Part 2 Approximation Template

Metric-TSP

- A metric (V, d) with $d: V \times V \to \mathbb{R}^{\geq 0}$ satisfies
  - $d(u,v) = d(v,u)$ (Symmetry)
  - $d(u,v) = 0$ iff $u = v$
  - $d(u,v) \leq d(u,x) + d(x,v)$ (Triangle Ineq.)
- Represented by a ***complete*** undirected graph $G = (V, E, d)$ with $E = V \times V$ and $d: V \times V \to \mathbb{R}^{\geq 0}$.
- **Problem:** Find a "tour" of the vertices $(v_0, v_1, \ldots, v_{n-1}, v_0)$ minimizing $\sum_{i=0}^{n-1} d(v_i, v_{i+1 \bmod n})$.

Hamiltonian cycle!

- **Problem:** An ***Euler tour*** of $G' = (V', E')$ is a path $(u_0, u_1, \ldots, u_{|E'|-1}, u_0)$ that visits every ***edge*** exactly once.
- **Theorem:** G' has an Euler tour iff G' is connected and all vertices have **even** degree.

- **Problem:** Given $G = (V', E', d)$, a ***minimum spanning tree*** T is a tree spanning V' minimizing $d(T) = \sum_{e \in E(T)} d(e)$.
- **Lemma.** $d(MST) \leq d(TSP)$.

(if not get better MST by removing last edge to TSP)

→ so really

$d(MST) \leq (1 - \frac{1}{n})\, d(TSP)$

→ remove longest edge | $\frac{1}{n}$ by pidgeonhole

How to make the graph have an Euler tour (Eulerian)?

Soln: Double every edge! in MST.

- TSP Approximation Algorithm:

1. Find MST T of $G = (V, E, d)$.
2. $[2T]$ = double up every edge in T.
3. $C_0 = (u_0, \ldots, u_{2(n-1)-1}, u_0)$ = an Euler tour of $[2T]$.
4. $C = (u_{i_0}, u_{i_1}, \ldots, u_{i_{n-1}}, u_{i_0})$ splice out redundant occurrances in C_0

at most twice TSP

→ ≤ 2 TSP by triangle

→ 2 approximation.

- **Problem:** Given $G' = (V', E', d)$, a ***min-cost perfect matching*** is a matching $M \subset E'$ minimizing $d(M) = \sum_{e \in M} d(e)$
- **Lemma.** If $|V'|$ is even, $d(\text{MCPM}) \leq \frac{1}{2} \cdot d(\text{TSP})$.

→ can decompose tsp into 2 matchings.

- Better TSP Approximation [Christofides 1976]

1. Find MST T of $G = (V, E, d)$.
2. V_{odd} = set of vertices that have odd degree in T.
3. M = min-cost perfect matching in graph induced by V_{odd}
4. $C_0 = (u_0, \ldots, u_{2(n-1)-1}, u_0)$ = an Euler tour of $T \cup M$.
5. $C = (u_{i_0}, u_{i_1}, \ldots, u_{i_{n-1}}, u_{i_0})$ splice out redundant occurrances in C_0

even number of odd edges by handshake!

→ $\frac{3}{2}$ approx!

Partition.

k-Terminal Cut

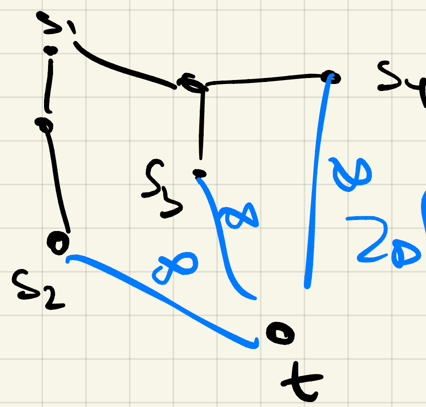
- ***Clustering*** is a big problem in big data sets.
- **Problem.** Given undirected graph $G = (V, E, c)$, $c: E \to \mathbb{R}^+$, and k terminals $s_1, s_2, \dots, s_k \in V$, partition V into $V_1, \dots, V_k$ such that $s_i \in V_i$ so that the ***cost*** is minimized:

$$\sum_{\substack{\{u,v\}\in E:\\ u\in V_i, v\in V_j, i\neq j}} c(u,v)$$

- **Obs.** When $k = 2$ the problem is in P. Why?
- **Thm.** When $k \geq 3$ the problem is NP-hard. (Hard exercise.)
- How can we use a max-flow/min-cut algorithm to find an approximately optimal solution?

→ max flow / min cut

→ look at all k max-flow min cut as such



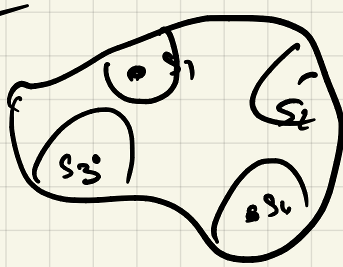
2 → add these

isolate each terminal ← → get best cut $s_1 \to \{s_2, s_3, s_4\}$

$s_2 \to \{s_1, s_3, s_4\}$

⋮

→ look like



→ guaranteed that cuts have no overlap.

→ then assign the ones in no man's land to s_4.

let $C_1, \dots, C_k$ all of the cuts.

→ WLOG costliest

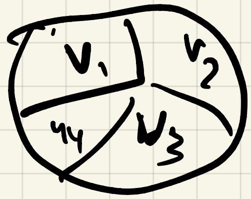
$$C = C_1 \cup C_2 \cup \dots \cup C_k$$

$$c(k) \leq \left(1 - \frac{1}{k}\right) \sum_{i=1} c(C_i) \underset{\text{double counting}}{\leq} 2\left(1 - \frac{1}{k}\right) \underbrace{\sum c(OPT_i)}_{\text{OPT soln}}$$

→ 2 approx.

→ we will bring this to 1.5 soln by rounding LP.

LP Rounding!



let vertex v $\quad x_v^i = \begin{cases} 1 & \text{if } v \in V_i \\ 0 & \text{otherwise.} \end{cases}$

const in one. $\quad 1 = \sum_{i=1}^{k} x_v^i$

$x_{s_i}^i = 1$ $\quad$ Constr

$0 \le x_v^i \le 1 \qquad x_v^i \in \mathbb{Z}$

$d(u,v) = \frac{1}{2}\sum_i |x_u^i - x_v^i|$ $\quad$ Objective: $\text{Min} \sum_{(u,v)} c(u,v)\, d(u,v)$

$\hookrightarrow 2$ if in diff

$\hookrightarrow$ slight cheating with $|\cdot|$

but can fix.

$d(u,v) = \frac{1}{2}\sum_i x_{uv}^i$

$x_{uv}^i \ge x_u^i - x_v^i$

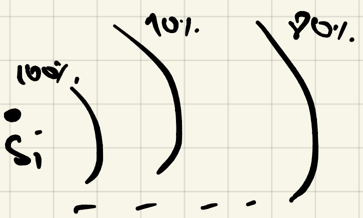
$x_{uv}^i \ge x_v^i - x_u^i$

forces to consider x_{uv}^i an abs val cons.

let x^* opt to LP (drop integrality).

$\hookrightarrow$ how do we round?. $\quad B(s_i, \rho) \to$ ball s_i radius ρ

$\hookrightarrow v$ so that $x_v^i \ge \rho$



$\rho \in [0,1)$ uniformly at random

pick σ permutation S_k unif at rand.

For $i = 1 \dots k-1$

$V_{\sigma(i)} = B(s, \rho) \setminus \bigcup_{j=1}^{i} V_{\sigma(j)}$

$V_{\sigma(k)} = V \setminus \bigcup_{j=1}^{i-1} V_{\sigma(j)}$

C = all edges crossing $V_1 \dots V_k$.

$E(c(C))$

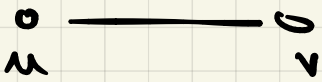
$= \sum_{(u,v)} c(u,v)\, P(u,v \text{ in diff sets})$

$\downarrow$ $\boxed{\text{WTS}}$

$\le \sum c(u,v) \frac{3}{2} d(u,v)$

$\le \frac{3}{2}\text{OPT}(LP) \le \frac{3}{2}\text{OPT}(\text{ILP})$

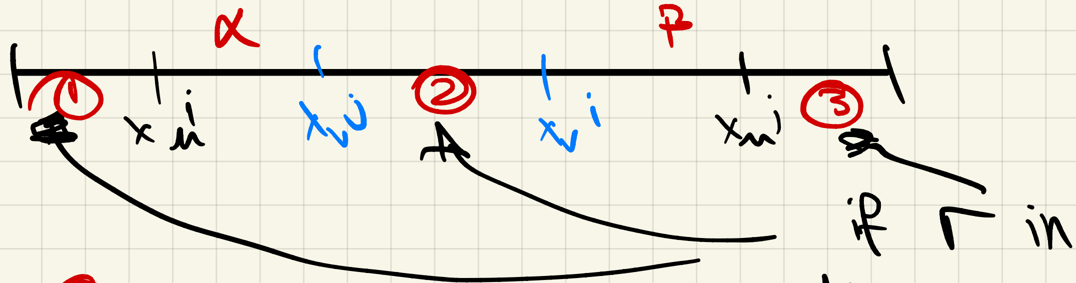
Conclude



let u,v differ in a word (simplifying assump)

↳ i, j

$$x_u^i + x_u^j = x_v^i + x_v^j$$



if ③ end up in default partition then.
or same one

↳ they don't separate

↳ behave id.

⇒ don't cut edge same for ①, ②, ③

if $\rho \in \beta$ ⇒ u ⇒ u belongs to j

v ⇒ not in j's partition!

⇒ if $\rho \in \alpha$ ⇒ if j goes first

↳ both in j

if i goes first then not

↳ v in i but not j

⇒ $P(u,v)$ cut

↳ $P(i$ precedes $j)$ (50/50)

& $\rho \in \alpha$

$+ P(\rho \in \beta)$

$$\Rightarrow \boxed{|\beta| + \frac{|\alpha|}{2}} = \frac{3}{2} d(u,v).$$

↳ since more cases.