Multiagentes para a Filtragem de Páginas *Web*

Flávia Coimbra Delicato, Luci Pirmez e Luiz Fernando Rust da Costa Carmo
NCE/UFRJ - Núcleo de Computação Eletrônica - Universidade Federal do Rio de janeiro
Tel: 021 5983159 - Caixa Postal: 2324 Rio de Janeiro RJ Brasil
E-mails: flavia@eng.uerj.br, luci, rust@nce.ufrj.br

Resumo

Atualmente, a Internet disponibiliza uma extensa quantidade de informações, para uma vasta gama de usuários, tornando difícil sua manipulação. O presente trabalho propõe o uso de um sistema multiagentes inteligentes para a filtragem personalizada de informações na Web. O sistema proposto é formado por um conjunto de agentes autônomos e adaptativos, cujo objetivo é fornecer automaticamente informações relevantes de acordo com as preferências de seus usuários. Os agentes aprendem através de realimentação do usuário e refinam as suas buscas, obtendo resultados melhores ao longo do tempo. Esse artigo apresenta a descrição do sistema e os promissores resultados iniciais obtidos em testes feitos em ambiente simulado. O sistema proposto demonstrou ser uma ferramenta útil para reduzir a quantidade de informações com que o usuário deve lidar.

Abstract

With the current growth of the information available in Internet, users are facing an information overload. This work proposes a multiagent system for Web pages personalized filtering. The system is composed by a set of autonomous and adaptive agents that automatically provide relevant documents to the user according to a preferences profile. The agents learn with the user feedback and attempt to produce better results over time. This work presents the system description and the promising results of tests performed in a simulated environment. The proposed system proved to be a useful tool to recommend successfully relevant information to a well-defined preferences user.

Palavras-chaves: Sistemas multiagentes, *Internet*, filtragem de informações.

1 Introdução

O crescimento exponencial da quantidade de informações disponíveis na Internet vem provocando um impacto significativo na comunidade de usuários. A introdução da *World Wide Web* (WWW) foi a principal responsável pelo crescimento explosivo nas publicações da Internet, tornando a rede acessível a uma gama muito grande de usuários leigos.

Esse grande aumento das informações disponibilizadas pela Internet, embora favoreça a disseminação do conhecimento e a obtenção de produtos e serviços, também torna a procura de material relevante um verdadeiro desafio. Surgem questões sobre como os usuários serão capazes de localizar a informação necessária, ou como poderão encontrar a melhor oferta para um determinado serviço. Uma possível solução para este problema consiste no uso de agentes.

Agentes podem ser definidos como softwares cujo objetivo é realizar tarefas em benefício de seus usuários, geralmente de forma autônoma, desempenhando o papel de assistentes pessoais.

Este trabalho propõe o uso de agentes inteligentes para a filtragem personalizada de informações. O sistema proposto é formado por um conjunto de agentes autônomos, adaptativos e fixos, com o objetivo de satisfazer as necessidades de informação dos seus usuários. Os agentes recebem realimentação do usuário sobre a relevância das informações recuperadas e refinam sua busca, obtendo resultados melhores ao longo do tempo.

Os agentes são autônomos pelo fato de poderem executar suas tarefas sem a presença do usuário, baseando-se em um perfil de preferências previamente construído. Além disso, os agentes podem processar informações obtidas da Internet sem necessidade de manter conexão durante todo o tempo de processamento. Essa característica é uma das feições que classificam um agente como autônomo [Nissen, 1995].

O sistema é adaptativo porque aprende as preferências do usuário e adapta-se às suas mudanças ao longo do tempo. Os mecanismos de aprendizagem adotados pelos agentes são a realimentação por relevância, amplamente usado em sistemas de recuperação de informações [Frakes, 1992], e algoritmos genéticos, técnica de inteligência artificial baseada nos princípios da evolução natural [Booker, 1990], [Goldberg, 1989], [Goldberg, 1994]. Para a representação das informações é usado o modelo vetor espacial [Salton, 1989], onde consultas e documentos são representados como vetores em um espaço vetorial. Optou-se por esse modelo devido a sua eficiência comprovada em diversos trabalhos na área de recuperação de informações [Sheth, 1994], [Balabanovic, 1997], [Buckley, 1994], e pela sua relativa facilidade de implementação.

Os resultados descritos no trabalho foram obtidos através de uma série de sessões com usuários simulados. Para a avaliação do desempenho do sistema foi adotada a medida da distância normalizada (ndpm), sugerida em [Yao, 1995].

Este artigo está organizado da seguinte forma: a seção 2 apresenta uma revisão dos trabalhos relacionados ao presente assunto. Na seção 3 é feita uma breve introdução aos sistemas de filtragem de informações. A seção 4 descreve o sistema desenvolvido, abordando a metodologia e o ambiente de desenvolvimento utilizados e detalhando a arquitetura do sistema. A análise dos resultados é apresentada na seção 5 e, finalmente, algumas conclusões são descritas na seção 6.

2 Revisão Bibliográfica

Variações da técnica de realimentação por relevância têm sido estudadas no contexto da tarefa de roteamento de informações, conforme descrito nas conferências TREC (*TExt Retrieval Conferences*) [Harman, 1994] [Buckley, 1994] [Allan, 1995]. Alguns trabalhos de filtragem de informações que utilizam a realimentação por relevância como mecanismo de

aprendizado são o de Sheth e Maes [Sheth, 1993] e o de Foltz e Dumais [Foltz, 1992]. Há, também, vários trabalhos de comparações entre essa técnica e técnicas de aprendizado não-incrementais [Lang, 1995] [Pazzani, 1996]. Técnicas não-incrementais necessitam de um grande número de exemplos necessários antes do algoritmo de aprendizado poder ser aplicado [Balabanovic, 1997].

O sistema Newst [Sheth, 1994] é um agente que usa realimentação por relevância e algoritmos genéticos (AG) para fornecer filtragem personalizada de artigos da *usenet*.

Outros sistemas baseados em agentes usam diferentes técnicas para tentar detectar padrões no comportamento do usuário. Por exemplo, o *InfoScope* [Fischer, 1991] aprende usando um sistema baseados em regras que registra os tópicos interessantes vistos pelo usuário no passado. As recomendações de novos tópicos para o usuário são feitas com base em quão recentes, frequentes e espaçados são os tópicos vistos anteriormente. Da mesma forma, sistemas de navegação assistida [Armstrong, 1995] recomendam links para páginas baseando-se em seções da *Web* já visitadas pelo usuário.

Balabanovic [1998] propôs um sistema multiagentes que usa a abordagem de filtragem de texto baseada em conteúdo em conjunto com filtragem colaborativa na construção de um sistema de recomendação de páginas *Web*. Esse trabalho adota o modelo vetor espacial [Salton, 1983], o método de aprendizado baseado em realimentação por relevância e sugere o uso de algoritmos genéticos [Goldberg, 1989] como possível solução para alguns dos problemas encontrados na filtragem baseada em texto.

3 Sistemas de Filtragem de Informações

Sistemas de filtragem de informações envolvem repetidas interações ao longo de múltiplas sessões onde os usuários possuem objetivos a longo prazo. Esses sistemas diferem dos mecanismos de busca de informações, onde os usuários tipicamente têm uma necessidade de informação a curto prazo, que é satisfeita em uma única sessão.

Um sistema de filtragem de informações assiste ao usuário filtrando o fluxo de dados e liberando apenas as informações relevantes. As preferências de informações variam muito de acordo com o usuário, portanto, esses sistemas devem ser altamente personalizados. Um sistema de filtragem personalizado deve satisfazer 3 requisitos:

- Especialização: uma vez que a filtragem envolve interações repetidas com o usuário, o sistema deve ser capaz de identificar padrões no seu comportamento; o sistema deve inferir seus hábitos e especializar-se a eles, isto é, recomendar o máximo de assuntos relevantes e o mínimo de irrelevantes;
- Adaptação: Os interesses do usuário não devem ser considerados constantes; quando esses mudarem, o sistema deve ser capaz de perceber e adaptar seu comportamento a essas mudanças;
- Exploração: um sistema de filtragem deve ser capaz de explorar novos domínios de informações para encontrar assuntos de interesse potencial para o usuário.

Atualmente encontram-se na literatura três abordagens distintas para os sistemas de filtragem de informações personalizados:

- sistemas que se baseiam no perfil do usuário para filtrar as informações, tentando adequá-las

aos seus interesses e expectativas [Brusilovsky, 1994];

- sistemas que realizam a filtragem de forma cooperativa, compartilhando informações [Twidale, 1995]; e
- sistemas que utilizam agentes, nos quais a mobilidade, autonomia e habilidade de interagir independentemente da presença de seus usuários são fatores fundamentais [Nissen, 1995].

A abordagem adotada neste trabalho baseia-se na tecnologia de agentes para realizar a filtragem personalizada de informações.

4 O Sistema Fenix

O sistema desenvolvido, denominado de sistema Fenix, é formado por um conjunto de agentes autônomos, adaptativos e fixos, cuja função é satisfazer as necessidades de informações de seus usuários. Os agentes realizam buscas por documentos na Internet, recebem realimentação do usuário sobre a relevância dos ítens recuperados e constróem um perfil de interesses, o qual vai se especializando ao seu usuário particular ao longo de múltiplas sessões de interação. Os resultados das buscas vão sendo gradualmente melhorados conforme o perfil se ajusta aos interesses específicos de seu usuário, com um número crescente de documentos relevantes sendo recuperados e irrelevantes sendo descartados. O mecanismo de aprendizagem adotado pelos agentes é baseado em realimentação por relevância e algoritmos genéticos. Os perfis usados para filtrar as informações são compostos de termos extraídos dos documentos pesquisados. O modelo adotado para a representação de perfis e documentos é o vetor-espacial [Salton, 1983]. O domínio de atuação dos agentes é a *Web*, sendo que uma página WWW é considerada como um item de informação para o sistema.

No sistema proposto, cada agente é modelado com um grupo de perfis individuais. Em conjunto, todos os perfis tentam atender aos interesses de um usuário e adaptar-se a eles. Um usuário pode ter vários agentes, cada um atendendo às suas necessidades de informação em um determinado domínio.

Cada agente é responsável por disparar a execução de tarefas de busca e filtragem, uma para cada perfil. Os agentes de busca comunicam-se com diferentes mecanismos de busca existentes na *Web* e classificam os documentos recuperados sem necessidade de interação com o usuário, inferindo as preferências do mesmo a partir de seu perfil. Os documentos selecionados para apresentação são aqueles com maior grau de similaridade com o perfil correspondente. O usuário pode fornecer realimentação positiva ou negativa para os documentos apresentados. Essa realimentação é usada para modificar o perfil responsável pela busca do respectivo documento. Esse mecanismo é conhecido como realimentação por relevância e é a principal forma do sistema Fenix aprender.

O algoritmo genético funciona como um mecanismo de aprendizado complementar para os agentes, visando introduzir diversidade nos resultados das buscas e, assim, aumentar a capacidade exploratória do sistema.

4.1 Metodologia e Ambiente de Desenvolvimento

O Sistema Fenix foi desenvolvido segundo a abordagem de orientação a objetos. Para a identificação dos objetos que compõem o sistema Fenix foi adotada a infra-estrutura MVC (modelo-vista-controle), um modelo clássico de arquitetura usado em sistemas orientados a objetos [Burbeck, 1987]. Nele, as classes são organizadas em tríades de componentes com diferentes finalidades. Durante a fase de projeto, os objetos identificados durante a análise foram organizados como módulos funcionais, cada um responsável por executar uma parte essencial do sistema.

A linguagem de programação adotada para a implementação do sistema foi Java, da *Sun Microsystems*, e o ambiente de desenvolvimento foi o Jbuilder[1], da *Borland Corporation*, que utiliza o JDK (*Java Development Kit*) versão 1.1.

O sistema foi implementado como um aplicativo Java que roda localmente na máquina do usuário.

4.2 Arquitetura

O sistema Fenix é composto de seis módulos, implementados como conjuntos de classes relacionadas. Os módulos funcionais que compõem o sistema são (figura 1):

- módulo de interface: permite que o usuário crie seus agentes para assuntos específicos, carregue agentes já existentes, leia documentos recuperados e forneça realimentação para documentos lidos.
- módulo de aprendizado: é responsável por garantir que o conjunto de perfis reflita os interesses do usuário, adaptando-se a eles ao longo do tempo. É composto pelo sub-módulo de realimentação por relevância e pelo sub-módulo de algoritmos genéticos.
- módulo de filtragem das informações: é responsável por encontrar documentos semelhantes aos perfis dos usuários. O processo de filtragem consiste em transformar documentos em suas representações vetoriais, calcular valores de similaridade (*score*) entre documentos e perfis, ordenar os documentos de acordo com seu score, e apresentar ao usuário apenas aqueles com score maior que um limite pré-fixado;
- módulo de busca: é responsável por invocar mecanismos de busca existentes na *Web* passando-lhes as palavras-chaves do usuário, obter seus resultados, e armazená-los em um banco de dados local;
- banco de dados local: é composto por um conjunto de arquivos nos formatos ASCII e binário, contendo os dados dos usuários, os seus respectivos perfis e as páginas capturadas na rede WWW.
- módulo controlador: é responsável por controlar o comportamento global do sistema, o comportamento do conjunto de agentes de um usuário e o comportamento específico de cada agente.

A seguir, será feita a descrição detalhada de cada um dos módulos.

[1] Disponível em: http://www.borland. com/jbuilder/

4.2.1 Módulo de Interface

Este módulo apresenta uma interface gráfica através da qual é feita a interação com o usuário. A interação do usuário com o sistema Fenix começa com o seu cadastramento, onde são fornecidos dados pessoais e escolhidos um *login* e uma senha de acesso. O usuário, após se identificar, pode escolher entre criar um novo agente, carregar um agente já existente ou ativar o modo de execução autônoma do sistema.

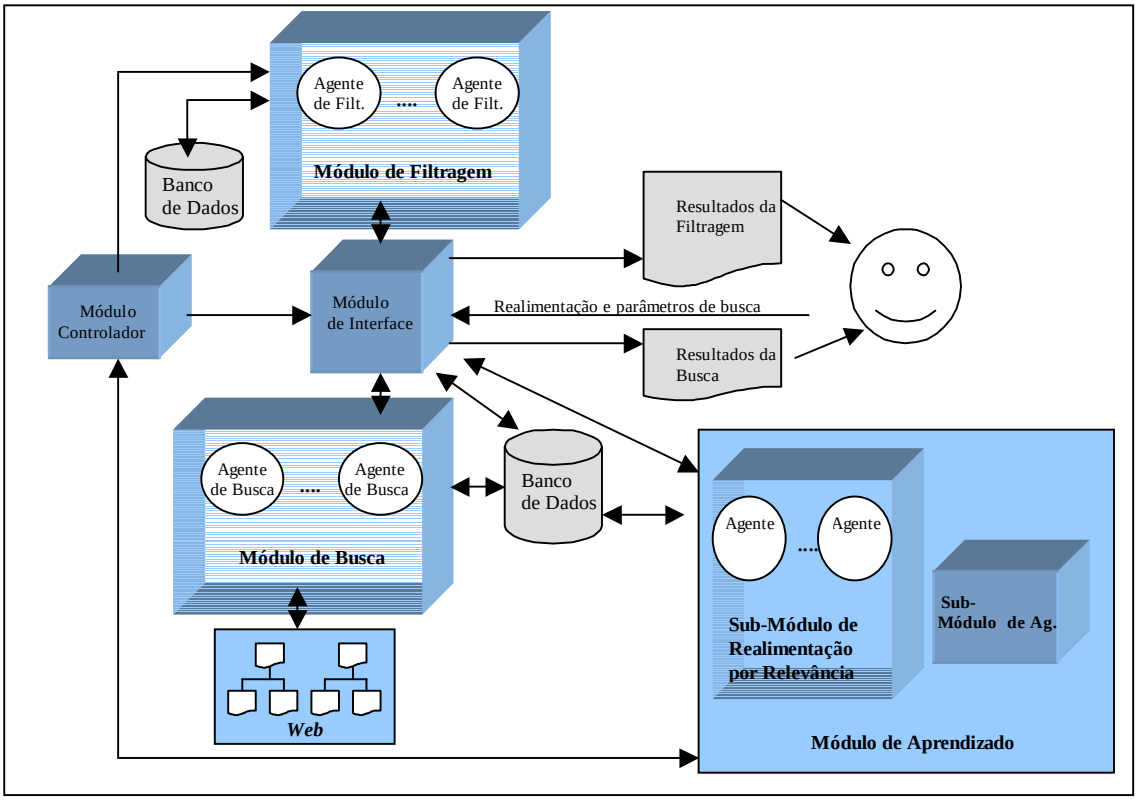

Figura 1: Arquitetura do Sistema Fenix

Ao criar um novo agente, o usuário deve escolher um nome e fornecer os seguintes parâmetros de busca: número máximo de páginas por documento, profundidade de links por página, número máximo de documentos recuperado por sessão, e a expressão de consulta. Uma consulta no sistema Fenix é uma combinação de palavras-chaves (tecnicamente chamadas de *termos*) e do conectivo lógico &. Como resultado da busca inicial, é apresentada uma lista com os nomes dos documentos recuperados. Após a leitura dos documentos desejados, o usuário pode fornecer realimentação positiva ou negativa conforme sua relevância para aquele assunto específico.

O usuário pode visualizar as páginas dos documentos através de um botão que ativa o navegador local, por exemplo o *Netscape* ou o *Internet Explorer*, o qual deve estar

configurado no sistema.

Ao terminar uma sessão, o usuário pode ou não salvar o agente recém-criado. Caso opte por salvar o agente, são armazenadas as referências para os documentos que receberam realimentação positiva (suas URLs). Os conteúdos desses documentos são usados para criar os vetores de termos e pesos que irão compor os perfis iniciais referentes àquele assunto e ao usuário específico. Cada documento tem, associado a ele, um campo de "status", que indica se o mesmo já foi ou não lido, se já foi avaliado pelo agente ou se é um documento que faz parte do perfil inicial do agente. Além disso, possui os campos "feedback", que armazena o valor da realimentação fornecida pelo usuário, e "score", que armazena o valor de similaridade com relação ao perfil, calculado pelo agente.

O usuário, após sua identificação, pode carregar um agente existente selecionando um dentre uma lista de agentes pertencentes a ele. A seguir, pode escolher uma das seguintes ações: ler algum documento recuperado, fornecer realimentação sobre algum documento lido ou iniciar uma nova busca.

Além das opções de criar ou carregar agentes, o usuário pode selecionar o “Modo Autônomo”, no qual são realizadas sessões de busca e filtragem sem interação com o usuário, a intervalos de tempo pré-fixados. Caso haja documentos novos resultantes dessas buscas, o usuário é informado pelo sistema.

4.2.2 Módulo de Filtragem

Após uma busca inicial, baseada nas palavras-chaves fornecidas pelo usuário, o sistema apresenta uma lista de documentos para serem lidos e avaliados pelo usuário. Os ítens que recebem realimentação positiva são convertidos para a sua representação vetorial e são usados para criar os perfis iniciais do usuário. Cada documento com realimentação positiva dá origem a um arquivo de perfil, que contém, inicialmente, a URL desse documento, seu status, o valor do feedback recebido, bem como a representação vetorial do documento (termos e pesos).

Nas buscas posteriores, cada novo documento recuperado é convertido para sua representação vetorial, sua similaridade com relação ao respectivo perfil é calculada e um score lhe é atribuído. Apenas documentos com score maior do que um limite fixado são apresentados. O agente controlador é responsável por reunir os documentos gerados pelos diferentes perfis, classificá-los de acordo com suas similaridades e apresentar os de maior relevância para o usuário.

A representação adotada neste trabalho é baseada no modelo vetor espacial [Salton, 1983] comumente usada em sistemas de recuperação de informações. Nesse modelo, as palavras (termos) que compõem os documentos são extraídas e pesos lhes são atribuídos na proporção de sua presumida relevância para fins de identificação do conteúdo do documento. Assim, um documento é representado como um vetor de pesos e termos [Salton, 1983].

Adotou-se como valor do peso dos termos o produto da frequência do termo e de seu inverso da frequência de documento [Salton, 1983]. A frequência de termo (tf) é a frequência de ocorrência do termo no documento e normalmente reflete a importância desse termo. O

inverso da frequência do documento (idf) é um fator que realça os termos que aparecem em poucos documentos, enquanto desvaloriza os termos que ocorrem em muitos documentos. O peso dos termos (**Wij)** é, então, dado por:

$$\mathbf{Wij = tf_{ik} \ X \ idf_k,} \qquad \mathbf{(1)}$$

onde tf_{ik} é o número de ocorrências do termo t_k no documento Ti, e idf_k é o inverso da frequência de documento do termo t_k na coleção de documentos. Uma medida comumente usada para idf é $idf_k = \log (N / nK)$, onde N é o número total de documentos na coleção, dos quais nK contêm um termo t_k.

Na representação vetor-espacial, uma medida de similaridade comumente usada é o cosseno do ângulo entre vetores. Isso pode ser calculado através do produto escalar dos 2 vetores.

No sistema Fenix, a similaridade entre um documento e um perfil é uma função da similaridade entre os vetores de termos dos documentos e dos perfis, conforme a equação descrita em [Salton, 1989].

Quando os documentos são apresentados ao usuário, os valores de similaridade (score) são convertidos para uma escala de classificação (tabela 1).

PONTUAÇÃO (*SCORE*) (S)	**CATEGORIA**
S < = 0.1	TERRÍVEL
S > 0.1 e S < 0.2	REGULAR
S > = 0.2 e S < 0.3	NEUTRO
S > = 0.3 e S < 0.5	BOM
S > = 0.5	EXCELENTE

Tabela 1: Conversão das pontuações (*scores*) de similaridade em categorias de classificação.

O valor máximo de score ou similaridade é 1.0, e só ocorre quando a representação do perfil e do documento são idênticas. Nesse trabalho, foi adotado o limite de 0.2 como o valor de score mínimo que um documento deve ter para ser apresentado ao usuário.

4.2.3 Módulo de Aprendizado

Os métodos de aprendizagem adotados pelo sistema são: a realimentação por relevância, e algoritmos genéticos. Ambos os métodos foram projetados como sub-módulos separados e apenas o método de realimentação por relevância foi implementado nesta fase do trabalho.

4.2.3.1 Sub-Módulo de Realimentação por Relevância

A realimentação por relevância é um método de reformulação automática de consultas amplamente usado em sistemas de recuperação de informações [Frakes, 1992], [Rocchio, 1971]. Uma consulta pode ser melhorada iterativamente usando-se um vetor (de termos) de consulta disponível e adicionando termos a partir de documentos relevantes, enquanto se subtraem termos a partir de documentos irrelevantes. Uma única iteração de realimentação por relevância freqüentemente produz melhoras de 40 a 60% na precisão da busca [Salton, 1989].

No sistema proposto, a realimentação dada pelo usuário aos documentos recuperados pelos perfis é usada para alterar a representação vetorial do respectivo perfil.

Para representações vetor-espaciais, o método para reformulação de uma consulta em resposta a realimentação do usuário é ajuste vetorial. Uma vez que consultas e documentos são ambos vetores, o vetor de consulta é movido espacialmente para mais perto de vetores representando documentos que receberam realimentação positiva e para mais longe dos vetores de documentos que receberam realimentação negativa, segundo as fórmulas descritas em [Sheth, 1994].

O efeito resultante da aplicação de método é que, para termos já presentes no perfil, seus pesos são modificados em proporção a realimentação recebida. Os termos que não estejam presentes no perfil são adicionados a este.

4.2.3.2 Sub-Módulo de Algoritmo Genético

No sistema proposto, um agente é modelado como um conjunto de perfis, onde cada perfil busca documentos semelhantes a ele, os quais serão de interesse potencial para o usuário. Para a aplicação do algoritmo genético, considera-se que cada agente corresponde a uma população P, onde cada elemento é um par de perfil e sua aptidão. Cada perfil corresponde a um indivíduo ou cromossomo da população. Os operadores genéticos de *crossover* e mutação atualizam a população a cada geração, introduzindo novos membros e aproveitando aqueles com maior valor de aptidão. O objetivo final é evoluir a população em direção a uma otimização global.

O algoritmo genético compreende as etapas descritas a seguir.

- Representação dos perfis em cromossomos - Os passos para converter um perfil de um agente em sua representação como cromossomo são:
  - toma-se o vetor de termos de cada um dos arquivos de perfil do agente em questão e constrói-se um vetor de termos único para a população;
  - para cada arquivo de perfil, constrói-se um cromossomo (vetor binário) baseado na existência (1) ou não (0) de cada termo do vetor da população no vetor de termos do perfil. O número de genes (bits) do cromossomo é, então, o número total de termos, sem repetição, de todos os vetores de termos dos perfis da população.
- Inicialização da população - A população inicial é composta pela representação binária de um conjunto de documentos que foram julgados relevantes por um usuário através de realimentação.
- Avaliação da Função de Aptidão - A função de aptidão adotada é baseada na medida de similaridade entre o perfil (cromossomo) e os documentos recuperados em uma busca. A aptidão do perfil é a média aritmética dos valores de similaridade de todos os documentos recuperados em uma busca realizada por aquele perfil. Após calcular a aptidão dos perfis, calcula-se a aptidão média para a população, que é a média das aptidões de todos os perfis. O objetivo final a ser alcançado pelo AG é aumentar a aptidão média da população. Assim, a cada iteração do AG calcula-se a aptidão média da população a fim de decidir se novas iterações serão realizadas ou não. Caso os valores de aptidão diminuam ou estabilizem, considera-se que o AG convergiu.
- Reprodução e Recombinação - Devido ao pequeno tamanho da população, todos os cromossomos são usados para a reprodução, não sendo necessária nenhuma função de seleção dos indivíduos. O usuário deve estar ciente de que, para aplicar o módulo de AG, o

agente deverá ter um número mínimo de perfis, e esse número deverá ser par, a fim de permitir as operações de recombinação. Assumiu-se como 10 esse número mínimo. Inicialmente, os cromossomos da população são ordenados por sua aptidão média, a fim de se fazer a recombinação entre pares de indivíduos com aptidões mais semelhantes. Cada par é, então, submetido aos operadores genéticos de *crossover* e, eventualmente, de mutação.

Após a reprodução, *crossover*, e mutação, a nova população está pronta para sua próxima geração. O restante das evoluções são simplesmente repetições cíclicas dos passos acima até o sistema alcançar um número pré-determinado de gerações (fixado pelo usuário) ou convergir (isto é, não apresentar melhora na aptidão média da população).

Embora não tenham sido implementadas, as classes do algoritmo genético foram especificadas. A implementação e testes do sub-módulo serão realizadas posteriormente.

4.2.4 Módulo de Busca

Este módulo é responsável por obter informações de páginas *Web* sobre o assunto escolhido e salvá-las no banco de dados local. Neste trabalho optou-se por usar mecanismos de busca já existentes, como o Altavista[2], sendo o módulo de busca responsável por fazer a interface com esses mecanismos.

O sistema mantém uma lista de mecanismos de busca como parte de seus parâmetros de configuração. As palavras-chaves escolhidas pelo usuário são fornecidas para esses mecanismos e a primeira página de resultados recuperada por cada um deles (em geral é a página com os links mais relevantes) é processada pelos agentes do módulo de busca. O processamento envolve converter a página para a forma de string, analisar seu conteúdo identificando os links significativos e extrair o conteúdo textual das páginas apontadas pelos links, armazenando-o no banco de dados. Cada usuário possui um agente de busca para cada um de seus perfis de preferências.

4.2.5 Outros Módulos

O banco de dados do sistema é composto por todas as informações dos usuário, dos seus agentes e respectivos perfis, bem como pelos documentos recuperados nas buscas.

Esse banco é implementado através de um conjunto de classes que incorporam a persistência do sistema. Neste trabalho, diferentes tipos de arquivos foram usados, sendo aproveitadas as classes de manipulação de arquivos já existentes nas APIs padrão de Java.

Além dos módulos descritos, o sistema Fenix possui um módulo controlador, responsável por criar todas as classes do sistema e invocar seus métodos, controlando a comunicação entre os demais módulos. Este módulo controla o comportamento global do sistema, o conjunto de agentes de cada usuário e o comportamento específico de cada agente.

5 Análise dos Resultados

Neste trabalho adotou-se a medida de performance proposta por Yao [1995]. A medida ndpm (*normalized distance-based performance measure*) é uma distância, normalizada para estar entre 0 e 1, entre a classificação dada pelo usuário a um conjunto de documentos e a

[2] Disponível em: http://www.altavista.digital.com

classificação dada pelo sistema aos mesmos documentos.

Para a realização dos testes, adotou-se o esquema proposto por Balabanovic [1998], onde se tenta evitar que a classificação do usuário seja tendenciosa. A intervalos regulares é fornecida aos usuários uma lista especial de documentos obtidos na *Web*, a fim de que os usuários os classifiquem de acordo com seus interesses em algum assunto. A classificação consiste em cinco categorias: Excelente - Bom - Neutro - Pobre - Terrível. Essas classificações dos usuários formam a classificação desejada e não são usadas para o aprendizado do sistema, apenas para testar seu desempenho.

Os agentes de filtragem calculam a similaridade dos documentos avaliados pelo usuário em relação ao perfil daquele usuário. As pontuações calculadas pelo sistema são convertidas para cada categoria de classificação de acordo com a faixa de valores da tabela 1. As classificações obtidas formam as classificações previstas. Tendo-se os valores de classificação prevista e desejada, pode-se calcular a distância ndpm entre elas, segundo as fórmulas propostas em [Balabanovic, 1998] e, assim, construir a curva de aprendizado do sistema.

O resultado desejado nos testes é que a distância ndpm diminua gradualmente com o tempo, conforme o perfil do usuário vai sendo ajustado.

Cento e vinte agentes foram criados para controlar trinta assuntos de interesse do usuário. Para cada agente, foi executado um certo número de sessões de interação, variando de 6 (seis) a 22 (vinte e dois). Vários parâmetros de configuração do sistema foram testados nas sessões. Para sumarizar os resultados obtidos pode-se dizer que o sistema obteve o melhor desempenho quando:

- os termos das consultas eram mais específicos, em contraste com consultas genéricas;
- os agentes eram compostos por no mínimo 4 (quatro) e no máximo 10 (dez) perfis; e
- os vetores de termos dos documentos tinham tamanho mínimo de 300 termos.

A figura 2 apresenta a distância ndpm média calculada para todos os agentes ao longo das 6 (seis) primeiras sessões de avaliação (número mínimo de sessões adotado para todos os agentes). Para todos os trinta assuntos, observou-se uma diminuição progressiva dessa distância ao longo das sessões. Essa diminuição indica que os agentes foram se adaptando às preferências do "usuário", aumentando, assim, a probabilidade de recuperar um maior número de documentos relevantes e de descartar os irrelevantes.

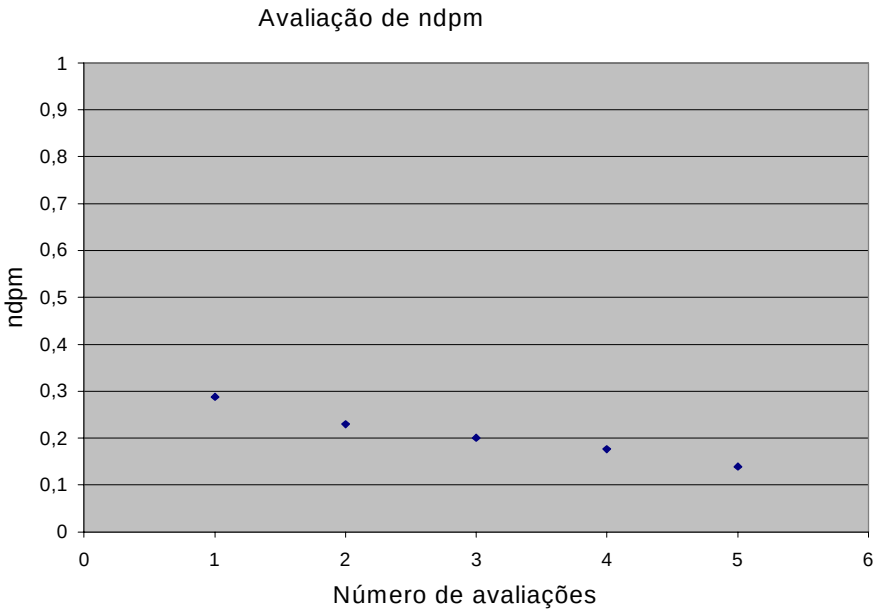

Figura 2: Distância ndpm média para todos os agentes ao longo de 6 (seis) sessões de avaliação

Para fins de comparação com o trabalho descrito em [Balabanovic, 1998], 9 (nove) agentes foram testados ao longo de um número de sessões superior a 20 (vinte). Balabanovic propôs uma arquitetura híbrida em que combinava filtragem baseada em conteúdo com filtragem colaborativa, para a recomendação de páginas *Web*. O desempenho de seu sistema foi avaliado com a medida ndpm ao longo de vinte e cinco sessões e a curva de aprendizado teve comportamento bastante similar à obtida nos testes do sistema Fenix (figura 3).

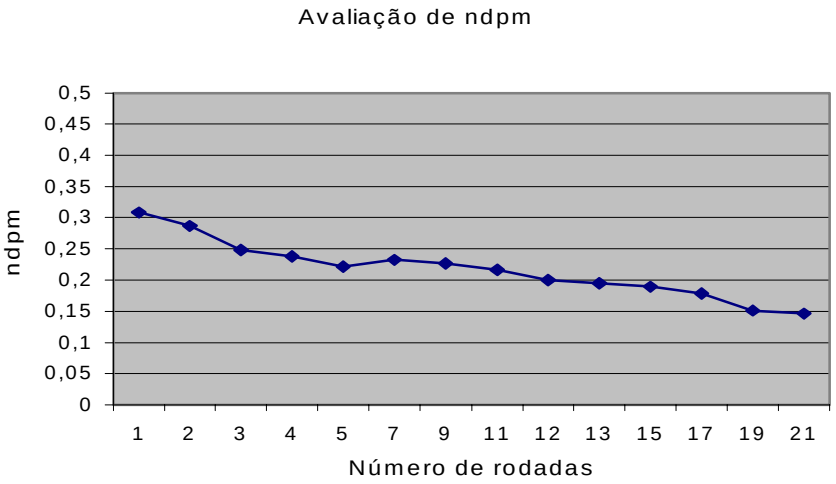

Figura 3: Distância ndpm média para 9 (nove) agentes usados para comparação.

6 Conclusões

O sistema Fenix é um Sistema de Filtragem de Informação que deve ser capaz de especializar-se conforme os interesses do usuário, adaptar-se às mudanças e explorar o domínio de informações potencialmente relevantes.

Este sistema demonstrou ser uma poderosa ferramenta de filtragem de informações. Os resultados experimentais apresentados confirmaram que o sistema, empregando o modelo vetor espacial com aprendizado baseado em realimentação por relevância pode filtrar com sucesso documentos de interesse para usuários com preferências bem definidas.

Uma contribuição significativa do presente trabalho consistiu na proposição de um modelo de representação de consultas e documentos baseado em diversos trabalhos obtidos na literatura de recuperação de informações, e adaptado para uso com páginas WWW.

Os valores de performance obtidos em testes simulados baseados na distância ndpm foram similares aos encontrados em outros trabalhos de filtragem de informações. Balabanovic [1998] propôs uma abordagem para a recomendação de páginas *Web* que combinava o método de filtragem baseado em conteúdo usando realimentação por relevância, com o método de filtragem colaborativa, no qual as recomendações de outros usuários afetam a resposta do sistema. Os resultados descritos em seu trabalho foram bastante semelhantes aos obtidos com o Fenix, onde foi adotada uma abordagem bem mais simples de implementar.

O impacto de sistemas como o Fenix em uma empresa poderá ser altamente significativo, especialmente para negócios que lidem com um domínio de informações predominantemente textuais, como os ligados a assuntos jurídicos ou científicos. Muitas das tarefas de busca de informações poderiam ser automatizadas, pois o sistema é capaz de filtrar e priorizar diversos assuntos interessantes, reduzindo, assim o consumo de tempo gasto nessas atividades. Agentes de Filtragem de Informação são uma grande promessa para o gerenciamento das inúmeras informações disponíveis.

7 Referências Bibliográficas

[Allan, 1995] Allan, J. 1995. Relevance feedback with too much data. In Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.

[Armstrong, 1995] Armstrong, R., *et al*. "*Web*Watcher: A learning apprentice for the World Wide *Web*". In anais do congresso **AAAI Spring Symposium on Information Gathering from Heterogeneous, Distributed Resources**. Stanford, March 1995.

[Balabanovic, 1997] Balabanovic , M. An Adaptive *Web* Page Recomendation Service. Stanford Universal Digital Libraries Project Working Papers SIDL - WP. 1997.

[Balabanovic, 1998] Balabanovic , M. Learning to Surf: Mutiagent Systems for Adaptive *Web* Page Recommendation. Dissertação de Doutorado submetida ao Departamento de Ciências da Computação da Universidade de Stanford. Março, 1998.

[Booker, 1990] Booker, L.B., Goldberg, D.E. and Holland, J.H. Classifier Systems and Genetic Algorithms. In Machine Learning, Paradigms and Methods, Pages 235-282,

Carbonell, J.G. Editor, The MIT Press, Cambridge, MA, 1990.

[Brusilovsky, 1994] Brusilovsky, P. Adaptive Hypermedia: an attempt to analise and generalize.1994.Disponível em: http://www.cs.bgsu.edu/hypertext.adaptive/um94.html

[Buckley, 1994] Buckley, C.; Salton, G.; Allan, J; and Singhal, A.. "Automatic query expansion using SMART: TREC-3". In anais do congresso **3rd Text Retrieval Conference**. Gaithersburg, Maryland, USA. November, 1994.

[Burbeck, 1987] Burbeck, Steve, 1987 - Application Programming in Smalltalk-80 (TM) : How to use Model-View-Controller (MVC). Disponível em: http://burks.bton.ac.uk/ burks/ language/smaltalk/mvc.htm

[Fischer, 1991] Fischer, G., Stevens, C., Information access in complex, poorly structured information spaces. Human Factors in Computing Systems CHI'91 Conference Proceedings, 1991, pp. 63-70.

[Foltz , 1992] Foltz, P. W., Dumais, S. T. "Personalized Information Delivery: An Analysis of information filtering methods". **Communications of ACM.** v. 35 (12), pp.29-38, December 1992.

[Frakes, 1992] Frakes, William B. & Baeza-Yates, R. – Information Retrieval – Data Structures & Algorithms. Prentice Hall PTR, New Jersey, 1992.

[Goldberg, 1989] Goldberg, D. E. Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, Reading, MA, 1989.

[Goldberg, 1994] Goldberg, D. E. Genetic and Evolutionary Algorithms come of age. Communications of the ACM, 37(3):113-119, March 1994.

[Harman, 1994]. "Overview of the Third Text Retrieval Conference (TREC-3)". In anais do congresso 3rd Text Retrieval Conference. Gaithersburg, Maryland, USA. November, 1994.

[Lang, 1995] Lang, K. 1995. NewsWeeder: Learning to filter netnews. In Proceedings of the 12th International Conference on Machine Learning.

[Nissen, 1995] Nissen, M. et al.. Intelligent Agents: a Technology and Business Aplication Analysis, BA248D: Telecomunications and Distributed Processing, Intelligencia, Inc, Novembro, 1995.

[Pazzani, 1996] Pazzani, M; Muramatsu, J.; and Billsus, D. 1996. Syskill & *Web*ert: Identifying interesting *Web* sites. In Proceedings of the 13th National Conference on Artificial Intelligence.

[Rocchio, 1971] Rocchio, J.J. Relevance feedback in information retrieval**.** In: The Smart Retrieval System - Experiments in automatic Document Processing, p. 313-323, Englewood Cliffs: Prentice-Hall, 1971.

[Salton, 1983] Salton, G., McGill, M. J., Introduction to Modern Information Retrieval, McGraw-Hill, 1983.

[Salton, 1989] Salton, G., Automatic Text Processing – The Transformation, Analysis and Retrieval of Information by Computer. Addison-Wesley Publishing Company, Inc., Reading, MA, 1989.

[Schutze, 1995] Schutze, H.; Hull, D. A.; and Pedersen, J. O. 1995. A comparison of classifiers and documents representations for the routing problem. In Proceedings of the 18 th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.

[Sheth, 1994] Sheth, B. NEWT: A learning approach to personalized information filtering. Dissertação. MIT Department of Electrical Engineering and Computer Science.

s.l.:1994. Disponível na Internet via WWW. URL: http://www.cs.bham.ac.uk/~sra/People/Stu/Sheth/index.html#ALearning. Arquivo consultado em 2000.

[Sheth, 1993] Sheth, B., and Maes , P. 1993. Evolving agents for personalized information filtering In Proceedings of the 9th IEEE Conference on Artificial Intelligence for Applications.

[Twidale, 1995] Twidale, M. B., Nichols, D. M. and Paice, C. D.. Browsing is a Collaborative Process.Tecnical Report - CSEG/1/96-Computing Department, Lancaster University, 1996. Disponível em: http:// www.comp.lancs.ac.uk/computing /research/ cseg/ projects/ariadne/docs/bcp.html.

[Yao, 1995] Yao, Y. Y. 1995. Measuring retrieval effectiveness based on user preference of documents. Journal of the American Society for Information Science 46(2):133-145.