中国科学技术大学
University of Science and Technology of China

数据挖掘与数据仓库

第四讲　模式挖掘

October 26, 2017

Outline

知识表示

- 用计算机可接受的方式把知识描述出来，于人工智能的作用，相当于数据结构作用于程序设计
- 命题逻辑/谓词逻辑，“P→Q”表示“如果 P，那么 Q”，P,Q 是两个命题
- 产生式知识表示，简单看，就是“如果 P，那么 Q”的表述结构，可以简写成 P→Q

P→Q 的例子

- 天下雨，地上湿；“原因 → 结果”100%
- 如果把水加热到 0° 以上，冰就会溶化为水；“条件 → 结论”100%
- 夜来风雨声，花落知多少；“事实 → 进展”90%
- 若能找到一根合适的杠杆，就能撬起那座大山；“前提 → 操作”100%
- 才饮长沙水，又食武昌鱼；“事实 → 进展”10%
- 刚才开机了，意味着发出了捕获目标图像的信号。“情况 → 行为”90%

P→Q 在数据库中的例子

- 给定关系模式 student(sno,sname,dept,ssex,sage)，其中有函数依赖 sno→sname, sno→ssex 等，100%
- 数据仓库中，有维经过映射，得到聚集的“度量”，可以类似描述成 (time,location)→dollars_sold, 100%
- 相关性分析，属性 A, B 相关吗？χ^2 检验，相关系数，协方差分析等，最终结论可以写成 A→B, 0-100%

现实中的困惑：你能找到 P→Q 吗？

- 相关性分析，OK，方法之一
- 函数依赖分析的方法，OK，成熟的套路
- 数据仓库中的“度量”，OK，研究很多
- 更广阔范围内的这种知识呢，如何寻找？一般性方法？
- 尤其涉及到红色的百分比，置信水平（这东西让人有多么信任呢？）如何确定/计算？
- 与置信水平相关的一个新的概念：频繁模式，一个频繁出现的对象（标量值/向量值），越频繁，越能让人相信这个对象是“对的”，这是直观的理解

给一个超市销售记录

- 属性是“购买商品件数”，一个大于 0 的整数值
- 问题：它的频繁模式是什么？如何计算？
- 对象是个标量值，利用统计方法，先计算所有件数的频数，即每个“件数”的出现次数
- 归一化/规范化所有的频数到 [0 1]，得到“概率分布”
- 取概率最大的若干“件数”值，这就是所谓的频繁模式
- 所以，我们一般不考虑“标量值”的频繁模式挖掘任务

单个属性（标量值），无法获得属性间的 P→Q

实际多属性频繁模式的例子

- 购物篮分析：啤酒和尿片？
- 跨市场营销：买了 PC 后，接下来会买什么？软件？外设？
- 分类设计：网站中主分类设计？
- 促销分析：如何设计促销活动，提高销售？
- web 点击流分析：用户点击网页链接可以用于改进网页设计吗？web 自动分类？
- DNA 序列分析：哪一种 DNA 对新药敏感？

多个属性间，可能就有某种 $P \rightarrow Q$ 的知识

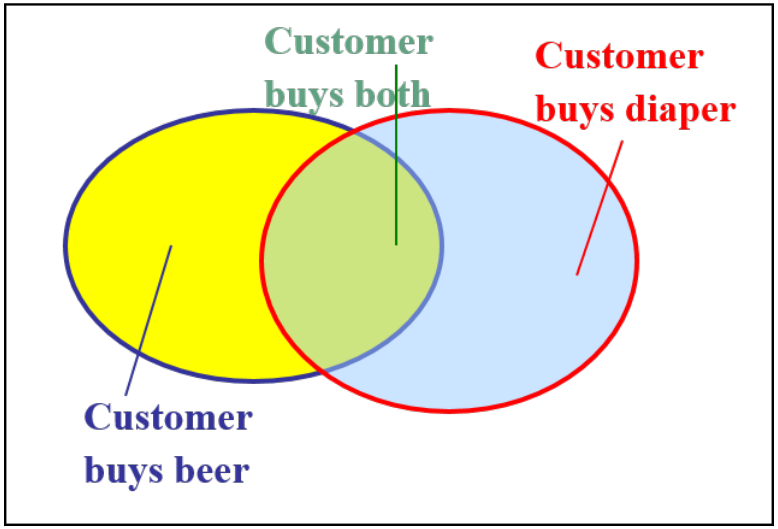
问题复杂性的来源

- 假设共有 n 个属性 $A_1, A_2, \ldots, A_n$
- 若是对单个属性 A_i 找频繁模式，则至多 n 个属性，每个都执行一次统计
- 若对多维属性进行频繁模式挖掘，则可能的有效属性子集有 $2^n - 1$ 个，每个属性子集上都进行频繁模式的挖掘？计算量！
- 若是选择某个属性子集进行，选择那些属性子集，标准？
- 不妨设，要对属性子集 $A_1, A_2, \ldots, A_k$ 进行频繁模式统计时，所有可能的模式（也就是任何一个可能取值的向量）共有 $\prod_{i=1}^{k} |D(A_i)|$，其中 $|D(A_i)|$ 表示属性 A_i 的大小（不同值的个数），模式个数至少是属性维数的指数函数。实际应用中，在进行频数统计时，会出现大量的缺失值或 0。稀疏性！样本数量问题

购物篮分析

TID	*List of item_IDs*
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

(a) 交易记录数据



(b) 顾客集合视图

Figure: 购物篮分析

基本概念

- 总项集/itemset，所有项目/商品/item 构成的集合
- k-项集/k-itemset，包含 k 个项的集合，总项集的 k 元子集，一般简称为项集或 k-项集

项集/k-项集的表示

TID	*List of item_IDs*
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

每条交易记录长短不一
计算机处理较为麻烦
规范化处理
将所有商品都列举出来
1 表示买了，0 表示没买

Beer	Nuts	Diaper	milk	coffee	eggs	...
1	1	1	0	0	0	...
1	0	1	0	1	0	...
1	0	1	0	0	1	...
0	1	1	1	1	1	

假设商品共有 N 种，则

- 总项集就是 N 维全 1 的向量 $(1, 1, \ldots, 1)$
- k-项集就是 N 维向量，其中 1 的个数为 k
- 每个交易记录是一个 k-项集，不同的交易，k 可能是不同的数值

描述“频繁”

Beer	Nuts	Diaper	milk	coffee	eggs	...
1	1	1	0	0	0	...
1	0	1	0	1	0	...
1	0	1	0	0	1	...
0	1	1	1	1	1	

L2：milk和coffee的绝对支持度为1，
beer和milk的绝对支持度为0，
beer和diaper的绝对支持度为3

beer 的绝对支持度为 3，相对支持度为 75%

预定义”频繁”的阈值为 50%

则频繁项集集合 $L_1 = \{\{bear\}, \{diaper\}\}$

则频繁项集集合 $L_2 = \{\{bear, diaper\}\}$

支持度

- 给定一个交易数据表 Table，包含 M 行，将交易数据表 Table 转换为 $M \times N$ 的表，统一表示
- 对给定项集 X，可以定义 X 的支持度，描述 X 的特征
- 绝对支持度：或者叫做支持计数，是指项集 X 在交易表中出现的次数/频度
- 相对支持度：X 出现次数/M，也就是 M 条总交易记录中出现 X 这种项集的“比例”/“概率”/“可能性”
- 定义“频繁”，就是指一个项集，其相对支持度大于预定义的阈值，比如 10%
- 把所有频繁的 k-项集收集起来，放在一起（一个集合中），这个由频繁的 k-项集所构成的集合，记作 L_k

关联规则：改进的产生式知识表示

P→Q 的改进

- 谓词逻辑/命题逻辑/产生式，P→Q，"if P then Q"
- 扩展/改造，成为数据挖掘中的知识表示
- 关联规则：$P \Rightarrow Q[support = u\%; confidence = v\%]$，$P, Q$ 是两个不同的项集
- 关联规则的这个定义，企图回答两个问题：$P \Rightarrow Q$ 出现的频率/可能性有多少 (support)？P 出现了，就一定会有 Q 出现吗，你有多确定这事 (confidence)？

还存在的其它问题

- 在频繁模式中，support 有了，但是 confidence 如何计算？
- 为什么是 $P \Rightarrow Q$，而不是 $Q \Rightarrow P$? 可以是 $Q \Rightarrow P$ 吗？

confidence 的计算

Beer	Nuts	Diaper	milk	coffee	eggs	...
1	1	1	0	0	0	...
1	0	1	0	1	0	...
1	0	1	0	0	1	...
0	1	1	1	1	1	

给定频繁模式 (P, Q)，P, Q 是两个属性

- $confidence(P \Rightarrow Q) = Prob(Q|P) = \frac{support(P \cup Q)}{support(P)}$，条件概率
- $confidence(Q \Rightarrow P) = Prob(P|Q) = \frac{support(P \cup Q)}{support(Q)}$，条件概率
- 例子中：$P = beer, Q = diaper$

  $support(beer) = 3,\ support(diaper) = 4,\ support(beer \cup diaper) = 3,$

  故，两个完整的关联规则：

  $beer \Rightarrow diaper[75\%, 100\%],\ diaper \Rightarrow beer[75\%, 75\%]$

  support(bUd)/M=3/4=75%

挑战：挑选频繁模式

多维频繁模式的发现是关联规则发现的基础

- 有 N 个项/属性（二值）的表，其模式总数是 $2^N - 1$，存储/计算都是不可能的，当 N 较大的时候
- 必须选择其中的某些模式，最简单的做法，扫描一遍这些模式，挑选出满足预定义条件的模式（不可行！不能遍历所有模式）

频繁项集发现算法：L_{k+1} 由 L_k, L_1 扩充而成

Input: DS: 数据集; $support_{min}$: 最小支持度
Output: $L = \{L_1, L_2, \ldots\}$: 频繁项集集合的集合
$k = 1, L \longleftarrow \phi$, scan DS, 得到 L_k;扫描数据集得到属性出现次数，选择超出阈值的属性，得到L1
while $L_k \neq \phi$ **do**
    $L = L \cup \{L_k\}$;
    $L_{k+1} = \phi$;
    foreach x *in* L_1 **do**　　x长度为1，y长度为k，z长度为k+1（y中可能存在x）
        foreach y *in* L_k **do**
            $z \longleftarrow x \cup y$;
            if $\#z == k+1$ *and* $support(z) \geq support_{min}$ **then**
                $L_{k+1} = L_{k+1} \cup \{z\}$;
    $k \longleftarrow k+1$;
return L;

算法评论

- 算法直观，简洁; 但是时间复杂度相当高!
- 其中 if 语句条件判断的时间复杂度是 $O(m(k+1))$，其中 m 是数据集大小（行数/元组数），也就是说近乎于要完整扫描一遍数据集
- 故整个算法可能需要扫描数据集的次数达到 n^2 次，n 是数据集的列数，当 m, n 大时，时间开销太大，需要对算法优化
- 修改两个 foreach 循环，降低扫描数据集的次数

频繁项集发现：改进算法-1

Input: DS: 数据集; $support_{min}$: 最小支持度
Output: $L = \{L_1, L_2, \ldots\}$: 频繁项集集合的集合
;
$k = 1, L \longleftarrow \phi$, scan DS, 得到 L_k;
while $L_k \neq \phi$ **do**
  $L = L \cup \{L_k\}$;
  generate C_{k+1} /*C_{k+1} 是 $k+1$ 项集，不一定是频繁的，等待“频繁”判定的候选集 */;
  compute $support(C_{k+1})$ /*$support(C_{k+1})$ 表示 C_{k+1} 中每个项集的支持数 */;
  $L_{k+1} = \{x | x \in C_{k+1} and\ support(x) > support_{min}\}$;
  $k \longleftarrow k+1$;
return L;

generate C_{k+1}

编号由小变大

Input: L_k: 每个项集有序的 k-项集集合
Output: C_{k+1}: 候选“频繁”的 (k+1)-项集集合
;
第k项不相同
满足前 $k-1$ 项相同的 k-项集合并，产生 (k+1)-项集;
; 会漏掉不频繁项集，但不会漏掉频繁的项集
return $support(C_{k+1})$

compute $support(C_{k+1})$

Input: DS: 数据集，C_{k+1}: 候选“频繁”的 (k+1)-项集集合
Output: C_{k+1} 中每个项集的支持度
;
扫描一遍 DS 的元组，检查它包含的哪些 C_{k+1} 中的项集，
  将被包含的项集的支持度 +1;
return $support(C_{k+1})$

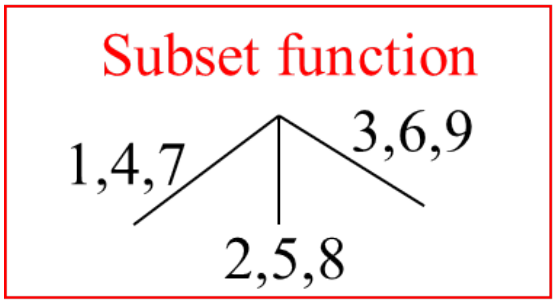
频繁项集发现：改进算法-1

Input: DS: 数据集; $support_{min}$: 最小支持度
Output: $L = \{L_1, L_2, \ldots\}$: 频繁项集集合的集合
;
$k = 1, L \longleftarrow \phi$, scan DS, 得到 L_k;
while $L_k \neq \phi$ **do**
    $L = L \cup \{L_k\}$;
    generate C_{k+1} /*C_{k+1} 是 $k+1$ 项集，不一定是频繁的，等待“频繁”判定的候选集 */;
    compute $support(C_{k+1})$ /*$support(C_{k+1})$ 表示 C_{k+1} 中每个项集的支持数 */;
    $L_{k+1} = \{x | x \in C_{k+1} and\ support(x) > support_{min}\}$;
    $k \longleftarrow k + 1$;
return L;

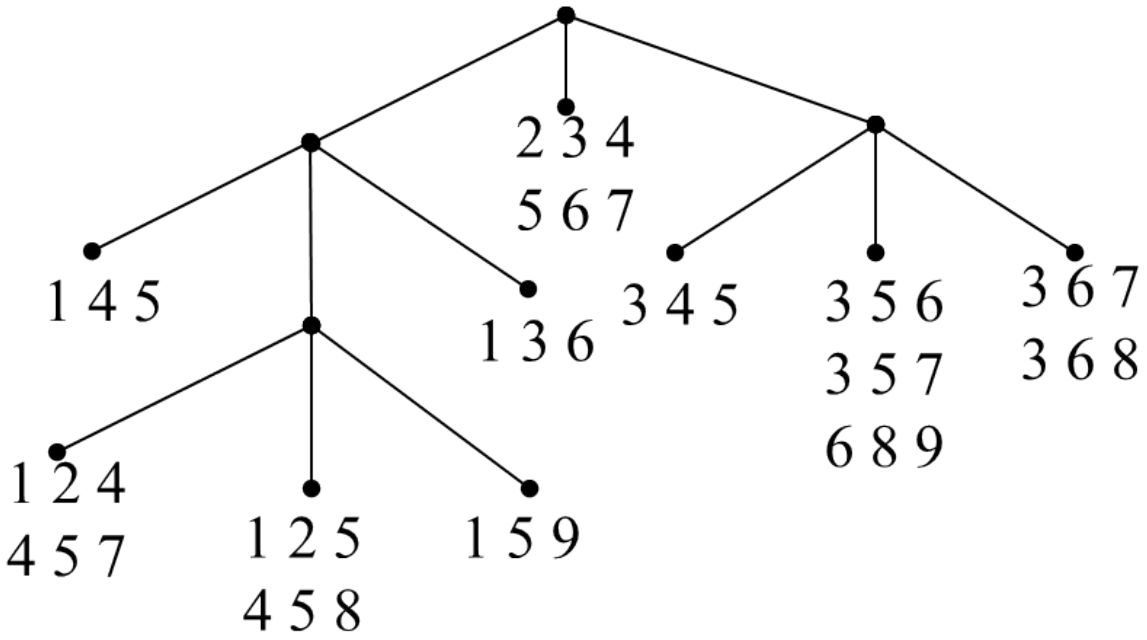
算法评论

- Apriori 算法，proposed by Agrawal & Srikant @VLDB'94。每个 k，扫描一遍数据集。
- C_{k+1} 究竟会有多大？一个元组中会包括多少个候选项集？

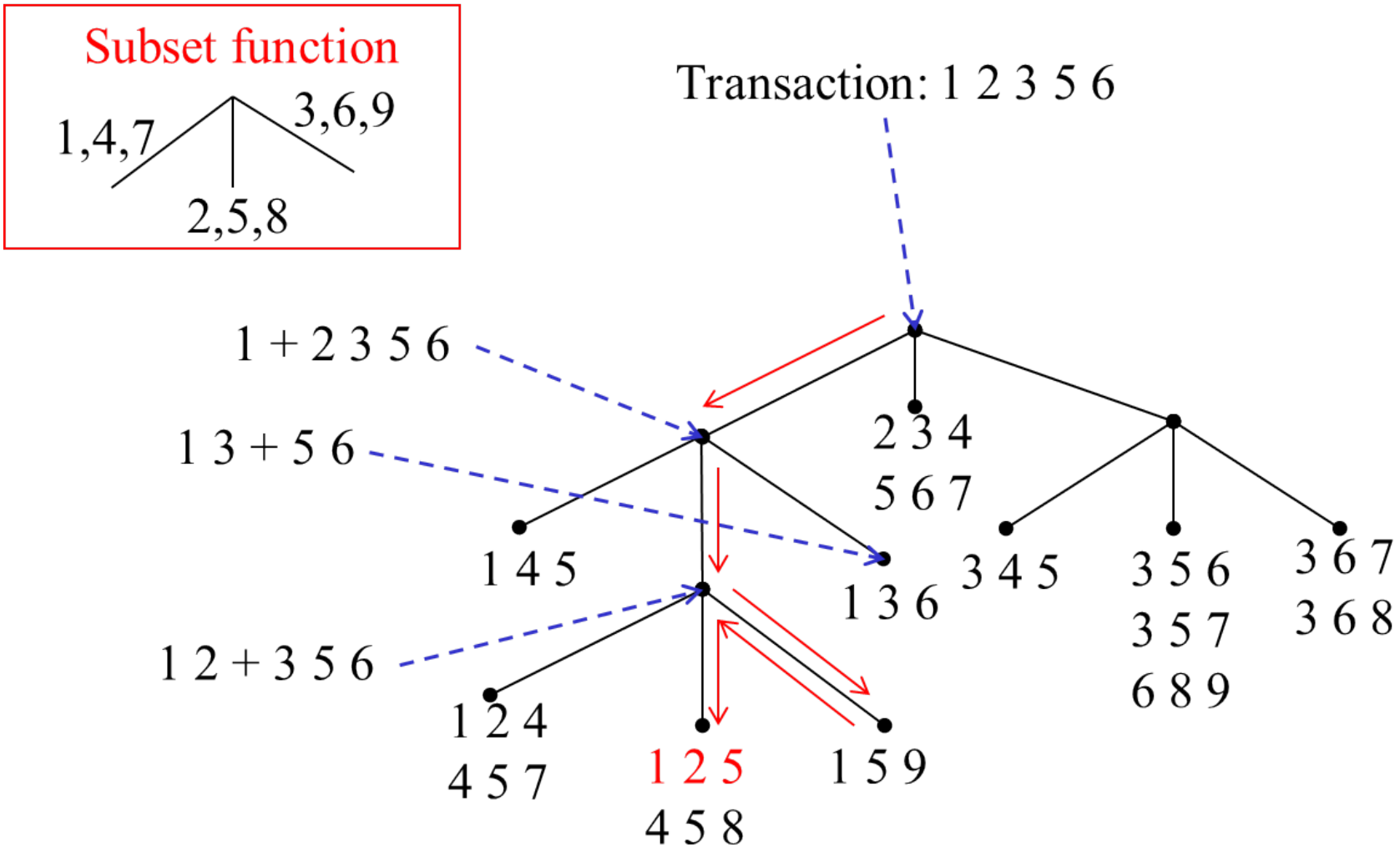
改进 $support(C_{k+1})$ 的计算



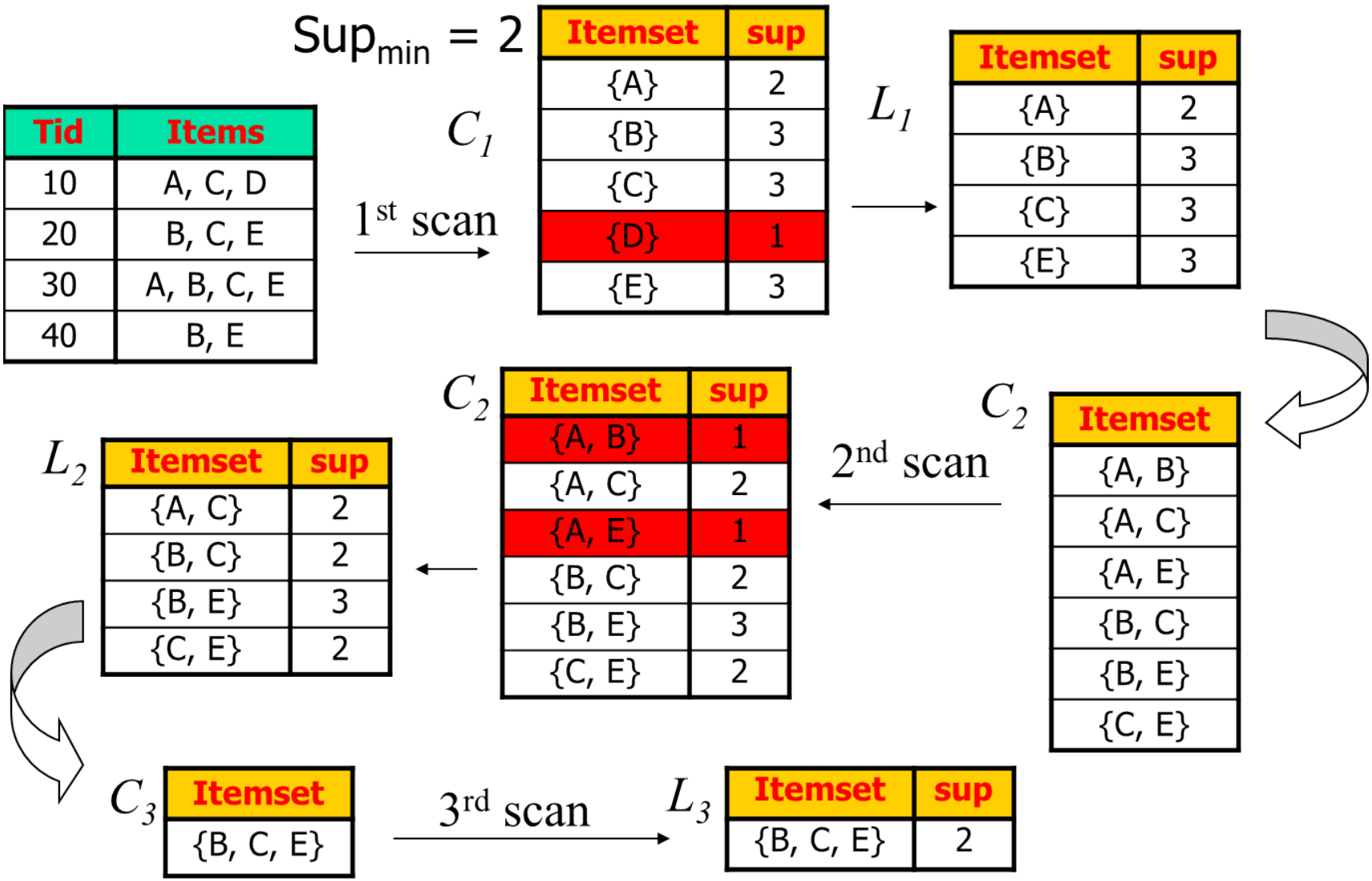
叶节点代表一个3-项集
任何一个中间节点，表示一个subset函数
中间结点最多3层，根表示对3-项集的第一个项用subset函数进行划分第二层中间节点，表示对3-项集的第二个项用subset函数进行划分



改进 $support(C_{k+1})$ 的计算



例子：计算频繁项集


Sup_min = 2
Tid
Items
10
A, C, D
20
B, C, E
30
A, B, C, E
40
B, E
C1
1st scan
Itemset
sup
{A}
2
{B}
3
{C}
3
{D}
1
{E}
3
L1
Itemset
sup
{A}
2
{B}
3
{C}
3
{E}
3
C2
Itemset
{A, B}
{A, C}
{A, E}
{B, C}
{B, E}
{C, E}
2nd scan
C2
Itemset
sup
{A, B}
1
{A, C}
2
{A, E}
1
{B, C}
2
{B, E}
3
{C, E}
2
L2
Itemset
sup
{A, C}
2
{B, C}
2
{B, E}
3
{C, E}
2
C3
Itemset
{B, C, E}
3rd scan
L3
Itemset
sup
{B, C, E}
2

改进算法-2：只扫描两次数据集

- 数据集被划分成若干个不相交的部分/子集
- 若一个项集 X 是频繁的，那么能够保证这些子数据集中至少一个，项集 X 是频繁的（易证明，反证法）
- 由此，得到新的频繁项集发现算法：
- 划分整个数据集为若干子数据集，第一遍扫描数据集，计算每个子数据集的频繁项集（调用“改进算法-1”），这些频繁项集称为“局部频繁项集”；
- 将所有找到的局部频繁项集作为候选项集，在整个数据集上进行“频繁”判决（“改进算法-1”最外层循环调用一次），这时进行第二遍数据扫描
- proposed by A. Savasere, E. Omiecinski and S. Navathe @VLDB'95
- 真的只扫描了两次数据集？

内存和外存的读写

内部存储器/外部存储器存取速度相差极大

- 存在外部磁盘上的数据，必须装入内存才能分析处理
- 算法设计的时候，基本原则：同一个数据从外存装入内存的次数越少越好
- 数据挖掘中，常见数据集巨大，存储在外部磁盘，而无法一次性装入内存中
- 我们现在区分两个概念：“装入”和“扫描”数据集；一次装入可被“快速扫描”多次，一次扫描可能需要分片多次”装入“
- 比较下面两段代码的性能：（假设数据集极大）

多次扫描数据集：方法一

```
...;
foreach x in L_1 do
    foreach y in L_k do
        foreach t in DS do
            ...;
...;
```

多次扫描数据集：方法二

```
...;
foreach t in DS do
    foreach x in L_1 do
        foreach y in L_k do
            ...;
...;
```

频繁项集发现：改进算法-3

- 想法：能否减少元组数量？“划分数据集”是一种想法，有没有其它的想法？
- 删除那些今后没有“用处”的元组/事务/数据集中的行
- 只需改动”改进算法-1”中的一个步骤，其它不用改动，很容易做到这一点

compute $support(C_{k+1})$

Input: DS: 数据集，C_{k+1}: 候选“频繁”的 (k+1)-项集集合
Output: C_{k+1} 中每个项集的支持度
扫描一遍 DS 元组，检查它包含的 C_{k+1} 中的项集，将被包含的项集的支持度 $+1$，如果没有任何项集的支持度被更新，从数据集中删除该元组
return $support(C_{k+1})$

频繁项集发现：改进算法-4

- 能不能减少 C_{k+1} 集合的大小；也就是说能不能有简单的判断方法，”粗略地“去掉”那些支持度不够的项集？
- proposed by J. Park, M. Chen, and P. Yu. @SIGMOD'95

Create hash table H_2 using hash function $h(x, y) = ((order\ of\ x) \times 10 + (order\ of\ y))\ mod\ 7$

H_2

bucket address	0	1	2	3	4	5	6
bucket count	2	2	4	2	2	4	4
bucket contents	{I1, I4} {I3, I5}	{I1, I5} {I1, I5}	{I2, I3} {I2, I3} {I2, I3} {I2, I3}	{I2, I4} {I2, I4}	{I2, I5} {I2, I5}	{I1, I2} {I1, I2} {I1, I2} {I1, I2}	{I1, I3} {I1, I3} {I1, I3} {I1, I3}

hash 桶计数法： 改进“改进算法-1”的初始化步骤

思想：；

1. 在统计 L_1 时，需要扫描整个数据集;
2. 对每个数据集的元组，计算其每个 2-项集在 hash 表 H_2 中的桶地址，并将其丢入对应桶中（桶计数器 $+1$）；
3. 桶计数器值大于最小支持度的桶，其桶内的项集可能完全相同，也可能不同，这些项集加入 C_2;

- 精度换速度？数据量极大时，可作为一个考虑
- proposed by H. Toivonen, @VLDB'96

改进算法-5：抽样数据集

思想：；

1. 采用随机抽样，获得 DS 的一个子集，在该子集上应用"改进算法-1"，找到"局部"频繁项集 (可能会丢失部分全局频繁项集);
2. 扫描 DS，验证获得的局部频繁项集中的闭频繁项集;
3. 再次扫描 DS，发现丢失的全局频繁项集;

频繁项集发现：改进算法-6

- 候选集中所有项集大小一样？一定要这样吗？
- 维持一个候选集 C 即可
- proposed byS. Brin R. Motwani, J. Ullman, and S. Tsur. @SIGMOD'97

改进算法-6：动态项集计数

1. 把数据集标记若干“起点”（元组），“起点”意味着，此时可以检查项集的支持度;
2. 在“起点”，把频繁的项集添加到候选项集 C;
3. 在”起点”，开始统计新频繁项集两两并集（更大项集）的支持度（异步？）

容易获得长度大的项集，项集中的各项都是频繁项

特点与不足

- 宽度优先/分层搜索的思想
- 产生候选项，再测试的思想
- 控制候选项产生的数量/候选项快速过滤方法的要求
- 能不能有改进？

一些想法

- 宽度优先搜索 $\longrightarrow$ 深度优先搜索
- 别产生候选项

例子：频繁模式树

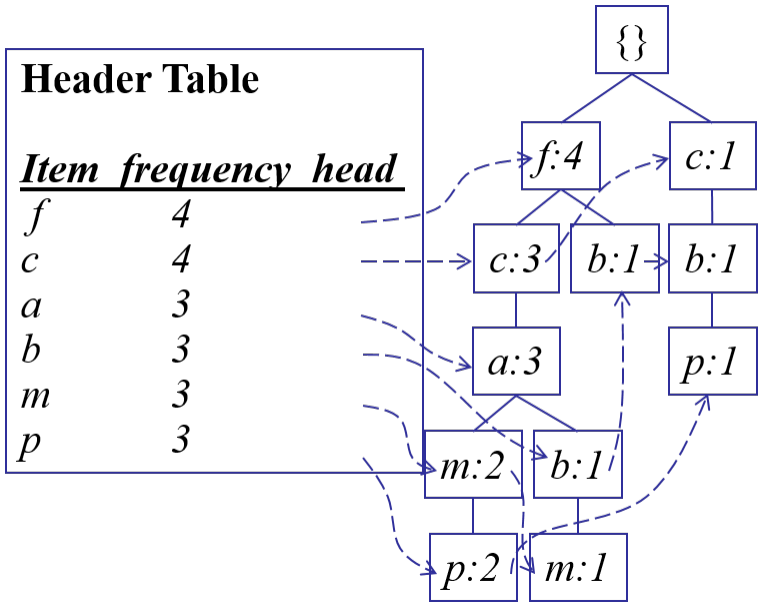
TID	Items bought	(ordered) frequent items
100	{f, a, c, d, g, i, m, p}	{f, c, a, m, p}
200	{a, b, c, f, l, m, o}	{f, c, a, b, m}
300	{b, f, h, j, o, w}	{f, b}
400	{b, c, k, s, p}	{c, b, p}
500	{a, f, c, e, l, p, m, n}	{f, c, a, m, p}

见课本P166

min_support = 3

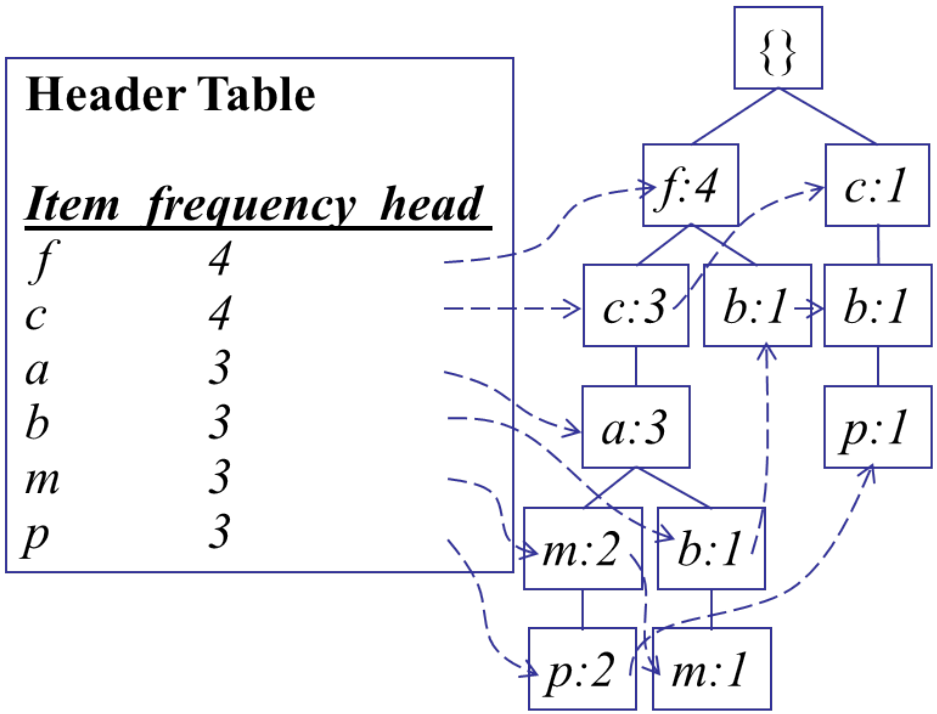
FP-tree

- 第一步，扫描数据集，计算 L_1，将原始事务数据集 DS 投影到 L_1 上（可按照 L_1 的次序调整事务数据库）
- 将 L_1 中的频繁项集降序排列（按支持度降序），如例子中有 $F-list = f-c-a-b-m-p$
- 扫描投影后得到的 DS 数据集，构建 FP-tree：
  - 树根是空集，其它节点是 L_1 中的 item/项，每个非根节点/项有一个计数器，记录经过该节点的路径数目
  - 每个元组是一条从根出发到某个节点路径。构建树时，扫描 DS 中的元组，元组第一个项是根的后继节点，元组其它项是前一个项的后继结点；若某个节点 i 的后继结点 j 已经存在，正处理的新元组又要给 i 添加后继结点 j 时，仅仅将后继结点的计数器 $+1$
  - 这棵 FP-tree 是数据集 DS 的“压缩”表示！构建树的过程就是压缩过程
  - 注意到：一个项可能在树中有多个对应的节点
  - 为处理方便，增加一个 header table，每个频繁项对应一个链表，链表指向了该项在 FP-tree 中的各个节点，如有图所示



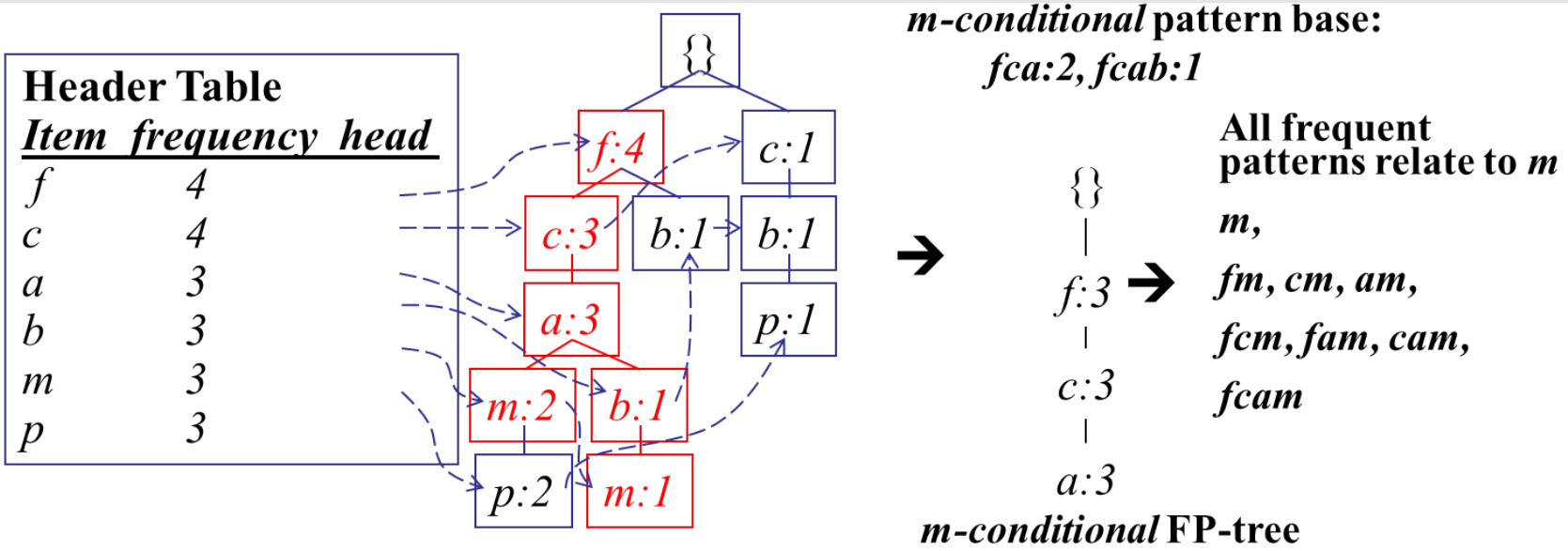
寻找路径的“主干”：条件模式基

- 对 L_1 中的每个项，寻找 FP-tree 中从根到代表它的节点的路径及其条数（可能同一条路径出现在多个元组中）
- 去掉路径起点和终点，得到路径的“主干”，即“条件模式基”



***Conditional* pattern bases**

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*



项 x 的“路径主干”可用来恢复成“小”一点的事务数据集 DS'

- DS' 这个数据集中任何一个元组，必然包括项 x，故可以把每个元组中的 x 丢掉
- 任何包括 x 的频繁项集必然要从这个数据集 DS' 中发现，任何 DS' 中的频繁项集并上 x 就是包括 x 的频繁项集
- 这样，形成递归算法，该递归算法返回所有包括 x 的频繁项集，递归停止条件是：DS' 构造的 FP-tree 只有一条路径

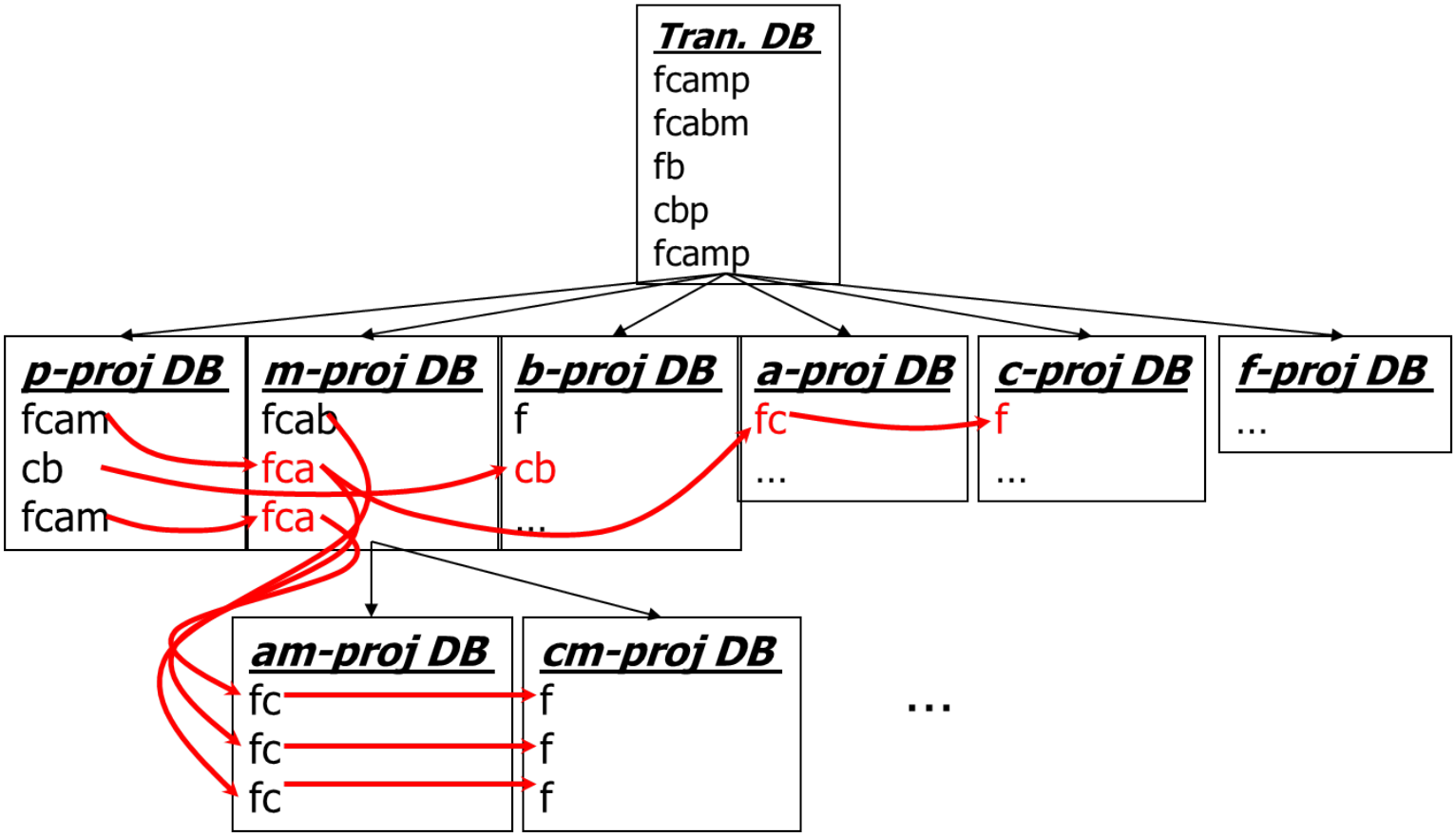
递归地使用 FP-tree，可以得到包含某个项 x 的所有频繁项集

- 大思路：
- 找到所有包括 x 的频繁项集，
- 在所有不包括 x 的频繁项集中找包括 y 的频繁项集，
- 在所有不包括 x, y 的频繁项集中找包括 z 的频繁项集，
- 以此类推,.....，直至 L_1 中的项都遍历完毕。

完全的和紧凑的

如果 FP-tree 在内存中无法存储下，怎么办？

- 并行投影，每个频繁项，它的路径主干对应一个小的数据集，每个数据集可以在不同的机器上并行计算，费空间



频繁模式增长的优缺点与发展

- 分而治之，易并行，无候选模式集，压缩数据集，不需要重复装入全部数据集等
- 算法提出，J. Han, J. Pei, and Y. Yin, @SIGMOD'00
- 各种改进：PrefixSpan (ICDE'01), CloSpan (SDM'03), BIDE (ICDE'04),CLOSET (DMKD'00), FPclose, and FPMax (Grahne, Zhu, Fimi'03),gSpan (ICDM'02), CloseGraph (KDD'03), Convertible constraints (ICDE'01), gPrune (PAKDD'03), H-tree, H-cubing, and Star-cubing (SIGMOD'01, VLDB'03), MaPle (Pei, et al., ICDM'03), Mining frequent and discriminative patterns (Cheng, et al, ICDE'07)
- 其他挖掘算法：FPgrowth+ (Grahne and Zhu, FIMI'03), AFOPT (Liu, et al. @ KDD'03), Carpenter (Pan, et al. @ KDD'03), TD-Close (Liu, et al, SDM'06)

ECLAT: 利用垂直数据格式来发现频繁项集

- 垂直数据格式：元组是一个项集，它的列是包括该项集的所有“交易 id”，所谓“交易 id”就是原始数据集（也叫水平数据格式）每个元组的 id
- 主要思想和 Apriori(改进算法-1）是一样的，由 k-频繁项集集合产生 (k+1)-项集候选集，然后测试支持度，垂直格式的优点是测试支持度非常简单，不需要装入数据集，直接元组中“交易 id”的个数是否超过最小支持度即可
- 算法的主要开销在：生成 L_1 时，要转换数据格式；C_{k+1} 是 L_k 中元素的两两相交

TID	*List of item_IDs*
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

(a) 水平数据格式

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}

(b) 垂直数据格式 L_1

{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}

(c) 垂直数据格式 L_2

闭模式/闭项集

- 一个项集 X 是闭的，当且仅当不存在任何项集 Y，满足 $Y \supset X, support(Y) = support(X)$
- 理解：对于任何 $Y, Y \supset X$，有 $support(Y) \leq support(X)$
- 理解：不要求 X 是频繁的，只要求没有和它具有一样支持度的超集即可
- 若添加一个条件，要求 X 是频繁的，那么就有”闭频繁项集”
- proposed by Pasquier, et al @ICDT'99
- 令集合 C 包括所有的闭项集，并且记录了每个闭项集的支持度，我们能知道任何一个项集的支持度，只要这个项集是 C 中某个项集的子集，具体方法：找到这个项集的 C 中的具有最大支持度的超集及其支持度

极大频繁模式/极大频繁项集

- 一个项集 X 是极大频繁项集，当且仅当它的任何超集都不是频繁项集，即 $\forall Y, Y \supset X, support(Y) < support_{min}$
- 理解：弄清楚闭频繁项集和极大频繁项集的关系，概念相近，易混淆
- 闭频繁项集不一定就是极大频繁项集，例如存在闭频繁项集，它的所有超集的支持度都比它的支持度小，但是其中有可能存在超集是频繁的，故闭频繁项集不一定极大；
- 极大频繁项集一定是闭的，因为它所有的超集支持度都小于最小支持度，也小于它的支持度，所以它的超集的支持度不可能和它的支持度一样，故是闭的
- proposed by Bayardo @SIGMOD'98
- 令集合 M 包括了所有极大频繁项集，并记录了其中每个极大频繁项集的支持度，区别于闭项集集合 C，我们不能知道其它项集的支持度，仅仅能够判断这个项集是否是频繁的，若这个项集是 M 中某个项集的子集
- 找到了 M，我能获得所有的频繁项集，why?
- 证明：频繁项集的子集必是频繁项集，任何频繁项集必然是 M 中某个项集的子集。（找极大频繁子集就可以了!!!）

闭频繁项集/极大频繁项集的例子

假定数据集仅包括两个事务记录，100 个不同的项/属性

- 数据集仅两个元组/两个事务记录
  $(a_1, a_2, \ldots, a_{50}), (a_1, a_2, \ldots, a_{100})$，预定义 $support_{min} = 1$
- 所有可能的模式有 $2^{100} - 1$ 个
- 所有闭频繁项集（2 个）构成了集合:$C = \{\{a_1, a_2, \ldots, a_{50}\}^2, \{a_1, a_2, \ldots, a_{100}\}^1\}$，集合上标注的指数表示它的支持度
- 所有极大繁项集（1 个）构成了集合:$M = \{\{a_1, a_2, \ldots, a_{100}\}^1\}$，集合上标注的指数表示它的支持度

如何找闭频繁项集？

可利用的事实 1

- 若 X 是频繁项集，Y 是项集，在数据集中，任何事务/元组 t，若包含 X，则一定包含 Y，但是不包含其它任何更多的项，也就是说 Y 极大项集，则 $X \cup Y$ 就是闭频繁项集

Min_sup=2

TID	Items
10	a, c, d, e, f
20	a, b, e
30	c, e, f
40	a, c, d, f
50	c, e, f

具体步骤

- 将所有频繁项按支持度由小到大排列，如左图，$flist \triangleq d - a - f - e - c$
  最小的支持度d
- 寻找包括 d 的闭频繁项集：任何包括 d 的事务/元组，若包括 $flist$ 的某个子集，如 acf，则 $dacf$ 是闭频繁项集
- 思考：算法如何实现？如何找到所有的闭频繁项集

可利用的事实 2

- 已发现闭频繁项集 Y，它的任何真子集 X，其支持度和该闭频繁项集一样，则可以删除 X

  X不是闭频繁项集

思考：如何利用？

进一步思考：如何找极大频繁项集？如何利用投影技术来改进这些算法？垂直数据格式和闭频繁模式/极大频繁模式的结合？

算法时间复杂度？最低支持度和频繁项集数目的关系？

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

打篮球与吃麦片

- 关联规则：打篮球 $\Rightarrow$ 吃麦片 $[40\%, 66.7\%]$
- 这个关联规则会误导人：很多人打篮球，打篮球的大多数人都吃麦片，单看没问题；实际情况，这是错误的：反过来看，75%的吃麦片，但是这些人如果打篮球，吃麦片的人少了（比例66.7%），负相关！
- 新关联规则：打篮球 $\Rightarrow$ 不吃麦片 $[20\%, 33.3\%]$ 更有意义！？

从关联规则到相关性：找有用的规则

3750/5000=75% 2000/3000=66.7%
这些人不打篮球，吃麦片的人多了1750/2000=87.5%

- 构成关联规则的两个项集 A, B，相关吗？正相关？负相关？
- 提升度/lift：$lift(A,B) = \frac{P(A \cup B)}{P(A)P(B)}$，1：表示独立；$<1$：表示负相关；$>1$：表示正相关
- 上例中：$lift(B,C) = \frac{2000/5000}{3000/5000 * 3750/5000} = 0.89$(负相关)，
  $lift(B,NC) = \frac{1000/5000}{3000/5000 * 1250/5000} = 1.33$（正相关）

各种“兴趣度”的度量

symbol	measure	range	formula
ϕ	ϕ-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	$\frac{\Sigma_j \max_k P(A_j,B_k)+\Sigma_k \max_j P(A_j,B_k)-\max_j P(A_j)-\max_k P(B_k)}{2-\max_j P(A_j)-max_k P(B_k)}$
M	Mutual Information	0 ...1	$\frac{\Sigma_i\Sigma_j P(A_i,B_j)\log\frac{P(A_i,B_j)}{P(A_i)P(B_j)}}{\min(-\Sigma_i P(A_i)\log P(A_i)\log P(A_i), -\Sigma_i P(B_i)\log P(B_i)\log P(B_i))}$
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	$\frac{P(A,B)}{\sqrt{P(A)P(B)}}$
γ	coherence(Jaccard)	0 ...1	$\frac{P(A,B)}{P(A)+P(B)-P(A,B)}$
α	all_confidence	0 ...1	$\frac{P(A,B)}{\max(P(A),P(B))}$
o	odds ratio	0 ...∞	$\frac{P(A,B)P(\overline{A},\overline{B})}{P(\overline{A},B)P(A,\overline{B})}$
V	Conviction	0.5 ...∞	$\max(\frac{P(A)P(\overline{B})}{P(A\overline{B})}, \frac{P(B)P(\overline{A})}{P(B\overline{A})})$
λ	lift	0 ...∞	$\frac{P(A,B)}{P(A)P(B)}$
S	Collective strength	0 ...∞	$\frac{P(A,B)+P(\overline{AB})}{P(A)P(B)+P(\overline{A})P(\overline{B})} \times \frac{1-P(A)P(B)-P(\overline{A})P(\overline{B})}{1-P(A,B)-P(\overline{AB})}$
χ^2	χ^2	0 ...∞	$\Sigma_i \frac{(P(A_i)-E_i)^2}{E_i}$

如果存在大量的这种数据，会怎样？如何评价 measures?

Table 6: Properties of interestingness measures. Note that none of the measures satisfies all the properties.

Symbol	Measure	Range	P1	P2	P3	O1	O2	O3	O3'	O4
ϕ	ϕ-coefficient	$-1 \cdots 0 \cdots 1$	Yes	Yes	Yes	Yes	No	Yes	Yes	No
λ	Goodman-Kruskal's	$0 \cdots 1$	Yes	No	No	Yes	No	No*	Yes	No
α	odds ratio	$0 \cdots 1 \cdots \infty$	Yes*	Yes	Yes	Yes	Yes	Yes*	Yes	No
Q	Yule's Q	$-1 \cdots 0 \cdots 1$	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
Y	Yule's Y	$-1 \cdots 0 \cdots 1$	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
κ	Cohen's	$-1 \cdots 0 \cdots 1$	Yes	Yes	Yes	Yes	No	No	Yes	No
M	Mutual Information	$0 \cdots 1$	Yes	Yes	Yes	No**	No	No*	Yes	No
J	J-Measure	$0 \cdots 1$	Yes	No	No	No**	No	No	No	No
G	Gini index	$0 \cdots 1$	Yes	No	No	No**	No	No*	Yes	No
s	Support	$0 \cdots 1$	No	Yes	No	Yes	No	No	No	No
c	Confidence	$0 \cdots 1$	No	Yes	No	No**	No	No	No	Yes
L	Laplace	$0 \cdots 1$	No	Yes	No	No**	No	No	No	No
V	Conviction	$0.5 \cdots 1 \cdots \infty$	No	Yes	No	No**	No	No	Yes	No
I	Interest	$0 \cdots 1 \cdots \infty$	Yes*	Yes	Yes	Yes	No	No	No	No
IS	Cosine	$0 \cdots \sqrt{P(A,B)} \cdots 1$	No	Yes	Yes	Yes	No	No	No	Yes
PS	Piatetsky-Shapiro's	$-0.25 \cdots 0 \cdots 0.25$	Yes	Yes	Yes	Yes	No	Yes	Yes	No
F	Certainty factor	$-1 \cdots 0 \cdots 1$	Yes	Yes	Yes	No**	No	No	Yes	No
AV	Added value	$-0.5 \cdots 0 \cdots 1$	Yes	Yes	Yes	No**	No	No	No	No
S	Collective strength	$0 \cdots 1 \cdots \infty$	No	Yes	Yes	Yes	No	Yes*	Yes	No
ζ	Jaccard	$0 \cdots 1$	No	Yes	Yes	Yes	No	No	No	Yes
K	Klosgen's	$(\frac{2}{\sqrt{3}}-1)^{1/2}[2-\sqrt{3}-\frac{1}{\sqrt{3}}] \cdots 0 \cdots \frac{2}{3\sqrt{3}}$	Yes	Yes	Yes	No**	No	No	No	No

where: P1: $O(\mathbf{M}) = 0$ if $det(\mathbf{M}) = 0$, *i.e.*, whenever A and B are statistically independent.
P2: $O(\mathbf{M_2}) > O(\mathbf{M_1})$ if $\mathbf{M_2} = \mathbf{M_1} + [k \ -k; \ -k \ k]$.
P3: $O(\mathbf{M_2}) < O(\mathbf{M_1})$ if $\mathbf{M_2} = \mathbf{M_1} + [0 \ k; \ 0 \ -k]$ or $\mathbf{M_2} = \mathbf{M_1} + [0 \ 0; \ k \ -k]$.
O1: Property 1: Symmetry under variable permutation.
O2: Property 2: Row and Column scaling invariance.
O3: Property 3: Antisymmetry under row or column permutation.
O3': Property 4: Inversion invariance.
O4: Property 5: Null invariance.
Yes*: Yes if measure is normalized.
No*: Symmetry under row or column permutation.
No**: No unless the measure is symmetrized by taking $\max(M(A,B), M(B,A))$.