Question 1. Network Flow: Short Questions [15 MARKS]

Let $G = (V, E, c)$ be a $s - t$ flow problem. Here $s, t \in V$ are the source and sink vertices, respectfully, and $c(e) > 0$ is the integer-valued capacity for each edge $e \in E$. (Other conditions on G are as specified in the lecture notes for $s - t$ flow problems.) In addition, suppose f is a feasible integer-valued flow on G and suppose $G_f = (V, E_f, c_f)$ is the associated residual graph.

Part (a) [2 MARKS]

For the above flow f and any **one** edge $e = (u, v) \in E$, draw the corresponding edge(s) in the residual graph G_f and indicate their capacities. (Include a comment to indicate if there are any special cases where an edge you have drawn is not actually in the residual graph.)

$c_f((u,v)) = c((u,v)) - f(e)$ $\quad (e = (u,v))$

$c_f(e^R) = f(e)$

" (v,u)

// Either edge is not in residual graph if $c_f(e)$ or $c_f(e^R)$ is zero.

Part (b) [2 MARKS]

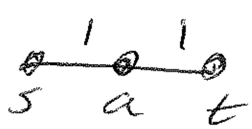
Given any s-t cut (A, B), define the net flow from A to B in terms of the flow $f(e)$ on edges in the cutset of A.

$$N_f(A) = \left[\sum_{\substack{e=(u,v)\in E \\ u\in A,\, v\in B}} f(e)\right] - \left[\sum_{\substack{e=(u,v)\in E \\ u\in B,\, v\in A}} f(e)\right]$$

Part (c) [2 MARKS]

Is it true that any s-t flow problem G has a unique minimum s-t cut (A, B)? If yes, briefly explain, otherwise give an example.

No.



$cap(\{s\}, \{a, t\}) = 1$

$= cap(\{s, a\}, \{t\})$.

Part (d) [3 MARKS]

Suppose P is a simple augmenting path in G_f with the bottleneck rate $b > 0$. Let g be the updated flow obtained in the standard way from f by using P and b. Let u be any vertex on P. Is s reachable from u in the residual graph, G_g, for the updated flow? Explain.

Yes, by working backwards from u along the path P, for example. Two types of edges on this path:

Case 1: $e = (u,v)$ is on P and $e \in E$

Then $g(e) = f(e) + b > 0$ so $e^R \in G_g$.

Case 2: $e^R = (u,v)$ is on P and (v,u) $\cancel{(u,v)} \in E$.

Then $g(e) = f(e) - b < f(e) \leq c(e)$.

Therefore $e \in G_g$.

In both cases $\cancel{(u,v) \in G_g}$ $(v,u) \in G_g$. So reverse P is in G_g.

Part (e) [3 MARKS]

Assume f is a maximal flow and $D \equiv \{w \mid w \in V, \text{ and } t \text{ is reachable from } w \text{ in the residual graph } G_f\}$. What can you say about the flow on any edge $e = (u, v) \in E$ such that $u \notin D$ but $v \in D$? Explain.

$u \notin D \Rightarrow t$ not reachable from u in G_f

$v \in D \Rightarrow t$ is " " v " "

$\therefore \underline{(u,v) \notin G_f}$ but $\underline{(u,v) \in E}$

Can only happen if $f((u,v)) = c((u,v))$

i.e. edge e is at capacity.

Part (f) [3 MARKS]

Suppose f is a feasible flow for the s-t flow problem (G, V, c), and supose $C \subseteq V$ is **any subset** of the vertices. Give an expression for the net flow from C to $V \backslash C$, where this expression involves only the value of the flow, $\nu(f)$, and properties of C (i.e., without directly referring to the flow $f(e)$ on individual edges $e \in E$). State the key property from the definition of network flows that you could use to prove this result (although you do not prove your result here nor explain how you derived it).

Net flow C to $V \setminus C$

$$\equiv \hat{N}_f(C) = v(f)\,\delta(s \in C) - v(f)\,\delta(t \in C)$$

Here $v(f) \equiv$ value of flow (as above).

$\delta(\text{true}) = 1$

$\delta(\text{false}) = 0$.

Used flow conservation at all vertices except s ~~m~~ and t.

Flow out of s is $v(f)$.

Turns out, flow *into* t must be $v(f)$.

Question 2. Dynamic Programming: Longest Abs-Three Subsequence [15 MARKS]

Suppose we are given a list of n integers, say $X = (x(1), \dots, x(n))$. The problem is to find the longest subsequence of X, say $S = (x(j_1), x(j_2), \dots, x(j_K))$, where $|x(j_{k+1}) - x(j_k))| = 3$ for each $k = 1, 2, \dots, K-1$. Note that for S to be a subsequence we must have $1 \le j_1 < j_2 < \ldots < j_K \le n$, that is, the elements of S must be chosen in the same order they appear in X.

For example, if $X = (1, 5, 3, 4, 2, 5, -1, 2)$ then $S = (1, 4)$ and $S = (5, 2, 5)$ are possible subsequences, but $S = (5, 2, -1, 2)$ is the longest subsequence with the desired "abs-three" property.

Part (a) [10 MARKS]

Use dynamic programming approach that, given any input sequence X, solves for the maximum possible length, $|S|$, of such a subsequence. You should explain: i) what the terms in your recurrence relation refer to; ii) why your recurrence relation is correct; and iii) what the maximum possible length is in terms of the solution of your recurrence relation. (You do not need to formally prove anything.)

Your method must use at most $O(n)$ space and run in $O(n^2)$ time.

Define $\mathcal{O}(k)$ = length of longest such subsequence ending with $j_{\mathcal{O}(k)} = k$. i.e. current sequence ends by selecting x_k.

$k \in \{1, \dots, n\}$

Define $\mathcal{O}(0) = 0$.

$$\mathcal{O}(j) = \max\{\{\mathcal{O}(k) + 1 \mid 1 \le k < j \text{ and } |x(k) - x(j)| = 3\} \cup \{\mathcal{O}(0)\}\}$$

~~Define max{empty set}~~

~~Define max {} = 0~~

included on max.

This is correct since the longest sequence ending at $x(j)$ must be one longer than the longest ending at some suitable $x(k)$, for $k < j$, where "suitable" means $|x(k) - x(j)| = 3$.

Compute $\mathcal{O}(i)$ for $i = 0, \dots, n$ (in that order).

Max possible length $|S| = \max_{1 \le k \le n} \mathcal{O}(k)$

Part (b) [5 MARKS]

Suppose that, from your solution in part (a) above, you only kept the table of optimal scores and not any additional information. Explain how you could recover an optimal subsequence S directly from such a table of scores and the original sequence X. This algorithm must also run in at most $O(n^2)$ time.

$k \leftarrow \arg\max_{1 \le k \le n} \{ \mathcal{O}(k) \}$

$S \leftarrow \{k\}$

$L \leftarrow \mathcal{O}(k) - 1$ // Length of prefix yet to be found.

while $L > 0$:

$\quad k \leftarrow \text{find} \{ j \mid 1 \le j < k, \mathcal{O}(j) == L,$ and $|x(j) - x(k)| == 3 \}$

$\quad S \leftarrow S.\text{prepend}(k)$

$\quad L \leftarrow L - 1.$