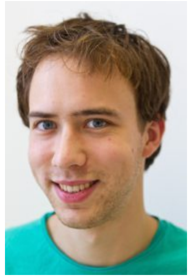
Deep Policy Dynamic Programming for Vehicle Routing Problems

Wouter Kool

University of Amsterdam

ORTEC

AMLAB Amsterdam Machine Learning Lab

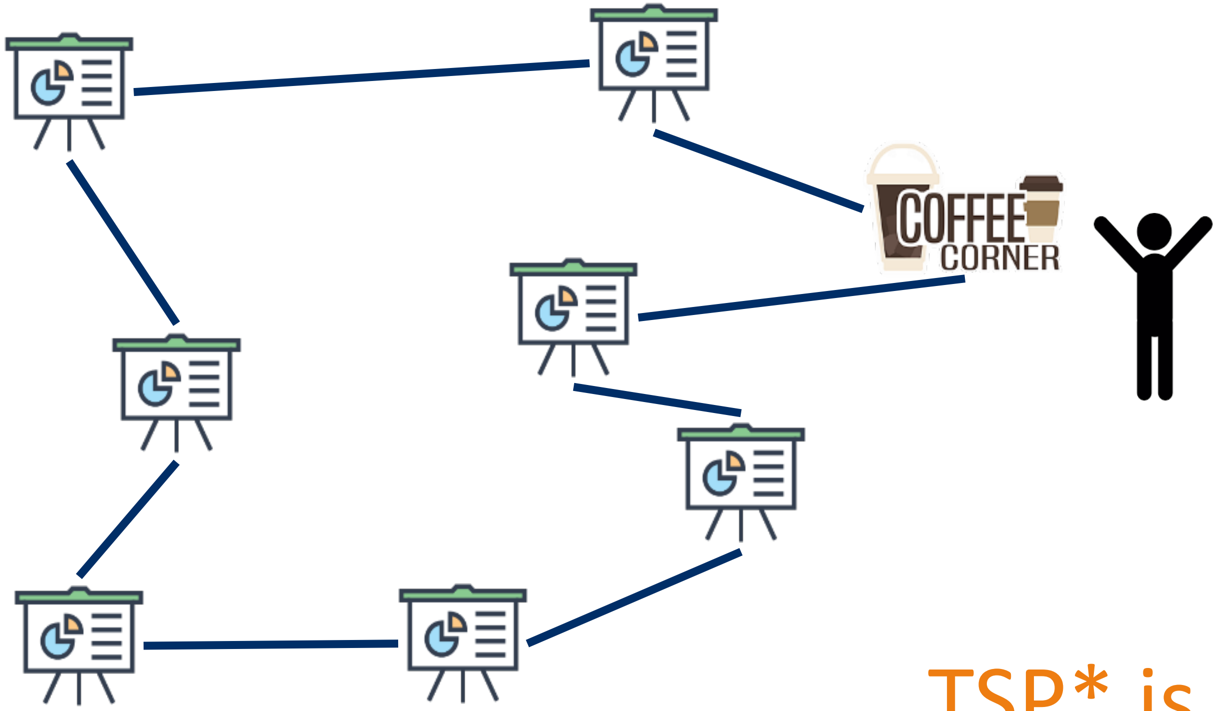


Herke van Hoof

Joaquim Gromicho

Max Welling

Travelling Scientist Problem (TSP)



TSP* is (NP-)hard!

Kool et al., 2019

What does it mean?

Finding *optimal* solutions for *all* problem instances

MISSION: IMPOSSIBLE

* unless P = NP

Finding *acceptable* solutions for *relevant* problem instances

MISSION: ~~IM~~POSSIBLE

DEEP LEARNING

We use HEURISTICS

Can be seen as ‘rules of thumb’

‘next location should be nearby’

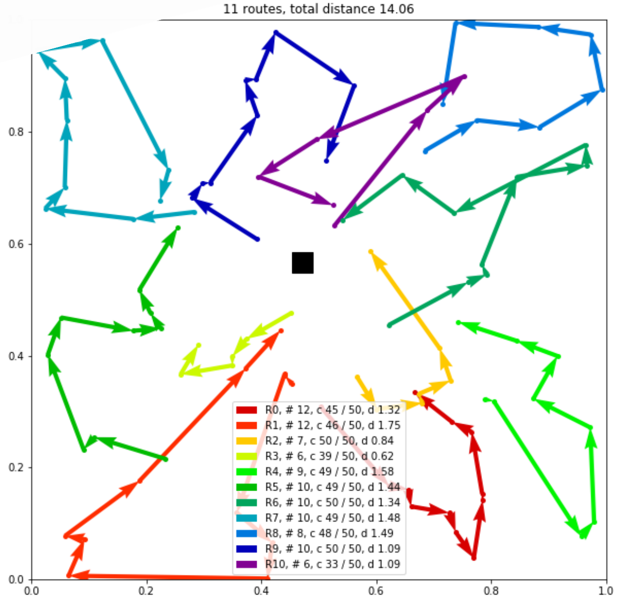
First try (few years back)

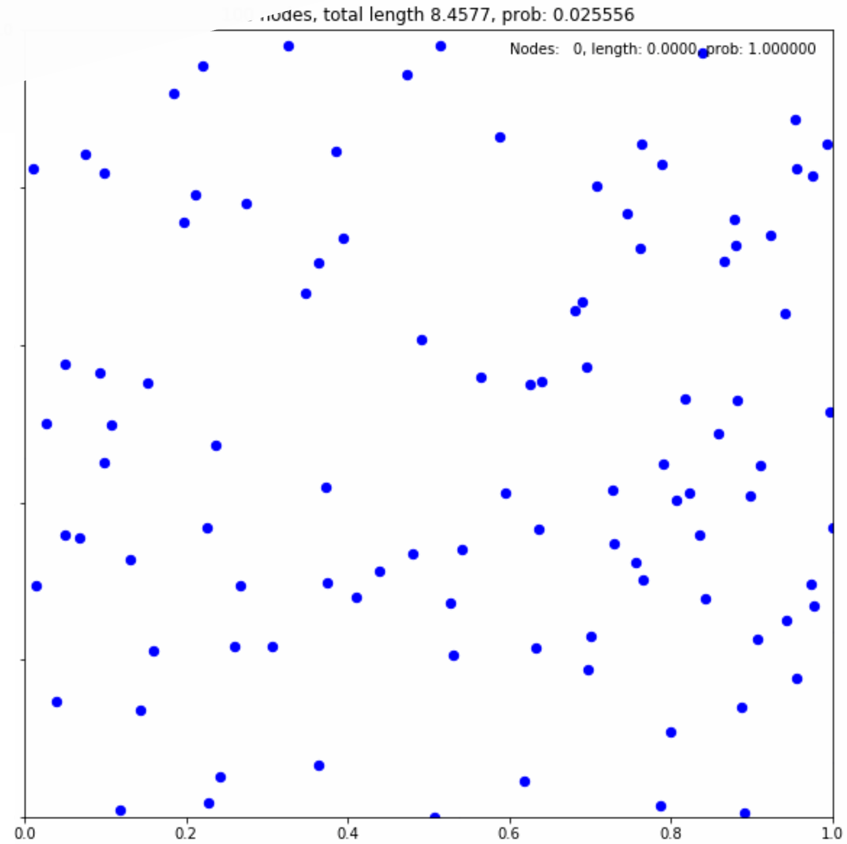

ATTENTION, LEARN TO SOLVE ROUTING PROBLEMS!

Wouter Kool
University of Amsterdam
ORTEC
w.w.m.kool@uva.nl

Herke van Hoof
University of Amsterdam
h.c.vanhoof@uva.nl

Max Welling
University of Amsterdam
CIFAR
m.welling@uva.nl

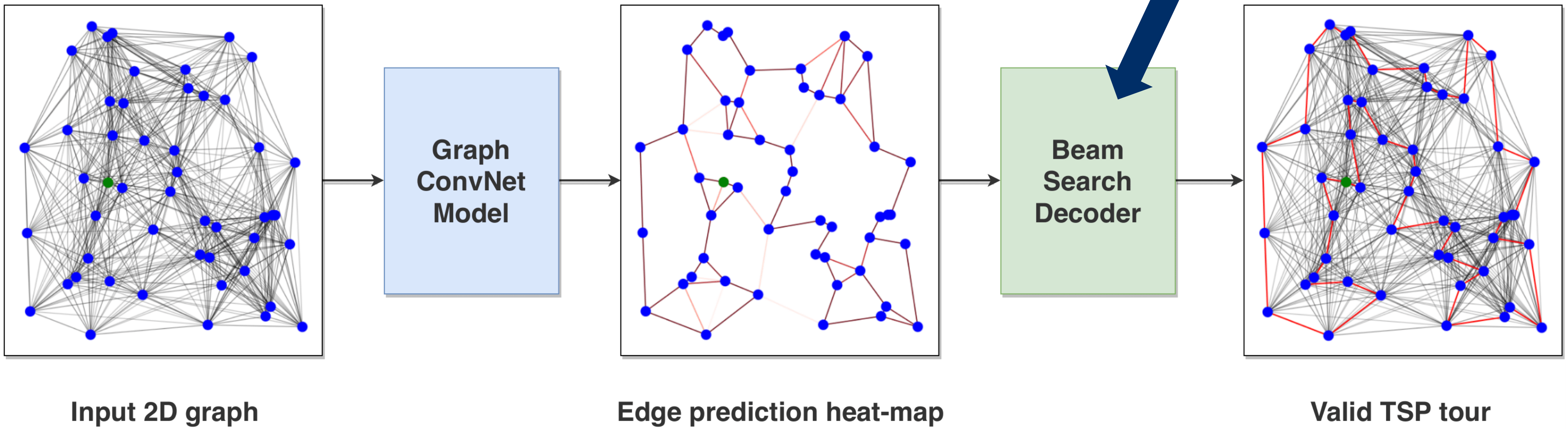




ORTEC

Non-autoregressive approach (Joshi et al., 2019)

We can do better!

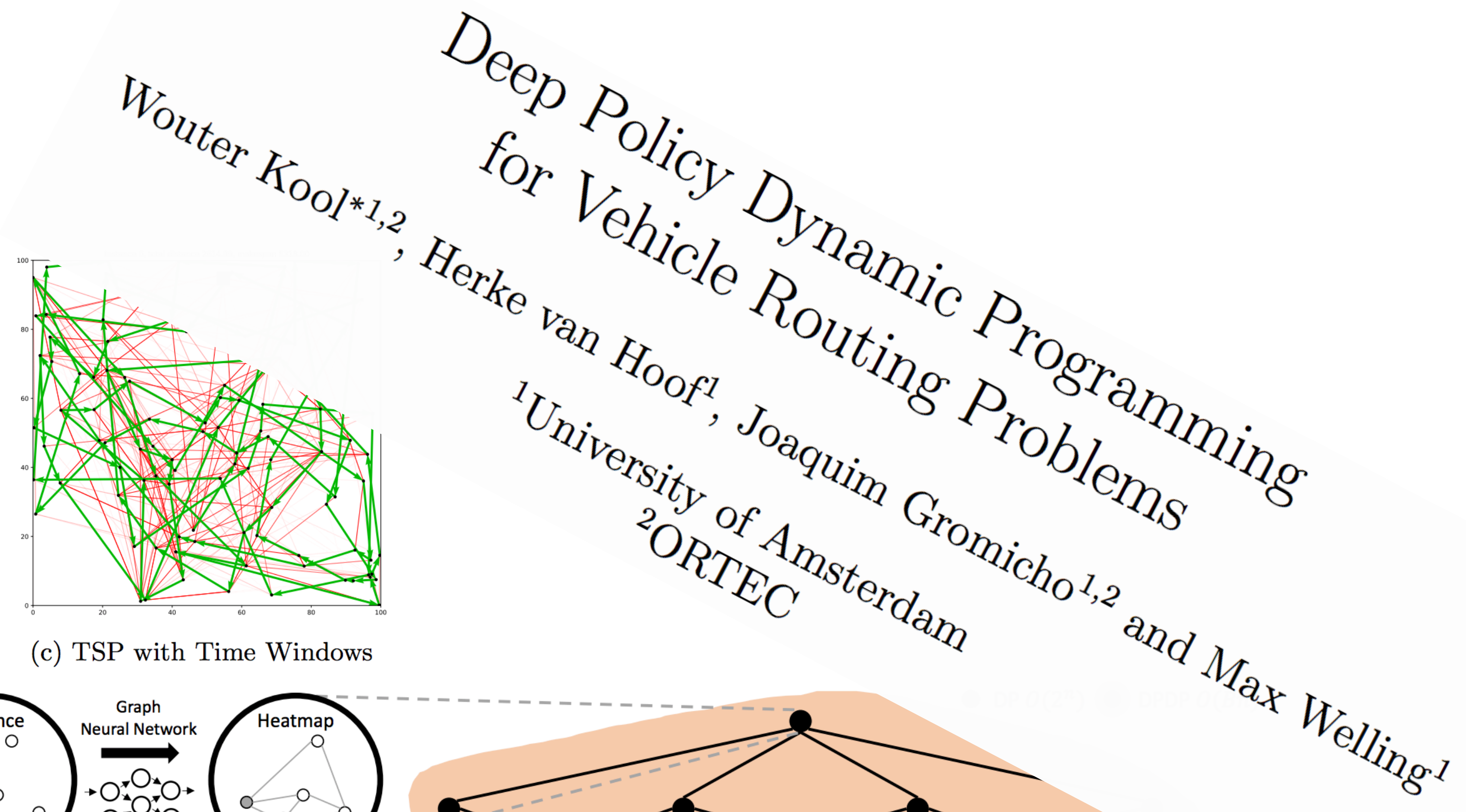


Picture by Joshi et al., 2019

More efficient AND better results!

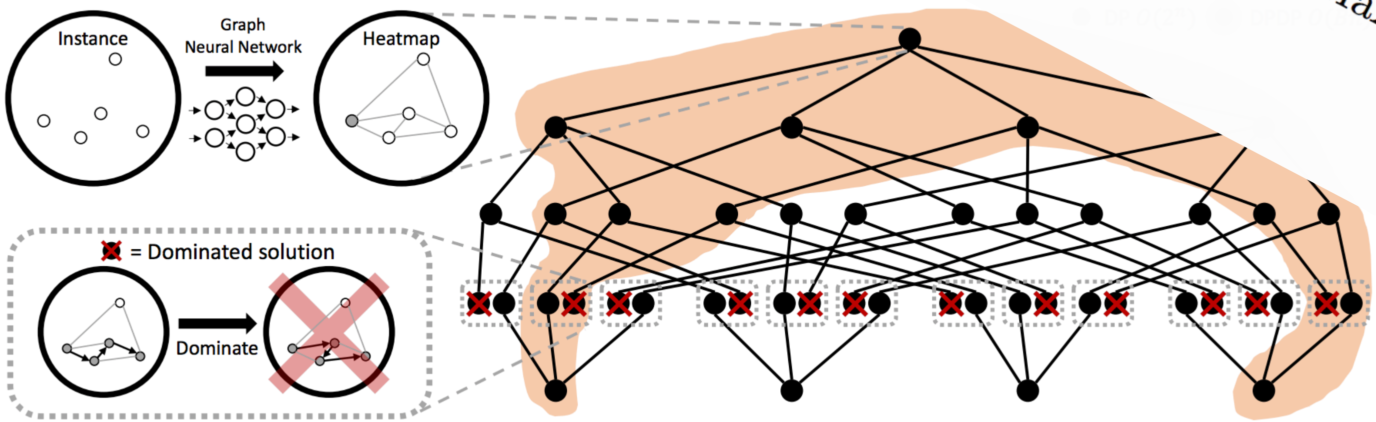
Bringing us to DPDP

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


Deep Policy Dynamic Programming for Vehicle Routing Problems

Wouter Kool*[1,2], Herke van Hoof[1], Joaquim Gromicho[1,2] and Max Welling[1]

[1]University of Amsterdam
[2]ORTEC



ORTEC

TL;DR

"

DPDP is a flexible framework for vehicle routing problems…

…that restricts a dynamic program to promising parts of the state space…

…using a heatmap of promising edges predicted by a neural network!

"

Small side note…

If you like ‘neural vehicle routing’

Chaitanya K. Joshi

Home Publications **Blog** Talks Contact CV

Recent Advances in Deep Learning for Routing Problems

Developing neural network-driven solvers for combinatorial optimization problems such as the Travelling Salesperson Problem have seen a surge of academic interest recently. This blogpost presents a **Neural Combinatorial Optimization** pipeline that unifies several recently proposed model architectures and learning paradigms into one single framework. Through the lens of the pipeline, we analyze recent advances in deep learning for routing problems and provide new directions to stimulate future research towards practical impact.

Chaitanya K. Joshi, Rishabh Anand
Jan 12, 2022 · 20 min read

https://www.chaitjo.com/post/deep-learning-for-routing-problems/

Starting July 1st…

Goal: bring together

Operations Research and *Machine Learning*

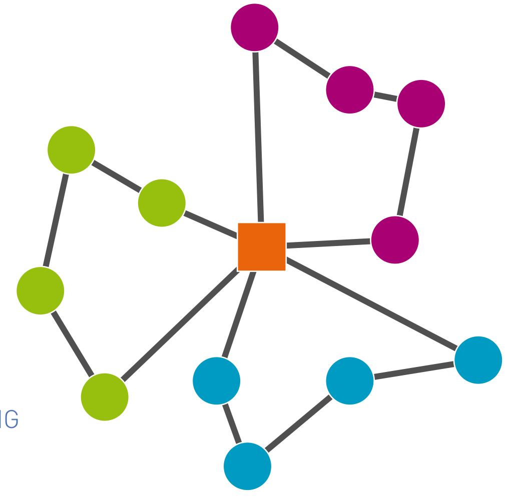
to solve a *static* and *dynamic* VRP with time windows!

EURO Meets NeurIPS 2022 Vehicle Routing Competition

EURO
THE ASSOCIATION OF EUROPEAN OPERATIONAL RESEARCH SOCIETIES

EWG DSO
DATA SCIENCE MEETS OPTIMIZATION

EWG VEROLOG
VEHICLE ROUTING AND LOGISTICS



NEURAL INFORMATION PROCESSING SYSTEMS

ORTEC

EAISI EINDHOVEN AI SYSTEMS INSTITUTE

TU/e

More info? https://euro-neurips-vrp-2022.challenges.ortec.com/

The technical part

Held-Karp DP for TSP (Held & Karp, 1962; Bellman, 1962)

Cost/distance from j to i

$$C(S,i) = \min_{j\in S\setminus\{i\}} C(S\setminus\{i\}, j) + c_{ji}$$

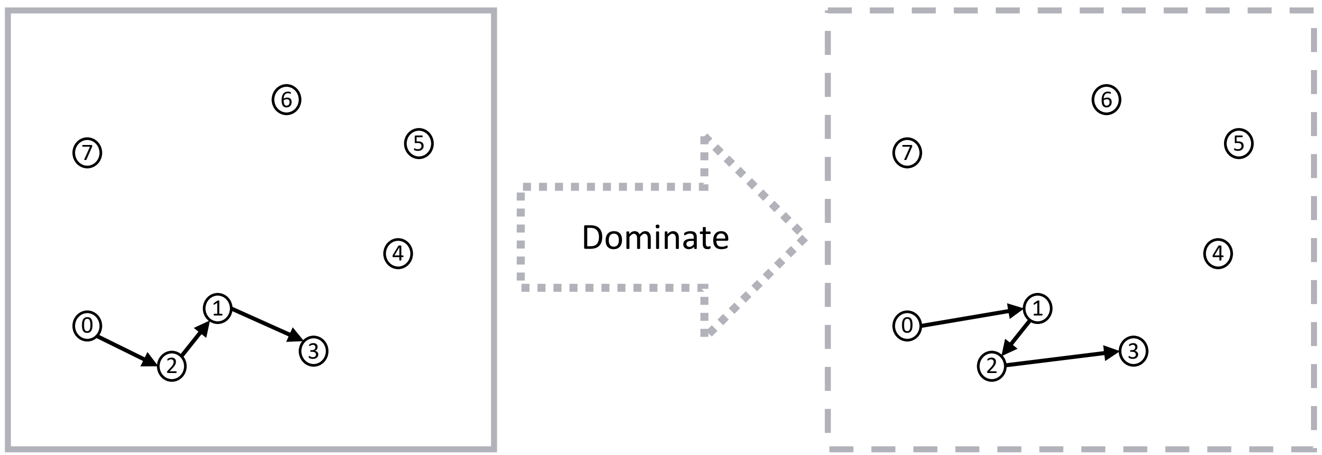
Minimum cost to go from 0 to i visiting all nodes in S

Set of visited nodes

Current node

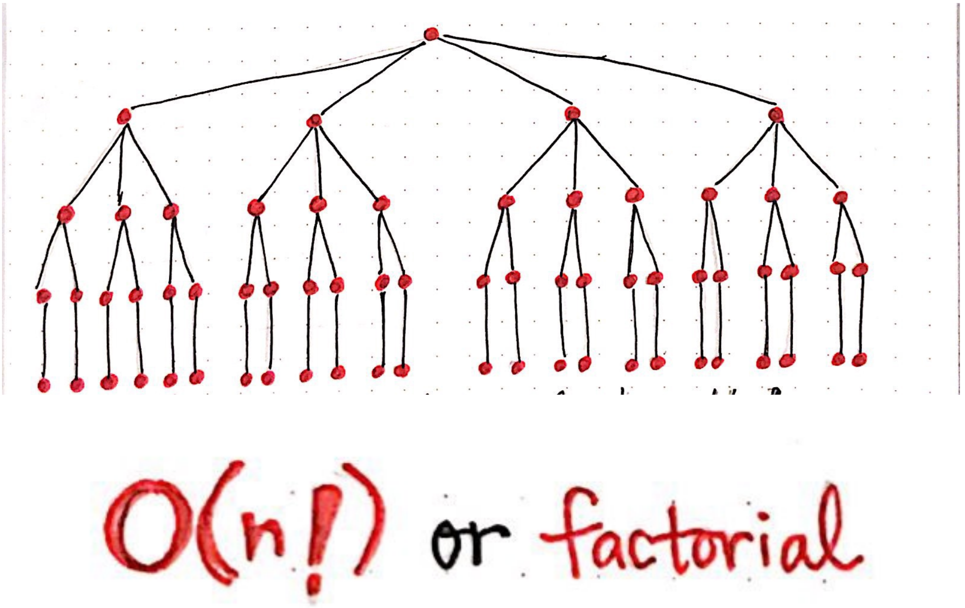
DP state

Find best solution for each DP state

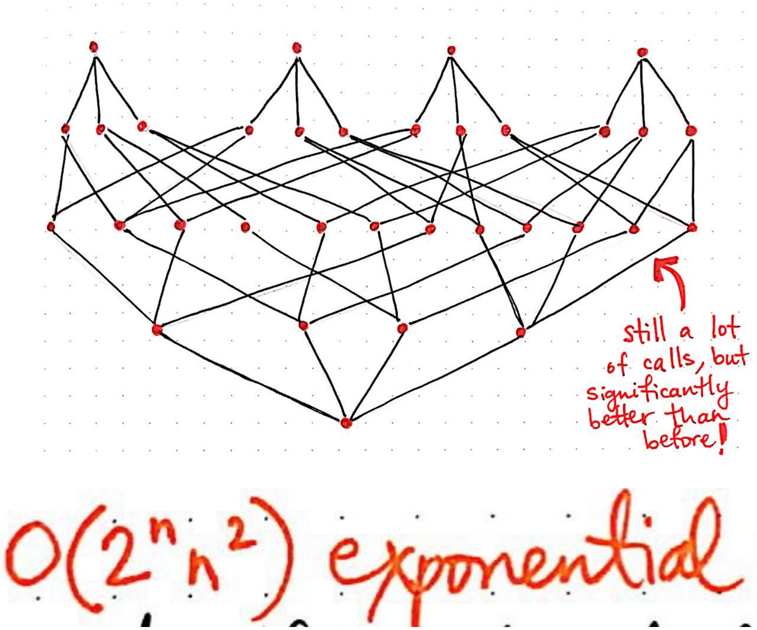


Held-Karp DP for TSP (Held & Karp, 1962; Bellman, 1962)

Brute-force (forward view)



DP (top-down or backward view)

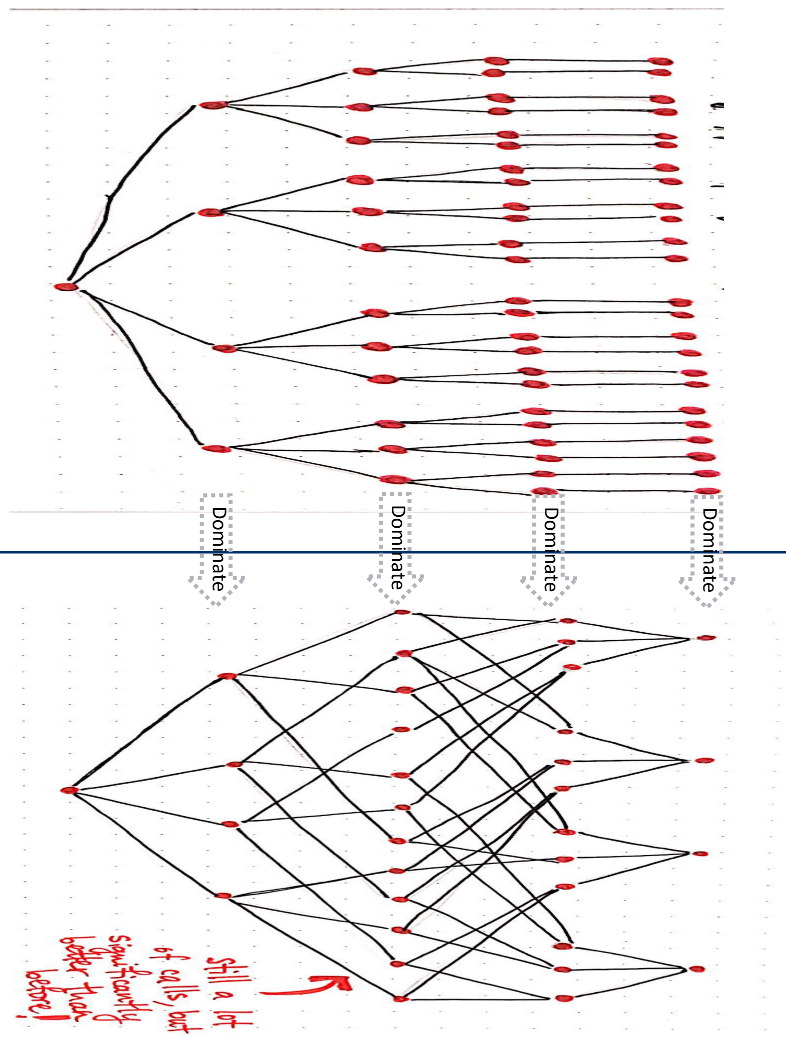


Artwork by Vaidehi Joshi, https://medium.com/basecs/speeding-up-the-traveling-salesman-using-dynamic-programming-b76d7552e8dd

Held-Karp DP for TSP (Held & Karp, 1962; Bellman, 1962)

Brute-force

$O(n!)$ or factorial



Forward view

Still impractical!

DP

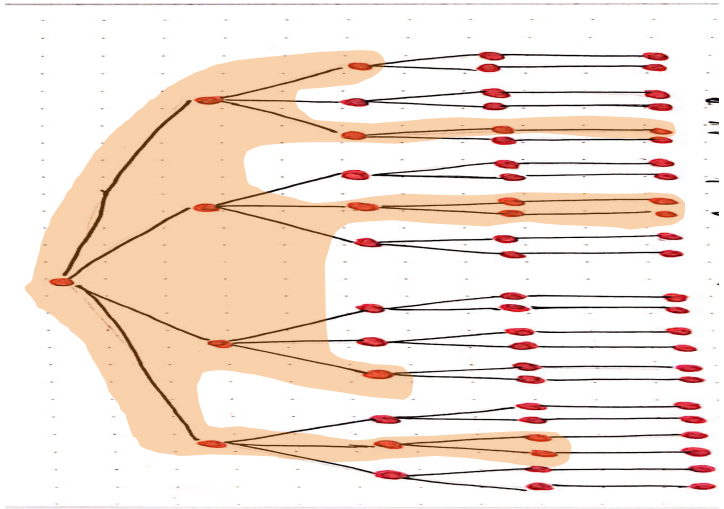
$O(2^n n^2)$ exponential

UNIVERSITY OF AMSTERDAM ORTEC

Held-Karp DP for TSP (Held & Karp, 1962; Bellman, 1962)

Beam search

$O(Bn)$ or *linear*

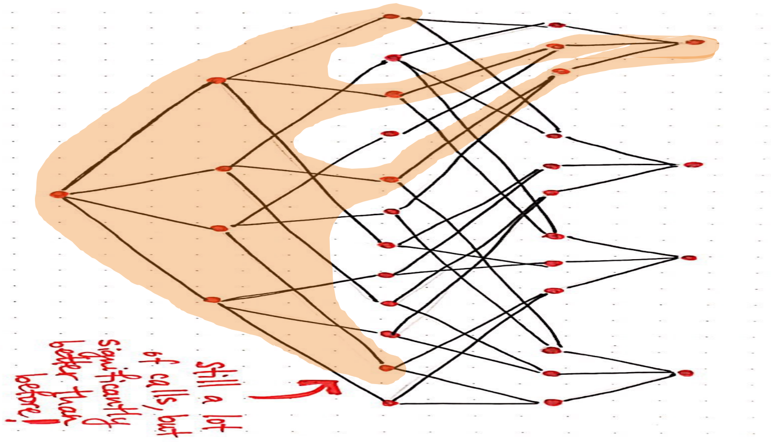


We need a *good policy* to restrict the search space!

Forward view

Restricted DP

$O(Bn)$ or *linear*



Malandraki & Dial, 1996
Gromicho et al., 2012

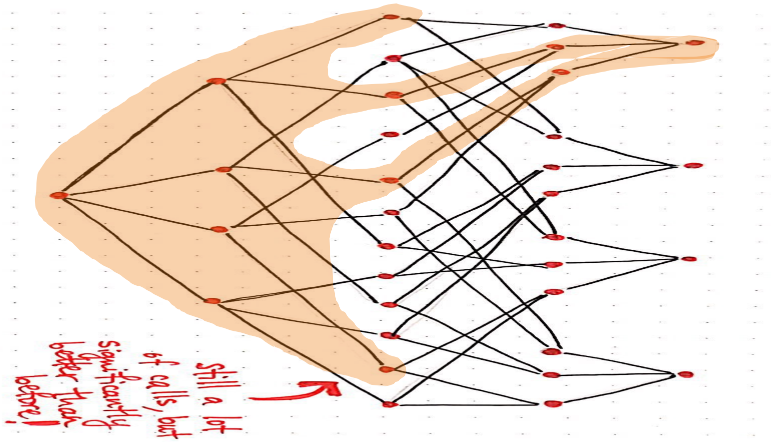
UNIVERSITY OF AMSTERDAM ORTEC

Deep Policy Dynamic Programming (DPDP)

https://arxiv.org/abs/2102.11756

DPDP

$O(Bn)$ or linear



For each iteration
- Expand solutions
- Remove dominated solutions
- Select top B according to *policy*
- Repeat

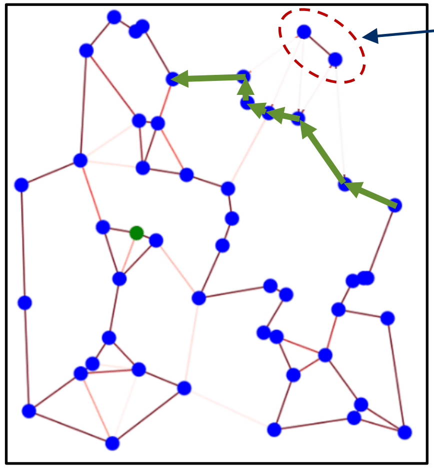
Forward view

Policy: select top B solutions that maximize the score.

$$\text{SCORE} = \text{HEAT} + \text{POTENTIAL}$$

Heat of edges in solution

Estimate for unvisited nodes
based on remaining edges



Picture by Joshi et al., 2019

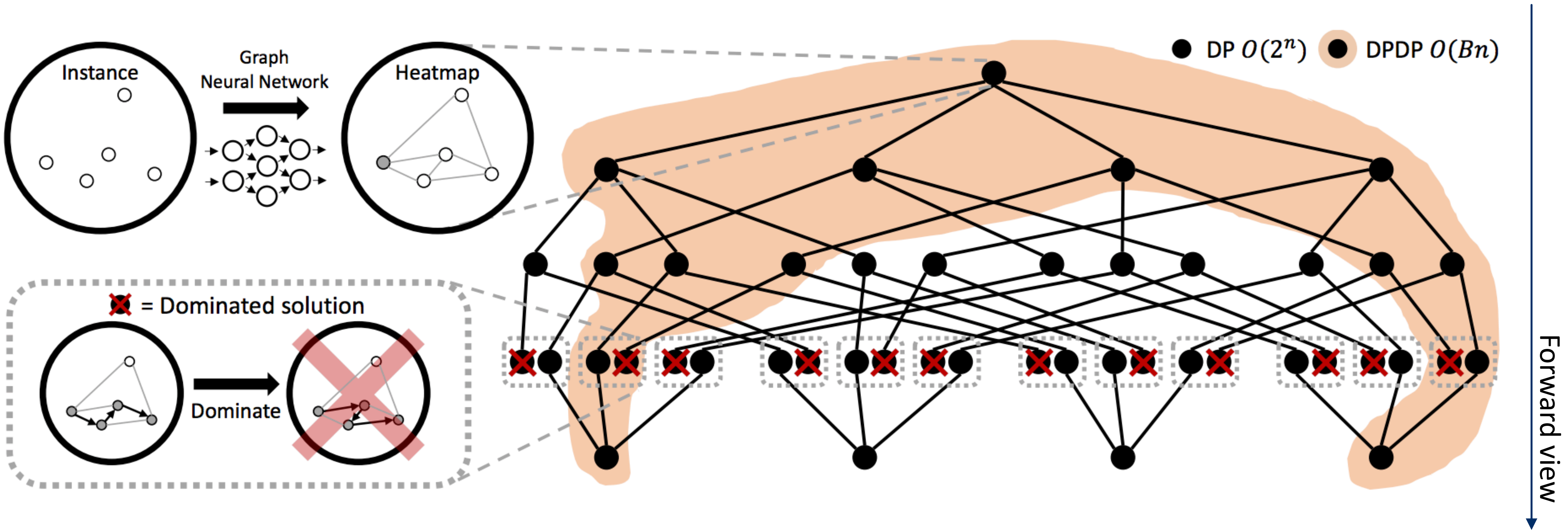
Potential avoids 'skipped nodes'

Heat
$h_{ij} \in (0,1)$

UNIVERSITY OF AMSTERDAM ORTEC

Deep Policy Dynamic Programming (DPDP)

https://arxiv.org/abs/2102.11756

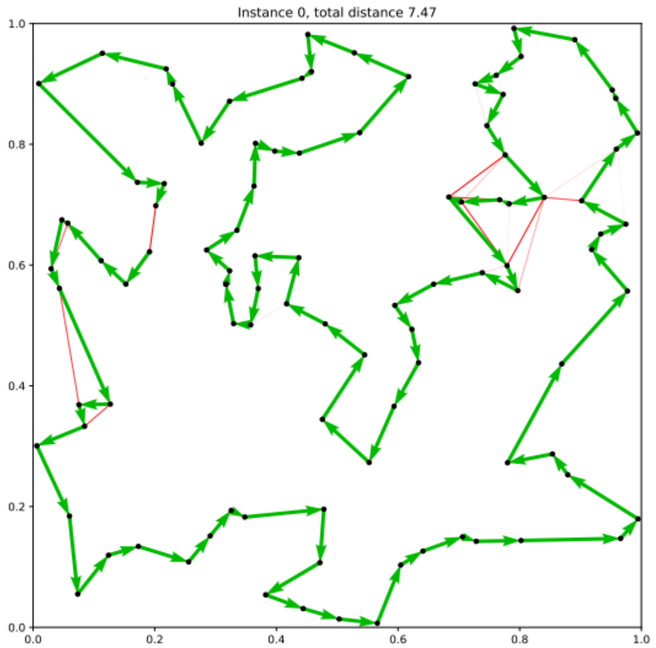


$O(Bn)$ or *linear*

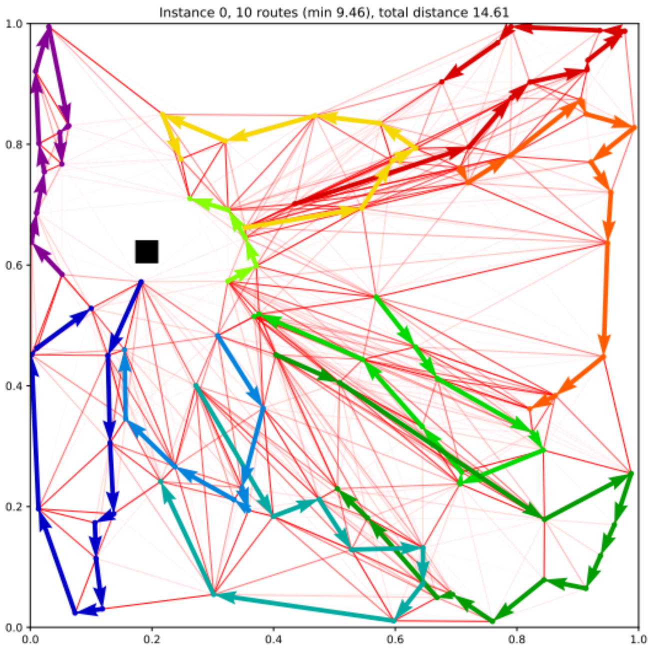
For each iteration

- Expand solutions
- Remove dominated solutions
- Select top B according to *policy*
- Repeat

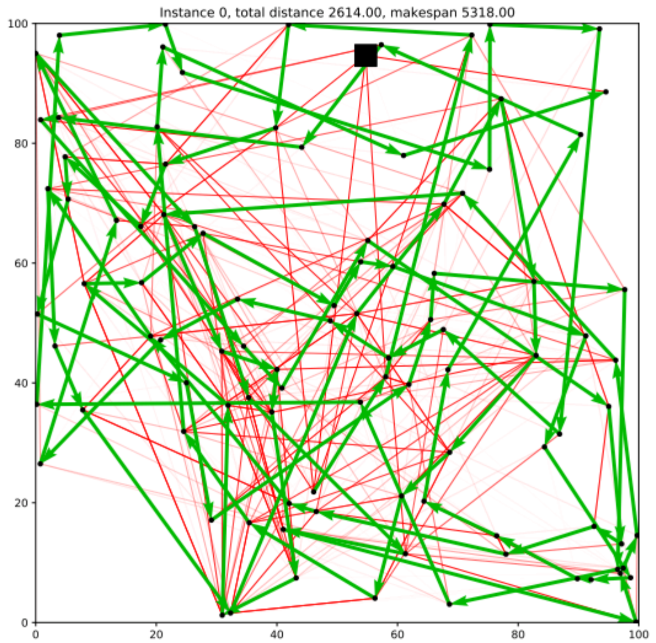
Experiments



(a) Travelling Salesman Problem



(b) Vehicle Routing Problem



(c) TSP with Time Windows

Results (TSP/VRP)

Table 1: Mean cost, gap and *total time* to solve 10000 TSP/VRP test instances.

Problem	TSP100			VRP100		
Method	Cost	Gap	Time	Cost	Gap	Time
Concorde [2]	7.765	0.000 %	6m			
Hybrid Genetic Search [63, 62]				15.563	0.000 %	6h11m
Gurobi [24]	7.776	0.151 %	31m			
LKH [27]	7.765	0.000 %	42m	15.647	0.536 %	12h57m
GNN Heatmap + Beam Search [32]	7.87	1.39 %	40m			
Learning 2-opt heuristics [11]	7.83	0.87 %	41m			
Merged GNN Heatmap + MCTS [19]	7.764*	0.04 %	4m + 11m			
Attention Model + Sampling [36]	7.94	2.26 %	1h	16.23	4.28 %	2h
Step-wise Attention Model [67]	8.01	3.20 %	29s	16.49	5.96 %	39s
Attn. Model + Coll. Policies [34]	7.81	0.54 %	12h	15.98	2.68 %	5h
Learning improv. heuristics [66]	7.87	1.42 %	2h	16.03	3.00 %	5h
Dual-Aspect Coll. Transformer [45]	7.77	0.09 %	5h	15.71	0.94 %	9h
Attention Model + POMO [37]	7.77	0.14 %	1m	15.76	1.26 %	2m
NeuRewriter [9]				16.10	3.45 %	1h
Dynamic Attn. Model + 2-opt [54]				16.27	4.54 %	6h
Neur. Lrg. Neighb. Search [30]				15.99	2.74 %	1h
Learn to improve [43]				15.57*	-	4000h
DPDP 10K	7.765	0.009 %	10m + 16m	15.830	1.713 %	10m + 50m
DPDP 100K	7.765	0.004 %	10m + 2h35m	15.694	0.843 %	10m + 5h48m
DPDP 1M				15.627	0.409 %	10m + 48h27m

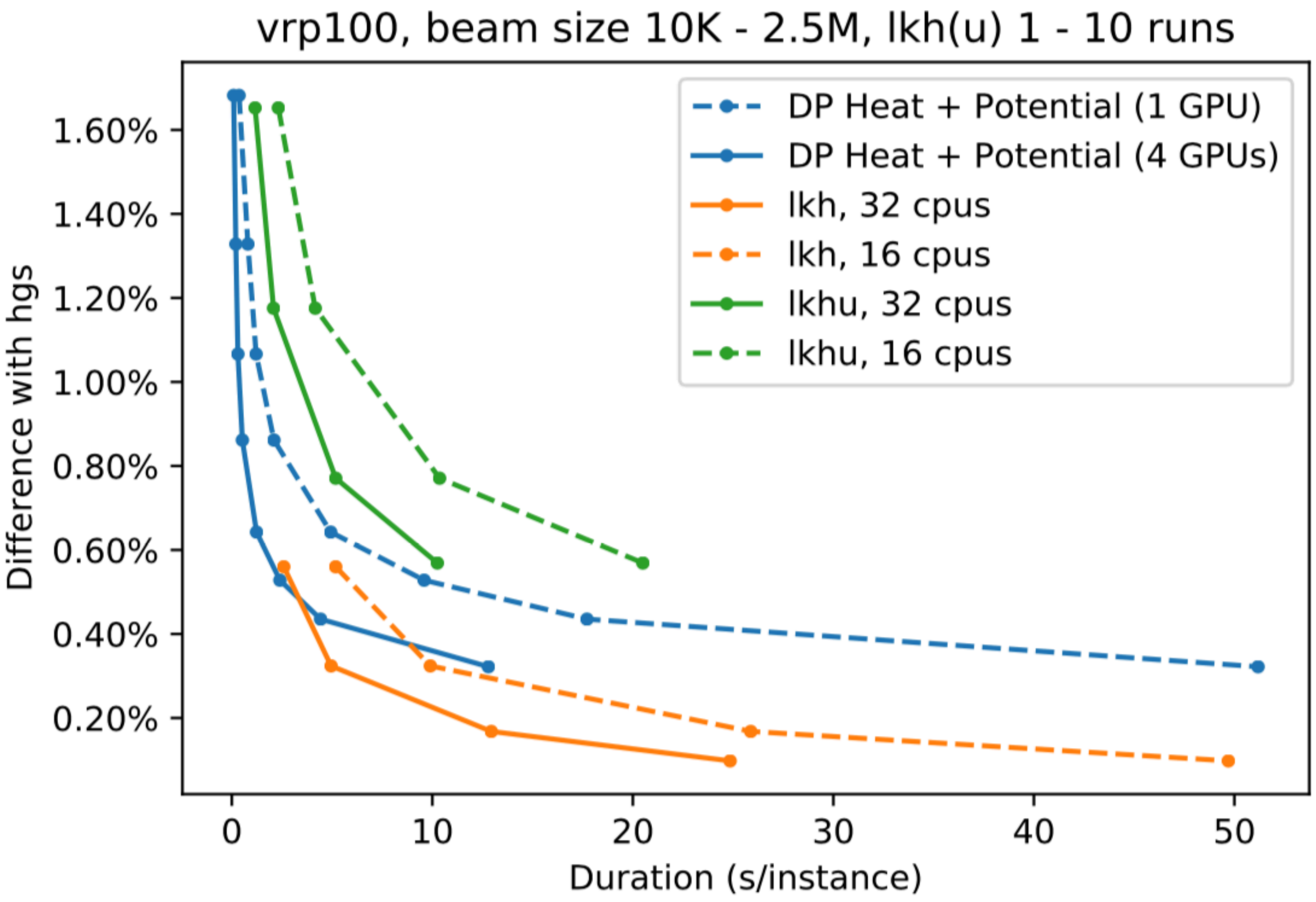
Results (TSP with time windows)

Table 3: Mean cost, gap and *total time* to solve TSPTW100 instances.

Problem Method	Small time windows [8] (100 inst.) Cost	Gap	Fail	Time	Large time windows [12] (10K inst.) Cost	Gap	Fail	Time
GVNS 30x [12]	5129.58	0.000 %		7s	2432.112	0.000 %		37m15s
GVNS 1x [12]	5129.58	0.000 %		<1s	2457.974	1.063 %		1m4s
LKH 1x [27]	5130.32	0.014 %	1.00 %	5m48s	2431.404	-0.029 %		34h58m
BaB-DQN* [8]	5130.51	0.018 %		25h				
ILDS-DQN* [8]	5130.45	0.017 %		25h				
DPDP 10K	5129.58	0.000 %		6s + 1s	2431.143	-0.040 %		10m + 8m7s
DPDP 100K	5129.58	0.000 %		6s + 1s	2430.880	- 0.051 %		10m + 1h16m

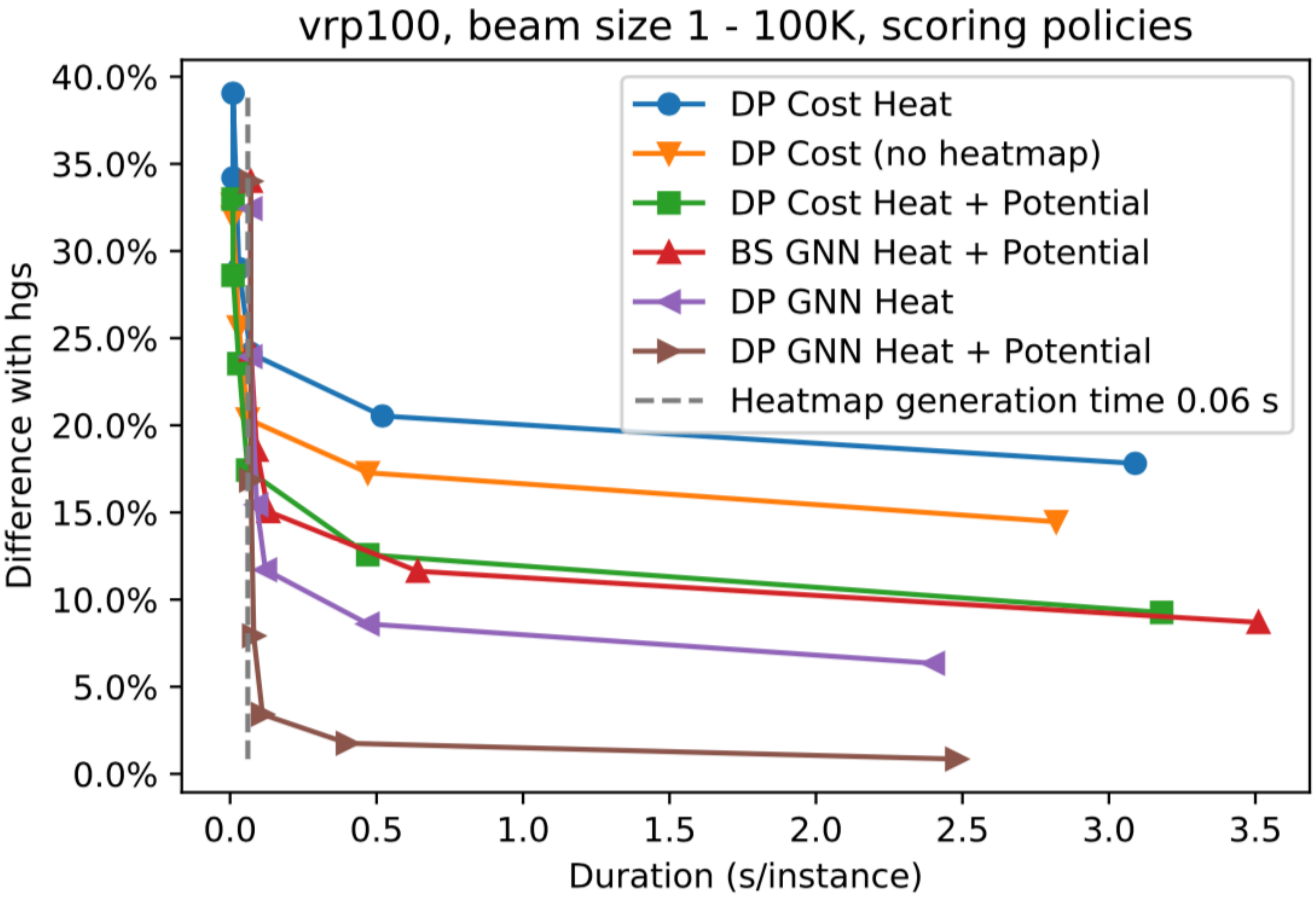
UNIVERSITY OF AMSTERDAM ORTEC

Quality vs. computation



UNIVERSITY OF AMSTERDAM ORTEC

Ablations



UNIVERSITY OF AMSTERDAM ORTEC

Deep Policy Dynamic Programming (DPDP)

https://arxiv.org/abs/2102.11756

- DP is flexible framework for many VRP variants e.g. time windows
- Suitable for GPU implementation
- Natural trade-off compute vs. performance -> asymptotically optimal
- Supervised training based on example solutions
- Test time: only evaluate NN once!


OPTIMIZED BY ORTEC
ORTEC.COM