Universiteit van Amsterdam

Informatics Institute - Faculty of Science

UvA - BOSCH DELTA LAB

AMLAB Amsterdam Machine Learning Lab

Learning new heuristics for combinatorial problems

Herke van Hoof

Learning for combinatorial optimisation

Learning for combinatorial optimisation

Traveling scientists / salesman problem (TSP)

Kool, van Hoof & Welling, ICLR 2019

Learning for combinatorial optimisation

Traveling scientists / salesman problem (TSP)

NP-hard, so no polynomial-time complete solvers (probably)

Large problems solved using hand-crafted heuristics

Limitations of such heuristics?

Kool, van Hoof & Welling, ICLR 2019

Learning for combinatorial optimisation

On well-known problem like TSP, decades of optimisations have yielded powerful heuristics, but:

- In practice, almost always have additional objectives or constraints: heuristics might not work
- ‘Best’ heuristic depends on the type of problem. How to choose which one to use?

Kool, van Hoof & Welling, ICLR 2019

Learning for combinatorial optimisation

On well-known problem like TSP, decades of optimisations have yielded powerful heuristics, but:

- In practice, almost always have additional objectives or constraints: heuristics might not work
- ‘Best’ heuristic depends on the type of problem. How to choose which one to use?

Instead, try *learning a heuristic* appropriate for current problem formulation and instance type

Kool, van Hoof & Welling, ICLR 2019

How does that work?

In a nutshell….

Kool, van Hoof & Welling, ICLR 2019

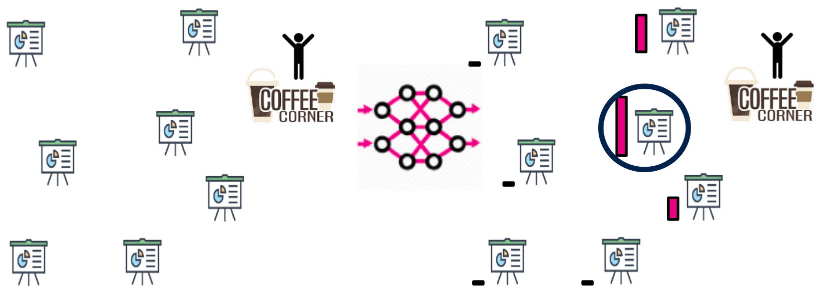
How does that work?

In a nutshell….

Kool, van Hoof & Welling, ICLR 2019

How does that work?

In a nutshell….



Kool, van Hoof & Welling, ICLR 2019

How does that work?

In a nutshell….

Kool, van Hoof & Welling, ICLR 2019

How does that work?

In a nutshell….

Kool, van Hoof & Welling, ICLR 2019

How does that work?

In a nutshell….

Kool, van Hoof & Welling, ICLR 2019

How does that work?

In a nutshell….

Kool, van Hoof & Welling, ICLR 2019

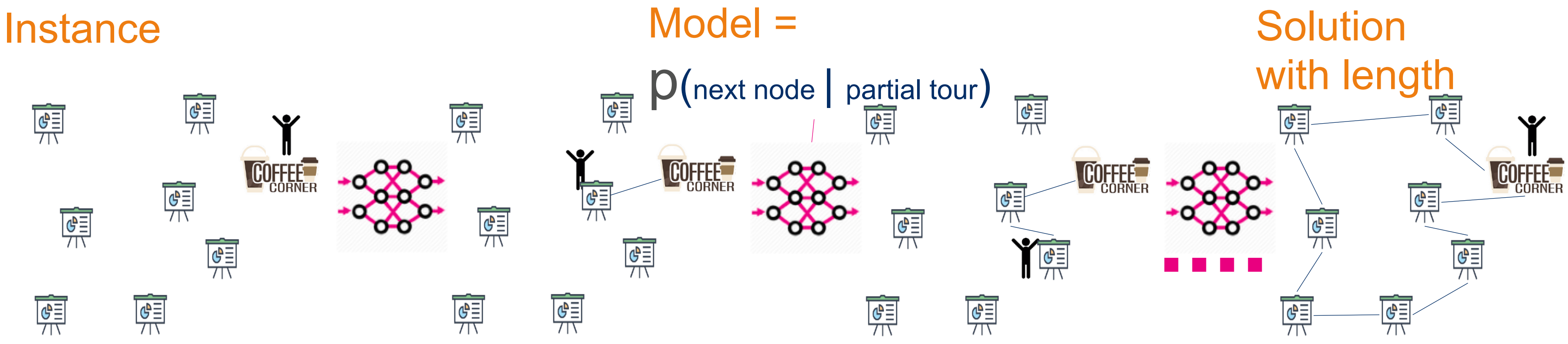
How does that work?

In a nutshell….

Kool, van Hoof & Welling, ICLR 2019

How does that work?

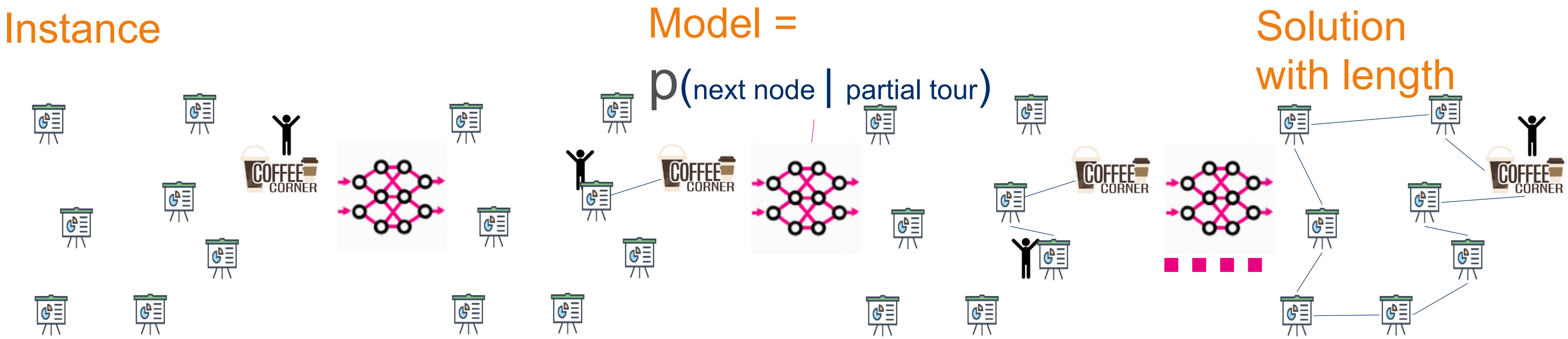
In a nutshell….



Kool, van Hoof & Welling, ICLR 2019

How does that work?

In a nutshell….



Randomized algorithm defined by network parameters θ

Kool, van Hoof & Welling, ICLR 2019

How does that work?

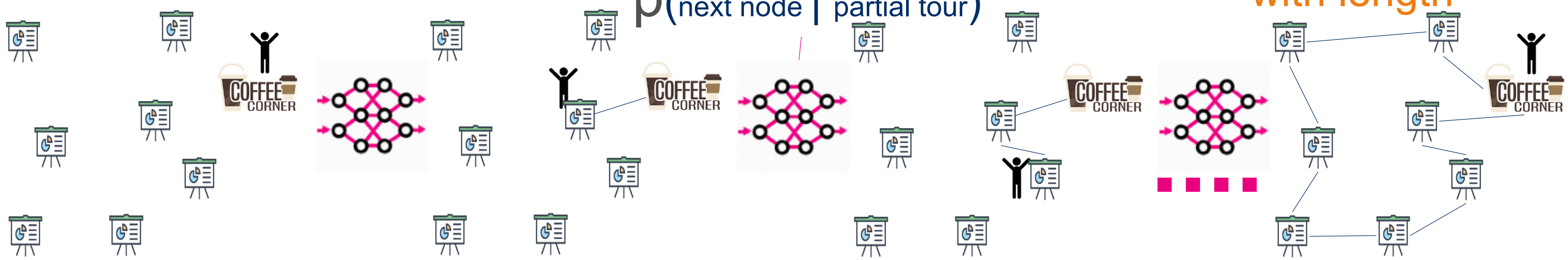
In a nutshell….

Instance

Model =

p(next node | partial tour)

Solution
with length



Randomized algorithm defined by network parameters θ

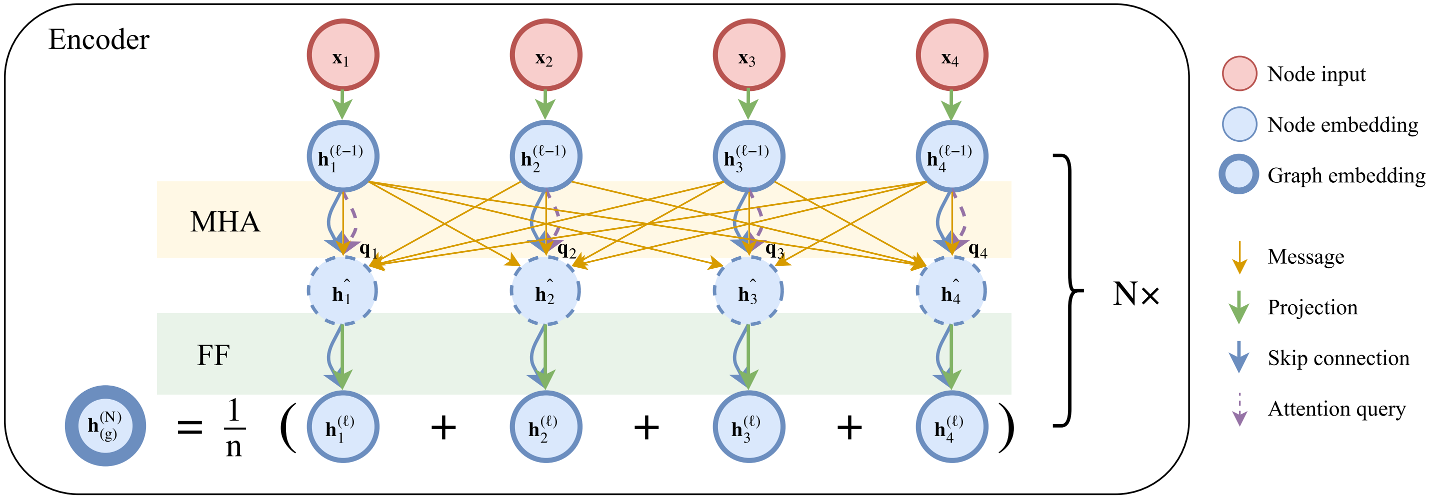
Optimize expected cost

$$\mathbb{E}_{p_{\theta}(\tau|s)}[L(\tau)]$$

Kool, van Hoof & Welling, ICLR 2019

Attention model architecture

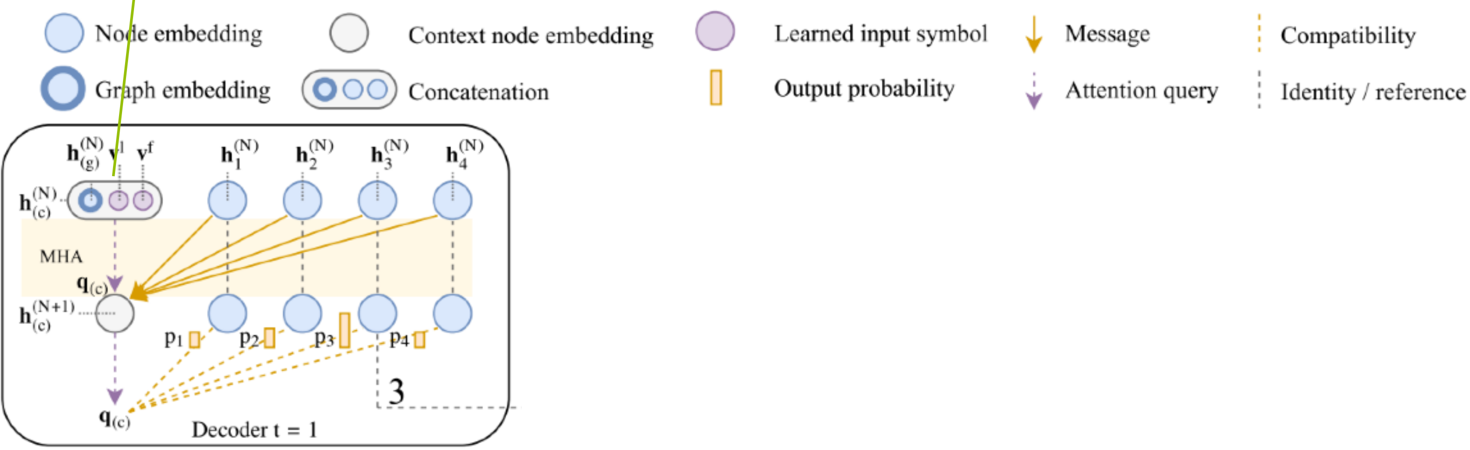
Architecture using graph convolutions



Kool, van Hoof & Welling, ICLR 2019

Attention model architecture

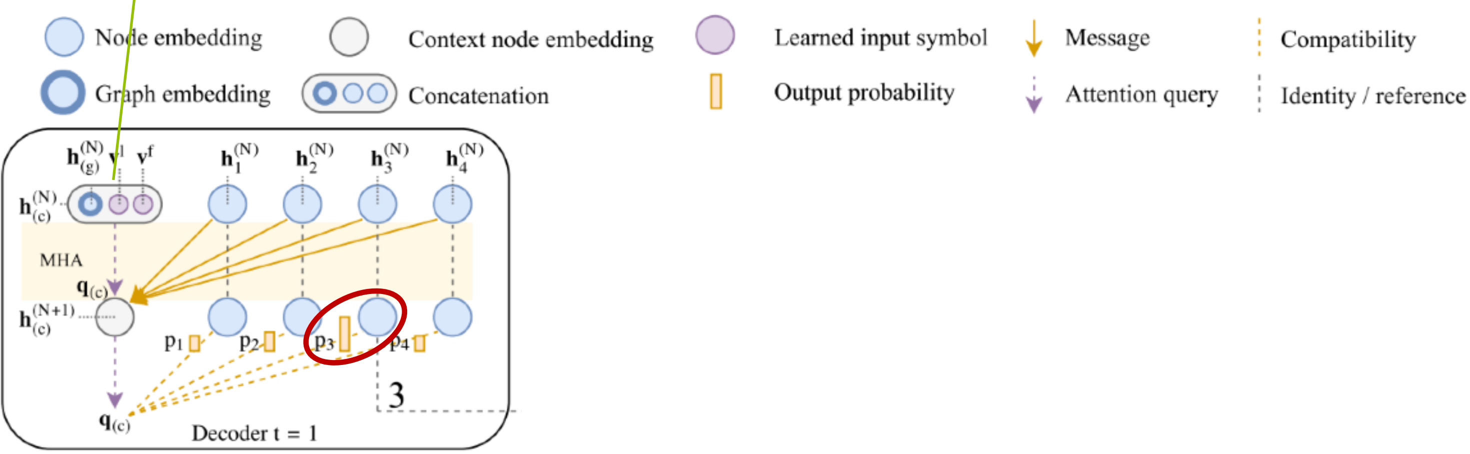
Decoder context:
(graph, first node, last node)



Kool, van Hoof & Welling, ICLR 2019

Attention model architecture

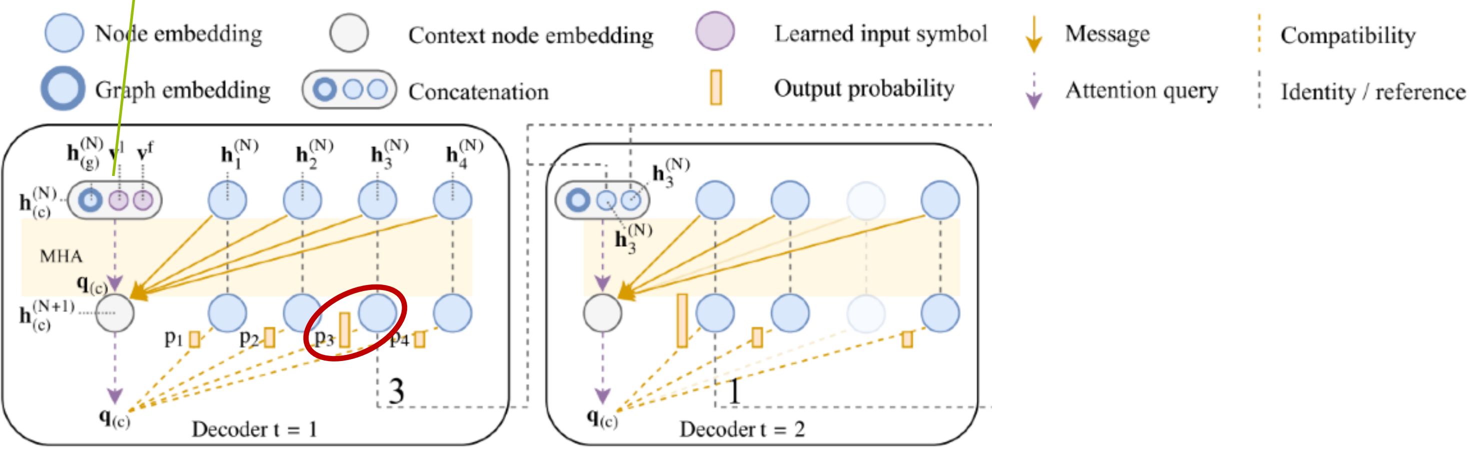
Decoder context:
(graph, first node, last node)



Kool, van Hoof & Welling, ICLR 2019

Attention model architecture

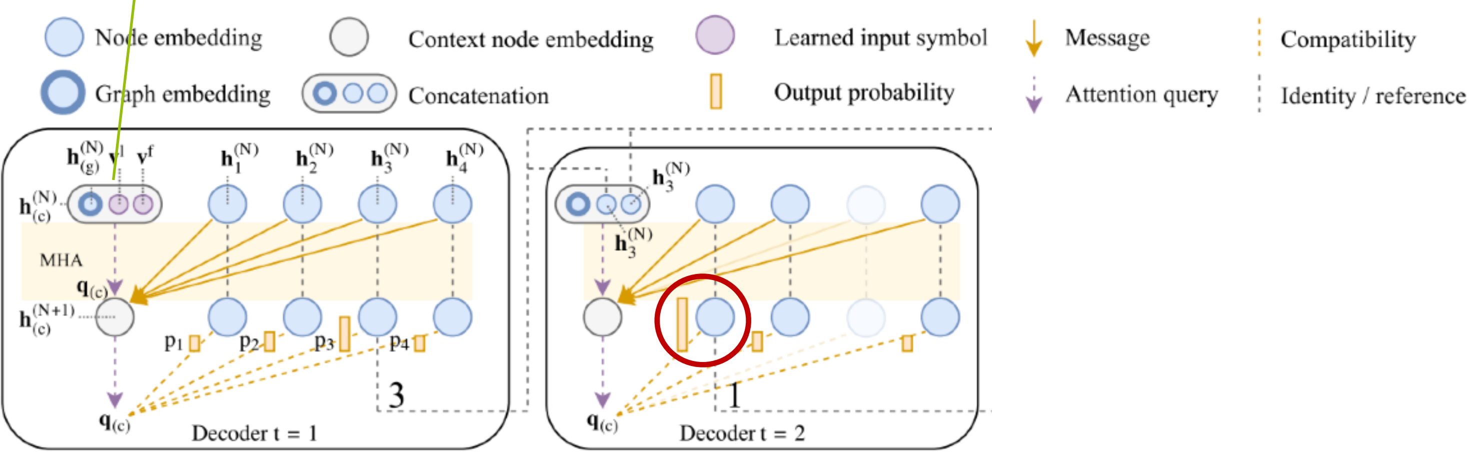
Decoder context:
(graph, first node, last node)



Kool, van Hoof & Welling, ICLR 2019

Attention model architecture

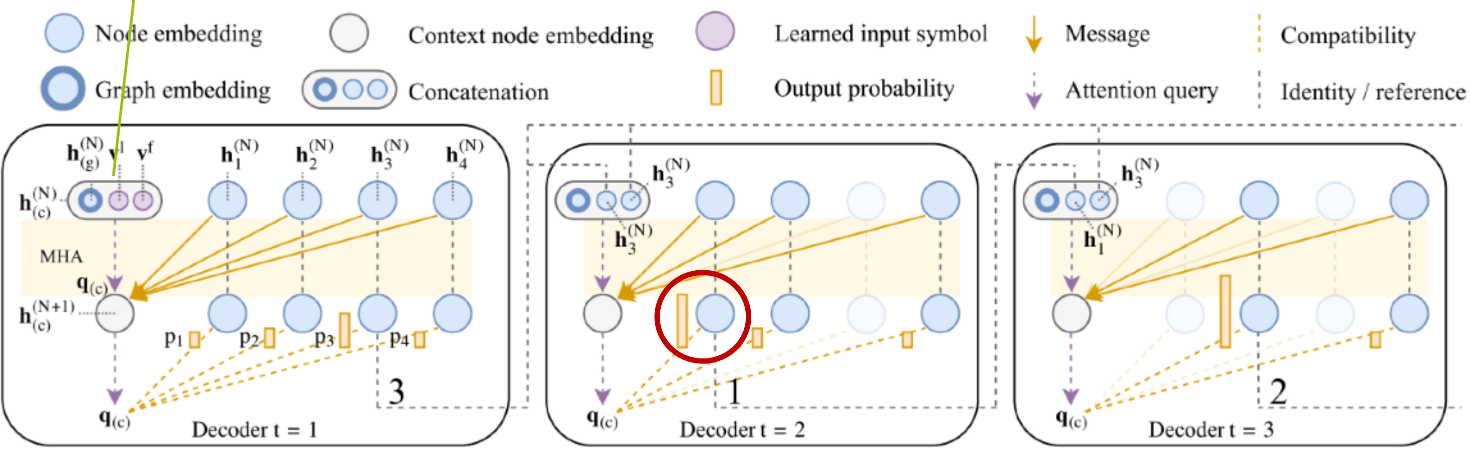
Decoder context:
(graph, first node, last node)



Kool, van Hoof & Welling, ICLR 2019

Attention model architecture

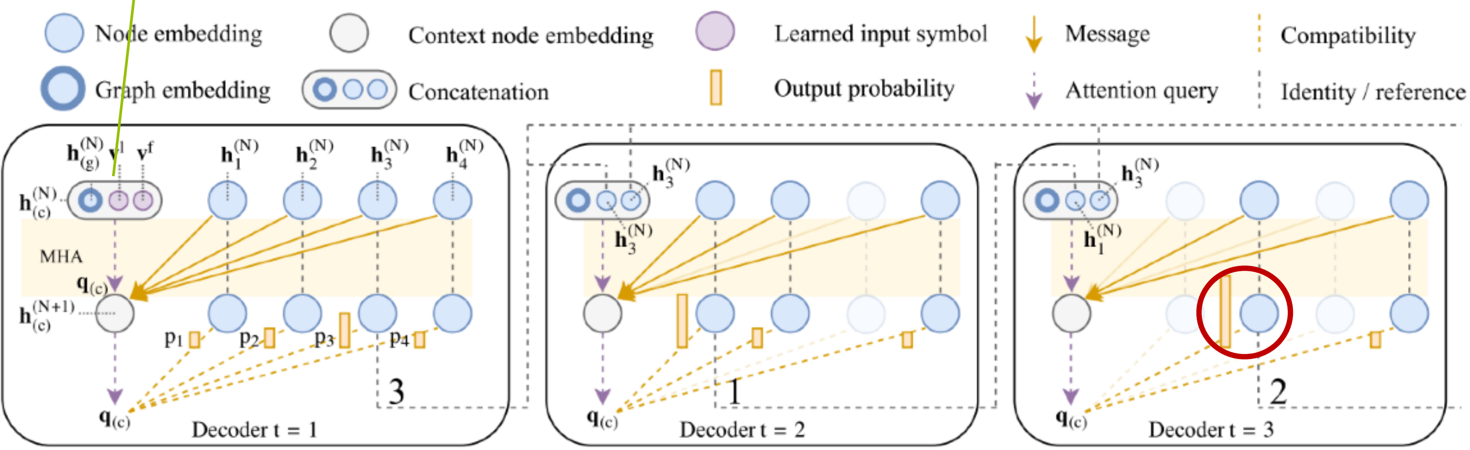
Decoder context:
(graph, first node, last node)



Kool, van Hoof & Welling, ICLR 2019

Attention model architecture

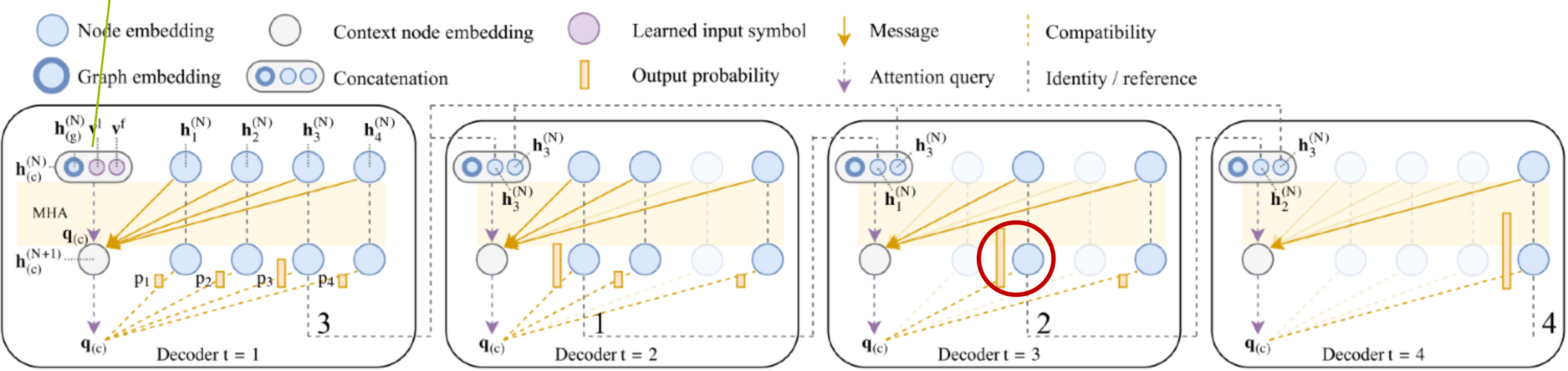
Decoder context:
(graph, first node, last node)



Kool, van Hoof & Welling, ICLR 2019

Attention model architecture

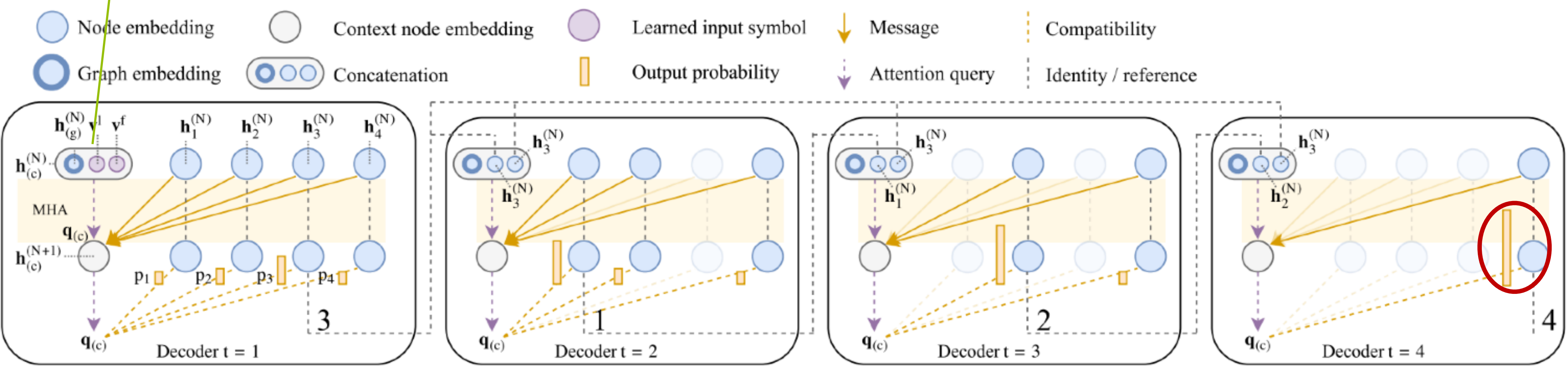
Decoder context:
(graph, first node, last node)



Kool, van Hoof & Welling, ICLR 2019

Attention model architecture

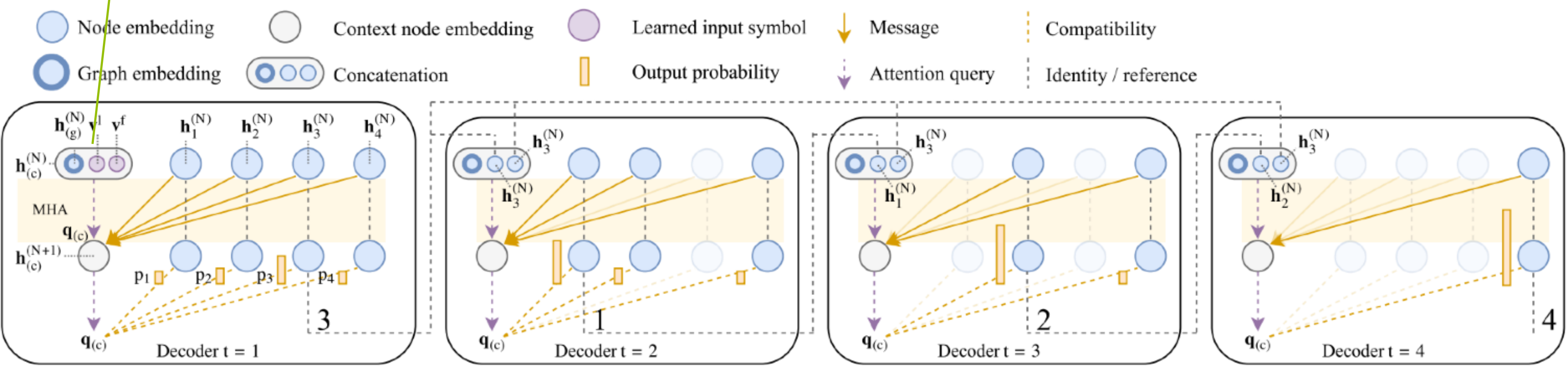
Decoder context:
(graph, first node, last node)



Kool, van Hoof & Welling, ICLR 2019

Attention model architecture

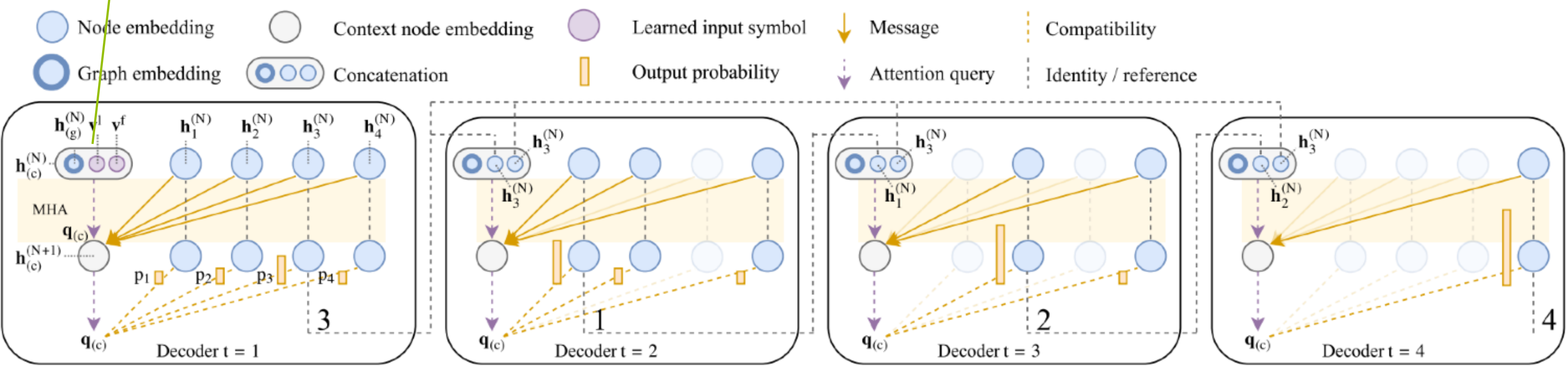
Decoder context:
(graph, first node, last node)



Kool, van Hoof & Welling, ICLR 2019

Attention model architecture

Decoder context:
(graph, first node, last node)



$$\tau = (3, 1, 2, 4)$$

Kool, van Hoof & Welling, ICLR 2019

Parameter optimisation

Randomized algorithm with expected cost: $$\mathbb{E}_{p_{\theta}(\tau|s)}[L(\tau)]$$

Evaluation requires sum over all tours

Kool, van Hoof & Welling, ICLR 2019

Parameter optimisation

Randomized algorithm
with expected cost:

$$\mathbb{E}_{p_{\theta}(\tau|s)}[L(\tau)]$$

Evaluation requires sum over all tours

How to optimize

NEURAL COMBINATORIAL OPTIMIZATION
WITH REINFORCEMENT LEARNING

Irwan Bello*, Hieu Pham*, Quoc V. Le, Mohammad Norouzi, Samy Bengio
Google Brain
{ibello,hyhieu,qvl,mnorouzi,bengio}@google.com

Kool, van Hoof & Welling, ICLR 2019

Parameter optimisation

Randomized algorithm with expected cost: $\mathbb{E}_{p_{\theta}(\tau|s)}[L(\tau)]$

How to optimize

NEURAL COMBINATORIAL OPTIMIZATION WITH REINFORCEMENT LEARNING

Irwan Bello*, Hieu Pham*, Quoc V. Le, Mohammad Norouzi, Samy Bengio
Google Brain
{ibello,hyhieu,qvl,mnorouzi,bengio}@google.com

Evaluation requires sum over all tours

Reinforcement learning approximates gradient using samples from $p_\theta(\tau|s)$ (Bello)

'good tours': probability ↑

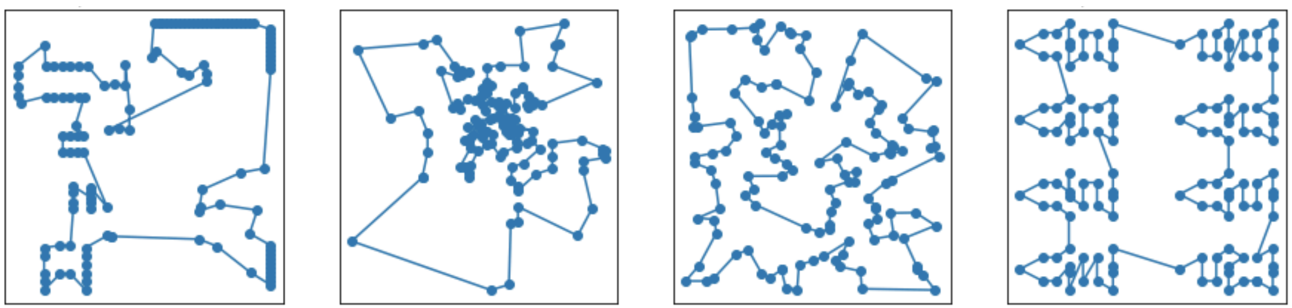
'bad tours': probability ↓

Decide whether τ 'good' or 'bad' by comparing to most likely tour (Kool)

Kool, van Hoof & Welling, ICLR 2019

Learning TSP heuristics

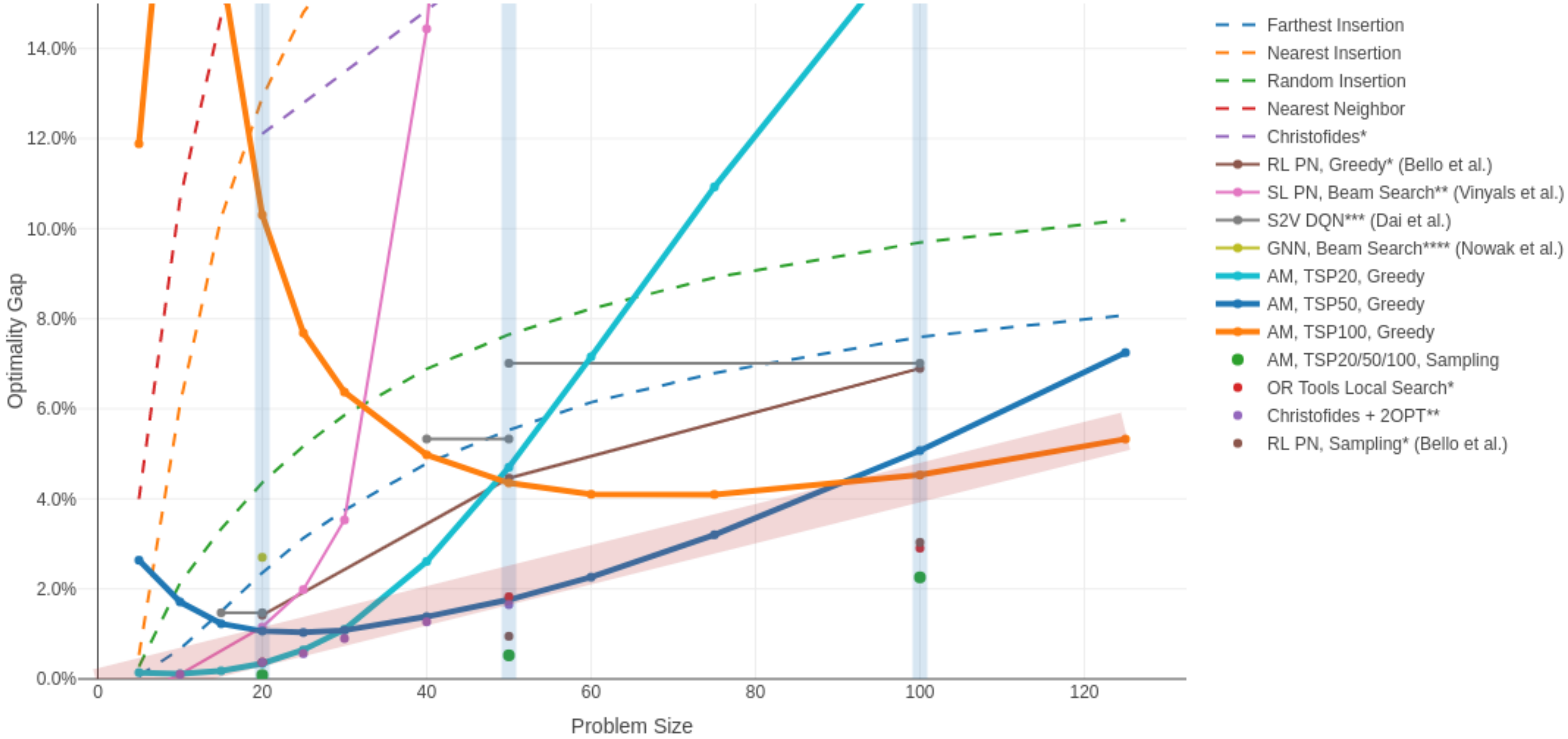
Now we have a model and an algorithm, we just need to generate many problems and keep training….



- Generate a problem
- ‘Guess’ a tour using current network parameters
- < length of most likely tour: increase probability using backprop (and vice versa)
- repeat…

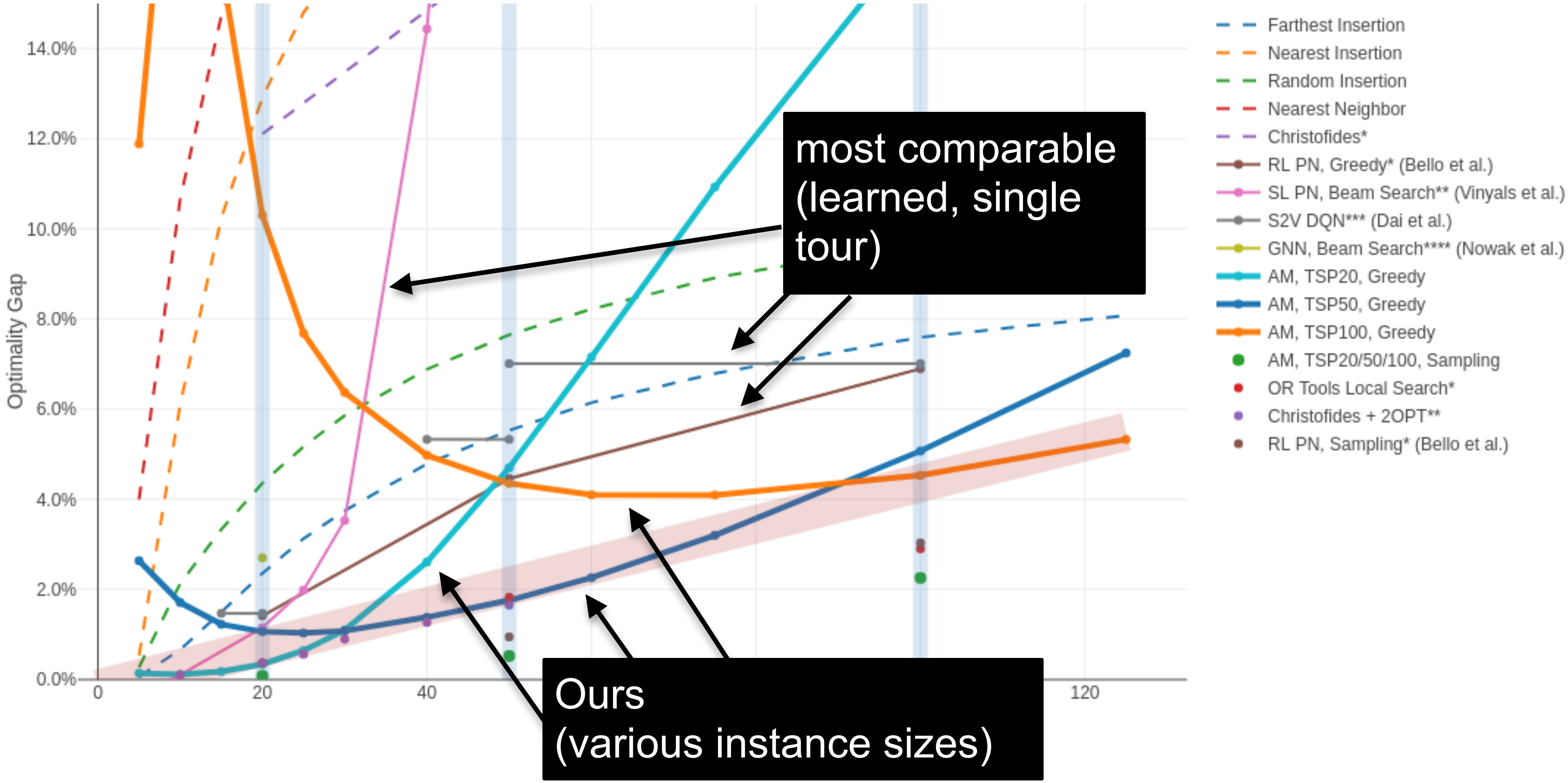
Kool, van Hoof & Welling, ICLR 2019

Performance



Kool, van Hoof & Welling, ICLR 2019

Performance



Kool, van Hoof & Welling, ICLR 2019

Further improvements

If we are sampling multiple tours, we might sample duplicates. In this setting, duplicates don't provide information.

- Can we sample 'without replacement'?

For a single learning step, generate two solutions (sampled tour & most likely tour)

- Can we compare sampled tours to each other, and thus eliminate the need to generate 'baseline' solutions?

Further improvements

Reinforcement learning

Learning involves *random sampling* from the model.

Randomness ensures exploration of new potential solutions

Duplicate samples do not give us any new information.

Direct search (BFS/DFS/BS)

Direct search for best route

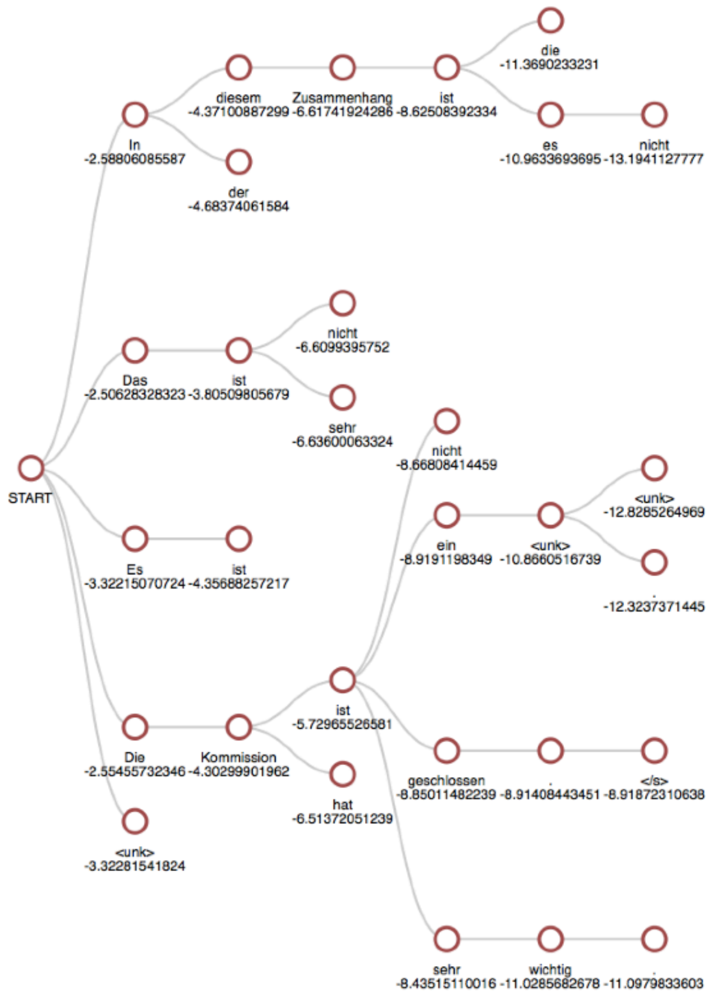
Deterministic process (doesn't explore, limited use for learning)

Multiple good routes without duplicates

Can we combine best of both worlds?

Kool, van Hoof & Welling, ICML 2019

Beam search



OpenNMT/MIT licence

Further improvements

We can get the best of both worlds!

Stochastic beam search provides ‘sampling without replacement’ for sequence models

In NLP experiments, showed:

- As optimiser, a good trade-off between *higher diversity* and *performance* in finding good translations
- As sampler, *lower variance* than sampling-with-replacement schemes

How about its use in combinatorial optimisation?

Kool, van Hoof & Welling, ICML 2019

Further improvements

If we are sampling multiple tours, we might sample duplicates. In this setting, duplicates don't provide information.

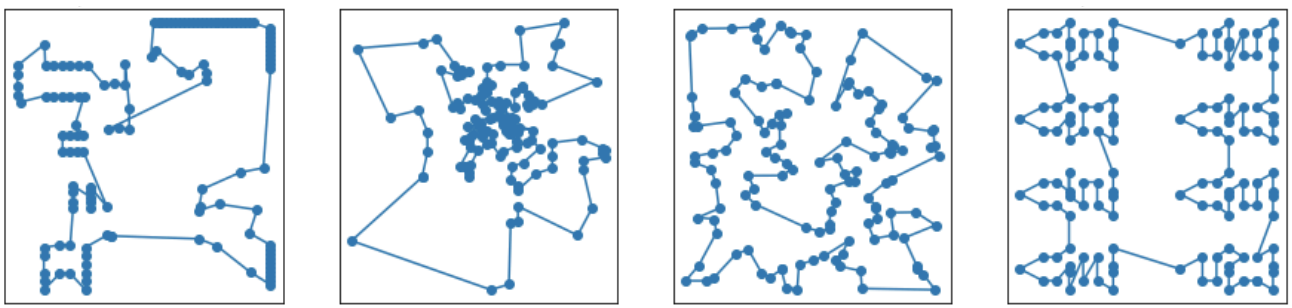
- Can we sample 'without replacement'?

For a single learning step, generate two solutions (sampled tour & most likely tour)

- Can we compare sampled tours to each other, and thus eliminate the need to generate 'baseline' solutions?

Stochastic beam search for faster learning

Remember the training procedure

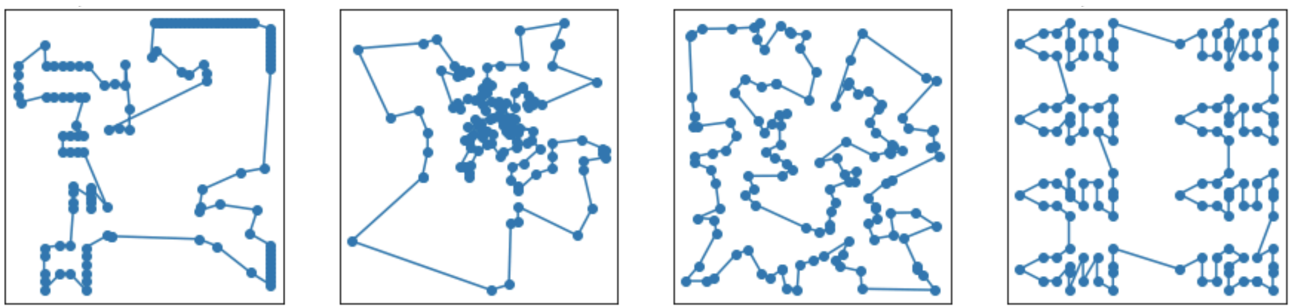


- Generate a problem
- ‘Guess’ a tour using current network parameters
- < most likely tour length: increase probability using backprop (and vice versa)
- repeat…

Kool, van Hoof & Welling, ICLR structured prediction WS 2019
Kool, van Hoof & Welling, ICLR 2020

Stochastic beam search for faster learning

Remember the training procedure

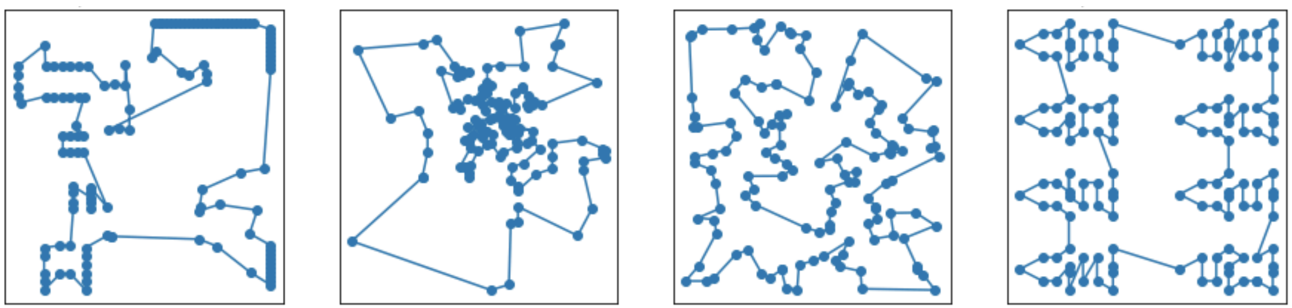


- Generate a problem
- ‘Guess **k tours** using current network parameters **without replacement**
- < most likely tour length: increase probability using backprop (and vice versa)
- repeat…

Kool, van Hoof & Welling, ICLR structured prediction WS 2019
Kool, van Hoof & Welling, ICLR 2020

Stochastic beam search for faster learning

Remember the training procedure



- Generate a problem
- ‘Guess **k tours** using current network parameters **without replacement**
- < **average** tour length: increase probability using backprop (and vice versa)
- repeat…

Kool, van Hoof & Welling, ICLR structured prediction WS 2019
Kool, van Hoof & Welling, ICLR 2020

Stochastic beam search for faster learning

Using samples without replacement changes the sampling distribution!

- Importance weights? (high variance)
- Normalized importance weights (biased!)

Instead of treating sampled elements independently, can derive a different gradient estimator based on complete sampled set:

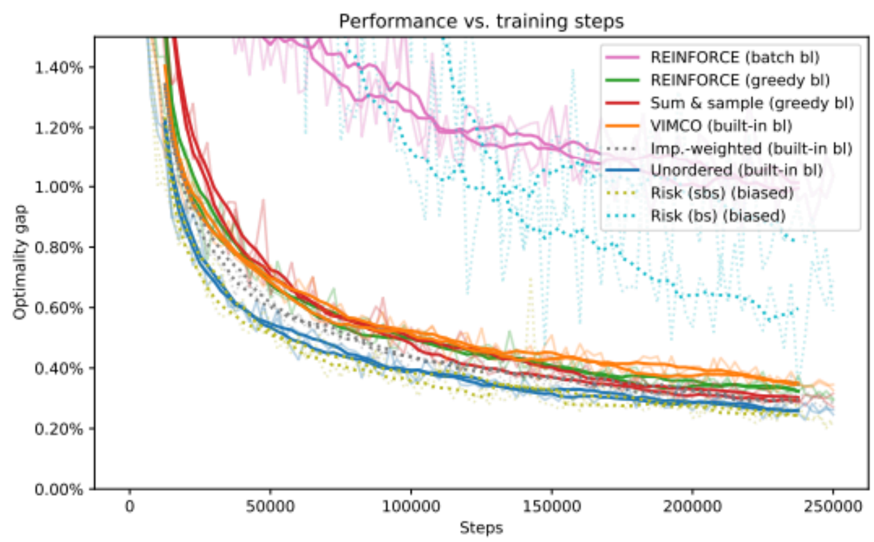
Unordered set policy gradient estimator

Lower-variance estimate that is still unbiased!

Kool, van Hoof & Welling, ICLR 2019 WS
Kool, van Hoof & Welling, ICLR 2020

Stochastic beam search for faster learning

Traveling salesman problem using low-variance estimator

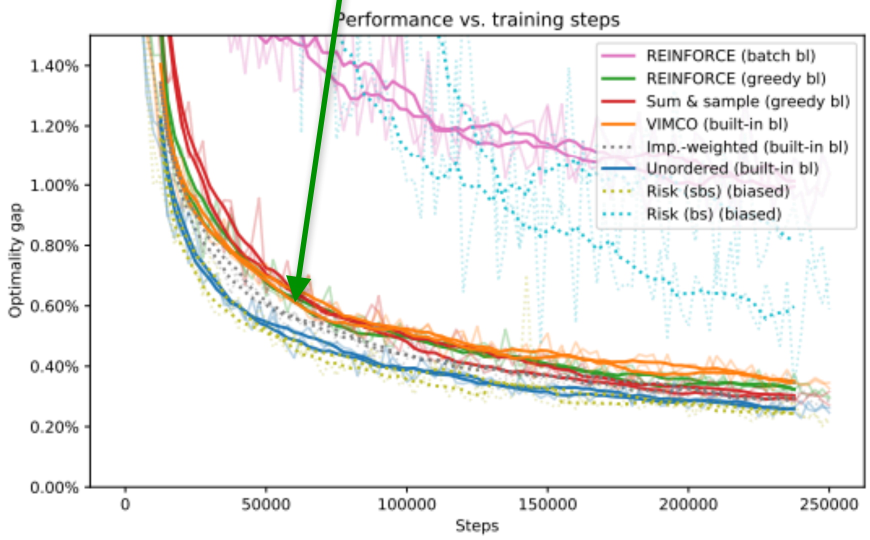


Kool, van Hoof & Welling, ICLR 2020

Stochastic beam search for faster learning

Traveling salesman problem using low-variance estimator

single sample [ICLR 2019]



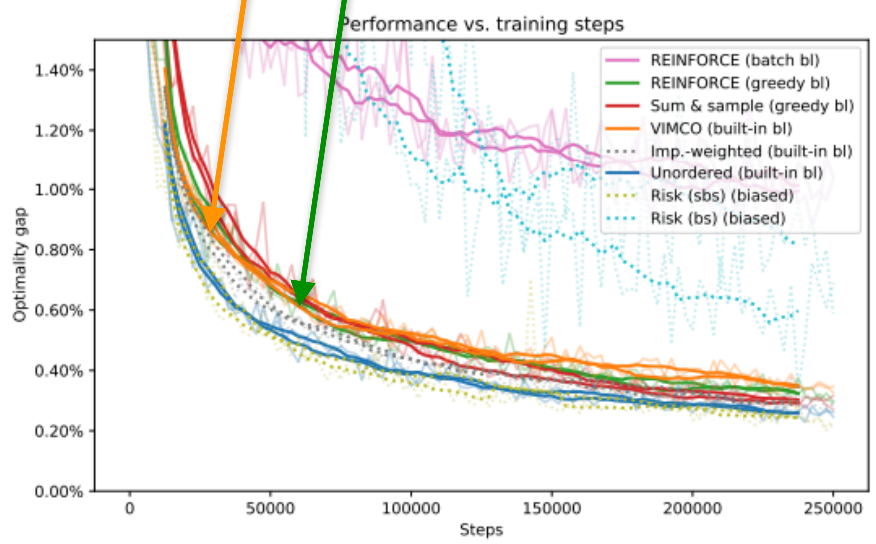
Kool, van Hoof & Welling, ICLR 2020

Stochastic beam search for faster learning

Traveling salesman problem using low-variance estimator

with replacement [ICLR 2019 WS; Mnih & Rezende 2016]

single sample [ICLR 2019]



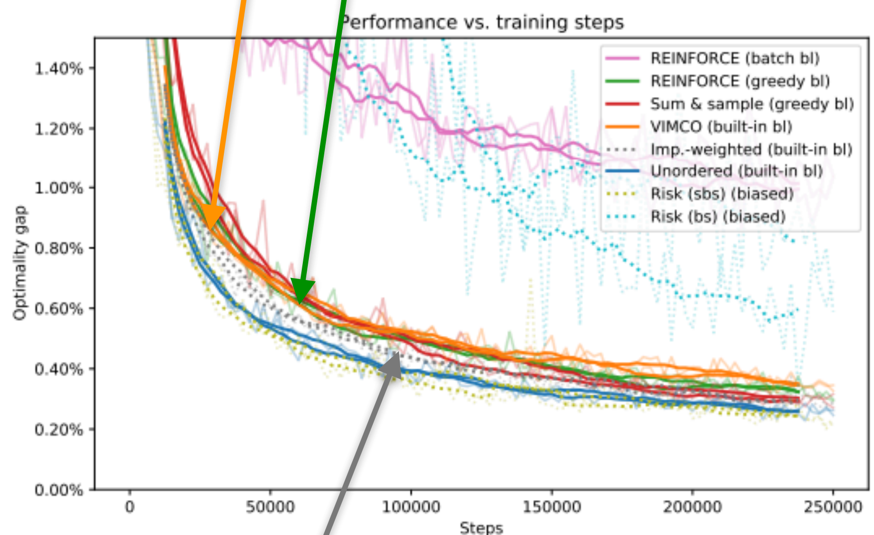
Kool, van Hoof & Welling, ICLR 2020

Stochastic beam search for faster learning

Traveling salesman problem using low-variance estimator

with replacement [ICLR 2019 WS; Mnih & Rezende 2016]

single sample [ICLR 2019]



without replacement, normalised IW (biased) [ICLR 2019 WS]

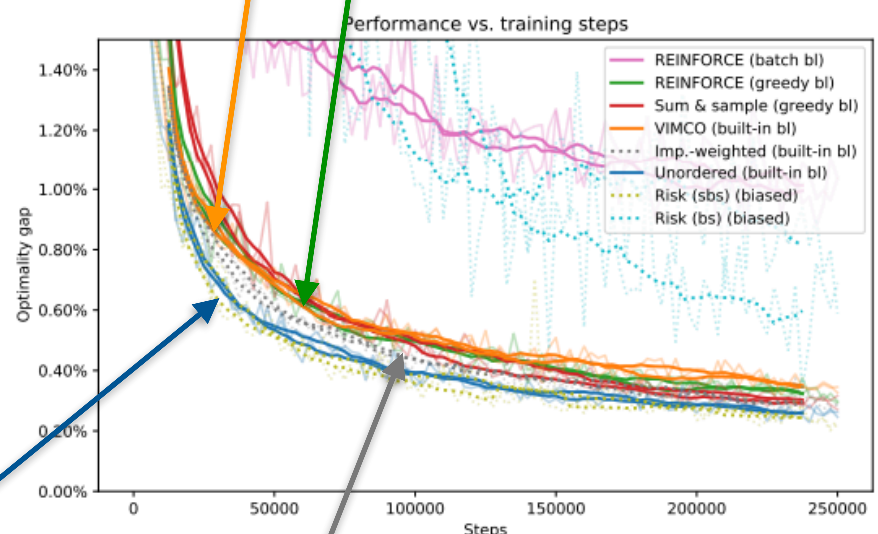
Kool, van Hoof & Welling, ICLR 2020

Stochastic beam search for faster learning

Traveling salesman problem using low-variance estimator

with replacement [ICLR 2019 WS; Mnih & Rezende 2016]

single sample [ICLR 2019]



without replacement, normalised IW (biased) [ICLR 2019 WS]

without replacement, unordered set estimator [ICLR 2020]

Kool, van Hoof & Welling, ICLR 2020

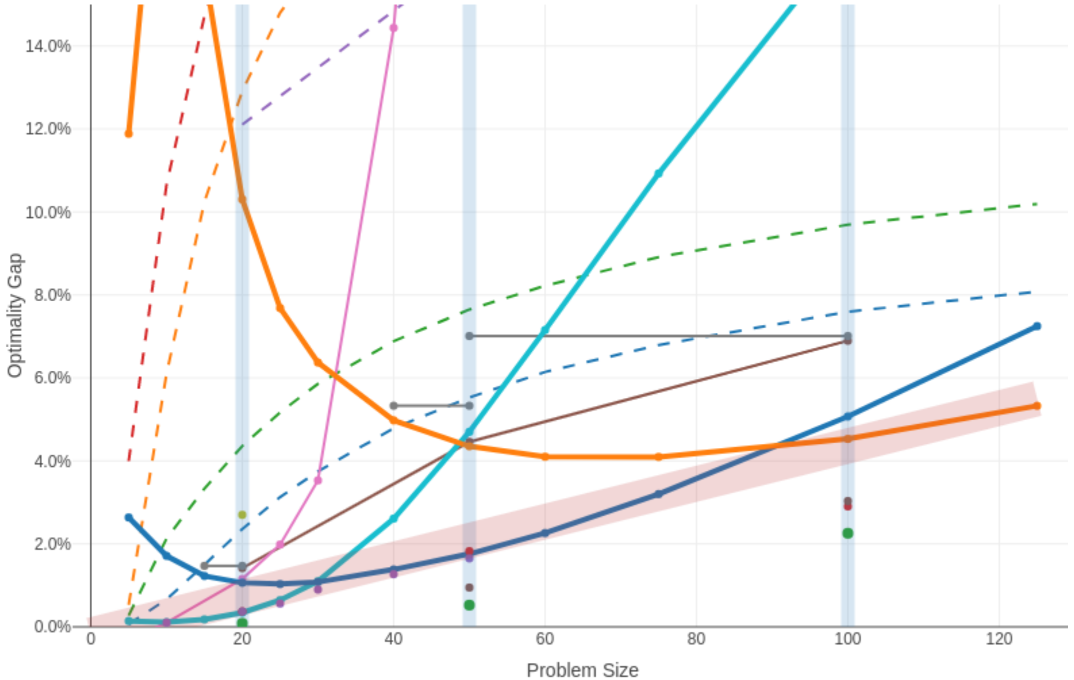
Scalability a main limiting factor!

Generalization to problems of different size

- Network trained on small instances performs so-so on large ones

Sample efficiency

- Machine learning and RL need lots of data (600k tours for TSP)



Computational efficiency

Earlier approach requires network pass for each city

Means overall quadratic scaling

[Kool, van Hoof, Gromicho, Welling; Working paper - arXiv:2102.11756]

Computational efficiency

Earlier approach requires network pass for each city

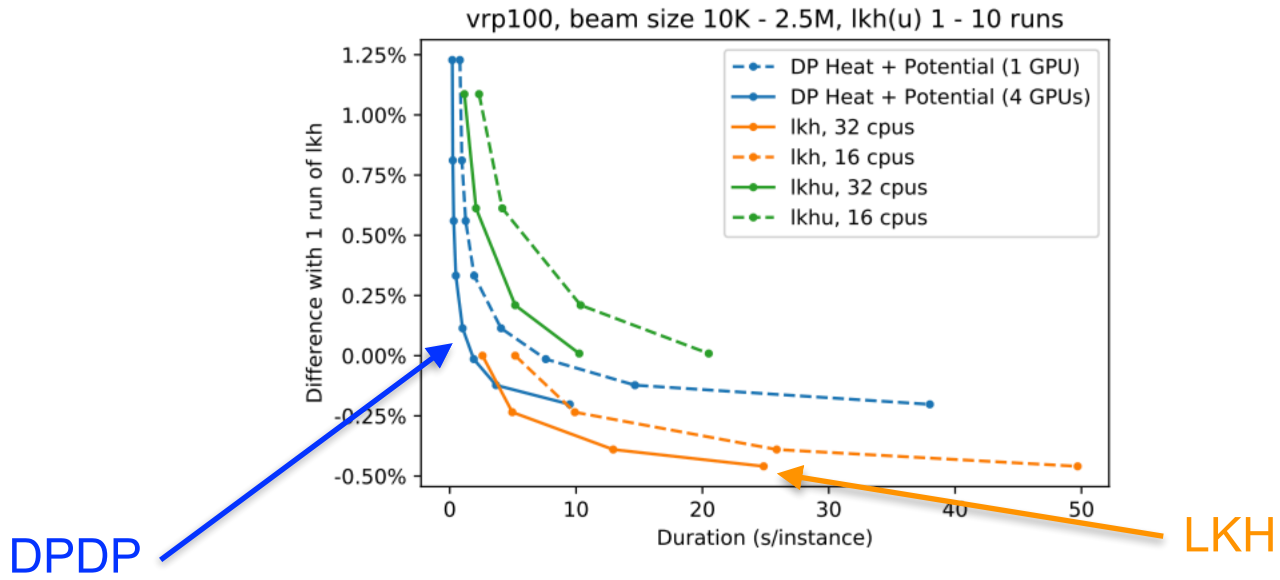
Means overall quadratic scaling

Alternative idea:
DPDP: Deep Policy Dynamic Programming

Single neural network pass to identify good ‘edges’.
Then conventional dynamic programming to find tour

[Kool, van Hoof, Gromicho, Welling; Working paper - arXiv:2102.11756]

Computational efficiency



Competitive to highly optimised LKH heuristic

[Kool, van Hoof, Gromicho, Welling; Working paper - arXiv:2102.11756]

Dynamic problems

Dynamic problems, e.g. routing with dynamically changing costs, constraints, objective, of practical interest

ML & RL suitable to handle the inherent uncertainty

Current public-private (NWO/NS) project about train shunting
(With Matthew Macfarlane, Diederik Roijers (HU), Wan-Jui Lee (NS))

- Finding plans that are robust to minor changes in specification
- Learning local repair heuristics
- Dealing with trade-off between performance and robustness

Conclusions

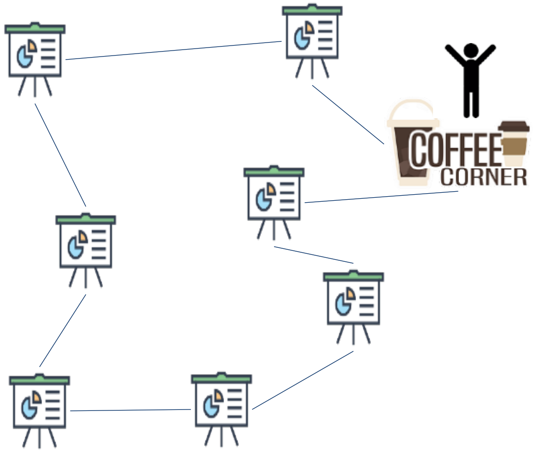
Combinatorial optimisation problems

- Allow end-to-end deep learning as alternative to manually defined heuristics
- Have special structure, can be exploited to learn solutions
- Scaling to large instances is still challenging
- Dynamic problems: interesting challenges & opportunities

Conclusions

Combinatorial optimisation problems

- Allow end-to-end deep learning as alternative to manually defined heuristics
- Have special structure, can be exploited to learn solutions
- Scaling to large instances is still challenging
- Dynamic problems: interesting challenges & opportunities



Thanks

Wouter Kool

Max Welling

Thanks

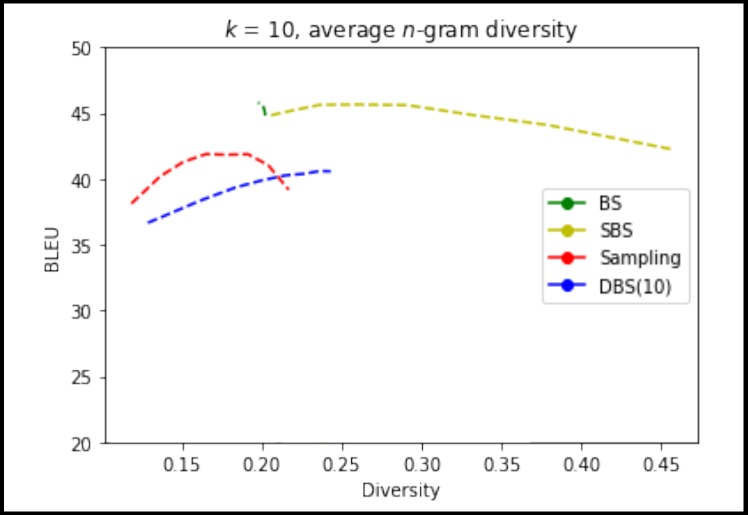
Wouter Kool

Max Welling

Questions?

Experiment: Translation Diversity

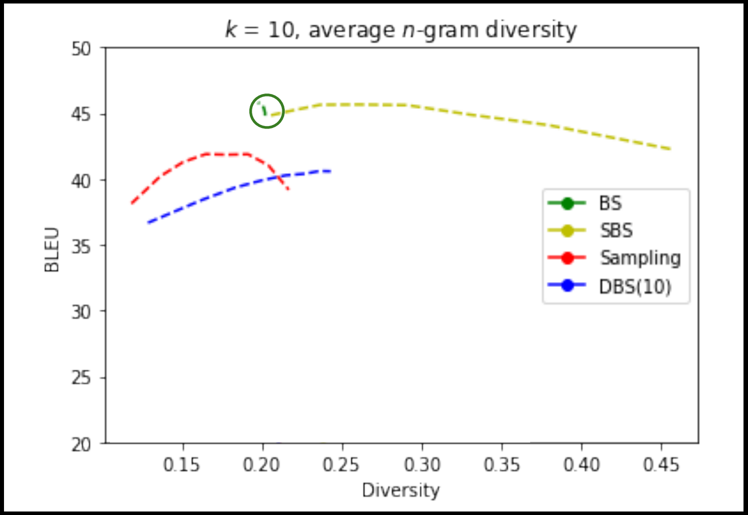
- Generate k translations
- Plot BLEU against diversity
- Vary softmax temperature
- Compare:
  - Beam Search
  - Stochastic Beam Search
  - Sampling
  - Diverse Beam Search (Vijayakumar et al., 2018)



Kool, van Hoof & Welling, ICML 2019

Experiment: Translation Diversity

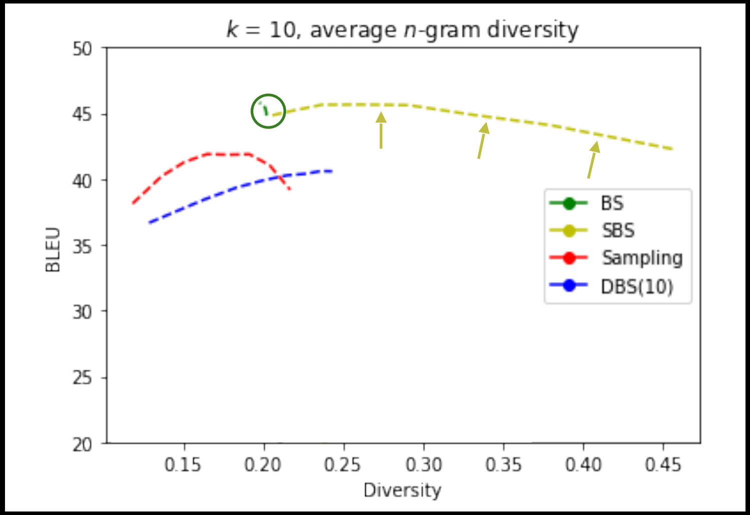
- Generate k translations
- Plot BLEU against diversity
- Vary softmax temperature
- Compare:
  - Beam Search
  - Stochastic Beam Search
  - Sampling
  - Diverse Beam Search (Vijayakumar et al., 2018)



Kool, van Hoof & Welling, ICML 2019

Experiment: Translation Diversity

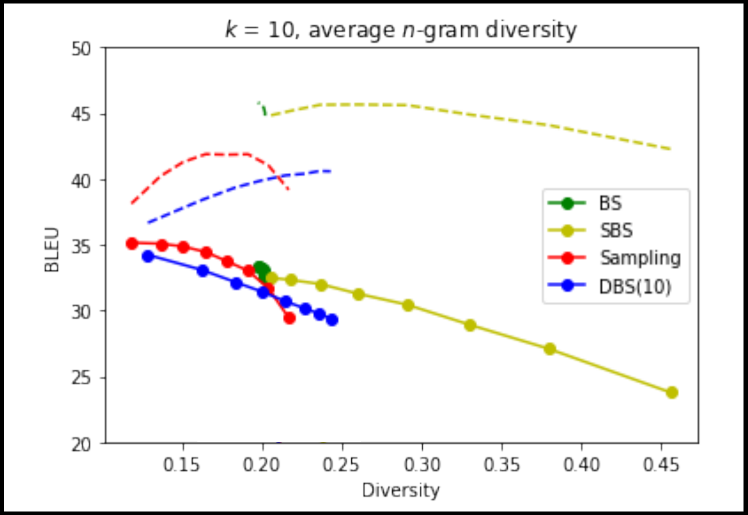
- Generate k translations
- Plot BLEU against diversity
- Vary softmax temperature
- Compare:
    - Beam Search
    - Stochastic Beam Search
    - Sampling
    - Diverse Beam Search (Vijayakumar et al., 2018)



Kool, van Hoof & Welling, ICML 2019

Experiment: Translation Diversity

- Generate k translations
- Plot BLEU against diversity
- Vary softmax temperature
- Compare:
    - Beam Search
    - Stochastic Beam Search
    - Sampling
    - Diverse Beam Search (Vijayakumar et al., 2018)



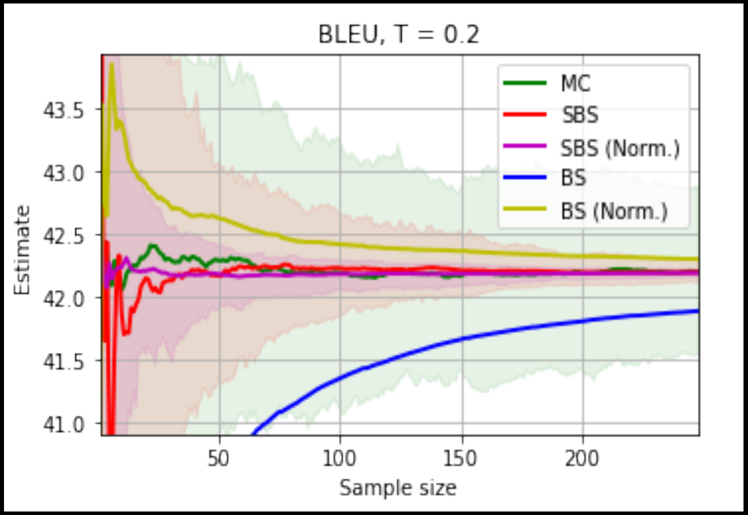
Kool, van Hoof & Welling, ICML 2019

Experiment: BLEU score estimation

- Estimate expected sentence-level BLEU

- Plot mean and 95% interval vs. num samples

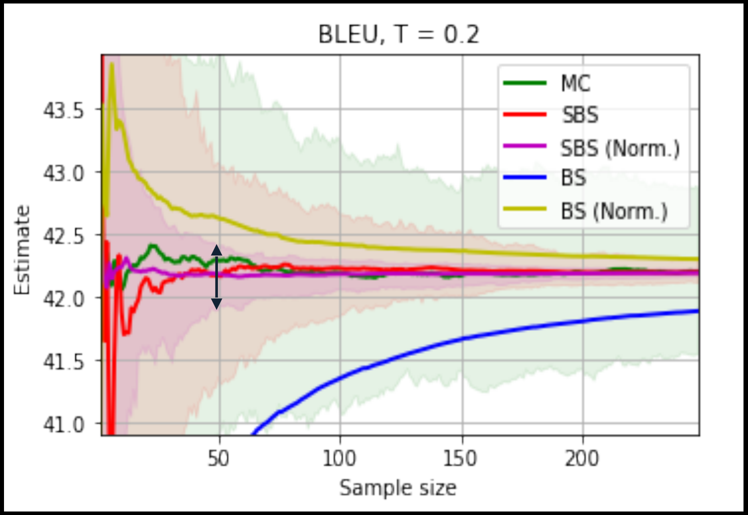
- Compare:
  - Monte Carlo Sampling
  - Stochastic Beam Search with (normalized) Importance Weighted estimator
  - Beam Search with deterministic estimate



Kool, van Hoof & Welling, ICML 2019

Experiment: BLEU score estimation

- Estimate expected sentence-level BLEU
- Plot mean and 95% interval vs. num samples
- Compare:
  - Monte Carlo Sampling
  - Stochastic Beam Search with (normalized) Importance Weighted estimator
  - Beam Search with deterministic estimate



Kool, van Hoof & Welling, ICML 2019