CSE 2320 Notes 13: Minimum Spanning Trees

(Last updated 11/1/06 7:50 PM)

CLRS, 23.1, 23.2 - omit Kruskal's algorithm

CONCEPTS

Given a weighted, connected, undirected graph, find a minimum (total) weight free tree connecting the vertices. (AKA bottleneck shortest path tree)

Observation: Suppose S and T partition V such that

1. $S \cap T = \varnothing$

2. $S \cup T = V$

3. $|S| > 0$ and $|T| > 0$

then there is some MST that includes a minimum weight edge {s, t} with $s \in S$ and $t \in T$.

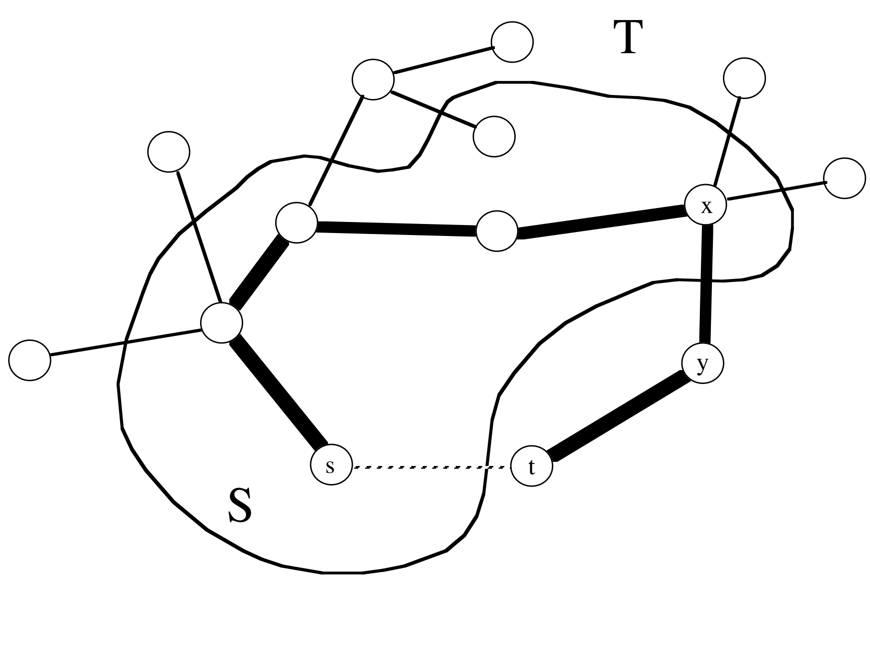
Proof:

Suppose there is a partition with a minimum weight edge {s, t}.

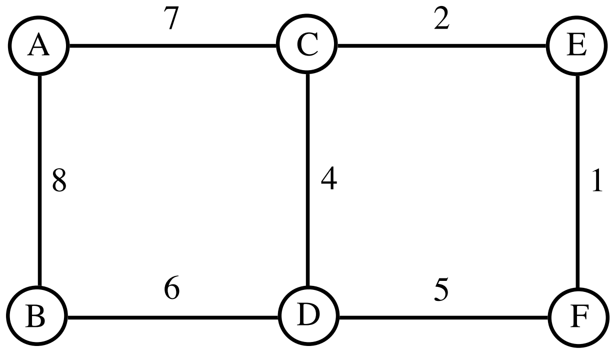
A spanning tree without {s, t} must still have a path between s and t.

Since $s \in S$ and $t \in T$, there must be at least one edge {x, y} on this path with $x \in S$ and $y \in T$.

By removing {x, y} and including {s, t}, a spanning tree whose total weight is no larger is obtained. •••



The proof suggests a slow approach - remove a maximum weight edge from every cycle:



Prim's algorithm applies the observation by having S as vertices connected together by a subtree of the eventual MST and T contains vertices that have not yet been connected. *The algorithm avoids including the maximum weight edges for all cycles.*

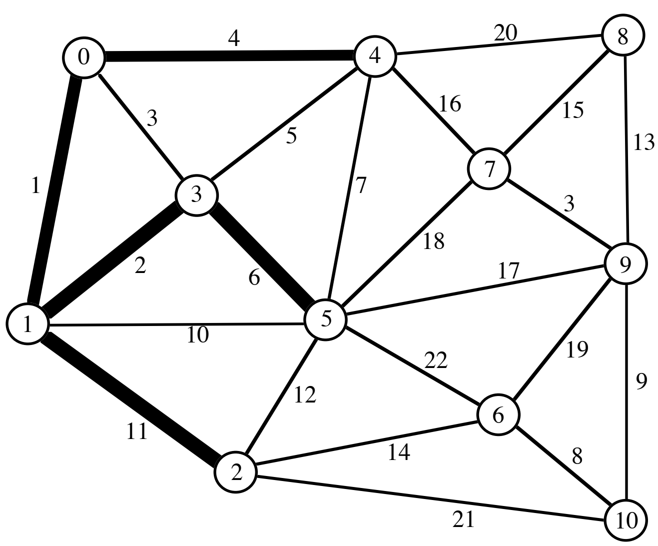
PRIM'S ALGORITHM – Three versions

1. "Memoryless" – Only saves partial MST and current partition. (`primMemoryless.c`)

Place any vertex $x \in V$ in S.
$T = V - \{x\}$
while $T \neq \emptyset$
    Find the minimum weight edge {s, t} over all $t \in T$ and all $s \in S$. (Scan adj. list for each t)
    Include {s, t} in MST.
    $T = T - \{t\}$
    $S = S \cup \{t\}$

Since no substantial data structures are used, this takes $\Theta(EV)$ time.

Which edge does Prim's algorithm select next?

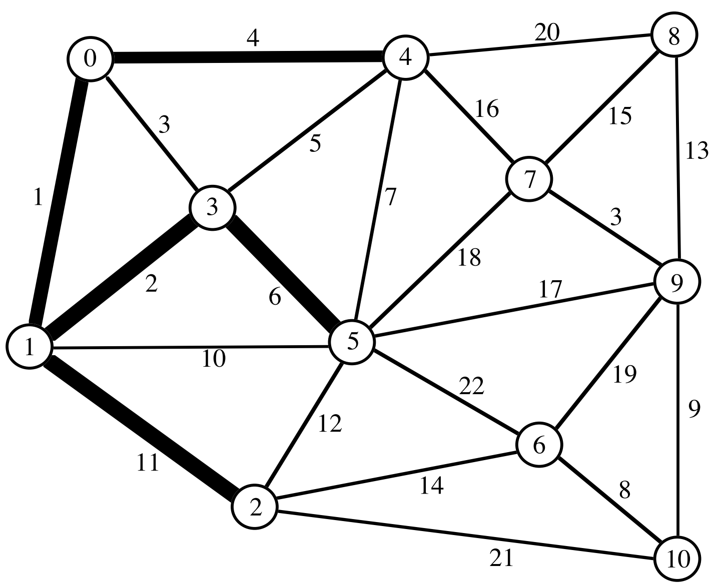


2. Maintains T-table that provides the closest vertex in S for each vertex in T. (`primTable.c`)

Eliminates scanning all T adjacency lists in every phase, but still scans the list of the last vertex moved from T to S.

Place any vertex x ∈ V in S.
T = V – {x}
for each t ∈ T
    Initialize T-table entry with weight of {t, x} (or ∞ if non-existent) and x as best-S-neighbor
while T ≠ ∅
    Scan T-table entries for the minimum weight edge {t, best-S-neighbor[t]}
        over all t ∈ T and all s ∈ S.
    Include edge {t, best-S-neighbor[t]} in MST.
    T = T – {t}
    S = S ∪ {t}
    for each vertex x in adjacency list of t
        if x ∈ T and weight of {x, t} is smaller than T-weight[x]
            T-weight[x] = weight of {x, t}
            best-S-neighbor[x] = t

What are the T-table contents before and after the next MST vertex is selected?



Analysis:

Initializing the T-table takes Θ(V).

Scans of T-table entries contribute $\Theta(V^2)$.

Traversals of adjacency lists contribute Θ(E).

$\Theta\left(V^2 + E\right)$ overall worst-case.

3. Replace T-table by a heap. (`primHeap.cpp`, `minHeap.cpp`)

The time for updating for best-S-neighbor increases, but the time for selection of the next vertex to move from T to S improves.

```
Place any vertex x ∈ V in S.
T = V – {x}
for each t ∈ T
    Load T-heap entry with weight (as the priority) of {t, x} (or ∞ if non-existent) and x as
        best-S-neighbor
BUILD-MIN-HEAP(T-heap)
while T ≠ ∅
    Use HEAP-EXTRACT-MIN to obtain T-heap entry with the minimum weight edge over all t ∈ T
        and all s ∈ S.
    Include edge {t, best-S-neighbor[t]} in MST.
    T = T – {t}
    S = S ∪ {t}
    for each vertex x in adjacency list of t
        if x ∈ T and weight of {x, t} is smaller than T-weight[x]
            T-weight[x] = weight of {x, t}
            best-S-neighbor[x] = t
            MIN-HEAP-DECREASE-KEY(T-heap)
```

Analysis:

Initializing the T-heap takes Θ(V).

Total cost for HEAP-EXTRACT-MINs is Θ(V log V).

Traversals of adjacency lists and MIN-HEAP-DECREASE-KEYs contribute Θ(E log V).

$\Theta(E \log V)$ overall worst-case, since E > V.

Which version is the fastest?

		Sparse $(E = O(V))$	Dense $(E = \Omega(V^2))$
1.	$\Theta(EV)$	$\Theta(V^2)$	$\Theta(V^3)$
2.	$\Theta(V^2 + E)$	$\Theta(V^2)$	$\Theta(V^2)$
3.	$\Theta(E \log V)$	$\Theta(V \log V)$	$\Theta(V^2 \log V)$

True or False (Prove or give counterexample):

Suppose an MST has been found for a graph.

Claim: For any pair of vertices, there is a *shortest path* that uses only MST edges.

True or False:

If no two edges in an undirected graph have the same weight, then the MST is unique.