ORIE 633 Network Flows October 23, 2007

Lecture 14

Lecturer: David P. Williamson *Scribe: Abhimanyu Mitra*

1 Algorithms for minimum-cost circulations

1.1 A cost-scaling algorithm

We now turn to an algorithm for the minimum-cost circulation problem.

Cost Scaling (Goldberg, Tarjan '90)

Let f be any feasible circulation
Initialize $\epsilon \leftarrow C$, $p_i \leftarrow 0 \ \ \forall i \in V$
while $\epsilon \geq \frac{1}{n}$
    $(\star)$
    $\epsilon \leftarrow \frac{\epsilon}{2}$
    $(f, p) \leftarrow$ Run Subroutine: find ϵ-optimal circulation given input (f, ϵ, p)
    Find potentials p such that f is $\epsilon(f)$ optimal.

The idea is that given a 2ϵ-optimal circulation f with respect to potentials p, the subroutine will find an ϵ-optimal circulation f' with respect to potentials p'. Since the initial circulation is C-optimal and the final circulation is $< \frac{1}{n}$-optimal (and hence optimal by a lemma in a previous lecture), we will need at most $\log(nC)$ iterations of the while loop.

We can also show that the number of iterations is strongly polynomial by tweaking one of our previous theorems. Recall the following definition and result.

Definition 1 *An arc (i, j) is* ϵ-fixed *if the flow on (i, j) is the same for all ϵ-optimal circulations.*

Theorem 1 *For $\epsilon > 0$ and circulation f with respect to potentials p, if $|c^p_{ij}| \geq 2n\epsilon$, then (i, j) is ϵ-fixed.*

We then have the following theorem:

Theorem 2 *For circulation f and $\epsilon' < \frac{\epsilon(f)}{2n}$, the set of ϵ'-fixed arcs strictly contains the set of $\epsilon(f)$-fixed arcs.*

Proof: Clearly if an arc is ϵ'-fixed, then it is also $\epsilon(f)$-fixed. We now want to show that there exists an arc that is ϵ'-fixed, but not $\epsilon(f)$-fixed. Let p be the potentials such that f is $\epsilon(f)$-optimal. Then there exists a cycle $\Gamma \in A_f$ such that $-\epsilon(f) = \frac{c^p(\Gamma)}{|\Gamma|}$ by a previous theorem. We also know that $c^p_{ij} \geq -\epsilon(f) \, \forall (i, j) \in A_f$ by definition. Hence $c^p_{ij} = -\epsilon(f) \, \forall (i, j) \in \Gamma$.

If we cancel cycle Γ, the resulting circulation, $\hat{f}$, is still ϵ-optimal. Thus no arc in Γ is ϵ-fixed. Now let f' be any ϵ'-optimal circulation with respect to potentials p'. Then $-\epsilon(f) = \frac{c^{p'}(\Gamma)}{|\Gamma|} < -2n\epsilon'$ and thus $\exists (i,j) \in \Gamma$ such that $c_{ij}^{p'} \leq -2n\epsilon'$. Therefore (i,j) is ϵ'-fixed but not $\epsilon(f)$-fixed. □

We now want to claim the following corollary.

Corollary 3 *Every* $\log(2n)$ *iterations of the while loop, a new arc is fixed.*

But note that the lemma states that ϵ' must be a factor of $2n$ less than $\epsilon(f)$, not just any ϵ such that f is ϵ-optimal. In order to make this true, at step $(\star)$ in the Cost Scaling algorithm, we must add a subroutine to find potentials p such that f is $\epsilon(f)$-optimal and then set $\epsilon \leftarrow \epsilon(f)$. This will only decrease ϵ as the procedure continues. Then we can claim the corollary above.

Since we can fix at most m arcs, we have the following theorem.

Theorem 4 *After* $\min(m\log(2n), \log(nC))$ *iterations, Cost Scaling finds a min-cost circulation.*

Now, we'll give an algorithm for the subroutine find-ϵ-opt-circ(f, ϵ, p) based on the ideas from the push/relabel algorithm that we saw for the maximum flow problem.

find-ϵ-opt-circ

- **Input:** 2ϵ-opt circulation f, potentials p s.t. $c_{ij}^{p} \geq -2\epsilon, \forall (i,j) \in A_f$
- **Goal:** 2ϵ-opt circulation f', potentials p' s.t. $c_{ij}^{p'} \geq -\epsilon, \forall (i,j) \in A_{f'}$

The basic idea of the algorithm is that we will first convert the 2ϵ-optimal circulation to an ϵ-optimal *pseudoflow*, and then convert the ϵ-optimal pseudoflow to an ϵ-optimal circulation.

Definition 2 *A* pseudoflow $f : A \to \mathbb{R}$ *satisfies the following:*

- $f_{ij} = -f_{ji}$, *for all* $(i,j) \in A$
- $f_{ij} \leq u_{ij}$, *for all* $(i,j) \in A$.

Note that a pseudoflow obeys antisymmetry and capacity constraints but not flow conservation.

Definition 3 *For pseudoflow* f, *the* excess *at node* $i \in V$ *is*

$$e_i^f = \sum_{k:(k,i)\in A} f_{ki}$$

Note that this quality may be negative. If so, then negative excess is sometimes called a *deficit*.

How can we convert a 2ϵ-optimal circulation to an ϵ-optimal pseudoflow? It's easy; we just saturate every edge with negative cost. That is, for $(i,j) \in A$ such that $c_{ij}^p < 0$, set f_{ij} to u_{ij}. Then f is a 0-optimal pseudoflow.

To use a push/relabel scheme, we need to specify the conditions needed (and actions taken) for doing a push operation and a relabel operation. Obviously, in order to get from a pseudoflow to a circulation, we'd like to get rid of all excesses; following the idea of the push/relabel algorithm for maximum flow, we'll do a push on nodes with positive excess. Recall that in the maximum flow case, we only pushed along *admissible* arcs that met some criterion with their distance label. What should be the concept of an admissible arc in this case? Here we say an arc $(i,j) \in A_f$ is admissible if $c_{ij}^p < 0$. Thus we push from node i with $e_i^f > 0$ if there exists j such that $u_{ij}^f > 0$ and $c_{ij}^p < 0$. As in the maximum flow case, we will push $\delta = \min(e_i^f, u_{ij}^f)$ units of flow along (i,j).

Observe that ϵ-optimality is maintained during a push operation on (i,j) since if (j,i) is created in the residual graph, it will have reduced cost $c_{ji}^p = -c_{ij}^p > 0$.

What happens during a relabel operation? We need to relabel if there is excess at a node i, but there are no admissible arcs leaving i. In this case, all arcs with residual capacity must have non-negative reduced cost. When changing the potential of an arc, we keep two things in mind: (1) We want to maintain ϵ-optimality; (2) We want there to exist an (i,j) such that $u_{ij}^f > 0$, $c_{ij}^p < 0$. To create some admissible arc, we will simply alter the potential p_i at node i. In particular, we set

$$p_i \leftarrow \max_{(i,j) \in A_f} (p_j - c_{ij} - \epsilon).$$

Note that after a relabel operation at node i, we have

- $c_{ij} + p_i - p_j \geq -\epsilon, \forall (i,j) \in A_f$
- $c_{ij} + p_i - p_j = -\epsilon$ for some $(i,j) \in A_f$

Since previously $c_{ij}^p \geq 0$ for all $(i,j) \in A_f$, it must be the case that p_i is decreased by at least ϵ. By the above, f maintains ϵ-optimality.

Putting these together, we obtain the following algorithm.

Push/relabel find-ϵ-opt-circ(f, ϵ, p)

$\forall (i,j) \in A_f$ if $c_{ij}^p < 0, f_{ij} \leftarrow u_{ij}$
While $\exists$ active $i \in V$ $(e_i^f > 0)$
    If $\exists j$ s.t. $u_{ij}^f > 0$ and $c_{ij}^p < 0$
        Push $\delta = \min(e_i^f, u_{ij}^f)$ flow on (i,j)
    Else
        Relabel $p_i \leftarrow \max_{(i,j) \in A_f} (p_j - c_{ij} - \epsilon)$
Return (f, p)

We now want to show that the algorithm is correct and bound its running time. We'll state the following lemma and return later to its proof.

Lemma 5 *For any i, p_i decreases by at most $3n\epsilon$ during the algorithm.*

For now, let us draw out the implications of the lemma for the running time of the algorithm. We can now give the following corollary.

Corollary 6 *The total number of relabels is at most $3n^2$.*

Proof: Since p_i decreases by at least ϵ in each relabel operation, there can be at most $3n$ relabels of i. This implies that there are at most $3n^2$ relabel operations in total. □

Recall that a *push* operation is said to be *saturating* if $\delta = u_{ij}^f$, or *non-saturating* otherwise (in which case $\delta = e_i^f$). As in the case of the push/relabel algorithm for the maximum flow problem, we now bound the number of push operations by considering the two types of pushes separately.

Lemma 7 *The number of saturating pushes in the above algorithm is at most $3nm$.*

Proof: Pick any arc (i, j). Initially, $c_{ij}^p \geq 0$ if $u_{ij}^f > 0$. Therefore, we have to relabel i before we can push on (i, j), since for (i, j) to be admissible, we need $c_{ij}^p < 0$. Having had a saturating push on (i, j), in order to push flow on it again, we must first push flow back on (j, i), which implies $c_{ji}^p < 0$, which in turn implies $c_{ij}^p \geq 0$. Therefore, we need to relabel i once more to push flow on (i, j) again. This leads directly to a bound of at most $3n$ saturating pushes on (i, j). Thus for all m arcs in the graph, there can be at most $3nm$ saturating pushes. □

Now we wish to find an upper bound for the total number of non-saturating pushes in this algorithm. We need the following lemma to help us with this bound.

Lemma 8 *The set of admissible arcs is acyclic.*

Proof: We prove this lemma by induction on the algorithm. The base case of the algorithm is simple since initially no admissible arcs exist. Now suppose that the claim holds in the middle of the algorithm. Each time a *push* is executed, it can only remove admissible arcs from the residual graph, but cannot add them, so the claim holds. Each time a *relabel* is executed, it adds admissible outgoing arcs of vertex i, but removes all of i's admissible incoming arcs because all of the reduced costs of the arcs entering i are increased by at least ϵ. Since every arc entering i had reduced cost at least $-\epsilon$, after the relabel, all arcs entering i have nonnegative reduced cost, and so are not admissible. Thus no cycles can be created by the new admissible arcs coming out of vertex i. □

Now we can bound the number of non-saturating pushes.

Lemma 9 *The number of non-saturating pushes in the algorithm is $O(n^2m)$.*

Proof: Define Φ_i to be the number of vertices reachable from i via the admissible arcs, and let $\Phi = \sum_{i:e_i>0} \Phi_i$. Initially $\Phi \leq n$ (since every vertex can reach only itself); when the algorithm terminates, $\Phi = 0$, since there are no active vertices i.

What makes Φ increase? A *saturating push* on the arc (i, j) could result in a new active node j, and therefore Φ can increase by at most n. In addition, a *relabel* can increase Φ_i by at most n, but for a vertex j such that $j \neq i$, the relabel does not increase Φ_j, since all arcs entering i are no longer admissible. So, the amount that Φ increases is at most $n\left(3nm + 3n^2\right)$.

What then makes Φ decrease? From the algorithm above, we see that a *non-saturating push* decreases Φ by at least 1: after such a push, i has turned inactive, and even if some other vertex j became active as a result of the non-saturating push, it would still reach fewer vertices than i by the acyclicity of the admissible arcs.

So, the total number of non–saturating pushes in this algorithm is at most $n\left(3nm + 3n^2\right) + n = 3n^2m + 3n^3 + n = O\left(n^2m\right)$. □

From the above lemmas, we see that the total number of push/relabel operations of the algorithm is at most $O(n^2m)$. Given an implementation with $O(1)$ time per operation (which we will not discuss), we may obtain the overall computational time of the *Push/Relabel find-ε-opt-circ* subroutine:

Theorem 10 *The* Push/Relabel find-ε-opt-circ *subroutine takes $O(n^2m)$ time. Furthermore, with a FIFO implementation of Push/Relabel, the subroutine runs in $O(n^3)$ time.*

Combining this with the bound on the number of iterations of the cost-scaling algorithm, we obtain the following.

Theorem 11 *(Goldberg, Tarjan '90) The cost-scaling algorithm for the minimum-cost circulation problem can be implemented in $O(n^3 \min(\log(nC), m \log n))$ time.*

Note that if we replace *Push/Relabel find-ε-opt-circ* with a subroutine based on blocking flows, the cost-scaling algorithm can be shown to run in $O(mn \log n \cdot \min(m \log n, \log(nC)))$ time.