# CS561 HW10

Ryan Scherbarth

November 2024

1. **Prove the following fact, from Slide 114, amortized lecture, about the Union-Find data structure using PC-UNION:**

   - **For any set $X$, size$(X) \geq 2^{\textbf{rank(leader}(X)}$**

   **Prove this by induction on the size of $X$. Don't forget to include the BC, IH, and IS.**

   **Base Case:**
   If $x = 0$, then $S$ is a single element. For any set $S$ with $rank(leader(S)) = 0$, then $size(S) = 2^0 = 1$, so the base case holds true.

   **Inductive Hypothesis:**
   Assume that for any $J$ s..t $size(J) < x$, it holds that $size(J) \geq 2^{rank(leader(J))}$.

   **Inductive Step:**
   There are two cases that can occur when performing a UNION on two sets, $S_1$ and $S_2$, each satisfying the inductive hypothesis;

   **Case 1:** $size(S_1) = size(S_2)$
   If $rank(leader(S_1)) = rank(leader(S_2))$, the union of $S_1$ and $S_2$ will increase the rank by exactly 1.

   We will now have $rank(leader(S)) = rank(leader(S_1)) + 1$,
   therefore $size(S) \geq 2^{rank(leader(S_1))+2^{rank(leader(S_2))}} \geq 2^{rank(leader(S))}$.

   We can then confirm the case holds that $size(S) \geq 2^{rank(leader(S))}$.

   **Case 2:** $size(S_1) \neq size(S_2)$
   Let $x_1 = rank(leader(S_1))$ and $x_2 = rank(leader(S_2))$ and lets define them s.t. $S_1 \geq S_2$.

   When performing a union on two equal height sets, the rank will remain the same. Since $x_1 > x_2$, we can confirm that the result will be equal to the size of $x_1$.

   Therefore $size(S) = size(S_1) \geq 2^{x_1}$ and $x_1 = rank(leader(S_1))$.

2. **Professor Moe conjectures that for any connected graph $G$, the set of edges** $\{(u, v) : \text{there exists a cut } (S, V-S) \text{ such that } (u, v) \text{ is a light edge crossing}(S, V-S)\}$ **always forms a minimum spanning tree. Give a simple example of a connected graph that proves him wrong.**

   Say we have the graph $V$ s.t. we have 4 vertexes; $A, B, C, D$, and 4 edges; $\{(A, B), (B, C), (C, D), (D, A), (B, D)\}$ with corresponding weights as $1, 2, 3$, and 4. Our graph is in the form of a straight line containing the 4 vertexes, each of which are connected and then we have a final connection looping around between $A$ and $D$, and an edge between $B$ and $D$ with weight 1.

   In this case, $\{(u, v) : \text{there exists a cut } (S, V-S) \text{ such that } (u, v) \text{ is a light edge crossing}(S, V-S)\}$ results in the set $\{(A, B), (B, C), (C, D), (B, D)\}$. We define the minimum spanning tree $M$ as a subset of some larger graph, $T$, s.t. $M$ includes all vertices of $T$, and forms a connected, acyclic sub-graph with the minimal value to the weight function, $w(T) = \sum_{e \in E_T} w(e)$.

   Given our definition we can trivially prove that the sub-graph generated from Professor Moe's conjecture does not satisfy the criteria for a minimum spanning tree. With this example, our graph returned does meet the first criteria of a minimum spanning tree by minimizing the weight function, $w$, however it contains a cycle between $A, C, D$. This fails the first condition of a MST in that it is not an acyclic sub-graph of $T$, and therefore disproves Professor Moe's conjecture by counter example.

3. **Consider a connected graph $G = (V, E)$. Call a subset of edges, $F$, a *cycle cover* if every cycle in $G$ contains at least one edge in $F$. In other words, removing the edges of $F$ from $G$ results in an acyclic graph. You want to find a cycle cover, $F$ of $G$, with minimum weight, i.e., the sum of the weight of all edges in $F$ is minimized over all cycle covers. Give an efficient algorithm to solve this, and give the runtime of your algorithm as a function of $n = |V|$, and $m = |E|$. Hint: Think about the maximum-weight spanning tree problem.**

   Our algorithm will start my finding the $MST$ of $G$ using Kruskal's algorithm. After we've created the $MST$, the set of nodes that is not included in this $MST$ will give us our cycle cover for $F$. We can get this by taking the set difference $G - MST$.

   When we form the $MST$ we are selecting the maximum weight edges at each iteration. A side-effect of this is that the remaining edges satisfy the requirements as defined in the problem as a cycle cover, having the minimum total weight. Removing any edge from the $MST$ also creates a cycle, so we can conclude that the remaining edges cover all cycles in the graph.

   The runtime of this algorithm is determined by how long it takes to find the $MST$ of the set of nodes. To run Kruskal's algorithm we will spend $O(E \log(E))$ time to sort the edges, which will ultimately bound our runtime to $O(m \log(m))$.

4. **Professor Matsumoto conjectures the following converse of the safe edge theorem: Let $G = (V, E)$ be a connected, undirected, weighted graph, with weight function $w$. Let $A$ be a subset of $E$ that is included in some minimum spanning tree of $G$. Let $(S, V - S)$ be any cut of $G$ that respects $A$, and let $(u, v)$ be a safe edge for $A$ that crosses $(S, V - S)$. Then $(u, v)$ is a light edge for the cut. Is this conjecture true? If so, prove it. If not, give a counterexample.**

   Let us define a graph, $G$, consisting of vertexes; $V = \{a, b, c\}$, and edges $E = \{(a, b), (b, c), (c, a)\}$ with corresponding weights 1, 2, and 3.

   Creating the MST for this graph will use the edges $(a, b)$ and $(b, c)$ with a total weight of 3. If we let $A = \{(a, b)\}$ which is a part of our MST, we can then make a cut $(S, V - S) = (\{a\}, \{b, c\})$, which respects $A$ since $(a, b) \in A$ crosses this cut.

   In this case, our safe edge becomes $(b, c)$. This is the only possible safe edge that crosses $(S, V - S)$. We see this to be a counter-example as the edge $(b, c)$ is not the lightest edge crossing the cut, the lightest is actually edge $(a, b)$. Therefore, this conjecture is not valid.

5. **Prove that if an edge $(u, v)$ is in some minimum spanning tree for a graph, $G$, then $(u, v)$ is a light edge crossing some cut in $G$.**

   Let $T$ be a MST containing the edge $(u, v)$. We define a cut $(S, V - S)$ s.t. $u \in S$ and $v \in V - S$ separating $u$ and $v$.

   The safe edge theorem tells us that an edge $e$ is safe for set $A$, where $A$ is a subset of edges forming an MST, if it is a light edge crossing a cut that respects $A$. Since $T$ is an MST, every edge in $T$ must be a safe edge for some set of edges $A$ with respected to $T$.

   Since the cut $(S, V - S)$ is defined to have separated $u$ and $v$, then edge $(u, v)$ must be the lightest edge across this cut that does not violate any of the rules of a MST.

   Therefore, using the safe edge theorem we can prove that any edge in a MST satisfies the property of being a light edge for some cut in $G$.