6.891 Approximation Algorithms	April 21, 2000
Lecture 11	
Lecturer: David P. Williamson	*Scribe: Ken Iwamoto*

1 Metric Methods

In today's lecture, we continue our discussion of metric methods. We will consider applications to the minimum multicut and balanced cut problems.

1.1 Minimum Multicut

Recall the minimum multicut problem:

Minimum Multicut Problem

- **Input:**
  - An undirected graph $G = (V, E)$
  - k pairs of vertices $(s_i, t_i), i = 1 \ldots k$
  - Costs $c_e \geq 0$ for each edge $e \in E$
- **Goal:** Find a minimum-cost set of edges F such that for all i, s_i and t_i are in different connected components of $G' = (V, E - F)$.

Recall that this problem is NP-hard even if G is a tree.

For a given G, let $\mathcal{P}_i$ denote the set of all paths P from s_i to t_i. As usual, we begin by modelling this problem as an integer program. One (admittedly exponentially large) integer programming formulation of the problem is:

$$
\begin{array}{lll}
\text{Min} & \displaystyle\sum_{e \in E} c_e x_e & \\
\text{subject to:} & & \\
& \displaystyle\sum_{e \in P} x_e \geq 1 & \forall P \in \mathcal{P}_i, \quad \forall i \\
& x_e \in \{0, 1\}. & \forall e \in E.
\end{array}
$$

We then relax the integer program to a linear program by replacing the constraints $x_e \in \{0, 1\}$ by $x_e \geq 0$.

This linear program can be solved in polynomial time by the *ellipsoid method*, even though it is exponential-sized, if we can provide a polynomial-time *separation oracle* for the problem. A separation oracle for a problem is a subroutine which, when given a possibly feasible vector x either certifies that x is feasible for the LP, or supplies a violated LP inequality which the x fails to satisfy.

In this case, a separation oracle is simple, since we can just use a subroutine which solves the shortest path problem. That is, let x_e be the length of the edge e. The LP constraints, $\sum_{e \in P} x_e \geq 1$ for all $s_i - t_i$ paths P, will be satisfied if the shortest $s_i - t_i$ path is at least 1 in length. If the shortest $s_i - t_i$ path for some i is less than 1 in length, then that path provides a violated primal inequality. On the other hand, if, for all i the shortest $s_i - t_i$ path is at least 1 in length, then all paths are at least 1 in length, and x is a feasible vector. So a separation oracle is trivial to construct.

For that matter, we can also construct a polynomial-sized linear program which is equivalent to this LP and use more practical linear programming algorithms than the ellipsoid method.

We can gain some insight into the linear program by considering a physical interpretation of it. We think of the LP as a pipe system, where $e = (i, j) \in E$ means there's a pipe between i and j, x_e is the length of the pipe, and c_e is the cross-sectional area of the pipe. Thus the volume of pipe e is $c_e x_e$, its cross-sectional area times its length. For an example, see Figure 1. Note that the LP computes the pipe system with minimum total volume, where all $s_i - t_i$ pairs are at least 1 unit apart.

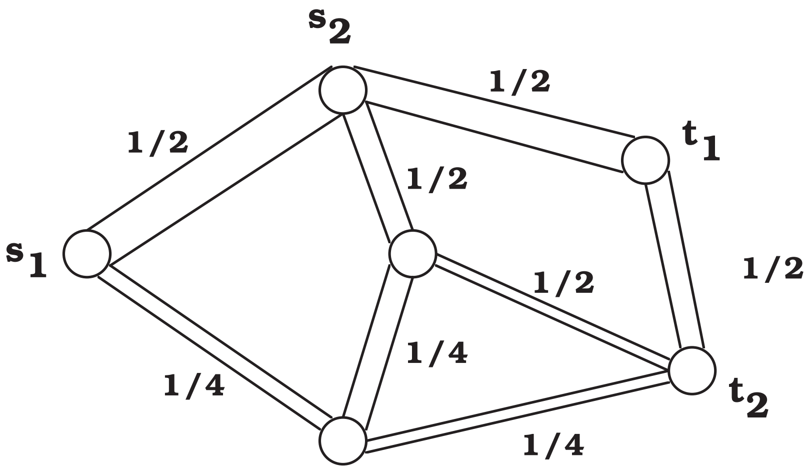

Figure 1: LP solution as a pipe system.

Given a feasible LP solution, x, let $dist_x(u, v)$ be the shortest path length between u and v with edge lengths x. Let $B_x(u, r) = \{v : dist_x(u, v) \leq r\}$ be the ball of radius r around u with the edge lengths x. See Figure 2 for an example.

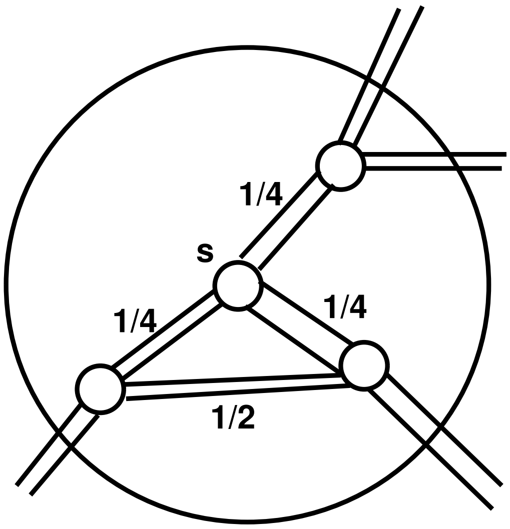

Figure 2: Example of a ball of radius 3/8.

This gives rise to a very simple approximation algorithm for the problem:

GVY

> $F \leftarrow \emptyset$.
> Solve LP and get optimal solution x.
> While there exists some connected s_i, t_i pair in current graph:
>   Let $S = B_x(s_i, r)$ for some choice of r s.t. $r < 1/2$.
>   Add $\delta(S)$ to F.
>   Remove S and $\delta(S)$ from current graph.
> Return F.

Two simple lemmas are immediate.

Lemma 1 *The algorithm terminates.*

Proof: The algorithm terminates because we disconnect at least one pair per iteration of the while loop; s_i and t_i are separated. $\square$

Lemma 2 *The algoritm returns a multicut.*

Proof: The only difficulty in proving this statement is that in some iteration in which we separated s_i and t_i, we ended up with both s_j and t_j in S for some j. Then the algorithm would remove S from the graph and s_j and t_j might not be separated in the solution returned by the algorithm.

Suppose this happens. Then by construction of the algorithm it follows that $dist_x(s_i, s_j) \leq r$ and $dist_x(s_i, t_j) \leq r$. This means that a path from s_j to s_i to t_j exists of length less than $2r$. But, since $r < 1/2$, that means there's a path between s_j and t_j of length less than 1, which violates the LP feasibility of x. Hence, this can't happen, and the algorithm returns a feasible solution to the LP. $\square$

Fix some iteration of the algorithm in which the pair (s_i, t_i) is chosen. To prove the performance guarantee of the algorithm, define the following two quantities

Definition 1

$$V_x(s_i, r) = \frac{V^*}{k} + \sum_{e=(u,v):u,v \in B_x(s_i,r)} c_e x_e + \sum_{e=(u,v)\in\delta(B_x(s_i,r))} c_e(r - dist_x(s_i, u)),$$
$$C_x(s_i, r) = \sum_{e\in\delta(B_x(s_i,r))} c_e.$$

That is, $V_x(s_i, r)$ is the total volume of pipe in the ball of radius r around s_i plus an extra term V^*/k. Also, $C_x(s_i, r)$ is the cost of the cut defined by the vertices in the ball of radius r around s_i.

Let us observe some properties of these two functions. Clearly, $V_x(s_i, r)$ is an increasing function of r (as we increase r, the ball of radius r contains more volume). In fact this is piece-wise linear function, with possible discontinuities at those values of r for which there exists a vertex v such that $dist_x(s_i, v) = r$. So, if there doesn't exist a vertex $v \in V$ such that $dist_x(s_i, v) = r$, then $V_x(s_i, r)$ is differentiable at r. Moreover, from the definition of $V_x(s_i, r)$ and $C_x(s_i, r)$, it is clear that

$$\frac{d(V_x(s_i, r))}{dr} = C_x(s_i, r)$$

We will prove the following theorem.

Theorem 3

$$\exists \quad r < 1/2 \quad s.t. \quad \frac{C_x(s_i, r)}{V_x(s_i, r)} \leq 2 \ln 2k.$$

Furthermore, we can find such an r in polynomial time.

Intuitively, this theorem says that if we choose this particular value of r, then we can charge the cost of the cut $C_x(s_i, r)$, i.e., the set of edges removed, to the volume of the pipe system we remove. Let us make this formal by showing how the theorem leads to a good approximation algorithm for the minimum multicut problem.

Theorem 4 *(Garg, Vazirani, Yannakakis '96) GVY is a $(4 \ln 2k)$-approximation algorithm for the minimum multicut problem.*

Proof: To prove the bound, we charge the cost of the edges in each cut $\delta(B_x(s_i, r))$ added to F in each iteration against the volume removed from the graph plus V^*/k; that is, against $V_x(s_i, r)$. We know that $C_x(s_i, r) \leq (2\ln 2k)V_x(s_i, r)$. Since the edges in $B_x(s_i, r)$ and $\delta(B_x(s_i, r))$ are removed from the graph, we can only charge against these edges once. Thus the total cost of edges in our solution can be no more than $2\ln 2k$ times the total volume of the graph plus $k \times V^*/k$. That is,

$$\begin{aligned}\sum_{e\in F} c_e &\leq 2\ln 2k(V^* + V^*)\\ &\leq (4\ln 2k)V^*\\ &\leq (4\ln 2k)OPT,\end{aligned}$$

since V^* is the value of a linear programming relaxation of the problem. □

Proof of Theorem 3: Relabel the vertices according to their distance from s_i, i.e., the vertices are relabeled as $s_i = v_1, v_2, \ldots, v_n$. Define $r_j = dist(s_i, v_j)$. So, $0 = r_1 \leq r_2 \leq \ldots \leq r_l = \frac{1}{2}$.

Suppose the statement of the theorem is not true. Then, for $r \in (r_j, r_{j+1})$ (recall that $V_x(s_i, r)$ is differentaible for this r), the fact that $C_x(s_i, r)$ equals the derivative of $V_x(s_i, r)$ implies that

$$\frac{1}{V_x(s_i, r)}\frac{d(V_x(s_i, r))}{dr} > 2\ln 2k$$

We now integrate both sides from r_j to r_{j+1} (actually it should be from r_j^+ to r_{j+1}^-). we get

$$\begin{aligned}\int_{r_j}^{r_{j+1}} \frac{1}{V_x(s_i, r)}\frac{d(V_x(s_i, r))}{dr}dr &> \int_{r_j}^{r_{j+1}} 2\ln 2k dr\\ \text{i.e.,}\quad \ln(V_x(s_i, r_{j+1})) - \ln(V_x(s_i, r_j)) &> 2(r_{j+1} - r_j)\ln 2k\end{aligned}$$

Now, we would like to add these terms for j varying from 1 to $l-1$ so that all but two terms on the LHS cancel out. But there is a caveat. We are ignoring the discontinuities of $V_x(s_i, r)$ at r_j for all j. But the fact that $V_x(s_i, r)$ is an increasing function implies that by doing this we will only underestimate the LHS of this telescopic sum. So, we do not get the wrong result if we sum the above equation from $j = 1$ to $l - 1$. By summing the above equation from $j = 1$ to $l - 1$ and cancelling terms, we get

$$\ln V_x(s_i, r_l) - \ln V_x(s_i, 0) > 2(r_l - 0)\ln 2k$$

We know that $r_l = \frac{1}{2}$ and $V_x(s_i, 0) = \frac{V^*}{k}$. Substituting these facts, we get

$$\ln V_x(s_i, \frac{1}{2}) > \ln\left(\frac{V^*}{k}\right) + \ln 2k$$

Removing logarithms on both sides, we get

$$V_x(s_i, \frac{1}{2}) > 2V^*$$

which is a contradiction because $V_x(s_i, \frac{1}{2})$ is only a part of the total volume. Thus, the theorem must be true.

In algorithm GVY, for our choice of an appropriate $r < 1/2$, we choose an $r < 1/2$ such that the theorem is true. How can we find such an r? First, sort vertices of G according to their distance from s_i. That is, consider vertices $v_1, v_2, \ldots, v_n$ such that $s_i = v_1$, and $r_j = dist_x(s_i, v_j)$ for $r_1 = 0 \leq r_2 \leq \ldots \leq r_l = 1/2$. Notice that in any interval (r_j, r_{j+1}) the value of $V_x(s_i, r)$ is increasing, while the value of $C_x(s_i, r)$ stays the same. Thus in any interval (r_j, r_{j+1}) the ratio is largest at the very end of the interval. Thus, given that we know that the theorem is true, we only need check the ratio at $r_{j+1} - \epsilon$ for some tiny value of $\epsilon > 0$, for each j. □

1.2 Balanced Cut

The method we used to approximate solutions to the minimum multicut problem, namely interpreting an LP solution as a "length" or metric in some clever way, extends nicely to other problems. We consider now one of these, the balanced cut problem:

Balanced Cut Problem

- **Input:**
  - $G = (V, E)$
  - Costs $c_e \geq 0$ for each edge $e \in E$
  - A number, $b \in (0, \frac{1}{2}]$
- **Goal:** Find a set $S \subseteq V$ which minimizes $\sum_{e \in \delta(S)} c_e$ and which satisfies $bn \leq |S| \leq (1-b)n$.

Note that $b = \frac{1}{2}$ gives graph bisections, in which half the nodes of the graph are on each side of the cut. Another typical value is $b = \frac{1}{3}$. But why do we care about balanced cuts? As it turns out, balanced cuts are useful as subroutines in some divide and conquer strategies. Because each side of the cut contains some constant fraction of the nodes, if we apply this recursively, we can only do this $O(\log n)$ times. Furthermore, the minimization of the edges in the cut makes the "merge" step of such strategies easier or in some way cheaper.

Definition 2 *By $OPT(b)$, we mean the optimal value of a b-balanced cut.*

Innocent as the balanced cut problem sounds and important as it is, there was very known about it until quite recently. The best result until very recently is due to Leighton and Rao (1988):

Theorem 5 *There exists a polynomial-time algorithm for finding a b-balanced cut with $b \leq \frac{1}{3}$ of value $O(\log n) OPT(b')$ for $b' > b + \epsilon$, for any fixed $\epsilon > 0$.*

There is a small "cheat" above, in the sense that we don't get an algorithm truly in the spirit of those we have considered in this course: $OPT(b')$ could be quite large compared to $OPT(b)$. We will consider a simplified version of the above result, due to Even, Naor, Rao, and Schieber (1995):

Theorem 6 *There exists a polynomial time algorithm for finding a $\frac{1}{3}$-balanced cut of value $O(\log n) OPT(\frac{1}{2})$.*

At STOC 2000, Feige, Krauthgamer, and Nissim will be presenting that comes within a factor of $O(\sqrt{n} \log n)$ of optimal, and Feige and Krauthgamer have recently announced an approximation algorithm for bisection with a performance guarantee of $O(\log^4 n)$. But consider that our current stage of knowledge does not even allow us to disprove the existence of a polynomial-time approximation scheme. So, let us now study the simplified approach, to learn what we can.

Definition 3 *By $\mathcal{P}_{uv}$ we mean the set of all paths from $u \in V$ to $v \in V$.*

Now, consider the following linear program, whose optimal value we can use as a bound:

$$Z_{LP} := \quad \text{Min} \quad \sum_{e \in E} c_e x_e$$

subject to:

$$\sum_{v \in S} \sum_{e \in P_{uv}} x_e \geq \left(\frac{2}{3} - \frac{1}{2} \right) n \quad \forall\ S\ s.t.\ |S| \geq \frac{2}{3} n, u \in S,\ P_{uv} \in \mathcal{P}_{uv}$$

$$x_e \geq 0.$$

The quantification is intended to read that for any S such that $|S| \geq \frac{2}{3} n$, we pick any $u \in S$, and for each $v \in S$, sum the x_e over some u-v path. The total sum over all $v \in S$ should be at least $(\frac{2}{3} - \frac{1}{2})n$.

We show that this LP is a relaxation of the minimum bisection problem.

Lemma 7 $Z_{LP} \leq OPT(\frac{1}{2})$.

Proof: Given an optimal bisection S, construct a solution $\bar{x}$ for the LP by setting $\bar{x}_e = 1$ if $e \in \delta(S)$, and $\bar{x}_e = 0$ otherwise. We show that this is a feasible solution, which is enough to prove the lemma.

Consider any S' such that $|S'| \geq \frac{2}{3} n$, and any $u \in S'$. Note that there must be at least $(\frac{2}{3} - \frac{1}{2})n$ verticies in $S' - S$, because in the worst case, $S \subseteq S'$ (do the math).

Suppose $u \in S$. As "proof-by-picture", consider Figure 3. By our observation about $|S' - S|$, it is easy to see that $\sum_{v \in V} \sum_{e \in P_{uv}} \bar{x}_e \geq \left(\frac{2}{3} - \frac{1}{2}\right) n$.

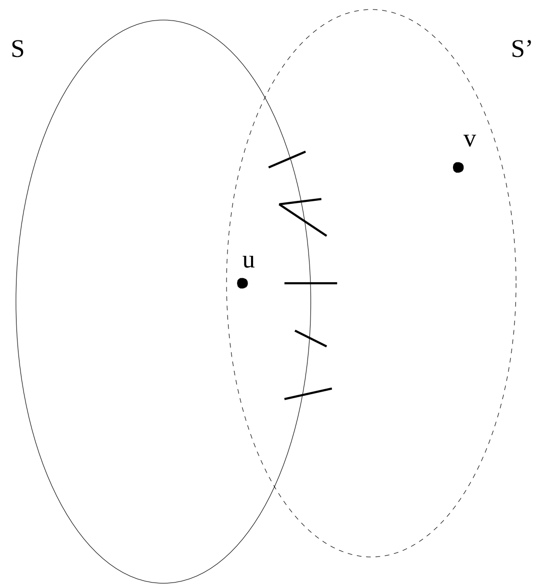

Figure 3: Dark lines represent edges whose variable is set to one.

When $u \notin S$, the argument is essentially the same (because there are at least $\left(\frac{2}{3} - \frac{1}{2}\right) n$ vertices in $S' \cap S$). So, the solution given by $\bar{x}$ is in fact feasible and hence, $Z_{LP} \leq OPT(\frac{1}{2})$. $\square$

There is a problem we have ignored in all this – the LP could be quite large. If we can find a polynomial-time separation oracle, we can apply the ellipsoid method and not worry about the LP being too large.

We can get our oracle as follows: given a solution x to check for feasibility, fix a node $u \in V$, run Dijkstra's algotithm (with x_e as edge lengths) to get an ordering of the nodes $\{u = v_1, v_2, \ldots, v_n\}$ from closest to farthest from u. Now consider the set $S = \left\{v_1, v_2, \ldots, v_{\frac{2}{3}n}\right\}$. If some constraint is violated for this choice of u, then certainly it must be violated for S since these are the sets of vertices closest to u; that is, the sum of the path lengths can be no smaller for any other S such that $|S| \geq \frac{2}{3}n$.

We are now in position to state an algorithm, and begin its analysis.

ENRS

Solve the LP for optimal x.
$S \leftarrow \emptyset$
$F \leftarrow \emptyset$
While $\lvert S\rvert < \frac{n}{3}$
    Choose some u, v pair in the current graph such that
        $dist_x(u,v) \geq \left(\frac{2}{3} - \frac{1}{2}\right) = \frac{1}{6}$
    $C \leftarrow B_x(u,r)$, for an appropriate $r < \frac{1}{12}$
    $C' \leftarrow B_x(v,r')$, for an appropriate $r' < \frac{1}{12}$
    Add C or C' to S, whichever gives $\min\{\lvert C\rvert, \lvert C'\rvert\}$
        and $\delta(C)$ (or $\delta(C')$) to F
    Remove C (or C') from the graph.

Lemma 8 *If $|S| < \frac{n}{3}$, there exists a u, v pair in the current graph such that $dist_x(u,v) \geq \frac{1}{6}$.*

Proof: Consider $\bar{S} = V - S$. Then $|\bar{S}| > \frac{2n}{3}$. This implies that $\sum_{v \in \bar{S}} \sum_{e \in P_{uv}} x_e \geq (\frac{2}{3} - \frac{1}{2})n = \frac{n}{6}, \forall\ u \in S,\ P_{uv} \in \mathcal{P}_{uv}$. But then the average path length from u to a $v \in \bar{S}$ is at least $\frac{1}{6}\frac{n}{|\bar{S}|} \geq \frac{1}{6}$. Thus by the pigeon-hole principle, there exists some $v \in \bar{S}$ such that $dist_x(u,v) \geq \frac{1}{6}$. □

Lemma 9 *ENRS outputs a $\frac{1}{3}$-balanced cut.*

Proof: $|S| \geq \frac{n}{3}$ at termination, by design. So, we only need to show that $|S| \leq \frac{2n}{3}$. Choose *any* iteration of the while loop. At the begining of the iteration, we certainly have $|S| < \frac{n}{3}$, by design, and at the end of the iteration, $|S| \leftarrow |S| + \min\{|C|, |C'|\}$. Note that $dist_x(u,v) \geq \frac{1}{6}$. Since $C = B_x(u,r)$ and $C' = B_x(v,r')$ for $r, r' < \frac{1}{12}$, it must be the case that C and C' are disjoint. This implies that $\min\{|C|, |C'|\} \leq \frac{1}{2}(n - |S|)$, i.e. the smaller of C and C' can be no more than half the remaining vertices. So,

$$\begin{aligned} |S| + \min\{|C|, |C'|\} &\leq |S| + \frac{1}{2}(n - |S|) \\ &= \frac{1}{2}n + \frac{1}{2}|S| \\ &\leq \frac{2n}{3} \end{aligned}$$

□

We have reached the point where the analysis will begin to look very familiar, i.e. we follow closely the model of the minimum multicut analysis. So, we present the following definitions, analagous to those we have seen before:

Definition 4

$$V_x(u,r) := \frac{Z_{LP}}{n} + \sum_{e=(u,v)\ :\ v,w\ \in\ B_x(u,r)} c_e + \sum_{e=(v,w)\ \in\ \delta(B_x(u,r))} c_e(r - dist_x(u,w))$$

Definition 5

$$C_x(u,r) = \sum_{e=(v,w)\ \in\ \delta(B_x(u,r))} c_e$$

And now we prove a familiar looking theorem, whose bound is somewhat different, but whose proof is practically identical (and so we omit it):

Theorem 10 *There exists* $r < \frac{1}{12}$ *such that* $\frac{C_x(u,r)}{V_x(u,r)} \leq\ 12 \ln 2n$

We find the "appropriate choice" of r as before (i.e., one that achieves the bound in the theorem). Now we may state the final theorem, due to Even, Naor, Rao, and Schieber (1995):

Theorem 11 *There exists a polynomial time algorithm for finding a* $\frac{1}{3}$*-balanced cut* S *such that*

$$\sum_{e\ \in\ \delta(S)} c_e \leq (24 \ln 2n) Z_{LP} = (24 \ln 2n) OPT\left(\frac{1}{2}\right).$$

We omit the proof due to its similarity to the analagous proof in the analysis of the minimum multicut problem.