Adaptive Multi-Objective Local Search Algorithms

Permutation Flowshop Scheduling Problem

Aymeric Blot[1] Marie-Éléonore Kessaci[1] Laetitia Jourdan[1]
Patrick De Causmaecker[2]

[1]Université de Lille, CRIStAL – UMR CNRS 9189, ORKAD team, France

[2]KU Leuven, Department of Computer Science, CODeS, KULAK, Belgium

LION 12 – Kalamata, Greece – June, 2018

Université de Lille

CRIStAL
Centre de Recherche en Informatique, Signal et Automatique de Lille

KU LEUVEN

Context

Multi-Objective Optimisation

- Local search algorithms
- Bi-objective optimisation
- Permutation problems (PFSP, TSP, QAP)

Automatic Algorithm Design

- Offline design (algorithm selection, parameter tuning)
- Online design (hyper-heuristics, parameter control)

[Eiben et al., 1999][Hamadi et al., 2012][Karafotias et al., 2015]

Motivations

Offline Design

- Prediction based
- Instance classes / distributions
- Computationally expensive

Online Design

- Adaptation based
- Single current instance
- *Slight* overhead

Beyond Automatic Offline Design

- How to get an adaptive version of our MOLS algorithm?
- What parameter(s) can we control?
- What control mechanism can we use?

Obstacles

Adaptive algorithm

- No easy recipe

Search space

- Too many parameters
- Categorical main parameters

Control mechanisms

- Infrequently generalisable

Multi-Objective Local Search (MOLS) Algorithms

MOLS Algorithms

- Efficient metaheuristics
- Used on many problems (e.g., scheduling, routing, assignment)
- Many combinations of parameters/strategies

MOLS Principles

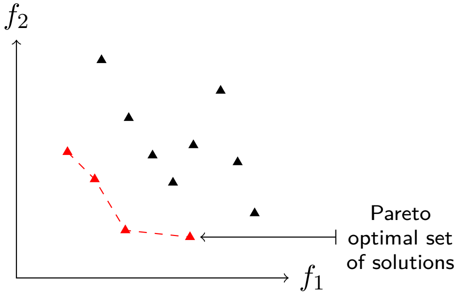
- Iteratively improve a set of solutions
- Using a neighbouring relation

[Blot et al., Journal of Heuristics, 2018]

Multi-Objective Optimisation

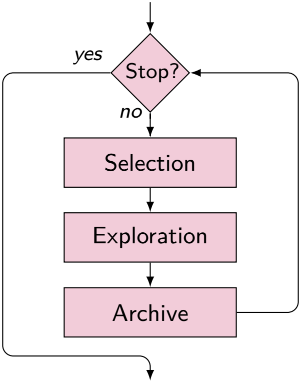
Performance Criteria

- Convergence
- Distribution
- Diversity
- Size



Static MOLS Algorithm

Core MOLS

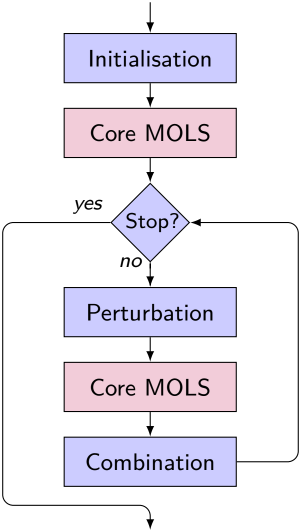


Core MOLS

- Termination
- Selection
- Exploration
- Archive

Static MOLS Algorithm

Iterated MOLS



Iterated MOLS

- Initialisation
- Perturbation
- Combination

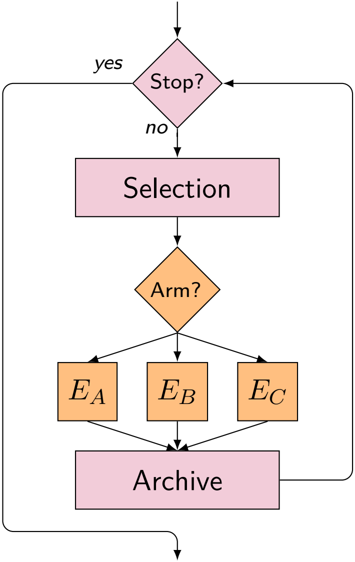
MOLS Configuration Space

Phase	Parameter	Parameter values
Selection	`select-strat`	{`all`, `rand`, `newest`, `oldest`}
Selection	`select-size`	$\mathbb{N}^+$
Exploration	`explor-strat`	{`all`, `all-imp`, `imp`, `ndom`, `imp-ndom`}
Exploration	`explor-ref`	{`sol`, `arch`}
Exploration	`explor-size`	$\mathbb{N}^+$
Archive	`bound-strat`	{`unbounded`, `rand`, `replace`}
Archive	`bound-size`	$\mathbb{N}^+$
Perturbation	`perturb-strat`	{`kick`, `kick-all`, `restart`}
Perturbation	`perturb-size`	$\mathbb{N}^+$
Perturbation	`perturb-strength`	$\mathbb{N}^+$

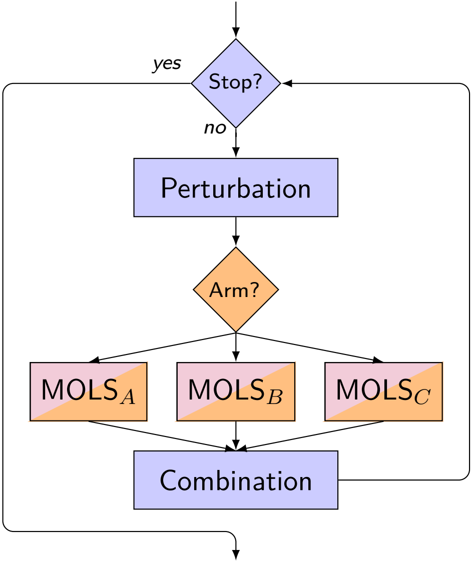
MOLS Configuration Space

Phase	Parameter	Parameter values
Selection	`select-strat`	{`all`, `rand`, `newest`, `oldest`}
Selection	`select-size`	$\mathbb{N}^+$
Exploration	`explor-strat`	{`all`, `all-imp`, `imp`, `ndom`, `imp-ndom`}
Exploration	`explor-ref`	{`sol`, `arch`}
Exploration	`explor-size`	$\mathbb{N}^+$
Archive	`bound-strat`	{`unbounded`, `rand`, `replace`}
Archive	`bound-size`	$\mathbb{N}^+$
Perturbation	`perturb-strat`	{`kick`, `kick-all`, `restart`}
Perturbation	`perturb-size`	$\mathbb{N}^+$
Perturbation	`perturb-strength`	$\mathbb{N}^+$

Adaptive MOLS Algorithm

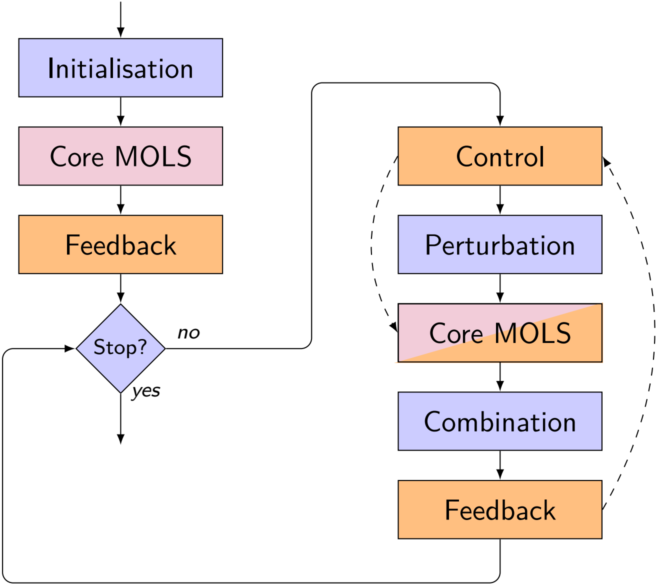


Core MOLS



Iterated MOLS

Adaptive MOLS Algorithm



Control Mechanisms

Constraints

- Single categorical parameter
- Few possible values (up to 9)

Generic Parameter Control

- Random mechanisms
- Probability based mechanisms
- Multi-armed bandits mechanisms
- Reinforcement learning

[Karafotias et al., 2015]

Control Mechanisms

Constraints

- Single categorical parameter
- Few possible values (up to 9)

Generic Parameter Control

- Random mechanisms
- Probability based mechanisms
- Multi-armed bandits mechanisms
- Reinforcement learning

[Karafotias et al., 2015]

Feedback

- Hypervolume (convergence performance criteria)

Selected Generic Parameter Control

Uniform Random Control

$$p_i(t+1) = \frac{1}{N}$$

ε-greedy

$$p_i(t+1) = \begin{cases} (1-\varepsilon) + \ {}^{\varepsilon}/_{N}, & \text{if } i = arg\ max_j\ q_j(t) \\ {}^{\varepsilon}/_{N}, & \text{otherwise} \end{cases}$$

Why no UCB1?

- Very few possible arms (2 or 3)
- Very few possible control behaviours (2?)
- First results

Experimental Setup I

3-arm Control

- ndom
- imp-ndom
- imp
- random { ndom, imp-ndom, imp }
- 0.1-greedy { ndom, imp-ndom, imp }

2-arm Control

- ndom
- imp-ndom
- random { ndom, imp-ndom }
- 0.1-greedy { ndom, imp-ndom }

Experimental Setup II

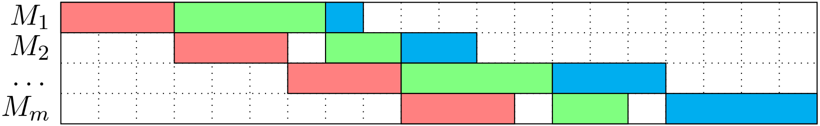
Long Term Learning (random)

- random { ndom, imp-ndom, imp }
- random { ndom, imp-ndom, imp } $\rightarrow$ (50%) { ndom, imp-ndom }
- random { ndom, imp-ndom, imp } $\rightarrow$ (20%) { ndom, imp-ndom }
- random { ndom, imp-ndom }

Long Term Learning (0.1-greedy)

- 0.1-greedy { ndom, imp-ndom, imp }
- 0.1-greedy { ndom, imp-ndom, imp } $\rightarrow$ (50%) { ndom, imp-ndom }
- 0.1-greedy { ndom, imp-ndom, imp } $\rightarrow$ (20%) { ndom, imp-ndom }
- 0.1-greedy { ndom, imp-ndom }

Permutation Flowshop Scheduling Problem



PFSP Instances

- Classical Taillard instances
  - $n \in \{20, 50, 100, 200, 500\}$ jobs
  - $m \in \{5, 10, 20\}$ machines
  - 12 valid combinations
  - 120 instances
- 2 classical objectives
  - Makespan (max of completion times)
  - Flowtime (sum of completion times)

Experimental Results

3-arm Ranking

Approach	Instance (N, M)											Avg.
	20			50			100			200		500	
	5	10	20	5	10	20	5	10	20	10	20	20	
imp	5	5	5	5	5	5	5	5	5	5	5	5	5
imp-ndom	4	4	3	4	4	4	4	1	2	1	2	1	2.8
ndom	1	1	3	1	1	1	1	1	1	1	1	1	1.2
rand_3	1	1	1	1	1	1	1	1	2	3	3	3	1.6
greedy_3	1	1	1	1	1	1	1	1	2	3	3	3	1.6

Wilcoxon signed ranked tests, Friedman post-hoc analysis

200 runs/dataset

Experimental Results

2-arm Ranking

Approach	Instance (N, M)											Avg.
	20			50			100			200		500	
	5	10	20	5	10	20	5	10	20	10	20	20	
imp-ndom	4	4	3	4	4	4	4	4	4	4	4	1	3.7
ndom	1	1	3	1	1	1	1	1	1	1	1	1	1.2
rand_2	1	1	1	1	1	1	1	1	2	1	1	1	1.1
greedy_2	1	1	1	1	1	1	1	1	2	1	1	1	1.1

Wilcoxon signed ranked tests, Friedman post-hoc analysis

200 runs/dataset

Experimental Results

Long Term Learning Rankings

Approach	Instance (N, M)												Avg.
	20			50			100			200		500	
	5	10	20	5	10	20	5	10	20	10	20	20	
rand_3	4	4	2	4	4	4	4	4	4	4	4	3	3.8
rand_ltl_50	3	1	2	1	1	1	3	3	3	2	3	3	2.2
rand_ltl_20	1	1	2	1	1	1	1	1	1	2	2	2	1.3
rand_2	1	1	1	1	1	1	1	1	1	1	1	1	1
greedy_3	1	1	1	1	4	4	4	4	4	4	4	3	2.9
greedy_ltl_50	1	1	1	1	1	1	3	3	3	3	2	3	1.9
greedy_ltl_20	1	1	1	1	3	1	1	1	1	1	2	2	1.3
greedy_2	1	1	1	1	1	1	1	1	1	1	1	1	1

Wilcoxon signed ranked tests, Friedman post-hoc analysis
200 runs/dataset

Wrap-up

Conclusions

- Very good strategies are essential
- Good strategies don't hurt control
- Bad strategies can be identified during the search

Perspectives

- New strategies
- New complex control mechanisms
- More than a single strategy