Embedding Finite Metrics into Tree Metrics

In this class we will obtain the following embedding result.

Let $G=(V,E)$ be an undirected edge-weighted graph which defines a metric space on V

$d_G(uv)$: shortest path distance from u to v in G

There is a randomized polynomial time algorithm that produces an edge-weighted tree $T=(V_T, E_T)$ s.t

1. $V \subseteq V_T$
2. $d_G(uv) \leq d_T(uv)$ for all $u,v \in V$
3. $E[d_T(uv)] = O(\log^2 n)\, d_G(uv)$ for all $u,v \in V$

$E[d_T(uv)]$ is the *expected distance* in T

Embedding Finite Metrics into Tree Metrics

In other words the algorithm is generating a probability distribution on *dominating* tree metrics s.t for any pair uv, the expected distance in the tree is $O(\log^2 n)$ times the distance of the pair in the graph.

The trees are dominating in the sense that for all pairs uv, $d_T(uv) \geq d_G(uv)$ (the distances only increase)

One can show a better bound of $O(\log n)$ which is the best possible. That is, there are graphs for which no probability distribution over dominating tree metrics can guarantee better than $\Omega(\log n)$ expected distance

Random low-diameter graph partitions

Let $\Delta(G)$ denote diameter of G

The key to the tree embedding is a way to partition the graph G into low-diameter pieces such that no pair uv is separated with too high a probability

More precisely we want to randomly partition G into $G_1, G_2, \ldots, G_k$ s.t

1. $\Delta(G_i) \leq \Delta(G)/2$ for $1 \leq i \leq k$
2. $\Pr[uv \text{ not in same } G_i] \leq \alpha \, d_G(uv)/\Delta(G)$ for small α

Random low-diameter graph partitions

We will prove a stronger statement: given a parameter $\delta \leq \Delta(G)$, there is an algorithm that produces a random partition of G into sub-graphs $G_1, G_2, \ldots, G_k$ such that

1. $\Delta(G_i) \leq \delta$ for $1 \leq i \leq k$
2. $\Pr[uv \text{ not in same } G_i] \leq O(\log n)\, d_G(uv)/\delta$

We can strengthen the second property as follows:

$\Pr[\text{Ball}(u, \rho) \text{ is cut}] \leq O(\log n)\, \rho/\delta$

Random low-diameter graph partitions

The algorithm is similar to the one for Multicut in last class

Let $V = \{v_1, v_2, \ldots, v_n\}$

Pick a random permutation π of nodes of G

Pick θ uniformly at random from $[\delta/4, \delta/2)$

$color(v_i) = 0$ for all $v_i \in V$

for $i = 1$ to n do
  for all $u \in V$ s.t $u \in Ball(v_{\pi(i)}, \theta)$
    if ($color(u) = 0$)
      $color(u) = i$
  end for
end for

Random low-diameter graph partitions

As before a vertex u gets assigned to the "first" ball in the permutation

Let $V_i = \{ v_j \mid \text{color}(v_j) = i \}$, all vertices assigned color i

We can also see V_i as
$V_i = \text{Ball}(v_i,\theta) \setminus \cup_{j \leq \pi(i)} \text{Ball}(v_{\pi(j)}, \theta)$

The random partition is simply $G_1, G_2, \ldots, G_n$ where $G_i = G[V_i]$. We ignore the empty graphs

By construction each G_i has $\Delta(G_i) \leq 2\theta \leq \Delta(G)/2$

Random low-diameter graph partitions

We will now show that
Pr[ab separated (not in same G_i)] $\leq$ O(log n) $d_G(ab)/\delta$

The analysis is similar to the earlier analysis for Multicut

Fix the pair ab: let L_i = min $\{d(v_i, a), d(v_i, b)\}$,

U_i = max $\{d(v_i, a), d(v_i, b)\}$
Assume wlog that $L_1 \leq L_2 \leq \ldots \leq L_n$

We now evaluate Pr[ab *first* cut by v_i]
Note that Pr[ab separated] $\leq \sum_i$ Pr[ab first cut by v_i]

Random low-diameter graph partitions

We now evaluate Pr[ab *first* cut by v_i]

ab is first cut by v_i implies that $\min\{color(a),color(b)\} = i$ and $\max\{color(a), color(b)\} > i$

The above happens only if v_i comes before $v_1, \ldots, v_{i-1}$ in the random permutation π *and* $\theta \in [L_i, U_i)$]

This happens with probability at most
$(1/i)\ \ 4(U_i - L_i)/\delta \leq 4\ d(ab)/(i\delta)$

Therefore Pr[ab separated] $\leq 4\ H_n\ d(ab)/\ \delta$

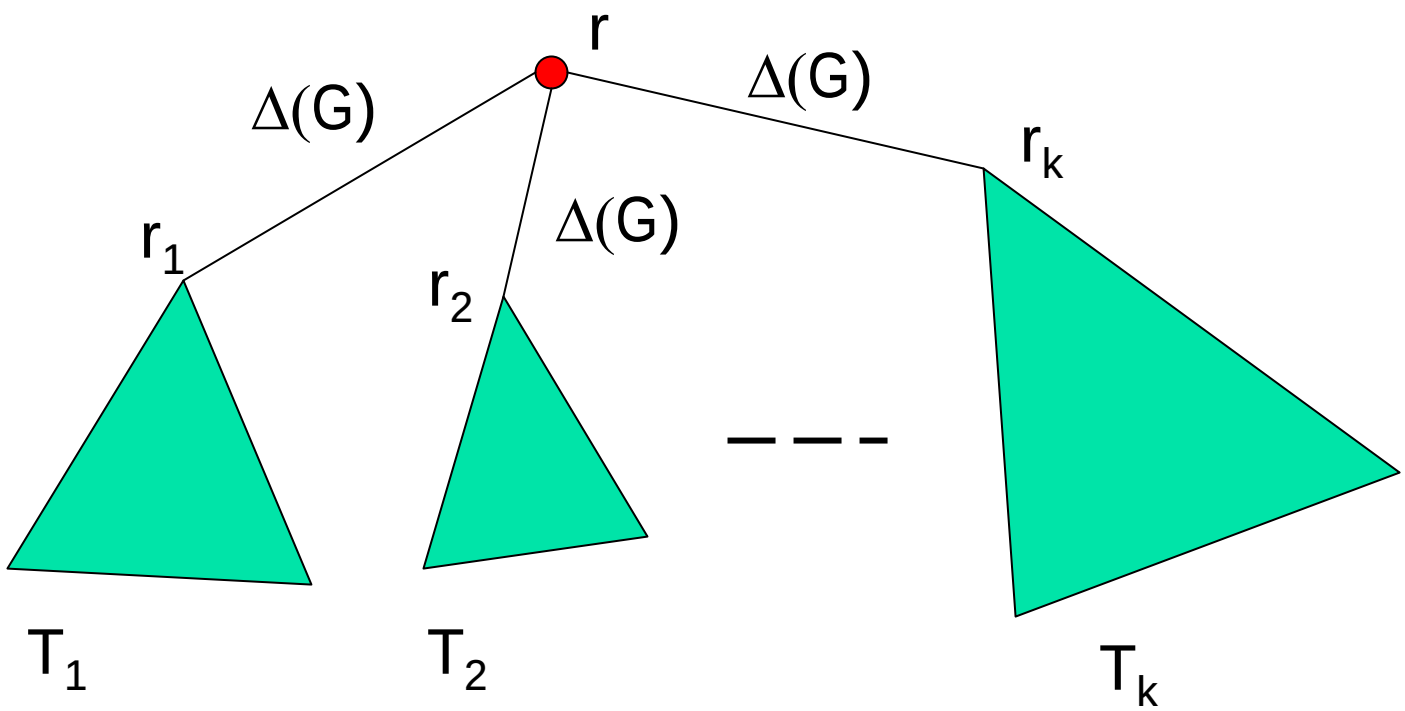
One can easily modify above analysis to prove that
Pr[Ball(a,ρ) cut] $\leq O(\log n)\ \rho/\delta$

Tree embedding

Now we use the random partitioning scheme to produce a tree embedding of G as follows:

- Base case: if G consists of a single node v, output $T=\{v\}$
- Use random partitioning with parameter $\delta = \Delta(G)/2$ to obtain $G_1, \ldots, G_k$
- Recursively create trees $T_1, \ldots, T_k$ for $G_1,\ldots,G_k$ with roots $r_1, \ldots, r_k$
- Create T by adding a root r and connect it to each r_i with an edge of length $\Delta(G$
- Output T

Construction of T



Analysis

$d_T(uv) \geq d_G(uv)$ for all uv

This is true for all uv that are separated at top level since they are in different G_i and in T the path from a to b has to use two edges of length $\Delta(G)$ each, hence $d_T(uv) \geq 2\Delta(G) \geq d_G(uv)$

If uv are not separated at top level then the property holds inductively: note that if $u,v \in G_i$ then $d_{G_i}(uv) = d_G(uv)$

Analysis

First we observe the following: $\Delta(T) \leq 4\Delta(G)$

We claim that length of any root to leaf path $\leq 2\Delta(G)$

The first edge is of length $\Delta(G)$, the next edge will be of length $\leq \Delta(G)/2$ and so on.

This prove the bound on $\Delta(T)$

Analysis

We now estimate the expected distance between any given pair uv

We claim that $E[d_T(uv)] \leq (c \log n \log \Delta(G)) \, d_G(uv)$

let p = probability that uv is cut at level 0

From the random partitioning process
$p \leq c' \log n \, d_G(uv)/\Delta(G)$ for some constant c'

We verify claim by using

$E[d_T(uv)] = p \, E[d_T(uv) \mid uv \text{ cut at level } 0] +$
$(1-p) \, E[d_T(uv) \mid uv \text{ not cut at level } 0]$

Analysis

Inductively

$E[d_T(uv) \mid uv \text{ not cut at level } 0] \leq (c \log n \log \Delta(G)/2)\, d_G(uv)$

since the diameter of graph reduces by factor of 2 in each level of recursion

and we have

$E[d_T(uv) \mid uv \text{ cut at level } 0] \leq \Delta(T) \leq 4\, \Delta(G)$

Therefore

$p\, E[d_T(uv) \mid uv \text{ cut at level } 0] \leq 4c' \log n\, d_G(uv)$

Analysis

We thus have that

$E[d_T(uv)]$

$\leq 4\,c' \log n\, d_G(uv) + (1-p)\,(c \log n \log \Delta(G)/2)\, d_G(uv)$

$\leq 4\,c' \log n\, d_G(uv) + (c \log n \log \Delta(G)/2)\, d_G(uv)$

$\leq c\,(\log n\,(1 + \log \Delta(G)/2)\, d_G(uv)$ for $c \geq 4c'$

$\leq c\,(\log n \log \Delta(G))\, d_G(uv)$

Obtaining $O(\log^2 n)$

In the analysis we note that the bound we can establish is $O(d \log n)$ where d is the depth of recursion. Since we have $d = O(\log \Delta(G))$ we obtained a bound of $O(\log \Delta(G) \log n)$

We modify the construction slightly as follows.

In the partitioning procedure, we wish to ensure that a pair uv has zero probability of being cut if $d_G(uv)$ is too small compared to $\Delta(G)$: too small is if $d_G(uv) \leq \Delta(G)/n^3$

If we do above then a pair uv participates only in $O(\log n)$ levels of recursion

Obtaining $O(\log^2 n)$

In the partitioning procedure, we wish to ensure that a pair uv is not cut if $d_G(uv)$ is too small compared to $\Delta(G)$: that is $d_G(uv) < \Delta(G)/n^3$

One way to do this is notice that in the random partitioning procedure, the probability that a pair uv with $d_G(uv) < \Delta(G)/n^3$ gets cut is at most $O(\log n/n^3)$. Thus with high probability no such pair actually gets cut! (the probability *any* such pairs gets cut is $< O(\log n /n)$)

We can stop the construction if we encounter such a bad event and repeat the process and we will have a good probability of success

Obtaining $O(\log^2 n)$

Another way to do this is to first shrink all vertices at distance $< \delta(G)/n^3$ into a super vertex and then do the random partitioning process. This will ensure that no such pair is cut. One can check that the other distances don't change too much because of the shrinking.

We do not explore the full details of above in this class.

Hierarchicaly Separated Trees

Notice that the trees generated by the algorithm had a nice property: the length of the edges decreased by a factor of at least 2 on any path from the root to a leaf

Such trees are called hierarchically well-separated (HSTs) and this additional property is often useful in applications

We can change the factor 2 to any constant k at the cost of increasing the distance approximation by a factor of k

Using some care one can also ensure that all edges at the same level have the same length

Note that all vertices of G are at leaves of the tree

Ultrametrics

A metric on V is called an *ultrametric* if there is a tree T s.t

- V are the leaves of T
- $d(uv) = d_T(uv)$ for all u,v
- all root – leaf paths have the same length in T

One can transform an HST into an ultrametric and thus we can obtain an ultrametric embedding for any finite metric space. This is sometimes useful in applications.

Applications

There are many applications of the tree embedding theorem in approximation and online algorithms

It allows one to reduce certain class of problems on graphs to problems on trees and it is much easier to design algorithms on trees!

The ratio one obtains using this approach might not be optimal but gives one a quick upper bound on the approximation ratio. For some problems this is still the best approach known

Steiner Forest

Given edge weighted graph $G = (V,E)$ and pairs of nodes $s_1t_1, s_2t_2, \ldots, s_kt_k$

Find min-cost subset $E' \subseteq E$ s.t each s_it_i is connected in $G[E']$

Generalizes Steiner tree problem

One can obtain a $2(1-1/k)$ approximation for Steiner forest using LP based methods but we can obtain a trivial algorithm using tree embeddings

Steiner Forest via Tree embeddings

Given G, we use the randomized algorithm to pick T (without looking at the pairs!)

Now we connect the the pairs in T. For each $s_i t_i$ there is a unique path P_i in T and there is no choice, we have to pick P_i

Claim: the expected cost of a Steiner forest for given pairs in T is $O(\log^2 n)$ OPT

Results in an $O(\log^2 n)$ approximation

Steiner Forest via Tree embeddings

Given G, we use the randomized algorithm to pick T (without looking at the pairs!)

Now we connect the the pairs in T. For each $s_i t_i$ there is a unique path P_i in T and there is no choice, we have to pick P_i

Claim: the expected cost of a Steiner forest for given pairs in T is $O(\log^2 n)$ OPT

Results in an $O(\log^2 n)$ approximation

Steiner Forest via Tree embeddings

It is useful to see the nitty gritty details of the approximation resulting from tree embeddings in the simple Steiner forest case

First we prove that the length of the expected cost of the solution for the problem on T is $O(\log^2 n)$ OPT

Let $E^* \subseteq E$ be an optimum solution to the problem

For each $uv \in E^*$ we add the path P_{uv} in T where P_{uv} is the unique path connecting u to v in T

Steiner Forest via Tree embeddings

Let $E^* \subseteq E$ be an optimum solution to the problem

For each $uv \in E^*$ we add the path P_{uv} in T where P_{uv} is the unique path connecting u to v in T

We claim that $\cup_{uv \in E^*} P_{uv}$ is a feasible solution to the Steiner forest problem induced on T

To see this, consider a path P_i that connects s_i and t_i in E^*. Say $P_i = e_1, e_2, \ldots, e_h$ where $e_i = u_i v_i$

Since we add $P_{u_i v_i}$ in T for each i, it follows that $s_i t_i$ will be connected in $\cup_{uv \in E^*} P_{uv}$

Steiner Forest via Tree embeddings

Let $E^* \subseteq E$ be an optimum solution to the problem

For each $uv \in E^*$ we add the path P_{uv} in T where P_{uv} is the unique path connecting u to v in T

What is the expected cost of $\cup_{uv \in E^*} P_{uv}$?

We can upper bound this cost by
$\sum_{uv \in E^*} cost(P_{uv}) = \sum_{uv \in E^*} d_T(uv)$
But $Expec[d_T(uv)] \leq O(\log^2 n)\, d_G(uv) \leq O(\log^2 n)\, c(uv)$

Therefore, by linearity of expectations, the expected cost of the solution in T is at most $O(\log^2 n)\, c(E^*)$

Steiner Forest via Tree embeddings

Now we see how can translate a solution for the problem in T to a solution in the original graph of no more cost

In order to do this we do the following. Notice that the vertices of V(G) are the leaves of T. We associate with each internal node x of T with a node x' of V(G) where x' is some *arbitrary* leaf in the subtree of T rooted at x

We each edge xy of T we associate a *shortest path* $P_{x'y'}$ connecting x' and y' in G

Steiner Forest via Tree embeddings

We each edge xy of T we associate a *shortest path* $P_{x'y'}$ connecting x' and y' in G

We observe that the length of the edge xy in T is at least as large as the length of the path $P_{x'y'}$ in G

(Why? Prove this inductively. Notice that we set the rr_i edge length in the construction of T to be $\Delta(G)$)

Now any set A of edges in T can be associated with a set of edges in A' in G, where $A' = \cup_{uv \in A} P_{u'v'}$

Further any two vertices s,t that are connected by A in T will be connected by A' in G

And $cost(A') \leq cost(A)$ by the above observation