UMA REVISÃO DE LITERATURA SOBRE SISTEMAS DE RECUPERAÇÃO DE INFORMAÇÃO NA WEB

Talles Brito Viana

Instituto Federal de Educação, Ciência e Tecnologia do Ceará – *campus* Crato

tallesbrito@ifce.edu.br

RESUMO: *Sistemas de recuperação de informações têm a função de conduzir usuários para itens com a maior probabilidade de satisfazer as necessidades de informação dos mesmos. Estes sistemas estão presentes em diversos cenários. Um dos cenários é a World Wide Web, no qual deve ser fornecida para os usuários uma interface de consulta para um conjunto amplo e de larga-escala de documentos os quais são previamente indexados. Neste contexto, este trabalho apresenta uma revisão de literatura sobre sistemas de recuperação de informação na Web com o objetivo de apresentar uma análise das tecnologias e métodos adotados no projeto de tais sistemas de busca e recuperação.*

Palavras-chave: *Revisão de Literatura. Recuperação de Informação. Busca na Web.*

ABSTRACT: *Information retrieval systems are meant to lead users to objects with the highest probability of meeting their user information needs. These systems are present in several scenarios. One scenario is the World Wide Web, in which a query interface should be provided for users in order to look for documents within a wide range and large-scale previously indexed collection. In such a context, this paper presents a survey of information retrieval systems on the Web. Besides, in this paper is presented an analysis of employed technologies and methods in the design of such retrieval systems.*

Keywords: *Survey. Information Retrieval. Web Search.*

1. INTRODUÇÃO

Em geral, sistemas de recuperação e descoberta de informações têm a função de conduzir os usuários para os documentos ou objetos que irão satisfazer com maior probabilidade necessidades de informação. De forma mais genérica, o objetivo de um sistema de recuperação de informações é obter informações a partir de uma fonte de conhecimento que auxiliará os usuários do sistema a resolver algum tipo de problema (BELKIN; CROFT, 1992).

Uma das características da situação em que o usuário tem um problema é a necessidade de busca por alguma informação, situação no qual o usuário é estimulado a iniciar um procedimento de pesquisa, como por exemplo, ao submeter uma consulta para um sistema de recuperação de informações. A consulta deve ser expressa em uma linguagem que possa ser entendida pelo sistema e pode ser vista como a representação da necessidade de informação. Vale ressaltar que, na maioria dos casos, as consultas podem ser imperfeitas e expressar as necessidades de informação de maneira aproximada (BELKIN; CROFT, 1992).

Além disto, um sistema de recuperação de informação tem a função de representar um conjunto de documentos ou objetos para posterior recuperação. O processo de representar o significado de documentos ou objetos de forma a serem receptivos ao processamento por sistemas de computação é geralmente denominado de indexação, e é de central importância nos sistemas de recuperação de informação (BELKIN; CROFT, 1992). Estes conceitos teóricos da teoria de recuperação de informação são aplicados em diversos cenários. Um dos cenários é a Web, no qual um conjunto de larga-escala de documentos deve ser representado e deve ser fornecida uma interface de consulta para esta coleção (BOWMAN et al.,1994).

Neste contexto, este artigo apresenta uma revisão de literatura sobre sistemas de recuperação na Web. O objetivo deste trabalho é analisar e identificar técnicas que podem ser adaptadas, melhoradas e adotadas no projeto e implementação de novos sistemas de recuperação de informação para Web. Em específico, isto inclui a investigação, identificação e análise dos mecanismos de indexação e as linguagens de consulta suportadas por sistemas de recuperação de informação na Web existentes. A principal contribuição deste trabalho é enumerar e detalhar como sistemas de recuperação na Web são projetados a fim de gerar conhecimento básico e fundamental deste tópico que ainda é bastante intrigante.

2. FUNDAMENTAÇÃO TEÓRICA

A Web é um espaço para publicação de conteúdos distribuídos que engloba diversos recursos, tais como, páginas pessoais, bibliotecas digitais, museus virtuais,

catálogos de produtos e serviços, repositórios de arquivos FTP, repositórios de notícias e e-mails (GUDIVADA et al.,1994). O aumento de popularidade da Internet, bem como a crescente ordem de conteúdos publicados e disponíveis na Web acarreta em uma maior dificuldade para se encontrar informações na mesma (BOWMAN et al.,1994).

Neste contexto, é necessária a existência de métodos para buscar e recuperar informações na Web de forma eficiente e que forneçam resultados de qualidade. Apesar disto, existe um conjunto de fatores que dificulta a construção de métodos eficientes. Em (BAEZA-YATES; RIBEIRO-NETO, 1999), são citados: o fato dos dados serem distribuídos; o alto volume de dados voláteis disponíveis na Web; o crescimento exponencial dos dados; o fato dos dados disponíveis serem não estruturados e bastante redundantes; e a heterogeneidade e mudança dinâmica dos dados disponíveis.

Neste cenário, os sistemas de busca deveriam prover resultados de ótima qualidade, e para isto, devem apresentar melhores formas de apresentação dos documentos e ranqueamento dos mesmos, além disto, responder rapidamente pelas consultas e eliminar links quebrados ou redundantes, e também, evitar a apresentação de documentos que não seriam interessantes aos usuários (HAWKING, 2006a). Para prover um serviço eficiente e de custo-efetivo, os sistemas de busca devem indexar somente o conteúdo da Web que realmente é merecedor de ser indexado, e de forma automatizada. Estes são alguns dos desafios que estão em volta da tarefa de construir sistemas de busca na Web.

Nesta seção, conceitos fundamentais relacionados a sistemas de busca na Web são apresentados a fim de iniciar uma discussão sobre o detalhamento de como estes tipos de sistemas são geralmente projetados. Assim, em seguida, são discutidos os modelos gerais utilizados para coletar informações disponíveis na Web, bem como, modelos gerais utilizados na indexação e consultas de documentos na Web. Estes conceitos e modelos serão referenciados na posterior revisão de literatura que será apresentada na Seção 3.

2.1. Coleta de Dados na Web

Em geral, na arquitetura de um sistema de busca na Web existe um módulo de coleta (*Crawler*). Um coletor atravessa toda a Web a fim de coletar documentos através de um caminho obtido através dos links nos documentos. Este caminho pode ser visto como uma árvore que pode ser percorrida por profundidade ou largura. O mais simples algoritmo de coleta de documentos na Web utiliza-se de um uma fila de URLs a serem visitadas e um mecanismo para determinar se uma URL já foi coletada (HAWKING, 2006a). O coletor inicia fazendo requisições HTTP para um documento e, ao obtê-lo inspeciona-o com o objetivo de obter novas URLs a serem coletadas caso já não tenham sido anteriormente. Os documentos recuperados são armazenados em bases de dados para posterior indexação. Vale ressaltar que, segundo Hawking (2006a), projetar um coletor escalável para Web não é uma tarefa fácil. Coletores devem tratar aspectos de paralelismo a fim de se conseguir uma taxa aceitável do processo de coleta. Além disto, as questões relacionadas à comunicação com milhares de servidores Web devem ser levadas em conta, no qual é importante considerar pontos de tolerância a falhas no projeto dos coletores, pois comumente os servidores Web podem falhar.

2.2. Indexação de Documentos na Web

Em geral, sistemas de busca utilizam-se de índices invertidos para identificar quais documentos possuem termos informados através de consultas (HAWKING, 2006b). Para cada um dos termos de um restrito vocabulário, um *índice invertido* armazena uma listagem de documentos tal que o termo ocorre. Isto é, para cada termo existe uma lista com os identificadores dos documentos que o contém. O processo de indexação geralmente ocorre em duas fases. Na primeira fase, documentos são inspecionados a fim de se extrair os termos que eles possuem. Na segunda fase, os dados gerados na primeira fase são processados com o objetivo de construção dos índices invertidos, no qual os documentos e termos são identificados univocamente na mesma. Numa consulta, os termos devem ser localizados rapidamente no índice invertido, e para isto, os sistemas de busca podem utilizar árvores ou tabelas *hash*. Além disto, da mesma forma que no processo de coleta, o processo de indexação pode ser dividido entre clusters e técnicas de compressão de dados podem ser utilizadas para reduzir a quantidade de espaço ocupada pelos índices e dados temporários gerados.

2.3. Consulta e Ranqueamento de Documentos na Web

De uma forma geral, os sistemas de busca atuais retornam os documentos que contém todos os termos informados em uma consulta (HAWKING, 2006b). Para isto, cada termo informado na consulta é procurado no índice invertido, e com isto, são encontrados os documentos que contém algum dos termos informados. No caso de vários termos, o processador de consulta verifica no conjunto de documentos retornados, quais deles possuem ao mesmo tempo, todos os termos informados na consulta. Por fim, os documentos resultantes após a fase de consulta são geralmente ordenados (ranqueados) através de uma função de escore, para isso deve-se utilizar um método de ranqueamento. O *ranqueamento* consiste em empregar algoritmos e modelos matemáticos para quantificar o grau de relevância de cada um dos documentos retornados em

relação à necessidade de informação do usuário do sistema de busca.

2.4. Metabusca na Web

Um *metabuscador* (*metasearcher*) é uma ferramenta de busca que automaticamente e simultaneamente direciona uma consulta do usuário para diversos motores de busca convencionais ou bancos de dados e, exibe os resultados combinados destes motores de busca fonte para o usuário. Os resultados obtidos dos diversos sistemas de busca fonte são coletados, analisados, possivelmente reconfigurados e por fim, apresentados para o usuário em formato uniforme. Por exemplo, um metabuscador pode ordenar documentos a partir de diferentes atributos, tais como, data do documento ou popularidade (LAM, 2001).

3. UMA REVISÃO DE LITERATURA SOBRE SISTEMAS DE RECUPERAÇÃO DE INFORMAÇÃO NA WEB

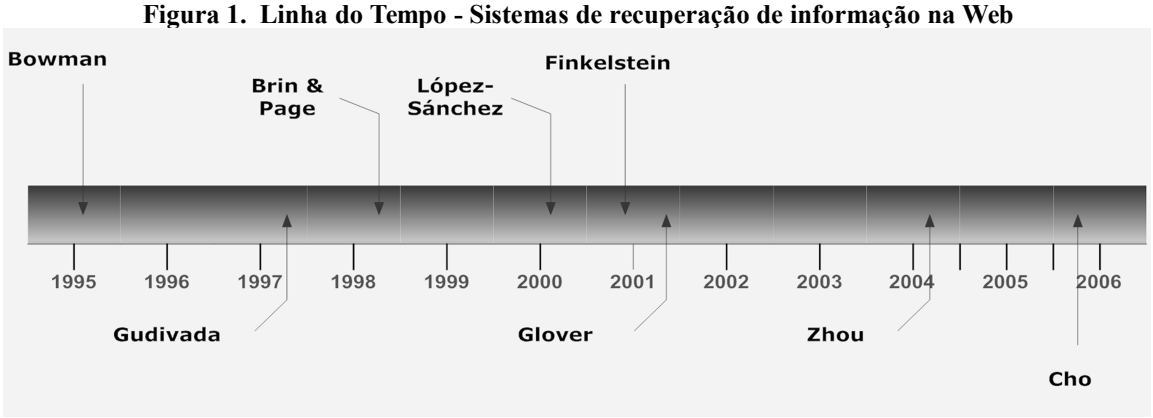
A seguir, de acordo com a linha de tempo apresentada na Figura 1, será apresentado e discutido um conjunto de sistemas de busca e recuperação de informação na Web. Os trabalhos escolhidos trouxeram algum tipo de contribuição relevante à área e permitem mostrar como que os sistemas de busca podem ser projetados para indexação e recuperação de documentos em larga escala. Além disso, os trabalhos foram escolhidos de acordo com o impacto e mudanças geradas nos paradigmas de projeto que os sistemas de busca na Web sofreram. O objetivo disto é mostrar e discutir as mudanças de paradigmas que ocorreram no projeto de sistemas Web com o passar do tempo, bem como o surgimento de novos desafios e aprimoramentos. Sendo assim, na sequência, os sistemas de busca denominados de *Harvest* (BOWMAN et al., 1994), *Google* (BRIN & PAGE, 1998), *iBot* (LÓPEZ-SÁNCHEZ et al., 2000), *Inquirus* (GLOVER et al., 2001), *InteliZap* (FINKELSTEIN et al., 2001), *Coopeer* (ZHOU et al., 2004) e *WebBase* (CHO et al., 2006) são apresentados, detalhados e discutidos:

Harvest: A Scalable, Customizable Discovery and Access System (BOWMAN et al., 1994): O Harvest (BOWMAN et al.,1994) pode ser considerado como um dos primeiros mecanismos para possibilitar a descoberta de informações na Web. Seu surgimento ocorreu em um cenário de crescimento das informações disponíveis na internet, no qual, encontrar informações se tornava cada vez mais difícil. Harvest tenta resolver o problema de descoberta de informações na web, ao permitir a construção de ferramentas para busca de informações em diversos repositórios, tais como FTP, HTTP, Gopher. Além disso, permite a existência de mecanismos de indexação específicos, que podem ser utilizados em diferentes cenários.

Um dos objetivos de projeto do Harvest é prover um sistema flexível e eficiente sobre o qual sistemas de busca possam ser construídos. Este sistema é fortemente caracterizado pelo fato de adotar uma abordagem escalável e descentralizada, no qual diversas instâncias do mesmo podem trocar informações e indexar conteúdos. Para isto, ele provê uma arquitetura configurável, no qual é possível indexar conteúdos distribuídos através de um mecanismo de indexação que pode ser configurado dependentemente de domínio ou aplicação.

Na arquitetura do Harvest, são definidos subsistemas com papéis e objetivos específicos. Dentre estes, os *Providers* são responsáveis por prover informações através de serviços padrões da internet (Gopher, FTP, HTTP). Já os *Gatheres* têm a função de coletar informações de um ou mais providers. Por fim, os *Brokers* têm a função de indexar as informações obtidas pelos gatherers e fornecer uma interface de consulta sob tais informações indexadas.

Figura 1. Linha do Tempo - Sistemas de recuperação de informação na Web



Tal abordagem é defendida pelo fato de diminuir o tráfego de informações ao possibilitar uma maior coordenação e possível cooperação entre estes subsistemas. Por exemplo, ao invés de vários diferentes e independentes sistemas de indexação acessarem a variados repositórios de informações (FTP, HTTP), um gatherer poderia coletar tais informações dos repositórios e fornecê-las para diversos sistemas de indexação (brokers) de uma forma coordenada, em que se tenha um uso mais eficiente da rede e redução de tráfego.

Um gatherer coleta informações periodicamente de providers, estes que são programas residentes nos servidores de informações dedicados a servir informações para os gatherers. Por exemplo, a abordagem usual de se obter informações através da coleta por profundidade através de links poderia ser evitada em troca uma abordagem customizada (por quem serve as informações), e preferencialmente, otimizada. Além disso, o gatherer pode gerar resumos (uma forma simplificada de descrição de um objeto) para posterior indexação. Vale ressaltar que, a maneira de extração destes resumos, bem como critérios de seleção de quais objetos devem ser selecionados para indexação são flexivelmente configuráveis através do Harvest.

O broker é composto de quatro módulos: *Collector, Registry/Storage, Index/Search Engine e Query Manager*. O módulo Collector periodicamente requer atualizações para um Gatherer ou outro Broker a fim de atualizar ou obter informações. O módulo Registry/Storage mantém uma lista de objetos indexados no broker. Já o módulo Query Manager recebe consultas e as converte para um formato de representação interno que é enviado para o módulo Index/Search Engine, este por sua vez, retorna uma lista com a descrição de objetos que satisfazem a consulta. Além disso, é importante ressaltar dois pontos. O primeiro diz respeito ao fato da possibilidade de cooperação entre descentralizadas instâncias de brokers, que são possibilitadas por um protocolo de transferência de dados implementado pelo *Query Manager*. O segundo referencia ao quesito que está em volta do sistema Harvest: flexibilidade de configuração. Vários parâmetros de um broker podem ser configurados, tais como, a lista de gatherers e brokers que devem ser contatados e a frequência de contato.

Por fim, iremos considerar os mecanismos de consulta e indexação do Harvest. O módulo Index/Search Engine de Harvest define uma interface que pode acomodar diversos motores de busca com seus próprios mecanismos de indexação e ranqueamento. Desta forma, permite-se a construção de brokers com mecanismos de indexação diferentes e específicos. Por exemplo, é possível a construção de um broker indexador de artigos científicos enquanto outro broker poderia ser específico na indexação de artefatos de software. O único requisito sobre tais brokers é que eles suportem consultas com operadores booleanos e consultas baseadas em palavras-chave ou pares atributo-valor. Isto acontece devido ao fato do módulo Query Manager, em qualquer broker, satisfazer uma interface de consulta uniforme. Como exemplo da versatilidade do Harvest, em (BOWMAN et al.,1994) são demonstrados dois sistemas implementados: *Glimpse* e *Nebula*. Enquanto o Glimpse suporta eficientes mecanismos de indexação e consultas flexíveis interativas, o Nebula suporta buscas de resposta rápida e complexos modos de consulta.

Information Retrieval on the World Wide Web (GUDIVADA et al., 1997): Gudivada et al. (1997) expõe uma avaliação experimental dos sistemas de busca disponíveis na época de redação do artigo. A partir disto, Gudivada et al. (1997) apresentou questões e desafios abertos relacionados com a busca de informações na Web nesta época. Inicialmente, foi definida uma taxonomia sobre ferramentas de busca na Web. Sobre tal taxonomia, são definidas suas categorias: *Tipo 1* e *Tipo 2*. As ferramentas de busca de Tipo 1 são consideradas aquelas que escondem a complexidade de organização e indexação do conteúdo da Web. Já as ferramentas do Tipo 2 organizam as informações da Web categoricamente, no qual o usuário navega entre uma estrutura de diretórios para localizar informações.

O que caracteriza a existência de ferramentas de Tipo 1 é a utilização de coletores que atravessam a Web a fim de obter conteúdo para indexação. Ao contrário, as ferramentas do Tipo 2 não fazem uso deste tipo de recurso, mas apresentam um catálogo de páginas disponíveis na Web, no qual o julgamento sob a categorização é feita manualmente através da submissão de itens pelos usuários da ferramenta.

Dentre as ferramentas de busca na Web analisadas, podemos destacar os sistemas de busca Altavista e Yahoo. Altavista é uma ferramenta de Tipo 1 que indexa textualmente todos os termos de páginas HTML capturadas pelo seu coletor. As consultas são feitas através de palavras-chave e aceitam operadores booleanos. Além disso, no Altavista os mecanismos de ranqueamento de documentos são baseados em heurísticas que consideram a frequência, posição no documento e proximidade dos termos.

Já o Yahoo (na época do trabalho) era uma ferramenta do Tipo 2 que provia operadores booleanos para consulta e apresentava documentos relacionados por categorias. Estes sistemas podem ser considerados não acadêmicos, de forma que não existiam documentos públicos que especificavam o projeto e implementação dos mesmos.

A principal contribuição deste trabalho foram os resultados obtidos através de uma avaliação das principais ferramentas de busca (de Tipo 1 e Tipo 2) disponíveis na época. Foi comparada a eficiência dos sistemas em termos

qualitativos. Os sistemas foram submetidos a um conjunto de consultas e os documentos retornados foram avaliados em sua relevância.

Como principal resultado, em geral, os documentos relevantes retornado eram intercalados com documentos irrelevantes. Isto significou que os usuários de tais sistemas não poderiam examinar somente os primeiros documentos no topo da lista de documentos retornados e descartar os documentos em posições mais baixa. Como o número de documentos retornado é geralmente de grande ordem, manualmente inspecionar os documentos retornados era uma tarefa tediosa para os usuários. Ou seja, existia a necessidade de sistemas de busca que prezassem essencialmente pela qualidade de seus resultados.

The Anatomy of a Large-Scale Hypertextual Web Search Engine (BRIN & PAGE, 1998): Este trabalho apresenta um novo sistema de busca denominado Google (BRIN; PAGE, 1998). O Google surge em um cenário onde as tecnologias de busca na Web necessitavam ser escaláveis devido ao crescente volume de informações disponíveis na Web, e ao mesmo tempo, necessitavam ser rápidas e proverem resultados atualizados e com qualidade. Para isto, seus autores citam os desafios que estavam em volta do objeto de desenvolver um sistema de busca escalável, do qual podemos citar: a tecnologia de coleta deveria ser rápida o suficiente para manter os índices sempre atualizados; o espaço de armazenamento deveria ser utilizado eficientemente para armazenar documentos e índices; os mecanismos de indexação deveriam processar uma quantidade de informações de grande ordem; as consultas deveriam ser processadas de forma rápida; e por fim, deveria ser aprimorada a qualidade dos resultados providos nas consultas. Neste sentido, iremos citar resumidamente como tais requisitos foram tratados pela abordagem proposta neste trabalho.

O Google é fortemente caracterizado por implementar dois importantes recursos que implicam efetivamente em melhores resultados, tais mecanismos são denominados *PageRank* e *AnchorText*. Google faz uso dos links entre páginas para obter o julgamento sobre a relevância das mesmas. Em outras palavras, faz uso da estrutura de links da Web para obter um ranqueamento das páginas. Este mecanismo é denominado PageRank, um algoritmo que calcula pesos para as páginas. Ele pode ser pensado como uma medida de importância que se assemelha ao comportamento de uso dos usuários quando estão buscando por páginas na Web. Intuitivamente, páginas que são bem referenciadas em vários lugares da Web tem um alto valor de PageRank, e consequentemente deveriam ser inspecionadas pelos usuários. Além disso, páginas que apesar de pouco referenciadas, mas por páginas com alta relevância, também terão alta relevância. Isso ocorre devido à propagação de pesos considerada no algoritmo. O PageRank é calculado usando um algoritmo iterativo simples e pode ser conferido com mais detalhes em (ARASU et al.,2001). Já o mecanismo AnchorText, pode ser visto como uma abordagem que defende a ideia de que na indexação, os termos dos links são os melhores descritores da página que são referenciadas, isto até mesmo do que a própria página. Este mecanismo provê qualitativamente melhores resultados.

Resumidamente, a arquitetura do Google é composta por módulos responsáveis por resolver endereços e coletar páginas da Web; repositórios distribuídos de documentos e índices; módulos que provêm indexação e ranqueamento de documentos, bem como, a execução de consultas.

O módulo *URLServer* envia endereços de páginas para coletores distribuídos a fim de obter documentos para posterior indexação. Cada coletor tem aproximadamente 300 conexões abertas ao mesmo tempo, necessárias para indexar a Web de forma eficiente. Os documentos coletados são comprimidos e guardados em repositórios e são identificados univocamente através de um identificador. O módulo *Indexer* lê o repositório, extrai os termos contidos nos documentos e constrói índices (não invertidos) temporários que são armazenados em um repositório distribuído, em que cada unidade que o compõe é denominada *Barrel.* Além de extrair os termos, o Indexer extrai os links dos documentos e os armazena nos módulos *Anchors* e *Links,* estes links são utilizados como entrada para os mecanismo AnchorText e PageRank, bem como são utilizados pelo URLServer para entregar novos itens a serem coletados pelos coletores. Já o módulo *Sorter* tendo como entrada o índice não invertido armazenado nos Barrels, constrói os índices invertidos, no qual os termos serão identificados por um identificador. Por fim, o módulo *Searcher* é executado em um servidor Web e responde às consultas executada ao sistema.

As consultas são expressas através de palavras-chave. A consulta é analisada e seus termos são extraídos. No caso de um único termo, os Barrels são percorridos a fim de encontrar os documentos que têm a ocorrência do termo. No caso de uma consulta de vários termos, os Barrels são percorridos a fim de encontrar documentos que apresentem a ocorrência de todos os termos simultaneamente. Após esta fase, é calculado o ranqueamento destes documentos considerando o PageRank juntamente com uma medida que avalia o tipo, posição, cor e fonte dos termos nos documentos.

Além disto, questões sobre a escalabilidade e desempenho da arquitetura física do sistema são detalhados em (BARROSO et al.,2003). Para tratar as consultas de forma eficiente e sem gargalos, o serviço é distribuído em múltiplos clusters espalhados geograficamente. As consultas são executadas sob as informações dispersas entre os clusters. A busca por um

documento em um índice de larga-escala é transformada em uma busca por documentos em um conjunto menor de índices, seguido por uma fase final pouco custosa de junção destes resultados. Para isto, o arquivo de índices é quebrado em pedaços, os quais são espalhados entre as diferentes máquinas do cluster.

Quanto à arquitetura física, em vez da adoção de poucos servidores de grande porte, foi optado por utilizar um cluster composto por *PCs* comuns. É mostrado em (BARROSO et al.,2003), que esta solução tem um custo mais efetivo. O que favorece a utilização desta abordagem é o paralelismo inerente requerido pelo sistema. Neste sentido, como um alto nível de paralelismo é explorado, justifica-se o uso de várias máquinas, em vez da utilização poucas máquinas oferecendo um desempenho *single-thread*. Por isto, em vez de utilizar servidores caros e confiáveis, foi preferível a utilização de máquinas menos confiáveis e baratas, em compensação, a confiabilidade das mesmas deve ser garantida em nível de software.

Além disto, as estruturas de dados implementadas pelo Google são otimizadas a fim de que uma grande quantidade de documento possa ser coletada e indexada com baixo custo. Elas foram projetadas a fim evitar buscas de disco sempre que possível. Maiores detalhes sobre tais estruturas de dados podem ser encontradas em (BRIN; PAGE, 1998).

Agent Communication Inside an Internet Search Engine (LÓPEZ-SÁNCHEZ et al., 2000): Lopéz-Sanchés et al. (2000) apresenta a arquitetura de um sistema de busca baseado em agentes, denominado iBot. O iBot é provido por um conjunto de agentes que tem funcionalidades bem definidas. A arquitetura do sistema é composta pelos seguintes componentes: Agentes para coletar as informações da Web (*Crawling Agent Community*); um agente responsável por escolher e indicar as URLS para os agentes de coleta (*URL Broker Agent*); agentes que analisam e filtram as informações coletadas e as convertem para um formato definido (*Filtering Agent Community*); um agente indexador que é responsável por criar os índices (*Indexer Agent*); agentes de consulta que consultam os índices e respondem as consultas (*Query Agent Community*); e por fim, agentes de interface (*Interface Agent Community*) que apresentam os resultados das consultas para os usuários, bem como, permitem que o usuário inicie consultas.

Estes agentes trabalham coletivamente com o objetivo de coletar, indexar e fornecer interfaces de consulta aos usuários. Os agentes de coleta (Crawling Agent Community) formam um grupo dedicado para fazer requisições HTTP e obter páginas da Web, cada agente é autônomo e decide quando chamar o agente URL Broker para obter novas URLs para serem vasculhadas. Após processar uma página, o agente de coleta (Crawling Agent) provê ao agente URL Broker as novas URLs extraídas a partir dos documentos coletados. Toda vez que um agente de coleta obtém (Crawling Agent) uma página, um dos agentes de filtragem (Filtering Agent Community) analisa o documento e deriva um arquivo XML (de formato pré-definido) para posterior indexação. Uma vez que um suficiente número de páginas for filtrado, o agente de indexação (Indexer Agent) gera uma estrutura que contém índices invertidos e pesos que podem ser usados para fins de ranqueamento. Por fim, agentes de interface (User Interface Agent) apresentam interfaces de consulta para os usuários, bem como, os resultados das consultas. Estas consultas são enviadas para agentes de consulta (Query Agent Community) que se utilizam dos índices invertidos para responder as consultas e retornar as páginas que melhor se adéquam às mesmas. A arquitetura proposta é bastante semelhante com outros trabalhos que são constituídos por módulos de indexação, coleta e indexação. A única diferença é ser baseada em agentes.

Web Search - Your Way (GLOVER et al., 2001): Glover et al. (2001) apresenta a arquitetura de um metabuscador que tem como principal objetivo oferecer para o usuário um mecanismo mais apurado para definir suas preferências. Esta arquitetura considera informações de preferência dos usuários para determinar o comportamento das consultas e como ranquear os resultados. Tal arquitetura é realizada em um sistema denominado *Inquirus*.

A arquitetura proposta para o Inquirus define um novo mecanismo para selecionar sistemas de busca fonte e aprimorar as consultas ao analisar resultados através de um processo de reordenação dos documentos obtidos. Para alcançar estes objetivos, as consultas no Inquirus não são limitadas a somente palavras-chave, além disso, o usuário pode explicitar informações que controlam a estratégia de busca a ser usada pelo sistema.

As preferências são expressas através de categorias. Cada categoria provê uma descrição de alto nível sobre a necessidade do usuário, e é associada a um conjunto de atributos que influem no comportamento da estratégia de busca. Por exemplo, as categorias “Research Pages About”, “Current Events”, representam necessidades do usuário em encontrar artigos científicos e eventos recentes, respectivamente. Neste contexto, para ajustar a estratégia de busca, o Inquirus se utiliza de três técnicas: modificação de consultas, seleção de sistema de busca fonte e políticas de ordenação baseada em uma função atributo-valor. O mecanismo de modificação de consultas consiste em adicionar ou remover termos das consultas para os sistemas de busca fonte, em que os métodos da modificação da consulta são dependentes da categoria escolhida. A seleção de sistemas de busca fonte permite que os sistemas fontes aos quais são enviadas as consultas sejam selecionados dependentemente da

necessidade do usuário. Por exemplo, para a categoria "Research Pages About", sistemas de busca de bases de dados de artigos científicos deveriam ser consultados.

Por fim, a reordenação dos resultados pelos vários sistemas de busca consultados pelo Inquirus adota uma política que considera pesos associados a atributos, estes que são atribuídos a cada uma das categorias. Em outras palavras, o ranqueamento em cada categoria pode ser representado como a união dos valores de um conjunto de atributos. Por exemplo, a categoria "Current Events" é dependente dos atributos "TopicalRelevance" e "DaysOld". O primeiro é uma métrica que prediz quanto um documento satisfaz a respeito de uma dada consulta. No Inquirus, essa métrica é calculada a partir de uma heurística baseada na distância dos termos procurados no documento. Já o segundo atributo é um valor relativo a quanto recente é a data do conteúdo do documento.

As consultas no Inquirus são feitas informando palavras chave e escolhendo uma das categorias de acordo com as necessidades do usuário. Em (GLOVER et al.,2001), consultas de exemplo são mostradas, no qual se observa que a arquitetura proposta pode melhorar a eficiência dos sistemas de busca. Apesar das categorias auxiliarem os usuários a encontrar informações mais adequadas, tanto os mecanismos de busca fonte e os comportamentos dos atributos de cada categoria devem ser feito manualmente: o sistema permite que se entre com funções (descrições comportamentais) para cada uma das categorias. Além disto, a arquitetura permite que usuários definam como cada atributo deverá ser considerado, ou seja, o peso de cada um deles. Vale ressaltar que a determinação das categorias ou atributos é escolhida subjetivamente pelos usuários do sistema, de forma a expressar em alto nível as peculiaridades do domínio em que deverão ser procuradas as informações. Nesse sentido, os autores citam a necessidade de uma função que descubra e defina os comportamentos de cada categoria automaticamente, bem como, personalizá-las para cada usuário do sistema.

Placing Search in Context: The Concept Revisited (FINKELSTEIN et al., 2001): A noção de "*Contexto*" em sistemas de busca é referenciada em vários trabalhos com diferentes significados. Pode-se dizer a respeito da explicitação do contexto das consultas através de restrições feitas por categorias (como exemplificado em (GLOVER et al.,2001)). Além disso, podemos considerar o contexto como sendo a análise das informações e documentos que são manipulados ou apresentados para os usuários. Também se pode considerar o contexto como sendo o conjunto de informações previamente requisitadas pelos usuários, no qual tais informações podem auxiliar no ajuste de preferências, a fim de personalizá-las para os usuários. Ao contrário de tais significados, o trabalho de Finkelstein et al. (2001) define o contexto como tendo o seu significado mais natural: trata-se das informações que estão em volta dos termos da consulta do usuário e que provêm uma consistente interpretação para a mesma. Nesse sentido, a abordagem proposta no trabalho de Finkelstein et al. (2001) segue a premissa básica de que as informações deveriam ser processadas de acordo com o contexto que está por volta das mesmas. Isto permitiria melhores resultados, pois o contexto refletiria melhor as intenções do usuário.

Finkelstein et al. (2001) propõe uma abordagem que altera o comportamento da busca de acordo com o contexto em que a consulta está sendo executada. A proposta é implementada em um sistema denominado InteliZap, no qual as consultas derivadas do contexto são repassadas a este sistema, que por sua vez, faz com que as consultas sejam repassadas para outros sistemas de busca fonte (metabusca). Além disso, através do contexto, também se pode inferir domínios de aplicação relativos à consulta a fim de direcionar consultas para sistemas de busca especializados em um domínio de aplicação. Após serem feitas as consultas, um módulo de ranqueamento reordena os resultados a partir de uma métrica de similaridade entre os resumos dos documentos retornados e o próprio contexto.

Para alcançar esses objetivos, o sistema utiliza duas classes de heurísticas ao modificar as consultas através das informações obtidas do contexto. Dizemos que os termos da consulta são denominados de *texto* e todas as informações que estão em volta destes são denominadas *contexto*. Neste sentido, a primeira heurística é que quando mais frequente os termos do texto, então eles são menos relevantes e mais informações do contexto devem ser consideradas. Esta é uma função da amplitude do contexto em função da frequência dos termos do texto. Já a segunda heurística define que os termos do texto podem ser enfatizados na consulta através da duplicação dos mesmos ou utilizando-se operadores booleanos a fim de uniformizar a importância do texto perante aos termos do contexto. De acordo com Finkelstein, essas heurísticas provêm certo aprimoramento na relevância das informações retornadas para o usuário. Em (FINKELSTEIN et al.,2001), são exemplificadas consultas que são modificadas utilizando-se estas heurísticas.

No sistema InteliZap, ao visualizar um documento textual, o usuário pode selecionar termos contido no mesmo, no qual o texto consiste dos termos selecionados. Após isso, o sistema extrai as informações que estão ao redor desses termos (o contexto) utilizando as heurísticas citadas, e os envia juntamente com o texto para um servidor que processa as consultas e retorna os resultados.

Ao processar as consultas, o servidor do InteliZap identifica as mais importantes palavras do contexto. Para isto, emprega-se um algoritmo que agrupa

(algoritmo de *clustering*) os termos da consulta em seus diferentes aspectos. Consultas de um determinado aspecto são construídas ao combiná-las com termos importantes em relação a cada um dos aspectos. Estas consultas proveem um melhor resultado ao enfatizar aspectos semânticos da consulta original. Após isso, estas consultas além de serem enviadas para sistemas de busca, fonte de propósito geral são enviadas para sistemas de busca fonte de domínio específico, a partir de uma classificação da consulta inferida partir do contexto. Vale ressaltar que, para potencializar estas funções, InteliZap utiliza-se de uma rede semântica em uma abordagem baseada em vetores. Cada termo é representado como um vetor e cada domínio de aplicação é uma dimensão do espaço vetorial. Os vetores foram obtidos através da análise da frequência dos termos em uma série de domínios (no total 27 domínios).

Por fim, após as consultas serem enviadas para os diversos sistemas de busca fonte, um algoritmo de reordenação analisa a lista de resultados comparando-as semanticamente (também se utilizando da rede semântica) com o texto e o contexto. Isto é feito calculando-se a distância entre termos dos resumos dos resultados e os termos do texto e do contexto. A avaliação empírica da proposta apresenta bons resultados ao ser comparado com outros sistemas de busca tradicionais. Logo, a proposta de Finkelstein pode ser vista como uma maneira promissora para localizar informações da Web com melhor qualidade.

Towards a Fully Distributed P2P Web Search Engine (ZHOU et al., 2004): Zhou et al. (2004) ressalta um dos problemas existentes com os sistemas de busca centralizados: a não existência da colaboração entre humanos que potencialmente poderia aprimorar a qualidade dos resultados providos. Outro problema é a não consideração de preferências e hábitos do usuário. Desta forma, a mesma consulta feita por qualquer usuário sempre proverá os mesmos resultados. Neste sentido, um sistema que considere as preferências e permita a manutenção de perfis dos usuários introduziria certo grau de personalização nas buscas. Além disso, as experiências de usuários com interesses semelhantes poderiam ser exploradas coletivamente a fim de reduzir o domínio de interesse e consequentemente uma melhor filtragem dos resultados. Neste contexto, o trabalho de Zhou et al. (2004), propõe uma nova maneira para a busca de informações na Web, no qual os interesses e experiência dos usuários são compartilhados em uma abordagem *peer-to-peer* (P2P).

A abordagem de Zhou et al. (2004) é fundamentada em três mecanismos que foram implementados em um sistema denominado Coopeer: *(i)* PeerRank, uma técnica que ranqueia as páginas de acordo com o conjunto global de julgamentos feitos pelos usuários da rede P2P como um todo. *(ii)* Ao invés da indexação dos termos dos documentos, as próprias consultas são indexadas para representar os documentos. *(iii)* Um algoritmo de roteamento semântico, no qual os usuários com preferências semelhantes são auto-organizados a fim de compartilhar experiências em comum.

Em cada peer existe um índice que descreve o assunto de interesse das consultas de outros peers. Este índice é utilizado a fim de localizar outros peers. Vale ressaltar que, neste índice, peers que têm interesses semelhantes são priorizados. Assim, um peer envia consultas para peers que provavelmente tem interesse similar a consulta que ele deseja realizar. Para representar semanticamente os interesses de cada um dos peers, é utilizada uma forma de representação que considera as consultas que já foram anteriormente executadas por eles.

Além disto, é adotada uma representação dos documentos baseada nas próprias consultas. Este mecanismo assemelha-se com a construção convencional de índices invertidos a partir dos documentos, mas, em vez disso, os índices invertidos são criados a partir dos termos das consultas. Para cada peer, um índice invertido é mantido, cujos documentos inspecionados pelo peer são registrados. Tal representação é utilizada para organizar os documentos a fim de responder consultas feitas remotamente por outros peers.

O PeerRank considera que as avaliações e experiências de um usuário ao inspecionar documentos pode auxiliar o processo de descoberta para outros usuários. Desta forma, o PeerRank analisa as opiniões dos usuários que avaliaram os documentos para obter um ranqueamento dos mesmos. Documentos com uma maior quantidade de avaliações positivas têm uma melhor pontuação no ranqueamento. Isto é, a relevância de um documento é obtida ao examinar todos os peers que já o recomendaram. Nesse sentido, caso o peer que fez a consulta também recomende o documento, seu valor de pontuação no ranqueamento é recalculado. Vale ressaltar que, este novo valor de ranqueamento afeta todos os peers que já recomendaram o documento anteriormente.

É importante ressaltar que o sistema não foi desenvolvido para substituir sistemas de busca centralizados, mas atuar conjuntamente com eles. Na arquitetura do Coopeer existe um agente (Web-Searcher Agent) que é responsável por enviar as consultas para sistemas de busca centralizados, utilizar-se de seus resultados e assim permitir o ranqueamento através da colaboração. Uma das vantagens desta abordagem está sobre o fato que sistemas de busca centralizados teriam bastantes dificuldades em manter os índices e preferências individuais de cada peer, ao contrário , como se trata de um sistema distribuído, no Coopeer , as informações dos usuários são mantidas pelos próprios peers. Experimentos mostrados em (ZHOU et al.,2004) comprovam que a

colaboração e recomendação podem aprimorar qualitativamente os resultados obtidos.

Stanford WebBase Components and Applications (CHO et al., 2006): WebBase é a continuação do trabalho realizado na universidade de Stanford sob a versão original do sistema de busca Google. Por volta do fim da década de 90, foi ramificada a versão comercial do Google, e sua versão acadêmica foi renomeada para WebBase. Ao invés de uma versão comercial, o WebBase é um sistema aberto voltado para a pesquisa e experimentação, e é disponibilizado também para pesquisadores de fora da universidade de Stanford (CHO et al.,2006). Além disto, a principal contribuição deste trabalho é detalhar questões de projeto e implementação de um sistema de busca com um nível de detalhe ainda não apresentado.

WebBase é constituído por um conjunto de módulos que geralmente também estão presentes no projeto arquitetural de sistemas de busca na Web, dentre tais módulos devemos destacar: *(i) Crawler Module* que é responsável por coletar documentos na Web, *(ii) Indexer/Analysis Module* são responsáveis por criar e gerar índices e *(iii) Multicast Module* que é um módulo de distribuição de dados de alto volume permitindo que pesquisadores tenham acesso ao repositório do WebBase com o objetivo de desenvolver novas pesquisas.

O módulo de coleta (Crawler Module) é responsável por obter páginas da web e guardá-las em repositórios. Ele utiliza uma estrutura de dados do tipo fila para armazenar os endereços dos documentos que devem ser inspecionados, bem como, armazena os endereços dos documentos que já foram inspecionados a fim de evitar duplicação de processamento de documentos. Além disso, um componente denominado URL Seed Dispenser tem a função de distribuir endereços para um conjunto definido de coletores, de maneira que a coleta de documentos possa ser paralelizada. Nesse sentido, o projeto do WebBase considera os *websites* como uma unidade de paralelização (que podem ser vistos como o conjunto de documentos relativos a um domínio da Internet). Os autores justificam a necessidade de uma estrutura não monolítica e paralelismo na coleta de documentos, de acordo com os seguintes quesitos: dividir o espaço para armazenar estruturas de dados que mantém a lista de endereços a visitar e endereços visitados em cada coletor; baixa necessidade de coordenação, pois neste caso cada coletor é independente; e a modularização que permitiria especificar configurações e regras específicas para cada coletor.

Já os módulos de indexação (Indexer Module) e análise (Analysis Module) são responsáveis por manter e gerenciar os índices. O módulo de indexação é constituído de três índices: *offset index* possui um identificador numérico que corresponde à posição física de cada documento no repositório; *text index* é um índice invertido; e o *link index* suporta a busca em função de links, permitindo obter os documentos que um determinado documento aponta. Já o módulo de análise usa os índices produzidos pelo módulo de indexação para derivar novos e específicos índices, como por exemplo, um índice resultante após a utilização de uma técnica de ranqueamento.

O projeto do módulo de indexação leva em consideração os seguintes requisitos: a paralelização e distribuição, no qual existe uma distribuição do custo da computação dos índices, bem como, a compressão e *caching* dos índices para reduzir o tempo de indexação. O mecanismo de indexação é distribuído, no qual o índice é construído em três estágios. No primeiro estágio, cada indexador recebe um conjunto disjunto de documentos selecionados. No segundo estágio, cada indexador analisa os documentos, extrai estruturas intermediárias e as armazena temporariamente em disco. Após esse processo, as estruturas intermediárias de cada indexador são mescladas para criar os índices invertidos. O processo de geração de tais estruturas intermediárias ocorre em três fases: inicialmente os documentos são carregados em memória. Em seguida, com objetivo de se extrair os termos que os documentos contêm, cada documento é analisado e é gerada uma lista ordenada de termos. Por fim, as listas ordenadas são gravadas no disco. Projetado em fases, o processo de indexação é implementado como um *pipeline*, no qual cada *buffer* do pipeline trata de uma das fases. Experimentos mostram que a utilização desta abordagem diminui o tempo de indexação.

O WebBase permite acessar seu repositório através do modulo de transmissão (Multicast Module). Em outras palavras, podem-se recuperar documentos que já foram anteriormente coletados pelo sistema, a fim de, por exemplo, indexar ou analisar informações. Além disso, é possível depositar novas informações no repositório. Para o cliente do WebBase é fornecida uma interface que permite ao mesmo receber informações do repositório. Esta interface é programaticamente configurável, de forma que o cliente pode definir seus próprios mecanismos para tratar e analisar as informações conforme as mesmas vão sendo recebidas.

As principais contribuições deste trabalho foram sugestões e detalhes do projeto e implementação de um sistema de busca que pode indexar e gerenciar um alto volume de dados. Vale ressaltar que as direções de projeto citadas foram experimentadas e seus resultados podem ser encontrados em (CHO et al.,2006).

4. DISCUSSÃO E CONSIDERAÇÕES FINAIS

Gudivada et al. (1994) afirma que modelos de recuperação de informação são caracterizados por parâmetros que incluem a forma de representação de

consultas e documentos; e os métodos adotados para ranquear os resultados de uma consulta. Utilizamos a aplicação destes parâmetros de caracterização de modelos de recuperação de informação para resumir os sistemas de busca na Web apresentados e discutidos neste trabalho. Desta forma, estes parâmetros provêm uma visão de comparação entre os sistemas de busca na Web analisados e os resultados desta análise estão dispostos na Tabela 1.

Para o aprimoramento dos resultados de ranqueamento gerados em sistemas de busca reafirmamos as direções de projetos indicadas inicialmente por Gudivada et al. (1997). É sugerido um conjunto de direções para os trabalhos de recuperação de informações na Web. Estas direções incluem técnicas que considerem: feedback dos usuários; modificação das consultas e da representação dos documentos; e mecanismos baseados em agentes. Neste sentido, poder-se-ia obter feedback dos usuários com o objetivo de modificar a representação dos documentos e consultas a fim de melhorar a eficiência dos sistemas de busca. As consultas e documentos indexados poderiam ser expandidos, transformados e modificados de acordo com os itens que foram avaliados positivamente ou negativamente pelos usuários.

Tais abordagens são evidentes nos sistemas Inquirus (GLOVER et al.,2001), InteliZap (FINKELSTEIN et al.,2001) e Coopeer (ZHOU et al.,2004). As consultas no Inquirus são feitas informando palavras chave e escolhendo categorias de acordo com as necessidades do usuário com o objetivo de especializar a busca para determinados domínios. Já a proposta do InteliZap leva em conta informações contextuais do usuário para aprimorar o processamento de consultas. Além disso, experimentos utilizando o sistema Coopeer comprovam que a colaboração e recomendação entre usuários numa abordagem peer-to-peer pode aprimorar qualitativamente os resultados obtidos na busca.

Gudivada et al. (1997) também ressalta a importância da utilização de abordagens orientadas a agentes como forma de capturar de modelos do usuário ou modelos específicos de domínio que representem os interesses e preferências dos usuários de forma transparente. Neste contexto, agentes trabalhariam coletivamente com o objetivo de indexar e filtrar documentos, bem como, automaticamente direcionar informações relevantes para os usuários. O iBot (LOPEZ-SANCHEZ et al.,2000) é um dos exemplos deste tipo de sistema baseado em agentes.

Apesar disso, o trabalho de Lopéz-Sanchés et al. (2000) descreve os protocolos de comunicação entre estes agentes, mas em compensação, não apresenta aspectos de implementação ou validação da arquitetura, de forma que não é mostrado se existe algum tipo de aprimoramento qualitativo ou quantitativo na adoção desta arquitetura.

Tabela 1. Comparação entre os trabalhos analisados.

	Representação de Consultas	**Representação de Documentos**	**Estratégias de Ranqueamento**
Harvest (BOWMAN et al.,1994)	Palavras-chave; pares atributo-valor.	Não específica e configurável.	Não específica e configurável.
Altavista (GUDIVADA et al.,1994)	Palavras-chave.	Índices invertidos de termos.	Baseado em heurísticas.
Google (BRIN; PAGE, 1998)	Palavras-chave.	Índices invertidos de termos.	PageRank; heurísticas dependentes das propriedades dos termos contidos nos documentos.
iBot (LOPEZ-SANCHEZ et al.,2000)	Não detalhado.	Índices invertidos de termos.	Pesos atribuídos aos documentos.
Inquirus (GLOVER et al.,2001)	Palavras-chave; preferências expressas através de categorias.	Metabuscador.	Considera pesos associados às categorias.
InteliZap (FINKELSTEIN et al.,2001)	Dependente do contexto.	Metabuscador	Métricas de similaridade entre documentos e o contexto.
Coopeer (ZHOU et al.,2004)	Palavras-chave.	Índice invertidos de termos das consulta.	PeerRank
WebBase (CHO et al.,2006)	Não detalhado.	Índices invertidos de termos.	Não específico.

Dentre os sistemas discutidos destaca-se o sistema de busca Google (BRIN; PAGE, 1998). É importante ressaltar que este trabalho trouxe bastantes contribuições, pois apresentou novos métodos de ranqueamento (em especial o algoritmo PageRank) que têm como principal objetivo prover resultados de alta qualidade, de forma a tentar resolver de forma considerável os problemas em relação a qualidade dos resultados providos pelos sistemas de busca, questão já anteriormente apresentada por Gudivada et al. (1997).

Por adotar uma arquitetura escalável, o Google é adequado a ambientes onde existe uma grande quantidade de dados para ser indexada, tal como a Web. Também devemos considerar que o Google adota uma abordagem de sistema centralizado, ao invés de abordagens descentralizadas, tal como adotada pelo Harvest (BOWMAN et al.,1994), isto é justificado pelo alto custo de administração do último. Apesar disso, Brin & Page consideram que a adoção de abordagens descentralizadas seria ideal e isto poderia aprimorar drasticamente os resultados apresentados por sistemas de busca na Web, mas isto somente quando a redução de custo de administração destes sistemas for possível.

Em relação à escalabilidade de sistemas de busca algumas direções são indicadas por Cho et al. (2006) como resultados de experiência no projeto do sistema WebBase. Dentre tais direções de projeto, podemos destacar os seguintes aspectos gerais: paralelização da atividade coleta de documentos; controle centralizado das atividades de coleta de documentos; abordagem pipeline, distribuída e paralelizada da atividade de indexação; e distribuição em alto volume de dados que pode ser paralelizada.

Por fim, como considerações finais, deve-se destacar que a principal contribuição deste trabalho está em compilar e apresentar como foram projetados os principais sistemas de busca na Web e as mudanças de paradigmas que ocorreram no projeto destes sistemas com o passar do tempo. Espera-se que as discussões apresentadas, neste trabalho, difundam o conhecimento sobre sistemas de recuperação de informação e sistemas de busca na Web.

5. REFERÊNCIAS BIBLIOGRÁFICAS

BELKIN, N. J.; CROFT, W. B. 1992. **Information filtering and information retrieval: two sides of the same coin?**. Commun. ACM 35, 12 (Dec. 1992), 29-38.

BOWMAN, C. M.; DANZIG ,P. B; HARDY, D. R; MANBER, U; SCHWARTZ, M. F. **Harvest: A Scalable, Customizable Discovery Access System.** Technical Report CU-CS-732-94, Department of Computer Science, University of Colorado, Boulder, August 1994.

GUDIVADA, V.N.; RAGHAVAN, V.V.; GROSKY, W.I.; KASANAGOTTU, R., **Information retrieval on the World Wide Web**, *Internet Computing, IEEE* , vol.1, no.5, pp.58-68, Sep/Oct 1997.

BAEZA-YATES, R. A.; RIBEIRO-NETO, B., **Modern Information Retrieval**, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 1999.

HAWKING, D. **Web Search Engines: Part 1**, *Computer*, vol. 39, nº6, pp. 86-88, Aug., 2006.

HAWKING, D. **Web Search Engines: Part 2**, *Computer*, vol. 39, no. 8, pp. 88-90, Aug., 2006.

LAM, S. **The Overview of Web Search Engines**. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.28.6344, 2001.

BRIN, S., PAGE L., **The Anatomy of a Large-Scale Hypertextual Web Search Engine**, In Proc. of the 7th Int'l World Wide Web Conf., 1998, 26 pages.

ARASU, A.; CHO, J.; GARCIA-MOLINA, H.; PAEPCKE, A.; RAGHAVAN, S. (2001), **Searching the web**, ACM Transactions on Internet Technology, Vol. 1 No.1, pp.2-43.

BARROSO, L.A.; DEAN, J.; HOLZLE, U., **Web search for a planet: The Google cluster architecture**, *IEEE Micro* , vol.23, no.2, pp. 22-28, March-April 2003.

M. LOPEZ-SANCHEZ, F.J; MARTIN, J; GARCIA, X; CANALS, X; DRUDIS, I; RUIZ; A. REYES. **Agent communication inside an Internet search engine,** in Proc. Simposio Espantildeol de Informatica Distribuida, May 5, 2000.

GLOVER, E. J.; LAWRENCE, S.; GORDON, M. D.; BIRMINGHAM, W. P.; GILES, C. L., **Web Search---Your Way**, Communications of the ACM, v.44 n.12, p.97-102, December 2001.

FINKELSTEIN, L.; GABRILOVICH, E.; MATIAS, Y.; RIVLIN, E.; SOLAN, Z.; WOLFMAN, G.; RUPPIN, E. **Placing search in context: the concept revisited**. In Proceedings of the Tenth International World Wide Web Conference, 2001.

ZHOU, J.; LI, K.; TANG, L., **Towards a fully distributed P2P Web search engine**, *Distributed Computing Systems, 2004. FTDCS 2004. Proceedings. 10th IEEE International Workshop on Future Trends of* , vol., no., pp. 332-338, 26-28 May 2004.

CHO, J.; GARCIA-MOLINA, H.; , HAVELIWALA, T.; LAM, W.; , PAEPCKE, A.; RAGHAVAN, S.; WESLEY, G. **Stanford WebBase components and applications**, ACM Transactions on Internet Technology (TOIT), v.6 n.2, p.153-186, May 2006.