Algorithm 9.3 Maximum cardinality search for constructing an elimination ordering

```
Procedure Max-Cardinality (
    H    // An undirected graph over X
)
1   Initialize all nodes in X as unmarked
2   for k = |X| ... 1
3     X ← unmarked variable in X with largest number of marked neighbors
4     π(X) ← k
5     Mark X
6   return π
```

Example 9.2

We can illustrate this construction on the graph of figure 9.11a. The maximal cliques in the induced graph are shown in b, and a clique tree for this graph is shown in c. One can easily verify that each sepset separates the two sides of the tree; for example, the sepset $\{G, S\}$ separates C, I, D (on the left) from L, J, H (on the right). The elimination ordering C, D, I, H, G, S, L, J, an extension of the elimination in table 9.1 that generated this induced graph, is one ordering that might arise from the construction of theorem 9.9. For example, it first eliminates C, D, which are both in a leaf clique; it then eliminates I, which is in a clique that is now a leaf, following the elimination of C, D. Indeed, it is not hard to see that this ordering introduces no fill edges. By contrast, the ordering in table 9.2 is not consistent with this construction, since it begins by eliminating the variables G, I, S, none of which are in a leaf clique. Indeed, this elimination ordering introduces additional fill edges, for example, the edge $H \rightarrow D$. ■

maximum cardinality

An alternative method for constructing an elimination ordering that introduces no fill edges in a chordal graph is the Max-Cardinality algorithm, shown in algorithm 9.3. This method does not use the clique tree as its starting point, but rather operates directly on the graph. When applied to a chordal graph, it constructs an elimination ordering that eliminates cliques one at a time, starting from the leaves of the clique tree; and it does so without ever considering the clique tree structure explicitly.

Example 9.3

Consider applying Max-Cardinality *to the chordal graph of figure 9.11. Assume that the first node selected is S. The second node selected must be one of S's neighbors, say J. The node that has the largest number of marked neighbors are now G and L, which are chosen subsequently. Now, the unmarked nodes that have the largest number of marked neighbors (two) are H and I. Assume we select I. Then the next nodes selected are D and H, in any order. The last node to be selected is C. One possible resulting ordering in which nodes are marked is thus S, J, G, L, I, H, D, C. Importantly, the actual elimination ordering proceeds in reverse. Thus, we first eliminate C, D, then H, and so on. We can now see that this ordering always eliminates a variable from a clique that is a leaf clique at the time. For example, we first eliminate C, D from a leaf clique, then H, then G from the clique $\{G, I, D\}$, which is now (following the elimination of C, D) a leaf.* ■

As in this example, Max-Cardinality always produces an elimination ordering that is consistent with the construction of theorem 9.9. As a consequence, it follows that Max-Cardinality, when applied to a chordal graph, introduces no fill edges.