Randomized Approximation Algorithms for TSP and Its Generalizations

Michael Khachay[1]

[1]Krasovsky Institute of Mathematics and Mechanics
Ural Federal University

Summer School on Operations Research and Applications
Nizhny Novgorod
May 11, 2017

Introduction

- Bad news. As numerous well known combinatorial optimization problems, the Traveling Salesman Problem (TSP) and its modifications are strongly NP-hard
- Therefore, efficient (polynomial time) optimal algorithms and even good approximation algorithms for these problems are hardly can be constructed ever

Introduction

- All facts above are concerned with so called worst case or min max principle
  - Algorithm is called efficient, if it finds an optimal (or good suboptimal solution) for any instance of the problem
- Good news. Promising results are obtained in a way of relaxation of this minmax principle
- Relaxation directions

  subclassing: considering special cases of the intractable problem, e.g. metric, Euclidean settings, etc. (Lecture 1 and 2)

  averaging: constructing algorithms efficient in average, e.g. simplex method for LP

  randomization: developing algorithms having high accuracy bounds and small time consumption with high probability

Introduction

- All facts above are concerned with so called worst case or min max principle
  - Algorithm is called efficient, if it finds an optimal (or good suboptimal solution) for any instance of the problem
- Good news. Promising results are obtained in a way of relaxation of this minmax principle
- Relaxation directions

subclassing: considering special cases of the intractable problem, e.g. metric, Euclidean settings, etc. (Lecture 1 and 2)

averaging: constructing algorithms efficient in average, e.g. simplex method for LP

randomization: developing algorithms having high accuracy bounds and small time consumption with high probability

Introduction

- All facts above are concerned with so called worst case or min max principle
  - Algorithm is called efficient, if it finds an optimal (or good suboptimal solution) for any instance of the problem
- Good news. Promising results are obtained in a way of relaxation of this minmax principle
- Relaxation directions

subclassing	considering special cases of the intractable problem, e.g. metric, Euclidean settings, etc. (Lecture 1 and 2)
averaging	constructing algorithms efficient in average, e.g. simplex method for LP
randomization	developing algorithms having high accuracy bounds and small time consumption with high probability

Algorithms with bounds

- Consider a subclass $\mathcal{I}_n$ of our problem $\mathcal{I}$ consisting of instances of length n
  - e.g., for TSP, $\mathcal{I}_n$ contains instances defined by graphs on n nodes ...
- On $\mathcal{I}_n$, define a probabilistic measure $\mathbf{Pr} = \mathbf{Pr}_n$
- Algorithm $\mathcal{A}$ has an accuracy bound $\varepsilon = \varepsilon(n)$ with a confidence $\delta = \delta(n)$ if

$$\mathbf{Pr}\left\{\left|\frac{\mathrm{APP}(I) - \mathrm{OPT}(I)}{\mathrm{OPT}(I)}\right| > \varepsilon(n)\right\} \leq \delta(n)$$

- Algorithm $\mathcal{A}$ is called asymptotically optimal [Gimadi, Perepelitsa (1974)] (or AO-algorithm), if

$$\lim_{n\to\infty} \varepsilon(n) = 0, \text{ and } \lim_{n\to\infty} \delta(n) = 0$$

- AO-algorithm $\mathcal{A}$, for which $\delta(n) = 0$ for any $n \geq n_0$, is called deterministic asymptotically optimal (DAO-algorithm)

Algorithms with bounds

- Consider a subclass $\mathcal{I}_n$ of our problem $\mathcal{I}$ consisting of instances of length n
  - e.g., for TSP, $\mathcal{I}_n$ contains instances defined by graphs on n nodes ...
- On $\mathcal{I}_n$, define a probabilistic measure $\mathbf{Pr} = \mathbf{Pr}_n$
- Algorithm $\mathcal{A}$ has an accuracy bound $\varepsilon = \varepsilon(n)$ with a confidence $\delta = \delta(n)$ if

$$\mathbf{Pr}\left\{\left|\frac{\mathrm{APP}(I) - \mathrm{OPT}(I)}{\mathrm{OPT}(I)}\right| > \varepsilon(n)\right\} \leq \delta(n)$$

- Algorithm $\mathcal{A}$ is called asymptotically optimal [Gimadi, Perepelitsa (1974)] (or AO-algorithm), if

$$\lim_{n\to\infty} \varepsilon(n) = 0, \text{ and } \lim_{n\to\infty} \delta(n) = 0$$

- AO-algorithm $\mathcal{A}$, for which $\delta(n) = 0$ for any $n \geq n_0$, is called deterministic asymptotically optimal (DAO-algorithm)

Algorithms with bounds

- Consider a subclass $\mathcal{I}_n$ of our problem $\mathcal{I}$ consisting of instances of length n
  - e.g., for TSP, $\mathcal{I}_n$ contains instances defined by graphs on n nodes ...
- On $\mathcal{I}_n$, define a probabilistic measure $\mathbf{Pr} = \mathbf{Pr}_n$
- Algorithm $\mathcal{A}$ has an accuracy bound $\varepsilon = \varepsilon(n)$ with a confidence $\delta = \delta(n)$ if

$$\mathbf{Pr}\left\{\left|\frac{\mathrm{APP}(I) - \mathrm{OPT}(I)}{\mathrm{OPT}(I)}\right| > \varepsilon(n)\right\} \leq \delta(n)$$

- Algorithm $\mathcal{A}$ is called asymptotically optimal [Gimadi, Perepelitsa (1974)] (or AO-algorithm), if

$$\lim_{n\to\infty} \varepsilon(n) = 0, \text{ and } \lim_{n\to\infty} \delta(n) = 0$$

- AO-algorithm $\mathcal{A}$, for which $\delta(n) = 0$ for any $n \geq n_0$, is called deterministic asymptotically optimal (DAO-algorithm)

Algorithms with bounds

- Consider a subclass $\mathcal{I}_n$ of our problem $\mathcal{I}$ consisting of instances of length n
  - e.g., for TSP, $\mathcal{I}_n$ contains instances defined by graphs on n nodes ...
- On $\mathcal{I}_n$, define a probabilistic measure $\mathbf{Pr} = \mathbf{Pr}_n$
- Algorithm $\mathcal{A}$ has an accuracy bound $\varepsilon = \varepsilon(n)$ with a confidence $\delta = \delta(n)$ if

$$\mathbf{Pr}\left\{\left|\frac{\mathrm{APP}(I) - \mathrm{OPT}(I)}{\mathrm{OPT}(I)}\right| > \varepsilon(n)\right\} \leq \delta(n)$$

- Algorithm $\mathcal{A}$ is called asymptotically optimal [Gimadi, Perepelitsa (1974)] (or AO-algorithm), if

$$\lim_{n\to\infty} \varepsilon(n) = 0, \quad \text{and} \quad \lim_{n\to\infty} \delta(n) = 0$$

- AO-algorithm $\mathcal{A}$, for which $\delta(n) = 0$ for any $n \geq n_0$, is called deterministic asymptotically optimal (DAO-algorithm)

Contents

Euclidean Max-TSP

Max-TSP

Input: a complete weighted graph $G = (V, E, w)$

Required: to find a Hamiltonian cycle H of maximal weight

- As above, Max-TSP is called the Euclidean, if $V \subset \mathbb{R}^d$ (for some $d > 1$) and $w(v_i, v_j) = \|v_i - v_j\|_2$.
- The Euclidean Max-TSP has a deterministic asymptotically optimal algorithm with time complexity $O(n^3)$.

Euclidean Max-TSP

Max-TSP

Input: a complete weighted graph $G = (V, E, w)$

Required: to find a Hamiltonian cycle H of maximal weight

- As above, Max-TSP is called the Euclidean, if $V \subset \mathbb{R}^d$ (for some $d > 1$) and $w(v_i, v_j) = \|v_i - v_j\|_2$.
- The Euclidean Max-TSP has a deterministic asymptotically optimal algorithm with time complexity $O(n^3)$.

Gimadi-Serdyukov algorithm :: preliminaries

- In complete weighted graph, a maximum weight perfect matching can be found (by J. Edmonds' 'blossom' algorithm) in time $O(n^3)$ (see, e.g. [Lovász, Plummer (1986)])
- For any fixed dimension $d > 1$, any sufficient large collection of line segments in $\mathbb{R}^d$ contains a couple of nearly parallel ones
- Butterfly gadget: for any pair of line segments $[A, B]$ and $[C, D]$

Gimadi-Serdyukov algorithm :: preliminaries

- In complete weighted graph, a maximum weight perfect matching can be found (by J. Edmonds' 'blossom' algorithm) in time $O(n^3)$ (see, e.g. [Lovász, Plummer (1986)])
- For any fixed dimension $d > 1$, any sufficient large collection of line segments in $\mathbb{R}^d$ contains a couple of nearly parallel ones
- Butterfly gadget: for any pair of line segments $[A, B]$ and $[C, D]$

Gimadi-Serdyukov algorithm :: preliminaries

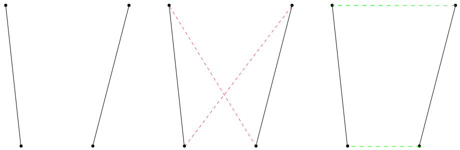
- In complete weighted graph, a maximum weight perfect matching can be found (by J. Edmonds' 'blossom' algorithm) in time $O(n^3)$ (see, e.g. [Lovász, Plummer (1986)])
- For any fixed dimension $d > 1$, any sufficient large collection of line segments in $\mathbb{R}^d$ contains a couple of nearly parallel ones
- Butterfly gadget: for any pair of line segments $[A, B]$ and $[C, D]$ in the Euclidean space,

$$\max\{|A, C| + |B, D|, |A, D| + |B, C|\} \geq \max\{|A, B|, |C, D|, (|A, B| + |C, D|) \cos \frac{\alpha}{2}\}$$

where $0 \leq \alpha < \pi/2$ is an angle between the segments

Gimadi-Serdyukov algorithm :: preliminaries

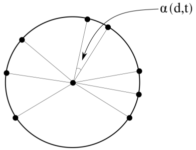
- In complete weighted graph, a maximum weight perfect matching can be found (by J. Edmonds' 'blossom' algorithm) in time $O(n^3)$ (see, e.g. [Lovász, Plummer (1986)])
- For any fixed dimension $d > 1$, any sufficient large collection of line segments in $\mathbb{R}^d$ contains a couple of nearly parallel ones
- Butterfly gadget: for any pair of line segments $[A, B]$ and $[C, D]$



Gimadi-Serdyukov algorithm :: angular packing on the Euclidean sphere

- The fact '*for any fixed dimension $d > 1$, any sufficient large collection of line segments in $\mathbb{R}^d$ contains a couple of nearly parallel ones*' follows from compactness of the unit Euclidean sphere S_{d-1} in d-dimensional space wrt *angular* distance

$$x, y \in S_{d-1},\ \operatorname{dist}(x, y) = \arccos(x, y)$$



Gimadi-Serdyukov algorithm :: angular packing on the Euclidean sphere

- The fact '*for any fixed dimension* $d > 1$, *any sufficient large collection of line segments in* $\mathbb{R}^d$ *contains a couple of nearly parallel ones*' follows from compactness of the unit Euclidean sphere S_{d-1} in d-dimensional space wrt *angular* distance

$$x, y \in S_{d-1},\ \operatorname{dist}(x, y) = \arccos(x, y)$$

Lemma [Serdyukov, (1984)]

Let E be a set of t linear segments in $\mathbb{R}^d$ for some $d > 1$. Then, the minimum inter-segment angle $\alpha(d, t)$ satisfies the equation

$$\sin^2 \frac{\alpha(d,t)}{2} \leq \frac{\gamma_d}{t^{2/(d-1)}}$$

Gimadi-Serdyukov algorithm :: scheme

1. Find a maximum weight perfect matching $M^* = \{e_1, \ldots, e_\mu\}$, where $\mu = \lfloor n/2 \rfloor$ and $w(e_1) \geq w(e_2) \geq \ldots \geq w(e_\mu)$
2. For some number $2 \leq t \leq n/4$ (will be specified later) take subsets $M_1^* = \{e_1, \ldots, e_{\mu-t+2}\}$ and $M_2^* = \{e_{\mu-t+3}, \ldots, e_\mu\}$ such that $M_1^* \cup M_2^* = M^*$ and $|M_2^*| = t - 2$. We call elements of M_1^* and M_2^* heavy and light, respectively
3. Applying Serdyukov's lemma recurrently, construct sequences $S_1, \ldots, S_{t-1}$ of heavy edges such that, for any sequence $S_i = (e_{i_1}, \ldots, e_{i_k})$,

$$\widehat{e_{i_j}, e_{i_{j+1}}} \leq \alpha(d, t), \ (1 \leq j < k)$$

Gimadi-Serdyukov algorithm :: scheme

1. Find a maximum weight perfect matching $M^* = \{e_1, \dots, e_\mu\}$, where $\mu = \lfloor n/2 \rfloor$ and $w(e_1) \geq w(e_2) \geq \dots \geq w(e_\mu)$
2. For some number $2 \leq t \leq n/4$ (will be specified later) take subsets $M_1^* = \{e_1, \dots, e_{\mu-t+2}\}$ and $M_2^* = \{e_{\mu-t+3}, \dots, e_\mu\}$ such that $M_1^* \cup M_2^* = M^*$ and $|M_2^*| = t - 2$. We call elements of M_1^* and M_2^* heavy and light, respectively
3. Applying Serdyukov's lemma recurrently, construct sequences $S_1, \dots, S_{t-1}$ of heavy edges such that, for any sequence $S_i = (e_{i_1}, \dots, e_{i_k})$,

$$\widehat{e_{i_j}, e_{i_{j+1}}} \leq \alpha(d, t), \ (1 \leq j < k)$$

Gimadi-Serdyukov algorithm :: scheme

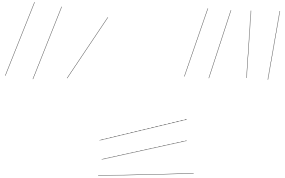
1. Find a maximum weight perfect matching $M^* = \{e_1, \ldots, e_\mu\}$, where $\mu = \lfloor n/2 \rfloor$ and $w(e_1) \geq w(e_2) \geq \ldots \geq w(e_\mu)$
2. For some number $2 \leq t \leq n/4$ (will be specified later) take subsets $M_1^* = \{e_1, \ldots, e_{\mu-t+2}\}$ and $M_2^* = \{e_{\mu-t+3}, \ldots, e_\mu\}$ such that $M_1^* \cup M_2^* = M^*$ and $|M_2^*| = t - 2$. We call elements of M_1^* and M_2^* heavy and light, respectively
3. Applying Serdyukov's lemma recurrently, construct sequences $S_1, \ldots, S_{t-1}$ of heavy edges such that, for any sequence $S_i = (e_{i_1}, \ldots, e_{i_k})$,

$$\widehat{e_{i_j}, e_{i_{j+1}}} \leq \alpha(d, t), \ (1 \leq j < k)$$

Gimadi-Serdyukov algorithm :: scheme

1. Find a maximum weight perfect matching $M^* = \{e_1, \ldots, e_\mu\}$, where $\mu = \lfloor n/2 \rfloor$ and $w(e_1) \geq w(e_2) \geq \ldots \geq w(e_\mu)$
2. For some number $2 \leq t \leq n/4$ (will be specified later) take subsets $M_1^* = \{e_1, \ldots, e_{\mu-t+2}\}$ and $M_2^* = \{e_{\mu-t+3}, \ldots, e_\mu\}$ such that $M_1^* \cup M_2^* = M^*$ and $|M_2^*| = t - 2$. We call elements of M_1^* and M_2^* heavy and light, respectively
3. Applying Serdyukov's lemma recurrently, construct sequences $S_1, \ldots, S_{t-1}$ of heavy edges such that, for any sequence $S_i = (e_{i_1}, \ldots, e_{i_k})$,

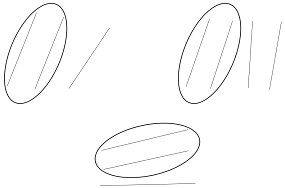
$$\widehat{e_{i_j}, e_{i_{j+1}}} \leq \alpha(d, t), \ (1 \leq j < k)$$



Gimadi-Serdyukov algorithm :: scheme

1. Find a maximum weight perfect matching $M^* = \{e_1, \dots, e_\mu\}$, where $\mu = \lfloor n/2 \rfloor$ and $w(e_1) \geq w(e_2) \geq \dots \geq w(e_\mu)$
2. For some number $2 \leq t \leq n/4$ (will be specified later) take subsets $M_1^* = \{e_1, \dots, e_{\mu-t+2}\}$ and $M_2^* = \{e_{\mu-t+3}, \dots, e_\mu\}$ such that $M_1^* \cup M_2^* = M^*$ and $|M_2^*| = t - 2$. We call elements of M_1^* and M_2^* heavy and light, respectively
3. Applying Serdyukov's lemma recurrently, construct sequences $S_1, \dots, S_{t-1}$ of heavy edges such that, for any sequence $S_i = (e_{i_1}, \dots, e_{i_k})$,

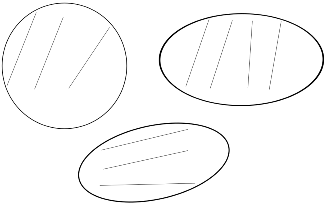
$$\widehat{e_{i_j}, e_{i_{j+1}}} \leq \alpha(d,t), \; (1 \leq j < k)$$



Gimadi-Serdyukov algorithm :: scheme

1. Find a maximum weight perfect matching $M^* = \{e_1, \ldots, e_\mu\}$, where $\mu = \lfloor n/2 \rfloor$ and $w(e_1) \geq w(e_2) \geq \ldots \geq w(e_\mu)$
2. For some number $2 \leq t \leq n/4$ (will be specified later) take subsets $M_1^* = \{e_1, \ldots, e_{\mu-t+2}\}$ and $M_2^* = \{e_{\mu-t+3}, \ldots, e_\mu\}$ such that $M_1^* \cup M_2^* = M^*$ and $|M_2^*| = t - 2$. We call elements of M_1^* and M_2^* heavy and light, respectively
3. Applying Serdyukov's lemma recurrently, construct sequences $S_1, \ldots, S_{t-1}$ of heavy edges such that, for any sequence $S_i = (e_{i_1}, \ldots, e_{i_k})$,

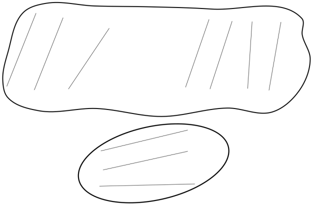
$$\widehat{e_{i_j}, e_{i_{j+1}}} \leq \alpha(d,t), \quad (1 \leq j < k)$$



Gimadi-Serdyukov algorithm :: scheme

1. Find a maximum weight perfect matching $M^* = \{e_1, \ldots, e_\mu\}$, where $\mu = \lfloor n/2 \rfloor$ and $w(e_1) \geq w(e_2) \geq \ldots \geq w(e_\mu)$
2. For some number $2 \leq t \leq n/4$ (will be specified later) take subsets $M_1^* = \{e_1, \ldots, e_{\mu-t+2}\}$ and $M_2^* = \{e_{\mu-t+3}, \ldots, e_\mu\}$ such that $M_1^* \cup M_2^* = M^*$ and $|M_2^*| = t - 2$. We call elements of M_1^* and M_2^* heavy and light, respectively
3. Applying Serdyukov's lemma recurrently, construct sequences $S_1, \ldots, S_{t-1}$ of heavy edges such that, for any sequence $S_i = (e_{i_1}, \ldots, e_{i_k})$,

$$\widehat{e_{i_j}, e_{i_{j+1}}} \leq \alpha(d,t), \ (1 \leq j < k)$$

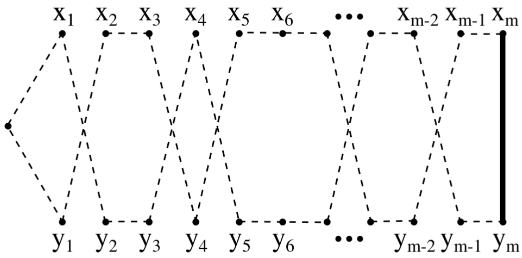


Gimadi-Serdyukov algorithm :: scheme cont.

4. Consider the edges of M^* in the following order:
   $S_1, e_{l_1}, S_2, \ldots, e_{l_{t-2}}, S_{t-1}$
5. Replacing any pair of consecutive edges according to the butterfly gadget obtain Hamiltonian cycle $H = H_t$

Gimadi-Serdyukov algorithm :: scheme cont.

4. Consider the edges of M^* in the following order: $S_1, e_{l_1}, S_2, \ldots, e_{l_{t-2}}, S_{t-1}$
5. Replacing any pair of consecutive edges according to the butterfly gadget obtain Hamiltonian cycle $H = H_t$



Gimadi-Serdyukov algorithm :: scheme cont.

4. Consider the edges of M^* in the following order: $S_1, e_{l_1}, S_2, \ldots, e_{l_{t-2}}, S_{t-1}$
5. Replacing any pair of consecutive edges according to the butterfly gadget obtain Hamiltonian cycle $H = H_t$

- Stage 1 of the algorithm is the most expensive
- Therefore, the overall time consumption of GS-algorithm is $O(n^3)$

Gimadi-Serdyukov algorithm :: accuracy bound

Technical Lemma 1

Weights $w(H_t)$ and $w(M^*)$ satisfy the following equation

$$w(H_t) \geq 2w(M^*) \left(1 - \frac{t-2}{\mu}\right) \cos \frac{\alpha(d,t)}{2}$$

Technical Lemma 2

Let H^* be a maximum weight Hamiltonian cycle (an optimal solution). Then

$$\frac{w(M^*)}{w(H^*)} \geq \frac{\mu}{n}$$

Gimadi-Serdyukov algorithm :: accuracy bound

Technical Lemma 1

Weights $w(H_t)$ and $w(M^*)$ satisfy the following equation

$$w(H_t) \geq 2w(M^*)\left(1 - \frac{t-2}{\mu}\right)\cos\frac{\alpha(d,t)}{2}$$

Technical Lemma 2

Let H^* be a maximum weight Hamiltonian cycle (an optimal solution). Then

$$\frac{w(M^*)}{w(H^*)} \geq \frac{\mu}{n}$$

Gimadi-Serdyukov algorithm :: accuracy bound

Technical Lemma 1

Weights $w(H_t)$ and $w(M^*)$ satisfy the following equation

$$w(H_t) \geq 2w(M^*)\left(1 - \frac{t-2}{\mu}\right)\cos\frac{\alpha(d,t)}{2}$$

Technical Lemma 2

Let H^* be a maximum weight Hamiltonian cycle (an optimal solution). Then

$$\frac{w(M^*)}{w(H^*)} \geq \frac{\mu}{n}$$

For $n = 2\mu$, TL2 is evidently follows from $2w(M^*) \geq w(H^*)$

For $n = 2\mu + 1$, we obtain $2w(M^*) \geq (1 - 1/n)w(H^*)$

Gimadi-Serdyukov algorithm :: accuracy bound

Main Lemma

$$\frac{w(H_t)}{w(H^*)} \geq 1 - 2\frac{t-1}{n} - \frac{\gamma_d}{t^{2/(d-1)}}$$

Theorem

For $t = \max\{\lceil n^{(d-1)/(d+1)}/4\rceil, 2\}$, we have

$$\frac{w(H^*) - w(H_t)}{H^*} \leq \frac{\beta_d}{n^{2/(d+1)}}$$

i.e. GS-algorithm is deterministic asymptotically optimal

Gimadi-Serdyukov algorithm :: accuracy bound

Main Lemma

$$\frac{w(H_t)}{w(H^*)} \geq 1 - 2\frac{t-1}{n} - \frac{\gamma_d}{t^{2/(d-1)}}$$

Theorem

For $t = \max\{\lceil n^{(d-1)/(d+1)}/4 \rceil, 2\}$, we have

$$\frac{w(H^*) - w(H_t)}{H^*} \leq \frac{\beta_d}{n^{2/(d+1)}}$$

i.e. GS-algorithm is deterministic asymptotically optimal

Contents

Maximum weight k-Size Cycle Cover Problem

Max-k-SCCP

Input: graph $G = (V, E, w)$.

Find: a maximum-weight collection $\mathcal{C} = C_1, ..., C_k$ of vertex-disjoint cycles such that $\bigcup_{i \in \mathbb{N}_k} V(C_i) = V$.

Maximum weight k-Size Cycle Cover Problem

Max-k-SCCP

Input: graph $G = (V, E, w)$.

Find: a maximum-weight collection $\mathcal{C} = C_1, ..., C_k$ of vertex-disjoint cycles such that $\bigcup_{i \in \mathbb{N}_k} V(C_i) = V$.

$$\max \quad \sum_{i=1}^{k} w(C_i) \equiv \sum_{i=1}^{k} \sum_{e \in E(C_i)} w(e)$$

$$s.t.$$

$$C_1, \ldots, C_k \text{ are cycles in } G$$

$$C_i \cap C_j = \varnothing$$

$$V(C_1) \cup \ldots \cup V(C_k) = V$$

Gimadi-Rykov algorithm :: main idea

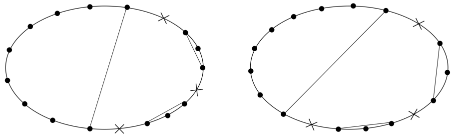
- Gimadi-Serdyukov asymptotically optimal algorithm for the Euclidean Max-TSP
- Haimovich and Rinnoy Kan Iterative Tour Partition (ITP) heuristic
- Cycle joining heuristic based on the butterfly gadget

Gimadi-Rykov algorithm :: main idea

- Gimadi-Serdyukov asymptotically optimal algorithm for the Euclidean Max-TSP
- Haimovich and Rinnoy Kan Iterative Tour Partition (ITP) heuristic
- Cycle joining heuristic based on the butterfly gadget

Gimadi-Rykov algorithm :: scheme

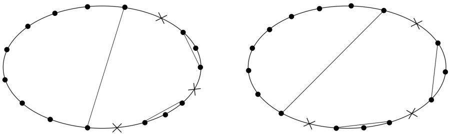
1. Using GS-algorithm, find an approximate solution $\tilde{H}$ of the auxiliary Euclidean Max-TSP
2. Take an arbitrarily integer partition $l_1 + \ldots + l_k = n$ s.t. $l_j > 2$
3. Applying ITP, build n candidate cycle covers $\mathcal{C}_1, \ldots, \mathcal{C}_n$



4. Output $\tilde{\mathcal{C}} = \arg\max\{\mathcal{C}_j : j = 1, n\}$

Gimadi-Rykov algorithm :: scheme

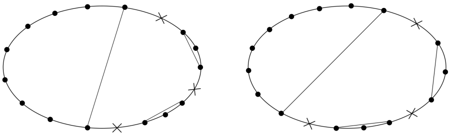
1. Using GS-algorithm, find an approximate solution $\tilde{H}$ of the auxiliary Euclidean Max-TSP
2. Take an arbitrarily integer partition $l_1 + \ldots + l_k = n$ s.t. $l_j > 2$
3. Applying ITP, build n candidate cycle covers $\mathcal{C}_1, \ldots, \mathcal{C}_n$



4. Output $\tilde{\mathcal{C}} = \arg\max\{\mathcal{C}_j : j = \overline{1, n}\}$

Gimadi-Rykov algorithm :: scheme

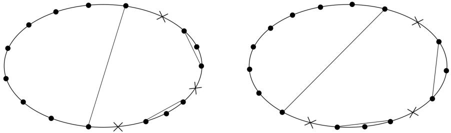
1. Using GS-algorithm, find an approximate solution $\tilde{H}$ of the auxiliary Euclidean Max-TSP
2. Take an arbitrarily integer partition $l_1 + \ldots + l_k = n$ s.t. $l_j > 2$
3. Applying ITP, build n candidate cycle covers $\mathcal{C}_1, \ldots, \mathcal{C}_n$



4. Output $\tilde{\mathcal{C}} = \arg\max\{\mathcal{C}_j : j = 1, n\}$

Gimadi-Rykov algorithm :: scheme

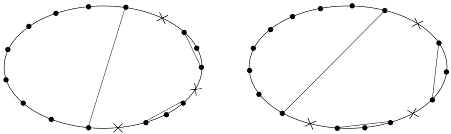
1. Using GS-algorithm, find an approximate solution $\tilde{H}$ of the auxiliary Euclidean Max-TSP
2. Take an arbitrarily integer partition $l_1 + \ldots + l_k = n$ s.t. $l_j > 2$
3. Applying ITP, build n candidate cycle covers $\mathcal{C}_1, \ldots, \mathcal{C}_n$



4. Output $\tilde{\mathcal{C}} = \arg\max\{\mathcal{C}_j : j = 1, n\}$

Gimadi-Rykov algorithm :: scheme

1. Using GS-algorithm, find an approximate solution $\tilde{H}$ of the auxiliary Euclidean Max-TSP
2. Take an arbitrarily integer partition $l_1 + \ldots + l_k = n$ s.t. $l_j > 2$
3. Applying ITP, build n candidate cycle covers $\mathcal{C}_1, \ldots, \mathcal{C}_n$



4. Output $\tilde{\mathcal{C}} = \arg\max\{\mathcal{C}_j : j = 1, n\}$

Time complexity of this algorithm is $O(n^3)$

Gimadi-Rykov algorithm :: accuracy bound

ITP Lemma

$$w(\tilde{\mathcal{C}}) \geq \left(1 - \frac{k}{n}\right) w(\tilde{H})$$

- Indeed, by construction, any edge of $\tilde{H}$ belongs to $\mathcal{C}_j$ $n-k$ times (and k times is rejected)
- Therefore, $\sum_{j=1}^{n} w(\mathcal{C}_j) \geq (n-k)w(\tilde{H})$
- Finally, $w(\tilde{\mathcal{C}}) \geq 1/n \sum_{j=1}^{n} w(\mathcal{C}_j) \geq (1-k/n)w(\tilde{H})$

Gimadi-Rykov algorithm :: accuracy bound

ITP Lemma

$$w(\tilde{\mathcal{C}}) \geq \left(1 - \frac{k}{n}\right) w(\tilde{H})$$

- Indeed, by construction, any edge of $\tilde{H}$ belongs to $\mathcal{C}_j$ $n-k$ times (and k times is rejected)
- Therefore, $\sum_{j=1}^{n} w(\mathcal{C}_j) \geq (n-k)w(\tilde{H})$
- Finally, $w(\tilde{\mathcal{C}}) \geq 1/n \sum_{j=1}^{n} w(\mathcal{C}_j) \geq (1-k/n)w(\tilde{H})$

Gimadi-Rykov algorithm :: accuracy bound

ITP Lemma

$$w(\tilde{\mathcal{C}}) \geq \left(1 - \frac{k}{n}\right) w(\tilde{H})$$

- Indeed, by construction, any edge of $\tilde{H}$ belongs to $\mathcal{C}_j$ $n-k$ times (and k times is rejected)
- Therefore, $\sum_{j=1}^{n} w(\mathcal{C}_j) \geq (n-k)w(\tilde{H})$
- Finally, $w(\tilde{\mathcal{C}}) \geq 1/n \sum_{j=1}^{n} w(\mathcal{C}_j) \geq (1-k/n)w(\tilde{H})$

Combining with the previous results, obtain

Technical Lemma 3

$$w(\tilde{\mathcal{C}}) \geq 2w(M^*) \left(1 - \frac{k}{n}\right) \left(1 - \frac{t-1}{\mu}\right) \left(1 - \frac{\gamma_d}{t^{2/(d+1)}}\right)$$

Gimadi-Rykov algorithm :: accuracy bound

- On the other hand, an optimal cycle cover $\mathcal{C}^*$ can be restructured to a Hamiltonian cycle H (by cycle joining and butterfly gadgets)
- It is easy to check that $w(H) \geq (1 - k/n)w(\mathcal{C}^*)$
- Then,

$$2w(M^*) \geq \left(1 - \frac{1}{n}\right)\left(1 - \frac{k}{n}\right)w(\mathcal{C}^*)$$

- since $2w(M^*) \geq (1 - 1/n)w(H)$

Gimadi-Rykov algorithm :: accuracy bound

- On the other hand, an optimal cycle cover $\mathcal{C}^*$ can be restructured to a Hamiltonian cycle H (by cycle joining and butterfly gadgets)
- It is easy to check that $w(H) \geq (1 - k/n)w(\mathcal{C}^*)$
- Then,

$$2w(M^*) \geq \left(1 - \frac{1}{n}\right)\left(1 - \frac{k}{n}\right)w(\mathcal{C}^*)$$

- since $2w(M^*) \geq (1 - 1/n)w(H)$

Gimadi-Rykov algorithm :: accuracy bound

As for GS-algorithm

Main Lemma

$$\frac{w(\tilde{\mathcal{C}})}{w(\mathcal{C}^*)} \geq 1 - 2\frac{k+t-1}{n} - \frac{\gamma_d}{t^{2/(d-1)}}$$

Theorem

For $t = \max\{\lceil n^{(d-1)/(d+1)}/4\rceil, 2\}$ and $k = o(n)$ GR-algorithm is deterministic asymptotically optimal

Gimadi-Rykov algorithm :: accuracy bound

As for GS-algorithm

Main Lemma

$$\frac{w(\tilde{\mathcal{C}})}{w(\mathcal{C}^*)} \geq 1 - 2\frac{k+t-1}{n} - \frac{\gamma_d}{t^{2/(d-1)}}$$

Theorem

For $t = \max\{\lceil n^{(d-1)/(d+1)}/4\rceil, 2\}$ and $k = o(n)$ GR-algorithm is deterministic asymptotically optimal

Contents

Nearest Neighbor Heuristic

Many simple algorithms having poor worst case accuracy (in minmax setting) perform good on random inputs

NN for Min-TSP

1. start with a partial tour consisting of a single, arbitrarily taken node v_1
2. If the current partial tour is $a_1, \ldots, a_k$ and $k < n$, take $a_k + 1$ from nodes not in the tour, which is closest to a_k and construct a new tour $a_1, \ldots, a_k, a_{k+1}$
3. stop when the current tour contains all n nodes

Nearest Neighbor Heuristic

Many simple algorithms having poor worst case accuracy (in minmax setting) perform good on random inputs

NN for Min-TSP

1. start with a partial tour consisting of a single, arbitrarily taken node v_1
2. If the current partial tour is $a_1, \ldots, a_k$ and $k < n$, take $a_k + 1$ from nodes not in the tour, which is closest to a_k and construct a new tour $a_1, \ldots, a_k, a_{k+1}$
3. stop when the current tour contains all n nodes

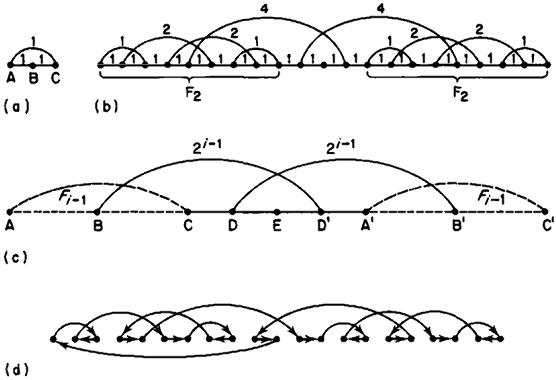
Time complexity of NN is $O(n^2)$

NN :: worst case accuracy bound

Theorem [Rozencrantz et al. (1977)]

For every $r > 1$ there exist n-node instance I of Metric Min-TSP such that

$$\mathrm{APP}(I) > r \cdot \mathrm{OPT}(I)$$

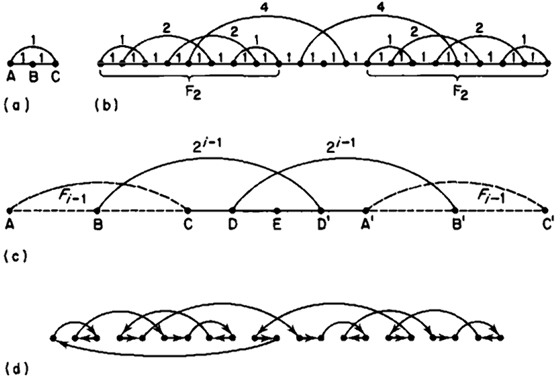


NN :: worst case accuracy bound

Theorem [Rozencrantz et al. (1977)]

For every $r > 1$ there exist n-node instance I of Metric Min-TSP such that

$$\mathrm{APP}(I) > r \cdot \mathrm{OPT}(I)$$



Actually, it is proved that NN is $O(\log n)$-approximation algorithm

NN :: accuracy on random inputs

- Consider Random Min-TSP, weights $w_{i,j}$ are i.i.d. in $[a_n, b_n]$

Theorem [Gimadi (2001)]

For NN algorithm, equation

$$\mathbf{Pr}\left\{\left|\frac{\mathrm{APP}(I)-\mathrm{OPT}(I)}{\mathrm{OPT}(I)}\right| > \varepsilon(n)\right\} \leq \delta(n)$$

is valid for

$$\varepsilon(n) = 2\frac{(b_n - a_n)/a_n}{n/\ln n}, \quad \delta(n) = O(n^{-1})$$

Moreover, the algorithm is asymptotically optimal when

$$\frac{b_n - a_n}{a_n} = o\left(\frac{n}{\ln n}\right)$$

NN :: accuracy on random inputs

- Theorem follows from the well known Petrov's measure concentration theorem
- The result is extended to the case of Gaussian and exponential distributions and any other distribution majorizing them

Conclusion

- Many intractable problems can be solved efficiently in special cases or on random inputs
- It is curious, but sometimes the 'curse of dimensionality' principle fails (asymptotic optimal algorithms)
- some poor in worst case algorithms (like Nearest Neighbor) are quit good on random inputs

Thank you for your attention!