資料探勘
DATA MINING

張家瑋 助理教授

國立臺中科技大學資訊工程系

國立成功大學工程科學系

jwchang@nutc.edu.tw

張家瑋.大平台.tw

關聯規則學習

Association Rule Learning

概念

- 在大型資料庫中發現項目間關聯的方法。
  - {牛奶, 麵包}→{可樂}：代表某人同時買了牛奶和麵包，就可能會買可樂。

- 該方法常使用於電子商務上，通常可為促銷、產品推薦等行銷活動的決策依據。

定義

- 商品的項目集合(itemset)，$I = \{ I_1, I_2, \ldots, I_m \}$。#Item
- 交易資料庫(Database)，$D = \{t_1, t_2, \ldots, t_n\}$。 #Transaction
- 關聯規則(Association Rule)， $X \rightarrow Y$

案例

TID	網球拍	網 球	運動鞋	羽毛球
1	1	1	1	0
2	1	1	0	0
3	1	0	0	0
4	1	0	1	0
5	0	1	1	1
6	1	1	0	0

- 顧客購買記錄的資料庫 *D*，包含 6 個 Transactions
- 項目集 *I* = {網球拍, 網球, 運動鞋, 羽毛球}

觀察關聯規則，網球拍 → 網球。

1. Transaction 1, 2, 3, 4, 6 包含網球拍。
2. Transaction 1, 2, 6 同時包含網球拍和網球。
3. 支持度 = 3/6 = 0.5，信心度 = 3/5 = 0.6。

- 若最小支持度為 0.5，最小信心度為 0.6。
- 關聯規則 “網球拍→網球” 是存在強關聯的。

- 1-itemset (4): {網球拍}, {網球}, {運動鞋}, {羽毛球}
- 2-itemset (7): {網球拍, 網球}, {網球拍, 運動鞋}, {網球拍, 羽毛球},
  {網球, 運動鞋}, {網球, 運動鞋}, {網球, 羽毛球}
  {運動鞋, 羽毛球}
- 3-itemset (4): {網球拍, 網球, 運動鞋}, {網球拍, 網球, 羽毛球}, {網球拍, 運動鞋, 羽毛球}
  {網球, 運動鞋, 羽毛球}

APRIORI

概念

- 逐層搜索的迭代方法。

- k-itemset 用於探索（$k+1$）- itemset。
  1. 找出 frequent 1-itemset，L_1。L_1 用來找 frequent 2-itemset，L_2。而 L_2 用來找到 L_3。直到不能找到 k-itemset。
  2. 每找一個 L_k 需要掃描一次資料庫。為提高頻繁項集逐層產生的效率， Apriori 性質則可減少搜索。

- Apriori 性質：frequent itemset 的所有非空子集都必須是頻繁的。
  - 若某個 k-itemset 的 candidate 的 subsets 不在 (k-1)-itemset 時，這個 candidate 就可以直接刪除。

案例

TID	網球拍	網 球	運動鞋	羽毛球
1	1	1	1	0
2	1	1	0	0
3	1	0	0	0
4	1	0	1	0
5	0	1	1	1
6	1	1	0	0

- 顧客購買記錄的資料庫 *D*，包含 6 個 Transactions
- 項目集 *I* = {網球拍, 網球, 運動鞋, 羽毛球}

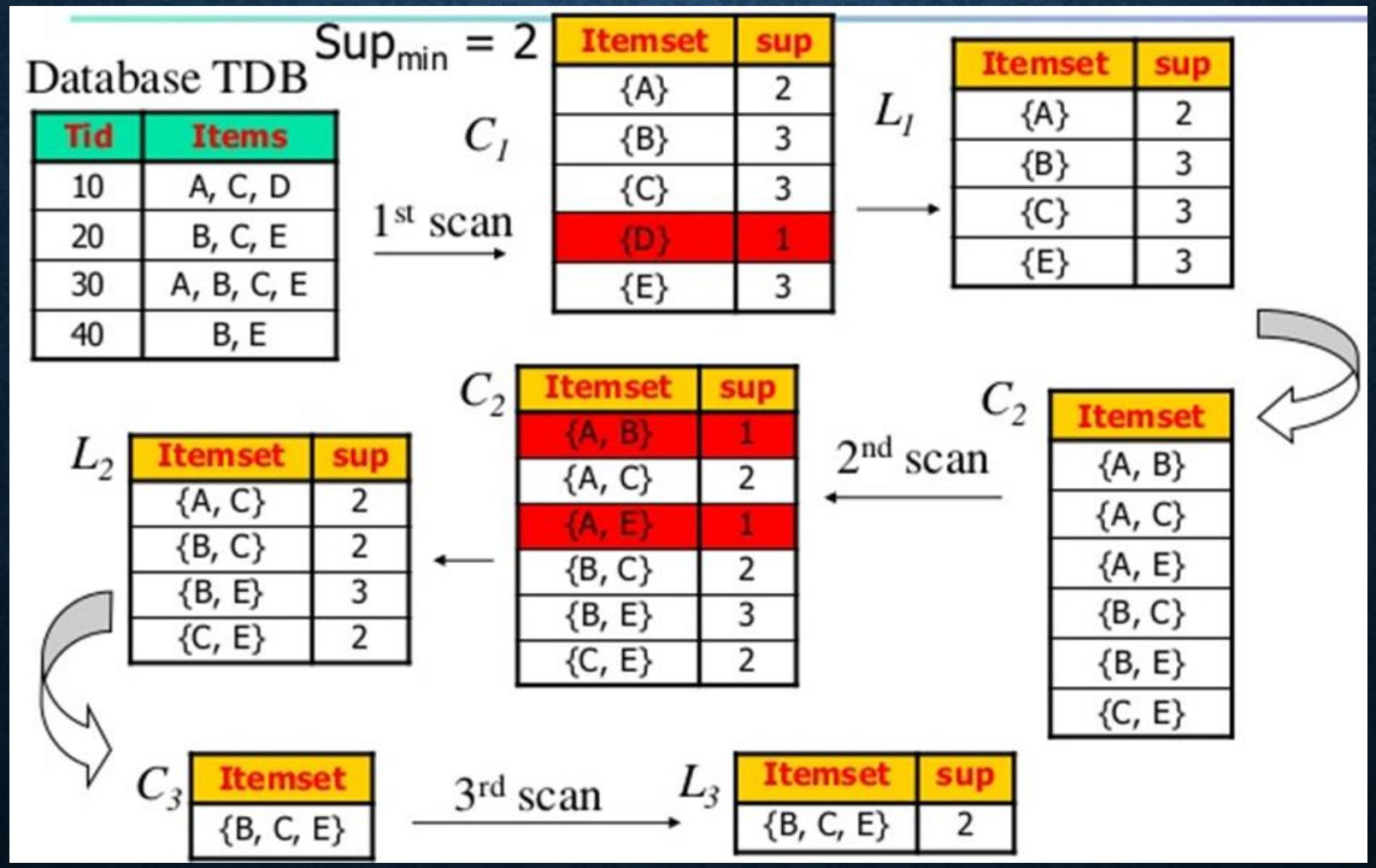
觀察關聯規則，網球拍 → 網球。

1. Transaction 1, 2, 3, 4, 6 包含網球拍。
2. Transaction 1, 2, 6 同時包含網球拍和網球。
3. 支持度 = 3/6 = 0.5，信心度 = 3/5 = 0.6。

- 若最小支持度為 0.5，最小信心度為 0.6。
- 關聯規則 “網球拍→網球” 是存在強關聯的。

- 1-itemset (4): {網球拍}, {網球}, {運動鞋}, ~~{羽毛球}~~
- 2-itemset (7): {網球拍, 網球}, {網球拍, 運動鞋}, {網球拍, 羽毛球}, {網球, 運動鞋}, {網球, 運動鞋}, ~~{網球, 羽毛球}~~ ~~{運動鞋, 羽毛球}~~
- 3-itemset (4): {網球拍, 網球, 運動鞋}, ~~{網球拍, 網球, 羽毛球}~~, ~~{網球拍, 運動鞋, 羽毛球}~~ ~~{網球, 運動鞋, 羽毛球}~~

方法


Database TDB
Tid Items
10 A, C, D
20 B, C, E
30 A, B, C, E
40 B, E
Sup_min = 2
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

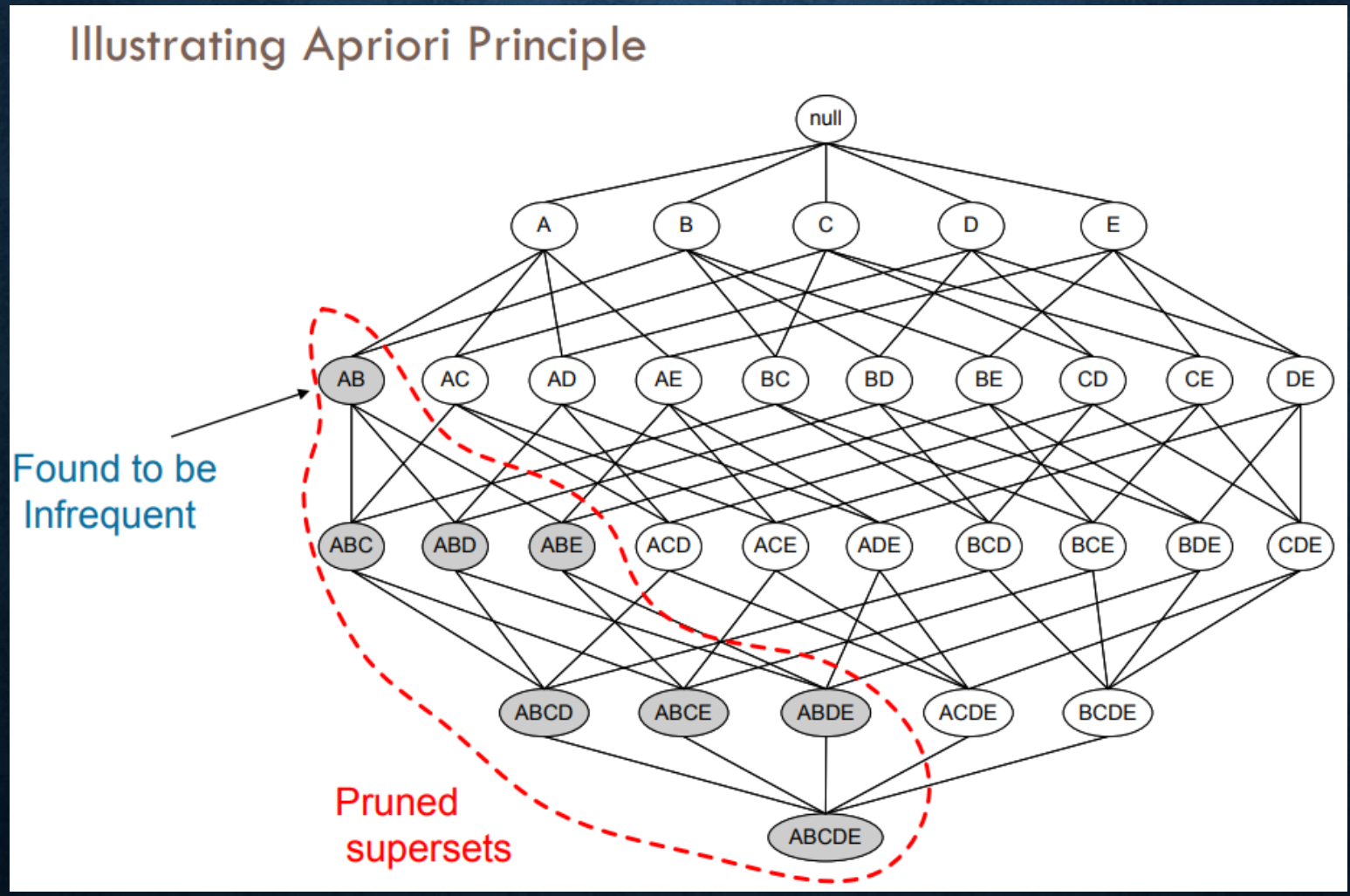
方法

1. C_3 = L_2 的組合
   - L_2 = {{A, C}, {B, C}, {B, E}, {C, E}}
     {{A, C}, {B, C}, {B, E}, { C, E}}
     = {{A, B, C}, {A, C, E}, {B, C, E}}

2. 使用 Apriori 性質剪枝：某個 frequent itemset 的所有 subsets 必須是頻繁的，對 candidate itemset C_3，我們可以刪除其非頻繁的 subsets ：
   - {A, B, C} 的 2-itemset 是 {A, B}, {A, C}, {B, C}，其中 {A, B} 不是 L_2 的元素，所以刪除;
   - {A, C, E} 的 2-itemset 是 {A, C}, {A, E}, {C, E}，其中 {A, E} 不是 L_2 的元素，所以刪除;
   - {B, C, E} 的 2-itemset 是 {B, C}, {B, E}, {C, E}，所有 2-itemset 都是 L_2 的元素，因此保留。

3. 剪枝後得到 C_3 ={{B, C, E}}

剪枝


Illustrating Apriori Principle
null
A
B
C
D
E
AB
AC
AD
AE
BC
BD
BE
CD
CE
DE
ABC
ABD
ABE
ACD
ACE
ADE
BCD
BCE
BDE
CDE
ABCD
ABCE
ABDE
ACDE
BCDE
ABCDE
Found to be Infrequent
Pruned supersets

THINKING TIME

重點

1. 在每一步產生 candidate itemset 時產生的組合過多，沒有排除不應該參與組合的元素。

2. 每次計算 itemset 的支持度時都對全部的 transactions 掃描一遍，造成龐大的I / O開銷。這種代價是隨著資料的增加而產生幾何級數的增長。

FP-GROWTH

概念

- 不用產生 candidate itemsets。
- 以樹(Tree)的結構儲存 frequent itemsets，即 frequent pattern tree (FP-tree)。
- 只要遞迴地探勘這棵樹。

FP-TREE 建造方法

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

min_support = 3

a	3
b	3
c	4
f	4
m	3
p	3

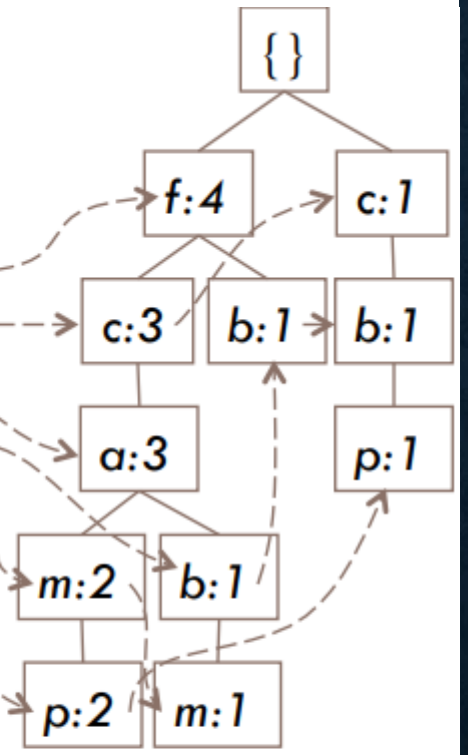
f	4
c	4
a	3
b	3
m	3
p	3

FP-TREE 建造方法

f	4
c	4
a	3
b	3
m	3
p	3

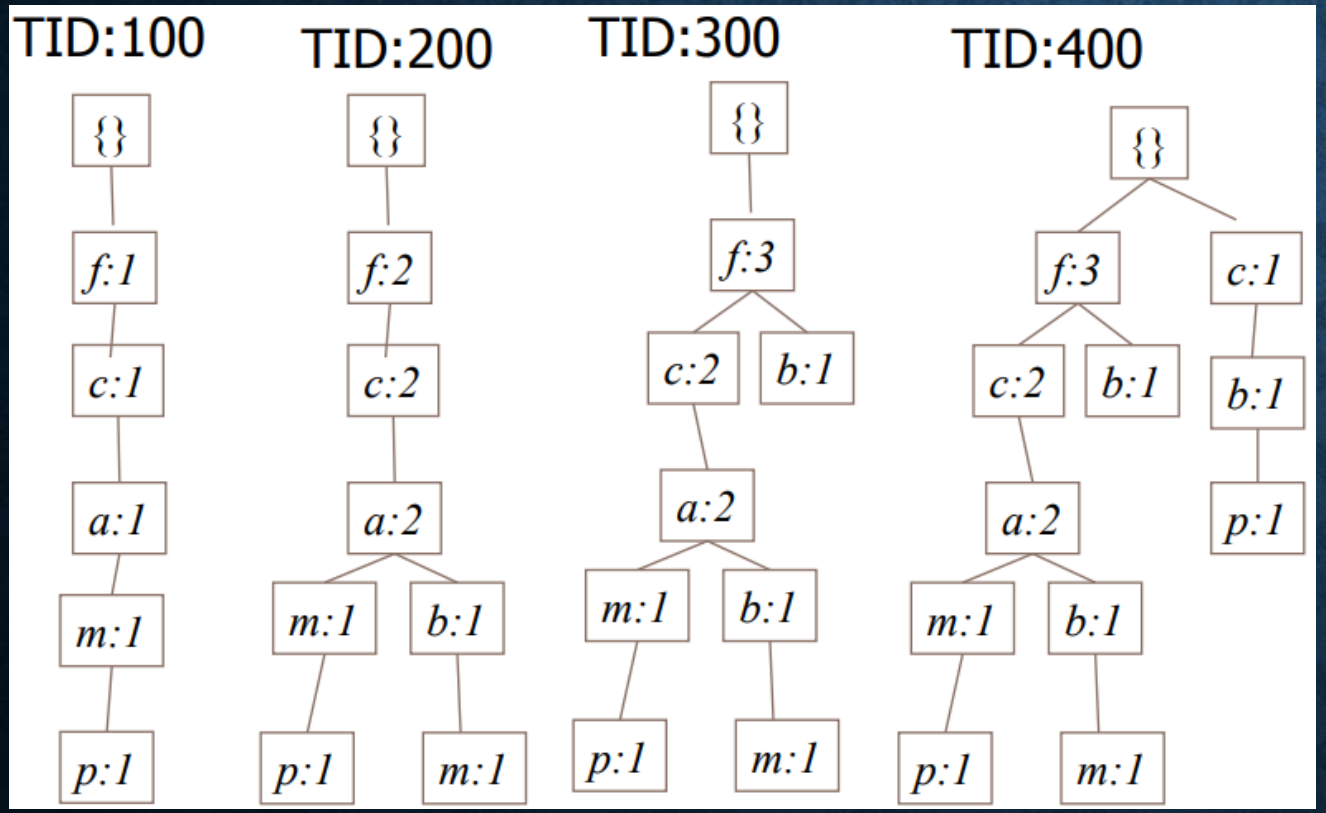
Header Table

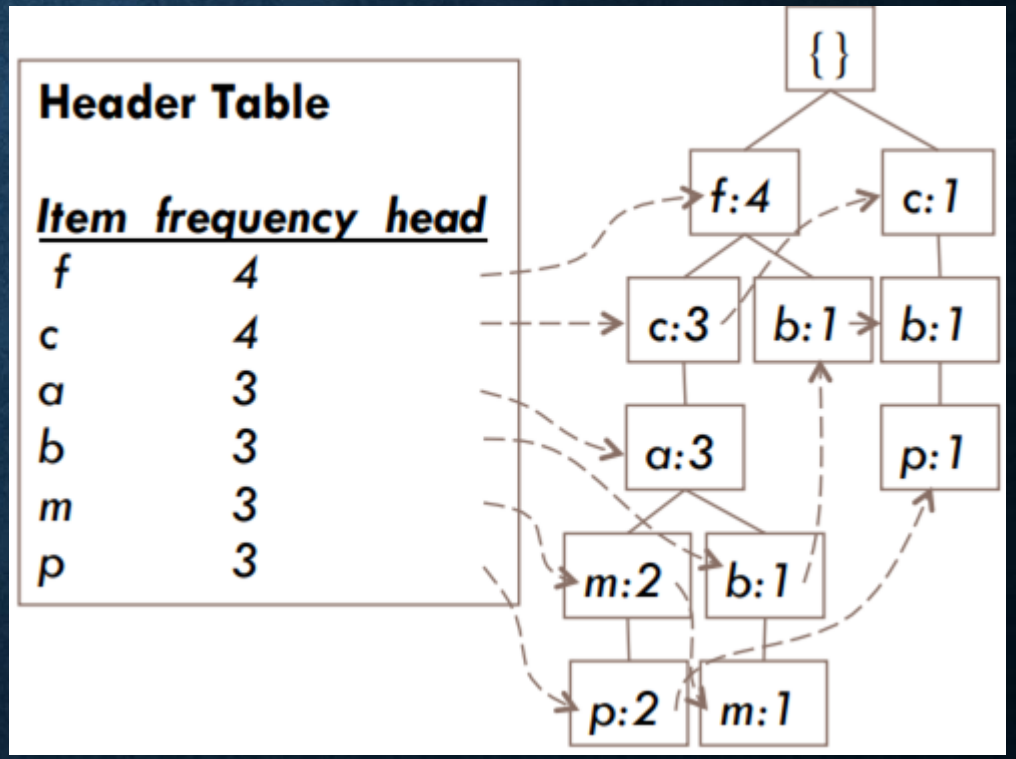
Item	*frequency*	*head*
f	*4*	
c	*4*	
a	*3*	
b	*3*	
m	*3*	
p	*3*	



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

FP-TREE 建造方法

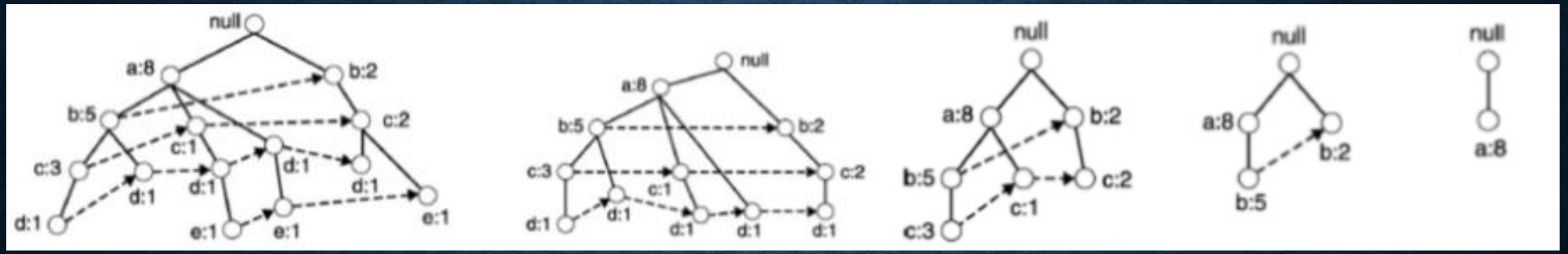

TID:100
{}
f:1
c:1
a:1
m:1
p:1
TID:200
{}
f:2
c:2
a:2
m:1
b:1
p:1
m:1
TID:300
{}
f:3
c:2
b:1
a:2
m:1
b:1
p:1
m:1
TID:400
{}
f:3
c:1
c:2
b:1
b:1
a:2
p:1
m:1
b:1
p:1
m:1


Header Table
Item frequency head
f 4
c 4
a 3
b 3
m 3
p 3
{}
f:4
c:1
c:3
b:1
b:1
a:3
p:1
m:2
b:1
p:2
m:1

MINING TREE

- Bottom-Up 探索，依序檢視每個項目。
- 遞迴建子樹，找到所有 k-itemsets。

MINING TREE



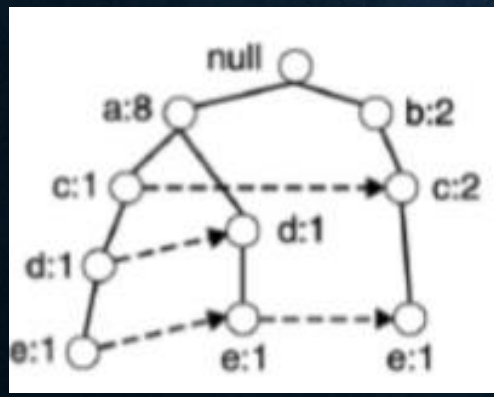
FP-tree

包含d節點
的子樹

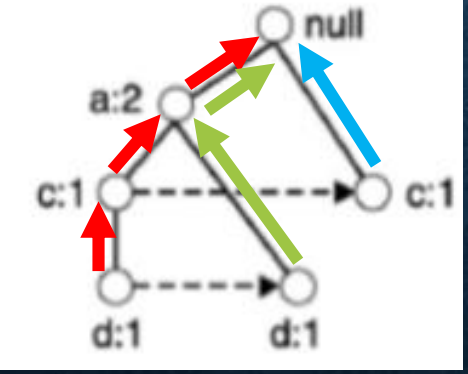
包含c節點
的子樹

包含b節點
的子樹

包含a節點
的子樹



包含e節點
的子樹



{e}的條件FP-tree
If *Sup{b} < 2*

THINKING TIME

重點

- 避免多次掃描資料庫(for support)，節省了IO與運算成本。
- 不產生 candidate itemset。

THANK YOU

REFERENCE

- https://zh.wikipedia.org/wiki/%E5%85%B3%E8%81%94%E8%A7%84%E5%88%99%E5%AD%A6%E4%B9%A0
- https://www.slideshare.net/waynechung944/fp-growth-intro
- 成功大學資工系高宏宇教授的簡報