BFS

DFS

\[
d(w) < f(w) \quad d(v) < f(v)
\]

Parentesis Theorem

\[
[([ )])
\]

\[
[ ([ ]]
\]

\[
[ ]([ ]
\]

\[
[ ( ) ]
\]

\[
[ ( [ )]
\]

\[
[ ([ )]
\]

\[
[ ( [ )]
\]
PERT chart

- Tree edges
- Back edges
- Cross edges
- Forward edges
22.4 Topological sort

(a) Professor Bumstead topologically sorts his clothing when getting dressed. Each directed edge \((u, v)\) means that garment \(u\) must be put on before garment \(v\). The discovery and finishing times from a depth-first search are shown next to each vertex. (b) The same graph shown topologically sorted, with its vertices arranged from left to right in order of decreasing finishing time. All directed edges go from left to right.

pants). A directed edge \((u, v)\) in the dag of Figure 22.7(a) indicates that garment \(u\) must be donned before garment \(v\). A topological sort of this dag therefore gives an order for getting dressed. Figure 22.7(b) shows the topologically sorted dag as an ordering of vertices along a horizontal line such that all directed edges go from left to right.

The following simple algorithm topologically sorts a dag:

**TOPOLOGICAL-SORT**(*G*)

1. call DFS(*G*) to compute finishing times \(v.f\) for each vertex \(v\)
2. as each vertex is finished, insert it onto the front of a linked list
3. return the linked list of vertices

Figure 22.7(b) shows how the topologically sorted vertices appear in reverse order of their finishing times.

We can perform a topological sort in time \(\Theta(V + E)\), since depth-first search takes \(\Theta(V + E)\) time and it takes \(O(1)\) time to insert each of the \(|V|\) vertices onto the front of the linked list.

We prove the correctness of this algorithm using the following key lemma characterizing directed acyclic graphs.