# 第5章：挖掘关联规则

- 关联规则挖掘
- 事务数据库中(单维布尔)关联规则挖掘的可伸缩算法
- 挖掘各种关联/相关规则
- 基于限制的关联挖掘-
- 顺序模式挖掘
- 小结

# 关联规则

- 关联规则反映一个事物与其他事物之间的相互依存性和关联性。如果两个或者多个事物之间存在一定的关联关系，那么，其中一个事物就能够通过其他事物预测到。

- 典型的关联规则发现问题是对超市中的货篮数据（**Market Basket**）进行分析。通过发现顾客放入货篮中的不同商品之间的关系来分析顾客的购买习惯。

# 什么是关联规则挖掘

- **关联规则挖掘**
  - **首先被Agrawal, Imielinski and Swami在1993年的SIGMOD会议上提出**
  - **在事务、关系数据库中的项集和对象中发现频繁模式、关联规则、相关性或者因果结构**
  - **频繁模式: 数据库中频繁出现的项集**
- **目的: 发现数据中的规律**
  - **超市数据中的什么产品会一起购买？— 啤酒和尿布**
  - **在买了一台PC之后下一步会购买?**
  - **哪种DNA对这种药物敏感?**
  - **我们如何自动对Web文档进行分类?**

# 频繁模式挖掘的重要性

- 许多重要数据挖掘任务的基础
  - 关联、相关性、因果性
  - 序列模式、空间模式、时间模式、多维
  - 关联分类、聚类分析

- 更加广泛的用处
  - 购物篮分析、交叉销售、直销
  - 点击流分析、DNA序列分析等等

# 关联规则基本模型

- **IBM公司Almaden研究中心的R.Agrawal首先提出关联规则模型，并给出求解算法AIS。随后又出现了SETM和Apriori等算法。其中，Apriori是关联规则模型中的经典算法。**
  - **给定一组事务**
  - **产生所有的关联规则**
  - **满足最小支持度和最小可信度**

# 关联规则基本模型

- 设$I=\{i_1,\ldots,i_m\}$为所有项目的集合，$D$为事务数据库，事务$T$是一个项目子集（$T\subseteq I$）。每一个事务具有唯一的事务标识$TID$。
- 设$A$是一个由项目构成的集合，称为**项集**。事务$T$包含项集$A$，当且仅当$A\subseteq T$。
  - 如果项集$A$中包含$k$个项目，则称其为**$k$项集**。
- 项集$A$在事务数据库$D$中出现的次数占$D$中总事务的百分比叫做项集的**支持度**。
- 如果项集的支持度超过用户给定的**最小支持度阈值**，就称该项集是**频繁项集**（或**大项集**）。

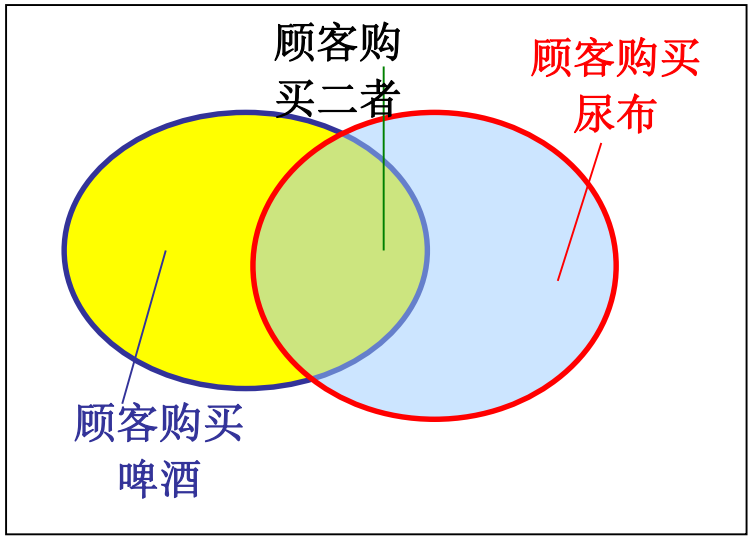
| 交易ID | 购买的商品 |
|---|---|
| 2000 | A,B,C |
| 1000 | A,C |
| 4000 | A,D |
| 5000 | B,E,F |

# 关联规则基本模型

- 关联规则是形如$X \Rightarrow Y$的逻辑蕴含式，其中$X \subset I$，$Y \subset I$，且$X \cap Y = \varnothing$。
- 如果事务数据库$D$中有$s\%$的事务包含$X \cup Y$，则称关联规则$X \Rightarrow Y$的**支持度为$s\%$**
  - 实际上，支持度是一个概率值。是一个相对计数。
  - $support\ (X \Rightarrow Y) = P\ (X \cup Y)$
- 项集的**支持度计数(频率) support_count**
  - 包含项集的事务数
- 若项集$X$的**支持度**记为$support\ (X)$，规则的**信任度**为$support\ (X \cup Y)$ / $support\ (X)$。
  - 是一个条件概率$P\ (Y \mid X)$。$confidence\ (X \Rightarrow Y) = P\ (Y \mid X)$
  - $= support\_count(X \cup Y)$ / $support\_count\ (X)$

# 频繁模式和关联规则

| Transaction-id | Items bought |
|---|---|
| 10 | A, B, D |
| 20 | A, C, D |
| 30 | A, D, E |
| 40 | B, E, F |
| 50 | B, C, D, E, F |




- **Itemset X={$x_1$, …, $x_k$}**
- 找出满足最小支持度和置信度的所规则 ***X*→*Y***
  - 支持度, *s*, 事务包含 X∪Y 的概率
  - 置信度, *c*, 事务含 X 也包含 *Y* 的条件概率.

令 $sup_{min}$ = 50%, $conf_{min}$ = 50%

*Freq. Pat.:* {*A:3, B:3, D:4, E:3, AD:3*}

关联规则Association rules:

*A* → *D* (60%, 100%)

*D* → *A* (60%, 75%)

# 挖掘关联规则—一个例子

| Transaction-id | Items bought |
|---|---|
| 10 | A, B, C |
| 20 | A, C |
| 30 | A, D |
| 40 | B, E, F |

最小支持度 50%
最小置信度 50%

| Frequent pattern | Support |
|---|---|
| {A} | 75% |
| {B} | 50% |
| {C} | 50% |
| {A, C} | 50% |

规则 $A \Rightarrow C$:

支持度 = support({$A$}∪{$C$}) = 50%

置信度 = support({$A$}∪{$C$})/support({$A$}) = 66.6%

# 闭频繁项集 and 极大频繁项集

- 一个长模式包含子模式的数目, **e.g., $\{a_1, \ldots, a_{100}\}$ contains $\binom{100}{1} + \binom{100}{2} + \ldots + \binom{1}{1}\binom{0}{0}\binom{0}{0} = 2^{100} - 1 = 1.27*10^{30}$ sub-patterns!**

- 解: ***Mine closed patterns and max-patterns instead***
- 一个频繁项集**X** 是闭的，如果 **X** 是频繁的，且不存在真超项集***no super-pattern* Y Ↄ X,** 有相同的支持度计数
  - **(proposed by Pasquier, et al. @ ICDT'99)**

- 项集 **X**是极大频繁项集 **if X is frequent and there exists no frequent super-pattern Y Ↄ X**
  - **(proposed by Bayardo @ SIGMOD'98)**
- 两者有不同，极大频繁项集定义中对真超集要松一些。

# 闭频繁项集 and 极大频繁项集

- **Exercise. DB = {<$a_1$, …, $a_{100}$>, < $a_1$, …, $a_{50}$>}**
  - **Min_sup = 1.**
- **What is the set of closed itemset?**
  - **<$a_1$, …, $a_{100}$>: 1**
  - **< $a_1$, …, $a_{50}$>: 2**
- **What is the set of max-pattern?**
  - **<$a_1$, …, $a_{100}$>: 1**
- **What is the set of all patterns?**
  - **!!**

# 关联规则基本模型

- **关联规则就是支持度和信任度分别满足用户给定阈值的规则。**
- 
- **发现关联规则需要经历如下两个步骤：**
  - **找出所有频繁项集。**
  - **由频繁项集生成满足最小信任度阈值的规则。**

# 第5章：挖掘关联规则

- 关联规则挖掘
- 事务数据库中(单维布尔)关联规则挖掘的可伸缩算法
- 挖掘各种关联/相关规则
- 基于限制的关联挖掘-
- 顺序模式挖掘
- 小结

# Apriori算法的步骤

- **Apriori算法命名源于算法使用了频繁项集性质的先验（Prior）知识。**

- **Apriori算法将发现关联规则的过程分为两个步骤：**
  - 通过迭代，检索出事务数据库中的所有频繁项集，即支持度不低于用户设定的阈值的项集；
  - 利用频繁项集构造出满足用户最小信任度的规则。

- **挖掘或识别出所有频繁项集是该算法的核心，占整个计算量的大部分。**

# 频繁项集

- 为了避免计算所有项集的支持度（实际上频繁项集只占很少一部分），Apriori算法引入潜在频繁项集的概念。
- 若潜在频繁$k$项集的集合记为$C_k$，频繁$k$项集的集合记为$L_k$，$m$个项目构成的$k$项集的集合为$C_m^k$，则三者之间满足关系$L_k \subseteq C_k \subseteq C_m^k$。
- 构成潜在频繁项集所遵循的原则是“频繁项集的子集必为频繁项集”。

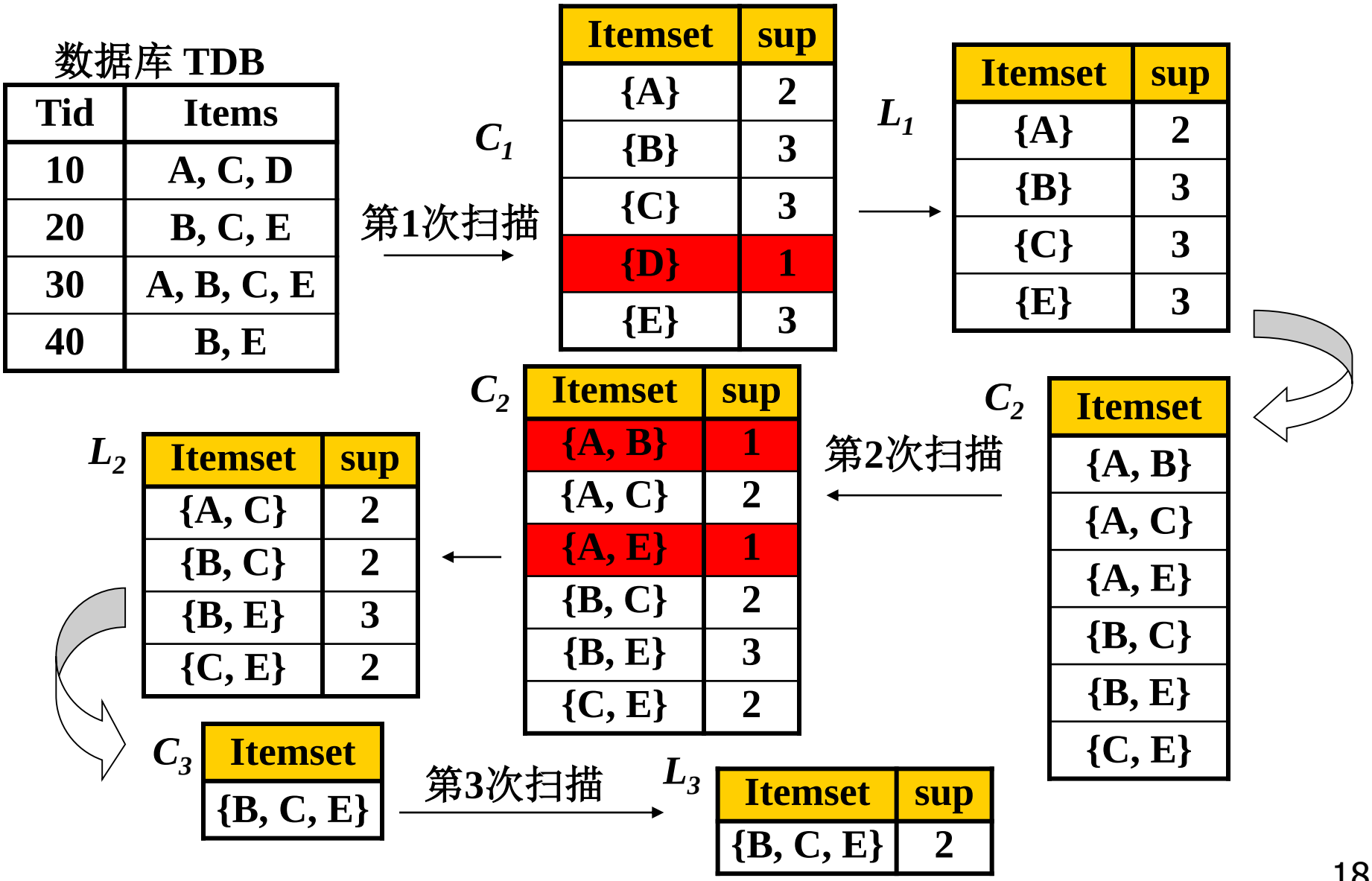
# 关联规则的性质

- **性质1：频繁项集的子集必为频繁项集。**
- **性质2：非频繁项集的超集一定是非频繁的。**
- **Apriori算法运用性质1，通过已知的频繁项集构成长度更大的项集，并将其称为潜在频繁项集。**
  - **潜在频繁$k$项集的集合$C_k$是指由有可能成为频繁$k$项集的项集组成的集合。**
- **以后只需计算潜在频繁项集的支持度，而不必计算所有不同项集的支持度，因此在一定程度上减少了计算量。**

# Apriori: 一种候选产生-测试方法

- 频繁项集的任何子集必须是频繁的
  - 如果 {beer, diaper, nuts} 是频繁的, {beer, diaper}也是
  - 每个包含 {beer, diaper, nuts}的事务 也包含 {beer, diaper}
- Apriori 剪枝原则:
  - 如果一个项集不是频繁的, 将不产生/测试它的超集!
- 方法:
  - 由长度为k的频繁项集产生长度为 (k+1) 的候选项集, 并且
  - 根据 DB测试这些候选
- 性能研究表明了它的有效性和可伸缩性

# Apriori 算法 — 一个例子



# Apriori算法

- (1) $L_1$={频繁1项集};
- (2) for($k$=2;$L_{k-1}\neq\varnothing$;$k$++) do begin
- (3) $C_k$=apriori_gen($L_{k-1}$); //新的潜在频繁项集
- (4) for all *transactions* $t\in D$ do begin
- (5) $C_t$=subset($C_k$,$t$); //找出t中包含的潜在的频繁项
- (6) for all *candidates* $c\in C_t$ do
- (7) *c.count*++;
- (8) end;
- (9) $L_k=\{c\in C_k | c.count\geq minsup\}$
- (10) end;
- (11) Answer= $\bigcup_k L_k$

# Apriori的重要细节

- 如何产生候选?
  - 步骤 1: $L_k$的自连接
  - 步骤 2: 剪枝
- 候选产生的例子
  - $L_3$={*abc, abd, acd, ace, bcd*}
  - 自连接: $L_3$*$L_3$
    - *Abcd*：由 *abc* 和 *abd*
    - *Acde*：由 *acd* 和 *ace*
  - 剪枝:
    - *acde* 被删除, 因为 *ade* 不在 $L_3$
  - $C_4$={*abcd*}

# 如何产生候选?

- 假定 $L_{k-1}$ 中的项集已排序(按字典序排序)
- 步骤 1: $L_{k-1}$自连接

```
procedure apriori_gen(L_{k-1}:frequent (k-1)-itemsets)
(1)   for each itemset l_1 ∈ L_{k-1}
(2)     for each itemset l_2 ∈ L_{k-1}
(3)       if (l_1[1] = l_2[1]) ∧ (l_1[2] = l_2[2]) ∧ ... ∧ (l_1[k-2] = l_2[k-2]) ∧ (l_1[k-1] < l_2[k-1]) then {
(4)         c = l_1 ⋈ l_2; // join step: generate candidates
(5)         if has_infrequent_subset(c, L_{k-1}) then
(6)           delete c; // prune step: remove unfruitful candidate
(7)         else add c to C_k;
(8)       }
(9)   return C_k;
```

- **Step 2: 剪枝**

```
procedure has_infrequent_subset(c: candidate k-itemset;
            L_{k-1}: frequent (k-1)-itemsets); // use prior knowledge
(1)   for each (k-1)-subset s of c
(2)     if s ∉ L_{k-1} then
(3)       return TRUE;
(4)   return FALSE;
```

# 例子-支持计数=2

AllElectronics 数据库

| TID | List of item_ID's |
|---|---|
| T100 | I1,I2,I5 |
| T200 | I2,I4 |
| T300 | I2,I3 |
| T400 | I1,I2,I4 |
| T500 | I1,I3 |
| T600 | I2,I3 |
| T700 | I1,I3 |
| T800 | I1,I2,I3,I5 |
| T900 | I1,I2,I3 |

$C_1$

| 项集 | 支持度计数 |
|---|---|
| {I1} | 6 |
| {I2} | 7 |
| {I3} | 6 |
| {I4} | 2 |
| {I5} | 2 |

比较候选支持度计数与最小支持度计数 →

$L_1$

| 项集 | 支持度计数 |
|---|---|
| {I1} | 6 |
| {I2} | 7 |
| {I3} | 6 |
| {I4} | 2 |
| {I5} | 2 |

由$L_1$产生候选$C_2$ →

$C_2$

| 项集 |
|---|
| {I1,I2} |
| {I1,I3} |
| {I1,I4} |
| {I1,I5} |
| {I2,I3} |
| {I2,I4} |
| {I2,I5} |
| {I3,I4} |
| {I3,I5} |
| {I4,I5} |

扫描D，对每个候选计数 →

$C_2$

| 项集 | 支持度计数 |
|---|---|
| {I1,I2} | 4 |
| {I1,I3} | 4 |
| {I1,I4} | 1 |
| {I1,I5} | 2 |
| {I2,I3} | 4 |
| {I2,I4} | 2 |
| {I2,I5} | 2 |
| {I3,I4} | 0 |
| {I3,I5} | 1 |
| {I4,I5} | 0 |

比较候选支持度计数与最小支持度计数

$L_2$

| 项集 | 支持度计数 |
|---|---|
| {I1,I2} | 4 |
| {I1,I3} | 4 |
| {I1,I5} | 2 |
| {I2,I3} | 4 |
| {I2,I4} | 2 |
| {I2,I5} | 2 |

比较候选支持度计数
与最小支持度计数

$L_2$

| 项集 | 支持度计数 |
|---|---|
| {I1,I2} | 4 |
| {I1,I3} | 4 |
| {I1,I5} | 2 |
| {I2,I3} | 4 |
| {I2,I4} | 2 |
| {I2,I5} | 2 |

由$L_2$产生候选$C_3$ →

$C_3$

| 项集 |
|---|
| {I1,I2,I3} |
| {I1,I2,I5} |

扫描D，对每个候选计数 →

$C_3$

| 项集 | 支持度计数 |
|---|---|
| {I1,I2,I3} | 2 |
| {I1,I2,I5} | 2 |

比较候选支持度计数
与最小支持度计数 →

$L_3$

| 项集 | 支持度计数 |
|---|---|
| {I1,I2,I3} | 2 |
| {I1,I2,I5} | 2 |

1. 连接：$C_3 = L_2 \quad L_2$={{I1,I2},{I1,I3},{I1,I5},{I2,I3},{I2,I4},{I2,I5}}
{{I1,I2},{I1,I3},{I1,I5},{I2,I3},{I2,I4},{I2,I5}} =
{{I1,I2,I3},{I1,I2,I5},{I1,I3,I5},{I2,I3,I4},{I2,I3,I5},{I2,I4,I5}}

2. 使用 Apriori 性质剪枝：频繁项集的所有子集必须是频繁的。存在候选项集，其子集不是频繁的吗？

- {I1,I2,I3}的 2-项子集是{I1,I2}，{I1,I3}和{I2,I3}。{I1,I2,I3}的所有 2-项子集都是 $L_2$ 的元素。因此，保留{I1,I2,I3}在 $C_3$ 中。
- {I1,I2,I5}的 2-项子集是{I1,I2}，{I1,I5}和{I2,I5}。{I1,I2,I5}的所有 2-项子集都是 $L_2$ 的元素。因此，保留{I1,I2,I5}在 $C_3$ 中。
- {I1,I3,I5}的 2-项子集是{I1,I3}，{I1,I5}和{I3,I5}。{I3,I5}不是 $L_2$ 的元素，因而不是频繁的。这样，由 $C_3$ 中删除{I1,I3,I5}。

# 由频繁项集产生关联规则

- 根据公式产生关联规则

$$confidence(A \Rightarrow B) = P(B|A) = \frac{support\_count(A \cup B)}{support\_count(A)}$$

- 对于每个频繁项集l，产生所有的非空子集
- 对于l的每个非空子集s，如果 $\frac{support\_count(l)}{support\_count(s)} \geq min\_conf$, 则输出规则”s➔(l-s)”

# 频繁模式挖掘的挑战

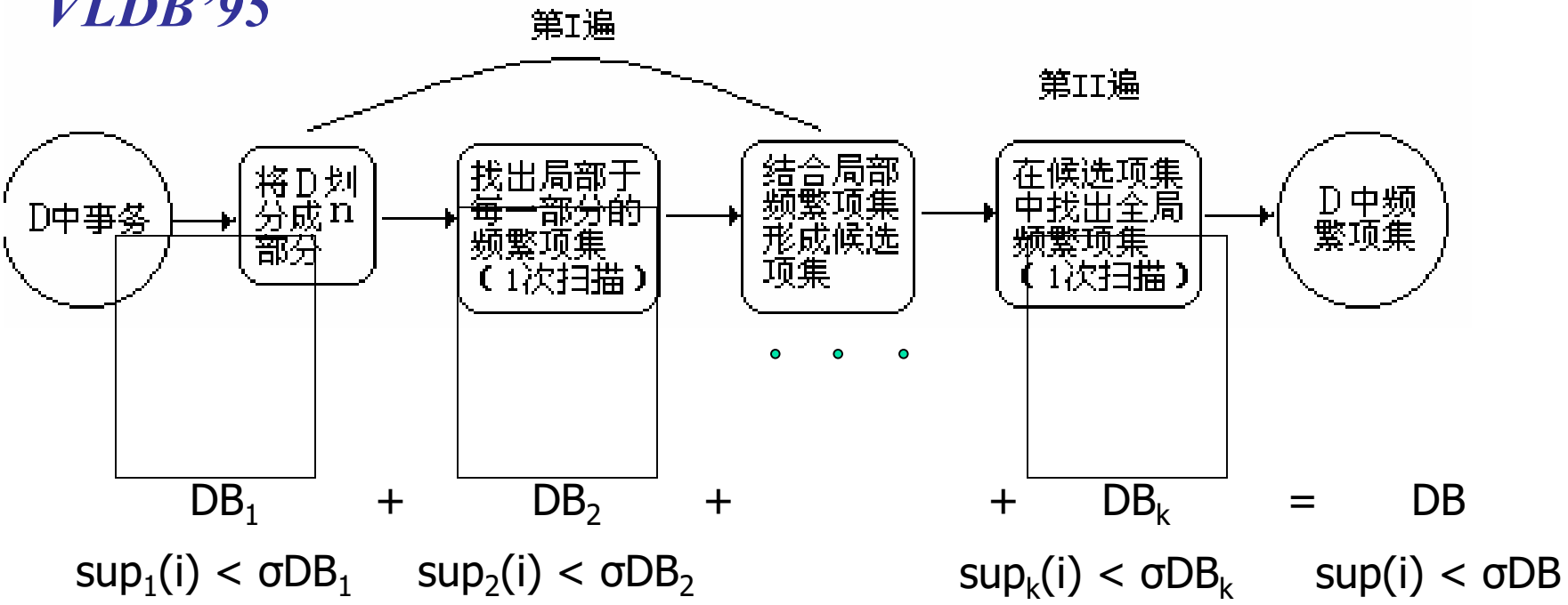
- 挑战
  - 事务数据库的多遍扫描
  - 数量巨大的候选
  - 候选支持度计数繁重的工作量
- 改进 **Apriori:** 基本思想
  - 减少事务数据库的扫描遍数
  - 压缩候选数量
  - 便于候选计数

# 提高Apriori算法的方法

- **Hash-based itemset counting（散列项集计数）**
- **Transaction reduction（事务压缩）**
- **Partitioning（划分）**
- **Sampling（采样）**

# 划分: 只扫描数据库两次

- 项集在DB中是频繁的, 它必须至少在DB的一个划分中是频繁的
  - 扫描 1: 划分数据库, 并找出局部频繁模式 local frequent itemset
  - 扫描 2: 求出全局频繁模式
- A. Savasere, E. Omiecinski, and S. Navathe. An efficient algorithm for mining association in large databases. In *VLDB'95*



# 抽样-频繁模式

- 选取原数据库的一个样本, 使用Apriori 算法在样本中挖掘频繁模式
- 扫描一次数据库, 验证在样本中发现的频繁模式.

- 再次扫描数据库, 找出遗漏的频繁模式

- 牺牲一些精度换取有效性。
- H. Toivonen. Sampling large databases for association rules. In *VLDB'96*

# DHP: 压缩候选的数量

- 散列项集到对应的桶中，一个其hash桶的计数小于阈值的*k*-itemset 不可能是频繁的
- J. Park, M. Chen, and P. Yu. An effective hash-based algorithm for mining association rules. In *SIGMOD'95*

使用散列函数

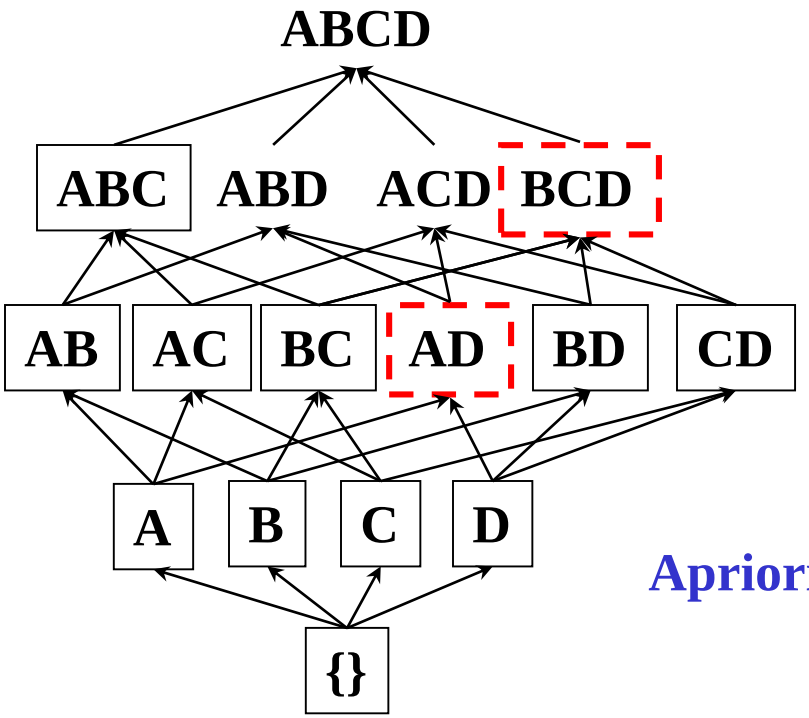
hash(x, y)=((order of x) *10+(order of y)) mod 7

创建散列表 $H_2$

$H_2$

| 桶地址 | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| 桶计数 | 2 | 2 | 4 | 2 | 2 | 4 | 4 |
| 桶内容 | {I1,I4}<br>{I3,I5} | {I1,I5}<br>{I1,I5} | {I2,I3}<br>{I2,I3}<br>{I2,I3}<br>{I2,I3} | {I2,I4}<br>{I2,I4} | {I2,I5}<br>{I2,I5} | {I1,I2}<br>{I1,I2}<br>{I1,I2}<br>{I1,I2} | {I1,I3}<br>{I1,I3}<br>{I1,I3}<br>{I1,I3} |

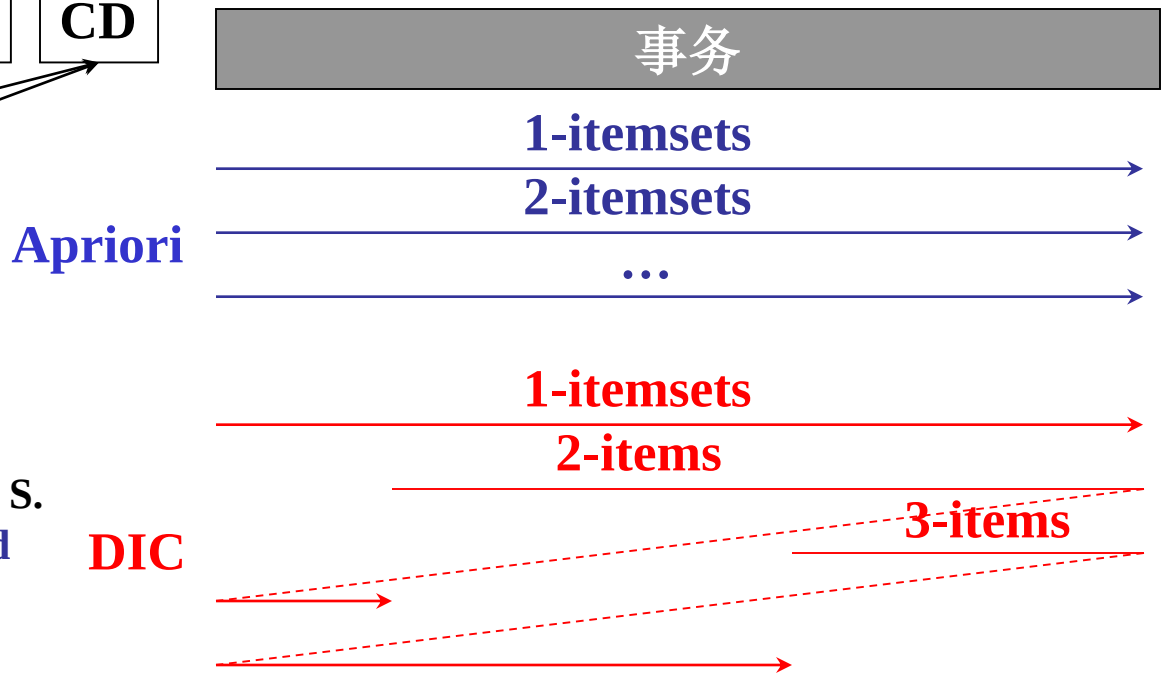
# DIC(Dynamic itemset counting)：减少扫描次数




Itemset lattice

S. Brin R. Motwani, J. Ullman, and S. Tsur. Dynamic itemset counting and implication rules for market basket data. In *SIGMOD'97*

- 一旦确定 A 和 D 是频繁的, 立即开始 AD 的计数
- 一旦确定 BCD 的两个长度为2的子集是频繁的, 立即开始BCD 的计数



# 使用垂直数据格式挖掘频繁项集Vertical Data Format

- 使用tid-list, 包含item的事务的标识的集合;
  - M. Zaki et al. New algorithms for fast discovery of association rules. In KDD'97
- 扫描一次数据集将水平格式数据转化为垂直格式
- 通过频繁k项集的tid-list的交集，计算对应(k+1)项集的tid-list

| *itemset* | *TID_set* |
|---|---|
| I1 | {T100, T400, T500, T700, T800, T900} |
| I2 | {T100, T200, T300, T400, T600, T800, T900} |
| I3 | {T300, T500, T600, T700, T800, T900} |
| I4 | {T200, T400} |
| I5 | {T100, T800} |

| *itemset* | *TID_set* |
|---|---|
| {I1, I2, I3} | {T800, T900} |
| {I1, I2, I5} | {T100, T800} |

The 2-itemsets in vertical data format.

| *itemset* | *TID_set* |
|---|---|
| {I1, I2} | {T100, T400, T800, T900} |
| {I1, I3} | {T500, T700, T800, T900} |
| {I1, I4} | {T400} |
| {I1, I5} | {T100, T800} |
| {I2, I3} | {T300, T600, T800, T900} |
| {I2, I4} | {T200, T400} |
| {I2, I5} | {T100, T800} |
| {I3, I5} | {T800} |

# 频繁模式挖掘的瓶颈

- 多遍数据库扫描是 昂贵的
- 挖掘长模式需要很多遍扫描, 并产生大量候选
  - 挖掘频繁模式 $i_1 i_2 \ldots i_{100}$
    - 扫描次数: 100
    - 候选个数: $\binom{100}{1} + \binom{100}{2} + \ldots + \binom{100}{100} = 2^{100}-1 = 1.27*10^{30}$ !
- 瓶颈: 候选产生-测试
- 能够避免候选产生吗?

# 挖掘频繁模式而不产生候选

- 使用局部频繁的项, 由短模式增长产生长模式
  - **“abc” 是频繁模式**
  - **得到包含 “abc”的所有事务: DB|abc**
  - **“d” 是 DB|abc 中的局部频繁项→ abcd 是频繁模式**

# 由事务数据库构造FP-树

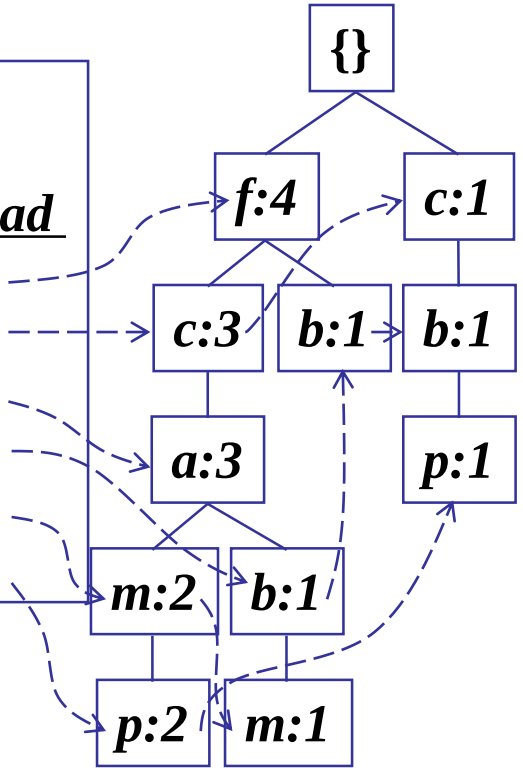
| *TID* | *Items bought* | *(ordered) frequent items* |
|---|---|---|
| 100 | {*f, a, c, d, g, i, m, p*} | {*f, c, a, m, p*} |
| 200 | {*a, b, c, f, l, m, o*} | {*f, c, a, b, m*} |
| 300 | {*b, f, h, j, o, w*} | {*f, b*} |
| 400 | {*b, c, k, s, p*} | {*c, b, p*} |
| 500 | {*a, f, c, e, l, p, m, n*} | {*f, c, a, m, p*} |

***min_support = 3***

1. 扫描 DB 一次, 找出频繁 1-itemset (单个项的模式)
2. 按频率的降序将频繁项排序, 得到 f-list
3. 再次扫描 DB, 构造 FP-树

**Header Table**

| *Item* | *frequency* | *head* |
|---|---|---|
| *f* | *4* | |
| *c* | *4* | |
| *a* | *3* | |
| *b* | *3* | |
| *m* | *3* | |
| *p* | *3* | |



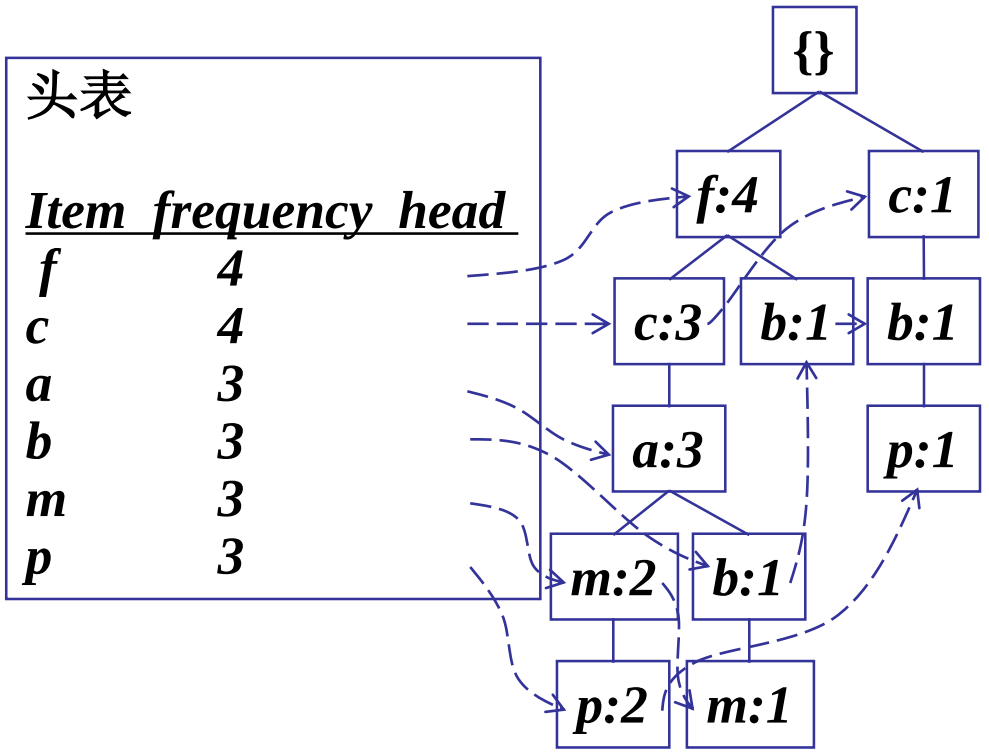

**F-list=f-c-a-b-m-p**

# 划分模式和数据库

- 可以按照f-list 将频繁模式划分成子集
  - F-list=f-c-a-b-m-p
  - 包含 p的模式
  - 包含 m 但不包含 p的模式
  - …
  - 包含 c 但不包含 a, b, m, p
  - 模式 f
- 完全性和非冗余性

# 从p-条件数据库找出含p的模式

- 从FP-树的频繁项头表开始
- 沿着频繁项*p* 的链搜索FP-树
- 收集项 *p* 的所有*变换的前缀路径 形成p* 的模式基




*条件* 模式基

| *item* | *cond. pattern base* |
|---|---|
| *c* | *f:3* |
| *a* | *fc:3* |
| *b* | *fca:1, f:1, c:1* |
| *m* | *fca:2, fcab:1* |
| *p* | *fcam:2, cb:1* |

# 通过建立条件模式库得到频繁集

| 项 | 条件模式库 | 条件FP-tree |
|---|---|---|
| p | {(fcam:2), (cb:1)} | {(c:3)}\|p |
| m | {(fca:2), (fcab:1)} | {(f:3, c:3, a:3)}\|m |
| b | {(fca:1), (f:1), (c:1)} | Empty |
| a | {(fc:3)} | {(f:3, c:3)}\|a |
| c | {(f:3)} | {(f:3)}\|c |
| f | Empty | Empty |

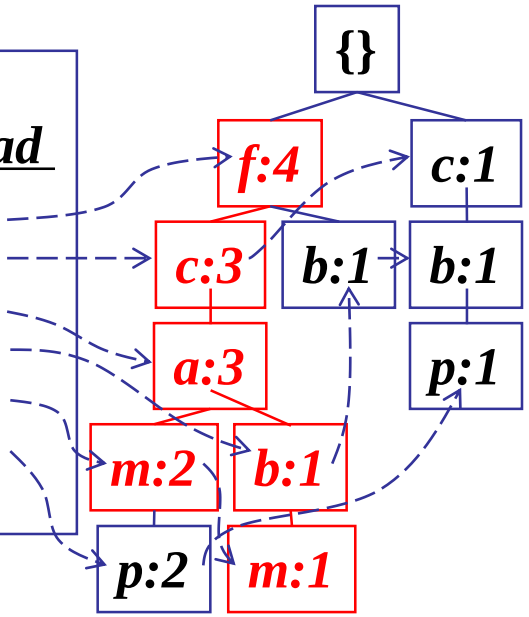
# 从条件模式基到条件FP-树

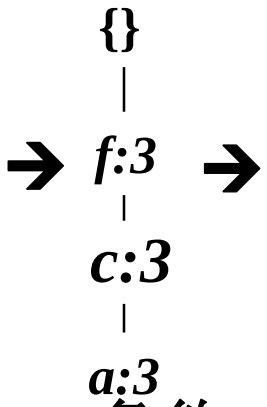
- 对于每个条件模式基
  - 累计条件模式基中每个项的计数
  - 构造模式基中频繁项的FP-树

*m-条件* 模式基：
*fca:2, fcab:1*

头表

| *Item* | *frequency* | *head* |
|---|---|---|
| *f* | *4* | |
| *c* | *4* | |
| *a* | *3* | |
| *b* | *3* | |
| *m* | *3* | |
| *p* | *3* | |



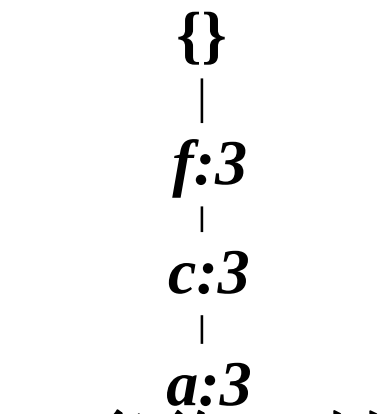




*m-条件* FP-树

*m*的所有频繁模式

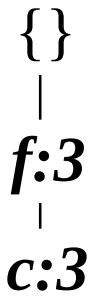
*m,*
*fm, cm, am,*
*fcm, fam, cam,*
*fcam*

# 递归: 挖掘每个条件 FP-树



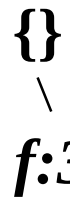

*m-条件* FP-树

“am”的条件模式基: (fc:3)



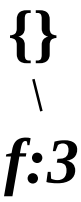

*am-条件* FP-树

“cm”的条件模式基: (f:3)




*cm-条件* FP-树

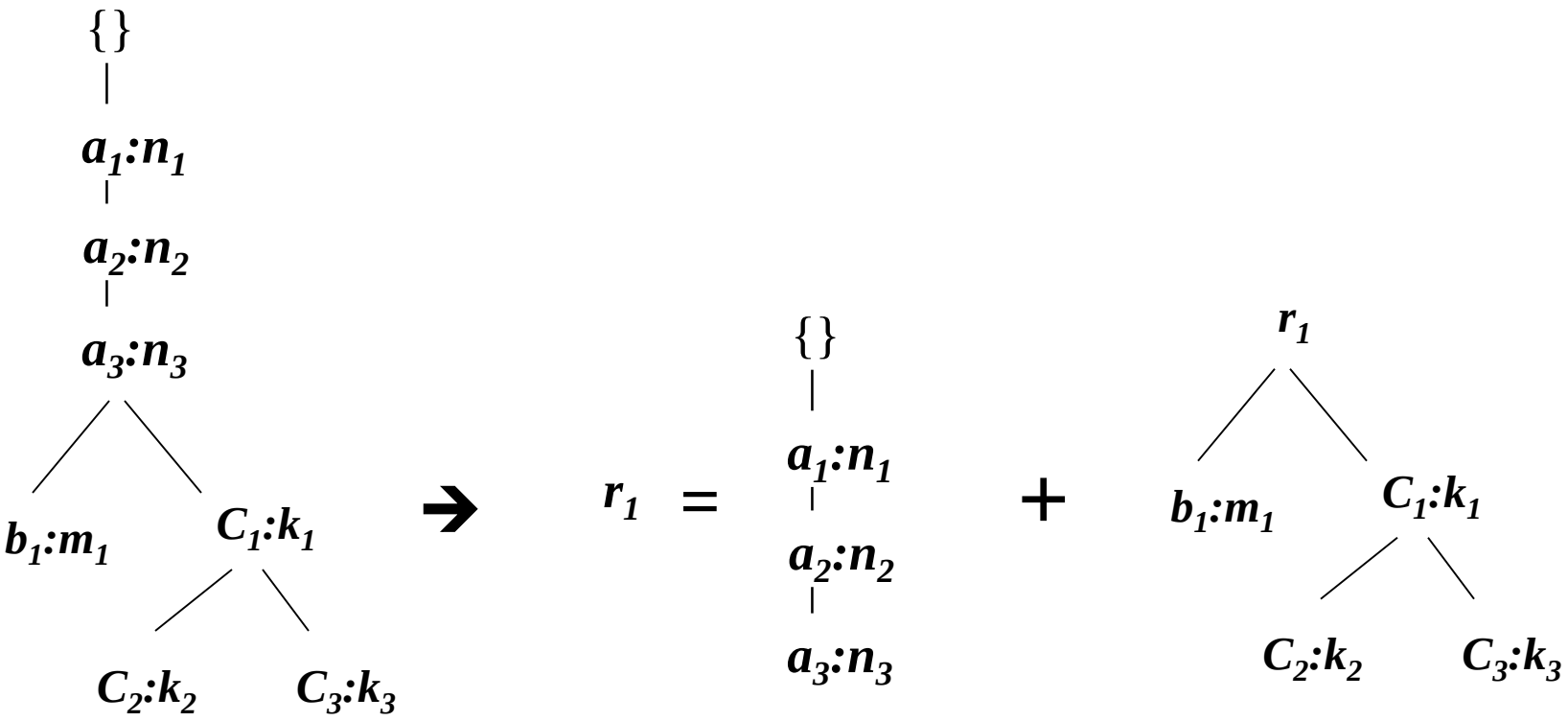
“cam”的条件模式基: (f:3)




*cam-条件* FP-树

# 特殊情况: FP-树中的单个前缀路径

- 假定 (条件) FP-树 T 具有单个共享的前缀路径P
- 挖掘可以分解成两步
  - 将单个前缀路径归约成 一个结点
  - 连接两部分的挖掘结果



# 使用FP-树挖掘频繁模式

- 基本思想: 频繁模式增长
  - 通过模式和数据库划分递归地增长频繁模式
- 方法
  - (1)对于每个频繁项, 构造它的条件模式基
  - (2)然后构造它的条件 FP-树
  - (3)在新构造的条件FP-树上重复这一过程
    - 直到结果条件 FP-树为空，或者它只包含一条路径—单个路径将产生其子路径的所有组合，每个子路径是一个频繁模式

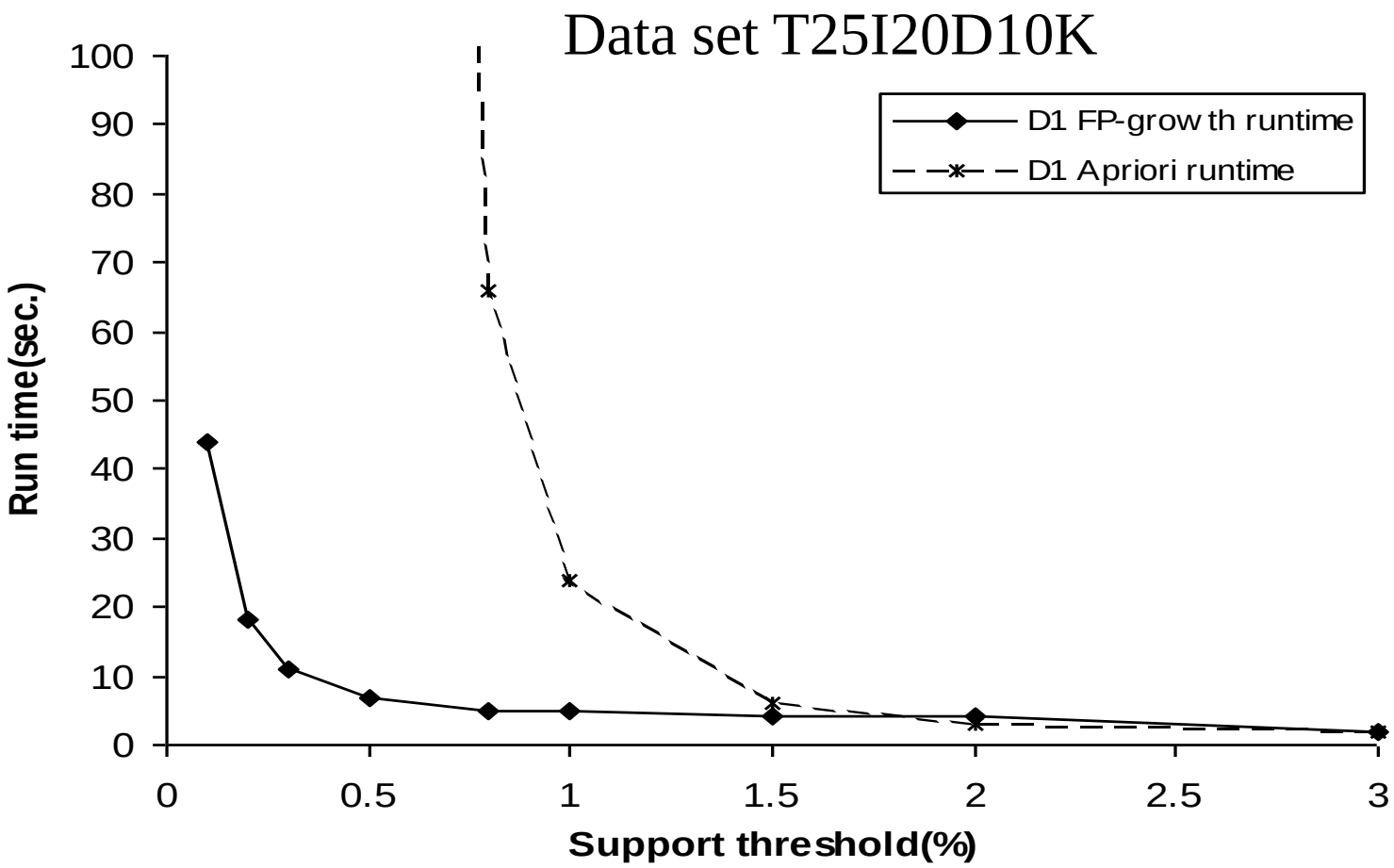
# FP-树结构的优点

- 完全性
  - 保留频繁模式挖掘的完整信息
  - 不截断任何事务的长模式
- 压缩性
  - 压缩无关信息—非频繁的项被删除
  - 项按频率的降序排列: 越是频繁出现, 越可能被共享
  - 绝对不比原来的数据库大 (不计结点链和计数字段)

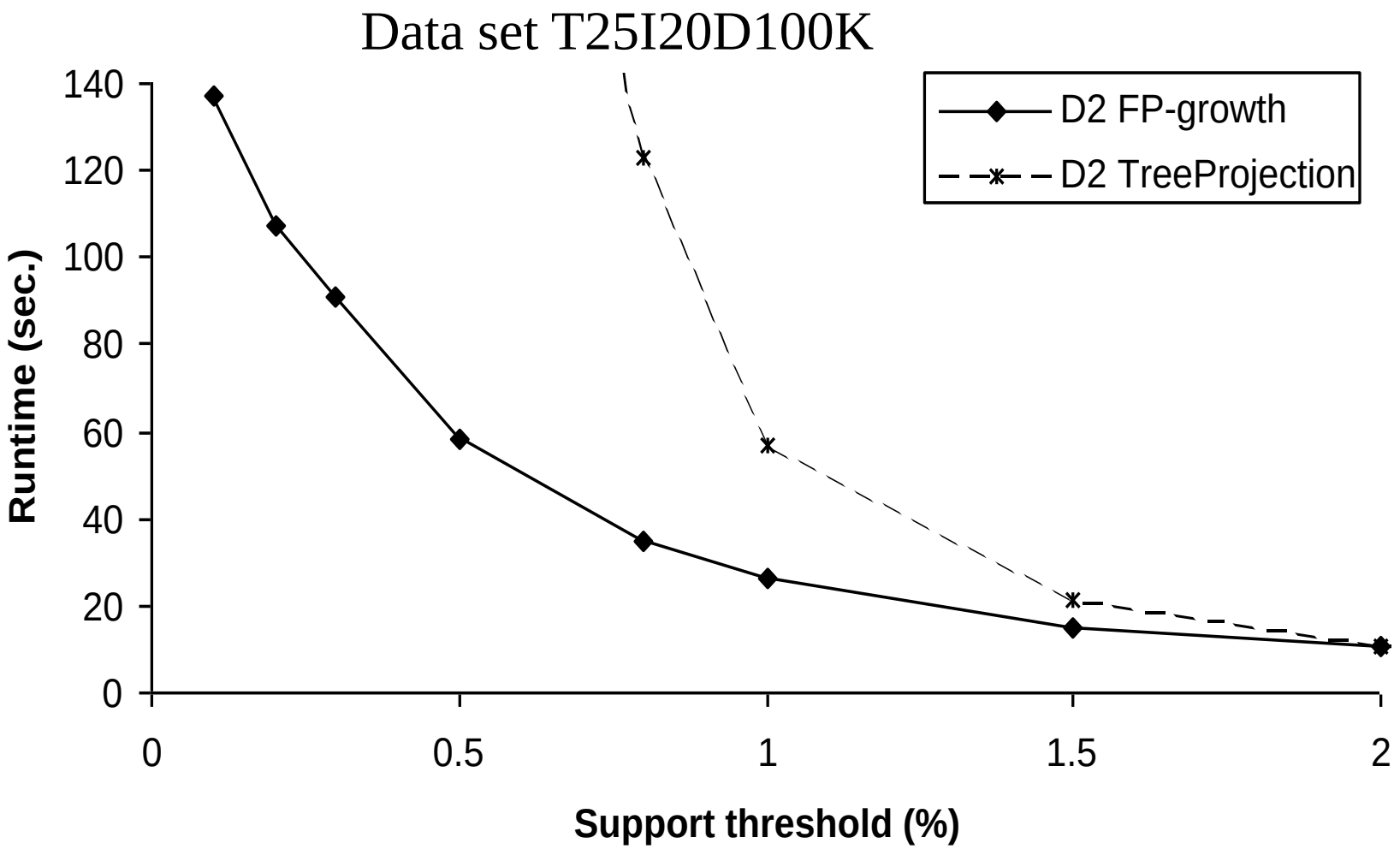
# FP-增长的规模化

- FP-树不能放在内存, 怎么办?—数据库投影
- 数据库投影
  - 首先将数据库划分成一组投影 数据库
  - 然后对每个投影数据库构造并挖掘FP-树

# FP-增长 vs. Apriori: 随支持度增长的可伸缩性



# FP-增长 vs. 树-投影:随支持度增长的可伸缩性



# 为什么FP-增长是赢家?

- 分治:
  - 根据已经得到的频繁模式划分任务和数据库
  - 导致较小的数据库的聚焦的搜索
- 其它因素
  - 没有候选产生, 没有候选测试
  - 压缩数据库：FP-树结构
  - 不重复地扫描整个数据库
  - 基本操作—局部频繁项计数和建立子FP-树, 没有模式搜索和匹配

# 有关的其他方法

- 挖掘频繁闭项集合和最大模式
  - **CLOSET (DMKD'00)**
- 挖掘序列模式
  - **FreeSpan (KDD'00), PrefixSpan (ICDE'01)**
- 频繁模式的基于限制的挖掘
  - **Convertible constraints (KDD'00, ICDE'01)**
- 计算具有复杂度量的冰山数据方
  - **H-tree and H-cubing algorithm (SIGMOD'01)**

# 最大模式

- 频繁模式 $\{a_1, \ldots, a_{100}\}$ 包含 $\binom{100}{1} + \binom{100}{2} + \ldots + \binom{1}{1}\binom{0}{0}\binom{0}{0} = 2^{100}-1 = 1.27*10^{30}$ 频繁子模式!
- 最大模式: 频繁模式, 其真超模式都不是频繁的
  - BCDE, ACD 是最大模式
  - BCD 不是最大模式

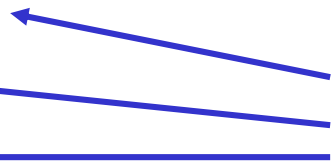
Min_sup=2

| Tid | Items |
|---|---|
| 10 | A,B,C,D,E |
| 20 | B,C,D,E, |
| 30 | A,C,D,F |

# MaxMiner: 挖掘最大模式

| Tid | Items |
|---|---|
| 10 | *A,B,C,D,E* |
| 20 | *B,C,D,E,* |
| 30 | *A,C,D,F* |

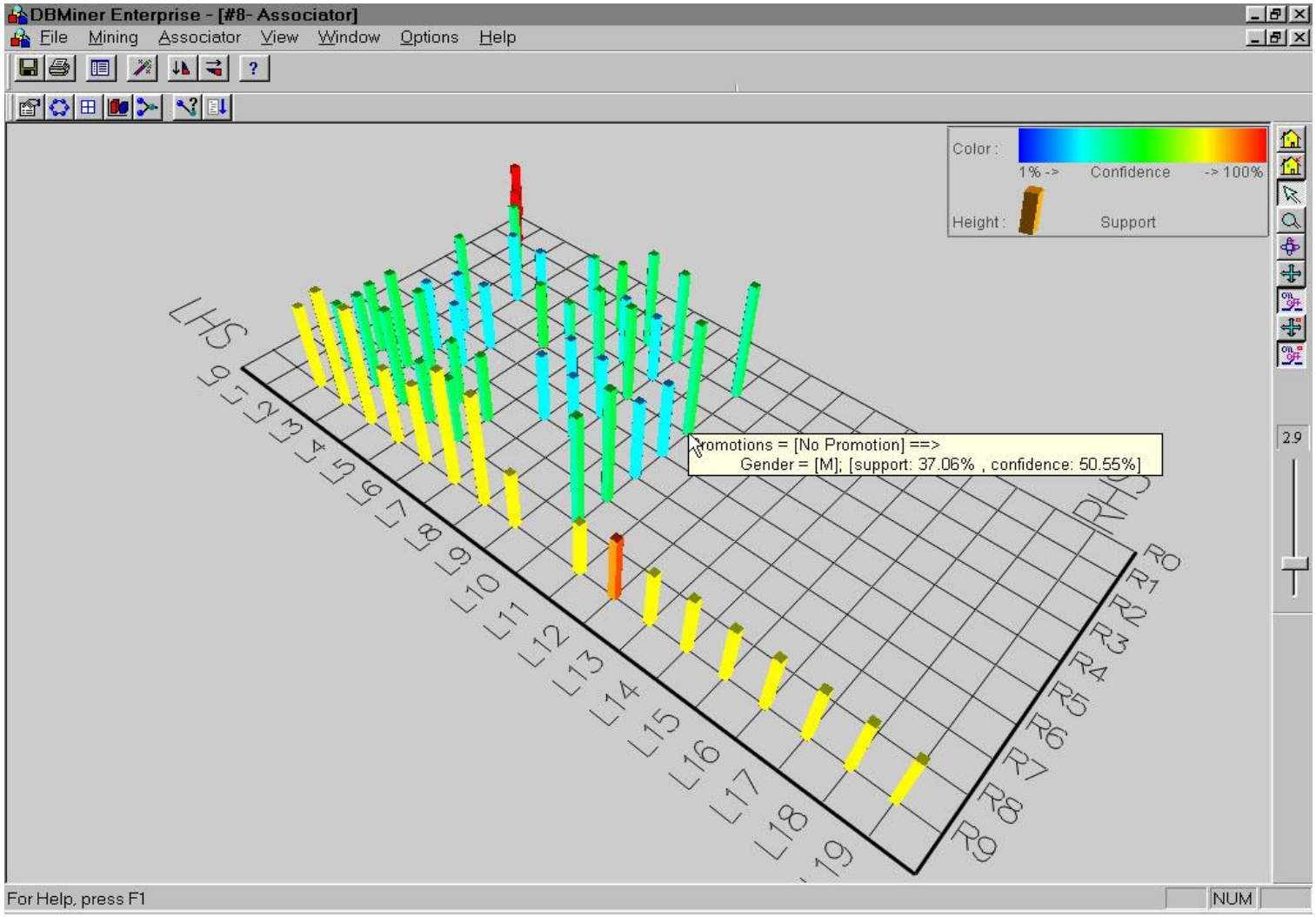
- 扫描1: 找出频繁项
  - *A, B, C, D, E*
- 扫描2: 找出以下项集的支持度
  - *AB, AC, AD, AE, ABCDE*
  - *BC, BD, BE, BCDE*
  - *CD, CE, CDE, ~~DE~~*



潜在的最大模式

- 由于 *BCDE* 是最大模式,
  不必在此后的扫描时检查 *BCD, BDE, CDE*
- **R. Bayato. Efficiently mining long patterns from databases. In *SIGMOD'98***

# 关联规则的可视化: Pane Graph



# 第5章：挖掘关联规则

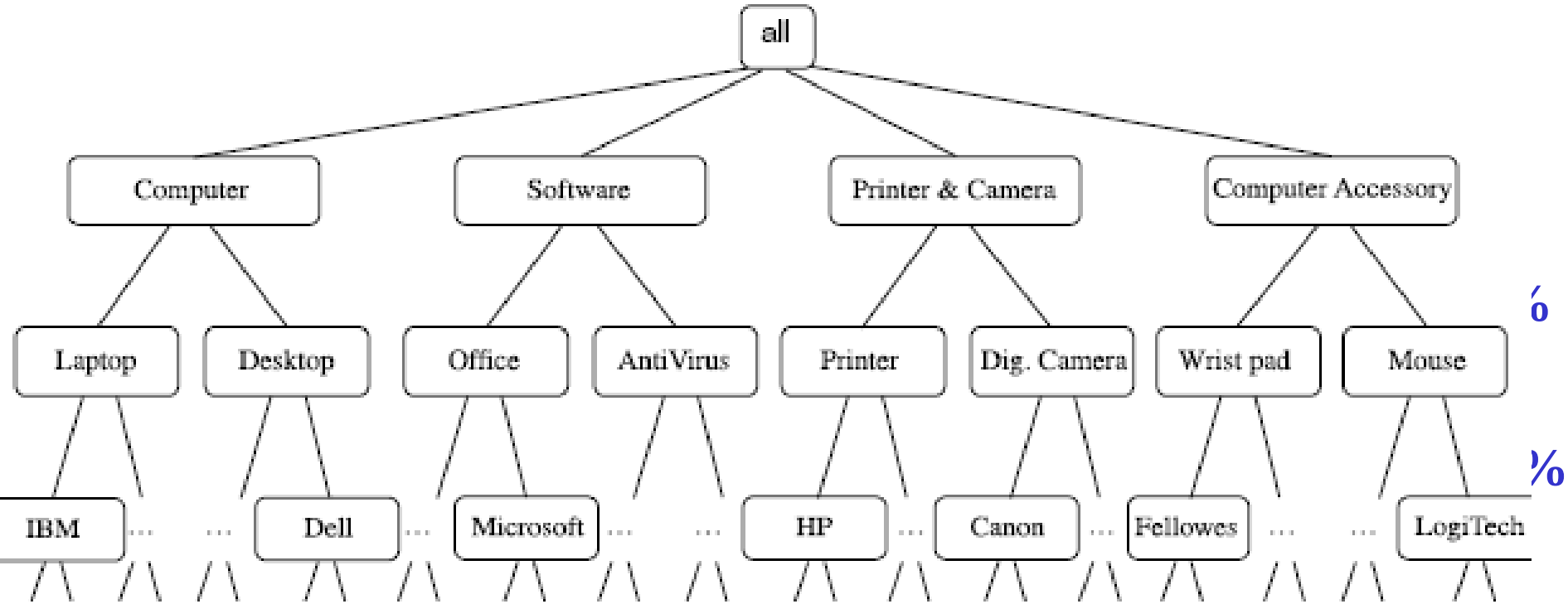
- 关联规则挖掘
- 事务数据库中(单维布尔)关联规则挖掘的可伸缩算法
- 挖掘各种关联/相关规则
- 基于限制的关联挖掘
- 顺序模式挖掘
- 小结

# 挖掘各种规则或规律性

- 多层关联规则,
- 多维关联规则,
- 量化关联规则,
- 相关性和因果关系, 比率规则, 序列模式, 显露模式, 时间关联, 局部周期性

# 多层关联规则

- 项常常形成层次结构-概念分层
- 多个抽象层次上挖据得到的关联规则-多层关联规则
- 灵活的支持度设定: 较低层中的项一般具有较低的支持度.



# 多层关联: 冗余过滤

- 由于项之间的 “祖先” 联系, 有些规则可能是多余的.
- 例
  - milk ⇒ wheat bread [support = 8%, confidence = 70%]
  - 2% milk ⇒ wheat bread [support = 2%, confidence = 72%]
  - 其中2% milk 占milk的1/4
- 我们可以说第一个规则是第二个规则的祖先.
- 一个规则是冗余的, 如果根据规则的祖先, 其支持度和置信度都接近于 “期望” 值.

# 多层挖掘: 逐步深入

- 一种自顶向下, 逐步深入的方法:
  - 首先挖掘最高层的频繁模式:
    **milk (15%), bread (10%)**
  - 然后挖掘它们下层 “较弱的” 频繁模式:
    **2% milk (5%), wheat bread (4%)**
- 多层之间的不同的最小支持度阈值导致不同的算法 :
  - 如果不同层之间采用相同的***min_support***
    则丢弃 ***t*** 如果 ***t'***的任意祖先是非频繁的.
  - 如果在较低层采用递减的 ***min_support***
    则只考察其祖先为频繁的项集.

# 多维关联规则

- 单维规则:包括单个谓词（可以多次出现）或单个维

  buys(X, “milk”) ⇒ buys(X, “bread”)

- 多维规则: 维或谓词 ≥ 2
  - 维间关联规则 (*不含重复谓词*)

    age(X,”19-25”) ∧ occupation(X,“student”) ⇒ buys(X,“coke”)
  - 混合维关联规则 (*含重复谓词*)

    age(X,”19-25”) ∧ buys(X, “popcorn”) ⇒ buys(X, “coke”)
- 数据的属性可分为两类
  - 分类属性
    - 有限个不同值, 值之间无序
  - 量化属性
    - 数值的, 值之间隐含次序

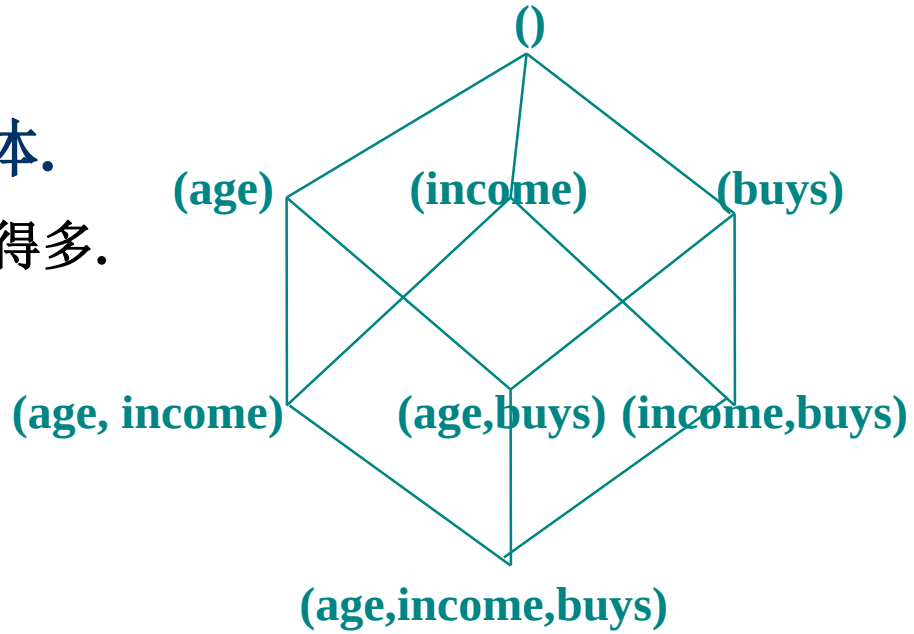
# 挖掘多维关联规则的技术

- 搜索频繁 *k*-谓词集 :包含k个合取谓词的集合
  - 例: {age, occupation, buys} 是一个 3-谓词集.
  - 可以按如何处理 age 对技术分类.
1. 使用量化属性的静态离散化
   - 使用预先定义的概念分层, 对量化属性静态地离散化.
2. 量化关联规则
   - 根据数据的分布, 将量化属性离散化到 “箱” .
3. 基于距离的关联规则
   - 是一种动态的离散化过程, 它考虑数据点之间的距离.

# 量化属性的静态离散化

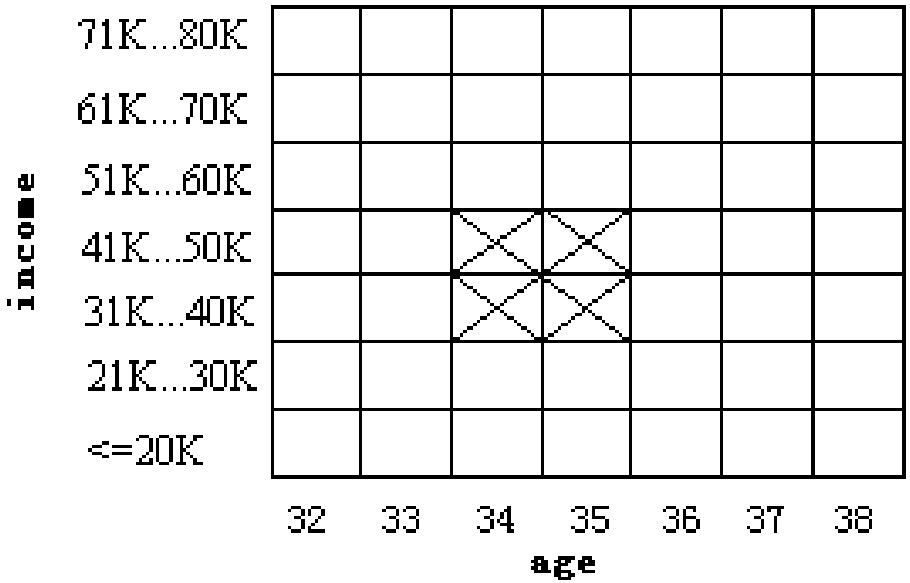
- 使用概念分层, 在挖掘之前离散化.
  - 数值用区间值替换.
- 在关系数据库中, 找出所有的频繁 k-谓词集需要$k$ 或 $k$+1 次表扫描.
- 数据立方体非常适合挖掘.
  - n-维方体

    对应于谓词集合的方体.
  - 从数据立方体挖掘可以快得多.



# 量化关联规则

- 数值属性动态地离散化
  - 使挖出的规则的置信度或紧凑性最大化.
- 2-维量化关联规则: $A_{quan1} \wedge A_{quan2} \Rightarrow A_{cat}$（分类属性）
- ARCS方法：使用2-D栅格,
  - 1)对属性进行（等宽）分箱
  - 2)找频繁谓词集
  - 3)规则聚类：对“相邻的” 关联
  聚类形成一般关联规则.
- 例:
  age(X,”34-35”) ∧
  income(X,”31K - 50K”)
  ⇒ buys(X,”high resolution TV



# 挖掘基于距离的关联规则

- 分箱方法不能紧扣区间数据的语义
- 基于距离的划分, 更有意义的离散化考虑：
  - 区间内点的密度/数量
  - 区间内点的“紧密性”

| Price($) | Equi-width (width $10) | Equi-depth (depth 2) | Distance-based |
|---|---|---|---|
| 7 | [0,10] | [7,20] | [7,7] |
| 20 | [11,20] | [22,50] | [20,22] |
| 22 | [21,30] | [51,53] | [50,53] |
| 50 | [31,40] | | |
| 51 | [41,50] | | |
| 53 | [51,60] | | |

# 具有灵活的支持度限制的多层ML/MD多维关联规则

- 为什么?
  - 现实中项的出现频率差异很大
    - 购物中的钻石, 表, 笔
  - 一致的支持度可能不是一种好的模型
- 灵活的模型
  - 通常, 层越低, 维的组合越多, 长模式越长, 支持度越小
  - 一般规则应当是特指的, 易于理解的
  - 特殊的项或特殊的项群可能被个别地指定, 并具有较高的优先权

# 兴趣度度量: 相关性(Lift)

- ***play basketball*** $\Rightarrow$ ***eat cereal*** **[40%, 66.7%]** 是误导
  - 吃谷类食品的学生所占的百分比为75%, 比 66.7%还高.
- ***play basketball*** $\Rightarrow$ ***not eat cereal*** **[20%, 33.3%]** 更准确, 其支持度和置信度都较低
- 依赖/相关事件的度量:

$$lift = \frac{P(A \cup B)}{P(A)P(B)}$$

| | Basketball | Not basketball | Sum (row) |
|---|---|---|---|
| Cereal谷类 | 2000 | 1750 | 3750 |
| Not cereal | 1000 | 250 | 1250 |
| Sum(col.) | 3000 | 2000 | 5000 |

$$lift(B,C) = \frac{2000/5000}{3000/5000 * 3750/5000} = 0.89$$

$$lift(B,\neg C) = \frac{1000/5000}{3000/5000 * 1250/5000} = 1.33$$

$$all\_conf = \frac{\sup(X)}{\max\_item\_\sup(X)}$$

# Which Measures Should Be Used?

- 提升度和$\chi^2$不是好的相关度量，对于大的交易数据库
- all-conf **or** coherence **could be good measures (Omiecinski@TKDE′03)**
- **Over 20 interestingness measures have been proposed (see Tan, Kumar, Sritastava @KDD′02)**
- **Which are good ones?**

| symbol | measure | range | formula |
|---|---|---|---|
| $\phi$ | $\phi$-coefficient | -1 . . . 1 | $\frac{P(A,B)-P(A)P(B)}{\sqrt{P(A)P(B)(1-P(A))(1-P(B))}}$ |
| $Q$ | Yule's Q | -1 . . . 1 | $\frac{P(A,B)P(\overline{A},\overline{B})-P(A,\overline{B})P(\overline{A},B)}{P(A,B)P(\overline{A},\overline{B})+P(A,\overline{B})P(\overline{A},B)}$ |
| $Y$ | Yule's Y | -1 . . . 1 | $\frac{\sqrt{P(A,B)P(\overline{A},\overline{B})}-\sqrt{P(A,\overline{B})P(\overline{A},B)}}{\sqrt{P(A,B)P(\overline{A},\overline{B})}+\sqrt{P(A,\overline{B})P(\overline{A},B)}}$ |
| $k$ | Cohen's | -1 . . . 1 | $\frac{P(A,B)+P(\overline{A},\overline{B})-P(A)P(B)-P(\overline{A})P(\overline{B})}{1-P(A)P(B)-P(\overline{A})P(\overline{B})}$ |
| $PS$ | Piatetsky-Shapiro's | -0.25 . . . 0.25 | $P(A,B)-P(A)P(B)$ |
| $F$ | Certainty factor | -1 . . . 1 | $\max(\frac{P(B\|A)-P(B)}{1-P(B)}, \frac{P(A\|B)-P(A)}{1-P(A)})$ |
| $AV$ | added value | -0.5 . . . 1 | $\max(P(B\|A)-P(B), P(A\|B)-P(A))$ |
| $K$ | Klosgen's Q | -0.33 . . . 0.38 | $\sqrt{P(A,B)}\max(P(B\|A)-P(B), P(A\|B)-P(A))$ |
| $g$ | Goodman-kruskal's | 0 . . . 1 | $\frac{\sum_j \max_k P(A_j,B_k)+\sum_k \max_j P(A_j,B_k)-\max_j P(A_j)-\max_k P(B_k)}{2-\max_j P(A_j)-max_k P(B_k)}$ |
| $M$ | Mutual Information | 0 . . . 1 | $\frac{\sum_i\sum_j P(A_i,B_j)\log\frac{P(A_i,B_j)}{P(A_i)P(B_J)}}{\min(-\sum_i P(A_i)\log P(A_i)\log P(A_i), -\sum_i P(B_i)\log P(B_i)\log P(B_i))}$ |
| $J$ | J-Measure | 0 . . . 1 | $\max(P(A,B)\log(\frac{P(B\|A)}{P(B)})+P(A\overline{B})\log(\frac{P(\overline{B}\|A)}{P(\overline{B})}))$ |
| | | | $P(A,B)\log(\frac{P(A\|B)}{P(A)})+P(\overline{A}B)\log(\frac{P(\overline{A}\|B)}{P(\overline{A})})$ |
| $G$ | Gini index | 0 . . . 1 | $\max(P(A)[P(B\|A)^2+P(\overline{B}\|A)^2]+P(\overline{A}[P(B\|\overline{A})^2+P(\overline{B}\|\overline{A})^2]-P(B)^2-P(\overline{B})^2,$ |
| | | | $P(B)[P(A\|B)^2+P(\overline{A}\|B)^2]+P(\overline{B}[P(A\|\overline{B})^2+P(\overline{A}\|\overline{B})^2]-P(A)^2-P(\overline{A})^2)$ |
| $s$ | support | 0 . . . 1 | $P(A,B)$ |
| $c$ | confidence | 0 . . . 1 | $max(P(B\|A), P(A\|B))$ |
| $L$ | Laplace | 0 . . . 1 | $\max(\frac{NP(A,B)+1}{NP(A)+2}, \frac{NP(A,B)+1}{NP(B)+2})$ |
| $IS$ | Cosine | 0 . . . 1 | $\frac{P(A,B)}{\sqrt{P(A)P(B)}}$ |
| $\gamma$ | coherence(Jaccard) | 0 . . . 1 | $\frac{P(A,B)}{P(A)+P(B)-P(A,B)}$ |
| $\alpha$ | all_confidence | 0 . . . 1 | $\frac{P(A,B)}{\max(P(A),P(B))}$ |
| $o$ | odds ratio | 0 . . . $\infty$ | $\frac{P(A,B)P(\overline{A},\overline{B})}{P(\overline{A},B)P(A,\overline{B})}$ |
| $V$ | Conviction | 0.5 . . . $\infty$ | $\max(\frac{P(A)P(\overline{B})}{P(A\overline{B})}, \frac{P(B)P(\overline{A})}{P(B\overline{A})})$ |
| $\lambda$ | lift | 0 . . . $\infty$ | $\frac{P(A,B)}{P(A)P(B)}$ |
| $S$ | Collective strength | 0 . . . $\infty$ | $\frac{P(A,B)+P(\overline{AB})}{P(A)P(B)+P(\overline{A})P(\overline{B})}\times\frac{1-P(A)P(B)-P(\overline{A})P(\overline{B})}{1-P(A,B)-P(\overline{AB})}$ |
| $\chi^2$ | $\chi^2$ | 0 . . . $\infty$ | $\sum_i\frac{(P(A_i)-E_i)^2}{E_i}$ |

# 第5章：挖掘关联规则

- 关联规则挖掘
- 事务数据库中(单维布尔)关联规则挖掘的可伸缩算法
- 挖掘各种关联/相关规则
- 基于限制的关联挖掘
- 顺序模式挖掘
- 频繁模式挖掘的应用/扩展
- 小结

# 基于约束的数据挖掘

- 自动地找出数据库中的所有模式? — 不现实!
  - 模式可能太多, 并不聚焦!
- 数据挖掘应当是一个 交互的 过程
  - 用户使用数据挖掘查询语言 (或图形用户界面) 指导需要挖掘什么
- 基于约束的挖掘
  - 用户灵活性: 提供挖掘的 约束
  - 系统优化: 考察限制, 寻找有效的挖掘—基于约束的挖掘

# 数据挖掘的约束

- 知识类型约束:
  - 分类, 关联, 等.
- 数据约束 （指定任务相关的数据集）— 使用类 SQL查询
  - 找出 Vancouver 2000年12月份一起销售的产品对
- 维/层约束-指定数据属性/概念分层结构的层次
  - 关于 region, price, brand, customer category
- 兴趣度约束
  - 强规则 : min_support ≥ 3%, min_confidence ≥ 60%
- 规则 (或模式) 约束-指定规则形式
  - 小额销售 (价格 < $10) 触发大额销售 (sum > $200)

# 元规则制导挖掘 Meta-Rule Guided Mining

- 元规则是带有部分约束谓词和常量的规则

  $P_1(X, Y) \wedge P_2(X, W) \Rightarrow buys(X, \text{“iPad”})$

- 一个导致的规则

  age(X, “15-25”) ^ profession(X, “student”) => buys(X, “iPad”)

- 通常情况, 元规则如下形式的规则模板

  $P_1 \wedge P_2 \wedge \ldots \wedge P_l \Rightarrow Q_1 \wedge Q_2 \wedge \ldots \wedge Q_r$

- 挖掘过程
  - 找出所有的频繁 (l+r) 谓词集 (基于最小支持度阈值)
    - 比须保留l子集的支持度/计数（计算规则的置信度）
  - （挖掘过程中）尽可能推进约束(见约束推进技术)
  - 尽可能地应用置信度, 相关和其他度量

# 规则约束-剪枝搜索空间

- 规则约束的分类
  - 反单调性Anti-monotonic
  - 单调性Monotonic
  - 简洁性 Succinct:
  - 可转变的Convertible:
  - 不可转变的

# 规则约束-反单调性

- 反单调性
  - *当项集 S 违反规则约束时, 它的任何超集合也违反约束*
  - ***sum(S.Price)*** $\leq v$ 是反单调的
  - ***sum(S.Price)*** $\geq v$ 不是反单调的
- 例. C: range(S.profit) $\leq$ 15是反单调的
  - 项集 *ab* 违反约束 C
  - *ab*的每个超集也违反约束C

TDB (min_sup=2)

| TID | Transaction |
|---|---|
| 10 | a, b, c, d, f |
| 20 | b, c, d, f, g, h |
| 30 | a, c, d, e, f |
| 40 | c, e, f, g |

| Item | Profit |
|---|---|
| a | 40 |
| b | 0 |
| c | -20 |
| d | 10 |
| e | -30 |
| f | 30 |
| g | 20 |
| h | -10 |

# 规则约束-单调性

- 单调性
  - *当项集 S 满足约束时, 它的任何超集合也满足约束*
  - ***sum(S.Price) ≥ v*** 是单调的
  - ***min(S.Price) ≤ v*** 是单调的
- 例. C: range(S.profit) ≥ 15
  - 项集 *ab* 满足 C
  - *ab*的每个超集合也满足C

**TDB (min_sup=2)**

| TID | Transaction |
|---|---|
| 10 | a, b, c, d, f |
| 20 | b, c, d, f, g, h |
| 30 | a, c, d, e, f |
| 40 | c, e, f, g |

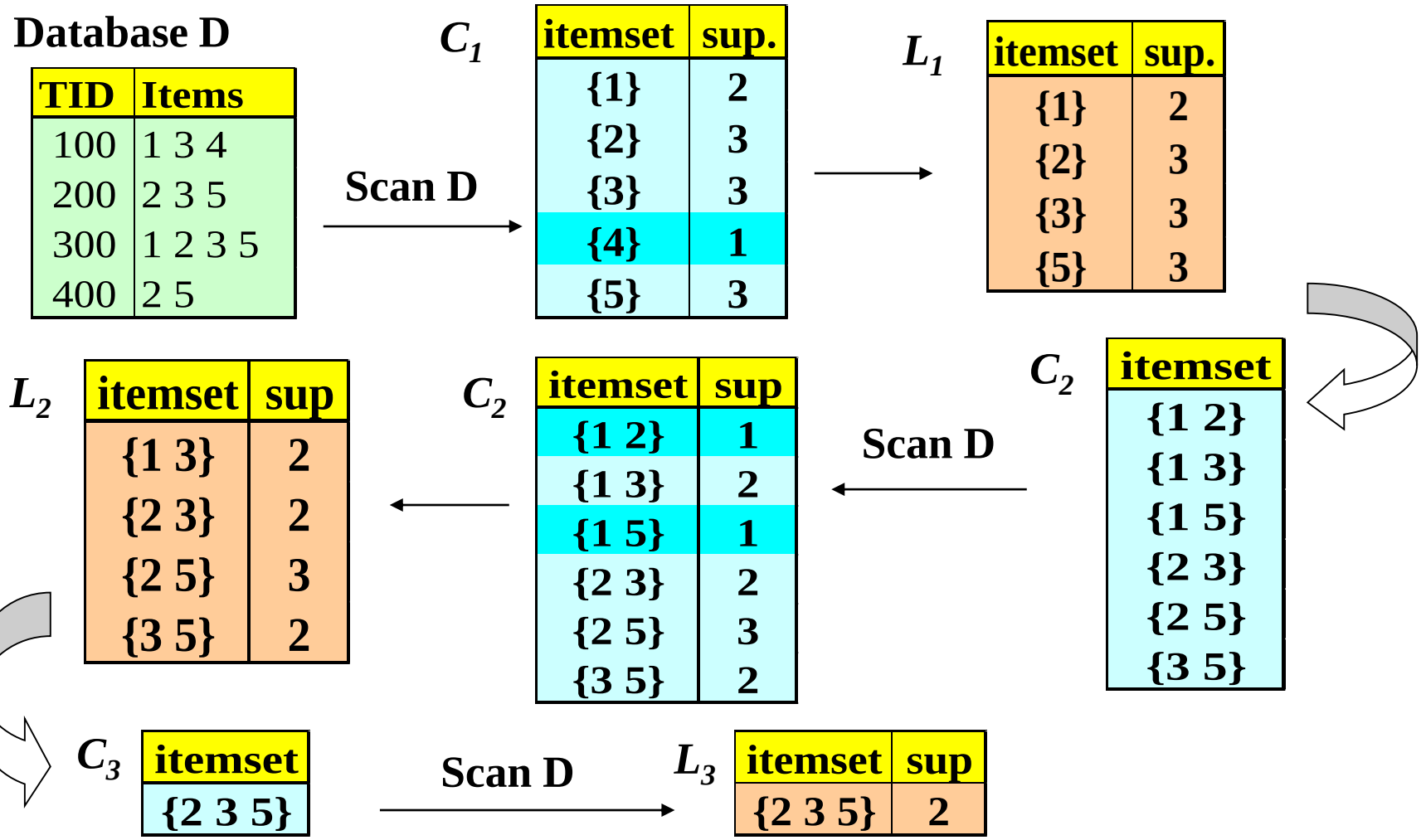
| Item | Profit |
|---|---|
| a | 40 |
| b | 0 |
| c | -20 |
| d | 10 |
| e | -30 |
| f | 30 |
| g | 20 |
| h | -10 |

# 简洁性

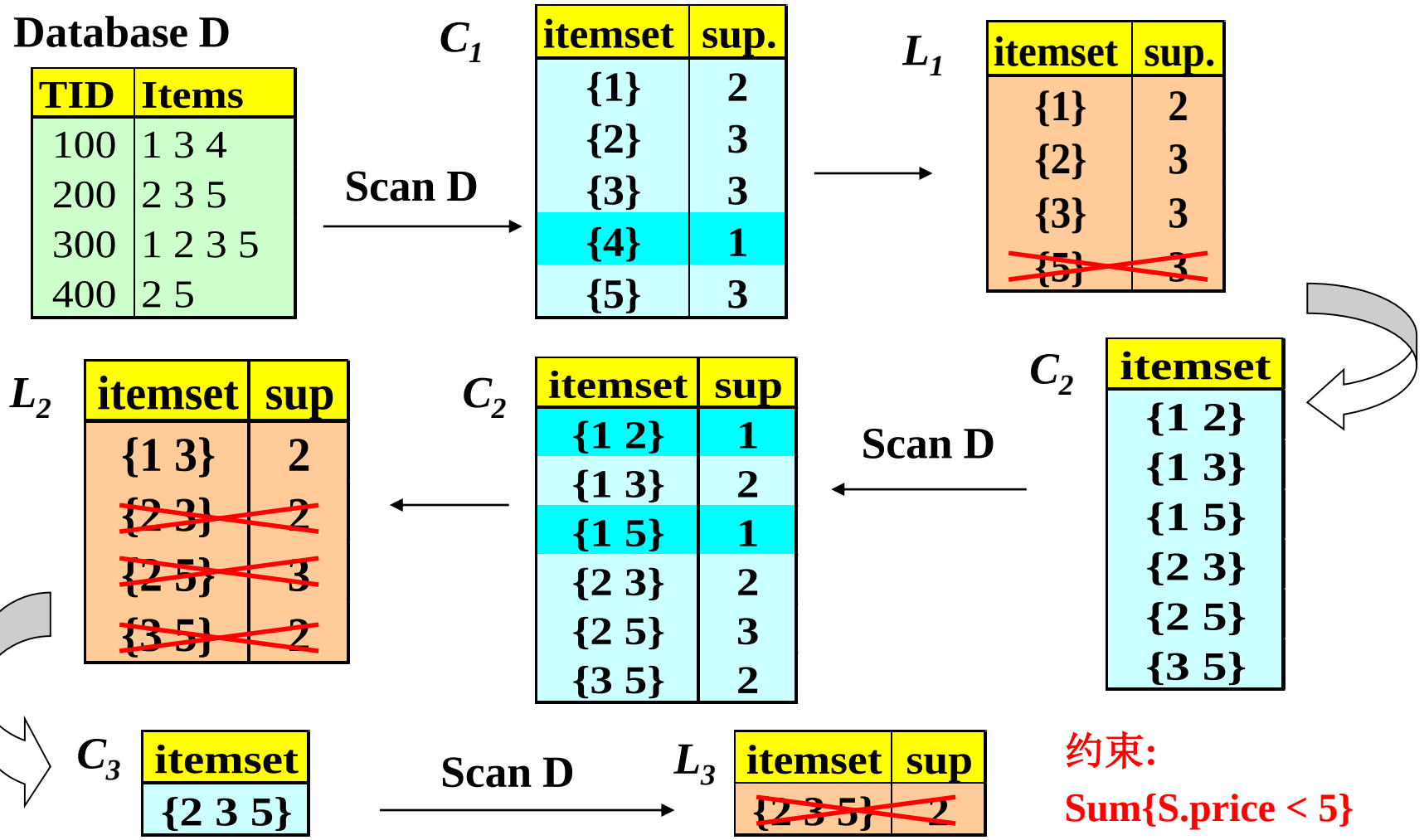
- 简洁性:
  - 给定满足约束$C$ 的项的集合 $A_1$, 则满足$C$ 的任意集合$S$ 都基于 $A_1$ , 即, $S$ 包含一个属于$A_1$ 的子集
  - 思想: 不查看事务数据库, 项集$S$ 是否满足约束C可以根据选取的项确定
  - $min(S.Price) \leq v$ 是简洁的
  - $sum(S.Price) \geq v$ 不是简洁的

- 优化: 如果 $C$ 是简洁的, $C$ 是预计数可推进的(pre-counting pushable)

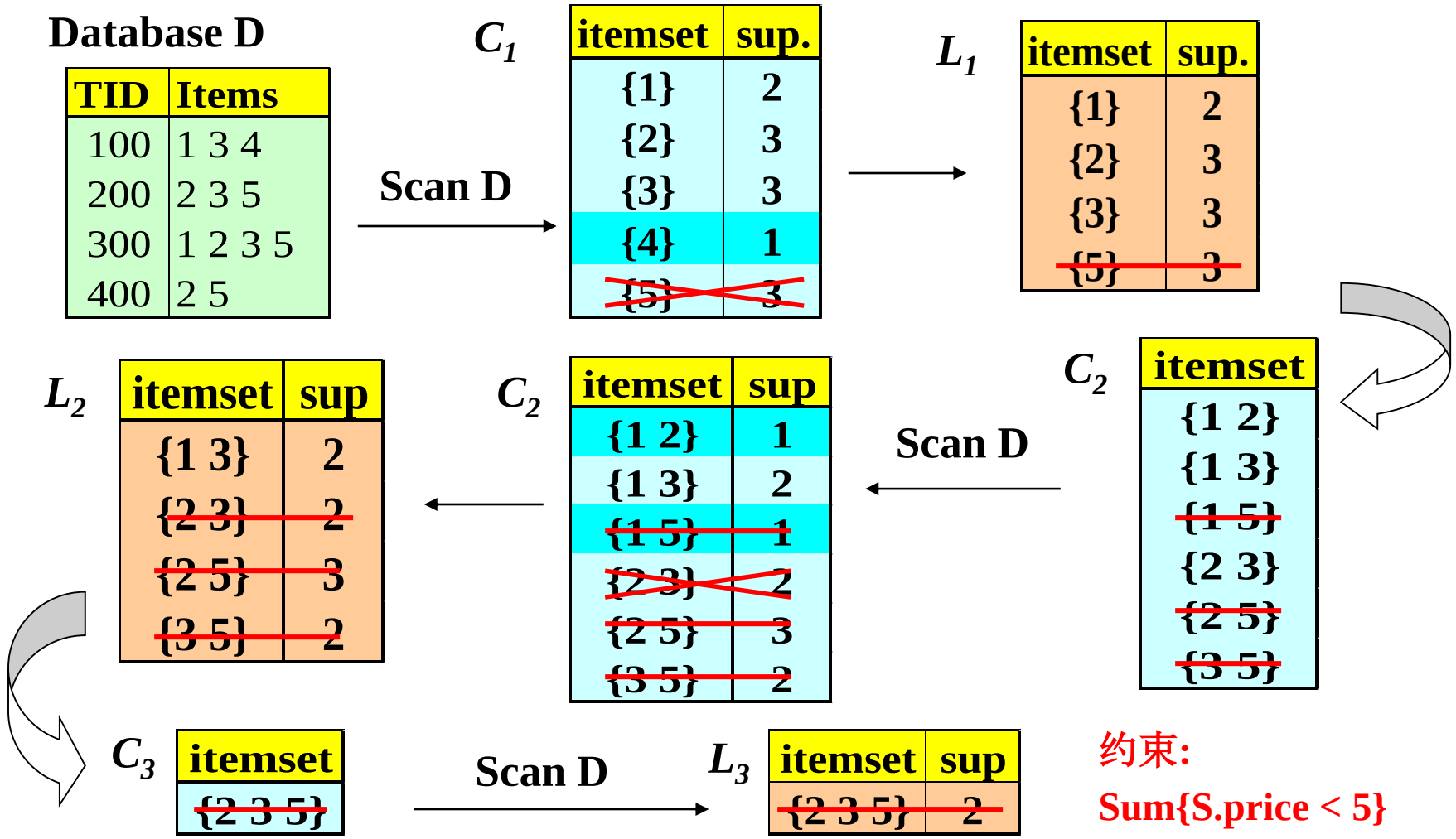
# Apriori 算法 — 一个例子



# 朴素算法: Apriori + 约束



# 受约束的Apriori 算法: 推进反单调约束



# 转换“强硬的”约束

- 通过将项适当地排序, 将强硬的约束转换成反单调的或单调的
- 例 C: avg(*S*.profit) ≥ 25
  - 将项按profit值的递减序排序
    - <*a, f, g, d, b, h, c, e*>
  - 如果项集 *afb* 违反 C
    - *afbh*, *afb**也违反C
    - 约束C成为 反单调的!

TDB (min_sup=2)

| TID | Transaction |
|---|---|
| 10 | a, b, c, d, f |
| 20 | b, c, d, f, g, h |
| 30 | a, c, d, e, f |
| 40 | c, e, f, g |

| Item | Profit |
|---|---|
| a | 40 |
| b | 0 |
| c | -20 |
| d | 10 |
| e | -30 |
| f | 30 |
| g | 20 |
| h | -10 |

# 可转变的约束

- 设 R 项集的项以特定次序安排，
- 可转变反单调
  - 如果项集 *S* 违反约束 C, 每个关于R以 *S* 为前缀的项集也违反约束C
  - 例. $avg(S) \geq v$ ， *如果*项值递减序排列
- 可转变单调
  - 如果项集 *S* 满足约束 C, 每个关于R以 *S* 为前缀的项集也满足约束C.
  - 例. $avg(S) \geq v$ ， *如果*项值递增序排列

# 强可转变约束

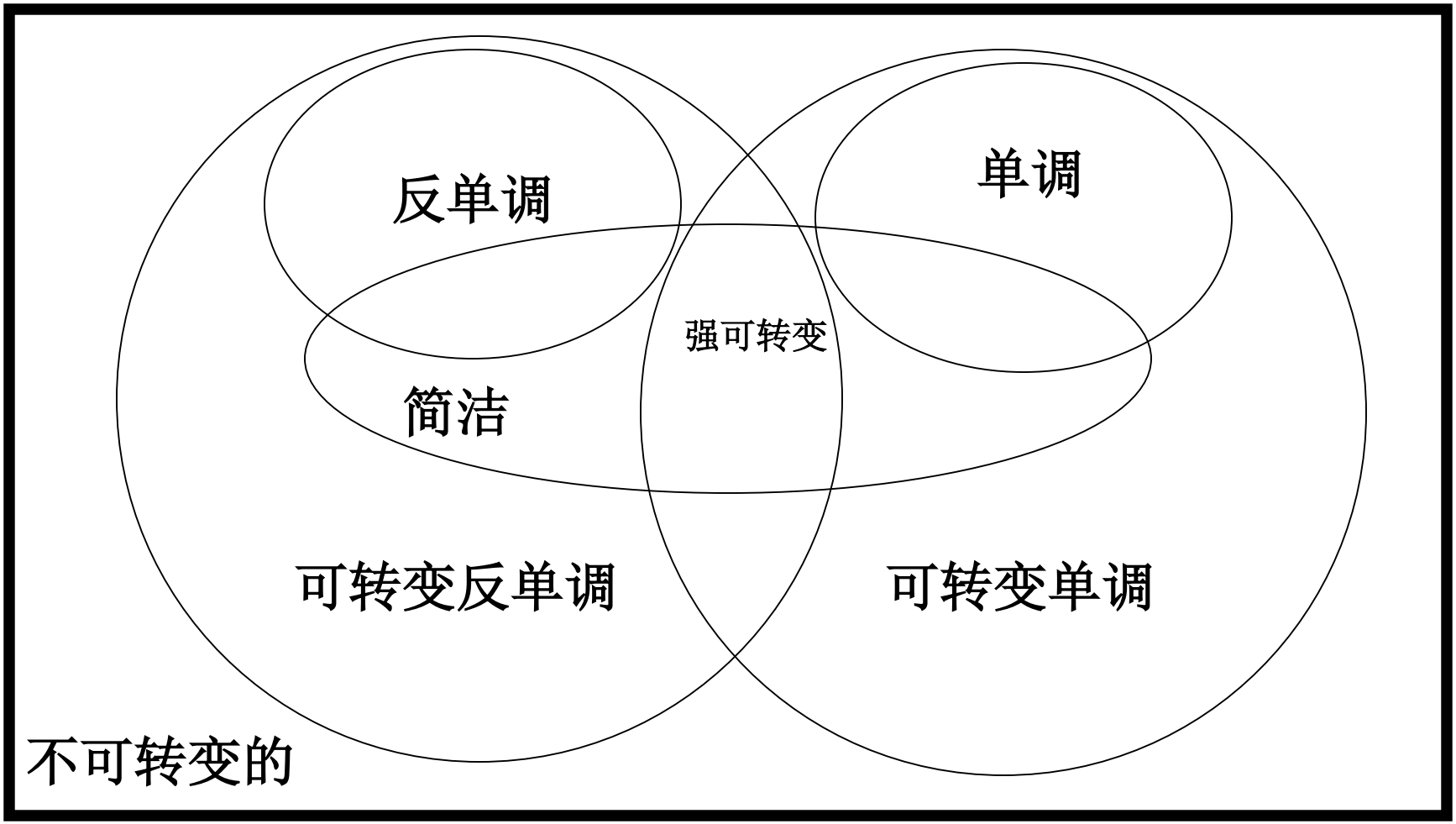
- **avg(X) ≥ 25 关于项值的递减序R: <*a, f, g, d, b, h, c, e*>是可转变反单调的**
  - 如果项集 *af* 违反约束 C, 每个以 *af* 为前缀的项集也违反C, 如 *afd*
- **avg(X) ≥ 25关于项值的递增序$R^{-1}$: <*e, c, h, b, d, g, f, a*>是可转变单调的**
  - 如果项集 *d* 满足约束 *C*, *df* 和 *dfa* 也满足, 它们具有前缀 *d*
- **这样, avg(X) ≥ 25 是 强可转变的**

| Item | Profit |
|---|---|
| a | 40 |
| b | 0 |
| c | -20 |
| d | 10 |
| e | -30 |
| f | 30 |
| g | 20 |
| h | -10 |

# 约束的性质汇总

| 约束 | 反单调 | 单调 | 简洁 |
|---|---|---|---|
| $v \in S$ | no | yes | yes |
| $S \supseteq V$ | no | yes | yes |
| $S \subseteq V$ | yes | no | yes |
| $\min(S) \leq v$ | no | yes | yes |
| $\min(S) \geq v$ | yes | no | yes |
| $\max(S) \leq v$ | yes | no | yes |
| $\max(S) \geq v$ | no | yes | yes |
| $\text{count}(S) \leq v$ | yes | no | weakly |
| $\text{count}(S) \geq v$ | no | yes | weakly |
| $\text{sum}(S) \leq v$ ( $a \in S, a \geq 0$ ) | yes | no | no |
| $\text{sum}(S) \geq v$ ( $a \in S, a \geq 0$ ) | no | yes | no |
| $\text{range}(S) \leq v$ | yes | no | no |
| $\text{range}(S) \geq v$ | no | yes | no |
| $\text{avg}(S)\ \theta\ v, \theta \in \{=, \leq, \geq\}$ | convertible | convertible | no |
| $\text{support}(S) \geq \xi$ | yes | no | no |
| $\text{support}(S) \leq \xi$ | no | yes | no |

# 约束的分类



# Apriori 能够处理可转变的约束吗?

- 可转变的, 但既不是单调, 反单调, 也不是简洁的约束不能推进到 Apriori 挖掘算法的挖掘过程中
  - 在逐级的框架下, 不能做直接基于该约束的剪枝
  - 项集 *df* 违反 约束 C: avg(X)>=25
  - 由于 *adf* 满足 C, Apriori 需要 *df* 来组装 *adf*, 因此不能将*df* 剪去
- 但是, 在模式增长框架下该约束可以推进到挖掘过程中!

| Item | Value |
|---|---|
| a | 40 |
| b | 0 |
| c | -20 |
| d | 10 |
| e | -30 |
| f | 30 |
| g | 20 |
| h | -10 |

# 具有可转变约束的挖掘

- C: avg(X)>=25, min_sup=2
- 以值的递减序 R:

  <*a, f, g, d, b, h, c, e*>

  列出事务中的每一个项
  - 关于R, C是可转变反单调的
- 扫描 TDB 一次
  - 删除非频繁项
    - 项 *h* 被删除
  - 项 a 和 f 是好的, …
- 基于投影的挖掘
  - 利用项投影的适当次序
  - 许多强硬的约束可以转变成(反)单调的

TDB (min_sup=2)

| tem | Profit |
|---|---|
| a | 40 |
| f | 30 |
| g | 20 |
| d | 10 |
| b | 0 |
| h | -10 |
| c | -20 |
| e | -30 |

| TID | Transaction |
|---|---|
| 10 | a, f, d, b, c |
| 20 | f, g, d, b, c |
| 30 | a, f, d, c, e |
| 40 | f, g, h, c, e |

# 讨论—处理多个约束

- 不同的约束需要不同的, 甚至相互冲突的项序
- 如果存在序 ***R***, 使得约束 $C_1$ 和 $C_2$ 关于**R**是可转变的, 则两个可转变的约束之间不存在冲突
- 如果项序存在冲突
  - 试图先满足一个约束
  - 然后使用另一约束的序, 在相应的投影数据库中挖掘频繁项集

# 文献: 频繁模式挖掘方法


- **R. Agarwal, C. Aggarwal, and V. V. V. Prasad. A tree projection algorithm for generation of frequent itemsets. Journal of Parallel and Distributed Computing, 2000.**
- **R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. SIGMOD'93, 207-216, Washington, D.C.**
- **R. Agrawal and R. Srikant. Fast algorithms for mining association rules. VLDB'94 487-499, Santiago, Chile.**
- **J. Han, J. Pei, and Y. Yin: "Mining frequent patterns without candidate generation". In Proc. ACM-SIGMOD'2000, pp. 1-12, Dallas, TX, May 2000.**
- **H. Mannila, H. Toivonen, and A. I. Verkamo. Efficient algorithms for discovering association rules. KDD'94, 181-192, Seattle, WA, July 1994.**

# 文献: 频繁模式挖掘方法


- A. Savasere, E. Omiecinski, and S. Navathe. An efficient algorithm for mining association rules in large databases. VLDB'95, 432-443, Zurich, Switzerland.
- C. Silverstein, S. Brin, R. Motwani, and J. Ullman. Scalable techniques for mining causal structures. VLDB'98, 594-605, New York, NY.
- R. Srikant and R. Agrawal. Mining generalized association rules. VLDB'95, 407-419, Zurich, Switzerland, Sept. 1995.
- R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. SIGMOD'96, 1-12, Montreal, Canada.
- H. Toivonen. Sampling large databases for association rules. VLDB'96, 134-145, Bombay, India, Sept. 1996.
- M.J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li. New algorithms for fast discovery of association rules. KDD’97. August 1997.

# 文献: 频繁模式挖掘 (性能改进)


- **S. Brin, R. Motwani, J. D. Ullman, and S. Tsur. Dynamic itemset counting and implication rules for market basket analysis. SIGMOD'97, Tucson, Arizona, May 1997.**
- **D.W. Cheung, J. Han, V. Ng, and C.Y. Wong. Maintenance of discovered association rules in large databases: An incremental updating technique. ICDE'96, New Orleans, LA.**
- **T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Data mining using two-dimensional optimized association rules: Scheme, algorithms, and visualization. SIGMOD'96, Montreal, Canada.**
- **E.-H. Han, G. Karypis, and V. Kumar. Scalable parallel data mining for association rules. SIGMOD'97, Tucson, Arizona.**
- **J.S. Park, M.S. Chen, and P.S. Yu. An effective hash-based algorithm for mining association rules. SIGMOD'95, San Jose, CA, May 1995.**

# 文献: 频繁模式挖掘 (性能改进)


- G. Piatetsky-Shapiro. Discovery, analysis, and presentation of strong rules. In G. Piatetsky-Shapiro and W. J. Frawley, Knowledge Discovery in Databases,. AAAI/MIT Press, 1991.
- J.S. Park, M.S. Chen, and P.S. Yu. An effective hash-based algorithm for mining association rules. SIGMOD'95, San Jose, CA.
- S. Sarawagi, S. Thomas, and R. Agrawal. Integrating association rule mining with relational database systems: Alternatives and implications. SIGMOD'98, Seattle, WA.
- K. Yoda, T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Computing optimized rectilinear regions for association rules. KDD'97, Newport Beach, CA, Aug. 1997.
- M. J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li. Parallel algorithm for discovery of association rules. Data Mining and Knowledge Discovery, 1:343-374, 1997.

# 文献: 频繁模式挖掘 (外延)


- **S. Brin, R. Motwani, and C. Silverstein. Beyond market basket: Generalizing association rules to correlations. SIGMOD'97, 265-276, Tucson, Arizona.**
- **J. Han and Y. Fu. Discovery of multiple-level association rules from large databases. VLDB'95, 420-431, Zurich, Switzerland.**
- **M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, and A.I. Verkamo. Finding interesting rules from large sets of discovered association rules. CIKM'94, 401-408, Gaithersburg, Maryland.**
- **F. Korn, A. Labrinidis, Y. Kotidis, and C. Faloutsos. Ratio rules: A new paradigm for fast, quantifiable data mining. VLDB'98, 582-593, New York, NY.**

# 文献: 频繁模式挖掘 (外延)


- B. Lent, A. Swami, and J. Widom. Clustering association rules. ICDE'97, 220-231, Birmingham, England.
- R. Meo, G. Psaila, and S. Ceri. A new SQL-like operator for mining association rules. VLDB'96, 122-133, Bombay, India.
- R.J. Miller and Y. Yang. Association rules over interval data. SIGMOD'97, 452-461, Tucson, Arizona.
- A. Savasere, E. Omiecinski, and S. Navathe. Mining for strong negative associations in a large database of customer transactions. ICDE'98, 494-502, Orlando, FL, Feb. 1998.
- D. Tsur, J. D. Ullman, S. Abitboul, C. Clifton, R. Motwani, and S. Nestorov. Query flocks: A generalization of association-rule mining. SIGMOD'98, 1-12, Seattle, Washington.
- J. Pei, A.K.H. Tung, J. Han. Fault-Tolerant Frequent Pattern Mining: Problems and Challenges. SIGMOD DMKD'01, Santa Barbara, CA.

# 文献: 挖掘最大模式和闭项集


- R. J. Bayardo. Efficiently mining long patterns from databases. SIGMOD'98, 85-93, Seattle, Washington.
- J. Pei, J. Han, and R. Mao, "CLOSET: An Efficient Algorithm for Mining Frequent Closed Itemsets", Proc. 2000 ACM-SIGMOD Int. Workshop on Data Mining and Knowledge Discovery (DMKD'00), Dallas, TX, May 2000.
- N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. ICDT'99, 398-416, Jerusalem, Israel, Jan. 1999.
- M. Zaki. Generating Non-Redundant Association Rules. KDD'00. Boston, MA. Aug. 2000
- M. Zaki. CHARM: An Efficient Algorithm for Closed Association Rule Mining, SIAM'02

# 文献: 基于约束的频繁模式挖掘


- **G. Grahne, L. Lakshmanan, and X. Wang. Efficient mining of constrained correlated sets. ICDE'00, 512-521, San Diego, CA, Feb. 2000.**
- **Y. Fu and J. Han. Meta-rule-guided mining of association rules in relational databases. KDOOD'95, 39-46, Singapore, Dec. 1995.**
- **J. Han, L. V. S. Lakshmanan, and R. T. Ng, "Constraint-Based, Multidimensional Data Mining", COMPUTER (special issues on Data Mining), 32(8): 46-50, 1999.**
- **L. V. S. Lakshmanan, R. Ng, J. Han and A. Pang, "Optimization of Constrained Frequent Set Queries with 2-Variable Constraints", SIGMOD'99**

# 文献: 基于约束的频繁模式挖掘


- R. Ng, L.V.S. Lakshmanan, J. Han & A. Pang. “Exploratory mining and pruning optimizations of constrained association rules.” SIGMOD’98
- J. Pei, J. Han, and L. V. S. Lakshmanan, "Mining Frequent Itemsets with Convertible Constraints", Proc. 2001 Int. Conf. on Data Engineering (ICDE'01), April 2001.
- J. Pei and J. Han "Can We Push More Constraints into Frequent Pattern Mining?", Proc. 2000 Int. Conf. on Knowledge Discovery and Data Mining (KDD'00), Boston, MA, August 2000.
- R. Srikant, Q. Vu, and R. Agrawal. Mining association rules with item constraints. KDD'97, 67-73, Newport Beach, California.

# 文献: 序列模式挖掘方法


- R. Agrawal and R. Srikant. Mining sequential patterns. ICDE'95, 3-14, Taipei, Taiwan.
- R. Srikant and R. Agrawal. Mining sequential patterns: Generalizations and performance improvements. EDBT’96.
- J. Han, J. Pei, B. Mortazavi-Asl, Q. Chen, U. Dayal, M.-C. Hsu, "FreeSpan: Frequent Pattern-Projected Sequential Pattern Mining", Proc. 2000 Int. Conf. on Knowledge Discovery and Data Mining (KDD'00), Boston, MA, August 2000.
- H. Mannila, H Toivonen, and A. I. Verkamo. Discovery of frequent episodes in event sequences. Data Mining and Knowledge Discovery, 1:259-289, 1997.

# 文献: 序列模式挖掘方法


- **J. Pei, J. Han, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu, "PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth", Proc. 2001 Int. Conf. on Data Engineering (ICDE'01), Heidelberg, Germany, April 2001.**
- **B. Ozden, S. Ramaswamy, and A. Silberschatz. Cyclic association rules. ICDE'98, 412-421, Orlando, FL.**
- **S. Ramaswamy, S. Mahajan, and A. Silberschatz. On the discovery of interesting patterns in association rules. VLDB'98, 368-379, New York, NY.**
- **M.J. Zaki. Efficient enumeration of frequent sequences. CIKM'98. Novermber 1998.**
- **M.N. Garofalakis, R. Rastogi, K. Shim: SPIRIT: Sequential Pattern Mining with Regular Expression Constraints. VLDB 1999: 223-234, Edinburgh, Scotland.**

# 文献: 空间, 多媒体, 文本和 Web 数据库频繁模式挖掘


- **K. Koperski, J. Han, and G. B. Marchisio, "Mining Spatial and Image Data through Progressive Refinement Methods", Revue internationale de gomatique (European Journal of GIS and Spatial Analysis), 9(4):425-440, 1999.**
- **A. K. H. Tung, H. Lu, J. Han, and L. Feng, "Breaking the Barrier of Transactions: Mining Inter-Transaction Association Rules", Proc. 1999 Int. Conf. on Knowledge Discovery and Data Mining (KDD'99), San Diego, CA, Aug. 1999, pp. 297-301.**
- **J. Han, G. Dong and Y. Yin, "Efficient Mining of Partial Periodic Patterns in Time Series Database", Proc. 1999 Int. Conf. on Data Engineering (ICDE'99), Sydney, Australia, March 1999, pp. 106-115.**

# 文献: 空间, 多媒体, 文本和 Web 数据库频繁模式挖掘


- **H. Lu, L. Feng, and J. Han, "Beyond Intra-Transaction Association Analysis:Mining Multi-Dimensional Inter-Transaction Association Rules", ACM Transactions on Information Systems (TOIS'00), 18(4): 423-454, 2000.**
- **O. R. Zaiane, M. Xin, J. Han, "Discovering Web Access Patterns and Trends by Applying OLAP and Data Mining Technology on Web Logs," Proc. Advances in Digital Librar ies Conf. (ADL'98), Santa Barbara, CA, April 1998, pp. 19-29**
- **O. R. Zaiane, J. Han, and H. Zhu, "Mining Recurrent Items in Multimedia with Progressive Resolution Refinement", Proc. 2000 Int. Conf. on Data Engineering (ICDE'00), San Diego, CA, Feb. 2000, pp. 461-470.**

# 文献: 用于分类和数据方计算的频繁模式挖掘


- K. Beyer and R. Ramakrishnan. Bottom-up computation of sparse and iceberg cubes. SIGMOD'99, 359-370, Philadelphia, PA, June 1999.
- M. Fang, N. Shivakumar, H. Garcia-Molina, R. Motwani, and J. D. Ullman. Computing iceberg queries efficiently. VLDB'98, 299-310, New York, NY, Aug. 1998.
- J. Han, J. Pei, G. Dong, and K. Wang, “Computing Iceberg Data Cubes with Complex Measures”, Proc. ACM-SIGMOD’2001, Santa Barbara, CA, May 2001.
- M. Kamber, J. Han, and J. Y. Chiang. Metarule-guided mining of multi-dimensional association rules using data cubes. KDD'97, 207-210, Newport Beach, California.
- K. Beyer and R. Ramakrishnan. Bottom-up computation of sparse and iceberg cubes. SIGMOD’99
- T. Imielinski, L. Khachiyan, and A. Abdulghani. Cubegrades: Generalizing association rules. Technical Report, Aug. 2000