AMH: a Framework to Design Adaptive MetaHeuristics

Aymeric Blot   Marie-Éléonore Kessaci-Marmion   Laetitia Jourdan

Université de Lille, Inria, UMR 9189 – CRIStAL

MIC – 06 July, 2017

Context

Metaheuristics

- Approximation algorithms for optimisation problems
- Few assumptions about the problem (genericity)

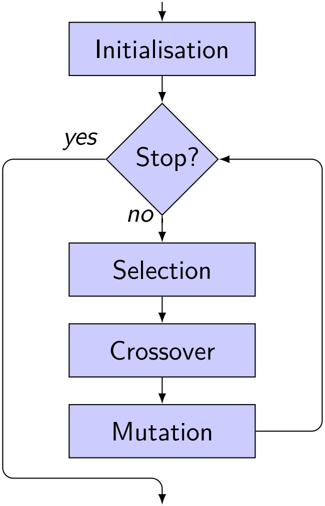
Performance

- Differs with the problem
- Differs with the instance
- Depends on its parameters

Adaptation

- Fine-tuning the parameters to the instance

Example: Genetic Algorithms



Configuration

- Initialisation?
- Selection
  - Ranking? Tournament?
- Crossover
  - Single-point? Two-point? Uniform?
  - Crossover rate?
- Mutation
  - Mutation operator?
  - Mutation rate?
- Other
  - Population size?
  - . . . ?

Algorithm Adaptation

What is the best configuration?

Offline Configuration

- Pre-solving external mechanism
- Find the most promising algorithm configuration
- Automatic tuning tools (*e.g.*, irace [López-Ibáñez *et al.*, 2016], MO-ParamILS [Blot *et al.*, 2016], GGA++ [Ansótegui *et al.*, 2016])

$\Rightarrow$ requires versatility

Online Control

- In-solving internal mechanism
- Adapt the current parameters and strategies during the search
- Generally algorithm-specific

$\Rightarrow$ requires dynamic implementation

Algorithm Design Frameworks

Some Available Frameworks

- ParadisEO[1] (C++)
- jMetal[2] (java)

Adaptation?

- Possible offline configuration (not straightforward)
- Difficult online control

[1] http://paradiseo.gforge.inria.fr/
[2] https://jmetal.github.io/jMetal/

Motivation

AMH: A Single Framework

- To facilitate offline configuration
  - Single implementation $\rightarrow$ multiple algorithms
- To enable online adaptation
  - Static algorithm $\rightarrow$ dynamic / adaptive algorithm
  - Generic control mechanisms

AMH Principles

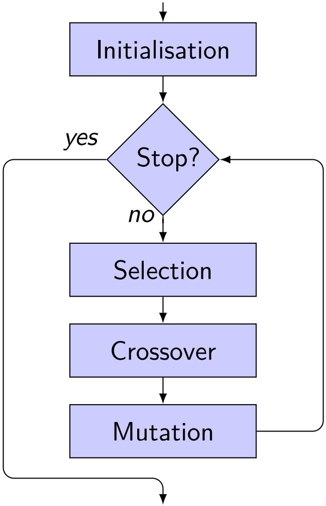
Philosophy

- Algorithm $\Leftrightarrow$ Execution flow
- Everything is a function

Ideas

- Handle its own algorithm execution flow
- Build the algorithm at runtime
- Keep the execution flow dynamic

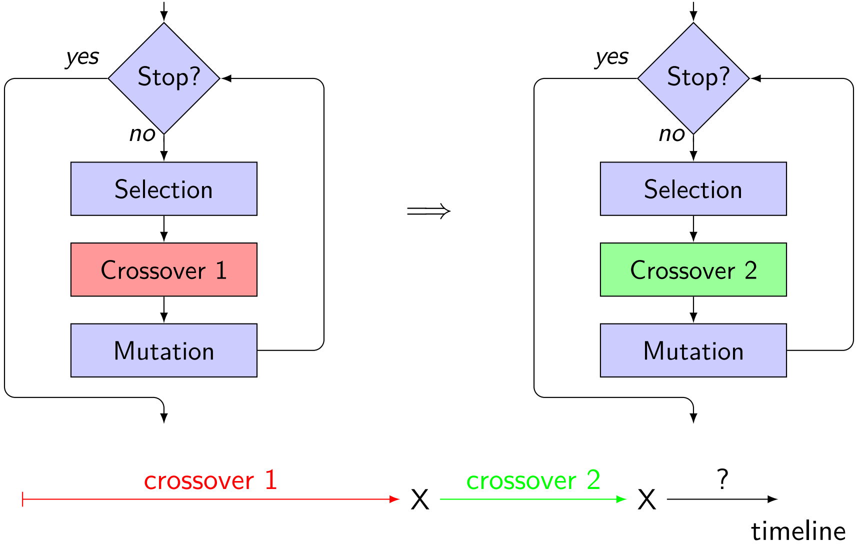
Offline Design in AMH



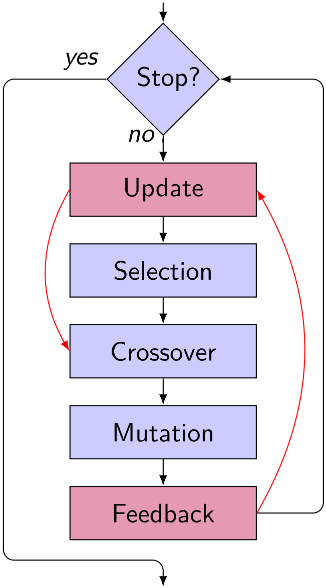
Design Process

1. Encapsulation
2. Composition
3. Integration

Online Design in AMH



Generic Control Mechanisms



Design Process

1. Update the component
2. Compute feedback

Proposed Control Mechanisms

- Random
- Probability matching
- Adaptive Pursuit
- Multi-armed bandit

AMH: *Adaptive MetaHeuristics*[3]

- Stand-alone C++ framework
- Can be used with other C++ frameworks (*e.g.*, ParadisEO)
- Handle the algorithm execution flow
- Easy to build an algorithm from basic blocks
- Easy to modify it during the execution
- Offers generic online mechanisms

[3] `https://github.com/amh-framework`

Expectations

Validate Offline Configuration with AMH

- Implement a parametric algorithm using AMH
- Use it together with automatic configurators

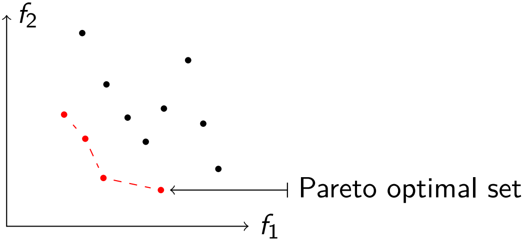
Validate Online Control with AMH

- Implement online models and mechanisms
- Use them together with static algorithms

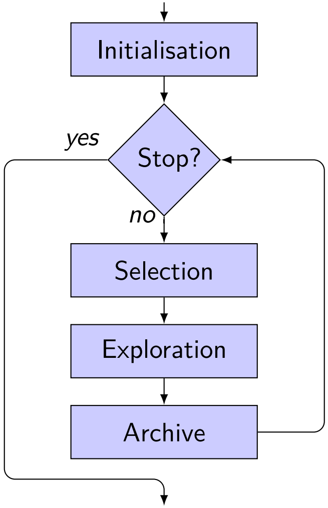
Case Study

MOLS: Multi-objective Local Search Algorithms

- Efficient metaheuristics
- Used on many problems (*e.g.*, scheduling, routing, assignment)
- Many strategies and parameters



Case Study: Multi-objective Local Search Algorithms



Example Parameters

- Selection
  - Type and number of solutions
- Exploration
  - Neighbourhood
  - Reference point
  - Type and number of neighbours
- Archive
  - Archive size
  - Type of solutions

Protocol: Exhaustive Analysis vs Automatic Design

Permutation Flowshop Scheduling Problem

- Classical Taillard instances
- Bi-objective optimisation
  - Makespan
  - Flowtime

189 MOLS Configurations

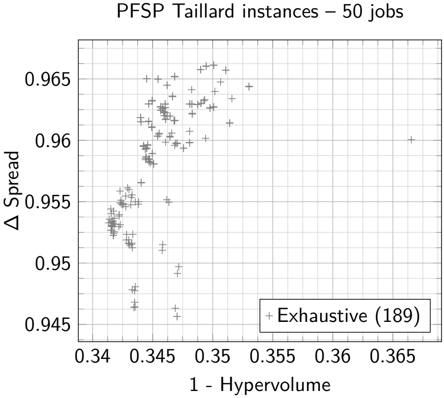
Parameter	Parameter values
`initStrat`	{`rand`, `neh`, `ig`}
`selectStrat`	{`all`, `rand`, `newest`, `oldest`}
`selectSize`	$\{1, 3\}$
`explorStrat`	{`all`, `imp`, `ndom`}
`explorRef`	{`pick`, `arch`}
`explorSize`	$\{1, 3\}$

Automatic Design Tool

- MO-ParamILS
- Performance indicators
  - Convergence
  - Spread

Result: Exhaustive Analysis vs MO-ParamILS

Exhaustive computational time: 115 days ; MO-ParamILS: 7 days



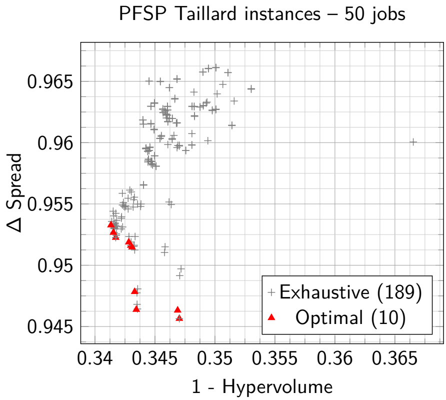
[Blot *et al.*, GECCO 2017]

Optimal Configurations

Init	Selection		Exploration		
ig	oldest	3	imp	3	pick
ig	rand	3	imp	3	pick
ig	all	-	imp	1	arch
ig	newest	3	ndom	3	pick
ig	all	-	all	-	arch
ig	rand	1	ndom	1	arch
ig	newest	3	ndom	3	arch
ig	newest	3	ndom	1	arch
ig	oldest	3	ndom	1	arch
ig	oldest	3	ndom	3	arch

Result: Exhaustive Analysis vs MO-ParamILS

Exhaustive computational time: 115 days ; MO-ParamILS: 7 days



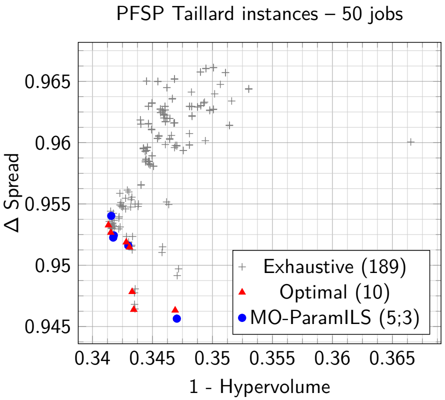
[Blot *et al.*, GECCO 2017]

Optimal Configurations

Init	Selection		Exploration		
ig	oldest	3	imp	3	pick
ig	rand	3	imp	3	pick
ig	all	-	imp	1	arch
ig	newest	3	ndom	3	pick
ig	all	-	all	-	arch
ig	rand	1	ndom	1	arch
ig	newest	3	ndom	3	arch
ig	newest	3	ndom	1	arch
ig	oldest	3	ndom	1	arch
ig	oldest	3	ndom	3	arch

Result: Exhaustive Analysis vs MO-ParamILS

Exhaustive computational time: 115 days ; MO-ParamILS: 7 days



[Blot *et al.*, GECCO 2017]

Optimal Configurations

Init	Selection		Exploration			•
ig	oldest	3	imp	3	pick	
ig	rand	3	imp	3	pick	
ig	all	-	imp	1	arch	✓
ig	newest	3	ndom	3	pick	
ig	all	-	all	-	arch	✓
ig	rand	1	ndom	1	arch	
ig	newest	3	ndom	3	arch	
ig	newest	3	ndom	1	arch	
ig	oldest	3	ndom	1	arch	
ig	oldest	3	ndom	3	arch	✓

Next

Validate Offline Configuration with AMH

- Implement a parametric algorithm using AMH
- Use it together with automatic configurators

Validate Online Control with AMH

- Implement online models and mechanisms
- Use them together with static algorithms

Conclusion

AMH: *Adaptive MetaHeuristics*[4]

- Stand-alone C++ framework
- Tested with MOLS
- Facilitates offline configuration
- Enable online control

Take-home Message

- Keep designing alternative strategies
- Adapt automatically your algorithms

[4] `https://github.com/amh-framework`