CMPUT 675: Combinatorial Optimization **Winter 2021**

Lecture 9 (Feb 8): Min-Cost Flows via Min-Ratio Cycle Cancelling

Lecturer: Zachary Friggstad *Scribe: Saeed Najafi*

9.1 Minimum Mean Cycle-Cancelling Algorithm

In the previous lecture, we defined the minimum-cost Flow problem and designed the Successive Shortest Paths algorithm which is similar to Ford-Fulkerson by augmenting along a min-cost path in its residual graph. The algorithm maintains the optimal criterion that there is no negative cost cycle at any stage. This lecture introduces the Minimum Mean Cycle Canceling Algorithm and establishes a polynomial time complexity for the min-cost Flow problem. The provided proofs follow our main text book [1]. We begin the lecture by the following lemma.

Lemma 1 *Consider the real values $a_1, a_2, ..., a_k \in \mathbb{R}$ and $b_1, b_2, ..., b_k \in \mathbb{R}_{>0}$, then* $\min_i \frac{a_i}{b_i} \leq \frac{\sum_i a_i}{\sum_i b_i}$.

Proof. Let $p_i = \frac{b_i}{\sum_j b_j}$. So $0 < p_i$ and $\sum_i p_i = 1$. It is clear that the minimum value of a set of numbers is smaller than any weighted mean. Therefore, $\min_i \frac{a_i}{b_i} \leq \sum_i p_i \times \frac{a_i}{b_i} = \sum_i \frac{b_i}{\sum_j b_j} \times \frac{a_i}{b_i} = \frac{1}{\sum_j b_j} \times \sum_i a_i = \frac{\sum_i a_i}{\sum_j b_j}$. ■

9.1.1 Finding Min Ratio Cycle

In the directed graph $\mathcal{G} = (V, E)$ with the edge costs $c : E \to \mathbb{R}$ (cost of an edge can be negative), we are interested to find a cycle $\mathcal{C}$ minimizing $\frac{c(\mathcal{C})}{|\mathcal{C}|}$. Although computing the minimum-cost cycle generally is an NP hard problem if there are negative edge costs in the graph, the following Theorem based on the Bellman-Ford algorithm computes the min-ratio cycle in a polynomial time.

Theorem 1 *We can find a cycle $\mathcal{C}$ with the minimum ratio $\frac{c(\mathcal{C})}{|\mathcal{C}|}$ in time $O(mn^2)$.*

Proof of Theorem 1. For $1 \leq k \leq n$, define $\delta_k(u, v)$ to be the cost of the cheapest $u - v$ walk using exactly k (not necessarily distinct) edges. If there is no such walk, let $\delta_k(u, v) = \infty$. The general recurrence rule used in the Bellman Ford algorithm is $\delta_k(u, v) = \min_{wv \in E} c(w, v) + \delta_{k-1}(u, w)$ for $k > 0$ (we compute this recursion for all possible source nodes). For $k = 0$, the base case is $\delta_k(u, v) = 0$ if $u = v$, otherwise ∞. Computing all of these values can be done in $O(mn^2)$ time: for each k and each u the total number of "iterations" of the recurrence for calculating $\delta_k(u, v)$ terms is bounded by m. To complete the proof, we specify the following Lemma having access to the table values $\delta_k(u, v)$.

Lemma 2 *Let u^* and k^* minimize $\frac{\delta_k(u,u)}{k}$; among all such minimizers choose k^* to be the smallest. Then the walk from u^* to itself in k^* steps must be a cycle, and thus is a minimum ratio cycle.*

Proof. Let $\mathcal{C}$ be a min-ratio cycle and let u' be any node on $\mathcal{C}$. Then $\frac{\delta_{|\mathcal{C}|}(u', u')}{|\mathcal{C}|} \leq \frac{c(\mathcal{C})}{|\mathcal{C}|}$ since $\mathcal{C}$ gives a walk with the length $|\mathcal{C}|$ from u back to u. Therefore, whatever values of u and k that minimize $\frac{\delta_k(u,u)}{k}$, it won't be worse than the ratio of the min-ratio cycle.

Now suppose k^*, u^* minimize $\frac{\delta_k(u,u)}{k}$, and W is the corresponding tour (walk). As we proved in the assignment 2, we can decompose W into a disjoint union of simple cycles $W = \cup_i \mathcal{C}_i$. According to the Lemma 1, we have $\min_i \frac{c(\mathcal{C}_i)}{|\mathcal{C}_i|} \leq \frac{\sum_i c(\mathcal{C}_i)}{\sum_i |\mathcal{C}_i|} = \frac{\delta_{k^*}(u^*,u^*)}{k^*}$. However, according to the definition, among all the possible minimizers of $\frac{\delta_k(u,u)}{k}$, we choose k^* to be the smallest.

Since W is a simple cycle with $\frac{c(W)}{|W|} = \frac{\delta_{k^*}(u^*,u^*)}{k^*} \leq \frac{\delta_{|\mathcal{C}|}(u,u)}{|\mathcal{C}|} = \frac{c(\mathcal{C})}{|\mathcal{C}|}$ and $\mathcal{C}$ was a minimum-ratio cycle, then W is also a minimum ratio cycle.

■ ■

9.1.2 The Algorithm

Let $\mathcal{G} = (V, E)$ be a directed graph with edge costs $c : E \to \mathbb{R}$ and capacities $\mu : E \to \mathbb{R}_{\geq 0}$. We want to find a minimum-cost $s - t$ max flow. We already know that a flow f has minimal cost if $\mathcal{G}_f$ contains no negative-cost cycles. This suggests that a strategy for finding minimal cost max flows is obtained from successively augmenting along negative-cost cycles from $\mathcal{G}_f$ until there are none left. In the Mean Cycle-Cancelling Algorithm, we will select the negative cycles with the minimum ratio according to method described by the Theorem 1. The procedure is summarized in the following algorithm.

Algorithm 1 The Minimum Mean Cycle Canceling Algorithm

Input: Directed graph $\mathcal{G} = (V, E)$ with edge costs $c : E \to \mathbb{R}$ and capacities $\mu : E \to \mathbb{R}_{\geq 0}$, $s, t \in V$ $(s \neq t)$.
Output: A minimum-cost $s - t$ max flow f.

```
f ← s − t max flow (Found in O(mn^2) with the push-relabel).
while G_f contains negative weight cycles (using Bellman Ford) do
    C ← a cycle in G_f which minimizes its mean cost c(C)/|C| (using Theorem 1)
    Augment f along C
return f
```

The algorithm certainly terminates, since the cost of f decreases for every iteration of the algorithm, at a rate bounded by the rational values of the edge costs. The following theorem provides the running time analysis of the algorithm.

Theorem 2 *With the integer edge costs, the number of iterations of the Minimum Mean Cycle Cancelling Algorithm is $O(m.n.\log(nc_{max}))$ where c_{max} is the maximum edge cost in the graph.*

Proof of Theorem 2. Let $f_0, f_1, f_2, ..., f_k$ $(f = f_0)$ be the flows in each iteration of the algorithm, together with the cycles $\mathcal{C}_0, \mathcal{C}_1, ..., \mathcal{C}_{k-1}$ as the min-ratio cycles in the residual graphs $G_{f_0}, G_{f_1}, ..., G_{f_{k-1}}$, respectively. Let $r_i = \frac{c(\mathcal{C}_i)}{|\mathcal{C}_i|}$ for $0 \leq i \leq k-1$. We now introduce the following two Lemmas:

Lemma 3 *The ratios never decrease: $r_i \leq r_{i+1}$ $(0 \leq i \leq k-1)$, so these negative ratios go up to zero throughout the algorithm.*

Proof. Let define the graph $\mathcal{H} = (V, \mathcal{C}_i \cup \mathcal{C}_{i+1} - \{e, \overleftarrow{e} | e, \overleftarrow{e} \in \mathcal{C}_i \cup \mathcal{C}_{i+1}\})$ (edges appearing both in C_i and C_{i+1} are included twice). We know that $E(H) \subseteq E(G_{f_i})$ (i.e. every simple sub-graph of $\mathcal{H}$ is also a sub-graph of G_{f_i}) because each edge in $E(G_{f_{i+1}}) \setminus E(G_{f_i})$ must be the reverse of an edge in $E(\mathcal{C}_i)$ so if such an edge was on $\mathcal{C}_{i+1}$ would be excluded from the edges of H.

By the cycle decomposition argument (proved in assignment 2) and the Lemma 1, the minimum cycle ratio at iteration i must be smaller than the ratio of $E(\mathcal{H})$: $r_i \leq \frac{c(E(\mathcal{H}))}{|E(\mathcal{H})|}$. We know that $|\mathcal{C}_i| + |\mathcal{C}_{i+1}| \geq |E(\mathcal{H})|$ and $r_i < 0$, so we have $r_i \times (|\mathcal{C}_i| + |\mathcal{C}_{i+1}|) \leq r_i \times |E(\mathcal{H})| \leq c(E(\mathcal{H}))$. Since the total weight of each pair of reverse edges is zero, we also have $c(E(\mathcal{H})) = c(\mathcal{C}_i) + c(\mathcal{C}_{i+1}) = r_i \times |\mathcal{C}_i| + r_{i+1} \times |\mathcal{C}_{i+1}|$. Finally, we have $r_i \times (|\mathcal{C}_i| + |\mathcal{C}_{i+1}|) \leq r_i \times |\mathcal{C}_i| + r_{i+1} \times |\mathcal{C}_{i+1}|$. We can conclude that $r_i \leq r_{i+1}$. ∎

Lemma 4 *For all $i < j$ such that $\mathcal{C}_i \cup \mathcal{C}_j$ contains an edge and its reverse, then $r_i \leq \frac{n}{n-2} \times r_j$.*

So these ratios make a noticeable leap towards zero.

Proof. Consider $i < j$ such that for $i < k < j$, $\mathcal{C}_i \cup \mathcal{C}_k$ does not contain any pair of reverse edges, but only $\mathcal{C}_i \cup \mathcal{C}_j$ contains an edge and its reverse (j is the min index of such indices). Consider the graph $\mathcal{H} = (V, \mathcal{C}_i \cup \mathcal{C}_j - \{e, \overleftarrow{e} | e, \overleftarrow{e} \in \mathcal{C}_i \cup \mathcal{C}_j\})$. Similar to the argument presented in the previous proof, every simple sub-graph of $\mathcal{H}$ is a sub-graph of G_{f_i}. Similarly by cycle decomposition, we have $r_i \leq \frac{c(E(\mathcal{H}))}{|E(\mathcal{H})|}$ and $c(E(\mathcal{H})) = c(\mathcal{C}_i) + c(\mathcal{C}_j) = r_i \times |\mathcal{C}_i| + r_j \times |\mathcal{C}_j|$. We also know that $|\mathcal{C}_i| + |\mathcal{C}_j| - 2 \geq |E(\mathcal{H})|$ (we removed the pair of reverse edges). Since $|\mathcal{C}_j| \leq n$, we have $|\mathcal{C}_i| + \frac{n-2}{n} \times |\mathcal{C}_j| \geq |\mathcal{C}_i| + |\mathcal{C}_j| - 2$ (as $|\mathcal{C}_j| \leq n$). Therefore, we have $r_i \times (|\mathcal{C}_i| + \frac{n-2}{n} \times |\mathcal{C}_j|) \leq r_i \times |E(\mathcal{H})| \leq c(E(\mathcal{H})) = r_i \times |\mathcal{C}_i| + r_j \times |\mathcal{C}_j|$. We can conclude that $r_i \leq \frac{n}{n-2} \times r_j$. ∎

To finish the proof of the Theorem, we claim that $r_i \leq 2 \times r_{i+m.n}$. Within the m consecutive min-ratio cycles $\mathcal{C}_i, \mathcal{C}_{i+1}, \mathcal{C}_{i+2}, ..., \mathcal{C}_{i+m}$, there must be $\mathcal{C}_k$ and $\mathcal{C}_j$ $(i \leq k < j \leq i+m)$ whose union contains a pair of reverse edges. According to the Lemmas 3 & 4, we have $r_i \leq r_k \leq \frac{n}{n-2} \times r_j \leq \frac{n}{n-2} \times r_{i+m}$. Therefore in every $n.m$ iterations, r_i decreases by at least a factor of $(\frac{n}{n-2})^n$. It is also clear that $(\frac{n}{n-2})^n > e^2 > 2$. We can conclude that $r_i \leq 2 \times r_{i+m.n}$.

For the initial ratio r_0, we have $-c_{max} \leq r_0 \leq 2^a \times r_{a.m.n}$. If we pick $a = \lceil n.c_{max} \rceil + 1$, then $r_{a.m.n} > \frac{-1}{n}$. Since edge costs are integer, then $r_{a.m.n} \geq 0$. Therefore, the algorithm terminates after $O(m.n.\log(nc_{max}))$ iterations. ∎

Theorem 3 *Without the assumption of having integer edge costs, the Goldberg and Tarjan's analysis proves that the Minimum Mean Cycle-Cancelling Algorithm runs in at most $O(m^2.n.\log(n))$ iterations.*

Proof of Theorem 3. We show that in every $m.n.(\lceil \log(n) \rceil + 1)$ iterations at least one edge is fixed, i.e. the flow on this edge changes for the last time. Therefore there are at most $O(m^2.n.\log(n))$ iterations.

Let f be the flow at the iteration i, and let f' be the flow $k = m.n.(\lceil \log(n) \rceil + 1)$ iterations later. Define the new costs $c'(e) = c(e) - r_{i+k}$ $(\forall e \in E(G_{f'}))$ that are shifted up by r_{i+k}. Under these new edge costs, we won't have a negative cost cycle in $G_{f'}$ (Proved in Assignment 3), so there is a feasible potential ϕ for $(G_{f'}, c')$. We then have $0 \leq c'_\phi(e) = c_\phi(e) - r_{i+k}$, so $c_\phi(e) \geq r_{i+k}$ $(\forall e \in E(G_{f'}))$. Now let $\mathcal{C}_i$ be the circuit of minimum ratio in G_f that is chosen in the algorithm to augment f. From the argument presented in the Theorem 2, we have $r_i \leq 2^{\lceil \log n \rceil + 1} \times r_{i+k} \leq 2n \times r_{i+k}$.

As the vertex potentials cancel out each other along a cycle, we have $\sum_{e \in \mathcal{C}_i} c_\phi(e) = \sum_{e \in \mathcal{C}_i} c(e) = r_i \times |\mathcal{C}_i| \leq 2.n \times r_{i+k} \times |\mathcal{C}_i|$. Therefore, $\exists e' \in \mathcal{C}_i$ such that $c_\phi(e') \leq 2.n \times r_{i+k}$. Earlier we proved that $c_\phi(e) \geq r_{i+k}$ $(\forall e \in E(G_{f'}))$, so $e' \notin G_{f'}$ and $e' \notin \mathcal{C}_{i+k}$. We complete the proof by showing that e' never appears again in the later iterations $\forall j > i+k$.

Suppose that the opposite is correct and e' re-appears in the residual graph of the future flow f'' at the iteration $j > i+k$ $(e' \in E(G_{f''}))$. According to the Proposition 9.4 in [1], we could build a Fully Conservative Flow g in $G_{f'}$ with $g(\overleftarrow{e'}) > 0$ (as $e' \in E(G_{f''}) \setminus E(G_{f'})$). If we decompose G_g into cycles, there exists $\overline{\mathcal{C}}$ that contains $\overleftarrow{e'}$, then $c_\phi(\overleftarrow{e'}) = -c_\phi(e') \geq -2n \times r_{i+k}$. For the rest of edges in $\overline{\mathcal{C}}$, we have $c_\phi(e) \geq r_{i+k}$ as proved earlier

$(\forall e \in E(G_{f'}))$. Therefore, $c(\overline{\mathcal{C}}) = \sum_{e \in \overline{\mathcal{C}}} c_\phi(e) \geq -2n \times r_{i+k} + (|\overline{\mathcal{C}}| - 1) \times r_{i+k} > -n \times r_{i+k}$. But the reverse of $\overline{\mathcal{C}}$ is an $f^{''}$-augmenting cycle, and its total weight is less than $n \times r_{i+k}$. This means that $G_{f''}$ contains a circuit whose mean weight is less than r_{i+k} which contradicts the Lemma 3 as $j > i + k$.

We can conclude that $e^{'}$ never appears again in the later iterations. So in every $m.n.(\lceil \log(n) \rceil + 1)$ iterations at least one edge is fixed, and the algorithm terminates in at most $O(m^2.n. \log(n))$ iterations. ■

References

[1] B. Korte and J. Vygen, *Combinatorial Optimization: Theory and Algorithms*. Springer Publishing Company, Incorporated, 6th ed., 2012.