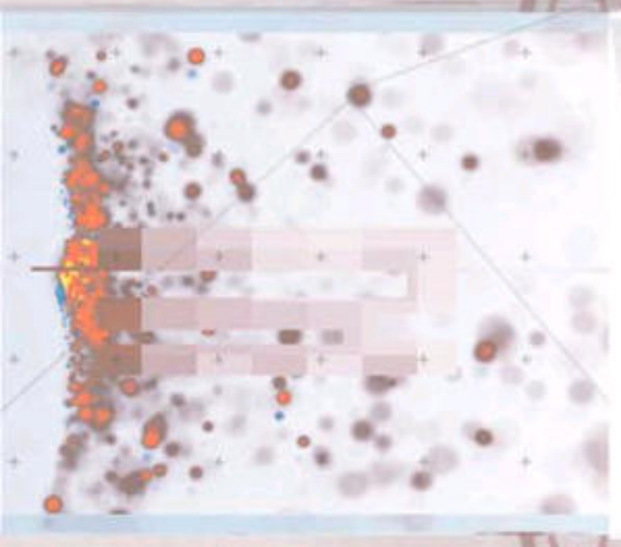


The Apriori Algorithm

Apriori: A Candidate Generation & Test Approach

- Outline of Apriori (level-wise, candidate generation and test)
  - Initially, scan DB once to get frequent 1-itemset
  - Repeat
    - Generate length-(k+1) candidate itemsets from length-k frequent itemsets
    - Test the candidates against DB to find frequent (k+1)-itemsets
    - Set k := k +1
  - Until no frequent or candidate set can be generated
  - Return all the frequent itemsets derived

The Apriori Algorithm (Pseudo-Code)

C_k: Candidate itemset of size k

F_k : Frequent itemset of size k

K := 1;
F_k := {frequent items}; // frequent 1-itemset
While (F_k != ∅) **do {** // when F_k is non-empty
  C_{k+1} := candidates generated from F_k; // candidate generation
  Derive F_{k+1} by counting candidates in C_{k+1} with respect to *TDB* at minsup;
  k := k + 1
  }
return $\cup_k F_k$ // return F_k generated at each level

The Apriori Algorithm—An Example

Database TDB

Tid	Items
10	A, C, D
20	B, C, E
30	A, B, C, E
40	B, E

minsup = 2

C_1 → 1st scan → Scan DB

Itemset	sup
{A}	2
{B}	3
{C}	3
{D}	1
{E}	3

F_1

Itemset	sup
{A}	2
{B}	3
{C}	3
{E}	3

C_{k+1} 从 F_k 中生成

C_2

Itemset
{A, B}
{A, C}
{A, E}
{B, C}
{B, E}
{C, E}

2nd scan → Scan DB

C_2

Itemset	sup
{A, B}	1
{A, C}	2
{A, E}	1
{B, C}	2
{B, E}	3
{C, E}	2

F_2

Itemset	sup
{A, C}	2
{B, C}	2
{B, E}	3
{C, E}	2

F_{k+1} 从 C_{k+1} 中生成

因为 {A, B} 不 frequent，所以 {A, B, C} 不可能 frequent。

C_3

Itemset
{B, C, E}

3rd scan → Scan DB

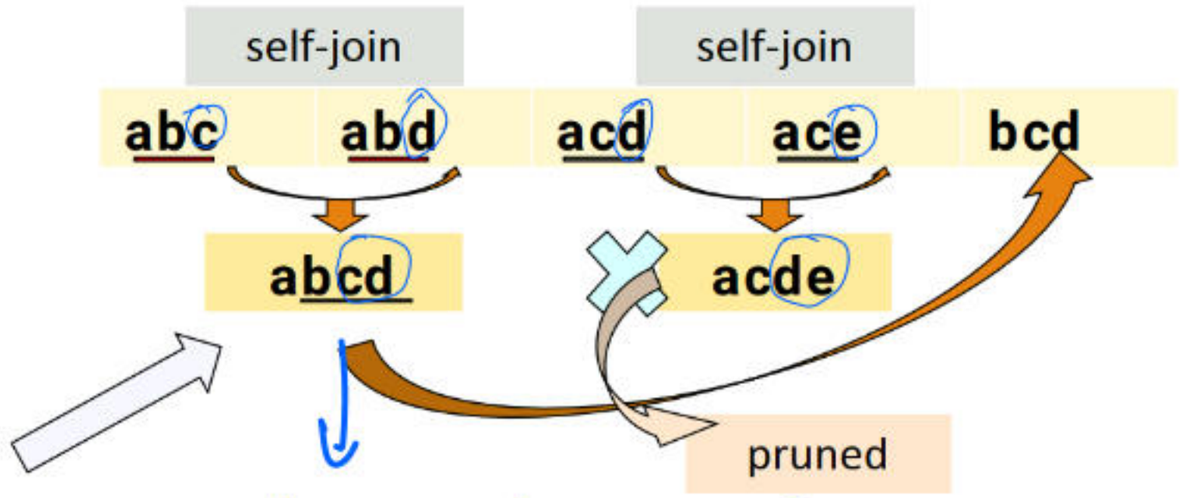
F_3

Itemset	sup
{B, C, E}	2

算法输出 F_1、F_2、F_3

Apriori: Implementation Tricks

- How to generate candidates?
  - Step 1: self-joining F_k 自连接
  - Step 2: pruning 剪枝
- Example of candidate-generation
  - F_3 = {*abc, abd, acd, ace, bcd*}
  - Self-joining: F_3*F_3
    - *abcd* from *abc* and *abd*
    - *acde* from *acd* and *ace*
  - Pruning:
    - *acde* is removed because *ade* is not in F_3
  - C_4 = {*abcd*}



abcd的 subset bcd 是 frequent.
∴ abcd 留了. 而 acde 中没有 subset 是 frequent, 所以 pruned.

Candidate Generation: An SQL Implementation

- Suppose the items in F_{k-1} are listed in an order
- Step 1: self-joining F_{k-1}

  insert into C_k

  select $p.item_1, p.item_2, \ldots, p.item_{k-1}, q.item_{k-1}$

  from F_{k-1} as p, F_{k-1} as q

  where $p.item_1 = q.item_1, \ldots, p.item_{k-2} = q.item_{k-2}, p.item_{k-1} < q.item_{k-1}$
- Step 2: pruning

  for all *itemsets c in* C_k do

  for all *(k-1)-subsets s of c* do

  if *(s is not in* F_{k-1}*)* **then delete** *c* **from** C_k