2910321 Algorithmic graph theory

Examiner's report: Zone A

General remarks

Hints and suggestions as to how good answers to the questions on this paper could look are given below. Candidates are strongly advised to study the paper together with this report. Papers from the two previous years and the corresponding examiners' reports are also essential reading during the revision for an examination.

Question 1
Let N be a connected network.

(a) When describing an algorithm, three things must be included: a description of the initial step, the recursive step and the stop-condition. Kruskal's algorithm stops when $i = |V(N)| - 1$.

Kruskal's algorithm:
Initial Step: Choose an edge e_1 of minimum weight in N and let H_1 be the spanning subgraph of N with $E(H_1) = \{e_1\}$.

Recursive step: Suppose we have constructed a spanning subgraph H_i with $E(H_i) = \{e_1, e_2, \ldots, e_i\}$ for some $i \geq 1$. While $i < |V(N)| - 1$, choose an edge e_{i+1} of N such that
(a) $H_i + e_{i+1}$ contains no cycles, and
(b) subject to (a), $w(e_{i+1})$ is as small as possible.

(b) The minimum weight spanning tree formed by using Kruskal's algorithm has weight 99 and consists of the following edges (listed in the order chosen by the algorithm):

$$v_5v_6, v_1v_6, v_5v_7, v_2v_7, v_4v_5, v_3v_4, v_3v_8.$$

Note that edge v_4v_8 could have been chosen instead of edge v_3v_8.

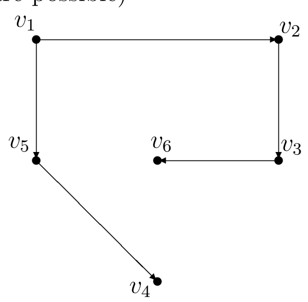
(c) A digraph D is strongly connected when every ordered pair of vertices u, v are joined by a directed walk in D.

A strongly connected component of D is a maximal strongly connected subgraph of D.

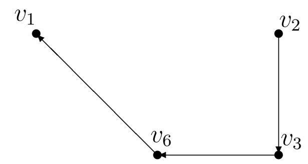
(d) Let u, v be vertices in $V(H_1 \cup H_2)$. We must show that in all cases there exists a directed path from u to v and a directed path from v to u. Because H_1 and H_2 are both strongly connected, if u and v are both in $V(H_1)$ or both in $V(H_2)$, the required pair of directed paths joining u to v exists.

So suppose (without loss of generality) that $u \in V(H_1)$ and $v \in V(H_2)$. Also let x be a vertex in $V(H_1) \cap V(H_2)$. Then because H_1 is strongly connected, there is a directed path joining u to x and because H_2 is strongly connected, there is a directed path joining x to v. We can combine these paths to form a directed path from u to v. Similarly, we can find a directed path from v to u via x. Thus $H_1 \cup H_2$ is strongly connected as required.

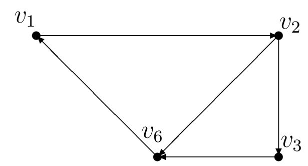
(e) A maximal tree T^+ is grown from vertex v_1 such that all arcs are directed away from v_1, e.g. (other trees are possible)



A maximal tree T^- is grown from vertex v_1 such that all arcs are directed towards v_1, e.g. (other trees are possible)



We combine them as follows. If D_1 is the strongly connected component containing v_1, we have $V(D_1) = V(T^+) \cap V(T^-)$ and $E(D_1)$ is the set of all arcs from D that use only vertices in $V(D_1)$:



This is the strongly connected component containing v_1.

Question 2

(a) (i) Consider an $s \in U$ and a $t \in \overline{U}$ and assume $st \in A(D)$. Since $s \in U$, D has a directed us-path. Thence, if $st \in A(D)$ we can extend this us-path along st to give us a ut-path in D, contradicting that $t \notin U$. Hence there are no arcs from U to $\overline{U}$, proving that $d_D^+(U) = 0$.

(ii) D is *k-arc-connected* when $D - S$ is strongly connected for every set of arcs S with $|S| < k$.

(iii) Necessity: Suppose that $d_D^+(U) < k$ for some proper subset U of $V(D)$. Putting $S = A_D(U, \overline{U})$, we have that $D - S$ is *not* strongly connected and $|S| = d_D^+(U) < k$. Thus D is not k-arc-connected.

Sufficiency: Suppose that D is not k-arc-connected. This means that there exists a subset of arcs S with $|S| < k$ such that the graph $H = D - S$ is not strongly connected. By (i) we can then find a proper subset U of $V(H)$ with $d_H^+(U) = 0$. Thus all arcs in D from U to $\overline{U}$ belong to S. Hence $d_D^+(U) \leq |S| < k$.

(b) (i) The value of a maximum flow is equal to the capacity of a minimum xy-cut. That is:
$\max\{val(f) : f \text{ is an } xy\text{-flow}\} = \min\{c^+(U) : A_N(U, \overline{U}) \text{ is an } xy\text{-cut in } N\}$.

(ii) Candidates should use a tree-growing procedure to grow maximal flow augmenting trees from x, identify flow-augmenting xy-paths and augment along them. More than one solution is possible. One possibility is to augment by 1 along the path x, v_1, v_3, v_6, y and then by 1 along the path x, v_5, v_4, v_7, y to obtain the final flow.

After augmenting along these paths, candidates should draw a maximal f-unsaturated tree from x which will include the vertices $U = \{x, v_1, v_2, v_3, v_4, v_5\}$ and no others. This set U has $c^+(U) = 3 + 1 + 3 + 5 = 12$ and the value of the flow is $8 + 1 + 3 = 12$ also. Thus, the flow is a maximum xy-flow in N.

Question 3

(a) (i) $M \subseteq E(G)$ is a *matching* when no two edges of M have a common end vertex. M is a *maximum matching* if $|M| \geq |M_0|$ for any matching M_0. M is a *perfect matching* when every vertex of G is M-saturated.

(ii) Hall's theorem states that G on vertex set $X \cup Y$ has a matching that saturates X if and only if $|N_G(S) \geq |S|$ for all $S \subseteq X$.

(b) Unsaturated vertices in X are x_2 and x_6. By growing an M_1-alternating tree from x_2 we identify an M_1-augmenting path $P_1 : x_2y_2, y_2x_1, x_1y_1$. Thus,

$$M_2 = M_1 \Delta P_1 = \{x_1y_1, x_2y_2, x_3y_4, x_4y_6, x_5y_5\}.$$

By growing an M_2-alternating tree from x_6 we identify an M_2-augmenting path path $P_2 : x_6y_4, y_4x_3, x_3y_2, y_2x_2, x_2y_6, y_6x_4, x_4y_3$. Thus,

$$M_3 = M_2 \Delta P_2 = \{x_1y_1, x_2y_6, x_3y_2, x_4y_3, x_5y_5, x_6y_4\}.$$

In M_3 every vertex in X is saturated, so M_3 is a maximum (and perfect) matching.

(c) Consider a subset $S \subseteq X$ and let n denote the number of edges leaving S. Let k be the minimum vertex degree in X, then $n \geq |S|k$. Because $d_G(x) \geq d_G(y)$ for all

$x \in X$ and $y \in Y$ we have $k \geq d_G(y)$ for all $y \in N_G(S)$. Therefore the number of edges leaving $N_G(S)$ is at most $|N_G(S)|k$, and we thus have

$$|S|k \leq n \leq \text{ number of edges leaving } N_G(S) \leq |N_G(S)|k.$$

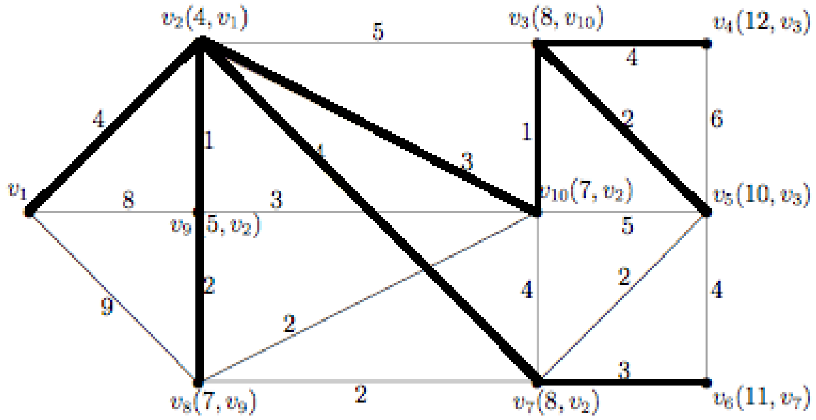
Therefore $|S| \leq |N_G(S)|$ because k is positive due to G being connected. As this holds for all choices of subset S, Hall's Theorem gives the existence of the required matching.

Question 4

(a) G has an Euler tour if and only if G is connected and every vertex of G has even degree.

(b) Let K be the weighted complete graph whose vertices are the vertices with odd degree in N. Further let each edge uv in $E(K)$ have weight equal to the length of the shortest uv-path in N. Let M be a minimum-weight perfect matching in K, and let N^* be the network obtained from N by, for each $uv \in M$, doubling the edges along the shortest uv-path in N. Then N^* has all its vertices with even degree and it thus possesses an Euler tour W. This Euler tour corresponds to the required walk in N.

(c) Below is a tree with shortest paths from vertex v_1 to every other vertex, grown using Dijkstra's algorithm.

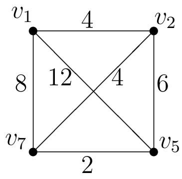


The vertices get permanent labels in the following order:

$$v_1, v_2, v_9, v_8, v_{10}, v_3, v_7, v_5, v_6, v_4.$$

In order to gain full credit here, candidates must show that the labelling and updating procedure has been performed according to the algorithm, the actual answer is less relevant and carries few marks.

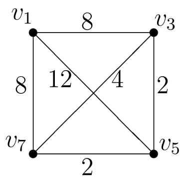
(d) The set of vertices of odd degree is $\{v_1, v_2, v_5, v_7\}$. Constructing a complete graph K_4 on these, weighted with shortest path lengths as described in (b) yields



The minimum weight perfect matching is $\{v_1v_2, v_5v_7\}$ of weight 6. Thus $m_1 = 6$ because the required walk W_1 can be achieved by doubling the edges along the shortest paths v_1v_2 and v_5v_7.

In order to have a walk W_2 which starts at v_2 and ends at v_3, we must ensure that every vertex has even degree, except for v_2 and v_3 which should have odd degree. Thus we must double edges in such a way that the degrees of v_1, v_5, v_7 change from odd to even and the degree of v_3 changes from even to odd.

As can be seen from the complete graph below, we must double along a shortest v_1v_7-path and a shortest v_3v_5-path. Note that the solution is not unique as there is more than one perfect matching of minimum weight here. We add edges $v_1v_2, v_2v_{10}, v_{10}v_7, v_3v_5$ with total weight 10 to N to achieve the required walk, and thus $m_2 = 10$.



Question 5

(a) Let G be a simple connected graph with n vertices.

(i) G is Hamiltonian when it contains a cycle that includes every vertex of G.

(ii) We construct the n-closure by recursively adding an edge between pairs of vertices whose degrees sum to at least n.

(iii) G is *1-tough* when $comp(G - S) \leq |S|$ for all proper subsets $S \subset V$. Here $comp(G - S)$ denotes the number of connected components of $G - S$.

(b) (i) $G^{(n)}$ is formed by successive additions of edges uv where u and v are vertices of G with $d(u) + d(v) \geq n$. Hence, by the assumed result, when $G^{(n)}$ is Hamiltonian, so is G.

(ii) Suppose that G is Hamiltonian and let C be a Hamiltonian cycle in G. Further, let S be a proper subset of $V(G)$. Then $S \subseteq V(C)$. Also, the vertices of S divide C into $|S|$ segments. Thus $comp(G - S) \leq comp(C - S) \leq |S|$, and it follows that G is 1-tough.

(c) Take the vertex set $S = \{v_1, v_{10}, v_5\}$ then $comp(G - S) = 4$ which is bigger than $|S| = 3$. Hence the given graph is not 1-tough, and by the result in (b) it is therefore not Hamiltonian either.

(d) Given a connected network N, find a shortest closed walk which includes every vertex of N at least once.

(e) Suppose an algorithm A efficiently solves The Travelling Salesman Problem. Define algorithm B as follows:

- Given a connected graph G on n vertices, assign each edge a length of 1.
- Run A on this network.
- If it finds a shortest closed walk visiting every vertex at least once with length n, output YES, otherwise output NO.

Clearly B outputs yes if and only if G is Hamiltonian, because the solution to the Travelling Salesman Problem will have length n if and only if the walk is a Hamiltonian cycle. Also, B is efficient, because the steps additional to A are not computationally heavy and so can be performed efficiently.