Shortest route to success

Wouter Kool & co(lleagues)
Joep Olde Juninck
Ernst Roos
Kamiel Cornelissen
Pim Agterberg
Jelke van Hoorn
Thomas Visser

Public

ORTEC

Together we increase our impact on the world

CARGLASS
coles
FedEx
Heineken
MOLSON Coors
P&G
SHV
SIRVA
Airgas
brussels airport
Erasmus MC
G4S
Initial
Kraft
selecta
Shell

About ORTEC

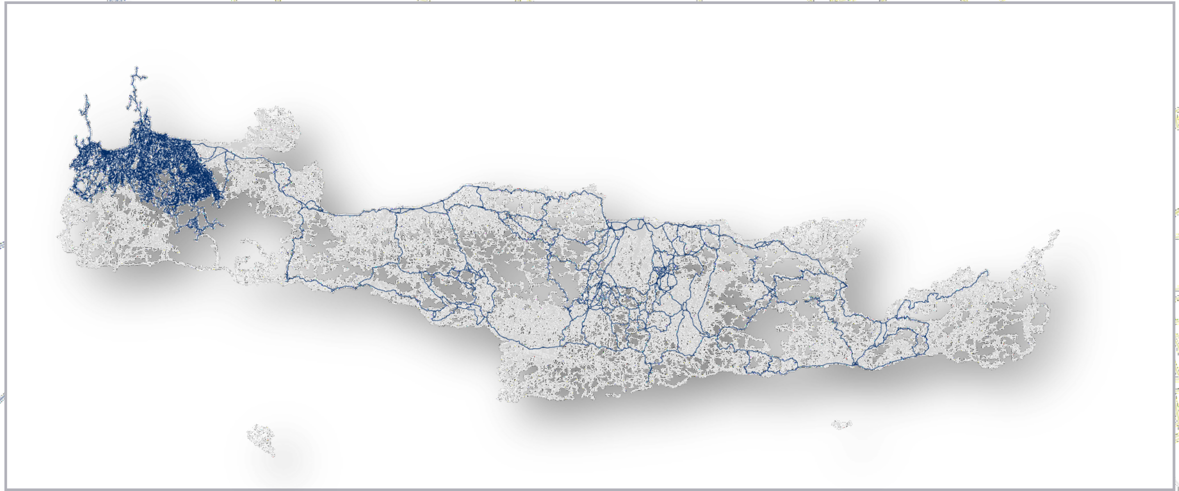
- Founded in 1981
- Offices in 13 countries
- 1.200 customers
- > 1.000 employees

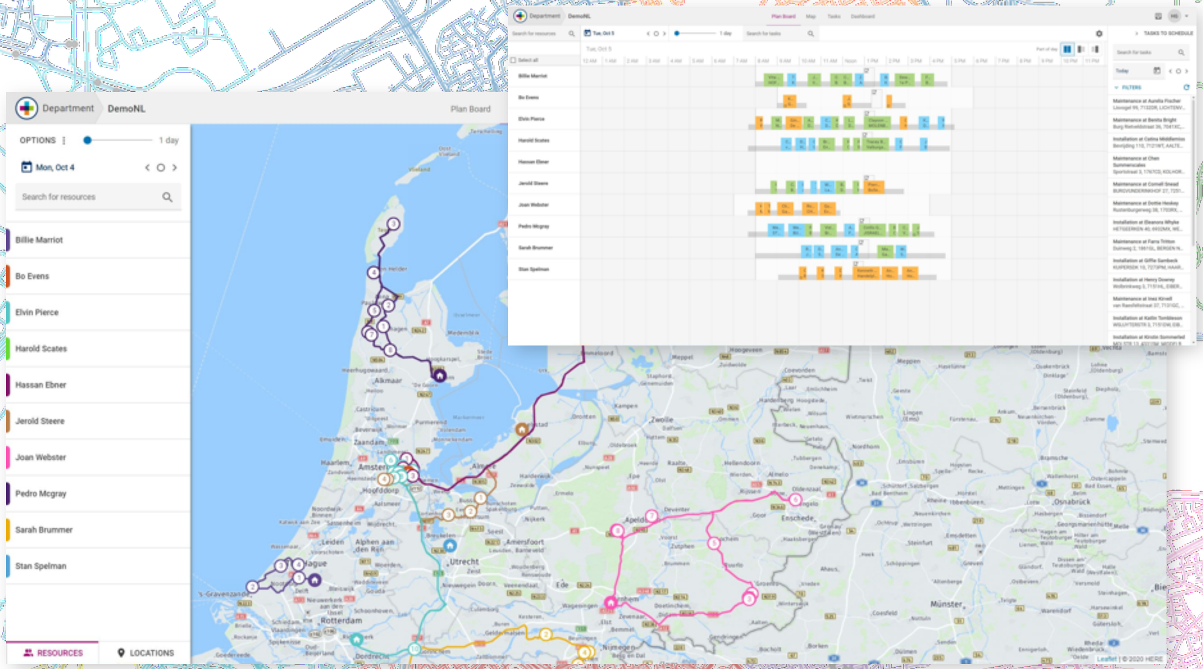
At ORTEC… we make drivers happy!

What does that mean?

"*We plan routes over roads*
We take into account traffic
We don't make drivers wait
(but we do give them breaks)"

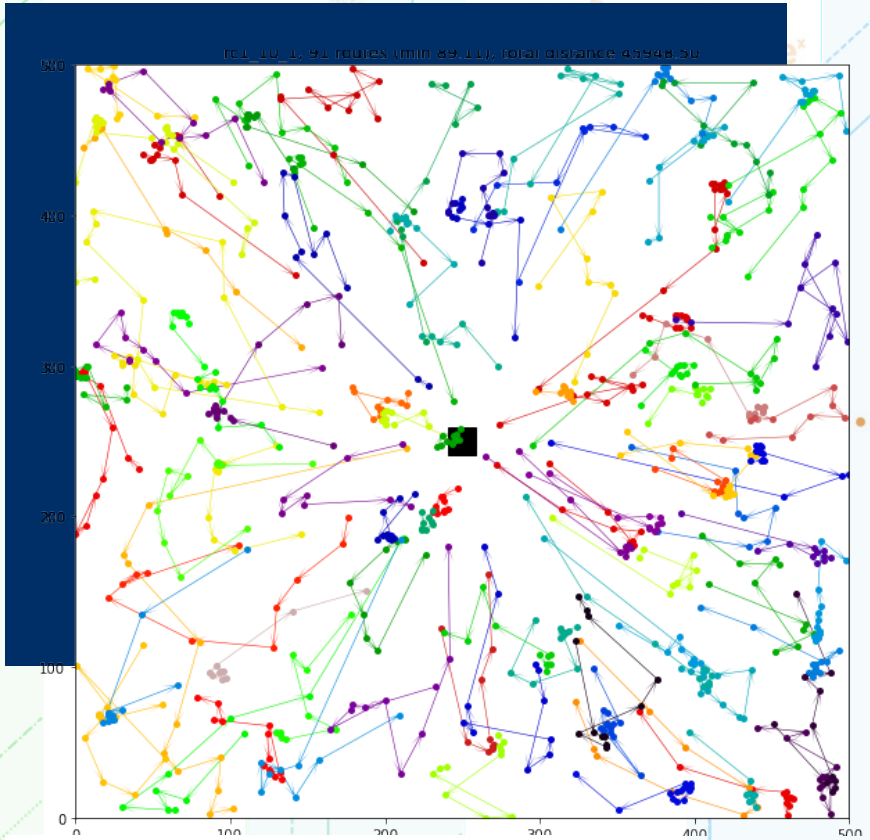
Routing @ ORTEC





In the VRPTW challenge

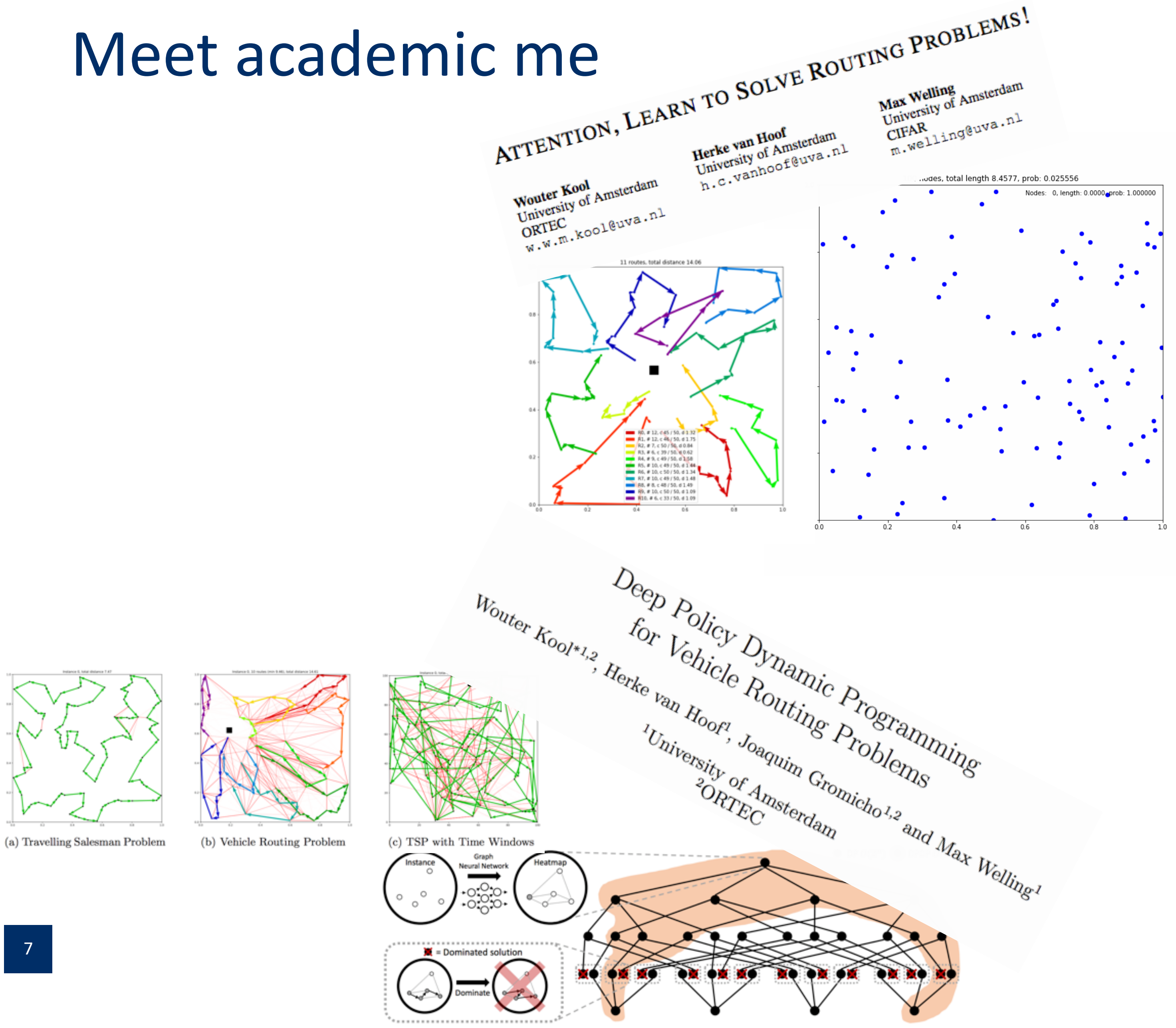
- We drive as the crow flies
- There is no traffic
- We wait a lot
- (maybe that's why we don't take breaks)



We need something special

ORTEC

Meet academic me



What you get for 25 000 000 $

"

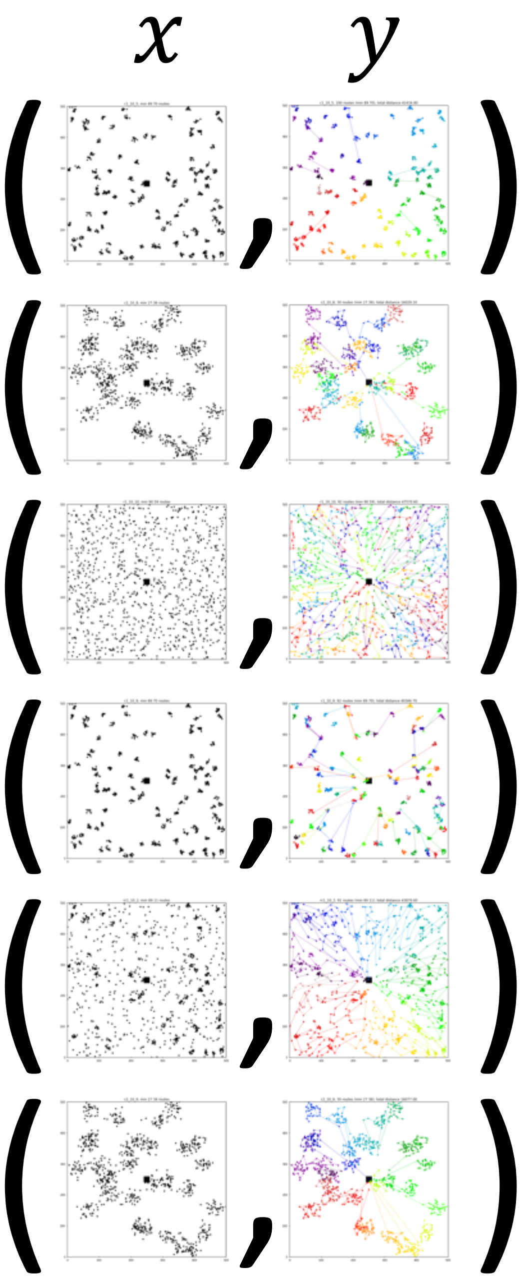
Training machine learning models takes a lot of time and effort

"

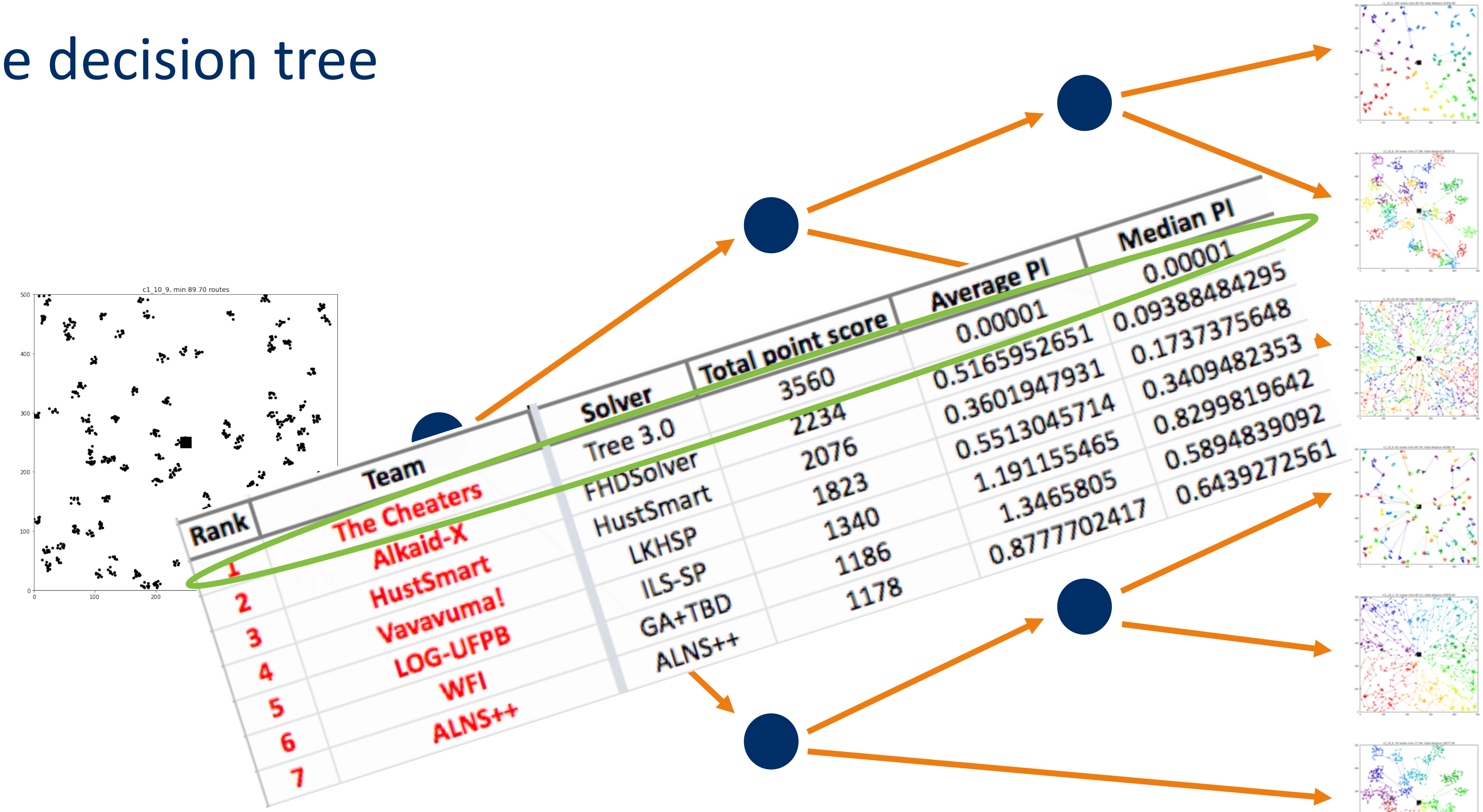
https://www.nature.com/articles/nature.2017.22858

ORTEC

The dataset



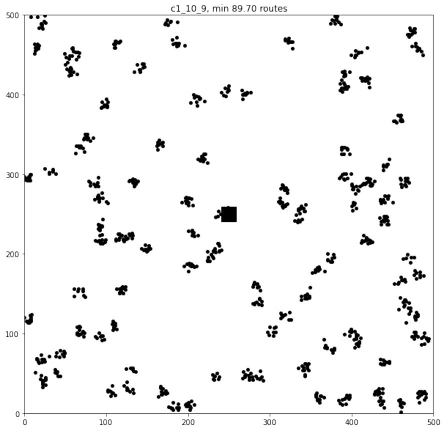
The decision tree

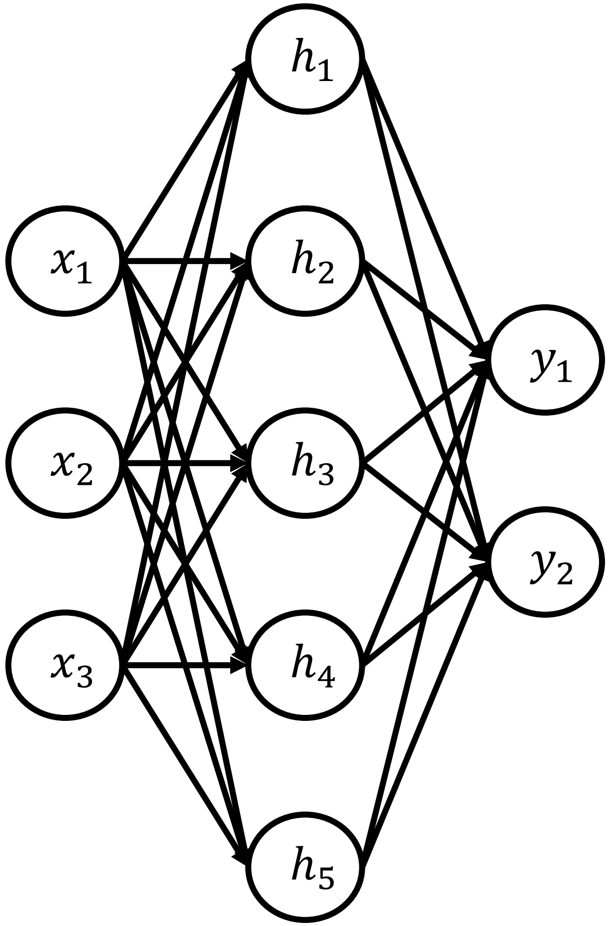


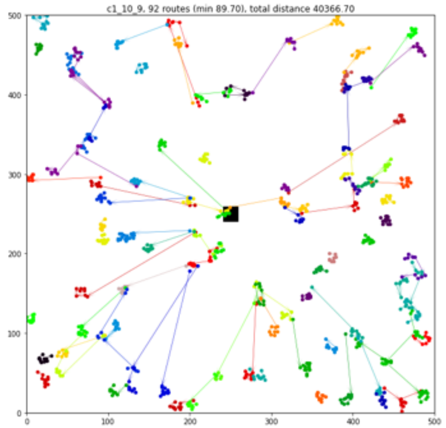
Rank	Team	Solver	Total point score	Average PI	Median PI
1	The Cheaters	Tree 3.0	3560	0.00001	0.00001
2	Alkaid-X	FHDSolver	2234	0.5165952651	0.09388484295
3	HustSmart	HustSmart	2076	0.3601947931	0.1737375648
4	Vavavuma!	LKHSP	1823	0.5513045714	0.3409482353
5	LOG-UFPB	ILS-SP	1340	1.191155465	0.8299819642
6	WFI	GA+TBD	1186	1.3465805	0.5894839092
7	ALNS++	ALNS++	1178	0.8777702417	0.6439272561

(don't mention phase 2)

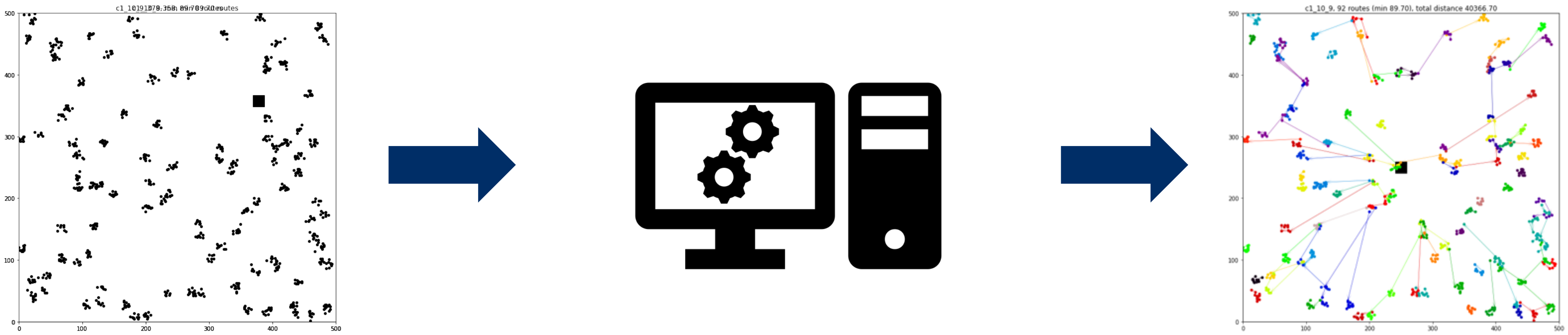
The neural network







The heavily tuned algorithm

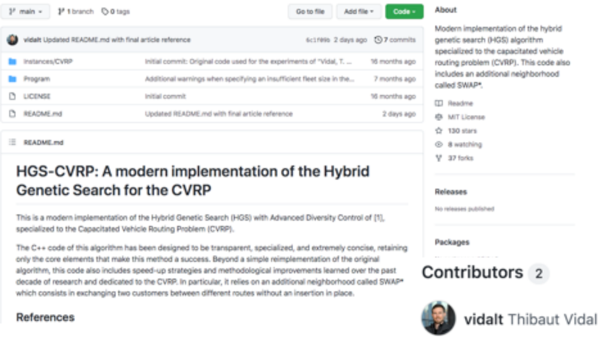


Conclusion

No machine learning here ☹
Use good old human intelligence?

The recipe (by culinary me)

1 open-source repo



main · 1 branch · 0 tags · Go to file · Add file · Code

vidalt Updated README.md with final article reference · 2 days ago · 7 commits

Instances/CVRP	Initial commit: Original code used for the experiments of "Vidal, T. ...	16 months ago
Program	Additional warnings when specifying an insufficient fleet size in the...	7 months ago
LICENSE	Initial commit	16 months ago
README.md	Updated README.md with final article reference	2 days ago

README.md

HGS-CVRP: A modern implementation of the Hybrid Genetic Search for the CVRP

This is a modern implementation of the Hybrid Genetic Search (HGS) with Advanced Diversity Control of [1], specialized to the Capacitated Vehicle Routing Problem (CVRP).

The C++ code of this algorithm has been designed to be transparent, specialized, and extremely concise, retaining only the core elements that make this method a success. Beyond a simple reimplementation of the original algorithm, this code also includes speed-up strategies and methodological improvements learned over the past decade of research and dedicated to the CVRP. In particular, it relies on an additional neighborhood called SWAP* which consists in exchanging two customers between different routes without an insertion in place.

References

About

Modern implementation of the hybrid genetic search (HGS) algorithm specialized to the capacitated vehicle routing problem (CVRP). This code also includes an additional neighborhood called SWAP*.

Readme · MIT License · 130 stars · 8 watching · 37 forks

Releases

No releases published

Packages

Contributors 2

vidalt Thibaut Vidal

1+ excellent paper

A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows

Thibaut Vidal Teodor Gabriel Crainic [c], Michel Gendreau [d], Christian Prins [a]

...ISL Université de Technologie de Troyes, 12 rue Marie Curie, BP 2060, 10010 Troyes Cedex, France
[b] CIRRELT et Département d'informatique et de recherche opérationnelle, Université de Montréal, Montréal, Canada H3C 3J7
[c] CIRRELT et Département de management et technologie, École des sciences de la gestion, UQAM, Montréal, Canada H3C 3P8
[d] CIRRELT et Département de mathématiques et génie industriel, École Polytechnique, Montréal, Canada H3C 3A7

ARTICLE INFO

Available online 31 July 2012

Keywords:
Vehicle routing problems
Time windows
Hybrid genetic algorithm
Diversity management
Neighbourhood search
Decomposition

ABSTRACT

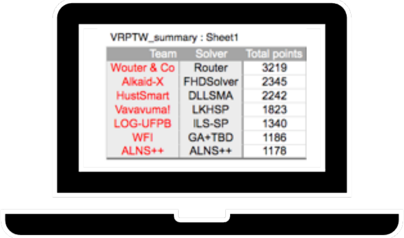
The paper presents an efficient Hybrid Genetic Search with Advanced Diversity Control for a large class of time-constrained vehicle routing problems, introducing several new features to manage the temporal dimension. New move evaluation techniques are proposed, accounting for penalized infeasible solutions with respect to time-window and duration constraints, and allowing to evaluate moves from any classical neighbourhood based on arc or node exchanges in amortized constant time. Furthermore, geometric and structural problem decompositions are developed to address efficiently large problems. The proposed algorithm outperforms all current state-of-the-art approaches on classical literature benchmark instances for any combination of periodic, multi-depot, site-dependent, and duration-constrained vehicle routing problem with time windows.

© 2012 Elsevier Ltd. All rights reserved.

6 intelligent colleagues

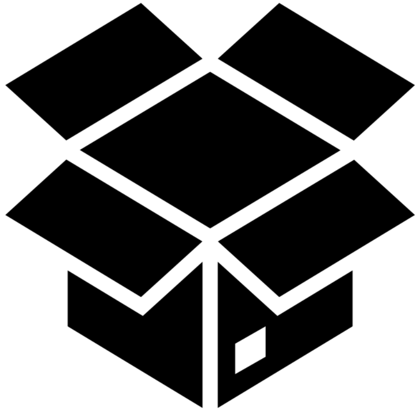
Joep Olde Junninck · Ernst Roos · Pim Agterberg
Kamiel Cornelissen · Thomas Visser · Jelke van Hoorn

Add some oil



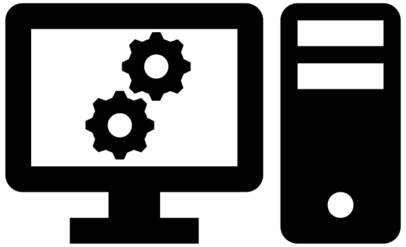
Team	Solver	Total points
Wouter & Co	Router	3219
Alkaid-X	FHDSolver	2345
HustSmart	DLLSMA	2242
Vavavuma!	LKHSP	1823
LOG-UFPB	ILS-SP	1340
WFI	GA+TBD	1186
ALNS++	ALNS++	1178

Submit & wait for results



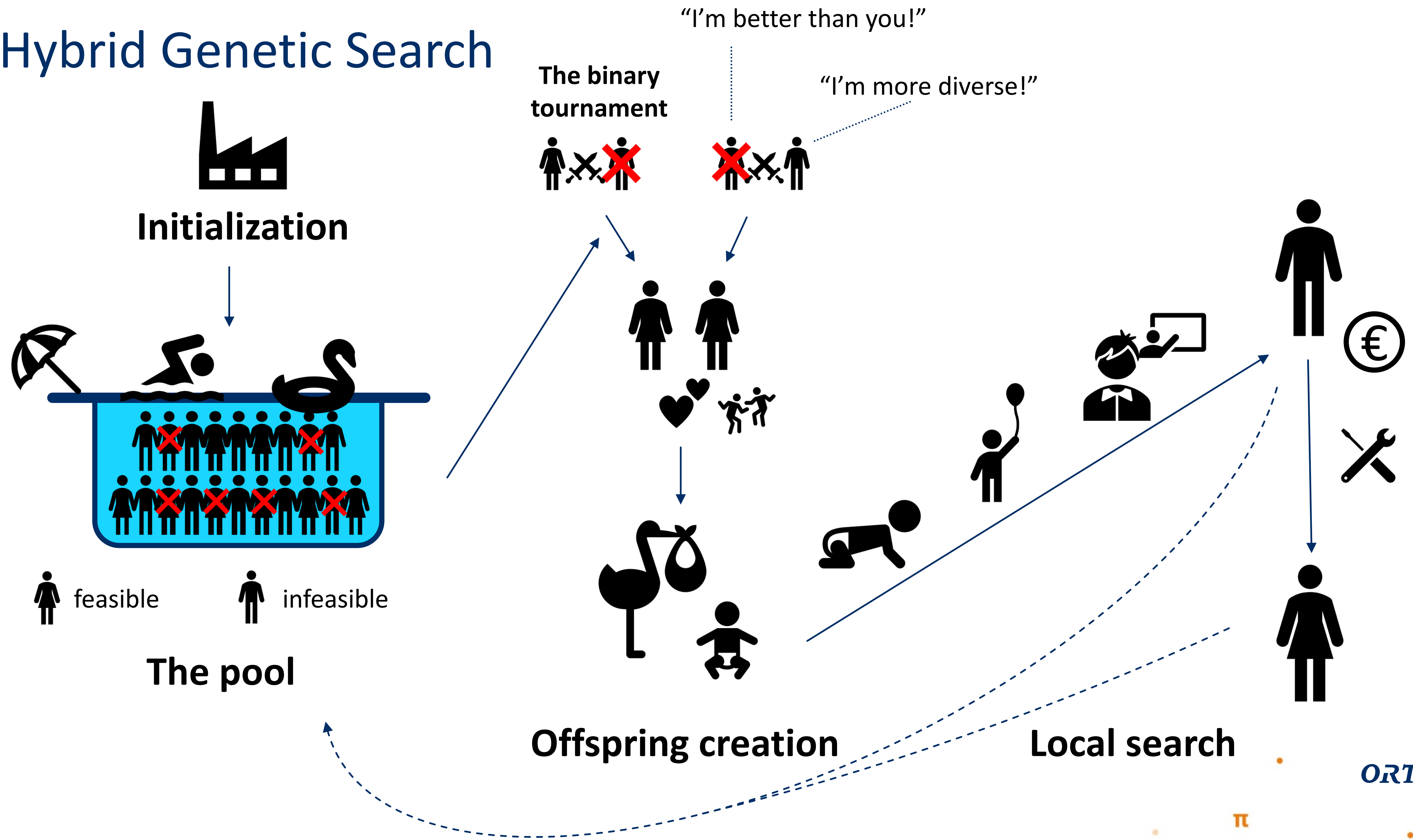
Put in a black box (a.k.a. online meeting)

Mix for 1 week



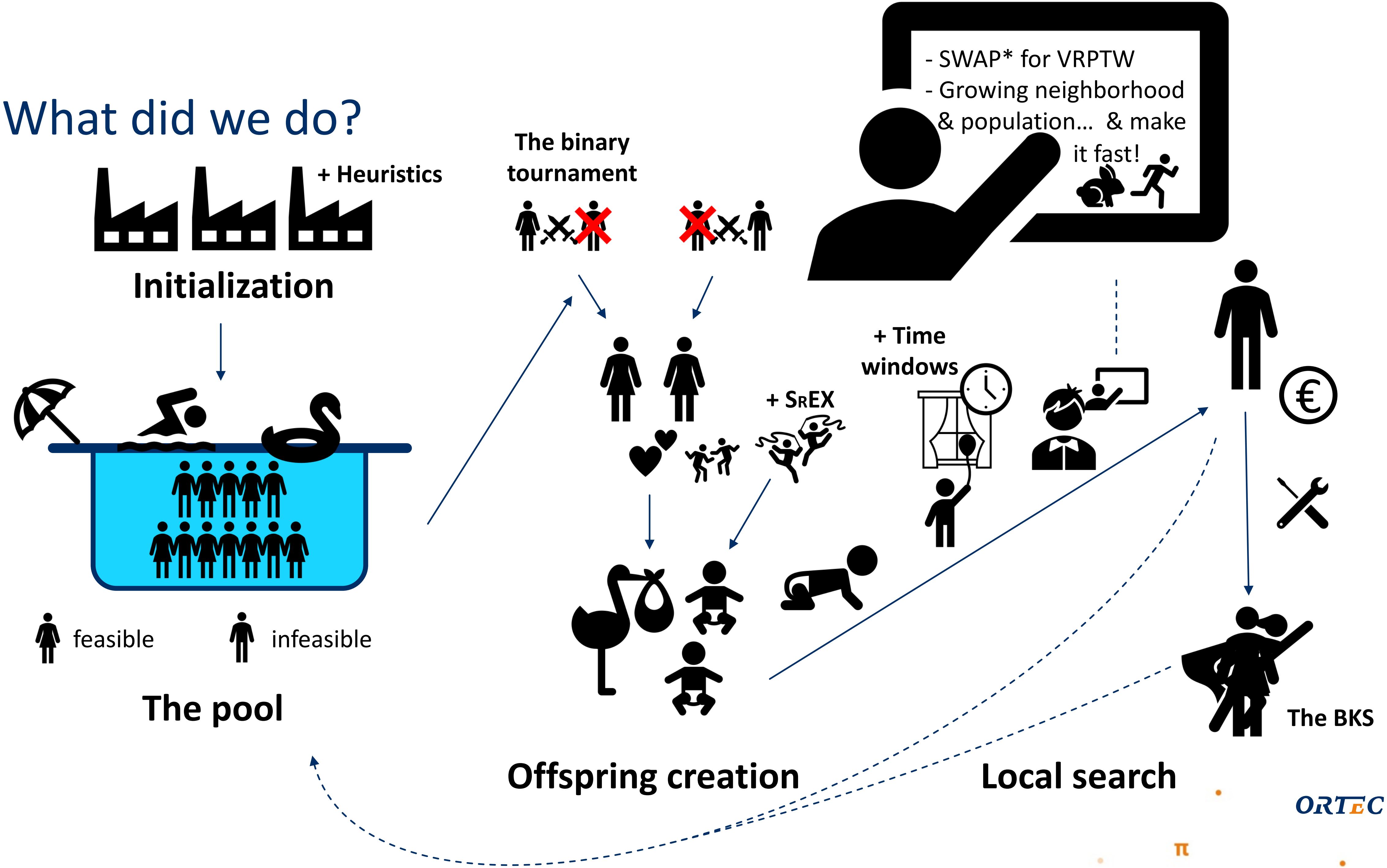
Run algorithm for 3 days

Hybrid Genetic Search



ORTEC

What did we do?



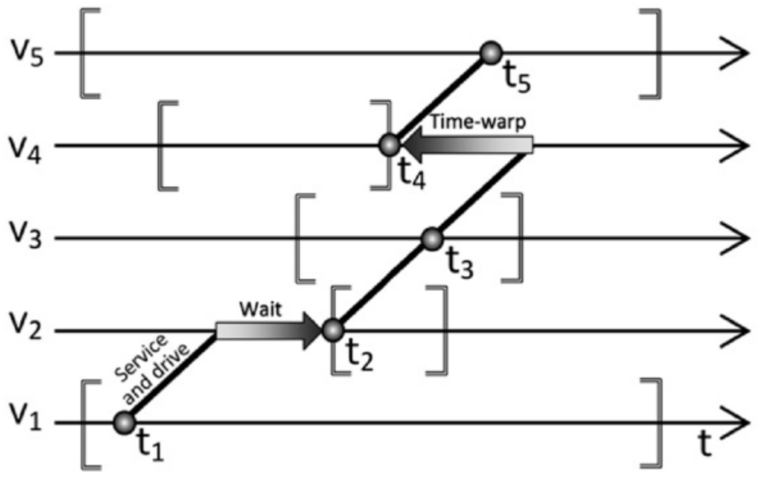


Fig. 1. Illustration of waiting times and time warps.

Proposition 1 (*Concatenation of two sequences*). *Let* $\sigma = (\sigma_i, \dots, \sigma_j)$ *and* $\sigma' = (\sigma'_{i'}, \dots, \sigma'_{j'})$ *be two subsequences of visits. The concatenated subsequence* $\sigma \oplus \sigma'$ *is characterized by the following data:*

$$D(\sigma \oplus \sigma') = D(\sigma) + D(\sigma') + \delta_{\sigma_j \sigma'_{i'}} + \Delta_{WT} \tag{5}$$

$$TW(\sigma \oplus \sigma') = TW(\sigma) + TW(\sigma') + \Delta_{TW} \tag{6}$$

$$E(\sigma \oplus \sigma') = \max\{E(\sigma') - \Delta, E(\sigma)\} - \Delta_{WT} \tag{7}$$

$$L(\sigma \oplus \sigma') = \min\{L(\sigma') - \Delta, L(\sigma)\} + \Delta_{TW} \tag{8}$$

$$C(\sigma \oplus \sigma') = C(\sigma) + C(\sigma') + c_{\sigma_j \sigma'_{i'}} \tag{9}$$

$$Q(\sigma \oplus \sigma') = Q(\sigma) + Q(\sigma') \tag{10}$$

where $\Delta = D(\sigma) - TW(\sigma) + \delta_{\sigma_j \sigma'_{i'}}$, $\Delta_{WT} = \max\{E(\sigma') - \Delta - L(\sigma), 0\}$ *and* $\Delta_{TW} = \max\{E(\sigma) + \Delta - L(\sigma'), 0\}$.

Source: Vidal et al. 2012

Supporting time windows

- Use time-warp principle
- Cache computation for prefix and postfix of routes
- Use two-level hierarchy for fast queries in middle of route
- Penalty booster: increase penalty by 100% if no feasible solution found

ORTEC

Initial heuristics

- 85% random:
  - Randomly order nodes
  - Split into routes using SPLIT algorithm (Vidal, 2012)
- 5% farthest, 5% nearest insertion:
  - Insert the node with shortest detour (given this is feasible)
  - If no feasible insertion, start new route with farthest/nearest node from depot
- 5% sweep:
  - Sort by angle to depot and add until capacity violated
  - For each route, sort customers with short time windows by time
  - For each route, insert customers with long time windows by shortest detour

Offspring generation

Selective Route Exchange (SREX)

540 Y. Nagata and S. Kobayashi

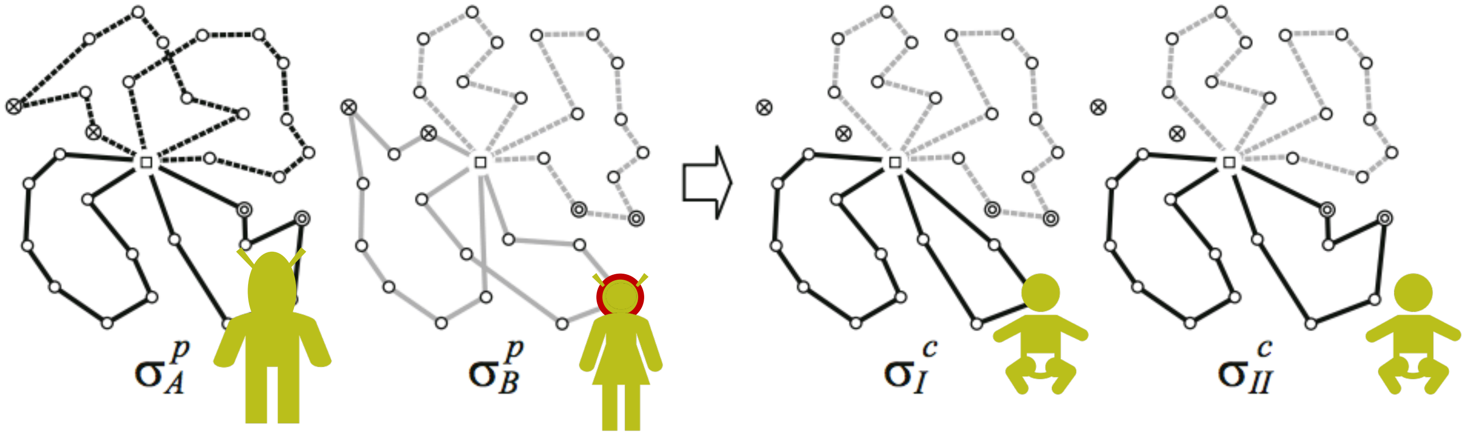


Fig. 1. Illustration of the SREX. σ_A^p and σ_B^p are parents. Routes S_A and S_B are represented as dotted lines, customer nodes $V_{A \setminus B}$ are represented by circles with x-mark, and the customer nodes in $V_{B \setminus A}$ are represented by double circles. σ_I^c and σ_{II}^c are intermediate offspring solutions obtained after Step 2.

Source: Nagata et al. 2010

Ordered Crossover (OX)

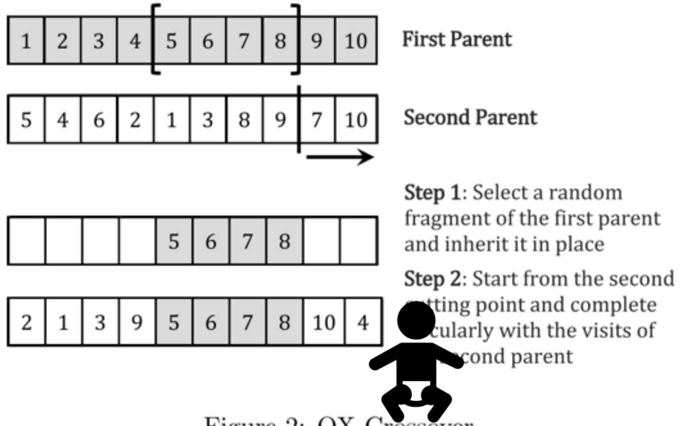


Figure 2: OX Crossover

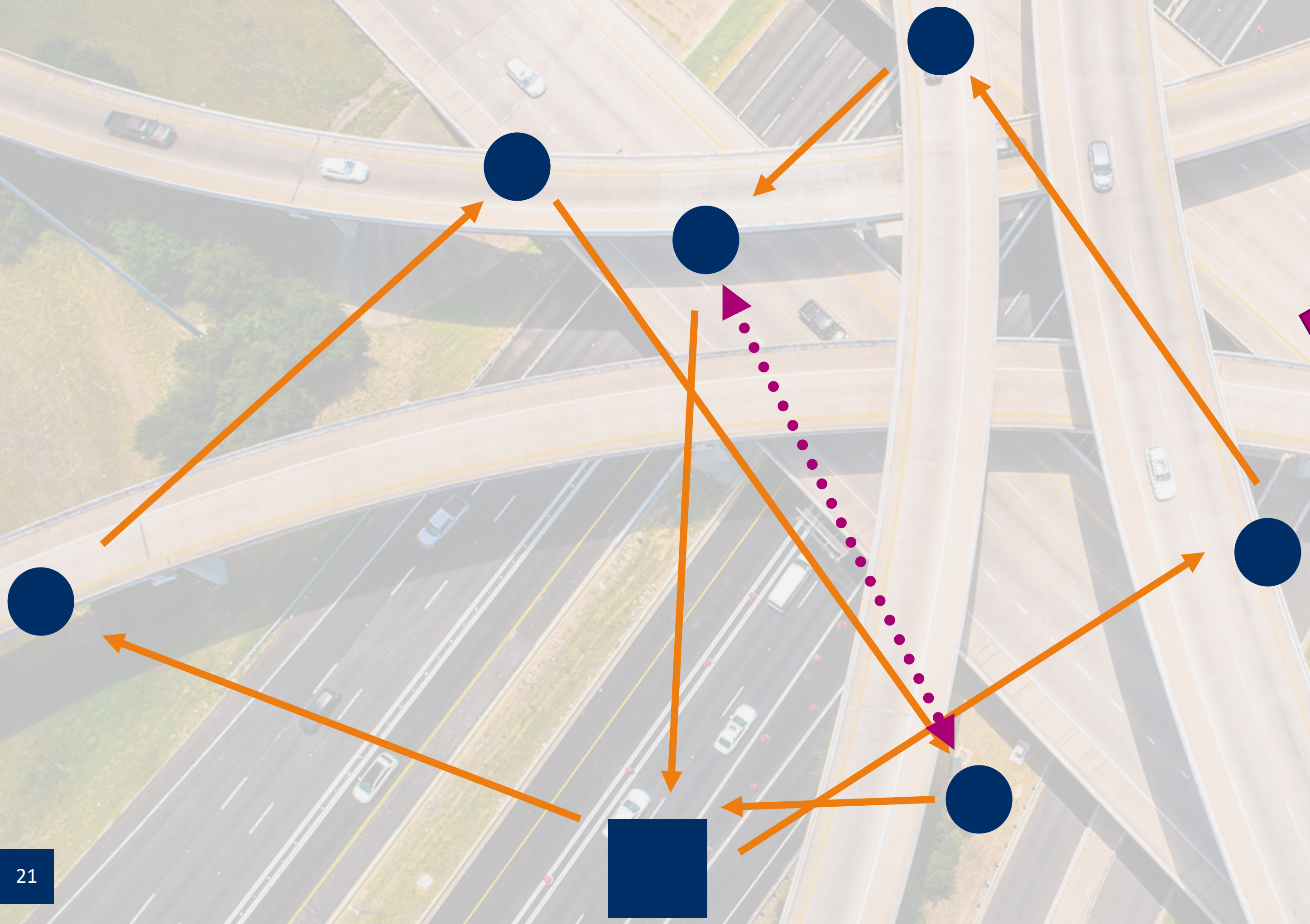
Source: Vidal 2021

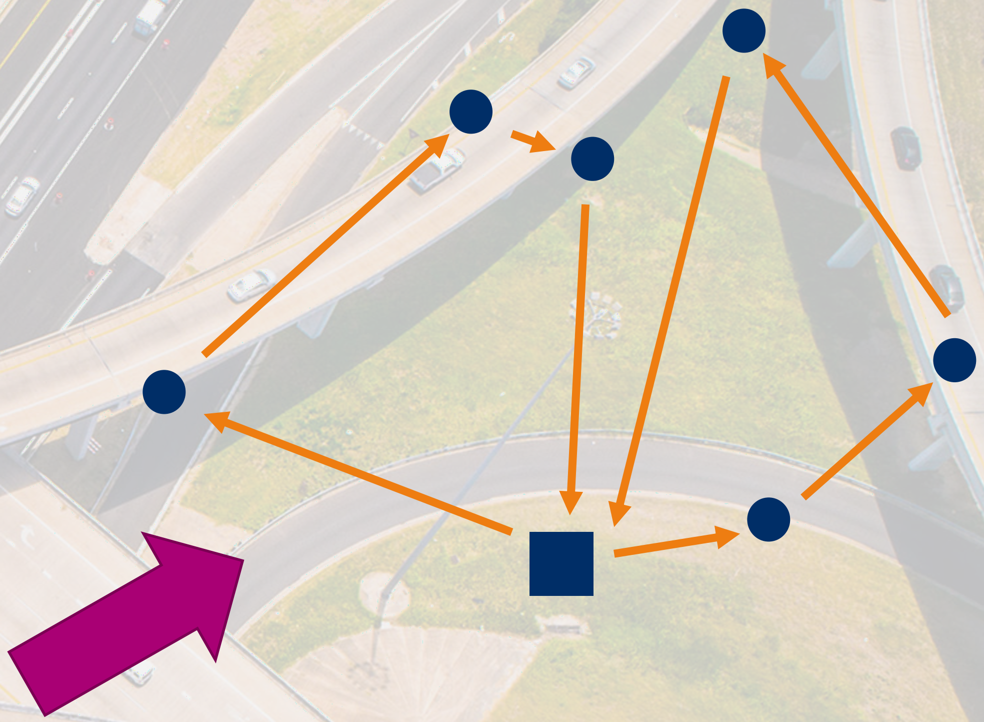
ORTEC

Local search

- SWAP, RELOCATE, 2-OPT, 2-OPT*
- Moves between near neighbors
- Smart ‘pre-checks’
- SWAP*, see next slide

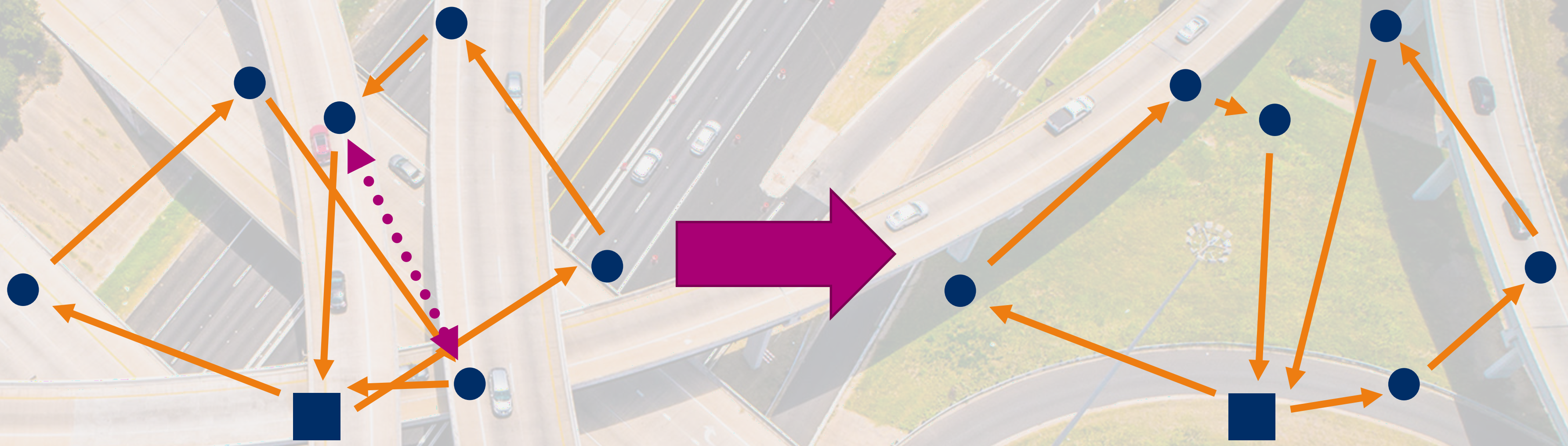
SWAP*





Exchange two nodes, insert at best position in other route

SWAP*



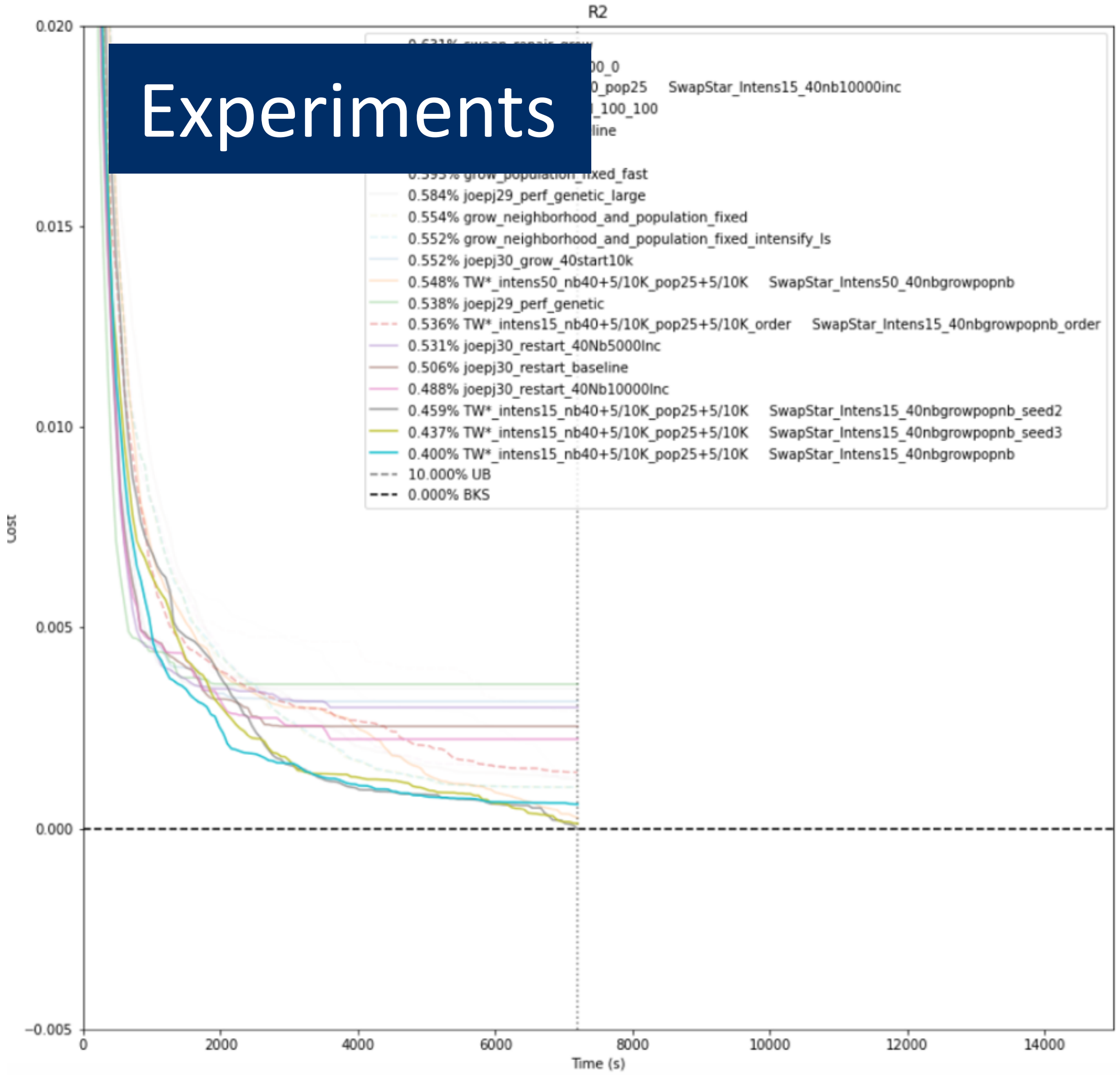
- Cache top 3 insertion positions
- Exact for CVRP
- Approximate for VRPTW

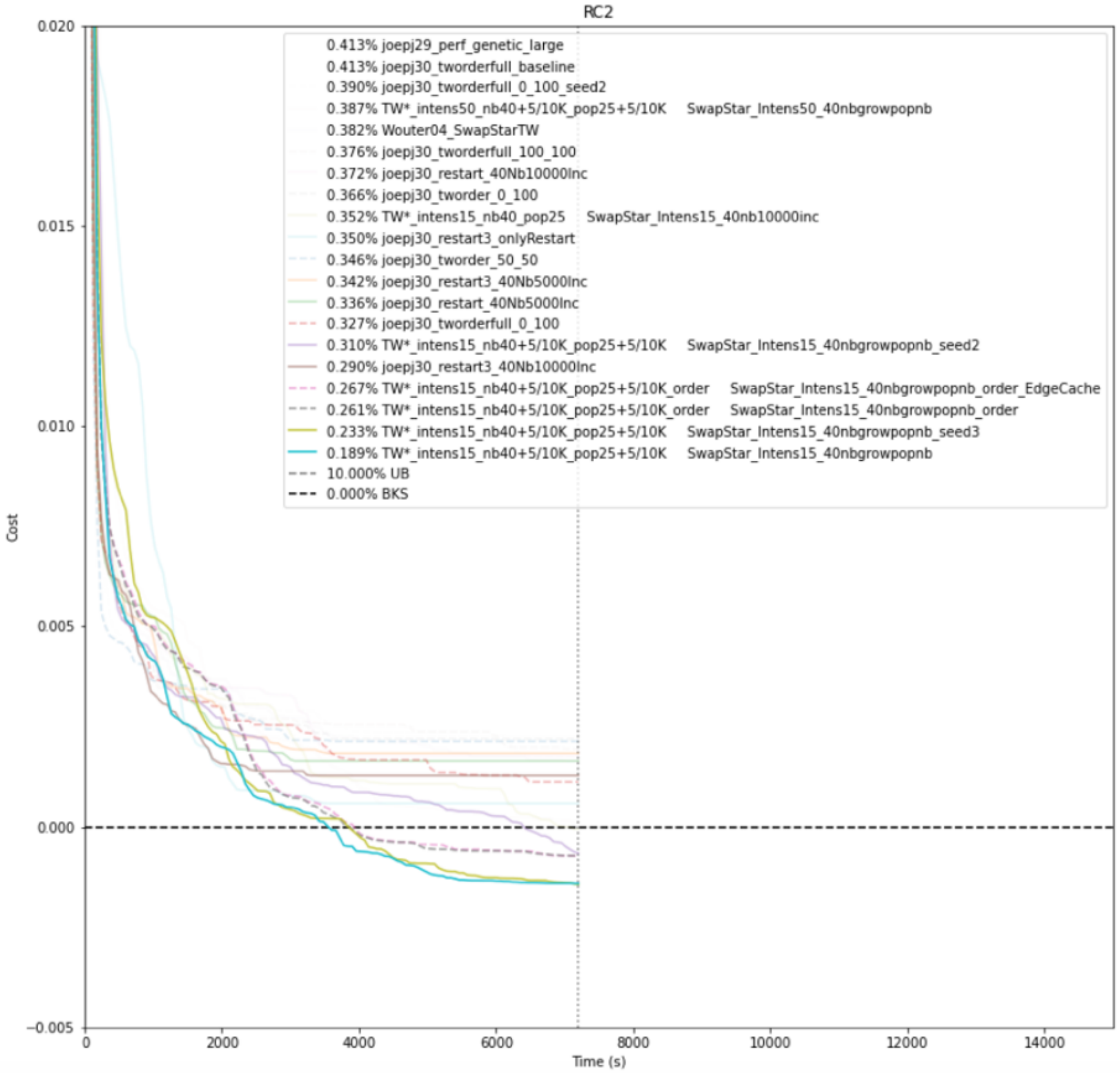
Growing the neighborhood & population

- Every 10K iterations
  - Grow neighborhood by 5
  - Grow population size by 5

- *Slightly different schedule for different instances

Experiments





Results

Dataset	C1	C2	R1	R2	RC1	RC2	Mean
Solomon	0,000%	0,000%	-0,003%	[illegible]	[illegible]	[illegible]	0,000%
GH200	0,000%	0,004%	[illegible]	[illegible]	0,016%	0,026%	0,009%
GH400	0,000%	0,000%	[illegible]	[illegible]	0,030%	-0,050%	-0,010%
GH600	-0,014%	[illegible]	[illegible]	-0,022%	-0,012%	-0,123%	-0,017%
GH800	0,030%	[illegible]	[illegible]	0,090%	0,112%	-0,222%	0,023%
GH1000	0,123%	-0,013%	[illegible]	-0,090%	0,094%	-0,158%	0,022%
Mean	[illegible]	0,001%	0,060%	0,002%	0,030%	-0,088%	0,004%

(a) Gap to reference solution

0,004%


OPTIMIZED BY ORTEC
ORTEC.COM