Algorithms Final Exam

Ozaner Hansha

May 12, 2020

Problem 1

Consider a recurrance relation of the form $T(n) = 8T\left(\frac{n}{b}\right) + O(n^d)$ where $T(1) = 1$.

Part a: Assign the smallest positive integer values to b and d so that $T(n) = O(n^3)$.

Solution: Call the $O(n^d)$ term in $T(n)$, $f(n)$. Now recall that:

$$f(n) = O(n^{\log_b 8 - \epsilon}) \implies T(n) = \Theta(n^{\log_b 8})$$

Where ϵ is some positive contant. We must have that $b = 2$ so that $\log_2 8 = 3$ and thus $T(n) = \Theta(n^3)$. We must now pick a d such that $O(n^d) = O(n^{\log_3 8 - \epsilon}) = O(n^{3-\epsilon})$. The smallest positive integer that satisfies this role is $d = 1$. Thus our variables are $\boldsymbol{b = 2}$ and $\boldsymbol{d = 1}$.

Part b: Assign values to b and d so that $T(n) = O(n^2 \log n)$.

Solution: Call the $O(n^d)$ term in $T(n)$, $f(n)$. Now recall that:

$$f(n) = \Theta(n^{\log_b 8}) \implies T(n) = \Theta(n^{\log_b 8} \log n)$$

We must have that $b = 2\sqrt{2}$ so that $\log_{2\sqrt{2}} 8 = 2$ and thus $T(n) = \Theta(n^2 \log n)$. We must now pick a d such that $O(n^d) = \Theta(n^{\log_{2\sqrt{2}} 8}) = \Theta(n^2)$. We can simply pick $d = 2$. Thus our variables are $\boldsymbol{b = 2\sqrt{2}}$ and $\boldsymbol{d = 2}$.

Problem 2

Problem: Given an unsorted array A of size n, give a linear time algorithm to output all elements between 80th and 90th percentile.

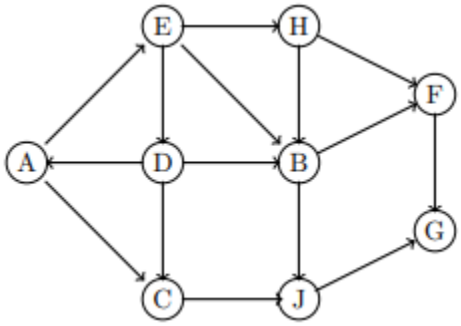
Solution: Let $k_1 = \lfloor .8n \rfloor$ and $k_2 = \lfloor .9n \rfloor$. We now perform the kth smallest element algorithm on the list A for k_1, which takes $O(n)$ time. This nets us the 80th percentile a_1 of A. We now repeat this process for k_2, again taking $O(n)$ time, netting us the 90th precentile a_2 of A.

With these percentiles in hand, we now simply iterate through all n elements of A and print out the elements a for which $a_1 \leq a \leq a_2$. This iteration again takes $O(n)$ time.

This algorithm prints out all values of A that fall in between its 80th and 90th percentile. And, as desired, the total complexity of the algorithm is given by $3 \cdot O(n) = O(n)$.

Problem 3

Problem: Do a DFS on the following graph starting at A, assuming vertices are considered in alphabetical order. Show discovery and finish times and classify edges as tree edges, forward edges, back edges, and cross edges.



Solution: The discovery and finish times for each vertex are given below:

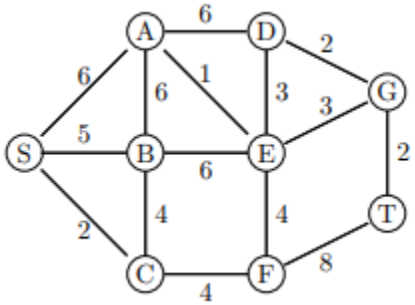
Vertex	Discovery Time	Finished Time
A	1	18
B	9	12
C	2	7
D	13	14
E	8	17
F	10	11
G	4	5
H	15	16
J	3	6

Each of the edges is classified below:

Tree Edges	Forward Edges	Cross Edges	Backward Edges
$(A,C),(C,J),(J,G),$ $(A,E),(E,B),(B,F),$ $(E,D),(E,H)$		$(D,C),(D,B),(H,B),$ $(H,F),(F,G),(B,J),$	(D,A)

Problem 4

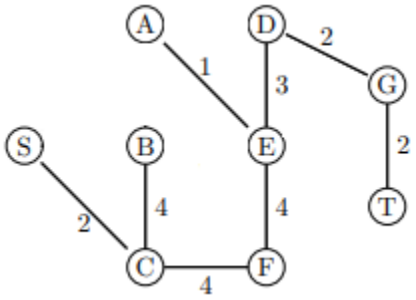
Part a: Compute the MST of the following graph using Kruskal's algorithm. Show the order of the selected edges. If there is a tie, select edges in dictionary ordering, i.e. if you choose edge (u,v), then uv lexicographically precedes any other edge label xy:



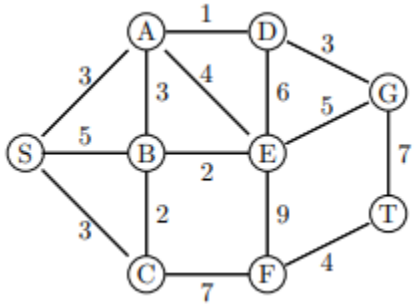
Solution: The sets of spanned vertices and corresponding selected edge are given in the following table:

Time Step	Selected Edge	Added?	Sets of Spanned Vertices
0	-	-	$\{A\},\{B\},\{C\},\{D\},\{E\},\{F\},\{G\},\{S\},\{T\}$
1	(A,E)	Yes	$\{A,E\},\{B\},\{C\},\{D\},\{F\},\{G\},\{S\},\{T\}$
2	(C,S)	Yes	$\{A,E\},\{B\},\{C,S\},\{D\},\{F\},\{G\},\{T\}$
3	(D,G)	Yes	$\{A,E\},\{B\},\{C,S\},\{D,G\},\{F\},\{T\}$
4	(G,T)	Yes	$\{A,E\},\{B\},\{C,S\},\{D,G,T\},\{F\}$
5	(D,E)	Yes	$\{A,D,E,G,T\},\{B\},\{C,S\},\{F\}$
6	(E,G)	No	$\{A,D,E,G,T\},\{B\},\{C,S\},\{F\}$
7	(B,C)	Yes	$\{A,D,E,G,T\},\{B,C,S\},\{F\}$
8	(C,F)	Yes	$\{A,D,E,G,T\},\{B,C,F,S\}$
9	(E,F)	Yes	$\{A,B,C,D,E,F,G,S,T\}$

At step 9, all sets have been combined and our MST is complete:



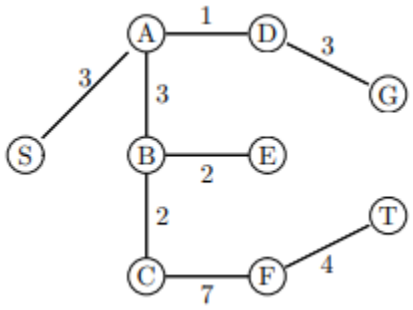
Part b: Compute the MST of the following graph using Prim's algorithm. Start at S and give the order of the selected edges:



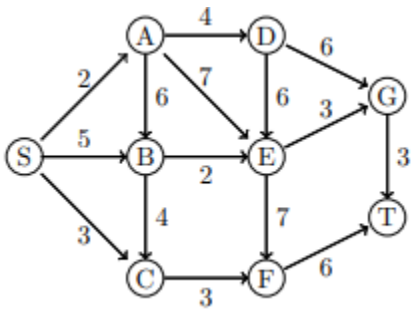
Solution: Running Prim's algorithm starting at S nets us:

Order	Edge	Spanned Vertices
1	(A,S)	A,S
2	(A,D)	A,D,S
3	(A,B)	A,B,D,S
4	(B,C)	A,B,C,D,S
5	(B,E)	A,B,C,D,E,S
6	(D,G)	A,B,C,D,E,G,S
7	(C,F)	A,B,C,D,E,F,G,S
8	(F,T)	A,B,C,D,E,F,G,S,T

At step 8, all vertices have been spanned and our MST is complete:



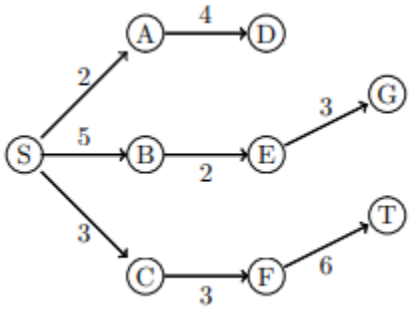
Part c: Use Dijkstra's algorithm to find the shortest path from S to all other vertices.



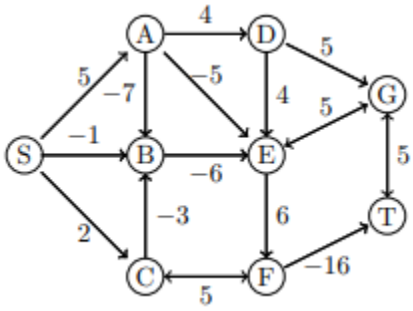
Solution: Each iteration of Dijkstra's algorithm is given below:

Iteration	***S***	***A***	***B***	***C***	***D***	***E***	***F***	***G***	***T***
0	0, Null	∞, Null	∞, Null	∞, Null	∞, Null	∞, Null	∞, Null	∞, Null	∞, Null
1	**0, Null**	2, S	5, S	3, S	∞, Null	∞, Null	∞, Null	∞, Null	∞, Null
2	-	**2, *S***	5, S	3, S	6, A	9, A	∞, Null	∞, Null	∞, Null
3	-	-	5, S	**3, *S***	6, A	9, A	6, C	∞, Null	∞, Null
4	-	-	**5, *S***	-	6, A	7, B	6, C	∞, Null	∞, Null
5	-	-	-	-	**6, *A***	7, B	6, C	12, D	∞, Null
6	-	-	-	-	-	7, B	**6, *C***	12, D	12, F
7	-	-	-	-	-	**7, *B***	-	10, E	12, F
8	-	-	-	-	-	-	-	**10, *E***	12, F
9	-	-	-	-	-	-	-	-	**12, *F***

From this table we can construct the following shortest path tree with root S:



Part d: By inspection, find the shortest distance from S to all other vertices, finite or infinite, in the following graph:



Solution: By inspection, we find the following minimum distance paths from S to all other vertices:

Vertex	Min. Dist.	Min. Path
A	5	SA
B	-2	SAB
C	2	SC
D	9	SAD
E	-8	$SABE$
F	-2	$SABEF$
G	-13	$SABEFTG$
S	0	-
T	-18	$SABEFT$

Problem 5

Problem: For the patterns $AAAAAB$ and $ABABAC$, give fail(k) for each $k \in [1..6]$.

Solution: Running the KMP algorithm on both patterns we find:

k	1	2	3	4	5	6
Character	A	A	A	A	A	B
fail(k)	0	1	2	3	4	5

k	1	2	3	4	5	6
Character	A	B	A	B	A	C
fail(k)	0	1	1	2	3	4

Problem 6

Part a: Given a CNF formula on n Boolean variables $x_1, \cdots, x_n$ and their negations with 5 clauses, describe a polynomial-time algorithm that would test if it is satisfiable.

Solution: First we construct the corresponding graph of the CNF, and check if a clique of size 5 exists. If so, then the CNF is satisfiable. If not, then the CNF isn't satisfiable. Note that for a fixed k, finding a k-clique in a graph takes only polynomial time.

Part b: Convert the following CNF to an equivalent 3-CNF formula:

$$(\overline{x_1} \vee x_2 \vee x_3 \vee \overline{x_4}) \wedge (x_1 \vee x_2 \vee x_3 \vee x_4 \vee \overline{x_5})$$

Solution: We convert the above 5-CNF to a 3-CNF by introducing 3 new variables q_1, q_2, and $q3$:

$$(\overline{x_1} \vee x_2 \vee q_1) \wedge (\overline{q_2} \vee x_3 \vee \overline{x_4}) \wedge (x_1 \vee x_2 \vee q_2) \wedge (\overline{q_2} \vee x_3 \vee q_3) \wedge (\overline{q_3} \vee x_4 \vee \overline{x_5})$$

Problem 7

Part a: Given an undirected graph $G = (V, E)$, a subset of vertices V' is an independent set if each edge in E is incident to at most one vertex in V'. The independent set problem asks if G has an independent set of size k. Find a maximal independent set in the undirected form of the graph from Problem 3.

Solution: The maximum size an independent set on this graph can be is 3. One such maximal independent set is $\{A, J, F\}$.

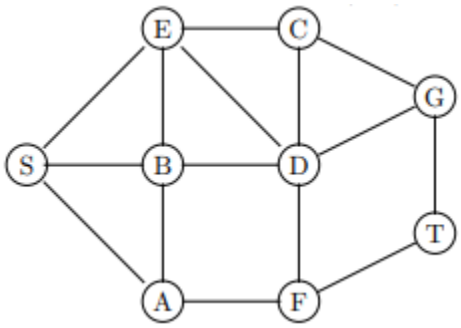
Part b: For a given graph G, consider $\overline{G}$, the complement graph of $\overline{G}$ (same vertices, complementary edges) and the clique problem. What relationship can you state between an independent set in G and clique in $\overline{G}$? What can you conclude about the complexity class of independent set problem?

Solution: Given an independent set S on some graph G, that same set S is a clique on the complementary graph $\overline{G}$. This implies that the both the maximal and minimal indpendent set of G equal the maximal and minimal clique in $\overline{G}$, respectively, and vice versa.

In terms of complexity, since constructing a graph G's complement $\overline{G}$ takes polynomial time, we know that solving the independent set problem is in the same complexity class as the clique problem, and vice versa, up to some polynomial. In particular, since the independent set problem is in NP, we know the clique problem must also be in NP. If the clique problem was in, say, P, then we could easily convert it to an independent set problem in polynomial time and solve the independent set problem in polynomial time, causing a contradiction.

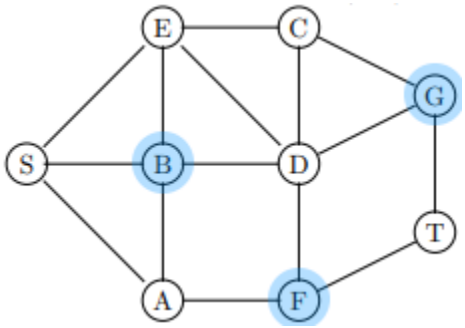
Problem 8

Consider the following graph:



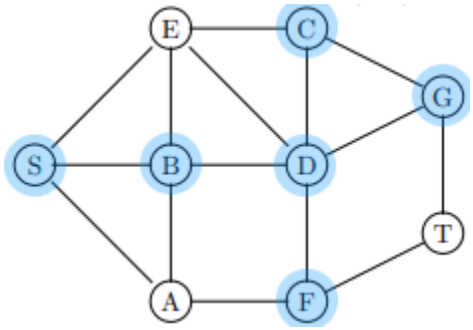
Part a: Give a maximal independent set of the graph.

Solution: This graph is essentially identical to that of Problem 7, and thus also has a maximal independent set of size 3. We present one such set below:



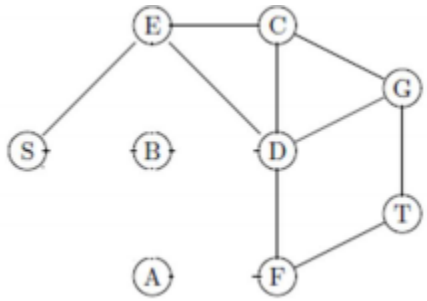
Part b: Give a minimal vertex cover of the graph.

Solution: Recall from Problem 7 part b that the maximum clique size of this graph is equal to the maximum independent set size: 3. Now also recall that, for a max clique size of k, the size of the minimum vertex cover of a graph is $|V| - k$. As such we expect a minimum vertex cover size of $9 - 3 = 6$, and indeed that is what we find:

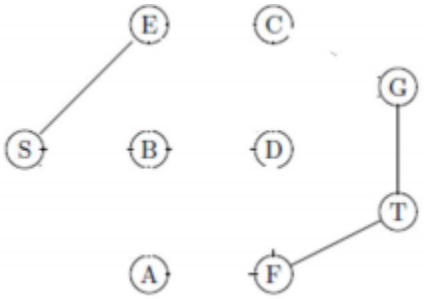


Part c: Apply the vertex cover approximation algorithm described in lecture, using edges in dictionary ordering, i.e. if you choose edge (u, v), then uv lexicographically precedes any other edge label xy.

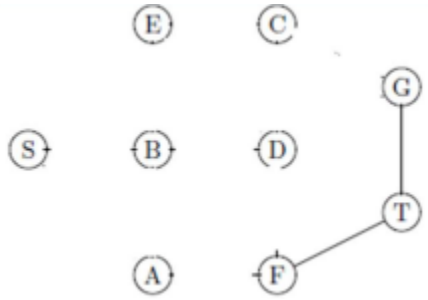
Solution: Starting with edge (A, B), our cover set is now $\{A, B\}$ and the remaining edges are:



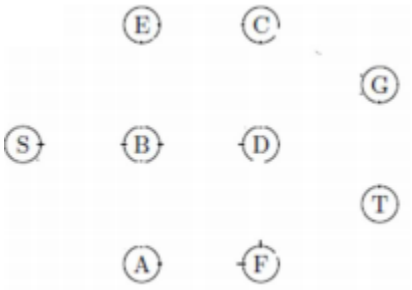
Moving onto edge (C, D), our cover set is now $\{A, B, C, D\}$ and the remaining edges are:



Moving onto edge (E, S), our cover set is now $\{A, B, C, D, E, S\}$ and the remaining edges are:



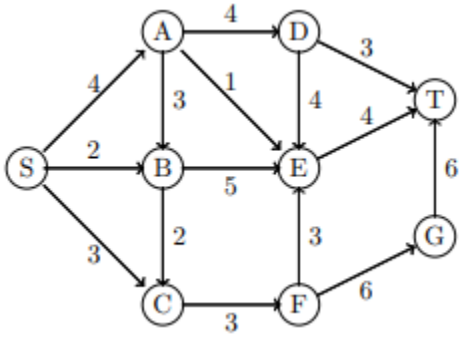
Finally, we pick edge (F, T), our cover set is now $\{A, B, C, D, E, F, S, T\}$ and there are no more remaining edges:



This leaves us with the following cover set: $\{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, \mathbf{E}, \mathbf{F}, \mathbf{S}, \mathbf{T}\}$.

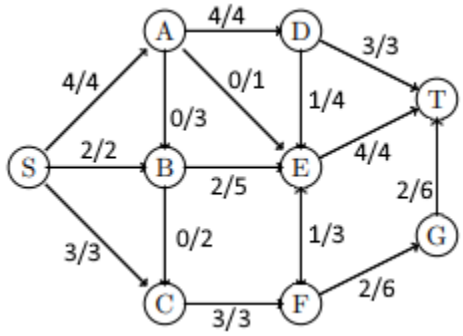
Problem 9

Consider the following directed graph:



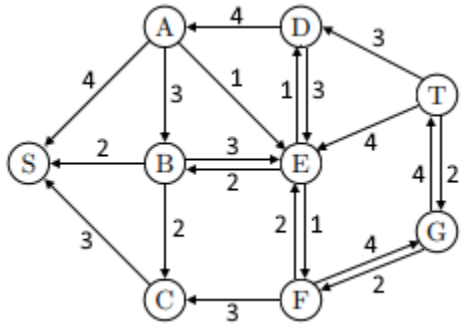
Part a: Find the maximum flow from S to T by inspection.

Solution: The maximum flow from S to T is 9. Note that there is only $4 + 2 + 3 = 9$ capacity across all the outgoing edges of S, which means the flow cannot be any higher than 9. This, combined with the example below which has a total of 9 flow into the sink T, is sufficient to prove that 9 is the max flow:



Part b: Give the residual graph corresponding to the optimal flow.

Solution: Below is the corresponding residual graph of the optimal solution given in part a:

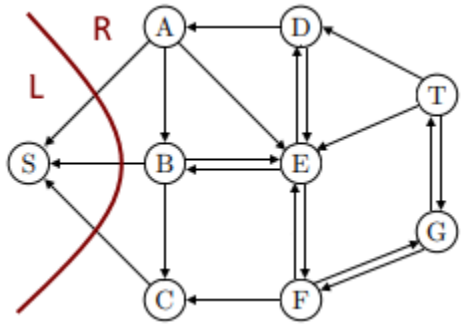


Part c: Give the minimum (S, T) cut from the residual graph in part b.

Solution: The minimum cut is given by the following partition of V:

$$L = \{S\}$$
$$R = \{A, B, C, D, E, F, T, G\}$$

Where L is the set of vertices reachable from S in the residual graph, and $R = L^{\complement}$. We can represent this graphically:

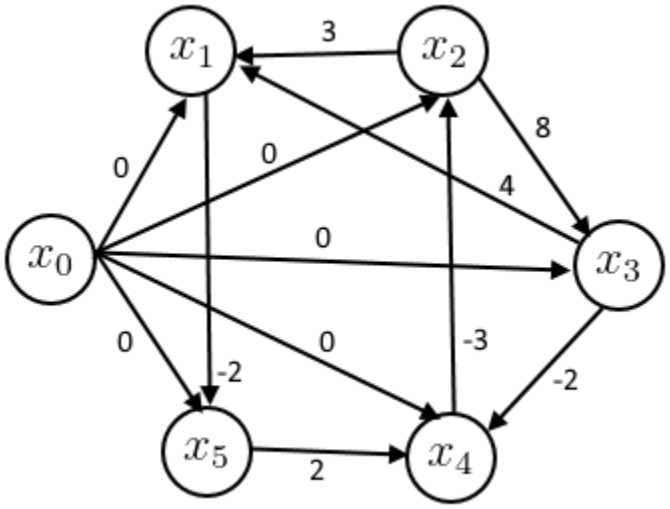


Problem 10

Problem: Using the shortest path algorithm for graphs, but via inspection, determine if the following system of inequalities has a feasible solution. If it has a feasible solution, find one. Otherwise, say why it is not feasible:

$$x_1 - x_3 \leq 4 \qquad x_5 - x_1 \leq -2 \qquad x_1 - x_2 \leq 3 \qquad x_4 - x_3 \leq -2$$
$$x_2 - x_4 \leq -3 \qquad x_3 - x_2 \leq 8 \qquad x_4 - x_5 \leq 2$$

Solution: The corresponding graph of this system of inequalities is given by:



Upon inspection, we note that there are no negative cycles and thus has a feasible solution. To calculate a solution, we proceed with the Bellman-Ford algorithm:

Iteration	$\boldsymbol{x_0}$	$\boldsymbol{x_1}$	$\boldsymbol{x_2}$	$\boldsymbol{x_3}$	$\boldsymbol{x_4}$	$\boldsymbol{x_5}$
0	0, Null	∞, Null	∞, Null	∞, Null	∞, Null	∞, Null
1	0, Null	0, x_0	-5, x_4	0, x_0	-2, x_3	-2, x_1
2	0, Null	-2, x_2	-5, x_4	0, x_0	-2, x_3	-2, x_1
3	0, Null	-2, x_2	-5, x_4	0, x_0	-2, x_3	-4, x_1
4	0, Null	-2, x_2	-5, x_4	0, x_0	-2, x_3	-4, x_1
5	**0, Null**	$\boldsymbol{-2, x_2}$	$\boldsymbol{-5, x_4}$	**0,** $\boldsymbol{x_0}$	$\boldsymbol{-2, x_3}$	$\boldsymbol{-4, x_1}$

And so a solution is given by the following assignment of variables:

$$x_1 = -2 \qquad x_2 = -5 \qquad x_3 = 0$$
$$x_4 = -2 \qquad x_5 = -4$$

Problem 11

Problem: Are the following statements true or false?

a) To decide if a graph is 2-colorable is in P.

b) In any complete graph with nonnegative weights on edges, the weight of any minimum spanning tree is no larger than the weight of any Hamiltonian cycle.

c) Computing the minimum vertex cover of a graph G that is a tree is solvable in polynomial time.

d) To test if an integer n is not prime is in co-NP.

Solution:

a) True

b) True

c) True

d) True