

UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE CIÊNCIA DA COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO



MINERAÇÃO DE PREFERÊNCIAS CONDICIONAIS

NÁDIA FÉLIX FELIPE DA SILVA

Uberlândia - Minas Gerais

2011

UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE CIÊNCIA DA COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO



NÁDIA FÉLIX FELIPE DA SILVA

MINERAÇÃO DE PREFERÊNCIAS CONDICIONAIS

Dissertação de Mestrado apresentada à Faculdade de Ciência da Computação da Universidade Federal de Uberlândia, Minas Gerais, como parte dos requisitos exigidos para obtenção do título de Mestre em Ciência da Computação.

Área de concentração: Banco de Dados.

Orientadora:

Prof^a. Dr^a. Sandra Aparecida de Amo

Uberlândia, Minas Gerais
2011

UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE CIÊNCIA DA COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Os abaixo assinados, por meio deste, certificam que leram e recomendam para a Faculdade de Ciência da Computação a aceitação da dissertação intitulada “**Mineração de Preferências Condicionais**” por **Nádia Félix Felipe da Silva** como parte dos requisitos exigidos para a obtenção do título de **Mestre em Ciência da Computação**.

Uberlândia, 25 de Fevereiro de 2011

Orientadora:

Profª. Drª. Sandra Aparecida de Amo
Universidade Federal de Uberlândia

Banca Examinadora:

Profª. Drª. Gina Maira Barbosa de Oliveira
Universidade Federal de Uberlândia

Profª. Drª. Marilde Terezinha Prado Santos
Universidade Federal de São Carlos

UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE CIÊNCIA DA COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Data: Fevereiro de 2011

Autor: **Nádia Félix Felipe da Silva**
Título: **Mineração de Preferências Condicionais**
Faculdade: **Faculdade de Ciência da Computação**
Grau: **Mestrado**

Fica garantido à Universidade Federal de Uberlândia o direito de circulação e impressão de cópias deste documento para propósitos exclusivamente acadêmicos, desde que o autor seja devidamente informado.

Autor

O AUTOR RESERVA PARA SI QUALQUER OUTRO DIREITO DE PUBLICAÇÃO DESTE DOCUMENTO, NÃO PODENDO O MESMO SER IMPRESSO OU REPRODUZIDO, SEJA NA TOTALIDADE OU EM PARTES, SEM A PERMISSÃO ESCRITA DO AUTOR.

Dedicatória

A Deus, aos meus pais Dioclemar e Iva, aos meus irmãos Deibymar e Erasmo e aos meus sobrinhos Ana Júlia, Maria Eduarda e João Antônio.

A meu namorado Márcio.

Agradecimentos

Embora uma dissertação seja, pela sua finalidade acadêmica, um trabalho individual, é imprescindível agradecer a todos aqueles que participaram direta e indiretamente deste trabalho, contribuindo de formas diversas e que não podem e nem devem deixar de ser realçados. Por essa razão, desejo expressar os meus sinceros agradecimentos:

A Deus, pelas inúmeras oportunidades de crescimento que me proporciona, me permitindo chegar ao final de mais uma etapa importante de minha vida.

Aos meus pais Diocleamar e Iva, pelo amor incondicional, pelo apoio, pela confiança e por estar ao meu lado em todas as circunstâncias.

Aos meus irmãos, Deibymar e Erasmo pelo carinho e apoio.

Aos meus sobrinhos Ana Júlia, Maria Eduarda e João Antônio pelos momentos de descontração e alegria.

Ao meu namorado Márcio, pela compreensão e amor, por ter estado ao meu lado, mesmo à distância, me entendendo e apoiando.

A todos os meus amigos, em especial Fabíola, Luciana, Diogo, Gildo, Núbia, Marcos Bueno, Marcos Roberto, Rodrigo, Liliane e Lacordaire pelo apoio, contribuição e amizade.

Agradeço ainda aos meus amigos do Grupo de Oração Universitário Santa Mônica e à Comunidade São Paulo Apóstolo pelo apoio espiritual, pelos momentos de profundo encontro com Deus, pelos momentos de comunhão, amizade e carinho.

Aos professores do Departamento de Pós-Graduação da FACOM/UFU pelo conhecimento compartilhado durante todo o período do mestrado. Também ao secretário da Pós-Graduação Erisvaldo, pela gentileza, carinho e amizade com que sempre me atendeu.

A CAPES, pelo apoio financeiro concedido na realização deste trabalho.

Principalmente à professora Sandra de Amo pelo profissionalismo, apoio, paciência, amizade e orientação em todos os momentos da realização deste trabalho.

De modo geral, agradeço a todos que contribuíram, de alguma forma, para a concretização deste trabalho e também de minha formação como pessoa.

“Nunca seremos suficientemente gratos a Deus, aos nossos pais e aos nossos mestres.”
(Aristóteles)

Resumo

A personalização de conteúdo em consultas a bancos de dados por meio do uso das *preferências do usuário* é um tópico de pesquisa que recentemente vem despertando muito interesse na comunidade de banco de dados. Tais preferências são utilizadas para ordenação e seleção das melhores tuplas, aquelas que mais se adequam ao desejo expresso pelo usuário. Um tópico de interesse dentro deste contexto é a *elicitação de preferências*, constituída de meios do usuário informar sua escolha sobre pares de objetos pertencentes a um banco de dados. Tal tarefa pode demandar um grande esforço do usuário, dependendo do tamanho do banco de dados, fato este que pode desmotivar o uso do sistema por parte do usuário. Neste trabalho, propomos uma metodologia para *inferir as preferências do usuário de maneira automática* através da tarefa de mineração, a partir de um conjunto de amostras de preferências do usuário. O algoritmo proposto foi projetado para minerar um tipo especial de preferências chamadas *preferências condicionais*. Tal metodologia utiliza o *framework* das redes bayesianas de crença, formalismo conhecido por sua eficiência na construção de modelos de situações envolvendo incerteza.

Palavras chave: mineração de preferências, preferências condicionais, redes bayesianas, redes de preferências.

Abstract

Content customization on queries to databases through the use of *user's preferences* is a research topic that has been raising a lot of interest within the database community in recent years. Such preferences are used for sorting and selecting the best tuples, those which most fulfill the user's wishes. A topic of interest within this context is the *elicitation of preferences*, consisting of methods to enable the user to inform his choice on pairs of objects belonging to a database. This task may require a great effort from the user, depending on the size of the database, a fact that may discourage the use of the system by the user. In this work, we propose a methodology *to infer user preferences in an automatic way* through the task of mining, from a sample set of user preferences. The algorithm was designed to mine a special type of preferences called *conditional preferences*. The method we propose is based on the framework of Bayesian Belief Networks, a very efficient formalism for modeling situations involving uncertainty.

Keywords: preference mining, condicional preferences, bayesian networks, preference networks.

Sumário

Lista de Figuras	xi
Lista de Tabelas	xiii
1 Introdução	15
1.1 Obtenção das informações sobre as preferências do usuário	16
1.2 Utilizar de informações sobre preferências do usuário para melhor atendê-lo	19
1.3 Contexto deste trabalho	19
1.4 Contribuições	20
1.5 Metodologia	21
1.6 Organização da dissertação	22
2 Fundamentos Teóricos	24
2.1 Modelos de Preferências	24
2.1.1 Preferências não condicionais	25
2.1.2 Preferências condicionais	28
2.2 Metodologia para aprendizado de redes bayesianas	33
2.2.1 Aprendizado de parâmetros	34
2.2.2 Aprendizado de estrutura	36
2.2.3 O algoritmo $K2$	39
2.3 Considerações finais	40
3 Trabalhos Correlatos em Mineração de Preferências	41
3.1 Mineração de preferências qualitativas	42
3.1.1 Aprendizado de preferências qualitativas utilizando redes bayesianas	44
3.2 Mineração de preferências quantitativas	45
3.3 Considerações finais	47
4 Formalização do Problema	48
4.1 Definições preliminares	48
4.2 Formalização do problema	57
4.3 Considerações finais	59

5	Métodos Propostos	60
5.1	Aprendizagem da estrutura de rede de preferências	61
5.1.1	Funções <i>scores</i> consideradas	61
5.1.2	Estratégia de busca	62
5.1.3	Algoritmo <i>PrefK2</i>	62
5.1.4	Algoritmo <i>PrefK2*</i>	67
5.2	Treinamento da rede de preferências	70
5.3	Considerações finais	72
6	Validação dos métodos propostos	73
6.1	Dados utilizados na validação	73
6.1.1	Dados sintéticos	73
6.1.2	Dados reais	76
6.2	Resultados Experimentais	79
6.2.1	Medidas consideradas na comparação dos algoritmos	79
6.2.2	Técnica de amostragem: <i>cross-validation</i>	80
6.2.3	Resultados em dados sintéticos	80
6.2.4	Resultados com dados reais	90
6.2.5	Algumas redes encontradas nos dados reais	91
6.3	Considerações finais	93
7	Conclusão e Trabalhos Futuros	94
A	Outros métodos para aprendizado da rede de preferência	96
A.1	Aprendizado da estrutura de redes de preferências	96
A.1.1	Outro método de busca guloso	96
A.2	Treinamento em redes de preferências	98
A.2.1	Método 1: Treinamento de uma rede de preferências (utilizando o algoritmo ID3)	98
A.2.2	Método 2: Treinamento de uma rede de preferências (utilizando o cálculo da Esperança)	99
B	Exemplo de Tabelas de Probabilidades Condicionais encontradas nos dados reais	101
B.1	Usuário nº821 com <i>input</i> de 100 amostras de preferências	101
B.2	Usuário nº3781 com <i>input</i> de 1000 amostras de preferências	103
B.3	Usuário nº620 com <i>input</i> de 10.000 amostras de preferências	106
	Referências Bibliográficas	110

Lista de Figuras

1.1	Ciclo de vida de preferências do usuário	18
1.2	Busca Tradicional vs. busca personalizada. Extraído de [Koutrika e Ioan- nidis 2006]	19
1.3	Analogia do problema de ordenação ao problema de classificação	21
2.1	Preferência PARETO	28
2.2	Preferência CASCADE	28
2.3	CP-net para as preferências sobre filmes	29
2.4	TCP-net	30
2.5	Ordem de Preferência induzida pela Teoria Γ de Preferência Condicional .	33
2.6	Contribuição dada pelo trabalho de [Wilson 2004]	33
2.7	Tipos de Conexões	37
2.8	Exemplo de uma estrutura bayesiana	38
3.1	Preferências sobre atributos “soft” derivam preferências sobre atributos di- tos técnicos (<i>technical</i>). Figura extraída de [Radde e Freitag 2010]	44
4.1	Rede de preferências <i>PrefNet</i> ₁	50
4.2	Comparação entre a ordem de [Wilson 2004] e a ordem proposta em nossa abordagem	51
4.3	Rede de preferências	52
4.4	Rede de preferências exemplo	53
4.5	Rede de preferências para o exemplo	54
4.6	Exemplo	56
4.7	Particionamentos realizados nas amostras da Tabela A.2	57
4.8	Contextualização do problema de construção de uma rede de preferências .	58
5.1	Esquema do algoritmo <i>PrefK2</i>	62
5.2	Rede de preferências <i>PrefNet</i> ₁	63
5.3	Rede de preferências <i>PrefNet</i> ₂	64
5.4	Árvore de índice para a partição segundo $\text{Comp}(D)$ da Figura 4.7(d), su- pondo que D depende do atributo C	65

5.5	Algoritmo <i>PrefK2</i>	65
5.6	Esquema do algoritmo <i>PrefK2*</i>	67
5.7	Rede de preferências B_{S1}	68
5.8	Rede de preferências B_{S2}	68
5.9	Algoritmo <i>PrefK2*</i>	70
5.10	Banco de Amostras de preferências espelhadas do usuário e a respectiva rede de preferências inferida	71
6.1	Validação	74
6.2	Rede de preferência de <i>input</i> (Gerada aleatoriamente)	75
6.3	Intervalo numérico para geração de amostras do atributo A e B referente ao primeiro nível da rede de preferências da Figura 6.2	76
6.4	Parte do esquema relacional do <i>benchmark</i> utilizado para testes em dados reais com a nossa abordagem	77
6.5	<i>3-fold cross-validation</i>	80
6.6	Gráfico Acurácia X N° de amostras	82
6.7	Gráfico Precisão X N° de amostras	83
6.8	<i>Performance</i> X Número de Amostras	84
6.9	Gráfico Acurácia X N° de Atributos	85
6.10	Precisão X Número de Atributos	85
6.11	<i>Performance</i> X Numero de Atributos	86
6.12	Gráfico complexidade da rede X acurácia	87
6.13	Precisão X Variação de Complexidade	88
6.14	<i>Performance</i> X Complexidade	88
6.15	Rede obtida pelo algoritmo <i>PrefK2</i>	90
6.16	Rede obtida pelo algoritmo <i>PrefK2*</i>	90
6.17	Acurácia X variação do número de amostras	91
6.18	Precisão X variação do número de amostras	92
6.19	Redes encontradas para as amostras do usuário cuja Id é 821	92
6.20	Redes encontradas para as amostras do usuário cuja Id é 3781	92
6.21	Redes encontradas para as amostras do usuário cuja Id é 620	93
A.1	Algoritmo <i>PrefDelta</i>	98
A.2	Algoritmo de treinamento de uma rede de preferências	99

Lista de Tabelas

2.1	Exemplo de preferências do usuário e respectiva tradução segundo a especificação de [Kießling 2002]	26
2.2	Banco de dados I.	27
2.3	Banco de dados II.	32
3.1	Exemplos de filmes onde o usuário especifica quais são inferiores e superiores.	43
4.1	BD R	48
4.2	Amostras de preferências	48
4.3	Espelhamento realizado na Tabela 4.2	49
4.4	Amostras de preferências	57
5.1	A tabela mostra o número de diferentes grafos direcionados acíclicos (DAGs) que podem ser gerados para um dado número de nós (atributos). Por exemplo, existem $4.2 \cdot 10^{18}$ diferentes grafos direcionados acíclicos com 10 nós ([Robinson 1977]).	61
5.2	Inversões.	66
5.3	Cálculos N_{ij1} , N_{ij0} e N^l para cada $Comp(A_i)$	69
6.1	Amostras de preferências obtidas pelo método <i>Probabilistic Logic Sampling</i>	74
6.2	Usuários $\times$ número de Filmes avaliados	77
6.3	tabela $I_UserRatings$	78
6.4	Pré-processamento intermediário	78
6.5	Amostras de preferências obtidas pelo pré-processamento	79
6.6	Parâmetros considerados nos dados sintéticos	81
6.7	Acurácia de <i>PrefK2</i> variando o número de amostras	82
6.8	Acurácia de <i>PrefK2*</i> variando o número de amostras	82
6.9	Precisão <i>PrefK2</i> variando o número de amostras	83
6.10	Precisão <i>PrefK2*</i> variando o número de amostras	83
6.11	<i>Performance PrefK2</i> variando o número de amostras	83
6.12	<i>Performance PrefK2*</i> variando o número de amostras	84
6.13	Acurácia de <i>PrefK2</i> variando o número de atributos	84

6.14	Acurácia de <i>PrefK2*</i> variando o número de atributos	84
6.15	Precisão de <i>PrefK2</i> variando o número de atributos	85
6.16	Precisão <i>PrefK2*</i> variando o número de atributos	85
6.17	<i>Performance PrefK2</i> variando o número de atributos	86
6.18	<i>Performance PrefK2*</i> variando o número de atributos	86
6.19	Acurácia de <i>PrefK2</i> variando a complexidade da rede	86
6.20	Acurácia de <i>PrefK2*</i> variando a complexidade da rede	87
6.21	Precisão de <i>PrefK2</i> variando a complexidade da rede	87
6.22	Precisão de <i>PrefK2*</i> variando a complexidade da rede	87
6.23	<i>Performance</i> de <i>PrefK2</i> variando a complexidade da rede de preferências .	88
6.24	<i>Performance</i> de <i>PrefK2*</i> variando a complexidade da rede de preferências	88
6.25	Amostras de preferências	90
6.26	Cálculo da acurácia variando as amostras de preferências do banco de dados reais - Algoritmo <i>PrefK2</i>	90
6.27	Cálculo da precisão variando as amostras de preferências do banco de dados reais - Algoritmo <i>PrefK2</i>	91
6.28	Cálculo da acurácia variando as amostras de preferências do banco de dados reais - Algoritmo <i>PrefK2*</i>	91
6.29	Cálculo da precisão variando as amostras de preferências do banco de dados reais - Algoritmo <i>PrefK2*</i>	91
A.1	Amostras de preferências	99
A.2	Espelhamento da Tabela A.1	99

Capítulo 1

Introdução

O crescimento do conteúdo *web* e de aplicações baseadas na *internet* podem levar a um comportamento de insatisfação por parte dos usuários. Instrumentos de busca retornam um grande número de resultados, sendo que muitos desses resultados não interessam ao usuário. Tal comportamento, leva não somente a usuários frustrados, mas também a uma redução nas vendas no comércio eletrônico, tendo em vista que os clientes que são propensos a comprar não podem fazer isso se não encontram o produto certo, num tempo razoável. Para solucionar esse problema, pesquisas recentes vêm sendo voltadas para a questão das *preferências* do usuário [Koutrika et al. 2010], [Mobasher 2007], de maneira que os resultados de uma busca possam ser filtrados e organizados.

Um usuário pode estabelecer suas preferências sobre quase tudo que é feito no dia-a-dia, inclusive em situações comerciais. Sob o ponto de vista filosófico, alguns trabalhos abordando preferências já foram realizados, dentre eles é possível citar [Castañeda 1958], [von Wright 1963], [von Wright 1972], [Kron e Milovanovic 1975], [Trapp 1985] e [Hansson 1996].

Recentemente pesquisadores voltaram seu interesse para a especificação, representação e raciocínio com preferências na linha de pesquisa de Inteligência Artificial e para linguagens de consultas com suporte a preferências, técnicas de elicitación e dedução de preferências na linha de pesquisa de Banco de Dados.

Em Inteligência Artificial o uso de preferências se dá através de aplicações como Sistemas de Recomendação [Nguyen e Haddawy 1999] [Resnick e Varian 1997] (tais sistemas são capazes de fazer sugestões ao usuário), Personalização de Conteúdo [Domshlak et al. 2001] [Brafman et al. 2004] (tendo em vista que o objetivo é a personalização de conteúdo de maneira automática) e em aplicações médicas [Brafman 2005].

Os Sistemas de Recomendação aplicados ao *e-Commerce* vêm se tornando uma área de grande sucesso, sendo estudados tanto em Inteligência Artificial, quanto em Banco de Dados. Em Inteligência Artificial trabalhos como [Sarwar et al. 2000, Mobasher et al. 2002, Mobasher et al. 2000] com os chamados “Filtros Colaborativos” são bastante abordados, onde dado um registro de uma ação do cliente é possível inferir a ação de um outro

cliente com o mesmo perfil do anterior por meio da tecnologia de filtragem colaborativa (usando por exemplo o algoritmo *K-Nearest Neighbors*). A recomendação de conteúdo segundo o ponto de vista de pesquisadores de Banco de Dados será tratada mais adiante.

Em Banco de Dados é possível dividir o campo de pesquisa segundo dois importantes aspectos: 1) Obtenção das informações sobre as preferências do usuário, isto é, técnicas de elicitación e dedução de preferências do usuário e 2) Utilização das informações sobre preferências do usuário para melhor atendê-lo. A seguir é explicado cada um desses possíveis campos de pesquisa.

1.1 Obtenção das informações sobre as preferências do usuário

Tendo em vista que os usuários possuem características e gostos distintos, uma importante fase do processo de utilização das preferências em personalização de conteúdo está na questão de como obter as informações sobre as preferências do usuário, traduzindo-as para um modelo computacional sem que haja perda da expressividade, o que chamamos de *elicitación preferências*. A elicitación pode ser feita de *maneira manual* onde o próprio usuário informa suas preferências e estas são traduzidas segundo algum modelo ou *framework*, ou de *maneira automática* utilizando *técnicas de mineração de dados*.

A elicitación de preferências manual ou automática pode ser especificada de maneira *quantitativa* ou *qualitativa*. Por exemplo, dada uma certa lista de filmes, para se conhecer quais filmes são mais preferidos por um usuário de forma quantitativa, pode-se solicitar ao mesmo que dê uma nota a cada filme e então seleciona-se aqueles filmes com maiores notas. A nota dada pelo usuário a cada filme é chamada de *função de utilidade* ou *função de ranking* $f(o)$ [Agrawal e Wimmers 2000] e representa seu grau de preferência em relação ao objeto, ou seja, quanto maior o valor de $f(o)$ maior sua preferência. No entanto, esse método pode ser inviável para uma coleção de filmes muito grande. De forma qualitativa, pode-se obter do usuário informações sobre quais atributos afetam sua preferência. No Exemplo 1.1 são demonstradas as duas formas de elicitación de preferências.

Exemplo 1.1. A elicitación de preferências de forma quantitativa e de forma qualitativa será feita com os seguintes filmes na forma de objetos pertencentes ao esquema relacional Filmes(Título, Gênero, Diretor, Ator, Década):

- o_1 = (“Esse Mundo é um Hospício”, “Comédia”, “F.Capra”, “Raymond Massey”, 1960)
- o_2 = (“Marnie, Confissões de uma Ladra”, “Comédia”, “Hitchcock”, “Raymond Massey”, 1960)
- o_3 = (“Marie Baie Des Anges”, “Comédia”, “F.Capra”, “P. Newman”, 1950)
- o_4 = (“The Manxman”, “Drama”, “Hitchcock”, “P. Newman”, 1950)

Um usuário X pode expressar suas preferências de forma quantitativa como se segue:

- $f(o_1) = 6$;
- $f(o_2) = 4$;
- $f(o_3) = 10$;
- $f(o_4) = 2$;

Um usuário Y pode expressar suas preferências de forma qualitativa através das seguintes “regras”:

1. “Filmes de comédia são melhores do filmes de drama”;
2. “Para filmes de Comédia prefiro os da década de 50 aos da década de 60, pois as comédias da década de 50 são mais inteligentes”;
3. “Para filmes da década de 60 prefiro os do diretor F.Capra pois os filmes do Hitchcock nesse período não me faziam rir”;

É possível constatar que tanto o usuário X quanto o usuário Y expressam as mesmas preferências sobre formas diferentes, classificando os objetos na mesma ordem:

- o_3 é preferido a o_1 , (tanto para X quanto para Y);
- o_1 é preferido a o_2 , (tanto para X quanto para Y);
- o_2 é preferido a o_4 , (tanto para X quanto para Y);

A ordenação feita pelas preferências do usuário X se dá de maneira direta usando os valores retornados pela função $f(o_i)$ organizando-os em ordem crescente. No caso do usuário Y, a primeira regra permite inferir que o_1 , o_2 e o_3 são melhores do que o_4 , a segunda regra permite inferir que o_3 é melhor que o_2 e o_1 e pela terceira regra é possível inferir que o_1 é melhor que o_2 ;

A elicitación de preferências de forma qualitativa é considerada mais eficiente pois se a quantidade de objetos for muito grande, o que é comum em banco de dados, será exigido um enorme esforço do usuário e um razoável consumo de tempo na especificação de preferências de forma quantitativa por ser necessário atribuir uma nota a cada objeto. Desta forma a especificação de preferências quantitativa é tida muitas vezes como inconveniente e em certos casos impraticável, por que em muitas aplicações os usuários não estão aptos ou não possuem tempo suficiente para fornecer mais do que informações sobre preferências qualitativas.

A mineração de dados surge nesse cenário como uma estratégia automática para dedução de preferências, evitando o transtorno para o usuário em informar a sua preferência de forma explícita e muitas vezes exaustiva.

Para entender como o processo de mineração de dados pode ser usado para inferência automática de preferências, vamos considerar que um certo usuário frequentemente acesse

o site *Amazon* e compre *dvds* de filmes de temas diversos. Essas informações são armazenadas pelo site em um banco de *logs* de dados do usuário. A partir dessas informações referentes a escolhas do usuário em situações passadas é possível inferir preferências em um mesmo domínio em situações futuras.

Na Figura 1.1 é possível visualizar o ciclo de vida de preferências do usuário. Tal processo consiste na obtenção de preferências do usuário através de um algoritmo de mineração de dados. Em um primeiro momento o usuário acessa o site e o comportamento referente ao seu acesso é armazenado no banco de dados; a partir do histórico de acessos do usuário são obtidas as preferências do cliente (tarefa da mineração de dados); as preferências obtidas na Figura 1.1 foram especificadas de maneira qualitativa, na forma de regras de preferências, ou seja, para o usuário em questão em um contexto de filmes, o gênero suspense é preferido ao gênero ação, o gênero romance é preferido ao gênero terror, e assim por diante. A partir das regras de preferências obtidas é possível personalizar o conteúdo retornado ao cliente, indicando novos produtos, ofertas especiais e direcionadas. O ciclo de vida de preferências pode ser repetido, de forma que o modelo de preferências obtido possa ser aprimorado.

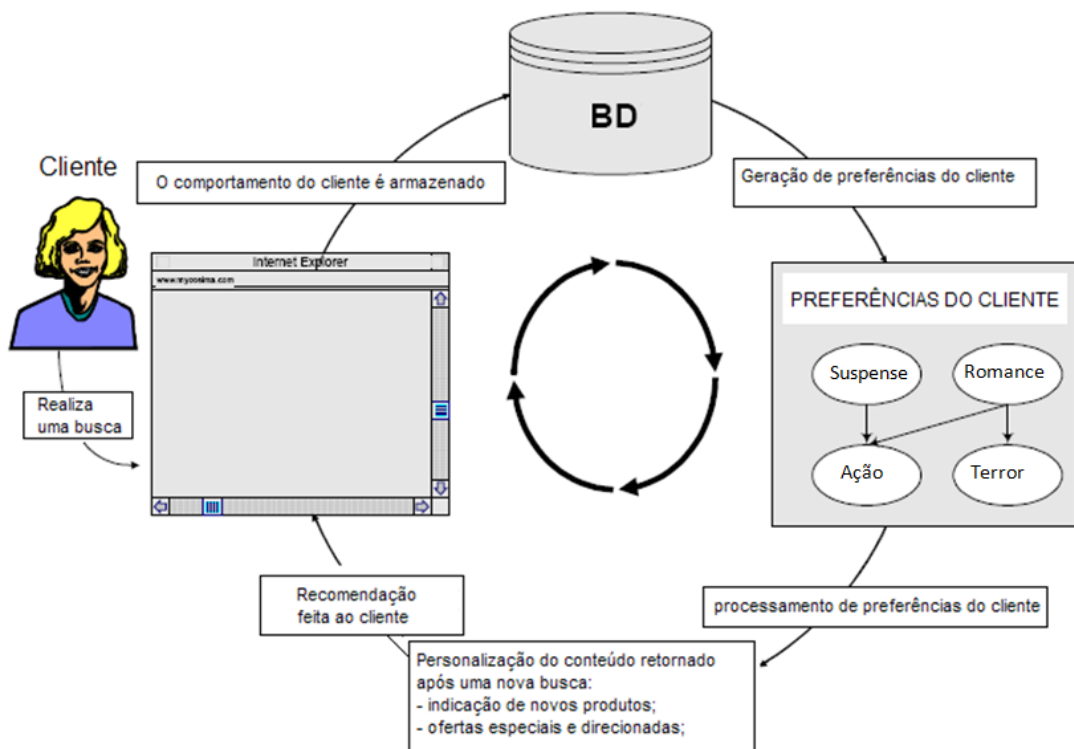


Figura 1.1: Ciclo de vida de preferências do usuário

A mineração de dados aplicada à preferências é intitulada *preference mining* ou *mineração de preferências* [Vojtás e Eckhardt 2008].

1.2 Utilizar de informações sobre preferências do usuário para melhor atendê-lo

Outro problema relacionado a preferências está em utilizá-las de forma a melhorar os resultados de uma busca, restringindo os valores da mesma aos valores preferidos pelo usuário (ver Figura 1.2). Partindo dessa necessidade, pesquisadores têm trabalhado na especificação e álgebra de preferências como em [Chomicki 2003] e [Borzsonyi et al. 2001], além da incorporação de operadores de preferências em linguagens de consulta padrão como a *Preference SQL* definida em [Kießling e Köstler 2002], a *CPrefSQL* apresentada em [de Amo e Ribeiro 2009] e [de Amo e Pereira 2010] e a *Preference XPath* especificada em [Kießling et al. 2001].

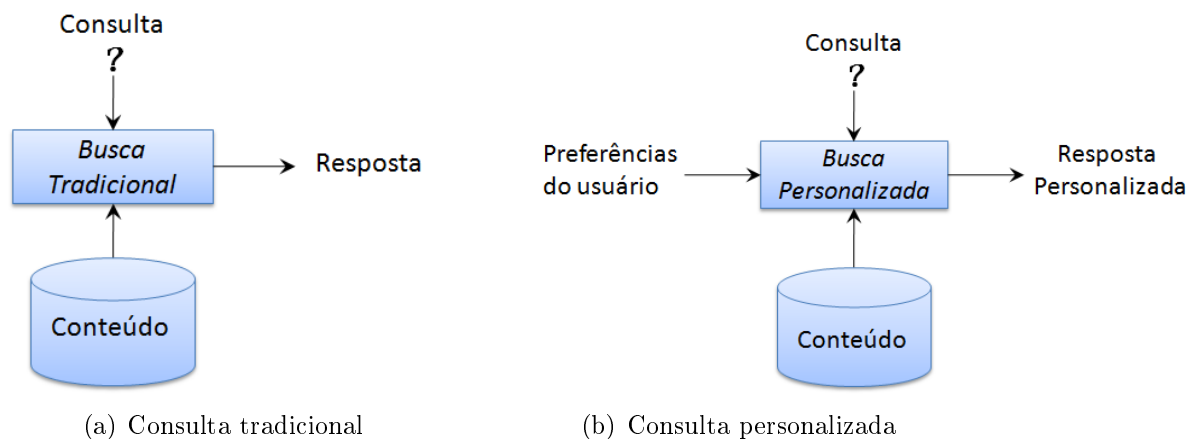


Figura 1.2: Busca Tradicional vs. busca personalizada. Extraído de [Koutrika e Ioannidis 2006]

Há ainda as pesquisas destinadas a descrever Métodos de Acesso para processamento eficiente e otimização de consultas com operadores de preferências [Endres e Kießling 2008].

1.3 Contexto deste trabalho

O universo de pesquisa em torno das preferências ainda é pouco explorado, restando muito a se fazer e descobrir. A partir dos aspectos tratados anteriormente, este trabalho têm como principal objetivo prover ferramentas para a *Obtenção de preferências* de maneira eficiente, com expressividade e corretude, utilizando de *mineração de preferências* como solução para isto.

Ao se falar em Mineração de dados, deve-se responder inicialmente a três perguntas: 1) *De quem serão as preferências a ser mineradas*: um único usuário, um grupo de usuários? 2) *Como as preferências serão formalizadas*, isto é, qual o modelo de inferência utilizado? 3) *Quais tipos de preferências serão obtidas*: condicionais, não condicionais?.

Neste trabalho estamos interessados na personalização de conteúdo a partir das preferências fornecidas por um *único usuário*. Propomos neste trabalho um modelo de inferência de preferências baseado em um modelo de grafos análogo ao modelo das redes bayesianas clássicas, formalismo conhecido por sua eficiência na construção de modelos de situações envolvendo incerteza. As preferências obtidas serão do tipo *qualitativas condicionais* e são baseadas no *framework* das CP-Nets de [Boutilier et al. 2004a].

Temos como principais contribuições, os itens descritos na Seção 1.4.

1.4 Contribuições

As principais contribuições desse trabalho podem ser enumeradas como se seguem:

1. Neste trabalho propomos uma primeira abordagem para eliciação de preferências condicionais de maneira automática utilizando a mineração de dados para tal fim, o que intitulamos de mineração de *preferências qualitativas condicionais*. Até onde sabemos não existem trabalhos envolvendo mineração de preferências qualitativas condicionais.
2. Propomos neste trabalho o modelo de inferência de preferências o qual batizamos de **rede de preferências** baseado em um modelo de grafos análogo ao modelo das redes bayesianas clássicas. Assim como as redes bayesianas, uma rede de preferências é constituída de *uma topologia* e um *conjunto de tabelas de probabilidades condicionais*. A construção ou aprendizado de uma rede de preferências é dividido em duas partes:
 - (a) Aprendizado da topologia da rede.
 - (b) Treinamento da rede (obtenção do conjunto de tabelas de probabilidades condicionais).
3. Implementação do Método Proposto.
4. Validação do método em dados sintéticos e reais.
 - (a) Construção dos dados sintéticos através de um gerador que construímos, que nos permite variações de parâmetros para análise de *performance* e escalabilidade em diferentes cenários.
 - (b) Formatação e limpeza dos dados reais para adequação à entrada do método proposto.

As principais contribuições deste trabalho se concentram nos capítulos 4, 5, 6 e 7.

1.5 Metodologia

Para a obtenção dos resultados apresentados nesta dissertação, começamos realizando uma leitura cuidadosa no estado da arte, definindo o escopo do problema a ser resolvido, isto é, definir um método de inferência de preferências condicionais automático, usando mineração de dados como ferramenta de automação. Há uma certa semelhança entre o problema de ordenação e o problema de classificação (como pode ser visto na Figura 1.3), isto é, nos problemas de classificação busca-se classificar uma dada tupla em uma certa classe e analogamente nos problemas de ordenação busca-se obter uma ordem sobre um par de tuplas. Baseado nesta premissa, foi feita uma pesquisa com o objetivo de encontrar um bom classificador (eficiente e com bons resultados) para inspirar o nosso modelo de ordenação. O classificador com tais características foi o classificador redes bayesianas.

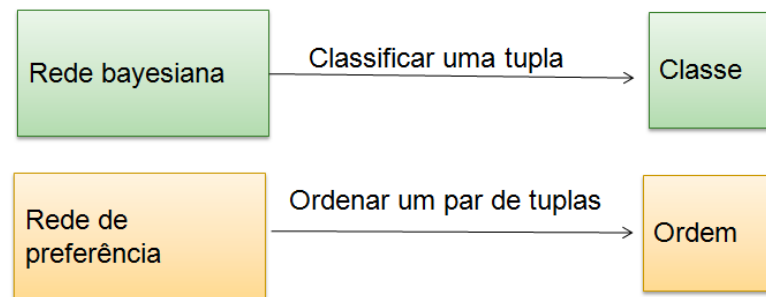


Figura 1.3: Analogia do problema de ordenação ao problema de classificação

O próximo passo foi formalizar o nosso modelo intitulado como *redes de preferências*, modelo constituído por duas componentes: 1) um grafo direcionado acíclico, onde os vértices representam os atributos e as arestas uma dependência condicional entre tais atributos; 2) tabelas de probabilidades condicionais associadas a cada vértice do grafo.

Construir o modelo redes de preferências assim como contruir uma rede bayesiana clássica constitui-se em dois passos: 1) aprender a partir do banco de dados de *input*¹ um grafo direcionado acíclico e 2) de posse do grafo direcionado acíclico e do banco de dados de *input* inferir as tabelas de probabilidade condicional (treinamento da rede de preferências).

O treinamento da rede de preferências constitui um passo considerado *trivial* se comparado ao aprendizado da topologia da rede. No entanto, no início do trabalho não tínhamos ainda este discernimento e o próximo passo que seguimos foi o de propor um método de aprendizado de tabelas de probabilidade (propomos dois métodos para treinamento da rede de preferências apresentados no Anexo A). Mais tarde descobrimos que o aprendizado das tabelas de probabilidade realizado por contagem dos eventos (apresentado em [Jensen e Nielsen 2007]) é o mais eficiente em relação aos demais.

¹no nosso caso a *input* é um banco de dados de amostras de preferências, enquanto que no modelo de redes bayesianas clássicas trata-se de um banco de dados de tuplas simples

De posse de um método para treinamento da rede de preferências eficaz, buscamos encontrar um método para aprendizagem da topologia de rede. Por questões de simplicidade, uma vez que o problema de mineração de preferências condicionais já possui uma certa complexidade, buscamos encontrar um método de aprendizagem de topologia que atendesse a nossa necessidade, sem a preocupação de ser um método guloso ou não. O método proposto foi baseado no *algoritmo K2* [Larrañaga et al. 1997], algoritmo bastante conhecido dentre os estudiosos de redes bayesianas e que vem servindo como base para outros métodos de aprendizagem de estruturas de redes bayesianas.

De posse de um método para treinamento da rede de preferências e também de um método para aprendizagem da topologia da mesma, já foi possível obter os primeiros resultados com relação à acurácia e precisão.

A tentativa de usar a complexidade do grafo de dependência (número de pais que um nó pode ter por exemplo) no aprendizado da topologia de rede teve como objetivo melhorar os resultados obtidos e sanar uma curiosidade: saber se a conformidade da rede de preferências aos dados de *input* poderia ser influenciada pela complexidade da rede, ou ainda duas topologias igualmente prováveis, mas com complexidades diferentes poderiam produzir os mesmos resultados. Chegamos a conclusão que não, que redes menos complexas podem ter resultados melhores.

1.6 Organização da dissertação

A apresentação dessa dissertação está organizada da seguinte maneira: no Capítulo 2 está descrita a fundamentação teórica necessária à compreensão deste trabalho. Tal Capítulo está dividido em duas partes, sendo que inicialmente abordamos conceitos inerentes ao contexto de preferências para que o leitor possa se familiarizar com o tema abordado nesta dissertação. A segunda parte deste Capítulo descreve o aprendizado em redes bayesianas, tal *framework* será constantemente discutido, visto que o método que propomos para mineração de preferências condicionais é baseado neste modelo.

No Capítulo 3 temos o estado da arte, onde descrevemos os principais trabalhos em mineração de preferências.

No Capítulo 4 formalizamos o problema de Mineração de Preferências Condicionais, cujo estudo constitui a principal contribuição desta dissertação. Também neste capítulo, apresentamos os conceitos principais que desenvolvemos para fundamentar teoricamente o nosso método para Mineração de Preferências Condicionais, descrevendo precisamente os conceitos que criamos.

Em seguida, no Capítulo 5 é apresentado nossos métodos para obtenção do minerador de preferências condicionais, o modelo o qual intitulamos de *rede de preferências*.

O Capítulo 6 é dividido em duas partes: na primeira parte abordamos a construção dos dados sintéticos e descrevemos brevemente o *benchmark* para recomendação de filmes

usado nos testes com dados reais. Na segunda parte discutimos sobre os exaustivos testes realizados com o protótipo obtido tanto em dados sintéticos quanto em dados reais.

Finalmente, no Capítulo 7 serão apresentadas as conclusões e perspectivas para trabalhos futuros.

Capítulo 2

Fundamentos Teóricos

Neste capítulo apresentamos o referencial teórico importante para o entendimento desta dissertação. Abordamos conceitos iniciais sobre preferências e redes bayesianas. A nossa proposta de mineração de preferências utiliza o *framework* de redes bayesianas clássico, sendo importante que o leitor tenha uma familiaridade com o tema. Desta forma este capítulo foi dividido em duas partes principais: (1) modelos de preferências e (2) metodologia para aprendizagem de redes bayesianas.

2.1 Modelos de Preferências

Uma *relação de preferência* sobre um conjunto finito de objetos $A = \{a_1, a_2, \dots, a_n\}$ é uma relação de ordem sobre A . Uma relação de ordem sobre A é uma relação $R \subseteq A \times A$ satisfazendo as seguintes propriedades:

1. *irreflexiva*: $\forall a \in A : (a, a) \notin R$;
2. *transitiva*: $\forall a, b, c \in A : (a, b) \in R \text{ e } (b, c) \in R, \text{ então } (a, c) \in R$.

Normalmente uma relação de ordem é representada pelo símbolo $<$. Assim, denotamos $a < b$ se $(a, b) \in$ a relação $<$. Se $<$ é uma relação de preferência entende-se por $a < b$ o fato de b *ser preferido a* a .

Uma relação de ordem é considerada *total* se $\forall a, \forall b \in A, (a, b) \in R$ ou $(b, a) \in R$ e *parcial* caso contrário.

Os trabalhos envolvendo formalismos para especificação de preferências abordam a modelagem das mesmas, descrevendo um padrão formal para tradução de uma linguagem natural para uma linguagem de especificação de preferências.

Os formalismos para especificação de preferências se dividem em duas abordagens. Na primeira abordagem as preferências sobre os valores de cada atributo não dependem eventualmente de preferências sobre os valores dos outros atributos, ou seja, os atributos são *independentes*. Na segunda abordagem as preferências sobre os valores de um atributo podem depender dos valores de outros atributos. No primeiro caso chamamos de

preferências não condicionais, no segundo caso de *preferências condicionais*. A seguir são apresentados trabalhos característicos de cada abordagem.

2.1.1 Preferências não condicionais

O *framework* de [Kießling 2002] aborda preferências cuja modelagem é da forma qualitativa, não condicionais, definindo construtores de preferências básicos para dados categóricos e para dados numéricos.

Os construtores para preferências em domínios categóricos são:

1. **POS(A, POS-set)** - A relação $<$ é uma preferência POS dados duas tuplas $x, y \in R$, onde $x < y$, se e somente se, $x \notin POS\text{-}set$ e $y \in POS\text{-}set$. O conjunto *POS-set* está contido no domínio do atributo A de uma preferência POS que define um conjunto de valores que são melhores do que todos os outros valores do domínio A. Como exemplo, têm-se na Tabela 2.1, Linha 1, as informações de preferências fornecidas pelo usuário e a respectiva especificação dada pelo *framework* de [Kießling 2002] para a preferência do tipo POS(A, POS-set).
2. **NEG(A, NEG-set)** - A relação $<$ é uma preferência NEG dados duas tuplas $x, y \in R$, onde $x < y$, se e somente se, $y \notin NEG\text{-}set$ e $x \in NEG\text{-}set$. O conjunto *NEG-set* de preferências NEG descreve valores não desejados para o atributo A. Como exemplo, têm-se na Tabela 2.1, Linha 2, as informações de preferências fornecidas pelo usuário e a respectiva especificação dada pelo *framework* de [Kießling 2002] para a preferência do tipo NEG(A, NEG-set).
3. **POS/NEG(A, POS-set;NEG-set)** - A relação $<$ é uma preferência POS/NEG dados duas tuplas $x, y \in R$, onde $x < y$, se e somente se, $(x \in NEG\text{-}set \text{ e } y \notin NEG\text{-}set) \vee (x \notin NEG\text{-}set \text{ e } x \notin POS\text{-}set \text{ e } y \in POS\text{-}set)$. As preferências do tipo POS/NEG formam a combinação de preferências positivas e negativas, onde são apresentados os valores de atributos que são melhores que todos os outros (*POS-set*) e os indesejados (*NEG-set*) para o atributo A. Como exemplo, têm-se na Tabela 2.1, Linha 3, as informações de preferências fornecidas pelo usuário e a respectiva especificação dada pelo *framework* de [Kießling 2002] para a preferência do tipo POS/NEG(A, POS-set;NEG-set).
4. **POS/POS(A, POS1-set; POS2-set)** - A relação $<$ é uma preferência POS/POS, dados duas tuplas $x, y \in R$, onde $x < y$ se e somente se, $(x \in POS2\text{-}set \text{ e } y \in POS1\text{-}set) \vee (x \notin POS1\text{-}set \text{ e } x \notin POS2\text{-}set \text{ e } y \in POS2\text{-}set) \vee (x \notin POS1\text{-}set \text{ e } x \notin POS2\text{-}set \text{ e } y \in POS1\text{-}set)$. E em POS/POS tem-se os valores de preferências ótimas em *POS1-set* e os valores alternativos *POS2-set* (os preferidos caso não seja satisfeitas as preferências de *POS1-set*) para o atributo A. Como exemplo, têm-se na Tabela 2.1, Linha 4, as informações de preferências fornecidas pelo usuário e a

respectiva especificação dada pelo *framework* de [Kießling 2002] para a preferência do tipo POS/POS($A, POS1\text{-}set; POS2\text{-}set$).

Linha	Informação sobre as preferências do usuário	Especificação
1	“Para mim, o Gênero Romance é melhor do que qualquer outro Gênero.”	POS(Gênero, Romance);
2	“Não gosto do diretor Steven Spielberg .”	NEG(Diretor, Steven Spielbergr);
3	“Gosto muito dos filmes do diretor Hitchcock e não gosto dos filmes do diretor D. Lynch, acho os temas pouco envolventes.”	POS/NEG(Diretor, Hitchcock; D. Lynch);
4	“Prefiro Romance, mas caso não tenha filmes desse Gênero admito também filmes de Suspense.”	POS/POS(Gênero,Romance; Suspense);

Tabela 2.1: Exemplo de preferências do usuário e respectiva tradução segundo a especificação de [Kießling 2002]

Além dos construtores categóricos são abordados também os construtores numéricos, sendo que tais construtores de preferências incluem: **AROUND**(A, z), **BETWEEN**($A, [low, up]$), **LOWEST**(A) e **HIGHEST**(A). Em um construtor **AROUND** de preferências o valor desejado é z , mas se isto não é possível retorna valores próximos de z . Para o construtor **BETWEEN** os valores dentro do intervalo de valores $[low, up]$ são os ótimos. Para o **LOWEST** (**HIGHEST**) os menores valores para o atributo em questão (ou os maiores no caso do **HIGHEST**) são os melhores.

A composição de preferências básicas formam *os construtores complexos*, sendo eles: o construtor **PARETO** e o construtor **CASCADE**, a seguir tratados e exemplificados.

1. **O operador PARETO** ($P_1 \otimes P_2$) atribui igual importância às preferências utilizadas na composição, ou seja, P_1 e P_2 são consideradas igualmente importantes. Formalmente para um objeto $x = (x_1, x_2)$ ser melhor que um objeto $y = (y_1, y_2)$ segundo a preferência PARETO, dadas duas preferências básicas $P_1 = (A_1, <_{P_1})$ e $P_2 = (A_2, <_{P_2})$, para $x, y \in dom(A_1) \times dom(A_2)$, é possível definir, $x <_{P_1 \otimes P_2} y$ se e somente se:

$$(x_1 <_{P_1} y_1 \wedge (x_2 <_{P_2} y_2 \vee x_2 = y_2)) \vee (x_2 <_{P_2} y_2 \wedge (x_1 <_{P_1} y_1 \vee x_1 = y_1))$$

Sendo assim, a preferência PARETO $<_{P_1 \otimes P_2}$ é uma preferência P sobre os atributos A_1 e A_2 , isto é, $P = (A_1 \cup A_2, <_{P_1 \otimes P_2})$.

Como exemplo temos as preferências $P_1 = \text{AROUND}(\text{Duração}, 3)$ e $P_2 = \text{HIGHEST}(\text{Década})$ de um usuário, especificadas sobre a Tabela 2.2, e desejamos obter a preferência PARETO $P = (\text{Duração} \cup \text{Década}, <_{P_1 \otimes P_2})$. Ao observar o gráfico de preferências denominado BTG (*Better than Graph*) [Agrawal et al. 1989], representado pela Figura 2.1, podemos obter preferências como:

- (a) O filme 01 não é comparável ao filme 02 (o fato de não ser comparável é representado pelo símbolo $=$, logo $01 = 02$), pois segundo a preferência P_1 o filme 01 é considerado o melhor, ou seja, o valor do atributo duração está mais próximo de 3 horas, por outro lado, segundo a preferência P_2 o filme 02 é considerado o melhor, pois a década é maior.
- (b) O filme 03 é melhor que o filme 02 segundo as duas preferências P_1 e P_2 , pois satisfaz em ambas.
- (c) O filme 04 é melhor que o filme 05, apenas pela preferência P_2 , pois pela preferência P_1 eles são incomparáveis tendo em vista que o atributo duração do filme 04 tem o mesmo valor do filme 05.

2. **O operador Priorização ou CASCADE** ($P_1 \& P_2$) atribui um grau de prioridade a uma determinada preferência, ou seja, P_1 é considerada mais importante que P_2 . Formalmente, dados $P_1 = (A_1, <_{P_1})$ e $P_2 = (A_2, <_{P_2})$, para $x, y \in \text{dom}(A_1) \times \text{dom}(A_2)$, é definido que $x <_{P_1 \& P_2} y$ ocorre se somente se:

$$x_1 <_{P_1} y_1 \vee (x_1 = y_1 \wedge x_2 <_{P_2} y_2)$$

Como exemplo, utilizando o banco especificado na Tabela 2.2 e as preferências $P_1 = \text{AROUND}(\text{Duração}, 3)$ e $P_2 = \text{HIGHEST}(\text{Década})$ especificadas pelo usuário onde P_1 é a preferência prioritária, é possível obter o Gráfico de preferências BTG, representado pela Figura 2.2. O operador CASCADE obtido contém as mesmas comparações alcançadas por PARETO, entretanto, solucionando casos em que duas preferências se conflitam, como por exemplo ao comparar o filme 03 ao filme 04, por PARETO temos segundo a preferência P_1 que o filme 03 é o mais preferível e segundo a preferência P_2 que o filme 04 é o mais preferível. Mas com o uso do operador CASCADE é possível dizer que o filme 03 é preferível ao filme 04, tendo em vista que a preferência P_1 tem uma prioridade maior do que a preferência P_2 .

Filme-id	Duração(horas)	Década	Gênero
01	1,5	40	Drama
02	1	50	Comédia
03	3	70	Romance
04	2	80	Drama
05	2	60	Romance
06	3	60	Romance

Tabela 2.2: Banco de dados I.

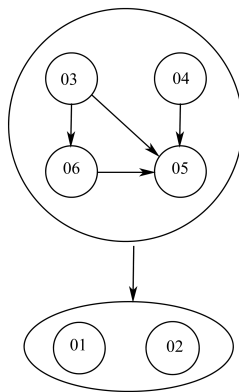


Figura 2.1: Preferência PARETO

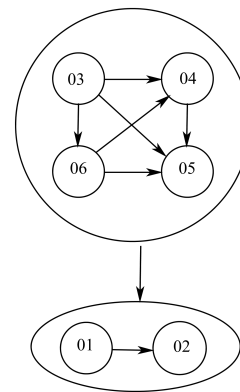


Figura 2.2: Preferência CASCADE

2.1.2 Preferências condicionais

Nesta seção serão apresentados os principais formalismos para representação de preferências condicionais sobre objetos. Ao contrário das preferências não condicionais apresentadas na subseção 2.1.1, nas preferências condicionais o valor de um atributo pode influenciar nas preferências sobre os valores dos demais atributos. Tal assunto têm sido alvo de diversas pesquisas na área de Inteligência Artificial. Alguns dos formalismos propostos são: CP-nets, TCP-nets e linguagem de preferências condicionais.

CP-nets e TCP-nets

Nos trabalhos de [Boutilier et al. 2004a, Boutilier et al. 1999] foi proposto um modelo para representar preferências de forma qualitativa sobre objetos com o uso de grafos, chamado CP-net. A principal vantagem da representação de instruções de preferências qualitativas por meio de CP-nets está na forma compacta com que as preferências são reproduzidas permitindo o uso de dependência preferencial condicional, em que a preferência sobre os valores de um atributo é influenciada pelos valores de outros atributos.

Uma CP-net é um grafo dirigido sobre um conjunto de atributos onde cada nó representa um atributo e as arestas representam as dependências preferenciais condicionais. Quando há uma aresta do atributo X para o atributo Y significa que as preferências sobre os valores do atributo Y são condicionadas pelo valor assumido pelo atributo X . Cada atributo X possui vinculada ao seu respectivo nó da CP-net uma *tabela de preferência condicional* que descreve as preferências sobre o atributo X levando em conta a combinação dos possíveis valores assumidos pelos atributos pais de X . No Exemplo 2.1 a seguir, são expostas algumas preferências e criada uma CP-net (Figura 2.3) para representá-las:

Exemplo 2.1. Suponha as seguintes preferências especificadas por um usuário sobre atributos relacionados a um Banco de dados de Filmes:

1. Filmes cujo Gênero (G) é “Romance” (r) são melhores do que Filmes do Gênero “Suspense” (s).

2. Filmes cuja Década (D) é “60” são melhores do que da Década de “70”.
3. Para Filmes do Gênero “Romance” da Década “60” ou Filmes de “Suspense” da Década de “70” é preferível Filmes do Diretor (Di) “Alfred Hitchcock” (a) a de “Frank Capra” (f). Para Filmes do Gênero “Romance”, mas da Década de “70” ou Filmes de “Suspense” da Década de “60” é preferível Filmes do Diretor “Frank Capra” a do Diretor “Alfred Hitchcock”.

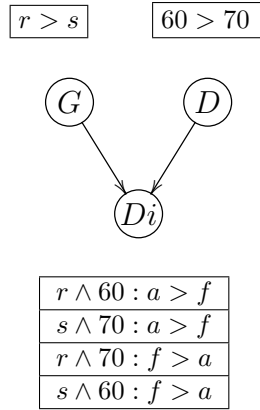


Figura 2.3: CP-net para as preferências sobre filmes

A partir da CP-net da Figura 2.3 é possível fazer as seguintes interpretações: No primeiro vértice G , o Gênero Romance (r) é melhor do que Gênero Suspense (s) e a preferência sobre tal atributo é independente de outros atributos, logo a tabela de preferência referente ao nó G informa que $r > s$; O vértice D também não depende de outros atributos e o usuário diz que Filmes cuja Década é 60 são melhores do que Filmes da Década de 70, logo a tabela de preferência informa somente que $60 > 70$; E para o nó C , têm-se uma preferência condicional, onde os valores para o atributo Diretor (Di) dependem dos valores Gênero e Década do filme, logo têm-se se uma aresta do atributo Gênero para o atributo Diretor e também uma aresta do atributo Década para o atributo Diretor;

Em diversos trabalhos como [Boutilier et al. 1999, Boutilier et al. 2004a, Boutilier et al. 2004b] foram propostos algoritmos que exploram a estrutura das CP-nets para tarefas de otimização e testes de dominância. O trabalho de [Boutilier et al. 1999] propôs um algoritmo de otimização que obtém o objeto ótimo ou o objeto mais preferido, caminhando através da estrutura da CP-net e tomando os melhores valores para os atributos de acordo com suas tabelas de preferências condicionais.

Em [Brafman et al. 2006] é descrita uma extensão da CP-net, chamada de TCP-net. A TCP-net é um grafo que possui as mesmas características de uma CP-net e ainda possui mais dois tipos de arestas, uma para representar *importância absoluta* e outra para representar *importância relativa*.

Importância absoluta é quando, dados dois atributos X e Y preferencialmente independentes, X é mais importante do que Y , denotado por $X \blacktriangleright Y$. Já na importância relativa, um atributo é mais importante do que outro de acordo com os valores de um

terceiro atributo ou um conjunto de atributos. Isto é, dados três atributos X , Y e Z , X é mais importante do que Y dependendo do valor assumido por Z . Uma aresta de importância relativa é uma aresta não direcionada rotulada pelos atributos que condicionam a importância relativa, esta aresta contém o símbolo $\blacksquare$ e possui também uma *tabela de importância condicional* para definir a importância relativa. A tabela de importância condicional descreve a importância relativa de acordo com os atributos que condicionam a importância.

O Exemplo 2.2 exhibe as preferências de um usuário sobre Filmes. A partir delas, é feita a especificação de uma TCP-net que é apresentada na Figura 2.4. A quarta preferência manifestada pelo usuário é uma preferência absoluta.

Exemplo 2.2. Suponha as seguintes preferências especificadas por um usuário sobre atributos relacionados a um Banco de dados de Filmes:

1. Filmes cujo Gênero (G) é “Romance” (r) são melhores do que Filmes do Gênero “Suspense” (s).
2. Filmes cuja Década (D) é “60” são melhores do que da Década de “70”.
3. Para Filmes do Gênero “Romance” da Década “60” ou Filmes de “Suspense” da Década de “70” é preferível Filmes do Diretor (Di) “Alfred Hitchcock” (a) a de “Frank Capra” (f). Para Filmes do Gênero “Romance”, mas da Década de “70” ou Filmes de “Suspense” da Década de “60” é preferível Filmes do Diretor “Frank Capra” a do Diretor “Alfred Hitchcock”.
4. O Gênero é mais importante do que a Década.

A Figura 2.4 exhibe uma TCP-net para as preferências especificadas, onde os nós são as variáveis, as arestas simples são as relações de dependência preferencial, a aresta de G para D indica a relação de importância relativa e as tabelas especificam as preferências sobre a variável.

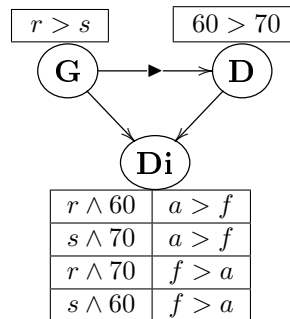


Figura 2.4: TCP-net

Regras de preferências condicionais

Em [Wilson 2004] foi especificada uma linguagem de preferências condicionais $\mathcal{L}$ que permite expressar preferências mais genéricas do que aquelas tratadas pelas CP-nets e TCP-nets.

A representação de preferências nesta linguagem considera um conjunto de atributos $A = \{X_1, \dots, X_n\}$, onde $\mathbf{Dom}(X_i)$ denota o domínio de X_i , ou seja, os possíveis valores que podem ser assumidos por X_i . O conjunto $\mathbf{Dom}(X_1) \times \dots \times \mathbf{Dom}(X_n)$ é denotado por $\mathcal{O}$ e representa o conjunto de objetos formados por todas as combinações possíveis de valores para os atributos de A . Se $o = (x_1, \dots, x_n)$ é um objeto então $o[X_i] = x$ denota valor x para o atributo X_i no objeto o .

A representação de preferências na linguagem $\mathcal{L}$ é realizada através de *regras de preferências condicionais*. Uma regra de preferência condicional é uma instrução no formato $\varphi : u \rightarrow (X = x) > (X = x')[W]$. A condição u é uma fórmula no formato $(X_{i_1} = x_1) \wedge \dots \wedge (X_{i_k} = x_k)$ onde $X_{i_j} \in A$ e $x_j \in \mathbf{Dom}(X_{i_j})$ para todo $j \in \{1, \dots, k\}$. Sendo que $W \subseteq (A - \{X, X_{i_1}, \dots, X_{i_k}\})$ é o conjunto de atributos desconsiderados pela regra de preferência.

Quando se têm um conjunto finito de regras de preferências condicionais, diz-se tratar uma *Teoria de Preferência Condicional*.

O Exemplo 2.3 a seguir mostra uma Teoria de Preferência Condicional sobre um conjunto de objetos.

Exemplo 2.3. Suponha as preferências a seguir, em que o conjunto de atributos em questão é $A = \{G, D, Di\}$, quanto aos domínios é assumido que $\mathbf{Dom}(G) = \{r, s\}$, $\mathbf{Dom}(D) = \{60, 70\}$ e $\mathbf{Dom}(Di) = \{a, f\}$:

1. *Filmes cujo Gênero (G) é “Romance” (r) são melhores do que Filmes do Gênero “Suspense” (s).*
2. *Filmes cuja Década (D) é “60” são melhores do que da Década de “70”.*
3. *Para Filmes do Gênero “Romance” da Década “60” ou Filmes de “Suspense” da Década de “70” é preferível Filmes do Diretor (Di) “Alfred Hitchcock” (a) a de “Frank Capra” (f). Para Filmes do Gênero “Romance”, mas da Década de “70” ou Filmes de “Suspense” da Década de “60” é preferível Filmes do Diretor “Frank Capra” a do Diretor “Alfred Hitchcock”.*
4. *O Gênero é mais importante do que a Década.*

As preferências do Exemplo 2.3 podem ser expressadas pela Teoria de Preferência condicional composta pelas seguintes regras:

- $\varphi_1 : (G = r) > (G = s)[D]$
- $\varphi_2 : (D = 60) > (D = 70)$

- $\varphi_3 : (G = r) \wedge (D = 60) \rightarrow (Di = a) > (Di = f)$
- $\varphi_4 : (G = s) \wedge (D = 70) \rightarrow (Di = a) > (Di = f)$
- $\varphi_5 : (G = r) \wedge (D = 70) \rightarrow (Di = f) > (Di = a)$
- $\varphi_6 : (G = s) \wedge (D = 60) \rightarrow (Di = f) > (Di = a)$

A Teoria de Preferência condicional apresentada anteriormente é equivalente a TCP-net da Figura 2.4. Outra importante contribuição do trabalho de [Wilson 2004] foi a especificação de um método para inferir os objetos ótimos a partir de uma Teoria de Preferência Condicional. O método consiste em: dada uma Teoria de Preferência Condicional $\Gamma = \{\varphi_1, \varphi_2, \dots, \varphi_n\}$, seguir os passos:

1. Obtém-se todos os pares de tuplas que pode-se ordenar diretamente utilizando cada uma das regras φ_i , onde $\{i = 1, \dots, n\}$;
2. Faz-se a união dos conjuntos de pares inferidos por cada regra φ_i ;
3. Inclui todos os pares de tuplas obtidos por transitividade.

Para exemplificar a inferência de objetos ótimos a partir de uma Teoria de Preferência Condicional utilizaremos a Teoria Γ , obtida a partir do Exemplo 2.3 e o Banco de Dados da Tabela 2.3.

Filme-id	Gênero	Década	Diretor
01	r	60	a
02	r	70	a
03	r	70	f
04	r	60	f
05	s	60	a
06	s	70	f

Tabela 2.3: Banco de dados II.

Considerando cada uma das regras de preferência condicional estabelecidas pela Teoria Γ , é possível obter uma ordem dada pela união dos conjuntos de ordens obtidos:

1. Pela regra φ_1 temos as seguintes ordens: $01 > 05, 02 > 05, 03 > 06, 04 > 06$
2. Pela regra φ_2 temos a seguinte ordem: $01 > 02$
3. Pela regra φ_3 temos a seguinte ordem: $01 > 04$
4. A regra φ_4 não nos permite comparar quaisquer duas tuplas.
5. Pela regra φ_5 temos a seguinte ordem: $03 > 02$
6. A regra φ_6 não nos permite comparar quaisquer duas tuplas.
7. Por transitividade temos que:

- (a) $03 > 02$, $02 > 05$ então $03 > 05$;
(b) $01 > 04$, $04 > 06$ então $01 > 06$.

A Figura 2.5, apresenta o grafo BTG, o qual demonstra a ordem de preferência induzida pela Teoria Γ . As flechas pontilhadas indicam comparações por transitividade, e as flechas não pontilhadas indicam comparações feitas pelas regras φ_i , onde $i = 1, \dots, 6$ da Teoria Γ . Como foi apresentado, no trabalho de [Wilson 2004] é abordado um forma-

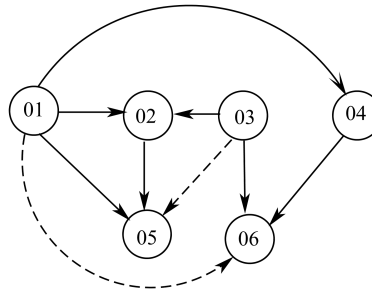


Figura 2.5: Ordem de Preferência induzida pela Teoria Γ de Preferência Condicional

lismo para especificação das preferências do usuário por meio de regras de preferências condicionais e a partir das regras obtidas é proposto um método para inferir uma ordem sobre os objetos do banco de dados. A contribuição dada pelo trabalho de [Wilson 2004] é melhor explicitada na Figura 2.6.

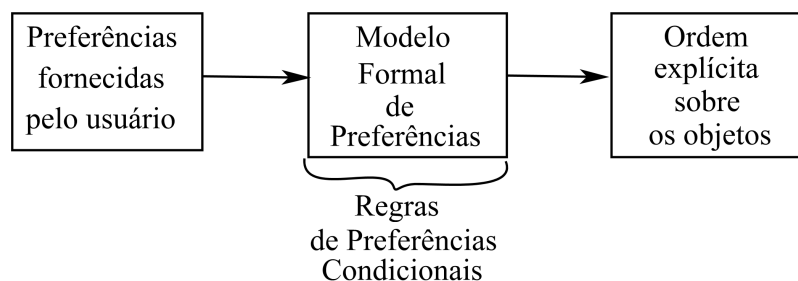


Figura 2.6: Contribuição dada pelo trabalho de [Wilson 2004]

2.2 Metodologia para aprendizaje de redes bayesianas

As *redes bayesianas* foram desenvolvidas no início dos anos 80 para facilitar a tarefa de predição em sistemas de inteligência artificial [Pearl 1988]. Em resumo, redes bayesianas também conhecidas como *redes de opinião*, *redes causais*, *redes de crença*, *gráficos de dependência probabilística*, são modelos gráficos para *raciocínio* (conclusões) *baseado na incerteza*. As redes bayesianas são constituídas de duas componentes: *um grafo direcionado acíclico* (DAG) e *tabelas de probabilidades condicionais* (TPC). No (DAG), os

vértices representam os atributos do domínio em questão e *os arcos representam a dependência direta entre eles*. Cada vértice possui uma tabela de probabilidade condicional associada que quantifica os efeitos que os pais têm sobre ele.

Originalmente o conceito das redes bayesianas foi desenvolvido supondo-se uma dependência de especialistas humanos para a definição do grafo direcionado e das tabelas de probabilidades condicionais [Neapolitan 1990], entretanto, a necessidade de automação de tal processo fez com que pesquisadores buscassem encontrar métodos de aprendizado das redes bayesianas a partir de bancos de dados sem (ou com pouca) intervenção humana. A construção manual de uma rede bayesiana pode ser um processo bastante trabalhoso e caro para grandes aplicações, e em domínios complexos sua especificação além de consumir bastante tempo está propensa a erros.

O *aprender* pode ser interpretado como o processo de aquisição de uma representação interna efetiva para as restrições existentes no mundo, fatos e regras [Pearl 1988] ou, em outras palavras, a aquisição de conceitos e de conhecimentos estruturados.

O *aprendizado de redes bayesianas* consiste em *induzir*, a partir de um conjunto de dados, as distribuições de probabilidades simples e condicionais e/ou identificar as relações de interdependência entre as variáveis de um domínio de dados, que se constitui na população de interesse. Esse *processo de aprendizado indutivo* pode ser de *dois tipos*: *aprendizado de estrutura*, quando não se tem a estrutura e *aprendizado de parâmetros* quando não se tem as TPC's de cada atributo envolvido no problema. Nas seções seguintes abordaremos, de maneira sucinta, cada um dos aprendizados relatados anteriormente no contexto de redes bayesianas.

2.2.1 Aprendizado de parâmetros

Seja $M=(S,\theta)$ uma rede bayesiana com estrutura S e parâmetros θ , e U o conjunto dos atributos em M . Além disso, seja D um banco de dados de casos, onde cada caso é a configuração de todas as variáveis em U . Cada caso é dito ser um *caso completo*, isto é, *todos os atributos possuem valores não vazios*. Na comunidade de aprendizado, um parâmetro é tipicamente denotado por θ . Sendo assim, para garantir que os parâmetros podem ser aprendidos independentemente são admitidas duas hipóteses [Jensen e Nielsen 2007]:

1. Independência Global: supõe-se que os parâmetros dos vários atributos são independentes, isto é, as tabelas de probabilidades dos diferentes atributos são independentes sendo possível alterá-las independentemente.
2. Independência Local: os parâmetros dentro de cada tabela são independentes no que diz respeito aos valores dos pais dos atributos. Para ser mais preciso, dadas duas configurações:

- (a) $A=a, B=b$;
- (b) $A=a', B=b'$;

A probabilidade do evento $C = c$ dado que $A = a$ e $B = b$ (representada por $P[c|a, b]$) não tem nenhuma influência sobre a probabilidade do evento $C = c$ dado que $A = a'$ e $B = b'$ (representada por $P[c|a', b']$), tais eventos são independentes. Enquanto que se alterássemos a probabilidade $P[c|a', b']$ ela influenciaria a probabilidade $P[c'|a', b']$. Isto é, pode-se modificar os valores das probabilidades condicionais de forma independente para diferentes valores dos pais ($P[c|a, b]$, sendo que tal fato é independente do valor de $P[c|a', b']$).

Para cada caso $d \in$ ao banco de dados D , podemos calcular a probabilidade conjunta $P(d|M)$, obtida aplicando-se as probabilidades condicionais de M . A partir dessa probabilidade é possível definir o princípio da *verossimilhança máxima* que mede o quanto o modelo M se “conforma” aos dados (estatisticamente reflete os dados em questão).

O Princípio da verossimilhança máxima (PVC) ou *Maximum likelihood*

Para cada caso $d \in D$, podemos calcular a probabilidade conjunta $P(d|M)$, obtida aplicando-se as probabilidades condicionais de M . Define-se a *verossimilhança de M dado D* como sendo $L(M|D) =$ Produto de todas as $P(d|M)$ (ou $\log_2$), ou seja:

$$L(M|D) = \prod_{d \in D} P(d|M). \quad (2.1)$$

Frequentemente é usado o logaritmo no cálculo $L(M|D)$:

$$LL(M|D) = \prod_{d \in D} \log_2 P(d|M). \quad (2.2)$$

Este princípio mede o quanto o modelo M se “conforma” aos dados D , isto é, o quanto o modelo M reflete de fato a distribuição probabilística dos dados D . Se as tabelas de probabilidades condicionais θ não forem apropriadas para os dados em questão, a verossimilhança terá um valor bem baixo (tendendo a zero).

Tendo em vista a necessidade de redes bayesianas que se conformem ao máximo com os dados, é almejado tabelas de probabilidade θ que *maximizem* a verossimilhança do modelo com respeito aos dados.

O princípio da verossimilhança máxima diz que a verossimilhança máxima é atingida quando o vetor de probabilidades θ correspondente às probabilidades atingidas é obtido através de contagens, isto é, a verossimilhança é máxima quando as tabelas de probabilidades são obtidas através da frequência dos eventos, por exemplo para o cálculo da

probabilidade $P(A = a|B = b, C = c)$ é necessário calcular:

$$\frac{N(A = a, B = b, C = c)}{N(B = b, C = c)},$$

onde $N(A = a, B = b, C = c)$ é o número de casos no banco de dados para $A = a, B = b, C = c$.

2.2.2 Aprendizado de estrutura

Existem basicamente *dois enfoques* para aprendizado da topologia de redes bayesianas clássicas: o enfoque baseado em *restrições* e o enfoque baseado em *Score*.

Enfoque baseado em restrições

O aprendizado da estrutura de uma rede bayesiana sobre o enfoque baseado em restrições é também conhecido como aprendizado de estrutura através da *análise de dependência* ou *independência condicional*. Nesta abordagem, procura-se uma rede que represente da melhor maneira possível a distribuição conjunta que surge a partir do banco de dados. É fundamental que esta rede represente todas as relações de independência e dependência da distribuição conjunta induzida pelos dados.

Seja $M(S, \theta)$ uma rede bayesiana qualquer. A independência condicional na distribuição de probabilidade representada por M é codificada na estrutura S desta rede, podendo ser encontrada a partir de tal estrutura com a utilização do critério *d-separação* [Pearl 1988].

Duas variáveis A e B em uma rede qualquer $M(S, \theta)$ são *d-separadas* (“d” para “grafo direcionado”- *directed graph*) se para todos os caminhos entre A e B existe uma variável intermediária V (diferente de A e B) tal que:

- A conexão entre A e B é *serial* e o valor de V é conhecido.
- A conexão entre A e B é *divergente* (são filhos de V) e o valor de V é conhecido.
- A conexão entre A e B é *convergente* (são pais de um mesmo filho V) e o valor de V é desconhecido.

Considerando os tipos de conexões serial, divergente e convergente da Figura 2.7, pode-se tirar as seguintes conclusões: a Figura 2.7(a) mostra que em uma conexão serial qualquer evidência (se a variável é conhecida) sobre A influencia B que, por sua vez, afeta C . Isto é, se o estado de B é conhecido, então o canal é bloqueado e A e C se tornam independentes. Então A e C são *independentes* dado B .

Nos grafos divergentes (ver Figura 2.7(b)), a influência pode ser transmitida entre todos os filhos de A , a menos que o estado de A seja conhecido. Dizemos que $B, C, \dots, G$ são *d-separadas* se o estado de A é conhecido.

Nos grafos convergentes (Figura 2.7(c)), se nada se sabe sobre A, exceto o que pode ser inferido a partir de B, C,...,G, então os pais de A são independentes: a evidência sobre um deles não influi sobre a certeza dos demais. A evidência pode ser transmitida através de uma conexão convergente se a variável da conexão ou um de seus descendentes recebeu evidência.

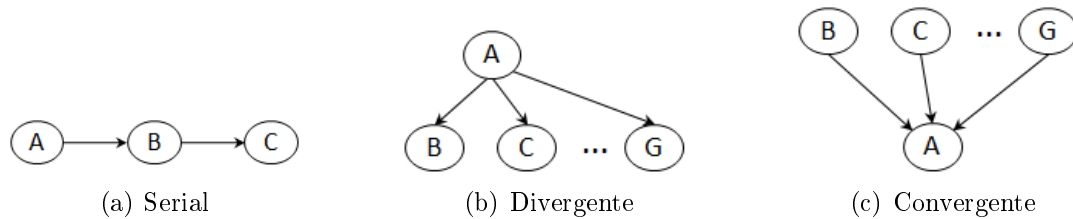


Figura 2.7: Tipos de Conexões

Os três casos descritos na Figura 2.7 ilustram como evidências podem ser transmitidas através de uma variável, e seguindo as regras é possível decidir para qualquer par de variáveis em uma rede causal se elas são dependentes dadas as evidências conhecidas ¹. Isto é, se A e B são *d-separadas*, então alterações nas evidências de A não tem impacto sobre B.

Desta forma, se duas variáveis são *d-separadas* são consequentemente condicionalmente independentes, o que justifica no modelo final não haver uma aresta entre essas duas variáveis, e nas possíveis estruturas que contêm esta aresta, ela poder ser excluída a partir dessa consideração.

Enfoque baseado em *scores*

Os métodos de aprendizado de topologias de redes bayesianas baseados em *Score* consistem em associar a cada possível topologia um número (*Score*) que mede a compatibilidade da estrutura com relação aos dados. A *topologia com maior score é a almejada*. Assim, os métodos de aprendizado baseados em *Score* têm duas componentes:

1. Uma função *Score*;
2. Um método de busca, que procura pelo máximo valor da função *Score*. Tal método deve ser computacionalmente tratável e deve produzir resultados satisfatórios.

Os algoritmos com este enfoque geralmente iniciam com uma estrutura de grafo sem arcos e fazem uso de algum método de busca para incluir um arco no grafo. Em seguida, usam um método de pontuação (cálculo de *Score*) para definir se a nova estrutura é melhor do que a antiga. Caso a nova estrutura seja melhor, o novo arco inserido é mantido, e o algoritmo tenta inserir outro. Este processo se repete até que nenhuma estrutura nova seja melhor do que a estrutura atual.

¹As evidências são obtidas a partir dos dados de entrada

A função *score* utilizada para avaliar a nova estrutura pode levar em consideração a complexidade da rede ou não. A seguir detalhamos cada uma das abordagens.

Enfoque baseado em *score* levando em consideração a complexidade da rede

A função *Score* pode levar em consideração tanto a compatibilidade da estrutura em relação aos dados quanto a complexidade da estrutura corrente no momento da busca pela melhor topologia [Jensen e Nielsen 2007].

A complexidade de uma rede bayesiana pode ser medida em número de *links* (arestas do grafo direcionado acíclico) ou número de parâmetros independentes, sendo que, estas duas medidas estão relacionadas. O número de parâmetros independentes é o número de parâmetros de probabilidades existentes na rede. A complexidade da rede M é dito o tamanho da rede ($Size(M)$) dado na Equação 2.3.

Seja M uma rede bayesiana com os atributos descritos no conjunto U e assuma que os parâmetros θ_M são localmente e globalmente independentes, então, o número de parâmetros independentes (ou o tamanho de M) é dado por:

$$Size(M) = \sum_{X \in U} |pa(X)| \cdot (|dom(X)| - 1), \quad (2.3)$$

onde $dom(X)$ é o domínio do atributo X em questão.

Observe que a complexidade da rede ($Size(M)$) depende somente da estrutura da rede e do domínio dos atributos, não tendo qualquer relação com o banco de dados.

Por exemplo, assumindo que todas as variáveis são binárias, então o tamanho do modelo na Figura 2.8 é $1 + 2 + 1 + 4 + 2 = 10$.

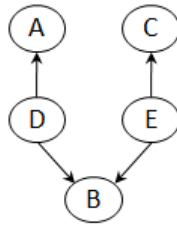


Figura 2.8: Exemplo de uma estrutura bayesiana

A partir de um princípio geral de indução de aprendizado levado em consideração no aprendizado de estruturas em redes bayesianas, o *Princípio de Ockham's Razor* (tal princípio recomenda escolher as hipóteses mais simples (de menor complexidade) coerentes com as observações) é possível concluir que as estruturas com menor complexidade são as que melhor representam os dados. Desta forma em [Jensen e Nielsen 2007] são apresentadas algumas conclusões importantes, descritas a seguir.

- Seja M uma rede Bayesiana cujos seus vértices são atributos $\in U = X_1, \dots, X_n$. A partir de M podemos calcular as probabilidades conjuntas $P[X_1 = x_1, \dots, X_n = x_n]$

para todos os valores do domínio $dom(X_i)$ dos atributos X_i , onde $i \in 1, \dots, n$. Estas probabilidades são resumidas na notação $P_M(U)$, onde U é o conjunto dos atributos de M . As redes de menor complexidade e que refletem a distribuição probabilística dos dados a partir dos quais foram obtidos com o maior *score* possível são as redes que possuem todos seus *links essenciais*. Dizemos que um *link* de N é *essencial* se quando o retiramos, a rede obtida altera as probabilidades $P_M(U)$. Dizemos que uma rede bayesiana T *representa* (reflete a distribuição probabilística dos dados) $P_M(U)$ se $P_T(U) = P_M(U)$.

- Seja M uma rede bayesiana que só contém *links* essenciais. Seja T uma outra rede, tal que $P_M(U) = P_T(U)$. Então M tem o mesmo número de *links* que T e $Size(M) = Size(T)$.

A partir de tais fatos podemos chegar a seguinte conclusão: suponha que tenhamos um conjunto de amostras D e o objetivo é encontrar uma rede bayesiana que melhor se conforme à distribuição probabilística dos dados em D . Se estes dados correspondem a uma rede bayesiana M com todos os *links* essenciais, então M seria o modelo *minimal* com respeito ao tamanho e ao número de *links*, portanto a menos complexa. Sendo assim, o objetivo é encontrar uma boa função *score* que avalie tanto a compatibilidade da topologia (quanto maior melhor) quanto a sua complexidade (quanto menor melhor).

Enfoque baseado em *score* não levando em consideração a complexidade da rede

Quando não se considera a complexidade da rede é levado em consideração somente a adequação da rede aos dados. Os métodos para construção de uma topologia de redes bayesianas tratados em [Cooper e Dietterich 1992], [Heckerman et al. 1994], [Cheng et al. 1997], [David et al. 1997], [Buntine 1996], [Larrañaga et al. 1994], [Bouckaert et al. 1993], [Lam e Bacchus 1992], [Larrañaga et al. 1997], [Habrant 1999] se enquadram nessa categoria. Abordaremos resumidamente um dos mais referenciados métodos neste contexto, o algoritmo *K2* para aprendizado de topologias em redes bayesianas.

2.2.3 O algoritmo *K2*

O algoritmo *K2* é um algoritmo de aprendizagem de estrutura mais representativo entre os baseados em *score* e se tornou bastante popular devido aos resultados obtidos quando aplicado ao conjunto de dados da rede ALARM, um *benchmark* amplamente aceito para os algoritmos de aprendizagem bayesiana [Cooper e Dietterich 1992].

O algoritmo *K2* avalia as possíveis topologias de uma rede bayesiana calculando a probabilidade dessa topologia refletir a base de dados em questão. O algoritmo começa assumindo que um vértice não tem antecessores e incrementa o número de antecessores

adicionando o antecessor que resulta no maior aumento de probabilidade da estrutura refletir estatisticamente a base de dados. Quando a adição de mais um antecessor ao vértice não aumenta mais a probabilidade, o vértice pára de receber antecessores e o algoritmo faz o mesmo para o nó seguinte.

Este algoritmo requer uma ordenação das variáveis e faz uso de um método de pontuação bayesiana para alcançar seu objetivo, que é encontrar a estrutura de rede bayesiana mais provável, dado um conjunto de dados $\mathcal{D}$. A pontuação bayesiana diz respeito à probabilidade da estrutura refletir o banco de dados em questão calculado por meio de uma função de avaliação disponível detalhada em [Cooper e Dietterich 1992].

Além da ordem sobre as variáveis o algoritmo *K2* recebe como entrada o parâmetro u (*upper bound*), um limite superior ao número de pais que um nó pode ter. Em se tratando de um grande número de atributos tal parâmetro é justificável na redução da complexidade dos cálculos.

Os métodos para aprendizagem de rede de preferência apresentados nesta dissertação são baseados no algoritmo *K2*.

2.3 Considerações finais

Neste capítulo foram apresentados diversos conceitos e propriedades preliminares relacionados à preferências sobre objetos e aprendizagem de redes bayesianas. Foram enfatizados os principais trabalhos envolvendo formalismos para representação de preferências condicionais e não condicionais. Tanto os conceitos apresentados quanto os trabalhos abordados são essencialmente importantes pois serão utilizados e referenciados em diversos momentos neste trabalho.

Tendo em vista que o método proposto neste trabalho utiliza o *framework* das redes bayesianas, fez-se importante abordar as suas principais características, principalmente no que tange ao seu aprendizado. Destinamos este capítulo a abordar as principais formas de aprendizado da topologia e parâmetros de uma rede bayesiana, uma vez que o aprendizado de uma rede de preferências também usará tais técnicas.

Capítulo 3

Trabalhos Correlatos em Mineração de Preferências

Os trabalhos em *preference mining* normalmente partem do pressuposto que os usuários fornecem algumas informações iniciais sobre as suas preferências. Tais informações funcionarão como *entrada* para os algoritmos de mineração de preferência. A seguir algumas maneiras do usuário fornecer informações sobre suas preferências:

1. Com base em uma lista fornecida ao usuário, o mesmo seleciona os objetos mais preferidos, como em [Jiang et al. 2008], [Holland et al. 2003].
2. Com base em uma lista fornecida ao usuário, o mesmo seleciona para cada objeto um número ≥ 0 representando o nível de preferência deste objeto. Este número é chamado *ranking* do objeto. Esta abordagem foi utilizada em [Crammer e Singer 2001].
3. Uma terceira forma é fornecer ao usuário pares de objetos aleatórios e o usuário identifica em cada par de objetos qual deles é o preferido [Cohen et al. 1998].

As informações sobre as preferências fornecidas como entrada podem ser oriundas de *um único* usuário ou de um *grupo* de usuários.

Com relação a saída dos algoritmos de mineração de preferência é possível dizer que podem fornecer:

1. *Regras de preferências*, como as fornecidas pelo *framework* de [Kießling 2002]. A partir das Regras de preferência obtidas é possível *inferir uma ordem* sobre os objetos do banco de dados.
2. *Uma Regra de predição ou Função de Predição*, a qual atribui a cada objeto um valor, isto é, *fornece uma ordem explícita* diretamente sobre os objetos.

A seguir, estaremos classificando os trabalhos relacionados a mineração de preferências com base na saída do algoritmo, isto é, trabalhos que propõem algoritmos que fornecem

como saída regras de preferências, serão classificados como trabalhos em *Mineração de Preferências Qualitativas*, os trabalhos que fornecem como saída regras de predição, ou seja, uma ordem explícita sobre os objetos, serão classificados em *Mineração de Preferências Quantitativas*.

3.1 Mineração de preferências qualitativas

Em [Holland et al. 2003] são apresentadas técnicas de mineração nas quais as preferências obtidas são do tipo *qualitativas não condicionais* e baseadas no *framework* de [Kiefling 2002]. Os algoritmos de mineração propostos têm como entrada *logs* de dados, ou seja, dados gerados pelo servidor quando o usuário está acessando um site. Os *logs* de dados possuem os objetos preferidos definidos pelo cliente, ou seja, a entrada fornecida pelo usuário são seus objetos preferidos.

Os *logs* de dados são arquivos contendo registros de transações do usuário e podem ter várias fontes como servidor web, arquivos de *log* ou transações nos servidores de aplicações. Esses arquivos podem cumprir formatos padronizados como o *Common Log File Format*, e incluem dados de identificação do cliente (IP, nome, *login*), a hora atual e o URL (*Uniform Resource Locator*) que está visitando. Informações valiosas sobre os desejos de um usuário são armazenadas a partir da URL, pois contém não apenas o endereço solicitado, mas também palavras-chave ou propriedades dos produtos preferidos [Holland et al. 2003].

Para detectar as preferências o autor faz algumas suposições, tais como a suposição do mundo fechado ou *Closed Word Assumption(CWA)*, tal suposição diz que o cliente conhece todas as possibilidades de um atributo, ou seja, ele conhece todo o domínio de um dado atributo.

O CWA é requerido para detecção de preferências negativas, uma vez que somente se o usuário conhece todos os valores existentes é possível assumir como valores não desejados os valores que nunca foram selecionados, ou seja, não constam no *log* de dados.

Definição 3.1. (Frequência de um valor.) Dado A sendo um atributo de uma relação de *logs* R e $x \in \text{dom}(A)$. O número de entradas de x em $R(A)$ é chamado de frequência de x ou $\text{freq}A(x)$.

O autor propõe um algoritmo para mineração de preferências de dados categóricos, um algoritmo para mineração de preferências em dados numéricos, todos utilizando os conceitos de CWA e de frequência de um valor ($\text{freq}A(x)$), e ainda algoritmos para mineração de preferências complexas onde se faz a composição das mesmas por meio dos operadores PARETO e CASCADE definidos em [Kiefling 2002].

Uma outra abordagem de Mineração de Preferências é apresentada em [Jiang et al. 2008], onde os autores propõem utilizar de exemplos de preferências do usuário denominados superiores e inferiores.

Os exemplos superiores são exemplos de objetos mais preferidos definidos pelo próprio usuário com base em uma lista (uma amostra de dados, não necessariamente composta pelo banco de dados por completo) e os exemplos inferiores são exemplos de objetos menos preferidos também definidos pelo próprio usuário.

Os autores dessa última abordagem partem de alguns pressupostos, como o fato das preferências sobre atributos numéricos serem generalizadas a todos os usuários, por exemplo: “Prefiro os filmes com uma duração maior”, ou ainda, “Prefiro os filmes que são mais recentes”, tais atributos são denominados **Atributos Determinados**, enquanto os atributos Categóricos são chamados **Atributos Indeterminados**. Partindo dessa justificativa eles propõem algoritmos para mineração de preferências em dados categóricos, tendo em vista que preferências obtidas a partir desse tipo de dado não podem ser generalizadas, e necessitam de alguma entrada para identificar a preferência individualmente, por usuário. Para entender como o método funciona, é apresentado na Tabela 3.1 alguns exemplos de preferências, onde um usuário especifica quais são superiores e quais são inferiores.

	Atrib. Determinados		Atrib. Indeterminados		
Filme-id	Duração(horas)	Década	Gênero	Diretor	Label
01	1,2	1960	Romance	Steven Spielberg	
02	1	1990	Romance	D. Lynch	Superior
03	1,5	1980	Suspense	Hitchcock	Inferior
04	2,2	1960	Drama	James Cameron	
05	2	1990	Romance	Hitchcock	
06	2,2	1960	Romance	D.Lynch	
07	1,3	1970	Drama	Mel Gibson	Superior
08	2,5	1640	Drama	Hitchcock	
09	3	1990	Drama	F.Capra	Inferior
10	2	1940	Drama	Akira Kurosawa	

Tabela 3.1: Exemplos de filmes onde o usuário especifica quais são inferiores e superiores.

Considerando que os filmes mais preferidos, em relação aos atributos determinados, são os que possuem a menor duração e são mais recentes, é possível dizer que o Filme 02 é o mais preferido para todos os usuários. Entretanto, considerando os atributos Indeterminados e os exemplos superiores é possível obter as seguintes preferências: D.Lynch > Steven Spielberg, D.Lynch > Hitchcock, Steven Spielberg > Hitchcock, Romance > Suspense, Drama > Suspense, dentre outras regras de preferências possíveis a esse conjunto de exemplos. É feita uma verificação na consistência das preferências obtidas, ou seja se não ocorre alguma regra do tipo Hitchcock > Akira Kurosawa e Akira Kurosawa > Hitchcock o que quer dizer que as preferências estão inconsistentes. De posse das regras bem definidas é feito a ordenação segundo tais regras no Banco de Dados completo.

No trabalho descrito anteriormente o autor minera preferências baseando-se no *framework* de [Kießling 2002], as quais são do tipo *não condicionais*.

3.1.1 Aprendizado de preferências qualitativas utilizando redes bayesianas

Nesta seção é apresentado com mais detalhes o trabalho de [Radde e Freitag 2010], um trabalho bastante recente o qual evidencia um método de elicitación de preferências que também utiliza o *framework* de redes bayesianas para tal objetivo. A seguir falaremos brevemente de tal trabalho, comparando-o à nossa abordagem.

Em [Radde e Freitag 2010] é apresentado um método de elicitación de preferências baseado em conhecimento que não requer um *rankeamento* prévio de objetos do banco de dados. Em tal método é assumido que o usuário não estará interagindo com o sistema com muita frequência, o que portanto inviabiliza a construção de um modelo de usuário baseado em históricos de compra ou acessos. Em vez disso é construído uma rede bayesiana padrão a partir do banco de dados do produto em questão (não *rankeado* pelo usuário, isto é, sem nenhuma preferência estabelecida por nenhum usuário).

O principal objetivo da abordagem proposta em [Radde e Freitag 2010] é inferir conhecimento sobre *dados técnicos* de um certo produto. Dados técnicos são propriedades sobre determinados produtos complexos (como celulares e aparelhos eletrônicos em geral), onde a maioria dos usuários não é capaz de manifestar suas preferências. São exemplos de atributos técnicos *bluetooth*, *wi-fi*, *UMTS* (acrônimo de *Universal Mobile Telecommunication System*) no contexto de celulares.

Como é feita a inferência dos atributos técnicos? De posse da rede bayesiana previamente construída, são feitas perguntas ao usuário, o autor chama tais perguntas de “soft” (Ver Figura 3.1), isto é, perguntas que usuário é capaz de responder e expressar seus desejos, como : “Você aprecia se seu celular é fácil de usar?”, “ Você usará seu novo celular para exibir conteúdo multimídia?”, “Você usará seu celular para ouvir música?”, “Você usará seu celular para acessar a internet?”, entre outras. A partir das respostas a tais perguntas (as respostas são valores discretos), isto é, das *evidências*, são inferidas as respostas para os atributos técnicos.

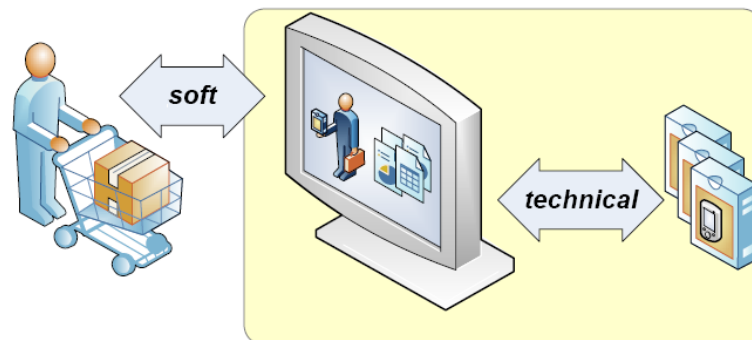


Figura 3.1: Preferências sobre atributos “soft” derivam preferências sobre atributos ditos técnicos (*technical*). Figura extraída de [Radde e Freitag 2010]

Os autores não enfatizam a construção da rede bayesiana (aprendizado da topologia e tabelas de probabilidade), deixando explícito apenas que a mesma foi construída com auxílio do software GeNIe (disponível em <http://dsl.sis.pitt.edu/>) a partir dos dados disponíveis sobre o produto objeto da recomendação, no caso do artigo, celulares. Tal fato é justificável, uma vez que a construção do modelo pode ser feito segundo os padrões já conhecidos para construção de redes bayesianas, sem a necessidade de alteração nesse processo.

As preferências obtidas em [Radde e Freitag 2010] são qualitativas e exploram a propriedade condicional *apenas no processo de inferência dos atributos técnicos*. Algumas diferenças da abordagem de [Radde e Freitag 2010] em relação à nossa abordagem podem ser citadas:

- Estendemos o *framework* de redes bayesianas para explicitar preferências em relação a todos os atributos do banco de dados de amostras de preferências enquanto a abordagem proposta em [Radde e Freitag 2010] infere preferências apenas sobre atributos técnicos.
- A rede bayesiana de preferência proposta por nós é construída com base em um histórico de preferências explicitadas inicialmente pelo usuário enquanto que o modelo proposto por [Radde e Freitag 2010] propõe a construção de uma rede bayesiana padrão e única para todos os usuários.
- Propomos um método de inferência de preferências condicionais enquanto que em [Radde e Freitag 2010] é proposto um método de inferência de atributos técnicos.
- Em [Radde e Freitag 2010] o processo de inferência é *on-line* enquanto neste trabalho propomos um método de inferência baseado no histórico de acessos onde o usuário manifestou suas preferências. Deste modo, na nossa abordagem é construído um modelo de usuário, enquanto em [Radde e Freitag 2010] não, o processo de inferência, como dito anteriormente, é *on-line* a partir de preferências sobre atributos “soft” e uma rede bayesiana previamente construída.
- Por fim, em [Radde e Freitag 2010] as preferências obtidas são utilizadas para formular uma consulta utilizando a linguagem de consulta com incorporação de operadores de preferências *Preference SQL*, definida em [Kießling e Köstler 2002] para retornar os objetos mais preferidos. Na nossa abordagem de posse do modelo de preferências do usuário é possível inferir uma ordem parcial sobre os objetos do domínio do banco de dados.

3.2 Mineração de preferências quantitativas

O trabalho de [Crammer e Singer 2001] aborda mineração de preferências de maneira quantitativa, onde o principal objetivo é encontrar uma regra de predição, de maneira

automática, que atribui a cada instância ou objeto do banco de dados um valor, o qual é tão perto quanto possível do *ranking* real da tupla. Como entrada para os algoritmos propostos por [Crammer e Singer 2001], é solicitado ao usuário, com base em uma lista, que selecione para cada objeto um número ≥ 0 representando o nível de preferência deste objeto. Para a obtenção da função de *ranking*, os autores utilizam de técnicas de classificação (Máquinas de Suporte Vetorial e Redes Neurais) adaptadas para o problema de *rankeamento* de tuplas. Entretanto, as regras de *rankings* obtidos com os algoritmos propostos pelo autor não permitem estabelecer preferências condicionais, somente permitem fazer comparações simples entre os objetos, ordenando-os de acordo com a preferência numérica obtida pela regra de predição.

Outro trabalho que também aborda Mineração de Preferências de maneira quantitativa está expresso em [Cohen et al. 1998], onde o objetivo é obter uma função de preferência a partir de pares de objetos ordenados fornecidos pelo usuário. De posse da função de preferência é possível obter uma ordem para todos os objetos do Banco de Dados. A solução proposta é baseada em aprendizagem *on-line*, mais precisamente no algoritmo de *Hedge*, o qual é um algoritmo para predição de informações muito utilizado em jogos.

Em [Agrawal e Wimmers 2000], é apresentado um *framework* para expressar e combinar preferências. Uma das principais características desse *framework* está em introduzir um atributo no Banco de Dados chamado *score*, tal atributo armazenará uma nota para cada objeto. A nota para alguns objetos é fornecida pelo usuário, a partir das quais é obtida a função de preferência. O usuário tem como possibilidade de voto: os valores numéricos entre 0 e 1, valor 1 indica que o objeto é o mais preferido e 0 é o menos preferido. O usuário ainda tem como opção vetar o objeto ($\perp$) ou declarar explicitamente ser indiferente àquele objeto ($\perp$). A partir dos *scores* dados pelo usuário são definidos para cada registro do banco de dados e para cada atributo deste registro, uma função de preferência. Por exemplo, considerando o esquema Filmes(CodFilme:Int, Autor:String, Genero:String, Duração:Real, Década:Int) têm-se as funções: Filmes.Autor, Filmes.Genero, Filmes.Duração e Filmes.Década, as quais retornam as preferências sobre o domínio daquele atributo.

As funções de preferências podem ser combinadas, de tal maneira que ao final se têm uma única função. É proposto também uma função de combinação, que recebe como parâmetro de entrada as funções de preferência. Tal combinação é feita de maneira a manter a autonomia de cada preferência utilizada.

O trabalho de [Freund et al. 2003] propõe apresentar um algoritmo de aprendizagem para obter precisamente o *ranking* para um conjunto de objetos pela combinação de uma dada coleção de *rankings* ou funções de preferência. Este tipo de problema de combinação de preferências surge em muitas aplicações, como por exemplo combinação de resultados de diferentes sistemas de busca, ou o problema de “filtros colaborativos”, muito utilizado em Sistemas de Busca de filmes, onde é possível prever os *rankings* de filmes para um usuário baseado nos *rankings* de filmes providos por outros usuários.

Dado X um banco de dados, cuja as instâncias são objetos a serem *rankeados*, cada *ranking* é uma função de preferência da forma $f_i : X \rightarrow R$, onde R constitui números reais, e é o *ranking* obtido.

O algoritmo proposto utiliza de técnicas semelhantes aos problemas de classificação, sendo que instâncias já *rankeadas* fornecidas pelo usuário são usadas como dados de treinamento. A partir desses dados são obtidas funções de preferências. O diferencial proposto neste trabalho são as funções de *feedback* que além dos *rankings* de outros usuários dá ao usuário corrente a opção de também opinar sobre o processo de ordenação.

Dentre os trabalhos que compõe a família das abordagens de mineração de preferências com o uso de funções está o trabalho de [Lin et al. 2007], onde a proposta é utilizar *preference mining* para obter os melhores objetos a partir de “cliques” do usuário sobre os resultados retornados de uma consulta e depois utilizar uma função de *ranking*, a qual é responsável por inferir uma ordem sobre os objetos do Banco de Dados.

Os “cliques” fornecidos como entrada, constituem os objetos preferidos do usuário e são formalmente representados como uma tripla (q, r, c) , onde q é a consulta feita pelo usuário, r é o resultado retornado e c são os objetos “clicados” pelo usuário. O conjunto c de objetos escolhidos forma o conjunto de amostras positivas. As amostras negativas são inferidas a partir dos objetos não escolhidos fazendo $(r - c)$.

Para a obtenção da *função de ranking* são utilizadas algumas amostras positivas (no trabalho são chamados de espiões ou *spies*) e todos as amostras negativas como dados de treinamento em um classificador bayesiano. A partir do classificador bayesiano é obtida uma função de *ranking* mais geral. E funcionando como um otimizador da função de *ranking* obtida é utilizado também um outro classificador, o SVM (Máquina de Suporte Vetorial), o qual produz uma função de *ranking* mais específica e compacta.

3.3 Considerações finais

Neste capítulo nos atemos a abordar os principais trabalhos envolvendo *mineração de preferências*. Inferir preferências a partir de um conjunto de informações repassadas pelo usuário é ainda um campo novo, o que justifica termos tão poucos trabalhos relacionados. No campo da mineração de preferências condicionais, deve-se ainda realçar que não encontramos trabalhos envolvendo tais formalismos, nos dando o mérito de estarmos propondo neste trabalho uma primeira abordagem.

Capítulo 4

Formalização do Problema

Neste capítulo vamos introduzir formalmente o problema que pretendemos resolver como tema desta dissertação. Para um melhor entendimento daremos inicialmente algumas definições importantes para a descrição da proposta deste trabalho.

4.1 Definições preliminares

Definição 4.1. (Amostra de Preferência) Seja um esquema relacional $R(A_1, A_2, \dots, A_n)$ e X um conjunto de tuplas sobre R . Sabendo que X_i e X_j são tuplas no conjunto X , um elemento de $X_i \times X_j \times \{1, 0\}$, com $X_i \neq X_j$, é dito uma *amostra de preferência*. Intuitivamente, o par $[(u, v), 1]$ com $u, v \in X$, significa que a tupla u é preferida à tupla v , ou seja, $v < u$. O par $[(u, v), 0]$ significa a tupla v é preferida à tupla u , isto é, $u < v$.

Exemplo 4.1. Seja o esquema relacional $R(A, B, C, D)$ e o domínio dos atributos $dom(A) = \{a_1, a_2, a_3\}$, $dom(B) = \{b_1, b_2\}$, $dom(C) = \{c_1, c_2\}$, $dom(D) = \{d_1, d_2\}$. Na Tabela 4.2 temos um conjunto de amostras construído a partir das tuplas do banco de dados R da relação da Tabela 4.1.

Id	A	B	C	D
1	a_1	b_1	c_1	d_1
2	a_1	b_1	c_1	d_2
3	a_2	b_1	c_1	d_2
4	a_1	b_2	c_1	d_2
5	a_2	b_1	c_2	d_1
6	a_3	b_1	c_1	d_1

Tabela 4.1: BD R

Id	A	B	C	D	A	B	C	D	Pref
1	a_1	b_1	c_1	d_1	a_1	b_1	c_1	d_2	1
2	a_1	b_1	c_1	d_1	a_2	b_1	c_1	d_2	1
3	a_1	b_2	c_1	d_2	a_2	b_1	c_2	d_1	1
4	a_2	b_1	c_2	d_1	a_1	b_1	c_1	d_1	1
5	a_3	b_1	c_1	d_1	a_2	b_1	c_2	d_1	0
6	a_2	b_1	c_2	d_1	a_2	b_1	c_1	d_2	0
7	a_1	b_2	c_1	d_2	a_1	b_1	c_1	d_1	1

Tabela 4.2: Amostras de preferências

As amostras apresentadas na Tabela 4.2 não representam todas as opções possíveis de pares de tuplas mediante o banco de dados R da Tabela 4.1. Tal fato é claramente

justificável, uma vez que em aplicações reais a necessidade do usuário avaliar todas as opções de pares de tuplas tornaria o nosso método inviável.

Para minimizar a complexidade das amostras de preferências realizamos um pré-processamento o qual chamamos de *espelhamento das amostras de preferências*. O espelhamento constitui-se em transformar as amostras do tipo $[(u, v), 0]$ em amostras do tipo $[(v, u), 1]$. Após a realização de tal tarefa o uso do atributo de preferência é dispensado, uma vez que em todas as amostras a primeira tupla será a preferida pelo usuário. Algumas vezes neste trabalho chamamos uma amostra de preferência simplesmente de *bitupla*, uma vez que uma *bitupla*, no contexto deste trabalho, é uma amostra de preferência sem o atributo de preferência, isto é uma amostra de preferência previamente espelhada, um par de tuplas.

Na Tabela 4.3 é possível visualizar as amostras da Tabela 4.2 após a realização do espelhamento. Observe que a inversão das tuplas somente foi necessária nas *bituplas* da Tabela 4.2 cujo $Id = 5$ e $Id = 6$.

Id	A	B	C	D	A	B	C	D
1	a_1	b_1	c_1	d_1	a_1	b_1	c_1	d_2
2	a_1	b_1	c_1	d_1	a_2	b_1	c_1	d_2
3	a_1	b_2	c_1	d_2	a_2	b_1	c_2	d_1
4	a_2	b_1	c_2	d_1	a_1	b_1	c_1	d_1
5	a_2	b_1	c_2	d_1	a_3	b_1	c_1	d_1
6	a_2	b_1	c_1	d_2	a_2	b_1	c_2	d_1
7	a_1	b_2	c_1	d_2	a_1	b_1	c_1	d_1

Tabela 4.3: Espelhamento realizado na Tabela 4.2

Definição 4.2. (Consistência de um conjunto de exemplos de preferências) Um conjunto Υ de *amostras de preferência* é dito *consistente* se para $\forall$ amostra de preferência $[(u, v), n] \in \Upsilon$, se $\exists [(v, u), m]$, então as seguintes condições para m e n são satisfeitas:

- Se $n = 1$, então $m = 0$, isto é, o par $[(u, v), 1]$ nos diz que $v < u$ (u é preferido a v) e o par $[(v, u), 0]$ também nos diz que $v < u$ (u é preferido a v), logo a consistência se mantém;
- Se $n = 0$ então $m = 1$, isto é, o par $[(u, v), 0]$ nos diz que $v < u$ (u é preferido a v) e o par $[(u, v), 1]$ também nos diz que $v < u$ (u é preferido a v), logo a consistência se mantém;

Definição 4.3. (Ordenador) Seja um esquema relacional $R(A_1, A_2, \dots, A_n)$ e X um conjunto de tuplas sobre R . Uma função $\rho : X \times X \rightarrow \{0, 1\}$ é dito um *ordenador ou modelo de ordenação* de X . Um *ordenador* ρ induz uma ordem parcial sobre X , ou seja, dados duas tuplas $u, v \in X$, dizemos que $v < u$ se $\rho(u, v) = 1$, ou que $u < v$ se $\rho(u, v) = 0$. Estamos interessados em ordenadores que induzem ordens de preferências condicionais. Chamamos tais ordenadores de *ordenadores condicionais*.

Definição 4.4. (Rede de preferência) Uma *rede de preferências* é um *ordenador* baseado nos classificadores bayesianos. Os classificadores bayesianos são classificadores estatísticos que fornecem regras de classificação, sendo que tais regras nos permitem classificar um objeto numa determinada classe baseando-se na probabilidade condicional deste objeto pertencer a esta classe. Analogamente, uma *rede de preferências* fornece regras de preferência, segundo as quais é possível inferir uma *ordem* sobre os objetos de uma base de dados.

Uma *rede de preferências* sobre um esquema relacional $R(A_1, \dots, A_n)$ é uma estrutura $(B_S, \mathfrak{S})$ onde:

1. B_S é um *Grafo Acíclico Direcionado*, cujos vértices são *atributos* $A_i \in \{A_1, \dots, A_n\}$ e as arestas representam *dependências* entre os atributos.
2. $\mathfrak{S}$ é função que associa a cada vértice A_i do grafo B_S uma *tabela de probabilidades condicionais de preferências* do tipo $P[E_2|E_1]$, onde E_1 corresponde a um evento do tipo $(A_1 = a_1) \wedge \dots \wedge (A_i = a_i)$, tal que $\forall i \in \{1, \dots, n\} a_i \in \text{dom}(A_i)$.

Cada Tabela de Probabilidade de Preferências $P[E_2|E_1]$ associa uma *ordem parcial* “<” sobre os valores de $\text{Dom}(X_i)$ dado seus *pais*(X_i) por meio de um valor entre 0 e 1. Tal valor representa a probabilidade de uma preferência $x_{i1} < x_{i2}$ ser verdadeira, onde x_{i1} e $x_{i2} \in \text{dom}(X_i)$.

Exemplo 4.2. Seja o esquema relacional dado no Exemplo 4.1. Na Figura 4.1 é mostrada uma possível rede de preferências (PrefNet_1) sobre este esquema.

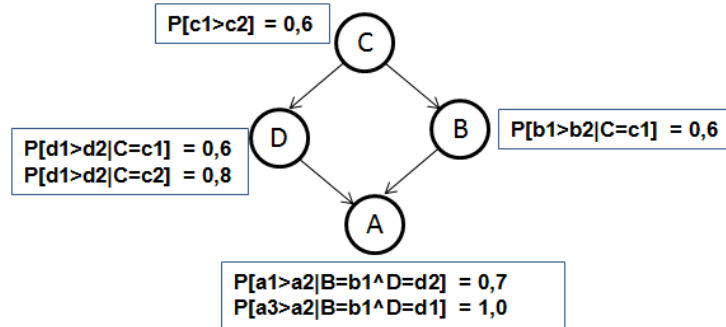


Figura 4.1: Rede de preferências PrefNet_1

Como a partir de uma rede de preferências obter uma ordem?

Assim como os classificadores bayesianos possibilitam classificar uma tupla em uma dada classe, as redes de preferências são ordenadores que permitem ordenar um par de tuplas α, β segundo as preferências fornecidas pelo usuário. Desta forma, nossa tarefa é comparar duas tuplas α, β construídas sobre um esquema relacional R , sendo R constituído de um conjunto de atributos, $\text{Atr}(R)$. Tal tarefa será realizada utilizando as redes de preferência como veículo ordenador.

Para cada teoria de preferências condicionais com grafo de dependências acíclico e localmente consistente é possível associar uma ordem lexicográfica $>_l$ *ex* (tal proposição é provada em [Wilson 2004]). Sendo assim, é possível a partir de uma rede de preferências obter uma ordem lexicográfica. O trabalho de [Wilson 2004] foi abordado de maneira geral no Capítulo 2. Para um maior entendimento veja a Figura 4.2, que trás a comparação entre a ordem lexicográfica proposta por [Wilson 2004] e a nossa abordagem.

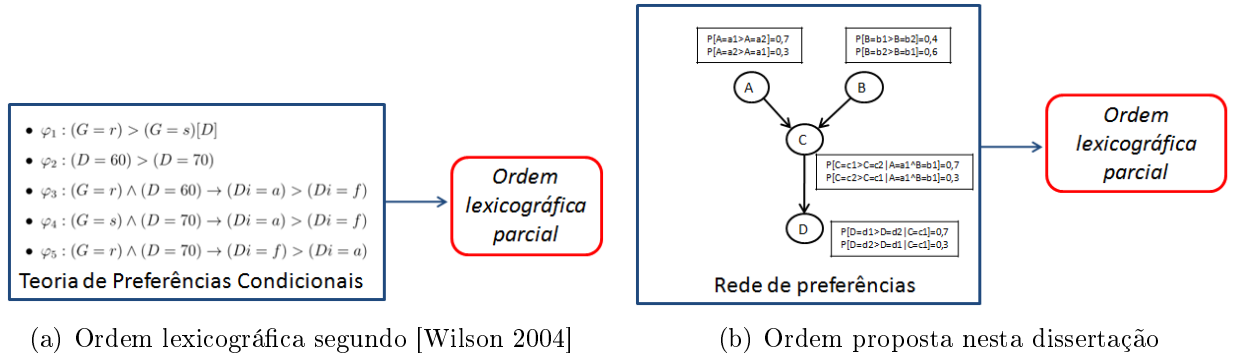


Figura 4.2: Comparação entre a ordem de [Wilson 2004] e a ordem proposta em nossa abordagem

A fim de obter uma ordem entre as tuplas $\alpha, \beta \in R$ via rede de preferências propomos o processo descrito a seguir baseado no trabalho de [Wilson 2004].

Processo para ordenação de um par de tuplas segundo uma rede de preferências

Seja $M(G, \theta)$ uma rede de preferências composta por uma estrutura S e tabelas de probabilidades condicionais θ . Seja ainda R um esquema relacional e $Atr(R) = \{X_1, \dots, X_n\}$ os atributos de R , o processo para a ordenação do par de tuplas α, β é:

1. Determinar o conjunto dos atributos onde α e β diferem, o qual denominamos $\Delta(\alpha, \beta)$. Sendo assim, $\Delta(\alpha, \beta) = \{X \in Atr(R) \text{ tal que } \alpha(X) \neq \beta(X)\}$.
2. Determinar o conjunto de atributos de $\Delta(\alpha, \beta)$ que não são dominados por outros atributos de $\Delta(\alpha, \beta)$ no grafo de dependências, chamamos $min(\Delta(\alpha, \beta)) = \{X \in Atr(R) | X \text{ não é dominado por atributo de } \Delta(\alpha, \beta)\}$. Sendo assim, a partir de $min(\Delta(\alpha, \beta))$ é possível dizer que X não possui ancestral $Y \in \Delta(\alpha, \beta)$ na estrutura de rede de preferências G .

Exemplo 4.3. Seja a rede de preferências da Figura 4.3. Uma rede de preferências como proposto neste trabalho é constituída de um grafo direcionado acíclico ou seja um grafo de dependências G e um conjunto de tabelas de probabilidades condicionais associadas a cada vértice desse grafo. Seja também duas tuplas $\alpha = (a, 4, 2)$ e $\beta = (a', 4, 3)$ construídas sobre o esquema relacional $R(A, B, C)$, onde os domínios

dos respectivos atributos são: $dom(A) = \{a, a'\}$, $dom(B) = \{1, 2, 4\}$ e $dom(C) = \{1, 2, 3\}$. O conjunto dos atributos que diferem em seus respectivos valores em α e β é $\Delta(\alpha, \beta) = \{A, C\}$. E o conjunto dos atributos de $\Delta(\alpha, \beta)$ que não possuem atributos ancestrais no grafo de dependência existentes em $\Delta(\alpha, \beta)$ é $min(\Delta(\alpha, \beta)) = \{A\}$.

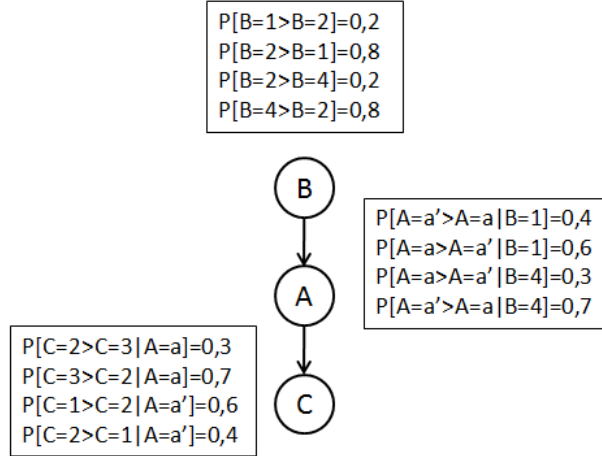


Figura 4.3: Rede de preferências

3. Para cada atributo $X \in min(\Delta(\alpha, \beta))$, inferir a probabilidade P_0^X e a probabilidade P_1^X (Complementar da probabilidade P_0^X) a partir da rede de preferências, onde $P_0^X = P[\alpha(X) >_u^X \beta(X)]$ e $P_1^X = P[\beta(X) >_u^X \alpha(X)]$ dado que u é a condição de que os atributos pais de X em α sejam iguais aos atributos pais de X em β , ou seja, $u = \alpha|_{Pais(X)} = \beta|_{Pais(X)}$.

Exemplo 4.4. Considere a situação do Exemplo 4.3. Tendo em vista que o $min(\Delta(\alpha, \beta)) = \{A\}$, então as probabilidades relevantes dizem respeito somente ao atributo A . Logo $P_0^A = P[\alpha(A) >_u^A \beta(A)] = 0,3$, e a probabilidade $P_1^A = P[\beta(A) >_u^A \alpha(A)] = 0,7$ sendo que $u = \alpha|_{Pais(A)} = \beta|_{Pais(A)} = \{B = 4\}$.

4. Se o conjunto $min(\Delta(\alpha, \beta))$ é maior que 1 faz-se então o produto das probabilidades $P_k^{X_i}$ para $k = 0$ e também para $k = 1$. Ao final teremos então uma probabilidade $\mathcal{P}_0$ resultado do multiplicatório de todas as probabilidades $P_0^{X_i}$ e uma probabilidade $\mathcal{P}_1$ fruto do multiplicatório de todas as probabilidades $P_1^{X_i}$.

$$\mathcal{P}_k = \prod_{i=0}^n P_k^{X_i}, k = \{0, 1\}$$

O máximo valor entre essas duas probabilidades $\mathcal{P}_0$ e $\mathcal{P}_1$ nos induz a uma ordem entre o par de tuplas α e β .

Exemplo 4.5. Considere a situação do Exemplo 4.3 e do Exemplo 4.4. Como $\min(\Delta(\alpha, \beta)) = \{A\}$, então a única probabilidade a ser levada em consideração é referente ao atributo A. Sendo assim, devemos considerar $\mathcal{P}_0 = P_0^A = 0,3$ e $\mathcal{P}_1 = P_1^A = 0,7$. A maior probabilidade nos dá a inferência da ordem sobre o par de tuplas, ou seja, é possível inferir que β é preferida a α , pois $\mathcal{P}_1 > \mathcal{P}_0$.

A seguir, numa tentativa de deixar mais clara a obtenção de uma ordem sobre um par de tuplas, apresentamos nos Exemplos 4.6 e 4.7 redes de preferências mais elaboradas e a respectiva ordem sobre alguns pares de tuplas do domínio.

Exemplo 4.6. Seja o esquema relacional $R(A, B, C, D, E)$ e a rede de preferências da Figura 4.4. O domínio dos atributos referente a cada atributo é: $\text{dom}(A) = \{a1, a2\}$, $\text{dom}(B) = \{b1, b2\}$, $\text{dom}(C) = \{c1, c2\}$, $\text{dom}(D) = \{d1, d2\}$ e $\text{dom}(E) = \{e1, e2\}$. Deseja-se inferir uma ordem sobre o par de tuplas $\alpha(a1, b1, c1, d1, e1)$ e $\beta(a2, b1, c2, d2, e2)$.

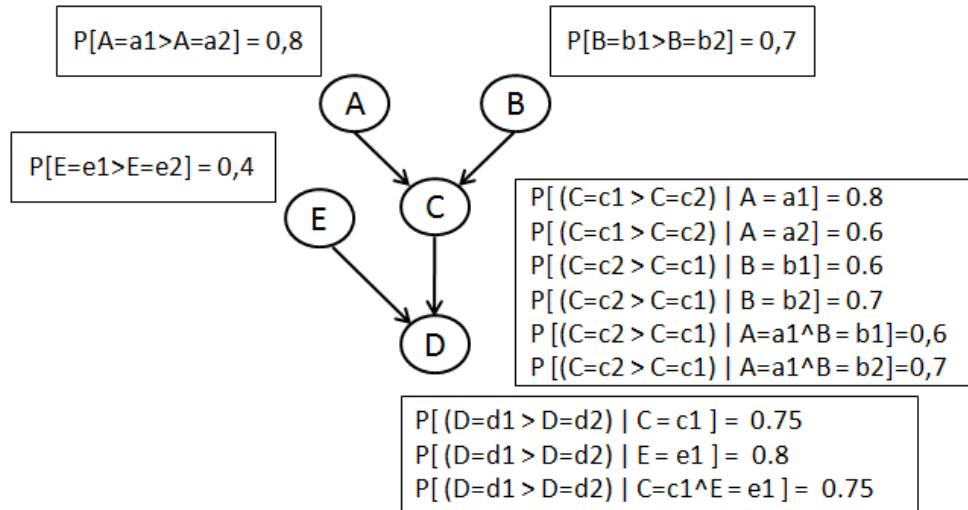


Figura 4.4: Rede de preferências exemplo

Para inferir uma ordem sobre o par de tuplas $\alpha(a1, b1, c1, d1, e1)$ e $\beta(a2, b1, c2, d2, e2)$ é realizado a sequência de passos a seguir:

1. Inicialmente é calculado o valor de $\Delta(\alpha, \beta)$, desta forma, $\Delta(\alpha, \beta) = \{A, C, D, E\}$;
2. Na sequência é calculado o valor de $\min(\Delta(\alpha, \beta))$, assim temos que $\min(\Delta(\alpha, \beta)) = \{A, E\}$;
3. O próximo passo é obter das tabelas de probabilidade o $\mathcal{P}_k$, isto é, $\mathcal{P}_0$ e o $\mathcal{P}_1$. Assim temos que:

$$(a) \mathcal{P}_0 = P_0^A * P_0^E \Rightarrow \mathcal{P}_0 = 0,8 * 0,4 = 0,32$$

$$(b) \mathcal{P}_1 = P_1^A * P_1^E \Rightarrow \mathcal{P}_1 = 0,2 * 0,6 = 0,12$$

4. Como $\mathcal{P}_0 > \mathcal{P}_1$ é possível inferir que a tupla α é preferida a tupla β .

Exemplo 4.7. Seja o esquema relacional $R(A, B, C, D, E, F, G)$ e a rede de preferências da Figura 4.5. O domínio dos atributos referente a cada atributo é: $dom(A) = \{a1, a2\}$, $dom(B) = \{b1, b2\}$, $dom(C) = \{c1, c2\}$, $dom(D) = \{d1, d2\}$, $dom(E) = \{e1, e2\}$, $dom(F) = \{f1, f2\}$ e $dom(G) = \{g1, g2\}$. Deseja-se inferir uma ordem sobre os pares de tuplas a seguir:

1. $\alpha = (a1, b1, c1, d1, e1, f1, g1)$ e $\beta = (a1, b1, c2, d2, e1, f1, g1)$;
2. $\mu = (a1, b1, c1, d1, e1, f1, g1)$ e $\kappa = (a2, b2, c2, d2, e2, f2, g2)$.

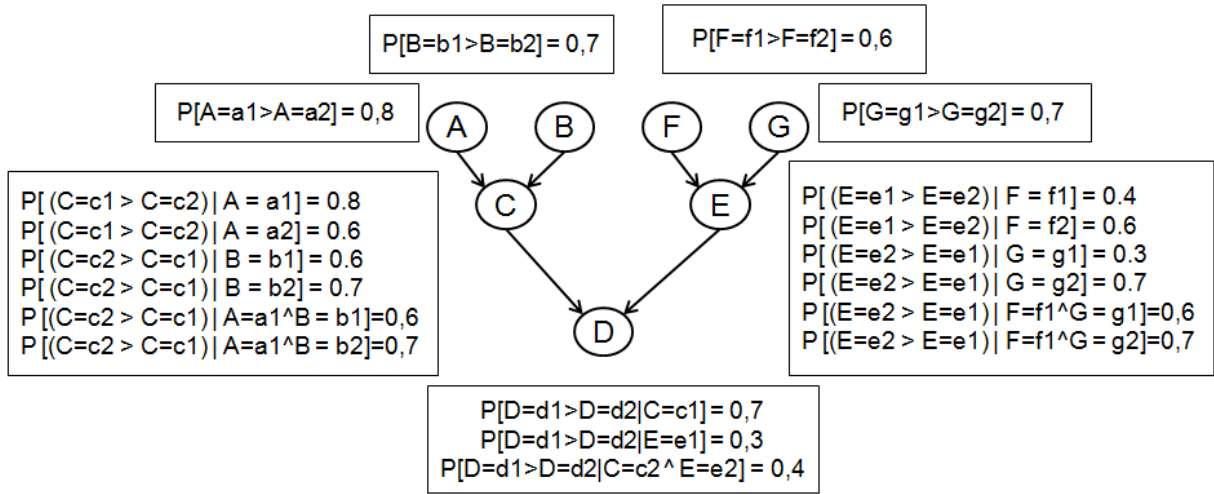


Figura 4.5: Rede de preferências para o exemplo

Para inferir uma ordem sobre o par de tuplas $\alpha = (a1, b1, c1, d1, e1, f1, g1)$ e $\beta = (a1, b1, c2, d2, e1, f1, g1)$ é realizado a sequência de passos a seguir:

1. Inicialmente é calculado o valor de $\Delta(\alpha, \beta)$, desta forma, $\Delta(\alpha, \beta) = \{C, D\}$;
2. Na sequência é calculado o valor de $\min(\Delta(\alpha, \beta))$, assim temos que $\min(\Delta(\alpha, \beta)) = \{D\}$;
3. O próximo passo é obter das tabelas de probabilidade o $\mathcal{P}_k$, isto é, $\mathcal{P}_0$ e o $\mathcal{P}_1$. Assim temos que:

(a) $\mathcal{P}_0 = P_0^D = 0,3$;

(b) $\mathcal{P}_1 = P_1^D = 0,7$;

4. Como $\mathcal{P}_1 > \mathcal{P}_0$ é possível inferir que a tupla β é preferida a tupla α .

Para inferir uma ordem sobre o par de tuplas $\mu = (a1, b1, c1, d1, e1, f1, g1)$ e $\kappa = (a2, b2, c2, d2, e2, f2, g2)$ é realizado a sequência de passos a seguir:

1. Inicialmente é calculado o valor de $\Delta(\mu, \kappa)$, desta forma, $\Delta(\mu, \kappa) = \{A, B, C, D, E, F, G\}$;
2. Na sequência é calculado o valor de $\min(\Delta(\mu, \kappa))$, assim temos que $\min(\Delta(\mu, \kappa)) = \{A, B, F, G\}$;
3. O próximo passo é obter das tabelas de probabilidade o $\mathcal{P}_k$, isto é, $\mathcal{P}_0$ e o $\mathcal{P}_1$. Assim temos que:

$$(a) \mathcal{P}_0 = P_0^A * P_0^B * P_0^F * P_0^G = 0,8 * 0,7 * 0,6 * 0,7 = 0,2352$$

$$(b) \mathcal{P}_1 = P_1^A * P_1^B * P_1^F * P_1^G = 0,2 * 0,3 * 0,4 * 0,3 = 0,0072$$

4. Como $\mathcal{P}_0 > \mathcal{P}_1$ é possível inferir que a tupla μ é preferida a tupla κ .

Será usado duas métricas para avaliação da corretude dos métodos de ordenação propostos nesta dissertação: a *acurácia* de uma rede de preferências e a *precisão* de uma rede de preferências. A seguir é dado uma definição sucinta de tais metodologias de avaliação.

Definição 4.5. (Acurácia de uma rede de preferências) A *acurácia* de uma rede de preferência é medida de forma análoga à acurácia de uma rede bayesiana. Considera-se um banco de dados $\mathcal{H}$ de *amostras de preferências*, divide-se o mesmo em duas partes $\mathcal{H}_1$ e $\mathcal{H}_2$. $\mathcal{H}_1$ é o conjunto de *amostras de treinamento* utilizado para gerar o modelo de ordenação, isto é, a rede de preferências. $\mathcal{H}_2$ é o conjunto de *amostras de testes* e é utilizado para calcular a acurácia da rede de preferências gerada. Para cada par de tuplas (t_1, t_2) pertencente a $\mathcal{H}_2$, a rede de preferências infere uma ordem de preferência entre t_1 e t_2 . A acurácia da rede de preferências é a porcentagem de pares de tuplas de $\mathcal{H}_2$ cuja ordem inferida pela rede de preferências coincide com a ordem original dada em $\mathcal{H}_2$. Na Equação 4.1 sintetizamos o cálculo da acurácia da rede de preferências.

$$\text{Acurácia de uma rede de preferências} = \frac{\text{nº de objetos de } \mathcal{H}_2 \text{ bem ordenados}}{\text{nº total de objetos em } \mathcal{H}_2} \quad (4.1)$$

Definição 4.6. (Precisão de uma rede de preferências) A *precisão* de uma rede de preferência é medida de forma análoga à precisão de uma rede bayesiana. Considera-se um banco de dados $\mathcal{H}$ de *amostras de preferências*, divide-se o mesmo em duas partes $\mathcal{H}_1$ e $\mathcal{H}_2$. Assim como no cálculo da acurácia, $\mathcal{H}_1$ é o conjunto de *amostras de treinamento* utilizado para gerar o modelo de ordenação, isto é, a rede de preferências. $\mathcal{H}_2$ é o conjunto de *amostras de testes* e é utilizado para calcular a precisão da rede de preferências gerada. A precisão diz respeito ao número de *bituplas* ordenadas corretamente em $\mathcal{H}_2$ dividido pelo número total de *bituplas* que o ordenador conseguiu ordenar em $\mathcal{H}_2$, considerando que ele pode não conseguir ordenar algumas tuplas.

$$\text{Precisão de uma rede de preferências} = \frac{\text{nº de bituplas ordenadas corretamente}}{\text{nº de bituplas que o ordenador conseguiu ordenar}} \quad (4.2)$$

Exemplo 4.8. Seja a rede de preferências da Figura 4.6 a seguir, e a amostra de preferência da Figura 4.6(a):

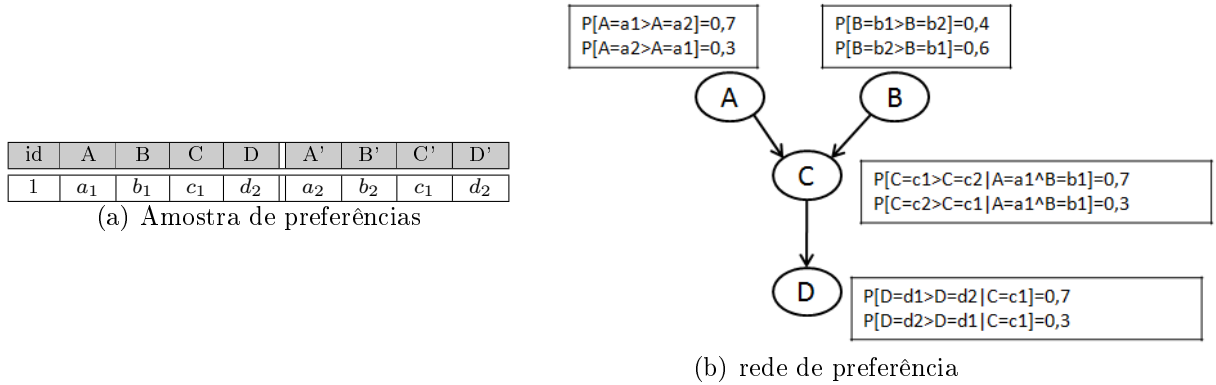


Figura 4.6: Exemplo

Ordenar a *bitupla* da Figura 5.10(a), considerando que somente as preferências sobre os atributos A e B são relevantes consiste em calcular as seguintes probabilidades:

1. $P[A = a_1 > A = a_2]$ e $P[B = b_1 > B = b_2] = 0,7 * 0,4 = 0,28$
2. $P[A = a_2 > A = a_1]$ e $P[B = b_1 > B = b_2] = 0,3 * 0,4 = 0,12$
3. $P[A = a_1 > A = a_2]$ e $P[B = b_2 > B = b_1] = 0,7 * 0,6 = 0,42$
4. $P[A = a_2 > A = a_1]$ e $P[B = b_2 > B = b_1] = 0,3 * 0,6 = 0,18$

A probabilidade *tupla 1* ser preferida a *tupla 2* é calculada no item 1. Esta probabilidade é 0,28.

A probabilidade da *tupla 2* ser preferida a *tupla 1* é calculada no item 4. Esta probabilidade é 0,18.

A probabilidade das tuplas serem incomparáveis é a soma das probabilidades dos itens 2 e 3 $= 0,12 + 0,42 = 0,54$.

Como a maior probabilidade corresponde ao fato das tuplas serem incomparáveis, então a rede permite concluir que *as tuplas são incomparáveis*.

A rede de preferências generaliza a noção de CP-net acrescentando o conhecimento probabilístico. Cada probabilidade associada ao nó significa uma preferência com um certo grau de crença, por exemplo, $P[D = d_1 < D = d_2 | C = c_2] = 0,6$ significa que “Se $C = c_2$, a probabilidade de $D = d_2$ ser preferido a $D = d_1$ é de 60% e a probabilidade de $D = d_1$ ser preferido a $D = d_2$ é de 40%, isto é, o seu complementar”.

Definição 4.7. ($\text{Comp}(A_i)$) Seja A_i um atributo do esquema relacional $R(A_1, \dots, A_n)$ e $\mathcal{H}$ um conjunto de amostras de preferências sobre R . Denotamos por $\text{Comp}(A)$ o conjunto dos pares p do tipo (a_1, a_2) tais que $\exists \text{ bituplas } (t_1, t_2) \in \mathcal{H} \text{ e } t_1[A] = a_1, t_2[A] = a_2 \text{ e } a_1 \neq a_2$.

Exemplo 4.9. Seja o esquema descrito no Exemplo 4.1 e as amostras de preferências da Tabela 4.4, então temos que $\text{comp}(A) = \{(a_1, a_2), (a_3, a_2)\}$.

Id	A	B	C	D	A	B	C	D	Pref
1	a_1	b_1	c_1	d_1	a_1	b_1	c_1	d_2	1
2	a_1	b_1	c_1	d_1	a_2	b_1	c_1	d_2	1
3	a_1	b_2	c_1	d_2	a_2	b_1	c_2	d_1	1
4	a_2	b_1	c_2	d_1	a_1	b_1	c_1	d_1	1
5	a_3	b_1	c_1	d_1	a_2	b_1	c_2	d_1	0
6	a_2	b_1	c_2	d_1	a_2	b_1	c_1	d_2	0
7	a_1	b_2	c_1	d_2	a_1	b_1	c_1	d_1	1

Tabela 4.4: Amostras de preferências

Na Figura 4.7 demonstramos como é feito o particionamento do banco de amostras da Tabela 4.4 segundo $\text{Comp}(A)$, $\text{Comp}(B)$, $\text{Comp}(C)$ e $\text{Comp}(D)$ respectivamente. Observe que a primeira partição (amostras 2,3 e 4) da Figura 4.7(a) abriga amostras de preferências que possuem pares tanto do tipo (a_1, a_2) quanto do tipo (a_2, a_1) .

Id	A	B	C	D	A	B	C	D
2	a_1	b_1	c_1	d_1	a_2	b_1	c_1	d_2
3	a_1	b_2	c_1	d_2	a_2	b_1	c_2	d_1
4	a_2	b_1	c_2	d_1	a_1	b_1	c_1	d_1

Id	A	B	C	D	A	B	C	D
5	a_2	b_1	c_2	d_1	a_3	b_1	c_1	d_1

(a) Particionamento segundo $\text{Comp}(A)$

Id	A	B	C	D	A	B	C	D
3	a_1	b_2	c_1	d_2	a_2	b_1	c_2	d_1
7	a_1	b_2	c_1	d_2	a_1	b_1	c_1	d_1

(b) Particionamento segundo $\text{Comp}(B)$

Id	A	B	C	D	A	B	C	D
3	a_1	b_2	c_1	d_2	a_2	b_1	c_2	d_1
4	a_2	b_1	c_2	d_1	a_1	b_1	c_1	d_1
5	a_2	b_1	c_2	d_1	a_3	b_1	c_1	d_1
6	a_2	b_1	c_1	d_2	a_2	b_1	c_2	d_1

(c) Particionamento segundo $\text{Comp}(C)$

Id	A	B	C	D	A	B	C	D
1	a_1	b_1	c_1	d_1	a_1	b_1	c_1	d_2
2	a_1	b_1	c_1	d_1	a_2	b_1	c_1	d_2
3	a_1	b_2	c_1	d_2	a_2	b_1	c_2	d_1
6	a_2	b_1	c_1	d_2	a_2	b_1	c_2	d_1
7	a_1	b_2	c_1	d_2	a_1	b_1	c_1	d_1

(d) Particionamento segundo $\text{Comp}(D)$

Figura 4.7: Particionamentos realizados nas amostras da Tabela A.2

4.2 Formalização do problema

Nosso problema é formalizado da seguinte maneira:

Entrada: Um conjunto $A \subseteq X \times X \times \{0,1\}$ de *amostras de preferências consistentes* sobre um conjunto X de tuplas;

Saída: Um *ordenador* $\rho : X \times X \rightarrow \{0,1\}$ que seja compatível com as amostras de preferências A e que tenha uma *boa* acurácia.

O ordenador construído será uma rede de preferências (Ver Definição 4.4), a partir do qual serão obtidas regras de preferências.

As regras de preferências obtidas seguem o formalismo das Cp-nets [Boutilier et al. 2004a], tal formalismo foi descrito neste trabalho na Seção 2.1.2. De posse das regras de preferências é possível inferir uma ordem sobre um conjunto de tuplas que pertence ao domínio do problema.

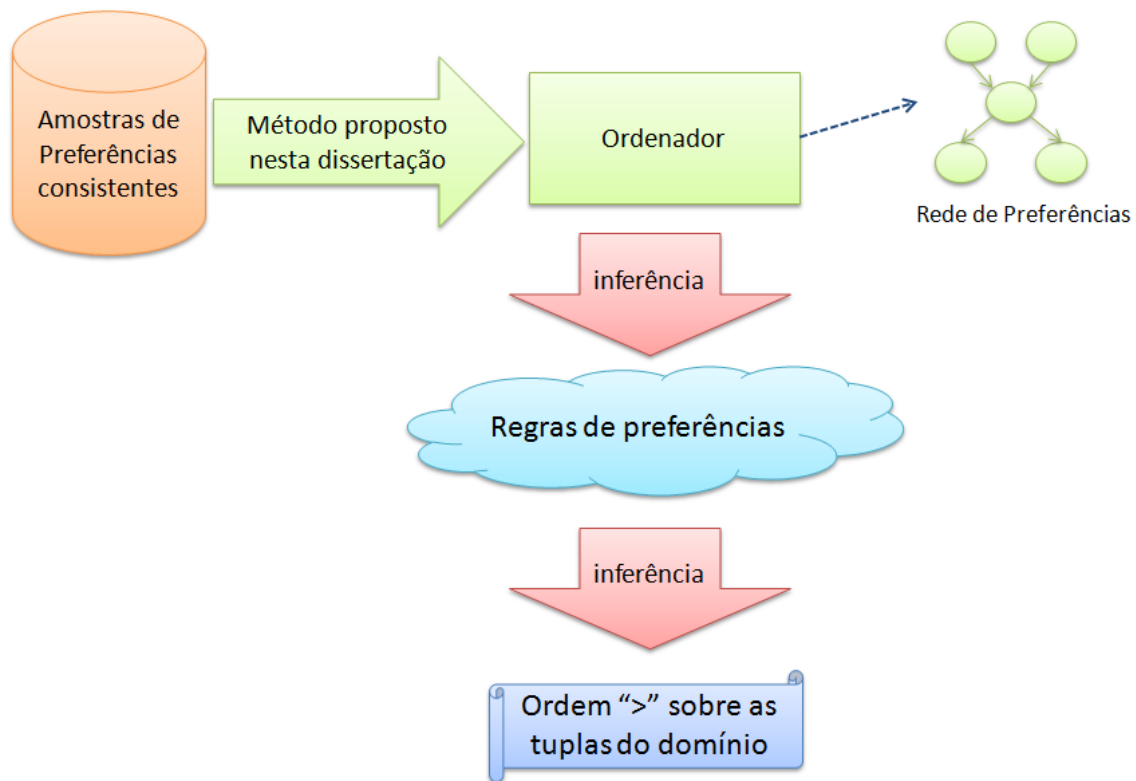


Figura 4.8: Contextualização do problema de construção de uma rede de preferências

A ideia é dividir o problema em dois subproblemas:

Para alcançar os objetivos propostos neste trabalho, seguimos a técnica de dividir para conquistar, particionando o problema de construir um *ordenador condicional* em dois subproblemas:

1. Identificar as relações de interdependência entre as variáveis de um domínio de dados, ou seja, o *processo de aprendizado da estrutura ou topologia da rede de preferências*;

2. A partir da topologia da rede de preferências e de um conjunto de amostras de preferências obter as tabelas de probabilidades condicionais, isto é, *treinar a rede de preferências*.

Os métodos propostos para a realização das tarefas anteriormente apresentadas serão descritos no Capítulo 5 .

4.3 Considerações finais

Neste capítulo introduzimos formalmente o problema que propomos resolver ao longo deste trabalho: *mineração de preferências condicionais*. Foram definidos inicialmente os conceitos preliminares para entendimento da dissertação como um todo e principalmente do modelo de ordenação *rede de preferências* proposto, mostrando como obter uma ordem sobre um par de tuplas a partir de uma *rede de preferências*. Ainda neste capítulo foi feita uma formalização e contextualização do problema de construção de uma *rede de preferências*.

Capítulo 5

Métodos Propostos

Construir uma rede de preferências assim como construir uma rede bayesiana consiste em um processo de aprendizado, o qual é dividido em duas partes: *aprendizado da estrutura ou topologia da rede de preferências* e *aprendizado de probabilidades simples e condicionais*, ou seja, o *treinamento da rede de preferências*.

A tarefa de aprendizado da topologia em redes de preferências assim como a tarefa de aprendizado da estrutura das redes bayesianas é uma tarefa não trivial que apresenta três grandes problemas. O primeiro problema é o espaço de busca de estruturas de redes bayesianas extremamente grande: em [Robinson 1977] é apresentada uma função para o cálculo do número de estruturas diferentes (grafos direcionados acíclicos - DAG), a função $f(n)$ da Equação 5.1. O número de grafos direcionados acíclicos cresce mais que exponencialmente no número de nós n (alguns exemplos de cálculos podem ser vistos na Tabela 5.1).

$$f(n) = \sum_{i=1}^n (-1)^{i+1} \binom{n}{i} 2^{i(n-i)} f(n-i). \quad (5.1)$$

O segundo problema é que em um espaço de busca de estruturas bayesianas clássicas é possível encontrar várias estruturas igualmente boas. Por exemplo, uma rede bayesiana ao longo de um grafo completo pode representar qualquer distribuição sobre o seu universo, entretanto, uma rede bayesiana ao longo de um grafo completo dificilmente será a resposta correta.

E por fim o problema de *overfitting*: ao buscar pela estrutura bayesiana clássica que melhor reflita a distribuição probabilística dos dados é possível encontrar estruturas que *falsamente* sejam determinadas como uma boa candidata. Por exemplo, um grafo completo é capaz de representar qualquer distribuição probabilística, mas o banco de dados pode melhor ser representado por um grafo esparso.

Número de Nós	Número de DAGs	Número de Nós	Número de DAGs
1	1	13	$1.9 \cdot 10^{31}$
2	3	14	$1.4 \cdot 10^{36}$
3	25	15	$2.4 \cdot 10^{41}$
4	543	16	$8.4 \cdot 10^{46}$
5	29281	17	$6.3 \cdot 10^{52}$
6	$3.8 \cdot 10^6$	18	$9.9 \cdot 10^{58}$
7	$1.1 \cdot 10^9$	19	$3.3 \cdot 10^{65}$
8	$7.8 \cdot 10^{11}$	20	$2.35 \cdot 10^{72}$
9	$1.2 \cdot 10^{15}$	21	$3.5 \cdot 10^{79}$
10	$4.2 \cdot 10^{18}$	22	$1.1 \cdot 10^{87}$
11	$3.2 \cdot 10^{22}$	23	$7.0 \cdot 10^{94}$
12	$5.2 \cdot 10^{26}$	24	$9.4 \cdot 10^{102}$

Tabela 5.1: A tabela mostra o número de diferentes grafos direcionados acíclicos (DAGs) que podem ser gerados para um dado número de nós (atributos). Por exemplo, existem $4.2 \cdot 10^{18}$ diferentes grafos direcionados acíclicos com 10 nós ([Robinson 1977]).

5.1 Aprendizagem da estrutura de rede de preferências

Vimos no capítulo 2 que os algoritmos para aprendizado de topologias de redes bayesianas podem ser classificados segundo dois enfoques: baseado em *restrições* ou baseado em *scores*. O presente trabalho, como dito em capítulos anteriores, é uma primeira abordagem no que diz respeito a mineração de preferências condicionais. E nesta primeira abordagem nos atemos a algoritmos de aprendizagem de topologias de redes de preferências segundo o *enfoque baseado em scores*.

Assim como nas redes bayesianas, os algoritmos para aprendizado de estruturas em redes de preferências, cujo enfoque é baseado em *score*, tem duas componentes: 1) uma boa função *score* e 2) um bom método de busca pelo máximo da função *score*.

Nas seções 5.1.1 e 5.1.2 abordaremos as funções *scores* e o método de busca considerado para aprendizagem da rede de preferências.

5.1.1 Funções *scores* consideradas

Para medir a qualidade das topologias de rede é utilizado uma função de avaliação que pode avaliar tanto a compatibilidade da topologia com os dados (quanto maior melhor) quanto a sua complexidade (quanto menor melhor). Neste trabalho foram consideradas duas *funções scores*:

1. $Score(B_S, \mathcal{H})$ (Definição 5.1): Seja B_S a topologia de uma rede de preferências e $\mathcal{H}$ um conjunto de amostras de preferências, a função $Score(B_S, \mathcal{H})$ é dada como uma função que avalia o quanto uma estrutura B_S reflete estatisticamente um conjunto de dados $\mathcal{H}$. $Score(B_S, \mathcal{H})$ foi adaptada de [Cooper e Dietterich 1992] para o nosso

contexto¹. A função $Score(B_S, \mathcal{H})$ avalia somente a compatibilidade da topologia com os dados não levando em consideração a complexidade da rede.

2. $Score_{complex}(B_S, \mathcal{H})$: foi adaptada de [Jensen e Nielsen 2007] e além de avaliar a compatibilidade da topologia com os dados, avalia também a complexidade da rede. A função é melhor definida na seção 5.1.4.

Na seção 2.2.2 é possível entender melhor a diferença entre uma função *score* levando em consideração a complexidade de uma rede de preferências e não levando tal premissa em consideração.

5.1.2 Estratégia de busca

Dada uma função de avaliação, a tarefa é encontrar a estrutura de rede de preferências com o maior valor (*score*) de todas as estruturas possíveis. Tal tarefa de aprendizado estrutural é então reduzida a um problema de busca. A parte desafiante desse problema é que o tamanho do espaço de todas as estruturas é super exponencial em relação ao número de vértices (ver Tabela 5.1), fazendo com que uma enumeração exaustiva de todas as estruturas não seja possível.

5.1.3 Algoritmo *PrefK2*

Para o aprendizado de topologia de rede de preferências propomos o algoritmo *PrefK2*, baseado no algoritmo *K2* ([Cooper e Dietterich 1992]) para aprendizado de topologia em redes bayesianas clássicas. O algoritmo *K2* recebe como entrada uma base de dados R e uma ordem sobre os atributos pertencentes a este banco de dados e retorna uma topologia (grafo) que possui a maior probabilidade de refletir a distribuição dos dados em questão. O algoritmo *PrefK2* que propomos recebe como entrada um conjunto de amostras de preferências $\mathcal{H}$ e uma ordem sobre os atributos envolvidos no conjunto de amostras de preferências e retorna uma estrutura B_S que melhor reflita a distribuição das amostras $\mathcal{H}$. O esquema do algoritmo *PrefK2* é mostrado na Figura 5.1.

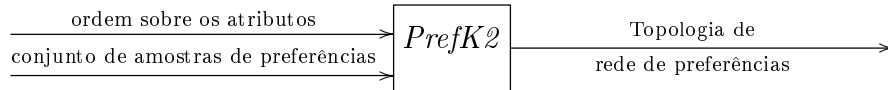


Figura 5.1: Esquema do algoritmo *PrefK2*

Definição 5.1. ($Score(B_S, \mathcal{H})$) Seja um esquema relacional $R = \{A_1, \dots, A_n\}$, um banco de dados $\mathcal{H}$ de k amostras de preferências construído sobre as tuplas de R . Seja uma estru-

¹Em [Cooper e Dietterich 1992] é apresentada a função $P(B_{S_i}|D)$ que corresponde à probabilidade da rede B_{S_i} refletir o banco de dados D de tuplas.

tura de rede de preferências denotada por B_S , onde cada atributo A_i em B_S tem um conjunto de pais, os quais são representados com uma lista de atributos em $\pi_i = \{A_1, \dots, A_n\}$ tal que $A_i \notin \pi_i$. Seja $Comp(A_i)$ igual a um conjunto de pares $\{p_1, \dots, p_m\}$ obtidos a partir de $\mathcal{H}$ para o atributo A_i e $l \in \{1, \dots, m\}$. A partir de cada $p_l \in Comp(A_i)$ podemos obter uma partição com no máximo dois possíveis pares para p_l : Supondo $dom(A_i) = \{u, v\}$ para a partição l , então podemos ter $p_l = (u, v)$ ou $p_l = (v, u)$. Denotamos por w_{ij}^l a j -ésima instanciiação de $\pi_i = \{A_1, \dots, A_n\}$ da partição l de $Comp(A_i)$, onde tal instanciiação é relativa a uma *bitupla* (t_1, t_2) e obedece à seguinte restrição: $t_1[A_1] = t_2[A_1] \wedge \dots \wedge t_1[A_n] = t_2[A_n]$, para todos atributos $\in \pi_i$. Supomos que há q_i^l instanciiações únicas de π_i segundo cada partição l do atributo A_i . Denotamos por N_{ij1} o número de amostras da partição l que possuem $Comp(X_i) = p_l$, $p_l = (u, v)$ e π_i instanciado como w_{ij}^l . E denotamos por N_{ij0} o número de amostras da partição l que possuem $p_l = (v, u)$ e também π_i instanciado como w_{ij}^l ;

A função $Score(B_S, \mathcal{H})$ é definida pela equação 5.2.

$$Score(B_S, \mathcal{H}) = \prod_{i=1}^n \sum_{l=1}^m \frac{1}{m} \prod_{j=1}^{q_i^l} \frac{N_{ij1}! N_{ij0}!}{(N_{ij1} + N_{ij0})!}. \quad (5.2)$$

Exemplo 5.1. Seja o esquema relacional dado no Exemplo 4.1. Na Figura 5.2 é mostrada uma possível rede de preferências ($PrefNet_1$) sobre este esquema.

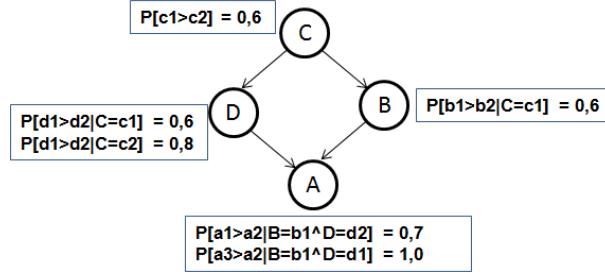


Figura 5.2: Rede de preferências $PrefNet_1$

Exemplo 5.2. Para exemplificar o cálculo do N_{ij0} e N_{ij1} referente ao atributo A da rede de preferências dada na Figura 5.2, considere os atributos pais de A , isto é, $\pi_i = \{B, D\}$ e o particionamento segundo $Comp(A)$ feito na Figura 4.7(a). Para a partição 1 da Figura 4.7(a) temos que $N_{ij0} = 1$, uma vez que a única instanciiação possível para seus pais é $B = b_1$ e $D = d_1$. A única amostra que possui a instanciiação $B = b_1$ e $D = d_1$ para os pais de A e possui o par (a_2, a_1) é a *bitupla* cujo $Id = 4$. Ainda na partição 1, obtemos o $N_{ij1} = 0$, uma vez que não existe no particionamento 1 o par (a_1, a_2) com a instanciiação $B = b_1$ e $D = d_1$ para seus pais. No caso da segunda partição dado que $\pi_i = \{B, D\}$, obtemos $N_{ij1} = 1$, uma vez possuímos uma tupla onde os atributos pais têm instanciiação $B = b_1$ e $D = d_2$. E por fim $N_{ij0} = 0$.

Aplicando a equação 5.2 para computar o $Score$ da rede de preferências dada na Figura 5.2 e o banco de amostras dado na Tabela 4.2, temos:

$$Score(PrefNet_1, \mathcal{H}) = \frac{\frac{2!2!}{(2+2)!}}{1} \frac{\frac{2!1!}{(2+1)!}}{1} \frac{\frac{1!0!}{(1+0)!}}{1} \frac{\frac{1!0!}{(1+0)!}}{2} \frac{\frac{1!0!}{(1+0)!}}{1} = Score(PrefNet_1, \mathcal{H}) = 0.0277 \quad (5.3)$$

No cálculo do $Score(PrefNet_1, \mathcal{H})$ obtivemos um valor 0.0277. Entretanto, outras redes poderiam melhor refletir as amostras descritas na Tabela 4.2, por exemplo, a rede da Figura 5.3 possui $Score(PrefNet_2, \mathcal{H}) = 0.0833$ e por ser um valor maior em relação ao $Score$ anterior, esta rede possui uma maior probabilidade de refletir tais dados.

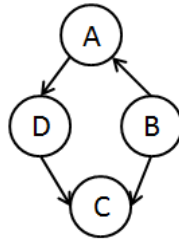


Figura 5.3: Rede de preferências $PrefNet_2$

O algoritmo *PrefK2* assume uma ordem entre os atributos e também assume que a priori todas as estruturas são igualmente prováveis. Ele busca para cada atributo A_i um conjunto de pais que maximize a função $g(i, \pi_i)$.

$$g(i, \pi_i) = \sum_{l=1}^m \frac{1}{m} \prod_{j=1}^{q_i^l} \frac{N_{ij1}! N_{ij0}!}{(N_{ij1} + N_{ij0})!}. \quad (5.4)$$

Considerando a complexidade no cálculo dos valores de N_{ij1} e N_{ij0} para um atributo A_i propomos neste trabalho utilizar uma árvore de índices T_i , a qual nós definimos a seguir:

Definição 5.2. (Árvore de índices T_i) Seja um esquema relacional $R = \{A_1, \dots, A_n\}$, um banco de dados $\mathcal{H}$ de z amostras de preferências construído sobre as tuplas de R . Seja $Comp(A_i)$, um conjunto de pares $P = \{p_1, \dots, p_m\}$ obtidos a partir de $\mathcal{H}$ para o atributo A_i , $l \in \{1, \dots, m\}$ e $p_i = (t_1, t_2) \in P$ tais que t_1 e t_2 são tuplas que obedecem a restrição: $t_1[A_1] = t_2[A_1] \wedge \dots \wedge t_1[A_n] = t_2[A_n]$. Seja π_i uma lista de atributos pais de A_i . Seja T_i uma *árvore de índices* definida para cálculo dos valores N_{ijk} , onde $k = 1$ ou $k = 0$, sendo que $k = 1$ refere-se a $t_1 > t_2$ e $k = 0$ refere-se a $t_2 > t_1$. Cada ramificação de um nó no nível d em T_i representa um valor de um $(d + 1)$ -ésimo pai de A_i . Um caminho a partir da raiz até as folhas corresponde a alguma instanciiação para os pais de A_i . Então, a profundidade da árvore é igual ao número de pais $|\pi_i|$ de A_i . Uma dada folha em T_i contém o total de valores de preferências para X_i que são condicionados à instanciiação de seus pais especificados da raiz até a folha de T_i . Este caminho corresponde a j -ésima

instanciação única de π_i (isto é, $\pi_i = w_{ij}^l$, onde l corresponde a partição sobre a qual a árvore está sendo construída). As folhas de T_i são denotas por f_j . Então, f_j em T_i corresponde a lista de valores N_{ijk} para $k = 0$ ou $k = 1$.

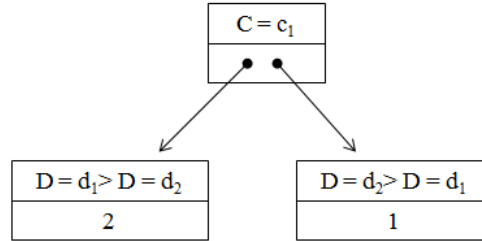


Figura 5.4: Árvore de índice para a partição segundo $\text{Comp}(D)$ da Figura 4.7(d), supondo que D depende do atributo C

A seguir passaremos à apresentação do algoritmo *PrefK2* (Figura 5.5) para construção da topologia de rede de preferências. Tal algoritmo é a composição de um método de busca guloso e da função de avaliação definida na Equação 5.2. Note que o algoritmo *PrefK2* apresentado não possui o parâmetro u (*upper bound*) utilizado no algoritmo *K2*. Não o utilizamos pois foi possível realizar os cálculos da sua função *score*, sem a necessidade de diminuir a complexidade dos mesmos estabelecendo um limite superior para o número de pais, seria o mesmo que considerar o parâmetro $u = n - 1$, onde n é o número de nós ².

Entrada: Um conjunto n de atributos discretos $A_i \in R(A_1, \dots, A_n)$, uma ordem sobre os atributos A_i, um banco de dados de k amostras de preferências em R. Saída: Um grafo B_S com os vértices em $A_1, \dots, A_n$.
1. Para i de 1 até n faça 2. Construa as partições P para cada $\text{Comp}(A_i)$ 3. $\Pi_i := \phi$; //acumula os pais das partições de A_i 4. Para cada partição $p_j \in P$ 5. $\pi_i := \phi$; //armazena os melhores pais da partição 6. $\text{Score}_{old} := g(i, \pi_i)$; 7. $\text{Proceder} := \text{TRUE}$; 8. Enquanto Proceder faça 9. Dado Z o conjunto dos nós em $\text{Predecessores}(A_i) - \pi_i$ que maximiza $g(i, \pi_i \cup \{Z\})$; 10. $\text{Score}_{new} := g(i, \pi_i \cup \{Z\})$; 11. Se $\text{Score}_{new} > \text{Score}_{old}$ então 12. $\text{Score}_{old} := \text{Score}_{new}$; 13. $\pi_i := \pi_i \cup \{Z\}$; 14. Senão $\text{Proceder} := \text{FALSE}$ 15. $\Pi_i := \Pi_i \cup \pi_i$ 16. Escreva ('No:', X_i, 'Pais deste nó', π_i)

Figura 5.5: Algoritmo *PrefK2*

²A mesma afirmativa vale para o algoritmo *PrefK2** descrito na seção 5.1.4

Ordem sobre os atributos

Um dos parâmetros de entrada do algoritmo $K2$ é a ordem sobre os atributos, que tem um impacto relevante na obtenção das estruturas de redes de preferências. Cobrir todas as ordens possíveis é uma tarefa inviável, tendo em vista que uma estrutura com n atributos possui $n!$ possíveis ordens. Por exemplo, supondo $n = 3$ atributos (A, B e C), temos as seguintes ordens:

- A,B,C
- A,C,B
- B,A,C
- B,C,A
- C,A,B
- C,B,A

A nossa proposta é utilizar o conceito de inversões em uma permutação em uma tentativa de cobrir o espaço de ordens possíveis escolhendo apenas as ordens que possuem maiores inversões.

Definição 5.3. (Inversões em uma Permutação) Seja $\vartheta \in S_n$ e sejam $i, j \in \{1, 2, \dots, n\}$ com $i < j$. O par i, j é chamado uma *inversão* em ϑ se $\vartheta(i) > \vartheta(j)$.

É mais fácil entendermos inversões quando a permutação é escrita como uma tabela $2 \times n$.

Seja:

$$\vartheta = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 2 & 1 & 5 & 3 \end{bmatrix}$$

Há $C_2^5 = 10$ maneiras de escolher um par de elementos $1 \leq i < j \leq 5$. Na tabela seguinte, listamos todos os pares i, j e verificamos se $\vartheta(i) > \vartheta(j)$.

i	j	$\vartheta(i)$	$\vartheta(j)$	Inversão
1	2	4	2	Sim
1	3	4	1	Sim
1	4	4	5	Não
1	5	4	3	Sim
2	3	2	1	Sim
2	4	2	5	Não
2	5	2	3	Não
3	4	1	5	Não
3	5	1	3	Não
4	5	5	3	Sim

Tabela 5.2: Inversões.

Assim, ϑ tem cinco inversões. O número máximo de inversões de qualquer ϑ é dado pela equação 5.5, onde n corresponde ao número de atributos.

$$\text{Número máximo de inversões} = \frac{n(n-1)}{2}, \quad (5.5)$$

Para maiores detalhes sobre inversões em uma permutação, ver [Scheinerman 2003] e [Rosen 1991].

5.1.4 Algoritmo *PrefK2**

O método tratado na seção 5.1.3 teve um comportamento não interessante para o nosso problema. Quando duas topologias com complexidades diferentes possuem a mesma avaliação, o algoritmo *PrefK2* escolhe uma das topologias sem estabelecer um critério para isto. Tal fato claramente não é o ideal. Uma solução para este problema é utilizar um termo que penalize a complexidade do modelo. Isto significa que o *score* de uma estrutura pode ser definido como um *trade-off* entre quão bom é o modelo e a complexidade/tamanho da estrutura de rede de preferências. No algoritmo *PrefK2** propomos um método que utiliza tal justificativa como premissa. Desde modo, o algoritmo *PrefK2** é também a composição de uma algoritmo de busca (o mesmo algoritmo de busca utilizado no *PrefK2*) e uma função *score*, sendo que a função *score* do algoritmo *PrefK2** leva em consideração também a complexidade da topologia de rede de preferências.

Assim como o algoritmo *PrefK2*, o algoritmo *PrefK2** possui como entrada uma ordem entre os atributos e uma base de amostras de preferências e como saída uma topologia de rede preferências. Na Figura 5.6 é apresentado o esquema do algoritmo *PrefK2**.

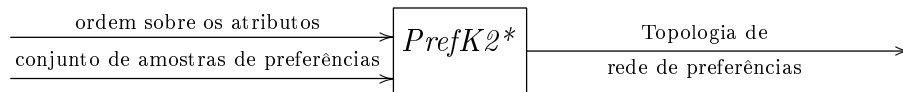


Figura 5.6: Esquema do algoritmo *PrefK2**

Em [Jensen e Nielsen 2007] é apresentada uma função de avaliação que satisfaz duas propriedades importantes necessárias ao contexto deste trabalho: ela faz um balanceamento entre a acurácia da estrutura e a complexidade da topologia, além de ser computacionalmente tratável, provendo escalabilidade dos dados.

Adaptamos a função disposta em [Jensen e Nielsen 2007] para o contexto de redes de preferências, sendo esta a equação 5.6 apresentada a seguir:

$$Score_{complex}(B_S, \mathcal{H}) = \sum_{i=1}^n \sum_{l=1}^m \frac{1}{m} \sum_{j=1}^{q_i^l} \left(N_{ij0} \log_2 \left(\frac{N_{ij0}}{N_{ij0} + N_{ij1}} \right) + N_{ij1} \log_2 \left(\frac{N_{ij1}}{N_{ij0} + N_{ij1}} \right) \right) - complex(B_S), \quad (5.6)$$

onde a complexidade da estrutura de rede de preferências $complex(B_S)$ é dada pela equação 5.7, que diz que a complexidade da rede é o somatório da complexidade de cada nó X_i :

$$complex(B_S) = \sum_{i=1}^n \sum_{l=1}^m q_i^l \cdot (r_i^l - 1) \cdot \frac{\log_2^{N_i^l}}{2}. \quad (5.7)$$

A complexidade do nó X_i é dada pela equação 5.8

$$complex(X_i) = \sum_{l=1}^m q_i^l \cdot (r_i^l - 1) \cdot \frac{\log_2^{N_i^l}}{2} \quad (5.8)$$

Desta forma podemos dizer que a complexidade da rede B_S é dada por

$$complex(B_S) = \sum_{i=1}^n complex(X_i). \quad (5.9)$$

Exemplo 5.3. Considere as duas redes de preferências sobre o esquema relacional $R(A, B, C, D)$ mostradas nas Figuras 5.7 e 5.8, as amostras de preferências da Tabela 4.3 e a Tabela 5.3 de valores N_{ij1} , N_{ij0} e N calculados com base nas partições da Figura 4.7.

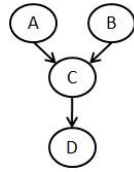


Figura 5.7: Rede de preferências B_{S1}

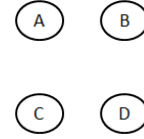


Figura 5.8: Rede de preferências B_{S2}

Calculando o $Score_{complex}(B_{S1}, \mathcal{H})$ a partir da Equação 5.6, tem-se que:

$$\begin{aligned} & Score_{complex}(B_{S1}, \mathcal{H}) \\ &= \frac{1}{2} \left(2 \cdot \log_2 \left(\frac{2}{2+1} \right) + 1 \cdot \log_2 \left(\frac{1}{2+1} \right) + 1 \cdot \log_2 \left(\frac{1}{1+0} \right) + 0 \cdot \log_2 \left(\frac{0}{0+1} \right) \right) \\ &+ \frac{1}{1} \left(2 \cdot \log_2 \left(\frac{2}{2+0} \right) + 0 \cdot \log_2 \left(\frac{0}{2+0} \right) \right) + \frac{1}{1} \left(1 \cdot \log_2 \left(\frac{1}{1+0} \right) + 0 \cdot \log_2 \left(\frac{0}{0+1} \right) \right) \\ &+ 1 \cdot \log_2 \left(\frac{1}{1+2} \right) + 2 \cdot \log_2 \left(\frac{2}{1+2} \right) + \frac{1}{1} \left(2 \cdot \log_2 \left(\frac{2}{2+1} \right) + 1 \cdot \log_2 \left(\frac{1}{1+2} \right) \right) \\ &- complex(B_{S1}) \end{aligned}$$

Mas, $complex(B_{S1})$

$$\begin{aligned} &= \left(1 \cdot (2-1) \cdot \frac{\log_2 3}{2} + 1 \cdot (1-1) \cdot \frac{\log_2 1}{2} \right) + \left(1 \cdot (1-1) \cdot \frac{\log_2 2}{2} \right) + \left(2 \cdot (2-1) \cdot \frac{\log_2 4}{2} \right) \\ &+ \left(1 \cdot (2-1) \cdot \frac{\log_2 5}{2} \right) \\ &= 0,7924 + 0 + 2,0 + 1,609640 \\ &= 3,95 \end{aligned}$$

Substituindo $complex(B_{S1})$ em $Score_{complex}(B_{S1}, \mathcal{H})$, temos que:

Dados da rede de preferências B_{S1} da Figura 5.7				
Partição	$Comp(A)$	$Comp(B)$	$Comp(C)$	$Comp(D)$
$l = 0$	$N_{ij0} = 2((a_1 > a_2) \text{ e } \pi_i = \emptyset),$ $N_{ij1} = 1 \text{ e } N^l = 3$	$N_{ij0} = 2((b_2 > b_1) \text{ e } \pi_i = \emptyset),$ $N_{ij1} = 0 \text{ e } N^l = 2$	$N_{ij0} = 1 ((c_1 > c_2) \text{ e } \text{instancias dos pais (valores para } \pi_i) = \{A = a_2 \text{ e } B = b_1\}),$ $N_{ij1} = 0 \text{ e } N^l = 4$	$N_{ij0} = 1 ((d_2 > d_1) \text{ e } \text{instancias dos pais (valores para } \pi_i) = \{C = c_1\}),$ $N_{ij1} = 2 \text{ e } N^l = 5$
$l = 0$			$N_{ij0} = 2 ((c_2 > c_1) \text{ e } \text{instancias dos pais (valores para } \pi_i) = \{B = b_1\}),$ $N_{ij1} = 1 \text{ e } N^l = 4$	
$l = 1$	$N_{ij0} = 1 ((a_2 > a_3) \text{ e } \pi_i = \emptyset),$ $N_{ij1} = 0 \text{ e } N^l = 1$			
Dados da rede de preferências B_{S2} da Figura 5.8				
Partição	$Comp(A)$	$Comp(B)$	$Comp(C)$	$Comp(D)$
$l = 0$	$N_{ij0} = 2((a_1 > a_2) \text{ e } \pi_i = \emptyset),$ $N_{ij1} = 1 \text{ e } N^l = 3$	$N_{ij0} = 2((b_2 > b_1) \text{ e } \pi_i = \emptyset),$ $N_{ij1} = 0 \text{ e } N^l = 2$	$N_{ij0} = 2 ((c_1 > c_2) \text{ e } \pi_i = \emptyset),$ $N_{ij1} = 2 \text{ e } N^l = 4$	$N_{ij0} = 3 ((d_2 > d_1) \text{ e } \pi_i = \emptyset),$ $N_{ij1} = 2 \text{ e } N^l = 5$
$l = 1$	$N_{ij0} = 1 ((a_2 > a_3) \text{ e } \pi_i = \emptyset),$ $N_{ij1} = 0 \text{ e } N^l = 1$			

Tabela 5.3: Cálculos N_{ij1} , N_{ij0} e N^l para cada $Comp(A_i)$

$$Score_{complex}(B_{S1}, \mathcal{H}) = -1,377 - 0 - 2,7548 - 2,7548 - 3,95 = -10,8363$$

Calculando o $Score_{complex}(B_{S2}, \mathcal{H})$ a partir da Equação 5.6, tem-se que:

$$\begin{aligned}
& Score_{complex}(B_{S2}, \mathcal{H}) \\
&= \frac{1}{2} \left(2 \cdot \log_2 \left(\frac{2}{2+1} \right) + 1 \cdot \log_2 \left(\frac{1}{2+1} \right) + 1 \cdot \log_2 \left(\frac{1}{1+0} \right) + 0 \cdot \log_2 \left(\frac{0}{0+1} \right) \right) \\
&+ \frac{1}{1} \left(2 \cdot \log_2 \left(\frac{2}{2+0} \right) + 0 \cdot \log_2 \left(\frac{0}{2+0} \right) \right) + \frac{1}{1} \left(2 \cdot \log_2 \left(\frac{2}{2+2} \right) + 2 \cdot \log_2 \left(\frac{2}{2+2} \right) \right) \\
&+ \frac{1}{1} \left(2 \cdot \log_2 \left(\frac{2}{2+3} \right) + 3 \cdot \log_2 \left(\frac{3}{2+3} \right) \right) - complex(B_{S2})
\end{aligned}$$

Mas, $complex(B_{S2})$

$$\begin{aligned}
&= \left(1 \cdot (2-1) \cdot \frac{\log_2 3}{2} + 1 \cdot (1-1) \cdot \frac{\log_2 1}{2} \right) + \left(1 \cdot (1-1) \cdot \frac{\log_2 2}{2} \right) + \left(1 \cdot (2-1) \cdot \frac{\log_2 4}{2} \right) \\
&+ \left(1 \cdot (2-1) \cdot \frac{\log_2 5}{2} \right) \\
&= 0,7924 + 0 + 1,0 + 1,1609640 \\
&= 2,953364
\end{aligned}$$

Substituindo $complex(B_{S2})$ em $Score_{complex}(B_{S2}, \mathcal{H})$, temos que:

$$Score_{complex}(B_{S1}, \mathcal{H}) = -1,377 - 0 - 4,0 - 4,8547 - 2,953364 = -13,178064$$

Portanto, levando em consideração o $Score_{complex}$ das estruturas de preferências, a topologia que melhor representa os dados é a B_{S1} tendo em vista que é $Score_{complex}(B_{S1}, \mathcal{H}) > Score_{complex}(B_{S2}, \mathcal{H})$.

O algoritmo $PrefK2^*$ assume uma ordem entre os atributos e também assume que a priori todas as estruturas são igualmente prováveis. Ele busca para cada atributo A_i um conjunto de pais que maximize a função $g^*(i, \pi_i)$.

$$g^*(i, \pi_i) = \sum_{l=1}^m \frac{1}{m} \sum_{j=1}^{q_i^l} \left(N_{ij0} \log_2 \left(\frac{N_{ij0}}{N_{ij0} + N_{ij1}} \right) + N_{ij1} \log_2 \left(\frac{N_{ij1}}{N_{ij0} + N_{ij1}} \right) \right) - complex(X_i) \quad (5.10)$$

Assim como o algoritmo $PrefK2$, o algoritmo $PrefK2^*$ proposto neste trabalho é uma heurística gulosa, onde iniciamos com uma estrutura vazia e avaliamos o ganho para cada nova aresta adicionada ao mesmo. Na Figura 5.9 a seguir é apresentado o algoritmo $PrefK2^*$ para aprendizado de uma topologia de rede de preferências.

Entrada: Um conjunto n de atributos discretos $A_i \in R(A_1, \dots, A_n)$, uma ordem sobre os atributos A_i, um banco de dados de k amostras de preferências em R. Saída: Um grafo B_S com os vértices em $A_1, \dots, A_n$.
1. Para i de 1 até n faça 2. Construa as partições P para cada $Comp(A_i)$ 3. $\Pi_i := \phi$; //acumula os pais das partições de A_i 4. Para cada partição $p_j \in P$ 5. $\pi_i := \phi$; //armazena os melhores pais da partição 6. $Score_{old} := g^*(i, \pi_i)$; 7. $Proceder := \mathbf{TRUE}$; 8. Enquanto $Proceder$ faça 9. Dado Z o conjunto dos nós em $Predecessores(A_i) - \pi_i$ que maximiza $g^*(i, \pi_i \cup \{Z\})$; 10. $Score_{new} := g^*(i, \pi_i \cup \{Z\})$; 11. Se $Score_{new} > Score_{old}$ então 12. $Score_{old} := Score_{new}$; 13. $\pi_i := \pi_i \cup \{Z\}$; 14. Senão $Proceder := \mathbf{FALSE}$ 15. $\Pi_i := \Pi_i \cup \pi_i$ 16. Escreva ('No:', X_i, 'Pais deste nó', π_i)

Figura 5.9: Algoritmo $PrefK2^*$

5.2 Treinamento da rede de preferências

O treinamento de uma rede de preferências assim como o treinamento de uma rede bayesiana pode ser feito de várias maneiras. Entretanto, como enfatizado no Capítulo

2, o *princípio da verossimilhança máxima* é o que apresenta melhores resultados. Tal princípio diz que a melhor forma de calcular as tabelas de probabilidade é realizando a contagem (frequência) dos valores.

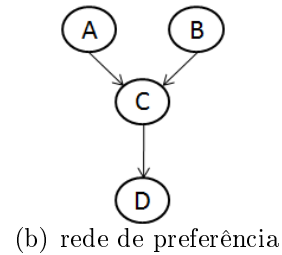
Definição 5.4. (Frequência) A frequência de uma preferência sobre um Atributo X_i , denotada por $freq(X_i = a_1 < X'_i = a_2)$, onde $dom(X_i) = dom(X'_i) = \{a_1, a_2\}$, corresponde ao número de vezes que o par de atributos X_i, X'_i , com $X_i \neq X'_i$, aparece no conjunto de amostras de preferências com atributo $pref = 1$ dividido pelo número total de ocorrências de pares onde têm-se $X_i = a_1$ e $X'_i = a_2$ ou $X_i = a_2$ e $X'_i = a_1$. Por exemplo, supondo os atributos A, A' da relação de preferências da Tabela 5.10(a), a frequência $freq(A = a_1 < A' = a_2) = 2/4 = 0,5$ ou em 100% dos pares de tuplas contendo os valores de $A = a_1$ ou $A = a_2$ e $A' = a_1$ ou $A' = a_2$, 50% desses casos têm a preferência $A = a_1 < A' = a_2$.

Para exemplificar como foi feito o treinamento de uma rede de preferências considere o Exemplo 5.4 a seguir:

Exemplo 5.4. Seja a estrutura de rede de preferências da Figura 5.10(b) e o conjunto de amostras de preferências da Figura 5.10(a), realizar o treinamento da rede:

id	A	B	C	D	A'	B'	C'	D'
1	a_1	b_1	c_1	d_2	a_2	b_2	c_1	d_1
2	a_1	b_2	c_1	d_2	a_2	b_1	c_1	d_1
3	a_2	b_2	c_1	d_1	a_1	b_1	c_1	d_2
4	a_2	b_1	c_3	d_3	a_2	b_1	c_2	d_1
5	a_2	b_1	c_2	d_3	a_2	b_1	c_1	d_1

(a) Amostras de preferências



(b) rede de preferência

Figura 5.10: Banco de Amostras de preferências espelhadas do usuário e a respectiva rede de preferências inferida

O cálculo das tabelas de probabilidade é feito a partir do cálculo da frequência dos valores de seus atributos, sendo assim teremos:

1. Tabelas do atributo A: o atributo A não depende de nenhum outro atributo, segundo o grafo de dependências da Figura 5.10(b). Sendo assim contabilizamos aqueles valores onde o atributo A é diferente de A', ou seja, são contabilizados o número de vezes em que o atributo $A = a_1$ e $A' = a_2$, dentre todas as *bituplas* onde $A \neq A'$ e $A = a_1$ ou $A = a_2$ e $A' = a_1$ ou $A' = a_2$. Por exemplo: $P[A = a_1 > A = a_2] = 2/3 = 0,66$ e a $P[A = a_2 > A = a_1] = 1/3 = 0,33$;
2. Tabelas do atributo B: o atributo B recai sobre o mesmo caso do atributo A, isto é, como não possui pais, é necessário calcular a frequência dos valores onde o atributo B é diferente de B'. Por exemplo: $P[B = b_1 > B = b_2] = 0,33$ e $P[B = b_2 > B = b_1] = 0,66$.

3. Tabelas do atributo C: o atributo C possui pais A e B, segundo o grafo de dependências da Figura 5.10(b). Sendo assim, para calcular a tabela de probabilidade é necessário contabilizarmos a frequência somente quando os atributos $A=A'$ e $B=B'$ e $C \neq C'$ das amostras de preferências. Então temos que: $P[C = c_3 > C = c_2 | A = a_2 \wedge B = b_1] = 1$ e $P[C = c_2 > C = c_1 | A = a_2 \wedge B = b_1] = 1$.
4. Tabelas do atributo D: o atributo D possui pai C, sendo assim é necessário que o atributo C seja igual a C' e $D \neq D'$. Sendo assim, temos que: $P[D = d_2 > D = d_1 | C = c_1] = 0,66$ e $P[D = d_1 > D = d_2 | C = c_1] = 0,33$.

Definição 5.5. (Suporte de uma regra de preferência) Como foi falado até o momento, o treinamento de uma rede de preferência é realizado a partir de um banco de dados de amostras de preferências e uma topologia de rede de preferência. Entretanto, uma regra de preferência que aparece uma única vez no banco de dados pode não ser interessante para o usuário. A definição de *suporte* de uma regra de preferência veio exatamente para solucionar esse tipo de problema, onde explicitamente o suporte é o número de vezes que uma regra de preferência precisa aparecer no banco de dados de amostras para ser relevante ao usuário.

5.3 Considerações finais

Neste capítulo foram abordados os métodos de aprendizagem de redes de preferências propostos nesta dissertação. Os métodos de aprendizagem da estrutura de uma rede de preferências propostos são baseados nos métodos de aprendizagem de redes bayesianas classificados sob o enfoque conhecido como *baseado em scores*. Tais métodos possuem dois componentes: uma função *score* e um método de busca. Neste trabalho realizamos testes com duas funções *scores* e o método de busca *K2* proposto em [Cooper e Dietterich 1992], chamamos os dois algoritmos gerados a partir dessas combinações de *PrefK2* e *PrefK2**. Ainda neste capítulo falamos sobre o treinamento de uma rede de preferências. O treinamento de uma rede de preferências é feito baseando-se no *princípio da verossimilhança máxima*.

Capítulo 6

Validação dos métodos propostos

Neste capítulo será abordado aspectos relevantes quanto à construção dos dados sintéticos e o uso dos dados reais, e ainda apresentaremos os resultados experimentais obtidos na validação dos métodos propostos.

6.1 Dados utilizados na validação

A fim de validar o modelo proposto neste trabalho, dois conjuntos de dados para teste são utilizados: dados sintéticos construídos através do uso de uma técnica de simulação estocástica adaptada para o nosso contexto e dados reais obtidos com um *benchmark* para personalização de consultas. Nas seções a seguir apresentamos uma descrição dos conjuntos de dados sintéticos e dos dados reais.

6.1.1 Dados sintéticos

Para a validação dos algoritmos *PrefK2* e *PrefK2** de aprendizado da topologia, foi criada uma rotina que gera topologias diversas e tabelas de probabilidades randômicas as quais em conjunto chamamos de rede de *preferência de input* (ver Figura 6.1). A partir de uma rede de preferências de *input* são geradas amostras de preferências sintéticas. Tais amostras estão estatisticamente coerentes com a rede de preferências em questão, isto é, as amostras de preferências refletem a distribuição probabilística da rede de preferências original.

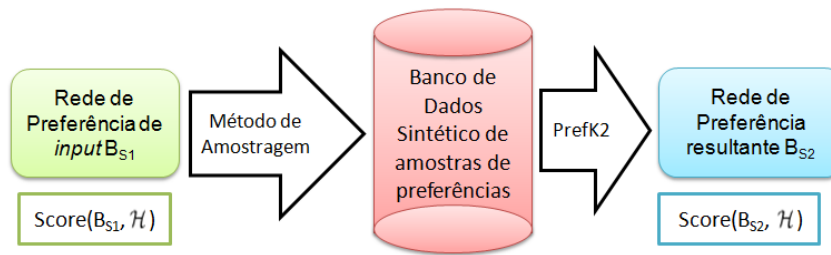


Figura 6.1: Validação

Nos algoritmos de simulação estocástica (também chamados de Amostragem Monte Carlo, Simulação Estocástica, ou Simulação Randômica), a probabilidade de um evento acontecer é estimada usando a frequência com que ocorre em um conjunto de *amostras*, tendo em vista que a simulação estocástica parte do pressuposto que não se tem acesso ao banco de dados e tenta-se simular esse acesso gerando amostras.

Diferenças nos algoritmos de amostragem estão em principalmente como as amostras são geradas e em como as probabilidades são estimadas a partir das configurações amostradas [Jensen e Nielsen 2007].

Para construção dos dados sintéticos adaptamos a técnica denominada *Amostragem Lógica Probabilística* (*Probabilistic Logic Sampling*) definida em [Henrion 1986]. Tal técnica é uma das primeiras propostas na literatura. Na Figura 6.1, mostramos como o processo de validação foi realizado.

Descrição do processo de geração de amostras sintéticas

Cada atributo é randomicamente instanciado para um dos seus possíveis valores seguindo a probabilidade de seus estados, dado os estados instanciados de seus pais. Cada instanciação realizada é feita seguindo a ordem topológica, ou seja, os pais são amostrados antes de seus filhos. O método original de amostragem para redes bayesianas sugere a possibilidade de usar fatores externos como a experiência humana para melhorar o resultado da amostragem, entretanto, isto não foi utilizado na nossa abordagem.

Para exemplificar o processo de geração de amostras sintéticas utilizaremos a rede de preferências definida na Figura 6.2.

Em uma única passagem pela rede de preferências de *input* da Figura 6.2 é possível obter as amostras randômicas da Tabela 6.1.

Amostra	A	B	C	D	A'	B'	C'	D'	Pref
1	a2	b2	c1000	d1000	a1	b1	c'1000	d'1000	1
2	a1	b1	c2	d1001	a1	b1	c1	d'1001	1
3	a1002	b1002	c1	d2	a1002	b1002	c1	d1	1

Tabela 6.1: Amostras de preferências obtidas pelo método *Probabilistic Logic Sampling*.

Para um maior entendimento do método, considere a primeira amostra da Tabela 6.1, ela é referente ao primeiro nível da rede de preferências que é constituído pelos

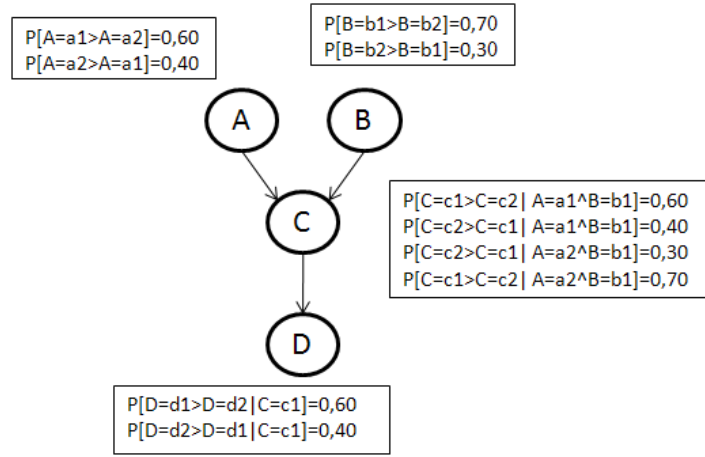


Figura 6.2: Rede de preferência de *input* (Gerada aleatoriamente)

atributos A e B da rede de preferências de *input* da Figura 6.2. Na primeira amostra, randomicamente foi obtido o valor a2 para o atributo A e a1 para o atributo A', e b2 para o atributo B e b1 para o atributo B'. Como tanto A quanto B não possuem pais os demais valores são preenchidos com preferências não pertencentes ao domínio dos dados e que aparecerão somente uma vez no banco de dados inteiro. Isto garante que os valores de A e B nessa tupla não dependerão de nenhum outro atributo, e portanto está coerente com a rede de preferências de *input*.

Para níveis que possuem mais de um atributo a seleção randômica se dá pela probabilidade conjunta destes atributos, isto é, considerando ainda os atributos A e B do primeiro nível da rede teríamos as seguintes probabilidades de seus valores ocorrerem conjuntamente:

- $P[A = a1 > A = a2] * P[B = b1 > B = b2] = 0,60 * 0,70 = 0,42$
- $P[A = a1 > A = a2] * P[B = b2 > B = b1] = 0,60 * 0,30 = 0,18$
- $P[A = a2 > A = a1] * P[B = b1 > B = b2] = 0,40 * 0,70 = 0,28$
- $P[A = a2 > A = a1] * P[B = b2 > B = b1] = 0,40 * 0,30 = 0,12$

Deste modo é selecionado randomicamente um valor entre 0 e 1, sendo que o valor obtido nos dirá qual preferência será inserida na amostra, conforme Figura 6.3.

No caso em que os atributos possuem pais, é gerado também randomicamente valores para seus pais, e os demais atributos são preenchidos com preferências não pertencentes ao domínio dos dados e não usadas anteriormente. Isto garante que os valores do atributo em questão nessa tupla só dependerão dos seus respectivos pais, e portanto está coerente com a rede de preferências de *input*.

Para cada nível são geradas por passagem um número de amostras referentes ao número de regras de dependência combinadas entre os atributos do nível. Por exemplo, no primeiro nível da rede de preferência da Figura 6.2 temos para o atributo A uma regra

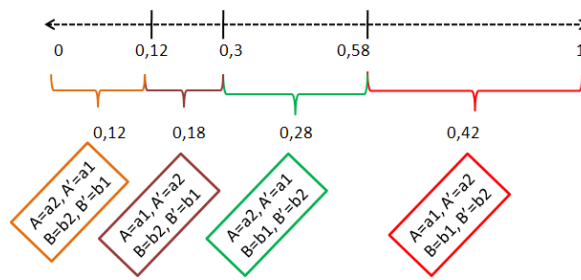


Figura 6.3: Intervalo numérico para geração de amostras do atributo A e B referente ao primeiro nível da rede de preferências da Figura 6.2

de dependência (referente a $A = a1$ e $A = a2$) e uma regra de dependência referente ao atributo B (referente a $B = b1$ e $B = b2$), o que nos leva a ter somente uma *bitupla* para cada passagem.

Para que preferências do tipo $b1000 > b'1000$ não apareçam nas preferências mineradas foi criado um *threshold* que funciona como um suporte (ver Definição 5.5), onde garantimos que preferências com suporte baixo como $b1000 > b'1000^1$ não apareçam nos resultados.

6.1.2 Dados reais

Os dados reais para validação do método proposto nesta dissertação foram obtidos a partir do *benchmark* para personalização de consultas disponível em <http://apmd.prism.uvsq.fr/-SubProject4/TestPlatform/IntegratedDB.html>. Tal *benchmark* é um banco de dados relacional que integra dados dos sites *MovieLens* para recomendação de filmes (<http://www.movielens.org>) e IMDb (<http://www.imdb.com>) também sobre filmes e foi construído exclusivamente para o projeto APMD (<http://apmd.prism.uvsq.fr>) de personalização de conteúdo em grandes massas de dados. Devido o fato de preferências condicionais serem ainda um tópico de pesquisa bastante novo, não existem dados construídos efetivamente para lidar com este tipo de preferências, devido a isso, utilizamos então os dados do *benchmark* do projeto APMD para buscar preferências condicionais.

O banco de dados considerado tem 6.040 usuários que avaliam 3.881 filmes. O esquema do banco de dados consiste de 52 tabelas descrevendo filmes, empresas e pessoas relativas à filmes, além de um atributo referente à preferência do usuário em relação ao filme em questão. Atributos referentes ao usuário não foram considerados, uma vez que nosso interesse inicial diz respeito à preferências condicionais em relação aos filmes e não em relação a atributos do usuário. A Figura 6.4 diz respeito às tabelas utilizadas para os testes deste trabalho.

Como neste trabalho estamos interessados nas preferências relativas às escolhas de um usuário em específico, fez-se necessário separar as preferências por id do usuário. Na Tabela 6.2 apresentamos os usuários com maior número de preferências.

¹pelo método de amostragem garantimos que a preferência $b1000 > b'1000$ só aparece uma única vez

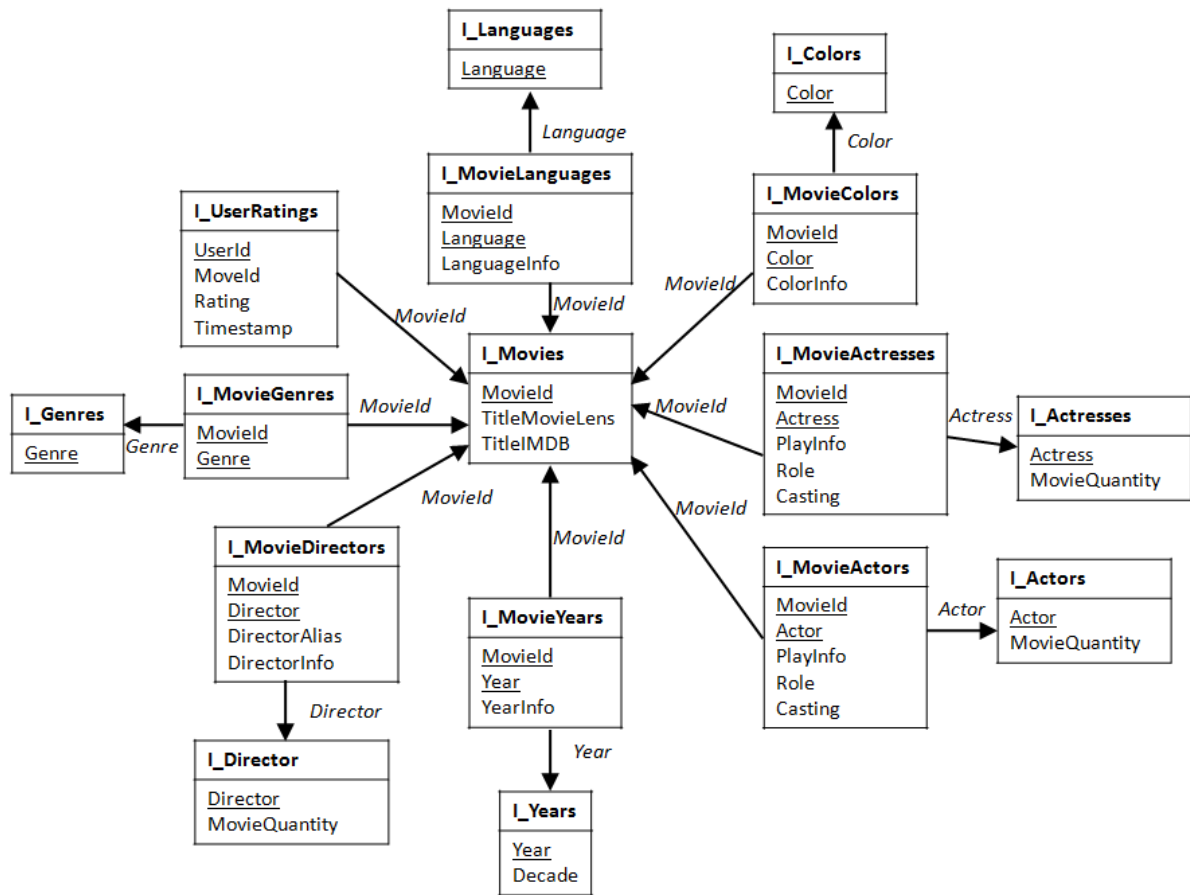


Figura 6.4: Parte do esquema relacional do *benchmark* utilizado para testes em dados reais com a nossa abordagem

IdUsuário	Nº de filmes avaliados
4169	2308
1680	1846
4277	1739
1971	1591
1941	1591
1181	1518
3618	1341

Tabela 6.2: Usuários × número de Filmes avaliados

Para os testes foram utilizados somente os atributos: atriz (principal), gênero, ator (principal), diretor (principal), década, linguagem e cor.

Na tabela *I_UserRatings* é armazenado as notas dadas pelo usuário a cada filme (*ranking* de 1 a 5). Sendo assim, foi necessário adequar as escolhas do usuário ao formato de entrada do nosso método. Para exemplificar a formatação necessária considere a Tabela a seguir povoada com algumas instâncias de *I_UserRatings*.

A seguir descrevemos o pré-processamento realizando nos dados, incluindo junção de

UserId	MovieId	Rating
1	1	5
1	4	2
1	6	3
1	2	3

Tabela 6.3: tabela $I_UserRatings$

tabelas e formatação dos dados em pares de tuplas.

Pré-processamento dos dados reais

Para transformar as instâncias da Tabela $I_UserRatings$ (um exemplo de instâncias dessa tabela pode ser visto na Tabela 6.3) em amostras de preferências, inicialmente ordenamos a tabela $I_UserRatings$. Isto se dá para ser possível separar em arquivos onde todas as avaliações feitas sejam de um único usuário.

Tendo as avaliações feitas sobre os diversos filmes separadas por usuário, foi observado que os vários atributos possuíam valores multivalorados. Para facilitar o tratamento de preferências transformamos os atributos com *valores multivalorados* em atributos *simples*. Um exemplo de atributos multivalorados no contexto do *benchmark* utilizando diz respeito ao atributo *gênero*, onde um dado filme pode ser tanto do *gênero comédia* quanto do *gênero romance*. Logo realizamos a correspondência $Gênero = comédia, romance$ que passará a existir no banco como $Gênero = G1$.

Após a transformação de atributos multivalorados em atributos simples, foi feita uma junção de todas as tabelas criando o esquema relacional $Ratings(Gênero, Ator, Atriz, Diretor, Década, Linguagem, Cor, Ranking)$. Uma relação com este esquema pode ser vista na Tabela 6.4.

Gên	Ator	Atriz	Dir	Déc	Língua	Cor	Ranking
G1	John	Mary	Bob	50	Inglês	Color	5
G2	John	Mary	Bob	40	Francês	BW	2
terror	Pablo	Beth	Charles	60	Espanhol	Color	3
drama	Fred	Katy	Charles	50	Inglês	Color	3

Tabela 6.4: Pré-processamento intermediário

De posse do esquema relacional $Ratings$, uma comparação entre pares de tuplas pertencentes a este esquema pelo atributo *Ranking* de cada uma deve ser realizado. Se o *Ranking* da primeira tupla é *maior* que o *Ranking* da segunda tupla então concluímos que *a primeira tupla é preferida à segunda tupla*. Se a segunda tupla possui *maior Ranking* então ela é preferida à primeira tupla. E se elas possuem o valores de *rankings* iguais elas foram consideradas incomparáveis e não aparecem no banco de dados de amostras de preferências. A preferência é explicitada pelo acréscimo do atributo *Pref* de preferência. O atributo *Pref* recebe valor **1** quando *a primeira tupla é preferida a segunda* e **0** caso contrário.

Desde modo transformamos uma tabela como a tabela 6.4 em uma tabela de amostras de preferências como a Tabela 6.5:

Gên	Ator	Atriz	Dir	Déc	Lingua	Cor	Gên'	Ator'	Atriz'	Dir'	Déc'	Lingua'	Cor'	Pref
G1	John	Mary	Bob	50	Inglês	Color	G2	John	Mary	Bob	40	Francês	BW	1
G1	John	Mary	Bob	50	Inglês	Color	terror	Pablo	Beth	Charles	60	Espanhol	Color	1
G1	John	Mary	Bob	50	Inglês	Color	drama	Fred	Katy	Charles	50	Inglês	Color	1
G2	John	Mary	Bob	40	Francês	BW	terror	Pablo	Beth	Charles	60	Espanhol	Color	0
G2	John	Mary	Bob	40	Francês	BW	drama	Fred	Katy	Charles	50	Inglês	Color	0

Tabela 6.5: Amostras de preferências obtidas pelo pré-processamento

Os dados reais já processados e utilizados nesta dissertação para a validação dos métodos podem ser acessados em <http://www.lsi.ufu.br/cprefminer/downloads>.

6.2 Resultados Experimentais

Nesta seção abordamos os testes realizados para a validação dos métodos propostos nesta dissertação. Enfatizamos que todos os testes foram executados em um computador com processador Intel Quad Core Q9550 (2.83 GHz) com 4 Gigabytes de memória. Utilizamos o sistema operacional Windows e linguagem de implementação Java. A fim de facilitar a leitura, antes de abordarmos os resultados, relembremos as medidas de avaliação consideradas na comparação dos algoritmos e a técnica de amostragem utilizada nos testes com os algoritmos *PrefK2* e *PrefK2**.

6.2.1 Medidas consideradas na comparação dos algoritmos

Repetimos as definições de Precisão e Acurácia (Definições 4.5 e 4.6) dadas no Capítulo 4 para facilitar o trabalho do leitor.

Precisão

A precisão diz respeito ao número de *bituplas* ordenadas corretamente dividido pelo número total de *bituplas* que o ordenador conseguiu ordenar, considerando que ele pode não conseguir algumas tuplas.

Acurácia

Considera-se aqui que uma *bitupla* sempre será ordenada. A acurácia será então o número de *bituplas* ordenadas corretamente sobre o número total de *bituplas*. No Exemplo 4.8, para o cálculo da acurácia realizamos somente o cálculo das probabilidades conjuntas em vermelho, o maior valor entre as duas probabilidades nos dá a preferência, isto é, para a *bitupla* do exemplo o atributo Pref é igual a 1 (a primeira tupla é preferida à segunda).

Performance

A *performance* diz respeito ao tempo de aprendizagem da estrutura (topologia) da rede de preferência, tendo em vista que algoritmo para aprendizado das tabelas de probabilidade (treinamento da rede de preferências) foi o mesmo tanto para o algoritmo *PrefK2* quanto para o *PrefK2**.²

6.2.2 Técnica de amostragem: *cross-validation*

Cross-validation ou *validação cruzada* é um método estatístico de avaliação e comparação de algoritmos de aprendizagem pela divisão dos dados em dois segmentos: um usado para aprender ou treinar o modelo e o outro usado para validar o modelo.

O método mais simples da validação cruzada é conhecido como *k-fold cross-validation*, onde inicialmente o banco de dados é dividido em k partes $D_1, \dots, D_k$ de tamanhos aproximados. Posteriormente, k iterações de treinamento e validação são executadas de tal forma que em cada iteração uma diferente partição de dados é usada para validação enquanto as restantes são usadas para aprendizagem. Para um maior entendimento do método *k-fold cross-validation*, consideremos um exemplo onde $k=3$ na Figura 6.5.

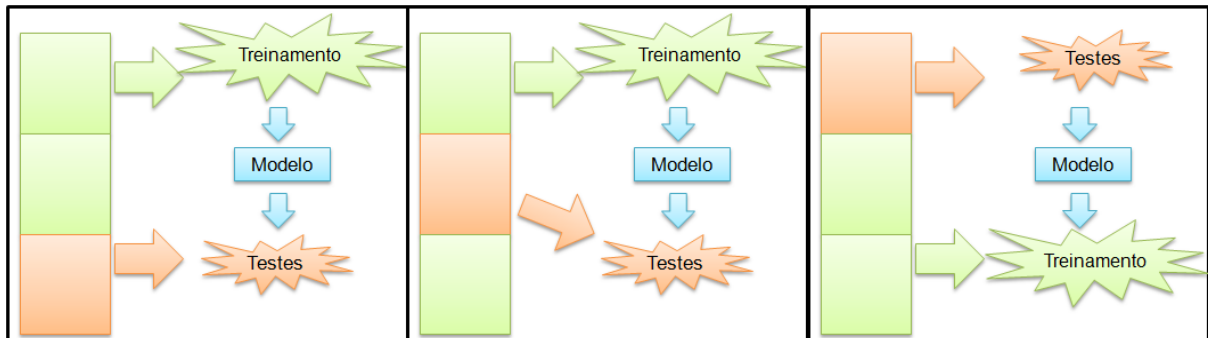


Figura 6.5: *3-fold cross-validation*

Desta forma, cada tupla do banco de dados é utilizada o mesmo número de vezes como tupla de treinamento e uma única vez como tupla de teste. Assim, a *acurácia do método* (ou mesmo a *precisão* e a *performance*) é medida como o número de tuplas bem ordenadas nas k iterações dividido pelo total de tuplas no banco de dados original. No nosso contexto consideramos o parâmetro k igual a 5.

6.2.3 Resultados em dados sintéticos

Apresentamos nesta seção os resultados obtidos pelos nossos métodos com os dados sintéticos. Faremos uma análise acerca de tais resultados ao final desta seção.

²A aprendizagem das tabelas de probabilidade se dá pela contagem, isto é frequência dos eventos no banco de dados como visto no Capítulo 5.

Para realização dos testes em dados sintéticos foram gerados um conjunto de N amostras compatíveis com uma rede de preferência $M(S, \theta)$. Sendo que os parâmetros descritos a seguir foram considerados para variação dos testes realizados.

- N = número de amostras;
- P = número de atributos da rede de preferência S ;
- pa = número máximo de pais que um nó de S pode ter;

Consideramos o suporte (ver Definição 5.5) nos dados sintéticos como sendo o valor 1, isto é, regras que aparecem uma única vez no banco de dados de amostras não são interessantes para o usuário.

Parâmetro	1	2	3	4	5	6	Default
N	320	1.024	3.008	10.048	30.080	50.000	10.024
P	4	6	8	10	15	20	10
pa	2	3	4	5	6	-	4

Tabela 6.6: Parâmetros considerados nos dados sintéticos

O cálculo da acurácia, relembrando a Definição dada em 4.5 diz respeito à porcentagem de pares tuplas bem ordenados em cada partição. Sendo assim, na Tabela 6.7 o valor para acurácia = 0,8875 alcançado na primeira linha diz respeito à porcentagem de pares de tuplas bem ordenados, realizando o cálculo: $(58 + 57 + 57 + 56 + 56)/320$.

A precisão, como já abordado na Definição 4.6, diz respeito ao número de amostras bem ordenadas dentre as amostras que o ordenador conseguiu ordenar. Sendo assim, na Tabela 6.9, o valor de precisão 0,962547 na primeira linha, é alcançado realizando o somatório das amostras bem ordenadas em cada rodada (partição) dividido pelo somatório de amostras que o ordenador conseguiu ordenar, ou seja, $(56 + 51 + 48 + 50 + 52)/(58 + 53 + 49 + 53 + 54)$.

Variando o número de amostras

Apresentamos nesta seção uma sequência de resultados obtidos com a execução dos algoritmos *PrefK2* e *PrefK2** variando o número de amostras de preferências conforme Tabela 6.6.

Como foi dito na seção anterior a técnica de amostragem usada foi a *k-fold cross-validation* com $k=5$. Deste modo, o termo rodada nas tabelas seguintes referentes aos resultados diz respeito à partição e varia de 1 a 5.

Nº de Amostras	1ª Rodada	2ª Rodada	3ª Rodada	4ª Rodada	5ª Rodada	Acurácia
320	58	57	57	56	56	0,8875
1.024	180	186	183	182	189	0,8984375
3.008	539	543	533	534	544	0,895279255
10.048	1.836	1.793	1.796	1.777	1.787	0,894605892
30.080	5.416	5.375	5.405	5.372	5.380	0,89587766
50.000	8.975	8.955	8.904	8.972	8.957	0,89526

Tabela 6.7: Acurácia de *PrefK2* variando o número de amostras

Nº de Amostras	1ª Rodada	2ª Rodada	3ª Rodada	4ª Rodada	5ª Rodada	Acurácia
320	58	58	57	53	56	0,88125
1.024	179	186	184	182	189	0,8984375
3.008	546	543	533	534	540	0,896276596
10.048	1.836	1.793	1.796	1.777	1.797	0,895601115
30.080	5.416	5.375	5.405	5.372	5.380	0,89587766
50.000	8.975	8.955	8.904	8.972	8.957	0,89526

Tabela 6.8: Acurácia de *PrefK2** variando o número de amostras

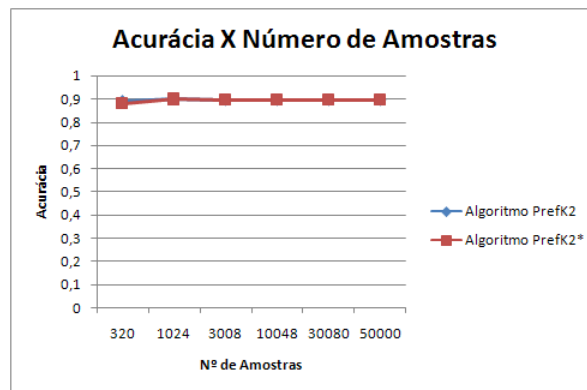


Figura 6.6: Gráfico Acurácia X N° de amostras

Nº de Amostras	1ª Rodada		2ª Rodada		3ª Rodada		4ª Rodada		5ª Rodada		Precisão
	Corretas	Total	Corretas	Total	Corretas	Total	Corretas	Total	Corretas	Total	
320	56	58	51	53	48	49	50	53	52	54	0,962547
1.024	158	168	155	166	153	161	162	177	164	175	0,935065
3.008	470	497	474	502	451	481	456	481	470	497	0,944264
10.048	1.579	1.655	1.549	1.638	1.532	1.636	1.536	1.629	1.549	1.654	0,943132
30.080	4.673	4.941	4.655	4.953	4.668	4.951	4.621	4.925	4.668	4.950	0,94195
50.000	7.791	8.241	7.698	8.180	7.633	8.155	7.700	8.154	7.688	8.173	0,941496

Tabela 6.9: Precisão *PrefK2* variando o número de amostras

Nº de Amostras	1ª Rodada		2ª Rodada		3ª Rodada		4ª Rodada		5ª Rodada		Precisão
	Corretas	Total	Corretas	Total	Corretas	Total	Corretas	Total	Corretas	Total	
320	56	58	51	53	48	49	50	53	52	54	0,962547
1.024	158	168	155	166	154	161	162	177	164	175	0,936246
3.008	470	497	474	502	451	481	456	481	470	497	0,944264
10.048	1.579	1.655	1.549	1.638	1.532	1.636	1.536	1.629	1.549	1.654	0,943132
30.080	4.673	4.941	4.655	4.953	4.668	4.951	4.621	4.925	4.668	4.950	0,94195
50.000	7.791	8.241	7.698	8.180	7.633	8.155	7.700	8.154	7.688	8.173	0,941496

Tabela 6.10: Precisão *PrefK2** variando o número de amostras

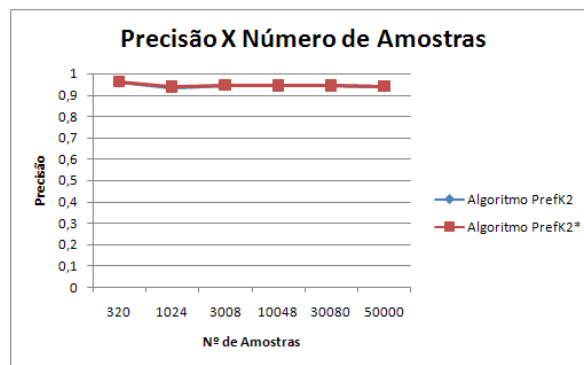
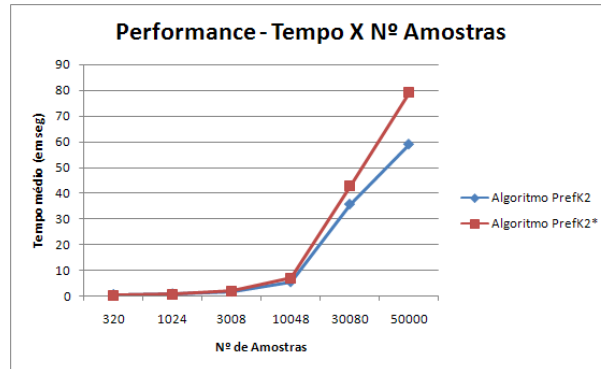


Figura 6.7: Gráfico Precisão X N° de amostras

Nº de Amostras	1ª Rodada	2ª Rodada	3ª Rodada	4ª Rodada	5ª Rodada	tempo em seg
320	468	458	484	515	468	0,47
1.024	733	718	734	718	718	0,7242
3.008	1.732	1.701	1.701	1.700	1.700	1,7068
10.048	5.397	5.382	5.303	5.366	5.303	5,3502
30.080	35.614	35.693	35.634	35.834	35.599	35,6748
50.000	59.187	58.953	59.171	58.999	59.108	59,0836

Tabela 6.11: *Performance PrefK2* variando o número de amostras

Nº de Amostras	1ª Rodada	2ª Rodada	3ª Rodada	4ª Rodada	5ª Rodada	tempo em seg
320	452	421	437	421	421	0,4304
1.024	842	889	843	843	842	0,8518
3.008	2.152	2.153	2.121	2.153	2.122	2,1402
10.048	7.066	6.896	6.911	7.020	6.927	6,964
30.080	42.682	42.651	42.589	43.652	42.754	42,8656
50.000	79.187	78.953	79.171	78.999	79.108	79,0836

Tabela 6.12: *Performance PrefK2** variando o número de amostrasFigura 6.8: *Performance X Número de Amostras*

Variando o número de atributos

Apresentamos nesta seção uma sequência de resultados obtidos com a execução dos algoritmos *PrefK2* e *PrefK2** variando o número de atributos dos bancos de dados de testes conforme Tabela 6.6, realizando uma comparação entre os resultados obtidos.

Nº de Atributos	1ª Rodada	2ª Rodada	3ª Rodada	4ª Rodada	5ª Rodada	Acurácia
4	1.776	1.781	1.789	1.763	1.776	0,8885
6	1.819	1.819	1.842	1.840	1.832	0,9152
8	1.966	1.969	1.969	1.975	1.964	0,9843
10	1.809	1.802	1.830	1.812	1.812	0,9065
15	1.871	1.891	1.877	1.892	1.889	0,942
20	1.908	1.913	1.904	1.913	1.921	0,9559

Tabela 6.13: Acurácia de *PrefK2* variando o número de atributos

Nº de Atributos	1ª Rodada	2ª Rodada	3ª Rodada	4ª Rodada	5ª Rodada	Acurácia
4	1.776	1.781	1.789	1.763	1.776	0,8885
6	1.819	1.819	1.842	1.840	1.832	0,9152
8	1.966	1.967	1.973	1.974	1.966	0,9846
10	1.801	1.802	1.829	1.805	1.805	0,9042
15	1.871	1.891	1.877	1.892	1.889	0,942
20	1.904	1.908	1.903	1.909	1.924	0,9548

Tabela 6.14: Acurácia de *PrefK2** variando o número de atributos

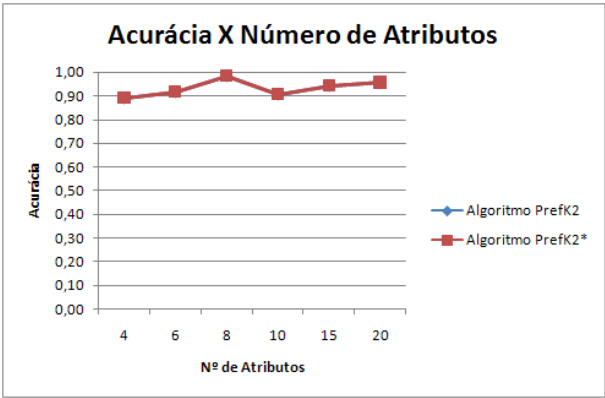


Figura 6.9: Gráfico Acurácia X N° de Atributos

Nº de Atributos	1ª Rodada		2ª Rodada		3ª Rodada		4ª Rodada		5ª Rodada		Precisão
	Corretas	Total	Corretas	Total	Corretas	Total	Corretas	Total	Corretas	Total	
4	1.526	1.653	1.550	1.670	1.556	1.669	1.520	1.646	1.539	1.663	0,926515
6	1.493	1.628	1.486	1.611	1.507	1.624	1.522	1.644	1.524	1.642	0,924285188
8	1.381	1.385	1.420	1.424	1.387	1.389	1.398	1.403	1.403	1.412	0,996577784
10	1.582	1.672	1.564	1.674	1.607	1.691	1.569	1.660	1.582	1.676	0,943986624
15	1.368	1.416	1.378	1.422	1.394	1.445	1.375	1.415	1.404	1.455	0,967286453
20	1.035	1.081	1.011	1.065	1.047	1.060	1.045	1.092	1.036	1.071	0,963680387

Tabela 6.15: Precisão de *PrefK2* variando o número de atributos

Nº de Atributos	1ª Rodada		2ª Rodada		3ª Rodada		4ª Rodada		5ª Rodada		Precisão
	Corretas	Total	Corretas	Total	Corretas	Total	Corretas	Total	Corretas	Total	
4	1.526	1.653	1.550	1.670	1.556	1.669	1.520	1.646	1.539	1.663	0,926514878
6	1.493	1.628	1.486	1.611	1.507	1.624	1.522	1.644	1.524	1.642	0,924285188
8	1.381	1.385	1.420	1.424	1.387	1.389	1.398	1.403	1.403	1.412	0,996577784
10	1.582	1.672	1.564	1.674	1.607	1.691	1.569	1.660	1.582	1.676	0,943986624
15	1.368	1.416	1.378	1.422	1.394	1.445	1.375	1.415	1.404	1.455	0,967286453
20	1.035	1.081	1.011	1.065	1.047	1.060	1.045	1.092	1.036	1.071	0,963680387

Tabela 6.16: Precisão *PrefK2** variando o número de atributos

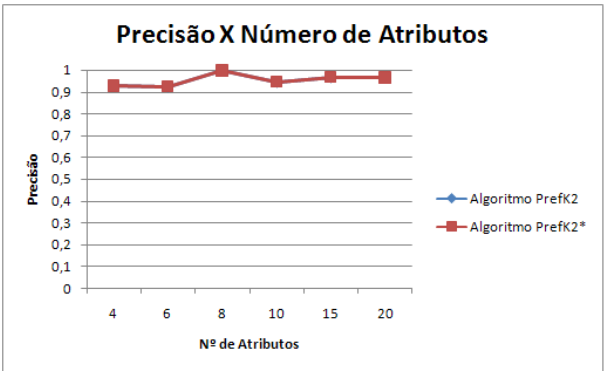
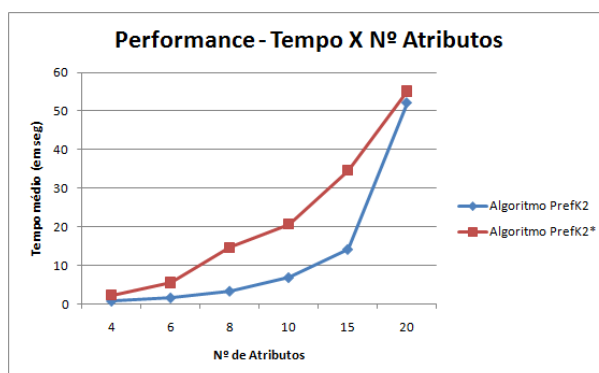


Figura 6.10: Precisão X Número de Atributos

Nº de Atributos	1ª Rodada	2ª Rodada	3ª Rodada	4ª Rodada	5ª Rodada	tempo em seg
4	827	827	827	811	811	0,8206
6	1.700	1.716	1.669	1.685	1.669	1,6878
8	3.385	3.385	3.401	3.385	3.416	3,3944
10	6.989	6.848	6.832	6.864	6.988	6,9042
15	20.521	12.536	12.568	12.630	12.708	14,1926
20	6.344	63304	63460	63305	63928	52,0682

Tabela 6.17: *Performance PrefK2* variando o número de atributos

Nº de Atributos	1ª Rodada	2ª Rodada	3ª Rodada	4ª Rodada	5ª Rodada	tempo em seg
4	2.224	2.309	2.324	2.340	2.309	2,3012
6	5.522	5.585	5.522	5.553	5.538	5,544
8	14.618	14.618	14.618	14.742	14.679	14,655
10	20.748	20.733	20.685	20.701	20.639	20,7012
15	34.648	34.618	34.649	34.617	34.633	34,633
20	55.039	55.147	55.365	55.365	55.038	55,1908

Tabela 6.18: *Performance PrefK2** variando o número de atributosFigura 6.11: *Performance X Numero de Atributos*

Variando a complexidade da rede de preferências

A complexidade da rede foi medida variando o número máximo de pais que um nó pode ter.

Nº máx de pais	1ª Rodada	2ª Rodada	3ª Rodada	4ª Rodada	5ª Rodada	Acurácia
2	1.760	1.765	1.790	1.808	1.771	0,8894
3	1.844	1.837	1.855	1.810	1.830	0,9176
4	1.786	1.805	1.794	1.784	1.802	0,8971
5	1.797	1.783	1.772	1.767	1.788	0,8907
6	1.724	1.733	1.729	1.757	1.714	0,8657

Tabela 6.19: Acurácia de *PrefK2* variando a complexidade da rede

Nº máx de pais	1ª Rodada	2ª Rodada	3ª Rodada	4ª Rodada	5ª Rodada	Acurácia
2	1.760	1.766	1.790	1.808	1.771	0,8895
3	1.844	1.845	1.854	1.814	1.830	0,9187
4	1.817	1.828	1.822	1.813	1.825	0,9105
5	1.803	1.783	1.772	1.767	1.788	0,8913
6	1.716	1.734	1.735	1.747	1.706	0,8638

Tabela 6.20: Acurácia de $PrefK2^*$ variando a complexidade da rede

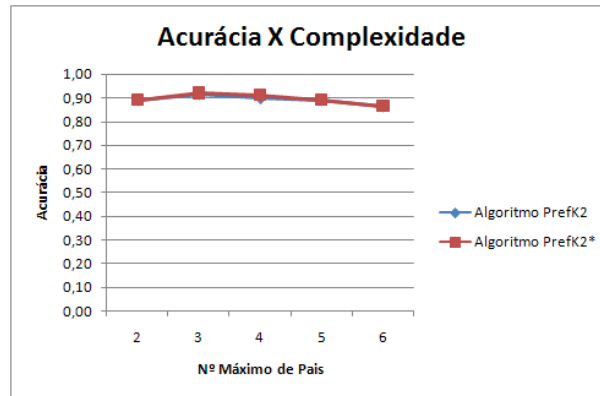


Figura 6.12: Gráfico complexidade da rede X acurácia

Nº máx de pais	1ª Rodada		2ª Rodada		3ª Rodada		4ª Rodada		5ª Rodada		Precisão
	Corretas	Total	Corretas	Total	Corretas	Total	Corretas	Total	Corretas	Total	
2	1.512	1.618	1.523	1.629	1.543	1.646	1.535	1.620	1.536	1.639	0,93829735
3	1.714	1.790	1.705	1.803	1.717	1.798	1.687	1.795	1.718	1.813	0,949105456
4	1.658	1.808	1.661	1.796	1.644	1.788	1.633	1.777	1.659	1.805	0,919879652
5	1.628	1.756	1.608	1.748	1.610	1.761	1.588	1.748	1.609	1.758	0,916999202
6	1.538	1.710	1.555	1.731	1.534	1.704	1.567	1.722	1.504	1.696	0,898984001

Tabela 6.21: Precisão de $PrefK2$ variando a complexidade da rede

Nº máx de pais	1ª Rodada		2ª Rodada		3ª Rodada		4ª Rodada		5ª Rodada		Precisão
	Corretas	Total	Corretas	Total	Corretas	Total	Corretas	Total	Corretas	Total	
2	1.512	1.618	1.523	1.629	1.543	1.646	1.535	1.620	1.536	1.639	0,93829735
3	1.714	1.790	1.705	1.803	1.717	1.798	1.687	1.795	1.718	1.813	0,949105456
4	1.684	1.808	1.683	1.796	1.672	1.788	1.662	1.777	1.681	1.805	0,934031647
5	1.628	1.756	1.608	1.748	1.610	1.761	1.588	1.748	1.609	1.758	0,916999202
6	1.538	1.710	1.555	1.731	1.534	1.704	1.567	1.722	1.504	1.696	0,898984001

Tabela 6.22: Precisão de $PrefK2^*$ variando a complexidade da rede

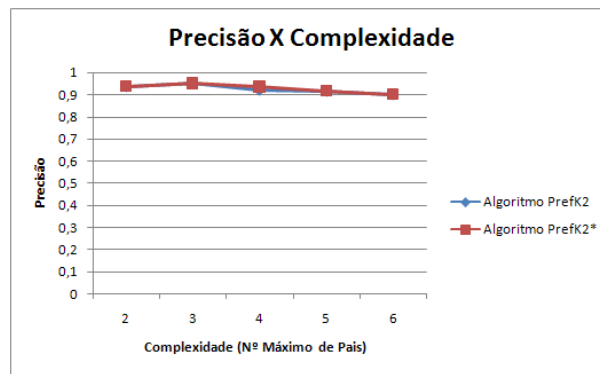


Figura 6.13: Precisão X Variação de Complexidade

Nº máx de país	1ª Rodada	2ª Rodada	3ª Rodada	4ª Rodada	5ª Rodada	tempo em seg
2	7.644	7.697	7.597	7.613	7.628	7,6358
3	8.268	8.314	8.362	8.362	8.283	8,3178
4	8.471	8.480	8.531	8.517	8.518	8,5034
5	8.799	8.767	8.798	8.799	8.798	8,7922
6	10.249	10.436	10.249	10.265	10.172	10,2742

Tabela 6.23: *Performance* de *PrefK2* variando a complexidade da rede de preferências

Nº máx de país	1ª Rodada	2ª Rodada	3ª Rodada	4ª Rodada	5ª Rodada	tempo em seg
2	22.370	22.246	22.386	22.308	22.277	22,3174
3	24.133	24.071	24.008	24.055	24.121	24,0776
4	24.071	24.102	24.008	24.492	24.071	24,1488
5	24.071	24.117	24.102	24.102	24.024	24,0832
6	25.896	26.270	25.990	25.880	25.833	25,9738

Tabela 6.24: *Performance* de *PrefK2** variando a complexidade da rede de preferências

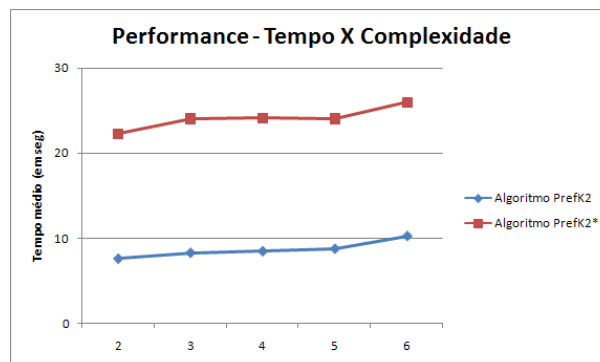


Figura 6.14: *Performance* X Complexidade

Considerações sobre os resultados obtidos com os dados sintéticos:

Observando os gráficos apresentados nas subseções anteriores, variando o número de amostras, o número de atributos e a complexidade da rede de preferências para as medidas de precisão e acurácia o leitor pode estar se perguntando por que não se pôde fazer distinção entre o algoritmo *PrefK2* e *PrefK2**, isto é, por que tais algoritmos possuem o

mesmo valor para todas as medidas com exceção da *performance* (tempo de execução). Tal fato se dá em decorrência das redes obtidas e da forma pela qual os dados sintéticos foram gerados.

A primeira constatação a se fazer é que nas redes encontradas os atributos não possuem falsos pais, isto é, se um atributo é independente ambos algoritmos detectam isso e não associam a ele nenhum pai. Isto se dá por que na fase de geração dos dados, nos casos em que são colocados valores iguais nos atributos $X1$ e $X'1$ (quando estes não são pais verdadeiros) é garantido que tal valor aparece uma única vez ali, não sendo portanto relevante para o cálculo dos melhores pais. Sendo assim a probabilidade de preferência dos atributos independentes é a mesma nas redes obtidas tanto no algoritmo *PrefK2* quanto no algoritmo *PrefK2**.

Com relação aos atributos dependentes, ambos algoritmos encontram pais que são válidos, isto é, pais que existem na rede original (rede de *input* dada para a geração dos dados sintéticos inicialmente). Para entender melhor o processo considere as duas redes das Figuras 6.15 e 6.16 inferidas pelos algoritmos *PrefK2* e *PrefK2** respectivamente a partir do conjunto de dados sintéticos da Tabela 6.25. Considere que a rede obtida pelo algoritmo *PrefK2** é a rede ideal, isto é, a rede encontrada pelo algoritmo *PrefK2** é exatamente a mesma rede original (a rede dada como *input* para geração dos dados sintéticos). Imagine que a Tabela 6.25 é apenas uma parte do conjunto de dados sintéticos (uma única passagem pela rede de preferências de *input*) e que na verdade possuímos 100 amostras de preferências. Isto quer dizer que em 100 amostras dos dados sintéticos temos as seguintes porcentagens, onde a palavra *new* indica um novo valor que só vai aparecer uma única vez nos dados sintéticos:

- 12% das *bituplas* são do tipo: $a1, b1, c_{new}, d_{new} | a2, b2, c_{new'}, d_{new'}, 1$
- 18% das *bituplas* são do tipo: $a1, b2, c_{new}, d_{new} | a2, b1, c_{new'}, d_{new'}, 1$
- 28% das *bituplas* são do tipo: $a2, b1, c_{new}, d_{new} | a1, b2, c_{new'}, d_{new'}, 1$
- 42% das *bituplas* são do tipo: $a2, b2, c_{new}, d_{new} | a1, b1, c_{new'}, d_{new'}, 1$
- 40% das *bituplas* são do tipo: $a1, b1, c1, d_{new} | a1, b1, c2, d_{new}, 1$
- 60% das *bituplas* são do tipo: $a1, b1, c2, d_{new} | a1, b1, c1, d_{new'}, 1$
- 80% das *bituplas* são do tipo: $a_{new}, b_{new}, c1, d1 | a_{new}, b_{new}, c1, d2, 1$
- 20% das *bituplas* são do tipo: $a_{new}, b_{new}, c1, d2 | a_{new}, b_{new}, c1, d1, 1$

Sendo assim é possível concluirmos que no caso do atributo C depender do atributo A (Rede de preferência gerada pelo algoritmo *PrefK2* - Figura 6.15) temos que $P[C = c1 > C = c2 | A = a1] = 0,40$, ou do atributo C depender dos atributos A e B (Rede de preferência gerada pelo algoritmo *PrefK2**) temos que $P[C = c1 > C = c2 | A = a1 \wedge B =$

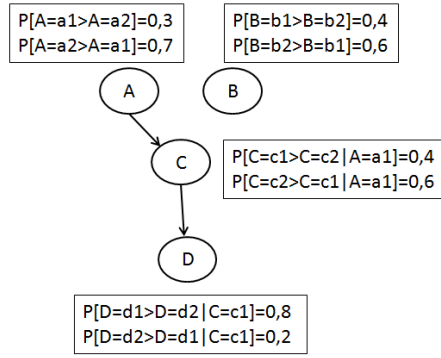


Figura 6.15: Rede obtida pelo algoritmo *PrefK2*

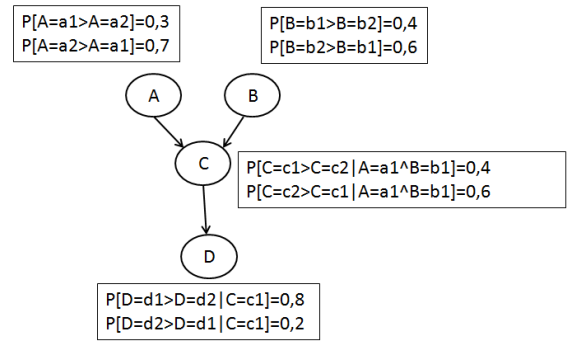


Figura 6.16: Rede obtida pelo algoritmo *PrefK2**

A	B	C	D	A'	B'	C'	D'	Pref
a2	b2	c100	d100	a1	b1	c100'	d'000	1
a1	b1	c2	d101	a1	b1	c1	d101'	1
a102	b102	c1	d1	a102	b102	c1	d2	1

Tabela 6.25: Amostras de preferências

$b1] = 0,40$, ou ainda do atributo C depender somente do atributo B a probabilidade obtida também seria $P[C = c1 > C = c2|B = b1] = 0,40$.

Com as informações anteriores concluímos que as tabelas de probabilidades tendem a ter o mesmo valor, se o algoritmo encontra no mínimo um pai do conjunto dos pais prováveis, e isto está acontecendo para os dois algoritmos. Como o cálculo da acurácia e a precisão são baseados nas tabelas de probabilidades da rede, e estas são iguais para ambos algoritmos, então justifica-se aí o fato da acurácia e precisão serem iguais para o algoritmo *PrefK2* e *PrefK2**.

6.2.4 Resultados com dados reais

As amostras dos dados reais variaram de 100 a 30.000. A seguir apresentamos os dados colhidos dos algoritmos *PrefK2* e *PrefK2** referentes à acurácia e precisão para os dados reais apresentados anteriormente.

Nº de amostras	suporte	1ª Rodada	2ª Rodada	3ª Rodada	4ª Rodada	5ª Rodada	Acurácia
100	10	9	16	13	14	18	0,7
300	10	26	39	53	45	52	0,716667
1.000	20	99	149	164	176	174	0,762
3.000	40	402	460	492	433	414	0,73366667
10.000	40	1075	1638	1414	1491	1418	0,7036
30.000	40	3609	4060	4303	4614	4752	0,71126667

Tabela 6.26: Cálculo da acurácia variando as amostras de preferências do banco de dados reais - Algoritmo *PrefK2*

Amostr as	1ª Rodada	2ª Rodada	3ª Rodada	4ª Rodada	5ª Rodada	Precisão
100	2 acertos em 5	6 acertos em 7	12 acertos em 13	11 acertos em 11	18 acertos em 18	0,907407407
300	21 acertos em 30	15 acertos em 17	13 acertos em 16	25 acertos em 28	15 acertos em 20	0,801801802
1.000	38 acertos em 59	39 acertos em 39	42 acertos em 48	30 acertos em 32	63 acertos em 64	0,876033058
3.000	155 acertos em 183	175 acertos em 194	189 acertos em 206	144 acertos em 168	191 acertos em 213	0,885892116
10.000	489 acertos em 632	548 acertos em 601	473 acertos em 512	663 acertos em 751	603 acertos em 689	0,871585557
30.000	1280 acertos em 1821	1672 acertos em 2024	1621 acertos em 1899	1579 acertos em 1806	1409 acertos em 1686	0,818644435

Tabela 6.27: Cálculo da precisão variando as amostras de preferências do banco de dados reais - Algoritmo *PrefK2*

Nº de amostras	suporte	1ª Rodada	2ª Rodada	3ª Rodada	4ª Rodada	5ª Rodada	Acurácia
100	10	9	11	13	14	18	0,65
300	10	30	39	53	45	53	0,733333333
1.000	20	99	149	164	176	174	0,762
3.000	40	464	490	506	477	481	0,806
10.000	40	1112	1587	1398	1558	1394	0,7049
30.000	40	4387	5025	4983	4983	5138	0,8172

Tabela 6.28: Cálculo da acurácia variando as amostras de preferências do banco de dados reais - Algoritmo *PrefK2**

Amostr as	1ª Rodada	2ª Rodada	3ª Rodada	4ª Rodada	5ª Rodada	Precisão
100	2 acertos em 5	6 acertos em 7	12 acertos em 13	11 acertos em 11	18 acertos em 18	0,907407407
300	21 acertos em 30	15 acertos em 17	13 acertos em 16	25 acertos em 28	15 acertos em 20	0,801801802
1.000	38 acertos em 59	39 acertos em 39	40 acertos em 46	26 acertos em 28	63 acertos em 64	0,872881356
3.000	247 acertos em 290	229 acertos em 262	242 acertos em 262	206 acertos em 229	273 acertos em 323	0,876281113
10.000	642 acertos em 799	627 acertos em 662	610 acertos em 662	840 acertos em 940	748 acertos em 872	0,881067344
30.000	2251 acertos em 3083	2855 acertos em 3427	2704 acertos em 3107	2585 acertos em 2968	2165 acertos em 2507	0,832228995

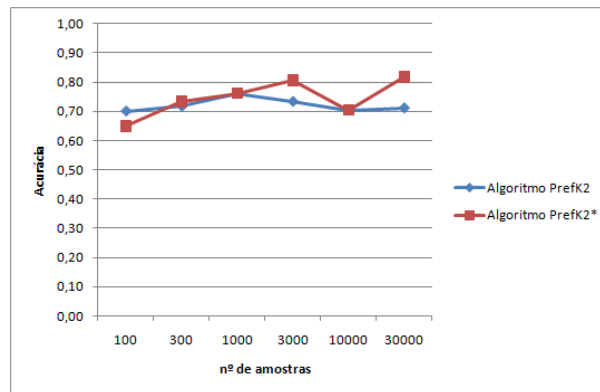
Tabela 6.29: Cálculo da precisão variando as amostras de preferências do banco de dados reais - Algoritmo *PrefK2**

Figura 6.17: Acurácia X variação do número de amostras

6.2.5 Algumas redes encontradas nos dados reais

Nesta seção apresentamos algumas topologias de redes de preferências encontradas nos dados reais. Para fins de curiosidade, o leitor pode estar vendo também algumas tabelas de probabilidade encontradas nos dados reais no Apêndice B.

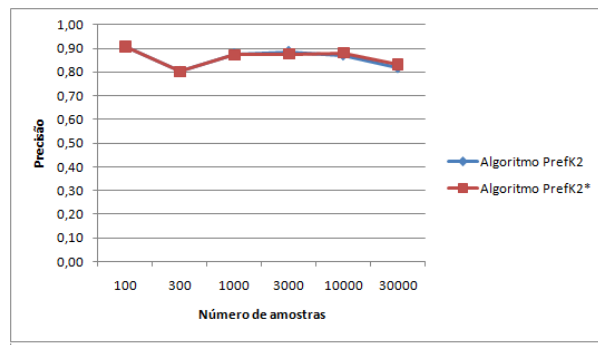
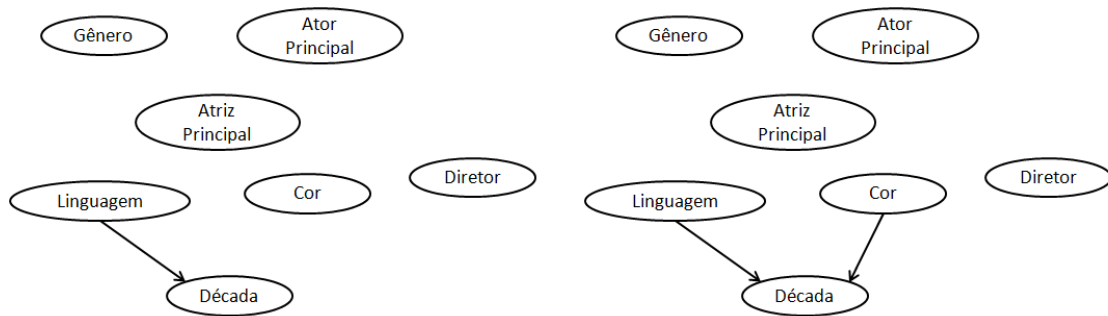


Figura 6.18: Precisão X variação do número de amostras

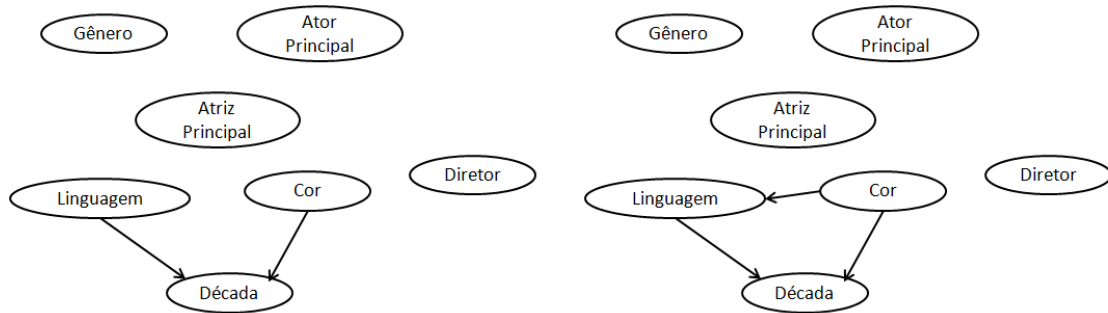


(a) Rede encontrada com o algoritmo *PrefK2* (b) Rede encontrada com o algoritmo *PrefK2**

Figura 6.19: Redes encontradas para as amostras do usuário cuja Id é 821

Usuário nº821 com *input* de 100 amostras de preferências

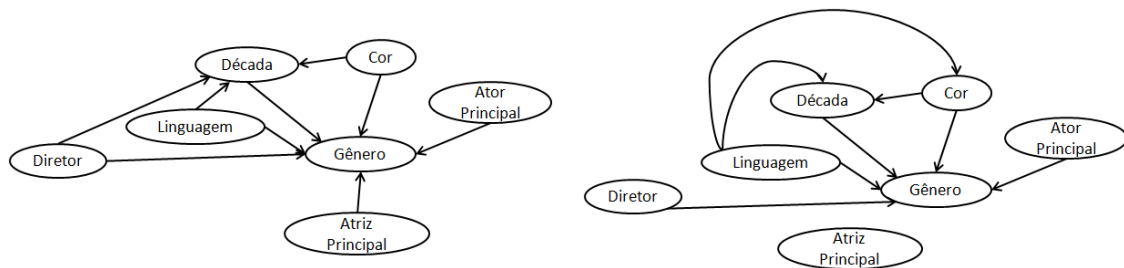
Usuário nº3781 com *input* de 1000 amostras de preferências



(a) Rede encontrada com o algoritmo *PrefK2* (b) Rede encontrada com o algoritmo *PrefK2**

Figura 6.20: Redes encontradas para as amostras do usuário cuja Id é 3781

Usuário nº620 com *input* de 10.000 amostras de preferências



(a) Rede encontrada com o algoritmo *PrefK2* (b) Rede encontrada com o algoritmo *PrefK2**

Figura 6.21: Redes encontradas para as amostras do usuário cuja Id é 620

6.3 Considerações finais

Neste Capítulo foi elucidado todo processo de geração dos dados sintéticos, seus pontos fortes e pontos fracos, além de uma descrição sucinta dos dados reais, origem, objetivos. Além disso, também descrevemos os resultados obtidos com nossa abordagem em dados sintéticos e dados reais. Como foi visto, tais resultados são ainda preliminares, mas já podem ser considerados promissores para a tarefa de mineração de preferências condicionais.

Capítulo 7

Conclusão e Trabalhos Futuros

A *mineração de preferências condicionais* constitui um importante e recente problema dentro de um contexto macro denominado *Recomendação de Conteúdo*, a qual está atrelada a uma maior área de pesquisa que é a de Banco de Dados. Neste trabalho propomos o conceito de *rede de preferências*, uma metodologia pela qual é possível inferir uma ordem para quaisquer pares de tuplas. A rede de preferências foi proposta para minerar preferências condicionais.

A mineração de preferências é um tema que demanda muita pesquisa, principalmente no que tange a trabalhos envolvendo mineração de preferências condicionais. Até onde sabemos não existem trabalhos que envolvam mineração de preferências condicionais.

Realizamos vários experimentos para avaliar a *performance* e a escalabilidade dos algoritmos *PrefK2* e *PrefK2**. Os resultados obtidos confirmam a teoria da promissoriedade do método proposto.

Nos resultados com os dados sintéticos nos deparamos com uma adversidade: não foi possível fazer distinção entre os resultados obtidos com os algoritmos *PrefK2* e *PrefK2**. Identificamos o problema na seção 6.2.3 e devido a não disponibilidade de tempo pré-defesa deste trabalho, propomos a resolução de tal problema em trabalhos futuros.

A abordagem que propomos é ainda uma abordagem inicial, mas que mostra ser bastante promissora pelos resultados alcançados, abrindo um vasto campo de pesquisas futuras, tais como:

- Desenvolver um método para construção de dados sintéticos mais eficiente que viabilize a distinção dos resultados tanto com o algoritmo *PrefK2* quanto com o algoritmo *PrefK2**.
- Implementar e testar o algoritmo de aprendizado de estruturas de redes de preferências *PrefDelta* apresentado no Anexo A. Tal algoritmo não necessita de uma ordem sobre os atributos como entrada (diferente dos algoritmos *PrefK2* e *PrefK2**).
- Desenvolver métodos para aprendizado de estruturas de redes de preferências baseados nos métodos para aprendizado de redes bayesianas segundo o enfoque de análise

de independência.

- Usar as regras de preferências obtidas pelo processo de mineração de preferências como *input* em linguagens de consultas como a *CPrefSQL* descrita em [de Amo e Ribeiro 2009] e [de Amo e Pereira 2010]. A linguagem *CPrefSQL* é uma extensão da SQL com suporte a preferências qualitativas condicionais. Em <http://www.lsi.ufu.br/cprefsql/in> é possível realizar testes com a linguagem *CPrefSQL*. No futuro, os métodos de mineração propostos neste trabalho serão incorporados nesta linguagem e estarão acessíveis neste site.

Em [Ferreira et al. 2010] foi incorporado um método relativamente simples de mineração de preferências do usuário no contexto de dados de imagens. As preferências são do tipo quantitativas. O método proposto nesta dissertação envolve preferências mais complexas e que incorporadas à linguagem *CPrefSQL* poderão produzir resultados mais satisfatórios ao usuário, já que a semântica das preferências é mais rica. O trabalho de [Ferreira et al. 2010] mostra a viabilidade de incorporar um método de mineração de preferências à uma linguagem de consultas que possui operadores de filtragem de respostas de acordo com as preferências mineradas.

- No presente trabalho realizamos testes nos dados reais levando em consideração apenas as preferências sobre os atributos referentes somente ao objeto pretendido, não levamos em consideração atributos referentes ao perfil do usuário, como por exemplo sexo, idade, renda, dentre outros. Pretendemos levar em consideração tais atributos na preferência do usuário.
- Considerar a propriedade de transitividade das amostras de preferências, fato que influenciará a geração da rede de preferências.
- Realizar testes de consistência nas amostras de preferências corrigindo possíveis erros que o usuário pode cometer na fase de demonstrar suas necessidades.
- Utilizar outras técnicas de mineração para obter preferências condicionais tais como algoritmos de descoberta de padrões como o clássico Apriori.

O tema abordado nessa dissertação deu origem a um artigo apresentado em [Félix e de Amo 2010]. Tal trabalho foi considerado um dos cinco melhores, sendo convidado para integrar uma edição especial do *Journal of Information and Data Management* (JIDM) em 2011.

Apêndice A

Outros métodos para aprendizado da rede de preferência

Apresentamos neste capítulo outros métodos de aprendizagem da estrutura e de treinamento de uma rede de preferências. Tais métodos foram propostos durante o desenvolvimento deste trabalho, mas devido a pouca disponibilidade de tempo não nos foi possível testá-los.

A.1 Aprendizado da estrutura de redes de preferências

Nesta seção é apresentado um método de aprendizado de estrutura de rede de preferência, onde não há a necessidade de fornecer como entrada uma ordem sobre os atributos.

A.1.1 Outro método de busca guloso

Pesquisadores tem considerado estratégias de buscas heurísticas para encontrar a melhor estrutura com o intuito de melhorar a *performance* dos métodos de busca gulosos nesses domínios. Uma das estratégias mais comuns é realizar pequenas mudanças iterativamente na estrutura corrente comparando o *score* da rede resultante com o *score* da rede anterior.

A definição de espaço de busca determina a definição de operadores de busca usados para a partir de uma estrutura se chegar a outra estrutura derivada da anterior. Tipicamente os operadores consistem de:

- Adição de uma aresta: insere-se uma aresta simples entre dois vértices não adjacentes.
- Deleção de uma aresta: remove-se uma aresta simples entre dois vértices.
- Reverção de uma aresta: inverte-se a direção de uma aresta simples.

Sendo assim representamos por $op(S, A)$ o resultado da operação A sobre a estrutura S .

Uma importante propriedade desses operadores é que eles resultam somente em mudanças locais nas estruturas correntes. Por exemplo, se um arco é inserido a partir de um vértice X_i para X_j , então somente a família do vértice X_j sofre mudanças, e similarmente se um arco é deletado. Se um arco é invertido, então a família de ambos X_i e X_j são alteradas. Esta propriedade pode ser explorada quando nós temos uma função *score decomponível*.

Uma função *score* é dita *decomponível* se ela pode ser expressada como a soma de todos os *scores locais*, um para cada família de vértices na estrutura:

$$score(D, S) = \sum_{i=1}^n score(X_i, pa(X_i), D).$$

A propriedade de *decomposição* pode ser usada quando nós avaliamos o benefício de realizar uma mudança em uma aresta. Por exemplo, se nós inserimos um arco de X_i para X_j , então somente o *score* local para X_j sofrerá as mudanças, isto é, quando for feita a avaliação se a mudança foi benéfica nós somente avaliaremos a diferença de *scores* (ou o ganho):

$$\Delta(X_i \rightarrow X_j) = score(X_j, pa(X_j) \cup \{X_i\}, D) - score(X_j, pa(X_j), D).$$

Algoritmo PrefDelta

O procedimento de busca é guloso e é caracterizado como a seguir: escolha alguma estrutura inicial (usualmente um estrutura vazia, uma estrutura escolhida randomicamente ou mesmo uma estrutura especificada pelo usuário) e calcule o ganho para cada operação válida realizada; por operação válida entende-se que a operação realizada resulte em um grafo acíclico. A operação realizada com maior ganho é a operação que prevalece na estrutura. Mais formalmente temos:

Entrada: Um conjunto de n de atributos discretos $A_i \in R(A_1, \dots, A_n)$, um grafo B_S vazio com os vértices em A_i, um banco de dados de k amostras de preferências em R. Saída: Um grafo B_S^* com os vértices em $A_1, \dots, A_n$ que melhor reflete a distribuição probabilística das k amostras de entrada.	
1.	Construa as partições P para cada $Comp(A_i)$
2.	Para cada partição $p_j \in P$
3.	Repita
4.	Calcular o ganho Δ para cada operação op válida sobre cada par de atributos $\in A_i$
5.	- Depois de calcular todos os ganhos, pega-se a operação op que produziu o maior ganho Δ^* .
6.	Se $\Delta^* > 0$, então //Se o ganho for positivo
7.	- Seja $B_S^* = op(B_S, A_i^*)$. /* Considera-se a estrutura B_S^* obtida a partir de B_S aplicando-se a operação op ganhadora e repete-se o processo para a estrutura $B_S^* * /$
8.	Até $\Delta^* \leq 0$. //O algoritmo pára quando o ganho máximo obtido for menor ou igual a zero.

Figura A.1: Algoritmo *PrefDelta*

A.2 Treinamento em redes de preferências

Nesta seção são apresentados dois métodos de treinamento de redes de preferências, estudados durante o decorrer deste trabalho. Tais métodos foram implementados, porém não testados.

A.2.1 Método 1: Treinamento de uma rede de preferências (utilizando o algoritmo ID3)

A partir de um conjunto de amostras de preferências P , uma topologia de rede de preferências *PrefNet*, e o domínio de todos os atributos, propomos um algoritmo para obter as tabelas de probabilidade.

No algoritmo proposto inicialmente realizamos um *Espelhamento* das tuplas que possuem preferências do tipo $[(v, u), 0]$ transformando-as em preferências do tipo $[(u, v), 1]$. Isto é feito para diminuir os nós folha quando utilizarmos do algoritmo ID3¹ [Quinlan 1990]) na construção da árvore de preferências. Um espelhamento é feito na Tabela A.1 tendo como resultado a Tabela A.2.

A tabela A.1 a seguir apresenta um conjunto de amostras de preferências determinado por um usuário a partir de um banco de dados $R(A, B, C, D)$.

Na Figura A.2.1 é apresentado o algoritmo para treinamento de uma rede de preferências.

¹No primeiro momento estamos supondo que os atributos são categóricos, entretanto estaremos generalizando as idéias expressas aqui para tratar preferências também em atributos contínuos

Id	A	B	C	D	Id	A'	B'	C'	D'	Pref
t_1	a_1	b_2	c_2	d_1	t_6	a_1	b_1	c_1	d_1	1
t_2	a_1	b_2	c_2	d_2	t_7	a_2	b_2	c_2	d_2	0
t_3	a_1	b_2	c_2	d_1	t_8	a_2	b_1	c_1	d_1	0
t_4	a_2	b_2	c_2	d_2	t_9	a_1	b_1	c_1	d_1	0
t_5	a_1	b_1	c_1	d_1	t_{10}	a_2	b_2	c_1	d_2	1

Tabela A.1: Amostras de preferências

Id	A	B	C	D	Id	A'	B'	C'	D'	Pref
t_1	a_1	b_2	c_2	d_1	t_6	a_1	b_1	c_1	d_1	1
t_7	a_2	b_2	c_2	d_2	t_2	a_1	b_2	c_2	d_2	1
t_8	a_2	b_1	c_1	d_1	t_3	a_1	b_2	c_2	d_1	1
t_9	a_1	b_1	c_1	d_1	t_4	a_2	b_2	c_2	d_2	1
t_5	a_1	b_1	c_1	d_1	t_{10}	a_2	b_2	c_1	d_2	1

Tabela A.2: Espelhamento da Tabela A.1

Entrada: Um conjunto de amostras de preferências P, uma topologia de rede de preferências $PrefNet$, e o domínio de todos os atributos $dom(X_i)$. Saída: Tabela de Probabilidade para cada atributo da rede de preferências
1. Faça o Espelhamento de P 2. Seguindo a ordem topológica do grafo de dependências escolha o atributo X_i 3. Enquanto estrutura de rede preferências não for vazia faça 4. Se X_i é atributo dominante faça 5. Agrupe os pares de valores (X_i, X'_i) 6. Selecione os pares de valores, onde $X_i \neq X'_i$ 7. Calcule a $freq(X_i = a_1 < X'_i = a_2)$ 8. Extraia as $P[X_i = a_1 < X'_i = a_2]$ e $P[X_i = a_2 < X'_i = a_1]$ 9. Retire da rede de preferências o nó já calculado 10. Se X_i é um atributo condicional faça 11. Considere $Pais(X_i) = X_1, \dots, X_n$ 12. Para cada par de valores (X_i, X'_i) agrupe as tuplas 13. Selecione os pares de valores, onde $X_i \neq X'_i$ 14. Aplique o algoritmo ID3 de árvore de decisão². 15. Teremos ao final nas folhas da árvore de decisão um conjunto de regras de preferências 16. Calcule a frequência $freq(X_i = a_1 < X'_i = a_2 Pais(X_i))$ 17. Extraia as probabilidades correspondentes 18. Retire o nó já calculado.

Figura A.2: Algoritmo de treinamento de uma rede de preferências

A.2.2 Método 2: Treinamento de uma rede de preferências (utilizando o cálculo da Esperança)

Dada a topologia de uma rede de preferências e um banco de amostras de preferências, nesta seção apresentamos outro método de realizar o treinamento de uma rede de preferências, isto é, derivar as tabelas de probabilidade a partir do cálculo da *esperança*

das probabilidades, conforme Definição A.1.

Definição A.1. (Esperança) Seja θ_{ijk} denotando a probabilidade $P[X_i = v_{ik} > X_i = v_{ik'} | \pi_i = w_{ij}]$ — o que é a probabilidade de $X_i = v_{ik}$ ser preferido a $X_i = v_{ik'}$ para algum k variando de 1 a r_i , dado que os pais de X_i , representados por π_i , são instanciados como w_{ij} . Chamamos θ_{ijk} a probabilidade condicional do atributo. Chamamos o valor de $\mathcal{E}[\theta_{ijk} | \mathcal{D}, B_S]$ a esperança de θ_{ijk} dado o banco de amostras $\mathcal{D}$ e a rede de preferências B_S . As definições e formas de cálculo de N_{ij0} , N_{ij1} , N_{ij} foram dadas na Definição 5.1.

$$\mathcal{E}[\theta_{ijk} | \mathcal{D}, B_S] = \frac{N_{ij0} + N_{ij1} + 1}{N_{ij} + r_i} \quad (\text{A.1})$$

Apêndice B

Exemplo de Tabelas de Probabilidades Condicionais encontradas nos dados reais

B.1 Usuário nº821 com *input* de 100 amostras de preferências

Algumas das 354 tabelas de probabilidades encontradas segundo a topologia de preferências da Figura 6.19(b) obtida pelo algoritmo *prefK2**:

- $P[\text{Genero} = \text{GM183} > \text{Genero} = \text{GM121}] = 1.0;$
 $P[\text{Genero} = \text{GM121} > \text{Genero} = \text{GM183}] = 0.0;$
- $P[\text{Genero} = \text{GM057} > \text{Genero} = \text{GM121}] = 1.0;$
 $P[\text{Genero} = \text{GM121} > \text{Genero} = \text{GM057}] = 0.0;$
- $P[\text{Genero} = \text{GM479} > \text{Genero} = \text{GM121}] = 1.0;$
 $P[\text{Genero} = \text{GM121} > \text{Genero} = \text{GM479}] = 0.0;$
- $P[\text{Genero} = \text{GM124} > \text{Genero} = \text{GM076}] = 1.0;$
 $P[\text{Genero} = \text{GM076} > \text{Genero} = \text{GM124}] = 0.0;$
- $P[\text{Genero} = \text{GM073} > \text{Genero} = \text{GM057}] = 1.0;$
 $P[\text{Genero} = \text{GM057} > \text{Genero} = \text{GM073}] = 0.0;$
- $P[\text{Genero} = \text{GM124} > \text{Genero} = \text{GM121}] = 1.0;$
 $P[\text{Genero} = \text{GM121} > \text{Genero} = \text{GM124}] = 0.0;$
- $P[\text{Genero} = \text{GM124} > \text{Genero} = \text{Comedy}] = 1.0;$
 $P[\text{Genero} = \text{Comedy} > \text{Genero} = \text{GM124}] = 0.0;$

- $P[\text{Ator} = \text{Harrison}; \text{Rex}(I) > \text{Ator} = \text{Neeson}; \text{Liam}] = 0.5;$
 $P[\text{Ator} = \text{Neeson}; \text{Liam} > \text{Ator} = \text{Harrison}; \text{Rex}(I)] = 0.5;$
- $P[\text{Ator} = \text{Washington}; \text{Denzel} > \text{Ator} = \text{DeNiro}; \text{Robert}] = 1.0;$
 $P[\text{Ator} = \text{DeNiro}; \text{Robert} > \text{Ator} = \text{Washington}; \text{Denzel}] = 0.0;$
- $P[\text{Ator} = \text{Morse}; \text{David}(I) > \text{Ator} = \text{Wahlberg}; \text{Mark}(I)] = 1.0;$
 $P[\text{Ator} = \text{Wahlberg}; \text{Mark}(I) > \text{Ator} = \text{Morse}; \text{David}(I)] = 0.0;$
- $P[\text{Ator} = \text{Crudup}; \text{Billy} > \text{Ator} = \text{Freeman}; \text{Morgan}(I)] = 1.0;$
 $P[\text{Ator} = \text{Freeman}; \text{Morgan}(I) > \text{Ator} = \text{Crudup}; \text{Billy}] = 0.0;$
- $P[\text{Ator} = \text{McQueen}; \text{Steve}(I) > \text{Ator} = \text{Wahlberg}; \text{Mark}(I)] = 1.0;$
 $P[\text{Ator} = \text{Wahlberg}; \text{Mark}(I) > \text{Ator} = \text{McQueen}; \text{Steve}(I)] = 0.0;$
- $P[\text{Atriz} = \text{Burstyn}; \text{Ellen} > \text{Atriz} = \text{Polo}; \text{Teri}] = 1.0;$
 $P[\text{Atriz} = \text{Polo}; \text{Teri} > \text{Atriz} = \text{Burstyn}; \text{Ellen}] = 0.0;$
- $P[\text{Atriz} = \text{Panettiere}; \text{Hayden} > \text{Atriz} = \text{Polo}; \text{Teri}] = 1.0;$
 $P[\text{Atriz} = \text{Polo}; \text{Teri} > \text{Atriz} = \text{Panettiere}; \text{Hayden}] = 0.0;$
- $P[\text{Atriz} = \text{Goodall}; \text{Caroline} > \text{Atriz} = \text{Lewis}; \text{Juliette}(I)] = 1.0;$
 $P[\text{Atriz} = \text{Lewis}; \text{Juliette}(I) > \text{Atriz} = \text{Goodall}; \text{Caroline}] = 0.0;$
- $P[\text{Atriz} = \text{Panettiere}; \text{Hayden} > \text{Atriz} = \text{Theron}; \text{Charlize}] = 1.0;$
 $P[\text{Atriz} = \text{Theron}; \text{Charlize} > \text{Atriz} = \text{Panettiere}; \text{Hayden}] = 0.0;$
- $P[\text{Ator} = \text{Tirelli}; \text{Jaime} > \text{Ator} = \text{Braugher}; \text{Andre}] = 1.0;$
 $P[\text{Ator} = \text{Braugher}; \text{Andre} > \text{Ator} = \text{Tirelli}; \text{Jaime}] = 0.0;$
- $P[\text{Diretor} = \text{Crowe}; \text{Cameron} > \text{Diretor} = \text{Kusama}; \text{Karyn}] = 1.0;$
 $P[\text{Diretor} = \text{Kusama}; \text{Karyn} > \text{Diretor} = \text{Crowe}; \text{Cameron}] = 0.0;$
- $P[\text{Diretor} = \text{Yakin}; \text{Boaz} > \text{Diretor} = \text{Gray}; \text{James}(I)] = 1.0;$
 $P[\text{Diretor} = \text{Gray}; \text{James}(I) > \text{Diretor} = \text{Yakin}; \text{Boaz}] = 0.0;$
- $P[\text{Diretor} = \text{Spielberg}; \text{Steven}(I) > \text{Diretor} = \text{Gray}; \text{James}(I)] = 1.0;$
 $P[\text{Diretor} = \text{Gray}; \text{James}(I) > \text{Diretor} = \text{Spielberg}; \text{Steven}(I)] = 0.0;$
- $P[\text{Diretor} = \text{Spielberg}; \text{Steven}(I) > \text{Diretor} = \text{Lurie}; \text{Rod}] = 1.0;$
 $P[\text{Diretor} = \text{Lurie}; \text{Rod} > \text{Diretor} = \text{Spielberg}; \text{Steven}(I)] = 0.0;$
- $P[\text{Diretor} = \text{Lurie}; \text{Rod} > \text{Diretor} = \text{Gray}; \text{James}(I)] = 1.0;$
 $P[\text{Diretor} = \text{Gray}; \text{James}(I) > \text{Diretor} = \text{Lurie}; \text{Rod}] = 0.0;$

- $P[\text{Diretor} = \text{Crowe}; \text{Cameron} > \text{Diretor} = \text{Lucas}; \text{George}] = 1.0;$
 $P[\text{Diretor} = \text{Lucas}; \text{George} > \text{Diretor} = \text{Crowe}; \text{Cameron}] = 0.0;$
- $P[\text{Decada} = 1970 > \text{Decada} = 2000 | \text{Linguagem} = \text{English}] = 0.5;$
 $P[\text{Decada} = 2000 > \text{Decada} = 1970 | \text{Linguagem} = \text{English}] = 0.5;$
- $P[\text{Decada} = 2000 > \text{Decada} = 1990 | \text{Cor} = \text{Color}] = 0.7;$
 $P[\text{Decada} = 1990 > \text{Decada} = 2000 | \text{Cor} = \text{Color}] = 0.3;$
- $P[\text{Decada} = 1960 > \text{Decada} = 1990 | \text{Cor} = \text{Color}] = 1.0;$
 $P[\text{Decada} = 1990 > \text{Decada} = 1960 | \text{Cor} = \text{Color}] = 0.0;$
- $P[\text{Decada} = 1960 > \text{Decada} = 1990 | \text{Linguagem} = \text{English} \wedge \text{Cor} = \text{Color}] = 1.0;$
 $P[\text{Decada} = 1990 > \text{Decada} = 1960 | \text{Linguagem} = \text{English} \wedge \text{Cor} = \text{Color}] = 0.0;$
- $P[\text{Linguagem} = \text{LM051} > \text{Linguagem} = \text{English}] = 0.7142857142857143;$
 $P[\text{Linguagem} = \text{English} > \text{Linguagem} = \text{LM051}] = 0.2857142857142857;$
- $P[\text{Linguagem} = \text{LM051} > \text{Linguagem} = \text{LM003}] = 1.0;$
 $P[\text{Linguagem} = \text{LM003} > \text{Linguagem} = \text{LM051}] = 0.0;$
- $P[\text{Linguagem} = \text{LM051} > \text{Linguagem} = \text{LM168}] = 1.0;$
 $P[\text{Linguagem} = \text{LM168} > \text{Linguagem} = \text{LM051}] = 0.0;$
- $P[\text{Linguagem} = \text{LM003} > \text{Linguagem} = \text{LM168}] = 1.0;$
 $P[\text{Linguagem} = \text{LM168} > \text{Linguagem} = \text{LM003}] = 0.0;$
- $P[\text{Linguagem} = \text{LM168} > \text{Linguagem} = \text{English}] = 0.11111111111111111;$
 $P[\text{Linguagem} = \text{English} > \text{Linguagem} = \text{LM168}] = 0.88888888888888888;$
- $P[\text{Linguagem} = \text{English} > \text{Linguagem} = \text{LM003}] = 0.93333333333333333;$
 $P[\text{Linguagem} = \text{LM003} > \text{Linguagem} = \text{English}] = 0.066666666666666667;$
- $P[\text{Cor} = \text{Color} > \text{Cor} = \text{Dois}] = 0.166666666666666666;$
 $P[\text{Cor} = \text{Dois} > \text{Cor} = \text{Color}] = 0.83333333333333334;$

B.2 Usuário nº3781 com *input* de 1000 amostras de preferências

Algumas das 2.625 tabelas de probabilidades encontradas segundo a topologia de preferências da Figura 6.20(a) obtida pelo algoritmo *prefK2*:

- $P[\text{Genero} = \text{GM100} > \text{Genero} = \text{GM025}] = 1.0;$
 $P[\text{Genero} = \text{GM025} > \text{Genero} = \text{GM100}] = 0.0$

- $P[\text{Genero} = \text{Drama} > \text{Genero} = \text{GM013}] = 0.78;$
 $P[\text{Genero} = \text{GM013} > \text{Genero} = \text{Drama}] = 0.22;$
- $P[\text{Genero} = \text{Drama} > \text{Genero} = \text{GM029}] = 0.87;$
 $P[\text{Genero} = \text{GM029} > \text{Genero} = \text{Drama}] = 0.13;$
- $P[\text{Genero} = \text{GM025} > \text{Genero} = \text{GM013}] = 0.67;$
 $P[\text{Genero} = \text{GM013} > \text{Genero} = \text{GM025}] = 0.33;$
- $P[\text{Genero} = \text{GM029} > \text{Genero} = \text{GM013}] = 0.33;$
 $P[\text{Genero} = \text{GM013} > \text{Genero} = \text{GM029}] = 0.67;$
- $P[\text{Genero} = \text{GM100} > \text{Genero} = \text{GM018}] = 1.0;$
 $P[\text{Genero} = \text{GM018} > \text{Genero} = \text{GM100}] = 0.0;$
- $P[\text{Ator} = \text{Cosmo}; \text{James} > \text{Ator} = \text{Moore}; \text{Roger}(I)] = 1.0;$
 $P[\text{Ator} = \text{Moore}; \text{Roger}(I) > \text{Ator} = \text{Cosmo}; \text{James}] = 0.0;$
- $P[\text{Ator} = \text{Keitel}; \text{Harvey} > \text{Ator} = \text{Cruise}; \text{Tom}] = 0.2;$
 $P[\text{Ator} = \text{Cruise}; \text{Tom} > \text{Ator} = \text{Keitel}; \text{Harvey}] = 0.8;$
- $P[\text{Ator} = \text{Cruise}; \text{Tom} > \text{Ator} = \text{Damon}; \text{Matt}] = 0.66;$
 $P[\text{Ator} = \text{Damon}; \text{Matt} > \text{Ator} = \text{Cruise}; \text{Tom}] = 0.33;$
- $P[\text{Ator} = \text{Auteuil}; \text{Daniel} > \text{Ator} = \text{Moore}; \text{Roger}(I)] = 1.0;$
 $P[\text{Ator} = \text{Moore}; \text{Roger}(I) > \text{Ator} = \text{Auteuil}; \text{Daniel}] = 0.0;$
- $P[\text{Ator} = \text{DeNiro}; \text{Robert} > \text{Ator} = \text{Gibson}; \text{Mel}(I)] = 1.0;$
 $P[\text{Ator} = \text{Gibson}; \text{Mel}(I) > \text{Ator} = \text{DeNiro}; \text{Robert}] = 0.0;$
- $P[\text{Ator} = \text{Williams}; \text{Robin}(I) > \text{Ator} = \text{Moore}; \text{Roger}(I)] = 1.0;$
 $P[\text{Ator} = \text{Moore}; \text{Roger}(I) > \text{Ator} = \text{Williams}; \text{Robin}(I)] = 0.0;$
- $P[\text{Atriz} = \text{Ormond}; \text{Julia} > \text{Atriz} = \text{Elliott}; \text{Alison}(I)] = 1.0;$
 $P[\text{Atriz} = \text{Elliott}; \text{Alison}(I) > \text{Atriz} = \text{Ormond}; \text{Julia}] = 0.0;$
- $P[\text{Atriz} = \text{Hunter}; \text{Holly} > \text{Atriz} = \text{Lewis}; \text{Juliette}(I)] = 1.0;$
 $P[\text{Atriz} = \text{Lewis}; \text{Juliette}(I) > \text{Atriz} = \text{Hunter}; \text{Holly}] = 0.0;$
- $P[\text{Atriz} = \text{Maxwell}; \text{Roberta} > \text{Atriz} = \text{Lane}; \text{Diane}(I)] = 1.0;$
 $P[\text{Atriz} = \text{Lane}; \text{Diane}(I) > \text{Atriz} = \text{Maxwell}; \text{Roberta}] = 0.0;$
- $P[\text{Atriz} = \text{Moore}; \text{Demi} > \text{Atriz} = \text{Keen}; \text{Pat}] = 1.0;$
 $P[\text{Atriz} = \text{Keen}; \text{Pat} > \text{Atriz} = \text{Moore}; \text{Demi}] = 0.0;$

- $P[Atriz = McDonnell; Mary(I) > Atriz = Paltrow; Gwyneth] = 1.0;$
 $P[Atriz = Paltrow; Gwyneth > Atriz = McDonnell; Mary(I)] = 0.0;$
- $P[Diretor = Coppola; FrancisFord > Diretor = Hicks; Scott] = 1.0;$
 $P[Diretor = Hicks; Scott > Diretor = Coppola; FrancisFord] = 0.0;$
- $P[Diretor = Petersen; Wolfgang(I) > Diretor = Scott; Ridley] = 1.0;$
 $P[Diretor = Scott; Ridley > Diretor = Petersen; Wolfgang(I)] = 0.0;$
- $P[Diretor = Demme; Jonathan > Diretor = Petersen; Wolfgang(I)] = 1.0;$
 $P[Diretor = Petersen; Wolfgang(I) > Diretor = Demme; Jonathan] = 0.0;$
- $P[Diretor = Cameron; James(I) > Diretor = Robinson; PhilAlden] = 1.0;$
 $P[Diretor = Robinson; PhilAlden > Diretor = Cameron; James(I)] = 0.0;$
- $P[Diretor = Hickenlooper; George > Diretor = McGrath; Douglas] = 1.0;$
 $P[Diretor = McGrath; Douglas > Diretor = Hickenlooper; George] = 0.0;$
- $P[Decada = 1990 > Decada = 2000 | Linguagem = English \wedge Cor = Color] = 0.88;$
 $P[Decada = 2000 > Decada = 1990 | Linguagem = English \wedge Cor = Color] = 0.12;$
- $P[Decada = 1990 > Decada = 1970 | Cor = Color] = 1.0;$
 $P[Decada = 1970 > Decada = 1990 | Cor = Color] = 0.0;$
- $P[Decada = 2000 > Decada = 1990 | Cor = Color] = 0.083;$
 $P[Decada = 1990 > Decada = 2000 | Cor = Color] = 0.917;$
- $P[Decada = 1990 > Decada = 1980 | Linguagem = English \wedge Cor = Color] = 0.79;$
 $P[Decada = 1980 > Decada = 1990 | Linguagem = English \wedge Cor = Color] = 0.21;$
- $P[Decada = 1980 > Decada = 1990 | Cor = Color] = 0.57;$
 $P[Decada = 1990 > Decada = 1980 | Cor = Color] = 0.43;$
- $P[Decada = 1980 > Decada = 1970 | Cor = Color] = 1.0;$
 $P[Decada = 1970 > Decada = 1980 | Cor = Color] = 0.0;$
- $P[Decada = 1990 > Decada = 2000 | Linguagem = English] = 0.9;$
 $P[Decada = 2000 > Decada = 1990 | Linguagem = English] = 0.2;$
- $P[Decada = 2000 > Decada = 1970 | Cor = Color] = 1.0;$
 $P[Decada = 1970 > Decada = 2000 | Cor = Color] = 0.0;$
- $P[Decada = 1980 > Decada = 2000 | Cor = Color] = 1.0;$
 $P[Decada = 2000 > Decada = 1980 | Cor = Color] = 0.0;$

- $P[Decada = 1980 > Decada = 1990 | Linguagem = English] = 0.20;$
 $P[Decada = 1990 > Decada = 1980 | Linguagem = English] = 0.80;$
- $P[Linguagem = English > Linguagem = LM012] = 0.90;$
 $P[Linguagem = LM012 > Linguagem = English] = 0.20;$
- $P[Cor = Color > Cor = BlackandWhite] = 1.0;$
 $P[Cor = BlackandWhite > Cor = Color] = 0.0;$

B.3 Usuário nº620 com *input* de 10.000 amostras de preferências

Algumas das 28.817 tabelas de probabilidades encontradas segundo a topologia de preferências da Figura 6.21(b) obtida pelo algoritmo *prefK2**:

- $P[Genero = GM138 > Genero = GM035 | Linguagem = English \wedge Cor = Color] = 0.5;$
 $P[Genero = GM035 > Genero = GM138 | Linguagem = English \wedge Cor = Color] = 0.5;$
- $P[Genero = Comedy > Genero = GM024 | Cor = Color \wedge Decada = 1990 \wedge$
 $Diretor = Columbus; Chris] = 1.0;$
 $P[Genero = GM024 > Genero = Comedy | Cor = Color \wedge Decada = 1990 \wedge$
 $Diretor = Columbus; Chris] = 0.0;$
- $P[Genero = GM161 > Genero = GM002 | Linguagem = English \wedge Cor = Color \wedge$
 $Diretor = Shadyac; Tom] = 1.0;$
 $P[Genero = GM002 > Genero = GM161 | Linguagem = English \wedge Cor = Color \wedge$
 $Diretor = Shadyac; Tom] = 0.0;$
- $P[Genero = GM025 > Genero = GM158 | Linguagem = English] = 0.3076923076923077;$
 $P[Genero = GM158 > Genero = GM025 | Linguagem = English] = 0.6923076923076923;$
- $P[Genero = GM370 > Genero = GM275 | Decada = 1990 \wedge Ator = Myers; Mike(I)] = 1.0;$
 $P[Genero = GM275 > Genero = GM370 | Decada = 1990 \wedge Ator = Myers; Mike(I)] = 0.0;$
- $P[Genero = GM486 > Genero = GM289 | Decada = 1990] = 1.0;$
 $P[Genero = GM289 > Genero = GM486 | Decada = 1990] = 0.0;$

- $P[\text{Genero} = \text{GM146} > \text{Genero} = \text{GM013} | \text{Linguagem} = \text{English} \wedge \text{Decada} = 1990] = 0.3333333333333333$;
 $P[\text{Genero} = \text{GM013} > \text{Genero} = \text{GM146} | \text{Linguagem} = \text{English} \wedge \text{Decada} = 1990] = 0.6666666666666666$;
- $P[\text{Genero} = \text{GM065} > \text{Genero} = \text{GM219} | \text{Linguagem} = \text{English} \wedge \text{Cor} = \text{Color} \wedge \text{Diretor} = \text{Donner; Richard} \wedge \text{Ator} = \text{Gibson; Mel}(I)] = 1.0$;
 $P[\text{Genero} = \text{GM219} > \text{Genero} = \text{GM065} | \text{Linguagem} = \text{English} \wedge \text{Cor} = \text{Color} \wedge \text{Diretor} = \text{Donner; Richard} \wedge \text{Ator} = \text{Gibson; Mel}(I)] = 0.0$;
- $P[\text{Genero} = \text{GM220} > \text{Genero} = \text{GM430} | \text{Cor} = \text{Color} \wedge \text{Decada} = 1990 \wedge \text{Ator} = \text{Schwarzenegger; Arnold}] = 1.0$;
 $P[\text{Genero} = \text{GM430} > \text{Genero} = \text{GM220} | \text{Cor} = \text{Color} \wedge \text{Decada} = 1990 \wedge \text{Ator} = \text{Schwarzenegger; Arnold}] = 0.0$;
- $P[\text{Ator} = \text{Carrey; Jim} > \text{Ator} = \text{Clooney; George}] = 0.1666666666666666$;
 $P[\text{Ator} = \text{Clooney; George} > \text{Ator} = \text{Carrey; Jim}] = 0.8333333333333334$;
- $P[\text{Ator} = \text{Stallone; Sylvester} > \text{Ator} = \text{Nielsen; Leslie}] = 0.3333333333333333$;
 $P[\text{Ator} = \text{Nielsen; Leslie} > \text{Ator} = \text{Stallone; Sylvester}] = 0.6666666666666666$;
- $P[\text{Ator} = \text{Nielsen; Leslie} > \text{Ator} = \text{Williams; Robin}(I)] = 0.6666666666666666$;
 $P[\text{Ator} = \text{Williams; Robin}(I) > \text{Ator} = \text{Nielsen; Leslie}] = 0.3333333333333333$;
- $P[\text{Atriz} = \text{Goodall; Caroline} > \text{Atriz} = \text{Hurley; Elizabeth}(I)] = 1.0$
 $P[\text{Atriz} = \text{Hurley; Elizabeth}(I) > \text{Atriz} = \text{Goodall; Caroline}] = 0.0$
- $P[\text{Atriz} = \text{Bening; Annette} > \text{Atriz} = \text{Potente; Franka}] = 1.0$;
 $P[\text{Atriz} = \text{Potente; Franka} > \text{Atriz} = \text{Bening; Annette}] = 0.0$;
- $P[\text{Atriz} = \text{Hannigan; Alyson} > \text{Atriz} = \text{Mol; Gretchen}] = 1.0$
 $P[\text{Atriz} = \text{Mol; Gretchen} > \text{Atriz} = \text{Hannigan; Alyson}] = 0.0$
- $P[\text{Atriz} = \text{Kensit; Patsy} > \text{Atriz} = \text{Russo; Rene}] = 1.0$
 $P[\text{Atriz} = \text{Russo; Rene} > \text{Atriz} = \text{Kensit; Patsy}] = 0.0$
- $P[\text{Diretor} = \text{Tykwer; Tom} > \text{Diretor} = \text{Donner; Richard}] = 1.0$;
 $P[\text{Diretor} = \text{Donner; Richard} > \text{Diretor} = \text{Tykwer; Tom}] = 0.0$;
- $P[\text{Diretor} = \text{Soderbergh; Steven} > \text{Diretor} = \text{Herek; Stephen}] = 1.0$;
 $P[\text{Diretor} = \text{Herek; Stephen} > \text{Diretor} = \text{Soderbergh; Steven}] = 0.0$;
- $P[\text{Diretor} = \text{Sonnenfeld; Barry} > \text{Diretor} = \text{Brooks; JamesL.}] = 1.0$;
 $P[\text{Diretor} = \text{Brooks; JamesL.} > \text{Diretor} = \text{Sonnenfeld; Barry}] = 0.0$;

- $P[\text{Diretor} = \text{Mendes}; \text{Sam} > \text{Diretor} = \text{Marshall}; \text{Penny}(I)] = 1.0;$
 $P[\text{Diretor} = \text{Marshall}; \text{Penny}(I) > \text{Diretor} = \text{Mendes}; \text{Sam}] = 0.0;$
- $P[\text{Diretor} = \text{Hanson}; \text{Curtis}(I) > \text{Diretor} = \text{Levinson}; \text{Barry}(I)] = 1.0;$
 $P[\text{Diretor} = \text{Levinson}; \text{Barry}(I) > \text{Diretor} = \text{Hanson}; \text{Curtis}(I)] = 0.0;$
- $P[\text{Diretor} = \text{Okawara}; \text{Takao} > \text{Diretor} = \text{DePalma}; \text{Brian}] = 1.0;$
 $P[\text{Diretor} = \text{DePalma}; \text{Brian} > \text{Diretor} = \text{Okawara}; \text{Takao}] = 0.0;$
- $P[\text{Diretor} = \text{Hitchcock}; \text{Alfred}(I) > \text{Diretor} = \text{Herek}; \text{Stephen}] = 1.0;$
 $P[\text{Diretor} = \text{Herek}; \text{Stephen} > \text{Diretor} = \text{Hitchcock}; \text{Alfred}(I)] = 0.0;$
- $P[\text{Decada} = 1970 > \text{Decada} = 2000 | \text{Linguagem} = \text{English} \wedge \text{Cor} = \text{Color}] = 1.0;$
 $P[\text{Decada} = 2000 > \text{Decada} = 1970 | \text{Linguagem} = \text{English} \wedge \text{Cor} = \text{Color}] = 0.0;$
- $P[\text{Decada} = 1970 > \text{Decada} = 1960 | \text{Linguagem} = \text{English} \wedge \text{Cor} = \text{Color}] = 1.0;$
 $P[\text{Decada} = 1960 > \text{Decada} = 1970 | \text{Linguagem} = \text{English} \wedge \text{Cor} = \text{Color}] = 0.0;$
- $P[\text{Decada} = 1960 > \text{Decada} = 1950 | \text{Linguagem} = \text{English} \wedge \text{Cor} = \text{BlackandWhite}] = 1.0;$
 $P[\text{Decada} = 1950 > \text{Decada} = 1960 | \text{Linguagem} = \text{English} \wedge \text{Cor} = \text{BlackandWhite}] = 0.0;$
- $P[\text{Linguagem} = \text{LM018} > \text{Linguagem} = \text{LM072}] = 1.0;$
 $P[\text{Linguagem} = \text{LM072} > \text{Linguagem} = \text{LM018}] = 0.0;$
- $P[\text{Linguagem} = \text{English} > \text{Linguagem} = \text{LM196}] = 0.12162162162162163;$
 $P[\text{Linguagem} = \text{LM196} > \text{Linguagem} = \text{English}] = 0.8783783783783784;$
- $P[\text{Linguagem} = \text{LM022} > \text{Linguagem} = \text{English}] = 0.9502762430939227;$
 $P[\text{Linguagem} = \text{English} > \text{Linguagem} = \text{LM022}] = 0.049723756906077346;$
- $P[\text{Linguagem} = \text{LM021} > \text{Linguagem} = \text{English}] = 0.8783783783783784;$
 $P[\text{Linguagem} = \text{English} > \text{Linguagem} = \text{LM021}] = 0.12162162162162163;$
- $P[\text{Cor} = \text{BlackandWhite} > \text{Cor} = \text{Color} | \text{Linguagem} = \text{LM003}] = 1.0;$
 $P[\text{Cor} = \text{Color} > \text{Cor} = \text{BlackandWhite} | \text{Linguagem} = \text{LM003}] = 0.0;$
- $P[\text{Cor} = \text{BlackandWhite} > \text{Cor} = \text{Dois} | \text{Linguagem} = \text{English}] = 1.0;$
 $P[\text{Cor} = \text{Dois} > \text{Cor} = \text{BlackandWhite} | \text{Linguagem} = \text{English}] = 0.0;$
- $P[\text{Cor} = \text{Dois} > \text{Cor} = \text{Color} | \text{Linguagem} = \text{English}] = 0.3901639344262295;$
 $P[\text{Cor} = \text{Color} > \text{Cor} = \text{Dois} | \text{Linguagem} = \text{English}] = 0.6098360655737705;$
- $P[\text{Cor} = \text{BlackandWhite} > \text{Cor} = \text{Color} | \text{Linguagem} = \text{English}] = 0.972972972972973;$
 $P[\text{Cor} = \text{Color} > \text{Cor} = \text{BlackandWhite} | \text{Linguagem} = \text{English}] = 0.02702702702702703;$

As siglas LM, DM e GM correspondem a Linguagem Múltipla, Diretor Múltiplo e Gênero Múltiplo respectivamente e foram usadas no sentido de *monovalorizar* atributos que são multivalorados, por exemplo, o gênero GM 138 diz respeito ao gênero composto pelos valores *Comedy*, *Drama*, *Fantasy*, *Romance* e *Thriller*.

Referências Bibliográficas

- [Agrawal et al. 1989] Agrawal, R., Borgida, A., e Jagadish, H. V. (1989). Efficient Management of Transitive Relationships in Large Data and Knowledge Bases. In *SIGMOD Conference*, pp. 253–262.
- [Agrawal e Wimmers 2000] Agrawal, R. e Wimmers, E. L. (2000). A framework for expressing and combining preferences. *SIGMOD Rec.*, 29(2):297–306.
- [Borzsonyi et al. 2001] Borzsonyi, S., Kossmann, D., e Stocker, K. (2001). The Skyline Operator. In *IEEE Conf. on Data Engineering*, pp. 421–430, Heidelberg, Germany.
- [Bouckaert et al. 1993] Bouckaert, R., Bouckaert, R. R., e Bouckaert, R. R. (1993). Probabilistic Network Construction Using the Minimum Description Length Principle. pp. 41–48.
- [Boutilier et al. 2004a] Boutilier, C., Brafman, R. I., Domshlak, C., Hoos, H. H., e Poole, D. (2004a). CP-nets: A Tool for Representing and Reasoning with Conditional Ceteris Paribus Preference Statements. *Journal of Artificial Intelligence Research (JAIR)*, 21:135–191.
- [Boutilier et al. 2004b] Boutilier, C., Brafman, R. I., Domshlak, C., Hoos, H. H., e Poole, D. (2004b). Preference-based Constrained Optimization with CP-nets. *Computational Intelligence*, 20(2):137–157.
- [Boutilier et al. 1999] Boutilier, C., Brafman, R. I., Hoos, H. H., e Poole, D. (1999). Reasoning with Conditional Ceteris Paribus Preference Statements. In *Proceedings of the 15th Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, pp. 71–80, Stockholm, Sweden. Morgan Kaufmann Publishers.
- [Brafman 2005] Brafman, R. I. (2005). Adaptive rich media presentations via preference-based constrained optimization. In *In Proceedings of the IJCAI-05 Workshop on Advances in Preference Handling*.
- [Brafman et al. 2004] Brafman, R. I., Domshlak, C., e Shimony, S. E. (2004). Qualitative decision making in adaptive presentation of structured information. *ACM Trans. Inf. Syst.*, 22(4):503–539.
- [Brafman et al. 2006] Brafman, R. I., Domshlak, C., Shimony, S. E., e Silver, Y. (2006). Preferences over Sets. In *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI)*, pp. 71–80, Menlo Park, California. AAAI Press.
- [Buntine 1996] Buntine, W. (1996). A Guide to the Literature on Learning Probabilistic Networks From Data.

- [Castañeda 1958] Castañeda, H. N. (1958). *On the Logic of Better*, volume 19,266. *Philosophy and Phenomenological Research*.
- [Cheng et al. 1997] Cheng, J., Bell, D. A., e Liu, W. (1997). Learning belief networks from data: an information theory based approach. In *CIKM '97: Proceedings of the sixth international conference on Information and knowledge management*, pp. 325–331, Las Vegas, Nevada, United States, New York, NY, USA. ACM.
- [Chomicki 2003] Chomicki, J. (2003). Preference formulas in relational queries. *ACM Transactions on Database Systems*, 28(4):427–466.
- [Cohen et al. 1998] Cohen, W. W., Schapire, R. E., e Singer, Y. (1998). Learning to Order Things. *Journal of Artificial Intelligence Research*, 10:243–270.
- [Cooper e Dietterich 1992] Cooper, G. F. e Dietterich, T. (1992). A Bayesian method for the induction of probabilistic networks from data. In *Machine Learning*, pp. 309–347.
- [Crammer e Singer 2001] Crammer, K. e Singer, Y. (2001). Pranking with Ranking. In *Advances in Neural Information Processing Systems 14*, pp. 641–647. MIT Press.
- [David et al. 1997] David, J. C., Bell, D. A., e Liu, W. (1997). An Algorithm for Bayesian Belief Network Construction from Data. In *In Proceedings of AI STAT 97*, pp. 83–90.
- [de Amo e Pereira 2010] de Amo, S. e Pereira, F. S. F. (2010). Evaluation of Conditional Preference Queries. *JIDM*, 1(3):503–518.
- [de Amo e Ribeiro 2009] de Amo, S. e Ribeiro, M. R. (2009). CPref-SQL: A Query Language Supporting Conditional Preferences. In *24th Annual ACM Symposium on Applied Computing (ACM SAC 2009)*, Honolulu, Hawaii, USA.
- [Domshlak et al. 2001] Domshlak, C., Brafman, R. I., e Shimony, S. E. (2001). Preference-based Configuration of Web Page Content. In *In Proceedings of Seventeenth International Joint Conference on Artificial Intelligence*, pp. 1451–1456.
- [Endres e Kießling 2008] Endres, M. e Kießling, W. (2008). Optimization of Preference Queries with Multiple Constraints. In *International Workshop on Personalized Access, Profile Management, and Context Awareness: Databases (PersDB)*, Auckland, New Zealand.
- [Ferreira et al. 2010] Ferreira, M. R. P., Pereira, F. S. F., de Amo, S., Traina, A. J. M., Jr., C. T., Silva, M. P., e Chbeir, R. (2010). Integrating User Preference to Similarity Queries over Medical Images Datasets. *The 23rd IEEE International Symposium on Computer-Based Medical System (CBMS 2010)*, Perth, Australia, 23rd.
- [Félix e de Amo 2010] Félix, N. e de Amo, S. (2010). CPrefMiner: Um Minerador Bayesiano de Preferências Condicionais. In *25th Brazilian Symposium on Databases.*, Belo Horizonte, Brazil.
- [Freund et al. 2003] Freund, Y., Iyer, R., Schapire, R. E., e Singer, Y. (2003). An efficient boosting algorithm for combining preferences. *J. Mach. Learn. Res.*, 4:933–969.
- [Habrant 1999] Habrant, J. (1999). Structure Learning of Bayesian Networks from Databases by Genetic Algorithms-Application to Time Series Prediction in Finance. In *ICEIS*, pp. 225–231.

- [Hansson 1996] Hansson, S. O. (1996). What is ceteris paribus preference? *Journal of Philosophical Logic*, 25(307-332).
- [Heckerman et al. 1994] Heckerman, D., Geiger, D., e Chickering, D. (1994). Learning Bayesian Networks: The Combination of Knowledge and Statistical Data. In *Machine Learning*, pp. 197–243.
- [Henrion 1986] Henrion, M. (1986). Propagating Uncertainty in Bayesian Networks by Probabilistic Logic Sampling. In *Proceedings of the Uncertainty in Artificial Intelligence 2 Annual Conference on Uncertainty in Artificial Intelligence (UAI-86)*, pp. 149–163, Amsterdam, NL. Elsevier Science.
- [Holland et al. 2003] Holland, S., Ester, M., e Kießling, W. (2003). Preference Mining: A Novel Approach on Mining User Preferences for Personalized Applications. In *PKDD*, pp. 204–216.
- [Jensen e Nielsen 2007] Jensen, F. V. e Nielsen, T. D. (2007). *Bayesian Networks and Decision Graphs*. Springer Publishing Company, Incorporated.
- [Jiang et al. 2008] Jiang, B., Pei, J., Lin, X., Cheung, D. W., e Han, J. (2008). Mining preferences from superior and inferior examples. In *KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 390–398, Las Vegas, Nevada, USA, New York, NY, USA. ACM.
- [Kießling 2002] Kießling, W. (2002). Foundations of preferences in database systems. In *VLDB '02: Proceedings of the 28th international conference on Very Large Data Bases*, pp. 311–322, Hong Kong, China. VLDB Endowment.
- [Kießling et al. 2001] Kießling, W., Holland, S., Hafenrichter, e Fischer (2001). Preference XPath: a Query Language for E-Commerce. In *PKDD*, pp. 253–262, Augsburg, Germany, USA.
- [Kießling e Köstler 2002] Kießling, W. e Köstler, G. (2002). Preference SQL - Design, Implementation, Experiences. In *28th International Conference International Conference on Very Large Data Bases (VLDB)*, pp. 990–1001, Hong Kong, China.
- [Koutrika e Ioannidis 2006] Koutrika, G. e Ioannidis, Y. (2006). Personalization of Queries based on User Preferences. In Bosi, G., Brafman, R. I., Chomicki, J., e Kießling, W. (editores), *Preferences: Specification, Inference, Applications*, número 04271 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany.
- [Koutrika et al. 2010] Koutrika, G., Pitoura, E., e Stefanidis, K. (2010). Representation, composition and application of preferences in databases. In *IEEE 26th International Conference on Data Engineering (ICDE)*, pp. 1214–1215.
- [Kron e Milovanovic 1975] Kron, A. e Milovanovic (1975). *Preference and Choice*, volume 6(2). Theory and Decisions.
- [Lam e Bacchus 1992] Lam, W. e Bacchus, F. (1992). Learning Bayesian Belief Networks - An approach based on the MDL Principle. *Computational Intelligence*, 10:269–293.

- [Larrañaga et al. 1994] Larrañaga, P., Poza, M., Yurramendi, Y., Murga, R. H., e Kuijpers, C. M. H. (1994). Structure Learning of Bayesian Networks by Genetic Algorithms: A Performance Analysis of Control Parameters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18:912–926.
- [Larrañaga et al. 1997] Larrañaga, P., Sierra, B., Gallego, M. J., Michelena, M. J., e Picaza, J. M. (1997). Learning Bayesian Networks by Genetic Algorithms: A Case Study in the Prediction of Survival in Malignant Skin Melanoma. In *AIME*, pp. 261–272.
- [Lin et al. 2007] Lin, Lee, W., e Deng, D. L. (2007). Mining User preference using Spy voting for search engine personalization. *ACM Trans. Internet Technol.*, 7(4):19.
- [Mobasher 2007] Mobasher, B. (2007). Recommender Systems. *KI*, 21(3):41–43.
- [Mobasher et al. 2000] Mobasher, B., Cooley, R., e Srivastava, J. (2000). Automatic personalization based on Web usage mining. *Commun. ACM*, 43:142–151.
- [Mobasher et al. 2002] Mobasher, B., Dai, H., Luo, T., e Nakagawa, M. (2002). Discovery and Evaluation of Aggregate Usage Profiles for Web Personalization. *Data Min. Knowl. Discov.*, 6:61–82.
- [Neapolitan 1990] Neapolitan, R. E. (1990). *Probabilistic reasoning in expert systems: theory and algorithms*. John Wiley & Sons, Inc., New York, NY, USA.
- [Nguyen e Haddawy 1999] Nguyen, H. e Haddawy, P. (1999). The Decision-Theoretic Interactive Video Advisor. In *UAI*, pp. 494–501.
- [Pearl 1988] Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.
- [Quinlan 1990] Quinlan, J. R. (1990). Induction of Decision Trees. In Shavlik, J. W. e Dietterich, T. G. (editores), *Readings in Machine Learning*. Morgan Kaufmann. Originally published in *Machine Learning* 1:81–106, 1986.
- [Radde e Freitag 2010] Radde, S. e Freitag, B. (2010). Using Bayesian Networks To Infer Product Rankings From User Needs. In *Workshop on Intelligent Techniques for Web Personalisation and Recommender Systems.*, Passau, Germany.
- [Resnick e Varian 1997] Resnick, P. e Varian, H. R. (1997). Recommender systems. *Commun. ACM*, 40(3):56–58.
- [Robinson 1977] Robinson, R. (1977). Counting unlabeled acyclic digraphs. In C.H.C. Little (Ed.), *Lecture notes in mathematics Combinatorial mathematics 622*.
- [Rosen 1991] Rosen, K. H. (1991). *Discrete mathematics and its applications (2nd ed.)*. McGraw-Hill, Inc., New York, NY, USA.
- [Sarwar et al. 2000] Sarwar, B., Karypis, G., Konstan, J., e Riedl, J. (2000). Analysis of Recommendation Algorithms for E-Commerce.
- [Scheinerman 2003] Scheinerman, E. R. (2003). *Matemática Discreta - Uma Introdução*. Thomson.

- [Trapp 1985] Trapp, R. W. (1985). *Utility theory and preference logic*, volume 22(1-3). *Erkenntnis*.
- [Vojtás e Eckhardt 2008] Vojtás, P. e Eckhardt, A. (2008). Considering Data-Mining Techniques in User Preference Learning. *Web Intelligence and Intelligent Agent Technology, IEEE/WIC/ACM International Conference on*, 3:33–36.
- [von Wright 1963] von Wright, G. H. (1963). *The Logic of Preference: An Essay*. Edinburgh University Press.
- [von Wright 1972] von Wright, G. H. (1972). *The Logic of Preference reconsidered*, volume 3. Theory and Decisions.
- [Wilson 2004] Wilson, N. (2004). Extending CP-Nets with Stronger Conditional Preference Statements. In *19th National Conference on Artificial Intelligence (AAAI)*, pp. 735–741, San Jose, California, USA.