UERJ

Dissertação de Mestrado em Engenharia de Computação
Área de Concentração em Geomática

OTIMIZAÇÃO COMPUTACIONAL E ESTUDO COMPARATIVO DAS TÉCNICAS DE EXTRAÇÃO DE CONHECIMENTO DE GRANDES REPOSITÓRIOS DE DADOS

Autor: Fernando Luiz Coelho Senra
Orientadora: Profª Drª Maria Luiza Fernandes Velloso

Programa de Pós Graduação em Engenharia de Computação
Área de Concentração em Geomática

Setembro - 2009

Faculdade de Engenharia

OTIMIZAÇÃO COMPUTACIONAL E ESTUDO COMPARATIVO DAS TÉCNICAS DE EXTRAÇÃO DE CONHECIMENTO DE GRANDES REPOSITÓRIOS DE DADOS

Fernando Luiz Coelho Senra

Dissertação submetida ao corpo docente da Faculdade de Engenharia da Universidade do Estado do Rio de Janeiro – UERJ, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia de Computação – Área de Concentração Geomática.

Orientadora: Prof[a] Dr[a] Maria Luiza Fernandes Velloso

Programa de Pós Graduação em Engenharia de Computação
Área de Concentração em Geomática

RIO DE JANEIRO, RJ - BRASIL.

SENRA, FERNANDO LUIZ COELHO

Estudo Comparativo das Técnicas de Extração de conhecimento de grandes repositórios de dados [Rio de Janeiro]
2009
X, 91 p. 29,7 cm (FEN/UERJ, M.Sc., Engenharia da Computação – Área de Concentração Geomática, 2009)
Tese - Universidade do Estado do Rio de Janeiro – UERJ

1. Apriori *Fuzzy* 2.Regras de Associação 3. Lógica Nebulosa

I. FEN/UERJ II. Título (série)

UNIVERSIDADE DO ESTADO DO RIO DE JANEIRO
CENTRO DE TECNOLOGIA E CIÊNCIAS
FACULDADE DE ENGENHARIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO
ÁREA DE CONCENTRAÇÃO GEOMÁTICA

ATA DE DEFESA DE DISSERTAÇÃO MESTRADO

ATA DA 84ª DEFESA DE DISSERTAÇÃO PARA CONCESSÃO DO GRAU DE MESTRE EM ENGENHARIA DE COMPUTAÇÃO, ÁREA DE CONCENTRAÇÃO GEOMÁTICA

CANDIDATO: **Fernando Luís Coelho Senra**

TÍTULO DA DISSERTAÇÃO: "**ESTUDO COMPARATIVO DAS TÉCNICAS DE EXTRAÇÃO DE CONHECIMENTO DE GRANDES REPOSITÓRIOS DE DADOS**".

BANCA EXAMINADORA

NOME COMPLETO (Letra de forma)	*CPF*
Maria Luiza Fernandes Velloso, D.Sc., UERJ (Orientadora)	255.153.657-04
Orlando Bernardo Filho, D.Sc., UERJ	994.490.727-87
Karla Tereza Figueiredo Leite, Ph.D., PUC-RJ	902.620.777-87

LOCAL: Sala 5.032-D

DATA: 16 de setembro de 2009 **HORA DE INÍCIO**: 14h00min

Em sessão pública, após exposição de cerca de 40 *minutos, o candidato foi argüido oralmente pelos membros da banca tendo como resultado:*

(X) APROVAÇÃO POR UNANIMIDADE;
( ) APROVAÇÃO CONDICIONADA À SATISFAÇÃO DAS EXIGÊNCIAS CONSTANTES NA FOLHA DE MODIFICAÇÕES NO PRAZO FIXADO PELA BANCA NÃO SUPERIOR A 60 (SESSENTA) DIAS;
( ) REPROVAÇÃO

Na forma regulamentar foi lavrada a presente ata que é abaixo assinada pelos membros da banca na ordem acima estabelecida e pelo candidato.

Rio de Janeiro, 16 de setembro de 2009.

ORIENTADORA: Maria Luiza Velloso

Orlando Bernardo Filho

CANDIDATA:

DEDICATÓRIA

À minha esposa Odete, pelo amor incondicional, aos meus filhos Daniel e Lara Fernanda, razão principal de todos meus esforços e pelo amor que sinto, cuja intensidade jamais desconfiava possível até eles surgirem em minha vida, e aos meus pais, fonte inesgotável do amor que hoje repasso aos meus filhos e esposa.

RESUMO

Estudo Comparativo das Técnicas de Extração de conhecimento de grandes repositórios de dados.

Ao se realizar estudo em qualquer área do conhecimento, quanto mais dados se dispuser, maior a dificuldade de se extrair conhecimento útil deste banco de dados. A finalidade deste trabalho é apresentar algumas ferramentas ditas inteligentes, de extração de conhecimento destes grandes repositórios de dados.

Apesar de ter várias conotações, neste trabalho, irá se entender extração de conhecimento dos repositórios de dados a ocorrência combinada de alguns dados com freqüência e confiabilidade que se consideram interessantes, ou seja, na medida e que determinado dado ou conjunto de dados aparece no repositório de dados, em freqüência considerada razoável, outro dado ou conjunto de dados irá aparecer.

Executada sobre repositórios de dados referentes a informações georreferenciadas dos alunos da UERJ (Universidade do Estado do Rio de Janeiro), irá se analisar os resultados de duas ferramentas de extração de dados, bem como apresentar possibilidades de otimização computacional destas ferramentas.

ABSTRACT

Comparative Study of Techniques for Extracting knowledge from large data repositories.

When conducting the study in any field of knowledge, the more data is available, the greater the difficulty in extracting useful knowledge from this database. The purpose of this paper is to present some tools called intelligent, knowledge extraction of these large data repositories.

Although many connotations, this work will understand knowledge extraction from data repositories on the combined occurrence of some data with frequency and reliability that are considered interesting, ie, the extent and specific data or data set appears in the data, at a rate deemed reasonable, other data or data set will appear.

Runs on repositories of data on georeferenced data of students UERJ (Universidade do Estado do Rio de Janeiro), will analyze the results of two tools to extract data and present opportunities for optimization of these computational tools

SUMÁRIO

CAPÍTULO I

1.1. INTRODUÇÃO

Desde os primórdios da evolução humana, o conhecimento tem-se revelado de inestimável valor nas mais diversas situações. Determinada informação no momento apropriado, pode se mostrar decisiva.

Nos julgamentos religiosos realizados na Idade Média, por exemplo, obter o depoimento do acusado era essencial para a autoridade julgadora decidir a respeito de sua sentença condenatória. De igual modo, em diversas situações cotidianas, o conhecimento, de qualquer natureza, pode se revelar de essencial importância.

Todavia, apesar de em nenhum momento da história ter se negado a importância fundamental da informação, até algumas dezenas de anos atrás um grave limitador existente era a impossibilidade de armazená-la em grandes quantidades em um único lugar, o que, consequentemente, dificultava seu acesso e a sua propagação. A melhor opção à época era consultar grandes bibliotecas, onde havia uma maior quantidade de informação disponível, limitada somente pelo espaço físico.

Com o advento da microinformática (com a disseminação dos computadores pessoais) e, principalmente, da Rede Mundial de Computadores (Internet), o conhecimento ficou extremamente mais acessível e disponível a todos. Neste contexto, todas as instituições de médio e grande porte, e muitas de pequeno porte, passaram a armazenar em seus computadores todas as informações que pudessem vir a interessar a sua atividade.

Neste cenário atual, a realidade vivida pela maioria das instituições é de ter a sua disposição, em grandiosos bancos de dados, toda a informação que lhe puder ser útil. No entanto, a grande quantidade de informações hoje armazenadas – por ser uma situação inédita em relação à vivenciada em momentos anteriores – por vezes não é utilizada da maneira mais adequada, o que dificulta a tomada de decisões.

Esta situação, aparentemente sem sentido, surge da dificuldade de extrair conhecimentos, potencialmente ao seu dispor, de um conjunto gigantesco de dados. A fim de ilustrar esta situação, vislumbremos o seguinte exemplo hipotético:

1. Determinada rede de vendas no varejo tem em seus registros que foram vendidas 10 milhões de bolsas no ano de 2005;
2. Das 10 milhões de bolsas vendidas, 350 mil foram vendidas para mulheres em Duque de Caxias-RJ;
3. Das 350 mil bolsas vendidas para mulheres em Duque de Caxias, 70 mil bolsas foram vendidas para advogadas entre 25 e 30 anos;
4. Das 70 mil bolsas vendidas para advogadas de Duque de Caxias, 65 mil bolsas foram de uma única marca de preço extremamente elevado.

Concluindo, de posse desta informação (venda concentrada de determinada marca de bolsa para grande maioria de suas clientes advogadas entre 25 e 30 anos), muito provavelmente a rede de vendas irá atuar sobre esse público alvo. Com esse exemplo fica clara a necessidade de se responder duas questões de grande relevância:

1. Como organizar os dados?
2. Como extrair conhecimento dos dados organizados?

A primeira questão pode ser equacionada através da construção de um *DataWarehouse*, tecnologia que permite armazenar as informações, anteriormente dispersas, através da identificação, compreensão, integração e agregação dos dados, de forma a posicioná-los nos locais mais apropriados visando a atender à estratégia organizacional das instituições.

Em resposta à segunda questão, para extrair conhecimento de um sistema de *DataWarehouse,* são necessárias ferramentas de exploração, hoje conhecidas como *Data Mining* (Mineração de Dados).

O *Data Mining* reúne uma série de técnicas, com destaque para as estatísticas, probabilísticas e de inteligência computacional, capazes de fornecer respostas a várias questões ou mesmo descobrir novas informações em grandes bancos de dados. O *Data Mining* é especialmente útil nos casos em que não se conhece a pergunta, mas, mesmo assim, existe a necessidade de respostas.

Esta dissertação tem como objetivo apresentar um estudo sobre mineração de dados e, mais especificamente, como será abordado mais adiante, um estudo sobre mineração de regras de associação, dividida da seguinte forma:

- No capítulo 1 será apresentado breve histórico da importância da informação ao longo do tempo, enfatizando-se a dificuldade de se obter informações a medida em que se aumenta a quantidade de dados disponíveis.

Nesse histórico, ver-se-á de modo ilustrado como existe relação inversa entre a quantidade de dados a ser explorado em determinado repositório e a facilidade de se extrair conhecimento útil desse mesmo repositório.

Serão apresentadas, conceitualmente, as ferramentas Data Mining e *DataWarehouse* como ferramentas úteis para organização e exploração dos grandes repositórios de dados, por fim, breve síntese dos capítulos vindouros.

- No capítulo 2 será apresentado estudo conceitual da mineração de dados, ou seja, como continuação daquilo que foi apresentado no capítulo 1, começará a se estudar nas possíveis soluções para extração de conhecimento dos grandes repositórios de dados.

Nesse mesmo capítulo, irá se apresentar alguns conceitos do que seja conhecimento inerente a um repositório de dados, realizando-se estudo do que sejam regras de associação (quando um item do banco de dados ocorre em determinada freqüência associado a outro item), padrão seqüencial (quando itens do banco de dados ocorrem associados entre si com regulares intervalos de tempo) e árvores de classificação (quando os itens do banco de dados são classificados das mais diversas formas entre si).

Serão, por fim, abordados os conceitos de Data Mining e *Executive Information System*;

- No capítulo 3 será visto um dos pontos chaves do presente trabalho, e os possíveis resultados da extração de conhecimento de grandes quantidades de dados organizados em *DataWarehouse*, far-se-á, nesse capítulo, estudo a respeito de extração de regras de associação envolvendo dados de um *DataWarehouse*;

- No capítulo 4 serão estudadas alguns conceitos fundamentais utilizados no Algoritmo *Apriori Fuzzy* (ferramenta imprescindível na principal ferramenta de extração de regras de associação de um *DataWarehouse*), que são os conceitos de:

1. Lógica Fuzzy: como tratar proposições lógicas / assertivas que não são integralmente analisadas na dualidade imposta pela lógica booleana (ex.: um copo de água a 70% de sua capacidade não pode ser corretamente tratado na dualidade copo cheio / vazio);
2. Conjuntos Fuzzy: como relacionar variáveis (campos) reais a variáveis lingüísticas associadas a esses campos (ex.: como relacionar o conjunto de CR's em uma base de dados aos conceitos CR alto / médio e baixo)

- No capítulo 5 será apresentado estudo conceitual da principal ferramenta existente de extração de regras de associação de grandes repositórios de dados, o Algoritmo "Apriori", e suas principais variações ("*Apriori Group*" e "*Apriori Fuzzy*"), apresentando-o nas seguintes etapas:

1. origens e propostas de sua implementação;
2. as etapas em que é realizado (apriori_gen, onde são realizadas as contagens de seus itemsets e genrules, onde, a partir dos itemsets determinados na 1º etapa são extraídas as regras de associação existentes)

3. diagrama de atividades inerente a sua realização;

4. exemplos em que se visualiza a realização de cada uma de suas etapas e percebe-se semelhanças e diferenças entre o Apriori Tradicional e suas variações (Apriori Group e Fuzzy);

- No capítulo 6 será proposto estudo, com vistas a otimizar a implementação do algoritmo *Apriori* e variantes quando da seleção dos candidatos a itemsets-frequentes-k.

Na 2ª fase da etapa Apriori_gen é selecionado grande quantitativo de itemsets candidatos k, onde uma grande maioria não possui a totalidade de seus k conjuntos de k-1 elementos sendo itemsets freqüentes k-1 (o que elimina sua possibilidade de ser um itemset freqüente k).

A proposta dessa otimização é gerar um quantitativo menor de itemsets candidatos k, pois, em vez de serem gerados a partir da combinação dos itemsets freqüentes k-1, eles serão gerados a partir da combinação dos itemsets freqüentes k-1 e itemsets freqüentes 1.

- No capítulo 7 será apresentado estudo de caso em que serão extraídas diversas regras de associação contidas implicitamente no banco de dados de alunos e ex-alunos desta instituição.

Sobre o banco de dados alvo do estudo de caso (dados georreferenciados dos alunos e ex-alunos), será extraídas regras de associação existentes aplicando-se duas variantes do Algoritmo Apriori (Apriori Tradicional e *Fuzzy*), a partir de dois cenários distintos elaborados.

Por fim, se realizará comparação dos resultados obtidos com os Algoritmos *Apriori* e *Apriori Fuzzy*, nos dois cenários elaborados a fim de destacar semelhanças e discrepâncias entre as possibilidades de cada uma das variantes do Algoritmo *Apriori*;

- No capítulo 8 será apresentada conclusão do trabalho com análise dos resultados obtidos.

- No capítulo 9 será apresentada a bibliografia.

- No Anexo 1 será apresentada relação exaustiva dos resultados obtidos (itemsets freqüentes e regras de associação determinadas para cada um dos dois cenários / ferramentas utilizados)

- No anexo II será apresentado código fonte do algoritmo desenvolvido (Apriori Tradicional e *Fuzzy*)

1.2. OBJETIVOS

1.2.1. Objetivo geral

O objetivo deste trabalho é obter conhecimento, através da extração de regras de associação, de uma grande massa de dados. Esta extração será realizada utilizando-se as ferramentas "*Apriori*" e "*Apriori Fuzzy*".

1.2.2. Objetivo específico

Este trabalho tem os seguintes objetivos:

1. Extrair conhecimento do banco de dados dos alunos da UERJ, através co conhecimento de regras de associação existentes entre seus diversos dados;

2. Confrontar os resultados obtidos (regras de associação descobertas) a partir da utilização dos algoritmos *Apriori* e *Apriori Fuzzy*; e
3. Apresentar possibilidades de otimização de desempenho computacional na execução dos algoritmos *Apriori* e variações.

CAPÍTULO II

2. MINERAÇÃO DE DADOS

Definida como a tecnologia empregada para revelar informações estratégicas escondidas em grandes massas de dados e descrever características do passado, assim como predizer tendências para o futuro (GIMENES, 2000). É utilizada em diversas áreas, como em análise de riscos, *marketing* direcionado, controle de qualidade e exame de dados.

Mineração de dados, ou *Data Mining*, é o processo de análise de conjuntos de dados, cujo objetivo é a descoberta de padrões interessantes, que podem representar informações úteis.

Um padrão é definido como sendo uma afirmação sobre uma distribuição probabilística. Tais padrões são expressos, principalmente, na forma de regras, fórmulas e funções (SOUZA, 2003).

Descoberta de conhecimento em banco de dados, freqüentemente abreviada como *KDD* (*Knowledge Discovery in Databases*), é um processo que envolve a automação da identificação e do reconhecimento de padrões em um banco de dados; divide-se em seis fases:

1. Limpeza dos dados: etapa onde são eliminados ruídos e dados inconsistentes.

2. Integração dos dados: etapa onde diferentes fontes de dados podem ser combinadas, produzindo um único repositório de dados;

3. Seleção: etapa onde são selecionados os atributos que interessam ao usuário;

4. Transformação dos dados: etapa onde os dados são transformados num formato apropriado para aplicação de algoritmos de mineração;

5. Mineração: consiste na aplicação de técnicas inteligentes, a fim de se extrair padrões interessantes;

6. Visualização dos resultados: etapa onde são utilizadas as técnicas de representação de conhecimento, com o objetivo de apresentar ao usuário o resultado dos dados minerados (AMO, 2004). O resultado da mineração de dados pode chegar às seguintes descobertas: regras de associação, padrão seqüencial e árvores de classificação.

i. Regras de associação.
Exemplo: Quando o cliente compra um equipamento de vídeo, ele também obtém um outro artigo eletrônico.

ii. Padrão seqüencial.
Exemplo: Um cliente adquire uma câmera e, dentro de três meses, obtém itens fotográficos, e dentro de seis meses, um item acessório.

iii. Árvores de classificação.
Exemplo: Clientes podem ser classificados pela freqüência de visitas, pelo tipo de financiamento usado, pela quantidade de compras ou pela necessidade de certos tipos de produto (ELMASRI; NAVATHE, 2000).

CAPÍTULO III

3. MINERAÇÃO DE REGRAS DE ASSOCIAÇÃO

Pode-se dizer que há uma relação inversa entre o volume de informações existentes e a dificuldade de obtenção de conhecimento estratégico contido nessas informações.

Apesar de as informações necessárias e significativas para tomada de decisão não serem significativamente volumosas, geralmente elas não estão disponíveis de modo imediato, exigindo sua extração a partir de grandes quantidades de dados.

Conforme citado anteriormente, quanto maior a quantidade de dados disponível, mais difícil se torna a extração de informações.

Uma seqüência natural de procedimentos a serem adotados à extração seria: Dados → Informação → Conhecimento → Decisão. Neste contexto, o desafio que se apresenta para as organizações pode ser encarado como a resolução das duas questões básicas citadas na introdução:

1. Como organizar os dados?
2. Como extrair conhecimento dos dados organizados?

A primeira questão, conforme já foi dito, pode ser equacionada através da construção de um *DataWarehouse*, tecnologia que permite armazenar as informações, anteriormente dispersas, através da identificação, compreensão, integração e agregação dos dados, de forma a posicioná-los nos locais mais apropriados visando a atender à estratégia organizacional das empresas (Brackett, 1996).

A segunda questão, para extrair conhecimento de um sistema de *DataWarehouse,* são necessárias ferramentas de exploração, hoje conhecidas como *Data Mining* (Mineração de Dados). O *Data Mining* reúne uma série de técnicas, com destaque para as estatísticas,

probabilísticas e de inteligência computacional, capazes de fornecer respostas a várias questões ou mesmo descobrir novas informações em grandes bancos de dados. O *Data Mining* é especialmente útil em casos nos quais não se conhece a pergunta, mas, mesmo assim, existe a necessidade de respostas. Isto o distingue, por exemplo, de um *Executive Information System1* (Nigro, 1997).

Existem diversas técnicas de *Data Mining* disponíveis na literatura (Chen *et alli*, 1996; Cheung *et alli*, 1996). Uma das técnicas mais atraentes é a Mineração de Regras de Associação, que tem como destaque o algoritmo *Apriori*. Ele pode trabalhar com um número grande de atributos, gerando várias alternativas combinatórias entre eles. O algoritmo *Apriori* realiza buscas sucessivas em toda a base de dados, mantendo um ótimo desempenho em termos de tempo de processamento (Agrawal & Srikant, 1994).

O processo de extração de regras de associação é uma técnica que encontra relacionamentos entre a ocorrência de itens em transações de uma base de dados. Seja $I = \{i_{1,\ldots,} i_n\}$ um conjunto de literais, denominados itens. Um conjunto $X \subseteq I$ é chamado de ***itemset***. Um *itemset X* com *k* elementos é chamado de ***itemset*-k**. Uma regra de associação é uma expressão da forma $X \rightarrow Y$, onde *X* e *Y* são itemsets. O **suporte** da regra é a fração do número de transações que contém $X \cup Y$ em *I*. A **confiança** é a fração do número de transações contendo *X* que também contém *Y*.

Um exemplo de regra de associação envolvendo dados de uma cesta de compras é: "70% das compras que contém notebook também contém MP3-player e 7% de todas as compras contém esses dois itens". Nesse exemplo, 70% é a confiança da regra e 7% é o suporte da regra. O problema de minerar regras de associação consiste em encontrar **regras fortes**, que são as regras que satisfazem as restrições de suporte mínimo e de confiança mínima especificadas pelo usuário.

---

1 O EIS é um poderoso veículo de informação que torna possível a visualização de informações disponíveis nas bases de dados da empresa ou do ambiente externo, onde possui capacidade de *drill-down*, ou seja aprofundamento em detalhes de acordo com as necessidade do executivo, facilitando a análise de exceções por meio de parametrização do próprio executivo" – FONTE: http://www.inf.furb.br/%7Ezamba/artigos/Monitorando_Clientes.pdf, acessado em 23/07/2009

Os primeiros algoritmos de mineração de dados foram o AIS e SETM. Posteriormente foi apresentado o algoritmo *Apriori,* que hoje é o algoritmo mais conhecido de mineração de regras de associação (Agrawal, 1993). Após a apresentação do algoritmo *Apriori*, novos algoritmos foram desenvolvidos para tornar a tarefa de mineração mais eficiente, dentre eles destacam-se os algoritmos *Partition e FP-Growth,* entre outros, que podem ser resumidos da seguinte forma:

- Partition: dado um conjunto de objetos descritos por múltiplos valores (atributos), os mesmos são divididos em grupos (clusters) homogêneos de maneira a: maximizar a similaridade de objetos dentro de um mesmo cluster e minimizar a similaridade de objetos entre clusters distintos. Devendo ser atribuída uma descrição para cada cluster formado.

Um bom método de agrupamento fornece grupos de alta qualidade com alta similaridade intra-grupo e baixa similaridade inter-grupo, sendo que a qualidade do resultado de um agrupamento depende tanto da medida de similaridade usada pelo método como da sua implementação, podendo ser medida também pela sua habilidade para descobrir os padrões escondidos.

- FP-Growth: bastante similar ao algoritmo *Apriori*, apresenta como diferencial, a possibilidade de determinar os itemsets freqüentes a partir de incrementações a resultados obtidos anteriormente. Ou seja, diferentemente dos demais algoritmos, cada vez que a base original é alterada, os itemsets freqüentes, e, consequentemente, as regras de associação existentes serão “atualizados”, em vez de novamente levantados.

Esta possibilidade se deve ao fato de os objetos serem classificados em árvores que são atualizadas em cada manipulação do BD original, sem que se perca o levantamento anteriormente realizado.

Uma grande ênfase foi também dada à definição de novas medidas de interesse para obtenção de informações significativas e mais particulares sobre o comportamento dos dados. Dentre as medidas alternativas de interesse propostas destacam-se: a medida de peculiaridade e os graus de interesses. Apesar dessas novas medidas terem sido propostas, as medidas de suporte e confiança apresentadas inicialmente continuam sendo as mais utilizadas.

3.1. Agrupamentos

O processo de mineração de regras de associação definido anteriormente determina se uma regra é ou não forte, através do uso das medidas de interesse *suporte* e *confiança*, as quais dependem diretamente da proporção da ocorrência de seus itens nas transações da base de dados.

No entanto, as ocorrências de um item na base não refletem o comportamento desse *item* se a base possuir conjuntos de transações relacionadas (agrupamentos). A fim de ilustrar esta situação, consideremos a base de dados *Paciente* apresentada na Tabela 1, com os dados referentes a pacientes de uma clínica médica, seus sintomas e diagnósticos.

Tabela 1. Base de dados *Paciente*

IdPaciente	Fumante	Alérgico	Sintoma	Diagnóstico
1	Sim	Não	Febre	Resfriado
2	Sim	Sim	Hipertensão	Distúrbio Hormonal
3	Não	Sim	Inflamação	Infecção Bactéria
3	Não	Sim	Coriza	Infecção Vírus
3	Não	Sim	Dor	Fratura
4	Sim	Não	Hipertensão	Distúrbio Hormonal

A regra de associação *Fumante=Não*; *Alérgico=SIM, suporte=50% e confiança= 100%* é extraída da base de dados *Paciente* através do processo tradicional de mineração, indicando que os pacientes *não-fumantes* tendem a ser alérgicos, sendo que 50% dos pacientes são *não-fumantes* e *alérgicos* e, dos pacientes *não-fumantes* 100% são *alérgicos*.

A regra anterior é enganosa, pois somente 25% dos pacientes (somente o paciente 3) são *não-fumantes*. Esse engano ocorrido deve-se ao fato da mineração tradicional desconsiderar os agrupamentos inerentes nos dados para computar as medidas de interesse, tratando as ocorrências de um mesmo item em um agrupamento como ocorrências distintas.

Uma alternativa para resolver esse problema de inconsistência seria remover os agrupamentos da base através de um pré-processamento dos dados, em uma etapa anterior ao processo de mineração, agrupando em uma única transação todos os itens envolvidos em um conjunto de transações de um agrupamento. A base de dados *Paciente* após esse procedimento de pré-processamento é apresentada na Tabela 2.

Tabela 2: Base de dados *Paciente* pré-processada

IdPaciente	Fumante	Alérgico	Sintoma	Diagnóstico
1	Sim	Não	Febre	Resfriado
2	Sim	Sim	Hipertensão	Distúrbio Hormonal
3	Não	Sim	Inflamação, Coriza e Dor.	Infecção Bactéria, Infecção Vírus e Fratura.
4	Sim	Não	Hipertensão	Distúrbio Hormonal

A partir da base pré-processada, a regra *Fumante=Não → Alérgico=Sim,* discutida anteriormente, passa a ter o valor de *suporte=25% e confiança=100%* , indicando que 25% dos pacientes são não-fumantes e alérgicos o que está correto.

No entanto, o pré-processamento que, a uma primeira vista parece ser solução dos problemas de agrupamentos de dados, gera um grande inconveniente: cria relacionamentos entre itens que na base original não existem, fazendo com que o resultado da mineração tenha regras errôneas.

Por exemplo, a seguinte regra *Coriza → Fratura, suporte=25% e confiança=100%* é extraída da base de dados *Paciente* pré-processada. Essa regra indica que os pacientes com *coriza* tendem a ter *fratura*, sendo que 25% dos pacientes apresentaram *coriza e fratura* e, dos pacientes que apresentam *coriza*, 100% também apresentam *fratura*.

Entretanto, pode-se observar que o *itemset {coriza, fratura}* não ocorreu em transações da base de dados original (apresentada na Tabela 1), portanto, o suporte dessa regra deveria ser 0%, pois essa associação não existe.

Do exemplo exposto, é possível perceber que o pré-processamento acarreta a perda de informações do relacionamento entre os dados, fazendo com que o processo de mineração gere regras enganosas. Assim, o pré-processamento dos dados não é uma solução efetiva para tratar bases de dados com agrupamentos.

3.2. Mineração de Regras de Associação usando Identificador de Agrupamentos.

Para agrupar os dados para o processo de mineração é necessário que a base a ser analisada tenha um atributo que seja o identificador dos agrupamentos. Para o exemplo da base *Paciente* (Tabela 1), o atributo identificador dos agrupamentos é o *idPaciente*, pois

transações com o mesmo valor para esse atributo constituem um agrupamento, e transações de diferentes agrupamentos têm valores distintos para esse atributo.

Ao agrupar os dados, os relacionamentos entre os mesmos devem ser mantidos e as duplicações devem ser contabilizadas como uma única ocorrência. Agrupando os dados por um conjunto de um ou mais atributos, múltiplas ocorrências de um *itemset i* em um agrupamento passam a ser contabilizadas como uma única ocorrência.

Assim, o número de *ocorrências-agrupadas* de um itemset *i* (oca_i) diminuirá e o suporte passará a ser contabilizado como $oca_i/|Da|$, onde $|Da|$ é o número de agrupamentos existentes na base *D*. A Tabela 3 mostra a diferença entre os valores de suporte convencional e o suporte calculado usando agrupamentos de alguns *itemsets i* existentes na base *Paciente* (Tabela 1).

Tabela 3: Diferença entre valores de suporte obtidos usando e não usando agrupamentos

Itemset i	*Suporte – Tab. 1*	*Suporte com Agrupamentos – Tab. 2*	*Suporte com Identificadores de Agrupamentos – Tab. 1 Agrupada em IdCliente*
Fumante=não	50% (3 em 6)	25% (1 em 4)	25% (1 em 4)
Alérgico=sim	67% (4 em 6)	50% (2 em 4)	50% (2 em 4)
Hipertensão, Distúrbio Hormonal.	33% (2 em 6)	50% (2 em 4)	50% (2 em 4)
Coriza, Infecção Vírus.	16% (1 em 6)	25% (1 em 4)	25% (1 em 4)
Coriza, Fratura.	0% (0 em 6)	25% (1 em 4)	0% (0 em 4)

Embora a amostra usada seja pequena, os valores apresentados na Tabela 3 mostram que o suporte calculado usando agrupamentos diverge, em muitos casos, do suporte calculado usando diretamente o número de ocorrências, razão pela qual o algoritmo *Apriori-Groups* não altera a base de dados original, sendo simplesmente uma variante do algoritmo *Apriori* que considera, em suas análises, os agrupamentos por meio de seu(s) identificador (res) do banco de dados.

3.3. Apuração de medidas de interesse baseada em Agrupamentos

O agrupamento, cuja definição é apresentada a seguir, é a unidade para o cálculo do suporte baseado em agrupamentos. Considere que *A* seja o conjunto de atributos ($|A| \geq 1$) da base de dados *D*, escolhido como unidade de análise no processo de mineração.

Definição 1: O **suporte** usando agrupamento (supa) de um itemset X é a razão entre o número de agrupamentos em que X ocorre (oca(X)) e o total de agrupamentos de D (|Da|): **supa(X)=oca(X)/|Da|**.

Definição 2: O **suporte** usando agrupamento (supa) de uma regra X→Y de R, é a razão entre o número de agrupamentos em que o itemset $X \cup Y$ ocorre (oca($X \cup Y$)), e o número total de agrupamentos de D (|Da|): **supa(X→Y) = oca($X \cup Y$)/|Da|**

Definição 3: A **confiança usando agrupamento (confa)** de uma regra X→Y é a fração do número de agrupamentos contendo X que também contém Y: **confa(X→Y)**=oca($X \cup Y$)/|oca(X)| = **supa(X→Y) / supa(X)**

Observe que, se os agrupamentos tiverem uma única transação, então as regras extraídas usando agrupamentos são as mesmas regras extraídas usando o processo normal de mineração de regras de associação e $|Da| = |D|$.

O cálculo das medidas de interesse baseado em agrupamento, discutido nesta subseção, é usado para computar o suporte e a confiança no algoritmo *Apriori-Group* apresentado no capítulo V.

CAPÍTULO IV

4. LÓGICA FUZZY

Aristóteles, filósofo grego (384 - 322 a.C.), além de ser o fundador da ciência da lógica, estabeleceu um conjunto de regras rígidas para que conclusões pudessem ser aceitas como válidas.

O emprego da lógica de Aristóteles levava a uma linha de raciocínio baseado em premissas e conclusões. Como, por exemplo, se observa que "todo ser vivo é mortal" (premissa 1), a seguir é constatado que "João é um ser vivo" (premissa 2); como conclusão, temos que "João é mortal". Desde então, a lógica booleana, assim chamada, tem sido binária, isto é, uma declaração é falsa ou verdadeira, não admitindo possibilidades parciais, tais quais, sentença relativamente verdadeira, argumento parcialmente válido, etc... .

Tal suposição e a lei da não-contradição, que coloca que "U e não-U" cobrem todas as possibilidades, formam a base do pensamento lógico ocidental (TAKEMURA, 2004).

A Lógica Difusa (*Fuzzy Logic*) viola essas suposições. O conceito de dualidade, o qual estabelece que algo pode e deve coexistir com o seu oposto, faz a lógica difusa parecer natural, até mesmo inevitável.

A lógica de Aristóteles trata dos valores "verdade" das afirmações, classificando-as como verdadeiras ou falsas. Não obstante, muitas das experiências humanas não podem ser classificadas simplesmente como verdadeiras ou falsas, sim ou não, branco ou preto.

Por exemplo: Aquele homem é alto ou baixo? A taxa de risco para aquele empreendimento é grande ou pequena? Um sim ou um não como resposta a tais questões é, na maioria das vezes, uma resposta incompleta e imperfeita.

Lógica *Fuzzy* pode ser definida como a lógica que suporta os modos de raciocínio aproximados, em vez de exatos, como estamos naturalmente acostumados a trabalhar. Ela

está baseada na teoria dos conjuntos nebulosos, e difere dos sistemas lógicos tradicionais em suas características e detalhes (TAKEMURA, 2004).

Em conjuntos clássicos, a transição entre conjuntos ocorre abruptamente. Nos nebulosos, esta transição ocorre de forma gradual e a função de inclusão tem um modo de definir diferente daquele para conjuntos clássicos. As fronteiras entre eles não são nitidamente definidas e um elemento pode pertencer com certo grau a um conjunto, podendo este grau variar entre zero e um.

Ex: em uma classificação booleana, ao se dizer que rendimentos mensais até R$ 3.000,00 são baixos, e a partir daí são altos, um salário de R$ 3.000,00 é baixo e um de R$ 3.000,01 é alto, enquanto em uma classificação difusa, um salário de R$ 3.000,00 é 100% baixo e 0% alto, enquanto um de R$ 3.000,01 é 99,99% baixo e 0,01% alto.

A função de inclusão de um conjunto nebuloso A é definida no seu universo de discurso, sendo caracterizada pela função MA: X Є [0,1], que mapeia cada elemento de X em um número real no intervalo (0,1). Para um particular elemento, a função representa o grau de inclusão no conjunto (FREITAS, 2004).

Funções de inclusão são ferramentas matemáticas simples, utilizadas para indicar a participação em um determinado conjunto e modelar o significado dos rótulos associados aos conjuntos, podendo representar, ainda, a maneira subjetiva pela qual um indivíduo entende uma determinada classe de objetos ou pessoas. Funções de inclusão podem ser obtidas de diversas maneiras.

Por exemplo: funções podem ser criadas a partir das percepções dos especialistas sobre um determinado assunto.

Outra maneira é obter funções a partir de medidas estatísticas. Tais funções podem ser alteradas por resultados obtidos durante os testes do sistema no qual a lógica nebulosa foi empregada (CRUZ, 2003).

4.1. Conceitos Básicos da Teoria dos Conjuntos Fuzzy

A teoria dos conjuntos nebulosos foi desenvolvida a partir de 1965 com os trabalhos de Lotfi Zadeh, professor na Universidade da Califórnia em Berkeley.

Formalmente, um conjunto nebuloso A do universo de discurso é definido por uma função de pertinência □$_A$: Ω → [0; 1]. Essa função associa a cada elemento x o grau □$_A$(x), com o qual x pertence a A. A função de pertinência $\mu_A(x)$ indica o grau de compatibilidade entre x e o conceito expresso por A:

$\mu_A(x) = 1$ indica que x é completamente compatível com A;
$\mu_A(x) = 0$ indica que x é completamente incompatível com A;
$0 < \mu_A(x) < 1$ indica que x é parcialmente compatível com A, com grau $\mu_A(x)$.

Um conjunto A da teoria clássica dos conjuntos pode ser visto como um conjunto nebuloso específico, denominado usualmente de "crisp", para o qual μ_A: Ω → {0; 1}, ou seja, a pertinência é do tipo "tudo ou nada", "sim ou não", e não gradual como para os conjuntos nebulosos.

A diferença entre estes conceitos em relação à variável idade é ilustrada abaixo, nas representações do conceito "adolescente" através de um conjunto "crisp" e de um conjunto nebuloso.

O conjunto "crisp" não exprime completamente o conceito de "adolescente", pois uma pessoa com 12 anos e 11 meses seria considerada completamente incompatível com este conceito.

Na verdade, qualquer intervalo "crisp" que se tome para representar este conceito é arbitrário. Já o conjunto nebuloso permite exprimir que qualquer pessoa com idade entre 13 e 17 anos é um adolescente, acima de 19 ou abaixo de 11 não é considerado um adolescente, e no intervalo [11; 13] (respectivamente [17; 19]) é considerado tanto mais adolescente quanto mais próxima de 13 (respectivamente de 17) é sua idade.

CAPÍTULO V

5. ALGORITMO APRIORI

O algoritmo *Apriori* é um dos algoritmos mais conhecidos quando o assunto é mineração de regras de associação em grandes bancos de dados centralizados.

A funcionalidade deste algoritmo consiste em verificar existência de regras de associação existentes entre os valores dos diversos campos de um *datawarehouse.*

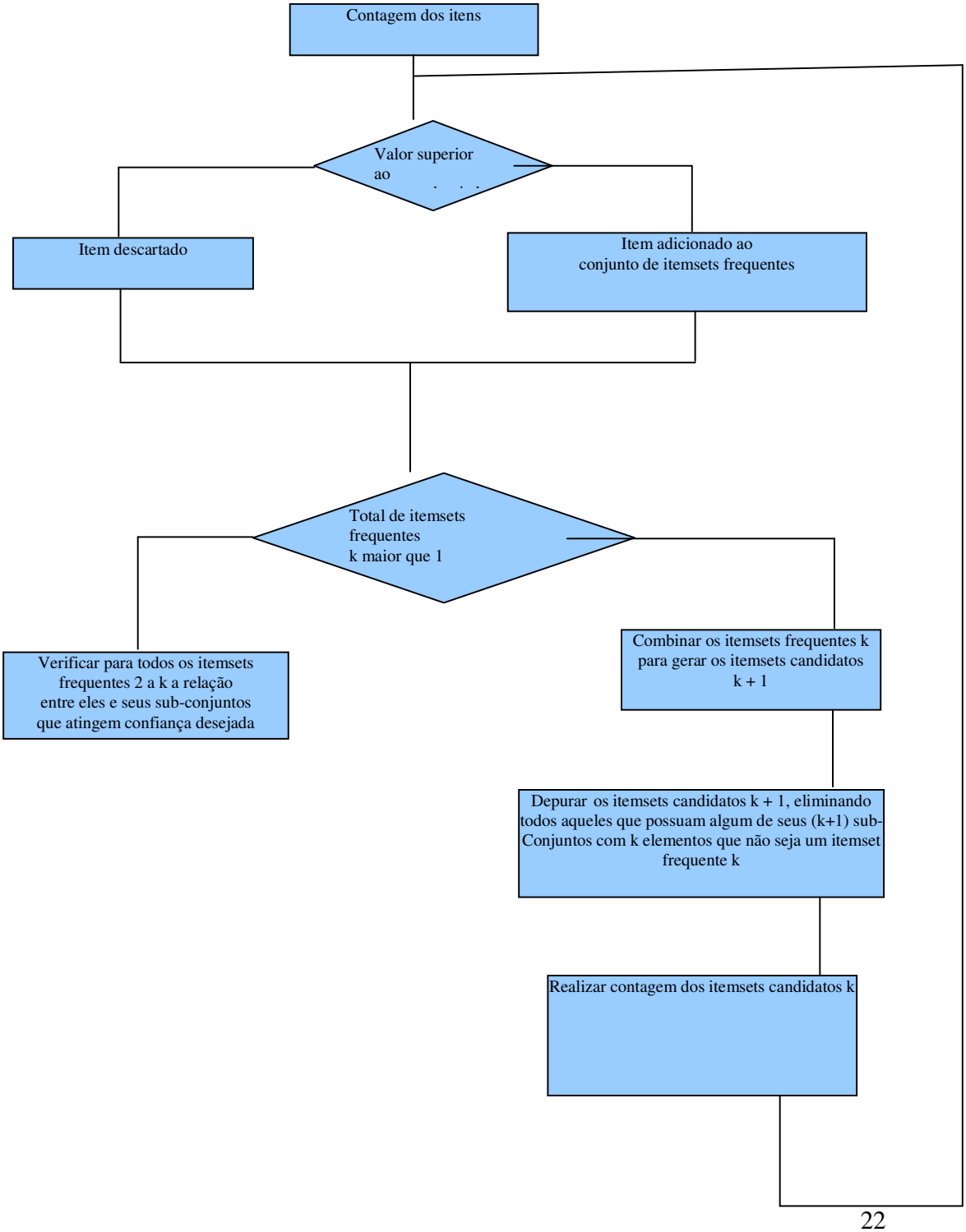
Desenvolvido por Agrawal e Srikant em 1.994, divide-se em duas funções:

Função 1: Comumente denominada "Apriori_gen", encontra todos os conjuntos de itens freqüentes, denominados *itemsets freqüentes* (L_k), que, na verdade, são os valores combinados de "*k"* campos do banco de dados, com ocorrência superior ao percentual de suporte desejado.

Função 2: Comumente denominada "Genrules", já tendo conhecimento dos *itemsets freqüentes*, utiliza-os para extrair regras de associação.

A depender do tipo de variáveis a serem pesquisados, trabalhamos com três tipos de Algoritmo *Apriori*, que são os algoritmos APRIORI FUZZY e APRIORI TRADICIONAL e APRIORI GROUP.

De modo ilustrado, pode-se visualizar o algoritmo APRIORI por meio do seguinte diagrama de atividades:


Contagem dos itens
Valor superior
ao
Item descartado
Item adicionado ao
conjunto de itemsets frequentes
Total de itemsets
frequentes
k maior que 1
Verificar para todos os itemsets
frequentes 2 a k a relação
entre eles e seus sub-conjuntos
que atingem confiança desejada
Combinar os itemsets frequentes k
para gerar os itemsets candidatos
k + 1
Depurar os itemsets candidatos k + 1, eliminando
todos aqueles que possuam algum de seus (k+1) sub-
Conjuntos com k elementos que não seja um itemset
frequente k
Realizar contagem dos itemsets candidatos k

5.1. Algoritmo *Apriori* Tradicional

Em prosseguimento às explicações realizadas na introdução deste capítulo, que são válidas para os três tipos de Algoritmo *Apriori*, o primeiro passo do algoritmo *Apriori* é realizar a contagem de ocorrências dos itens de cada campo do Banco de Dados, a fim de determinar os *itemsets freqüentes* de tamanho unitário (*itemsets freqüentes* L_1).

Em outras palavras, nesse momento realiza-se a contagem de todos os valores de todos os campos de meu banco de dados, definindo-se como *itemsets freqüentes* L_1 os valores de cada campo do Banco de Dados com ocorrência superior ao suporte estipulado.

Como objeto de estudo e com a finalidade de se obter melhor entendimento a respeito das funções do algoritmo *Apriori*, explicar-se-á a função supracitada tendo como exemplo o pequeno *datawarehouse* abaixo:

Artigo (item)	Número que o representa
Pão	1
Leite	2
Açúcar	3
Papel Higiênico	4
Manteiga	5
Fralda	6
Cerveja	7
Refrigerante	8
Iogurte	9
Suco	10

Sendo que o banco de dados contém transações efetuadas pelos clientes e seus rendimentos

Clientes	Mercadorias	Rendimento mensal cliente
101	{1, 3,5}	7.000,
102	{1, 2, 3, 7,5}	3.000,
103	{1, 2, 4,9}	5.000,
104	{1, 2, 3, 5,9}	8.000,
105	{1, 3, 4, 5, 6,8}	1.000,
106	{2, 8,9}	10.000,

Supondo desejar-se um suporte igual ou superior a 50%, interessarão os valores (*itemsets freqüentes* L_1) que tenham freqüência de ocorrência igual ou superior a 3 (equivalente a 50% do total das 6 tuplas). Logo, tem-se a partir dos seguintes *candidatos a itemsets freqüentes* L_1 (que correspondem a totalidade de itens de meu banco de dados):

Mercadoria 1 – 5 ocorrências

Mercadoria 2 – 4 ocorrências

Mercadoria 3 – 4 ocorrências

Mercadoria 4 – 2 ocorrências

Mercadoria 5 – 4 ocorrências

Mercadoria 6 – 1 ocorrência

Mercadoria 7 – 1 ocorrência

Mercadoria 8 – 2 ocorrências

Mercadoria 9 – 3 ocorrências

Salário 1.000, – 1 ocorrência

Salário 3.000, – 1 ocorrência

Salário 5.000, – 1 ocorrência

Salário 7.000, – 1 ocorrência

Salário 8.000, – 1 ocorrência

Salário 10.000, – 1 ocorrência

Tem-se como *itemsets freqüentes* L_1 as mercadorias 1, 2, 3, 5, 9.

O passo seguinte do algoritmo será efetuado do seguinte modo:

1. Os *itemsets freqüentes* L_{k-1}, encontrados no passo anterior (k-1) são utilizados para gerar os conjuntos de itens potencialmente freqüentes, os *itemsets candidatos* (C_k), ou seja, a combinação de valores de "k-1" campos dos meus itemsets freqüentes k-1 irá resultar em conjuntos de "k" campos do banco de dados, que podem ter ocorrência superior ao suporte desejado.

2. A geração dos *itemsets candidatos*, de antemão, toma como argumento L_{k-1}, o conjunto de todos (k-1)-*itemsets freqüentes*. Para tanto, utiliza-se a função *Apriori_gen*, que retorna um superconjunto de todos os *k-itemsets freqüentes.* A intuição por trás desse procedimento é que se um *itemset* X tem suporte mínimo, todos os seus subconjuntos também terão (Agrawal & Shafer, 1996). Por analogia, se um subconjunto k-1 de um candidato c_k não possui suporte mínimo (ou seja, não é um itemset freqüente k-1), o candidato c_k também não o terá. A função, em um primeiro estágio, une L_{k-1} com L_{k-1}. No estágio seguinte, são eliminados os *itemsets candidatos ck*, desde que um dado (k-1)-*subset* de c_k não pertença a L_{k-1}. Uma das propostas deste trabalho, para fins de economia computacional, é gerar os *itemsets candidatos* (C_k), a partir dos *itemsets freqüentes* L_{k-1} e dos *itemsets freqüentes* L_1, que certamente gerará uma quantidade menor de itemsets candidatos (c_k), diminuindo a fase de eliminação dos itemsets candidatos c_k com sobconjuntos não pertencentes a L_{k-1}.

3. É realizada uma nova busca no banco de dados, contando-se o suporte de cada candidato em C_k.

Os passos 2 e 3 se darão da seguinte forma em nosso exemplo:

Mercadorias (1,2), (1,3), (1,5), (1,9), (2,3), (2,5), (2,9), (3,5), (3,9), (5,9), com as seguintes freqüências:

Mercadorias (1,2) – 3 ocorrências - *itemsets freqüentes* L_2
Mercadorias (1,3) – 4 ocorrências - *itemsets freqüentes* L_2
Mercadorias (1,5) – 4 ocorrências - *itemsets freqüentes* L_2
Mercadorias (1,9) – 2 ocorrências
Mercadorias (2,3) – 2 ocorrências
Mercadorias (2,5) – 2 ocorrências
Mercadorias (2,9) – 3 ocorrências - *itemsets freqüentes* L_2
Mercadorias (3,5) – 4 ocorrências - *itemsets freqüentes* L_2
Mercadorias (3,9) – 1 ocorrência
Mercadorias (5,9) – 1 ocorrência

De modo análogo à pesquisa acima, a partir de combinações dos valores dos *itemsets freqüentes* L_2 e dos *itemsets freqüentes* L_1 formam-se os *itemsets candidatos* L_3. No caso em análise tem-se os seguintes *itemsets candidatos* L_3:

Itemsets freqüentes L_1 – {1,2,3,5,9}
Itemsets freqüentes L_2 – {(1,2),(1,3),(1,5),(2,9),(3,5)}
Itemsets candidatos C_3 – {(1,2,3), (1,2,5), (1,2,9), (1,3,2), (1,3,5), (1,3,9), (1,5,2), (1,5,3), (1,5,9), (2,9,1), (2,9,3), (2,9,5), (3,5,1), (3,5,2), (3,5,9)}

Destes itemsets candidatos, os primeiros a serem eliminados são os itemsets (1,3,2), (1,5,2), (1,5,3), (2,9,1), (3,5,1) por serem iguais a outros itemsets candidatos.

Posteriormente, dos itemsets candidatos restantes, são eliminados os itemsets (1,2,3), por possuir o subconjunto (2,3); (1,2,5), por possuir o subconjunto (2,3); (1,2,9), por possuir o subconjunto (1,9); (1,3,9), por possuir o subconjunto (1,9); (1,5,9), por possuir o subconjunto (5,9); (2,9,3), por possuir o subconjunto (3,9); (2,9,5), por possuir o subconjunto (2,5); (3,5,2), por possuir o subconjunto (2,3); (3,5,9), por possuir o subconjunto (3,9).

Portanto, restou somente o itemset candidato (1,3,2), com 2 ocorrências, o que não faz dele um itemset freqüente L_3

O último passo é a descoberta das regras de associação, obtida através da função *Genrules*. A geração de regras, para qualquer *itemset freqüente*, significa encontrar todos os *subsets* não vazios de *l*. Assim, para todo e qualquer *subset a*, produz-se uma regra $a \rightarrow (l - a)$ somente se a razão (suporte (*l*)/suporte(*a*)) é ao menos igual à confiança mínima estabelecida pelo usuário.

Para gerar regras com múltiplos conseqüentes, são considerados todos os *subsets*. Por exemplo, dado um *itemset ABCD*, considera-se primeiro o *subset ABC*, seguido de *AB*, etc. Se $ABC \rightarrow D$ não atinge uma confiança suficiente (confiança < *minconf*), não é necessário verificar se $AB \rightarrow CD$.

No exemplo dado, tem-se os seguintes itemsets freqüentes L_2 = {(1,2), (1,3), (1,5), (2,9), (3,5)}, que servirão como "geradores de subsets", gerando as seguintes pesquisas de regras de associação:

1 → (1,2) – confiança 60%
2 → (1,2) – confiança 75%
1 → (1,3) – confiança 100%
3 → (1,3) – confiança 100%
1 → (1,5) – confiança 80%
5 → (1,5) – confiança 80%
2 → (2,9) – confiança 75%
9 → (2,9) – confiança 80%

3 → (3,5) – confiança 100%

5 → (3,5) – confiança 100%

Supondo que se deseje uma confiança mínima de 80%, serão relacionamentos válidos, os seguintes:

- 1 → (1,3); 3 → (1,3); 1 → (1,5); 5 → (1,5); 9 → (2,9); 3 → (3,5); 5 → (3,5)

5.2. O algoritmo *Apriori-Group*

A mineração de regras de associação baseada em agrupamentos é mais complexa do que a mineração de regras de associação tradicional, devido principalmente à necessidade de identificação dos *agrupamentos* para a contagem correta do suporte de um *itemset*. O algoritmo *Apriori-Group* estende o algoritmo *Apriori* e permite obter regras de associação confiáveis em bases com agrupamentos de transações.

O algoritmo *Apriori-Group* segue os seguintes passos:

1. Cria agrupamentos entre tuplas com o mesmo valor do atributo identificador do agrupamento.

2. Percorre a base de dados e o contador de cada item que ocorre em um agrupamento é incrementado.

3. Os *itemsets* que satisfazem o mínimo suporte passam a fazer parte do conjunto L_1 de *itemsets_1* freqüentes. Os *itemsets_k* freqüentes são combinados entre si para formar *itemsets_k+1* candidatos através da função *Apriori-Gen*.

4. A função *Apriori-Gen* gera *itemsets_k+1*, combinando *itemsets_k* e removendo os que têm subconjuntos de tamanho *k* não freqüentes.

5. Para cada conjunto C_k de *itemsets_k* candidatos, a base de dados é varrida e o contador de cada *itemset_k* é incrementado quando ele ocorre em um agrupamento distinto.

6. Os *itemsets_k* candidatos que satisfazem o mínimo suporte passam a fazer parte do conjunto L_k de *itemsets_k* freqüentes. Para todos os *itemsets* freqüentes encontrados são geradas regras e aquelas que satisfazem a mínima confiança são retornadas pelo algoritmo.

Analisando-se as etapas do algoritmo *Apriori-Group*, observa-se que ele é essencialmente igual ao algoritmo *Apriori* tradicional. No exemplo apresentado anteriormente, pode-se observar que as informações foram agrupadas por cliente, ou seja, o caso acima já é uma aplicação do algoritmo *Apriori-Group*, sendo o atributo identificador do sistema, o campo cliente.

5.3. O algoritmo Apriori Fuzzy

O algoritmo *Apriori Fuzzy*, que pode ou não utilizar o conceito de agrupamentos por atributo identificador apresentado no *Apriori-Group*, diferencia-se dos demais pelos seguintes motivos:

1. Em vez de efetuar-se contagem simples de itens, irá-se calcular sua pertinência relacionada a variáveis lingüísticas correspondentes.

2. A contagem irá se transformar no somatório dos índices de pertinências que vinculam os valores às variáveis lingüísticas.

Vejamos o exemplo apresentado anteriormente trabalhado pelo algoritmo *Apriori Fuzzy*:

Artigo (item)	Número que o representa
Pão	1
Leite	2
Açúcar	3
Papel Higiênico	4
Manteiga	5
Fralda	6
Cerveja	7
Refrigerante	8
Iogurte	9
Suco	10

Sendo que meu banco de dados contém transações efetuadas pelos clientes e seus rendimentos

Clientes	Mercadorias	Rendimento mensal cliente
101	{1,3,5}	7.000,
102	{2,1,3,7,5}	3.000,
103	{4,9,2,1}	5.000,
104	{5,2,1,3,9}	8.000,
105	{1,8,6,4,3,5}	1.000,
106	{9,2,8}	10.000,

Sendo que as equações que associam os salários aos conceitos {baixo, médio e alto} são os seguintes:

Salário baixo: superior a R$ 5.000, - pertinência 0
entre R$ 0, e R$ 5.000, - pertinência = (5000 – salário) / 5000
inferior a R$ 0, - pertinência = 1

Salário médio: superior a R$ 10.000, - pertinência 0
entre R$ 0, e R$ 10.000, - pertinência = (-|salário-5000|+5000) / 5000
inferior a R$ 0, - pertinência = 0

Salário alto: superior a R$ 10.000, - pertinência 1
entre R$ 5.000, e R$ 10.000, - pertinência = (salário-5000) / 5000
inferior a R$ 5.000, - pertinência = 0

Nesse sentido, o banco de dados pode ser reescrito da seguinte forma:

Clientes	Mercadorias		Rendimento mensal cliente		
			Baixo	Médio	Alto
101	{1,3,5}	7.000,	0	0,6	0,4
102	{1,2,3,5,7}	3.000,	0,4	0,6	0
103	{1,2,4,9}	5.000,	0	1	0
104	{1,2,3,5,9}	8.000,	0	0,4	0,6
105	{1,3,4,5,6,8}	4.000,	0,2	0,8	0
106	{2,8,9}	10.000,	0	0	1

Importante observar que TODAS as variáveis devem ser transformadas em variáveis lingüísticas. O fato de aparentemente as mercadorias não sofrerem alteração se deve a elas estarem sendo trabalhadas da seguinte forma:

- Ao manipular o conjunto {1,3,5} deve-se dizer que este conjunto tem 100% de pertinência com as variáveis lingüísticas {1,3,5} e 0% de pertinência com as demais, porém nada impede que cada mercadoria seja atrelada a variáveis lingüísticas do tipo {preço baixo, médio e alto}, e possua valores de pertinência entre 0 e 1.

Mercadoria 1 – 5 ocorrências
Mercadoria 2 – 4 ocorrências
Mercadoria 3 – 4 ocorrências
Mercadoria 4 – 2 ocorrências
Mercadoria 5 – 4 ocorrências
Mercadoria 6 – 1 ocorrência
Mercadoria 7 – 1 ocorrência
Mercadoria 8 – 2 ocorrências
Mercadoria 9 – 3 ocorrências
Salário baixo – 0,6 ocorrência
Salário médio – 3,4 ocorrência
Salário alto – 2 ocorrências

Teremos como *itemsets freqüentes* L_1 as mercadorias 1, 2, 3, 5, 9 e salário médio (sm).

O passo seguinte do algoritmo será efetuado do seguinte modo:

3. Do mesmo modo que nos casos anteriores de algoritmo *Apriori*, os *itemsets freqüentes* L_{k-1}, encontrados no passo anterior (k-1), são utilizados para gerar os conjuntos de itens potencialmente freqüentes, os *itemsets candidatos* (C_k), ou seja, a combinação de valores de "k" campos do banco de dados que podem ter ocorrência superior ao suporte desejado

4. A geração dos itemsets candidatos (c_k) seguirá a mesma forma que nos algoritmos anteriores, observando-se que em vez de realizar contagens simples irá se trabalhar, conforme dito, com índices de pertinências de valores a variáveis lingüísticas apresentadas.

Os passo 3 e 4 se darão da seguinte forma em nosso exemplo:

Mercadorias (1,2), (1,3), (1,5), (1,9), (1,sm), (2,3), (2,5), (2,9), (2,sm) (3,5), (3,9), (3,sm), (5,9), (5,sm) e (9,sm), com as seguintes freqüências:

Mercadorias (1,2) – 3 ocorrências - *itemsets freqüentes* L_2
Mercadorias (1,3) – 4 ocorrências - *itemsets freqüentes* L_2
Mercadorias (1,5) – 4 ocorrências - *itemsets freqüentes* L_2
Mercadorias (1,9) – 2 ocorrências
Mercadorias (1,sm) – 3,4 ocorrências - *itemsets freqüentes* L_2
Mercadorias (2,3) – 2 ocorrências
Mercadorias (2,5) – 2 ocorrências
Mercadorias (2,9) – 3 ocorrências - *itemsets freqüentes* L_2
Mercadorias (2,sm) – 2 ocorrências
Mercadorias (3,5) – 4 ocorrências - *itemsets freqüentes* L_2
Mercadorias (3,9) – 1 ocorrência
Mercadorias (3,sm) – 2,4 ocorrências
Mercadorias (5,9) – 1 ocorrência
Mercadorias (5,sm) – 2,4 ocorrências
Mercadorias (9,sm) – 1,4 ocorrências

De modo análogo à pesquisa acima, a partir de combinações dos valores dos *itemsets freqüentes* L_2 e dos *itemsets freqüentes* L_1 formam-se os *itemsets candidatos* L_3. No caso em análise tem-se os seguintes *itemsets candidatos* L_3:

Itemsets freqüentes L_1 – {1,2,3,5,9,sm}
Itemsets freqüentes L_2 – {(1,2),(1,3),(1,5),(2,9),(3,5),(1,sm)}

Itemsets candidatos C_3 – {(1,2,3), (1,2,5), (1,2,9), (1,2,sm), (1,3,2), (1,3,5), (1,3,9), (1,3,sm), (1,5,2), (1,5,3), (1,5,9), (1,5,sm), (2,9,1), (2,9,3), (2,9,5), (2,9,sm), (3,5,1), (3,5,2), (3,5,9), (3,5,sm),(1,sm,2), (1,sm,3), (1,sm,5), (1,sm,9)}

Destes itemsets candidatos, os primeiros a serem eliminados são os itemsets (1,3,2), (1,5,2), (1,5,3), (2,9,1), (3,5,1), (1,sm,2), (1,sm,3), (1,sm,5), por serem iguais a outros itemsets candidatos.

Posteriormente, dos itemsets candidatos restantes são eliminados os itemsets (1,2,3), por possuir o subconjunto (2,3); (1,2,5), por possuir o subconjunto (2,3); (1,2,9), por possuir o subconjunto (1,9); (1,3,9), por possuir o subconjunto (1,9); (1,5,9), por possuir o subconjunto (5,9); (2,9,3), por possuir o subconjunto (3,9); (2,9,5), por possuir o subconjunto (2,5); (3,5,2), por possuir o subconjunto (2,3); (3,5,9), por possuir o subconjunto (3,9); todos os candidatos que possuem o item "sm", uma vez que o único itemset freqüente L_2 que possui "sm" é o (1,sm), logo qualquer itemset candidato L_3 da forma (x,y,sm) terá x ou y diferente de 1, logo o subconjunto (elemento diferente de 1, sm) não é itemset freqüente L_2.

Portanto, restou somente o itemset candidato (1,3,2), com 2 ocorrências, o que não faz dele um itemset freqüente L_3

O último passo é a descoberta das regras de associação, obtida através da função *Genrules*. A geração de regras, para qualquer *itemset freqüente*, significa encontrar todos os *subsets* não vazios de *l*. Assim, para todo e qualquer *subset a*, produz-se uma regra $a \rightarrow (l - a)$ somente se a razão (suporte (*l*)/suporte(*a*)) é ao menos igual a confiança mínima estabelecida pelo usuário.

Para gerar regras com múltiplos conseqüentes, são considerados todos os *subsets*. Por exemplo, dado um *itemset ABCD*, considera-se primeiro o *subset ABC*, seguido de *AB*, etc. Se $ABC \rightarrow D$ não atinge uma confiança suficiente (confiança < *minconf*), não é necessário verificar se $AB \rightarrow CD$.

No exemplo dado, tem-se os seguintes itemsets freqüentes L_2 – {(1,2), (1,3), (1,5), (2,9), (3,5), (1,sm)}, que servirão como "geradores de subsets". Gerando as seguintes pesquisas de regras de associação:

1 → (1,2) – confiança 60%

2 → (1,2) – confiança 75%

1 → (1,3) – confiança 100%

3 → (1,3) – confiança 100%

1 → (1,5) – confiança 80%

5 → (1,5) – confiança 80%

2 → (2,9) – confiança 75%

9 → (2,9) – confiança 80%

3 → (3,5) – confiança 100%

5 → (3,5) – confiança 100%

1 → (1,sm) – confiança 100% (3,4/3,4)

sm → (1,sm) – confiança 100% (3,4/3,4)

Supondo que se deseje uma confiança mínima de 80%, serão relacionamentos válidos, os seguintes:

- 1 → (1,3); 3 → (1,3); 1 → (1,5); 5 → (1,5); 9 → (2,9); 3 → (3,5); 5 → (3,5); 1 → (1,sm); sm → (1,sm)

5.4. Comparação algoritmos Apriori

Apresentadas as teorias dos algoritmos *Apriori* ilustradas com exemplo, observa-se ser muito mais eficiente o Algoritmos *Apriori Fuzzy*, em relação àquele que trabalha com variáveis discretas, na manipulação de variáveis quantitativas.

Como forma de ilustrar a eficiência de um método relativamente ao outro, deve-se observar que o algoritmo *Apriori* tradicional não capturou nenhum itemset freqüente no

campo “Salário”, enquanto o *Apriori Fuzzy* identificou itemsets freqüentes e relacionamentos envolvendo o itemset “Salário médio”.

Relativamente a comparação entre os três algoritmos *Apriori*, pode se apresentar a seguinte síntese:

Algoritmo *Apriori*: eficiente em trabalhar variáveis não quantitativas, não vinculadas a variáveis lingüísticas.

Algoritmo *Apriori Fuzzy*: eficiente em trabalhar variáveis quantitativas e/ ou associadas a variáveis lingüísticas.

Algoritmo *Apriori Group*: esclarecendo poder existir os Algoritmo *Group Fuzzy* e Algoritmo *Group* Tradicional, caracteriza-se como ferramenta essencial para manipular dados que devem ser agrupados em função de determinado campo identificador (no exemplo apresentado, o campo identificador é o cliente).

CAPÍTULO VI

6. OTIMIZAÇÃO COMPUTACIONAL IMPLEMENTADA NO ALGORITMO *APRIORI* E VARIANTES

Conforme foi explanado em capítulos anteriores, o algoritmo *Apriori* pode ser descrito de modo sucinto como sendo a realização de buscas sucessivas em um repositório de dados, tendo como resultado a localização de relações de associação entre alguns desses dados.

Um dos focos do trabalho foi a otimização da realização de buscas em tal repositório. Para tanto foram analisadas as etapas do algoritmo *Apriori*, entendido o objetivo almejado em cada uma delas, e a partir daí analisou-se a possibilidade de otimização da mesma, da seguinte forma:

- Etapa 1: nessa etapa é realizada a contagem dos itemsets-candidatos-12 da seguinte forma:

    1. *Apriori Tradicional*: realiza-se uma contagem de cada um dos itens do repositório de dados, tornando-se itemsets-frequentes-1 os itemsets-candidatos-1 com suporte igual ou superior ao desejado;

    2. *Apriori Group*: realiza-se uma contagem de cada um dos itens do repositório de dados por identificador de agrupamento, tornando-se itemsets-frequentes-1 os itemsets-candidatos-1 com suporte igual ou superior ao desejado

2 Nesta primeira etapa, os itemsets candidatos são todos os itens de meu repositório de dados

3. *Apriori Fuzzy*: realiza-se o cálculo do índice de pertinência de cada item do repositório de dados à variáveis lingüísticas a elas relacionadas, torna-se-ão itemsets-frequentes-1 as variáveis lingüísticas com somatório de índices de pertinência igual ou superior ao suporte desejado

Como conclusão ao estudo desta etapa, não se verificou possibilidade de otimização das rotinas implementadas

- Etapa 2: nessa etapa é realizada a apuração dos itemsets-candidatos-k da seguinte forma:

Apriori Tradicional, Apriori Group e *Apriori Fuzzy*:

- Passo 1: Realiza-se a combinação dos itemsets-frequentes-(k-1), para geração dos candidatos a itemsets-candidatos-k;

- Passo 2: Realiza-se uma depuração dos candidatos a itemsets-candidatos-k, eliminando-se aqueles que possuem sub-conjunto k-1 diferente de qualquer itemset-frequente-(k-1);

- Passo 3: Realiza-se a contagem dos itemsets-candidatos-k restantes do passo anterior de modo semelhante ao explanado na Etapa 1, observando-se que no *Apriori Fuzzy*, os valores de freqüência de ocorrência serão os somatórios dos produtos dos valores de pertinência de cada combinação de ocorrência dos k itens do itemset-candidato-k

Nesta etapa verificou-se geração de grande quantidade de candidatos a itemsets-candidatos-k ao se combinar os itemsets-frequentes-(k-1), a otimização proposta nessa etapa foi a geração dos candidatos a itemsets-candidatos-k a partir da combinação dos

itemsets-frequentes-(k-1) e itemsets-frequentes-1, gerando uma quantidade significativamente menor de candidatos a serem testados no passo 2.

- Etapa 3: nessa etapa é realizada a apuração das regras de associação ao se verificar se a razão entre a freqüência de ocorrência dos itemsets-frequentes-k e seus sub-conjuntos (que obrigatoriamente também são itemsets-frequentes – observar passo 2 da etapa 2) satisfazem a confiança estabelecida, caso satisfaça verificou-se relação de regra de associação entre esses itens do subconjunto e os k itens do itemset-frequente-k.

Como resultado da alteração da implementação do *Apriori Tradicional, Apriori Group* e *Apriori Fuzzy*, verificou-se razoável ganho de desempenho computacional no passo 1 da etapa 2.

CAPÍTULO VII

7. ESTUDO DE CASO – APLICAÇÃO DOS ALGORITMOS *APRIORI* E *APRIORI FUZZY* - BANCO DE DADOS DE ALUNOS E EX-ALUNOS DA UERJ

7.1. Banco de Dados utilizado

Neste capítulo será realizado estudo de caso relativamente a repositório de dados de informações dos alunos desta instituição.

Com base nas regras de associação existentes entre os dados e extraídas por um e outro método, far-se-á um estudo comparativo entre ambos.

Do presente banco de dados, foram selecionados alguns campos cadastrais e relativos a informações sociais e geo econômicas, relativamente a 230 ex-alunos da instituição:

1 – Matrícula
2 – Sexo
3 – Colégio Anterior
4 – CR
5 – Salário
6 – Endereço
7 – CEP

Possuindo ainda os campos "CR-Fuzzy" e "Salário_Fuzzy" não contendo nenhuma informação originária, servindo apenas de apoio aos cálculos de pertinência realizados pelo algoritmo *Apriori Fuzzy*.

Os campos a serem trabalhados foram escolhidos com a finalidade de realizar estudo no sentido de tentar determinar relacionamentos entre variáveis georreferenciadas (Endereço, CEP), variáveis quantitativas (CR, salário) e variáveis qualitativas (Matrícula, sexo e colégio anterior)

Apesar de o presente estudo se dar com dados educacionais fictícios, o mesmo é plenamente aplicável a bancos de dados educacionais reais.

7.2. Cenários estabelecidos de suporte e confiança para utilização dos algoritmos Apriori e Apriori Fuzzy

Ao se realizar a pesquisa relativamente à extração de regras de associação elaborou-se os seguintes cenários:

Cenário 1

1.Suporte de 5%
2.Confiança de 50%
3.Dados agrupados relativamente ao campo matrícula

Cenário 2

1.Suporte de 20%
2.Confiança de 50%
3.Dados agrupados relativamente ao campo matrícula

E utilizando-se as seguintes fórmulas de cálculo de pertinência:

Salário Alto: 1. Se Salário < 4000 então Salário Alto = 0

2. Se Salário > 10000 então Salário Alto = 1

3. Senão Salário Alto = (Salário – 4000) / 6000

Salário Médio: 1. Se Salário< 500 ou Salário> 10000 então Salário Médio = 0

2. Se Salário > 4000 então Salário Médio = (10000-salário) / 6000

3. Senão Salário Médio = (Salário – 3500) / 3500

Salário Baixo: 1. Se Salário > 4000 então Salário Baixo = 0

2. Se Salário < 500 então Salário Baixo = 1

3. Senão Salário Baixo = (3500 - salário) / 3500

CR Alto: 1. Se CR < 50 então CR Alto = 0

2.Se CR > 90 então CR Alto = 1

3.Senão CR Alto = (CR – 50) / 40

CR Médio:

1.Se CR < 10 ou CR > 90 então CR Médio = 0

2.Se CR > 50 então CR Médio = (90-CR) / 40

3.Senão CR Médio = (CR – 40) / 40

CR Baixo:

1.Se CR > 50 então CR Baixo = 0

2.Se CR < 10 então CR Baixo = 1

3.Senão CR Baixo = (40 - CR) / 40

Para os campos Salário e CR o somatório de valores de pertinência de cada valor relativamente ao conjunto de variáveis lingüísticas relativamente a cada campo deve ser sempre igual a 1

Ex.: CR = 65 CR alto = 0,375; CR médio = 0,625
CR baixo = 0

0,375 + 0,625 + 0 = 1 (ESSE SOMATÓRIO SEMPRE DEVE SER IGUAL A 1)

Nos demais campos, o conjunto de variáveis lingüísticas foi o conjunto de itemsets-candidatos-1, considerando-se a pertinência = 1 para a variável lingüística que era igual ao item e pertinência = 0 relativamente às demais variáveis lingüísticas.

7.3. Regras artificiais inseridas

A fim de melhor verificar a eficiência de um e outro algoritmo *Apriori*, algumas regras de relacionamento foram inseridas artificialmente, que foram as seguintes:

1.O colégio Aplicação foi direcionado para CR médio, curso de Engenharia e salário alto;
2.O CMRJ foi direcionado para CR médio, curso de Direito e salário médio;
3.O Pedro II foi direcionado para CR médio, curso de Direito e salário alto;
4.O São José foi direcionado para CR alto, curso de Matemática e salário médio
5.As mulheres foram direcionadas para o colégio Aplicação; CR's alto e médio; Engenharia e salário baixo;
6.Os homens foram direcionados para o CMRJ e o colégio Pedro II; CR alto; cursos de Engenharia e Direito; salário alto;

Deve-se observar que no presente banco de dados, as regras supracitadas foram escolhidas para serem inseridas de modo aleatório, fazendo-se variar os graus de suporte e confiança existentes para cada uma delas.

CAPÍTULO VIII

8. ANÁLISE DOS RESULTADOS OBTIDOS NO ESTUDO DE CASO DOS ALUNOS DA UERJ

Face estudo realizado sobre os resultados obtidos, não resta dúvidas que em banco de dados que possuem dados numéricos o algoritmo *Apriori Fuzzy*, se mostra muito mais eficiente na descoberta de regras de associação.

As regras de associação envolvendo variáveis lingüísticas de campos numéricos são as que apresentam suporte mais elevado.

Ao se realizar as pesquisas de relacionamento utilizando-se o cenário 2, o algoritmo *Apriori* tradicional mostrou-se completamente ineficiente, enquanto o *Apriori Fuzzy* determinou algumas regras de associação.

Utilizando como objeto de estudo repositório de dados georreferenciado dos alunos da UERJ, a ferramenta de extração *APRIORI FUZZY* se mostrou muito mais eficiente na coleta de regras de associação envolvendo variáveis literais e quantitativas e envolvendo somente variáveis quantitativas.

Por fim, deve-se observar que a análise dos resultados obtidos no algoritmo *APRIORI FUZZY* deve ser feita de modo mais cuidadoso do que no algoritmo APRIORI TRADICIONAL.

Com a finalidade de ilustrar citado cuidado, observemos a seguinte regra de associação típica do *APRIORI FUZZY.*

Salário: Alto - 37,2% /// Sexo: F - 24,1%

Esta regra de associação não significa que de cada 1000 alunos, 372 (37,2% de 1.000) se enquadrem no conceito “Salário: Alto”, uma vez que o valor 372 neste caso

significa o somatório dos índices de pertinências dos valores reais da variável Salário com o conceito “Alto”.

Ou seja, a análise posterior com fins de extração de conhecimento da relação apresentada deverá considerar limites de valores para salário alto.

Ex.: ao observar que a regra “Salário: Alto - 37,2% /// Sexo: F - 24,1%” tem forte confiança (64,8% = 241/372 * 100%), um rede de comércio varejista pode oferecer às suas clientes convenio determinada loja de luxo, cuja carteira de clientes tem rendimentos superiores a R$ 10.000, mensais.

CAPÍTULO IX

9. BIBLIOGRAFIA

- R. Agrawal, T. Imielinski, A. Swami, "*Mining association rules between sets of items in large databases*," in Proc. of the ACM SIGMOD Int'l Conf. on Management of Data, Washington, D.C., USA, 1993, pp. 207-216.

- D. S´anchez, J. R. S´anchez, J. M. Serrano, and M. A. Vila. Reglas de asociacion sobre dominios imprecisos aplicando relaciones difusas de similitud (in spanish). In *Procs. Of ESTYLF'2004*, 2004.

- G. Dong and J. Li, "*Interestingness of Discovered Association Rules in Terms of Neighborhood-based Unexpectedness*," in Proc. of the Second Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'98), Melbourne, Austrália, 1998

- H. Blockeel and M. Sebag, "*Scalability and efficiency in multi-relational data mining*," *ACM SIGKDD Explorations Newsletter*, vol. 5, pp. 17-30, 2003.

- KDD-Cup 2000 Organizers' Report: Peeling the Onion. SIGKDD Exploration 2(2):86–93.

- Kohavi, R., Bradley, C.E., Frasca, B., Mason, L., and Zheng, Z. (2000).

- LANDEN, Mauro. Gestão do conhecimento organizacional e tecnologia de informação no suporte à decisão: o planejamento de um data *warehouse* de indicadores sociais e pobreza. Dissertação de Mestrado> Rio de Janeiro: Universidade Federal do Estado do Rio de Janeiro, 2000.

- L. Deshape and L. Raedt, "*Mining association rules in multiple relations*," in Proc. of the 7th Intl. Workshop on Inductive Logic Programming, Prague, Czech Republic, 1997, pp. 125-132.

- S. o. Džeroski, "*Multi-relational data mining: an introduction*," *ACM SIGKDD Explorations Newsletter*, vol. 5, pp. 1 - 16, 2003.

-V. Aranda, J. Calero, G. Delgado, S'anchez D., Serrano J.M., and Vila M.A. Flexible land classification for olive cultivation using user knowledge. In *Proceedings of 1st. Int. ICSC Conf. On Neuro-Fuzzy Technologies(NF'2002)*, 2002.

ANEXO I

1. Algoritmo *Apriori Fuzzy* aplicado

1.1. Algoritmo *Apriori Fuzzy* aplicado no cenário 1

Aplicando-se o algoritmo *Apriori Fuzzy* obtem-se os seguintes resultados, no cenário 1:

1. Itemsets-frequentes-1

Sexo: F - 54,7%
Sexo: M - 45,2%
Colégio anterior: Aplicação - 27,8%
Colégio anterior: CMRJ - 26,5%
Colégio anterior: Pedro II - 23,9%
Colégio anterior: São José - 21,7%
Curso Graduação: Direito - 27,3%
Curso Graduação: Engenharia - 26,9%
Curso Graduação: Matemática - 19,5%
Curso Graduação: Medicina - 26%
CR: Alto - 26,2%
CR: Baixo - 24,7%
CR: Médio - 35,2%
Salário: Alto - 37,2%
Salário: Baixo - 13,3%
Salário: Médio – 19,9%

2. Itemsets-frequentes-2

Sexo: F; Colégio anterior: Aplicação - 17,8%
Sexo: F; Colégio anterior: CMRJ - 13,9%
Sexo: F; Colégio anterior: Pedro II - 11,7%
Sexo: F; Colégio anterior: São José - 11,3%
Sexo: F; Curso Graduação: Direito - 13,4%
Sexo: F; Curso Graduação: Engenharia - 19,1%
Sexo: F; Curso Graduação: Matemática - 9,1%
Sexo: F; Curso Graduação: Medicina - 13%
Sexo: F; CR: Alto - 13,9%
Sexo: F; CR: Baixo - 13,4%
Sexo: F; CR: Médio - 20,7%
Sexo: F; Salário: Alto - 24,1%
Sexo: F; Salário: Baixo - 6,7%
Sexo: F; Salário: Médio - 6,8%
Sexo: M; Colégio anterior: Aplicação - 10%
Sexo: M; Colégio anterior: CMRJ - 12,6%
Sexo: M; Colégio anterior: Pedro II - 12,1%
Sexo: M; Colégio anterior: São José - 10,4%
Sexo: M; Curso Graduação: Direito - 13,9%
Sexo: M; Curso Graduação: Engenharia - 7,8%
Sexo: M; Curso Graduação: Matemática - 10,4%
Sexo: M; Curso Graduação: Medicina - 13%
Sexo: M; CR: Alto - 12,2%
Sexo: M; CR: Baixo - 11,3%
Sexo: M; CR: Médio - 14,5%
Sexo: M; Salário: Alto - 13%

Sexo: M; Salário: Baixo - 6,5%

Sexo: M; Salário: Médio - 13%

Colégio anterior: Aplicação; Curso Graduação: Engenharia - 11,7%

Colégio anterior: Aplicação; Curso Graduação: Medicina - 8,6%

Colégio anterior: Aplicação; CR: Alto - 5,8%

Colégio anterior: Aplicação; CR: Baixo - 7,9%

Colégio anterior: Aplicação; CR: Médio - 11%

Colégio anterior: Aplicação; Salário: Alto - 10,5%

Colégio anterior: Aplicação; Salário: Médio - 5,1%

Colégio anterior: CMRJ; Curso Graduação: Direito - 9,1%

Colégio anterior: CMRJ; Curso Graduação: Engenharia - 5,2%

Colégio anterior: CMRJ; Curso Graduação: Medicina - 7,8%

Colégio anterior: CMRJ; CR: Alto - 5,3%

Colégio anterior: CMRJ; CR: Baixo - 8,4%

Colégio anterior: CMRJ; CR: Médio - 9,5%

Colégio anterior: CMRJ; Salário: Alto - 10,4%

Colégio anterior: CMRJ; Salário: Médio - 6,5%

Colégio anterior: Pedro II; Curso Graduação: Direito - 8,6%

Colégio anterior: Pedro II; Curso Graduação: Engenharia - 7,3%

Colégio anterior: Pedro II; CR: Alto - 6,7%

Colégio anterior: Pedro II; CR: Médio - 8,4%

Colégio anterior: Pedro II; Salário: Alto - 8,1%

Colégio anterior: São José; Curso Graduação: Matemática - 8,2%

Colégio anterior: São José; Curso Graduação: Medicina - 6%

Colégio anterior: São José; CR: Alto - 8,1%

Colégio anterior: São José; CR: Médio - 6,2%

Colégio anterior: São José; Salário: Alto - 8%

Curso Graduação: Direito; CR: Alto - 6,5%

Curso Graduação: Direito; CR: Baixo - 6,9%

Curso Graduação: Direito; CR: Médio - 10,9%

Curso Graduação: Direito; Salário: Alto - 10,2%

Curso Graduação: Direito; Salário: Médio - 5,9%

Curso Graduação: Engenharia; CR: Alto - 7,3%

Curso Graduação: Engenharia; CR: Baixo - 6,9%

Curso Graduação: Engenharia; CR: Médio - 8,2%

Curso Graduação: Engenharia; Salário: Alto - 9,7%

Curso Graduação: Matemática; CR: Alto - 5,5%

Curso Graduação: Matemática; CR: Baixo - 5,5%

Curso Graduação: Matemática; CR: Médio - 5,5%

Curso Graduação: Matemática; Salário: Alto - 7,5%

Curso Graduação: Medicina; CR: Alto - 6,7%

Curso Graduação: Medicina; CR: Baixo - 5,3%

Curso Graduação: Medicina; CR: Médio - 10,4%

Curso Graduação: Medicina; Salário: Alto - 9,6%

Curso Graduação: Medicina; Salário: Baixo - 5,7%

Curso Graduação: Medicina; Salário: Médio - 5,1%

CR: Alto; Salário: Alto - 8,9%

CR: Baixo; Salário: Alto - 9,9%

CR: Baixo; Salário: Médio - 5,4%

CR: Médio; Salário: Alto - 13,6%

CR: Médio; Salário: Baixo - 6,1%

CR: Médio; Salário: Médio - 7%

3. Itemsets-frequentes-3

Sexo: F; Colégio anterior: Aplicação; Curso Graduação: Engenharia - 9,5%

Sexo: F; Colégio anterior: Aplicação; CR: Médio - 6,9 %

Sexo: F; Colégio anterior: Aplicação; Salário: Alto - 8,1 %

Sexo: F; Colégio anterior: CMRJ; CR: Médio - 5,3 %

Sexo: F; Colégio anterior: CMRJ; Salário: Alto - 7,2 %

Sexo: F; Colégio anterior: Pedro II; Curso Graduação: Direito - 5,2%

Sexo: F; Curso Graduação: Direito; CR: Médio - 5,5 %

Sexo: F; Curso Graduação: Direito; Salário: Alto - 5,9 %

Sexo: F; Curso Graduação: Engenharia; CR: Baixo - 5,2 %

Sexo: F; Curso Graduação: Engenharia; CR: Médio - 6,5 %

Sexo: F; Curso Graduação: Engenharia; Salário: Alto - 8,2 %

Sexo: F; Curso Graduação: Matemática; Salário: Alto - 5,3 %

Sexo: F; Curso Graduação: Medicina; CR: Médio - 5,6 %

Sexo: F; CR: Alto; Salário: Alto - 6,6 %

Sexo: F; CR: Baixo; Salário: Alto - 5,3 %

Sexo: F; CR: Médio; Salário: Alto - 8,7 %

Sexo: M; Colégio anterior: CMRJ; Curso Graduação: Direito - 5,2%

Sexo: M; Curso Graduação: Direito; CR: Médio - 5,3 %

Colégio anterior: Aplicação; CR: Médio; Salário: Alto - 5,1 %

4. Relacionamentos tipo regras de associação com 1 antecedente e 2 conseqüentes

Colégio anterior: Aplicação - 27,8% /// Sexo: F - 17,8%

Colégio anterior: CMRJ - 26,5% /// Sexo: F - 13,9%

Colégio anterior: Pedro II - 23,9% /// Sexo: M - 12,1%

Colégio anterior: São José - 21,7% /// Sexo: F - 11,3%

Curso Graduação: Direito - 27,3% /// Sexo: M - 13,9%

Curso Graduação: Engenharia - 26,9% /// Sexo: F - 19,1%

Curso Graduação: Matemática - 19,5% /// Sexo: M - 10,4%

CR: Alto - 26,2% /// Sexo: F - 13,9%

CR: Baixo - 24,7% /// Sexo: F - 13,4%

CR: Médio - 35,2% /// Sexo: F - 20,7%

Salário: Alto - 37,2% /// Sexo: F - 24,1%

Salário: Baixo - 13,3% /// Sexo: F - 6,7%

Salário: Médio - 19,9% /// Sexo: M – 13%

5. Relacionamentos tipo regras de associação com 1 antecedente e 3 conseqüentes

- Não foi determinado nenhum relacionamento deste tipo

6. Relacionamentos tipo regras de associação com 2 antecedentes e 3 conseqüentes

Sexo: F; Colégio anterior: Aplicação - 17,8% /// Curso Graduação: Engenharia - 9,5%

Sexo: F; Colégio anterior: CMRJ - 13,9% /// Salário: Alto - 7,2%

Sexo: F; Curso Graduação: Matemática - 9,1% /// Salário: Alto - 5,3%

Colégio anterior: Aplicação; Curso Graduação: Engenharia - 11,7% /// Sexo: F - 9,5%

Colégio anterior: Aplicação; CR: Médio - 11% /// Sexo: F - 6,9%

Colégio anterior: Aplicação; Salário: Alto - 10,5% /// Sexo: F - 8,1%

Colégio anterior: CMRJ; Curso Graduação: Direito - 9,1% /// Sexo: M - 5,2%

Colégio anterior: CMRJ; CR: Médio - 9,5% /// Sexo: F - 5,3%

Colégio anterior: CMRJ; Salário: Alto - 10,4% /// Sexo: F - 7,2%

Colégio anterior: Pedro II; Curso Graduação: Direito - 8,6% /// Sexo: F - 5,2%

Curso Graduação: Direito; CR: Médio - 10,9% /// Sexo: F - 5,5%

Curso Graduação: Direito; Salário: Alto - 10,2% /// Sexo: F - 5,9%

Curso Graduação: Engenharia; CR: Baixo - 6,9% /// Sexo: F - 5,2%

Curso Graduação: Engenharia; CR: Médio - 8,2% /// Sexo: F - 6,5%

Curso Graduação: Engenharia; Salário: Alto - 9,7% /// Sexo: F - 8,2%

Curso Graduação: Matemática; Salário: Alto - 7,5% /// Sexo: F - 5,3%

Curso Graduação: Medicina; CR: Médio - 10,4% /// Sexo: F - 5,6%

CR: Alto; Salário: Alto - 8,9% /// Sexo: F - 6,6%

CR: Baixo; Salário: Alto - 9,9% /// Sexo: F - 5,3%

CR: Médio; Salário: Alto - 13,6% /// Sexo: F - 8,7%

1.2. Algoritmo *Apriori Fuzzy* aplicado no cenário 2

1. Itemsets-frequentes-1

Sexo: F - 54,7%

Sexo: M - 45,2%

Colégio anterior: Aplicação - 27,8%

Colégio anterior: CMRJ - 26,5%

Colégio anterior: Pedro II - 23,9%

Colégio anterior: São José - 21,7%

Curso Graduação: Direito - 27,3%

Curso Graduação: Engenharia - 26,9%

Curso Graduação: Medicina - 26%

CR: Alto - 26,2%

CR: Baixo - 24,7%

CR: Médio - 35,2%

Salário: Alto - 37,2%

2. Itemsets-frequentes-2

Sexo: F; CR: Médio - 20,7%

Sexo: F; Salário: Alto - 24,1%

3. Itemsets-frequentes-3

Não foi determinado nenhum relacionamento deste tipo

4. Relacionamentos tipo regras de associação com 1 antecedente e 2 conseqüentes
CR: Médio - 35,2% /// Sexo: F - 20,7%
Salário: Alto - 37,2% /// Sexo: F - 24,1%

5. Relacionamentos tipo regras de associação com 1 antecedente e 3 conseqüentes

- Não foi determinado nenhum relacionamento deste tipo

6. Relacionamentos tipo regras de associação com 2 antecedentes e 3 conseqüentes

- Não foi determinado nenhum relacionamento deste tipo

2. Algoritmo *Apriori Tradicional* aplicado

2.1. Algoritmo *Apriori Tradicional* aplicado no cenário 1

Aplicando-se o algoritmo *Apriori Tradicional* obtêm-se os seguintes resultados:

1. Itemsets-frequentes-1

Sexo: F - 54,7%
Sexo: M - 45,2%
Colégio anterior: Aplicação - 27,8%

Colégio anterior: CMRJ - 26,5%

Colégio anterior: Pedro II - 23,9%

Colégio anterior: São José - 21,7%

Curso Graduação: Direito - 27,3%

Curso Graduação: Engenharia - 26,9%

Curso Graduação: Matemática - 19,5%

Curso Graduação: Medicina - 26%

2. Itemsets-frequentes-2

Sexo: F; Colégio anterior: Aplicação - 17,8%

Sexo: F; Colégio anterior: CMRJ - 13,9%

Sexo: F; Colégio anterior: Pedro II - 11,7%

Sexo: F; Colégio anterior: São José - 11,3%

Sexo: F; Curso Graduação: Direito - 13,4%

Sexo: F; Curso Graduação: Engenharia - 19,1%

Sexo: F; Curso Graduação: Matemática - 9,1%

Sexo: F; Curso Graduação: Medicina - 13%

Sexo: M; Colégio anterior: Aplicação - 10%

Sexo: M; Colégio anterior: CMRJ - 12,6%

Sexo: M; Colégio anterior: Pedro II - 12,1%

Sexo: M; Colégio anterior: São José - 10,4%

Sexo: M; Curso Graduação: Direito - 13,9%

Sexo: M; Curso Graduação: Engenharia - 7,8%

Sexo: M; Curso Graduação: Matemática - 10,4%

Sexo: M; Curso Graduação: Medicina - 13%

Colégio anterior: Aplicação; Curso Graduação: Engenharia - 11,7%

Colégio anterior: Aplicação; Curso Graduação: Medicina - 8,6%

Colégio anterior: CMRJ; Curso Graduação: Direito - 9,1%

Colégio anterior: CMRJ; Curso Graduação: Engenharia - 5,2%
Colégio anterior: CMRJ; Curso Graduação: Medicina - 7,8%
Colégio anterior: Pedro II; Curso Graduação: Direito - 8,6%
Colégio anterior: Pedro II; Curso Graduação: Engenharia - 7,3%
Colégio anterior: São José; Curso Graduação: Matemática - 8,2%
Colégio anterior: São José; Curso Graduação: Medicina - 6%

3. Itemsets-frequentes-3

Sexo: F; Colégio anterior: Aplicação; Curso Graduação: Engenharia - 9,5%
Sexo: F; Colégio anterior: Pedro II; Curso Graduação: Direito - 5,2%
Sexo: M; Colégio anterior: CMRJ; Curso Graduação: Direito - 5,2%

4. Relacionamentos tipo regras de associação com 1 antecedente e 2 conseqüentes

Colégio anterior: Aplicação - 27,8% /// Sexo: F - 17,8%
Colégio anterior: CMRJ - 26,5% /// Sexo: F - 13,9%
Colégio anterior: Pedro II - 23,9% /// Sexo: M - 12,1%
Colégio anterior: São José - 21,7% /// Sexo: F - 11,3%
Curso Graduação: Direito - 27,3% /// Sexo: M - 13,9%
Curso Graduação: Engenharia - 26,9% /// Sexo: F - 19,1%
Curso Graduação: Matemática - 19,5% /// Sexo: M - 10,4%

5. Relacionamentos tipo regras de associação com 1 antecedente e 3 conseqüentes

-Não foi determinado nenhum relacionamento deste tipo

1. Relacionamentos tipo regras de associação com 2 antecedentes e 3 conseqüentes

Sexo: F; Colégio anterior: Aplicação - 17,8% /// Curso Graduação: Engenharia - 9,5%

Colégio anterior: Aplicação; Curso Graduação: Engenharia - 11,7% /// Sexo: F - 9,5%

Colégio anterior: CMRJ; Curso Graduação: Direito - 9,1% /// Sexo: M - 5,2%

Colégio anterior: Pedro II; Curso Graduação: Direito - 8,6% /// Sexo: F - 5,2%

2.2. Algoritmo *Apriori Tradicional* aplicado no cenário 2

1. Itemsets-frequentes-1

Sexo: F - 54,7%

Sexo: M - 45,2%

Colégio anterior: Aplicação - 27,8%

Colégio anterior: CMRJ - 26,5%

Colégio anterior: Pedro II - 23,9%

Colégio anterior: São José - 21,7%

Curso Graduação: Direito - 27,3%

Curso Graduação: Engenharia - 26,9%

Curso Graduação: Medicina - 26%

2. Itemsets-frequentes-2

- Não foi determinado nenhum relacionamento deste tipo

3. Itemsets-frequentes-3

- Não foi determinado nenhum relacionamento deste tipo

4. Relacionamentos tipo regras de associação com 1 antecedente e 2 conseqüentes

- Não foi determinado nenhum relacionamento deste tipo

5. Relacionamentos tipo regras de associação com 1 antecedente e 3 conseqüentes e com 2 antecedentes e 3 conseqüentes

- Não foi determinado nenhum relacionamento deste tipo

ANEXO 2

Código Fonte dos algoritmos *Apriori Fuzzy* e *Apriori* Tradicional desenvolvidos em Visual Basic 6.0

```
Option Explicit
Dim controle1 As Boolean
Dim controle2 As Boolean
Dim controle3 As Boolean

Dim agenda As Database

Dim soma_pertinencias As Double
Dim coef_a As Double
Dim coef_b As Double
Dim coef_c As Double
Dim contador2 As Double
Dim v_ini As Double
Dim limite2 As Double

Dim a As Integer
Dim b As Integer
Dim c As Integer
Dim campoigual(2) As Integer
Dim contador As Integer
Dim d As Integer
Dim limite As Integer
Dim posicao As Integer
Dim soma As Integer
Dim total As Integer
```

```
Dim tabelas_agenda As Recordset
Dim tabelas_apriori As Recordset
Dim tabelas_contagem As Recordset
Dim tabelas_contagem_1 As Recordset
Dim tabelas_contagem_2 As Recordset
Dim tabelas_contagem_3 As Recordset
Dim tabelas_pesquisa As Recordset

Dim arquivo As String
Dim criterio_pesquisa As String
Dim criterio_pesquisa_1 As String
Dim criterio_pesquisa_2 As String
Dim kz As String
Dim kk As String
Dim nome As String
Dim tabelas_temp(50) As String
Dim tabelas() As String
Dim tabela As String
Dim valor(3) As String

Dim itemset1() As apriori_1
Dim itemset2() As apriori_2
Dim itemset3() As apriori_3

Dim fs
Dim aaa
```

Private Sub Command1_Click()

```
    'Botão do painel que aciona o conjunto de rotinas "apura_apriori"
    Call apura_apriori
```

End Sub

Private Sub Form_Load()

'Aqui tem início o programa com a rotina inicial "início" e o algoritmo "apura_apriori"
Call inicio
Call apura_apriori

End Sub

Sub apura_apriori()

Call inicio_bd	*'Inicializa o banco de dados*
Call limpa_resultados	*'Limpa os arquivos onde ficarão armazenados os itemsets freqüentes determinados, bem como os relacionamentos encontrados*
Call contagem_itemset_1	*'Faz o contagem dos itemsets-frequentes-1*
Call contagem_itemset_2	'*Faz o contagem dos itemsets-frequentes-2*
Call contagem_itemset_3	*'Faz o contagem dos itemsets-frequentes-3*
Call verifica_1_para_2	*'Faz o determinação dos relacionamentos com 1 antecedente e 1 conseqüente*
Call verifica_1_para_3	*'Faz o determinação dos relacionamentos com 1 antecedente e 2 conseqüente*
Call verifica_2_para_3	*'Faz o determinação dos relacionamentos com 2 antecedentes e 1 conseqüente*
Call fimbd	*'Inicializa o banco de dados*

End Sub

Sub inicio()

```
kk = IIf(Right(App.Path, 1) = "\", App.Path, App.Path & "\")
kz = kk
If Right(kk, 7) = "fontes\" Or Left(Right(kk, 1 + Len(Dir(App.Path, vbDirectory))), 4) = "2009" Then
   kk = Left(kk, Len(kk) - 1 - Len(Dir(App.Path, vbDirectory)))
End If
arquivo = "dados.mdb"
```

```
Set fs = CreateObject("Scripting.FileSystemObject")
For a = 0 To 100
   Combo1.AddItem Format(a, "00")
   Combo2.AddItem Format(a, "00")
Next a
Combo1.Text = "50"
Combo2.Text = "5"
```

End Sub

Sub inicio_bd()

```
Set agenda = OpenDatabase(kk & arquivo)
Set tabelas_agenda = agenda.OpenRecordset("Dados", dbOpenDynaset)
tabelas_agenda.MoveFirst
tabelas_agenda.MoveNext
tabelas_agenda.MoveFirst
total = tabelas_agenda.RecordCount
Label6.Caption = "Total: " & total
Set tabelas_apriori = agenda.OpenRecordset("Fuzzy", dbOpenDynaset)
For a = 1 To 50
   tabelas_temp(b) = ""
Next a
b = 1
For a = 0 To tabelas_agenda.Fields.Count - 1
   tabelas_apriori.FindFirst "Campo = '" & tabelas_agenda.Fields(a).Name & "'"
   If Not tabelas_apriori.NoMatch Then
      If tabelas_apriori.Fields("Apriori") Then
         If tabelas_apriori.Fields("Fuzzy") Then
            tabelas_temp(b) = tabelas_agenda.Fields(a).Name & "_Fuzzy"
         Else
            tabelas_temp(b) = tabelas_agenda.Fields(a).Name
         End If
         b = b + 1
      End If
```

```
    End If
  Next
  limite = b - 1
  ReDim Preserve tabelas(limite) As String
  For a = 1 To limite
    If Form2.fuzzy Then
      tabelas(a) = tabelas_temp(a)
    Else
      tabelas(a) = Replace(tabelas_temp(a), "_Fuzzy", "")
    End If
  Next a
  ReDim Preserve itemset1(1000)
```

End Sub

Sub contagem_itemset_1()

```
  'Passo 1: verifica os itens distintos em cada campo e faz sua contagem
  List1.Clear
  For b = 1 To 1000
    itemset1(b).campo(1) = 0
    itemset1(b).valor(1) = ""
  Next b

  For a = 1 To limite
    If Right(tabelas(a), 5) = "Fuzzy" Then
      tabela = Left(tabelas(a), Len(tabelas(a)) - 6)
      Set tabelas_contagem = agenda.OpenRecordset("Select * FROM Fuzzy WHERE campo='" & tabela & "';", dbOpenDynaset)
      If Not tabelas_contagem.EOF Or Not tabelas_contagem.BOF Then tabelas_contagem.MoveFirst
      While Not tabelas_contagem.EOF
        contador2 = contagem_fuzzy(tabela, tabelas_contagem.Fields("Conceito"), tabelas_contagem.Fields("Ponto baixo 1"), tabelas_contagem.Fields("Ponto baixo 2"), tabelas_contagem.Fields("Ponto alto"))
        If contador2 > (Combo2.Text * tabelas_agenda.RecordCount) / 100 Then
```

```
            itemset1(List1.ListCount + 1).campo(1) = a
            itemset1(List1.ListCount + 1).valor(1) = tabelas_contagem.Fields("Conceito")
            itemset1(List1.ListCount + 1).contador = contador2
            itemset1(List1.ListCount + 1).cont_percentual = Int(1000 * contador2 / total) / 10
            posicao = List1.ListCount + 1
            List1.AddItem Left(tabelas(a), Len(tabelas(a)) - 6) & ": " & itemset1(posicao).valor(1) & " - " &
Int(1000 * contador2 / total) / 10 & "%"
            Call adiciona("contagem_itemset_1", Left(tabelas(a), Len(tabelas(a)) - 6) & ": " &
itemset1(posicao).valor(1) & " - " & Int(1000 * contador2 / total) / 10 & "%")
          End If
          tabelas_contagem.MoveNext
        Wend
      Else
        criterio_pesquisa = "Select DISTINCT [" & tabelas(a) & "] FROM Dados;"
        Set tabelas_contagem_1 = agenda.OpenRecordset(criterio_pesquisa, dbOpenDynaset)
        If Not tabelas_contagem_1.EOF Or Not tabelas_contagem_1.BOF Then
tabelas_contagem_1.MoveFirst
        While Not tabelas_contagem_1.EOF
          criterio_pesquisa_1 = "SELECT COUNT([" & tabelas(a) & "]) AS total FROM Dados WHERE ["
& tabelas(a) & "] = '" & tabelas_contagem_1.Fields(0) & "';"
          criterio_pesquisa_2 = "SELECT COUNT([" & tabelas(a) & "]) AS total FROM Dados WHERE ["
& tabelas(a) & "] =  " & tabelas_contagem_1.Fields(0) & " ;"
          criterio_pesquisa = IIf(tabelas_agenda.Fields(tabelas(a)).Type <> 10, criterio_pesquisa_2,
criterio_pesquisa_1)
          Set tabelas_contagem_2 = agenda.OpenRecordset(criterio_pesquisa, dbOpenDynaset)
          If tabelas_contagem_2.Fields("Total") > (Combo2.Text * tabelas_agenda.RecordCount) / 100 Then
            itemset1(List1.ListCount + 1).campo(1) = a
            itemset1(List1.ListCount + 1).valor(1) = tabelas_contagem_1.Fields(0)
            itemset1(List1.ListCount + 1).contador = tabelas_contagem_2.Fields("Total")
            itemset1(List1.ListCount + 1).cont_percentual = Int(1000 * tabelas_contagem_2.Fields("Total") /
total) / 10
            posicao = List1.ListCount + 1
            List1.AddItem tabelas(a) & ": " & itemset1(posicao).valor(1) & " - " & Int(1000 *
tabelas_contagem_2.Fields("Total") / total) / 10 & "%"
```

```
            Call adiciona("contagem_itemset_1", tabelas(a) & ": " & itemset1(posicao).valor(1) & " - " &
Int(1000 * tabelas_contagem_2.Fields("Total") / total) / 10 & "%")
          End If
          tabelas_contagem_1.MoveNext
        Wend
      End If
    Next a
    ReDim Preserve itemset1(List1.ListCount)
```

End Sub

Sub contagem_itemset_2()

```
    ReDim Preserve itemset2(1000)

    List3.Clear
    'Passo 1: realizar as combinações dos itemsets(1)
    For a = 1 To UBound(itemset1)
      For b = a To UBound(itemset1)
        If itemset1(a).campo(1) < itemset1(b).campo(1) Then
          Call contagem_efetiva_2(itemset1(a).campo(1), itemset1(a).valor(1), itemset1(b).campo(1),
itemset1(b).valor(1), itemset1(a).campo(1), itemset1(b).campo(1))
        End If
      Next b
    Next
    List2.Clear
    For a = 0 To List3.ListCount - 1
      List2.AddItem List3.List(a)
    Next
    List3.Clear
    ReDim Preserve itemset2(List2.ListCount)
```

End Sub

Sub contagem_itemset_3()

```
ReDim Preserve itemset3(1000)

List4.Clear
List3.Clear
'Passo 1: realizar as combinações dos itemsets(1)
For a = 1 To UBound(itemset2)
   For b = 1 To UBound(itemset1)
      If itemset2(a).campo(1) <> itemset1(b).campo(1) And itemset2(a).campo(2) <> itemset1(b).campo(1)
Then
         If depuracao_3(itemset2(a).campo(1), itemset2(a).valor(1), itemset2(a).campo(2),
itemset2(a).valor(2), itemset1(b).campo(1), itemset1(b).valor(1)) Then
            Call contagem_efetiva_3(itemset2(a).campo(1), itemset2(a).valor(1), itemset2(a).campo(2),
itemset2(a).valor(2), itemset1(b).campo(1), itemset1(b).valor(1), itemset2(a).campo(1), itemset2(a).campo(2),
itemset1(b).campo(1))
         End If
      End If
   Next b
Next a
For a = 0 To List4.ListCount - 1
   List3.AddItem List4.List(a)
Next
List4.Clear
ReDim Preserve itemset3(List3.ListCount)
```

End Sub

Sub contagem_efetiva_2(ByVal tabela1 As Integer, ByVal valor1 As String, ByVal tabela2 As Integer, ByVal valor2 As String, ByVal a As Integer, ByVal b As Integer)

```
'Faz a contagem dos itemsets candidatos 2 com a finalidade de determinar os itemsets freqüentes 2.
'Caso entre os campos pesquisados, haja algum campo quantitativo, encaminha essa tarefa para a
'rotina "contagem_efetiva_fuzzy_2"

If Right(tabelas(tabela1), 5) = "Fuzzy" Or Right(tabelas(tabela2), 5) = "Fuzzy" Then
   Call contagem_efetiva_fuzzy_2(tabela1, valor1, tabela2, valor2, a, b)
Else
```

```
        valor(1) = IIf(tabelas_agenda.Fields(tabela1).Type <> 10, valor1, "'" & valor1 & "'")
        valor(2) = IIf(tabelas_agenda.Fields(tabela2).Type <> 10, valor2, "'" & valor2 & "'")
        criterio_pesquisa = "SELECT COUNT([" & tabelas(tabela1) & "]) AS total FROM Dados WHERE [" &
tabelas(tabela1) & "] = " & valor(1) & " AND [" & tabelas(tabela2) & "] = " & valor(2) & ";"
        'MsgBox criterio_pesquisa
        Set tabelas_contagem_2 = agenda.OpenRecordset(criterio_pesquisa, dbOpenDynaset)
        If tabelas_contagem_2.Fields("Total") > (Combo2.Text * tabelas_agenda.RecordCount) / 100 Then
          valor(1) = Replacevb5(valor(1), "'", "")
          valor(2) = Replacevb5(valor(2), "'", "")
          List3.AddItem tabelas(tabela1) & ": " & valor(1) & "; " & tabelas(tabela2) & ": " & valor(2) & " - " &
Int(1000 * tabelas_contagem_2.Fields("Total") / total) / 10 & "%"
          Call adiciona("contagem_itemset_2", tabelas(tabela1) & ": " & valor(1) & "; " & tabelas(tabela2) & ":
" & valor(2) & " - " & Int(1000 * tabelas_contagem_2.Fields("Total") / total) / 10 & "%")
          If a < b Then
            itemset2(List3.ListCount).campo(1) = a
            itemset2(List3.ListCount).campo(2) = b
            itemset2(List3.ListCount).valor(1) = valor(1)
            itemset2(List3.ListCount).valor(2) = valor(2)
          Else
            itemset2(List3.ListCount).campo(1) = b
            itemset2(List3.ListCount).campo(2) = a
            itemset2(List3.ListCount).valor(1) = valor(2)
            itemset2(List3.ListCount).valor(2) = valor(1)
          End If
          itemset2(List3.ListCount).contador = tabelas_contagem_2.Fields("Total")
          itemset2(List3.ListCount).cont_percentual = Int(1000 * (tabelas_contagem_2.Fields("Total") / total)) /
10
        End If
      End If
```

End Sub

Sub contagem_efetiva_3(ByVal tabela1 As Integer, ByVal valor1 As String, ByVal tabela2 As Integer, ByVal valor2 As String, ByVal tabela3 As Integer, ByVal valor3 As String, ByVal a As Integer, ByVal b As Integer, ByVal c As Integer)

```
        'Faz a contagem dos itemsets candidatos 3 com a finalidade de determinar os itemsets freqüentes 3.
        'Caso entre os campos pesquisados, haja algum campo quantitativo, encaminha essa tarefa para a
        'rotina "contagem_efetiva_fuzzy_3"
    If Right(tabelas(tabela1), 5) = "Fuzzy" Or Right(tabelas(tabela2), 5) = "Fuzzy" Or Right(tabelas(tabela3), 5) = "Fuzzy" Then
        Call contagem_efetiva_fuzzy_3(tabela1, valor1, tabela2, valor2, tabela3, valor3, a, b, c)
    Else
        valor(1) = IIf(tabelas_agenda.Fields(tabela1).Type <> 10, valor1, "'" & valor1 & "'")
        valor(2) = IIf(tabelas_agenda.Fields(tabela2).Type <> 10, valor2, "'" & valor2 & "'")
        valor(3) = IIf(tabelas_agenda.Fields(tabela3).Type <> 10, valor3, "'" & valor3 & "'")
        criterio_pesquisa = "SELECT COUNT([" & tabelas(tabela1) & "]) AS total FROM Dados WHERE [" & tabelas(tabela1) & "] = " & valor(1) & " AND [" & tabelas(tabela2) & "] = " & valor(2) & " AND [" & tabelas(tabela3) & "] = " & valor(3) & ";"
        Set tabelas_contagem_3 = agenda.OpenRecordset(criterio_pesquisa, dbOpenDynaset)
        If tabelas_contagem_3.Fields("Total") > (Combo2.Text * tabelas_agenda.RecordCount) / 100 Then
            valor(1) = Replacevb5(valor(1), "'", "")
            valor(2) = Replacevb5(valor(2), "'", "")
            valor(3) = Replacevb5(valor(3), "'", "")
            nome = tabelas(tabela1) & ": " & valor(1) & "; " & tabelas(tabela2) & ": " & valor(2) & "; " & tabelas(tabela3) & ": " & valor(3) & " - " & Int(1000 * tabelas_contagem_3.Fields("Total") / total) / 10 & "%"
            List4.AddItem nome
            Call adiciona("contagem_itemset_3", nome)
            Call preenche_3(a, b, c, valor(1), valor(2), valor(3))
            itemset3(List4.ListCount).contador = tabelas_contagem_3.Fields("Total")
            itemset3(List4.ListCount).cont_percentual = Int(1000 * (tabelas_contagem_3.Fields("Total") / total)) / 10
        End If
    End If
    If List4.ListCount > 990 Then
        MsgBox "ERRO"
        End
    End If

End Sub
```

Sub verifica_1_para_2()

```
List5.Clear
For a = 1 To UBound(itemset1)
  For b = 1 To UBound(itemset2)
    If itemset1(a).campo(1) = itemset2(b).campo(1) And itemset1(a).valor(1) = itemset2(b).valor(1) Then
      If itemset2(b).contador * 100 / itemset1(a).contador > Val(Combo1.Text) Then
        nome = Replacevb5(tabelas(itemset1(a).campo(1)) & ": " & itemset1(a).valor(1) & " - " & itemset1(a).cont_percentual & "% /// " & tabelas(itemset2(b).campo(2)) & ": " & itemset2(b).valor(2) & " - " & itemset2(b).cont_percentual & "%", "_Fuzzy", "")
        List5.AddItem nome
        Call adiciona("verifica_1_para_2", nome)
      End If
    ElseIf itemset1(a).campo(1) = itemset2(b).campo(2) And itemset1(a).valor(1) = itemset2(b).valor(2) Then
      If itemset2(b).contador * 100 / itemset1(a).contador > Val(Combo1.Text) Then
        nome = Replacevb5(tabelas(itemset1(a).campo(1)) & ": " & itemset1(a).valor(1) & " - " & itemset1(a).cont_percentual & "% /// " & tabelas(itemset2(b).campo(1)) & ": " & itemset2(b).valor(1) & " - " & itemset2(b).cont_percentual & "%", "_Fuzzy", "")
        List5.AddItem nome
        Call adiciona("verifica_1_para_2", nome)
      End If
    End If
  Next b
Next a
```

End Sub

Sub verifica_1_para_3()

```
List6.Clear
For a = 1 To UBound(itemset1)
  For b = 1 To UBound(itemset3)
    If itemset1(a).campo(1) = itemset3(b).campo(1) And itemset1(a).valor(1) = itemset3(b).valor(1) Then
      If itemset3(b).contador * 100 / itemset1(a).contador > Val(Combo1.Text) Then
```

```
            nome = Replacevb5(tabelas(itemset1(a).campo(1)) & ": " & itemset1(a).valor(1) & " - " &
itemset1(a).cont_percentual & "% /// " & tabelas(itemset3(b).campo(2)) & ": " & itemset3(b).valor(2) & "; "
& tabelas(itemset3(b).campo(3)) & ": " & itemset3(b).valor(3) & " - " & itemset3(b).cont_percentual & "%",
"_Fuzzy", "")
            List6.AddItem nome
            Call adiciona("verifica_1_para_3", nome)
          End If
        ElseIf itemset1(a).campo(1) = itemset3(b).campo(2) And itemset1(a).valor(1) = itemset3(b).valor(2)
Then
          If itemset3(b).contador * 100 / itemset1(a).contador > Val(Combo1.Text) Then
            nome = Replacevb5(tabelas(itemset1(a).campo(1)) & ": " & itemset1(a).valor(1) & " - " &
itemset1(a).cont_percentual & "% /// " & tabelas(itemset3(b).campo(1)) & ": " & itemset3(b).valor(1) & "; "
& tabelas(itemset3(b).campo(3)) & ": " & itemset3(b).valor(3) & " - " & itemset3(b).cont_percentual & "%",
"_Fuzzy", "")
            List6.AddItem nome
            Call adiciona("verifica_1_para_3", nome)
          End If
        ElseIf itemset1(a).campo(1) = itemset3(b).campo(3) And itemset1(a).valor(1) = itemset3(b).valor(3)
Then
          If itemset3(b).contador * 100 / itemset1(a).contador > Val(Combo1.Text) Then
            nome = Replacevb5(tabelas(itemset1(a).campo(1)) & ": " & itemset1(a).valor(1) & " - " &
itemset1(a).cont_percentual & "% /// " & tabelas(itemset3(b).campo(1)) & ": " & itemset3(b).valor(1) & "; "
& tabelas(itemset3(b).campo(2)) & ": " & itemset3(b).valor(2) & " - " & itemset3(b).cont_percentual & "%",
"_Fuzzy", "")
            List6.AddItem nome
            Call adiciona("verifica_1_para_3", Replacevb5(tabelas(itemset1(a).campo(1)) & ": " &
itemset1(a).valor(1) & "-" & itemset1(a).contador & " /// " & tabelas(itemset3(b).campo(1)) & ": " &
itemset3(b).valor(1) & "; " & tabelas(itemset3(b).campo(2)) & ": " & itemset3(b).valor(2) & " - " &
itemset3(b).contador, "_Fuzzy", ""))
          End If
        End If
      Next b
    Next a
End Sub
```

Sub verifica_2_para_3()

```
   List7.Clear
   For a = 1 To UBound(itemset2)
      For b = 1 To UBound(itemset3)
         contador = 0
         For c = 1 To 3
            If itemset2(a).campo(1) = itemset3(b).campo(c) And itemset2(a).valor(1) = itemset3(b).valor(c)
Then
               contador = contador + 1
               campoigual(1) = c
            End If
            If itemset2(a).campo(2) = itemset3(b).campo(c) And itemset2(a).valor(2) = itemset3(b).valor(c)
Then
               contador = contador + 1
               campoigual(2) = c
            End If
         Next c
         If contador = 2 Then
            If itemset3(b).contador * 100 / itemset2(a).contador > Val(Combo1.Text) Then
               nome = tabelas(itemset2(a).campo(1)) & ": " & itemset2(a).valor(1)
               nome = nome & "; " & tabelas(itemset2(a).campo(2)) & ": " & itemset2(a).valor(2) & " - " &
itemset2(a).cont_percentual & "%"
               For d = 1 To 3
                  If d <> campoigual(1) And d <> campoigual(2) Then Exit For
               Next d
               nome = nome & " /// " & tabelas(itemset3(b).campo(d)) & ": " & itemset3(b).valor(d) & " - " &
itemset3(b).cont_percentual & "%"
               List7.AddItem Replacevb5(nome, "_Fuzzy", "")
               Call adiciona("verifica_2_para_3", Replacevb5(nome, "_Fuzzy", ""))
            End If
         End If
      Next b
   Next a
```

End Sub

Sub fimbd()

```
agenda.Close
Set agenda = Nothing
```

End Sub

Private Sub Form_Unload(Cancel As Integer)

```
    'Ao se fechar esse form, volta para tela inicial
If Dir(App.Path & "\" & App.EXEName & ".exe") <> "" Then
   Shell App.Path & "\" & App.EXEName & ".exe", vbNormalFocus
End If
End
```

End Sub

Private Sub Option1_Click()

'Através da função visivel, se escolhe o conjunto de relacionamentos que se deseja visualizar, nesse caso, visualize-se o relacionamentos 1 antecedente e 1 consequente

```
Call visivel(True, False, False)
```

End Sub

Private Sub Option2_Click()

'Através da função visivel, se escolhe o conjunto de relacionamentos que se deseja visualizar, nesse caso, visualize-se o relacionamentos 1 antecedente e 2 consequente

```
Call visivel(False, True, False)
```

End Sub

Private Sub Option3_Click()

'Através da função visivel, se escolhe o conjunto de relacionamentos que se deseja visualizar, nesse caso, visualize-se o relacionamentos 2 antecedente e 1 consequente

```
Call visivel(False, False, True)
```

End Sub

Function visivel(crit1 As Boolean, crit2 As Boolean, crit3 As Boolean)

```
Label7.Visible = crit1
Label8.Visible = crit2
Label9.Visible = crit3
List5.Visible = Label7.Visible
List6.Visible = Label8.Visible
List7.Visible = Label9.Visible
```

End Function

Function contagem_fuzzy(tabela As String, conceito As String, pb1 As Single, pb2 As Single, palto As Single) As Double

'Faz a contagem dos itemsets candidatos 1 (todo o conjunto de dados) com a finalidade de determinar os itemsets freqüentes 1, caso o campo pesquisado seja quantitativo

```
Set tabelas_contagem_1 = agenda.OpenRecordset("Select * FROM Dados;", dbOpenDynaset)
If Not tabelas_contagem_1.EOF Or Not tabelas_contagem_1.BOF Then tabelas_contagem_1.MoveFirst
soma_pertinencias = 0
While Not tabelas_contagem_1.EOF
   v_ini = tabelas_contagem_1.Fields(tabela)
   If conceito = "Alto" Then v_ini = (v_ini / (palto - pb1)) - 1
   If conceito = "Baixo" Then v_ini = (-1) * (v_ini / (pb2 - palto)) + 1
   If conceito = "Médio" Then v_ini = IIf(v_ini > palto, (pb2 - v_ini) / (pb2 - palto), v_ini / palto)
   If v_ini < 0 Then v_ini = 0
   If v_ini > 1 Then v_ini = 1
   tabelas_contagem_1.Edit
      tabelas_contagem_1.Fields(tabela & "_Fuzzy") = v_ini
   tabelas_contagem_1.Update
   soma_pertinencias = soma_pertinencias + tabelas_contagem_1.Fields(tabela & "_Fuzzy")
   tabelas_contagem_1.MoveNext
Wend
contagem_fuzzy = soma_pertinencias
```

End Function

Function contagem_efetiva_fuzzy_2(ByVal tabela1 As Integer, ByVal valor1 As String, ByVal tabela2 As Integer, ByVal valor2 As String, ByVal a As Integer, ByVal b As Integer)

```
    controle1 = False
    controle2 = False
    If Right(tabelas(tabela1), 5) = "Fuzzy" Then
        controle1 = True
        tabela = Left(tabelas(tabela1), Len(tabelas(tabela1)) - 6)
        Set tabelas_pesquisa = agenda.OpenRecordset("Select * FROM Fuzzy WHERE Campo = '" & tabela & "' AND Conceito = '" & valor1 & "';", dbOpenDynaset)
        Call contagem_fuzzy(tabela, valor1, tabelas_pesquisa.Fields("Ponto baixo 1"), tabelas_pesquisa.Fields("Ponto baixo 2"), tabelas_pesquisa.Fields("Ponto alto"))
    End If
    If Right(tabelas(tabela2), 5) = "Fuzzy" Then
        controle2 = True
        tabela = Left(tabelas(tabela2), Len(tabelas(tabela2)) - 6)
        Set tabelas_pesquisa = agenda.OpenRecordset("Select * FROM Fuzzy WHERE Campo = '" & tabela & "' AND Conceito = '" & valor2 & "';", dbOpenDynaset)
        Call contagem_fuzzy(tabela, valor2, tabelas_pesquisa.Fields("Ponto baixo 1"), tabelas_pesquisa.Fields("Ponto baixo 2"), tabelas_pesquisa.Fields("Ponto alto"))
    End If

    soma_pertinencias = 0

    If controle1 And controle2 Then
        Set tabelas_contagem_1 = agenda.OpenRecordset("Select * FROM Dados;", dbOpenDynaset)
        If Not tabelas_contagem_1.EOF Or Not tabelas_contagem_1.BOF Then tabelas_contagem_1.MoveFirst
        While Not tabelas_contagem_1.EOF
            soma_pertinencias = soma_pertinencias + tabelas_contagem_1.Fields(tabelas(tabela1)) * tabelas_contagem_1.Fields(tabelas(tabela2))
            tabelas_contagem_1.MoveNext
        Wend
    ElseIf controle1 Then
```

```
    Set tabelas_contagem_1 = agenda.OpenRecordset("Select * FROM Dados WHERE [" & tabelas(tabela2) & "] = '" & valor2 & "';", dbOpenDynaset)
    If Not tabelas_contagem_1.EOF Or Not tabelas_contagem_1.BOF Then tabelas_contagem_1.MoveFirst
    While Not tabelas_contagem_1.EOF
      soma_pertinencias = soma_pertinencias + tabelas_contagem_1.Fields(tabelas(tabela1))
      tabelas_contagem_1.MoveNext
    Wend
  Else
    Set tabelas_contagem_1 = agenda.OpenRecordset("Select * FROM Dados WHERE [" & tabelas(tabela1) & "] = '" & valor1 & "';", dbOpenDynaset)
    If Not tabelas_contagem_1.EOF Or Not tabelas_contagem_1.BOF Then tabelas_contagem_1.MoveFirst
    While Not tabelas_contagem_1.EOF
      soma_pertinencias = soma_pertinencias + tabelas_contagem_1.Fields(tabelas(tabela2))
      tabelas_contagem_1.MoveNext
    Wend
  End If

  If soma_pertinencias > (Combo2.Text * tabelas_agenda.RecordCount) / 100 Then
    valor(1) = Replacevb5(valor1, "'", "")
    valor(2) = Replacevb5(valor2, "'", "")
    List3.AddItem Replacevb5(tabelas(tabela1), "_Fuzzy", "") & ": " & valor(1) & "; " & Replacevb5(tabelas(tabela2), "_Fuzzy", "") & ": " & valor(2) & " - " & Int(1000 * soma_pertinencias / total) / 10 & "%"
    Call adiciona("contagem_itemset_2", Replacevb5(tabelas(tabela1), "_Fuzzy", "") & ": " & valor(1) & "; " & Replacevb5(tabelas(tabela2), "_Fuzzy", "") & ": " & valor(2) & " - " & Int(1000 * soma_pertinencias / total) / 10 & "%")
    If a < b Then
      itemset2(List3.ListCount).campo(1) = a
      itemset2(List3.ListCount).campo(2) = b
      itemset2(List3.ListCount).valor(1) = valor(1)
      itemset2(List3.ListCount).valor(2) = valor(2)
    Else
      itemset2(List3.ListCount).campo(1) = a
      itemset2(List3.ListCount).campo(2) = b
```

```
        itemset2(List3.ListCount).valor(1) = valor(1)
        itemset2(List3.ListCount).valor(2) = valor(2)
      End If
      itemset2(List3.ListCount).contador = soma_pertinencias
      itemset2(List3.ListCount).cont_percentual = Int(1000 * (soma_pertinencias / total)) / 10
   End If
```

End Function

Function contagem_efetiva_fuzzy_3(ByVal tabela1 As Integer, ByVal valor1 As String, ByVal tabela2 As Integer, ByVal valor2 As String, ByVal tabela3 As Integer, ByVal valor3 As String, ByVal a As Integer, ByVal b As Integer, ByVal c As Integer)

```
   controle1 = False
   controle2 = False
   controle3 = False
   If Right(tabelas(tabela1), 5) = "Fuzzy" Then
      controle1 = True
      tabela = Left(tabelas(tabela1), Len(tabelas(tabela1)) - 6)
      Set tabelas_pesquisa = agenda.OpenRecordset("Select * FROM Fuzzy WHERE Campo = '" & tabela & "' AND Conceito = '" & valor1 & "';", dbOpenDynaset)
      Call contagem_fuzzy(tabela, valor1, tabelas_pesquisa.Fields("Ponto baixo 1"), tabelas_pesquisa.Fields("Ponto baixo 2"), tabelas_pesquisa.Fields("Ponto alto"))
   End If
   If Right(tabelas(tabela2), 5) = "Fuzzy" Then
      controle2 = True
      tabela = Left(tabelas(tabela2), Len(tabelas(tabela2)) - 6)
      Set tabelas_pesquisa = agenda.OpenRecordset("Select * FROM Fuzzy WHERE Campo = '" & tabela & "' AND Conceito = '" & valor2 & "';", dbOpenDynaset)
      Call contagem_fuzzy(tabela, valor2, tabelas_pesquisa.Fields("Ponto baixo 1"), tabelas_pesquisa.Fields("Ponto baixo 2"), tabelas_pesquisa.Fields("Ponto alto"))
   End If
   If Right(tabelas(tabela3), 5) = "Fuzzy" Then
      controle3 = True
      tabela = Left(tabelas(tabela3), Len(tabelas(tabela3)) - 6)
```

```
    Set tabelas_pesquisa = agenda.OpenRecordset("Select * FROM Fuzzy WHERE Campo = '" & tabela &
"' AND Conceito = '" & valor3 & "';", dbOpenDynaset)
    Call contagem_fuzzy(tabela, valor3, tabelas_pesquisa.Fields("Ponto baixo 1"),
tabelas_pesquisa.Fields("Ponto baixo 2"), tabelas_pesquisa.Fields("Ponto alto"))
  End If

  soma_pertinencias = 0

  If controle1 And controle2 And controle3 Then
    Set tabelas_contagem_1 = agenda.OpenRecordset("Select * FROM Dados;", dbOpenDynaset)
    If Not tabelas_contagem_1.EOF Or Not tabelas_contagem_1.BOF Then tabelas_contagem_1.MoveFirst
    While Not tabelas_contagem_1.EOF
      soma_pertinencias = soma_pertinencias + tabelas_contagem_1.Fields(tabelas(tabela1)) *
tabelas_contagem_1.Fields(tabelas(tabela2)) * tabelas_contagem_1.Fields(tabelas(tabela3))
      tabelas_contagem_1.MoveNext
    Wend
  ElseIf controle1 And controle2 Then
    Set tabelas_contagem_1 = agenda.OpenRecordset("Select * FROM Dados WHERE [" &
tabelas(tabela3) & "] = '" & valor3 & "';", dbOpenDynaset)
    If Not tabelas_contagem_1.EOF Or Not tabelas_contagem_1.BOF Then tabelas_contagem_1.MoveFirst
    While Not tabelas_contagem_1.EOF
      soma_pertinencias = soma_pertinencias + tabelas_contagem_1.Fields(tabelas(tabela1)) *
tabelas_contagem_1.Fields(tabelas(tabela2))
      tabelas_contagem_1.MoveNext
    Wend
  ElseIf controle1 And controle3 Then
    Set tabelas_contagem_1 = agenda.OpenRecordset("Select * FROM Dados WHERE [" &
tabelas(tabela2) & "] = '" & valor2 & "';", dbOpenDynaset)
    If Not tabelas_contagem_1.EOF Or Not tabelas_contagem_1.BOF Then tabelas_contagem_1.MoveFirst
    While Not tabelas_contagem_1.EOF
      soma_pertinencias = soma_pertinencias + tabelas_contagem_1.Fields(tabelas(tabela1)) *
tabelas_contagem_1.Fields(tabelas(tabela3))
      tabelas_contagem_1.MoveNext
    Wend
```

```
   ElseIf controle2 And controle3 Then
      Set tabelas_contagem_1 = agenda.OpenRecordset("Select * FROM Dados WHERE [" &
tabelas(tabela1) & "] = '" & valor1 & "';", dbOpenDynaset)
      If Not tabelas_contagem_1.EOF Or Not tabelas_contagem_1.BOF Then tabelas_contagem_1.MoveFirst
      While Not tabelas_contagem_1.EOF
         soma_pertinencias = soma_pertinencias + tabelas_contagem_1.Fields(tabelas(tabela2)) *
tabelas_contagem_1.Fields(tabelas(tabela3))
         tabelas_contagem_1.MoveNext
      Wend
   ElseIf controle1 Then
      Set tabelas_contagem_1 = agenda.OpenRecordset("Select * FROM Dados WHERE [" &
tabelas(tabela2) & "] = '" & valor2 & "' AND [" & tabelas(tabela3) & "] = '" & valor3 & "';", dbOpenDynaset)
      If Not tabelas_contagem_1.EOF Or Not tabelas_contagem_1.BOF Then tabelas_contagem_1.MoveFirst
      While Not tabelas_contagem_1.EOF
         soma_pertinencias = soma_pertinencias + tabelas_contagem_1.Fields(tabelas(tabela1))
         tabelas_contagem_1.MoveNext
      Wend
   ElseIf controle2 Then
      Set tabelas_contagem_1 = agenda.OpenRecordset("Select * FROM Dados WHERE [" &
tabelas(tabela1) & "] = '" & valor1 & "' AND [" & tabelas(tabela3) & "] = '" & valor3 & "';", dbOpenDynaset)
      If Not tabelas_contagem_1.EOF Or Not tabelas_contagem_1.BOF Then tabelas_contagem_1.MoveFirst
      While Not tabelas_contagem_1.EOF
         soma_pertinencias = soma_pertinencias + tabelas_contagem_1.Fields(tabelas(tabela2))
         tabelas_contagem_1.MoveNext
      Wend
   ElseIf controle3 Then
      Set tabelas_contagem_1 = agenda.OpenRecordset("Select * FROM Dados WHERE [" &
tabelas(tabela1) & "] = '" & valor1 & "' AND [" & tabelas(tabela2) & "] = '" & valor2 & "';", dbOpenDynaset)
      If Not tabelas_contagem_1.EOF Or Not tabelas_contagem_1.BOF Then tabelas_contagem_1.MoveFirst
      While Not tabelas_contagem_1.EOF
         soma_pertinencias = soma_pertinencias + tabelas_contagem_1.Fields(tabelas(tabela3))
         tabelas_contagem_1.MoveNext
      Wend
   End If
```

```
   If soma_pertinencias > (Combo2.Text * tabelas_agenda.RecordCount) / 100 Then
      valor(1) = Replacevb5(valor1, """, "")
      valor(2) = Replacevb5(valor2, """, "")
      valor(3) = Replacevb5(valor3, """, "")
      nome = Replacevb5(tabelas(tabela1), "_Fuzzy", "") & ": " & valor(1) & "; " & Replacevb5(tabelas(tabela2), "_Fuzzy", "") & ": " & valor(2) & "; " & Replacevb5(tabelas(tabela3), "_Fuzzy", "") & ": " & valor(3) & " - " & Int(1000 * soma_pertinencias / total) / 10 & " %"
      List4.AddItem nome
      Call adiciona("contagem_itemset_3", nome)
      Call preenche_3(a, b, c, valor(1), valor(2), valor(3))
      itemset3(List4.ListCount).contador = soma_pertinencias
      itemset3(List4.ListCount).cont_percentual = Int(1000 * (soma_pertinencias / total)) / 10
   End If
```

End Function

Function depuracao_3(campo1 As Integer, valor1 As String, campo2 As Integer, valor2 As String, campo3 As Integer, valor3 As String) As Boolean

'Elimina os itemsets candidates 3 que algum de seus 3 subconjuntos com 2 elementos não seja itemset freqüente 2

```
   soma = 0
   depuracao_3 = False
   For d = 1 To UBound(itemset2)
      If itemset2(d).campo(1) = campo1 And itemset2(d).campo(2) = campo2 And itemset2(d).valor(1) = valor1 And itemset2(d).valor(2) = valor2 Then
         soma = soma + 1
      End If
      If itemset2(d).campo(1) = campo1 And itemset2(d).campo(2) = campo3 And itemset2(d).valor(1) = valor1 And itemset2(d).valor(2) = valor3 Then
         soma = soma + 1
      End If
```

```
    If itemset2(d).campo(1) = campo2 And itemset2(d).campo(2) = campo3 And itemset2(d).valor(1) = valor2 And itemset2(d).valor(2) = valor3 Then
        soma = soma + 1
    End If
  Next d
  If soma = 3 Then depuracao_3 = True
```

End Function

Function preenche_3(ByVal a As Integer, ByVal b As Integer, ByVal c As Integer, ByVal valor_1 As String, ByVal valor_2 As String, ByVal valor_3 As String)

```
  Dim grupo(3) As grupo_valores
  Dim troca As Boolean
  Dim temp1 As Integer
  Dim temp2 As String

  grupo(1).campo = a
  grupo(1).valor = valor(1)
  grupo(2).campo = b
  grupo(2).valor = valor(2)
  grupo(3).campo = c
  grupo(3).valor = valor(3)

  troca = True
  While troca
  troca = False
    For d = 1 To 2
      If grupo(d).campo > grupo(d + 1).campo Then
        troca = True
        temp1 = grupo(d).campo
        grupo(d).campo = grupo(d + 1).campo
        grupo(d + 1).campo = temp1
        temp2 = grupo(d).valor
```

```
            grupo(d).valor = grupo(d + 1).valor
            grupo(d + 1).valor = temp2
        End If
    Next d
Wend

itemset3(List4.ListCount).campo(1) = grupo(1).campo
itemset3(List4.ListCount).campo(2) = grupo(2).campo
itemset3(List4.ListCount).campo(3) = grupo(3).campo
itemset3(List4.ListCount).valor(1) = grupo(1).valor
itemset3(List4.ListCount).valor(2) = grupo(2).valor
itemset3(List4.ListCount).valor(3) = grupo(3).valor
```

End Function

Sub adiciona(ByVal arquivo As String, ByVal texto As String)

```
        'adiciona linha no arquivo de resultado
Set aaa = fs.OpenTextFile(kz & arquivo & "_" & Form2.nome_apriori & ".txt", 8, 0)
aaa.Writeline texto
aaa.Close
```

End Sub

Sub limpa_resultados()

```
nome = "condicao.txt"
nome = Left(nome, Len(nome) - 4) & "_" & Form2.nome_apriori & ".txt"
If Dir(kz & nome) <> "" Then
   fs.deletefile kz & nome, True
End If
Set aaa = fs.CreateTextFile(kz & nome, True)
aaa.Writeline "Confiança = " & Combo1.Text
aaa.Writeline "Suporte = " & Combo2.Text
aaa.Close
```

```
nome = "contagem_itemset_1.txt"
nome = Left(nome, Len(nome) - 4) & "_" & Form2.nome_apriori & ".txt"
If Dir(kz & nome) <> "" Then
   fs.deletefile kz & nome, True
End If
Set aaa = fs.CreateTextFile(kz & nome, True)
aaa.Close

nome = "contagem_itemset_2.txt"
nome = Left(nome, Len(nome) - 4) & "_" & Form2.nome_apriori & ".txt"
If Dir(kz & nome) <> "" Then
   fs.deletefile kz & nome, True
End If
Set aaa = fs.CreateTextFile(kz & nome, True)
aaa.Close

nome = "contagem_itemset_3.txt"
nome = Left(nome, Len(nome) - 4) & "_" & Form2.nome_apriori & ".txt"
If Dir(kz & nome) <> "" Then
   fs.deletefile kz & nome, True
End If
Set aaa = fs.CreateTextFile(kz & nome, True)
aaa.Close

nome = "verifica_1_para_2.txt"
nome = Left(nome, Len(nome) - 4) & "_" & Form2.nome_apriori & ".txt"
If Dir(kz & nome) <> "" Then
   fs.deletefile kz & nome, True
End If
Set aaa = fs.CreateTextFile(kz & nome, True)
aaa.Close

nome = "verifica_1_para_3.txt"
nome = Left(nome, Len(nome) - 4) & "_" & Form2.nome_apriori & ".txt"
```

```
If Dir(kz & nome) <> "" Then
   fs.deletefile kz & nome, True
End If
Set aaa = fs.CreateTextFile(kz & nome, True)
aaa.Close

nome = "verifica_2_para_3.txt"
nome = Left(nome, Len(nome) - 4) & "_" & Form2.nome_apriori & ".txt"
If Dir(kz & nome) <> "" Then
   fs.deletefile kz & nome, True
End If
Set aaa = fs.CreateTextFile(kz & nome, True)
aaa.Close
```

End Sub