Exam 2: Tuesday, April 6
(logistics will be posted on Piazza next week)

Emails

Letters

Strongly Connected Components (SCC)

**Input:** Directed graph $G = (V, E)$.

**Output:** All SCCs of $G$.

**Algorithm**

1. Do DFS$(G)$

2. Compute $G^T$

3. Do DFS$(G^T)$, but in the main loop of DFS, we process vertices in the $\downarrow f[i].$ of $f[i]$. in the DFS stack.
4. Vertices in each tree are vertices in the SCCs of G.

Running time: \( O(n + m) \)

We can also construct \( G^{\text{scc}} \) in linear time.

**Correctness**

For any set \( S \subseteq V \),

\[
\delta(S) = \min_{u \in S} \{ \delta(u) \}, \quad \phi(S) = \max_{u \in S} \{ \phi(u) \}
\]

**Lemma**: Consider SCCs \( C \) and \( C' \) and let there be an edge from \( C \) to \( C' \) in \( G^{\text{scc}} \). Then \( \phi(C) > \phi(C') \).

**Proof**: 

![Diagram](attachment:image.png)
Can I: \( d(C) < d(C') \)

That is, some vertex, say \( u \in C \) is discovered before all other vertices \( v \in C' \).

\( v \neq u \)

Let \( v \in C \) (and \( v' \in C' \)) be any arbitrary but particular vertex. At \( d[u] \),

there is a viable path from \( u \) to \( v \)

in \( G \) and a viable path from \( u \) to \( v' \) in \( G \).

Thus \( v \) and \( v' \) are descendants of \( u \) in the DFS forest & hence we have

\[ d[u] < d[v] < f[v] < f[u] \]

\[ d[u] < d[v'] < f[v'] < f[u] \]

Can II: \( d(c') < d(c) \) . \( \xrightarrow{v} u' \rightarrow C' \)
let \( v' \in C' \) be a vertex that is discovered before all other vertices in \( C \cup C' \). Let \( v' \in C' \), \( v' \neq u \) and \( v \in C \) be any two vertices. At time \( d[v'] \) there is a path from \( u \) to \( v' \) in \( G \) and there is no path from \( u \) to \( v \) in \( G \). Thus \( v' \) is a descendant of \( v \) in the DFS forest and \( v \) is not a descendant of \( u \) in the DFS forest. Thus \( v' \) finishes and then \( v \) is discovered & hence 
\[ f[C] \neq f[C']. \]

The above lemma shows why the
vertex with the largest \( f[] \) belongs to
the source vertex in \( G_{sc} \).

\[ f[] > f[] \]

_Theorem_: Our algorithm works.

**Induction on \# trees.**

**IH:** let \( k > 0 \) be an integer. Assume that
the vertices in the first \( k \) DFS trees form
valid SCCs.
BC: We want to show that the first tree is "good". Let \( u_i \) be the root of the first tree. Let \( C_i \) be the SCC in \( G \) to which \( u_i \) belongs to.

Clearly, in Step 3 of the alg. all vertices in \( C_i \) will be reachable from \( u_i \). Hence will be in the tree rooted at \( u_i \). How do we guarantee that no other vertex from some other SCC is also in the tree rooted at \( u_i \) ? This is because after the edges are reversed, \( C_i \) becomes the sink in the \((A^T)^{SCC}\).
IS: Consider the (k+1)th tree. Let \( u_{k+1} \) be the root of the tree.

By IH each of the first k trees are "good". Thus all vertices of \( C_{k+1} \), the SCC that contains \( u_{k+1} \) are not discovered at \( J(u_{k+1}) \). Thus there is a WP from \( u_{k+1} \) to all vertices in \( C_{k+1} \) & hence they become descendants of \( u_{k+1} \). Note that now \( f \) the vertices from
CUk2U...Uk2 will be in the tree rooted at Uk1. What about the vertices to the "right"? How do we know that vertices from the "right" will not be in the tree rooted at Uk1?

\[
\text{largest } f[i] \quad \text{ and } \quad \text{dict}_L
\]

\[
\text{in}(G_T) \quad \text{SCC} \quad T_{k+1} \quad \text{dict}_L
\]

\[
C_1 \quad \text{Uk} = \text{1}
\]

\[
C_{k+2} \quad \text{C}_{k+2}
\]
In the graph induced by \( G_{\text{sink}} \) in \( G \), the vertices \( C_{kt+1}, C_{kt+2}, \ldots, h \) are the sources. Thus, when the edges are reversed, \( C_{kt+1} \) becomes the sink.

Minimum Spanning Trees.

Input: Undirected, connected graph \( G = (V, E) \) with non-negative edges.

Objective: To find a connected spanning tree.
Subgraph $T$ of $G$ of minimum total weight.

Obs: Note that $O/p$ must be a tree.

We will assume that all edges are distinct.

Dijkstra.
Kruskal's alg.

- Choose the smallest wt. edge & keep going that unless it forms a cycle. If so, do not include the edge in the set.
Wrong Dijkstra's alg. (Prim's alg).

Lemma: \( SC \), \( S \neq \emptyset \).

Let \( e = (u, v) \) be the edge with the smallest \( w(e) \) that crosses the cut \( (S, V \setminus S) \). Then \( e \) belongs to every MST.
Proof: Assume for contradiction that there is a MST $T$ that does not contain $e$. Since $u$ and $v$ are on different sides of the cut, there must be an edge in the MST, say $f$ that must have one endpoint in $S$ and the other in $V_S$. 

[Diagram of a graph with labeled vertices and edges, showing a cut with one edge marked for contradiction.]
Consider $T' = T - f + e$.

Since we cut $f$, $T'$ has a smaller total cut than $T$, a contradiction.

The above proof is bogus. $f$ needs to be chosen carefully. Choose $f$ to be the edge on the unweighted path $T$ that crosses the cut. Such an edge must exist as $\lambda$ & $\eta$ are in different partitions. All other arguments in the above proof hold & note that $T'$ is a tree.