Reinforcement Learning for Combinatorial Optimization

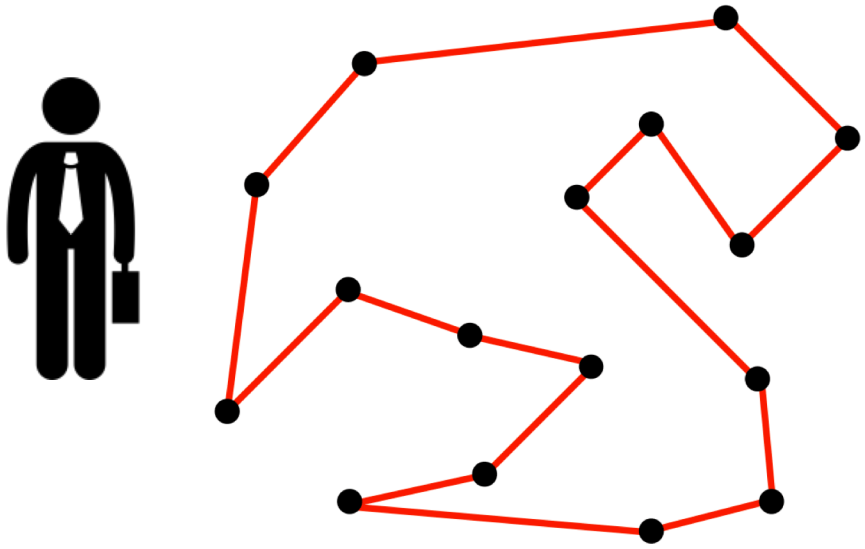
BYUNGSOO JEON

2ND YEAR PHD STUDENT IN CSD @ CMU

Combinatorial Optimization

Definition

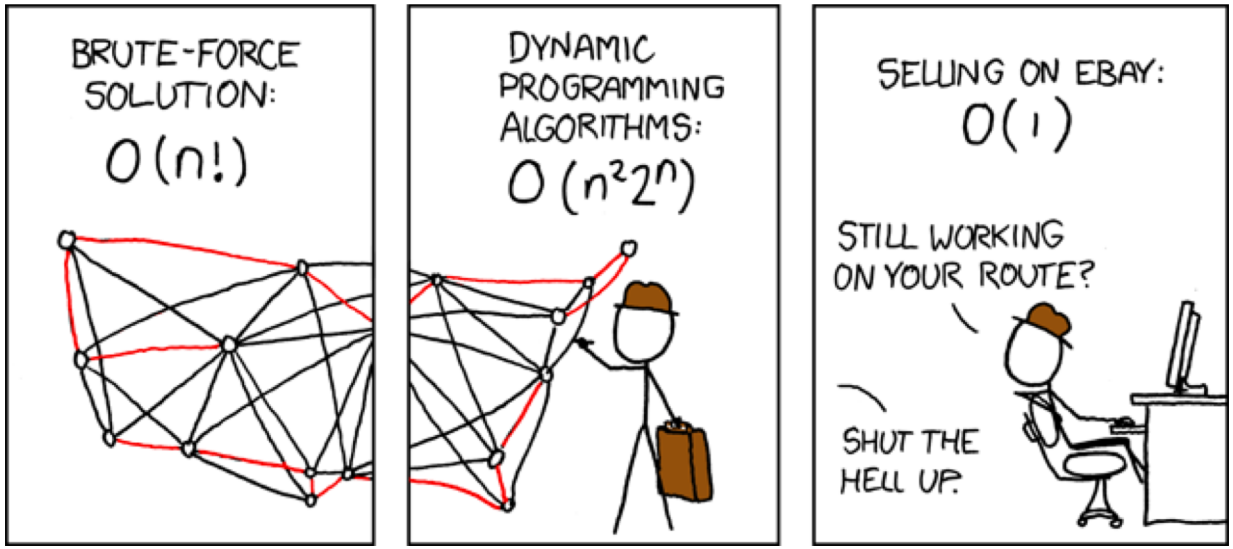
- The process of searching for maxima (or minima) of an objective function **F** whose domain is a **discrete** but **large configuration space**.



Combinatorial Optimization

Characteristic

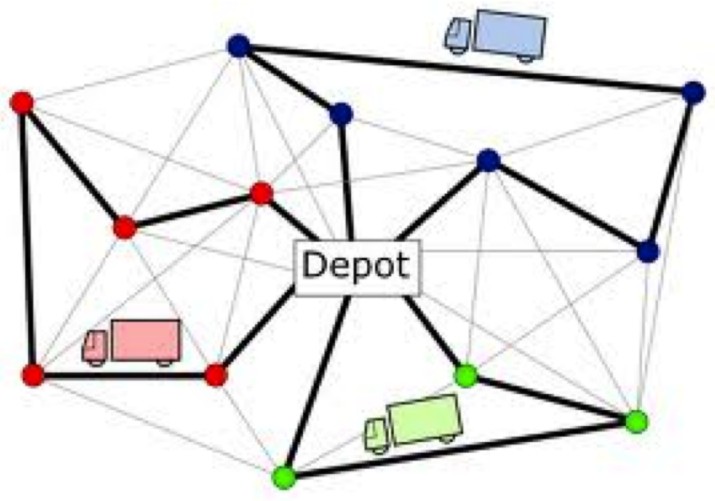
- It is often NP-hard (e.g. Traveling Salesman Problem)
- Verification of optimality for a given solution is hard
  - Branch-and-Bound algorithm often makes it easier

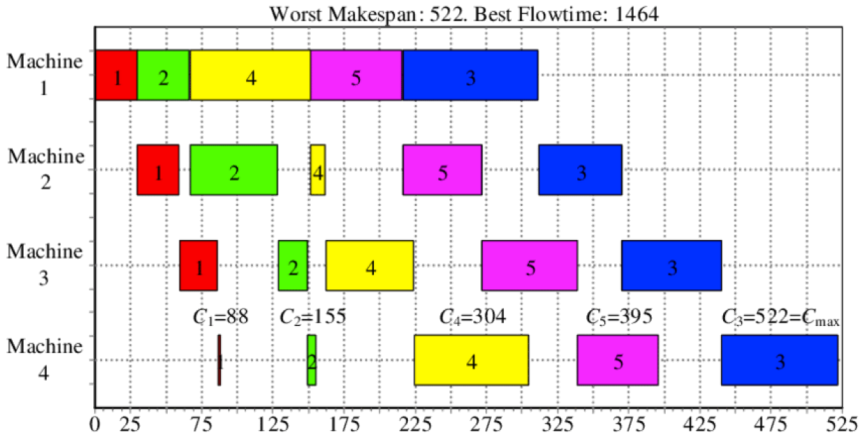


Motivation

Why is combinatorial optimization important?

- Countless application: supply chain, transportation, finance, energy, scheduling, etc.



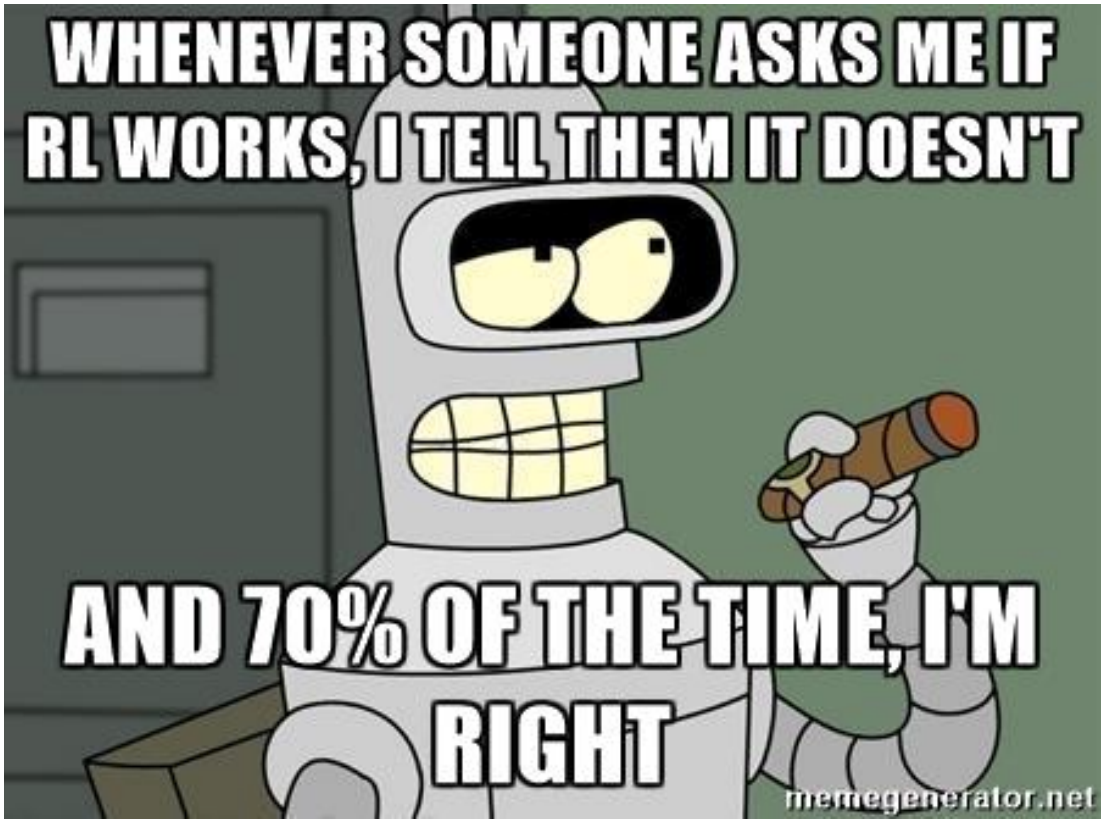


Motivation

Why is leveraging RL to solve such problems important?

- Heuristics can be **automated** or **augmented** through RL
- Produce better solutions in online settings where OR algorithms are bad at

Motivation


WHENEVER SOMEONE ASKS ME IF RL WORKS, I TELL THEM IT DOESN'T
AND 70% OF THE TIME, I'M RIGHT
memegenerator.net

Goals of Talk

Find out the specific challenges in applying RL to combinatorial optimization problems

Explore ways to leverage reinforcement learning to solve combinatorial optimization problems

Establish the possible connection of RL for combinatorial optimization to lifelong learning agent

Challenges

Feasibility

- Don’t know how far the generated solution is from the optimal or if it even respects the given constraints

Modeling

- The architectures used to learn good policies in CO may be very different from what is currently used

Generalization

- If a model trained on instances up to some size, the challenge exists in terms of generalization

Overview

	Feasibility	Modeling	Generalization	Note
PN [VFJ15]		✓	✓	Imitation
NCO [BPL16]	✓	✓		Pointer Network
RL [NOS18]		✓✓	✓	Attention

Pointer Network [VFJ15]

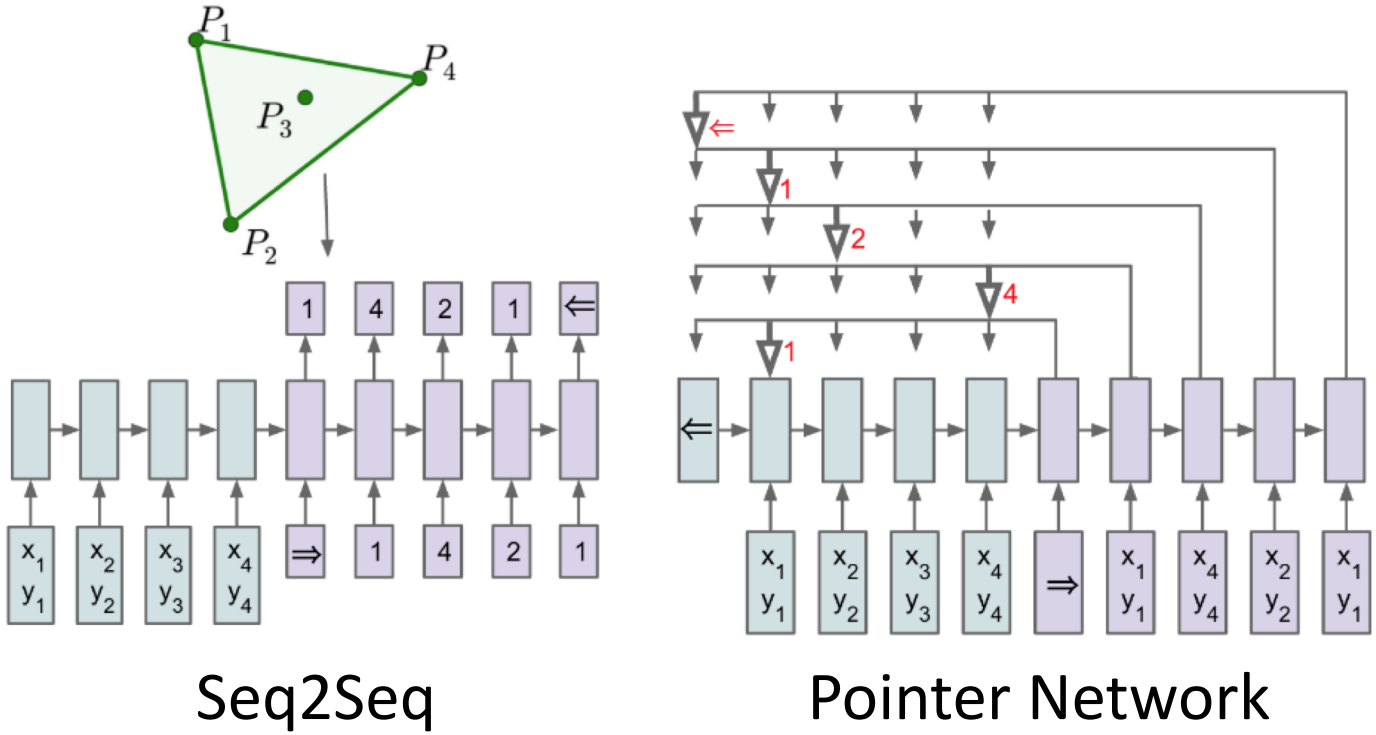
Problem

- Seq2Seq methods still require the size of the output dictionary to be fixed a priori

Contribution

- Propose a new architecture dealing with the fundamental problem of representing variable length dictionaries
- Show that the learned model generalizes to test problems with more points than the training problems

Pointer Network [VFJ15]



Seq2Seq

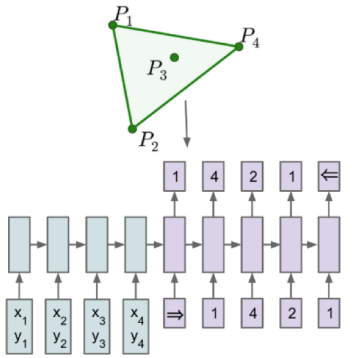
Pointer Network

Attention is used as pointers to the input elements!

Pointer Network [VFJ15]

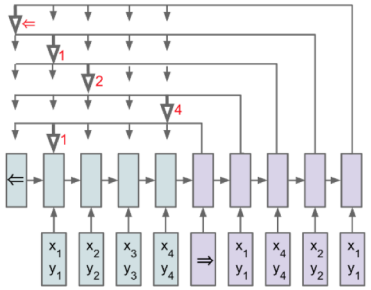
Seq2Seq

$$
\begin{aligned}
u_j^i &= v^T \tanh(W_1 e_j + W_2 d_i) \quad j \in (1, \ldots, n) \\
a_j^i &= \text{softmax}(u_j^i) \quad j \in (1, \ldots, n) \\
d_i' &= \sum_{j=1}^{n} a_j^i e_j
\end{aligned}
$$



Pointer Network

$$
\begin{aligned}
u_j^i &= v^T \tanh(W_1 e_j + W_2 d_i) \quad j \in (1, \ldots, n) \\
p(C_i | C_1, \ldots, C_{i-1}, \mathcal{P}) &= \text{softmax}(u^i)
\end{aligned}
$$



Distribution of the attention is the answer!

Pointer Network [VFJ15]

Result

Application	Baselines	Problem Size	Generalization Power Test
Convex Hull	Seq2Seq Seq2Seq + Attn	Train: <= 50 Infer: <= 500	Yes
Delaunay Triangulation	N/A	<= 50	No
TSP	Optimal Algorithm 1 Algorithm 2 Algorithm 3	<= 50	Yes

Pointer Network [VFJ15]

Limitation

- Dependence on the labels
- It requires a huge number of instances in order to train this model

NCO with RL [BPL16]

Problem

- Supervised learning for CO problems is often not enough

Contribution

- Propose a framework to tackle CO problems (mainly TSP) using neural network and reinforcement learning **without optimal labels**
- It is proven to achieve close to optimal results on Euclidean TSP and Knapsack problems

NCO with RL [BPL16]

Model

- Designed for Euclidean TSP
- Cost function: Expected tour length
  - Reward: negative expected tour length
- Training scheme: REINFORCE with Actor-critic training
- Actor and critic: Pointer Network

Search Strategy

- Sampling from the stochastic policy
- Active Search

NCO with RL [BPL16]

Result

Application	Baselines	Problem Size	Generalization Power Test
Euclidean TSP	Optimal Pointer Network Google OR Tool Greedy heuristic Active Search (AS) RL-greedy RL-greedy@16 RL-sampling RL-AS	<= 100	No
0-1 Knapsack		<= 200	No

NCO with RL [BPL16]

Limitation

- It assumes the environment is static over time
- Searching at inference time is required to achieve better performance than Google OR tool
- Lack of the evaluation for generalization power
- Fail to justify the need for RL on 0–1 Knapsack problems

RL for VRP [NOS18]

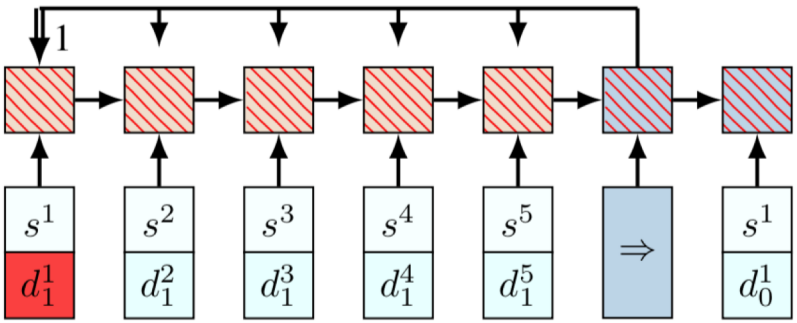
Problem

- Modeling dynamic components of a system is challenging

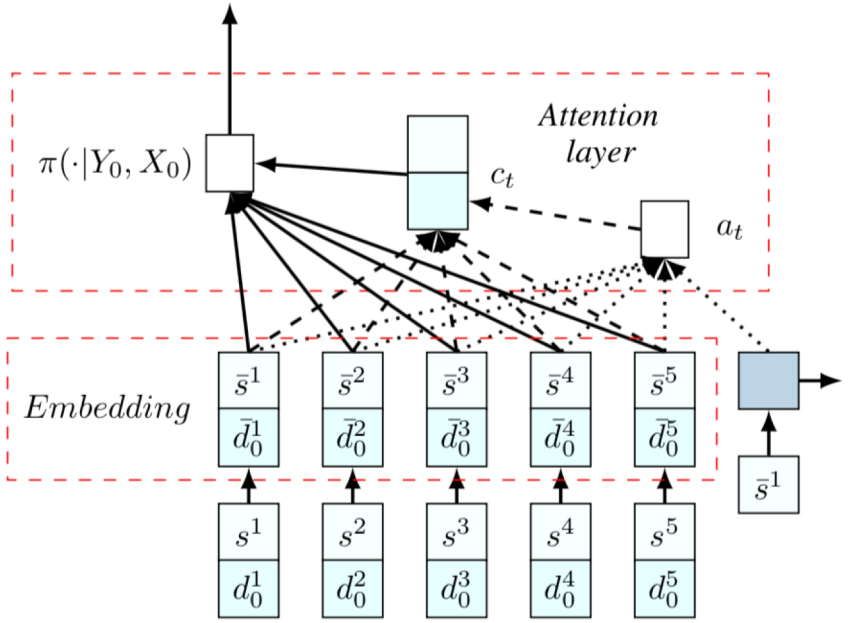
Contribution

- It handles dynamic elements of the system with static and dynamic element embeddings
- It produces near-optimal solution in less than a second and performs well on any problem sampled from a given distribution

RL for VRP [NOS18]



Pointer Network



Proposed Model

RL for VRP [NOS18]

Result

Application	Baselines	Problem Size	Generalization Power Test
VRP	CW-Greedy CW-Rnd(5,5) CW-Rnd(10,10) SW-Basic SW-Rnd(5) SW-Rnd(10) Google OR-Tool RL-greedy RL-BS(5) RL-BS(10)	<= 100	Yes

RL for VRP [NOS18]

Limitation

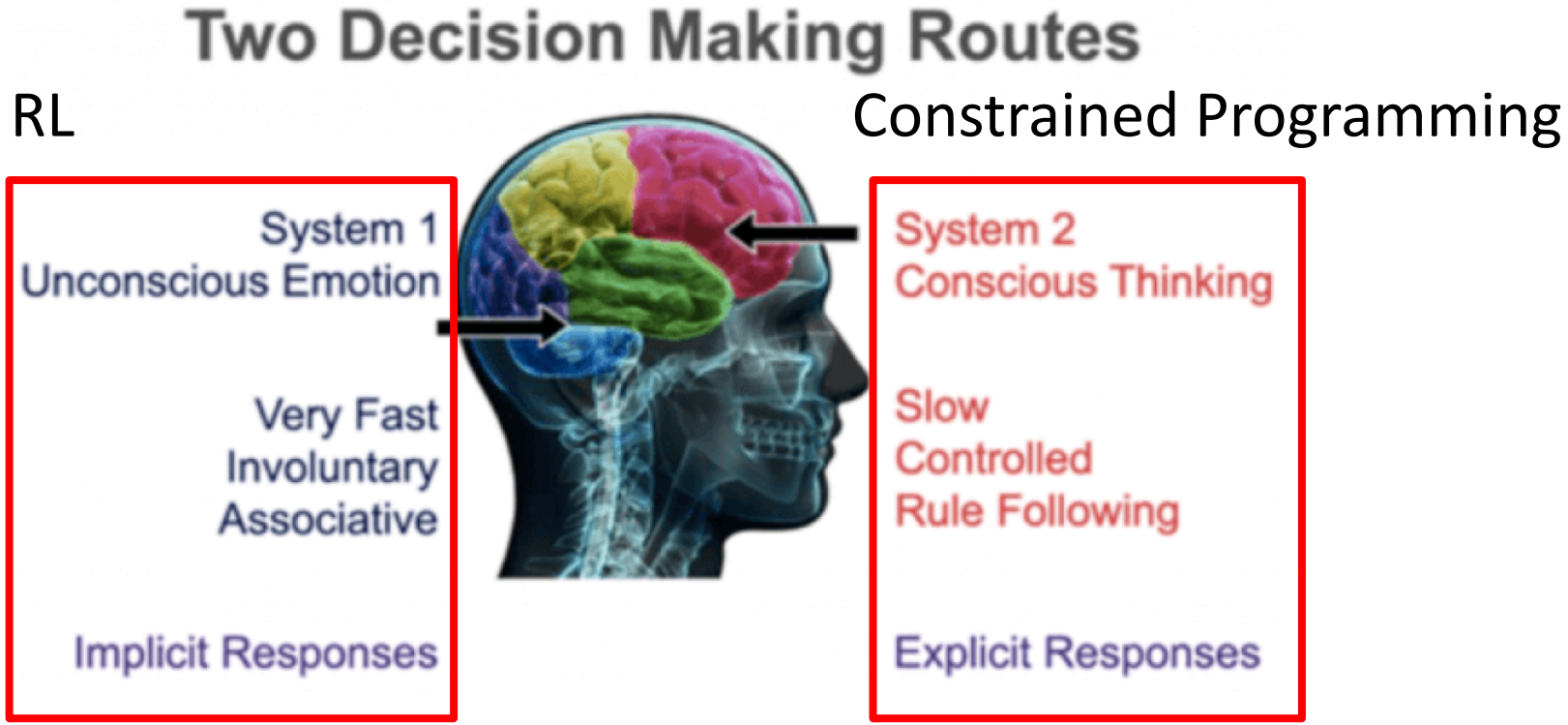
- Degradation when the testing problems are very different from the training ones
- Limited evaluation on the stochastic VRP case while it was told as a main contribution of this paper

Table 5: Satisfied demand under different strategies.

Method	Random	Largest-Demand	Max-Reachable	A3C
Avg. Dem.	24.83	75.11	88.60	112.21
% satisfied	5.4%	16.6%	19.6%	28.8%

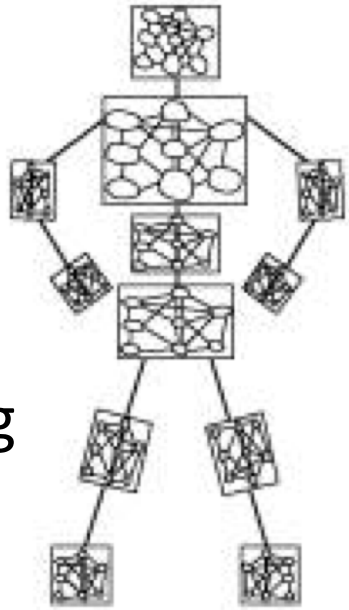
Connection to Lifelong learner

Sequential decision making process can be modeled as a combinatorial optimization



Connection to Lifelong learner

Constrained Programming (Symbolic)



RL or ML (Connectionist)

Good playground to train lifelong learning agent

- Symbolic vs. Connectionist
- Easy to visualize learned (meta)heuristics or constraints
- Test with online stochastic environment, which is realistic

Question or Discussion

References

[BLP18] Yoshua Bengio et al., “Machine learning for combinatorial optimization: a methodological tour d’horizon”, arXiv 2018

[HoT85] John J Hopfield et al., ““Neural” computation of decisions in optimization problems”, Biological cybernetics, 1985

[Smi99] Kate A Smith, “Neural networks for combinatorial optimization: a review of more than a decade of research”, INFORMS Journal on Computing, 1999.

[VFJ15] Oriol Vinyals et al., “Pointer networks”, NIPS 2015

[VBK16] Oriol Vinyals et al., “Order matters: Sequence to sequence for sets”, ICLR 2016.

[DKZ17] Hanjun Dai et al., “Learning combinatorial optimization algorithms over graphs”, NIPS 2017

[BPL16] Irwan Bello, et al., “Neural combinatorial optimization with reinforcement learning”, arXiv 2016

References

[VSP17] Ashish Vaswani et al., “Attention is all you need”, NIPS 2017

[NVB17] Alex Nowak et al., “A note on learning algorithms for quadratic assignment with graph neural networks”, arXiv 2017

[NOS18] Mohammadreza Nazari et al., “Reinforcement learning for solving the vehicle routing problem”, NIPS 2018.

[KHW19] Wouter Kool, et al. “Attention, learn to solve routing problems!”, ICLR 2019

[KaW18] Yoav Kaempfer et al., “Learning the multiple traveling salesmen problem with permutation invariant pooling networks”, Arxiv 2018

[DCL18] Michel Deudon et al., “Learning heuristics for the TSP by policy gradient”, International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research 2018

[BBB19] Bharathan Balaji et al., “ORL: Reinforcement Learning Benchmarks for Online Stochastic Optimization Problems”, ICML Workshop 19

[KLM19] Weiwei Kong et al., “A new dog learns old tricks: Rl findsclassic optimization algorithms”, ICLR 2019