Approximation Algorithms

Tapas Kumar Mishra
NIT Rourkela

Part 1
Introduction

Source: *Approximation Algorithms* (Vazirani, Springer Press)

Why approximation algorithms?

Task: Solve **NP**-hard optimization problem A
$\rightarrow$ no efficient algorithm (unless $\mathbf{NP} = \mathbf{P}$)

Possible approaches:

- exponential time algorithms $\rightarrow$ some theory but too slow and no lower bounds
- heuristic $\rightarrow$ fast, easy but no guarantee, not much theory
- approximation algorithms $\rightarrow$ rich theory in many cases good lower bounds

Running times: n = number of objects in instance, B biggest appearing number, $\varepsilon > 0$ constant

- exponential: $2^n, n \cdot B$
- polynomial: $n^2, n^{100}, n \cdot \log B, n \cdot 2^{1/\varepsilon}, n^{O(1/\varepsilon)^{O(1/\varepsilon)}}$

Basic definitions

Definition

Let Π be an optimization problem and I is instance for A. Then $OPT_\Pi(I)$ is the value of the optimum solution.

Definition

Let $\alpha \geq 1$. A is an α-approximation algorithm for a minimization problem Π if

$$A(I) \leq \alpha \cdot OPT_\Pi(I) \quad \forall \text{ instances } I$$

where $A(I)$ is the value of the solution, that A returns for I.

- Typical values for α: $1.5, 2, O(1), O(\log n)$
- Usually we omit Π and I in $OPT_\Pi(I)$
- For a maximization problem: $A(I) \geq \frac{1}{\alpha} \cdot OPT_\Pi(I)$
- Attention: Sometimes in literature $\alpha < 1$ for maximization problems. For example $\frac{1}{2}$-apx means $A(I) \geq \frac{1}{2} OPT_\Pi(I)$

Definition PTAS

Definition

A_ε is a polynomial time approximation scheme (PTAS) for a minimization problem Π if

$$A_\varepsilon(I) \leq (1+\varepsilon) \cdot OPT(I) \quad \forall \text{ instances } I$$

and for every fixed $\varepsilon > 0$, the running time of A_ε is polynomial in the input size.

Typical running times: $O(n/\varepsilon), 2^{1/\varepsilon} n^2 \log^2(B), n^{O(1/\varepsilon)^{O(1/\varepsilon)}}$

Definition FPTAS

Definition

A_ε is a fully polynomial time approximation scheme (FPTAS) for a minimization problem Π if for every $\varepsilon > 0$

$$A_\varepsilon(I) \leq (1+\varepsilon) \cdot OPT(I) \quad \forall \text{ instances } I$$

and the running time of A_ε is polynomial in the input size and $1/\varepsilon$.

- Typical running time: $O(n^3/\varepsilon^2)$

Part 2
Steiner tree

Source: *Approximation Algorithms* (Vazirani, Springer Press)

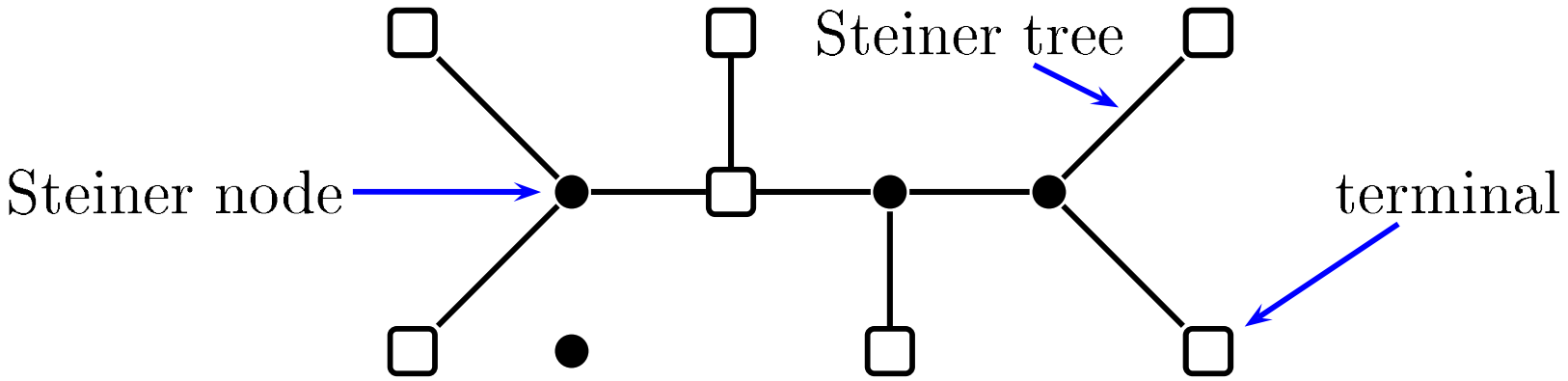
Steiner Tree

Problem: STEINER TREE

- Given: Undirected graph $G = (V, E)$, metric cost function $c : E \to \mathbb{Q}_+$, terminals $R \subseteq V$
- Find: Minimum cost tree T connecting all terminals R:

$$OPT = \min\{c(T) \mid T \text{ spans } R\}$$

- $c(T) := \sum_{e \in T} c_e$
- metric: $\forall u, v, w \in V : c_{uw} \leq c_{uv} + c_{vw}$ (triangle inequality)



Steiner tree (2)

Fact

If $R = V$, then Steiner Tree is just the Minimum Spanning Tree Problem which can be solved optimally by picking greedily the cheapest edges (without closing a cycle).

Algorithm:

(1) Compute the minimum spanning tree T on R

(2) Return T

Theorem

The algorithm gives a 2-approximation.

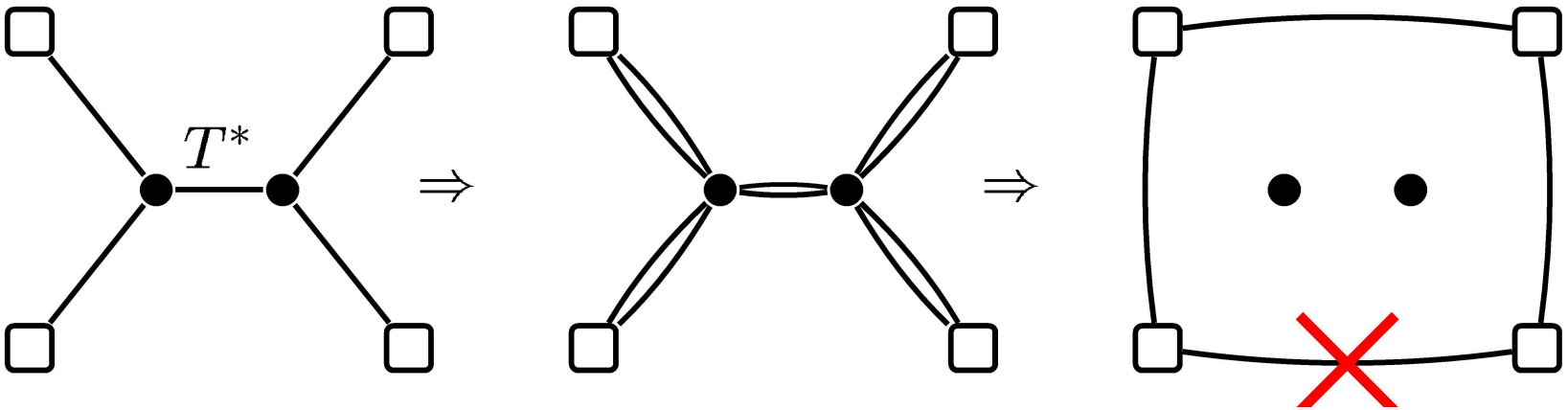
Proof of approximation guarantee

- Claim: $\exists$ spanning tree of cost $\leq 2 \cdot OPT$
- Let T^* be optimum Steiner tree
- Double the edges of T^*
- Observe: Degrees now even $\Rightarrow \exists$ Euler tour $\mathcal{E}$ visiting each terminal

Theorem (Euler)

Given an undirected, connected graph $G = (V, E)$. Then G has an Euler tour (tour containing each edge exactly once) if and only if $|\delta(v)|$ is even for all $v \in V$.

- Shortcut $\mathcal{E}$ such that each terminal is visited once
- Remove an edge $\Rightarrow$ spanning tree of cost $\leq 2 \cdot c(T^*)$ $\square$



State of the art

Known results:

- There is a 1.39-approximation.
- For quasi-bipartite graphs (no Steiner nodes incident): 1.22-apx
- No $< \frac{96}{95}$-apx unless $\mathbf{NP} = \mathbf{P}$.

Part 3
k-Center

Source: *Approximation Algorithms* (Vazirani, Springer Press)

k-Center

Problem: k-Center

- Given: Undirected, metric graph $G = (V, E)$, $k \in \mathbb{N}$. Define

$$\ell(v, F) := \min_{u \in F} c_{uv}$$

- Find: k many centers $F \subseteq V$ that minimize the maximum distance from any $v \in V$ to the nearest center:

$$OPT = \min_{F \subseteq V, |F| = k} \max_{v \in V} \{\ell(v, F)\}$$

The algorithm

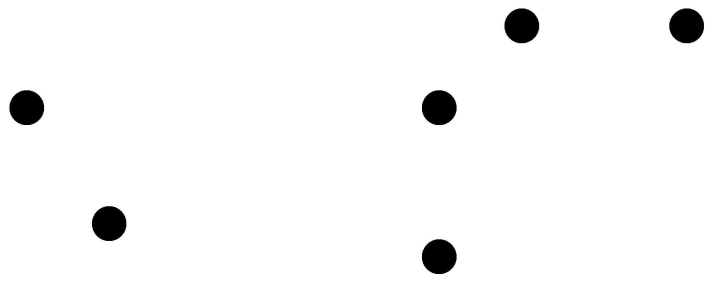
Algorithm:

(1) Guess $OPT \in \{c_{uv} \mid u, v \in V\}$

(2) $F := \emptyset$

(3) REPEAT

(4) IF $\exists v \in V : \ell(v, F) > 2 \cdot OPT$ THEN $F := F \cup \{v\}$
ELSE RETURN F



The algorithm

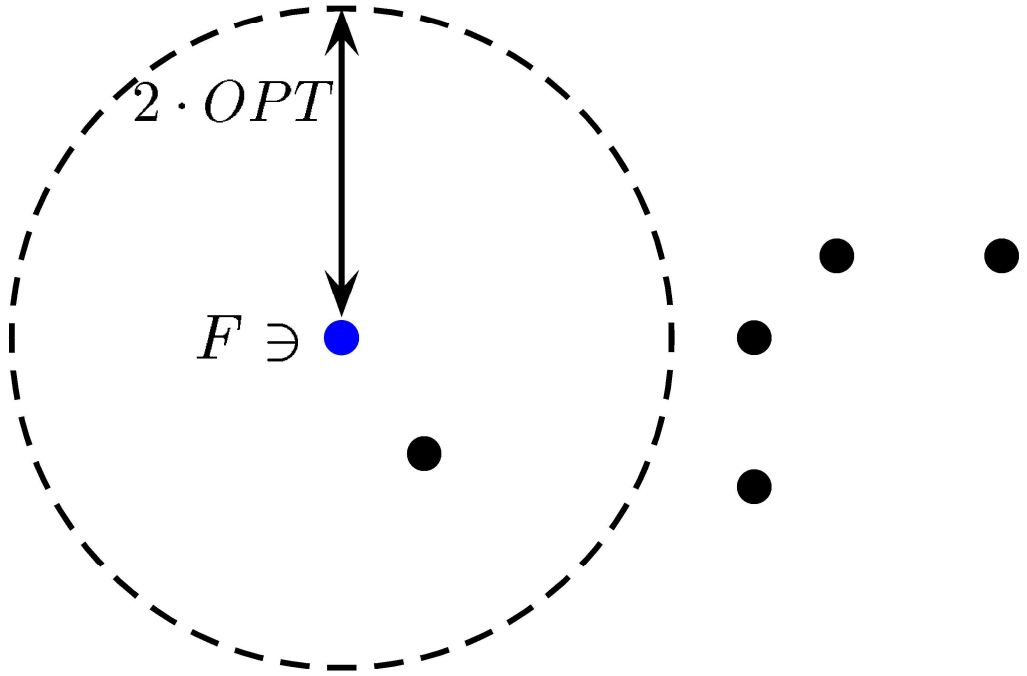
Algorithm:

(1) Guess $OPT \in \{c_{uv} \mid u, v \in V\}$

(2) $F := \emptyset$

(3) REPEAT

(4) IF $\exists v \in V : \ell(v, F) > 2 \cdot OPT$ THEN $F := F \cup \{v\}$
ELSE RETURN F



The algorithm

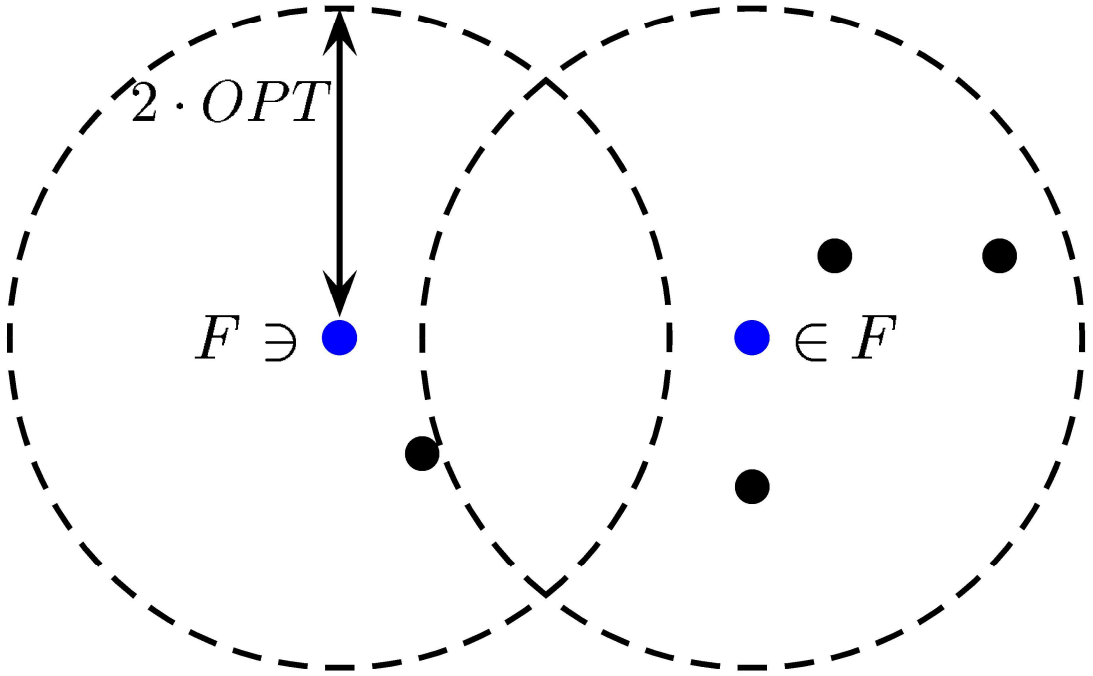
Algorithm:

(1) Guess $OPT \in \{c_{uv} \mid u, v \in V\}$

(2) $F := \emptyset$

(3) REPEAT

(4) IF $\exists v \in V : \ell(v, F) > 2 \cdot OPT$ THEN $F := F \cup \{v\}$
ELSE RETURN F



Guessing

For simplicity we sometimes **guess** parameters:

Algorithm with guessing:

(1) Guess a parameter m

(2) ... compute a solution $\mathcal{S}$ using m ...

(3) return $\mathcal{S}$

Algorithm without guessing:

(1) FOR all choices of m DO

  (2) ... compute a solution $\mathcal{S}(m)$ ...

(3) return the best found solution $\mathcal{S}(m)$

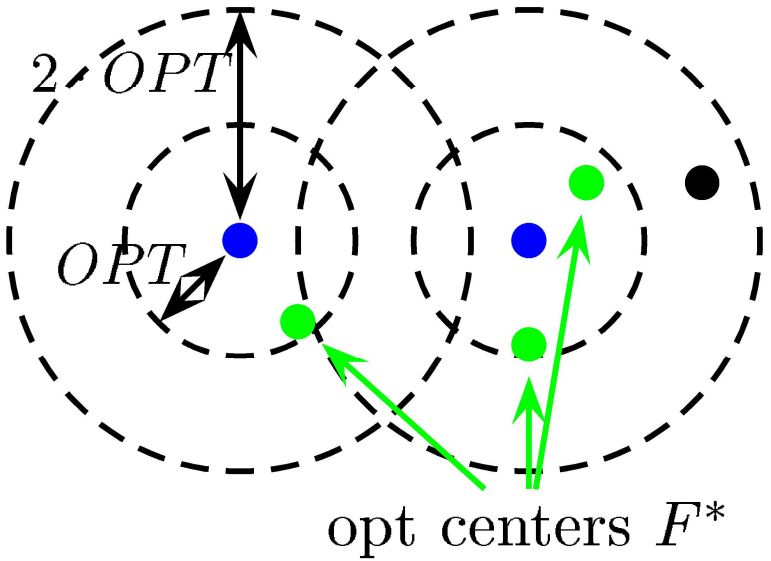
- Still polynomial if the domain of m is polynomial
- Typical guesses: OPT, $O(1)$ many nodes in a graph

The analysis

Theorem

One has $|F| \leq k$ and $\ell(v, F) \leq 2 \cdot OPT$ for all $v \in V$.

- $\ell(v, F) \leq 2 \cdot OPT$, otherwise algo would not have stopped.
- Remains to show $|F| \leq k$.
- Let $F^* \subseteq V, |F^*| = k$ be optimum solution.
- Observe: $c_{uv} > 2 \cdot OPT \; \forall u, v \in F : u \neq v$
- Hence the centers in F^* that serve u and v must be different $\Rightarrow |F| \leq |F^*| \leq k$.

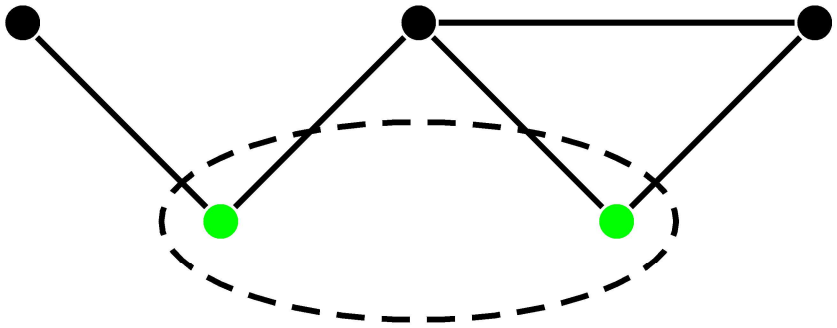


Dominating Set

Problem: Dominating Set

- Given: Undirected graph $G = (V, E)$
- Find: Dominating set $U \subseteq V$ of minimum size

$$OPT_{DS} = \min\{|U| \mid U \subseteq V, U \cup \bigcup_{u \in U} \delta(u) = V\}$$



dominating set

Theorem

Given (G, k), it is **NP***-hard to decide, whether $OPT_{DS} \leq k$.*

Hardness of k-Center

Theorem

Unless $\mathbf{NP} = \mathbf{P}$, *for all* $\varepsilon > 0$, *there is no* $(2-\varepsilon)$*-approximation algorithm for* k-Center.

- Let (G, k) be DominatingSet instance.
- Suppose A is a $(2-\varepsilon)$-algorithm for k-Center
- Define complete graph G' on nodes V with

$$c(u,v) := \begin{cases} 1 & (u,v) \in E \\ 2 & \text{otherwise} \end{cases}$$

- $\exists$ DS of size $\leq k \Rightarrow k$-Center solution with value 1
- $\exists k$-Center solution with value $\leq 1 \Rightarrow \exists$ DS of size $\leq k$
- Run A on G':
  - $A(G') < 2 \Rightarrow A(G') = 1 \Rightarrow$ answer to DS instance is YES
  - $A(G') \geq 2 \Rightarrow$ answer is NO

□

Part 4
Traveling Salesman Problem

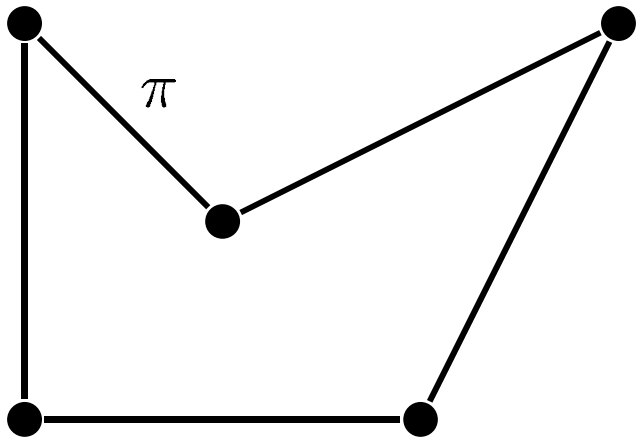
Source: *Approximation Algorithms* (Vazirani, Springer Press)

TSP

Problem: Traveling Salesman Problem (Tsp)

- Given: Undirected graph $G = (V, E)$ with metric cost $c : E \to \mathbb{Q}_+$
- Find: Minimum cost tour visiting all nodes

$$\min_{\text{tour } \pi: V \to V} \left\{ \sum_{v \in V} c(v, \pi(v)) \right\}$$



A 2-approximation for TSP

Algorithm:

(1) Compute an MST T on G

(2) Double the edges in T

(3) Compute Euler tour $\mathcal{E}$ using edges in T

(4) Shortcut to obtain a tour π

Theorem

Algorithm yields a 2-apx.

- Let π^* be optimum tour
- $\exists$ a spanning tree on G of cost $c(T) \leq OPT$ (just delete an arbitrary edge from π^*)
- Degrees are even after doubling, hence $\mathcal{E}$ exists and $c(\mathcal{E}) \leq 2 \cdot OPT$
- $c(\pi) \leq 2 \cdot OPT$ (G is metric, hence shortcutting does not increase the cost) □

A 3/2-approximation for TSP

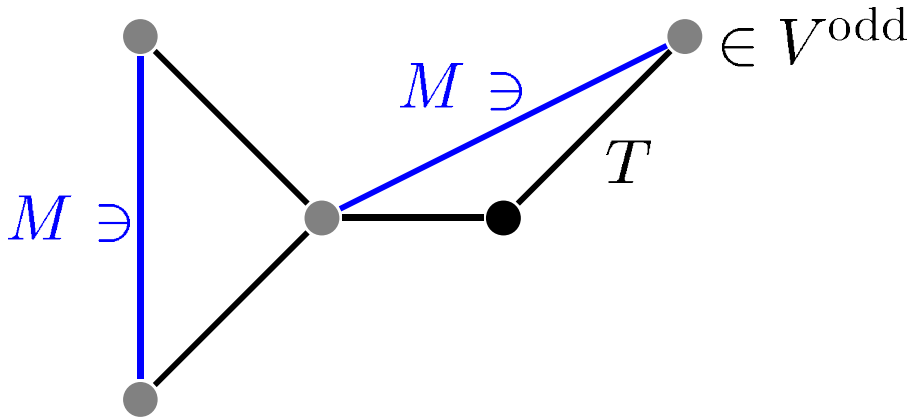
Algorithm (Christofides):

(1) Compute an MST T

(2) Find min cost perfect matching M on nodes $V^{\text{odd}} \subseteq V$ with odd degree in T

(3) Find Euler tour in $T \cup M$.

(4) Return π obtained by shortcutting the Euler tour



Reminder

A perfect matching in an undirected graph $G' = (V', E')$ is an edge set $M \subseteq E'$ with $|\delta_M(v)| = 1 \; \forall v \in V'$. The cheapest perfect matching can be found in poly-time.

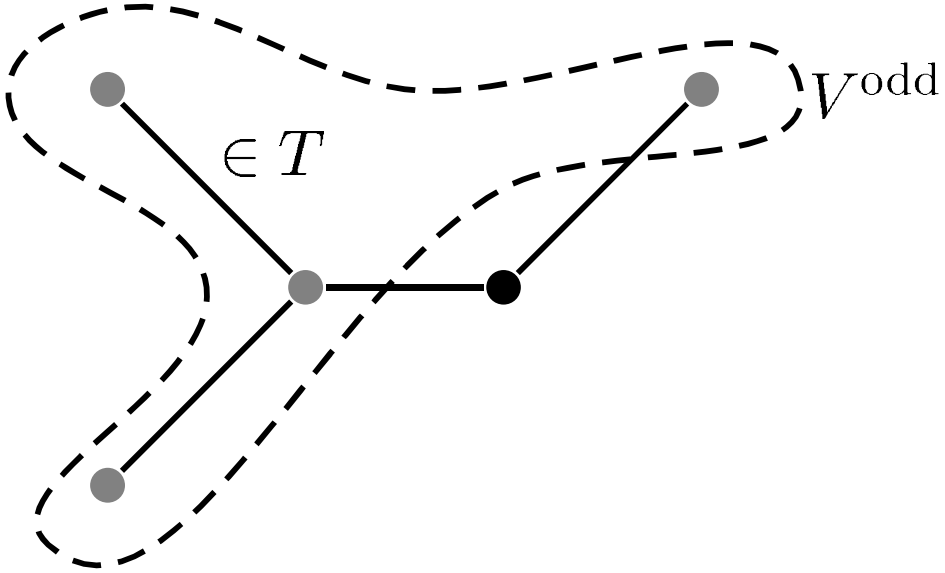
A 3/2-approximation for TSP (2)

Theorem

The algorithm gives a 3/2-apx.

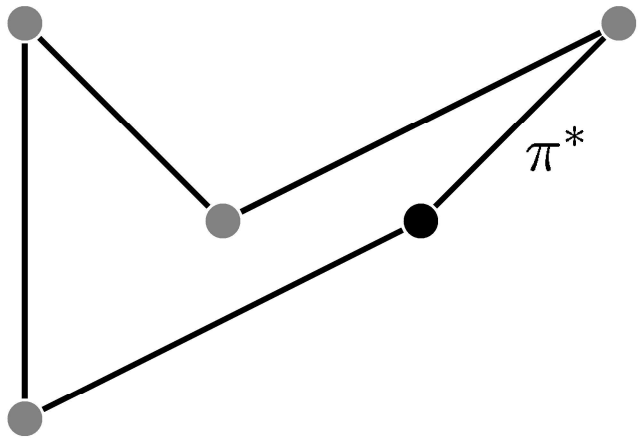
- Again $c(T) \leq OPT$
- $V^{\text{odd}} := \{v \in V \mid |\delta_T(v)| \text{ odd}\}$.
- Claim: $|V^{\text{odd}}|$ is even because

$$|V^{\text{odd}}| \equiv_2 \sum_{v \in V^{\text{odd}}} |\delta_T(v)| \equiv_2 \sum_{v \in V} |\delta_T(v)| \equiv_2 0$$



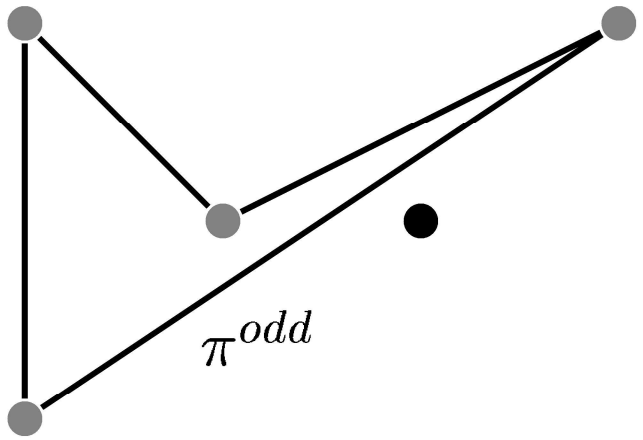
A 3/2-approximation for TSP (3)

- Let π^* be optimum tour. Obtain shortcutted tour π^{odd} on V^{odd}: $c(\pi^{\text{odd}}) \leq OPT$.
- Partition π^{odd} into 2 matchings M_1, M_2 on V^{odd}
- Let $M \in \{M_1, M_2\}$ be the cheaper of both matchings
- $c(M) \leq \frac{1}{2} c(\pi^{\text{odd}}) \leq \frac{1}{2} OPT$
- In $T \cup M$ all nodes have even degree, hence $T \cup M$ contains an Euler tour of cost $\leq c(T) + c(M) \leq \frac{3}{2} OPT$.



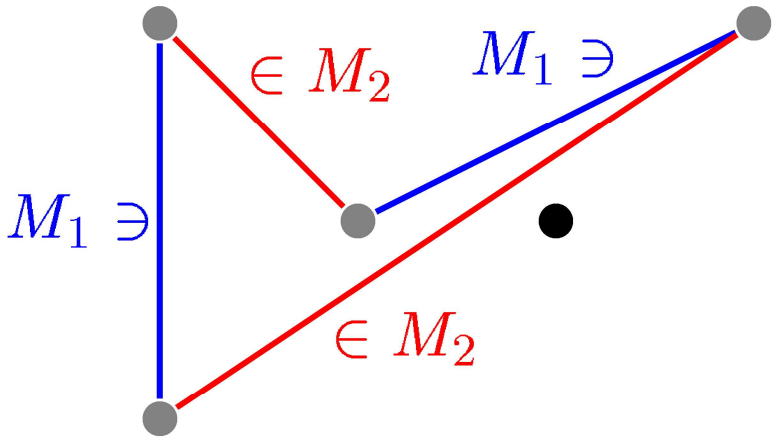
A 3/2-approximation for TSP (3)

- Let π^* be optimum tour. Obtain shortcutted tour π^{odd} on V^{odd}: $c(\pi^{\text{odd}}) \leq OPT$.
- Partition π^{odd} into 2 matchings M_1, M_2 on V^{odd}
- Let $M \in \{M_1, M_2\}$ be the cheaper of both matchings
- $c(M) \leq \frac{1}{2} c(\pi^{\text{odd}}) \leq \frac{1}{2} OPT$
- In $T \cup M$ all nodes have even degree, hence $T \cup M$ contains an Euler tour of cost $\leq c(T) + c(M) \leq \frac{3}{2} OPT$.



A 3/2-approximation for TSP (3)

- Let π^* be optimum tour. Obtain shortcutted tour π^{odd} on V^{odd}: $c(\pi^{\text{odd}}) \leq OPT$.
- Partition π^{odd} into 2 matchings M_1, M_2 on V^{odd}
- Let $M \in \{M_1, M_2\}$ be the cheaper of both matchings
- $c(M) \leq \frac{1}{2} c(\pi^{\text{odd}}) \leq \frac{1}{2} OPT$
- In $T \cup M$ all nodes have even degree, hence $T \cup M$ contains an Euler tour of cost $\leq c(T) + c(M) \leq \frac{3}{2} OPT$.



Open Problems on TSP

Open Problem

- Is there a $< 3/2$-apx for TSP?
- Held-Karp LP relaxation is conjectured to have integrality gap $4/3$.
- No $(\frac{5381}{5380} - \varepsilon)$-apx even if $c_e \in \{1, 2\}$

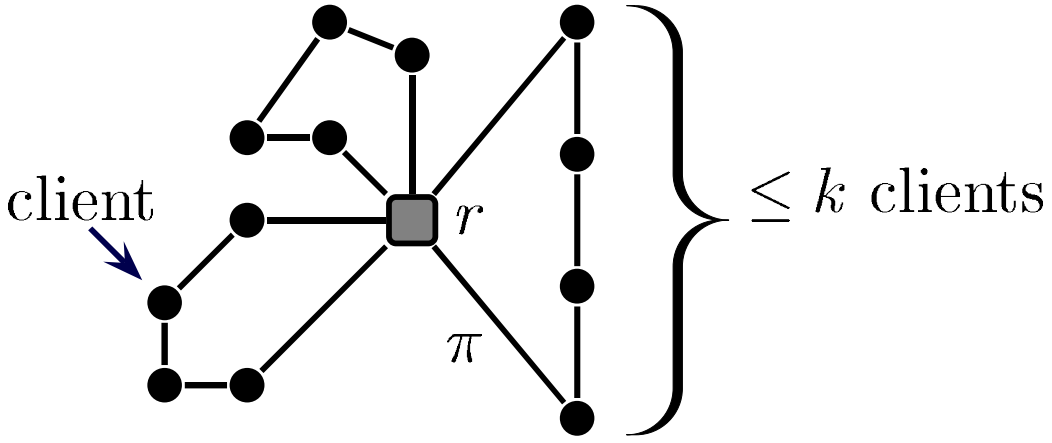
Part 5
The Capacitated Vehicle Routing Problem

Source: *Bounds and Heuristics for capacitated routing problems* (Haimovich, Rinnooy Kan)
http://www.jstor.org/stable/3689422

The Capacitated Vehicle Routing Problem

Problem: CVRP

- Given: Undirected graph $G = (C \cup \{r\}, E)$ with metric costs $c : E \to \mathbb{Q}_+$, depot r, clients C and vehicle capacity k
- Find: A tour π of minimal cost which visits all clients at least once, but must revisit the depot after each $\leq k$ client visits



Assume: $|C| = \mathbb{Z} \cdot k$ (otherwise add clients at the depot)

A 5/2-apx for CVRP

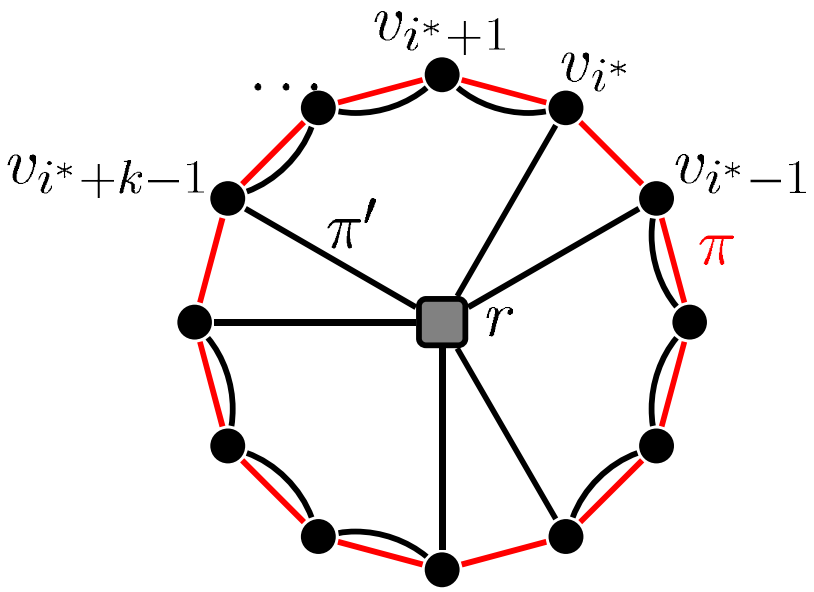
Algorithm:

(1) Compute a 3/2-approximate TSP tour π on clients

(2) Let $v_0, \ldots, v_{n-1}$ be clients in visiting order

(3) Choose randomly a starting node v_{i^*}

(4) Starting from v_{i^*} revisit r every k many clients (i.e. augment the tour with edges $r \to v_i, v_{i-1} \to r$ if $i \equiv_k i^*$) to obtain a CVRP solution π'



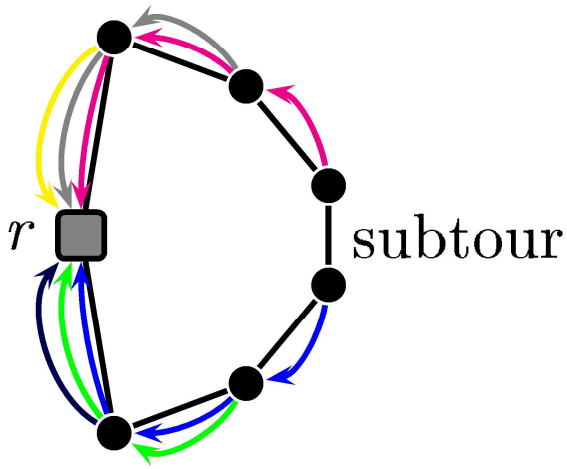
The analysis

Lemma

$E[APX] \leq \frac{5}{2} OPT$

- Opt. TSP tour costs $OPT_{\text{TSP}} \leq OPT$ hence $c(\pi) \leq \frac{3}{2} OPT$
- $\Pr[\text{need edge } (r, v_i)] = \frac{2}{k}$
- $E[APX] \leq c(\pi) + \frac{2}{k} \sum_{v \in C} c(r, v)$

- Look at a subtour in optimum CVRP solution. Send $k/2$ clients [counter-]clockwise to r: edges in subtour used $\leq k/2$ times
  $\Rightarrow \sum_{v \in C} c(v, r) \leq \frac{k}{2} OPT$



$$E[APX] \leq c(\pi) + \frac{2}{k} \sum_{v \in C} c(r, v) \leq \frac{3}{2} OPT + \frac{2}{k} \cdot \frac{k}{2} OPT = \frac{5}{2} OPT$$

Part 6
Set Cover

Source: *Approximation Algorithms* (Vazirani, Springer Press)

Set Cover

Problem: Set Cover

- Given: Elements $U := \{1, \ldots, n\}$, sets $S_1, \ldots, S_m \subseteq U$ with cost $c(S_i)$
- Find:

$$OPT = \min_{I \subseteq \{1,\ldots,m\}} \left\{ \sum_{i \in I} c(S_i) \mid \bigcup_{i \in I} S_i = U \right\}$$

Greedy algorithm:

(1) $I := \emptyset$
(2) WHILE not yet all elements covered DO
  (3) $price(S) := \frac{c(S)}{|S \setminus \bigcup_{i \in I} S_i|}$
  (4) $I := I \cup \{$ set S with minimum $price(S)\}$

Theorem

The greedy algorithm yields a $O(\log n)$-approximation.

Analysis

- Let $e_1, \ldots, e_n$ be elements in the order of covering.
- Suppose S ($S \in I$) newly covered $e_k, \ldots, e_\ell$

$$e_1, e_2, e_3, \ldots, \overbrace{e_k, \ldots, e_j, \ldots, e_\ell}^{\text{covered by } S}, \ldots, e_n \quad \overbrace{\qquad\qquad\qquad\qquad}^{n-k+1 \text{ elements}}$$

- Define $price(e_j) := price(S)$ for $j \in \{k, \ldots, \ell\}$.
- Consider the iteration, when S was chosen: Still $n - k + 1$ elements where uncovered and it was still possible to cover them all at cost OPT. Since S minimizes the price:

$$price(e_j) = price(e_k) \leq \frac{OPT}{n-k+1} \leq \frac{OPT}{n-j+1}$$

- Finally

$$APX = \sum_{j=1}^{n} price(e_j) \leq \sum_{j=1}^{n} \frac{OPT}{n-j+1} = OPT \cdot \sum_{j=1}^{n} \frac{1}{j} = O(\log n) \cdot OPT$$