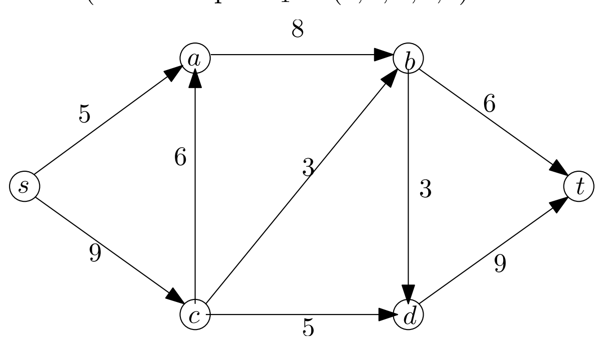
COMP 360 - Winter 2014 - Assignment 1

Due: 6pm Jan 31st.

General rules: In solving these questions you may consult books but you may not consult with each other. There are in total 120 points, but your grade will be considered out of 100. You should drop your solutions in the assignment drop-off box located in the Trottier Building.

1. (15 Points) Consider the following maximum flow problem, and apply the fattest path algorithm to find a maximum flow. List the augmenting paths used by the algorithm and their bottlenecks in the right order (For example $P_1 = (s, c, a, b, t)$ and Bottleneck(P_1) = 6).



2. (10 Points) Consider the following three algorithms for finding the maximum flow:

Algorithm 1: Scaling max-flow

- Initally set $f(e) := 0$ for all edges e.
- Set Δ 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:

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

Algorithm 3: The fattest path algorithm

- Initally 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.

(a) 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 $2m\lceil \log_2 K \rceil$.

(b) Explain why Algorithm 2 and Algorithm 3 are essentailly the same algorithms (Hence the the number of augmentations in the fattest path algorithm is also at most $2m\lceil \log_2 K \rceil$).

3. (15 Points) Consider a minimum cut (A, B) in a flow network with exactly two edges e_1 and e_2 going from A to B.

   (a) Prove: Decreasing the capacity of e_1 by 1 and meanwhile increasing the capacity of e_2 by 1 cannot increase the value of the maximum flow.

   (b) Disprove: Decreasing the capacity of e_1 by 1 and meanwhile increasing the capacity of e_2 by 1 cannot decrease the value of the maximum flow.

4. (10 points) Recall that for every flow f and every cut (A, B), we have $\text{val}(f) = f^{out}(A) - f^{in}(A)$ where $f^{out}(A) = \sum_{\substack{uv \in E \\ u \in A, v \in B}} f(uv)$ and $f^{in}(A) = \sum_{\substack{uv \in E \\ u \in B, v \in A}} f(uv)$. Use this fact to show that $\text{val}(f)$ is equal to the total flow on the edges going into the sink.

5. (10 points) Prove that if an edge e goes from A to B in a minimum cut (A, B), then every *maximum* flow f uses the full capacity of e, that is $f(e) = c_e$.

6. (20 points) Let $(G, s, t, \{c_e\})$ be a flow network that satisfies the following property. For every edge e in G, there exists at least one minimum cut (A, B) such that e goes from A to B. Prove that G does not contain any directed cycles.

7. (20 points) Let $(G, s, t, \{c_e\})$ be a flow network, and let F be the set of all edges e for which there exists at least one minimum cut (A, B) such that e goes from A to B. Give a polynomial time algorithm that finds all edges in F.

8. (Bonus: 20 Points) Consider the 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 $f^{\text{in}}(v) \leq c_v$ for every node v. Solve this problem using the original maximum flow problem.