ITMO UNIVERSITY

How to Win Coding Competitions: Secrets of Champions

Week 5: Algorithms on Graphs 1
Lecture 5: Topological sort

Maxim Buzdalov
Saint Petersburg 2016

Assume you are solving a problem from a competition.
What should you do?

Assume you are solving a problem from a competition.
What should you do?

Write the solution template

Get the names of I/O files

Read the statement

Write the solution

Invent the algorithm

Submit the solution

Assume you are solving a problem from a competition.
What should you do?
And in which order?

Write the solution template

Get the names of I/O files

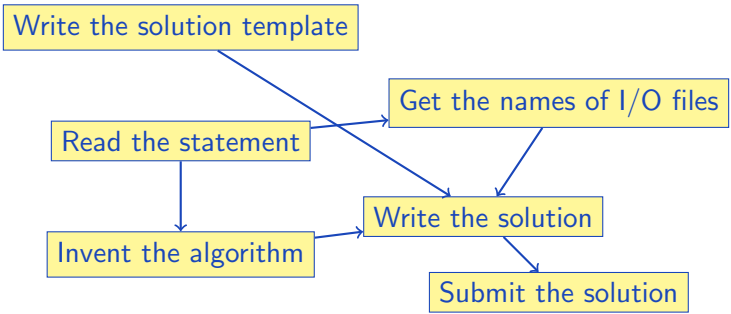
Read the statement

Write the solution

Invent the algorithm

Submit the solution

Assume you are solving a problem from a competition.
What should you do?
And in which order?

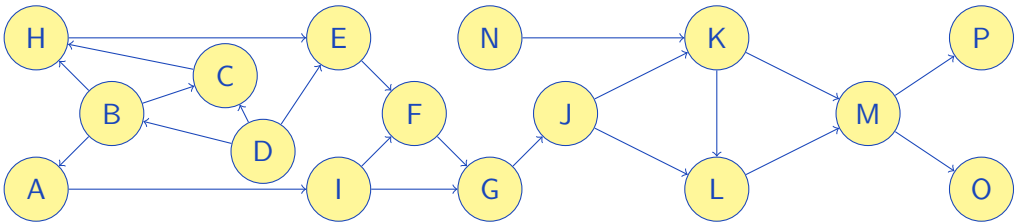


For the directed acyclic graph $G = \langle V, E \rangle$,
a topological sort is an assignment $I : V \to [1; N]$ of indices to vertices, such that:

- for every two distinct vertices u, v it holds that $I(u) \neq I(v)$
- for each edge $(u, v) \in E$ it holds that $I(u) < I(v)$

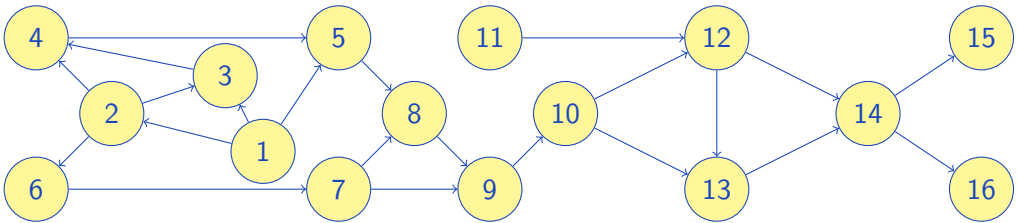
For the directed acyclic graph $G = \langle V, E \rangle$,
a topological sort is an assignment $I : V \to [1; N]$ of indices to vertices, such that:

- for every two distinct vertices u, v it holds that $I(u) \neq I(v)$
- for each edge $(u, v) \in E$ it holds that $I(u) < I(v)$



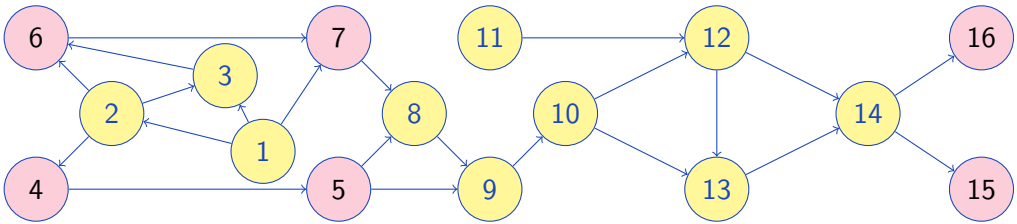
For the directed acyclic graph $G = \langle V, E \rangle$,
a topological sort is an assignment $I : V \to [1; N]$ of indices to vertices, such that:

- for every two distinct vertices u, v it holds that $I(u) \neq I(v)$
- for each edge $(u, v) \in E$ it holds that $I(u) < I(v)$



For the directed acyclic graph $G = \langle V, E \rangle$,
a topological sort is an assignment $I : V \to [1; N]$ of indices to vertices, such that:

- for every two distinct vertices u, v it holds that $I(u) \neq I(v)$
- for each edge $(u, v) \in E$ it holds that $I(u) < I(v)$

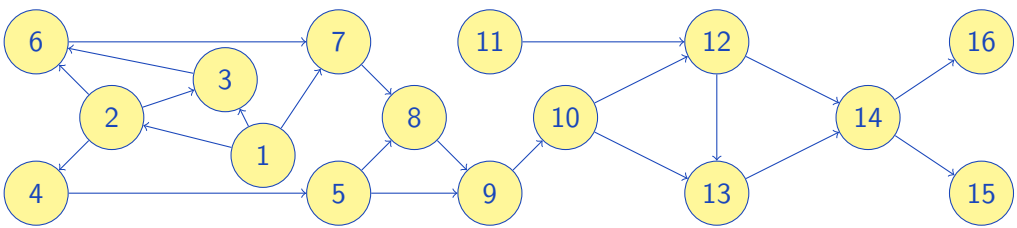


For the directed acyclic graph $G = \langle V, E \rangle$,
a topological sort is an assignment $I : V \to [1; N]$ of indices to vertices, such that:

- for every two distinct vertices u, v it holds that $I(u) \neq I(v)$
- for each edge $(u, v) \in E$ it holds that $I(u) < I(v)$

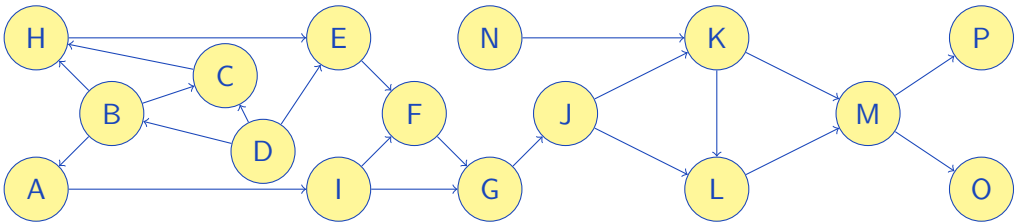
Sometimes it is convenient to have an array $o[i]$ such that:

- $o[I(v)] = v$, or, just the same, $I(o[i]) = i$



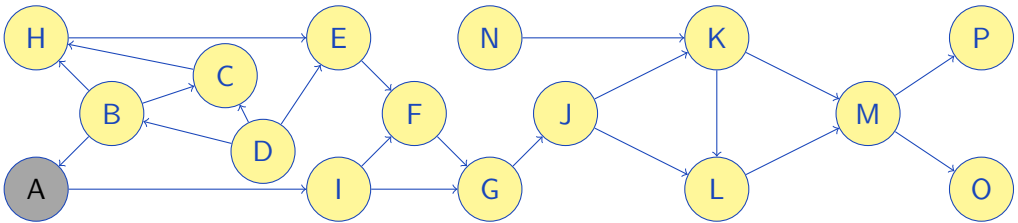
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



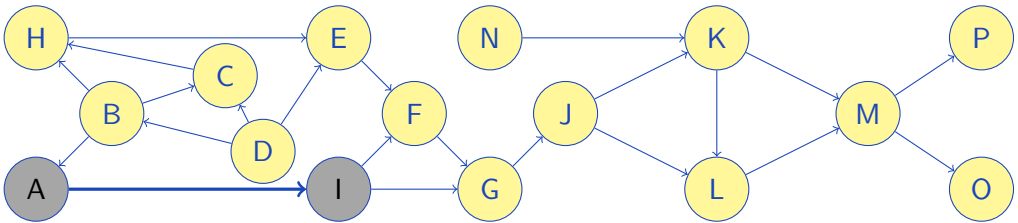
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



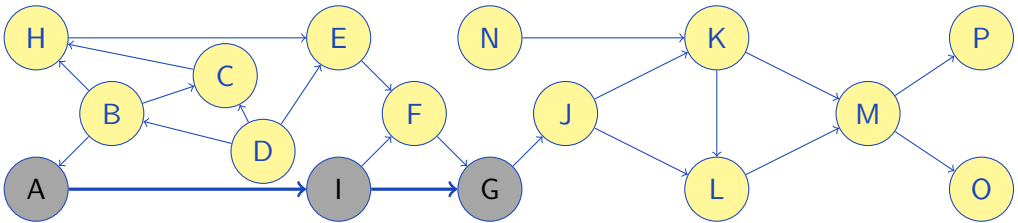
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



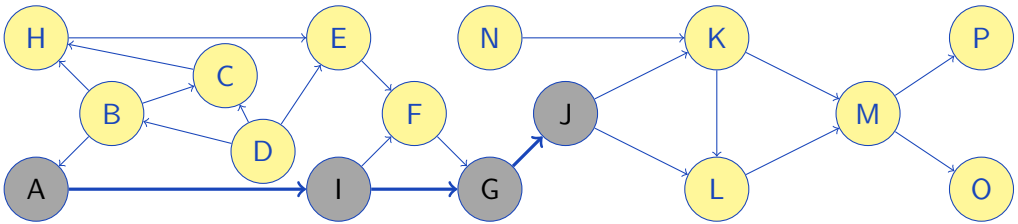
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



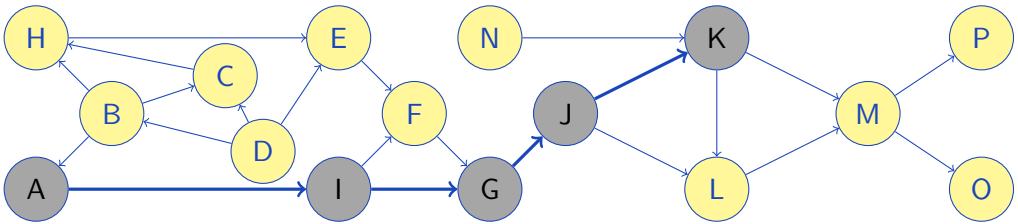
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



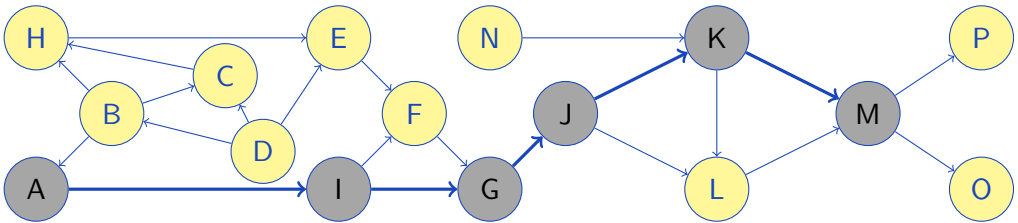
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



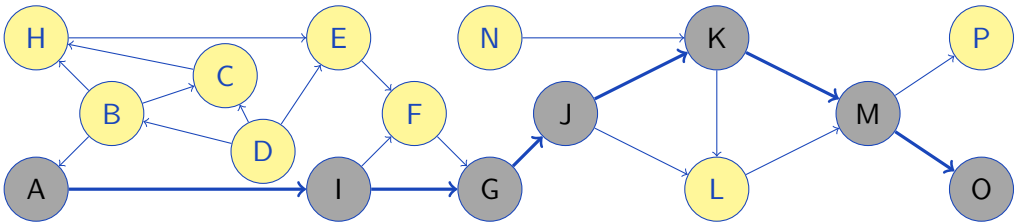
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



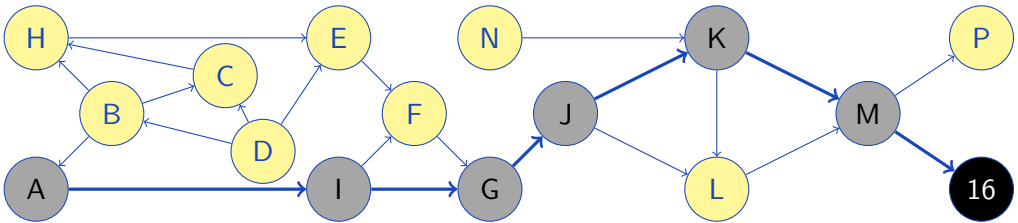
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



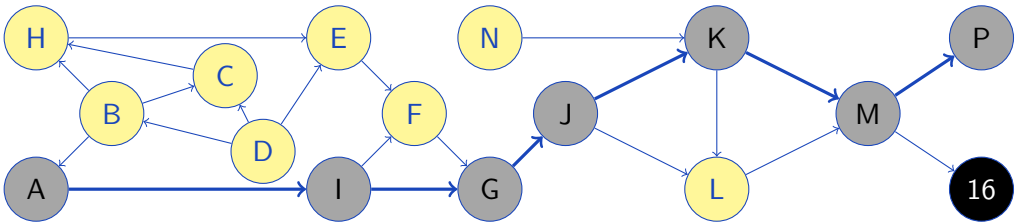
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



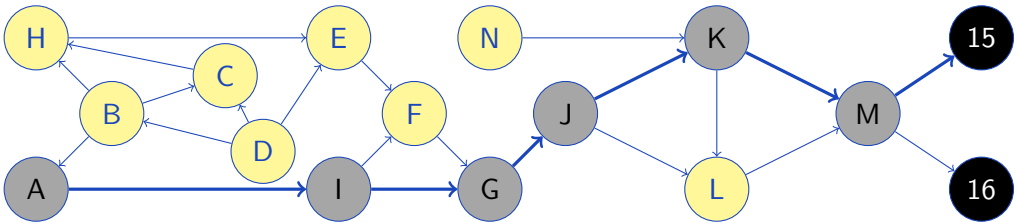
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



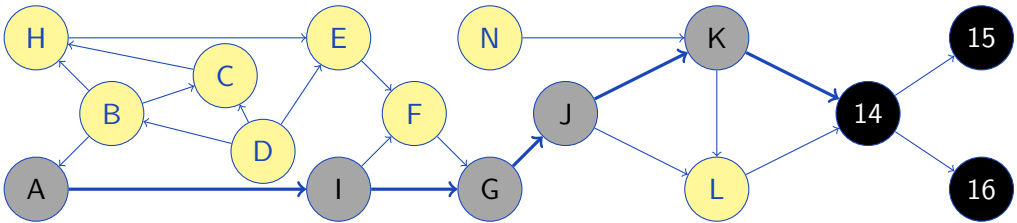
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



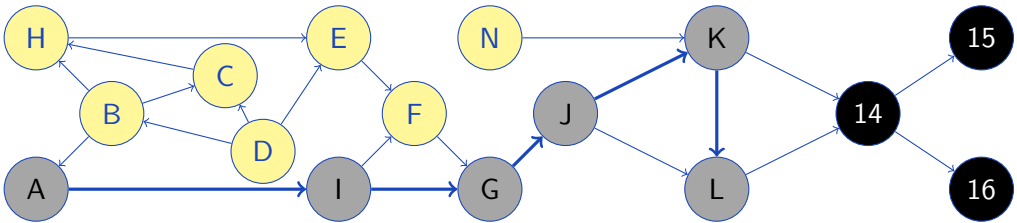
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



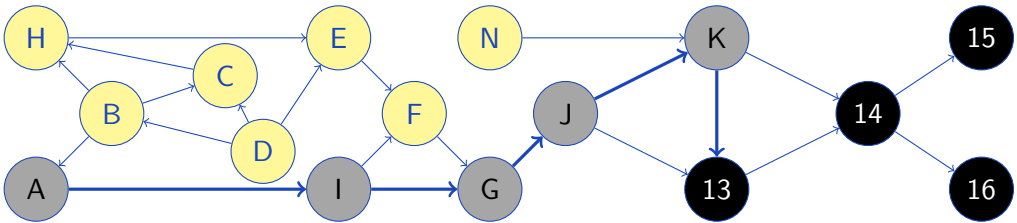
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



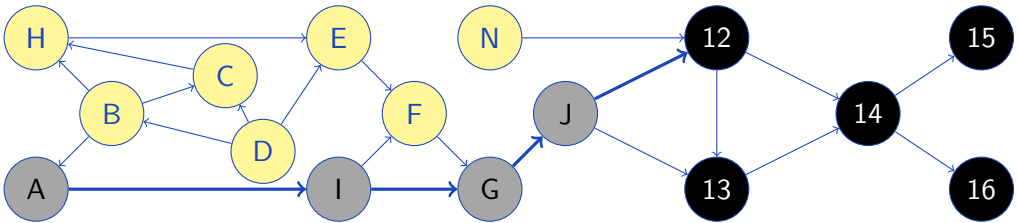
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



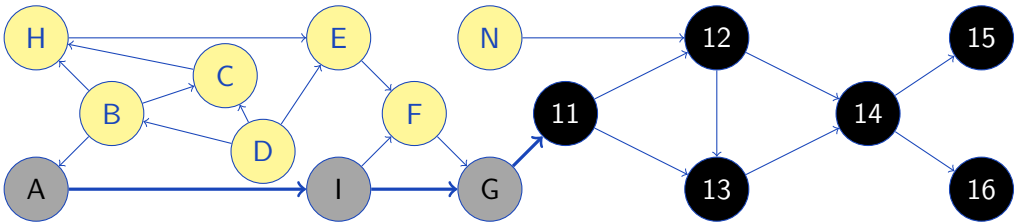
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



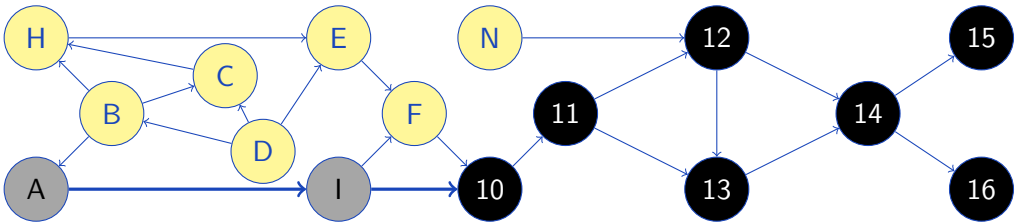
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



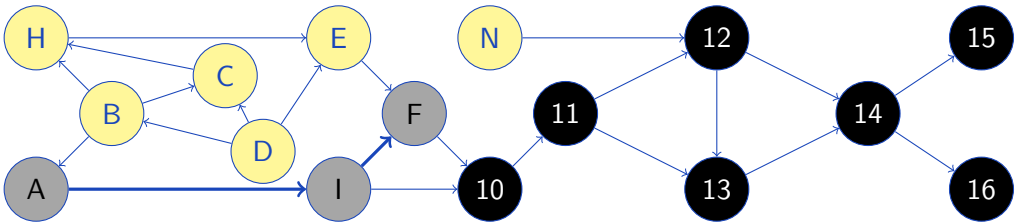
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



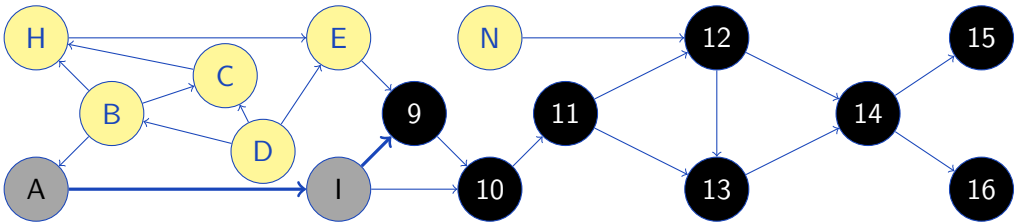
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



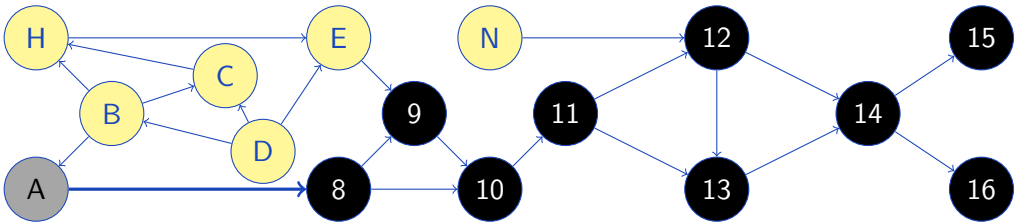
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



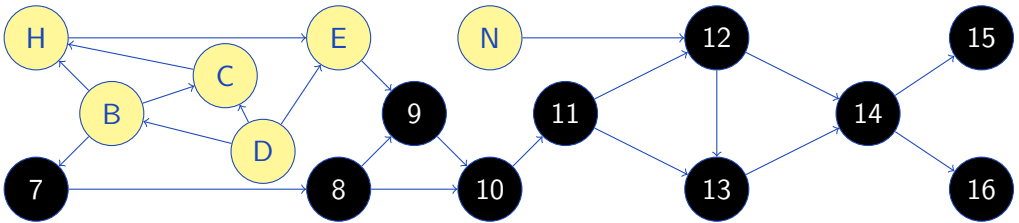
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



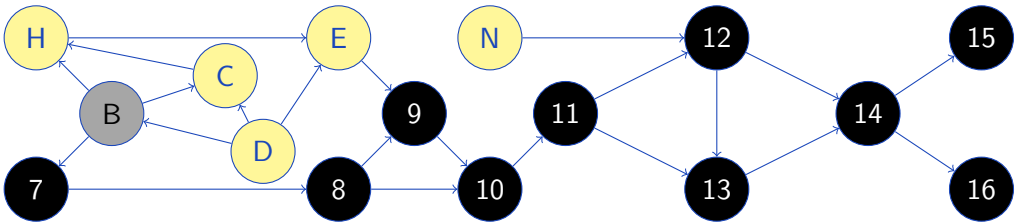
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



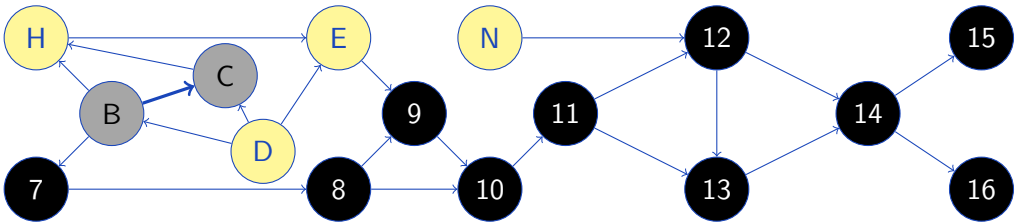
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



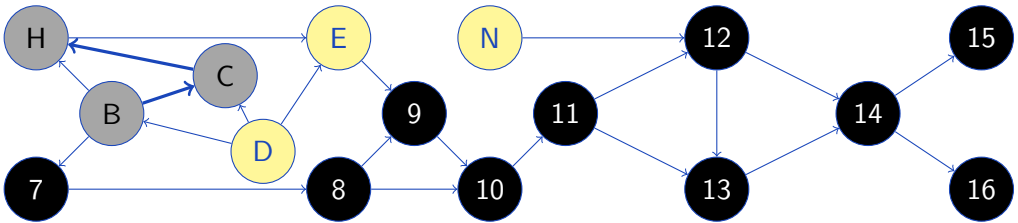
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



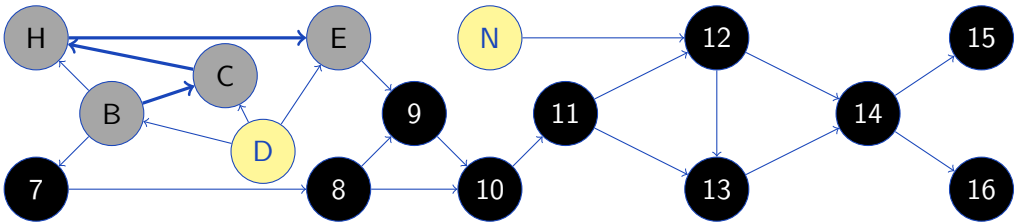
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



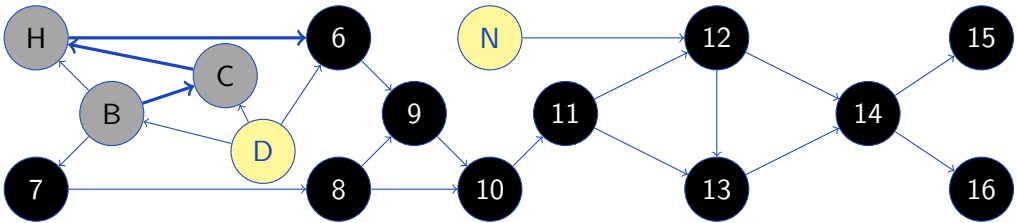
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



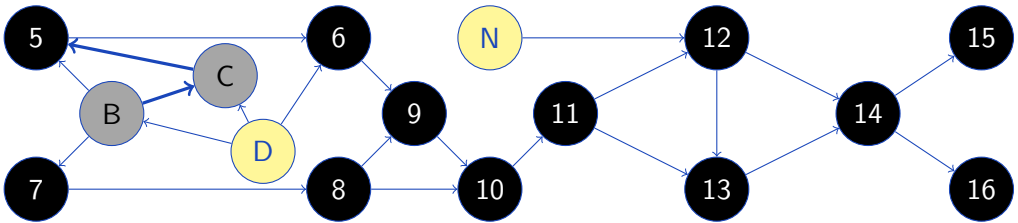
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



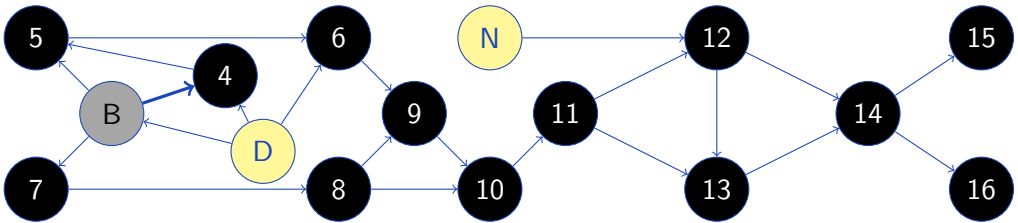
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



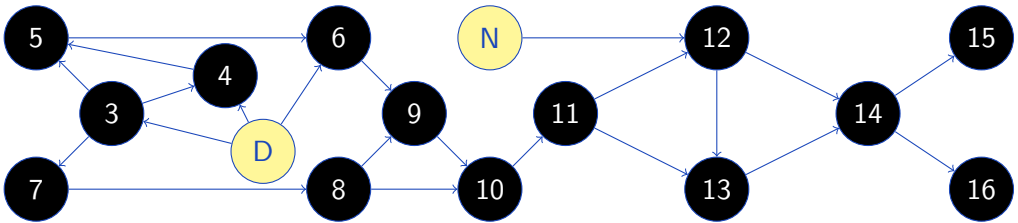
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



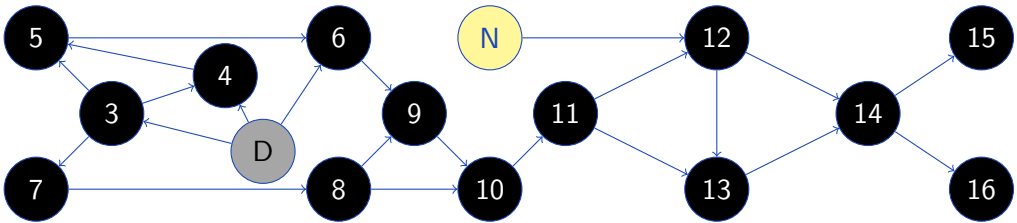
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



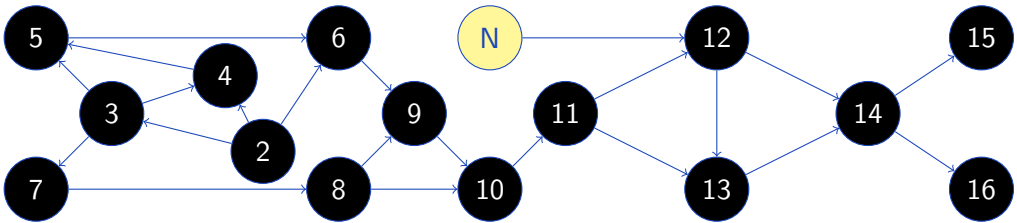
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



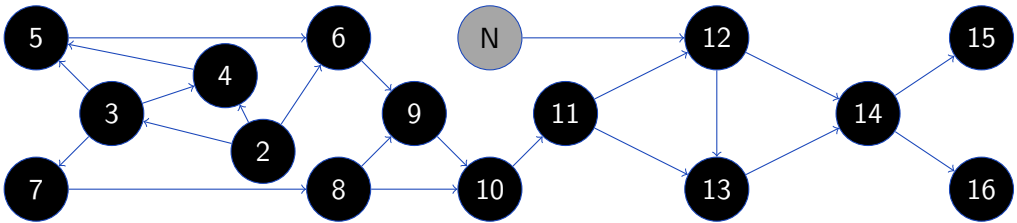
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



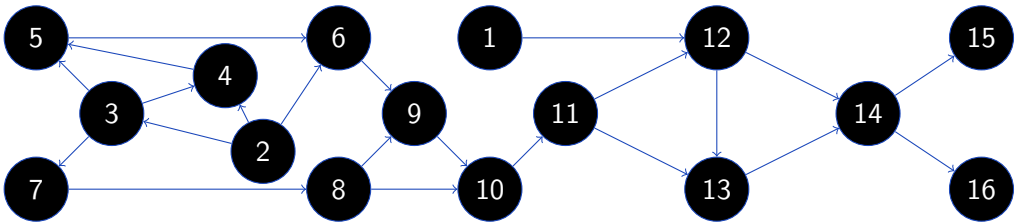
procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



procedure TopoSort(V, E)
  $U \leftarrow \emptyset$, $I \leftarrow \emptyset$, $t \leftarrow |V|$, $o \leftarrow \{\}$
  $A(v) = \{u \mid (v, u) \in E\}$
  for $v \in V$ **do**
    if $v \notin U$ **then** DFS(v, A) **end if**
  end for
end procedure

procedure DFS(v, A)
  $U \leftarrow U \cup v$
  for $u \in A(v)$ **do**
    if $u \notin U$ **then** DFS(u, A) **end if**
  end for
  $I(v) \leftarrow t$, $o[t] \leftarrow v$, $t \leftarrow t - 1$
end procedure



What will happen if topological sort is called on a graph with cycles?

What will happen if topological sort is called on a graph with cycles?

- DFS will eventually find a cycle

What will happen if topological sort is called on a graph with cycles?

- DFS will eventually find a cycle
- It may return an error which indicates that there is no topological sort

What will happen if topological sort is called on a graph with cycles?

- DFS will eventually find a cycle
- It may return an error which indicates that there is no topological sort
- But what if we ignore it?

What will happen if topological sort is called on a graph with cycles?

- DFS will eventually find a cycle
- It may return an error which indicates that there is no topological sort
- But what if we ignore it?
  - Every two vertices u and v, where u is reachable from v, but not vice versa, will have $l(u) > l(v)$

What will happen if topological sort is called on a graph with cycles?

- DFS will eventually find a cycle
- It may return an error which indicates that there is no topological sort
- But what if we ignore it?
  - Every two vertices u and v, where u is reachable from v, but not vice versa, will have $l(u) > l(v)$
- Condensation of a graph G: a graph whose vertices are strong connectivity components of G and whose edges are the edges between vertices from different components

What will happen if topological sort is called on a graph with cycles?

- DFS will eventually find a cycle
- It may return an error which indicates that there is no topological sort
- But what if we ignore it?
  - Every two vertices u and v, where u is reachable from v, but not vice versa, will have $l(u) > l(v)$
- Condensation of a graph G: a graph whose vertices are strong connectivity components of G and whose edges are the edges between vertices from different components
  - Condensation is a tree

What will happen if topological sort is called on a graph with cycles?

- DFS will eventually find a cycle
- It may return an error which indicates that there is no topological sort
- But what if we ignore it?
  - Every two vertices u and v, where u is reachable from v, but not vice versa, will have $l(u) > l(v)$
- Condensation of a graph G: a graph whose vertices are strong connectivity components of G and whose edges are the edges between vertices from different components
  - Condensation is a tree
  - ... and this tree will be traversed in a correct order by topological sort!

What will happen if topological sort is called on a graph with cycles?

- DFS will eventually find a cycle
- It may return an error which indicates that there is no topological sort
- But what if we ignore it?
  - Every two vertices u and v, where u is reachable from v, but not vice versa, will have $l(u) > l(v)$
- Condensation of a graph G: a graph whose vertices are strong connectivity components of G and whose edges are the edges between vertices from different components
  - Condensation is a tree
  - ... and this tree will be traversed in a correct order by topological sort!
  - So we may use topological sort to find the strong connectivity components

```
procedure CONDENSATION(V, E)
    TOPOSORT(V, E)
    A^R(v) = {u | (u, v) ∈ E}
    C ← {−1}
    K ← 1
    for i from 1 to |V| do
        if C[o[i]] = −1 then
            DFS(o[i], A^R, K)
            K ← K + 1
        end if
    end for
end procedure
```

```
procedure DFS(v, A, K)
    C[v] ← K
    for u ∈ A(v) do
        if C[u] = −1 then
            DFS(u, A, K)
        end if
    end for
end procedure
```

```
procedure CONDENSATION(V, E)
    TOPOSORT(V, E)
    A^R(v) = {u | (u, v) ∈ E}
    C ← {−1}
    K ← 1
    for i from 1 to |V| do
        if C[o[i]] = −1 then
            DFS(o[i], A^R, K)
            K ← K + 1
        end if
    end for
end procedure
```

```
procedure DFS(v, A, K)
    C[v] ← K
    for u ∈ A(v) do
        if C[u] = −1 then
            DFS(u, A, K)
        end if
    end for
end procedure
```

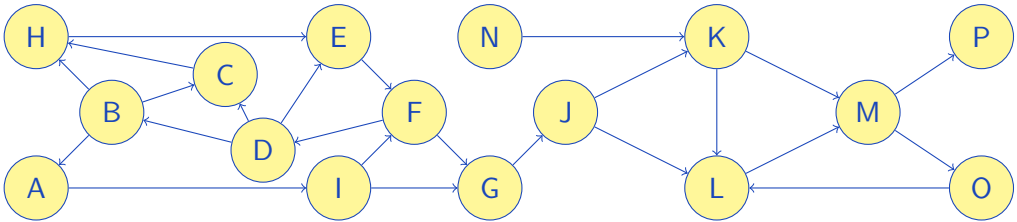
- Second DFS traverses vertices in the order of topological sort

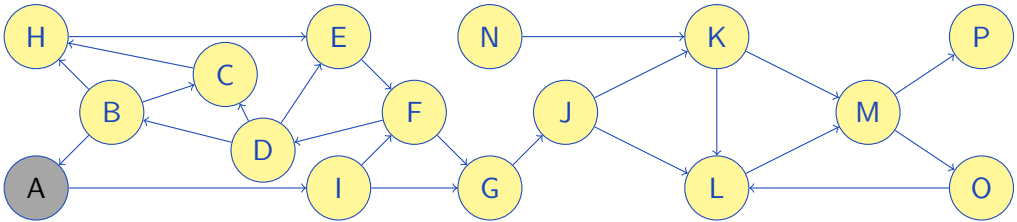
```
procedure CONDENSATION(V, E)
    TOPOSORT(V, E)
    A^R(v) = {u | (u, v) ∈ E}
    C ← {−1}
    K ← 1
    for i from 1 to |V| do
        if C[o[i]] = −1 then
            DFS(o[i], A^R, K)
            K ← K + 1
        end if
    end for
end procedure
```

```
procedure DFS(v, A, K)
    C[v] ← K
    for u ∈ A(v) do
        if C[u] = −1 then
            DFS(u, A, K)
        end if
    end for
end procedure
```

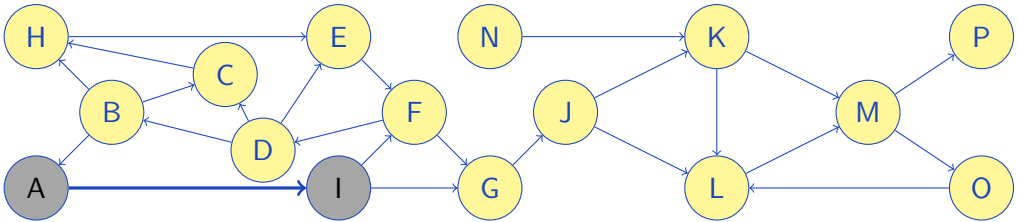
- Second DFS traverses vertices in the order of topological sort
- ... and does it using reversed edges

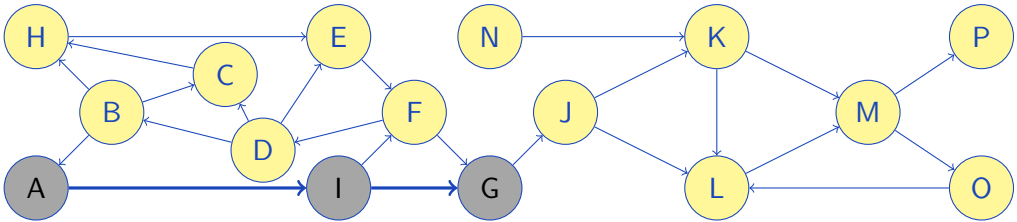
Example of condensation



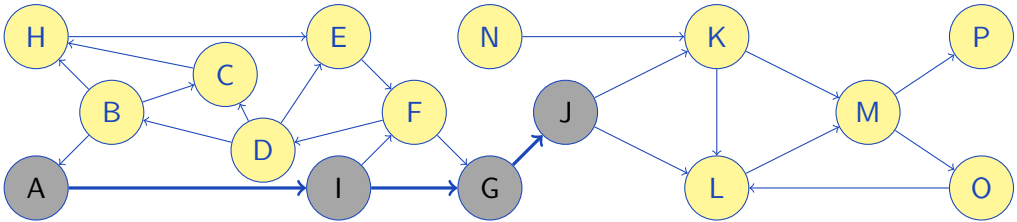

H
E
N
K
P
C
B
F
J
M
D
A
I
G
L
O

Example of condensation

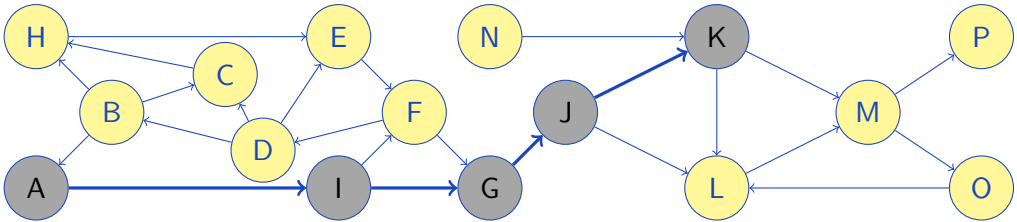


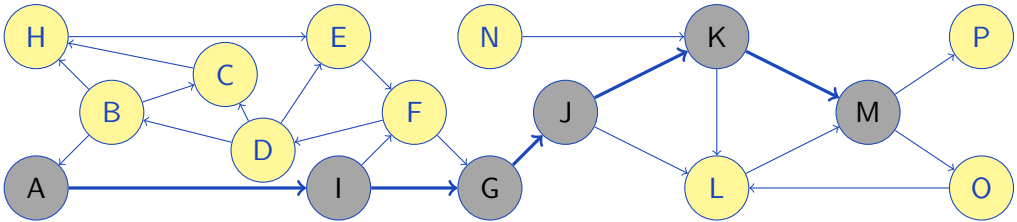

H
E
N
K
P
C
B
F
J
M
D
A
I
G
L
O

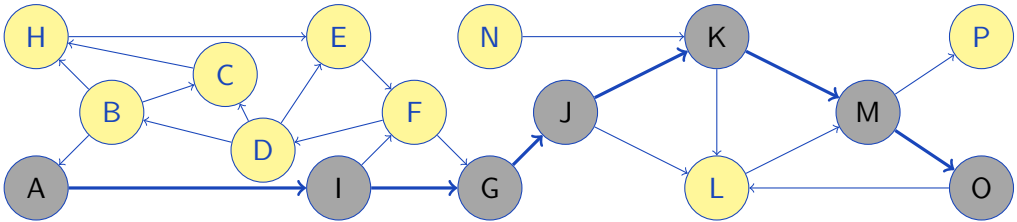
Example of condensation

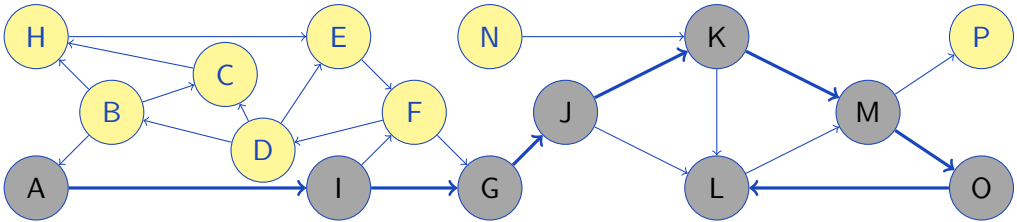


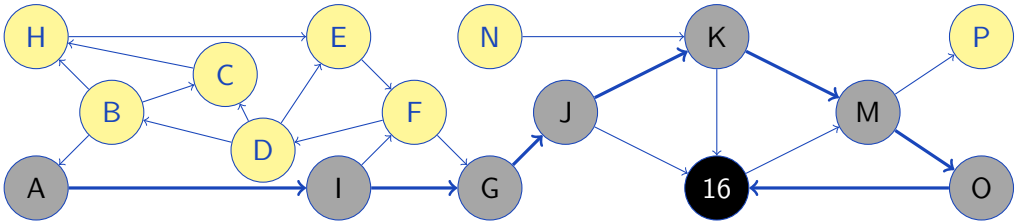
Example of condensation

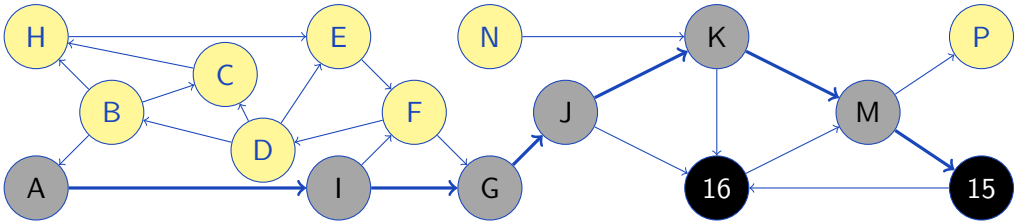


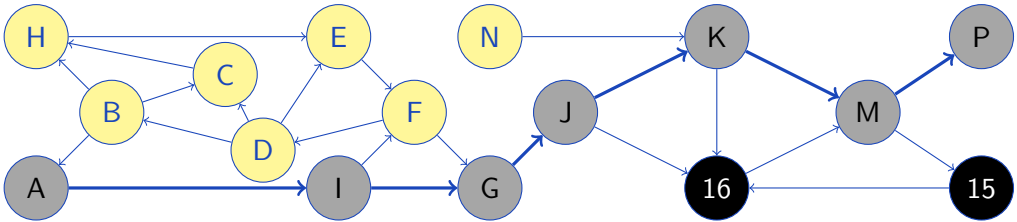

H
E
C
B
F
D
A
I
G
N
K
P
J
M
L
O

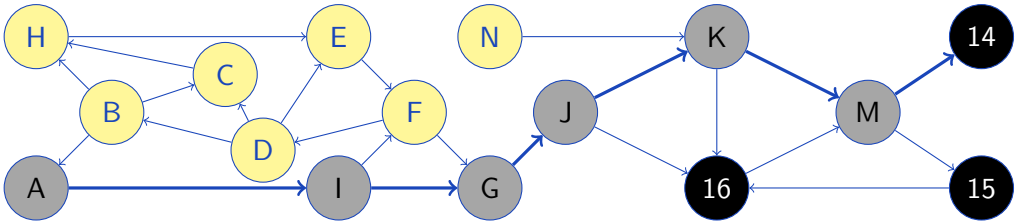

H
E
C
B
F
D
A
I
G
N
K
P
J
M
L
O

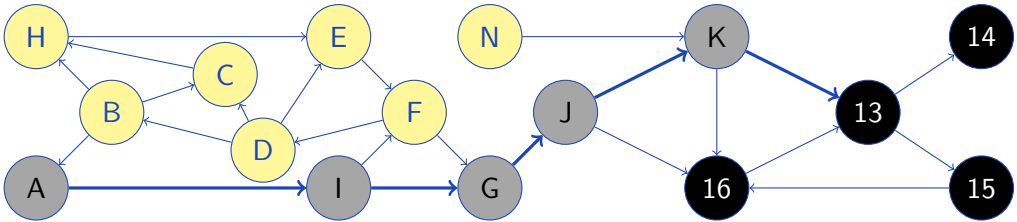

H
E
C
B
F
D
A
I
G
N
K
P
J
M
L
O

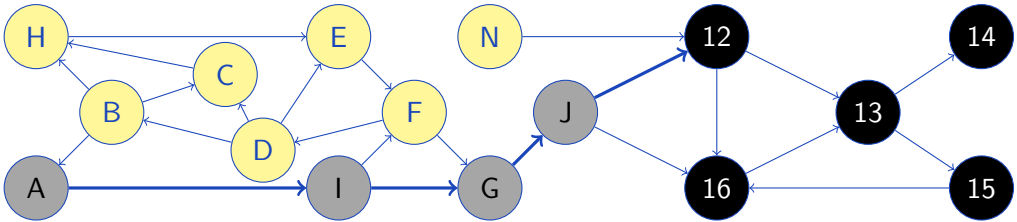

H
E
N
K
P
C
B
F
J
M
D
A
I
G
16
O

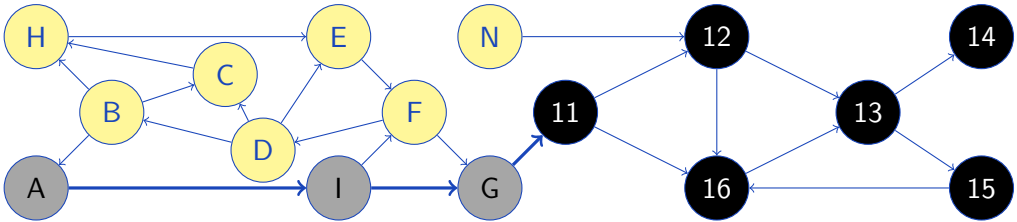

H
E
C
B
F
D
A
I
G
N
K
P
J
M
16
15

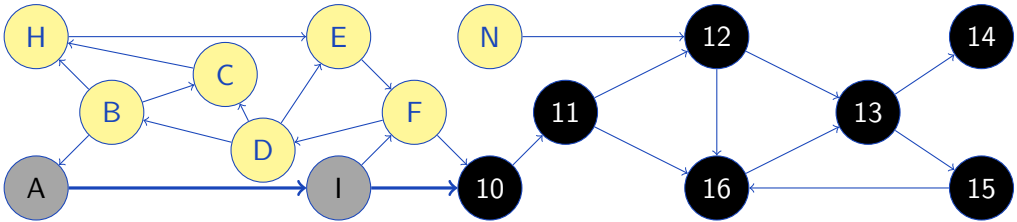

H
E
N
K
P
C
B
F
J
M
D
A
I
G
16
15

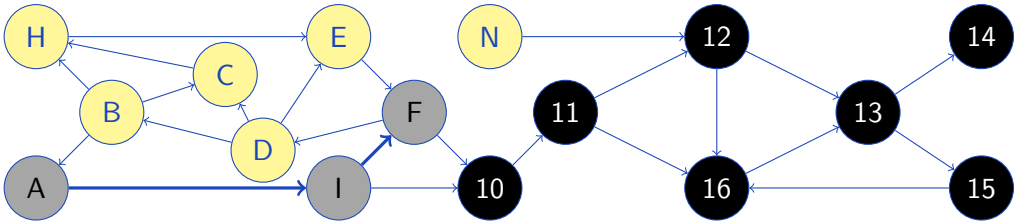

H
E
N
K
14
C
B
F
J
M
D
A
I
G
16
15

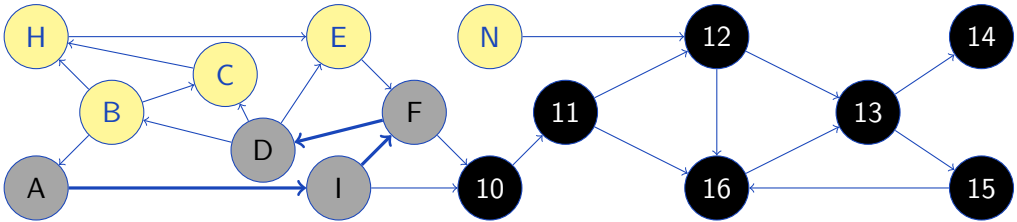

H
E
N
K
14
C
B
F
J
13
D
A
I
G
16
15

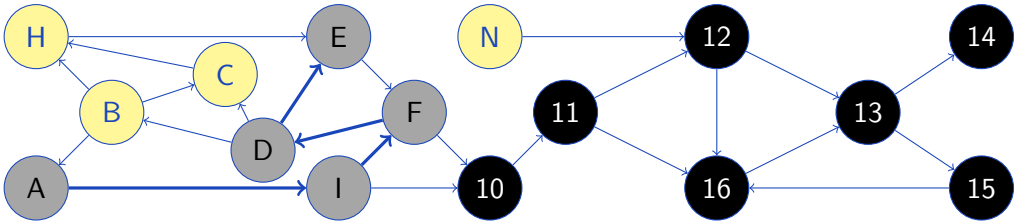

H
E
C
B
F
D
A
I
G
N
12
14
J
13
16
15

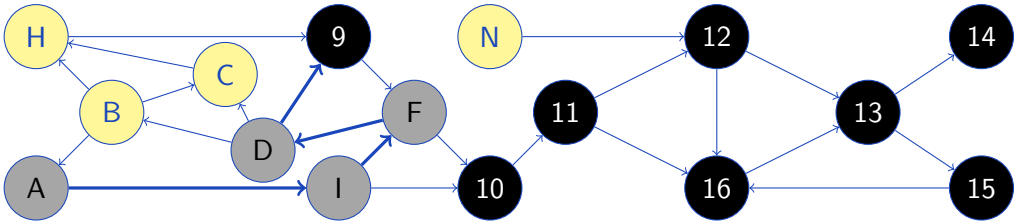

H
E
C
B
F
D
A
I
G
N
12
14
11
13
16
15

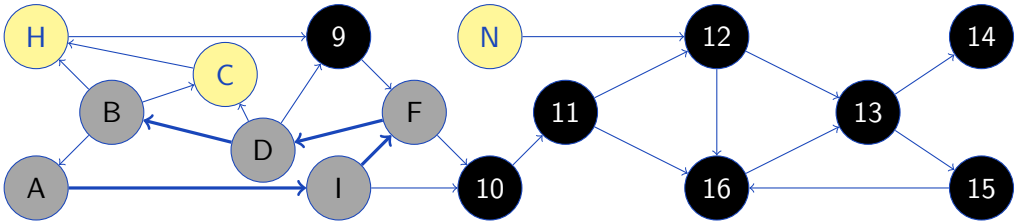

H
E
C
B
F
D
A
I
10
N
12
14
11
13
16
15

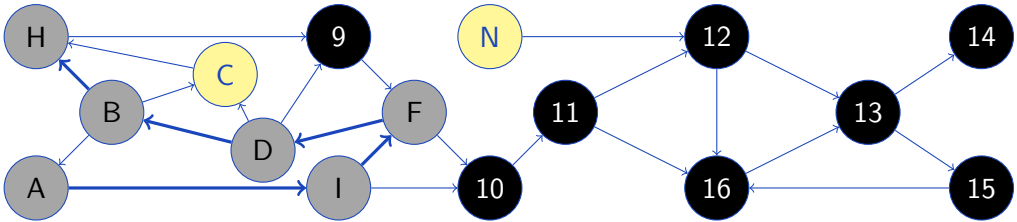

H
E
C
B
F
D
A
I
N
12
14
11
13
10
16
15

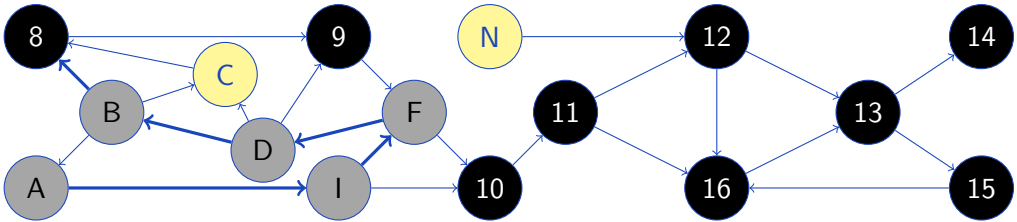

H
E
C
B
F
D
A
I
N
12
14
11
13
10
16
15

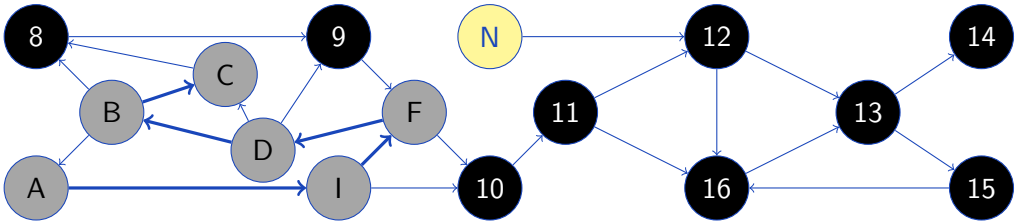

H
E
C
B
F
D
A
I
N
12
14
11
13
10
16
15

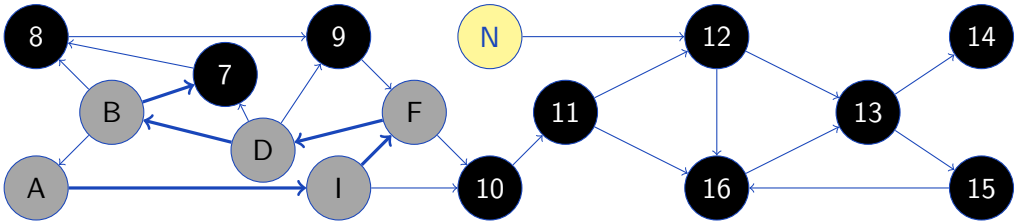

H
9
C
B
F
D
A
I
10
N
12
14
11
13
16
15

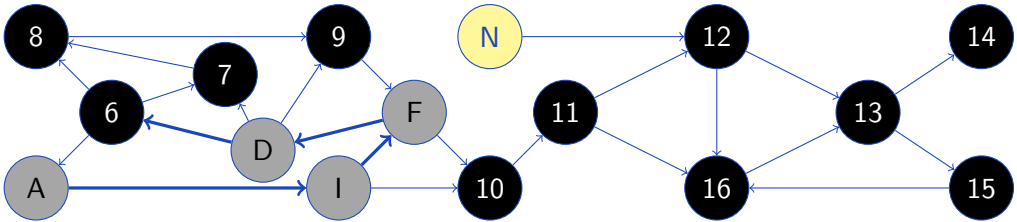

H
9
C
B
F
D
A
I
10
N
12
14
11
13
16
15


H
9
C
B
F
D
A
I
10
N
12
14
11
13
16
15

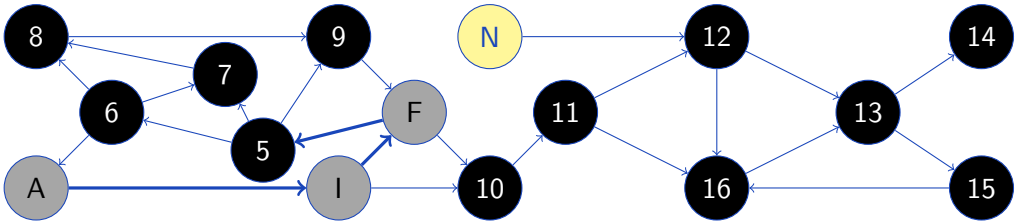

8
9
C
B
F
D
A
I
10
N
12
14
11
13
16
15

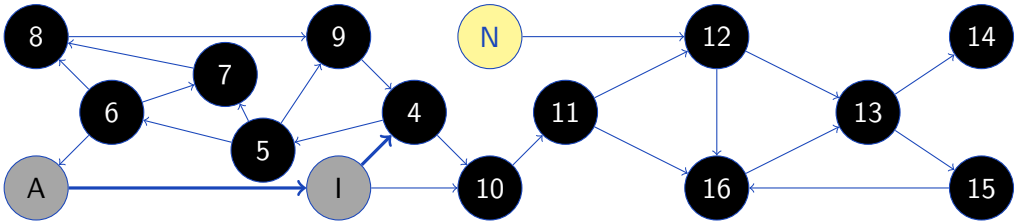

8
9
C
B
F
D
A
I
10
N
12
14
11
13
16
15


8
9
7
B
F
D
A
I
10
N
12
14
11
13
16
15

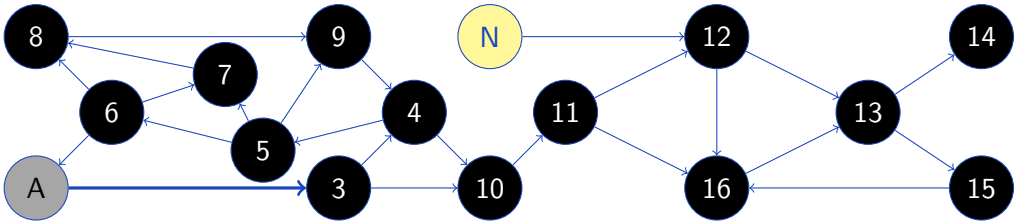

8
9
7
6
F
D
A
I
10
N
12
14
11
13
16
15

Example of condensation

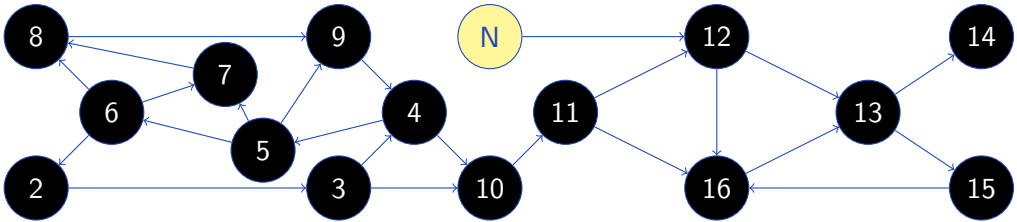



8
9
7
6
4
5
A
I
10
N
12
14
11
13
16
15

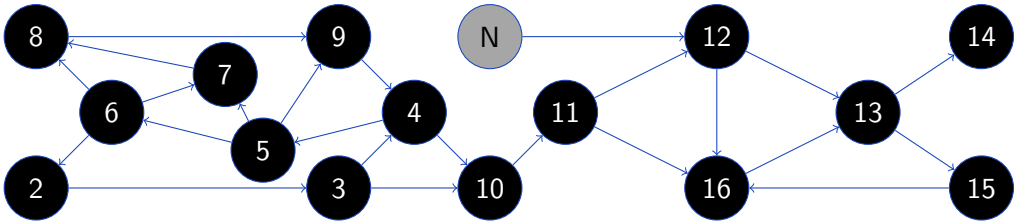
Example of condensation



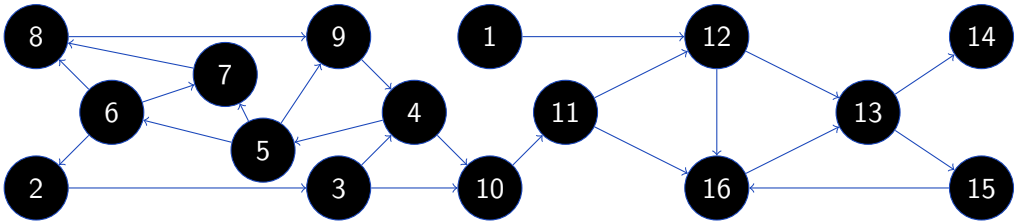
Example of condensation



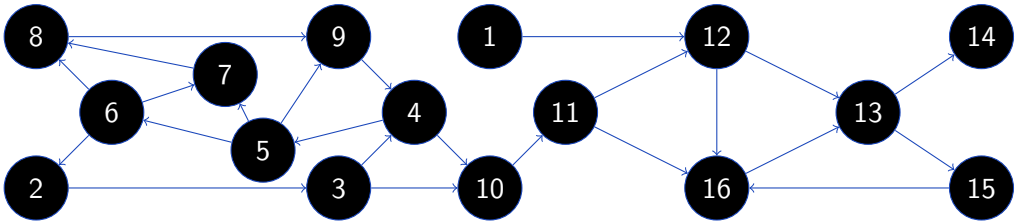
Example of condensation



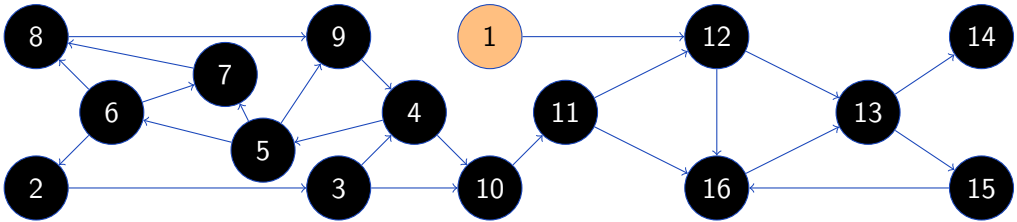
Example of condensation



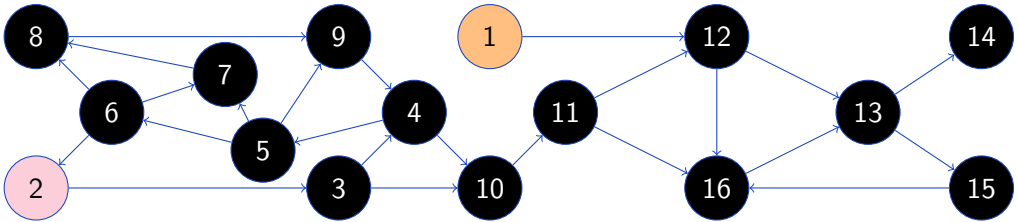
Example of condensation



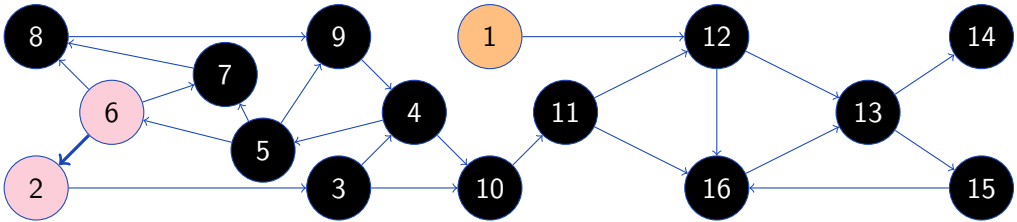
Example of condensation

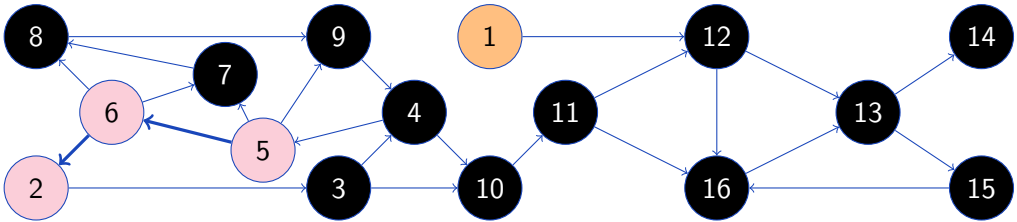


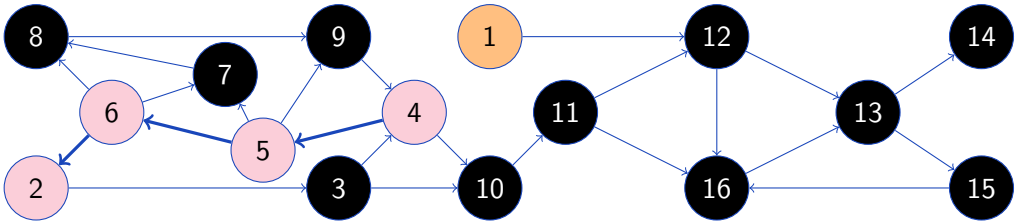
Example of condensation

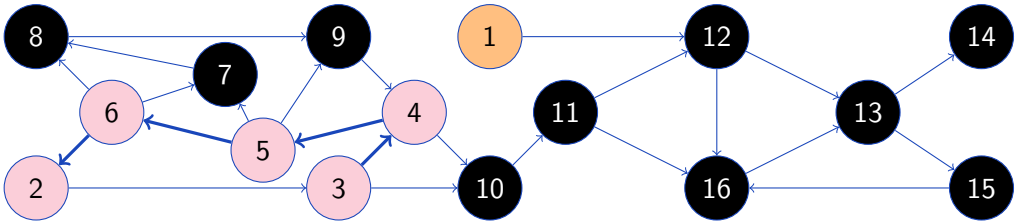


Example of condensation

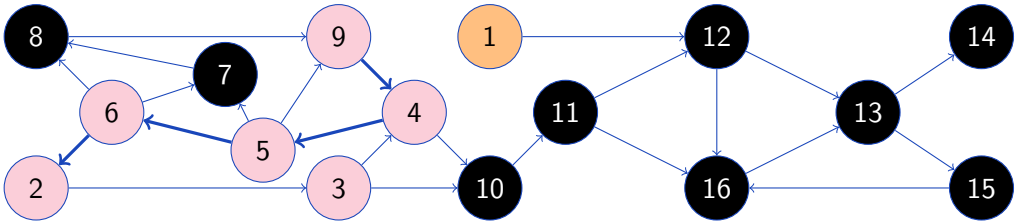


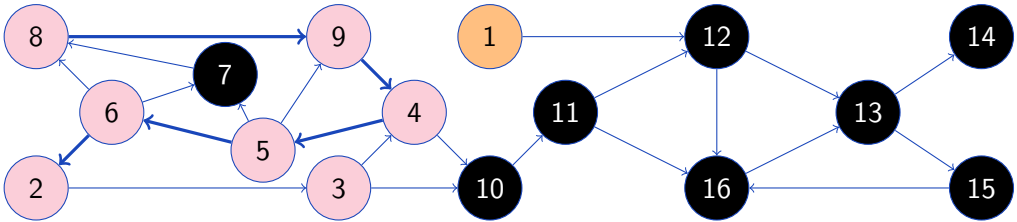

8
9
1
12
14
7
6
4
11
13
5
2
3
10
16
15


8
9
7
6
4
5
2
3
10
1
12
14
11
13
16
15

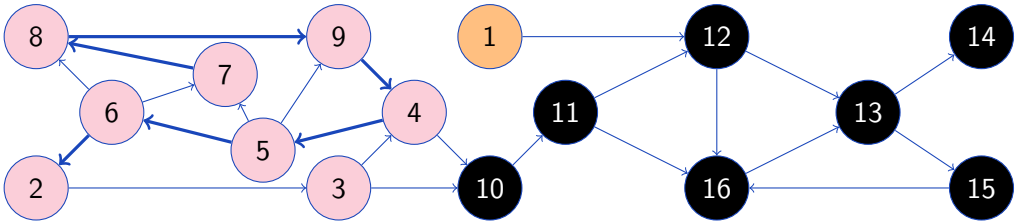

8
9
7
6
4
5
2
3
1
12
14
11
13
10
16
15

Example of condensation

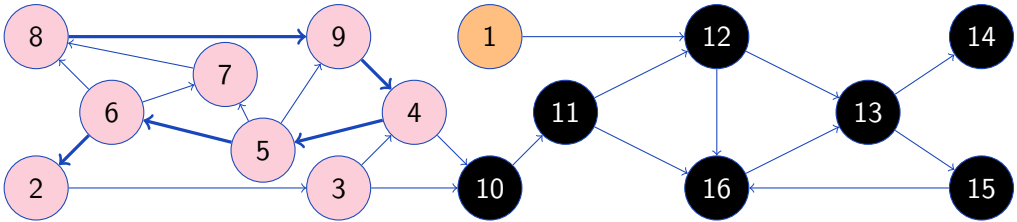



8
9
7
6
4
5
2
3
10
1
12
14
11
13
16
15

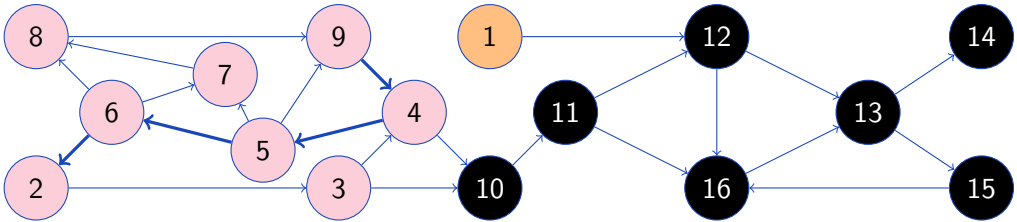
Example of condensation



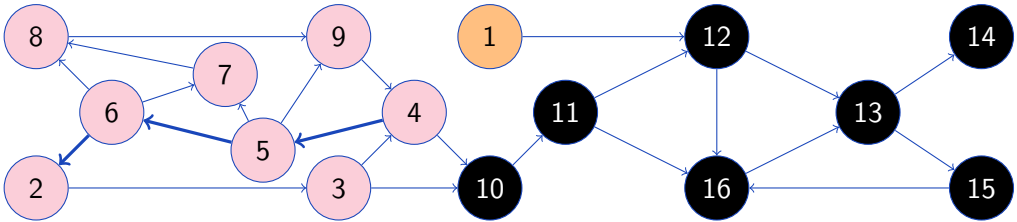
Example of condensation



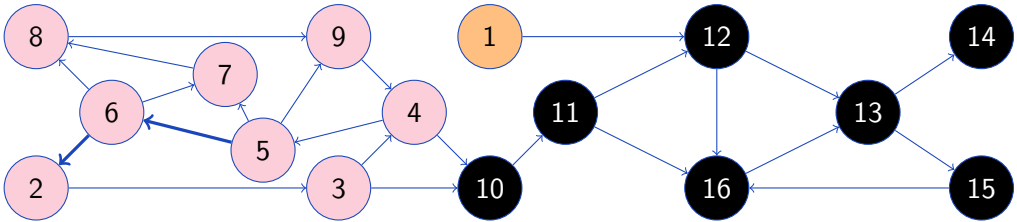
Example of condensation



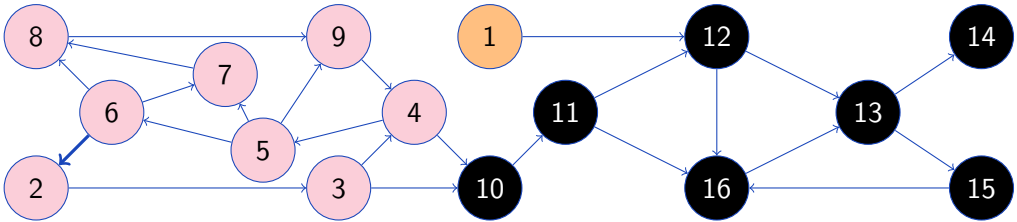
Example of condensation



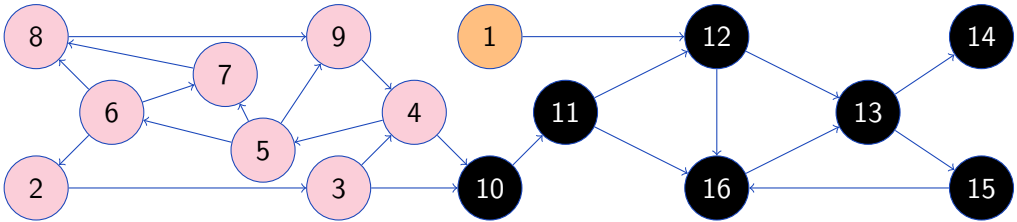
Example of condensation



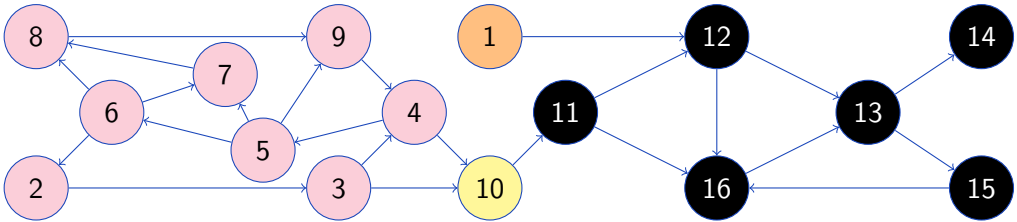
Example of condensation



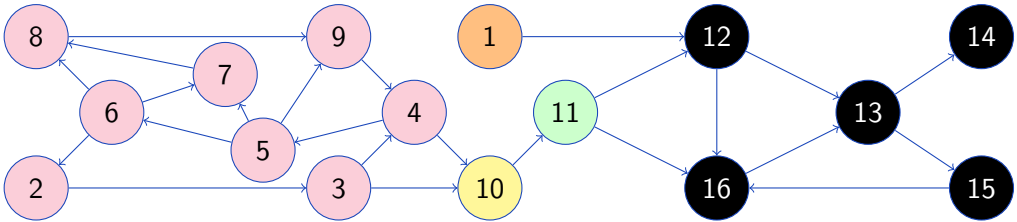
Example of condensation



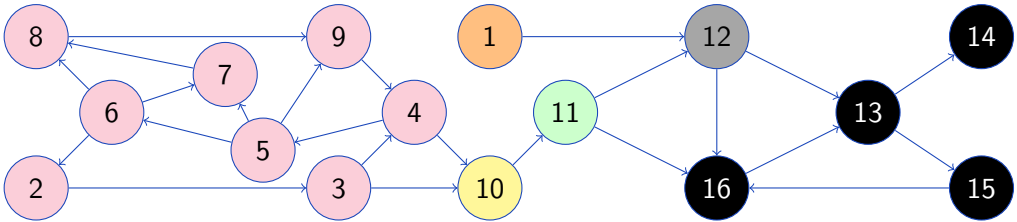
Example of condensation



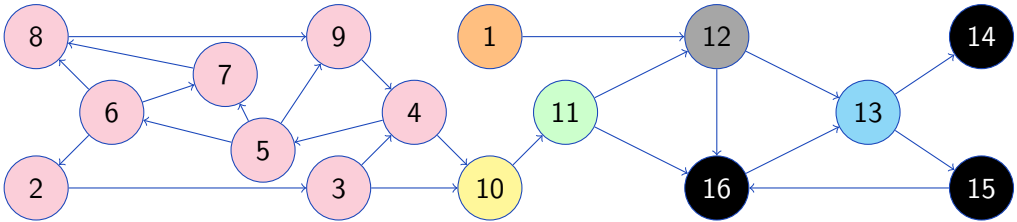
Example of condensation



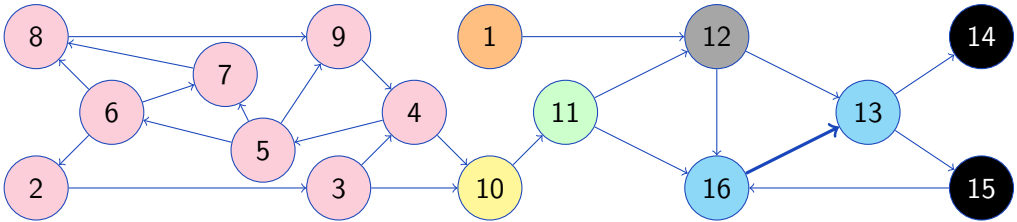
Example of condensation



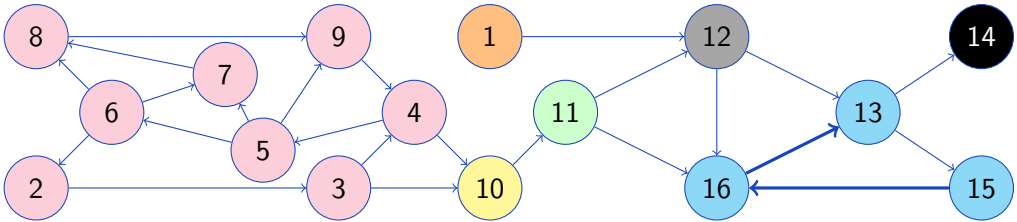
Example of condensation



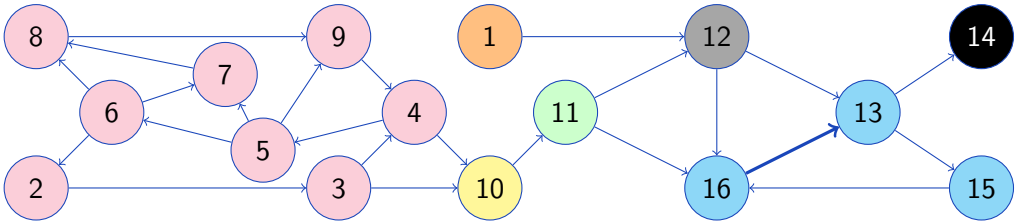
Example of condensation



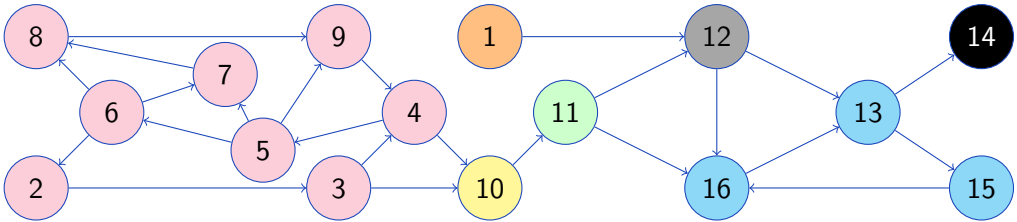
Example of condensation

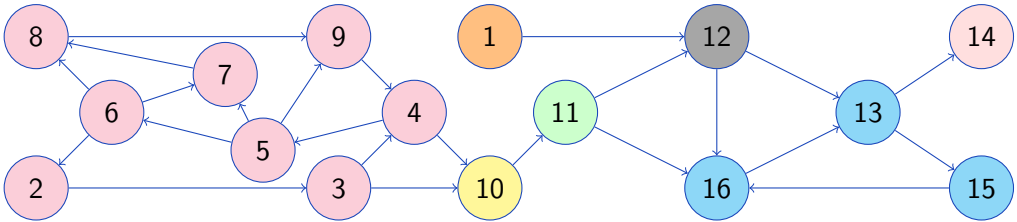


Example of condensation



Example of condensation




8
9
1
12
14
7
6
4
11
13
5
2
3
10
16
15