

Hélio Martins do Nascimento Júnior

*Sistema de Recomendação Híbrido para  
Bibliotecas Digitais que Suportam o  
Protocolo OAI-PMH*

Maceió – AL

Dezembro / 2008

Hélio Martins do Nascimento Júnior

*Sistema de Recomendação Híbrido para  
Bibliotecas Digitais que Suportam o  
Protocolo OAI-PMH*

Dissertação apresentada à Coordenação do Mestrado em Modelagem Computacional de Conhecimento da Universidade Federal de Alagoas para a obtenção do título de Mestre em Modelagem Computacional de Conhecimento.

Orientador:

Prof. Dr. Evandro de Barros Costa

MESTRADO EM MODELAGEM COMPUTACIONAL DE CONHECIMENTO  
INSTITUTO DE COMPUTAÇÃO  
UNIVERSIDADE FEDERAL DE ALAGOAS

Maceió – AL

Dezembro / 2008

**Catálogo na fonte**  
**Universidade Federal de Alagoas**  
**Biblioteca Central**  
**Divisão de Tratamento Técnico**  
**Bibliotecária Responsável: Helena Cristina Pimentel do Vale**

N244s Nascimento Júnior, Hélio Martins do.  
Sistema de recomendação híbrido para bibliotecas digitais que suportam o protocolo OAI-PMH / Hélio Martins do Nascimento Júnior. – Maceió, 2008.  
105 f. : il.

Orientador: Evandro de Barros Costa.  
Dissertação (mestrado em Modelagem Computacional de Conhecimento) – Universidade Federal de Alagoas. Instituto de Computação. Maceió, 2008.

Bibliografia: f. 99-103.

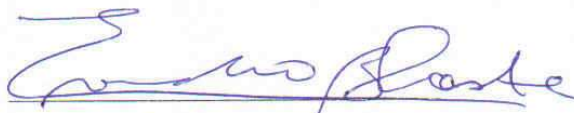
Anexos: f. 104-105.

1. Bibliotecas digitais – Sistemas de recomendação. 2. Bibliotecas digitais – Perfil do usuário. 3. Filtragem de informação. 4. Metadados. 5. Inteligência artificial. 6. Modelagem computacional. I. Título.

CDU: 004.8:027.021

---

Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre em Modelagem Computacional de Conhecimento pelo Programa Multidisciplinar de Pós-Graduação em Modelagem Computacional de Conhecimento, da Universidade Federal de Alagoas, aprovada pela comissão examinadora que abaixo assina:



**Prof. Dr. Evandro de Barros Costa**

UFAL – Instituto de Computação

Orientador



**Prof. Dr. Henrique Pacca Loureiro Luna**

UFAL – Instituto de Computação

Examinador



**Prof. Dr. Guilherme Ataíde Dias**

UFPB – Departamento de Ciência da Informação

Examinador

Maceió, dezembro de 2008.

*Dedico esta dissertação a meus pais,  
cujo exemplo de honestidade e trabalho  
tem sido um norteador para a minha vida,  
e para minha esposa, que tem  
me dado apoio nos momentos mais difíceis  
e mostrado a simplicidade de ter esperança.*

# *Agradecimentos*

Agradeço à Deus, pela vida, pelas oportunidades e por estar sempre ao meu lado me dando forças para sempre seguir em frente, apesar dos tropeços.

Agradeço a minha família por esta comigo nas horas mais difíceis.

Agradeço aos meus pais que sempre se esforçaram o máximo para me fornecer uma boa educação.

Agradeço aos amigos que me apoiaram nesse desafio de conseguir o título de mestre.

Agradeço ao meu orientador Evandro de Barros Costa, pela ajuda, oportunidades concedidas e por ter me ensinado tantas coisas, principalmente, ser simples e otimista.

Agradeço ao Colegiado do mestrado em Modelagem Computacional de Conhecimento.

Agradeço aos meus amigos da Unidade de Palmeira dos Índios, particularmente ao Hécio, Assis, Adalberto, Carlos Guedes, Israel, Émerson e Alan.

Agradeço a professora Ana Cristina Quixabera que me incentivou muito durante todo o processo.

Agradeço ao pessoal do Laboratório de Inteligência Artificial e do projeto Arco.

Agradeço aos professores Silvio Chagas e Willy Carvalho Tiengo que contribuíram bastante na reta final do trabalho.

*“Os homens perdem a saúde para juntar dinheiro, e depois perdem o dinheiro para a recuperar. Por pensarem ansiosamente no futuro, esquecem o presente, de tal forma que acabam por nem viver no presente nem no futuro.*

*Vivem como se nunca fossem morrer e morrem como se não tivessem vivido... ”*

***Confúcio***

# *Resumo*

O crescimento acelerado das tecnologias Web tem beneficiado pesquisadores e acadêmicos, pois as publicações de pesquisa podem ser acessadas eletronicamente tão logo elas tenham sido finalizadas e publicadas. Nesse contexto, surgem as Bibliotecas Digitais como um sistema de informação complexo que possui uma série de atividades que integram coleções, serviços e pessoas em suporte ao completo ciclo de criação, disseminação, acesso e preservação de dados, informação e conhecimento. No entanto, devido a enorme quantidade de conteúdo presente na Web, em particular nas Bibliotecas Digitais, usuários acabam se deparando com uma diversidade muito grande de opções, o que leva ao fenômeno conhecido como sobrecarga de informação. Com o objetivo de contribuir para amenizar ou até mesmo eliminar essas dificuldades, sistemas de recomendação para Bibliotecas Digitais têm sido propostos e desenvolvidos. Este trabalho segue essa direção, investigando soluções alternativas para alcançar mais qualidade nas indicações geradas por um sistema de recomendação na sua tarefa de ajudar os seus usuários. Para isso estudou-se as abordagens tratadas na literatura especializada sobre tais sistemas, propondo-se em seguida, um sistema de recomendação personalizada de artigos científicos para Bibliotecas Digitais. Tal sistema seguiu uma abordagem híbrida, procurando tirar proveito das características interessantes identificadas nas técnicas de filtragem e recomendação baseadas em conteúdo e colaborativa. Nesse sentido desenvolveu-se um engenho de recomendação híbrido que se utiliza de tecnologias padrão para a descrição de conteúdo (*Padrão Dublin Core*), comunicação com Bibliotecas Digitais (*Protocolo OAI-PMH*) e perfil do pesquisador (Currículo Lattes). Finalmente, avaliou-se o sistema proposto sobre uma base de dados do CiteSeer contendo artigos no formato *Dublin Core*, tendo os resultados preliminares mostrado-se satisfatórios melhorando a precisão na recomendação e a cobertura quando comparado com sistemas que implementam abordagens baseada em conteúdo e colaborativa isoladamente.

**Palavras-Chave:** Bibliotecas digitais - Sistemas de recomendação, Bibliotecas digitais - Perfil do usuário, Filtragem de Informação, Metadados, Inteligência Artificial, Modelagem computacional.



# *Abstract*

The growth of Web technologies has benefited researchers and the academic community by supporting the access of electronic publications as soon as they have been finished and published. In this context, Digital Libraries emerges as complex information systems which are essential for disseminating and preserving data, information and knowledge. However, due to the high amount of content available on the Web, specially in Digital Libraries, users face many correlated options, what result in the phenomenon known as information overload. Aiming to decrease or even eliminate these difficulties, recommender systems for Digital Libraries have been proposed and developed. This work presents a personalized recommender system which presents alternative ways to achieve better query results. For this, the main existing approaches of automatic recommendation have been studied in order to identify extension points and points to be improved. The proposed recommender system follows a hybrid approach which combines filtering techniques, content-based recommendation and collaborative recommendation. A hybrid recommendation engine has been proposed, which uses standard technologies for content description (*Dublin Core*), for communication with Digital Libraries (*OAI-PMH Protocol*), as well as the user profile extracted from the curriculum vitae Lattes. The proposed solution has been evaluated in the context of the CiteSeer database, which contains papers and articles in the *Dublin Core* format. The preliminary results has showed an improvement in the quality of recommendation, thus presenting a better precision and coverage, when compared with existing approaches based either on content-based recommendation or on collaborative recommendation.

**Keywords:** Digital Libraries - Recommender Systems, Digital Libraries - User Profile , Information Filtering, Metadata, Artificial Intelligence e Modeling Computational.

# *Sumário*

**Lista de Figuras**

**Lista de Tabelas**

**Lista de Siglas**

<b>1</b>	<b>Introdução</b>	p. 17
1.1	Contextualização e Problemática . . . . .	p. 17
1.2	Objetivo . . . . .	p. 18
1.3	Relevância . . . . .	p. 19
1.4	Organização . . . . .	p. 19
<b>2</b>	<b>Bibliotecas Digitais</b>	p. 21
2.1	Introdução . . . . .	p. 22
2.2	Infra-estrutura Tecnológica . . . . .	p. 24
2.3	Metadados . . . . .	p. 26
2.3.1	MARC . . . . .	p. 27
2.3.2	METS(CONGRESS., 2006) . . . . .	p. 29
2.3.3	Dublin Core . . . . .	p. 30
2.4	Interoperabilidade . . . . .	p. 32
2.4.1	<i>Open Archives Initiative - Protocol for Metadata Harvesting</i> (OAI- PMH) . . . . .	p. 32
<b>3</b>	<b>Sistemas de Recomendação</b>	p. 36

3.1	Sistemas de Recomendação, filtragem e recuperação de informação . . .	p. 37
3.2	Formalização do Problema (ADOMAVICIUS; TUZHILIN, 2005) . . . . .	p. 39
3.3	Coleta de Informações . . . . .	p. 41
3.4	Filtragem Baseada em Conteúdo ( <i>Content-based Filtering</i> ) . . . . .	p. 42
3.5	Filtragem Colaborativa ( <i>Collaborative Filtering</i> ) . . . . .	p. 43
3.6	Filtragem Híbrida ( <i>Hybrid Filtering</i> ) . . . . .	p. 45
3.7	Pesquisa em Sistemas de Recomendação . . . . .	p. 47
<b>4</b>	<b>Trabalhos Relacionados</b>	p. 50
<b>5</b>	<b>Sistema de Recomendação Proposto</b>	p. 54
5.1	Descrição do Sistema . . . . .	p. 55
5.2	Arquitetura . . . . .	p. 55
5.3	O módulo Lattes . . . . .	p. 57
5.4	Processo de Obtenção dos Metadados dos Documentos . . . . .	p. 60
5.5	O módulo de Avaliação . . . . .	p. 63
5.6	O módulo de Recomendação . . . . .	p. 63
5.6.1	Filtragem Colaborativa - FC . . . . .	p. 63
5.6.2	Filtragem Baseada em Conteúdo - FBC . . . . .	p. 64
5.6.3	Construção do Perfil do Usuário . . . . .	p. 66
5.6.4	Recomendação Híbrida . . . . .	p. 68
<b>6</b>	<b>Implementação</b>	p. 71
6.1	Introdução . . . . .	p. 72
6.2	Módulo Lattes . . . . .	p. 72
6.3	Módulo de Coleta OAI . . . . .	p. 74
6.4	Módulo de Recomendação utilizando Filtragem Colaborativa . . . . .	p. 77
6.5	Módulo de Recomendação Baseada em Conteúdo . . . . .	p. 78

6.6	Módulos de Recomendação Híbrido . . . . .	p. 82
<b>7</b>	<b>Experimentos e Resultados</b>	p. 89
7.1	Dados . . . . .	p. 90
7.2	Método Experimental . . . . .	p. 91
7.3	Métricas . . . . .	p. 92
7.4	Teste . . . . .	p. 93
7.5	Apresentação dos Resultados . . . . .	p. 93
7.5.1	Resultados da Filtragem Colaborativa . . . . .	p. 93
7.5.2	Resultados da Filtragem Baseada em Conteúdo . . . . .	p. 94
7.5.3	Resultados da Filtragem Híbrida utilizando o Algoritmo Misto . . . . .	p. 96
7.5.4	Resultados da Filtragem Híbrida utilizando o Algoritmo Ponderado . . . . .	p. 96
7.6	Avaliação dos Resultados . . . . .	p. 97
<b>8</b>	<b>Conclusões</b>	p. 98
	Trabalhos Futuros . . . . .	p. 99
	<b>Referências</b>	p. 100
	<b>Anexo</b>	p. 105
	Recuperação de Informação Utilizando o Modelo de Espaço Vetorial (VSM)	p. 105

# *Lista de Figuras*

1	Exemplo de registro Marc21 . . . . .	p. 28
2	Interpretação do registro Marc21 . . . . .	p. 29
3	Interação entre as entidades básicas do OAI-PMH . . . . .	p. 34
4	Processo de Recomendação utilizando Filtragem Colaborativa (imagem obtida de (QUEIROZ, 2002)) . . . . .	p. 44
5	Características herdadas pela Filtragem Híbrida (imagem obtida de (RE-ATEGUI; CAZELLA, 2005)) . . . . .	p. 46
6	Arquitetura do Sistema de Recomendação Proposto . . . . .	p. 56
7	Algoritmo Misto . . . . .	p. 68
8	Algoritmo Ponderado . . . . .	p. 69
9	Exemplo de Cálculo Realizado pelo Algoritmo Ponderado . . . . .	p. 70
10	Arquitetura do Módulo Lattes . . . . .	p. 73
11	Arquitetura do Módulo de Coleta OAI . . . . .	p. 75
12	Implementação do Algoritmo Misto. Figura adaptada de (BURKE, 2007)	p. 85
13	Implementação do Algoritmo Ponderado. Figura adaptada de (BURKE, 2007) . . . . .	p. 88
14	Artigos por Usuário . . . . .	p. 91
15	Avaliações por Usuário . . . . .	p. 91
16	Influência do Número de Vizinhos . . . . .	p. 94
17	Filtragem Baseada em Conteúdo . . . . .	p. 95
18	Comparação do resultado das técnicas . . . . .	p. 97

# *Lista de Tabelas*

1	Campos básicos do Formato MARC21 . . . . .	p. 28
2	Conjunto de elementos do formato Dublin Core . . . . .	p. 31
3	Conjunto de elementos adicionais do formato Dublin Core Qualificado . . . . .	p. 32
4	Verbos OAI-PMH . . . . .	p. 34
5	Classificação de Sistemas Híbridos . . . . .	p. 47
6	Pesquisa em Recomendação Baseada em Conteúdo . . . . .	p. 48
7	Pesquisa em Recomendação Colaborativa . . . . .	p. 48
8	Pesquisa em Recomendação Híbrida . . . . .	p. 49
9	URLs Base de alguns provedores de dados . . . . .	p. 61
10	Atributos do currículo lattes . . . . .	p. 67
11	Perfil inicial do usuário . . . . .	p. 67
12	Perfil modificado pelo usuário . . . . .	p. 67
13	Similaridade usando Pearson entre o usuário1 e todos os usuários . . . . .	p. 93
14	Top-5 artigos recomendados para o usuário1 utilizando filtragem colaborativa com 1 vizinho . . . . .	p. 94
15	Termos e pesos do perfil do usuário1 . . . . .	p. 95
16	Top-5 artigos recomendados para o usuário1 utilizando filtragem baseada em conteúdo . . . . .	p. 95
17	Top-5 artigos recomendados para o usuário1 utilizando filtragem híbrida mista . . . . .	p. 96
18	Top-5 artigos recomendados para o usuário1 utilizando filtragem híbrida ponderada com $p=0,6$ . . . . .	p. 96

19	Top-5 artigos recomendados para o usuário1 utilizando filtragem híbrida ponderada com $p=0,01$ . . . . .	p.97
----	--	------

# *Lista de Siglas*

ARL	Association of Research Libraries
DLI	Digital Library Initiative
OAI	Open Archives Initiative
OAI-PMH	Open Archives Initiative Protocol for Metadata Harvesting
IFLA	International Federation of Library Associations
W3C	World Wide Web Consortium
MARC	MAchine-Readable Cataloging
METS	Metadata Encoding and Transmission Standard
XML	Extensible Markup Language
DC	Dublin Core
PDF	Portable Document Format
RI	Recuperação de Informação
FC	Filtragem Colaborativa
FBC	Filtragem Baseada em Conteúdo
TF-IDF	Term-Frequency Inverse-Document-Frequency
SR	Sistema de Recomendação
VSM	Vector Space Model
EAD	Educação a Distância
RDF	Resource Description Framework



# *Lista de Códigos*

5.1	Exemplo de um arquivo XML do Currículo Lattes . . . . .	p. 57
5.2	Ontologia XML para a Unidade de Informação de Currículos . . . . .	p. 59
5.3	Exemplo de um arquivo XML no formato Dublin Core . . . . .	p. 61
6.1	Trecho de Código do Parse XML . . . . .	p. 73
6.2	Trecho de Código do Parser XML dos Metadados . . . . .	p. 75
6.3	Trecho de Código da Filtragem Colaborativa . . . . .	p. 77
6.4	Trecho de Código do Indexador . . . . .	p. 79
6.5	Trecho de Código do Vetor de Consulta . . . . .	p. 80
6.6	Trecho de Código do Processo de Busca . . . . .	p. 82
6.7	Trecho de Código do Processo de Recomendação Híbrida utilizando o Algoritmo Misto . . . . .	p. 83
6.8	Trecho de Código do Processo de Recomendação Híbrida utilizando o Algoritmo Ponderado . . . . .	p. 85

# 1 *Introdução*

Frequentemente, as pessoas necessitam de informações para realizar suas atividades, sejam estas atendendo por exemplo, suas necessidades pessoais ou profissionais. Entretanto, encontrar rapidamente a informação correta e de interesse está se tornando cada vez mais difícil devido a enorme quantidade de informação presente em meios impressos ou eletrônicos.

Considerada a maior fonte de informações do mundo, a Web se tornou um local importante, contendo muito conteúdo e facilitando a troca de experiências entre seus usuários, pois, por exemplo, resultados de pesquisas podem ser observados eletronicamente assim que os mesmos são publicados na Web. Nesse contexto, tem surgido as Bibliotecas Digitais que são coleções de documentos com serviços associados, tendo-se os sistemas de recomendação como um serviço desenvolvido para facilitar o acesso aos seus recursos.

## 1.1 *Contextualização e Problemática*

Com a enorme quantidade de conteúdo presente na Web, especialmente em Bibliotecas Digitais, usuários acabam se deparando com uma diversidade muito grande de opções, o que leva ao fenômeno conhecido como sobrecarga de informação. Nesse caso, o grande problema é, portanto, como encontrar informações relevantes diante de tamanha diversidade e grande volume de conteúdo. Sistemas de busca, como o *google*, podem ajudar bastante nesta tarefa. Mas um problema prático dos sistemas de busca está no fato de que o usuário precisa fornecer uma *query* de busca algumas vezes complexa ou mesmo requerer para ter sucesso do uso de termos específicos de um determinado domínio para auxiliar o processo de recuperação da informação. Além disso, é muito comum obter resultados de buscas totalmente insatisfatórios devido a enorme quantidade de informação recuperada ou até mesmo resultados que não possuem relação nenhuma com o que foi pesquisado, o que pode se tornar um fator complicador para usuários com pouca ou quase nenhuma experiência realizar escolhas diante de tamanha variedade de opções (MONTANER; LÓPEZ;

ROSA, 2003).

Uma solução para amenizar os problemas mencionados, particularmente o problema de sobrecarga de informação, é oferecer um sistema de recomendação que possa ajudar pessoas a filtrar informações úteis baseando-se principalmente em seus interesses individuais. O desenvolvimento de um sistema de recomendação, normalmente, envolve problemas fundamentais localizados principalmente nas áreas de Recuperação e filtragem de informação, Mineração de dados/textos, Modelagem do usuário e Inteligência Artificial.

A motivação desse trabalho é minimizar o problema da sobrecarga de informação em bibliotecas digitais, para este fim, um sistema de recomendação para bibliotecas digitais que interoperem utilizando o protocolo OAI-PMH foi desenvolvido.

## 1.2 Objetivo

O objetivo geral desse trabalho é a elaboração de um sistema de recomendação personalizada de artigos científicos para bibliotecas digitais, combinando técnicas de filtragem de informação, através de uma abordagem híbrida. Nesse contexto, se utilizando de duas técnicas: uma técnica baseada em conteúdo e uma técnica colaborativa. Considera-se aqui qualquer biblioteca digital que provê metadados no formato Dublin Core (DC) e dá suporte ao protocolo OAI-PMH, pode ser utilizada como fonte para prover informações sobre os artigos a serem recomendados.

A realização deste objetivo geral conduziu à concretização dos seguintes objetivos específicos:

- Definir um perfil de usuário, considerando-se atributos do currículo Lattes relevantes para a formação do perfil do usuário;
- Desenvolver de um mecanismo que permita ao usuário contribuir para refinar o seu perfil, de forma que esse possa refletir melhor os seus interesses atuais;
- Desenvolver uma arquitetura para um sistema de recomendação híbrido que utilize o perfil, metadados e avaliações dos artigos;
- Experimentar a solução proposta num conjunto de metadados extraídos de uma biblioteca digital, avaliando-se os resultados obtidos.

## 1.3 Relevância

Sistemas de Recomendação são interessantes pois, assistem os usuários de um modo personalizado num grande conjunto de opções possíveis a encontrarem os melhores itens que combinem com suas necessidades e preferências. As pesquisas nesta área podem ser localizadas em uma grande variedade de domínios, tais como: Bibliotecas Digitais (CALLAN et al., 2003), Comunidades Virtuais (PAPAGELIS; PLEXOUSAKIS, 2003), E-learning (TANG; MCCALLA, 2003) e Sistemas Multiagentes (CAZELLA; ALVARES, 2006) (KAELBLING; SAFFIOTTI, 2005), dentre muitos outros.

A relevância deste trabalho está em buscar um "modelo" que possa oferecer mais efetividade na aplicação de técnicas de recomendação híbrida para recomendar artigos científicos numa biblioteca digital que utilize o protocolo OAI-PMH. Esse utiliza-se de tecnologias padrão para a criação do perfil do usuário (currículo Lattes), descrição dos documentos (formato *Dublin Core*) e comunicação entre Bibliotecas Digitais (protocolo OAI-PMH). Isso é importante porque garante que o sistema possa interoperar com qualquer Biblioteca Digital que faça uso de tais tecnologias. Adicionalmente, apresenta-se uma arquitetura de uma sistema de recomendação híbrido que utiliza as tecnologias relacionadas anteriormene.

- Amenizar o problema da sobrecarga de informação enfrentada pelos usuários de Bibliotecas Digitais, os quais continuam em busca de ferramentas mais efetivas.
- Utilizar tecnologias padrão para a criação do perfil do usuário (currículo Lattes), descrição dos documentos (formato Dublin Core) e comunicação entre Bibliotecas Digitais (protocolo OAI-PMH). Isso é importante porque garante que o sistema possa interoperar com qualquer Biblioteca Digital que faça uso de tais tecnologias.

## 1.4 Organização

Este trabalho está organizado da seguinte forma:

- No **Capítulo 2**, são abordados os principais conceitos relacionados a Bibliotecas Digitais, enfatizando principalmente suas características e principais tecnologias utilizadas no desenvolvimento das mesmas.
- No **Capítulo 3**, é realizada uma discussão sobre sistemas de recomendação, filtragem e recuperação de informação, destacando-se o problema da recomendação, suas

motivações e abordagens, além de alguns exemplos de sistemas de recomendação para Bibliotecas Digitais.

- No **Capítulo 4**, são apresentados os trabalhos relacionados.
- No **Capítulo 5**, são apresentados o sistema proposto nesse trabalho e um exemplo ilustrando alguns dos passos do processo de recomendação realizado pelo sistema.
- No **Capítulo 6**, é apresentada a implementação de cada módulo do sistema.
- No **Capítulo 7**, são apresentados os experimentos e resultados.
- No **Capítulo 8**, são mostradas as considerações finais do trabalho, discutindo os resultados alcançados e possíveis contribuições, além das principais limitações e propostas de trabalhos futuros.

## *2 Bibliotecas Digitais*

*“A máquina tecnologicamente mais eficiente que um homem jamais inventou é o livro.”*

Northrop Frye

Neste capítulo são apresentados os principais conceitos relacionados ao tema biblioteca digital.

## 2.1 Introdução

Uma biblioteca tem como função básica tornar o acesso à informação mais disseminado, rápido e eficiente. Na década de 80, a informação armazenada estava contida basicamente em papel: livros, manuais, revistas e seu acesso era excessivamente controlado. Conforme a tecnologia evoluiu, o acesso a informação diversificou-se e as bibliotecas deixaram de trabalhar apenas com livros e passou a receber vários outros tipos de registros informacionais como fotografias, conteúdos multimídias: áudio e/ou vídeo, programas de computador, etc.

O surgimento das tecnologias da informação e da comunicação, principalmente, com o surgimento da Internet possibilitou o registro, publicação e disseminação de um volume cada vez mais crescente de informação digital, em suas mais variadas formas. Assim, começaram a surgir bases de dados que disponibilizaram a informação dos seus textos completos através da Web (MARCONDES; AND; SAYÃO, 2006).

Segundo a *Association of Research Libraries* (ARL), existem várias definições para bibliotecas digitais, ou seja, não existe uma definição consensual, assim como existem termos como biblioteca eletrônica e virtual. O Trabalho de Drabenstott (DRABENSTOTT, 1994) identifica alguns elementos comuns nas definições de bibliotecas digitais, os quais são descritos a seguir:

- A biblioteca digital não é uma simples entidade;
- A biblioteca digital requer tecnologias para interconectar os recursos de outras bibliotecas digitais e serviços de informação;
- A interoperabilidade entre várias bibliotecas digitais e serviços de informação é transparente aos usuários finais;
- A meta principal é o acesso universal a bibliotecas digitais e serviços de informação;
- As bibliotecas digitais não se limitam a referências bibliográficas ou informações referenciais: elas se estendem aos artefatos digitais que não podem ser representados ou distribuídos em formato impresso.

Alguns conceitos de bibliotecas digitais são apresentados a seguir:

Segundo Callan (CALLAN; SMEATON, 2003), Bibliotecas Digitais são coleções de informação com serviços de entrega para usuários de uma comunidade, podendo utilizar

uma variedade de tecnologias. Tais informações podem ser científica, dados pessoais e até mesmo informações sobre negócios e normalmente são representadas como um texto digitalizado, vídeo, música ou outra mídia qualquer. Com o advento da Web, a utilização de Bibliotecas Digitais pelas mais variadas comunidades tem crescido exponencialmente e características como colaboração e compartilhamento têm se tornado elementos sociais importantes.

A primeira fase da *Digital Library Initiative* (DLI<sup>1</sup>) conceitua DL como: uma coleção de serviços e de objetos informacionais que suportam usuários no manuseio desses, assim como a organização e apresentação desses objetos direta ou indiretamente disponíveis via meio eletrônico ou digital.

Na segunda fase da DLI<sup>2</sup> a definição de uma biblioteca digital é :

A biblioteca digital não é meramente equivalente a uma coleção digitalizada com ferramentas de gestão de informação. Trata-se, também, de uma série de atividades que integram coleções, serviços e pessoas em suporte ao completo ciclo de criação, disseminação, acesso e preservação de dados, informação e conhecimento.

Para esta dissertação iremos utilizar a definição de bibliotecas digitais fornecida por (GONCALVES et al., 2004). O autor utiliza a abordagem 5S (*streams, structures, spaces, scenarios e societies*)<sup>3</sup>. Existe uma definição informal e outra formal. As duas definições são mostradas a seguir:

Um biblioteca digital é um sistema de informação complexo que:

- Descreve as propriedades dos conteúdos das bibliotecas digitais (***Streams***);
- Especifica aspectos organizacionais do conteúdo da biblioteca digital (***Structures***);
- Define visões lógica e de apresentação dos componentes da biblioteca digital (***Spaces***);
- Detalha o comportamento dos serviços da biblioteca digital (***Scenarios***);
- Define atores que utilizam esses serviços (***Societies***).

Na visão formal uma biblioteca digital é uma a 4-tupla (R,Cat, Serv,Soc), onde:

---

<sup>1</sup><http://dli.grainger.uiuc.edu/national.htm>

<sup>2</sup><http://www.dli2.nsf.gov/>

<sup>3</sup>Manteremos as definições em inglês conforme aparecem no trabalho original



- R é um repositório;
- Cat = DMC1 ,DMC2 , ...,DMCK é um conjunto de catalogos de metadados para todas as coleções C1,C2, ...,CK no repositório;
- Serv é um conjunto de serviços contendo pelo menos serviços de indexação, busca e *browsing*;
- Soc é uma sociedade.

Vale ressaltar que a definição anterior captura a sintaxe de uma biblioteca digital, ou seja, o que é uma biblioteca digital. A definição anterior foi considerada por ser a única definição formal de bibliotecas digitais existente e por contemplar várias visões e perspectivas das definições de bibliotecas digitais.

## 2.2 Infra-estrutura Tecnológica

Segundo Vidoti (VIDOTTI; SANT'ANA, 2006), o desenvolvimento de uma biblioteca digital se baseia no planejamento de uma biblioteca tradicional/convencional, desde o processo de aquisição, o processamento técnico, a recuperação, o atendimento ao usuário, até a preservação. Cada um dos elementos do planejamento é detalhado a seguir:

No processo de aquisição são definidos os conteúdos/recursos informacionais que compõem o acervo digital por meio de compra, assinatura, digitalização e seleção de obras pertencentes a outras bibliotecas digitais, repositórios institucionais, periódicos científicos e/ou outros *websites*. Em relação ao acervo, este pode ser composto por recursos multimídias (texto, som e imagens) interligados por informações referenciais e/ou contextuais.

Após o processo de aquisição dos recursos informacionais, inicia-se o processamento técnico dessas obras: catalogação, classificação e indexação dos metadados para que seja possível identificar, localizar e recuperar os documentos. Por outro lado, a Iniciativa de Arquivos Abertos (OAI<sup>4</sup>) provê uma forma padrão para tornar disponível os metadados de um acervo digital via Internet, bem como a coleta de informações de outros acervos via consultas através do protocolo OAI-PMH<sup>5</sup>. Maiores detalhes sobre metadados e OAI serão vistos nas próximas seções.

---

<sup>4</sup>Do inglês *Open Archives Initiative* disponível em <http://www.openarchives.org/>

<sup>5</sup>(*Open Archives Initiative Protocol for Metadata Harvesting* disponível em <http://www.openarchives.org/OAI/openarchivesprotocol.html>)

O terceiro passo é o processo de recuperação. Esse é baseado em estrutura de diretórios ou através de uma ferramenta de busca. No primeiro, as estruturas de diretórios que classificam as obras ou recursos digitais são baseadas num sistema de classificação hierárquico pré-definido. O segundo consiste em percorrer toda a base de metadados à procura das informações que satisfaçam a expressão de busca. Adicionalmente, a recuperação pode ser feita em diferentes bibliotecas digitais e ou outros *websites* de forma similar a um sistema de metabusca.

O processo de disseminação pode ser estático ou dinâmico. Na perspectiva estática, este consiste na elaboração e envio de boletins eletrônicos, e-mails, listas de discussão, enfim, atividades que divulgam a incorporação de novos conteúdos. Na perspectiva dinâmica, o processo de disseminação é visto como uma atividade dinâmica estabelecida conforme ocorre a interação com o usuário do sistema. Nesta visão cada usuário recebe conteúdos específicos.

O atendimento ao usuário pode ser feito por várias formas de comunicação que vão desde e-mails, *chats*, até sistemas que utilizam agentes de interfaces sob a forma de assistentes virtuais.

A política de preservação dos recursos ou objetos deve ser em termos de integridade lógica e física do ambiente informacional, que consiste na instalação de *software* e *hardware* de segurança que devem levar em consideração as questões de acesso, uso, manutenção e atualização dos suportes informacionais.

Para tornar os serviços e conteúdos da biblioteca digital acessíveis através da Internet, alguns serviços devem ser instalados e configurados, dentre eles, destacam-se:

- **Serviços de banco de dados:** Os servidores de banco de dados compreendem os aplicativos que compõem o Sistema Gerenciador de Banco de Dados (SGBD). Dentre eles, destacamos o MySQL e o PostgreSQL. O MySQL tem sido utilizado por ser gratuito, de fácil instalação, está disponível em várias plataformas e possui um bom desempenho. Maiores detalhes podem ser encontrados em <http://www.mysql.com/>. Já o PostgreSQL é um sofisticado SGBD objeto-relacional que suporta recursos como controle de transações e definição de tipos de dados pelos usuários. Também é gratuito e pode ser encontrado em <http://www.postgresql.org.br/>. O PostgreSQL é mais utilizado em aplicações com fortes componentes transacionais.
- **Serviços Web:** Os servidores Web são responsáveis pelas respostas as solicitações que são recebidas, sendo que estas são originadas pelos navegadores dos usuários.

Entre as várias opções para realizar esta tarefa, são citados na literatura o *Internet Information Services* (IIS) da Microsoft e o Apache que pode ser encontrado em <http://www.apache.org/>.

- Serviço de Manutenção de objetos digitais que podem ser armazenados por outras bibliotecas digitais ou mesmo em sites com outros fins. Os objetos digitais normalmente estão armazenados em formato *Portable Document Format* (PDF).
- Serviços de busca: As ferramentas de busca, máquinas de busca ou *search engines*, utilizam *softwares* que indexam e catalogam as páginas e/ou sites da Internet em base de dados, com a finalidade de recuperar os documentos solicitados pelos usuários, segundo as estratégias de busca e os critérios adotados.
- Serviço de gestão de bibliotecas digitais: Existem inúmeros softwares disponíveis para o desenvolvimento de uma biblioteca digital, tais como: Greenstone, Fedora e DSpace, que vão desde aplicativos para o gerenciamento de coleções digitais até a gestão completa de todas as atividades e serviços de uma biblioteca digital.

As bibliotecas digitais, além das atividades oferecidas pelas bibliotecas tradicionais, têm características próprias, que possibilitam a otimização do uso das tecnologias da informação, agregando valores aos serviços oferecidos, possibilitando ao usuário o acesso independentemente de tempo e espaço.

## 2.3 Metadados

Bibliotecas digitais tradicionalmente possuem grandes quantidades de dados (as maiores bibliotecas digitais possuem dados na ordem de *terabytes*) que precisam ser manipulados freqüentemente, de forma eficiente, para fornecer condições de uso favoráveis para os distintos usuários. A medida que as coleções aumentam descrever documentos vem se tornando uma condição fundamental para possibilitar sua posterior recuperação entre os itens da coleção e posterior utilização. Os metadados são utilizados com a finalidade de facilitar a recuperação e o acesso aos objetos digitais de uma biblioteca digital.

Metadados podem ser definidos como dados acerca de outros dados. Segundo a *International Federation of Library Associations* (IFLA) define metadados da seguinte forma:

”Metadados são dados sobre dados. O termo se refere a qualquer informação utilizada para a identificação, descrição e localização de recursos.”

O consórcio W3C (*World Wide Web Consortium*) define metadados utilizando uma visão mais voltada para a Web Semântica. Metadados são definidos como:

”Informações para a Web que podem ser processadas por máquinas.”

No contexto da Web, um dos maiores objetivos dos metadados é permitir que programas especiais, robôs e agentes de *software* processem os metadados associados a documentos e possam então recuperá-los, avaliar sua relevância e manipulá-los com mais eficiência. Entretanto, a definição acima é muito restrita, pois limita a visão de metadados a um ambiente eletrônico, baseado na Internet.

Assim como existem várias definições para o termo metadados, vários padrões também surgiram. A seguir os principais padrões são mostrados.

### 2.3.1 MARC

MARC é o acrônimo de *MAchine-Readable Cataloging*: um conjunto de padrões para identificar, armazenar, e comunicar informações bibliográficas em formato legível por máquina, de forma que diferentes computadores e programas possam reconhecer, processar e estabelecer pontos de acesso dos elementos que compõem a descrição bibliográfica. (CONGRESS., 2007)

O termo *Machine-Readable* é utilizado com o significado de que uma máquina ou um computador possam ler e/ou interpretar dados contidos num catálogo. Um catálogo é um registro bibliográfico. Um registro MARC é composto por três elementos:

- Estrutura: A estrutura de um registro MARC é uma implementação de registros nacionais e internacionais. Exemplos: *Information Interchange Format* (ANSI Z39.2) e *Format for Information Exchange* (ISO 2709).
- Designação do conteúdo: São as *tags*, os códigos e convenções estabelecidas para identificar e adicionalmente caracterizar os elementos de dados dentro de um registro e suportar a manipular desses dados, isto é definido pelo formato MARC 21.
- Conteúdo: O conteúdo da maioria dos elementos de dados são definidos fora do padrão. Exemplo: *Anglo-American Cataloguing Rules, Library of Congress Subject Headings* e *National Library of Medicine Classification*

O formato possibilita a descrição bibliográfica de diferentes tipos de documentos, utilizando-se uma estrutura de campos fixos e variáveis, subcampos e indicadores. Todo

Tabela 1: Campos básicos do Formato MARC21

Campos	Descrição
0XX	Informações de controle, números e códigos
1XX	Autoria (nome pessoal, entidade, evento)
2XX	Títulos, edição, impressão
3XX	Descrição física
4XX	Série
5XX	Notas
6XX	Entradas de assunto
7XX	Entradas secundárias (nome pessoal, entidade, evento, título)
8XX	Entradas secundárias de série
9XX	Uso local

registro MARC é dividido logicamente em campos. Há um campo para a informação de autor, um para título e assim por diante. Como a descrição de cada campo é muito extensa para ser definida dentro do registro, foram definidas *tags* (etiquetas) de três dígitos numéricos para identificar cada campo.

O MARC 21 surgiu em 1999, oriundo da fusão do USMARC (utilizado pela Library of Congress) e do CAN/MARC (utilizado pela National Library of Canada)(CONGRESS., 2007). Na Tabela 1, estão descritos os campos básicos do formato MARC 21.

A Figura 1 abaixo mostra um exemplo de registro no formato MARC21.

090		\$a 308 \$c R484p \$8 1 / \$8 13 / \$8 5 \$a 301.2 \$c R484p \$8 9
100	1	\$a Ribeiro, Darcy, 1922-
245	1 2	\$a O povo brasileiro : \$b a formacao e o sentido do Brasil / \$c Darcy Ribeiro. -
250		\$a 2.ed. -
260	#	\$a Sao Paulo : \$b Companhia das Letras, \$c 1995.
300		\$a 476p. : \$b il.
650	0 3	\$a Antropologia - Brasil. \$a Cultura - Brasil. \$a Etnologia - Brasil.
901		\$a Livro
910		\$a BC \$a CAC \$a FCH \$a CE

Figura 1: Exemplo de registro Marc21

A Figura 2 abaixo mostra como os dados da 1 são interpretados.

<b>Número de Chamada</b>	308 R484p Biblioteca Central / Bib. Filosofia e C. Humanas / Bib. Artes e Comunicacao 301.2 R484p Bib. Educação
<b>Autor Principal</b>	Ribeiro, Darcy, 1922-
<b>Título Principal</b>	O povo brasileiro : a formacao e o sentido do Brasil
<b>Publicação</b>	Sao Paulo : Companhia das Letras, 1995.
<b>Edição</b>	2.ed. -
<b>Descrição Física</b>	476p. : il.
<b>Assuntos</b>	Antropologia - Brasil. Cultura - Brasil. Etnologia - Brasil.

Figura 2: Interpretação do registro Marc21

Segundo (CONGRESS., 2007), apesar de não ter sido desenvolvido especificamente para objetos digitais, o formato MARC, mesmo sendo um padrão já antigo, é robusto e flexível o suficiente para ser adaptado, além de bem estabelecido. O MARC tem como aplicação principal a catalogação bibliográfica.

### 2.3.2 METS(CONGRESS., 2006)

METS é o acrônimo de *Metadata Encoding and Transmission Standard*. O esquema METS é um padrão de codificação e transmissão de metadados, que inclui elementos descritivos, administrativos e estruturais para trabalhos textuais e baseados em imagens.

O METS provê um formato XML, baseado em *XML Schema*, para codificar metadados necessários tanto para gestão de objetos de bibliotecas digitais num repositório quanto para a troca desses objetos entre repositórios (ou entre repositórios e seus usuários).

Um documento METS consiste de sete seções principais, descritas a seguir:

1. Cabeçalho: Contém metadados sobre o documento METS em si, incluindo informações tais como autor, editor, etc.;
2. Metadados descritivos: Estes podem ser internos ao documento ou apontar para registros externos, inclusive em outro formato que não o próprio METS (MARC,

Dublin Core etc.). Múltiplas instâncias de ambos os tipos de documentos são permitidas ;

3. Metadados administrativos: Descreve como os arquivos que compõem a obra foram criados e armazenados, direitos de *copyright*, metadados sobre o objeto original de onde deriva o que está sendo tratado, etc. . Também podem ser internos ou externos ao documento METS;
4. Seção de arquivos: Lista os arquivos que formam o objeto digital em si;
5. Mapa estrutural: É a parte principal de um documento METS e apresenta a hierarquia entre os componentes do objeto digital e faz a ligação desses componentes aos arquivos e metadados de cada componente;
6. Ligações estruturais: Registra a existência de *hiperlinks* entre nós da hierarquia do mapa estrutural;
7. Comportamento: Pode ser usada para associar comportamentos executáveis a conteúdos do objeto METS. Cada comportamento possui uma interface que define abstratamente um conjunto de comportamentos e um módulo executável que implementa e executa os comportamentos definidos na interface.

Exemplos de objetos digitais no formato METS podem ser encontrados na biblioteca digital FEDORA<sup>6</sup>.

### 2.3.3 Dublin Core

Apesar do conceito de metadado ser anterior à Internet (HILLMANN, 2005), especialmente a Web, o interesse global no uso de padrões de metadados aumentou de forma significativa motivado pela sobrecarga de informação, como resultado da vasta quantidade de informações não estruturadas disponíveis na Web. Apenas a adoção em larga escala de padrões de metadados para a descrição de recursos, possibilitará a melhora na recuperação de recursos relevantes em um dado contexto.

Com essa problemática em mente, foi desenvolvido o Dublin Core (DUBLIN, 2008), que nada mais é do que um padrão com um conjunto simples e efetivo de elementos utilizados para descrever uma grande variedade de recursos. As principais características do Dublin Core são:

---

<sup>6</sup><http://www.fedora.info/>

- Simplicidade na criação e manutenção permitindo o seu uso por não-especialistas;
- Uma semântica com entendimento universal que facilita a interpretação de usuários com diferentes formações; e a
- Extensibilidade que permite a adição de elementos para atender as especificidades de diferentes comunidades.

O padrão Dublin Core possui dois níveis: Simples e Qualificado. O Dublin Core Simples possui quinze elementos (ver Tabela 2), enquanto o Dublin Core Qualificado possuem três elementos adicionais (ver Tabela 3), além de um grupo de elementos de refinamentos (conhecidos como qualificadores) que refina a semântica dos elementos, de forma a ser útil na descoberta de recursos. A semântica do Dublin Core tem sido estabelecida por um grupo internacional e multidisciplinar de profissionais (HILLMANN, 2005).

Tabela 2: Conjunto de elementos do formato Dublin Core

<i>Nome do Elemento</i>	Definição
<i>dc:title</i>	Um nome dado ao recurso. (ex.: título)
<i>dc:creator</i>	Uma entidade primariamente responsável pela criação do conteúdo do recurso. (ex.: autores)
<i>dc:subject</i>	Um tópico de conteúdo do recurso. (ex.: palavras-chave)
<i>dc:description</i>	Um apanhado do conteúdo do recurso. (ex.: resumo)
<i>dc:publisher</i>	Uma entidade responsável pela disponibilização do recurso. (ex.: editora)
<i>dc:contributor</i>	Uma entidade responsável por fazer contribuições ao conteúdo do recurso.
<i>dc:date</i>	Data associada com a criação ou disponibilização do recurso.
<i>dc:type</i>	A natureza ou gênero do conteúdo do recurso.
<i>dc:format</i>	A manifestação física ou digital do recurso.
<i>dc:identifier</i>	Uma referência, não ambígua do recurso, dentro de um dado contexto. (ex.: URL, ISBN)
<i>dc:source</i>	Uma referência para um recurso do qual o presente recurso é derivado.
<i>dc:language</i>	Idioma no qual o conteúdo intelectual do recurso está escrito.
<i>dc:relation</i>	Uma referência para um recurso relacionado.
<i>dc:coverage</i>	A extensão ou escopo do conteúdo do recurso. (tipicamente, <i>dc:coverage</i> irá incluir uma localização geográfica)
<i>dc:rights</i>	Informações sobre direitos, propriedade intelectual ou condições de uso do recurso.



Tabela 3: Conjunto de elementos adicionais do formato Dublin Core Qualificado

<i>Nome do Elemento</i>	Definição
<i>dc_qual:audience</i>	Uma entidade para a qual o recurso é dirigido ou útil.
<i>dc_qual:provenance</i>	Indicação de mudanças na posse e/ou custódia do recurso, desde a sua criação, que forem significativas para garantir sua autenticidade, integridade e interpretação.
<i>dc_qual:rightsHolder</i>	Uma pessoa ou organização que possui ou controla os direitos sobre o recurso.

## 2.4 Interoperabilidade

A capacidade de bases de dados trocarem e compartilharem documentos, consultas e serviços, usando diferentes plataformas, estrutura de dados e interfaces, é chamada de interoperabilidade. Através dessa troca e compartilhamento são realizadas interações entre sistemas. A principal característica para o sucesso dessas interações é a consistência, a qual é alcançada através do uso de padrões.

No contexto de biblioteca digital, é utilizada a Iniciativa de Arquivos Abertos(OAI<sup>7</sup>). A *Open Archives initiative* (SOMPEL, 2000) é uma iniciativa que tem como papel principal desenvolver e promover padrões de interoperabilidade com o intuito de facilitar a disseminação eficiente de conteúdo. Tal iniciativa surgiu em 1999 após a realização da convenção de Santa Fé, onde foi apresentado um modelo técnico e organizacional simples para suportar interoperabilidade básica entre arquivos de *e-prints* (LOPES, 2007). Com isso, a OAI começou a ser utilizada pelos mais variados provedores de conteúdo, especialmente publicações científicas, além de possibilitar que Bibliotecas Digitais ao redor do mundo pudessem interoperar formando uma federação (SOMPEL, 2000).

### 2.4.1 *Open Archives Initiative - Protocol for Metadata Harvesting* (OAI-PMH)

Para possibilitar a interoperabilidade entre diferentes Bibliotecas Digitais a OAI desenvolveu um protocolo de comunicação conhecido como OAI-PMH (*Open Archives Initiative - Protocol for Metadata Harvesting*) (OAI-PMH, 2008), o qual provê um modelo que garante interoperabilidade independente de aplicação baseado em colheita de metadados (*metadata harvesting*). O protocolo OAI-PMH define como se dá a transferência de metadados entre duas entidades básicas: provedores de dados e provedores de serviços.

<sup>7</sup>do inglês Open Archives Initiative

Os **provedores de dados** (*data providers*) são entidades que administram sistemas que suportam o protocolo OAI-PMH como meio de exposição de metadados. São exemplos de provedores de dados o E-prints, DSpace e Kepler. O provedor de dados é responsável por gerenciar um repositório que é utilizado para expor os metadados para os *harvesters*. Um **repositório** é um servidor de rede acessível capaz de processar as seis requisições do OAI-PMH. Tais requisições serão descritas ao final desta seção.

Os **provedores de serviços** utilizam os metadados como base para fornecer serviços especiais. Tais metadados são obtidos através de uma colheita OAI-PMH. Além disso, os provedores de serviços são responsáveis por operar um **harvester**, aplicação cliente que envia requisições OAI-PMH, com o intuito de realizar colheita de metadados dos repositórios. São exemplos de provedores de serviço o Arc, CYCLADES, OAIster e my.OAI

Para se adequar as mais variadas configurações de repositórios, o OAI-PMH distingue entre três entidades utilizadas para fazer os metadados acessíveis pelo OAI-PMH, que são:

- **Recurso** (*resource*) é o objeto descrito pelos metadados. A natureza do recurso, se físico ou digital, se é armazenado no repositório ou é constituinte de outra base de dados, está fora do escopo do OAI-PMH.
- **Item** é um constituinte de um repositório através do qual metadados sobre um recurso podem ser disseminados. Conceitualmente, um item é um contêiner que armazena ou gera dinamicamente metadados sobre um único recurso em diversos formatos, cada um podendo ser obtido como um **registro** via OAI-PMH. Cada item possui um **identificador único** (*unique identifier*), o qual identifica um item de forma não ambígua dentro de um repositório. O identificador único é usado nas requisições OAI-PMH para a extração de metadados de um item. Itens podem conter metadados em múltiplos formatos. O identificador único mapeia o item, e todos os possíveis registros disponibilizados de um único item, de forma a compartilharem o mesmo identificador único.
- **Registro** (*Record*) são metadados espessos em um formato específico. Um registro é retornado, codificado em XML, em resposta a uma requisição OAI-PMH de um metadado de um item. Um registro é identificado de forma não ambígua pela combinação do identificador único do item no qual o registro é disponibilizado, o *metadataPrefix* que identifica o formato do metadado do registro, e uma marca de tempo (*timestamp*), como por exemplo, a data de criação.

Tabela 4: Verbos OAI-PMH

<i>Nome do Verbo</i>	Definição
<i>Identify</i>	Este verbo é utilizado para recuperar as informações sobre o repositório.
<i>ListMetadataFormats</i>	Este verbo recupera os formatos de metadados disponíveis no repositório.
<i>GetRecord</i>	Este verbo é utilizado para recuperar um único registro do repositório.
<i>ListRecords</i>	Este verbo é utilizado para coletar os metadados do repositório.
<i>ListIdentifiers</i>	Este verbo é uma versão abreviada do verbo ListRecords, que retorna apenas os cabeçalhos dos registros.
<i>ListSets</i>	Este verbo lista a estrutura do conjunto de um repositório .

A interação entre o provedor de serviços e o provedor de dados acontece da seguinte maneira (ver Figura 3):

1. O provedor de serviços que deseja realizar uma colheita de metadados, inicialmente envia uma requisição HTTP para o provedor de dados utilizando os *verbos* definidos no protocolo OAI-PMH;
2. Dependendo da requisição solicitada, o provedor de dados envia como resposta os metadados solicitados em um arquivo XML;
3. Com os metadados em mãos, o provedor de serviços pode então utilizá-los para disponibilizar um novo serviço, como por exemplo, uma recomendação.

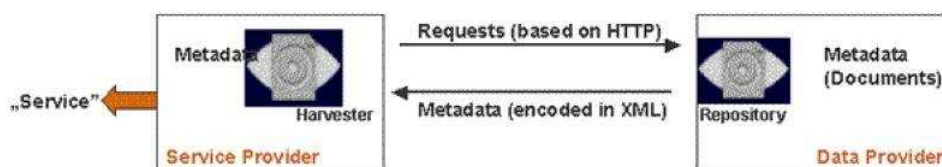


Figura 3: Interação entre as entidades básicas do OAI-PMH

A colheita de metadados de um provedor de dados só é possível utilizando-se dos seis tipos de requisições especificados no protocolo OAI-PMH, que são: *Identify*, *ListMetadataFormats*, *ListSets*, *ListIdentifiers*, *ListRecords* e *GetRecords*. A Tabela 4 mostra a definição resumida desses verbos. A descrição detalhada sobre cada um desses verbos pode ser encontrada na especificação do protocolo OAI-PMH em (OAI-PMH, 2008).

Requisições OAI-PMH devem ser expressas como requisições HTTP utilizando os métodos GET ou POST, ou seja, os repositórios devem suportar ambos os métodos. Existe uma URL base, para todas as requisições. A URL base especifica o *host* e a porta, opcionalmente o caminho, do servidor HTTP atuando como um repositório. Todas as requisições consistem de uma lista de argumentos da forma *key=value*, onde:

- *key* é a *string verb*;
- *value* é um dos seis verbos definidos pelo protocolo OAI-PMH.

As respostas a todas as requisições do protocolo OAI-PMH são codificadas em um arquivo XML. Para maiores informações sobre o formato das requisições e respostas, sugere-se (OAI-PMH, 2008).

O protocolo OAI-PMH possui algumas limitações. Essas limitações são citadas abaixo:

- É necessário percorrer milhares de registros para encontrar os registros com uma determinada palavra-chave.
- Se um registro, numa determinada posição *n*, for modificado, têm-se que percorrer todos os *n-1* registros para obtermos acesso a modificação que foi realizada.
- O número de registros retornados por consulta é fornecido apenas por algumas poucas implementações de repositório digital.

## 3 *Sistemas de Recomendação*

*“O ser humano não pode deixar de cometer erros; é com os erros, que os homens de bom senso aprendem a sabedoria para o futuro.”*

Plutarco

Este capítulo apresenta um breve histórico, conceitos, classificações, técnicas e exemplos de pesquisas na área de Sistemas de recomendação. O objetivo do capítulo é fornecer uma visão geral e uma base da área de sistemas de recomendação para o restante do trabalho.

### 3.1 Sistemas de Recomendação, filtragem e recuperação de informação

A grande quantidade de informação em meio eletrônico e a facilidade de disponibilização e acesso a essas através da Internet/Web, fazem com que as pessoas possuam uma enorme diversidade de opções de escolha. Esta sobrecarga de informação dificulta o processo de escolha dos usuários. Adicionalmente, muitas vezes os usuários possuem pouca ou quase nenhuma experiência num determinado assunto ou tema, dificultando assim o processo de conseguir informações relevantes e interessantes.

Uma solução para este problema é construir sistemas que se adaptem às necessidades dos usuários, assegurando que apenas informações interessantes para os usuários sejam recuperadas e apresentadas do modo que esta seja mais adequada para aqueles usuários. Diariamente, uma forma cotidiana de resolvermos este problema é confiarmos nas recomendações que são fornecidas por outras pessoas (*word of mouth*)(SHARDANAND; MAES, 1995), cartas de recomendação, opiniões de revisores de livros, filmes e músicas, revistas e jornais, entre outros.

Os sistemas de recomendação auxiliam neste processo de indicação já bastante conhecido na relação social entre seres humanos (RESNICK; VARIAN, 1997). Segundo os autores, num sistema típico as pessoas fornecem recomendações como entrada e o sistema agregam e direcionam as recomendações para as pessoas interessadas em recebê-las.

O primeiro sistema de recomendação foi chamado de Tapestry (RESNICK; VARIAN, 1997)(GOLDBERG et al., 1992a). Segundo Goldberg, Tapestry é um sistema experimental de e-mail desenvolvido pela Xerox, cuja motivação veio do aumento do uso do e-mail que resultou numa inundação de documentos nas caixas postais dos usuários. A idéia por trás do sistema é que uma filtragem mais efetiva pode ser conseguida utilizando humanos no processo de filtragem. Os desenvolvedores criaram a expressão "filtragem colaborativa", neste tipo de sistema onde as pessoas colaboram umas com as outras para realizar a filtragem registrando suas reações a documentos que elas leram. As reações são geralmente chamadas de anotações e podem afirmar se um documento é interessante ou não.

Segundo (RESNICK; VARIAN, 1997) o termo sistema de recomendação é um termo genérico e, adicionalmente, é preferível por duas razões:

1. Os recomendadores podem explicitamente não colaborar com aqueles que recebem

a recomendação, pois pode ser que eles nem se conheçam.

2. Recomendações podem sugerir itens particularmente interessantes, incluindo aqueles que poderiam ser descartados.

Segundo (BELKIN; CROFT, 1992) filtragem de informação e recuperação de informação são dois lados da mesma moeda. Em seu artigo os autores comentam sobre as semelhanças e diferenças entre os dois termos. Segundo os autores, filtragem de informação é o nome utilizado para descrever uma variedade de processos que envolvem a entrega de informação para as pessoas que realmente necessitam dela. Alguns artigos técnicos descrevem aplicações focadas em filtragem aplicadas a e-mail, sistemas multimídias e documentos eletrônicos em escritórios, porém não fica clara a distinção entre filtragem e processos relacionados com recuperação de informações somente.

Segundo (BAEZA-YATES; RIBEIRO-NETO, 1999), um modelo de recuperação de informação é uma quádrupla  $[D, Q, F, R(q_i, d_j)]$  onde:

1.  $D$  é um conjunto composto por representações para os documento na coleção.
2.  $Q$  é um conjunto composto pela necessidade das informações dos usuários. Tais informações são chamadas de consultas .
3.  $F$  é um *framework* para modelar representação de documentos, consultas e seus relacionamentos.
4.  $R(q_i, d_j)$  é uma função de *ranking* que associa um número real com uma consulta  $q_i \in Q$  e uma representação do documento  $d_j \in D$ . Tal *ranking* define uma ordem entre os documentos que satisfazem a consulta  $q_i$ .

Um Sistema de recuperação de informação tem a função de conduzir o usuário para aqueles documentos que melhor irão habilitá-lo a satisfazer suas necessidades de informação Robertson apud (BELKIN; CROFT, 1992). O campo de recuperação de informação tradicionalmente desenvolve tecnologias para armazenar, indexar e recuperar documentos textuais. Num sistema de recuperação de informação os usuários descrevem suas necessidades na forma de uma consulta (*query*) e o sistema tenta encontrar itens que combinem a consulta com o conjunto de documentos armazenados.

Nos sistemas de recuperação de informação a necessidade de informação do usuário é dinâmica e temporária (HERLOCKER, 2000a). Este tipo de abordagem tende a manter uma

base com características mais estáticas no armazenamento das informações, sendo que a interação ocorre por iniciativa do usuário e este deve saber explicitar através de palavras chave a sua real necessidade. No contexto da Web, os motores de busca e os bancos de dados bibliográficos são os exemplos mais significativos de sistemas de recuperação de informação.

Diferentemente dos sistemas de recuperação de informação, nos sistemas de filtragem de informação a necessidade de informação dos interesses dos usuários são de longa duração. As preferências, interesses e necessidades dos usuários ou grupos são armazenadas em perfis. A filtragem deve ser aplicada a cada novo item adicionado na base de dados, procurando verificar se este atende ao perfil do usuário. Os sistemas de recomendação utilizam as técnicas de filtragem de informação para solucionar o problema da sobrecarga de informação.

Com o surgimento do comércio eletrônico, a Internet tornou-se um ambiente ideal para que a personalização seja utilizada (TORRES, 2004). Uma excelente forma de aplicar personalização é através dos sistemas de recomendação. Os websites de comércio eletrônico são atualmente o maior foco de utilização dos sistemas de recomendação, empregando diversas técnicas para encontrar os produtos mais adequados para seus clientes e aumentar deste modo a lucratividade e a fidelização destes. Introduzido em 1996, o My Yahoo foi o primeiro web site a utilizar os sistemas de recomendação em grandes proporções, utilizando a estratégia de customização (MANBER; PATEL; ROBISON, 2000).

Os sistemas de recomendação são aplicados a uma variedade de sites de comércio eletrônico e têm sido utilizados em pesquisas em bibliotecas digitais (CALLAN et al., 2003), comunidades virtuais (PAPAGELIS; PLEXOUSAKIS, 2003) , e-learning (TANG; MCCALLA, 2003) e sistemas multiagentes (CAZELLA; ALVARES, 2006) (KAELBLING; SAFFIOTTI, 2005).

## **3.2 Formalização do Problema (ADOMAVICIUS; TUZHILIN, 2005)**

Definição: Seja  $C$  o conjunto de todos os usuários de um determinado sistema, e seja  $S$  o conjunto de todos os possíveis itens que podem ser recomendados como livros, filmes ou restaurantes. Seja  $u$  a função utilidade que mede o quão útil é um determinado item  $s$  para um determinado usuário  $c$ , i.e.,  $u : C \times S \rightarrow \mathbb{R}$ , onde  $\mathbb{R}$  é um conjunto totalmente ordenado. Então, para cada usuário  $c \in C$ , procura-se um item  $s' \in S$  que maximiza a função utilidade do usuário. Isto pode ser expressado pela equação abaixo:



$$\forall c \in C, s'_c = \operatorname{argmax}_{s \in S}(c, s) \quad (3.1)$$

Em um sistema de recomendação a utilidade de um item é geralmente representada por uma avaliação que indica o quanto um determinado usuário gosta de um item (e.g. Rosângela deu ao filme "Titanic" a nota 7 de um total de 10). No entanto, conforme descrito na definição acima, a função de utilidade pode ser uma função arbitrária.

Cada elemento do espaço de usuários  $C$  pode ser definido através de um perfil que inclui as características do usuário, como a sua idade, sexo, estado civil, renda, etc. No caso mais simples, o perfil pode conter um único elemento como uma identificação do usuário. Da mesma forma, cada item do espaço  $S$  pode ser definido por um conjunto de características. Por exemplo, na recomendação de filmes, na qual  $S$  é uma coleção de filmes, cada filme pode ser representado não apenas pelo seu ID, mas também pelo seu título, gênero, diretor, ano de lançamento, atores principais, etc.

O problema central dos sistemas de recomendação reside no fato da função utilidade  $u$  geralmente não ser definida em todo o espaço  $C \times S$ , mas apenas em um subconjunto deste. Isto significa que  $u$  precisa ser extrapolado para todo o espaço  $C \times S$ . Geralmente em sistemas de recomendação, a utilidade é definida através de avaliações, e estas são definidas apenas nos itens previamente avaliados pelos usuários. Deste modo, o algoritmo de recomendação deve ser capaz de estimar (predizer) as avaliações não realizadas para os pares usuário-item e de fazer recomendações apropriadas baseadas nestas predições.

A extrapolação de avaliações conhecidas para avaliações inexistentes é geralmente feita através de dois modos:

1. Especificação de heurísticas que definem a função utilidade e validam empiricamente sua performance e
2. pelas estimativas da função utilidade através da otimização de algum critério de performance como o erro médio quadrático.

Assim que avaliações desconhecidas são estimadas, o sistema de recomendação seleciona aquelas com maiores avaliações para serem recomendadas.

A predição de avaliações de itens ainda não avaliados pode ser feita de diferentes formas utilizando métodos de aprendizagem de máquina, teoria de aproximação e vários tipos de heurísticas. Os sistemas de recomendação são classificados de acordo com o

método de predição utilizado. Esta classificação é feita usualmente em três categorias propostas (HERLOCKER, 2000b; CLAYPOOL et al., 1999; HUANG et al., 2002; BALABANOVIC; SHOHAM, 1997):

- **Recomendações Baseadas em Conteúdo:** O usuário receberá recomendações de itens similares a itens preferidos no passado;
- **Recomendações Colaborativas:** O usuário receberá recomendações de itens que pessoas com gostos similares aos dele preferiram no passado. Este método é subdividido em duas categorias: a primeira baseada em memória (*memory-based*), e a segunda baseada em modelo (*model-based*);
- **Recomendações Híbridas:** Estes métodos combinam tanto estratégias de recomendação baseadas em conteúdo quanto estratégias baseadas em colaboração.

Nas próximas seções comentaremos sobre cada uma das técnicas. Mas, antes disso devemos nos preocupar em como coletar as informações do usuário.

### 3.3 Coleta de Informações

Para ser possível realizar uma recomendação para um determinado usuário, primeiramente é necessário adquirir algum conhecimento de quem é esse usuário, ou seja, é preciso capturar e armazenar dados pessoais e comportamentais relativos ao usuário. A coleta de informações relativas ao usuário pode ocorrer de duas formas: **explícita** ou **implícita**.

- Na coleta de informações de forma explícita, solicita-se diretamente ao usuário que ele especifique as suas preferências. Normalmente é disponibilizado para o usuário uma área de cadastro com informações pessoais como: nome, data de nascimento, sexo, endereço, preferências entre outras.
- Na coleta de informações de forma implícita o sistema infere as preferências do usuário através da observação das ações do usuário durante a utilização do sistema. Normalmente, nesse processo são utilizados *cookies*, um mecanismo pelo qual um site Web consegue identificar que determinado computador está se conectando mais uma vez a ele e as suas interações com o sistema. Ações como, colocar páginas nos favoritos e visualizar páginas por um longo período de tempo podem indicar o interesse do usuário por determinado conteúdo.

### 3.4 Filtragem Baseada em Conteúdo (*Content-based Filtering*)

Na técnica de Filtragem Baseada em Conteúdo (FBC) o sistema recomenda itens similares aos que o usuário já consumiu (e provavelmente gostou) no passado. A abordagem de recomendar itens baseado em seu conteúdo tem suas origens nas técnicas de Recuperação de Informação (RI). Por exemplo, documentos são recomendados se seu conteúdo, ao compará-lo com o perfil de um usuário, obtiver um grau de similaridade superior a um limiar.

Segundo (ADOMAVICIUS; TUZHILIN, 2005) nesta técnica, a utilidade  $u(c, s)$  de um item  $s$  para um usuário  $c$  é estimada baseada nas utilidades  $u(c, s_i)$  já assinaladas pelo usuário  $c$  aos itens  $s_i \in S$  e que são similares ao item  $s$ . Para exemplificar, em uma aplicação de recomendação de filmes, no intuito de recomendar filmes para o usuário  $c$ , o algoritmo baseado em conteúdo tenta entender as características dos filmes que o usuário  $c$  avaliou no passado com notas altas (atores específicos, diretores, gêneros, etc.). Assim, apenas os filmes que possuem grande grau de semelhança com estes filmes avaliados com altas notas serão recomendados.

Em sistemas baseados em conteúdo a função de utilidade  $u(c,s)$  é definida como:

$$u(c, s) = \text{score}(\text{PerfilBaseadoemConteudo}(c), \text{Conteudo}(s)) \quad (3.2)$$

Onde:  $\text{PerfilBaseadoemConteudo}(c)$  - é o perfil do usuário  $c$ , contendo seus gostos e suas preferências.  $\text{Conteudo}(s)$  - é o perfil do item  $s$ , por exemplo um conjunto de atributos que caracterizam o item  $s$ .

Geralmente o perfil e o conteúdo do item são representados como vetores, utilizando assim o modelo vetorial, que será visto na seção 8.

De acordo com (HERLOCKER, 2000b), cientistas têm empregado grandes esforços na tentativa de aliviar o problema ocasionado pela sobrecarga de informações, através da elaboração de projetos que visam o desenvolvimento de tecnologias capazes de reconhecer e categorizar informações de forma automatizada. Ou seja, já é possível encontrar *software* que possuem como objetivo principal a geração automática de descrições dos conteúdos dos itens e comparar essas descrições com os interesses dos usuários, na tentativa de identificar se o item possui ou não relevância para o usuário. Esta técnica é chamada de filtragem baseada em conteúdo, pois realiza uma seleção baseada na análise do conteúdo dos itens a ser recomendado e no perfil do usuário.

O perfil do usuário é estabelecido através de informações fornecidas por ele próprio ou através de ações, como seleção e consumo de itens. Uma forma de se trabalhar com a filtragem baseada em conteúdo é através da solicitação da análise de itens feita ao próprio usuário, onde este avaliaria alguns itens e indicaria se os mesmos são de seu interesse ou não. Após a realização da avaliação pelo usuário, o sistema pode buscar itens que possuem conteúdo semelhante ao item que foi avaliado como relevante para o usuário e desconsiderar os itens cujo conteúdo não é similar aos itens avaliados pelo usuário.

É comum em sistemas que utilizam filtragem baseada em conteúdo, apresentar limitações como: difícil análise de conteúdos pouco estruturados, como imagens e músicas; o entendimento do conteúdo do texto pode ser prejudicado com o uso de sinônimos, o que pode acarretar a recomendação de itens que não são relevantes ao usuário; e existe ainda, a possibilidade de ocorrer uma “super especialização”, ou seja, não há surpresa na recomendação, pois o sistema só recomenda itens cujo conteúdo se assemelhe ao perfil do usuário (REATEGUI; CAZELLA, 2005). Dessa forma, itens que podem ser relevantes ao usuário, mas não são semelhantes ao perfil do mesmo, não serão recomendados.

Além dos métodos baseados em recuperação de informação existem outras técnicas baseadas em conteúdo, tais como classificadores bayesianos (MOONEY; ROY, 2000), várias técnicas de aprendizagem de máquina, incluindo agrupamento, árvores de decisão e redes neurais (PAZZANI; BILLSUS, 1997). Essas técnicas são baseadas num modelo aprendido.

### 3.5 Filtragem Colaborativa (*Collaborative Filtering*)

Como consequência das principais limitações associadas a filtragem baseada em conteúdo, foi desenvolvida uma nova estratégia, chamada de filtragem colaborativa (HERLOCKER, 2000b). A grande diferença da filtragem colaborativa quando comparada a filtragem baseada em conteúdo, está no fato de não ser necessária a compreensão ou conhecimento do conteúdo dos itens.

Segundo (ADOMAVICIUS; TUZHILIN, 2005) os sistemas de filtragem colaborativa tentam prever a utilidade dos itens para um usuário particular com base nos itens previamente avaliados por outros usuários que também participam do sistema. Mais formalmente, a utilidade  $u(c, s)$  do item  $s$  para o usuário  $c$  é estimada baseada nas utilidades  $u(c_j, s)$  assinaladas para o item  $s$  pelos usuários  $c_j \in C$  que são similares ao usuário  $c$ .

A técnica de Filtragem Colaborativa baseia-se no fato de que as melhores recomendações para um indivíduo são aquelas fornecidas por pessoas que possuem gostos

similares aos dele.

A figura abaixo ilustra o processo da filtragem colaborativa.

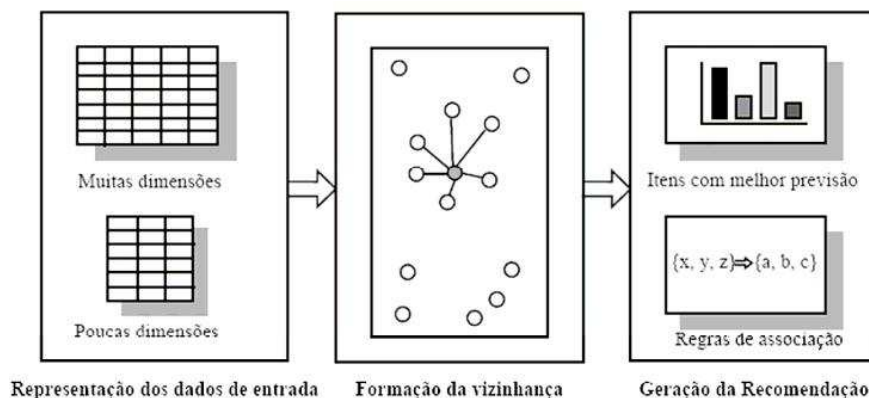


Figura 4: Processo de Recomendação utilizando Filtragem Colaborativa (imagem obtida de (QUEIROZ, 2002))

O processo de Filtragem Colaborativa pode ser descrito em três passos:

1. **Representação dos dados de entrada:** o usuário expressa suas preferências fornecendo notas a itens propostos pelo sistema. Estas avaliações (positivas e negativas) indicam os interesses do usuário em itens específicos, e são armazenadas como o perfil do usuário. A forma mais simples de armazenar o perfil é como uma matriz de  $m$  itens  $\times$   $n$  usuários, onde as células contêm as avaliações fornecidas. Note que as avaliações também podem ser fornecidas de forma implícita, por exemplo um site de comércio eletrônico pode considerar que o usuário demonstra interesse por um produto ao comprá-lo;
2. **Formação da vizinhança:** para fornecer uma recomendação, o sistema compara este perfil com os perfis dos outros usuários, e atribui um peso a estes de acordo com o grau de similaridade com o perfil do usuário que receberá a recomendação. A métrica usada para determinar a similaridade pode variar. O resultado desta comparação é um conjunto dos 'vizinhos mais próximos' do usuário. Este conjunto formaliza o conceito de pessoas com gostos similares;
3. **Geração da recomendação:** finalmente, o sistema usa a informação contida no conjunto dos vizinhos mais próximos para recomendar itens para o usuário, isto é, os itens mais apreciados pelos vizinhos serão recomendados.

A essência dos sistemas que utilizam filtragem colaborativa está exatamente na troca

de experiências entre os usuários que possuem interesses em comum. Nos sistemas colaborativos, a filtragem dos itens ocorre simplesmente observando-se as avaliações realizadas pelos usuários do sistema. Como exemplo, (HERLOCKER, 2000b) cita o sistema Tapestry (GOLDBERG et al., 1992b), o qual permitia ao usuário especificar uma consulta como: mostre-me todos os memorandos que um determinado usuário considera importante. Dessa forma, membros de uma comunidade podiam se beneficiar da experiência de outros usuários. Portanto, em um sistema de filtragem colaborativa, um usuário deve pontuar cada item experimentado, indicando o quanto este item é relevante para o usuário.

A filtragem colaborativa apresenta vantagens como: possibilidade de apresentar aos usuários recomendações inesperadas, ou seja, recomendação de itens que não estavam sendo pesquisados de uma forma ativa; possibilidade de formação de comunidades virtuais através da identificação de interesses similares entre usuários. Mas a filtragem colaborativa também apresenta alguns problemas (REATEGUI; CAZELLA, 2005), como por exemplo:

- **Problema do primeiro avaliador:** quando um novo item é adicionado ao sistema, não existe a possibilidade de este item ser recomendado para um usuário até que este item seja classificado por outro usuário.
- **Problemas de pontuações esparsas:** se o número de usuário do sistema for pequeno se comparado ao volume de informações, é possível que as pontuações para os itens se tornem muito esparsas.
- **Similaridade:** caso o usuário possua interesses diferentes quando comparado aos demais usuários do sistema (usuários com gostos incomuns), então esse usuário pode não se beneficiar de sistemas que utilizem filtragem colaborativa (CLAYPOOL et al., 1999).

### 3.6 Filtragem Híbrida (*Hybrid Filtering*)

Em uma abordagem de filtragem híbrida, procura-se combinar os pontos fortes da filtragem colaborativa e filtragem baseada em conteúdo de modo a desenvolver um sistema capaz de melhor atender as necessidades do usuário, ou seja, o que se quer é uma maior exatidão nas recomendações geradas (HUANG et al., 2002). Segundo (CLAYPOOL et al., 1999), com a combinação de ambas as estratégias, podem ser alcançados os benefícios da filtragem baseada em conteúdo (predição para todos os itens e usuários, sem a dependência do número de usuários e do número de avaliações dos itens), enquanto se

ganha em exatidão nas predições de filtragem colaborativa conforme o número de usuários e avaliações cresce.

Algumas das características que a filtragem híbrida herda (REATEGUI; CAZELLA, 2005) da filtragem colaborativa e filtragem baseada em conteúdo, são (ver Figura 5):

- Descoberta de novos relacionamentos entre usuários.
- Recomendação de itens diretamente relacionado ao histórico.
- Bons resultados para usuários incomuns.
- Precisão independente do número de usuários.

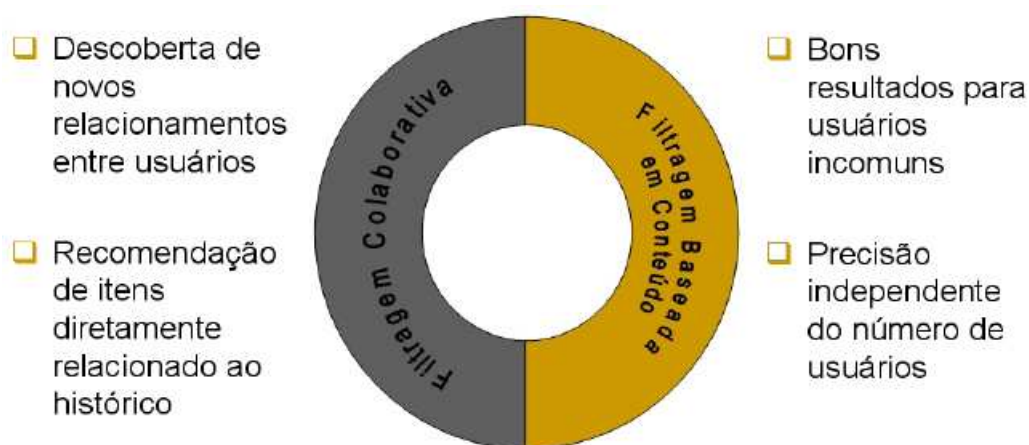


Figura 5: Características herdadas pela Filtragem Híbrida (imagem obtida de (REATEGUI; CAZELLA, 2005))

Existem diferentes formas de combinar técnicas colaborativas e técnicas baseadas em conteúdo em um sistema de recomendação híbrido. Podemos dividir estas formas das seguintes maneiras:

1. implementando os métodos colaborativos e baseados em conteúdo em separado e combinando suas predições;
2. incorporando algumas características de sistemas baseados em conteúdo dentro de abordagens colaborativas;
3. incorporando algumas características de sistemas colaborativos dentro de abordagens baseadas em conteúdo, e
4. construindo um modelo unificado que incorpora ambas características.

Na solução 1, um sistema de recomendação híbrido pode implementar separadamente diferentes abordagens (colaborativas ou baseadas em conteúdo) e então combiná-las de diferentes maneiras: uma forma é fazer a regressão linear das avaliações encontradas, ou utilizar o método denominado voting scheme (PAZZANI, 1999). Outra forma para fazer a combinação entre as avaliações é escolher uma em um dado momento e a outra em outra situação. A escolha por uma ou outra avaliação dependerá de uma métrica pré-estabelecida para medir a "qualidade" da predição. Assim itens que possuam uma vizinhança com forte correlação com outros itens poderão ser preditos através de uma abordagem colaborativa, já outros itens com baixa correlação poderão ser recomendados para usuários que possuam um estereótipo bem conhecido dentro de uma representação já conhecida (e.g. ontologia) em um sistema de recomendação baseado em conteúdo.

Burke (BURKE, 2002) introduziu uma taxonomia para sistemas de recomendação híbridos. A Tabela 5 mostra a taxonomia criada.

Tabela 5: Classificação de Sistemas Híbridos

Forma	Descrição
<b>Ponderado</b>	A similaridade de um item é computada a partir de uma combinação de várias técnicas de recomendação, com diferentes pesos para cada uma delas.
<b>Alternado</b>	O sistema utiliza um critério para alternar a técnica que gera a recomendação.
<b>Cascata</b>	Uma técnica de recomendação refina as recomendações fornecidas por outra técnica.
<b>Combinação de features</b>	Features de diferentes dados são utilizados em um único algoritmo.
<b>Aumento de features</b>	As recomendações geradas por uma técnica de recomendação são utilizadas como entrada para outra técnica.
<b>Misto</b>	Recomendações de várias técnicas são apresentadas na mesma lista.
<b>Meta-level</b>	O modelo aprendido por uma técnica de recomendação é utilizado como informação de entrada para outra técnica.

### 3.7 Pesquisa em Sistemas de Recomendação

A seguir são apresentadas três tabelas de classificação de pesquisa de técnicas aplicadas a Sistemas de Recomendação proposta por Adomavicius (ADOMAVICIUS; TUZHILIN, 2005).



Nestas tabelas o autor cita uma série de pesquisas que fazem uso das técnicas apresentadas, classificando-as por técnica de recomendação e outras técnicas aplicadas. Esta tabela procura ilustrar o número de pesquisas que têm sido realizadas na área de Sistemas de Recomendação, destacando a aplicação de técnicas variadas.

Deve-se observar que o autor apresenta duas grandes classes de técnicas aplicadas a Sistemas de Recomendação (Breese apud (ADOMAVICIUS; TUZHILIN, 2005)): baseada em Heurística (baseado em memória) e baseada em Modelo. O autor define técnicas baseadas em Heurística como sendo heurísticas que fazem previsões de avaliações baseando-se em toda a coleção de avaliações feitas pelos usuários aos itens, enquanto técnicas baseadas em Modelo utilizam a coleção de avaliações para aprender o modelo, o qual será utilizado para realizar previsões de avaliações.

Tabela 6: Pesquisa em Recomendação Baseada em Conteúdo

Técnicas de Recomendação	Técnicas Aplicadas	
	Baseada em Heurística	Baseada em Modelo
Baseada em Conteúdo	<p>Técnicas mais utilizadas: TF-IDF (recuperação de informação) Agrupamento Exemplos de Pesquisas:</p> <ul style="list-style-type: none"> <li>• Lang, 1995</li> <li>• Balabanovic &amp; Shoham 1997</li> <li>• Pazzani &amp; Billsus 1997</li> </ul>	<p>Técnicas mais utilizadas: Classificadores Bayesianos Agrupamento Árvores de Decisão Redes Neurais Exemplos de Pesquisas:</p> <ul style="list-style-type: none"> <li>• Pazzani &amp; Billsus 1997</li> <li>• Billsus &amp; Pazzani 1999, 2000</li> <li>• Mooney et al. 1998</li> <li>• Mooney &amp; Roy 1999</li> <li>• Zhang et al. 2002</li> </ul>

Tabela 7: Pesquisa em Recomendação Colaborativa

Collaborativa	<p>Técnicas mais utilizadas:  Vizinho Mais Próximo (coseno, correlação)  Teoria dos Grafos</p> <p>Exemplos de Pesquisas:</p> <ul style="list-style-type: none"> <li>• Resnick et al. 1994</li> <li>• Hill et al. 1995</li> <li>• Shardanand &amp; Maes 1995</li> <li>• Breese et al. 1998</li> <li>• Nakamura &amp; Abe 1998</li> <li>• Aggarwal et al. 1999</li> <li>• Delgado &amp; Ishii 1999</li> <li>• Pennock &amp; Horwitz 1999</li> <li>• Sarwar et al. 2001</li> </ul>	<p>Técnicas mais utilizadas:  Redes Bayesianas  Agrupamento  Redes Neurais  Regressão Linear  Modelos Probabilísticos</p> <p>Exemplos de Pesquisas:</p> <ul style="list-style-type: none"> <li>• Billsus &amp; Pazzani 1998</li> <li>• Breese et al. 1998</li> <li>• Ungar &amp; Foster 1998</li> <li>• Chien &amp; George 1999</li> <li>• Getoor &amp; Sahami 1999</li> <li>• Pennock &amp; Horwitz 1999</li> <li>• Goldberg et al. 2001</li> <li>• Kumar et al. 2001</li> <li>• Pavlov &amp; Pennock 2002</li> <li>• Shani et al. 2002</li> <li>• Yu et al. 2002, 2004</li> <li>• Hofmann 2003, 2004</li> <li>• Marlin 2003</li> <li>• Si &amp; Jin 2003</li> </ul>
---------------	---	---

Tabela 8: Pesquisa em Recomendação Híbrida

Técnicas de Recomendação	Técnicas Aplicadas	
	Baseada em Heurística	Baseada em Modelo
Híbrida	<p>Combinando componentes baseados em conteúdo e colaborativos:  Combinação linear de avaliação previstas  Esquemas variados de votação  Incorporando um componente como parte da heurística de outro</p> <p>Exemplos de Pesquisas:</p> <ul style="list-style-type: none"> <li>• Balabanovic &amp; Shoham 1997</li> <li>• Claypool et al. 1999</li> <li>• Good et al. 1999</li> <li>• Pazzani 1999</li> <li>• Billsus &amp; Pazzani 2000</li> <li>• Tran &amp; Cohen 2000</li> <li>• Melville et al. 2002</li> </ul>	<p>Combinando componentes baseados em conteúdo e colaborativos:  Incorporando um componente como parte de um modelo em outro  Construindo um modelo unificado</p> <p>Exemplos de Pesquisas:</p> <ul style="list-style-type: none"> <li>• Basu et al. 1998</li> <li>• Condliff et al. 1999</li> <li>• Soboroff &amp; Nicholas 1999</li> <li>• Ansari et al. 2000</li> <li>• Popescul et al. 2001</li> <li>• Schein et al. 2002</li> </ul>

## 4 *Trabalhos Relacionados*

*“Nunca deixe de acreditar nos seus sonhos”*

Autor desconhecido

Este capítulo são apresentados alguns sistemas de recomendação para Bibliotecas Digitais, destacando-se os propósitos da recomendação realizada e analisando-se as estratégias de recomendação.

O sistema de recomendação proposto por Hwang em (S-Y.; W-C.; W-S., 2003) está inserido no projeto *Networked Digital Library Project* da *National Sun Yat-sen University* em Taiwan. Dentre os objetivos principais deste projeto, destaca-se o desenvolvimento de tecnologias que dão suporte a serviços digitais, sendo, uma das etapas, o desenvolvimento de um sistema de recomendação de literatura.

Para realizar o processo de recomendação, esse sistema emprega o uso de *logs* Web. O processo consiste de três passos principais:

1. Preparação dos dados dos *logs* de uso Web (mineração de uso na Web);
2. Descoberta de associações entre artigos;
3. Recomendação dos artigos.

Este sistema analisa o uso da literatura e gera recomendações classificadas de acordo com as preferências do usuário ativo no sistema. Para que a recomendação aconteça, várias características das publicações e das interações dos usuários com o sistema são levadas em conta. O perfil do usuário é estabelecido, levando-se em conta o conjunto de itens recentemente utilizados.

A abordagem de recomendação utilizada é a focada em tarefa (*task-focused approach*), a qual é representada por uma combinação de princípios da filtragem colaborativa e da mineração de dados (*data mining*) (HERLOCKER; KONSTAN, 2001). Neste sistema, primeiramente são montados *clusters* (aglomerados) de artigos acessados. Esses artigos, irão servir de base para a geração da recomendação, levando-se em conta o perfil do usuário (arquivos acessados por ele). Ou seja, esta abordagem é considerada uma variação da filtragem colaborativa, pois utiliza um perfil de usuário (obtido através de suas interações recentes no sistema), e realiza a combinação entre o perfil do usuário e os *clusters* (construídos com base no comportamento de acesso realizado por outros usuários). Nessa abordagem, não é necessário haver um “casamento” explícito entre perfis de usuários.

No trabalho de Huang (HUANG et al., 2002) é apresentado um modelo de grafo de duas camadas, utilizado na recomendação de livros. Esse modelo, possibilita a utilização das três abordagens: filtragem baseada em conteúdo, filtragem colaborativa e filtragem híbrida. Nesse sistema, são utilizadas informações sobre o conteúdo dos livros (*book*), informações demográficas sobre os clientes (*customer*), e seus históricos de compra. No método proposto é utilizado um grafo de duas camadas (*book layer, customer layer*) e

ainda, ligações entre as duas camadas, que representam o histórico de compra relacionando usuários e itens (*purchase history*).

O tipo de abordagem utilizado irá variar de acordo com os pesos de similaridade considerados para a predição dos itens a serem recomendados. Caso forem utilizados somente os pesos da similaridade *book-to-book* (camada *layer book*) tem-se uma abordagem baseada em conteúdo. Mas se forem utilizados somente os pesos de similaridade *customer-to-customer* (camada *customer layer*) e históricos de compras (*purchase histories*), para gerar a recomendação, então tem-se uma abordagem de filtragem colaborativa. É possível ainda combinar ambas as abordagens, através do uso de todos os pesos de associação e histórico de compras, resultando assim em uma abordagem híbrida.

Visando a pesquisa e o desenvolvimento de sistemas de recomendação para bibliotecas digitais, o projeto *Adaptative Recommendation Project* (ARP, 2001) desenvolveu um sistema de recomendação, chamado de *TalkMine*, para a biblioteca de pesquisa do *Los Alamos National Laboratory*.

Esse sistema disponibiliza para os usuários uma interface de busca para os documentos da biblioteca digital. Dependendo das pesquisas realizadas pelo usuário, são definidas diferentes "personalidades de busca" para este, com históricos **IR** (palavras-chave utilizadas na busca) diferentes e contextos de conhecimento independentes. Com isso, o algoritmo de recomendação integra o contexto de conhecimento da personalidade corrente do usuário, com os recursos de informação (documentos) buscados, possibilitando assim, recomendações apropriadas. Segundo (ROCHA, 2001), o sistema é implementado utilizando-se a filtragem baseada em conteúdo e a filtragem colaborativa, resultando assim em uma abordagem híbrida.

No trabalho de Cazella (CAZELLA; ALVARES, 2004) é proposto um sistema de recomendação multiagentes para artigos científicos. Nesse trabalho o currículo Lattes é utilizado como fonte de informações para a criação do perfil do usuário. Para o processo de recomendação, a estratégia adotada é filtragem híbrida. Inicialmente, a filtragem baseada em conteúdo é utilizada, analisando-se o conteúdo dos artigos e verificando se os mesmos são relevantes ao usuário, levando-se em conta o seu perfil. A filtragem colaborativa também é utilizada, nesse caso, grupos de usuários são formados, criando assim comunidades virtuais, onde os artigos relevantes para os usuários de uma certa comunidade serão recomendados para os outros usuários dentro dessa mesma comunidade.

No trabalho de Torres (TORRES et al., 2004) é proposto um Sistema de recomendação híbrido de artigos científicos que combina as técnicas de filtragem colaborativa e baseada

em conteúdo. As técnicas exploram o texto do artigo em si e a cadeia de citações, que liga artigos a outros artigos interessantes. Os algoritmos híbridos desenvolvidos seguem o modelo de *feature augmentation* e *mixed* da classificação de Burke (BURKE, 2002). O primeiro algoritmo utiliza as recomendações de uma técnica como entrada de uma segunda técnica, responsável pelas recomendações. O segundo algoritmo gera as recomendações de duas técnicas e combiná-las numa única lista de recomendação.

Ainda no trabalho de Torres, foi desenvolvido o sistema TechLens+ onde é possível obter uma opinião dos usuários sobre a qualidade dos algoritmos desenvolvidos. Para isso, esse sistema oferecia recomendações reais de artigos, com base nas preferências de cada usuário. O perfil do usuário é obtido de forma explícita através de um artigo de um determinado autor.

No trabalho de Giselle (LOPES, 2007), um Sistema de recomendação por conteúdo de artigos científicos, armazenados em bibliotecas digitais. Segundo a autora, o sistema proposto foi desenvolvido sob a perspectiva da Web Semântica, à medida que faz uso de suas tecnologias emergentes como: uso de metadados *Dublin Core*, uso de XML para a descrição do perfil através do Currículo Lattes e provedores de serviços e de dados (OAI) para a geração das recomendações. Nesse trabalho é utilizada a biblioteca digital da BDBComp - (Biblioteca Digital Brasileira de Computação).

## 5 *Sistema de Recomendação Proposto*

*“Os ignorantes, que acham que sabem tudo, privam-se de um dos maiores prazeres da vida: APRENDER.”*

Gustavo de Assis

O presente capítulo apresenta uma descrição do sistema de recomendação híbrido no contexto de bibliotecas digitais, dando conta da sua arquitetura.

## 5.1 Descrição do Sistema

O sistema de recomendação personalizada proposto neste trabalho tem como objetivo principal a recomendação de artigos científicos, tendo como base informações obtidas do currículo Lattes do usuário e avaliações dos usuários sobre um conjunto de artigos. O sistema foi pensado de maneira que qualquer biblioteca digital que provê metadados no formato Dublin Core (DC) e dá suporte ao protocolo OAI-PMH, pode ser utilizada como fonte para prover informações sobre os artigos a serem recomendados. Portanto, cabe salientar que os dados utilizados como fonte para o processo de recomendação consistem de:

- Informações do usuário, obtidas a partir do currículo Lattes em XML;
- Informações sobre os documentos digitais, obtidas através de metadados no formato Dublin Core codificados em XML;
- Avaliações (*ratings*) sobre os documentos.

O *Curriculum Vitae* (CV) da Plataforma Lattes oferece um padrão XML o qual é mantido pela comunidade CONSCIENTIAS (Comunidade para Ontologias em Ciência, Tecnologia e Informações de Aperfeiçoamento de Nível Superior). A gramática construída para tal padrão, na linguagem de esquema (*XML Schema*) do Consórcio W3C, pode ser obtida em (LMPL-CNPQ, 2008). Já o XML utilizado para descrever os documentos digitais, obtido através do processo de colheita, segue o *XML Schema* apresentado em (DC-OAI, 2008).

## 5.2 Arquitetura

Esta seção apresenta a arquitetura do sistema de recomendação proposto neste trabalho.

A Figura 6 mostra os módulos presentes na arquitetura.

O funcionamento do sistema proposto ocorrerá da seguinte maneira:

1. Primeiramente, o usuário irá se cadastrar no sistema, onde ele deverá fornecer um login, senha e o arquivo XML correspondente ao seu currículo Lattes.



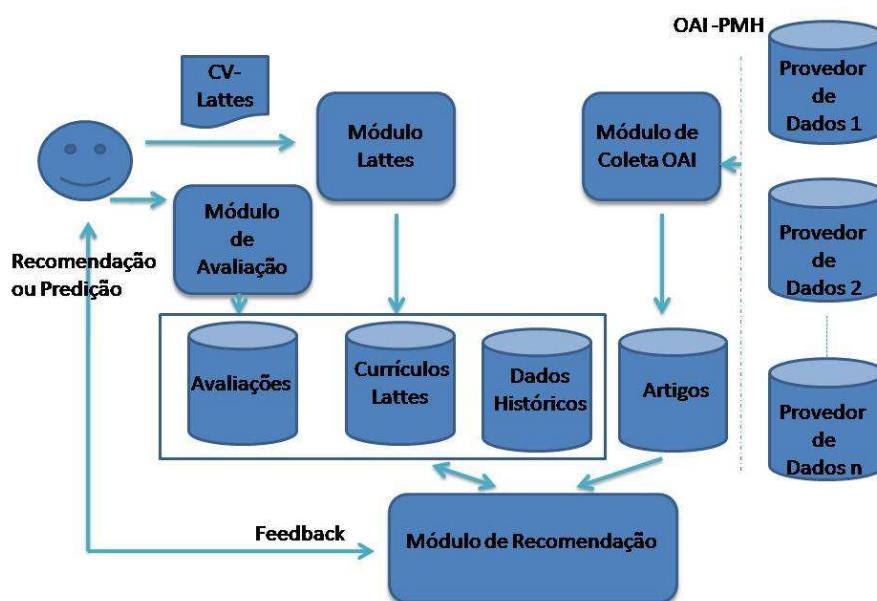


Figura 6: Arquitetura do Sistema de Recomendação Proposto

2. Em seguida, o **módulo Lattes** analisará o currículo Lattes do usuário e armazenará as informações relevantes na base de dados **Currículos Lattes**.
3. Após isto, o **módulo de Coleta OAI** enviará requisições OAI-PMH a um ou vários provedores de dados cadastrados no sistema, na tentativa de fazer colheita (*harvesting*) de metadados de documentos digitais específicos de cada Biblioteca Digital. Esse módulo receberá um documento XML como resposta. Nessa etapa, os metadados relevantes no formato Dublin Core serão extraídos do documento XML e armazenados na base de dados local **Artigos**.
4. Através do **módulo de Avaliação** o usuário fornecerá avaliações (ratings) numa escala de 1-5 aos artigos de suas áreas de interesse cadastradas no seu currículo lattes. Essas avaliações são armazenadas na base de dados **Avaliações**. Essas avaliações são utilizadas, posteriormente, no módulo de filtragem colaborativa.
5. A base de **Dados Históricos** armazenam informações sobre itens consultados ou adquiridos pelos usuários, evitando-se assim o envio de artigos que o usuário já tenha visto ou avaliado.
6. Para finalizar, o **módulo de Recomendação** é acionado e então a recomendação ocorrerá.

## 5.3 O módulo Lattes

Este módulo tem como função extrair informações relevantes dos pesquisadores e armazená-las na base de dados *Currículos Lattes*. O módulo recebe como entrada um arquivo xml, correspondente ao currículo do pesquisador.

Na Listagem 5.1 é apresentado um exemplo de um trecho de currículo lattes em XML de um determinado pesquisador. O documento inicia com o prólogo xml e nota-se a presença do *tag* raiz CURRICULO-VITAE.

Listagem 5.1: Exemplo de um arquivo XML do Currículo Lattes

```

1 <?xml version="1.0" encoding="ISO-8859-1" standalone="no" ?>
2 <CURRICULO-VITAE SISTEMA-ORIGEM-XML="LATTES-OFFLINE" DATA-ATUALIZACAO
   ="21082008" HORA-ATUALIZACAO="182841"
3     xmlns:lattes="http://www.cnpq.br/2001/XSL/Lattes">
4 <DADOS-GERAIS NOME-COMPLETO="Evandro de Barros Costa" NOME-EM-CITACOES-
   BIBLIOGRAFICAS="COSTA, E. B." NACIONALIDADE="B"
5     PAIS-DE-NASCIMENTO="Brasil" UF-NASCIMENTO="AL" CIDADE-NASCIMENTO="
   Maceió" DATA-NASCIMENTO="23111963"
6     SEXO="MASCULINO" NOME-DO-PAI="Manoel de Barros Costa" NOME-DA-MAE="
   Maria Barbosa Costa" PERMISSAO-DE-DIVULGACAO="NAO">
7 <RESUMO-CV TEXTO-RESUMO-CV-RH="Bacharel em Ciência da Computação pela
   Universidade Federal da Paraíba (1988), mestre em Engenharia Elétrica
   pela Universidade Federal da Paraíba (1989) e doutor em Engenharia
   Elétrica pela Universidade Federal da Paraíba (1997). Atualmente é
   professor associado da Universidade Federal de Alagoas, lotado no seu
   Instituto de Computação. Tem experiência na área de Ciência da
   Computação, com ênfase em Inteligência Artificial e engenharia de
   software baseada em agentes. Atua principalmente nos seguintes temas:
   sistemas multi-agentes, sistemas tutores inteligentes, representação
   de conhecimento e Web Semântica, agentes inteligentes, informática na
   educação e engenharia de software." />
8 <OUTRAS-INFORMACOES-RELEVANTES OUTRAS-INFORMACOES-RELEVANTES="" />
9 <ENDERECO FLAG-DE-PREFERENCIA="ENDERECO_RESIDENCIAL">
10 <ENDERECO-PROFISSIONAL CODIGO-INSTITUICAO-EMPRESA="033100000009" NOME-
   INSTITUICAO-EMPRESA="Universidade Federal de Alagoas" CODIGO-UNIDADE
   ="" NOME-UNIDADE="" CODIGO-ORGAO="" NOME-ORGAO="" PAIS="Brasil" UF="AL
   " LOGRADOURO-COMPLEMENTO="Campus A. C. Simões, S/N BR 104, Bloco 12"
   BAIRRO="Tabuleiro dos Martins" CIDADE="Maceio" CAIXA-POSTAL="" CEP
   ="57072-970" DDD="82" TELEFONE="32141401" RAMAL="" FAX="" E-MAIL="
   evandro@tci.ufal.br" HOME-PAGE="http://" />
11 </ENDERECO>

```

```
12 <FORMACAO-ACADEMICA-TITULACAO>
13 <GRADUACAO SEQUENCIA-FORMACAO="1" NIVEL="1" TITULO-DO-TRABALHO-DE-
    CONCLUSAO-DE-CURSO="" NOME-DO-ORIENTADOR="" CODIGO-INSTITUICAO
    ="008300000001" NOME-INSTITUICAO="Universidade Federal da Paraíba"
    CODIGO-CURSO="90000001" NOME-CURSO="Bacharelado Em Ciência da
    Computação" CODIGO-AREA-CURSO="10300007" STATUS-DO-CURSO="CONCLUIDO"
    ANO-DE-INICIO="1985" ANO-DE-CONCLUSAO="1988" FLAG-BOLSA="NAO" CODIGO-
    AGENCIA-FINANCIADORA="" NOME-AGENCIA="" NUMERO-ID-ORIENTADOR="" CODIGO
    -CURSO-CAPE=" " TITULO-DO-TRABALHO-DE-CONCLUSAO-DE-CURSO-INGLES=""
    NOME-CURSO-INGLES="" />
14 <MESTRADO SEQUENCIA-FORMACAO="2" NIVEL="3" CODIGO-INSTITUICAO
    ="008300000001" NOME-INSTITUICAO="Universidade
15 Federal da Paraíba" CODIGO-CURSO="22000097" NOME-CURSO="Engenharia
    Elétrica" CODIGO-AREA-CURSO="30400007" STATUS-DO-CURSO="CONCLUIDO"
16 ANO-DE-INICIO="1988" ANO-DE-CONCLUSAO="1989" FLAG-BOLSA="SIM" CODIGO-
    AGENCIA-FINANCIADORA="002200000000" NOME-AGENCIA="Conselho
17 Nacional de Desenvolvimento Científico e Tecnológico" ANO-DE-OBTENCAO-
    DO-TITULO="1989" TITULO-DA-DISSERTACAO-TESE="MOL3D:
18 Um Sistema Gráfico de Apoio à Visualização do Comportamento Espacial
    de Estruturas Moleculares" NOME-COMPLETO-DO-ORIENTADOR="Paulo
19 Roberto Campos de Araújo" NUMERO-ID-ORIENTADOR="" CODIGO-CURSO-CAPE="
    ="24009016003P8" TITULO-DA-DISSERTACAO-TESE-INGLES="" NOME-CURSO-
    INGLES="">
20 <AREAS-DO-CONHECIMENTO />
21 </MESTRADO>
22 <DOUTORADO SEQUENCIA-FORMACAO="3" NIVEL="4" CODIGO-INSTITUICAO
    ="008300000001" NOME-INSTITUICAO="Universidade
23 Federal da Paraíba" CODIGO-CURSO="22000097" NOME-CURSO="Engenharia
    Elétrica" CODIGO-AREA-CURSO="30400007" STATUS-DO-CURSO="CONCLUIDO"
24 ANO-DE-INICIO="1993" ANO-DE-CONCLUSAO="1997" FLAG-BOLSA="NAO" CODIGO-
    AGENCIA-FINANCIADORA="" NOME-AGENCIA=""
25 ANO-DE-OBTENCAO-DO-TITULO="1997" TITULO-DA-DISSERTACAO-TESE="Um Modelo
    de Ambiente Interativo de Aprendizagem Baseado numa
26 Arquitetura Multi-Agentes" NOME-COMPLETO-DO-ORIENTADOR="Angelo
    Perkusich e Edilson Ferneda" TIPO-DOUTORADO=""
27 CODIGO-INSTITUICAO-DOUT="" NOME-INSTITUICAO-DOUT="" CODIGO-INSTITUICAO
    -OUTRA-DOUT="" NOME-INSTITUICAO-OUTRA-DOUT=""
28 NOME-ORIENTADOR-DOUT="" NUMERO-ID-ORIENTADOR="" CODIGO-CURSO-CAPE="
    ="24009016003P8" TITULO-DA-DISSERTACAO-TESE-INGLES="" NOME-CURSO-
    INGLES="">
29 <AREAS-DO-CONHECIMENTO />
30 </DOUTORADO>
31 </FORMACAO-ACADEMICA-TITULACAO>
```

```

32         . . . .
33         . . . .
34         . . . .
35 </DADOS-GERAIS>
36 </CURRICULO-VITAE>

```

A Listagem acima mostra o currículo do professor Evandro de Barros Costa que possui graduação em Ciência da Computação e, mestrado e doutorado em Engenharia Elétrica pela Universidade Federal da Paraíba. Além dessas existem informações de artigos publicados, palavras-chaves, área de interesse. Portanto, o Lattes é uma excelente fonte de informação para se obter dados para a construção do perfil do usuário.

O Exemplo acima deve obedecer a uma gramática específica para ser considerado um currículo lattes. Um trecho da DTD do currículo lattes esta descrita logo abaixo:

Listagem 5.2: Ontologia XML para a Unidade de Informação de Currículos

```

1 <?xml version="1.0" encoding="ISO-8859-1"?>
2
3 <!ELEMENT CURRICULO-VITAE (DADOS-GERAIS, PRODUCAO-BIBLIOGRAFICA?,
4 PRODUCAO-TECNICA?, OUTRA-PRODUCAO?, DADOS-COMPLEMENTARES?)>
5 <!ATTLIST CURRICULO-VITAE
6     SISTEMA-ORIGEM-XML CDATA #REQUIRED
7     NUMERO-IDENTIFICADOR CDATA #IMPLIED
8     FORMATO-DATA-ATUALIZACAO NMTOKEN #FIXED "DDMMAAAA"
9     DATA-ATUALIZACAO CDATA #IMPLIED
10    FORMATO-HORA-ATUALIZACAO NMTOKEN #FIXED "HHMMSS"
11    HORA-ATUALIZACAO CDATA #IMPLIED
12    xmlns:lattes CDATA #IMPLIED
13 >
14 <!ELEMENT DADOS-GERAIS (RESUMO-CV?,OUTRAS-INFORMACOES-RELEVANTES? ,
15 ENDERECO?, FORMACAO-ACADEMICA-TITULACAO?, ATUACOES-PROFISSIONAIS?,
16 AREAS-DE-ATUACAO?, IDIOMAS?, PREMIOS-TITULOS?)> <!ATTLIST
17 DADOS-GERAIS
18     NOME-COMPLETO CDATA #REQUIRED
19     NOME-EM-CITACOES-BIBLIOGRAFICAS CDATA #REQUIRED
20     NACIONALIDADE CDATA #REQUIRED
21     CPF CDATA #IMPLIED
22     NUMERO-DO-PASSAPORTE CDATA #IMPLIED
23     PAIS-DE-NASCIMENTO CDATA #IMPLIED
24     UF-NASCIMENTO CDATA #IMPLIED
25     CIDADE-NASCIMENTO CDATA #IMPLIED
26     FORMATO-DATA-DE-NASCIMENTO NMTOKEN #FIXED "DDMMAAAA"

```

```
27 DATA-NASCIMENTO CDATA #IMPLIED
28 SEXO (MASCULINO | FEMININO) #REQUIRED
29 NUMERO-IDENTIDADE CDATA #IMPLIED
30 ORGAO-EMISSOR CDATA #IMPLIED
31 UF-ORGAO-EMISSOR CDATA #IMPLIED
32 FORMATO-DATA-DE-EMISSAO NMTOKEN #FIXED "DDMMAAAA"
33 DATA-DE-EMISSAO CDATA #IMPLIED
34 NOME-DO-PAI CDATA #IMPLIED
35 NOME-DA-MAE CDATA #IMPLIED
36 PERMISSAO-DE-DIVULGACAO (SIM | NAO) #REQUIRED
37 NOME-DO-ARQUIVO-DE-FOTO CDATA #IMPLIED
38 TEXTO-RESUMO-CV-RH CDATA #IMPLIED
39 OUTRAS-INFORMACOES-RELEVANTES CDATA #IMPLIED
40 >
```

Conforme descrito anteriormente, a raiz da árvore é o elemento CURRICULO-VITAE. Analisando o trecho, observa-se que um CURRICULO-VITAE é composto por dados gerais, produção bibliográfica, produção técnica, outras produções e dados complementares. O único item obrigatório são os dados gerais. Para um melhor entendimento do trecho acima consultar [www.w3.org/XML](http://www.w3.org/XML) .

## 5.4 Processo de Obtenção dos Metadados dos Documentos

Para que seja realizado o processo de recomendação, primeiramente é necessário obter os metadados que descrevem os arquivos que serão recomendados. Tal funcionalidade, é realizada pelo módulo *Coleta OAI*, o qual é o *harvester* do provedor de serviços. Este módulo envia requisições OAI-PMH para realizar a colheita de metadados de provedores de dados de Bibliotecas Digitais.

Para ser realizada a coleta dos metadados de uma biblioteca digital é necessário saber a URL do provedor de dados que disponibiliza os metadados. Após a obtenção da URL base utiliza-se os set verbos OAI-PMH vistos na seção 2.4 . Três URLs de diferentes provedores de dados são mostradas na tabela 9.

Tabela 9: URLs Base de alguns provedores de dados

Nome do Repositório	URL Base
<i>BDBComp</i>	http://www.lbd.dcc.ufmg.br/cgi-bin/bdbcomp/oai2/oai.pl
<i>Citeseer</i>	http://cs1.ist.psu.edu/cgi-bin/oai.cgi
<i>BDTD Ibict</i>	http://dataproducer.ibict.br/myoai/oai2.php

Um exemplo de uma requisição realizada pelo *Módulo de Coleta OAI* seria:

- [http://cs1.ist.psu.edu/cgi-bin/oai.cgi?verb=ListRecords&metadataPrefix=oai\\_dc](http://cs1.ist.psu.edu/cgi-bin/oai.cgi?verb=ListRecords&metadataPrefix=oai_dc)

Esta requisição solicita os metadados no formato Dublin Core, que descrevem os artigos publicados no *Citeseer*. Neste exemplo são retornados os primeiros mil artigos expostos pelo provedor de dados.

Como resposta, o *harvester* do provedor de serviços receberá um arquivo XML como o apresentado no Código 5.3. Ao receber um arquivo XML como resposta, o módulo *Coleta OAI* é acionado. Esse módulo é encarregado de interpretar os arquivos XML, recebidos pelo *harvester*, e armazenar as informações que são utilizadas durante o processo de recomendação na base de dados *Artigos*.

Listagem 5.3: Exemplo de um arquivo XML no formato Dublin Core

```

1  <?xml version="1.0" encoding="ISO-8859-1" ?>
2  - <OAI-PMH
3  xmlns="http://www.openarchives.org/OAI/2.0/" xmlns:xsi="http://www.w3.org
   /2001/XMLSchema-instance"
4     xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/http://www.
   openarchives.org/OAI/2.0/OAI-PMH.xsd">
5     <responseDate>2008-07-15T10:38:53Z</responseDate>
6     <request verb="ListRecords" metadataPrefix="oai_dc">http://cs1.ist.psu.
   edu/cgi-bin/oai.cgi</request>
7  - <ListRecords>
8  - <record>
9  - <header>
10     <identifier>oai:CiteSeerPSU:1</identifier>
11     <datestamp>1993-08-11</datestamp>
12     <setSpec>CiteSeerPSUset</setSpec>
13     </header>
14  - <metadata>
15  - <oai_dc:dc xmlns:oai_dc="http://www.openarchives.org/OAI/2.0/oai_dc/"
16     xmlns:dc="http://purl.org/dc/elements/1.1/"

```

```

17   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
18   xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/oai_dc/
19   http://www.openarchives.org/OAI/2.0/oai_dc.xsd">
20   <dc:title>36 Problems for Semantic Interpretation</dc:title>
21   <dc:creator>Gabriele Scheler</dc:creator>
22   <dc:subject>Gabriele Scheler 36 Problems for Semantic Interpretation</dc:
23       subject>
24   <dc:description>This paper presents a collection of problems
25       for natural language analysis derived mainly
26       from theoretical linguistics. Most of these
27       problems present major obstacles for computational
28       systems of language interpretation.
29       .....
30       .....
31   </dc:description>
32   <dc:contributor>The Pennsylvania State University CiteSeer Archives</dc:
33       contributor>
34   <dc:publisher>unknown</dc:publisher>
35   <dc:date>1993-08-11</dc:date>
36   <dc:format>ps</dc:format>
37   <dc:identifier>http://citeseer.ist.psu.edu/1.html</dc:identifier>
38   <dc:source>ftp://flop.informatik.tu-muenchen.de/pub/fki/fki-179-93.ps.gz
39       </dc:source>
40   <dc:language>en</dc:language>
41   <dc:rights>unrestricted</dc:rights>
42   </oai_dc:dc>
43   </metadata>
44   </record>
45   .....
46   .....
47   <resumptionToken>!!!1001!oai_dc</resumptionToken>
48   </ListRecords>
49   </OAI-PMH>

```

A Listagem acima mostra um trecho da resposta a requisição feita ao *CiteSeer*. O *tag* raiz é OAI-PMH que indica que é um resposta a uma requisição OAI-PMH. Alguns atributos como a data da resposta, o verbo utilizado na requisição e o tipo dos metadados são mostrados. Cada registro é indicado com um campo *record* e cada record contém os campos no formato *Dublin Core*. O *tag resumptionToken* indica que ainda existem registros a serem obtidos e que a próxima requisição deve utilizá-lo.

## 5.5 O módulo de Avaliação

Este módulo tem a função de armazenar avaliações (*ratings*) dos usuários sobre os artigos vistos pelos mesmos. Os *ratings* são armazenados na base avaliações. Cada usuário avalia um artigo utilizando-se uma escala de zero a cinco pontos. Um artigo que tem a mínima contribuição para o usuário é avaliado como zero e um artigo excelente é avaliado com nota cinco.

As avaliações são armazenadas para serem utilizadas posteriormente, pelo módulo de filtragem colaborativa. Os Usuários não avaliam artigos em que os mesmos são autores ou co-autores. Inicialmente, os artigos são sugeridos com base na área de interesse dos usuários.

## 5.6 O módulo de Recomendação

O módulo de recomendação possui a função de recomendar artigos aos pesquisadores baseado no perfil de cada um deles. Este módulo utiliza um algoritmo híbrido para a recomendação dos artigos. O algoritmo híbrido é composto por dois tipos de técnicas:

- Uma baseada em filtragem colaborativa
- Uma baseada em filtragem por conteúdo

Antes de entrarmos em detalhes sobre o algoritmo híbrido utilizado, mostraremos cada abordagem em separado.

### 5.6.1 Filtragem Colaborativa - FC

A técnica colaborativa aplica o típico algoritmo baseado em vizinhança utilizado em (MILLER, 1995). Esta técnica é dividida em três passos:

1. Medir a similaridade entre os usuários;
2. Criar vizinhanças;
3. Computar predições e gerar recomendações.

No primeiro passo, a correlação de Pearson é utilizada



$$W_{a,u} = \frac{\sum_{i=1}^m (r_{a,i} - \bar{r}_a) * (r_{u,i} - \bar{r}_u)}{\sqrt{\sum_{i=1}^m (r_{a,i} - \bar{r}_a)^2} * \sqrt{\sum_{i=1}^m (r_{u,i} - \bar{r}_u)^2}}$$

Na equação,  $W_{a,u}$  é a correlação do usuário ativo  $a$  com um determinado usuário  $u$ ,  $r_{a,i}$  é a avaliação que o usuário ativo deu ao item  $i$ , e  $\bar{r}_a$  é a média de todas as avaliações do usuário ativo. Ao calcular a similaridade entre dois usuários, todas as somas e médias na fórmula são computadas somente para os itens em que ambos os usuários avaliaram.

A vizinhança é determinada com base no número de vizinhos. Neste processo, vizinhos com correlação positiva são selecionados. Com a intenção de melhorar a precisão das recomendações, a predição para um artigo é produzida se o número de vizinhos for no mínimo cinco.

A predição pode ser estabelecida por meio de uma média ponderada das avaliações dos vizinhos usando a fórmula abaixo:

$$P_{a,i} = \bar{r}_a + \frac{\sum_{u=1}^n (r_{u,i} - \bar{r}_u) * W_{a,u}}{\sum_{u=1}^n |W_{a,u}|}$$

A fórmula acima mede como cada vizinho avalia o item e pondera utilizando sua similaridade com o usuário ativo.

### 5.6.2 Filtragem Baseada em Conteúdo - FBC

Para a recomendação baseada em conteúdo precisamos de um modelo para representar os artigos científicos e uma forma de consultá-los. Portanto, foi utilizado o modelo vetorial (SALTON; BUCKLEY, 1988).

Antes de qualquer cálculo de similaridade, aos textos são aplicados um pré-processamento em que são aplicadas duas técnicas:

- Eliminação de *stopwords*
- *stemming*

Na eliminação de *stopwords* são retiradas do texto palavras que não são importantes na análise, por exemplo artigos e preposições.

O processo de *stemming* consiste em extrair o radical das palavras e utilizá-lo no momento do cálculo de similaridade.

Cada artigo presente na base de dados é representado por um vetor de termos. As informações que são utilizadas para compor o vetor de termos são os campos:

- dc:title
- dc:description

O campo *title* contém o título do artigo enquanto que o campo *description* armazena um resumo sobre o artigo.

Para determinar a similaridade entre os artigos é utilizado o TF-IDF (*Term-Frequency Inverse-Document-Frequency*). O TF-IDF possui duas componentes importantes devem ser calculadas: o IDF (Inverse Document Frequency), o qual descreve o quanto uma palavra é discriminante dentro de uma coleção de documentos e o TF, o qual exprime o número de frequência de um termo em cada documento. Portanto, cada palavra possui um único valor de IDF para toda coleção de documentos e possivelmente valores diferentes de TF para esta mesma coleção.

O IDF pode ser calculado da seguinte forma:

Considere  $D$  como sendo o número total de documentos e  $N_p$  o número de documentos da coleção em que a palavra  $p$  ocorreu.

$$idf(p) = \lg \frac{D}{N_p}$$

O peso de cada termo na coleção de documentos é calculado multiplicando-se o TF pelo IDF.

O vetor de consulta é construído com base dos artigos publicados pelo pesquisador. O processo de construção do vetor é realizado através de numa adaptação do procedimento da dissertação de (LOPES, 2007). Os atributos considerados para o vetor de consulta são:

- O título de cada artigo publicado.
- As palavras chaves de cada artigo publicado.
- O ano em que o artigo foi publicado.

Considera-se também que termos de artigos que foram publicados recentemente são mais importantes que termos de artigos publicados há mais tempo, já que artigos mais recentes descrevem melhor a preferência do pesquisador. Para esta dissertação, termos presentes no título e nas palavras-chave possuem pesos diferentes. Um termo presente no título têm mais importância do que os termos presentes nas palavras-chave. O peso do título é calculado utilizando-se a seguinte fórmula:

- peso = pesoAno \* pesoTitulo;

O peso das palavras-chave é calculado utilizando-se a seguinte fórmula:

- peso = pesoAno \* pesoPalavraChave;

Nas fórmula acima o peso do ano é calculado pela seguinte fórmula:

- pesoAno = 1 - (anoAtual - anoArtigo)\*(0.45/(anoAtual - menorAno))

Os valores do peso do título e das palavras-chave foram calculados com base no trabalho da dissertação de (LOPES, 2007), resultando num peso=1 para o título = 1 e um peso=0.95 para as palavras-chave.

Após ter ser obtida a representação do documento e do vetor de consulta, utiliza-se o cálculo do cosseno entre eles para medir a similaridade.

$$\cos(\vec{a}, \vec{u}) = \frac{\vec{a} \bullet \vec{u}}{\|\vec{a}\| * \|\vec{u}\|}$$

Na equação,  $\vec{a}$  e  $\vec{u}$  são os vetores do perfil do usuário  $u$  e do artigos usuários  $a$ , respectivamente.

O símbolo  $\|\vec{a}\|$  indica a norma do vetor  $\vec{a}$ .

### 5.6.3 Construção do Perfil do Usuário

O processo de construção do perfil do usuário se dá através da obtenção de informações contidas em seu currículo Lattes mais o conjunto de artigos avaliados <sup>1</sup>. A plataforma Lattes foi utilizada nesse trabalho porque oferece uma base de dados padrão dos currículos dos pesquisadores e acadêmicos no Brasil.

A Tabela 5.6.3 apresenta o subconjunto de elementos de metadados do currículo Lattes que foram utilizados para a construção do perfil.

Apesar do currículo Lattes conter informações relevantes para o processo de criação do perfil do usuário, acredita-se que o mesmo pode não refletir, em um dado momento, os interesses atuais do usuário. Afinal, há sempre alguém com um currículo Lattes que

<sup>1</sup>O currículo Lattes é uma iniciativa do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq)

Tabela 10: Atributos do currículo lattes

<i>Nome-Completo</i>
<i>Área de Atuação</i>
<i>Artigo: título</i>
<i>Artigo: Palavras-Chaves</i>
<i>Artigo: Ano</i>

Tabela 11: Perfil inicial do usuário

<i>Termo</i>	<i>Peso</i>
Web Semântica	0,85
Sistemas Recomendação	0,55
Sistemas Multiagentes	0,45

contém informações desatualizadas ou que não façam mais parte das suas preferências atuais. Pensando nisso, o sistema proposto dará ao usuário a possibilidade de refinar o seu perfil. Para isso, o sistema apresentará ao usuário o perfil criado e permitirá que esse usuário modifique os termos e pesos presentes em seu perfil e/ou eventualmente adicione novos termos. Por exemplo, imagine que o perfil do usuário seja composto pelos termos apresentados na Tabela 11. Observe que esses termos foram obtidos do currículo Lattes do usuário. Vamos supor que o usuário não possua mais interesse em *Sistemas de Recomendação*, que agora ele está interessado em *Redes Neurais* e que seu interesse por *Sistemas Multiagentes* tenha aumentado. Então, o usuário poderia manipular seu perfil e resultar com um perfil parecido com o da Tabela 12. Observe que neste caso, o usuário retirou o termo *Sistemas Recomendação*, adicionou o termo *Redes Neurais* e aumentou o peso do termo *Sistemas Multiagentes*. Com isso, o novo perfil do usuário está representando melhor os seus reais interesses. Portanto, o perfil construído a partir do currículo Lattes pode ser modificado pelo usuário, visando provê-lo de informações mais atualizadas e fidedignas às preferências atuais desse usuário.

Tabela 12: Perfil modificado pelo usuário

<i>Termo</i>	<i>Peso</i>
Web Semântica	0,85
Redes Neurais	0,85
Sistemas Multiagentes	0,8

### 5.6.4 Recomendação Híbrida

Segundo (BURKE, 2002), os sistemas de recomendação híbridos combinam duas ou mais técnicas de recomendação com o objetivo de melhorar a performance e evitar as limitações apresentadas em sistemas que aplicam apenas uma abordagem. Nesta dissertação combinamos a filtragem colaborativa e a filtragem baseada em conteúdo.

São dois os algoritmos utilizados no nosso sistema:

- O algoritmo Misto
- O algoritmo Ponderado

O algoritmo misto implementa os métodos colaborativo e baseado em conteúdo separadamente e combina as predições geradas por ambos: desta forma é possível combinar as avaliações obtidas individualmente em cada um dos métodos para oferecer uma recomendação final.

O Funcionamento do algoritmo misto utilizado é mostrado na figura 7 . Os dois



Figura 7: Algoritmo Misto

algoritmos são executados em paralelo e o resultado final é obtido juntando as listas dos módulos colaborativo e baseado em conteúdo. Para combinar as recomendações de cada técnica, cada item presente nas listas recebe um valor correspondente a sua posição. A lista final é gerada da forma: cada item presente em ambas as listas é adicionado a lista final com uma pontuação. Essa pontuação é a soma de suas posições nas listas de recomendações originais. A lista é, então, ordenada de forma crescente, onde quanto menor a pontuação de um artigo, mais próximo do início ele está. Para os artigos que não estiverem presentes em ambas as listas, serão inseridos na lista final intercalados um a um, começando pela lista gerada por filtragem colaborativa.

Por exemplo, se FBC recomenda  $L1 = (A,B,D,H,K)$  e FC recomenda  $L2 = (D,P,K,J)$ , a lista final de recomendação será  $LF = (D,K,P,A,J,B,H)$ . O artigo D é o primeiro, pois sua pontuação ( $3 = 2+1$ ) é a menor dos artigos presentes na lista.

O algoritmo misto utilizado é baseado algoritmo Fusion do trabalho de Torres (TORRES, 2004).

No segundo algoritmo, o ponderado, o peso de um determinado item recomendado é computado dos resultados de todas as técnicas disponíveis de recomendação atuais do sistema. Este híbrido combina os pesos de cada componente usando uma fórmula linear.

O Funcionamento do algoritmo ponderado utilizado é mostrado na figura 8 .



Figura 8: Algoritmo Ponderado

Por exemplo, sendo dois o número de recomendadores de um recomendador híbrido ponderado eles podem ser combinados da seguinte forma:

$$RHP(a1) = p * RX(a1) + (1-p) * RY(a1), \text{ onde:}$$

- $RX(a1)$  - é a predição para o artigo  $a1$  computada pelo recomendador X.
- $RY(a1)$  - é a predição para o artigo  $a1$  computada pelo recomendador Y.
- $RHP(a1)$  - é a predição para o artigo  $a1$  computada pelo recomendador híbrido ponderado.
- $p$  - peso associado ao recomendador.

Um exemplo do cálculo realizado pelo recomendador híbrido ponderado é mostrado na figura 9.

Artigo	Rank1	Pontuação1	Pontuação2	Rank2	peso	1-peso	Pontuação Computada	Rank Híbrido
					0,4	0,6		
a	1	0,9	0,5	6			0,66	2
b	2	0,7	0,6	4			0,64	3
c	3	0,65	0,95	1			0,83	1
d	4	0,6	0,58	5			0,59	4
f	5	0,4	0,46	7			0,44	6
g	6	0,2	0,3	8			0,26	8
h	7	0,1	0,88	2			0,57	5
i	8	0,04	0,1	10			0,08	10
l	9	0,03	0,66	3			0,41	7
m	10	0,02	0,23	9			0,15	9

Figura 9: Exemplo de Cálculo Realizado pelo Algoritmo Ponderado

Os ranks 1 e 2 são os ranks do recomendador1 e do recomendador2, embora não mostrado na figura 9, podem existir artigos do recomendador1 ou do recomendador2 que não tenham pontuações.

A pontuação híbrida computada para o artigo **a** é :

- $RHP(a) = p \cdot R1(a) + (1-p) \cdot R2(a)$
- $RHP(a) = 0,4 * 0,9 + 0,6 * 0,5 = 0,66$

Este cálculo é realizado para todos os artigos e em seguida, os resultados são colocados em ordem decrescente.

## 6 *Implementação*

*“Nosso caráter é resultado de nossa conduta.”*

Aristóteles

No presente capítulo serão discutidos os principais aspectos de implementação e as ferramentas utilizadas para construção de cada módulo do sistema.



## 6.1 Introdução

Nesta seção serão descritos os aspectos de implementação do sistema de recomendação híbrido com o objetivo de gerar recomendações de artigos científicos de bibliotecas digitais compatíveis com o protocolo OAI-PMH.

A linguagem de programação adotada para o desenvolvimento do sistema teve como critérios robustez, confiabilidade e, principalmente, a portabilidade necessária para o desenvolvimento baseado na Web. Dessa maneira, obedecendo ao critério definido, a tecnologia Java foi escolhida como a mais adequada para a solução proposta, devido ao fato de ser robusta e proporcionar segurança, confiabilidade e escalabilidade.

A escolha do SGBD foi o MySQL, pois possui todas as características necessárias para um ambiente de desenvolvimento Web além do suporte dado pela comunidade brasileira e pelo fato do software ser gratuito.

A seguir será descrita a implementação dos principais módulos do sistema.

## 6.2 Módulo Lattes

Conforme descrito anteriormente, este módulo tem a função de extrair as informações do currículo lattes em XML dos pesquisadores e armazená-las na base de dados Currículos Lattes. Para que isto aconteça é necessária a construção de um parser XML.

Neste módulo foi utilizado o *Digester* da *Apache Software Foundation* que mapeia um arquivo em XML para objetos Java. Para maiores informações sobre o Digester consultar a URL <http://commons.apache.org/digester/>.

Para o mapeamento foi utilizado o Hibernate. O Hibernate é uma ferramenta que faz o mapeamento objeto/relacional<sup>1</sup> no ambiente Java. O Hibernate foi utilizado pois fornece facilidade de consultas e recuperação de dados, podendo também reduzir significativamente o tempo de desenvolvimento gasto com a manipulação manual de dados no SQL e JDBC.

A Figura 10 mostra a arquitetura do módulo Lattes.

O currículo lattes em formato XML é fornecido como entrada, em seguida, este

---

<sup>1</sup>O termo de mapeamento de objeto/relacional (ou ORM *Object/Relational Mapping*) se refere a técnica de mapear uma representação de dados de um modelo de objeto para dados de modelo relacional com o esquema baseado em SQL.

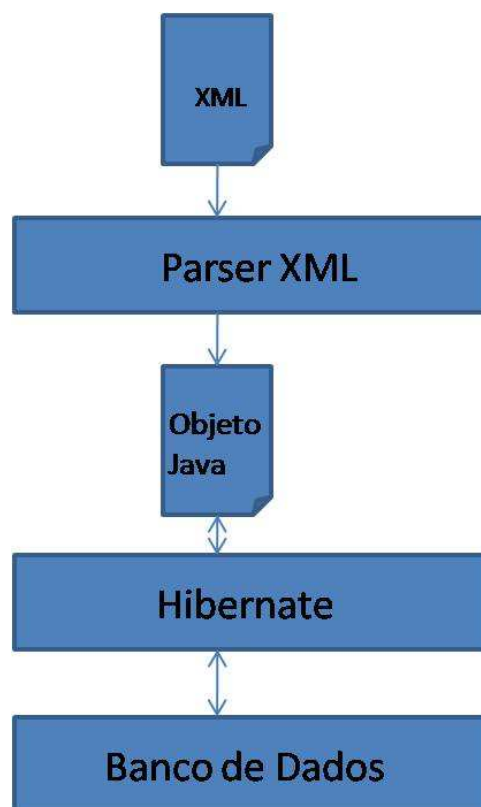


Figura 10: Arquitetura do Módulo Lattes

currículo é submetido a um parser XML que transforma o currículo xml em um objeto Java, após isto este objeto Java é armazenado numa base de dados MYSQL utilizando o Hibernate.

Um pequeno trecho de código da inserção do artigos publicados num determinado currículo é mostrado na listagem 6.1

Listagem 6.1: Trecho de Código do Parse XML

```
1
2 import java.io.File;
3
4 import org.hibernate.Session;
5 import org.hibernate.SessionFactory;
6 import org.hibernate.Transaction;
7
8 import org.hibernate.cfg.AnnotationConfiguration;
9
10 import org.hibernate.cfg.Configuration;
11
12 import br.ufal.ic.parser.CurriculoVitaeXmlParser;
13
```

```
14 import br.ufal.ic.parser.bean.CurriculoVitae;
15
16 import br.ufal.ic.parser.bean.producaobibliografica.ArtigoPublicado;
17
18 public class Teste {
19
20     public static void main(String[] args) throws Exception {
21         Configuration cfg = new AnnotationConfiguration();
22         cfg.configure();
23         SessionFactory sessionFactory = cfg.buildSessionFactory();
24         Session session = sessionFactory.openSession();
25         Transaction tx = session.beginTransaction();
26         CurriculoVitae curriculoVitae = new CurriculoVitaeXmlParser().parse
            (new File("xml/8338908425321389.xml"));
27         session.save(curriculoVitae);
28         for (ArtigoPublicado artigoPublicado : curriculoVitae.
            getProducaoBibliografica().getArtigosPublicados()) {
29             CurriculoVitae curriculoVitae2 = new CurriculoVitae();
30             curriculoVitae2.setId(curriculoVitae.getId());
31             artigoPublicado.setCurriculoVitae(curriculoVitae);
32             session.save(artigoPublicado);
33         }
34         // System.out.println(session.createCriteria(CurriculoVitaeXmlParser.
            class).list());
35         tx.commit();
36         session.close();
37     }
38
39 }
```

No código acima foram armazenados no banco de dados os atributos do Currículo Lattes(dados gerais) e os dados dos artigos publicados(título, idioma, palavras chave) do currículo lattes de cada pesquisador.

## 6.3 Módulo de Coleta OAI

O módulo de Coleta OAI recebe como entrada um arquivo correspondente ao da Listagem 5.3 para ser processado e armazena os metadados de cada artigo no banco de dados.

A Figura 11 mostra a arquitetura do módulo de Coleta OAI.

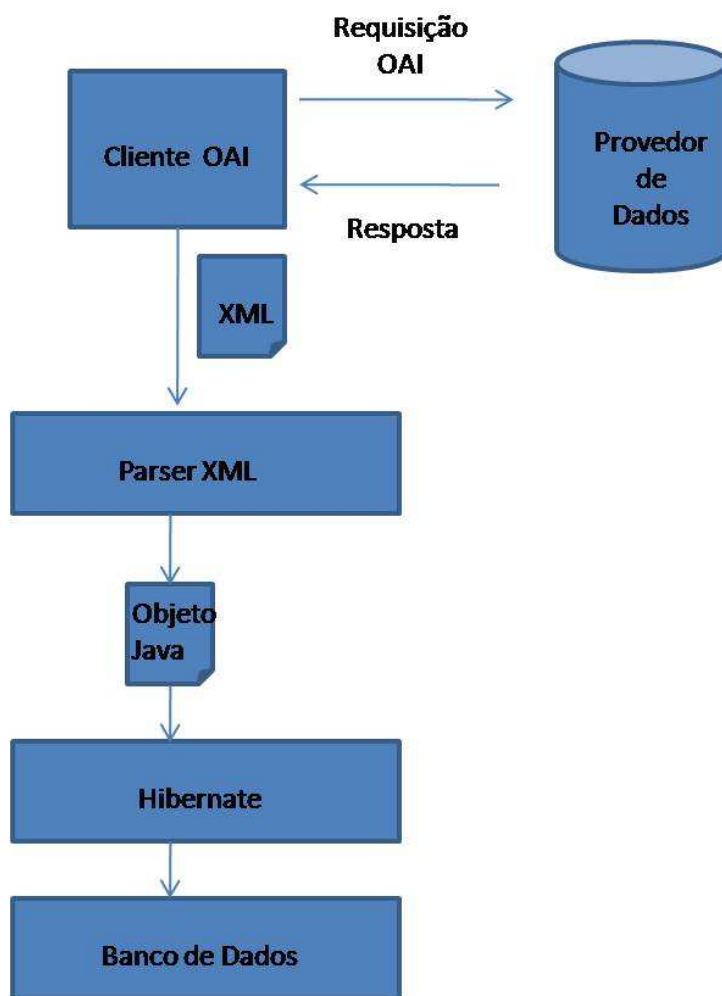


Figura 11: Arquitetura do Módulo de Coleta OAI

O cliente faz uma requisição a um provedor de dados específico, em seguida é retornada uma resposta em XML contendo a lista de arquivos do provedor. A resposta XML é armazenada num arquivo. Após isto, o arquivo é submetido a um parser XML que extrai uma lista dos registros contendo metadados de cada artigo. A saída do parser XML é um objeto Java que contém os artigos armazenados numa estrutura de dados. Finalmente, este objeto Java é armazenado numa base de dados MYSQL utilizando o Hibernate.

Um pequeno trecho de código do parser que recebe um arquivo XML com o conjunto de metadados é mostrado na listagem 6.2

Listagem 6.2: Trecho de Código do Parser XML dos Metadados

```

1
2 public Document getDocument() {
3     Document d = null;
4     SAXBuilder builder = new SAXBuilder();
5     try {
  
```

```
6         d = builder.build(new File("formatociteseer.xml"));
7     } catch (JDOMException e) {
8         e.printStackTrace();
9     } catch (IOException e) {
10        e.printStackTrace();
11    }
12    return d;
13 }
14
15 public void mapping() {
16     Document d = this.getDocument();
17     Element rootElement = d.getRootElement();
18     List<Element> rootChildren = rootElement.getChildren();
19     for(Element children : rootChildren) {
20         map = new Hashtable<String, String>();
21         System.out.println("rootElement: " + rootElement.getName());
22         List<Element> children1 = children.getChildren();
23         for(Element child : children1) {
24             if(child.getText() != null) {
25                 this.map.put("set"+child.getName(), child.getText());
26             }
27             List<Element> newChild = child.getChildren();
28             for(Element newChild1 : newChild) {
29                 if((newChild1.getText() != null) && (newChild1.
30                     getAttributeValue("name") != null)) {
31                     this.map.put("set"+newChild1.getName(),
32                         removeSpaces(newChild1.getText()+newChild1.
33                             getAttributeValue("name")));
34                 } else {
35                     if(newChild1.getName().equals("identifier") &&
36                         newChild1.getNamespacePrefix().equals("dc")) {
37                         this.map.put("set"+newChild1.getName()+"1",
38                             newChild1.getText().trim());
39                     } else {
40                         this.map.put("set"+newChild1.getName(),
41                             newChild1.getText());
42                     }
43                 }
44             }
45             List<Element> newChild2 = newChild1.getChildren();
46             for(Element lastChild : newChild2) {
47                 if(lastChild.getText() != null) {
48                     this.map.put("set"+lastChild.getName(),
49                         lastChild.getText());
50                 }
51             }
52         }
53     }
54 }
```

```
42         }
43     }
44 }
45 }
46 try {
47     this.save();
48 } catch (IllegalArgumentException e) {
49     // TODO Auto-generated catch block
50     e.printStackTrace();
51 } catch (IllegalAccessException e) {
52     // TODO Auto-generated catch block
53     e.printStackTrace();
54 } catch (InvocationTargetException e) {
55     // TODO Auto-generated catch block
56     e.printStackTrace();
57 } catch (InstantiationException e) {
58     // TODO Auto-generated catch block
59     e.printStackTrace();
60 } catch (ClassNotFoundException e) {
61     // TODO Auto-generated catch block
62     e.printStackTrace();
63 }
64 } // FOR
65
66 }
```

Após a execução do referido código todos os dados do padrão *Dublin Core* de cada artigo são armazenados na base de dados Artigos.

## 6.4 Módulo de Recomendação utilizando Filtragem Colaborativa

Na implementação do algoritmo de filtragem colaborativa foi utilizado, devido a simplicidade e rapidez nas respostas, o engenho Taste (OWEN, 2007).

Neste algoritmo a base de Avaliações é utilizada e uma matriz Usuários X Artigos é gerada. Cada célula da matriz contém a informação sobre como um determinado usuário U avaliou um determinado artigo A.

A listagem abaixo mostra o código do Algoritmo Usuário-Usuário:

Listagem 6.3: Trecho de Código da Filtragem Colaborativa

```
1      DataModel model = new MySQLJDBCDataModel( dataSource , "
2          Avaliações" , "IDUsuario" , "IDArtigo" , "Rating" );
3
4      UserCorrelation correlacao = new PearsonCorrelation( model );
5
6      UserNeighborhood vizinhos = new NearestUserNeighborhood( 5 ,
7          correlacao , model );
8
9      Recommender recommender = new GenericUserBasedRecommender( model
10         , vizinhos , correlacao );
11
12      Recommender cachingRecommender = new CachingRecommender (
13         recommender );
14
15      List<RecommendedItem> recommendations = cachingRecommender .
16         recommend( "5282" , 10 );
17      for( RecommendedItem item : recommendations ) {
18         System.out.println( item.getValue() + " - " + item.getItem ()
19             .getID () );
20     }
```

Na linha 1 cria-se um Modelo de dados que utiliza acesso ao Banco de Dados.

Na linha 3 utiliza-se a correlação de Pearson como medida de similaridade na recomendação.

Na linha 5 escolhem-se os cinco vizinhos mais próximos para formar a vizinhança.

Nas linhas 8 e 9 cria-se no Recomendador um cache para para melhorar a performance do sistema de recomendação.

Nas linhas 12, 13 e 14 apresentam-se as dez primeiras recomendações para o usuário.

## 6.5 Módulo de Recomendação Baseada em Conteúdo

Na implementação do algoritmo de Filtragem Baseada em Conteúdo foi utilizado o projeto *Hibernate Search* que em *background* utiliza a API Lucene. O Hibernate Search foi utilizado por oferecer uma rápida integração como *Hibernate Core* e por fornecer suporte completo ao modelo vetorial através do Lucene. Maiores informações sobre o *Hibernate Search* podem ser consultadas em <http://www.hibernate.org> . Para maiores informações

sobre o Lucene podem ser consultadas em <http://www.apache.org> .

Inicialmente, cria-se um indexador para os artigos do provedor de dados escolhido. A classe abaixo ilustra o indexador criado.

Listagem 6.4: Trecho de Código do Indexador

```
1 package br.ufal.ic.repository;
2
3 import java.util.List;
4
5 import org.hibernate.Session; import org.hibernate.Transaction;
6 import org.hibernate.search.FullTextSession; import
7 org.hibernate.search.Search; import
8 br.ufal.ic.persistence.hibernate.HibernateUtil;
9
10 // Indexa os artigos da tabela artigo
11 public class IndexarArtigos {
12
13     public static void main(String [] args) {
14         // First unit of work
15         Session session2 = HibernateUtil.getSessionFactory().openSession();
16         FullTextSession fullTextSession2 = Search.createFullTextSession(
17             session2);
18         Transaction tx2 = fullTextSession2.beginTransaction();
19         List<Artigo> artigos = session2.createQuery("from Artigo as artigo
20             ").list();
21
22         for (Artigo artigo : artigos) {
23             fullTextSession2.index(artigo);
24         }
25
26         tx2.commit();
27         session2.close();
28
29         // Shutting down the application
30         HibernateUtil.shutdown();
31         System.out.println("indexacao terminada");
32     }
33 }
```

Após os artigos serem indexados, o próximo passo é recuperar os artigos publicados do pesquisador para construir o vetor de consulta. O trecho de código abaixo ilustra o processo de construção do vetor de consulta.



Listagem 6.5: Trecho de Código do Vetor de Consulta

```
1 //monta a consulta através do perfil dos artigos publicados do pesquisador
2 public BooleanQuery montaQuery(String id_curriculo) throws IOException{
3
4     session2 = HibernateUtil.getSessionFactory().openSession();
5     List<ArtigoPublicado> artigop = session2.createQuery(" from
6         ArtigoPublicado where curriculoVitae_id="+id_curriculo).list();
7
8     //para cada artigo publicado
9     for (ArtigoPublicado artigo : artigop) {
10         anoArtigo = Integer.parseInt(artigo.getDadosBasicosArtigo().
11             getAnoArtigo());
12
13         pesoAno = 1 - (anoAtual - anoArtigo)*(0.45/(anoAtual - menorAno
14             ));
15
16         peso = pesoAno * pesoTitulo; // referente ao título
17         array = AnalyzerUtils.tokensFromAnalysis2(new HelioAnalyzer(),
18             artigo.getDadosBasicosArtigo().getTituloArtigo());
19
20         int tam=0;
21         peso = pesoAno * pesoPalavraChave; // para palavras chaves
22
23         for(int i=0;i<tam;i++){
24             palavra = palavraschaves.get(i);
25             if (!palavra.equals("")){
26                 if (palavrapeso.containsKey(palavra)){
27                     if (palavrapeso.get(palavra).doubleValue()<peso){
28                         palavrapeso.remove(palavra);
29                         palavrapeso.put(palavra ,new Double(peso));
30                     }
31                 }else{
32                     palavrapeso.put(palavra ,new Double(peso));
33                 }//else
34             }//if
35         }//for
36
37     }// for artigo publicado
38
39     // Montando o query com os termos do título
40     BooleanQuery tituloQuery = new BooleanQuery();
41     String texto="";
```

```
39     TermQuery t ;
40     for (Enumeration<String> e = titulopeso.keys(); e.hasMoreElements()
41         ;) {
42         texto = e.nextElement();
43         t = new TermQuery(new Term("title", texto));
44         t.setBoost(titulopeso.get(texto).floatValue()+1);
45         tituloQuery.add(t, BooleanClause.Occur.SHOULD);
46     }
47     // Montado o query para as palavras chaves
48     BooleanQuery palavraQuery = new BooleanQuery();
49     Term t1;
50     PhraseQuery f ;
51     for (Enumeration<String> e = palavrapeso.keys(); e.hasMoreElements
52         ());) {
53         texto = e.nextElement();
54         t1 = new Term("description", texto);
55         f = new PhraseQuery();
56         f.add(t1);
57         f.setBoost(palavrapeso.get(texto).floatValue()+1);
58         palavraQuery.add(f, BooleanClause.Occur.SHOULD);
59     }
60     BooleanQuery completoQuery = new BooleanQuery();
61     completoQuery.add(tituloQuery, BooleanClause.Occur.SHOULD);
62     completoQuery.add(palavraQuery, BooleanClause.Occur.SHOULD);
63
64     tx2.commit();
65     session2.close();
66     return completoQuery;
67 }
```

Nas linhas de 5 a 7 são recuperados os artigos publicados por um determinado pesquisador.

Nas linhas de 9 a 34 são recuperados o ano, título e palavras-chave de cada artigo publicado pelo pesquisador. Nestes atributos são aplicados os processos de remoção de *stopwords* e *stemming*. Os termos presentes no título e nas palavras-chave possuem fatores de importância diferentes, sendo termos no título mais relevantes do que nas palavras-chave. Os artigos que são mais recentes possuem uma importância maior do que os artigos mais antigos, pois representam os interesses atuais do pesquisador.

Nas linhas de 36 a 67 é construída uma consulta booleana contendo os termos do título e das palavras-chave.

Após a construção do vetor de consulta, o próximo passo é fazer a busca no índice. O trecho de código abaixo ilustra o processo de busca através desse vetor.

Listagem 6.6: Trecho de Código do Processo de Busca

```
1
2 // busca os n artigos baseados no query
3   public List busca(BooleanQuery q){
4
5       Session session2 = HibernateUtil.getSessionFactory().openSession();
6       FullTextSession fullTextSession2 = Search.createFullTextSession(
7           session2);
8       Transaction tx2 = fullTextSession2.beginTransaction();
9       List result=null;
10      try{
11          FullTextQuery fullTextQuery = fullTextSession2.
12              createFullTextQuery(q, Artigo.class);
13          fullTextQuery.setProjection( FullTextQuery.SCORE, FullTextQuery
14              .THIS);
15          System.out.println(" Resultados "+fullTextQuery.getResultSize())
16              ;
17          result = fullTextQuery.list();
18      }catch(Exception p){
19          p.getMessage();
20      }
21      tx2.commit();
22      session2.close();
23      return result;
24  }
```

O trecho de código acima mostra que é retornado para o pesquisador uma lista de artigos que "casam" com o vetor de consulta. Estes artigos são mostrados em ordem de pontuação decrescente.

## 6.6 Módulos de Recomendação Híbrido

Após implementado os algoritmos de Filtragem Colaborativa e Baseado em Conteúdo, são implementados os algoritmos Híbridos. Na implementação dos algoritmos híbridos

foram utilizados os algoritmos Misto e Ponderado.

No algoritmo Misto, os artigos presentes em ambas as listas recebem uma prioridade maior em relação aos algoritmos que não estão nessa. O trecho de código abaixo ilustra o processo de recomendação híbrida utilizando o algoritmo Misto.

Listagem 6.7: Trecho de Código do Processo de Recomendação Híbrida utilizando o Algoritmo Misto

```

1
2     int p =0; // posicao do item
3     FiltragemColaborativa f1 = new FiltragemColaborativa ();
4     f1.setVizinhos (1);
5     f1.setRecomendador ();
6     List<RecommendedItem> listaFC = f1.getRecomendacoes ("6", numero);
7     for (RecommendedItem item : listaFC) {
8         p++;
9         lFC.add(item.getItem().getID().toString());
10        hFC.put(item.getItem().getID().toString(), p);
11    }
12
13    FiltragemBaseadaConteudo f2 = new FiltragemBaseadaConteudo ();
14    List listaFBC = f2.busca(f2.montaQuery("6"));
15    for (int i=0;i<numero;i++){
16        Object [] resultado = (Object []) listaFBC.get(i);
17        lFBC.add( ((Artigo)(resultado [1])) .getId()+"");
18        hFBC.put(((Artigo)(resultado [1])) .getId()+"", i);
19    }
20
21    Set<String> uniao = new HashSet<String>(lFC);
22    uniao.addAll(lFBC);
23    Set<String> intersecao = new HashSet<String>(lFC);
24    intersecao.retainAll(lFBC);
25    int pontuacao =0;
26    for (Iterator iter = intersecao.iterator(); iter.hasNext();) {
27        String element = (String) iter.next();
28        pontuacao = hFC.get(element).intValue()+hFBC.get(element).
29            intValue();
30        pontuacoes.add(new Pontuacao(element, pontuacao));
31    }
32    Collections.sort(pontuacoes);
33    for (Iterator iter = pontuacoes.iterator(); iter.hasNext();) {
34        Pontuacao element = (Pontuacao) iter.next();
35        recomendacoes.add(element.getId());

```

```
35     }
36
37     Set<String> diferenca12 = new HashSet<String>(lFC);
38     diferenca12.removeAll(lFBC);
39     System.out.println("L1-L2: "+diferenca12);
40
41     Set<String> diferenca21 = new HashSet<String>(lFBC);
42     diferenca21.removeAll(lFC);
43     System.out.println("L2-L1: "+diferenca21);
44
45     List<String> listaFCrestante = new ArrayList<String>();
46     List<String> listaFBCrestante = new ArrayList<String>();
47     listaFCrestante.addAll(diferenca12);
48     listaFBCrestante.addAll(diferenca21);
49
50     if (listaFCrestante.size() >= listaFBCrestante.size()) {
51         for (int i = 0; i < listaFCrestante.size(); i++) {
52             recomendacoes.add(listaFCrestante.get(i));
53             if (i < listaFBCrestante.size()) {
54                 recomendacoes.add(listaFBCrestante.get(i));
55             }
56         }
57     }
58     else { //
59         for (int i = 0; i < listaFBCrestante.size(); i++) {
60             recomendacoes.add(listaFBCrestante.get(i));
61             if (i < listaFCrestante.size()) {
62                 recomendacoes.add(listaFCrestante.get(i));
63             }
64         }
65     }
66 }
67
68 Session session = HibernateUtil.getSessionFactory().openSession();
69 for (Iterator iter = recomendacoes.iterator(); iter.hasNext();) {
70     String element = (String) iter.next();
71     Artigo artigo = (Artigo) session.createQuery("from Artigo where
72         id =" + element).uniqueResult();
73     System.out.println(artigo.getTitle());
74 }
75 session.close();
```

Nas linhas de 1 a 11 realiza-se o processo de recomendação colaborativa. Os artigos

recomendados são armazenados numa determinada lista .

Nas linhas de 13 a 19 realiza-se o processo de recomendação baseada em conteúdo. Os artigos recomendados são armazenados numa determinada lista .

Nas linhas de 21 a 67 mostra-se como é gerada a lista final de recomendação com a soma das posições de cada elemento pertencente a interseção das listas e os elementos que não estão no conjunto sendo intercalados na lista de recomendação.

Nas linhas de 70 a 76 os titulos dos artigos presentes na lista de recomendação são mostrados aos usuários.

O processo de implementação do algoritmo misto é mostrado na figura 12 .

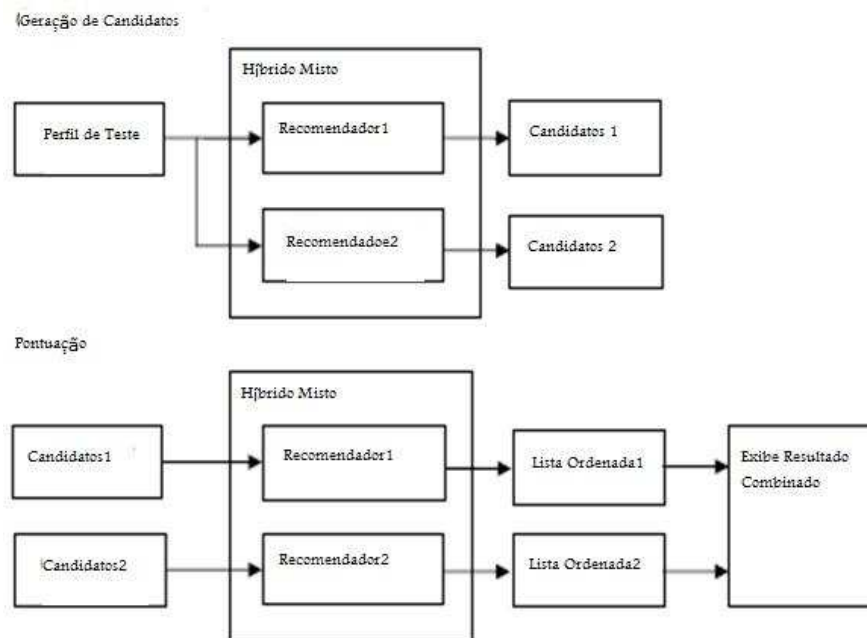


Figura 12: Implementação do Algoritmo Misto. Figura adaptada de (BURKE, 2007)

No algoritmo ponderado, cada algoritmo de recomendação possui um peso. O peso final desse é formado pela combinação linear dos pesos de cada algoritmo participante do sistema. O trecho de código abaixo ilustra o processo de recomendação híbrida utilizando o algoritmo ponderado.

Listagem 6.8: Trecho de Código do Processo de Recomendação Híbrida utilizando o Algoritmo Ponderado

```

1
2     final double pesoFC = 0.8;
3     final double pesoFBC = 0.2;

```

```

4         final int numero = 5; // número de recomendações de cada tipo
           de filtragem
5
6
7         FiltragemColaborativa f1 = new FiltragemColaborativa ();
8         f1.setVizinhos(1);
9         f1.setRecomendador ();
10        List<RecommendedItem> listaFC = f1.getRecomendacoes("6", numero)
           ;
11
12        for (RecommendedItem item : listaFC) {
13
14            lFC.add(item.getItem().getID().toString());
15            hFC.put(item.getItem().getID().toString(), item.getValue())
           ;
16        }
17
18        FiltragemBaseadaConteudo f2 = new FiltragemBaseadaConteudo ();
19        List listaFBC = f2.busca(f2.montaQuery("6"));
20        for (int i=0;i<numero;i++){
21            Object [] resultado = (Object []) listaFBC.get(i);
22            lFBC.add( ((Artigo)(resultado [1])) .getId()+"");
23            hFBC.put(((Artigo)(resultado [1])) .getId()+"", Double.
                parseDouble(resultado [0]+"")*5); // O 5 é por causa da
                diferenca de escala de 0-1 e 0-5
24        }
25        // inserindo os artigos da interseção
26        for (Iterator iter = intersecao.iterator(); iter.hasNext();) {
27            String element = (String) iter.next();
28            pontuacao = hFC.get(element).doubleValue()*pesoFC+hFBC.get(
                element).doubleValue()*pesoFBC;
29            pontuacoes.add(new Pontuacao(element, pontuacao));
30        }
31
32        //inserindo os artigos que só estão em FC
33        Set<String> diferenca12 = new HashSet<String>(lFC);
34        diferenca12.removeAll(lFBC);
35        System.out.println("L1-L2: "+diferenca12);
36        for (Iterator iter = diferenca12.iterator(); iter.hasNext();) {
37            String element = (String) iter.next();
38            pontuacao = hFC.get(element).doubleValue()*pesoFC;
39            pontuacoes.add(new Pontuacao(element, pontuacao));
40        }

```

```
41
42     //inserindo os artigos que só estão em FBC
43     Set<String> diferenca21 = new HashSet<String>(lFBC);
44     diferenca21.removeAll(lFC);
45     System.out.println("L2-L1: "+diferenca21);
46     for (Iterator iter = diferenca21.iterator(); iter.hasNext();) {
47         String element = (String) iter.next();
48         pontuacao = hFBC.get(element).doubleValue()*pesoFBC;
49         pontuacoes.add(new Pontuacao(element, pontuacao));
50     }
51
52     // ordenando os artigos pela pontuação
53
54     Collections.sort(pontuacoes);
55     Collections.reverse(pontuacoes);
56
57     for (Iterator iter = pontuacoes.iterator(); iter.hasNext();) {
58         Pontuacao element = (Pontuacao) iter.next();
59         recomendacoes.add(element.getId());
60     }
61     Session session = HibernateUtil.getSessionFactory().openSession
62         ();
63     for (Iterator iter = recomendacoes.iterator(); iter.hasNext();)
64     {
65         String element = (String) iter.next();
66         Artigo artigo = (Artigo)session.createQuery("from Artigo
67             where id =" +element).uniqueResult();
68         System.out.println(artigo.getTitle());
69     }
70     session.close();
```

Nas linhas de 1 a 5 os pesos dos algoritmos e o número de recomendações de cada algoritmo são ajustados.

Nas linhas de 7 a 16 realiza-se o processo de recomendação colaborativa. Os artigos recomendados são armazenados numa determinada lista L1.

Nas linhas de 18 a 24 realiza-se o processo de recomendação baseada em conteúdo. Os artigos recomendados são armazenados numa determinada lista L2.

Nas linhas de 26 a 35 calcula-se a pontuação de cada artigo presente na interseção.

Nas linhas de 37 a 45 calcula-se a pontuação dos artigos presentes apenas na filtragem



colaborativa .

Nas linhas de 46 a 55 calcula-se a pontuação dos artigos presentes apenas na filtragem baseada em conteúdo.

Nas linhas de 58 a 71 os artigos são ordenados através de suas pontuações e são apresentados aos usuários os títulos dos artigos presentes na lista de recomendação.

O processo de implementação do algoritmo ponderado é mostrado na figura 13 .

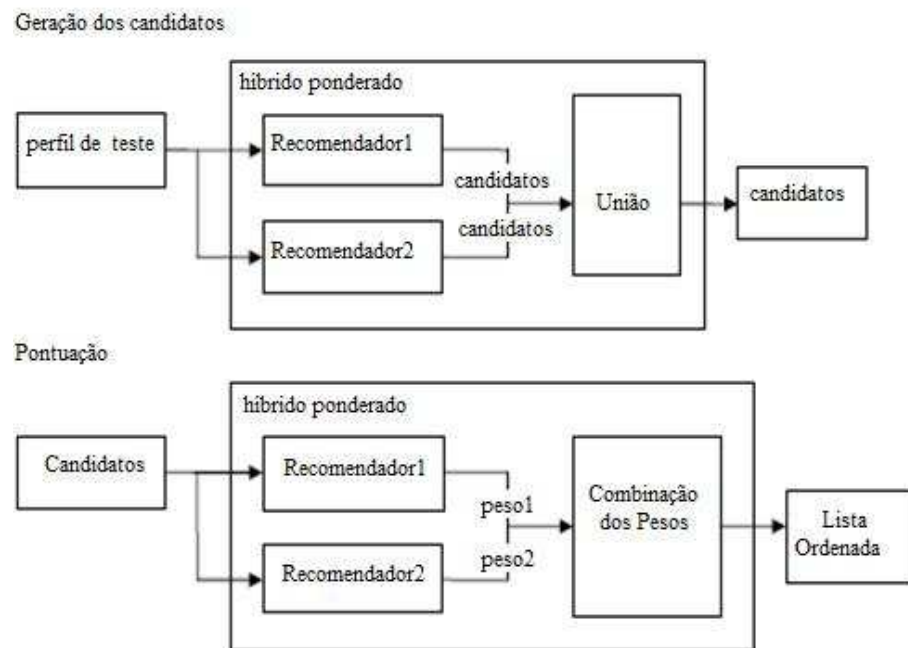


Figura 13: Implementação do Algoritmo Ponderado. Figura adaptada de (BURKE, 2007)

## 7 *Experimentos e Resultados*

*“São as crianças que, sem falar, nos ensinam as razões para viver. Elas não têm saberes a transmitir. No entanto, elas sabem o essencial da vida.”*

Cirilo Veloso Moraes

Este capítulo apresenta os experimentos realizados e discute sobre os resultados obtidos.

## 7.1 Dados

Para a realização dos experimentos foi necessário um conjunto de dados com usuários e seus respectivos currículos lattes, artigos e avaliações.

Em relação aos usuários, foram solicitados os currículos lattes em xml de dez pesquisadores. Cada usuário possui no mínimo um artigo publicado, isto é necessário para que a filtragem baseada em conteúdo possa ser utilizada. Os artigos foram obtidos da biblioteca digital do CiteSeer (BOLLACKER; LAWRENCE; GILES, 1999), um repositório online<sup>1</sup> de artigos de Ciência da Computação. O CiteSeer é um tipo inovador de biblioteca digital, onde toda a coleta, estruturação e a busca de documentos são feitas automaticamente por computadores. Os artigos foram obtidos utilizando o protocolo OAI-PMH, tendo como base a URL <http://cs1.ist.psu.edu/cgi-bin/oai.cgi> . O CiteSeer disponibiliza aproximadamente setecentos mil artigos no formato *Dublin Core*.

As avaliações dos artigos foram obtidas de um subconjunto da base de dados do MovieLens. O MovieLens é uma base de dados utilizada para teste de vários sistemas de recomendação e está disponível em <http://www.grouplens.org/node/73>. Existem duas versões do MovieLens que são disponibilizadas. A primeira consiste de cem mil avaliações de novecentos e quarenta e três usuários. A segunda consiste de um milhão de avaliações de seis mil e quarenta usuários. As avaliações são realizadas considerando-se a escala de um à cinco.

Para testar nossos algoritmos rapidamente, nós construímos uma base de dados contendo uma amostra de dois mil artigos escolhidos aleatoriamente do CiteSeer e um subconjunto de quatro mil quinhentos e setenta e oito avaliações da base do MovieLens também escolhidos de forma aleatória.

A figura 14 mostra a quantidade de artigos por usuário.

A figura 15 mostra o número de avaliações por usuário.

---

<sup>1</sup>O CiteSeer encontra-se disponível em <http://citeseer.ist.psu.edu/>



Figura 14: Artigos por Usuário



Figura 15: Avaliações por Usuário

## 7.2 Método Experimental

Segundo Herlocker (HERLOCKER et al., 2004), avaliações de sistemas de recomendação podem ser realizadas com análises *off-line*, *on-line*, ou uma combinação de ambos. Na análise *off-line*, são utilizados conjuntos de dados históricos de um sistema de recomendação, no qual testes são feitos e métricas são analisadas para comparações de resultados. Na análise *on-line*, o usuário pode ser submetido, após receber as recomendações, à uma série de avaliações, tais como, a análise de satisfação.

Os nossos testes foram realizados através de experimentos *off-line*. Esse tipo de análise

tem o objetivo de verificar a capacidade de recomendar corretamente artigos. O experimento *off-line* foi escolhido, pois, a realização dos testes é rápida, não necessitando de envolvimento do usuário, e os testes podem ser facilmente repetidos.

## 7.3 Métricas

Para avaliar a recomendação top-n nós usamos duas medidas amplamente utilizadas na comunidade de pesquisa de recuperação de informação, são elas: precisão (*precision*) e cobertura (*recall*). Entretanto, (SARWAR et al., 2000) modifica ligeiramente a definição de cobertura e da precisão e definí-as no contexto de sistema de recomendação. A seguir é mostrado o procedimento utilizado: Inicialmente, os dados são divididos em dois conjuntos - um conjunto de treinamento e um conjunto de teste. Os algoritmos de recomendação trabalham no conjunto de treinamento, e geram um número de recomendações, que chamamos de **top-n**. O principal objetivo é observar o conjunto de teste, (por exemplo, a porção escondida dos dados adquiridos) e comparar os artigos com o conjunto top-n. Os artigos que aparecem em ambos os conjuntos são membros de um conjunto especial, que nós chamamos de conjunto *hit*. Abaixo, mostra-se as definições de cobertura e precisão no contexto de sistema de recomendação .

Cobertura: Razão entre o tamanho do conjunto hit sobre o tamanho conjunto de teste.  $Cobertura = \frac{|hit|}{|teste|}$  .

Precisão: Razão entre o tamanho do conjunto hit sobre o tamanho do conjunto top-n.  $Precisão = \frac{|hit|}{|top - n|}$  .

Segundo (SARWAR et al., 2000) estas duas medidas são de natureza conflitante. Ao passo que, aumentando o número n tende a aumentar a cobertura, mas diminui a precisão. O fato de que ambos são críticos para o julgamento da qualidade, nos leva a usar uma combinação dos dois. Em particular, nós usamos a métrica padrão F1, que fornece peso igual a ambos. Computa-se F1 para cada usuário individual e calcula-se o valor médio .

F1 : Média harmônica entre cobertura e precisão.  $F1 = \frac{2 * cobertura * precisão}{cobertura + precisão}$ ).

## 7.4 Teste

Cada algoritmo recebe como entrada os dados de treinamento e manipulá-os da forma que o convém. A avaliação é executada retendo-se aleatoriamente vinte avaliações positivas (avaliação 4 ou 5) de cada usuário. Esses serão os artigos da base de teste em que o desempenho do recomendador será avaliado. Todas as outras avaliações são consideradas parte do perfil de treinamento. Ao algoritmo de recomendação é fornecido a base de treinamento, sem os artigos positivamente avaliados, e o recomendador realiza dez recomendações. O resultado do processo da recomendação é um subconjunto classificado da base de dados que contém aqueles artigos possivelmente de interesse do usuário. Desse conjunto, nós registramos a posição de cada artigo positivamente avaliado do teste. Idealmente, essa posição deve ser o mais baixa quanto possível - o quanto mais perto da parte inicial o artigo estiver, mais precisamente o recomendador está refletindo as preferências do usuário.

## 7.5 Apresentação dos Resultados

### 7.5.1 Resultados da Filtragem Colaborativa

A tabela 13 mostra a similaridade de todos os usuários em relação ao usuário1, através do coeficiente de Pearson.

Tabela 13: Similaridade usando Pearson entre o usuário1 e todos os usuários

<i>Usuário</i>	<i>Similaridade</i>
1	1,0
2	-0,017892922197050555
3	0,1709765025387663
4	0,30480224542068957
5	0,08221849337356409
6	0,248981564461246
7	0,14388551457155654
8	0,06977915405003718
9	0,4504023308083442
10	0,3063380596004754

Observando os dados da tabela 13 vemos que o usuário1 possui como melhores dois vizinhos os usuários 4 e 9 (excetuando-se o próprio usuário1).

A tabela 14 mostra as 5 primeiras recomendações em ordem decrescente de predição para o usuário1.

Tabela 14: Top-5 artigos recomendados para o usuário1 utilizando filtragem colaborativa com 1 vizinho

<i>id do artigo</i>	<i>título</i>
177	Closed Loop Identification Of Nonlinear Systems.
180	SuperWeb: Research Issues in Java-Based Global Computing.
194	Neutrino Oscillations With Three-Generation Mixings and Mass Hierarchy.
68	Nozomi - A Fast, Memory-Efficient Stack Decoder For Lvcsr.
689	Selforganization in a System of Binary Strings With Topological Interactions.

Segundo Herlocker (HERLOCKER et al., 1999), o número de vizinhos influencia a qualidade do processo de recomendação.

A figura 16 mostra a precisão de cada usuário variando-se o número de vizinhos  $k$ .

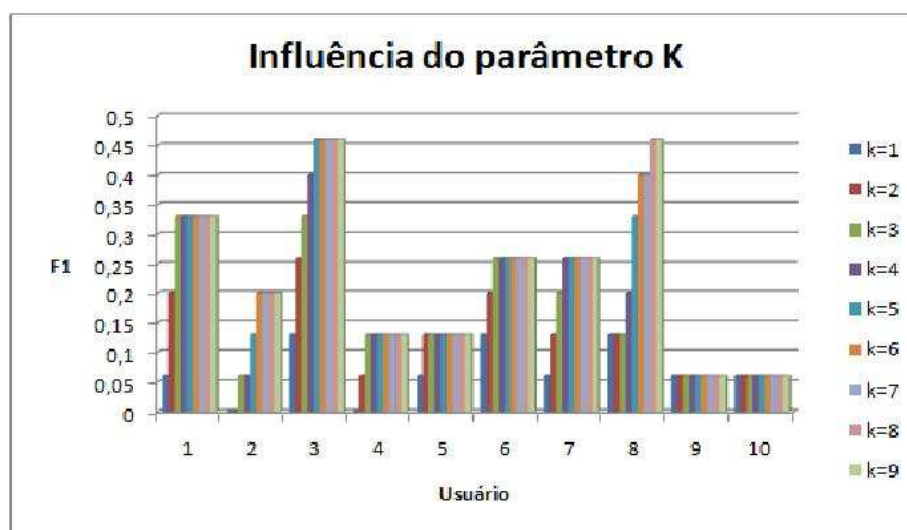


Figura 16: Influência do Número de Vizinhos

Observa-se na figura 16 que os valores oito e nove para o parâmetro  $k$ , produzem um melhor valor para a métrica F1. Portanto, o valor de  $k=8$  será utilizado no restante dos experimentos.

## 7.5.2 Resultados da Filtragem Baseada em Conteúdo

A tabela 15 mostra alguns os termos do perfil do usuário1 junto com os seus respectivos pesos.

Tabela 15: Termos e pesos do perfil do usuário1

<i>termo</i>	<i>peso</i>
agent	1,0
ambient	0,775
educ	1,0
web	1,0
intellig	0,55

A tabela 16 mostra as 5 primeiras recomendações em ordem decrescente para o usuário1.

A figura 17 mostra a métrica F1 de cada usuário para a filtragem por conteúdo.

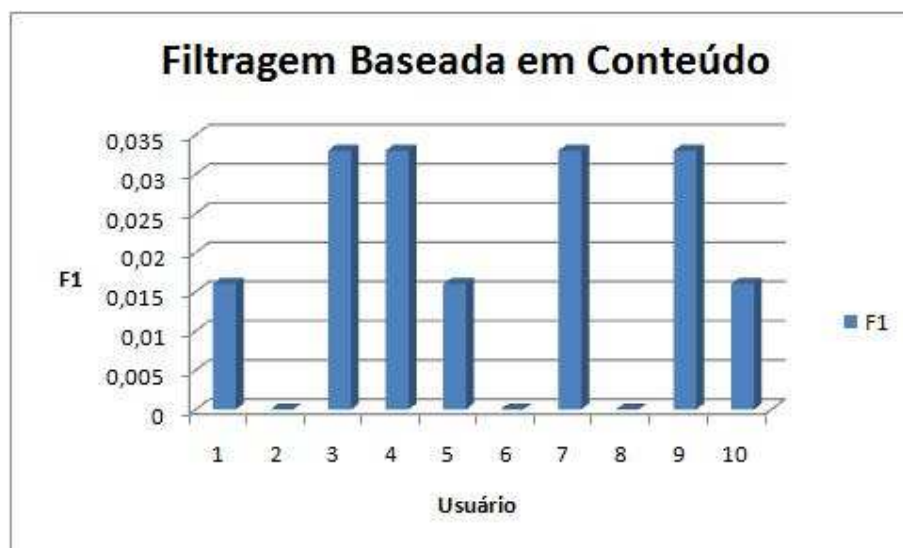


Figura 17: Filtragem Baseada em Conteúdo

Tabela 16: Top-5 artigos recomendados para o usuário1 utilizando filtragem baseada em conteúdo

<i>id do artigo</i>	<i>título</i>
1813	The Open Meeting: A Web-Based System for Conferencing and Collaboration.
39	Towards 3-D model-based tracking and recognition of human movement: a multi-view approach.
932	Intelligent Interface Agents for Intelligent Environments.
1464	Towards Reliable Autonomous Agents.
136	Towards a Better Understanding of Memory-Based Reasoning Systems.



### 7.5.3 Resultados da Filtragem Híbrida utilizando o Algoritmo Misto

A tabela 17 mostra as 5 primeiras recomendações em ordem decrescente para o usuário1.

Tabela 17: Top-5 artigos recomendados para o usuário1 utilizando filtragem híbrida mista

<i>id do artigo</i>	<i>título</i>
180	SuperWeb: Research Issues in Java-Based Global Computing.
1464	Towards Reliable Autonomous Agents.
689	Selforganization in a System of Binary Strings With Topological Interactions.
932	Intelligent Interface Agents for Intelligent Environments.
68	Nozomi - A Fast, Memory-Efficient Stack Decoder For Lvcsr.

### 7.5.4 Resultados da Filtragem Híbrida utilizando o Algoritmo Ponderado

A tabela 18 mostra as 5 primeiras recomendações em ordem decrescente para o usuário1.

Tabela 18: Top-5 artigos recomendados para o usuário1 utilizando filtragem híbrida ponderada com  $p=0,6$

<i>id do artigo</i>	<i>título</i>
177	Closed Loop Identification Of Nonlinear Systems.
194	Neutrino Oscillations With Three-Generation Mixings and Mass Hierarchy.
68	Nozomi - A Fast, Memory-Efficient Stack Decoder For Lvcsr.
180	SuperWeb: Research Issues in Java-Based Global Computing.
689	Selforganization in a System of Binary Strings With Topological Interactions.

A tabela 19 mostra as 5 primeiras recomendações em ordem decrescente para o usuário1.

Tabela 19: Top-5 artigos recomendados para o usuário1 utilizando filtragem híbrida ponderada com  $p=0,01$

<i>id do artigo</i>	<i>título</i>
381	Web Based Parallel/Distributed Medical Data Mining Using Software Agents.
1813	The Open Meeting: A Web-Based System for Conferencing and Collaboration.
39	Towards 3-D model-based tracking and recognition of human movement: a multi-view approach.
932	Intelligent Interface Agents for Intelligent Environments.
1464	Towards Reliable Autonomous Agents.

## 7.6 Avaliação dos Resultados

O valor médio de F1 para filtragem colaborativa foi de 0,235. O valor médio de F1 para filtragem baseada em conteúdo foi de 0,018. O valor médio de F1 para filtragem híbrida mista foi de 0,266. O valor médio de F1 para filtragem híbrida ponderada com peso máximo para Filtragem Colaborativa foi de 0,255.

A figura 18 mostra a métrica F1 de cada usuário para as quatro técnicas.



Figura 18: Comparação do resultado das técnicas

O melhor resultado obtido foi para o algoritmo Misto com 0,266. Observa-se, assim, a melhoria da filtragem híbrida em relação a filtragem baseada em conteúdo ou colaborativa.

## 8 *Conclusões*

Este trabalho apresentou uma proposta de um sistema de recomendação personalizada de artigos científicos para Bibliotecas Digitais. Tal sistema, objetiva a personalização da informação, considerando o perfil do usuário. Foi proposta uma solução para representar o perfil do usuário, através da extração de informações de um arquivo xml referente ao currículo Lattes do usuário. Além disso, foi desenvolvido um mecanismo que possibilita a modificação do perfil por parte do usuário, permitindo-lhe tratar eventuais informações desatualizadas extraídas do seu Lattes, relativamente as suas atuais preferências. O sistema garante que qualquer biblioteca digital que provê metadados no formato Dublin Core (DC) e dá suporte ao protocolo OAI-PMH, possa ser utilizada como fonte para prover informações sobre os artigos a serem recomendados.

Para o processo de recomendação, foi implementada uma abordagem híbrida baseada em um algoritmo misto e ponderado, para que isto fosse possível foi implementado um algoritmo baseado em filtragem por conteúdo e um baseado em filtragem colaborativa.

Acredita-se que o sistema proposto possui aplicações importantes, como por exemplo, a recomendação de artigos científicos para auxiliar o processo de aprendizagem em sistemas de educação à distância (EAD).

Os resultados dos experimentos mostraram-se satisfatórios, pois a filtragem híbrida melhorou a métrica F1 em relação as filtragens colaborativa e conteúdo. Com isso pode-se ganhar mais qualidade com relação às medidas de precisão e cobertura .

Dentre as limitações do trabalho, pode-se destacar:

- Apesar da existência de um mecanismo que permite que o usuário modifique seu perfil, de forma que este represente suas necessidades atuais, esse mecanismo ainda necessita de uma intervenção direta do usuário.
- Apesar do sistema proposto garantir que qualquer biblioteca digital que provê metadados no formato Dublin Core (DC) possa ser utilizada, isso pode constituir uma

limitação, pois bibliotecas digitais que não utilizarem esse padrão para descrição do conteúdo, não poderão interoperar com o sistema proposto.

## **Trabalhos Futuros**

Como trabalhos futuros pode-se citar:

- Mais experimentos com outras técnicas de filtragem híbrida e compará-las com os resultados obtidos.
- Integração deste sistema com metadados que utilizem o padrão RDF para que o mesmo possa utilizar as vantagens da Web Semântica.

## *Referências*

- ADOMAVICIUS, G.; TUZHILIN, A. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, IEEE Educational Activities Department, Piscataway, NJ, USA, v. 17, n. 6, p. 734–749, 2005.
- ARP. *ARP - Active Recommendation Project*. 2001. Disponível em: <http://informatics.indiana.edu/rocha/lwww>. Acesso: Jan. 2008.
- BAEZA-YATES, R.; RIBEIRO-NETO, B. *Modern Information Retrieval*. [S.l.]: Addison-Wesley, 1999.
- BALABANOVIC, M.; SHOHAM, Y. Combining content-based and collaborative recommendation. *Communications of the ACM*, v. 40, n. 3, March 1997.
- BELKIN, N. J.; CROFT, W. B. Information filtering and information retrieval: two sides of the same coin? *Commun. ACM*, ACM, New York, NY, USA, v. 35, n. 12, p. 29–38, 1992. ISSN 0001-0782.
- BERNERS-LEE, T.; LASSILA, O.; HENDLER, J. The semantic web. *Scientific American*, v. 254, n. 5, p. 34–43, 2001.
- BOLLACKER, K. D.; LAWRENCE, S.; GILES, C. L. A system for automatic personalized tracking of scientific literature on the web. In: *In Digital Libraries 99 - The Fourth ACM Conference on Digital Libraries*. [S.l.]: ACM Press, 1999. p. 105–113.
- BURKE, R. Hybrid recommender systems survey and experiments. 2002.
- BURKE, R. Hybrid web recommender systems. In: BRUSILOVSKY, P.; KOBSA, A.; NEJDL, W. (Ed.). *The Adaptive Web: Methods and Strategies of Web Personalization*. Berlin, Heidelberg: Springer, 2007, (Lecture Notes in Computer Science, v. 4321). cap. 12, p. 377–408.
- CALLAN, J.; SMEATON, A. *Personalisation and Recommender Systems in Digital Libraries*. 2003. Disponível em: [http://www.dli2.nsf.gov/internationalprojects/working\\_group\\_reports/personalisation.pdf](http://www.dli2.nsf.gov/internationalprojects/working_group_reports/personalisation.pdf). Acesso: Jan. 2008.
- CALLAN, J. et al. (Ed.). *Personalisation and Recommender Systems in Digital Libraries*. [S.l.: s.n.], 2003.
- CAMPOS, M. L. d. A. C. M. L. M.; CAMPOS, L. M. Infra-estrutura tecnológica de uma biblioteca digital: elementos básicos. In: \_\_\_\_\_. 2. ed. [S.l.]: EDUFBA, 2006. cap. 4, p. 77–93.

- CAZELLA, S. C.; ALVARES, L. O. C. *Creating virtual web communities through a hybrid recommender system*. 2004. II WORKSHOP DE TESES E DISSERTAÇÕES EM INTELIGÊNCIA ARTIFICIAL. São Luis, Brazil.
- CAZELLA, S. C.; ALVARES, L. O. C. An architecture based on multi-agent system and data mining for recommending research papers and researchers. In: *SEKE*. [S.l.: s.n.], 2006. p. 67–72.
- CLAYPOOL, M. et al. *Combining Content-Based and Collaborative Filters in an Online Newspaper*. 1999. Disponível em: <[citeseer.ist.psu.edu/claypool99combining.html](http://citeseer.ist.psu.edu/claypool99combining.html)>.
- CONGRESS., L. of. *METS Metadata Encoding and Transmissions Standard*. 2006. Acessado em janeiro de 2007. Disponível em: <<http://www.loc.gov/standards/mets/METSOverview.v2.html>>.
- CONGRESS., L. of. *MARC Standards*. 2007. Acessado em janeiro de 2007. Disponível em: <<http://www.loc.gov/marc/>>.
- DC-OAI. *A XML Schema for validating Unqualified Dublin Core metadata associated with the reserved oai\_dc metadataPrefix*. 2008. Disponível em: [http://www.openarchives.org/OAI/2.0/oai\\_dc.xsd](http://www.openarchives.org/OAI/2.0/oai_dc.xsd). Acesso: Jan. 2008.
- DRABENSTOTT, K. M. *Analytical review of the library of the future*. [S.l.]: Washington, DC, USA : Council on Library Resources, 1994., 1994.
- DUBLIN. *Core Metadata Initiative*. 2008. Disponível em: <http://www.dublincore.org/>. Acesso: Jan. 2008.
- GOLDBERG, D. et al. Using collaborative filtering to weave an information tapestry. *Commun. ACM*, ACM Press, New York, NY, USA, v. 35, n. 12, p. 61–70, December 1992. ISSN 0001-0782. Disponível em: <<http://portal.acm.org/citation.cfm?id=138867>>.
- GOLDBERG, D. et al. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, v. 35, n. 12, p. 61–70, 1992.
- GONCALVES, M. A. et al. Streams, structures, spaces, scenarios, societies (5s): A formal model for digital libraries. *ACM Trans. Inf. Syst.*, ACM, New York, NY, USA, v. 22, n. 2, p. 270–312, 2004. ISSN 1046-8188.
- GROSSMAN, D. A. *Information Retrieval*. 2. ed. [S.l.]: Springer, 2004.
- HAN, J.; KAMBER, M. *Data Mining: concepts and techniques*. [S.l.]: Morgan-Kaufmann, 2001.
- HERLOCKER, J. L. *Understanding and improving automated collaborative filtering systems*. Tese (Doutorado), 2000. Adviser-Joseph A. Konstan.
- HERLOCKER, J. L. *Understanding and improving automated collaborative filtering systems*. Tese (Doutorado em Ciência da Computação) — University of Minnesota, Minnesota, 2000.
- HERLOCKER, J. L.; KONSTAN, J. A. Content-independent task-focused recommendation. *IEEE Internet Computing*, v. 5, n. 6, p. 40 – 47, 2001.

- HERLOCKER, J. L. et al. An algorithmic framework for performing collaborative filtering. In: *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. New York, NY, USA: ACM, 1999. p. 230–237. ISBN 1-58113-096-1.
- HERLOCKER, J. L. et al. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, ACM, New York, NY, USA, v. 22, n. 1, p. 5–53, 2004. ISSN 1046-8188.
- HILLMANN, D. *Using Dublin Core*. 2005. Disponível em: <http://www.dublincore.org/documents/usageguide/>. Acesso: Jan. 2008.
- HUANG, Z. Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Mining and Knowledge Discovery*, v. 2, n. 3, p. 283 – 304, 1998.
- HUANG, Z. et al. *A Graph-based Recommender System for Digital Library*. 2002. Disponível em: <<http://dlist.sir.arizona.edu/428/01/huang4.pdf>>.
- KAELBLING, L. P.; SAFFIOTTI, A. (Ed.). *IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, UK, July 30-August 5, 2005*. [S.l.]: Professional Book Center, 2005. ISBN 0938075934.
- LMPL-CNPQ. *Padronização XML*. 2008. Disponível em: <http://lmpl.cnpq.br/lmpl/?go=cv.jsp>. Acesso: Jan. 2008.
- LOPES, G. R. *Sistema de Recomendação para Bibliotecas Digitais Sob a Perspectiva da Web Semântica*. Dissertação (Mestrado) — Universidade Federal do Rio Grande do Sul, Instituto de Informática, Programa de Pós-Graduação em Computação, 2007.
- MANBER, U.; PATEL, A.; ROBISON, J. Experience with personalization of yahoo! *Commun. ACM*, ACM, New York, NY, USA, v. 43, n. 8, p. 35–39, 2000. ISSN 0001-0782.
- MARCONDES, C. H. Metadados: descrição e recuperação de informações na web. In: \_\_\_\_\_. 2. ed. [S.l.]: EDUFBA, 2006. cap. 5, p. 95–111.
- MARCONDES, C. H.; AND, L. B. T. H. K.; SAYÃO, L. *Bibliotecas Digitais: Saberes e Práticas*. 2. ed. [S.l.]: UFBA, 2006.
- MILLER, B. N. *GroupLens: An Open Architecture for Collaborative Filtering*. 1995. Disponível em: <[citeseer.ist.psu.edu/miller95grouplens.html](http://citeseer.ist.psu.edu/miller95grouplens.html)>.
- MONTANER, M.; LÓPEZ, B.; ROSA, J. L. D. L. A taxonomy of recommender agents on the internet. *Artif. Intell. Rev.*, Kluwer Academic Publishers, Norwell, MA, USA, v. 19, n. 4, p. 285–330, 2003. ISSN 0269-2821.
- MOONEY, R. J.; ROY, L. Content-based book recommending using learning for text categorization. In: *DL '00: Proceedings of the fifth ACM conference on Digital libraries*. New York, NY, USA: ACM, 2000. p. 195–204. ISBN 1-58113-231-X.
- OAI. *Open Archives initiative*. 2008. Disponível em: <http://www.openarchives.org/>. Acesso: Jan. 2008.
- OAI-PMH. *Open Archives Initiative - Protocol for Metadata Harvesting*. 2008. Disponível em: <http://www.openarchives.org/OAI/openarchivesprotocol.html>. Acesso: Jan. 2008.

- OWEN, S. *Taste Collaborative Filtering For Java*. 2007. Disponível em: <http://taste.sourceforge.net/>. Acesso: Jan. 2007.
- PAPAGELIS, M.; PLEXOUSAKIS, D. Recommendation based discovery of dynamic virtual communities. In: *CAiSE Short Paper Proceedings*. [S.l.: s.n.], 2003.
- PAZZANI, M. J. A framework for collaborative, content-based and demographic filtering. *Artif. Intell. Rev.*, Kluwer Academic Publishers, Norwell, MA, USA, v. 13, n. 5-6, p. 393–408, 1999. ISSN 0269-2821.
- PAZZANI, M. J.; BILLSUS, D. Learning and revising user profiles: The identification of interesting web sites. *Machine Learning*, v. 27, n. 3, p. 313–331, 1997. Disponível em: [citeseer.ist.psu.edu/pazzani97learning.html](http://citeseer.ist.psu.edu/pazzani97learning.html).
- QUEIROZ, S. R. M. *Recomendação para Grupos Através da Filtragem Colaborativa*. 2002. Trabalho de graduação em Inteligência Artificial. Recife, Pernambuco.
- REATEGUI, E. B.; CAZELLA, S. C. *Sistemas de Recomendação*. 2005. Disponível em: [http://www.addlabs.uff.br/enia\\_site/dw/sistemasrecomendacao.zip](http://www.addlabs.uff.br/enia_site/dw/sistemasrecomendacao.zip). Acesso: Jan. 2008.
- RESNICK, P.; VARIAN, H. R. Recommender systems - introduction to the special section. *Commun. ACM*, v. 40, n. 3, p. 56–58, 1997.
- ROCHA, L. M. Talkmine: a soft computing approach to adaptive knowledge recommendation. *Soft Computing Agents: New Trends for Designing Autonomous Systems.*, p. 89–116, 2001.
- S-Y., H.; W-C., H.; W-S., Y. Content-independent task-focused recommendation. *Online Information Review*, v. 27, n. 3, p. 169–182, 2003.
- SALTON, G.; BUCKLEY, C. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, v. 24, n. 5, p. 513 – 523, 1988.
- SALTON, G.; MCGILL, M. J. *Introduction to Modern Information Retrieval*. [S.l.]: McGraw-Hill, 1983.
- SALTON, G.; WONG, A.; YANG, C. S. A vector space model for automatic indexing. *Communications of the ACM*, v. 18, n. 11, p. 613 – 620, 1975.
- SARWAR, B. et al. Analysis of recommendation algorithms for e-commerce. In: . [S.l.]: ACM Press, 2000. p. 158–167.
- SHARDANAND, U.; MAES, P. Social information filtering: Algorithms for automating “word of mouth”. In: *Proceedings of ACM CHI’95 Conference on Human Factors in Computing Systems*. [s.n.], 1995. v. 1, p. 210–217. Disponível em: [citeseer.ist.psu.edu/article/upendra95social.html](http://citeseer.ist.psu.edu/article/upendra95social.html).
- SOMPPEL, H. V. de. The santa fe convention of the open archives initiative. *D-Lib Magazine*, 2000.
- TANG, T.; MCCALLA, G. Smart recommendation for an evolving e-learning system. In: *Workshop on Technologies for Electronic Documents for Supporting Learning*. [S.l.: s.n.], 2003.



TORRES, R. *Personalização na Internet*. [S.l.]: Novatec, 2004.

TORRES, R. et al. Enhancing digital libraries with techlens+. In: *JCDL '04: Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries*. New York, NY, USA: ACM, 2004. p. 228–236. ISBN 1-58113-832-6.

VIDOTTI, S. A. B. gregorio; SANT'ANA, R. G. Infra-estrutura tecnológica de uma biblioteca digital: elementos básicos. In: \_\_\_\_\_. 2. ed. [S.l.]: EDUFBA, 2006. cap. 4, p. 77–93.

## *Anexo*

### **Recuperação de Informação Utilizando o Modelo de Espaço Vetorial (VSM)**

Segundo (LOPES, 2007) o processo de recomendação pode ser visto como recuperação da informação, no qual documentos relevantes aos usuários devem ser recuperados e recomendados. Vários modelos de recuperação da informação foram propostos, como por exemplo, modelo booleano, modelo de espaço vetorial e modelo probabilístico (SALTON; MCGILL, 1983; BAEZA-YATES; RIBEIRO-NETO, 1999; GROSSMAN, 2004). Tais modelos consideram cada documento como um conjunto de termos de indexação, que são palavras ou expressões usadas para identificação e representação do conteúdo do documento. Uma simplificação adotada por tais modelos, é não considerar qualquer correlação entre os termos de indexação.

Para o presente trabalho, o VSM (*Vector Space Model*) é o modelo adotado por ser um modelo baseado em conteúdo, com pesos associados aos termos de indexação e cujo resultado da função de similaridade é dado na forma de *ranking*. Nesse modelo (SALTON; WONG; YANG, 1975), documentos e consultas são representados como vetores de termos de indexação, onde cada termo possui um peso associado, para prover distinção entre os termos de acordo com sua importância. Conforme descrito por (SALTON; BUCKLEY, 1988) os pesos podem variar entre 0 e 1, onde termos com pesos próximos a 1 possuem maior relevância enquanto que termos próximos a 0 possuem menor relevância. O VSM faz uso do espaço  $n$ -dimensional para representar os termos, onde  $n$  corresponde ao número de termos distintos. Para cada vetor de documentos ou consulta, os pesos representam as coordenadas do vetor na dimensão correspondente. O princípio do VSM é baseado na relação inversa entre a distância (ângulo) entre vetores de termos no espaço e a similaridade entre os documentos que eles representam, ou seja, quanto menor a distância entre os vetores de termos, maior a similaridade entre eles.

Para a realização do cálculo de similaridade, o co-seno (Equação 8.1) pode ser utilizado (produto escalar entre dois vetores, dividido pela multiplicação dos módulos desses

vetores). O resultado desse cálculo representa o grau de relevância entre a consulta ( $Q$ ) e o documento ( $D$ ), onde  $w$  representa os pesos dos termos contidos em  $Q$  e  $D$ , e  $t$  representa o número de termos (tamanho do vetor). Segundo Salton (SALTON; BUCKLEY, 1988), essa equação provê uma saída, classificada com base na ordem decrescente dos valores de similaridade obtidos.

Para entender o funcionamento de tal modelo, considere o seguinte exemplo: considere dois documentos,  $D_1(0,05;0,4;0,55)$  e  $D_2(0,0;0,5;0,45)$  representados no espaço tri-dimensional. Observa-se que os valores indicam as coordenadas dos termos nessas dimensões e representam os pesos associados a cada um dos termos na representação dos documentos. Considere também uma consulta  $Q(0,1;0,3;0,5)$ , onde o peso dos termos representa a importância de cada termo para a consulta. Aplicando-se a equação 8.1 para calcular a similaridade entre os documentos e a consulta realizada, os seguintes resultados são obtidos:

- *Similaridade* ( $Q, D_1$ ) = 0,9915 = 99,15%

- *Similaridade* ( $Q, D_2$ ) = 0,4899 = 48,99%

Observando-se o resultado obtido, percebe-se que o documento  $D_1$  é mais similar à consulta  $Q$ , do que o documento  $D_2$ .

$$\text{Similaridade}(Q, D) = \frac{\sum_{k=1}^t W_{qk} \cdot W_{dk}}{\sqrt{\sum_{k=1}^t (W_{qk})^2 \cdot \sum_{k=1}^t (W_{dk})^2}} \quad (8.1)$$