EVOLUTIONARY ALGORITHMS
LECTURE: UNSUPERVISED LEARNING AND EVOLUTIONARY COMPUTATION USING R

Jakob Bossek

MALEO Group, Department of Computer Science, Paderborn University, Germany

27$^{\text{th}}$ Jan, 2025

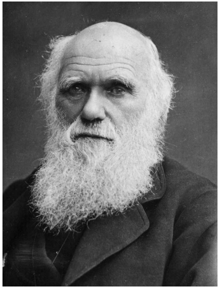
Reminder

Optimisation problems can be ...

- intractable or $\mathcal{NP}$-hard
- dynamic / stochastic / noisy
- of black-box nature
- etc.

...or any combination of the above!

$\leadsto$ **exact methods severely limited!**

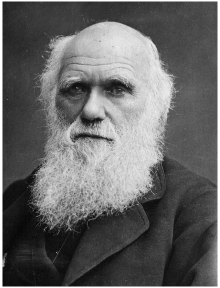


Charles Darwin (1809-1882)

"These laws, taken in the largest sense, being growth with reproduction; inheritance which is almost implied by reproduction; variability from the indirect and direct action of the external conditions of life, and from use and disuse; a ratio of increase so high as to lead to a struggle for life, and as consequence to natural selection, entailing divergence of character and the extinction of less improved forms."

Charles Darwin (1859)

Evolutionary Optimisation
Biological Inspiration

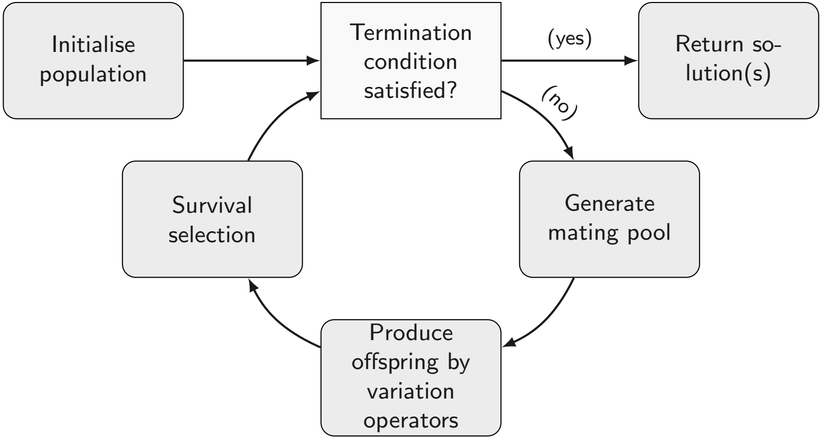


Charles Darwin (1809-1882)

"These laws, taken in the largest sense, being growth with reproduction; inheritance which is almost implied by reproduction; variability from the indirect and direct action of the external conditions of life, and from use and disuse; a ratio of increase so high as to lead to a struggle for life, and as consequence to natural selection, entailing divergence of character and the extinction of less improved forms."

Charles Darwin (1859)

Generic Evolutionary Loop



Evolutionary Algorithms
Outline

1. Vocabulary plus Simple examples of Evolutionary Algorithms (EAs)
2. Some (early) shining examples of evolutionary optimisation
3. Building blocks with focus on genetic algorithms (combinatorial optimisation)
4. Multi-objective Problems
5. Evolutionary Multi-Objective Algorithms (EMOAs)

Evolutionary Algorithms
Vocabulary

- **Individual**: a solution to our optimisation problem
- **Encoding/representation**: defines how solutions are represented by the EA
- **Population**: a multi-set[1] of individuals
- **Fitness** of an individual x: often just the objective value of x; can also be an extension of the objective function (e.g., penalising infeasible solutions)
- **Recombination**: an operator that takes (at least) two individuals x_1 and x_2 (so-called **parents**) and produces another **offspring** individual by combining the information of both parents
- **Mutation**: performs (usually small) changes to an individual x
- **Parent selection**: process of selecting individuals for mating/recombination/mutation
- **Survival selection**: process of selecting individuals that should "survive" and compose the population of the next generation

[1] It is possible that there are multiple copies of the same individual.

Some Introductory Examples of Evolutionary Algorithms

Zero-One Knapsack Problem (KP)

Given n items with positive weights w_i and positive profits p_i for $i \in [n]$ and a (knapsack) capacity limit $W > 0$. Find a subset $S^* \subseteq [n]$ such that

$$S^* = \max_{S \subseteq [n]} \sum_{i \in S} p_i \quad \text{s.t.} \quad \sum_{i \in S} w_i \leq W.$$

Example with $W = 7 \rightsquigarrow S^* = \{(1, 3\}$ with profit 7 and weight $6 < W$

i	1	2	3	4
p_i	4	2	3	2
w_i	2	2	4	5

[1] Clipart taken from `https://www.pinclipart.com/`

Evolutionary Algorithms

A Simple $(1+1)$-EA for Zero-One KP

- **Encoding:** represent solutions as bit-strings $x \in \{0,1\}^n$
  $\leadsto x_i = 1$ (item packed), $x_i = 0$ (item not packed)
- **Fitness:** make sure solutions that violate the capacity limit get fitness values lower than feasible solutions:

$$f(x) = \begin{cases} \sum_{i \in [n]} x_i \cdot p_i & \text{if } \sum_{i \in [n]} x_i \cdot w_i \leq W \\ 0 & \text{otherwise.} \end{cases}$$

- **Recombination**: generate a single individual y by bit-wise mutation of parent x
- **Population**: most basic population size of 1
- **Survival selection**: keep child y if $f(y) \geq f(x)$

Goal: $f : \{0,1\}^n \to \mathbb{R} \to \max!$

Algorithm $(1+1)$-EA

1: Generate $x \in \{0,1\}^n$ uniformly at random
2: **while** termination condition not met **do**
3: Generate y from x by flipping each bit of x independently with probability $1/n$
4: **if** $f(y) \geq f(x)$ **then** $x \leftarrow y$
5: **return** x

Goal: $f : \{0,1\}^n \to \mathbb{R} \to \max!$

Algorithm Exemplary (μ, λ)-EA

1: $P \leftarrow \{x_1, \ldots, x_\mu\}$ with $x_i \in \{0,1\}^n$ for $i \in [\mu]$
2: **while** termination condition not met **do**
3: $\quad P' \leftarrow \emptyset$
4: $\quad$ **loop** λ times
5: $\quad\quad$ Select $x \in P$ with probability $f(x)/\sum_{y \in P} f(y)$ ▷ (f-proportional selection)
6: $\quad\quad$ Let y be x with each bit-position being flipped with prob. c/n
7: $\quad\quad$ $P' \leftarrow P' \cup \{y\}$
8: $\quad P \leftarrow P'$ ▷ Keep offspring solutions only
9: **return** $\arg\max_{x \in P} f(x)$ ▷ Return best individual

---

Goal: $f : \mathbb{R}^n \to \mathbb{R} \to \min!$

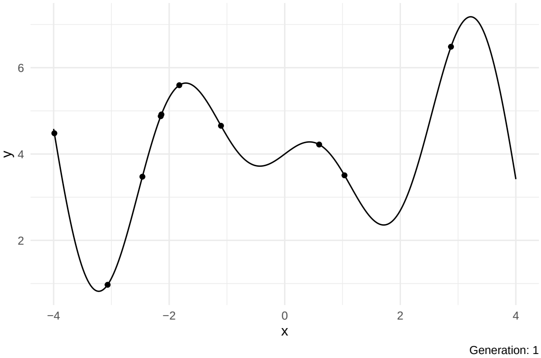
Algorithm Exemplary $(\mu + 1)$-EA

1: Initialise population $P = \{x_1, \ldots, x_\mu\}$ of μ individuals at random
2: **while** termination condition not met **do**
3: Select $x \in P$ uniformly at random ▷ (i.e., with probability $1/\mu$)
4: $y \leftarrow x + s$ with s sampled from a $\mathcal{N}(0, \Sigma)$ distribution
5: $P \leftarrow P \cup \{y\}$
6: $P \leftarrow P \setminus \{x'\}$ with $x' = \arg\max_{x \in P} f(x)$ ▷ I.e., drop worst individual
7: **return** $\arg\min_{x \in P} f(x)$ ▷ Return best individual

Evolutionary Optimisation

Example: Single-Objective Continuous Optimisation

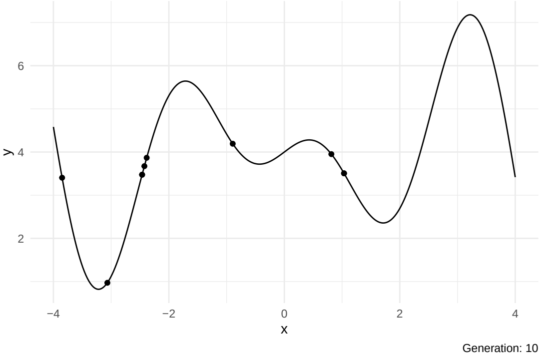
Objective function $f : \mathbb{R} \to \mathbb{R}, f(x) = x \cdot \cos(2x) + 4 \to \min!$ s. t. $x \in [-4, 4]$.
Example run of a simple $(10 + 1)$ EA:



Evolutionary Optimisation

Example: Single-Objective Continuous Optimisation

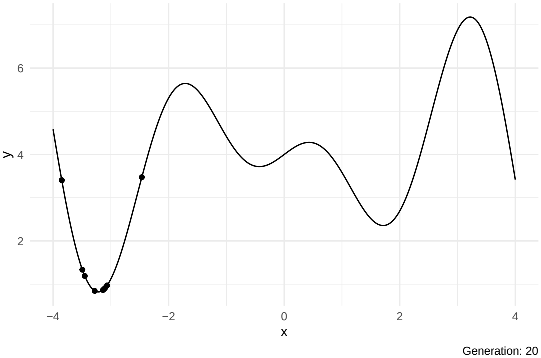
Objective function $f : \mathbb{R} \to \mathbb{R}, f(x) = x \cdot \cos(2x) + 4 \to \min!$ s. t. $x \in [-4, 4]$.
Example run of a simple $(10 + 1)$ EA:



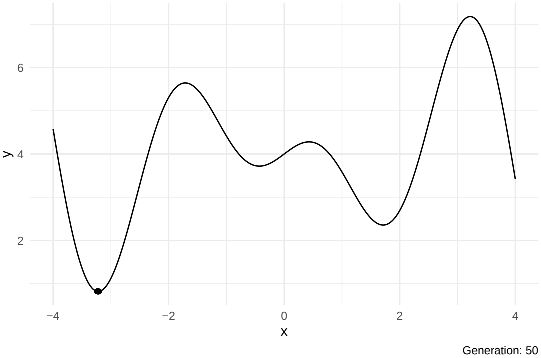
Evolutionary Optimisation

Example: Single-Objective Continuous Optimisation

Objective function $f : \mathbb{R} \to \mathbb{R}, f(x) = x \cdot \cos(2x) + 4 \to \min!$ s. t. $x \in [-4, 4]$.
Example run of a simple $(10 + 1)$ EA:



Objective function $f : \mathbb{R} \to \mathbb{R}, f(x) = x \cdot \cos(2x) + 4 \to \min!$ s. t. $x \in [-4, 4]$.
Example run of a simple $(10 + 1)$ EA:



Evolutionary Algorithms
Historical branches

Modern umbrella terms **Evolutionary {Algorithms, Optimisation}**

Evolution Strategies (ES)

- Invented by Rechenberg and Schwefel (Klockgether and Schwefel 1970) in the 1960s
- Problems typically real-valued: $f : \mathbb{R}^n \to \mathbb{R}$
- Using mutation, recombination

Evolutionary Programming (EP)

- Fogel *et al.* (Fogel, Owens, and Walsh 1966)
- Evolution of deterministic finite automata (DFA)

Genetic Algorithms (GA)

- Invented by John Holland (Holland 1975)
- Problems on bit-strings: $f : \{0, 1\}^n \to \mathbb{R}$
- Using recombination and proportional selection, little mutation

Genetic Programming (GP)

- Proposed by John Koza (Koza 1992)
- Evolve computer programs, parse trees, tree structures

Shining Examples from Engineering Applications

Optimisation of a two-phase nozzle

Often in engineering:

- Physics, chemistry, etc. of a process are unknown or (yet) not well understood
- No model or simulation software available
- **Idea:** work with real objects, i.e., realisations of the objects of interest in a lab

$\rightsquigarrow$ refine iteratively ("*hardware in the loop*")

- Early example: supersonic two-phase nozzle with turbulent flow (Klockgether and Schwefel 1970)

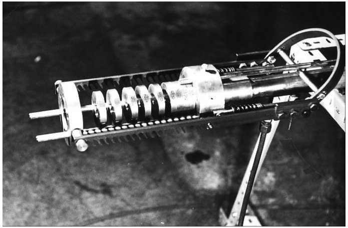
Optimisation of a two-phase nozzle
Experimental setup

- Manufacture different sizes conic nozzle parts
- Repeat the following:
  1. Choose nozzle parts are order (by an EA)
  2. Clamp conic nozzle parts (by hand in the lab)
  3. Pass steam under high pressure into nozzle and actually **measure the degree of efficiency**
  4. Replace last solution if the new one is better

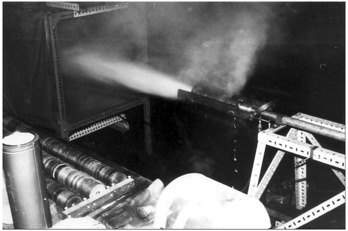
Optimisation of a two-phase nozzle

Impressions

Collection of nozzle parts



Device for clamping of nozzle parts



Measurement of nozzle efficiency

Optimisation of a two-phase nozzle

Results

Initial nozzle (left) and final nozzle (right)
(Klockgether and Schwefel 1970)

Optimisation of a two-phase nozzle

Results

Initial nozzle (left) and final nozzle (right)
(Klockgether and Schwefel 1970)

Optimisation of a two-phase nozzle
Results


Initial nozzle (left) and final nozzle (right)
(Klockgether and Schwefel 1970)

- 250 experiments/iterations with 45 improvements[2]

$\rightsquigarrow$ impressive **32% gain** in efficiency

- "The result was a rather strange nozzle contour."

[2] J. Klockgether and H. P. Schwefel (1970). "Two-phase nozzle and hollow core jet experiments". In: *Proc. 11th Symp. Engineering Aspects of Magnetohydrodynamics*, pp. 141–148.

Building Blocks

Building Block of EAs

EAs are very generic! Almost all components can be specifically designed:

1. Representation
2. Fitness function
3. Selection operators
   3.1 Parent selection
   3.2 Survival selection
4. Variation operators
   4.1 Recombination
   4.2 Mutation
5. Initialisation
6. Termination criteria

Representation / Encoding / Genotype

- Basically two levels of existence:

Representation / Encoding / Genotype

- Basically two levels of existence:
  - Phenotype: solutions in the original problem context

Representation / Encoding / Genotype

- Basically two levels of existence:
  - Phenotype: solutions in the original problem context
  - Genotype: encoding of the solution in the context of an EA

Representation / Encoding / Genotype

- Basically two levels of existence:
  - Phenotype: solutions in the original problem context
  - Genotype: encoding of the solution in the context of an EA
- Standard representation:

Representation / Encoding / Genotype

- Basically two levels of existence:
  - Phenotype: solutions in the original problem context
  - Genotype: encoding of the solution in the context of an EA
- Standard representation:
  - **Real-valued vectors**: standard in continuous optimisation (Evolution Strategies (ES))

Representation / Encoding / Genotype

- Basically two levels of existence:
  - Phenotype: solutions in the original problem context
  - Genotype: encoding of the solution in the context of an EA
- Standard representation:
  - **Real-valued vectors**: standard in continuous optimisation (Evolution Strategies (ES))
  - **Bit-strings** $x \in \{0,1\}^n$: ideal, e.g., for combinatorial problems (subset selection problems) or in the broad field of so-called *pseudo-Boolean* $f : \{0,1\}^n \to \mathbb{R}$ (Genetic Algorithms (GA))

Representation / Encoding / Genotype

- Basically two levels of existence:
  - Phenotype: solutions in the original problem context
  - Genotype: encoding of the solution in the context of an EA
- Standard representation:
  - **Real-valued vectors**: standard in continuous optimisation (Evolution Strategies (ES))
  - **Bit-strings** $x \in \{0,1\}^n$: ideal, e.g., for combinatorial problems (subset selection problems) or in the broad field of so-called *pseudo-Boolean* $f : \{0,1\}^n \to \mathbb{R}$ (Genetic Algorithms (GA))
  - **Tree structures** (Genetic Programming (GP))

Representation / Encoding / Genotype

- Basically two levels of existence:
  - Phenotype: solutions in the original problem context
  - Genotype: encoding of the solution in the context of an EA
- Standard representation:
  - **Real-valued vectors**: standard in continuous optimisation (Evolution Strategies (ES))
  - **Bit-strings** $x \in \{0,1\}^n$: ideal, e.g., for combinatorial problems (subset selection problems) or in the broad field of so-called *pseudo-Boolean* $f : \{0,1\}^n \to \mathbb{R}$ (Genetic Algorithms (GA))
  - **Tree structures** (Genetic Programming (GP))
  - Arbitrary custom, problem-dependent representations

Selection Operators

- **Parent-selection** decides which individuals to select for offspring production (usually stochastic)

Selection Operators

- **Parent-selection** decides which individuals to select for offspring production (usually stochastic)
- **Survival-selection** decides which individuals to carry over into the next generation (often deterministic)

Selection Operators

- **Parent-selection** decides which individuals to select for offspring production (usually stochastic)
- **Survival-selection** decides which individuals to carry over into the next generation (often deterministic)
- Selection takes place on the level of the population (i. e., independent of the representation)

Tournament Selection

Idea: repeated tournaments between multiple individuals

- Select a subset $P' = \{x_1, \ldots, x_l\}$ of $l \geq 2$ individuals uniformly at random from P with or without replacement
  - Select the fitness-wise best individual from P' with probability p
  - Select the 2^{nd} best individual from P' with probability $p(1-p)$
  - Select the 3^{rd} best individual from P' with probability $p(1-p)^2$
  - etc.
- Repeat k times to obtain k individuals

Tournament Selection
Selection Probabilities

Selection Probabilities depend on:

- **Rank**: individuals' ranks in the population
- **Tournament size** l: higher l-values increase the chance to include above-average individuals into P' (higher selection pressure)
- **Probability** p: usually $p = 1$ (*deterministic tournament*); lower values of p (*stochastic tournament*) decrease the selection pressure
- **Sampling**: without replacement leads to least fit $k - 1$ individuals having no chance to be selected at all. Thus, usually selection is performed with replacement.

Tournament Selection
Properties

- Invariant to translation and transposition ☺
- Applicable in settings where no objective function / quantitative measure is given
  E.g., two individuals can be compared via simulation
- Works in parallel architectures (i.e., calculation distributed on several workers)[3]
- Selection pressure is easily adjustable
- Easy to implement

[3] Here, collection fitness values – if available – is not easy or even impossible.

Variation Operators

Variation operators are responsible for the generation of new candidate solutions

- Two types depending on the number A of involved parents:
  - $A = 1$: mutation
  - $A \geq 2$: recombination
- Variation takes place at the level of an individual, i. e., variation operators are explicitly designed for certain types of EAs (such as Evolution Strategies or Genetic Algorithms)

  $\leadsto$ dependent on the chosen representation

Mutation Operators for Binary Representation
Bit-wise Mutation

Let $x \in \{0,1\}^n$ be a bit-string. **Bit-wise mutation** with mutation probability $p_m \in (0, {}^1/_2)$ produces y from x as follows:

- Make copy y of x
- Flip each bit $y_i, \in i \in [n]$ <u>independently</u> with probability p_m

$\rightsquigarrow$ often $p_m = \frac{c}{n}, c > 0$ or $c = 1$. Then the expected number of flipped bits is $n \cdot p_m = 1$

Original bit-string x		Mutated bit-string y
0 1 0 1 1 0 0 1	$\longrightarrow$	1 1 1 1 1 0 1 1

Mutation Operators for Binary Representation

Bit-wise Mutation

For $x, y \in \{0,1\}^n$ the *Hamming distance* is defined as

$$H(x,y) := \#\{i \in [n] \mid x_i \neq y_i\}.$$

Exercise: show that the following definition is equivalent:

$$H(x,y) = \sum_{i=1}^{n} (x_i + y_i - 2x_i y_i)\,.$$

Recombination for Binary Representation

Uniform Mutation

Let $x^1, x^2 \in \{0,1\}^n$ be two parent bit-strings. **Uniform mutation** generates y by setting $y_i = x_i^1$ with probability $p \in (0,1)$ and $y_i = x_i^2$ otherwise for $i \in [n]$.

Parent x^1

0	1	0	1	1	0	0	1

Parent x^2

1	1	1	0	1	1	0	0

Child y

0	1	1	0	1	1	0	1

Recombination for Binary Representation

1-Point Crossover

Let $x^1, x^2 \in \{0,1\}^n$ be two parent bit-strings. 1-**point crossover** generates y^1 and y^2 by selecting a "cut-point" $r \in [1..n-1]$, splitting both parents at position r and exchanging tails.

Parent x^1	0	1	0	1	1	0	0	1
Parent x^2	1	1	1	0	1	1	0	0
Child y^1	0	1	0	0	1	1	0	0
Child y^2	1	1	1	1	1	0	0	1

Survivor Selection

Question: How to select μ individuals out of $\mu + \lambda$?

Survival selection is typically deterministic based on

- fitness, e. g., best μ individuals
- age, e.g., the best and most recent individuals

Typical fitness-based selection methods:

- $(\mu + \lambda)$-strategy: select the best μ individuals from $P_\mu \cup P_\lambda$
- (μ, λ)-strategy: select the best μ individuals out of P_λ only!
  Requires $\lambda \geq \mu$.

Initialisation

Question: How to generate the initial population?

- Typically sampled uniformly at random over the search space.
- Can be generated from other heuristics, expert knowledge or experimental design techniques.

Termination Criteria

Question: when should the calculations stop?

- Maximum number of function evaluations (FEs), i. e., queries to the function f.
- Maximum time
- Convergence-based: very small diversity of solutions.
- Insignificant or no improvement in several consecutive generations
- Certain target fitness value reached
- In benchmarking: distance of best solution (in decision or objective space) below some threshold; i. e., solution is good enough.

Foundations of Multi-Objective Optimisation

Multi-Objective Optimisation (MOO)

So far: single-objective optimisation problems

$$f : \mathcal{X} \to \mathbb{R}, x \mapsto f(x) \in \mathbb{R}$$

Multi-Objective Optimisation (MOO)

So far: single-objective optimisation problems

$$f : \mathcal{X} \to \mathbb{R}, x \mapsto f(x) \in \mathbb{R}$$

Now: multi-objective optimisation problems

$$f : \mathcal{X} \to \mathbb{R}^p, x \mapsto f(x) = \begin{pmatrix} f_1(x) \\ f_2(x) \\ \vdots \\ f_p(x) \end{pmatrix} \in \mathbb{R}^p.$$

Multi-Objective Optimisation (MOO)

So far: single-objective optimisation problems

$$f : \mathcal{X} \to \mathbb{R}, x \mapsto f(x) \in \mathbb{R}$$

Now: multi-objective optimisation problems

$$f : \mathcal{X} \to \mathbb{R}^p, x \mapsto f(x) = \begin{pmatrix} f_1(x) \\ f_2(x) \\ \vdots \\ f_p(x) \end{pmatrix} \in \mathbb{R}^p.$$

W.l.o.g. we assume that all objectives $f_i, i \in [p] := \{1, \ldots, p\}$ are to be minimised.[4]

---

[4] This is no restriction at all since $f_i \to \max!$ is equivalent to $-f_i \to \min!$.

1st Challenge

How do we compare two vectors with respect to quality?

1^{st} Challenge

How do we compare two vectors with respect to quality?

Challenge: **Incomparibility of solutions**

$$\begin{pmatrix}1\\2\end{pmatrix} \overset{\checkmark}{\leq} \begin{pmatrix}5\\4\end{pmatrix}, \begin{pmatrix}8\\7\end{pmatrix} \overset{\checkmark}{\geq} \begin{pmatrix}4\\7\end{pmatrix}$$

1^{st} Challenge

How do we compare two vectors with respect to quality?

Challenge: Incomparibility of solutions

$$\begin{pmatrix}1\\2\end{pmatrix} \overset{\checkmark}{\leq} \begin{pmatrix}5\\4\end{pmatrix}, \begin{pmatrix}8\\7\end{pmatrix} \overset{\checkmark}{\geq} \begin{pmatrix}4\\7\end{pmatrix} \text{ but } \begin{pmatrix}9\\3\end{pmatrix} \overset{?}{\sim} \begin{pmatrix}7\\5\end{pmatrix}$$

Observation

There is no **total order** in $\mathbb{R}^p$, $p \geq 2$.

1^{st} Challenge - Solution

Let $x, y \in \mathbb{R}^p$ for $p \geq 2$ be two vectors. We say

1. x **weakly dominates** y, $x \preceq y$, iff[5]

$$\forall i \in [p] : x_i \leq y_i.$$

[5] The word *iff* is short for "if and only if".

1st Challenge - Solution

Let $x, y \in \mathbb{R}^p$ for $p \geq 2$ be two vectors. We say

1. x **weakly dominates** y, $x \preceq y$, iff[5]

$$\forall i \in [p] : x_i \leq y_i.$$

2. x **dominates** y, $x \prec y$ iff

$$\forall i \in [p] : x_i \leq y_i \text{ and } \exists i \in [p] : x_i < y_i.$$

[5] The word *iff* is short for "if and only if".

1st Challenge - Solution

Let $x, y \in \mathbb{R}^p$ for $p \geq 2$ be two vectors. We say

1. x **weakly dominates** y, $x \preceq y$, iff[5]

$$\forall i \in [p] : x_i \leq y_i.$$

2. x **dominates** y, $x \prec y$ iff

$$\forall i \in [p] : x_i \leq y_i \text{ and } \exists i \in [p] : x_i < y_i.$$

3. If neither $x \preceq y$ nor $y \preceq x$ we say that x and y are **incomparable**, denoted as $x \parallel y$.

[5] The word *iff* is short for "if and only if".

Pareto-dominance

Given $f : \mathcal{X} \to \mathbb{R}^p$ a point $x \in \mathcal{X}$ **Pareto-dominates** another point $y \in \mathcal{X}$ if

$$\forall i \in [p] : f_i(x) \leq f_i(y) \text{ and } \exists i \in [p] : f_i(x) < f_i(y).$$

Examples

Example: buying a used car

	VW	Opel	Audi	Toyota	
Price (in €)	$16k$	$14k$	$15k$	$13k$	$\rightarrow$ min!
Performance (in kW)	65	55	58	55	$\rightarrow$ max!
Fuel consumption (litre per 100km)	7.3	7.0	7.5	7.8	$\rightarrow$ min!

Example: buying a used car

	VW	Opel	Audi	Toyota	
Price (in €)	$16k$	$14k$	$15k$	$13k$	$\rightarrow$ min!
Performance (in kW)	65	55	58	55	$\rightarrow$ max!
Fuel consumption (litre per 100km)	7.3	7.0	7.5	7.8	$\rightarrow$ min!

Three (conflicting) goals/objectives, four options
$\rightsquigarrow$ what is the best alternative?

Foundations of Multi-Objective Optimisation

Definition

Multi-Objective Optimisation Problem (MOOP) Let $\mathcal{X}$ be a set (termed the **decision space**) and $f : \mathcal{X} \to F \subseteq \mathbb{R}^p$ be a vector-valued function. Then the **multi-objective optimisation problem** (MOOP) is given by

$$(f_1(x), f_2(x), \ldots f_p(x))^T \to \min!$$

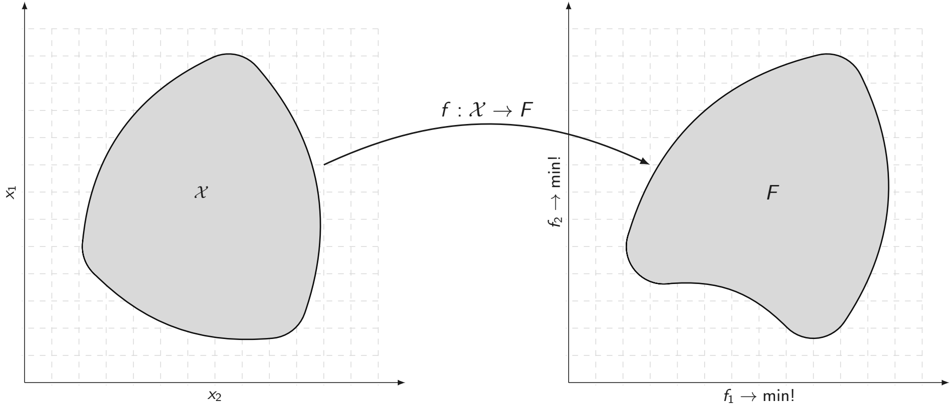
- If $f(x) \prec f(y)$ we say that $f(x)$ **dominates** $f(y)$ and x **dominates** y
- $x^* \in \mathcal{X}$ is **Pareto-optimal** iff $\nexists x \in \mathcal{X}$ with $f(x) \prec f(x^*)$
- If x^* is Pareto-optimal, $f(x^*)$ is also called **efficient**
- The set of all Pareto-optima is the **Pareto-set** (PS)

$$\mathcal{X}^* = \{x^* \in \mathcal{X} \mid \nexists x \in \mathcal{X} : f(x) \prec f(x^*)\}$$

- Its image $F^* = f(\mathcal{X}^*)$ in objective space is termed **Pareto-front** (PF)

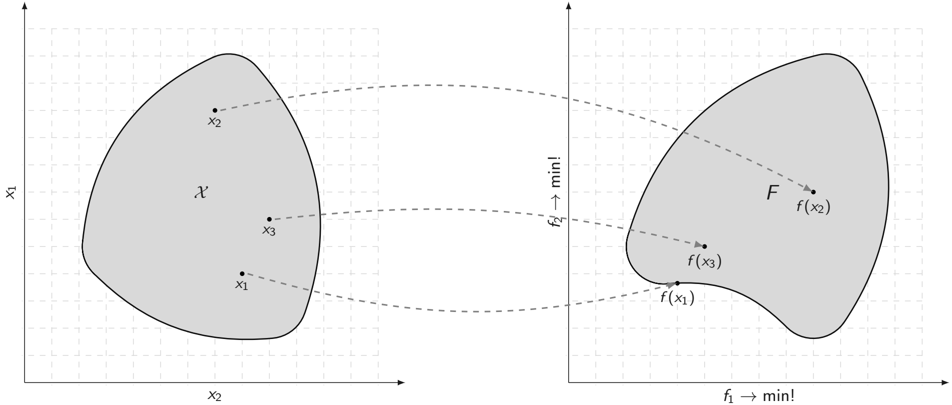
Foundations of Multi-Objective Optimisation

Visualisation of a function $f : \mathbb{R}^2 \to \mathbb{R}^2$



Foundations of Multi-Objective Optimisation

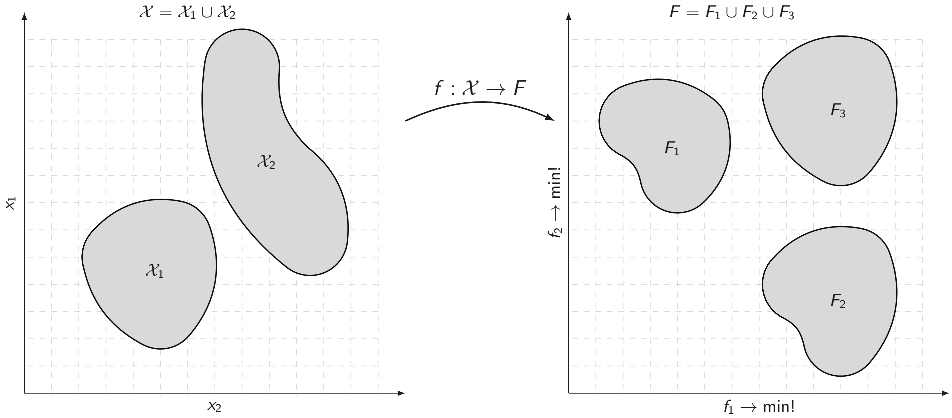
Visualisation of a function $f : \mathbb{R}^2 \to \mathbb{R}^2$



Foundations of Multi-Objective Optimisation

Visualisation of a function $f : \mathbb{R}^2 \to \mathbb{R}^2$

Note: decision and objective space not necessarily connected

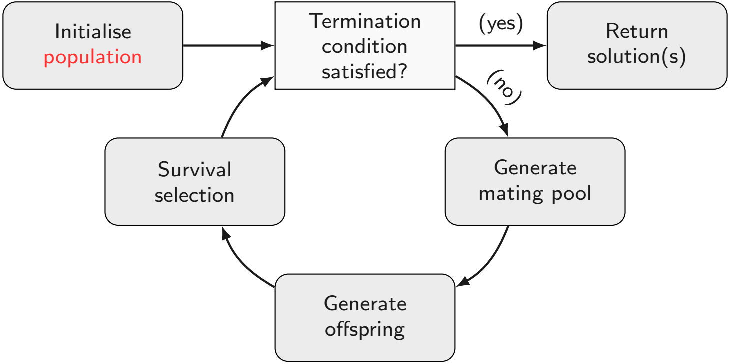


Evolutionary Multi-Objective Algorithms

EAs are state-of-the-art methods for (black-box) multi-objective problems

Evolutionary Multi-Objective Algorithms

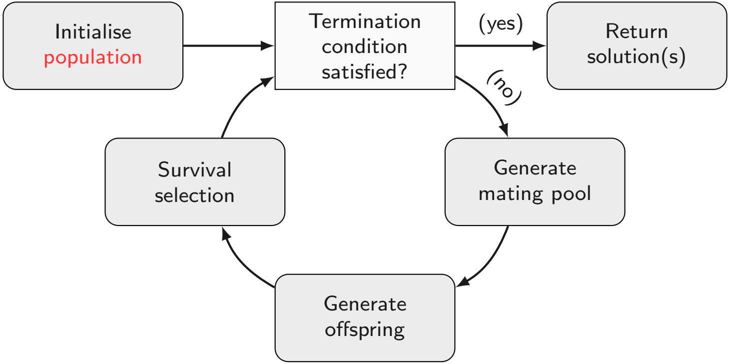
EAs are state-of-the-art methods for (black-box) multi-objective problems



[6] A multi-set may contain the same element multiple times.

Evolutionary Multi-Objective Algorithms

EAs are state-of-the-art methods for (black-box) multi-objective problems



EAs maintain a (multi-)set[6] of solutions!

[6] A multi-set may contain the same element multiple times.

Today's Learning goals

After this lecture you will understand the basics of

- Performance measurement of multi-objective randomised search heuristics
  - Recap: metrics
  - Different performance measures and their focus/(dis)advantages
- Single-point approaches ($(1+1)$ EA[7] alike)
- Population-based multi-point methods

[7] Population size 1 and one offspring produced per iteration.

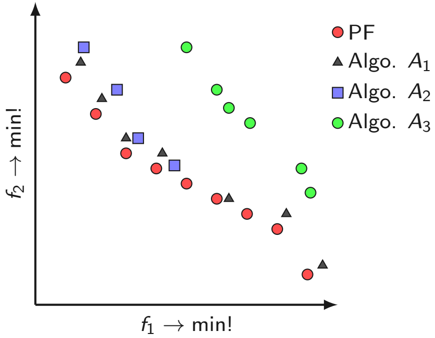
Performance Measurement

Comparison of Solution Sets

How do we compare the solutions sets obtained by multiple heuristics?

- Alg. 3 is far away from the Pareto-front
- Alg. 2 close to Pareto-front, but bad coverage
- Alg. 1 shows good coverage

$\rightsquigarrow$ Need measures to quanitfy the quality!



Comparison of Solution Sets

Why performance measures?

- Need means to compare the approximation sets of two or more algorithms

$\rightsquigarrow$ Come up with some indicator/measure

$$I : \mathbb{R}^p \to \mathbb{R}$$

(w.l.o.g. to be maximised) which establishes a total ordering between partially ordered sets

- Many evolutionary algorithms use indicators in order to make decisions on which points to keep/drop
- There is a plethora of performance measures
  $\rightsquigarrow$ for evaluation always multiple should be adopted!

Comparison of Solution Sets

A First Measure

Error Ratio

Let A be an approximation set and F^* be the Pareto-front. Then the **error ratio** is defined as

$$e(A) = \frac{|\{a \in A \mid a \in F^*\}|}{|A|}$$

Comparison of Solution Sets

A First Measure

Error Ratio

Let A be an approximation set and F^* be the Pareto-front. Then the **error ratio** is defined as

$$e(A) = \frac{|\{a \in A \mid a \in F^*\}|}{|A|}$$

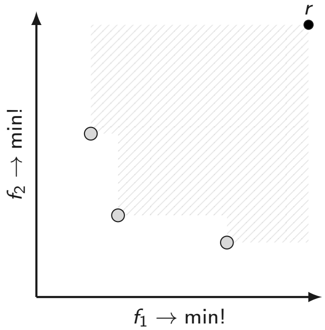
Obvious drawbacks:

- Pareto-set or Pareto-front must be known (not applicable for real-world problems)
- A set of one bad point is better than a set with close to optimal points
- Not suited to "guide" an algorithm in its search process

Let $A \subseteq \mathbb{R}^p$ be a set of points and $r \in \mathbb{R}^p$ an *anti-optimal reference point*[8]. Then the *Hypervolume-indicator* – also called the $\mathcal{S}$*-metric* – is defined as

$$\mathrm{HV}(A, r) = \lambda_p \left(\bigcup_{a \in A} [a \preceq a' \preceq r] \right).$$

where λ_p is the *p-dimensional Lebesgue measure*.



8 Anti-optimal means that the point is dominated by all points in the set A.

Comparison of Solution Sets
(Dominated) Hypervolume / $\mathcal{S}$-metric

Properties

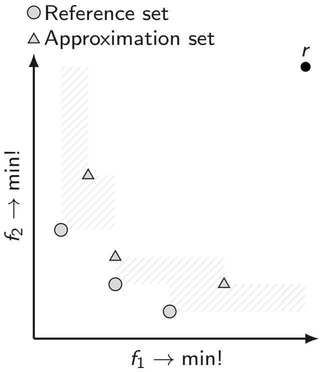
- Both in favor of convergence and spread!
- Pareto-set/-front or reference set not necessary
- So-called *soundness property* holds: if $A \preceq B$ and $A \neq B$ it follows that
$$\mathrm{HV}(A, r) > \mathrm{HV}(B, r)$$
- Drawback: sensitive to the choice of reference point $r \in \mathbb{R}^p$

Comparison of Solution Sets

Indicator: Dominated Hypervolume Difference

Given a set of points $A \subseteq \mathbb{R}^p$, a reference set $R \subseteq \mathbb{R}^p$ and an *anti-optimal reference point* $r \in \mathbb{R}^p$ the *Hypervolume-difference (indicator)* is defined as

$$\mathrm{HVD}(A, R, r) = \mathrm{HV}(R, r) - \mathrm{HV}(A, r).$$



Comparison of Solution Sets
Conclusion

- Some measures focus on spread/diversity, others on closeness to a reference set

$\rightsquigarrow$ Multi-objective performance measurement is a multi-objective problem (spread/diversity, closeness to real Pareto-front)

- Desirable properties not always fulfilled by measures

$\rightsquigarrow$ always use at least two complementary measures

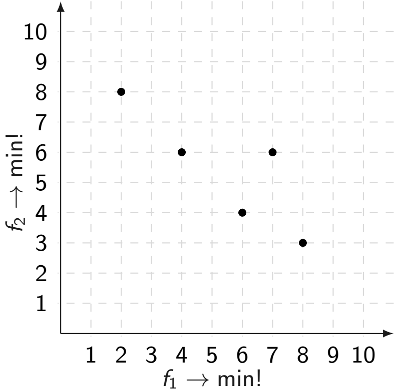
Evolutionary Multi-Objective Algorithms

Towards Non-Dominated Sorting

General Idea

$(5+5)$ EMOA

Given population P_t and offspring set Q_t how to select the 5 "best" out of $|P_t \cup Q_t| = 10$ individuals?

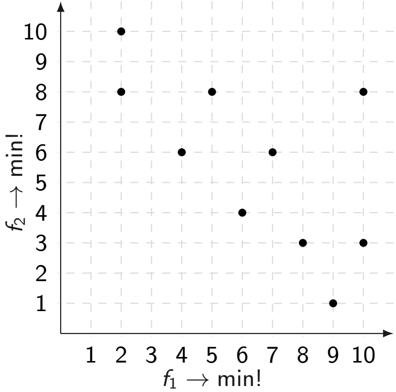


Towards Non-Dominated Sorting

General Idea

$(5+5)$ EMOA

Given population P_t and offspring set Q_t how to select the 5 "best" out of $|P_t \cup Q_t| = 10$ individuals?



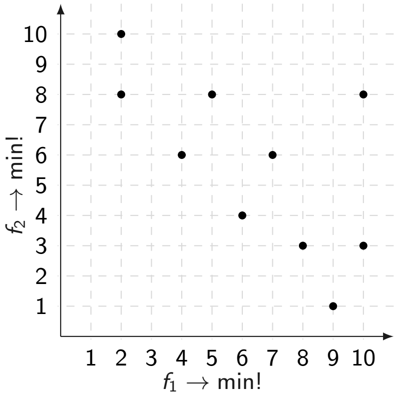
Towards Non-Dominated Sorting

General Idea

$(5+5)$ EMOA

Given population P_t and offspring set Q_t how to select the 5 "best" out of $|P_t \cup Q_t| = 10$ individuals?

Idea: keep only the non-dominated points!

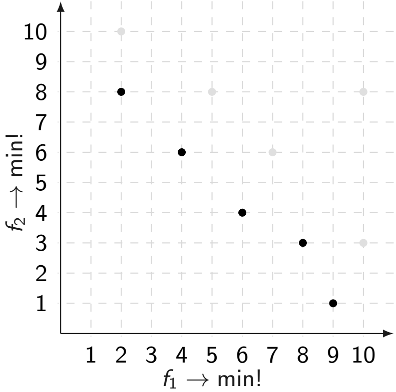


$(5+5)$ EMOA

Given population P_t and offspring set Q_t how to select the 5 "best" out of $|P_t \cup Q_t| = 10$ individuals?

Idea: keep only the non-dominated points!

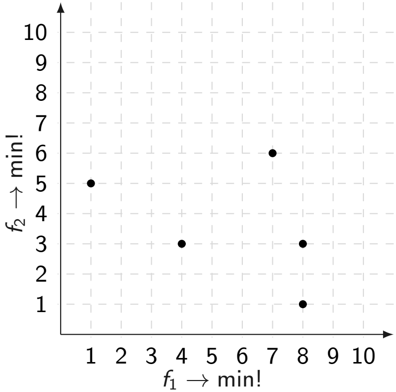
Here it works fine since $|\mathrm{ND}(P_t \cup Q_t)| = 5$
$\rightsquigarrow P_{t+1} = \mathrm{ND}(P_t \cup Q_t)$



Towards Non-Dominated Sorting
General Idea

$(5+5)$ EMOA

Given population P_t and offspring set Q_t how to select the 5 "best" individuals from $R_t = P_t \cup Q_t$?

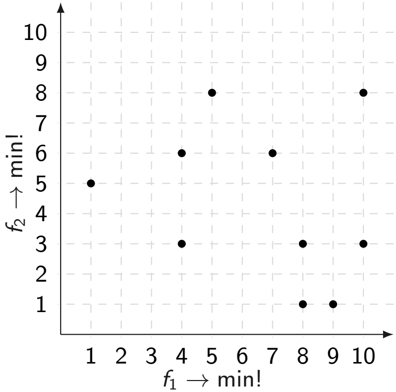


Towards Non-Dominated Sorting

General Idea

$(5+5)$ EMOA

Given population P_t and offspring set Q_t how to select the 5 "best" individuals from $R_t = P_t \cup Q_t$?

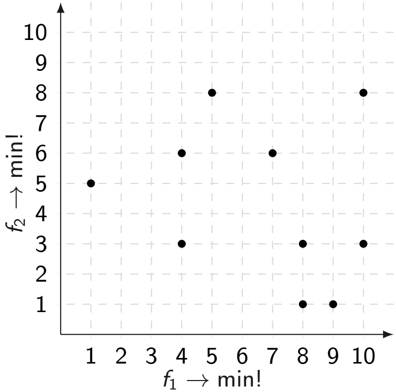


$(5+5)$ EMOA

Given population P_t and offspring set Q_t how to select the 5 "best" individuals from $R_t = P_t \cup Q_t$?

Let $F_1 = \text{ND}(R_t)$

- ⤳ We have $|F_1| = 3 < 5$
- ⤳ Drop F_1 from R_t and use $F_2 = \text{ND}(R_t \setminus F_1)$ to fill up the population
  - ▸ Here $|F_1 \cup F_2| = 5$ works fine!
- ⤳ **General idea**: use **non-domination layers** $F_1, F_2, \ldots, F_h$ until population size is reached



Towards Non-Dominated Sorting

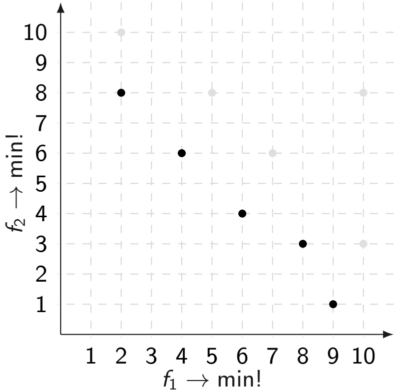
General Idea

$(5+5)$ EMOA

Given population P_t and offspring set Q_t how to select the 5 "best" individuals from $R_t = P_t \cup Q_t$?

Let $F_1 = \mathrm{ND}(R_t)$

- $\rightsquigarrow$ We have $|F_1| = 3 < 5$
- $\rightsquigarrow$ Drop F_1 from R_t and use $F_2 = \mathrm{ND}(R_t \setminus F_1)$ to fill up the population
  - Here $|F_1 \cup F_2| = 5$ works fine!
- $\rightsquigarrow$ **General idea**: use **non-domination layers** $F_1, F_2, \ldots, F_h$ until population size is reached

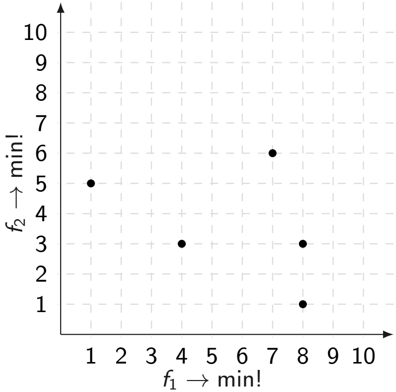


Towards Non-Dominated Sorting

General Idea

$(5+5)$ EMOA

Let $F_1, \ldots, F_h$ be the non-domination layers of $R_t = P_t \cup Q_t$.

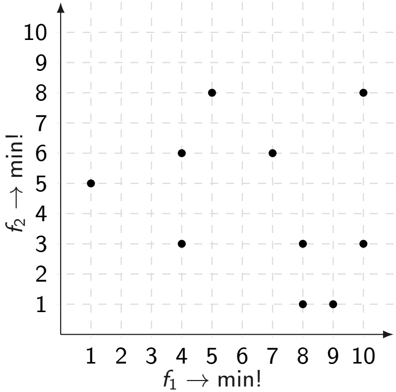


Towards Non-Dominated Sorting

General Idea

$(5+5)$ EMOA

Let $F_1, \ldots, F_h$ be the non-domination layers of $R_t = P_t \cup Q_t$.



Towards Non-Dominated Sorting

General Idea

$(5+5)$ EMOA

Let $F_1, \ldots, F_h$ be the non-domination layers of $R_t = P_t \cup Q_t$.

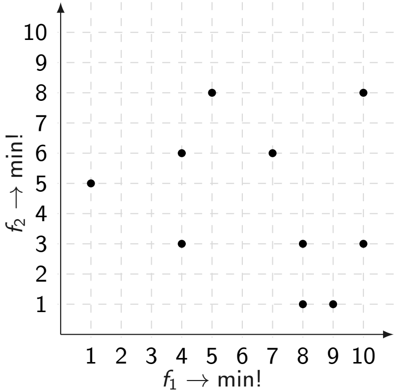
Problem: what if exists $l \in [h]$ such that

$$\left|\bigcup_{i=1}^{l-1} F_i\right| < \mu \text{ and } \left|\bigcup_{i=1}^{l} F_i\right| > \mu?$$

- Obvious: keep only subset of the layer F_l

$\rightsquigarrow$ **secondary selection criterion** necessary

- Random selection $\rightsquigarrow$ potential loss of diversity



Towards Non-Dominated Sorting
General Idea

$(5+5)$ EMOA

Let $F_1, \ldots, F_h$ be the non-domination layers of $R_t = P_t \cup Q_t$.

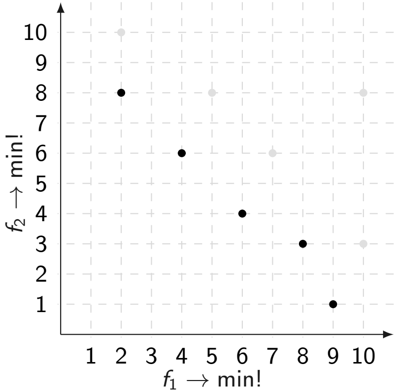
Problem: what if exists $l \in [h]$ such that

$$\left|\bigcup_{i=1}^{l-1} F_i\right| < \mu \text{ and } \left|\bigcup_{i=1}^{l} F_i\right| > \mu?$$

- Obvious: keep only subset of the layer F_l

$\rightsquigarrow$ **secondary selection criterion** necessary

- Random selection $\rightsquigarrow$ potential loss of diversity



Non-Dominated Sorting EMOA by Goldberg

Historical remark

Idea: split population P into anti-chains $F_1, \ldots, F_h$

Algorithm NDS (Goldberg, 1989)

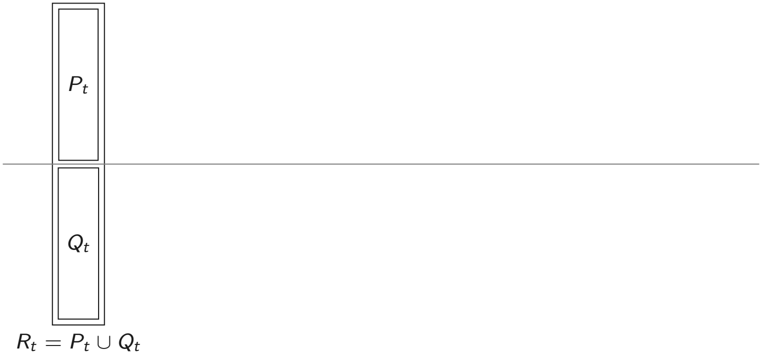
```
Initialise population P at random
while termination condition not met do
    Evaluate all individuals x ∈ P
    Let F_1, F_2, ..., F_h be a sequence of anti-chains
    Set rank(x) = i if f(x) ∈ F_i
    Perform rank-based selection
    Apply variation operators
```

Never implemented!

Non-Dominated-Sorting Genetic Algorithm-II (NSGA-II)

General idea

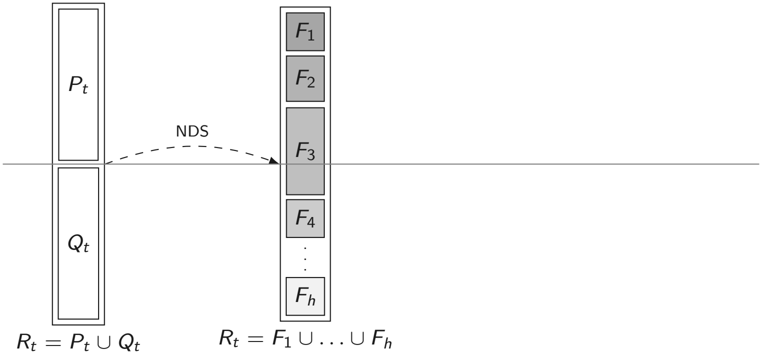
$(\mu + \mu)$-strategy with two-step survival selection:
NDS + crowding distance (K. Deb et al. 2002)



Non-Dominated-Sorting Genetic Algorithm-II (NSGA-II)

General idea

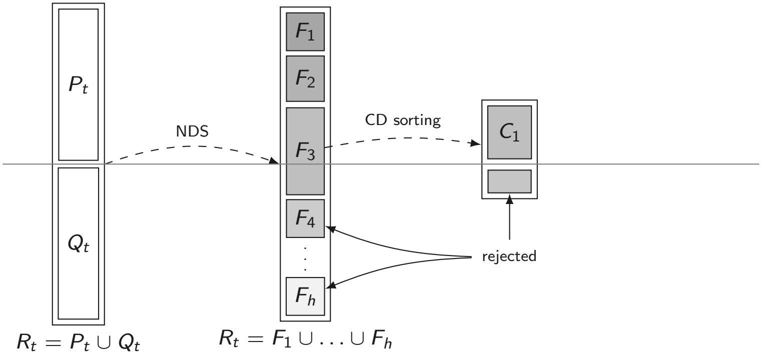
$(\mu + \mu)$-strategy with two-step survival selection:
NDS + crowding distance (K. Deb et al. 2002)



Non-Dominated-Sorting Genetic Algorithm-II (NSGA-II)

General idea

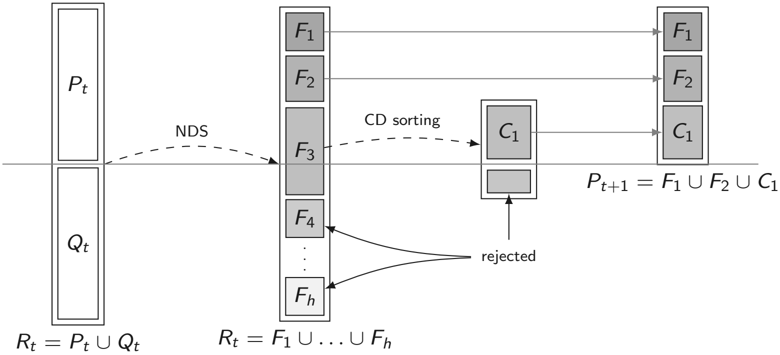
$(\mu + \mu)$-strategy with two-step survival selection:
NDS + crowding distance (K. Deb et al. 2002)



Non-Dominated-Sorting Genetic Algorithm-II (NSGA-II)

General idea

$(\mu + \mu)$-strategy with two-step survival selection:
NDS + crowding distance (K. Deb et al. 2002)



Non-Dominated-Sorting Genetic Algorithm-II

Non-dominated sorting

First key component: *non-dominated sorting* (NDS)

Algorithm Non-dominated sorting (NDS)

Require: Set of points $A \subset \mathbb{R}^p$.

$i \leftarrow 1$ $\triangleright$ Layer counter

while $A \neq \emptyset$ **do**

    $A_i \leftarrow$ set of non-dominated points in A $\triangleright = M(A, \preceq)$

    $A \leftarrow A \setminus A_i$ $\triangleright$ Remove non-dominated points

    $i \leftarrow i + 1$

return $A_1, A_2, \ldots, A_{i-1}$ $\triangleright$ Hierarchy of anti-chains

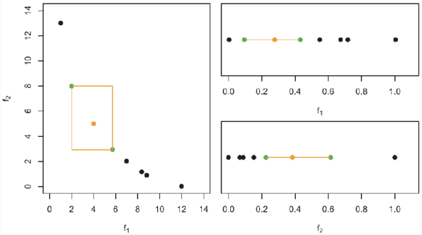
Non-Dominated-Sorting Genetic Algorithm-II

Secondary selection criterion

Second key component: *crowding distance (CD)*

$$\mathrm{CD}(x_{(i)}) = \sum_{j=1}^{p} \frac{|f_j(x_{(i+1)}) - f_j(x_{(i-1)})|}{\max_i f_j(x_i) - \min_i f_j(x_i)}$$

Measures "space" around individual solutions in which no other solutions are found.



$f_1(x)$	$f_2(x)$	$CD(x)$
1.00	13.00	∞
2.00	8.00	0.888
4.00	5.00	0.728
5.71	2.93	0.503
7.00	2.00	0.378
8.37	1.15	0.254
8.84	0.88	0.419
12.00	0.00	∞

Non-Dominated-Sorting Genetic Algorithm-II

Algorithm outline

Algorithm NSGA-II (K. Deb et al. 2002)

Require: Population P_t, set of offspring Q_t at iteration t.

$F_1, F_2, \ldots \leftarrow \mathrm{NDS}(P_t \cup Q_t)$ ▷ Sort by NDS
$P_{t+1} \leftarrow \emptyset$
$i \leftarrow 1$
while $|P_{t+1}| + |F_i| \leq \mu$ **do**
    $P_{t+1} \leftarrow P_{t+1} \cup F_i$
    $i \leftarrow i + 1$
Calculate CD for F_i and sort in descending order
Add first $(\mu - |P_{t+1}|)$ elements of F_i to P_{t+1}
Generate Q_{t+1} from P_{t+1} by variation
$t \leftarrow t + 1$

NSGA-II
Properties

- The original paper[9] has 56 342 citations at Google scholar[10]
  $\rightsquigarrow$ one of the most widely used EMOAs
- Only effective for up to three objectives
  (reasons will be presented in the section on many-objective optimisation)
- For more objectives no convergence to the Pareto-front (degenerates towards focus on diversity)
- Extension which works for more objectives: NSGA-III (Kalyanmoy Deb and Jain 2014)

---

9 K. Deb et al. (2002). "A fast and elitist multiobjective genetic algorithm: NSGA-II". In: *IEEE Transactions on Evolutionary Computation* 6.2, pp. 182–197. DOI: 10.1109/4235.996017.

10 Checked June 15th, 2024 at 11:00am.

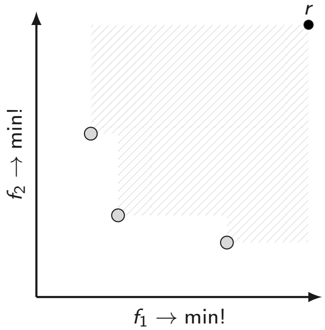
Reminder: Comparison of Solution Sets

(Dominated) Hypervolume / $\mathcal{S}$-metric

Let $A \subseteq \mathbb{R}^p$ be a set of points and $r \in \mathbb{R}^p$ an *anti-optimal reference point*[11]. Then the *Hypervolume-indicator* – also called the $\mathcal{S}$*-metric* – is defined as

$$\mathrm{HV}(A, r) = \lambda_p \left(\bigcup_{a \in A} [a \preceq a' \preceq r] \right) .$$

where λ_p is the *p-dimensional Lebesgue measure*.

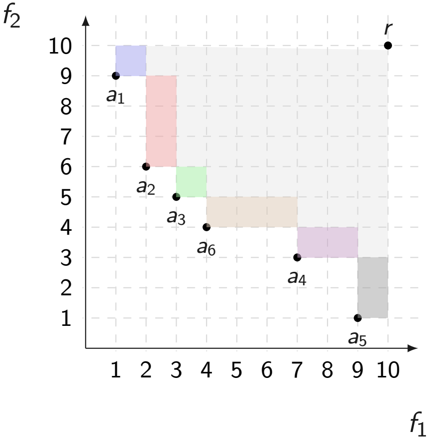


[11] Anti-optimal means that the point is dominated by all points in the set A.

Comparison of Solution Sets
Individual Hypervolume Contribution

Given a set of points $A \subseteq \mathbb{R}^p$, a reference point $r \subseteq \mathbb{R}^p$ one might ask for the **individual contribution of each** $a \in A$ to the dominated hypervolume, i.e.,

$$HVC(a; r) = HV(A, r) - HV(A \setminus \{a\}, r)$$



$\mathcal{S}$-Metric Selection EMOA (SMS-EMOA)
Core idea

- $(\mu + 1)$-strategy (*steady-state*)
- Fits the framework of *indicator-based EMOAs* (Zitzler and Künzli 2004)
- Two-step survival selection
  1. Primary criterion: *non-dominated sorting*
     $\leadsto$ Store solutions of the foremost layers until $\geq \mu$ solutions are collected.
  2. Secondary criterion (if $> \mu$ solutions selected in first phase): *hypervolume-contribution*
     $\leadsto$ drop individual with smallest HV contribution from the last layer that does not entirely fit into next population of μ individuals

$\mathcal{S}$-Metric Selection EMOA (SMS-EMOA)
Pseudo-code

Algorithm SMS-EMOA (variant 2; Beume, Naujoks, and Emmerich 2007)

```
Initialise P_0; t ← 0
while termination condition not met do
    o ← variation(P_t)                                        ▷ Generate one new solution
    F_1, ..., F_h ← NDS(P_t ∪ {o})                            ▷ Sequence of anti-chains
    Let x* be the individual with the lowest HV-contribution in F_h
    P_{t+1} ← (P_t ∪ {o}) \ {x*}
    t ← t + 1
return P_t
```

$\mathcal{S}$-Metric Selection EMOA (SMS-EMOA)

Pseudo-code (slightly more efficient)

Algorithm SMS-EMOA (variant 2; Beume, Naujoks, and Emmerich 2007)

Initialise P_0; $t \leftarrow 0$
while termination condition not met **do**
    $o \leftarrow \text{variation}(P_t)$ ▷ Generate one new solution
    $F_1, \ldots, F_h \leftarrow \text{NDS}(P_t \cup \{o\})$ ▷ Sequence of anti-chains
    if $\#F_h = 1$ **then** ▷ Case 1
        $P_{t+1} \leftarrow (P_t \cup \{o\}) \setminus F_h$
    else ▷ Case 2
        Let x^* be the individual with the lowest HV-contribution in F_h
        $P_{t+1} \leftarrow (P_t \cup \{o\}) \setminus \{x^*\}$
    $t \leftarrow t + 1$
return P_t

$\mathcal{S}$-Metric Selection EMOA (SMS-EMOA)

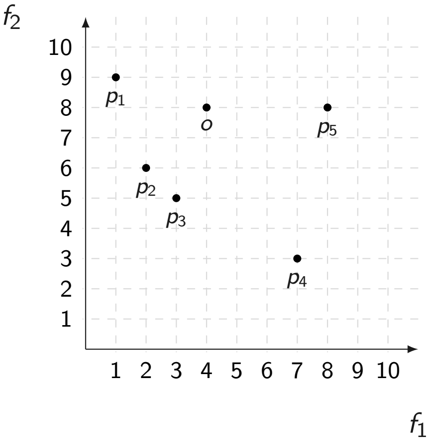
Example

$(5+1)$ **SMS-EMOA**

with $P_t = \{p_1, \ldots, p_5\}$ and single offspring o

- Non-domination layers:
  $F_1 = \{p_1, p_2, p_3, p_4\}$, $F_2 = \{o\}$ and
  $F_3 = \{p_5\}$
  $\rightsquigarrow$ Case 1 in Alg. 8
- We see $\#F_3 = 1$

$\rightsquigarrow$ Drop p_5, i.e., $P_{t+1} = \{p_1, p_2, p_3, p_4, o\}$



$\mathcal{S}$-Metric Selection EMOA (SMS-EMOA)
Example

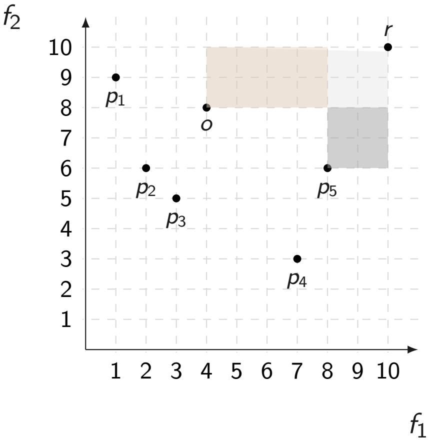
$(5+1)$ SMS-EMOA

with $P_t = \{p_1, \ldots, p_5\}$, single offspring o, and reference point $r = (10, 10)^T$

- $F_1 = \{p_1, \ldots, p_4\}$, $F_2 = \{p_5, o\}$
  $\rightsquigarrow$ Case 2 in Alg. 8

$\rightsquigarrow$ Calculate HV-contributions for elements of F_2

$\rightsquigarrow$ Drop p_5, i.e., $P_{t+1} = \{p_1, p_2, p_3, p_4, o\}$



$\mathcal{S}$-Metric Selection EMOA (SMS-EMOA)
Example

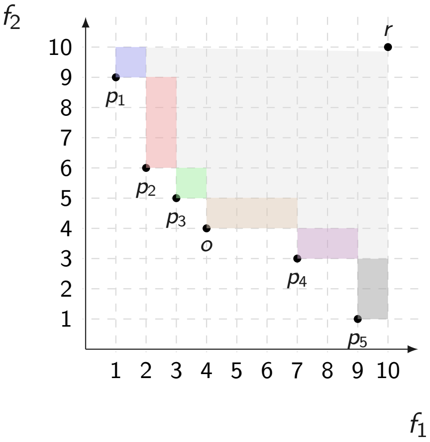
$(5+1)$ SMS-EMOA

with $P_t = \{p_1, \ldots, p_5\}$, single offspring o, and reference point $r = (10, 10)^T$

- $F_1 = P_t \cup \{o\}$ (i.e., all individuals are non-dominated)
  $\rightsquigarrow$ Case 2 in Alg. 8

$\rightsquigarrow$ Drop either p_3 or p_1 since both points' HVCs are 1

$\rightsquigarrow$ Drop p_3, i.e., $P_{t+1} = \{p_1, p_2, p_4, p_5, o\}$ (since implementations would keep the extreme solutions)



$\mathcal{S}$-Metric Selection EMOA (SMS-EMOA)
Properties

- Very effective for $p = 2$ objectives
- HV-computation is expensive: $\mathcal{O}(\mu^{p/2} \cdot \log \mu)$ for μ points an p objectives
- SMS-EMOA implements a $(\mu + 1)$-strategy
  (*steady-state* approach; $(\mu + \lambda), \lambda > 1$ doable, but super-expensive!
- HV can be approximated to improve performance

References I

Klockgether, J. and H. P. Schwefel (1970). “Two-phase nozzle and hollow core jet experiments”. In: *Proc. 11th Symp. Engineering Aspects of Magnetohydrodynamics*, pp. 141–148.

Holland, John H. (1975). *Adaptation in Natural and Artificial Systems.* second edition, 1992. Ann Arbor, MI: University of Michigan Press.

Fogel, L. J., A. J. Owens, and M. J. Walsh (1966). *Artificial Intelligence through Simulated Evolution.* John Wiley.

Koza, John R. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* Cambridge, MA, USA: MIT Press. ISBN: 0-262-11170-5.

Deb, K. et al. (2002). “A fast and elitist multiobjective genetic algorithm: NSGA-II”. In: *IEEE Transactions on Evolutionary Computation* 6.2, pp. 182–197. DOI: 10.1109/4235.996017.

Deb, Kalyanmoy and Himanshu Jain (2014). “An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point-Based Nondominated Sorting Approach, Part I: Solving Problems With Box Constraints”. In: *IEEE Transactions on Evolutionary Computation* 18.4, pp. 577–601. DOI: 10.1109/TEVC.2013.2281535.

Zitzler, Eckart and Simon Künzli (2004). “Indicator-Based Selection in Multiobjective Search”. In: *Parallel Problem Solving from Nature - PPSN VIII*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 832–842.

References II

Beume, Nicola, Boris Naujoks, and Michael Emmerich (2007). “SMS-EMOA: Multiobjective selection based on dominated hypervolume”. In: *European Journal of Operational Research* 181.3, pp. 1653–1669. ISSN: 0377-2217. DOI: `https://doi.org/10.1016/j.ejor.2006.08.008`. URL: `https://www.sciencedirect.com/science/article/pii/S0377221706005443`.