MUNI
FACULTY
OF INFORMATICS

Elegance of Simplicity: Designing Metaheuristics for Real-World Applications

Hana Rudová

Faculty of Informatics, Masaryk University
Brno, Czech Republic

PATAT 2026

Application domains

1. Course Timetabling
2. Vehicle Routing
3. Cross-Domain Search
4. Warehouse Planning

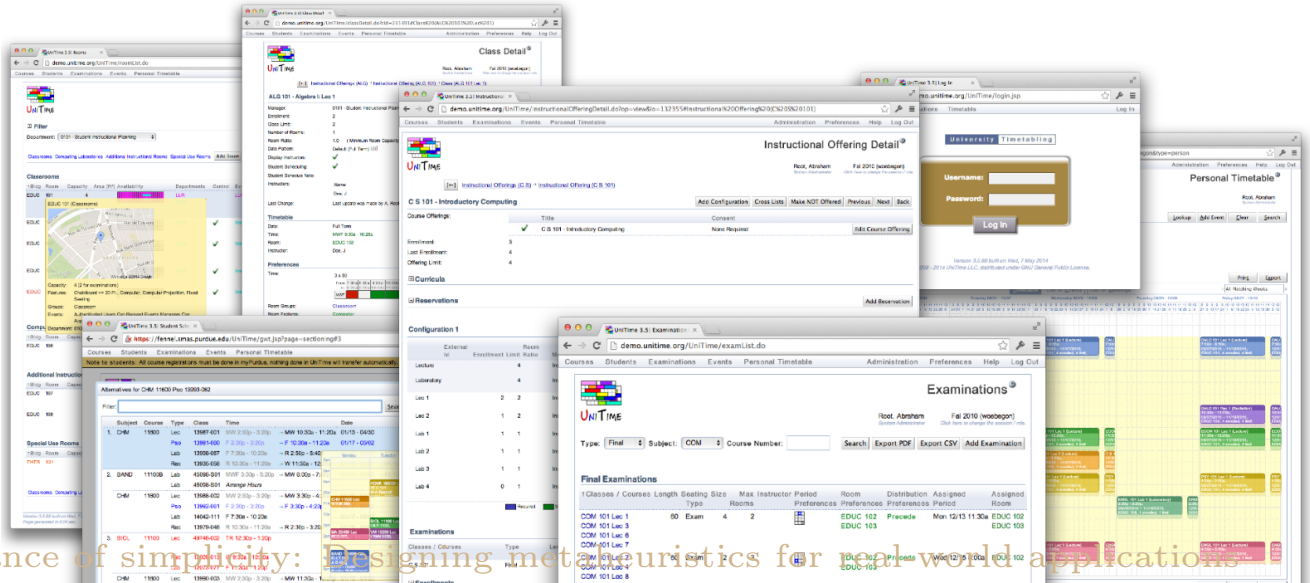
Course Timetabling

University course timetabling

- Curriculum-based timetabling
  - curriculum: set of courses
  - each student is enrolled in (one or more) curricula
  - goal: timetabling all courses in a curriculum without overlapping
  - also typical for timetabling in high schools
- Enrollment-based timetabling
  - each student is individually pre-enrolled/enrolled in a set of courses
  - student conflict:
    student is unable to take (two) enrolled courses due to overlap
  - goal: finding a timetable minimizing the number of student conflicts

University course timetabling in UniTime

- Curriculum-based timetabling
  - curriculum: set of courses
  - each student is enrolled in (one or more) curricula
  - goal: timetabling all courses in a curriculum without overlapping
  - also typical for timetabling in high schools
- Enrollment-based timetabling
  - each student is individually pre-enrolled/enrolled in a set of courses
  - student conflict:
    student is unable to take (two) enrolled courses due to overlap
  - goal: finding a timetable minimizing the number of student conflicts



Developed at Purdue University (USA) in collaboration with FI MU and MFF UK

Comprehensive system for university timetabling

- timetabling of courses with curricula and enrollments
- timetabling for students, exams, teachers
- open source, web-based, in Java
- developed since 2001
- in apereo foundation from 2015
  - for open-source projects in higher education

Applications worldwide

- voluntary registrations: over 600 from 113 countries
- in production: over 100 institutions from 58 countries

Model in UniTime

Optimization problem with hard and soft constraints

- domain for each variable
  - course timetabling: encodes combination of eligible times & classrooms
- hard constraints: must be satisfied, e.g.,
  - time & classroom requirements
  - course structure
  - non-overlapping for classrooms, teachers, and other resources
- soft constraints: represent objective functions,
  whose weighted sum is minimized, e.g.,
  - the number of student conflicts
  - preferences on time & classroom

Iterative forward search: Algorithm

Constructive non-systematic algorithm

- works with partial solutions
  - empty $\longrightarrow$ partial $\longrightarrow$ complete solution
- works with feasible/consistent solutions
  - hard constraints must be satisfied in any partial solution

Iterative forward search: Algorithm

Constructive non-systematic algorithm

- works with partial solutions
  - empty $\longrightarrow$ partial $\longrightarrow$ complete solution
- works with feasible/consistent solutions
  - hard constraints must be satisfied in any partial solution

Basic idea: mimics manual timetable construction

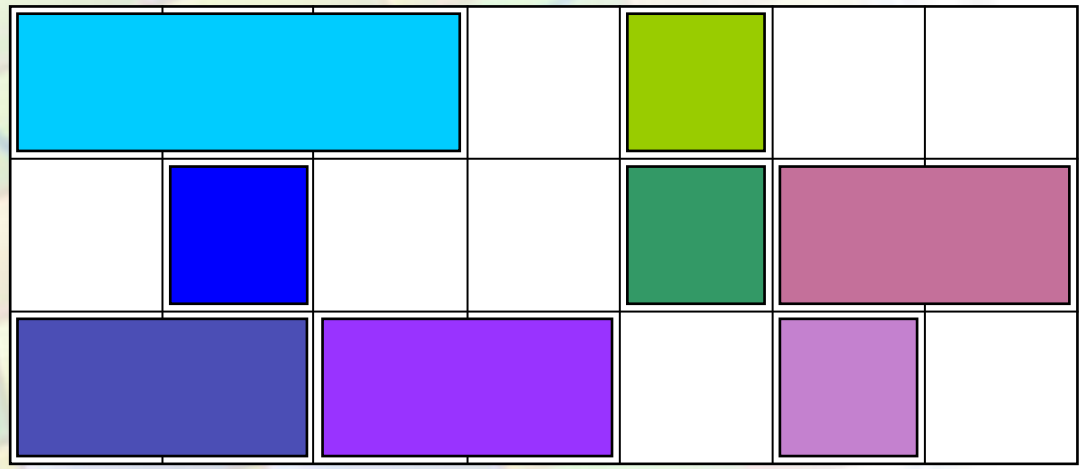
- starts with an empty solution
- selects a new variable to assign
- if it finds a conflict,
  unassigns all variables conflicting with the selected variable

Conflict-based statistics

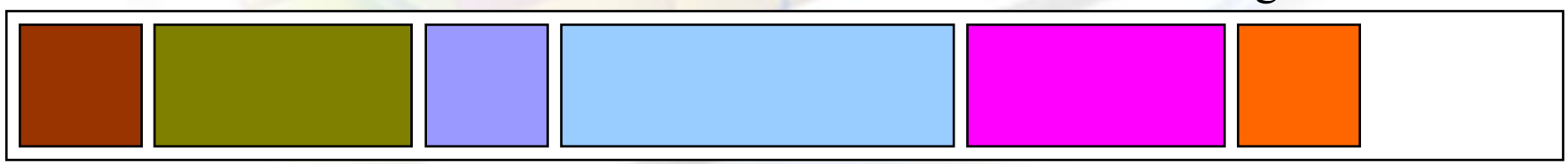
- simple learning process to avoid cycling

Iterative Forward Search Algorithm

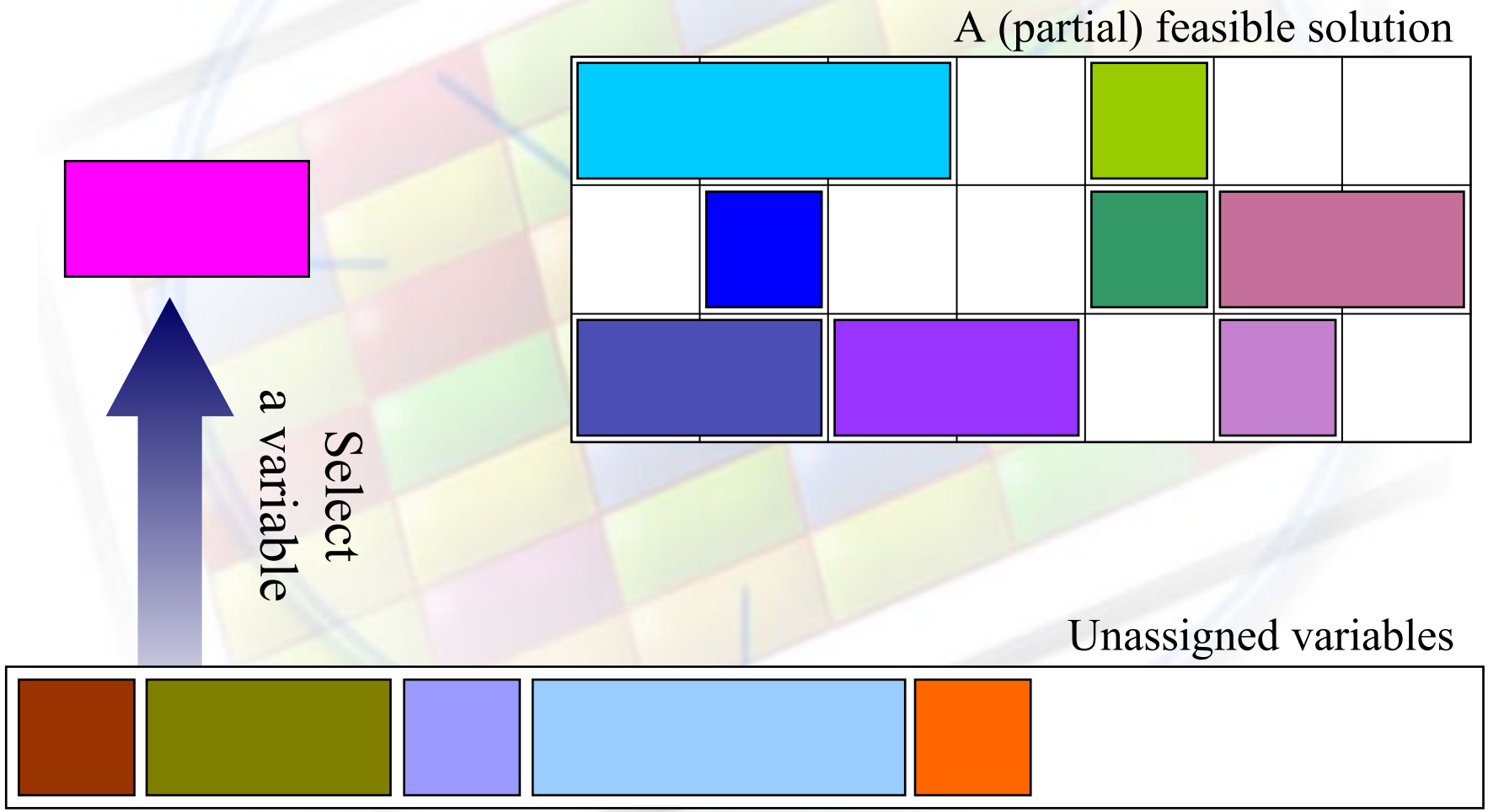
A (partial) feasible solution



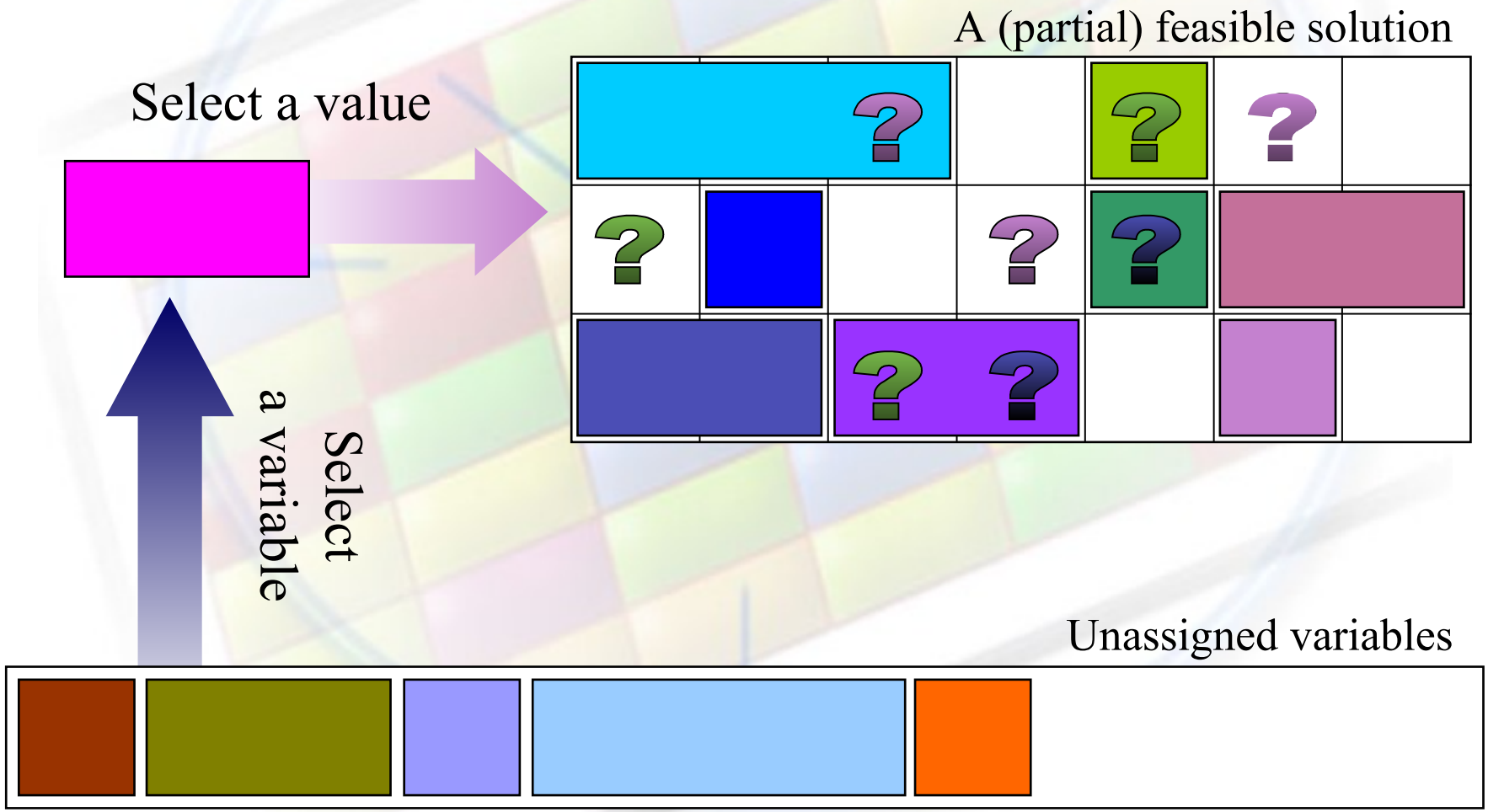
Unassigned variables



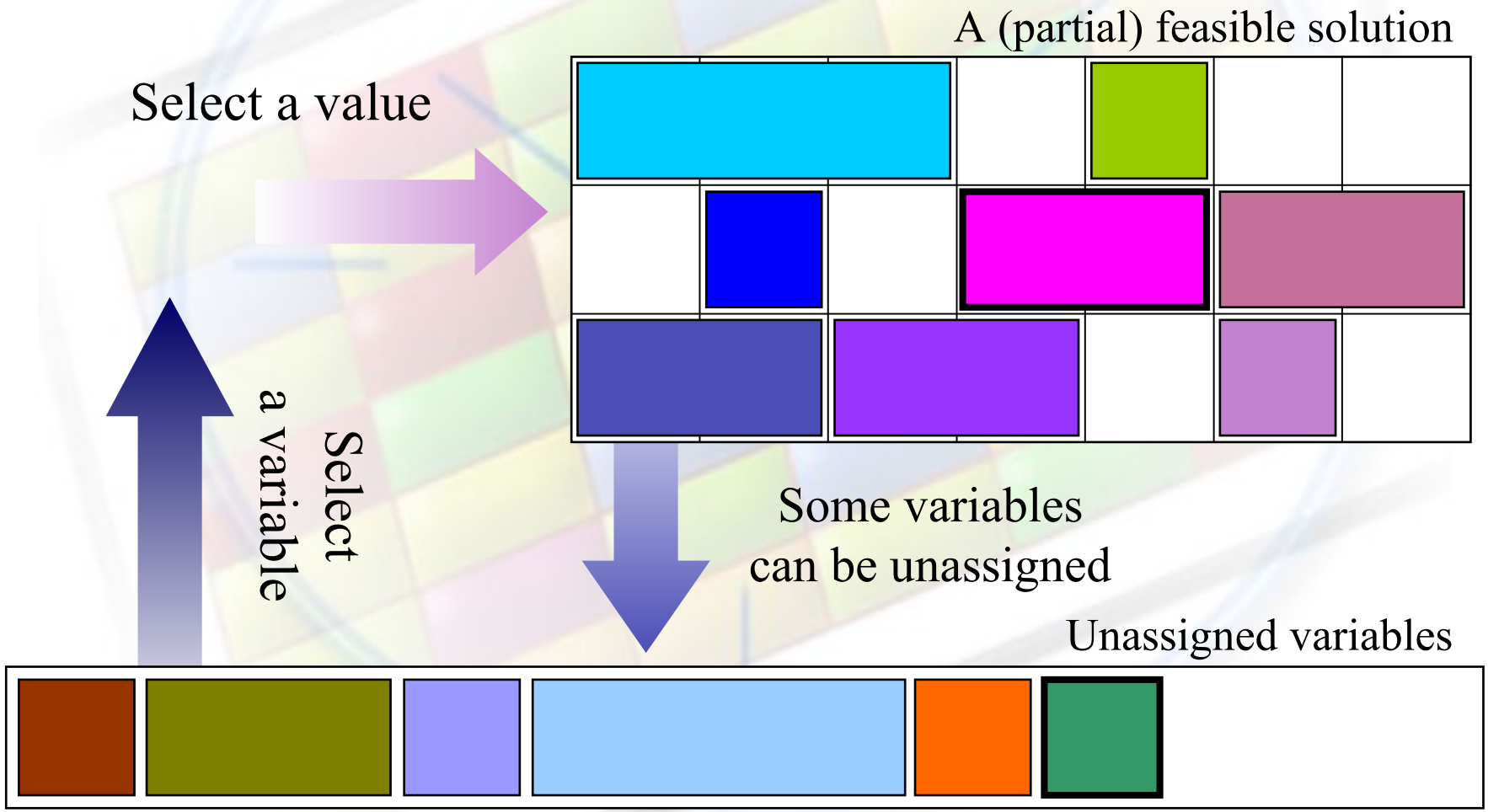
Iterative Forward Search Algorithm



Iterative Forward Search Algorithm



Iterative Forward Search Algorithm



Conflict-based statistics (CBS): Example

Crucial: how to select value for each variable?

A partial timetable

Room/Time	8:00	9:00	10:00
100	Geography (G) *James*	Biology (B) *Peter*	
200	Physics (P) *Thomas*	English (E) *David*	Chemistry (Ch) *John*

Requirements for Math (M) with assigned instructor *Peter*

- must be in the classroom 200
- must be scheduled at 8:00 or 9:00

Conflict-based statistics (CBS): Simple learning

Room/Time	8:00	9:00	10:00
100	Geography (G) *James*	Biology (B) *Peter*	
200	Physics (P) *Thomas*	English (E) *David*	Chemistry (Ch) *John*

Data stored in CBS

The CBS remembers how many times an assignment caused another variable to be unassigned in past iterations

Conflict-based statistics (CBS): Simple learning

Room/Time	8:00	9:00	10:00
100	Geography (G) *James*	Biology (B) *Peter*	
200	Physics (P) *Thomas*	English (E) *David*	Chemistry (Ch) *John*

Data stored in CBS

The CBS remembers how many times an assignment caused another variable to be unassigned in past iterations:

- Placing Math at 08:00 caused Physics to be unassigned from 200 5 times in the past: $M_{200,8:00} \Rightarrow 5 \times \neg P_{200,8:00}$

Conflict-based statistics (CBS): Simple learning

Room/Time	8:00	9:00	10:00
100	Geography (G) *James*	Biology (B) *Peter*	
200	Physics (P) *Thomas*	English (E) *David*	Chemistry (Ch) *John*

Data stored in CBS

The CBS remembers how many times an assignment caused another variable to be unassigned in past iterations:

- Placing Math at 08:00 caused Physics to be unassigned from 200 5 times in the past: $M_{200,8:00} \Rightarrow 5 \times \neg P_{200,8:00}$
- Placing Math at 09:00 caused English to be unassigned from 200 4 times in the past: $M_{200,9:00} \Rightarrow 4 \times \neg E_{200,9:00}$

Conflict-based statistics (CBS): Simple learning

Room/Time	8:00	9:00	10:00
100	Geography (G) *James*	Biology (B) *Peter*	
200	Physics (P) *Thomas*	English (E) *David*	Chemistry (Ch) *John*

Data stored in CBS

The CBS remembers how many times an assignment caused another variable to be unassigned in past iterations:

- Placing Math at 08:00 caused Physics to be unassigned from 200 5 times in the past: $M_{200,8:00} \Rightarrow 5 \times \neg P_{200,8:00}$
- Placing Math at 09:00 caused English to be unassigned from 200 4 times in the past: $M_{200,9:00} \Rightarrow 4 \times \neg E_{200,9:00}$

  Placing Math at 09:00 caused Biology to be unassigned from 100 2 times in the past: $M_{200,9:00} \Rightarrow 2 \times \neg B_{100,9:00}$

Value selection using CBS

Value selection heuristic:

- select the value with the lowest number of conflicts weighted by their past frequency

Options:

- Math at 8:00: $M_{200,8:00} \Rightarrow 5 \times \neg P_{200,8:00}$
  $\Rightarrow$ P is unassigned, CBS: 5

Value selection using CBS

Value selection heuristic:

- select the value with the lowest number of conflicts weighted by their past frequency

Options:

- Math at 8:00: $M_{200,8:00} \Rightarrow 5 \times \neg P_{200,8:00}$
  $\Rightarrow$ P is unassigned, CBS: 5
- Math at 9:00: $M_{200,9:00} \Rightarrow 4 \times \neg E_{200,9:00}$
  $M_{200,9:00} \Rightarrow 2 \times \neg B_{100,9:00}$
  (B and M taught by *Peter*, i.e., $B_{100,9:00}$ conflicts with $M_{200,9:00}$)
  $\Rightarrow$ both E and B are unassigned, CBS: $4 + 2$

Value selection using CBS

Value selection heuristic:

- select the value with the lowest number of conflicts weighted by their past frequency

Options:

- Math at 8:00: $M_{200,8:00} \Rightarrow 5 \times \neg P_{200,8:00}$
  $\Rightarrow$ P is unassigned, CBS: 5
- Math at 9:00: $M_{200,9:00} \Rightarrow 4 \times \neg E_{200,9:00}$
  $M_{200,9:00} \Rightarrow 2 \times \neg B_{100,9:00}$
  (B and M taught by *Peter*, i.e., $B_{100,9:00}$ conflicts with $M_{200,9:00}$)
  $\Rightarrow$ both E and B are unassigned, CBS: $4 + 2$

Using CBS:

- IFS chooses 8:00 for Math because $5 < 4 + 2$.
- Math is assigned to 8:00
- Physics is unassigned and will be resolved in the next iteration

Value selection using CBS

Value selection heuristic:

- select the value with the lowest number of conflicts weighted by their past frequency

Options:

- Math at 8:00: $M_{200,8:00} \Rightarrow 5 \times \neg P_{200,8:00}$
  $\Rightarrow$ P is unassigned, CBS: 5
- Math at 9:00: $M_{200,9:00} \Rightarrow 4 \times \neg E_{200,9:00}$
  $M_{200,9:00} \Rightarrow 2 \times \neg B_{100,9:00}$
  (B and M taught by *Peter*, i.e., $B_{100,9:00}$ conflicts with $M_{200,9:00}$)
  $\Rightarrow$ both E and B are unassigned, CBS: $4 + 2$

Using CBS:

- IFS chooses 8:00 for Math because $5 < 4 + 2$.
- Math is assigned to 8:00
- Physics is unassigned and will be resolved in the next iteration
- CBS counter $M_{200,8:00} \Rightarrow \neg P_{200,8:00}$ incremented from 5 to 6

When do we count a conflict from CBS?

We only count a conflict if it leads to the removal of the assignment

Math has assigned instructor *Peter* $\Rightarrow$ conflicts with Biology

Room/Time	8:00	9:00	10:00
100	Geography (G) *James*	Biology (B) *Peter*	
200	Physics (P) *Thomas*	English (E) *David*	Chemistry (Ch) *John*

CBS

- $M_{200,8:00} \Rightarrow 5 \times \neg P_{200,8:00} \Rightarrow P$ is unassigned $5 < 4 + 2$
- $M_{200,9:00} \Rightarrow 4 \times \neg E_{200,9:00}$

  $M_{200,9:00} \Rightarrow 2 \times \neg B_{100,9:00}$

When do we count a conflict from CBS?

We only count a conflict if it leads to the removal of the assignment

Math has assigned instructor *Peter* $\Rightarrow$ no conflict with Biology

Room/Time	8:00	9:00	10:00
100	Geography (G) *James*	Biology (B) *John*	
200	Physics (P) *Thomas*	English (E) *David*	Chemistry (Ch) *John*

CBS

- $M_{200,8:00} \Rightarrow 5 \times \neg P_{200,8:00}$
- $M_{200,9:00} \Rightarrow 4 \times \neg E_{200,9:00} \Rightarrow E$ is unassigned $4 < 5$

  $M_{200,9:00} \Rightarrow 2 \times \neg B_{100,9:00}$: not counted now

- IFS with CBS used for course timetabling in UniTime
  to construct a reasonable initial complete solution
  - no parameter tuning
- Initial solution then improved with
  great deluge or simulated annealing with reheating

⇒ Remarkably simple approach which works very well!

- Presentation: today 15:40
  T. Müller, Instructor Scheduling in UniTime
- Overview paper about course timetabling in UniTime
  H. Rudová, T. Müller, K. Murray, Complex university course timetabling. *Journal of Scheduling*, 14(2): 187–207, 2011

International Timetabling Competition 2019

- 10 institutions from 5 continents
  have provided data
  for the ITC 2019 competition

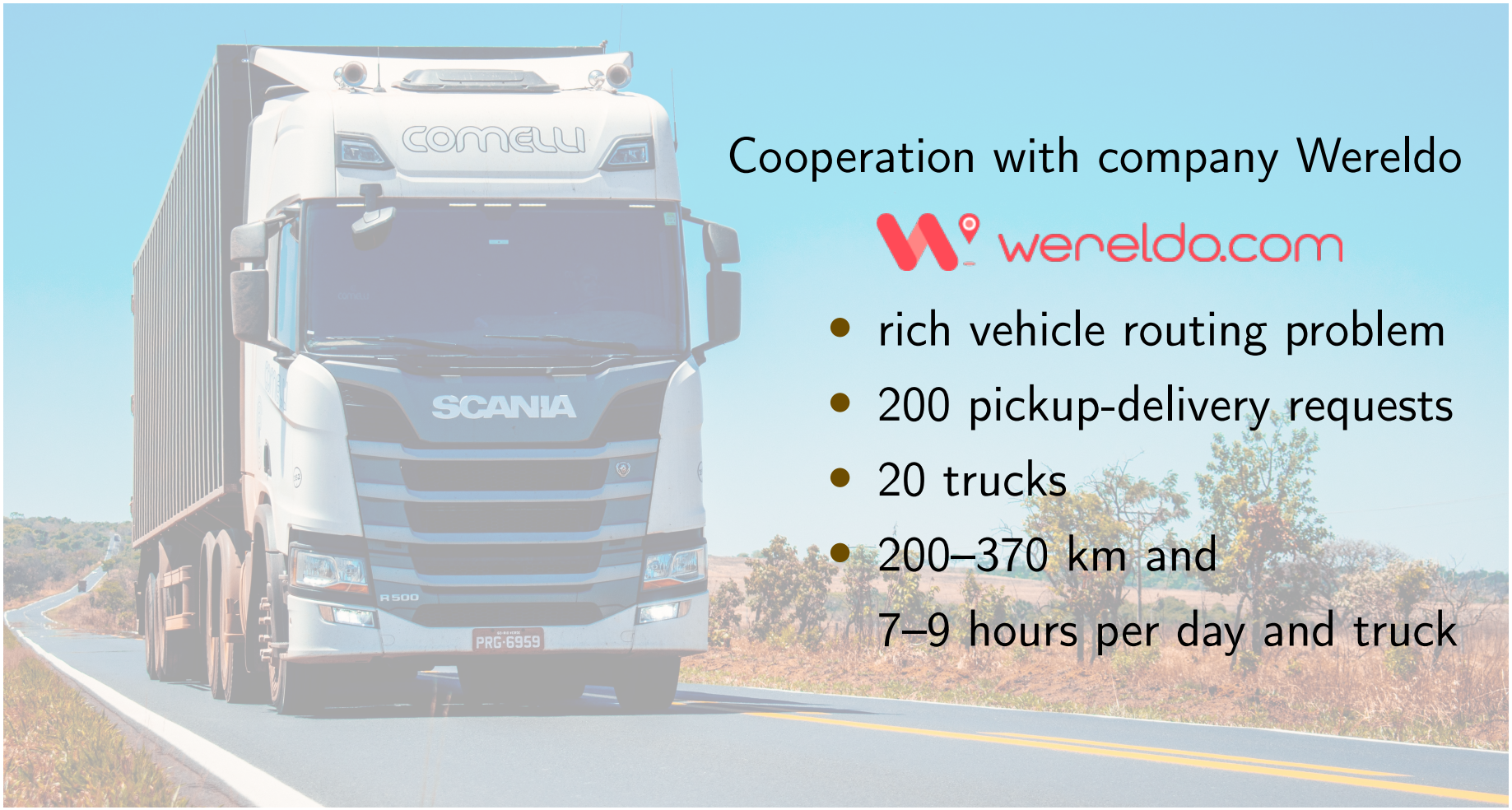
ITC 2019

- 30+5 real-world data instances available using UniTime
  - different size, different characteristics
  - XML format
  - solution validator
- Enrollment-based timetabling
  minimize the weighted sum of
  - time and room assignment penalties
  - penalties of broken soft distribution constraints
  - the total number of student conflicts
- You can compare your results with competitors!
  - about 800 registrations from 80 countries

- https://www.itc2019.org
- *T. Müller, H. Rudová, Z. Müllerová, Real-world university course timetabling at the International Timetabling Competition 2019, Journal of Scheduling, 28(2): 247–267, 2025.*

Vehicle Routing

Freight Transportation Problem



Cooperation with company Wereldo

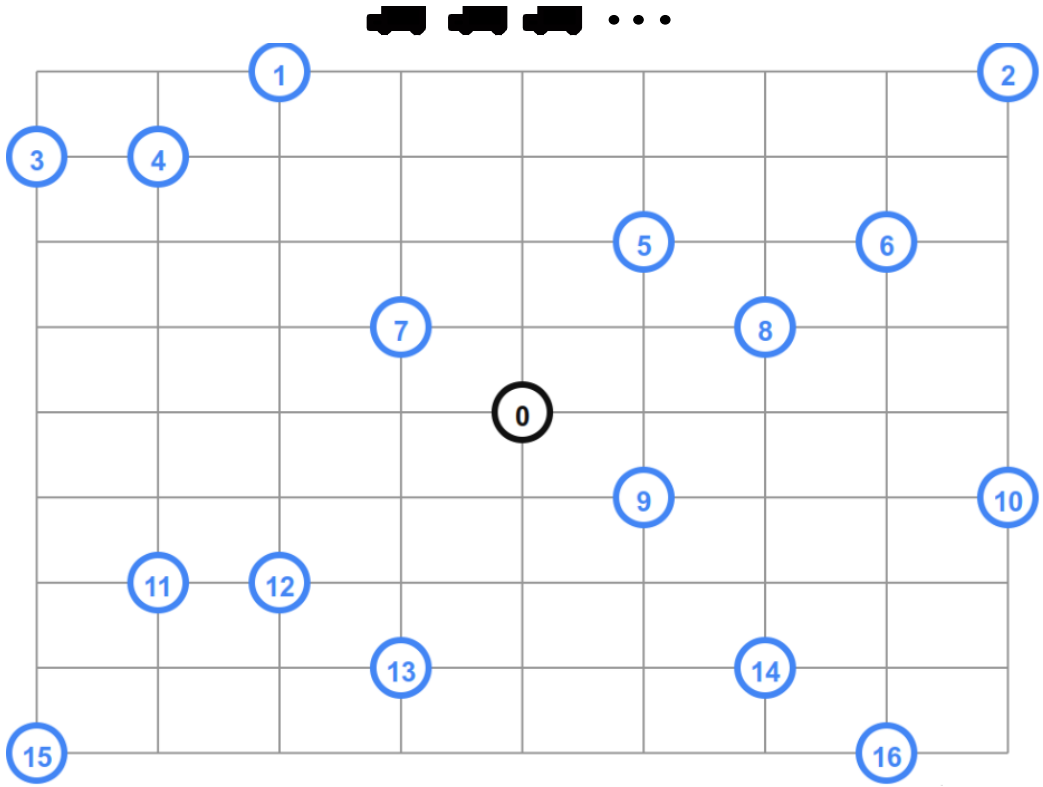
wereldo.com

- rich vehicle routing problem
- 200 pickup-delivery requests
- 20 trucks
- 200–370 km and
  7–9 hours per day and truck

- *V. Sassmann, H. Rudová, M. Gabonay and V. Sobotka (2023). Real-world vehicle routing using adaptive large neighborhood search. In EvoCop 2023, LNCS 13987, p. 34–49.*
- *V. Sobotka and H. Rudová, Uncertainty in Real-World Vehicle Routing. In ECAI 2024, p. 4311–4318.*

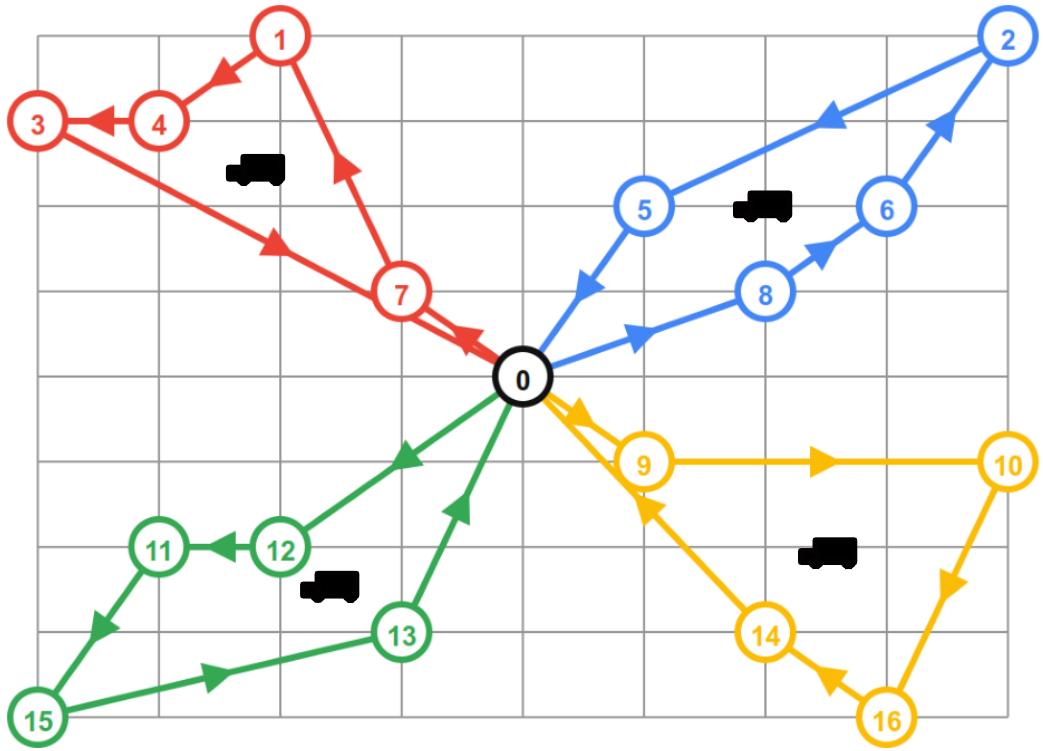
Vehicle routing problem (VRP)

- Requests
- Depot
- Vehicles



Vehicle routing problem (VRP)

- Requests
- Depot
- Vehicles
- Routes
  - serve all requests
  - minimize distance



Rich vehicle routing problem

Requests

- time windows
- joint service time
- weights & volumes
- pickup & deliveries
- request-vehicle compatibility

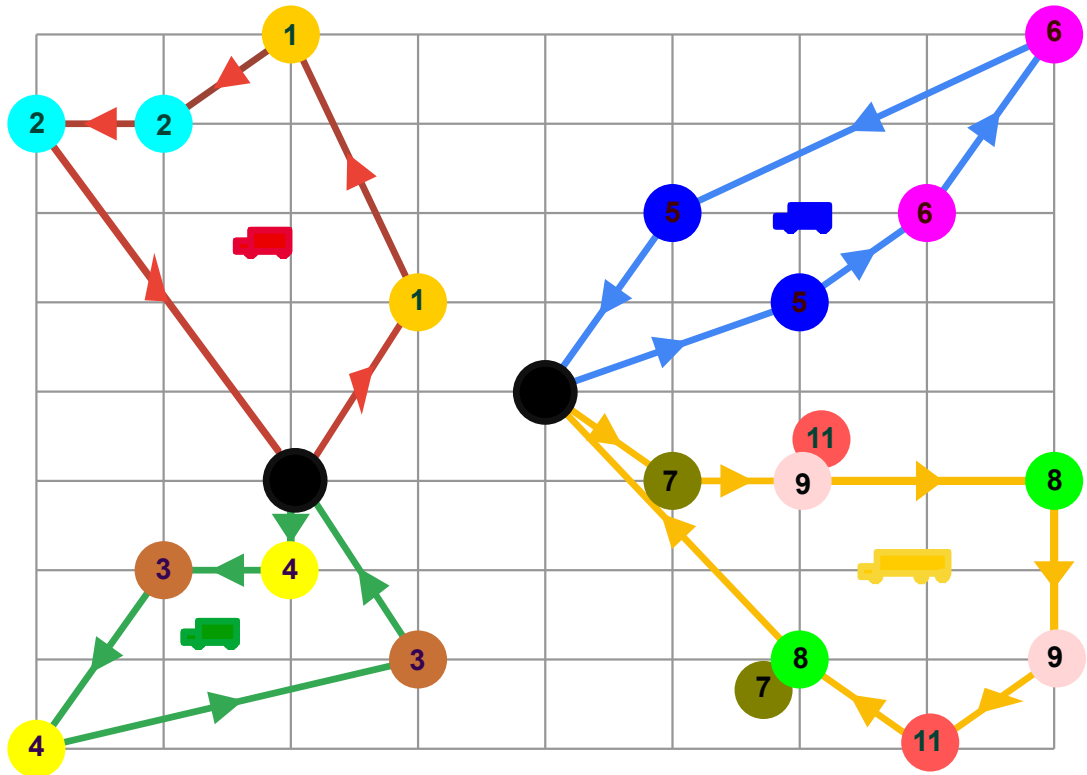
Vehicles

- multiple depots
- heterogeneous fleet

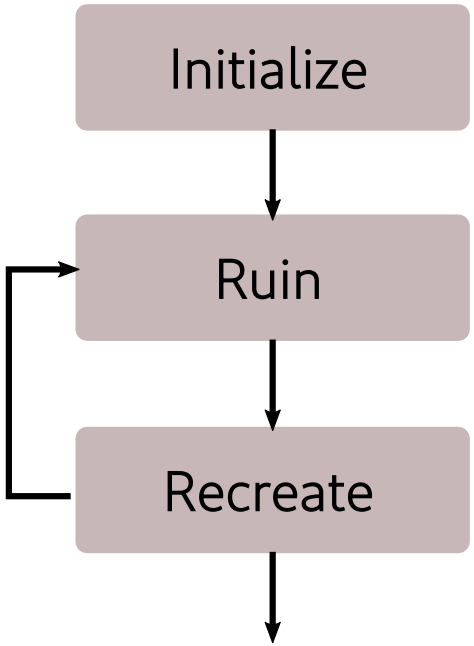
Routes

- route duration limit
- route length limit

...



Large neighborhood search (LNS)



1. Find an initial fesible solution

2. Select a part of the solution &
   Unassign its variables

3. Reoptimize the unassigned part &
   Keep the rest fixed

Ruin & recreate vs. destroy & repair

- *P. Shaw (1998). Using constraint programming and local search methods to solve vehicle routing problems. In Principles and Practice of Constraint Programming – CP98, 417–431. LNCS 1520, Springer.*

Ruin & recreate heuristics

Ruin heuristics

- random removal
- removal of worst requests based on cost
- Shaw removal: remove similar requests
  - idea: they can be easily exchanged together (in following recreate)

Ruin & recreate heuristics

Ruin heuristics

- random removal
- removal of worst requests based on cost
- Shaw removal: remove similar requests
  - idea: they can be easily exchanged together (in following recreate)

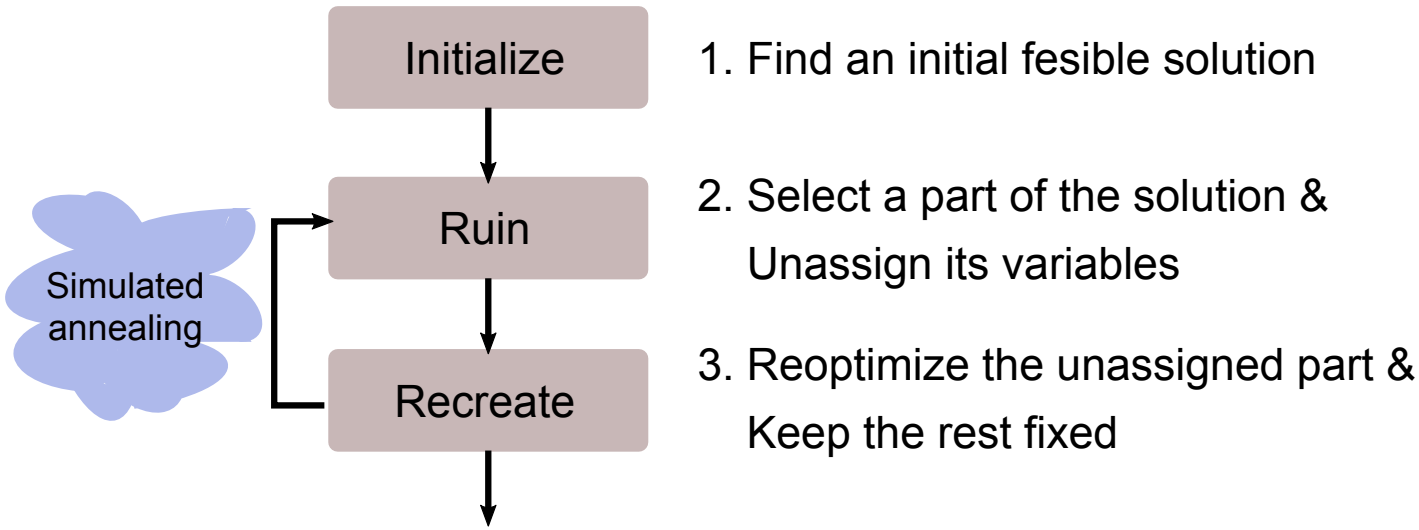
Recreate heuristics:

recompute costs each time

- greedy insertion heuristic
- regret insertion heuristic
  - it picks the request with the largest cost loss between its best and second best place
  - it then assigns that request to its best place to avoid regretting not having done that

Slack induction by string removals (SISRs)

- SISRs: large neighborhood search heuristic solver

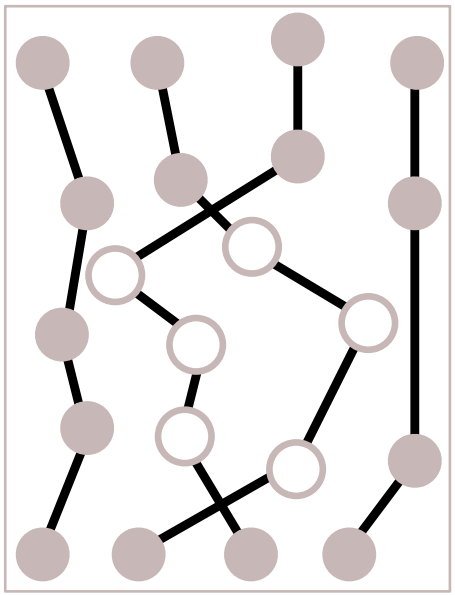


- Algorithmic simplicity and versatility w.r.t. problem variants
  - common algorithmic core
  - minimal adaptation to specific VRP variants

- *J. Christiaens and G. V. Berghe. Slack induction by string removals for vehicle routing problems. Transportation Science, 54:417–433, 2020.*

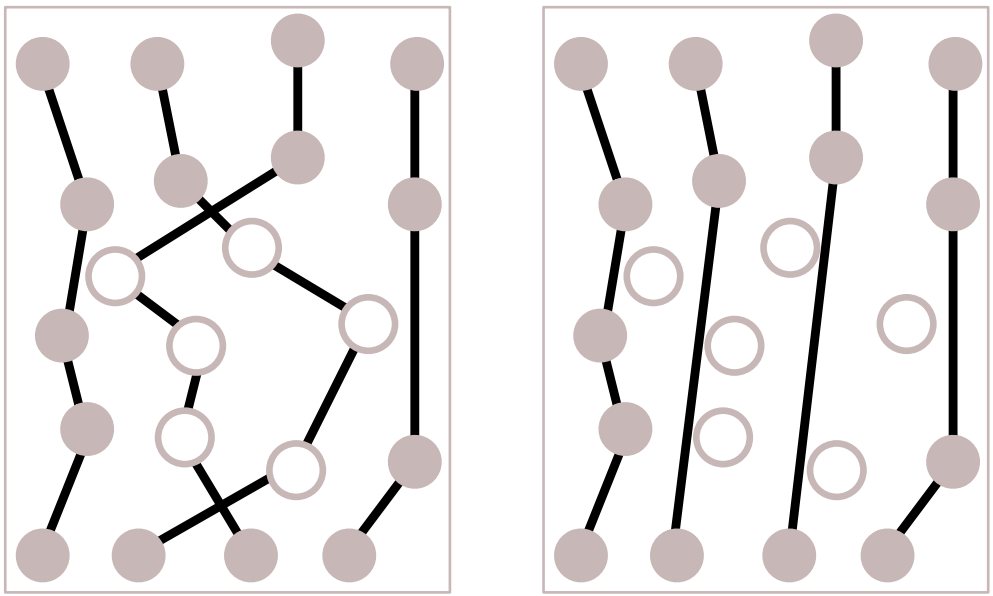
SISRs: Ruin phase

- Core idea: introduce capacity & spatial slacks
- Remove continuous strings of customer requests within routes
- Strings from two routes should be close to each other, i.e., adjacent



SISRs: Ruin phase

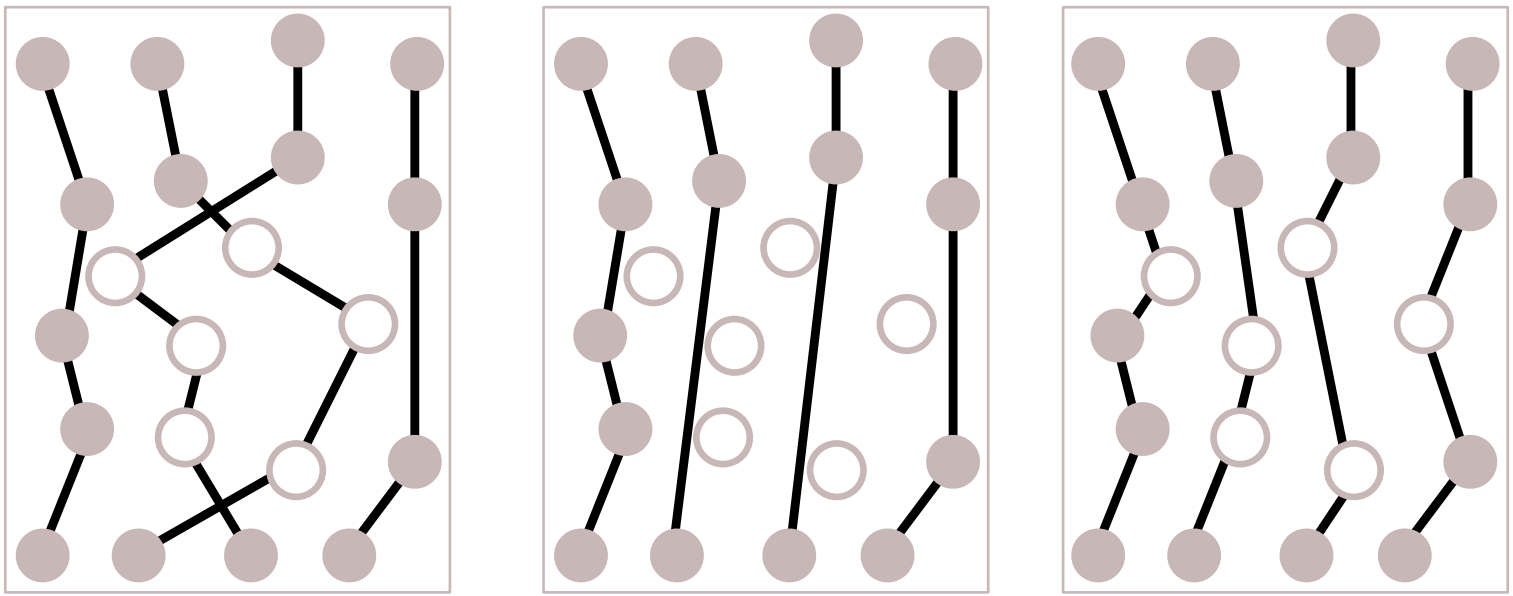
- Core idea: introduce capacity & spatial slacks
- Remove continuous strings of customer requests within routes
- Strings from two routes should be close to each other, i.e., adjacent



- Ruin phase $\Rightarrow$ a set of absent customer requests A

SISRs: Ruin phase

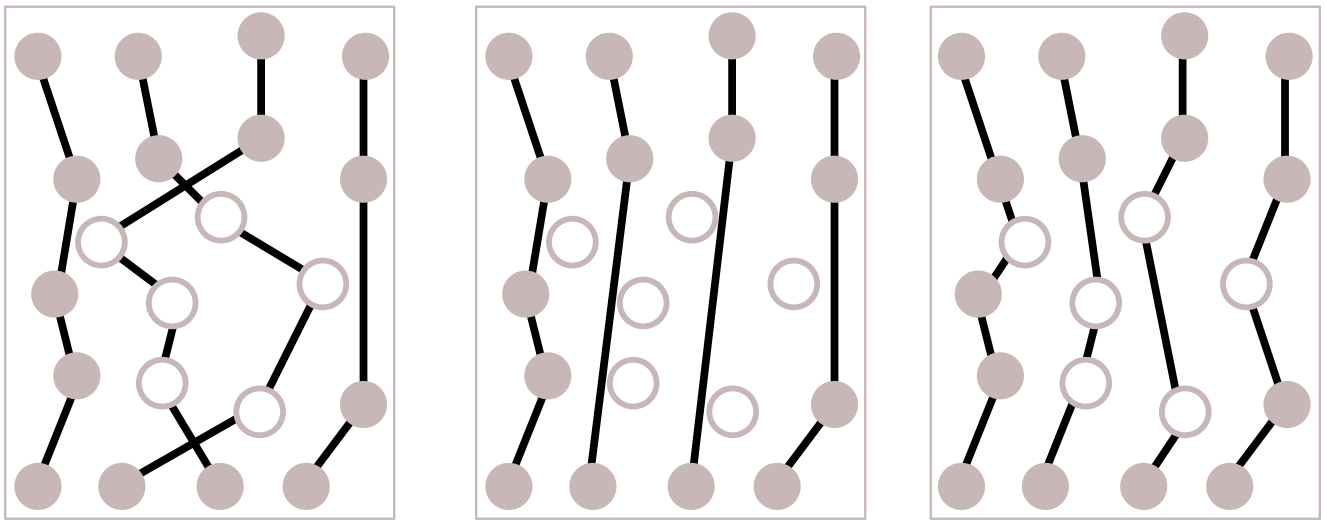
- Core idea: introduce capacity & spatial slacks
- Remove continuous strings of customer requests within routes
- Strings from two routes should be close to each other, i.e., adjacent



- Ruin phase $\Rightarrow$ a set of absent customer requests A

SISRs: Recreate phase by sorting

- Examples of sorting criteria
  - customer demand, descending
  - time window width, ascending
  - simple random shuffling
- Randomly select one sorting criterion & sort absent requests A
  - sorting criteria weights
- Reinsert customer requests from sorted A one-by-one
  - cheapest insertion based on distance, blinks

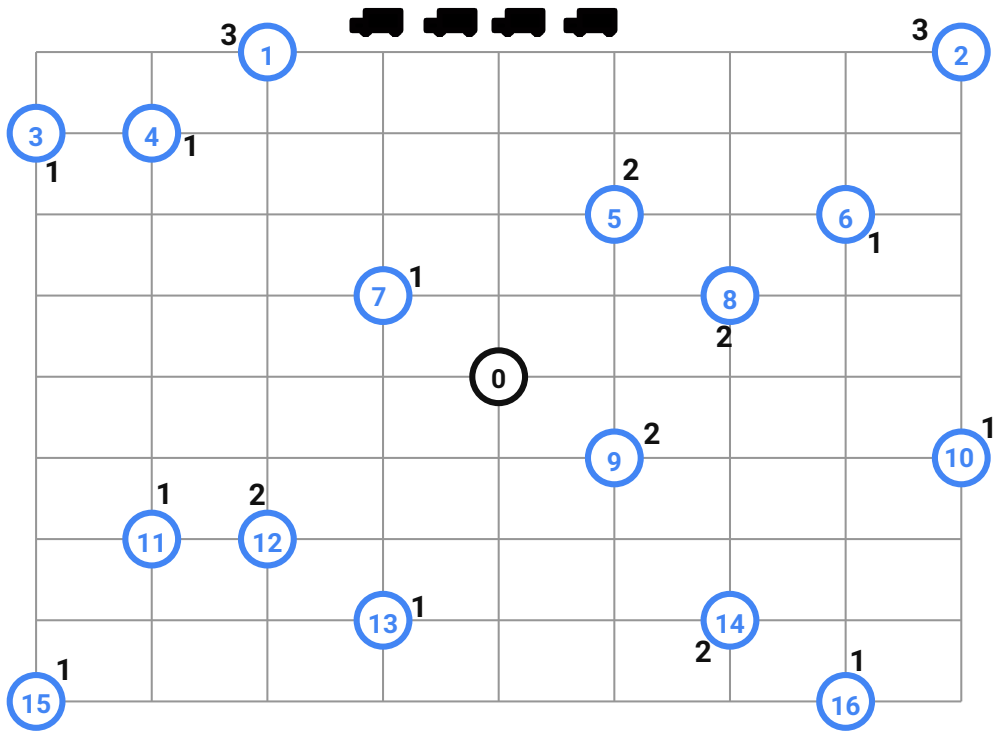


SISRs on top of LNS

- SISRs ruin: strings & slack
- LNS recreate
  - each request placed in a position with the smallest increase in cost
  - recompute cost for each customer request
- SISRs recreate: sorting
  - compute insertion costs at the beginning
  - reinsert sorted customer requests one-by-one
  - sorting adjusted for the problem
    - based on capacity, time windows, score of customers
- Simplified process sufficient!
  - smaller complexity

Team orienteering problem (TOP)

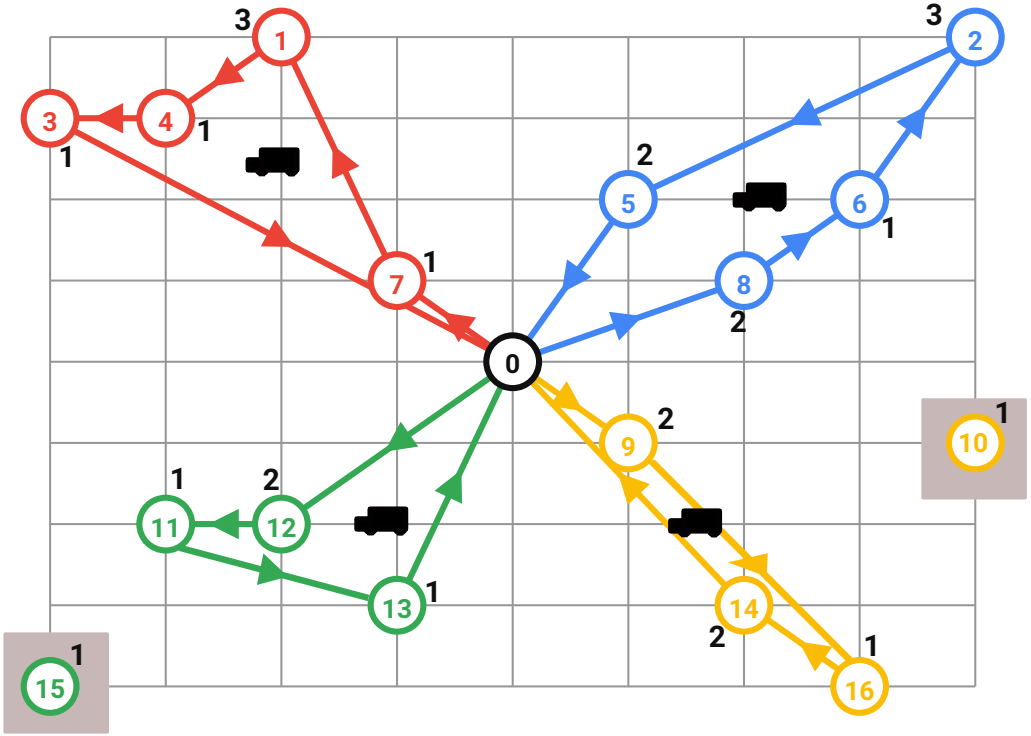
- Requests
- Score of request
- Depot
- Limited number of vehicles
- Routes
  - minimize distance
  - maximize the total score of served customers



- Various problem variants
  - base TOP, capacities CTOP, time windows TOPTW, ...

Team orienteering problem (TOP)

- Requests
- Score of request
- Depot
- Limited number of vehicles
- Routes
  - minimize distance
  - maximize the total score of served customers
  - route length limit



- Various problem variants
  - base TOP, capacities CTOP, time windows TOPTW, ...

SISRs for TOPs

- Our extension of SISRs for TOPs
  - essential: new sorting methods for recreate
- Experimental comparison with state-of-the-art methods
  - for TOP, TOPTW, CTOP
  - standard benchmarking datasets and experimental setups
- TOP & TOPTW: performance on par with state-of-the-art methods
- CTOP: benchmarks dominated by our SISRs adaptation
- Overall: found $\sim$ 85 % of all best-known solutions
- SoTA approach: HALNS for TOP & CTOP
  - combines ALNS with exact integer programming to solve sub-route optimization and set packing problems

---

- *M. Pajerský, V. Sobotka, and H. Rudová. Open-Source Implementation of Slack Induction by String Removals for Routing and Orienteering Problems. In CPAIOR 2026, p. 359–377.*

Open-source implementation of SISRs solvers

`https://github.com/hankarudova/open-source-sisr-routing`

- *M. Pajerský, V. Sobotka, and H. Rudová. Open-Source Implementation of Slack Induction by String Removals for Routing and Orienteering Problems. In CPAIOR 2026, p. 359–377.*

Cross-Domain Search

Hyper-heuristics

Main idea: automatically construct the search algorithm

Selection hyper-heuristics: adaptive selection of low level heuristics

Search operators ≈ low-level heuristics (LLHs)

- local search
  - neighborhood-based search moves
  - guarantee of non-worsening solutions
- perturbative heuristics
  - ruin & recreate
  - mutation
- crossover
  - recombination of two solutions

Online selection cross-domain hyper-heuristics

Characteristics

- online: the algorithm is constructed as it searches
- selection: adaptive selection of LLHs
- cross-domain: aim to perform well across diverse problem domains

Broad range of approaches & algorithms

- variable-neighborhood search, iterated local search, evolutionary approaches, Monte Carlo tree search, reinforcement learning, ...

- *Drake, J.H., A. Kheiri, E. Özcan, E.K. Burke (2020). Recent advances in selection hyper-heuristics, EJOR, 285(2):405–428.*
- *Razali, M. K. M., Ayob, M., Rahman, A. H. A., Jarmin, R., Liu, C. Y., Maaya, M., Izaham, A., and Kendall, G. (2025). Unveiling effective heuristic strategies: A review of cross-domain heuristic search challenge algorithms. CMES, 142(2):1233–1288.*
- *Burke, E. K., Gendreau, M., Hyde, M., Kendall, G., McCollum, B., Ochoa, G., Parkes, A. J., and Petrovic, S. (2011). The cross-domain heuristic search challenge—an international research competition. In LION, 631–634.*

How to select vs. what to select from

- Prevailing assumptions: the provided LLH set is fixed and used as is
- Key focus: how to select LLHs
  - e.g., involved reinforcement-learning-based selection

- Our focus: what LLHs to select from
  - idea: transform the LLH set and operate on the transformed LLH set
  - goal: control critical properties of LLHs and improve cross-domain performance

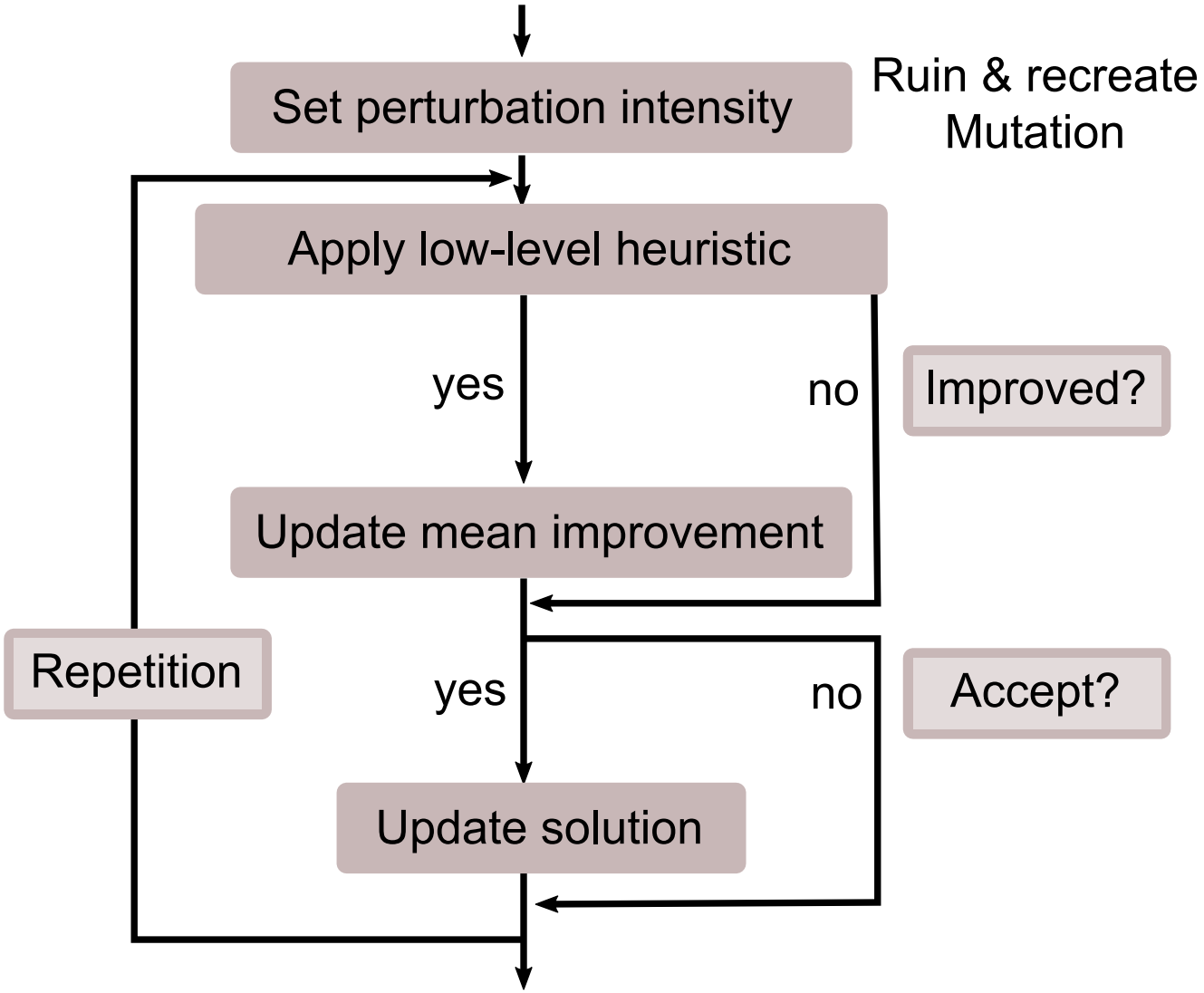
Key principles in cross-domain search

The search process should operate in a reasonably safe environment (1) with normalized granularity at which new LLHs are sampled (2), while allowing the perturbation intensity to vary (3).

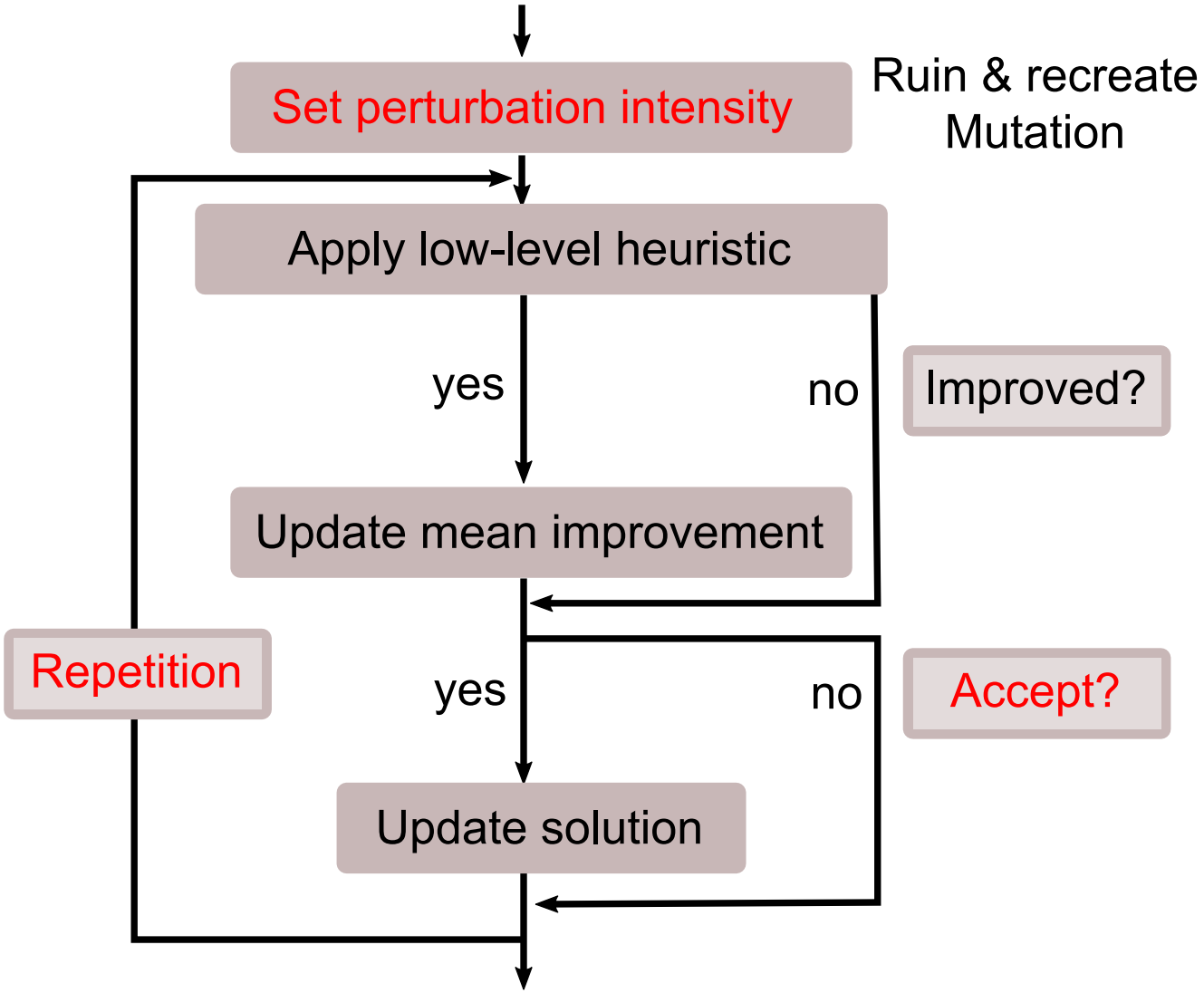
1. Solution acceptance
   - balance between intensification and diversification
2. Repetitions
   - fairness in time-allocation among LLHs
3. Perturbation intensity
   - balance between intensification and diversification

- *V. Sobotka, L. Kletzander, N. Musliu, and H. Rudová (2025). Key Principles in Cross-Domain Hyper-Heuristic Performance. Pre-print at arXiv.org.*

Hyper-heuristics



Hyper-heuristics: Key principles



Naive hyper-heuristic

Start with a naive hyper-heuristic

- uniformly random selection of an LLH category
  - local search, ruin & recreate, mutation, crossover
- and then uniformly random selection of an LLH within the category
- while always accepting worse solutions

Naive hyper-heuristic & LLH set transformation

Start with a naive hyper-heuristic

- uniformly random selection of an LLH category
  - local search, ruin & recreate, mutation, crossover
- and then uniformly random selection of an LLH within the category
- while always accepting worse solutions

Key principles

- varying perturbation intensity for each low-level heuristic
  - each perturbative LLH replaced by a set of LLHs based on a predefined set of intensities
- time-limited repetition of each low-level heuristic
  - original LLH replaced with a new LLH enforcing repetition timeout
- acceptance based on regular mean improvement updates

Naive hyper-heuristic & LLH set transformation

Start with a naive hyper-heuristic

- uniformly random selection of an LLH category
  - local search, ruin & recreate, mutation, crossover
- and then uniformly random selection of an LLH within the category
- while always accepting worse solutions

Key principles

- varying perturbation intensity for each low-level heuristic
  - each perturbative LLH replaced by a set of LLHs based on a predefined set of intensities
- time-limited repetition of each low-level heuristic
  - original LLH replaced with a new LLH enforcing repetition timeout
- acceptance based on regular mean improvement updates

LLH set transformation: acceptances + repetitions + intensities

- extend the NHH with LLH transformation: NHH*
- extend an existing hyper-heuristic X with LLH transformation: X*

HyFlex benchmark

- Cross-domain heuristic search competition 2011 (CHeSC)
- Standard benchmark for hyper-heuristics
- 6 problem domains
  - MAXSAT
  - binpacking
  - personnel scheduling
  - flowshop
  - traveling salesman problem
  - vehicle routing problem

- *Burke, E. K., Gendreau, M., Hyde, M., Kendall, G., McCollum, B., Ochoa, G., Parkes, A. J., and Petrovic, S. (2011). The cross-domain heuristic search challenge—an international research competition. In LION, 631–634.*

State-of-the-art hyper-heuristics

- Genetics with a large number of adaptive mechanisms (GIHH) [1]
  - winner of the original competition
- Lean GIHH [2]
  - simplified variant of original GIHH
- Fair share iterated local search (FSILS) [3]
  - simple hyper-heuristic outperforming [1]
- Thompson sampling iterated local search (TSILS)[4]
  - currently the best reported results

---

[1] Mısır, M., Verbeeck, K., De Causmaecker, P., Vanden Berghe, G. An Intelligent Hyper-Heuristic Framework for CHeSC 2011. In Learning and Intelligent Optimization (LION), 2012.

[2] Adriaensen S. and Nowé A. Case Study: An Analysis of Accidental Complexity in a State-of-the-art Hyper-heuristic for HyFlex. In Evolutionary Computation (CEC), 2016.

[3] Adriaensen, S., Brys, T., and Nowé, A. Fair-share ILS: a simple state-of-the-art iterated local search hyperheuristic. In Annual Conference on Genetic and Evolutionary Computation (GECCO), 2014.

[4] S. A. Adubi, O. O. Oladipupo and O. O. Olugbara. Configuring the Perturbation Operations of an Iterated Local Search Algorithm for Cross-domain Search: A Probabilistic Learning Approach. In Congress on Evolutionary Computation (CEC), 2021.

Reinforcement learning hyper-heuristics

- Monte Carlo reinforcement learning hyper-heuristic (MC) [1]
- Large-state reinforcement learning with iterated local search (LASTRL) [2]
- Q-learning to select LLH and acceptance (QHH) [3]

---

[1] Mischek, F., and Musliu, N. Reinforcement learning for cross-domain hyper-heuristics (2022). In International Joint Conference on Artificial Intelligence (IJCAI).

[2] Kletzander, L., and Musliu, N. Large-state reinforcement learning for hyper-heuristics (2023). In Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence. AAAI Press.

[3] Choong, S. S., Wong, L.-P., and Lim, C. P. (2018). Automatic design of hyper-heuristic based on reinforcement learning. Information Sciences, 436–437:89–107.

LLH set transformations: Raw power

- HyFlex benchmark
- F1 score
  - classical Formula 1 ranking used for comparison of hyper-heuristics
- NHH: unbiased random LLH selector
  $\Rightarrow$ 0 F1 points
- NHH*: NHH controlled LLH set

Rank	Method	F1 score
1.	TSILS*	105
2.	GEPHH	97
3.	LGIHH*	91
4.	FSILS*	79
5.	LGIHH	79
6.	MC*	78
7.	TSILS	76
8.	FSILS	74
9.	LUBY*	68
10.	MCTS	65
11.	**NHH***	**57**
12.	EAILS*	50
13.	MC	48
14.	GIHH	33
15.	ML	29
16.	EAILS	28
17.	LASTRL	28
18.	LUBY	27
19.	QHH	23

LLH set transformations: Raw power

- HyFlex benchmark
- F1 score
  - classical Formula 1 ranking used for comparison of hyper-heuristics
- NHH: unbiased random LLH selector
  $\Rightarrow$ 0 F1 points
- NHH*: NHH controlled LLH set
- Compare:
  - NHH* vs. GIHH

Rank	Method	F1 score
1.	TSILS*	105
2.	GEPHH	97
3.	LGIHH*	91
4.	FSILS*	79
5.	LGIHH	79
6.	MC*	78
7.	TSILS	76
8.	FSILS	74
9.	LUBY*	68
10.	MCTS	65
11.	**NHH***	**57**
12.	EAILS*	50
13.	MC	48
14.	**GIHH**	**33**
15.	ML	29
16.	EAILS	28
17.	LASTRL	28
18.	LUBY	27
19.	QHH	23

LLH set transformations: Raw power

- HyFlex benchmark
- F1 score
  - classical Formula 1 ranking used for comparison of hyper-heuristics
- NHH: unbiased random LLH selector
  $\Rightarrow$ 0 F1 points
- NHH*: NHH controlled LLH set
- Compare:
  - NHH* vs. GIHH
  - NHH* vs. recent RL-based methods

Rank	Method	F1 score
1.	TSILS*	105
2.	GEPHH	97
3.	LGIHH*	91
4.	FSILS*	79
5.	LGIHH	79
6.	MC*	78
7.	TSILS	76
8.	FSILS	74
9.	LUBY*	68
10.	MCTS	65
11.	**NHH***	**57**
12.	EAILS*	50
13.	**MC**	**48**
14.	GIHH	33
15.	ML	29
16.	EAILS	28
17.	**LASTRL**	**28**
18.	LUBY	27
19.	**QHH**	**23**

LLH set transformations: Other hyper-heuristics

- Many other hyper-heuristics extended by LLH set transformation:
  - MC, LUBY
  - LGIHH, FSILS, TSILS, EAILS
    - added only intensities
- Compare:
  - TSILS vs. TSILS*

Rank	Method	F1 score
1.	**TSILS***	**105**
2.	GEPHH	97
3.	LGIHH*	91
4.	FSILS*	79
5.	LGIHH	79
6.	MC*	78
7.	**TSILS**	**76**
8.	FSILS	74
9.	LUBY*	68
10.	MCTS	65
11.	NHH*	57
12.	EAILS*	50
13.	MC+	48
14.	GIHH	33
15.	ML	29
16.	EAILS	28
17.	LASTRL	28
18.	LUBY+	27
19.	QHH	23

LLH set transformations: Other hyper-heuristics

- Many other hyper-heuristics extended by LLH set transformation:
  - MC, LUBY
  - LGIHH, FSILS, TSILS, EAILS
    - added only intensities
- Compare:
  - TSILS vs. TSILS*
  - LGIHH vs. LGIHH*
  - ...

Rank	Method	F1 score
1.	TSILS*	105
2.	GEPHH	97
3.	**LGIHH***	**91**
4.	FSILS*	79
5.	**LGIHH**	**79**
6.	MC*	78
7.	TSILS	76
8.	FSILS	74
9.	LUBY*	68
10.	MCTS	65
11.	NHH*	57
12.	EAILS*	50
13.	MC	48
14.	GIHH	33
15.	ML	29
16.	EAILS	28
17.	LASTRL	28
18.	LUBY	27
19.	QHH	23

Real-world domains

- Developed LLH set transformations tested on 3 real-world domains
- Rich pickup and delivery problem
- Minimum shift design
- Bus driver scheduling
- Comparison of available hyper-heuristics
  - X and X* variants using the developed LLH set transformations
- Comparison to problem-specific methods and state-of-the-art results

---

- *Sassmann, V., H. Rudová, M. Gabonay and V. Sobotka (2023). Real-world vehicle routing using adaptive large neighborhood search. In EvoCop 2023, LNCS 13987, p. 34–49.*
- *Kletzander, L. and Musliu, N. (2024). Hyper-heuristics for personnel scheduling domains. Artificial Intelligence, 334:104172.*

Results on real-world domains

- Key trends same as for benchmarks:
  - $X^* \gg X$
  - exception: FSILS vs. FSILS*
- Simplicity common to top 3 methods
  - FSILS, NHH* by design
  - TSILS* simplified from TSILS
- 35 new best-known solutions (BKS)
  - 24/10/1 BKSs: X^*/X/shared

Rank	Method	F1 score
1.	**TSILS***	**306**
2.	**FSILS**	**303**
3.	**NHH***	**301**
4.	EAILS*	276
5.	LGIHH*	271
6.	TSILS	228
7.	EAILS	209
8.	MC*	205
9.	LGIHH	196
10.	LUBY*	166
11.	FSILS*	129
12.	MC$^+$	98
13.	LUBY$^+$	83

Discussion

Transformation of the low-level heuristics' set

- improved cross-domain performance for many recent hyper-heuristics

Design simplicity is your friend

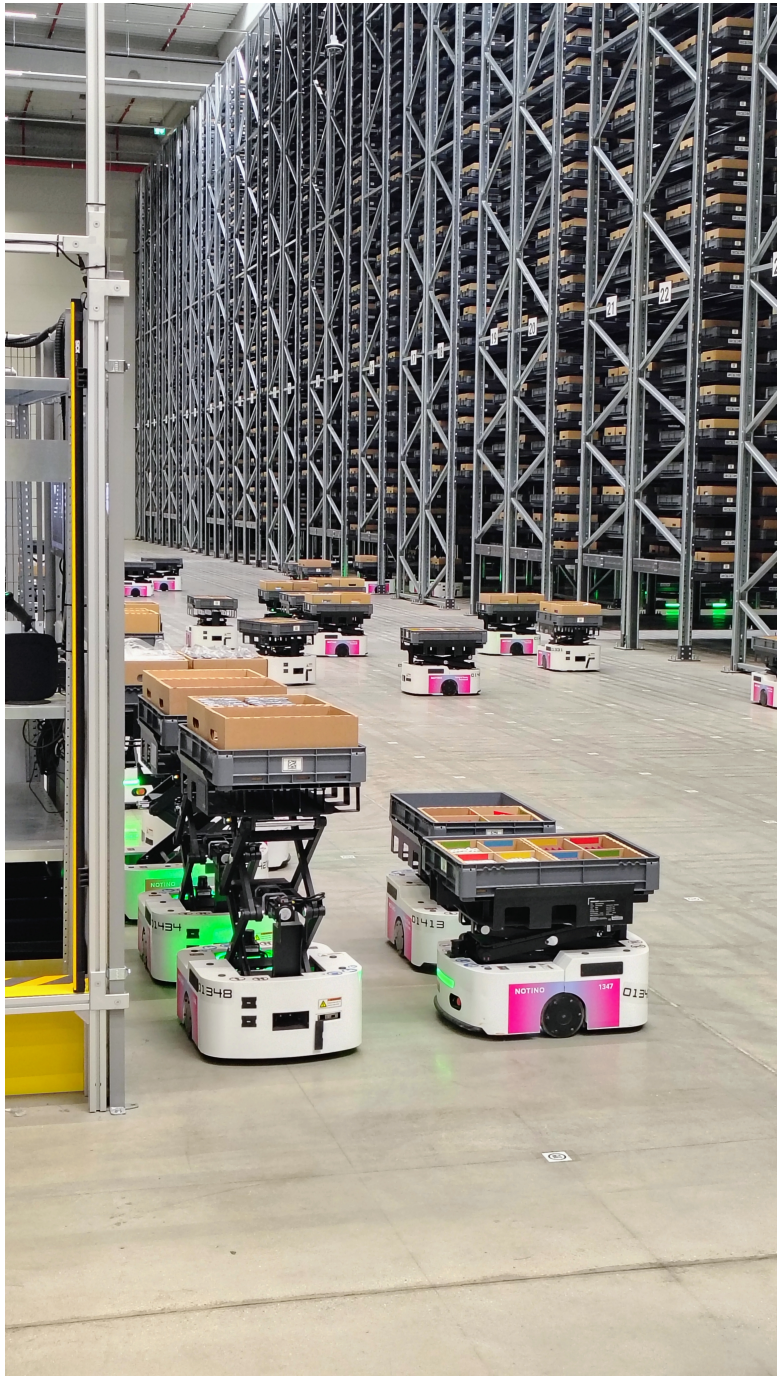
- performance of NHH*
- LGIHH vs. GIHH
- FSILS
- TSILS, EAILS vs. TSILS*, EAILS*

. . . especially on real-world problems

Trivial selection mechanism: new best-known results

- comparable or even outperforming recent hyper-heuristics

Warehouse Planning



Warehouses of Notino company

Real-world problems

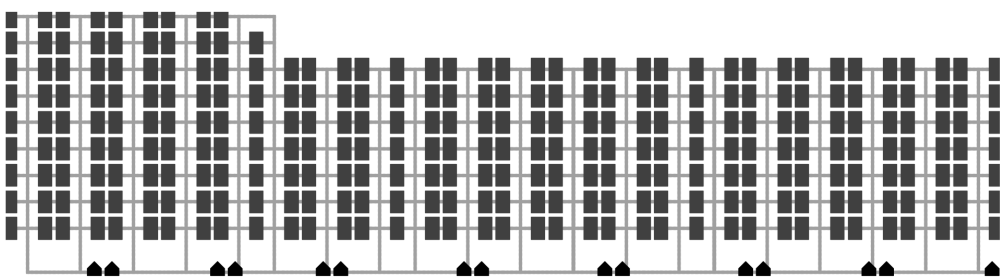
Manual picker-to-parts warehouse

NOTINO

- pickers walk through pick locations of orders
- warehouse in Czech Republic
- on average $50,000$ / up to $160,000$ orders per day
- 3.5 products per order

Robotic parts-to-picker warehouse

- parts (products) moved to pickers by automated systems (robots)
- warehouse in Romania
- $10,000$ – $23,000$ orders per day
- 4.5 products per order



Real-world vs. generated data

- streams of orders

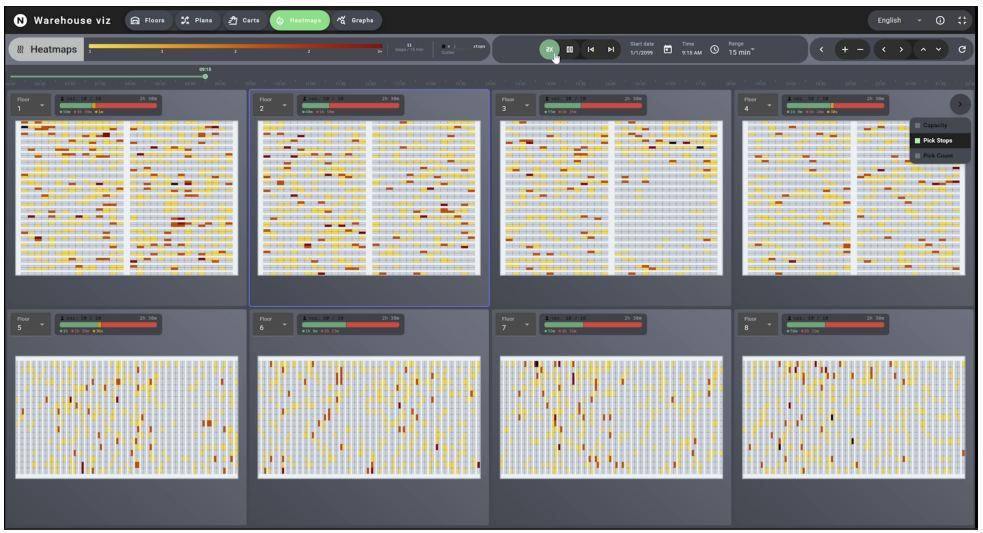
Simulation of warehouse processes

Simulation environment

- storage assignment: how to distribute storage across warehouse
- order processing: which orders to jointly process per picking tour

Planning algorithms for particular processes

- order batching into pickboxes, item position selection, routing, ...
- study of their interactions



Performance metrics

- lateness for orders
- throughput of warehouse

Order batching into pickboxes

- simple hill climbing improved performance significantly
- better than a higher complexity approach

Conclusion

Acknowledgements

Václav Sobotka, Robert Zvara

Martin Blažek, Filip Gregora, Anna Košuličová, Jakub Kováč, Samuel Krivánek, Milan Kubík, Patrik Mažári, Martin Pajerský, Vojtěch Sassmann, Jan Ševeček

Tomáš Müller, Zuzana Müllerová

Lucas Kletzander, Nysret Musliu

Michal Gabonnay, Tomáš Zahradník

Martin Kavřík, Ondřej Hastík, Ksenia Shakurova

Elegance of simplicity

Real-world problems

- course timetabling
- vehicle routing
- warehouse planning

This presentation:

Elegance of simplicity

Real-world problems

- course timetabling
- vehicle routing
- warehouse planning

Can you simplify your methodology?

- by design?
- by generalization?
- by complexity?

Can you make it more elegant?

This presentation:

Contents