer uma série de opções no sentido de conseguir terminar este trabalho em tempo útil, e dar prioridade aos aspectos essenciais da tese.

O cliente STARTS disponibilizado é bastante complexo e precisa, por isso, duma interface muito simples que possa ser usada por utilizadores eventuais. No seu estado actual o cliente está adequado à demonstração das funcionalidades do sistema. Justifica-se a concepção e concretização de um cliente mais amigável que esconda a complexidade do protocolo STARTS, mantendo o máximo das suas potencialidades.

Pelo facto de se usar neste trabalho as colecções de teste TREC, o robot teve menos atenção. É necessário estudar com mais detalhe os aspectos relacionados com a colecta da informação quer no que respeita aos novos documentos quer na actualização dos já existentes. Esta problemática está relacionada com as actualizações em linha dos índices de cada BDT e com os sumários do EI.

Na parte da extracção de informação dos documentos, houve também que estabelecer algumas restrições. Foi preterida a possibilidade de existência de documentos em várias línguas e como tal a necessidade do reconhecimento da língua e a posterior invocação dos respectivos dicionários negativos e radicalizadores. Embora haja trabalho já avançado para o processamento do Português, teriam que ser obtidos de alguma forma as componentes específicas para outras línguas europeias para além do inglês, que é para já a única língua com suporte total.

Também precisa de ser contemplado o reconhecimento de outros tipos de documentos para além do HTML e dos tipos de documentos disponibilizados para a colecção TREC. Como exemplo, apresente-se mensagens de correio electrónico, artigos das Usenet News, documentos word, Adobe PDF, LaTeXentre outros.

Tanto a estratégia de replicação utilizada como a alocação baseada no conteúdo são bastante simples e destinam-se apenas a trabalho laboratorial. Importa trabalhar no sentido de encontrar métodos de replicação mais adequados a ambientes operacionais e usar os algoritmos de aglomeração convencionais para alocação baseada no conteúdo.

No processamento das interrogações, estão para já concretizadas apenas um sub-conjunto das expressões de pontuação, tendo sido ignorados para já os operadores de proximidade e a componente relacionada com documentos estruturados. Os filtros booleanos foram também deixados para uma próxima fase de concretização. O módulo de processamento das interrogações está concebido para a fácil inclusão dos componentes em falta.

O Dicionário de Termos também prevê a inclusão da informação de localização e da estrutura do documento que foram intencionalmente desligados na obtenção dos índices por opção na configuração.

Os mecanismos bem conhecidos de realimentação de relevância e expansão de interrogações embora previstos na arquitectura não foram também ainda concretizados. No que toca a cada BDT a sua concretização é muito simples. Para o EI, a questão já é mais complexa e exige que se tome como ponto de partida trabalho já realizado para esse efeito [XC98].

O trabalho realizado relativamente à meta-informação é insuficiente. Seguiu-se uma abordagem simples que não permite por esse facto explorar uma série de possibilidades. A curto prazo, vai ser encarada e avaliada a substituição do módulo actual de metadata por um módulo que utilize mesmo uma BD relacional, utilizando uma interface JDBC ou uma BD mais simples das que existem em domínio público. Esta substituição vai permitir estabelecer a correspondência entre a interface definida no STARTS, pelos menos em termos dos filtros booleanos sobre campos não indexados pelo motor (por exemplo data, língua) em interrogações que permitam limitar o espaço de pesquisa aos documentos que satisfaçam essa restrição.

Essa possibilidade vai permitir às BDTs exportarem para o EI também sumários dos campos não indexados na mesma linha do que é feito por exemplo no Whois++ [WFS96b]. As BDTs

podem exportar para os EIs sumários das sua meta-informação que podem ser usadas nos EIs para restringirem o espaço de pesquisa em tempo de selecção.
Por outro lado a utilização de BDs convencionais para a concretização do módulo de meta-informação, parece ser uma forma natural de iniciar o estudo do problema mais vasto da integração entre os motores de busca e as BDs convencionais.
Embora tenha sido em alguns módulos vários fios de execução, tratou-se duma utilização ocasional e não se tirou partido de forma eficiente desta possibilidade. Assim, é necessário paralelizar o mais possível os vários módulos e estudar com algum detalhe os aspectos de sincronização associados. À primeira vista há bastante possibilidades de execução de tarefas em paralelo quer na criação dos índices como no processamento das interrogações.
Uma hierarquia de EIs e BDTs precisa duma interface de gestão simples que possa ser realizada através dum único interface conjuntamente com os servidores de documentos respectivos. Esta pretensão põe na ordem do dia a necessidade da gestão deste tipo de aplicações através do SNMP [CFSD89]. Esta pretensão exige, como ponto prévio, a definição duma MIB [KH97] para a BDT e para o EI.

Qualquer programa cresce em complexidade até ultrapassar a capacidade do programador. (Anónimo)

Capítulo 8

Conclusões e Trabalho Futuro

Sumário

Ao longo dos últimos capítulos, foram apresentados os resultados obtidos e feita uma discussão para qualificá-los em face de outros resultados da literatura, sendo referido o trabalho necessário para introduzir melhoramentos. Neste capítulo, faz-se uma apreciação qualitativa mais global e apontam-se algumas direcções para o trabalho futuro e relacionado mais a médio e longo prazo. O trabalho futuro é implicado ou potenciado pelo descrito nesta tese.

8.1 Conclusões

Foi concebida e concretizada uma arquitectura completa para recuperação distribuída de informação textual.

Esta arquitectura foi utilizada para experimentar alguns algoritmos propostos ao longo do trabalho desta tese ou disponíveis na literatura que foram avaliados com base nas metodologias e métricas estabelecidas. Uma das hipóteses avançadas nesta tese, a importância da sobreposição entre as diferentes BDTs nos algoritmos de recuperação distribuída da informação, foi demonstrada com a proposta de novos algoritmos e/ou modificação dos existentes (algoritmo CORI para selecção, lógicas de fusão convencionais) para os quais se obtiveram resultados experimentais.

Nos cenários experimentais utilizados, o recurso a informação de sobreposição permitiu melhorar a eficácia dos algoritmos de fusão e selecção. Enquanto na fusão se utilizou a sobreposição que realmente ocorre em cada interrogação, no caso da selecção a indisponibilidade desse valor fez com que se tivesse que usar apenas o seu indicador, deduzido da sobreposição estática.

Os resultados experimentais obtidos mostraram que é tão importante o valor médio de sobreposição como a sua variância. Os resultados experimentais obtidos permitem concluir que enquanto o algoritmo de fusão é mais sensível aos valores médios de sobreposição, o algoritmo de selecção é dependente da variância.

No caso da selecção, pelo facto de não se ter usado um estimador da sobreposição dinâmica, obteve-se na maior parte das vezes, apenas uma melhoria ligeira da eficácia do algoritmo. É provável que uma melhoria da estimação da sobreposição permitirá melhorarar ainda mais a eficácia do algoritmo.

Pelo que foi mencionado anteriormente, pode-se concluir que a hipótese colocada no início desta tese, a importância da utilização da informação de sobreposição entre as diferentes BDTs nos algoritmos de RI distribuída, foi confirmada.

Os objectivos do trabalho foram alcançados. Os resultados obtidos foram satisfatórios embora necessitem de ser melhor explorados. Uma série de outros parâmetros poderiam ser conjugados, mas a multiplicação do número de experiências a realizar fez com que se tivesse optado por centrar apenas na sobreposição e na distribuição. E, mesmo para esses dois parâmetros, as combinações possíveis foram propositadamente restringidas.

Nas próximas sub-secções apresenta-se mais detalhadamente as conclusões obtidas nos diversos tópicos que foram tratados nesta tese.

8.1.1 A Bancada Webtrieve

Foi concebida e concretizada uma arquitectura de motor de busca distribuído que contempla todos os componentes necessários como Robot, BDT, EI, Alocador de Documentos e Clientes.

Foram além disso concebidos e concretizados um conjunto um conjunto de programas auxiliares que permitem a automatização da criação de fontes suportadas por BDT distribuídas ou centralizadas. Essas fontes podem ser avaliadas em termos de eficácia e eficiência usando metodologias usadas na maior parte dos trabalhos reportados na literatura. Destaque-se em particular a utilização das colecções de teste TREC para avaliação da eficácia do sistema.

Foram utilizados procotocolos normalizados como o HTTP e STARTS para comunicação entre os diversos componentes da arquitectura. A inclusão do STARTS garantiu uma abordagem completa e abrangente à problemática da RI distribuída

O Dicionário de Termos foi concebido usando uma abordagem dual, para suportar as listas de ocorrência longas e curtas, que permite a utilização do Webtrieve em cenários dinâmicos como o WWW.

O Alocador de Documentos dispõe de algoritmos de alocação e replicação que permitem a simulação em Laboratório de cenários de replicação coordenada e não coordenada. Nesta tese utilizaram-se cenários de replicação não coordenada.

O protótipo Webtrieve foi concretizado usando a linguagem Java usando uma abordagem modular e extensível. Já foi testado em laboratório em ambiente multi-plataforma.

Para além da experiência adquirida e dos resultados obtidos neste trabalho, construiu-se uma verdadeira bancada de experiências para recuperação distribuída de informação textual. Esta bancada permite realizar trabalho futuro em que seja possível concentrar, ao contrário do que passou neste trabalho, as energias no problema concreto a resolver sem ter de se criar condições a usar como ponto de partida.

8.1.2 Sobreposição

A abordagem utilizada permitiu distinguir claramente entre a sobreposição estática entre diferentes BDTs e a sobreposição dinâmica que acontece na resposta às mesmas interrogações. Para exemplificar, para variações de sobreposição estática de menos de 1% obteve-se variações de sobreposição dinâmica da ordem dos 20%.

Na RI distribuída a sobreposição dinâmica pode ser usada na fusão de resultados e precisa de ser estimada na selecção de BDTs.

Mostrou-se através de resultados experimentais que embora a sobreposição estática não permita estimar a sobreposição dinâmica, pode ser usada como seu indicador. Mesmo tratando-se dum indicador grosseiro, pode ser útil como parâmetro do algoritmo de selecção e em casos extremos (de sobreposição nula ou completa) permitir mais eficiência no algoritmo de fusão.

Como conclusão apontou-se a possibilidade de utilização de uma BDT com documentos colectados aleatoriamente das várias BDTs indexadas pelo EI, que permita uma estimação da sobreposição por interrogação. Esta BDT deve permitir obviamente documentos replicados, usando uma heurística que identifique a BDT orginal. Os custos desta BDT são partilhados pela sua utilização para expansão de interrogações.

8.1.3 Fusão

Em ambientes em que não estejam disponíveis as estatísticas dos documentos devolvidos para os diferentes termos da interrogação, não é possível realizar em tempo útil a repontuação dos documentos.

Esta repontuação torna-se ainda mais necessária quando se tratam de ambientes heterogéneos em que os motores de busca cujas respostas pretendemos fundir têm subjacentes modelos de RI diferentes e funções de pontuação diferentes, para além das eventuais disparidades nas estatís ticas da colecção.

A abordagem usada nesta tese é a utilização de documentos replicados nos vários conjuntos de resultados a fundir como pontos de referência. Para esses pontos de referênca são usadas lógicas de fusão convencionais. Para os documentos únicos é usada uma aproximação linear simples, usando os pontos de referência. Senão houver sobreposição é usada a intercalação simples. Os resultados experimentais mostraram melhorias da ordem de 30% na precisão das respostas quando comparadas com as devolvidas pelos algoritmos que usam apenas as lógicas de fusão convencionais.

Adicionalmente, os resultados experimentais mostraram, como seria de prever, que a eficácia da fusão tem tendência a melhorar com o aumento da sobreposição e piorar com o número de BDTs.

8.1.4 Selecção

Propôs-se um algoritmo de selecção que pressupõe uma distribuição normal dos termos pelos diferentes documentos na BDT. Usando este pressuposto, conseguiu-se valores de eficácia da selecção da mesma ordem de grandeza de algoritmos bem conhecidos da literatura, pelo menos para os cenários experimentais utilizados. Embora precise de mais informação na síntese do sumário (a informação para cálculo do desvio padrão), prevê-se que consiga determinar com mais precisão o número de documentos a pedir a cada BDT.

Inclui-se o parâmetro de sobreposição no algoritmo de selecção e conseguiu-se melhorias na eficácia mesmo usando valores grosseiros de sobreposição estática. Estas experiências foram realizadas usando o algoritmo CORI, bem conhecido na literatura. A melhoria na eficácia embora não muito significativa é bastante prometedora.

As métricas de avaliação da selecção propostas na literatura foram também reformuladas para permitirem a sua avaliação em ambientes com replicação não coordenada.

Uma outra conclusão importante é que o algoritmo de selecção é mais influenciado pela variância da sobreposição entre as diferentes BDTs do que pela sua média.

8.2 Trabalho Futuro

Este trabalho, embora sendo de uma certa envergadura, eventualmente não totalmente reflectida nesta dissertação, constitui apenas um ponto de partida num longo percurso que se pode fazer quer especificamente na engenharia deste tipo de sistemas, quer em temáticas associadas como a filtragem e sumarização da informação, em que se pode incluir paulatinamente técnicas aprendizagem e mesmo processamento de linguagem natural, principalmente o Português.

Uma outra direcção possível de desenvolvimento pode ser o processamento de outros média nomedamente imagens, som e video. Um dos passos decisivos a dar nesta área é a interpretação semântica de video. Isto passa pela extracção e eventual transformação do seus diversos componentes como texto, som e imagem para posterior processamento.

O trabalho centrou-se fundamentalmente nos aspectos relacionados com a eficácia de sistemas distribuídos de informação. Entretanto e pela sua extensão há uma série de problemas que não foram aprofundados e concretizados nomeadamente a expansão de interrogações e a realimentação de relevância quer em termos da RI distribuída quer mesmo de cada motor de busca. Uma série de mecanismos baseados em técnicas de aprendizagem [Mit97] poderão ser introduzidos nesta aplicação.

8.2.1 Modularidade e extensibilidade

O sistema foi concretizado usando uma abordagem modular e usando uma linguagem orientada aos objectos. Este facto facilita quer a introdução de novos módulos quer a modificação dos já existentes. A utilização do STARTS,WWW e HTTP também garante que o sistema está pronto para acomodar novas facilidades e funcionalidades.

Em termos de concepção e concretização a utilização de ambientes de modelação de software também já foi referida e pode ser introduzida para facilitar o ciclo de desenvolvimento [BJRR98, Sch99] e permitir a reengenharia do próprio sistema.

8.2.2 Processamento das Interrogações

Embora a interface de interrogação do Webtrieve tente ao máximo estar de acordo com o STARTS, apenas as expressões de pontuação foram para já concretizadas. Falta concretizar os filtros booleanos e os operadores de proximidade assim como o suporte à busca de informação estruturada.

A importância da meta-informação e a sua utilização na redução do espaço de pesquisa através da utilização dos filtros booleanos da interrogação STARTS teve que ser substancialmente reduzida. No que toca as expressões boolenas justificava-se adicionalmente uma componente para a sua simplificação, utilizando técnicas conhecidas e utilizadas nos sistemas digitais.

Tanto a meta-informação como a estrutura dos documentos pode ser tratada conjuntamente usando a tecnologia XML[Hol00]. Esta abordagem permite simultaneamente a descrição dos objectos trocados em linha ou via ficheiro substituindo a linguagem de marcas que foi introduzida. A utilização da tecnologia XML em sistemas distribuídos de recuperação de informação é uma temática que precisa só por si de ser explorada [FGK00].

8.2.3 Migração para Web

A utilização do Webtrieve para indexação do Web, é um dos próximos passos a realizar. Para esse efeito preve-se a inclusão dum Robot para colecta da informação no WWW. O robot foi apenas usado no sistema de ficheiros local, uma vez que no trabalho experimental se usou colecções de teste TREC. Por essa razão a sua complexidade foi minimizada. Como já se referiu embora se trate dum componente funcionalmente simples, pode-se tornar relativamente complexo se se pretender indexar um espaço de informação grande [HN99, BP99]. Este componente está a ser estudado no quadro dum novo projecto que tem como objectivo a obtenção de estatísticas do Web em Portugal [CMBS01]. Este trabalho oferece aliás outras aplicações para os robots nomeadamente a sua utilização para percursos aleatórios no WWW [HHMN99, HHMN00].

Um outro componente indispensável é um cliente Web mais amigável. Existe já um cliente Web (no endereço http://polo.uminho.pt/webtrieve) usado apenas para efeitos de desenvolvimento. É necessário, usando como ponto de partida os clientes dos serviços de busca mais populares da Internet [Goo, ALT96], conceber um cliente simples mas flexível, permitindo tirar partido das funcionalidades disponíveis nomeadamente os diversos operadores, realimentação de relevância, busca estruturada de documentos, limitação do espaço da pesquisa, etc...

Embora se tenha concebido e concretizado uma arquitectura para recuperação distribuída de informação textual, deixou-se para já de fora o facto da maior parte da informação actualmente disponível ser semi-estruturada e hipertexto. Não se usa, por ora, as ligações entre os vários documentos para aumentar a eficácia das respostas.

O Webtrieve começa-se a transformar num sistema suficientemente complexo para se justificar uma interface de gestão e configuração via Web, como é comum na maior dos sistemas comercializados. A interface deve ser configurável e para utilização em cenário de investigação ou operacional. Neste cenário de investigação a maior parte do trabalho a realizar é a realização de módulos que permitam a vizualização de gráficos em linha. O processo de obtenção de gráficos de avaliação do sistema precisa de ser completamente automatizado.

Conforme se disse no Capítulo 4, migrando para ambiente operacional, seria necessário usar uma estratégia parecida com a proposta em [BB98] para calcular a sobreposição estática ou usar uma pequena colecção que permita estimar a sobreposição dinâmica.

8.2.4 Eficiência e escala

Os aspectos relacionados com a eficiência do sistema foram colocados num segundo plano. Paradoxalmente, foi utilizada uma única máquina em que as várias BDT e o EI eram processos. Desta forma, os tempos de processamento aumentaram bastante com a distribuição! Para se avaliar a eficiência deste sistema ele deve ser suportado por uma série de computadores interligados em rede local em que se possa colocar cada BDT ou EI num computador dedicado. Embora o software esteja concebido para correr em várias máquinas, com a utilização duma única máquina simplificou-se o protótipo sem perdas significativas em face dos objectivos pretendidos.

O código está concretizado em Java e houve um certo cuidado na construção de componentes mais dependentes da máquina (manipulação do sistema de ficheiros, por exemplo). Embora a parte experimental tenha sido levada a cabo numa Sun com Unix Solaris, o software já correu em sistemas com Windows NT/Windows 2000. Pensa-se assim que não há qualquer dificuldade na execução do software desde que o computador em questão suporte a JVM (máquina virtual Java) com a versão 1.2 ou superior.

Como a componente de resposta a interrogações já está suportada sobre HTTP, há apenas que modificar os programas de construção e gestão de fontes (alocação, indexação, inversão, etc...). Nomeadamente, trata-se de utilizar algumas classes RMI (Remote Method Invocation) do Java em vez da utilização exclusiva de métodos locais. Para esse efeito alguns objectos deverão concretizar a interface *java.rmi.remote*.

Esta migração será realizada mal se consiga disponibilizar um conjunto de PCs para correr uma versão distribuída do Webtrieve.

Ao mesmo tempo, os diversos componentes devem ser estudados no sentido da sua concepção e concretização usando vários fios de execução. Há diversos componentes onde a utilização intensiva de recursos de rede, processamento e memória secundária permite antever ganhos de eficiência como resultado da sua execução simultânea. Uma outra possibilidade para esses ganhos pode ser a utilização duma cache quer no EI como nas diversas BDTs. Embora as caches estejam previstas na arquitectura a sua concretização e o estudo dos algoritmos associados ainda não foi possível nesta fase do projecto.

A eficiência dos sistema de ficheiros precisa entretanto também de ser melhorada com a utilização simultânea de vários discos.

Os problemas de escala, pelo facto de não se ter usado simultaneamente um ambiente de simulação, não foram convenientemente explorados. A escalabilidade da arquitectura e dos algoritmos usados nos diversos componentes será estudada proximamente com auxílio duma ferramenta de simulação.

8.2.5 Alocação e Replicação

Os métodos de alocação e replicação utilizados são muito simples e foram concebidos apenas para criar os cenários necessários para as experiências a realizar no âmbito deste trabalho. Usaram-se estratégias aleatórias e algumas técnicas adicionais para garantir determinados valores médios e variâncias na sobreposição entre as bases de dados do sistema distribuído. As técnicas de aglomeração baseadas no conteúdo não foram convenientemente exploradas. As técnicas de aglomeração de documentos podem ser um componente importante para fazer face ao aumento de escala usando a hipótese de aglomeração. Isto é os documentos podem ser aglomerados por conteúdo, limitando desta forma, o espaço de procura de acordo com o conteúdo da interrogąo.

Os mecanismos de replicação podem ser exploradas no sentido do balanceamento da carga e para introduzir a tolerância a faltas. Podem inclusivé possibilitar a poupança de largura de banda da rede se as réplicas forem localizadas de acordo com os pedidos dos utilizadores.

Toda esta problemática pode ser objecto de trabalho futuro.

8.2.6 Várias línguas

Um outro aspecto que precisa de ser investigado, e ainda se torna mais pertinente num sistema distribuído, é o facto de poderem coexistir nos vários sistemasi, documentos em várias línguas. Actualmente só é suportado o inglês e podem-se introduzir outras línguas com ênfase particular no Português. Foi feito algum trabalho para um radicalizador baseado em regras para Português que não foi incluído por não ter sido terminado, e não caber na temática central desta tese. A opção pelo Português foi posta fora de questão por não existir qualquer colecção de teste (corpus, interrogações e julgamentos de relevância) e a parte experimental ficar bastante limitada.

8.2.7 Vários média

As arquitecturas distribuídas podem adequar-se particularmente para o suporte de documentos multimédia, onde se possa combinar a busca em cada um dos média com a utilização de um sistema intermediário que, para além da selecção de BDTs e fusão de respostas, seja capaz de traduzir as interrogações do utilizador para cada um dos media suportados.

8.2.8 Resumindo e concluindo

Pelos tópicos de desenvolvimento aqui apresentados é fácil de perceber que realmente se trata dum ponto de partida e existem muitas direcções em que esta bancada de testes pode ser explorada. É uma área aplicacional multidisciplinar pelo que seria impossível abarcá-la totalmente neste contexto. A recuperação de informação cosntitui uma área aliciante e promissora de investigação e desenvolvimento.

Conhecimento real é saber a extensão da nossa ignorância.
(Confúcio)

Bibliografia

[AF77] R. Attar e A. S. Fraenkel. Local feedback in full-text retrieval system. *Journal of the ACM*, 24(3):397–417, 1977.

[ALT96] Frequently Asked Questions about AltaVista. 1996. URL:http://altavista.digital.com.

[AS72] Milton Abramowitz e Irene A. Stegun, editores. *Handbook of Mathematical Functions*. Dover Publications, New York, 1972.

[ASF98] Manuel Afonso, Alexandre Santos, e Vasco Freitas. QoS in Web Caching. Em *3rd International WWW Caching Workshop*, Junho 1998. Manchester, England.

[Bau97] Christoph Baumgarten. A Probabilistic Model for Distributed Information Retrieval. Em *Proc. of 20th ACM/SIGIR Conference on Research and Devol opment in information Retrieval, Seattle, WA,EUA*, pg.s 258–266, Philadelphia PA, EUA, 1997.

[Bau99] Christoph Baumgarten. A Probabilistic Solution for Distributed Information Retrieval. Em *Proc. of 22th ACM/SIGIR Conference on Research and Devolopment in information Retrieval, Seattle, WA*, pg.s 246–256, Berkeley,CA, EUA, 1999.

[BB98] Khrisna Bharat e Andrei Broder. A technique for measuring the relative size and overlap of public search engines. *Computer Networks and ISDN Systems*, 30(1-7):379–388, 1998.

[BBM+97] M. Baentsch, L. Baum, Georg Molter, S. Rothkugel, e P. Sturm. World Wide Web Caching: The Application-Level View of the Internet . *IEEE Communications*, Junho 1997.

[BC92] N. Belkin e W. B. Croft. Information filetring and information retrieval: Two sides of the same coin? *Communications of the ACM*, 35(12):29–38, Dezembro 92.

[BDGM95] Sergey Brin, James Davis, e Hector Garcia-Molina. Copy Detection Mechanism for Digital Documents. Em *Proceedings of ACM SIGMOD International Conference on Management of Data*, San Francisco,USA, Maio 1995.

[BG94] Michael Buckland e Fredic Gey. The Relationship between Recall and Precision. *Journal of the American Society for Information Science*, 45(1):12–19, Janeiro 1994.

[BJRR98] Grady Booch, Ivar Jacobson, James Rumbaugh, e Jim Rumbaugh. *The Unified Modeling Language User Guide* . The Addison-Wesley Object Technology Series, Outubro 1998.

[BKFS95] N. J. Belkin, Paul Kantor, E. A. Fox, e J. A. Shaw. Combining the evidence of Multiple Query representations for Information Retrieval. *Information Processing & Management*, 31(3):431–448, 1995.

[BLC95] T. Berners-Lee e D. Connolly. Hypertext markup language - 2.0. Request for Comments 1866, Internet Engineering Task Force, 1995.

[BLFF96] T. Berners-Lee, R. Fielding, e H. Frystyk. Hypertext transfer protocol – HTTP/1.0. Request for Comments 1945, Internet Engineering Task Force, 1996.

[BMS95] M. Baentsch, G. Molter, e P. Sturm. Introducing application-level replication and naming into today Web . Em *Proceedings of 5th WWW Conference, Paris, France*, Maio 1995.

[BP99] Sergey Brin e Lawrence Page. The Anatomy of Large-Scale Hipertextual Search Engine. Relatório Técnico, Computer Science Department, Stanford University, 1999.

[Buc85] Chris Buckley. Implementation of the SMART Information Retrieval System. Relatório Técnico TR85-686, Cornell University, Computer Science Department,EUA, 1985.

[Bur92a] F. Burkowski. An algebra for hierachically text-dominated databases. *Information Processing & Management*, 28(3):333–348, 1992.

[Bur92b] F. Burkowski. Retrieval activities in a database consisting of heterogenous collections of structured text. Em *Proc. of 15th ACM/SIGIR Conference on Research and Devol opment in information Retrieval*, pg.s 112–125, Copenhaga, Dinamarca, 1992.

[BYN96] R. Baeza-Yates e G. Navarro. Integrating contens and structure for text retrieval. *ACM SIGMOD Record*, 25(1):67–79, Março 1996.

[BYN99] Ricardo Baeza-Yates e Berthier Ribeiro Neto. *Modern Information Retrieval.* ACM Press,Addison Wesley, 1999.

[CCB94] Charles Clarke, G. Cormack, e F. Burkowski. An Algebra for Structured Text Search and A Framework for its Implementation. Relatório técnico, Computer Science Department,University of Warteloo, Canada, 1994. Acessível por WWW.

[CCGMP96] Kevin Chen-Chuan, Hector Garcia-Molina, e Andreas Paepcke. Boolean Query Mapping Across Heterogenous Information Sources . *IEEE Transations on Knowledge and Data Engineering*, 8(4), Agosto 1996.

[CCGMP97] Kevin Chen-Chuan, Hector Garcia-Molina, e Andreas Paepcke. Evaluating the Cost of Boolean Query Mapping. Relatório Técnico, Electrical Engineering Department, Stanford University, Agosto 1997.

[CCH92] James P. Callan, W. Bruce Croft, e Stephen M. Harding. The INQUERY Retrieval System. Em *Proceedings of the 3rd International Conference on Database and Expert Systems Applications*, pg.s 78–83, 1992.

[CFSD89] J. D. Case, M. Fedor, M. L. Schoffstall, e C. Davin. Simple network management protocol (SNMP). Request for Comments 1098, Internet Engineering Task Force, Abril 1989.

[CH79] William B. Croft e D.J. Harper. Using probablistic models of retrieval without relevance information. *Journal of Documentation*, 35(4):285–295, 1979.

[CHT99] Nick Craswell, David Hawking, e Paul B. Thistlewaite. Merging results from isolated search engines. Em *Australasian Database Conference*, pg.s 189–200, 1999.

[CLC95] James P. Callan, Zhihong Lu, e W. Bruce Croft. Searching Distributed Collections with Inference Networks. Em *Proceedings of ACM SIGIR Conference on Research in Information Retrieval, Seattle, WA*, Julho 1995.

[CM00] Brendon Cahoon e Kathryn S. McKinley. Evaluating the Performance of Distributed Architectures for Information Retrieval using a variety of workloads. *ACM Transations on Information Systems*, 8(10):1–43, Janeiro 2000.

[CMBS01] António Costa, Joaquim Macedo, Orlando Belo, e Alexandre Santos. NetCensus, Obtenção de Estatísticas para o Web em Portugal. Relatório Técnico, Departamento de Informática, Universidade do Minho, 2001. Em preparação.

[CMK66] C.W. Cleverdon, J. Mills, e M. Keen. Determining the Performance of Indexing Systems, Vol I, Design, Vol II, Test Results. Relatório Técnico, ASLIB Cranfield Project, London, 1966.

[Com95] Oracle Company. Strategies and Techniques for Use Oracle7 Replication . Relatório Técnico, Oracle, 1995.
http://www.oracle.com/produts/servers/replication/html/collateral.html.

[Coo94] William S. Cooper. The formalism of probablity theory in IR: A foundation or an emcumbrance? . Em *Proc. of ACM/SIGIR Conference on Research and Devolopment in Information Retrieval*, pg.s 242–247, Dublin, Irlanda, 1994. Triennal ACM-SIGIR Award Paper.

[CPC+99] James C.French, Allison L. Powell, Jamie Callan, Charles L. Viles, Travis Emmit, Kevin J. Prey, e You Mou. Comparing the Performance of Database Selection Algorithms . Em *Proc. of 22th ACM/SIGIR Conference on Research and Development in information Retrieval, Berkeley, CA, EUA*, pg.s 238–245, Berkeley,CA, EUA, 1999.

[CPV98] James C.French, Allison L. Powell, e Charles L. Viles. Evaluating Database Selection Tecnhniques: A Testbed and Experiment . Em *Proc. of 21th ACM/SIGIR Conference on Research and Development in information Retrieval, Melbourne, Australia*, pg.s 121–129, Agosto 1998.

[CSS98] William Cohen, Robert E. Schapire, e Yoram Singer. Learning to order things. Relatório técnico, ATT Shanon Laboratory, EUA, 1998.

[DGH+97] M. Dreger, K. Grossjohann, C. Haber, S. Lohrum, e R. Weber. Medoc - Searching Heterogeneous Bibliographic and Text Databases. Relatório Técnico, University of Dortmund, Alemanha, 1997.

[Exp97] José Exposto. Aglomeração não Hierárquica em Sistemas Distribuídos de Recuperação de Informação. Tese de Mestrado, Departamento de Informática, Universidade do Minho, Setembro 1997.

[FGK00] Norbert Fuhr, Kai Grobjohann, e Stefan Kokkelink. CAP7: Searching and Browsing Distributed Document Collections. Em *Research and Advanced Technology for Digital Libraries, 4th European Conference, Lisboa, Lectures Notes in Computer Science*, Setembro 2000.

[Fox92] Christopher Fox. *Information Retrieval - Data Structures and Algorithms - Lexical Analysis and Stoplists*, captulo 7, pg.s 102–130. Prentice-Hall, 1992.

[Fra92] W. B. Frakes. *Information Retrieval - Data Structures and Algorithms - Stemming Algorithms*, captulo 8, pg.s 131–159. Prentice-Hall, 1992.

[FSW96] P. Faltstrom, R. Schoultz, e C. Weider. How to interact with a whois++ mesh. Request for Comments 1914, Internet Engineering Task Force, 1996.

[Fuh96] Norbert Fuhr. Optimum Database Selection in Networked IR. Em *Proceedings of the SIGIR-96, Workshop on Networked Information Retrieval*, Agosto 1996.

[GBY90] G.H. Gonnet e R. Baeza-Yates. *Handbook of Algoritmhs and Data Structures in Pascal and C.* Addison-Wesley, 1990. 2ªEdição.

[GCGM+97] Luis Gravano, Kevin Chang, Hector Garcia-Molina, Carl Lagoze, e Andreas Paepcke. STARTS: Stanford Protocol Proposal for Internet Retrieval and

Search. Relatório técnico, Digital Library Project, Stanford University, EUA, Janeiro 1997.

[GGM95] Luis Gravano e Héctor García-Molina. Generalizing GlOSS to Vector-Space Databases and Broker Hierarchies. Em *Proc. of the 21st VLDB Conference, Zurich, Switzerland*, 1995.

[GGMT94] Luis Gravano, Hector Garcia-Molina, e Anthony Tomasic. The Efectiveness of Gloss for the Text Database Problem. Em *Proceedings of ACM SIGMOD International Conference on Management of Data*, pg.s 126–137, 1994.

[Goo] Google! Search Engine.
URL:http://google.stanford.edu.

[Gre95] Gregory Grefenstette. Comparing two language identification schemes. Em *Proceedings of 3rd International Conference on Statistical Analysis of Textual Data*, Roma, Itália, Dezembro 1995.

[GSD+88] G.W.Furnas, S.Deerwster, S.T. Dumais, T.k. Landauer, R.A. Harsham, L.A Streeter, e K.E. Lochbaum. Information Retrieval using a singular value decomposition model of latent semantic structure. Em *Proc. of 11th ACM/SIGIR Conference on Research and Development in Information Retrieval*, pg.s 465–480, 1988.

[Har92a] Donna Harman. *Information Retrieval - Data Structures and Algorithms*, captulo Ranking Algorithms, pg.s 363–391. Prentice-Hall, 1992.

[Har92b] Donna Harman. *Information Retrieval - Data Structures and Algorithms*, captulo Relevance Feedback and Other Query Modification Techniques, pg.s 128–153. Prentice-Hall, 1992.

[Har93] Donna Harman. Overview of the First Text REtrieval Conference. Em *Proc. of 16th ACM/SIGIR Conference on Research and Development in information Retrieval, Pittsburgh, PA USA*, Julho 1993.

[HFBYL92] Donna Harman, Edward Fox, R. Baeza-Yates, e W. Lee. *Information Retrieval - Data Structures and Algorithms* , captulo Inverted Files, pg.s 28–43. Prentice-Hall, 1992.

[HHMN99] Monika R. Henzinger, Allan Heydon, Michael Mitzenmacher, e Marc Najork. Measuring Index Quality using Random Walks on the Web . Em *Eight Internacional World Wide Web Conference*, pg.s 213–225, Toronto, Canada, Maio 1999.

[HHMN00] Monika R. Henzinger, Allan Heydon, Michael Mitzenmacher, e Marc Najork. On Near-Uniform URL Sampling. Em *Nine Internacional World Wide Web Conference*, pg.s 295–308, Amsterdão, Holanda, Maio 2000.

[HMW+95] D. Hardy, U. Manber, D. Wessel, M. Bowman, P. Danz ig, e M. Schwartz. Harvest: A Scalable, Customizable Discovery and Access System, Março 1995. URL:http://www.ncsa.uiuc.edu/SDG/IT94/Proceedings/Searching/ schwartz.harvest/schwartz.harvest.html.

[HN99] Allan Heydon e Marc Najotk. Mercator: A Scalable, Extensible Web Crawler. Relatório Técnico 85-686, Compaq Systems Research Center, 1999.

[Hol00] Steven Holzner. *Inside XML*. New Riders Publishing, Novembro 2000.

[Hor83] M. R. Horton. Standard for interchange of USENET messages. Request for Comments 850, Internet Engineering Task Force, Junho 1983.

[HS93] Darren R. Hardy e Michael F. Schwartz. Essence: A Resource Discovery System Based on Semantic File Indexing. Em *Proceedidgs of the USENIX Winter Conference*, pg.s 361–374, San Diego, California, Janeiro 1993.

[Int] The Internet Archive.
URL:http://www.archive.org.

[ITU93] ITU. *The Directory Recommendations X.500-X.521,ISO/IEC Information Technology - Open Systems Interconnection*. ITU - International Telecommunication Union, Dezembro 1993.

[Jon72] Karen Sparck Jones. A statistical interpretation of term specificity and its application in retrieval . *Journal of Documentation*, 28:11–21, 1972.

[Jon95] Karen Sparck Jones. Reflections on TREC. *Information Processing and Management, Vol 31, no 3*, pg.s 291–314, 1995.

[JY92] Carolyn J.Crouch e Bokyung Yang. Experiments in automatic statistical thesaurus construction. Em *Proceedings of ACM-SIGIR Conference on Research and Development in Information Retrieval*, pg.s 77–88, Copenhaga,Dinamarca, 1992.

[KH97] C. Kalbfleisch e Harrie Hazewinkel. Definitions of Managed Objects for WWW servers, Internet-draf t (work in progress). Março 1997. URL=ftp://funet.fi/internet-drafts/draft-ietf-applmib-wwwmib -02.txt.

[KL86] B. Kantor e P. Lapsley. Network news transfer protocol. Request for Comments 977, Internet Engineering Task Force, Fevereiro 1986.

[KS96] Fergus Kelledy e Alan F. Smeaton. Thresholding the Posting Lists in Information Retrieval. Relatório Técnico, School of Computer Applications, Dublin City University, 1996.

[Kun95] Joanh A. Kunze. Basic Z39.50 Server Concepts and Creation, Agosto 1995.

[LCC96] Zhihong Lu, James P. Callan, e W. Bruce Croft. Applying Inference Networks to Multiple Collections Searching. Relatório técnico, Computer Science Department,University of Massachusetts, 1996. Accessible by WWW.

[Lea97] Doug Lea. *Concurrent Programming in Java, Design Principles and Patterns.* Addison-Wesley, 1997.

[Lee97] Joon Ho Lee. Analyses of Multiple Evidence Combination. Em *Proceedings of 20th ACM SIGIR Conference on Research in Information Retrieval, Philadelphia, EUA*, pg.s 267–276, Julho 1997.

[LG98] Ulana Ledgeza e John Guttag. Using Network Level Support to Improve Cache Routing . Em *3rd International WWW Caching Workshop*, Junho 1998. Manchester, Inglaterra.

[Lim96] Cybotics Technologies Limited. White Paper: Cybotics Search Engine. Relatório Técnico, 1996.

[LM99] Zhihong Lu e Katrhryn S. McKinley. Partial Replica Selection Based on Relevance for Information Retrieval . Em *Proc. of 22th ACM/SIGIR Conference* *on Research and Devol opment in Information Retrieval, Berkeley, CA, USA*, Agosto 1999.

[LS99] Ora Lassila e Ralp. R. Swick. Resource Description Framework (RDF) Model and Syntax Specification. Relatório Técnico, W3 Consortium, Fevereiro 1999. Recomendação do W3C.

[Lu99] Zhihong Lu. *Scalable Distributed Architectures for Information Retrieval.* Tese de doutoramento, University of Massachusetts at Amherst, USA, Maio 1999.

[LY99] Tim Lindholm e Frank Yellin. *Java Virtual Machine Specification.* The Java Series. Addison-Wesley, 1999. 2ªEdição.

[MA98] Daniel Menascé e Virgilio Almeida. *Capacity Planning for Web Performance.* Prentice Hall, 1998.

[Mar92] Gary Marchionini. Interfaces for End-USER Information Seeking. *Journal of the American Society for Information Science*, 43(2):156–163, Março 1992.

[MB98] Robert Miller e Krishna Bharat. SPHINX: A Framework for Creating Personal, Site-Specific Web Crawlers. Relatório Técnico, School for Computer Science, Carnegie Mellon University Pittsburg, April 1998.

[MCF98] Joaquim Macedo, Antonio Costa, e Vasco Freitas. Webtrieve: Uma arquitectura distribuída para Bases de Dados de Texto. Relatório técnico, Universidade do Minho, 1998. acessível em ftp://marco.uminho.pt/Docs/local/RT/webtrieve. ps.

[MCF00a] Joaquim Macedo, António Costa, e Vasco Freitas. Fusion of Overlapped Results Sets. Em *Research and Advanced Technology for Digital Libraries, 4th European Conference, Lisboa, Lectures Notes in Computer Science*, Setembro 2000.

[MCF00b] Joaquim Macedo, António Costa, e Vasco Freitas. Webtrieve: A Testbed Systems for Distributed Information Retrieval . Em *The Proceedings of 3rd International Conference of Asian Digital Libraries, Seoul, Korea*, Dezembro 2000.

[MCF01] Joaquim Macedo, António Costa, e Vasco Freitas. Selection of overlapped dabatases. Relatório Técnico, Departamento de Informática, Universidade do Minho, Maio 2001. para submissão.

[Met00] Metamata. JavaCC-Java Compiler Compiler , 2000. URL=http://www.metamata.com/javacc/index.html.

[Mit97] Tom Mitchell. *Machine Learning.* McGraw-Hill, 1997.

[MNRZ98] Scott Michel, Khoi Nguyen, Adam Rosenstein, e Lixia Zhang. Adaptive Web Caching: Towards a New Caching Architecture. Relatório técnico, Computer Science Department, UCLA,USA, 1998. http://wwwcache.ja.net/events/workshop/25/3w3.html.

[Moc87a] P. V. Mockapetris. Domain names - concepts and facilities. Request for Comments 1034, Internet Engineering Task Force, Novembro 1987.

[Moc87b] P. V. Mockapetris. Domain names - implementation and specification. Request for Comments 1035, Internet Engineering Task Force, Novembro 1987.

[MZ95] Alistair Moffat e Justion Zobel. Information retrieval systems for large document collections. Em *3rd Text Retrieval Evaluation Conference (TREC 3)*, 1995.

[NBY95] G. Navarro e R. Baeza-Yates. A language for queries on sctructure and contents of textual databases. Em *Proc. of 18th ACM/SIGIR Conference on Research and Devol opment in information Retrieval*, pg.s 93–101, Seattle, EUA, 1995.

[NBY97] G. Navarro e R. Baeza-Yates. Proximal Nodes: A model to query document databases by content and structure. *ACM Transactions on Office and Information Systems*, 15(4):401–435, 1997.

[OMK91] Y. Ogawa, T. Morita, e K. Kobayashi. A fuzzy document retrieval system using the keyword connection matrix and a learning method . *Fuzzy Sets and Systems*, 39:163–179, 1991.

[PFG+94] M. St. Pierre, J. Fullton, K. Gamiel, J. Goldman, B. Kahle, J. Kunze, H. Morris, e F. Schiettecatte. WAIS over Z39.50-1988. Request for Comments 1625, Internet Engineering Task Force, 1994.

[Por80] M. F. Porter. An Algorithm for Suffix Stripping. *Program*, 14(3):130–137, 1980.

[Pow01] Allison Powell. *Database Selection in Distributed Information Retrieval: A Study of Multi-Collection Information Retrieval.* Tese de doutoramento, Faculty of School of Engineering and Appleid Sciences, University of Virginia, EUA, Janeiro 2001.

[QF93] Yonggang Qiu e H. P. Frei. Concept based query expansion. Em *Proceedings of 16th ACM-SIGIR Conference on Research and Development in Information Retrieval*, pg.s 160–169, Pittsburgh, PA, EUA, 1993.

[Riv92] R. Rivest. The MD5 Message-Digest Algorithm. Request for Comments 1321, Internet Engineering Task Force, 1992.

[RJ76] George Robertson e Karen Sparck Jones. Relevance weigthing of search terms. *Journal of the American Society for Information Science*, 27(3):129–146, 1976.

[RM96] Paul Resnick e Jim Miller. PICS: Internet Access Controls Without Censorship. *Communications of ACM*, 39(10):87–93, 1996.

[RM01] Sriram Raghavan e Hector Garcia Molina. Crawlling the Hidden Web. Relatório Técnico, Computer Science Departament, Stanford University, 2001. to appear on VLDB 2001 Conference.

[RMCF95] Miguel Rio, Joaquim Macedo, António Costa, e Vasco Freitas. Supporting a URI Infrastructure by Message Broadcasting. Em *HyperMedia Proc of INET'95 - Internet Society's 1995 International Networking Conference, http://info.isoc.org/HMP/PAPER/116/ps/paper.ps, Honolulu, Hawaii, EUA*, Junho 1995.

[RMF97] Miguel Rio, Joaquim Macedo, e Vasco Freitas. A Distributed Weighted Centroid-based Indexing System. Em Terena, editor, *Proc. of JENC8 the 8th*

Joint European Networking Conference, Maio 12-15, Edimburgh, Scotland, Maio 1997.

[Sal91] Gerard Salton. The State of Retrieval System Evaluation. Relatório Técnico TR 91-1206, Department of Computer Science, Cornell University, Maio 1991.

[SB88] Gerald Salton e Chris Buckley. Term weighting approaches in Automatic Text Retrieval. *Information Processing and Management*, 24(5):513–523, 1988.

[SC97] Alan F. Smeaton e Francis Crimmins. Using a Data Fusion Agent for Searching the WWW. Relatório técnico, School of Computer Applications, Dublin City University, Agosto 1997.

[Sch93] Peter Schaube. SPIDER: A Multiuser Information Retrieval System for Semisctrutured and Dynamic Data. Em *Proc. of 16th ACM/SIGIR Conference on Research and Devol opment in information Retrieval*, 1993.

[Sch99] Joseph Schmuller. *Sams Teach Yourself UML in 24 Hours*. Sams, Julho 1999.

[SFW83] Gerald Salton, Edward Fox, e Harry Wu. Extended boolean information retrieval systems. *Communications of the ACM*, 26(11):1022–1036, Dezembro 1983.

[SGM94] Naraynan Shivakumar e Hector Garcia-Molina. SCAM: A Copy Detection Mechanism for Digital Documents. Relatório técnico, Electrical Engineering Department, Stanford University, EUA, 1994. Acessível por WWW.

[SGM95] Narayanan Shivakumar e Hector Garcia-Molina. The SCAM Approach to Copy Detection in Digital Libraries. *d-lib magazine*, Novembro 1995. http://www.dlib.org/dlib/november95/scam/11shivakumar.html.

[SL68] Gerald Salton e M. Lesk. On the specification of terms values in automatic indexing . *Journal of the ACM*, 15(1):8–36, Janeiro 1968.

[SM71] Gerald Salton e M. McGill. *The SMART Retrieval System- Experiments in Automatic Processing.* Prentice Hall,Inc, 1971.

[SM83] Gerald Salton e M. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, New York, EUA, 1983.

[SM94] K. Sollins e L. Masinter. Functional requirements for uniform resource names. Request for Comments 1737, Internet Engineering Task Force, 1994.

[Sol98] K. Sollins. Architectural principles of uniform resource name resolution. Request for Comments 2276, Internet Engineering Task Force, 1998.

[SSMB94] Amit Singhal, Gerard Salton, Mandar Mitra, e Chris Buckley. Document Length Normalization. Relatório técnico, Department of Computer Science, Cornell University, Novembro 1994.

[SY73] Gerald Salton e C.S. Yang. On the specification of terms values in automatic indexing . *Journal of Documentation*, 29:351–372, Janeiro 1973.

[Tay62] Robert S. Taylor. The Process of Asking Questions. *American Documentation*, 13(4):391–396, 1962.

[TC90] Howard Turtle e W.Bruce Croft. Inference networks for document retrieval . Em *Proc. of 13th ACM/SIGIR Conference on Research and Devolopment in Information Retrieval*, pg.s 1–24, Bruxelas, Bélgica, 1990.

[TC91] Howard Turtle e William Bruce Croft. Evaluation of an inference network-based retrieval model. *ACM Transations on Information Systems*, 9(3):187–222, Jullho 1991.

[TGMS94] Anthony Tomasic, Hector Garcia-Molina, e Kurt Shoens. Incremental Updates of Inverted List for Text Information Retrieval. Em Richard T. Snodgrass e Marianne Winslett, editores, *Proceedings of ACM SIGMOD International Conference on Management of Data*, pg.s 289–300, Minneapolis,Minnesota, EUA, Maio 1994.

[VF95] Charles L. Viles e James C. French. Dissemination of Collection Wide Information in a Distributed Information Retrieval System. Em Edward A. Fox, Peter Ingwersen, e Raya Fidel, editores, *Proceedings of the 18th Annual International Conference on Research and Development in Information Retrieval (SIGIR'95)*, pg.s 12–20, New York, NY, EUA, Julho 1995. ACM Press.

[VGB61] J. Verhoeff, W. Goffmann, e Jack Belzer. Inefficiency of the use of Boolean functions for information retrieval systems. *Communications of the ACM*, 4(12):557–558, Dezembro 1961.

[VGJL95] Ellen M. Voorhees, Narendra K. Gupta, e Ben Johnson-Laird. Learning collection fusion strategies. Em *Proceedings of the Eighteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pg.s 172–179, 1995.

[VL00] D. Viswanathan e S. Liang. Java Virtual Machine Profiler Interface. *IBM Systems Journal*, 39(1):82–95, 2000.

[Voo86] Ellen M. Voorhees. *The effectiveness and Efficiency of Agglomerative Hierachical Clustering in Document Retrieval.* Tese de doutoramento, Cornell University, 1986.

[Voo95] Ellen M. Voorhees. Siemens TREC-4 Report: Further Experiments with Database Merging. Em *Proceedings of Text Retrieval Evaluation Conference (TREC-4), Maryland, USA*, Novembro 1995.

[vR95] C.J. van Rijsbergen. Information Retrieval, 1995.
URL=http://www.dcs.gla.ac.uk/Keith/Preface.html.

[W3C96] W3C. Overview of CGI, 1996.
URL=http://www.w3.org/hypertext/WWW/CGI/Overview.html.

[War92] S. Wartick. *Information Retrieval - Data Structures and Algorithms*, captulo Boolean Operations, pg.s 264–292. Prentice-Hall, 1992.

[WFS96a] C. Weider, J. Fullton, e S. Spero. Architecture of the whois++ index service. Request for Comments 1913, Internet Engineering Task Force, 1996.

[WFS96b] C. Weider, J. Fullton, e S. Spero. Architecture of the whois++ index service. Request for Comments 1913, Internet Engineering Task Force, Fevereiro 1996.

[WM97] Stuart Weibel e Eric Miller. Dublin Core Metadata Element Set: Reference Description, Janeiro 1997.
URL=http://purl.org/metadata/dublin_core_elements.

[WMB94] Ian Witten, Alistair Moffat, e Timothy Bell. *Managing Gigabytes.* International Thomson Publishing, New York, EUA, 1994.

[WZRW85] S.K.M. Wong, W. Ziarko, V.V. Raghavan, e P.C.N. Wong. Generalized vector space model in information retrieval. Em *Proc. of 8th ACM/SIGIR Conference on Research and Devolopment in Information Retrieval*, pg.s 18–25, New York, EUA, 1985.

[WZRW87] S.K.M. Wong, W. Ziarko, V.V. Raghavan, e P.C.N. Wong. On modeling of information retrieval concpets in vector spaces. *ACM Transations on Database Systems*, 12(2):299–321, 1987.

[XC96] J. Xu e W. B. Croft. Query expansion using local and global document analysis. Em *Proceedings of ACM-SIGIR Conference on Research and Development in Information Retrieval*, pg.s 4–11, Zurich, Suiça, 1996.

[XC98] Jinxi Xu e Jamie Callan. Effective Retrieval with Distributed Collections. Em *Proc. of 21th ACM/SIGIR Conference on Research and Devol opment in Information Retrieval, Melbourne, Australia*, Agosto 1998.

[YS76] C.T. Yu e Gerald Salton. Precision weigthing — an effective automatic index method . *Journal of the ACM*, 29:351–372, Janeiro 1976.

Quando se rouba de um autor chama-se plágio; quando se rouba de muitos chama-se investigação. (Wilson Mizner)

Glossário

Aglomeração (*Clustering*)
Agrupamento de documentos ou termos que partilham um conjunto de propriedades comuns. A ideia é agrupar documentos similares ou termos que ocorram frequentemente nos mesmos documentos

Aglomeração Global
Agrupamento de todos documentos da colecção.

Aglomeração Local
Agrupamento do sub-conjunto de documentos relevantes para a interrogação, ignorando os restantes documentos.

Algoritmo de corte
Thresholding na terminologia inglesa. Permitem processar só parte das listas de ocorrência aumentando a eficiência no processamento da interrogação.

Anagrama (*n-gram*)
qualquer subpalavra de comprimento n.

Aprendizagem Computacional (*Machine Learning*)
Algoritmos ou programas que melhoram o seu comportamento com o tempo, aprendendo com a sua eficácia e eficiência no passado.

Base de Dados de Texto
Sistema que permite a Recuperação de Informação para uma determinada colecção de documentos. Motor de busca ou *search engine*, em inglês.

BDT
Acrónimo de Base de Dados de Texto

Basic-1
Esquema de registo para meta-informação

Cobertura (*Recall*)

Medida de eficácia usada em RI que corresponde à fracção de documentos relevantes existentes na colecção que foram devolvidos

Cobertura Alternativa

Medida de eficácia usada na selecção de BDTs em alternativa à cobertura. Em vez de se usar no denominador da fórmula o número de documentos relevantes existentes para as n BDTs seleccionadas como na cobertura, usa-se sempre o total de documentos relevantes existentes.

Colecção

Um grupo de itens, normalmente documentos. Nas bibliotecas digitais isso corresponde a um conjunto de documentos agrupados por qualquer razão de gestão.

Colecção distribuída

Colecção composta por colecções mais pequenas mas vistas como uma só pelo utilizador pelo facto de usar um sistema distribuído de RI.

Colecção de Teste

Uma colecção usada para testar modelos e algoritmos de recuperação de informação. Normalmente inclui um conjunto de documentos, um conjunto de interrogações de teste e os documentos consideardos como relevantes para cada interrogação.

Colecção TREC

Colecção de teste de referência.

Curva de Actualizações

Ritmo de actualizações, que pode ser experimental (obtido com base em registos históricos) ou distribuições estatísticas. Serve para animar as actualizações em Laboratório e avaliar a eficiência do sistema de RI face a actualizações.

Curva de Precisão e Cobertura

Gráfico de precisão e cobertura.

Curva de tráfego

Ritmo de interrogações, que pode ser experimental (obtido com base em registos históricos) ou distribuições estatísticas. Serve para animar as interrogações em Laboratório e avaliar a eficiência do sistema de RI face a interrogações.

Dados Multimédia

Dados que combinam vários media tal como texto, imagens, som e video.

Dados semi-estruturados

Dados cuja estrutura pode não unificar ou unifica parcialmente a estrutura descrita por um esquema de dados.

Dados semi-estáticos

Um colecção de textos que não muda muito frequentemente.

DC

Acrónimo de Dublin Core

Desacerto (*fall-out*)

Medida de eficácia. Proporção de documentos não relevantes devolvidos.

Detector de Língua

Programa que permite determinar a língua em que determinado documento está escrito. Existem várias técnicas baseadas em anagramas ou em palavras frequentes.

DIMF

Acrónimo de *Distributed Index Markup Language*.

Dicionário Negativo

Lista de palavras que ocorrem frequentemente no texto de qualquer documento. Exemplos são artigos, preposições e conjunções.

Distributed Index Markup Language

Linguagem de etiquetas usada no Webtrieve para especificar os vários tipos de objectos persistentes trocados em ficheiro pelos vários programas.

Documento

A unidade usada na recuperação de informação. Pode ser um artigo, uma página web, um livro, um parágrafo, uma se cção, uma imagem, etc...

Documento indexado

Representante do documento depois de processado pelo Indexador de Documentos Tipicamente corresponde a uma tabela de ocorrências e um registo de meta-informação.

Directório

Uma categorização hierárquica de conceitos numa área de conhecimento. Muitos portais mantêm os documentos organizados em directórios.

DTD

Definição de Tipo de Documento: Definição SGML para uma linguagem de etiquetas.

Dublin Core

Esquema para registo de meta-informação para documentos

Eficácia

Uma medida da correcção com que o sistema de RI responde às interrogações dos utilizadores. Tipicamente é aferida com base em curvas de precisão e cobertura.

Eficiência

Uma medida da relação entre a velocidade de resposta e os recursos (processamento, armazenamento e rede) gastos pelo sistema de RI.

EI

Acrónimo de Encaminhador de Interrogações

Encaminhador de Interrogações

Componente do sistema distribuído de RI responsável pelo encaminhamento das interrogações

Encaminhamento de interrogações

Ver Selecção de fontes

Escala

Tamanho do problema a resolver. No caso da RI os parâmetros desta escala são dados pelo número de documentos, número de BDTs, ritmo de interrogações e actualizações.

Escalável

Um sistema, algoritmo ou abordagem é escalável se se comporta normalmente em termos de eficiência e eficácia quando se aumenta a escala do problema a resolver.

Estrutura do documento

Informação incluída num documento para além do seu conteúdo que permite relacionar as suas partes de forma semântica

Estrutura do texto

Ver estrutura do documento

Etiqueta

Uma palavra que é usada para marcar o princípio ou fim dum elemento estrutural do texto

Expansão da interrogação

Um processo de adicionar novos termos à interrogação numa tentativa de uma melhor contextualização para devolução dos documentos relevantes para o utilizar.

Expressão regular

Padão genérico que permite a expressões com *strings* alternativas, repetições e concatenação de *substrings*.

Ficheiro invertido

Resultado da inversão das tabelas de ocorrência de todos os documentos que constituem a colecção.

Filtragem de informação

Sistema de RI que indexa perfis de informação que correspondem a necessidades de informação e compara com os documentos dum fluxo fazendo chegar aos utilizadores os documentos considerados relevantes pelo respectivo perfil.

Fonte

Entenda-se uma BDT centralizada ou distribuída que permita a pesquisa e o acesso a uma colecção de documentos. O conjunto de documentos é agrupado em colecção por qualquer razão administrativa, geográfica, política ou outra qualque

Fusão de resultados

Um processo de junção de respostas vindas das várias fontes interrogadas para juntar numa única a devolver ao utilizador. Trata-se dum problema importantíssimo da recuperação distruíde informação.

HPROF

Traçador de perfis que usa o JVMPI.

HTML

Acrónimo de*HyperText Markup Language* (HTML), a linguagem de etiquetas usada no Web.

HTTP

Acrónimo de*HyperText Transfer Protocol*, protcolo usado no Web para transferência de páginas de hipertexto.

ID

Acrónimo de Indexador de Documentos.

Indexador de Documentos

Componente do sistema de RI que processa os documentos extraindo a informação considerada útil para construir o ficheiro invertido e os registos de meta-informação.

Interrogação

Uma expressão da necessidade de informação do utilizador na linguagem disponibilizada pelo sistema. Normalmente inclui um conjunto de termos e operadores booleanos ou de proximidade.

JVMPI

Acrónimo de *Java Virtual Machine Profiler Interface.* Trata-se duma Interface entre a máquina virtual Java e o traçador de perfis.

Linguagem de etiquetas

Terminologia em Português para designar o conceito *markup language*.

Linguagem de interrogação

Linguagem usada para descrever as necessidades de informação em formato processável pelo sistema de RI.

Lista de ocorrência

Conjunto de registos de ocorrência relativo a um termo, isto é, informação a respeito da ocorrência do termo em cada documento. A terminologia usada em inglês é *Posting List*.

Listas não sobrepostas

Ver Modelo de Listas não sobrepostas.

LOC

Acrónimo de Lista de Ocorrência Curta

LOL

Acrónimo de Listas de Ocorrência Longa

MARC

Esquema normalizado para meta-informação usado nas Biliotecas para documentos

MBasic-1

Esquema para registo de meta-informação para BDTs ou EIs

Metadados

Dados a respeito de dados. Neologismo usado como sinónimo de meta-informação.

Meta-informação

Informação a respeito da informação. Atributos do documento tal como autor, título tamanho, hiperligações, etc... Normalmente é mantido num esquema como Dublin Core, MARC ou Basic-1

Modelo Booleano

Modelo clássico de recuperação de informação baseado na teoria dos conjuntos

Modelo Booleano Extendido

Um modelo de recuperação de informação basado numa extensão do modelo booleano clássico. A ideia é a interpretação das unificações parciais como distâncias euclideanas representadas num espaço vectorial de termos de índice.

Modelo do Espaço Vectorial

Modelo clássico de recuperação de informação baseado na representação de documentos e interrogações como vectores de termos. O modelo pressupõe a indepndência entre os termos.

Modelo Probabilístico

Modelo clássico de recuperação de informação baseado na interpretação probabilística da relevância dum documento para uma dada interrogação.

Modelo de Listas não sobrepostas

Um modelo de recuperação de documentos estruturados através de estruturas de índices concretizados como listas não sobrepostas.

Modelo de Nós Próximos

Um modelo para recuperação de documentos estruturados através duma estrutura hierárquica de índices.

Modelo de Recuperação de Informação

Um conjunto de premissas e uma algoritmo para pontuar documentos para uma determminada interrogação do utilizador.

Navegação (*Browsing*)

Processo interactivo em que o utilizador está mais interassado em explorar e conhecer os documentos do que satisfazer uma necessidade específica de informação.

Necessidade de informação do utilizador

Uma frase em linguagem natural que especifique a necessidade de informação do utilizador. Por exemplo, *Procure documentos que discutam as implicações políticas resultantes dum Deputado do CDS/PP e Presidente da Câmara de Ponte de Lima, Daniel Campelo, ter viabilizado com a sua abstenção, a aprovação do Orcamento de Estado de 2000 apresentado pelo Governo do PS.*

Nós próximos

ver Modelo dos Nós Próximos.

Palavra chave

Ver Termo de índice

Perfil de informação

Especificação duma necessidade permanente de informação do utilizador, com uma estrutura complexa, usada em sistemas de filtragem de informação.

Precisão

Medida de eficácia usada em RI que corresponde à fracção de documentos relevantes no total de devolvidos.

Preservação Digital

Serviço que assegura que um objecto digital continue acessível e útil durante um período de tempo (centenas de anos) acessível e passível de utilização. Isso requer normalmente transferência de media (de *tape* para CD,por exemplo) e conversão de formatos

Radicalização (*Stemming*)

Uma técnica para reduzir as palavras ao seu radical gramatical.

Realimentação de relevância

Mecanismo iterativo em que o utilizador vai usando as respostas do sistema para melhorar a sua interrogação ate obter uma resposta que o satisfaça. Trata-se da utilização, no contexto da RI, do mecanismo clássico de realimentação de sistemas de controle.

Recuperação de Dados

A recuperação de itens(tuplos, documentos, páginas Web, etc...) cujo conteúdo satisfaz exactamente as condições especificadas na interrogação (tipo expressão regular)

Recuperação distribuída de informação

A utilização de técnicas de computação distribuída para resolver o problema da recuperação da informação

Recuperação de Informação Multimedia

Sistema de recuperação de informação que manipula documentos multimedia.

Redes de Crença (*Belief Networks*)

Um modelo probabilístico de RI baseado na interpretação de documentos, interrogações e termos de índice como nós duma rede de Bayes. Este modelo é diferente das redes de inferência.

Redes de Inferência

Um modelo probabilístico de RI baseado na interpretação de documentos, interrogações e termos de índice como nós duma rede de Bayes. Este modelo é diferente das redes de crença.

Recuperação de Informação (*Information Retrieval*)

Área das ciências da computação que estuda a recuperação de informação (não de dados) duma colecção de documentos. Os documentos devolvidos têm como objectivo satisfazer uma necessidade de informação do utilizador expressa normalmente em linguagem natural

Registo de Ocorrência

Informação de ocorrência dum termo num documento. Pode incluir informação de localização e estatísticas de ocorrência. É designado por *posting*, na terminologia inglesa.

Refinamento da interrogação

Processo iterativo que permite ao utilizador ir melhorando a interrogação representando mais fielmente a sua necessidade de informação.

Relevância

Qualidade dos documentos que satisfazem a necessidade de informação do utilizador, explicitada na interrogação.

Representante

Forma mais compacta de representar os objectos relevantes (documentos e interrogações) num sistema de RI.

Selecção de fontes

Selecção no conjunto de colecções de documentos disponíveis daquelas que é suposto conterem os documentos relevantes para determinada interrogação. É um dos problemas mais importantes da recuperação distribuida de de informação

Servidor de Documentos

Um servidor que devolve documentos ao utilizador. Um servidor WWW, por exemplo.

SGML

Acrónimo para *Standard Generalized Markup Language*. Meta linguagem de etiquetas. O HTML é baseado no SGML.

Sobrecarga de informação

Grande número de documentos e fontes de informação disponíveis, e o utilizador tem dificuldade em escolher as mais relevantes.

Sobreposição

Documentos replicados em duas ou mais BDTs distintas ou em respostas a uma mesma interrogação.

STARTS

Acrónimo de *Stanford Proposal for Internet Meta-Searching*. O STARTS é um protocolo para RI distribuída em ambientes heterogéneos, desenvolvido na Universidade de Stanford, EUA, em cooperação com a indústria de produtos de RI.

Sumário da BDT

Resumo de informação da BDT exportada para o EI. Neste resumo existe informação do número de ocorrências e número de documentos para cada termo do vocabulário de termos.

Tempo de processamento

Tempo de CPU necessário para para correr determinado programa ou método.

Tempo de corrida

Tempo medido entre o início e o fim do programa.

Termo de índice

Um palavra usada para identificar o conteúdo dum documento. Normalmente é um nome ou uma frase (2 ou mais nomes).

Thesaurus

Uma estrutura de dados composta de uma lista de palavras importantes dum dado domínio de conhecimento e para palavra da lista, um lista de palavras relacionadas (sinónimos, etc...).

Totalidade

Usada como sinónimo de cobertura.

Traçador de Perfil

Pacote de software que permite estabelecer o perfil dum outro programa em termos de recursos necessários de processador, memória, E/S com base na monitorização de eventos.

Unificação exacta (*Exact Match*)

Mecanismo através do qual apenas os objectos que satisfazem exactamente algum critério bem especificado, é devolvido o utilizador como resposta à interrogação

URL

Acrónimo de *Uniform Resource Locator*, termo em inglês usado para designar o endereço de um recurso Internet, incluindo as páginas Web.

Vocabulário

O conjunto de todas as palavras diferentes num texto e ou numa colecção.

WAIS

Acrónimo de *Wide Area Information Service*: conjunto de protocolos concebidos para publicar information e interrogar bases de dados na Internet.

Webtrieve

Nome do protótipo de sistema de RI distribuído, concebido e concretizado no âmbito da parte experimental desta tese. O nome é o acrónimo usado para WEB reTRIEVE.

XML

Acrónimo de *eXtended Markup Language*. O XML é um subconjunto do SGML especialmente definido para o Web. No XML é mais simples definir novas linguagens de etiquetas.

Z39.50

Uma norma dos EUA que se tornou numa norma internacional. Trata-se dum protocolo para comunicação cliente/servidor em sistemas de RI. Permite a interacção de um cliente com mais que um sistema de RI, ou um sistema de RI (ou gateways) comunicarem com outros sistemas de RI. Suporta sessões orientadas à conexão havendo a possibilidade de um sistema descrever-se em resposta ao comando *EXPLAIN*, submissão de interrogações, obtenção de informação, lista de resultados e os próprios documentos encontrados.