Heuristic Optimization Methods
Final Assignment

Julius de Clercq (582360)

Teachers:
prof.dr. Wout Dullaert
dr. Maryam Karimi-Mamaghan

business
data science

October 2024

Mechanism 1: Choosing a Metaheuristic

- Iterated Greedy (IG) algoriths found to perform well in PFSP (Ruiz & Stützle, 2007)
    - Despite low complexity and flexible framework
    - Found to be state-of-the-art in meta-study (Fernandez-Viagas et al., 2017)
        - With a best-improvement insertion local search (LS) operator with certain tie-break criteria (Fernandez-Viagas & Framinan, 2014)
        - With initialization according to the NEH heuristic (Nawaz et al., 1983)
        - And employing Taillard's acceleration (Taillard, 1990)
        - Argument against more complex, parameterized metaheuristics (e.g. Simulated Annealing)
- But IG only uses one LS operator...
- Use Iterated Local Search (ILS) instead (Stützle, 1998)
    - Can apply the same insights found for IG
    - Same framework as extended to Q-Learning by Karimi-Mamaghan, 2022
    - Accepting non-improving solutions using Metropolis acceptance scheme (Metropolis et al., 1953)
        - With constant temperature, conform the literature (Stützle, 1998, Ruiz and Stützle, 2007, Karimi-Mamaghan et al., 2023)

Mechanism 1: Choosing LS Operators

1. Insertion (1-Opt, best-improvement)
   - In random order, reinsert jobs in another (optimal) position in the sequence
   - In line with state-of-the-art (Fernandez-Viagas et al., 2017)
   - Implementation following Karimi-Mamaghan, 2022
   - Complexity: $\mathcal{O}(n^3 m)$
2. Destruction-Construction
   - Randomly select d jobs to be removed from the solution
   - Optional insertion LS prior to reconstruction (unused)
   - Reinsert removed jobs at optimal positions, in random order
   - Perturbation size controlled by the d parameter
   - Implementation following Karimi-Mamaghan, 2022
   - Complexity: $\mathcal{O}(d(n-d)m)$
3. Swap (2-Opt, best-improvement) with Tabu list
   - Swap a job with the best possible other job in the sequence
   - More perturbing than insertion
   - No additional parameters
   - Implemented Tabu list with previously accepted sequences
   - Complexity: $\mathcal{O}(n^3 m)$

Mechanism 1: ILS

Generalized IG algorithm of Ruiz and Stützle, 2007. Using their parameter values for d and τ.

1. Initialize with NEH constructive heuristic (with Taillard's acceleration)
2. Initial LS with Swap algorithm
3. While stopping criterion is not met:
   1. Generate new solution with Destruction-Construction perturbation with $d = 4$
   2. Insertion LS
   3. Accept with Metropolis acceptance probability, with constant temperature and $\tau = 0.4$

- Complexity: $\mathcal{O}(I * n^3 m)$, where I is the number of iterations of the ILS

Mechanism 2: QILS

- Framework set out in Karimi-Mamaghan, 2022 and Karimi-Mamaghan et al., 2023
- Actions are values for the d parameter

1. Initialize with NEH constructive heuristic (with Taillard's acceleration) ($\mathcal{O}(n^2m)$)
2. Initial LS with Swap algorithm
3. Initialize Q-table with zero-values and initialize state
4. Choose d parameter at random from action set: $\mathcal{A} = \{1, 2, 5\}$
5. While stopping criterion is not met:
    1. for episode in 1:E (where E is set to six, as instructed)
        1. Destruction-construction perturbation
        2. Insertion LS
        3. Acceptance with Metropolis acceptance probability (same as in ILS)

    Update Q-table through Q-learning algorithm

- Complexity same as ILS

Mechanism 2: Q - Learning

$$Q(s, a) = Q(s, a) + \alpha \left[r + \gamma \arg\max_{a} Q(s', a) - Q(s, a) \right]$$

- Learning only happens at the end of each episode
  - . . . and episodes are expensive!
- Q-table must be kept small
- Therefore chosen for $|\mathcal{A}| = 3$ and $\mathcal{S} = \{0, 1\}$
  - $\mathcal{S}$ represents having found a local optimum
- Low complexity: $\mathcal{O}(|A|)$
- Implementation following Karimi-Mamaghan et al., 2023

Mechanism 0: Single-Operator Metaheuristic

- ILS for comparability (experimental design)
- But now removing all but one operator
- Perturbation and LS are both operators, and only one is permitted...
- Choice for Deconstruction-Construction Operator
  - Only meaningful perturbator
  - Lacking intensification power
  - Activated optional insertion LS on destroyed sequence before reconstruction

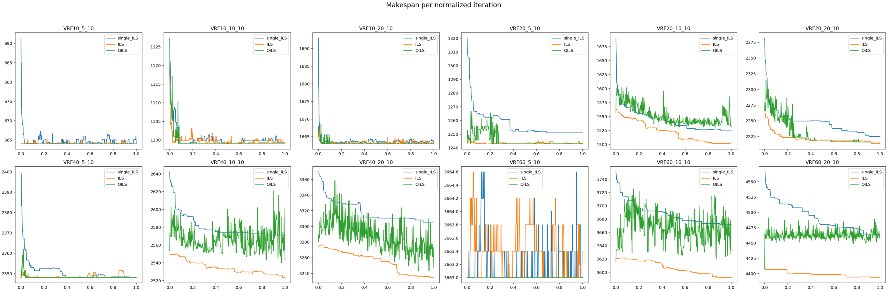
Experimental Design

1. Objectives
   1. Single-operator MH vs. Multi-Operator MH?
   2. MH1 vs. MH2?
   3. Competitiveness against state-of-the-art?
   4. Impact of instance size?
2. Problem instances, algorithms and stopping criterion discussed elsewhere
3. Parameter Tuning
   - Followed tuning done in literature
   - All parameter values taken from Ruiz and Stützle, 2007 or Karimi-Mamaghan et al., 2023
4. Performance metrics
   - Convergence considered as improvement of objective function throughout the search
     - Follows Karimi-Mamaghan, 2022
   - Runtime proxies for computational effort
   - Performance directly measured by makespans
   - Average Relative Percentage Deviation (ARPD) to compare with State-of-the-Art
     - (Fernandez-Viagas et al., 2017)

Implementation and Execution

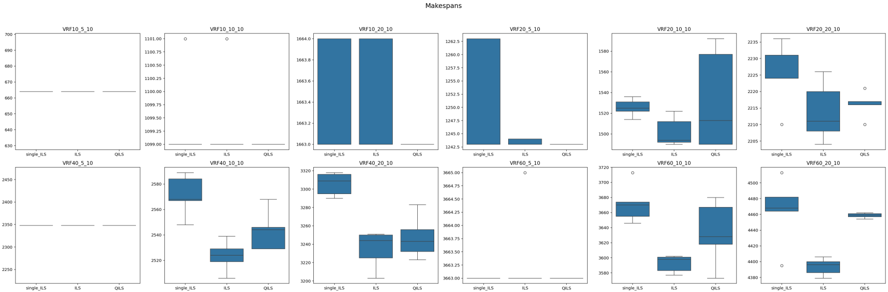
1. Execution on Snellius supercomputer
2. Parallelized execution of instances per metaheuristic
3. . . . but VRF-large instances never finished!
   - Even when setting the iteration limit to 1 . . .
4. Therefore using VRF-small instances instead
5. Iteration limit set to 600 for the sake of time-management

Convergence



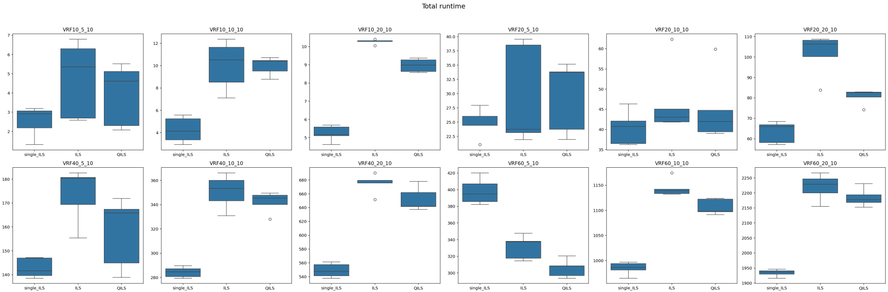
- Average over five runs per instance
- (Multi-operator) ILS more stable than QILS and Single-operator ILS (i.e. single_ILS)
- Explorative behaviour of QILS, no convincing convergence in largest instances
- single_ILS converges well, but starts poorly

Makespans



- Single-operator ILS yields worse and less robust makespans than ILS
- QILS performs worse than ILS on larger instances

Run times



- Run time given in seconds
- QILS faster than ILS
- Single-Operator ILS faster than ILS
- Comparable scaling of runtime for all algorithms

Wilcoxon tests

Instance	Makespan single_ILS, ILS	ILS, QILS	Run time single_ILS, ILS	ILS, QILS
VRF10_5_10			0.125	0.125
VRF10_10_10	1.000	0.317	0.063	1.000
VRF10_20_10	1.000	0.157	0.063	0.063
VRF20_5_10	0.285	0.157	0.625	1.000
VRF20_10_10	0.068	0.273	0.188	1.000
VRF20_20_10	0.313	0.625	0.063	0.063
VRF40_5_10			0.063	0.313
VRF40_10_10	0.063	0.066	0.063	0.625
VRF40_20_10	0.063	0.438	0.063	0.188
VRF60_5_10	0.317	0.317	0.063	0.063
VRF60_10_10	0.063	0.125	0.063	0.063
VRF60_20_10	0.063	0.063	0.063	0.063

- All values are signficant at 5%-level, but not all at 10%-level

ARPD

instance	_single_ILS	_ILS	_QILS
VRF10_5_10	0.0000	0.0000	0.0000
VRF10_10_10	0.0004	0.0004	0.0000
VRF10_20_10	0.0002	0.0002	0.0000
VRF20_5_10	0.0064	0.0003	0.0000
VRF20_10_10	0.0246	0.0087	0.0291
VRF20_20_10	0.0118	0.0067	0.0078
VRF40_5_10	0.0000	0.0000	0.0000
VRF40_10_10	0.0376	0.0183	0.0263
VRF40_20_10	0.0588	0.0361	0.0402
VRF60_5_10	0.0000	0.0001	0.0000
VRF60_10_10	0.0514	0.0287	0.0404
VRF60_20_10	0.0624	0.0455	0.0611

Research questions

1. Single-operator MH vs. Multi-Operator MH?
   - Single-operator worse on all metrics
2. MH1 vs. MH2?
   - MH1 has better performance, but MH2 tends to be a bit faster (surprisingly)
   - Comparably robust, MH2 has difficulties converging
3. Competitiveness against state-of-the-art?
   - Up to 6% ARPD
   - Not terrible, but need larger instances for better comparison
4. Impact of instance size?
   - Linear scaling with M, quadratic or cubic scaling with N
   - Seen both in empirical runtimes and in theoretical complexity

Conclusion

1. Multi-operator MH is better than single operator
2. Q-learning did not yield improvements
3. Not terrible compared to State-of-the-Art
   - But inconclusive due to small instances
4. Large instances infeasible due to computational effort

Thank you!

References I

Fernandez-Viagas, V., & Framinan, J. M. (2014).On insertion tie-breaking rules in heuristics for the permutation flowshop scheduling problem. *Computers & Operations Research*, *45*, 60–67.

Fernandez-Viagas, V., Ruiz, R., & Framinan, J. M. (2017).A new vision of approximate methods for the permutation flowshop to minimise makespan: State-of-the-art and computational evaluation. *European Journal of Operational Research*, *257*(3), 707–721.

Karimi-Mamaghan, M. (2022). *Hybridizing metaheuristics with machine learning for combinatorial optimization: A taxonomy and learning to select operators* [Doctoral dissertation, Ecole nationale supérieure Mines-Télécom Atlantique].

Karimi-Mamaghan, M., Mohammadi, M., Pasdeloup, B., & Meyer, P. (2023).Learning to select operators in meta-heuristics: An integration of q-learning into the iterated greedy algorithm for the permutation flowshop scheduling problem. *European Journal of Operational Research*, *304*(3), 1296–1330.

References II

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E. (1953).Equation of state calculations by fast computing machines. *The journal of chemical physics*, *21*(6), 1087–1092.

Nawaz, M., Enscore Jr, E. E., & Ham, I. (1983).A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega*, *11*(1), 91–95.

Ruiz, R., & Stützle, T. (2007).A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. *European journal of operational research*, *177*(3), 2033–2049.

Stützle, T. (1998). *Applying iterated local search to the permutation flow shop problem* (tech. rep.). Citeseer.

Taillard, E. (1990).Some efficient heuristic methods for the flow shop sequencing problem. *European journal of Operational research*, *47*(1), 65–74.