Problem 1. Computing shortest paths

Let G be an undirected graph with N nodes $\{s_1, \dots, s_N\}$. Suppose G is complete, i.e., every pair of nodes is connected by an edge. For $i \neq j$, let $a_{ij} \in \mathbb{R}_{>0}$ be the cost of the edge connecting s_i and s_j. For $i = j$, we set $a_{ij} = 0$. By interpreting the nodes as states and the edges as actions, we may directly apply the dynamic programming method seen in lecture to compute shortest paths in G.

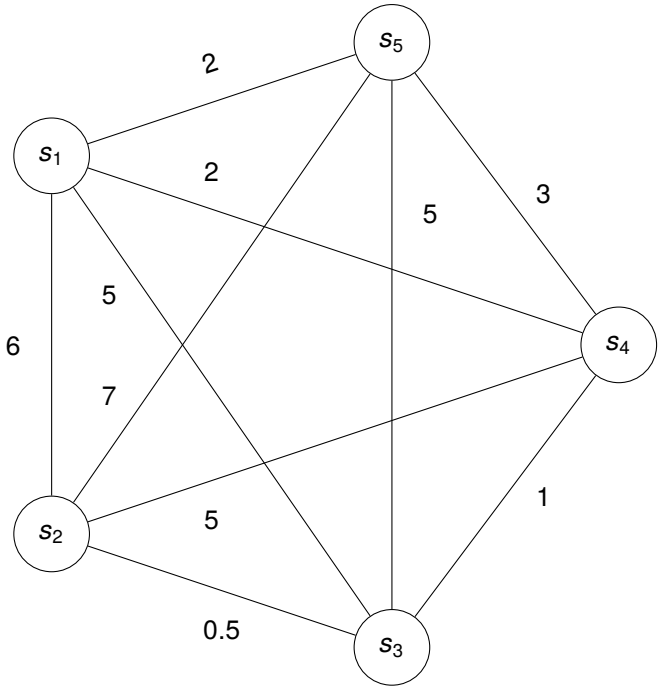

Figure 1: The graph G.

a) For the graph in Figure 1, use dynamic programming to compute the cost of the shortest path from each s_i, $i \in \{1, \dots, 4\}$, to s_5.

Hint: For each $k \in \{1, \dots, N\}$, and node s_i, let $V_k(s_i)$ be the optimal cost, i.e. shortest path, to s_N when starting from s_i with $N - k$ steps remaining. Clearly, $V_N(s_N) = 0$ and $V_N(s_j) = \infty$ for all $j \neq N$. Based on this, compute $V_{N-1}(s_i)$ for all s_i, then $V_{N-2}(s_i)$, and so on, until $V_1(s_i)$.

b) Use the computed values to reconstruct the shortest paths (i.e. the sequence of nodes that gives optimal cost) from each s_i, $i \in \{1, \dots, 4\}$, to s_5.

c) A naive way to compute shortest paths is to enumerate and compare all paths with at most $N - 1$ steps from a given node s_i, $i \in \{1, \dots, N-1\}$, to s_N. How many such paths[1] are there for each s_i in a complete graph with N vertices? How does the number of computations compare (asymptotically) to the dynamics programming method above?

[1] To simplify counting, you may count e.g. $s_1 s_1 s_1 s_5 s_5$ and $s_1 s_5 s_5 s_5 s_5$ as distinct paths.

Problem 2. Escape game

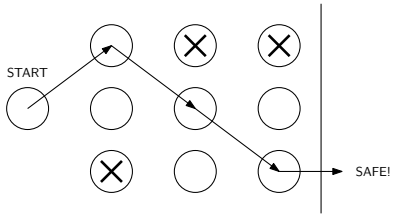

Figure 2: The escape game.

Player 1 (Alice) is trying to escape, going from the **start node** to the **safe zone** without being intercepted by Player 2 (Eve) who is trying to stop her. We model this as a dynamic game with stages $k = 1, \dots, K$ and states $\mathcal{S} = \{s_U, s_M, s_D, s_E\}$. States s_U,s_M,s_D represent Alice's current position (up, middle, or down), and state s_E is entered (and never left) once Eve has caught her.

In Figure 2, each row corresponds to a stage of the game. At each stage, Alice can decide to continue on the same row, or instead move diagonally one row up or one row down. Eve is aware of the current state, i.e. Alice's position, and she is allowed to block one of the three rows in the next stage, taking her decision simultaneously with Alice. If she selects the row corresponding to Alice's next move, the game enters state s_E which will result in cost 1 for Alice at the final stage. Otherwise, if Alice survives, i.e. the game is not in state s_E at the final stage, Alice gets cost -1. Eve's costs are the costs of Alice multiplied by -1.

The actions available to Alice depend on the current state s, namely,

$$\mathcal{U}_s = \begin{cases} \{U, M, D\}, & \text{if } s \in \{s_M, s_E\}; \\ \{U, M\}, & \text{if } s = s_U; \\ \{M, D\}, & \text{if } s = s_D. \end{cases}$$

For Eve, the action set is state-independent and given by $\mathcal{V} = \{U, M, D\}$.

a) Based on the game description, define its evolution map $f(s, u, v)$ for each s and $u \in \mathcal{U}_s$, $v \in \mathcal{V}$.

b) Additionally, we define a stage cost function $g_k(s, u, v)$ representing the cost for Alice, by setting, for any $u \in \mathcal{U}_s$, $v \in \mathcal{V}$,

$$g_k(s, u, v) = \begin{cases} 1, & \text{if } k = K \text{ and } s = s_E; \\ -1, & \text{if } k = K \text{ and } s \neq s_E; \\ 0, & \text{otherwise.} \end{cases}$$

Determine $V_K(s)$ for all $s \in \mathcal{S}$.

c) Observe that for $k \in \{1, \dots, K-1\}$ and $s \in \mathcal{S}$, we have $V_k(s) = \min_{u \in \Delta(\mathcal{U}_s)} \max_{v \in \Delta(\mathcal{V})} V_{k+1}(f(s, u, v))$. Using dynamic programming, for all $s \in \mathcal{S}$, determine the values of $V_{K-1}(s)$, and then $V_{K-2}(s)$.

Hint: For each $s \in \mathcal{S}$, observe that any $u \in \mathcal{U}_s$, $v \in \mathcal{V}$ uniquely determine the next state through the evolution map. Hence, given s, you can write down the matrix representation of the respective normal-form game and determine the value of its Nash equilibrium, i.e., a saddle point of $\min_{u \in \Delta(\mathcal{U}_s)} \max_{v \in \Delta(\mathcal{V})} V_{k+1}(f(s, u, v))$.

d) As $K \to \infty$, if the backward iteration converges, we would have $V(s) = \min_{u \in \Delta(\mathcal{U}_s)} \max_{v \in \Delta(\mathcal{V})} V(f(s, u, v))$ and we would obtain stationary policies $u^\star(s)$, $v^\star(s)$. Give a set of equations $V(s)$ would have to satisfy for such stationary policies.