

# Metric Methods

---

A metric space is given by a set  $V$  and a distance function  $d : V \times V \rightarrow \mathcal{R}^+$

The distance function  $d$  satisfies

triangle inequality:  $d(u,v) + d(v,w) \geq d(u,w)$  for all  $u,v,w \in V$

symmetry:  $d(u,v) = d(v,u)$

reflexivity:  $d(u,v) = 0 \Rightarrow u = v$

If we drop the reflexivity then we get a semi-metric

If we drop symmetry we simply get a distance function

# Metrics/Distances in Algorithms

---

Metrics/distances arise for three main reasons

- As input to a problem
- As desired output for a problem
- As part of solving a problem

# Metrics/distances as relaxations

---

An important application of metrics is in solving cut problems. In a cut problem we are given some graph and the goal is to remove some edges of minimum cost to satisfy some constraints. Typically the constraints ask for separation of some sets of vertices. Metrics arise naturally in trying to solve these problems as follows.

Given a subset  $S$  of edges in  $G$ , we can define a distance function  $d_S$  on the vertices of  $V$  as

$d_S(u, v) = 1$  if  $u$  and  $v$  are separated by removing  $S$

$d_S(u, v) = 0$  otherwise

# Metrics/distances as relaxations

---

Given a subset  $S$  of edges in  $G$ , we can define a distance function  $d_S$  on the vertices of  $V$  as

$d_S(u, v) = 1$  if  $u$  and  $v$  are separated by removing  $S$

$d_S(u, v) = 0$  otherwise

Note that if  $G$  is undirected then  $d_S$  is symmetric and hence is a semi-metric while  $d_S$  need not be symmetric in directed graphs

A cut problem consists of choosing a min-cost set of edges to remove so that some separation constraints on vertices are satisfied. We can express that as an integer program as follows

# Metrics and cuts

---

we have a binary variable  $d_e$  for each edge  $e$  and it indicates whether  $e$  is chosen in the cut or not

$c_e$  is cost of  $e$

$$\min \sum_e c_e d_e$$

s.t

$$d(s, t) = 1 \text{ for all pairs } (s, t) \text{ that need to be separated}$$
$$d_e \in \{0, 1\}$$

In the above  $d(s,t)$  is the shortest path distance between  $s$  and  $t$  with edge lengths given by  $d_e$

We can express shortest path distances using linear constraints as shown below

# Metrics and cuts

---

Given  $d_e$  we wish to obtain  $d(u,v)$  for each pair of vertices  $u,v$  where  $d(u,v)$  is the distances between  $u$  and  $v$  in the graph with edge lengths given by  $d_e$

We simply write the simple triangle inequalities

$$d(u,v) + d_e \leq d(u, w) \text{ if } e = (v,w)$$

A less cumbersome way to do this is to assume that the given graph is a complete graph. This is wlog since any edge which is not in the original graph can be zero cost

# Metrics and cuts

---

If we assume above then the edge and distance variables are essentially same so we simply have variables  $d(u,v)$  for each pair of vertices  $(u,v)$  then we can write cut problems as

$$\min \sum_{u,v} c(u,v) d(u,v)$$

s.t

$d(s,t) = 1$  for each pair  $s,t$  that needs to be separated

$d(u,v) + d(v,w) \geq d(u,w)$  for all  $u,v,w$

$d(u,v) \in \{0,1\}$

# Multiway-cut and LP Rounding

---

Recall the multiway-cut problem

Given undirected graph  $G=(V,E)$  with edge costs  $c: E \rightarrow \mathcal{R}^+$  and terminals  $T = \{t_1, t_2, \dots, t_k\} \subset V$

Find a min-cost set of edges  $E' \subseteq E$  whose removal results in separating the terminals

Previously we saw a greedy  $2(1-1/k)$  approximation using two slightly different greedy algorithms

Here we use an LP relaxation to obtain the same result

# LP Relaxation

---

One formulation is the following:  
variable  $l_e$  for  $e \in E$  to indicate if  $e$  is cut or not

Let  $P_{ij} = \{ p \mid p \text{ is a path from } t_i \text{ to } t_j \text{ in } G \}$

$$\min \sum_e c_e l_e$$

s.t

$$\sum_{e \in p} l_e \geq 1 \quad p \in P_{ij}, \quad 1 \leq i < j \leq k$$

$$l_e \geq 0$$

# LP Relaxation

---

$$\min \sum_e c_e l_e$$

s.t

$$\sum_{e \in p} l_e \geq 1 \quad p \in P_{ij}, \quad 1 \leq i < j \leq k$$

$$l_e \geq 0$$

Note: formulation has exponential # of constraints but LP has a polynomial time separation oracle and hence can be solved in polynomial time.

# A polynomial sized formulation

---

One could also write a different formulation which is essentially equivalent but has polynomial size extra variable  $d(uv)$  for each unordered pair of vertices  $uv$ .  $d(uv)$  is to model the shortest path distance between  $u$  and  $v$  in the metric induced by  $l$

$$\min \sum_e c_e l_e$$

s.t

$$\begin{aligned} d(uv) + d(vw) &\geq d(uw) && u, v, w \in V \\ d(uv) &\leq l_e && \text{for each edge } e=uv \\ d(t_i t_j) &\geq 1 && 1 \leq i < j \leq k \\ l, d &\geq 0 \end{aligned}$$

# A polynomial sized formulation

---

In fact we do not need to maintain the variables  $l$  anymore since in any optimum solution  $d(uv) = l_{uv}$

Thus the formulation can be written as

$$\min \sum_{uv \in E} c(uv) d(uv)$$

s.t

$$d(uv) + d(vw) \geq d(uw) \quad u, v, w \in V$$

$$d(t_i t_j) \geq 1 \quad 1 \leq i < j \leq k$$

$$d \geq 0$$

# of variables is  $n(n-1)/2$  and # of constraints is  $\Theta(n^3)$

# Rounding the LP

---

Note that both LP's essentially assign lengths/distances to each  $e$

Let us work with the first LP

we let  $l(uv)$  denote the shortest path distance between  $u$  and  $v$  with edge lengths given by  $l$

Note that the LP ensures that  $l(t_i t_j) \geq 1$  for each  $i, j$

For a vertex  $v$  and a real value  $r$  let  $B(v, r) = \{u \mid l(vu) < r\}$  denote the *(open) ball of radius  $r$  around  $v$*

# Rounding the LP

---

Pick  $\theta$  uniformly at random from  $[0, 1/2)$

For each  $t_i$ , remove all edges  $\delta(B(t_i, \theta))$   
that is remove  $E' = \bigcup_{i=1}^k \delta(B(t_i, \theta))$

**Claim:**  $E'$  is a feasible solution

We observe that the only vertices reachable from  $t_i$  after removing  $E'$  are in  $B(t_i, \theta)$

And  $B(t_i, \theta) \cap B(t_j, \theta) = \emptyset$  for otherwise we would have

$$I(t_i, t_j) < 1$$

Note that  $B(t_i, 1/2) \cap B(t_j, 1/2) = \emptyset$  (Why?)

# Expected cost of $E'$

---

We now estimate  $\text{Expect}[c(E')]$  the expected cost of  $E'$   
It is enough to estimate the probability that  $e \in E'$

We prove the following lemma

Lemma:  $\Pr[e \in E'] \leq 2 l_e$

Assuming the lemma,

$$\text{Expect}[c(E')] \leq \sum_e 2c_e l_e \leq 2 \text{OPT}_{LP} \leq 2\text{OPT}$$

# Proof of Lemma

---

Focus on  $e = uv$

Four cases:

Case 1:  $u, v \in B(t_i, 1/2)$  for some  $i$

Wlog,  $l(t_i u) \leq l(t_i v)$ , that is  $u$  is closer to  $t_i$  than  $v$

Then

$$\begin{aligned} \Pr[e \text{ is cut}] &= \Pr[l(t_i u) \leq \theta \text{ and } l(t_i v) > \theta] \\ &= 2(l(t_i v) - l(t_i u)) \leq 2 l_e \end{aligned}$$

# Proof of Lemma

---

Case 2:  $u \in B(t_i, 1/2), v \in B(t_j, 1/2)$  for some  $i, j$

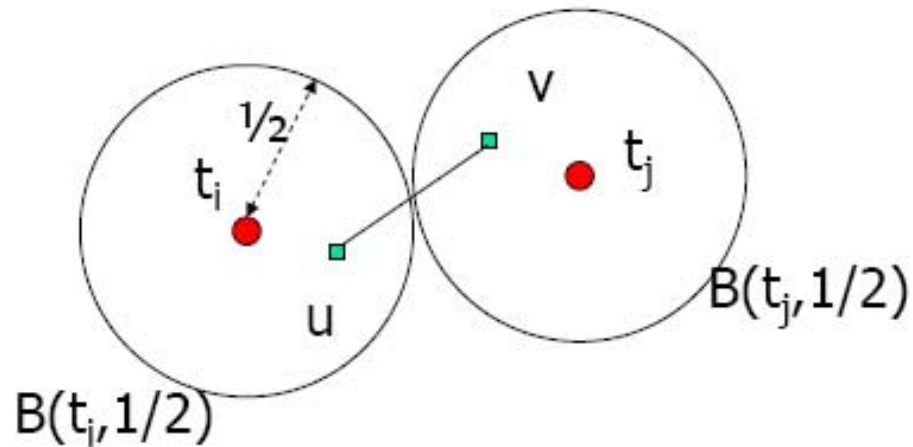
Note that in this case  $e$  can be cut either in  $B(t_i, 1/2)$  or  $B(t_j, 1/2)$  (see pic next slide)

Then

$$\begin{aligned} & \Pr[e \text{ is cut}] \\ & \leq \Pr[e \text{ is cut in } B(t_i, 1/2)] + \Pr[e \text{ is cut in } B(t_j, 1/2)] \\ & \leq \Pr[l(t_i, u) < \theta \leq 1/2] + \Pr[l(t_j, v) \leq \theta \leq 1/2] \\ & \leq 2(1/2 - l(t_i, u)) + 2(1/2 - l(t_j, v)) \\ & \leq 2(1 - l(t_i, u) - l(t_j, v)) \leq 2l_e \text{ (Why?)} \end{aligned}$$

# $u, v$ in different balls

---



# Other cases

---

Case 3:  $u \in B(t_i, 1/2)$  and  $v$  is not in any ball

In this case  $\Pr[e \text{ is cut}] = 2(1/2 - l(t_i u)) \leq 2l_e$  (Why)

Case 4:  $u, v$  are both not in any ball

In this case  $uv$  will not be cut

# Derandomization and improvement

---

We can derandomize the algorithm. Although  $\theta$  is picked from  $[0, 1/2)$  there are only a polynomial number of values at which  $B(t_i, \theta)$  changes when increasing from 0 to  $1/2$  corresponding to a BFS search from  $t_i$ . Hence we can try “all possible values” of  $\theta$  and pick the value at which we obtain the cheapest cut.

Note that we don't have to cut in all balls. See how you can improve analysis to obtain a  $2(1-1/k)$  bound

# Tight example

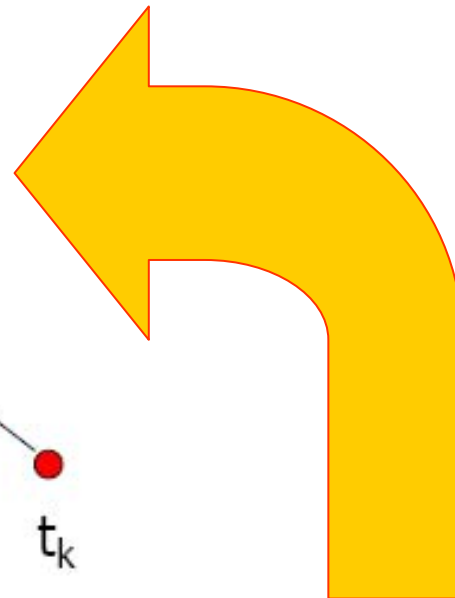
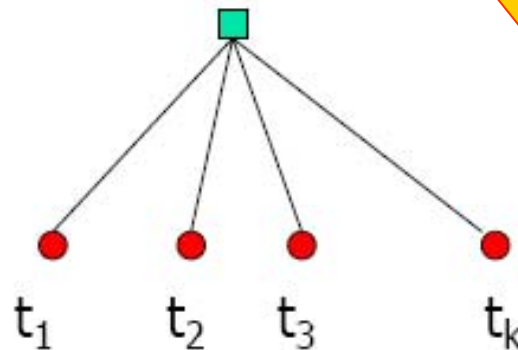
---

Star with  $k$  leaves each of which is a terminal

$$OPT = k-1$$

$$OPT_{LP} = k/2 \text{ (why?)}$$

$$\text{So integrality gap} = 2(1-1/k)$$



So to get a better performance guarantee,  
we must look at a different LP-relaxation !