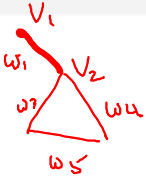
Data Structures and Algorithms

Prof. Ganesh Ramakrishnan,
Prof. Ajit Diwan,
Prof. D.B. Phatak

Department of Computer Science and Engineering
IIT Bombay

Session: Spanning Tree Algorithm
(Prim's Algorithm)

Prim's Algorithm



1. Greedy algorithm
2. Finds minimum spanning tree for a weighted undirected graph.

$T \subseteq G$ with $\min \sum_{e \in T} w(e)$ st $\forall$ vertex $v \in G$ $\exists$ $e \in T$ that covers v.

Prim's Algorithm

Key idea: ① Grow the MST T
② Keep track of the smallest
key[ ] { edge weight from each
$v \in \{V(G) \setminus V(T)\}$ to a vertex $w \in V(T)$
③ Add $v \in \{V(G) \setminus V(T)\}$ to T with
smallest value of such weight

```
Algorithm MST-Prim(G, w, r)
P ← new Min-heap(key)
for u ∈ G.getVertices() do
    key[u] = ∞
    predecessor[u] = NULL
    P.insert(u)
end for
key[r] = 0
while P.isNotEmpty() do
    u ← P.getMin()
    for v ∈ G.adjacentVertex(u) do
        if v ∈ P & w(u, v) < key(v) then
            predecessor[v] = u
            key[v] = w(u, v)
        end if
    end for
end while
```

minheap

→ Arbitrarily choose a root

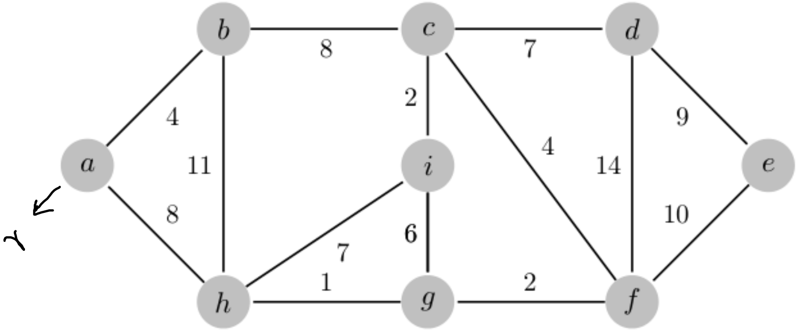
} updating key[ ] for adjacent vertices (to u)

$\forall (u,v)$ st
pred[u]=v
or pred[v]=u
edge $(u,v) \in E(T)$

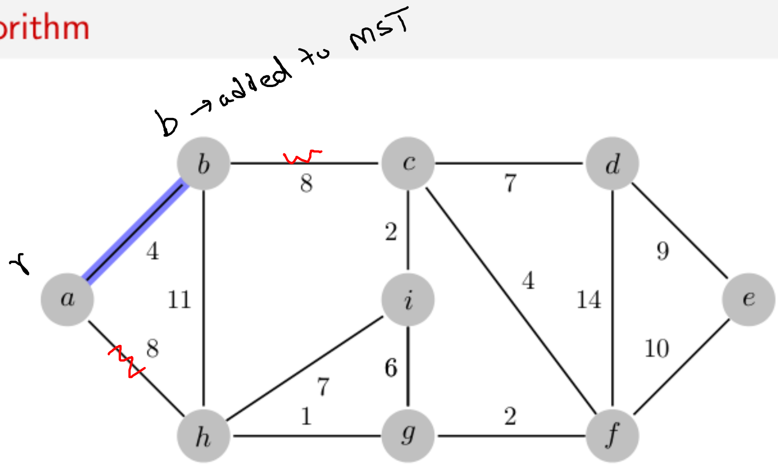
Figure: Prim's Algorithm

Loop invariant:
- Vertices in $V(G) \setminus P$ are already part of MST
- $\forall v \in P$ if $pred[v] \neq NULL$ then $key[v] < \infty$ & $key[v]$ is wt of $(v, \pi(v))$, $\pi(v) \in MST$

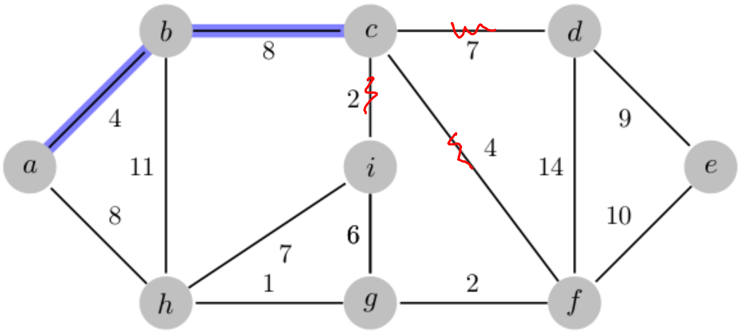
Prim's Algorithm



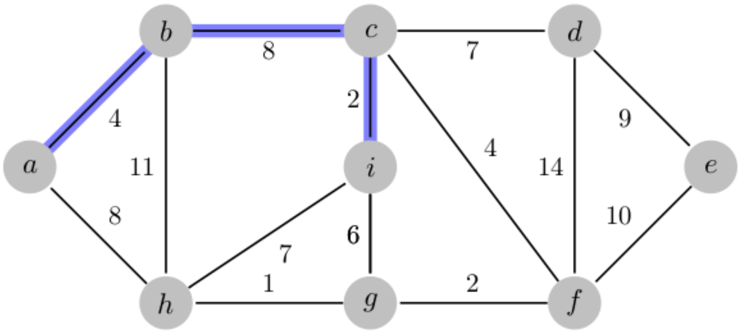
Prim's Algorithm



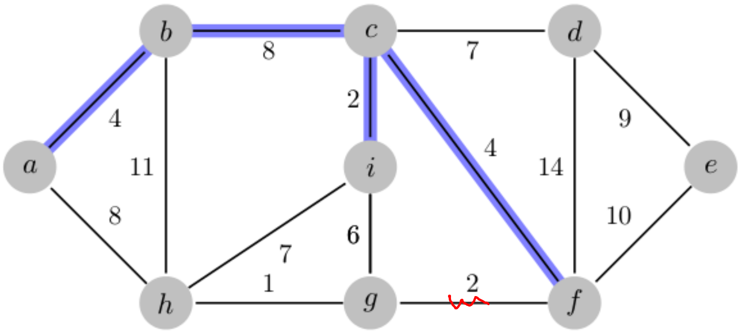
Prim's Algorithm



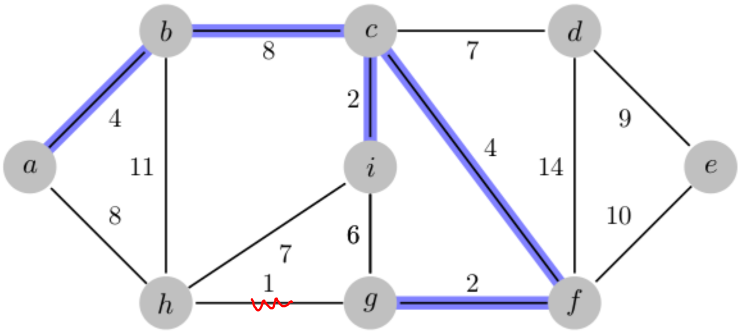
Prim's Algorithm



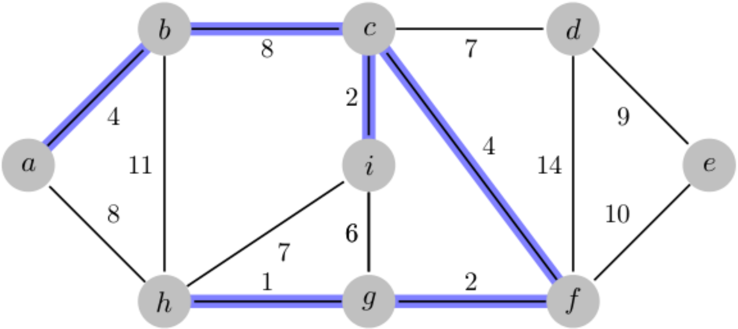
Prim's Algorithm



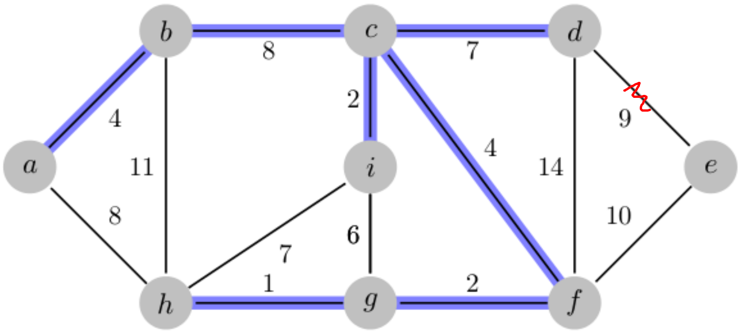
Prim's Algorithm



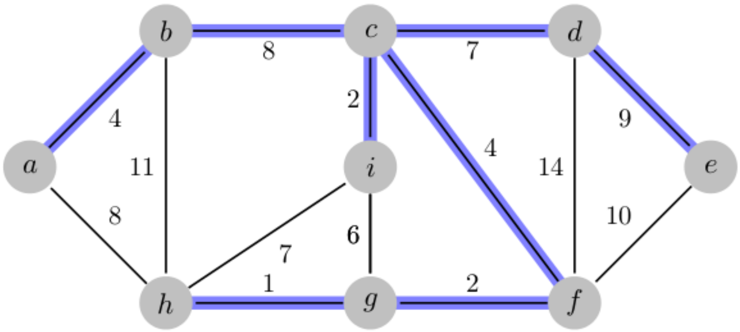
Prim's Algorithm



Prim's Algorithm



Prim's Algorithm



Analysis of Prim's Algorithm

Recall Heapification

```
Algorithm MST-Prim(G, w, r) (G)
P ← new Min-heap(key)
for  u ∈ G.getVertices() do
     key[u] = ∞
     predecessor[u] = NULL
     P.insert(u)  ⟹  c1
end for  ⟹  c2  × |V| times
key[r] = 0
while P.isNotEmpty() do
     u ← P.getMin()  ⟹  c3  × log|V| times
     for v ∈ G.adjacentVertex(u) do
          if v ∈ P  &  w(u, v) < key(v) then
               predecessor[v] = u
               key[v] = w(u, v)  ⟹  c4  × log|V| times
          end if
     end for  ⟹  c5  × deg(V) times
end while  ⟹  c6  × |V| times
```

} $O(|V|)$

updat. heap (P)

Figure: Prim's Algorithm

$$T(n) = c_4 c_5 \log|V| \underbrace{\sum_{v \in V} deg(v)}_{|E|} + \underline{c_3 c_6 |V| \log|V|} + c_1 c_2 |V| = O(|E| + |V|) \log|V|$$

Thank you