Review: Minimum Spanning Trees (MSTs)

Undirected graph:

Want: Tree connects all vertices, has minimum cost

2 Algorithms: Prim and Kruskal

1. Prim: Pick any node in the graph to be your starting tree $T$.
   While all nodes not connected:
   Choose edge $e$ from $T \rightarrow G \setminus T$ with min cost and add it to $T$.

\[
\begin{align*}
A, B, F, C, D, E \\
A-B, A-F, B-C, C-D, D-E
\end{align*}
\]

\[
\text{cost}(T) = 10
\]

2. Kruskal: Start with forest of all (unconnected) nodes.
   While all nodes not connected:
   Choose edge $e$ with min cost that connects two components of the forest.
A-B, C-D, A-E, B-C, D-E

\( \nabla \text{ Minimum Spanning Tree does not need to be unique!} \)

\[
\begin{align*}
\text{MSTs:} & \quad 1 & & 2 \\
1 & & 2 & & 1 \\
2 & & 1 & & 1
\end{align*}
\]

**Problem:** Does the following algorithm work: start with entire graph \( G \) and while \( 3 \) cycles, remove edge with max. cost that does not disconnect graph.

Yes, it does work.

Let \( e \) be edge of max. cost that does not disconnect graph.

\( \Rightarrow \) \text{ 3 MST without } e. \\

Why? \( e \) must be in a cycle (otherwise it would disconnect \( G \)) and it must be heaviest of all edges in all cycles that go through it (because none of these would disconnect graph if removed).
A-C: In a cycle A-C-B, A-C-F, A-C-E-F, etc. Also, heaviest among all edges in these cycles.

Claim: Kruskal's tree does not contain e. Why?

Kruskal adds edge with min cost that does not add cycle. ⇒ Kruskal will select any of other edges in all cycles through e before selecting e (they are valid options because cycle is not complete without e).

At every step of our algorithm, remove e:
\[ G \rightarrow G' = G - \{e\} \]

Kruskal produces same MST in G and G'.

When there are no cycles, MST is just that tree.

⇒ our algorithm gives valid MST.

Review: Hashing

Have N elements with large values: \{0, ..., M-1\}
Want to represent them with smaller values.

in \{0, ..., M-1\}, \ M < W

⇒ There will be collisions, want to minimize these.
Hash function: \( h_{\text{key}}(x) = x \)

\[ \text{in } \{0, \ldots, W-1\} \quad \text{in } \{0, \ldots, M-1\} \]

\( H = \{ h_{\text{key}} \} \) \hspace{1cm} \text{collection of hash functions}

(\( h \) with different keys)

**Definition:** It is universal if \( \forall x \neq y \)

\[ \Pr_{\text{key}} \left[ h_{\text{key}}(x) = h_{\text{key}}(y) \right] \leq \frac{1}{M}. \]

\[ \uparrow \quad \text{Collisions are "unlikely".} \]

**Example:** \( W = 2^w \quad M = 2^m \)

\( \text{key} = (a,b) \) \hspace{1cm} \text{where } a,b \in \{0, \ldots, W-1\}

(chosen at random)

\[ h_{a,b}(x) = \left\lfloor \frac{ax + b \pmod{2^w}}{2^{w-m}} \right\rfloor \]

\( w \) bits

\( m \) bits

"Best" method to hash integers \hspace{1cm} \text{(result from 1997)}

- efficient
- easy to implement \hspace{1cm} \text{(multiply & shift)\)
Thm: when \((a, b)\) are random, \(H = \{h_{ab}\}\) is universal.

Problem: S collection of \(N = 2^k\) not necessarily distinct items in \(\{0, \ldots, M-1\}\), with \(M = 2^k\). Give a randomized algorithm for determining if all elements in \(S\) are distinct. Want expected running time \(\Theta(n)\).

Create array \(A[0:n-1]\) use hash function \(h_{ab}(x) = \left[ax+b \pmod{2^k}\right] / 2^{k-k}\) (random \(a, b\)).

Output has \(k\) bits, so value \(\in \{0, \ldots, n-1\}\).

For each item \(x\), compute \(h_{ab}(x)\) and check if \(h_{ab}(x)\) is in \(A[h_{ab}(x)]\). If yes, report "NOT DISTINCT". Otherwise, add \(x\) to \(A[h_{ab}(x)]\).

Suppose e.g. each "bucket" in \(A\) has a linked list.

If at end, haven't reported anything, report "DISTINCT".

Running Time: For each \(x\):

\[O(1 + \text{\# elements in } A[h_{ab}(x)] \text{ that are } x)\]

depends on \(a, b\). What is expected value?
\[
\text{(\# Items in } A[h_{ab}(x)] \text{ that are } \neq x) \\
= \sum_{y \neq x} \mathbb{1}[h_{ab}(x) = h_{ab}(y)]
\]

**Notation:** counts as 1 if this holds or 0 if not.

\[
\text{(\# expected Items in } A[h_{ab}(x)] \text{ that are } \neq x) \\
= \sum_{y \neq x} \Pr[h_{ab}(x) = h_{ab}(y)]
\]

(by universality)

\[
= \sum_{y \neq x} \frac{1}{n} \leq \frac{n}{n} = 1.
\]

For each x: time = \(\Theta(1)\)

For all x: time = \(\Theta(n)\).