Minimum Spanning Trees

CLRS Reading: Chapter 23, pages 624 -- 642

Wiring the Boston Area

CLRS Reading: Chapter 23, pages 624 -- 642
Minimum Spanning Tree Problem

**Given**
A connected graph $G = \langle V, E \rangle$ such that each edge $e \in E$ is assigned a non-negative weight.

**Problem**
Find a subset $T$ of $E$ such that $\langle V, T \rangle$ remains connected, and the sum of the weights of the edges is as small as possible subject to above.

*Is $\langle V, T \rangle$ always a tree (i.e., an acyclic, connected graph)?*

Adapting Greedy to the MST Problem

```plaintext
Greedy(C)
{C is the set of choice candidate}
S = ∅ {S the solution set}
while not solution(S) and C ≠ ∅
x = greedy-choice(C)
C = C \ {x}
if feasible(S ∪ {x})
    S = S ∪ {x}
return S
```
In Other Words

MST(G = <V,E>)

\[ C = E \] \{C is the set of choice candidate\}
\[ T = \emptyset \] \{T the solution set\}

while not <V,T> is connected and C ≠ \emptyset

\[ x = \text{an element of } C \text{ of smallest weight} \]
\[ C = C \setminus \{x\} \]

if <V,(T ∪ \{x\})> is acyclic

\[ T = T ∪ \{x\} \]

return T

---

Kruskal’s Algorithm

Kruskal(G, w)

sort the edges by increasing weight

for each edge e on the sorted list

if e does not form a cycle with the edges already taken

include e in the spanning tree

else discard e

return resulting edge set
**Kruskal in Action**

![Graph with weights and distances]

**Is It a Good Idea to be Greedy?**

**Given**

A connected graph \( G = \langle V, E \rangle \) such that each edge \( e \in E \) is assigned a non-negative weight.

**Problem**

Find a subset \( T \) of \( E \) such that the edges of \( T \) form a simple closed path that includes every node.
Crime Does Not Pay

*Things can be worse still. Is it possible that Greedy will fail to produce a tour at all? Can you give an example?*

Kruskal’s Algorithm

\[ \text{Kruskal}(G, w) \]

sort the edges by increasing weight

for each edge \( e \) on the sorted list

if \( e \) does not form a cycle with the edges already taken

include \( e \) in the spanning tree

else discard \( e \)

return resulting edge set

W - 10
Proof of Correctness*

Given
Let $G = <V, E>$ be a connected graph such that each edge is assigned a non-negative weight.

Problem
For all $n \leq |E|$, if Kruskal's algorithm is applied and a set $T$ of $n$ edges have been selected, then $T$ is promising (i.e., can be completed to an optimal solution).

*By induction on the number of edges selected so far.

Base Case

- For $n = 0$, a set $T$ of $n$ edges have been selected.
- Show $T$ is promising (i.e., can be completed to an optimal solution).
Induction Step

- Let $n < |V|-1$. Assume that Kruskal’s algorithm produces a promising set of edges $T$ for all $|T| < n$.

- Let $T$ be the collection of edges chosen to date.

*The induction hypothesis implies that $T$ forms a forest partitioning $V$ into disjoint sets.*

We Should Be So Lucky

- Let $e$ be an edge of minimal weight that doesn’t form a cycle with $T$.

- Let $T'$ be a minimal spanning tree of $G$ such that $T$ is contained in $T'$. If $e \in T'$, no problem.
Otherwise,

- Adding \( e \) to \( T \) creates exactly one cycle.

- Removing \( e' \) from \( T \) yields a new tree that spans \( G \) and has weight no greater than \( T \). Since \( e' \not\in T \), \( T \cup \{e\} \) is promising.
Prim’s Algorithm

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)