Multiobjective Optimization Algorithms

Sébastien Verel

LISIC
Université du Littoral Côte d'Opale
Equipe OSMOSE

verel@univ-littoral.fr
http://www.lisic.univ-littoral.fr/~verel

Master informatique WeDSci, ULCO,

2023, version 0.1

Single Objective Optimization

Inputs

- Search space: Set of all feasible solutions,
$$\mathcal{X}$$
- Objective function: Quality criterium
$$f : \mathcal{X} \rightarrow \mathbb{R}$$

Goal

Find the best solution according to the criterium

$$x^\star = \text{argmax } f$$

But, sometime, the set of all best solutions, good approximation of the best solution, good 'robust' solution...

Context

Black box Scenario

We have only $\{(x_0, f(x_0)), (x_1, f(x_1)), \ldots\}$ given by an "oracle"
No information is either not available or needed on the definition of objective function

- Objective function given by a computation, or a simulation
- Objective function can be irregular, non differentiable, non continous, etc.
- (Very) large search space for discrete case (combinatorial optimization), *i.e.* NP-complete problems
- Continuous problem, mixt optimization problem

Real-world applications

Typical applications

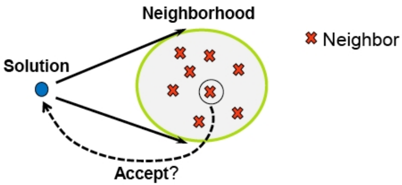
- Large combinatorial problems:
  Scheduling problems, planing problems, DOE,
  "mathematical" problems (Firing Squad Synchronization Pb.), etc.
- Calibration of models:
  Physic world ⇒ Model(params) ⇒ Simulator(params)
  $$\text{Model(Params)} = \mathrm{argmin}_M \, \mathrm{Error}(Data, M)$$
- Shape optimization:
  Design (shape, parameters of design)
  using a model and a numerical simulator

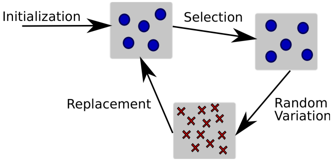
Search algorithms

Principle

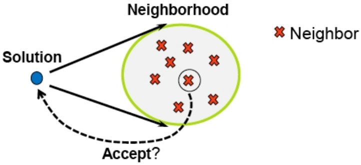
Enumeration of the search space

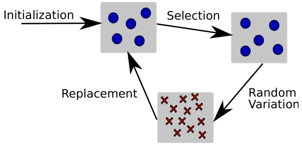
- A lot of ways to enumerate the search space
- Using random sampling: Monte Carlo technics
- Local search technics:





Search algorithms





- **Single solution-based**: Hill-climbing technics, Simulated-annealing, tabu search, Iterative Local Search, etc.
- **Population solution-based**: Genetic algorithm, Genetic programming, ant colony algorithm, etc.

Design components are well-known

- Probability to decrease,
- Memory of path, of sub-space
- Diversity of population, etc.

Research question: Parameters tuning

- One Evolutionary Algorithm key point:
  Exploitation / Exploration tradeoff
- One main practical difficulty:
  Choose operators, design components, value of parameters, representation of solutions
- Parameters setting (Lobo et al. 2007):
  - Off-line before the run: *parameter tuning*,
  - On-line during the run: *parameter control*.

One practical and theoretical question

How to combine correctly the design components according to the problem (in distributed environment...) ?

Research question: Expensive optimization

- Objective function based on a simulation:
  Expensive computation time
- One main practical difficulty:
  With few computation evaluation, choose operators, design components, value of parameters, ...
- Two main approaches:
  - Approximate objective function: *surrogate model*,
  - Parallel computation: *distributed computing*.

One practical and theoretical question

How to combine correctly the design components
with low computational budget
according to the problem in distributed environment... ?

How to solve a multi-criterium problem

Think about the decision problem!

1. Define decision variables
2. Define objective functions (criteria)
3. Define your goal: *a priori*, or *a posteriori*
4. Use an (optimization) algorithm
5. Analyze the result

A priori goal

A priori decision

Decision maker knows what he/she wants before optimization

Weighted sum

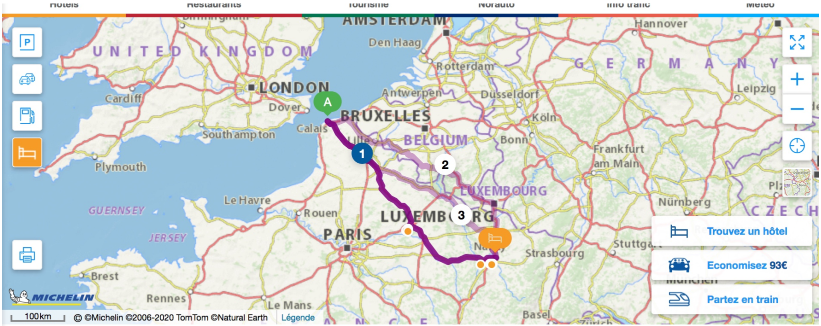
$$f_\lambda(x) = \lambda_1 f_1(x) + \ldots + \lambda_m f_m(x)$$

with $\lambda_i > 0$

- Basic model
- Often used technique
- Convert a multiobjective problem into a single-objective problem
- The definition, and the interpretation are not always straitforward

Small example

Road trip between Calais and Nancy



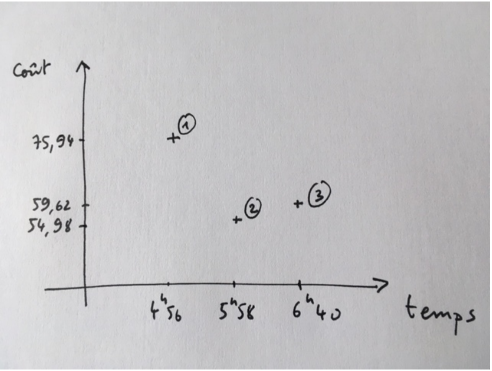
3 itinéraires possibles : *Calais > Nancy*

1 Via A26	**2 Via A4 A31**	**3 Via A26 A31**
04h56 479 km	05h58 510 km	06h40 474 km
75,94 €	**54,98 €**	**59,62 €**

Which one is better ?

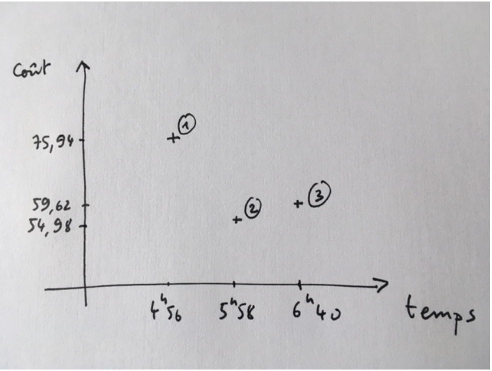
Small example

Road trip between Calais and Nancy



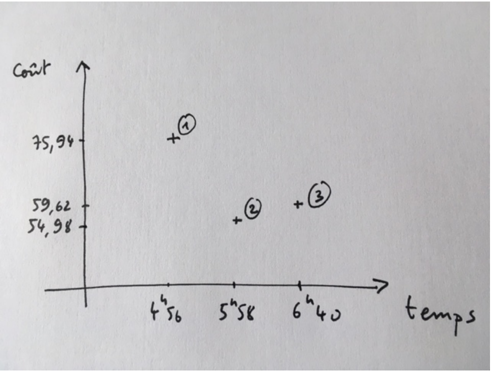
Small example

Road trip between Calais and Nancy



- According to time objective, 1 is better
- According to cost objective, 2 is better
- But, 2 is better than 3 for both objectives.

Pareto dominance



- 1 and 2 are incomparable
- 1 and 3 are incomparable
- 2 is better than 3

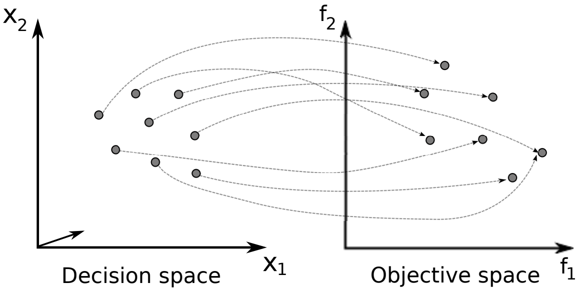
Pareto dominance

- 2 dominates 3
- 3 is dominated by 2

Multiobjective optimization

Multiobjective optimization problem

- $\mathcal{X}$: set of feasible solutions in the decision space
- $M \geqslant 2$ objective functions $f = (f_1, f_2, \ldots, f_M)$ (to maximize)
- $\mathcal{Z} = f(\mathcal{X}) \subseteq \mathbb{R}^M$: set of feasible outcome vectors in the objective space

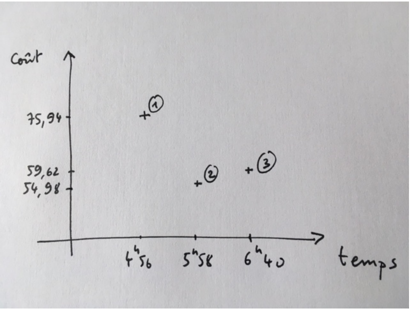


Pareto dominance definition

Pareto dominance relation (maximization)

A solution $x \in \mathcal{X}$ dominates a solution $x' \in \mathcal{X}$ ($x' \prec x$) iff

- $\forall i \in \{1, 2, \ldots, M\}$, $f_i(x') \leqslant f_i(x)$
- $\exists j \in \{1, 2, \ldots, M\}$ such that $f_j(x') < f_j(x)$

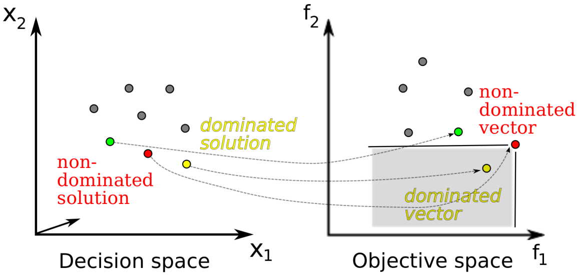


Pareto Optimale solution

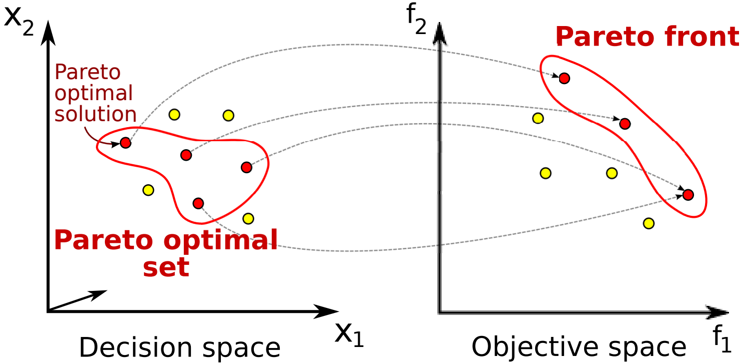
Definition: non-dominated solution

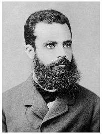
A solution $x \in \mathcal{X}$ is non-dominated (or Pareto optimal, efficient) iff

$$\forall x' \in \mathcal{X} \setminus \{x\},\ x \nprec x'$$



Pareto set, Pareto front



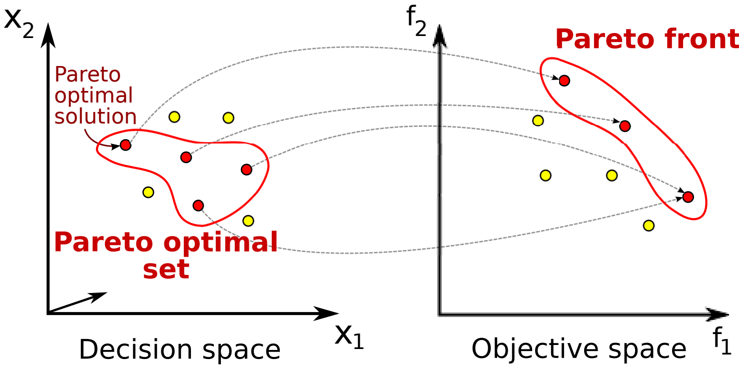


Vilfredo Pareto (1848 - 1923)
source: wikipedia

Multiobjective optimization goal

Goal

Find the Pareto Optimal Set,
or a good approximation of the Pareto Optimal Set
And not a single solution for a single aggregated objective



Challenges

- **Search space**:
  many variables, heterogeneous, dependent variables
- **Objective space**:
  many, heterogenous, expensive objective functions
- **NP-completeness**:
  deciding if a solution is Pareto optimal is difficult
- **Intractability**:
  number of Pareto optimal solutions grows exponentially with problem dimension

Methodology

Typical methodology with MO optimization

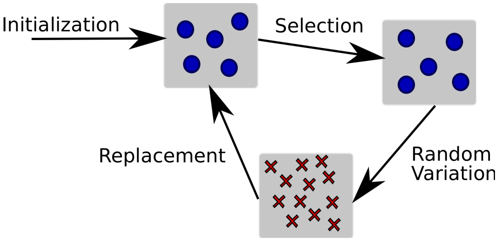
1. Define **decision variables**
2. Define all potential **objective**
3. Define **constraints** (hard/soft/objective)
4. Choose/design a relevant multiobjective **algorithm**
5. Search for an approximation of **Pareto optimal** solutions set
6. **Analyse/visualize** the solutions set

Loop between 1 to 6...

Multi-objective optimization algorithms

Population-based algorithm

A Multi-Objective (MO) algorithm is an Evolutionary Algorithm : the goal is to find a set of solutions



Evolutionary Multi-Objective (EMO) algorithm

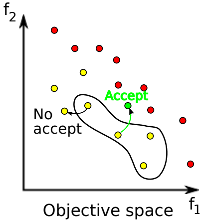
Main types of MO algorithms

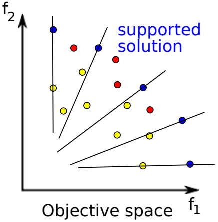
Three main classes:

(1) Pareto-based approaches: directly or indirectly focus the search on the Pareto dominance relation.
Pareto Local Search (PLS), Global SEMO, NSGA-II, etc.

(2) Indicator approaches: Progressively improvement the indicator function: IBEA, SMS-MOEA, etc.

(3) Scalar approaches: multiple scalarized aggregations of the objective functions: MOEA/D, etc.





(1) Pareto-based approaches

EMO based on dominance relation to update set of solutions (archive)

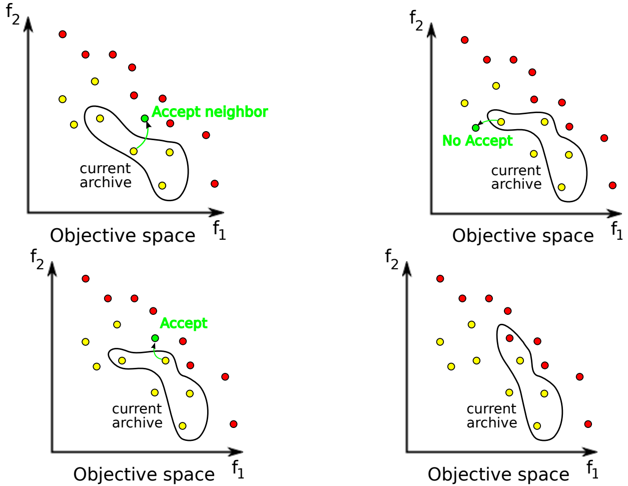
example of: Pareto Local Search (PLS)

Pick a random solution $x_0 \in X$
$A \leftarrow \{x_0\}$
repeat
  Select a non-visited $x \in A$
  Create neighbors $N(x)$ by flipping each bit of x in turns
  Flag x as visited
  $A \leftarrow$ non-dominated sol. from $A \cup N(x)$
until all-visited $\vee$ maxeval

[Paquete *et al.* 2004][9]

A Pareto-based approach: Pareto Local Search

- Archive solutions using **Dominance relation**
- Iteratively improve this archive by exploring the neighborhood



Pareto-based approaches : G-SEMO

local search:
Pareto Local Search (PLS)

Pick a random solution $x_0 \in X$
$A \leftarrow \{x_0\}$
repeat
  Select a non-visited $x \in A$
  Create $N(x)$ by flipping each bit of x in turns
  Flag x as visited
  $A \leftarrow$ non-dom. from $A \cup N(x)$
until all-visited $\vee$ maxeval

[Paquete *et al.* 2004][9]

global search:
Global-Simple EMO (G-SEMO)

Pick a random solution $x_0 \in X$
$A \leftarrow \{x_0\}$
repeat
  Select $x \in A$ at random
  Create x' by flipping each bit of x with a rate $1/N$
  $A \leftarrow$ non-dom. from $A \cup \{x'\}$
until maxeval

[Laumanns *et al.* 2004][6]

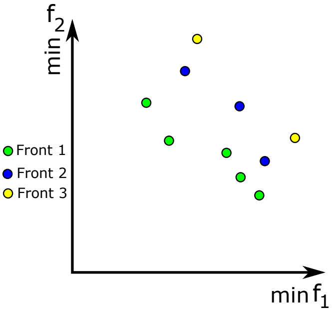
A Pareto-based approach: NSGA-II (Deb et al. 2000)

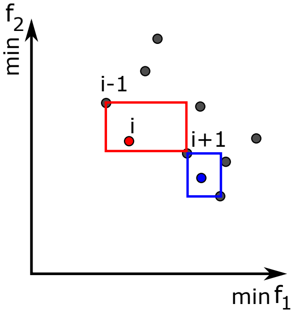
- No archive of solutions
- Classical EA based on crowding distance
- Replacement: elitist based on non-dominated sorting, and crowding distance

Evolutionary Algorithm (EA)

repeat
  selection(pop, children)
  random_variation(children)
  replacement(pop, children)
until stoping_criterium(pop)

NSGA-II: non-dominated sorting, crowding distance





- Selection:
  binary tournament using sorting, and crowding distance
- Random variation:
  crossover, mutation, etc.
- Replacement:
  elitist based on non-dominated sorting, and crowding distance

NSGA-II: non-dominated sorting, crowding distance

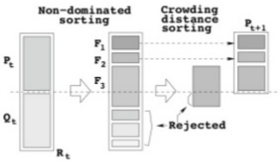


Figure: Non-dominated Selection [Deb et al. 2000]

+

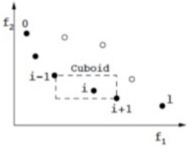


Figure: Crowding distance [Deb et al. 2000]

- Selection:
  binary tournament using sorting, and crowding distance
- Random variation:
  crossover, mutation, etc.
- Replacement:
  elitist based on non-dominated sorting, and crowding distance

(2) Indicator-based approches

Single objective optimization at population level :

- Associate one indicator (scalar value) to each population
- Optimization of this indicator

Possible indicators: hypervolume, epsilon-indicator, etc.

SMS-MOEA: $\mathcal{S}$ metric selection-MOEA [Beume *et al.* 2007][1]

$P \leftarrow$ initialization()
repeat
  $q \leftarrow$ Generate(P)
  $P \leftarrow$ Reduce($P \cup \{q\}$)
until maxeval

Generate

Use random variation (mutation, etc.) to create one candidate solution

Reduction

Remove the worst solution according to non-dominated sorting, and $\mathcal{S}$ metric

Algorithm 2. Reduce(Q)

1: $\{\mathcal{R}_1, \ldots, \mathcal{R}_v\} \leftarrow$ fast-nondominated-sort(Q)   /* all v fronts of Q */
2: $r \leftarrow \text{argmin}_{s \in \mathcal{R}_v}[\Delta_{\mathcal{S}}(s, \mathcal{R}_v)]$ /* $s \in \mathcal{R}_v$ with lowest $\Delta_{\mathcal{S}}(s, \mathcal{R}_v)$ */
3: **return** $(Q \setminus \{r\})$ /* eliminate detected element */

A $\mathcal{S}$-metric is an indicateur such hypervolume

IBEA: Indicator-Based Evolutionary algorithm

[Zitzler *et al.* 2004][12]

$P \leftarrow$ initialization()
repeat
  $P' \leftarrow$ selection(P)
  $Q \leftarrow$ random_variation(P')
  Evaluation of Q
  $P \leftarrow$ replacement(P, Q)
until maxeval

Fitness assignment

- Pairwise comparison of solutions in a population w.r.t. indicator i
- Fitness value: "loss in quality" in the population P if x was removed

$$f(x) = \sum_{x' \in P \setminus \{x\}} (-e^{-i(x',x)/\kappa})$$

- Often the ϵ-indicator is used

(3) Decomposition based approaches: MOEA/D

See the next section

(3) Decomposition based approaches: MOEA/D

Principe

Divide the multi-objective problem
    into several single-objective sub-problems

Cooperation
    between different single-objective sub-problems

Original MOEA/D [11] (minimization)

```
/* μ sub-problems defined by μ directions */
(λ^1, ..., λ^μ) ← initialization_direction()
Initialize ∀i = 1..μ B(i) the neighboring sub-problems of sub-problem i
/* one solution for each sub-problem */
(x^1, ..., x^μ) ← initialization_solution()
repeat
   for i = 1..μ do
      Select x and x' randomly in {x_j : j ∈ B(i)}
      y ← mutation_crossover(x, x')
      for j ∈ B(i) do
         if g(y|λ_j, z*_j) < g(x_j|λ_j, z*_j) then
            x_j ← y
         end if
      end for
   end for
until max_eval
```

$B(i)$ is the set of the T closest neighboring sub-problems of sub-problem i

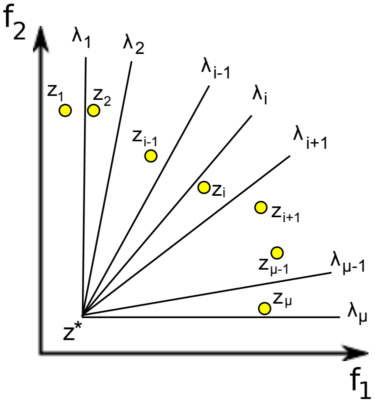
$g(\ |\lambda_i, z_i^\star)$: scalar function of sub-pb. i with λ_i direction, and $z_i^\star$ reference point

MOEA/D steady-state variant

Another MOEA/D (minimization)

/* μ *sub-problems defined by* μ *directions* */
$(\lambda^1, \ldots, \lambda^\mu) \leftarrow$ initialization_direction()
Initialize $\forall i = 1..\mu$ $B(i)$ the neighboring sub-problems of sub-problem i
/* *one solution for each sub-problem* */
$(x^1, \ldots, x^\mu) \leftarrow$ initialization_solution()
repeat
  Select i at random $\in 1..\mu$
  Select x randomly in $\{x_j \ : \ j \in B(i)\}$
  $y \leftarrow$ mutation_crossover(x_i, x)
  for $j \in B(i)$ **do**
    if $g(y|\lambda_j, z_j^\star) < g(x_j|\lambda_j, z_j^\star)$ **then**
      $x_j \leftarrow y$
    end if
  end for
until max_eval

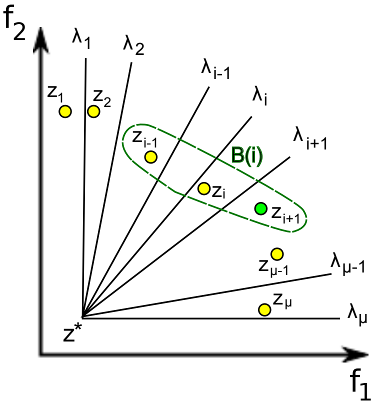
Representation of steady-state MOEA/D



Population at iteration t

- Minimization problem
- One solution x_i for each sub pb. i
- Representation of solutions in objective space: $z_i = g(x_i|\lambda_i, z_i^\star)$
- Same reference point for all sub-pb. $z^\star = z_1^\star = \ldots = z_\mu^\star$
- Scalar function g:
  Weighted Tchebycheff
- Neighborhood size $\sharp B(i) = T = 3$

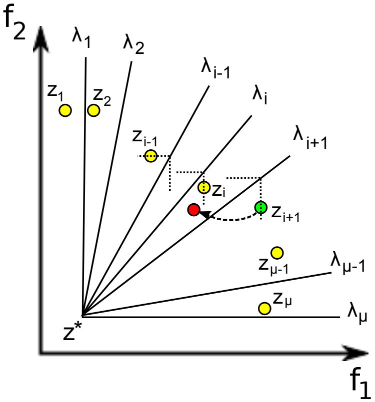
Representation of steady-state MOEA/D



From the neigh. $B(i)$ of sub-pb. i, x_{i+1} is selected

- Minimization problem
- One solution x_i for each sub pb. i
- Representation of solutions in objective space: $z_i = g(x_i|\lambda_i, z_i^\star)$
- Same reference point for all sub-pb. $z^\star = z_1^\star = \ldots = z_\mu^\star$
- Scalar function g: Weighted Tchebycheff
- Neighborhood size $\sharp B(i) = T = 3$

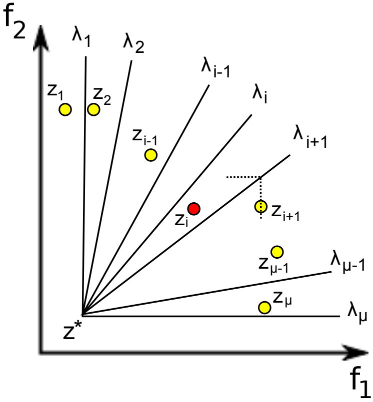
Representation of steady-state MOEA/D



The mutated solution y is created

- Minimization problem
- One solution x_i for each sub pb. i
- Representation of solutions in objective space: $z_i = g(x_i|\lambda_i, z_i^\star)$
- Same reference point for all sub-pb. $z^\star = z_1^\star = \ldots = z_\mu^\star$
- Scalar function g:
  Weighted Tchebycheff
- Neighborhood size $\sharp B(i) = T = 3$

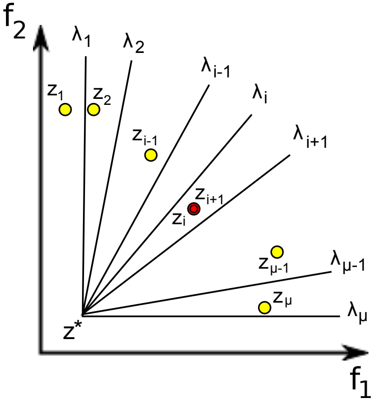
Representation of steady-state MOEA/D



According to scalar fonction,
y is worst than x_{i-1},
y is better than x_i and replaces it.

- Minimization problem
- One solution x_i for each sub pb. i
- Representation of solutions in objective space: $z_i = g(x_i | \lambda_i, z_i^\star)$
- Same reference point for all sub-pb. $z^\star = z_1^\star = \ldots = z_\mu^\star$
- Scalar function g: Weighted Tchebycheff
- Neighborhood size $\sharp B(i) = T = 3$

Representation of steady-state MOEA/D



According to scalar fonction,
y is also better than x_{i+1}
and replaces it for the next iteration.

- Minimization problem
- One solution x_i for each sub pb. i
- Representation of solutions in objective space: $z_i = g(x_i | \lambda_i, z_i^\star)$
- Same reference point for all sub-pb. $z^\star = z_1^\star = \ldots = z_\mu^\star$
- Scalar function g:
  Weighted Tchebycheff
- Neighborhood size $\sharp B(i) = T = 3$

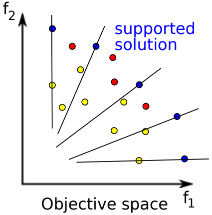
Decomposition based approaches: MOEA/D

Main issues

1. Impact of the scalar function:
   [Derbel *et. al.*, 2014] [2]
2. Direction of search:
   cf.[Derbel *et. al.*, 2014] [3]
3. Cooperation between sub-problems:
   [Gauvain *et al.*, 2014] [8]
4. Parallelization:
   cf. algorithm of "A fine-grained message passing MOEA/D" [Derbel *et al.*, 2015] [4]
   cf. [Drouet *et al.*, 2021] [5]

Scalar approaches: scalarizing function

- multiple scalarized aggregations of the objective functions



Different aggregations

- Weighted sum:

$$g(x|\lambda) = \sum_{i=1..m} \lambda_i f_i(x)$$

- Weighted Tchebycheff:

$$g(x|\lambda, z) = \max_{i=1..m} \{\lambda_i |z_i - f_i(x)|\}$$

MOEA/D-DE [7]

For solving numerical (continuous) optimization problems that combines

- Multiobjective MOEA/D
- Differential Evolution (DE) for the variation operators

Reminder: DE in short

DE algorithm: EA algorithm

```
Initialize(pop)
Evaluate(pop)
repeat
   Mutation(pop, offsprings)
   Xover(pop, offsprings)
   Evaluate(offsprings)
   Replace(pop, offsprings)
until not continue(pop)
```

DE operators

Mutation: Rand/1

For each element i of the population:

```
mutant[i] = pop[r1] + F * (pop[r2] - pop[r3])
```

with i, r1, r2, r3 four different indices with r1, r2, r3 random and $F \in [0, 2]$ a parameter (mutation factor)

Crossover

For each element i of the population:

```
jrand = random(0, d)
for(unsigned j = 0; j < d; j++)
  if (j = jrand or rnd() < CR)
        offspring[i][j] = mutant[i][j] ;
  else
        offspring[i][j] = parents[i][j] ;
```

with $CR \in [0, 1]$ a parameter (crossover rate)

Replacement

```
  if (offsprings[i] is better than parents[i])
        parents[i] = offsprings[i] ;
```

Algorithm MOEA/D-DE from [10]

$t \leftarrow 1$, initialize the population $\boldsymbol{P} = \{\boldsymbol{x}^1, ..., \boldsymbol{x}^\mu\}$;
for $i \in \{1, ..., \mu\}$ **do**
  Set the neighborhood index list $\boldsymbol{B}^i = \{i_1, ..., i_T\}$;
while The termination criteria are not met **do**
  for $i \in \{1, ..., \mu\}$ **do**
    if $\text{rand}[0,1] \leq \delta$ **then**
      $\boldsymbol{R} \leftarrow \boldsymbol{B}^i$;
    else
      $\boldsymbol{R} \leftarrow \{1, ..., \mu\}$;
    Select parent indices from $\boldsymbol{R}$ with an index selection method (Subsection 3.2);
    Generate the mutant vector $\boldsymbol{v}^i$ using a mutation strategy (Subsection 3.1);
    if $\boldsymbol{v}^i \notin \mathbb{S}$ **then**
      Repair $\boldsymbol{v}^i$ using a bound-handling method (Subsection 3.3);
    Generate the child $\boldsymbol{u}^i$ by crossing $\boldsymbol{x}^i$ and $\boldsymbol{v}^i$;
    Apply a GA mutation operator to $\boldsymbol{u}^i$;
    $c \leftarrow 1$;
    while $c \leq n^{\text{rep}}$ and $\boldsymbol{R} \neq \emptyset$ **do**
      Randomly select an index j from $\boldsymbol{R}$, and $\boldsymbol{R} \leftarrow \boldsymbol{R} \setminus \{j\}$;
      if $g(\boldsymbol{u}^i | \boldsymbol{w}^j, \boldsymbol{z}^*) \leq g(\boldsymbol{x}^j | \boldsymbol{w}^j, \boldsymbol{z}^*)$ **then**
        $\boldsymbol{x}^j \leftarrow \boldsymbol{u}^i$, $c \leftarrow c + 1$;
  $t \leftarrow t + 1$;

Nicola Beume, Boris Naujoks, and Michael Emmerich.
Sms-emoa: Multiobjective selection based on dominated hypervolume.
European Journal of Operational Research, 181(3):1653–1669, 2007.

Bilel Derbel, Dimo Brockhoff, Arnaud Liefooghe, and Sébastien Verel.
On the impact of multiobjective scalarizing functions.
In *Parallel Problem Solving from Nature–PPSN XIII*, pages 548–558. Springer, 2014.

Bilel Derbel, Jérémie Humeau, Arnaud Liefooghe, and Sébastien Verel.
Distributed localized bi-objective search.
European Journal of Operational Research, 239(3):731–743, 2014.

Bilel Derbel, Arnaud Liefooghe, Gauvain Marquet, and El-Ghazali Talbi.
A fine-grained message passing moea/d.
In *Evolutionary Computation (CEC), 2015 IEEE Congress on*, pages 1837–1844. IEEE, 2015.

V. Drouet, J.-M. Do, and S. Verel.
OPTIMIZATION OF LOAD-FOLLOW OPERATIONS OF A 1300MW PRESSURIZED WATER REACTOR USING EVOLUTIONNARY ALGORITHMS.
Gecco Conference, 2021.

Marco Laumanns, Lothar Thiele, and Eckart Zitzler.
Running time analysis of evolutionary algorithms on a simplified multiobjective knapsack problem.
Nat Comput, 3(1):37–51, 2004.

Bo Liu, Francisco V Fernández, Qingfu Zhang, Murat Pak, Suha Sipahi, and Georges Gielen.

An enhanced moea/d-de and its application to multiobjective analog cell sizing.
In *IEEE Congress on Evolutionary Computation*, pages 1–7. IEEE, 2010.

Gauvain Marquet, Bilel Derbel, Arnaud Liefooghe, and El-Ghazali Talbi.
Shake them all!
In *Parallel Problem Solving from Nature–PPSN XIII*, pages 641–651. Springer, 2014.

L. Paquete, M. Chiarandini, and T. Stützle.
Pareto local optimum sets in the biobjective traveling salesman problem: An experimental study.
In *Metaheuristics for Multiobjective Optimisation*, volume 535 of *Lecture Notes in Economics and Mathematical Systems*, chapter 7, pages 177–199. Springer, 2004.

Ryoji Tanabe and Hisao Ishibuchi.

Review and analysis of three components of the differential evolution mutation operator in moea/d-de.
Soft Computing, 23(23):12843–12857, 2019.

Qingfu Zhang and Hui Li.
Moea/d: A multiobjective evolutionary algorithm based on decomposition.
Evolutionary Computation, IEEE Transactions on, 11(6):712–731, 2007.

Eckart Zitzler and Simon Künzli.
Indicator-based selection in multiobjective search.
In *Parallel Problem Solving from Nature-PPSN VIII*, pages 832–842. Springer, 2004.