UNIVERSIDADE FEDERAL DE ALFENAS
INSTITUTO DE CIÊNCIAS EXATAS
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

Guilherme Machado Rodrigues

ALGORITMOS PARA OBTENÇÃO E MINERAÇÃO DE DADOS MATEMÁTICOS A PARTIR DO CONTEÚDO DA WIKIPEDIA

Alfenas, 27 de Junho de 2011.

UNIVERSIDADE FEDERAL DE ALFENAS
INSTITUTO DE CIÊNCIAS EXATAS
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

ALGORITMOS PARA OBTENÇÃO E MINERAÇÃO DE DADOS MATEMÁTICOS A PARTIR DO CONTEÚDO DA WIKIPEDIA

Guilherme Machado Rodrigues

Monografia apresentada ao Curso de Bacharelado em Ciência da Computação da Universidade Federal de Alfenas como requisito parcial para obtenção do Título de Bacharel em Ciência da Computação.

Orientador: Prof. Flavio Barbieri Gonzaga

Alfenas, 27 de Junho de 2011.

Guilherme Machado Rodrigues

ALGORITMOS PARA OBTENÇÃO E MINERAÇÃO DE DADOS MATEMÁTICOS A PARTIR DO CONTEÚDO DA WIKIPEDIA

A Banca examinadora abaixo-assinada aprova a monografia apresentada como parte dos requisitos para obtenção do título de Bacharel em Ciência da Computação pela Universidade Federal de Alfenas.

Prof. Ricardo Menezes Salgado

Universidade Federal de Alfenas

Profa. Mariane Moreira de Souza

Universidade Federal de Alfenas

Prof. Flavio Barbieri Gonzaga (Orientador)

Universidade Federal de Alfenas

Alfenas, 27 de Junho de 2011.

Dedico este trabalho ao meu pai, Luiz Cláudio, que me apoiou durante todo o tempo de faculdade, me ajudando nos momentos difíceis.

AGRADECIMENTO

Agradeço a todos meus familiares, amigos e a minha namorada pelo apoio a mim fornecido nos momentos em que precisei. Sem a ajuda deles este trabalho nunca teria chegado ao fim. Agradeço também ao meu professor e orientador Flávio Barbieri pelo tempo, dedicação e paciência em me orientar durante esse período do projeto. Agradeço a ele também, o voto de confiança dado a mim, ao me considerar uma pessoa competente para desenvolver tal trabalho. Agradeço a UNIFAL-MG e toda a equipe do curso de Bacharelado em Ciência da Computação pela qualidade do curso fornecido, qualidade esta que ajudou a me formar um profissional competente.

"UMA DAS FORÇAS DA INTERNET É SUA HABILIDADE DE AJUDAR OS CONSUMIDORES A ACHAR A AGULHA CERTA EM UM PALHEIRO DE DADOS DIGITAIS."

(JARED SANDBERG)

RESUMO

Em 1970, quando o termo *Internet* foi utilizado pela primeira vez por *Vinton Cerf* talvez não se tivesse noção do quanto e quão rápida esta idéia fosse crescer ao longo dos anos. Segundo *David Siegel "É preciso parar de encarar a Internet como uma rede de computadores. Ela é uma rede de pessoas.",* e de fato realmente é, visto que os usuários são os colaboradores, disponibilizam diversos tipos de conteúdo e tornam a *Internet* uma fonte de dados heterogênea. Dentro da natureza heterogênea da Internet, uma área em ascensão é a busca por assuntos específicos. Nesse escopo, um ramo ainda não contemplado é a busca por equações matemáticas, visto que a maioria dos sistemas de buscas existentes efetuam suas buscas de forma textual, tornando difícil a pesquisa por equações nesse emaranhado de dados. Nesse ponto, foi levantada a questão se existiria algum padrão dentro das áreas da matemática capaz de determinar com um grau de probabilidade a qual área uma determinada equação pertence de acordo com suas características. Separando esse conteúdo em áreas, o presente trabalho apresenta indícios de que a busca por esse tipo de conteúdo poderá ser simplificada. Esses padrões serão pesquisados sobre as equações existentes nas páginas matemáticas da *Wikipedia* através das técnicas de mineração de dados e da análise probabilística.

Palavras-Chave: Mineração de Dados, *Wikipedia* , equação, Matemática

ABSTRACT

In 1970, when the term Internet was first used by Vinton Cerf, might not had a sense of how much and how quickly this idea would grow over the years. According to David Siegel "People have to stop staring at the Internet as a computer network. It is a network of people." and in fact it really is, since users are collaborators, providing various types of content, making the Internet a source of heterogeneous data. Within the heterogeneous nature of the Internet, an area on the rise is the search for specific subjects. Within this scope, a branch that is not contemplated is the search for mathematical equations, because the vast majority of these systems performing searches of textual form. Thus it becomes difficult to search for the equation in this matted of data. At this point, the question arose whether there are any pattern within the areas of mathematics that could determine with a degree of probability which area belongs a certain equation according to their characteristics. Separating the content areas, this paper presents clues that the search for such content may be simplified. These patterns will be surveyed in the mathematical equations that are presents on the pages of Wikipedia through the techniques of Data Mining and Probabilistic Analysis.

Keywords: *Data Mining, Wikipedia, equation, mathematical*

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIAÇÕES

BD – Banco de dados

GB - *Gigabytes*

IP – *Internet Protocol*

KDD – K*nowledge Discovery in Databases*

LaReS – Laboratório de Redes e Sistemas Distribuídos da UNIFAL-MG

MB - *Megabytes*

MD – Mineração de Dados

PUC-Rio - Pontifícia Universidade Católica do Rio de Janeiro

RMI - *Remote Method Invocation*

SGBD – Sistema Gerenciador de Banco de Dados

SUMÁRIO

1
Introdução

Este capítulo apresenta na seção 1.1 a motivação deste projeto, na 1.2 os objetivos gerais e específicos e na 1.3 a organização desta monografia.

1.1 Motivação

Com o constante crescimento da *Internet,* a rede mundial de computadores, acessível apenas há alguns anos a partir de computadores pessoais (que possuíam um custo considerável), se populariza a cada dia, não apenas pela redução no custo dos computadores, mas também pelo aumento das possibilidades de acesso. Podendo ser acessada a partir dos mais diversos dispositivos, como por exemplo, *smartphones* e *tablets*, a grande rede se torna cada vez mais ubíqua, e como consequência, oferece uma maior possibilidade de interação com os usuários, que tendem a produzir ainda mais conteúdo disponibilizado *online*.

O aumento desse conteúdo passa por sistemas que surgem com a idéia de ter o usuário como um colaborador, não apenas mais como um leitor, conceito esse, aliás, conhecido como a *Web 2.0*.

> *"O termo Web 2.0 é utilizado para descrever a segunda geração da World Wide Web tendência que reforça o conceito de troca de informações e colaboração dos internautas com sites e serviços virtuais. A idéia é que o ambiente on-line se torne mais dinâmico e que os usuários colaborem para a organização de conteúdo."*
> *Jornal Folha de São Paulo - 10/06/2006*

A partir de *sites* como *wikis*[1] e redes sociais, os usuários produzem então conteúdo bastante heterogêneo, e cria-se um constante desafio no que tange a recuperação desse conteúdo de forma eficiente por parte de outros usuários.

[1] Coleção de muitas páginas interligadas e cada uma delas pode ser visitada e *editada* por qualquer pessoa.

Apenas exemplificando, efetuando-se buscas em diversos *sites*, é possível achar desde um desenho simples para uma criança cursando o primário colorir, até dados científicos para doutores ou estudantes que buscam defender suas teses. A grande questão é que novos tipos de conteúdo vêm surgindo, e alguns focos específicos ainda não possuem meios de serem encontrados/recuperados da rede.

Como a *Internet* possui várias informações e dados dos mais diversos tipos de assuntos e relacionados a diferentes áreas, então muitas vezes é difícil encontrar o que se procura visto que os mais conhecidos sistemas de buscas, como o buscador da *Google*, da *Yahoo* e o *Bing* da *Microsoft*, executam suas técnicas de busca de forma generalizada e podem mostrar dados que não condizem com o assunto pesquisado pelo usuário, dependendo da forma como o algoritmo de busca for construído. Pode-se citar como exemplo a busca por uma expressão matemática que tenha variância, que pode ser representada por *Var[X]*. Fica claro que um usuário que busque diretamente pelo termo "variância" terá resultados imediatamente relacionados ao assunto, mas um outro que busque pela representação como *Var[X]* receberá como resultado páginas que tratam da variância, mas também páginas que abordam declarações de variáveis em linguagens de programação, (com a palavra reservada *var*), já que existe semelhança na sintaxe de ambas (*var x : integer*, em *Pascal*).

Outro exemplo é a busca pelo elemento químico H_2O (água). Um pesquisador que necessite de alguma informação sobre tal elemento e efetue a busca no *Google*, por exemplo, terá como retorno, páginas do elemento químico e páginas relacionadas a cinema/televisão, devido ao fato de existirem filmes e séries com o nome desse elemento químico.

Esses exemplos demonstram uma dificuldade em se filtrar resultados referentes a conteúdos específicos. Quando se deseja efetuar pesquisas de uma forma geral, os buscadores são excelentes meios para se chegar aos resultados, pois seus bancos de dados possuem de milhares a bilhões de páginas. Desse modo para uma busca simples é improvável que não se consiga ter o resultado esperado.

Porém, quando se trata de um conteúdo especifico, por exemplo, uma pesquisa em sobre uma equação matemática, buscas de formas gerais acabam não retornando o resultado esperado devido ao fato de o buscador analisar de forma independente cada expressão da busca e também ao fato de o buscador não saber exatamente em qual área o usuário deseja efetuar a busca, se na área de Estatística ou na área de Linguagens de Programação, no caso de *Var[x]*.

Uma possível solução para esse problema é a separação do conteúdo da *Internet* em áreas. Dessa forma o usuário ao pesquisar por *Var[x]* poderia selecionar a área de Estatística, por exemplo, e o buscador retornaria apenas resultados referentes a essa área. Assim, os resultados referentes a Linguagens de Programação não apareceriam para o usuário. Existem sistemas com propósitos semelhantes que efetuam buscas apenas em um determinado tipo de material, como por exemplo, o *Google Scholar*[2] e o *Google Books*[3]. Esses sistemas diminuem sua área de busca visto que páginas na *Internet* não são inclusas na pesquisa. Este exemplo já mostra que existe uma demanda atual por buscas específicas. Contudo, deve-se observar que a essência da busca ainda continua sendo textual, mas em um escopo mais específico.

Também é possível encontrar disponível na *Internet*, bibliotecas eletrônicas de conteúdo mais específicos e que são menos heterogêneas do que toda a *Internet*, como por exemplo a *Wikipedia* (biblioteca onde pode-se encontrar páginas sobre diversos assuntos como história, geográfica, biológica, matemática, etc.), *DLMF*[4] e *MathWorld*[5] (bibliotecas com foco matemático). Porém, mesmo essas bibliotecas tendo seus conteúdos separados em áreas, ainda se encontram falhas semelhantes às descritas anteriormente quando se deseja pesquisar certos assuntos específicos, como o caso da busca por uma equação matemática. A equação $x^2 + \sin(30^o)$, se

---

2 Sistema de buscas do Google que efetua pesquisa apenas em artigos (scholar.google.com)

3 Sistema de buscas do Google que efetua pesquisa apenas em livros (books.google.com)

4 Digital Libray of Mathematical Functions – http://dlmf.nist.gov/

5 http://mathworld.wolfram.com/

for pesquisada na *Wikipedia* terá como retorno páginas que possuem apenas x^2 e/ou $sin(30^o)$, a própria equação, ou páginas não relacionadas (como por exemplo, sobre *X-Men*). Esse problema ocorre devido ao fato de essas ferramentas oferecerem apenas buscas por conteúdo textual, e não esperam receber como entrada na busca uma expressão matemática.

Em ferramentas de busca textual, se o usuário informar como entrada uma palavra, por exemplo, cachorro, a ferramenta pode utilizar do conceito de sinônimos, e retornar também páginas que possuam a palavra cão. Contudo, um usuário que deseje obter expressões matemáticas semelhantes a uma que ele tenha acabado de deduzir em sua pesquisa não poderá utilizar do mesmo princípio textual. Fica caracterizado então mais um ponto em que as ferramentas textuais não se adéquam a este tipo de conteúdo.

Para equações matemáticas existe uma técnica de busca feita por comparação de árvores [YANG. Rui et. al., 2005][YOUSSEF. Abdou. 2007]. A partir da equação informada pelo usuário, o sistema pode então produzir uma árvore da expressão, e buscar no banco de dados por árvores com características semelhantes. Essa árvore de equação é similar a árvore gerada por compiladores para análise de precedência, em que é necessário saber qual parte da equação deve ser executada primeiro. Para a equação $d+c+1/f\hat{}a$, ter-se-ia a seguinte árvore, exibida na Figura 1.

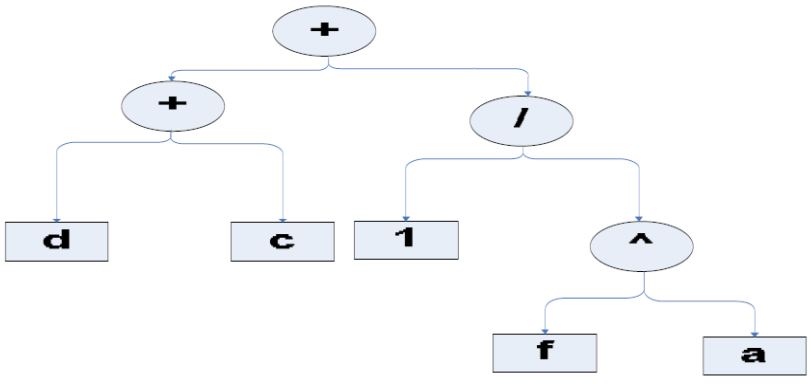


Figura 1: Árvore representado a equação d + c + 1 / f ^ a (YOUSSEF Abdou, 2007)

Fica claro neste ponto que equações equivalentes do ponto de vista semântico podem ser escritas de forma diferente, de modo a não serem equivalentes em sintaxe e, portanto, não darem origem a árvores semelhantes. Por exemplo, *1/x* e *x^-1*. Mas também pode-se observar que algumas características de equações são inerentes à área à qual as mesmas pertencem, e não mudam, mesmo que ocorra uma representação diferente. Por exemplo, uma equação que tenha o cálculo de uma integral pode possuir maior probabilidade de ser da área de Análise Matemática (que engloba Cálculo) ao invés de estar em outra área. Logo, pode não ser necessário comparar a árvore de uma equação digitada pelo usuário, com todas as árvores do banco de dados. Isso devido ao fato de poder existir padrões entre as equações que tornem possível determinar a qual área ela pertence analisando suas características, que é o foco deste trabalho.

Caso existam esses padrões, este trabalho estará contribuindo para o desenvolvimento de uma ferramenta de busca para expressões matemáticas, que receberia uma expressão matemática ao invés da tradicional entrada textual, retornando assim resultados mais condizentes com o que se procura.

1.2 Objetivos

1.2.1 Gerais

Verificar se é possível determinar a qual área uma equação pertence com base nas suas características.

1.2.2 Específicos

- Obter a base de dados da *Wikipedia*;
- Importar os dados para o *SGBD*;

- Desenvolver algoritmos para:
    - Analisar e extrair do conteúdo da *Wikipedia* apenas as páginas matemáticas;
    - Extrair das páginas matemáticas apenas as equações;
    - Organizar o conteúdo matemático em um novo *schema*, de modo a facilitar pesquisas futuras sobre o conteúdo matemático da *Wikipédia* a serem realizadas no *LaReS*;
    - Verificar as equações, de modo a catalogar as características de cada uma delas;
    - Usar o *software* de mineração de dados *Weka* para detectar possíveis padrões.
    - Utilizar a análise probabilística para comparar e/ou complementar os resultados da ferramenta *Weka*

1.3 Organização da Monografia

A partir deste ponto esta monografia terá a seguinte estrutura: O Capítulo 2 apresenta a *Wikipedia*, a história deste *site*, a estrutura de sua base de dados e uma estrutura simplificada proposta por esta monografia para facilitar o trabalho de pesquisadores. No Capítulo 3 será apresentado o conceito de Mineração de Dados, descrevendo sua importância, e também será apresentada a ferramenta para trabalho com Mineração de Dados chamada *Weka*. Será mostrada uma breve história desse *software* e suas funcionalidades. Este capítulo também traz informações sobre os arquivos gerados para serem analisados pelo *Weka* assim como define qual o algoritmo e configuração do *Weka* escolhida. O Capítulo 4 apresenta a análise probabilística, o algoritmo para gerar as probabilidades de cada operador em determinada área matemática de acordo com os dados obtidos na *Wikipedia* e os algoritmos para classificação das equações com base nos seus operadores e a técnica de análise probabilística. No Capítulo 5 serão mostrados

os resultados obtidos e as conclusões desta pesquisa e no Capítulo 6 os trabalhos futuros.

2
Obtenção e importação da *Wikipedia*

Este capítulo apresenta, na Seção 2.1 uma breve história da Wikipedia na Seção 2.2 é apresentado sua estrutura, mostrando suas principais tabelas e relações e na Seção 2.3 a estrutura mais simplificada, proposta por este trabalho.

2.1 Wikipedia

Com a ajuda do formato digital, as enciclopédias mudaram o conceito de pesquisa e armazenamento da informação. Antes da *Internet* se popularizar, elas pesavam quilos e ocupavam bastante espaço nas estantes das casas ou bibliotecas. A Figura 2 ilustra um livro onde foram impressas todas as páginas da *Wikipedia* que estão classificadas sob alguma categoria. Uma curiosidade é que apenas uma em cada 1.100 páginas da *Wikipedia* estão classificadas. Assim, se fosse impresso de fato todo o conteúdo, seriam necessários 1.100 livros do tamanho mostrado.

Dessa forma, hoje é possível encontrar as enciclopédias *online* que possuem um conteúdo em constante atualização, vantagem em relação às impressas, além é claro, de serem acessíveis de qualquer computador com conexão a *Internet*. Nesse meio pode se destacar a *Wikipedia,* enciclopédia já conhecida e bastante utilizada. Mantida através de doações, a mesma segue em constante crescimento desde a sua criação, fato esse que pode ser comprovado através da análise do tamanho do *backup* da sua base de dados, que pode ser obtido *online*[6].

[6] http://dumps.wikimedia.org/backup-index.html

Figura 2: Livro impresso com 3 mil artigos indicados pela Wikipedia (G1.Globo, 2011)

Dentro da *Wikipedia* é possível encontrar informações das mais diversificadas. Desde assuntos sobre a história do surgimento do universo até informações dos dias atuais. Esta é uma enciclopédia multilíngue *online* cujo conteúdo é livre e construído de forma colaborativa, em que qualquer usuário pode alterar ou incluir conteúdo (podendo passar por critérios de moderação antes de ser publicado), regido pelos termos da licença conhecida como *GNU/FDL ou GFDL*[7] (*Gnu Free Documentation License)*.

Seu projeto foi iniciado em Janeiro de 2001, na versão em língua inglesa, tendo como fundador *Jimmy Wales*[8], e sendo gerenciada pela Fundação *Wikimedia*[9]. Em um ano de existência já possuía quase 10 mil artigos e em Fevereiro de 2011, a *Wikipedia* estava disponível em 272 idiomas com mais de 17 milhões de artigos, sendo que 3.5 milhões são referente à versão inglesa e aproximadamente 680 mil artigos na versão portuguesa (*Wikimedia*, 2011). O total de páginas já ultrapassa os 67 milhões incluindo entre elas páginas de usuários, discussões, gestão dde projetos, entre outras. Segundo dados da própria *Wikpedia*[10], usando livros de 25 centímetros por 5 centímetros, com mais ou menos 400 páginas, dá aproximadamente 6 *MB* por volume. Como a versão da *Wikipedia* em inglês tem 4.4 *GB* de texto (em Outubro de 2006) tem-se então 750 volumes. E isso é uma medida

7 http://www.gnu.org/licenses/fdl.html

8 http://en.wikipedia.org/wiki/Jimmy_Wales

9 http://www.wikimedia.org/

10 http://en.wikipedia.org/wiki/File:Size_of_English_Wikipedia.svg

conservadora visto que não estão incluídos imagens e tabelas, que consomem muito mais espaço nas folhas.

Na Figura 3 é mostrado a página inicial da *Wikipedia* na qual é possível escolher em qual linguagem se deseja acessar o *site.*

Figura 3: Página inicial da Wikipedia, acessada em 03/06/2011

Apesar do fato de esta ser uma enciclopédia muito conhecida e talvez a mais utilizada, existem muitas críticas sobre de trabalhos feitos utilizando como referência a *Wikipedia*. Algumas pessoas defendem que ela não pode ser considerada como fonte de pesquisa para trabalhos científicos devido ao fato de muitas pessoas poderem alterar seus dados e escrever quaisquer informações tornando-os assim de pouca confiança. Porém muitos dos seus artigos possuem referencias de onde as informações foram tiradas (incluindo livros e artigos científicos). Desse modo é possível utilizar a *Wikipedia* como um meio para se obter informações científicas sobre determinado assunto, com a vantagem de que muitos dos temas já se encontram classificados. A classificação da *Wikipedia* é ainda outro tema de estudo. Trabalhos propõem mecanismos para reorganizar a classificação

de artigos (SUCHANEK, Fabina M., 2008)(uma vez que a classificação também ocorre de forma colaborativa), mas esse não é o foco do presente trabalho.

2.2 Estrutura da *Wikipedia*

Devido ao fato de à *Wikipedia* ser uma enciclopédia colaborativa onde qualquer usuário pode incluir arquivos, isso torna seu banco de dados de difícil manutenção. É necessária uma boa estrutura de *BD* para que não ocorra nenhum problema que ocasione a perda de dados ou resulte em queda de desempenho. No *link*[11] é possível encontrar o modelo completo do *BD* da *Wikipedia*.

Para um pesquisador ou usuário que queira efetuar pesquisas ou trabalhos sobre o conteúdo disponibilizado pela *Wikipedia* é possível baixar seu conteúdo[12] de forma livre. Porém são *Gygabytes* de arquivos, sendo necessários alguns meses estudando a estrutura do *BD* para que se possa começar a extrair o conteúdo desejado. Dessa forma, para se extrair apenas o conteúdo da área Matemática da base de dados da *Wikipedia,* foi necessário o estudo dessas tabelas, de forma a poder se extrair as páginas, relações de *links*, categorias e relações de categorias com as páginas e as próprias categorias.

Algumas dessas tabelas podem ser consideradas como as tabelas chaves, pois armazenam as principais informações. Segue uma breve descrição dessas tabelas:

- *Page*: Esta tabela pode ser considerada o "núcleo da *wiki*". Cada página tem uma entrada aqui, que a identifica por título e contém alguns metadados essenciais.

11 http://www.mediawiki.org/wiki/Manual:Database_layout

12 http://dumps.wikimedia.org/backup-index.html

- *PageLinks:* Esta tabela contém as referencias de *links* internos das páginas da *Wikipedia*.
- *Category:* Possui todas as categorias existentes na *Wikipedia*.
- *CategoryLinks:* Armazena os *links* de páginas para as categorias.
- *Revision:* Mantém um *log* de cada mudança feita nas páginas da *Wikipedia.* Armazena informações como o usuário que fez a edição, o momento em que a edição foi feita e uma referência para o texto novo na tabela *text*.
- *Text:* Mantém o código *HTML* de cada página.
- *Image:* Mantém as imagens e outros arquivos carregados. No entanto, as páginas de descrição das images são armazenados como outras páginas.
- *ImageLinks:* Esta tabela contém todos os links para os arquivos.
- *User:* Armazena informações dos usuários cadastrados.
- *User_groups:* Armazena informações de cada grupo de usuários como os seus privilégios e permissões de acessos no *site.*
- *Ipblocks:* Armazena detalhes dos endereços de *IP* e usuários que foram bloqueados pelo *site* para fazerem alterações.

Após essa breve descrição sobre as principais tabelas da *Wikipedia*, abaixo segue uma descrição mais detalhadas das tabelas necessárias e utilizadas por este trabalho para recuperação do conteúdo Matemático.

- *Page*:
  - *page_id*: Atributo inteiro, chave primária da tabela;
  - *page_namespace*: O nome da página é dividido em *namespace* e *title*. Este atributo contém um código do *namespace* da página. Esse código é o que diferencia duas páginas de mesmo nome, mas hierarquia diferente. Exemplo: Pode-se ter o Portal *Mathematics*, e a página de conteúdo *Mathematics*. Nesse caso, ambos terão o mesmo atributo *title* (*Mathematics*), mas diferentes valores de *namespace*[13]. Os mais comuns são: 0 para páginas de conteúdo "real" e artigos, 14 para páginas de

[13] http://en.wikipedia.org/wiki/Wikipedia:Namespace

categorias, 2 para páginas de usuários, 12 para páginas de ajuda e 100 para portais;

- *page_title*: É o título da página limpa, sem o seu *namespace*.

- *Revision*:
  - *rev_id:* Atributo inteiro, chave primária da tabela;
  - *rev_page:* Atributo inteiro, chave estrangeira para tabela *Page;*
  - *rev_text_id:* Atributo inteiro, chave estrangeira para tabela *Text.*

- *Text:*
  - *old_id:* Atributo inteiro, chave primária da tabela;
  - *old_text:* Fica armazenado o código *HTML* da página.

- *PageLinks*:
  - *pl_from*: Atributo chave estrangeira para *page_id* em *Page*. É o *id* da página origem que possui um *link* para outras páginas;
  - *pl_namespace*: Atributo chave estrangeira para *page_namespace* em *Page*. É o *namespace* da página destino, página que está sendo referenciada pelo *link*;
  - *pl_title*: Atributo chave estrangeira para *page_title* em *Page*. É o campo *title* da página destino, página que está sendo referenciada pelo *link* .

- *Category*:
  - *cat_id:* Atributo inteiro, chave primária;
  - *cat_title:* Nome da categoria;
  - *cat_pages:* Número de páginas na categoria;
  - *cat_subcats:* Número de sub-categorias na categoria.

- *CategoryLinks*:
  - *cl_from*: Armazena o *page_id* da página onde possui o *link* para a categoria;
  - *cl_to*: Armazena o nome da categoria indicada.

2.3 Estrutura proposta

Como mostrado na Seção 2.2, a *Wikipedia* possui um *BD* grande com várias tabelas o que torna o trabalho não trivial para pesquisadores que queiram efetuar pesquisas sobre uma determinada área. Dessa forma, esta pesquisa propõe uma nova estrutura, simplificada, de modo que fique armazenado apenas as áreas de interesse do pesquisador. Abaixo, na Figura 4, é apresentada a estrutura simplificada seguida pelas descrições das tabelas e os algoritmos para gerar as respectivas tabelas de modo que no final se tenha um novo *BD* com apenas os dados necessários para a pesquisa desta monografia.

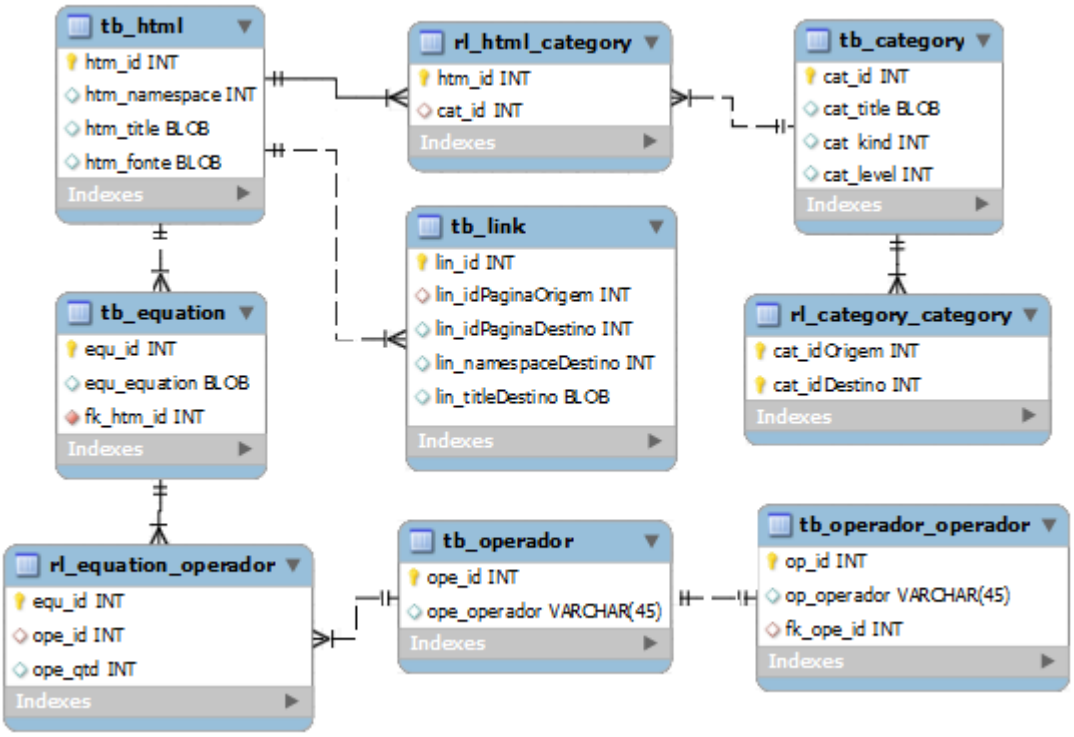


Figura 4: Estrutura simplificada

Abaixo segue a descrição de cada da tabela:

- *tb_html*: Armazena as páginas da *Wikipedia* que são de categorias da Matemática:

- *htm_id*: Atributo inteiro, chave primária;
- *htm_namespace:* O nome da página é dividido em *namespace* e *title*. Este atributo contém um código do *namespace* da página;
- *htm_title*: É o título da página limpa, sem o seu *namespace*;
- *htm_fonte:* Código fonte em *HTML*.

- *tb_link:* Armazena os *links* entre as páginas:
  - *lin_id*: Atributo inteiro, chave primária;
  - *lin_idPaginaOrigem:* Chave estrangeira, contém o código da página onde o *link* está;
  - *lin_idPaginaDestino*: Chave estrangeira, contém o código da página para a qual o *link* aponta;
  - *lin_namespaceDestino*: *namespace* da página para a qual o *link* aponta;
  - *lin_titleDestino*: *title* da página para a qual o *link* aponta.
- *tb_equation:* Armazena as equações extraídas das páginas:
  - *equ_id:* Atributo inteiro, chave primária;
  - *equ_equacao*: Equação extraída da página;
  - *fk_html_equacao*: chave estrangeira, relacionando a equação com a página onde a mesma está contida.
- *tb_category:* Armazena as categorias e subcategorias da *Wikipedia* referentes a Matemática:
  - *cat_id*: Atributo inteiro, chave primária;
  - *cat_title*: Nome da categoria;
  - *cat_kind*: Pode ser 1, 2 ou 3. Se for categoria matemática é 1. Se for categoria de matemáticos é 2. Se for categoria relacionada a matemática é 3;[14]
  - *cat_level*: Nível da categoria considerando a árvore de categorias. Categorias que estão no topo da arvore são nível 0, as subcategorias destas são nível 1. As subcategorias das categorias de nível 1 são de nível 2, e assim por diante. No nível 0 se encontra o Portal *Mathematics*.

[14] http://en.wikipedia.org/wiki/List_of_mathematics_categories

- *tb_operador*: Contém os possíveis operadores presentes nas equações matemáticas:
  - *ope_id*: Atributo inteiro, chave primária;
  - *ope_operador*: Armazena o operador. Exemplo: +, -, *, *and, \lim*.
- *rl_html_category*: Mantém as relações que indicam a quais categorias cada página pertence:
  - *htm_id*: Chave estrangeira, código da página;
  - *cat_id*: Chave estrangeira, código da categoria.
- *rl_category_category*: Mantém as relações entre as categorias. Através desta tabela é possível se montar a árvore de relações entre as categorias:
  - *cat_idOrigem*: Chave estrangeira, categoria que referencia outra. Está em um nível acima da categoria referenciada na árvore;
  - *cat_idDestino*: Chave estrangeira, categoria referenciada por outra. Está em um nível abaixo da categoria referenciadora na árvore.
- *rl_equation_operador*: Mantém a relação de quais operadores estão presentes na equação:
  - *equ_id*: Chave estrangeira, código da equação;
  - *op_id*: Chave estrangeira, código do operador;
  - *op_qtd*: Quantidade de vezes que o operador aparece na equação.
- *tb_operador_operador*: Um mesmo operador, por exemplo a multiplicação, pode ser representado de diferentes formas em linguagem *Tex* como *, *\times, \cdot*. Assim, essa tabela possui para esses casos, operadores que são diferentes do ponto de vista de sintaxe, mas equivalentes do ponto de vista semântico. Por exemplo, relacionado ao operador * na *tb_operador*, teria aqui nessa tabela as entradas informando que *\times* e *\cdot* são equivalentes a ele:
  - *op_id*: Atributo inteiro, chave primária;
  - *op_operador*: Contém o operador;
  - *fk_op_id*: Chave estrangeira para *tb_operador*.

  Os algoritmos para preenchimento destas tabelas são descritos abaixo:

Marcação das categorias Matemáticas

O algoritmo 1 lê os nomes das categorias da matemática contidas na página[15] e realiza a marcação no banco das categorias que compõem a matemática. Os nomes das categorias estão contidos no código *HTML* desta página, entre uma expressão regular que começa com '*:Category:*' e termina com ' |' ou ']]'. A marcação é feita através da seguinte distribuição no campo temporário denominado *math* criado em *category*:

1 - se categoria matemática;

2 - se categoria de matemáticos;

3 - se categoria relacionada a matemática.

Esse atributo será utilizado por outros algoritmos descritos posteriormente para recuperação de páginas e categorias da matemática.

Algoritmo 1

```
1.  Q ← Selecionar page_id de page
2.                onde page_title = 'List_of_mathematics_categories';
3.  P ← Selecionar rev_text_id de revision onde rev_page = Q.page_id;
4.  E ← Selecionar old_text de text onde old_id = P.rev_text_id;
5.  D ← Pesquisar por expressão regular começando com :Category:'
6.                       e terminando em '|' ou ']]' em E.old_text;
7.  para cada resultado em D faça
8.          category = D.expressao;
9.          V ← verificar se category é do tipo 1, 2 ou 3
10.                              de acordo com posição na página
11.         se V == 1
12.                Atualizar category setar math = 1 onde cat_title = category;
13.         se V == 2
14.                Atualizar category setar math = 2 onde cat_title = category;
15.         se V == 3
16.                Atualizar category setar math = 3 onde cat_title = category;
```

[15] http://en.wikipedia.org/wiki/List_of_mathematics_categories

Povoamento da tabela *tb_html*:

O algoritmo 3 recupera as páginas da área matemática do *BD* da *Wikipedia* e armazena na *tb_html* da estrutura proposta. Para isso, é necessário fazer uma junção das tabelas *page, revision* e *text* para que se possa recuperar o *page_id, page_namespace, page_title* e *old_text.* Porém antes disso é necessário marcar na tabela *page* quais são as páginas da Matemática. Para essa função, foi criado um atributo, denominado m*ath*, e se a página for da matemática é atribuído o valor 1 a ele, se não fica com o valor 0.

O algoritmo 2 é para fazer a marcação nas páginas relacionadas à matemática. Este algoritmo utiliza de uma mesma marcação na tabela *category* onde foi criado um atributo, denominado *math*, para se marcar as categorias que são da matemática.

Algoritmo 2

```
1.  Q ← Selecionar cat_title de category onde math = 1;
2.  para cada resultado em Q faça
3.          cat_title = Q.cat_title;
4.          P ← Selecionar cl_from de categorylinks onde cl_to = cat_title
5.          para cada resultado em P faça
6.                  page_id = P.cl_from
7.                  Atualizar page setar math = 1 onde page_id = page_id
8.          Atualizar page setar math = 1
9.                  onde page_title = cat_title e page_namespace = 14;
```

Algoritmo 3

```
1.  Q ← Selecionar page_id, page_namespace, page_title e old_text de page, text
2.       e revision onde page_id = rev_page e rev_text_id = old_id e math = 1
3.  para cada resultado em Q faça
4.          page_id = Q.page_id;
5.          page_namespace = Q.page_namespace;
6.          page_title = Q.page_title;
7.          old_text = Q.old_text;
8.          Inserir em tb_html valores
9.                  page_id, page_namespace, page_title e page_title
```

Povoamento da tabela *tb_link*:

Este algoritmo, 4, recupera todas as páginas matemáticas que possuem *links* para outras páginas matemáticas e armazena na *tb_link*. Para isso, foi criado um atributo temporário, denominado *lin_math,* que recebe 1 quando se atualiza o *link* de destino, dessa forma, os *links* que forem páginas matemáticas para páginas matemáticas terão *math* = 1. Ao final da execução do algoritmo basta excluir as tuplas com *math* diferente de 1 para retirar links com referencias para páginas não matemáticas.

Algoritmo 4

```
1.  Q ← Selecionar pl_from, pl_namespace e pl_title de pagelinks e page
2.                                        onde pl_from = page_id e math = 1;
3.  para cada resultado em Q faça
4.         inserir em tb_link valores
5.                         (0, Q.pl_from, 0, Q.pl_namespace, Q.pl_title, 0, 0);
6.   Q ← Selecionar htm_id, htm_namespace, htm_title de tb_html
7.   para cada resultado em Q faça
8.         htm_namespace = Q.htm_namespace;
9.         htm_id = Q.htm_id;
10.        htm_title = Q.htm_title;
11.        P ← Selecionar lin_id de tb_link onde
12.           lin_namespaceDestino = htm_namespace e lin_titleDestion = htm_title
13.        para cada resultado em P faça
14.              lin_id = P.lin_id;
15.              atualizar tb_link setando lin_math = 1 e
16.                       lin_idPagindaDestino = htm_id onde lin_id = lin_id
17.  deletar em tb_link onde lin_math = 0
```

Povoamento da tabela *tb_equation:*

Este algoritmo, 5, recupera as equações presentes nas páginas de matemática da *Wikipedia*. As equações estão entre as *tags* <*math*> e </*math*> no código *HTML* das páginas.

Algoritmo 5

```
1.  Q ← Selecionar htm_id, htm_fonte de tb_html
2.  para cada resultado de Q faça
3.         htm_id = Q.htm_id;
```

```
4.          htm_fonte = Q.htm_fonte;
5.          E ← buscar por expressões regulares entre
6.                                          <math> e </math> em htm_fonte
7.          para cada resultado em E faça
8.                 equacao = E.proximaEquacao();
9.                 se equacao não existir em tb_equation
10.                       Inserir em tb_equation valores (0, equacao, htm_id);
```

Povoamento da tabela *tb_category*

Para se gerar esta tabela basta ler as categorias em *category* onde o campo *math* for diferente de 0. O Algoritmo 6 faz exatamente isso.

Algoritmo 6

```
1.  Q ← Selecionar cat_id, cat_title e math de category onde math > 0;
2.  para cada resultado em Q faça
3.          cat_id = Q.cat_id;
4.          cat_title = Q.cat_tile;
5.          math = Q.math;
6.          Inserir em tb_category valores (cat_id, cat_title, math);
```

Povoamento das tabelas *tb_operador* e *tb_operador_operador*:

Estas tabelas não possuem algoritmos, elas foram geradas através dos operadores relacionados em (KNUTH, Donald Ervin. 1984), visto que este livro apresenta os operadores em notação *Tex* e as equações apresentadas nas páginas da *Wikipedia* são representadas em notação *Tex*.

Povoamento da tabela *rl_equation_operador*

Este algoritmo, 7, busca em cada equação quais são os operadores presentes e quantas vezes eles aparecem.

Algoritmo 7

```
1.  Q ← Selecionar todas as equações de tb_equation
2.  P ← Selecionar todos os operadors em rl_operador_operador
3.  para cada resultado em Q faça
4.         equ_id = Q.equ_id;
5.         equ_equation = Q.equ_equation;
```

```
6.          para cada resultado em P faça
7.                  ope_qtd = 0;
8.                  op_operador = P.op_operador;
9.                  op_id = P.op_id;
10.                 opw_id = P.fk_ope_id;
11.                 E ← buscar por expressões regulares
12.                                 de op_operador em equ_equation
13.                         para cada resultado em E faça
14.                                 ope_qtd = ope_qtd + 1;
15.                 se ope_qtd != 0
16.                         inserir em rl_equation_operador valores
17.                                 (equ_id, opw_id, ope_qtd);
```

Povoamento tabela *rl_html_category*:

Este algoritmo, 8, escreve nesta tabela a relação entre as páginas e as categorias da matemática. Em *categorylinks* já se encontra a relação de todas as páginas da *Wikipedia* com as suas categorias. Basta pesquisar nessa tabela pelas páginas da matemática e ver quais são as categorias da matemática relacionadas.

Algoritmo 8

```
1.  Q ← Selecionar htm_id de tb_html
2.  para cada resultado em Q faça
3.          htm_id = Q.htm_id;
4.          P ← Selecionar * de category_links onde cl_from = htm_id e math = 1
5.          para cada resultado em P faça
6.                  cl_to = P.cl_to;
7.                  E ← Selecionar cat_id de tb_category onde cat_title = cl_to;
8.                  para cada resultado em E faça
9.                          cat_id = E.cat_id;
10.                         Inserir em rl_html_category valores (htm_id, cat_id);
```

Povoamento da tabela *rl_category_category:*

Este algoritmo, 9, que grava nesta tabela a relação entre categorias, estabelecendo o nível de cada uma. A categoria M*athematics* recebe o nível 0. Categorias que se relacionam com ela recebem 1, e assim por diante. Cada página de categoria possui em seu código *HTML* o nome das categorias que as indicam, e

os nomes estão entre a expressão regular começando com *'[[Category:'* e terminando com '] ' ou ' | '.

Algoritmo 9

```
1.  Q ← Selecionar htm_fonte, htm_title de tb_html onde htm_namespace = 14;
2.  para cada resultado em Q faça
3.        htm_fonte = Q.htm_fonte;
4.        htm_title = Q.htm_title;
5.        P ← Pesquisar por expressão regular que começa com  '[[Category:'
6.                          e termina com  ']' ou  '|' ;
7.        para cada resultado em P faça
8.              cat_idOrigem = Selecionar cat_id de tb_category
9.                                  onde cat_title = P.expressao;
10.             cat_idDestino =  Selecioanr cat_id de tb_category
11.                                 Onde cat_title = htm_title;
12.             Inserir em rl_category_category
13.                           valores (cat_itOrigem, cat_idDestino));
14.  Atualizar tb_html h, tb_category c, rl_html_category r setar c.cat_level = 0
15.       onde h.htm_namespace = 100 e h.htm_title = 'Mathematics'
16.       e h.htm_id = r.htm_id e r.cat_id = c.cat_id e c.cat_kind = 1;
17.  nivel = 0;
18.  faça
19.       C ← Selecionar cat_idDestino de rl_category_category
20.             onde cat_idOrigem in (Selecionar cat_id de tb_category
21.                           onde cat_level = nivel)
22.       Se (C == vazio)
23.             pare a execução;
24.       senão
25.             para cada resultado em C faça
26.                   cat_idDest = C.cat_idDestino;
27.                   Atualizar tb_category setar cat_level = nivel + 1
28.                         onde cat_id = cat_idDest;
29.       nivel = nivel + 1;
```

3 Mineração de dados e o *Weka*

Este capítulo apresenta na seção 3.1 o conceito de Mineração de Dados, na Seção 3.2 é apresentada a ferramenta Weka e suas funcionalidades, na Seção 3.3 é mostrado o arquivo construído para ser analisado pelo Weka e na Seção 3.4 é apresentada a configuração da ferramenta e o algoritmo escolhido para a analise

3.1 Mineração de Dados

Grandes empresas normalmente possuem grandes bancos de dados, onde ficam armazenados dados de seus clientes, relações de compras e vendas, dados da empresa, etc. Esses dados costumavam não serem aproveitados devido ao fato de que se gastava muito tempo para analisá-los, principalmente há alguns anos atrás quando a tecnologia não suportava ou gastava muito tempo para trabalhar com grandes quantidades de dados. Porém, a tecnologia vem evoluindo de forma muito rápida, surgindo computadores capazes de trabalhar com *Gigabytes* de dados em questões de segundos. Com isso surgiu uma técnica denominada Mineração de Dados (MACHADO Carlos, 1999), que busca em grandes quantidades de dados padrões ou informações que sejam relevantes para uma aplicação, mas que não sejam triviais de serem visualizadas.

A Mineração de Dados teve início por volta dos anos 80. Nos seus primórdios, a Mineração era essencialmente extrair informação de grandes bases de dados de uma maneira automatizada. Com o passar do tempo e a demanda por informações mais relevantes, essa técnica passou a trabalhar mais nesses dados, utilizando algoritmos mais eficientes capazes de identificar melhores padrões na base de dados, ajudando assim as empresas em suas campanhas de *marketing* e

conquista de novos clientes. Segundo Sandra de Amo, pode-se destacar como os seguintes pontos importantes a razão pelo qual a Mineração vem se tornando necessária:

- Os volumes de dados são muito importantes para um tratamento utilizando somente técnicas clássicas de análise
- O usuário final não é necessariamente um estatístico
- A intensificação do tráfego de dados aumenta a possibilidade de acesso aos dados

É preciso esclarecer os pontos que diferem a Mineração de Dados de *Business Intelligence* e de *KDD*, pois algumas pessoas confundem os seus significados. Suponha uma empresa de meio porte que vende variados tipos de produtos. Essa empresa deseja saber quais produtos são os mais vendidos, dados estatísticos referentes às vendas diárias de cada um deles, etc., para poder planejar estratégias referentes à reposição de estoque, qual produto fica em prateleiras mais acessíveis, quais ficam em prateleiras mais altas, etc.. Nesse caso se aplicaria *Business Intelligence*, que são análises voltadas aos planos estratégicos da empresa. Se a empresa deseja saber informações melhores sobre seus clientes, padrões de compras como, por exemplo, toda vez que algum cliente compra o produto A, ele também compra o produto B, dessa forma a empresa poderia posicionar o produto A, perto do produto B para facilitar e agilizar o processo do cliente. Neste caso se aplica Mineração de Dados, pois é necessária uma análise mais aprofundada nos dados à procura de padrões para auxiliar a empresa no plano tático.

A diferença de um *KDD*, é que este é um processo mais amplo constituído de mais etapas, que são descritas em (AMO. Sandra) como sendo:

- Limpeza dos dados: etapa onde são eliminados ruídos e dados inconsistentes.
- Integração dos dados: etapa onde diferentes fontes de dados podem ser combinadas produzindo um único repositório de dados.

- Seleção: etapa onde são selecionados os atributos que interessam ao usuário. Por exemplo, o usuário pode decidir que informações como endereço e telefone não são relevantes para decidir se um cliente é um bom comprador ou não.
- Transformação dos dados: etapa onde os dados são transformados num formato apropriado para aplicação de algoritmos de mineração (por exemplo, através de operações de agregação).
- Mineração: etapa essencial do processo consistindo na aplicação de técnicas inteligentes a fim de se extrair os padrões de interesse.
- Avaliação ou Pós-processamento: etapa onde são identificados os padrões interessantes de acordo com algum critério do usuário.
- Visualização dos Resultados: etapa onde são utilizadas técnicas de representação de conhecimento a fim de apresentar ao usuário o conhecimento minerado.

Como se pode ver a Mineração de Dados é uma etapa dentro do *KDD*, responsável pela extração e localização de padrões que sejam relevantes para a empresa.

Pode-se descrever dois casos reais em que a mineração de dados trouxe benefícios para a empresa citados em um artigo do *IME* cujo autor é desconhecido:

Na *Walmart* a mineração de dados identificou um hábito curioso dos consumidores. Ao procurar eventuais relações entre o volume de vendas e os dias da semana, o software apontou que, às sextas-feiras, as vendas de cervejas cresciam na mesma proporção que as de fraldas. Uma investigação mais detalhada revelou que, ao comprar fraldas para seus bebês, os pais aproveitavam para abastecer as reservas de cerveja para o final de semana. Fica claro a existência de um padrão e uma possível atitude a ser tomada é manter sempre o estoque de cerveja e fralda cheios e um produto perto do outro.

Outro caso é o do vestibular da *PUC-Rio*. Utilizando as técnicas da mineração de dados, o *software,* depois de examinar milhares dados de alunos, forneceu a seguinte regra: se o candidato é do sexo feminino, trabalha e teve aprovação com boas notas no vestibular, então não efetivava a matrícula. Trata-se de um regra bastante estranha, mas uma reflexão justificava a regra oferecida pelo programa: de acordo com os costumes do Rio de Janeiro, uma mulher em idade de prestar vestibular, se trabalha é porque precisa, e neste caso deve ter também feito inscrição para ingressar na universidade pública gratuita. Se teve boas notas na *PUC-Rio* então provavelmente foi aprovada na universidade pública onde efetivará matrícula. Claro que há exceções, mas a grande maioria obedece à regra anunciada, e assim a *PUC-Rio* poderá, com esse conhecimento, já ter uma noção de quais alunos muito provavelmente não efetivarão a matricula e se preparar para tomar as medidas necessárias.

3.2 A Ferramenta *Weka*

O *Weka*[16] (*Waikato Environment for Knowledge Analysis*) teve seu início em 1993 sendo desenvolvido pela Universidade de Wakata, Nova Zelândia, e em 2006 o *software* foi adquirido pela empresa *Pentaho Corporation*. As primeiras versões foram desenvolvidas na linguagem *TCL/TK* e em *C*, porém em 1997 (surgimento do *Weka* 3) começou a ser desenvolvido na linguagem *Java*.

Este é um *software* que executa algoritmos provenientes de diferentes sub-áreas da Inteligência Artificial em uma base de dados. Esses algoritmos buscam aprender/obter conhecimento através de análises nos dados, técnica esta conhecida como Mineração de Dados (*Data Mining*), já descrita acima. Esses algoritmos estão dispostos em três interfaces diferentes do *Weka*: *The Explorer*, *The Knowledge Flow*,

[16] http://www.cs.waikato.ac.nz/ml/weka/

The Experimenter. Existe ainda a possibilidade de se utilizar o *Weka* por linha de comando. Uma breve descrição de cada interface é apresentada a seguir [WITTEN, Ian. 2005]:

- *The Explorer*: Interface mais simples e mais fácil de se utilizar do *Weka*, tornando-a assim a mais popular e escolhida pelos usuários. É possível ler arquivos *.ARFF* (padrão do *Weka*) e gerar árvores de decisão a partir dos dados. O *Explorer* também fornece guias rápidos para se trabalhar com pré-processamento, *clustering*, classificação, regressão, associação, visualização e seleção de recursos. As mais utilizadas são:
    - Pré-Processamento: É a analise inicial do *Weka*. Exibe a quantidade de instâncias do arquivo, dados referentes a cada atributo, algumas estatísticas mais simples, entre outras funcionalidades.
    - *Clustering*: Algoritmos para procurar grupos com certos padrões entre os dados. Consiste em dado uma base de dados X, agrupar os elementos de X de modo que objetos mais similares fiquem no mesmo *cluster*.
    - Classificação: prevêem uma ou mais variáveis discretas, com base nos outros atributos do conjunto de dados.
    - Regressão: prevêem uma ou mais variáveis contínuas, como lucro ou perda, com base nos outros atributos do conjunto de dados.
    - Associação: encontram correlações entre atributos diferentes em um conjunto de dados. A aplicação mais comum desse tipo de algoritmo é para criar regras de associação.
- *The Knowledge Flow*: Permite criar configurações de processamento de dados transmitidos, especificar um fluxo de dados através da ligação dos componentes que representam fontes de dados, entre outras. A diferença sobre o *Explorer*, é que este mantém tudo na memória

principal, ou seja, só pode ser aplicado para problemas de pequeno e medio porte.

- *The Experimenter*: Projetado para auxiliar na resposta de questões básicas para a aplicação de técnicas de classificação e de regressão tais como: Quais são os métodos e valores de parâmetros que trabalham melhor para o problema dado? Usuários avançados podem utilizar esta interface para distribuir a carga em várias máquinas usando *RMI*.

Além desses modos para se utilizar o *Weka*, ainda é possível chamá-lo através de outro programa *Java*, reutilizando seu código, visto que o *Weka* é um software sob a Licença *GNU General Public*, tendo seu código aberto e passível de ser alterado.

O *Weka* aceita arquivos *.CSV*, *.ARFF*, dados de um *link* e dados de uma tabela de um banco de dados, porém o arquivo padrão é o *.ARFF*. Este arquivo é em formato texto, podendo ser lido por qualquer editor. A seguir é apresentada a estrutura do mesmo:

- A primeira linha deve conter *@Relation nome_ da_ relação;*
- As linhas abaixo são para definição dos atributos (colunas) dos dados. Pode haver vários atributos, porém devem seguir a estrutura *@Attribute nome_do_atributo TIPO*, onde TIPO pode ser *Real*, *Integer*, etc. O último atributo é o atributo classe;
- Após essas linhas deverá vir *@Data*, informando que daquele momento em diante virá os dados;
- E em seguida os dados propriamente ditos, separados por vírgula:

  Ex.: 5.1,3.5,1.4,0.2,Iris-setosa

A Figura 5 mostra um exemplo de um arquivo *.ARFF*. As Figuras 6 e 7 mostram a escolha de interfaces do *Weka* e o *Explorer* com o arquivo contendo os dados deste trabalho carregados, respectivamente.

```
@RELATION iris
@ATTRIBUTE sepallength  REAL
@ATTRIBUTE sepalwidth   REAL
@ATTRIBUTE petallength  REAL
@ATTRIBUTE petalwidth   REAL
@ATTRIBUTE class    {Iris-setosa,Iris-versicolor,Iris-virginica}
@DATA
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
5.0,3.3,1.4,0.2,Iris-setosa
7.0,3.2,4.7,1.4,Iris-versicolor
6.4,3.2,4.5,1.5,Iris-versicolor
5.7,2.8,4.1,1.3,Iris-versicolor
6.3,3.3,6.0,2.5,Iris-virginica
5.8,2.7,5.1,1.9,Iris-virginica
6.3,2.9,5.6,1.8,Iris-virginica
```

Figura 5: Arquivo iris.arff

Figura 6: Seleção de interfaces do Weka

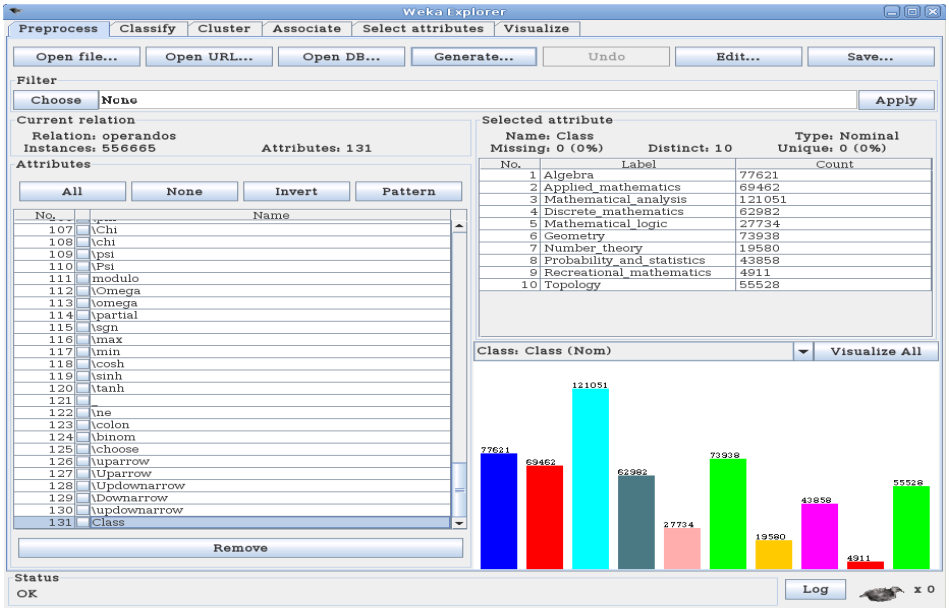


Figura 7: Arquivo operadores.arff carregado no Explorer

3.3 Arquivo construído para ser analisado

Para pesquisar por padrões nas equações optou-se por construir um arquivo contendo os dados a serem analisados, nos padrões que o *Weka* exige, para que se pudesse executar os algoritmos disponíveis no *software* sobre os dados. Esse arquivo foi construído com base nos dados de algumas tabelas da estrutura explicada na seção 2.2. Porém, como existem muitas categorias na área da matemática, optou-se por definir um conjunto de categorias, denominadas como as principais, para que pudesse ser feita a classificação das equações apenas nessas categorias. Essas categorias principais foram definidas com base na estrutura da biblioteca *MathWorld,* que apresenta em sua página inicial, *links* para essas categorias, e toda a sua estrutura se encontra relacionada com essas principais. Assim, de acordo com essa estrutura, equações de subcategorias das categorias principais, são classificadas como sendo das categorias principais. As principais são: *Algebra*, *Applied Mathematics, Mathematical Analysis, Discrete Mathematics, Mathematical Logic, Geometry, Number Theory, Probability and Statistics, Recreational Mathematics, Topology*.

Após a definição dessas categorias, foi necessário criar um programa para buscá-las no banco de dados, junto com as páginas e subcategorias relacionadas a elas, esse passo se repete até o final da estrutura, para que se possa extrair dessas páginas as equações existentes nelas. Após esse passo, para se ter o conjunto de operadores e classificá-los, foi necessário fazer uma análise em cada equação, separando quais operadores estão presentes nela e classificá-los de acordo com a categoria à qual a equação pertence. Para exemplificar, a Figura 8 representa um modelo da hierarquia das classes e páginas. As equações que estiverem presentes nas páginas P1, P2 e P3 serão classificadas como sendo da área *Geometry* visto que essas páginas pertencem a sub-áreas de *Geometry*.

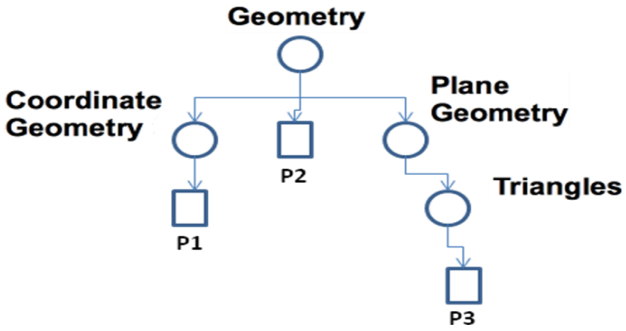


Figura 8: Representação da hierarquia de categorias

O pseudo-algoritmo para geração do arquivo é descrito abaixo:

```
1.  V ← Classes Principais;
2.  ARQ ← criar arquivo operadors.arff;
3.  ARQ.escrever( “@relation operadors” );
4.  Q ← Selecionar ope_operador em tb_operador;
5.  para cada resultado em Q faça
6.         ARQ.escrever( “@attribute ”  + Q.ope_operador +  “ {0, 1}” );
7.  ARQ.escrever( “@attribute class ”  +
8.              “{ V.classesPrincipais() }” );
9.  ARQ.escrever( “@data” );
10. para cada classe principal em V faça
11.         V é marcada como visitada
12.         Q ← Selecionar htm_id em rl_html_category
13.                                    onde cat_id = V.cat_id
14.         para cada resultado em Q faça
15.              se Q.htm_id não existir em P
16.                   P.adicionar (Q.htm_id);
17.         analisa_estrutura(V, P);
18.         para cada P em p faça
19.               analisa_operador (V, p, ARQ);

1.  funcao analisa_estrutura (V, P)
2.      X ← Selecionar cat_idDestino em rl_category_category
3.                                    onde cat_idOirgem = V.catId
4.      para cada resultado em X faça
5.          se X não foi visitada
6.               X é marcada como visitada
7.               D ← Selecionar htm_id em rl_html_category
8.                                    onde cat_id = X.cat_idDestino
9.               para cada resultado em D faça
10.                    se D.htm_id não existir em P
11.                          P.adicionar (D.htm_id);
12.              analisa_estrutura(X, P);
```

```
1.  funcao analisa_operador (x, p, ARQ)
2.       T ← Selecionar equ_id em tb_equation
3.                                onde fk_htm_id =  p.htm_id
4.       para cada resultado em T faça
5.                 OE ← Selecionar ope_id em rl_equation_operador
6.                                onde equ_id = T.equ_id;
7.                 O ← Selecionar ope_id em tb_operador
8.                 para cada resultado em O faça
9.                        se (O existir em OE)
10.                              linha ← linha +  "1," ;
11.                       senão linha ← linha +  "0," ;
12.                linha ← linha + x;
13.                ARQ.escrever (linha);
```

Após a execução desse algoritmo um arquivo contendo aproximadamente 550000 instâncias (equações) foi gerado. Este arquivo também contém os 130 operadores definidos, e mais um atributo de classe representando as 10 categorias principais, totalizando um total de 131 atributos .

Um trecho do arquivo é mostrado na Figura 9.

```
@relation operandos
@attribute + {0,1}
@attribute - {0,1}
@attribute * {0,1}
@attribute / {0,1}
@attribute \and {0,1}
@attribute \or {0,1}
@attribute \neg {0,1}
@attribute \cup {0,1}
.       .       .
.       .       .
.       .       .
@attribute \bigcup {0,1}
@attribute \Downarrow {0,1}
@attribute \updownarrow {0,1}
@attribute Class
{Algebra,Applied_mathematics, ... Recreational_mathematics,Topology}
@data
0,0,0,0,0,0,0,0,0,1,0, ...  0,0,0,0,1,0,0,0,0,0,0,0,0,0,Algebra
0,0,0,1,0,0,0,0,0,0,0, ...  0,0,0,0,0,0,0,0,0,0,0,0,0,0,Algebra
0,1,0,0,0,0,0,0,0,0,0, ...  0,0,0,0,0,0,0,0,0,0,0,0,0,0,Discrete_mathematics
0,0,0,0,0,0,0,0,0,0,0, ...  0,0,0,0,0,0,0,0,0,0,0,0,0,0,Mathematical_logic
0,0,0,1,0,0,0,0,0,0,0, ...  0,0,0,0,0,0,0,0,0,0,0,0,0,0,Discrete_mathematics
0,0,0,0,0,0,0,0,0,0,0, ...  0,0,0,0,0,0,0,0,0,0,0,0,0,0,Algebra
0,1,0,0,0,0,0,0,0,0,0, ...  0,0,0,0,0,0,0,0,0,0,0,0,0,0,Discrete_mathematics
0,0,0,0,0,0,0,0,0,0,0, ...  0,0,0,0,0,0,0,0,0,0,0,0,0,0,Algebra
1,0,0,0,0,0,0,0,0,0,0, ...  0,0,0,0,0,0,0,0,0,0,0,0,0,0,Topology
0,0,0,0,0,0,0,0,0,0,0, ...  0,0,0,0,0,0,0,0,0,0,0,0,0,0,Mathematical_logic
0,0,0,0,0,0,0,0,0,0,0, ...  0,0,0,0,0,0,0,0,0,0,0,0,0,0,Algebra
0,0,0,0,0,0,0,0,0,0,0, ...  0,0,0,0,0,0,0,0,0,0,0,0,0,0,Mathematical_logic
0,0,0,0,0,0,0,0,0,0,0, ...  0,0,0,0,0,0,0,0,0,0,0,0,0,0,Topology
1,0,0,0,0,0,0,0,0,0,0, ...  0,0,0,0,0,0,0,0,0,0,0,0,0,0,Topology
0,0,0,0,0,0,0,0,0,0,0, ...  0,0,0,0,0,0,0,0,0,0,0,0,0,0,Algebra
0,0,0,0,0,0,0,0,0,0,0, ...  0,0,0,0,0,0,0,0,0,0,0,0,0,0,Discrete_mathematics
0,0,0,0,0,0,0,0,0,0,0, ...  0,0,0,0,0,0,0,0,0,0,0,0,0,0,Algebra
.       .       .     .       .       .       .       .
.       .       .       .       .       .       .       .
.       .       .       .       .       .       .       .
```

Figura 9: Trecho do arquivo operadores.arff

Cada instância (após *@Data*) é referente a uma equação e nela estão contidos todos os 130 operadores mais o atributo de classe, onde cada '0' significa que aquele operador não esta presente na equação, e '1' significa que o operador esta presente na equação, podendo até estar contido mais de uma vez.

Outro arquivo também foi gerado e o algoritmo para a geração deste é similar ao descrito acima. As diferenças entre os arquivos são que os atributos ao invés de poderem possuir os valores 0 ou 1, podem assumir "sim", "não" ou ?, e as instâncias das equações ao invés de possuírem 1 informando que o operador está presente possui "sim", e no lugar do 0 possui ?. Dessa forma o *Weka* interpretará que onde tiver ? significa que o atributo é desconhecido e não o levará em conta para criação das regras. Isso evitará a criação explicita de associações negativas, como por exemplo, toda vez que a equação não possuir o operador +, então com 90% de confiança a mesma não terá o operador -, que não é o foco deste trabalho. Foi necessária a utilização dos dois arquivos, pois os gráficos gerados pelo *Weka* para cada um deles possuem características únicas.

3.4 Configuração da ferramenta, algoritmo escolhido

Após criar o arquivo e carregá-lo no *Weka* (como já mostrado na Figura 7) foi necessário definir qual técnica e algoritmo aplicar para se chegar aos resultados esperados. Em grande parte dos problemas não existe uma técnica certa ou errada para ser aplicada, porém existem as técnicas que fornecem resultados melhores a certos tipos de aplicações. Como o problema deste trabalho é buscar por padrões/características nas equações, dado os seus operadores, para se determinar a qual categoria ela pertence, com uma certa probabilidade, então a técnica

escolhida foi a de Associação, visto que esta fornece regras que representam padrões de relacionamento entre itens de uma base de dados.

Para essa técnica o *Weka* disponibiliza vários algoritmos, como *Apriori*, *FilteredAssociator*, *PredictiveApriori*, *Tertius*, entre outros. O algoritmo que foi escolhido para o trabalho foi o *Apriori*, por ser um dos mais utilizados e por ter encontrado resultados mais próximos do esperado. Os outros algoritmos foram executados algumas vezes para se ter uma amostra de resultados para que pudesse ser feita uma comparação e ver qual algoritmo melhor se adequava ao problema. Alguns algoritmos, como o *FilteredAssociator*, foram descartados devido à existência de uma falha no *software*.

Outra vantagem do *Apriori* são as possibilidades de configurações dos parâmetros, como suporte máximo e mínimo, valor mínimo de confiança, *lift*, *leverange*, quantidade de regras a serem encontradas, entre outras possíveis. Para entender o funcionamento do algoritmo, em (ROMÃO Wesley et al., 1999) se encontra um exemplo de fácil entendimento, descrito a seguir:

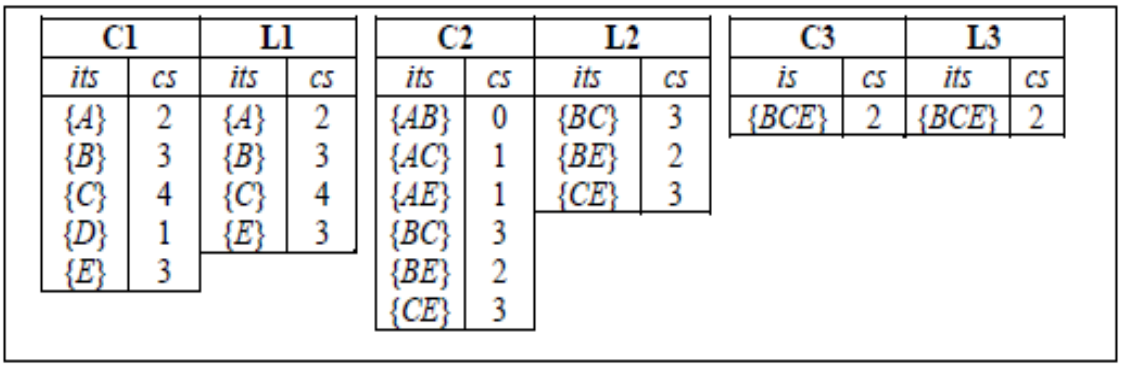
Suponha um banco de dados formado somente por um grupo de pesquisa, GP. Este grupo é composto por cinco pesquisadores, conforme mostrado na Tabela 1.

Tabela 1: Grupo de Pesquisa

Pesquisador	**A**	**B**	**C**	**D**	**E**
1	1	0	0	1	0
2	0	1	1	0	1
3	1	0	1	0	1
4	0	1	1	0	1
5	0	1	1	0	0
Frequência	**2**	**3**	**4**	**1**	**3**

Seja *Ck* o conjunto de *k-itens* candidatos, onde *k* = 5. Cada membro *ck* deste conjunto tem dois campos: *itemset* e contador de suporte, representados, respectivamente, por *its* e *cs* na Figura 10.

Seja *Lk* o conjunto dos *k-itemsets*. De modo análogo, cada membro deste conjunto também possui *its* e *cs*.



C1	
its	cs
{A}	2
{B}	3
{C}	4
{D}	1
{E}	3

L1	
its	cs
{A}	2
{B}	3
{C}	4
{E}	3

C2	
its	cs
{AB}	0
{AC}	1
{AE}	1
{BC}	3
{BE}	2
{CE}	3

L2	
its	cs
{BC}	3
{BE}	2
{CE}	3

C3	
is	cs
{BCE}	2

L3	
its	cs
{BCE}	2

Figura 10: Geração dos candidatos (C) e dos itemsets (L)

O primeiro passo do algoritmo conta a frequência com que os itens ocorrem para determinar os 1-*itemsets* (última linha da Tabela 1). Posteriormente, obtém-se o conjunto de candidatos 1-*itemsets*, *C1*, mostrado na Figura 10. Assumindo um suporte mínimo igual a dois, ou seja, o *cs* (contador de suporte) deve ser maior ou igual a 2. Como o conjunto possui 5 elementos, então o suporte mínimo é calculado através da divisão 2/5, ou seja, *minsup* = 40%. Na Figura 10, *L1* é composto pelos elementos de *C1* com suporte igual ou superior a 40%. No exemplo, somente o *itemset D* não atendeu a esta condição, ficando *L1* composto por {*A*}, {*B*}, {*C*} e {*E*}.

Para descobrir o conjunto dos 2-*itemsets*, de modo a continuar satisfazendo ao suporte mínimo, o *Apriori* usa a concatenação *L1*L1* para gerar o conjunto candidato *C2*, que consiste de 2 - *itemsets*. Por exemplo, {*C*} e {*E*} geram {*CE*}. Mais uma vez, cada ocorrência é computada. No caso {*CE*} ocorre três vezes em GP (registros 2, 3 e 4). *L2* é determinado com base no suporte de cada candidato de *C2*. Agora são excluídos {*AB*}, {*AC*} e {*AE*}, pois têm suporte inferior ao mínimo estabelecido.

A geração de *C3* é obtida a partir de *L2* de uma maneira distinta. Os futuros *itemsets* candidatos devem manter uma ordem lexicográfica[17], tal que quando a concatenação *L2* L2* for realizada deve-se obedecer duas regras:

1 - O primeiro item de um *itemset* deve ser idêntico ao primeiro item do outro *itemset* e assim sucessivamente.

2 - O último item do *itemset* deve ser menor, lexicograficamente, que o último item do outro *itemset.*

Na Figura 10, o *itemset* candidato {*BCE*}, em *C3*, foi formado concatenando {*BC*} com {*BE*}, pois *B* = *B* e *C* < *E*. Este foi o único conjunto que pôde ser formado, pois não há outra concatenação que satisfaça a regra 2. A concatenação {*BC*}*{*CE*}, por exemplo, não satisfaz (2), pois, lexicograficamente, *p1* = *B* é menor que *q1* = *C*.

O passo seguinte é descobrir as regras de associação. No caso do grupo de pesquisa GP, supondo uma confiança mínima de 60% e mantendo o suporte mínimo em 40%, uma regra provável seria *BC* -> *E*. Para ela, a confiança é igual suporte(*BCE*)/suporte(*BC*), cujo resultado é 2/3, ou 66%, satisfazendo a condição imposta (Tabela 2*)*.

Para a regra *BC* -> *E* ou (pesquisador brasileiro; sexo feminino) -> (pesquisador doutor), seu suporte seria o percentual de ocorrências de *BCE* com relação ao total de pesquisadores do grupo, que resulta em 40%. Então, esta é uma regra válida. Isto equivale a dizer que, das pesquisadoras brasileiras, 66% têm doutorado, muito embora estas brasileiras portadoras do título de doutor correspondam a apenas 40% dos indivíduos do grupo. Outra provável regra seria *B* -> *CE*. Para esta situação, o valor da confiança seria idêntico, pois a razão suporte (*BCE*)/suporte (*B*) também é igual a 2/3.

Como foi dito, esse algoritmo permite a configuração dos parâmetros de confiança e suporte, que são os principais, das regras a serem buscadas. O suporte é determinar a frequência que um *itemset* ocorre entre todas as transações da base de dados, sendo a porcentagem de transações onde este *itemset* aparece. A confiança

[17] Também conhecida como ordem do dicionário ou ordem alfabética.

mede a força da regra e determina a sua validade. Para dois atributos X e Y, o Suporte[X,Y] será o número de casos que contém X e Y dividido pelo número total de registros. A Confiança da regra X → Y, será o número de registros que contém X e Y dividido pelo número de registros que contém X. Desse modo, tem-se a probabilidade de que caso aconteça X, com uma confiança de z%, Y irá ocorrer.

Tabela 2: Ocorrências dos conjuntos de atributos

Pesquisador	**B**	**C**	**E**
1	0	0	0
2	1	1	1
3	0	1	1
4	1	1	1
5	1	1	0
Frequência	**3**	**4**	**3**

Para a análise do arquivo de operadores deste trabalho, diversas execuções do algoritmo *Apriori* foram realizadas, cada uma com configurações diferentes. Isso devido ao fato de não se ter como saber qual a configuração ideal para ter os melhores resultados. Pode-se ter uma idéia de qual configuração levará a resultados mais precisos, mas é necessário ir variando e executando novamente para se ter vários resultados e analisar qual o melhor. A Figura 11 mostra o *Weka* e os resultados após a execução do algoritmo *Apriori* para uma determinada configuração. Abaixo está descrita a configuração para o algoritmo *Apriori* que trouxe alguns resultados melhores, que serão descritos no Capítulo 5:

Confiança mínima = valores entre 0.6 e 1
Suporte mínimo = valores entre 0.3 e 0.45
Suporte máximo = valores entre 0.75 e 0.95
Numero de regras = valores entre 50 e 200

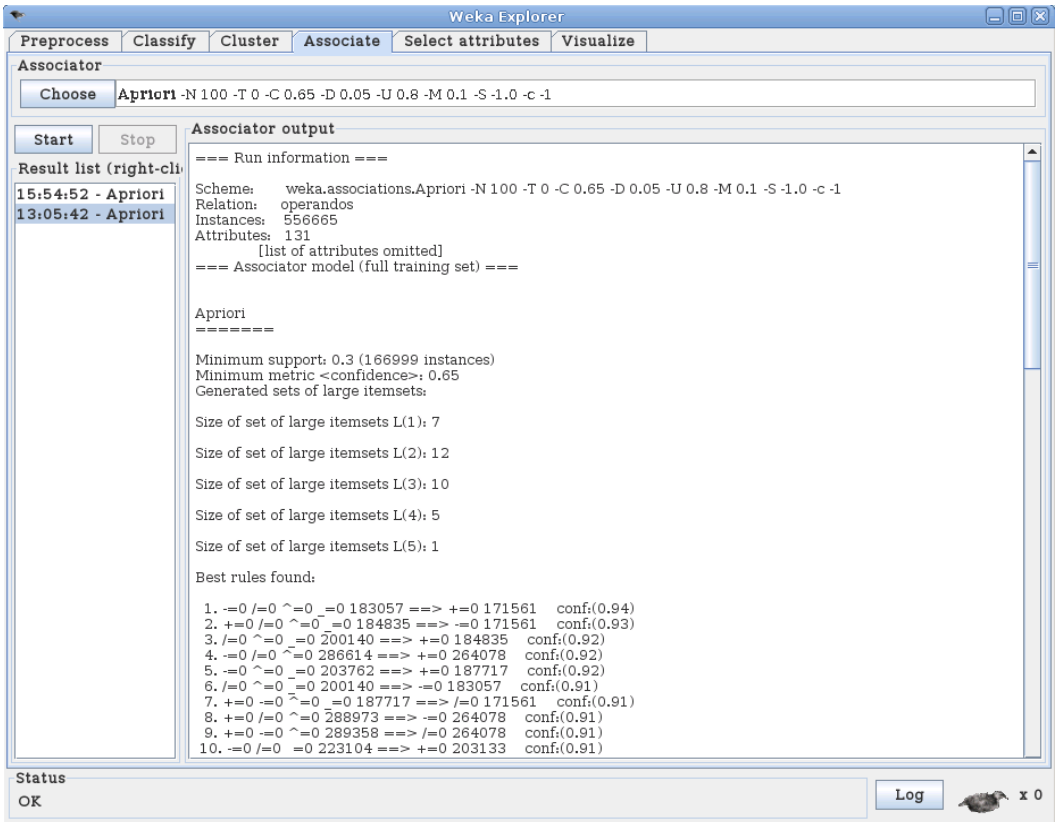


Figura 11: Resultados da execução do algoritmo Apriori

4
Análise probabilística

Este capítulo apresenta uma outra forma de análise do arquivo operador.arff para se tentar chegar aos resultados esperados

4.1 Descrição da técnica

Para reforçar e/ou comparar os resultados obtidos pelo *Weka* optou-se por utilizar outra técnica: através do estudo das probabilidades dos operadores nas categorias. Fazendo uso do primeiro arquivo descrito na Seção 3.3 é possível ter a informação de quantas vezes cada operador aparece em cada categorias. Por exemplo, suponha que o operador + apareça em um número y de páginas da categoria *Algebra*. Ao se analisar o operador + para todas as categorias, pode-se obter então informações de como é a distribuição do mesmo pelas categorias da matemática. Com esses dados é possível então calcular a probabilidade de cada um desses operadores estarem presentes nas categorias definidas. Dessa forma, dada uma equação com um conjunto de operadores, é possível calcular a probabilidade e dizer a qual categoria a equação possui maior probabilidade de pertencer, com base nos seus operadores, e nas probabilidades associadas aos mesmos.

Para calcular a probabilidade da equação pertencer a uma categoria optou-se por tratar os eventos como sendo independentes, apesar de a ocorrência de um operador afetar na probabilidade da ocorrência de outro, tornando os eventos dependentes. A análise considerando a dependência dos eventos (*Bayesiana*) é proposta como trabalhos futuros do presente projeto.

Um evento é considerado independente quando a ocorrência de um evento não afetar na probabilidade de outra ocorrência. Suponha que um casal queira ter

dois filhos, um menino e uma menina. A probabilidade de o casal ter um menino ou uma menina na primeira vez é de 50%, e a probabilidade de nascer uma menina ou um menino, dado que o primeiro filho nasceu continua de 50%, ou seja, o nascimento do primeiro não tem influência no nascimento do segundo, tornando os eventos independentes.

Já um evento dependente é o contrario. A ocorrência de um muda a probabilidade da ocorrência do outro. Suponha uma caixa contendo 10 bolas vermelhas e 10 bolas azuis. Deseja-se tirar duas bolas, uma vermelha e uma azul, sem haver reposição das bolas tiradas. A probabilidade de se tirar uma bola na primeira vez, azul ou vermelha, é de 50%. Porém ao se tirar uma bola, suponha azul, a probabilidade de se tirar uma bola vermelha não é mais de 50% pois agora existem mais bolas vermelhas do que azuis (10 vermelhas e 9 azuis), caracterizando dessa forma o evento como dependente.

Dessa forma, voltando aos operadores, suponha a equação: $x + \arccos(30)$. Dessa equação extrai-se como operadores o + e o *arccos*. O algoritmo proposto realiza então o seguinte cálculo: Considere como evento cada posição da tabela mostrada na Figura 12, por exemplo: P(A') = Probabilidade de uma equação que tem o + ser da *Algebra*, P(A'') = Probabilidade de uma equação que tem o + ser da *Applied Mathematics*, e assim por diante. Com o mesmo raciocínio, P(B') = Probabilidade de uma equação que tem o *arccos* ser da *Algebra,* seguindo o raciocínio para as demais categorias. Esse cenário, com dados reais obtidos nas páginas da *Wikipedia*, é mostrado na Figura 12:

	Algebra	Applied Mathematics	Mathematical analysis	Discrete mathematics	Mathematical Logic	Geometry	Number Theory	Probability and Statistics	Recreational Mathematics	Topology
OP										
+	0,1371	0,1073	0,2395	0,1074	0,031	0,1394	0,0501	0,0647	0,0149	0,1086
\arccos	0,1793	0,0202	0,2146	0,0934	0,0076	0,2677	0,0101	0,051	0	0,202
Total	0,0246	0,0022	0,0514	0,0100	0,0002	0,0373	0,0005	0,0033	0	0,0219

Figura 12: Probabilidades dos operadores + e arccos.

A seguinte pergunta então pode ser feita: Sabendo-se que uma equação que possua o operador + pertence à *Algebra* com probabilidade de 13,71%, e que uma equação que possua o operador *arccos* pertence à *Algebra* com probabilidade 17,93%, qual a probabilidade de que a equação dada anteriormente de fato seja da *Algebra*? Perguntas semelhantes podem ser feitas para todas as categorias em questão evidentemente. A Figura 12 mostra as probabilidades individualmente por operadores, mas quando analisa-se uma equação, onde mais de um operador pode acontecer, a ocorrência de um muda a probabilidade de que outro ocorra, e portanto, muda a probabilidade de classificar a equação como sendo de uma determinada categoria. Assim, fica claro que se em uma equação o operador *arccos* acontece, a probabilidade de que essa equação seja classificada como sendo *Recreational Mathematics* será de 0%. Dessa forma, fica caracterizada a dependência dos eventos em questão.

Segundo (DONAIRE. Denis, 1995) para se calcular um evento dependente pode-se utilizar de probabilidade condicional, onde o cálculo é feito através da probabilidade de ocorrência de um evento A dado que um evento B ocorreu (probabilidade de + dado que *arccos* ocorreu), que pode ser expressa pela fórmula $P(A \mid B) = P(A \cap B) / P(B)$. Porém esta é uma forma mais complicada para ser utilizada neste projeto, pois quando se trata de uma equação com 2 operadores,

é necessário calcular a interseção de A e B, ou seja, verificar quantas são as equações pertencentes a uma categoria x onde ocorrem os dois operadores e dividir este valor pela probabilidade da ocorrência do operador B nessa categoria. Se a equação tiver 3 operadores, é necessário verificar a quantidade de ocorrência dos 3 juntos em uma categoria, sobre a probabilidade dos outros 2 operadores, onde será necessário verificar a quantidade de ocorrência dos 2 juntos em uma categoria, e assim por diante. Conforme a quantidade de operadores vai aumentado, esse cálculo vai se tornando mais complexo.

Dessa forma, nessa análise optou-se por relaxar a restrição, e analisar os eventos como sendo independentes. Pela definição da independência, segundo (TRIVEDI, Kishor Shridharbhai. 2001), dois eventos são ditos independentes se

$$P(A \cap B) = P(A) * P(B)$$

Ou seja, voltando ao exemplo $x + \arccos(30)$, para se analisar a qual categoria a equação apresentada possui mais probabilidade de pertencer, foi feita a multiplicação das linhas onde os operadores da equação aparecem, como se a probabilidade de uma linha (operador) não dependesse da probabilidade da linha anterior (operador anterior). O resultado para a equação do exemplo é o mostrado na linha Total da Figura 12. Onde essa equação estaria contida com maior probabilidade nas áreas *Mathematical Analysis* (probabilidade 0,0514) ou *Geometry* (probabilidade 0,0373). Observe ainda que esse cálculo já exclui qualquer chance dessa equação pertencer à área de *Recreational Mathematics.*

4.2 Algoritmo criado para geração das probabilidades

Para se gerar o arquivo contendo as probabilidades de todos os operadores em relação a todas as categorias, criou-se um programa que tendo como entrada o arquivo *operadores.arff* analisou todas as linhas após *@data*, referente às equações e operadores em cada uma, para se calcular a quantidade de vezes que cada operador aparece em cada categoria, para que no final se tenha os dados para o cálculo da probabilidade. O pseudo-algoritmo é descrito abaixo:

```
1.  ARQProb ← criar arquivo probabilidades.txt;
2.  ARQ ← abrirArquivo ("operadors.arff");
3.  ARQ.posicionarApos( "@data" );
4.  enquanto (naoForFim(ARQ)) faça
5.          linha = ARQ.proximaLinha();
6.          quebraLinha[] = linha.recordar(",");
7.          classe = quebraLinha[quebraLinha.size - 1];
8.          para i = 0, enquanto i < quebraLinha.size - 1, faça
9.                  se quebraLinha[i] != 0
10.                         probabilidades[i][classe]++;
11. para i = 0, enquanto i < 130 faça
12.         para j = 0, enquanto j < 10 faça
13.                 soma = soma + probabilidades[i][j];
14.         para j = 0, enquanto j < 10 faça
15.                 ARQProb.escrever(probabilidades[i][j]*100/soma);
```

Ao final da execução deste algoritmo um arquivo será gerado. A Figura 13 mostra um trecho dele, onde cada linha é referente a um operador e cada coluna a uma categoria. Dessa forma, na interseção entre uma linha com uma coluna obtém-se um valor referente à probabilidade de o operador daquela linha estar na categoria daquela coluna. Utilizando este arquivo e a técnica descrita anteriormente, pode-se calcular a probabilidade de uma equação estar em uma

categoria dado o seu conjunto de operadores. Dessa forma obter-se-á resultados que apontem indícios da existência ou não dos padrões nas equações matemáticas.

```
13.71%  10.73%  23.95%  10.74%  3.10%   13.94%  5.01%   6.47%   1.49%   10.86%
13.72%  12.18%  23.77%  10.38%  2.63%   13.33%  4.44%   7.94%   1.19%   10.42%
17.10%  10.82%  22.96%  9.80%   3.63%   14.44%  3.67%   6.03%   0.98%   10.57%
12.31%  11.84%  24.66%  9.97%   2.37%   14.14%  4.48%   8.17%   1.07%   10.98%
17.53%  7.98%   11.48%  21.88%  20.44%  10.70%  0.35%   1.87%   0.02%   7.74%
17.96%  10.94%  0.74%   33.41%  32.59%  1.55%   0.07%   1.55%   0.00%   1.18%
12.89%  15.14%  15.52%  16.45%  12.47%  9.45%   1.77%   9.09%   0.25%   6.97%
11.40%  11.00%  12.72%  21.21%  15.27%  11.49%  1.36%   4.97%   0.44%   10.12%
7.77%   7.57%   18.26%  22.30%  16.35%  11.30%  1.72%   5.35%   0.40%   8.98%
13.11%  10.23%  16.28%  16.36%  8.96%   13.40%  1.82%   8.58%   0.17%   11.08%
11.49%  9.05%   16.87%  17.85%  10.51%  13.45%  1.71%   7.58%   0.00%   11.49%
11.63%  14.47%  22.22%  11.89%  7.24%   12.92%  0.00%   8.01%   0.00%   11.63%
9.99%   11.37%  23.09%  15.19%  5.07%   9.91%   4.80%   10.00%  1.42%   9.16%
9.33%   14.82%  19.42%  15.46%  6.01%   9.33%   4.09%   10.16%  2.68%   8.69%
14.73%  10.79%  23.68%  10.17%  3.00%   14.30%  4.37%   7.01%   1.09%   10.86%
16.74%  3.13%   26.45%  11.04%  0.87%   21.72%  1.53%   1.10%   0.24%   17.18%
16.89%  3.67%   26.09%  10.90%  0.79%   21.58%  1.24%   1.40%   0.23%   17.22%
```

Figura 13: Trecho do arquivo probabilidades.txt gerado

Na Figura 14 é apresentado um trecho do arquivo tratado, com os operadores já devidamente classificados em probabilidade por categoria.

OP	Algebra	Applied Mathematics	Mathematical analysis	Discrete mathematics	Mathematical Logic	Geometry	Number Theory	Probability and Statistics	Recreational Mathematics	Topology
+	13.71%	10.73%	23.95%	10.74%	3.10%	13.94%	5.01%	6.47%	1.49%	10.86%
-	13.72%	12.18%	23.77%	10.38%	2.63%	13.33%	4.44%	7.94%	1.19%	10.42%
*	17.10%	10.82%	22.96%	9.80%	3.63%	14.44%	3.67%	6.03%	0.98%	10.57%
/	12.31%	11.84%	24.66%	9.97%	2.37%	14.14%	4.48%	8.17%	1.07%	10.98%
\and	17.53%	7.98%	11.48%	21.88%	20.44%	10.70%	0.35%	1.87%	0.02%	7.74%
\or	17.96%	10.94%	0.74%	33.41%	32.59%	1.55%	0.07%	1.55%	0.00%	1.18%
\neg	12.89%	15.14%	15.52%	16.45%	12.47%	9.45%	1.77%	9.09%	0.25%	6.97%
\cup	11.40%	11.00%	12.72%	21.21%	15.27%	11.49%	1.36%	4.97%	0.44%	10.12%
\bigcup	7.77%	7.57%	18.26%	22.30%	16.35%	11.30%	1.72%	5.35%	0.40%	8.98%

Figura 14: Trecho do arquivo probabilidades.xls trabalhado em cima dos dados de probabilidade.txt

4.3 Algoritmo criado para classificação das equações

Utilizando do arquivo *probabilidades.txt*, descrito anteriormente, foi criado um programa em que dada uma equação, presente em uma das páginas da *Wikipedia*, analisasse os operadores presentes nesta equação e calculasse probabilisticamente a qual categoria esta equação pertenceria (usando a técnica explicada neste capítulo), comparando o resultado encontrado com a categoria classificada pela *Wikipedia*. O algoritmo é descrito abaixo:

```
1. probabilidades = abrir_arquivo("probabilidades.txt");
2. equation = buscar_equacao_do_bd_aleatoria();
3. operadores = analisar_operadores(equation);
4. provavel_area = calcular_probabilidade_area(operadores, probabilidades);
5. area = verificar_area_no_bd(equation);
6. se area == provavel_area
7.          Resposta Certa
8. senão Resposta Errada
```

Este programa foi executado para analisar todas as equações presentes na base de dados.

Outro programa, com propósito semelhante ao descrito acima foi criado, porém este fazia uma análise mais detalhada. Este novo programa separava todas as equações com 1 operadores e contava quantas delas estavam presentes em 1, 2, 3, etc. categorias, e marcava o índice de acerto para cada uma. O mesmo processo foi feito para equações com 2, 3, 4, etc. operadores. Dessa forma, ao final da execução deste programa, gerou-se um arquivo contendo o índice de acerto para as equações de forma mais detalha para que se pudesse efetuar uma análise mais critica.

5
Resultados e Conclusões

Este capítulo apresenta os resultados e conclusões obtidos ao final da execução de projeto. Primeiro, os resultados são descrito de uma forma geral. Na seção 5.1 os resultados do Weka são apresentados mais detalhados e na seção 5.2 os resultados da análise probabilística. Na seção 5.3 é apresentada uma conclusão geral para os resultados obtidos.

Com o término deste projeto um ponto pode ser destacado visto que o trabalho apresentado poderá ser utilizado por outros pesquisadores em seus projetos. Esse ponto são os algoritmos criados para recuperação de um conteúdo específico da *Wikipedia*. Esses algoritmos, se seguidos, podem ser utilizados por pesquisadores de diversas áreas para recuperar o conteúdo que lhe é de interesse da *Wikipedia*, fazendo apenas algumas modificações, visto que alguns desses algoritmos são específicos para o conteúdo matemático. Porém a estrutura da *Wikipedia* relacionada à localização de páginas de categorias, links, relação das categorias etc. permanece a mesma para todas as áreas.

Em se tratando dos resultados encontrados pelo *Weka,* não pôde-se chegar a nenhuma conclusão dos padrões esperados, visto que os resultados dessa ferramenta foram inconclusivos, talvez devido ao fato de ser necessário se aplicar um estudo mais detalhado na base de dados, modificando/retirando alguns atributos. Algumas modificações foram feitas no decorrer da pesquisa, tornando os resultados mais interessantes, porém ainda longe do esperado.

Já os resultados obtidos pela análise probabilística foram mais interessantes mostrando um índice satisfatório de acertos para as equações, principalmente quando a quantidade de operadores nas equações aumenta. Através da análise mais detalhada destes resultados, pode-se obter resultados que demonstraram que

é possível se ter uma otimização nas buscas por equações matemáticas, se for efetuada uma classificação previa da equação a ser buscada de acordo com seus operadores, fazendo com que não seja necessário efetuar comparação em toda a base de dados.

5.1 Resultados obtidos pelo *Weka*

Após diversas execuções do algoritmo *Apriori*, utilizando os dois arquivos gerados, alguns resultados foram obtidos. Porém, as regras de associação encontradas pela ferramenta *Weka* não se aproximaram das regras esperadas. As Figuras 15 e 16 apresentam algumas dessas regras.

```
Best rules found:

 1. \in=0 \le=0 modulo=0 432628 ==> \sum=0 418643    conf:(0.97)
 2. \in=0 \le=0 449268 ==> \sum=0 433805    conf:(0.97)
 3. \in=0 \le=0 \sum=0 433805 ==> modulo=0 418643    conf:(0.97)
 4. \in=0 \le=0 449268 ==> modulo=0 432628    conf:(0.96)
 5. /=0 435539 ==> \sum=0 419026    conf:(0.96)
 6. \in=0 modulo=0 472893 ==> \sum=0 454400    conf:(0.96)
 7. *=0 \le=0 \sum=0 435801 ==> modulo=0 418637    conf:(0.96)
 8. \le=0 \sum=0 473814 ==> modulo=0 454555    conf:(0.96)
 9. *=0 \in=0 454309 ==> \sum=0 435792    conf:(0.96)
10. \in=0 496368 ==> \sum=0 475544    conf:(0.96)
```

Figura 15: Resultados do arquivo 1 obtidos pelo Weka

```
Best rules found:

 1. -=sim 140164 ==> ^=sim 86780    conf:(0.62)
 2. +=sim 111509 ==> ^=sim 63975    conf:(0.57)
 3. /=sim 121126 ==> ^=sim 69134    conf:(0.57)
 4. -=sim 140164 ==> _=sim 79942    conf:(0.57)
 5. +=sim 111509 ==> _=sim 61640    conf:(0.55)
 6. ^=sim 183387 ==> _=sim 99195    conf:(0.54)
 7. /=sim 121126 ==> _=sim 65022    conf:(0.54)
 8. /=sim 121126 ==> -=sim 63420    conf:(0.52)
 9. ^=sim 183387 ==> -=sim 86780    conf:(0.47)
10. -=sim 140164 ==> /=sim 63420    conf:(0.45)
```

Figura 16: Resultados do arquivo 2 obtidos pelo Weka

Ao analisar os resultados da Figura 15, referentes ao primeiro arquivo descrito na Seção 3.3, onde os valores dos atributos são classificados como 0 ou 1, pode-se verificar que o *Weka* encontrou apenas regras de associação para a falta de operadores. Como o arquivo possui muitos atributos e muitas instâncias, sendo que a maioria dos atributos possuem valor 0, ou seja, são muito raras as equações que possuem todos ou a maioria dos operadores, a maior equação possui 19 operadores diferentes, então o *Weka* encontra várias regras no estilo da regra 12, que pode ser interpretada como: Quando não existe o operador *\le* (menor igual) na equação, então com 96% de confiança não se terá o operador módulo nesta mesma equação. Regras deste tipo não são interessantes para este trabalho, pois não é possível chegar a nenhuma conclusão sobre a qual categoria a equação pertence, visto que essas regras são apenas referentes a operadores que não estão presentes na equação. Desse modo, todas as execuções, variando a confiança e o suporte, não trouxeram resultados relevantes para esta monografia.

Já na Figura 16, percebe-se que as regras encontradas pelo *Weka* são associadas à presença de operadores nas equações. Essas regras foram encontradas ao analisar o segundo arquivo descrito na Seção 3.3. Como já dito, o formato desse arquivo evita que o *Weka* descubra regras de negação, forçando-o a procurar por novas regras. Dessa forma, pode-se interpretar a regra 1 da Figura 16 como sendo: Toda vez que o operador - é encontrado na equação então com 62% de confiança será encontrado também nesta mesma equação o operador ^. A equação $x-y\hat{}2$, dentre outras, exemplifica esta regra. É importante frisar que essas regras demonstram alguns padrões que não são fáceis de encontrar sem a utilização de alguma técnica de mineração. Porém, apesar de serem mais interessantes, ainda não é possível se chegar à resposta esperada.

Após o termino das análises desses e de outros resultados gerados pelo *Weka*, não foi possível se chegar a nenhuma conclusão se existe um padrão que demonstre com um valor relativamente alto de probabilidade se uma certa equação

com um conjunto de operadores pertence a uma categoria x. Porém através dos gráficos gerados pelo *Weka,* as Figuras 17, 18, 19 e 20 apresentam 4 dos 130 gráficos gerados, percebe-se que de forma geral, os operadores estão mais presentes em uma certa categoria do que em outra. Nesses gráficos, cada cor representa uma das 10 categorias definidas e explicadas na seção 3.3. Algumas categorias possuem a mesma cor, isso é provavelmente devido a algum problema que a ferramenta possui, porém não afetam na análise dos resultados. Analisando por exemplo o operador *arccsc* pode-se verificar que ele está presente em 4 categorias (de baixo pra cima: Álgebra, Análise Matemática, Matemática Discreta e Geometria), e que a categoria de cor verde limão é a mais predominante. Isso mostra que dentre todas as equações que possuem *arccsc* no banco de dados a maioria delas pertence à Geometria.

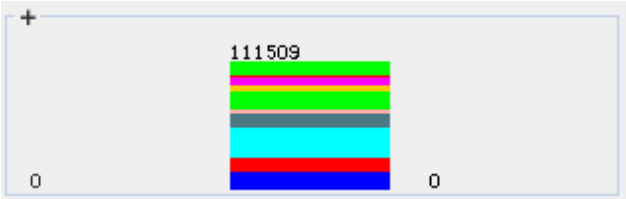


Figura 17: Gráfico para o operador +

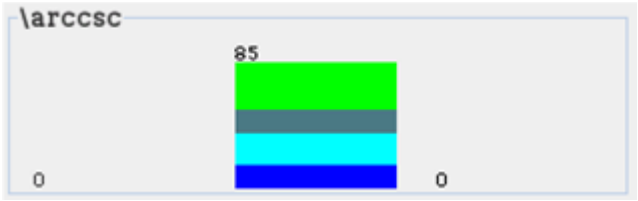


Figura 18: Gráfico para o operador \arccsc

Figura 19: Gráfico para o operador \Epsilon

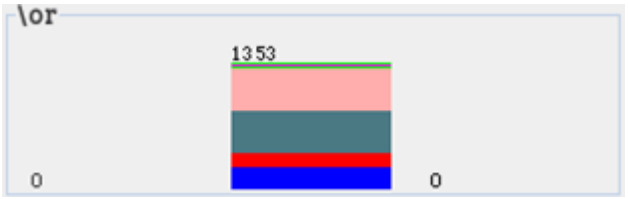


Figura 20: Gráfico para o operador \or

5.2 Resultados obtidos pela análise probabilística

Utilizando do primeiro programa descrito na seção 4.3, onde foram buscadas todas as equações do *BD* e classificadas de acordo com seus operadores obteve-se o resultado de aproximadamente 44% de acertos. O baixo índice de acerto nesta análise se deu por conta de existirem muitas equações que não possuem nenhum operador e equações que não pertencem a nenhuma das 10 categorias analisadas. Das, aproximadamente, 220.000 equações presentes no *BD*, 36.662 não possuem operadores (são equações de apenas uma variável, por exemplo x) e 18.858 não possuem categorias, ou seja, estão classificadas em categorias que não são uma das 10 definidas e nem são sub-categorias delas.

Já o segundo programa descrito na seção 4.3, que efetuava uma análise mais detalhada, trouxe resultados mais satisfatórios. Ao final da execução do mesmo chegou-se à conclusão que conforme a quantidade de operadores aumenta a probabilidade de acerto também aumenta. Ao analisar as equações com 1 operador pertencentes a apenas 1 categoria o índice de acerto é 37,68%, enquanto que as com 6 operadores e pertencentes a 1 categoria esse índice chega a 61,04%. A Tabela 3 mostra todos os resultados de acertos calculados onde não foram consideradas equações que não pertencem a uma das 10 categorias definidas, e não foram mostrados resultados de equações pertencentes a todas as 10 categorias, pois não existem tais equações. Na Tabela 3 as células que se encontram com um traço,

significa que não existem equações com aquela quantidade de operadores pertencentes a aquela quantidade de categorias, sendo assim, desprezada para o cálculo da média.

Tabela 3: Porcentagem de acerto do algoritmo

Operadores	Categorias									
	1	2	3	4	5	6	7	8	9	Média
1	37,68	38,47	73,06	91,06	94,63	96,7	100	100	100	81,29
2	39,57	39,64	78,12	93,2	95,04	98,59	100	100	100	82,68
3	41,91	42,35	80,7	94,44	96,17	99,39	100	100	100	83,88
4	46,2	43,97	83,41	95,82	96,81	99,51	100	100	100	85,08
5	55,95	47,01	86,75	96,67	97,72	99,08	100	100	100	87,02
6	61,04	50,5	89,76	97,57	97,74	98,36	100	100	100	88,33
7	62,66	55,18	92,51	97,69	98,68	100	100	100	100	89,64
8	64,96	59,41	91,04	97,1	99,26	99,69	100	100	-	88,93
9	67,55	65,33	91,53	97,25	99,65	99,06	100	100	-	90,05
10	72,01	62,6	93,06	98,18	100	100	100	100	100	91,76
11	71,56	58,5	93,81	99,4	100	100	100	-	-	89,04
12	74,39	58,62	94,55	99,15	100	100	100	-	-	89,53
13	78,52	51,36	93,75	100	100	100	100	-	-	89,09
14	67,5	56,57	96,73	95,58	100	100	100	100	-	89,55
15	45,45	57,14	95,58	100	100	100	100	-	-	85,45
16	92,5	12,5	100	100	100	100	-	100	-	86,43
17	100	-	100	100	100	-	-	-	-	100
18	75	-	-	100	100	-	-	-	-	91,67
19	-	-	-	-	100	-	-	-	-	100

Os gráficos das Figuras 21 e 22 demonstram os resultados para esta análise de forma mais detalhada, mostrando o índice de acerto, separando as equações por quantidade de operadores e de categorias à qual ela pertence. Esses resultados encontrados foram divididos em dois gráficos apenas para melhor visualização e análise.

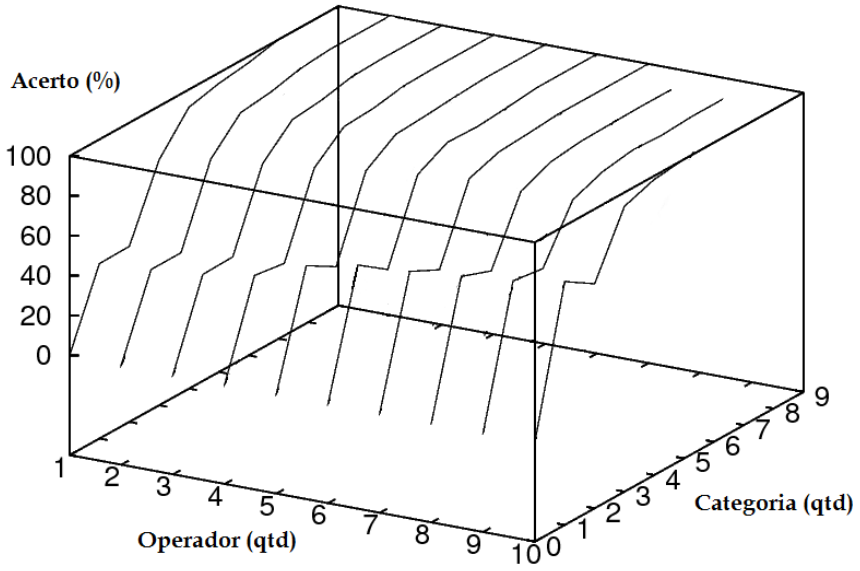


Figura 21: Gráfico 3D dos resultados para equações com até 10 operadores

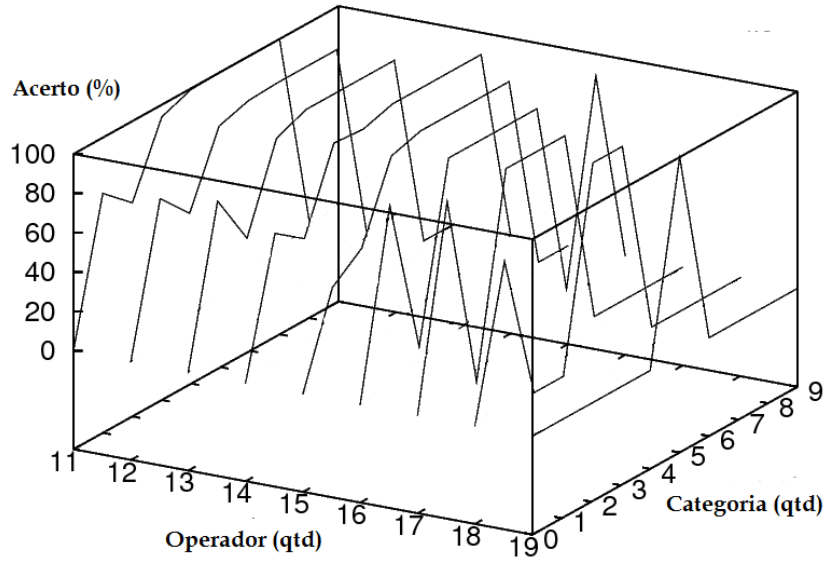


Figura 22: Gráfico 3D dos resultados para equações com 11 à 19 operadores

Pode-se notar nos dois gráficos que ocorre uma queda no índice de acerto em equações que pertencem a 1 e 2 categorias, voltando a subir com 3 categorias, o que parece estranho visto que conforme se aumenta a quantidade de categorias a probabilidade de acerto deveria aumentar. Porém isso ocorre devido à categoria de *Probability and Statistics* estar contida na categoria de *Applied Mathematics*. Assim,

equações que pertençam a apenas uma categoria, englobam todas as possibilidades, mas exclui equações de *Probability and Statistics* uma vez que uma equação contida nessa categoria, automaticamente faria parte da *Applied Mathematics*. Ao se analisar equações em 2 categorias, o que se tem é uma presença, em sua maioria, de equações então dessas duas categorias, o que resulta em um grupo menor do que o obtido apenas para equações de uma categoria, explicando, portanto uma redução no índice de acertos.

5.3 Conclusões

Através da análise de todos os resultados obtidos pela técnica probabilística e os gráficos gerados pelo *WEKA*, pode-se concluir que existe um padrão nos operadores que caracterizam uma equação tornando possível classificá-la previamente de forma satisfatória, dependendo do tamanho da mesma, em certa categoria.

Esses resultados demonstraram a possibilidade de se otimizar uma busca por equações matemáticas de uma futura ferramenta de busca que efetue sua pesquisa em cima de uma base de dados, onde essa busca pode ser feita por semelhança entre as árvores das equações.

Outra conclusão que pôde ser tirada foi em cima da estrutura proposta por este projeto. Essa estrutura se demonstrou eficaz para o seu propósito, onde foi possível rodar os algoritmos descritos por esta pesquisa de forma otimizada em comparação caso tivessem que terem sido rodados em cima da estrutura completa da *Wikipedia.* Utilizando das relações dessa estrutura é possível montar mais facilmente também grafos que demonstrem o nível de relacionamento entre as áreas, utilizando dos links existentes nas páginas. Esses grafos podem trazer informações e indícios para futuros trabalhos em diversas áreas.

6
Trabalhos Futuros

Este capítulo apresenta alguns trabalhos futuros relacionados com o trabalho proposto nesta monografia

Como já dito na introdução desta pesquisa, existem muitas bibliotecas eletrônicas disponíveis para acesso de todos. Dentre elas, duas que merecem destaque na área matemática são: *DLMF* e *MathWorld*. Essas duas bibliotecas são muito conhecidas pelos matemáticos e possuem um bom e crescente acervo referente à matemática.

Um possível trabalho futuro é executar os passos desta pesquisa nessas duas bibliotecas para poder comparar os resultados obtidos nas 3 bibliotecas e confirmar ou anular os padrões encontrados por este trabalho.

Outro trabalho que pode ser executado é considerar a dependência entre os operadores, e realizar uma análise probabilística *Bayesiana,* obtendo assim resultados mais precisos.

7
Referências Bibliográficas

AMO, Sandra. Técnicas de Mineração de Dados. UFU (Universidade Federa de Uberlândia). Disponível em http://www.deamo.prof.ufu.br/arquivos/JAI-cap5.pdf - acesso em 28/05/2011

Autor desconhecido. Overview Data Mining. Curso de Inteligência Tecnológica – IME, 2005

BORTOLI. Joel, Data Mining, Mineração de Dados. Disponível em http://www.guiafar.com.br/portal/index.php?option=com_content&view=article&id=159:data-mining-mineracao-de-dados&catid=43:tecnologia-da-informacao&Itemid=169&lang=pt&limitstart=2 - acesso em 26/05/2011

DONAIRE, Denis – Principios de Estatiística, Atlas – Ano: 1995, Edição: 4

G1.Globo. Artista transforma artigos da Wikipedia em livro de 5 mil páginas. Disponível em http://g1.globo.com/tecnologia/noticia/2011/03/artista-transforma-artigos-da-wikipedia-em-livro-de-5-mil-paginas.html - acesso em 06/05/2011

KNUTH, Donald Ervin – The TeXBook, Stanford University. Ano: 1984

MACHADO, Cargos. O abc da Mineração de Dados. Disponível em http://info.abril.com.br/edicoes/154/arquivos/3281_1.shl - acesso em 26/05/2011

ROMÃO. Wesley, NIEDERAUER. Carlos A.P., et al., EXTRAÇÃO DE REGRAS DE ASSOCIAÇÃO EM C&T: O ALGORITMO APRIORI, Ano: 1999

SUCHANEK, Fabina M., KASNECI, Gjergji - YAGO: A LargeOntology fromWikipedia andWordNet – Ano: 2008

TRIVEDI, Kishor Shridharbhai. – Probability and Statistics with Reliability, Queueing, and Computer Science Applications. Ano: 2001, Segunda Edição

WITTEN, Ian. & FRANK, Eibe. Data Mining – Practical Machine Learning Tools and Techniques. Second Edition, 2005

YANG. Rui et. al., Similarity evaluation on tree-structured data. National University of Singapore. Publicado em SIGMOD '05 Proceedings of the 2005 ACM SIGMOD international conference on Management of data

YOUSSEF Abdou - Equivalence Detection Using Parse Tree – 2007