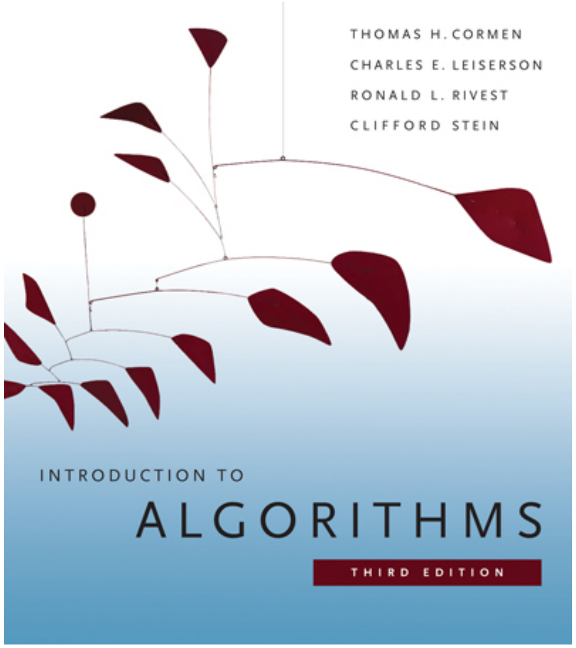
# 6.006- *Introduction to Algorithms*



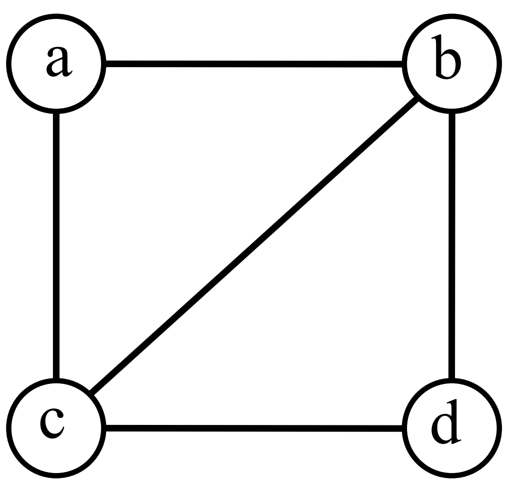

## *Lecture 14*

**Prof. Silvio Micali**

# Graphs

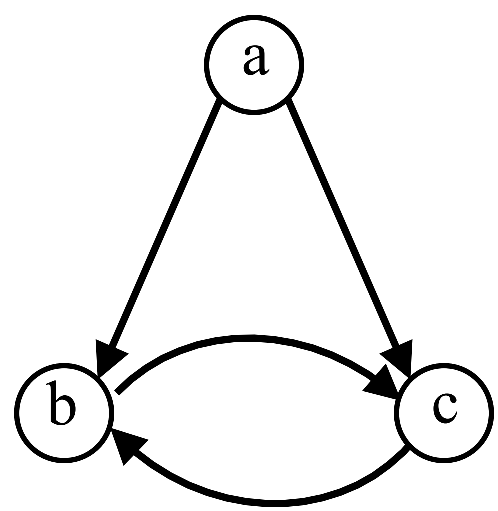
***Undirected***

- V={a,b,c,d}
- E={{a,b}, {a,c}, {b,c}, {b,d}, {c,d}}




***Directed***

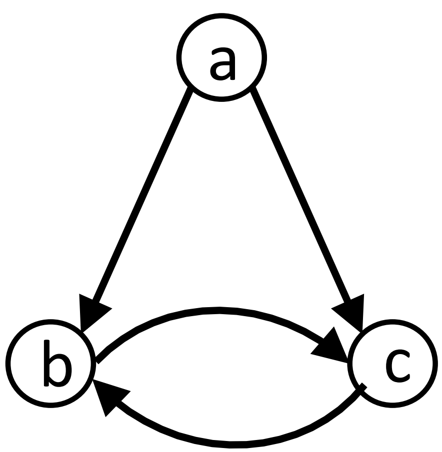
- V = {a,b,c}
- E = {(a,c), (a,b) (b,c), (c,b)}

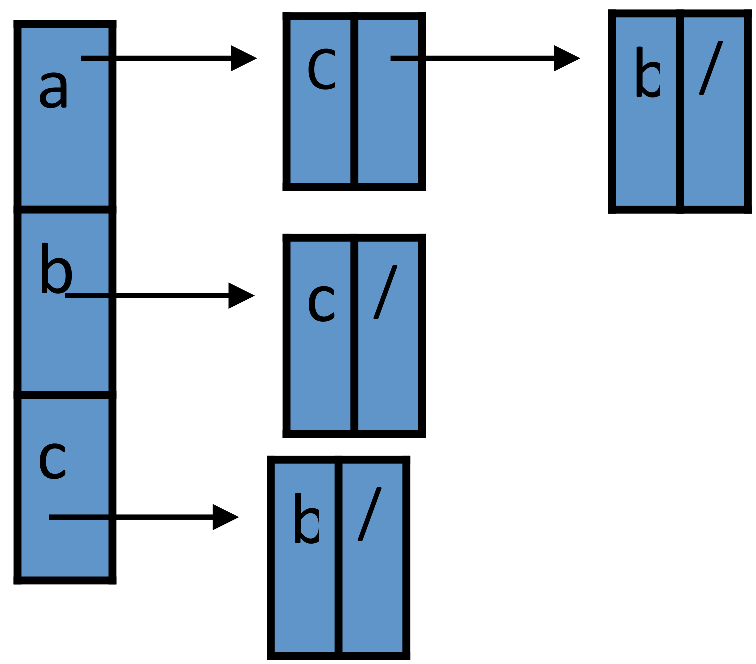


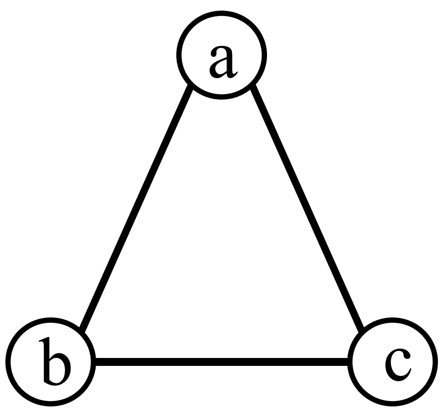
# Computer Representation

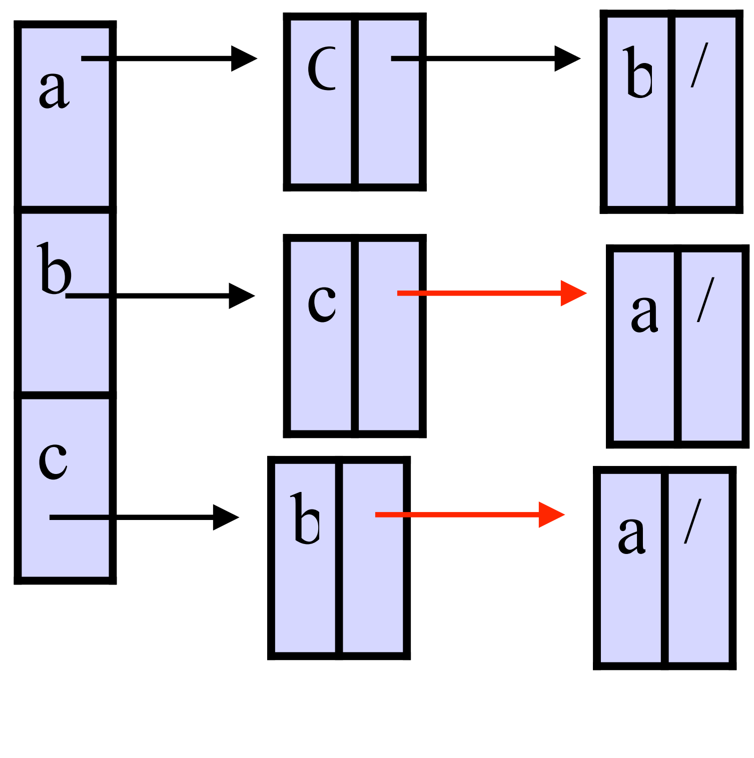
Four representations with pros/cons

***Adjacency lists*** (of neighbors of each vertex)




a
b
c


a
c
b
/
b
c
a
/
c
b
a
/

# Breadth First Search

- Start with vertex v
- List all its neighbors (distance 1)
- Then all their neighbors (distance 2)
- Etc.

# Augmented Breadth First Search =Shortest Path Alg

## $(Pseudo^2)$

Initially, $s$ is marked $0$, all other vertices are marked $\infty$

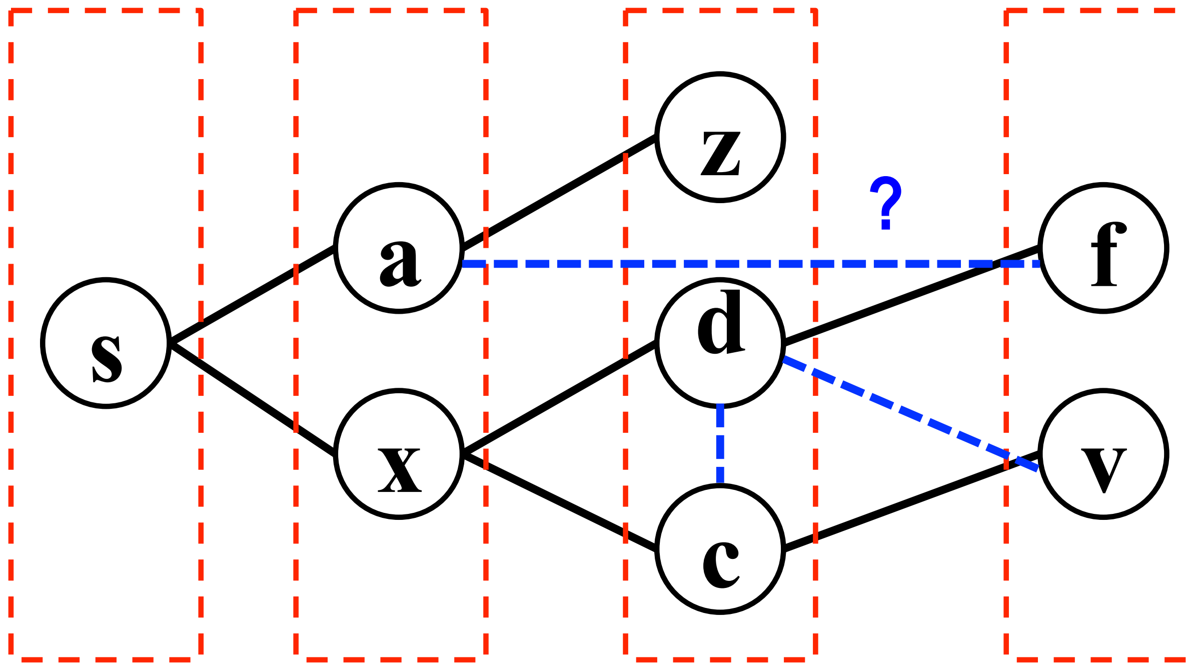
1. $i \leftarrow 0$
2. Find all neighbors of at least one vertex marked $i$. If none, STOP.
3. Mark all vertices found in (3) with $i + 1$.
4. $i \leftarrow i + 1$

**Thm:** Every vertex is marked with its distance form $s$

**Complexity:** $\boldsymbol{O(n+m)}$

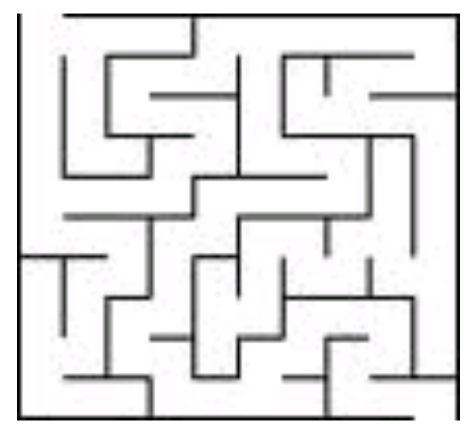
# BFS Tree Structure

- Spanning Tree with Lots of Structural Information



# Depth First Search

- Exploring a maze
- From current vertex, move to another
- Until you get stuck
- Then backtrack till you find the first new possibility for exploration



# DFS

0. Mark all edges "unused". For all $v \in V, \#(v) := 0.$ Let $i := 0$ and $CoA := s.$

1. $i \leftarrow i + 1$ $\quad \#(CoA) \leftarrow i$

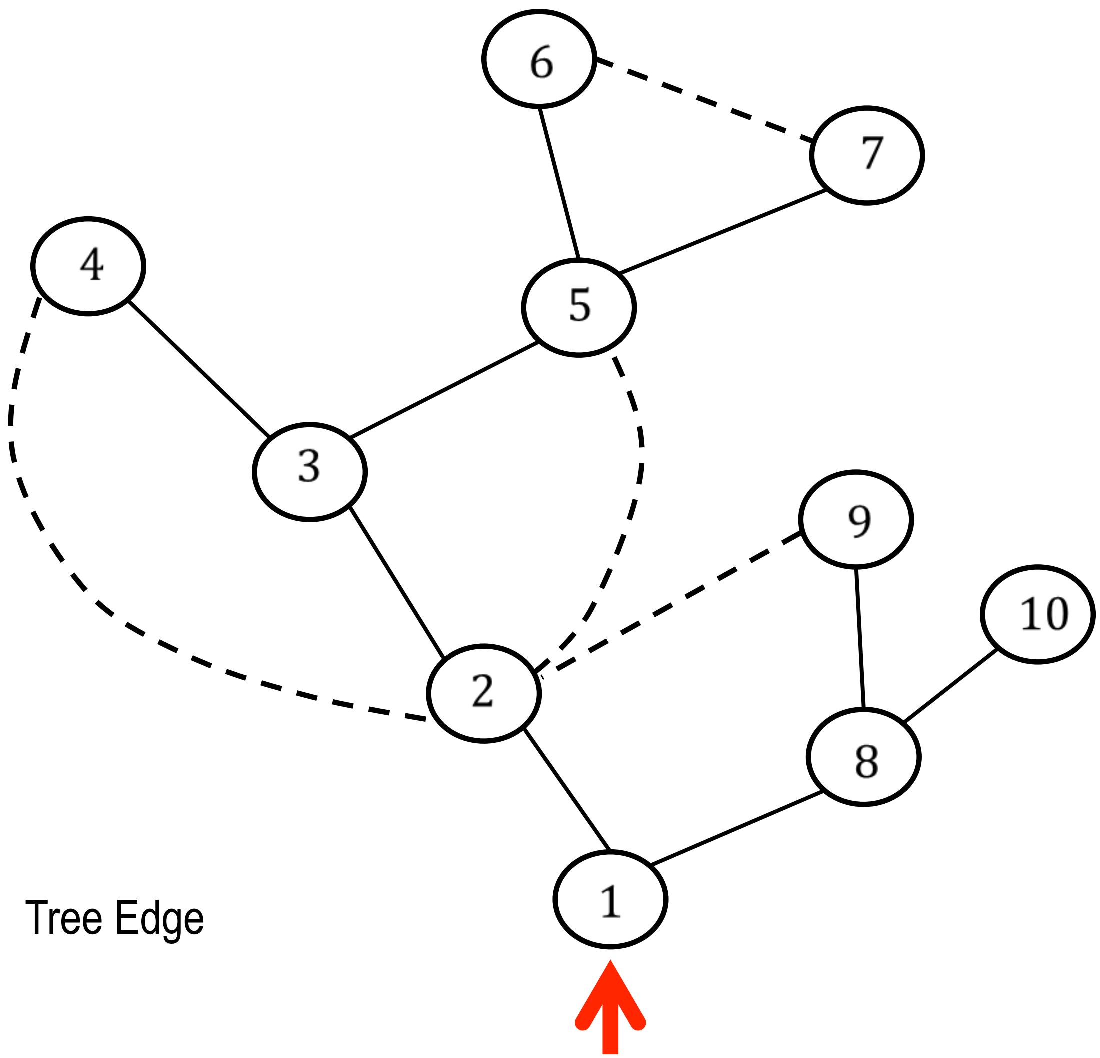
2. If $CoA$ has no unused edges, go to (4)

3. Choose an unused edge $CoA \overset{e}{\leftrightarrow} u.$ Mark $e$ used. If $\#(u) \neq 0$ go to (2). Else $F(u) \leftarrow CoA$ $\quad CoA \leftarrow u$ and go to (1)

4. If $\#(CoA) = 1$ HALT

5. $CoA \leftarrow F(CoA)$ and go to (2)

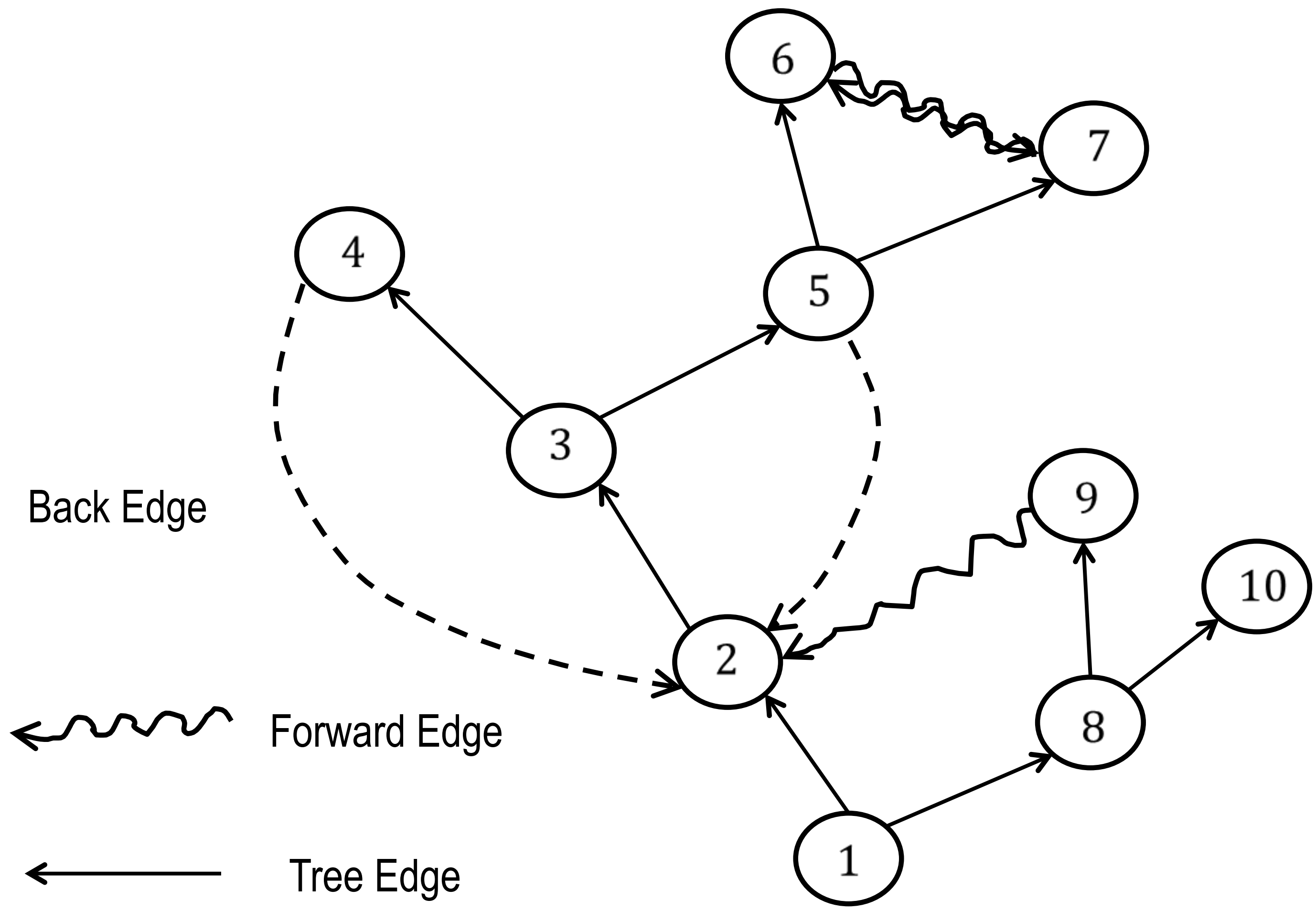
# DFS Tree




Back Edge

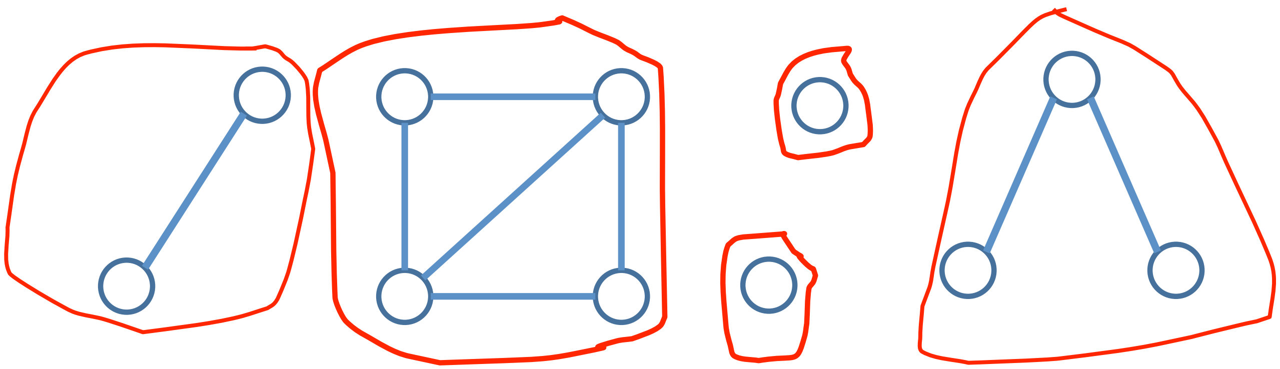
Tree Edge

# DFS Tree
# Directed Case



# Connected Components

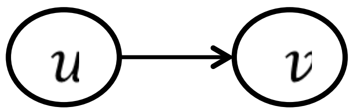
An equivalence relation



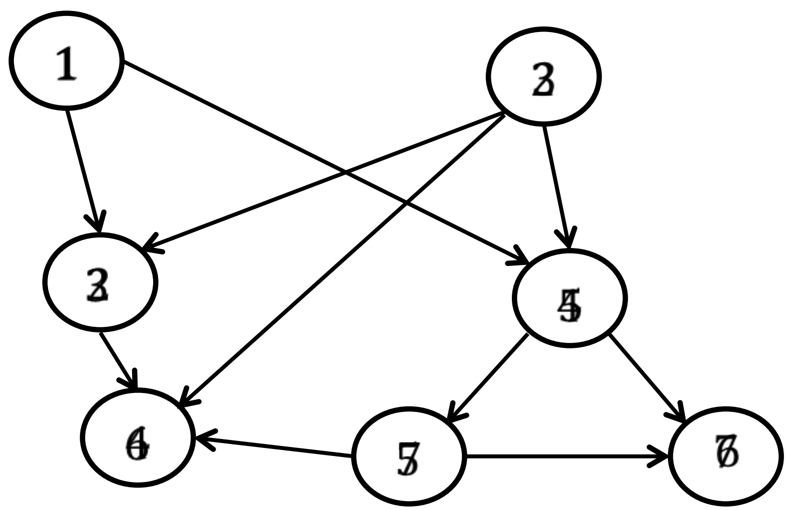
Linear with Good Counting!

# Topological Sort

***TS***: numbering of the vertices of a directed acyclic graph (DAG) such that

if 

then $TS(u) < TS(v)$



# Idea:

# **Topological Reverse**

***TR***: numbering of the vertices of a directed acyclic graph (DAG) such that

if $u \rightarrow v$ then $TR(u) < TR(v)$

$$TS(x) = n - TR(x)$$

# More General Shortest Paths for a given node $s$

Undirected (Directed in recitation!) graphs with non-negative edge length

$$G = (V, E) \qquad \ell : E \to [, +\infty)$$

Picture

# Dijstra's Algorithm

$\lambda$: label. If $\lambda(v) = x$, then there is a path from $s$ to $v$ of length $x$, not necessarily minimum

$T$: Set of temporarily labeled vertices

$P$: Set of permanently labeled vertices

0. $\lambda(s) \leftarrow 0 \qquad T \leftarrow \{s\} \qquad P \leftarrow \emptyset$

$1.\ \ While\ T \neq \emptyset$ do:

- Choose $v \in T$ with minimum label
- $T \leftarrow T \setminus \{v\} \qquad P \leftarrow P \cup \{v\}$
- $\forall\ v \xrightarrow{e} u\ \ do$
  - $if\ u \in T, then\ \lambda(u) \leftarrow \min\{\lambda(u), \lambda(v) + \ell(e)\}$
  - $Else, if\ u \notin P\ then \quad \lambda(u) \leftarrow \lambda(v) + \ell(e) \quad \& \quad T \leftarrow T \cup \{u\}$

# Analysis

# Cycles?