Neural Combinatorial Optimization: Recent Advances in Deep Learning for Routing Problems

Chaitanya K. Joshi, University of Cambridge, UK

Rishabh Anand, National University of Singapore

[**chaitjo.com**/post/deep-learning-for-routing-problems](https://chaitjo.com/post/deep-learning-for-routing-problems)

LoGaG Reading Group, 8 March 2022

Agenda

1. **Background and Motivation** (Quick)

2. Unified Neural Combinatorial Optimization Pipeline (Quick)

3. Case Studies of Recent Advances & Future Work (Fun Part!)

Combinatorial Optimization

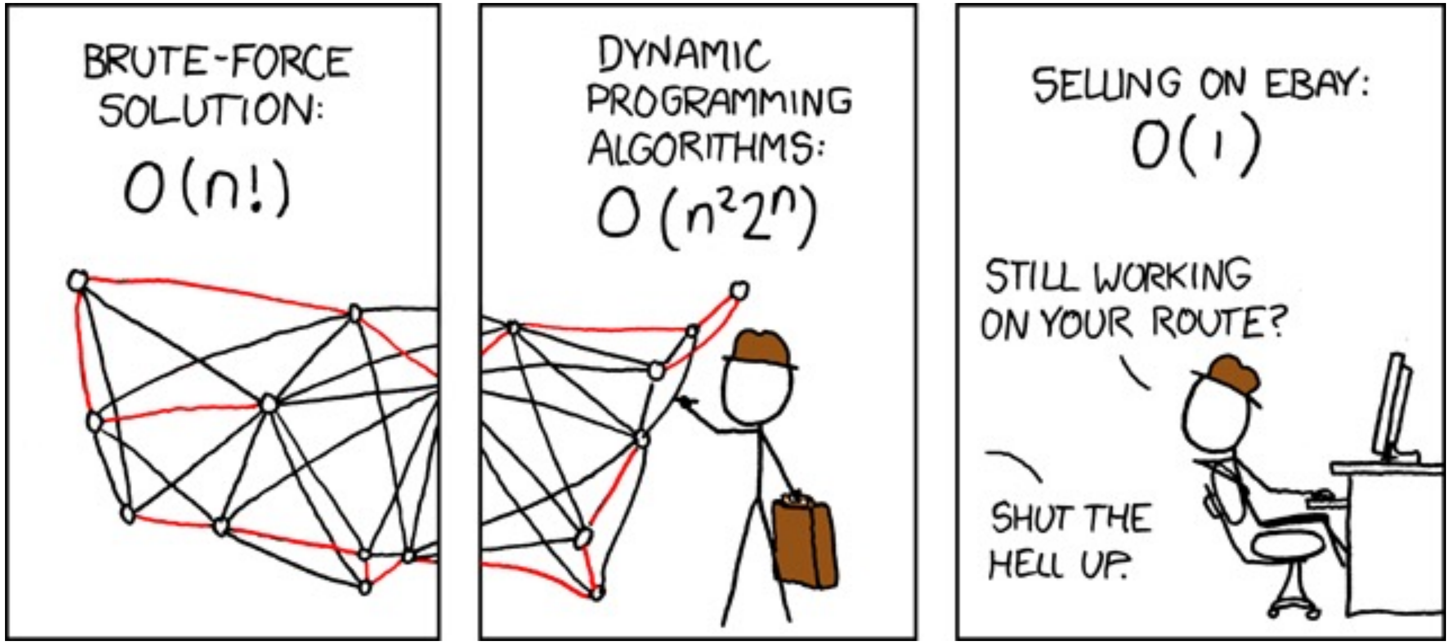
- **Combinatorial Optimization**
  - In the intersection of mathematics and computer science.
  - Solve constrained **optimization problems** which are **NP-Hard**.
- **NP-Hard problems**
  - Impossible to **solve optimally** at **large scales**: exhaustively searching for their solutions is beyond the limits of modern computers.
- **Why should we care?**
  - Robust and reliable approximation algorithms have immense practical applications and are the **backbone of modern industries**.
  - Usually defined on graphs → Graph Neural Networks!
  - Pre-empt many recent trends in GRL!

Travelling Salesperson Problem

"Given a list of cities and the distances between each pair of cities, what is the **shortest possible route** that a salesperson can take to **visit each city** and **returns to the origin city**?"

Travelling Salesperson Problem

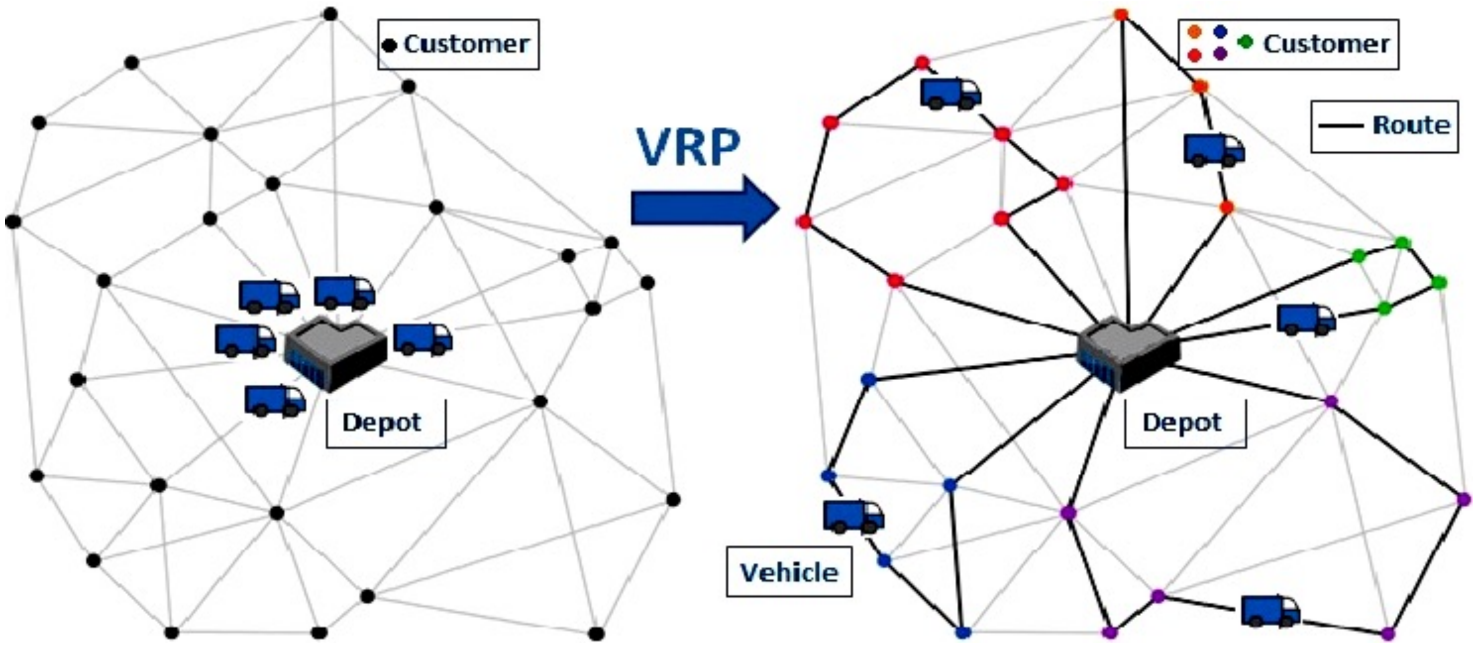
So famous, or notorious, that there is an xkcd comic on it:



Applications ranging from **logistics** and **scheduling** to **genomics**.

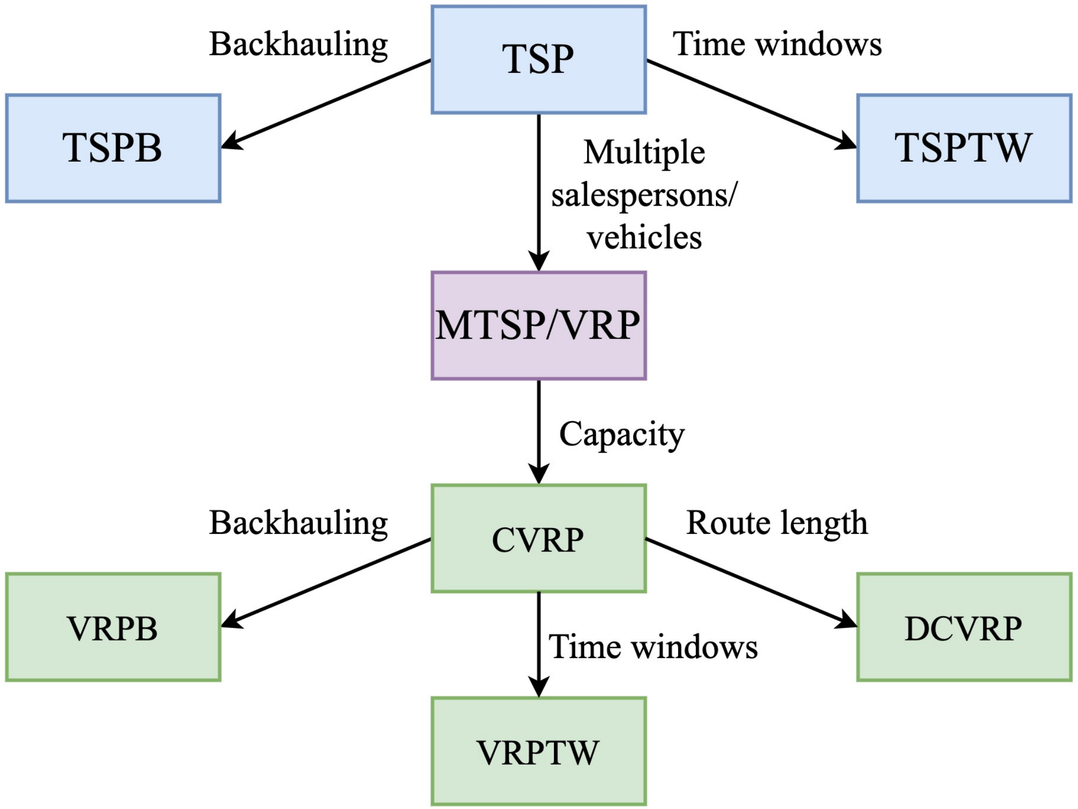
Routing Problems

- **Routing Problems**, a.k.a. Vehicle Routing Problems
  - Class of COPs that require **traversing** a sequence of nodes (e.g. cities) or edges (e.g. roads between cities) in a specific order, i.e. **routing**.
  - Routes must fulfil a set of **constraints** or optimise a set of variables.



Routing Problems

Real-world VRPs involve **challenging constraints** beyond the somewhat *vanilla* TSP. Some relatively well-studied ones include:



Deep Learning for Routing Problems

- Developing solvers: **expert intuition** and **years of trial-and-error**.
- **Concorde**[1]
  - State-of-the-art TSP solver.
  - Leverages over 50 years of research on **linear programming**, **cutting plane algorithms** and **branch-and-bound**.
  - Can find **optimal solutions** up to tens of thousands of nodes, but with extremely **long execution time**.
- Solvers for **complex VRPs** are even more challenging, especially with **real-world constraints** such as capacities or time windows in the mix.

[1] Applegate, et al., The traveling salesman problem: a computational study, Princeton university press, 2006.

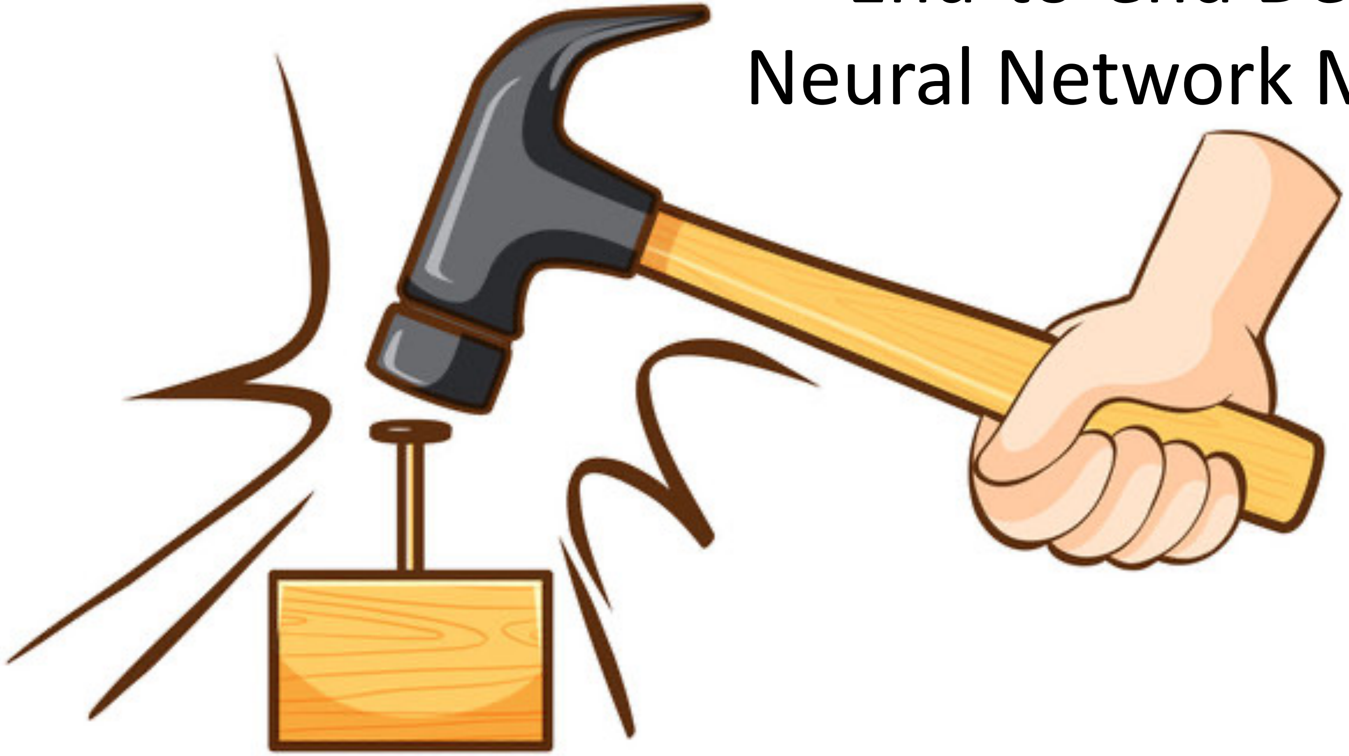
Big Picture Idea:

Can we use **Deep Learning** to **automate** and **augment** expert intuition required for solving **Combinatorial Optimization Problems**?[1]

[1] Bengio et al., Machine learning for combinatorial optimization: a methodological tour d'horizon, EJOR 2020

Neural Combinatorial Optimization

End-to-end Deep
Neural Network Model



Combinatorial
Optimization Problems

(Usually at the expense of theoretical guarantees and bounds on solutions…)

Neural Combinatorial Optimization

- Neural networks produce **approximate solutions** to COPs by directly learning from **problem instances** themselves (end-to-end)[1].
- **GNNs** at the core of deep learning-driven solvers[2].
- Why?

1. No Handcrafted Heuristics	2. Fast Inference on GPUs	3. Tackling Under-studied COPs
• Learnt heuristics via imitating optimal solvers or via RL.	• Real-time decision making. • Large-scale instances.	• Scientific discovery[3]. • Computer architecture[4].

[1] Vinyals et al., Pointer Networks, NeurIPS 2015
[2] Cappart et al., Combinatorial optimization and reasoning with GNNs, IJCAI 2021
[3] Jumper et al., Highly accurate protein structure prediction with AlphaFold, Nature 2021
[4] Mirhoseini et al., A graph placement methodology for fast chip design, Nature 2021

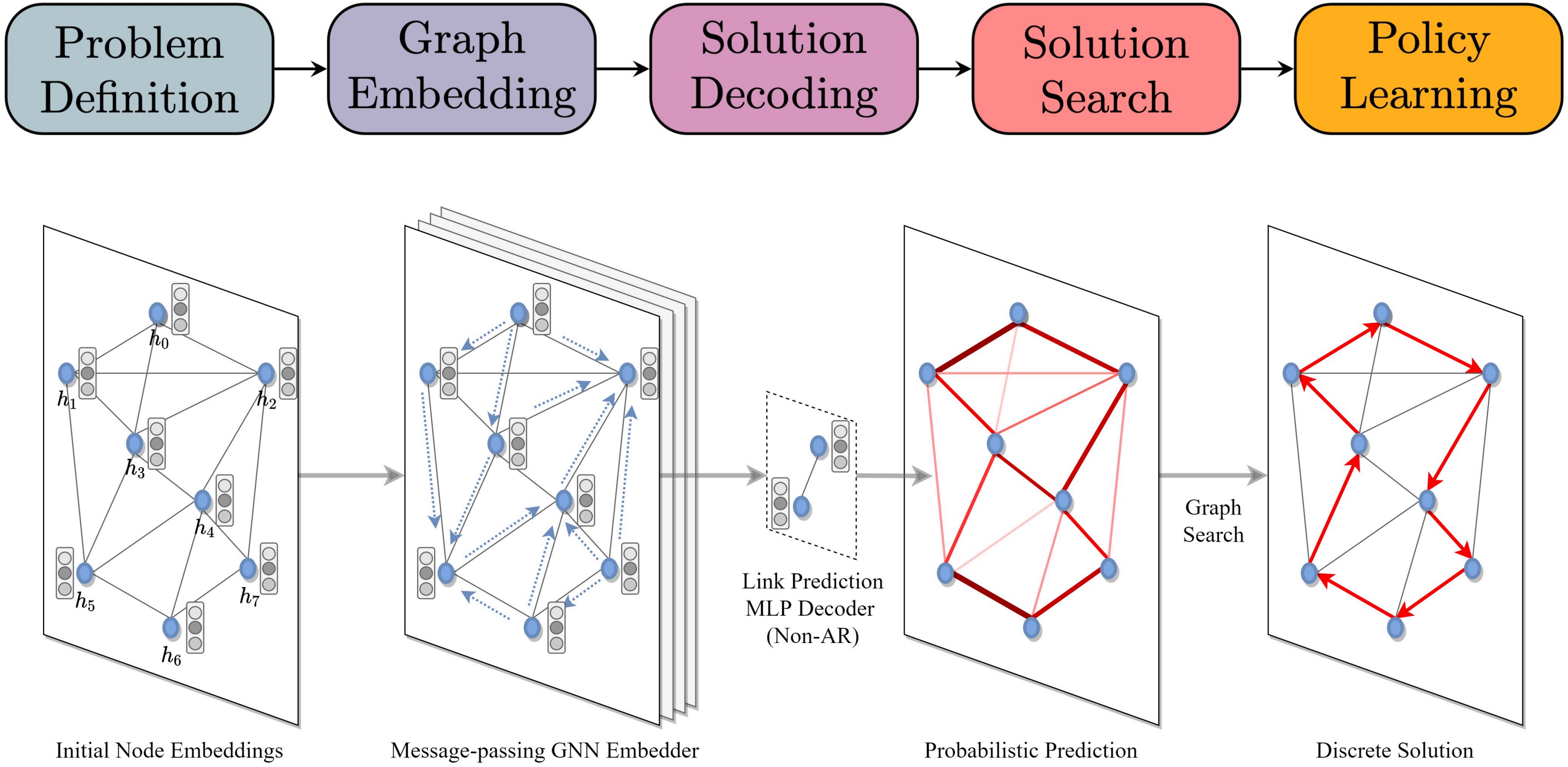
Agenda

1. Background and Motivation

2. **Unified Neural Combinatorial Optimization Pipeline**

3. Case Studies of Recent Advances & Future Work

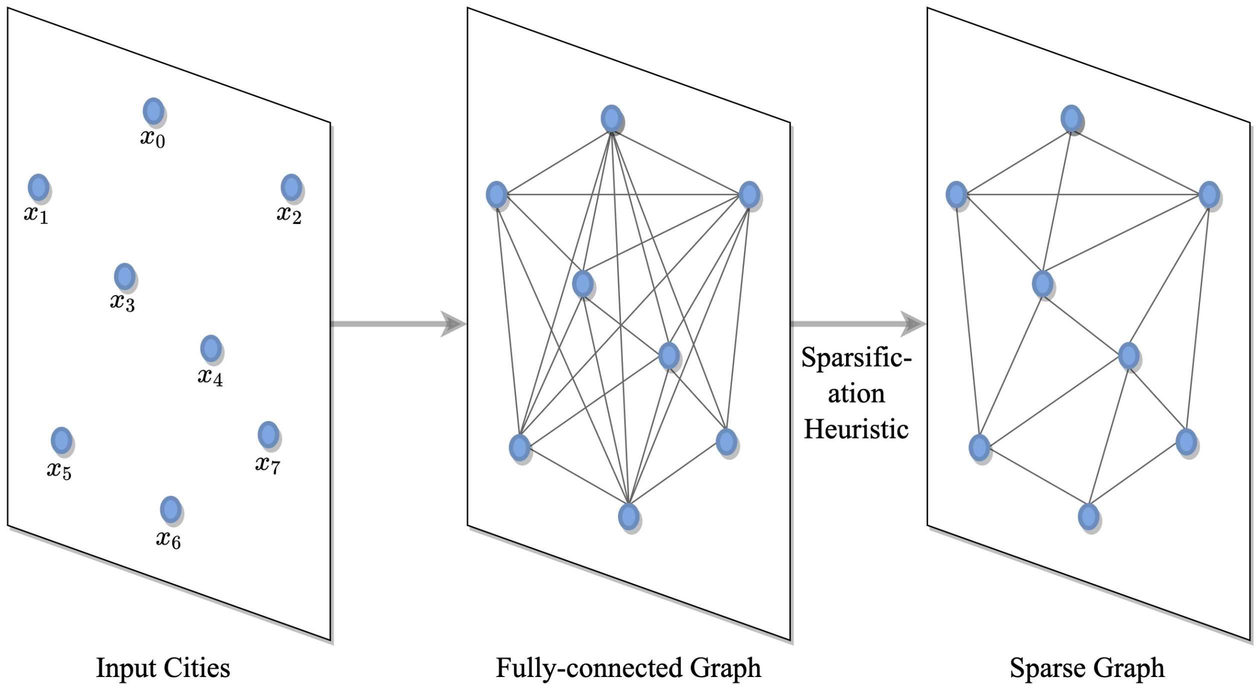
A Unified Pipeline for Neural Comb. Opt.[1]



[1] Joshi et al., Learning TSP Requires Rethinking Generalization, CP 2021

(1) Defining the problem via graphs

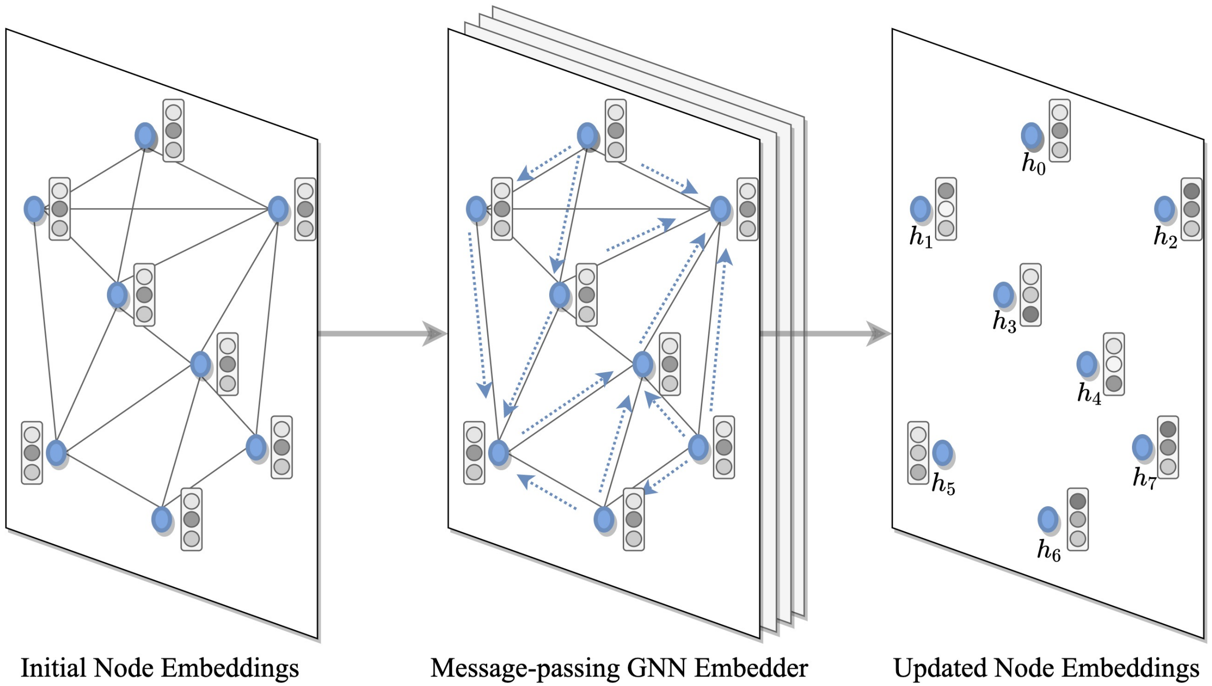
TSP is formulated via a **fully-connected graph** of cities/nodes, which can be **sparsified** further for learning on larger graphs[1] or faster[2].



[1] Khalil, Dai, et al., Learning combinatorial optimization algorithms over graphs, NeurIPS 2017
[2] Joshi et al., An Efficient Graph Convolutional Network for the TSP, arXiv 2019

(2) Obtaining embeddings for graph nodes and edges

Embeddings are obtained using a **GNN[1][2]/Transformer[3][4]** encoder, which builds **local structural features** via aggregating from neighbourhoods.



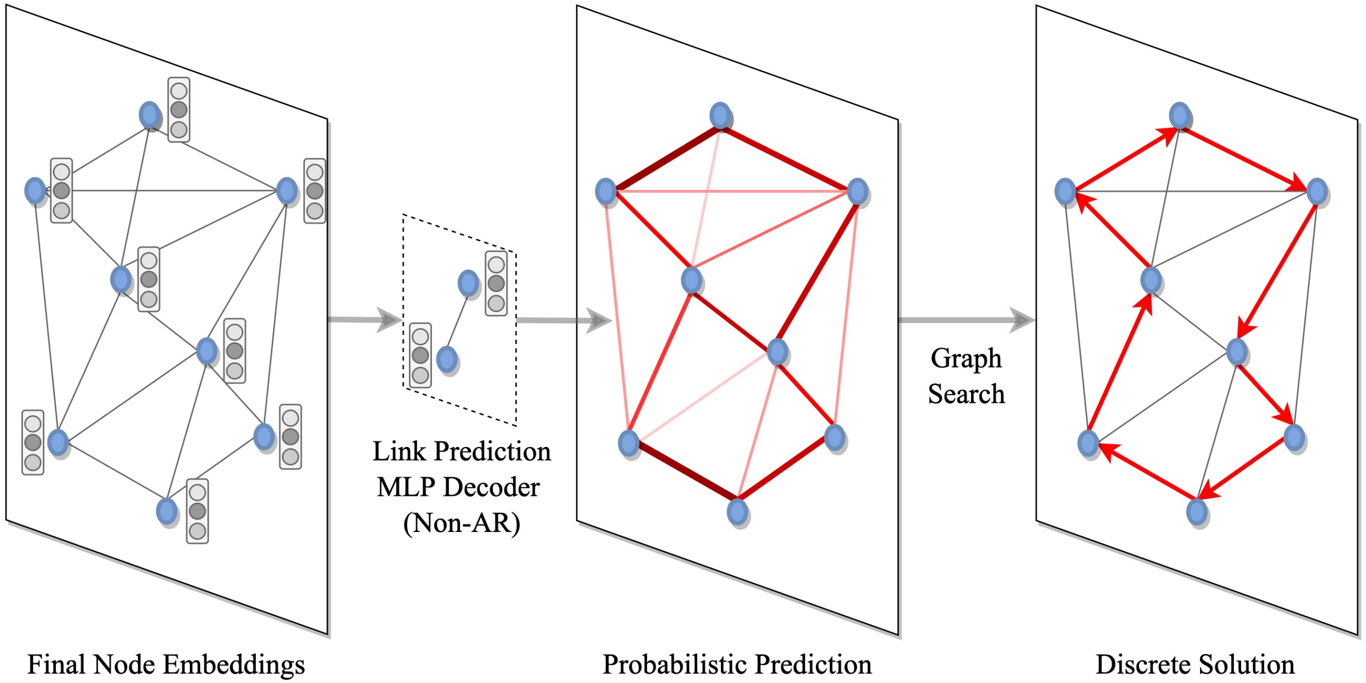
$$h_i^{\ell+1} = \phi\left(h_i^{\ell}, \oplus_{j\in\mathcal{N}_i}\left(\psi\left(h_i^{\ell}, h_j^{\ell}, e_{ij}\right)\right)\right)$$

Bonus: nice connections!

[1] Khalil, Dai, et al., Learning combinatorial optimization algorithms over graphs, NeurIPS 2017
[2] Joshi et al., An Efficient Graph Convolutional Network for the TSP, arXiv 2019
[3] Deudon et al., Learning heuristics for the tsp by policy gradient, CPAIOR 2018
[4] Kool et al., Attention, learn to solve routing problems!, ICLR 2019

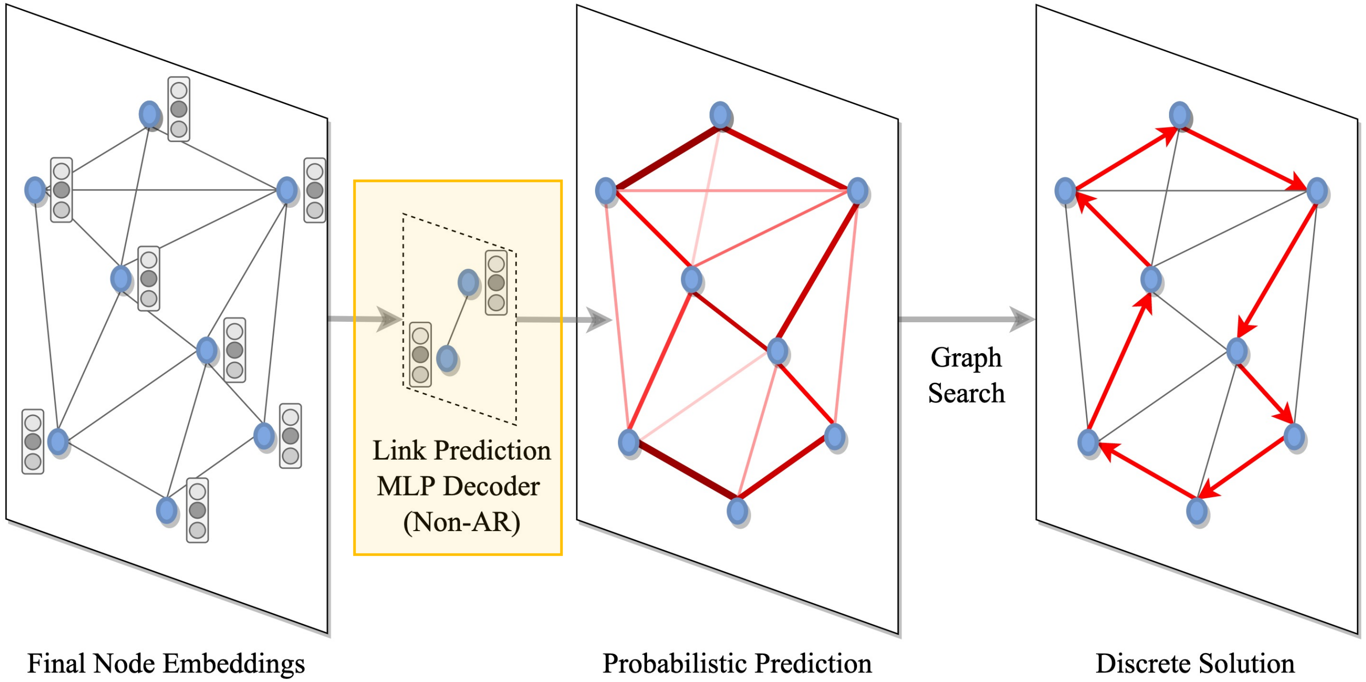
(3 + 4) Converting embeddings into discrete solutions

- Probabilities are assigned to each node or edge for **belonging to the solution set.**
- Converted into **discrete decisions** through classical graph search techniques, e.g. greedy search, beam search.



(3 + 4) Non-autoregressive Decoding

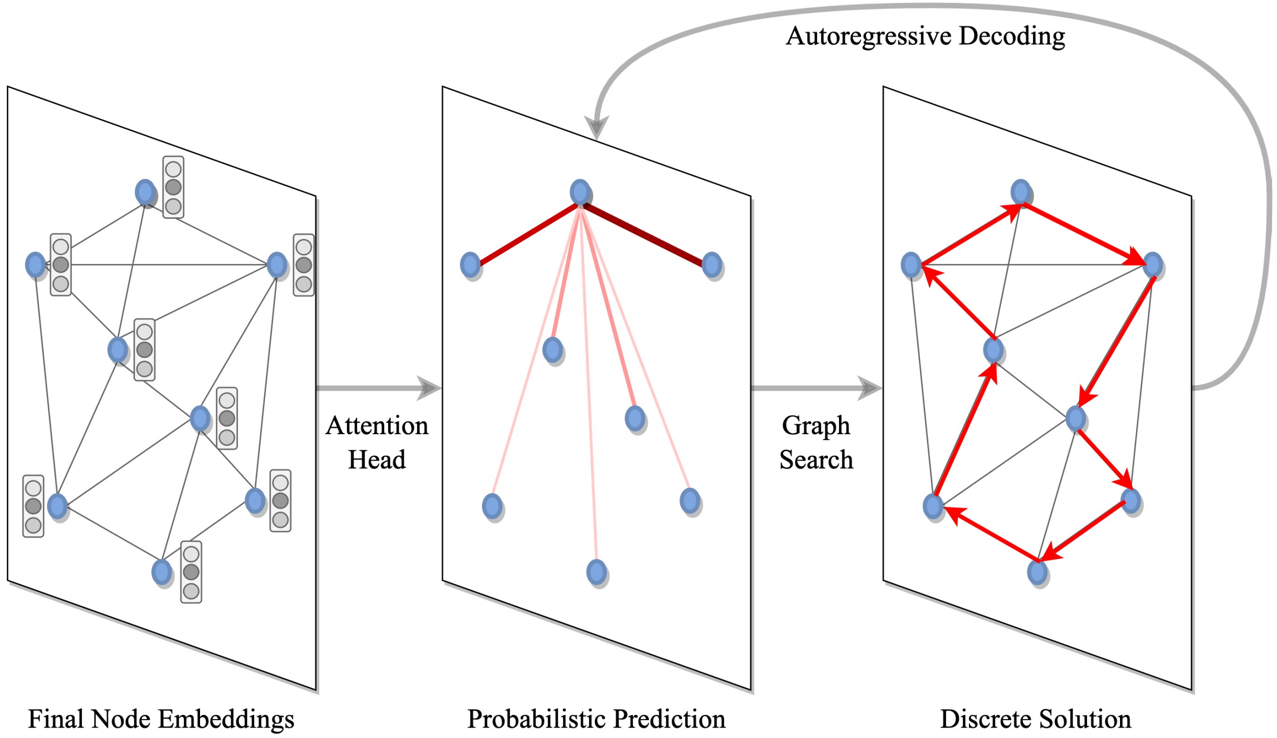
Non-autoregressive Decoding: MLP makes a prediction per edge to obtain a **'heatmap'** of edge probabilities[1][2].



[1] Nowak, et al., A note on learning algorithms for quadratic assignment with graph neural networks, arXiv 2017
[2] Joshi et al., An Efficient Graph Convolutional Network for the TSP, arXiv 2019

(3 + 4) Autoregressive Decoding

Autoregressive Decoding: Probabilities are assigned conditionally through step-by-step graph traversal via a **pointing mechanism** (attention)[1][2][3][4].



[1] Vinyals et al., Pointer Networks, NeurIPS 2015
[2] Bello et al., Neural Combinatorial Optimization, ICLR 2017
[3] Deudon et al., Learning heuristics for the tsp by policy gradient, CPAIOR 2018
[4] Kool et al., Attention, learn to solve routing problems!, ICLR 2019

(5) Training the pipeline: Imitation Learning

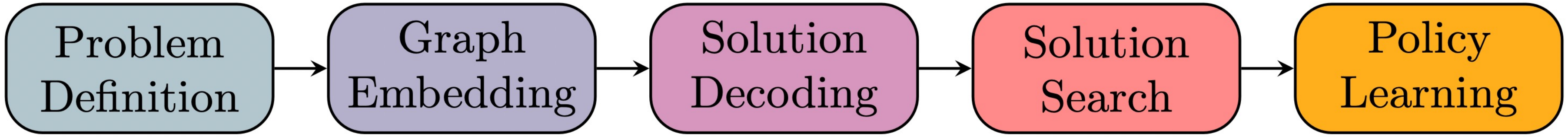
Problem Definition → Graph Embedding → Solution Decoding → Solution Search → Policy Learning

- Entire encoder-decoder model is trained in **end-to-end**.

- **Imitating an optimal solver**, i.e. supervised learning
  - **Concrode solver** is used to generate **labelled training datasets** of optimal tours for millions of random TSP instances.
  - **AR decoder models**: trained via **teacher-forcing** to output the optimal sequence of tour nodes (seq2seq)[1].
  - **NAR decoder models**: trained to **identify edges traversed** during the tour from non-traversed edges (binary classification over edges)[2].

[1] Vinyals et al., Pointer Networks, NeurIPS 2015
[2] Joshi et al., An Efficient Graph Convolutional Network for the TSP, arXiv 2019

(5) Training the pipeline: Reinforcement Learning

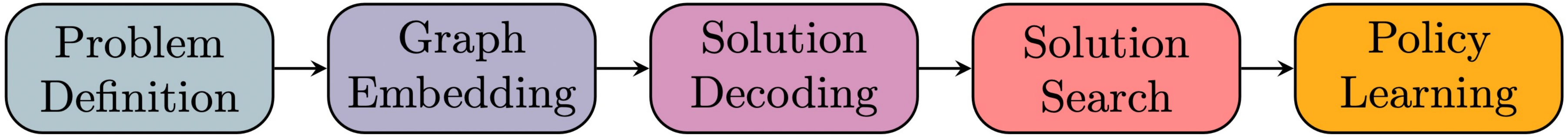


- **Reinforcement Learning**
  - Routing problems: **minimize a problem-specific cost functions** (e.g. the tour length for TSP) => elegantly cast in RL framework.
  - AR decoder models: sequential decision making => trained via standard **policy gradient algorithms**[1][2] or **Deep Q-Learning**[3].

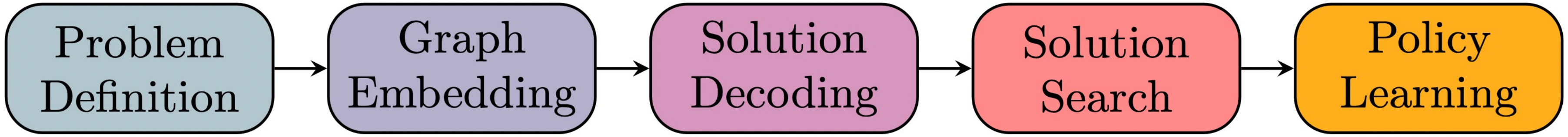
- Good alternative in the **absence** of **groundtruth solutions**, as is often the case for understudied problems, e.g. **chip design**[3].

[1] Bello et al., Neural Combinatorial Optimization, ICLR 2017
[2] Kool et al., Attention, learn to solve routing problems!, ICLR 2019
[3] Mirhoseini et al., A graph placement methodology for fast chip design, Nature 2021

Looking Back:
Characterizing Prominent Papers via the Pipeline

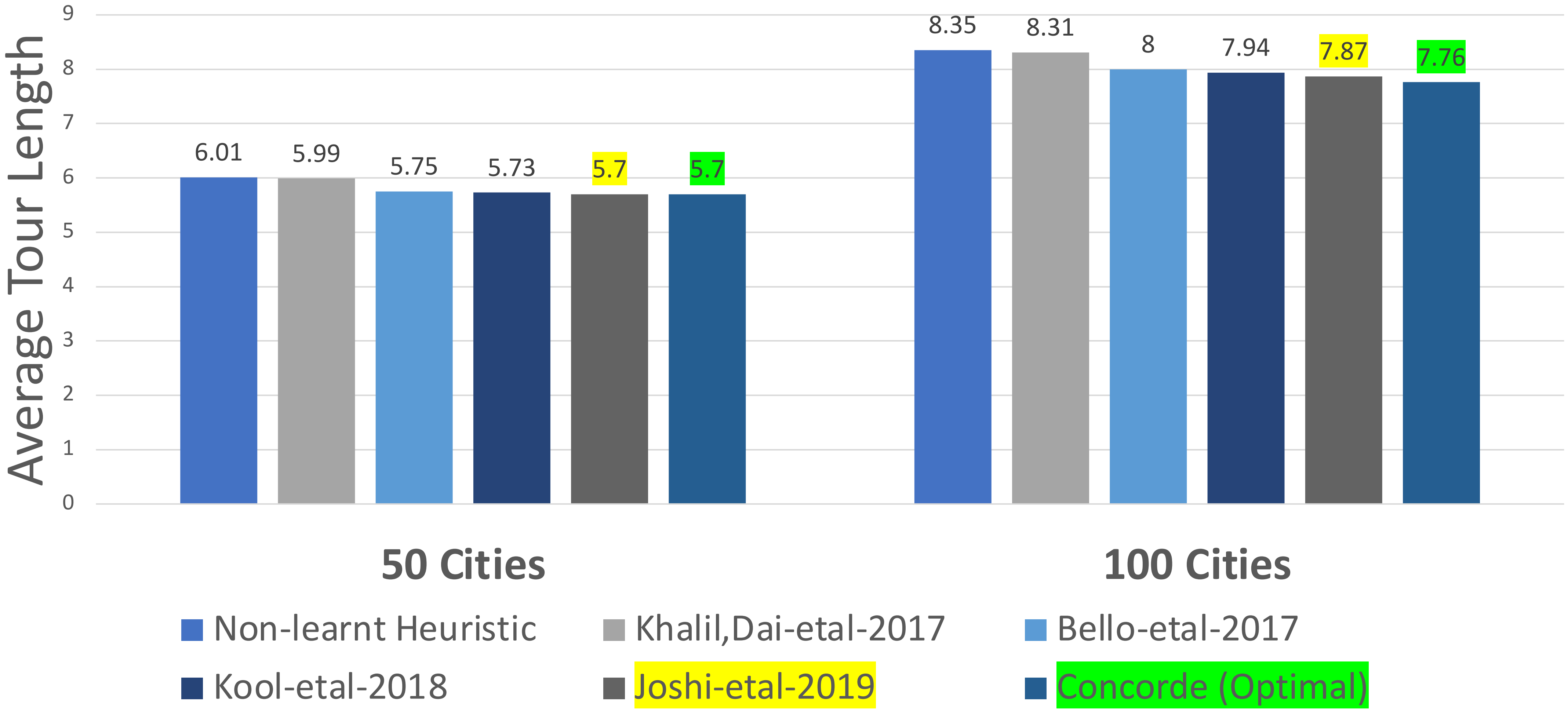


Paper	Definition	Graph Embedding	Solution Decoding	Solution Search	Policy Learning
Vinyals et al., 2015	Sequence	Seq2Seq	Attention (AR)	Beam Search	Immitation (SL)
Bello et al., 2017	Sequence	Seq2seq	Attention (AR)	Sampling	Actor-critic (RL)
Khalil et al., 2017	Sparse Graph	Structure2vec	MLP (AR)	Greedy Search	DQN (RL)



Paper	Definition	Graph Embedding	Solution Decoding	Solution Search	Policy Learning
Vinyals et al., 2015	Sequence	Seq2Seq	Attention (AR)	Beam Search	Immitation (SL)
Bello et al., 2017	Sequence	Seq2seq	Attention (AR)	Sampling	Actor-critic (RL)
Khalil et al., 2017	Sparse Graph	Structure2vec	MLP (AR)	Greedy Search	DQN (RL)
Deudon et al., 2018	Full Graph	Transformer Encoder	Attention (AR)	Sampling + Local Search	Actor-critic (RL)
Kool et al., 2019	Full Graph	Transformer Encoder	Attention (AR)	Sampling	Rollout (RL)
Joshi et al., 2019	Sparse Graph	Residual Gated GCN	MLP Heatmap (NAR)	Beam Search	Immitation (SL)
Ma et al., 2020	Full Graph	GCN	RNN + Attention (AR)	Sampling	Rollout (RL)

Below 1% Optimality Gap to Concorde for TSPs with 100s of Cities:



Agenda

1. Background and Motivation

2. Unified Neural Combinatorial Optimization Pipeline

3. **Case Studies of Recent Advances & Future Work**

Looking Ahead:
Recent Advances and Avenues for Future Work

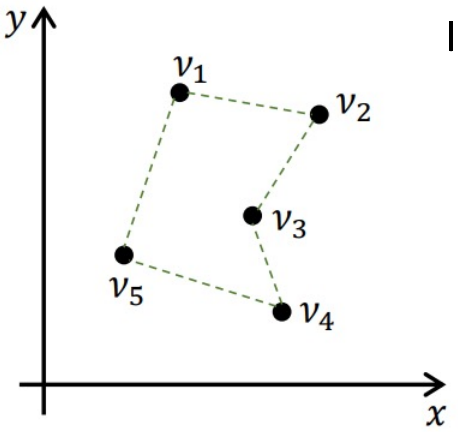
- With the unified 5-stage pipeline in place, let us highlight some **recent advances** and **trends** in deep learning for routing problems.

- We will also provide some **future research directions** with a focus on improving generalization to **large-scale** and **real-world** instances.

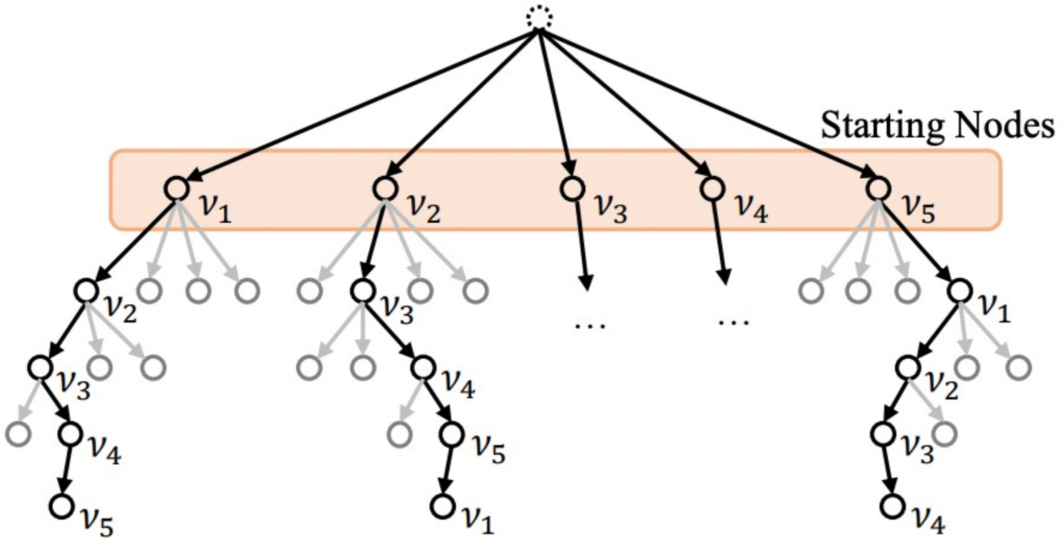
- Main challenges:
  - **Scaling:** Learning on or from very large-scale TSP instances.
  - **Generalization:** Transferring models from **small/synthetic** to **large-scale/real-world** TSP instances.

Leveraging Equivariance and Symmetries

- **Autoregressive decoding:** routing as **seq2seq** and solutions as **permutations of cities**.
  - This does not consider the **underlying symmetries** of **routing problems** – there may be multiple optimal permutations for the same tour.
- **POMO[1]:** Leverage invariance to the starting city.
  - Kool et al.'s model[2], but with a new reinforcement learning **objective/rollout** (pipeline step 5) which exploits the existence of multiple optimal tour permutations.



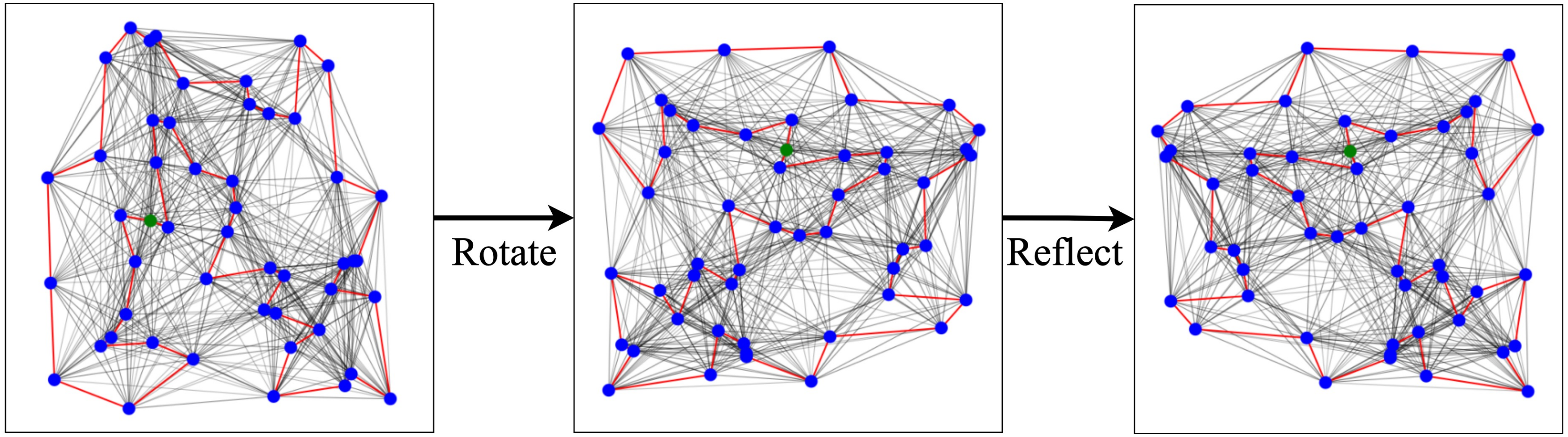
In seq2seq format



[1] Kwon et al., POMO: Policy Optimization with Multiple Optima, NeurIPS 2020
[2] Kool et al., Attention, learn to solve routing problems!, ICLR 2019

Leveraging Equivariance and Symmetries

- **eMAGIC**[1] upgrades Kool et al.'s model with equivariance to **rotations, reflections,** and **translations** (i.e. the **Euclidean symmetry** group) of the input city coordinates.
  - Ensure equivariance by: (1) **data augmentation** during problem definition (pipeline step 1); and (2) **relative coordinates** during graph encoding (pipeline step 2).
  - Super strong results on zero-shot generalization from **random instances** to the **real-world TSPLib benchmark** suite.



[1] Ouyang et al., Generalization in Deep RL for TSP Problems via Equivariance and Local Search, arXiv 2021

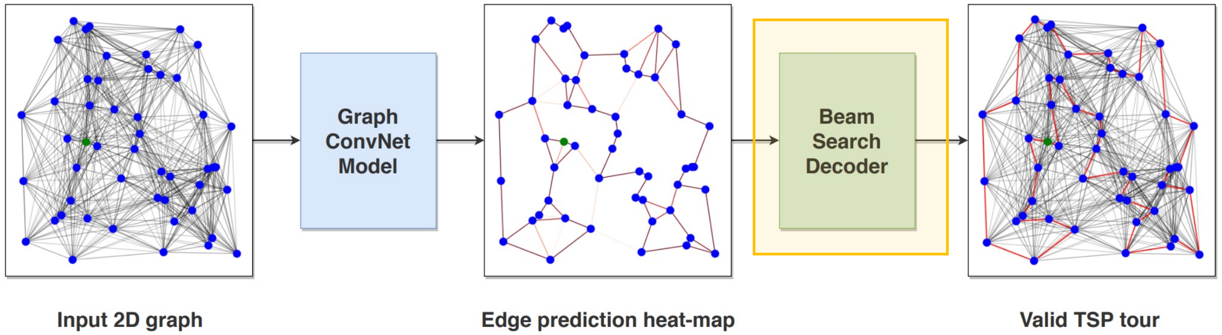
Leveraging Equivariance and Symmetries

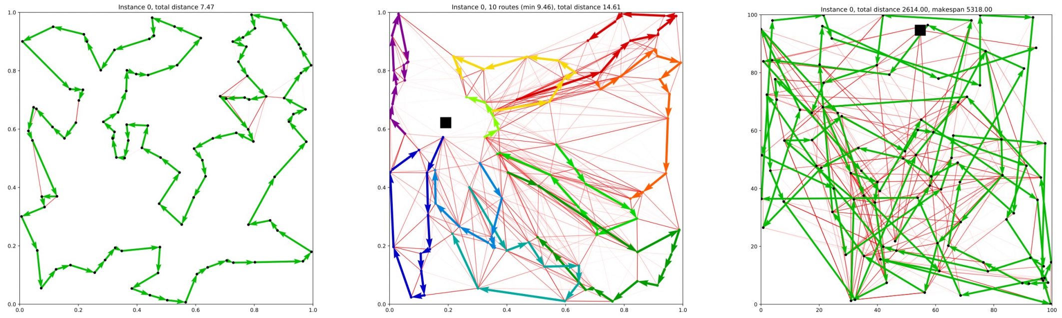
- Big picture: **Geometric Deep Learning**
  - Blueprint for architecture design: explicitly think about and incorporate the **symmetries** and **inductive biases** that govern the underlying data.
  - **Routing**: embedded in **Euclidean coordinates** and the **routes are cyclical**.

Paper	Definition	Graph Embedding	Solution Decoding	Solution Search	Policy Learning
Kool et al., 2019	Full Graph	Transformer Encoder	Attention (AR)	Sampling	Rollout (RL)
Kwon et al., 2020	Full Graph	Transformer Encoder	Attention (AR)	Sampling	POMO Rollout (RL)
Ouyang et al., 2021	Full Graph + Data Augmentation	Equivariant GNN	Attention (AR)	Sampling + Local Search	Policy Rollout (RL)

Improved Graph Search Algorithms

- One-shot, **non-autoregressive decoding**[1] + more powerful/flexible **graph search algorithms**, e.g. Dynamic Programming[2], MCTS[3].



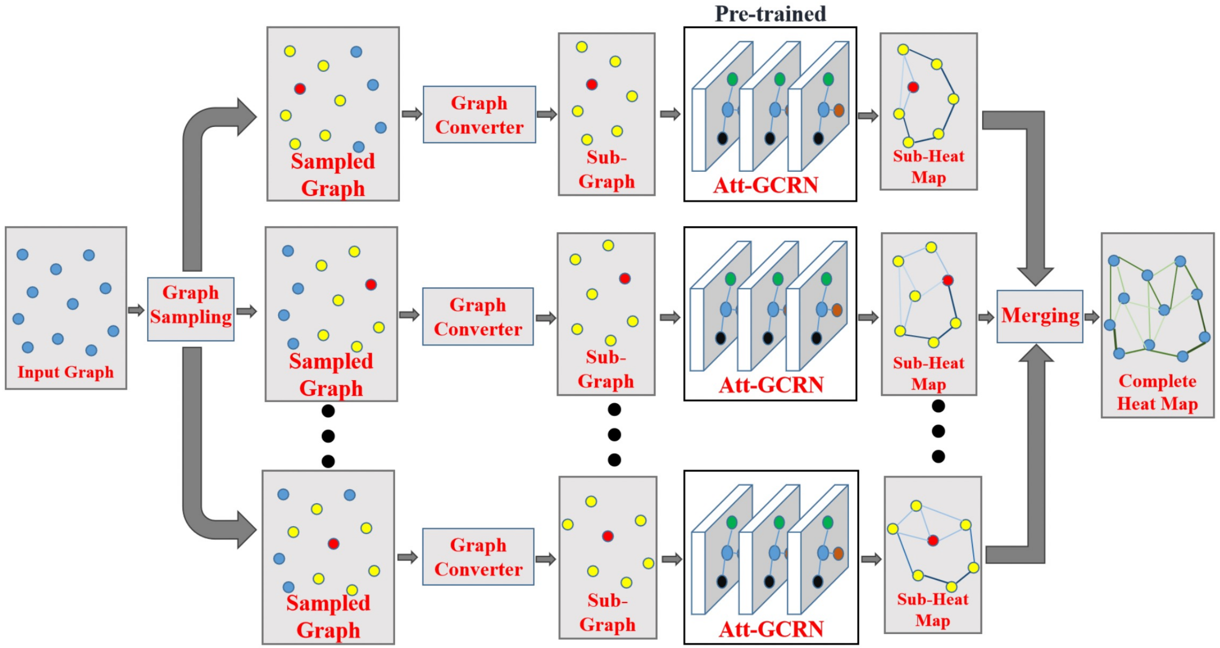


(a) Travelling Salesman Problem (b) Vehicle Routing Problem (c) TSP with Time Windows

[1] Joshi et al., An Efficient Graph Convolutional Network for the TSP, arXiv 2019
[2] Kool et al., Deep Policy Dynamic Programming for Vehicle Routing Problems, arXiv 2021
[3] Fu et al., Generalize a Small Pre-trained Model to Arbitrarily Large TSP Instances, AAAI 2021

Divide and Conquer for Extrapolation

- **Huge TSPs**: set of **small sub-graphs** of the same size as the graphs used for training the GNN.
- Sub-graph **heatmaps merged**[1] to obtain heatmap for the full graph, followed by **MCTS**.
- **Divide-and-conquer approach**[2] ensures that predictions by GNN generalize from smaller to larger instances. (Up to **10,000 node** TSPs at **3% optimality gap**!)



Bonus: nice connections!

[1] Fu et al., Generalize a Small Pre-trained Model to Arbitrarily Large TSP Instances, AAAI 2021
[2] Nowak et al., Divide and Conquer Networks, ICLR 2018

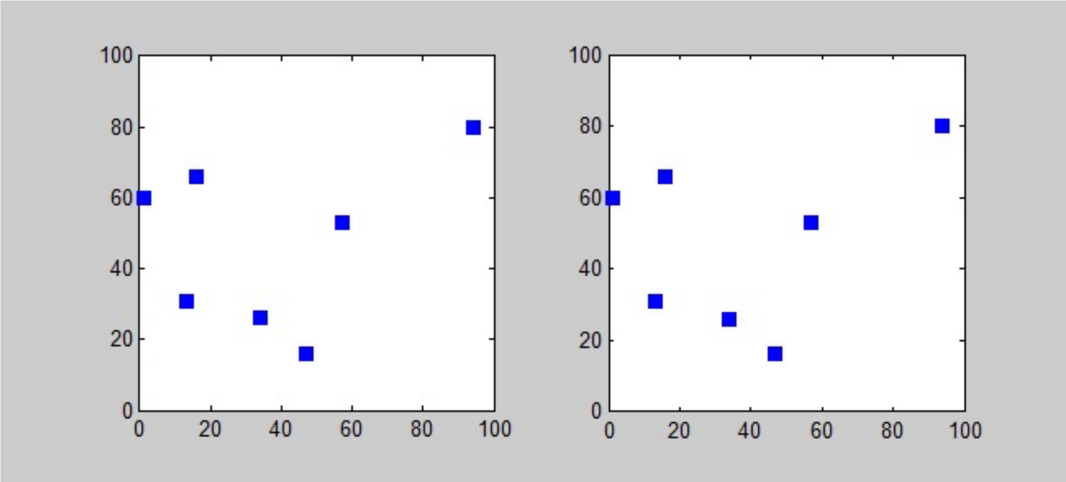
Neuro-symbolic AI

- Overall, **better search** + **divide-and-conquer** suggests **stronger coupling** between the design of the **neural** (GNNs + decoding) and **symbolic** (graph search) components is essential for out-of-distribution generalization.

Paper	Definition	Graph Embedding	Solution Decoding	Solution Search	Policy Learning
Joshi et al., 2019	Sparse Graph	Residual Gated GCN	MLP Heatmap (NAR)	Beam Search	Immitation (SL)
Fu et al., 2020	Sparse Sub-graphs	Residual Gated GCN	MLP Heatmap (NAR)	MCTS	Immitation (SL)
Kool et al., 2021	Sparse Graph	Residual Gated GCN	MLP Heatmap (NAR)	Dynamic Programming	Immitation (SL)

Learning to Improve Sub-optimal Solutions

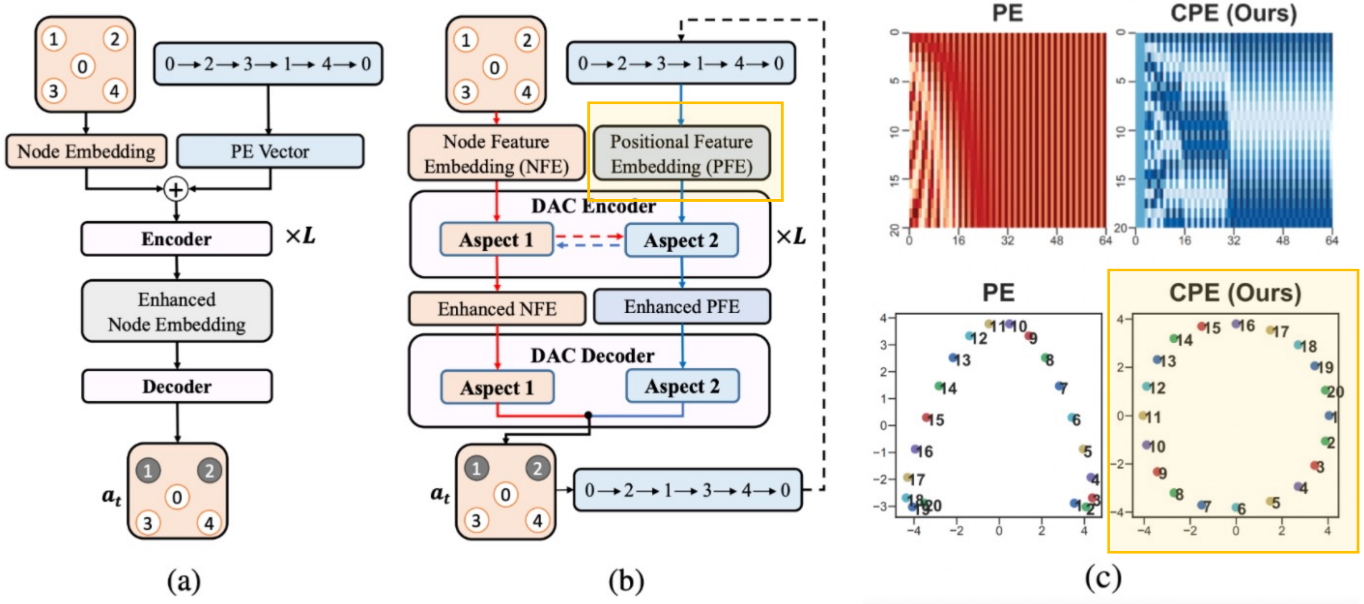
- **Learning** to iteratively **improve sub-optimal solutions**[1][2]:
  - a.k.a. learning to perform **local search**[3][4].
  - Alternative to 'constructive' AR and NAR decoding schemes**.**
- Learning to **guide decisions within** classical search heuristics (designed to work regardless of problem scale) → **implicitly better zero-shot generalization**.



[1] Chen and Tian, Learning to Perform Local Rewriting for Combinatorial Optimization, NeurIPS 2019
[2] Wu et al., Learning Improvement Heuristics for Solving Routing Problems, 2019 (TNNLS 2021)
[3] da Costa et al., Learning 2-opt Heuristics for the Traveling Salesman Problem via Deep Reinforcement Learning, AAAI 2021
[4] Hudson et al., Graph Neural Network Guided Local Search for the Traveling Salesperson Problem, ICLR 2022

Learning to Improve Sub-optimal Solutions

- Combined with symmetry: Dual-aspect Transformer[1] (structure + position updates) with **learnable cyclical positional encodings**.
- **'Neuralized' LKH algorithm[2]:** Up to 5,000 node TSPs at <1% optimality gap!



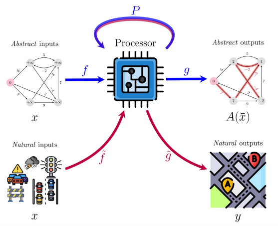
[1] Ma et al., Learning to Iteratively Solve Routing Problems with Dual-Aspect Collaborative Transformer, NeurIPS 2021
[2] Xin et al., NeuroLKH: Combining Deep Learning Model with Lin-Kernighan-Helsgaun Heuristic for Solving the Traveling Salesman Problem, NeurIPS 2021

Learning to Improve Sub-optimal Solutions

- **Potential limitation**: need for **hand-designed local search algorithms**, which may not exist for understudied COPs.

Paper	Definition	Graph Embedding	Solution Decoding	Solution Search	Policy Learning
Wu et al., 2021	Sequence + Position	Transformer Encoder	Transformer Decoder (L2I)	Local Search	Actor-critic (RL)
da Costa et al., 2020	Sequence	GCN	RNN + Attention (L2I)	Local Search	Actor-critic (RL)
Ma et al., 2021	Sequence + Cyclic Position	Dual Transformer Encoder	Dual Transformer Decoder (L2I)	Local Search	PPO + Curriculum (RL)
Xin et al., 2021	Sparse Graph	GAT	MLP Heatmap (NAR)	LKH Algorithm	Immitation (SL)
Hudson et al., 2021	Sparse Dual Graph	GAT	MLP Heatmap (NAR)	Guided Local Search	Immitation (SL)

Bonus: nice connections!



Learning Paradigms that Promote Generalization

- Explicit focus on **generalization** beyond SL and RL, **transfer learning**:
  - **Autoencoder** to learn a **continuous space** of routing problem solutions[1].
  - Neural solvers robust to **adversarial perturbations**[2].
  - **Fast finetuning** for adapting to each new TSP instance[3].

- **Pre-training revolution** from NLP[4]
  - What is the equivalent of **language modelling** for routing, e.g. TSP?
  - Can we transfer from **'easy' TSP** to more **complex VRPs**?

[1] Hottung et al., Learning a Latent Search Space for Routing Problems using Variational Autoencoders, ICLR 2021
[2] Geisler et al., Generalization of Neural Combinatorial Solvers Through the Lens of Adversarial Robustness, ICLR 2022
[3] Hottung et al., Efficient Active Search for Combinatorial Optimization Problems, arXiv 2021
[4] Ruder, NLP's ImageNet moment has arrived, 2018

Improved Evaluation Protocols

- Repeated calls for **more realistic evaluation** and **real-world impact**:
  - **Theory:** You are limited by **data generation** in the **NP-Hard** regime => you may be solving/measuring performance for a **simpler sub-problem** than the ‘real deal’[1].
  - **Practice:** Unrealistic **experiment design** and **evaluation protocols** => no real-world adoption from the OR community yet => **new guidelines**[2].

- Potential remedies:
  - **Real-world benchmarks**, e.g. TSPLib, CVRPLib.
  - **Community Competitions** on fresh data:
    - ML4CO @ NeurIPS 2021
    - AI4TSP @ IJCAI 2021

Machine Learning for
Combinatorial Optimization
COMPETITION 2021

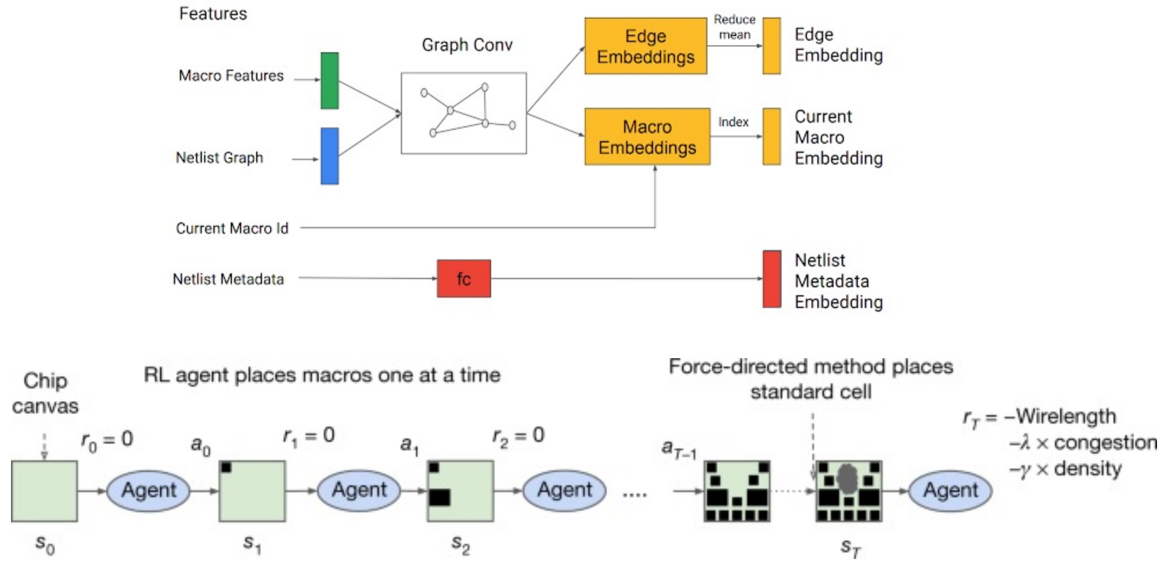
[1] Yehuda et al., It's Not What Machines Can Learn, It's What We Cannot Teach, ICML 2020.
[2] Accorsi et al., Guidelines for the Computational Testing of Machine Learning approaches to Vehicle Routing Problems, arXiv 2021.

More profound motivation?

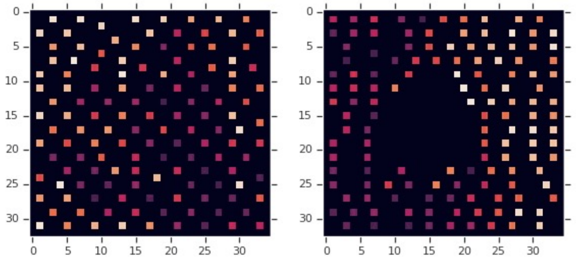
Neural Combinatorial Optimization can be used as a general tool for tackling **previously *un-encountered* NP-hard problems**, especially those that are non-trivial to design heuristics for[1].

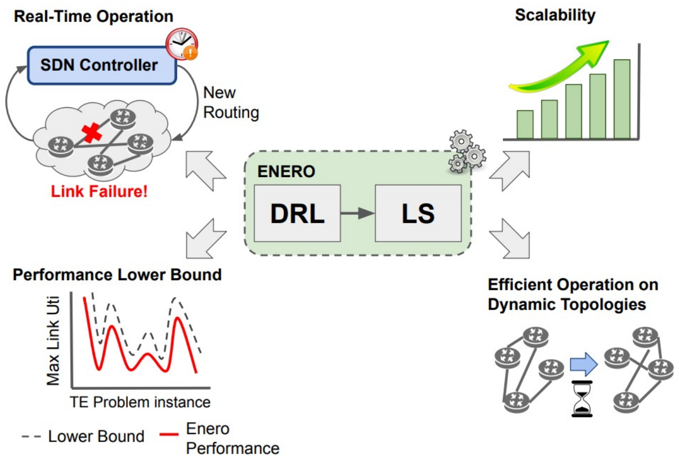
[1] Bello et al., Neural Combinatorial Optimization, ICLR 2017

Understudied Combinatorial Problems

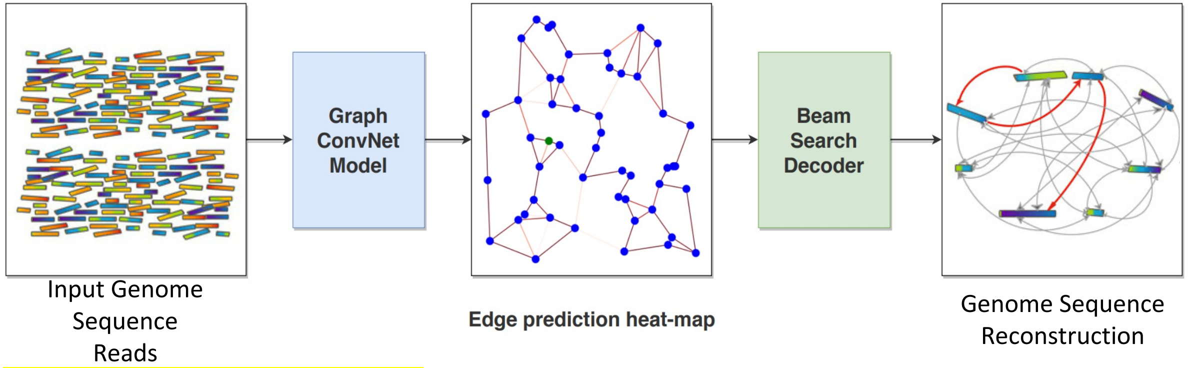


Chip Design[1]: GNN Encoder → CNN Decoder → RL to minimize chip metrics.

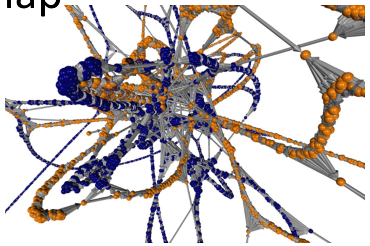




Communications Network Optimization[2]: GNN action selector → RL + Local Search to minimize traffic on WANs.

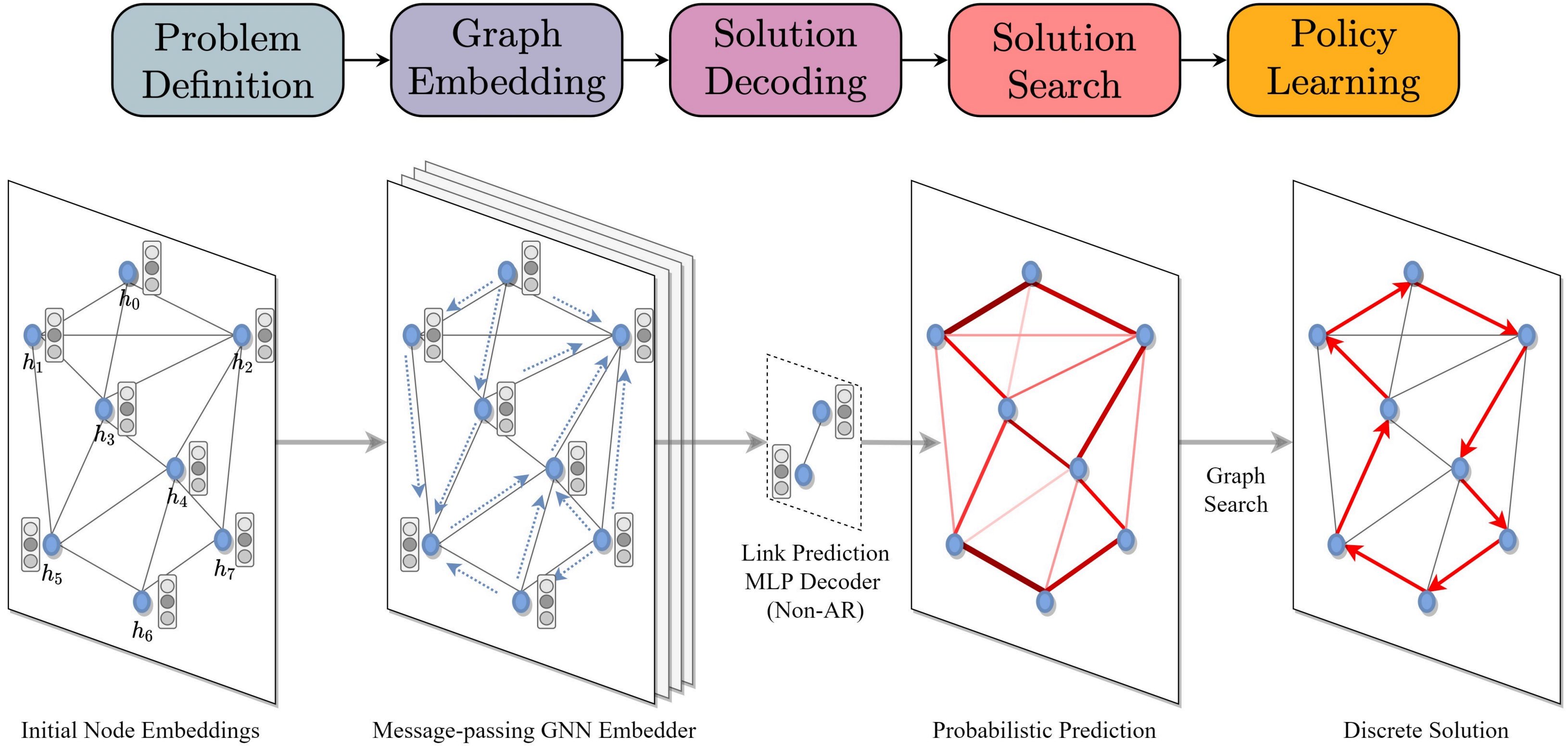


Genome Assembly[3]: GNN encoder → Heatmap → SL to reconstruct genome sequences.



[1] Mirhoseini et al., A graph placement methodology for fast chip design, **Nature 2021**
[2] Almasan et al., ENERO: Efficient Real-Time WAN Routing Optimization with Deep Reinforcement Learning, arXiv 2022
[3] Vrček et al., Genome Sequence Reconstruction Using Gated Graph Convolutional Network, openreview 2021.

Thank you for attending!



- Thank you to my co-author[1] (**R. Anand**), collaborators[2][3] (**X. Bresson**, **T. Laurent**, **Q. Cappart**, **L-M. Rousseau**), and people who gave feedback!
- Happy to chat about missing references, feedback, research, etc. – chaitjo@gmail.com

[1] Joshi and Anand, Recent Advances in Deep Learning for Routing Problems
[2] Joshi, Laurent, and Bresson, An Efficient Graph Convolutional Network for the TSP, arXiv 2019
[3] Joshi, Cappart, Rousseau, Laurent, Learning TSP Requires Rethinking Generalization, CP 2021