Given a set $J_a, J_b, J_c, J_d$ and some precedence constraints:

(i) $J_a < J_b$ : Job $a$ must be done prior to Job $b$

(ii) $J_a < J_d$

(iii) $J_d < J_c$

(iv) $J_c < J_b$

Can all jobs be scheduled?

$\checkmark$

Can all jobs be scheduled?

$\checkmark$

(i) $J_a < J_b$  (ii) $J_b < J_d$  (iii) $J_c < J_b$

(iv) $J_d < J_a$

Not feasible since $(J_a < J_b < J_d)$

there is cyclic precedence
Observation: There is no feasible schedule if there is a cyclic precedence between some subset of jobs.

If there is no cyclic precedence can we schedule?

Let us model it using a graph $V$: set of jobs, $E$: $(x, y)$ s.t. $x < y$

Do a DFS, always reveals if there is a cycle.

If there is no cycle in a directed graph $\Rightarrow$ Directed Acyclic Graph (DAG)
Observation: (i) There has to be a (source vertex) vertex with indegree 0
(ii) There must be a vertex with outdegree 0 (sink vertex)

A feasible schedule can be constructed by finding a source, deleting it, and repeating this in the remaining graph: $O(|V|^2)$

The precedence constraints define a "partial ordering" Schedule as a total ordering consistent with the partial ordering

Topological sort: Is an ordering of the vertices $V_1, V_2, V_3, \ldots, V_n$ such that for any edge $V_i \rightarrow V_j$, $j > i$
By keeping track of indegrees of vertices, can we get a faster algorithm? Which kind of data structure will yield a $O(V + E)$

Start line

Finish line

Decreasing finish line may work!
**Observation:** Consider a path in the DAG with vertices \( v_i (s_i, f_i) \) and \( v_j (s_j, f_j) \). Then, if \( v_i \rightarrow v_j \), then:

\[ f_i > f_j \]

**Case 1.** If \( s_i < s_j \), then it will visit \( v_j \) during the DFS. \( v_i \) and back up for \( v_j \) before \( v_i \):

\[ f_i > f_j \]

**Case 2.** If \( s_j < s_i \), then \( v_j \) precedes \( v_i \) in the DFS. Therefore:

\[ s_j < f_j < s_i < f_i \]
For any directed graph,

\[ v_i \rightarrow v_j \rightarrow v_i \quad \text{and} \quad v_i \rightarrow v_j \]

\[ v_i \rightarrow v_j \rightarrow v_i \]

\[ v_i \rightarrow v_j \]