Evolutionary Computation (CS5048)

Multimodal and Multi-objective Optimisation

Edgar Covantes Osuna
edgar.covantes@tec.mx

Tecnológico de Monterrey

Contents

Contents

Optimisation and Black box

What is optimisation and black box optimisation?

Optimisation in general is the attempt to find a state of a system that is better in quality than states known beforehand.

Optimisation and Black box

What is optimisation and black box optimisation?

Optimisation in general is the attempt to find a state of a system that is better in quality than states known beforehand.

The goal is to move from one state to another with better quality until the process lead us to an optimal state, this state represents the best of all possible states.

Optimisation and Black box

What is optimisation and black box optimisation?

Optimisation in general is the attempt to find a state of a system that is better in quality than states known beforehand.

The goal is to move from one state to another with better quality until the process lead us to an optimal state, this state represents the best of all possible states.

To be optimisable, a system must have input variables whose values at least partially determine its output and which can be adjusted by the utilised optimisation method.

Optimisation and Black box

What is optimisation and black box optimisation?

Optimisation in general is the attempt to find a state of a system that is better in quality than states known beforehand.

The goal is to move from one state to another with better quality until the process lead us to an optimal state, this state represents the best of all possible states.

To be optimisable, a system must have input variables whose values at least partially determine its output and which can be adjusted by the utilised optimisation method.

If nothing is known about the system to be optimised apart from the types and the number of input and output variables, the scenario is regarded as black box optimisation.

Optimisation and Black box

Something to think about

From the previous slide, we call an improvement if from a previous state, formed by its input variables and its output, we arrive to a better new state formed by its input variables and its output and so on.

Optimisation and Black box

Something to think about

From the previous slide, we call an improvement if from a previous state, formed by its input variables and its output, we arrive to a better new state formed by its input variables and its output and so on.

- If the global optimum is unknown, how can we know we have arrived to the optimal state?

Optimisation and Black box

Something to think about

From the previous slide, we call an improvement if from a previous state, formed by its input variables and its output, we arrive to a better new state formed by its input variables and its output and so on.

- If the global optimum is unknown, how can we know we have arrived to the optimal state?
- Do you think that the time complexity of recognising that you have found/arrived to the optimal state is the same as solving the optimisation problem itself?

Optimisation and Black box

Objective Function

Let f be the objective function, this is a measure of how well the input variables are able to solve the optimisation problem.

Optimisation and Black box

Objective Function

Let f be the objective function, this is a measure of how well the input variables are able to solve the optimisation problem.

Let $x \in S^n$ be an input variable vector (search point) with size n on the search space S.

Optimisation and Black box

Objective Function

Let f be the objective function, this is a measure of how well the input variables are able to solve the optimisation problem.

Let $x \in S^n$ be an input variable vector (search point) with size n on the search space S.

Then, we assume that the set of functions values $f(x)$ is bounded from below, and we are able to compute f only with finite (known) precision, so that the wanted global optimum can be found.

Optimisation and Black box

Objective Function

Let f be the objective function, this is a measure of how well the input variables are able to solve the optimisation problem.

Let $x \in S^n$ be an input variable vector (search point) with size n on the search space S.

Then, we assume that the set of functions values $f(x)$ is bounded from below, and we are able to compute f only with finite (known) precision, so that the wanted global optimum can be found.

Given a function f it is necessary to define if in our optimisation problem we need to found the highest global optimum (maximisation problem) or the lowest global optimum (minimisation problem).

Optimisation and Black box

Objective Function

Let f be the objective function, this is a measure of how well the input variables are able to solve the optimisation problem.

Let $x \in S^n$ be an input variable vector (search point) with size n on the search space S.

Then, we assume that the set of functions values $f(x)$ is bounded from below, and we are able to compute f only with finite (known) precision, so that the wanted global optimum can be found.

Given a function f it is necessary to define if in our optimisation problem we need to found the highest global optimum (maximisation problem) or the lowest global optimum (minimisation problem).

Without loss of generality we will focus on maximisation problems.

Optimisation and Black box

Local and Global Optimum

Definition 1 (Local and Global Optima)

Given a function $f : S^n \to \mathbb{R}$, the set $\mathbb{Y}$ of local optima is given by $\mathbb{Y} := \{y \mid y \in S^n, \forall x \in S^n : \mathrm{d}(y,x) < \varepsilon \Rightarrow f(y) \geq f(x)\}$. And the global optimum x_{opt} is defined as $x_{\mathrm{opt}} := \arg\max\{f(y) \mid y \in S^n\}$.

Optimisation and Black box

Local and Global Optimum

Definition 1 (Local and Global Optima)

Given a function $f : S^n \to \mathbb{R}$, the set $\mathbb{Y}$ of local optima is given by $\mathbb{Y} := \{y \mid y \in S^n, \forall x \in S^n : \mathrm{d}(y,x) < \varepsilon \Rightarrow f(y) \geq f(x)\}$. And the global optimum x_{opt} is defined as $x_{\mathrm{opt}} := \arg\max\{f(y) \mid y \in S^n\}$.

- A local optima is any search point contained in the search space with the highest objective function value with regards to its neighbourhood.
    - The neighbourhood of a search point $y \in S^n$ is defined by all the points $x \in S^n$ with distance metric $\mathrm{d}(y,x) < \varepsilon$.

Optimisation and Black box

Local and Global Optimum

Definition 1 (Local and Global Optima)

Given a function $f : S^n \to \mathbb{R}$, the set $\mathbb{Y}$ of local optima is given by $\mathbb{Y} := \{y \mid y \in S^n, \forall x \in S^n : \mathrm{d}(y, x) < \varepsilon \Rightarrow f(y) \geq f(x)\}$. And the global optimum x_{opt} is defined as $x_{\text{opt}} := \arg\max\{f(y) \mid y \in S^n\}$.

- A global optimum is any search point contained in the search space with the unique highest objective function value.
  - There is only one highest objective function value, but there may exist multiple or even infinitely many search points corresponding to it.

Contents

Unimodal vs Multimodal Optimisation

Unimodal and Multimodal Functions

In cases where a function f only contains one global optimum, f it is called unimodal function.

Definition 2 (Unimodal Function)

A function f is called unimodal if and only if for every search point x that is not a global optimum there is a neighbour y of x with $f(y) > f(x)$.

Functions with several local and global optima of equal or different objective function values are commonly called multimodal functions.

Unimodal vs Multimodal Optimisation

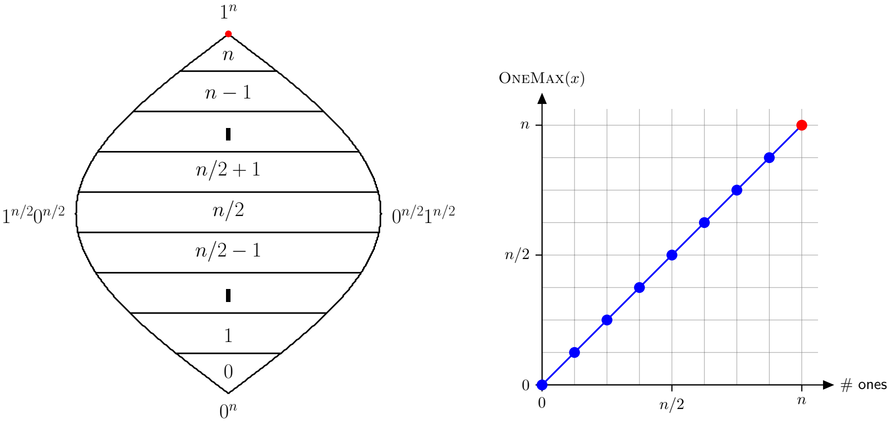
Simple Unimodal Function

Definition 3 ($\textsc{OneMax}$)

The function counts the number of ones in the bitstring $x \in \{0, 1\}^n$, then

$$\textsc{OneMax}(x) := \sum_{i=1}^{n} x_i.$$

- The goal of the optimisation process is to find the maximum number of ones in a bitstring.
- The global optimum is the 1^n bitstring.
- By symmetry, $\textsc{ZeroMax}(x) := n - \textsc{OneMax}(x)$ holds for the 0^n bitstring.



(a) Genotype (b) Phenotype

Figure 1: Sketches of the function OneMax with $n = 8$.

Unimodal vs Multimodal Optimisation

Simple Multimodal Function

Definition 4 (TwoMax, Pelikan and Goldberg, 2000)

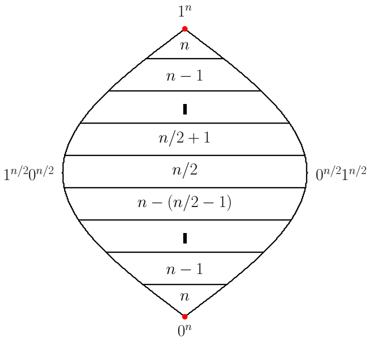
A bimodal function which consists of two different symmetric slopes ZeroMax and OneMax with 0^n and 1^n as global optima, respectively.

$$\text{TwoMax}(x) := \max\left\{\sum_{i=1}^{n} x_i, n - \sum_{i=1}^{n} x_i\right\}.$$

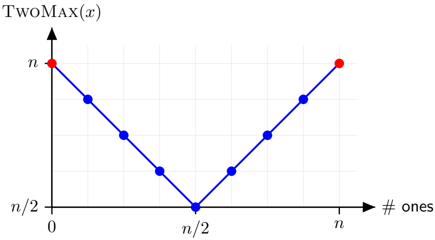
- Two symmetric branches.
- The goal of the optimisation process is to find the maximum number of ones and zeroes.
- The global optimum is the 0^n and 1^n bitstring.

Unimodal vs Multimodal Optimisation

Simple Multimodal Function



(a) Genotype



(b) Phenotype

Figure 2: Sketch of the function TwoMax with $n = 8$.

Multimodal Optimisation

What exactly is the task when we speak of multimodal optimisation?

When tackling a multimodal problem, we may address three related but different issues. These are (Preuss, 2015):

- Locate all search points corresponding to the global optimum.
- Extract the full set of optima and search points the problem possesses.
- Find the global optimum, together with at least one of its search points.

Multimodal Optimisation

In the context of Evolutionary Computation

An objective function is called fitness function.

An objective function value is called fitness function value.

A search point x is now called individual or solution.

A multi-set of search points is called population.

Every time the fitness function is used to measure the quality of an individual we will refer to it as a fitness evaluation.

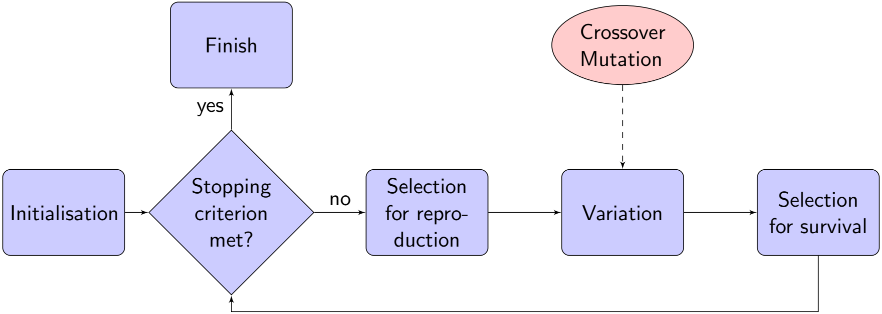
We will use the term fitness landscape to refer to the "shape" of the search space based on natural landscapes. For example, for the case of $\textsc{OneMax}$, the optimisation process consists of a "hill-climbing" task where the goal is to reach the top of a "hill".

Multimodal Optimisation

Evolutionary Computation

Evolutionary Strategy (ES)

- Is a randomised search heuristic inspired by the evolution process in nature.



Multimodal Optimisation

Evolutionary Computation

Evolutionary Strategy (ES)

- Is a randomised search heuristic inspired by the evolution process in nature.

Advantages

- Population of solutions.
  - Present several solutions.
  - Natural way to explore the fitness landscape.
  - Track, move without getting stuck in local optima.

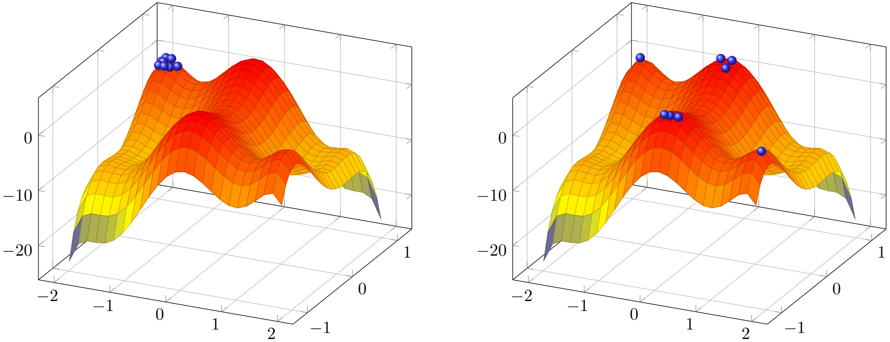
Disadvantages

- Premature convergence.
  - More difficult to create new individuals.
  - No reason for a population.

Multimodal Optimisation

Premature Convergence (the main problem)

- The population converging to a sub-optimal individual before the fitness landscape is explored properly[1].



[1] Six-Hump Camel Back function: 2 global optima, 2 local optima.

Multimodal Optimisation

Premature Convergence (the main problem)

How can we avoid premature convergence?

Multimodal Optimisation

Premature Convergence (the main problem)

How can we avoid premature convergence?

Equip the ES with some diversity mechanism to reduce the risk of premature convergence.

Multimodal Optimisation

Premature Convergence (the main problem)

How can we avoid premature convergence?

Equip the ES with some diversity mechanism to reduce the risk of premature convergence.

- Force the population to explore the fitness landscape simultaneously.

Multimodal Optimisation

Premature Convergence (the main problem)

How can we avoid premature convergence?

Equip the ES with some diversity mechanism to reduce the risk of premature convergence.

- Force the population to explore the fitness landscape simultaneously.
- Break clusters of similar individuals and promote the search to unexplored regions of the search space.

Multimodal Optimisation

Premature Convergence (the main problem)

How can we avoid premature convergence?

Equip the ES with some diversity mechanism to reduce the risk of premature convergence.

- Force the population to explore the fitness landscape simultaneously.
- Break clusters of similar individuals and promote the search to unexplored regions of the search space.
- Improve the performance of the ES by improving other operators like mutation and/or crossover.

Multimodal Optimisation

Premature Convergence (the main problem)

If you promote diversity in your population, then:

- Can be presented to a decision maker in multi-objective optimisation (covered in this course).
- Keep track of changing optima in dynamic optimisation (not covered in this course).
- Maintain/preserve “good” solutions for an exponential to infinite time period with respect to population size.

Intensification vs Diversification

Exploitation vs Exploration

There are two forces that largely determine the behaviour of an ES, intensification and diversification[2].

[2] To the interested reader, here are some literature reviews of several high level descriptions of intensification and diversification approaches: Battiti (1996); Glover and Laguna (1997); Mitchell (1998); Stützle (1999); Yagiura and Ibaraki (2001).

Intensification vs Diversification

Exploitation vs Exploration

There are two forces that largely determine the behaviour of an ES, intensification and diversification[2].

The ES should be designed with the aim of effectively exploring the search space (diversification or also commonly known as exploration).

[2] To the interested reader, here are some literature reviews of several high level descriptions of intensification and diversification approaches: Battiti (1996); Glover and Laguna (1997); Mitchell (1998); Stützle (1999); Yagiura and Ibaraki (2001).

Intensification vs Diversification

Exploitation vs Exploration

There are two forces that largely determine the behaviour of an ES, intensification and diversification[2].

The ES should be designed with the aim of effectively exploring the search space (diversification or also commonly known as exploration).

The search performed by an ES should be "clever" enough to intensively explore areas in the search space with high quality solutions (intensification or also commonly known as exploitation).

[2] To the interested reader, here are some literature reviews of several high level descriptions of intensification and diversification approaches: Battiti (1996); Glover and Laguna (1997); Mitchell (1998); Stützle (1999); Yagiura and Ibaraki (2001).

Intensification vs Diversification

Exploitation vs Exploration

There are two forces that largely determine the behaviour of an ES, intensification and diversification[2].

The ES should be designed with the aim of effectively exploring the search space (diversification or also commonly known as exploration).

The search performed by an ES should be "clever" enough to intensively explore areas in the search space with high quality solutions (intensification or also commonly known as exploitation).

In general, the goal is to design ES with an adequate balance between intensification and diversification. And that is a very difficult task!

[2] To the interested reader, here are some literature reviews of several high level descriptions of intensification and diversification approaches: Battiti (1996); Glover and Laguna (1997); Mitchell (1998); Stützle (1999); Yagiura and Ibaraki (2001).

Intensification vs Diversification

A small literature review

Classification of intensification and diversification components:

- Basic (or intrinsic) and strategic by Blum and Roli (2003).
- Uniprocess- and multiprocess-driven approaches by Liu et al. (2009).
- Hybrid methaheuristics by Lozano and García-Martínez (2010).
- Favouring diversification over intensification in population-based ES by Alba and Dorronsoro (2005); Chaiyaratana et al. (2007); Goldberg (1989); Koumousis and Katsaras (2006); Lozano et al. (2005).
- Favouring intensification over diversification in population-based ES by Kazarlis et al. (2001); Lozano et al. (2004); Noman and Iba (2008).

Happy reading!

Niching techniques: sharing and crowding

One particular way of diversity maintenance are the niching methods, based on the mechanics of natural ecosystems (Shir, 2012).

- A niche can be viewed as a subspace in the environment that can support different types of life.
- A specie is defined as a group of individuals with similar features capable of interbreeding among themselves but that are unable to breed with individuals outside their group.

In the context of Evolutionary Computation:

- A niche is used for the search space domain.
- A specie is the multi-set of individuals with similar characteristics in terms of distance metrics.

Niching techniques: sharing and crowding

Goals and Advantages

Niching methods have been developed to (Sareni and Krahenbuhl, 1998):

- reduce the effect of genetic drift resulting from the selection operator in standard ES,
- maintain the population diversity,
- allow the ES to investigate many peaks in parallel and,
- avoid getting trapped in local optima.

Niching techniques: sharing and crowding

Goals and Advantages

Niching methods have been developed to (Sareni and Krahenbuhl, 1998):

- reduce the effect of genetic drift resulting from the selection operator in standard ES,
- maintain the population diversity,
- allow the ES to investigate many peaks in parallel and,
- avoid getting trapped in local optima.

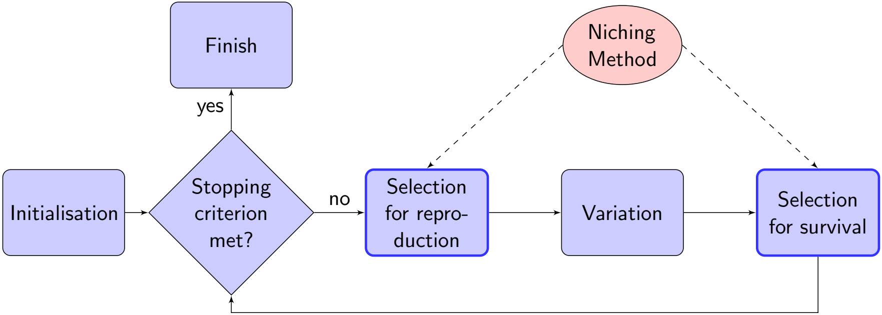
A diverse population can deal with multimodal problems as it can explore several hills in the fitness landscape simultaneously.

Niching techniques: sharing and crowding

How do they work?

Modify the selection process of individuals (Glibovets and Gulayeva, 2013):

- Select individuals taking into consideration not only the value of the fitness function but also the distribution of individuals in the space of genotypes or phenotypes.



Niching techniques: sharing and crowding

Most popular niching techniques

- Fitness sharing (Goldberg and Richardson, 1987).
- Clustering (Yin and Germay, 1993).
- Deterministic crowding (Mahfoud, 1995).
- Restricted tournament selection (Harik, 1995).
- Clearing (Pétrowski, 1996).
- Probabilistic crowding (Mengsheol and Goldberg, 1999).
- Generalised crowding (Galán and Mengshoel, 2010).
- ...

And many more methods covered in tutorials and surveys for diversity-preserving mechanisms (see Črepinšek et al., 2013; Squillero and Tonda, 2016).

Niching techniques: sharing and crowding

Crowding Techniques

In general, parents and offspring compete in a replacement-oriented survival process. The most well-known (or popular) crowding methods are:

- Standard Crowding (De Jong, 1975).
- Deterministic crowding (Mahfoud, 1995).
- Restricted tournament selection (Harik, 1995).
- Probabilistic crowding (Mengsheol and Goldberg, 1999).
- Generalised crowding (Galán and Mengshoel, 2010).

Niching techniques: sharing and crowding

Crowding Techniques

In general, parents and offspring compete in a replacement-oriented survival process. The most well-known (or popular) crowding methods are:

- Standard Crowding (De Jong, 1975).
- Deterministic crowding (Mahfoud, 1995).
- Restricted tournament selection (Harik, 1995).
- Probabilistic crowding (Mengsheol and Goldberg, 1999).
- Generalised crowding (Galán and Mengshoel, 2010).

Niching techniques: sharing and crowding

Deterministic Crowding

All elements are grouped into $\mu/2$ pairs (where μ is the population size and assuming μ to be even).

After the variation step, for each pair of offspring, two sets of parent-child tournaments are possible.

Each offspring competes against the most similar parent according to a distance metric and the offspring replace their closest parent if it is at least as good (Mahfoud, 1995).

Niching techniques: sharing and crowding

Deterministic Crowding

Just taking into consideration two parents and two offspring. The procedure is the same for the remaining parents and offspring.

Selection for replacement

x_1 and x_2

Variation

$V(x_1, x_2) = \{y_1, y_2\}$

Selection for survival

if $(\mathrm{d}(x_1, y_1) + \mathrm{d}(x_2, y_2) \leq \mathrm{d}(x_1, y_2) + \mathrm{d}(x_2, y_1))$ **then**
  if $f(y_1) \geq f(x_1)$ **then** replace x_1 with y_1.
  if $f(y_2) \geq f(x_2)$ **then** replace x_2 with y_2.
else
  if $f(y_2) \geq f(x_1)$ **then** replace x_1 with y_2.
  if $f(y_1) \geq f(x_2)$ **then** replace x_2 with y_1.

Niching techniques: sharing and crowding

Probabilistic Crowding

All elements are grouped into $\mu/2$ pairs (where μ is the population size and assuming μ to be even).

After the variation step, for each pair of offspring, two sets of parent-child tournaments are possible.

Each offspring competes against the most similar parent according to a distance metric and the offspring replace their closest parent proportionally according to their fitness (Mengsheol and Goldberg, 1999).

Niching techniques: sharing and crowding

Probabilistic Crowding

Just taking into consideration two parents and two offspring. The procedure is the same for the remaining parents and offspring.

Selection for replacement

x_1 and x_2

Variation

$V(x_1, x_2) = \{y_1, y_2\}$

Selection for survival

if $(\mathrm{d}(x_1, y_1) + \mathrm{d}(x_2, y_2) \leq \mathrm{d}(x_1, y_2) + \mathrm{d}(x_2, y_1))$ **then**

With prob. $\frac{f(y_1)}{f(y_1)+f(x_1)}$ replace x_1 with y_1.

With prob. $\frac{f(y_2)}{f(y_2)+f(x_2)}$ replace x_2 with y_2.

else

With prob. $\frac{f(y_2)}{f(y_2)+f(x_1)}$ replace x_1 with y_2.

With prob. $\frac{f(y_1)}{f(y_1)+f(x_2)}$ replace x_2 with y_1.

Niching techniques: sharing and crowding

Generalised Crowding

All elements are grouped into $\mu/2$ pairs (where μ is the population size and assuming μ to be even).

After the variation step, for each pair of offspring, two sets of parent-child tournaments are possible.

Each offspring competes against the most similar parent according to a distance metric and the offspring replace their closest parent proportionally according to their fitness by introducing a scaling factor $\phi \in [0, 1]$ that diminishes the fitness of the inferior search point (Galán and Mengshoel, 2010).

Niching techniques: sharing and crowding

Generalised Crowding

Just taking into consideration two parents and two offspring. The procedure is the same for the remaining parents and offspring.

Selection for replacement

x_1 and x_2

Variation

$V(x_1, x_2) = \{y_1, y_2\}$

Selection for survival

if $(\mathrm{d}(x_1, y_1) + \mathrm{d}(x_2, y_2) \leq \mathrm{d}(x_1, y_2) + \mathrm{d}(x_2, y_1))$ **then**

With prob. $\begin{cases} \frac{f(y_1)}{f(y_1)+\phi \cdot f(x_1)} & \text{if } f(y_1) > f(x_1), \\ 0.5 & \text{if } f(y_1) = f(x_1), \\ \frac{\phi \cdot f(y_1)}{\phi \cdot f(y_1)+f(x_1)} & \text{if } f(y_1) < f(x_1), \end{cases}$

replace x_1 with y_1.

The same for the case of x_2 and y_2.

else

The same for the case of x_1 and y_2.

The same for the case of x_2 and y_1.

Niching techniques: sharing and crowding

Generalised Crowding

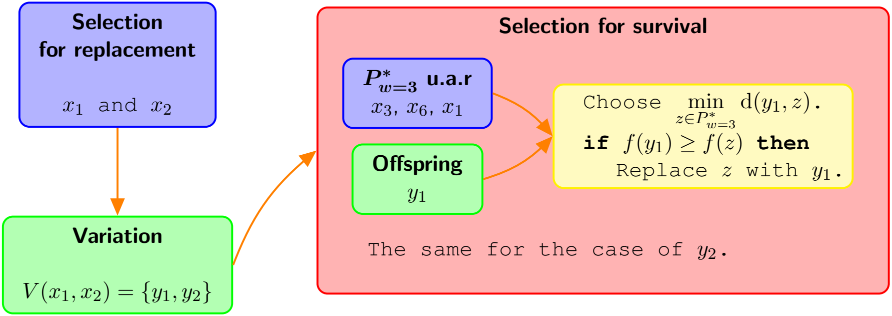
Main observations:

- If $\phi = 1$ we obtain probabilistic crowding.
- If $\phi = 0$ we obtain deterministic crowding as then the better offspring is selected with probability 1.
- In case of ties, the offspring is kept with probability $1/2$ in generalised crowding whereas in deterministic crowding the offspring is always preferred in this case.

Niching techniques: sharing and crowding

Restricted Tournament Selection

The offspring replace their most similar individual from a random subpopulation of size w (window size) if it is at least as good (Harik, 1995).



Standard Crowding (De Jong, 1975) works similar to RTS but instead of w (window size) is called CF (crowding factor), and the offspring always replace the closest individual from the subpopulation with size CF.

Niching techniques: sharing and crowding

Sharing Techniques

Like the name suggest these methods are inspired by the principle of "sharing" limited resources within a niche (or subpopulation) of individuals characterised by some similarities. The most well-known (or popular) sharing methods are:

- Fitness sharing (Goldberg and Richardson, 1987).
  - Population-based Fitness sharing (Friedrich et al., 2009).
- Clustering (Yin and Germay, 1993).
- Clearing (Pétrowski, 1996).
  - Modified Clearing (Singh and Deb, 2006).

Niching techniques: sharing and crowding

Sharing Techniques

Like the name suggest these methods are inspired by the principle of "sharing" limited resources within a niche (or subpopulation) of individuals characterised by some similarities. The most well-known (or popular) sharing methods are:

- Fitness sharing (Goldberg and Richardson, 1987).
  - Population-based Fitness sharing (Friedrich et al., 2009).
- Clustering (Yin and Germay, 1993).
- Clearing (Pétrowski, 1996).
  - Modified Clearing (Singh and Deb, 2006).

Niching techniques: sharing and crowding

Fitness Sharing

Restricts the growth of one type of individuals by sharing its real fitness assignment with nearby elements in the population (Goldberg and Richardson, 1987).

The amount of sharing contributed by each individual into its neighbour depends on the proximity between the individuals.

Niching techniques: sharing and crowding

Fitness Sharing

Functions and parameters to consider when using Fitness Sharing:

- The shared fitness $f_{\mathrm{sh}}(x, P)$ of individual x in the population P with fitness $f(x)$.
- A sharing function $\mathrm{sh}(x, y) \in [0, 1]$ that determines the similarity between individuals x and y (a large value corresponds to a great similarity and a 0 value implies no similarity).
  - A distance function $\mathrm{d}(x, y)$ is needed in $\mathrm{sh}(x, y)$.
  - α is a positive constant called scaling factor.
  - σ is called sharing radius, a threshold that determines if they belong to the same niche or not.

Niching techniques: sharing and crowding

Fitness Sharing

The shared fitness $f_{\mathrm{sh}}(x, P)$ of individual x in the population P with fitness $f(x)$ is calculated in the following way

Selection for replacement

P_t

Variation

$V(P_t) = Q_t$

Selection for survival

Let $P = P_t \cup Q_t$.

for all $x \in P$ **do**

$$f_{\mathrm{sh}}(x, P) = \frac{f(x)}{\sum_{y \in P}^{\mu} \mathrm{sh}(x, y)},$$

where $$\mathrm{sh}(x, y) = \begin{cases} 1 - \left(\frac{\mathrm{d}(x,y)}{\sigma}\right)^{\alpha}, & \text{if } \mathrm{d}(x, y) < \sigma; \\ 0, & \text{otherwise.} \end{cases}$$

Select μ individuals from P according to the selection mechanism.

Niching techniques: sharing and crowding

Clustering

A clustering algorithm (e. g., Macqueen's K-mean algorithm) is used to divide the population into niches (Yin and Germay, 1993).

In a similar way in which individuals from the same niche share their fitness in the fitness sharing method, individuals in the same cluster share their fitness.

Niching techniques: sharing and crowding

Clustering

Functions and parameters to consider when using Clustering:

- Use a clustering algorithm to divide the population into niches.
  - Yin and Germay (1993) used the Macqueen's K-mean algorithm. So it is necessary to know the k number of clusters beforehand.
- The shared fitness $f_{\mathrm{sh}}(x)$ of an individual x in the niche or cluster k with fitness $f(x)$.
- $\sigma_{\min}$ is the minimum threshold that represents the minimum distance allowed between each niche centroid.
- $\sigma_{\max}$ is the maximum threshold allowed between an individual and its niche centroid.
- α is a positive constant called scaling factor.
- A distance function $\mathrm{d}(x, c)$, any dissimilarity measure between and individual x and its corresponding c centroid.

Niching techniques: sharing and crowding

Clustering

Selection for replacement

P_t

Variation

$V(P_t) = Q_t$

Selection for survival

Let $P = P_t \cup Q_t$.
The best k individuals with $\sigma_{\min}$ are used as seeds (centroids) for the Macqueen's K-mean algorithm.
for all $k \in K$ **do**
  Let c be the centroid of cluster k.
  Let κ be the number of individuals in the k cluster.
  for all $x \in k$ **do**
    $f_{\text{sh}}(x) = \frac{f(x)}{\left(\kappa\left(1-\frac{\mathrm{d}(x,c)}{2\cdot\sigma_{\max}}\right)^{\alpha}\right)}$.
Select μ individuals from P according to the selection mechanism.

Niching techniques: sharing and crowding

Clearing

Supplies these resources only to the best individual of each niche: the winner. The winner takes all rather than sharing resources with the other individuals of the same niche as it is done with fitness sharing (Pétrowski, 1996).

The basic idea is to preserve the fitness of the individual that has the best fitness (also called dominant individual), while it resets the fitness of all the other individuals of the same niche to the zero[3].

The number of winners in each niche can be unique or can be dominated by several winners.

[3] Assuming that all fitness values are larger than 0 for simplicity. In case of a fitness function f with negative fitness values you can change clearing to reset fitness to $f_{\min} - 1$, where $f_{\min}$ is the minimum fitness value of f, such that all reset individuals are worse than any other individuals.

Niching techniques: sharing and crowding

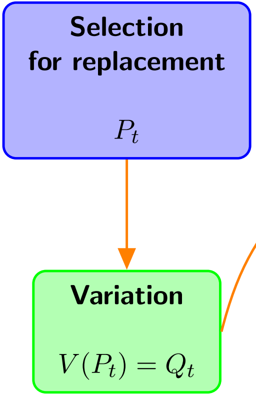
Clearing

Functions and parameters to consider when using Clearing:

- σ is called clearing radius, a threshold that determines if they belong to the same niche or not.
- κ is called niche capacity, best individuals of each niche defined as the maximum number of winners that a niche can accept.
- A distance function $\mathrm{d}(x, y)$, any dissimilarity measure between two individuals x and y in population P.

Niching techniques: sharing and crowding

Clearing



Selection for survival

Let $P = P_t \cup Q_t$.
Sort P according to fitness of individuals by decreasing values.

for $i := 1$ **to** $|P|$ **do**
   if $f(P[i]) > 0$ **then**
      $winners := 1$.
      for $j := i+1$ **to** $|P|$ **do**
         if $f(P[j]) > 0$ **and** $\mathrm{d}(P[i], P[j]) < \sigma$ **then**
            if $winners < \kappa$ **then**
               $winners := winners + 1$.
            else
               $f(P[j]) := 0$.

Select μ individuals from P according to the selection mechanism.

Contents

Multi-Objective Optimisation

A multimodal optimisation problem have only one objective function, then these problems form part of the family of single-objective optimisation (SO) problems.

When an optimisation problem involves more than one objective function, the task of finding one or more trade-offs between solutions is known as multi-objective optimisation (MOO).

Compared with SO problems, which have a unique solution, the solution to MOO problems consists of sets of trade-offs between objectives.

Multi-Objective Optimisation

What exactly is the task when we speak of multi-objective optimisation?

Explore all possible trade-offs between objectives. This is accomplished by performing the following tasks:

- Push the population close to the Pareto front.
- Spread the population along the front such that is well covered.

Pareto Optimisation

Pareto Front

Let us consider problems $f = (f_1, \ldots, f_m)\colon \mathbb{R}^n \to \mathbb{R}^m$. we will without loss of generality that each function f_i, $1 \leq i \leq m$, should be maximised. As there is no single point that maximise all functions simultaneously, the goal is to find a set of so-called Pareto-optimal solutions.

Definition 5 (Pareto Optimality, Maximisation)

Let $f : X \to F$, where $X \subseteq S^n$ is called decision space and $F \subseteq \mathbb{R}^m$ objective space. The elements of X are called decision vectors and the elements of F objective vectors. A decision vector $x \in X$ is Pareto optimal if there is no other $y \in X$ that dominates x. The set of all Pareto-optimal decision vectors X^* is called Pareto set. $F^* = f(X^*)$ is the set of all Pareto-optimal objective vectors and denoted as Pareto front.

Pareto Optimisation

Dominance Relations

- y dominates x, denoted as $y \succ x$, if $f_i(y) \geq f_i(x)$ for all $i = 1, \ldots, m$ and $f_i(y) > f_i(x)$ for at least one index i.
- y weakly dominates x, denoted by $y \succeq x$, if $f_i(y) \geq f_i(x)$, for all i.
- y is incomparable to x, denoted by $f_i(y) \parallel f_i(x)$ if neither $f_i(y)$ weakly dominates $f_i(x)$ nor $f_i(x)$ weakly dominates $f_i(y)$.
- y is indifferent to x, denoted by $f_i(y) \sim f_i(x)$ if $f_i(y)$ has the same value $f_i(x)$ in each objective.

Pareto Optimisation

Example of Pareto front

Definition 6 (LOTZ)

A pseudo-Boolean function $\{0,1\}^n \to \mathbb{N}^2$ defined as

$$\mathrm{LOTZ}(x_1,\ldots,x_n) := \left(\sum_{i=1}^{n}\prod_{j=1}^{i} x_j, \sum_{i=1}^{n}\prod_{j=i}^{n}(1-x_j)\right),$$

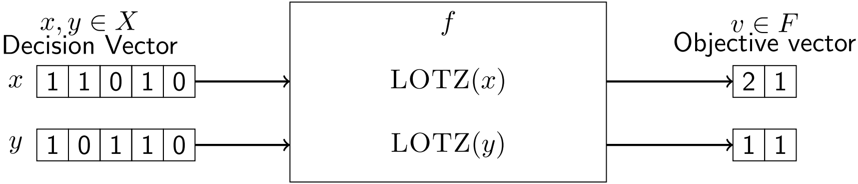
where the goal is to simultaneously maximise the number of leading ones and trailing zeroes. In the case of LOTZ, all non-Pareto optimal decision vectors only have Hamming neighbours that are better or worse.

Pareto Optimisation

Pareto Front

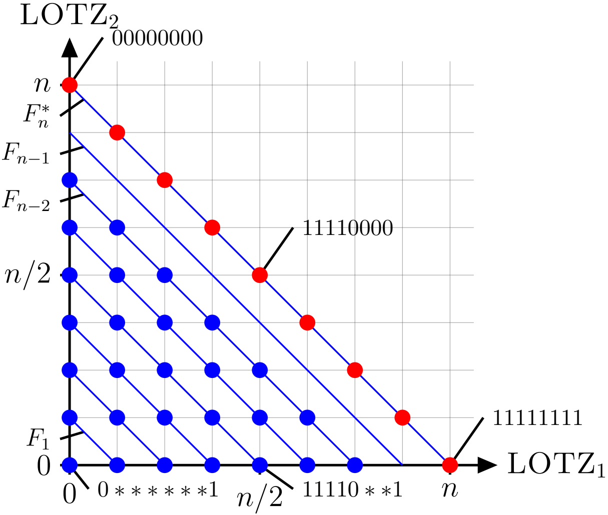
Given the function $f : X \to F$ (using LOTZ as example).

- $X \subseteq \{0,1\}^n$ is our decision space.
- $F \subseteq \mathbb{R}^m$ is our objective space.



Pareto Front

Sketch of LOTZ with $n = 8$



Evolutionary Algorithms for Multi-Objective Optimisation

Multi-Objective Evolutionary Algorithms (MOEAs)

The most well-known (popular or well-established) MOEAs are:

- PAES (Knowles and Corne, 1999),
- SPEA2 (Bleuler et al., 2001),
- NSGA-II (Deb et al., 2002),
- IBEA (Zitzler and Künzli, 2004),
- SMS-EMOA (Beume et al., 2007),
- and many more.

Evolutionary Algorithms for Multi-Objective Optimisation

Multi-Objective Evolutionary Algorithms (MOEAs)

The most well-known (popular or well-established) MOEAs are:

- PAES (Knowles and Corne, 1999),
- SPEA2 (Bleuler et al., 2001),
- NSGA-II (Deb et al., 2002),
- IBEA (Zitzler and Künzli, 2004),
- SMS-EMOA (Beume et al., 2007).
- and many more.

Evolutionary Algorithms for Multi-Objective Optimisation

NSGA-II Operators (Deb et al., 2002)

- Initialisation:
  - Uniformly at random initialisation.
- Termination Criterion:
  - Maximum number of fitness evaluations reached.
- Selection for reproduction:
  - Binary Tournament Selection.
- Variation for Binary-coded GAs:
  - Single-point crossover.
  - Bit-wise mutation.
- Variation for Real-coded GAs (Deb and Agrawal, 1995):
  - Simulated Binary Crossover (SBX).
  - Polynomial Mutation.
- Selection for survival:
  - Truncation Selection $(\mu + \lambda)$ that uses:
    - Fast Non-Dominated Sort procedure and the
    - Crowding Distance Assignment $(\succ_m)$ procedure .

Evolutionary Algorithms for Multi-Objective Optimisation

NSGA-II Operators (Deb et al., 2002)

- Initialisation:
  - Uniformly at random initialisation.
- Termination Criterion:
  - Maximum number of fitness evaluations reached.
- Selection for reproduction:
  - Binary Tournament Selection.
- Variation for Binary-coded GAs:
  - Single-point crossover.
  - Bit-wise mutation.
- Variation for Real-coded GAs (Deb and Agrawal, 1995):
  - Simulated Binary Crossover (SBX).
  - Polynomial Mutation.
- Selection for survival:
  - Truncation Selection $(\mu + \lambda)$ that uses:
    - Fast Non-Dominated Sort procedure and the
    - Crowding Distance Assignment $(\succ_m)$ procedure.

Evolutionary Algorithms for Multi-Objective Optimisation

The Fast Non-dominated Sorting procedure

Divides the population into sub-populations using its non-domination rank, i. e., each solution is assigned a fitness (or rank) equal to its non-domination level (1 is the best level, 2 is the next-best level, and so on).

This rank basically consists in the number of individuals that a certain individual dominates. If one individual is not dominated by any individual, then its rank is defined as 1.

Evolutionary Algorithms for Multi-Objective Optimisation

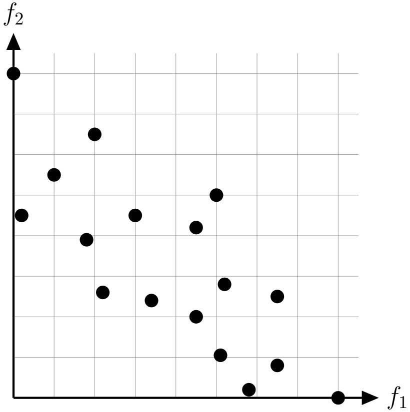
The Fast Non-dominated Sorting procedure

Algorithm 1 Fast Non-dominated Sorting

for all p individuals $\in P$ **do**
    Let $S_p = \emptyset$ and $n_p := 0$.
    for all q individuals $\in P$ **do**
        if $p \succ q$ **then** ▷ If p dominates q.
            Let $S_p = S_p \cup q$. ▷ Add q to the set of solutions dominated by p.
        else if $q \succ p$ **then**
            Let $n_p = n_p + 1$. ▷ Increment the domination counter of p.
    if n_p **equal to** 0 **then** ▷ p belongs to the first front.
        Let $p_{.\mathrm{rank}} := 1$ and $F_1 = F_1 \cup p$.
Let $i := 1$ ▷ Initialise the front counter.
while $F_i \neq \emptyset$ **do**
    Let $Q = \emptyset$. ▷ Used to store the members of the next front.
    for all p individuals $\in F_i$ **do**
        for all q individuals $\in S_p$ **do**
            Let $n_q := n_p - 1$.
            if n_q **equal to** 0 **then** ▷ q belongs to the next front.
                Let $q_{.\mathrm{rank}} := i + 1$ and $Q = Q \cup q$.
    Let $i := i + 1$ and $F_i = Q$.
return F

Evolutionary Algorithms for Multi-Objective Optimisation

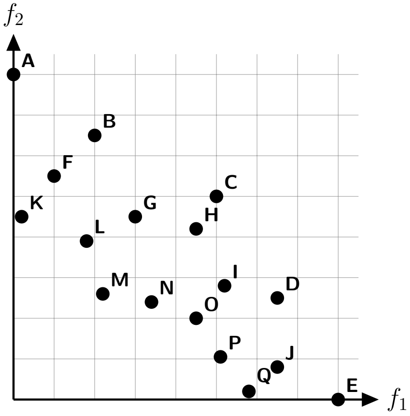
The Fast Non-dominated Sorting procedure



- From Line 1 to 10 of the Fast Non-dominated Sorting algorithm:
  - Find the set of individuals with n_p equal to 0 and
  - the set of individuals their dominate S_p.

Evolutionary Algorithms for Multi-Objective Optimisation

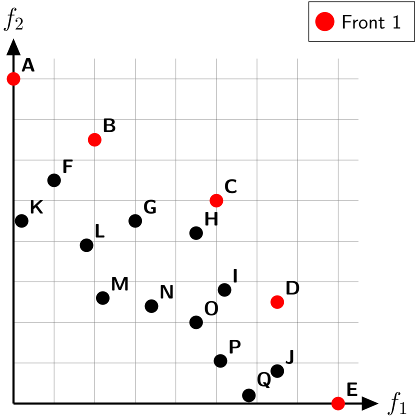
The Fast Non-dominated Sorting procedure



- Individual **A**:
  - $n_p := 0$.
  - $S_p = \{\mathbf{F}, \mathbf{K}, \ldots\}$.
- Individual **B**:
  - $n_p := 0$.
  - $S_p = \{\mathbf{F}, \mathbf{K}, \mathbf{L}, \ldots\}$.
- The same with the rest of the individuals.
- The individuals with a n_p equal to 0 belong to the Front 1.

Evolutionary Algorithms for Multi-Objective Optimisation

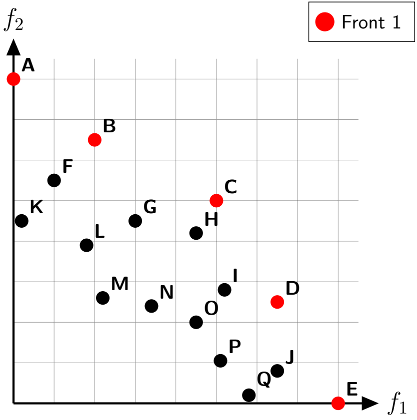
The Fast Non-dominated Sorting procedure



- Individual **A**:
  - $n_p := 0$.
  - $S_p = \{\mathbf{F}, \mathbf{K}, \ldots\}$.
- Individual **B**:
  - $n_p := 0$.
  - $S_p = \{\mathbf{F}, \mathbf{K}, \mathbf{L}, \ldots\}$.
- The same with the rest of the individuals.
- The individuals with a n_p equal to 0 belong to the Front 1.

Evolutionary Algorithms for Multi-Objective Optimisation

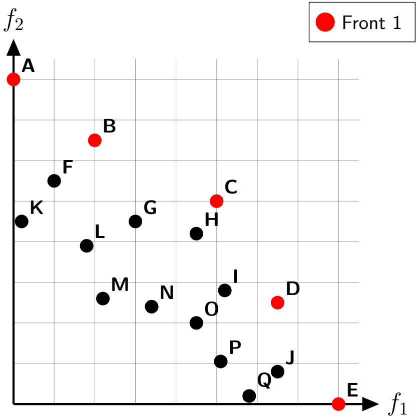
The Fast Non-dominated Sorting procedure



- From Line 11 to 18 of the algorithm we look for the individuals belonging to the Front 2, 3, and so on.

Evolutionary Algorithms for Multi-Objective Optimisation

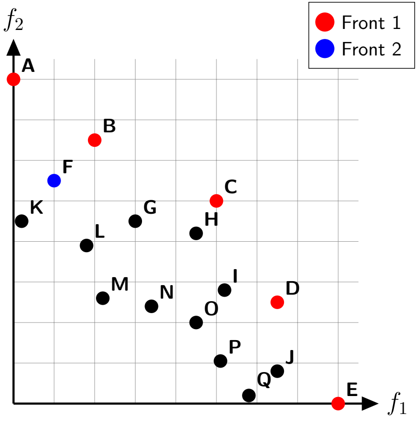
The Fast Non-dominated Sorting procedure



- Individual **A**:
  - $n_p := 0.$
  - $S_p = \{$**F**, **K**$, \ldots\}.$
- Individual **B**:
  - $n_p := 0.$
  - $S_p = \{$**F**, **K**, **L**$, \ldots\}.$
- Individual **F**:
  - $n_p := 2.$
  - $S_p = \{$**L**$\}.$
- From Front 1, **F** is dominated by individuals **A** and **B**.
- Subtract from n_p of **F** the number of individuals that dominate **F** until n_p is set to 0.

Evolutionary Algorithms for Multi-Objective Optimisation

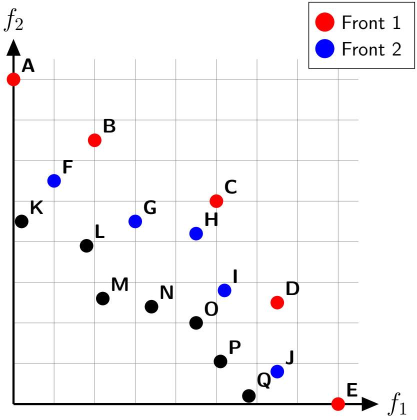
The Fast Non-dominated Sorting procedure



- Since the n_p counter of **F** is set to 0, it means that it was only nominated by elements on the Front 1.
- Thus, the corresponding front for **F** is the Front 2.
- Continue with the process for all the elements in Front 1 to find the remaining elements of Front 2.

Evolutionary Algorithms for Multi-Objective Optimisation

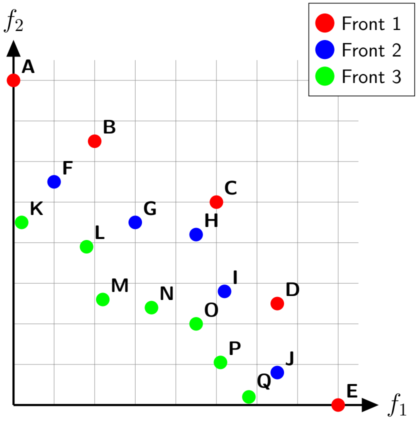
The Fast Non-dominated Sorting procedure



- Since the n_p counter of **F** is set to 0, it means that it was only nominated by elements on the Front 1.
- Thus, the corresponding front for **F** is the Front 2.
- Continue with the process for all the elements in Front 1 to find the remaining elements of Front 2.

Evolutionary Algorithms for Multi-Objective Optimisation

The Fast Non-dominated Sorting procedure



- The elements of Front 2 are now used to find the elements of Front 3.
- Continue the process until there are no more individuals to assign to other fronts.
- Finally, a population F consists of the population partitioned by the fronts found. In this example F_1 will consist of all individuals belonging to Front 1 (the red ones), and so on.

Evolutionary Algorithms for Multi-Objective Optimisation

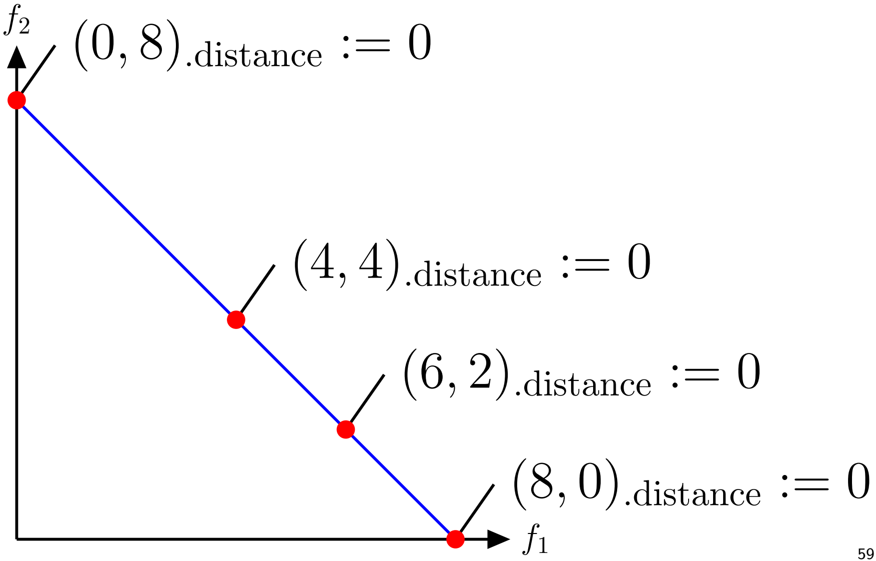
The Crowding Distance Assignment ($\succ_m$) procedure

It is a density metric used to determine the extent proximity of a solution with other surrounding solutions in the population.

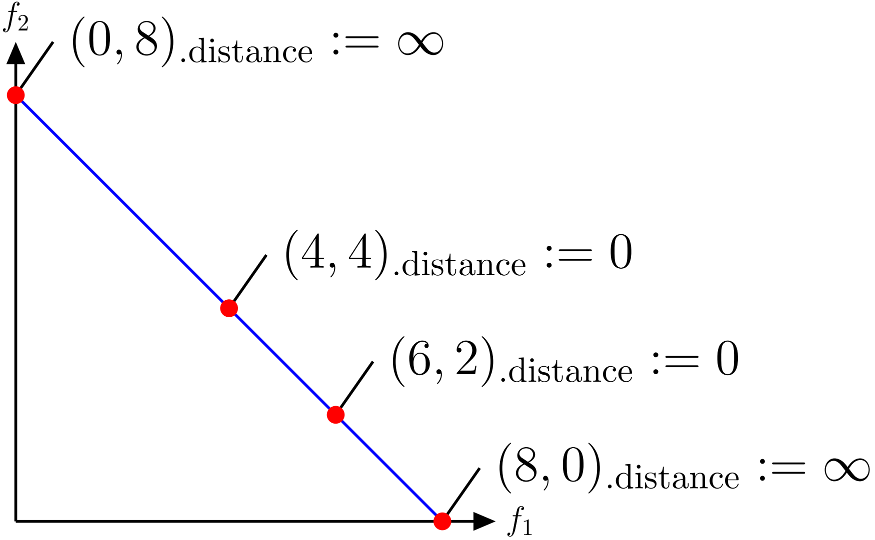
Algorithm 2 Crowding Distance Assignment ($\succ_m$)

Let $l := |P|$.
for all i individuals $\in P$ **do**
    Set $P[i]_{.\text{distance}} := 0$
for all m objectives **do**
    Sort P according to m objective function value in ascending order.
    $P[1]_{.\text{distance}} := P[l]_{.\text{distance}} := \infty$.
    for $i = 2$ **to** $l - 1$ **do**
        $P[i]_{.\text{distance}} := P[i]_{.\text{distance}} + \frac{P[i+1]_{.m} - P[i-1]_{.m}}{f_m^{\max} - f_m^{\min}}$.

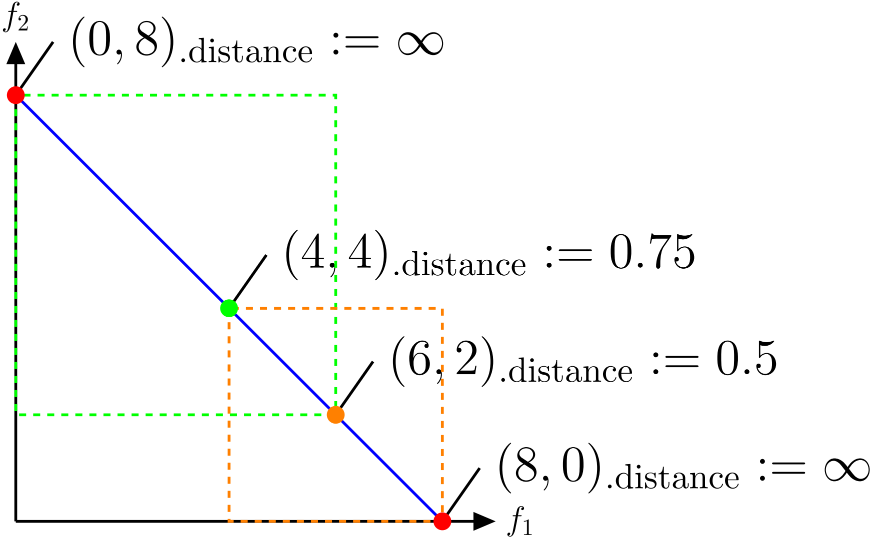
Diversity metrics - Crowding Distance Operator



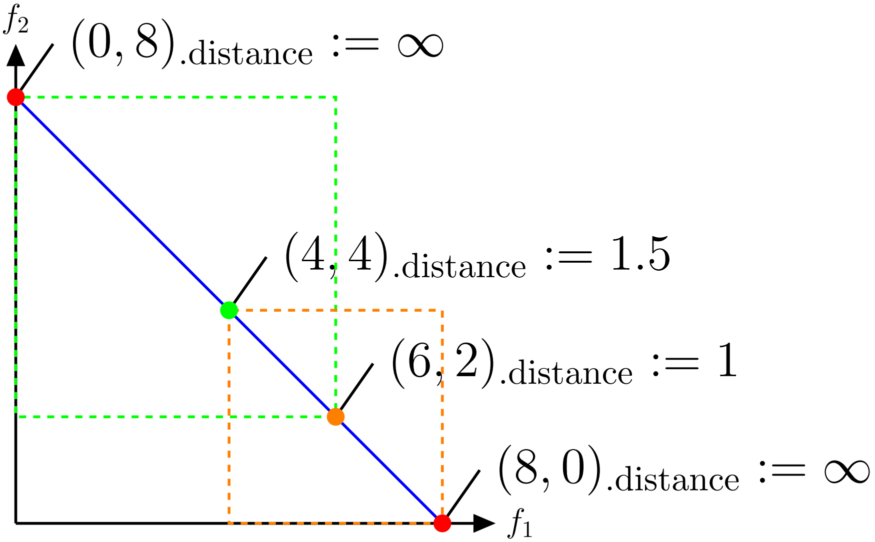
Diversity metrics - Crowding Distance Operator



Diversity metrics - Crowding Distance Operator



Diversity metrics - Crowding Distance Operator



Evolutionary Algorithms for Multi-Objective Optimisation

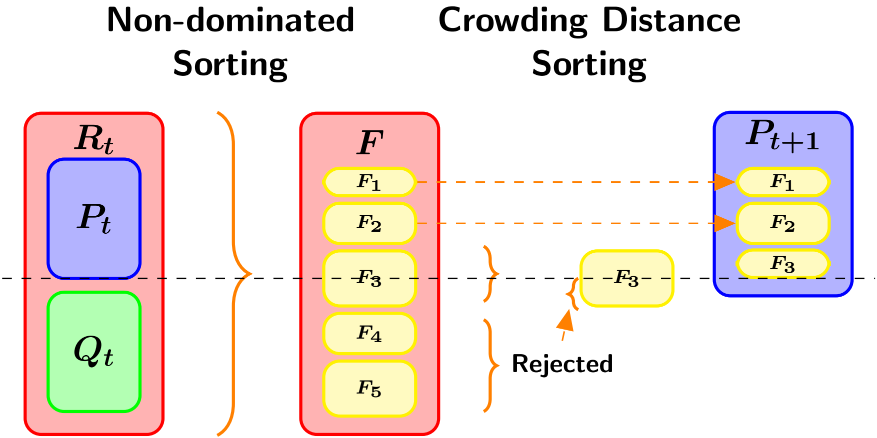
NSGA-II Algorithm

Algorithm 3 NSGA-II

Let $t := 0$ and initialise the population P_0 with μ individuals chosen uniformly at random.
while stopping criterion **not** met **do**
    From population P_t create an offspring population Q_t size λ using selection, crossover and mutation.
    Let $R_t = P_t \cup Q_t$.
    Let $F = \texttt{fast-non-dominated-sort}(R_t)$.
    Let $P_{t+1} = \emptyset$ and $i := 1$.
    while $|P_{t+1}| + |F_i| \leq \mu$ **do**
        Let $P_{t+1} = P_{t+1} \cup F_i$ and $i := i + 1$.
    $\texttt{crowding-distance-assignment}(F_i)$.
    Sort F_i in ascending order using $\succ_m$.
    Let $P_{t+1} = P_{t+1} \cup F_i[1 : \mu - |P_{t+1}|]$ and $t := t + 1$.

Evolutionary Algorithms for Multi-Objective Optimisation

NSGA-II Full Algorithm



Multimodal and Multi-opjective Optimisation

Recommendation

If you want to know more about these two topics, then I suggest you to:

- Spend some time reading the material cited here in this Lecture.
- If you want to practice, test your programming skills using ES to solve multimodal optimisation problems check the GECCO Competition on Niching Methods for Multimodal Optimization (Li et al., 2013).
- If you want to know more about MOEAs you can check the following material: Coello et al. (2007) and Falcón-Cardona and Coello (2020).
- Or you can always approach your lecturer if you want to discuss the materials you have read or to clarify doubts.

References I

Alba, E. and Dorronsoro, B. (2005). The exploration/exploitation tradeoff in dynamic cellular genetic algorithms. *IEEE Transactions on Evolutionary Computation*, 9(2):126–142.

Battiti, R. (1996). Reactive Search: Toward Self-Tuning Heuristics. In *Modern Heuristic Search Methods*, chapter 4, pages 61–83. John Wiley & Sons, Inc.

Beume, N., Naujoks, B., and Emmerich, M. (2007). SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, 181(3):1653–1669.

Bleuler, S., Brack, M., Thiele, L., and Zitzler, E. (2001). Multiobjective genetic programming: reducing bloat using SPEA2. In *IEEE Congress on Evolutionary Computation*, volume 1, pages 536–543.

References II

Blum, C. and Roli, A. (2003). Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison. *ACM Comput. Surv.*, 35(3):268–308.

Chaiyaratana, N., Piroonratana, T., and Sangkawelert, N. (2007). Effects of diversity control in single-objective and multi-objective genetic algorithms. *Journal of Heuristics*, 13(1):1–34.

Coello, C. C., Lamont, G. B., and van Veldhuizen, D. A. (2007). *Evolutionary Algorithms for Solving Multi-Objective Problems*. Genetic and Evolutionary Computation. Springer US, 2nd edition.

Črepinšek, M., Liu, S.-H., and Mernik, M. (2013). Exploration and Exploitation in Evolutionary Algorithms: A Survey. *ACM Comput. Surv.*, 45(3):35:1–35:33.

References III

De Jong, K. A. (1975). *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*. PhD thesis, Computer and Communications Science Department, University of Michigan.

Deb, K. and Agrawal, R. B. (1995). Simulated binary crossover for continuous search space. *Complex Systems*, 9:115–148.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197.

Falcón-Cardona, J. G. and Coello, C. A. C. (2020). Indicator-Based Multi-Objective Evolutionary Algorithms: A Comprehensive Survey. *ACM Comput. Surv.*, 53(2).

Friedrich, T., Oliveto, P. S., Sudholt, D., and Witt, C. (2009). Analysis of Diversity-preserving Mechanisms for Global Exploration. *Evolutionary Computation*, 17(4):455–476.

References IV

Galán, S. F. and Mengshoel, O. J. (2010). Generalized Crowding for Genetic Algorithms. In *Proc. of GECCO '10*, pages 775–782. ACM.

Glibovets, N. N. and Gulayeva, N. M. (2013). A Review of Niching Genetic Algorithms for Multimodal Function Optimization. *Cybernetics and Systems Analysis*, 49(6):815–820.

Glover, F. and Laguna, M. (1997). *Tabu Search*. Springer US, 1 edition.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., 1st edition.

Goldberg, D. E. and Richardson, J. (1987). Genetic Algorithms with Sharing for Multimodal Function Optimization. In *Proceedings of the Second International Conference on Genetic Algorithms on Genetic Algorithms and Their Application*, pages 41–49. L. Erlbaum Associates Inc.

References V

Harik, G. R. (1995). Finding Multimodal Solutions Using Restricted Tournament Selection. In *Proceedings of the 6th International Conference on Genetic Algorithms*, pages 24–31. Morgan Kaufmann Publishers Inc.

Kazarlis, S., Papadakis, S., Theocharis, J., and Petridis, V. (2001). Microgenetic algorithms as generalized hill-climbing operators for GA optimization. *IEEE Transactions on Evolutionary Computation*, 5(3):204–217.

Knowles, J. and Corne, D. (1999). The pareto archived evolution strategy: a new baseline algorithm for pareto multiobjective optimisation. In *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, volume 1, pages 98–105.

Koumousis, V. and Katsaras, C. (2006). A saw-tooth genetic algorithm combining the effects of variable population size and reinitialization to enhance performance. *IEEE Transactions on Evolutionary Computation*, 10(1):19–28.

References VI

Li, X., Engelbrecht, A., and Epitropakis, M. G. (2013). Benchmark Functions for CEC'2013 Special Session and Competition on Niching Methods for Multimodal Function Optimization. Technical report, Evolutionary Computation and Machine Learning Group, RMIT University, Australia.

Liu, S.-H., Mernik, M., and Bryant, B. R. (2009). To Explore or to Exploit: An Entropy-driven Approach for Evolutionary Algorithms. *Int. J. Know.-Based Intell. Eng. Syst.*, 13(3,4):185–206.

Lozano, M. and García-Martínez, C. (2010). Hybrid metaheuristics with evolutionary algorithms specializing in intensification and diversification: Overview and progress report. *Computers & Operations Research*, 37(3):481–497.

Lozano, M., Herrera, F., and Cano, J. R. (2005). Replacement Strategies to Maintain Useful Diversity in Steady-State Genetic Algorithms. In *Soft Computing: Methodologies and Applications*, volume 32 of *Advances in Soft Computing*, pages 85–96. Springer Berlin Heidelberg.

References VII

Lozano, M., Herrera, F., Krasnogor, N., and Molina, D. (2004). Real-coded Memetic Algorithms with Crossover Hill-climbing. *Evolutionary Computation*, 12(3):273–302.

Mahfoud, S. W. (1995). *Niching methods for genetic algorithms*. PhD thesis, University of Illinois at Urbana-Champaign.

Mengsheol, O. and Goldberg, D. (1999). Probabilistic Crowding: Deterministic Crowding with Probabilistic Replacement. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '99, pages 409–416.

Mitchell, M. (1998). *An Introduction to Genetic Algorithms*. MIT Press.

Noman, N. and Iba, H. (2008). Accelerating Differential Evolution Using an Adaptive Local Search. *IEEE Transactions on Evolutionary Computation*, 12(1):107–125.

References VIII

Pelikan, M. and Goldberg, D. E. (2000). Genetic Algorithms, Clustering, and the Breaking of Symmetry. In *Parallel Problem Solving from Nature – PPSN VI*, pages 385–394. Springer Berlin Heidelberg.

Pétrowski, A. (1996). A clearing procedure as a niching method for genetic algorithms. In *Proceedings of IEEE International Conference on Evolutionary Computation*, pages 798–803.

Preuss, M. (2015). *Multimodal Optimization by Means of Evolutionary Algorithms*. Natural Computing Series. Springer International Publishing, 1st edition.

Sareni, B. and Krahenbuhl, L. (1998). Fitness sharing and niching methods revisited. *IEEE Transactions on Evolutionary Computation*, 2(3):97–106.

Shir, O. M. (2012). Niching in Evolutionary Algorithms. In *Handbook of Natural Computing*, pages 1035–1069. Springer Berlin Heidelberg.

References IX

Singh, G. and Deb, K. (2006). Comparison of Multi-modal Optimization Algorithms Based on Evolutionary Algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '06, pages 1305–1312. ACM.

Squillero, G. and Tonda, A. (2016). Divergence of character and premature convergence: A survey of methodologies for promoting diversity in evolutionary optimization. *Information Sciences*, 329:782–799.

Stützle, T. G. (1999). *Local search algorithms for combinatorial problems - analysis, improvements, and new applications*, volume 220 of *DISKI. Dissertationen zur Künstlichen Intelligenz [Dissertations on Artificial Intelligence]*. Infix, Sankt Augustin.

Yagiura, M. and Ibaraki, T. (2001). On metaheuristic algorithms for combinatorial optimization problems. *Systems and Computers in Japan*, 32(3):33–55.

References X

Yin, X. and Germay, N. (1993). A Fast Genetic Algorithm with Sharing Scheme Using Cluster Analysis Methods in Multimodal Function Optimization. In *Artificial Neural Nets and Genetic Algorithms*, pages 450–457. Springer Vienna.

Zitzler, E. and Künzli, S. (2004). Indicator-Based Selection in Multiobjective Search. In *Parallel Problem Solving from Nature – PPSN VIII*, pages 832–842. Springer Berlin Heidelberg.