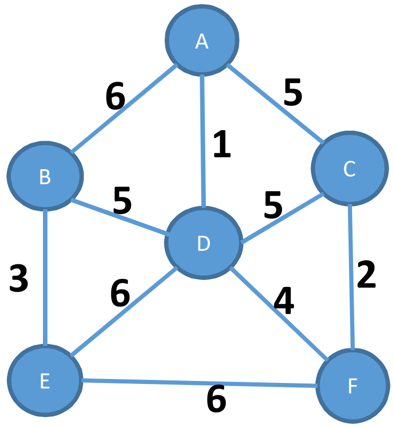
Prim's Algorithm Example
More Proofs
Application of MST to Clustering

CSCI 3100

Prim's algorithm with priority queue

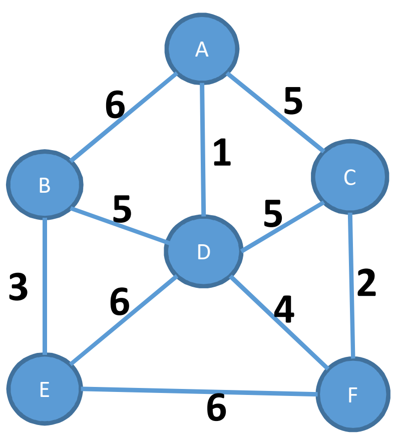
```
MST-PRIM(G, w, r)
for each u ∈ G.V
    u.key = ∞
    u.π = NIL
r.key = 0
Q = G.V
while Q ≠ ∅
    u = EXTRACT-MIN(Q)
    for each v ∈ G.Adj[u]
        if v ∈ Q and w(u, v) < v.key
            v.π = u
            v.key = w(u, v)
```

Prim’s algorithm example: r = 1



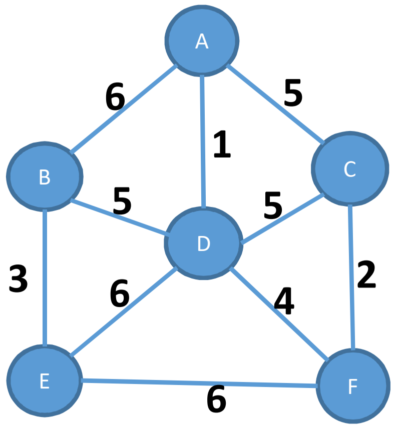
A.key =0 A.π = NIL
B.key = ∞ B.π = NIL
C.key = ∞ C.π = NIL
D.key = ∞ D.π = NIL
E.key = ∞ E.π = NIL
F.key = ∞ F.π = NIL
Q = A B C D E F
u = EXTRACT_MIN(Q)
u = A

Prim’s algorithm example: r = 1



Q = B C D E F
For each v in G.Adg(A)
 if v in Q and w (u, v) < v.key
  v. π = u
  v.key = w(u,v)
(A, B): B is in Q, w(A,B) < B.key
 B. π = A, B.key = 6, Q = B C D E F
(A, D): D is in Q, w(A,D) < D.key
 D. π = A, D.key = 1, Q = D B C E F
(A, C): C is in Q, w(A, C) < C.key
 C. π = A, C.key = 5, Q = D B C E F

B.key = 6, D.key = 1, C.key = 5, E.key = ∞, F.key = ∞



Q = D B C E F

u = EXTRACT_MIN(Q) = D

(D, A): A is not in Q

(D, B): B is in Q, w(D,B) < B.key

B. π = D, B.key = 5, Q = B C E F

(D, C): C is in Q, w(D, C) == C.key

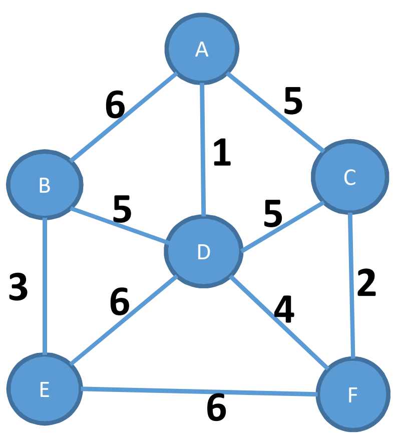
(D, E): E is in Q, w(D, E) < E.key

E. π = D, E.key = 6, Q = B C E F

(D, F): F is in Q, w(D, F) < F.key

F. π = D, F.key = 4, Q = F B C E

B.key = 5, C.key = 5, E.key = 6, F.key = 4



Q = F B C E

u = EXTRACT_MIN(Q) = F

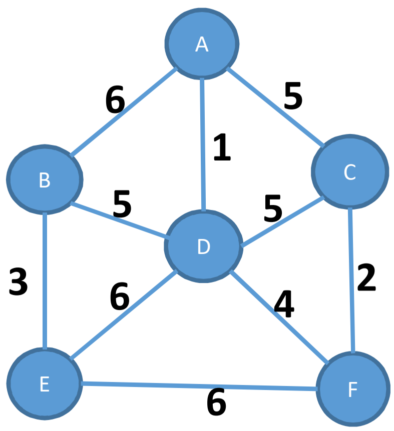
(F, E): E is in Q, w (F, E) > F.key

(F, D): D is not in Q

(F, C): C is in Q, w(F, C) < F.key

C. π = F, C.key = 2, Q = C B E

B.key = 5, C.key = 2, E.key = 6



Q = C B E

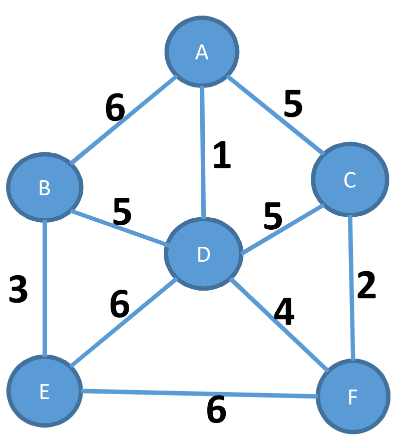
u = EXTRACT_MIN(Q) = C

(C, A): A is not in Q

(C, D): D is not in Q

(C, F): F is not in Q

B.key = 5, C.key = 2, E.key = 6



Q = B E

u = EXTRACT_MIN(Q) = B

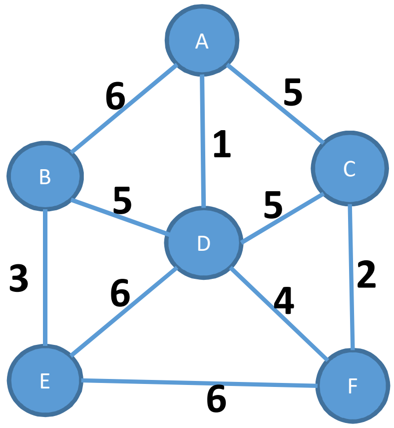
(B, A): A is not in Q

(B, D): D is not in Q

(B, E): E is in Q, w(B,E)<E.key

E. π = B, E.key = 3

Prim's algorithm example: r = 1



$A.\pi$ = NIL

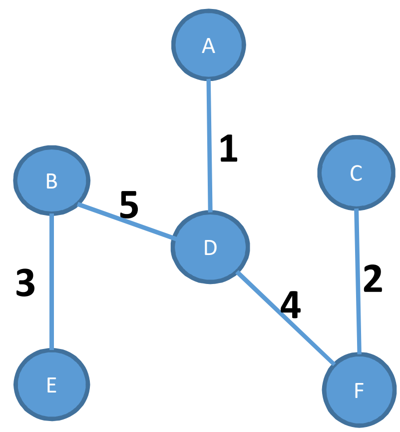
$B.\pi$ = D

$C.\pi$ = F

$D.\pi$ = A

$E.\pi$ = B

$F.\pi$ = D



Cycle property

Claim: Assume that all edge costs are distinct. Let C be any cycle in G = (V, E), and let edge e = (v, w) be the largest edge in C. Then e is not in any minimum spanning tree of G.

Which of the following is NOT a good approach for a proof of this claim?

A. Assume e is in some minimum spanning tree of G, show that this leads to a contradiction.

B. Let T be a spanning tree containing e, show that T is not a minimum spanning tree.

C. Assume e is not in any minimum spanning tree of G. Show that your assumption is consistent with the rest of the facts.

D. All of the above approaches are good.

Proof A:

Let T be a spanning tree that contains *e = (v, w)*. We need to show that T does not have the minimum possible weight.

Let's delete edge e from T. This partitions the vertices of G into two components S and V – S. Let node v be in S and node w be in V-S.

Since G has a cycle C and edge *e* is part of that cycle, there is a path from *v* to *w* in G that does not involve edge *e*. This path will have an edge *e'* with one vertex in component S and another vertex in component V-S.

Now consider the set of edges T' = T-{e} U {e'}. The graph (V, T') is connected and has no cycles, so it is a spanning tree of G. Since the weight of e > weight e', the weight of T' is less than the weight of T. Therefore T is not a minimum spanning tree.

Proof B (similar to proof A)

Assume e = (v, w) is in a minimum spanning tree T of G. *(We need to show that there is another spanning tree of G with smaller weight than T, thus leading to a contradiction.)*

Let's delete edge e from T. This partitions the vertices of G into two components S and V – S. Let node v be in S and node w be in V-S.

Since G has a cycle C and edge *e* is part of that cycle, there is a path from v to w in G that does not involve edge *e*. This path will have an edge *e'* with one vertex in component S and another vertex in component V-S.

Now consider the set of edges T' = T-{e} U {e'}. The graph (V, T') is connected and has no cycles, so it is a spanning tree of G. Since the weight of e > weight e', the weight of T' is less than the weight of T. Therefore T is not a minimum spanning tree. **This is a contradiction. This e cannot be in a minimum spanning tree of G**

More on proofs

- Once a property of fact is established, it can be used in later proofs
- Example: Suppose we have a situation where we have a cycle and an edge e in the cycle with the largest weight. We can safely claim that that edge e will not be in the MST of a graph, by the cycle property.
- Assume, without loss of generality, that all edge weights are distinct.
- Homework 4 – proof practice

Given a connected graph G, with distinct edge weights, let n be the number of vertices in G and m be the number of edges. A particular edge e = (v, w) of G is specified. Give an algorithm with running time O(m+n) to decide whether e is contained in a minimum spanning tree of G

1. Construct G’ from G by deleting edges with weight greater than the weight of e. Delete e as well.
2. See if there is a path from v to w in G’, then e is not included in any MST of G. Otherwise, e is included in the MST of G.