Universidade Federal de Minas Gerais
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Uma Abordagem Baseada em Gênero para Coleta Temática de Páginas da Web

Guilherme Tavares de Assis

Orientador: Alberto Henrique Frade Laender

Co-orientador: Marcos André Gonçalves

Belo Horizonte, Março 2008

Guilherme Tavares de Assis

Uma Abordagem Baseada em Gênero para Coleta Temática de Páginas da Web

Orientador: Alberto Henrique Frade Laender

Co-orientador: Marcos André Gonçalves

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Minas Gerais, como requisito parcial para a obtenção do grau de Doutor em Ciência da Computação.

Belo Horizonte

27 de Março de 2008

Guilherme Tavares de Assis

A Genre-Aware Approach to Focused Crawling of Web Pages

Advisor: Alberto Henrique Frade Laender

Co-advisor: Marcos André Gonçalves

Thesis presented to the Graduate Program in Computer Science of the Federal University of Minas Gerais in partial fulfillment of the requirements for the degree of Doctor in Computer Science.

Belo Horizonte

March 27, 2008

À minha mãe, Cleonice Tavares de Assis,
exemplo de amor, afeto e força.

À memória de meu pai, Joaquim de Assis Filho,
exemplo de saber e vitória.

Agradecimentos

Toda trajetória exige preparação, esforço e dedicação. A chegada ao ponto desejado deve ser comemorada com muita alegria e reconhecimento. É difícil, neste momento final, encontrar as palavras certas para expressar o tanto que necessito agradecer. O espaço torna-se pequeno demais mas, mesmo assim, quero registrar os meus agradecimentos por tão esperada conquista.

Ao meu orientador, Alberto Henrique Frade Laender, por ter acreditado em mim desde o início da trajetória em busca do doutoramento, proporcionando-me a grande honra de ser seu orientando, e pontuando o trabalho desenvolvido com seu talento, capacidade e compromisso.

Aos meus co-orientadores, Marcos André Gonçalves e Altigran Soares da Silva, parceiros fundamentais no trabalho e publicações.

Ao MCT/CNPq/CT-INFO através dos projetos GERINDO e 5S-VQ, ao UOL (www.uol.com.br) através do Programa UOL Bolsa Pesquisa, e à CAPES através dos programas de fomento a Pós-Graduação, pelo apoio financeiro para a realização do trabalho desenvolvido.

Aos meus amigos do Departamento de Ciência da Computação e do Laboratório de Banco de Dados, em particular Kissia, Karla, Joyce, Anderson, Denilson, Evandrino, Bárbara, Allan Jones, Rodrygo e Thierson, pela sinergia, apoio e companheirismo na luta diária pelo domínio e produção de conhecimento.

Aos meus parentes e amigos da vida, do Uni-BH e da FABRAI (onde tudo começou), pelo incentivo de todas as horas.

Ao meu grande amigo Geraldo Boaventura, pelo apoio, força, participação e presença constantes ao longo do doutoramento.

Aos meus irmãos, Alexandre, Mazé e Aninha, companheiros fiéis e fortíssima trindade de amor e fraternidade.

Aos meus cunhados, Raquel e Pedro, por fazerem a minha família crescer em número e afeto.

Ao meu sobrinho Felipe que, desde a barriga da mãe, já mandava sinais de força e coragem.

Às minhas sobrinhas gêmeas, Catarina e Vitória, exemplos de superação e vontade de viver, pela alegria dupla em minha vida e por terem concedido o seu dia de nascimento para a defesa do meu trabalho.

À minha maravilhosa e amada sobrinha-afilhada Júlia, por todas as vezes em que, ao seu lado, divirto-me e descubro que a vida pode ser ainda mais linda.

Ao meu pai, Joaquim, principal incentivador de minha educação, farol de luz e sabedoria que, do alto, segue iluminando todos os meus passos.

À minha mãe, Cleonice, amor meu de todos os dias, berço dos melhores exemplos de vida e fé, razão maior de todas as conquistas.

À Nossa Senhora, por proteger-me sempre, sob seu manto abençoado de amor e generosidade.

E principalmente a Deus, quem agradeço por ter colocado pessoas tão especiais em todos os momentos de minha vida: muito obrigado por tudo e sempre!

Resumo

Coletores temáticos apresentam o propósito maior de coletar páginas da Web que sejam relevantes a um tópico ou interesse específico do usuário, sendo importantes para uma grande variedade de aplicações. Em geral, eles funcionam tentando localizar e coletar todos os tipos de páginas que estejam relacionadas a um determinado tópico de interesse. Contudo, alguns usuários podem não estar simplesmente interessados em algum documento sobre um tópico; na verdade, podem estar interessados em recuperar documentos de um determinado estilo ou gênero referente ao tópico. Nesta tese, descrevemos uma abordagem para coleta temática que explora não somente informação relacionada ao conteúdo mas também informação de gênero presente em páginas Web para guiar o processo de coleta. Essa abordagem foi projetada para situações em que o tópico específico de interesse pode ser expresso por dois conjuntos de termos: o primeiro conjunto descrevendo aspectos de gênero das páginas desejadas e o segundo conjunto relacionado ao assunto ou conteúdo destas páginas. Além disso, uma das características da nossa abordagem consiste no fato de não necessitar de algum treinamento a priori nem de algum outro tipo de processamento prévio. A eficácia, a eficiência e a escalabilidade da abordagem proposta são demonstradas por um conjunto de experimentos, envolvendo a coleta de páginas Web referentes a planos de ensino de disciplinas do curso de ciência da computação, ofertas de trabalho na área de ciência da computação e ofertas de venda de equipamentos de informática. Tais experimentos mostram que coletores temáticos construídos de acordo com a nossa abordagem baseada em gênero alcançam níveis de F1 superiores a 88%, necessitando a análise de não mais do que 60% das páginas visitadas para localizar 90% das páginas relevantes. Além disso, analisamos experimentalmente o impacto da seleção de termos em nossa abordagem, variando a quantidade de termos de gênero e conteúdo usados para guiar um processo de coleta e avaliando uma estratégia proposta para geração semi-automática de tais termos. A análise mostra que um pequeno conjunto de termos selecionados por um especialista é geralmente suficiente para produzir resultados bons e que tal estratégia para geração semi-automática de termos é muito eficaz em suportar a tarefa de selecionar os conjuntos de termos necessários para guiar um processo de coleta utilizando nossa abordagem.

Abstract

Focused crawlers have as their main goal to crawl pages that are relevant to a specific topic or user interest, playing an important role for a great variety of applications. In general, they work by trying to find and crawl all kinds of pages deemed as related to an implicitly declared topic. However, users are often not simply interested in any document about a topic, but instead they may want only documents of a given type or genre on that topic to be retrieved. In this thesis, we describe an approach to focused crawling that exploits not only content-related information but also genre information present in Web pages to guide the crawling process. This approach has been designed to address situations in which the specific topic of interest can be expressed by specifying two sets of terms, the first describing genre aspects of the desired pages and the second related to the subject or content of these pages. Moreover, our approach does not require training or any kind of preprocessing. The effectiveness, efficiency and scalability of the proposed approach are demonstrated by a set of experiments involving the crawling of pages related to syllabi of computer science courses, job offers in the computer science field and sale offers of computer equipments. These experiments show that focused crawlers constructed according to our genre-aware approach achieve levels of F1 superior to 88%, requiring the analysis of no more than 60% of the visited pages in order to find 90% of the relevant pages. In addition, we experimentally analyze the impact of term selection on our approach by varying the number of genre and content terms used to guide a crawling process and evaluate a proposed strategy for semi-automatic generation of such terms. This analysis shows that a small set of terms selected by an expert is usually enough to produce good results and that such a strategy is very effective in supporting the task of selecting the sets of terms required to guide a crawling process using our approach.

Resumo Estendido

Originalmente, o documento desta tese foi redigido em inglês sob o título *A Genre-Aware Approach to Focused Crawling of Web Pages*. Com o objetivo de atender às normas da Universidade Federal de Minas Gerais, foi redigido este resumo estendido, em português, que apresenta uma descrição abreviada de cada capítulo contido na tese.

Capítulo 1 - Introdução

No Capítulo 1, é apresentada uma introdução sobre o trabalho desenvolvido, envolvendo a motivação para a sua realização, objetivos principais e específicos, visão geral e contribuições do trabalho, principais resultados, trabalhos diretamente relacionados ao tema abordado e a organização da tese.

Máquinas de busca de propósito geral, como Google ou MSN, não resolvem bem o problema de localizar páginas da Web referentes a um tópico específico, já que as coleções de páginas geradas por elas são bem volumosas e, geralmente, as consultas dos usuários são curtas envolvendo pouca informação. Desta forma, em muitas situações, a capacidade de responder às consultas dos usuários é limitada. Neste contexto, coletores temáticos servem para gerar coleções de páginas menores e restritas, já que apresentam o propósito maior de coletar páginas que sejam, da melhor forma possível, relevantes a um tópico ou interesse específico do usuário, a partir de uma especificação mais detalhada do que se deseja coletar. Esses coletores são importantes para uma grande variedade de aplicações e serviços presentes na Web, tais como, bibliotecas digitais, agentes pessoais, inteligência competitiva e diretórios de grandes portais. Além disso, coletores temáticos, em comparação com coletores tradicionais usados pelas máquinas de busca de propósito geral, promovem economia de recursos e escalam, já que não cobrem a Web inteira.

O desafio de identificar sub-espaços da Web específicos e relevantes, de acordo com algum tema, é tipicamente denominado de *construção de inteligência* em uma estratégia de coleta. Esta inteligência é usualmente alcançada por meio de heurísticas apropriadas que direcionam o processo de coleta e determinam a relevância de uma página em relação a um tópico ou interesse específico do usuário. Várias estratégias atuais de coleta temática utilizam

classificadores de texto para determinar tal relevância, com um custo adicional para serem treinados. Além disso, devido à generalidade das situações em que essas estratégias são aplicadas, elas alcançam baixos níveis de revocação e precisão, geralmente entre 40% e 70%. Distintamente, visando melhorar a escabilidade e a eficácia do processo de coleta temática, propomos uma abordagem que não necessita de treinamento apriori e está voltada para atender situações específicas. De uma forma geral, nossa abordagem consiste em considerar as evidências de gênero[1] e conteúdo presentes em uma determinada página e estabelece um grau de similaridade entre tais evidências e o tópico específico de interesse. Além disso, para melhor direcionar o processo de coleta temática, nossa abordagem utiliza uma eficiente política de enfileiramento das ligações entre páginas a serem seguidas pelo coletor.

Portanto, esta tese tem, como objetivo principal, estabelecer um arcabouço que permita a construção de coletores temáticos eficazes, eficientes e escaláveis, sem a necessidade de um treinamento a priori ou qualquer tipo de preprocessamento. Especificamente, propomos uma abordagem para coleta temática que é útil em situações onde um tópico de interesse possa ser expresso por meio de dois conjuntos distintos de termos: o primeiro descrevendo aspectos de gênero das páginas desejadas e o segundo referente ao assunto ou conteúdo descrito nessas páginas. Exemplos onde isto acontece incluem planos de ensino de disciplinas específicas, currículos de profissionais ou ofertas de emprego de uma determinada área, ofertas de venda de produtos específicos, documentação de software, bulas de medicamentos, etc.

Uma questão a ser considerada em nossa abordagem para coleta temática é a seleção dos termos necessários para descrever o gênero e o conteúdo das páginas de interesse que, geralmente, requer a assistência de um especialista no domínio de aplicação. Embora não seja uma tarefa complexa, a seleção de tais termos pode causar impacto na eficácia de um coletor construído de acordo com a abordagem proposta. Logo, nesta tese, também analisamos o impacto da seleção de termos em nossa abordagem, além de avaliar uma estratégia simples, mas efetiva, para a geração semi-automática dos termos de gênero e conteúdo necessários.

Em resumo, as principais contribuições do nosso trabalho são:

[1]Por gênero (dentre os seus significados possíveis [5]), entendemos o tipo, a categoria ou o estilo de texto de documentos específicos. Uma noção similar de gênero é explorada também em [50]. Vários trabalhos sobre *classificação de páginas Web baseada em gênero* conceituam gênero como a forma ou estrutura padrão de uma página Web, representando seu estilo de texto.

1. Uma descrição detalhada de uma abordagem inovadora para coleta temática que explora aspectos relacionados ao gênero e ao conteúdo das páginas desejadas e não requer uma fase de treinamento. Além disso, tal abordagem adota uma política de enfileiramento que altera dinamicamente a prioridade de coleta das páginas não visitadas, de acordo com o conteúdo das mesmas, proporcionando uma estratégia de coleta eficiente e escalável.

2. Um conjunto extenso de experimentos, envolvendo três domínios de aplicação distintos, que demonstra a eficácia, a eficiência e a escalabilidade de nossa abordagem.

3. Uma análise experimental do impacto do número de termos em nossa abordagem e a proposta e avaliação de uma estratégia para a geração semi-automática dos termos de gênero e conteúdo necessários para expressar as páginas desejadas.

Capítulo 2 - Coleta de Páginas da Web

O Capítulo 2 apresenta uma visão geral sobre coleta de páginas da Web, que constitui o tema central desta tese.

Devido à grande popularidade e, especialmente, ao rápido crescimento da Web, novas técnicas têm sido propostas e analisadas para auxiliar usuários a, efetivamente, localizar a informação que necessitam, em um tempo satisfatório, sem muita dificuldade. O acesso à informação na Web é feito, basicamente, através de máquinas de busca que exploram a estrutura em grafo da Web no intuito de localizar páginas relevantes a uma específica consulta. Para permitir isto, uma máquina de busca típica trabalha sobre uma coleção de páginas indexadas, gerada a partir da coleta do maior número possível de páginas da Web. Isto é realizado por programas específicos denominados coletores. Um coletor tradicional coleta páginas da Web, começando por uma página-semente e seguindo as ligações contidas nela, visitando assim, outras páginas. Tal processo se repete com as novas páginas, a partir das quais novas ligações são seguidas, até percorrer um número suficiente de páginas ou alcançar algum objetivo especificado. Então, um outro programa, denominado indexador, lê as páginas coletadas e cria um índice baseado nas palavras contidas nas mesmas. Cada máquina de busca usa um algoritmo próprio para criar seu índice de tal forma que, idealmente, somente resultados significativos sejam retornados para cada consulta formulada por um usuário.

Atualmente, coletores são amplamente usados. Coletores de máquinas de busca tentam visitar o maior número de páginas da Web para gerar suas coleções de páginas indexadas. Outros coletores podem visitar também várias páginas, mas procurando por determinada informação. Executar um coletor é uma tarefa desafiante. Existem questões referentes a desempenho e confiabilidade que devem ser bem tratadas, além de questões sociais e éticas. Coletar envolve interagir com centenas de milhares de servidores da Web que não são controlados pelo coletor. Então, externamente, o coletor deve evitar sobrecarregar servidores da Web e, internamente, o coletor deve tratar um grande volume de dados. A não ser que existam recursos computacionais e de tempo ilimitados, o coletor deve cuidadosamente decidir que ligações deve visitar e em que ordem, ou seja, ele deve definir a estratégia de escalonamento de ligações a serem visitadas. O coletor deve também decidir com qual freqüência deve revisitar as páginas para informar sobre as mudanças na Web. Além disso, o coletor deve assegurar o tratamento correto das condições de exceção e erros, e uma boa cobertura do conjunto de páginas coletadas com respeito ao conjunto de todas as páginas existentes.

Enquanto alguns coletores exaustivamente coletam páginas da Web, outros incorporam o conceito de "foco" para gerar coleções referentes a um tópico específico. Além disso, alguns coletores especiais estão direcionados a coletar páginas na Web oculta, que consiste na porção da Web constituída por informações presentes em bancos de dados de distintas organizações e geralmente não indexada por máquinas de busca tradicionais. Assim, esse capítulo trata esses três tipos de coletores a saber, coletores tradicionais, coletores temáticos e coletores para Web oculta, envolvendo conceitos, características, modelos de funcionamento, estratégias de escalonamento, métricas de avaliação e exemplos existentes na literatura.

Capítulo 3 - Coleta Temática Baseada em Gênero

No Capítulo 3, é apresentada uma descrição detalhada da nossa abordagem baseada em gênero para coleta temática, envolvendo características gerais, modelo de funcionamento, estratégia para determinação da relevância das páginas ao tópico desejado e políticas de enfileiramento.

Nossa abordagem para coleta temática baseia-se no fato de que alguns tópicos específicos de interesse podem ser representados por meio de evidências que caracterizem dois aspectos distintos: o gênero e o conteúdo das páginas. Por exemplo, se um usuário deseja coletar

páginas da Web que incluem planos de ensino de disciplinas de bancos de dados, um coletor temático deveria analisar uma página específica considerando, separadamente, os termos presentes em tal página que indicam se ela corresponde a um plano de ensino (termos que caracterizam o gênero "plano de ensino de uma disciplina") e os termos que indicam se a página está relacionada à área de bancos de dados (termos que aparecem no conteúdo do plano de ensino de uma disciplina de bancos de dados). Por outro lado, um coletor temático guiado por um classificador analisa o conteúdo de uma página específica da Web, mas não considera separadamente os dois tipos de informação. Assim, páginas que incluem planos de ensino de outras disciplinas ou páginas que dizem respeito à área de bancos de dados poderiam ser selecionadas por este coletor como sendo relacionadas ao plano de ensino de uma disciplina de bancos de dados (erros de precisão), da mesma forma que páginas com gênero e conteúdo mal especificados, mas correspondendo a planos de ensino de uma disciplina relacionada a bancos de dados, poderiam ser classificadas como não sendo do tópico desejado (erros de revocação).

Assim, nossa abordagem considera um conjunto de heurísticas para guiar um coletor, de tal forma que permita a análise separada do gênero e do conteúdo de uma página da Web. Além disso, a nossa abordagem considera o nome (texto) da URL da página, já que pode incluir termos relacionados ao gênero ou ao conteúdo do tópico de interesse desejado. Nossas heurísticas foram projetadas com dois objetivos principais: melhorar o nível de F1 (medida que combina as medidas tradicionais de precisão e revocação usadas na área de recuperação de informação) do processo de coleta e fazer com que as páginas relevantes sejam coletadas o quanto antes.

Dentro deste contexto, o Capítulo 3 encontra-se organizado em duas seções. Inicialmente, a primeira seção discute como nossa abordagem determina a relevância de uma página da Web em relação aos termos de gênero e conteúdo especificados. Em seguida, a segunda seção detalha nossa abordagem, descrevendo o algoritmo que implementa o processo de coleta.

Capítulo 4 - Avaliação Experimental

No Capítulo 4, são descritos os experimentos que realizamos e conduzimos para demonstrar a eficácia, eficiência e escalabilidade de nossa abordagem para coleta temática, e discutidos os resultados obtidos.

Para conduzir nossos experimentos, implementamos um coletor baseado em nossa abordagem para coleta temática. Nos experimentos, o coletor foi executado no contexto da Web Brasileira com o propósito de coletar páginas relacionadas a três tópicos: (1) planos de ensino de disciplinas específicas de um curso de ciência da computação, (3) ofertas de emprego na área de ciência da computação e (3) ofertas de venda de equipamentos de informática. Estes três tópicos estão relacionados ao mesmo assunto geral ("computação") mas suas páginas de interesse são de diferentes gêneros ("plano de ensino de uma disciplina", "oferta de emprego" e "oferta de venda").

Para o primeiro tópico, três disciplinas com características distintas foram escolhidas: "bancos de dados", que é uma disciplina cujo conteúdo está bem consolidado, "estruturas de dados", que é também uma disciplina bem consolidada porém seu conteúdo pode estar disperso em várias disciplinas, e "recuperação de informação", que é uma disciplina cujo conteúdo ainda não se encontra bem consolidado devido à sua rápida e recente evolução. Além disso, cinco páginas-semente de instituições brasileiras de ensino superior foram usadas. Para os outros dois tópicos, ofertas de emprego em computação e ofertas de venda de equipamentos de informática, usamos dez páginas-semente. Para cada tópico, um especialista definiu o conjunto de termos necessários para representar o gênero e o conteúdo das páginas desejadas, assim como o conjunto-resposta contendo as páginas relevantes dentre as páginas visitadas pelo coletor, para nos permitir medir a eficácia de nossa abordagem. Percebemos que alguns tópicos podem necessitar de mais termos do que outros devido à sua amplitude e complexidade, e que a especificação dos termos é uma tarefa bem simples quando comparada com o esforço requerido para treinar um classificador.

Para avaliar a eficácia da nossa abordagem, medimos o nível de F1 de cada processo de coleta executado por nosso coletor. Além disso, para estabelecer um *baseline* de comparação, desenvolvemos um coletor temático guiado pelo classificador SVM-RBF, um dos melhores classificadores para este tipo de aplicação, e o executamos para cada tópico de duas formas distintas: (1) seguindo a estratégia de coleta temática tradicional e (2) seguindo a política de enfileiramento que propusemos.

Os resultados obtidos pelos processos de coleta realizados, juntamente com as comparações devidas em relação ao *baseline*, estão descritos nas seções do Capítulo 4, levando em consideração os quesitos eficácia, eficiência e escalabilidade. De uma forma geral, os resultados desses experimentos mostram que coletores temáticos, construídos de acordo com a nossa

abordagem, alcançaram níveis de F1 superiores a 88%, necessitando a análise de não mais do que 60% das páginas visitadas para localizar 90% das páginas relevantes. Além disso, na última seção do Capítulo 4, foram descritos os experimentos realizados para avaliar diferentes e possíveis políticas de enfileiramento a serem usadas em nossa abordagem para coleta temática.

Capítulo 5 - Seleção de Termos

O Capítulo 5 investiga o impacto da seleção de termos em nossa abordagem para coleta temática.

Como já mencionado, nossa abordagem requer um especialista para definir um conjunto de termos que descrevam o gênero e o conteúdo das páginas de interesse. Embora não seja uma tarefa complexa, a seleção desses termos pode ter um impacto na eficácia e na eficiência de um coletor construído de acordo com a nossa abordagem. Assim, é necessário saber como termos de gênero e conteúdo devem ser definidos por um especialista para produzir resultados significantes. Além disso, para reduzir este esforço, é importante saber se termos gerados semi-automaticamente produziriam resultados que sejam tão bons quanto aqueles proporcionados com termos definidos por um especialista.

Assim, mais especificamente, reportamos os resultados de um estudo que realizamos para avaliar o impacto do número de termos, especificados para descrever o gênero e o conteúdo das páginas de interesse, no resultado de coletas realizadas. Além disso, propomos e avaliamos experimentalmente uma estratégia simples e efetiva para geração semi-automática desses termos, que demonstra que nossa abordagem não depende de um especialista para obter bons resultados. O Capítulo 5 encontra-se organizado em duas seções. Na primeira seção, é analisado o impacto da seleção de termos no resultado de coletas realizadas de acordo com a nossa abordagem. Na segunda seção, é apresentada e avaliada a estratégia proposta para geração semi-automática de termos.

De uma forma geral, os resultados dos experimentos realizados mostram que um pequeno conjunto de termos definidos por um especialista é geralmente suficiente para produzir bons resultados e que a estratégia proposta para geração semi-automática de termos é bastante eficaz como forma de auxiliar a tarefa de selecionar os conjuntos de termos necessários para guiar um processo de coleta.

Capítulo 6 - Conclusões

Finalmente, no Capítulo 6, são apresentadas as conclusões da tese e algumas direções para trabalhos futuros.

Os maiores benefícios de nossa abordagem para coleta temática, em relação às abordagens guiadas por classificadores, são: (1) melhoria do nível de F1 nos processos de coleta, (2) maior eficiência nos processos de coleta já que somente é preciso visitar um pequeno volume de páginas para coletar uma grande porcentagem de páginas relevantes e (3) maior escalabilidade já que nossa abordagem adota um conjunto simples de heurísticas para determinar a relevância de uma página e guiar o processo de coleta, não necessitando de uma fase de treinamento ou qualquer preprocessamento. Além disso, especificar um conjunto de termos de gênero e conteúdo para um processo de coleta não é uma tarefa complexa. Um pequeno conjunto de termos selecionados por um especialista é geralmente suficiente para produzir bons resultados. A tarefa de selecionar um conjunto de termos é facilitada pelo fato que a atenção do especialista deve estar focada nos termos de conteúdo, já que o gênero pode ser caracterizado por um pequeno número de termos. Além disso, nossa estratégia para geração semi-automática de termos mostrou ser bem efetiva em assistir um especialista (ou mesmo um simples usuário) na tarefa de especificar os conjuntos de termos de gênero e conteúdo para um processo de coleta.

Baseado no desenvolvimento e nos resultados obtidos nesta tese, algumas direções para trabalhos futuros são:

- utilizar nossa abordagem baseada em gênero em aplicações reais de grande vulto;
- avaliar nossa estratégia para geração semi-automática de termos, variando o número de páginas usadas como exemplo;
- estender nossa abordagem para considerar, além de gênero e conteúdo, outros aspectos tais como estrutura da página e indicações de região geográfica;
- investigar se grafos de contexto podem ser explorados em nossa abordagem para melhorar os resultados de coleta;
- usar a idéia central de nossa abordagem, ou seja, a distinção entre gênero e conteúdo por meio de conjuntos de termos específicos, em outros problemas de recuperação de informação tais como pesquisa e classificação.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Focused crawlers are an important class of programs that have as their main goal to efficiently crawl Web pages that are relevant to a specific topic of interest. This thesis proposes a novel approach to focused crawling aimed at crawling pages related to specific topics that can be expressed in terms of genre (text style) [5, 42] and content information. Thus, this approach was designed to situations in which the specific topic of interest can be expressed by two distinct sets of terms, the first describing genre aspects of the desired pages, and the second related to the subject or content of these pages, requiring no training nor any kind of preprocessing. Examples of such situations are syllabi of specific courses and curricula vitae of professionals or job offers in a particular field. The experimental results show that focused crawlers implemented according to our approach are more effective for genre-based crawling than classifier-based focused crawlers.

This chapter provides an introduction to our work and is organized as follows. Section 1.1 presents the motivation for our work. Section 1.2 provides an overview of the work, describing its objectives and the main results obtained. Section 1.3 summarizes the major contributions. Section 1.4 addresses related work. Finally, Section 1.5 outlines the thesis' chapters.

1.1 Motivation

Searching the Web is a difficult task. According to [29], machine learning techniques have been applied to one part of this task, namely ranking indexed pages by their estimated relevance with respect to user queries. Early search engines ranked pages mainly based on their lexical similarity to the query. The key strategy is to devise the best weighting

algorithm to represent Web pages and queries in a vector space, such that closeness in this space would be correlated with semantic relevance. More recently, the structure of hypertext links has been recognized as a powerful new source of evidence for Web semantics [1, 27, 36]. Several machine learning techniques have been employed for link analysis, so that a page's linking structure, together with its content, can be used to estimate its relevance. However, no matter how sophisticated the designed ranking algorithm is, the results can only be as good as the pages indexed by the search engine. This draws the attention to another aspect of the Web searching task, namely Web crawling, i.e., the gathering of pages to be indexed. The visible Web, which is very voluminous, offers a challenging information retrieval problem with respect to this task [26]. Search engines use Web crawlers to fetch as many Web pages as possible in order to generate a collection of relevant indexed pages.

Thus, due to the fact that general purpose search engines, such as Google or MSN, handle voluminous collections of indexed pages generated by Web crawlers, whose content addresses a wide spectrum of topics, in general, they do not solve well the problem of searching Web pages referring to a specific topic. In addition, the user-submitted queries are usually very short and do not carry enough information. Thus, in many situations, the capability of these search engines in answering specific queries is very limited. In this context, focused crawlers or topical crawlers [9, 27, 29, 30, 35, 45] are special purpose crawlers that serve to get from the Web smaller and more restricted collections of pages. They have as their main goal to efficiently crawl pages that are, in the best possible way, relevant to a specific topic or user interest. Focused crawlers are important for a great variety of applications, such as digital libraries [38], personal agents, competitive intelligence [34], and large Web directories [28], to name a few. Additionally, when compared with traditional crawlers used by general purpose search engines, they reduce the use of resources and favor the scalability of the crawling process, since they avoid the need for covering the entire Web.

The challenge of identifying specific and relevant sub-spaces of the Web, according to a theme, is typically referred to as "construction of intelligence" in some crawling strategies [35]. This process is usually carried out by means of appropriate heuristics, which guide the crawling process and aim at determining how relevant a certain page is to a specific topic of interest. Most of the current strategies rely on text classifiers to determine such relevance [1, 9, 16, 22, 35, 36, 44, 45], with the additional cost of having to train the classifiers. Moreover, due to the generality of the situations in which such strategies are applied, they reach low levels of recall and precision, usually between 40% and 70%. Other kinds of focused

crawler [7, 20, 27, 29, 30, 34, 38] also present similar levels of recall and precision.

1.2 Overview of our Work

We took a different approach to focused crawling by explicitly considering genre and content-related aspects of a page to evaluate its relevance to the information needs expressed by a user. We notice that users are often not simply interested in any document about a topic, but instead many times they want only documents of a given type or genre on that topic to be retrieved. By genre (among its several possible meanings [5]), we mean the kind, sort, or style of specific documents. A similar notion of genre is also explored in [50]. This is also the dominant view of several works on *genre-based Web page classification* [10, 51, 42, 46], which consider the concept of genre as related to the form or standard structure of a Web page, representing its text style.

The main goal of our work was to build a framework to allow the construction of effective, efficient and scalable focused crawlers that take into consideration both the genre and the content of the desired pages. More specifically, we proposed a focused crawling approach designed to situations in which the specific topic of interest can be expressed by two distinct sets of terms, the first describing genre aspects of the desired pages, and the second related to the subject or content of these pages, thus requiring no training nor any kind of preprocessing. Examples of such situations include syllabi of specific courses, curricula vitae of professionals or job offers in a particular field, sale offers of specific products, software documentation, medicine descriptions, etc. Thus, distinctly from previous approaches in the literature [16, 27, 29, 34, 35, 36], ours exploits both the genre and the content of the Web pages to guide the crawling process. For instance, Figures 1.1, 1.2 and 1.3 illustrate, respectively, a typical course syllabus page, a typical computer science job offer page and a typical computer equipment sale offer page found on the Web. As we can see, the genre and content terms that characterize these pages as such are easily recognized.

Thus, in this thesis, we present a detailed description of this novel genre-aware approach to focused crawling and describe the results of a set of experiments we have performed involving three application domains that demonstrate its effectiveness, efficiency and scalability. The results of these experiments show that focused crawlers constructed according to this genre-aware approach achieve levels of F1 (a measure that combines the traditional precision and recall measures used in the IR area [4]) superior to 88%, which represents an average

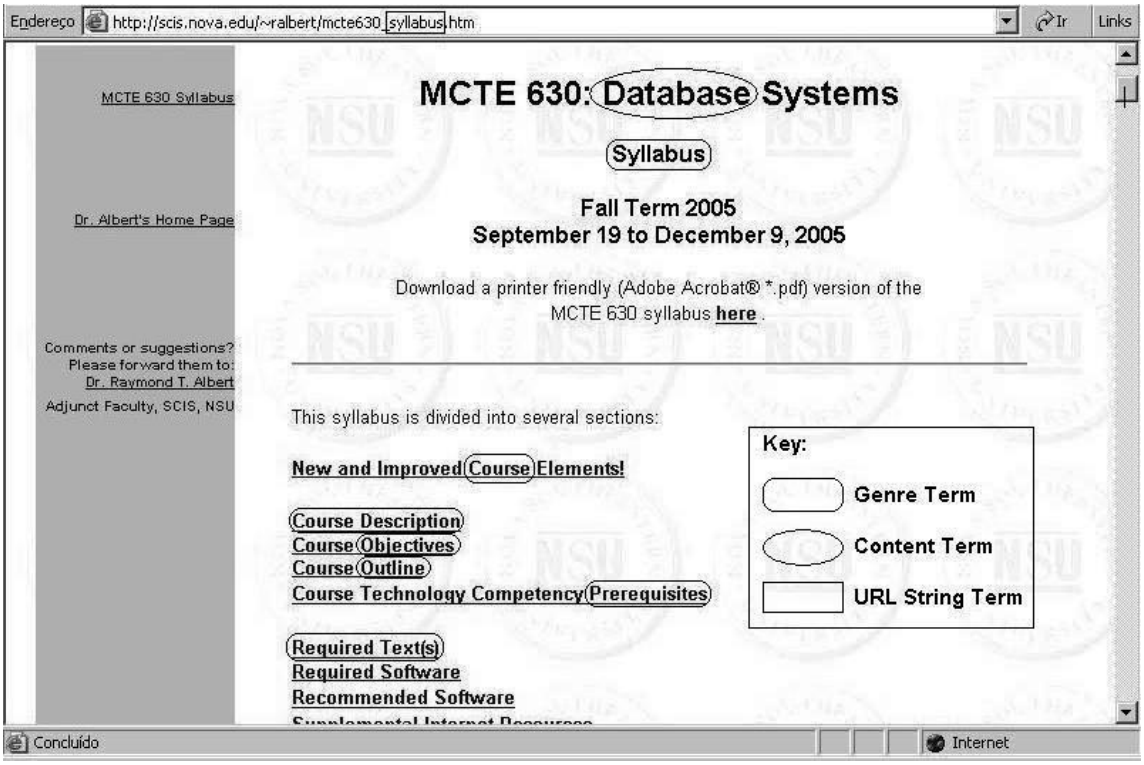

Figure 1.1: A typical course syllabus page

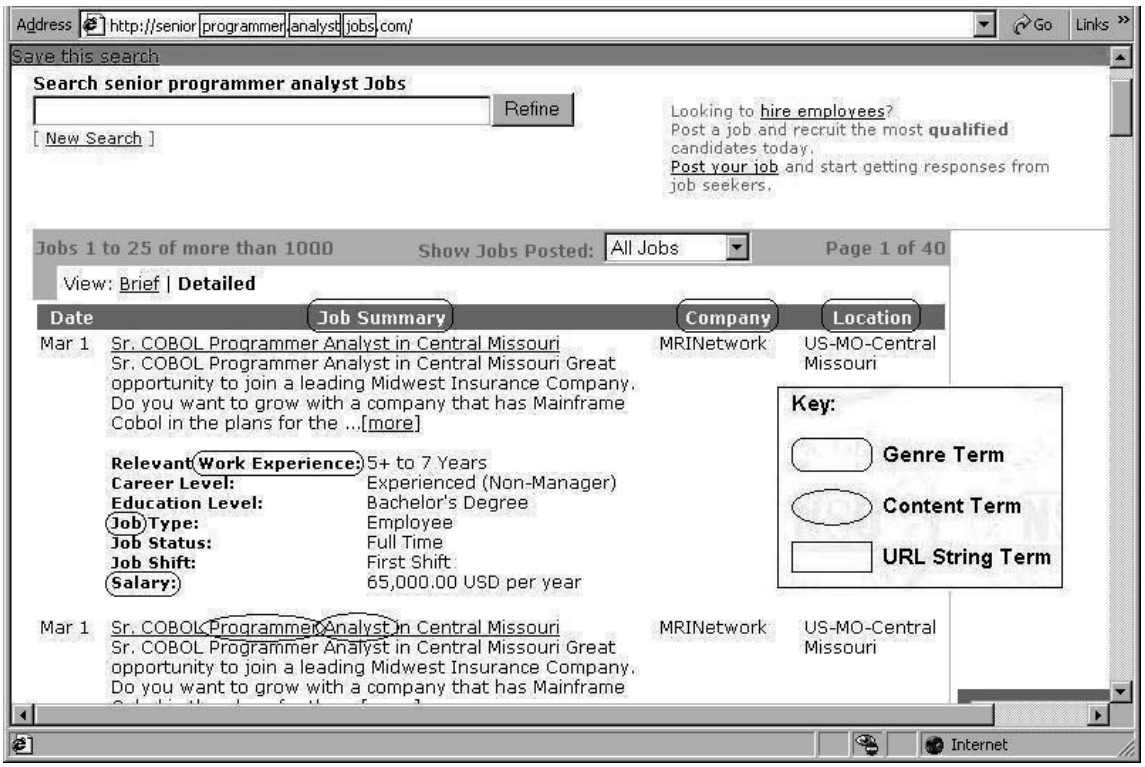

Figure 1.2: A typical computer science job offer page

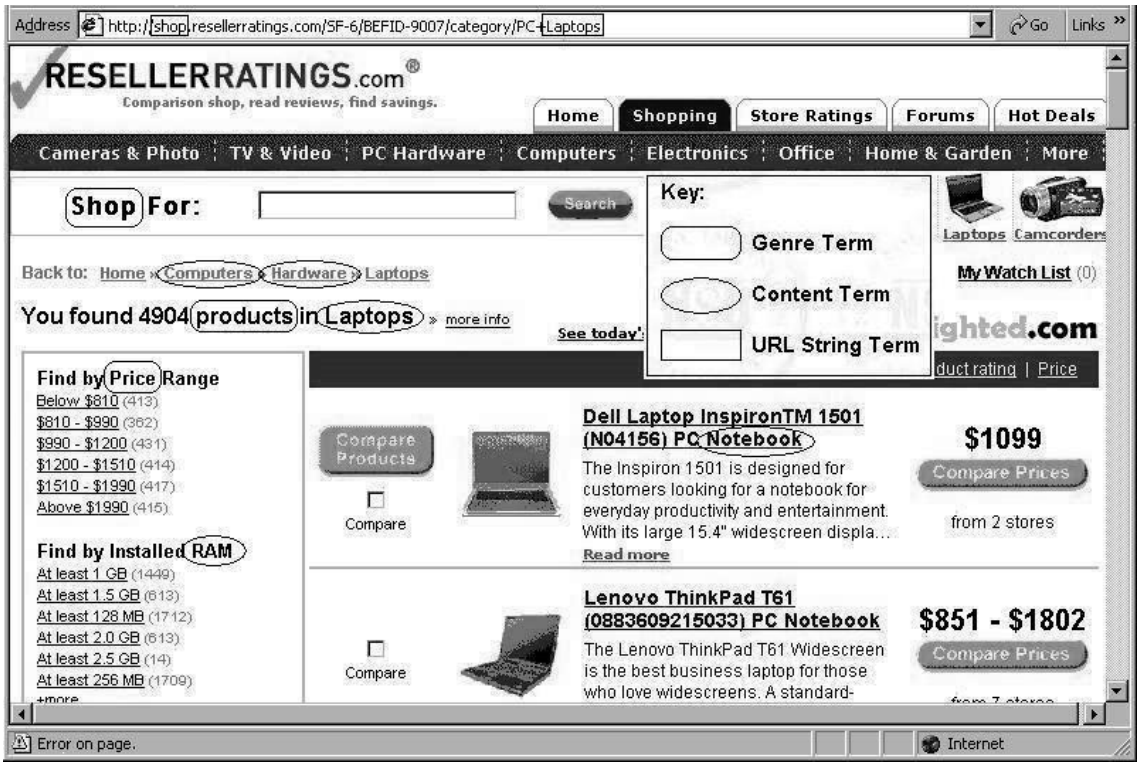

Figure 1.3: A typical computer equipment sale offer page

gain of 134% over traditional classifier-based focused crawlers.

In addition, our approach adopts a particular queuing policy, for determining in which order the pages must be fetched, that is suitable to situations in which a specific topic of interest can be represented in terms of genre and content information. As we have observed, in such situations the parent of a relevant page usually contains other URLs that also refer to relevant pages. This motivated us to develop a new queuing policy, which takes advantage of this feature to dynamically change the crawling priority of the non-visited pages, resulting in an efficient and scalable strategy for focused crawling. As shown by our experimental results, our focused crawler requires the analysis of no more than 60% of the visited pages in order to find 90% of the relevant pages, while a traditional classifier-based focused crawler has to analyze at least 90% of the visited pages to achieve the same performance.

A major issue in our genre-aware approach to focused crawling is the selection of terms required to describe the genre and the content (or subject) of the pages of interest, which usually requires the assistance of an expert on the application domain. Although not a complex task, the selection of these terms might have an impact on the effectiveness of a

crawler constructed according to our approach. Thus, we also analyze the impact of term selection in our approach. More specifically, we report on a set of experiments we have conducted to evaluate the impact of terms specified by an expert to describe the genre and content of the pages of interest on the crawling process. As we shall see, the results of these experiments show that focused crawlers constructed according to our genre-aware approach achieve good results even when a small number of relevant genre and content terms is used to describe the pages of interest. In addition, in order to facilitate even further the process of term selection, we describe and experimentally evaluate a simple strategy we have proposed for semi-automatic generation of these terms. This strategy, despite simple, has showed to be quite effective in suggesting suitable terms for the crawling process.

1.3 Contributions

As already mentioned, the main goal of this thesis is to propose a framework to allow the construction of effective, efficient and scalable focused crawlers that take into consideration both the genre and the content of the desired pages. Thus, its major contributions are:

1. A detailed description of a novel approach to focused crawling that exploits genre and content-related aspects of the desired pages and does not require a training or any preprocessing phase [2]. Moreover, this approach adopts a queuing policy that dynamically changes the crawling priority of the non-visited pages according to the content of the visited ones, therefore providing an efficient and more scalable crawling strategy.

2. An extensive set of experiments, involving three distinct application domains, that demonstrates the effectiveness, efficiency and scalability of our approach.

3. An experimental analysis of the impact of the number of terms in our approach, and the proposal and evaluation of a strategy for the semi-automatic selection of the terms required to express the genre and content-related aspects of the desired pages [3].

1.4 Related Work

Since the first proposals to focused crawlers, such as FishSearch [7], a great variety of methods has been developed for this purpose. According to Pant and Srinivasan [35], many crawling algorithms are variations of the Best-First crawler, in which a list of non-visited

URLs is kept as a priority queue. The non-visited URLs correspond to pages that have not yet been analyzed by the crawler. Each non-visited URL has a score associated to it, which reflects the benefits of following this URL, determining, therefore, its priority to be visited. Variations of the Best-First crawler can be created changing the heuristics used to give scores to a URL in the priority queue. The Naive Best-First crawler [30] measures the relevance of a page to a specific topic by calculating the cosine distance between the vector of terms that represent that page and the vector of terms that represent the topic of interest. Then, it uses this result to estimate the benefit of following the URLs found in that page. Naive Best-First crawlers and their variations have been described and analyzed in various works [29, 30]. FishSearch [7] uses a similar measure combined with the notion of depth limit, which prevents the exploration of paths leading to a sequence of irrelevant pages. This crawler searches more extensively those areas of the Web in which relevant pages have been previously found. At the same time, it discontinues searches in regions that do not yield relevant pages. SharkSearch [20] is a more aggressive version of FishSearch. It proposes a more sophisticated technique to classify the non-visited URLs, which is based on the anchor text, i.e., the text surrounding a link, and the inherited scores of ancestor pages, i.e., pages that appear in the crawling path for the URL. This technique uses a continuous valued function for measuring relevance as opposed to the binary relevance function in FishSearch. Like its predecessor, SharkSearch also keeps the notion of depth limit.

In [29], algorithmic aspects of focused crawlers are analyzed. For this purpose, a set of crawling algorithms considered as representative in the literature was implemented, including: Breadth-First, PageRank and SharkSearch. Based on the evaluation of such crawling strategies, a new class of crawling algorithms, named InfoSpiders, was designed and implemented, presenting an improved performance. These algorithms include adaptive mechanisms, that is, mechanisms that promote the change of the crawler's behavior according to the context found during the crawling. InfoSpiders correspond, thus, to a set of evolutionary algorithms that generate a population of adaptive crawlers. In order to evaluate all crawling algorithms mentioned, two performance metrics were defined: (1) recall level, which determines how well a crawler is able to crawl pages relevant to a specific topic, and (2) the average similarity between the topic description and the set of pages crawled. The aim of this comparative work was to obtain a more complete and trustable picture of the advantages and disadvantages of various crawling strategies found in the literature. It was concluded that a crawler that uses evolutionary algorithms reaches high scalability, due to the distribution of work through their concurrent agents, resulting in a better performance/cost ratio.

Another class of solutions for focused crawlers is based on Machine Learning techniques and makes extensive use of classifiers. In this context, a classifier has the role of deciding whether a URL found during the page traversal is to be followed or not. The work described in [9] was the first one to use a classifier, in this case a Naive Bayes classifier [41], in order to guide a focused crawler. The basic idea behind such a crawler is to classify the crawled pages within the categories of a given taxonomy. Examples of relevant pages are used to create a Bayesian classifier, which is capable of determining the probability $P(c|p)$ of a crawled page p to belong to a category c in the taxonomy. It was suggested that such a focused crawler can be generalized and used with a taxonomy where only two categories, relevant and irrelevant, are available. The work in [16] has proposed a variation of this crawler, naming it a context-focused crawler. Such a crawler uses a set of Naive Bayes classifiers that are trained to estimate the link distance between a crawled page and the relevant pages. In [44], the BINGO! system for focused crawling and its application to information portal generation and expert Web search are presented. The system uses a Support Vector Machine (SVM) classifier [14] in order to guide the crawling processes. The performed experiments show the potential of this focused crawling paradigm but also raises the difficulties for properly calibrating crawl setups to achieve good recall and high precision.

In [36], the effect of distinct link contexts (i.e., terms that appear in the text around a hyperlink within a Web page) on the performance of SVM classifier-guided focused crawlers is investigated. The results show that a crawler that exploits words in both the immediate vicinity of a hyperlink and the entire parent page performs significantly better than one that depends on just one of these pieces of evidence. Moreover, a crawler that uses the tag tree hierarchy of the Web pages provides a more effective coverage. The work described in [1] proposes a Latent Semantic Indexing (LSI) classifier [23] that combines link analysis with text content in order to retrieve and index domain-specific web documents. The implementation provides a different approach to focused crawling and aims to overcome the limitations imposed by the need to provide initial data for training, while maintaining a high recall/precision ratio. LSI is a concept-based automatic indexing method that models the semantics of the domain in order to suggest additional relevant keywords and to reveal the "hidden" concepts of a given corpus while eliminating high-order noise. The method captures the higher-order "latent" structure of word usage across the documents rather than just surface-level word choice, modelling the association between terms and documents based on how terms co-occur across documents [18].

The work described in [35] presents a systematic and comparative study involving experiments that explore many versions of the Naive Bayes, SVM and Neural Network classifying schemes [15, 47, 49]. The experiments were performed in a collection of over 100 topics, therefore allowing statistically valid conclusions to be drawn. A crawling environment was designed and developed, which allowed new classifiers to be flexibly added. The results show that Naive Bayes [41] is a weaker choice, when compared to SVM [14] and Neural Network [31], to control a focused crawler. SVM seems to be the best choice among the three, since it performs as well as the Neural Network, in addition to requiring a lower training cost. Among the various versions of the classifying schemes analyzed, the best were: Naive Bayes with kernel density, SVM with first degree kernel, and Neural Network with four hidden nodes and learning rate of 0.1. A second analysis of the experiments showed a divergence, between crawlers of similar performance, in terms of the Web sub-spaces covered by them. Another framework to evaluate different focused crawling strategies is described in [45].

Other distinct approaches to focused crawling, also based on Machine Learning techniques, are proposed in the literature. For instance, in [27], two probabilistic models, Hidden Markov Models and Conditional Random Fields, are used to model the link structure and the content of documents leading to target pages by learning from user's topic-specific browsing. A focused crawler constructed according to this approach is treated as a random surfer, over an underlying Markov chain of hidden states, defined by the distance from the targets, from which the actual topics of a Web page are generated. In [40], it is suggested a crawling algorithm based on reinforcement learning, a branch of machine learning that concerns itself with optimal sequential decision making, for learning a value function that maps hyperlinks to future discounted reward. In [11], Web personal crawlers based on best first search and genetic algorithm techniques are proposed. Such crawlers can dynamically take user's homepages and search for the most closely related homepages, based on the links and keywords found on the input pages.

In general, focused crawlers guided by classifiers [1, 9, 16, 22, 35, 36, 44, 45] present an additional cost of having to train the classifiers with positive and negative examples of pages to be crawled and, due to the generality of the situations in which they are applied, usually reach recall and precision levels between 40% and 70%. This scenario does not change much with other kinds of focused crawler [7, 20, 27, 29, 30, 34, 38]. Moreover, as we can see from the above discussion, previous work on focused crawling relies on a single concept space (i.e., the topic of the pages) for driving the crawling process. An exception

is the work described in [48] that presents a tool for generating structure-driven crawlers. This tool takes as input a sample page and an entry point of a Web site and generates a structure-driven crawler based on navigation patterns - sequences of patterns for the links that a crawler has to follow to reach pages structurally similar to the sample page. Content aspects, though, are not considered in this case.

Thus, in order to improve the effectiveness and the efficiency of the crawling process, we exploit another equally important concept space that has been neglected in the literature: the genre of the pages. Our genre-aware approach is, to the best of our knowledge, a first attempt towards this direction.

1.5 Thesis Organization

The rest of this thesis is organized as follows:

- Chapter 2 presents an overview of Web page crawling;
- Chapter 3 presents a detailed description of our genre-aware approach to focused crawling;
- Chapter 4 describes the set of experiments we conducted to evaluate our approach and discusses the results obtained;
- Chapter 5 analyzes the impact of term selection in our genre-aware approach to focused crawling;
- Chapter 6 concludes the thesis and gives some directions for future work.

Chapter 2

Web Page Crawling

Due to the great popularity and, especially, to the rapid growth of the Web, there has been an increasing demand for new techniques to help users effectively locate the information they need. The access to Web information is done, basically, through search engines, that exploit the graph structure of the Web to locate relevant pages to a specific query. To allow this, a typical search engine needs to fetch as many Web pages as possible in order to generate a collection of indexed pages. This is performed by specific programs called crawlers or spiders [12, 13, 32, 37, 43].

According to [13], crawlers are widely used today. Crawlers for the major search engines attempt to visit most text Web pages, in order to build content indexes. Other crawlers may also visit many pages, but may look only for certain types of information. Running a crawler is a challenging task. There are tricky performance and reliability issues and, more importantly, there are crucial social issues involved. Crawling is the most fragile task in search engines since it involves interacting with hundreds of thousands of Web servers and various name servers which are all beyond the control of the system. Then, externally, the crawler must avoid overloading Web servers. Internally, the crawler must deal with huge volumes of data. Unless it has unlimited computing resources and unlimited time, it must carefully decide which URLs to visit and in what order, i.e., it must define the scheduling strategy of URLs to be visited. The crawler must also decide how frequently to revisit pages it has already seen, in order to keep its client informed of changes on the Web. Moreover, the crawler must ensure the correct handling of exception conditions and errors of many kinds and a good coverage of the set of collected pages with respect to the set of existing pages.

While some crawlers exhaustively crawl pages on the Web, others incorporate "focus" in order to generate topic-specific collections. Moreover, some special crawlers are targeted to crawl pages in the hidden Web. Thus, this chapter provides an overview of the technology involved in the construction of Web crawlers and is organized as follows: Section 2.1 presents traditional crawlers, Section 2.2 discusses focused crawlers, and Section 2.3 addresses hidden web crawlers.

2.1 Traditional Crawlers

A traditional crawler crawls pages over the Web, starting from a seed page and following the links found there, visiting, therefore, other pages. Such a process is repeated with the new pages, from which new links are followed, until a sufficient number of pages has been covered, or a specific objective has been reached. Then, another program, the indexer, reads these pages and creates an index based on the words contained in each page. Each search engine uses a proprietary algorithm to create its index such that, ideally, only meaningful results are returned for each query formulated by a user.

Figure 2.1 illustrates, according to [37], the functioning model of a traditional crawler. The crawler maintains a queue of unvisited URLs called *Frontier*. The queue is initialized with seed URLs which may be provided by a user or another program. Each crawling loop involves picking the next URL to crawl from the *Frontier*, fetching the corresponding page from the Web, parsing the retrieved page to extract the URLs, and finally adding the unvisited URLs to the *Frontier*. The crawling process may be terminated when a certain number of pages have been crawled. If the crawler is ready to crawl another page and the *Frontier* is empty, the situation signals a dead-end for the crawler and it stops. Algorithm 1 describes the functioning model of the crawler presented in Figure 2.1. Thus, crawling can be viewed as a graph search problem. The Web is seen as a large graph with pages at its nodes and hyperlinks as its edges. A crawler starts from some few nodes (seeds) and then follows the edges to reach other nodes.

The operations *Enqueue*, *Empty* and *Dequeue* presented in Algorithm 1, according to [43], are usual for a FIFO queue F, with the following modifications: (1) the operation *Enqueue* only adds a new URL to the queue if it is not already there; (2) the operation *Dequeue* only marks the URL in front of the queue as "removed", instead of actually removing it, since it is necessary to know what pages were visited by the crawler in order to avoid revisiting

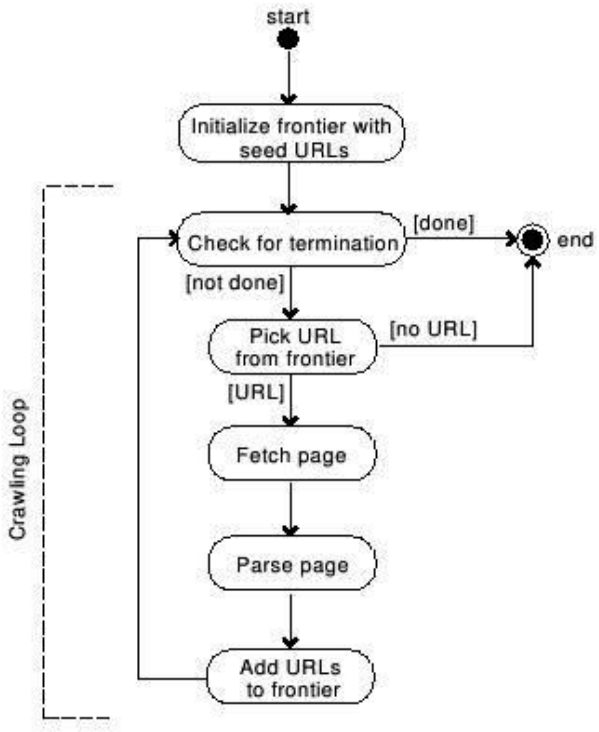

Figure 2.1: Functioning model of a traditional crawler [37]

them; and (3) the operation *Empty*, as a consequence, is true only when all URLs in the queue are marked as "removed". This scheduling strategy, based on a FIFO queue, is called *breadth-first*.

The *breadth-first* scheduling strategy produces a more comprehensive crawling, visits a larger number of sites and is more used since it is simple to implement. Another scheduling strategy, called *depth-first*, uses a stack structure to organize the unvisited URLs. Such a strategy results in a deeper crawling and visits a larger number of pages per site. However, these two strategies are not always the best option. For instance, Cho, Garcia-Molina and Page [13] propose a reordering of the unvisited URL queue such that the most important URLs would be visited before than the less important ones, when there are constraints on the time to crawl the pages and on the space available for storing local copies of them. In order to define the importance of a URL, they used five distinct metrics that can be combined:

- Similarity to a driving query: the importance of a URL is defined as the textual similarity between the corresponding page and the query.
- *Backlink count*: the importance of a URL is defined as the number of links to the

```
Algorithm:Traditional Crawler
begin
    Let I a list of seed URLs;
    Let F the Frontier;
    foreach URL i ∈ I do
        Enqueue(i,F);
    end
    while not Empty(F) do
        u ⟵ Dequeue(F);
        p ⟵ Get(u) {request page p pointed by u};
        Store p;
        Extract the hyperlinks from p;
        Let U the set of URLs cited in these hyperlinks;
        foreach URL u ∈ U do
            Enqueue(u,F);
        end
    end
end
```

Algorithm 1: Algorithm of the functioning model of a traditional crawler [43]

corresponding page that appear over the entire Web. Intuitively, a page that is linked to many pages is more important than one that is seldom referenced. On the Web, *backlink count* is useful for ranking query results, giving end-users pages that are more likely to be of general interest. Evaluating *backlink count* of a page requires counting backlinks over the entire Web. A crawler may estimate such a value with the number of links to the page that have been seen so far.

- *PageRank*: the importance of a URL is defined as the weighted sum of the backlinks to the corresponding page, since usually a page is more important than another, because of the relevance of references to it, and the *backlink count* metric treats all links equally. The *PageRank* metric has been found to be very useful for ranking the results of a user query.

- *Forward link count*: the importance of a URL is defined as the number of links that emanate from the corresponding page. Under this metric, a page with many outgoing links is very valuable, since it might be a Web directory. This kind of metric has been used in conjunction with other pieces of evidence to reasonably identify index pages.

- *Location metric*: the importance of a URL is a function of the location of the corresponding page, i.e., is a function of the URL itself. If URL u leads to page p, then the *location metric* is a function of u. For example, URLs ending with ".com" may be considered more useful than URLs with other endings, or URL containing the string "home" may be of more interest than other URLs. The *location metric* can be

evaluated simply by looking at the URL.

In general, the results obtained in [13] show that *PageRank* is an excellent ordering metric when either pages with many backlinks or with high *PageRank* are sought and that the scheduling strategy *breadth-first* is good. The work described in [32] extends the results of [13] regarding the effectiveness of crawling in *breadth-first* search order, using a much larger and more diverse data set. In order to measure the quality of downloaded pages over the life of the crawling process, it used the *PageRank* metric. Experimental results show that *breadth-first* search downloaded the relevant pages first, but also that the average quality of the pages decreased over the duration of the crawling process.

In general, according to [12], a crawler can update its index or local collection in two different ways. Traditionally, as already mentioned, the crawler visits the Web until the collection achieves a desirable number of pages, when it stops. Then, when it is necessary to refresh the collection, the crawler builds a brand new collection and then replaces the old collection with this new one. This type of crawler is called a periodic crawler. Alternatively, the crawler may keep visiting pages after the collection reaches its target size, to incrementally refresh the local collection. By this incremental update, the crawler refreshes existing pages and replaces "less-important" pages with new and "more-important" pages. This type of crawler is called an incremental crawler. In principle, the incremental crawler can be more effective than the periodic one. For instance, if the crawler can estimate how often pages change, the incremental crawler may revisit only the pages that have changed (with high probability), instead of refreshing the entire collection. But, clearly, the effectiveness of crawling techniques heavily depends on how Web pages change over time. If most Web pages change at similar frequencies, the periodic and the incremental crawlers may be equally effective, because both crawlers in fact revisit all pages at the same frequencies.

Due the fact that a crawler is a quite sensitive component of any commercial general purpose search engine, very few studies in the literature describe real case implementations. Some examples are *CoBWeb* [43], Mercator [21], and the Internet Archive crawler [8].

2.2 Focused Crawlers

According to [1], information retrieval systems that require high precision and targeted information must seek new unvisited pages in a more intelligent way. The crawler of a vertical search engine is assigned the task to automatically classify crawled Web pages to

existing categories and simultaneously have the ability to further discover Web information related to the specified topic. Due to its size and dynamic nature, the Web can be seen as an open and uncontrolled collection of hyperlinked documents. Consequently, the optimum path to relevant documents is always unknown. Thus, an efficient crawler has to guess promising links by taking decisions that are based on predefined rules or metrics, influencing its performance differently. In this context, a focused crawler [9, 27, 29, 30, 35, 45] analyzes its crawl boundary to find the links that are likely to be most relevant for the crawl while avoiding irrelevant regions of the Web. Focused crawling is a relatively new and promising approach to improving the recall of specific searches on the Web. It involves the automatic classification of visited pages into a user or community-specific topic hierarchy. They are becoming important tools to support applications such as specialized Web portals [28], digital libraries [38], online searching [45] and competitive intelligence [34].

Figure 2.2 details, according to [35], the high-level design of the infrastructure for focused crawlers. In a crawling loop, a URL is picked from the *Frontier* (e.g., a list of unvisited URLs), the corresponding page is fetched from the Web, the fetched page is processed through all the infrastructure layers and the unvisited URLs from the page are added to the *Frontier*. The networking layer is responsible for fetching and storing the pages and making this process transparent to the layers above it. The parsing and extraction layer parses the page and extracts the needed data such as hyperlink URLs and the page's context (e.g., words, phrases, etc). Then, the extracted data is represented in a formal notation by the representation layer before being passed to the intelligence layer, which associates priorities or scores with unvisited URLs from the page.

There are many techniques that can be used by the intelligence layer to associate scores with unvisited URLs in the priority queue. A key issue here is how to measure the effectiveness of these techniques since, for this, it is necessary to evaluate the focused crawler itself, using the results obtained in the crawling processes. However, according to [35], the evaluation of focused crawlers is a challenging problem due to the unavailability of a complete relevance set for any topic. Also, manual relevance judgement for each of the crawled pages is extremely costly in terms of man-hours when we have millions of crawled pages spread over hundreds of topics.

The literature shows a wide variety of evaluation metrics. Some examples are:

- percentage of relevant pages retrieved over the progress of the crawl [13];

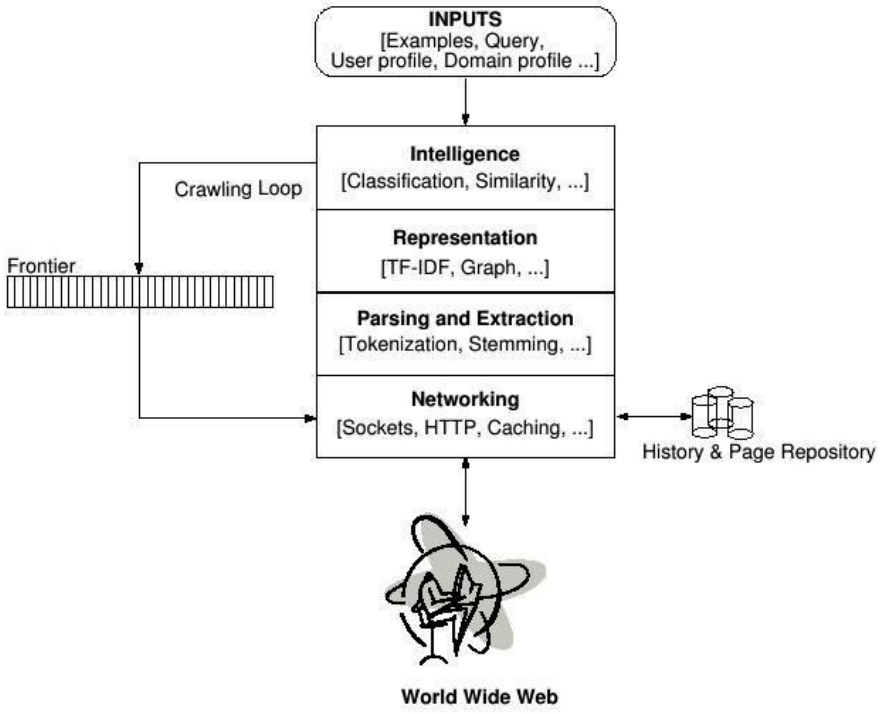

Figure 2.2: High-level design of the infrastructure for focused crawlers [35]

- search length, i.e., number of pages crawled until some predetermined fraction of the relevant pages has been visited [29, 30];
- average similarity between the topic description and the set of pages crawled [29];
- harvest rate estimating the precision measure, i.e., the running average over different time slices of the crawl [9, 35];
- target recall estimating the recall measure, i.e., the fraction of relevant pages that are downloaded by the crawler [34, 45];
- average in day on which the top n pages are retrieved, where n is a variable [32];
- average relevance of pages computed using a sliding window of 200 downloads [16];
- percentage of relevant pages collected [40];
- average rank of the retrieved pages over the progress of the crawl [45].

As already discussed in Section 1.4, several approaches to focused crawling have been proposed in the literature. Some approaches are based on heuristics specifically developed

to determine the relevance between a Web page and the desired topic. Others are based on Machine Learning techniques, making extensive use of classifiers.

In general, heuristic-based focused crawlers, such as *Best-First* [29], *FishSearch* [7], *Shark-Search* [20] and *InfoSpiders* [29], use three kinds of contextual information in the intelligence layer of Figure 2.2, in order to generate the score of a unvisited URL and estimate the benefit of following it:

- Link context: corresponds to the lexical content, which is displayed around the URL in the parent page, i.e., the page from which the URL was extracted. Most focused crawlers use such contextual information.
- Ancestor pages: determine the lexical content of the pages that lead to the current page that is associated with the URL in question. Few focused crawlers, as suggested in [16] and [22], use information taken from the ancestor pages, since the content of these pages provides little help when compared with the content of the actual parent page.
- Web graph: describes the structure of the Web sub-graph around the node (page) associated with the URL being processed. Such contextual information can provide key evidence to decide whether to follow the URL. For instance, if the parent page in which the URL appears is a good hub, that is, a page that contains links to many resourceful pages, then the priority to follow that URL should increase.

Distinctly, classifier-based focused crawlers use a classifier in the intelligence layer of Figure 2.2, in order to generate the score of a unvisited URL in the priority queue. Thus, a classifier has the role of deciding whether a URL found during the page traversal is to be followed or not. A classifier can model the topic of interest using positive and negative examples (a training set). The positive examples are Web pages that are considered relevant to the topic while the negative examples are Web pages that are considered irrelevant. A classifier learns from these examples through a training process.

Figure 2.3 illustrates, according to [35], the functioning model of a traditional focused crawler guided by a classifier. As we can see, firstly, a priority queue is initialized with the URLs of the seed pages. Then, a non-visited URL is selected from the queue based on a score assigned by the classifier, and the corresponding page is fetched from the Web and abstractly represented in terms of its content (e.g., by a feature vector). At this moment,

an abstract representation of the page is analyzed by a classifier and a score that measures the relevance of the page to the desired topic is given. All URLs of the analyzed page are placed in the priority queue with the same score of this page. This process is repeated until there are no more URLs in the queue. Notice that, previously to the crawling process, the classifier must be trained with examples of pages to be crawled.

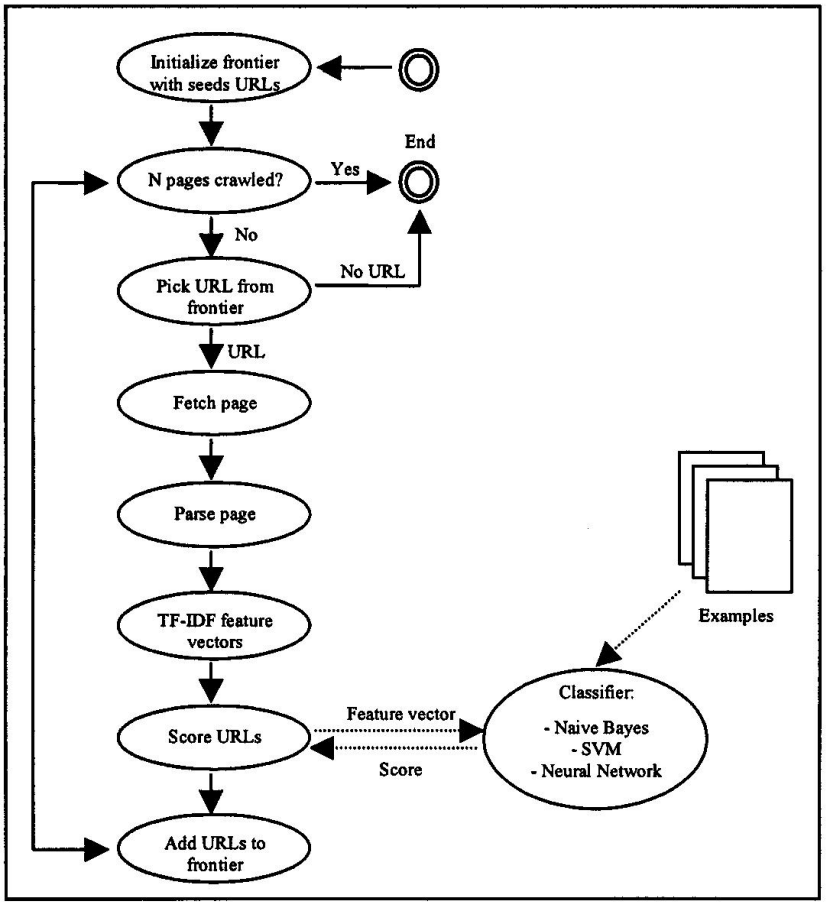

Figure 2.3: Functioning model of a focused crawler guided by a classifier [35]

2.3 Hidden Web Crawlers

According to [39], traditionally, crawlers have only targeted a portion of the Web called the Publicly Indexable Web (PIW), which refers to the set of pages reachable purely by following hypertext links, ignoring search forms and pages that require authorization or prior registration. However, a significant fraction of Web content lies outside the PIW. Specifically, large portions of the Web, called the hidden Web [17], are hidden behind search

forms. Pages in the hidden Web are dynamically generated in response to queries submitted via search forms. The hidden Web continues to grow, as organizations with large amounts of high-quality information are placing their content online, providing Web search facilities over existing databases.

There are significant technical challenges in designing a hidden Web crawler. First, the crawler must be designed to automatically parse, process and interact with search form interfaces, which are designed primarily for human use. Second, hidden Web crawlers must also provide input in the search form. This raises the issue of how best to equip crawlers with the necessary input values for use in search queries.

The work described in [24] presents a method to generate hidden Web crawlers for sites with common navigational characteristics. The method uses a set of heuristics and a sample data repository for automatically finding relevant forms, filling them in and crawling the resulting pages. Since developing a general solution to automatically generate crawlers for hidden Web pages is a very difficult task, because a dynamic page can be generated in many different ways, the method in [24] restricts its scope to sites respecting two navigation patterns that reflect common ways for users to navigate on the Web. A navigation pattern is represented as a directed graph. Figure 2.4 illustrates the two navigation patterns used in this work. In the first navigation pattern, the user starts from the site's main page, fills in a search form, and then submits it, getting a set of answer pages. The second navigation pattern adds an intermediate page containing a set of links designed to refine on the search, and from this page the user can get to the answer pages. Experimental results have shown that the proposed method is quite effective, achieving high levels of precision and recall, and being able to automatically generate a successfull crawler for 80% of the sites used in the experiments.

In [39], the authors adopt a task-specific and user-assisted approach. According to this approach, the implemented crawler selectively crawls portions of the hidden Web, extracting content based on the requirements of a particular task, and the user-assistance ensures that the crawler issues queries that are relevant to this particular task. Figure 2.5 presents the operational model of such an approach to hidden Web crawling. This operational model supports designing a variety of hidden Web crawlers, using simple and sophisticated natural language and knowledge representation techniques.

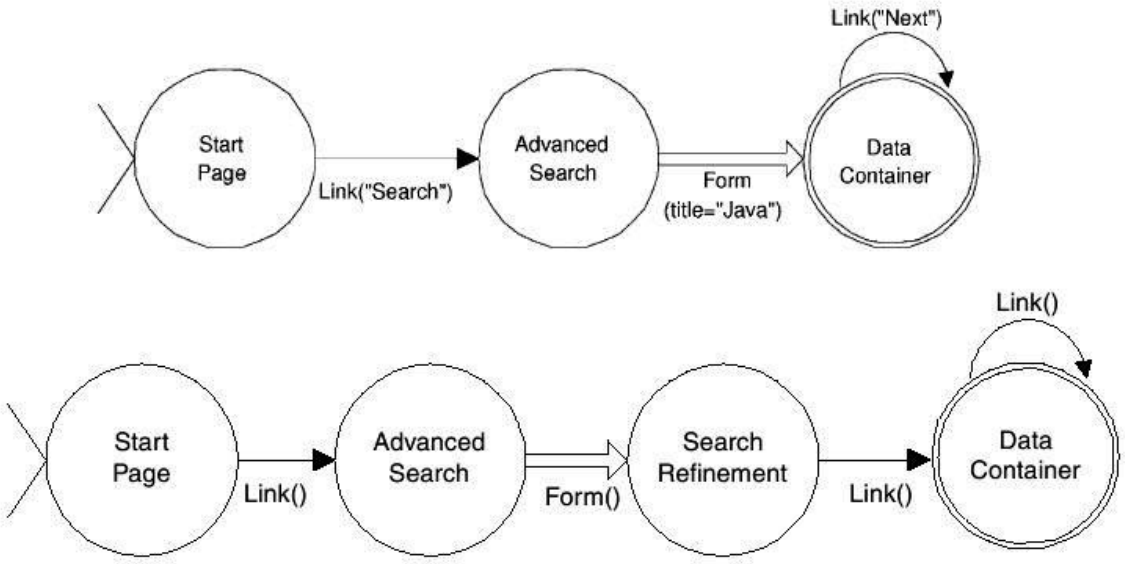

Figure 2.4: Navigation patterns according to [24]

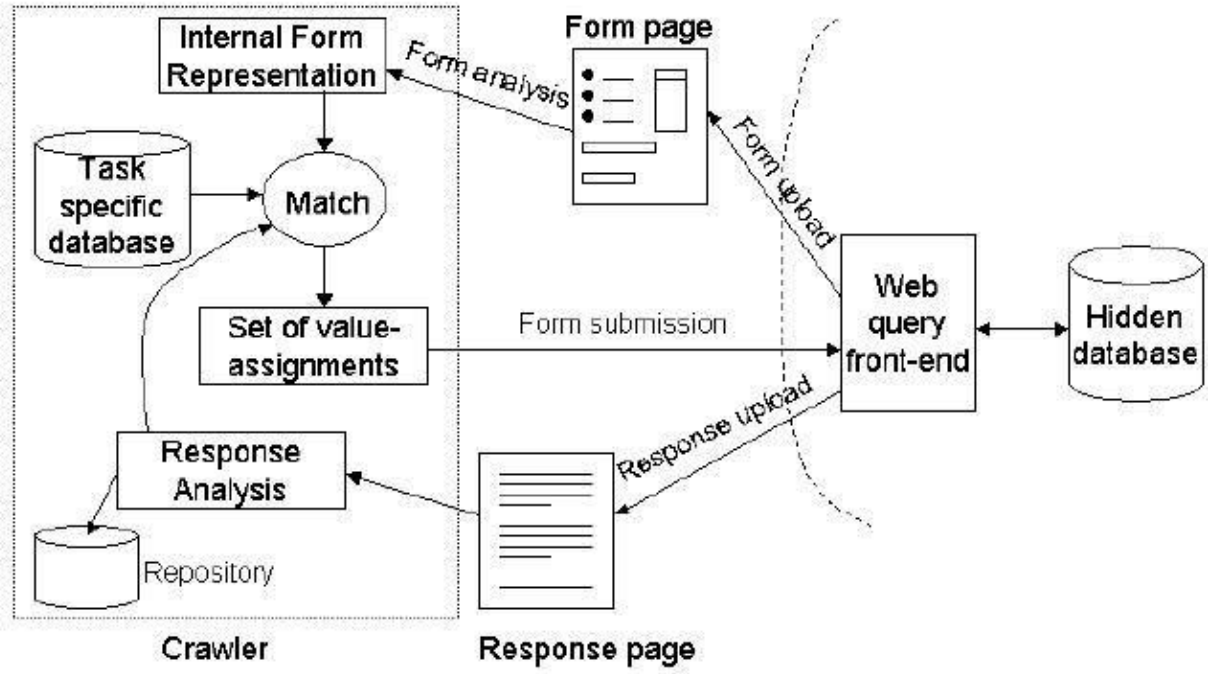

Figure 2.5: The operational model of a task-specific and user-assisted approach to hidden Web crawling [39]

In Figure 2.5, the terms "form page" and "response page" are used to denote the page containing a search form and the page received in response to a form submission, respectively. The operational model consists of the following four components:

- Internal Form Representation. In this component, when the crawler receives a form page, an internal representation of the search form is built.
- Task-specific database. A crawler is equipped, at least conceptually, with a task-specific database, which contains the necessary informations for the crawler to formulate search queries relevant to the particular task.
- Matching function. The crawler's matching algorithm takes, as input, an internal form representation and the current contents of the database, producing, as output, a set of value assignments. The crawler uses each value assignment to fill-out and submit the completed form.
- Response Analysis. This component receives the response to a form submission and stores the page in the crawler's repository. In addition, this component attempts to distinguish between pages containing search results and pages containing error messages. This feedback can be used to tune the matching function and update the set of value assignments.

Considering that an effective hidden Web crawler can have a tremendous impact on how users search information on the Web, the work described in [33] provides a theoretical framework to investigate the hidden Web crawling problem and propose effective ways of generating queries automatically. Three different query generation policies for the hidden Web are proposed: a policy that picks queries at random from a list of keywords, a policy that picks queries based on keywords frequency in a generic text collection, and a policy that adaptively picks a good query based on the content of the pages downloaded from the hidden Web site. Experimental evaluation shows that these policies have great potential. In particular, in certain cases, the adaptive policy can download more than 90% of a hidden Web site after issuing approximately 100 queries.

Chapter 3

Genre-Aware Focused Crawling

The genre-aware approach to focused crawling proposed in this work relies on the fact that some specific topics of interest can be represented by considering two distinct aspects: its genre (text style) and content-related information. For instance, if a user wants to crawl Web pages that include syllabi of databases courses, a focused crawler should analyze a specific Web page considering, separately, the terms present in that page that indicate that it corresponds to a syllabus (terms that characterize the course syllabus genre) and the terms that indicate that the page is related to the field of database (terms that appear in a syllabus content of a databases course). On the other hand, a traditional focused crawler guided by a classifier analyzes the content of a specific Web page, but does not consider separately the two kinds of information. Therefore, pages that include syllabi of other courses, as well any page referring to topics in the field of database, could be selected by this crawler as being related to a databases course (precision errors), whereas pages with genre and content poorly specified but including a syllabus of a databases-related course would be classified as belonging to the "other" category (recall errors).

Thus, our approach to focused crawling considers a set of heuristics to guide a crawler, in such way that it allows the separate analysis of the genre and the content of a Web page. In addition, it also considers the page URL string, since it may include terms related to both the genre and the content of the desired topic of interest. Our set of heuristics has been designed with two main objectives: improving the level of F1 of the crawling process and speeding up the crawling of relevant pages.

This chapter describes our approach to focused crawling. First, Section 3.1 discusses how our approach determines the relevance between a Web page and a set of specified terms. Then,

Section 3.2 details our approach by describing the algorithm that implements the crawling process.

3.1 Relevance Assessment

In order to determine the relevance between a Web page and the set of terms that represent the genre and the content of the desired topic, we use the cosine similarity function derived from the classic vector space model [4] widely used in the information retrieval area. The choice of this model was twofold: (1) it is easy to implement and usually provides an effective solution and (2) its cosine ranking formula allows sorting the crawled pages according to their degree of similarity to the desired topic, which is important to establish a crawling priority. Thus, consider a Web page p_j and a user specification on the genre or the content of the desired topic given by a set of terms s. The relevance of p_j with respect to s, determined by the degree of similarity between p_j and s, is given by

$$sim(p_j, s) = \frac{\sum_{i=1}^{t} w_{i,j} \times w_{i,s}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2} \times \sqrt{\sum_{i=1}^{t} w_{i,s}^2}} \tag{3.1}$$

where k_i is a term in set s, t is the number of terms in set s, $w_{i,j}$ is the weight associated with the pair (k_i,p_j) and $w_{i,s}$ is the weight associated with the pair (k_i,s). If $sim(p_j, s)$ is greater or equal to a similarity threshold previously specified, the Web page p_j is considered relevant.

The weight $w_{i,j}$ is given by

$$w_{i,j} = f_{i,j} \times idf_i \tag{3.2}$$

where $f_{i,j}$ is the normalized frequency of the term k_i in the page p_j and idf_i is the Inverse Document Frequency (IDF) of the term k_i that indicates the importance of the term k_i inside a collection.

The normalized frequency $f_{i,j}$ is given by

$$f_{i,j} = \frac{freq_{i,j}}{max_l(freq_{l,j})} \tag{3.3}$$

where $freq_{i,j}$ is the raw frequency of the term k_i in the page p_j (i.e., the number of times that the term k_i is mentioned in the text of the page p_j) and $max_l(freq_{l,j})$ is the biggest frequency among the frequencies of the terms k_i mentioned in the text of the page p_j.

The IDF value idf_i is given by

$$idf_i = log\frac{N}{n_i} \tag{3.4}$$

where N is the total amount of pages in a collection and n_i is the amount of pages of such collection in which the term k_i appears.

Finally, the weight $w_{i,s}$ is given by

$$w_{i,s} = (0.5 + \frac{0.5 \times freq_{i,s}}{max_l(freq_{l,s})}) \times idf_i \tag{3.5}$$

where $freq_{i,s}$ is the raw frequency of the term k_i in the set of terms s and $max_l(freq_{l,s})$ is the biggest frequency among the frequencies of the terms k_i in the set s. Since each term k_i appears only once in the set s, we can conclude that $w_{i,s} = idf_i$ for all terms k_i.

3.2 Crawling Process

To crawl Web pages according to our approach, we use the procedure FCrawl described by Algorithm 2. This procedure takes as input sets of terms that represent the genre (*GenreTerms*) and the desired information content (*ContentTerms*), and outputs a set of relevant pages (*RelevantPages*). Eventually, another set of terms, representing the URL string (*URLTerms*) of a page related to this topic, may also be specified. Each URL string term is related to the page genre or to the desired content. These sets of terms can be specified by an application domain expert or, as discussed in Section 5.2, they can be generated in a semi-automatic fashion.

This procedure also takes as input a set of URLs pointing to seed pages (*SeedPages*) and two additional parameters: (1) the similarity threshold[1] (*SimilarityThreshold*), that indicates whether a specific visited page is relevant or not, and (2) the change threshold (*Change-Threshold*), that indicates whether the score of a URL may be changed during the crawling process. A list of non-visited URLs is kept in a priority queue called *Frontier*. In addition, this procedure invokes the following pre-defined procedures and functions:

[1]It should be noticed that despite our crawling procedure having been implemented with the similarity threshold, a practical implementation for real users could consider a maximum number of pages to be visited as a parameter to be set by such users, for example, based on time constraints. The results could then be ranked by similarity. The use of the threshold allows us though to better evaluate the potential of our chosen strategies.

- InsertURL(*URL*,*score*,*URLtype*): inserts in *Frontier* a *URL* with a specific *score* and *URLtype* ("seed" or "non-seed");
- RemoveURL(*URL*,*URLtype*): removes from *Frontier* the URL with the highest score, returning such *URL* and its *URLtype*;
- FetchParsePage(*URL*,*page*,*terms*,*links*): fetches and parses the page pointed by *URL*, returning the actual *page* and its sets of *terms* and *links*;
- CosineDistance(*page*,*terms*): returns the cosine distance between a *page* and a set of *terms*;
- Mean(*similarity1*,*similarity2*,*weight1*,*weight2*): returns the weighted mean between two similarity values according to weights assigned to them;
- ChangeScores(*URL*,*newscore*): changes to *newscore* the score of the URLs in *Frontier* that correspond to sibling pages of a given *URL*.

Procedure FCrawl works as follows. Step 01 initializes *Frontier* with the URLs of the seed pages, setting the URL scores to 1. For each URL in *Frontier* (step 02), the corresponding page is visited (step 04) and its content analyzed (steps 05 to 09). This is done by determining the degree of similarity between the current page and the sets of terms that describe the desired topic of interest. Thus, the cosine similarity function is applied separately to each set of terms (steps 05, 06 and 08), generating a specific similarity score between the current page and the sets of terms that represent, respectively, the genre, the content and the URL string of the desired pages. Then, these scores are combined into a final single one (steps 07 and 09) and compared with a given threshold. If this final score is greater than or equal to this threshold, the visited page is included in the set of relevant pages (step 10). As we shall see, determining the relevance of a page to a specific topic by considering separately pieces of evidence related to its genre and content is the main reason for the F1 levels we have achieved with our genre-aware approach. Next, if the current page is a non-seed one, the final similarity score is compared with a second threshold to determine whether the scores of URLs in *Frontier* that correspond to children of the current page's parent may be changed (step 11). Notice that, for doing so, procedure ChangeScores should be able to locate the URLs of the siblings of the current page. Finally, the links previously extracted from the current page are inserted into *Frontier* (step 12) having their scores set to 0.

As we demonstrate in our experiments, the strategy of dynamically changing the crawling priority of the non-visited pages is the main reason for the good performance of our crawling

```
    Algorithm:Procedure FCrawl
    Input: GenreTerms, ContentTerms, URLTerms, SimilarityThreshold, ChangeThreshold, SeedPages;
    Output: RelevantPages;
    begin
        Let GW, CW, GCW and UW be weights of importance for the genre, the content, the genre-content
        combination and the URL string respectively;
 1      foreach pageURL ∈ SeedPages do
            InsertURL(pageURL,1,"seed");
        end
 2      while there are non-visited URLs in Frontier do
 3          RemoveURL(CurrentURL,URLType);
 4          FetchParsePage(CurrentURL,CurrentPage,PageTerms,PageLinks);
 5          GenreSimilarity ⟵ CosineDistance(PageTerms,GenreTerms);
 6          ContentSimilarity ⟵ CosineDistance(PageTerms,ContentTerms);
 7          CombinedSimilarity ⟵ Mean(GenreSimilarity,ContentSimilarity,GW,CW);
 8          URLSimilarity ⟵ CosineDistance(PageTerms,URLTerms);
 9          FinalSimilarity ⟵ Mean(CombinedSimilarity,URLSimilarity,GCW,UW);
10          if FinalSimilarity ≥ SimilarityThreshold then
                RelevantPages ⟵ RelevantPages + CurrentPage;
            end
11          if URLType = "non-seed" and FinalSimilarity ≥ ChangeThreshold then
                ChangeScores(CurrentURL,FinalSimilarity);
            end
12          foreach link ∈ PageLinks do
                InsertURL(link,0,"non-seed");
            end
        end
    end
```

Algorithm 2: The procedure FCrawl

approach since it allows for relevant pages to be crawled as soon as possible. This occurs because in situations where a specific topic can be separately represented in terms of genre and content, we observed that the parent of a relevant page often contains other URLs that also point to relevant pages. On the other hand, a relevant page does not usually include a link to another relevant page. For example, considering that a crawler has located a page that includes the syllabus of a databases course, URLs of other relevant pages can be found in the page of the graduate program which that course is part of or in the page that includes the list of courses taught by a certain professor, whereas it is unlikely to find URLs of relevant pages in the actual page that includes the course syllabus. This strategy, which speeds up the crawling process, performs well in domains where the parent page has a list of items related to the topic of interest. In fact, this is a very common navigation pattern found on the Web [24].

Figure 3.1 summarizes the main steps of our focused crawling approach. Notice that a focused crawler constructed according to our genre-aware approach follows the same basic

steps of a traditional focused crawler guided by a classifier (see Figure 2.3). However, the fact that our approach exploits separately genre and content information, as well as adopts a dynamic queuing policy, considerably improves our results as we shall see next.

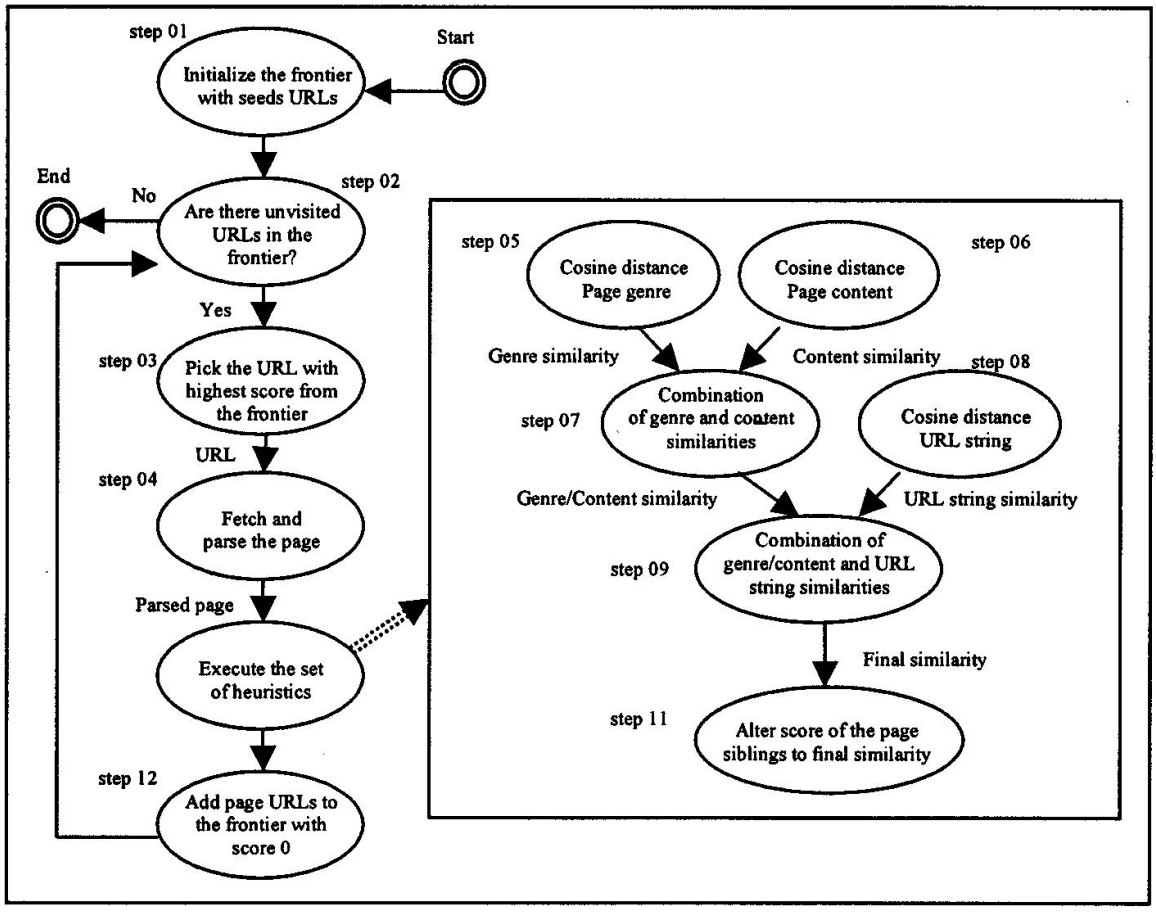

Figure 3.1: Functioning model of our focused crawling approach

Chapter 4

Experimental Evaluation

In this chapter, we describe the experiments we have designed and conducted to demonstrate the effectiveness, efficiency and scalability of our genre-aware approach to focused crawling, and discuss the results obtained.

4.1 Experimental Design and Setup

In order to conduct our experiments, we implemented a crawler based on the procedure FCrawl described by Algorithm 2. In our experiments, this crawler was run in the context of the Brazilian Web with the purpose of crawling pages related to three topics: (1) syllabi of specific computer science courses, (2) job offers in the computer science field, and (3) sale offers of computer equipments. These three topics are related to the same general subject ("computer science") but their pages of interest are of different genres ("course syllabus", "job offer" and "sale offer"). Notice that the fact that the topics of these experiments were related to computer science is not significant and other topics could also have been chosen. This choice was due to our familiarity with the topic which would help evaluate the experiment results.

For the first topic, three disciplines with distinct characteristics were chosen as specific subjects: "databases", which is a discipline whose content is well-consolidated, "data structures", which is also a well-consolidated discipline but whose content may be dispersed into several courses, and "information retrieval", which is a discipline whose content is not yet well-consolidated due to its fast recent evolution. In addition, five seed pages of Brazilian higher education institutions were used. For the other two topics, job offers in computer

science and sale offers of computer equipments, we used ten seed pages. The reason for this was to capture the trend we have found on the Web, where there are more pages for the two last topics than for the first one. More importantly, the same number of seed pages was given to both, our crawler and the baseline, allowing therefore a fair comparison. For each topic, an expert manually specified the set of terms (or phrases[1]) required to represent the genre and the content of the desired pages as well as the answer set containing the relevant pages among those visited by the crawler in order to allow us to assess the effectiveness of our approach. Table 4.1 shows the number of genre and content terms specified for each topic and Table 4.2 shows these terms. We notice that we used for the first topic ("course syllabi") exactly the same genre terms for all three disciplines. However, some subjects may require more terms than others due to their inherently broadness and complexity.

Table 4.1: Number of terms specified for each topic

Topic	Number of genre terms	Number of content terms
Syllabi (genre) of data structures courses (subject)	16	16
Syllabi (genre) of databases courses (subject)	16	24
Syllabi (genre) of information retrieval courses (subject)	16	24
Job offers (genre) in computer science (subject)	24	24
Sale offers (genre) of computer equipments (subject)	24	48

It is also important to notice that the specification of the sets of terms is a simple task when compared with the effort required to train a classifier. For instance, as already mentioned, the genre and content terms that characterize the pages presented in Figures 1.1, 1.2 and 1.3 are easily recognized.

In order to evaluate the effectiveness of our genre-aware approach, we measured the F1 level for each crawling process executed. According to Baeza-Yates and Ribeiro-Neto [4], the F1 measure is the harmonic mean of the two most widely used information retrieval performance measures, precision and recall, and assumes values in the interval [0,1]. In our context, precision corresponds to the fraction of the retrieved pages that is relevant, i.e,

$$Precision = \frac{|number\ of\ relevant\ pages\ retrieved|}{|number\ of\ pages\ retrieved|} \tag{4.1}$$

[1]Multi-word terms.

Table 4.2: Examples of terms used for each topic

Topic	Genre terms	Content terms
Data structures course syllabi	"syllabus", "course", "credits", "instructor", "prerequisites", "class hours", "objectives", "description", "outline", "topics", "tentative schedule", "grading", "exams", "materials", "required text", "textbooks"	"data structures", "algorithms and data structures", "abstract data type", "linear lists", "linked lists", "stacks", "queues", "trees", "pointers", "abstraction", "sorting", "searching", "hashing", "algorithm analysis"
Databases course syllabi	"syllabus", "course", "credits", "instructor", "prerequisites", "class hours", "objectives", "description", "outline", "topics", "tentative schedule", "grading", "exams", "materials", "required text", "textbooks"	"database", "dbms", "database management", "data model", "conceptual model", "relational model", "entity-relationship", "referential integrity", "functional dependencies", "normalization", "relational algebra", "relational calculus", "sql", "queries", "concurrency control", "query optimization", "data definition", "data manipulation", "triggers", "data warehousing", "distributed databases"
Information retrieval course syllabi	"syllabus", "course", "credits", "instructor", "prerequisites", "class hours", "objectives", "description", "outline", "topics", "tentative schedule", "grading", "exams", "materials", "required text", "textbooks"	"information retrieval", "index construction", "classic models", "boolean model", "vector model", "probabilistic model", "vector", "ranking", "similarity", "relevance", "precision", "recall", "relevance feedback", "query expansion", "search engines", "inverted files", "categorization", "crawling", "stemming", 'stopwords", "clustering"
Job offers in computer science	"jobs", "job offer", "vacancies", "company", "organization", "contact", "location", "city", "details", "description", "salary", "benefits", "career", "job type", "category", "work experience"", "requirements", "curriculum", "education", "responsibilities", "competencies", "qualifications", "restrictions", "work schedule"	"computer", "computer science", "computer technicians", "programming", "system analysis", "programmer", "developer", "system analysis", "security analyst", "support analyst", "administrator", "computer engineer", "software", "database", "computer network", "programming languages", "operational system", "mainframe", "technical support", "help desk", "internet, "webdesigner", "webmaster", "computer maintenance"
Sale offers of computer equipments	"sale", "buy", "shop", "virtual shop", "product", "products list", "brand", "used", "price", "offers", "promotion", "discount", "payment", "parcels", "availability", "location", "headline", "image", "contact", "add to", "classification", "sections", "categories", "groups"	"technology", "accessories", "components", "pc", "computer", "hardware", "software", "bits", "gb", "mb", "cartridge", "cd", "cdrom", "drive", "fax modem", "cpu", "hd", "printers", "laserjet", "toner", "motherboards", "memory", "monitor", "lcd", "widescreen", "screen", "keyboard", "mouse", "mp3 player", "nobreak", "notebook", "laptop", "desktop", "palm", "pen drive", "cables", "video cables", "processor", "core", "scanner", "speaker", "multimedia", "joysticks", "subwoofer", "webcam", "cam", "wireless", "batteries"

and recall corresponds to the fraction of the relevant pages that has been actually retrieved, i.e.,

$$Recall = \frac{|number\ of\ relevant\ pages\ retrieved|}{|number\ of\ relevant\ pages|} \tag{4.2}$$

The F1 measure is then computed as

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{4.3}$$

Thus, F1 assumes a high level only when both precision and recall are high. Therefore, determining the maximum level for F1 can be interpreted as an attempt to find the best possible compromise between recall and precision.

4.1.1 Pre-crawling: The IDF Factor

As we can see from Equation 3.2, in conventional information retrieval applications, in order to apply the cosine measure to evaluate a query against to a specific collection of documents, it is necessary to compute first, for each query term, an IDF value that indicates the importance of that term inside the collection. Thus, to be able to implement the function CosineDistance used by the procedure FCrawl, we computed IDF values for all terms specified by the experts, considering approximately 500,000 pages randomly chosen from WBR05, a representative collection of pages from the Brazilian Web that has also been used in [6].

In order to evaluate the importance of this pre-crawling phase in our focused crawling approach, we executed the following crawling processes:

- DS-1: topic "data structures course syllabi", using the IDF values calculated for each term;
- DS-2: topic "data structures course syllabi", considering the IDF values equal to 1 for all terms;
- DB-1: topic "databases course syllabi", using the IDF values calculated for each term;
- DB-2: topic "databases course syllabi", considering the IDF values equal to 1 for all terms;

- IR-1: topic "information retrieval course syllabi", using the IDF values calculated for each term;
- IR-2: topic "information retrieval course syllabi", considering the IDF values equal to 1 for all terms;
- Job-1: topic "job offers in computer science", using the IDF values calculated for each term;
- Job-2: topic "job offers in computer science", considering the IDF values equal to 1 for all terms;
- Sale-1: topic "sale offers of computer equipments", using the IDF values calculated for each term;
- Sale-2: topic "sale offers of computer equipments", considering the IDF values equal to 1 for all terms.

4.1.2 Baselines

As discussed in Section 2.2, classifiers are a natural option to guide a focused crawler. Therefore, in order to establish a baseline for comparing the results obtained with our genre-aware approach, we developed a focused crawler guided by an SVM Radial Basis Function (RBF) classifier [14], one of the best classifiers for this kind of application according to [35], in two distinct ways: (1) following the traditional focused crawling strategy described in Figure 2.3 and (2) following our proposed queuing strategy (changing the score of URLs in the priority queue that correspond to the sibling pages of a relevant URL). Thus, we considered as our baseline the following crawling processes, which have been executed using a focused crawler guided by an SVM RBF classifier:

- RBF-DS-1: topic "data structures course syllabi", following the traditional focused crawling strategy;
- RBF-DS-2: topic "data structures course" syllabi, following our proposed queuing strategy;
- RBF-DB-1: topic "databases course syllabi", following the traditional focused crawling strategy;
- RBF-DB-2: topic "databases course syllabi", following our proposed queuing strategy;

- RBF-IR-1: topic "information retrieval course syllabi", following the traditional focused crawling strategy;
- RBF-IR-2: topic "information retrieval course syllabi", following our proposed queuing strategy;
- RBF-Job-1: topic "job offers in computer science", following the traditional focused crawling strategy;
- RBF-Job-2: topic "job offers in computer science", following our proposed queuing strategy;
- RBF-Sale-1: topic "sale offers of computer equipments", following the traditional focused crawling strategy;
- RBF-Sale-2: topic "sale offers of computer equipments", following our proposed queuing strategy.

For each topic, the SVM RBF classifier was trained considering at least 30 pages related to the desired topic (i.e., positive examples of pages), a collection of 500,000 pages, chosen from WBR05, not related to the desired topic (negative examples of pages), and appropriate cost and gamma parameters determined experimentally. For instance, for the topic "databases course syllabi", the positive and negative examples were Web pages that corresponded, respectively, to syllabi of databases courses and other distinct subjects not related to syllabi of databases courses. The content of the examples pages was parsed by eliminating accents, capital letters and stopwords.

In the performed experiments, we measured the F1 level after the execution of each baseline crawling process. Table 4.3 shows the F1 levels obtained in these experiments. In order to measure the precision and mainly recall of the crawling process, it was necessary to consider a restricted subset of the Brazilian Web formed only by the pages included in the hosts of the specified seed pages. All pages in this subset were manually verified as relevant or not for the specified topics. Notice that these results correspond to both queuing strategies, since the F1 values remain the same and only the order in which the pages are visited changes.

Table 4.3: F1 levels obtained by the baseline crawling processes

Baseline crawling process	F1
RBF-DS	46.48%
RBF-DB	45.83%
RBF-IR	23.26%
RBF-Job	45.71%
RBF-Sale	66.87%

4.1.3 Parameter Setting

According to steps 07 and 09 of the procedure FCrawl, in order to determine the relevance of a fetched page to the topic of interest, we perform arithmetic combinations among previously calculated similarity values. Each combination is performed, simply, by calculating the weighted average of the similarity scores involved. Thus, it was necessary to establish a "weight of importance" for each similarity value. We established such a weighting scheme experimentally, varying the values of the weights and analyzing the results obtained. With respect to step 07, for the course syllabi crawling processes, the best results were reached considering the weight 5 for both genre (GW) and content (CW), since they are equally important. For the job and sale offers crawling processes, the best results were reached considering the weights 4 and 6 for genre and content respectively, since the genre related terms are more diverse. With respect to step 09, for all crawling processes, the best results were reached considering the weights 7 and 3, respectively for the genre-content combination (GCW) and the URL string (UW), since for most pages the terms that form their URLs are usually not very relevant.

Finally, for practical reasons, in our experiments we made the following decisions: (1) we disregarded pages of certain Brazilian Web domains (e.g., ".gov.br" that refers to Brazilian government sites) or that included some type of undesired content (e.g., ".exe"), which were not relevant to the context of the topics considered; (2) we limited the depth that the crawlers could reach within a site (maximum number of levels of the Web sub-graph of a site); (3) likewise, we limited the maximum number of URLs present in a page that should be followed; and (4) we standardized the content of the visited pages, by eliminating accents, capital letters and stopwords. The maximum depth was set to 7 and the maximum width to 200 pages. These values were determined experimentally and values beyond these

did not produce any significant gains.

Decisions 2 and 3 were key to make the experiment feasible in terms of time and necessary effort to determine the sets of relevant pages. Decision 3 is also based on results reported in the literature [21], that indicate that the best pages are in the higher levels of a site hierarchy and going deeper in the hierarchy does not provide gains. More importantly, these decisions were applied to both, our crawler and the baselines.

4.2 Effectiveness

As mentioned before, our genre-aware approach to focused crawling uses a set of guiding heuristics that has been designed with two main objectives: improving the F1 level of the crawling process and speeding up the crawling of relevant pages. In our experiments, we measured the F1 level after the execution of each crawling process. As already mentioned, in order to measure the precision and recall of a crawling process, it was necessary to consider a restrict subset of the Brazilian Web formed only by the pages that present the host of the seed pages, making it easy identifying all the relevant pages that should be retrieved. Thus, we notice that, in the executed crawling processes, our crawler visited almost 60,000 pages for the course syllabus subject, almost 200,000 pages for the job offer one and almost 300,000 pages for the sale offer one. Figures 4.1 and 4.2 show the F1 levels obtained by the crawling processes that use the IDF values calculated for each term and that consider the IDF values equal to 1 for all terms respectively, considering different threshold options.

According to Figures 4.1 and 4.2, it can be observed that, for all topics, our focused crawler reached F1 levels superior to 88%, with different similarity thresholds. That is, even considering topics with distinct characteristics, the results achieved by our focused crawler are very good. Moreover, once the F1 value starts to diminish, it never surpasses its peak value again since the recall measure also starts to diminish, meaning that it is not necessary to vary the threshold anymore. The crawling processes for job offers showed the worst results since, in these cases, the genre and content terms varies a lot, i.e., this topic does not use commom terms that clearly characterize both the genre and the content. Regarding the crawling processes for course syllabi, the discipline “data structures” shows the worst results due to the fact that the content of this discipline is dispersed in many distinct courses; thus, some syllabi have not been classified correctly because many content terms specified for this discipline did not appear in these syllabi.

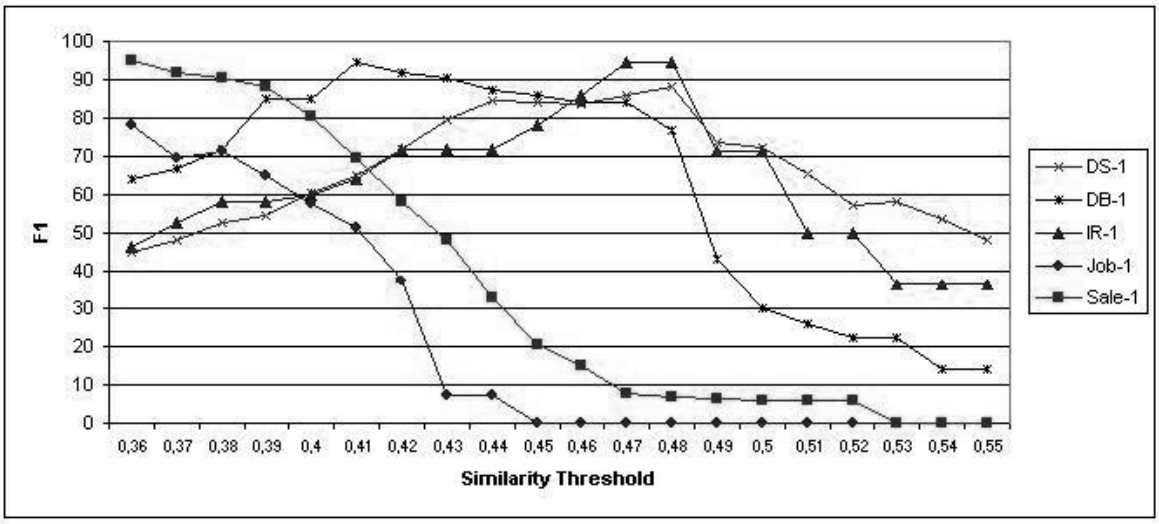

Figure 4.1: F1 x Similarity Threshold (our approach - IDF values calculated for each term)

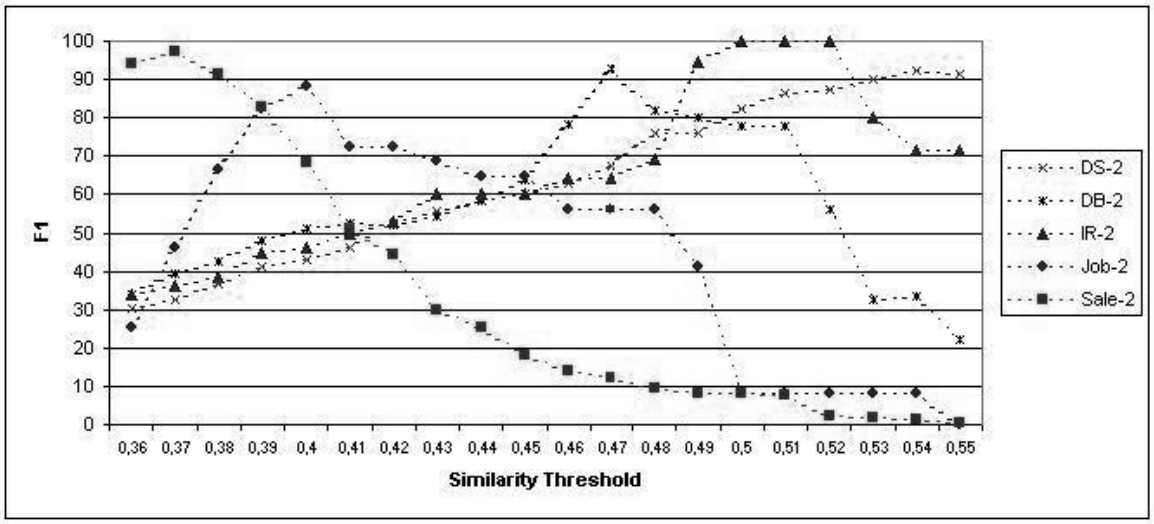

Figure 4.2: F1 x Similarity Threshold (our approach - IDF values equal to 1 for all terms)

We also notice that most crawling processes did not achieve 100% of F1. By manual inspection, we verified that this was mainly due to precision errors; for example, certain course syllabi included the content of other courses (e.g., a software engineering course included a databases module) and sale offers of certain products, such as digital cameras and cell phones, included content terms related to computer equipments. Moreover, some recall errors were due to ill designed syllabi, job offers or sale offers (e.g., some course syllabi, computer science job offers and computer equipment sale offers did not show any term that could properly characterize the respective genres).

It can also be observed, from Figures 4.1 and 4.2, that the pre-crawling phase (i.e., the calculation of the IDF values to each term specified by the expert) does not have a strong influence on the effectiveness of our genre-aware approach, since, for a wide range of threshold values, it reaches similar or even better F1 levels when we consider the IDF value equal to 1 for all terms. This allows us to disregard this pre-crawling, which has obvious implications in terms of performance.

In addition, these results show that our crawler clearly outperforms the crawler guided by the SVM RBF classifier, since it achieves, for all crawling processes, F1 levels that are significantly higher than the ones achieved by the crawler guided by the SVM RBF classifier (see Table 4.3). We also notice that the way we implement a focused crawler guided by a classifier (either following the traditional focused crawling strategy or following our proposed queuing strategy) does not influence the F1 level achieved. This is an important remark because it shows that the improvements we achieve with our queuing policy (see next) are a consequence of our approach that considers separately two kinds of evidence when analyzing the relevance of a Web page to a specific topic.

In sum, the great benefit of our proposed focused crawling strategy is to explicitly separate subject and genre concerns, which form, in fact, two orthogonal feature spaces. In order to further demonstrate the importance of considering separately these two sets of evidence, we also run, for all topics, our focused crawler without separating such evidence, i.e., considering a single set containing all genre and content terms specified for a topic together. Table 4.4 shows, for each crawling process run, the best F1 level achieved and the respective similarity threshold used to determine the relevance of a page to the corresponding topic. As we can see, the results of these experiments were much worse that the results obtained by the

focused crawler constructed according to our genre-aware approach (Figures 4.1 and 4.2).

Table 4.4: F1 levels obtained by our approach without separating genre and content

Crawling process	F1	Similarity threshold	Crawling process	F1	Similarity threshold
DS-1	72.39	0.44	DS-2	77.99	0.50
DB-1	78.26	0.39	DB-2	78.86	0.43
IR-1	51.43	0.39	IR-2	51.72	0.41
Job-1	28.60	0.39	Job-2	34.62	0.41
Sale-1	68.73	0.41	Sale-2	71.05	0.44

4.3 Efficiency and Scalability

A key issue for any crawler is efficiency, i.e., its capability of crawling relevant pages as fast as possible. Figures 4.3 and 4.4 show, for the similarity thresholds that achieved the best levels of F1 in the crawling processes DE-1, DB-1, IR-1, Job-1, Sale-1 (see Figure 4.1) and DE-2, DB-2, IR-2, Job-2, Sale-2 (see Figure 4.2) respectively, the percentage of relevant pages retrieved in comparison with the percentage of pages visited during the crawling processes. As we can see, our genre-aware approach is quite efficient, since a large percentage of relevant pages are retrieved after visiting only 15% of the pages and that, for all topics, at least 90% of all relevant pages are retrieved after visiting 60% of the pages. This is a consequence of our strategy of dynamically changing the crawling priority of the non-visited pages, as discussed in Chapter 3.

Analyzing Figures 4.3 and 4.4 more detailedly, we can see that once again the crawling processes for job offers showed the worst results for the same reason mentioned before. Likewise, regarding the crawling processes for course syllabi, the discipline “data structures” is again the one that shows the worst results due to the fact that its content is often scattered among pages from many distinct courses, which means that there would be more relevant pages to be crawled in other sites. On the other hand, the discipline “information retrieval”, that corresponds to a discipline not well-consolidated yet, shows the best results (only 20% of the visited pages required to find all relevant pages). This good performance might be due to the fact that there are few relevant pages on this subject to be crawled.

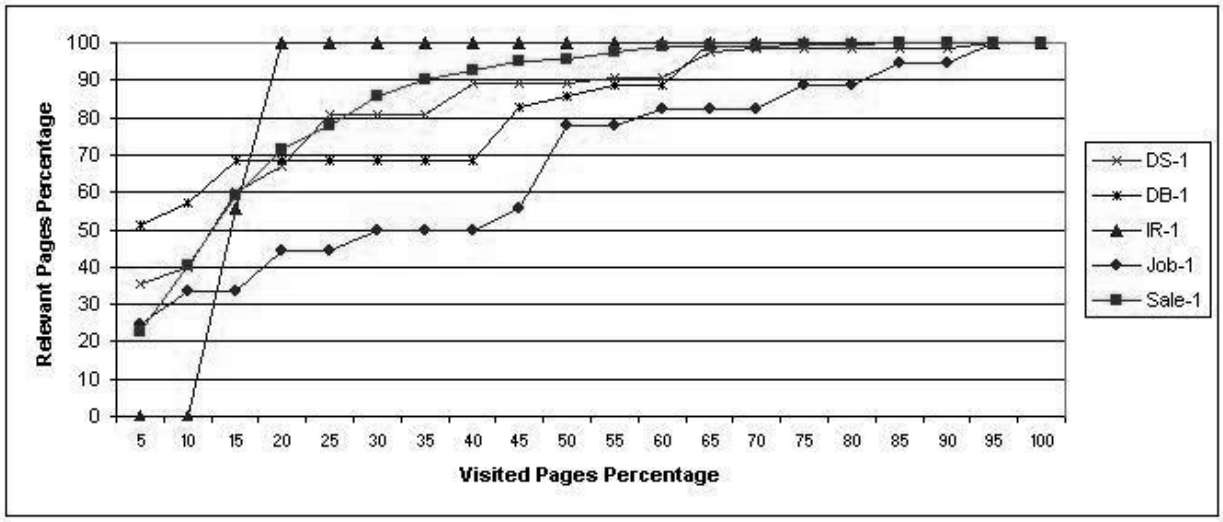

Figure 4.3: Percentage of relevant pages x Percentage of visited pages (our approach - IDF values calculated for each term)

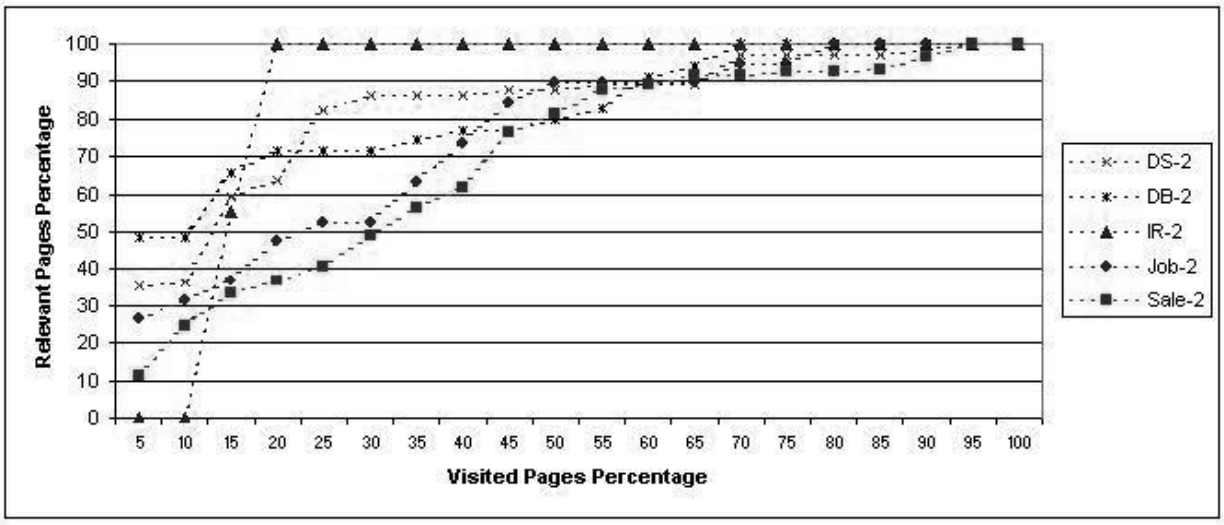

Figure 4.4: Percentage of relevant pages x Percentage of visited pages (our approach - IDF values equal to 1 for all terms)

Figures 4.5 and 4.6 show the percentage of relevant pages retrieved in comparison with the percentage of visited pages for the baseline crawling processes that follow the traditional focused crawling strategy and our proposed queuing strategy respectively. As we can see, our crawler is much more efficient than the crawler guided by the SVM RBF classifier since, for all crawling processes, it presents a better convergence to 100% of coverage of the relevant pages. Moreover, the traditional focused crawling strategy produces worse results than our proposed queuing strategy.

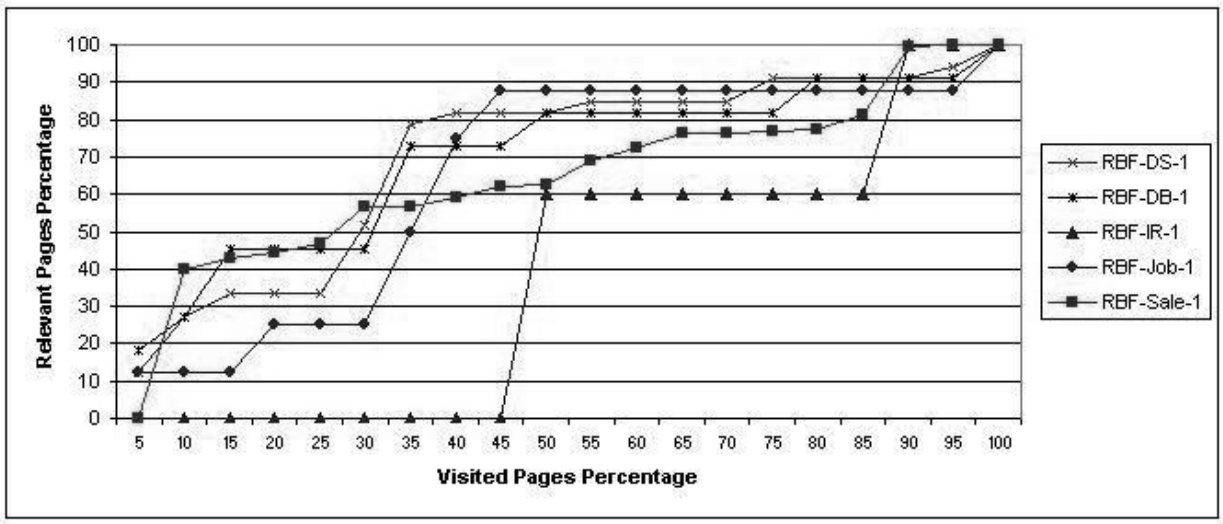

Figure 4.5: Percentage of relevant pages x Percentage of visited pages (baselines - traditional focused crawling strategy)

To evaluate the scalability of our genre-aware approach, an analysis of the average number of visited pages per relevant page retrieved was performed. The results are shown in Table 4.5, considering only the crawling processes that yielded the best results. As we can see, the baseline crawling processes, independently of the crawling strategy used, present an average number of visited pages per relevant page retrieved much higher than those presented by the crawling processes executed according to our approach. Moreover, comparing the average number of visited pages per relevant page in the baseline crawling processes, we notice that our proposed queuing strategy performs better than the traditional focused crawling strategy.

All crawling processes were executed on an Intel(R) 3 GHz Pentium(R) 4 CPU, with 1 GByte of RAM and 150 GBytes of hard disk. Our focused crawler took 3 to 4 hours to complete each crawling process, due to the simplicity of the set of heuristics it uses. On the other hand, the baseline crawling processes took 3 to 4 days to perform the same tasks; moreover,

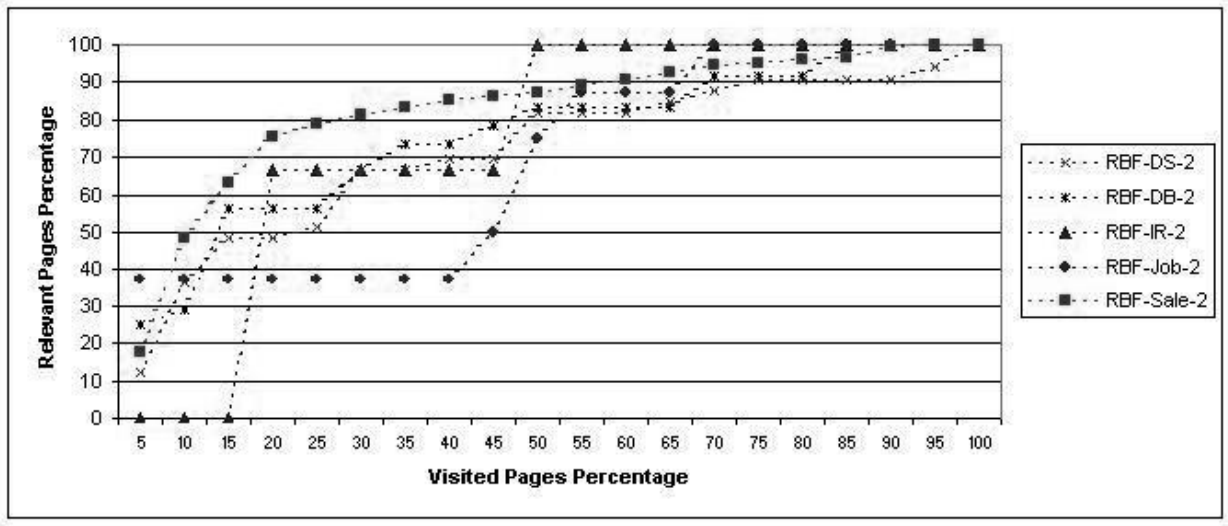

Figure 4.6: Percentage of relevant pages x Percentage of visited pages (baselines - our proposed queuing strategy)

Table 4.5: Number of visited pages per relevant page retrieved

Crawling (our approach)	Average number of visited pages	Crawling (baseline)	Average number of visited pages
DS-1	719.01	RBF-DS-1	1675.85
DS-2	710.55	RBF-DS-2	1625.97
DB-1	1046.3	RBF-DB-1	3984.0
DB-2	1105.4	RBF-DB-2	2037.3
IR-1	1104.33	RBF-IR-1	9178.00
IR-2	1199.00	RBF-IR-2	7656.67
Job-1	141.44	RBF-Job-1	336.00
Job-2	100.53	RBF-Job-2	232.88
Sale-1	84.75	RBF-Sale-1	172.81
Sale-2	87.20	RBF-Sale-2	161.55

they required an additional training phase that took around 4 hours.

4.4 Queuing Policies

As described in Chapter 3, our genre-aware focused crawling approach uses a new strategy to improve the crawling process that consists in dynamically changing the score of the URLs in the priority queue. This is done by changing the score of the URLs that correspond to sibling pages of a visited page with URL U to the score of U, if the score of U is higher than a given change threshold. In the experiment described in the previous subsection, we used a change threshold equal to 0.20, which was empirically determined as a suitable value.

In order to compare this queuing policy with others, another set of experiments was performed, considering the crawling process DB-2 with similarity threshold equal to 0.47 (the best threshold for this crawling process) and varying the queuing policies. The results are graphically illustrated in Figure 4.7. For a specific URL U with score S, the following queuing policies were considered:

- PS: change of the score of the URLs that correspond to child pages of U (this corresponds to the queuing policy used in traditional focused crawlers [9, 7, 20, 29, 35, 45]);
- FS-1: change of the score of the URLs that correspond to sibling pages of U, independently of the value of S;
- FS-2: change of the score of the URLs that correspond to sibling pages of U, if S is higher than 0.20;
- PFS-1: change of the score of the URLs that correspond to child pages of U and of the URLs that correspond sibling pages of U, independently of the value of S;
- PFS-2: change of the score of the URLs that correspond to child pages of U and of the URLs that correspond to sibling pages of U, if S is higher than 0.20.

From Figure 4.7, we can see that the PS queuing policy is much worse than the other policies that involve the change of the score of the URLs of the pages that are siblings of a visited page, especially when the percentage of visited pages is small. Comparing the FS-1 queuing policy with FS-2 and the PFS-1 queuing policy with PFS-2, we notice that, in general, using a change threshold yields better results, since it makes possible to change the

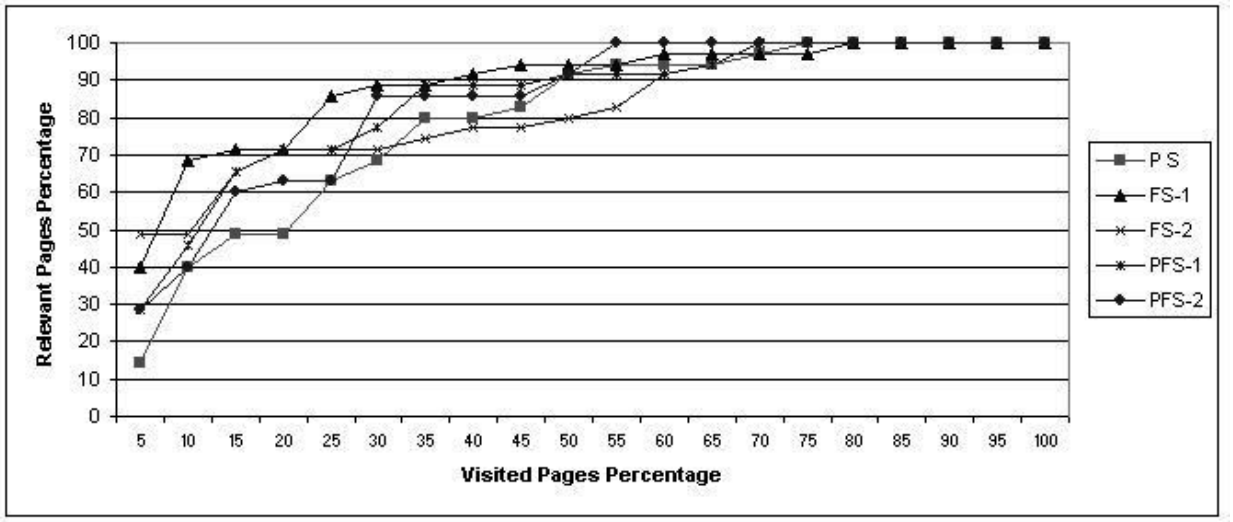

Figure 4.7: Percentage of relevant pages x Percentage of visited pages (varying the queuing policies)

score of URLs of child or sibling pages that only refer to relevant pages.

Moreover, considering the visited pages percentages lower than 30%, we notice that the queuing policies that change only the URLs of sibling pages of a visited page (FS-1 and FS-2) show best results than the queuing policies that change both the URLs of child pages and the URLs of sibling pages of a visited page (PFS-1 and PFS-2). However, the more pages our crawler visits, the worse the convergence to 100% of coverage of relevant pages. Thus, it can be concluded that queuing policies that change only the URLs of sibling pages of a visited page should be used when one wants to retrieve the highest possible number of relevant pages, with little time of execution for the specific tasks that we are considering in this work. On the other hand, the queuing policies that change the URLs of child and sibling pages of a visited page should be used when there is the need of retrieving the entire collection of relevant pages as fast as possible.

Chapter 5

Term Selection

As already discussed, our genre-aware approach requires an expert to specify a set of terms that describe the genre and the subject or content of the pages of interest. Although not a complex task, the selection of these terms might have an impact on the effectiveness of a crawler constructed according to our approach. Therefore, it is necessary to know how genre and content terms should be specified by an expert in order to produce significant results. Moreover, in order to reduce this effort, it is important to know whether semi-automatically generated terms produce results that are as good as those provided with terms specified by an expert.

Thus, this chapter analyzes the impact of term selection in our genre-aware approach to focused crawling. More specifically, in Section 5.1, we analyze the impact of the number of terms required in our genre-aware approach. Then, in Section 5.2, we propose and experimentally evaluate a simple, but effective strategy for semi-automatic generation of these terms.

5.1 Impact of the Number of Terms on the Crawling Process

In the experiments reported in Chapter 4, the sets of genre and content terms used to describe the respective topics of interest were specified by experts based exclusively on their judgment of which terms would be the most relevant to represent them. Particularly, no restriction was imposed on the number of such terms. Considering that the number of genre and content terms used to guide a crawler has a direct influence on the levels of F1

achieved by our approach, for instance, a large number of terms may improve precision but with a decrease in the recall level, it was necessary to analyze the impact of such terms on the crawling process.

To analyze the impact of the number terms on the effectiveness of a crawler implemented according to our approach, we performed a series of experiments in which we run a sequence of crawling processes for the same topics considered in Chapter 4, varying the number of genre and content terms specified by the expert. We notice that, based on the results reported in Section 4.2, we run all crawling processes considering the IDF value equal to 1 for all terms.

For each topic, we ran distinct crawling processes considering, respectively, 100%, 75%, 50% and 25% of the number of genre and content terms specified in Table 4.1. Then, to evaluate separately the influence of genre and content terms in the approach, for each topic, we run a sequence of crawling processes considering: (a) 100% of the genre terms and 75%, 50% and 25% of the content terms, and (b) 100% of content terms and 75%, 50% and 25% of the genre terms. For each crawling process, the reduction of the number of terms was done based on the expert judgment about which terms were less relevant to represent the subject. Tables 5.1 and 5.2 show, for each crawling process run, the best F1 level achieved and the respective similarity threshold used to determine the relevance of a page to the corresponding topic. For a better understanding of these tables, for each topic the following comments are due: (a) a crawling process named "16g24c", for example, means that it used 16 genre terms and 24 content terms to represent its topic; (b) the first four crawling processes vary the number of both genre and content terms simultaneously; (c) the middle three crawling processes vary the number of content terms, keeping fixed the number of genre terms; and (d) the last three crawling processes vary the number of genre terms, keeping fixed the number of content terms.

Analyzing the results shown in Tables 5.1 and 5.2, we can observe the following: (1) in the crawling processes for which we vary the number of genre and content terms simultaneously, as the number of terms diminishes, the best F1 levels are achieved when we consider higher similarity thresholds, since this prevents reducing precision, and (2) the crawling processes for which we vary only the number of genre terms achieve, in general, better F1 levels than the ones for which we vary only the number of content terms, implying that the number of genre terms has less impact on the crawling results than the number of content terms, i.e., the genre of the pages associated with a specific topic of interest are, in general, more easily

Table 5.1: Best F1 (varying number of terms) - topics related to the "course syllabi" genre

Discipline "data structures"			Discipline "databases"			Discipline "information retrieval"		
Crawling process	F1	Similarity threshold	Crawling process	F1	Similarity threshold	Crawling process	F1	Similarity threshold
16g16c	92.41	0.54	16g24c	92.54	0.47	16g24c	100.0	0.50
12g12c	87.65	0.65	12g18c	88.61	0.57	12g18c	96.77	0.63
08g08c	89.33	0.74	08g12c	92.11	0.65	08g12c	93.33	0.65
04g04c	84.35	0.79	04g06c	78.26	0.66	04g06c	75.00	0.76
16g12c	82.08	0.63	16g18c	86.75	0.52	16g18c	93.75	0.55
16g08c	84.47	0.68	16g12c	87.50	0.56	16g12c	75.68	0.61
16g04c	75.90	0.76	16g06c	72.73	0.61	16g06c	69.57	0.66
12g16c	90.32	0.67	12g24c	84.93	0.57	12g24c	96.77	0.67
08g16c	88.62	0.69	08g24c	88.00	0.58	08g24c	93.33	0.66
04g16c	83.33	0.73	04g24c	90.91	0.61	04g24c	76.92	0.70

Table 5.2: Best F1 (varying number of terms) - topics related to the "job offers" and "sale offers" genres

Topic "job offers in computer science"			Topic "sale offers of computer equipments"		
Crawling process	F1	Similarity threshold	Crawling process	F1	Similarity threshold
24g24c	88.37	0.40	24g48c	97.07	0.37
18g18c	65.87	0.55	18g36c	91.64	0.44
12g12c	61.11	0.63	12g24c	87.71	0.45
06g06c	73.17	0.68	06g12c	92.25	0.53
24g18c	63.41	0.55	24g36c	91.80	0.43
24g12c	59.09	0.59	24g24c	85.24	0.41
24g06c	70.00	0.63	24g12c	86.52	0.41
18g24c	65.52	0.47	18g48c	94.13	0.45
12g24c	64.29	0.52	12g48c	93.34	0.47
06g24c	70.29	0.59	06g48c	90.51	0.50

characterized than their content, which is usually broader.

We can also notice that, for the "course syllabi" crawling processes in which we varied only the number of genre terms, for two subjects (data structures and information retrieval), the fewer the number of terms, the lower the F1, as expected. The opposite phenomenon though occurred for the subject databases. This can be explained by the fact that, being databases a discipline with a well-consolidated content, course syllabi in this subject are more homogeneous and well structured. This even suggests a more common reuse of material. On the other hand, the dispersion of the content of data structures in many distinct and different courses as well as the constant evolution of the content in information retrieval courses promotes less homogeneous and structured syllabi for these disciplines. As such, the removal of genre terms for database courses allowed the retrieval of more syllabi, since matching constraints were reduced without loosing the characterization of the document genre. Evidence of this is given by the increasing in recall levels without loss of precision, when the corresponding similarity threshold was raised. This is an interesting finding that may show that the characterization of the genre of Web pages of a specific topic may not be completely dissociated from its subject.

Finally, it is important to notice that all crawling processes reported in Tables 5.1 and 5.2 achieved F1 levels higher than those achieved by the SVM-based crawler we implemented as our baseline (see Table 4.3). Thus, even using a small number of genre and content terms, focused crawlers implemented according to our genre-aware approach are more effective than traditional classifier-based crawlers.

5.2 Semi-automatic Generation of Terms

As already mentioned, although not a complex task, the selection of the genre and content terms required to describe the pages of interest and, therefore, to guide a crawling process according to our approach is usually done with the assistance of an expert on the respective application domain. In order to facilitate this task and help an expert to find the terms that most appropriately represent the pages of interest, we describe and evaluate in this section a strategy for the semi-automatic generation of such terms. This strategy becomes particularly appropriate when the expert already has a set of URLs of Web pages expressing the genre of interest or relevant Web pages related to the topic of interest.

Our strategy for the semi-automatic generation of genre and content terms related to a specific topic of interest is based on the procedure GenerateTerms described by Algorithm 3. This procedure takes as input the following parameters: (1) a set of URLs of Web pages that express the genre of interest (*GenreURLs*), (2) a set of URLs of relevant Web pages that are related to the topic of interest (*RelevantURLs*), (3) the number of genre terms required (*NG*), and (4) the number of content terms required (*NC*). For instance, if a user wants to crawl Web pages that include syllabi (genre) of database courses (content), the *GenreURLs* and *RelevantURLs* sets must contain URLs of pages that correspond, respectively, to syllabi of distinct courses and syllabi of database courses, and the *NG* and *NC* parameters must define, respectively, the number of genre and content terms required. This procedure outputs the sets of terms that represent the genre (*GenreTerms*) and the desired information content (*ContentTerms*). Here, we consider a term as being a sequence of, at most, three consecutive words.

In order to generate the sets of genre and content terms, procedure GenerateTerms uses a table, called *TermTable*, to keep track of the frequency of all terms found in the given pages and invokes the following pre-defined procedures and functions:

- FetchParsePage (*URL*, *L*): fetches and parses the page pointed by *URL*, returning in *L* the list of all its terms;
- UpdateTermTable (*L*): includes in *TermTable* the terms contained in the list *L*, updating their respective frequencies;
- FilterTerms (*N*): returns a set containing the *N* most frequent terms in the *TermTable*.

As we can see from Algorithm 3, procedure GenerateTerms works as follows. For each URL in *GenreURLs* (step 01), steps 02 and 03 are performed as follows: (a) the corresponding page is fetched and its content parsed, being the list of all its terms returned in *PageTerms*; (b) *TermTable* is then updated with the list of terms in *PageTerms*. After fetching and parsing all pages that express the desired genre, *FilterTerms* returns into *GenreTerms* the set of *NG* most frequent genre terms found in *TermTable* (step 04). Next, following similar steps (05 to 09), the set of content terms is generated. Notice that in step 07 we remove from the list of terms of each fetched page the set of genre terms previously generated since we now want to generate a set of terms that represents only the content (subject) of the pages of interest.

```
    Algorithm:Procedure GenerateTerms
    Input: GenreURLs, RelevantURLs, NG, NC
    Output: GenreTerms, ContentTerms
    begin
        Let TermTable be a table of terms with their respective frequencies;
        TermTable ⟵ " ";
1       foreach URL ∈ GenreURLs do
2           Fetch&ParsePage(URL,PageTerms);
3           UpdateTermTable(PageTerms);
        end
4       GenreTerms ⟵ FilterTerms(NG);
        TermTable ⟵ " ";
5       foreach URL ∈ RelevantURLs do
6           Fetch&ParsePage(URL,PageTerms);
7           PageTerms ⟵ PageTerms − GenreTerms;
8           UpdateTermTable(PageTerms);
        end
9       ContentTerms ⟵ FilterTerms(NC);
    end
```

Algorithm 3: The procedure GenerateTerms

The strategy for semi-automatic generation of genre and content related terms expressed by procedure GenerateTerms is very simple, very effective and requires only a small set of example pages. To evaluate such a strategy, we implemented the procedure GenerateTerms described by Algorithm 3 and ran it to generate, for each topic used in the previous experiments, two sets of 40 terms to represent, respectively, the genre and the content of pages of interest. Table 5.3 shows the number of URLs of example pages provided for generating the sets of terms for each topic. We notice that we used exactly the same pages as examples for the genre "course syllabi" for all three disciplines.

Table 5.3: Number of URLs provided for each topic

Topic	Pages of the desire genre	Relevant syllabus pages
Data structures course syllabi	36	31
Databases course syllabi	36	26
Information retrieval course syllabi	36	13
Job offers in computer science	30	30
Sale offers of computer equipments	25	35

Then, for each topic, we run four crawling processes using, respectively, 100%, 75%, 50%, and 25% of the genre and content terms generated by procedure GenerateTerms, selected according to their decreasing order of frequency. Next, for each topic, we run another

crawling process using a subset of the generated genre and content terms, now selected by an expert. Tables 5.4 and 5.5 show, for each crawling process, the best F1 level achieved and the respective similarity threshold used to determine the relevance of a page to the corresponding topic. The four first crawling processes are those for which we selected the terms according to their frequency and the last one is the one whose terms were selected by an expert. As in the previous section, we named the crawling processes according to the number of genre and content terms used to represent the topic of interest.

Table 5.4: Best F1 (generated terms) - topics related to the "course syllabi" genre

Discipline "data structures"			Discipline "databases"			Discipline "information retrieval"		
Crawling process	F1	Similarity threshold	Crawling process	F1	Similarity threshold	Crawling process	F1	Similarity threshold
40g40c	78.95	0.67	40g40c	67.69	0.64	40g40c	64.29	0.73
30g30c	84.39	0.71	30g30c	67.61	0.68	30g30c	64.29	0.76
20g20c	78.26	0.76	20g20c	69.23	0.66	20g20c	64.29	0.79
10g10c	78.95	0.79	10g10c	63.64	0.72	10g10c	58.06	0.80
15g19c	79.03	0.79	15g21c	80.90	0.66	15g15c	67.14	0.79

Table 5.5: Best F1 (generated terms) - topics related to the "job offers" and "sale offers" genres

Topic "job offers in computer science"			Topic "sale offers of computer equipments"		
Crawling process	F1	Similarity threshold	Crawling process	F1	Similarity threshold
40g40c	50.90	0.56	40g40c	85.00	0.42
30g30c	50.12	0.57	30g30c	84.81	0.46
20g20c	53.45	0.65	20g20c	85.06	0.52
10g10c	51.39	0.74	10g10c	84.63	0.58
22g18c	60.87	0.64	18g18c	86.49	0.51

Analyzing the results shown in Tables 5.4 and 5.5, we can observe the following: (1) the F1 levels achieved by the crawling processes are better than those achieved by the SVM-based crawler we used as a baseline (see Table 4.3), and (2) the F1 levels achieved by all crawling processes that used terms selected by the expert are very significant and correspond, for all topics but one ("data structures course syllabi"), to the best result. Therefore, we can say that our strategy for semi-automatic term generation, although very

simple, is very effective and provides a means to assist an expert in the task of specifying the sets of genre and content terms required by our genre-aware approach to focused crawling.

Finally, comparing the results shown in Tables 5.1 and 5.2 with the results shown in Tables 5.4 and 5.5, we again notice that, as the number of terms diminishes, the best F1 levels are achieved when we consider higher similarity thresholds, since this prevents reducing precision.

Chapter 6

Conclusions

6.1 Summary of the Work

Focused crawlers are an important class of programs that have as their main goal to efficiently crawl Web pages that are relevant to a specific topic of interest. In this thesis, we have proposed a novel focused crawling approach aimed at crawling pages related to specific topics that can be expressed in terms of genre and content information. Our approach adopts a particular queuing policy that is the main reason for the good performance of our approach since it allows for relevant pages to be crawled as soon as possible. The effectiveness, efficiency and scalability of our genre-aware approach are demonstrated by a set of experiments we conducted for crawling pages related to syllabi of specific computer science courses, job offers in the computer science field and sale offers of computer equipments. The results of these experiments show that focused crawlers constructed according to our approach achieve levels of F1 superior to 88%, requiring the analysis of no more than 60% of the visited pages in order to find 90% of the relevant pages. Moreover, we have evaluated the impact of the number of terms specified to describe the genre and content of the pages of interest on the crawling results using our genre-aware approach to focused crawling. In addition, we have proposed and experimentally evaluated a strategy for semi-automatic generation of these terms.

As shown by our experimental results, the major benefits of our focused crawling approach, compared with traditional crawlers based on classifiers are: (1) improvement of the level of F1 in the crawling process, (2) more efficiency in the crawling process since only a small percentage of pages is required to be visited to crawl a large percentage of relevant pages, and (3) higher scalability since our approach adopts a simple set of heuristics to determine

the relevance of a page as well as to guide the crawling, and does not require a training or any preprocessing phase. In addition, specifying a set of genre and content terms for a crawling process is not a complex task. A small set of terms selected by an expert is usually enough to produce good results. The task of selecting a proper set of terms is facilitated by the fact that the attention of the expert should be concentrated on the content terms, since the genre can be characterized by a very small number of terms. Moreover, our proposed strategy for semi-automatic generation of terms has shown to be very effective in assisting an expert (or even an ordinary user) in the task of specifying the sets of genre and content terms for a crawling process. We notice that more sophisticated strategies for automatic term generation, such as the one described in [19] and [25], could have been used but our main focus in this work was to show that our genre-aware approach to focused crawling does not depend on an expert to specify the sets of genre and content terms required to describe the pages of interest.

6.2 Future Work

Based on the development and results obtained in this thesis, we can indicate some directions for future work:

- Apply our genre-aware approach to real world applications. Currently, we are investigating its use for collecting product reviews for a shopping portal. In this case, the seed pages and the genre and content terms have been specified by people responsible for the portal, who, in future, will also analyze the pages retrieved by the crawling processes. In this application, each crawling process is related to a product review.

- Evaluate our strategy for semi-automatic generation of terms varying the number of URLs of example pages used. In the performed experiments, we evaluated our strategy considering a fix number of URLs, as presented in Table 5.3. Thus, the idea is to vary the number of URLs for each topic, in order to analyze the impact of the number of URLs on the effectiveness of our strategy.

- Extend our genre-aware approach to consider, besides genre and content, other aspects (e.g., geographic region, page structure). In this case, the idea is to exploit other important aspects in our approach to focused crawling in order to improve the effectiveness and the efficiency of the crawling process in specific situations.

- Investigate whether context graphs [16] can be exploited in our approach to improve crawling results. In this case, the idea is to evaluate whether our approach to focused

crawling can be adjusted in a way that the crawler is capable of modeling the context within which topically relevant pages occur on the web, in order to improve crawling results. Such a context model, called context graph, can capture typical link hierarchies within which valuable pages occur, as well as model content on documents that frequently co-occur with relevant pages.

- Use the idea behind our genre-aware approach to other information retrieval problems such as search and classification. In this case, the idea is to analyze whether our approach to focused crawling, which is based on genre and content information, can be adapted in order to help improving other information retrieval problems.

Bibliography

[1] G. Almpanidis, C. Kotropoulos, and I. Pitas. Combining Text and Link Analysis for Focused Crawling - An Application for Vertical Search Engines. *Information Systems*, 32(6):886–908, 2007.

[2] G. T. Assis, A. H. F. Laender, M. A. Gonçalves, and A. S. Silva. Exploiting Genre in Focused Crawling. In *Proceedings of the 14th Symposium on String Processing and Information Retrieval*, pages 62–73, Santiago, Chile, October 29-31 2007.

[3] G. T. Assis, A. H. F. Laender, M. A. Gonçalves, and A. S. Silva. The Impact of Term Selection in Genre-Aware Focused Crawling. In *Proceedings of the 23th ACM Symposium on Applied Computing*, pages 1158–1163, Fortaleza, Brazil, March 16-20 2008.

[4] R. A. Baeza-Yates and B. A. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press/Addison-Wesley, New York, NY, 1999.

[5] E. S. Boese. Stereotyping the web: Genre classification of web documents. Master's thesis, Computer Science Department, Colorado State University, Boulder, Colorado, USA, March 2005.

[6] K. A. V. Borges, A. H. F. Laender, C. B. Medeiros, and C. A. Davis. Discovering Geographic Locations in Web Pages Using Urban Addresses. In *Proceedings of the 4th ACM Workshop on Geographic Information Retrieval*, pages 31–36, Lisbon, Portugal, November 9 2007.

[7] P. De Bra and R. D. J. Post. Information Retrieval in the World Wide Web: Making Client-Based Searching Feasible. *Computer Networks and ISDN Systems*, 27(2):183–192, 1994.

[8] M. Burner. Crawling Towards Eternity: Building an Archive of the World Wide Web. *Web Techniques Magazine*, 2(5):37–40, 1997.

[9] S. Chakrabarti, M. Berg, and B. Dom. Focused crawling: A New Approach to Topic-Specific Web Resource Discovery. *Computer Networks*, 31(11–16):1623–1640, 1999.

[10] G. Chen and B. Choi. Web Page Genre Classification. In *Proceedings of the 23th ACM Symposium on Applied Computing*, pages 2353–2357, Fortaleza, Brazil, March 16-20 2008.

[11] H. Chen, Y. Chung, M. C. Ramsey, and C. C. Yang. A Smart Itsy Bitsy Spider for the Web. *American Society for Information Science*, 49(7):604–618, 1998.

[12] J. Cho and H. Garcia-Molina. The Evolution of the Web and Implications for an Incremental Crawler. In *Proceedings of the 26th International Conference on Very Large Databases*, pages 200–209, Cairo, Egypt, September 2000.

[13] J. Cho, H. Garcia-Molina, and L. Page. Efficient Crawling Through URL Ordering. *Computer Networks*, 30(1–7):161–172, 1998.

[14] C. Cortes and V. Vapnik. Support-Vector Networks. *Machine Learning*, 20(3):273–297, 1995.

[15] E. Dellandrea, H. Harb, and L. Chen. Zipf, Neural Networks and SVM for Musical Genre Classification. In *Proceedings of the 5th IEEE International Symposium on Signal Processing and Information Technology*, pages 57–62, Athens, Greece, December 18-21 2005.

[16] M. Diligenti, F. Coetzee, S. Lawrence, C. L. Giles, and M. Gori. Focused Crawling Using Context Graphs. In *Proceedings of the 26th International Conference on Very Large Data Bases*, pages 527–534, Cairo, Egypt, September 10-14 2000.

[17] D. Florescu, A. Y. Levy, and A. O. Mendelzon. Database Techniques for the World-Wide Web: A Survey. *SIGMOD Record*, 27(3):59–74, 1998.

[18] P. Foltz. Improving Human-proceedings Interaction: Indexing the CHI Index. In *Proceedings of the Conference on Human Factors in Computing Systems*, pages 101–102, Denver, Colorado, USA, May 07-11 1995.

[19] E. Glover, D. Pennock, S. Lawrence, and R. Krovetz. Inferring Hierarchical Descriptions. In *Proceedings of the 11th International Conference on Information and Knowledge Management*, pages 507–514, McLean, Virginia, USA, November 4-9 2002.

[20] M. Herscovici, M. Jacovi, Y. S. Maarek, D. Pelleg, M. Shtalhaim, and S. Ur. The Shark-Search Algorithm - An Application: Tailored Web Site Mapping. *Computer Networks*, 30(1–7):317–326, 1998.

[21] A. Heydon and M. Najork. Mercator: a Scalable, Extensible Web Crawler. *World Wide Web*, 2(4):219–229, 1999.

[22] J. Johnson, K. Tsioutsiouliklis, and C. L. Giles. Evolving Strategies for Focused Web Crawling. In *Proceedings of the 20th International Conference on Machine Learning*, pages 298–305, Washington, DC, USA, 2003.

[23] A. Kontostathis. Essential Dimensions of Latent Semantic Indexing (LSI). In *Proceedings of the 40th Hawaii International Conference on Systems Science*, page 73, Waikoloa, Big Island, HI, USA, January 3-6 2007.

[24] J. P. Lage, A. S. Silva, P. B. Golgher, and A. H. F. Laender. Automatic Generation of Agents for Collecting Hidden Web Pages for Data Extraction. *Data and Knowledge Engineering*, 49(2):177–196, 2004.

[25] K. Lagus and S. Kaski. Keyword Selection Method for Characterizing Text Document Maps. In *Proceedings of the 9th International Conference on Artificial Neural Networks*, pages 371–376, Edinburgh, UK, September 7-10 1999.

[26] S. Lawrence and C. L. Giles. Accessibility of information on the Web. *Intelligence*, 11(1):32–39, 2000.

[27] H. Liu, J. Janssen, and E. E. Milios. Using HMM to Learn User Browsing Patterns for Focused Web Crawling. *Data and Knowledge Engineering*, 59(2):270–291, 2006.

[28] A. McCallum, K. Nigam, J. Rennie, and K. Seymore. Automating the Construction of Internet Portals with Machine Learning. *Information Retrieval*, 3(2):127–163, 2000.

[29] F. Menczer, G. Pant, and P. Srinivasan. Topical Web Crawlers: Evaluating Adaptive Algorithms. *ACM Transactions on Internet Technology*, 4(4):378–419, 2004.

[30] F. Menczer, G. Pant, P. Srinivasan, and M. E. Ruiz. Evaluating Topic-Driven Web Crawlers. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 241–249, New Orleans, Louisiana, USA, September 9-13 2001.

[31] P. Muller and D. R. Insua. Issues in Bayesian Analysis of Neural Network Models. *Neural Computation*, 10(3):749–770, 1998.

[32] M. Najork and J. L. Wiener. Breadth-First Search Crawling Yields High-Quality Pages. In *Proceedings of the 10th International World Wide Web Conference*, pages 114–118, Hong Kong, China, May 2001.

[33] A. Ntoulas, P. Zerfos, and J. Cho. Downloading Textual Hidden Web Content Through Keyword Queries. In *Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 100–109, Denver, CO, USA, June 07-11 2005.

[34] G. Pant and F. Menczer. Topical Crawling for Business Intelligence. In *Proceedings of the 7th European Conference on Research and Advanced Technology for Digital Libraries*, pages 233–244, Trondheim, Norway, August 17-22 2003.

[35] G. Pant and P. Srinivasan. Learning to Crawl: Comparing Classification Schemes. *ACM Transactions on Information Systems*, 23(4):430–462, 2005.

[36] G. Pant and P. Srinivasan. Link Contexts in Classifier-Guided Topical Crawlers. *IEEE Transactions on Knowledge and Data Engineering*, 18(1):107–122, 2006.

[37] G. Pant, P. Srinivasan, and F. Menczer. Crawling the web. In M. Levene and A. Poulovassilis, editors, *Web Dynamics: Adapting to Change in Content, Size, Topology and Use*, pages 153–178. Springer-Verlag, 2004.

[38] G. Pant, K. Tsioutsiouliklis, J. Johnson, and C. L. Giles. Panorama: Extending Digital Libraries with Topical Crawlers. In *Proceedings of the 4th ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 142–150, Tuscon, AZ, USA, June 7-11 2004.

[39] S. Raghavan and H. Garcia-Molina. Crawling the Hidden Web. In *Proceedings of the 27th International Conference on Very Large Data Bases*, pages 129–138, San Francisco, CA, USA, September 11-14 2001.

[40] J. Rennie and A. McCallum. Using Reinforcement Learning to Spider the Web Efficiently. In *Proceedings of the 16th International Conference on Machine Learning*, pages 335–343, Bled, Slovenia, June 27-30 1999.

[41] I. Rish. An Empirical Study of the Naïve Bayes Classifier. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence*, pages 41–46, Seattle, Washington, USA, August 4-10 2001.

[42] M. A. Rosso. Using genre to improve web search. Master's thesis, School of Information and Library Science, University of North Carolina, Chapel Hill, North Carolina, USA, August 2005.

[43] A. S. Silva, E. A. Veloso, P. B. Golgher, B. A. Ribeiro-Neto, A. H. F. Laender, and N. Ziviani. CoBWeb - A Crawler for the Brazilian Web. In *Proceedings of the String Processing and Information Retrieval Symposium & International Workshop on Groupware*, pages 184–191, Cancun, Mexico, September 21-24 1999.

[44] S. Sizov, M. Theobald, S. Siersdorfer, G. Weikum, J. Graupmann, M. Biwer, and P. Zimmer. The BINGO! System for Information Portal Generation and Expert Web Search. In *Proceedings of the 1st Conference on Innovative Data Systems Research*, Asilomar, CA, USA, January 5-8 2003.

[45] P. Srinivasan, F. Menczer, and G. Pant. A General Evaluation Framework for Topical Crawlers. *Information Retrieval*, 8(3):417–447, 2005.

[46] E. Stamatatos, G. Kokkinakis, and N. Fakotakis. Automatic Text Categorization in Terms of Genre and Author. *Computational Linguistics*, 26(4):471–495, 2000.

[47] C. Tantug and G. Eryigit. Performance analysis of naïve bayes classification, support vector machines and neural networks for spam categorization. In *Applied Soft Computing Technologies: The Challenge of Complexity*, pages 495–504. Springer-Verlag, 2006.

[48] M. L. A. Vidal, A. S. Silva, E. S. Moura, and J. M. B. Cavalcanti. Structure-Driven Crawler Generation by Example. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 292–299, Seattle, Washington, USA, August 06-11 2006.

[49] C. Walt and E. Barnard. Data Characteristics that Determine Classifier Performance. In *Proceedings of the 16th Annual Symposium of the Pattern Recognition Association of South Africa*, pages 160–165, Parys, South Africa, 29 November - 1 December 2006.

[50] T. Yoshioka, G. Herman, J. Yates, and W. Orlikowski. Genre taxonomy: a Knowledge Repository of Communicative Actions. *ACM Transactions on Information Systems*, 19(4):431–456, 2001.

[51] Sven Meyer zu Eissen and Benno Stein. Genre Classification of Web Pages: User Study and Feasibility Analysis. In Susanne Biundo, Thom Frühwirth, and Günther Palm, editors, *KI 2004: Advances in Artificial Intelligence*, volume 3228 LNAI of *Lecture Notes in Artificial Intelligence*, pages 256–269. Springer, September 2004.