TUTORIAL 1

- The following is a pseudocode of Apriori algorithm
- Generate frequent itemsets of length 1
- 2. Repeat until no new frequent itemsets are identified
 - Generate length (k+1) candidate itemsets from length k
 frequent itemsets
 - Prune candidate itemsets containing subsets of length k that are infrequent
 - c. Count the support of each candidate by scanning the DB
 - d. Eliminate candidates that are infrequent, leaving only those that are frequent

Lecture slide: Anti-Monotone Property

- Any subset of a frequent itemset must be also frequent — an anti-monotone property
 - Any transaction containing {beer, diaper, milk} also contains {beer, diaper}
 - {beer, diaper, milk} is frequent → {beer, diaper} must also be frequent
- In other words, any superset of an infrequent itemset must also be infrequent
 - No superset of any infrequent itemset should be generated or tested
 - Many item combinations can be pruned!

Answer

- Suppose minsup =4 we have the following frequent items:
 - frequent items a,b,c,d,e,f
 - Frequent 2-itemsets: ab 4, ad 5, ae 4, bd 4, bf 4, de 4, df 4
 - If we generate 3 itemsets: abd?
 - If we generate 3 itemsets: abe?

we introduce how to generate length (k+1) candidate itemsets from length k frequent itemsets. Explain this with an example. Can give another way of generating candidates?

Candidate Generation: $F_{k-1} \times F_{k-1}$ Method

■ To generate C_{k+1} from F_k : Merge two frequent (k)-itemsets if their first (k-1) items are identical

- F₃ = {ABC,ABD,ABE,ACD,BCD,BDE,CDE}
 - Merge($\underline{AB}C$, $\underline{AB}D$) = $\underline{AB}CD$
 - Merge($\underline{AB}C$, $\underline{AB}E$) = $\underline{AB}CE$
 - Merge($\underline{AB}D$, $\underline{AB}E$) = $\underline{AB}DE$

Any k itemset subsets must be frequent. So the two kitemsets with the identical k-1 items should be frequent.

 Do not merge(<u>ABD</u>,<u>ACD</u>) because they share only prefix of length 1 instead of length 2

- Different ways of generating candidates.
 - They may generate different sets of candidate.
 - The real frequent itemsets must be included.
 - but after pruning, they will be the same.

Candidate Generation: Merge Fk-1 and F1 itemsets

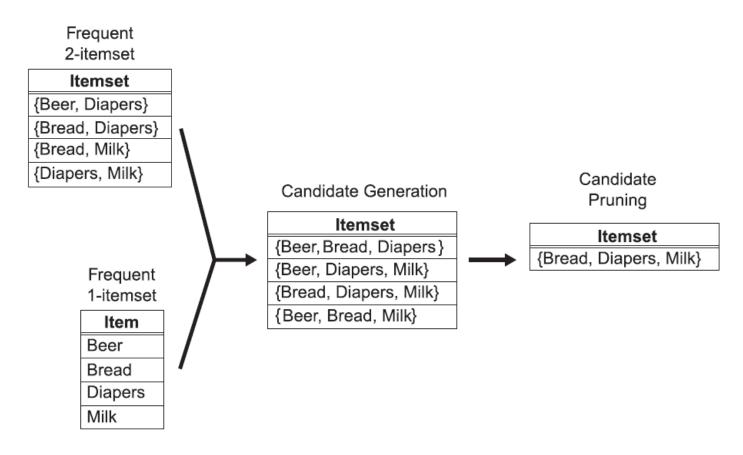


Figure 5.7. Generating and pruning candidate k-itemsets by merging a frequent (k-1)-itemset with a frequent item. Note that some of the candidates are unnecessary because their subsets are infrequent.

More example

- F₃ = {ABC,ABD,ABE,ACD,BCD,BDE,CDE}
- F1= {A, B, C, D, E}
- We generate
 - ABC: ABCD, ABCE
 - ABD:ABDE
 - ABE:
 - ACD: ACDE
 - BCD: BCDE
 - BDE:
 - CDE:

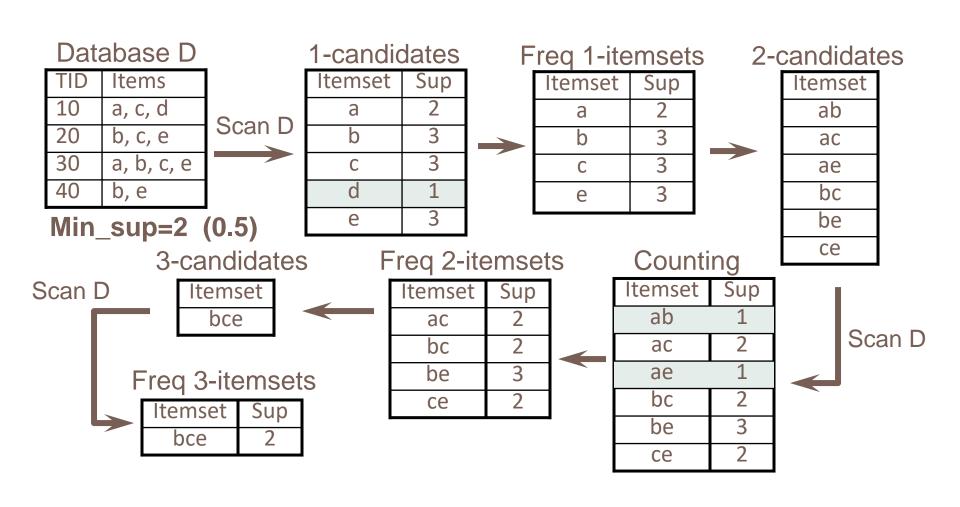
After prunning C₄ = {ABCD}

Q3 finding frequent itemsets

- □ 1: a, c,d
- □ 2, b, c,e
- □ 3, a, b, c, e
- □ 4, b, e,

□ Minsup =2

Example: Apriori-based Mining



□ Suppose {B,C,D} is a frequent itemset. Enumerate the candidate rules:

Step2: Rule generation

- Step 1: Find all frequent itemsets I
 - Generate all itemsets whose support ≥ minsup
- Step 2: Rule generation
 - Given a frequent itemset I, find all non-empty subsets A $\subset I$ such that $A \to I A$ satisfies the minimum confidence requirement minconf
 - If {A,B,C,D} is a frequent itemset, candidate rules:

```
ABC \rightarrowD, ABD \rightarrowC, ACD \rightarrowB, BCD \rightarrowA, A \rightarrowBCD, B \rightarrowACD, C \rightarrowABD, D \rightarrowABC AB \rightarrowCD, AC \rightarrow BD, AC, BD \rightarrowAC, CD \rightarrowAB
```

- An example to compute the rule confidence
 - confidence($A,B \rightarrow C,D$) = support(A,B,C,D) / support(A,B)
- Do the above for every frequent itemset, and output all the rules above the confidence threshold

• If {B,C,D} is a frequent itemset, candidate rules:

 $BC \rightarrow D$, $BD \rightarrow C$, $CD \rightarrow B$

 $B \rightarrow CD$, $C \rightarrow BD$, $D \rightarrow BC$

□ Consider the observation: If A,B,C→D is below confidence, so is A,B→C,D. Can we design an efficient order of generating rules based on the observation?

Prunning in Rule Generation

- Confidence does not have an anti-monotone property c(ABC →D) can be larger or smaller than c(AB →D)
- How to prune?
 - Confidence of rules generated from the same itemset has an anti-monotone property
 - E.g., Suppose {A,B,C,D} is a frequent 4-itemset:

$$c(ABC \rightarrow D) \ge c(AB \rightarrow CD) \ge c(A \rightarrow BCD)$$

Confidence is anti-monotone w.r.t. number of items on the RHS of the rule

- Observation: If $A,B,C \rightarrow D$ is below confidence, so is $A,B \rightarrow C,D$
- Can generate "bigger" rules from smaller ones (RHS)!

We check c(ABC \rightarrow D), next c(AB \rightarrow CD), followed by c(A \rightarrow BCD)

Rule Generation for Apriori Algorithm

