1)

a) This algorithm progressively builds longer and longer chains of subsets that retain minimum support for association. In order to keep the the number of new candidate items sets relatively small in each pass of the algorithm the the Apiori algorithm employs a dynamic programing tactic to use prior known large sets to be built on. This ordered chaining allows for only adding candidate sets that extend the previously known large chain while preventing infrequent sets. For example, given an L3={{1,2,3}, {1,2,4}, {3,5,6}}, not matching the first k-1 items in order would result in a series of infrequent sets that could not possibly meet minimum support to be kept.

b) The one item sets with minimum support of 10% are:

Candidate	Support
{A}	0.47
{B}	0.83
{C}	0.43
{D}	0.25

c) Candidates:

Candidate	Support
{A,B}	0.34
{A,C}	0.11
{A,D}	0.05
{B,C}	0.34
{B,D}	0.18
{C,D}	0.10

d) Dropped {A,D} due to lack of minimum support:

Candidate	Support
{A,B}	0.34
{A,C}	0.11
{B,C}	0.34
{B,D}	0.18
{C,D}	0.10

e) Candidate 3-item sets

Do join to consider candidates {A,B,C} and {B,C,D}. Candidates starting with C are excluded because there are no further options to match the common prefix of C with.

Prune step removes 0 candidates because each L3 item set is composed of subsets represented in L2.

Calculating minumum support on L3:

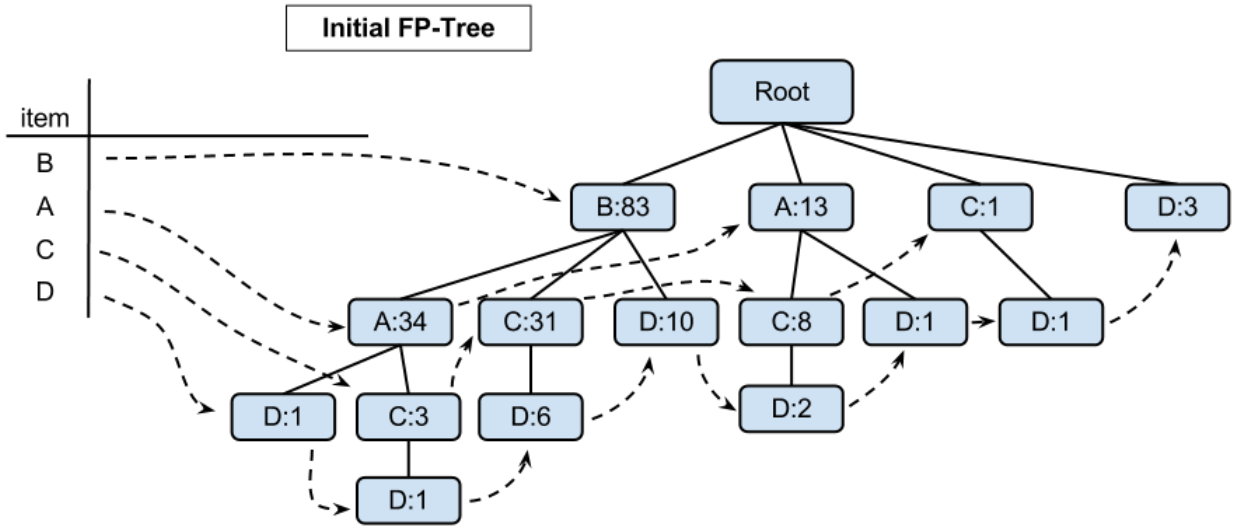
Candidate	Support
{A,B,C}	0.03
{B,C,D}	0.07

Since neither candidate 3-item set has minimum support use only the candidates from L2 in rule generation.

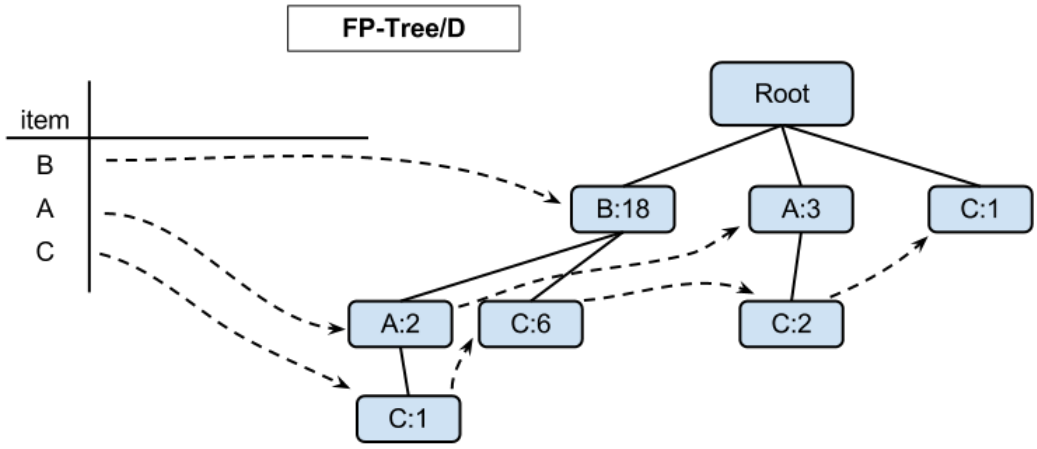
Candidate	Conf	Support
{A => B}	0.72	0.34
{B => A}	0.39	0.34
{A => C}	0.23	0.11
{C => A}	0.26	0.11
{B => C}	0.40	0.34
{C => B}	0.79	0.34
{B => D}	0.22	0.18
{D => B}	0.72	0.18
{C => D}	0.23	0.10
{D => C}	0.40	0.10

2)

a)



b)



2 sub-trees generated from the FP-Tree { (c:1), (d:1) } and { (d:3) }

c)

Itemsets:

Set	Support	
{B, A, C, D}	0.01	
{B, A, D}	0.02	
{B, D}	0.18	<- only frequent combination (minS >= 0.1)
{B, C, D}	0.06	
{A, C, D}	0.02	
{A, D}	0.03	
{C, D}	0.01	