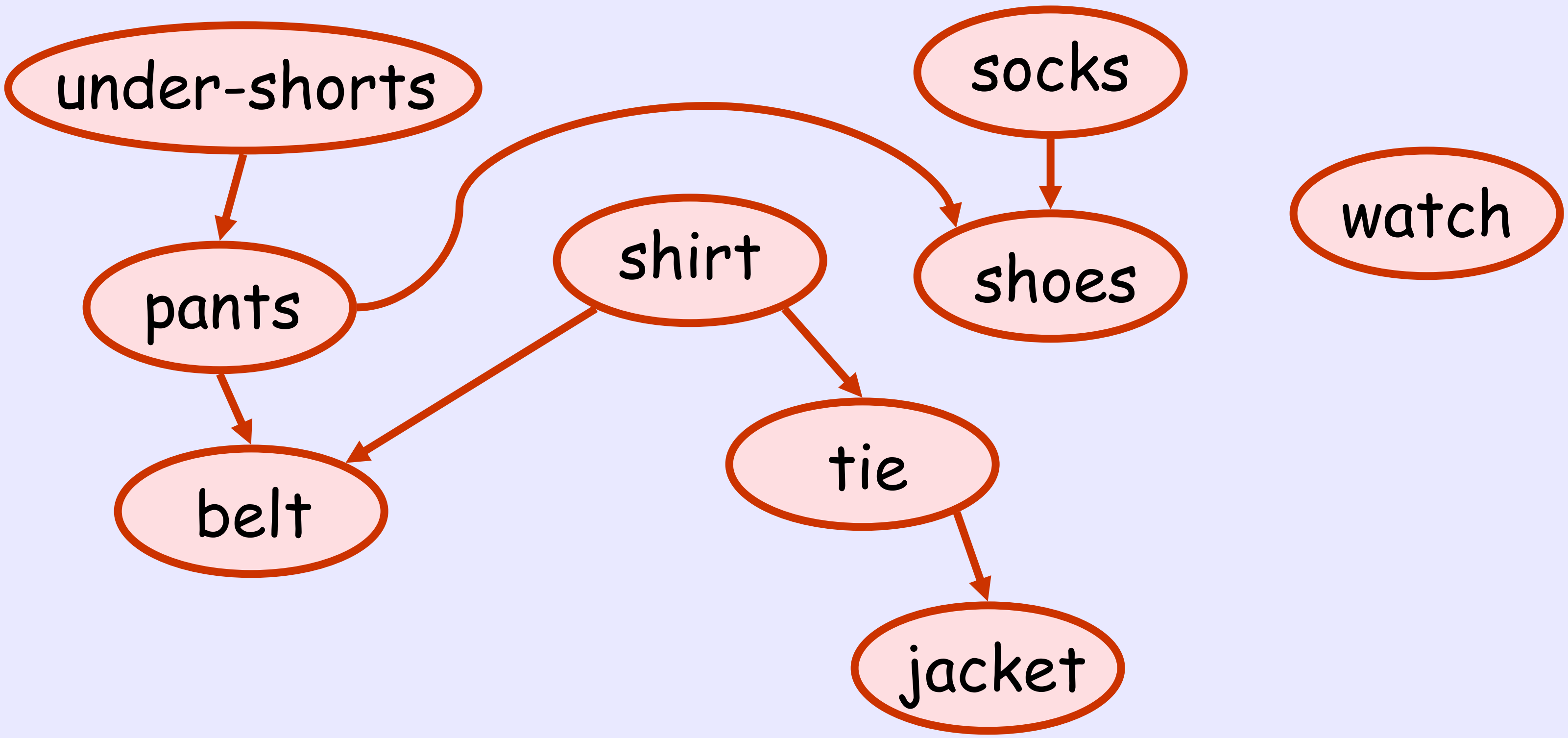
Chapter 20: Elementary Graph Algorithms III

About this lecture

- Topological Sort

Topological Sort

- Directed graph can be used to indicate precedence among a set of events
- e.g., a possible precedence is dressing



Topological Sort

- The previous directed graph is also called a precedence graph
- Question: Given a precedence (directed) graph G, can we order the events such that if (u, v) is in G (i.e., u should complete before v) then u appears before v in the ordering ?
- We call this problem topological sorting of G

Topological Sort

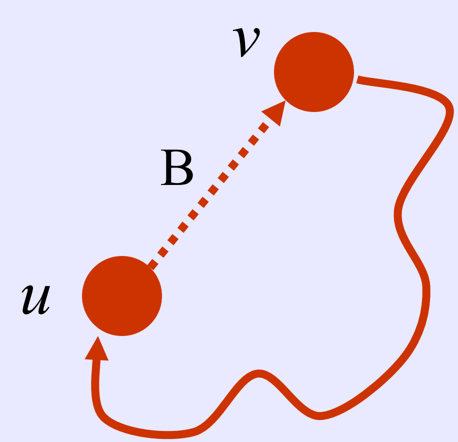
- Fact: If G contains a cycle, then it is impossible to find a desired ordering
- However, if G is acyclic (not contains any cycle) we show that the algorithm in next slide always find one of the desired ordering

Cycles in Directed Graph

- Theorem: For any DFS on a directed graph G, there is a back edge $\Leftrightarrow$ G has a cycle

Proof: $\Rightarrow$

- If there is a back edge (u, v), it implies there is a path from v to u. Thus, this back edge completes a cycle



Proof ($\Leftarrow$)

- If G has a cycle C, let v = first vertex discovered in C and (u, v) = v's preceding edge in C.

- Thus, when v is discovered, all nodes in C are still undiscovered (white)

  ➔ v is ancestor of u in DFS forest (why?)

  ➔ (u, v) becomes a back edge

Topological Sort

Topological-Sort(G)

{

1. Call DFS on G
2. If G contains a back edge, abort ;
3. Else, output vertices in decreasing order of their finishing times ;

}

Why is the algorithm correct?

Topological Sort

- Theorem: If G is acyclic, the previous algorithm produces a topological sort of G
- Proof: Let (u, v) be an directed edge in G. We shall show that $f(u) > f(v)$ so that u appears before v in the output ordering

Recall G is acyclic, there is no back edges.

There are two main cases …

Proof

- Case 1: (u, v) is a tree or forward edge

  → u is an ancestor of v

  → $d(u) < d(v) < f(v) < f(u)$ (why??)

- Case 2: (u, v) is a cross edge

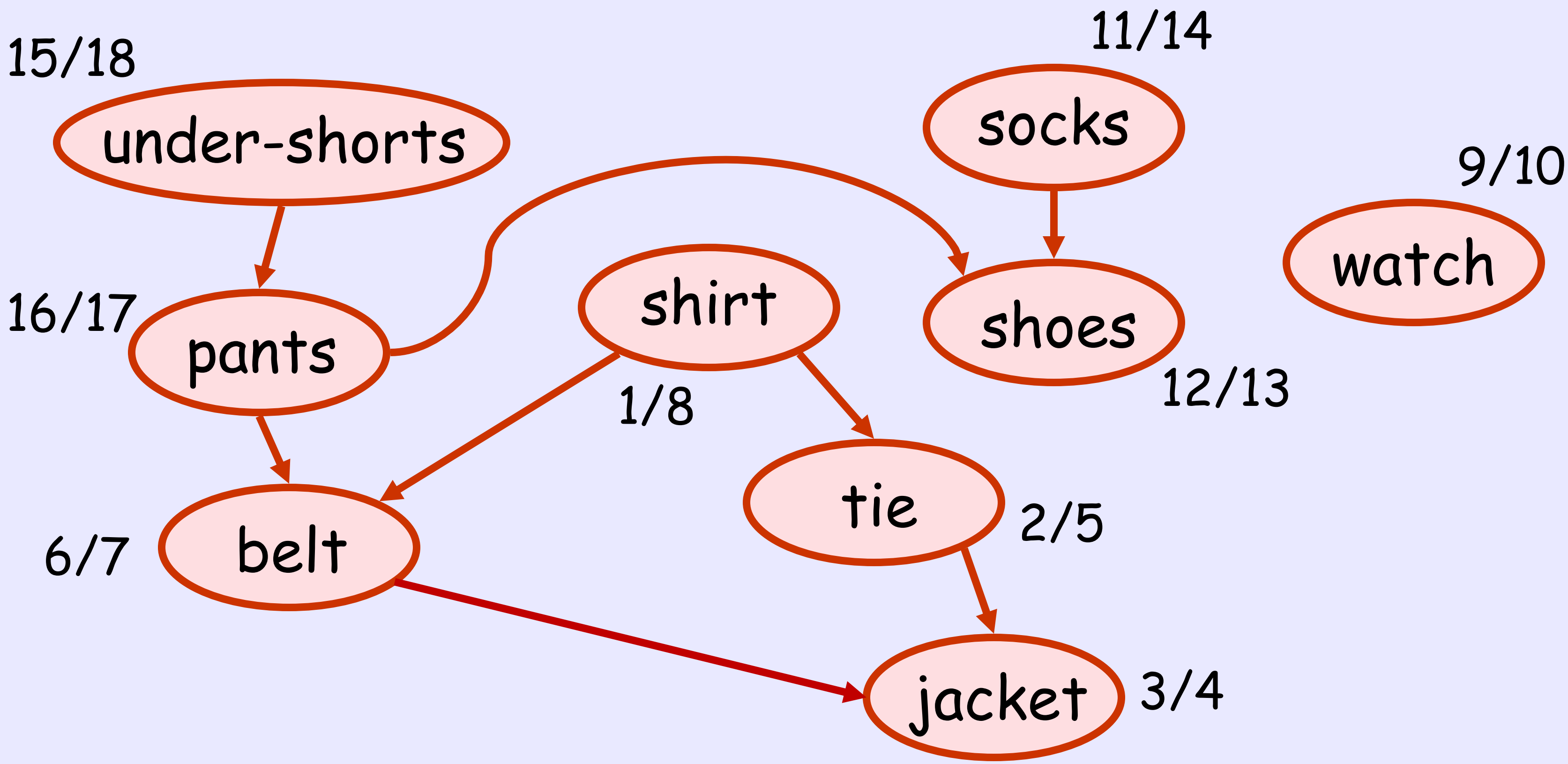
  → $d(v) < d(u)$ (otherwise, by white-path, u must be an ancestor of v, so that (u,v) cannot be a cross edge)

  → Since G is acyclic, v cannot reach u, so

  $$d(v) < f(v) < d(u) < f(u) \quad \text{(why??)}$$

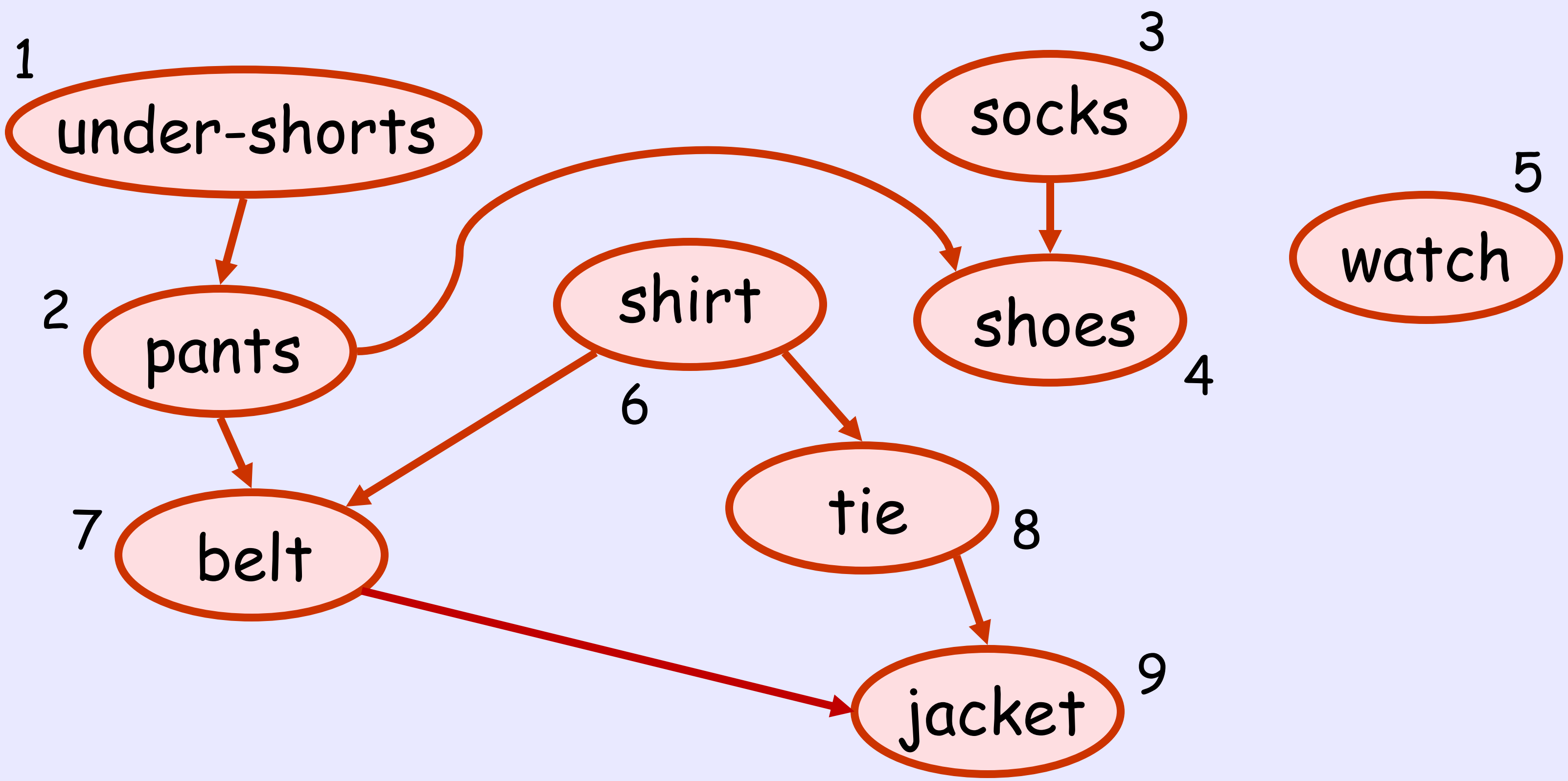
- Both cases show $f(u) > f(v)$ → Done !

Topological Sort (Example)



Discovery and Finishing Times
after a possible DFS

Ordering Finishing Times
(in descending order)



If we order the events from left to right, anything special about the edge directions ?

Performance

- Let $G = (V,E)$ be the input directed graph
- Running time for Topological-Sort :
  1. Perform DFS : $O(|V|+|E|)$ time
  2. Sort finishing times

     Naïve method: $O(|V| \log |V|)$ time

     Clever method: (use an extra stack S)

     During DFS, push a node into stack S once finished → no need to sort !!
- Total time: $O(|V|+|E|)$

Practice at home

- Exercises: 20.4-2, 20.4-3, 20.4-5