Direct Hashing and Pruning (Park-Chen-Yu)

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

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}

Is there some "magic" way to reduce the size of C_2 ?

Direct Hashing and Pruning

- Hash-based heuristic of generating candidate sets of high likelihood of being large itemsets
- Basic Idea:
  - Use hashing to filter out unnecessary itemsets for the next candidate itemset generation
- Implementation:
  - Accumulate information about (k+1)-itemsets in advance in such a way so that all possible (k+1)-itemsets of each transaction after some pruning are hashed into a hash table
    - Each bucket in the hash table consists of the count of itemsets that have been hashed into the bucket so far

Rule Generation

- Given a frequent itemset L, find all non-empty subsets $f \subset L$ such that $f \rightarrow L - f$ satisfies the minimum confidence requirement
  - If {A,B,C,D} is a frequent itemset, candidate rules:

    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

- If |L| = n, then there are 2n – 2 candidate association rules (ignoring $L \rightarrow \emptyset$ and $\emptyset \rightarrow L$)

Rule Generation

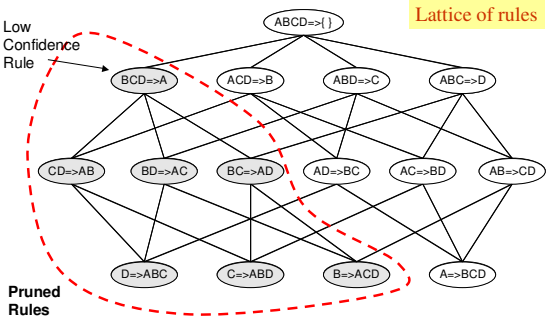
- How to efficiently generate rules from frequent itemsets?
  - In general, confidence does not have an anti-monotone property

    c(ABC →D) can be larger or smaller than c(AB →D)
  - But confidence of rules generated from the same itemset has an anti-monotone property

    e.g., L = {A,B,C,D}:

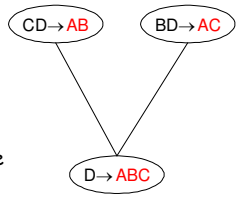
    $c(ABC \rightarrow D) \geq c(AB \rightarrow CD) \geq c(A \rightarrow BCD)$
  - Confidence is anti-monotone w.r.t. number of items on the RHS of the rule

Rule Pruning



Rule Generation

- Candidate rule is generated by merging two rules that share the same prefix in the rule consequent
- join(CD → AB, BD → AC) would produce the candidate rule D → ABC
- Prune rule D → ABC if its subset AD → BC does not have high confidence



Rule Generation Algorithm

Key fact:

Moving items from the antecedent to the consequent never changes support, and never increases confidence

Algorithm

- For each itemset I with *minsup*:
  - Find all *minconf* rules with a single consequent of the form $(I - L_1 \Rightarrow L_1)$
  - repeat:
    - Guess candidate consequents C_k by appending items from $I - L_{k-1}$ to L_{k-1}
    - Verify confidence of each rule $I - C_k \Rightarrow C_k$ using known itemset support values

Algorithm to Generate Association Rules

```
Input:
  D      //Database of transactions
  I      //Items
  L      //Large itemsets
  s      //Support
  α      //Confidence
Output:
  R      //Association Rules satisfying s and α
ARGen Algorithm:
  R = ∅;
  for each l ∈ L do
    for each x ⊂ l such that x ≠ ∅ and x ≠ l do
      if support(l) / support(x) ≥ α then
        R = R ∪ {x ⇒ (l − x)};
```


Factors Affecting Complexity

- Choice of minimum support threshold
  - lowering support threshold results in more frequent itemsets
  - this may increase number of candidates and max length of frequent itemsets
- Dimensionality (number of items) of the data set
  - more space is needed to store support count of each item
  - if number of frequent items also increases, both computation and I/O costs may also increase
- Size of database
  - since Apriori makes multiple passes, run time of algorithm may increase with number of transactions
- Average transaction width
  - transaction width increases with denser data sets
  - may increase max length of frequent itemsets

Compact Representation of Frequent Itemsets

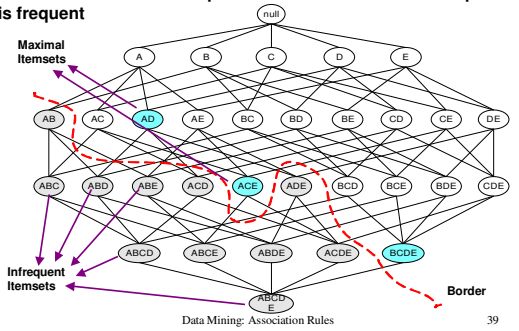
- Some itemsets are redundant because they have identical support as their supersets

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

- Number of frequent itemsets $= 3 \times \sum_{k=1}^{10} \binom{10}{k}$
- Need a compact representation

Maximal Frequent Itemset

An itemset is maximal frequent if none of its immediate supersets is frequent



Closed Itemset

- An itemset is closed if none of its immediate supersets has the same support as the itemset

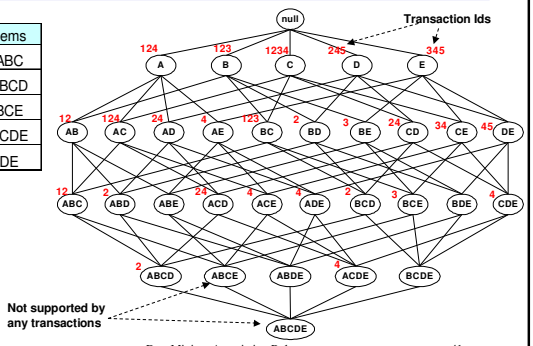
TID	Items
1	{A,B}
2	{B,C,D}
3	{A,B,C,D}
4	{A,B,D}
5	{A,B,C,D}

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

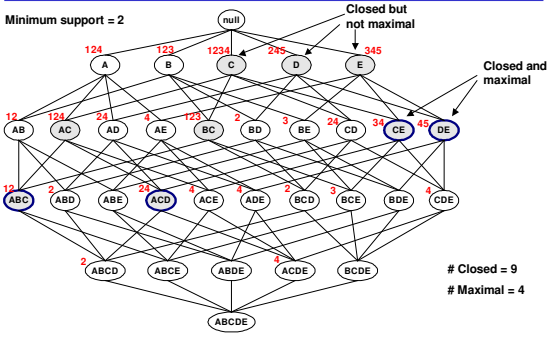
Itemset	Support
{A,B,C}	2
{A,B,D}	3
{A,C,D}	2
{B,C,D}	3
{A,B,C,D}	2

Maximal vs. Closed Itemsets

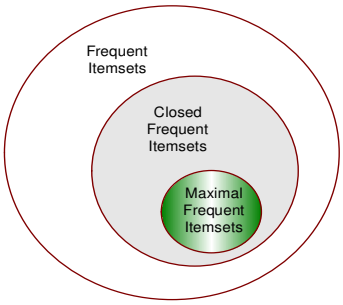
TID	Items
1	ABC
2	ABCD
3	BCE
4	ACDE
5	DE



Maximal vs. Closed Frequent Itemsets



Maximal vs. Closed Itemsets



Subsequent Research on Association Rules

- Mining association rules from sequences
  e.g. stocks with similar movements in stock prices, grocery items bought over a sequence of visits, etc.
- Finding "interesting" rules
  - Low-support, high-correlation mining
- Efficiently handling long itemsets
- Integration with query optimizers
- Adjustments to handle dense/relational databases
- Apply constraints to further filter association rules