Learning-Based Approaches to Combinatorial Optimization in Transportation

Changhyun Kwon

2025. 10. 15

Changhyun Kwon

KAIST 산업및시스템공학과
Department of Industrial and Systems Engineering

Associate Professor

- Computational Optimization Methods Lab
- http://comet.kaist.ac.kr

Omelet **CTO**

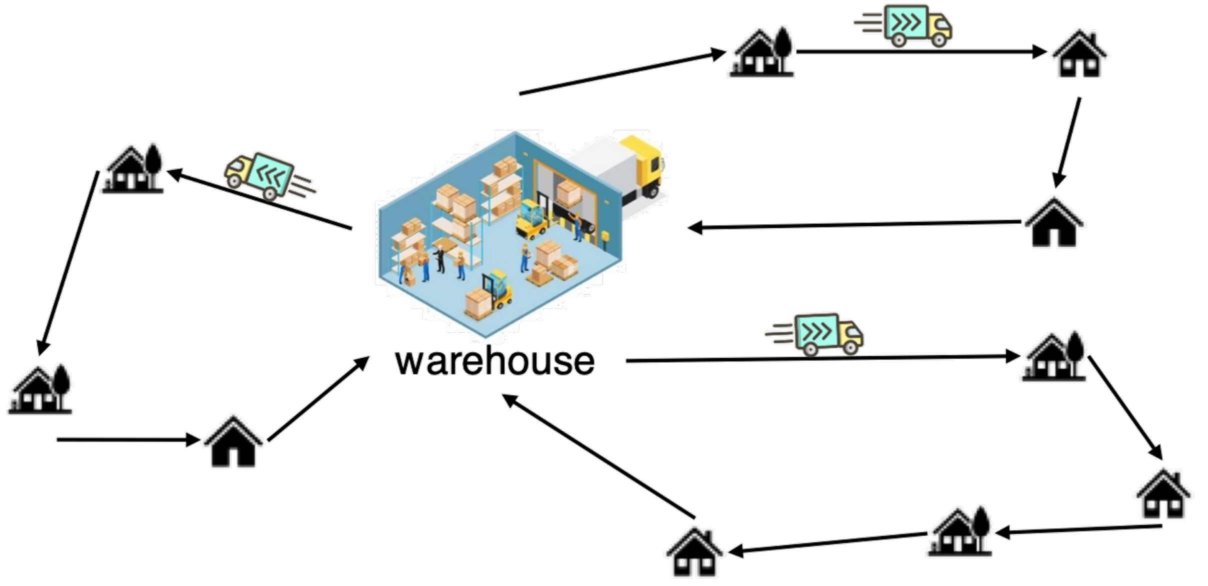
- Optimization and Machine Learning Technologies
- https://omelet.ai

chkwon@kaist.ac.kr

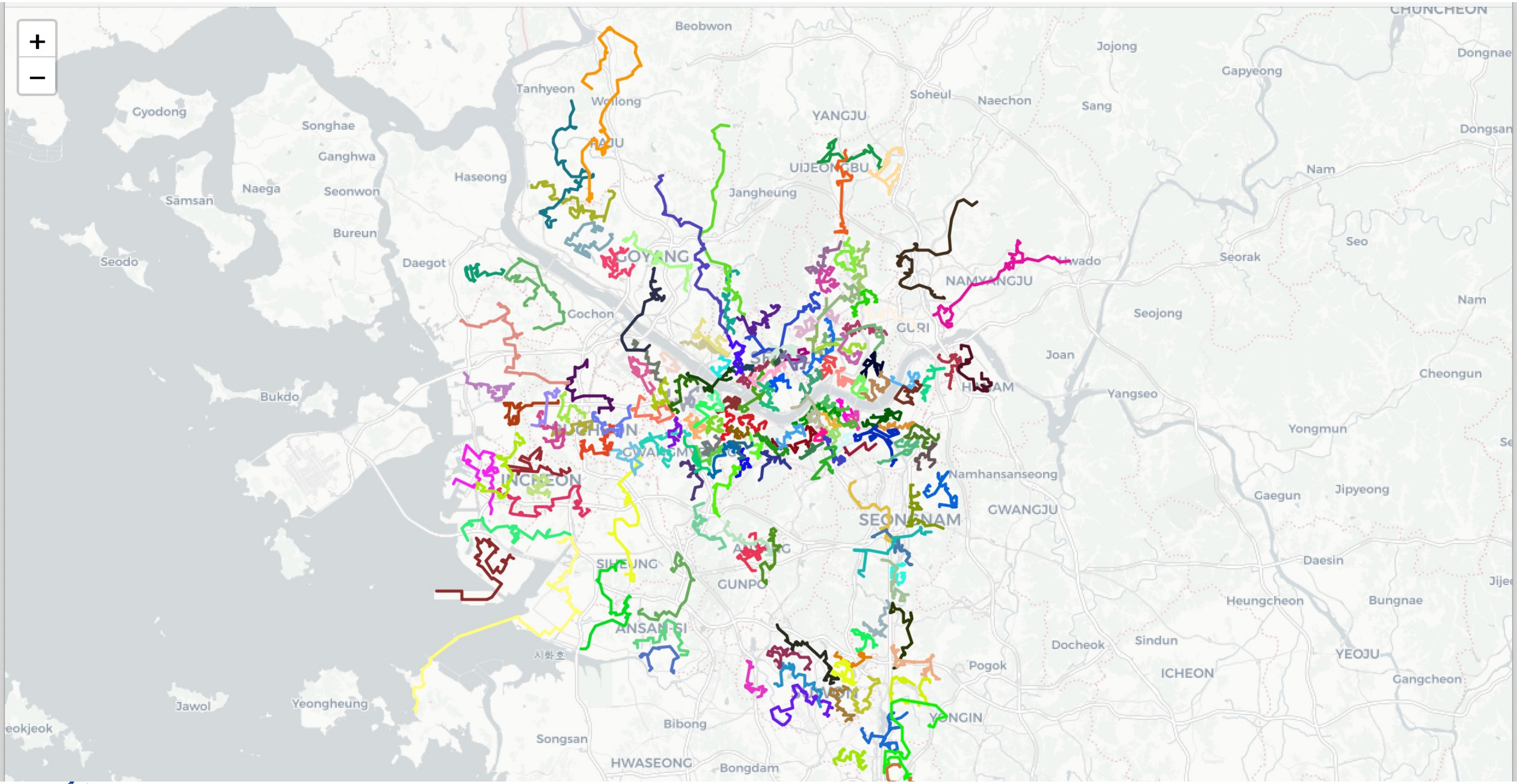
Vehicle Routing Problems (VRP)

Capacitated VRP (CVRP)

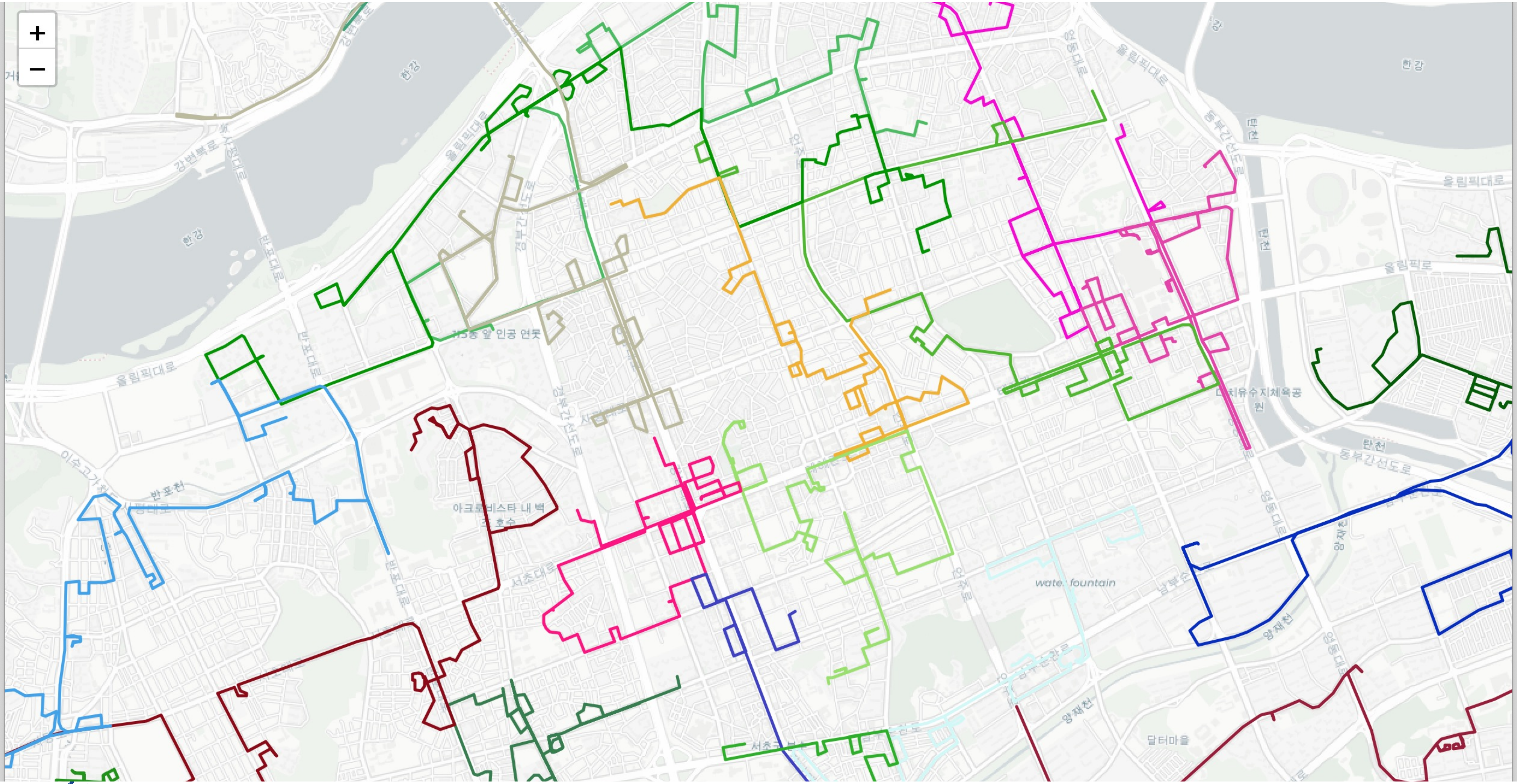
- Given a set of customers, considering the vehicle capacity, CVRP determines
  - Which customer is served by which vehicle
  - In what order
- To minimize the total distance traveled



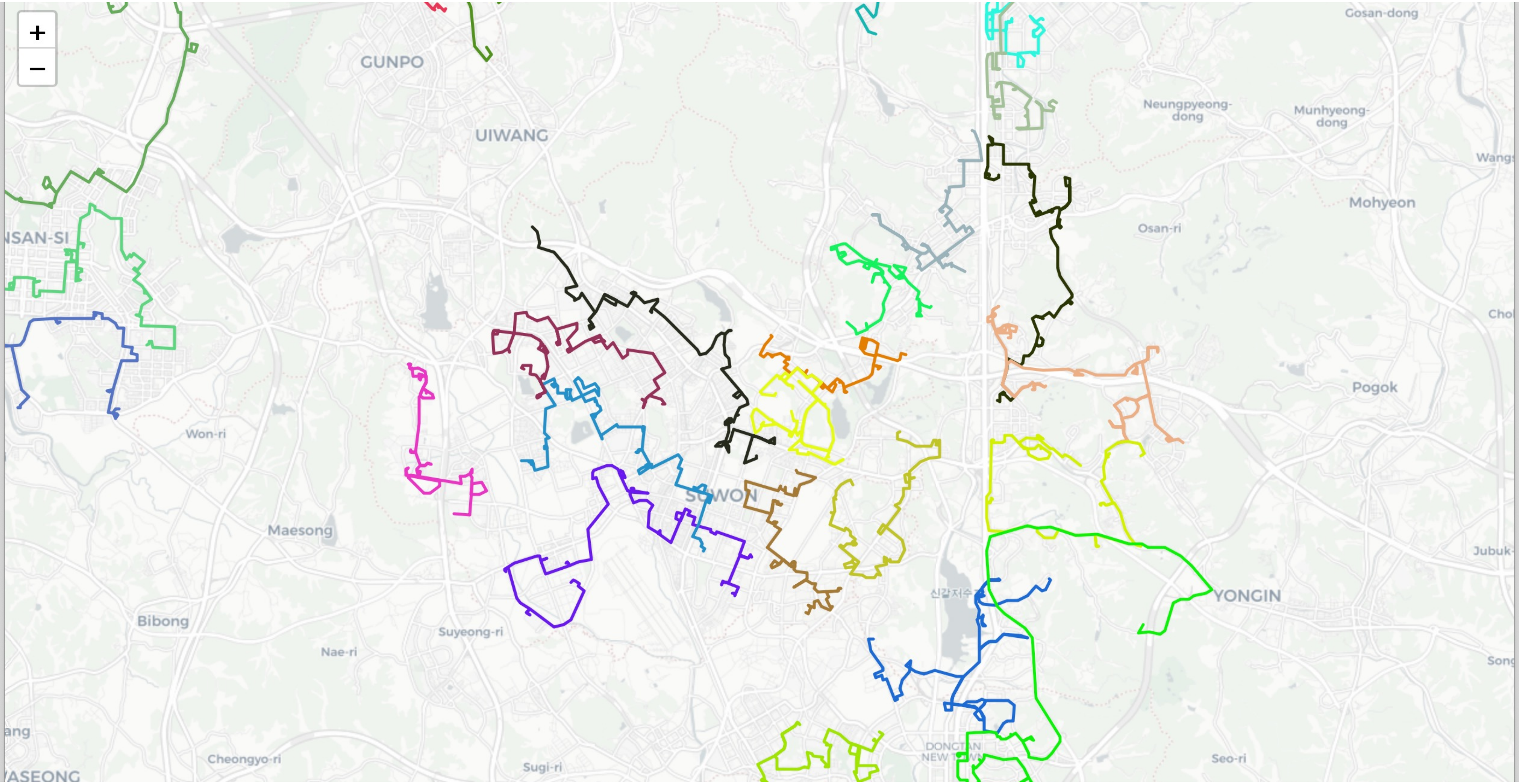
A Practical VRP



A Practical VRP



A Practical VRP



Vehicle Routing Problems

- NP-hard Combinatorial Optimization, in general
- Algorithms:
  - **Exact Methods**: Theory-based / Branch-and-Cut, Branch-and-Price, Branch-Cut-and-Price
  - **Metaheuristics**: Genetic Algorithm, Large Neighborhood Search
  - **Local Search Methods**
  - **Deep Learning Methods (Neural Combinatorial Optimization)**

Generative Models for Translation

- "the cat sat on the mat"

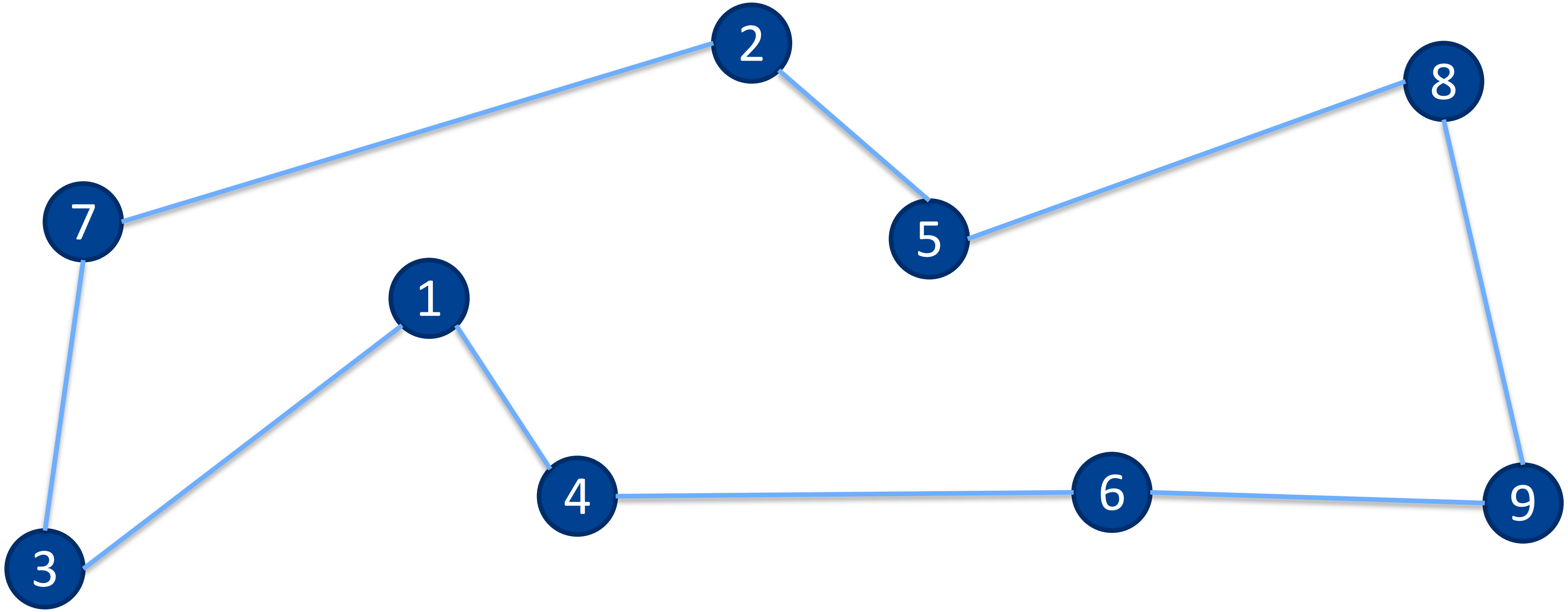
  → [generative model]

  → "le chat etait assis sur le tapis"

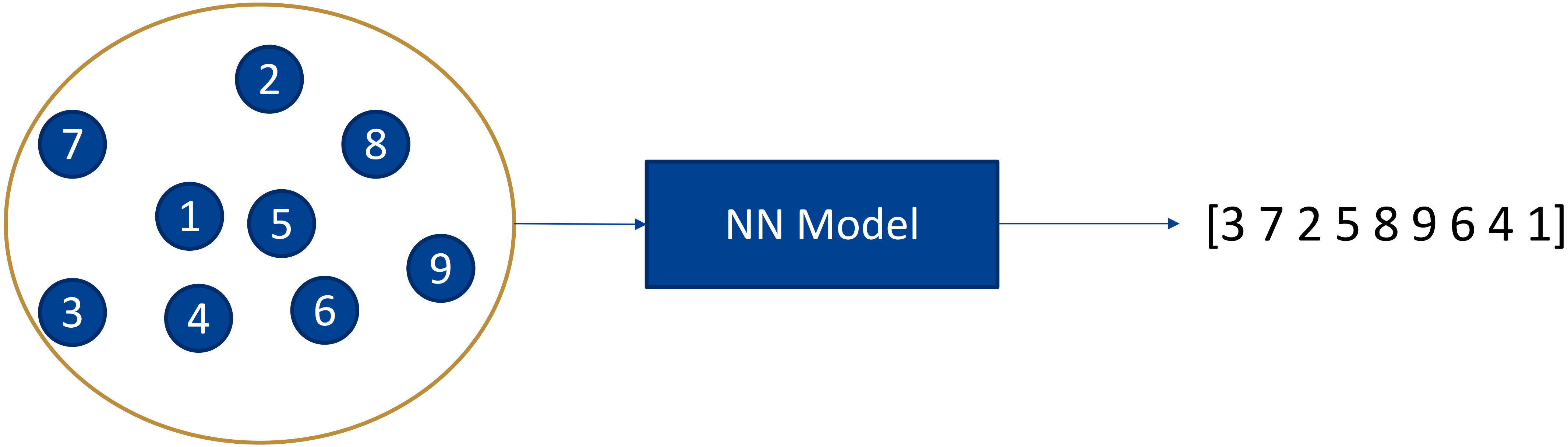
- "[1 2 3 4 5 6 7 8 9]" → [generative model] → "[9 3 2 8 4 5 1 6 7]"

Traveling Salesman Problem

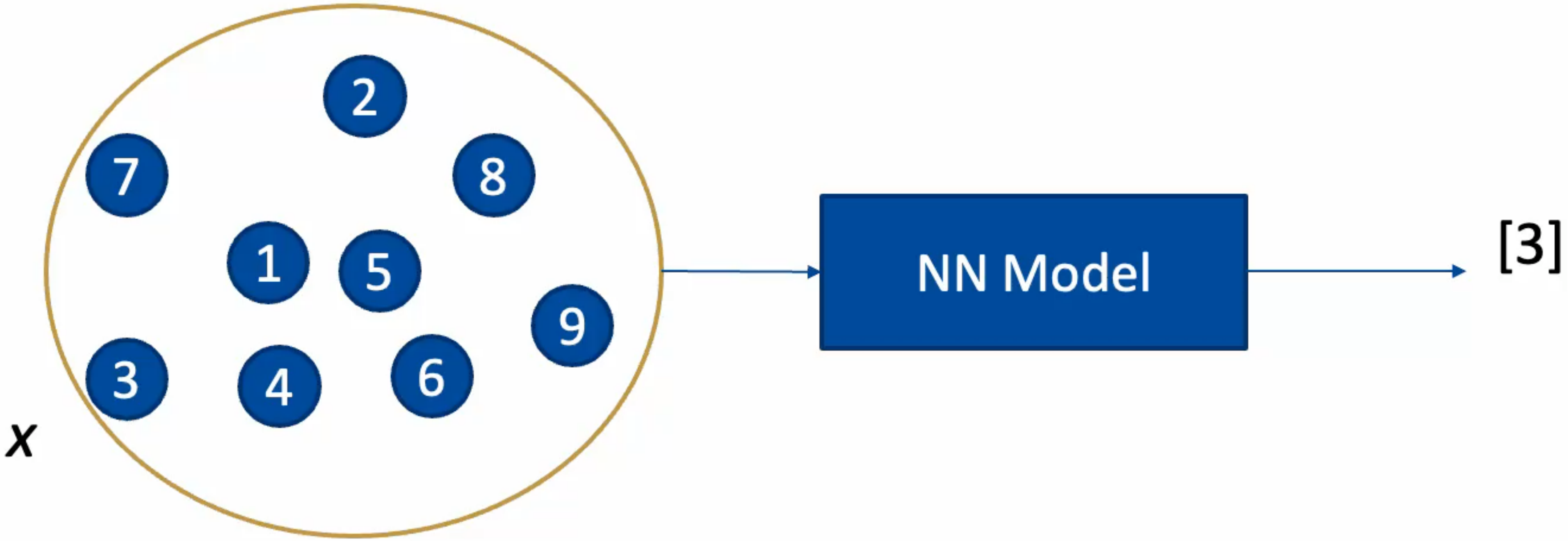
- Given n cities, find the **shortest tour** visiting all cities



Traveling Salesman Problem

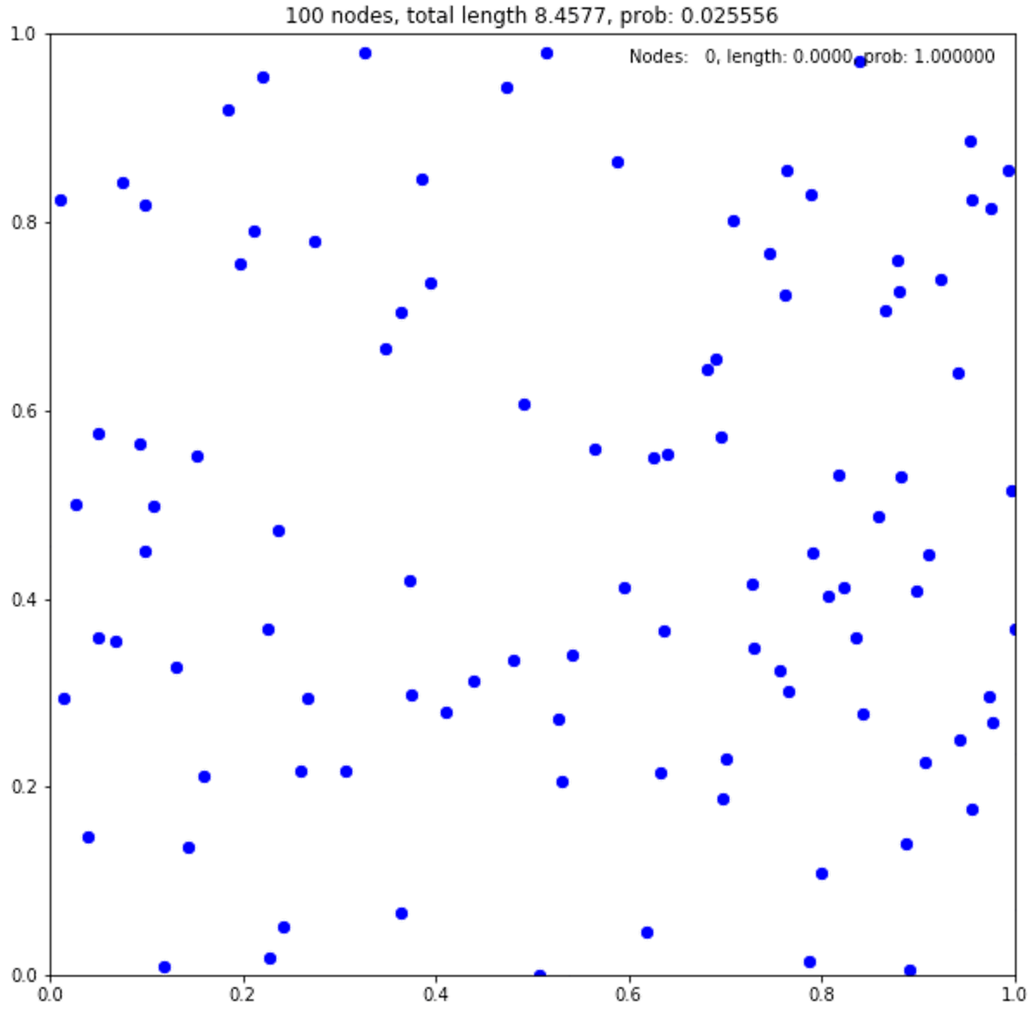


Traveling Salesman Problem



Attention Model (AM) for TSP

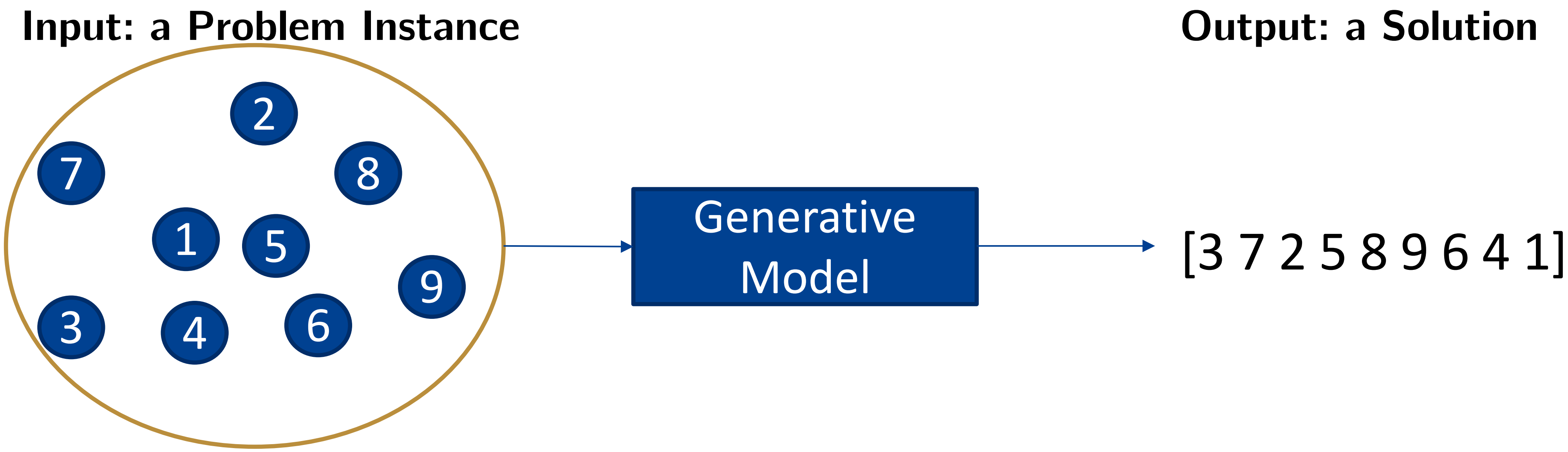
- **Transformer-like** encoder/decoder
- [Kool et al. (2019) ICLR]
- Each time, the node with the **highest probability** is chosen as the next city.



https://github.com/wouterkool/attention-learn-to-route

End-to-End Learning

- Learning a heuristic algorithm
- Providing an approximate solution

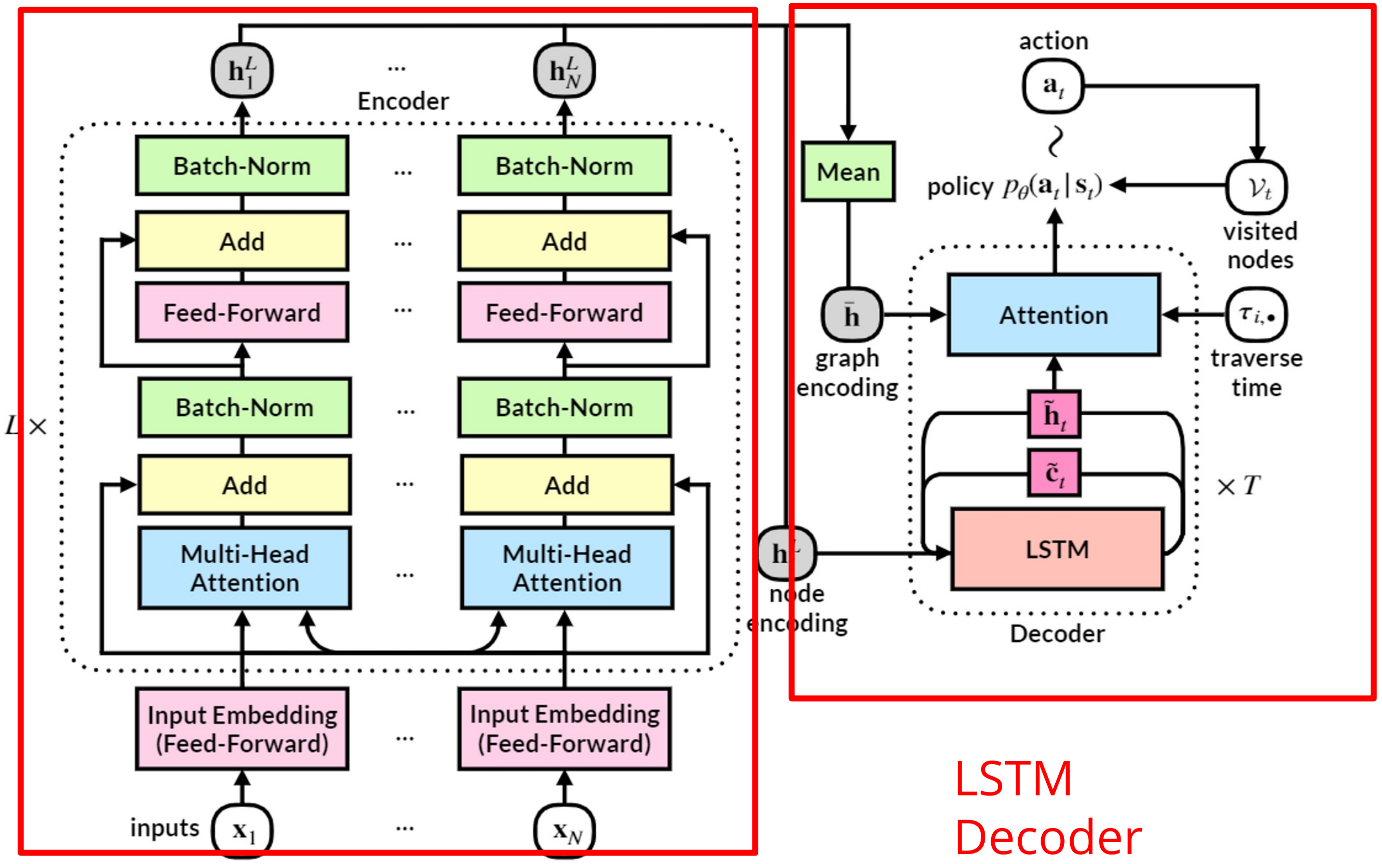


An Example: TSP with Drone

- A collaborative delivery between a truck and a drone

A Hybrid Model for TSP-D

- Attention Encoder
- LSTM Decoder
- Input:
  - Graph encoding
  - Node encoding
  - Traverse time
- Output:
  - Node choice probabilities

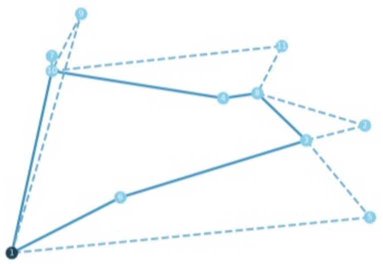
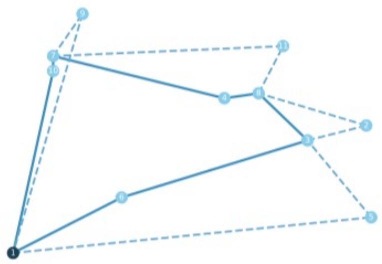
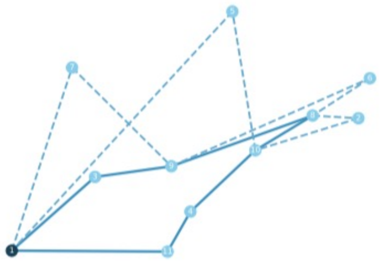
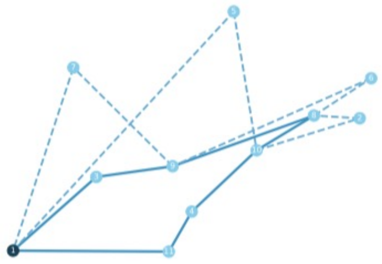
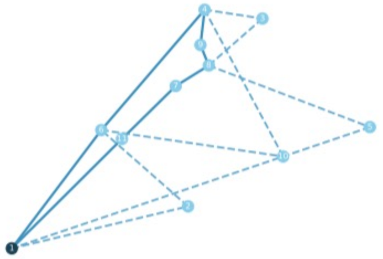
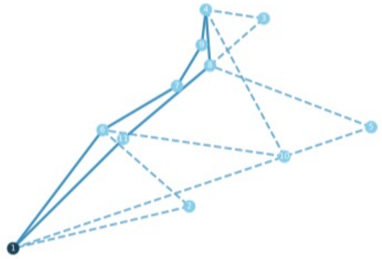


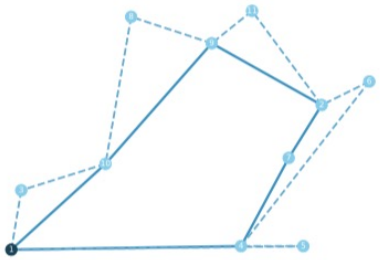
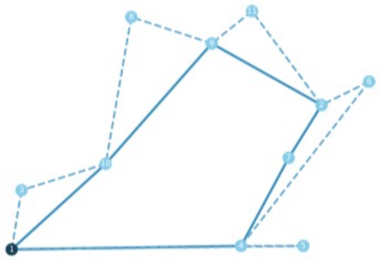
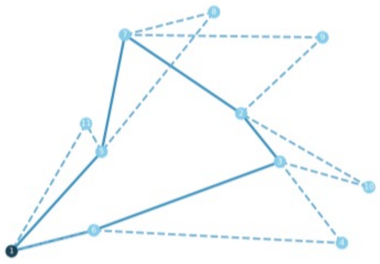
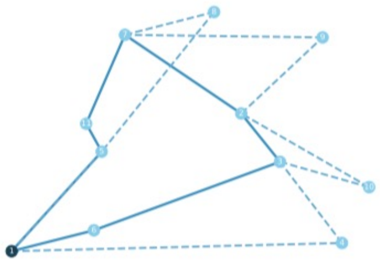
LSTM Decoder

Multi-Head Attention Encoder

Bogyrbayeva et al. (2023) Transportation Research Part C
http://dx.doi.org/10.1016/j.trc.2022.103981

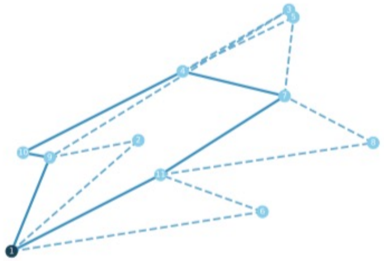
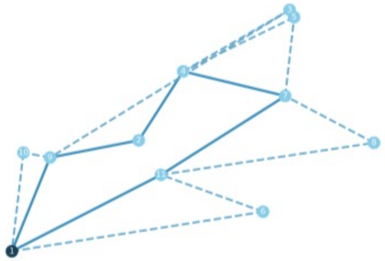
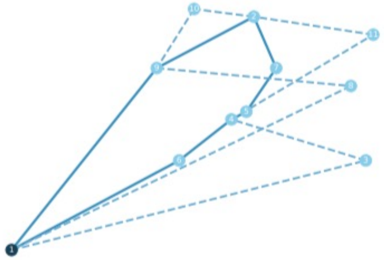
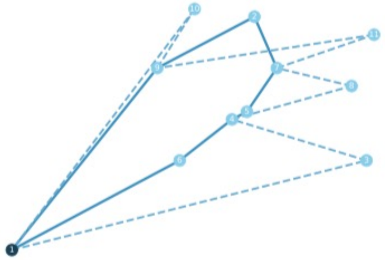
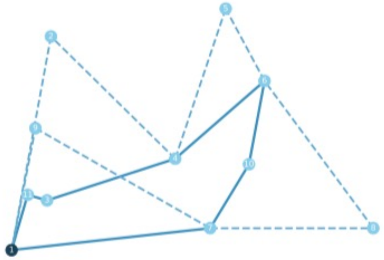
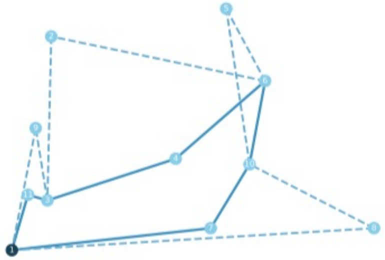
Toy problems with 11 nodes

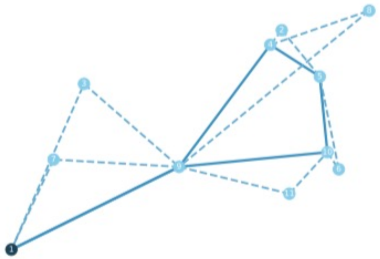
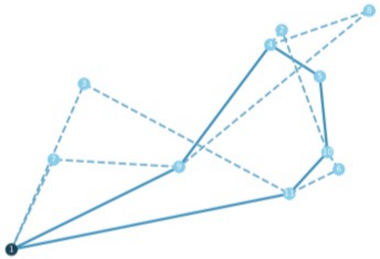
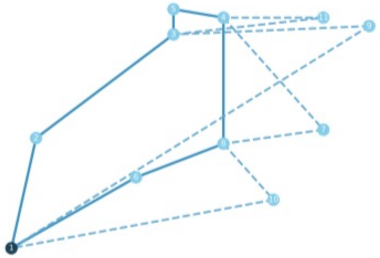
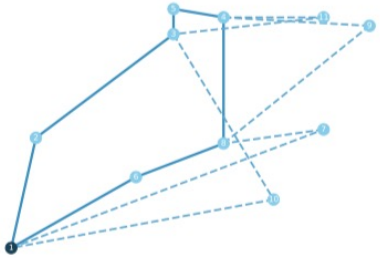
Instance	Optimal Solution	HM Greedy
1	 Cost = 221.19	 Cost = 223.41 (Gap = 1.00%)
2	 Cost = 205.76	 Cost = 205.76 (Gap = 0.00%)
3	 Cost = 192.96	 Cost = 193.99 (Gap = 0.53%)

Instance	Optimal Solution	HM Greedy
4	 Cost = 241.26	 Cost = 241.26 (Gap = 0.00%)
5	 Cost = 248.14	 Cost = 249.85 (Gap = 0.69%)

Bogyrbayeva et al. (2023) Transportation Research Part C
http://dx.doi.org/10.1016/j.trc.2022.103981

Toy problems with 11 nodes

Instance	Optimal Solution	HM Greedy
6	 Cost = 217.69	 Cost = 217.69 (Gap = 0.00%)
7	 Cost = 237.34	 Cost = 240.47 (Gap = 1.32%)
8	 Cost = 214.77	 Cost = 226.64 (Gap = 5.53%)

Instance	Optimal Solution	HM Greedy
9	 Cost = 256.34	 Cost = 256.83 (Gap = 0.19%)
10	 Cost = 227.90	 Cost = 227.90 (Gap = 0.00%)

Bogyrbayeva et al. (2023) Transportation Research Part C
http://dx.doi.org/10.1016/j.trc.2022.103981

Pros and Cons of End-to-End Learning

- **Pros**
  - Fast computation
  - Parallelizable on GPU
- **Cons**
  - Quality solutions, but not as good as mathematical optimization
  - Long training time

Learning-Optimization Hybrid Method

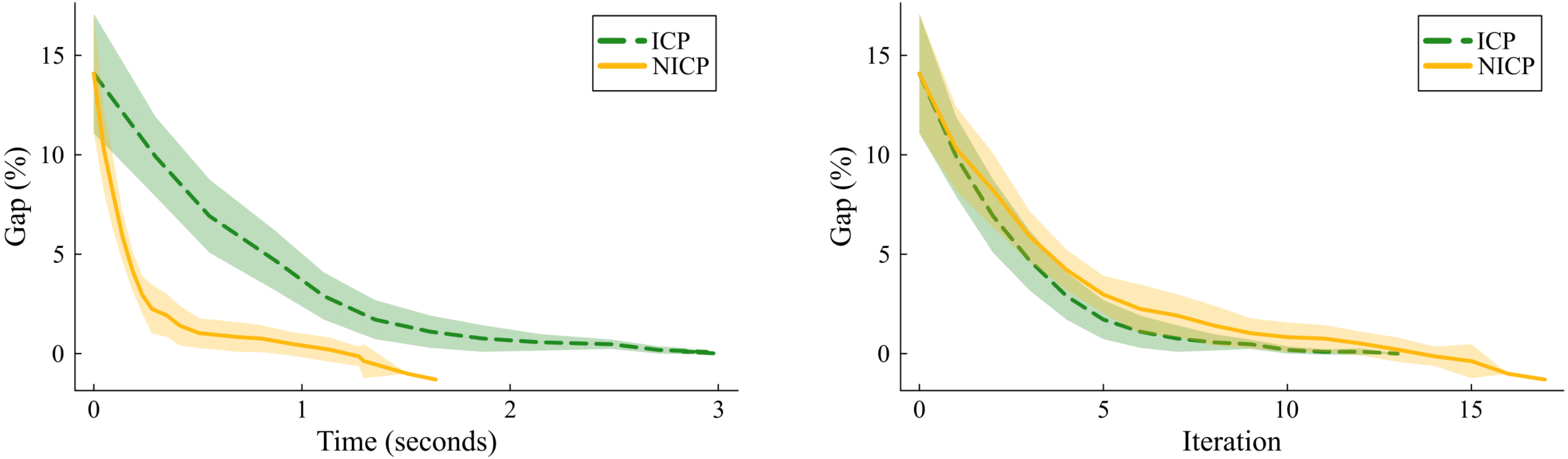
- For an optimization algorithm, replace a **human-designed component** with a **learning model**
  1. Heuristics + Learning Model
  2. Meta-heuristics + Learning Model
  3. Exact algorithms + Learning Model

Heuristics and Metaheuristics

with Learning

1. Heuristic + Learning

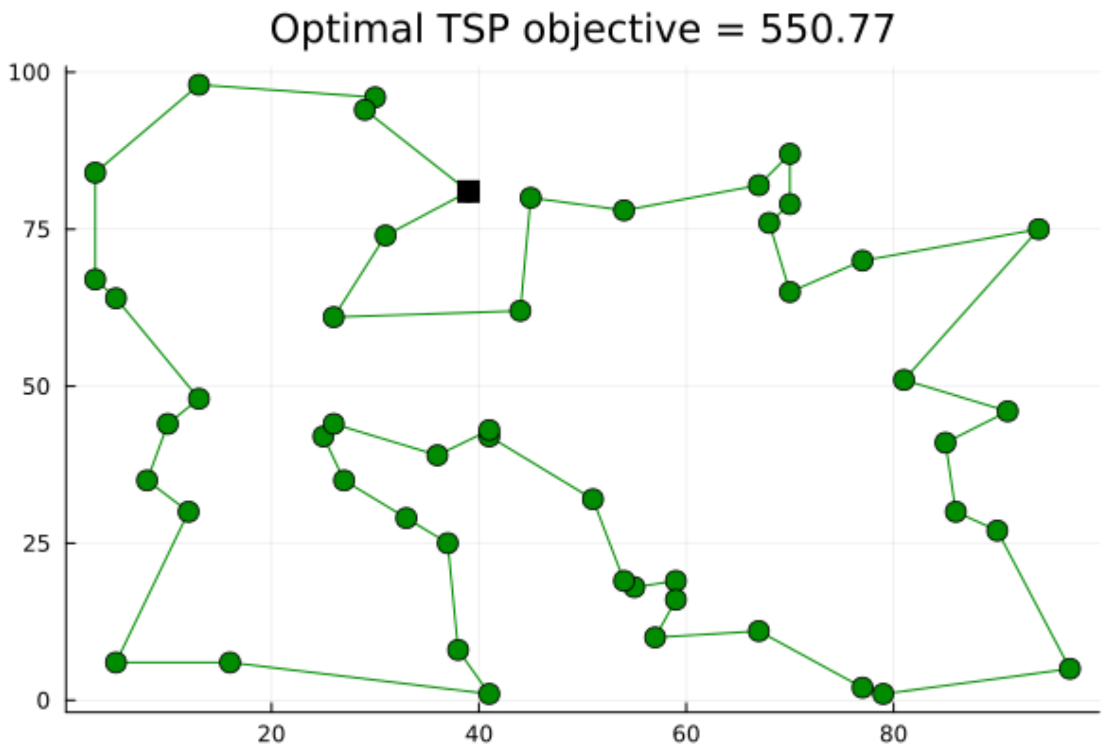
- ML model **guides** heuristic algorithms for **faster** computation

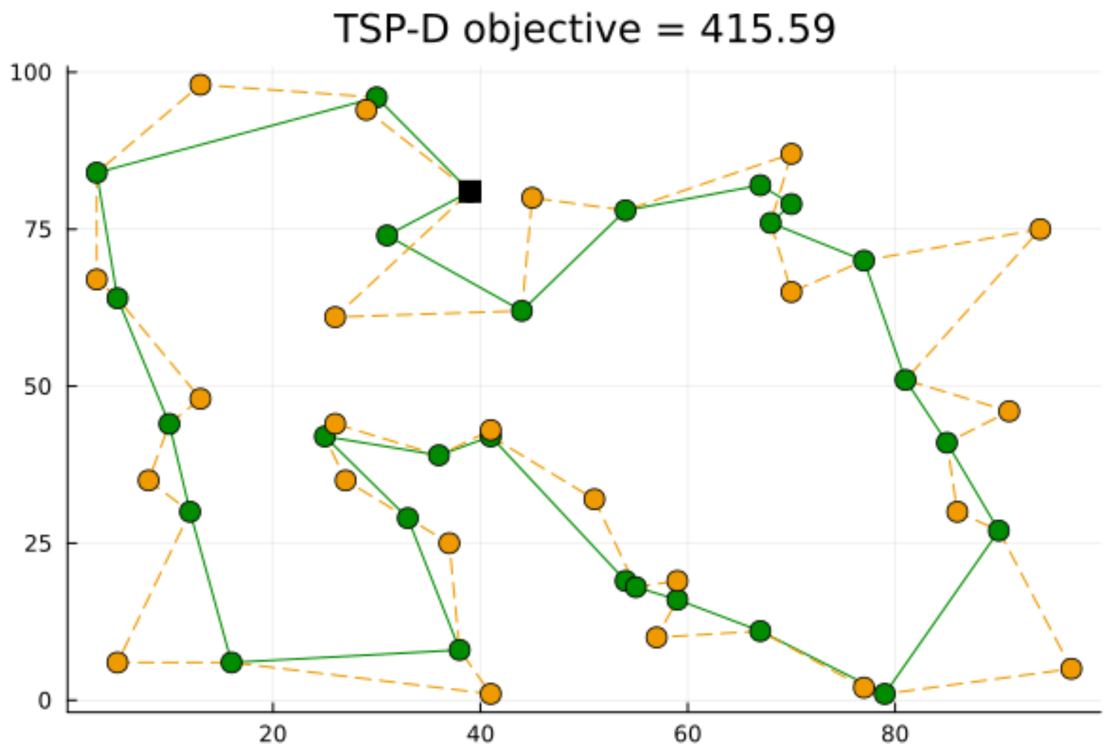


Lee, Kim, Kim, Park, Kwon (2025) The Iterative Chainlet Partitioning Algorithm for the Traveling Salesman Problem with Drone and Neural Acceleration, preprint. https://arxiv.org/abs/2504.15147

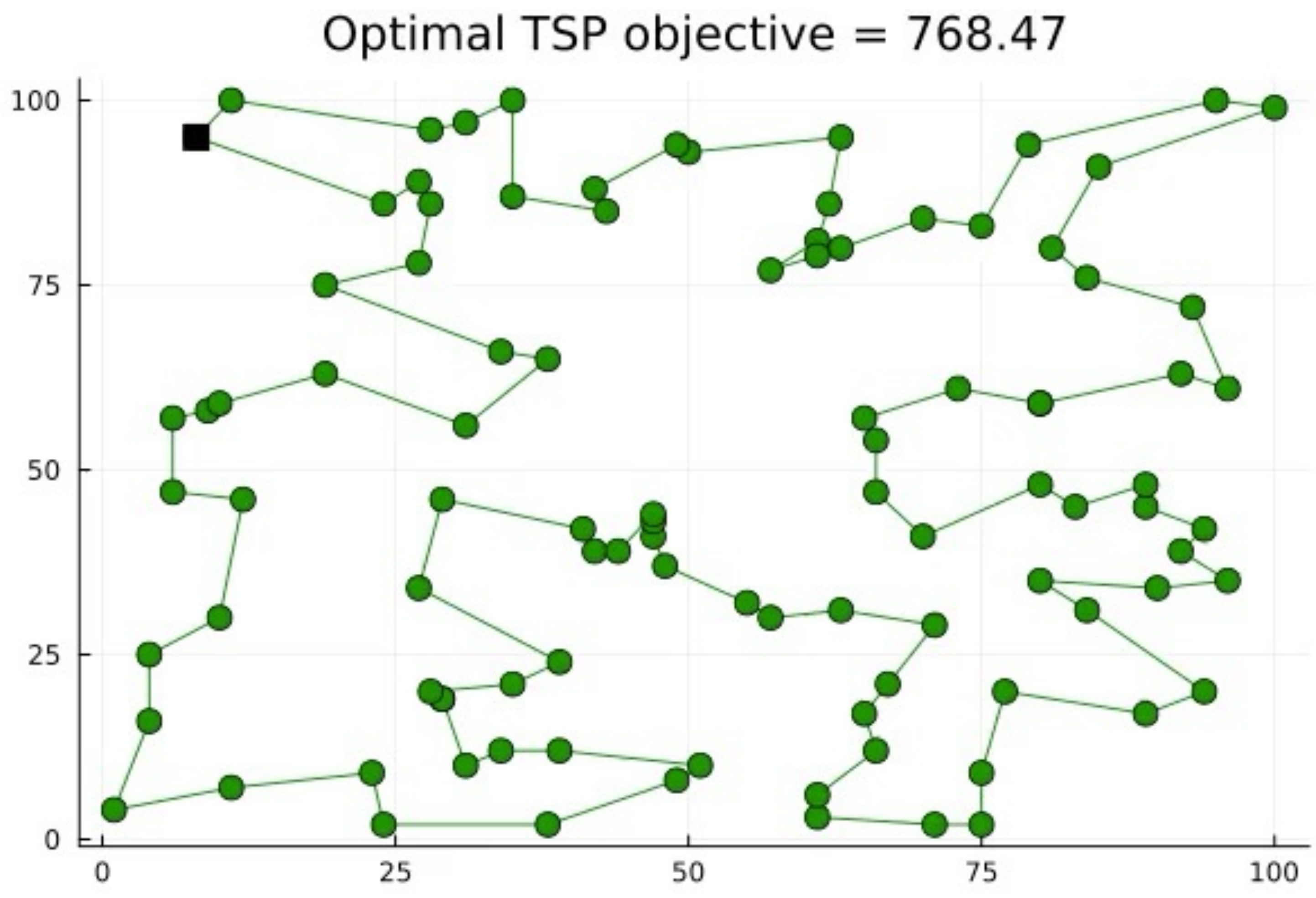
Exact Partitioning (EP)

- Given a TSP tour, partition nodes to truck nodes and drone nodes, while respecting the orders



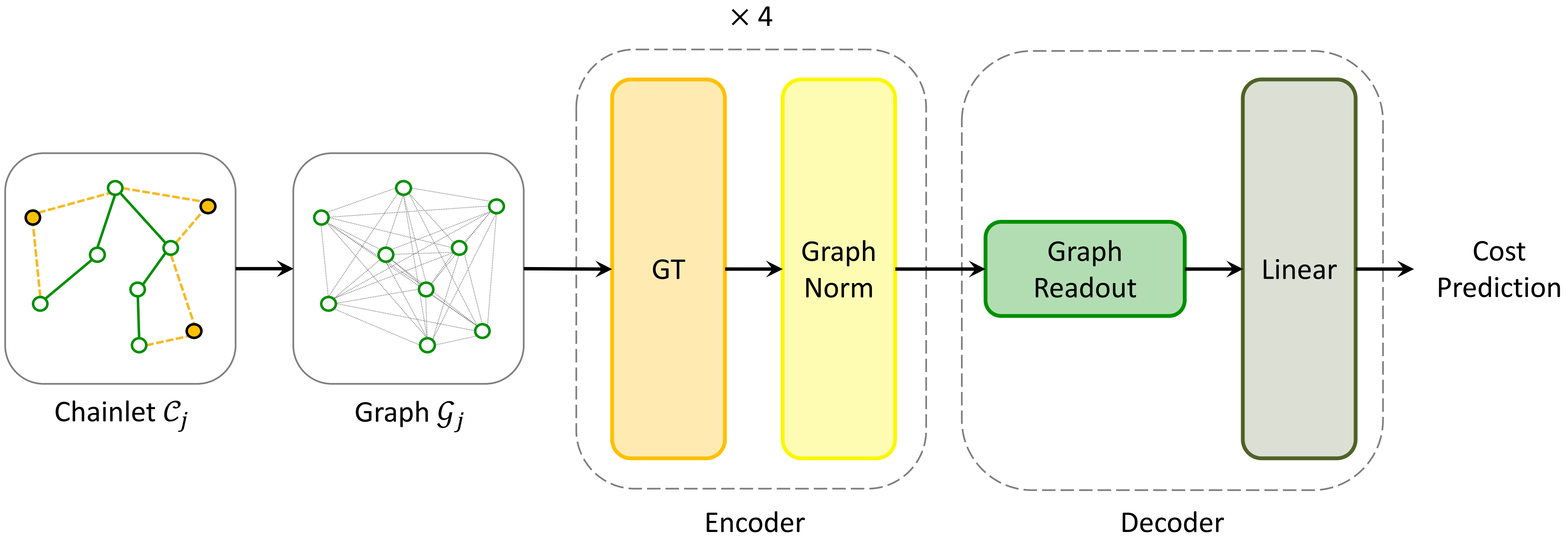


An Iterative Method



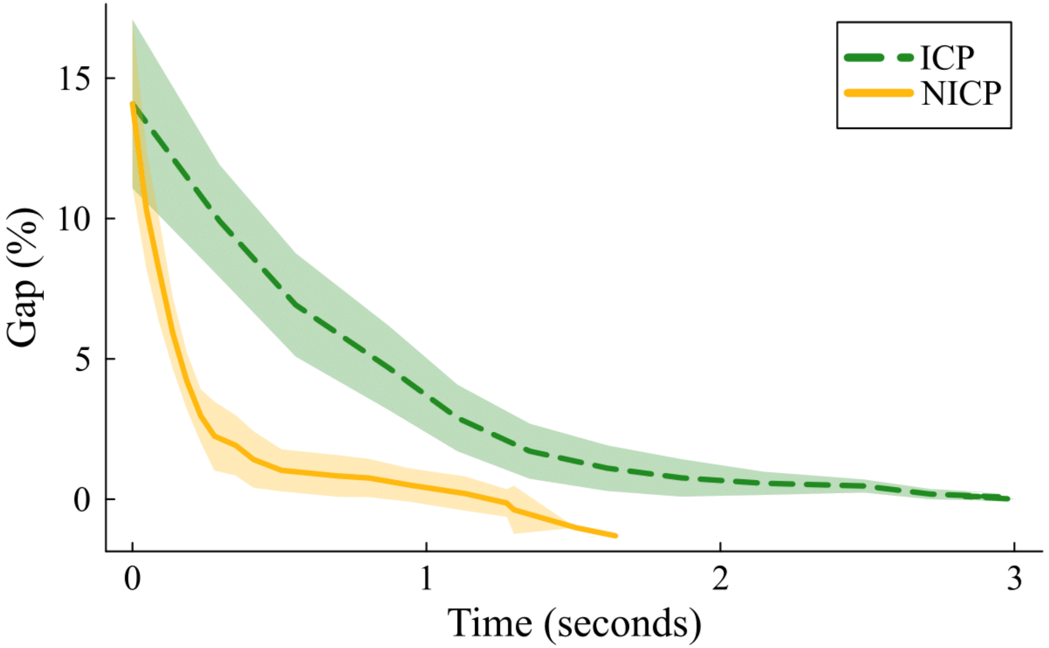
Can we make it faster?

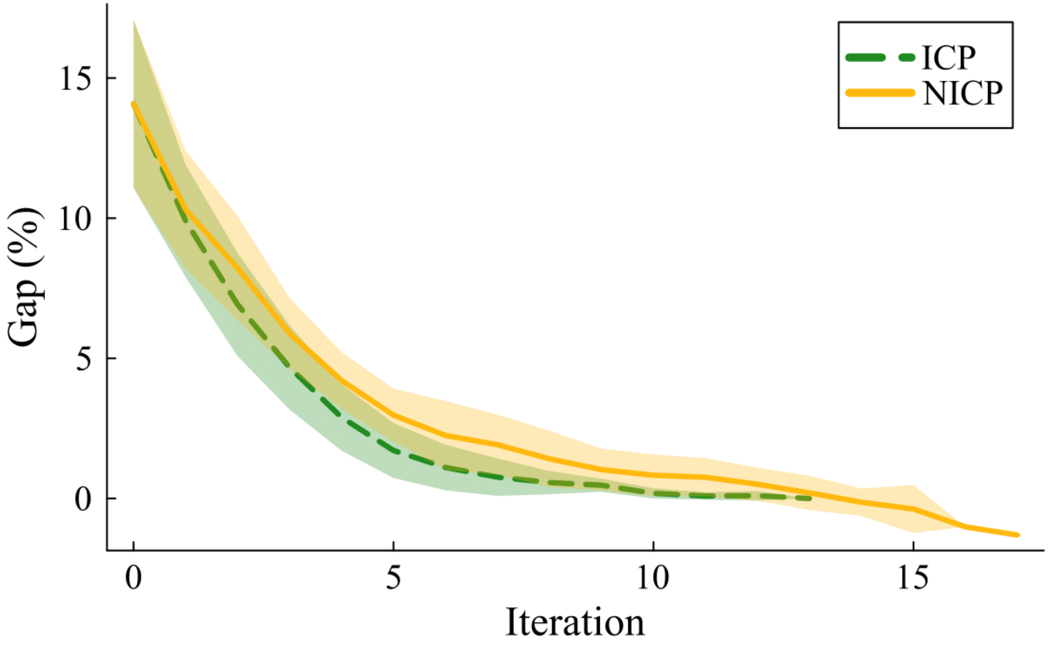
- To save computation time in ICP, instead of directly executing dynamic programming to obtain an improvement, the value is **predicted** through a **graph neural network**.



Heuristic + Learning

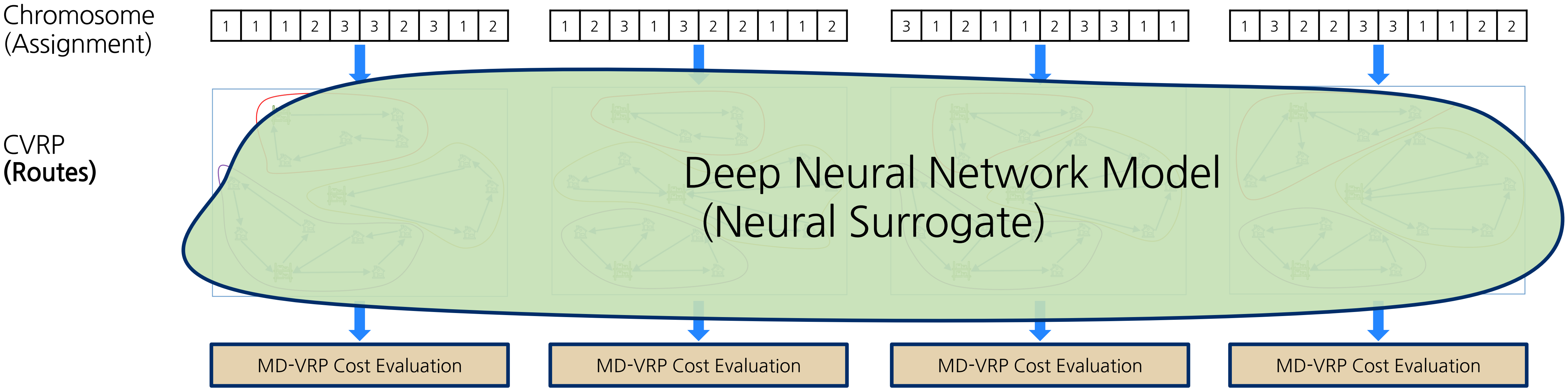
- ML model **guides** heuristic algorithms for **faster** computation





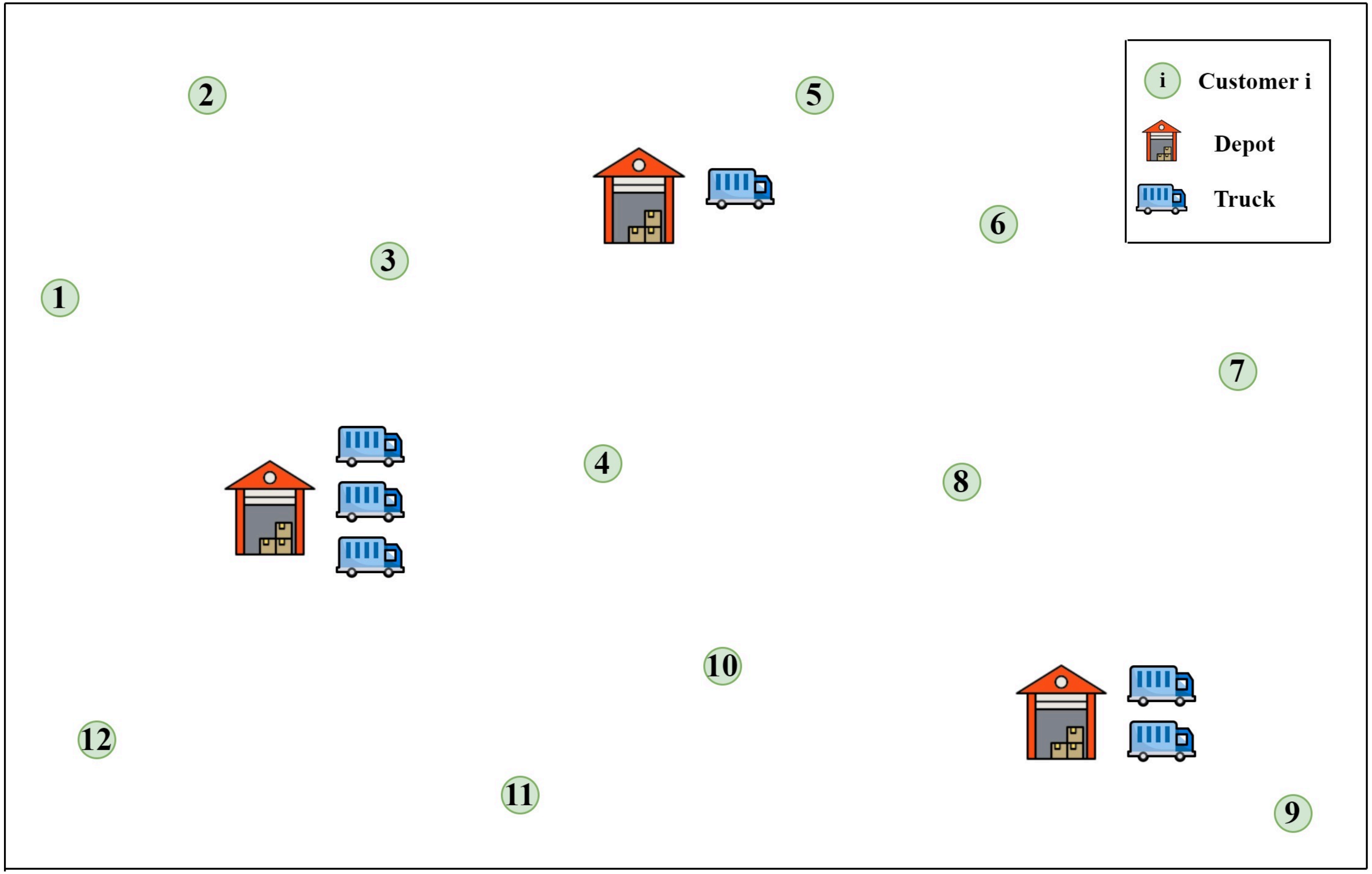
2. Meta-Heuristic + Learning

- ML model **replaces** computationally **expensive** components in meta-heuristics

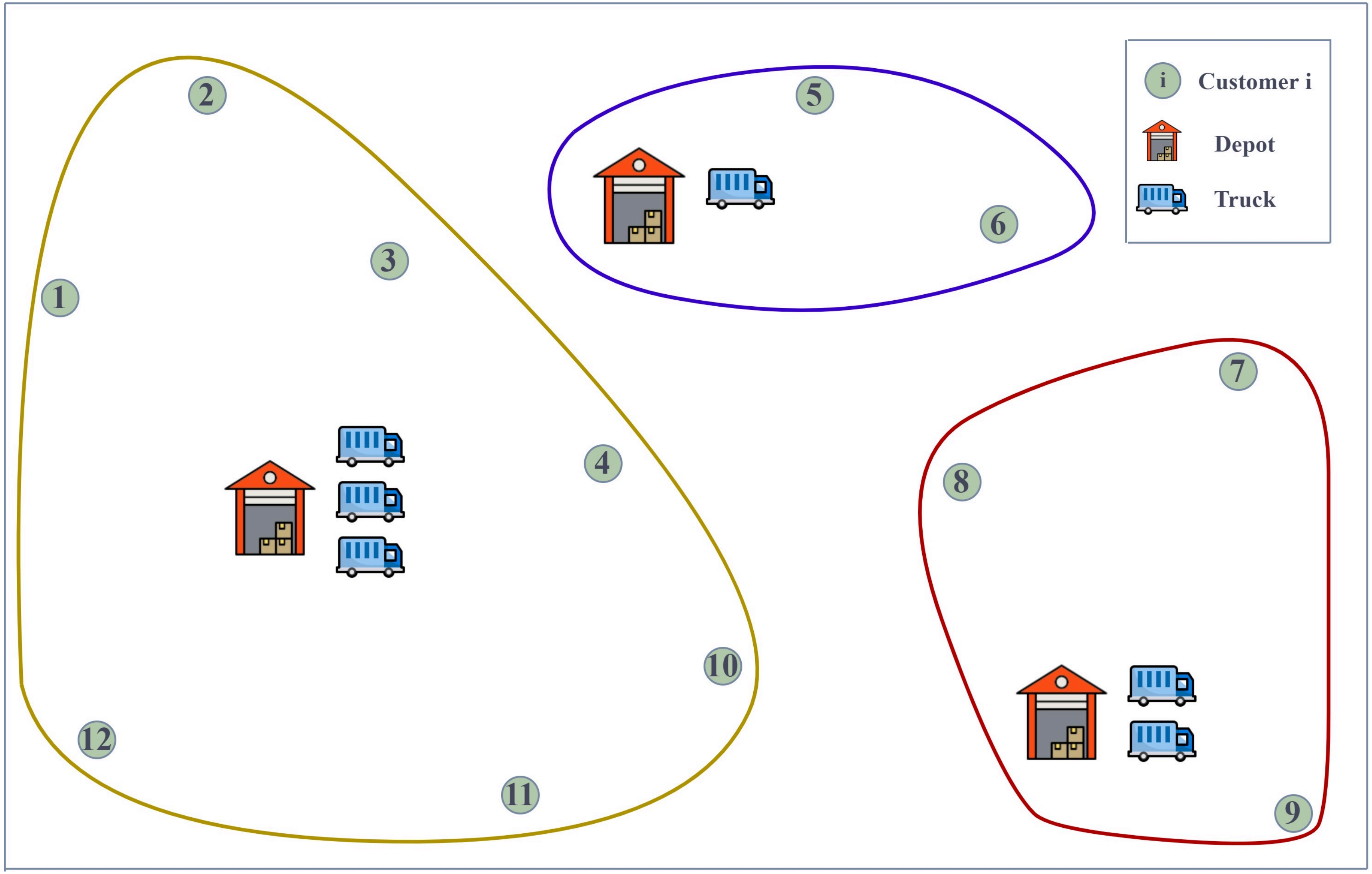


Sobhanan, Park, Park, Kwon (2025) Genetic Algorithms with Neural Cost Predictor for Solving Hierarchical Vehicle Routing Problems, *Transportation Science*, 52(2), 322–339 https://doi.org/10.1287/trsc.2023.0369

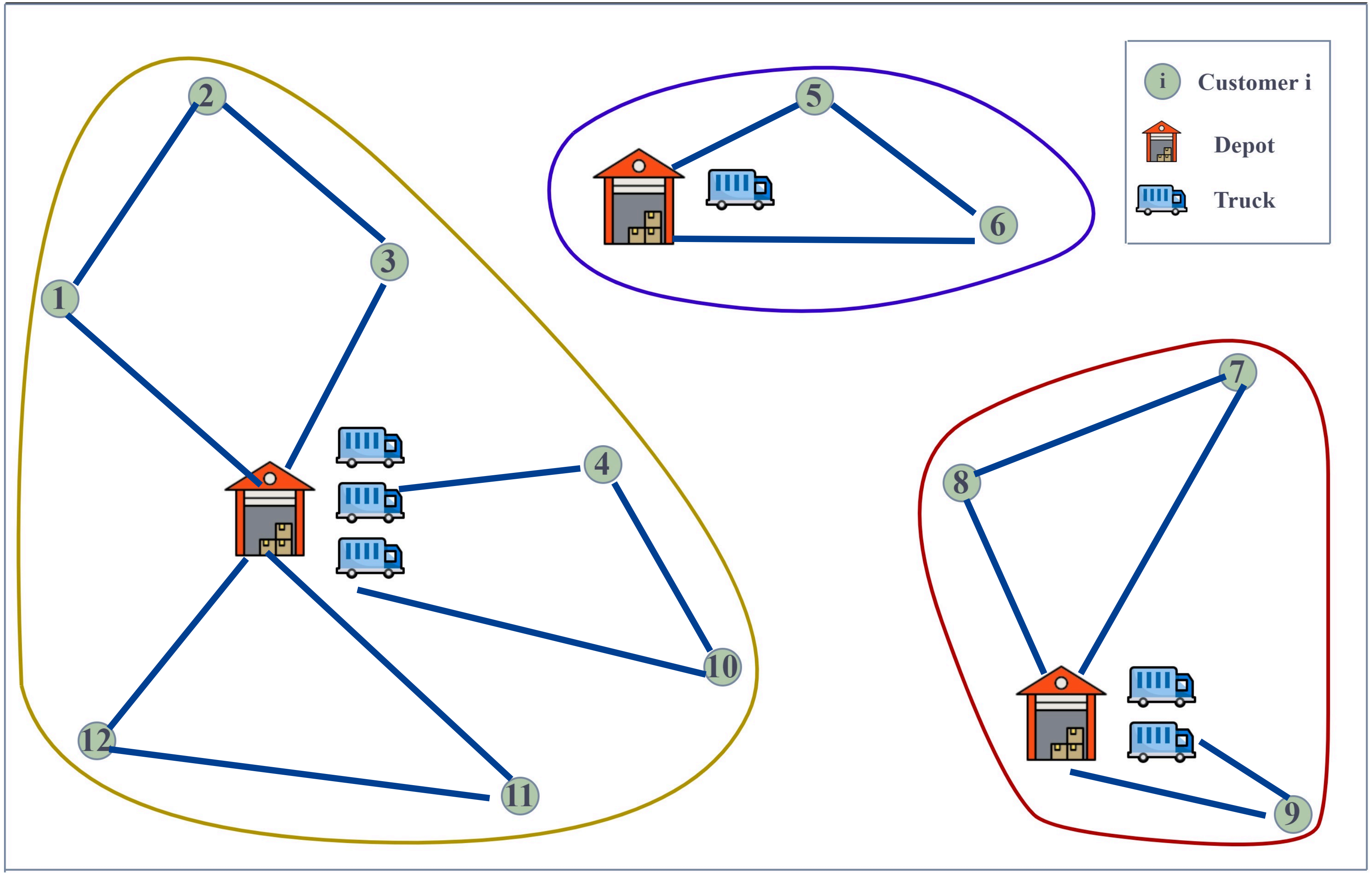
Multi-Depot VRP



(i) Customer-Depot Assignment

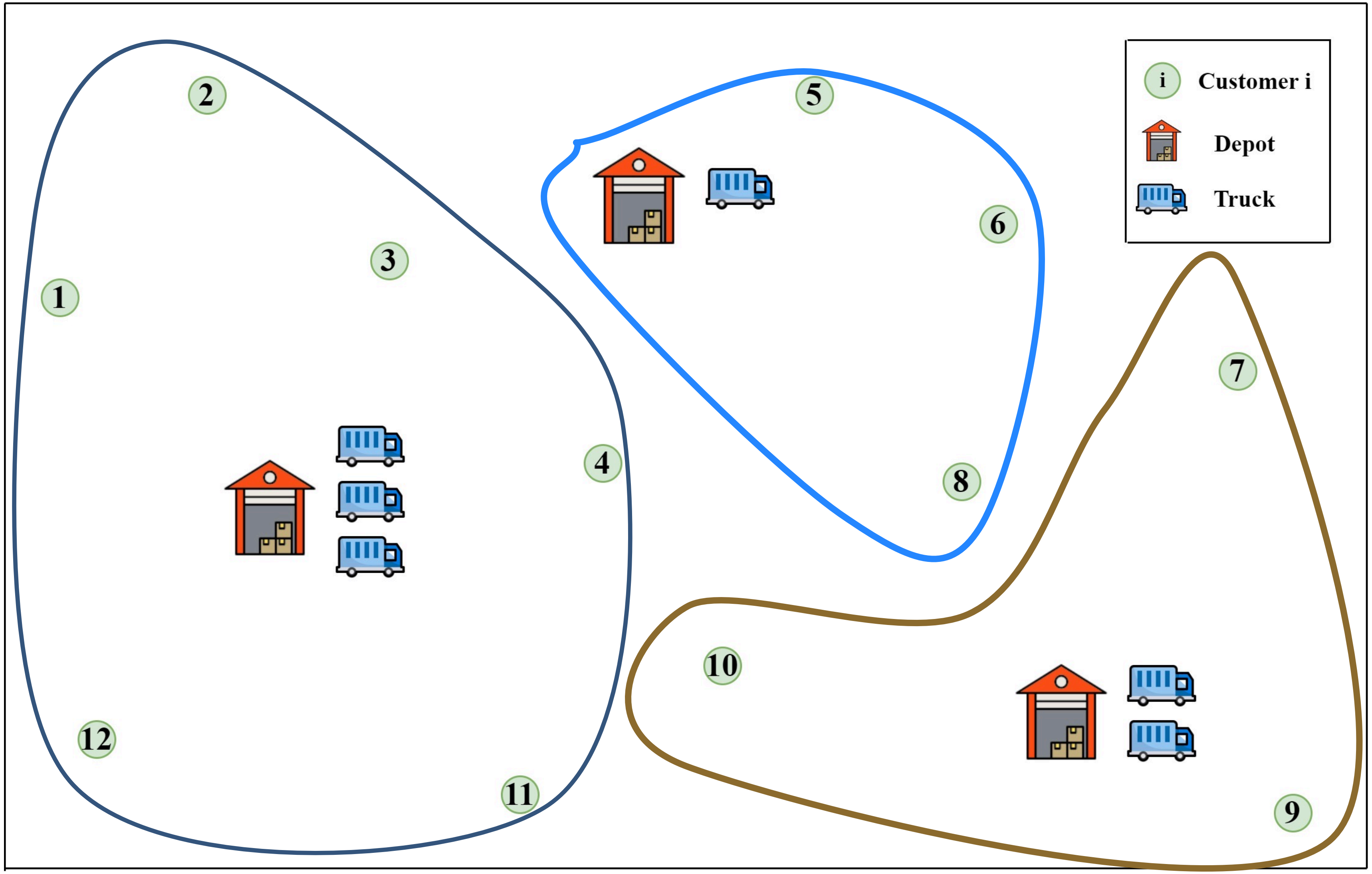


(ii) Routing



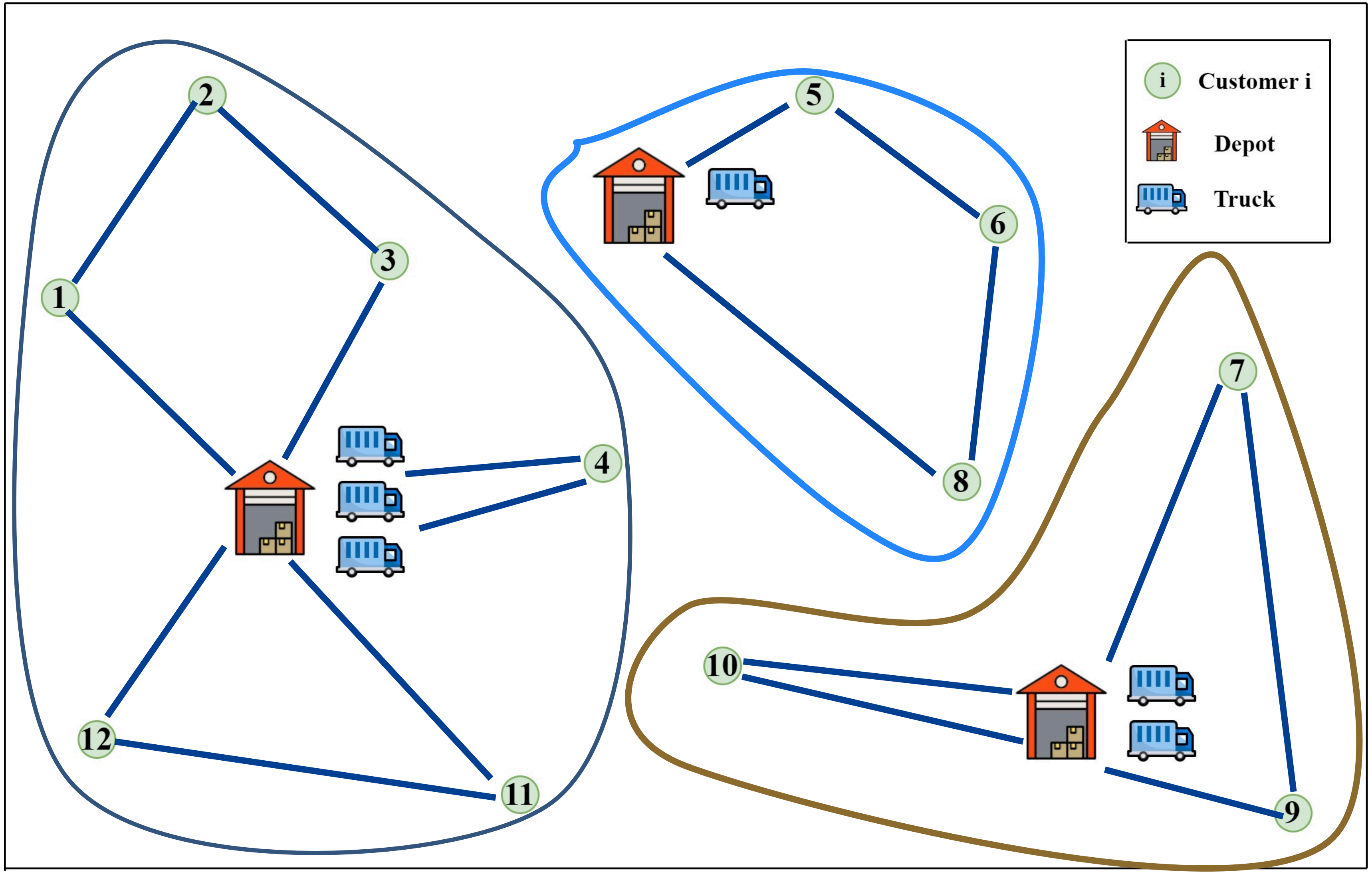
(i) Customer-Depot Assignment

Another Trial



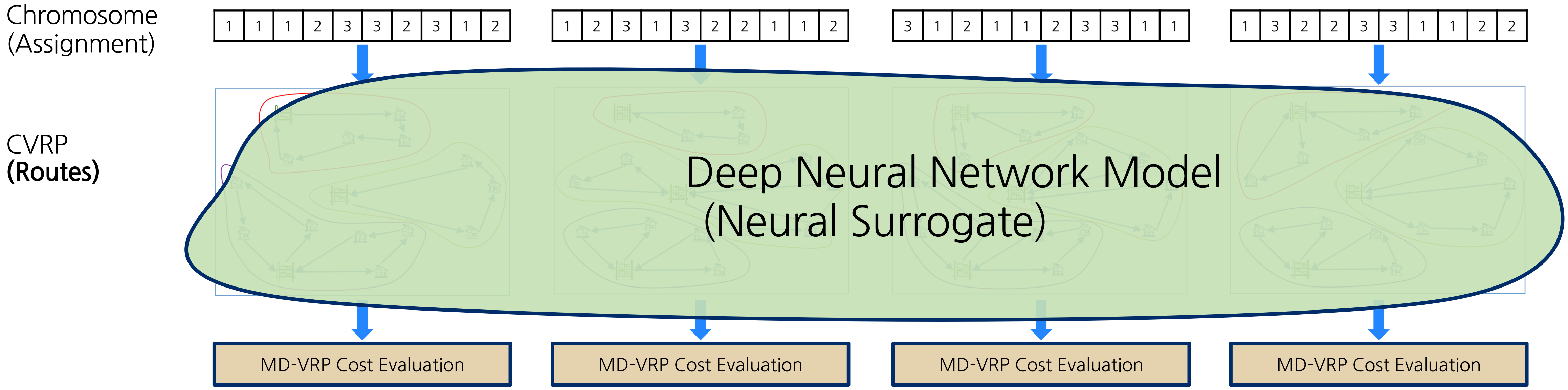
(ii) Routing

Another Trial



Genetic Algorithms with Neural Cost Predictor (GANCP)

- ML model **replaces** computationally **expensive** components in meta-heuristics



Results

# Nodes	VROOM-l Cost	VROOM-l Time (sec)	GANCP	Time (sec)
[100,200]	29498.94	1.63	29117.09	1.65
[201,300]	41080.90	4.24	40341.34	3.10
[301,400]	48987.85	9.14	48376.34	5.51
[401,500]	58347.64	15.88	57233.29	8.75
[501,600]	74519.29	25.28	74756.34	10.61
[601,700]	78745.10	34.38	78006.76	14.64
[701,800]	101364.25	47.82	99638.54	16.96
[801,900]	101510.61	52.40	100531.98	21.01
[901,1000]	118045.20	63.88	116908.49	24.98
[1001,1100]	125837.19	80.52	124396.17	29.29
[1101,1200]	124255.41	87.06	122553.92	36.35
[1201,1300]	141564.49	101.18	140751.63	42.02
[1301,1400]	155932.46	108.11	156527.15	47.33
[1401,1500]	167286.57	119.85	166412.10	50.08

Where are we heading?

Classical methods are strong baselines

- **As an automated heuristic**, a deep learning method can be directly applied to different problems.
- However, they often **underperform** classical **optimization** methods for each problem.

Table 5.2: Gaps obtain with different RL algorithms. Best in bold, second best underlined.

N=50. Gaps are in %, compared to the best exact or heuristic optimization algorithms

Method	TSP	CVRP	OP	PCTSP	PDP
A2C	2.22	7.09	8.64	14.96	10.02
AM (Rollout)	1.41	5.30	<u>4.40</u>	<u>2.46</u>	<u>9.88</u>
POMO	<u>0.89</u>	**3.99**	14.26	11.61	10.64
Sym-NCO	**0.47**	<u>4.61</u>	**3.09**	**2.12**	**7.73**

Deep learning methods

Berto et al. (2025) RL4CO, https://arxiv.org/abs/2306.17100

Automated Algorithm Development

TSP

Vehicle Routing

Bin Packing

Scheduling

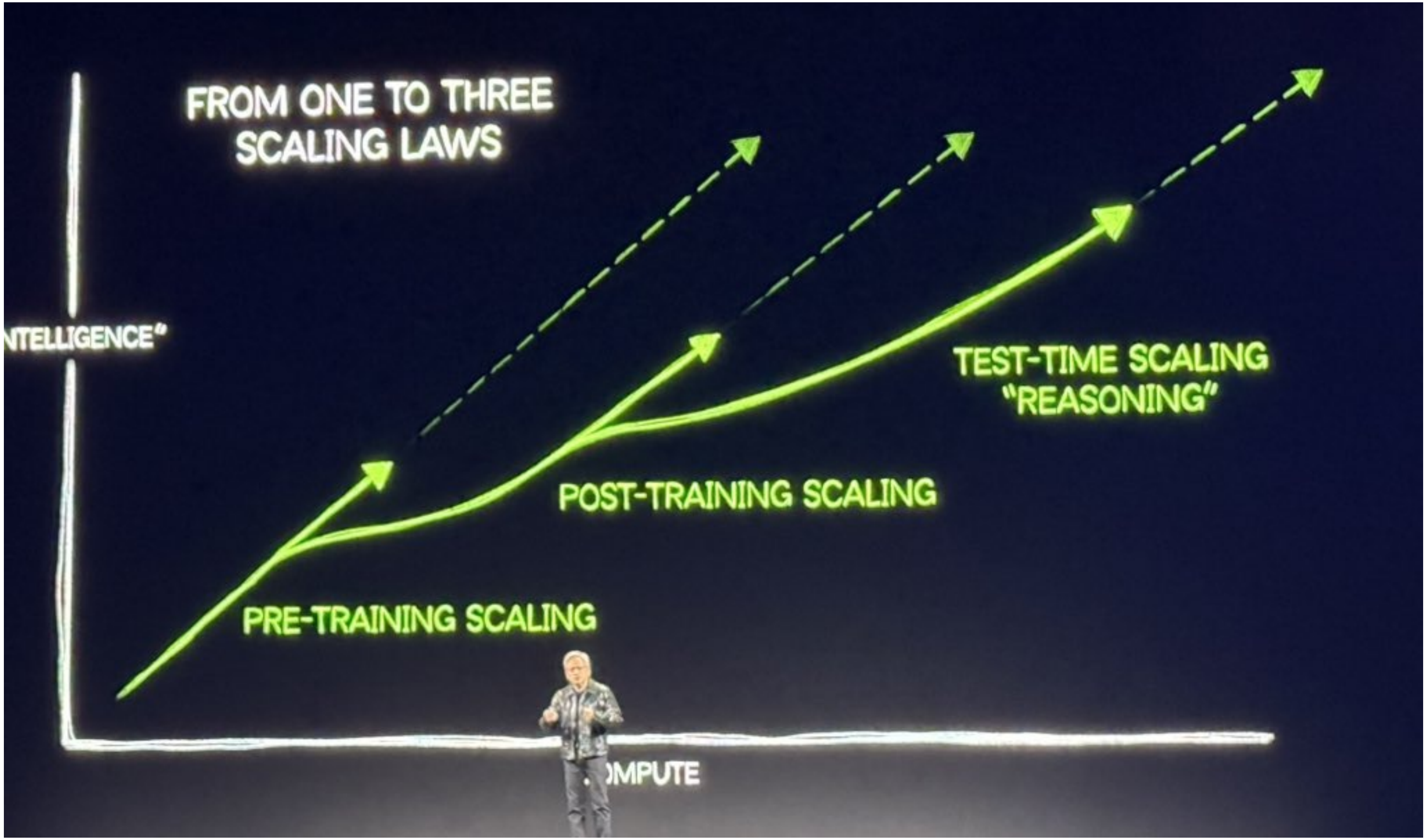
⋮

and their variants

A Foundational Neural Network Model For Combinatorial Optimization

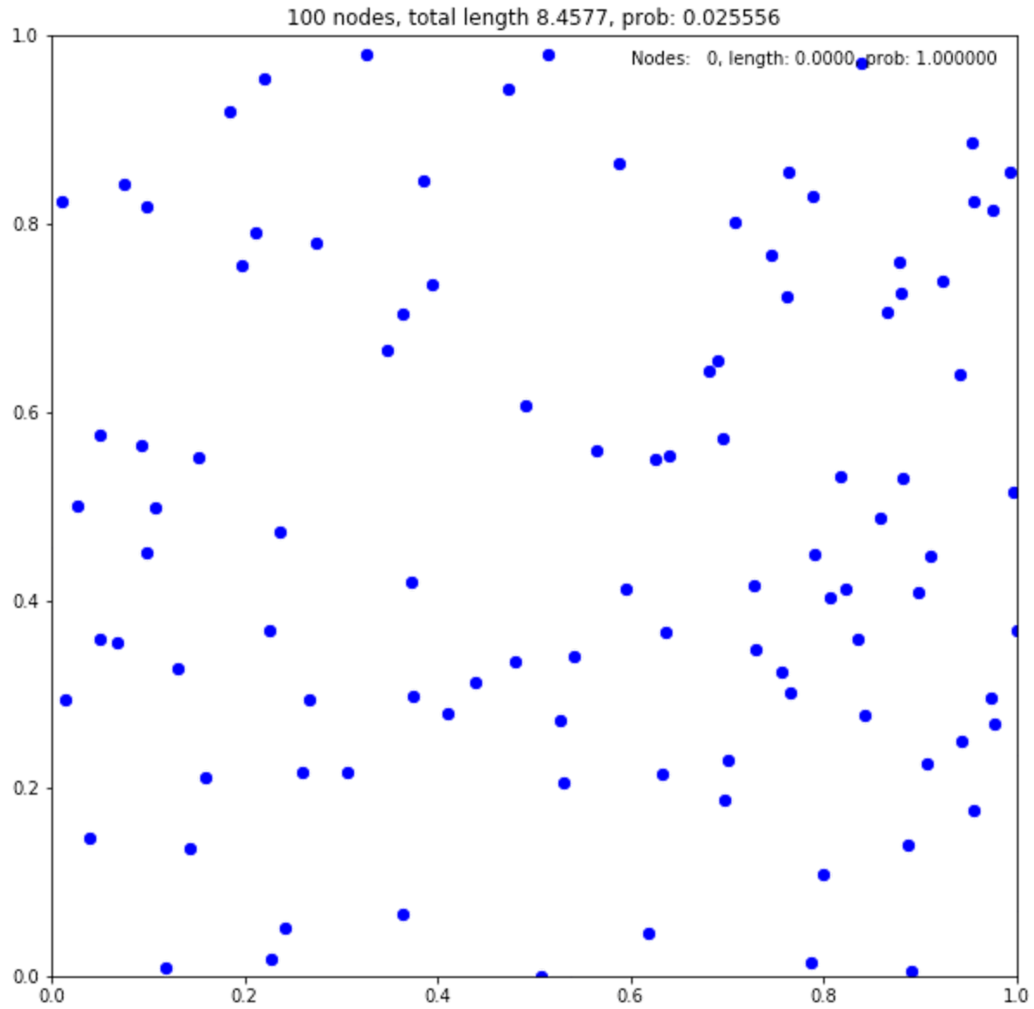
Automated Improving Search

Test-Time Search



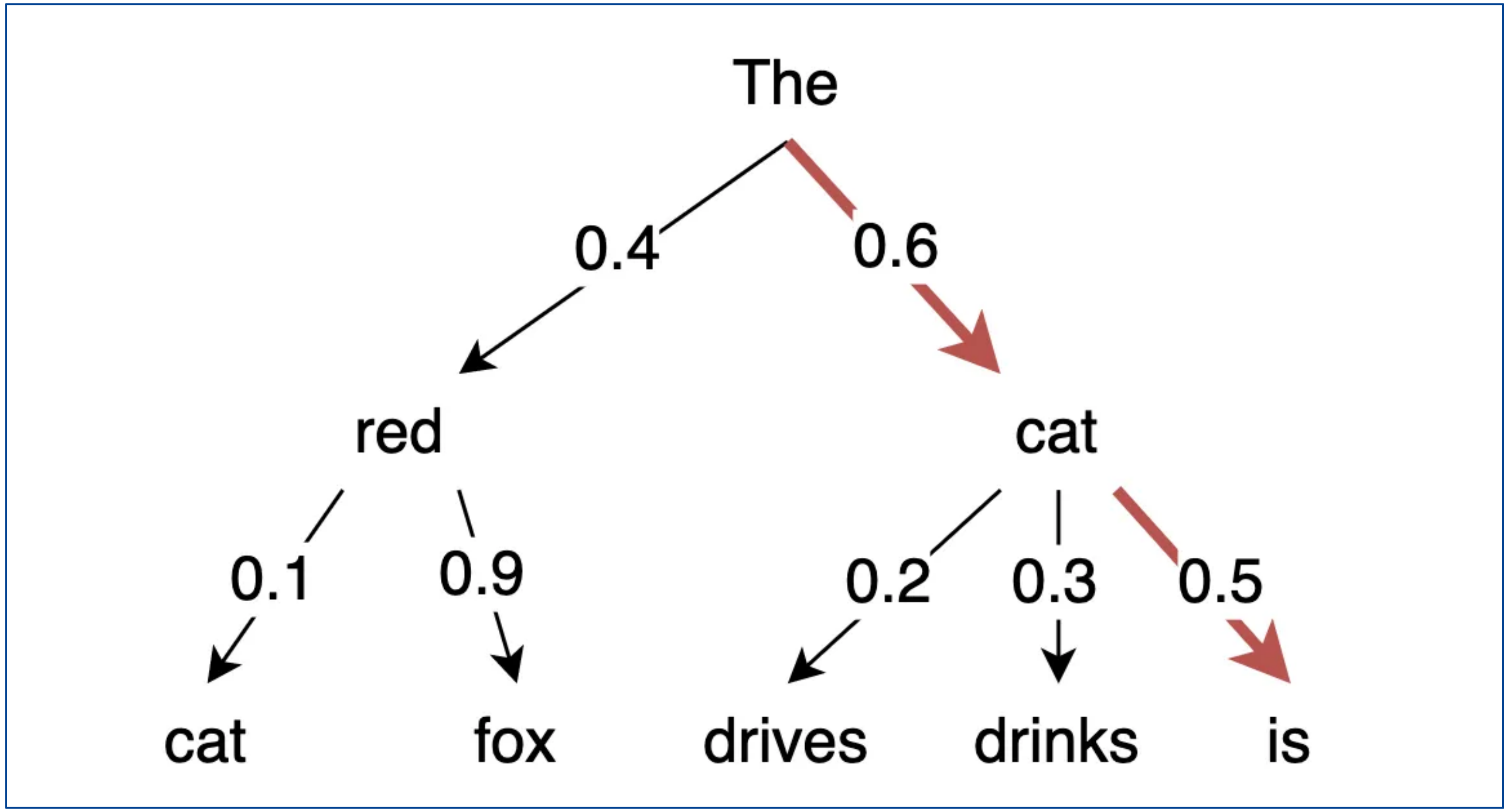
Attention Model (AM) for TSP

- Transformer-like encoder/decoder
- Kool et al. (2019) ICLR
- Each time, the node with the **highest probability** is chosen as the next city.



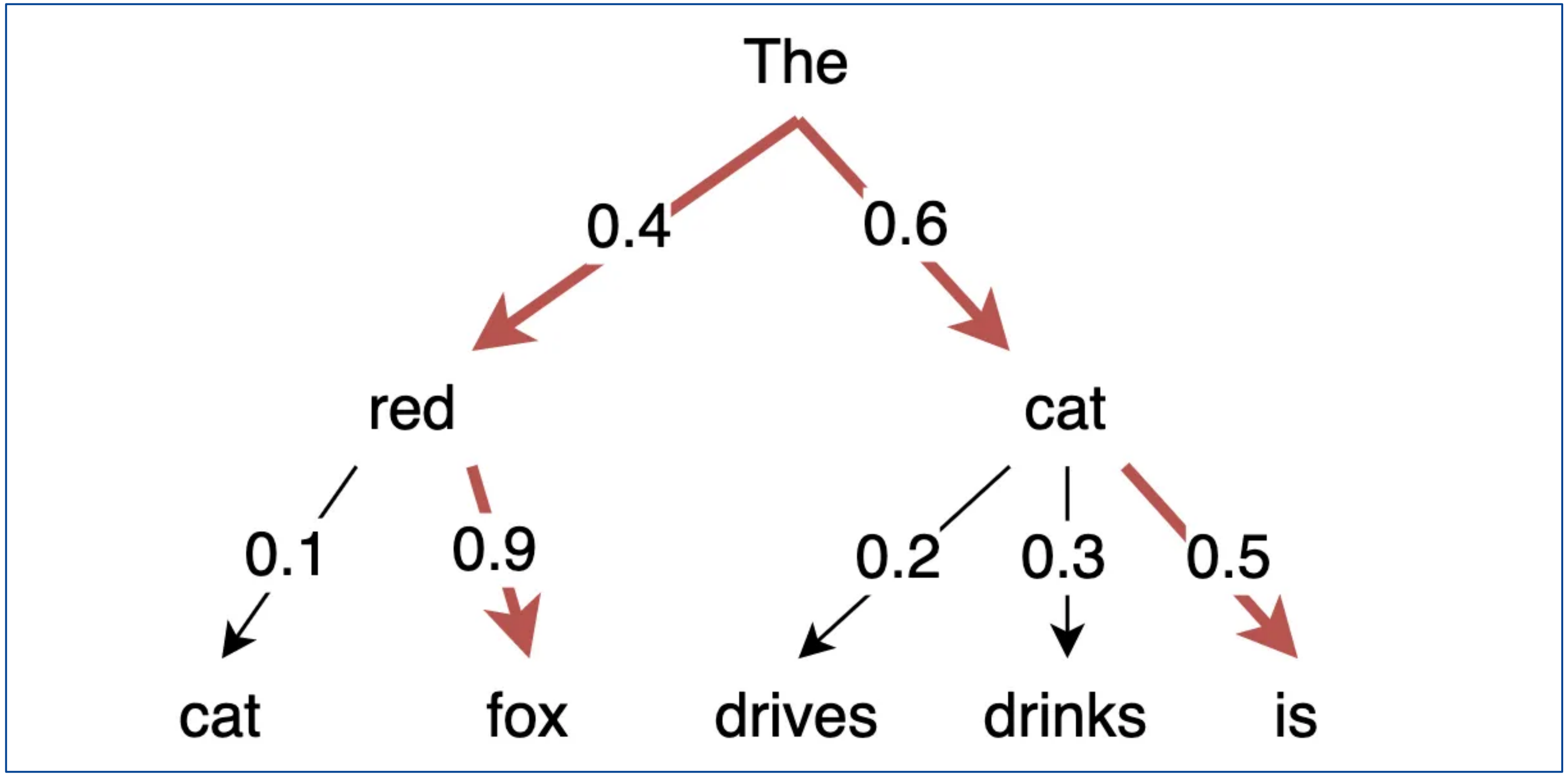
Decoding Strategies

- **Greedy Decoding**
- Beam Search
- Sampling
- Top-k
- Top-p



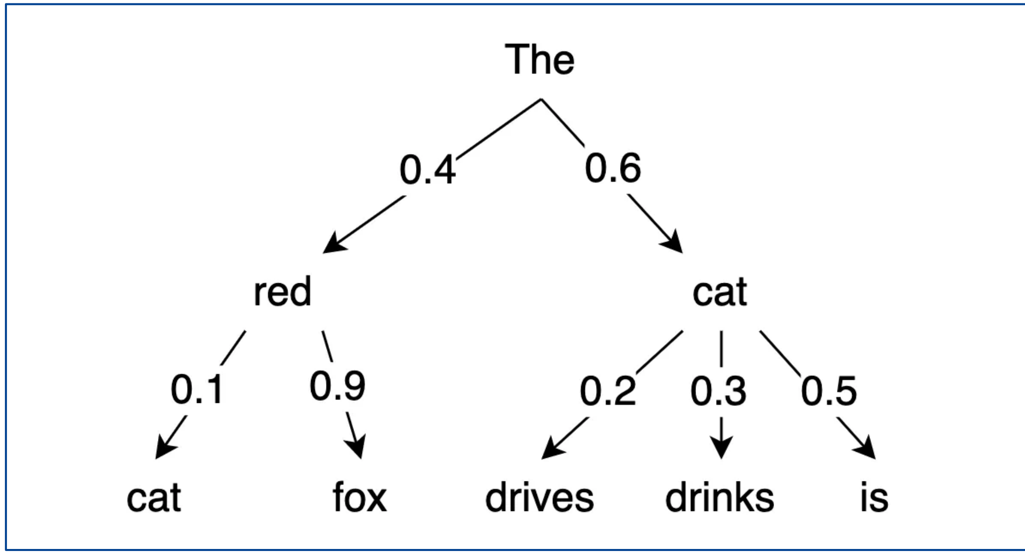
Decoding Strategies

- Greedy Decoding
- **Beam Search**
- Sampling
- Top-k
- Top-p



Beam size = 2

Decoding Strategies



- Greedy Decoding
- Beam Search
- **Sampling: the next one is chosen randomly among candidates**
  - Top-k sampling: among top k candidates
  - Top-p sampling: among top candidates whose total probabilities is no greater than p.

Neural Genetic Search





- Using the **exploration** power of **genetic search** methods **without** additional neural network training
- A **universally** applicable for various **discrete** optimization problems

Kim, Choi, Son, Park, Kwon (2025), Neural Genetic Search in Discrete Spaces, ICML 2025, https://arxiv.org/abs/2502.10433

Genetic algorithms (GA) for routing

- Genetic algorithms have been widely studied in routing

  e.g., EAX (for TSP), HGS (for CVRP), PyVRP, and Compass (for orienteering problem)

- Procedure
  - Step 1. Initialize populations
  - Step 2. Reproduce individuals (offspring)
    - Step 2-1. Select a mating pool
    - Step 2-2. Crossover
    - Step 2-3. Mutate
  - Step 3. Select the next population & repeat Step 2

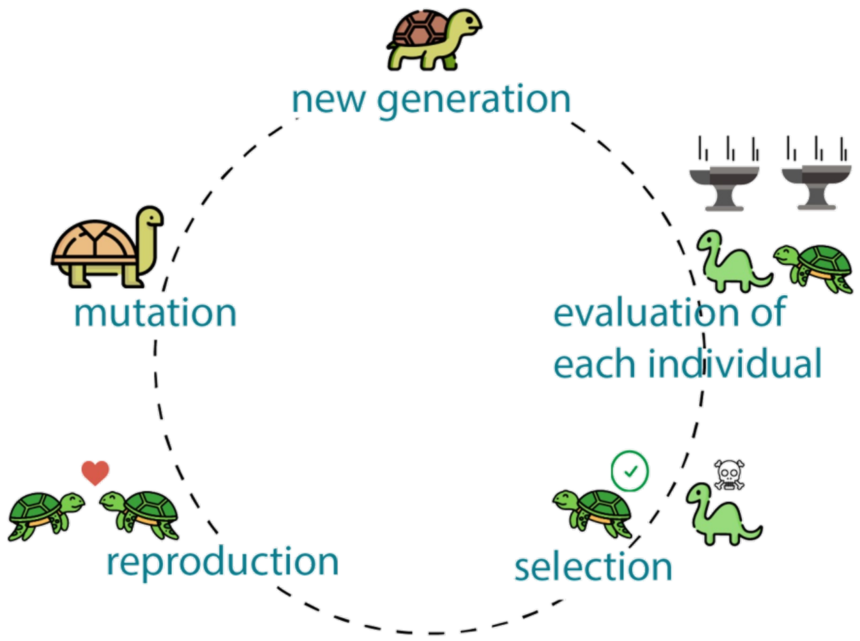
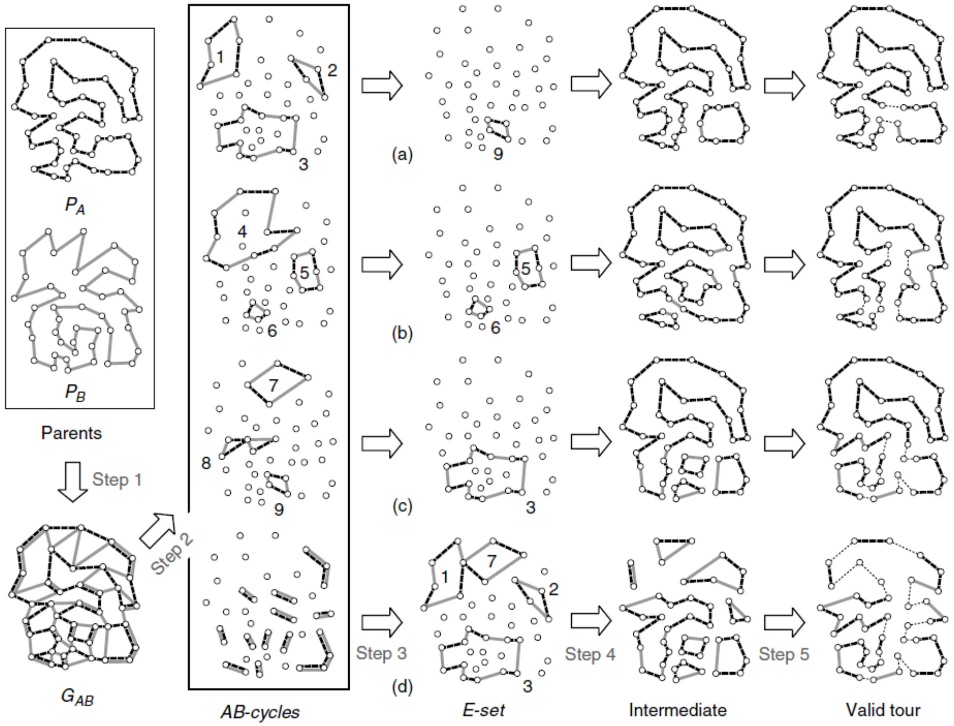


Image source: https://www.linkedin.com/pulse/genetic-algorithm-casting-gating-system-design-yokesh-kumar-d1vkc/

Problem-specific crossover operators

- Powerful, but the operators are *problem-specific* *and* *complicated*



Edge assemble crossover (EAX) for TSP ***(Nagata and Kobayashi, 2013)***

(Key idea) Replace the handcrafted design of crossover & mutation with deep learning

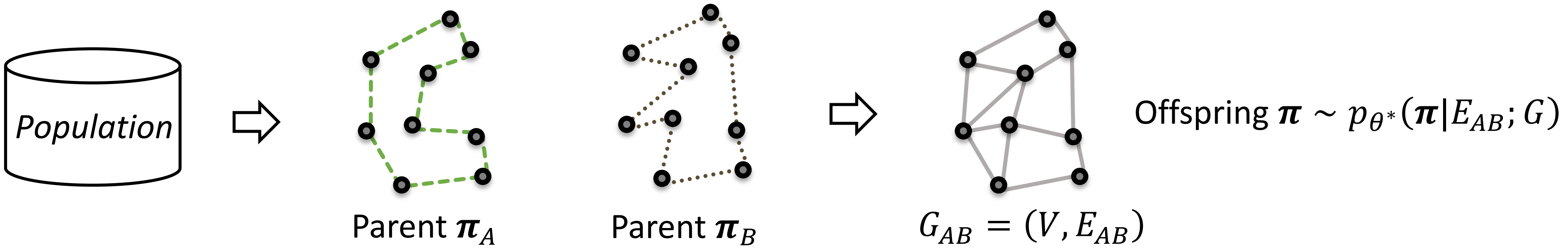
Our method: Neural Genetic Search

"Automated design of crossover & mutation with deep learning"

- Assume that we have **the trained neural network** p_{θ^*}
  - We can flexibly choose any constructive deep learning method for base
- Rethinking crossover and mutation
  - Meaning of "crossover": to generate new samples by recombining parents

  $\Rightarrow$ Generate new samples conditioned on parents

Parent-Conditioned Generation



- How can we sample from $p_{\theta*}(\boldsymbol{\pi}|\boldsymbol{\pi}_A,\boldsymbol{\pi}_B,G)$?
  – Similar to EAX, define the overlapped edge set $E_{AB} = E_A \cup E_B$
  – E_A and E_B are sets of edges consisting the parent tours $\boldsymbol{\pi}_A$ and $\boldsymbol{\pi}_B$
- An offspring tour consists of edges in E_{AB}

  ⇒ an additional masking rule for offspring generation

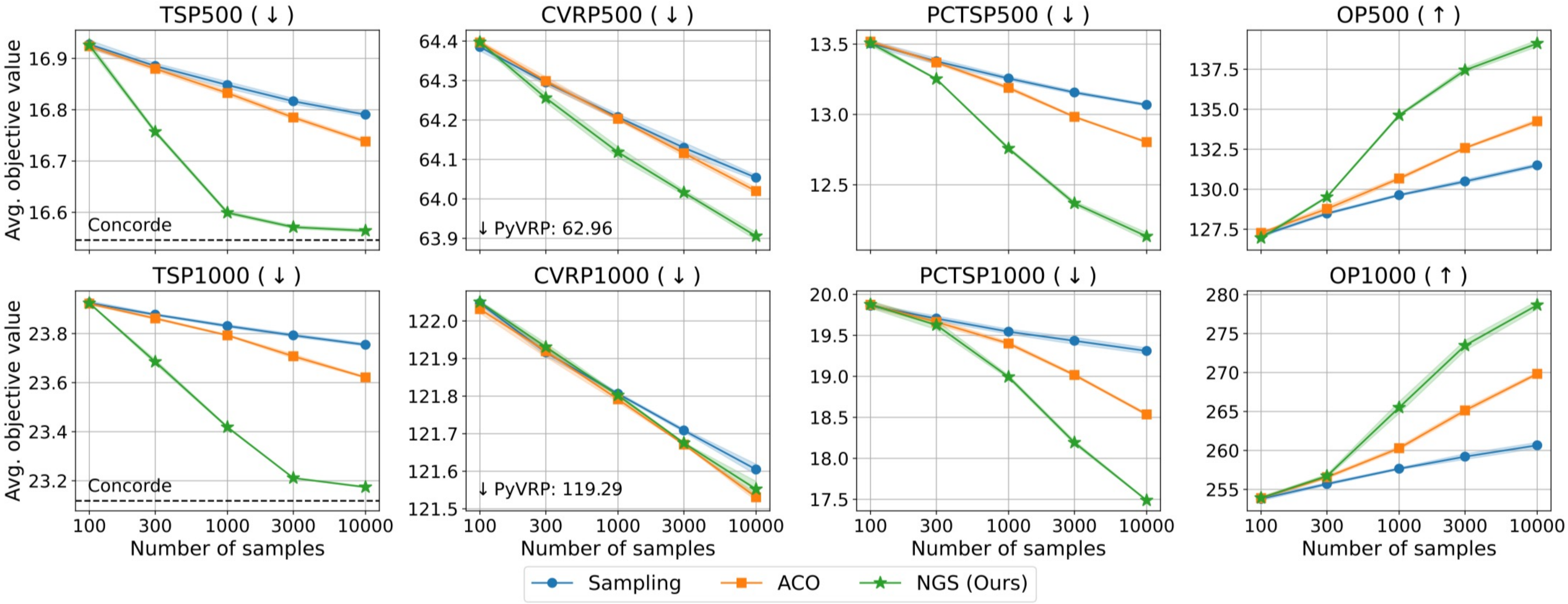
NGS outperforms other search methods

		$N = 200$		$N = 500$	
		Gap (%)	Time	Gap (%)	Time
TSP	Concorde	-	1s	-	10s
	LKH3 (Helsgaun, 2017)	0.001	10s	0.022	32s
	GNN (Kim et al., 2025)				
	+ Sampling	0.307	2s	1.827	10s
	+ BS ($w = 1000$)	1.378	4s	20.637	18s
	+ MCTS	0.164	20s	1.324	60s
	+ ACO	0.294	5s	1.733	17s
	+ NGS (Ours)	0.028	5s	0.322	17s
	+ Sampling (long)	0.130	16s	1.479	101s
	+ MCTS (long)	0.126	100s	1.268	300s
	+ ACO (long)	0.102	47s	1.162	167s
	+ NGS (Ours, long)	**0.011**	50s	**0.110**	170s

NGS outperforms other search methods

		$N = 200$		$N = 500$	
		Gap (%)	Time	Gap (%)	Time
CVRP	PyVRP	-	4.0m	-	21.0m
	LKH3 (Helsgaun, 2017)	0.304	2.4m	0.182	18.6m
	GNN (Kim et al., 2025)				
	+ Sampling	1.487	4s	1.982	7s
	+ BS ($w = 1000$)	2.659	4s	2.624	7s
	+ ACO	1.485	7s	1.975	14s
	+ NGS (Ours)	0.981	8s	1.840	15s
	+ Sampling (long)	1.104	0.7m	1.738	1.2m
	+ ACO (long)	1.055	1.2m	1.684	2.3m
	+ NGS (Ours, long)	**0.126**	1.3m	**1.502**	2.5m

NGS outperforms other search methods

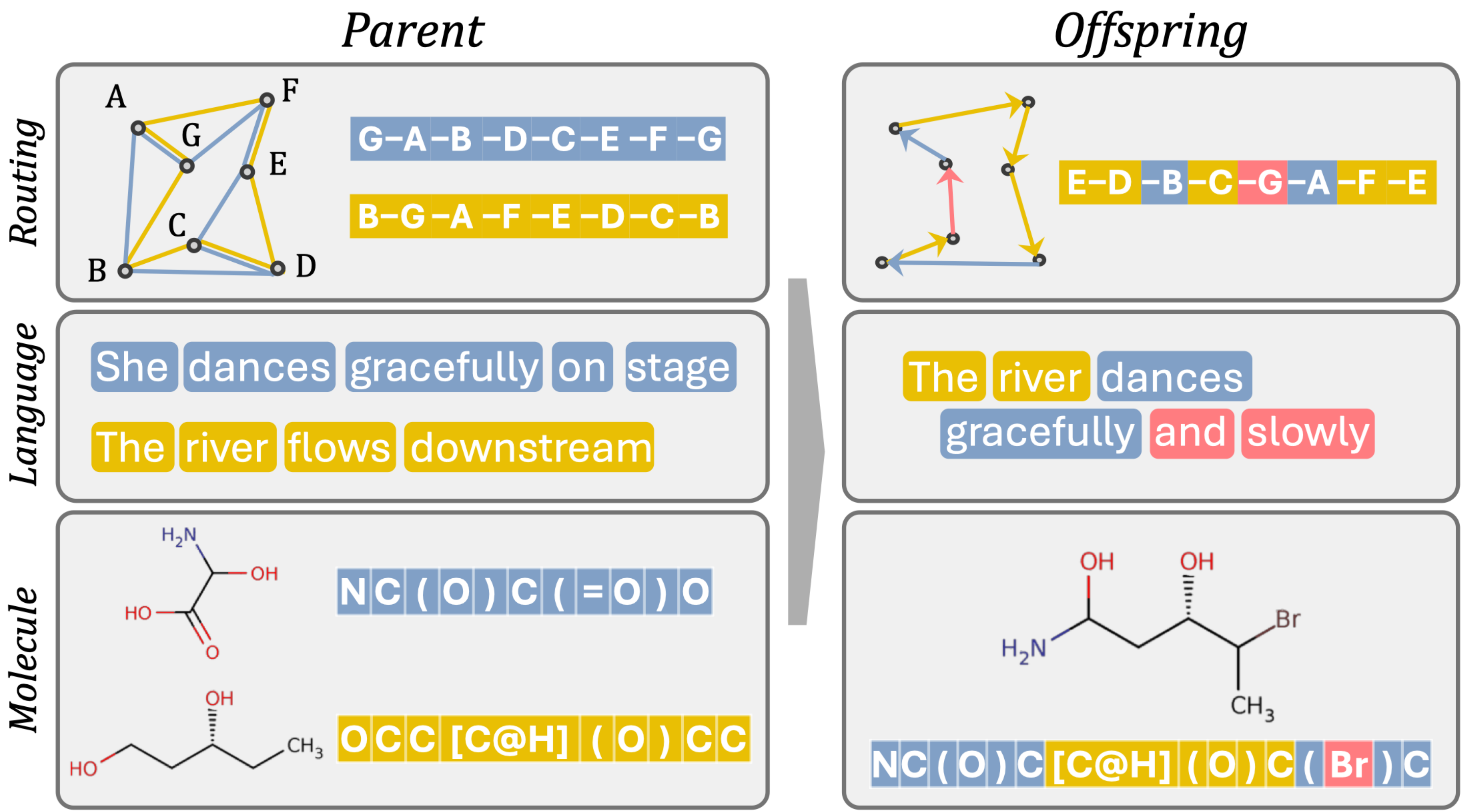


NGS applied in other domains

- **Requirements:**
  1. A trained neural network model to solve your problem
  2. Tokenizerable solutions

- In routing problems, **a token = an edge**.

Applications of Neural Genetic Search



Routing Problems

Language Models

Molecule Design

Exact Mathematical Optimization

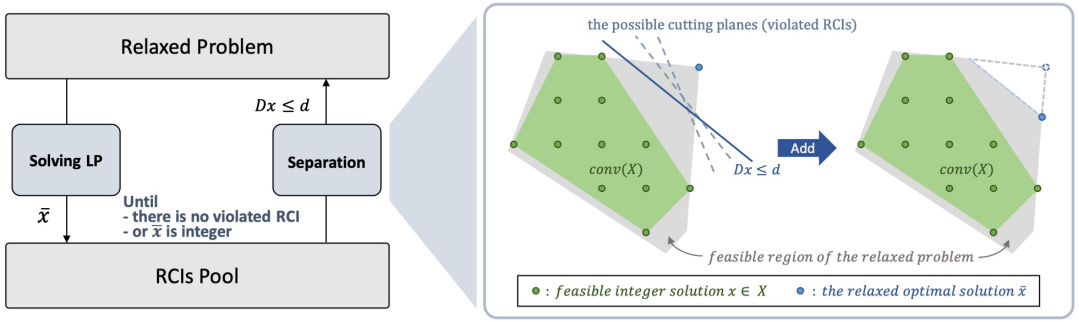
with Learning

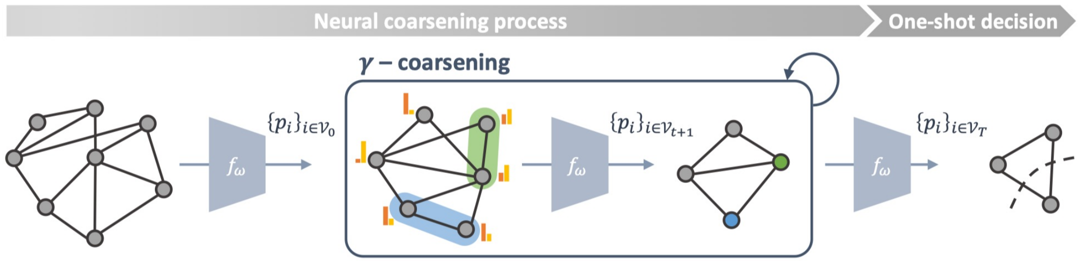
Learning-Optimization Hybrid Method

- For an optimization algorithm, replace a **human-designed component** with a **learning model**
  1. Heuristics + Learning Model
  2. Meta-heuristics + Learning Model
  3. Exact algorithms + Learning Model

3. Exact Method + Learning

- Modern MIP Optimization Solver: **Branch-and-Cut**
  - **How to Branch?** (Branch-and-Bound Tree Exploration)
  - **How to Cut?** (Cutting Plane)

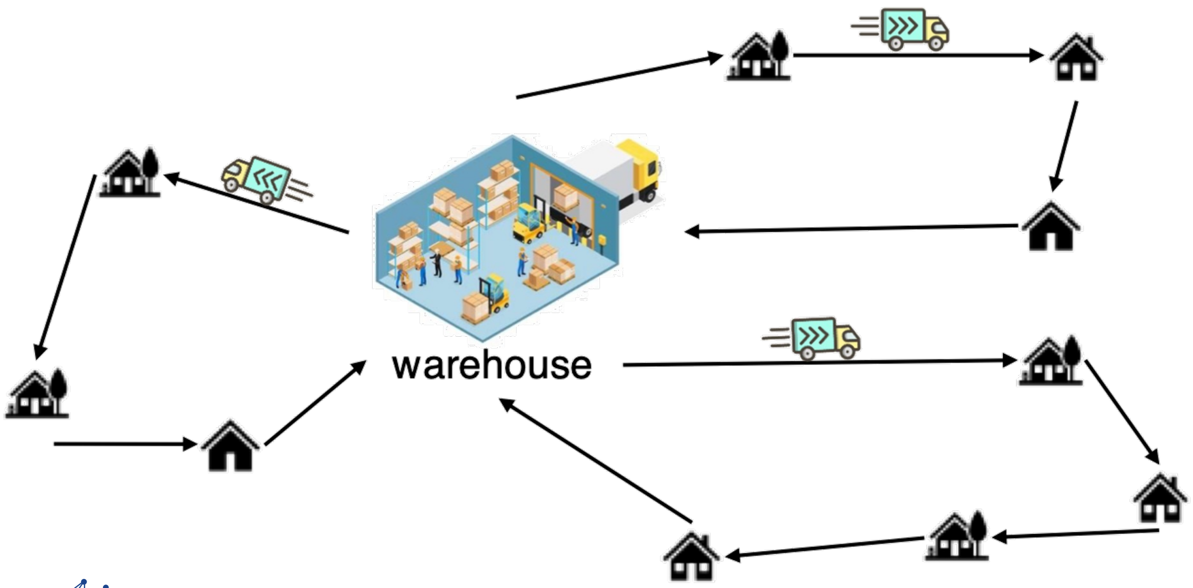




Kim, Park, Kwon (2024) A Neural Separation Algorithm for the Rounded Capacity Inequalities, *INFORMS Journal on Computing*, 36(4), 987–1005. https://doi.org/10.1287/ijoc.2022.0310

Capacitated VRP (CVRP)

- Given a set of customers, CVRP determines
- Which customer is served by which vehicle
- In what order
- To minimize the total distance traveled



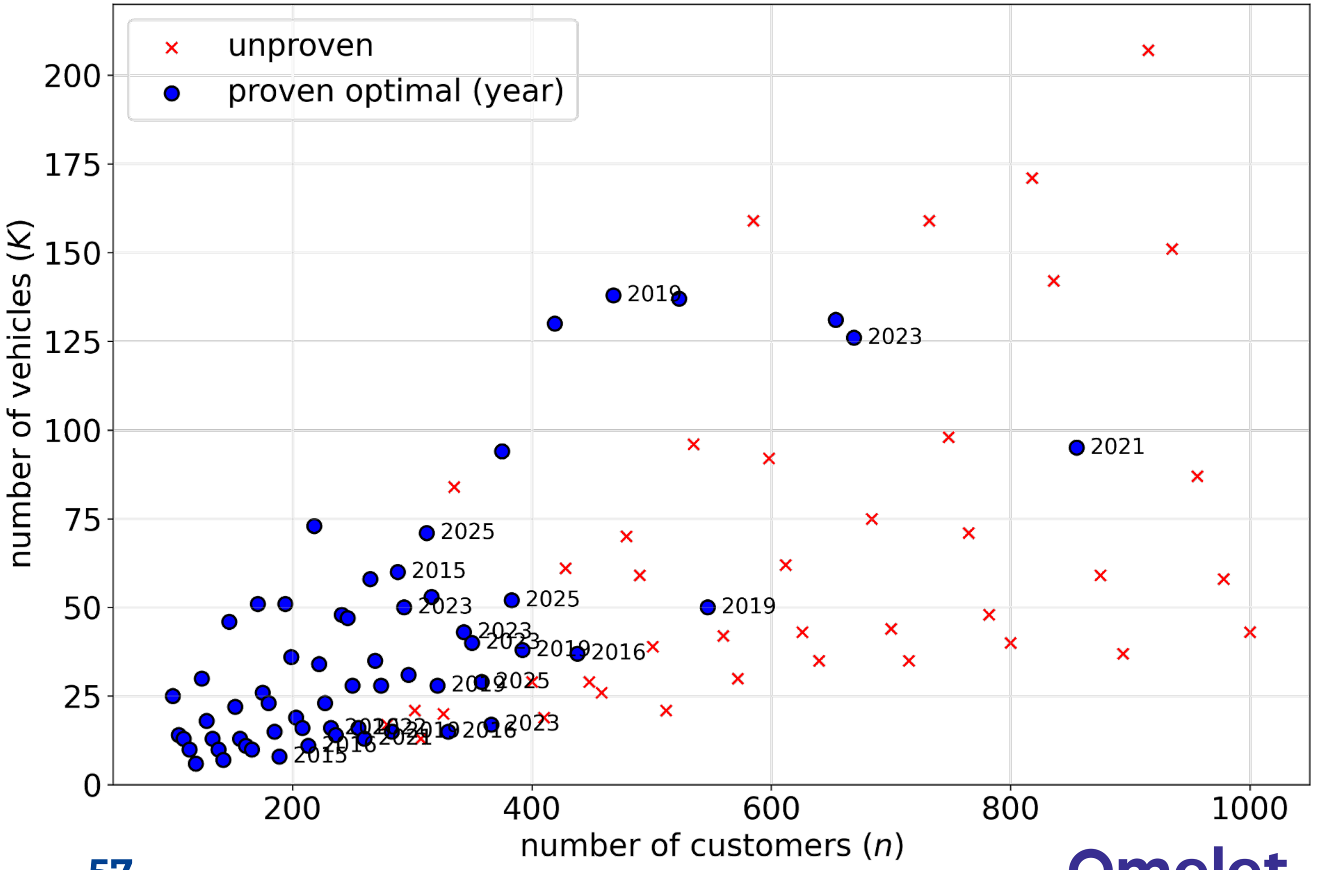
Current State

- Can we find provably optimal solutions to CVRP?

- $n < 100$, early 2000's
  - Branch-and-Cut

- $n < 300$, in 2010's
  - Branch-Cut-and-Price

(X-instances, as of October 10, 2025)



Traveling Salesman Problem (TSP)

- Exact solutions
  - 80 cities – mid 1960s
  - 318 cities – 1980
  - 666 cities – 1987
  - 2,392 cities – 1987
  - 3,038 cities – 1989
  - 13,509 cities – 1998
  - 15,112 cities – 2001
  - 24,979 cities – 2004
  - 85,900 cities – 2005 to 2006

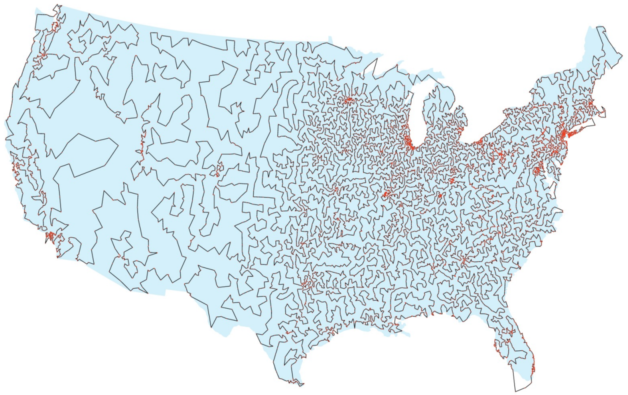
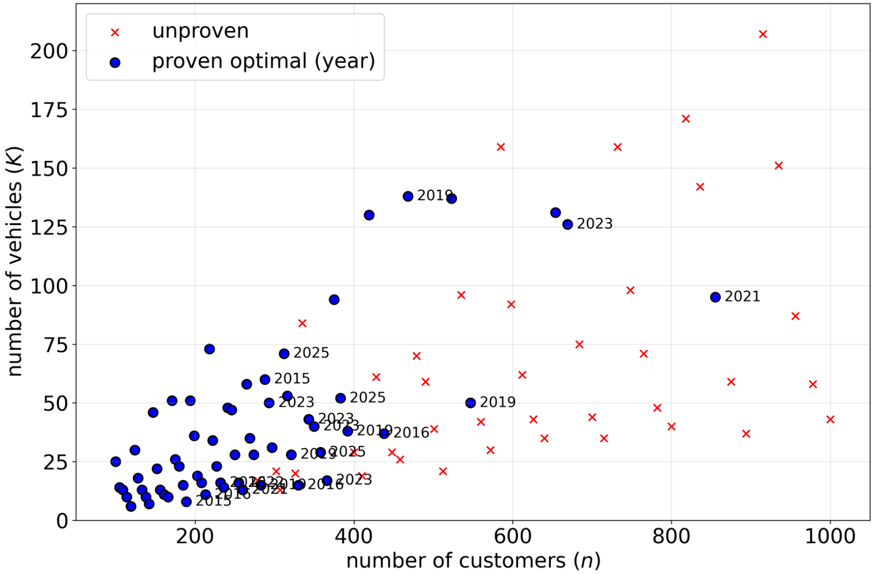


Figure 8.8
A tour of all 13,509 USA cities having population of at least 500.

1,000 cities: solved in about 30 seconds in my laptop

Branch-and-Bound
Cutting Plane Method
Separation Algorithms
Heuristic Algorithms

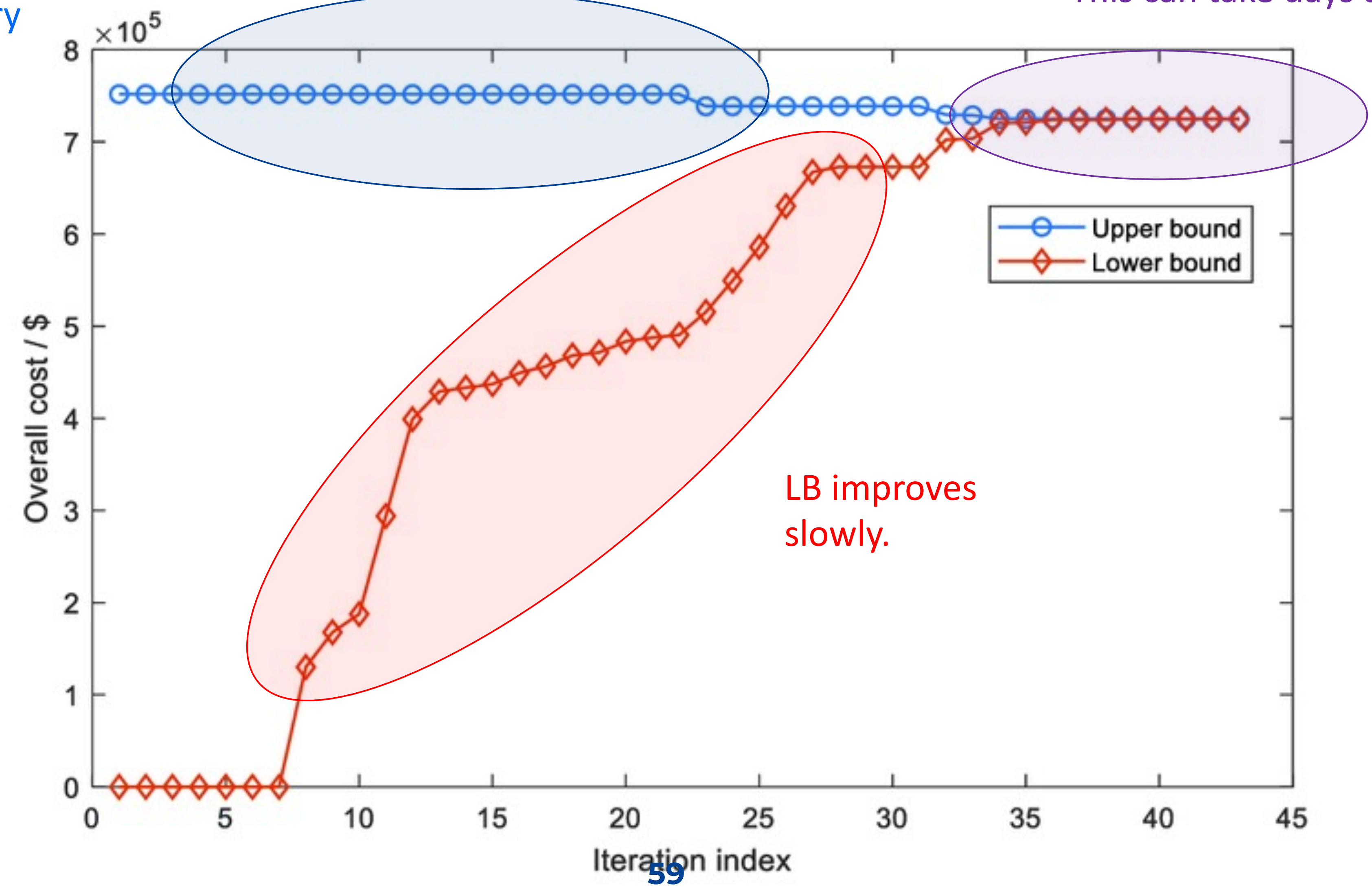
CVRP



Provably Optimal Solution

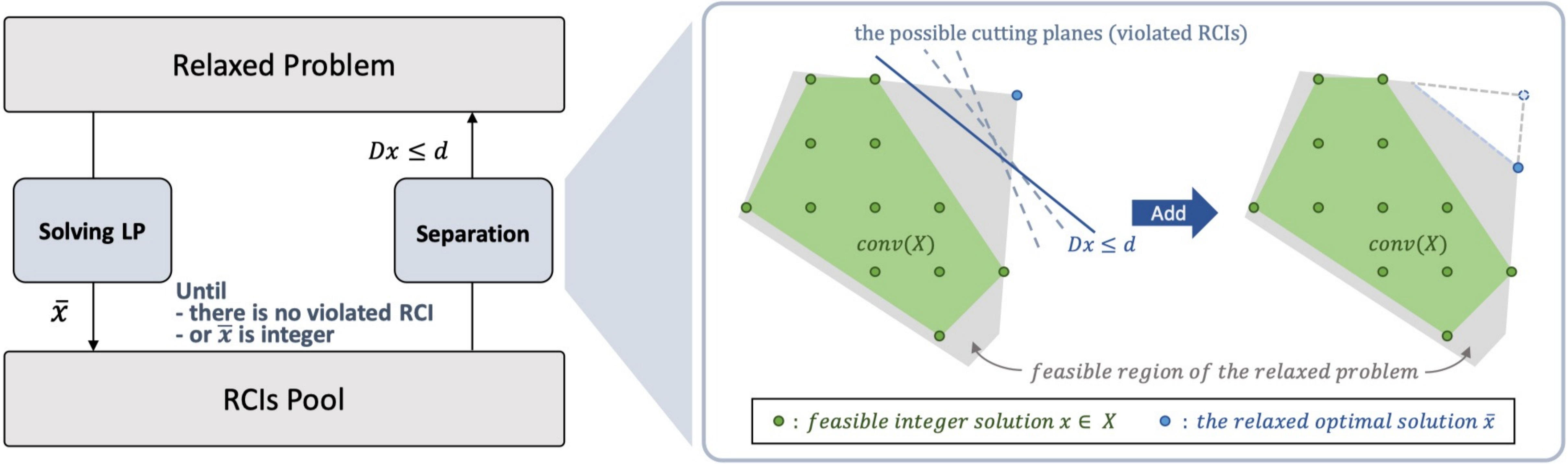
A good UB is found very quickly usually.

If UB meets LB, then it proved optimal. This can take days and years.



LB improves slowly.

Cutting Plane Algorithm



- How to separate fractional solutions and generate new cutting planes?

Rounded Capacity Inequalities (RCI)

- IP Formulation:

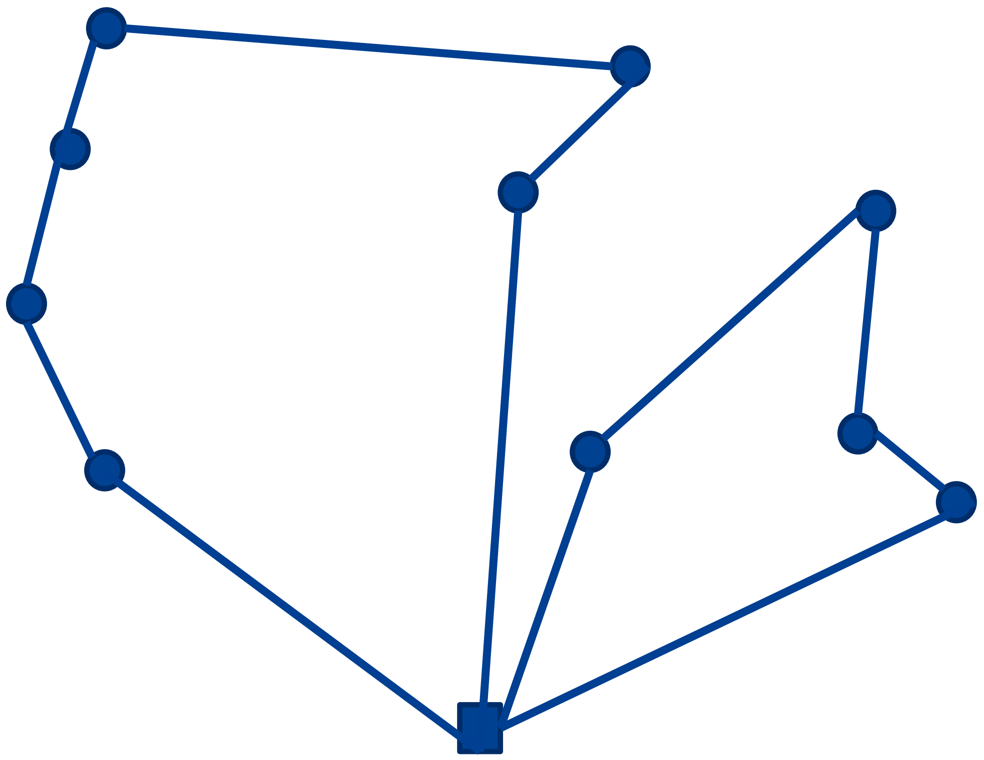
$$
\begin{aligned}
&\text{minimize} && \sum_{(i,j)\in E} c_{ij}x_{ij} && && (1)\\
&\text{subject to} && x(\delta(\{i\}))=2 && \forall i\in V_C && (2)\\
& && x(\delta(S))\geq 2b(S) && \forall S\subseteq V_C && (3)\\
& && x_{ij}\in\{0,1\} && \forall 1\leq i<j\leq |V| && (4)\\
& && x_{0j}\in\{0,1,2\} && \forall j\in V_C, && (5)
\end{aligned}
$$

$$x(\delta(S)) = \sum_{e\in\delta(S)} x_e \geq 2\left\lceil \frac{\sum_{i\in S} d_i}{Q} \right\rceil$$

Rounded Capacity Inequality (RCI) replaces (3)

RCI Example

- Suppose all demands are 1, and the vehicle capacity = 6.

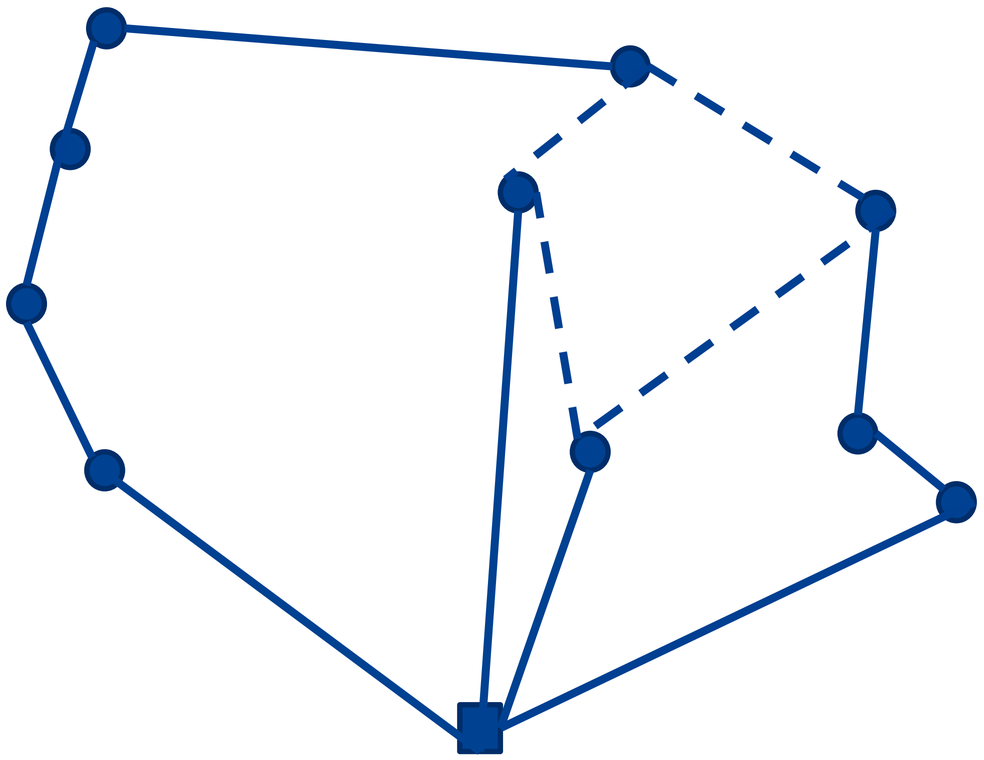


A feasible solution is given.

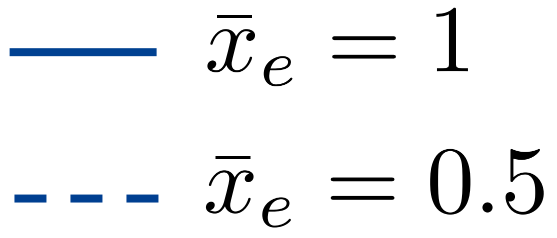
RCI Example How to find S for which $\bar{x}$ violates the RCI?

- Suppose all demands are 1, and the vehicle capacity = 6.

$\bar{x}$

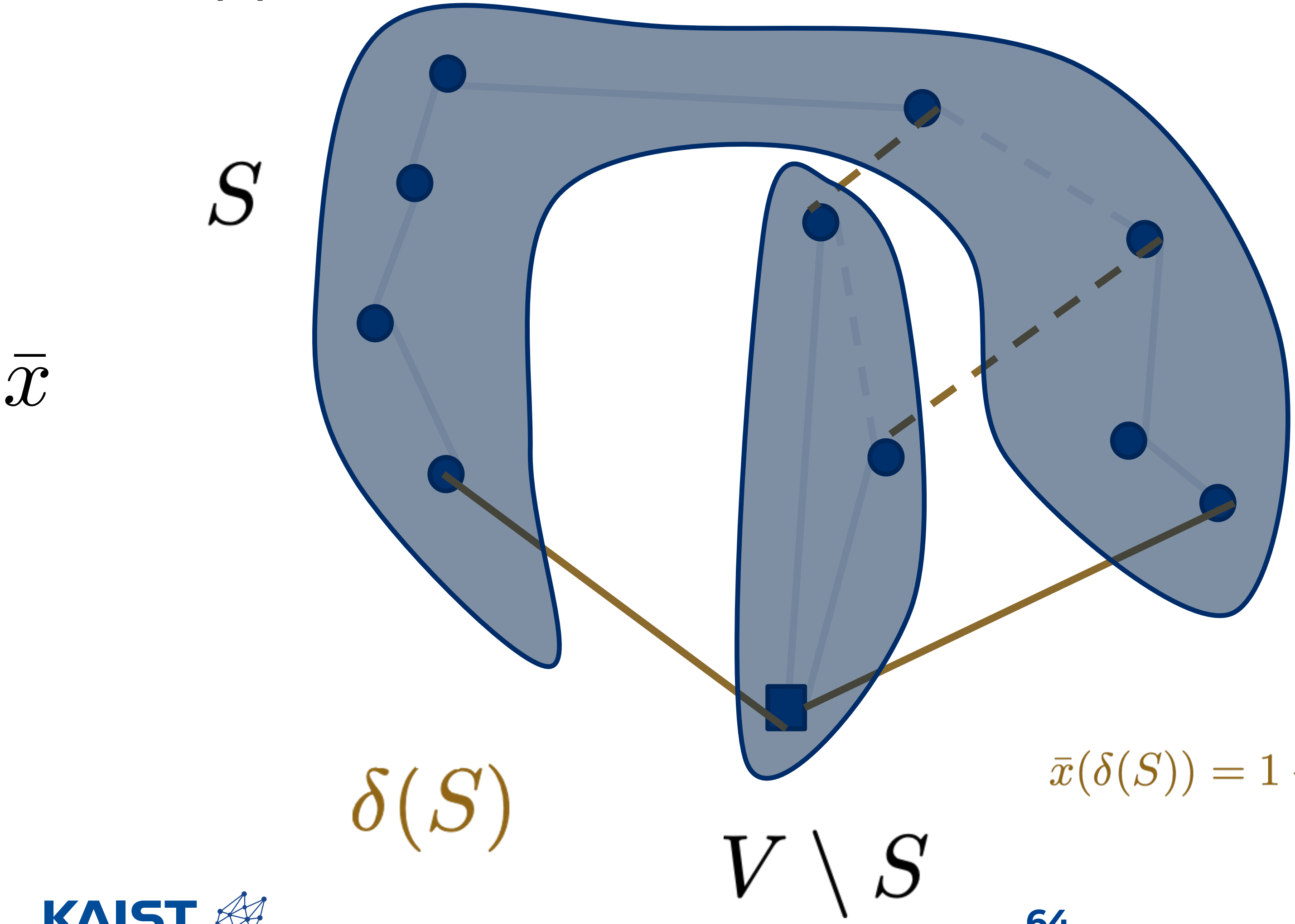


$$x(\delta(S)) = \sum_{e \in \delta(S)} x_e \geq 2 \left\lceil \frac{\sum_{i \in S} d_i}{Q} \right\rceil$$



RCI Example How to find S for which $\bar{x}$ violates the RCI?

- Suppose all demands are 1, and the vehicle capacity = 6.



$$x(\delta(S)) = \sum_{e \in \delta(S)} x_e \geq 2 \left\lceil \frac{\sum_{i \in S} d_i}{Q} \right\rceil$$

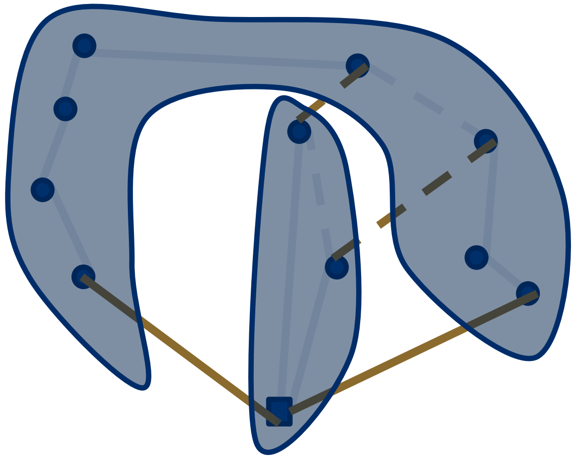
— $\bar{x}_e = 1$

- - - $\bar{x}_e = 0.5$

$$\bar{x}(\delta(S)) = 1 + 1 + 0.5 + 0.5 = 3 \not\geq 2 \left\lceil \frac{\sum_{i \in S} d_i}{Q} \right\rceil = 2 \left\lceil \frac{8}{6} \right\rceil = 4$$

NeuralSEP

- How to determine a subset S that violates the RCI?



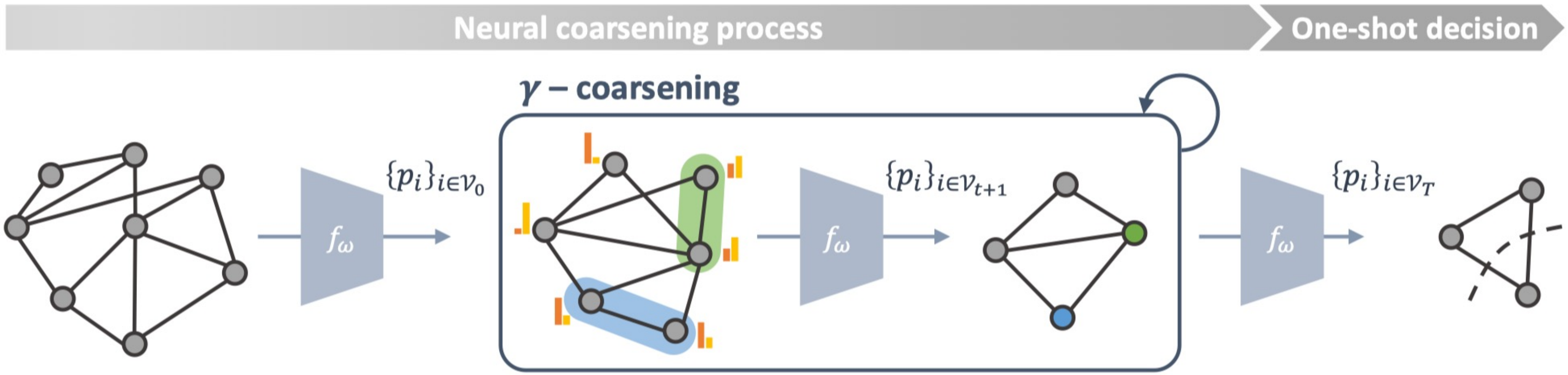
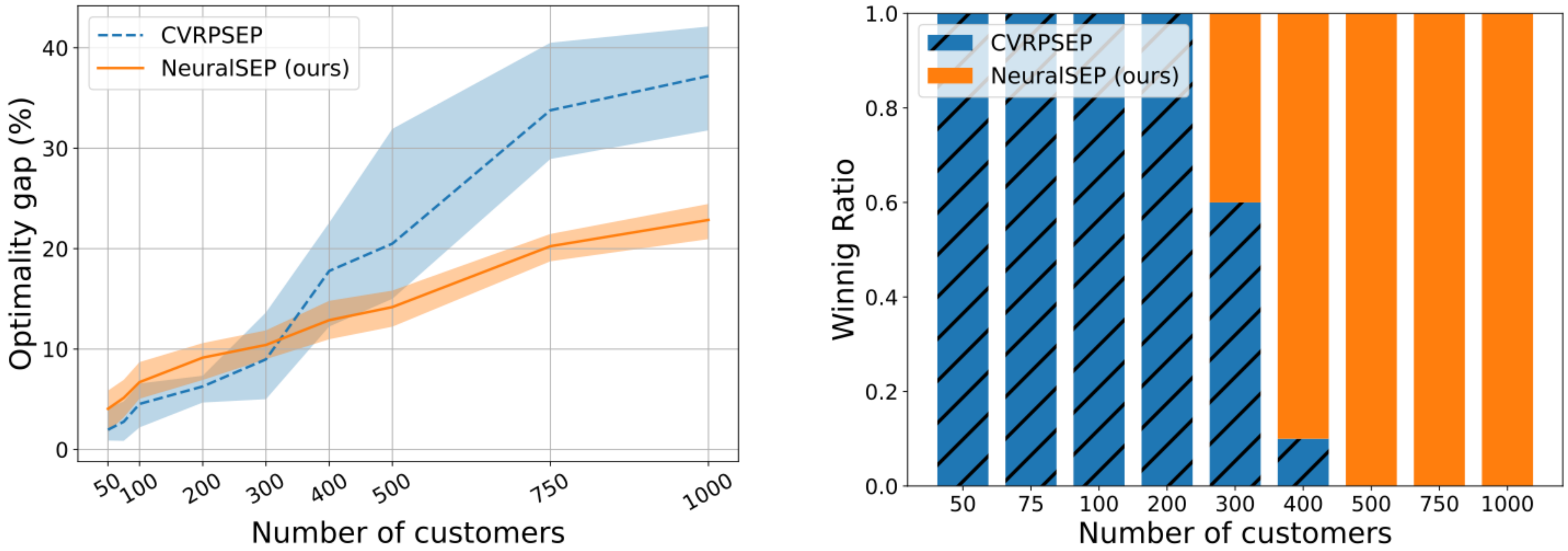


Figure 2 **The overall coarsening procedure of NeuralSEP: It iteratively predicts vertex selection probabilities and coarsens the given graph. NeuralSEP decides the set assignment on the coarsest graph, then maps the assignment to the original graph.**

Results



(a) The average and range of the optimality gap.

(b) Winning ratio out of 10 instances.

Figure 4: The results of the cutting plane method with CVRPSEP and NeuralSEP.

Results

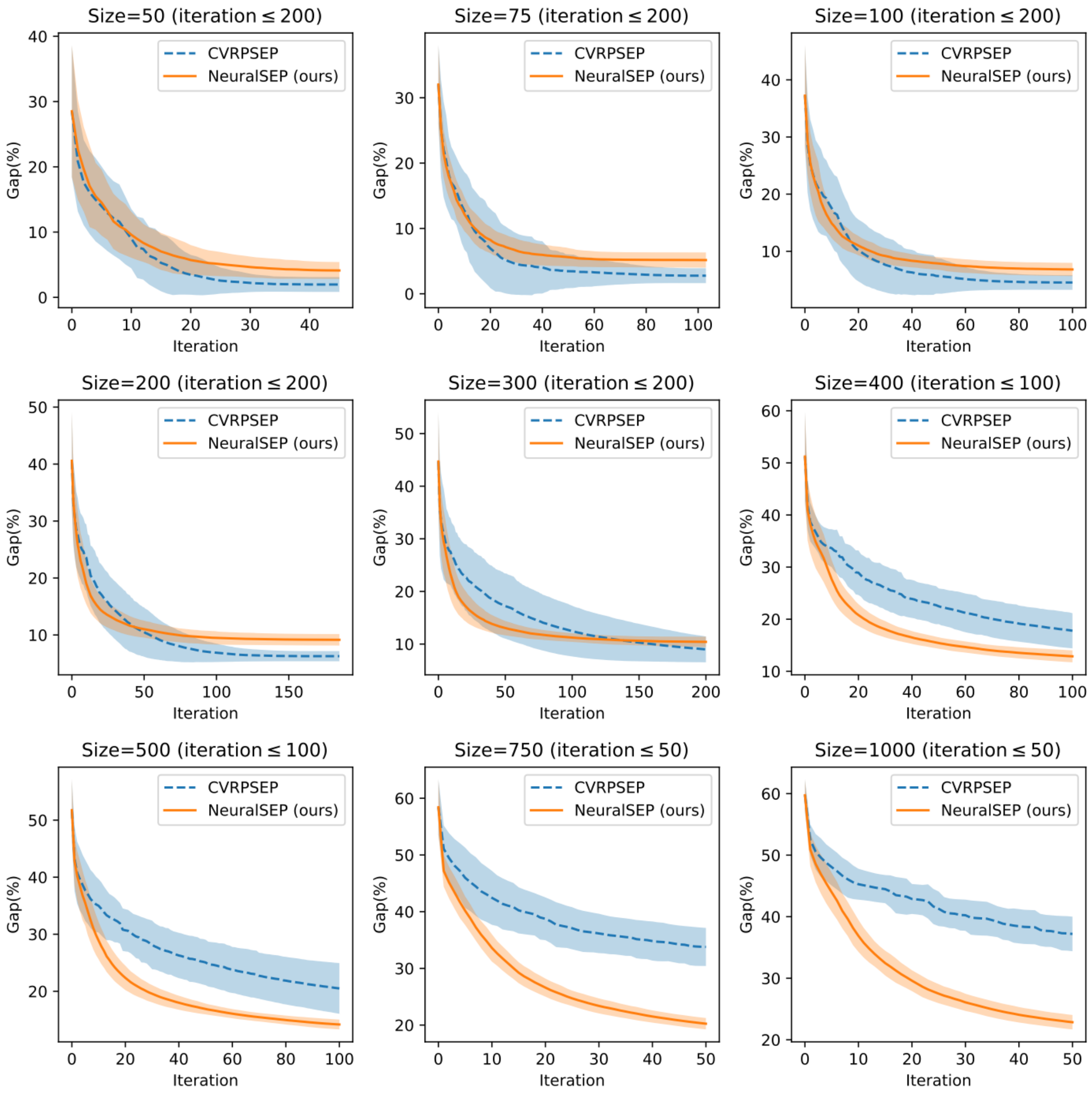
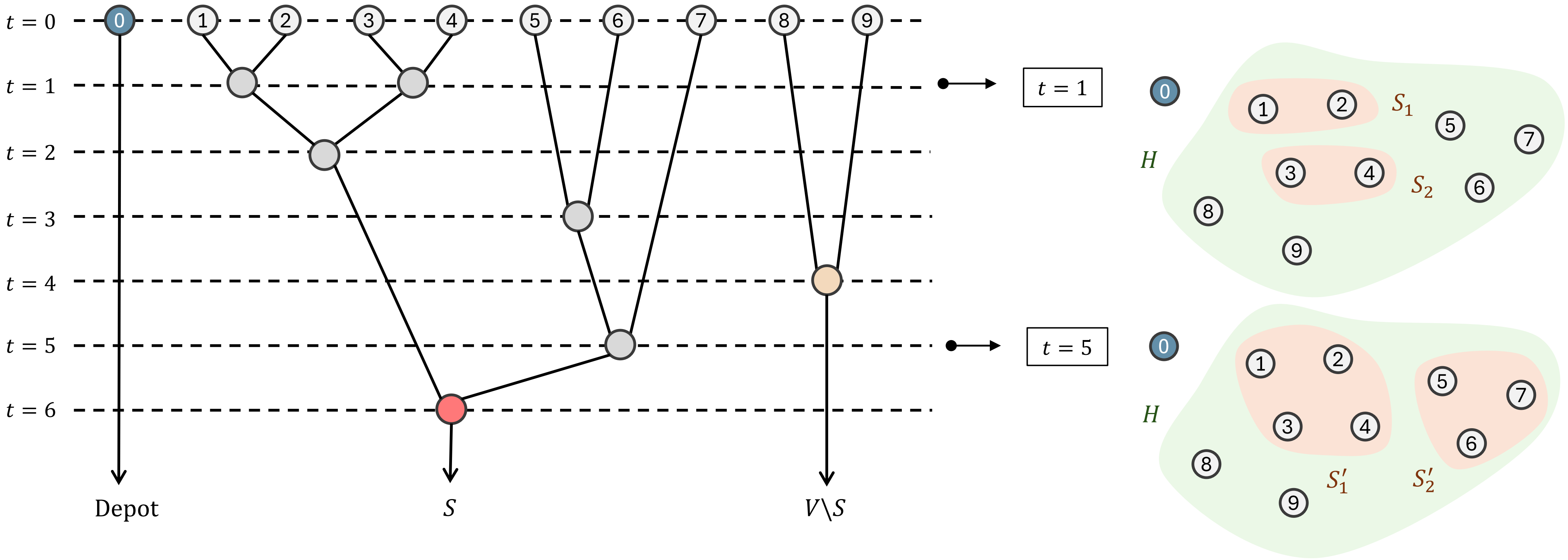


Figure 5: The optimality gap improvement according to iterations. The solid line in the plot denotes the average optimality gap of 10 instances, while the shaded area illustrates the range of the optimality gap.

Test-Time Search



- GraphCHiP: Graph Coarsening History-based Partitioning

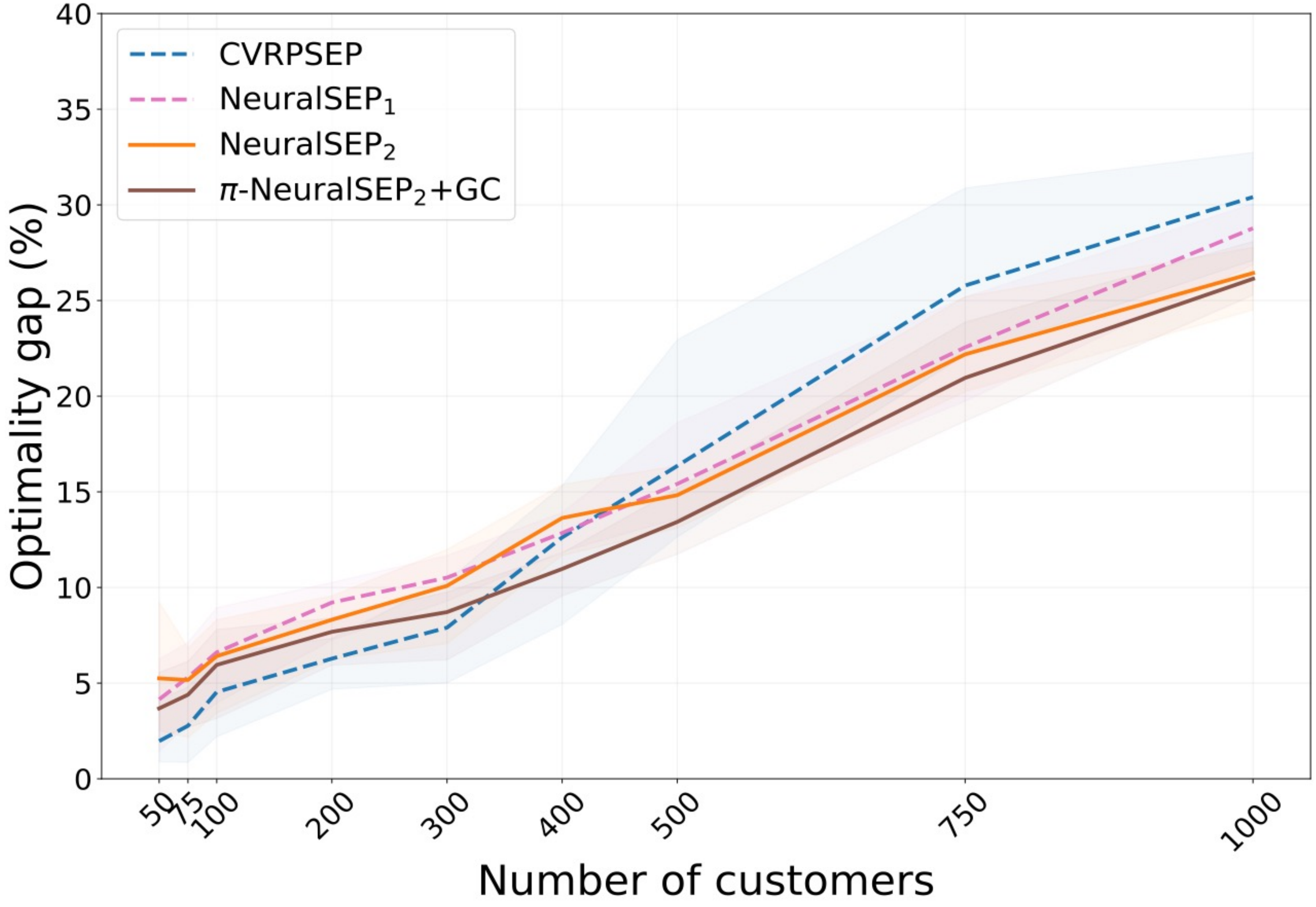


Sim, Kim, Kwon (2025) Test-Time Search in Neural Graph Coarsening Procedures for the Capacitated Vehicle Routing Problems, https://arxiv.org/abs/2510.00958

KAIST

Omelet

Results



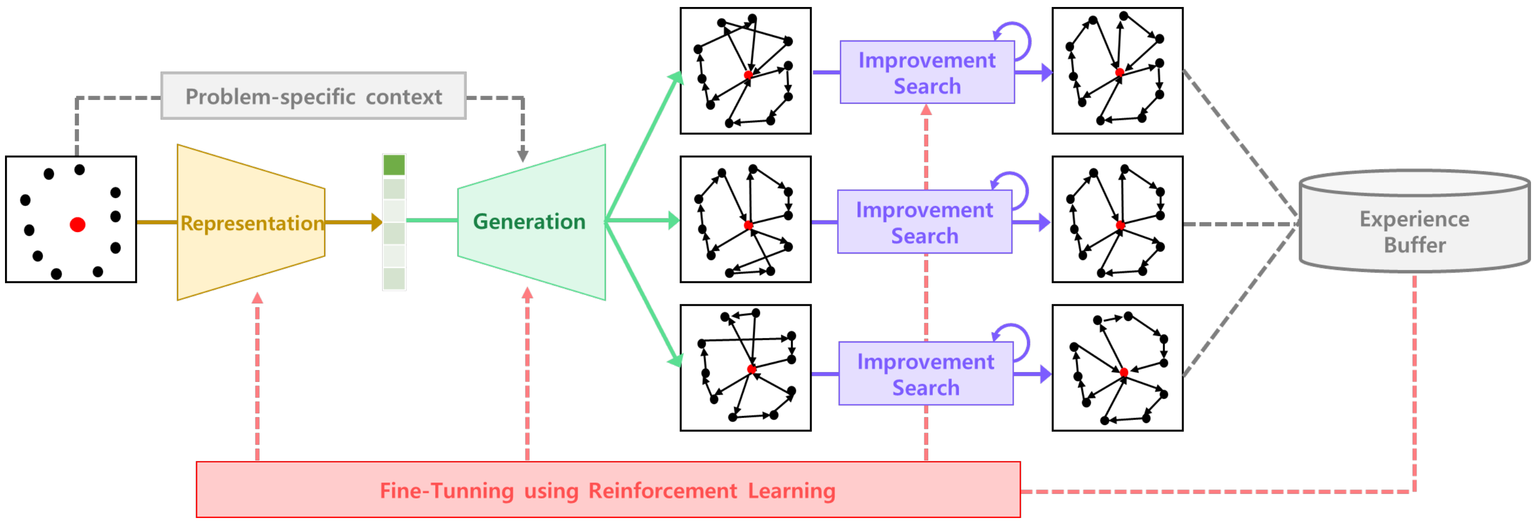
CVRPSEP
NeuralSEP1
NeuralSEP2
π-NeuralSEP2+GC
Optimality gap (%)
0
5
10
15
20
25
30
35
40
50
75
100
200
300
400
500
750
1000
Number of customers

Topics Covered

- For an optimization algorithm, replace a **human-designed component** with a **learning model**
  1. Heuristics + Learning Model
  2. Meta-heuristics + Learning Model
  3. Exact algorithms + Learning Model

+ Test-Time Search

- **Deep Learning Foundation Model for Combinatorial Optimization and Its Applications to Various Industrial Problems**




Jinkyoo Park
KAIST Omelet


Changhyun Kwon
KAIST Omelet


Woonam Hwang
KAIST


Sungbin Lim
KOREA UNIVERSITY

Thank you!

Automated Algorithm Development?

An Optimization Problem → **A Foundational Neural Network Model** → **Universal Test-Time Search Methods** → Solution

http://comet.kaist.ac.kr

chkwon@kaist.ac.kr

https://omelet.ai

KAIST

CREDIT: KAIST Characters by 김태수, 석현정, KAIST Brand Shop.

Omelet