CS 270 Combinatorial Algorithms and Data Structures, Spring 2015

Lecture 28 : Maximum flow

We see how to compute approximate maximum flow in undirected graphs faster by computing electric flow and using multiplicative update.

Maximum Flow in Undirected Graphs

In this problem, we are given an undirected graph where each edge e has a capacity c_e, a source vertex s, and a sink vertex t, and the objective is to find a maximum flow subjected to the capacity constraints, where a flow has to satisfy the flow conservation constraints.

More formally, we have two variables f_{uv} and f_{vu} for each edge uv, and the problem is to:

$$\max \sum_{e \in \delta^{out}(s)} f_e \qquad (\text{where } \delta^{out}(s) \text{ is the set of directed edges going out from } s)$$

$$\sum_{e \in \delta^{out}(v)} f_e = \sum_{e \in \delta^{in}(v)} f_e \qquad \text{for all } v \in V - \{s,t\} \quad (\text{where } \delta^{in}(v) \text{ is the set of incoming edges to } v)$$

$$f_e \leq c_e \qquad \text{for all } e$$

$$f_e \geq 0 \qquad \text{for all } e$$

For simplicity, we assume $c_e = 1$ for all edge. So the optimal value of the linear program is in $[0, m]$ where m is the number of edges. As mentioned before, we can reduce the optimization problem to $O(\log m)$ decision problems, by doing binary search on the objective value and replace the objective function by the constraint $\sum_{e \in \delta^{out}(s)} f_e = k$.

Multiplicative update method

As in homework 2, we will apply the multiplicative weight update method to solve the problem.

We think of the objective constraint ($\sum_{e \in \delta^{out}(s)} f_e = k$), the non-negative constraints ($f_e \geq 0 \ \forall e \in E$), and the flow conservation constraint ($\sum_{e \in \delta^{in}(v)} f_e = \sum_{e \in \delta^{out}(v)} f_e \ \forall v \in V - \{s,t\}$) as "easy" constraints, and we will always satisfy them.

The capacity constraints ($f_e \leq 1 \ \forall e \in E$) are the "hard" constraints, and we will use the multiplicative weight update method to deal with them.

By the analysis of the multiplicative weight update method, we can find an almost feasible

solution ($f_e \le 1+\varepsilon \ \forall e \in E$) if we could solve $O(\frac{p \ln n}{\varepsilon^2})$ subproblems of the following form:

(0) f satisfies the objective constraint, the non-negative constraints, and the flow conservation constraints.

(1) $\sum_e w_e f_e \le (1+\varepsilon) \sum_e w_e$. In the homework problem, this is $\sum_e w_e f_e \le \sum_e w_e$, but it is easy to check that it is okay to respond with an approximate feasible solution, as we do approximation anyway.

(2) $f_e \le p$, where p is the width parameter.

If these subproblems can be solved (for different w_e), then the average solution would be an almost feasible solution, and by scaling it would satisfy all capacity with objective $k(1-\varepsilon)$.

Remark: In the multiplicative update method in L17, the convergence rate is $O(\frac{p^2 \ln n}{\varepsilon^2})$.
That analysis is for the general case where the outcome is in $[-1,+1]$.
It is not difficult to show that if the outcome is in $[0,1]$, then the convergence rate is $O(\frac{p \ln n}{\varepsilon^2})$.

Electrical flow

In the homework problem, we use a shortest path (on the weight w_e) to solve the subproblem with width k.
Here we show that the electric flow algorithm can be used to solve the subproblem with the following properties:

(0) It returns f that satisfies flow conservation constraints, non-negative constraints, and objective value constraint.

(1) When the maximum flow problem is feasible, the oracle can always return f satisfying $\sum_e w_e f_e \le (1+\varepsilon) \sum_e w_e$.

(2) When the maximum flow problem is feasible, the oracle can always return f with $f_e \le O(\sqrt{m})$.

(3) The oracle can be implemented in $\tilde{O}(m)$ time.

Assuming this can be done, we can find an $(1-\varepsilon)$-approximate solution in $O(\frac{\sqrt{m} \ln n}{\varepsilon^2})$ iterations, and the total running time is $\tilde{O}(m^{3/2}/\varepsilon^2)$.

It remains to construct the oracle with the above guarantees. As we said earlier we use electric flow.

Oracle The oracle can be obtained by setting $r_e = w_e + \frac{\varepsilon W}{m}$ where $W = \sum_e w_e$, and compute the electric flow that sends k units of electric flow from s to t with r_e as the resistance of edge e.
This is the oracle. So the whole algorithm is very simple. In each iteration, w_e is updated by $w_e^{t+1} = w_e^t (1 + \frac{\varepsilon}{\rho} f_e^t)$ and update r_e accordingly, then compute the electric flow. So, if the flow on an edge is over its capacity, we will increase its resistance based on its violation to decrease the future flow on this edge. Taking the average over all the electric flow is a good approximation.

Analysis

We check the four properties one by one.

⓪ It is clear by construction that the flow conservation constraints, the non-negativity constraints, and the objective value constraint are all satisfied.

① If the maximum flow problem is feasible, then there is a flow of k units from s to t, without violating the capacity constraints (so $f_e \le 1 \ \forall e \in E$).

The total energy of this flow is at most $\sum_e \tilde{f}_e^2 r_e \le \sum_e r_e = \sum_e \left(w_e + \frac{\varepsilon W}{m}\right) = (1+\varepsilon)W$.

Since electric flow f minimizes the energy, we must have $\sum_e f_e^2 r_e \le (1+\varepsilon)W$, or otherwise we can conclude that the maximum flow problem is infeasible

We would like to bound $\sum_e w_e f_e$, and we would use Cauchy-Schwarz.

$\left(\sum_e w_e f_e\right)^2 \le \left(\sum_e w_e f_e^2\right)\left(\sum_e w_e\right) \le \left(\sum_e r_e f_e^2\right)\left(\sum_e w_e\right) \le (1+\varepsilon)W^2$, and so we have $\sum_e w_e f_e \le (1+\varepsilon)\sum_e w_e$, as required.

② Clearly the energy on one edge cannot be more than the total energy,

so $\frac{f_e^2 \varepsilon W}{m} \le f_e^2 r_e \le (1+\varepsilon)W \Rightarrow f_e \le \sqrt{\frac{(1+\varepsilon)}{\varepsilon} m} = O(\sqrt{m})$, as required.

③ As we saw in L25, computing electric flow is the same as solving Laplacian systems, which we saw in last time that can be done in $\tilde{O}(m)$ time.

This completes the $\tilde{O}(m^{1.5})$ time algorithm to give an $(1-\varepsilon)$-approximation of the maximum flow problem in undirected graphs, matching the best known combinatorial method.

Next we present an improvement to $\tilde{O}(m^{4/3})$ time.

Faster Algorithm

One bottleneck of the algorithm is that it takes $O(\sqrt{m})$ iterations, because the flow of an edge may be as high as $\Omega(\sqrt{m})$. Consider the example where there are k disjoint paths of length k between s and t, and there is one edge between s and t. Then there will be $(k+1)/2$ units of flow on the edge between s and t. As there are k^2+1 edges in the graph, the flow on that edge is $\Theta(\sqrt{m})$.

The idea of the improved algorithm is to delete the edges with too much flow going across them.

This actually sounds counterintuitive, as those edges seem to be in optimal solutions, but it turns out

that there will not be many of them, and so deleting them will not decrease the optimal value by much, while improving the convergence rate of the multiplicative update algorithm.

Modified Algorithm

The algorithm is the same except that whenever there is an edge with electric flow value greater than p, then we delete this edge from the graph.

Effective Conductance

To analyze the effect of deleting an edge, we need the concept of effective conductance.

We know by the Thompon's principle that the electric flow is the flow that minimizes energy, among all the flow that satisfies flow conservation constraints.

$\mathcal{E}_r(f) = \sum_e f_e^2 r_e = \sum_{uv \in E} \frac{(\phi_u - \phi_v)^2}{r_{uv}}$, where ϕ_v denotes the voltage on v.

If the flow f is an electric flow of 1 unit from s to t, then $R_{eff}(s,t) = \mathcal{E}_r(f)$.

Recall that $R_{eff}(s,t) = \phi_s - \phi_t$

By setting $\phi_t = 0$ and scaling $\phi_s = 1$, we have a flow of $1/R_{eff}(s,t)$ unit from s to t, and the energy of this flow is equal to $\mathcal{E}_r(f) / R_{eff}^2(s,t) = 1/R_{eff}(s,t)$, as the electric flow of c units from s to t has energy $\mathcal{E}_r(f)/c^2$ by scaling the flow on each edge by a factor of c.

Among all vectors on vertices, the voltage vector is the one that minimizes energy.

This is a "dual" version of Thompson's principle.

Theorem Let r be the vector of resistances. Let $C_{eff}(r) = \min_{\phi : \phi_s = 1, \phi_t = 0} \sum_{uv \in E} \frac{(\phi_u - \phi_v)^2}{r_{uv}}$.

Then $C_{eff}(r)$ is minimized by the voltage vector of the electric flow of $1/R_{eff}(s,t)$ units from s to t

proof We have seen that the value $1/R_{eff}(s,t)$ is attainable by that voltage vector.

So we just need to prove that any minimizer must be the voltage vector of an electric flow.

Consider the partial derivative on a variable ϕ_v, $\frac{\partial E(\phi)}{\partial \phi_v} = \sum_{u : uv \in E} \frac{2(\phi_u - \phi_v)}{R_{uv}} \quad \forall\, v \in V - \{s,t\}$.

A minimizer must have $\frac{\partial E(\phi)}{\partial \phi_v} = 0$, and this defines a potential flow that satisfies conservation constraints. □

We use this theorem to calculate the change of effective resistance after deleting an edge.

Lemma Suppose f is an electric flow and e is an edge such that $f_e^2 r_e \geq \beta \mathcal{E}_r(f)$.

The effective resistance R' after removing an edge is at least $\frac{R}{1-\beta}$ where R is the original eff. res.

proof Without loss of generality we assume that f is a flow from s to t of $1/R$ unit.

Using the previous theorem, $\frac{1}{R} = \min_{\phi:\, \phi_s=1,\, \phi_t=0} \frac{(\phi_u-\phi_v)^2}{r_{uv}} = \mathcal{E}_r(f)$

Let $e=xy$. Then $\frac{(\phi_x-\phi_y)^2}{r_{xy}} = f_{xy}^2 r_{xy} \geq \beta \mathcal{E}_r(f)$ by assumption.

By deleting the edge e, it is the same as increasing r_e to infinity.

So $\frac{1}{R'} = C_{eff}(r') \leq \sum_{\substack{uv \in E \\ uv \neq xy}} \frac{(\phi_u-\phi_v)^2}{r_{uv}} = \mathcal{E}_r(f) - \frac{(\phi_x-\phi_y)^2}{r_{xy}} \leq (1-\beta)\mathcal{E}_r(f) = \frac{1-\beta}{R}$.

Therefore, $R' \geq \frac{R}{1-\beta}$. □

Theorem For $p = \tilde{O}(m^{1/3})$, the number of edges deleted is $\tilde{O}(m^{1/3})$.

(optional) This implies that the total complexity is $\tilde{O}(m^{4/3})$.

Let $R(i)$ be the effective resistance at step i.

Initial effective resistance

Let k^* be an optimal maximum flow between s and t.

This implies that there is an s-t cut with k^* edges.

So one of the edges in the cut would have flow value at least $1/k^*$, when we send one unit of flow from s and t.

Thus, $R(0) \geq \frac{1}{(k^*)^2}$.

Lower Bound on R(T)

When we delete an edge, it has a flow value at least p, and thus the energy on that edge is at least $p^2 r_e = p^2\left(w_e + \frac{\varepsilon W}{m}\right) \geq \frac{p^2 \varepsilon W}{m} \geq \frac{p^2\varepsilon}{(1+\varepsilon)\cdot m}$ (total energy), as total energy $\leq (1+\varepsilon)W$.

By setting $\beta = \frac{p^2\varepsilon}{(1+\varepsilon)m}$ and apply the lemma, the effective resistance increases by a factor of $\left(1-\frac{p^2\varepsilon}{(1+\varepsilon)m}\right)^{-1}$ after we delete an edge.

So, if we have deleted h edges throughout the algorithm, then $R(T) \geq R(0)\cdot\left(1-\frac{p^2\varepsilon}{(1+\varepsilon)m}\right)^{-h} \geq \frac{1}{(k^*)^2}\left(1-\frac{p^2\varepsilon}{(1+\varepsilon)m}\right)^{-h}$

Upper Bound on R(T)

The energy of the flow of k units is at most $(1+\varepsilon)\cdot W(T)$.

(This is not precise because some edges have been removed. But we can ensure that the number of edges removed is at most a small fraction of k^*. Thus the energy of the flow is only slightly increased, by scaling up the flow on the remaining edges slightly. So the total energy would not be much higher than that.)

Thus, $R(T) \le \frac{(1+\varepsilon)\cdot W(T)}{k^2}$, by scaling down the flow value by a factor of k.

Note that $W(t+1) = \sum_e w_e(t)\left(1+\frac{\varepsilon}{\rho} f_e^t\right)$ by the multiplicative update rule

$$= \sum_e w_e(t) + \frac{\varepsilon}{\rho}\sum_e w_e(t) f_e^t \le W(t)\left(1+\frac{\varepsilon(1+\varepsilon)}{\rho}\right).$$

So, $W(T) \le m\cdot\left(1+\frac{\varepsilon(1+\varepsilon)}{\rho}\right)^T \le m\cdot e^{\frac{\varepsilon(1+\varepsilon)T}{\rho}} \le m\cdot e^{\frac{2\ln n}{\varepsilon}}$ as $T=\frac{\rho\ln n}{\varepsilon}$.

Therefore, $R(T) \le \frac{(1+\varepsilon)W(T)}{k^2} \le \frac{(1+\varepsilon)\,m\cdot e^{\frac{2\ln n}{\varepsilon}}}{k^2}$.

Putting together

Combining the upper and lower bound on $R(T)$, we have $\frac{1}{(k^*)^2}\left(1-\frac{\varepsilon p^2}{m(1+\varepsilon)}\right)^{-h} \le \frac{(1+\varepsilon)m\cdot e^{\frac{2\ln n}{\varepsilon}}}{k^2}$.

Using $k^*/k \le m$, this implies $\ln\left(\frac{1}{m^3}\right) \le \ln(1+\varepsilon) + \frac{2\ln n}{\varepsilon} + h\ln\left(1-\frac{\varepsilon p^2}{m(1+\varepsilon)}\right)$

$$\Leftrightarrow \quad h \le -\frac{\ln(1+\varepsilon) + \frac{2\ln n}{\varepsilon} + 3\ln m}{\ln\left(1-\frac{\varepsilon p^2}{m(1+\varepsilon)}\right)}$$

$$\Rightarrow \quad h \le \frac{\frac{6\ln n}{\varepsilon}}{\frac{\varepsilon p^2}{m(1+\varepsilon)}} \qquad \text{using } \ln(1-c) < -c \text{ for } c\in(0,1).$$

$$\Rightarrow \quad h \le O\left(\frac{m\ln n}{p^2\varepsilon^2}\right)$$

Setting $p = \tilde{O}(m^{1/3})$, we have $h = \tilde{O}(m^{1/3})$, as required.

Discussions

The material is from the paper "Electrical flows, Laplacian systems, and faster approximation of maximum flow in undirected graphs", by Christiano, Kelner, Madry, Spielman, Teng.

An open question is whether f_t converges to an optimal solution, instead of the average.

Since this paper, many faster algorithms for combinatorial problems are designed using Laplacian solvers.

For the maximum flow problem in undirected graphs, it is now known to get an $(1-\varepsilon)$-approximation in $\tilde{O}(m)$ time.

For the maximum flow problem in directed graphs, the Laplacian solver is combined with the interior point method to get an exact $O(m^{10/7})$ algorithm, and also the same approach gives an $\tilde{O}(m\sqrt{n})$ algorithm for minimum cost flow, both improving long-standing previous best bounds.

It is an exciting time when ideas from continuous optimization and combinatorial optimization interact

to get faster algorithms in both areas.

Concluding remarks

In a first course in algorithms, we saw that most combinatorial problems are solved by "combinatorial algorithms and data structures" (this course title).

In this course (different from the course title), we see that the ideas and techniques from probability, linear algebra and optimization are the keys to the recent developments.

I think that results may not be the most important in a theory course (although there are many great results in this course), but the ideas, the tools, the proofs, and the low level details are the most important.

I hope you find this course interesting and it opens the doors for your further study, and also you will find the ideas and techniques useful in your future work.

It is a wonderful experience to teach this course in Berkeley. Thanks for everyone showing me the energy and spirit here, especially those who actively participate in the class and bring it to life!

Thanks Tselil for being a great GSI!
