IIT ROORKEE

FREE ONLINE EDUCATION swayam

NPTEL

NPTEL ONLINE CERTIFICATION COURSE

VLSI Physical Design with Timing Analysis

Lecture – 6: Spanning Tree and Shortest Path Algorithms

Bishnu Prasad Das

Department of Electronics and Communication Engineering

Spanning Tree Algorithms

- Electronic circuit designs require connecting multiple component pins.
- The goal is to minimize wire usage while ensuring all pins are connected.
  - This is modeled as an undirected graph with pins as vertices and wire costs as edge weights.

Spanning Tree Algorithms

- **Graph Representation:**
  - Define graph G = (V, E).
  - V represents pins, and E represents potential connections.
  - Each edge (u, v) has a weight w(u, v) denoting wire cost between pins u and v.

Spanning Tree Algorithms

- **Objective:**
  - Find an acyclic subset $T \subseteq E$ that connects all vertices.
  - Minimize the total weight (wire length) of this subset T.

Spanning Tree Algorithms

- **Spanning Trees:**
  - The solution forms a tree (acyclic) known as a "spanning tree."
  - This tree connects all vertices in the original graph.

Spanning Tree Algorithms

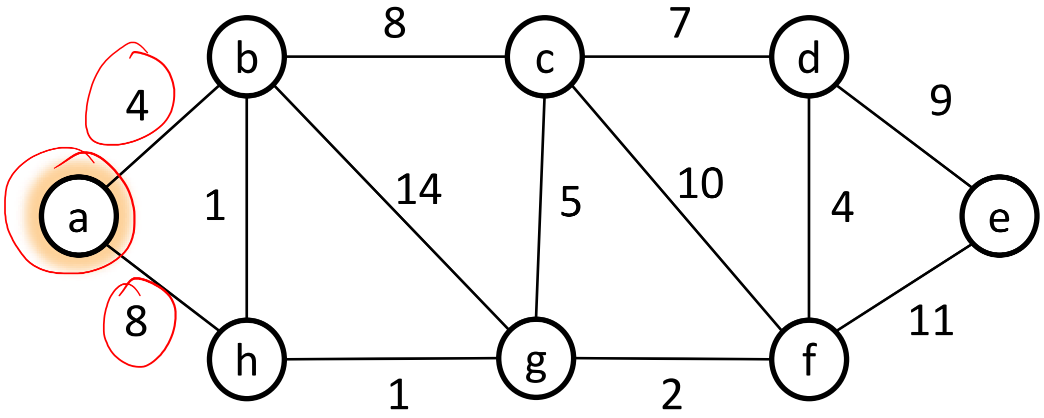
- The problem is called the "Minimum Spanning Tree (MST) Problem."
- MSTs are crucial in circuit design to optimize wire usage.

Prim's Algorithm

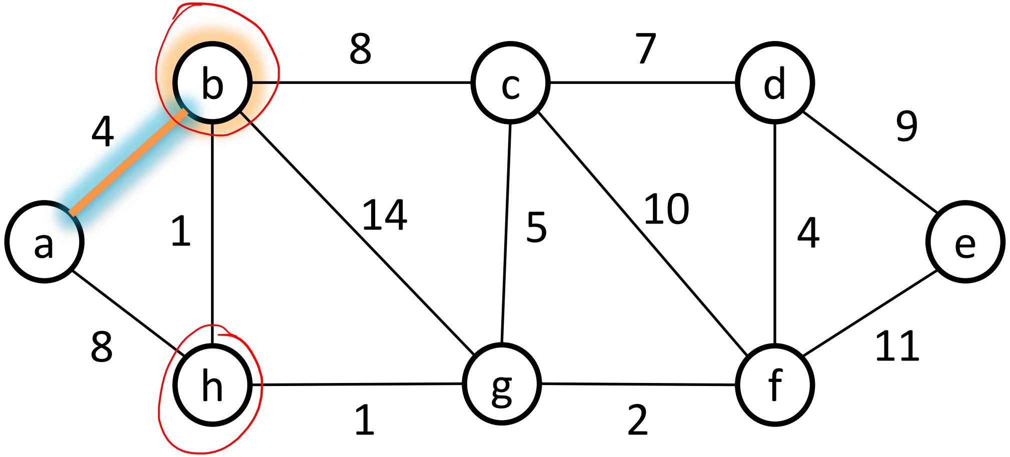
- MST-PRIM(G, w, r)
  - for each vertex u ϵ G.V
    - u.key = ∞
    - u.π = NIL
  - r.key = 0
  - Q = ∅
  - for each vertex u ϵ G.V
    - INSERT(Q, u)

– while $Q \neq \emptyset$
  - u = EXTRACT-MIN(Q)
  - for each vertex v in G.Adj[u]
    – if $v \in Q$ and $w(u, v) < v.key$
      » $v.\pi = u$
      » $v.key = w(u, v)$
      » DECREASE-KEY $(Q, v, w(u, v))$

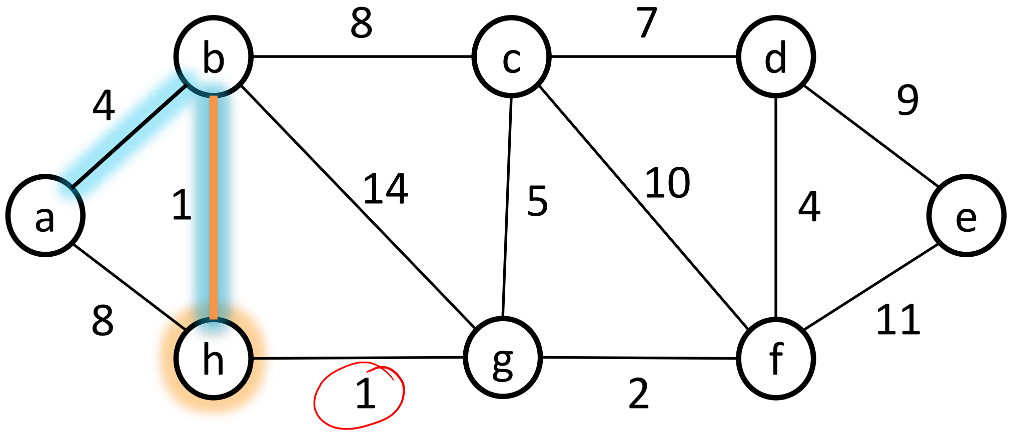
Prim's Algorithm - Example



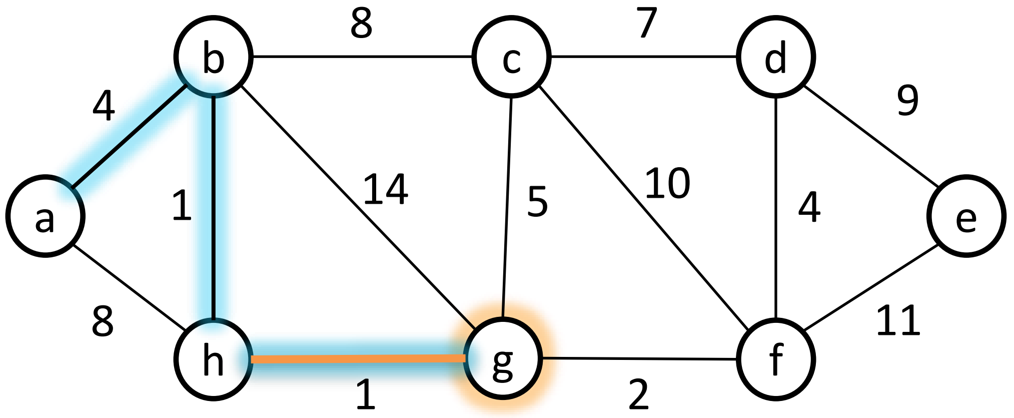
Prim’s Algorithm - Example



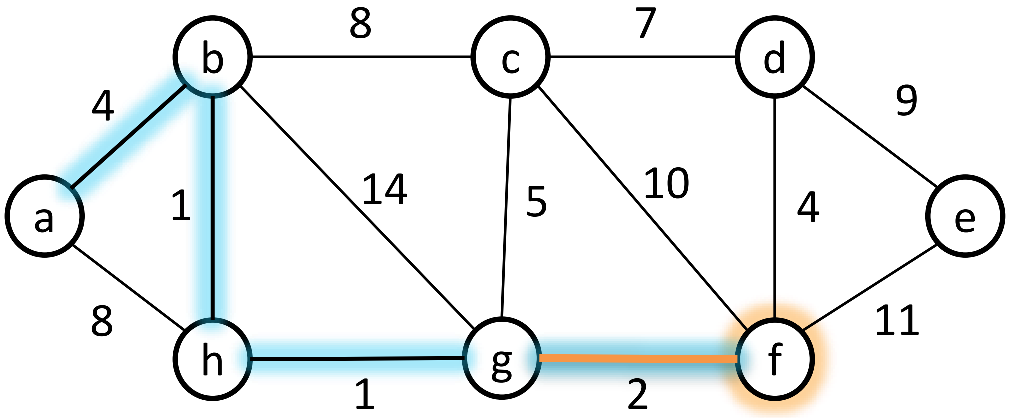
Prim's Algorithm - Example



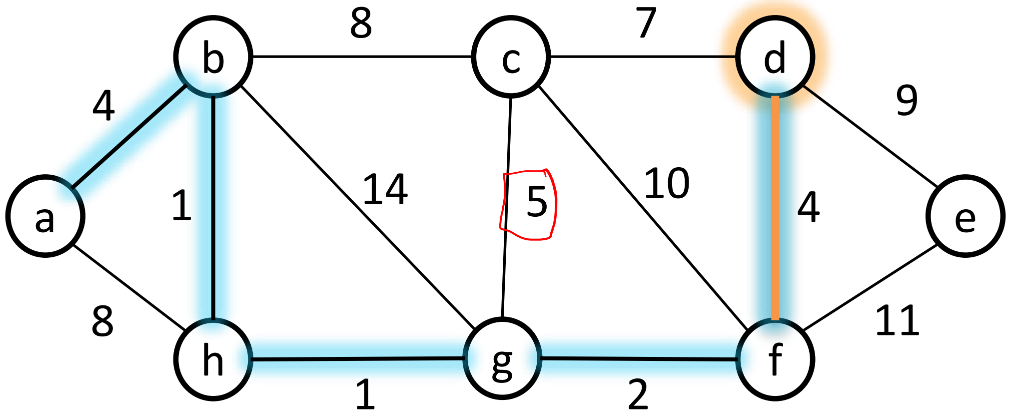
Prim’s Algorithm - Example



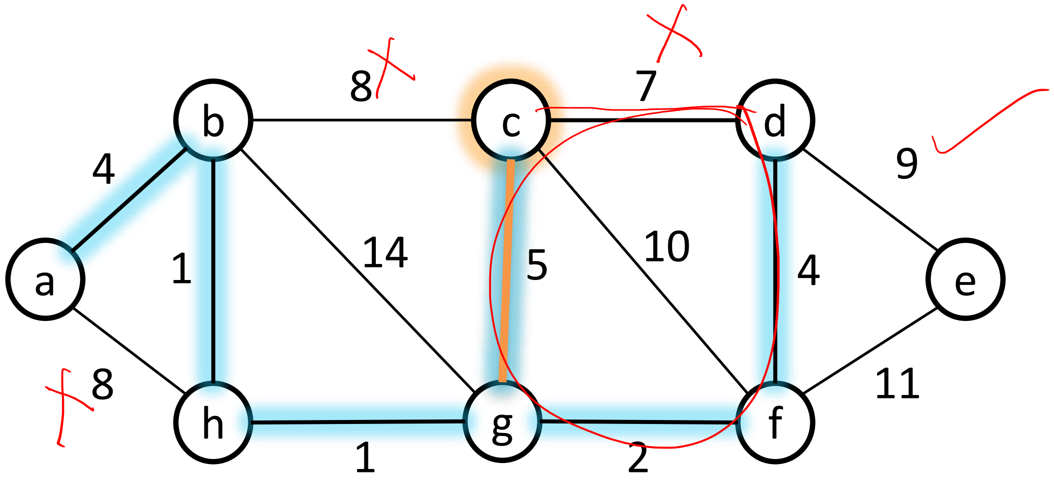
Prim’s Algorithm - Example



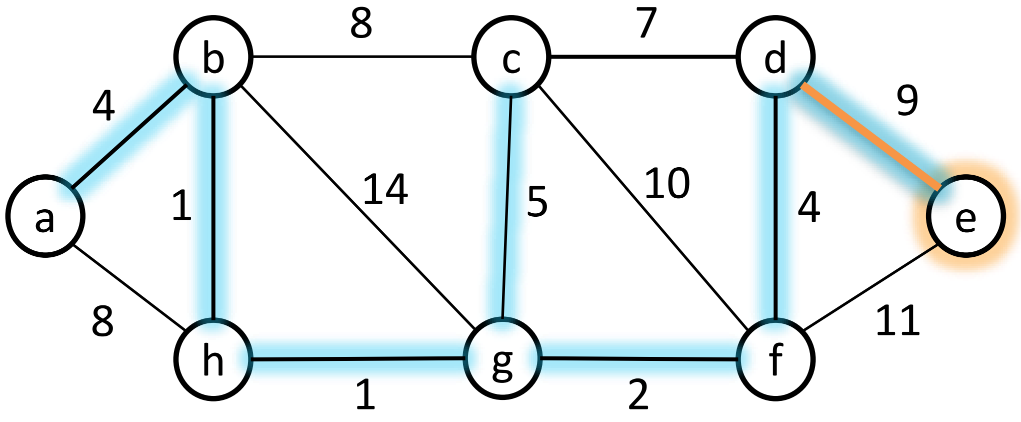
Prim’s Algorithm - Example



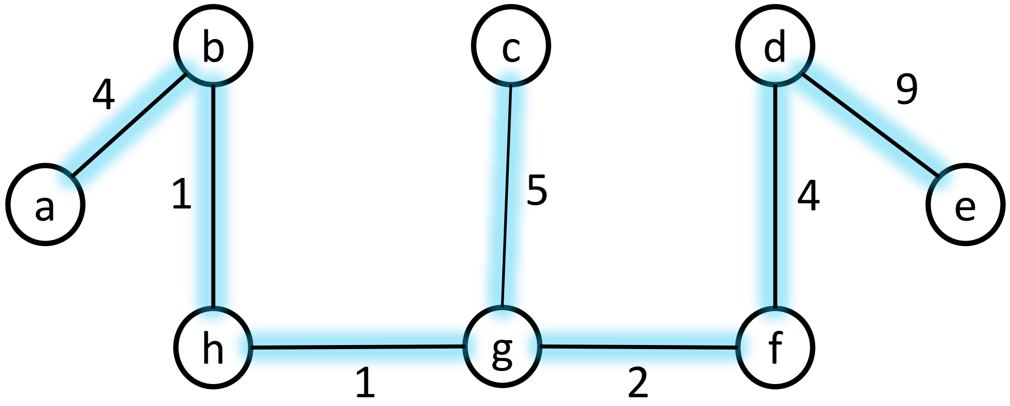
Prim's Algorithm - Example



Prim's Algorithm - Example



Minimum Spanning Tree



Kruskal's Algorithm

- MST-KRUSKAL(G, w)
  - A = Ø
  - for each vertex v ϵ G.V
    - MAKE-SET(v)
  - create a single list of the edges in G.E
  - sort the list of edges into monotonically increasing order by weight w

Initialize the set A to the empty set and create |V| trees, one containing each vertex.

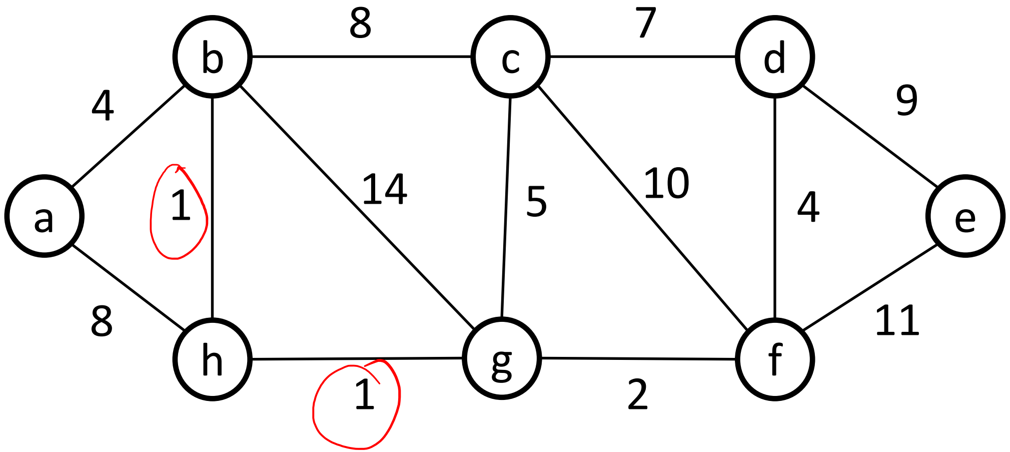
Kruskal's Algorithm

– for each edge (u, v) taken from the sorted list in order
  - if FIND-SET (u) ≠ FIND-SET (v)
    – A = A U {(u, v)}
    – UNION(u, v)
– return A

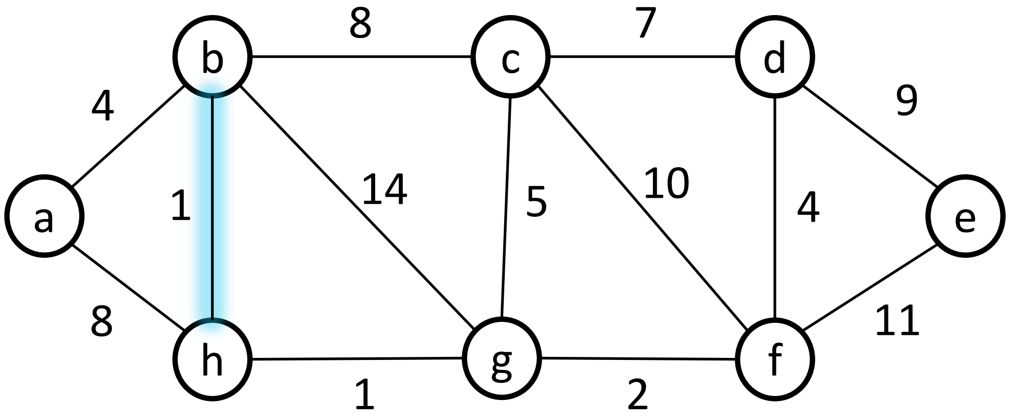
- The for loop examines edges in order of weight, from lowest to highest. The loop checks, for each edge (u, v), whether the endpoints u and v belong to the same tree.

- If they belong to different trees, add the edge (u, v) to A and merge the vertices in the two trees.
- Otherwise, ignore the edge.

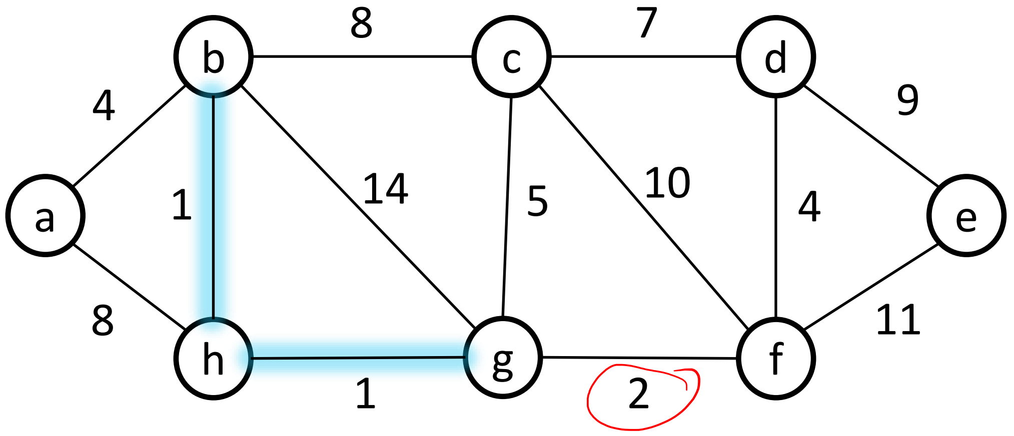
Kruskal's Algorithm : Example -1



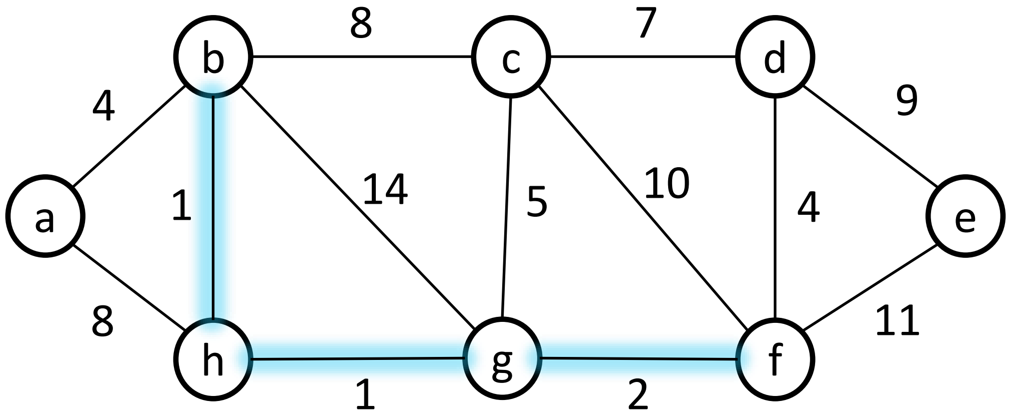
Kruskal's Algorithm : Example -1



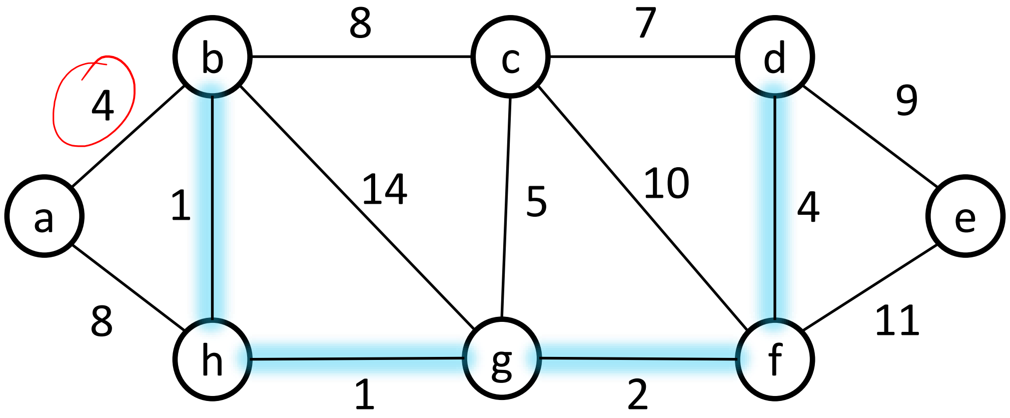
Kruskal's Algorithm : Example -1



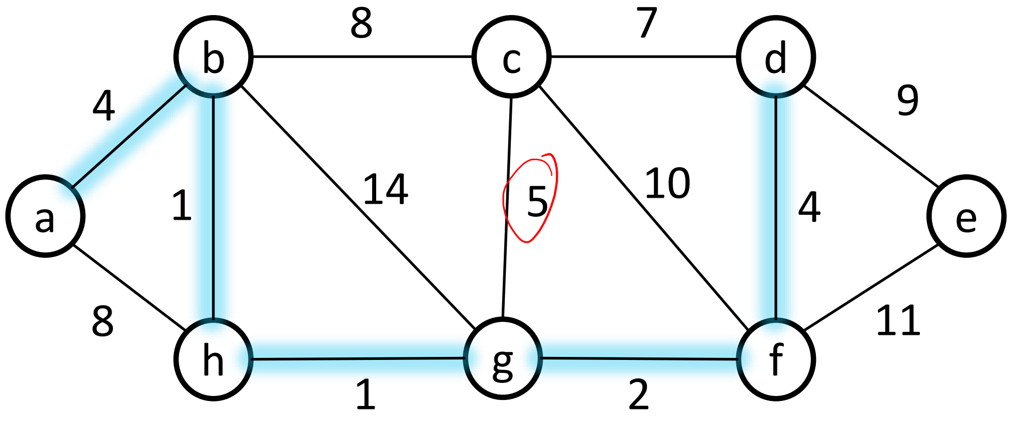
Kruskal's Algorithm : Example -1



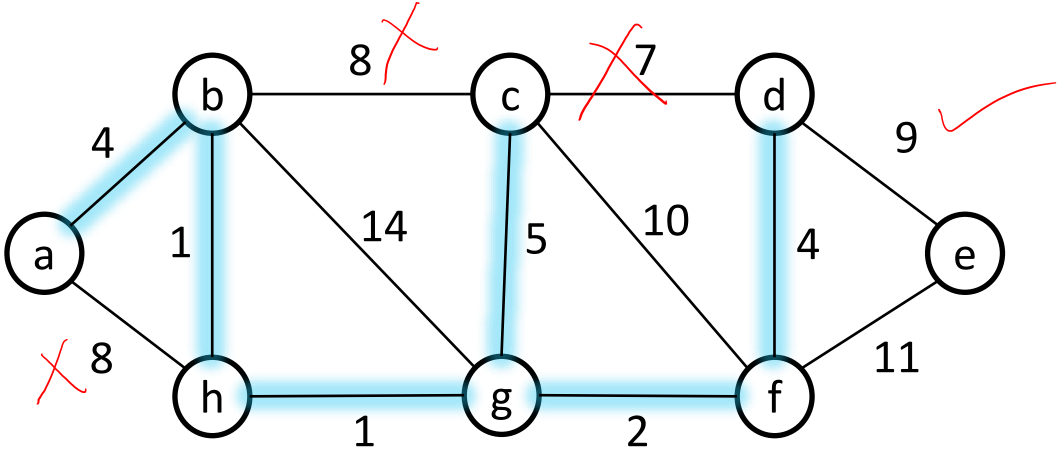
Kruskal's Algorithm : Example -1



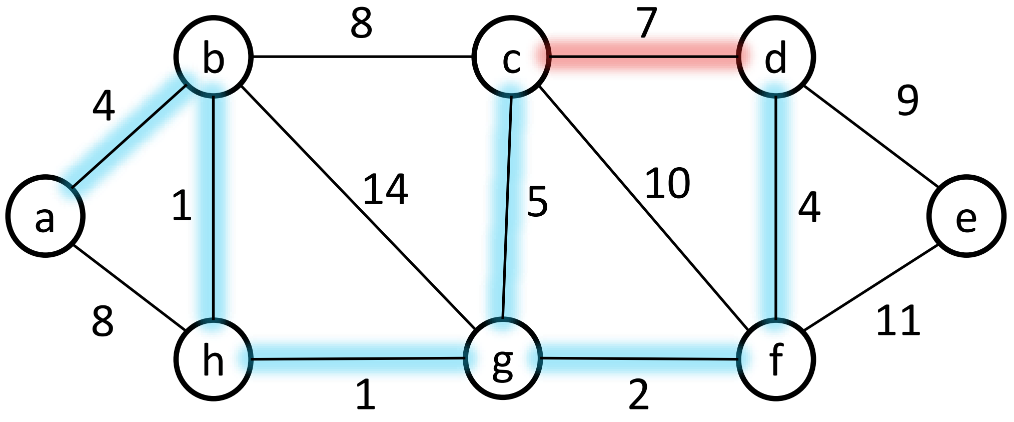
Kruskal's Algorithm : Example -1



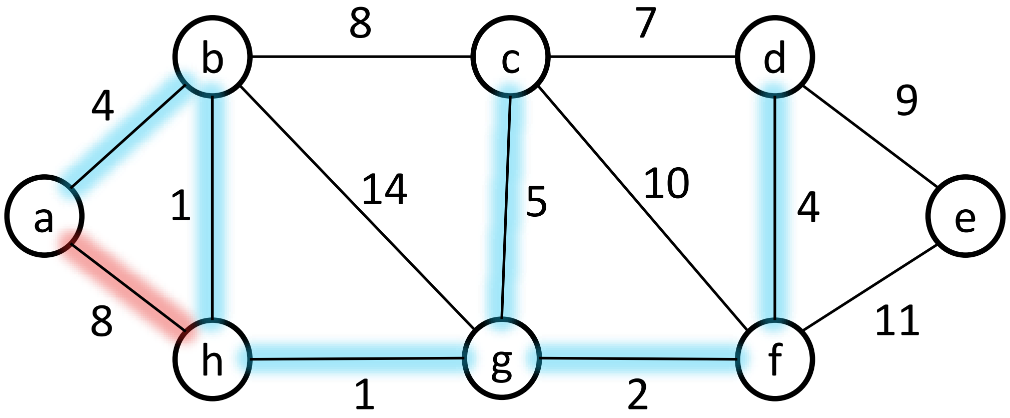
Kruskal's Algorithm : Example -1



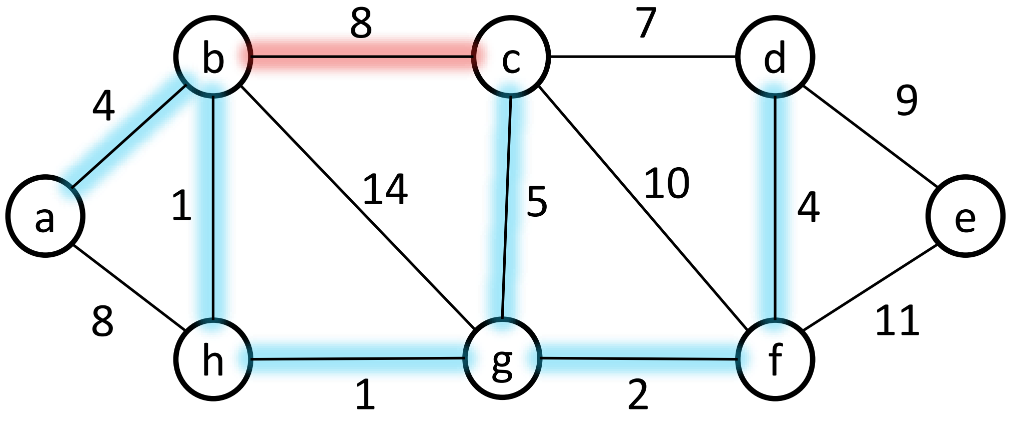
Kruskal’s Algorithm : Example -1



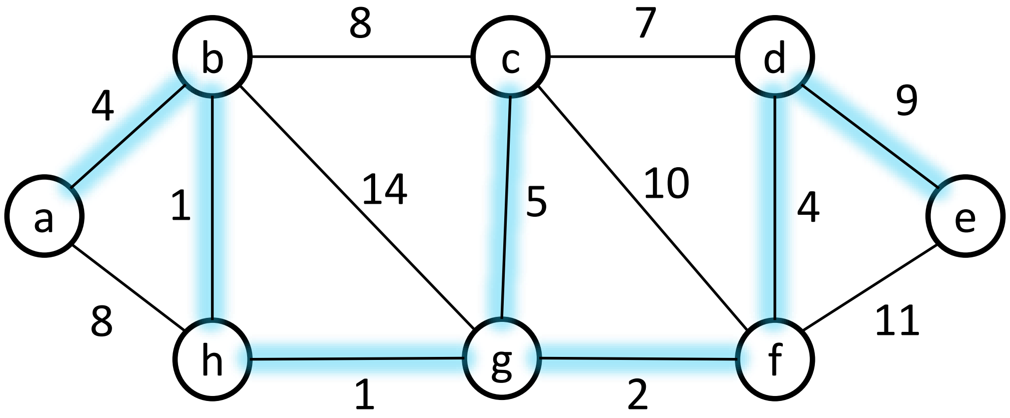
Kruskal’s Algorithm : Example -1



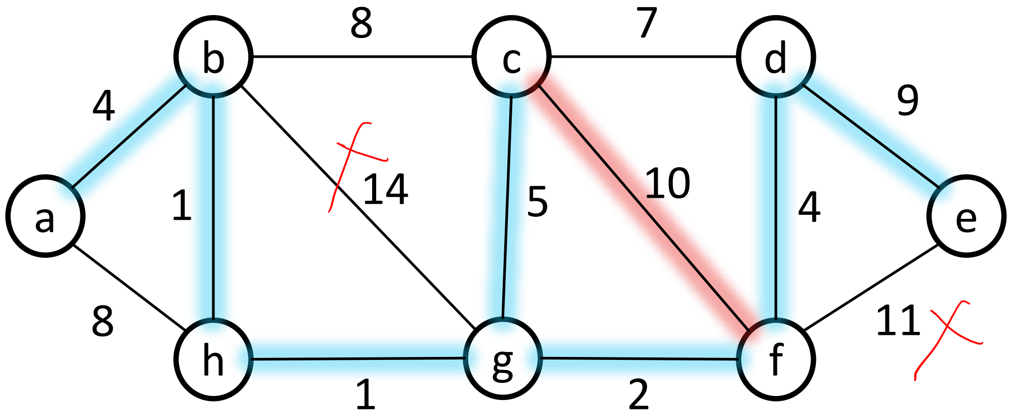
Kruskal's Algorithm : Example -1



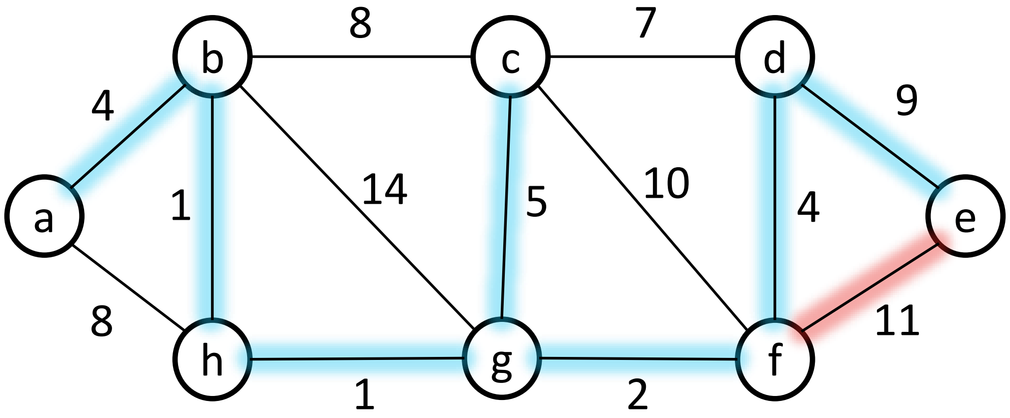
Kruskal's Algorithm : Example -1



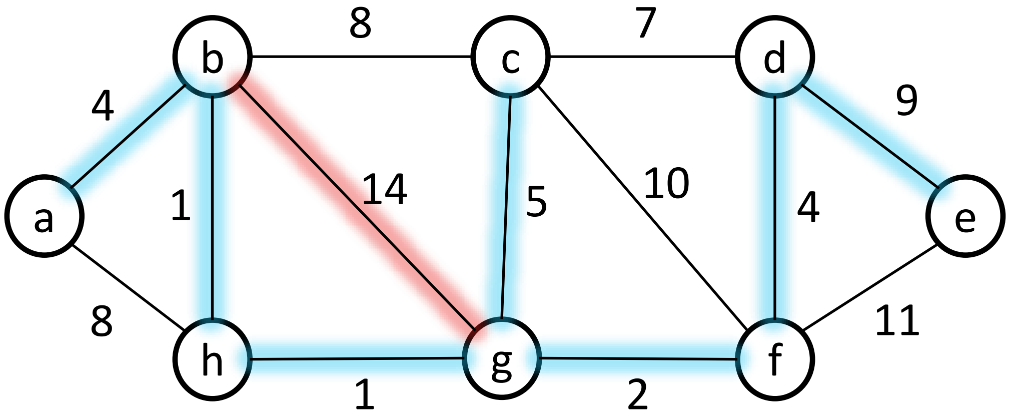
Kruskal's Algorithm : Example -1



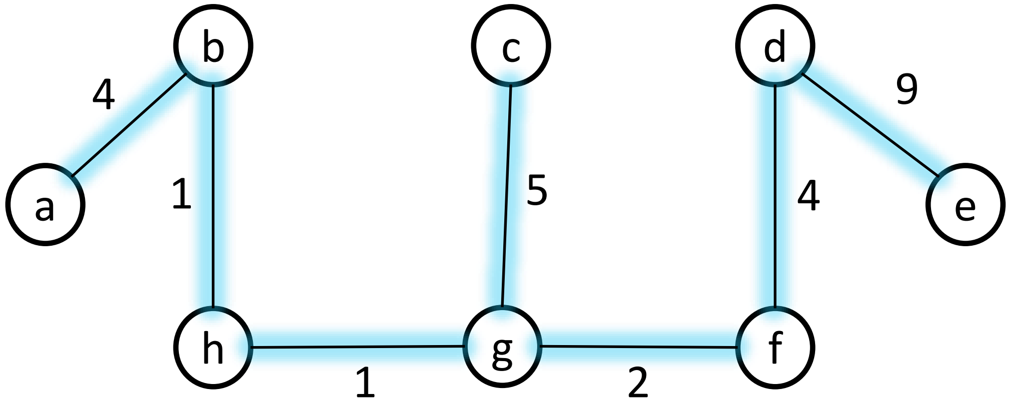
Kruskal's Algorithm : Example -1



Kruskal's Algorithm : Example -1



Minimum Spanning Tree



Shortest Path Algorithms

- **Dijkstra's algorithm** solves the single-source shortest-paths problem on a weighted, directed graph $G = (V, E)$ but requires nonnegative weights on all edges: $w(u, v) \geq 0$ for each edge $(u, v) \in E$.

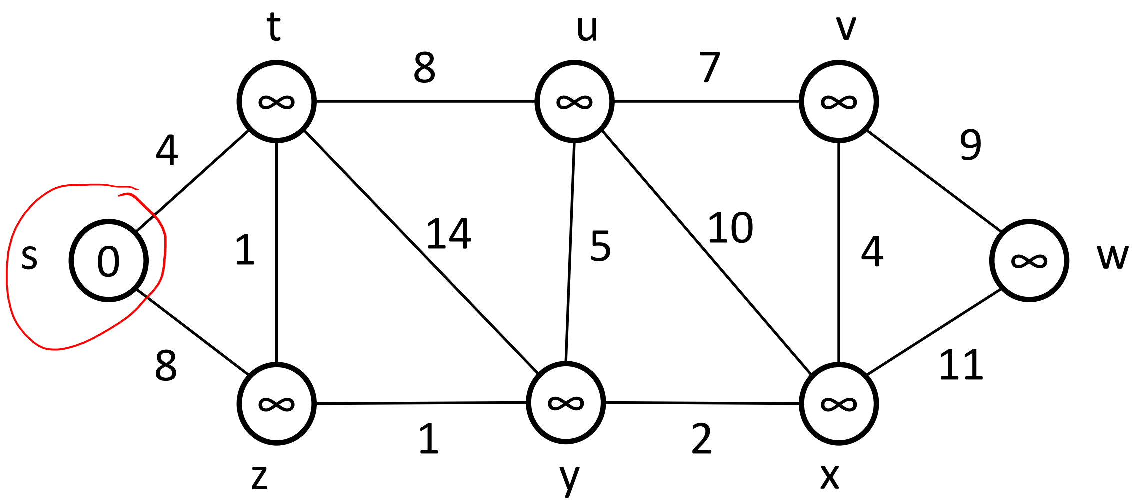
Dijkstra's algorithm

- DIJKSTRA(G, w, s)
  - INITIALIZE-SINGLE-SOURCE(G, s)
  - S = Ø
  - Q = Ø
  - for each vertex u ϵ G.V
    - INSERT(Q, u)

Dijkstra's algorithm

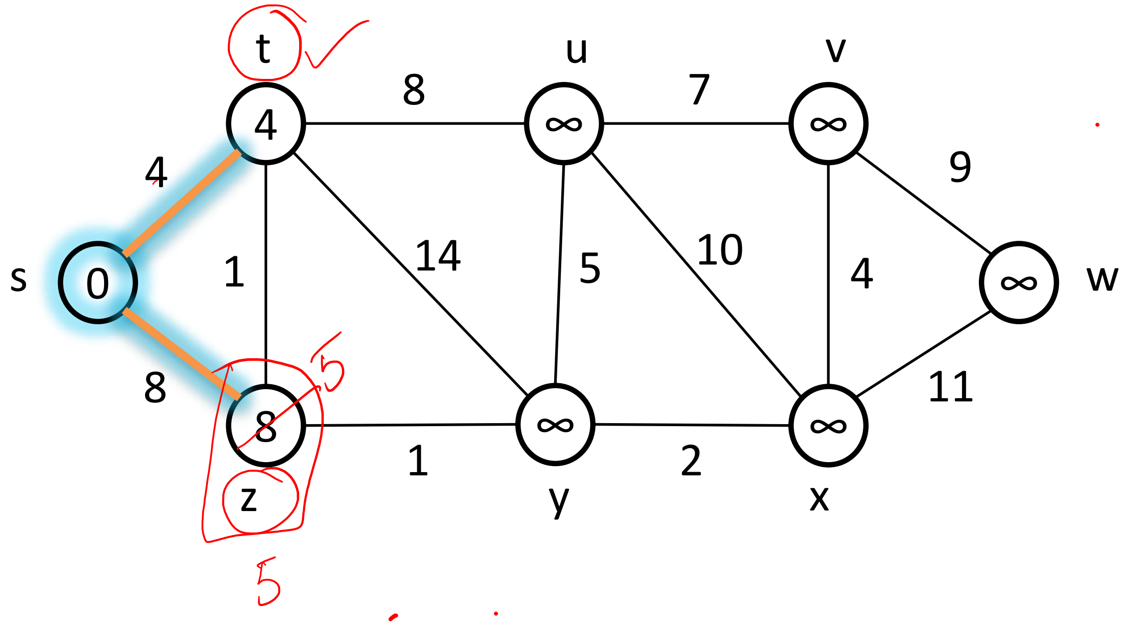
- while $Q \neq \emptyset$
  - u = EXTRACT-MIN (Q)
  - S = S U {u}
  - for each vertex v in G.Adj(u)
    - RELAX(u, v, w)
    - if the call of RELAX decreased v.d
      - » DECREASE-KEY (Q, v, v.d)

Dijkstra's algorithm - Example



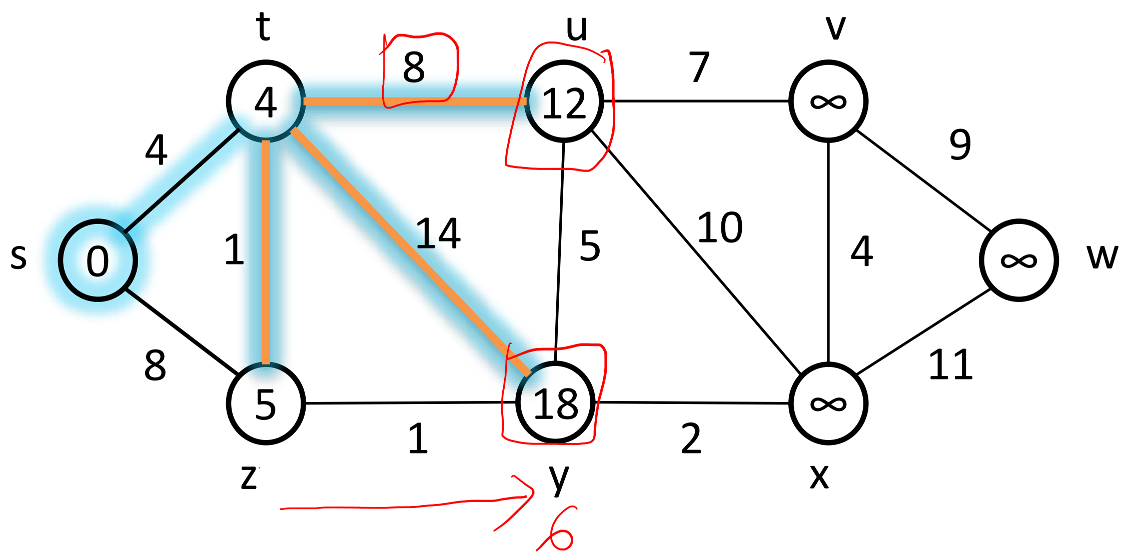
Dijkstra's algorithm - Example

Node 'z' $\cancel{8}$ 5



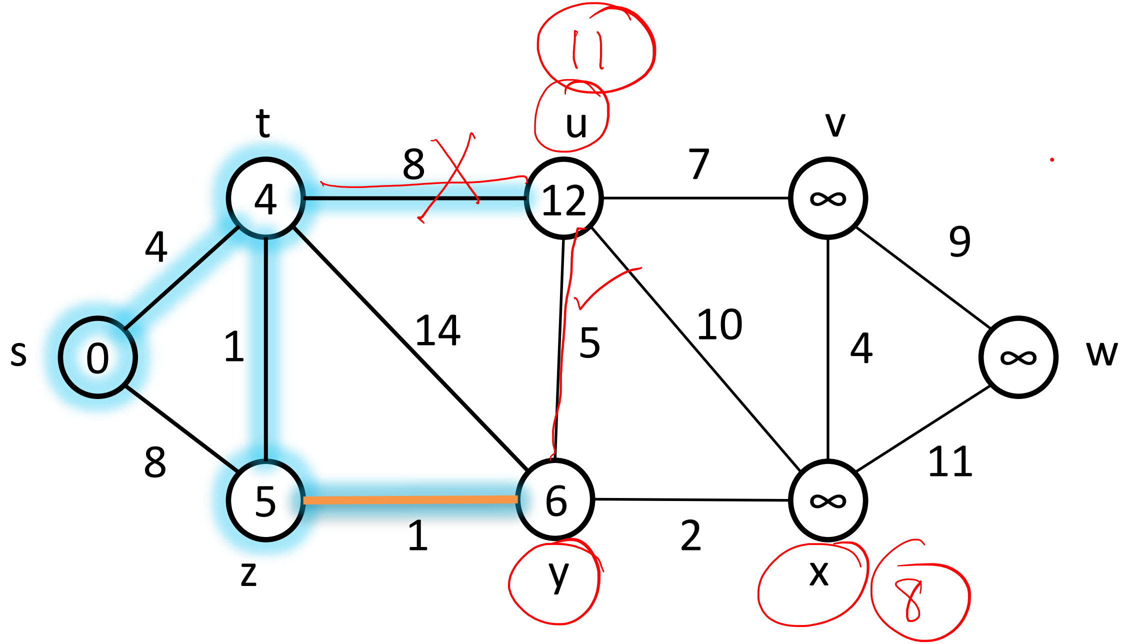
t	$0 + 4 = 4 < \infty$
z	$0 + 8 = 8 < \infty$

Dijkstra's algorithm - Example



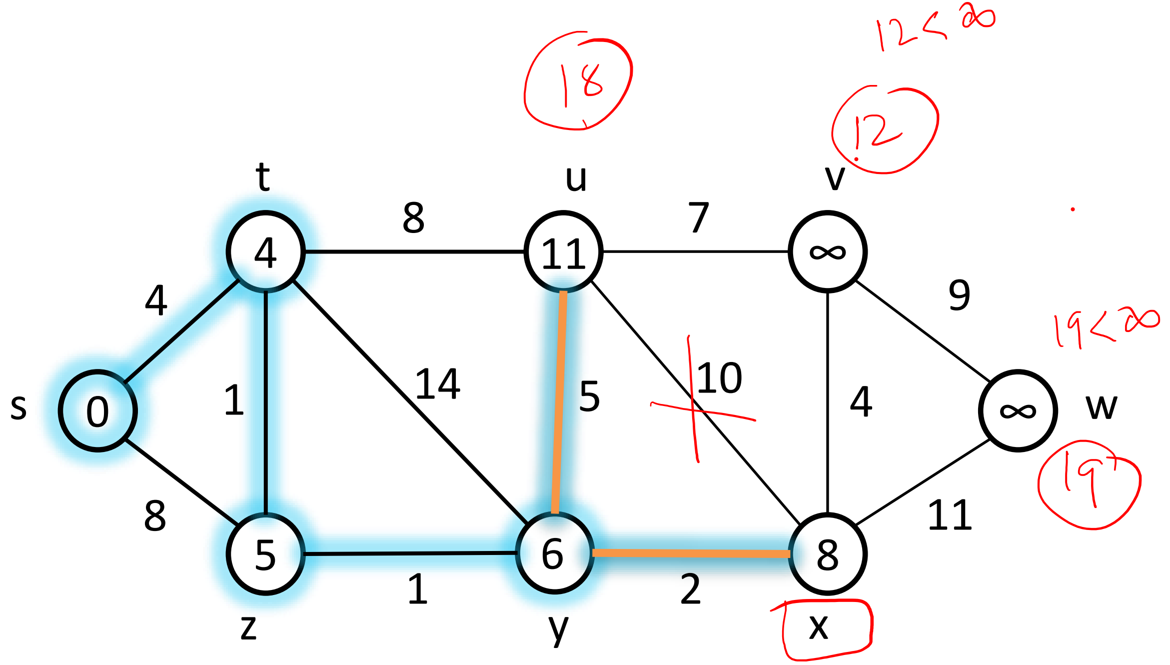
u	4 + 8 = 12 < ∞
y	4 + 14 = 18 < ∞
z	4 + 1 = 5 < 8

Dijkstra's algorithm - Example



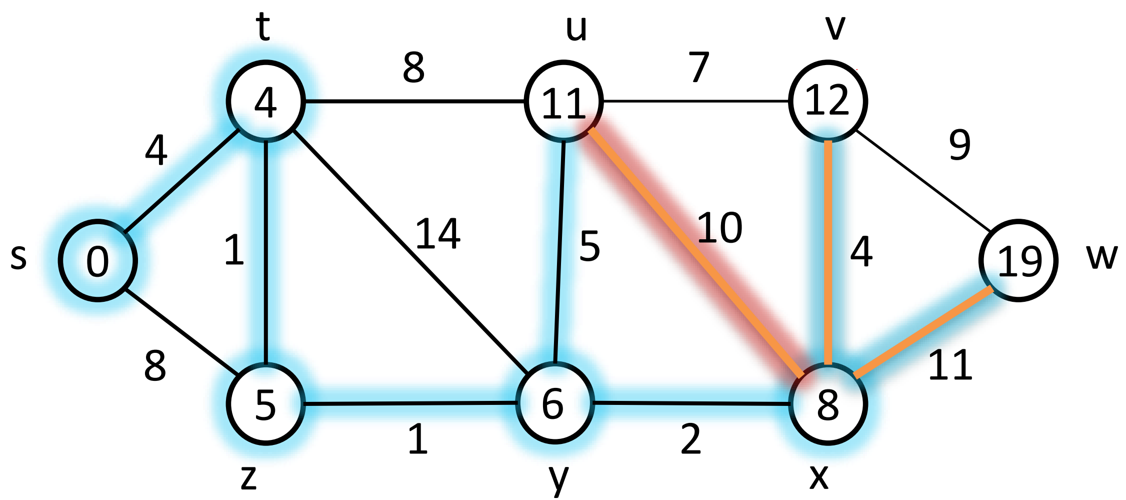
y	5 + 1 = 6 < 18

Dijkstra's algorithm - Example



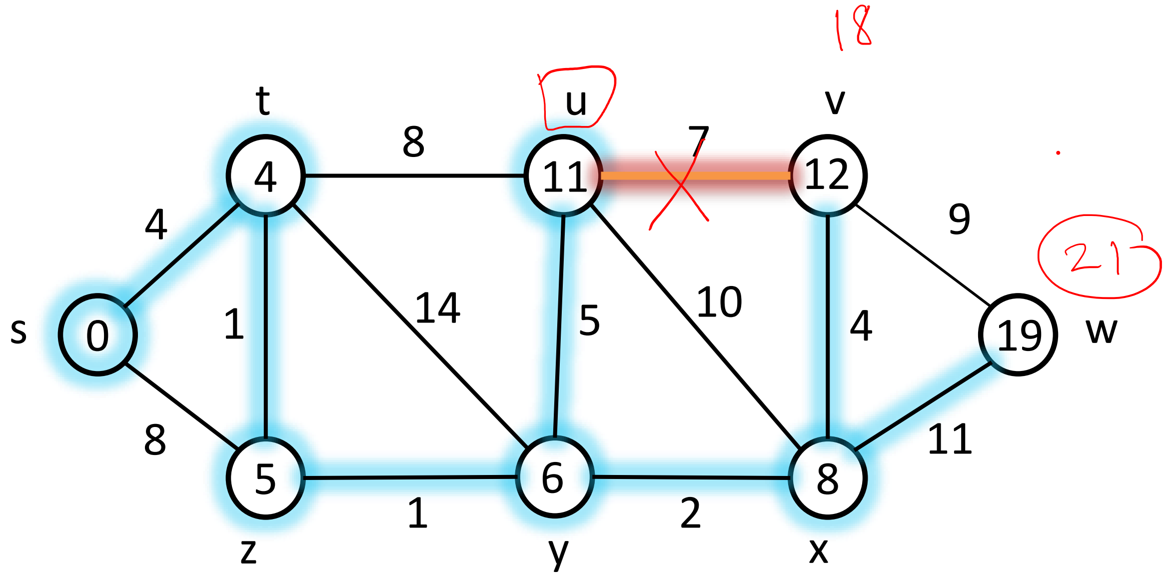
u	6 + 5 = 11 < 12
x	6 + 2 = 8 <∞

Dijkstra's algorithm - Example



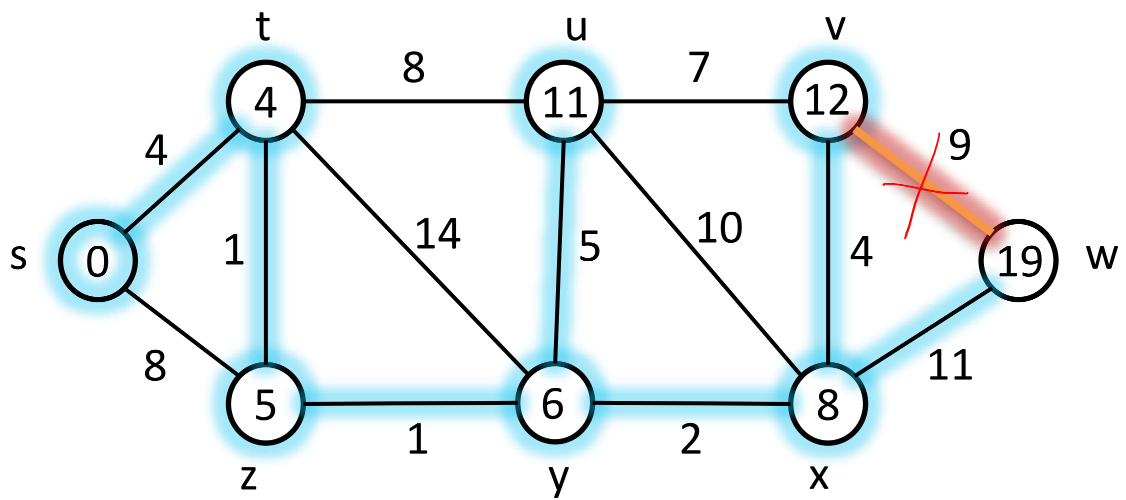
u	8 + 10 = 18 > 12
v	8 + 4 = 12 < ∞
w	8 + 11 = 19 < ∞

Dijkstra's algorithm - Example



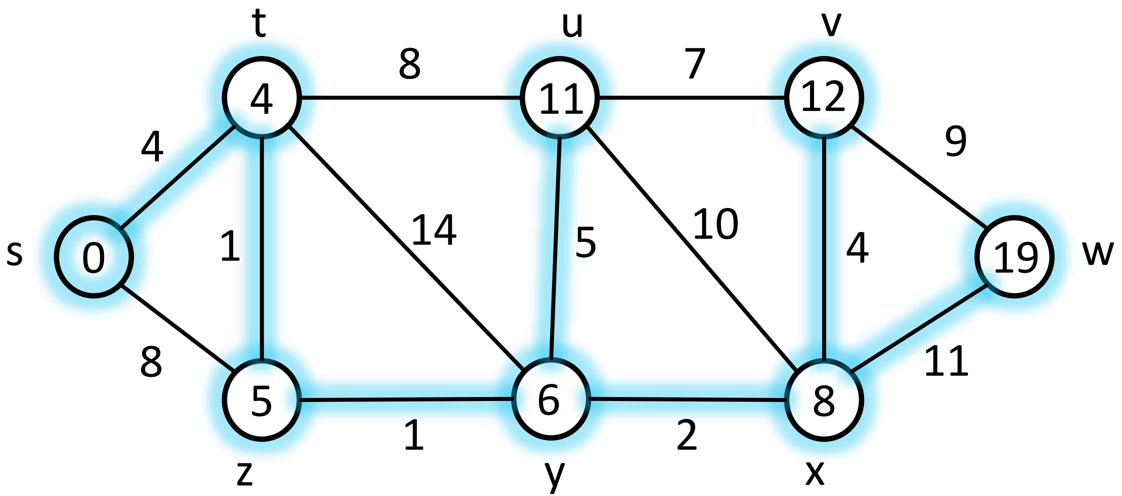
v	11 + 7 = 18 > 12

Dijkstra's algorithm - Example



v	$12 + 9 = 21 > 19$

Dijkstra's algorithm - Example



Summary

- Discussed Graph search algorithms: DFS and BFS
- Explained the Minimum Spanning Tree Algorithms: Prim's Algorithm and Kruskal's Algorithm
- Discussed shortest Path Algorithms: Dijkstra's Algorithm with examples
- These algorithms have a lot of applications in VLSI Physical Design flow.

Thank You