THE UNIVERSITY OF MELBOURNE

Dynamic Graph Combinatorial Optimization with Multi-Attention Deep Reinforcement Learning

ACM SIGSPATIAL 2022

Authors: Udesh Gunarathna, Dr. Renata Borovica-Gajic, Prof. Shanika Karunasekera, And Prof. Egemen Tanin

Combinatorial Optimization is Everywhere!

Applications of combinatorial optimization range from theoretical mathematics to a wide range of industries

Combinatorial optimization applications:

- Algorithm theory / Operations research
- Social networks
- Logistic planning / Supply chains
- Transportation
- Medical applications

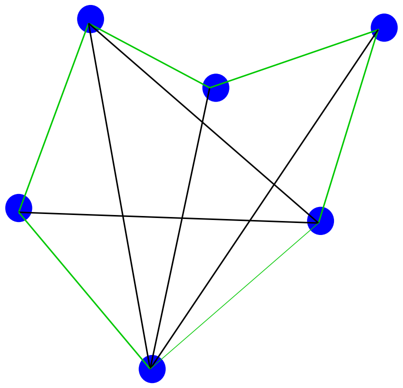
Many of these applications require solving a version of **dynamic** combinatorial optimization!

Why Consider "Dynamicity": TSP Example

Travelling Salesman Problem: We need to visit all the nodes in the shown graph and let's assume that the **travelling cost between nodes is proportional to both the thickness and the length of an edge.** The visited nodes are in blue color and the current path is shown in green color.

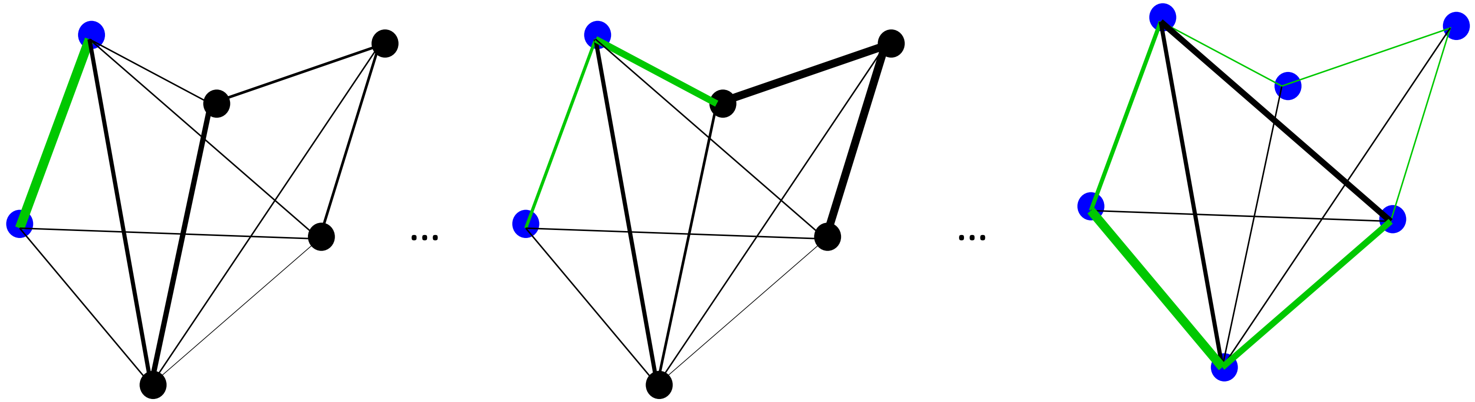
Static version:

The edge thickness does not change over time.



Dynamic version:

The edge thickness does change over time.



$t = t_0$ $t = t_i$ $t = t_n$

The solution in the static version may not be the optimal solution in the dynamic version

Can We Solve Dynamic Combinatorial Problems Efficiently?

Challenges

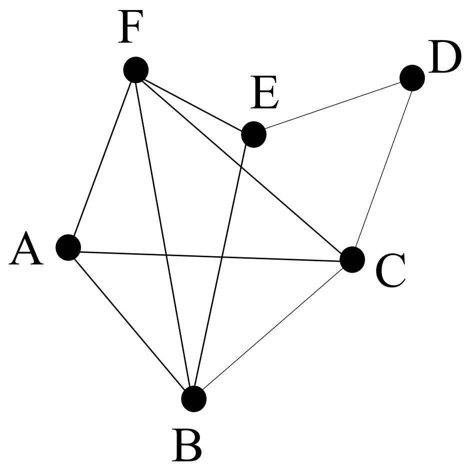
- Usually, static combinatorial problems are NP-hard and the dynamic nature makes the problem even more challenging to solve
- Exact solutions are computationally expensive and cannot apply to large problem instances
- Heuristic methods are faster compared to exact solutions but require domain knowledge (i.e. problem specific knowledge) or manual feature engineering to design such heuristics
- Heuristic needs to be separately designed for each combinatorial optimization problem

What if we have a heuristic that does not need human intervention and general enough to be applied to more than one single combinatorial problem

Neural Combinatorial Optimization (NCO)

- Using machine learning/neural networks to learn a heuristic to solve NP-hard combinatorial problems is known as **neural combinatorial optimization**
  – NCO can provide heuristics with near optimal performance without manual feature engineering

- TSP example:



Static problem

→ Neural Network + Reinforcement Learning (NCO) → {A, F, E, D, C, B, A}

NCO learns to solve the TSP problem for the graphs generated from the same distribution

Sequentially provides the node order to visit in order to minimize the cost

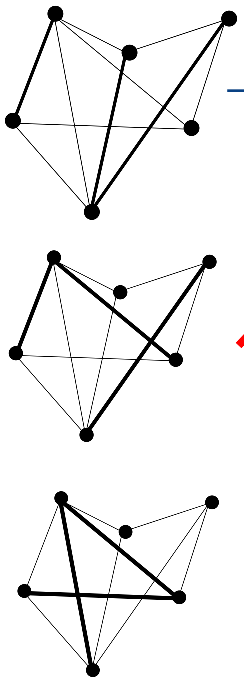
Challenges in Dynamic Neural Combinatorial Optimization

- Challenges in dynamic neural combinatorial optimization
  - Most of the previous work focus on the static version of combinatorial problems [1, 2, 3, 4]
    - Cannot handle temporal information (i.e. graph input at different time steps)

$t = t_0$

$t = t_i$

$t = t_n$



Dynamic problem

Neural Network + Reinforcement Learning (NCO)

How to compute the order?

How to provide graph information at different time steps

- Not only the order but when to select a node is also important (as shown in the first TSP example)
- To achieve this objective, the learning model needs to incorporate temporal information along with the spatial information

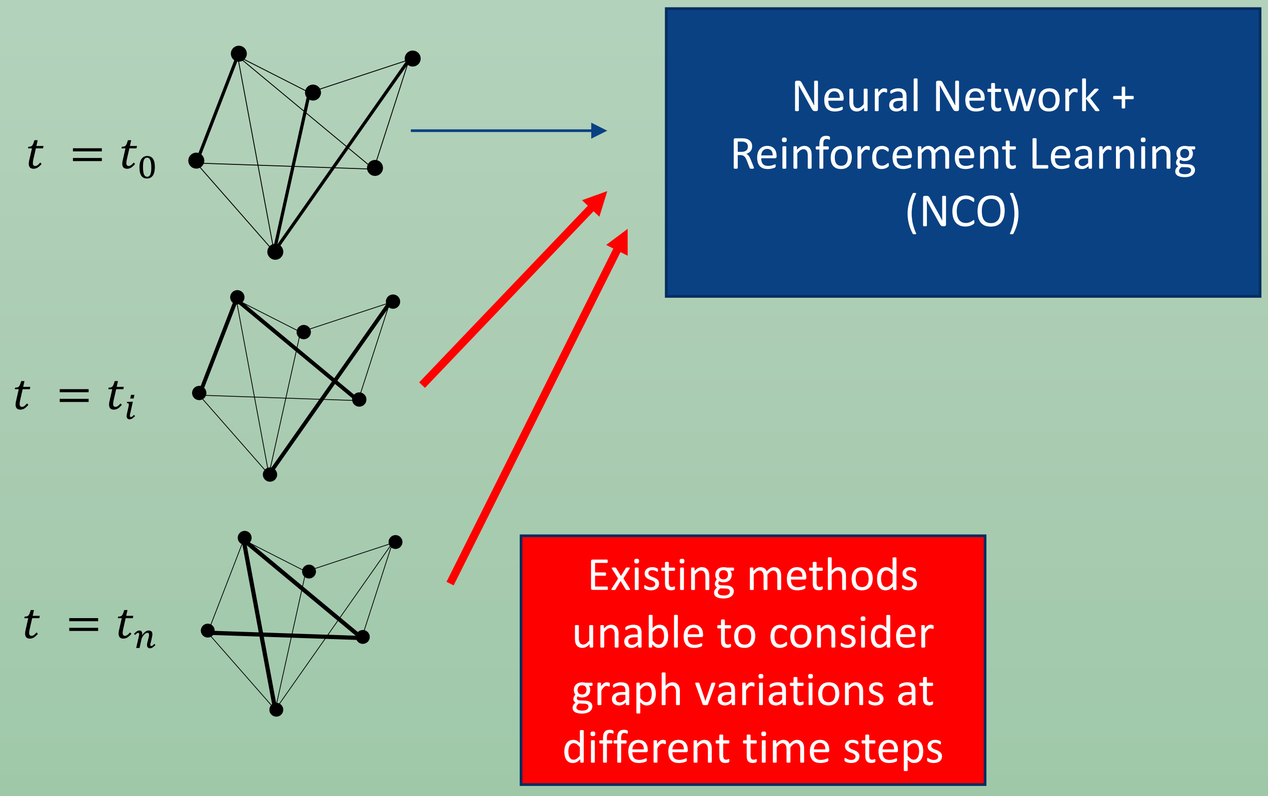
Graph Temporal Attention with Reinforcement Learning (GTA-RL)

- We propose a novel deep reinforcement learning architecture named **GTA-RL** to address the aforementioned limitations

- **GTA-RL** uses an encoder-decoder architecture
  - A temporal encoder : Consists of two parallel attention layers.
  – Spatial Attention: Encodes graph topology and node locations in a single time step
  – Temporal Attention: Encodes the node feature changes over time in the graph
  - A temporal decoder : Uses the encoder output to determine the solution utilizing the temporal information

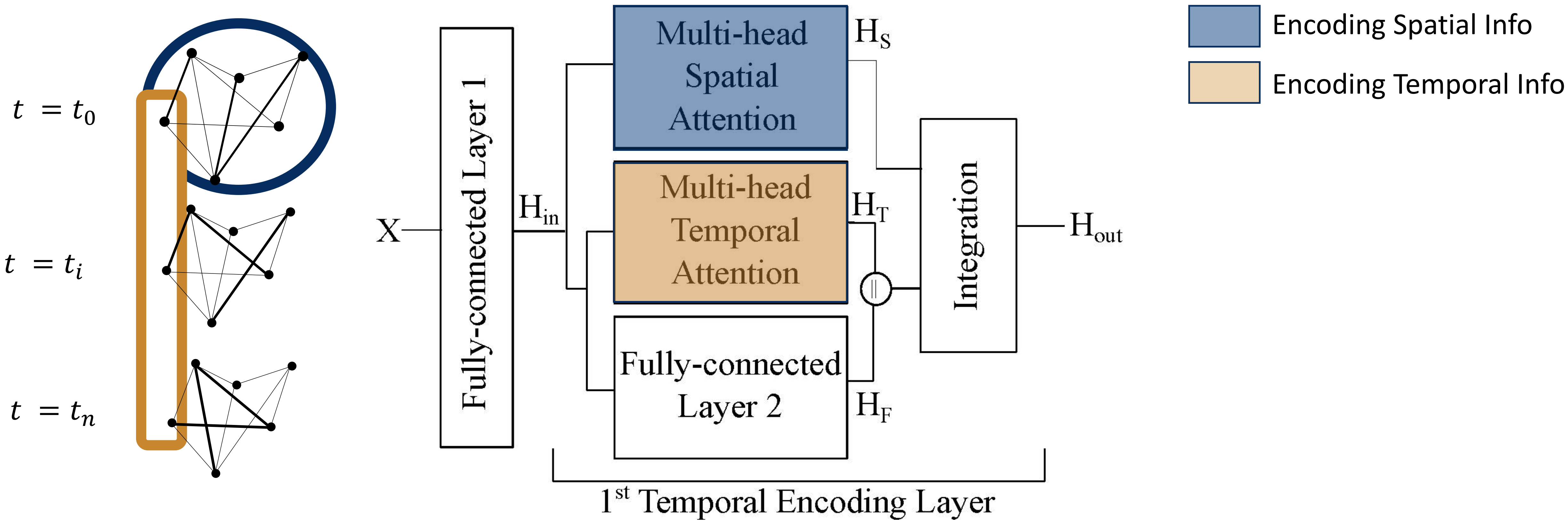
- A modified RL algorithm is used to train the **GTA** network

Graph Temporal Attention with Reinforcement Learning (GTA-RL) : Encoder



- Existing Methods -

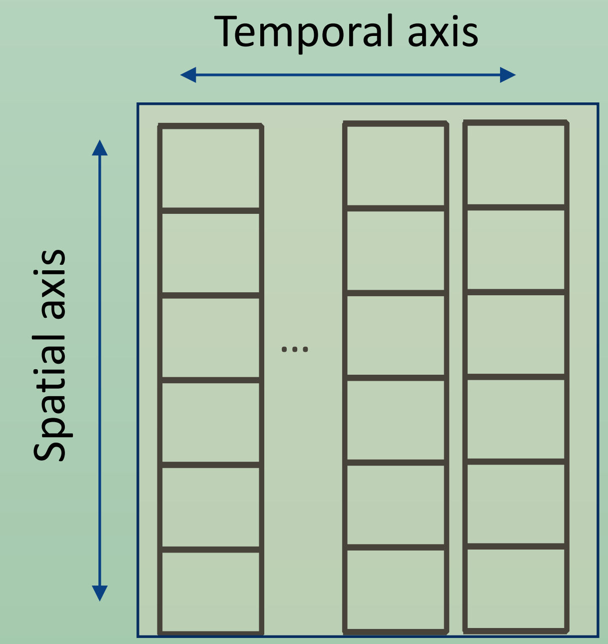
Graph Temporal Attention with Reinforcement Learning (GTA-RL) : Encoder



GTA-RL: Encoder

Graph Temporal Attention with Reinforcement Learning (GTA-RL) : Decoder

- Existing Methods -



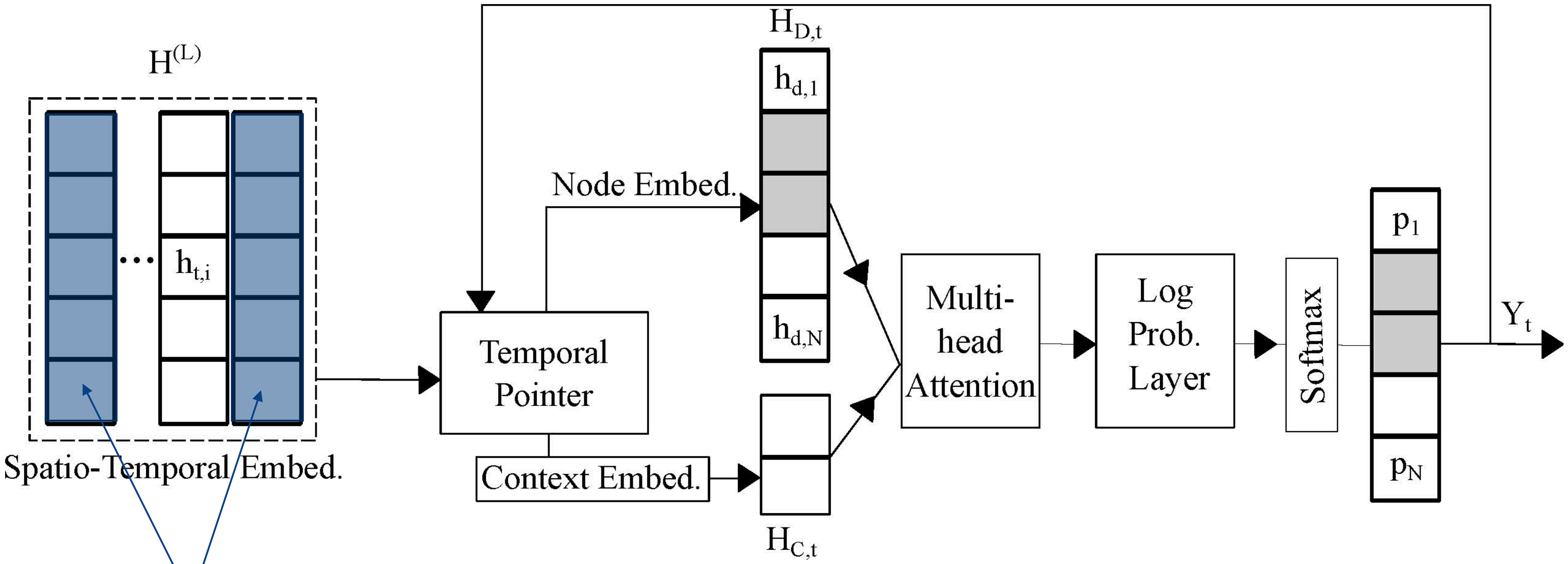
Encoder output

Neural Network Decoder

The entire information about the dynamic graph problem in a vector representation

Existing methods take all encoder information at once to the decoder and cannot focus on information that is required at a given time step

Graph Temporal Attention with Reinforcement Learning (GTA-RL) : Decoder



Temporal pointer can focus on specific parts of the embedding output according to the context

GTA-RL: Decoder

Experimental Setup

We test GTA-RL using two benchmark combinatorial problems:

1. Travelling Salesman Problem
2. Vehicle Routing Problem

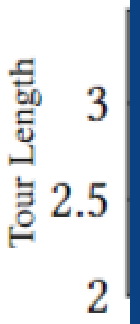
Dynamic Travelling Salesman Problem (TSP):

- In dynamic TSP, the initial node locations are assigned uniformly at random between (0,0)-(1,1) in 2d-space. The node locations are updated uniformly with a maximum change of 0.1 in coordinates. The cost of traveling between nodes changes over time in this setup
- The objective is to find the order of visiting all the nodes such that the total travelled distance is minimal
- This setup can be generalized to transport or telecommunication networks

Similar setup is created for **Vehicle Routing Problem (VRP)** as well

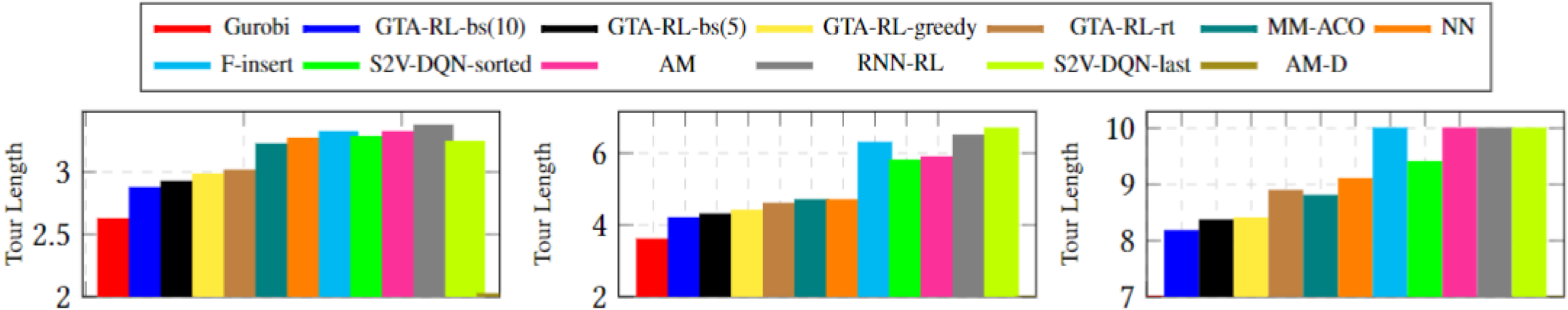
Experimental Results: Dynamic TSP

- S2V-DQN: S2V-DQN uses structure2vec for graph encoding and fitted Q-learning and does not support VRP. There are two variants named S2v-DQN-last and S2V-DQN-sorted
- RNN-RL: RNN-RL uses policy-gradient with two recurrent encoders named static and dynamic
- AM: An attention model which achieves SOTA in static TSP and VRP.
- AM-D: AM-D uses an additional dynamic encoder
- Gurobi: Optimal solver with integer programming
- NN: Nearest Neighbor heuristic where next nearest neighbor is selected
- Farthest Insertion (F-Insert): The next node is selected to minimize the current cost
- Min-Max Ant Colony Optimization (MM-ACO): Ant colony method to solve TSP
- GTA-RL-bs, GTA-RL-Greedy, GTA-RL-rt are variants of GTA-RL



Tour length of dynamic TSP in graphs with 10, 20 and 50 nodes respectively

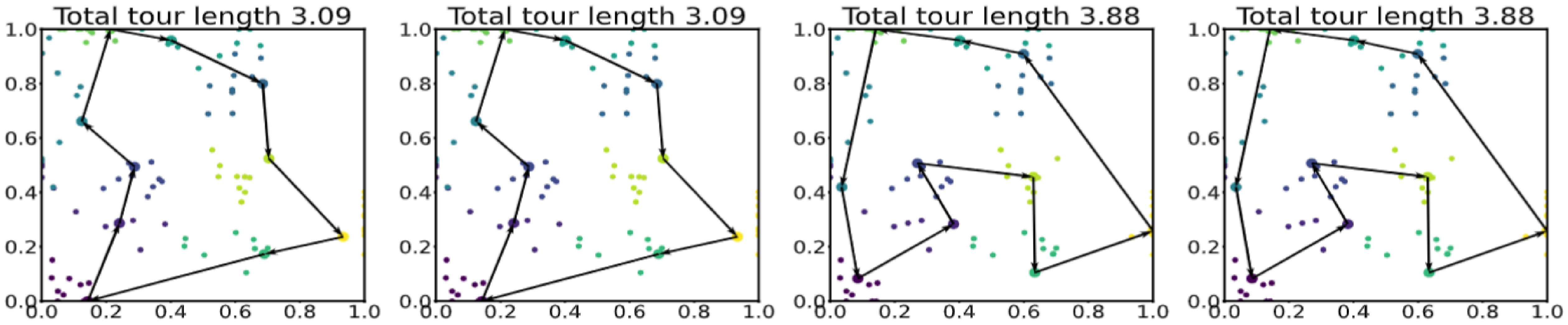
Experimental Results: Dynamic TSP



(a) Tour Length for TSP10 (b) Tour Length for TSP20 (c) Tour Length for TSP50

Tour length of dynamic TSP in graphs with 10, 20 and 50 nodes respectively

Visualizations: Why GTA-RL Performs Well?



(a) Gurobi (Optimal Solution)

(b) GTA-RL-bs

(c) Nearest Neighbor

(d) MM-ACO

- Figure shows the order of selected nodes by Gurobi optimal solver, GTA-RL, NN and MM-ACO. For easy visualization we use dynamic TSP10 where only 10 cities are present.
- The dots with the same color indicate the same city locations at different time steps. The algorithm selects to visit a node in one time step which has been highlighted by a dot larger than other nodes in the same color.

Future Directions

- Improve the scalability in dynamic neural combinatorial optimization domain
  - Scalability is an open problem in this domain. Our results show that GTA-RL trained in a smaller graph can be generalized to a larger graph (refer to the paper). This is an interesting direction to investigate.

- Reinforcement learning (or GTA-RL) for multi-commodity flows
  - GTA-RL can solve combinatorial problems that can be formulated as a sequentially addition of nodes. Multi-commodity flow problems require multiple paths and multiple vehicles to be considered. It is interesting to see how to extend GTA-RL for such problems.

References

[1]. Hanjun Dai, Elias Khalil, Yuyu Zhang, Bistra Dilkina, and Le Song. 2017. Learning Combinatorial Optimization Algorithms over Graphs. In Advances in Neural Information Processing Systems, Vol. 30. 6351–6361

[2]. ddo Drori, Anant Kharkar, William R. Sickinger, et al . 2020. Learning to Solve Combinatorial Optimization Problems on Real-World Graphs in Linear Time. In IEEE International Conference on Machine Learning and Applications (ICMLA). 19–24

[3].Wouter Kool, Herke van Hoof, and Max Welling. 2019. Attention, Learn to Solve Routing Problems!. In International Conference on Learning Representations

[4]. Mohammadreza Nazari, Afshin Oroojlooy, et al . 2018. Reinforcement Learning for Solving the Vehicle Routing Problem. In Proceedings of the 32nd International Conference on Neural Information Processing Systems (NIPS'18). 9861–9871.