

Exercise Sheet 3

Exercise 10 Apriori Algorithm

- a) Execute the Apriori algorithm (using the pseudo-code from the slides) for the transaction database shown on the right and a minimum support of $s_{\min} = 3!$
- | | |
|-----------|-----------|
| $a d f$ | $a b d$ |
| $a c d e$ | $b d e$ |
| $b d$ | $b c e g$ |
| $b c d$ | $c d f$ |
| $b c$ | $a b d$ |
- b) Execute the Apriori algorithm again, using the organization of the counters as a prefix tree!
- c) How is support counted in the (pseudo-code) Apriori algorithm? What are the advantages/disadvantages of this counting scheme? How is it counted if the support counters are organized as a prefix tree? (Hint: Consider subset tests versus generating and finding subsets.)
- d) What are the advantages of organizing the transactions as a prefix tree? (Hint: What redundant work can be avoided by this scheme?)
- (Attention: Do not confuse the prefix tree for organizing the support counters with the prefix tree that is used to represent the transaction database!)

Exercise 11 Divide-and-Conquer Search

- a) How can the search for frequent item sets be described as a divide-and-conquer algorithm? How does this give rise to a recursive search scheme?
- b) How can the divide-and-conquer scheme be described formally as a set of subproblems? How is each subproblem described?
- c) How is a subproblem processed? Especially, how are the elements of the subproblems it is split into derived from the subproblem that is processed?
- d) Why are the two transaction databases that are needed for the two subproblems of a split *not* complementary?

Exercise 12 Perfect Extensions

- a) What is a perfect extension? How is it defined formally?
- b) Let a and b be perfect extensions of an item set I and let $c \notin I$ be some other item (not a perfect extension).
Is a a perfect extension of $I \cup \{b\}$? Is a a perfect extension of $I \cup \{c\}$?
Is b a perfect extension of $I \cup \{a\}$? Is b a perfect extension of $I \cup \{c\}$?
What can be seen generally from the answers to the above questions?
- c) If in the divide-and-conquer scheme the split item is a perfect extension, how are the solutions of the two subproblems related to each other? Why are they related in this way? (Hint: Consider the respective transaction databases.)
- d) Based on this insight, how can perfect extensions be used to simplify the divide-and-conquer search? (Hint: Does one always have to solve both subproblems?)

Exercise 13 Transaction Database Representation

a) Construct horizontal and vertical representations of the matrix

	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>e</i>	<i>f</i>	<i>g</i>
1:	1	0	0	1	0	1	0
2:	1	0	1	1	1	0	0
3:	0	1	0	1	0	0	0
4:	0	1	1	1	0	0	0
5:	0	1	1	0	0	0	0
6:	1	1	0	1	0	0	0
7:	0	1	0	1	1	0	0
8:	0	1	1	0	1	0	1
9:	0	0	1	1	0	1	0
10:	1	1	0	1	0	0	0

- b) Can one execute the Apriori algorithm with a vertical transaction representation?
- c) Can one use a horizontal transaction representation in the divide-and-conquer approach to frequent item set mining?
- d) Characterize a prefix tree representation of a transaction database in terms of horizontal/vertical representations!