Aula 12 – Mineração de Dados
Regras de Associação

Profa. Elaine Faria

UFU

Material

- Este material foi construído por meio de traduções dos slides do do prof. Tan, disponíveis em:
  - https://www-users.cse.umn.edu/~kumar001/dmbook/index.php
  - Todas as figuras usadas foram retiradas dos slides do prof. Tan

- O material é baseado no livro
  - Tan P., SteinBack M. e Kumar V. Introduction to Data Mining, Pearson, 2006.
  - Faceli, K., Lorena, A. C., Gama, J., Carvalho, A. C. P. L. F., Inteligência Artificial: Uma abordagem de Aprendizado de Máquina, LTC, 2011.

Regra de Associação - Motivação

- Suponha que um gerente de supermercado esteja interessado em saber os hábitos de compra de seus clientes
  - Quais produtos os clientes costumam comprar ao mesmo tempo?
    - Planejar os catálogos de supermercado
    - Planejar os folhetos de promoção de produto
    - Promover Campanhas de publicidade
    - Organizar a localização dos produtos
  - Fonte de Dados para responder tais perguntas
    - Banco de dados de transações efetuadas pelos clientes

Regras de Associação

- Dado um conjunto de transações, encontrar regras que irão predizer a ocorrência de um item baseado na ocorrência dos outros itens da transação

Transações Carrinho do Supermercado

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Exemplo de Regra de Associação

{Diaper} → {Beer},
{Milk, Bread} → {Eggs,Coke},
{Beer, Bread} → {Milk},

Implicações significam co-ocorrências, não causalidade!

Definição: Itemset frequente

- **Itemset**
  - Uma coleção de um ou muitos itens
    - Exemplo: {Milk, Bread, Diaper}
  - k-itemset
    - Um itemset que contém k itens
- **Contagem do Suporte (σ)**
  - Frequência de ocorrência de um itemset
  - E.g. σ({Milk, Bread,Diaper}) = 2
- **Suporte**
  - Fração de transações que contém um itemset
  - E.g. s({Milk, Bread, Diaper}) = 2/5
- **Itemset Frequente**
  - Um itemset cujo suporte é maior ou igual a um limiar *minsup*

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Definição: Regra de Associação

- Regra de Associação
  - Uma expressão de implicação na forma $X \rightarrow Y$, onde X e Y são itemsets
  - Exemplo:
    {Milk, Diaper} $\rightarrow$ {Beer}

- Métrica para avaliação de regras
  - Supporte(s)
    - Fração de transações que contém X e Y
  - Confiança (c)
    - Medida de quão frequente items em Y aparecem na transação que contém X

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Exemplo:

$$\{\text{Milk}, \text{Diaper}\} \Rightarrow \{\text{Beer}\}$$

$$s = \frac{\sigma(\text{Milk}, \text{Diaper}, \text{Beer})}{|\text{T}|} = \frac{2}{5} = 0.4$$

$$c = \frac{\sigma(\text{Milk}, \text{Diaper}, \text{Beer})}{\sigma(\text{Milk}, \text{Diaper})} = \frac{2}{3} = 0.67$$

Definição: Regra de Associação

- Regra de Associação - Atenção
  - Exemplo
    {Milk, Diaper} → {Beer}
  - Significa que quem compra Milk e Diaper compra também Beer
  - Isso é diferente de
    {Beer} → {Milk, Diaper}

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Tarefa de Mineração de Regra de Associação

- Dado um conjunto de transações T, objetivo da regra de associação é encontrar todas as regras tendo
  - suporte ≥ limiar *minsup*
  - confiança ≥ limiar *minconf*

- Proposta de força-bruta:
  - Liste todas as possíveis regras de associação
  - Calcule o suporte e a confiança de cada regra
  - Pode as regras que não alcancem os limiares de *minsup* and *minconf*

  ⇒ Computacionamente proibitivo!

Minerando regras de associação

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Exemplos de regras:

{Milk,Diaper} → {Beer} (s=0.4, c=0.67)
{Milk,Beer} → {Diaper} (s=0.4, c=1.0)
{Diaper,Beer} → {Milk} (s=0.4, c=0.67)
{Beer} → {Milk,Diaper} (s=0.4, c=0.67)
{Diaper} → {Milk,Beer} (s=0.4, c=0.5)
{Milk} → {Diaper,Beer} (s=0.4, c=0.5)

Observações

- Todas as regras acima são partições binárias do mesmo itemset:
  {Milk, Diaper, Beer}
- Regras originárias do mesmo itemset tem suporte idêntico, mas podem ter diferentes confianças
- Assim, nós podemos separar os requisitos de suporte e confiança

Minerando regras de associação

- Proposta em 2 passos:
  1. **Geração de itemsets frequentes**
     - Gerar todos os itemsets frequentes cujo suporte $\geq$ minsup
  2. **Geração de regras**
     - Gerar regras de alta confiança a partir de cada itemset frequente, onde cada regra é uma partição binária de um itemset frequente

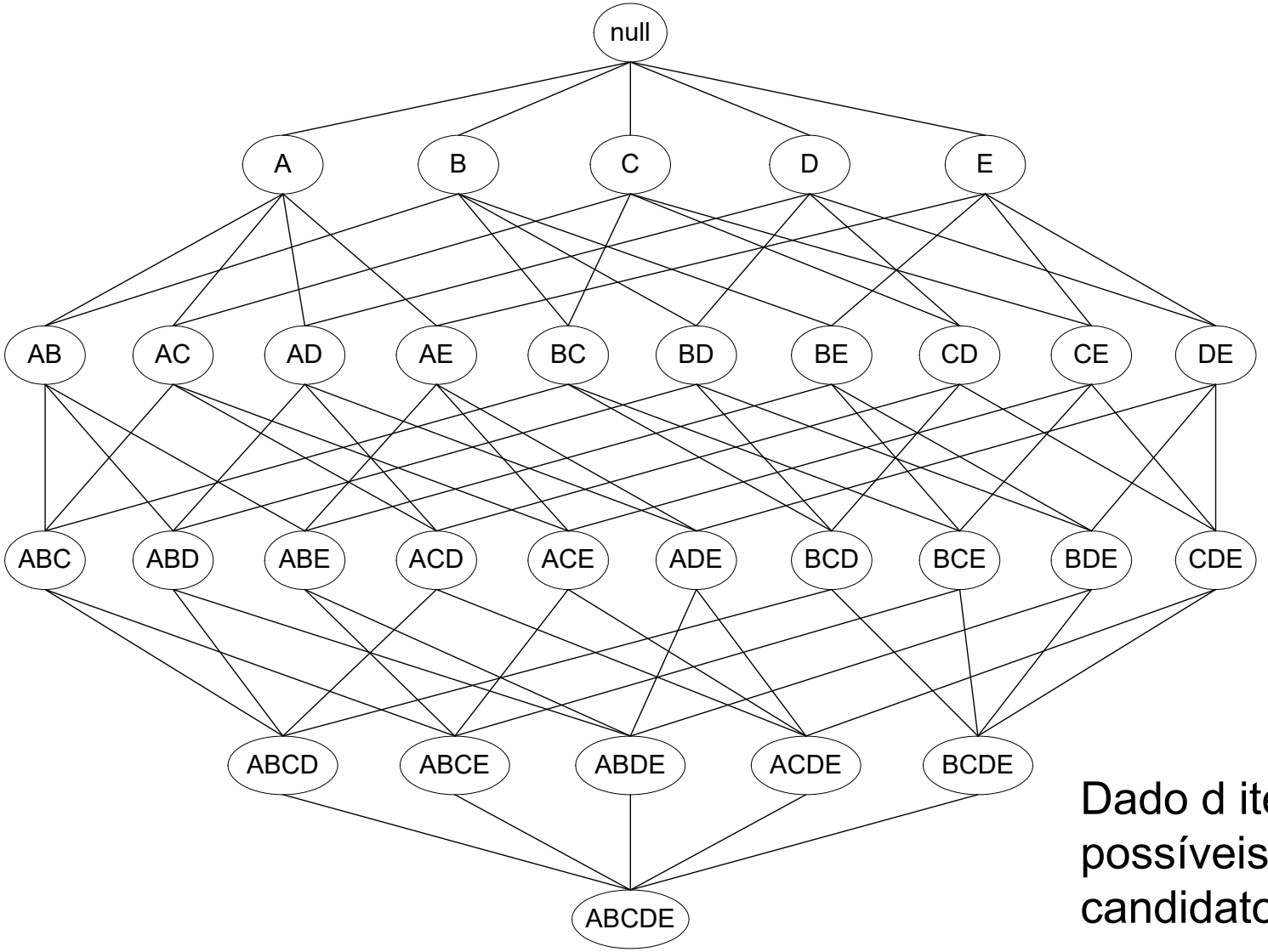
- Geração de itemsets frequentes é ainda computacionalmente cara

Geração de itemsets frequentes

- Dados
  - um conjunto A = $\{a_1, ...a_m\}$ de itens
  - uma tabela T = $(t_1, ..., t_n)$ de transações sobre A

- Objetivo
  - encontrar o conjunto de itens frequentes (itemset), tais que o suporte relativo de cada conjunto de itens é maior ou igual ao mínimo definido pelo usuário

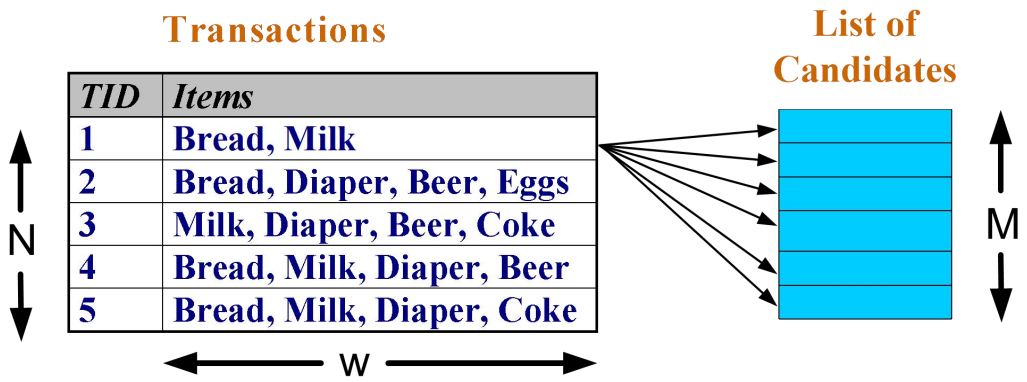
Geração de itemsets frequentes



Dado d itens, há 2^d possíveis itemsets candidatos

Geração de itemsets frequentes

- Proposta de força-bruta:
  - Cada itemset na grade é um candidato a itemset frequente
  - Contar o suporte de cada candidato escaneando a base de dados



  - Combinar cada transação contra cada candidato
  - Complexidade ~ O(NMw) => Caro porque M = 2^d !!!

Estratégias para geração de itemsets frequentes

- Reduzir o número de candidatos (M)
  - Pesquisa completa: $M=2^d$
  - Usar técnicas de poda para reduzir M

- Reduzir o número de transações (N)
  - Reduzir o tamanho de N à medida que o tamanho do itemset aumenta
  - Usado por algoritmos de mineração DHP e vertical-based

- Reduzir o número de comparações (NM)
  - Uso eficiente de estruturas de dados para armazenar os candidatos ou transações
  - Não é necessário combinar cada candidato contra cada transação

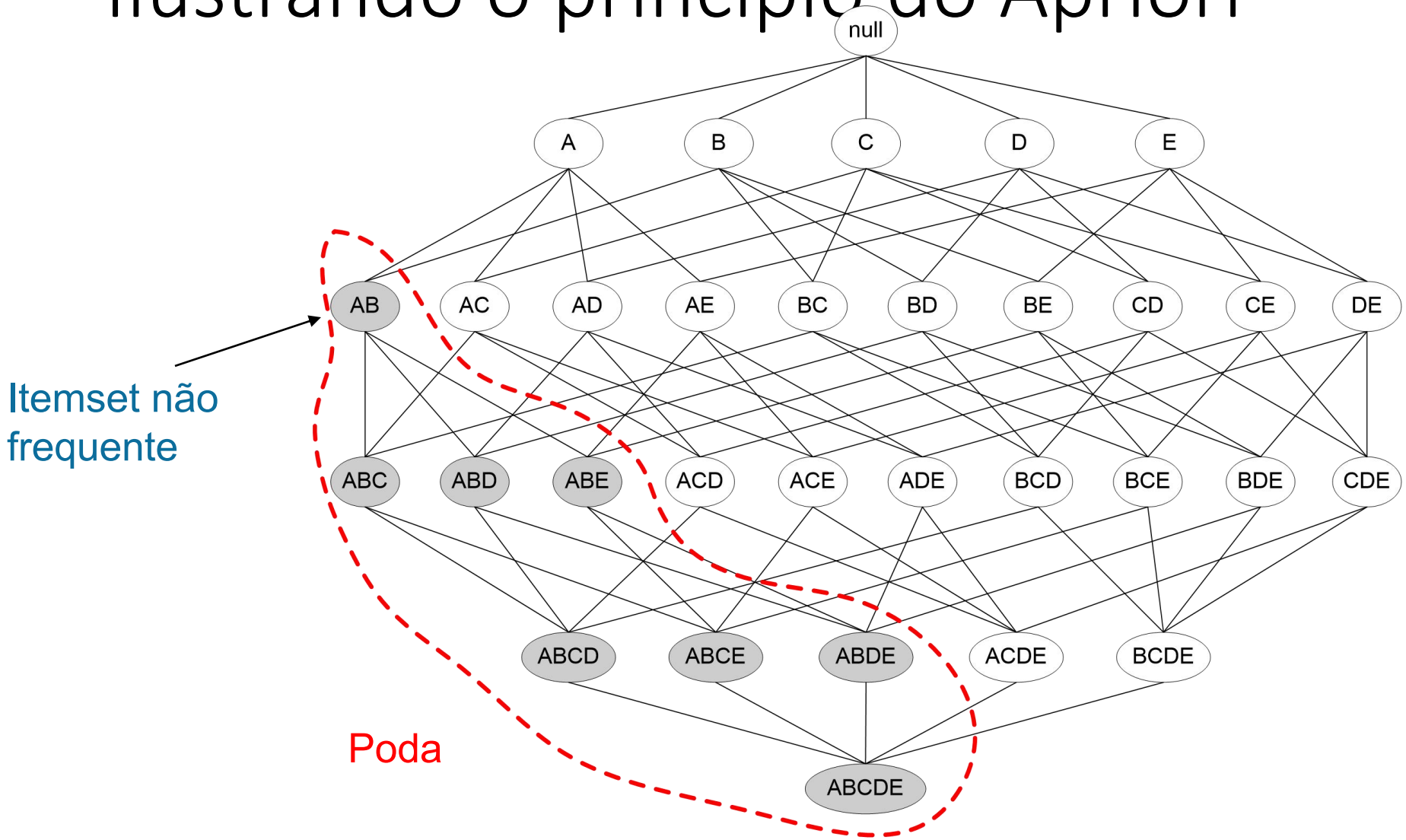
Reduzindo o nro de candidatos

- Princípio do Apriori:
  - Se um itemset é frequente, então todos os seus subconjuntos são também frequentes

- O princípio a Priori é válido devido à seguinte propriedade da medida de suporte:

  - O suporte de um itemset nunca excede o suporte dos seus subconjuntos
  - Isto é conhecido como a propriedade do suporte anti-monótona

Ilustrando o princípio do Apriori



Ilustrando o princípio do Apriori

TID	*Items*
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

Items (1-itemsets)

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Suporte mínimo = 3

Se cada subconjunto é considerado

$^6C_1 + {}^6C_2 + {}^6C_3$

$6 + 15 + 20 = 41$

Ilustrando o princípio do Apriori

TID	*Items*
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

Items (1-itemsets)

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Minimum Support = 3

Se cada subconjunto é considerado

$^6C_1 + {}^6C_2 + {}^6C_3$

6 + 15 + 20 = 41

Com a poda baseada no suporte,

6 + 6 + 4 = 16

Ilustrando o princípio do Apriori

TID	Items
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Items (1-itemsets)

Itemset
{Bread,Milk}
{Bread, Beer }
{Bread,Diaper}
{Beer, Milk}
{Diaper, Milk}
{Beer,Diaper}

Pares (2-itemsets)

(Não é necessário gerar candidatos envolvendo Coke ou Eggs)

Suporte mínimo = 3

Se cada subconjunto é considerado

$^6C_1 + {}^6C_2 + {}^6C_3$

$6 + 15 + 20 = 41$

Com a poda baseada no suporte,

$6 + 6 + 4 = 16$

Ilustrando o princípio do Apriori

TID	Items
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Items (1-itemsets)

Itemset	Count
{Bread,Milk}	3
{Beer, Bread}	2
{Bread,Diaper}	3
{Beer,Milk}	2
{Diaper,Milk}	3
{Beer,Diaper}	3

Pares (2-itemsets)

(Não é necessário gerar candidatos envolvendo Coke ou Eggs)

Suporte mínimo = 3

Se cada subconjunto é considerado

$^6C_1 + {}^6C_2 + {}^6C_3$

6 + 15 + 20 = 41

Com a poda baseada no suporte,

6 + 6 + 4 = 16

Ilustrando o princípio do Apriori

TID	Items
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Items (1-itemsets)

Itemset
{Bread,Milk}
{Bread, Beer }
{Bread,Diaper}
{Beer, Milk}
{Diaper, Milk}
{Beer,Diaper}

Pares (2-itemsets)

(Não é necessário gerar candidatos envolvendo Coke ou Eggs)

Suporte mínimo = 3

Itemset
{ Beer, Diaper, Milk}
{ Beer,Bread,Diaper}
{Bread,Diaper,Milk}
{ Beer, Bread, Milk}

Triplas (3-itemsets)

Se cada subconjunto é considerado

$${}^6C_1 + {}^6C_2 + {}^6C_3$$

$$6 + 15 + 20 = 41$$

Com a poda baseada no suporte,

$$6 + 6 + 4 = 16$$

Ilustrando o princípio do Apriori

TID	Items
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Items (1-itemsets)

Itemset
{Bread,Milk}
{Bread, Beer }
{Bread,Diaper}
{Beer, Milk}
{Diaper, Milk}
{Beer,Diaper}

Pares (2-itemsets)

(Não é necessário gerar candidatos envolvendo Coke ou Eggs)

Suporte mínimo = 3

Triplets (3-itemsets)

Itemset	Count
{ Beer, Diaper, Milk}	2
{ Beer,Bread, Diaper}	2
{Bread, Diaper, Milk}	2
{Beer, Bread, Milk}	1

Se cada subconjunto é considerado

$${}^6C_1 + {}^6C_2 + {}^6C_3$$

$$6 + 15 + 20 = 41$$

Com a poda baseada no suporte,

$$6 + 6 + 4 = 16$$

Ilustrando o princípio do Apriori

TID	Items
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Items (1-itemsets)

Itemset
{Bread,Milk}
{Bread, Beer }
{Bread,Diaper}
{Beer, Milk}
{Diaper, Milk}
{Beer,Diaper}

Pares (2-itemsets)

(Não é necessário gerar candidatos envolvendo Coke ou Eggs)

Suporte mínimo = 3

Se cada subconjunto é considerado

$^6C_1 + {}^6C_2 + {}^6C_3$

6 + 15 + 20 = 41

Com a poda baseada no suporte,

6 + 6 + 4 = 16

6 + 6 + 1 = 13

Triplets (3-itemsets)

Itemset	Count
{ Beer, Diaper, Milk}	2
{ Beer,Bread, Diaper}	2
{Bread, Diaper, Milk}	2
{Beer, Bread, Milk}	1

Algoritmo Apriori

- F_k: k-itemsets frequentes
- L_k: k-itemsets candidatos

- **Algoritmo**
  - Seja k=1
  - Gere F_1 = {1-itemsets frequentes}
  - Repita até que F_k seja vazio
    - **Geração de Candidatos**: Gere L_{k+1} a partir de F_k
    - **Poda dos Candidatos**: Pode os itemsets candidatos em L_{k+1} contendo subconjuntos de tamanho k que são infrequentes
    - **Contagem do Suporte**: Conte o suporte de cada candidato em L_{k+1} escaneando a base
    - **Eliminação de Candidatos**: Elimine candidatos em L_{k+1} que são infrequentes, deixando somente aqueles que são frequentes => F_{k+1}

Geração de candidatos: Método da força bruta

TID	Items
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

Items

Item
Beer
Bread
Cola
Diapers
Eggs
Milk

Candidate Generation

Itemset
{Beer, Bread, Cola}
{Beer, Bread, Diapers}
{Beer, Bread, Eggs}
{Beer, Bread, Milk}
{Beer, Cola, Diapers}
{Beer, Cola, Eggs}
{Beer, Cola, Milk}
{Beer, Diapers, Eggs}
{Beer, Diapers, Milk}
{Beer, Eggs, Milk}
{Bread, Cola, Diapers}
{Bread, Cola, Eggs}
{Bread, Cola, Milk}
{Bread, Diapers, Eggs}
{Bread, Diapers, Milk}
{Bread, Eggs, Milk}
{Cola, Diapers, Eggs}
{Cola, Diapers, Milk}
{Cola, Eggs, Milk}
{Diapers, Eggs, Milk}

Candidate Pruning

Itemset
{Bread, Diapers, Milk}

Frequent 2-itemset

Itemset
{Beer, Diapers}
{Bread, Diapers}
{Bread, Milk}
{Diapers, Milk}

Figure 5.6. A brute-force method for generating candidate 3-itemsets.

Geração dos Candidatos: Unir os itemsets F_{k-1} e F_1

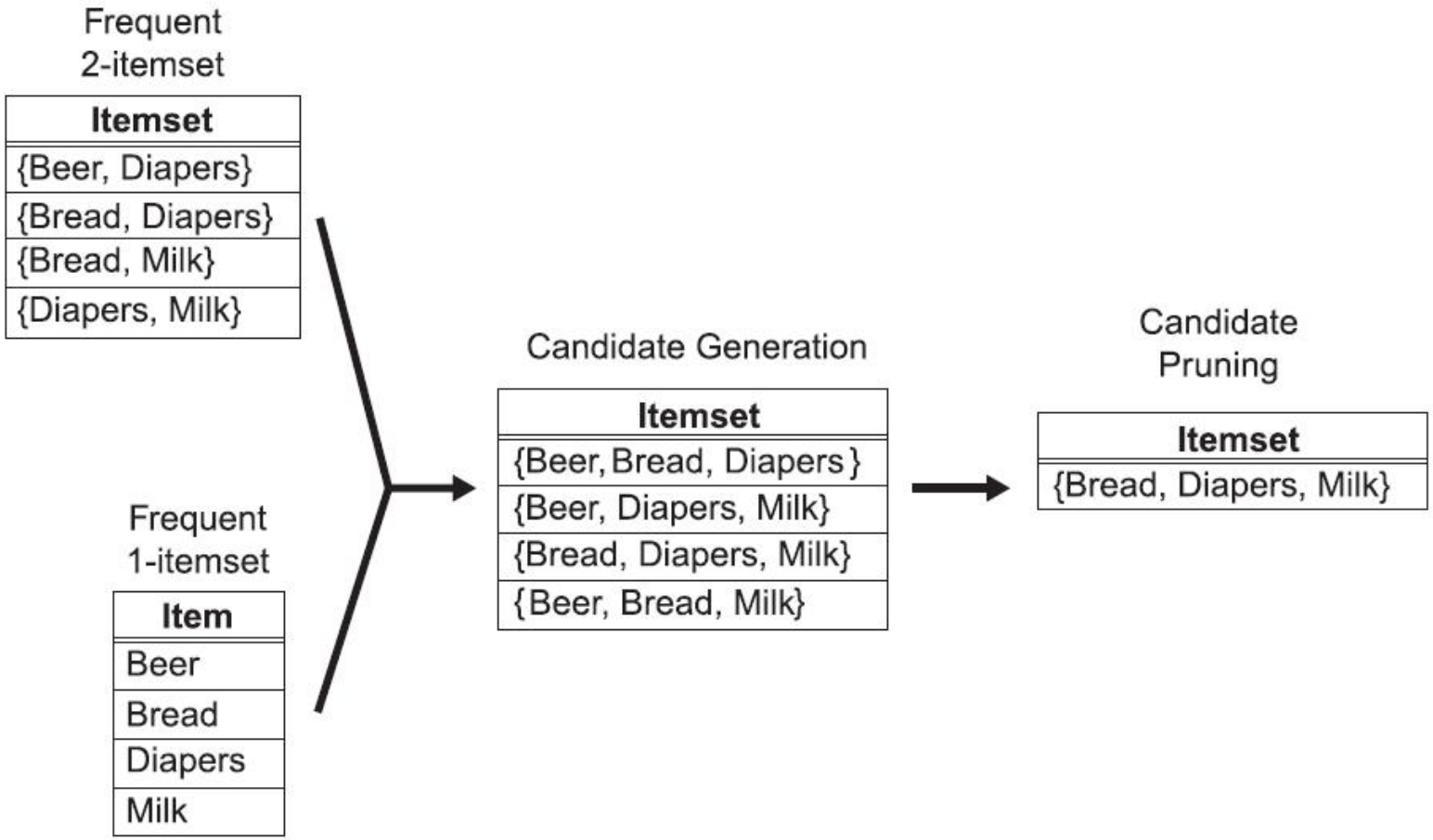


Figure 5.7. Generating and pruning candidate k-itemsets by merging a frequent $(k-1)$-itemset with a frequent item. Note that some of the candidates are unnecessary because their subsets are infrequent.

Geração de Candidatos: Método F_{k-1} x F_{k-1}

- Unir dois (k-1)-itemsets frequentes se os seus (k-2) primeiros items são idênticos
- F_3 = {ABC,ABD,ABE,ACD,BCD,BDE,CDE}
  - Unir (**AB**C, **AB**D) = **AB**CD
  - Unir (**AB**C, **AB**E) = **AB**CE
  - Unir (**AB**D, **AB**E) = **AB**DE

  - Não unir (**A**BD,**A**CD) porque eles dividem o mesmo prefixo de tamanho 1 ao invés de tamanho 2
- Método usado pelo **algoritmo apriori**

Poda dos Candidatos

- Seja F_3 = {ABC,ABD,ABE,ACD,BCD,BDE,CDE} o conjunto de 3-itemsets frequentes

- L_4 = {ABCD,ABCE,ABDE} é o conjunto de 4-itemsets candidatos gerados (ver slide anterior)

- Poda dos candidatos
  - Podar ABCE porque ACE e BCE são infrequentes
  - Podar ABDE porque ADE é infrequente

- Após podar os candidatos: L_4 = {ABCD}

Geração dos Candidatos: Unir os itemsets F_{k-1} x F_{k-1}

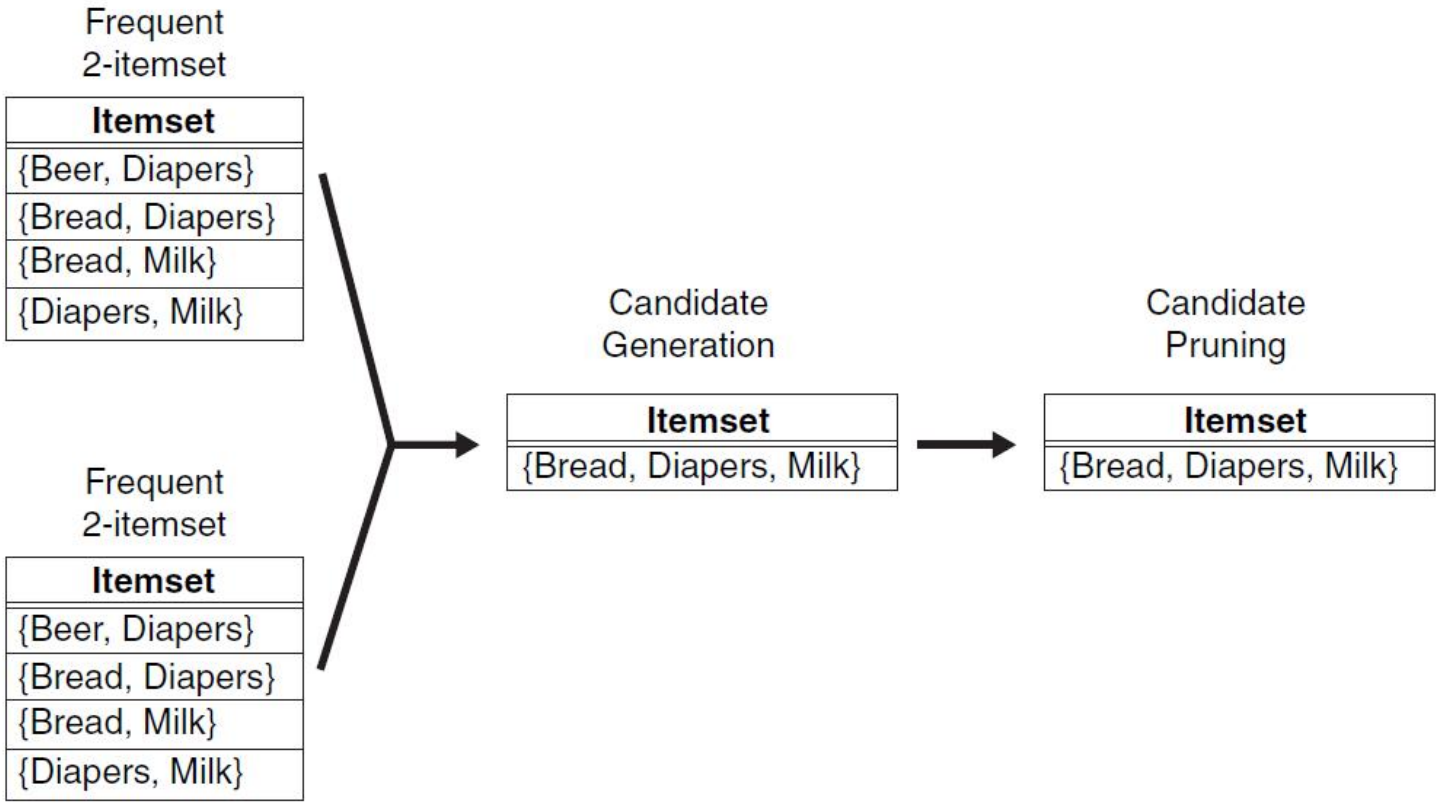


Figure 5.8. Generating and pruning candidate k-itemsets by merging pairs of frequent $(k-1)$-itemsets.

Ilustrando o princípio do Apriori

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Items (1-itemsets)

Itemset	Count
{Bread,Milk}	3
{Bread,Beer}	2
{Bread,Diaper}	3
{Milk,Beer}	2
{Milk,Diaper}	3
{Beer,Diaper}	3

Pares (2-itemsets)

(Não é necessário gerar candidatos envolvendo Coke ou Eggs)

Minimum Support = 3

If every subset is considered,

$^6C_1 + {}^6C_2 + {}^6C_3$

6 + 15 + 20 = 41

With support-based pruning,

6 + 6 + 1 = 13

Triplets (3-itemsets)

Itemset	Count
{Bread, Diaper, Milk}	2

Uso do método $F_{k-1} \times F_{k-1}$ para geração dos candidatos resulta em somente um 3-itemset. Este é eliminado depois do passo de contagem do suporte

Método alternativo para F_{k-1} x F_{k-1}

- Unir (k-1)-itemsets frequentes se os últimos (k-2) items do primeiro são idênticos aos primeiros (k-2) items do segundo.

- F_3 = {ABC,ABD,ABE,ACD,BCD,BDE,CDE}
  - Unir (A**BC**, **BC**D) = A**BC**D
  - Unir (A**BD**, **BD**E) = A**BD**E
  - Unir (A**CD**, **CD**E) = A**CD**E
  - Unir (B**CD**, **CD**E) = B**CD**E

Poda dos candidatos para o método Alternativo F_{k-1} x F_{k-1}

- Seja F_3 = {ABC,ABD,ABE,ACD,BCD,BDE,CDE} o conjunto de 3-itemsets frequentes

- L_4 = {ABCD,ABDE,ACDE,BCDE} é o conjunto de 4-itemsets candidatos gerados (ver slide anterior)
- Poda dos candidatos
  - Podar ABDE porque ADE é infrequente
  - Podar ACDE porque ACE e ADE são infrequentes
  - Podar BCDE porque BCE
- Após a poda dos candidatos: L_4 = {ABCD}

Contagem do Suporte dos Itemsets Candidatos

- Percorrer a base de dados de transações para determinar o suporte de cada itemset candidato
  - Deve-se combinar cada itemset candidato contra cada transação, o que é uma operação cara

TID	*Items*
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

Itemset
{ Beer, Diaper, Milk}
{ Beer,Bread,Diaper}
{Bread, Diaper, Milk}
{ Beer, Bread, Milk}

  - Para reduzir o nro de comparações, pode-se armazenar os itemsets candidatos em uma estrutura *hash*

Geração de Regras

- Dado um itemset frequente L, encontre todos os subconjuntos não-vazios $f \subset L$ tal que $f \rightarrow L - f$ satisfaça o requisito de confiança mínima
  - Se {A,B,C,D} é um itemset frequente, regras candidatas:

$$ABC \rightarrow D, \quad ABD \rightarrow C, \quad ACD \rightarrow B, \quad BCD \rightarrow A,$$
$$A \rightarrow BCD, \quad B \rightarrow ACD, \quad C \rightarrow ABD, \quad D \rightarrow ABC$$
$$AB \rightarrow CD, \quad AC \rightarrow BD, \quad AD \rightarrow BC, \quad BC \rightarrow AD,$$
$$BD \rightarrow AC, \quad CD \rightarrow AB,$$

- Se $|L| = k$, então há $2^k - 2$ regras de associação candidatas (ignorar $L \rightarrow \emptyset$ and $\emptyset \rightarrow L$)

Fatores afetando a Complexidade do Apriori

- Escolha do limiar do suporte mínimo
  - reduzir o limite de suporte resulta em mais conjuntos de itemsets frequentes
  - isso pode aumentar o número de candidatos e o comprimento máximo dos itemsets frequentes
- Dimensionalidade (número de items) da base de dados
  - É necessário mais espaço para armazenar a contagem do suporte
  - Se o número de conjuntos de itens frequentes também aumentar, os custos de computação e E / S também podem aumentar
- Tamanho da base de dados
  - o tempo de execução do algoritmo aumenta com o número de transações
- Largura média das transações
  - a largura da transação aumenta o comprimento máximo de conjuntos de itens
  - o número de subconjuntos em uma transação aumenta com sua largura, aumentando o tempo de computação para contagem de suporte

Leitura Recomendada

- Leitura do
  - Capítulo 6 (até a seção 6.3) do livro Tan et al, 2006.