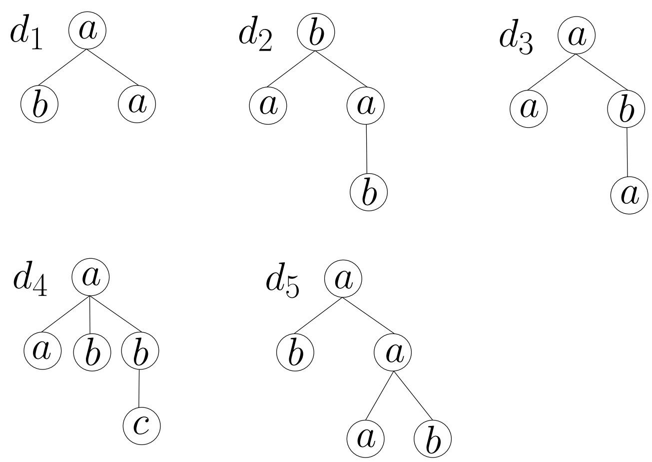
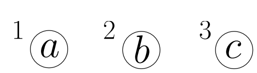
Data Mining: Frequent Tree Mining

Consider the following database of labeled ordered trees:



We aim to find all frequent ordered induced subtrees with $\sigma = 0.6$, i.e. a subtree is frequent if it occurs in at least three of the five data trees. The FREQT algorithm performs a level wise search starting with trees consisting of a single labeled node. To generate candidate frequent trees for level $k+1$ we add a frequent node to a frequent tree of size k, using the rightmost extension technique. Nodes in a tree are assumed to be numbered according to the pre-order traversal of the tree. At level 1 we have the following three candidates:

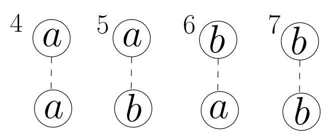


The following table contains the information for counting. The column numbers refer to the numbers of the candidate trees as given above. Each entry of the table contains the right-most occurrence list (RMO-list) of a candidate in a data tree. This is a list of node numbers in the data tree to which the right-most leaf of the candidate tree can be mapped. A candidate subtree is also called a pattern tree (as opposed to a data tree, which is a tree in the database).

	(1)	(2)	(3)
d_1	(1,3)	(2)	–
d_2	(2,3)	(1,4)	–
d_3	(1,2,4)	(3)	–
d_4	(1,2)	(3,4)	(5)
d_5	(1,3,4)	(2,5)	–
Support	5	5	1
Frequent?	Y	Y	N

It turns out that level 1 candidate (3), i.e. the single node with label c, is not frequent. Therefore it will not be extended, and neither will it be used to extend frequent trees.

At level 2 we have the following candidates:

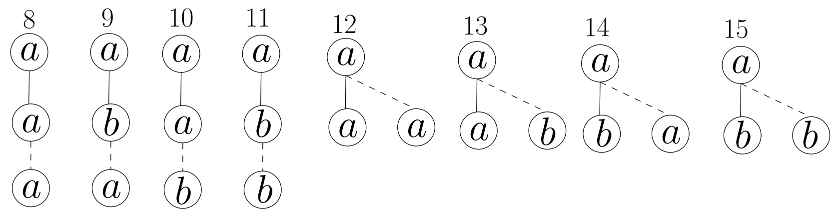


The RMO-lists are:

	(4)	(5)	(6)	(7)
d_1	(3)	(2)	–	–
d_2	–	(4)	(2,3)	–
d_3	(2)	(3)	(4)	–
d_4	(2)	(3,4)	–	–
d_5	(3,4)	(2,5)	–	–
Support	4	5	2	0
Frequent?	Y	Y	N	N

As an example, let us consider how the RMO-list of pattern tree (4) in data tree d_1 is determined. First of all, we note that (4) is an extension of (1), where we added a node labeled a to the rightmost leaf of (1). To determine the RMO-list of pattern tree (4) in d_1, we consider each element of the RMO-list of pattern tree (1), jump to that node in the data tree, and check whether it has a child with label a. If it does, we add the node number of that child to the RMO-list of pattern tree (4). So to process the first element of the RMO-list of pattern tree (1) in d_1, we jump to node 1 in d_1 (recall that the nodes are numbered according to pre-order traversal), and check whether it has a child node labeled a. As it turns out, it does, and we add its node number (which is 3) to the RMO-list of pattern tree (4). The second element of the RMO-list of pattern tree (1) is 3, so we jump to node 3 in data tree d_1 and check whether it has a child with label a. This is not the case, so we don't add anything to the RMO-list of pattern tree (4) in d_1. Now we have processed all elements of the RMO-list of pattern tree (1) in d_1, so we are done.

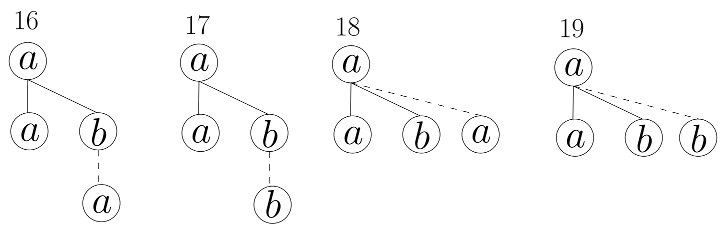
The level 3 candidates are:



The RMO-lists are:

	(8)	(9)	(10)	(11)	(12)	(13)	(14)	(15)
d_1	–	–	–	–	–	–	(3)	–
d_2	–	–	–	–	–	–	–	–
d_3	–	(4)	–	–	–	(3)	–	(4)
d_4	–	–	–	–	–	(3,4)	–	–
d_5	(4)	–	(5)	–	–	(5)	(3)	–
Support	1	1	1	0	0	3	2	1
Frequent?	N	N	N	N	N	Y	N	N

The level 4 candidates are:



The RMO-lists are:

	(16)	(17)	(18)	(19)
d_1	–	–	–	–
d_2	–	–	–	–
d_3	(4)	–	–	–
d_4	–	–	–	(4)
d_5	–	–	–	–
Support	1	0	0	1
Frequent?	N	N	N	N

As a final example, let us consider how the RMO-list of pattern tree (19) in d_4 is determined. We note that (19) is an extension of (13), so we process the RMO-list of (13) in d_4, which is $(3, 4)$. Pattern tree (19) is obtained from (13) by adding a node labeled b as the rightmost child to the parent of the rightmost child of (13). To process an element of the

RMO-list, we jump to that node in the data tree, jump to its parent, and check whether it has a child labeled b that is to the right of the element of the RMO-list. So to process the first element of the RMO-list, we jump to node 3 in d_4, then jump to its parent (node 1), and check whether node 1 has a child labeled b that is to the right of node 3. It does, namely node 4, so node 4 is added to the RMO-list of pattern tree (19) in d_4. To process the second element, we do the same thing and find out that this does not lead to success. Now all elements of the RMO-list of (13) in d_4 have been processed, so we have established the RMO-list of (19) in d_4 to be (4).

Since all level 4 candidates are infrequent, there are no level 5 candidates.

As the final result, the algorithm returns all frequent induced subtrees and their support: