COMP 360 - Winter 2016 - Assignment 2

Due: 6pm Feb 10th.

General rules: In solving these questions you may consult books but you may not consult with each other. You should drop your solutions in the assignment drop-off box located in the Trottier Building.

"Is this going to be on the midterm?"

---

1. (10 Points) Consider the following two algorithms for finding the maximum flow:

Algorithm 1: Scaling max-flow

```
• Initially set $f(e) := 0$ for all edges $e$.
• Set $\Delta$ to be $\max c_e$ rounded down to a power of 2.
• While $\Delta \geq 1$:
•       While there is an $s,t$-path $P$ in $G_f(\Delta)$:
•             Augment the flow using $P$ and update $G_f(\Delta)$.
•       Endwhile.
•       Set $\Delta := \Delta/2$.
• Endwhile.
• Output $f$.
```

Algorithm 2: The fattest path algorithm

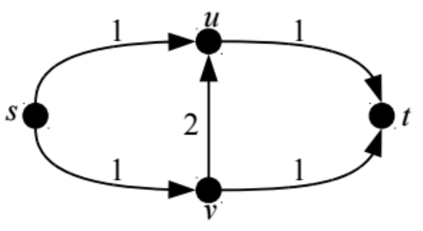
```
• Initially set $f(e) := 0$ for all edges $e$.
• While there exists an $s,t$-path in $G_f$:
•       Augment the flow using the fattest $s,t$-path $P$ in $G_f$.
•       Update $G_f$.
• Endwhile.
• Output $f$.
```

In the second algorithm the fattest means the largest bottleneck. From the class we know that the number of augmentations in Algorithm 1 is at most $2m\lceil\log_2 K\rceil$, where K is the maximum capacity of an edge. Deduce from this that the number of augmentations in Algorithm 2 is also at most $2m\lceil\log_2 K\rceil$.

Solution (sketch): The paths found in Algorithm 2, can also be used in Algorithm 1. Indeed to verify whether there is a path with Bottleneck$(P, f) \geq \Delta$, it suffices to consider the fattest path P.

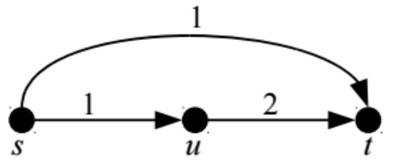
2. (35 points) A *key edge* of a flow network is an edge whose deletion causes the largest drop in the value of the maximum flow. Let f be an arbitrary maximum flow. True of False (Prove or Disprove)?

   (a) A key edge is always an edge with maximum capacity.
   Solution: FALSE. Consider the following flow network.



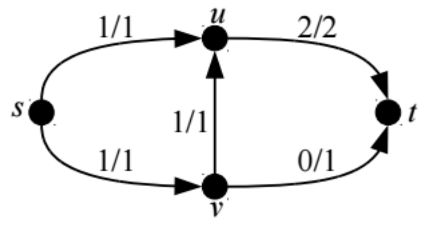
   In this figure the edge (v, u) has the maximum capacity, but it is not a key edge because removing (v, u) does not decrease the value of the maximum flow.

   (b) Deletion of a key edge e decreases the value of the maximum flow by the capacity of e.
   Solution: FALSE. Consider the following flow network.



   Every edge in the network is a key edge. But, removing an edge (v, t) does not decrease the value of the maximum flow by 2.

   (c) Deletion of a key edge e decreases the value of the maximum flow by $f(e)$.
   Solution: FALSE. Consider the following flow network.



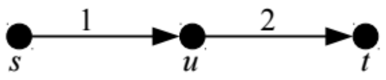
   In this figure, the number x/y on each edge e means $f(e) = x$ and $c(e) = y$. The edge (u, t) is a key edge and has $f(u, t) = 2$, and the maximum flow of this network is 2. However, removing (u, t) only decreases the value of the maximum flow by one. This is because we can still send a flow of value 1 along a path (s, v, t).

   (d) A key edge e in a minimum cut (A, B) has the maximum value of f among edges belonging to that cut (edges from A to B).

Solution: TRUE. This is because any edge e in the minimum cut (A, B) must be saturated, i.e., $c(e) = f(e)$. Since e is in the minimum cut (A, B), removing e decreases the size of the minimum cut by $c(e)$; thus, the value of the maximum flow also decreases by $c(e) = f(e)$.

(e) An edge that does not belong to some minimum cut is not a key edge.

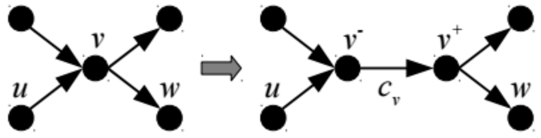
Solution: False. Consider the following flow network.



Every edge in this network is a key edge, but (u, t) is not in any minimum cut.

3. (15 points) Use the Max-Flow problem to solve the following problem. The input is a number k and a *bipartite* graph $G = (V, E)$ where some of the edges are colored red. We would like to find out whether we can color more edges red (if necessary) so that every vertex is incident to exactly k red edges.

Solution: We can deduce our problem to a variant of the maximum flow problem where every node v also has an integer capacity $c_v \geq 0$. We are interested in finding the maximum flow as before, but now with the extra restriction that $f^{in}(v) \leq c(v)$. First let us show how to solve this auxilary probelm using ordinary max flow algorithm.



We construct an auxiliary network N' by replacing each node v by v^+ and v^- and add an edge from v^+ to v^- with capacity c_v. We move heads of edges entering v to v^- and move tails of edges leaving v to v^+. That is, we replace each edge (u, v) by (u, v^-) and replace each edge (v, w) by (v^+, w). Then we find max flow in N'; it is the same as max flow in the original network N.

Now let's show how to deduce our original problem to this auxilary problem. Add a source s and connect it to all the vertices in A with edges of infinite capacity. Add a sink t and connect all the vertices in B to t with edges of infinite capacity. For every vertex other than s and t let $c(v)$ be k minus the number of pre-colored red edges. Let s and t have infinite capacities and all the remaining edges have capacity one. It is easy to see that if the max flow in this network is exactly $\sum_{v \in A} c(v)$ then there is such an extension of coloring.

4. (20 points) Show that for every flow network, there exists an execution of the Ford-Fulkerson algorithm that never decreases the value of the flow on any of the edges (i.e. never "pushes back" the flow on any of the edges).

Solution: Given a flow f in the network here is such an algorithm.

(1) Find an s, t-path P with positive flow, i.e., $f(e) > 0$ for all $e \in P$.

(2) Let $\Delta = \min_{e \in P} f(e)$. We construct a flow f_P by setting $f_P(e) = \Delta$ for all $e \in P$ and $f_P(e) = 0$ for all other edges. Then we decrease the flow f on each edge $e \in P$ by Δ.

(3) Repeat Step (1) and (2) until there is no such s, t-path left.

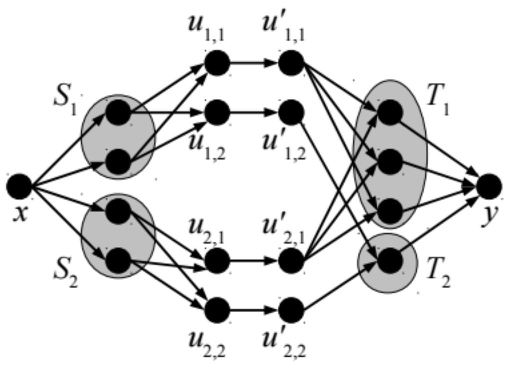
Each time we decrease the flow f on at least one edge to zero. Thus, the algorithm terminates in at most m times. So, we have a set P of at most m flow-paths. The flows f_{P_i} together with the corresponding paths P_i satisfy the requirement (i.e. if the algorithm chose these paths during the execution).

5. (20 points) Suppose that the set $\{1, \ldots, m\}$ is partitioned into disjoint sets $S_1, \ldots, S_k$. Also let $a_1, \ldots, a_t$ be positive integers with $a_1 + \ldots + a_t = m$. Given $S_1, \ldots, S_k$ and $a_1, \ldots, a_t$, construct a flow network with the following property: The network has maximum flow m if and only if there is a partition $T_1, \ldots, T_t$ of $\{1, \ldots, m\}$ such that

   (a) $|T_i| = a_i$ for all $i = 1, \ldots, t$;

   (b) The elements of every S_i belong to different T_j's (That is if $a, b \in S_i$ for some i, then $a \in T_x$ and $b \in T_y$ for some $x \neq y$).

   (Clarification: This is one question. Both conditions (a) and (b) must hold at the same time.)
   Solution: We construct the following network. First add a source node x and a sink node y. For each $i \in [m]$ create a node and join x to all of them. For each T_j, create a_j vertices and connect all these vertices to y. For each pair S_i and T_j, create two nodes $u_{i,j}$ and $u'_{i,j}$ and add an edge from $u_{i,j}$ to $u'_{i,j}$. Set all the capacities to one. It is easy to check that this network is as desired.



$u_{1,1}$
$u'_{1,1}$
S_1
T_1
$u_{1,2}$
$u'_{1,2}$
x
y
$u_{2,1}$
$u'_{2,1}$
S_2
T_2
$u_{2,2}$
$u'_{2,2}$