Reacting and Adapting to the Environment

Designing Autonomous Methods for Multi-Objective Combinatorial Optimisation

Aymeric Blot

Supervisor: Laetitia Jourdan
Co-advisor: Marie-Éléonore Kessaci
ORKAD team, CRIStAL, Université de Lille

PhD defence – September 21, 2018

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

Université de Lille

Thesis

Reacting and Adapting to the Environment
Designing Autonomous Methods
for Multi-Objective Combinatorial Optimisation

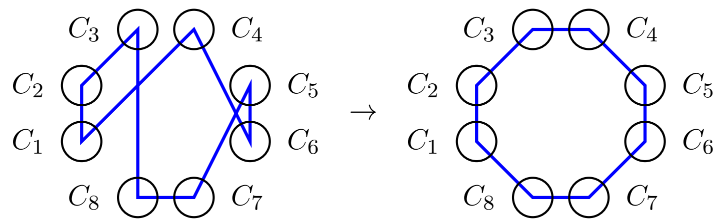
Topic *Automatic* algorithm design
Context *Multi-objective combinatorial* optimisation
Use Case *Multi-objective* local search algorithms

Contents

- **Introduction**
- **Context**
- **Multi-Objective Local Search**
- **Automatic Design**
- **Wrap-up**

Travelling Salesman Problem

Input Set of n cities, travel costs
Solutions Halmiltonian paths (permutations)
Quality Total cost (e.g., distance, time, money)



Thesis

Reacting and Adapting to the Environment
Designing Autonomous Methods
for Multi-Objective Combinatorial Optimisation

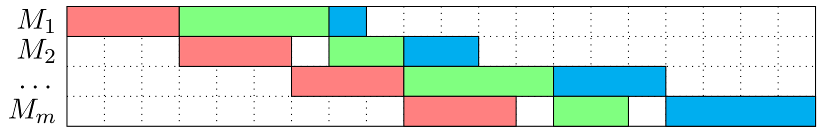
Environment

Problem Circuit board drilling? Order-picking? Vehicle routing?
Instance Sparse? Rich? Structured? Random?
Search Easy to improve? Stuck in local optima?

Permutation Flowshop Scheduling Problem

Input Set of n jobs, processing times on m machines
Solutions Jobs schedules (permutations)
Quality Various, e.g.:

- Makespan (max of completion times)
- Flowtime (sum of completion times)



Thesis

Reacting and Adapting to the Environment
Designing Autonomous Methods
for Multi-Objective Combinatorial Optimisation

Ambitions

Automatically, in a multi-objective context:

- Design algorithms variants for specific problem characteristics
- Benefit from many existing strategies
- Avoid relying on expert knowledge

Roadmap

Reacting and Adapting to the Environment
Designing Autonomous Methods
for Multi-Objective Combinatorial Optimisation

Topic Automatic algorithm design
Context Multi-objective combinatorial optimisation
Use Case Multi-objective local search algorithms

Automatic Algorithm Design

Algorithm Performance

- Differs with the problem
- Differs with the instance
- Depends on explicit or hidden design choices

Ideas

- Select from a set of existing algorithms
- Tune a specific algorithm
- Generate new algorithms

AAD: Taxonomy Proposition

Algorithmic viewpoint

Temporal viewpoint

	parameters	components	algorithms
problem features	Tuning	Configuration	Mapping
a priori features	Setting	Setting	Selection
search features	Control	Scheduling	Scheduling

AAD: Investigated Fields

Algorithmic viewpoint

Temporal viewpoint

	parameters	components	algorithms
problem features	Tuning	Configuration	Mapping
a priori features	Setting	Setting	Selection
search features	Control	Scheduling	Scheduling

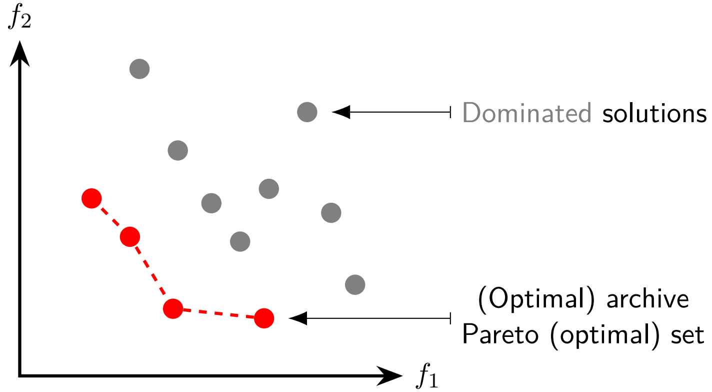
Roadmap

Reacting and Adapting to the Environment
Designing Autonomous Methods
for Multi-Objective Combinatorial Optimisation

Topic Automatic algorithm design
Context Multi-objective combinatorial optimisation
Use Case Multi-objective local search algorithms

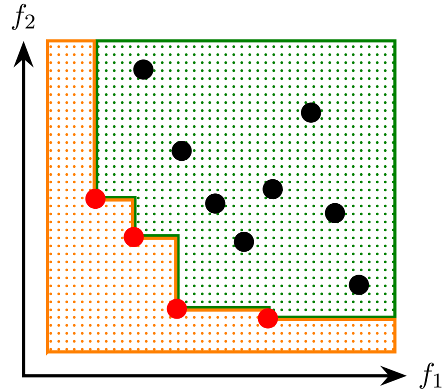
Multi-Objective Optimisation

Bi-objective minimisation

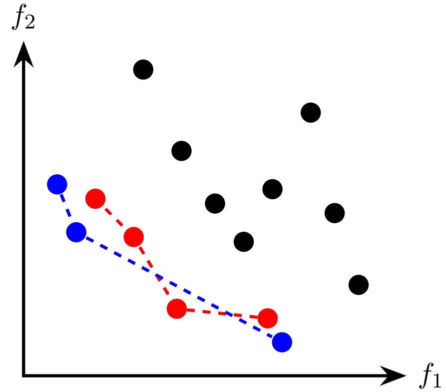


Performance Assessment

Hypervolume (1-HV)



Δ' Spread



Roadmap

Reacting and Adapting to the Environment
Designing Autonomous Methods
for Multi-Objective Combinatorial Optimisation

Topic Automatic algorithm design
Context Multi-objective combinatorial optimisation
Use Case Multi-objective local search algorithms

Questions:

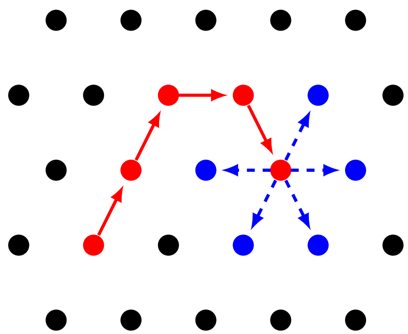
- General structure?
- Possible strategies?
- Efficiency?

Local Search Algorithms

"Similar solutions have similar quality"

Trajectory

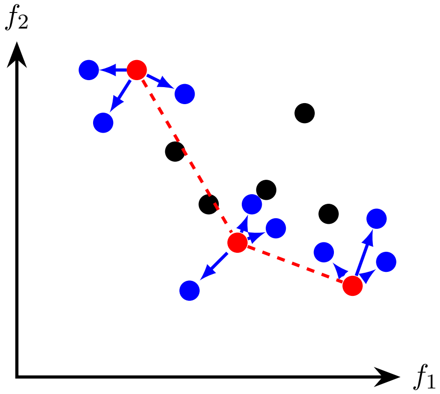
- Identify neighbours
- Move the current solution
- Iterate



Multi-Objective Local Search Algorithms

Selected History

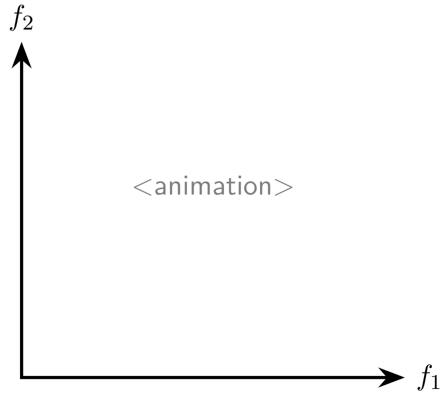
- Single trajectory
  - MOSA [Serafini, 1994]
  - TPLS [Paquete et al., 2003]
- Multiple trajectories
  - PSA [Czyzak et al., 1996]
  - MOTS [Hansen, 1997]
- Archive
  - PAES [Knowles et al.,1999]
  - PLS [Paquete et al., 2004]



MOLS Generalisation

Components

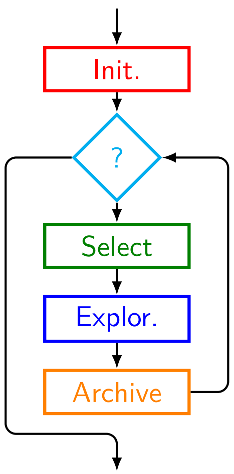
- Initialisation
- Selection
- Exploration
- Archive
- Stopping condition
- Perturbation



MOLS Generalisation

Components

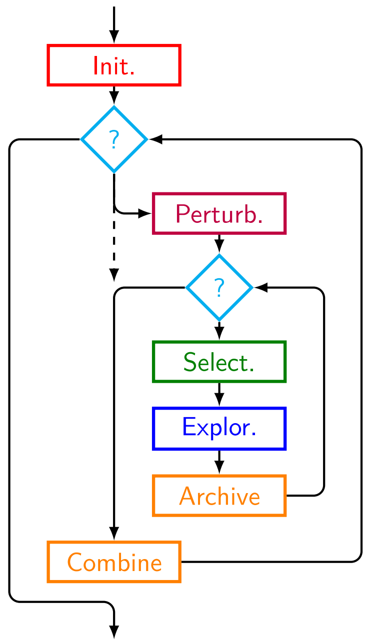
- Initialisation
- Selection
- Exploration
- Archive
- Stopping condition
- Perturbation



MOLS Generalisation

Components

- Initialisation
- Selection
- Exploration
- Archive
- Stopping conditions
- Perturbation



Selected MOLS Parameters

Parameter	Type	Parameter values
initStrat	category	$\{\dots\}$
selectStrat	category	{all, rand, newest, oldest}
selectSize	integer	$\mathbb{N}^*$
explorStrat	category	{all, imp, ndom, …}
explorRef	category	{pick, arch}
explorSize	integer	$\mathbb{N}^*$
archiveStrat	category	{bounded, unbounded, …}
archiveSize	integer	$\mathbb{N}^*$
iterationLength	integer	$\mathbb{N}^*$
iterationStagnation	integer	$\mathbb{N}^*$
perturbStrat	category	{restart, kick, …}
perturbSize	integer	$\mathbb{N}^*$
perturbStrength	integer	$\mathbb{N}^*$

Parameter Distribution Analysis

How efficient are the generated MOLS?

Protocol

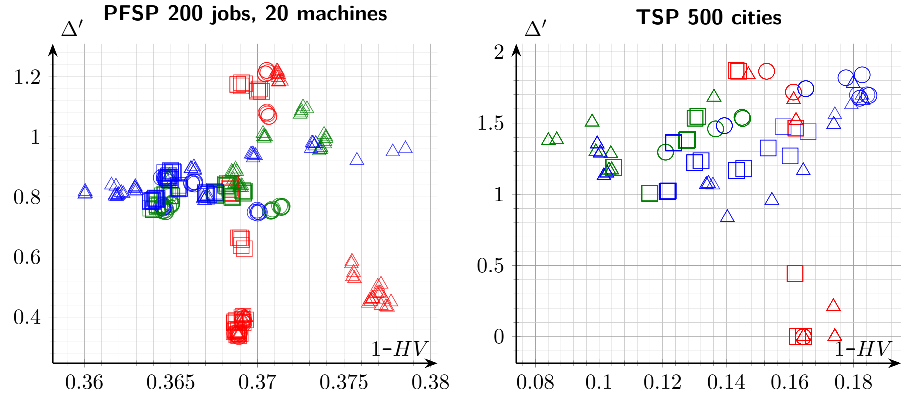
- 300 MOLS configurations
- 3 PFSP + 3 TSP scenarios
- 10 runs per instance
- Average $(1 - HV,\ \Delta')$

Scenarios

- PFSP (10 instances)
  - 50 jobs, 20 machines
  - 100 jobs, 20 machines
  - 200 jobs, 20 machines
- TSP (15 instances)
  - 100 cities
  - 300 cities
  - 500 cities

Blot, Kessaci-Marmion, and Jourdan – GECCO 2017

Results: Parameter Distribution Analysis



Exploration strategy: × imp × imp_ndom × ndom
Selection strategy: ○ all △ oldest □ rand

The configuration space is structured!
Knowledge can be extracted!
Expert knowledge is limited

Analysis

Conclusions

- Generated MOLS can be very efficient
- Parameters values are meaningful

Next Step

- Automatically design efficient MOLS algorithms

Roadmap

Reacting and Adapting to the Environment
Designing Autonomous Methods
for Multi-Objective Combinatorial Optimisation

Topic Automatic algorithm design
Context Multi-objective combinatorial optimisation
Use Case Multi-objective local search algorithms

Questions:

- How to automatically design efficient MOLS?
- Is it possible to beat expert knowledge?
- How to improve adaptability?

Algorithm Configurators

Automatic Algorithm Configuration

Goal Optimise performance over a given distribution of instances
Mean Optimisation, machine learning
Twist Data is unreliable and very expensive

Single-Objective Configuration

- irace [López-Ibáñez et al., 2016]
- ParamILS [Hutter et al., 2009]
- SMAC [Hutter et al., 2010]
- GGA++ [Ansótegui et al., 2015]

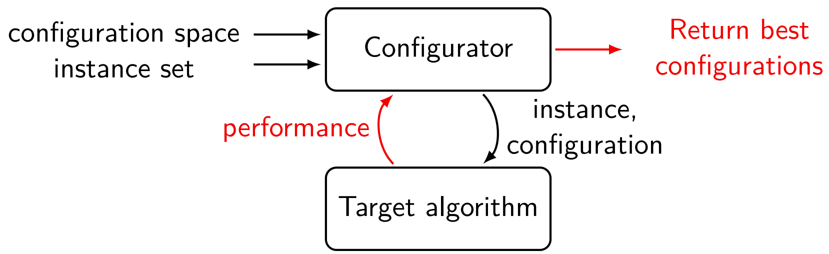
Multi-Objective Configuration

- SPRINT-Race [Zhang et al., 2015]
- MO-ParamILS [Blot et al., 2016]

MO-ParamILS

MO-ParamILS

- Extension of ParamILS for multiple performance indicators
- Iterated MOLS on the configuration space
- Outputs a Pareto set of configurations

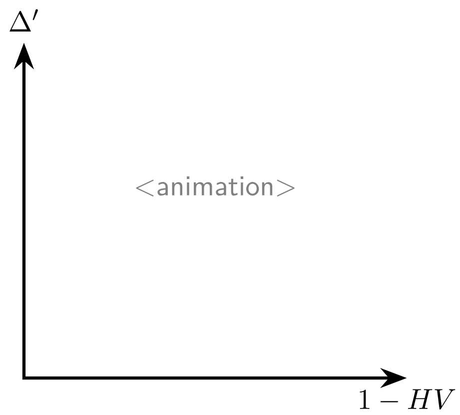


Configuration Protocol

How to ensure efficient predictions?

3 Phases

- Training
  - On training instances
  - Multiple times (e.g., $\times 20$)
- Validation
  - All final configurations
  - On training instances
- Test
  - Non-dominated configurations
  - On test instances



Automatic Configuration

How efficient is our multi-objective approach?

Configurators

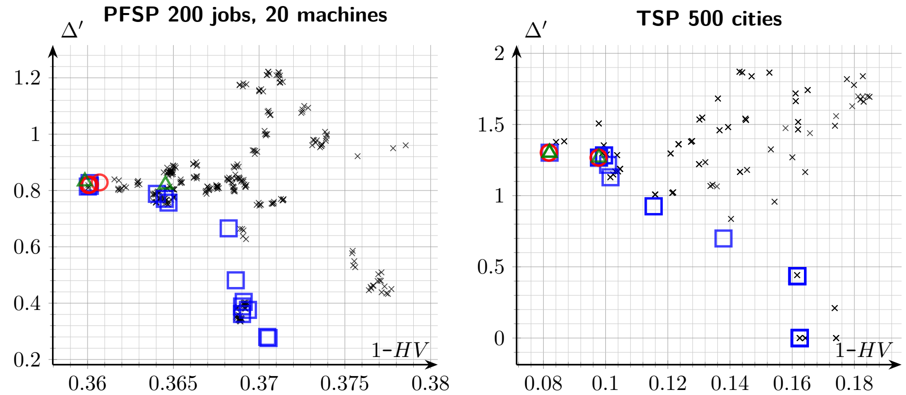
- ParamILS
  - Single-objective
  - $(1 - HV)$
- ParamILS
  - Single-objective
  - $\frac{3}{4}(1 - HV) + \frac{1}{4}\Delta'$
- MO-ParamILS
  - Multi-objective
  - $(1 - HV)$, Δ' simultaneously

Protocol

- Few configurations
  - 10×100 runs / 300 MOLS
  - 3 PFSP + 3 TSP scenarios
- More configurations
  - 20×1 000 runs / 10 920 MOLS
  - 3 PFSP + 3 TSP scenarios
- Crafted instances
  - 20×1 000 runs / 10 920 MOLS
  - 3 PFSP + 3 TSP scenarios

Blot et al. – EMO 2017 **Blot et al. – GECCO 2017** **Blot et al. – ICTAI 2018**

Results: Automatic Configuration



“Exhaustive” analysis: × (300 configurations)
Configurator: ◯ ParamILS △ ParamILS(0.75,0.25) □ MO-ParamILS

MO-ParamILS: excellent spread, no loss of convergence

Analysis

Conclusions

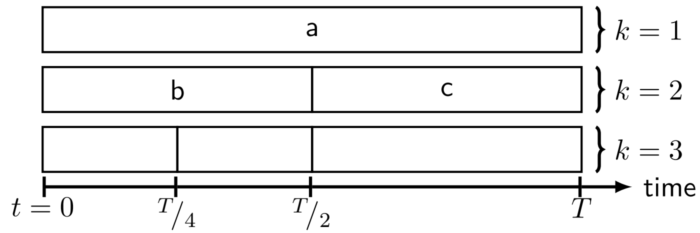
- MO-ParamILS allows much better context
- Configuration of MO algorithms is a MO problem
- Problem: predicts single configurations

Next Steps

- Scheduling
  - Sequence multiple strategies
- Control
  - Interweave multiple predictions
  - Delay predictions

Configuration Scheduling

How to better fit the algorithm to the search?



Configuration Schedules

- Performance may vary during the search
- Real-time decisions are difficult
- Static schedules can be optimised offline

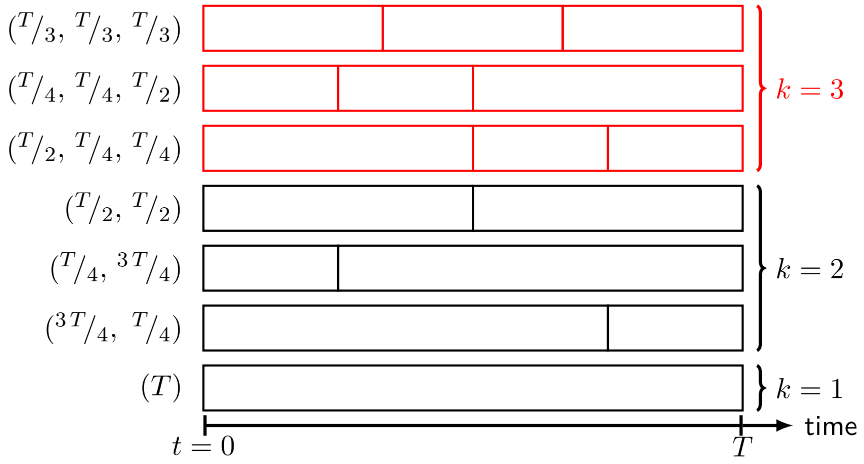
Experiments

How efficient are configuration schedules?

Protocol

- $K = 1 \quad (k = 1)$
  - Exhaustive analysis; single configurations
  - 60 configurations = 60 schedules
- $K = 2 \quad (k \in \{1, 2\})$
  - Automatic configuration; up to two configurations
  - 20×1 000 runs / 10 860 schedules
- $K = 3 \quad (k \in \{1, 2, 3\})$
  - Automatic configuration; up to three configurations
  - 20×10 000 runs / 658 860 schedules

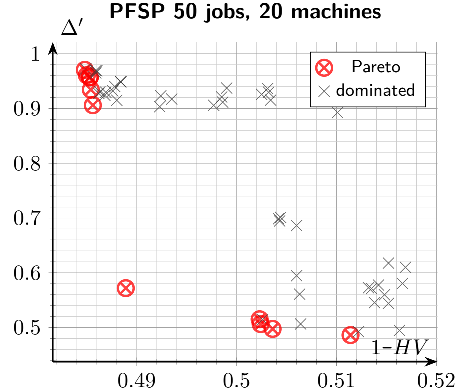
Selected $K = 3$ Configuration Schedules



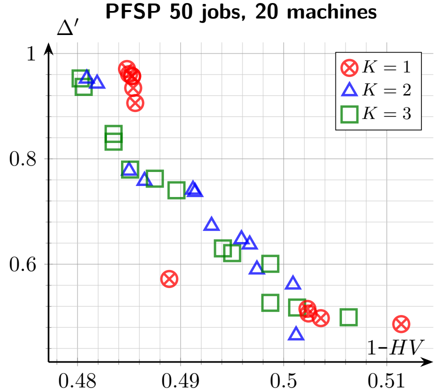
$$3 \times 60^3 + 3 \times 60^2 + 60 = 658\,860 \text{ schedules}$$

Results: Configuration Scheduling

$K = k = 1$ exhaustive analysis
PFSP 50 jobs, 20 machines



Automatic design
PFSP 50 jobs, 20 machines



Better balanced algorithms!

Analysis

Conclusions

- $k = 1$ schedules are limited
- Schedules can be optimised offline
- Combinatorial explosion

Offline Adaptation

- Schedules are still predicted
- No real-time decisions

Control

Offline Design

- Prediction based
- Instance classes / distributions
- Computationally expensive

Online Design

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

Motivations

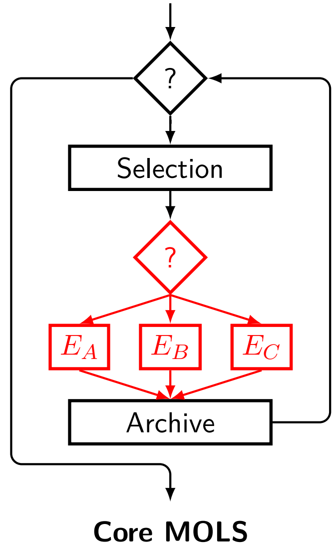
- Use control as an extension of offline learning
- Take advantage of multiple strategies during the run
- Delay the final prediction

Control Mechanisms

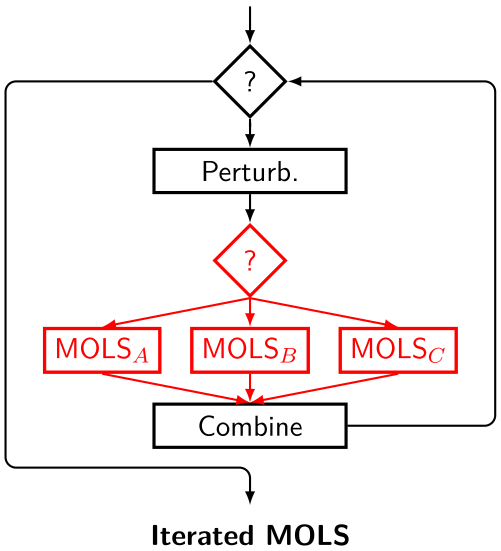
Generic Parameter Control

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

[Karafotias et al., 2015]



Adaptive MOLS Algorithm



Experiments

Can efficient strategies be determined online?

Protocol

- 2 simple control mechanisms
- 12 PFSP scenarios
- 200 runs per scenario

Strategies

- 3 arms (`imp`, `imp-ndom`, `ndom`)
- 2 arms (`imp-ndom`, `ndom`)
- 3 $\rightarrow$ 2 arms

Simple Control Mechanisms

- Uniform random: $p_i(t+1) = {}^1/_N$
- ε-greedy: $p_i(t+1) = \begin{cases} (1-\varepsilon) + {}^\varepsilon/_N, & \text{if } i = arg\,max_j\ q_j(t) \\ {}^\varepsilon/_N, & \text{otherwise} \end{cases}$

Results: 3-arm Ranking

Wilcoxon signed ranked tests, Friedman post-hoc analysis

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

Control fails on larger instances

Results: 2-arm Ranking

Wilcoxon signed ranked tests, Friedman post-hoc analysis

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

imp was the culprit

Results: Long Term Learning Ranking

Wilcoxon signed ranked tests, Friedman post-hoc analysis

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

Ineffective arms should be automatically removed

General Contributions and Conclusions

Automatic Algorithm Design

- Taxonomy proposition
- Multi-objective configuration, MO-ParamILS
  - MO algorithms are better optimised using a MO configurator
- Configuration scheduling
  - Better balanced algorithms can be predicted
- Control as extension of automatic configuration
  - Some design choices can be postponed to the search itself

Multi-objective Optimisation

- Wider generalisation of MOLS algorithms
- Automatic design of multi-objective algorithms

Short-Term Perspectives

Automatic design

- Extension to other algorithms
- Other multi-objective configurators
- Robustness in configurators

Automatic configuration

- Validation on other types of problems

Configuration scheduling

- Guided experimentation protocol
- More semantic representation

Online mechanisms

- More strategies, more complex mechanisms

Long-Term Perspectives

Anytime Behaviour of Algorithms

Insight Other applications of multi-objective algorithm design

Example Quality/running time trade-off

Ideas
- Designing for multiple running times
- Area-under-the-curve as fitness
- Configuration scheduling

Artificial Configuration Spaces

Insight Automatic configuration extremely time-expensive

Problem So is developing/improving/comparing configurators

Ideas
- Semantic parameter analysis
- Zero-cost configuration spaces

Publications I

- Blot, Hoos, Jourdan, Kessaci-Marmion, and Trautmann – LION 2016
  MO-ParamILS: A Multi-objective Automatic Algorithm Configuration Framework
- Blot, Pernet, Jourdan, Kessaci-Marmion, and Hoos – EMO 2017
  Automatically Configuring Multi-objective Local Search Using Multi-objective Optimisation
- Blot, Kessaci-Marmion, and Jourdan – MIC 2017
  AMH: a new Framework to Design Adaptive Metaheuristics
- Blot, Kessaci-Marmion, and Jourdan – GECCO 2017
  Automatic design of multi-objective local search algorithms: case study on a bi-objective permutation flowshop scheduling problem

Publications II

- Blot, Kessaci, Jourdan, and de Causmaecker – LION 2018
  Adaptive Multi-Objective Local Search Algorithms for the Permutation Flowshop Scheduling Problem
- Blot, López-Ibáñez, Kessaci, and Jourdan – PPSN 2018
  Archive-aware Scalarisation-based Multi-Objective Local Search for a Bi-objective Permutation Flowshop Problem
- Blot, Hoos, Kessaci, and Jourdan – ICTAI 2018
  Automatic Configuration of Multi-objective Optimization Algorithms. Impact of Correlation between Objectives
- Blot, Kessaci, and Jourdan – Journal of Heuristics, 2018
  Survey and Unification of Local Search Techniques in Metaheuristics for Multi-objective Combinatorial Optimisation