Recap: Minimum Spanning Tree (MST)

Graph \( G = (V,E) \) with edge weights \( w_e \)

Goal: Find a tree \( T \subseteq E \) connecting all the vertices with minimum total edge cost

Meta-algorithm

\[
X = \emptyset \\
\text{repeat until } |X| = |V| - 1 \\
pick a set \( S \subseteq V \) s.t. \( X \) has no edge from \( S \) to \( V \setminus S \) \\
let \( e \in E \) be the minimum-weight edge from \( S \) to \( V \setminus S \) \\
\( X = X \cup \{e\} \)
\]

Example: Kruskal's algorithm, Prim's algorithm
Prim's algorithm

X always forms a (connected) tree on a set of vertices S \& V
At each step, pick the lightest edge from S to V \& S

Similar to Dijkstra's. Implement w/ priority queue
$\text{prim}(G, w)$

$X = \emptyset$

$Q = \text{priority queue}(\cdot)$

for each $v \in V$, $Q.\text{insert}(v, \infty)$

from $[v] = \text{null}$

pick start vertex $s \in V$, $Q.\text{decrease\text{\_}\text{Key}}(s, 0)$

while $|X| \leq |V| - 1$

$u = \text{delete\_\_Min}(Q)$

if $u \neq s$, $X = X \cup \{v' \mid (u, v') \in E\}$

for all $v \in V$ s.t. $(u, v) \in E$

if $v \in Q$ still and $w(u, v) < v.\text{key}()$

$Q.\text{dec\_Key}(v, w(u, v))$

from $[v] = u$

---

**Runtime**

$n = |V|$, $m = |E|$

$n$ inserts

$n$ delete Mins

$O(m)$ decrease Keys

$O((m+n) \log n)$

w/ binary heap

$O(m+n \log n)$

w/ Fibonacci heap
MST runtimes

Kruskal: \( O((m+n) \log n) \) time

Prim: \( O(m + n \log n) \)

Karger, Klein, Tarjan 1995: \( O(m+n) \) expected time (randomized)

Chazelle 2000: Deterministic \( O(m \cdot A(m,n)) \)

\( \uparrow \) inverse Ackermann function \( A(m,n) \leq 5 \) for \( m,n \) reasonable

Pettie, Ramachandran 2002: Deterministic \( O(\text{optimal}) \)

optimal = ?
Rest of lecture: disjoint set/union find data structure

implementation: disjoint-set forest

- **make set (x)**
  (creates a singleton set containing x)  \( \Omega(1) \) time

- **find (x)**
  (which set is \( x \) in?)  \( \Omega(\log n) \)

- **union (x, j)**
  (merges the sets containing \( x, j \))  \( \Omega(\log n) \)

Kruskal's algorithm

- \( n \) make sets
- \( 2m \) finds
- \( n-1 \) unions
- \( \Omega(n \log n) \)

\( 0(n) \)
\( 0(m \log n) \)
\( 0(n \log n) \)
\[ + 0(m \log n) = \Omega(m \log n) \]
\[ \Omega((m+n) \log n) \]

(We will also see how to improve union find)
Union find using directed trees

\[ \{ B, E_3 \} \rightarrow \{ A, C, D, F, G, H_3 \} \]

representative/name of set

\[ \pi(x) = x \quad (\pi(x) = x's \ parent) \]

\[ \text{rank}(x) = 0 \]

\[ \text{makeSet}(x) \]

\[ \text{find}(x) \]

\[ \text{while } x \neq \pi(x) \]

\[ x = \pi(x) \]

\[ \text{return } x \]
union \((x, y)\) "union by rank"

\[
\begin{align*}
    r_x &= \text{find}(x) \\
    r_y &= \text{find}(y)
\end{align*}
\]

if \(r_x = r_y\), return

if \(\text{rank}(r_x) > \text{rank}(r_y)\)

\[
\pi(r_y) = r_x
\]

else if \(\text{rank}(r_x) < \text{rank}(r_y)\)

\[
\pi(r_x) = r_y
\]

else

\[
\pi(r_x) = r_y \\
\text{rank}(r_y) = \text{rank}(r_y) + 1
\]

\[
\text{runtime} = O(\text{runtime of find})
\]
**Claim 1**: For all \( x \), \( \text{rank}(x) \leq \text{rank}(\Pi(x)) \) (unless \( x = \Pi(x) \)).

**Claim 2**: Any root node of rank \( k \) has \( \geq 2^k \) nodes in its tree.

**Pf**: By induction on \( k \). Base case \( k = 0 \) ✓

Inductive step. Assume true for rank \( K \).

Suppose union \((x, y)\)
Claim 1: For all $x$, $\text{rank}(x) < \text{rank}(\pi(x))$ (unless $x = \pi(x)$)

Claim 2: Any root node of rank $k$ has $\geq 2^k$ nodes in its tree
(Also holds for non-root nodes)

Claim 3: If $n$ elements overall, at most $\frac{n}{2^k}$ elements of rank $k$.

**Pf:** Rank $k$ nodes have disjoint subtrees.

- $(\# \text{ of rank } k \text{ nodes}) \cdot 2^k \leq n$

**Corollary:** All nodes have rank $\leq \log(n)$

Hence, find and union take time $O(\log(n))$
Improving union find via path compression

\[
\text{find}(x) \quad \begin{cases} 
\text{while } x \neq \pi(x) \\
\quad x = \pi(x) \\
\quad \text{return } x 
\end{cases}
\]

\[
\text{find}(x) \quad \begin{cases} 
\text{if } x = \pi(x), \text{return } \pi(x) \\
\quad \text{else, } \text{return find}(\pi(x)) \\
\quad \text{return } \pi(x) 
\end{cases}
\]

\[
\text{don't update rank} \\
\text{rank no longer height}
\]
Runtime of union find with path compression

A sequence of $n$ makesets and $m$ union/finds take

$O(n + m \cdot \alpha(m, n))$ time

We will show: $O((ntm) \cdot \log^* n)$ time

- average operation takes time $\log^* n$

$\log^*(n) = \# \text{log}_2(n)$'s needed to bring $n$ to $\leq 1$

$\log^*(2) = 1$ \hspace{1cm} $\log^*(2^2) = 2$ \hspace{1cm} $\log^*(2^{2^2}) = 5$

Note: does not say each operation takes $\log^*(n)$ time
Consider intervals $[0], [1], (1,2], (2,4], (4,16], (16,65536], \ldots$

For $i \geq 1$, interval $i$ is $(2^{i+1}, 2^{i+2}]$, $2^i = 2^i$. 

Element $x$ is in interval $i$ at some time if:
- $x$ is not a root
- $x$'s rank $(x)$ is in interval $i$

Largest possible rank is $\log(n)$. It is in interval $i$ where $2^i < \log(n) \leq 2^{i+1}$.

So $i = \log^*(\log(n)) = \log^*(n) - 1$.

Let $k = 2^{i-1}$. Number of elements in $(k, 2^k]$ (interval $i$)

$$\leq \frac{n}{2^{ki}} + \frac{n}{2^{k+1}} + \ldots \leq \frac{n}{2^k}$$