数据挖掘
第5章 关联分析

教师：王东京
学院：计算机学院
邮箱：dongjing.wang@hdu.edu.cn

关联规则挖掘 Association Rule Mining

给定一组交易（transactions），请根据交易中其他项目（item）的发生情况找到可以预测某个项目发生的规则

购物篮交易事务（transactions）

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

Association Rules示例

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

关联规则挖掘 Association Rule Mining

问题1：关联规则挖掘的成本

问题2：关联规则的准确性

购物篮交易事务（transactions）

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

Association Rules示例

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

暗示（Implication）意味着共现（co-occurrence），而不是因果关系（causality）！

定义：频繁项集 Frequent Itemset

项集 Itemset

- 一个或多个项（item）的集合
  - 示例: {Milk, Bread, Diaper}
- K项集 k-itemset
  - 包含k个项的项集

支持度计数 Support count (σ)

- 项集在事务出现的频率
- 例如 σ({Milk, Bread,Diaper}) = ?
- 2

支持度 Support （s）

- 包含项集的交易/事务（transaction）比例
- 例如 s({Milk, Bread, Diaper}) = ?
- 2/5

频繁项集 Frequent Itemset

- 支持度大于等于最小阈值（ *minsup* threshold ）的项集

购物篮交易事务（transactions）

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

定义：关联规则 Association Rule

关联规则 Association Rule

- 形如X → Y的蕴含（implication）表达式其中 X 和 Y 都是项集（不相交）
- 例如:
  {Milk, Diaper} → {Beer}

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

规则评估指标 Rule Evaluation Metrics

- 支持度 Support (s)
  - 同时包含X和Y的交易/事务比例
- 置信度 Confidence (c)
  - 衡量Y中的项目在包含X的交易/事务中出现的频率

示例:

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

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

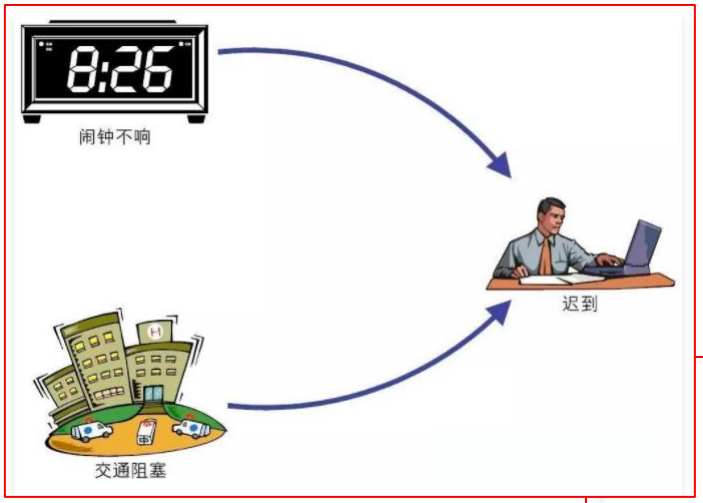
$$c = \frac{\sigma(\text{Milk, Diaper, Beer})}{\sigma(\text{Milk, Diaper})} = \frac{2}{3} = 0.67$$

$$s(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{N}$$

$$c(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)}$$

关联与因果

关联规则中的蕴含（Implication）意味着共现（co-occurrence），而不是因果关系（causality）！



设置

规则{Diaper} => {Beer} 的支持度s和置信度c分别为?

A s=0.4, c=1

B s=0.6, c=1

C s=0.4, c=0.75

D s=0.6, c=0.75

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

提交

关联规则挖掘任务

给定一组交易T，关联规则挖掘的目标是找到所有具有以下特征的规则：

- Support支持度 ≥ *minsup* 阈值
- Confidence置信度 ≥ *minconf* 阈值
- 如何寻找?

原始/蛮力方法 Brute-force approach:

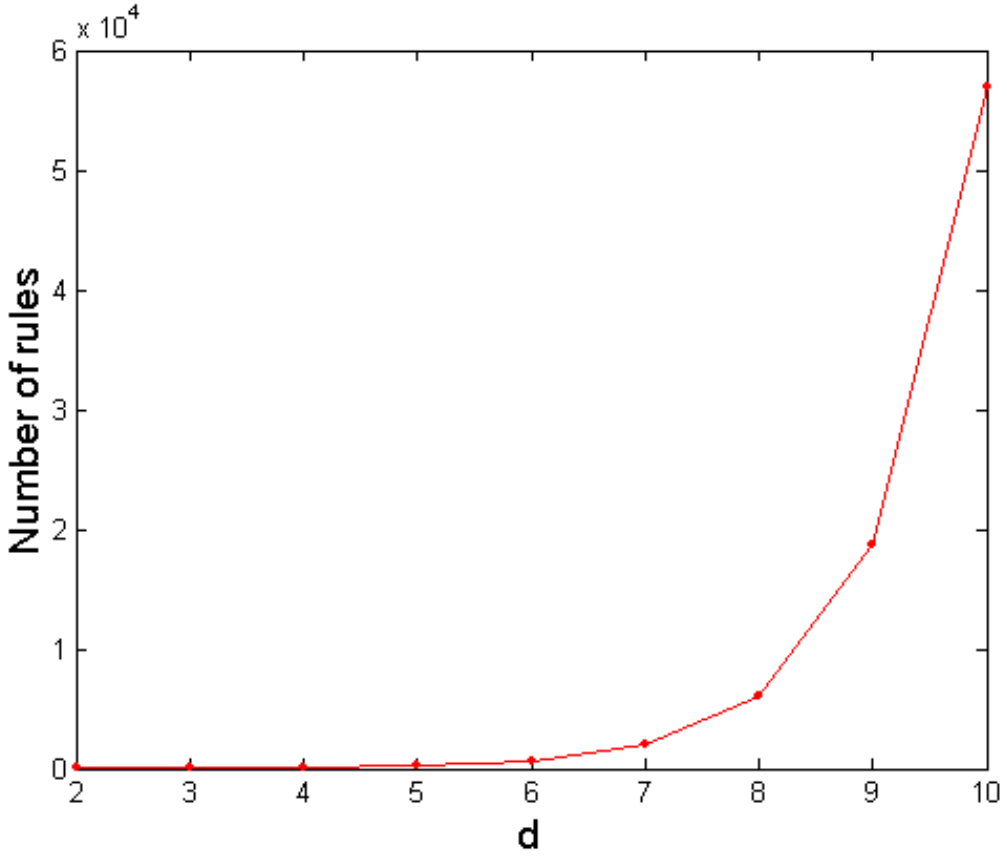
- 列出所有可能的关联规则
- 计算每个规则的支持度和置信度
- 修剪不满足*minsup*和*minconf*阈值的规则

⇒理论可行，实际计算不可行（Computationally prohibitive）！

计算复杂度 Computational Complexity

给定d个项（item）：

- 项集总数=？
  - 2^d
- 可能的关联规则总数：



$$R = \sum_{k=1}^{d-1}\left[\binom{d}{k} \times \sum_{j=1}^{d-k}\binom{d-k}{j}\right]$$

$$= 3^d - 2^{d+1} + 1$$

If d=6, R = 602 rules

关联规则挖掘 Mining Association Rules

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

规则示例:

{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)

观察结果 Observations:

- 上面所有规则都是相同项集的二元划分（binary partition）：
  {Milk, Diaper, Beer}
- 源自同一项集的规则具有相同的支持度，但可以具有不同的置信度
- 因此，我们可能会拆分（decouple）对支持度和置信度的要求

关联规则挖掘 Mining Association Rules

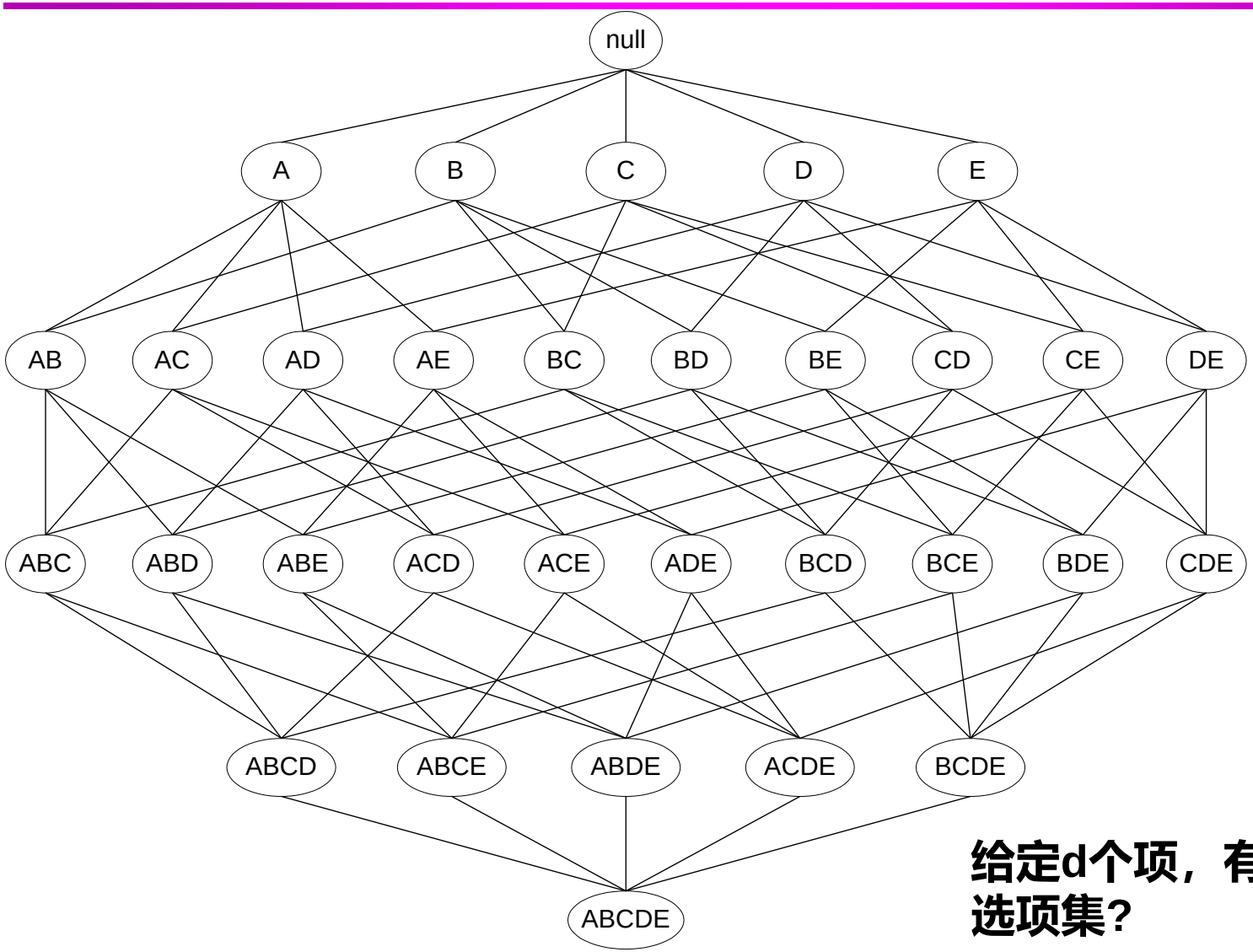
方法分为两步:

1. 频繁项集产生Frequent Itemset Generation
   - 生成所有支持度support ≥ minsup的项集（频繁项集）

2. 规则的产生Rule Generation
   - 从上一步发现的频繁项集中提取所有高置信度的规则，这些规则称作强规则（ strong rule）。

生成频繁项集在计算上的代价仍然非常大

频繁项集生成 Frequent Itemset Generation



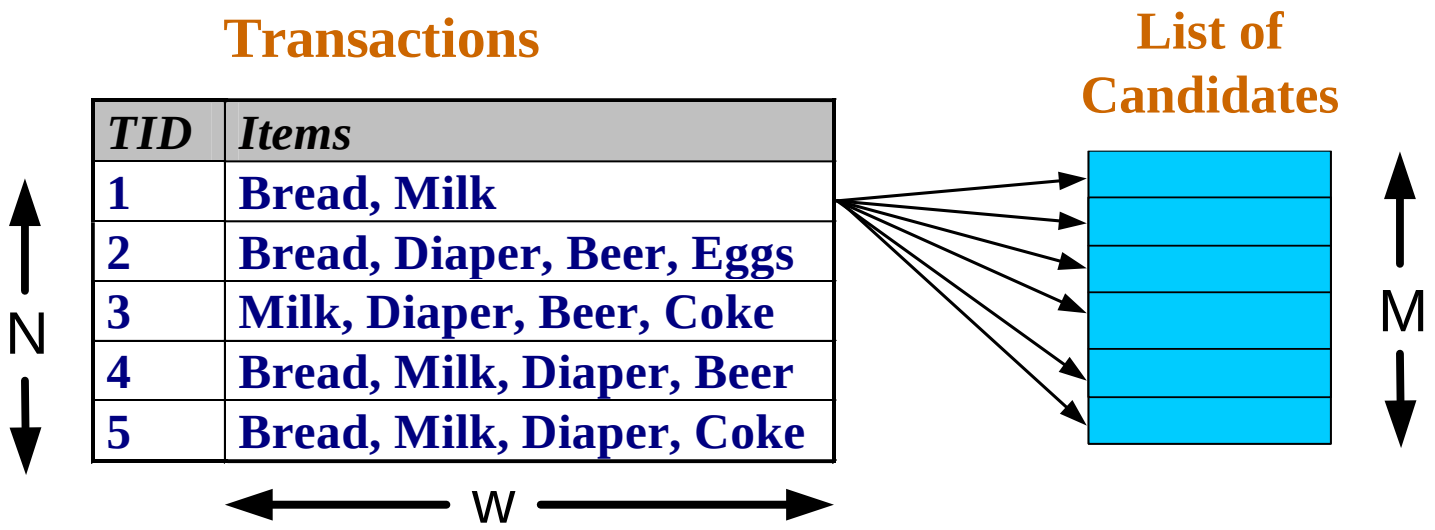
给定d个项，有__个可能的候选项集？

2^d-1（如果不包括空集）

频繁项集生成 Frequent Itemset Generation

原始方法 Brute-force approach:

- 每个格（lattice）对应的项集都是一个候选的频繁项集
- 通过扫描数据库计算每个候选项集的支持度



- 将每个事务与每个候选集进行比较
- 复杂度 ~ O(NMw) => Expensive since $M = 2^d$!!!
- 如何降低复杂度?

频繁项集生成策略（提高效率）

减少候选数目(M)

- 完全搜索 Complete search: $M=2^d$
- 利用剪枝技术（而非支持度）减少M

减少事务/交易（transactions）数目(N)

- 随着项集的大小增加，减少N的大小
- Used by DHP and vertical-based mining algorithms

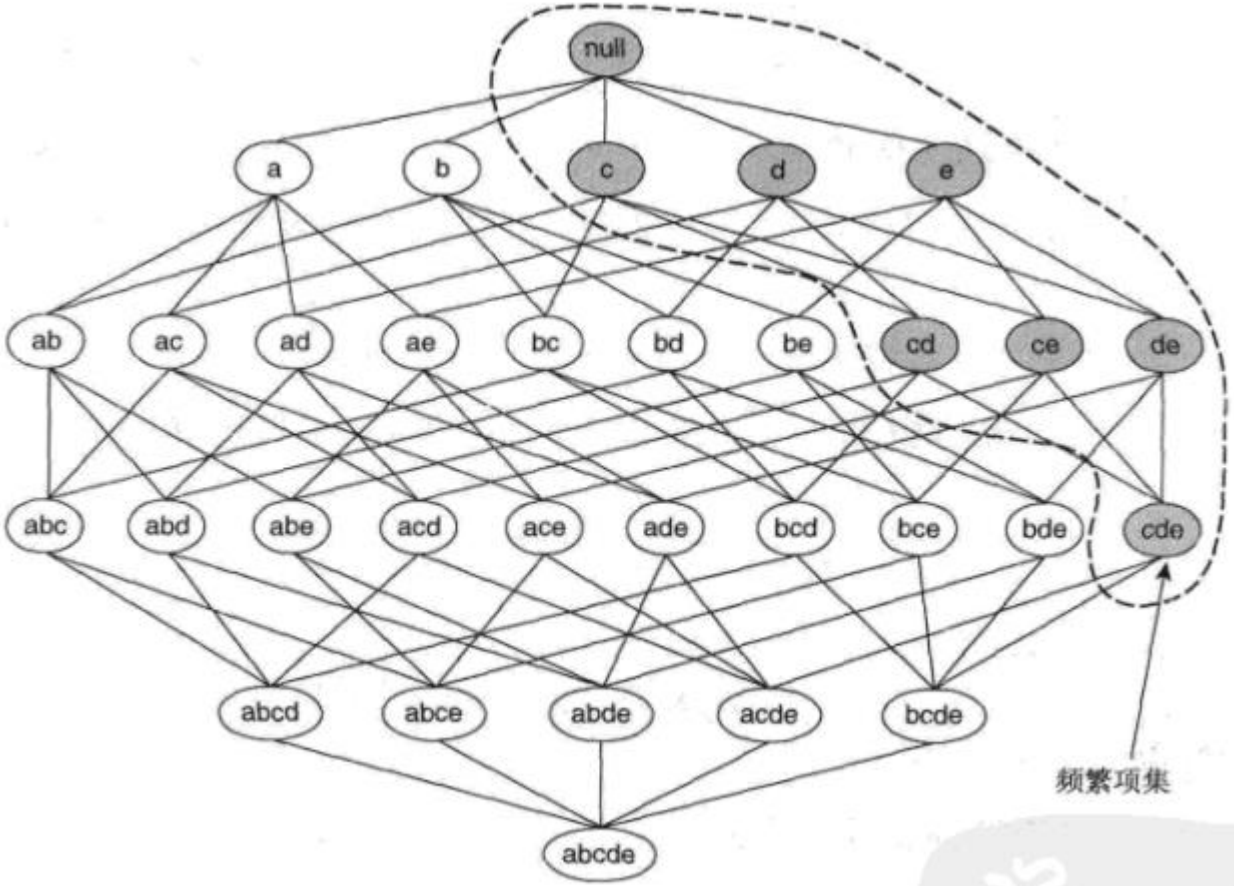
减少需要比较的次数(NM)

- 使用高效的数据结构存储候选项集或者事务
- 没有必要比较每一个候选项集和事务

减少候选数目 Reducing Number of Candidates

先验原则 Apriori principle:

- 如果一个项集是频繁的，那么它的所有子集也一定是频繁的



减少候选数目 Reducing Number of Candidates

先验原则 Apriori principle:

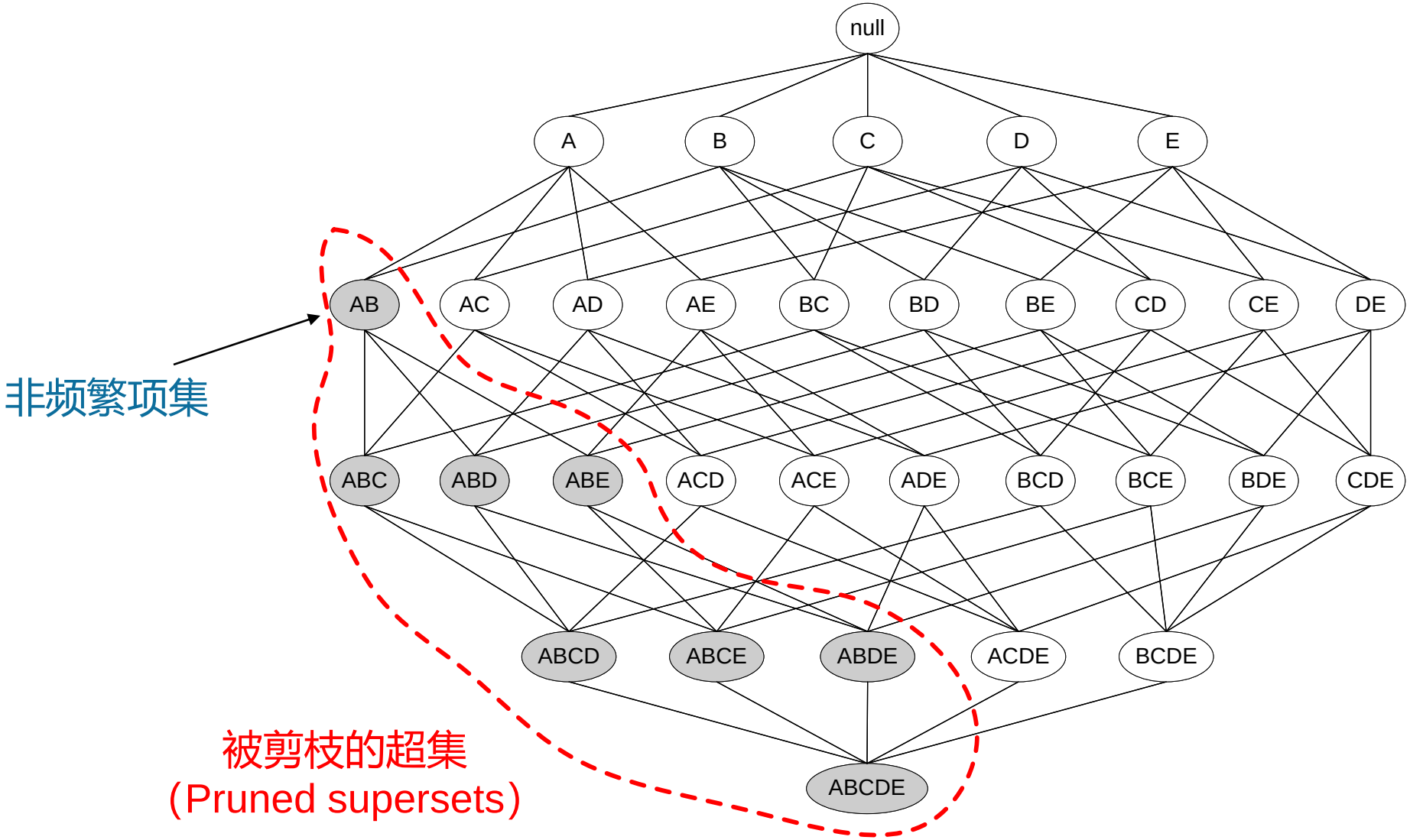
- 如果一个项集是频繁的，那么它的所有子集也一定是频繁的

由于支持度措施（support measure）的以下特性成立，因此Apriori principle原理成立：

$$\forall X, Y : (X \subseteq Y) \Rightarrow s(X) \geq s(Y)$$

- 项集的支持度永远不会超出其子集的支持度
- 这就是所谓的支持度的**反单调性（anti-monotone）**

先验原理阐述 Illustrating Apriori Principle



单选题 1分

设置

关于“如何提高频繁项集生成的效率”的策略说法**错误**的是?

A 利用剪枝计数减少候选项集数目

B 通过扫描事务数据库，将每个事务与每个候选集进行比较

C 减少事务（transaction）数目

D 减少比较候选项集和事务的次数

提交

Apriori 算法

候选产生：蛮力方法 Candidate Generation: Brute-force method

表 6-1 购物篮事务的例子

TID	项集
1	{面包，牛奶}
2	{面包，尿布，啤酒，鸡蛋}
3	{牛奶，尿布，啤酒，可乐}
4	{面包，牛奶，尿布，啤酒}
5	{面包，牛奶，尿布，可乐}

候选 1-项集

项	计数
啤酒	3
面包	4
可乐	2
尿布	4
牛奶	4
鸡蛋	1

最小支持度计数=3

候选 2-项集

项集	计数
{啤酒，面包}	2
{啤酒，尿布}	3
{啤酒，牛奶}	2
{面包，尿布}	3
{面包，牛奶}	3
{尿布，牛奶}	3

因支持度低而被删除的项集

候选 3-项集

项集	计数
{面包，尿布，牛奶}	3

图 6-5 使用 *Apriori* 算法产生频繁项集的例子

候选产生：蛮力方法 Candidate Generation: Brute-force method

表 6-1　购物篮事务的例子

TID	项　集
1	{面包，牛奶}
2	{面包，尿布，啤酒，鸡蛋}
3	{牛奶，尿布，啤酒，可乐}
4	{面包，牛奶，尿布，啤酒}
5	{面包，牛奶，尿布，可乐}

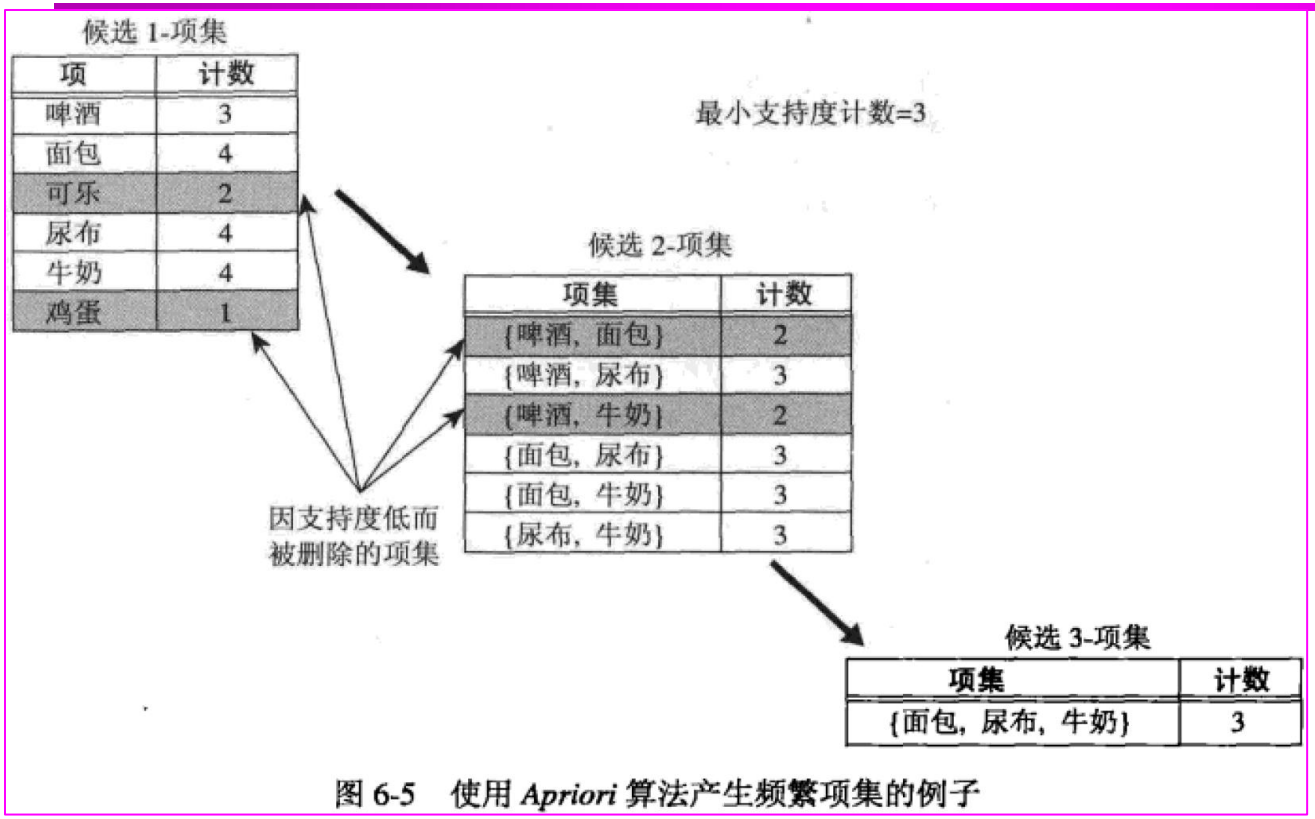


图 6-5　使用 *Apriori* 算法产生频繁项集的例子

通过计算产生的候选项集数目，可以看出先验剪枝策略的有效性。枚举所有项集（到 3-项集）的蛮力策略将产生 $C_6^1 + C_6^2 + C_6^3 = 6 + 15 + 20 = 41$ 个候选；而使用先验原理，将减少为 $C_6^1 + C_4^2 + 1 = 6 + 6 + 1 = 13$ 个候选。甚至在这个简单的例子中，候选项集的数目也降低了 68%。

Apriori 算法

- F_k: 频繁k-项集的集合 frequent k-itemsets
- C_k: 候选k-项集的集合 candidate k-itemsets

算法过程

- Let k=1
- Generate F_1 = {frequent 1-itemsets} #频繁1项集
- Repeat until F_k is empty
  - **Candidate Generation**: Generate C_{k+1} from F_k
  - **Candidate Pruning**: Prune candidate itemsets in C_{k+1} containing subsets of length k that are infrequent
  - **Support Counting**: Count the support of each candidate in C_{k+1} by scanning the DB
  - **Candidate Elimination**: Eliminate candidates in C_{k+1} that are infrequent, leaving only those that are frequent => F_{k+1}

Apriori 算法伪代码

算法 6.1 *Apriori* 算法的频繁项集产生

```
1:  k = 1
2:  F_k = {i | i∈I ∧ σ({i}) ≥ N × minsup}      {发现所有的频繁 1-项集}
3:  repeat
4:      k = k+1
5:      C_k = apriori-gen(F_{k-1})              {产生候选项集}
6:      for 每个事务 t∈T do
7:          C_t = subset(C_k, t)                {识别属于 t 的所有候选}
8:          for 每个候选项集 c∈C_t do
9:              σ(c) = σ(c) + 1                 {支持度计数增值}
10:         end for
11:     end for
12:     F_k = {c | c∈C_k ∧ σ(c) ≥ N × minsup}   {提取频繁 k-项集}
13: until F_k = ∅
14: Result = ∪F_k
```

候选剪枝 Candidate Pruning

算法 6.1 *Apriori* 算法的频繁项集产生

```
1:  k = 1
2:  F_k = {i | i∈I ∧ σ({i}) ≥ N × minsup}      {发现所有的频繁 1-项集}
3:  repeat
4:    k = k+1
5:    C_k = apriori-gen(F_{k-1})                {产生候选项集}
6:    for 每个事务 t∈T do
7:      C_t = subset(C_k, t)                    {识别属于 t 的所有候选}
8:      for 每个候选项集 c∈C_t do
9:        σ(c) = σ(c) + 1                       {支持度计数增值}
10:     end for
11:   end for
12:   F_k = {c | c∈C_k ∧ σ(c) ≥ N × minsup}     {提取频繁 k-项集}
13: until F_k = ∅
14: Result = ∪F_k
```

候选剪枝 Candidate Pruning

F_3 = {ABC,ABD,ABE,ACD,BCD,BDE,CDE} 是频繁3-项集的集合

C_4 = {ABCD,ABCE,ABDE} 是生成的频繁4-项集的集合 (from previous slide)

候选剪枝 Candidate pruning

- Prune ABCE because ACE and BCE are infrequent
- Prune ABDE because ADE is infrequent

候选剪枝后 : C_4 = {ABCD}

候选项集产生过程/方法

要求：

- 避免产生太多不必要的候选
- 确保候选项集的集合是完全的
- 不产生重复候选项集

方法：

- 蛮力方法（复杂度高）
- $F_{k-1} \times F_{k-1}$ 方法（自行了解）
  - 用其他频繁项来扩展每个频繁（k-1）项集：例如用频繁1-项集扩展频繁2-项集产生频繁3-项集

支持度计数：Support Counting of Candidate Itemsets

扫描交易事务（transaction）数据库以确定每个候选项集的支持度

- 必须将每个候选项目集与每个交易事务匹配，这是一项高成本的操作

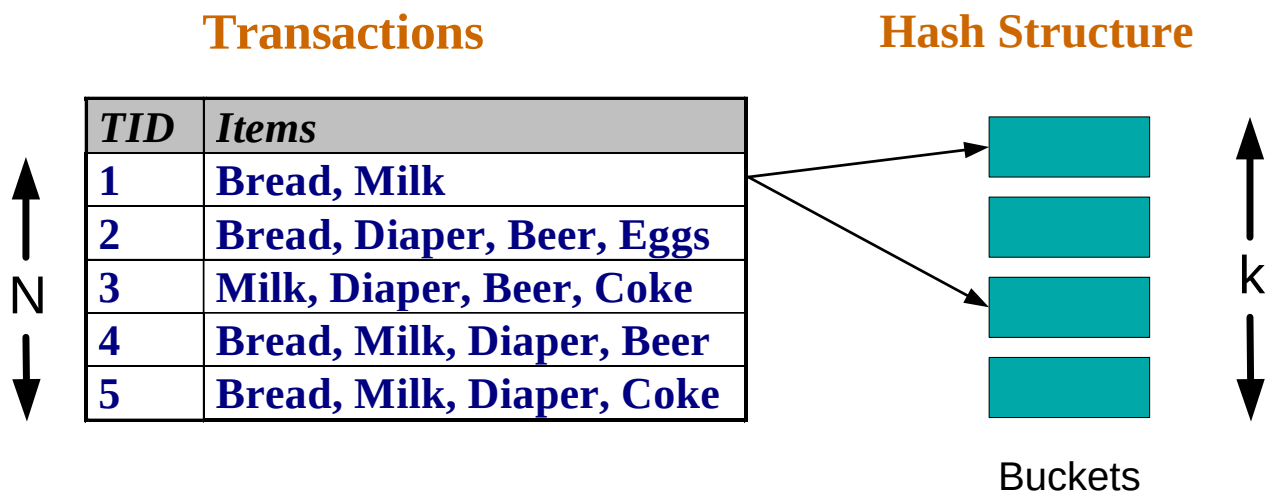
TID	*Items*
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

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

支持度计数：Support Counting of Candidate Itemsets

为了，将候选项目集存储在哈希结构（hash structure）中，以减少比较次数

- 与其将每个交易事务与每个候选者进行匹配，不如将其与哈希存储桶（hashed buckets）中包含的候选者进行匹配。

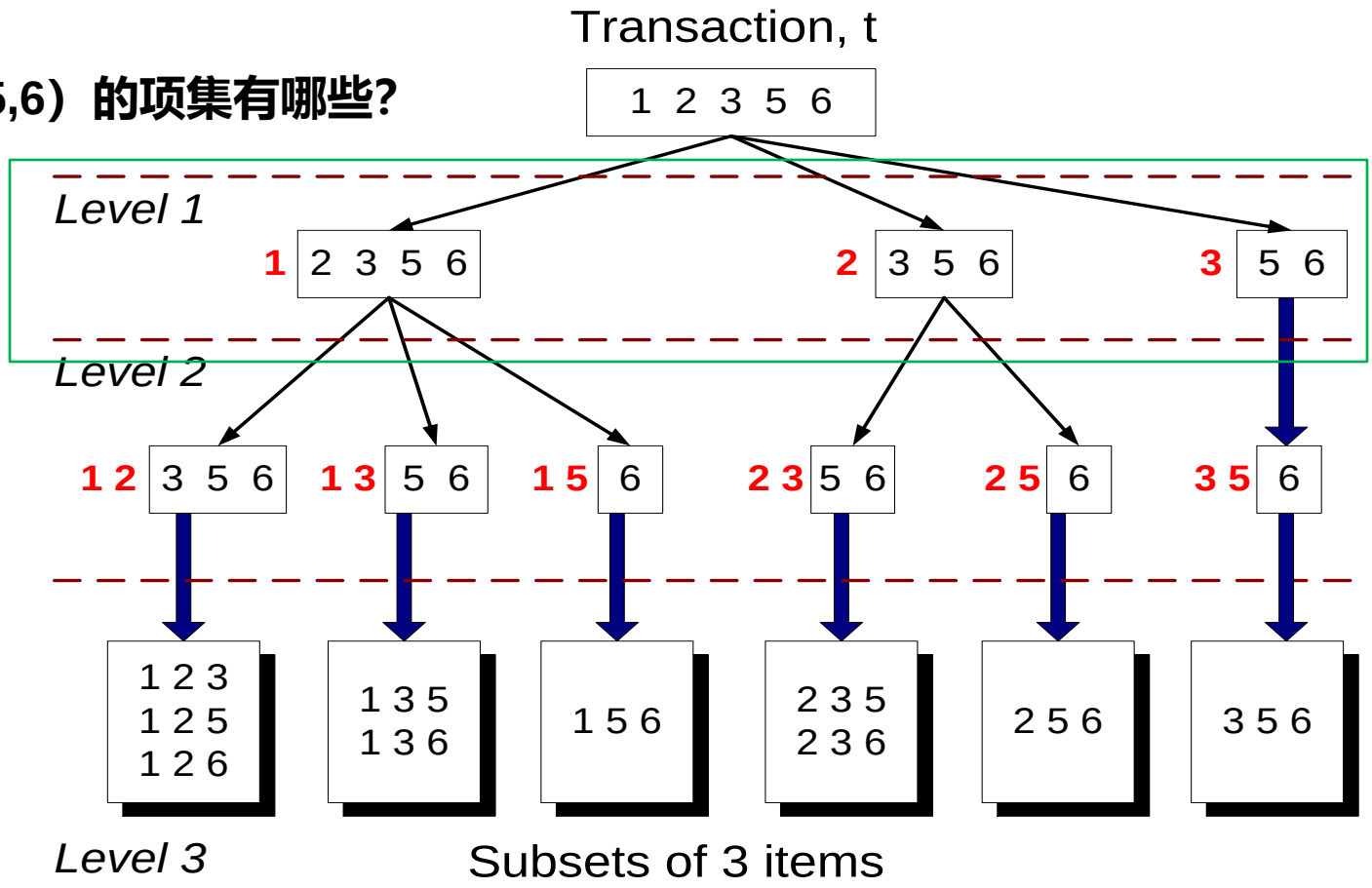


Support Counting: An Example

假设有15个长度为3的候选项集:

{1 4 5}, {1 2 4}, {4 5 7}, {1 2 5}, {4 5 8}, {1 5 9}, {1 3 6}, {2 3 4}, {5 6 7}, {3 4 5}, {3 5 6}, {3 5 7}, {6 8 9}, {3 6 7}, {3 6 8}

支持事务t=（1,2,3,5,6）的项集有哪些?

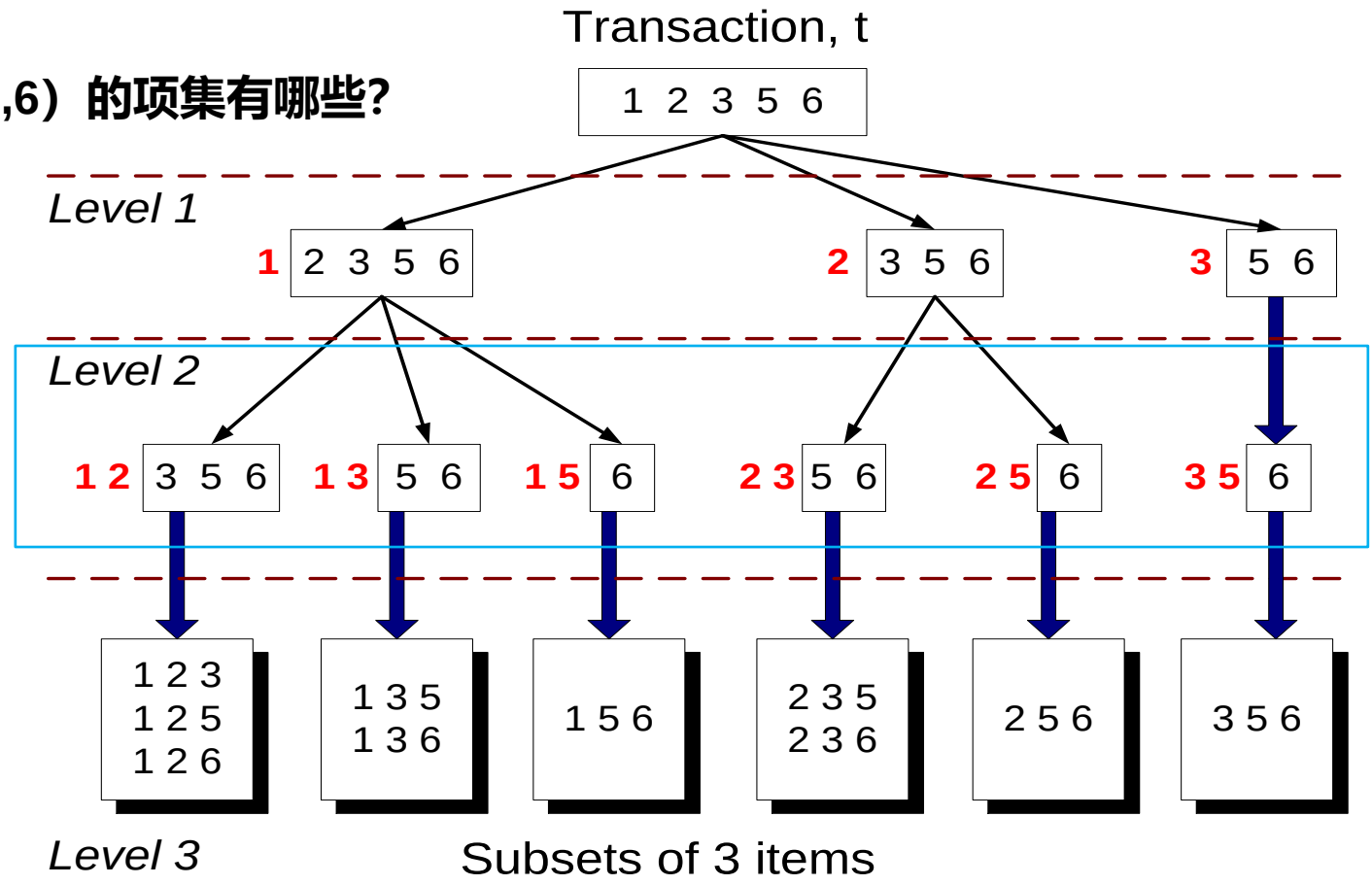


Support Counting: An Example

假设有15个长度为3的候选项集:

{1 4 5}, {1 2 4}, {4 5 7}, {1 2 5}, {4 5 8}, {1 5 9}, {1 3 6}, {2 3 4}, {5 6 7}, {3 4 5}, {3 5 6}, {3 5 7}, {6 8 9}, {3 6 7}, {3 6 8}

支持事务t=（1,2,3,5,6）的项集有哪些?



使用Hash树进行支持度计数
Support Counting Using a Hash Tree

假设有15个长度为3的候选项集:

{1 4 5}, {1 2 4}, {4 5 7}, {1 2 5}, {4 5 8}, {1 5 9}, {1 3 6}, {2 3 4}, {5 6 7}, {3 4 5}, {3 5 6}, {3 5 7}, {6 8 9}, {3 6 7}, {3 6 8}

需要:

- **哈希函数 Hash function**
- **最大叶大小（Max leaf size）：单个叶节点中存储的最大项集数量（如果候选项集的数量超过最大叶子大小，需划分节点）**

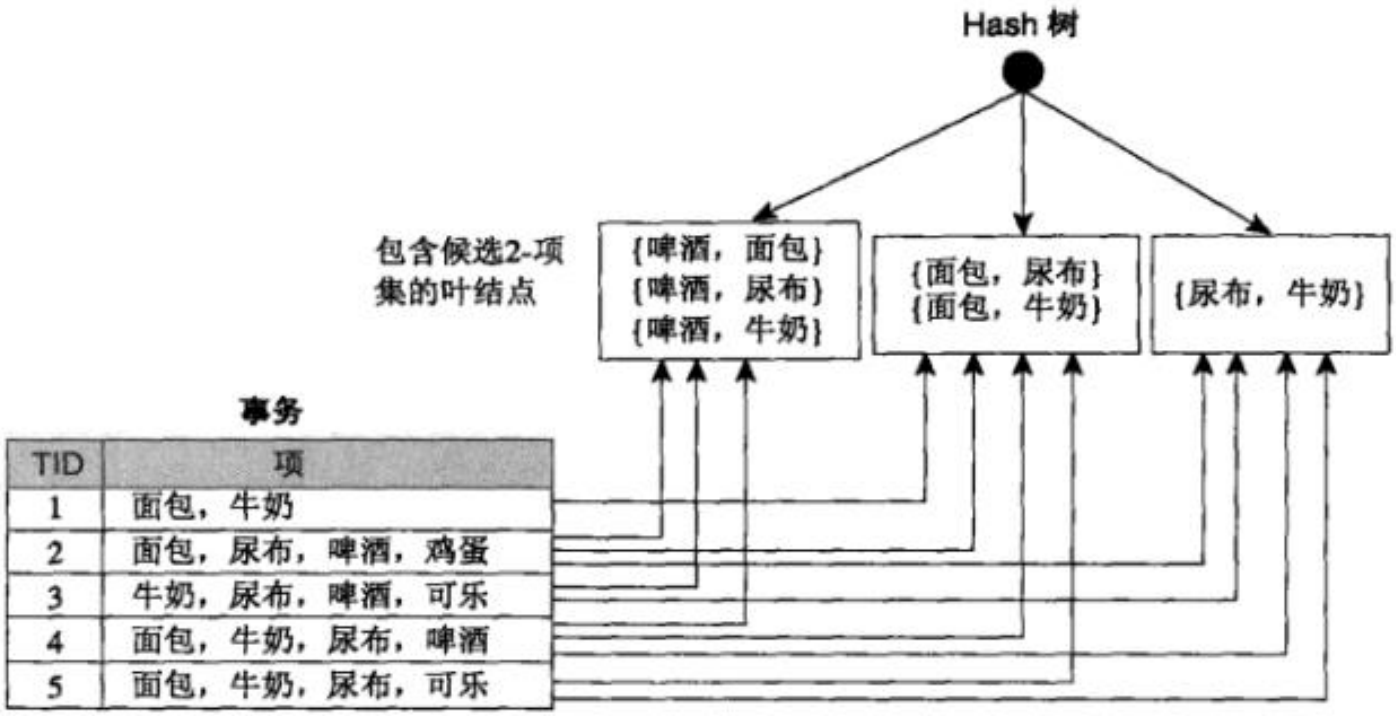
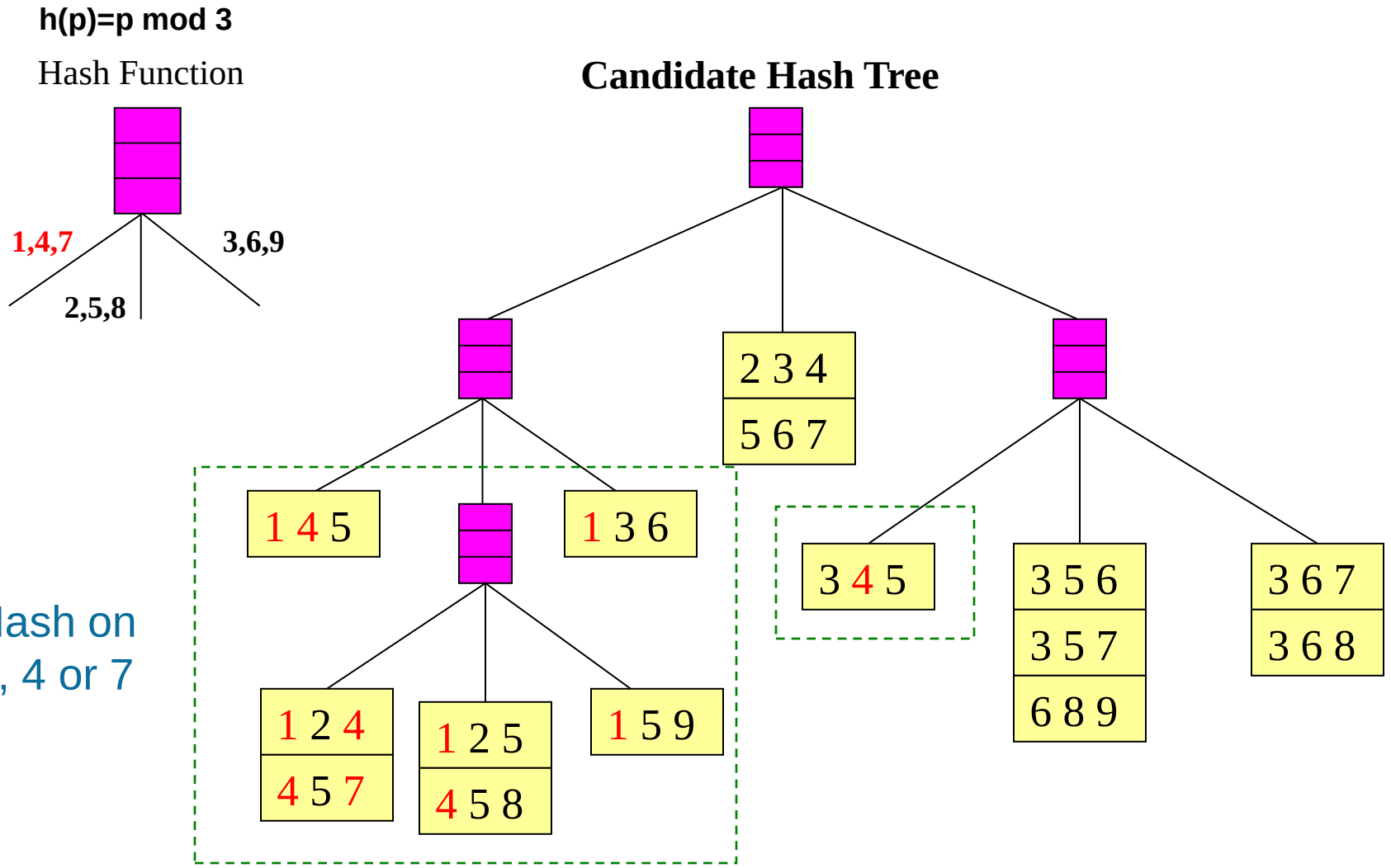
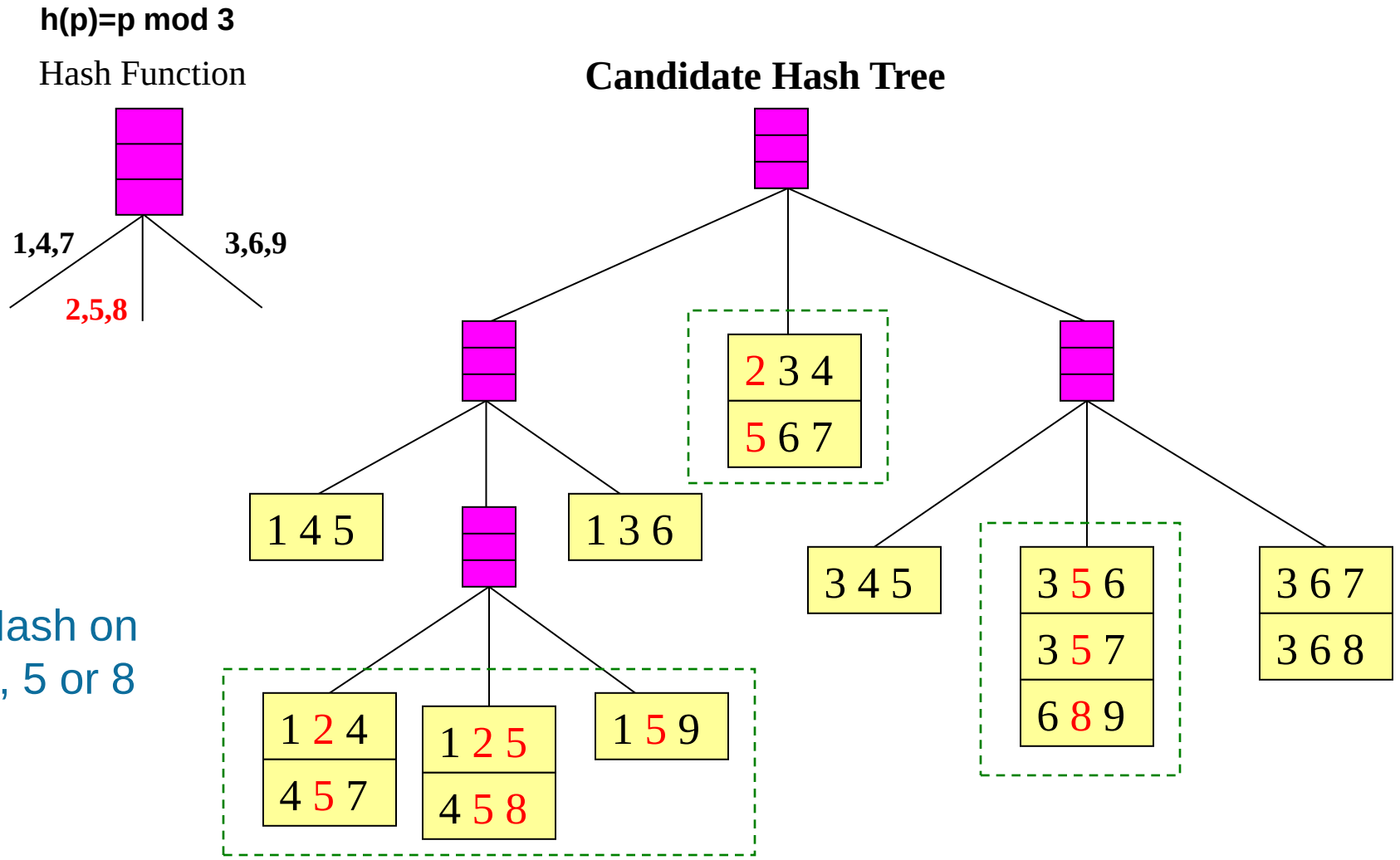


图 6-10　使用 Hash 树结构的项集支持度计数

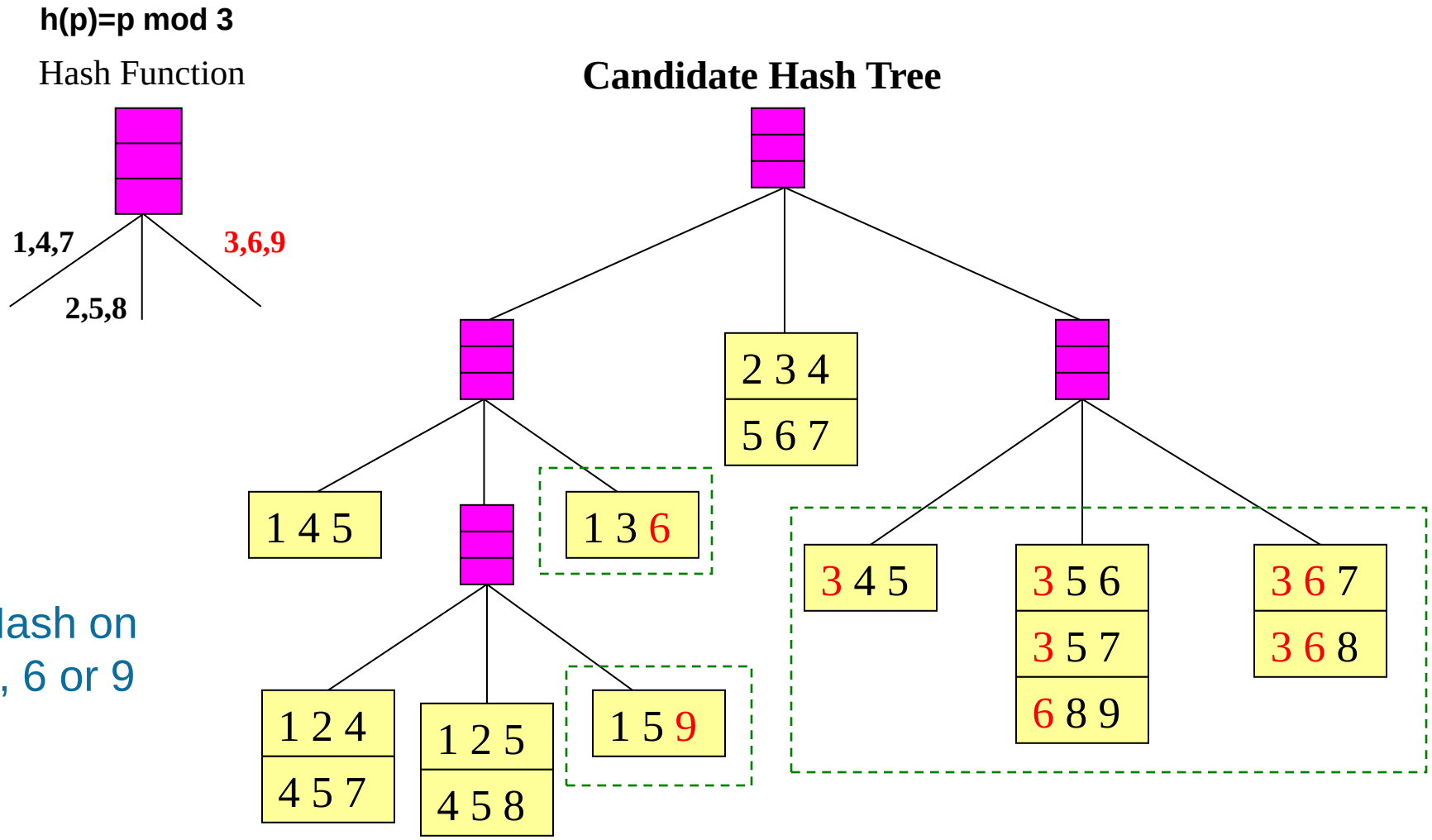
Support Counting Using a Hash Tree



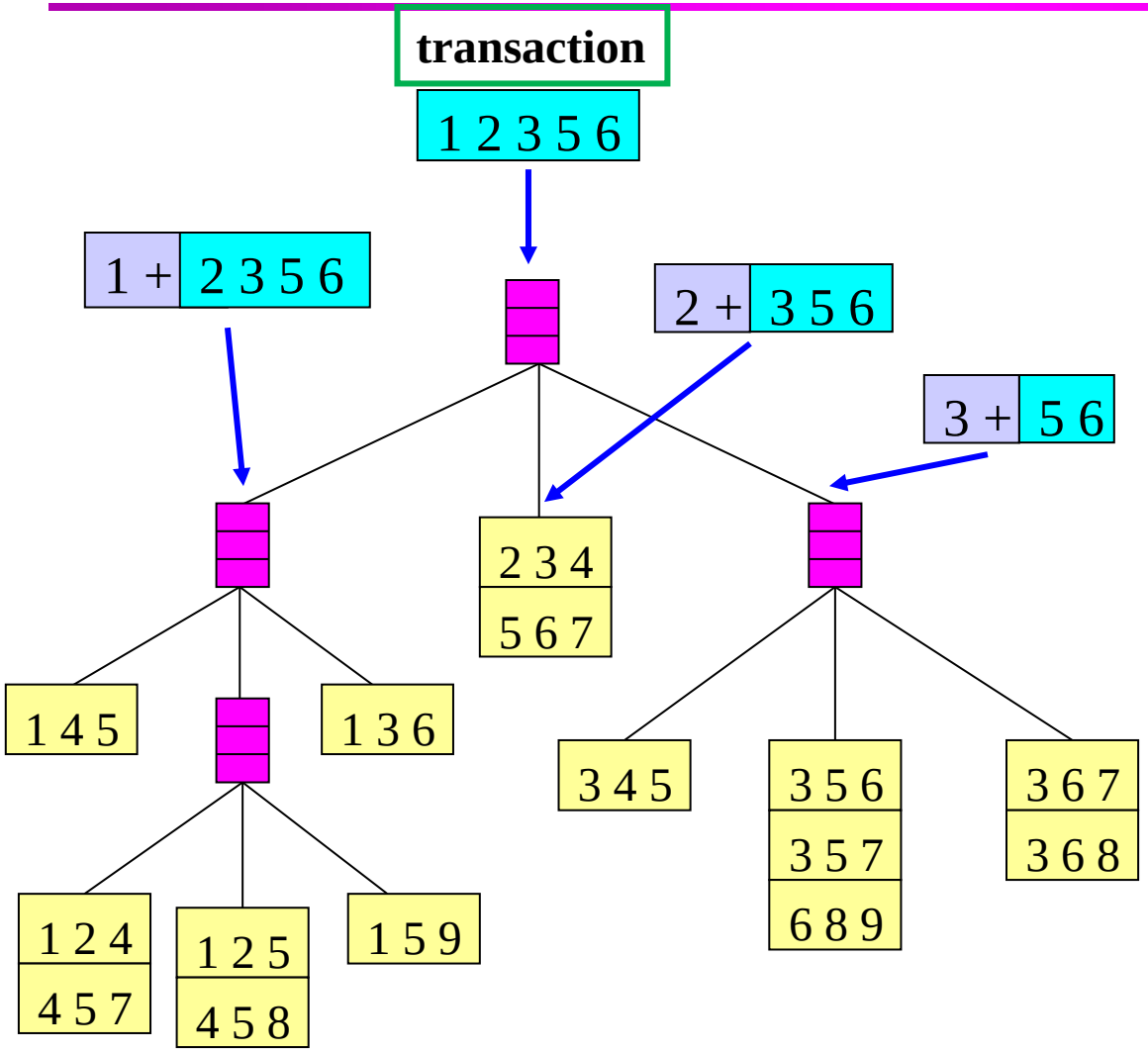
Support Counting Using a Hash Tree

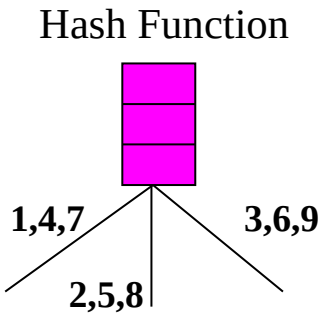


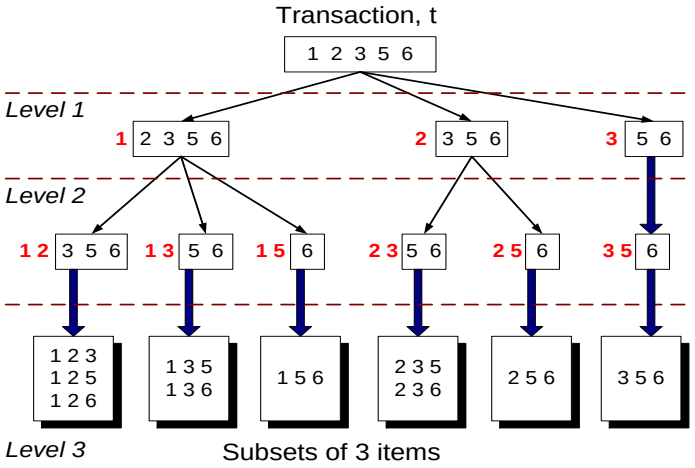
Support Counting Using a Hash Tree



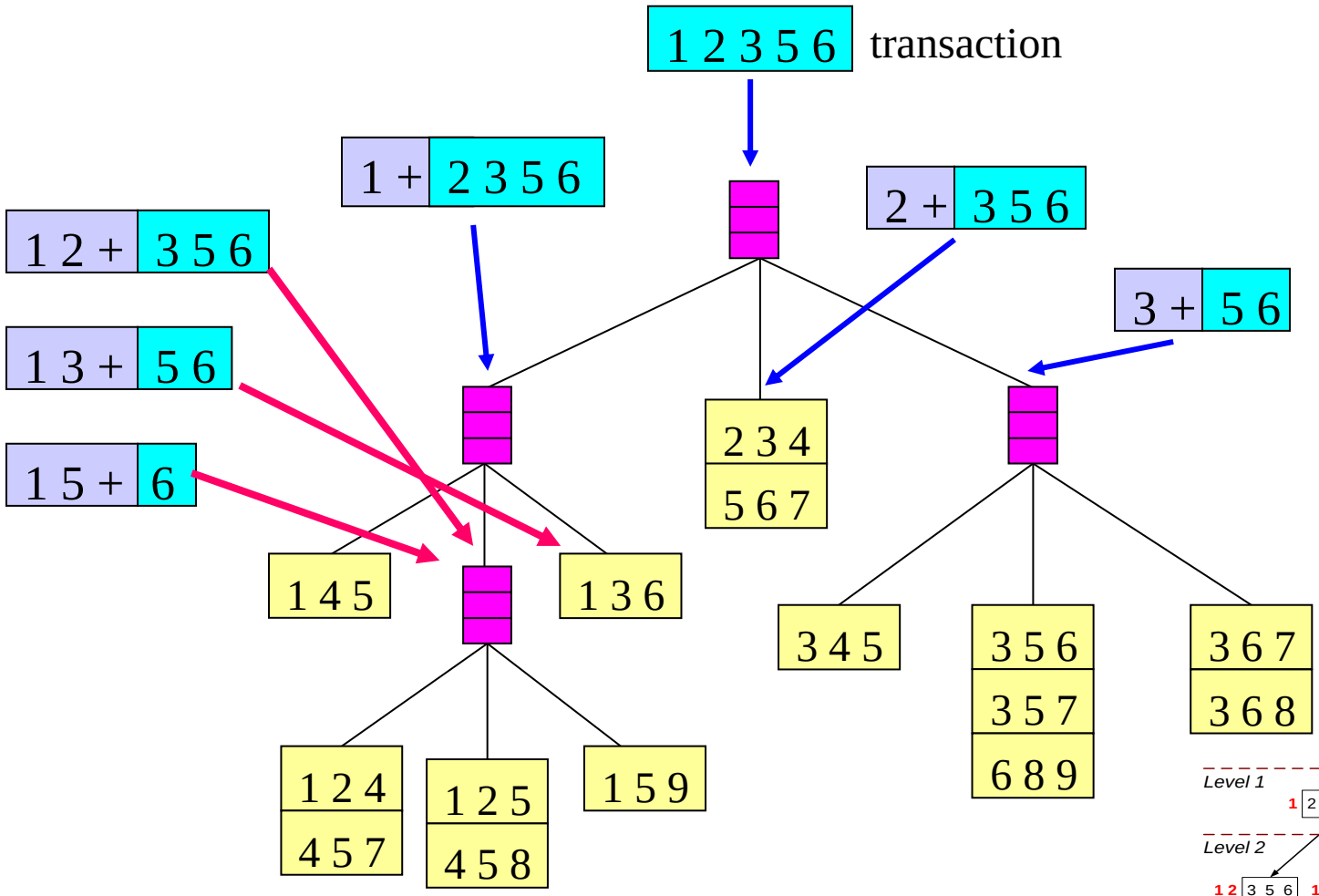
Support Counting Using a Hash Tree

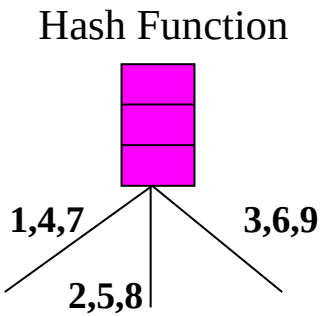


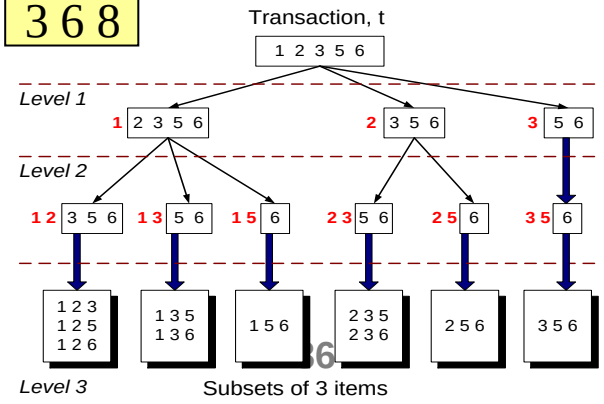




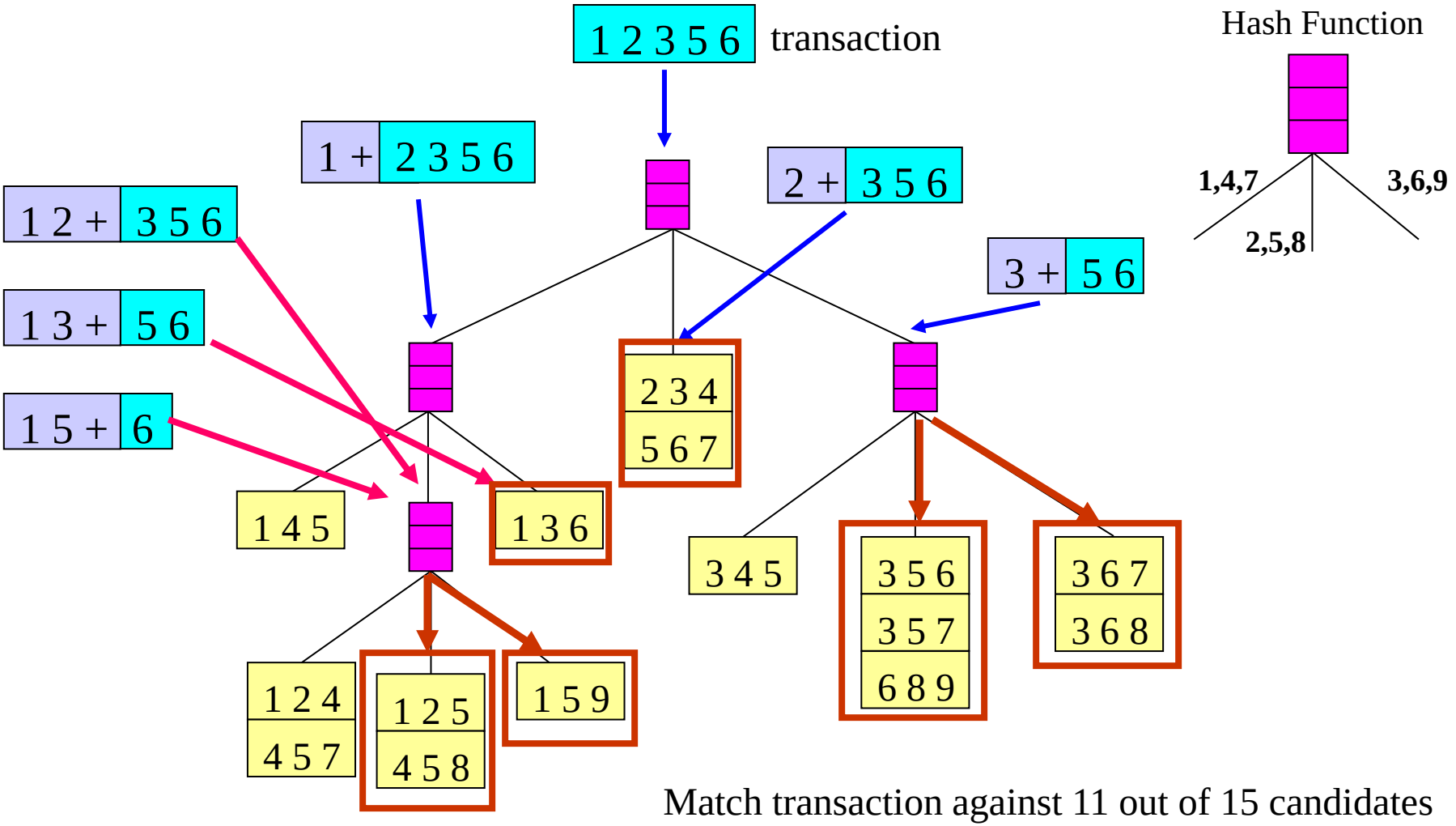
Support Counting Using a Hash Tree







Support Counting Using a Hash Tree



Match transaction against 11 out of 15 candidates

规则生成 Rule Generation

给定频繁项集 L,找到所有非空子集 f ⊂ L，其中f → L – f 满足最小置信度的要求

- 如果 {A,B,C,D} 是一个频繁项集, 候选规则如下:

ABC →D,	ABD →C,	ACD →B,	BCD →A,
A →BCD,	B →ACD,	C →ABD,	D →ABC
AB →CD,	AC → BD,	AD → BC,	BC →AD,
BD →AC,	CD →AB,		

如果 |L| = k, 那么共有 2^k – 2 条候选规则 (忽略 L → ∅ 和 ∅ → L)

规则生成 Rule Generation

通常，置信度不具有反单调性（anti-monotone）的属性

c(ABC →D) 可以大于或者小于 c(AB →D)

但是从同一项目集生成的规则的置信度具有反单调属性

- E.g., 假设 {A,B,C,D} 是一个频繁4-项集:

$$c(ABC \rightarrow D) \geq c(AB \rightarrow CD) \geq c(A \rightarrow BCD)$$

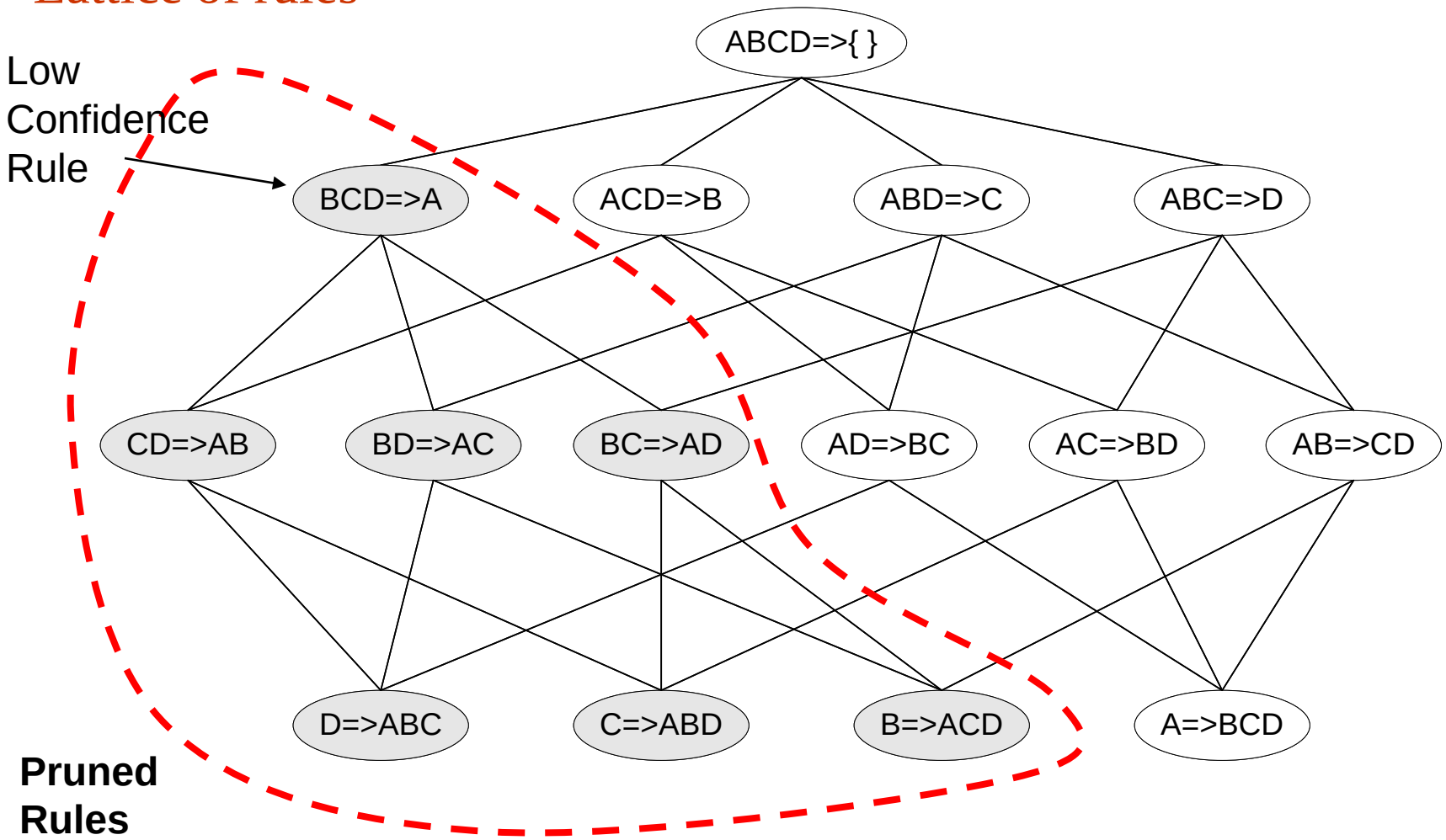
置信度是反单调的（关于规则的RHS上的项目数）

定理 6.2 如果规则 $X \rightarrow Y - X$ 不满足置信度阈值，则形如 $X' \rightarrow Y - X'$ 的规则一定也不满足置信度阈值，其中 X' 是 X 的子集。

为了证明该定理，考虑如下两个规则：$X' \rightarrow Y - X'$ 和 $X \rightarrow Y - X$，其中 $X' \subset X$。这两个规则的置信度分别为 $\sigma(Y) / \sigma(X')$ 和 $\sigma(Y) / \sigma(X)$。由于 X' 是 X 的子集，所以 $\sigma(X') \geq \sigma(X)$。因此，前一个规则的置信度不可能大于后一个规则。

Rule Generation for Apriori Algorithm

Lattice of rules



频繁项集的紧凑表示
Compact Representation of Frequent Itemsets

一些项集是多余的，因为它们与它们的超集具有相同的支持度

TID	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1

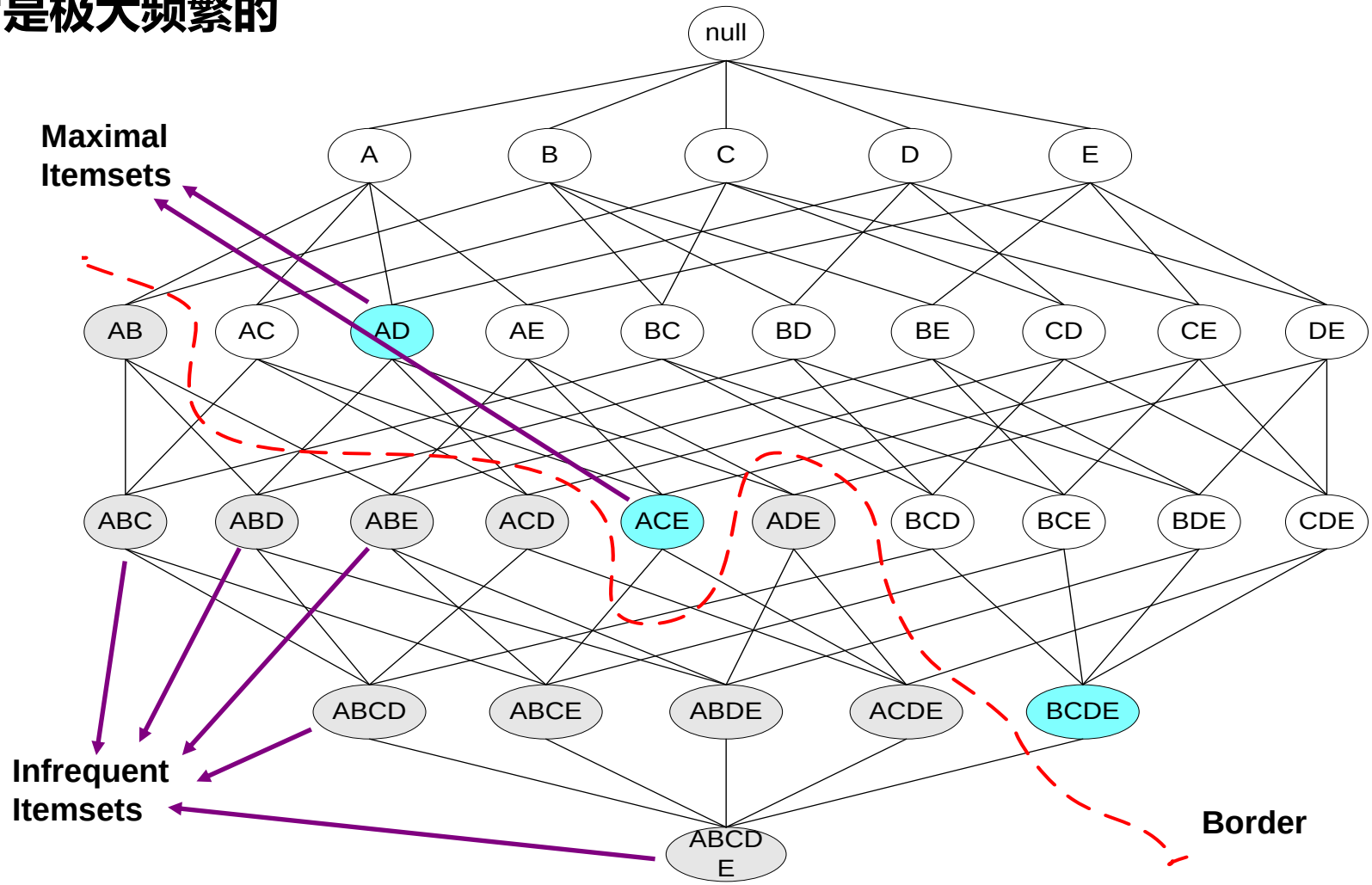
频繁项集数

$$= 3 \times \sum_{k=1}^{10} \binom{10}{k}$$

需要一个紧凑表示（compact representation）

极大频繁项集 Maximal Frequent Itemset

如果某项集很频繁且其直接超集（immediate supersets）都不频繁，则它是极大频繁的



闭合项集 Closed Itemset

如果项集X的所有直接超集（immediate supersets）的支持度计数（support count）与项集X均不相同，则X是闭合的（closed）

如果X的直接超集中，至少有一个与X的支持度计数相同，则X不是闭合的.

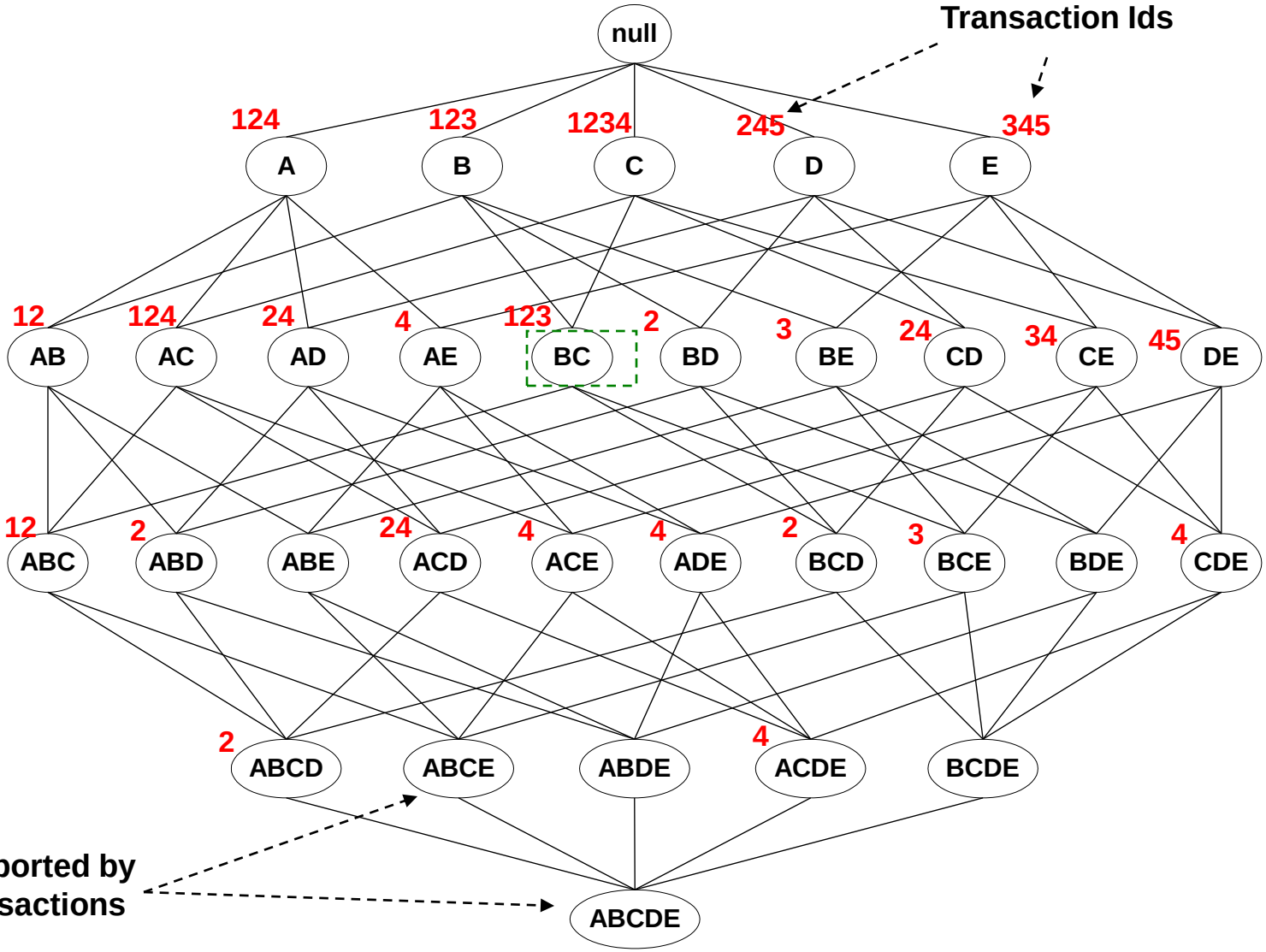
TID	Items
1	{A,B}
2	{B,C,D}
3	{A,B,C,D}
4	{A,B,D}
5	{A,B,C,D}

Itemset	Support
{A}	4
{B}	5
{C}	3
{D}	4
{A,B}	4
{A,C}	2
{A,D}	3
{B,C}	3
{B,D}	4
{C,D}	3

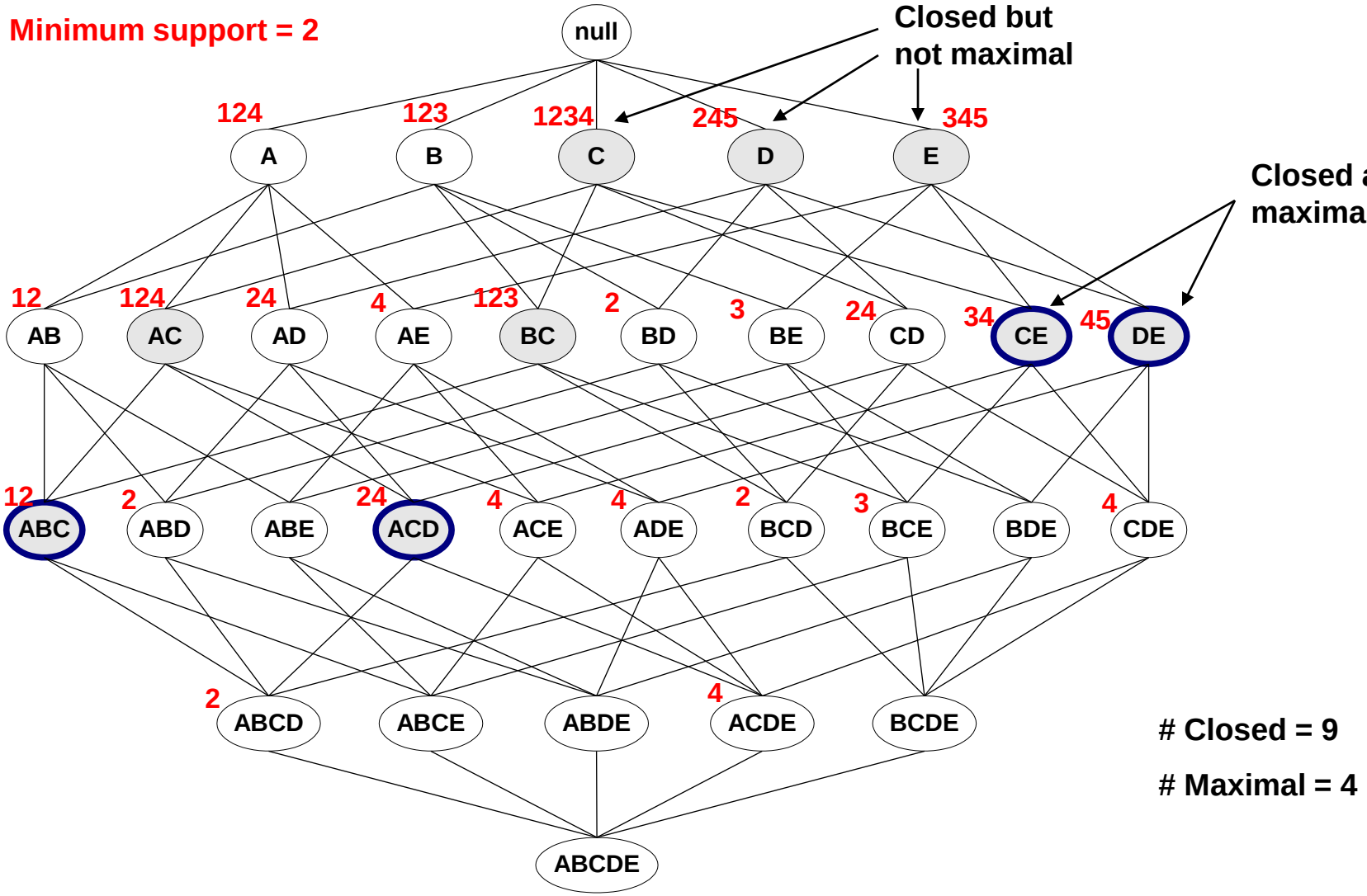
Itemset	Support
{A,B,C}	2
{A,B,D}	3
{A,C,D}	2
{B,C,D}	2
{A,B,C,D}	2

极大/闭合项集 Maximal vs Closed Itemsets

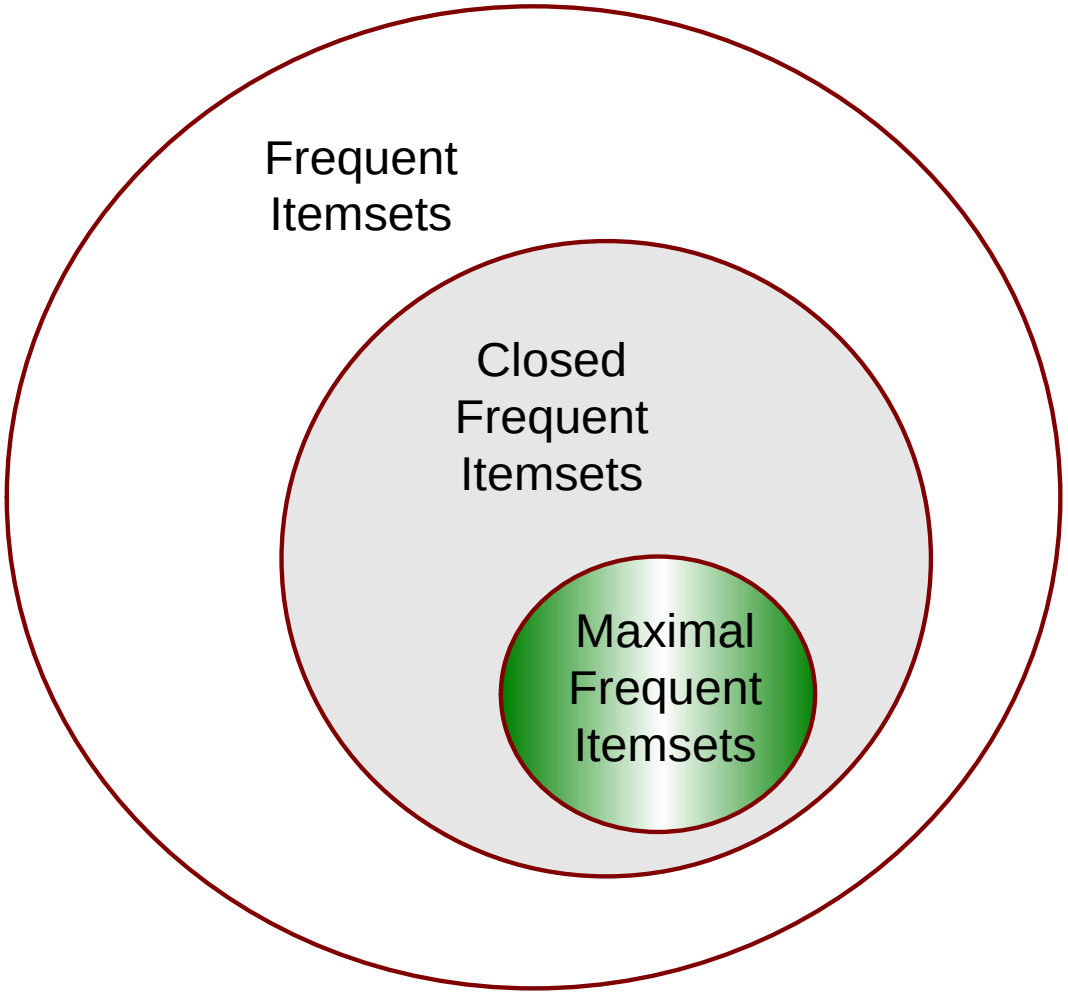
TID	Items
1	ABC
2	ABCD
3	BCE
4	ACDE
5	DE



极大/闭合项集 Maximal vs Closed Itemsets



极大/闭合项集 Maximal vs Closed Itemsets



模式评估 Pattern Evaluation

关联规则算法可以产生大量规则

可以使用兴趣度（Interestingness）度量来对模式（pattern）进行修剪或

- 客观兴趣度
  - 在最初的公式中，支持度（support）和置信度（confidence）是唯一采用的措施
- 主观兴趣度

计算客观兴趣度指标 Computing Interestingness Measure

给定 X → Y 或者 {X,Y},可从相依表（contingency table）中获得计算兴趣度所需的信息

Contingency table

	Y	$\overline{Y}$	
X	f_{11}	f_{10}	f_{1+}
$\overline{X}$	f_{01}	f_{00}	f_{o+}
	f_{+1}	f_{+0}	N

f_{11}: support of X and Y
f_{10}: support of $\underline{X}$ and $\overline{Y}$
f_{01}: support of $\underline{X}$ and $\underline{Y}$
f_{00}: support of X and Y

用于定义各种度量指标

support, confidence, Gini, entropy, etc.

置信度的局限性 Drawback of Confidence

Customers	Tea	Coffee	…
C1	0	1	…
C2	1	0	…
C3	1	1	…
C4	1	0	…
…			

	Coffee	$\overline{\text{Coffee}}$	
Tea	15	5	20
$\overline{\text{Tea}}$	75	5	80
	90	10	100

关联规则: Tea → Coffee

Confidence ≅ P(Coffee|Tea) = 15/20 = 0.75

Confidence > 50%, 意味着喝茶的人喝咖啡的可能性更大（相比于不喝咖啡）

所以规则似乎是合理的

置信度的局限性 Drawback of Confidence

	Coffee	$\overline{\text{Coffee}}$	
Tea	15	5	20
$\overline{\text{Tea}}$	75	5	80
	90	10	100

关联规则: Tea $\rightarrow$ Coffee

Confidence= P(Coffee|Tea) = 15/20 = 0.75

但是 P(Coffee) = 0.9, 这意味着知道一个人喝茶会降低该人喝咖啡的可能性!

$\Rightarrow$ 注意P(Coffee|$\overline{\text{Tea}}$) = 75/80 = 0.9375

关联规则的度量 Measure for Association Rules

那么，我们真正想要什么样的规则?

- Confidence(X → Y) 应该足够大
  - 确保购买X的人比不购买X的人更有可能购买Y

- Confidence(X → Y) > support(Y)
  - 否则，规则将具有误导性，因为拥有X项实际上减少了在同一笔交易事务中拥有Y项的机会
  - 是否有任何措施可以捕获此约束?
    - 答：是的。 有很多。

示例: Lift/Interest

	Coffee	$\overline{\text{Coffee}}$	
Tea	15	5	20
$\overline{\text{Tea}}$	75	5	80
	90	10	100

关联规则: Tea → Coffee

Confidence= P(Coffee|Tea) = 0.75

但是 P(Coffee) = 0.9

⇒ Lift = 0.75/0.9= 0.8333 (< 1, 因此是负关联)

Lift or Interest

	Y	$\bar{Y}$	
X	10	0	10
$\bar{X}$	0	90	90
	10	90	100

$$Lift = \frac{0.1}{(0.1)(0.1)} = 10$$

	Y	$\bar{Y}$	
X	90	0	90
$\bar{X}$	0	10	10
	90	10	100

$$Lift = \frac{0.9}{(0.9)(0.9)} = 1.11$$

统计独立 Statistical independence:

如果 P(X,Y)=P(X)P(Y) => Lift = 1

已有文献中提出了许多措施（自学）

#	Measure	Formula
1	ϕ-coefficient	$\frac{P(A,B)-P(A)P(B)}{\sqrt{P(A)P(B)(1-P(A))(1-P(B))}}$
2	Goodman-Kruskal's (λ)	$\frac{\sum_j \max_k P(A_j,B_k)+\sum_k \max_j P(A_j,B_k)-\max_j P(A_j)-\max_k P(B_k)}{2-\max_j P(A_j)-\max_k P(B_k)}$
3	Odds ratio (α)	$\frac{P(A,B)P(\overline{A},\overline{B})}{P(A,\overline{B})P(\overline{A},B)}$
4	Yule's Q	$\frac{P(A,B)P(\overline{AB})-P(A,\overline{B})P(\overline{A},B)}{P(A,B)P(\overline{AB})+P(A,\overline{B})P(\overline{A},B)} = \frac{\alpha-1}{\alpha+1}$
5	Yule's Y	$\frac{\sqrt{P(A,B)P(\overline{AB})}-\sqrt{P(A,\overline{B})P(\overline{A},B)}}{\sqrt{P(A,B)P(\overline{AB})}+\sqrt{P(A,\overline{B})P(\overline{A},B)}} = \frac{\sqrt{\alpha}-1}{\sqrt{\alpha}+1}$
6	Kappa (κ)	$\frac{P(A,B)+P(\overline{A},\overline{B})-P(A)P(B)-P(\overline{A})P(\overline{B})}{1-P(A)P(B)-P(\overline{A})P(\overline{B})}$
7	Mutual Information (M)	$\frac{\sum_i \sum_j P(A_i,B_j)\log\frac{P(A_i,B_j)}{P(A_i)P(B_j)}}{\min(-\sum_i P(A_i)\log P(A_i), -\sum_j P(B_j)\log P(B_j))}$
8	J-Measure (J)	$\max\left(P(A,B)\log(\frac{P(B\|A)}{P(B)})+P(A\overline{B})\log(\frac{P(\overline{B}\|A)}{P(\overline{B})}),\right.$ $\left.P(A,B)\log(\frac{P(A\|B)}{P(A)})+P(\overline{A}B)\log(\frac{P(\overline{A}\|B)}{P(\overline{A})})\right)$
9	Gini index (G)	$\max\Big(P(A)[P(B\|A)^2+P(\overline{B}\|A)^2]+P(\overline{A})[P(B\|\overline{A})^2+P(\overline{B}\|\overline{A})^2]$ $-P(B)^2-P(\overline{B})^2,$ $P(B)[P(A\|B)^2+P(\overline{A}\|B)^2]+P(\overline{B})[P(A\|\overline{B})^2+P(\overline{A}\|\overline{B})^2]$ $-P(A)^2-P(\overline{A})^2\Big)$
10	Support (s)	$P(A,B)$
11	Confidence (c)	$\max(P(B\|A),P(A\|B))$
12	Laplace (L)	$\max\left(\frac{NP(A,B)+1}{NP(A)+2},\frac{NP(A,B)+1}{NP(B)+2}\right)$
13	Conviction (V)	$\max\left(\frac{P(A)P(\overline{B})}{P(A\overline{B})},\frac{P(B)P(\overline{A})}{P(B\overline{A})}\right)$
14	Interest (I)	$\frac{P(A,B)}{P(A)P(B)}$
15	cosine (IS)	$\frac{P(A,B)}{\sqrt{P(A)P(B)}}$
16	Piatetsky-Shapiro's (PS)	$P(A,B)-P(A)P(B)$
17	Certainty factor (F)	$\max\left(\frac{P(B\|A)-P(B)}{1-P(B)},\frac{P(A\|B)-P(A)}{1-P(A)}\right)$
18	Added Value (AV)	$\max(P(B\|A)-P(B),P(A\|B)-P(A))$
19	Collective strength (S)	$\frac{P(A,B)+P(\overline{AB})}{P(A)P(B)+P(\overline{A})P(\overline{B})}\times\frac{1-P(A)P(B)-P(\overline{A})P(\overline{B})}{1-P(A,B)-P(\overline{AB})}$
20	Jaccard (ζ)	$\frac{P(A,B)}{P(A)+P(B)-P(A,B)}$
21	Klosgen (K)	$\sqrt{P(A,B)}\max(P(B\|A)-P(B),P(A\|B)-P(A))$

比较不同的度量 Comparing Different Measures

列联表（contingency tables）的10个示例:

Example	f_{11}	f_{10}	f_{01}	f_{00}
E1	8123	83	424	1370
E2	8330	2	622	1046
E3	9481	94	127	298
E4	3954	3080	5	2961
E5	2886	1363	1320	4431
E6	1500	2000	500	6000
E7	4000	2000	1000	3000
E8	4000	2000	2000	2000
E9	1720	7121	5	1154
E10	61	2483	4	7452

使用各种方法对列联表排序:

#	ϕ	λ	α	Q	Y	κ	M	J	G	s	c	L	V	I	IS	PS	F	AV	S	ζ	K
E1	1	1	3	3	3	1	2	2	1	3	5	5	4	6	2	2	4	6	1	2	5
E2	2	2	1	1	1	2	1	3	2	2	1	1	1	8	3	5	1	8	2	3	6
E3	3	3	4	4	4	3	3	8	7	1	4	4	6	10	1	8	6	10	3	1	10
E4	4	7	2	2	2	5	4	1	3	6	2	2	2	4	4	1	2	3	4	5	1
E5	5	4	8	8	8	4	7	5	4	7	9	9	9	3	6	3	9	4	5	6	3
E6	6	6	7	7	7	7	6	4	6	9	8	8	7	2	8	6	7	2	7	8	2
E7	7	5	9	9	9	6	8	6	5	4	7	7	8	5	5	4	8	5	6	4	4
E8	8	9	10	10	10	8	10	10	8	4	10	10	10	9	7	7	10	9	8	7	9
E9	9	9	5	5	5	9	9	7	9	8	3	3	3	7	9	9	3	7	9	9	8
E10	10	8	6	6	6	10	5	9	10	10	6	6	5	1	10	10	5	1	10	10	7

辛普森悖论 Simpson's Paradox

Buy HDTV	Buy Exercise Machine		
	Yes	No	
Yes	99	81	180
No	54	66	120
	153	147	300

$$c(\{\text{HDTV} = \text{Yes}\} \rightarrow \{\text{Exercise Machine} = \text{Yes}\}) = 99/180 = 55\%$$

$$c(\{\text{HDTV} = \text{No}\} \rightarrow \{\text{Exercise Machine} = \text{Yes}\}) = 54/120 = 45\%$$

=>购买HDTV的客户更有可能购买健身器材?

辛普森悖论 Simpson's Paradox

Customer Group	Buy HDTV	Buy Exercise Machine		Total
		Yes	No	
College Students	Yes	1	9	10
	No	4	30	34
Working Adult	Yes	98	72	170
	No	50	36	86

大学生:

$$c(\{\text{HDTV} = \text{Yes}\} \rightarrow \{\text{Exercise Machine} = \text{Yes}\}) = 1/10 = 10\%$$

$$c(\{\text{HDTV} = \text{No}\} \rightarrow \{\text{Exercise Machine} = \text{Yes}\}) = 4/34 = 11.8\%$$

上班族:

$$c(\{\text{HDTV} = \text{Yes}\} \rightarrow \{\text{Exercise Machine} = \text{Yes}\}) = 98/170 = 57.7\%$$

$$c(\{\text{HDTV} = \text{No}\} \rightarrow \{\text{Exercise Machine} = \text{Yes}\}) = 50/86 = 58.1\%$$

辛普森悖论 Simpson's Paradox

观察到的数据关系可能会受到其他混杂因素（隐藏变量）的影响

- 隐藏的变量可能导致观察到的关系消失或反转其方向！

需要进行适当的分层（stratification）以避免产生伪模式（spurious patterns）

总结

有效地挖掘关联规则

- 质量
- 支持度、置信度……
- 先得到频繁项集，再根据频繁项集得到关联规则

高效地挖掘关联规则

- 效率
- 提高获得频繁项集的效率
  - 支持度计算
- 提高关联规则挖掘的效率
  - 置信度计算

谢谢！

数据挖掘
教师：王东京
学院：计算机学院
邮箱：dongjing.wang@hdu.edu.cn