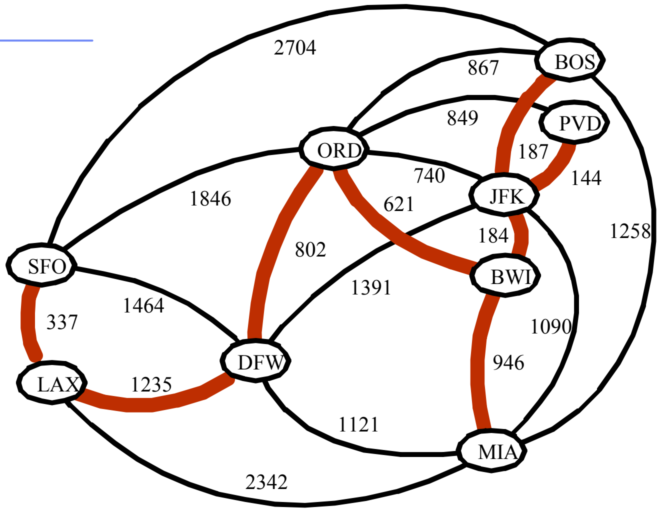
Minimum Spanning Trees



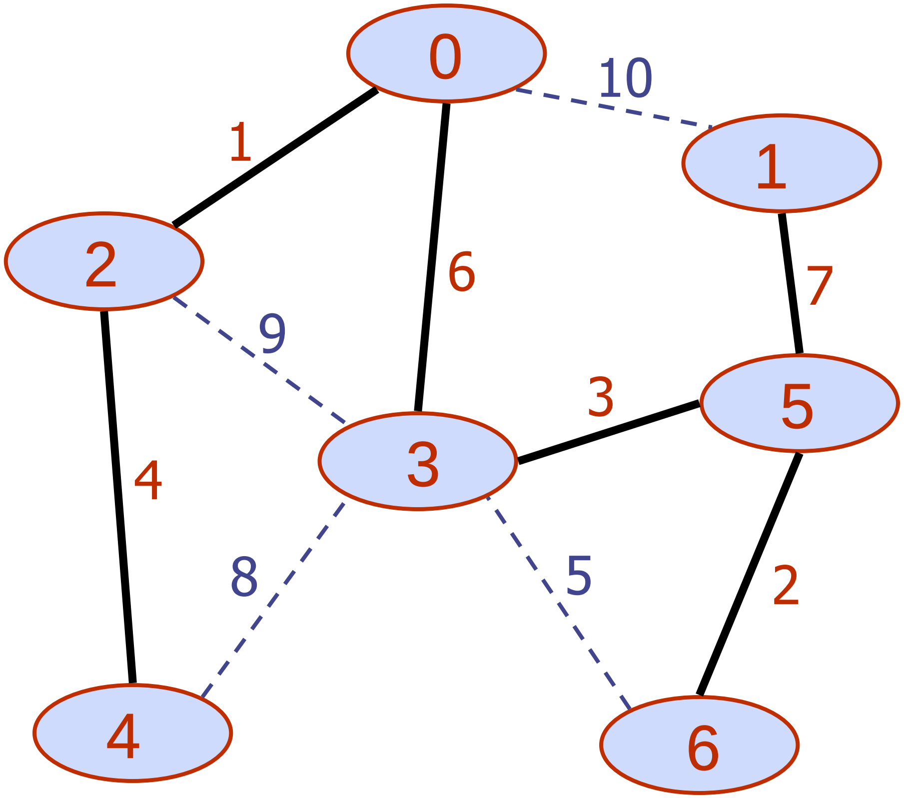
Minimum Spanning Trees (§ 12.7)

Spanning subgraph
- Subgraph of a graph G containing all the vertices of G

Spanning tree
- Spanning subgraph that is itself a (free) tree

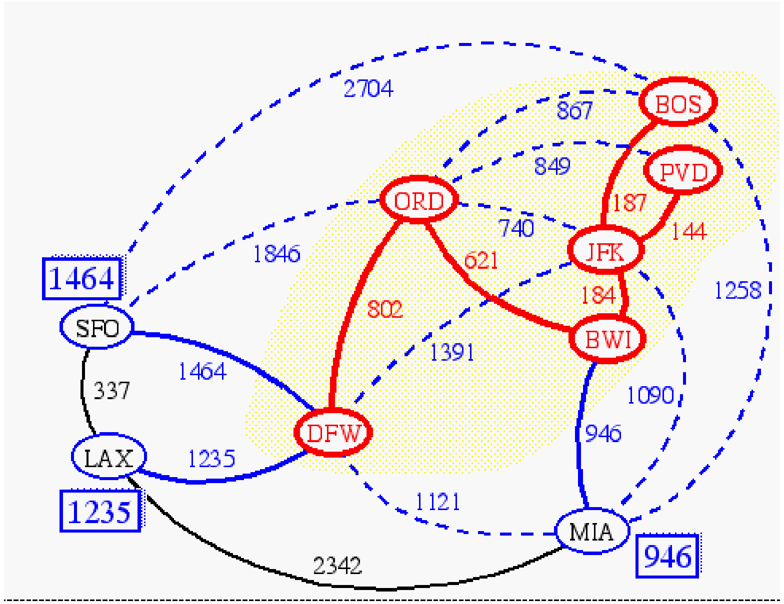
Minimum spanning tree (MST)
- Spanning tree of a weighted graph with minimum total edge weight

Applications
- Communications networks
- Transportation networks



Prim-Jarnik's Algorithm (§ 12.7.2)

- Similar to Dijkstra's algorithm (for a connected graph)
- We pick an arbitrary vertex s and we grow the MST as a cloud of vertices, starting from s
- We store with each vertex v a label $d(v)$ = the smallest weight of an edge connecting v to a vertex in the cloud
- At each step:
  - We add to the cloud the vertex u outside the cloud with the smallest distance label
  - We update the labels of the vertices adjacent to u



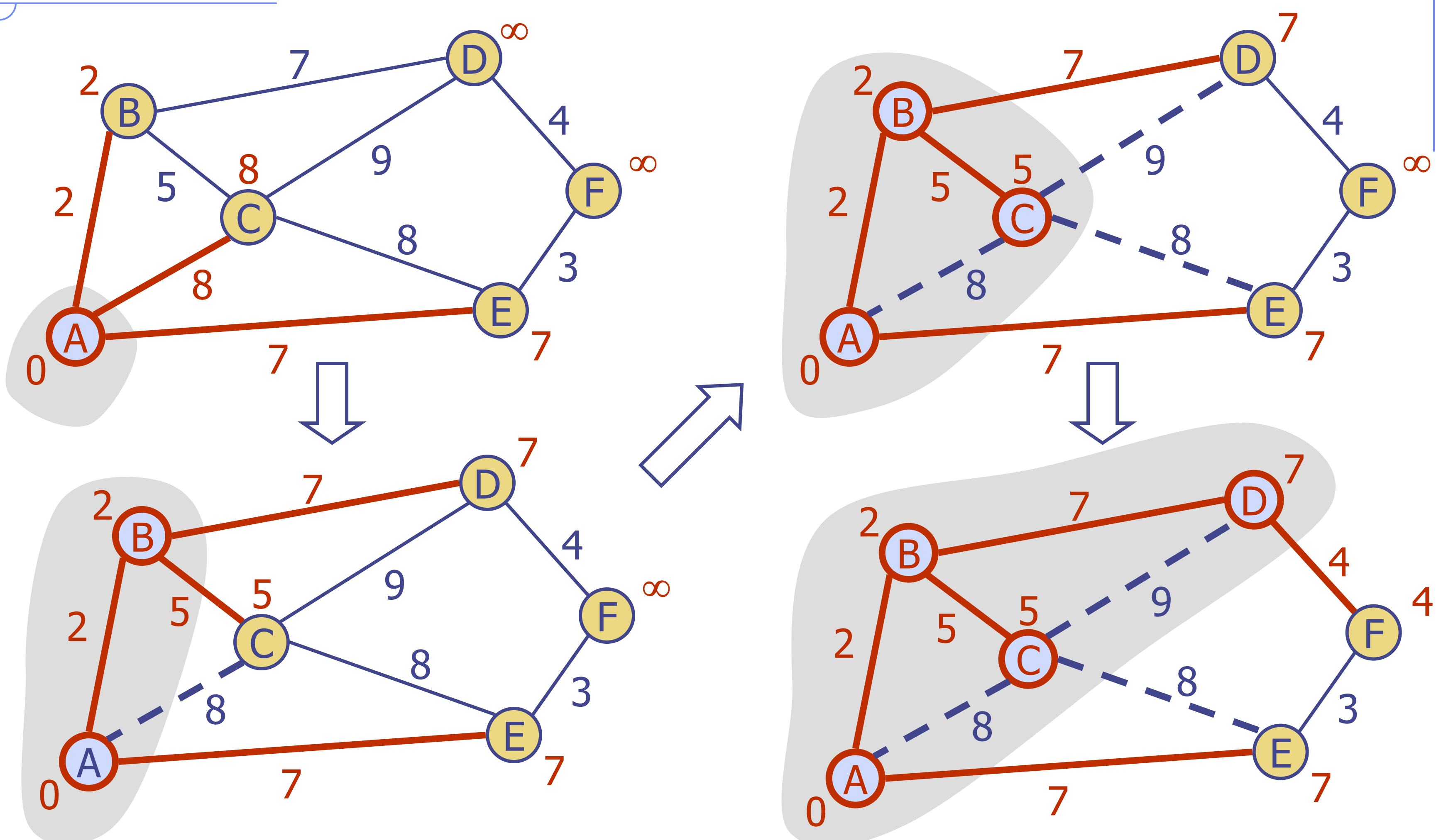
Prim-Jarnik's Algorithm (cont.)

- We store three labels with each vertex:
  - Distance
  - Parent edge in MST
  - Locator in priority queue
-

Algorithm *PrimJarnikMST*(*G*)
    $Q \leftarrow$ new queue
    $s \leftarrow$ a vertex of G
    for all $v \in G.vertices()$ {
        if $v = s$ **then** $v.d = 0$
        else $v.d = \infty$
        $setParent(v, \emptyset)$
        $l \leftarrow Q.insert(v)$
    }

while $\neg Q.isEmpty()$
    $u \leftarrow$ *node with smallest d value in Q*
    for all $e \in G.incidentEdges(u)$ {
        $z \leftarrow G.opposite(u,e)$
        $r \leftarrow weight(e)$
        if $r < z.d$ *then* {
            $z.d = r$; $setParent(z,e)$
        } }

Example



∞
D
7
2
B
4
9
8
2
5
C
F
∞
8
3
8
A
E
0
7
7
7
D
2
7
B
4
9
5
2
5
C
F
∞
8
3
8
A
E
0
7
7
7
D
7
2
B
4
9
5
2
5
C
F
∞
8
3
8
A
E
0
7
7
7
D
7
2
B
4
4
9
5
F
2
5
C
8
3
8
A
E
0
7
7

Example (contd.)