Assignment 6/7: Sample Solutions

1. (a) By definition $a_{i,j}^{[t]}$, the i,j-th entry of $A^{[t]}$, satisfies $a_{i,j}^{[t]} = 1$, if there is a path of exactly t edges from vertex v_i to vertex v_j in g (and $a_{i,j}^{[t]} = 0$, otherwise). Similarly, $a_{i,j}^{\langle t \rangle}$, the i,j-th entry of $A^{\langle t \rangle}$, satisfies $a_{i,j}^{\langle t \rangle} = 1$, if there is a path of at most t edges from vertex v_i to vertex v_j in G (and $a_{i,j}^{\langle t \rangle} = 0$, otherwise). Thus, $a_{i,j}^{[0]} = 1$, if and only if $i = j$, $a_{i,j}^{[1]} = 1$, if and only if $(v_i, v_j) \in E$, and $a_{i,j}^{\langle 1 \rangle} = 1$ if and only if $i = j$ or $(v_i, v_j) \in E$ (that is $a_{i,j}^{\langle 1 \rangle} = a_{i,j}^{[0]} \vee a_{i,j}^{[1]}$)..

   Now suppose that $a_{i,j}^{[2]} = 1$. This holds if and only if there exists a vertex v_k such that $(v_i, v_k) \in E$ and $(v_k, v_j) \in E$, which is true if and only if $\bigvee_k (a_{i,k} \wedge a_{k,j})$, the$ij$-th entry of the Boolean product of $A^{[1]}$ with itself, is equal to 1.

   Similarly, $a_{i,j}^{\langle 2 \rangle} = 1$ holds if and only if $a_{i,j}^{[2]} = 1$ or $a_{i,j}^{[1]} = 1$ or $a_{i,j}^{[0]} = 1$, which holds if and only if the ij-th entry of the Boolean product of $A^{[1]}$ with itself, or the Boolean product of $A^{[1]}$ with the identity matrix I, or the identity matrix I itself, is equal to 1. But this holds exactly when the ij-th entry of the Boolean product of $A^{\langle 1 \rangle} = (A^{[1]} \vee I)$ with itself is equal to 1.

   (b) This follows, by induction on t. In particular, (i) the basis of the induction ($t = 1$) is trivial, and (ii) the induction step follows by the observation that $A^{[t]} = A^{[t-1]} \cdot A^{[1]}$ and $A^{\langle t \rangle} = A^{\langle t-1 \rangle} \cdot A^{\langle 1 \rangle}$, using essentially the same argument as in part (a).

   (c) Here it suffices to observe that there exists a path in G joining vertex v_i to vertex v_j if and only if there exists such a path with at most $n-1$ edges (since any longer path must contain a cycle whose removal would produce a path with fewer edges). Thus $a_{ij}^* = 1$ if and only if $a_{i,j}^{\langle t \rangle} = 1$, for all $t \geq n - 1$.

2. (a) As suggested in the hint, we can represent the priority queue of d-values (maintained by Dijkstra's algorithm) as a list structure $L[0 : m + 1]$, where $L[i]$ points to a doubly-connected list containing all vertices $v \in V - S$ with $d[v] = i$, and $L[m + 1]$ points to a doubly-connected list of vertices $v \in V - S$ with $d[v] > m$. We maintain an index max of the maximum d-value extracted from the priority queue

so far (initially max $=0$). We exploit the fact that max increases monotonically over time.

We EXTRACT-MIN by:

> **while** ($L[\max] = nil$) $\max \leftarrow \max + 1$
> extract the first element from $L[\max]$

We DECREMENT-KEY(x, key) by:

> remove x from its list
> add x to list $L[key]$

Assuming that the lists are doubly-linked (for fast removal) the total cost for all priority queue operations is $O(n+m)$.

(b) Since $c_1(u,v) = \lfloor c(u,v)/2^{k-1} \rfloor \in \{0,1\}$, it follows that $\delta_1(s,v) \le n-1 \le m$ (since we can assume that our graph is connected). The result follows from part (a).

(c) Suppose that $c(u,v) = \sum_{0 \le j \le k} b_j 2^j$. That is, $c(u,v) = (b_{k-1} b_{k-2} \cdots b_0)_2$. Then $c_i(u,v) = (b_{k-1} \cdots b_{k-i})_2$ and $c_{i-1}(u,v) = (b_{k-1} \cdots b_{k-i+1})_2$. Hence, $c_i(u,v) = 2c_{i-1}(u,v) + b_{k-i}$.

Suppose that path P_{i-1} realizes $\delta_{i-1}(s,v)$ and path P_i realizes $\delta_i(s,v)$. That is $c_{i-1}(P_{i-1}) = \delta_{i-1}(s,v)$ and $c_i(P_i) = \delta_i(s,v)$. Then
$\delta_i(s,v) \le c_i(P_{i-1}) \le 2c_{i-1}(P_{i-1}) + |P| \le 2\delta_{i-1}(s,v) + n - 1$ and
$\delta_i(s,v) = c_i(P_i) \ge 2c_{i-1}(P_i) \ge 2\delta_{i-1}(s,v)$.

(d) Since $\delta_{i-1}(s,v) \le \delta_{i-1}(s,u) + c_{i-1}(u,v)$, by the optimality condition for δ_{i-1}, it follows that
$2\delta_{i-1}(s,v) \le 2\delta_{i-1}(s,u) + 2c_{i-1}(u,v) \le 2\delta_{i-1}(s,u) + c_i(u,v)$. Thus, $\hat{c}_i(u,v) \ge 0$.

(e) Let P be any path from s to v: $P = \langle v_0, v_1, \ldots, v_k \rangle$. Then

$$
\begin{aligned}
\hat{c}_i(P) &= \sum_{j=1}^{k} \hat{c}_i(v_{j-1}, v_j) \\
&= \sum_{j=1}^{k} [c_i(v_{j-1}, v_j) + 2\delta_{i-1}(s, v_{j-1} - 2\delta_{i-1}(s, v_j)] \\
&= [\sum_{j=1}^{k} [c_i(v_{j-1}, v_j)] - 2\delta_{i-1}(s,v) \\
&= c_i(P) - 2\delta_{i-1}(s,v).
\end{aligned}
$$

Hence $\hat{\delta}_i(s,v) = \delta_i(s,v) - 2\delta_{i-1}(s,v)$, and $\hat{\delta}_i(s,v) \le n-1 \le m$ (by part (c)).

(f) Given $\delta_{i-1}(s,v)$, for all $v \in V$, construct $\hat{c}_i$ values and compute $\hat{\delta}_{i-1}(s,v)$, for all $v \in V$, using (e). The cost is $O(m)$ by (a). Now construct $\delta_i(s,v)$, for all $v \in V$, using (e). Repeating this for i from 2 to k $(= \lg C)$, we construct $\delta_k(s,v) = \delta(s,v)$, for all $v \in V$, in $O(E \lg C)$ time in total.

3. (a) Suppose the G, k is an instance of the vertex cover problem. If we transform G to the edge coloured graph H as described, and we choose $s = v'_0$ and $t = v'_n$, then we claim that H has a path from s to t using at most k colours if and only if G has a vertex cover of size at most k. Suppose that G has a vertex cover $\{v'_{i_1}, \ldots v'_{i_k}\}$. Then every edge e_j in E_G is incident on at least one of the vertices in this set. It follows from the construction that every vertex v'_j of H has an incoming edge coloured by one of the k colours $c_{i_1}, \ldots, c_{i_k}$. Hence, there is a path from s to t in H using colours in the set $c_{i_1}, \ldots, c_{i_k}$. Similarly, if there is a path from s to t in H using colours in the set $c_{i_1}, \ldots, c_{i_k}$, then it follows from the construction that every vertex v'_j of H has an incoming edge coloured by one of the k colours $c_{i_1}, \ldots, c_{i_k}$. Thus every edge e_j in E_G is incident on at least one of the vertices in the set $\{v'_{i_1}, \ldots v'_{i_k}\}$, that is $\{v'_{i_1}, \ldots v'_{i_k}\}$ is a vertex cover of G.

   (b) The reduction is a polynomial time reduction since H has $|E_G|$ vertices, $2|E_G|$ edges and $|V_G|$ colours (and the decision as to which vertices to connect with a given edge and which colour to assign to a given edge can be made in $O(1)$ time).

   (c) It follows from the reduction above that the decision version of the minimum colour $s - t$ path problem is **NP**-hard (since the vertex cover problem is **NP**-hard). To show **NP**-completeness it remains to argue that the decision version of the minimum colour $s - t$ path problem is in **NP**. This follows because a yes-instance of the decision version of the minimum colour $s - t$ path problem can be certified by demonstrating a path from s to t using r colours (which is trivial to verify in polynomial time in the size of H).

4. (a) We use the notation $\overline{\alpha}$ to denote the negation of the literal α. Suppose there is an edge from literal α_i to a literal α_{i+1} in G. Then the disjunct $\overline{\alpha_i} \vee \alpha_{i+1}$ must be a disjunct in $\mathcal{E}$. This means that any truth assignment that satisfies $\mathcal{E}$ and assigns the truth value `true` to the literal α_i must assign `true` to the literal α_{i+1}. The more general result, that if there is a path from a literal α to a literal β in G, then any satisfying truth assignment of $\mathcal{E}$ that assigns `true` to α must also assign `true` to β, follows by induction of the length of the path.

   (b) From part (a) we conclude that if there is a path from a literal α to its negation $\overline{\alpha}$, and a path from $\overline{\alpha}$ to α, then any satisfying truth assignment of $\mathcal{E}$ that assigns `true` to α must also assign assign `true` to $\overline{\alpha}$ (and hence `false` to α), and any satisfying truth assignment of $\mathcal{E}$ that assigns `true` to $\overline{\alpha}$ must also assign `true` to α. Since both assignments lead to a contradiction it follows that $\mathcal{E}$ is not satisfiable.

(c) We argue by induction on the number of variables in our formula, noting that any formula with zero variable is trivially satisfiable.

Suppose that for all literals α, if $\overline{\alpha}$ is reachable from α in G then α is not reachable from $\overline{\alpha}$ in G. We describe a greedy algorithm to construct a satisfying truth assignment. Choose a literal α_1 arbitrarily that has the property that there is no path in G from α_1 to $\overline{\alpha_1}$, and assign the value `true` to α_1 and all literals reachable from α_1 in G.

Since for any path from α to β in G there is a corresponding (reversed) path from $\overline{\beta}$ to $\overline{\alpha}$, it follows that this partial truth assignment is *consistent* i.e, it does not assign `true` to any literal β as well as its negation $\overline{\beta}$ (otherwise, $\overline{\alpha_1}$ would be reachable from α_1, by the concatenation of paths from α_1 to β and β to $\overline{\alpha_1}$).

Furthermore, this partial truth assignment satisfies all disjuncts that contain one of the literals reachable from α_1, or their negation. (If a disjunct D contains literal β that is reachable from α_1 then it is obviously satisfied by the assignment `true` to β. On the other hand if D contains the literal $\overline{\beta}$, for some literal β that is reachable from α_1, then the other literal in D is reachable from α_1.) Thus, if we remove all such disjuncts we have a smaller formula, with fewer variables to which the induction hypothesis applies, so the greedy algorithm can continue and choose another literal, say α_2 whose truth value was not forced by the assignment to α_1.