CS/ECE 473 Fall 2020 — Ryan Prendergast (ryanp4)
**Homework 8 Problem 2** — Noah Watson (nwatson3)
Lawson Probasco (lawsonp2)

Let $G = (V, E)$ be a flow network with integer edge capacities and let $s, t \in V$ be two distinct nodes.

(a) An edge $e$ is called critical if $e$ is in every minimum $(s, t)$-cut. Describe an efficient algorithm that checks whether a given edge e is critical or not.

**Solution:** First we must check if $e$ is in any min-cut at all. We start by running Orlin's max flow algorithm on $G$. This gives us a minimum cut capacity $c$. Then we modify $G$ into $G'$ by deleting the edge $e$ (set its capacity to zero). We run Orlin's max flow on this graph $G'$, getting a cut capacity $c'$. If the graph's cut capacity does not decrease by at least the capacity of $e$, that is if $c - c' < w(e)$, then $e$ is not in a min-cut. Return false.

Next we must see if $e$ is in every min cut. We modify $G$ into a new $G''$, changing the edge capacity of $e$ to $\infty$. We run Orlin's max flow algorithm on $G''$, getting a new capacity $c''$. If $c'' \leq c$, then there must be a minimum s,t cut that does not include the edge $e$. Thus, $e$ is not critical. Otherwise, $e$ is a critical edge.

**Runtime:** We run Orlin's algorithm three times, on G and $G'$ and $G''$. $G'$ and $G''$ are constant time modificationn of $G$ (we just change one edge capacity). So, our total runtime is $O(VE)$. ■

(b) Describe an efficient algorithm that given $G, s, t$ checks whether there is a G has a unique $s - t$ minimum cut.

**Solution:** $G$ has a unique minimum cut iff every edge in a minimum cut of $G$ is critical. If we have a minimum cut that does not include a particular edge $e_0$, but that edge is included in another minimum cut of the same capacity, that other minimum cut must exclude one or more of the edges in our initial cut, as all edges are positive.

So, we must start by finding a minimum cut of $G$. Using Orlin's algorithm to find a maximum flow in $O(VE)$ time, we can find the edges that make up the minimum capacity cut in linear time $O(V + E)$. See UMass CS312 Slide 17.

Then, we run the algorithm in part (a) on each of the edges in the minimum cut. If all of them are critical, then $G$ has a unique $s, t$ min cut. Return True. If one or more is not critical, $G$ does not. Return False.

**Runtime:** To find the minimum cut edges, we spend $O(VE + V + E) = O(VE)$ time. Then, we run the $O(VE)$ algorithm from (a) once on each of the cut edges (bounded by $E$). This runs in $O(VE^2)$ time. Since this term dominates the previous, our total runtime is $O(VE^2)$. ■