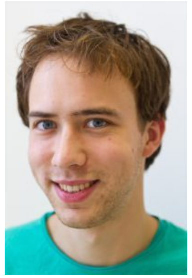
Deep Learning for Combinatorial Optimization

Count your flops and make them count!

Wouter Kool



Herke van Hoof

Joaquim Gromicho

Max Welling

University of Amsterdam

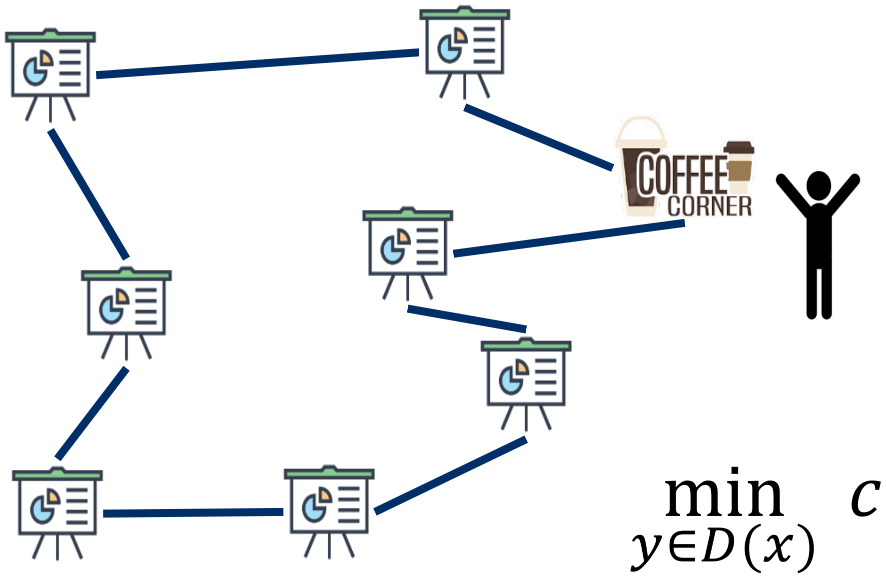
ORTEC

AMLAB
Amsterdam
Machine Learning Lab

What we’re talking about?

$$y = \sigma\left(\sum_i x_i \theta_i\right)$$

for

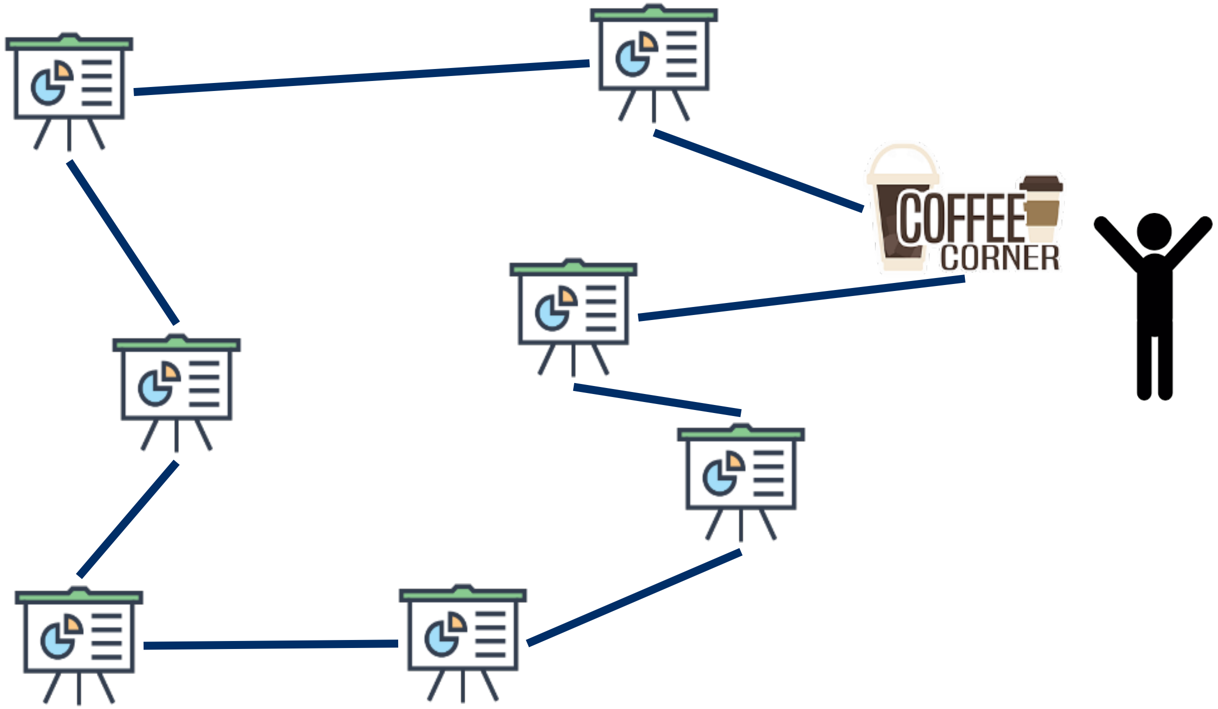


$$\min_{y \in D(x)} c(x, y)$$

Deep Learning

Combinatorial Optimization

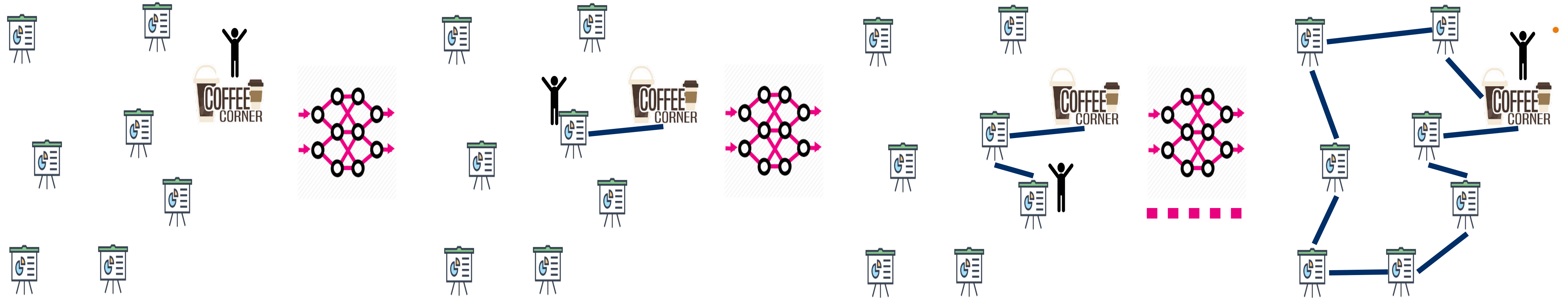
Travelling Scientist Problem (TSP)



Kool et al., 2019

UNIVERSITY OF AMSTERDAM ORTEC

Travelling Scientist Problem (TSP)



Vinyals et al., 2015
Bello et al., 2016
Kool et al., 2019

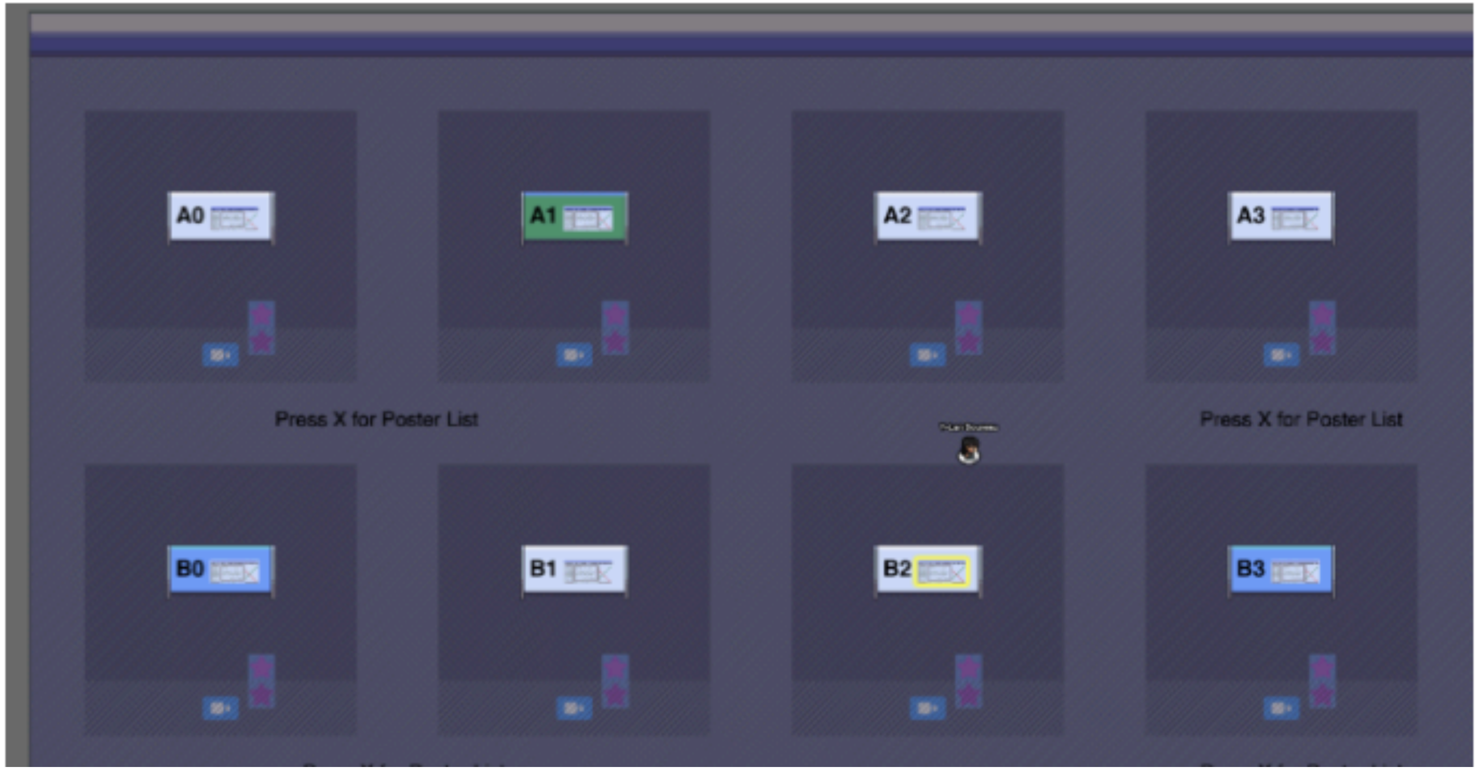
Neural Information Processing Systems Conference

Tweets sent to this account are not actively monitored. To contact us please go to http://neurips.cc/Help/Contact

Follow

41

simply wandering; each poster had a clearly marked presenter spot to easily spot the presenter; people could teleport directly to the poster of their choice from the NeurIPS website, and a coordinate systems allowed people to locate a poster of interest once they were in a room.

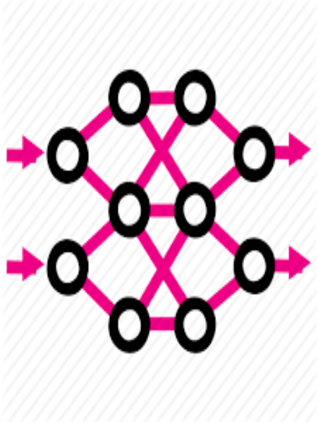


https://neuripsconf.medium.com/neurips-2020-online-experiments-gather-town-poster-sessions-and-mementor-ac1573d61c8a

‘Predicting’ translations

Neural
Machine
Translation

Sentence



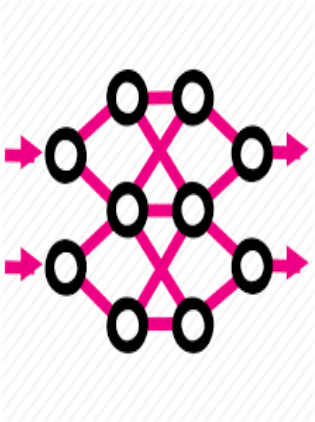
Translation

UNIVERSITY OF AMSTERDAM ORTEC

‘Predicting’ solutions

Neural Combinatorial Optimization

Problem



Solution

UNIVERSITY OF AMSTERDAM ORTEC

It’s not the same!

Machine Translation $\neq$ Combinatorial Optimization

It’s not the same!

Machine Translation $\neq$ Combinatorial Optimization

Learning problem

Scoring translations (learning a model)

Finding a good translation (inference)

No problem

Scoring solution (objective function)

Finding a good solution (optimization)

Computational problem

Stahlberg & Byrne, 2019

It’s not the same!

Machine Translation	$\neq$	Combinatorial Optimization
Maximize quality		Minimize cost
(computation is 2nd)		with minimum computation

UNIVERSITY OF AMSTERDAM ORTEC

It’s not the same!

Machine Translation ≠ Combinatorial Optimization

Maximize quality

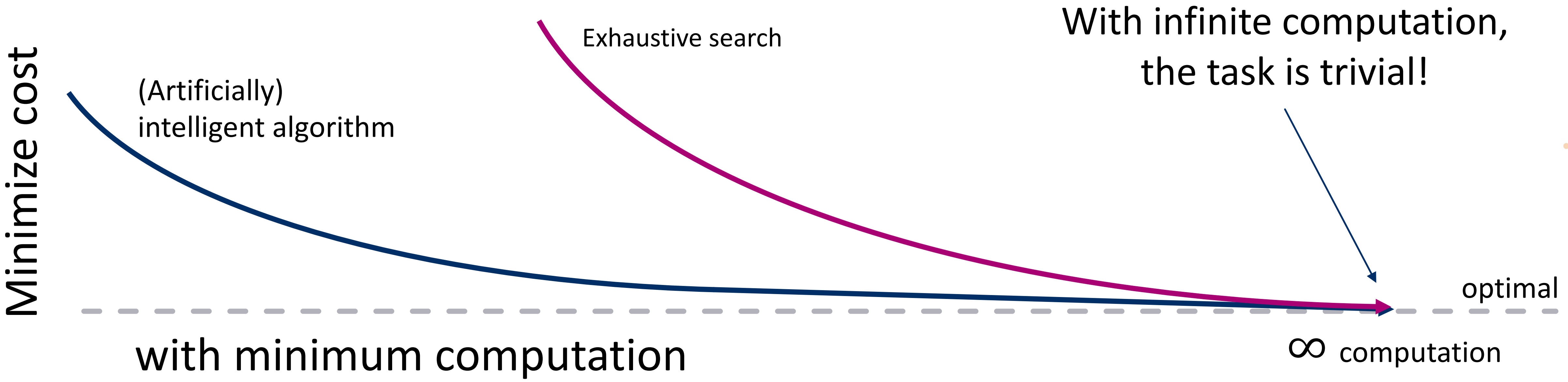
(computation is 2nd)

Minimize cost

with minimum computation

If we have infinite computation…

Combinatorial Optimization

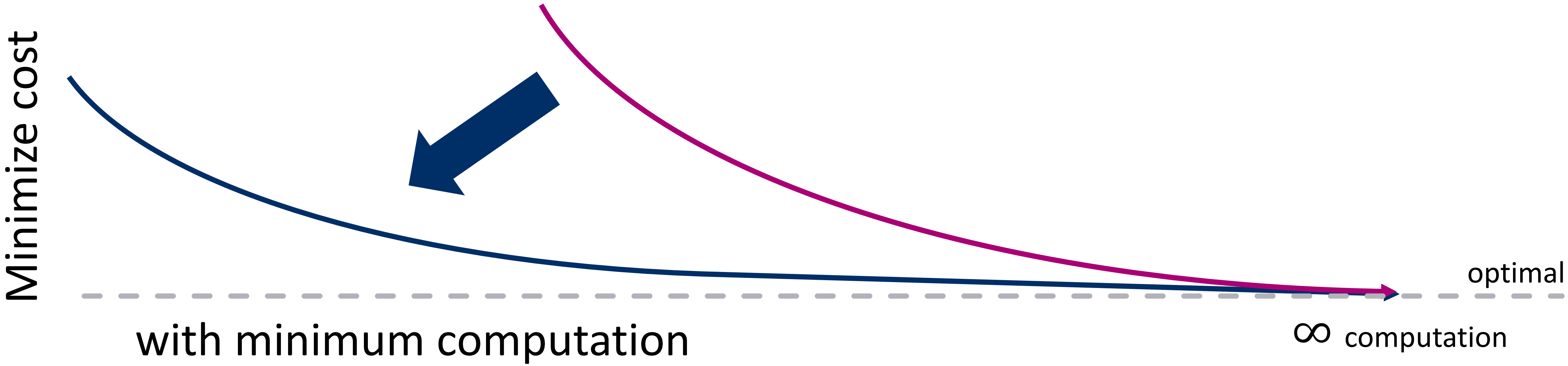


UNIVERSITY OF AMSTERDAM ORTEC

The goal

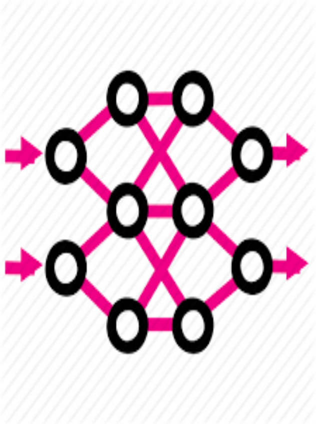
To find better solutions

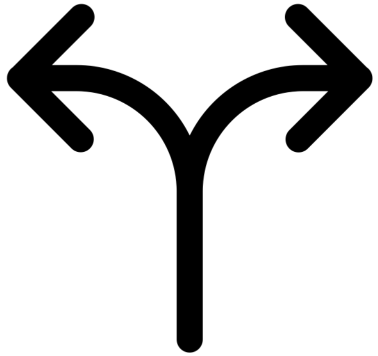
...with less computation!



UNIVERSITY OF AMSTERDAM ORTEC

How?



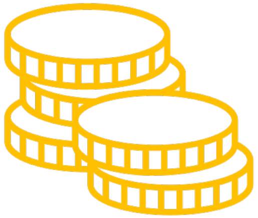


Using neural networks…

…to make better (heuristic) decisions!

Adding computation…

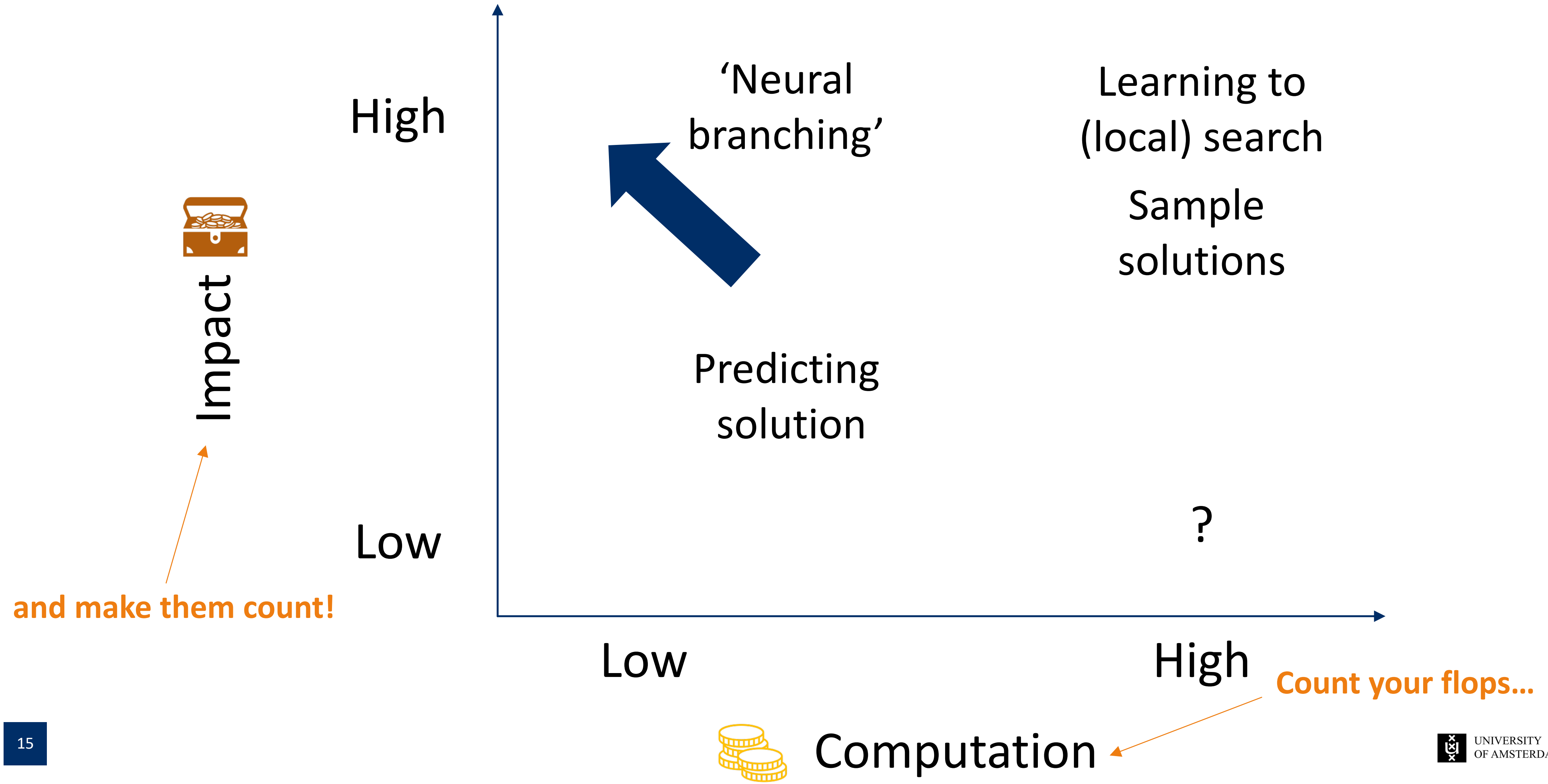
…to reduce computation!



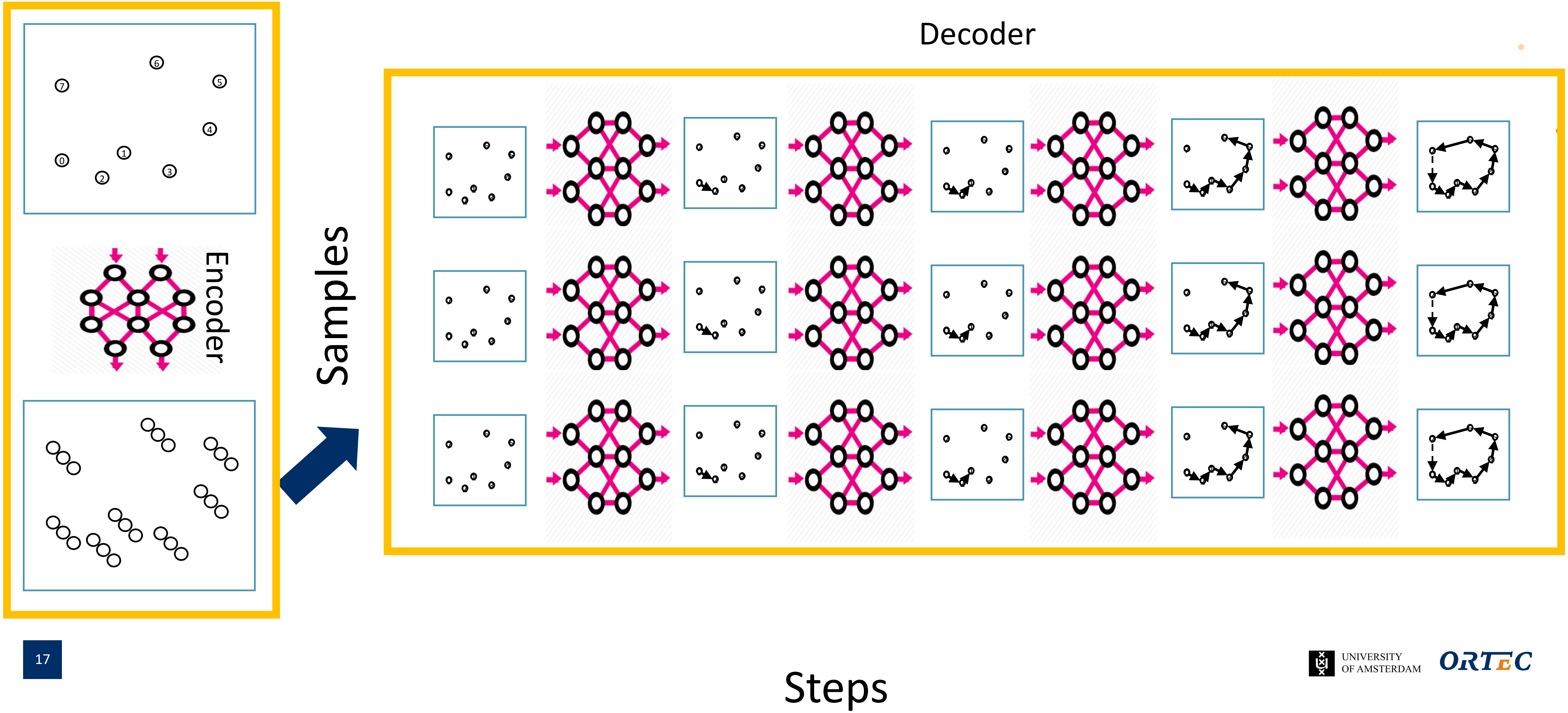
Investment

Pay-off

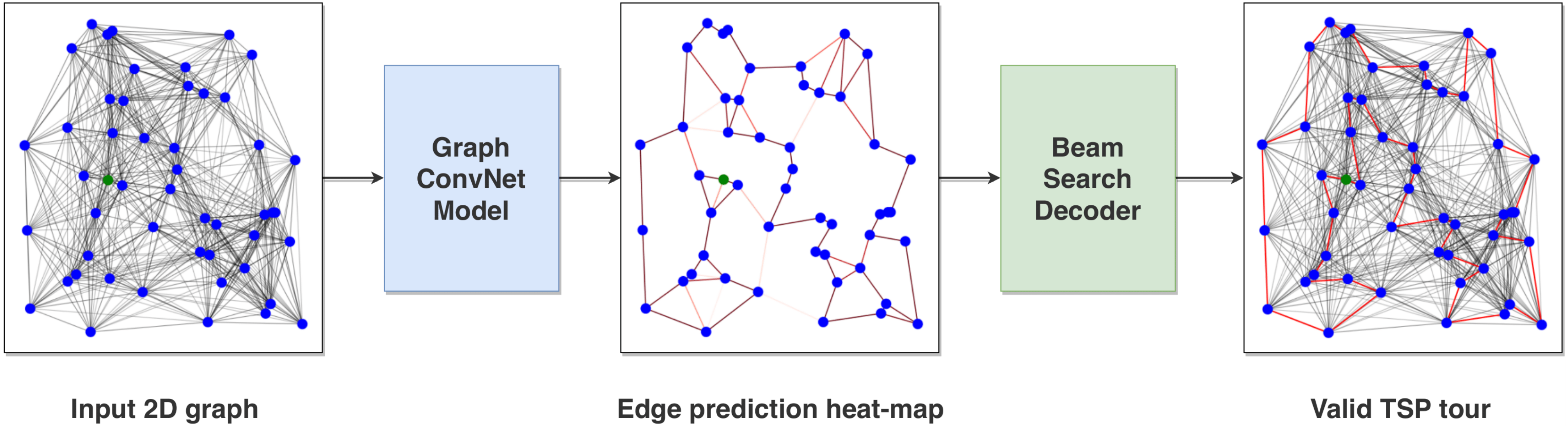
Impact vs. computation (of your neural network)



Example: sampling using Attention Model (Kool et al., 2019)

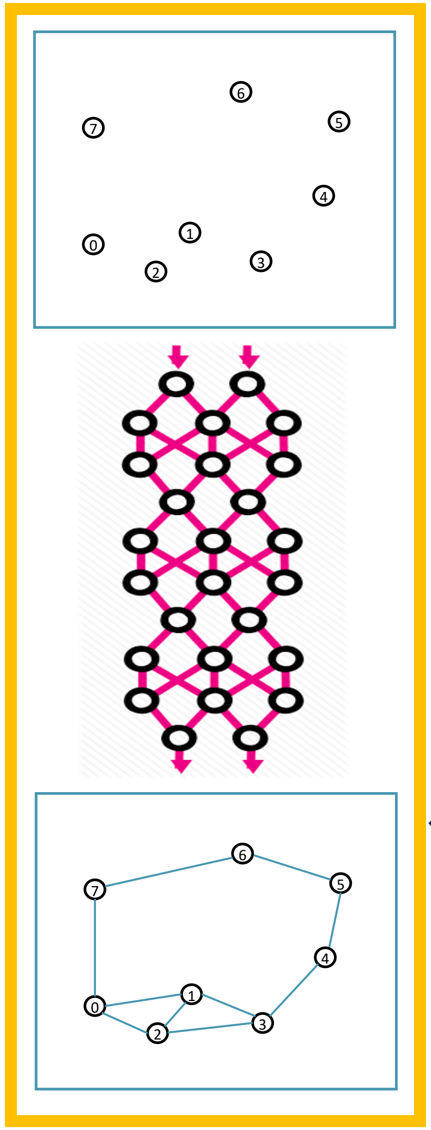


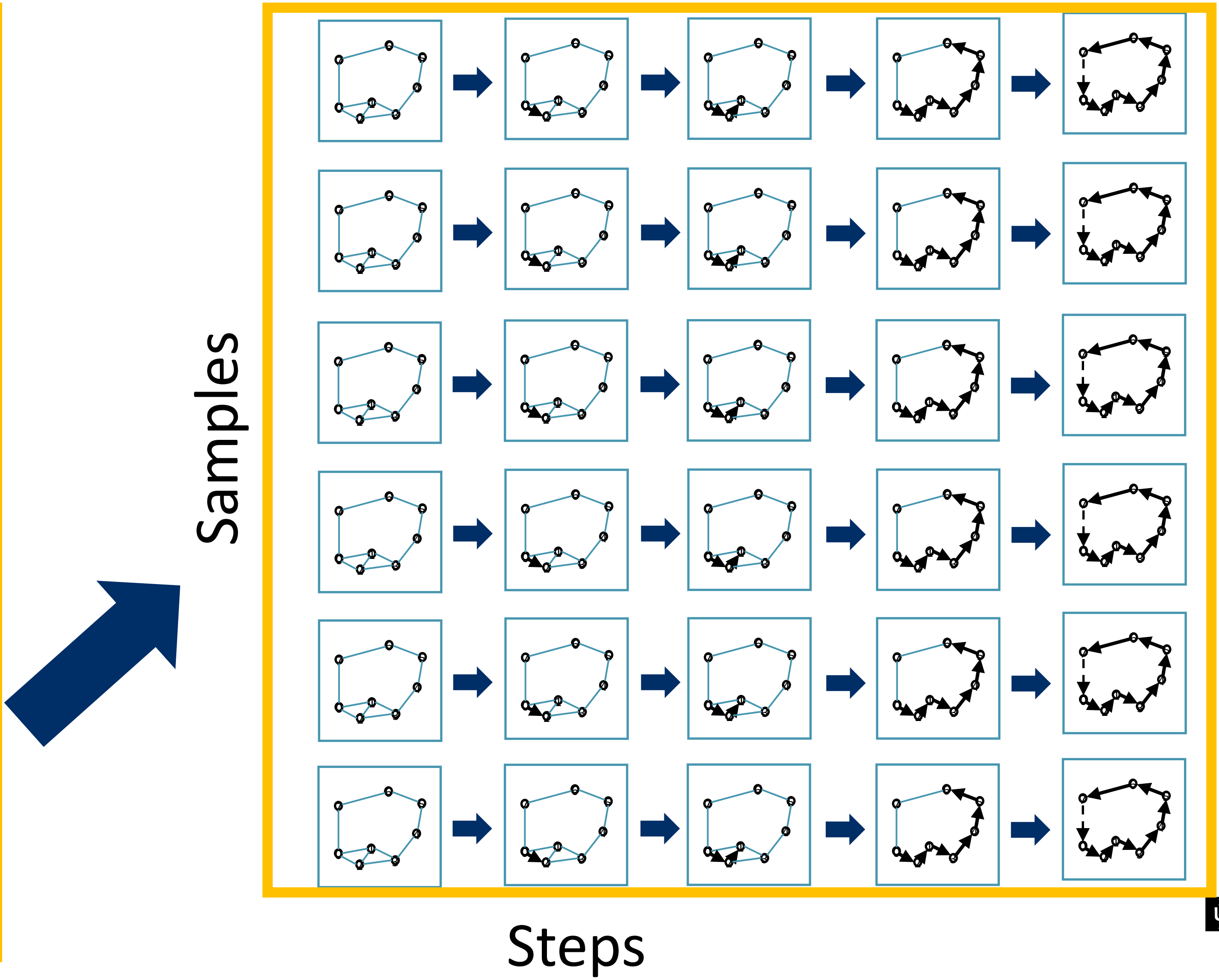
Example: sampling/BS using Heatmap Model (Joshi et al., 2019)



Picture by Joshi et al., 2019

Example: sampling/BS using Heatmap Model (Joshi et al., 2019)





UNIVERSITY OF AMSTERDAM ORTEC

Dynamic Programming

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

$$\underbrace{C(S, i)}_{\text{DP state}} = \min_{j \in S \setminus \{i\}} C(S \setminus \{i\}, j) + c_{ji}$$

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

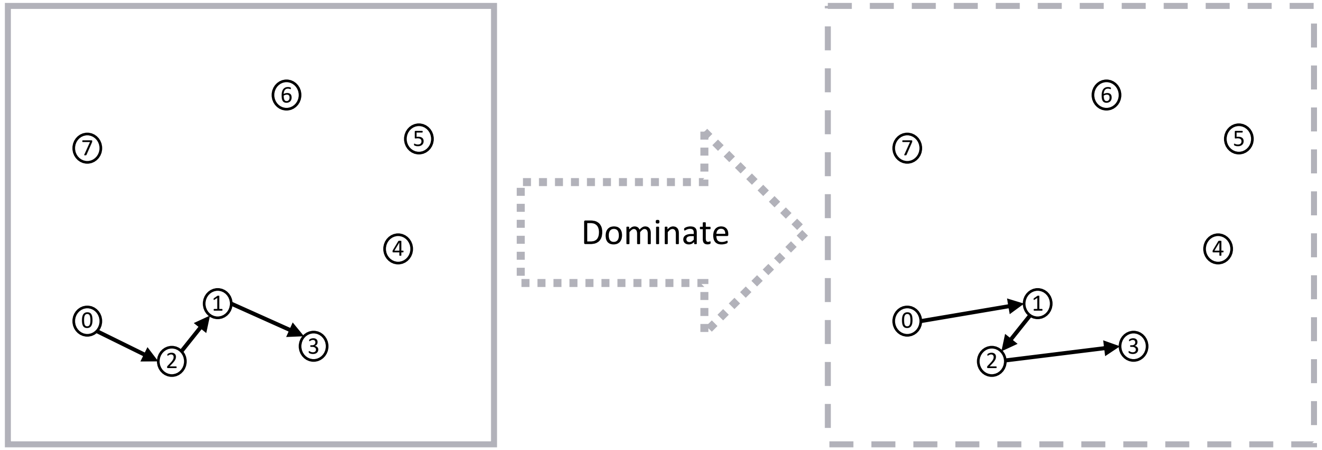
Set of visited nodes

Current node

DP state

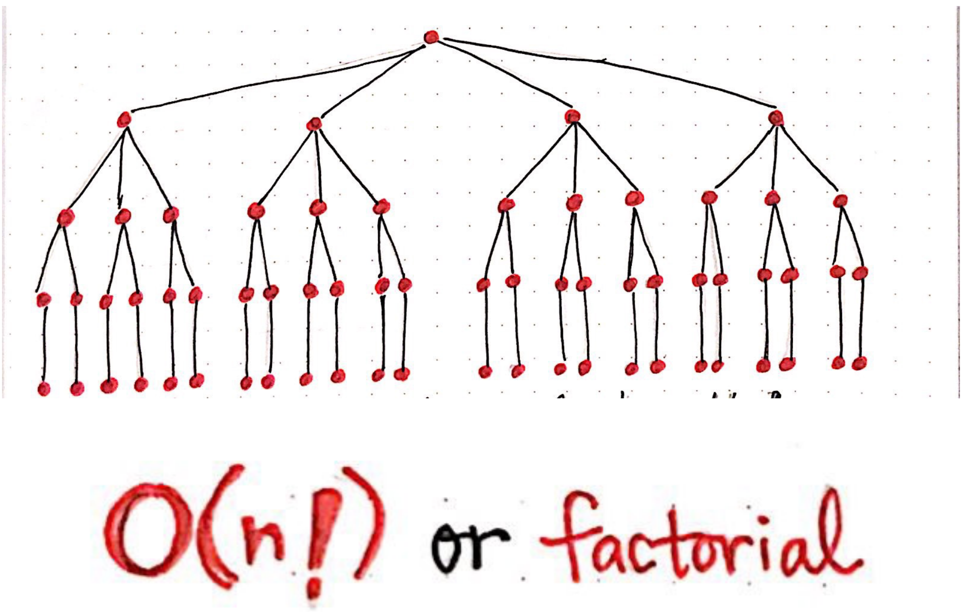
Cost/distance from j to i

Find best solution for each DP state

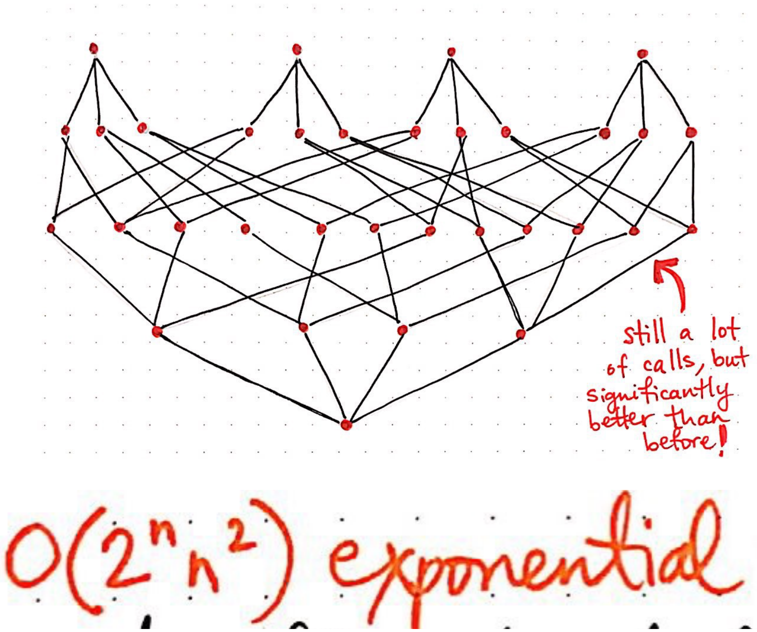


Held-Karp DP for TSP (Held & Karp, 1962; Bellman, 1962)

Brute-force (forward view)



DP (top-down or backward view)

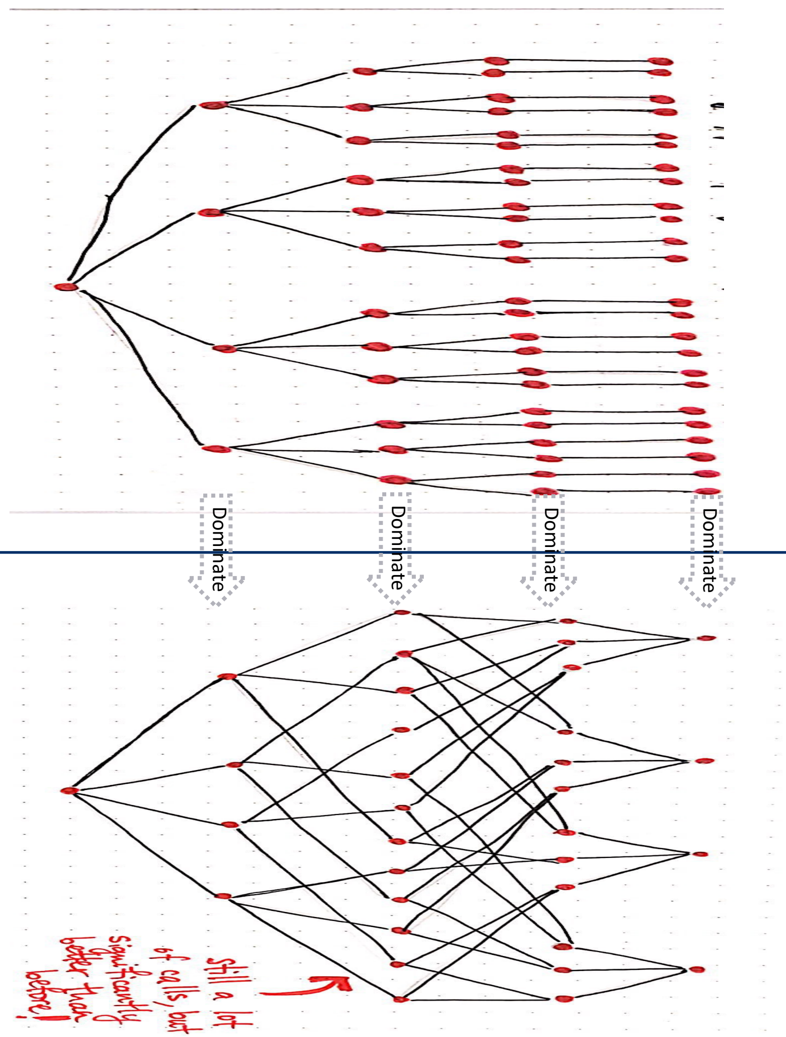


Artwork by Vaidehi Joshi, https://medium.com/basecs/speeding-up-the-traveling-salesman-using-dynamic-programming-b76d7552e8dd

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

Brute-force

$O(n!)$ or factorial



Forward view

Still impractical!

DP

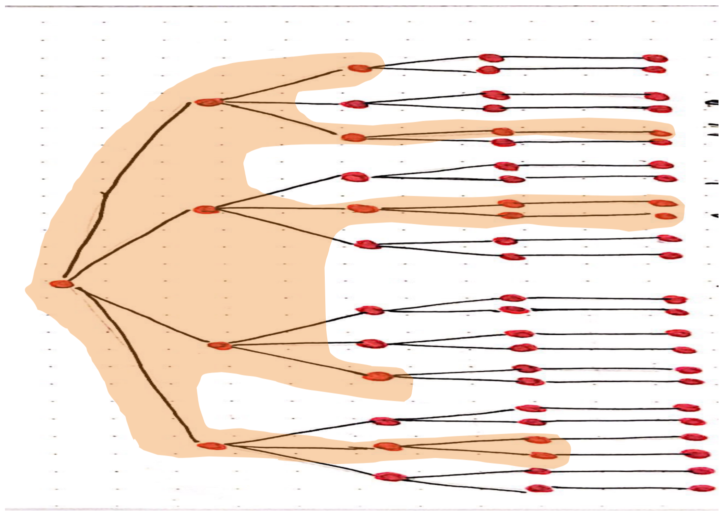
$O(2^n n^2)$ exponential

UNIVERSITY OF AMSTERDAM ORTEC

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

Beam search

$O(Bn)$ or *linear*

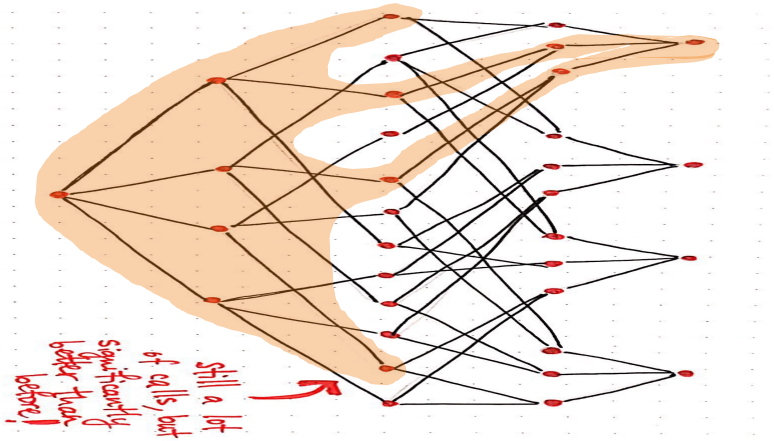


We need a *good policy* to restrict the search space!

Forward view

Restricted DP

$O(Bn)$ or *linear*



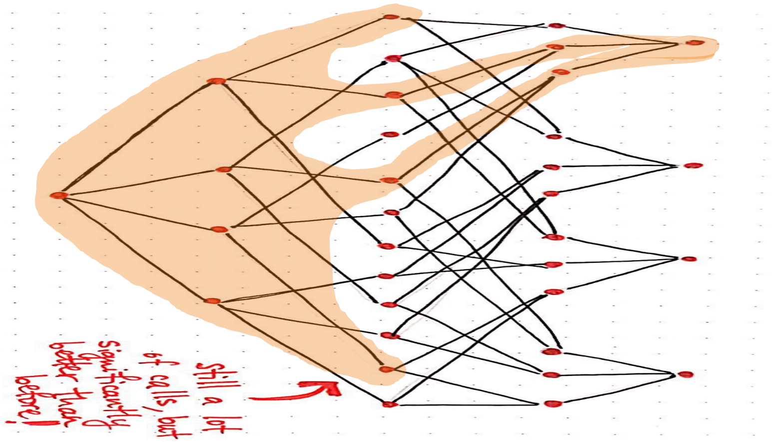
Malandraki & Dial, 1996
Gromicho et al., 2012

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

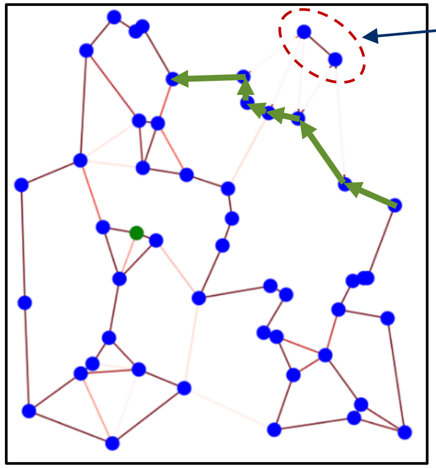
Forward view

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



Potential avoids 'skipped nodes'

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

Picture by Joshi et al., 2019

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!

I hear you thinking…

Show me the pay-off!

Results

Travelling Salesman Problem

Table 1. Main results for TSP with 100 nodes.

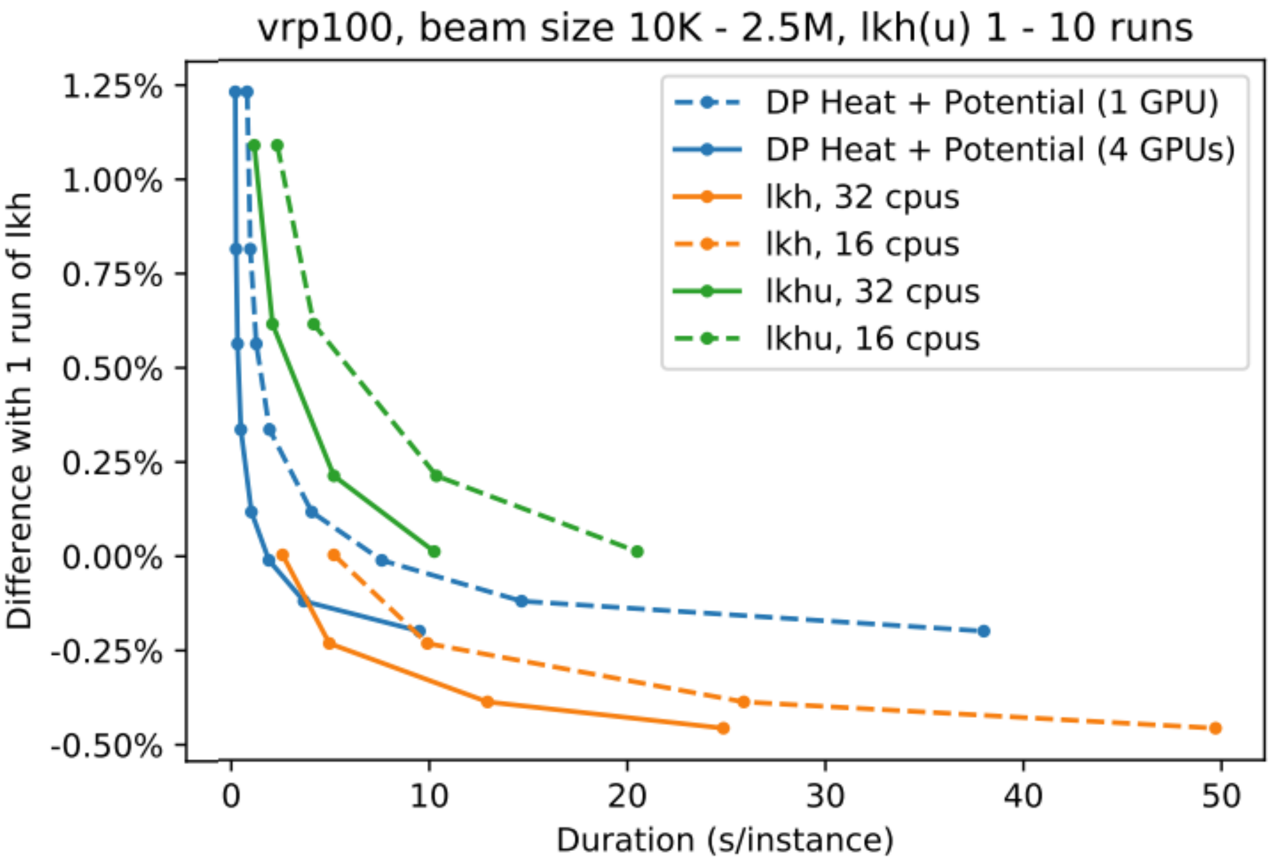
METHOD	COST	GAP	TIME
CONCORDE	7.765	0.000 %	6M
LKH	7.765	0.000 %	42M
GUROBI	7.776	0.15 %	31M
KOOL ET AL. (2019)	7.94	2.26 %	1H
JOSHI ET AL. (2019A)	7.87	1.39 %	40M
DA COSTA ET AL. (2020)	7.83	0.87 %	41M
FU ET AL. (2020)	7.764*	0.04 %	4M + 11M
DPDP 10K	7.765	0.009 %	10M + 1H06M
DPDP 100K	7.765	0.004 %	10M + 2H35M

Vehicle Routing Problem

Table 2. Main results for VRP with 100 nodes.

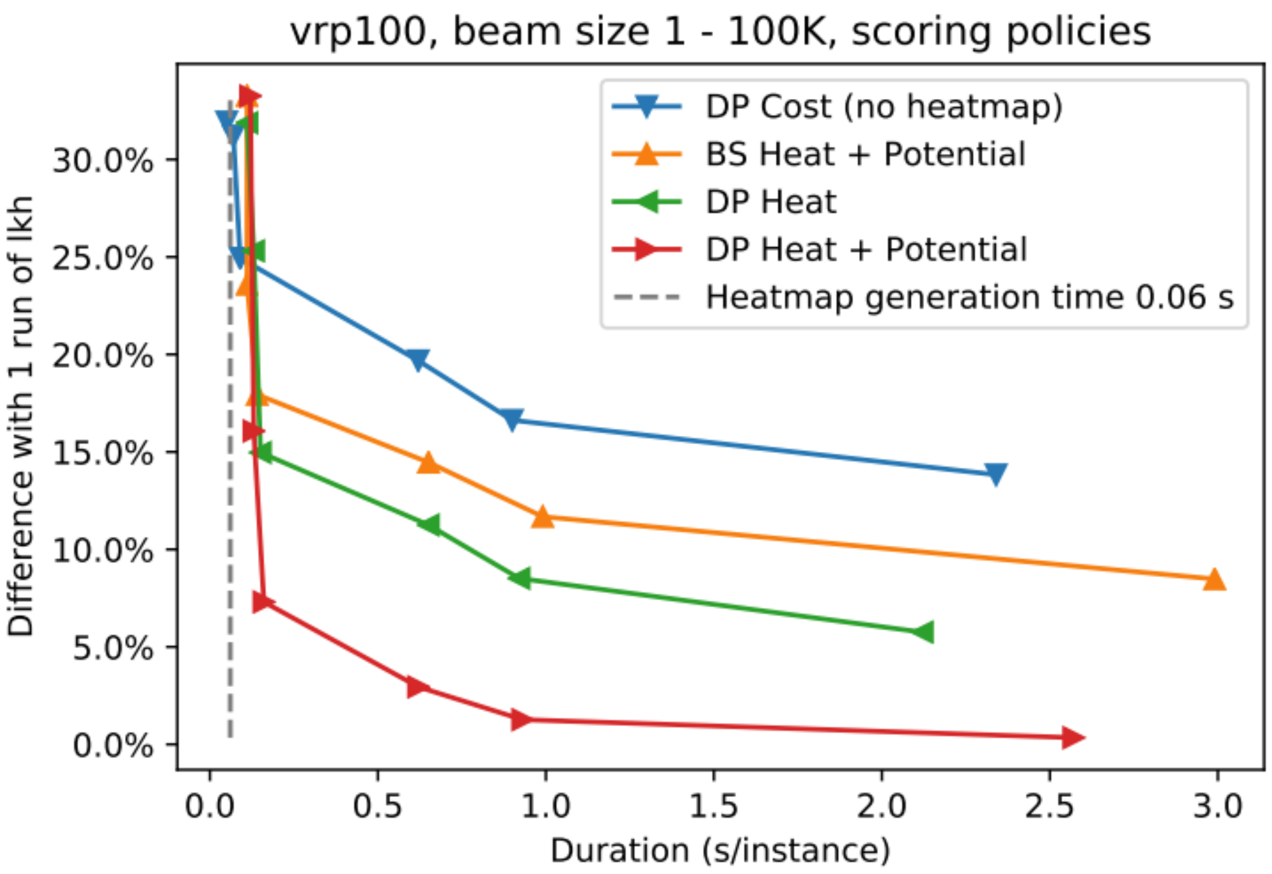
METHOD	COST	GAP	TIME
LKH	15.647	0.000 %	12H59M
XIN ET AL. (2020)	16.49	4.99 %	39S
KOOL ET AL. (2019)	16.23	3.72 %	2H
CHEN & TIAN (2019)	16.10	2.90 %	1H
PENG ET AL. (2019)	16.27	3.96 %	6H
WU ET AL. (2019)	16.03	2.47 %	5H
HOTTUNG & TIERNEY (2019)	15.99	1.02 %	1H
LU ET AL. (2020)	15.57*	-	4000H
DPDP 10K	15.832	1.183 %	10M + 2H24M
DPDP 100K	15.694	0.305 %	10M + 5H48M
DPDP 1M	15.627	- 0.127 %	10M + 48H27M

Quality vs. computation



UNIVERSITY OF AMSTERDAM ORTEC

Ablations



vrp100, beam size 1 - 100K, scoring policies
DP Cost (no heatmap)
BS Heat + Potential
DP Heat
DP Heat + Potential
Heatmap generation time 0.06 s
30.0%
25.0%
20.0%
15.0%
10.0%
5.0%
0.0%
Difference with 1 run of lkh
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Duration (s/instance)

That’s it! So remember…

Count your flops…

and make them count!