Queen Mary
University of London

Main Examination period 2021 – May/June – Semester B
Online Alternative Assessments

MTH6105: Algorithmic Graph Theory

You should attempt ALL questions. Marks available are shown next to the questions.

In completing this assessment:

- **You may use books and notes.**
- **You may use calculators and computers, but you must show your working for any calculations you do.**
- **You may use the Internet as a resource, but not to ask for the solution to an exam question or to copy any solution you find.**
- **You must not seek or obtain help from anyone else.**

All work should be **handwritten** and should **include your student number**.

You have **24 hours** to complete and submit this assessment. When you have finished:

- scan your work, convert it to a **single PDF file**, and submit this file using the tool below the link to the exam;
- e-mail a copy to **maths@qmul.ac.uk** with your student number and the module code in the subject line;
- with your e-mail, include a photograph of the first page of your work together with either yourself or your student ID card.

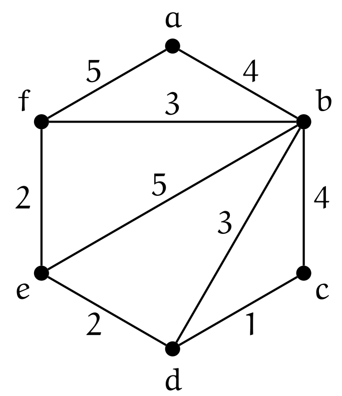
You are expected to spend about 2 hours to complete the assessment, plus the time taken to scan and upload your work. Please try to upload your work well before the end of the submission window, in case you experience computer problems. **Only one attempt is allowed – once you have submitted your work, it is final**.

Examiners: F. Fischer, R. Johnson

Continue to next page

In this paper, $\mathsf{V}(\mathsf{G})$ denotes the set of vertices of a graph or digraph G, $\mathsf{E}(\mathsf{G})$ the set of edges of a graph G, and $\mathsf{A}(\mathsf{G})$ the set of arcs of a digraph G. You may use any result from lecture notes and exercises without proving it, but you must state clearly which result you use.

Question 1 [22 marks]. Consider the following network (G, w), in which each vertex is labeled with its name and each edge $e \in \mathsf{E}(\mathsf{G})$ with its weight $w(e)$.



(a) Use Prim's algorithm starting from vertex a to find a minimum spanning tree of (G, w). Show your working and give the minimum spanning tree and its weight. **[10]**

(b) Find another minimum spanning tree of (G, w). Show your working and give the minimum spanning tree and its weight. **[4]**

(c) Does there exist a minimum spanning tree of (G, w) that does **not** contain the edge de? Justify your answer. **[4]**

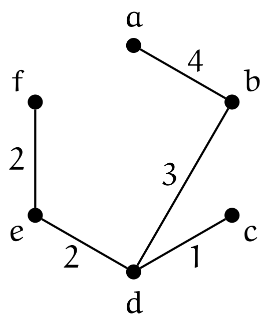
(d) Does there exist a minimum spanning tree of (G, w) that contains both of the edges bd and bf? Justify your answer. **[4]**

Model Solution

(a) Prim's algorithm finds a spanning tree by starting from the tree with vertex a and then repeatedly adding an edge of minimum weight among those that have exactly one endpoint in the current tree. It may for example add edges in the order

$$ab, bd, dc, de, ef$$

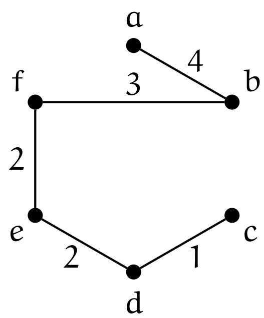
to obtain the following minimum spanning tree of weight 12.



(b) The algorithm may instead add edges in the order

$$ab, bf, fe, ed, dc$$

and obtain the following minimum spanning tree, which also has weight 12.



(c) If $S = \{e, f\}$, then the edge de is the unique edge of minimum weight among all edges in (G, w) with exactly one endpoint in S. By a result in the notes it must therefore be contained in every minimum spanning tree of (G, w).

(d) Assume for contradiction that there exists a minimum spanning tree T that contains the edges bd and bf. Since the edges $\{bd, bf, de, ef\}$ form a cycle, there must exists an edge $x \in \{de, ef\}$ such that $x \notin E(T)$. Let T' be the graph with $V(T') = V(T)$ and $E(T') = (E(T) \setminus \{bd\}) \cup \{x\}$, and observe that T' is a spanning tree of G. The weight of T' is smaller than the weight of T, a contradiction to minimality of T.

Question 2 [26 marks]. Consider a tree T in which the degree of each vertex is either 1 or 3. Let $n = |V(T)|$.

(a) Show that n is even. [**2**]

(b) Show that T has $\frac{n}{2} + 1$ leaves. [**6**]

(c) Determine the number of distinct simple graphs G such that T is a spanning tree of G. Explain your reasoning. [**6**]

Call a graph G a **tree-set** if every connected component of G is a tree.

(d) For each of the following graphs, determine if the graph is a tree-set. Justify your answer. [**6**]

(i) The graph G_1 with $V(G_1) = \{v_1, v_2, v_3, v_4, v_5, v_6\}$ and $E(G_1) = \emptyset$

(ii) The graph G_2 with $V(G_2) = V(G_1)$ and $E(G_2) = \{v_1v_2, v_2v_4, v_2v_6, v_4v_6\}$

(iii) The graph G_3 with $V(G_3) = V(G_1)$ and $E(G_3) = \{v_1v_5, v_2v_6, v_3v_4, v_5v_6\}$

Now consider an arbitrary graph G.

(e) Give a polynomial-time algorithm to determine whether G is a tree-set. Show that the algorithm is indeed a polynomial-time algorithm. [**6**]

Model Solution

(a) T is a graph, so it must have an even number of vertices of odd degree. In the given graph all n vertices have an odd degree, so n must be even.

(b) Let $m = |E(T)|$. Let n_1 and n_3 respectively denote the number of vertices of degree 1 and degree 3. Then $n = n_1 + n_3$ and thus $n_3 = n - n_1$. Since T is a tree, by a result in the notes, $m = n - 1$. Since T is a graph, by a result in the notes, $2m = \sum_{v \in V(T)} d_T(v) = n_1 + 3n_3 = n_1 + 3(n - n_1) = 3n - 2n_1$. Thus $2(n-1) = 3n - 2n_1$, $2n_1 = n + 2$, and $n_1 = \frac{n}{2} + 1$.

(c) If T is a spanning tree of G, then $V(G) = V(T)$ and $V(G) \supseteq V(T)$. Each of the $\binom{n}{2} - |E(T)| = \binom{n}{2} - (n-1)$ unordered pairs of vertices that are **not** an edge of T may or may not be an edge of G, so there are $2^{\binom{n}{2}-(n-1)}$ such graphs.

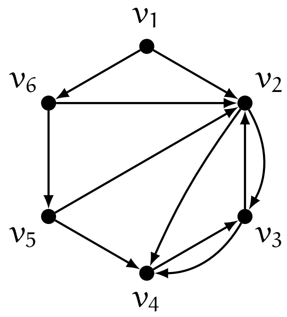
(d) (i) G_1 has six connected components, each of which consists of a single vertex and no edges and is therefore a tree. Therefore G_1 is a tree-set.

(ii) G_2, and indeed one of its connected components, contains the cycle v_2, v_4, v_6, v_4. That connected component is not a tree, so G_2 is not a tree-set.

(iii) G_3 has two connected components, a path of length 1 and a path of length 3. Both connected components are trees, so G_3 is a tree-set.

(e) For $v \in V(G)$, we can determine whether the connected component of G containing v is a tree by running tree search starting at v and checking in each iteration for the existence of edges with both endpoints inside the current tree. The additional check can be performed without increasing the running time by more than a constant factor, so the running time remains $O(|V(G)| \cdot |E(G)|)$. If G has more than one connected component tree search will not immediately find all vertices, and we need to repeat the procedure starting from a vertex that has not been found so far. We run tree search at most $|V(G)|$ times, so the overall running time is $O(|V(G)|^2 \cdot |E(G)|)$.

Question 3 [24 marks]. Consider the following digraph D, in which each vertex is labeled with its name.



(a) Determine the strongly connected components of D. Show your working. **[6]**

Now consider the directed network (D, w), where $w : A(D) \to \mathbb{R}$ with

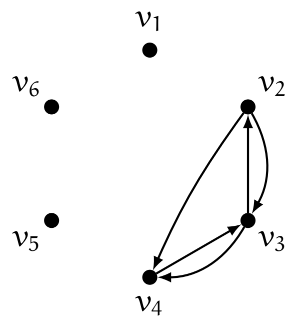
$$
\begin{array}{llll}
w(v_1v_2) = 4, & w(v_1v_6) = -1, & w(v_2v_3) = 1, & w(v_2v_4) = -2 \\
w(v_3v_2) = 2, & w(v_3v_4) = 2, & w(v_4v_3) = 1, & w(v_5v_2) = 2 \\
w(v_5v_4) = 1, & w(v_6v_2) = 4, & w(v_6v_5) = 1. &
\end{array}
$$

(b) Recall that Dijkstra's algorithm may fail to find shortest directed paths in the presence of negative weights. Illustrate this fact by giving vertices $u, v \in V(D)$ such that Dijkstra's algorithm fails to find a shortest directed u–v-path in (D, w). Explain your reasoning. **[6]**

(c) Use the Bellman-Ford algorithm to find a shortest directed v_1–v_3-path in (D, w). Show your working. **[12]**

Model Solution

(a) The strongly connected component containing a particular vertex $v \in V(D)$ can be found by running tree search twice, once to find the set of vertices u such that there exists a v—u-path in D and once to find the set of vertices u such that there exists a u—v-path. Executing this procedure repeatedly from a vertex that has not so far been discovered we obtain the following strongly connected components.



(b) If Dijkstra's algorithm is started from v_5 it adds the arc v_5v_4 and concludes that the path v_5—$, v_4$ of length 1 is a shortest v_5—v_4-path. There is, however, a shorter v_5—v_4-path, namely the path v_5, v_2, v_4 of length 0.

(c) The Bellman-Ford algorithm constructs the following table, where the entry in the row for a particular value of k and the column for vertex v is the length $\delta_k(v)$ of a shortest v_1—v-path in (G, w) that uses at most k arcs.

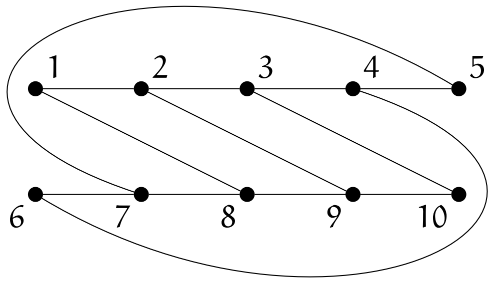
k	v_1	v_2	v_3	v_4	v_5	v_6
0	0	∞	∞	∞	∞	∞
1	0	$4(v_1)$	∞	∞	∞	$-1(v_1)$
2	0	$3(v_6)$	$5(v_2)$	$2(v_2)$	$0(v_6)$	$-1(v_1)$
3	0	$2(v_5)$	$3(v_4)$	$1(v_2)$	$0(v_6)$	$-1(v_1)$
4	0	$2(v_5)$	$2(v_4)$	$0(v_2)$	$0(v_6)$	$-1(v_1)$
5	0	$2(v_5)$	$1(v_4)$	$0(v_2)$	$0(v_6)$	$-1(v_1)$
6	0	$2(v_5)$	$1(v_4)$	$0(v_2)$	$0(v_6)$	$-1(v_1)$

The rows for $k = 5$ and $k = 6$ are identical, which means that the algorithm has not encountered a negative cycle and has therefore worked correctly. The entry for $k = 5$ and v_3 gives us the length of a shortest v_1—v_3-path, which is 1, as well as the predecessor on such a path, v_4. Tracing back the sequence of predecessors we obtain the shortest path itself, $v_1, v_6, v_5, v_2, v_4, v_3$.

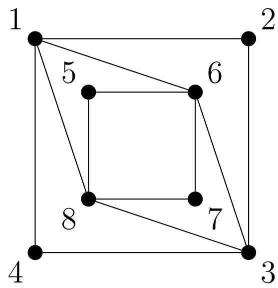
Question 4 [28 marks].

(a) For each of the following graphs, state if the graph is bipartite or not. Justify your answer. [**6**]

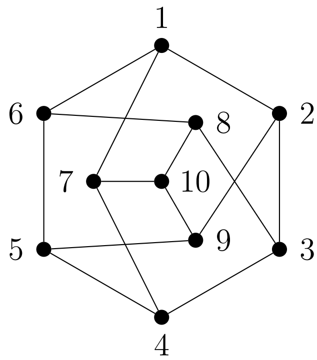
(i)



(ii)



(iii)



Now consider the bipartite graph G with

$$\begin{aligned} \mathsf{V}(\mathsf{G}) &= \{u_1, u_2, u_3, u_4, u_5, u_6, v_1, v_2, v_3, v_4, v_5, v_6\} \\ \mathsf{E}(\mathsf{G}) &= \{u_1v_2, u_1v_4, u_2v_1, u_2v_2, u_2v_3, u_2v_4, u_2v_5, u_2v_6, u_3v_2, u_3v_4, \\ &\qquad u_4v_2, u_4v_3, u_4v_4, u_5v_2, u_5v_4, u_6v_1, u_6v_2, u_6v_4, u_6v_5, u_6v_6\}. \end{aligned}$$

(b) Let $\mathsf{U} = \{u_1, u_3, u_5, v_2, v_4, v_6\}$. Draw $\mathsf{G}[\mathsf{U}]$, the induced subgraph of G with vertex set U. [**2**]

(c) For each of the following sets, state if the set is a matching of G or not. Justify your answer. [**6**]

(i) $M_1 = \{u_1v_2, u_2v_1, u_5v_2, u_6v_4\}$

(ii) $M_2 = \{u_1v_2, u_2v_1, u_3v_3, u_4v_4\}$

(iii) $M_3 = \{u_1v_4, u_2v_2, u_4v_3, u_6v_5\}$

(d) Find a maximum matching of G. Show your working. [**10**]

(e) Show that the matching you have found is indeed a maximum matching. [**4**]

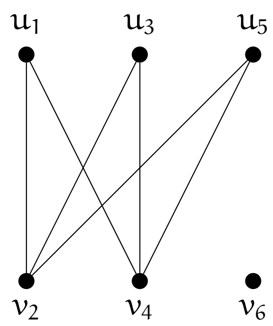
Model Solution

(a) (i) The graph contains the cycle $1, 2, 3, 4, 5, 7, 8, 1$ of odd length, so by a result in the notes it cannot be bipartite.

(ii) Let $L = \{1, 3, 5, 7\}$ and $R = \{2, 4, 6, 8\}$. Then L and R form a partition of the set of vertices of the graph, and every edge of the graph has one endpoint in L and one endpoint in R. Thus the graph is bipartite.

(iii) The graph contains the cycle $1, 7, 4, 5, 6, 1$ of odd length, so by a result in the notes it cannot be bipartite.

(b) $G[U]$ looks as follows.

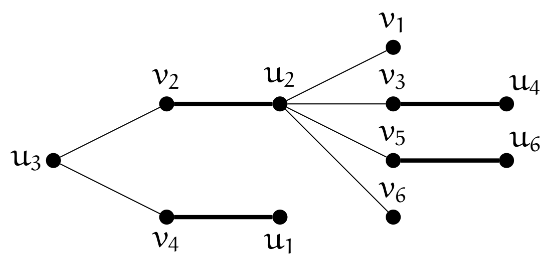


(c) (i) M_1 is not a matching of G because v_2 is an endpoint of more than one of the edges in M_1.

(ii) M_2 is not a matching of G because $M_2 \setminus E(G) = \{u_3v_3\}$ and thus $M_2 \not\subseteq E(G)$.

(iii) M_3 is a matching of G because $M_3 \subseteq E(G)$ and each vertex in $V(G)$ appears at most once as an endpoint of an edge in M_3.

(d) Starting from vertex u_3, which is not saturated by M_3, we can construct the following maximal tree of alternating paths:



This tree contains the augmenting path u_3, v_2, u_2, v_1, which in turn yields the matching $M = M_3 \triangle \{u_3v_2, u_2v_2, u_2v_1\} = \{u_1v_4, u_2v_1, u_3v_2, u_4v_3, u_6v_5\}$. M has cardinality $|M| = |M_3| + 1 = 5$, and we will see that it is a maximum matching.

(e) Since $|M| = 5$, M is a maximum matching unless the graph has a perfect matching. Let $X = \{u_1, u_3, u_5\}$, and observe that $N(X) = \{v_2, v_4\}$. Thus $|X| < |N(X)|$, so by Hall's theorem the graph does not have a perfect matching.

End of Paper.