CSCI 3100

More approximation: using geometry
Announcements

- Oral grading on Friday
- Next HW due after Wednesday break
Traveling Salesman (TSP)

Given \( n \) cities with pairwise distances between them, find the shortest cycle visiting all cities.

This is NP-hard:

Reduce Ham Cycle to TSP.

An unweighted graph, goal is to see if there is a simple cycle visiting all vertices.

Build \( G' \): same vertices
Add all edges possible:
\[
\begin{align*}
&w(e) = 1 \text{ in } G' \text{ if } e \in G \\
&w(e) = 2 \text{ in } G' \text{ if } e \notin G
\end{align*}
\]
Set \( k = n \).
Note: Nothing special about $7+2$ here!

In fact, I can use different values + show even approximating TSP is hard:

Ex: Let $G' = \begin{cases} w(e) = 1 & \text{if } e \in G \\ w(e) = n+1 & \text{if } e \notin G \end{cases}$

Here: Still have $G$ has Ham. cycle

$\iff G'$ has TSP tour of length $n$

But: If no Hamiltonian cycle, $G'$ only has tours of length $\geq 2n$

So a 2-approx alg for TSP would give exact soln to Ham. cycle.
Thm: For any polynomial $f(n)$, there is no $f(n)$-approx algorithm for TSP (unless $P=NP$).

pf: Build $G'$:

$$w(e) = 1 \text{ if } e \in G$$

$$w(e) = p(n) \text{ if } e \notin G$$
However:
These are strange $G'$ graphs:

If we have extra structure, can still approximate!

What are some common sources of graphs we might want to solve?

Roads!
**Def**: Triangle inequality:

For any \( u, v, w \in V \),
\[
l(u, w) \leq l(u, v) + l(v, w)
\]

\( u \)

\( v \)

\( w \)

**Note**: Always get this for certain graphs:

Geometric graphs (embedded in \( \mathbb{R}^2 \))
Thm: If $G$ satisfies the triangle inequality, can compute a $2$-approx for TSP.

Idea:

- Start w/ MST
  - After shortcuts, tour $\leq 2 \cdot$ MST
  - Shorter than other $2$ (by inequality)
The picture:

Steps:
- Compute MST
- Get DFS ordering
- Do shortcuts to avoid repeated edges + vertices
Claim: This algorithm is a 2-approximation.

\[ \text{pf: Let } \text{OPT be cost of the best TSP tour.} \]
\[ \text{Let } \text{MST be the total weight of the min. spanning tree.} \]
\[ \text{Our algorithm's TSP length } = A. \]

Bound A:

1. \( A \leq 2 \times \text{MST} \)  
   (see prior slide; use D-ineq.)
On the other hand, OPT (best tour) is a cycle. If you delete any edge from OPT, what do you have? path!

⇒ Since any path is also a potential min. spanning tree,

② MST ≤ OPT

A ≤ 2 MST ≤ 2 OPT

Result: we get a 2-approx.
Another: Clustering

Given a set of points $P = \{p_1, p_2, \ldots, p_n\}$ in $\mathbb{R}^2$ and an integer $k$, find a set of $k$ circles that contain all $n$ points such that the radius of the largest circle is as small as possible.

$k = 3$

Formally, find $C = \{c_1, \ldots, c_k\}$ of centers, such that

\[
\text{cost} = \max_i \min_j |p_i - c_j| \text{ is minimized.}
\]
Why?

Assign each point to closest center.
(This is the "circle").
Radius of circle is the max distance of center to point assigned to it.
This problem is NP-Hard, even to approximate with a factor of \( \approx 1.8 \).

However, simple and natural greedy strategy (Gonzalez '85), which gives a 2-approx.

Idea:

- Suppose I start with a point I've chosen to be a center.

What's a natural next point to grab? Furthest point → make this a center.
Algorithm

\[ \text{GONZALEZKCENTER}(P, k): \]

\[
\begin{align*}
&\text{for } i \leftarrow 1 \text{ to } n \\
&\quad d_i \leftarrow \infty \\
&\quad c_1 \leftarrow p_1 \\
&\text{for } j \leftarrow 1 \text{ to } k \\
&\quad r_j \leftarrow 0 \\
&\text{for } i \leftarrow 1 \text{ to } n \\
&\quad d_i \leftarrow \min\{d_i, |p_i, c_j|\} \\
&\quad \text{if } r_j < d_i \\
&\quad\quad r_j \leftarrow d_i; c_{j+1} \leftarrow p_i \\
&\text{return } \{c_1, c_2, \ldots, c_k\}
\end{align*}
\]

Runtime: \( O(nk) \)

The first five iterations of Gonzalez’s k-center clustering algorithm.
Thm: Gonzales k-center gives a 2-approx.

Pf: Let OPT be optimal k-center solution. Let ci + ri be the ith centers added by our algorithm.

Ci , C2 , C3 , ..., Ck

(If set of points with largest radius in best set of k-circles)

r1 , r2 , r3 , ..., rk

Cj is at least rj-1 away from C1, C2, ..., Cj-1

r1 ≥ r2 ≥ ... ≥ rk-1

⇒ |Cj, Cj| ≥ rk
Consider what $C_{k+1}$ would have been (rk away from one of $C_1 \ldots C_{k}$).

$C_1 \ldots C_{k+1}$ are $k+1$ points (from $P$).

OPT soln covered these.

OPT $\leq 2 \cdot$ OPT by $\Delta$-ineq.

$r_k \leq 2 \cdot$ OPT by cost of my soln.