**Min Spanning Tree (MST).**

Given an undirected connected weighted graph $G = (V, E), W : E \rightarrow \mathbb{R}^+$,

Find a connected subgraph that includes all

the vertices with minimum total weight

**eg.**

![Graph Diagram]

One sol’n:

$5 + 15 + 7 + 2 = 55$

Better Sol’n:

$5 + 7 + 2 + 21 = 35$

Best.

**Obs1:** Opt sol’n is acyclic.

Because removing

“any” edge from C

leaves the graph connected

(Appln: Network Design)

A feasible/opt sol’n is

acyclic, undirected graph

Opt sol’n is a tree.

**Obs2:** The #edges in a

tree with $n$ vertices is $(n-1)$. 
Tree with \( n \) vertices is \((n-1)\).

Idea 0: Enumerate \(\Rightarrow\) Exponential time.

Idea 1: Greedy!

* Kruskal's Alg'rn (1956): High-level.

1. \( T = \emptyset \)
2. repeat \{ \}
3. Pick next smallest weight edge \( e \)
4. \( \rightarrow \) if \( T \cup \{ e \} \) does not contain a cycle then insert \( e \) to \( T \).

```
Snap shot:
```

Naive Running Time:
line \( t \) : \( O(n) \) (BFS/DFS)
Total : \( O(mn) \)

```
Eg.
```
Faster Implementation using "union-find" data structure.

1. Sort edges in increasing order of weights. $= O(m \log n)$
2. Create $E \cup V$, $\forall v \in V$
3. For each $(u, v)$ in the order
4. If $u \cup v$ are in different "sets" then
5. Output $(u, v)$. Union the two sets.

Snapshot:

Forest

Running Time:

Since $\log n$:

Since $n \leq V \Rightarrow O(n \log n)$
Running time:

- Union-find:
  - Find set containing given vertex \( v \). \( \Rightarrow O(\alpha(n)) \) amortized.
  - Union two sets. \( \Rightarrow O(1) \)

\[ \alpha(n) \ll \log \log \log \ldots \log n \]

- Lines 3-5: \( O(m \cdot \alpha(n)) \)
- Total: \( O(m \log n + m \cdot \alpha(n)) = O(m \log n) \).

\* Correctness Pf: (Assume all weights are distinct)

**Key Lemma:** Given \( S \subseteq V \) smallest weight edge \( e \in E \) \( S \cup \{v \} \subseteq S \)

**Pf:** By contradiction. Suppose \( e \notin T^* \)

\[ w(e') > w(e) \]

(exchange arg.)
\[ T = T^* \cup \{ e \} \backslash e_1 \] is a tree that spans all the vertices.

\[ w(T) = w(T^*) + w(e) - w(e_1) < w(T^*) \]

because \( w(e) < w(e_1) \).

A contradiction to \( T^* \) being MST. \( \Box \)

Correctness for Kruskal's Algorithm:

For each edge \((u,v)\) that is inserted to \( T \).

We have that \( e \) is a smallest weight crossing \( S = \text{component of} \ u \cup (V \setminus S) \).

Then by the key lemma \((u,v)\) must be in the MST. \( \blacksquare \)

-----

Prim's Alg'm (1957): High-level like Dijkstra's.
Prim's Alg'm (1957): High-level like Dijkstra's

1. \( S = \{s\} \), \( T = \emptyset \)
2. while \( S \neq V \) do
3.     Pick an edge \((u,v)\) s.t. \( u \in S \), \( v \in V \setminus S \) with \( \min \{ w(u,v) \} \).
4.     Insert \((u,v)\) to \( T \).
5.     Insert \( v \) to \( S \)

Running Time: like Dijkstra using Fibonacci heap \( O(n \log m + m) \)

Correctness Pf: Just by the key lemma!

If we add \((u,v)\) to \( T \), then it must be that \((u,v)\) is min weight crossing edge for the current set \( S \).

\[ \text{Eq.} \]

\[ \text{Diagram} \]

\[ \text{Diagram} \]

\[ \text{Diagram} \]
* Other Alg'm:

- Boruvka (1926) \( O(m \log n) \)
- K (1956) \( O(m \log n) \)
- P (1957) \( O(m \log n + m) \)
- Yao '75 (1975) \( O(m \log \log n) \)
- Fredman, Tarjan (1985) \( O(m \log^* n) \)
- Gabow et al. (1986) \( O(m \log (\log^* n)) \)
- Karger, Klein, Tarjan (1995) \( O(m) \) Randomized.
- Cazelle '97 \( O(m \log \log n) \) det.

**OPEN** \( O(m) \) det?
OPEN. \( O(m) \) au.