2024 Optimization Grand Challenge

Effective Bundle Enumeration and Tree-structured Improvement

Team P R O

Members Park Yebin

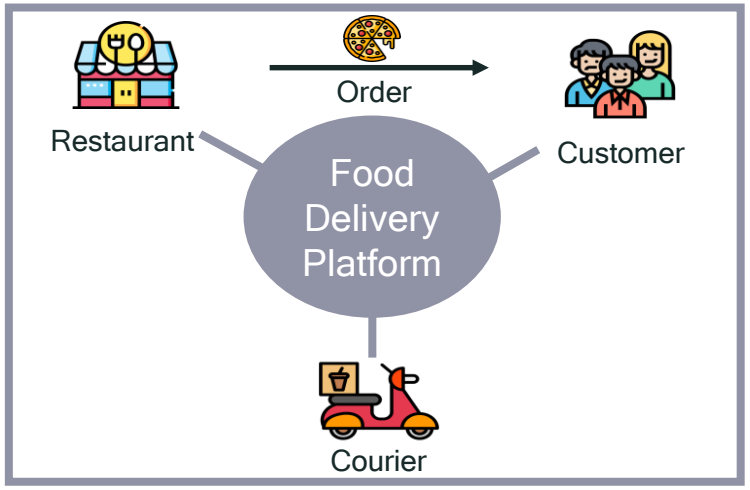
Roh Youngjoo

Oh Seyoung

October 24, 2024

Motivation

Backgrounds and Theoretical Basis



- **Order**
  - pickup, delivery
  - time window
  - volume
- **Courier**
  - capacity
  - speed
  - availability

Input → Food Delivery Platform → Output

What courier delivers
Which orders via
Which route

Bundle

Multiple Pickup and Delivery Problem

- NP-hard
- Limitation of Column Generation on Set partitioning Formulation

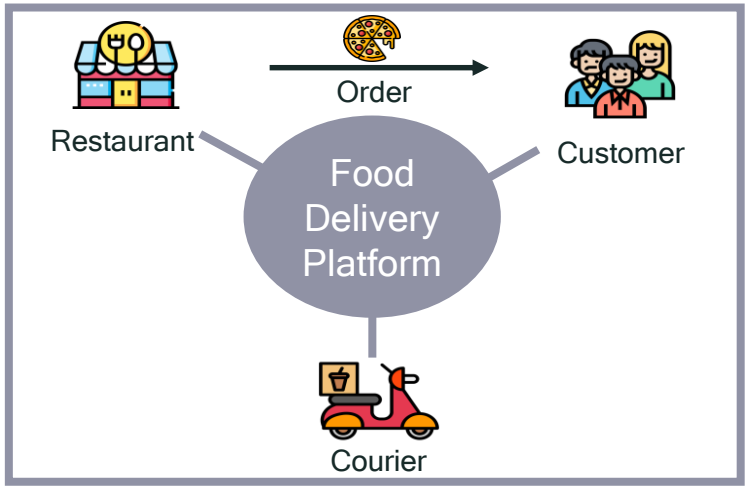
⇩

Practical and general solution approach

for real-world scale instances by leveraging structural advantages

Motivation

Backgrounds and Theoretical Basis



- **Order**
  - pickup, delivery
  - time window
  - volume
- **Courier**
  - capacity
  - speed
  - availability

Input → Food Delivery Platform → Output

What courier delivers
Which orders via
Which route

Bundle

Multiple Pickup and Delivery Problem

- NP-hard
- Limitation of Column Generation on Set partitioning Formulation

⇩

Practical and general solution approach

for real-world scale instances by leveraging structural advantages

$$\min \quad \sum_{r\in\Omega} c_r \mathbf{x}_r$$

$$\text{s.t.} \quad \sum_{r\in\Omega} a_{ir}\mathbf{x}_r = 1 \quad \forall i$$

$$\sum_{r\in\Omega} \mathbf{x}_r = |V|$$

$$\mathbf{x}_r \in \mathbb{B} \quad \forall r \in \Omega$$

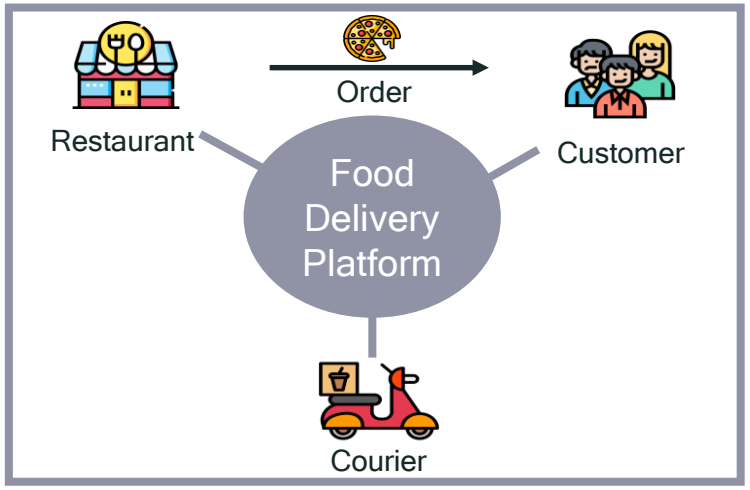
perfect formulation

if bundles are subtrees from a specific tree

Notation	n :	number of requests
	V :	set of vehicles
	Ω :	set of feasible routes
	c_r :	cost of route $r \in \Omega$
	a_{ir} :	binary constant, 1 if route $r \in \Omega$ includes request i
Variables	$\mathbf{x}_r$:	binary variable, 1 if route $r \in \Omega$ includes request i

Motivation

Backgrounds and Theoretical Basis



- **Order**
  - pickup, delivery
  - time window
  - volume
- **Courier**
  - capacity
  - speed
  - availability

Input → Food Delivery Platform → Output

What courier delivers
Which orders via
Which route

Bundle

Multiple Pickup and Delivery Problem

- NP-hard
- Limitation of Column Generation on Set partitioning Formulation

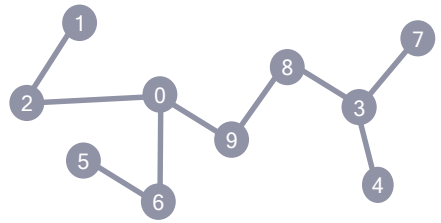
⇩

Practical and general solution approach

for real-world scale instances by leveraging structural advantages

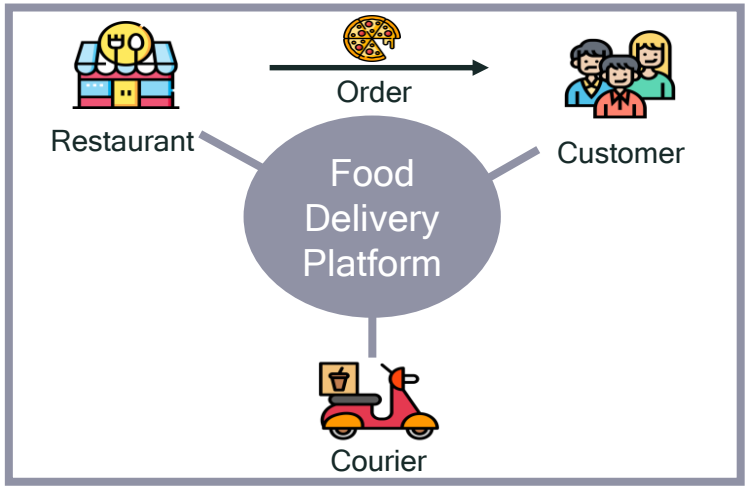
Subtrees on Tree

- Tree : an undirected, connected, acyclic graph
- Subtree: subgraph of the tree



Motivation

Backgrounds and Theoretical Basis



- **Order**
  - pickup, delivery
  - time window
  - volume
- **Courier**
  - capacity
  - speed
  - availability

Input → Food Delivery Platform → Output

What courier delivers
Which orders via
Which route

Bundle

Multiple Pickup and Delivery Problem

- NP-hard
- Limitation of Column Generation on Set partitioning Formulation

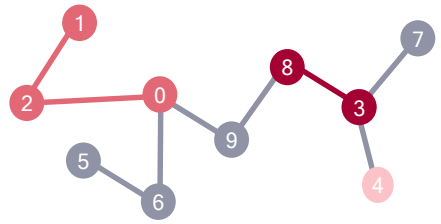
⇩

Practical and general solution approach

for real-world scale instances by leveraging structural advantages

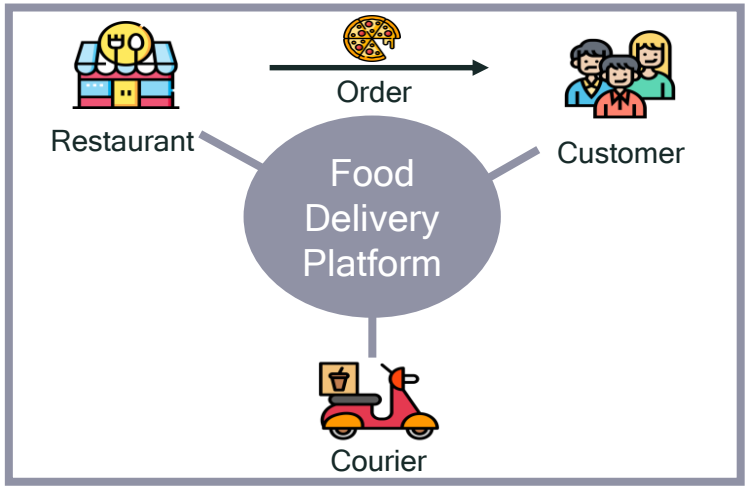
Subtrees on Tree

- Tree : an undirected, connected, acyclic graph
- Subtree: subgraph of the tree



Motivation

Backgrounds and Theoretical Basis



- **Order**
  - pickup, delivery
  - time window
  - volume
- **Courier**
  - capacity
  - speed
  - availability

Input → Food Delivery Platform → Output

What courier delivers
Which orders via
Which route

Bundle

Multiple Pickup and Delivery Problem

- NP-hard
- Limitation of Column Generation on Set partitioning Formulation

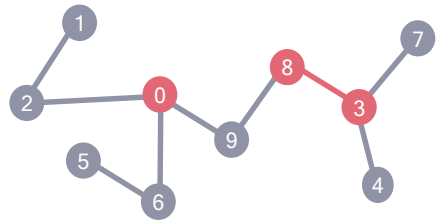
⇩

Practical and general solution approach

for real-world scale instances by leveraging structural advantages

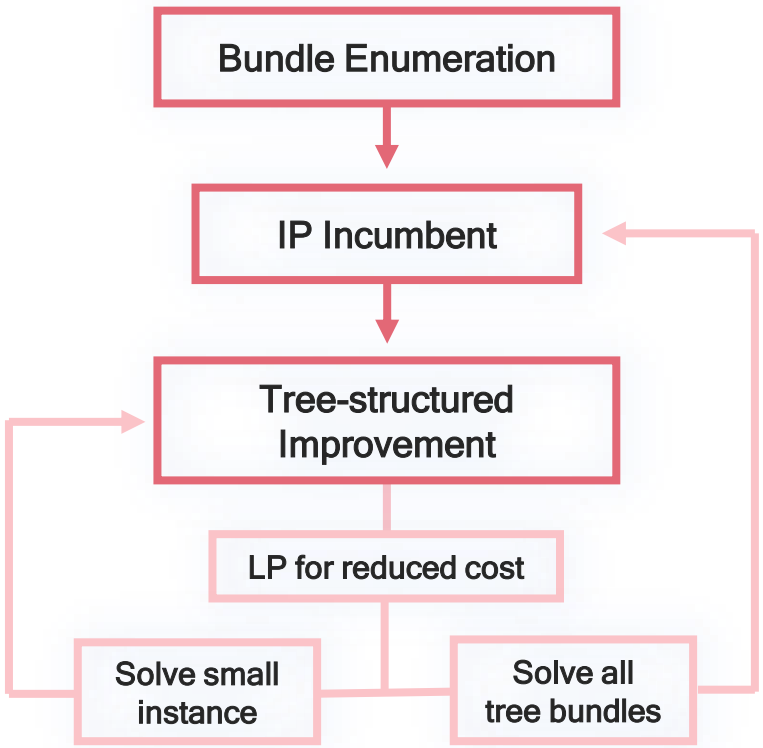
Subtrees on Tree

- Tree : an undirected, connected, acyclic graph
- Subtree: subgraph of the tree



Algorithm Framework

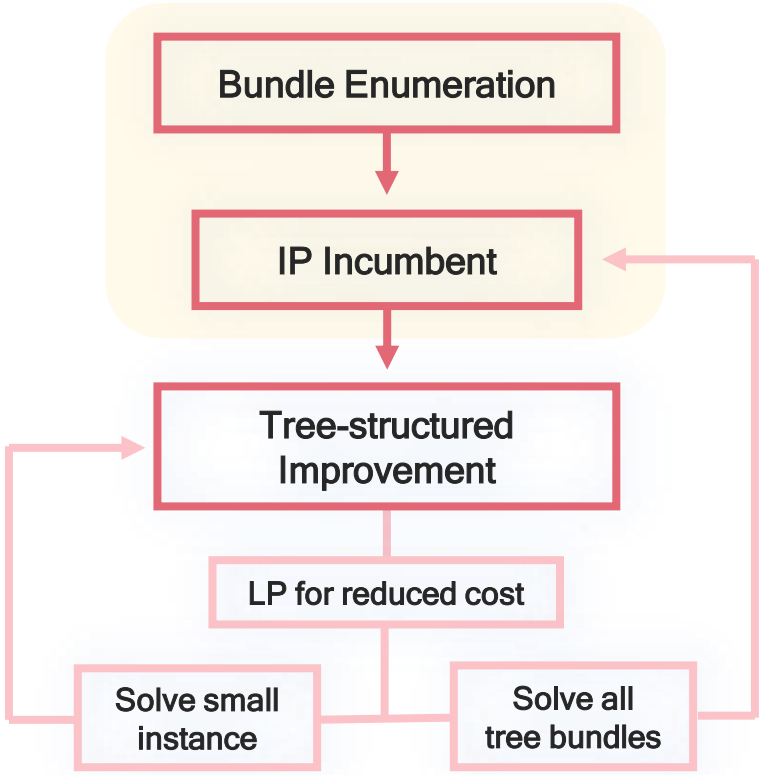
Logics and Details



< Algorithm Flowchart >

Algorithm Framework

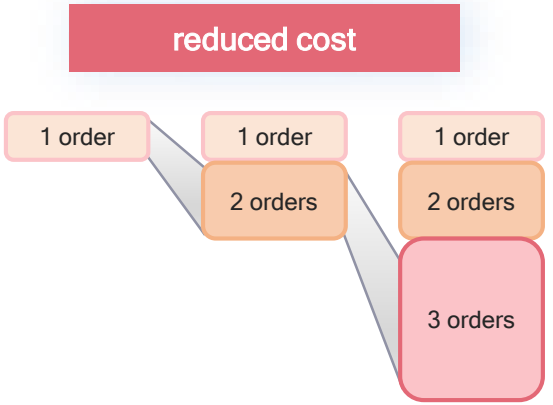
Logics and Details



< Algorithm Flowchart >

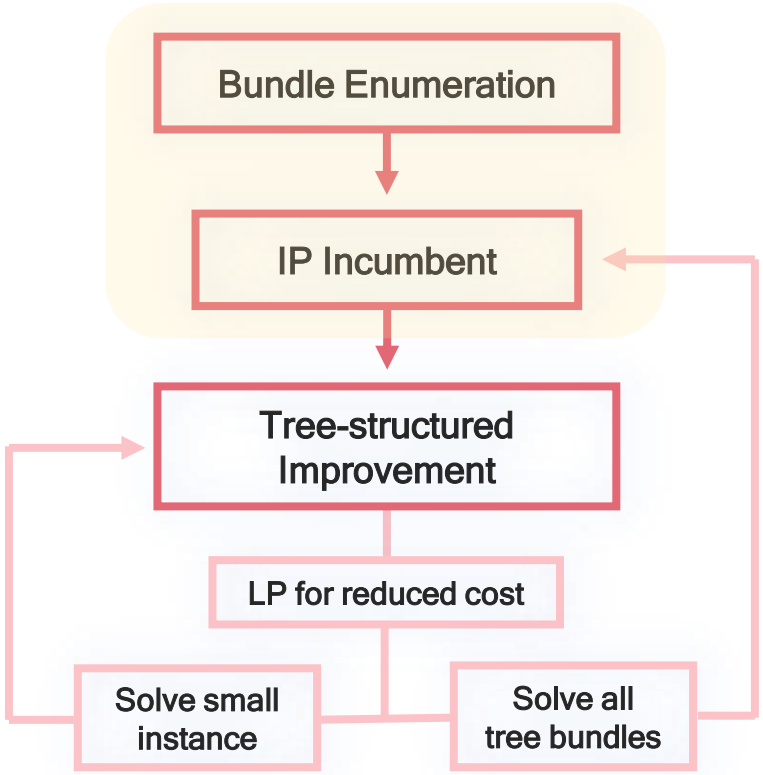
1. Bundle Enumeration

- **Bundle**: [What courier delivers, Which orders via, Which route]
- **Profitable Bundle Enumeration**



Algorithm Framework

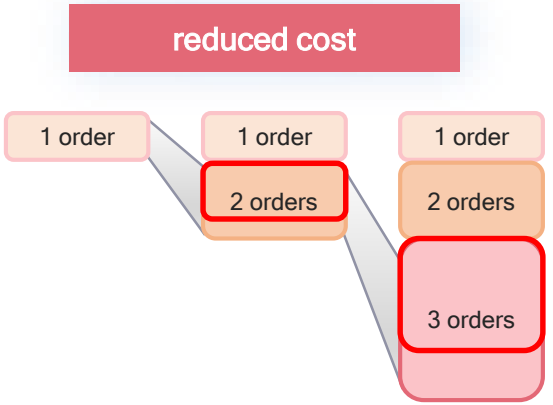
Logics and Details



< Algorithm Flowchart >

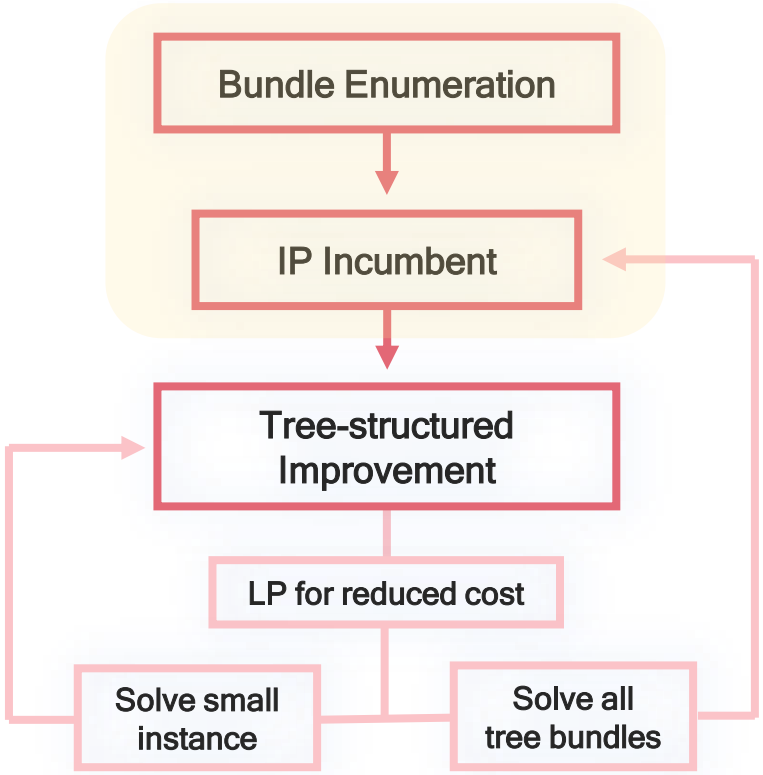
1. Bundle Enumeration

- **Bundle**: [What courier delivers, Which orders via, Which route]
- **Profitable Bundle Enumeration**



Algorithm Framework

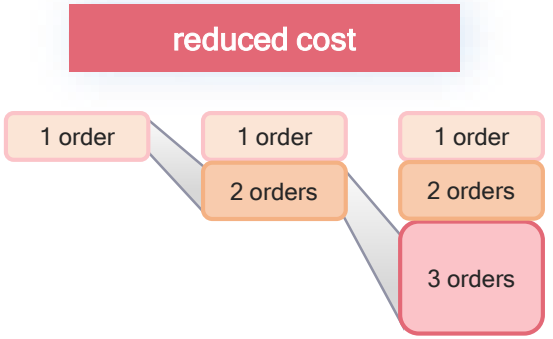
Logics and Details

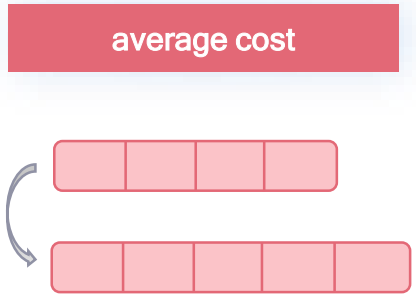


< Algorithm Flowchart >

1. Bundle Enumeration

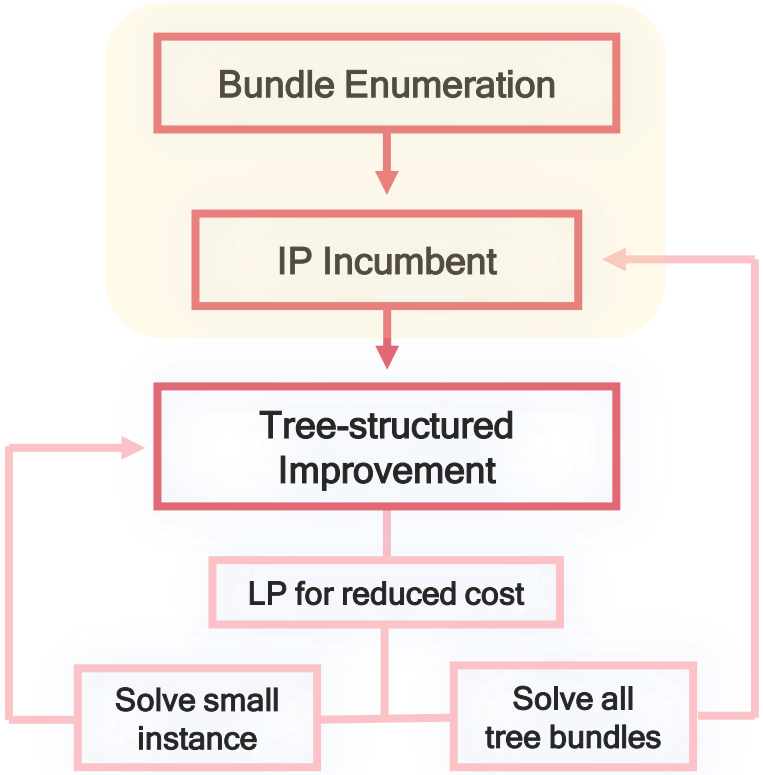
- **Bundle**: [What courier delivers, Which orders via, Which route]
- **Profitable Bundle Enumeration**





Algorithm Framework

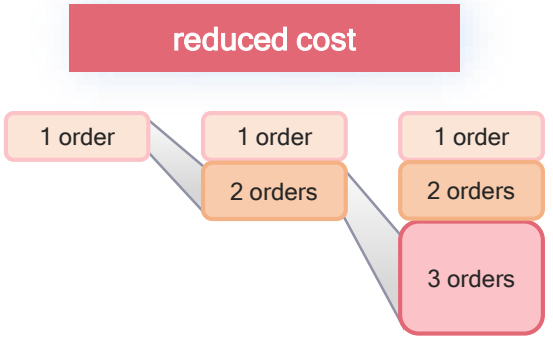
Logics and Details

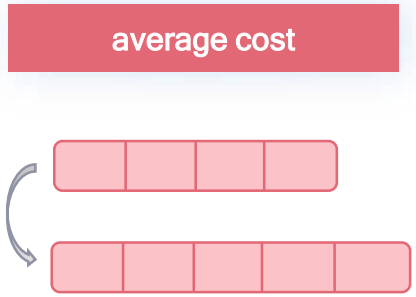


< Algorithm Flowchart >

1. Bundle Enumeration

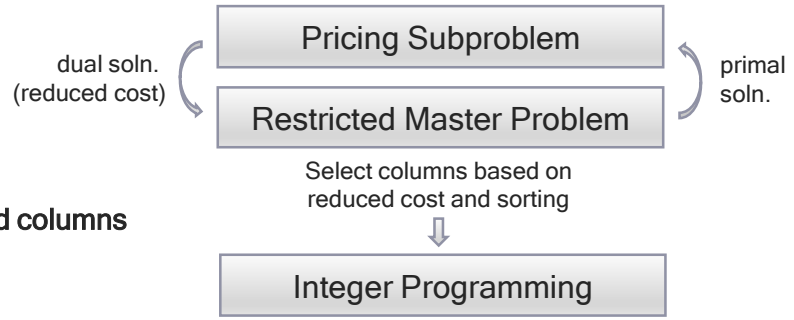
- **Bundle**: [What courier delivers, Which orders via, Which route]
- **Profitable Bundle Enumeration**





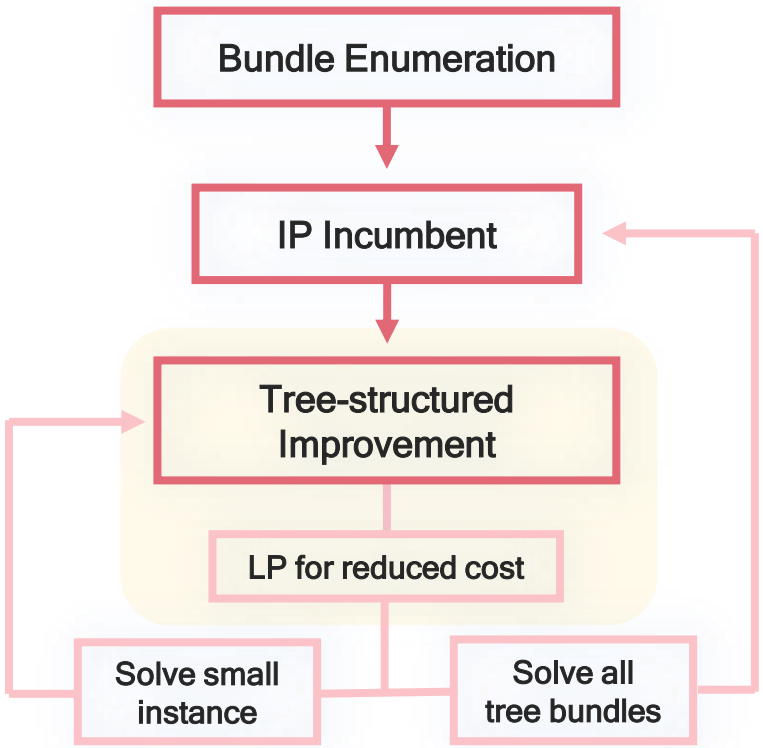
2. IP Incumbent

- **Column Selection by LP**
- **Restricted IP using selected columns**



Algorithm Framework

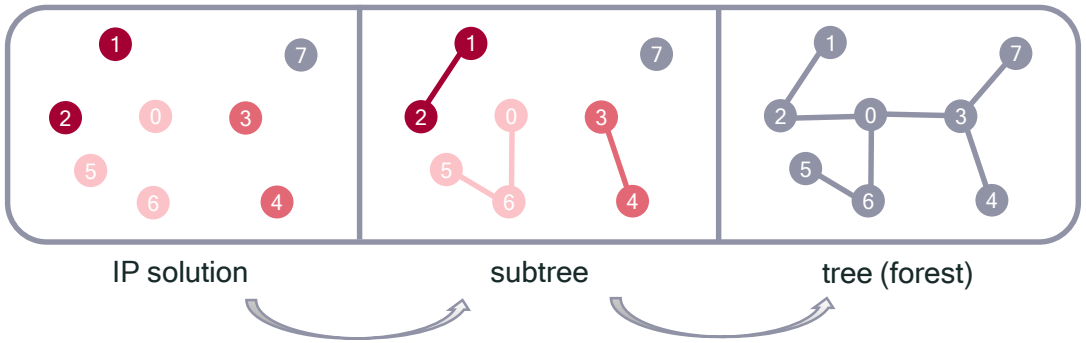
Logics and Details



< Algorithm Flowchart >

3. Tree-structured Improvement

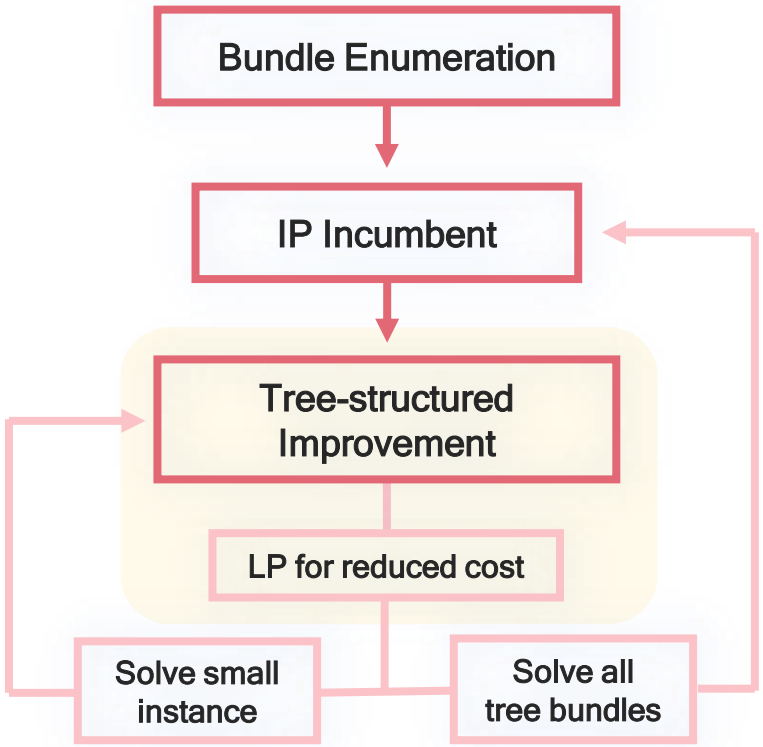
3-1. Tree Construction from IP Solution



1) Deadline slack = deadline - service time - arrival time

Algorithm Framework

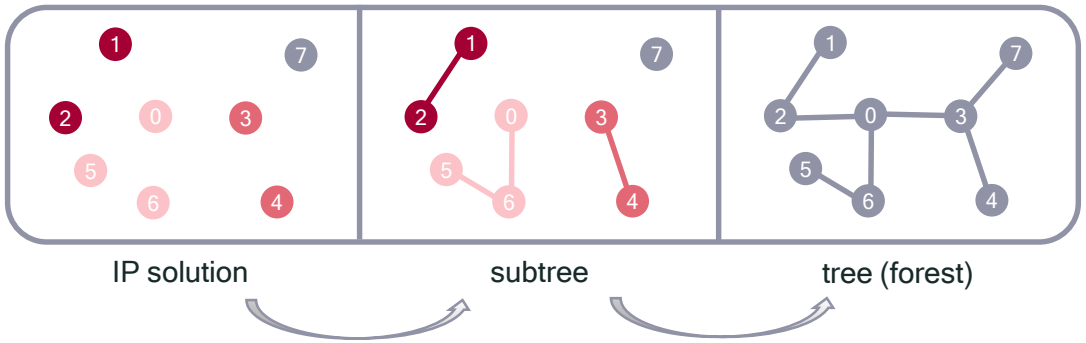
Logics and Details



< Algorithm Flowchart >

3. Tree-structured Improvement

3-1. Tree Construction from IP Solution

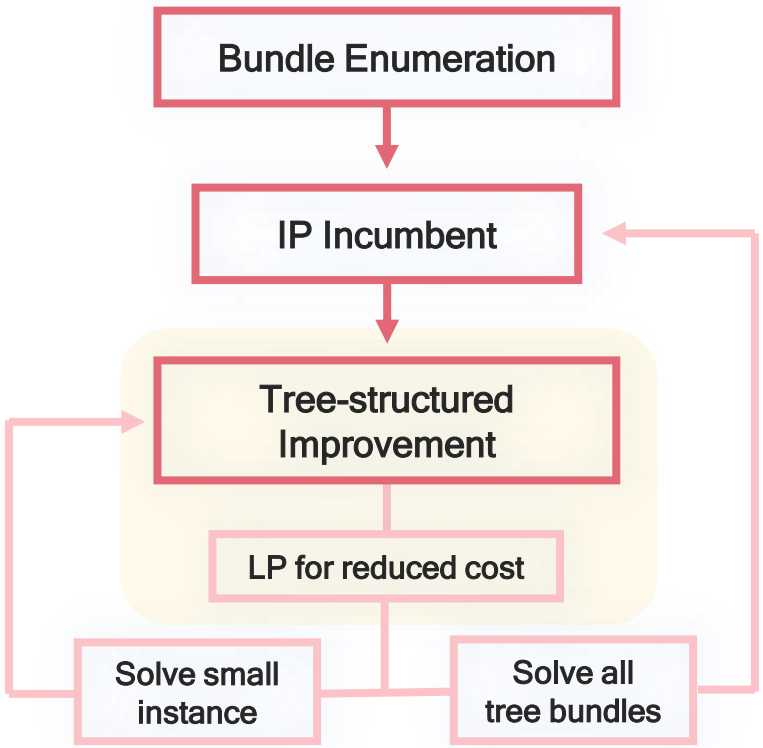


3-2. LP for reduced cost and tree improvement

1) Deadline slack = deadline - service time - arrival time

Algorithm Framework

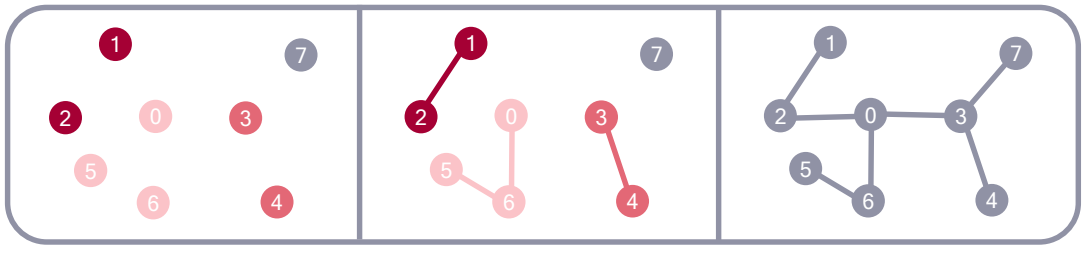
Logics and Details



< Algorithm Flowchart >

3. Tree-structured Improvement

3-1. Tree Construction from IP Solution



Alt 1. Delivery sequence
Alt 2. Pickup sequence
Alt 3. Deadline slack

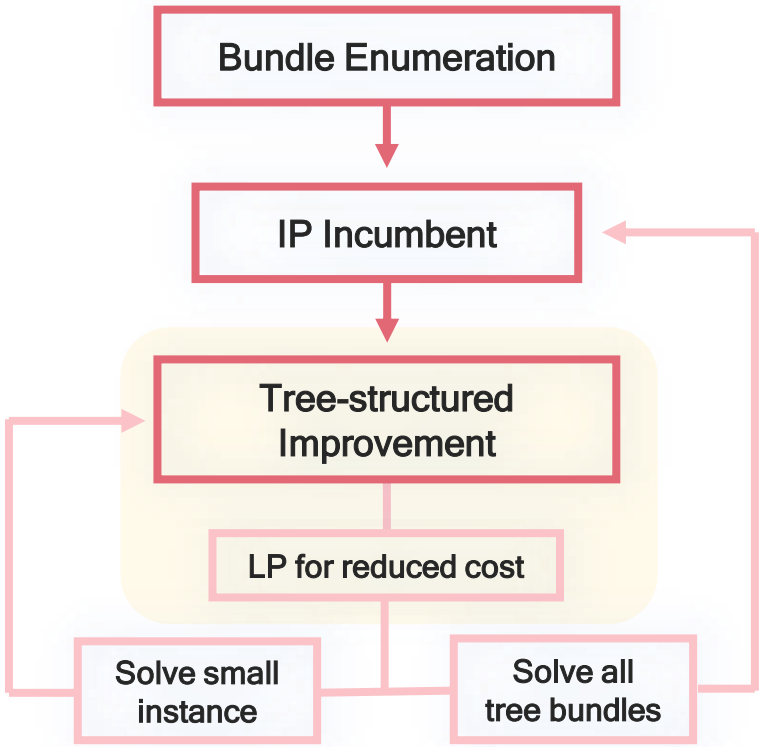
Deadline slack[1)]

3-2. LP for reduced cost and tree improvement

1) Deadline slack = deadline - service time - arrival time

Algorithm Framework

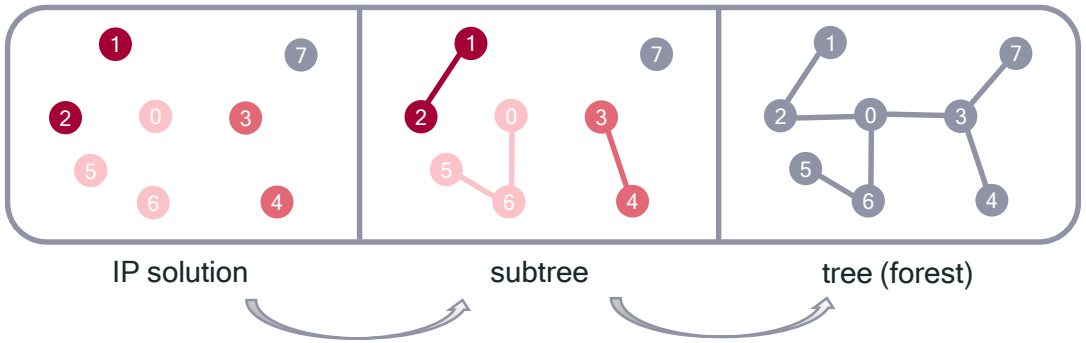
Logics and Details



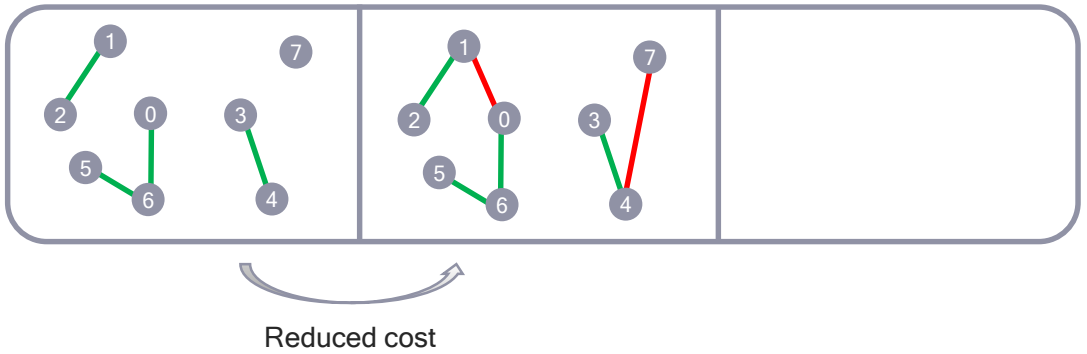
< Algorithm Flowchart >

3. Tree-structured Improvement

3-1. Tree Construction from IP Solution



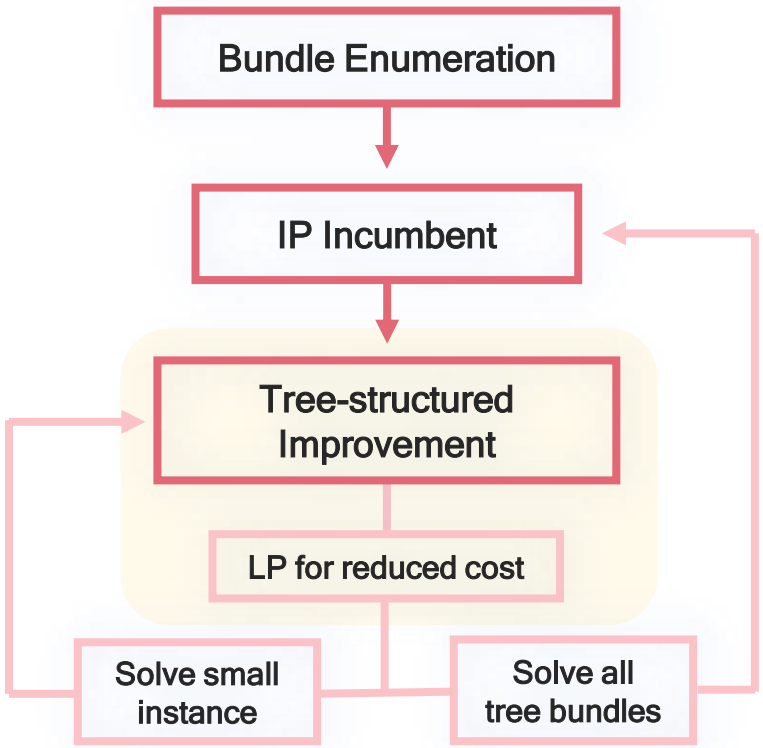
3-2. LP for reduced cost and tree improvement



1) Deadline slack = deadline - service time - arrival time

Algorithm Framework

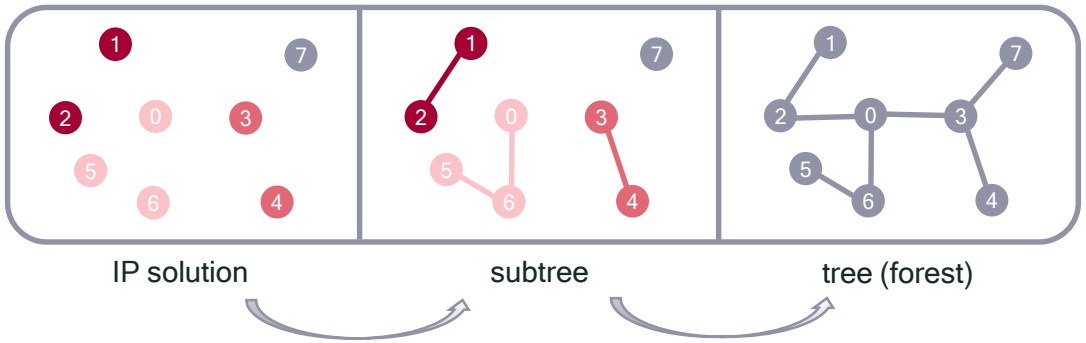
Logics and Details



< Algorithm Flowchart >

3. Tree-structured Improvement

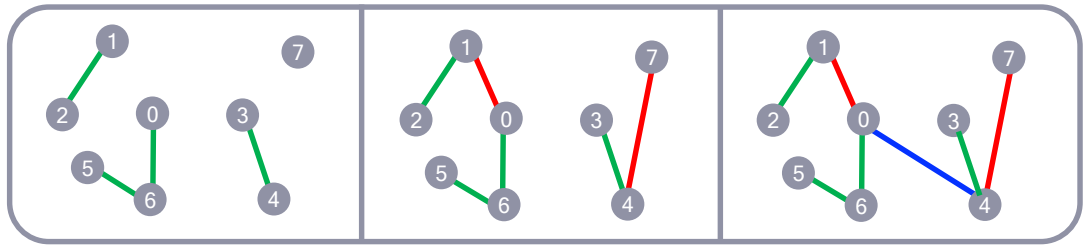
3-1. Tree Construction from IP Solution



Alt 1. Delivery sequence
Alt 2. Pickup sequence
Alt 3. Deadline slack

Deadline slack[1)]

3-2. LP for reduced cost and tree improvement



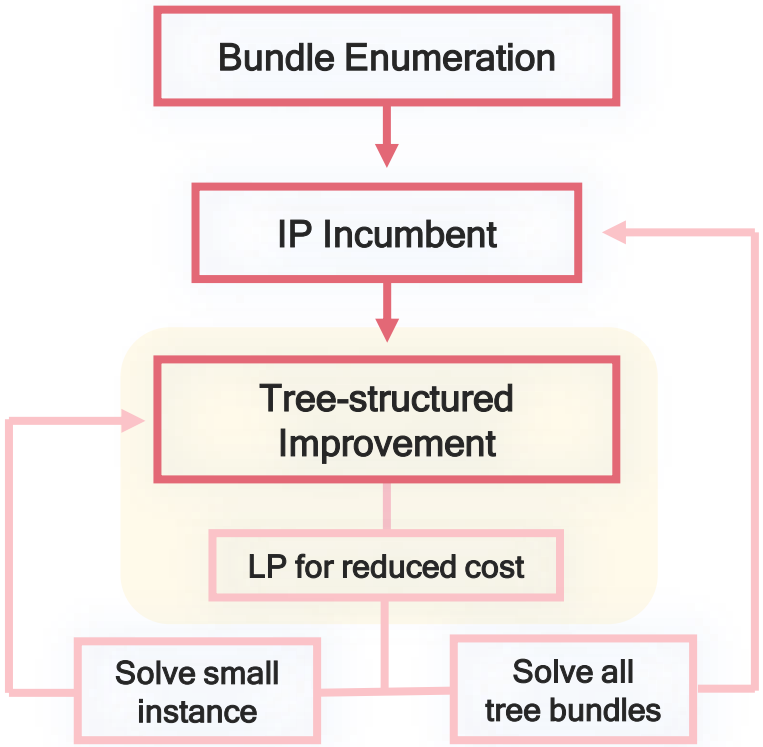
Reduced cost

Deadline slack

1) Deadline slack = deadline - service time - arrival time

Algorithm Framework

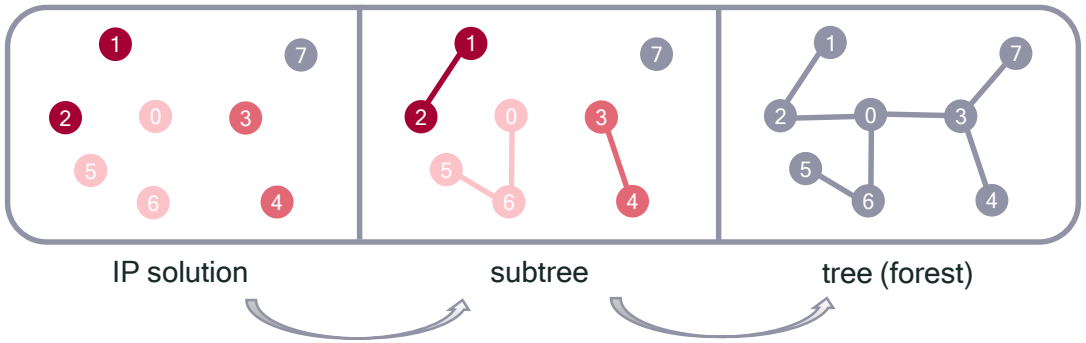
Logics and Details



< Algorithm Flowchart >

3. Tree-structured Improvement

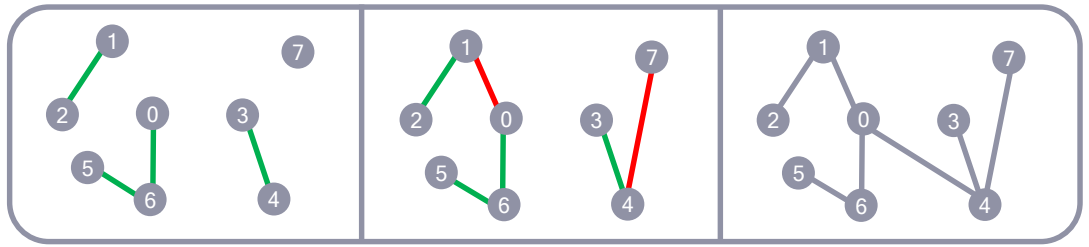
3-1. Tree Construction from IP Solution



Alt 1. Delivery sequence
Alt 2. Pickup sequence
Alt 3. Deadline slack

Deadline slack[1)]

3-2. LP for reduced cost and tree improvement



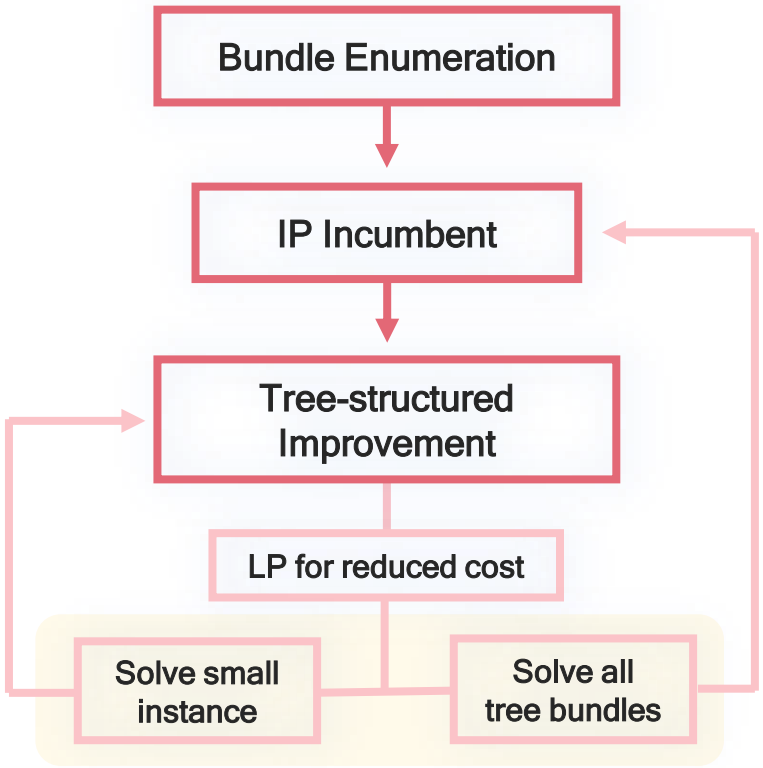
Reduced cost

Deadline slack

1) Deadline slack = deadline - service time - arrival time

Algorithm Framework

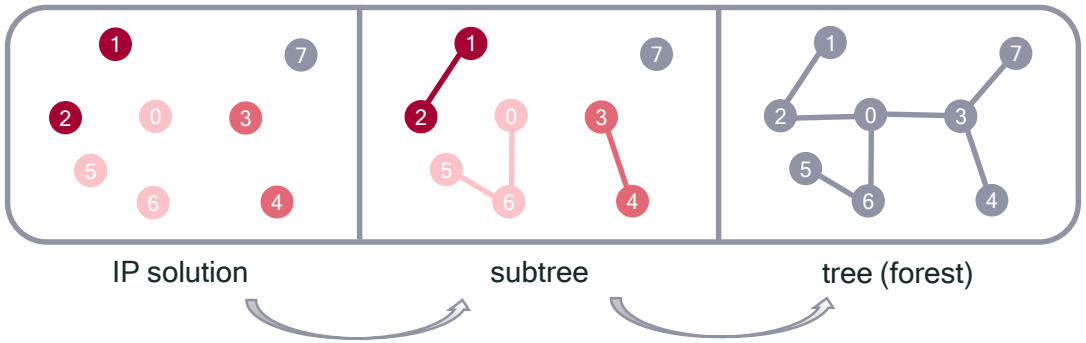
Logics and Details



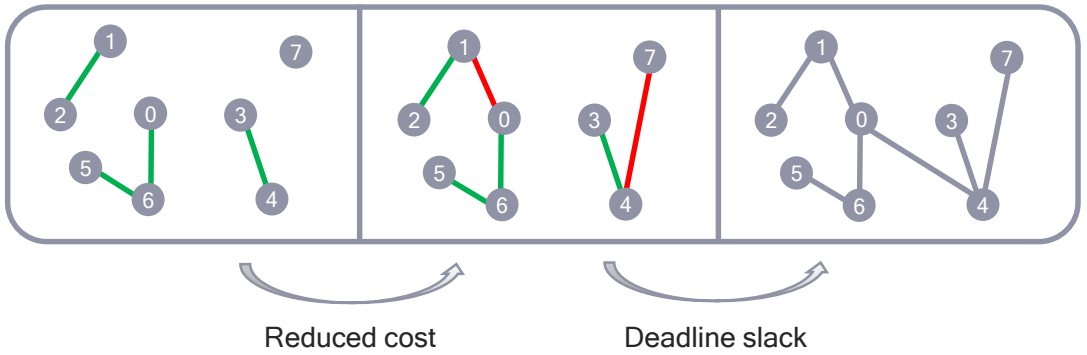
< Algorithm Flowchart >

3. Tree-structured Improvement

3-1. Tree Construction from IP Solution



3-2. LP for reduced cost and tree improvement



1) Deadline slack = deadline - service time - arrival time

Computational Implementations

Skills and Environments

- **Implementations**

Languages	**C**	for implementing all core algorithms
	Python	for handling data input and solution output
Packages	**FICO Xpress**	for optimization tasks
	pthread	for multithreading and parallel processing in bundle enumeration
	hashmap	for efficient data management using hash maps (open source)

- **Computational Environments**

▸Intel Core i7-4770S CPU processor (3.10GHz with 16GB RAM)

- **Instances**

▸stage 1, 2, 3 problems from OGC2024 & TEST instances from OGC2024

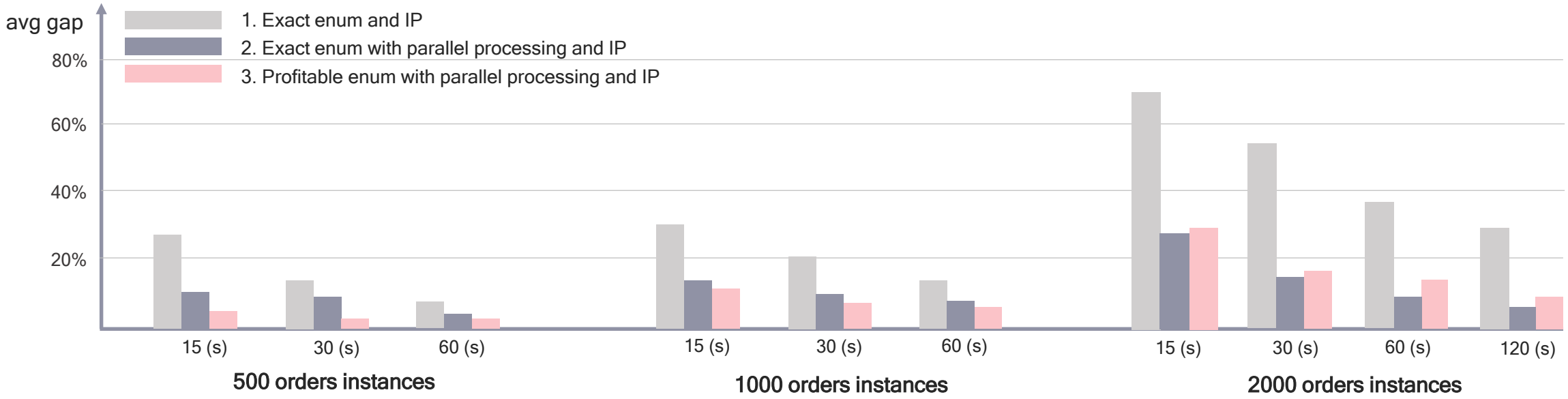
- 500 orders instances : stage 2-1, stage 2-3, stage 2-5, TEST 2-4, TEST 2-5
- 1000 orders instances : stage 2-2, stage 2-4, stage 2-6, TEST 2-1, TEST 2-2
- 2000 orders instances : stage 3-1, stage 3-2, stage 3-3, TEST 3-1, TEST 3-2

Key Strengths

Algorithmic Advantages and Related Results

Bundle Enumeration

- More Time Gain with Less Solution Quality Loss

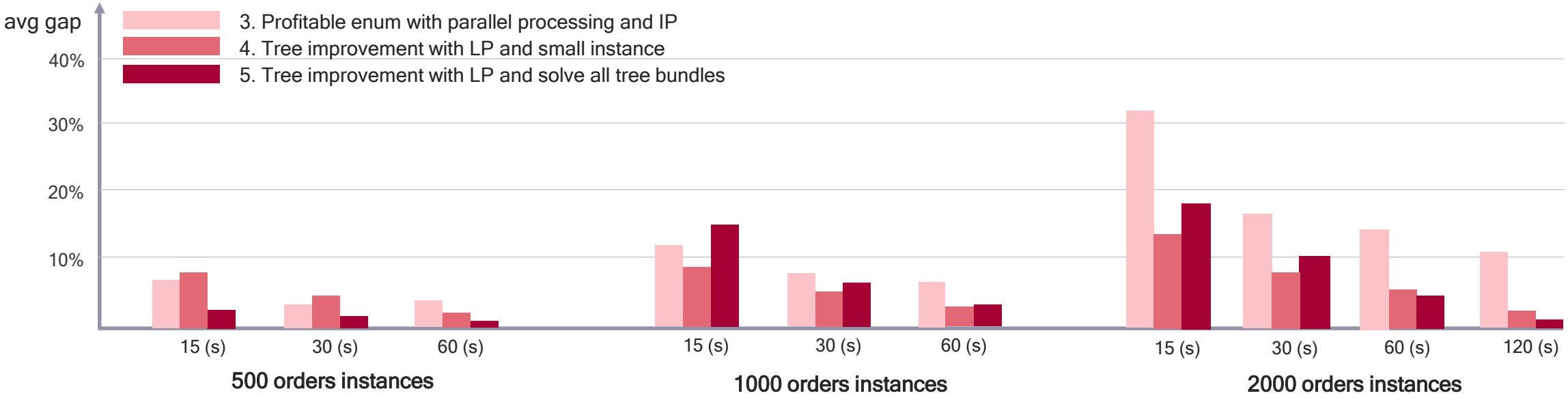


where avg gap = avg($\frac{obj - best\ obj}{best\ obj}$) $\times 100$ (%) with best obj from 300s time limit

Key Strengths

Algorithmic Advantages and Related Results

Bundle Enumeration	• More Time Gain with Less Solution Quality Loss
Tree Improvement	• Longer bundles observed • Monotonic improvement • Near similarity between IP and LP due to tree structure

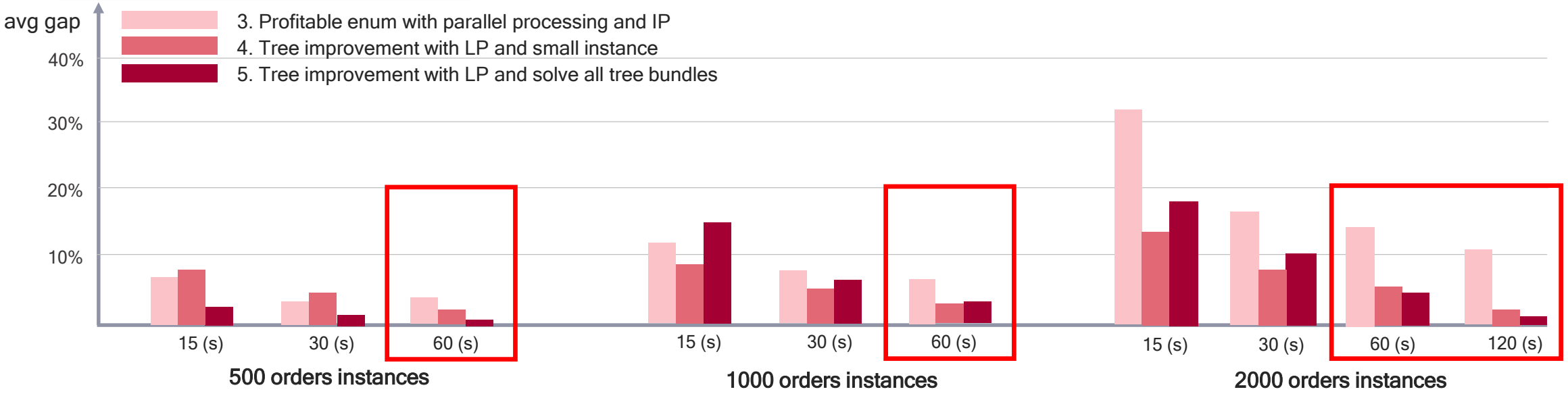


where avg gap = avg($\frac{obj - best\ obj}{best\ obj}$) $\times 100$ (%) with best obj from 300s time limit

Key Strengths

Algorithmic Advantages and Related Results

Bundle Enumeration	• More Time Gain with Less Solution Quality Loss
Tree Improvement	• Longer bundles observed • Monotonic improvement • Near similarity between IP and LP due to tree structure



where avg gap = avg($\frac{obj - best\ obj}{best\ obj}$) $\times 100$ (%) with best obj from 300s time limit

Concluding Remark

Future Works and Participation Experience

Future Works

- **Independent Stage Enhancements**
  - ▸Targeted improvements for each component
    (Bundle Enumeration / Tree Improvement)

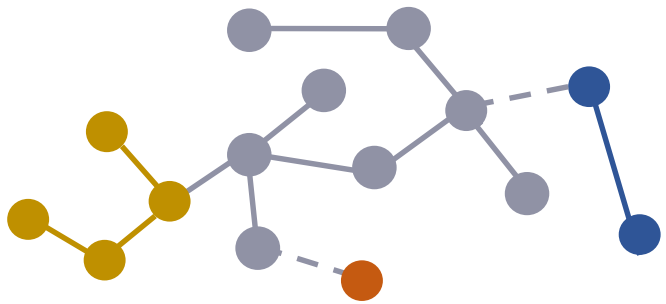
Concluding Remark

Future Works and Participation Experience

Future Works

- **Independent Stage Enhancements**
  - ▸Targeted improvements for each component
    (Bundle Enumeration / Tree Improvement)

- **Dynamic Order Handling : Tree Reconstruction**
  - ▸Incremental algorithm suitable for real-time order changes



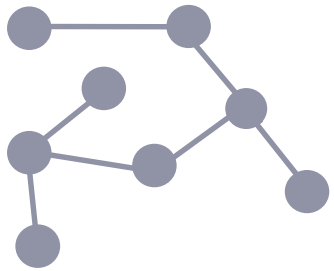
Concluding Remark

Future Works and Participation Experience

Future Works

- **Independent Stage Enhancements**
  - ▸Targeted improvements for each component
    (Bundle Enumeration / Tree Improvement)

- **Dynamic Order Handling : Tree Reconstruction**
  - ▸Incremental algorithm suitable for real-time order changes



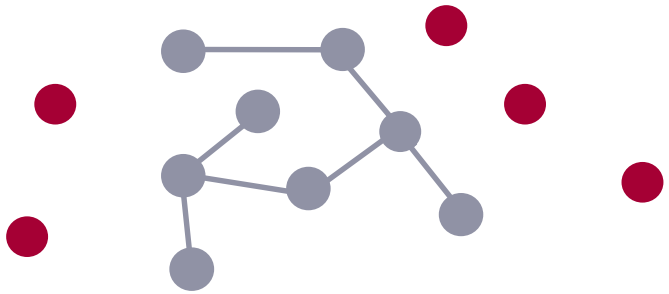
Concluding Remark

Future Works and Participation Experience

Future Works

- **Independent Stage Enhancements**
  - ▸Targeted improvements for each component
    (Bundle Enumeration / Tree Improvement)

- **Dynamic Order Handling : Tree Reconstruction**
  - ▸Incremental algorithm suitable for real-time order changes



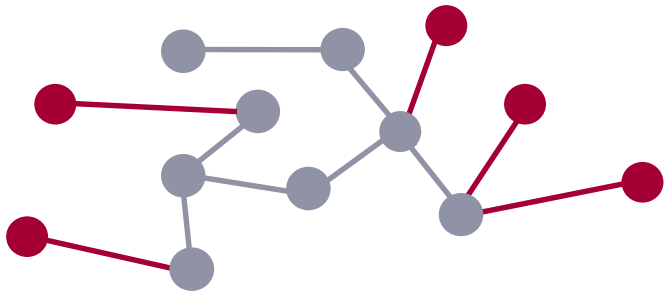
Concluding Remark

Future Works and Participation Experience

Future Works

- **Independent Stage Enhancements**
  - ▸Targeted improvements for each component
    (Bundle Enumeration / Tree Improvement)

- **Dynamic Order Handling : Tree Reconstruction**
  - ▸Incremental algorithm suitable for real-time order changes

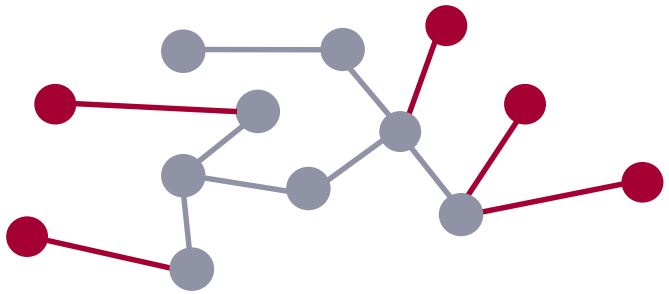


Concluding Remark

Future Works and Participation Experience

Future Works

- **Independent Stage Enhancements**
  - ▸Targeted improvements for each component
    (Bundle Enumeration / Tree Improvement)
- **Dynamic Order Handling : Tree Reconstruction**
  - ▸Incremental algorithm suitable for real-time order changes



Participation Experience

- **Suggested Future Directions for OGC**
  - ▸Regularly changing scoring instances for each stage, instead of fixed instances
  - ▸Providing additional information (e.g., amount of time exceeded, obj value), when the time limit is surpassed

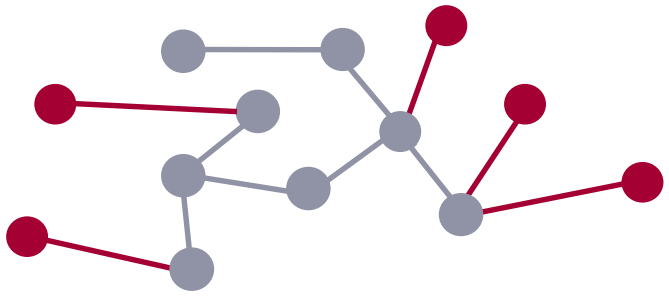
Concluding Remark

Future Works and Participation Experience

Future Works

- **Independent Stage Enhancements**
  - ▸Targeted improvements for each component
    (Bundle Enumeration / Tree Improvement)

- **Dynamic Order Handling : Tree Reconstruction**
  - ▸Incremental algorithm suitable for real-time order changes



Participation Experience

- **Suggested Future Directions for OGC**
  - ▸Regularly changing scoring instances for each stage, instead of fixed instances
  - ▸Providing additional information (e.g., amount of time exceeded, obj value), when the time limit is surpassed

- **Reflections on Participation**
  - ▸Opportunity to explore algorithms suitable for real-world problems using provided data
  - ▸Chance to develop and advance research topics

Thank You !

Team PRO