Flow patterns and the generalized spatio-temporal patterns can easily be modeled as trees/graphs with each vertex representing a variable labeled by the event, and each edge representing a spatial relationship, temporal relationship, or both. Take the flow pattern \( \langle F(l_{32}), F(l_{38}) \rangle \rightarrow F(l_{32}) \rightarrow F(l_{34}) \) as an example, the corresponding tree representation of the pattern is shown in Figure 10.1. In this tree representation, we model the “precede” temporal relationship as an edge.

Figure 10.1. Tree representation of a flow pattern
With suitable modeling, all spatio-temporal databases can be correspondingly transformed into a tree or graph database, and the problem of mining frequent spatio-temporal patterns then becomes the problem of finding frequent tree/sub-graphs. In this chapter, we examine the problem of mining spatio-temporal tree patterns.

We observe that many spatio-temporal trees patterns are both unordered and embedded. Unordered refers to the condition that the sequence between children of the same parent is not important (for example, the tree with bedroom1 as the left son and bedroom2 as the right son is the same as the tree with bedroom2 as the left son and bedroom 1 as the right son) while embedded suggests that it is not necessary to strictly keep the parent-child relations among nodes.

Existing tree mining algorithms treat data trees as either ordered or induced (Asai, Abe, & Kawasoe, 2002; Asai, Arimura, & Uno, 2003; Chi, Yang, & Muntz, 2003, 2004; Chi, Yang, & Xia, 2004; Miyahara, Shoudai, & Uchida, 2001; Nijsse & Kok, 2003; Wang, Hong, & Pei, 2004; Wang & Liu, 2000; Xiao, Yao, & Li, 2003; Yang, Lee, & Hsu, 2003; Zaki, 2002). In these cases, some candidates may not be correctly counted and interesting patterns may be lost. Compared to the other three classes of tree mining algorithms (i.e., the ordered induced sub-tree mining algorithms, the ordered embedded sub-tree mining algorithms, and the unordered induced sub-tree mining algorithms), the unordered embedded sub-tree mining algorithms usually have much higher complexities for two reasons. First, they require more complicated matching functions which have been proven to be NP-complete (Kilpeläinen, 1992). Second, they generate a larger candidate set which significantly increases the number of matches and database traversals.

To address these issues, we present a divide-and-conquer method called WTIMiner to efficiently mine unordered embedded sub-tree patterns. We introduce the notion of mapping equivalent class, and develop a framework for mining frequent sub-trees via frequent itemset mining. The new algorithm suffers less from combinatorial explosion as compared to normal candidate-generate-and-test methods, with the number of database scans strictly bounded by the number of frequent itemsets generated. The proposed algorithm discovers the complete set of unordered and embedded sub-trees patterns, and can be easily modified to mine other types of sub-tree patterns. Experimental results confirm the efficiency of WTIMiner in both time and space.