Introduction to Optimization and Automatic Algorithm Design

SSAAD 2023

MARIE-ÉLÉONORE KESSACI

Associate Professor at University of Lille

Université de Lille

Optimization problem

- An **optimization problem** is a pair (Ω, f) where
  - Search space = set of feasible (possible) solutions Ω
  - Objective function = quality criterion : $f: \Omega \rightarrow \mathbb{R}$
- **Solving** an optimization problem
  - Find the best solution(s) for the quality criterion
  - Example for a maximization problem

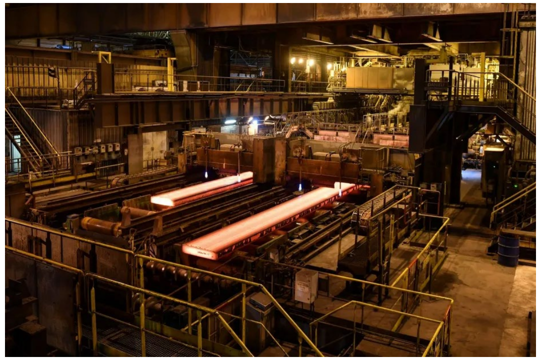
$$s^* = \operatorname*{argmax}_{s \in \Omega} f(s)$$

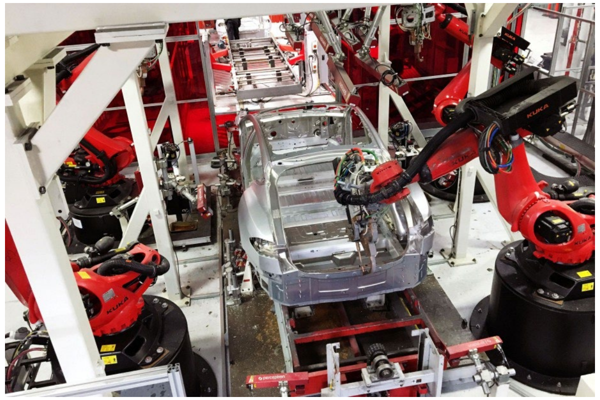
Categories of optimization problems

- Classification based on **variables**
  - **Discrete Optimization**: the set of feasible solutions is discrete or can be reduced to a discrete set
    - Combinatorial problems: traveling salesman problem, flowshop /jobshop/… scheduling, routing…
  - **Continuous Optimization**: set of real values between which there are no gaps.

- Classification based on the **criterion/criteria**
  - **Single-objective Optimization**
    - Only one criterion is minimized/maximized
  - **Multi-objective Optimization**
    - At least two criteria are optimized : simultaneously ? In sequence ? With the same interest ? …
  - **Bi-level Optimization**
    - Leader/follower each have their own criterion/criteria and evolve in separate but linked search spaces
  - …

Example of optimization problems





Solving methods

- Combinatorial Optimization Problems
  - Exact methods
    - Prog. linéaire
    - Prog. dynamique
    - Branch and Bound/Cut
  - Approximation methods
  - Heuristics
  - Metaheuristics
    - Trajectory-based
      - Local search / Climbers
      - Tabu Search
      - Simulated Annealing
      - Variable Neighborhood Search
    - Nature-inspired
      - Genetic Algorithm
      - Ant Colony Optimization
      - Particle Swarm Optimization

Trajectory-based metaheuristics

- Gradient methods are simple and efficient for numerical optimization
  - But : No relation/connection between solutions

NEIGHBORHOOD DEFINITION

- $\mathcal{N} : \Omega \rightarrow 2^{\Omega}$: neighborhood relation which associates with any solution $x \in \Omega$ a set of solutions of Ω
- $\mathcal{N}(x)$: set of neighboring solutions / neighbors of the solution x

- In practice, the neighborhood
  - Is defined from one (or more) move operators
  - Is used to connect solutions of the search space and **cross** the search space **step by step**

Optimality

- Global Optimum s^*
  - Optimale solution of the problem (may be several)

$$\forall s \in \Omega,\ \ s \preccurlyeq s^*$$

> $A \succ B$: A is (strictly) **better** than B in terms of optimization
> - minimization : $f(A) < f(B)$
> - maximization : $f(A) > f(B)$

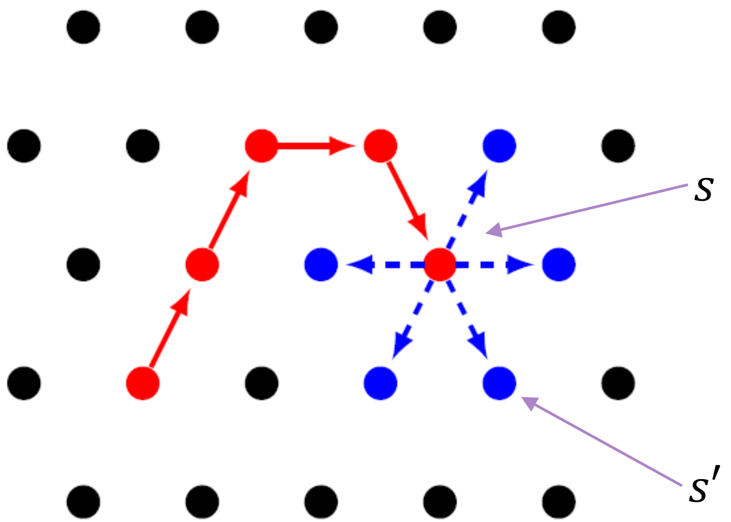
- Local Optimum s^{LO}
  - Linked to the definition of the neighborhood
  - Solution with no better neighboring solutions : $\forall s \in \mathcal{N}(s^{LO}),\ \ s \preccurlyeq s^{LO}$

Local Search

- Principle
  - Cross the search space step by step
  - Move from solutions to neighboring solutions

- Algorithm

Choose $s \in \Omega$ initial solution
Repeat
    Choose $s' \in \mathcal{N}(s)$
    If `accept` (s, s') **Then**
        $s \leftarrow s'$
Until termination criterion is reached



s is the **current solution**
s' is a **neighboring solution** of s

Exploration vs. Exploitation

- Exploration
  - Ability to explore the seach space **in width**
- Exploitation
  - Ability to explore the seach space **in depth**

- Definition of the method `accept`
  - Exploration : the solution is always accepted
    -> random walk
  - Exploitation : only improving solutions are accepted
    -> *Hill Climbing* algorithm

Choose $s \in \Omega$ initial solution
Repeat
  Choose $s' \in \mathcal{N}(s)$
  If `accept` (s, s') **Then**
    $s \leftarrow s'$
Until termination criterion is reached

- Definition of the termination criterion
  - Exploration : « never » stop, then must be fixed
    (ex : nb of iterations/evaluations or maximum runtime)
  - Exploitation : natural, because no improvement is possible among the neighborhood of the local optimum found

Choose $s \in \Omega$ initial solution
Repeat
  Choose $s' \in \mathcal{N}(s)$
  If `accept` (s, s') **Then**
    $s \leftarrow s'$
Until termination criterion is reached

Choice of the best neighbor

- Hill Climbing Algorithm
  - Inspired of the gradient descent
  - Different strategies to choose $s' \in \mathcal{N}(s)$

Choose $s \in \Omega$ initial solution
Repeat
    Choose $s' \in \mathcal{N}(s)$
    If `accept` (s, s') **Then**
        $s \leftarrow s'$
Until termination criterion is reached

- *Best improvement* Hill Climbing = Choose the best neighbors
  - Make the best (the deepest) possible move at each iteration
  - Need the evaluation of the whole neighborhood of a solution

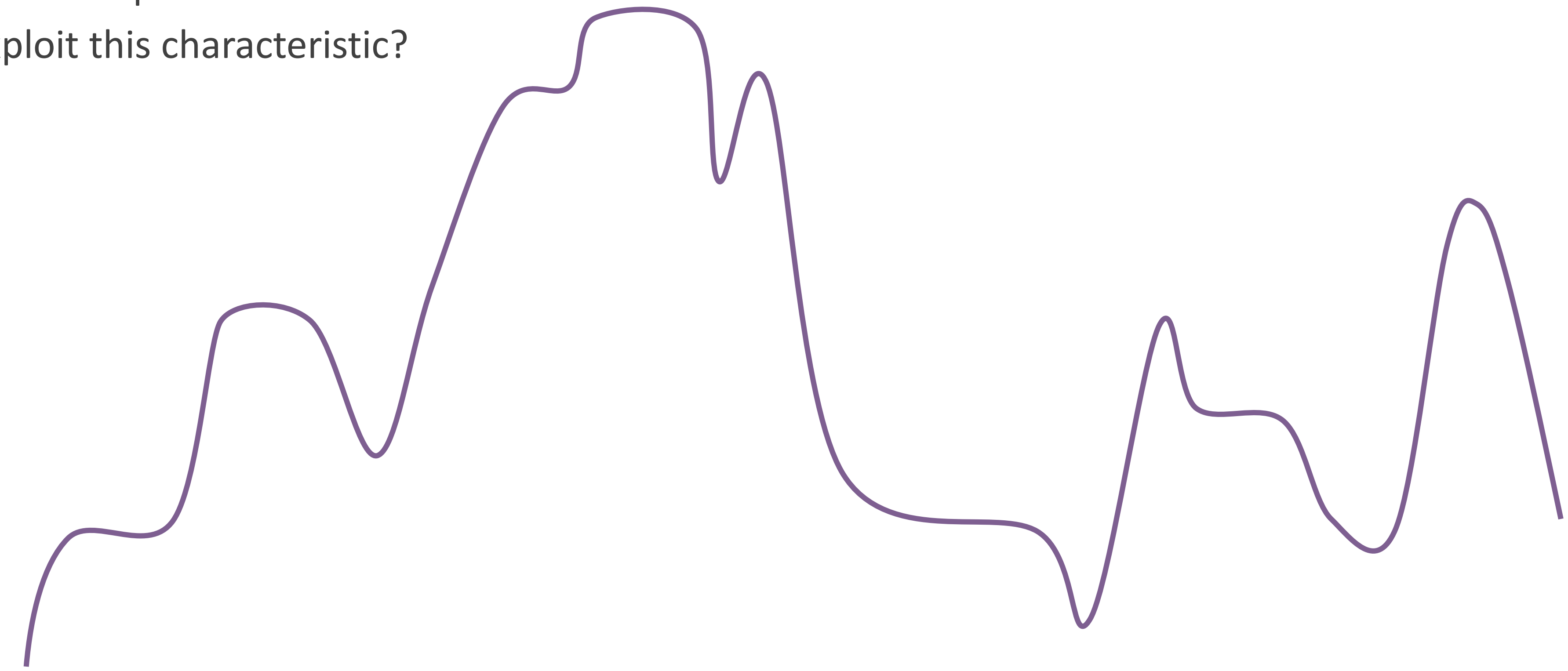
Choose $s \in \Omega$ initial solution
Evaluate s
Reapet
    Choose $s' \in \mathcal{N}(s)$ such as $f(s')$ is maximal
    If $f(s') > f(s)$ **Then**
        $s \leftarrow s'$
Until s is a local optimum

- *First improvement* Hill Climbing = Choose an improving neighbor
  - Consider small improvements
  - Increase (generally) the number of iterations
  - Evaluate (generally) the neighborhood of the current solution only partially

Choose $s \in \Omega$ initial solution
Evaluate s
Reapet
    Choose $s' \in \mathcal{N}(s)$ such as $f(s') > f(s)$
    (if possible)
    If $f(s') > f(s)$ **Then**
        $s \leftarrow s'$
Until s is a local optimum

Equivalent Solutions and Neutrality

- Optimization problems may have numerous solutions with the **same quality**
  - What is the impact on the search space?
  - Is it possible to exploit this characteristic?
  - How to exploit this characteristic?



Tradeoff Exploration/Exploitation

- HC has a good **exploitation** ability but remains trapped in a local optimum
- A random walk has a good **exploration** ability but little chance of improvement

- Need **sophisticated trajectory-based metaheuristics** to have a good tradeoff exploration/exploitation
  - *Simulated Annealing (SA)*
  - *Tabu Search (TS)*
  - *Iterated Local Search (ILS)*
  - *Guided Local Search (GLS)*
  - *Greedy Ramdomized Adaptive Search Procedure (GRASP)*
  - *Variable Neighborhood Search (VNS)*

Simulated Annealing [Kirkpatrick et al, 1983]

- **Goal**: Escape from local optimum solutions
- **Principle**: non-zero probability to select a non improving neighboring solution

In practice, we need:

- Neighborhood definition / move operator
- Temperature: control the acceptance of a non-improving solutions
  - Large: high exploration ability (> > random walk)
  - Small: high exploitation ability (> > hill climbing)
- Cooling schedule: control the variation of the temperature

Tabu Search [Glove, 1986]

- **Goal**: Escape from local optimum solutions
- **Principle**: use a memory in order to avoid recently visited solutions

In practice, we need:

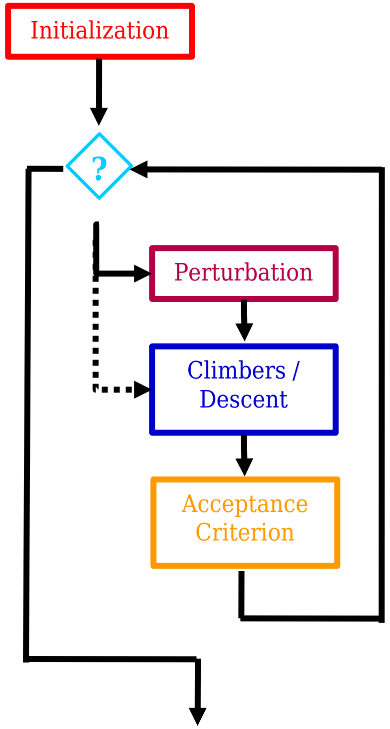
- Neighborhood
- Memory mechanism: forbid moves, forbid solutions
- Size of the memory:
  - If too small, the tabu may be unefficient
  - If too large, the exploration may be too diversified
- Aspiration criterion: Accept a tabu solution if it improves the best so far

Iterated Local Search [Lourenço et al, 2003]

- **Goal**: Escape from local optimum solutions
- **Principle**: move towards solutions that are “not to far” to restart a climber/descent

In practice, we need:

- Neighborhood
- Perturbation mechanism: manage the exploration strength
- Acceptance criterion: manage the start of the next iteration

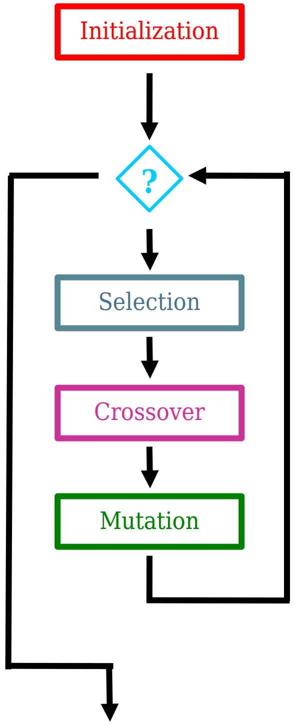


Genetic Algorithm [Holland, 1992]

- Inspired by Charles Darwin’s theory of natural evolution where the fittest individuals are selected to produce offspring
- **Principle**: merge and mutate solutions

In practice, we need:

- **Crossover** mechanism: merge two solutions to provide new solutions
- **Mutation** mechanism: add a random move in the generated solutions
- **Selection** mechanism: control the way to choose the solutions for the next iteration



Metaheuristics

Generic **&** **Flexible**

Size of tabu list

ILS

GA

Acceptance criterion

TS

HC strategy

Temperature

ACO

How to Design Efficient Metaheuristics?

SA

Perturbation mechanism

PSO

Selection

Mutation

VNS

Xover

Mutation rate

*** Design component parameters**
*** Numerical parameters**

Xover rate

Knowledge-based Design

Automatic Design

Algorithm Configuration
[López-Ibáñez 2016, Hutter 2009]

Parameter Control
[Eiben 1999, Karofatias 2015, Doerr 2018]

Algorithm Selection
[Rice 1976, Kotthoff 2014, Kerschke 2019]

Hyper-heuristics
[Burke 2013]

Landscape-based Design

Fitness Landscape Analysis
[Jones 1995]

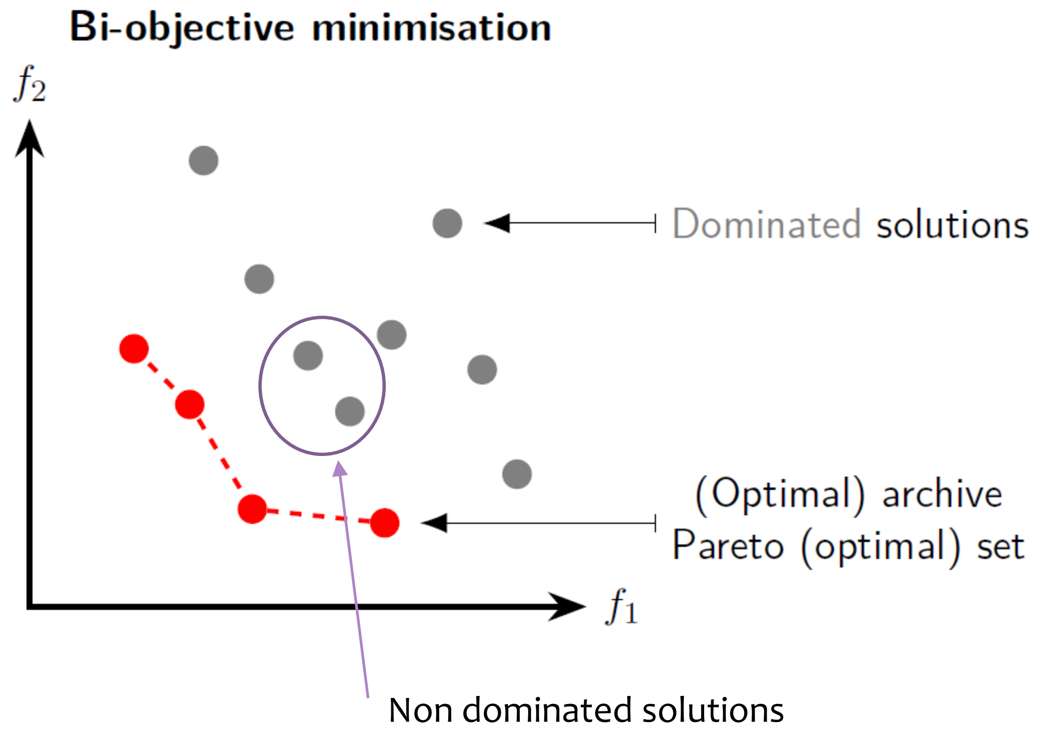
Local Optima Networks
[Ochoa 2008]

Multi-Objective Optimization Problems

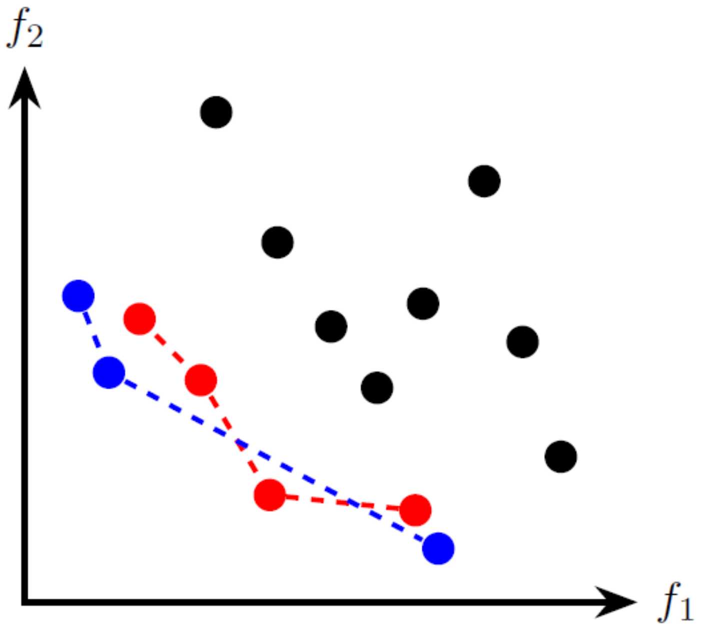
- Ω: Search space
- $\mathrm{F} = (f_1, f_2, \cdots, f_m)$: vector of m objective functions

$$\min_{x \in \Omega} \left(f_1(x), f_2(x), \cdots, f_m(x)\right)$$

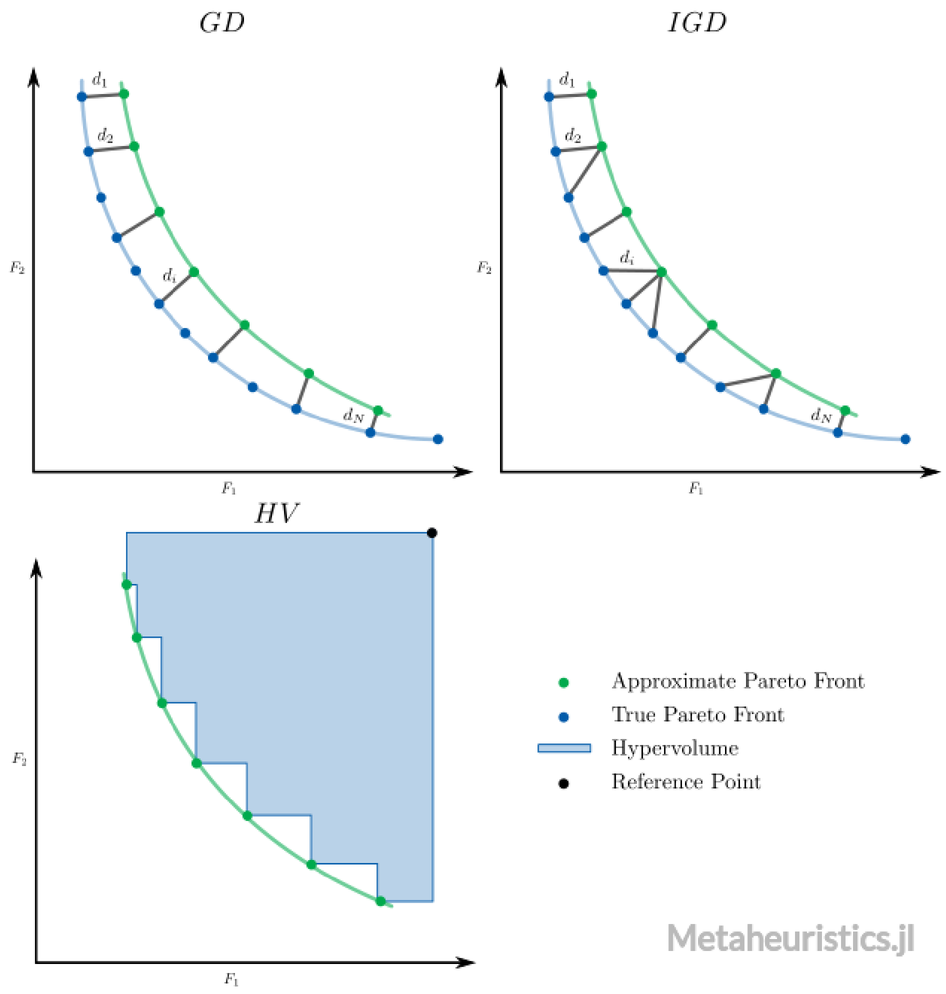
- Solving methodologies:
  - Lexicographic order
  - Aggregation
  - Pareto



Performance Assessment [Zitzler et al, 2003]



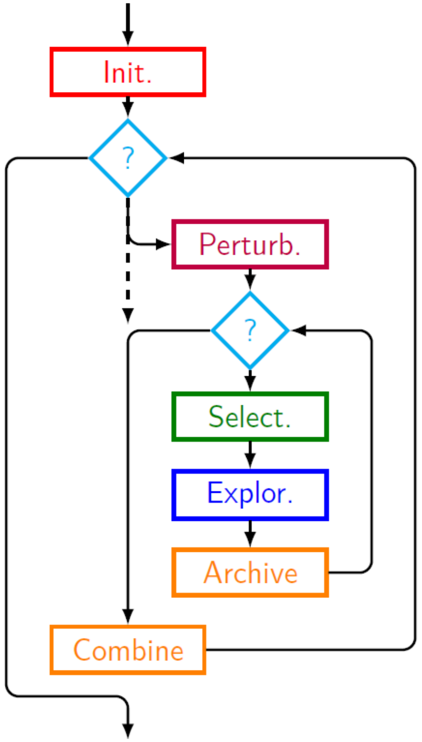
- Epsilon-indicator
- Hypervolume-indicator
- Generational distance / Inverted Generational Distance



Metaheuristics.jl

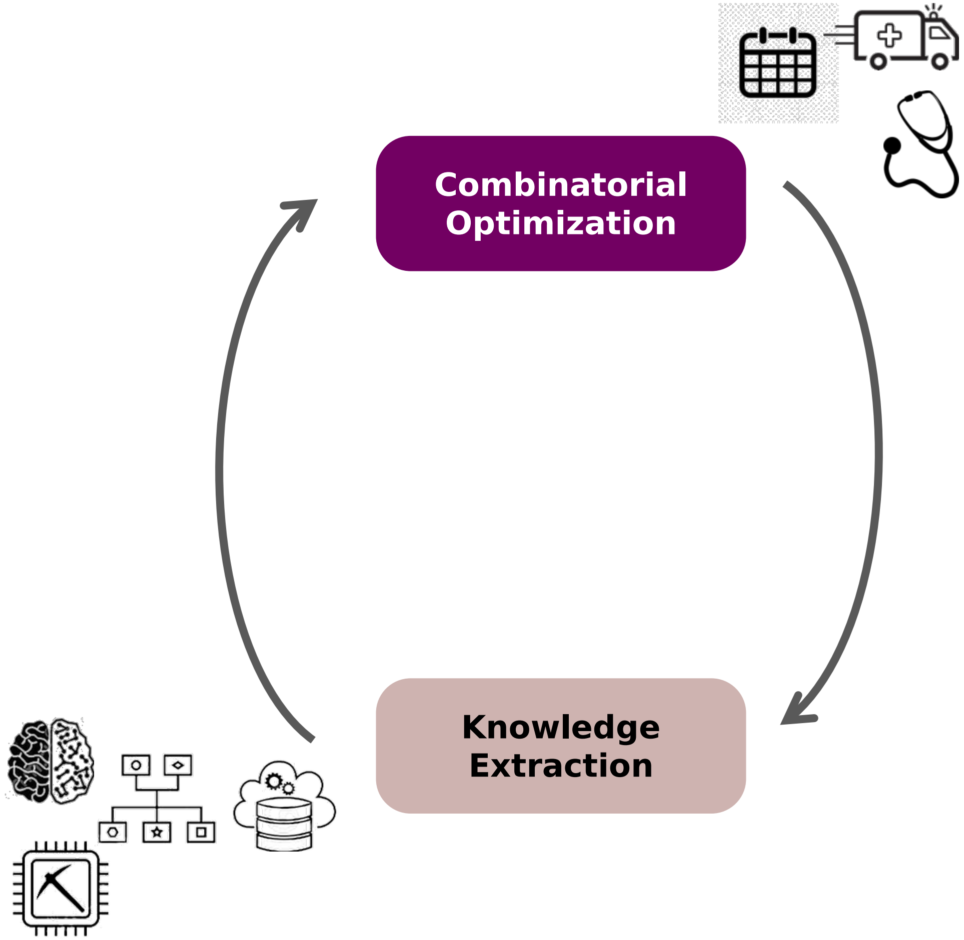
Multi-objective Approaches

- Multi-objective Local Search Algorithms [Blot et al, 2018]
  - Extension of the trajectory-based metaheuristics
  - Use an archive to store solutions

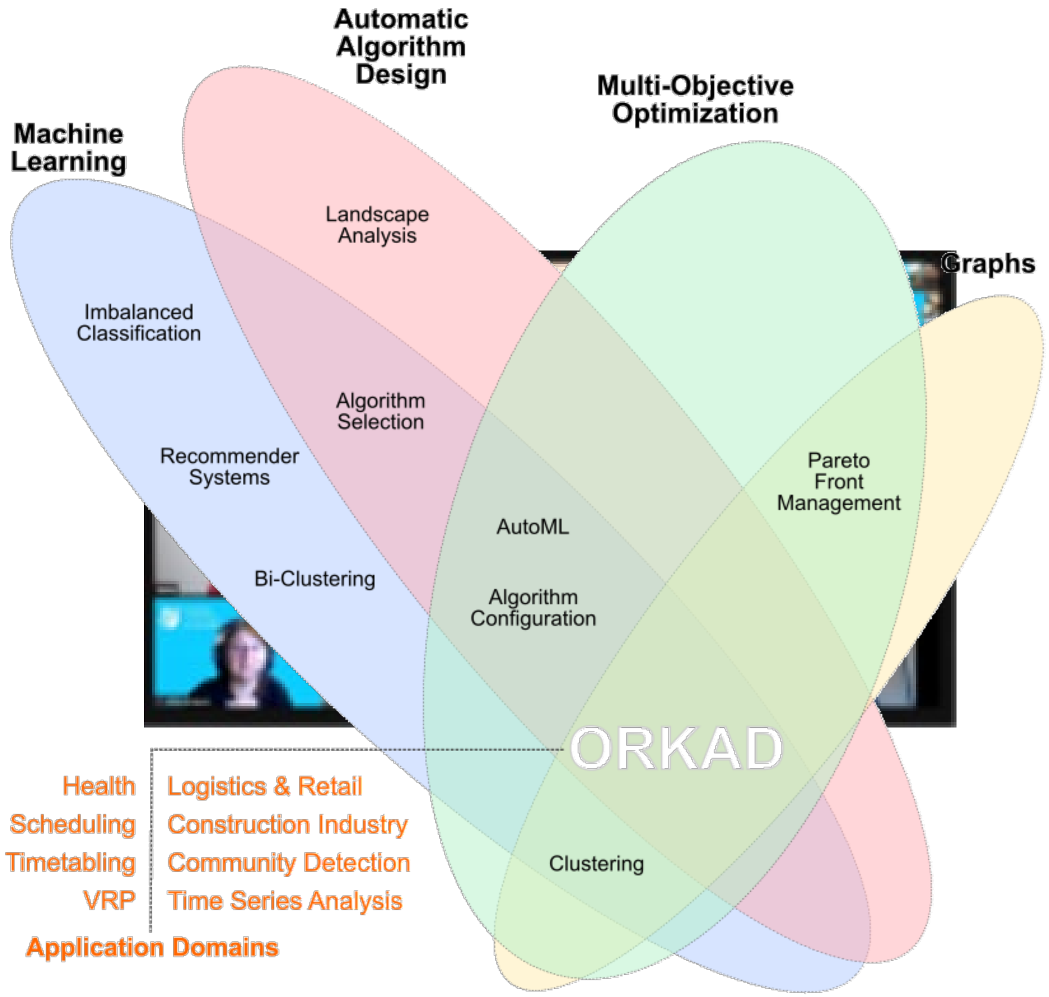


- Multi-objective nature-inspired Algorithms
  - Extension of GA: NSGA-2, NSGA-3
  - Extension of ACO: MOACO
  - Multiobjective Evolutionary Algorithm Based on Decomposition (MOEA/D)

ORKAD team

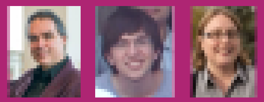


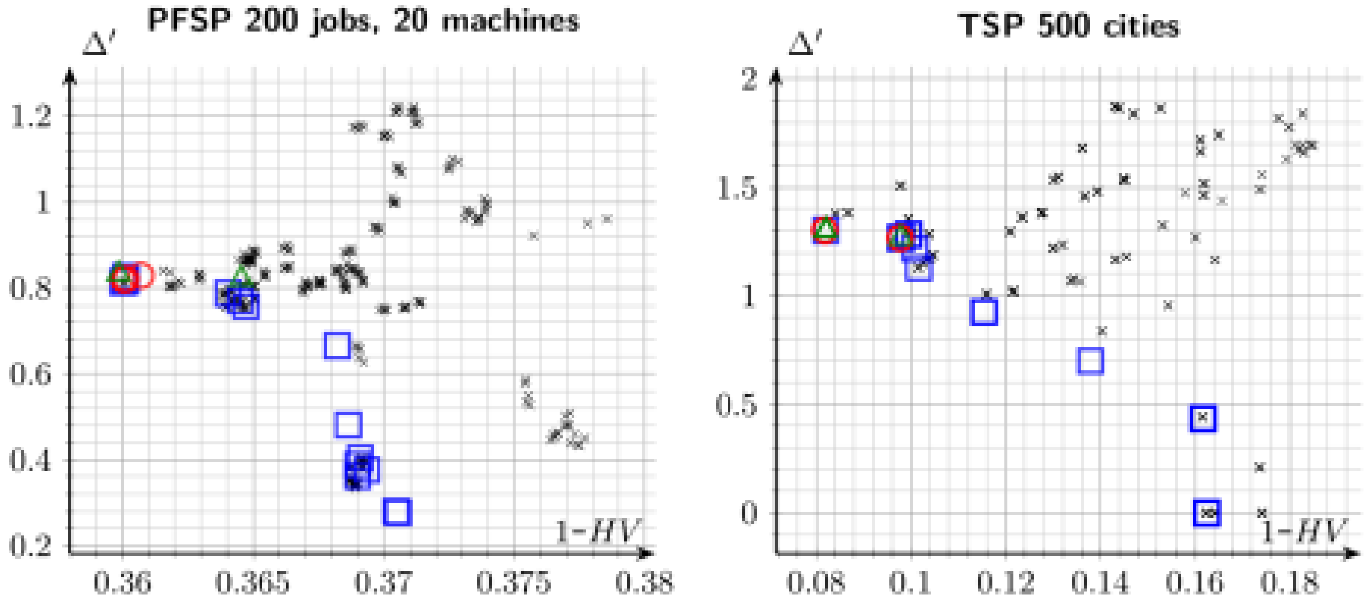
Combinatorial Optimization
Knowledge Extraction



Automatic Algorithm Design
Multi-Objective Optimization
Machine Learning
Graphs
Landscape Analysis
Imbalanced Classification
Algorithm Selection
Recommender Systems
Pareto Front Management
AutoML
Bi-Clustering
Algorithm Configuration
ORKAD
Health
Scheduling
Timetabling
VRP
Logistics & Retail
Construction Industry
Community Detection
Time Series Analysis
Clustering
Application Domains

Automatic Configuration of MO-SLS

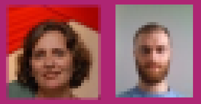


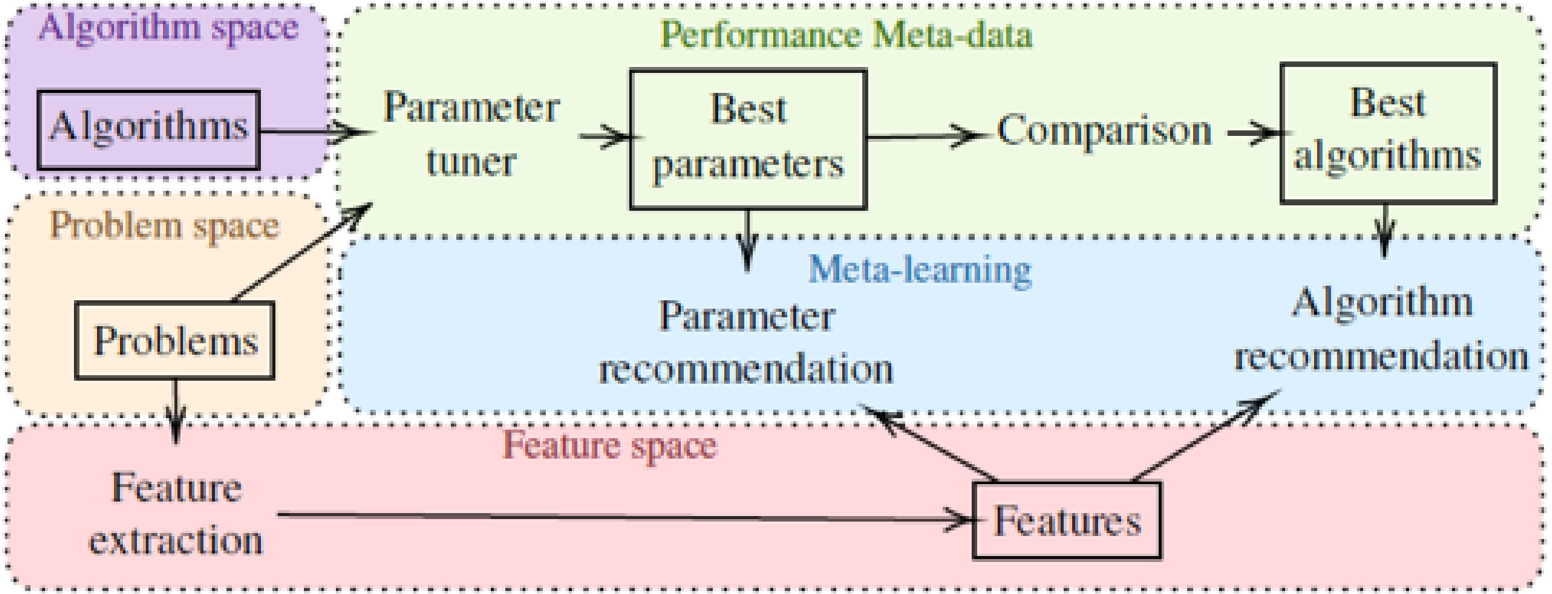


"Exhaustive" analysis: × (300 configurations)
Configurator: ◯ ParamILS △ ParamILS(0.75,0.25) ☐ MO-ParamILS

MO-AAC: excellent spread, no loss of convergence

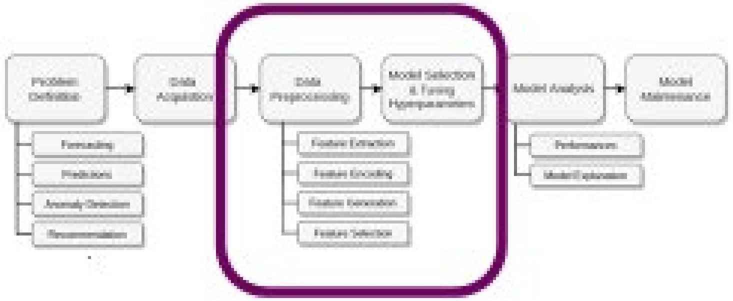
Algorithms & Parameters Recommendation

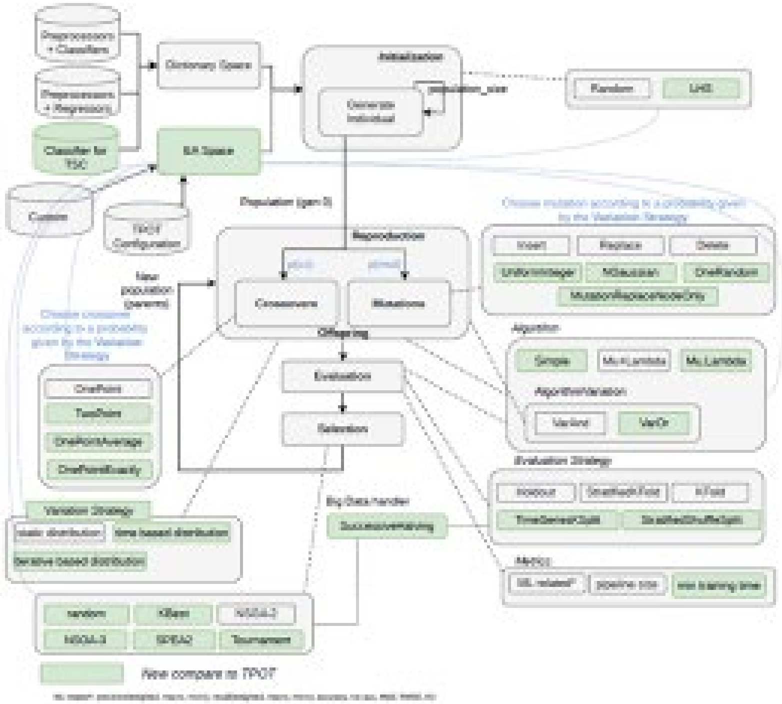




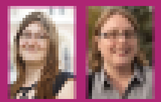
Pavelski, Delgado, Kessaci, Freitas - ITOR 2023

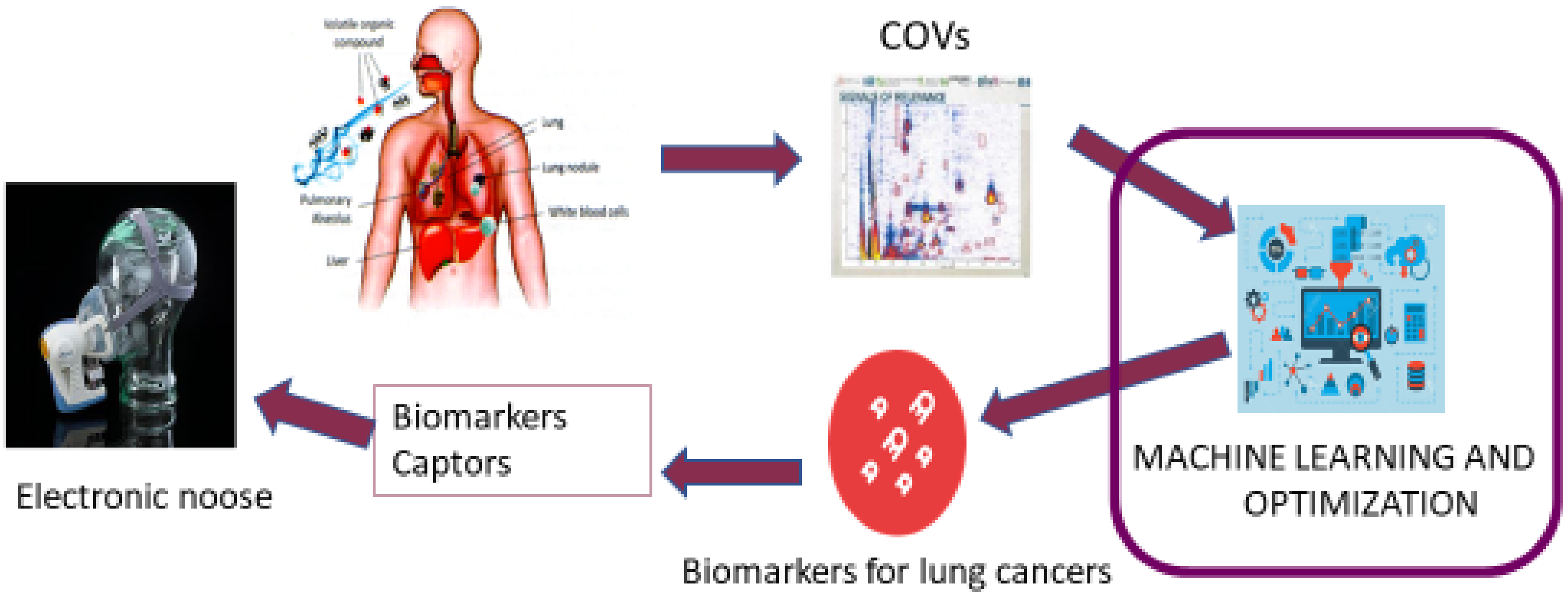
Multi-objective Auto-ML



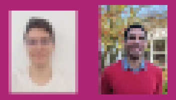


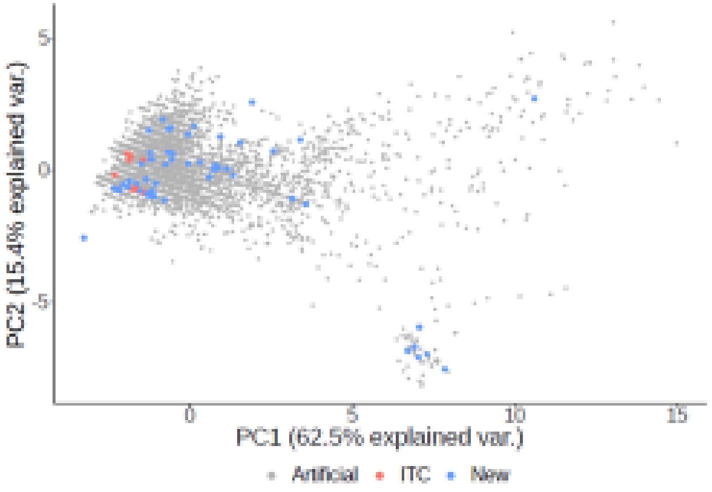
COV Detection

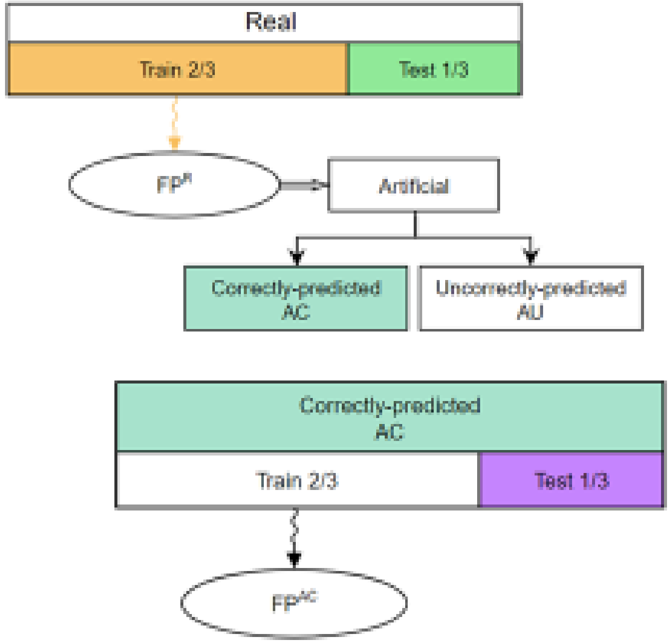




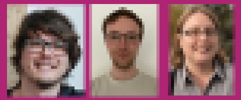
Feature-based Instances Selection

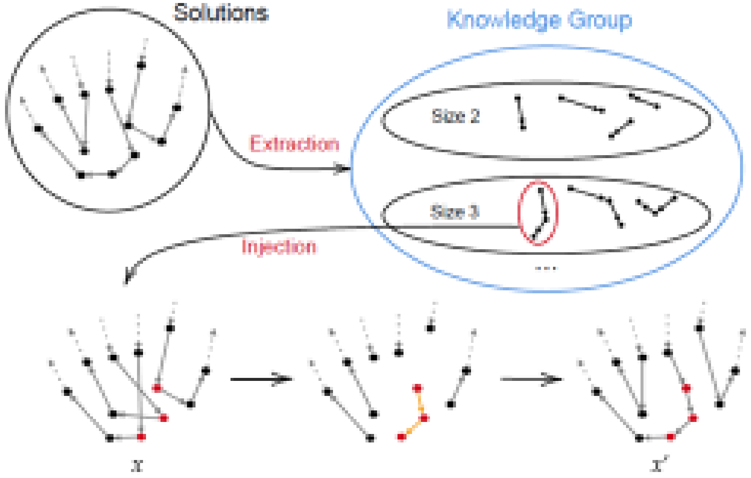


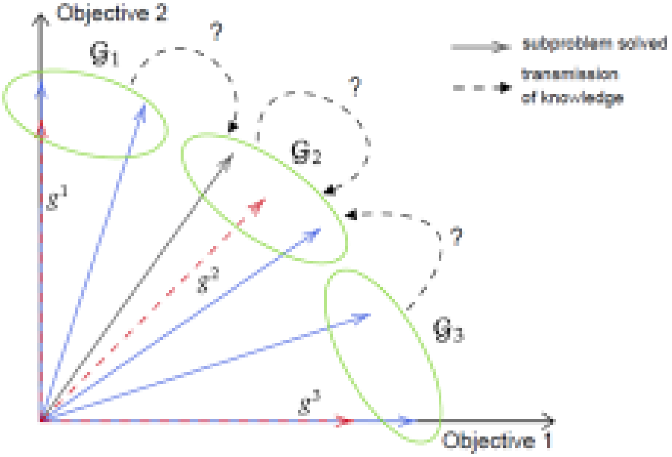


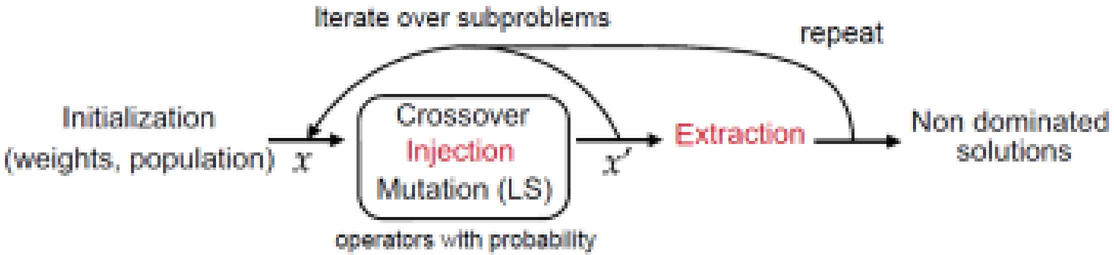


Improving MOEA/D with Knowledge Discovery



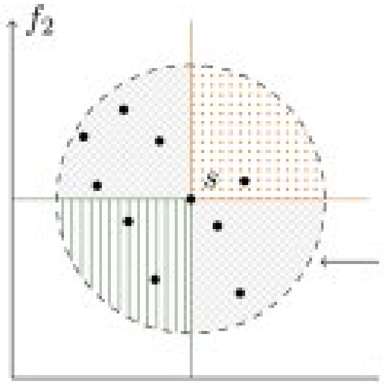




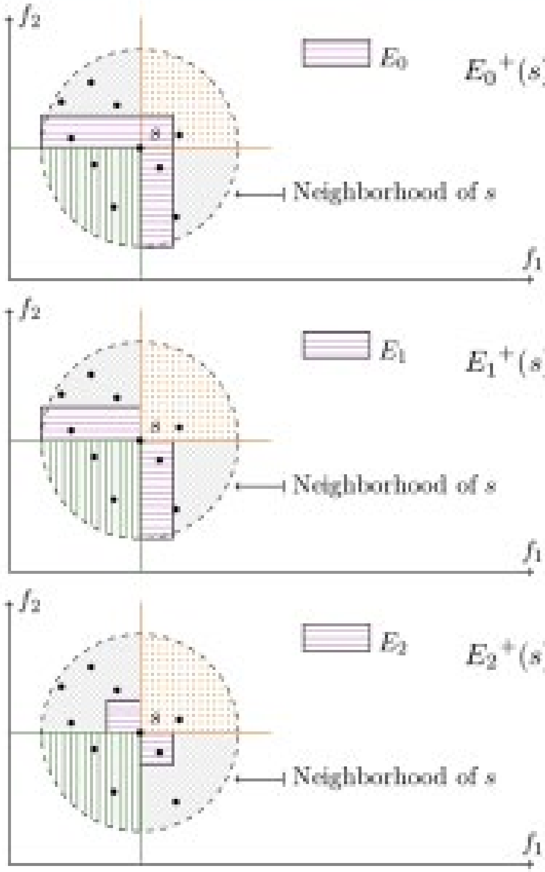


Neutrality in Multi-objective Combinatorial Optimization

First definitions



Pareto decomposition



$$E_0^{+}(s) = \{s' \in \mathcal{N}(s) \mid 0 < I_{\varepsilon+}(s', s) < \delta\}$$

$$E_1^{+}(s) = \{s' \in \mathcal{N}(s) \mid 0 < I_{\varepsilon}^{+}(s', s) < \delta \text{ and } 0 < I_{\varepsilon}^{+}(s, s')\}$$

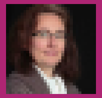
$$I_{\varepsilon+}(s', s) = max$$

$$E_2^{+}(s) = \{s' \in \mathcal{N}(s) \mid 0 < I_{\varepsilon}^{+}(s', s) < \delta \text{ and } 0 < I_{\varepsilon}^{+}(s, s') < \delta\}$$

$$\mathcal{N}_n^{\varepsilon^+}(s) = \{s' \in \quad > 0\}$$

Epsilon decomposition

Landscape-aware SLS

	Reduction of search space	Reduction of neighborhood
Case study	– no-wait FSP	– Feature selection problem
Analysis	– Structure of local optima – Definition of super-jobs	– Favorite moves (1 → 0) – Interactions between features
Knowledge integration	– Identification of super-jobs – Exploitation of super-jobs	– Estimation of neighbors quality – Intensification/diversification mechanism
Algorithm	– Iterated Greedy with super-jobs	– Tabu search

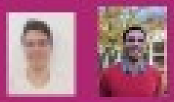
33

Mousin, Jourdan, Kessaci-Marmion and Dhaenens -- LION 2016

Mousin, Kessaci and Dhaenens -- LION 2017

Landscape-based Design

Landscape-based Performance Prediction of Local Search



Room	t0	t1	t2	t3	t4	t5	t6	t7
	Day 0				Day 1			
A	Data				Com			
B			Com				Data	
C			Prog	Data	Prog			
D	SQL			SQL		Prog