A PTAS for Capacitated Vehicle Routing on Trees

Claire Mathieu

CNRS Paris, France

Hang Zhou

École Polytechnique, France

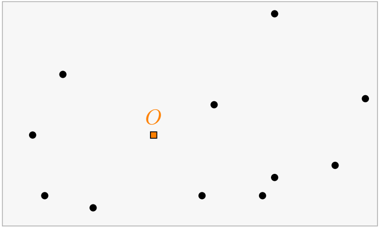
ICALP 2022

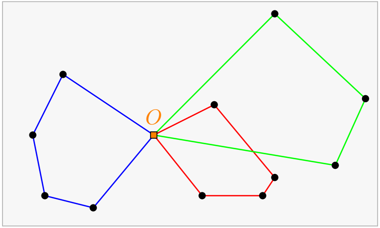
Capacitated vehicle routing

Input:

- **depot** O
- n **terminals**
- **tour capacity** k

Minimize total length of tours





Fundamental problem in operations research

e.g., more than 4000 articles on vehicle routing on DBLP

Capacitated vehicle routing

1985	HR
1990	AG
1997	AKTT
2006	BDO
2010	ACL
2010	DM
2017	BKS
2018	BKS
2019	BKS
2020	CFKL
2021	BTV
2022	JS

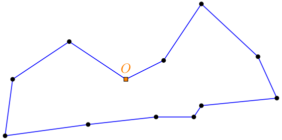
well-studied problem

- **general metric**
- **Euclidean space**
- **planar graph**
- **graph of bounded highway dimension**
- **graph of bounded treewidth**

Polynomial time approximation scheme (PTAS) only for small capacity k.

Iterated tour partitioning [Haimovich and Rinnooy Kan]

1. Compute a traveling salesman tour



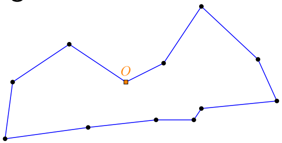
2. Partition the tour into segments of at most k terminals each

3. Connect the endpoints of each segment to the depot

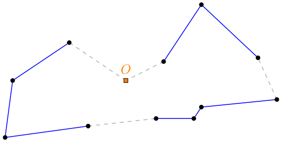
Approximation ratio $\approx 1 + \alpha_{\text{TSP}} \geq 2$

Iterated tour partitioning [Haimovich and Rinnooy Kan]

1. Compute a traveling salesman tour



2. Partition the tour into segments of at most k terminals each

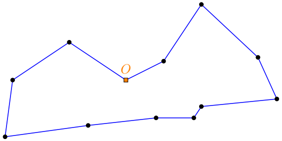


3. Connect the endpoints of each segment to the depot

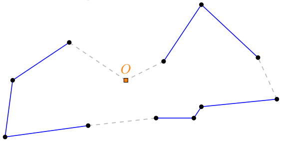
Approximation ratio $\approx 1 + \alpha_{\text{TSP}} \geq 2$

Iterated tour partitioning [Haimovich and Rinnooy Kan]

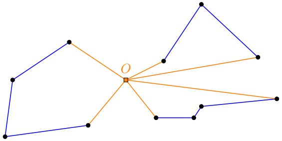
1. Compute a traveling salesman tour



2. Partition the tour into segments of at most k terminals each



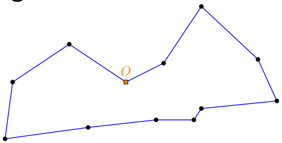
3. Connect the endpoints of each segment to the depot



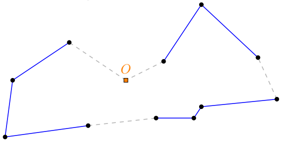
Approximation ratio $\approx 1 + \alpha_{\text{TSP}} \geq 2$

Iterated tour partitioning [Haimovich and Rinnooy Kan]

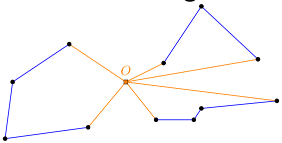
1. Compute a traveling salesman tour



2. Partition the tour into segments of at most k terminals each



3. Connect the endpoints of each segment to the depot



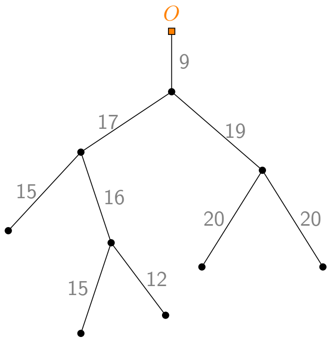
Approximation ratio $\approx 1 + \alpha_{\text{TSP}} \geq 2$

Capacitated vehicle routing on trees

NP-hard [LLM 1991]

Approximation algorithms:

- 1.5-approximation [HK 1998]
- 1.351-approximation [AKK 2001]
- 1.333-approximation [Bec 2018]
- bicriteria PTAS [BP 2019]
- QPTAS [JS 2022]



Our Result

PTAS for Capacitated Vehicle Routing on Trees

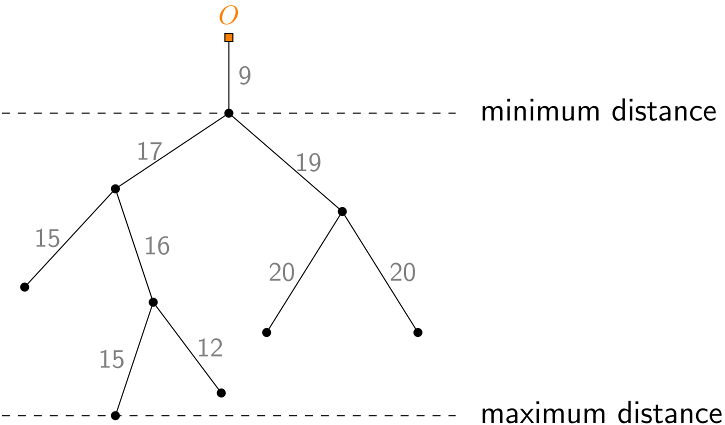
Note: First PTAS for general capacity k.

Capacitated vehicle routing on trees

Jayaprakash and Salavatipour [SODA 2022]:

"*it is not clear if it (the QPTAS)*
can be turned into a PTAS
without significant new ideas."

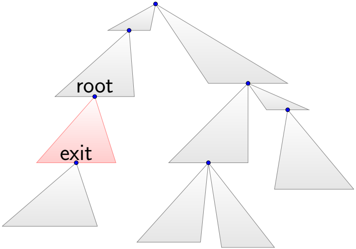
Preprocessing: bounded distance property



Can assume: $\dfrac{\textbf{minimum distance}}{\textbf{maximum distance}} > \epsilon.$

Decomposing the tree into components

Inspiration: clusters decomposition [Becker and Paul]



each **component** has:

- $\approx k/\epsilon$ terminals
- 1 root vertex
- ≤ 1 exit vertex

Structure Theorem

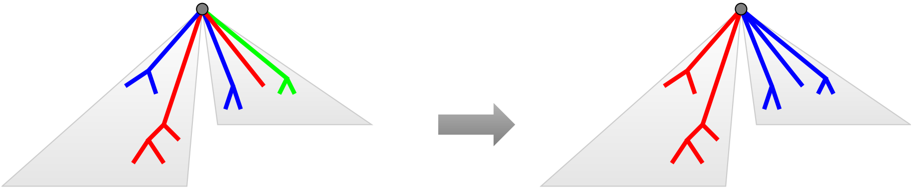
There is a near-optimal solution such that each subtour in a component visits either 0 or at least $\epsilon \cdot k$ terminals.

Proof of the Structure Theorem (1/4)

Definition: a subtour in a component is **small** if it visits more than 0 but less than $\epsilon \cdot k$ terminals.

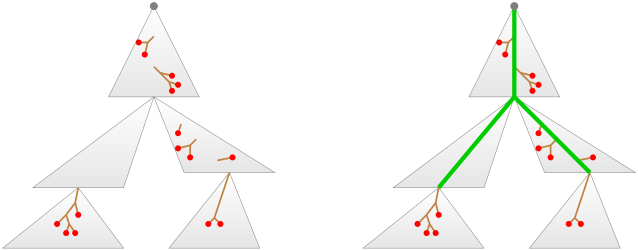
Elimination of small subtours

1. Detach small subtours
2. Combine small subtours within components
3. Reassign combined subtours [Becker and Paul]



Difficulty: **after reassignment some tours may exceed capacity**

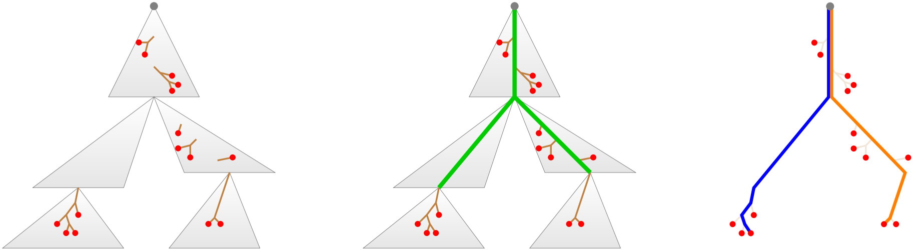
Proof of the Structure Theorem (2/4)



4. **Remove some subtours**
5. **Include spine subtours**

brown & green: **traveling salesman tour** covering the red terminals

Proof of the Structure Theorem (3/4)

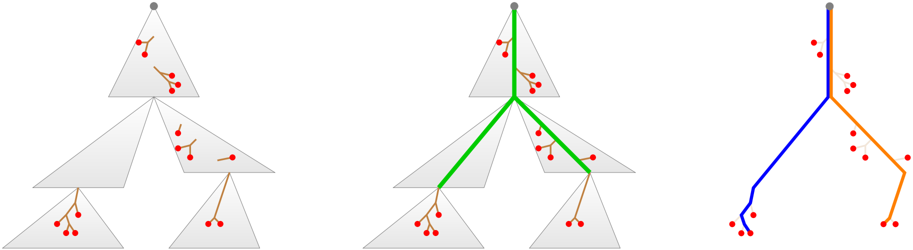


However, the traveling salesman tour may exceed capacity.

Our approach: **iterated tour partitioning**

6. Add connections to the depot (**blue** & **orange**)

Proof of the Structure Theorem (4/4)



Extra cost is negligible:

- **green**: properties of components
- **blue & orange**: bounded distance property

Structure Theorem

There is a near-optimal solution such that each subtour in a component visits either 0 or at least $\epsilon \cdot k$ terminals.

First attempt for the dynamic program

1. Computing solutions inside each component: polynomial time
2. Combining solutions from different components bottom-up:

   exponential time

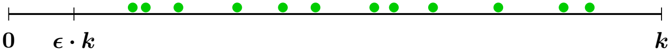
Q: How to improve the running time?

A: Adaptive rounding to reduce the number of subtour demands.

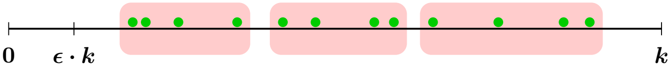
Adaptive rounding [Jayaprakash and Salavatipour]

At each vertex:

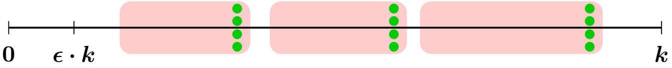
1. **Sort** subtour demands



2. **Make groups** of equal cardinality



3. **Round up** to maximum demand in group



Theorem [Jayaprakash and Salavatipour]

There is a near-optimal solution in which the subtour demands can be rounded up to **polylogarithmic** many values.

$\Longrightarrow$ **QPTAS**

Our Theorem

There is a near-optimal solution in which the subtour demands can be rounded up to **constant** many values.

$\Longrightarrow$ **PTAS**

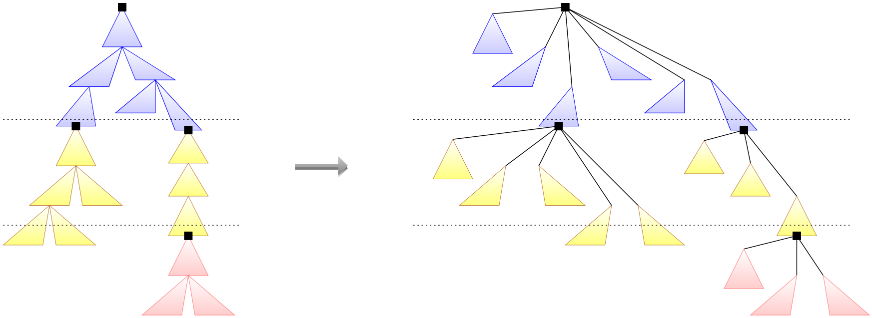
How do we bound the extra cost?

- Structure Theorem
- bounded distance property
- bounded height of components

Bounded height of components

Transform the tree of components so that:

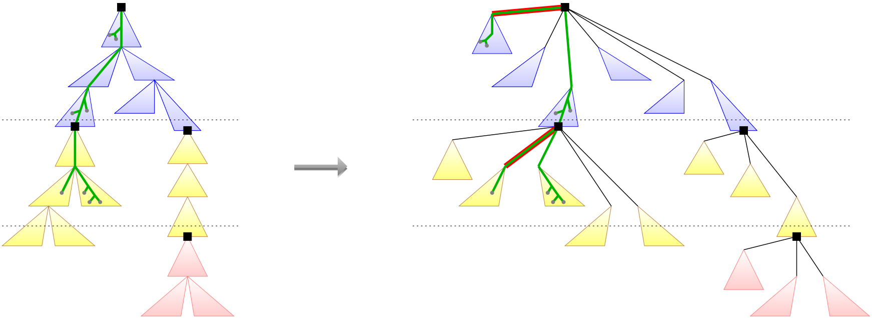
each leaf-root path traverses a bounded number of components



Bounded height of components

Transform the tree of components so that:

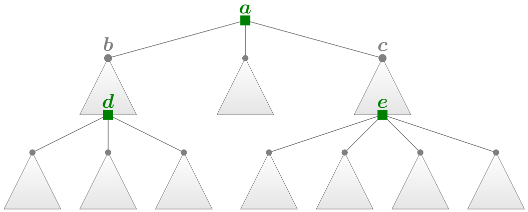
each leaf-root path traverses a bounded number of components



Extra cost of the tour is negligible:

- Structure Theorem
- bounded distance property

Dynamic programming and adaptive rounding



Dynamic program order of computation:

1. each component
2. subtrees rooted at d and e by **adaptive rounding**
3. subtrees rooted at b and c
4. subtree rooted at a by **adaptive rounding**

Polynomial time to obtain a $(1 + \epsilon)$-approximate solution

Summary

1. Preprocessing $\Longrightarrow$ **Bounded distance**
2. Decomposing the tree into components $\Longrightarrow$ **Structure Theorem**
3. Transforming the tree $\Longrightarrow$ **Bounded height of components**
4. Adaptive Rounding $\Longrightarrow$ **Constant many subtour demands**
5. Dynamic programming $\Longrightarrow$ $(1+\epsilon)$**-approximate solution**

Our Result

PTAS for Capacitated Vehicle Routing on Trees

Thank you!