Approximation Algorithms and Schemes for Traveling Salesman, Vechicle Routing, and Related Problems

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

- A vast majority of combinatorial optimization problems (e.g., Set Cover, Hitting Set Problem, Maximal Clique Problem, etc.) are known to be intractable and hardly approximable in general settings
- Meanwhile, for many actual special cases of these problems there are known efficient exact or approximation algorithms
- For instance, many combinatorial optimization problems become much more approximable being formulated in geometrical setting
- In this tutorial we consider three geometric problems generalizing the well known Traveling Salesman Problem (TSP)

Introduction

- A vast majority of combinatorial optimization problems (e.g., Set Cover, Hitting Set Problem, Maximal Clique Problem, etc.) are known to be intractable and hardly approximable in general settings
- Meanwhile, for many actual special cases of these problems there are known efficient exact or approximation algorithms
- For instance, many combinatorial optimization problems become much more approximable being formulated in geometrical setting
- In this tutorial we consider three geometric problems generalizing the well known Traveling Salesman Problem (TSP)

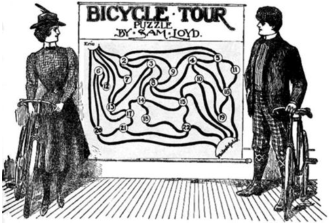
Introduction

- A vast majority of combinatorial optimization problems (e.g., Set Cover, Hitting Set Problem, Maximal Clique Problem, etc.) are known to be intractable and hardly approximable in general settings
- Meanwhile, for many actual special cases of these problems there are known efficient exact or approximation algorithms
- For instance, many combinatorial optimization problems become much more approximable being formulated in geometrical setting
- In this tutorial we consider three geometric problems generalizing the well known Traveling Salesman Problem (TSP)

Traveling Salesman Problem(TSP)

Problem statement

Input: complete weighted graph $G = (V, E, w)$
Required: to find a Hamiltonian cycle of the minimum (or maximum) weight

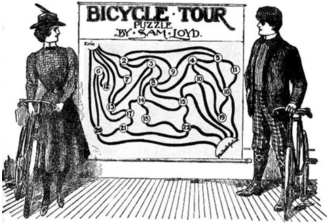


- For the first time TSP is mentioned in books about mathematical puzzles (S.Loyd, 1914)
- The first mathematical statement was introduced by Karl Menger (1930) and later in (G.Danzig and J.Ramser, 1959)

Traveling Salesman Problem(TSP)

Problem statement

Input: complete weighted graph $G = (V, E, w)$
Required: to find a Hamiltonian cycle of the minimum (or maximum) weight



- For the first time TSP is mentioned in books about mathematical puzzles (S.Loyd, 1914)
- The first mathematical statement was introduced by Karl Menger (1930) and later in (G.Danzig and J.Ramser, 1959)

Complexity bounds

- exhaustive search $\Theta(n!)$
- dynamic programming $\Theta(n^2 2^n)$

Curse of dimensionality

- **Benchmark:** use dynamic programming to solve TSP to optimality
- Suppose, our supercomputer can solve TSP for $n = 100$ in 1 sec
- Easy to see
  - for $n = 125$, we need more than year
  - for $n = 136$ — more then 2016 years!

Curse of dimensionality

- **Benchmark:** use dynamic programming to solve TSP to optimality
- Suppose, our supercomputer can solve TSP for $n = 100$ in 1 sec
- Easy to see
  - for $n = 125$, we need more than year
  - for $n = 136$ — more then 2016 years!

Curse of dimensionality

- **Benchmark:** use dynamic programming to solve TSP to optimality
- Suppose, our supercomputer can solve TSP for $n = 100$ in 1 sec
- Easy to see
  - for $n = 125$, we need more than year
  - for $n = 136$ — more then 2016 years!

Curse of dimensionality

- **Benchmark:** use dynamic programming to solve TSP to optimality
- Suppose, our supercomputer can solve TSP for $n = 100$ in 1 sec
- Easy to see
  - for $n = 125$, we need more than year
  - for $n = 136$ — more then 2016 years!

Combinatorial optimization problems

- Combinatorial optimization problem $\mathcal{I}$

$$I:\ OPT_I = \min\{COST_I(x)\colon x \in X_I\}$$

  $n := LEN(I)$ is instance length $I \in \mathcal{I}$.
- Algorithm is an arbitrary function $Alg : I \mapsto Alg(I) \in X_I$, computable in time $TIME_{Alg}(I)$
- Algorithm Alg is called polynomial time, if

$$TIME_{Alg}(I) = O(poly(LEN(I))\ (I \in \mathcal{I})$$

  In this lecture we consider polynomial time algorithms only
- Algorithm Alg is called optimal, if

$$APP_I := COST_I(Alg(I)) = OPT_I$$

- For the major pert of CO problems, optimal polynomial time algorithms are not investigated so far and are hardly be developed ever unles $P = NP$

Combinatorial optimization problems

- Combinatorial optimization problem $\mathcal{I}$

$$I: \ OPT_I = \min\{COST_I(x) \colon x \in X_I\}$$

$n := LEN(I)$ is instance length $I \in \mathcal{I}$.

- Algorithm is an arbitrary function $Alg : I \mapsto Alg(I) \in X_I$, computable in time $TIME_{Alg}(I)$
- Algorithm Alg is called polynomial time, if

$$TIME_{Alg}(I) = O(poly(LEN(I)) \ (I \in \mathcal{I})$$

In this lecture we consider polynomial time algorithms only

- Algorithm Alg is called optimal, if

$$APP_I := COST_I(Alg(I)) = OPT_I$$

- For the major pert of CO problems, optimal polynomial time algorithms are not investigated so far and are hardly be developed ever unles $P = NP$

Combinatorial optimization problems

- Combinatorial optimization problem $\mathcal{I}$

$$I: \ OPT_I = \min\{COST_I(x) \colon x \in X_I\}$$

  $n := LEN(I)$ is instance length $I \in \mathcal{I}$.
- Algorithm is an arbitrary function $Alg : I \mapsto Alg(I) \in X_I$, computable in time $TIME_{Alg}(I)$
- Algorithm Alg is called polynomial time, if

$$TIME_{Alg}(I) = O(poly(LEN(I)) \ (I \in \mathcal{I})$$

  In this lecture we consider polynomial time algorithms only
- Algorithm Alg is called optimal, if

$$APP_I := COST_I(Alg(I)) = OPT_I$$

- For the major pert of CO problems, optimal polynomial time algorithms are not investigated so far and are hardly be developed ever unles $P = NP$

Approximation algorithms and Schemes

- Let, for some $r = r(I)$ the equation
$$OPT_I \leq APP_I = COST_I(Alg(I)) \leq r \cdot OPT_I \quad (i \in \mathcal{I})$$
is valid.
Then Alg is called r-approximation algorithm for the problem $\mathcal{I}$
- Algorithms with fixed accuracy bounds $r(I) = \text{const}$ and approximation schemes (PTAS) attract most interest
- The problem $\mathcal{I}$ has PTAS, if for any $\varepsilon > 0$ there exists $(1+\varepsilon)$-approximation algorithm Alg_ε
- Time complexity bound $TIME_{Alg_\varepsilon}(I) = O(poly(LEN(I))$ depends on n polynomially but can have an arbitrarily dependence on ε
For instance, $TIME_{Alg_\varepsilon}(I) = LEN(I)^{\exp(1/\varepsilon^3)}$
- Approximation scheme is called efficient (EPTAS), if
$$TIME_{Alg_\varepsilon}(I) = f(1/\varepsilon) \cdot poly(LEN(I))$$
- and fully polynomial, if
$$TIME_{Alg_\varepsilon}(I) = poly(LEN(I), 1/\varepsilon)$$

Approximation algorithms and Schemes

- Let, for some $r = r(I)$ the equation

$$OPT_I \le APP_I = COST_I(Alg(I)) \le r \cdot OPT_I \quad (i \in \mathcal{I})$$

  is valid.
  Then Alg is called r-approximation algorithm for the problem $\mathcal{I}$
- Algorithms with fixed accuracy bounds $r(I) = \text{const}$ and approximation schemes (PTAS) attract most interest
- The problem $\mathcal{I}$ has PTAS, if for any $\varepsilon > 0$ there exists $(1+\varepsilon)$-approximation algorithm Alg_ε
- Time complexity bound $TIME_{Alg_\varepsilon}(I) = O(poly(LEN(I)))$ depends on n polynomially but can have an arbitrarily dependence on ε
  For instance, $TIME_{Alg_\varepsilon}(I) = LEN(I)^{\exp(1/\varepsilon^3)}$
- Approximation scheme is called efficient (EPTAS), if

$$TIME_{Alg_\varepsilon}(I) = f(1/\varepsilon) \cdot poly(LEN(I))$$

- and fully polynomial, if

$$TIME_{Alg_\varepsilon}(I) = poly(LEN(I), 1/\varepsilon)$$

Approximation algorithms and Schemes

- Let, for some $r = r(I)$ the equation
$$OPT_I \leq APP_I = COST_I(Alg(I)) \leq r \cdot OPT_I \quad (i \in \mathcal{I})$$
is valid.
Then Alg is called r-approximation algorithm for the problem $\mathcal{I}$
- Algorithms with fixed accuracy bounds $r(I) =$ const and approximation schemes (PTAS) attract most interest
- The problem $\mathcal{I}$ has PTAS, if for any $\varepsilon > 0$ there exists $(1+\varepsilon)$-approximation algorithm Alg_ε
- Time complexity bound $TIME_{Alg_\varepsilon}(I) = O(poly(LEN(I))$ depends on n polynomially but can have an arbitrarily dependence on ε
For instance, $TIME_{Alg_\varepsilon}(I) = LEN(I)^{\exp(1/\varepsilon^3)}$
- Approximation scheme is called efficient (EPTAS), if
$$TIME_{Alg_\varepsilon}(I) = f(1/\varepsilon) \cdot poly(LEN(I))$$
- and fully polynomial, if
$$TIME_{Alg_\varepsilon}(I) = poly(LEN(I), 1/\varepsilon)$$

Approximation algorithms and Schemes

- Let, for some $r = r(I)$ the equation
$$OPT_I \leq APP_I = COST_I(Alg(I)) \leq r \cdot OPT_I \quad (i \in \mathcal{I})$$
is valid.
Then Alg is called r-approximation algorithm for the problem $\mathcal{I}$
- Algorithms with fixed accuracy bounds $r(I) =$ const and approximation schemes (PTAS) attract most interest
- The problem $\mathcal{I}$ has PTAS, if for any $\varepsilon > 0$ there exists $(1+\varepsilon)$-approximation algorithm Alg_ε
- Time complexity bound $TIME_{Alg_\varepsilon}(I) = O(poly(LEN(I))$ depends on n polynomially but can have an arbitrarily dependence on ε
For instance, $TIME_{Alg_\varepsilon}(I) = LEN(I)^{\exp(1/\varepsilon^3)}$
- Approximation scheme is called efficient (EPTAS), if
$$TIME_{Alg_\varepsilon}(I) = f(1/\varepsilon) \cdot poly(LEN(I))$$
- and fully polynomial, if
$$TIME_{Alg_\varepsilon}(I) = poly(LEN(I), 1/\varepsilon)$$

TSP :: known complexity and approximation results

Complexity

- (Karp, 1972) TSP is strongly NP-hard
- (Sahni and Gonzales, 1976) TSP can not be approximated within $O(2^n)$ (unless $P = NP$)
- (Papadimitriou, 1977) Euclidean TSP is NP-hard

TSP :: known complexity and approximation results

Complexity

- (Karp, 1972) TSP is strongly NP-hard
- (Sahni and Gonzales, 1976) TSP can not be approximated within $O(2^n)$ (unless $P = NP$)
- (Papadimitriou, 1977) Euclidean TSP is NP-hard

Approximation

- (Christofides, 1976) Metric TSP belongs to Apx
- (Arora, 1996; Mitchell, 1996) First Polynomial Time Approximation Schemes (PTAS) for TSP on the plane
- (Arora, 1998) Euclidean TSP in $\mathbb{R}^d$ for any fixed $d > 1$ has EPTAS (but has no FPTAS unless $P = NP$)
- (Serdyukov, 1987; Gimadi, 2001) Euclidean max-TSP has asymptotically correct algorithms

Generalizations of TSP

1. Min-k-SCCP — the Multiple Traveling Salesmen Problem
   - Problem statement
   - Complexity and Approximability
   - Metric Min-k-SCCP
   - PTAS for Euclidean Min-2-SCCP on the plane
2. Generalized Traveling Salesman Problem
   - Problem statement
   - Dynamic programming
   - Precedence constraints
   - Practical application
   - Euclidean GTSP in Grid Clusters
3. Conslusion

Multiple TSP - overview

- For a given natural k, a problem of k collaborating salesmen sharing the same set of cities (nodes of graph) to serve is studied.
- We call it Minimum Weight k-Size Cycle Cover Problem (Min-k-SCCP).
- Related problems
  - Min-1-SCCP is Traveling Salesman Problem (TSP)
  - **Vertex-Disjoint Cycle Cover Problem**
  - k-Peripatetic Salesmen Problem
  - Min-L-CCP
- Min-k-SCCP can be considered as a special case of Vehicle Routing Problem (VRP)

Multiple TSP - overview

- For a given natural k, a problem of k collaborating salesmen sharing the same set of cities (nodes of graph) to serve is studied.
- We call it Minimum Weight k-Size Cycle Cover Problem (Min-k-SCCP).
- Related problems
  - Min-1-SCCP is Traveling Salesman Problem (TSP)
  - **Vertex-Disjoint Cycle Cover Problem**
  - k-Peripatetic Salesmen Problem
  - Min-L-CCP
- Min-k-SCCP can be considered as a special case of Vehicle Routing Problem (VRP)

Multiple TSP - overview

- For a given natural k, a problem of k collaborating salesmen sharing the same set of cities (nodes of graph) to serve is studied.
- We call it Minimum Weight k-Size Cycle Cover Problem (Min-k-SCCP).
- Related problems
  - Min-1-SCCP is Traveling Salesman Problem (TSP)
  - **Vertex-Disjoint Cycle Cover Problem**
  - k-Peripatetic Salesmen Problem
  - Min-L-CCP
- Min-k-SCCP can be considered as a special case of Vehicle Routing Problem (VRP)

Multiple TSP - overview

- For a given natural k, a problem of k collaborating salesmen sharing the same set of cities (nodes of graph) to serve is studied.
- We call it Minimum Weight k-Size Cycle Cover Problem (Min-k-SCCP).
- Related problems
  - Min-1-SCCP is Traveling Salesman Problem (TSP)
  - **Vertex-Disjoint Cycle Cover Problem**
  - k-Peripatetic Salesmen Problem
  - Min-L-CCP
- Min-k-SCCP can be considered as a special case of Vehicle Routing Problem (VRP)

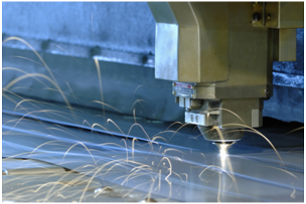
Multiple TSP :: motivation

- Nuclear Power Plant dismantling problem

Multiple TSP :: motivation

- Nuclear Power Plant dismantling problem

- high-precision metal shape cutting problem



Multiple TSP :: recent results

1. Min-k-SCCP is strongly NP-hard and hardly approximable in the general case
2. Metric and Euclidean cases are intractable as well
3. 2-approximation algorithm for Metric Min-k-SCCP is proposed
4. For any fixed $d > 1$, Polynomial-time approximation scheme (PTAS) for Min-k-SCCP in $\mathbb{R}^d$ is constructed

Definitions and Notation

Standard notation is used

- $\mathbb{R}$ — field of real numbers
- $\mathbb{Q}$ — field of rational numbers
- $\mathbb{N}_m$ — integer segment $\{1, ..., m\}$,
- $\mathbb{N}_m^0$ — segment $\{0, ..., m\}$.
- $G = (V, E, w)$ is a simple complete weighted (di)graph with loops, edge-weight function $w : E \to \mathbb{R}$

Minimum Weight k-Size Cycle Cover Problem (Min-k-SCCP)

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

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

Minimum Weight k-Size Cycle Cover Problem (Min-k-SCCP)

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

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

$$\begin{aligned} \min \quad & \sum_{i=1}^{k} W(C_i) \equiv \sum_{i=1}^{k} \sum_{e \in E(C_i)} w(e) \\ s.t. \quad & \\ & C_1, \ldots, C_k \text{ are cycles in } G \\ & C_i \cap C_j = \varnothing \\ & V(C_1) \cup \ldots \cup V(C_k) = V \end{aligned}$$

Metric and Euclidean Min-k-SCCP

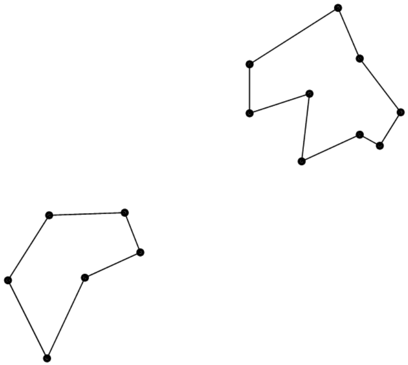
Metric Min-k-SCCP

- $w_{ij} \geqslant 0$
- $w_{ii} = 0$
- $w_{ij} = w_{ji}$
- $w_{ij} + w_{jk} \geqslant w_{ik} \quad (\{i, j, k\})$

Euclidean Min-k-SCCP

- For some $d > 1$, $V = \{v_1, \ldots, v_n\} \subset \mathbb{R}^d$
- $w_{ij} = \|v_i - v_j\|_2$

Instance of Euclidean Min-2-SCCP



Complexity

Theorem 1

For any $k \geqslant 1$, Min-k-SCCP is strongly NP-hard.

Complexity

Theorem 1

For any $k \geqslant 1$, Min-k-SCCP is strongly NP-hard.

Proof idea

- Reduce TSP to Min-k-SCCP by cloning the instance
- Spread them apart
- Show that any optimal solution of Min-k-SCCP consists of cheapest Hamiltonian cycles for the initial TSP

Complexity

Theorem 1

For any $k \geqslant 1$, Min-k-SCCP is strongly NP-hard.

Proof idea

- Reduce TSP to Min-k-SCCP by cloning the instance
- Spread them apart
- Show that any optimal solution of Min-k-SCCP consists of cheapest Hamiltonian cycles for the initial TSP

Corollary

- Min-k-SCCP also can not be approximated within $O(2^n)$ (unless $P = NP$)
- Metric Min-k-SCCP and Euclidean Min-k-SCCP are NP-hard as well

Minimum spanning forest

- k-forest is an acyclic graph with k connected components
- For any k-forest F, weight (cost)

$$W(F) = \sum_{e \in E(F)} w(e)$$

- k-Minimum Spanning Forest (k-MSF) Problem

Kruskal's algorithm for k-MSF

1. Start from the empty n-forest F_0.
2. For each $i \in \mathbb{N}_{n-k}$ add the edge

$$e_i = \arg\min\{w(e)\colon F_{i-1} \cup \{e\} \text{ remains acyclic}\}$$

   to the forest F_{i-1}.
3. Output k-forest F^*.

Kruskal's algorithm for k-MSF

1. Start from the empty n-forest F_0.
2. For each $i \in \mathbb{N}_{n-k}$ add the edge

$$e_i = \arg\min\{w(e) \colon F_{i-1} \cup \{e\} \text{ remains acyclic}\}$$

   to the forest F_{i-1}.
3. Output k-forest F^*.

Theorem 2

F^* is k-Minimum Spanning Forest.

2-approximation algorithm for Metric Min-k-SCCP

Following to the scheme of well-known 2-approx. algorithm for Metric TSP.
Wlog. assume $k < n$.
Algorithm:

1. Build a k-MSF F
2. Take edges of F twice
3. For any non-trivial connected component, find a Eulerian cycle
4. Transform them into Hamiltonian cycles
5. Output collection of these cycles adorned by some number of isolated vertices

Correctness proof

Assertion

Approximation ratio:

$$2(1-2/n) \leqslant \frac{APP}{OPT} \leqslant 2(1-1/n)$$

Running-time:

$$O(n^2 \log n).$$

Proof sketch

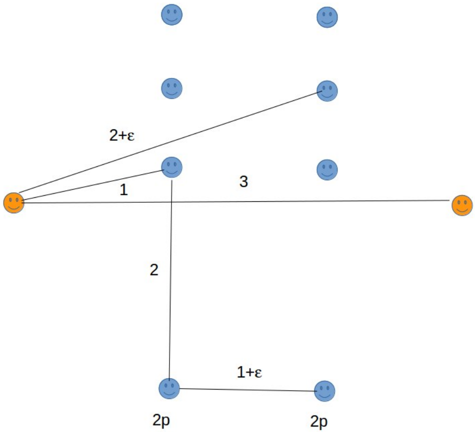
Consider optimal cycle cover $\mathcal{C}$ (with weight OPT).
Removing the most heavy edge from any non-empty cycle transform it into some spanning forest $F(\mathcal{C})$ with cost SF.
Then

$$MSF \leqslant SF \leqslant OPT(1-1/n),$$

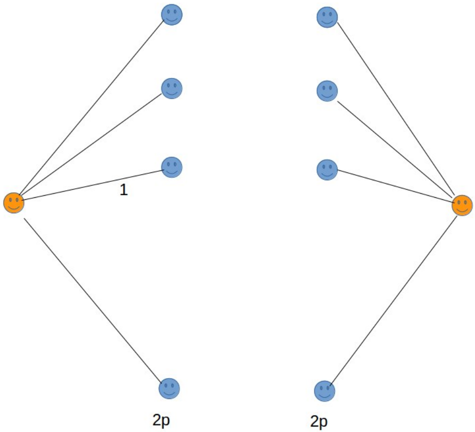
where

$$APP \leqslant 2 \cdot MSF \leqslant 2(1-1/n)OPT.$$

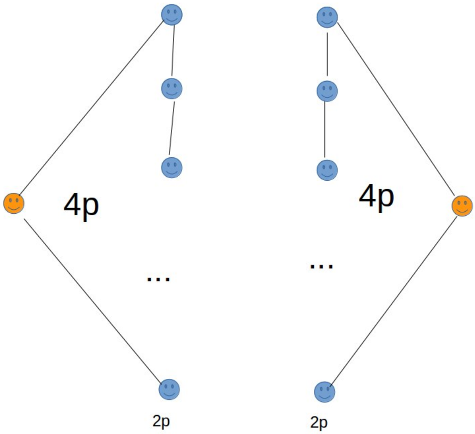
Lower bound - instance



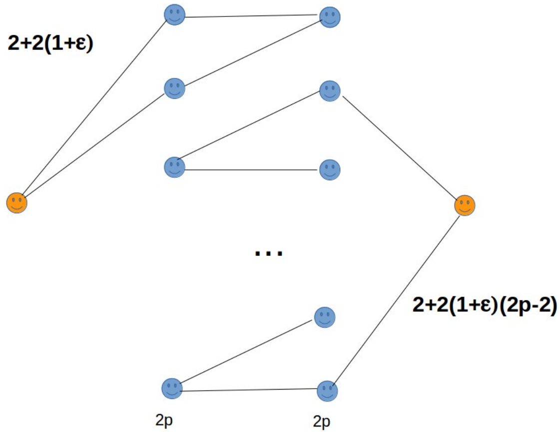
Lower bound - 2-forest



Lower bound - approximation



Lower bound - better approximation



Lower bound - discussion

- number of nodes $n = 4p + 2$
- $APP = 8p$
- $OPT \leq 4p + 2 + 2\varepsilon(2p - 1)$
- for approximation ratio r we have

$$r \geq \sup_{\varepsilon \in (0,1)} \frac{8p}{4p + 2 + 2\varepsilon(2p - 1)} = \frac{4p}{2p + 1} = 2(1 - 2/n)$$

PTAS for Euclidean Min-2-SCCP on the plane

Definition

For a combinatorial optimization problem, Polynomial-Time Approximation Scheme (PTAS) is a collection of algorithms such that for any fixed $c > 1$ there is an algorithm finding a $(1 + 1/c)$-approximate solution in a polynomial time depending on c.

Instance preprocessing

For an arbitrary instance of Min-2-SCCP, there exists one of the following alternatives (each of them can be verified in polynomial time)

1. The instance in question can be decomposed into 2 independent TSP instances;
2. Inter-node distance can be overestimated using some function that depends on OPT linearly.

Jung's inequality

Consider a set S of diameter D in d-dimensional Euclidean space, let R be a radius of the smallest containing sphere.
Then

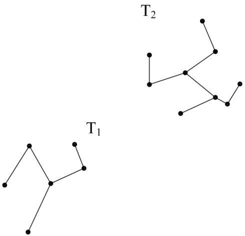
$$\frac{1}{2}D \leqslant R \leqslant \left(\frac{d}{2d+2}\right)^{\frac{1}{2}} D.$$

In particular, in the plane:

$$\frac{1}{2}D \leqslant R \leqslant \frac{\sqrt{3}}{3}D. \tag{1}$$

Instance preprocessing - ctd.

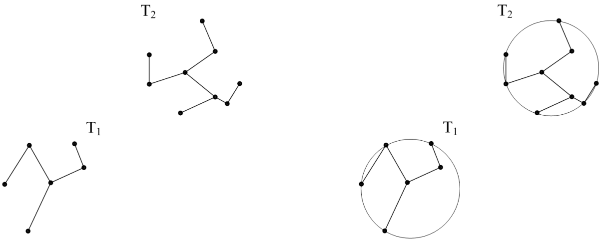
- Construct 2-MSF consisting of trees T_1 and T_2.



- let D_1, D_2 be diameters of T_1 and T_2, and R_1, R_2 be radia of the smallest circles $B(T_1)$ and $B(T_2)$ containing the trees T_1 and T_2. Denote $D = \max\{D_1, D_2\}$ and $R = \max\{R_1, R_2\}$.

Instance preprocessing - ctd.

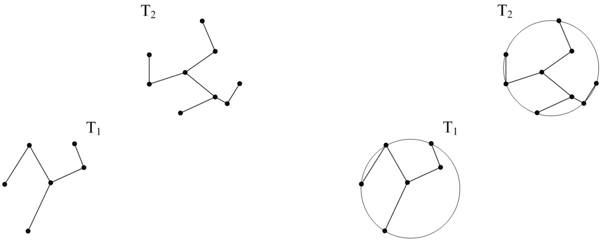
- Construct 2-MSF consisting of trees T_1 and T_2.



- let D_1, D_2 be diameters of T_1 and T_2, and R_1, R_2 be radia of the smallest circles $B(T_1)$ and $B(T_2)$ containing the trees T_1 and T_2. Denote $D = \max\{D_1, D_2\}$ and $R = \max\{R_1, R_2\}$.

Instance preprocessing - ctd.

- Construct 2-MSF consisting of trees T_1 and T_2.



- let D_1, D_2 be diameters of T_1 and T_2, and R_1, R_2 be radia of the smallest circles $B(T_1)$ and $B(T_2)$ containing the trees T_1 and T_2. Denote $D = \max\{D_1, D_2\}$ and $R = \max\{R_1, R_2\}$.

Problem decomposition

Define $\rho(T_1, T_2)$ as a distance between centers of circles $B(T_1)$ and $B(T_2)$.

Assertion

If $\rho(T_1, T_2) > 5R$ then the considered instance Min-2-SCCP can be decomposed into two TSP instances for $G(T_1)$ and $G(T_2)$.

Problem decomposition

Define $\rho(T_1, T_2)$ as a distance between centers of circles $B(T_1)$ and $B(T_2)$.

Assertion

If $\rho(T_1, T_2) > 5R$ then the considered instance Min-2-SCCP can be decomposed into two TSP instances for $G(T_1)$ and $G(T_2)$.

Proof sketch

Suppose, on the contrary, that there is an optimal 2-SCC $\mathcal{C} = \{C_1, C_2\}$ such that $C_1 \cap T_1 \neq \varnothing$ and $C_1 \cap T_2 \neq \varnothing$.

Problem decomposition

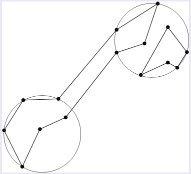
Define $\rho(T_1, T_2)$ as a distance between centers of circles $B(T_1)$ and $B(T_2)$.

Assertion

If $\rho(T_1, T_2) > 5R$ then the considered instance Min-2-SCCP can be decomposed into two TSP instances for $G(T_1)$ and $G(T_2)$.

Proof sketch

Suppose, on the contrary, that there is an optimal 2-SCC $\mathcal{C} = \{C_1, C_2\}$ such that $C_1 \cap T_1 \neq \varnothing$ and $C_1 \cap T_2 \neq \varnothing$.



Problem decomposition

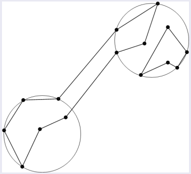
Define $\rho(T_1, T_2)$ as a distance between centers of circles $B(T_1)$ and $B(T_2)$.

Assertion

If $\rho(T_1, T_2) > 5R$ then the considered instance Min-2-SCCP can be decomposed into two TSP instances for $G(T_1)$ and $G(T_2)$.

Proof sketch

Suppose, on the contrary, that there is an optimal 2-SCC $\mathcal{C} = \{C_1, C_2\}$ such that $C_1 \cap T_1 \neq \varnothing$ and $C_1 \cap T_2 \neq \varnothing$.

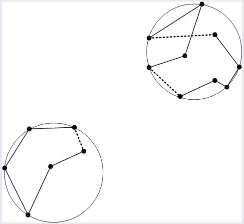


Then C_1 contains at least two edges, spanning T_1 and T_2

Problem decomposition

Proof (ctd.)

- By the condition, the weight of each of them is greater than $3R$
- Remove them and close the cycles inside $B(T_1)$ and $B(T_2)$



- Obtain the lighter 2-SCC

Problem decomposition

Statement

If $\rho(T_1, T_2) \leqslant 5R$ then the maximum inter-node distance $D(G)$ for the graph G is no more than $\frac{7\sqrt{3}}{3}OPT$.

Problem decomposition

Statement

If $\rho(T_1, T_2) \leqslant 5R$ then the maximum inter-node distance $D(G)$ for the graph G is no more than $\frac{7\sqrt{3}}{3} OPT$.

Proof sketch

- In our case $D(G) \leqslant 7R$
- Due to Young's inequality and $D \leqslant MSF \leqslant OPT$ we have

$$R \leqslant \frac{\sqrt{3}}{3} D \leqslant \frac{\sqrt{3}}{3} \cdot OPT,$$

- i.e. $D(G) \leqslant \frac{7\sqrt{3}}{3} \cdot OPT$.

Problem decomposition

Statement

If $\rho(T_1, T_2) \leqslant 5R$ then the maximum inter-node distance $D(G)$ for the graph G is no more than $\frac{7\sqrt{3}}{3} OPT$.

Proof sketch

- In our case $D(G) \leqslant 7R$
- Due to Young's inequality and $D \leqslant MSF \leqslant OPT$ we have

$$R \leqslant \frac{\sqrt{3}}{3} D \leqslant \frac{\sqrt{3}}{3} \cdot OPT,$$

- i.e. $D(G) \leqslant \frac{7\sqrt{3}}{3} \cdot OPT$.

In this case Min-2-SCCP instance can be enclosed into some axis-aligned square $\mathcal{S}$ of size $7/\sqrt{3} \cdot OPT$

Rounding

Definition

Instance of Min-2-SCCP is called *rounded* if

- every vertex of the graph G has integral coordinates $x_i, y_i \in \mathbb{N}^0_{O(n)}$
- for any edge e, $w(e) \geqslant 4$

Lemma 3

PTAS for rounded Min-2-SCCP implies PTAS for Min-2-SCCP (in the general case)

Rounding: proof sketch

- partition the surrounding square by axis-alined lines with step of $L/(2nc)$
- move any node to nearest line-crossing point; inter-node distance change is bounded by $L/(nc)$; cycle cover weight change bound is L/c
- shift the origin to left-bottom corner of the square; by scaling coordinates by $8nc/L$ obtain a 4-step integer grid

Rounding: proof sketch

- partition the surrounding square by axis-alined lines with step of $L/(2nc)$
- move any node to nearest line-crossing point; inter-node distance change is bounded by $L/(nc)$; cycle cover weight change bound is L/c
- shift the origin to left-bottom corner of the square; by scaling coordinates by $8nc/L$ obtain a 4-step integer grid

Rounding: proof sketch

- partition the surrounding square by axis-alined lines with step of $L/(2nc)$
- move any node to nearest line-crossing point; inter-node distance change is bounded by $L/(nc)$; cycle cover weight change bound is L/c
- shift the origin to left-bottom corner of the square; by scaling coordinates by $8nc/L$ obtain a 4-step integer grid

Rounding: proof sketch - ctd.

- for weights W and W' of any corresponding cycle covers in the initial and the rounded instances

$$\frac{8nc}{L}\Big(W-\frac{L}{c}\Big)\leqslant W'\leqslant\frac{8nc}{L}\Big(W+\frac{L}{c}\Big)$$

- For optimum values OPT and OPT' and weights W and W' of the approximate solutions

$$OPT'\leqslant W'\leqslant\Big(1+\frac{1}{c}\Big)OPT' \text{ and } \frac{8nc}{L}\Big(OPT-\frac{L}{c}\Big)\leqslant OPT'\leqslant\frac{8nc}{L}\Big(OPT+\frac{L}{c}\Big),$$

- Then,

$$\frac{8nc}{L}\Big(W-\frac{L}{c}\Big)\leqslant W'\leqslant\Big(1+\frac{1}{c}\Big)OPT'\leqslant\Big(1+\frac{1}{c}\Big)\frac{8nc}{L}\Big(OPT+\frac{L}{c}\Big),$$

- Therefore,

$$W-\frac{L}{c}\leqslant\Big(1+\frac{1}{c}\Big)\Big(OPT+\frac{L}{c}\Big)$$

- And

$$OPT\leqslant W\leqslant\Big(\frac{7\sqrt{3}}{3c^2}+\frac{17\sqrt{3}}{3c}+1\Big)OPT.$$

Rounding: proof sketch - ctd.

- for weights W and W' of any corresponding cycle covers in the initial and the rounded instances

$$\frac{8nc}{L}\left(W - \frac{L}{c}\right) \leqslant W' \leqslant \frac{8nc}{L}\left(W + \frac{L}{c}\right)$$

- For optimum values OPT and OPT' and weights W and W' of the approximate solutions

$$OPT' \leqslant W' \leqslant \left(1+\frac{1}{c}\right)OPT' \text{ and } \frac{8nc}{L}\left(OPT-\frac{L}{c}\right) \leqslant OPT' \leqslant \frac{8nc}{L}\left(OPT+\frac{L}{c}\right),$$

- Then,

$$\frac{8nc}{L}\left(W - \frac{L}{c}\right) \leqslant W' \leqslant \left(1 + \frac{1}{c}\right) OPT' \leqslant \left(1 + \frac{1}{c}\right)\frac{8nc}{L}\left(OPT + \frac{L}{c}\right),$$

- Therefore,

$$W - \frac{L}{c} \leqslant \left(1 + \frac{1}{c}\right)\left(OPT + \frac{L}{c}\right)$$

- And

$$OPT \leqslant W \leqslant \left(\frac{7\sqrt{3}}{3c^2} + \frac{17\sqrt{3}}{3c} + 1\right) OPT.$$

Rounding: proof sketch - ctd.

- for weights W and W' of any corresponding cycle covers in the initial and the rounded instances

$$\frac{8nc}{L}\Big(W-\frac{L}{c}\Big)\leqslant W'\leqslant\frac{8nc}{L}\Big(W+\frac{L}{c}\Big)$$

- For optimum values OPT and OPT' and weights W and W' of the approximate solutions

$$OPT'\leqslant W'\leqslant\Big(1+\frac{1}{c}\Big)OPT' \text{ and } \frac{8nc}{L}\Big(OPT-\frac{L}{c}\Big)\leqslant OPT'\leqslant\frac{8nc}{L}\Big(OPT+\frac{L}{c}\Big),$$

- Then,

$$\frac{8nc}{L}\Big(W-\frac{L}{c}\Big)\leqslant W'\leqslant\Big(1+\frac{1}{c}\Big)OPT'\leqslant\Big(1+\frac{1}{c}\Big)\frac{8nc}{L}\Big(OPT+\frac{L}{c}\Big),$$

- Therefore,

$$W-\frac{L}{c}\leqslant\Big(1+\frac{1}{c}\Big)\Big(OPT+\frac{L}{c}\Big)$$

- And

$$OPT\leqslant W\leqslant\Big(\frac{7\sqrt{3}}{3c^2}+\frac{17\sqrt{3}}{3c}+1\Big)OPT.$$

Rounding: proof sketch - ctd.

- for weights W and W' of any corresponding cycle covers in the initial and the rounded instances

$$\frac{8nc}{L}\left(W-\frac{L}{c}\right)\leqslant W'\leqslant\frac{8nc}{L}\left(W+\frac{L}{c}\right)$$

- For optimum values OPT and OPT' and weights W and W' of the approximate solutions

$$OPT'\leqslant W'\leqslant\left(1+\frac{1}{c}\right)OPT' \text{ and } \frac{8nc}{L}\left(OPT-\frac{L}{c}\right)\leqslant OPT'\leqslant\frac{8nc}{L}\left(OPT+\frac{L}{c}\right),$$

- Then,

$$\frac{8nc}{L}\left(W-\frac{L}{c}\right)\leqslant W'\leqslant\left(1+\frac{1}{c}\right)OPT'\leqslant\left(1+\frac{1}{c}\right)\frac{8nc}{L}\left(OPT+\frac{L}{c}\right),$$

- Therefore,

$$W-\frac{L}{c}\leqslant\left(1+\frac{1}{c}\right)\left(OPT+\frac{L}{c}\right)$$

- And

$$OPT\leqslant W\leqslant\left(\frac{7\sqrt{3}}{3c^2}+\frac{17\sqrt{3}}{3c}+1\right)OPT.$$

Rounding: proof sketch - ctd.

- for weights W and W' of any corresponding cycle covers in the initial and the rounded instances

$$\frac{8nc}{L}\Big(W - \frac{L}{c}\Big) \leqslant W' \leqslant \frac{8nc}{L}\Big(W + \frac{L}{c}\Big)$$

- For optimum values OPT and OPT' and weights W and W' of the approximate solutions

$$OPT' \leqslant W' \leqslant \Big(1+\frac{1}{c}\Big)OPT' \text{ and } \frac{8nc}{L}\Big(OPT-\frac{L}{c}\Big) \leqslant OPT' \leqslant \frac{8nc}{L}\Big(OPT+\frac{L}{c}\Big),$$

- Then,

$$\frac{8nc}{L}\Big(W - \frac{L}{c}\Big) \leqslant W' \leqslant \Big(1 + \frac{1}{c}\Big) OPT' \leqslant \Big(1 + \frac{1}{c}\Big)\frac{8nc}{L}\Big(OPT + \frac{L}{c}\Big),$$

- Therefore,

$$W - \frac{L}{c} \leqslant \Big(1 + \frac{1}{c}\Big)\Big(OPT + \frac{L}{c}\Big)$$

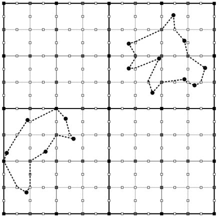
- And

$$OPT \leqslant W \leqslant \Big(\frac{7\sqrt{3}}{3c^2} + \frac{17\sqrt{3}}{3c} + 1\Big) OPT.$$

Main idea: construct PTAS for rounded instances

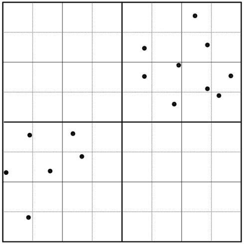
Randomized partitioning of the square $\mathcal{S}$ into smaller subsquares and subsequent search for minimum 2-SCC of special kind

1) every inter-node segment of its cycles is piece-wise linear and intersects all squares' borders at special points (*portals*) only;
2) portals number and locations together with maximum number of intersections (for each border) are defined in advance and depend on accuracy parameter c;



Quad-trees for rounded Min-2-SCCP

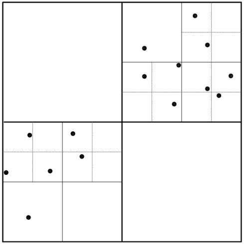
Set up a regular 1-step axis-aligned grid on the square $\mathcal{S}$ with side-length of $L = O(n)$.



We are using the concept of *quad-tree*

Quad-trees for rounded Min-2-SCCP

Root is the square $\mathcal{S}$. For every square (including the root), make a partition of the square into 4 child subsquares. Repeat it until all child squares will contain no more than 1 node of the instance.



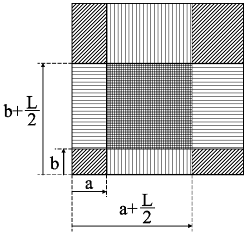
Shifted Quad-tree

Definition

Suppose, $a, b \in \mathbb{N}_L^0$, we call the Quad-tree $T(a, b)$ *shifted Quad-tree*, if coordinates of its center is

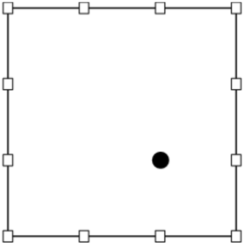
$$((L/2 + a) \mod L, (L/2 + b) \mod L).$$

Child squares of $T(a, b)$, as its center, is considered *modulo* L



Definition

- Consider fixed values $m, r \in \mathbb{N}$.
- For any square S, assign regular partition of its border, including vertices of the square and consisting of $4(m+1)$ points.
- Such a partition is called *m-regular partition*, and all its elements — *portals*.



Definitions

m-regular portal set

Union of m-regular partitions for all borders of not-a-leaf nodes of Quadro-tree $T(a, b)$ is called m-regular portal set. Denote it $P(a, b, m)$.

(m, r)-approximation

Suppose, π is a simple cycle in the Min-2-SCCP instance graph G (on the plane), $V(\pi)$ is its node-set. Closed piece-wise linear route $l(\pi)$ is called (m, r)-approximation (of the cycle π) if

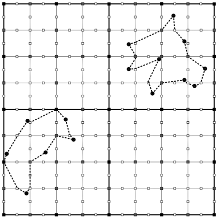
1) node-set of the route $l(\pi)$ is a some subset of $V(\pi) \cup P(a, b, m)$,
2) π and $l(\pi)$ visit the nodes from $V(\pi)$ in the same order,
3) for any square (being a node of $T(a, b)$), $l(\pi)$ intersects its arbitrary edge no more than r times, and exclusively in the points of $P(a, b, m)$.

Once more definition

(k, m, r)-cycle cover

k-scc consisting of (m, r)-approximations is called (k, m, r)-cycle cover

Obviously, an arbitrary $(1, m, r)$-cycle cover contains the only (m, r)-approximation which is a Hamiltonian cycle.
Let us consider $(2, m, r)$-cycle covers...



Structure Theorem for Euclidean Min-2-SCCP

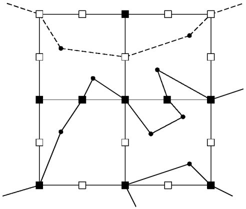
Theorem 4

- *Suppose* $c > 0$ *is fixed,*
- L *is size of square* $\mathcal{S}$ *for a given instance of rounded 2-MHC.*
- *Suppose discrete stochastic variables* a, b *are distributed uniformly on the set* $\mathbb{N}_L^0$.
- *Then for* $m = O(c \log L)$ *and* $r = O(c)$ *with probability at least* $\frac{1}{2}$ *there is* $(2, m, r)$*-cycle cover which weight is no more than* $(1 + \frac{1}{c})OPT$.

Dynamic Programming

$(2, m, r, S)$-segment

Let some $(2, m, r)$-cycle cover C and some node S of the tree $T(a, b)$ be chosen. A family of partial routes $C \cap S$ is called $(2, m, r, S)$-segment (of the cover C).



Bellman equation

Task (S, R_1, R_2, κ)

Input.

- Node S of the tree $T(a, b)$.
- Cortege $R_i : \mathbb{N}_{q_i} \to (P(a, b, m) \cap \partial S)^2$ defines a sequence of the start-finish pairs of portals (s^i_j, t^i_j) which are crossing-points of ∂S by (m, r)-approximation l_i.
- Number κ is equal to the number of cycles of the building $(2, m, r)$-cycle cover, intersecting the interior of S.

Output minimum-cost $(2, m, r, S)$-segment.

Denote by $W(S, R_1, R_2, \kappa)$ value of the task (S, R_1, R_2, κ).

$$W(S, R_1, R_2, \kappa) = \min_{\tau} \sum_{i=I}^{IV} W(S^i, R^i_1(\tau), R^i_2(\tau), \kappa^i(\tau)),$$

Derandomization

Denote by $APP(a, b)$ a weight of the approximate solution constructed by DP for the tree $T(a, b)$.

$$P\left(APP(a, b) \leqslant (1 + \frac{1}{c})OPT\right) \geqslant 1/2,$$

Hence, there is a pair $(a^*, b^*) \in \mathbb{N}_L^0$, for which the equation

$$OPT \leqslant APP(a^*, b^*) \leqslant (1 + 1/c)OPT$$

is valid.

Min-k-SCCP :: results

Theorem 5

Euclidean Min-2-SCCP has a Polynomial-Time Approximation Scheme with complexity bound $O(n^3(\log n)^{O(c)})$.

Theorem 6

For any $d > 1$, *the Euclidean Min-k-SCCP in* $\mathbb{R}^d$ *has PTAS with time complexity* $O(n^{d+1}2^k(k\log n)^{O((\sqrt{d}/\varepsilon)^{d-1})})$.

Contents

Contents

Overview

We consider the combinatorial optimization problem of visiting clusters of a fixed number of nodes (cities) under the special type of precedence constraints.

- This problem is a kind of the Generalized Traveling Salesman Problem (GTSP).
- To find an optimal solution of the problem, we propose a dynamic programming based on algorithm extending the well known Held and Karp technique.
- In terms of special type of precedence constraints, we describe subclasses of the problem, with polynomial (or even linear) in n upper bounds of time complexity.

Overview

We consider the combinatorial optimization problem of visiting clusters of a fixed number of nodes (cities) under the special type of precedence constraints.

- This problem is a kind of the Generalized Traveling Salesman Problem (GTSP).
- To find an optimal solution of the problem, we propose a dynamic programming based on algorithm extending the well known Held and Karp technique.
- In terms of special type of precedence constraints, we describe subclasses of the problem, with polynomial (or even linear) in n upper bounds of time complexity.

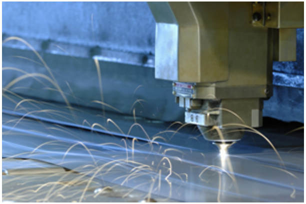
Overview

We consider the combinatorial optimization problem of visiting clusters of a fixed number of nodes (cities) under the special type of precedence constraints.

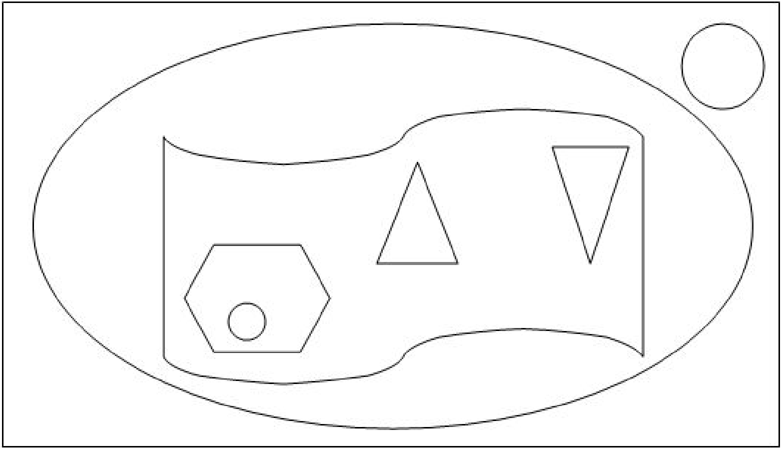
- This problem is a kind of the Generalized Traveling Salesman Problem (GTSP).
- To find an optimal solution of the problem, we propose a dynamic programming based on algorithm extending the well known Held and Karp technique.
- In terms of special type of precedence constraints, we describe subclasses of the problem, with polynomial (or even linear) in n upper bounds of time complexity.

Motivation revisited

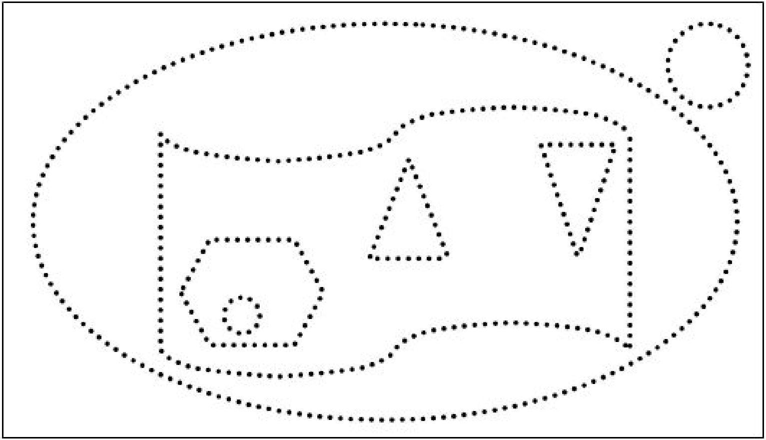
- high-precision metal shape cutting problem



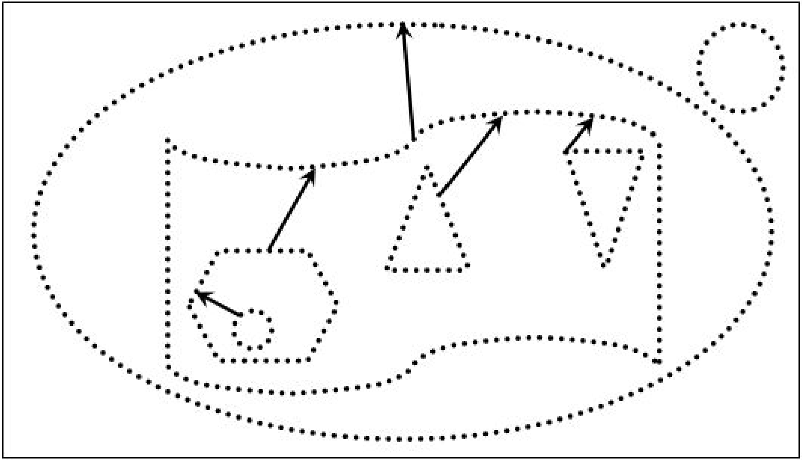
Motivation: precedence constraints



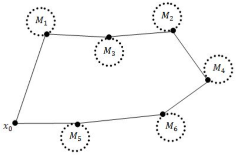
Motivation: precedence constraints



Motivation: precedence constraints



Problem statement

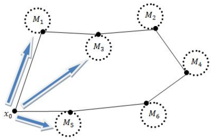


Inputs:

- disjunctive clusters $M_1, \ldots, M_n$, $M_j = \{g_{j1}, \ldots, g_{jp}\}$;
- start point $x_0 \notin \cup M_i$;
- transportation costs: $\hat{c}(x_0, g_{j\tau})$ and $\check{c}(g_{j\tau}, x_0)$ $c(g_{l\sigma}, g_{j\tau})$ for any $j, l \in \mathbb{N}_n = \{1, \ldots, n\}$, $j \neq l$ and $\sigma, \tau \in \mathbb{N}_p$;
- visiting (service) costs: $c'(g_{j\tau})$

Output:
the cheapest tour that starts and finishes at x_0.

Problem statement

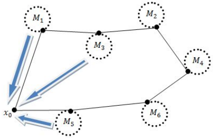


Inputs:

- disjunctive clusters $M_1, \ldots, M_n$, $M_j = \{g_{j1}, \ldots, g_{jp}\}$;
- start point $x_0 \notin \cup M_i$;
- transportation costs: $\hat{c}(x_0, g_{j\tau})$ and $\check{c}(g_{j\tau}, x_0)$ $c(g_{l\sigma}, g_{j\tau})$ for any $j, l \in \mathbb{N}_n = \{1, \ldots, n\}$, $j \neq l$ and $\sigma, \tau \in \mathbb{N}_p$;
- visiting (service) costs: $c'(g_{j\tau})$

Output:
the cheapest tour that starts and finishes at x_0.

Problem statement



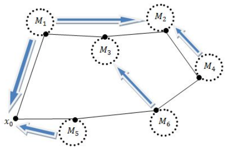
Inputs:

- disjunctive clusters $M_1, \dots, M_n$, $M_j = \{g_{j1}, \dots, g_{jp}\}$;
- start point $x_0 \notin \cup M_i$;
- transportation costs:
  $\hat{c}(x_0, g_{j\tau})$ and $\check{c}(g_{j\tau}, x_0)$
  $c(g_{l\sigma}, g_{j\tau})$ for any $j, l \in \mathbb{N}_n = \{1, \dots, n\}$, $j \neq l$ and $\sigma, \tau \in \mathbb{N}_p$;
- visiting (service) costs:
  $c'(g_{j\tau})$

Output:

the cheapest tour that starts and finishes at x_0.

Problem statement



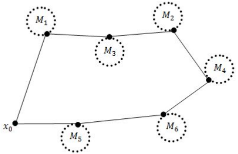
Inputs:

- disjunctive clusters $M_1, \ldots, M_n$, $M_j = \{g_{j1}, \ldots, g_{jp}\}$;
- start point $x_0 \notin \cup M_i$;
- transportation costs: $\hat{c}(x_0, g_{j\tau})$ and $\check{c}(g_{j\tau}, x_0)$ $c(g_{l\sigma}, g_{j\tau})$ for any $j, l \in \mathbb{N}_n = \{1, \ldots, n\}$, $j \neq l$ and $\sigma, \tau \in \mathbb{N}_p$;
- visiting (service) costs: $c'(g_{j\tau})$

Output:

the cheapest tour that starts and finishes at x_0.

Problem statement



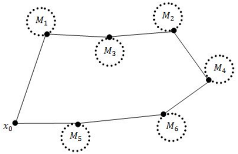
Inputs:

- disjunctive clusters $M_1, \ldots, M_n$, $M_j = \{g_{j1}, \ldots, g_{jp}\}$;
- start point $x_0 \notin \cup M_i$;
- transportation costs:
  $\hat{c}(x_0, g_{j\tau})$ and $\check{c}(g_{j\tau}, x_0)$
  $c(g_{l\sigma}, g_{j\tau})$ for any $j, l \in \mathbb{N}_n = \{1, \ldots, n\}$, $j \neq l$ and $\sigma, \tau \in \mathbb{N}_p$;
- visiting (service) costs:
  $c'(g_{j\tau})$

Output:

the cheapest tour that starts and finishes at x_0.

Problem statement



Inputs:

- disjunctive clusters $M_1, \dots, M_n$,
  $M_j = \{g_{j1}, \dots, g_{jp}\}$;
- start point $x_0 \notin \cup M_i$;
- transportation costs:
  $\hat{c}(x_0, g_{j\tau})$ and $\check{c}(g_{j\tau}, x_0)$
  $c(g_{l\sigma}, g_{j\tau})$ for any $j, l \in \mathbb{N}_n = \{1, \dots, n\}$,
  $j \neq l$ and $\sigma, \tau \in \mathbb{N}_p$;
- visiting (service) costs:
  $c'(g_{j\tau})$

Optimization stmt:

$$\hat{c}(x_0, g_{\pi(1)\tau(1)}) + \sum_{i=1}^{n-1} \left(c'(g_{\pi(i)\tau(i)}) + c(g_{\pi(i)\tau(i)}, g_{\pi(i+1)\tau(i+1)})\right) + \check{c}(g_{\pi(n)\tau(n)}, x_0) \to \min \quad (2)$$

Additional features

(i) transportation costs $c(g_{l\sigma}, g_{j\tau})$ and cluster visiting cost $c'(g_{j\tau})$ depend on the chosen sub-tour connecting x_0 and the node $g_{l\sigma}$;

(ii) Balas precedence constraints are defined on clusters:

Type I. For a natural number $k \leq n$, any feasible permutation π satisfies the equation

$$\forall i, j \in \mathbb{N}_n \quad (j \geq i + k) \Rightarrow (\pi(i) < \pi(j)). \tag{3}$$

Type II. For some natural values $1 \leq k(1), \ldots, k(n) \leq n$ and any feasible permutation π,

$$\forall i, j \in \mathbb{N}_n \ (j \geq i + k(i)) \Rightarrow (\pi(i) < \pi(j)). \tag{4}$$

Bellman equation

- Suppose, the optimal g-tour sourcing from x_0 and visiting for the first $i-1$ turns the clusters with indexes from $J \subset \mathbb{N}_n$, in the i-th turn, visits the cluster M_j at the node $g_{j\tau(i)} \in M_j$.
- Denote the cost of this g-subtour by $C(J, i, j, g_{j\tau(i)})$.
- The following recursive equations hold

$$C(\varnothing, 1, j, g_{j\tau(1)}) = \hat{c}(x_0, g_{j\tau(1)}), \tag{5}$$

$$\begin{aligned} C(J, i, j, g_{j\tau(i)}) = \min_{l \in J} \min_{g_{l\tau(i-1)} \in M_l} \{C(J \setminus \{l\}, i-1, l, g_{l\tau(i-1)}) \\ + c(g_{l\tau(i-1)}, g_{j\tau(i)}) + c'(g_{j\tau(i)})\}. \end{aligned} \tag{6}$$

- The optimum of the given instance of AGTSP can be found by the formula

$$C^* = \min_{j \in \mathbb{N}_n} (C(\mathbb{N}_n \setminus \{j\}, n, j, g_{j\tau(n)}) + \check{c}(g_{j\tau(n)}, x_0)). \tag{7}$$

Graphical representation: vertices

- Assign to the instance of the problem in question the following instance of the cheapest s-t-path problem in the appropriate $(n+2)$-layered edge-weighted digraph $G^*[p] = (V^*[p], A^*[p], w^*[p])$, whose vertices are states considered by dynamic programming procedure.
- Denote by $V_i^*[p]$ the vertex-set of the i-th layer such that

$$V_0^*[p] = \{s\}, V_{n+1}^*[p] = \{t\},$$

$$V_i^*[p] = \{(J, i, j, \tau) \colon j \in \mathbb{N}_n \setminus J,\ g_{j\tau} \in M_j,\ J \subset \mathbb{N}_n, |J| = i-1\} \quad (i \in \mathbb{N}_n).$$

- Assign vertices s and t both to the starting point x_0.
- Any other vertex (state) (J, i, j, τ) corresponds to i-turn subtour of the g-tour visiting clusters with indexes $J \cup \{j\}$, wherein the last visited cluster is M_j (at the node $g_{j\tau}$).

Graphical representation: vertices

- Assign to the instance of the problem in question the following instance of the cheapest s-t-path problem in the appropriate $(n+2)$-layered edge-weighted digraph $G^*[p] = (V^*[p], A^*[p], w^*[p])$, whose vertices are states considered by dynamic programming procedure.
- Denote by $V_i^*[p]$ the vertex-set of the i-th layer such that

$$V_0^*[p] = \{s\}, V_{n+1}^*[p] = \{t\},$$

$$V_i^*[p] = \{(J, i, j, \tau) \colon j \in \mathbb{N}_n \setminus J,\ g_{j\tau} \in M_j,\ J \subset \mathbb{N}_n, |J| = i-1\} \quad (i \in \mathbb{N}_n).$$

- Assign vertices s and t both to the starting point x_0.
- Any other vertex (state) (J, i, j, τ) corresponds to i-turn subtour of the g-tour visiting clusters with indexes $J \cup \{j\}$, wherein the last visited cluster is M_j (at the node $g_{j\tau}$).

Graphical representation: vertices

- Assign to the instance of the problem in question the following instance of the cheapest s-t-path problem in the appropriate $(n+2)$-layered edge-weighted digraph $G^*[p] = (V^*[p], A^*[p], w^*[p])$, whose vertices are states considered by dynamic programming procedure.
- Denote by $V_i^*[p]$ the vertex-set of the i-th layer such that

$$V_0^*[p] = \{s\}, V_{n+1}^*[p] = \{t\},$$

$$V_i^*[p] = \{(J, i, j, \tau) \colon j \in \mathbb{N}_n \setminus J,\ g_{j\tau} \in M_j,\ J \subset \mathbb{N}_n, |J| = i-1\} \quad (i \in \mathbb{N}_n).$$

- Assign vertices s and t both to the starting point x_0.
- Any other vertex (state) (J, i, j, τ) corresponds to i-turn subtour of the g-tour visiting clusters with indexes $J \cup \{j\}$, wherein the last visited cluster is M_j (at the node $g_{j\tau}$).

Graphical representation: vertices

- Assign to the instance of the problem in question the following instance of the cheapest s-t-path problem in the appropriate $(n+2)$-layered edge-weighted digraph $G^*[p] = (V^*[p], A^*[p], w^*[p])$, whose vertices are states considered by dynamic programming procedure.
- Denote by $V_i^*[p]$ the vertex-set of the i-th layer such that

$$V_0^*[p] = \{s\}, V_{n+1}^*[p] = \{t\},$$

$$V_i^*[p] = \{(J, i, j, \tau) \colon j \in \mathbb{N}_n \setminus J,\ g_{j\tau} \in M_j,\ J \subset \mathbb{N}_n, |J| = i-1\} \quad (i \in \mathbb{N}_n).$$

- Assign vertices s and t both to the starting point x_0.
- Any other vertex (state) (J, i, j, τ) corresponds to i-turn subtour of the g-tour visiting clusters with indexes $J \cup \{j\}$, wherein the last visited cluster is M_j (at the node $g_{j\tau}$).

Graphical representation: arcs

- Only vertexes of subsequent layers $V_i^*[p]$ and $V_{i+1}^*[p]$ can be adjacent.
- s is adjacent to any vertex from $V_1^*[p]$;
- any vertex from $V_n^*[p]$ is adjacent to t.
- Any other states (J, i, l, σ) and $(J', i+1, j, \tau)$ are adjacent if

$$|J| = i - 1,\ \ J' = J \cup \{l\},\ \ j \notin J',\ \ \sigma, \tau \in \mathbb{N}_p. \tag{8}$$

- We denote the set of arcs connecting $V_i^*[p]$ with $V_{i+1}^*[p]$ by $A_{i,i+1}^*[p]$.
- Arc weights are defined by the following equations

$$w^*[p](s, (\varnothing, 1, j, \tau)) = \hat{c}(x_0, g_{j\tau}),\ \ w^*[p]((\mathbb{N}_n \setminus \{j\}, n, j, \tau), t) = \check{c}(g_{j\tau}, x_0),$$

$$w^*[p]((J, i, l, \sigma), (J', i+1, j, \tau)) = c(g_{l\sigma}, g_{j\tau}) + c'(g_{j\tau}).$$

Graphical representation: arcs

- Only vertexes of subsequent layers $V_i^*[p]$ and $V_{i+1}^*[p]$ can be adjacent.
- s is adjacent to any vertex from $V_1^*[p]$;
- any vertex from $V_n^*[p]$ is adjacent to t.
- Any other states (J, i, l, σ) and $(J', i+1, j, \tau)$ are adjacent if

$$|J| = i - 1,\ \ J' = J \cup \{l\},\ \ j \notin J',\ \ \sigma, \tau \in \mathbb{N}_p. \tag{8}$$

- We denote the set of arcs connecting $V_i^*[p]$ with $V_{i+1}^*[p]$ by $A_{i,i+1}^*[p]$.
- Arc weights are defined by the following equations

$$w^*[p](s, (\varnothing, 1, j, \tau)) = \hat{c}(x_0, g_{j\tau}),\ \ w^*[p]((\mathbb{N}_n \setminus \{j\}, n, j, \tau), t) = \check{c}(g_{j\tau}, x_0),$$

$$w^*[p]((J, i, l, \sigma), (J', i+1, j, \tau)) = c(g_{l\sigma}, g_{j\tau}) + c'(g_{j\tau}).$$

Graphical representation: equivalence

Theorem 7

The set of feasible g-tours in AGTSP is isomorphic to the set of s-t-paths in the graph $G^*[p]$. Moreover, any corresponding g-tour and s-t-path have the same costs.

Corollary 8

The cheapest g-tour can be found in $O(|A^*[p]|)$ by the well known modification of the Ford-Bellman algorithm for circuit-free weighted digraph

Graphical representation: equivalence

Theorem 7

The set of feasible g-tours in AGTSP is isomorphic to the set of s-t-paths in the graph $G^*[p]$. Moreover, any corresponding g-tour and s-t-path have the same costs.

Corollary 8

The cheapest g-tour can be found in $O(|A^*[p]|)$ by the well known modification of the Ford-Bellman algorithm for circuit-free weighted digraph

- Unfortunately, for the general case of AGTSP, the number of arcs in the graph $G^*[p]$ grows exponentially as $n \to \infty$, i.e. time complexity of the proposed scheme of dynamic programming is exponential as well.
- Indeed, $|A^*[p]| = \Omega(np^2 2^n)$ for any $n \geq 2$.

Type I

Theorem 9

Suppose, for some $k \in \mathbb{N}$ and for any feasible permutation π,

$$\forall i, j \in \mathbb{N}_n \quad (j \geq i + k) \Rightarrow (\pi(i) < \pi(j)). \tag{9}$$

Then

$$|A^*[p]| = O(n \cdot p^2 k^2 2^{k-2}). \tag{10}$$

Corollary 10

- *If $k = o(\log n)$ and $p = O(\mathrm{poly}(n))$, then AGTSP can be solved optimally by dynamic programming in time $O(\mathrm{poly}(n))$.*
- *Moreover, for any fixed k and p, dynamic programming has time complexity $O(n)$.*

Type I

Theorem 9

Suppose, for some $k \in \mathbb{N}$ and for any feasible permutation π,

$$\forall i, j \in \mathbb{N}_n \quad (j \geq i + k) \Rightarrow (\pi(i) < \pi(j)). \tag{9}$$

Then

$$|A^*[p]| = O(n \cdot p^2 k^2 2^{k-2}). \tag{10}$$

Corollary 10

- *If $k = o(\log n)$ and $p = O(\mathrm{poly}(n))$, then AGTSP can be solved optimally by dynamic programming in time $O(\mathrm{poly}(n))$.*
- *Moreover, for any fixed k and p, dynamic programming has time complexity $O(n)$.*

Type II

Theorem 11

If, for some natural values $1 \le k(1), \ldots, k(n) \le n$ *and any feasible permutation* π*,*

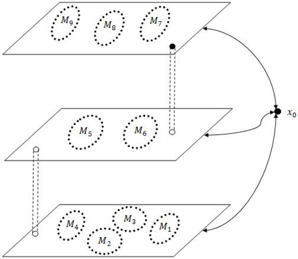
$$\forall i, j \in \mathbb{N}_n \ (j \ge i + k(i)) \Rightarrow (\pi(i) < \pi(j)), \tag{11}$$

then

$$|A^*[p]| = O\left(p^2 \sum_{i=1}^{n} k^*(i)(k^*(i)+1)2^{k^*(i)-2}\right),$$

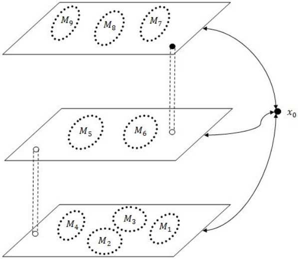
where $k^*(i) = \max\{k(j) \colon i - k(j) + 1 \le j \le i\}$.

Fire rescue plan



- Suppose, we need to construct the least expensive rescue plan visiting rooms located on three floors of some building.
- Rescue unit can start its job from any floor, to which it can be delivered for the vanishing cost.
- After the completion of the job, it can be escaped also from any floor.
- The main restriction is that moving from one floor to another can be done only through dedicated elevators and any such a transportation costs much more, than any moves around the floor.

Fire rescue plan



- Suppose, we need to construct the least expensive rescue plan visiting rooms located on three floors of some building.
- Rescue unit can start its job from any floor, to which it can be delivered for the vanishing cost.
- After the completion of the job, it can be escaped also from any floor.
- Equivalent representation

$$k(i) = \begin{cases} 5 - i, & \text{if } 1 \le i \le 4, \\ 7 - i, & \text{if } 5 \le i \le 6, \\ 10 - i, & \text{otherwise.} \end{cases}$$

Euclidean Generalized Traveling Salesman Problem (GTSP)

- We consider the Euclidian Generalized Traveling Salesman Problem in k Grid Clusters EGTSP-GC).

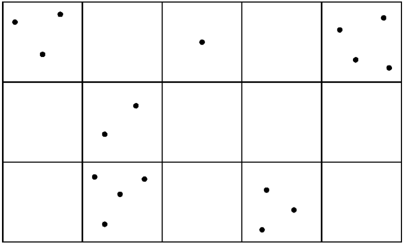


Figure: An EGTSP-GC instance for $k = 6$

Known results and related problems

- GTSP is strongly NP-hard even in Euclidean plane
- GTSP can be treated as a discrete version of Traveling Salesman Problem with Neighborhoods (TSPN) for which many solid approximation results are known (see, e.g. (Dumitrescu-Mitchell 2001), (Mitchell 2007), (Mitchell 2011))
- Good news: unlike TSPN, GTSP is polynomially solvable for any fixed k (Toth, 1995)
- The EGTSP-GC was introduced in (Bhattacharya et al. 2015). They showed that the problem is strongly NP-hard and proposed polynomial time $(1.5 + 8\sqrt{2} + \epsilon)$-approximation algorithm
- We present three approximation schemes for the EGTSP-GC. The first two are PTAS in the case, when $k = O(\log n)$, while the last one is a PTAS for $k = n - O(\log n)$

Known results and related problems

- GTSP is strongly NP-hard even in Euclidean plane
- GTSP can be treated as a discrete version of Traveling Salesman Problem with Neighborhoods (TSPN) for which many solid approximation results are known (see, e.g. (Dumitrescu-Mitchell 2001), (Mitchell 2007), (Mitchell 2011))
- Good news: unlike TSPN, GTSP is polynomially solvable for any fixed k (Toth, 1995)
- The EGTSP-GC was introduced in (Bhattacharya et al. 2015). They showed that the problem is strongly NP-hard and proposed polynomial time $(1.5 + 8\sqrt{2} + \epsilon)$-approximation algorithm
- We present three approximation schemes for the EGTSP-GC. The first two are PTAS in the case, when $k = O(\log n)$, while the last one is a PTAS for $k = n - O(\log n)$

Known results and related problems

- GTSP is strongly NP-hard even in Euclidean plane
- GTSP can be treated as a discrete version of Traveling Salesman Problem with Neighborhoods (TSPN) for which many solid approximation results are known (see, e.g. (Dumitrescu-Mitchell 2001), (Mitchell 2007), (Mitchell 2011))
- Good news: unlike TSPN, GTSP is polynomially solvable for any fixed k (Toth, 1995)
- The EGTSP-GC was introduced in (Bhattacharya et al. 2015). They showed that the problem is strongly NP-hard and proposed polynomial time $(1.5 + 8\sqrt{2} + \epsilon)$-approximation algorithm
- We present three approximation schemes for the EGTSP-GC. The first two are PTAS in the case, when $k = O(\log n)$, while the last one is a PTAS for $k = n - O(\log n)$

Known results and related problems

- GTSP is strongly NP-hard even in Euclidean plane
- GTSP can be treated as a discrete version of Traveling Salesman Problem with Neighborhoods (TSPN) for which many solid approximation results are known (see, e.g. (Dumitrescu-Mitchell 2001), (Mitchell 2007), (Mitchell 2011))
- Good news: unlike TSPN, GTSP is polynomially solvable for any fixed k (Toth, 1995)
- The EGTSP-GC was introduced in (Bhattacharya et al. 2015). They showed that the problem is strongly NP-hard and proposed polynomial time $(1.5 + 8\sqrt{2} + \epsilon)$-approximation algorithm
- We present three approximation schemes for the EGTSP-GC. The first two are PTAS in the case, when $k = O(\log n)$, while the last one is a PTAS for $k = n - O(\log n)$

Known results and related problems

- GTSP is strongly NP-hard even in Euclidean plane
- GTSP can be treated as a discrete version of Traveling Salesman Problem with Neighborhoods (TSPN) for which many solid approximation results are known (see, e.g. (Dumitrescu-Mitchell 2001), (Mitchell 2007), (Mitchell 2011))
- Good news: unlike TSPN, GTSP is polynomially solvable for any fixed k (Toth, 1995)
- The EGTSP-GC was introduced in (Bhattacharya et al. 2015). They showed that the problem is strongly NP-hard and proposed polynomial time $(1.5 + 8\sqrt{2} + \epsilon)$-approximation algorithm
- We present three approximation schemes for the EGTSP-GC. The first two are PTAS in the case, when $k = O(\log n)$, while the last one is a PTAS for $k = n - O(\log n)$

Outline

Approximation scheme based on dynamic programming

Algorithm 1 Scheme based on DP

Input: a given instance of the Euclidean GTSP on k grid clusters and a required accuracy level ε.
Output: a $(1+\varepsilon)$-approximate solution.

1: partition all k nonempty cells of the given grid into t^2 smaller subcells; the value t will be specified later;
2: to each j-th cell assign a finite set C_j consisting of centers of nonempty subcells;
3: **for all** $(c_1, \ldots, c_k) \in C_1 \times \ldots \times C_k$ **do**
4: using dynamic programming find an exact solution $S(c_1, \ldots, c_k)$ of the corresponding TSP instance;
5: **end for**
6: output the cheapest solution $S(c_1, \ldots, c_k)$.

Correctness proof

- Consider an arbitrary optimal solution of the initial GTSP-GC instance with k clusters
- The accumulated error caused by substitution of the initial nodes by the nearest centers does not exceed $k\sqrt{2}/t$
- To estimate k in terms of optimum of the initial GTSP, we use a recent approximation result for another combinatorial optimization problem defined on clusters, Generalized Minimum Spanning Tree Problem (GMSTP)

Lower bound for an optimum value

Theorem 12 (Bhattacharya, 2015)

Let OPT_{GMSTP} be an optimum value of an instance of the Euclidean GMSTP on k grid clusters, then $k \leq 4OPT_{GMSTP} + 4$.

Since any Hamiltonian cycle can be reduced to the corresponding spanning tree by excluding an arbitrary edge, therefore $OPT_{GTSP} \geq OPT_{GMSTP}$. Therefore, for the Euclidean GTSP, the same assertion is valid.

Corollary 13

Let $OPT_{GTSP\text{-}GC}$ be the optimum value of an instance of the Euclidean GTSP on k grid clusters, then $k \leq 4OPT_{GTSP\text{-}GC} + 4$.

Approximation scheme based on dynamic programming

So, for any $k > 4$ and $\varepsilon > 0$, taking a value of t such that

$$\frac{k\sqrt{2}}{t} \leq \frac{k-4}{4}\varepsilon \leq \varepsilon OPT_{\text{GTSP-GC}},$$

i.e.

$$t \geq \frac{4\sqrt{2}k}{(k-4)\varepsilon} = \frac{4\sqrt{2}}{\varepsilon}\left(1 + \frac{4}{k-4}\right) \geq \frac{20\sqrt{2}}{\varepsilon},$$

we guarantee that our accumulated error does not exceed $\varepsilon OPT_{\text{GTSP-GC}}$. It should be noticed that asymptotically we can obtain the same result even for $t \geq 4\sqrt{2}k/((k-4)\varepsilon)$ as $k \to \infty$.

Approximation scheme based on dynamic programming

Theorem 14

For any $\varepsilon > 0$, Algorithm 1 finds an $(1+\varepsilon)$-approximate solution of the GTSP on k grid clusters in time of $O(k^2(O(1/\varepsilon))^{2k}) + O(n)$.

Corollary 15

1. For any fixed number $k > 4$ and any $\varepsilon > 0$, Algorithm 1 finds an $(1+\varepsilon)$-approximate solution of the Euclidean GTSP on k clusters in a linear time with delay depending on ε.
2. For the Euclidean GTSP on $k = O(\log n)$ clusters Algorithm 1 is a PTAS with time complexity of $O((\log n)^2 n^{O(\log(1/\varepsilon))})$.

Outline

1. Min-k-SCCP — the Multiple Traveling Salesmen Problem
   - Problem statement
   - Complexity and Approximability
   - Metric Min-k-SCCP
   - PTAS for Euclidean Min-2-SCCP on the plane
2. Generalized Traveling Salesman Problem
   - Problem statement
   - Dynamic programming
   - Precedence constraints
   - Practical application
   - **Euclidean GTSP in Grid Clusters**
3. Conslusion

Extended Arora's scheme

Similarly to Arora's PTAS, the main idea of the proposed approximation scheme is based on randomized recursive partitioning of the axis-aligned bounding box of the given instance into smaller squares and successive searching for the minimum weight closed tour subject to the following constraints:

(i) any cluster V_i is visited at once;

(ii) between-node segments of the route are continuous piece-wise linear curves crossing the borders of all squares only in predefined points called *portals*;

(iii) locations of the portals and the maximum count of crossings for each border-line of the squares depend on the given accuracy ε.

Well-rounded instance of the EGTSP-GC

We call an instance of the EGTSP-GC *well-rounded* if

(i) where exists $L' = O(k)$ such that, for any node $v_i = [x_i, y_i]$ of the input graph G, its coordinates $x_i, y_i \in \{0, \ldots, L'\}$;

(ii) for any $u \neq v \in V$, $w(\{u, v\}) \geq 4$.

Lemma 16

Any PTAS for the well-rounded EGTSP-GC induces the appropriate PTAS for the EGTSP-GC with the same (up to the order) complexity bound.

Extended Arora's scheme

Let $\mathcal{S}$ be the smallest axis-aligned square containing the instance of the EGTSP-GC. W.l.o.g. let the side-length L' of $\mathcal{S}$ be some power of two.
Following the Arora's approach, we construct a dissection of $\mathcal{S}$ into smaller squares using vertical and horizontal lines. These lines are crossing the coordinate axes in integer-coordinate points with a step of length 1. By construction, every smallest-size square contains at most one node of the given instance.

Quadtree

The root of the tree is the bounding box $\mathcal{S}$. Each non-leaf square in the tree is partitioned into four equal child squares. This recursive partitioning stops on a square containing at most one node. By construction, the quadtree contains $O(k^2)$ leaves, $O(\log L') = O(\log k)$ levels and thus $O(k^2 \log k)$ squares in all.

Shifted quadtree $T(a, b)$

The center point of the quadtree is the point of crossing of the inner edges of the squares with the side-length $L'/2$. We consider a shifted quadtree $T(a, b)$ with the center point $((L'/2 + a) \mod L', (L'/2 + b) \mod L')$, where $a, b \in \mathbb{N}^0_{L'}$ are constants.
To some parameter values $m, r \in \mathbb{N}$, and any node in the quadtree $T(a, b)$ [square S], we assign a regular partition of the border S consisting of $4(m + 1)$ points including all the corners of S.

Shifted quadtree $T(a, b)$

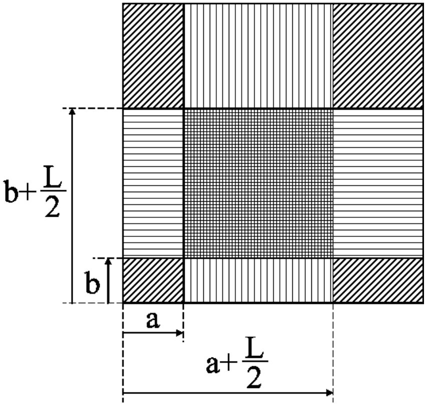


Figure: Shifted quadtree $T(a, b)$

(m, r)-approximation of the cycle

Definition 17

Let C be an arbitrary simple cycle in the graph G in the plane. The closed continuous piecewise linear route $l(C)$ is called an (m, r)-approximation of the cycle C if

(i) $l(C)$ bends only at nodes of given graph and portals;

(ii) the nodes of G are visited by $l(C)$ in the same order as by C;

(iii) for any side of any node of $T(a, b)$, the route $l(C)$ crosses this side at portals and at most r times.

Structure Theorem

Theorem 18

Let an instance of the well-rounded TSP in the plane be given by the graph G, let L be the side-length of the bounding box $\mathcal{S}$, and let constants $c > 1$ and $\eta \in (0, 1)$ be fixed. If the stochastic variables a and b are distributed uniformly in $\mathbb{N}_L$ and the parameters m and r are defined by the formulas

$$m = \lceil 2s \log L \rceil , \; r = s + 4, \quad \text{and } s = \lceil 36c/\eta \rceil .$$

Then, for an arbitrary simple cycle C of weight $W(C)$, with probability at least $1 - \eta$, there exists an (m, r)-approximation $l(C)$ of weight $W(l(C)) \leqslant (1 + 1/c)W(C)$.

The algorithm

Algorithm 2 Extended Arora's scheme

Input: a given instance of the Euclidean GTSP on k grid clusters and a required accuracy ε.
Output: a $(1+\varepsilon)$-approximate solution.

1: assign to the given instance the appropriate well-round instance enclosed in bouding box of size L';
2: **for all** $a, b \in \mathbb{N}^0_{L'}$ **do**
3: construct the shifted guadtree $T(a,b)$ and find $C(a,b)$ by dynamic procedure using approach proposed in (S.Arora, 1998) except that, for any internal node of the $T(a,b)$, the corresponding task along with conventional parameters depend on clusters $V_{i_1}, \ldots, V_{i_t}$ assigned to this node. Therefore, any child subtask of the Arora's DP produces up to 4^t copies according to all possible assignments of these clusters to this child;
4: **end for**
5: output the cheapest (m,r)-approximation $C(a,b)$.

Extended Arora's scheme

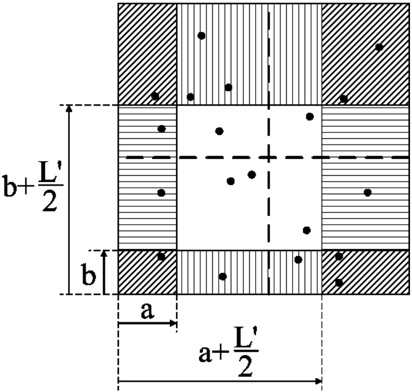


Figure: Arrangement example of clusters and shifted quadtree

Extended Arora's scheme

Theorem 19

For any fixed $\varepsilon \in (0,1)$ Algorithm 2 finds a $(1+\varepsilon)$-approximate solution for the EGTSP-GC in time of

$$2^{O(k)} k^4 (\log k)^{O(1/\varepsilon)} + O(n).$$

Corollary 20

1. For any fixed $k > 4$, Algorithm 2 is a LTAS for the EGTSP-GC.
2. For $k = O(\log n)$, Algorithm 2 is PTAS for this problem with time complexity of $O(n(\log n)^4 (\log\log n)^{O(1/\varepsilon)})$.

Outline

1. Min-k-SCCP — the Multiple Traveling Salesmen Problem
   - Problem statement
   - Complexity and Approximability
   - Metric Min-k-SCCP
   - PTAS for Euclidean Min-2-SCCP on the plane
2. Generalized Traveling Salesman Problem
   - Problem statement
   - Dynamic programming
   - Precedence constraints
   - Practical application
   - Euclidean GTSP in Grid Clusters
3. Conslusion

The case of fast growing k

Algorithm 3 Scheme based on the classic Arora's PTAS

Input: a given instance of the Euclidean GTSP on k grid clusters and a required accuracy ε.
Output: a $(1+\varepsilon)$-approximate solution.

1: consider a partition $V_1, \dots, V_k$ of the node set V of the given instance produced by the grid;
2: **for all** $(v_1, \dots, v_k) \in V_1 \times \dots \times V_k$ **do**
3:     find an $(1+\varepsilon)$-approximate solution $S(v_1, \dots, v_k)$ of the corresponding TSP instance using Arora's PTAS;
4: **end for**
5: output the cheapest solution $S(c_1, \dots, c_k)$.

The case of fast growing k

To prove the correctness of Algorithm 3, denote by t_i the number of nodes belonging to the i-th cluster. The number of ways to specify a TSP instance taking one node from each cluster is $t_1 \times \ldots \times t_k$. Maximizing this number subject to $\sum_{i=1}^{k} t_i = n$ we conclude that it does not exceed the value $(n/k)^k$ attained at point $t_i = n/k$.

The case of fast growing k

Since, for any $\varepsilon > 0$, time complexity of the Arora's PTAS for k-node instance of the Euclidean TSP is $O(k^3(\log k)^{O(\log(1/\varepsilon))})$, the time complexity of Algorithm 3 is

$$\left(\frac{n}{k}\right)^k k^3(\log k)^{O(1/\varepsilon)}. \tag{12}$$

Evidently, for any fixed k, equation (1) depends on n polynomially and Algorithm 3 is a PTAS for the Euclidean GTSP on k grid clusters.

The case of fast growing k

To prove the same claim for k depending on n, we need to restrict $k = k(n)$ such that

$$\left(\frac{n}{k}\right)^k \leq n^D \tag{13}$$

for some constant value $D > 0$. Suppose that $\frac{n-k(n)}{k(n)} \to 0$ as $n \to \infty$. Since, in this case,

$$\left(\frac{n}{k(n)}\right)^{k(n)} = \left(1 + \frac{n-k(n)}{k(n)}\right)^{k(n)} \leq e^{n-k(n)},$$

the inequality $k(n) \geq n - D\log n$ implies equation (2).

The case of fast growing k

Theorem 21

1. For any $\varepsilon > 0$, Algorithm 3 finds a $(1+\varepsilon)$-approximate solution of the Euclidean GTSP on k grid clusters in time of $n^k(\log k)^{O(1/\varepsilon)}$.
2. If $k = n - D\log n$, then time complexity of Algorithm 3 is $n^{D+3}(\log n)^{O(1/\varepsilon)}$.

References I

- M. Khachay and K. Neznakhina, *Approximation of Euclidean k-size cycle cover problem*, Croation Operational Research Review **5** (2014), no. 1, 177–188.
- M. Khachai and K. Neznakhina, *Approximability of the problem about a minimum-weight cycle cover of a graph*, Doklady Mathematics **91** (2015), no. 2, 240–245.
- M. Khachai and K. Neznakhina, *A polynomial-time approximation scheme for the Euclidean problem on a cycle cover of a graph*, Proceedings of the Steklov Institute of Mathematics **289** (2015), no. 1, 111–125.
- M. Khachay and K. Neznakhina, *Approximability of the minimum-weight k-size cycle cover problem*, Journal of Global Optimization **66** (2016), no. 1, 65–82.

References II

K.Neznakhina, *A PTAS for the Min-k-SCCP in a Euclidean space of arbitrary fixed dimension*, Proceedings of the Steklov Institute of Mathematics **295** (2016), suppl. 1.

M. Khachay and H. Zaytseva, *Polynomial Time Approximation Scheme for Single-Depot Euclidean Capacitated Vehicle Routing Problem*, Combinatorial optimization and applications: 9th international conference, COCOA 2015, Houston, TX, USA, December 18-20, 2015, LNCS 9486, 178–190.

A. Chentsov, M. Khachay, and D. Khachay, *An Exact Algorithm with Linear Complexity for a Problem of Visiting Megalopolises*, Proceedings of the Steklov Institute of Mathematics **295** (2016), suppl. 1.

A. Chentsov, M. Khachay, and D. Khachay, *Linear time algorithm for Precedence Constrained Asymmetric Generalized Traveling Salesman Problem*, IFAC-PapersOnline, **49**, 12 (2016), 651–655.

References III

Michael Khachay and Roman Dubinin. *PTAS for the Euclidean Capacitated Vehicle Routing Problem in* R^d. Discrete Optimization and Operations Research, DOOR 2016, Vladivostok, September 19-23, 2016, LNCS 9869, 193–205.

Thank you for your attention!