数据挖掘原理

主讲教师：李志勇

数据科学系
数字农业工程技术研究中心
移动：13882213811 电邮: lzy@sicau.edu.cn

第四章：关联规则

——购物营销 自由乾坤

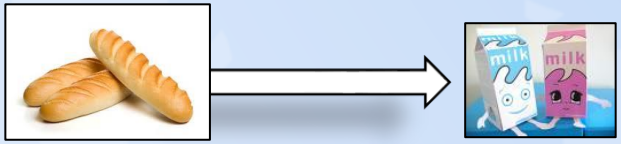
主讲教师：李志勇

主要介绍内容

- 4.1 概述
- 4.2 交易集
- 4.3 频繁集
- 4.4 关联规则
- 4.5 Apriori算法
- 4.6 FP-growth

4.1 概述

交易号 TID	顾客购买商品 Items
T1	bread cream milk tea
T2	bread cream milk
T3	cake milk
T4	milk tea
T5	bread cake milk



4.1 概述

Frequently Bought Together ➡ 频繁项集

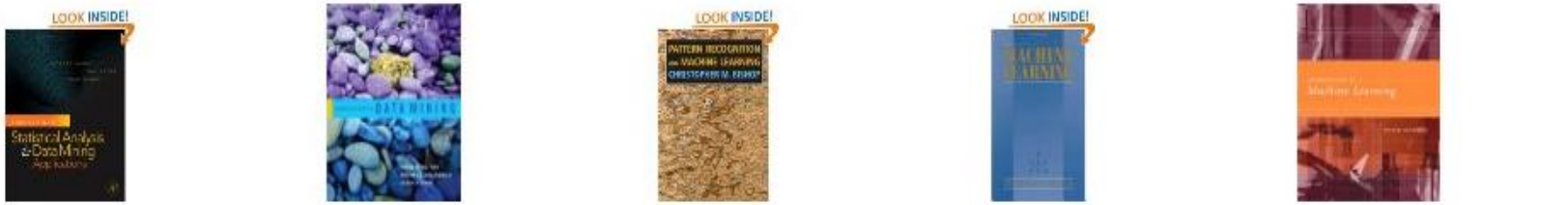
Price For All Three: $166.83

Add all three to Cart | Add all three to Wish List

Show availability and shipping details

- **This item:** Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems) by Eibe Frank
- The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition (Springer Series in Statistics) by Robert Tibshirani
- Pattern Recognition and Machine Learning (Information Science and Statistics) by Christopher M. Bishop

Customers Who Bought This Item Also Bought ➡ 关联规则



Handbook of Statistical Analysis and Data Mining... by John Elder IV
(9) $71.96

Introduction to Data Mining by Pang-Ning Tan
(15) $80.80

Pattern Recognition and Machine Learning... by Christopher M. Bishop
(47) $58.03

Machine Learning (Mcgraw-Hill International Edit) by Tom M. Mitchell
(38) $73.72

Introduction to Machine Learning (Adaptive Comp... by Ethem Alpaydin
(8) $43.79

4.2 交易集

- **定义：项目与项集**
  - 设$I=\{i_1, i_2, \ldots, i_m\}$是 m 个不同项目的集合，每个 i_k (k=1，2，……，m)称为一个项目(*Item*)。
  - 项目的集合 I 称为项目集合(*Itemset*)，简称为项集。其元素个数称为项集的长度，长度为 k 的项集称为 k-项集(k-*Itemset*)。

假定某超市销售的商品包括：*bread*、*bear*、*cake*、*cream*、*milk*和*tea*，每个销售商品就是一个项目，超市中出售的所有商品的项集为

$$I=\{bread, beer, cake, cream, milk, tea\}$$

4.2 交易集

- 定义：交易

- 每笔交易T(Transaction)是项集 I 上的一个子集，即T⊆I，但通常T⊂I。

- 对应每一个交易有一个唯一的标识——交易号，记作TID

- 交易的全体构成了交易数据库 D，或称交易记录集 D，简称交易集D。

- 交易集D中包含交易的个数记为 |D|。

4.2 交易集

Transactions	Items
1	Bread, Jelly, Peanut, Butter
2	Bread, Butter
3	Bread, Jelly
4	Bread, Milk, Butter
5	Chips, Milk
6	Bread, Chips
7	Bread, Milk
8	Chips, Jelly

4.2 交易集

购物篮数据的表示方法主要有两种：使用事务数据格式或者表数据格式。

●事务数据格式

事务数据格式需要两个字段（ID字段和内容字段），每条记录仅表示一个商品。

事务ID	项
1	Bread
2	Butter
3	Jelly
4	Milk
5	Chips
6	Chips
7	Milk
8	Jelly

4.2 交易集

●表数据格式

采用表数据格式时，每条记录表示不同的事务。每条记录中属性只有购买(1)和不购买(0)两种情况，不统计商品的任何其他信息，如下所示：

tid	面包	牛奶	尿布	啤酒	鸡蛋	可乐
1	1	1	0	0	0	0
2	1	0	1	1	1	0
3	0	1	1	1	0	1
4	1	1	1	1	0	0
5	1	1	1	0	0	0

4.3 频繁集

- 定义：项集的支持度

对于项集X，X⊂I，设定count(X⊆T)为交易集D中包含X的交易的数量

$$\text{support}(X) = \frac{\text{count}(X \subseteq T)}{|D|}$$

项集X的支持度support(X)就是项集X出现的概率，即项集X的交易数占所有交易数的比重，从而描述了X的重要性。

Transactions	Items
1	Bread, Jelly, Peanut, Butter
2	Bread, Butter
3	Bread, Jelly
4	Bread, Milk, Butter
5	Chips, Milk
6	Bread, Chips
7	Bread, Milk
8	Chips, Jelly

Itemset	Support	Itemset	Support
Bread	6/8	Bread, Butter	3/8
Butter	3/8	…	
Chips	2/8	Bread, Butter, Chips	0/8
Jelly	3/8	…	
Milk	3/8	Bread, Butter, Chips, Jelly	0/8
Peanut	1/8	…	
		Bread, Butter, Chips, Jelly, Milk	0/8
		…	
		Bread, Butter, Chips, Jelly, Milk, Peanut	0/8

4.3 频繁集

频繁模式主要作用是寻找到数据集中频繁出现的项集、序列或子结构。

- 定义：项集的最小支持度与频繁集
  - 发现关联规则要求项集必须满足的最小支持阈值，称为项集的最小支持度(Minimum Support)，记为supmin。从统计意义上讲，它表示用户关心的关联规则必须满足的最低重要性。只有满足最小支持度的项集才能产生关联规则。
  - 支持度大于或等于supmin的项集称为频繁项集，简称频繁集，反之则称为非频繁集。通常k-项集如果满足supmin，称为k-频繁集，记作L_k。

4.4 关联规则

•定义：关联规则

关联规则(Association Rules)是反映一个事物与其他事物之间的关联性，是一个形如“由于某些事件的发生而引起另外一些事件的发生”之类的规则。可以表示为一个蕴含式：

$$R：X \Rightarrow Y$$

•定义：关联规则的支持度

对于关联规则R：$X \Rightarrow Y$，其中$X \subset I$，$Y \subset I$，并且$X \cap Y = \Phi$，规则R的的支持度(Support)是交易集中同时包含X和Y的交易数与所有交易数之比。

$$support(X \Rightarrow Y) = \frac{count(X \cup Y)}{|D|}$$

4.4 关联规则

- 定义：关联规则的可信度

对于关联规则R：$X \Rightarrow Y$，其中$X \subset I, Y \subset I$，并且$X \cap Y = \Phi$，规则R的可信度(Confidence)是指包含X和Y的交易数与包含X的交易数之比

$$\text{confidence}(X \Rightarrow Y) = \frac{\text{count}(X \cup Y)}{\text{count}(X)}$$

也可以写作：

$$\text{confidence}(X \Rightarrow Y) = \frac{\text{support}(X \cup Y)}{\text{support}(X)}$$

4.4 关联规则

Transactions	Items
1	Bread, Jelly, Peanut, Butter
2	Bread, Butter
3	Bread, Jelly
4	Bread, Milk, Butter
5	Chips, Milk
6	Bread, Chips
7	Bread, Milk
8	Chips, Jelly

Bread → Milk

Support: 2/8

Confidence: 2/6

Milk → Bread

Support: 2/8

Confidence: 2/3

4.4 关联规则

- 定义：关联规则的最小支持度和最小可信度
  - 关联规则的最小支持度也就是衡量频繁集的最小支持度(Minimum Support)，记为supmin，它用于衡量规则需要满足的最低重要性。
  - 关联规则的最小可信度(Minimum Confidence)记为confmin，它表示关联规则需要满足的最低可靠性。

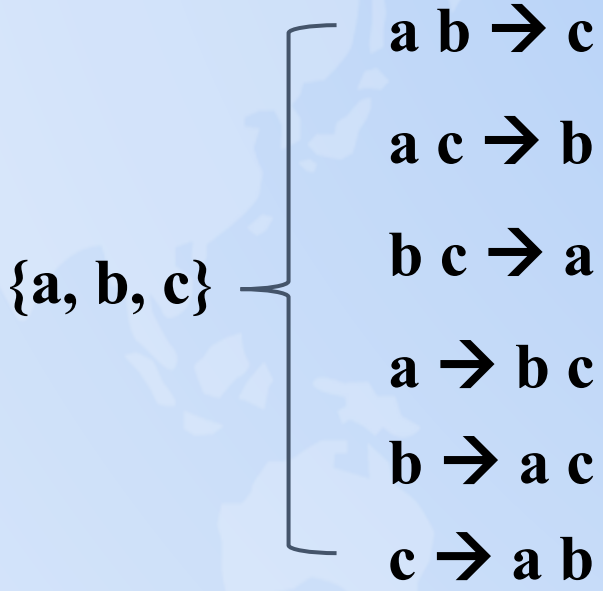
- 如果规则X⇒Y满足：

$$support(X \Rightarrow Y) \geq supmin \text{且} confidence(X \Rightarrow Y) \geq confmin$$

称关联规则X⇒Y为强关联规则，否则称关联规则X⇒Y为弱关联规则。

4.3 关联规则

- 步骤 1: 找到所有的频繁项集.
- 步骤 2: 利用频繁项集生成关联规则.
  - 对于每个频繁项集f，生成f的所有非空子集;
  - 对于f的所有非空子集s，当 support (f) / support (s) 大于最小可信度Φ时，生成 s → (f-s).



4.4 关联规则

- 关联规则的可信度并不是越高越好:
  - |D|=10000
  - #{DVD}=7500
  - #{Tape}=6000
  - #{DVD, Tape}=4000
  - Thresholds: σ=30%, Φ=50%
  - Support(Tape → DVD)= 4000/10000=40% > σ
  - Confidence(Tape → DVD)=4000/6000=66%> Φ

Tape → DVD是一个强关联规则

看起来Tapes 有助于销售DVDs.

但是，P(DVD)=75% > P(DVD | Tape) !!

购买Tape 的人不太可能购买DVDs.

4.4 关联规则

关联规则的可信度并不是越高越好

Transactions
Bread, Milk
Bread, Battery
Bread, Butter
Bread, Honey
Bread, Chips
Yogurt, Coke
Bread, Battery
Cookie, Jelly

$$P(Bread \mid Battery) = 100\% > P(Bread) = 75\%$$

4.4 关联规则

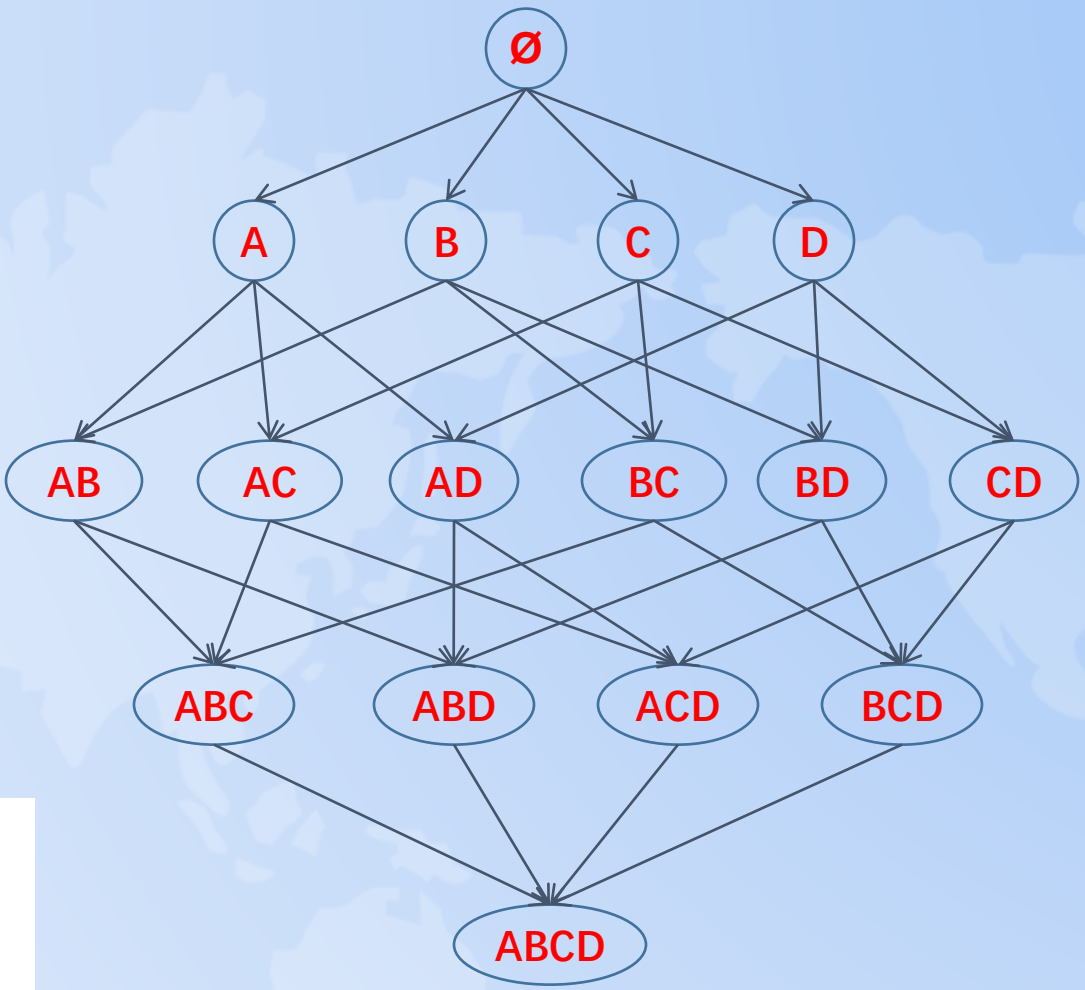
两件事情具有相关性并不代表因果关系

关联规则 ≠ 因果关系

P(Y|X) 仅仅是条件概率

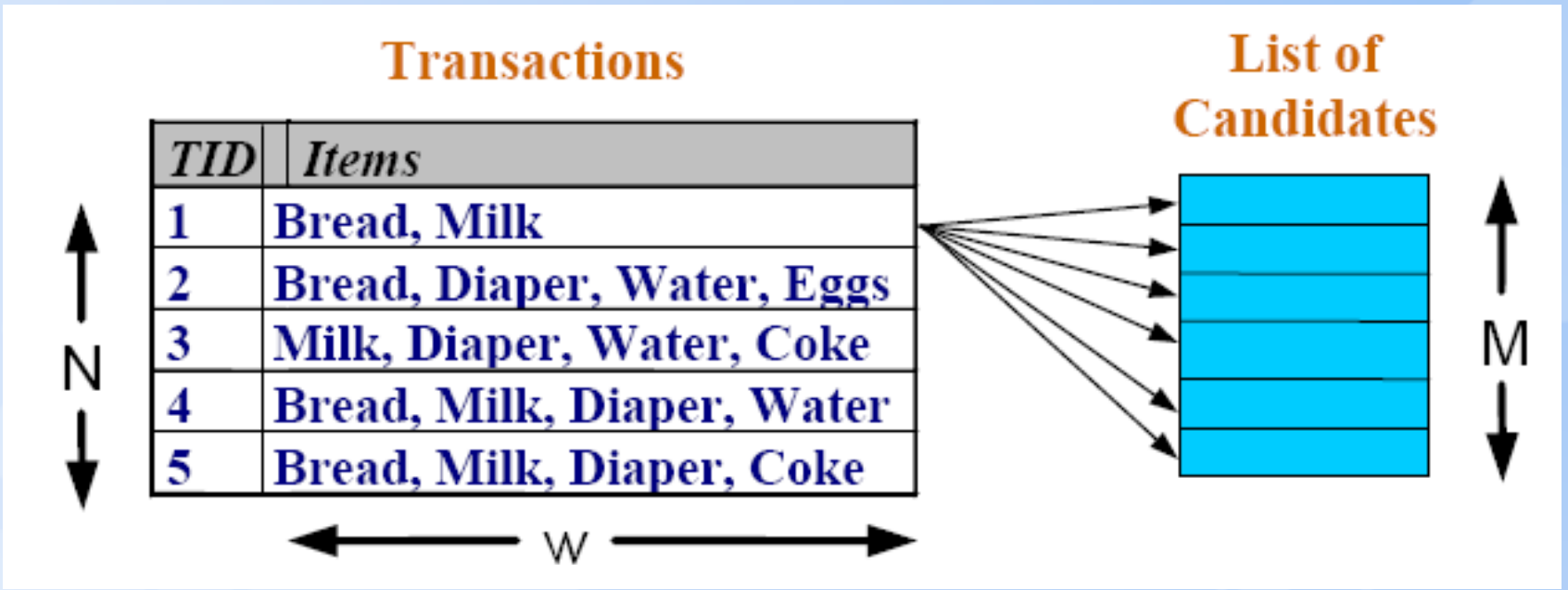
4.4 关联规则

4.5 Apriori算法



暴力求解方式对每种可能都会进行数据库的全表扫描
效率低下

4.5 Apriori算法



d件商品

$$O(NMW)$$

$$M = 2^d - 1$$

Itemset Calculation

4.5 Apriori算法

Apriori算法是常用的用于挖掘出数据关联规则的算法，它用来找出数据值中频繁出现的数据集合，找出这些集合的模式有助于我们做一些决策。它的核心思想是：

- 如果某个项集是频繁项集，那么它所有的子集也是频繁的。即如果{Milk, Bread}是频繁的，那么{Milk}, {Bread} 也一定是频繁的。
- 如果一个项集是非频繁的，那么它的所有超集也是非频繁的。即如果{Battery}是非频繁的，那么{Milk, Battery}也一定是非频繁的。

Title 1–20	Cited by	Year
Fast algorithms for mining association rules R Agrawal, R Srikant Proc. 20th int. conf. very large data bases, VLDB 1215, 487-499	19603	1994
Mining association rules between sets of items in large databases R Agrawal, T Imieliński, A Swami ACM SIGMOD Record 22 (2), 207-216	17129	1993
Mining sequential patterns R Agrawal, R Srikant Data Engineering, 1995. Proceedings of the Eleventh International Conference ...	6017	1995

4.5 Apriori算法

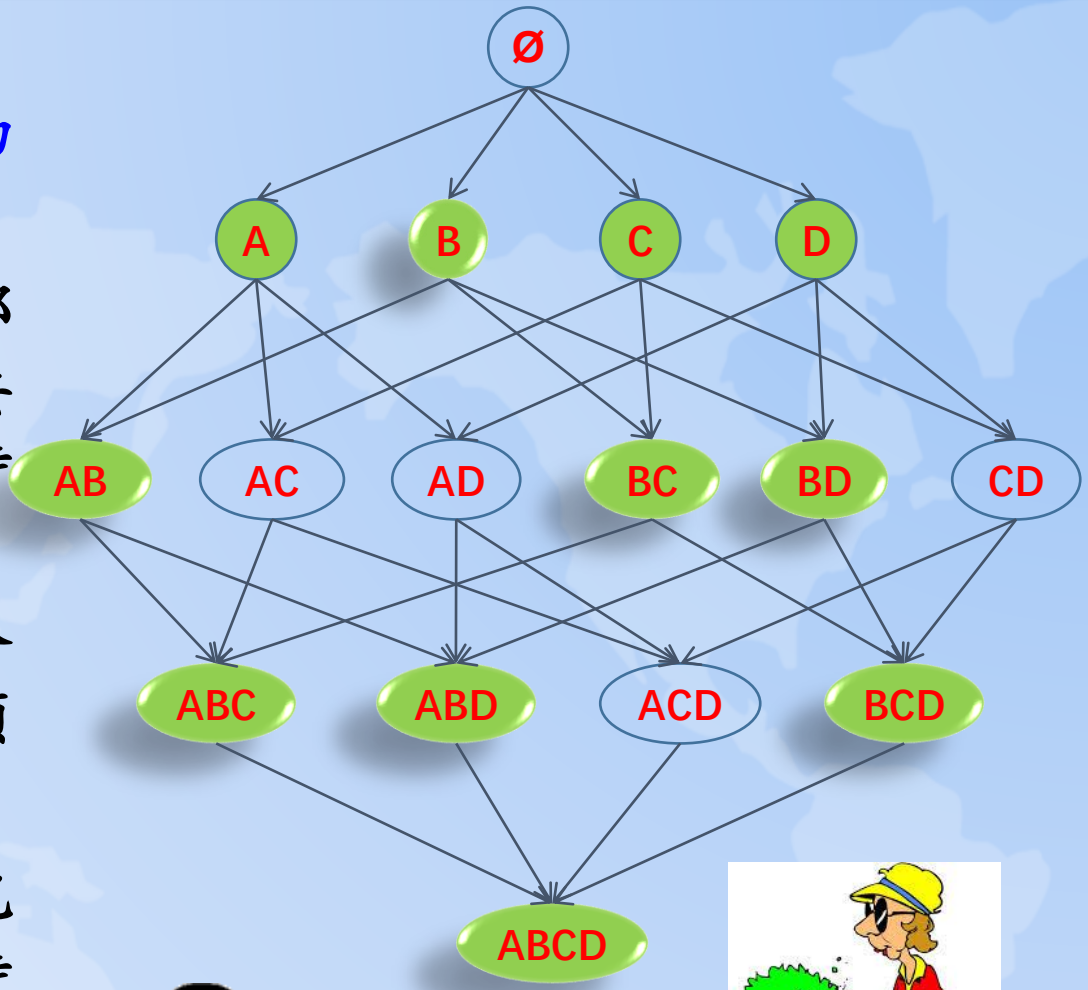
生成频繁项集主要步骤：

①首先会生成所有单个物品的项集列表；

②扫描交易记录来查看哪些项集满足最小支持度要求，那些不满足最小支持度的集合会被去掉；

③对剩下的集合进行组合以生成包含两个元素的项集；

④接下来重新扫描交易记录，去掉不满足最小支持度的项集，重复进行直到所有项集都被去掉。



候选集修剪

4.5 Apriori算法

输　入：交易数据库 D，最小支持度阈值 sup_{min}。

输　出：可以产生规则的所有频繁集 L_1。

C_k：k-候选频繁集。

L_k：k-频繁集。

```
(1) L1=find_frequent_1_itemset(D);//发现 1-频繁集
(2) for(k=2;Lk-1≠∅;k++){
(3)     Ck=apriori_gen(Lk-1,supmin);//根据 k-1-频繁集产生 k-候选集
(4)     for each t∈D {//扫描记录集，以确定每个候选集的支持度
(5)         Ct=subset(Ck,t);//获得 t 所包含的候选集
(6)     for each c∈Ct c.count++;
(7) }
(8) Lk={c∈Ck|c.count>supmin};
(9) return L=⋃k Lk;
```

4.5 Apriori算法

- apriori_gen(L_{k-1}, supmin)算法

输入：上一次循环扫描的结果 L_{k-1}，最小支持度阈值 sup_{min}

输出：候选频繁集 C_k。

```
(3.1) for each l1∈Lk-1
(3.2)  for each l2∈Lk-1
(3.3)  if((l1[1]=l2[1])∧…∧(l1[k-2]=l2[k-2])∧(l1[k-1]<l2[k-1])){
(3.4)  c= l1⊕l2 ; //将只差一项的两个项集连接到一起
(3.5)  if has_infrequent_subset(c,Lk-1)
(3.6)  delete c; //删去不可产生频繁项集的候选
(3.7)  else Ck=Ck∪{c};
(3.8)  }
(3.9) return Ck;
```

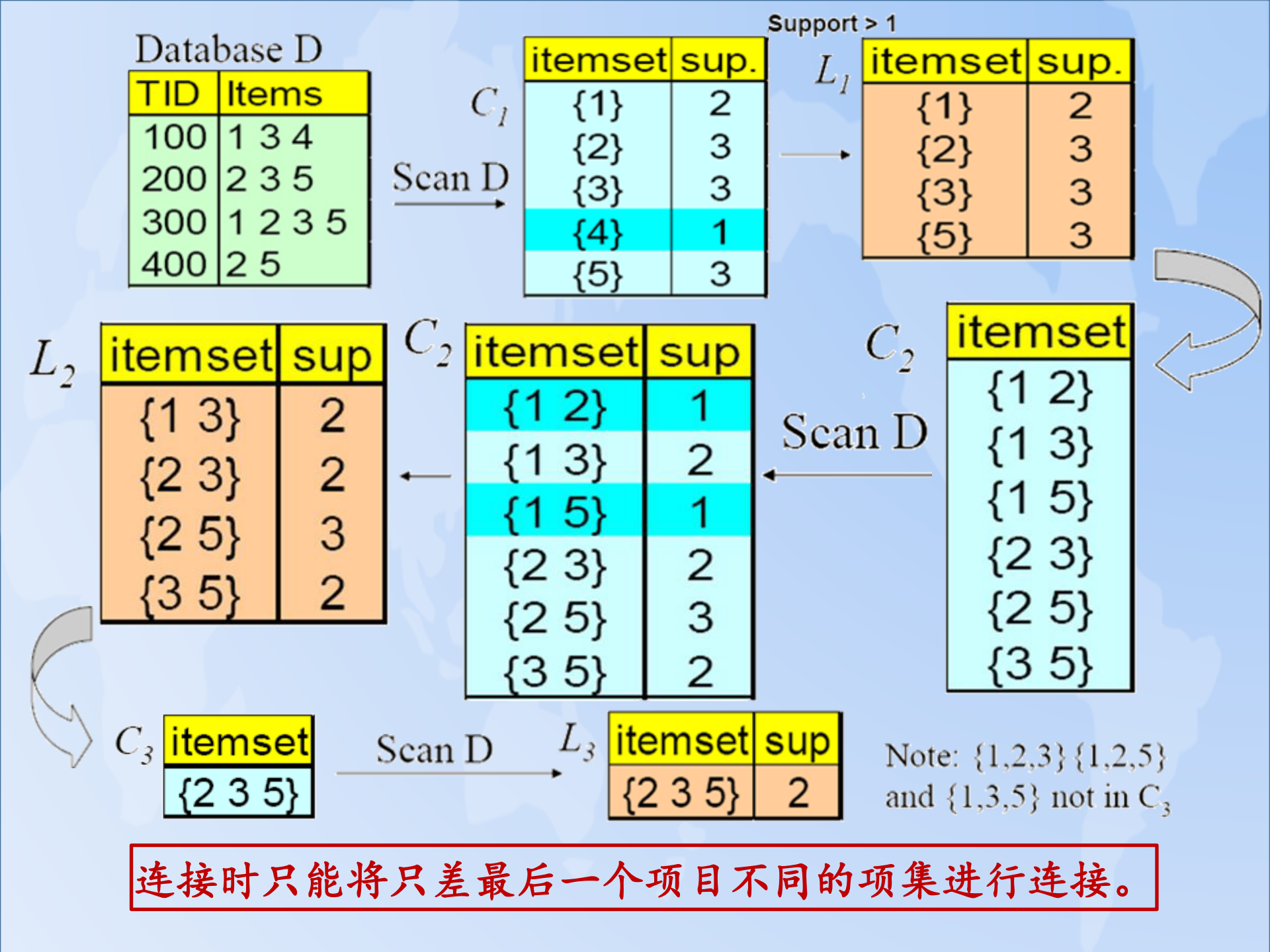
4.5 Apriori算法

- has_infrequent_subset(c, L_{k-1})算法

输　入：本次扫描产生的 C_k 的每个子集 c，上次扫描产生的 L_{k-1}。

输　出：c 是否将被从 C_k 中删除。

```
(3.5.1) for each (k-1)-subset s of c
          //根据算法性质：候选集的子集一定是频繁的
(3.5.2)   if s ∉ Lk-1 return TRUE ; //删除掉子集是不频繁的候选集
            else return FALSE ;
```


Database D
TID Items
100 1 3 4
200 2 3 5
300 1 2 3 5
400 2 5
Scan D
C_1
itemset sup.
{1} 2
{2} 3
{3} 3
{4} 1
{5} 3
Support > 1
L_1
itemset sup.
{1} 2
{2} 3
{3} 3
{5} 3
C_2
itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}
Scan D
C_2
itemset sup
{1 2} 1
{1 3} 2
{1 5} 1
{2 3} 2
{2 5} 3
{3 5} 2
L_2
itemset sup
{1 3} 2
{2 3} 2
{2 5} 3
{3 5} 2
C_3 itemset
{2 3 5}
Scan D
L_3 itemset sup
{2 3 5} 2
Note: {1,2,3} {1,2,5} and {1,3,5} not in C_3
连接时只能将只差最后一个项目不同的项集进行连接。

4.5 Apriori算法

设定sup_{min}=50%，$conf_{min}$=50%，使用Apriori算法完成下表所示的数据集关联规则挖掘。

交易TID	顾客购买商品Items
T_1	*A, B, C*
T_2	*A, C*
T_3	*A, D*
T_4	*B, E, F*

①先由交易表的所有项目直接产生1-候选集C_1，计算其支持度。去除支持度小于sup_{min}的项集，形成1-频繁集L_1；

项集C_1	支持度	项集L_1	支持度
A	3/4	*A*	3/4
B	1/2	*B*	1/2
C	1/2	*C*	1/2
D	1/4		
E	1/4		
F	1/4		

4.5 Apriori算法

②为发现2-频繁集L_2，首先利用L_1中的各项目组合连接，来产生2-候选集C_2；然后扫描记录集，以获得C_2中各项集的支持度。去除支持度小于sup_{min}的项集，形成2-频繁集L_2；

项集C_2	支持度	项集L_2	支持度
AB	1/4		
AC	2/4	AC	2/4
BC	1/4		

4.6 FP-growth算法

FP-growth(Frequent Pattern Tree, 频繁模式树)，是韩家炜老师提出的挖掘频繁项集的方法，是将数据集存储在一个特定的称作FP树的结构之后发现频繁项集或频繁项对，即常在一块出现的元素项的集合FP树。

FP-growth算法比Apriori算法效率更高，在整个算法执行过程中，只需遍历数据集2次，就能够完成频繁模式发现，其发现频繁项集的基本过程如下：

（1）构建FP树；

（2）从FP树中挖掘频繁项集。

4.6 FP-growth算法

FP-growth的一般流程如下：

1：先扫描一遍数据集，得到1-候选集，定义最小支持度删除那些小于最小支持度的项目，得到1-频繁集；然后将原始数据集中的条目按频繁集中降序进行排列。

2：第二次扫描，创建项头表（从上往下降序），以及FP树。

3：对于每个项目找到其条件模式基(CPB，conditional patten base)，递归调用树结构，删除小于最小支持度的项。如果最终呈现单一路径的树结构，则直接列举所有组合；非单一路径的则继续调用树结构，直到形成单一路径即可。

4.6 FP-growth算法

表4.1 超市交易数据集

交易TID	顾客购买商品Items	交易TID	顾客购买商品Items
T_1	*bread, cream, milk, tea*	T_6	*bread, tea*
T_2	*bread, cream, milk*	T_7	*beer, milk, tea,*
T_3	*cake, milk*	T_8	*bread, tea*
T_4	*milk, tea*	T_9	*bread, cream, milk, tea*
T_5	*bread, cake, milk*	T_{10}	*bread, milk, tea*

第一步：构建FP树

1. 扫描数据集，对每个物品进行计数

项目	*milk*	*bread*	*tea*	*cream*	*cake*	*beer*
计数	8	7	7	3	1	1

4.6 FP-growth算法

2. 设定最小支持度（即物品最少出现的次数）为3

3. 按降序重新排列物品集（如果出现计数小于3的物品则删除）

项目	*milk*	*bread*	*tea*	*cream*
计数	8	7	7	3

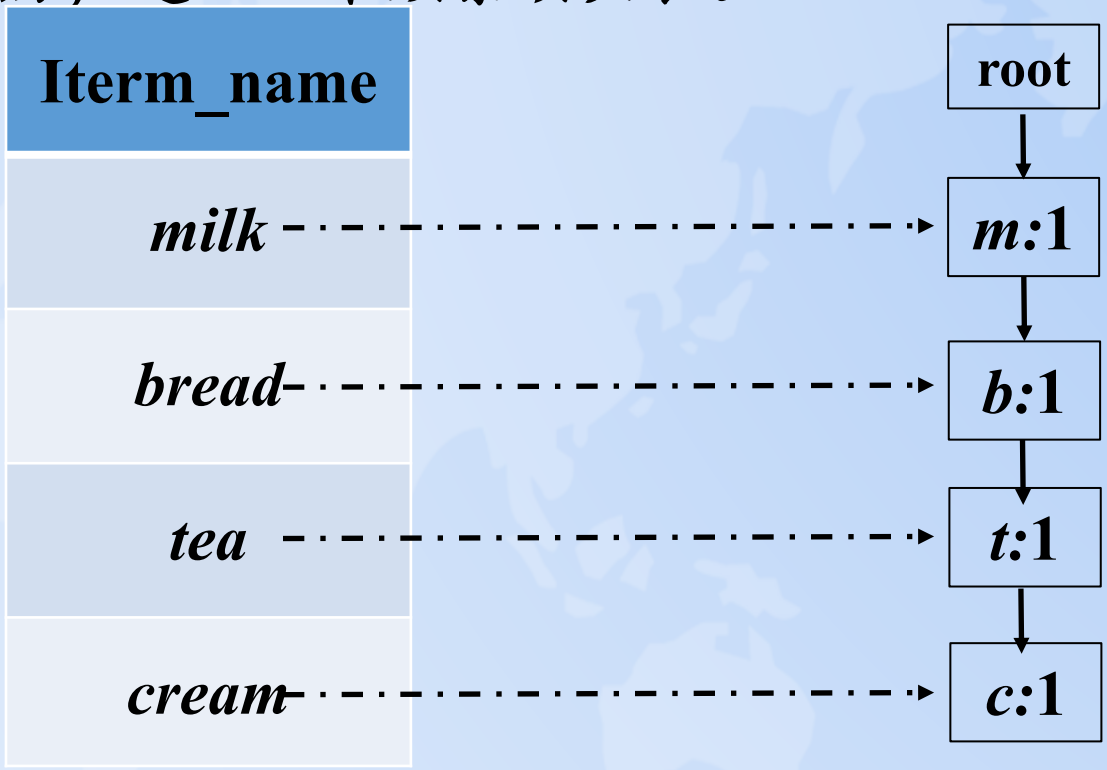
4. 根据项目（物品）出现的次数重新调整物品清单

交易TID	顾客购买商品Items	交易TID	顾客购买商品Items
T_1	*milk, bread, tea, cream*	T_6	*bread, tea*
T_2	*milk, bread, cream*	T_7	*milk, tea,*
T_3	*milk*	T_8	*bread, tea*
T_4	*milk, tea*	T_9	*milk, bread, tea, cream*
T_5	*Milk, bread*	T_{10}	*milk, bread, tea*

4.6 FP-growth算法

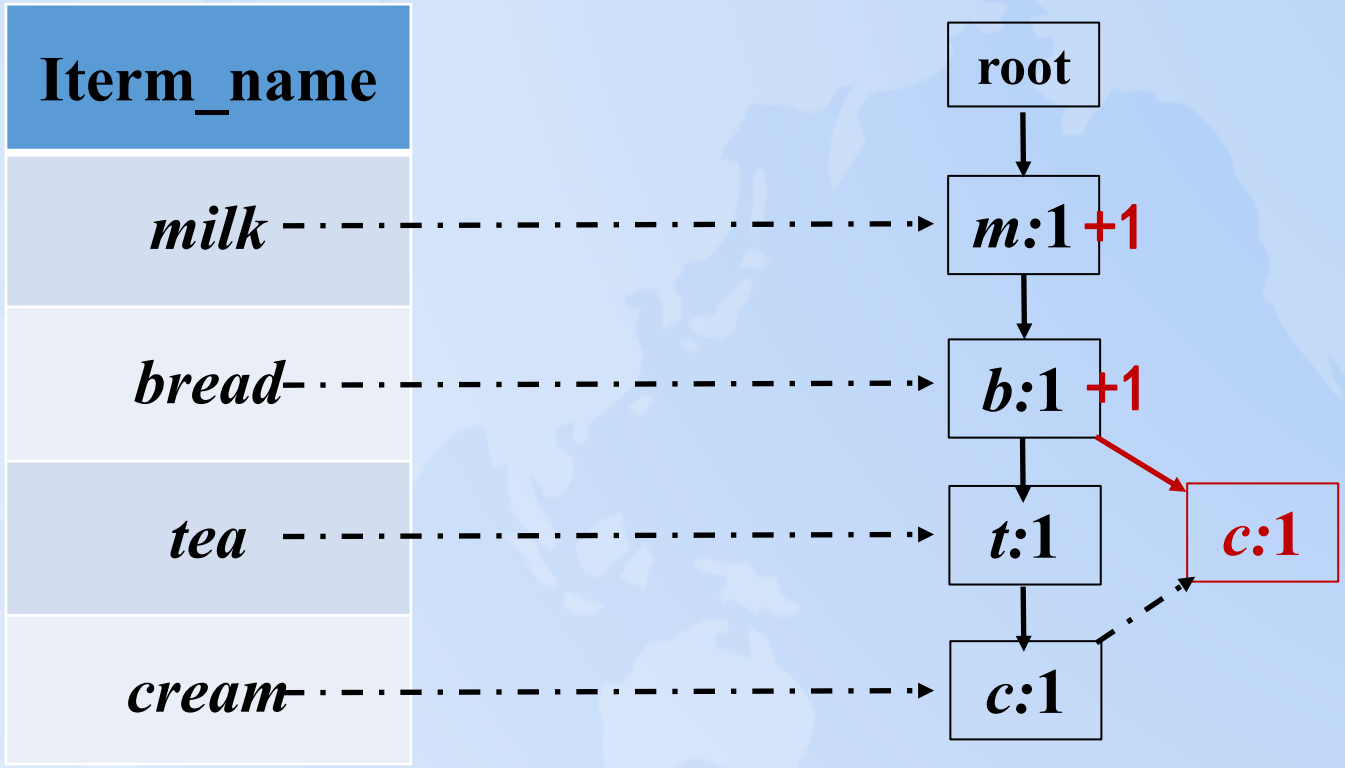
5. 构建FP树

创建一个标记为null的跟结点。开始对交易集的第二遍扫描。加入第一条清单T_1(*milk*, *bread*, *tea*, *cream*)。为了方便对树遍历，建立一个频繁项头表。



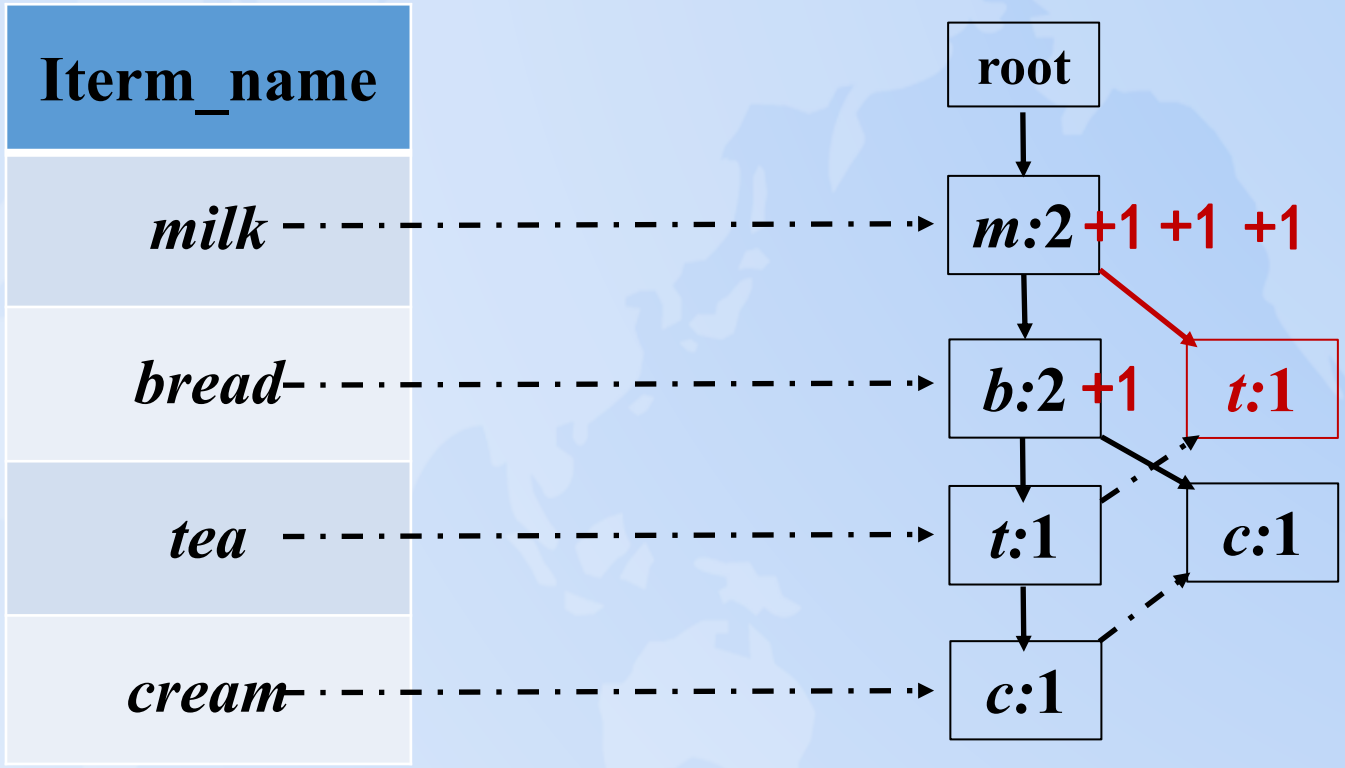
4.6 FP-growth算法

加入第二条清单T_2(*milk, bread, cream*)：
出现相同的节点(*milk, bread*)进行累加；
新结点(*cream*:1)被创建并且被作为结点(*bread*:2)的子结点。



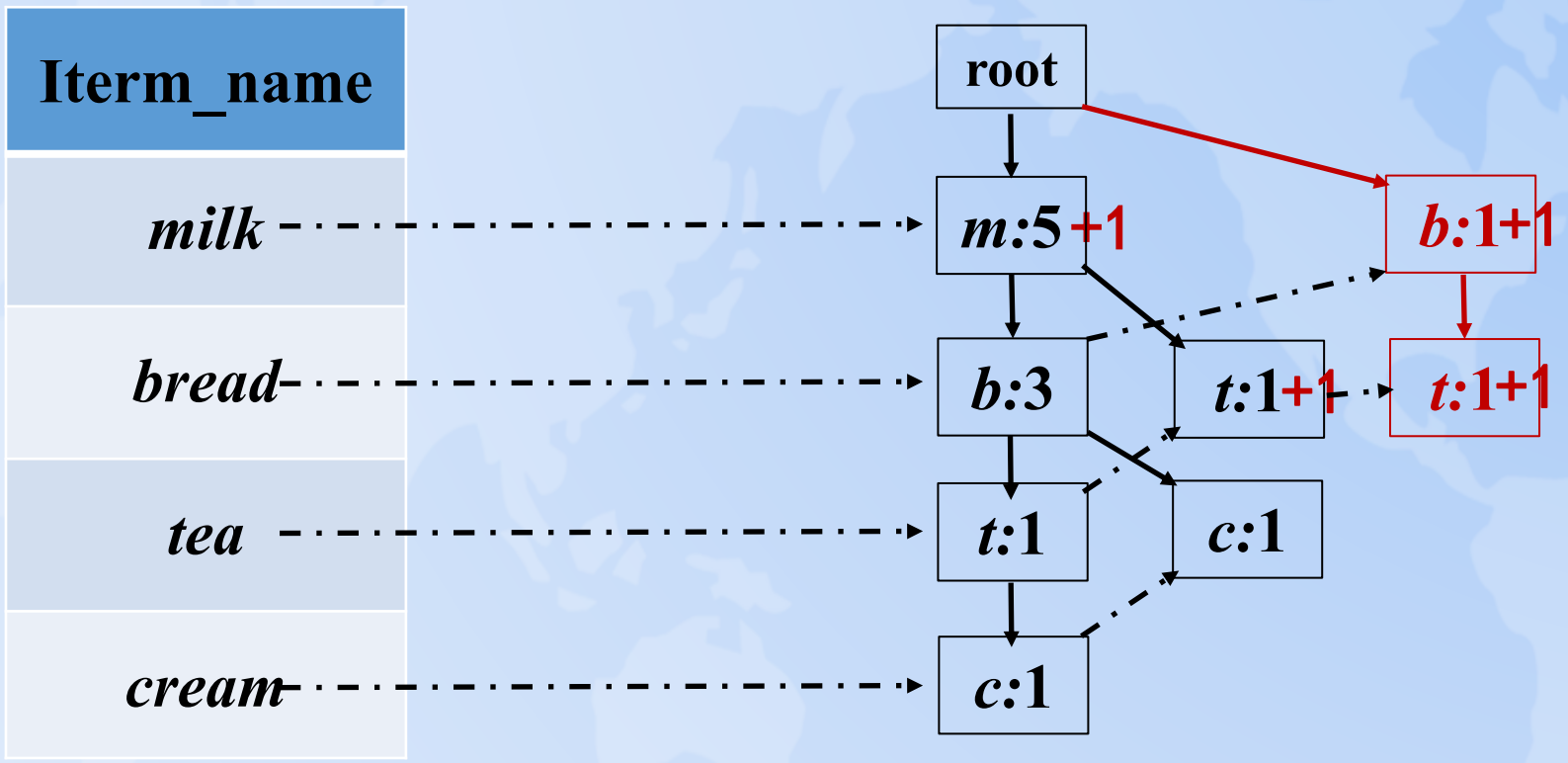
4.6 FP-growth算法

加入第三条清单T_3(*milk*)，出现相同的节点进行累加；
加入第四条清单T_4(*milk, tea*)，出现相同的节点进行累加；
加入第五条清单T_5(*milk, bread*)，出现相同的节点进行累加。



4.6 FP-growth算法

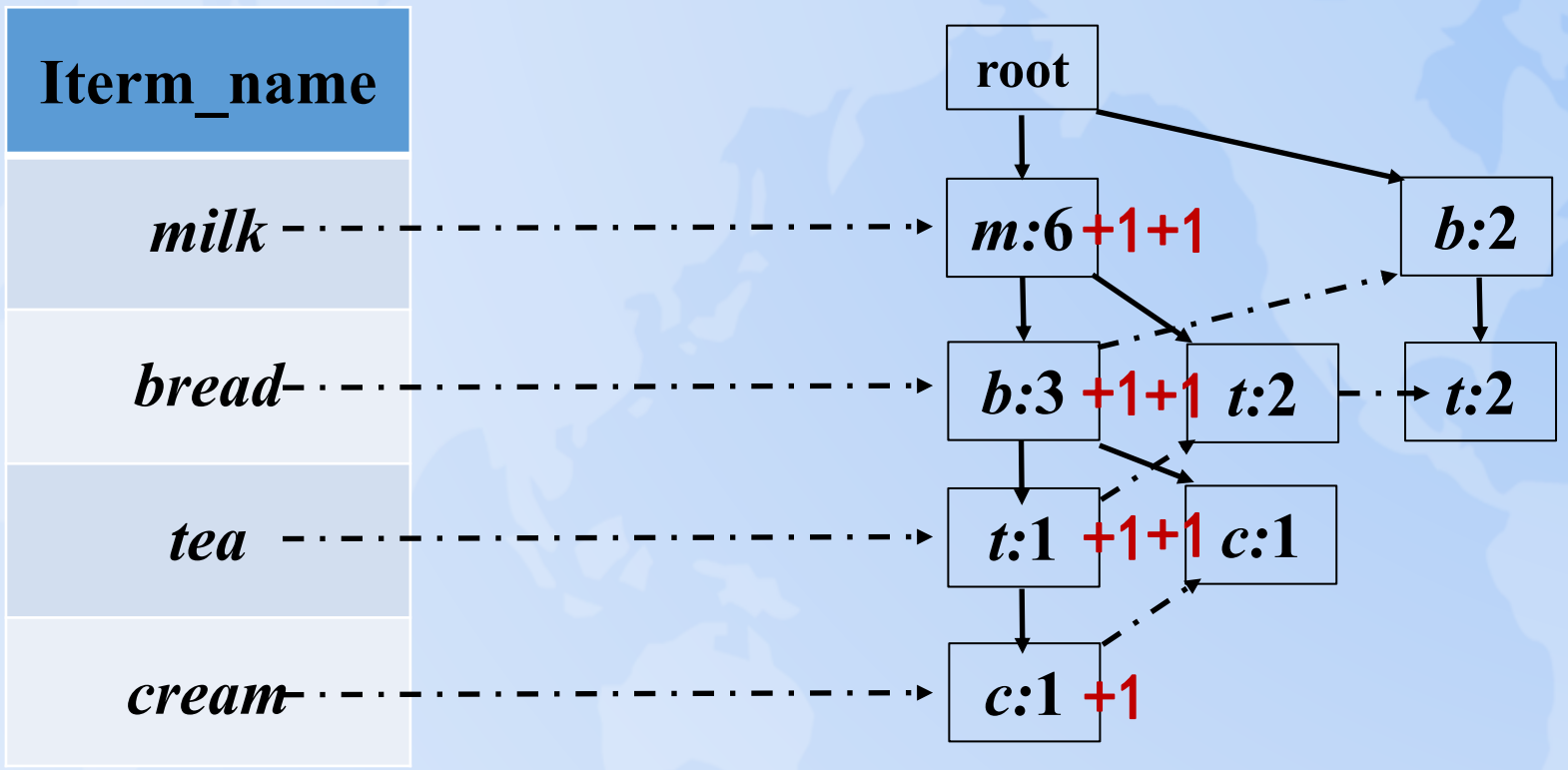
加入第六条清单T_6(*bread, tea*)；
加入第七条清单T_7(*milk, tea*)，出现相同的节点进行累加；
加入第八条清单T_8(*bread, tea*)，出现相同的节点进行累加。



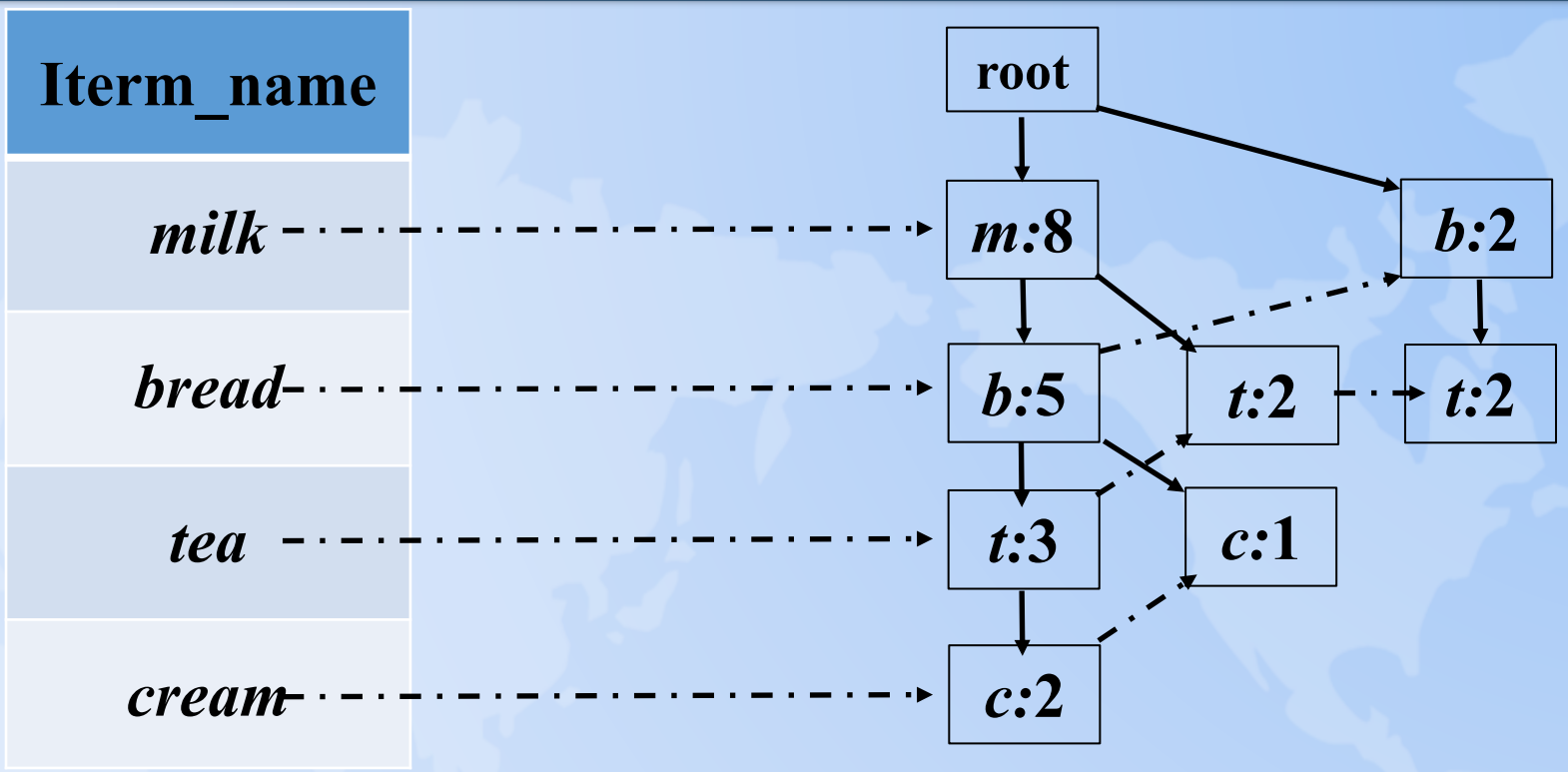
4.6 FP-growth算法

加入第九条清单T_9(*milk, bread, tea, cream*)，出现相同的节点进行累加；

加入第十条清单T_{10}(*milk, bread, tea*)，出现相同的节点进行累加。



4.6 FP-growth算法



包含*cream*的频繁集：{ c:3; cb:3; cm:3; cbm:3 }

包含*tea*的频繁集：{ t:7; tb:5; tm:5; tbm:3 }

包含*bread*的频繁集：{ b:7; bm:5 }

包含*milk*的频繁集：{ m:8 }