Review Assignment
Graph Algorithms

Yessengaliyeva Gulnazym

Problem 1 — Strongly Connected Components

Let $G = (V, E)$ be a directed graph.

Problem 1.1 — Reverse a directed graph in $O(V + E)$

Algorithm. To construct the reversed graph $\text{rev}(G)$, we create a new graph with the same set of vertices. For every directed edge $(u, v) \in E$, we add the reversed edge (v, u) to $\text{rev}(G)$.

Correctness. Each edge of the original graph is processed exactly once and its direction is reversed. No other modifications are performed, so the resulting graph is exactly $\text{rev}(G)$.

Time complexity. All vertices are initialized in $O(V)$ time and all edges are processed in $O(E)$ time. Thus, the total running time is:

$$O(V + E).$$

Problem 1.2 — Prove that $\text{scc}(G)$ is acyclic

Let $\text{scc}(G)$ be the condensation graph of G where each vertex represents a strongly connected component (SCC).

Claim. The graph $\text{scc}(G)$ is acyclic.

Proof. Assume, for contradiction, that $\text{scc}(G)$ contains a directed cycle:

$$C_1 \to C_2 \to \cdots \to C_k \to C_1.$$

Then, from every vertex in C_i there exists a path to every vertex in C_{i+1}, and from C_k back to C_1. Hence, all vertices in

$$C_1 \cup C_2 \cup \cdots \cup C_k$$

are mutually reachable. This means they belong to a single strongly connected component, contradicting the definition of SCCs.

Therefore, $\text{scc}(G)$ must be acyclic. □

Problem 1.3 — Prove that $\mathrm{scc}(\mathrm{rev}(G)) = \mathrm{rev}(\mathrm{scc}(G))$

Claim. Reversing the edges of a graph does not change its SCCs, and the condensation graph is reversed.

Proof. If vertices u and v are in the same SCC of G, then there exist paths

$$u \rightsquigarrow v \qquad \text{and} \qquad v \rightsquigarrow u.$$

After reversing all edges, these paths remain valid in opposite directions. Thus, u and v are still mutually reachable in $\mathrm{rev}(G)$.

Therefore, SCC partitions of G and $\mathrm{rev}(G)$ are identical. If a component C_1 has an edge to C_2 in $\mathrm{scc}(G)$, this edge is reversed in $\mathrm{scc}(\mathrm{rev}(G))$. Hence,

$$\mathrm{scc}(\mathrm{rev}(G)) = \mathrm{rev}(\mathrm{scc}(G)). \quad \square$$

Problem 1.4 — Reachability and SCC graph

Let $S(v)$ denote the SCC containing vertex v.

Claim. For any vertices $u, v \in V$,

$$u \rightsquigarrow v \text{ in } G \iff S(u) \rightsquigarrow S(v) \text{ in } \mathrm{scc}(G).$$

Proof.

($\Rightarrow$) Given a path from u to v in G, movement inside SCCs is unrestricted and transitions between SCCs induce edges in $\mathrm{scc}(G)$. Hence, $S(u)$ reaches $S(v)$.

($\Leftarrow$) If $S(u)$ reaches $S(v)$ in $\mathrm{scc}(G)$, for each edge between SCCs there exists a corresponding edge in G. Combining these with paths inside SCCs yields a path from u to v in G.

Thus, reachability is preserved. $\square$

Problem 2 — Euler Tour

Let $G = (V, E)$ be a strongly connected directed graph.

Problem 2.1 — Degree condition for Euler tour

Theorem. G has an Euler tour if and only if

$$\forall v \in V : \quad \mathrm{in}(v) = \mathrm{out}(v).$$

Proof.

($\Rightarrow$) In an Euler tour, every time the tour enters a vertex it must also leave it. Therefore, the number of incoming edges equals the number of outgoing edges.

($\Leftarrow$) If the graph is strongly connected and every vertex has equal in-degree and out-degree, it is always possible to traverse edges without getting stuck. Thus, an Euler tour exists. $\square$

Problem 2.2 — $O(E)$ algorithm (Hierholzer)

Algorithm.

1. Start at any vertex and follow unused edges until returning to the start.
2. This forms a cycle.
3. If unused edges remain, start another cycle from a vertex on the current cycle.
4. Merge the cycles.

Correctness. Each edge is used exactly once, and degree balance guarantees closure.
Complexity. Each edge is processed once, giving $O(E)$ time.

Problem 3 — Topological Sort

Vertices:

$$\{A, B, C, D, E, F, G\}$$

Edges:

$$\{A \to B, A \to C, B \to C, B \to D, C \to E, D \to E, D \to F, G \to F, G \to E\}$$

Topological ordering starting from A

$$\boxed{A \to B \to C \to D \to G \to E \to F}$$

Topological ordering starting from G

$$\boxed{G \to A \to B \to C \to D \to E \to F}$$

Both orderings satisfy all precedence constraints, illustrating that topological orders are not unique.