Artificial Life
Summer 2025

Learning from Nature
Evolutionary Algorithms
Optimization Inspired by Biology

Master Computer Science [MA-INF 4201]
Mon 14:15 – 15:45, HSZ, HS-2

Dr. Nils Goerke, Autonomous Intelligent Systems,
Department of Computer Science, University of Bonn

Overview:

- Some Basics on Optimization
- Evolutionary Computation
- Historic Remarks
- Different Approaches
- Idea of Evolutionary Algorithms (EA)
- EA Steps
  - Individual, Genome, Fitness, Population
  - Parent selection
  - Inheritance
  - Mutation
  - Fitness evaluation
  - External selection

Optimization (some basics):

Depending on the background and the context, optimization is mainly seen as a part of :

- Operations Research
- Artificial Intelligence, or more precise
- Computational Intelligence.

Some Artificial Life approaches are directly working in the following subsets of optimization:

- Stochastic Optimization
- Meta heuristic Optimization

Optimization (some basics):

(Global) optimization is typically based on optimizing a criterion of some objective function **f(s)**:

- minimizing some costs
- maximizing some fitness

Optimization is the process of finding a position **s*** in N-dimensional search space **S** that optimizes the objective function **f(s)** $\in \mathbb{R}$,

f(s*) $\leq$ **f(s)** $\forall\, s \in \mathbf{S}$ (in case of minimization)

f(s*) $\geq$ **f(s)** $\forall\, s \in \mathbf{S}$ (in case of maximization)

Minimization vs. Maximization:

Minimization is a kind of optimization with respect to (scalar) **costs**. Costs are typically defined to have a known lower bound: usually „zero“.

Maximization is a kind of optimization with respect to a (scalar) objective function; like **fitness**, profit, or performance. A realistic upper bound for this objective function is often not known in advance.

Re-formulating a **maximization** problem into a **minimization** problem is often possible, but not always advisable because of a probable non-linear complete re-scaling of the objective function.

Global vs. Local Optimization:

Global optimization attempts to find the absolute best (global) optimum for a given problem.
Finding a global optimum (there might be several) is definitively one of the the hardest tasks in optimization.

Global vs. Local Optimization:

Global optimization attempts to find the absolute best (global) optimum for a given problem.
Finding a global optimum (there might be several) is definitively one of the the hardest tasks in optimization.

Beside the global optimum, there can be local optima (typically there are a lot of them).

Local optimization is attempting to find an extremum in the vicinity of a starting point, that is good, or at least sufficient for the given problem.
There is a chance to find a global optimum, even with local optimization methods.

Global vs. Local Optimization:

Global optimization attempts to find the absolute best (global) optimum for a given problem.
Finding a global optimum (there might be several) is definitively one of the the hardest tasks in optimization.

Beside the global optimum, there can be local optima (typically there are a lot of them).

Local optimization is attempting to find an extremum in the vicinity of a starting point, that is good, or at least sufficient for the given problem.
There is a chance to find a global optimum, even with local optimization methods; **but you never know.**

Stochastic Optimization:

Stochastic optimization is including a stochastic component into the optimization process. The stochastic component can be part of every aspect of the process.

Stochastic Optimization:

Stochastic optimization is including a stochastic component into the optimization process. The stochastic component can be part of every aspect of the process.

In can be:

- in the data,
- in the objective function,
- in the optimization method as explicit component,
- in the schedule of the optimization process.

Stochastic Optimization:

Stochastic optimization is including a stochastic component into the optimization process. The stochastic component can be part of every aspect of the process.

In can be:

- in the data,
- in the objective function,
- in the optimization method as explicit component,
- in the schedule of the optimization process.

The stochastic component can be an inevitable property of the application one has to cope with,
or can be included deliberately to support the algorithm.

Stochastic Optimization:

There is a variety of other approaches from the field of Stochastic optimization:

- Random Search
- Random Optimization
- Monte Carlo Methods
- Simulated Annealing
- Single Start
- Multi Start
- Random Search Direction Methods
- Hill Climbing
- …

Optimization:

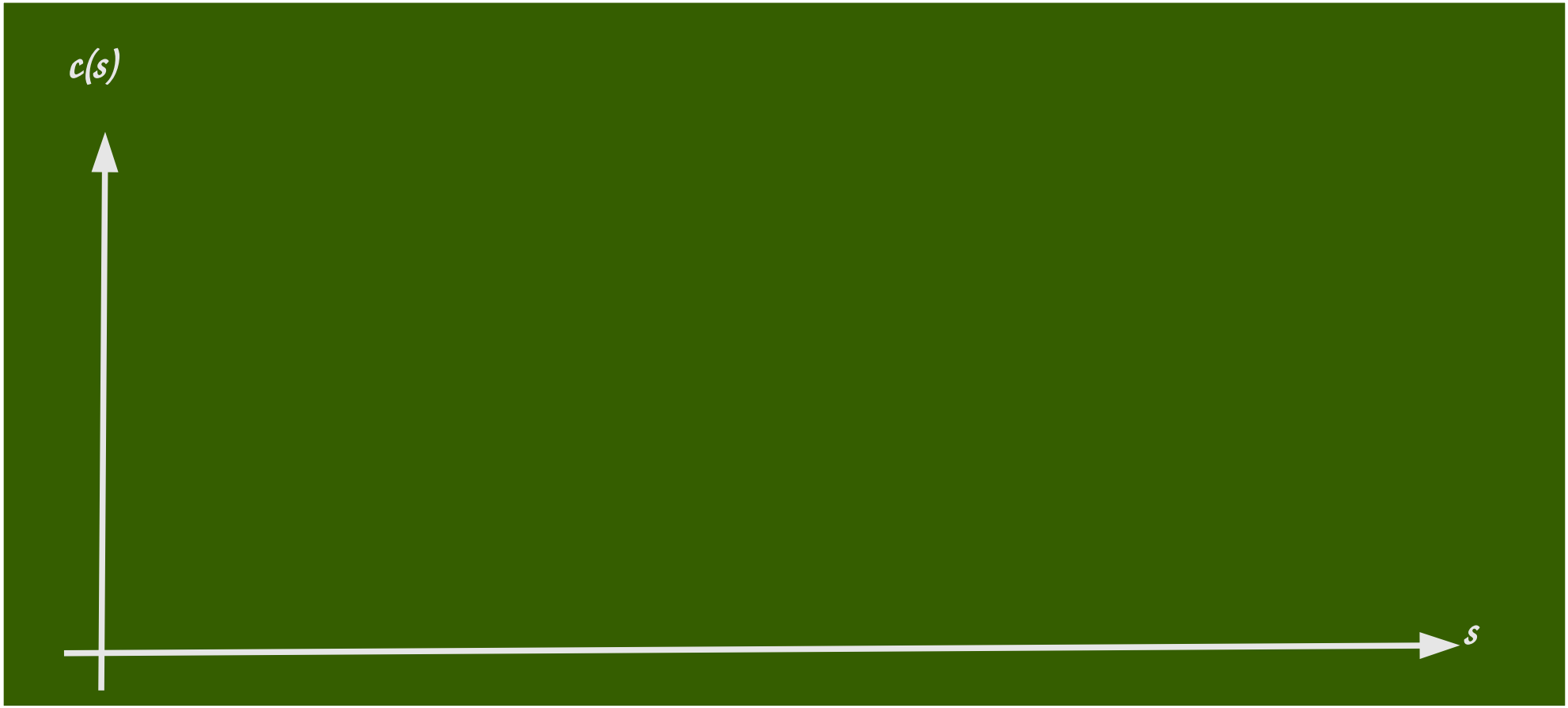
Draw a **nasty cost function** on the blackboard:

Optimization:

Draw a **nasty cost function** on the blackboard:

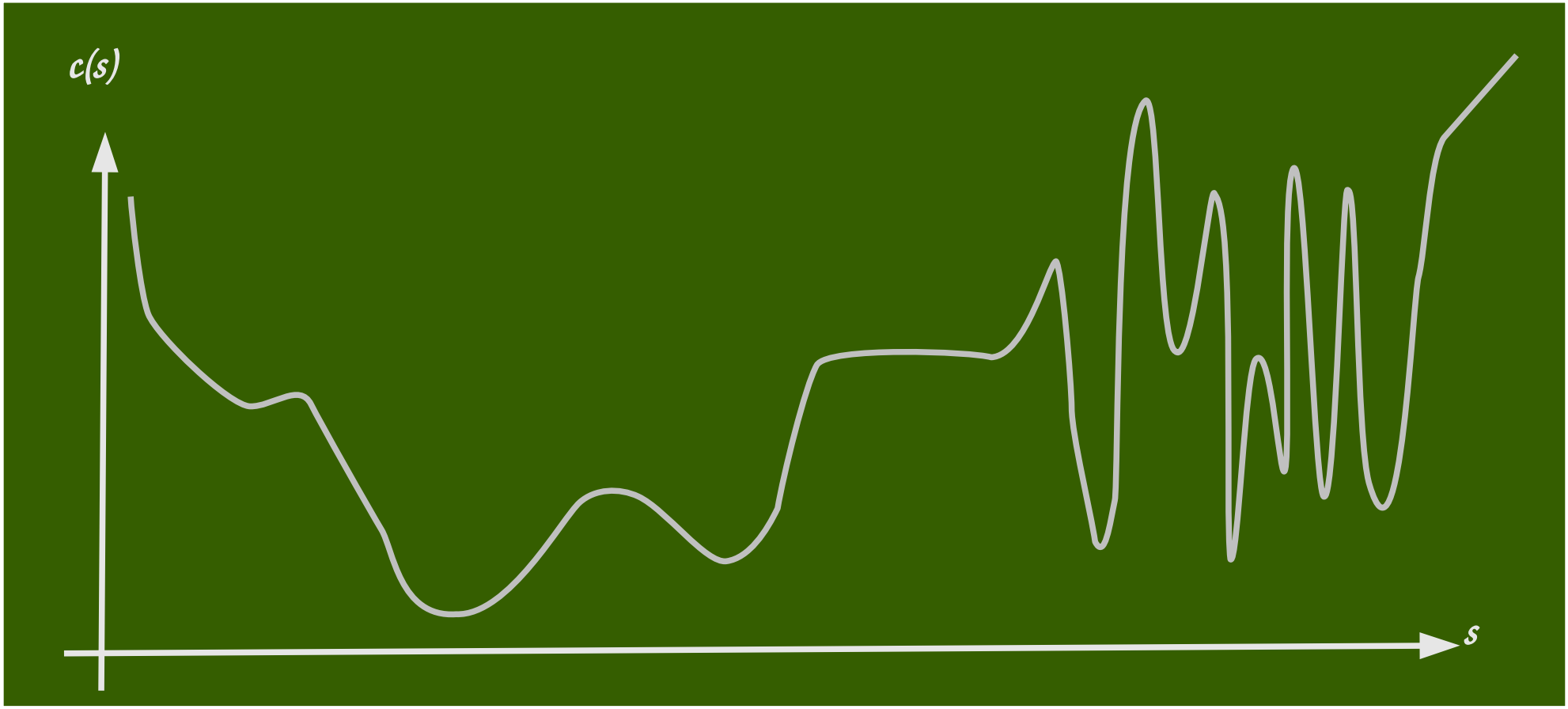
Optimization:

Draw a **nasty cost function** on the blackboard:



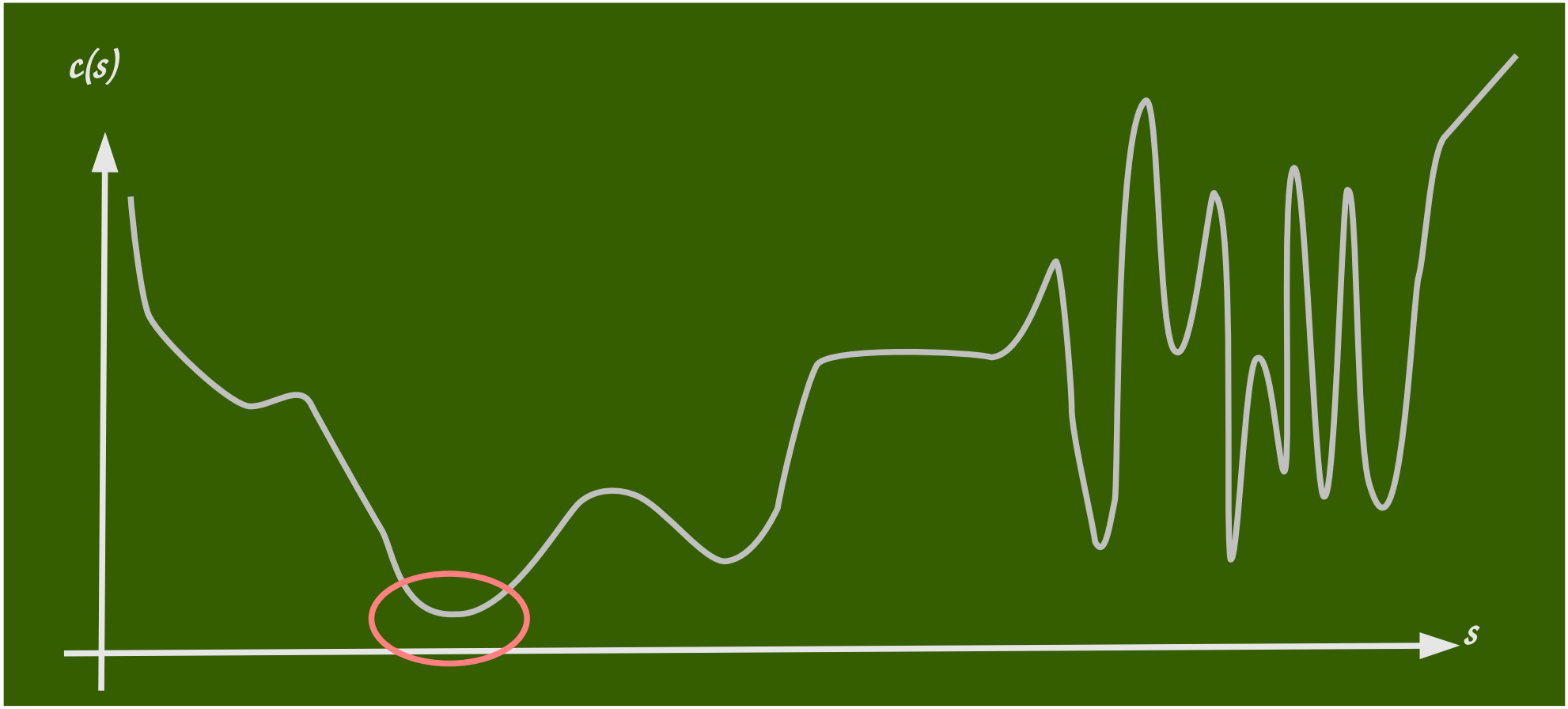
Optimization:

Draw a **nasty cost function** on the blackboard:



Optimization:

Draw a **nasty cost function** on the blackboard:

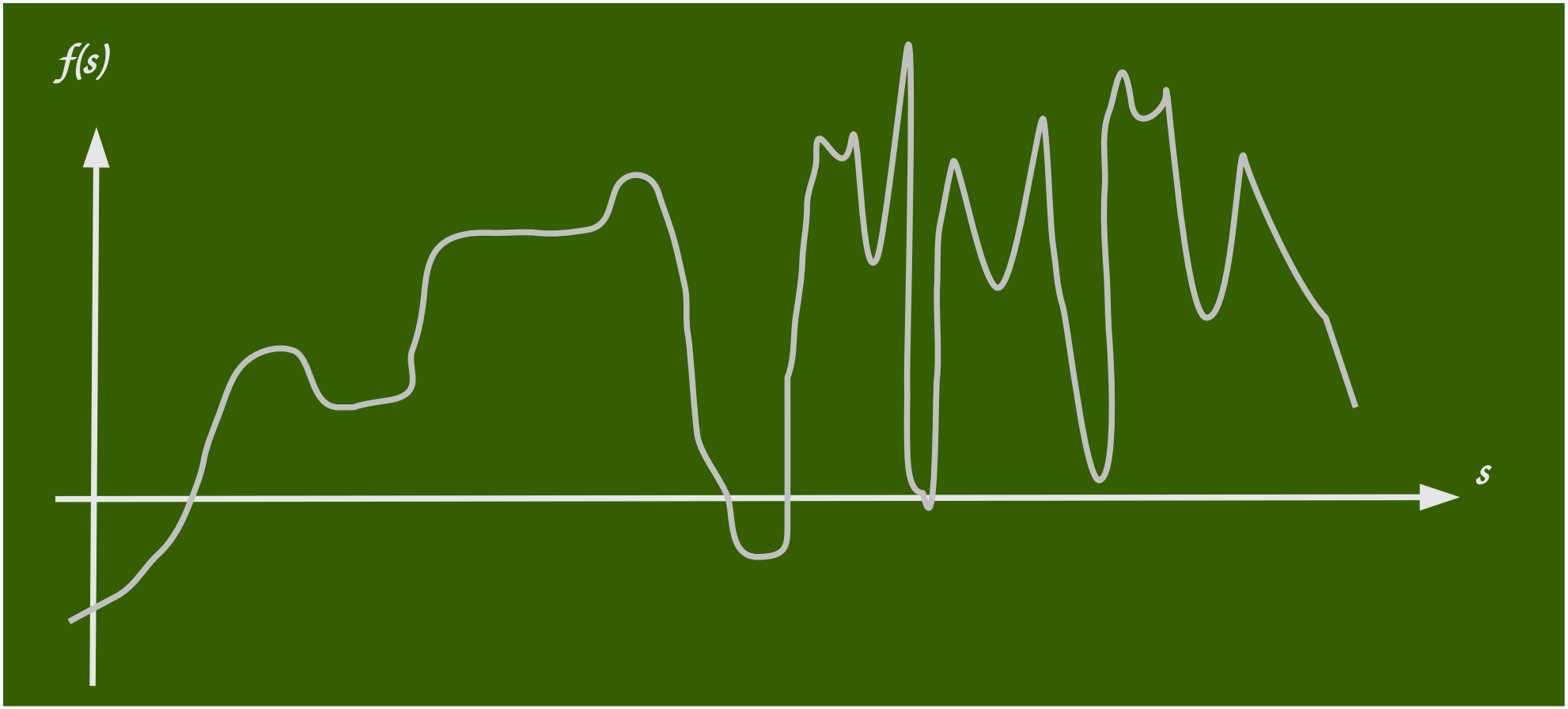


Optimization:

Draw a **nasty fitness function** on the blackboard:

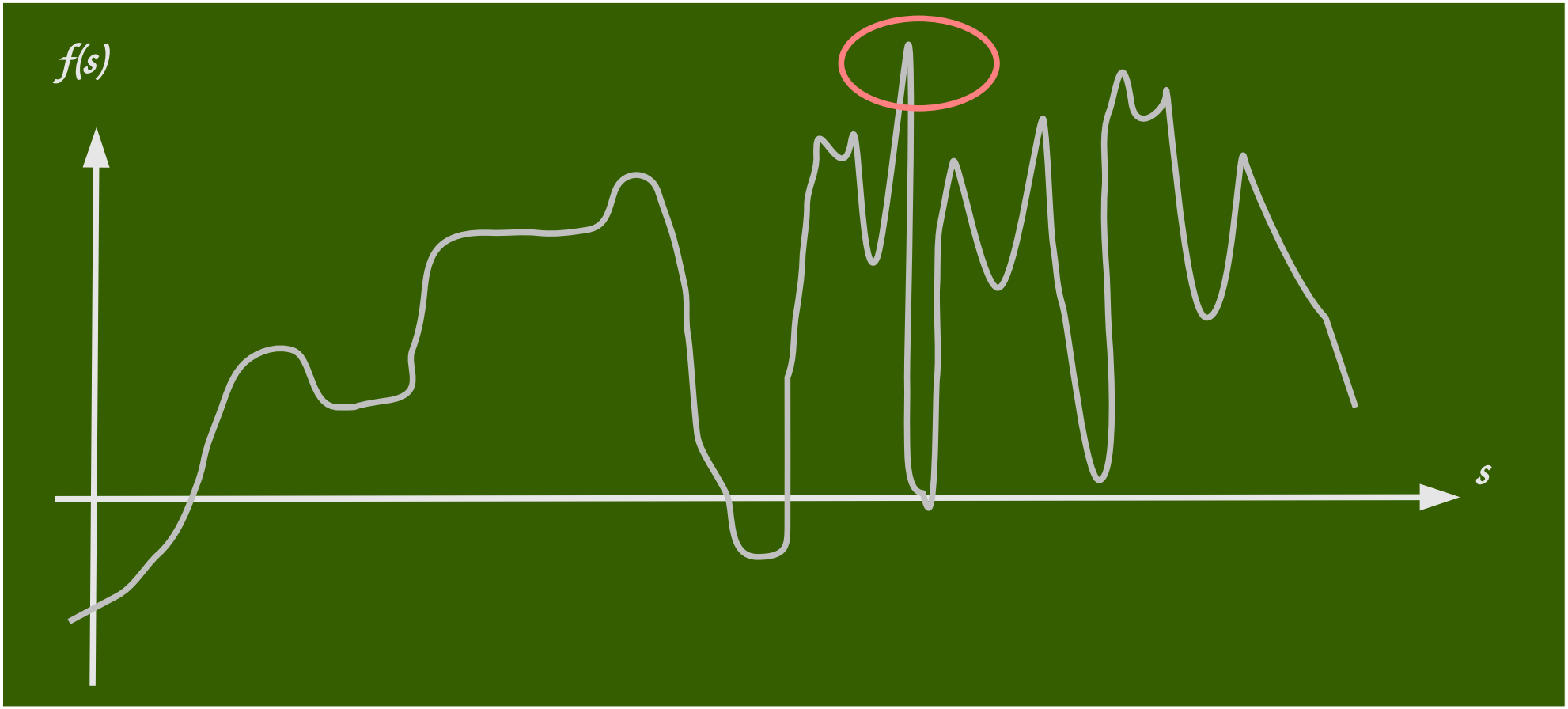
Optimization:

Draw a **nasty fitness function** on the blackboard:



Optimization:

Draw a **nasty fitness function** on the blackboard:



Optimization:

A typical **playground** to investigate the capabilities of optimization methods are **test functions** to be minimized.

Over the years, many researchers have invested a lot of time to design adequate and appealing test functions to demonstrate and evaluate the results of their optimization methods.

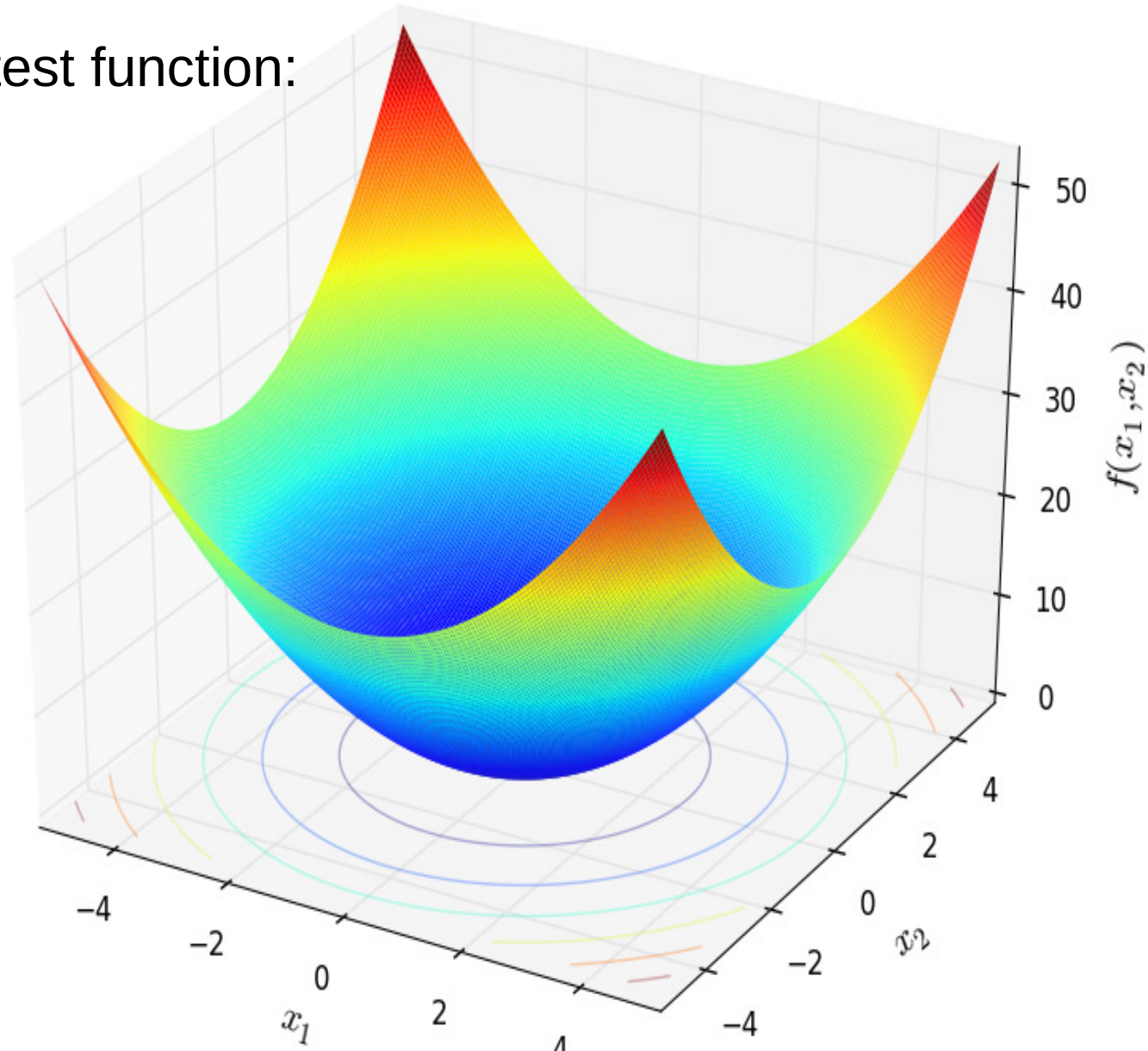
Meanwhile, many collections of such functions can be found.
A nice collections is:

http://infinity77.net/global_optimization/test_functions.html#test-functions-index

Optimization:

An easy to understand test function:

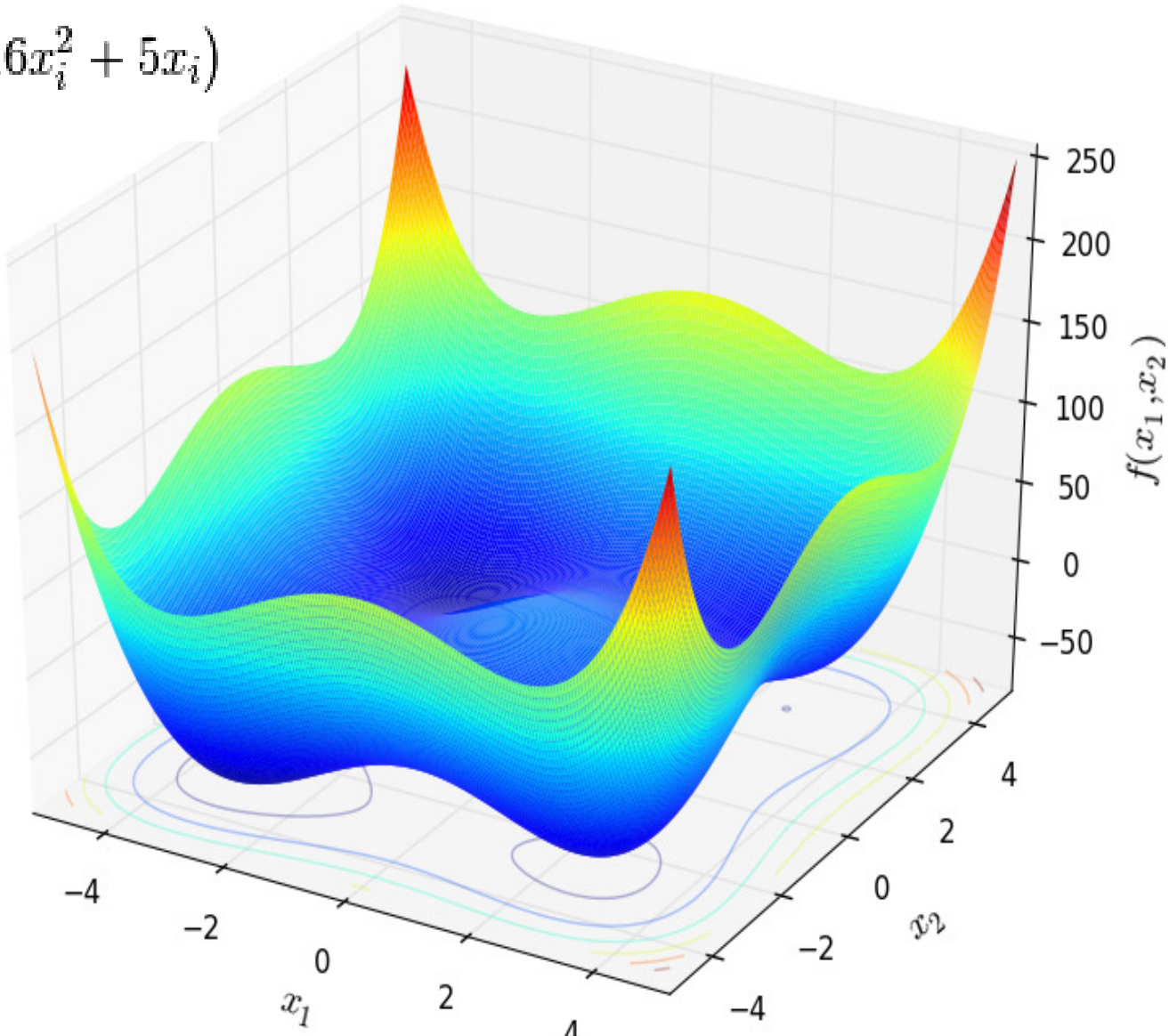
$$f_{\text{Sphere}}(\mathbf{x}) = \sum_{i=1}^{n} x_i^2$$



http://infinity77.net/global_optimization/test_functions_nd_S.html#go_benchmark.Sphere

Optimization:

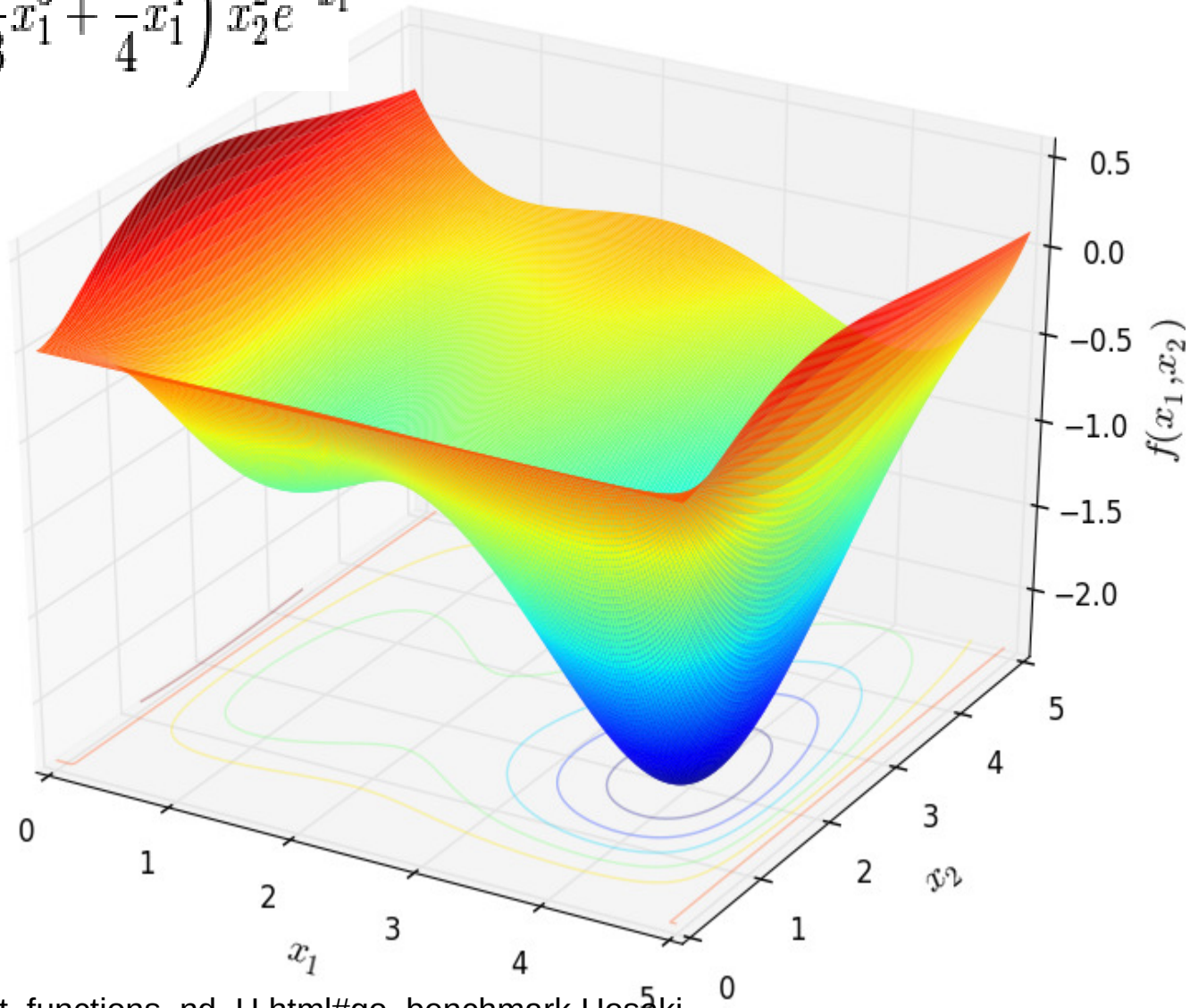
$$f_{\text{StyblinskiTang}}(\mathbf{x}) = \sum_{i=1}^{n} \left(x_i^4 - 16x_i^2 + 5x_i\right)$$



http://infinity77.net/global_optimization/test_functions_nd_S.html#go_benchmark.StyblinskiTang

Optimization:

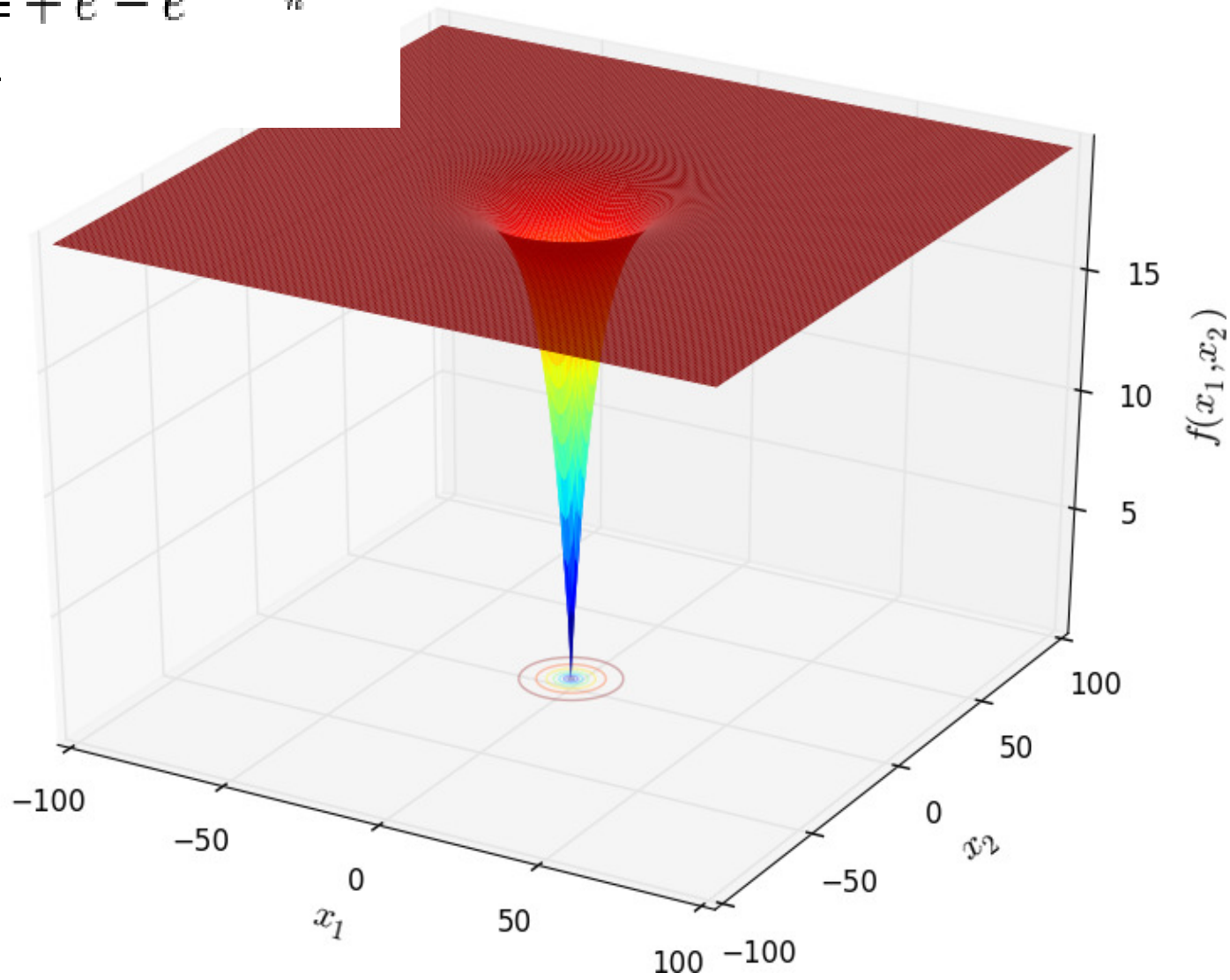
$$f_{\text{Hosaki}}(\mathbf{x}) = \left(1 - 8x_1 + 7x_1^2 - \frac{7}{3}x_1^3 + \frac{1}{4}x_1^4\right) x_2^2 e^{-x_1}$$



http://infinity77.net/global_optimization/test_functions_nd_H.html#go_benchmark.Hosaki

Optimization:

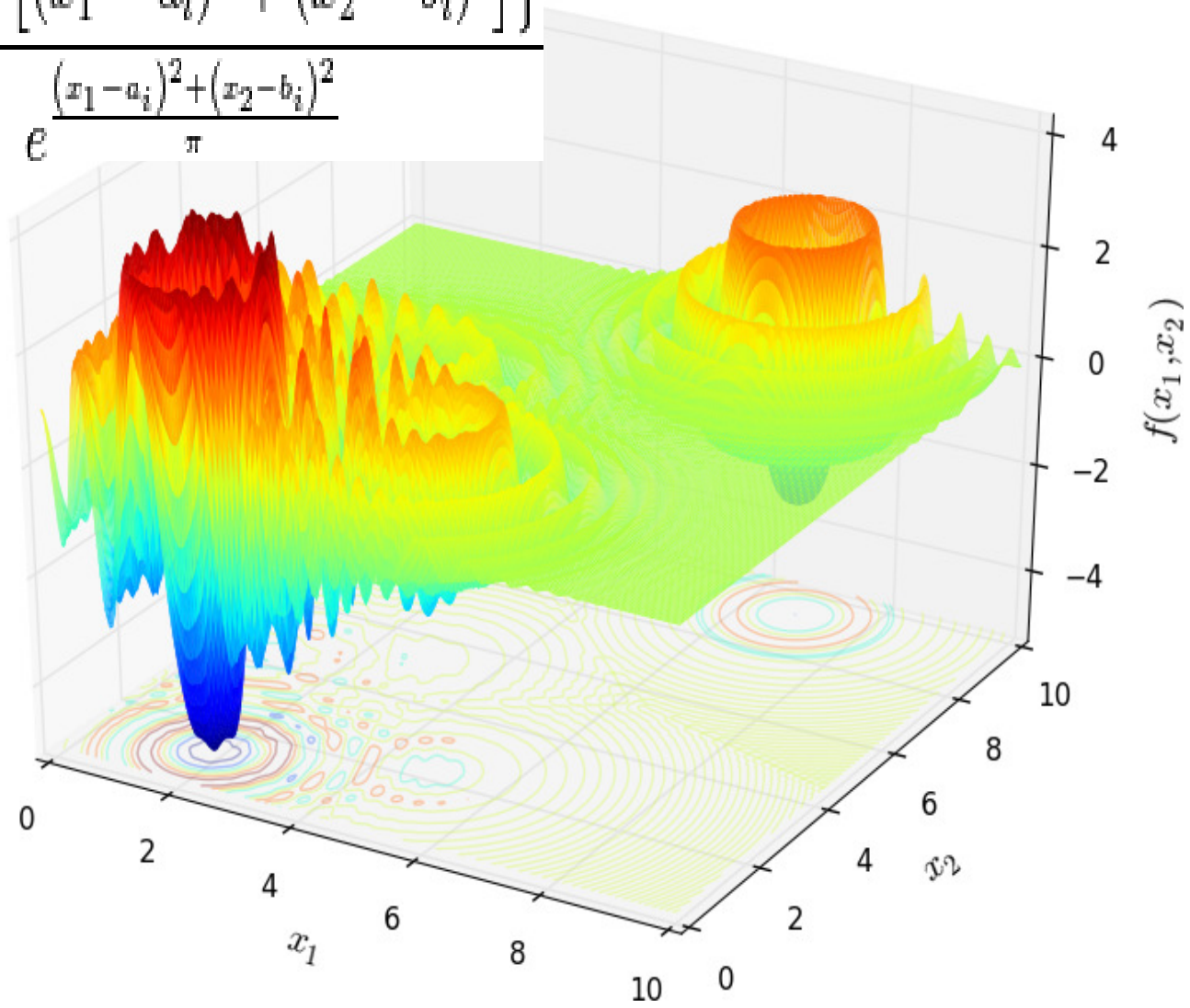
$$f_{\text{Easom}}(\mathbf{x}) = a - \frac{a}{e^{b\sqrt{\frac{\sum_{i=1}^{n} x_i^2}{n}}}} + e - e^{\frac{\sum_{i=1}^{n} \cos(cx_i)}{n}}$$



http://infinity77.net/global_optimization/test_functions_nd_E.html#go_benchmark.Easom

Optimization:

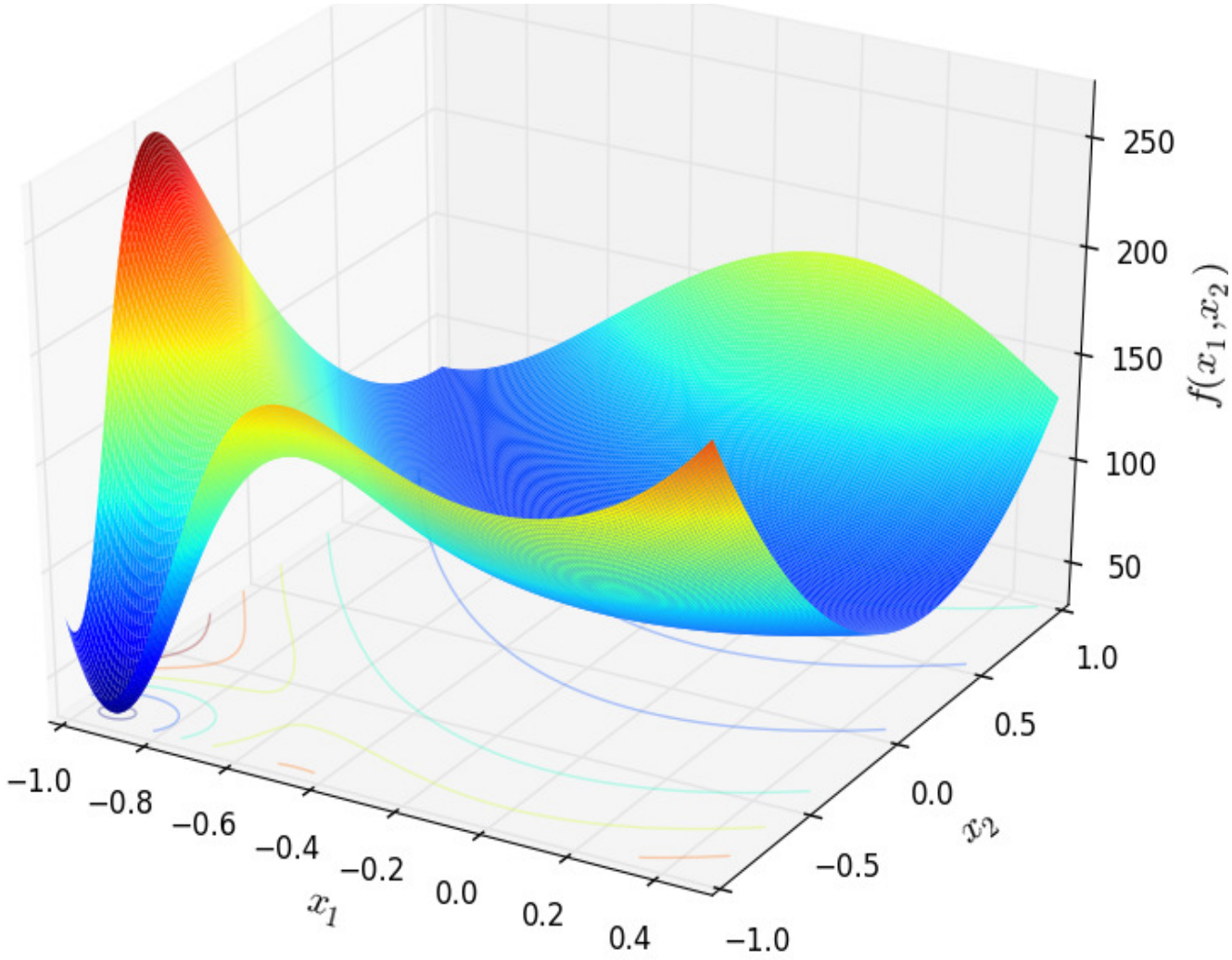
$$f_{\text{Langermann}}(\mathbf{x}) = -\sum_{i=1}^{5} \frac{c_i \cos\left\{\pi\left[(x_1 - a_i)^2 + (x_2 - b_i)^2\right]\right\}}{e^{\frac{(x_1-a_i)^2+(x_2-b_i)^2}{\pi}}}$$



http://infinity77.net/global_optimization/test_functions_nd_L.html#go_benchmark.Langermann

Optimization:

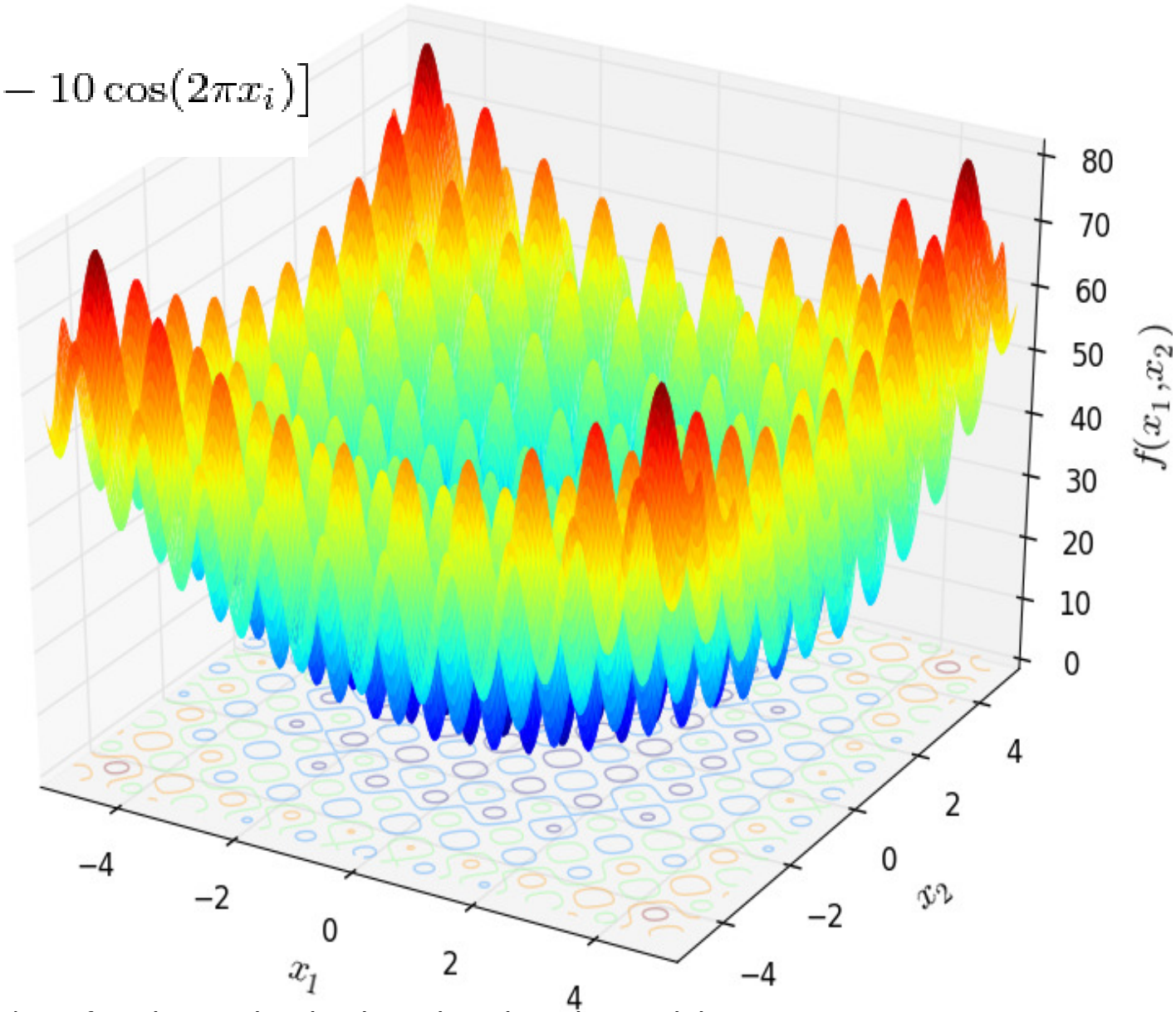
$$f_{\text{RosenbrockModified}}(\mathbf{x}) = 74 + 100(x_2 - x_1^2)^2 + (1 - x_1)^2 - 400e^{-\frac{(x_1+1)^2+(x_2+1)^2}{0.1}}$$



http://infinity77.net/global_optimization/test_functions_nd_R.html#go_benchmark.RosenbrockModified

Optimization:

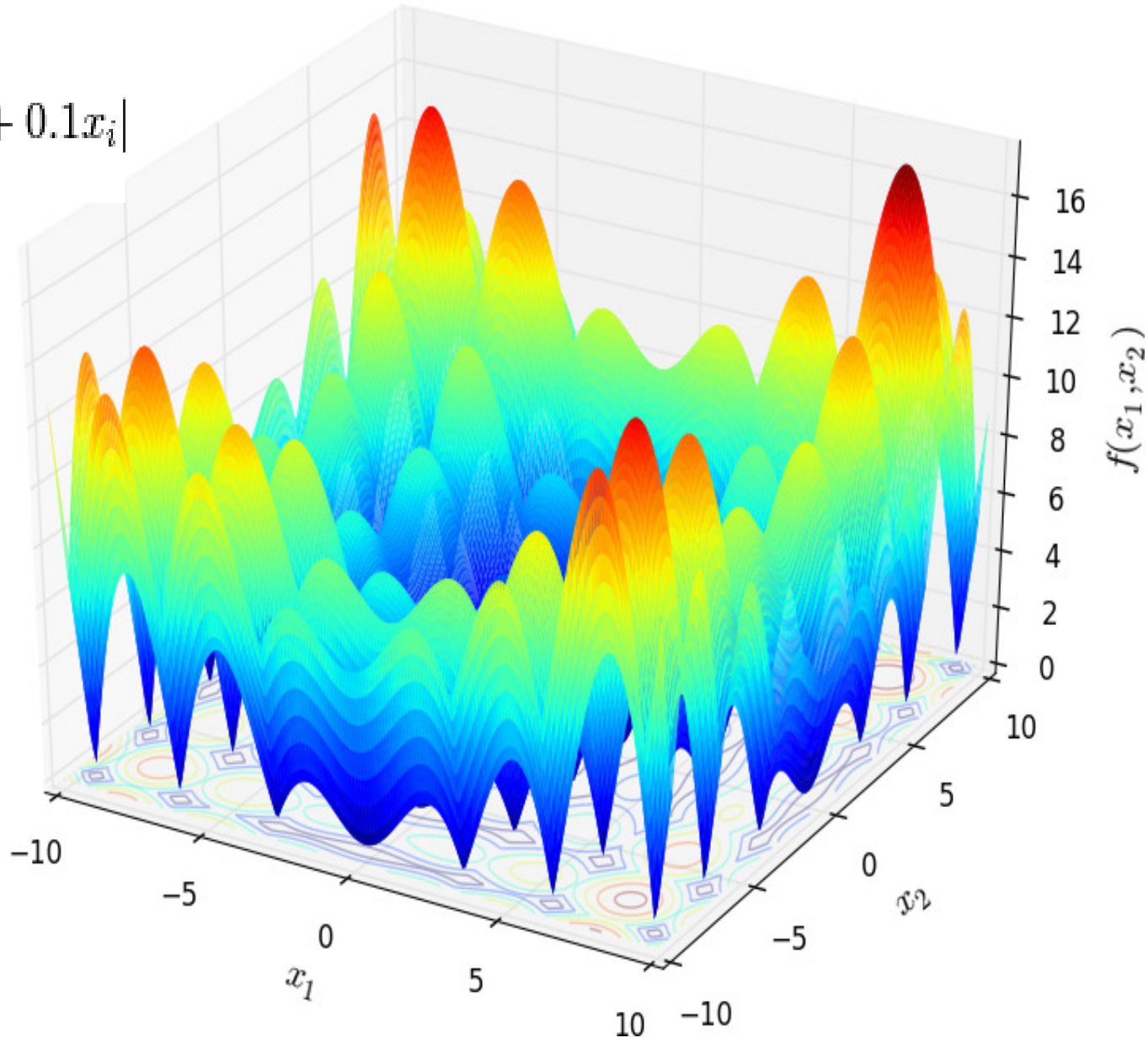
$$f_{\text{Rastrigin}}(\mathbf{x}) = 10n \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) \right]$$



http://infinity77.net/global_optimization/test_functions_nd_R.html#go_benchmark.Rastrigin

Optimization:

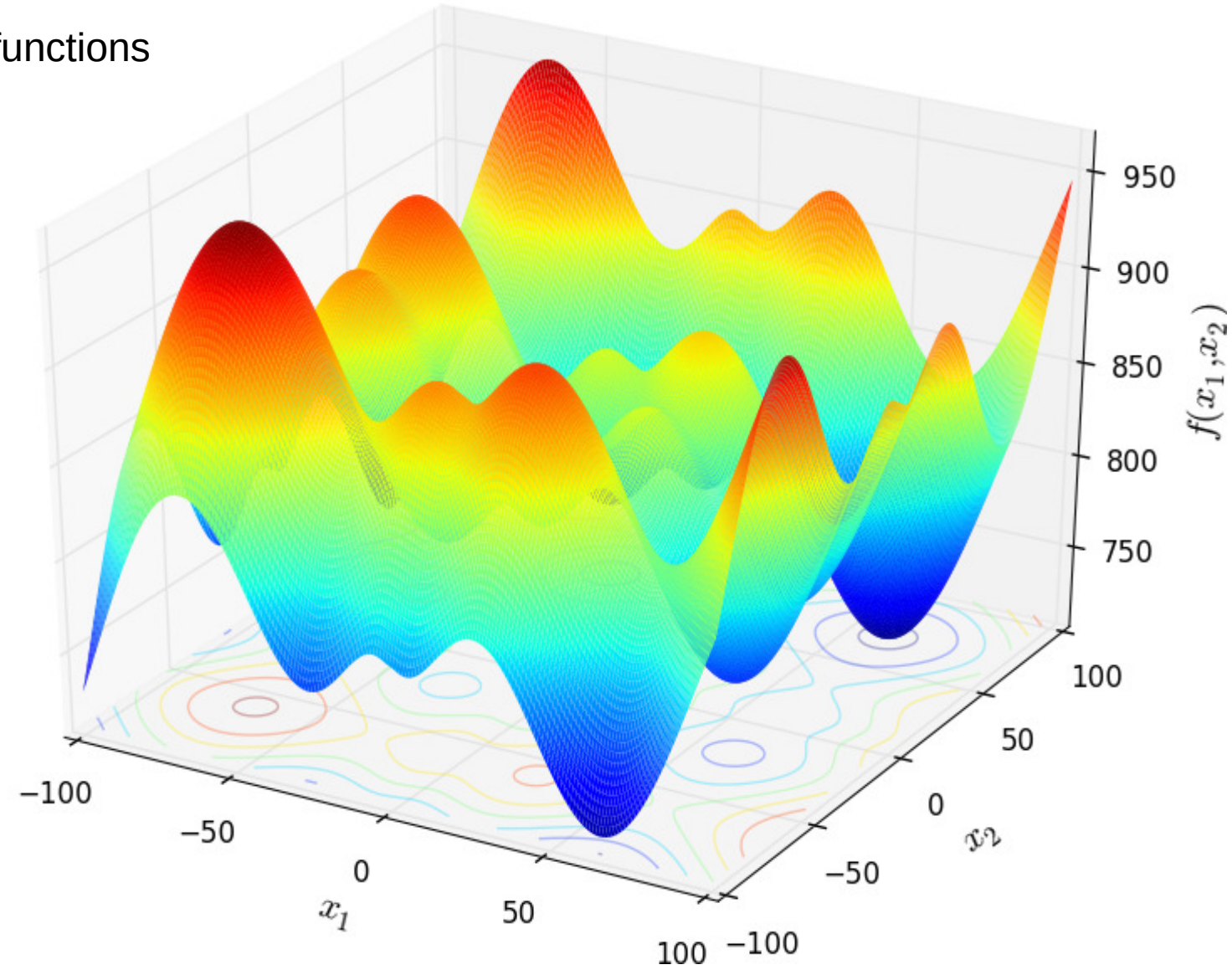
$$f_{\text{Alpine01}}(\mathbf{x}) = \sum_{i=1}^{n} |x_i \sin(x_i) + 0.1 x_i|$$



http://infinity77.net/global_optimization/test_functions_nd_A.html#go_benchmark.Alpine01

Optimization:

One of Schwefel's test functions



http://infinity77.net/global_optimization/test_functions_nd_A.html#go_benchmark.Schwefel26

Optimization:

Below is a list of common **test functions (to minimize)** that are used to investigate the capabilities of optimization methods:

- Schwefel's Function
- Generalized Rosenbrock's Function
- Rastrigin's Function
- Ackley's (path) Function
- Langermann's Function
- Michalewicz's Function
- Easoms Function
- Griewangk's Function
- Bohachevsky's Function
- Watson's Function
- Colville's Function
- ...

Overview:

- Some Basics on Optimization
- Evolutionary Computation
- Historic Remarks
- Different Approaches
- Idea of Evolutionary Algorithms (EA)
- EA Steps
  - Individual, Genome, Fitness, Population
  - Parent selection
  - Inheritance
  - Mutation
  - Fitness evaluation
  - External selection

Evolutionary Computation:

Optimization inspired by Biology

Evolutionary Computation (**EC**):

- What is it?
- Where does it come from?
- Where can it be found?
- Why is it interesting for Computer Science?

Historic Remarks, Different Approaches:

The field is at the moment denominated as **Evolutionary Computation (EC)**.

EC comprises several approaches and ideas

- that have been developed independently (60ies),
- that follow the same basic guideline and
- that have (later) been found to be so closely related to each other,
- that they can be seen as different variants or dialects of the same basic principles.

Historic Remarks, Different Approaches:

Evolutionary Computation (EC)

Swarm Behavior / Swarm Intelligence
- Ant Algorithm
- Ant Colony Optimization
- *Particle Swarm Optimization*

Evolutionary Algorithms (EA)
- Genetic Algorithms (GA)
- Genetic Programming (GP)
- Evolutionary Strategies (ES)
- Evolutionary Programming (EP)

Historic Remarks, Different Approaches:

Evolutionary Computation (EC)

Swarm Behavior / Swarm Intelligence
- Ant Algorithm
- Ant Colony Optimization
- *Particle Swarm Optimization*

Evolutionary Algorithms (EA)
- Genetic Algorithms (GA)
- Genetic Programming (GP)
- Evolutionary Strategies (ES)
- Evolutionary Programming (EP)

Historic Remarks, Different Approaches:

Evolutionary Computation (EC)

Evolutionary Algorithms (EA)

-
-
-
-

Historic Remarks, Different Approaches:

Evolutionary Computation (EC)

Evolutionary Algorithms (EA)

- **Genetic Algorithms (GA)**
  John Henry Holland (1975)
-
-
-

Historic Remarks, Different Approaches:

Evolutionary Computation (EC)

Evolutionary Algorithms (EA)

- **Genetic Algorithms (GA)**
  John Henry Holland (1975)
- **Evolutionary Strategies (ES)**
  Ingo Rechenberg (1965), Hans-Paul Schwefel (1970)
-
-

Historic Remarks, Different Approaches:

Evolutionary Computation (EC)

Evolutionary Algorithms (EA)

- **Genetic Algorithms (GA)**
  John Henry Holland (1975)
- **Evolutionary Strategies (ES)**
  Ingo Rechenberg (1965), Hans-Paul Schwefel (1970)
- **Evolutionary Programming (EP)**
  Lawrence J. Fogel (1964)
-

Historic Remarks, Different Approaches:

Evolutionary Computation (EC)

Evolutionary Algorithms (EA)

- **Genetic Algorithms (GA)**
  John Henry Holland (1975)
- **Evolutionary Strategies (ES)**
  Ingo Rechenberg (1965), Hans-Paul Schwefel (1970)
- **Evolutionary Programming (EP)**
  Lawrence J. Fogel (1964)
- **Genetic Programming (GP)**
  N.A.Barricelli (1954), R.M.Friedberg (1958)

Overview:

- Evolutionary Computation
- Historic Remarks
- Different Approaches
- Idea of Evolutionary Algorithms (EA)
- EA Steps
  - Individual, Genome, Fitness, Population
  - Parent selection
  - Inheritance
  - Mutation
  - Fitness evaluation
  - External selection

Optimization (some basics):

Optimization is typically based on optimizing a criterion of some objective function **f(s)**:

- minimizing some costs
- maximizing some fitness

Optimization is the process of finding a position **s*** in search space **S** that optimizes the objective function **f(s)** $\in \mathbb{R}$,

f(s*) $\leq$ **f(s)** $\forall\, s \in \mathbf{S}$ (in case of minimization)

f(s*) $\geq$ **f(s)** $\forall\, s \in \mathbf{S}$ (in case of maximization)

Global vs. Local Optimization:

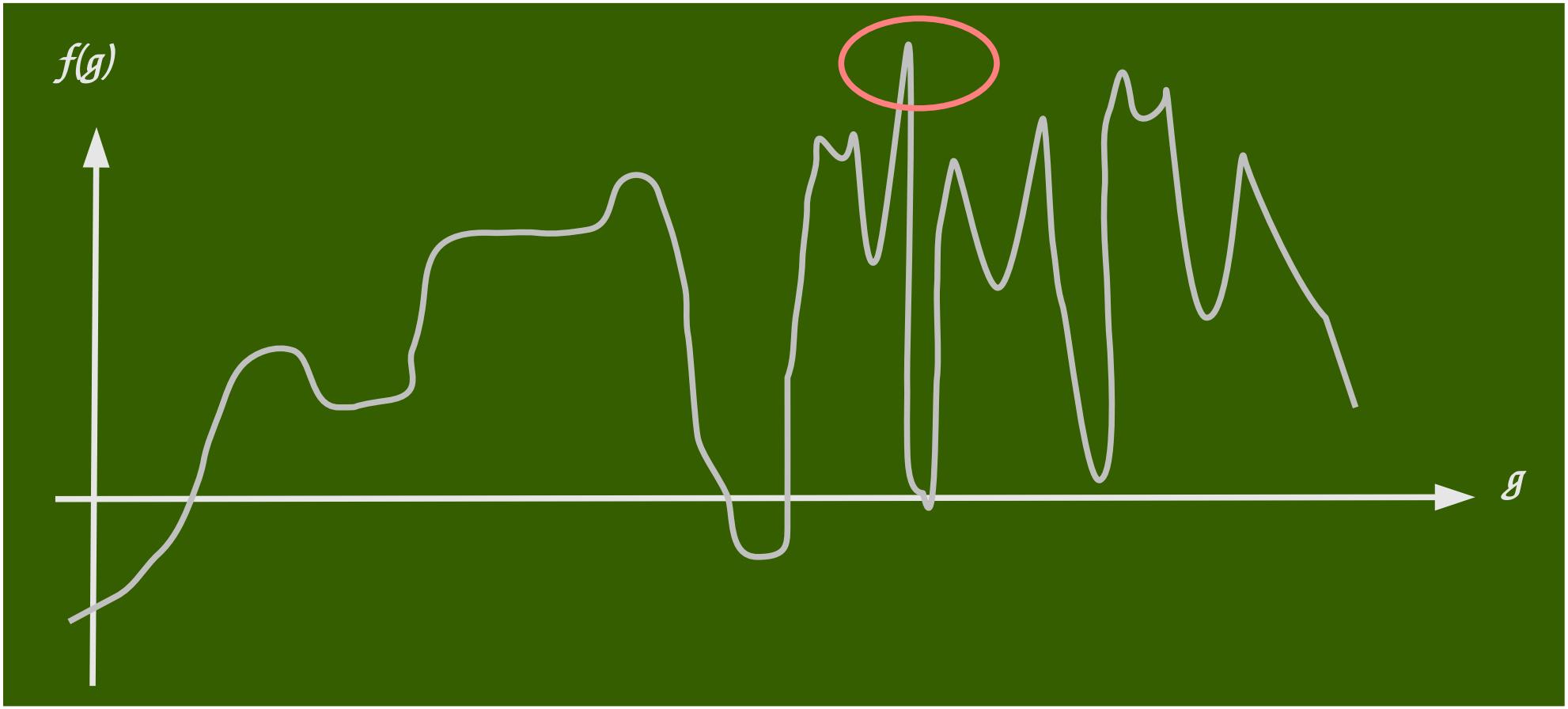
Global optimization attempts to find the absolute best (global) optimum for a given problem.
Finding a global optimum (there might be several) is definitively one of the the hardest tasks in optimization.

Beside the global optimum, there can be local optima (typically there are a lot of them).

Local optimization is attempting to find an extremum in the vicinity of a starting point, that is good, or at least sufficient for the given problem.
There is a chance to find a global optimum, even with local optimization methods; **but you never know.**

Optimization:

Draw a **nasty fitness function** on the blackboard:



Overview:

- Evolutionary Computation
- Historic Remarks
- Different Approaches
- Optimization, some basics
- **Idea of Evolutionary Algorithms (EA)**
- EA Steps
  - Individual, Genome, Fitness, Population
  - Parent selection
  - Inheritance
  - Mutation
  - Fitness evaluation
  - External selection

EA:

Example: Fkt Maximum

I. Rechenbergs tubing problem

The fuel flows into the tube from the left. The fuel should leave the tube (top right) such that the resistance of the fluid is minimized.

The question is how the two tubes should be connected, i.e. what is the shape of the connecting tube?

From: AI Lab, University of Zurich

EA:

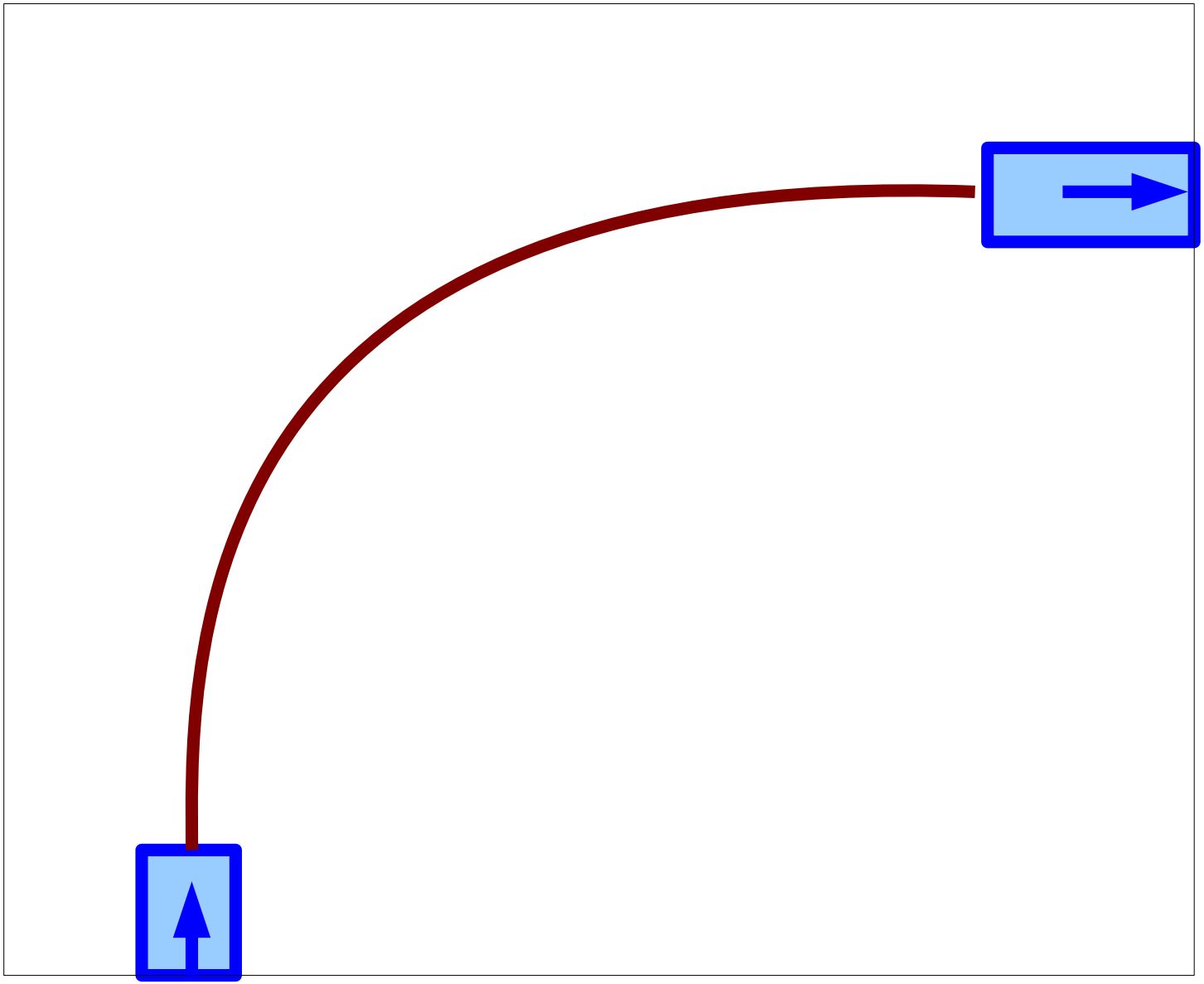
Example: Fkt Maximum

I. Rechenbergs tubing problem

The fuel flows into the tube from the left. The fuel should leave the tube (top right) such that the resistance of the fluid is minimized.

The question is how the two tubes should be connected, i.e. what is the shape of the connecting tube?

From: AI Lab, University of Zurich



EA:

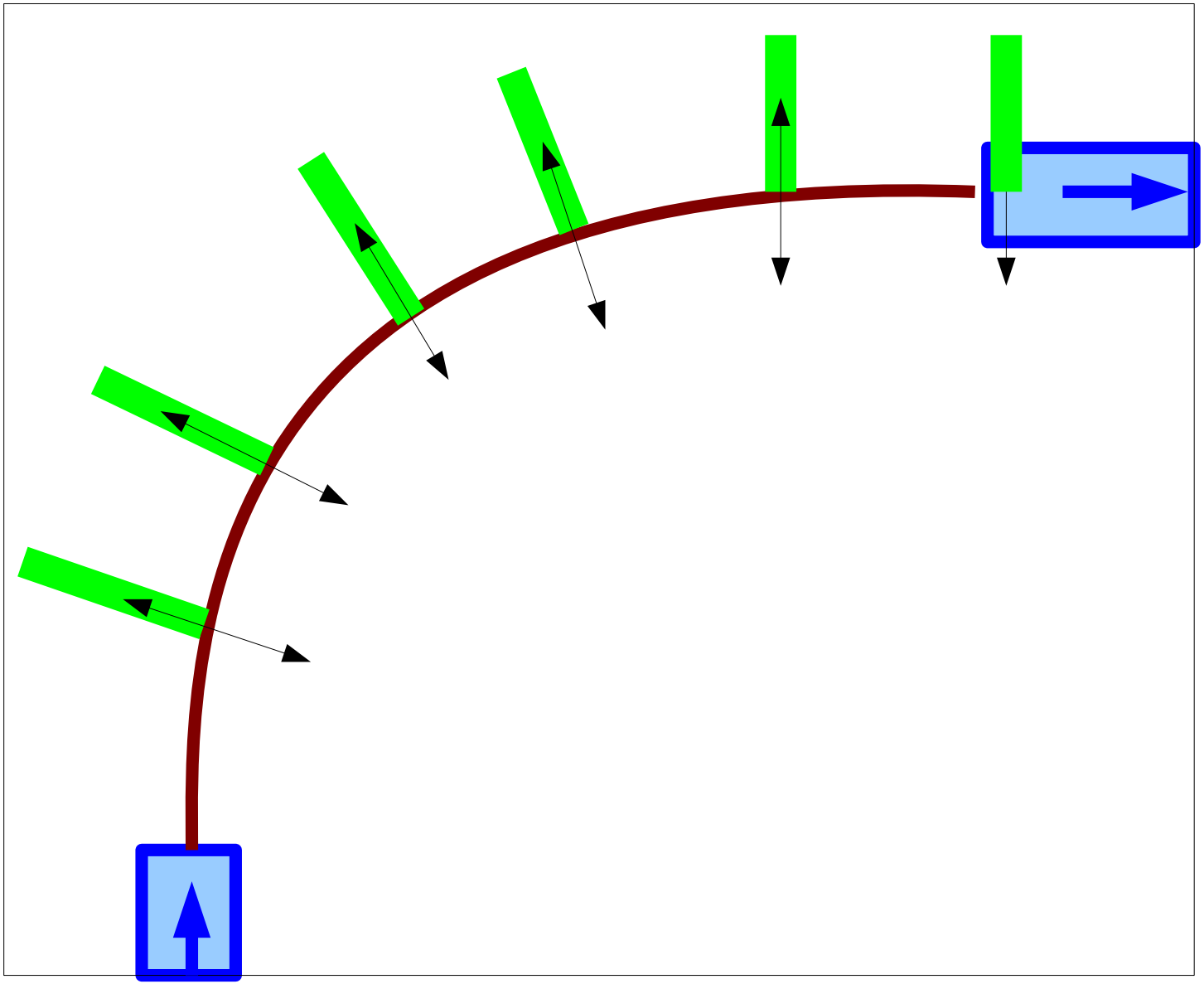
Example: Fkt Maximum

I. Rechenbergs tubing problem

The fuel flows into the tube from the left. The fuel should leave the tube (top right) such that the resistance of the fluid is minimized.

The question is how the two tubes should be connected, i.e. what is the shape of the connecting tube?

From: AI Lab, University of Zurich



EA:

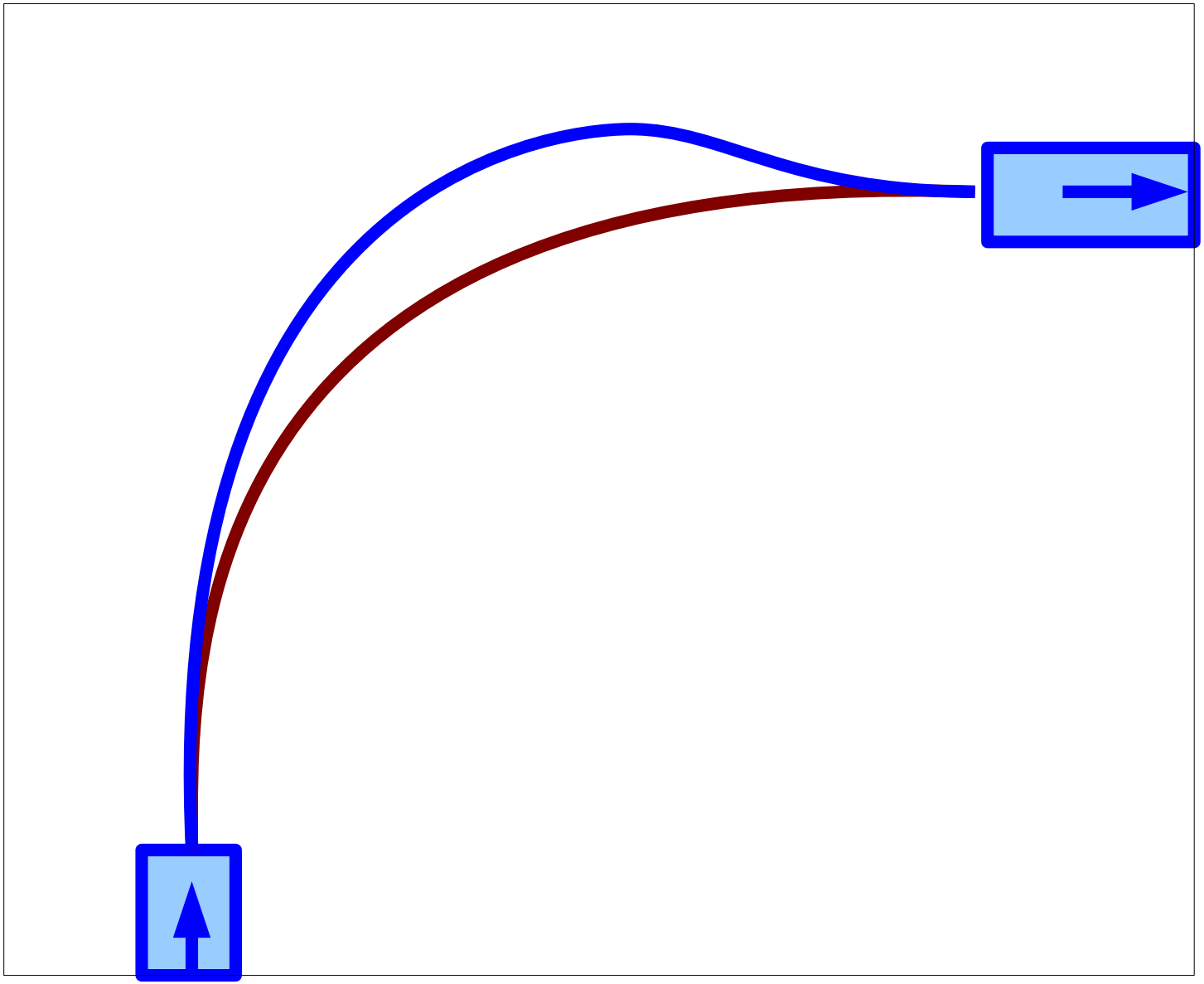
Example: Fkt Maximum

I. Rechenbergs tubing problem

The fuel flows into the tube from the left. The fuel should leave the tube (top right) such that the resistance of the fluid is minimized.

The question is how the two tubes should be connected, i.e. what is the shape of the connecting tube?

From: AI Lab, University of Zurich



EA:

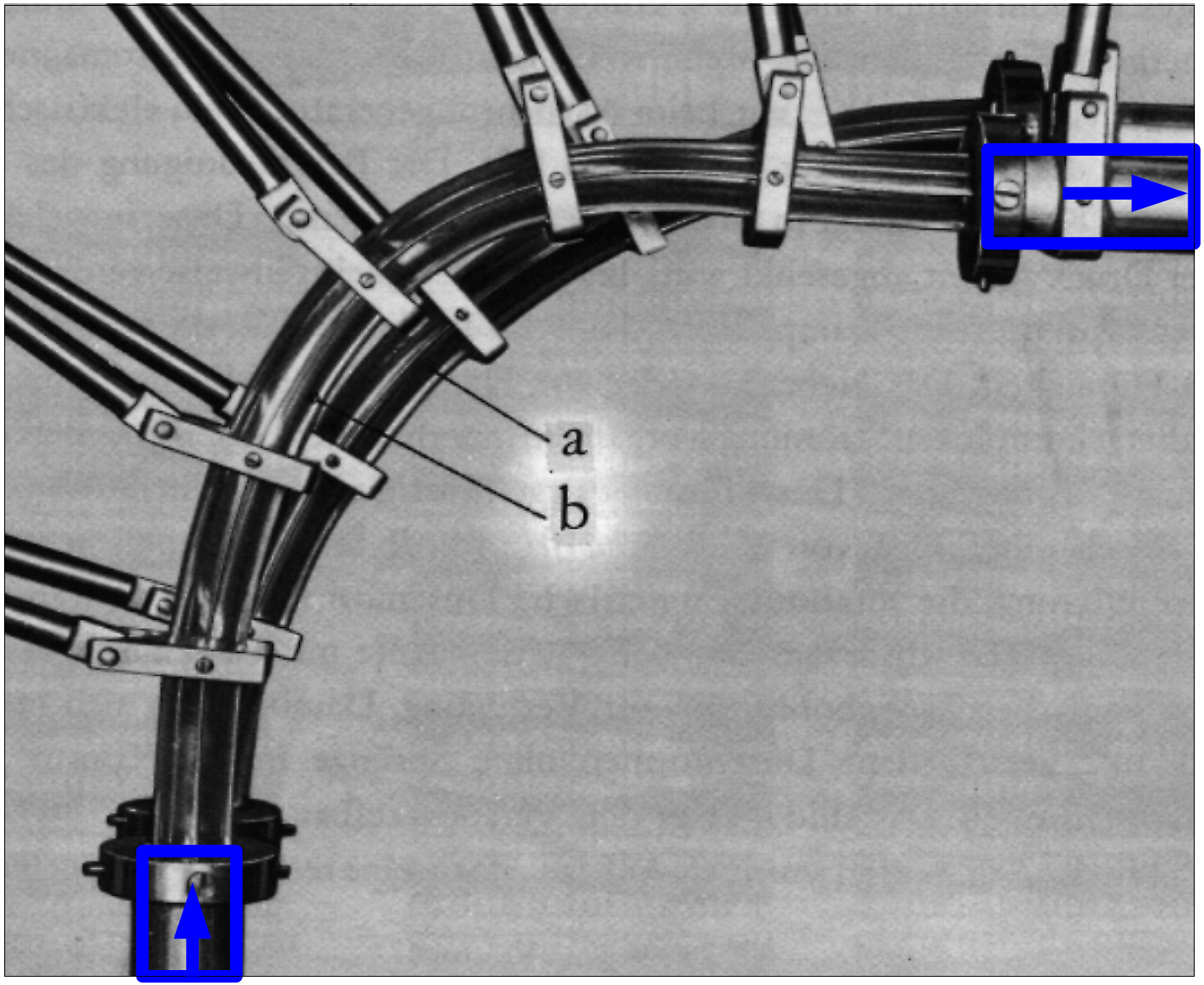
Example: Fkt Maximum

I. Rechenbergs tubing problem

The fuel flows into the tube from the left. The fuel should leave the tube (top right) such that the resistance of the fluid is minimized.

The question is how the two tubes should be connected, i.e. what is the shape of the connecting tube?

From: AI Lab, University of Zurich



Overview:

- Evolutionary Computation
- Historic Remarks
- Different Approaches
- **Idea of Evolutionary Algorithms (EA)**
- EA Steps
  - Individual, Genome, Fitness, Population
  - Parent selection
  - Inheritance
  - Mutation
  - Fitness evaluation
  - External selection

Idea of Evolutionary Algorithms (EA):

The whole family of algorithms denominated as **Evolutionary Algorithms** was inspired by a series of ideas and principles from living nature.

Especially the principles of Evolution (C.Darwin) and Reproduction and Inheritance (G.Mendel) and the discovery of the Genome, and the DNA (1953, J.Watson, F.Crick) have inspired the development of EAs in the 50ies and 60ies of the last century.

Thus, major parts of EAs are designed in analogy to the biological role model.

Different Approaches:

Evolutionary Computation (EC)

Evolutionary Algorithms (EA)

- **Genetic Algorithms (GA)**
  John Henry Holland (1975)
- **Evolutionary Strategies (ES)**
  Ingo Rechenberg (1965), Hans-Paul Schwefel (1970)
- **Evolutionary Programming (EP)**
  Lawrence J. Fogel (1964)
- **Genetic Programming (GP)**
  N.A.Barricelli (1954), R.M.Friedberg (1958)

Idea of Evolutionary Algorithms (EA):

Several aspects and terms of Evolutionary Algorithms are directly borrowed from biology:

- Population of Individuals
- Survival of the fittest
- Selection
- Inheritance
- Reproduction
- Mating
- Recombination
- Cross-Over
- Mutation
- Genome

Idea of Evolutionary Algorithms (EA):

Evolutionary Algorithms perform a stochastic optimization using a modified Multi-Start approach, (Multi-Hypothesis approach).

Instead of working with one single hypothesis, several different hypotheses will be regarded at the same time (working „quasi“ in parallel).

Within an Evolutionary Algorithm, each of the multi hypotheses is called an **Individual**.
The collective of all Individuals is called the **Population** of the **Evolutionary Algorithm**.

Idea of Evolutionary Algorithms (EA):

Each Individual of the Population will typically have a different position $\mathbf{s_p}$ in search space **S**, and thus, a different fitness $\mathbf{f(s_p)}$.

The position $\mathbf{s_p}$ in search space is defining the individual **p**;
$\mathbf{s_p}$ is called the **Genome** of **p**.

Some of the **Individuals** might represent acceptable solutions, close to a local optimum,
or even close to the global one.
Some of them represent mediocre solutions.
Some might represent just rubbish (most will).

Idea of Evolutionary Algorithms (EA):

The entire **Population** carries more information about the objective function than, any of those single **Individuals** alone.

Idea of Evolutionary Algorithms (EA):

The entire **Population** carries more information about the objective function than, any of those single **Individuals** alone.

Keep the good ones.
Discard the bad ones.

Idea of Evolutionary Algorithms (EA):

The entire **Population** carries more information about the objective function than, any of those single **Individuals** alone.

Keep the good ones.
Discard the bad ones.

This is the implementation of the biological principle:

Selection

„Survival of the fittest"

Idea of Evolutionary Algorithms (EA):

The **Selection** (whom to keep, whom to discard) is performed with respect to the result achieved by the objective function **f(s)**.
This is called „**External Selection**“, because the objective function **f(s)** is provided by the environment from external the EA.
(sometimes called „Environmental Selection“)

The result of this **External Selection** is a reduced population, with only the best individuals surviving.

Idea of Evolutionary Algorithms (EA):

The individuals, that survived the external selection process, are now entitled to restock the population to it's original size. They are called **parents**, allowed to generate **offspring.**

Re-stocking the population could in principle be done by generating complete new individuals.
But it is assumed (and proven) that using some of the already acquired information for the new individuals is preferable.

Idea of Evolutionary Algorithms (EA):

The individuals, that survived the external selection process, are now entitled to restock the population to it's original size. They are called **parents**, allowed to generate **offspring.**

Re-stocking the population could in principle be done by generating complete new individuals.
But it is assumed (and proven) that using some of the already acquired information for the new individuals is preferable.

This is an implementation of the biological principles:

Reproduction and *Inheritance*

Idea of Evolutionary Algorithms (EA):

To transport some of the information from the previous generation, into the next generation (**inheritance**), the **offspring** are generated out of the **genome** of the **parents**.

The most popular way in Evolutionary Algorithms is to select two parents from the remaining population (**parent selection**), and to generate offspring by mixing, or combining the genome of these parents.

Idea of Evolutionary Algorithms (EA):

To transport some of the information from the previous generation, into the next generation (**inheritance**), the **offspring** are generated out of the **genome** of the **parents**.

The most popular way in Evolutionary Algorithms is to select two parents from the remaining population (**parent selection**), and to generate offspring by mixing, or combining the genome of these parents.
This is called:

Mating and *Recombination*

Idea of Evolutionary Algorithms (EA):

Inheritance with parent selection, and reproduction of offspring via mating and recombination is used to benefit from the information already acquired (**exploitation** of knowledge).

Idea of Evolutionary Algorithms (EA):

Inheritance with parent selection, and reproduction of offspring via mating and recombination is used to benefit from the information already acquired (**exploitation** of knowledge).

On the other hand, it is a valuable idea to explore new positions **s** in search space **S,** on a stochastic or random basis.

Idea of Evolutionary Algorithms (EA):

Inheritance with parent selection, and reproduction of offspring via mating and recombination is used to benefit from the information already acquired (**exploitation** of knowledge).

On the other hand, it is a valuable idea to explore new positions **s** in search space **S,** on a stochastic or random basis.

This way of **exploration** is implemented in EAs using a random alteration of the existing **genomes.**
It is called:

Mutation

Idea of Evolutionary Algorithms (EA):

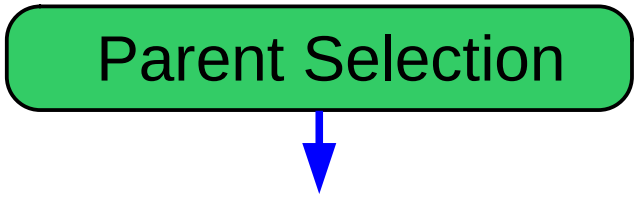
Several aspects and terms of Evolutionary Algorithms are directly borrowed from biology:

- Population of Individuals
- Survival of the fittest
- Selection
- Inheritance
- Reproduction
- Mating
- Recombination
- Cross-Over
- Mutation
- Genome

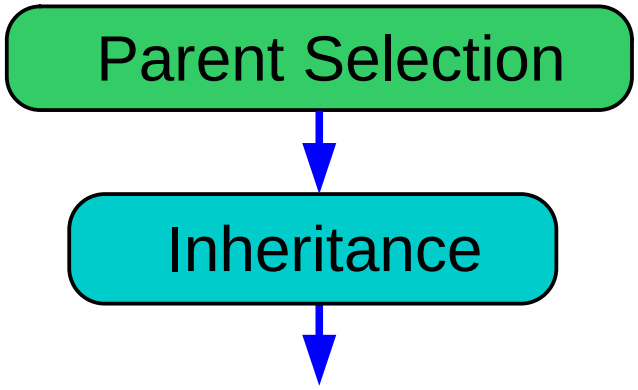
Overview:

- Evolutionary Computation
- Historic Remarks
- Different Approaches
- Idea of Evolutionary Algorithms (EA)
- **EA Steps**
  - Individual, Genome, Fitness, Population
  - Parent selection
  - Inheritance
  - Mutation
  - Fitness evaluation
  - External selection

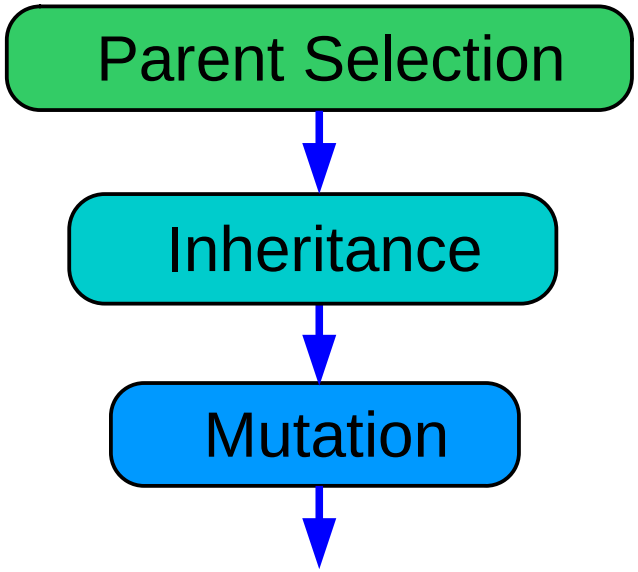
EA steps:



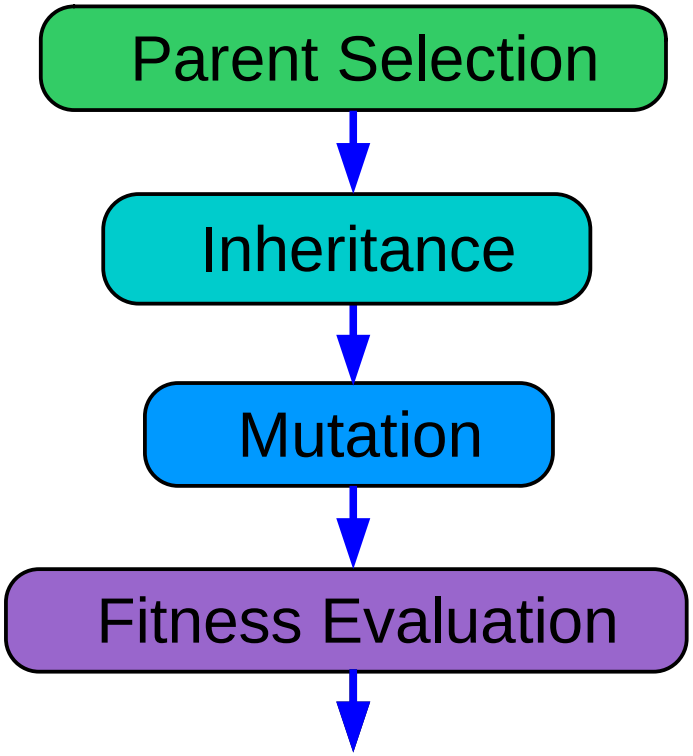
EA steps:



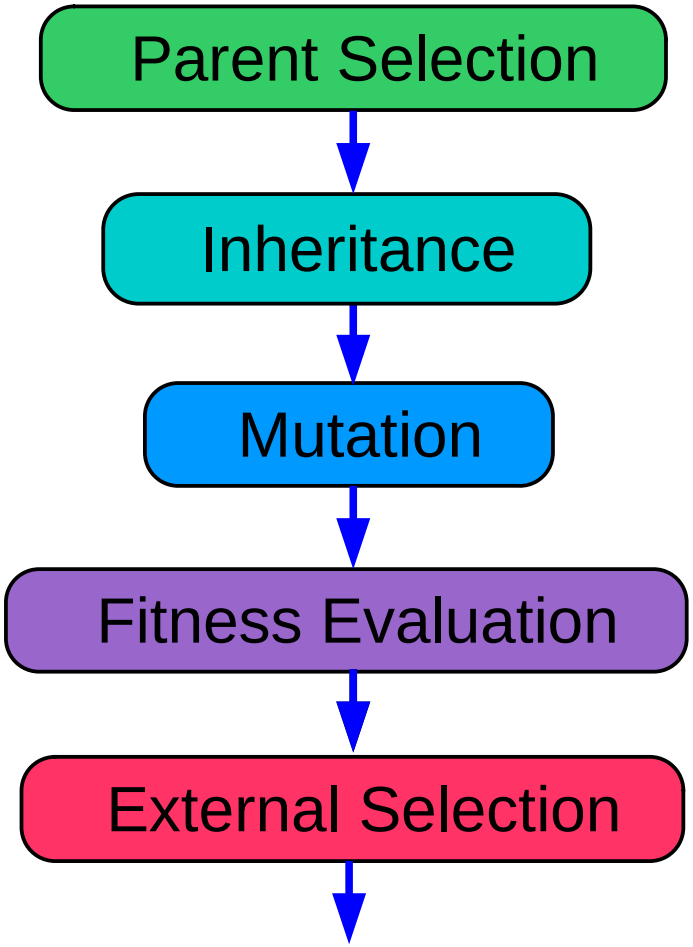
EA steps:



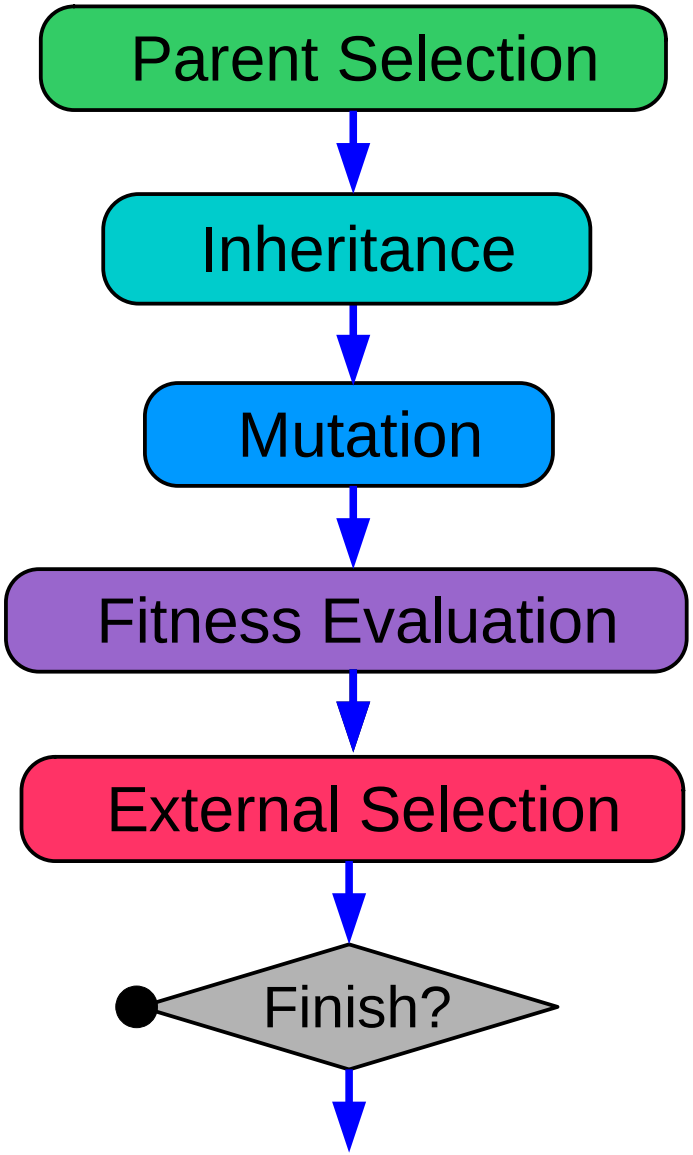
EA steps:



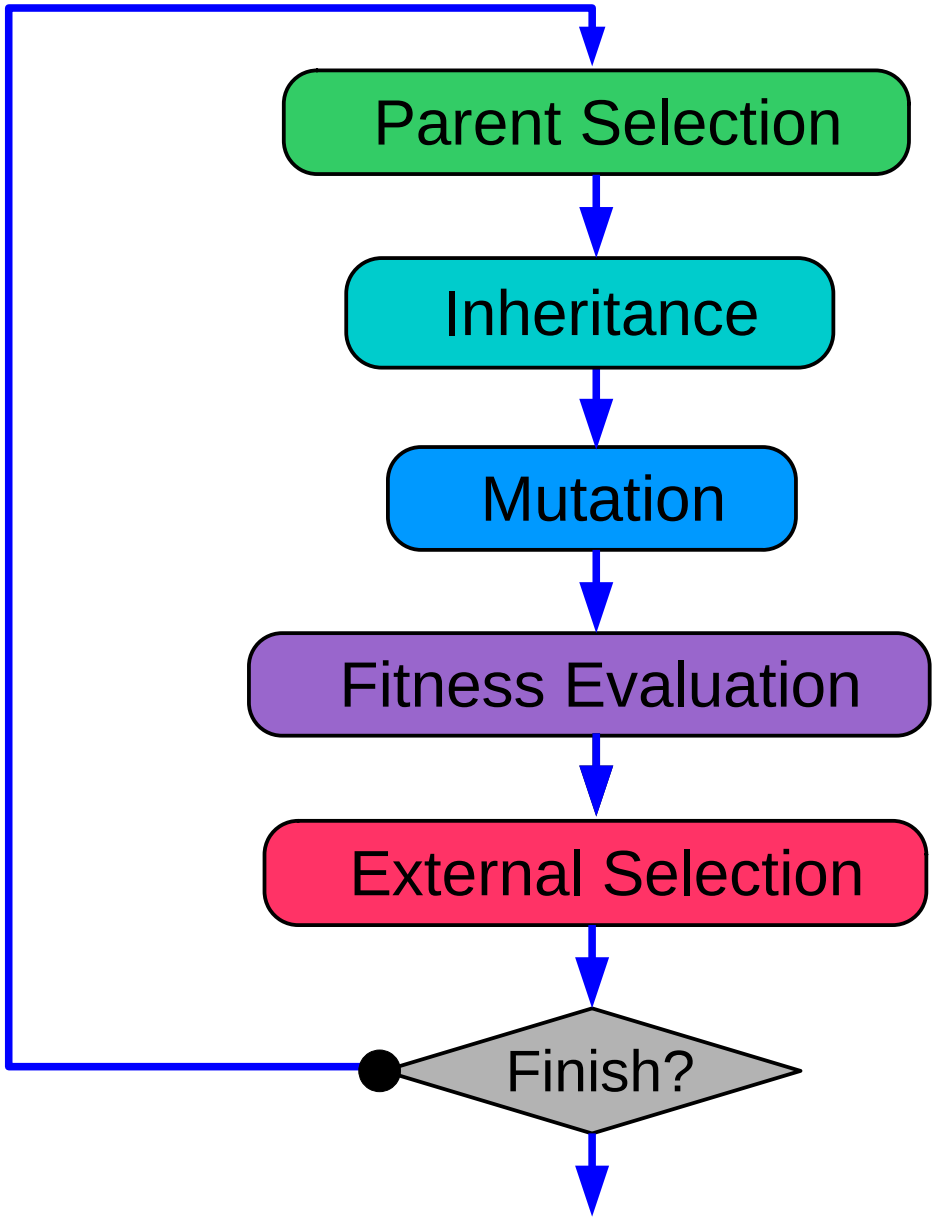
EA steps:



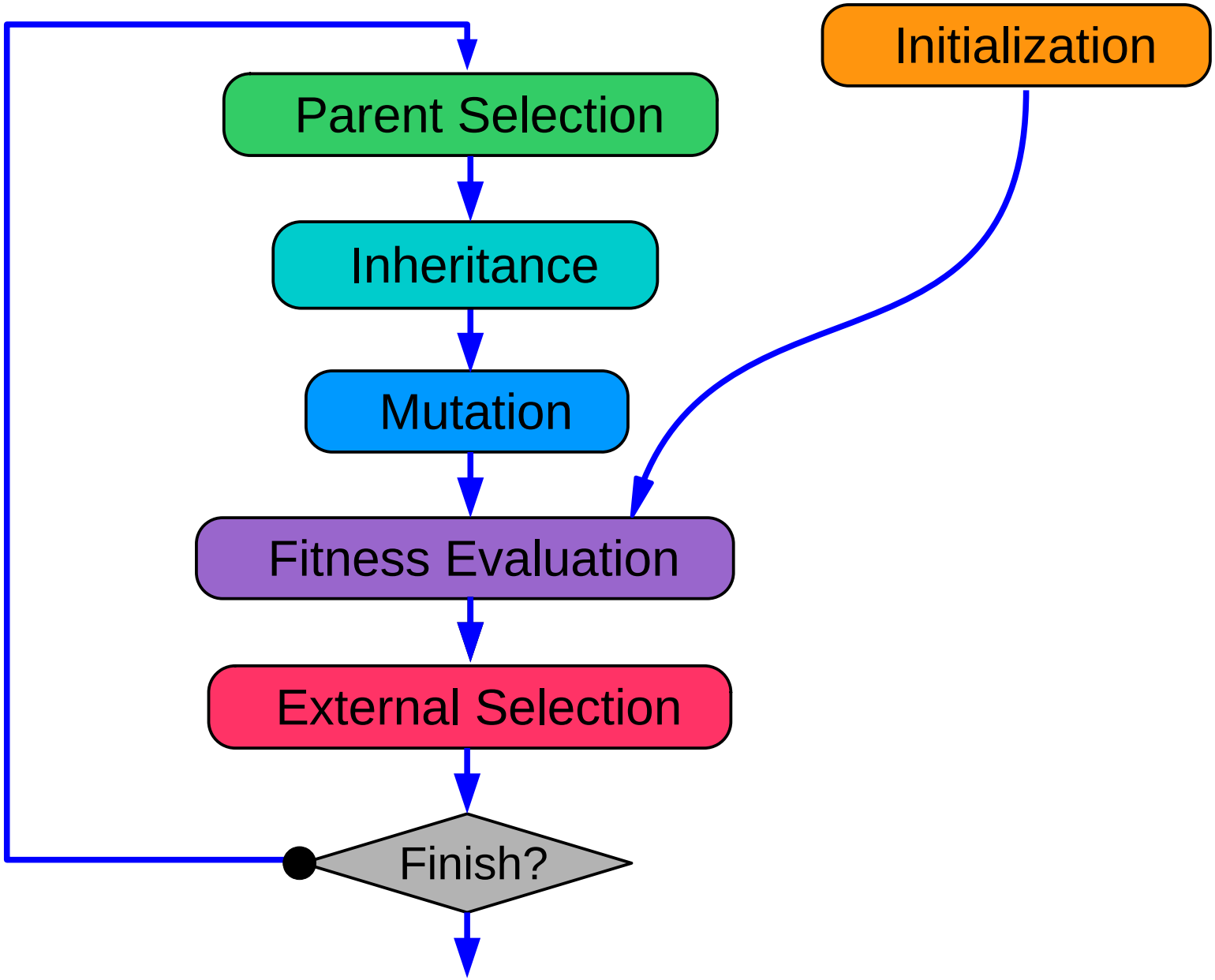
EA steps:



EA steps:



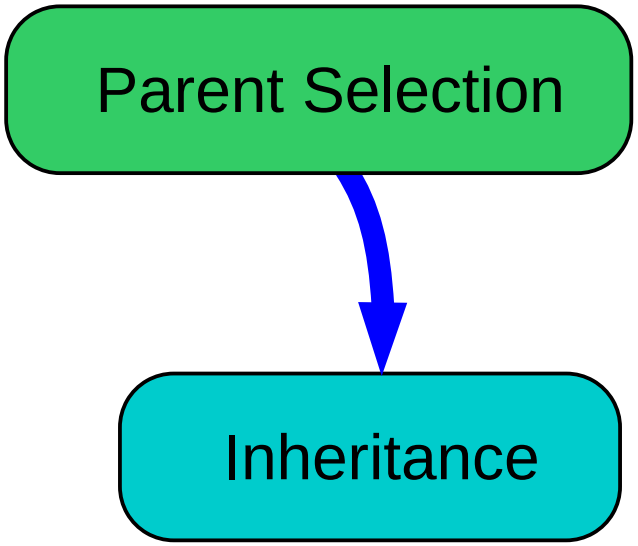
EA steps:



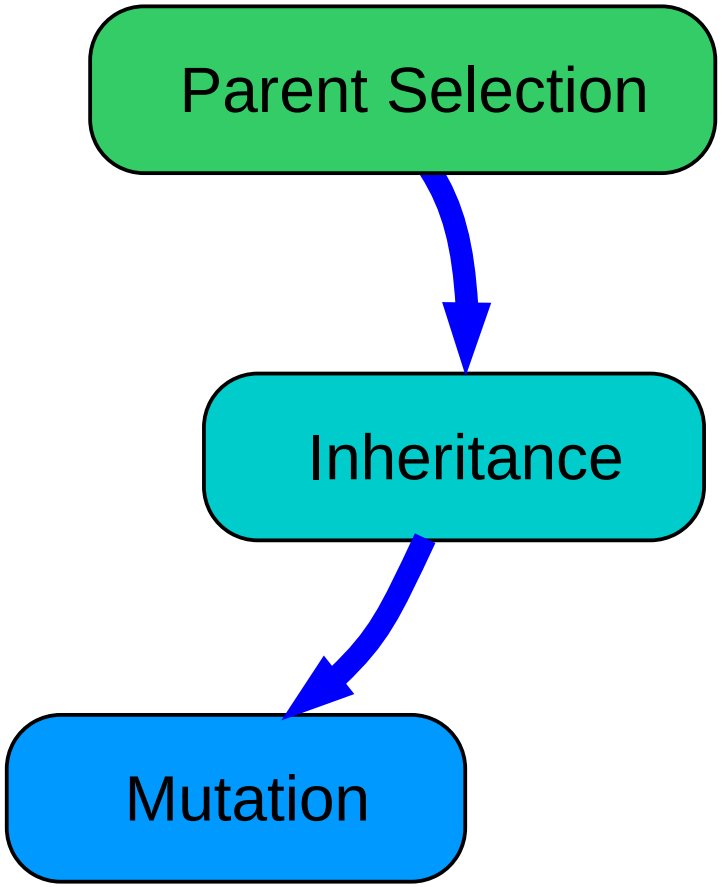
EA cycle of operations:

Parent Selection

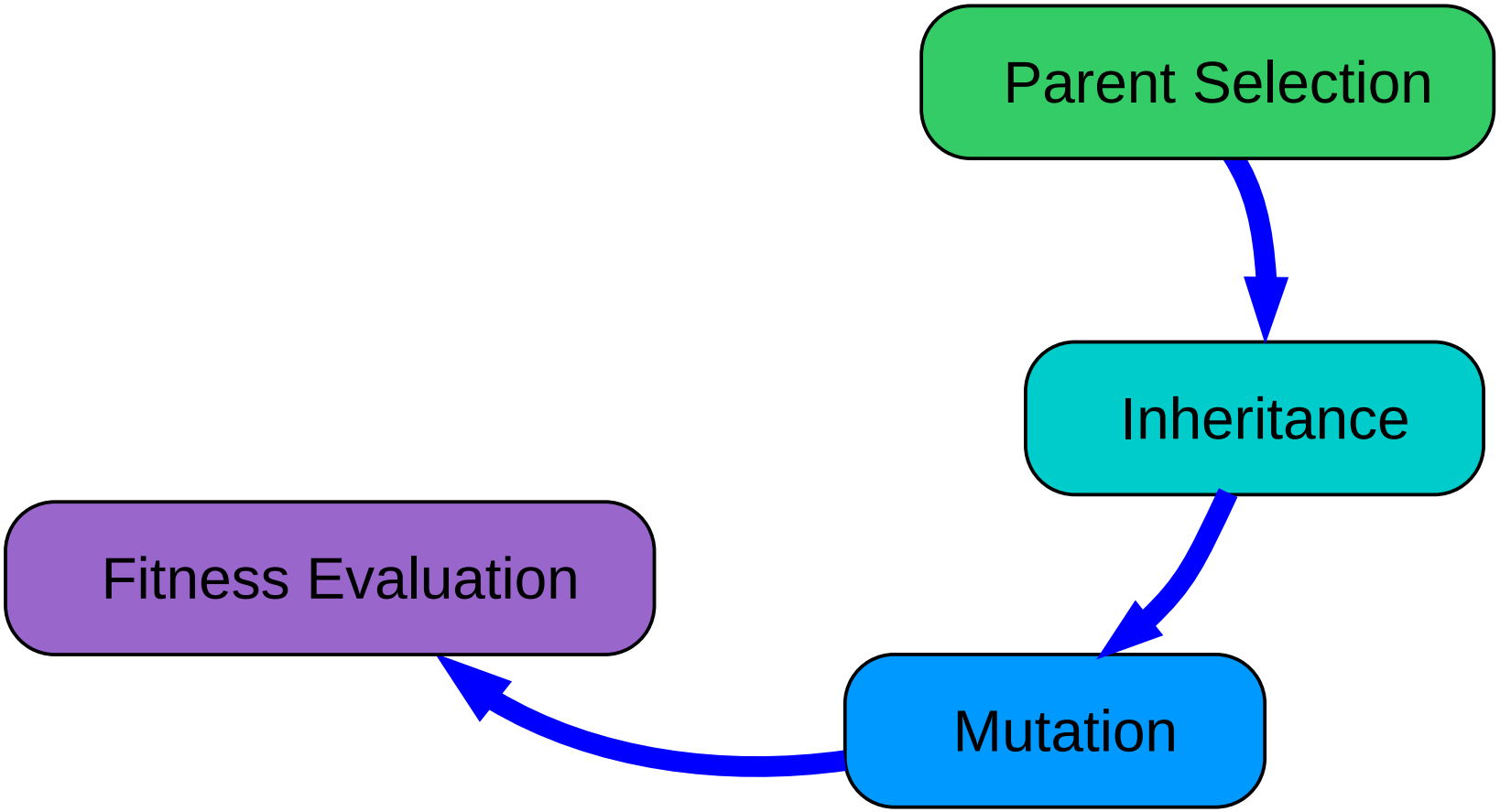
EA cycle of operations:



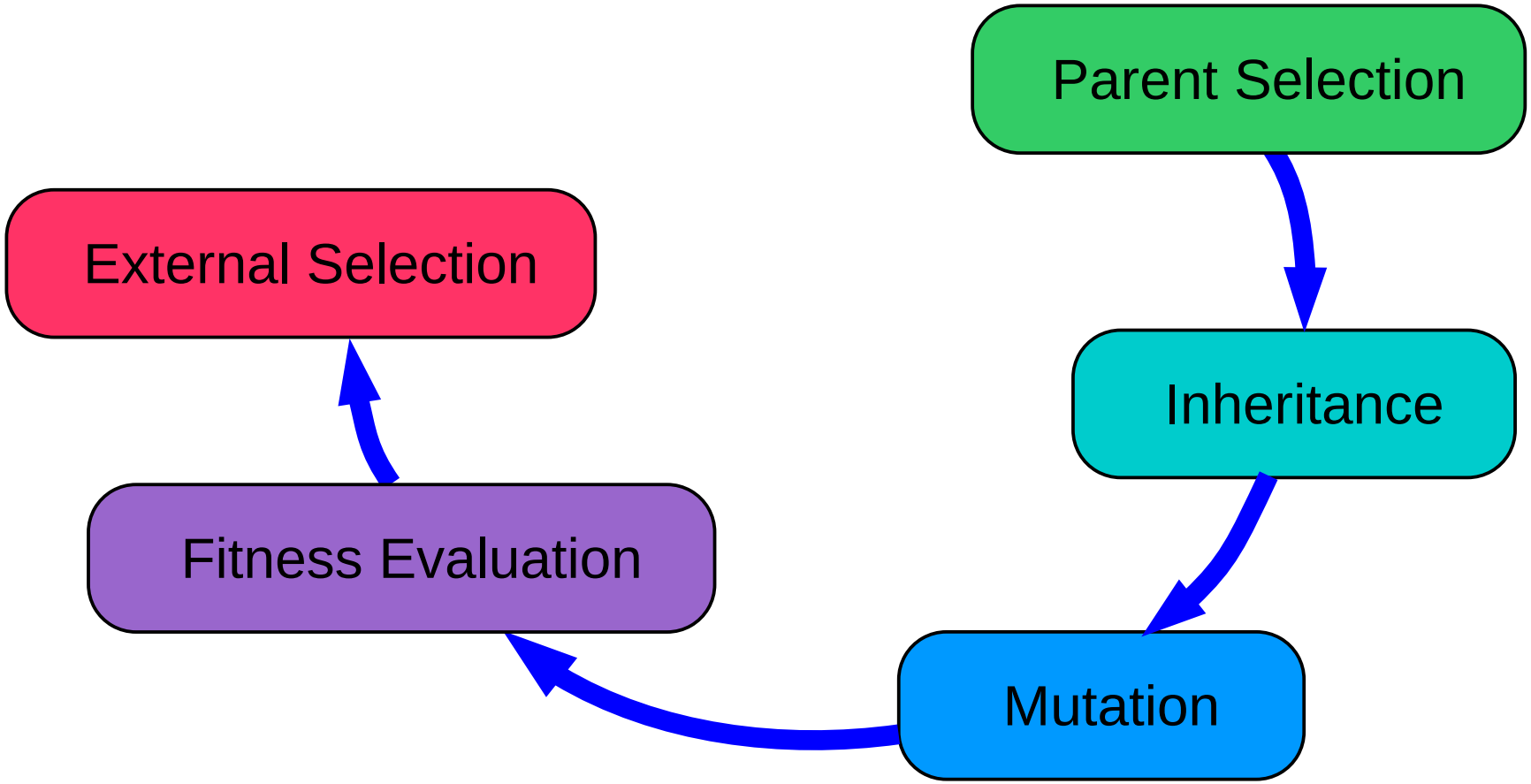
EA cycle of operations:



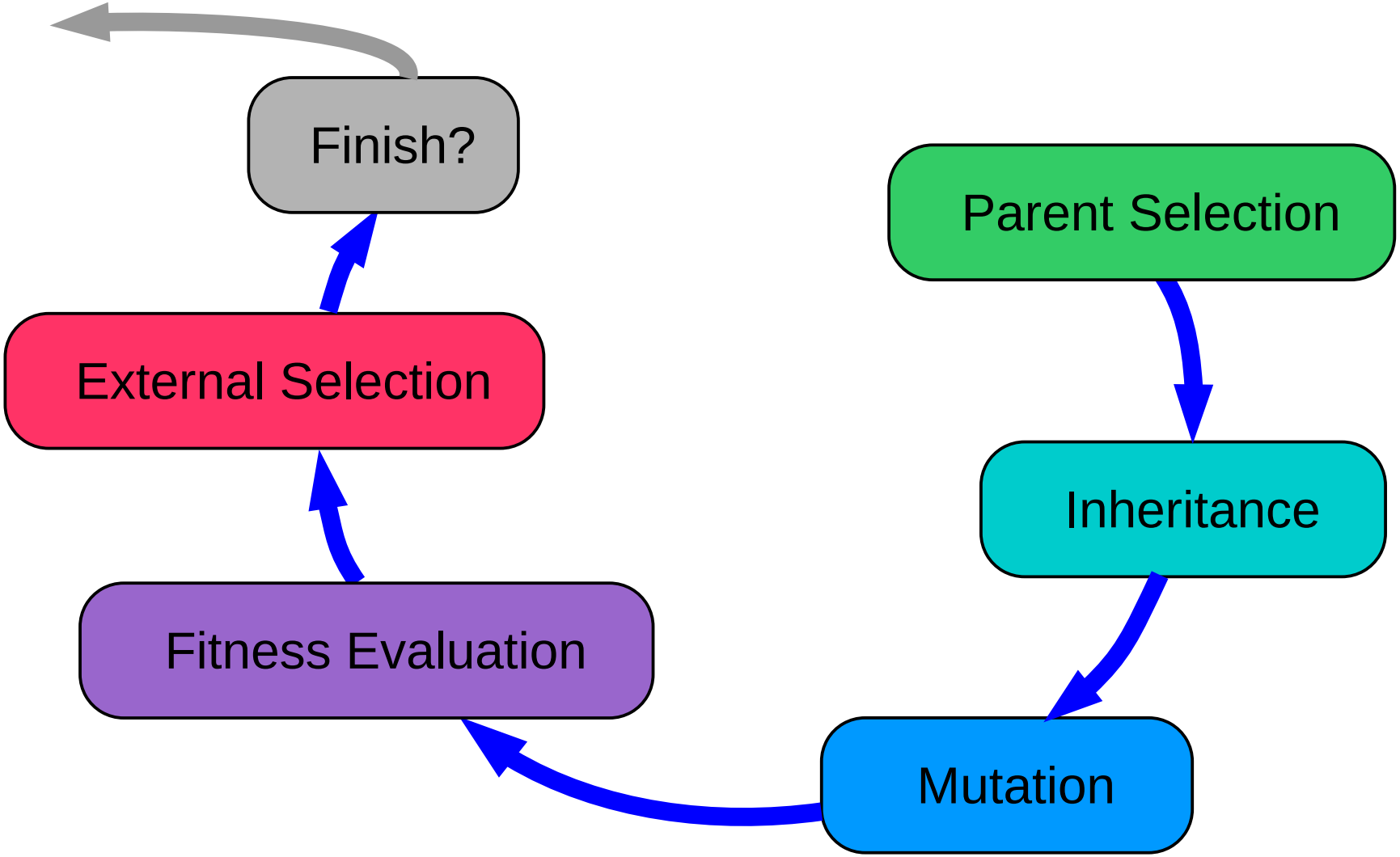
EA cycle of operations:



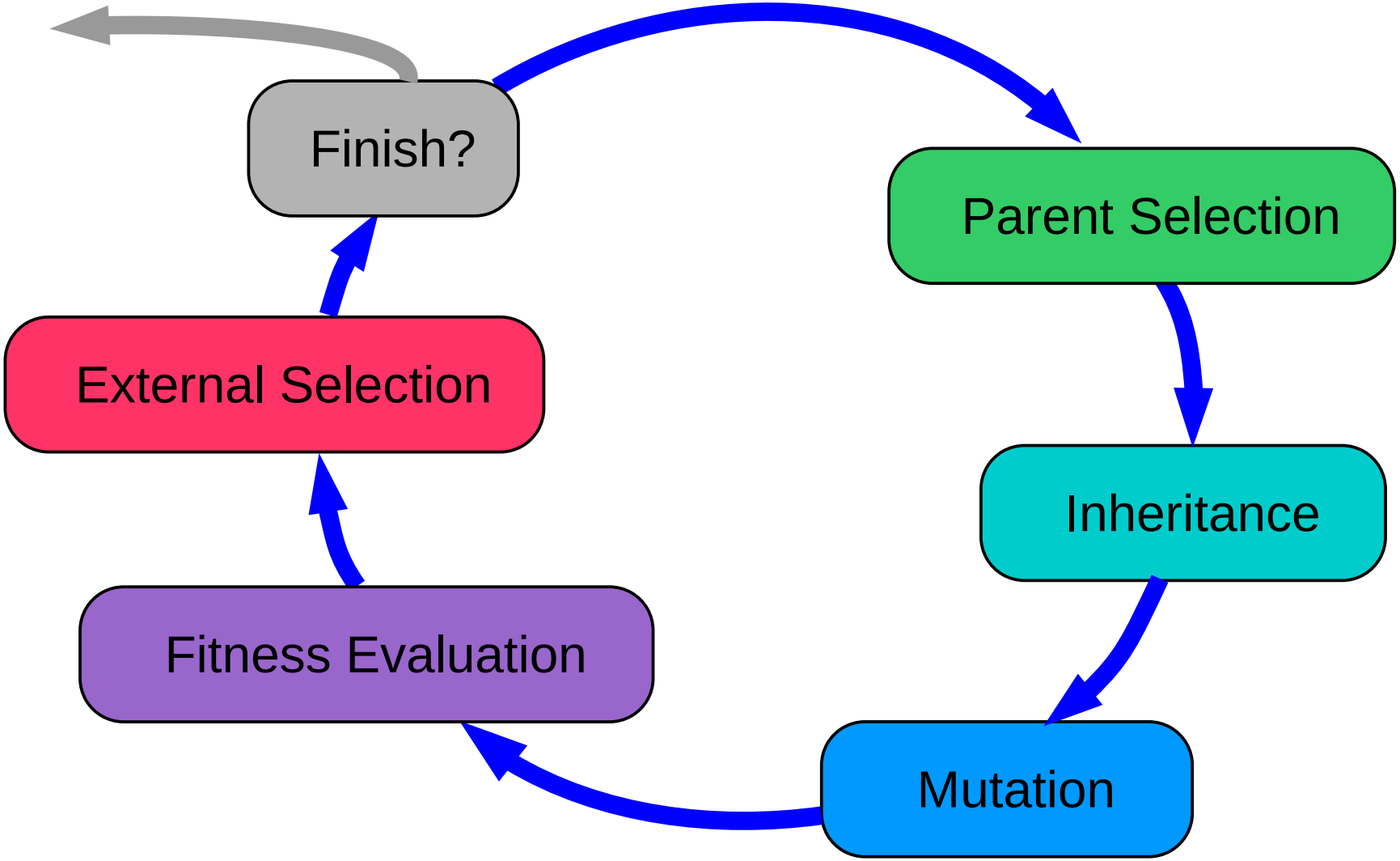
EA cycle of operations:



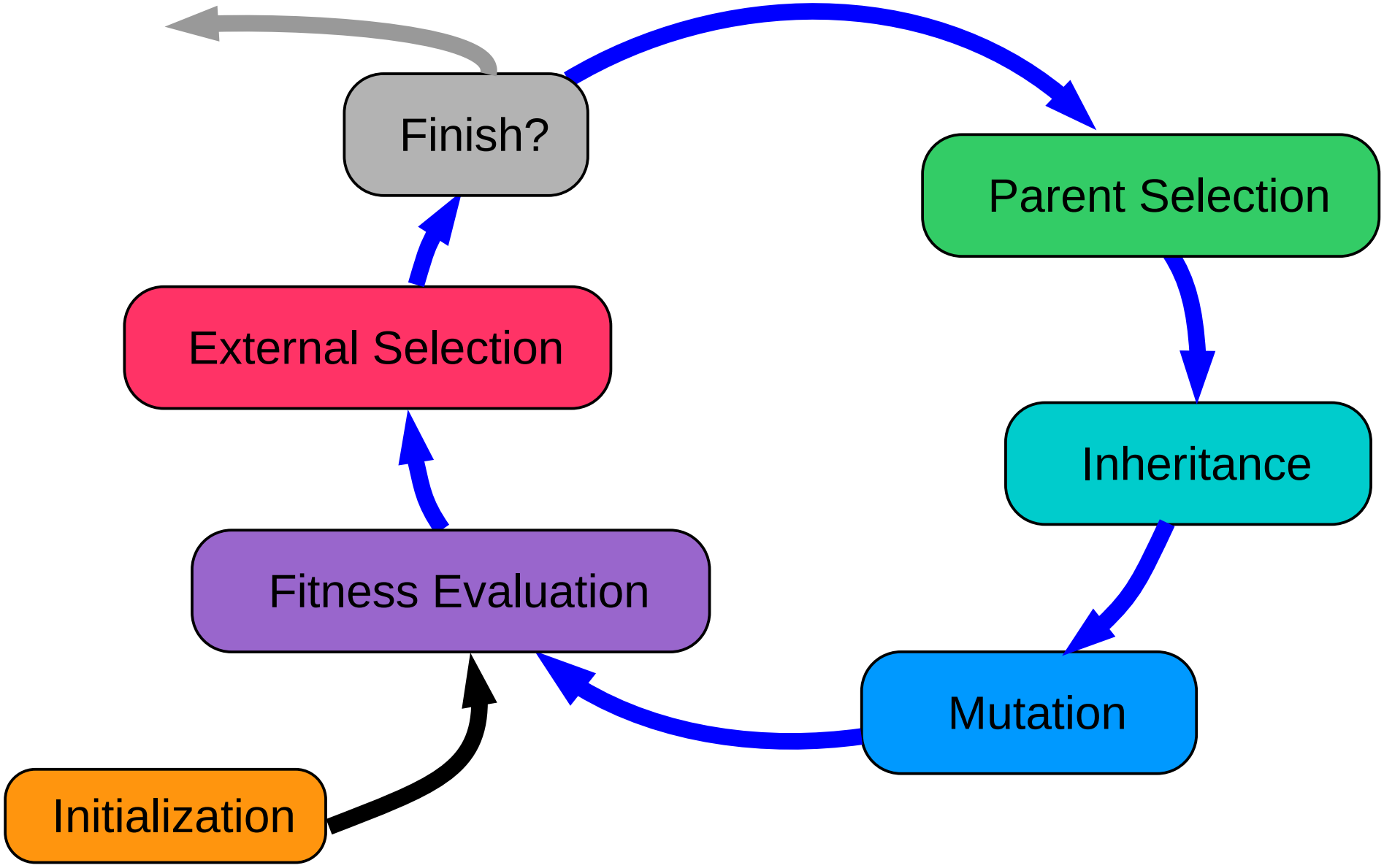
EA cycle of operations:



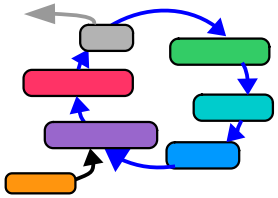
EA cycle of operations:

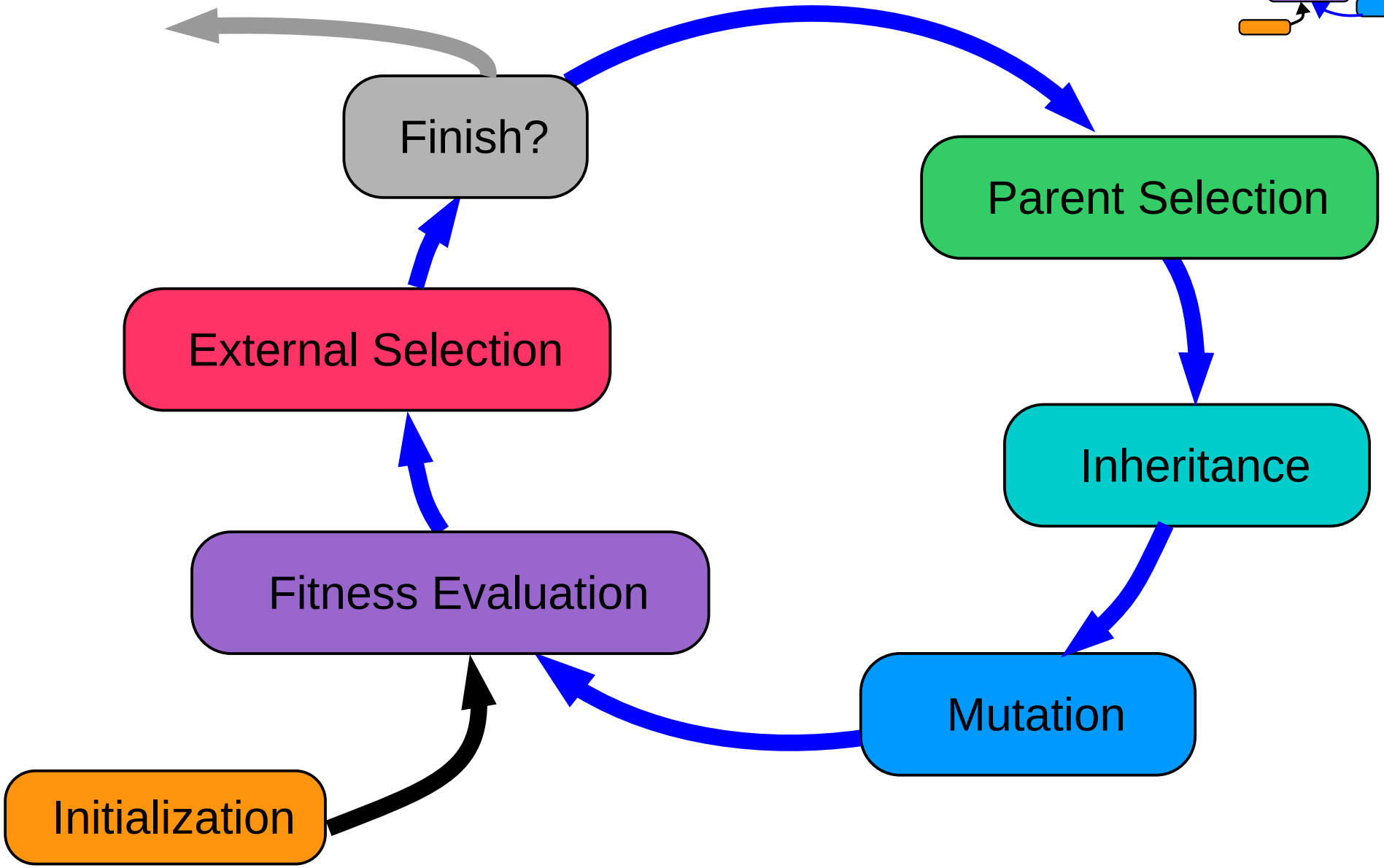


EA cycle of operations:



EA cycle of operations:

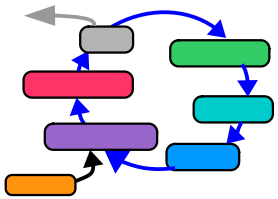




Overview:

- Evolutionary Computation
- Historic Remarks
- Different Approaches
- Idea of Evolutionary Algorithms (EA)
- EA Steps
  - **Individual, Genome, Fitness, Population**
  - Parent selection
  - Inheritance
  - Mutation
  - Fitness evaluation
  - External selection

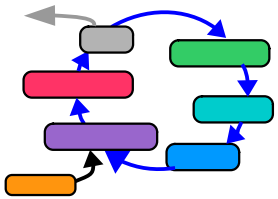
EA: Population, Individual, Genome, Fitness



An Evolutionary Algorithm works with a **population** of single **individuals**.

EA:

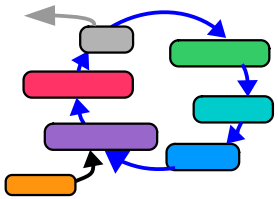
Population, Individual, Genome, Fitness



An Evolutionary Algorithm works with a **population** of single **individuals.**

Each individual owns a **genome.**

EA: Population, Individual, Genome, Fitness

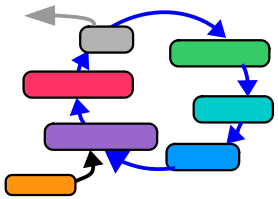


An Evolutionary Algorithm works with a **population** of single **individuals.**

Each individual owns a **genome.**

A **genome** is a set of parameters **s** out of a search space **S**.

EA: Population, Individual, Genome, Fitness



An Evolutionary Algorithm works with a **population** of single **individuals.**

Each individual owns a **genome.**

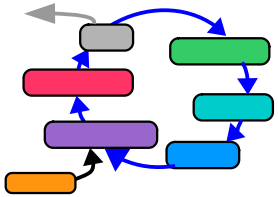
A **genome** is a set of parameters **s** out of a search space **S**.

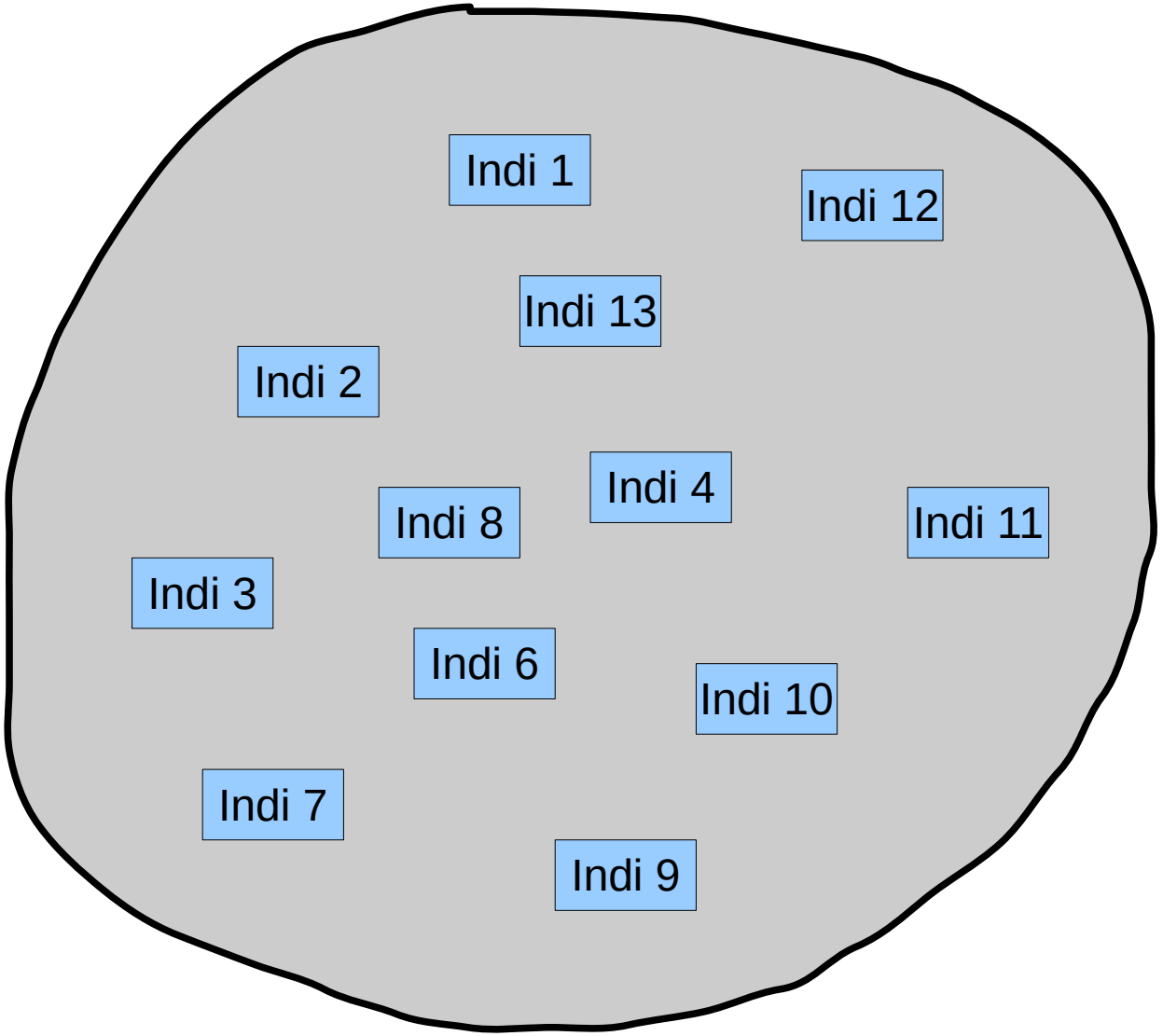
The genome $\mathbf{s_p}$ of an individual **p** is evaluated with respect to a fitness function **f** (objective function).
The result $\mathbf{f(s_p)}$ of this evaluation is called $\mathbf{f_p}$

fitness of the individual **p**.

EA:

Population of Individuals



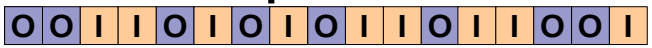


A **population** is a set or group of individuals.

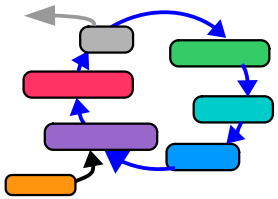
Indi 1 Indi 2 ...

Each **individual** has an individual genome.

A **genome** is a set of parameters **s** out of a search space **S**.



EA: Population of Individuals

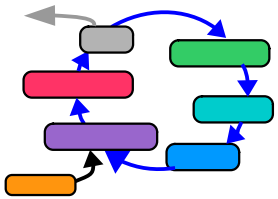


A **population** of individuals with individual **genomes**.

Indi 1	O	O	I	I	O	I	O	I	O	I	I	O	I	I	O	O	I
Indi 2	O	I	I	O	O	O	I	I	I	I	O	O	I	I	I	O	O
Indi 3	I	O	I	I	O	I	I	I	O	I	I	I	I	I	O	O	I
Indi 4	O	I	I	O	I	O	I	I	O	O	I	O	I	O	I	O	I
Indi 5	O	O	O	I	I	I	O	O	O	I	I	I	O	O	O	I	I
Indi 6	O	I	O	I	I	O	O	I	I	O	O	I	O	O	O	I	I
Indi 7	O	I	O	I	O	I	O	I	O	I	O	I	O	I	O	I	O
Indi 8	O	O	O	I	O	O	O	O	O	I	I	O	O	I	O	O	I

EA:

Genome

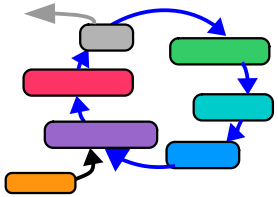


A **genome** is a set of parameters **s** out of a search space **S**. The genome determines the properties and capabilities of an individual (phenotype).

Indi 1	O	O	I	I	O	I	O	I	O	I	I	O	I	I	O	O	I

EA:

Genome

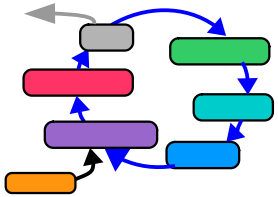


A **genome** is a set of parameters **s** out of a search space **S**. The genome determines the properties and capabilities of an individual (phenotype).

The implementation of a **genome** can be a simple string of bits, can be a vector of integer or real values, or can be a set of items, (or something else).
A genome can even be a string of characters, like a computer program, or a complete text.

EA:

Genome



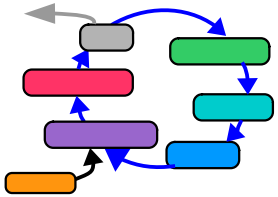
A **genome** is a set of parameters **s** out of a search space **S**. The genome determines the properties and capabilities of an individual (phenotype).

The implementation of a **genome** can be a simple string of bits, can be a vector of integer or real values, or can be a set of items, (or something else).
A genome can even be a string of characters, like a computer program, or a complete text.

A genome does not even has to have a fixed length.
The length can change during the optimization of the EA.

EA:

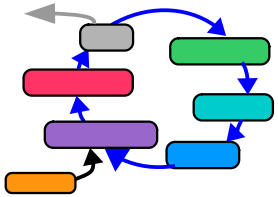
Genome



A **genome** can be

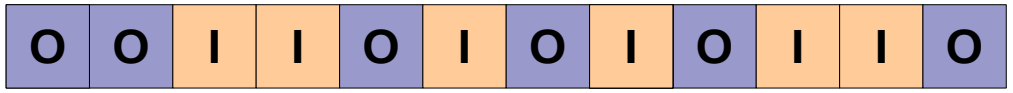
EA:

Genome



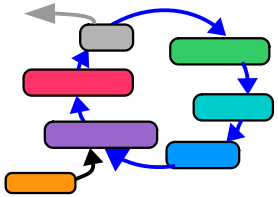
A **genome** can be

a simple string of bits:



EA:

Genome



A **genome** can be

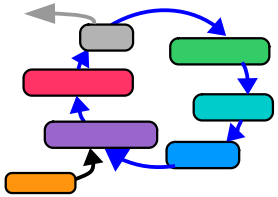
a simple string of bits:

O	O	I	I	O	I	O	I	O	I	I	O

a vector of values:

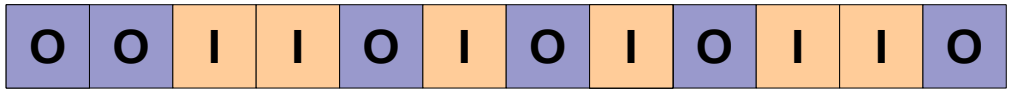
17	0.38	2.0e-7	42	3.141

EA: Genome

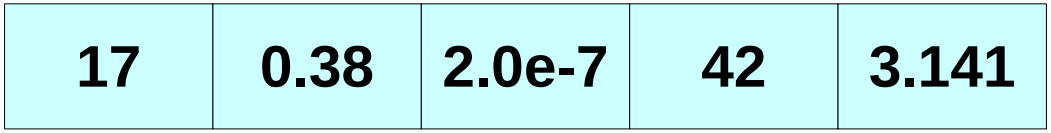


A **genome** can be

a simple string of bits:

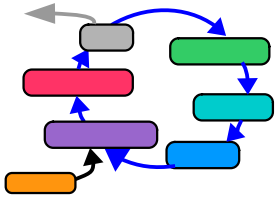


a vector of values:



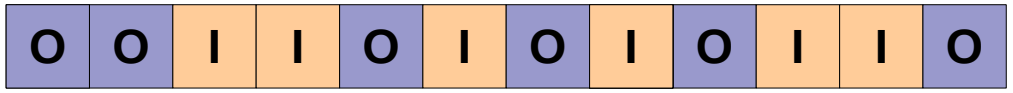
a set of items: { Car, Banana, Monday, USB-Stick }

EA: Genome

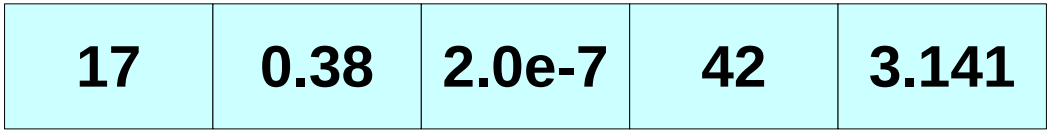


A **genome** can be

a simple string of bits:



a vector of values:

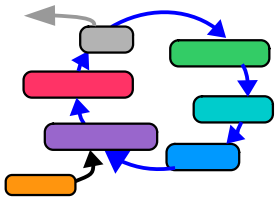


a set of items: { Car, Banana, Monday, USB-Stick }

a string of characters: „**Throatwobbler Mangrove**“

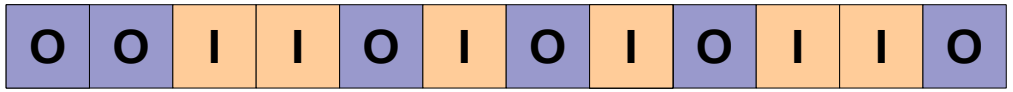
From: Monty Python's Flying Circus: Episode 22, Cosmetic surgery

EA: Genome

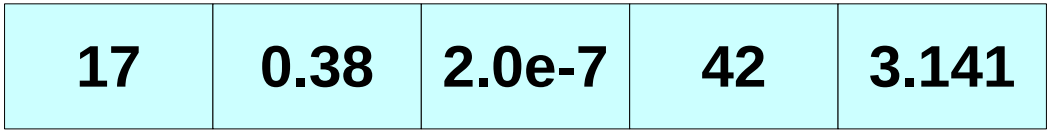


A **genome** can be

a simple string of bits:



a vector of values:



a set of items: { Car, Banana, Monday, USB-Stick }

a string of characters: „**Throatwobbler Mangrove**“

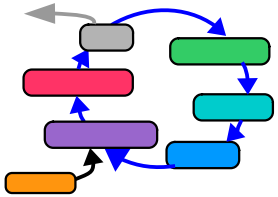
From: Monty Python's Flying Circus: Episode 22, Cosmetic surgery

a complete computer program.

```
#include “Evolutionary_Algorithms.h”
int main()
{
   int P = 1000;      // size of population
   int L= 20;         // length of genome
```

EA:

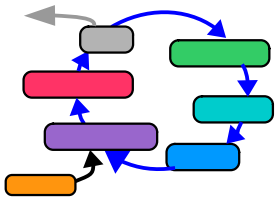
Fitnessfunction



The **fitness function f** calculates the fitness value **f(s)** for each individual **genome s**.
The fitness value is a scalar value.
Larger fitness is indicating a better individual.
The fitness is heavily depending on the application!

The mapping from **genome** to **fitness value**
can be directly, e.g. by a mathematical function,
or can be a complicated process, e.g. the performance of a soccer playing, walking robot.

EA: Fitness value



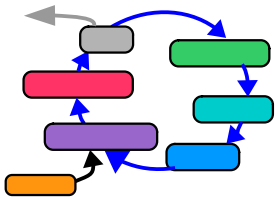
The **fitness function f** calculates the fitness value f(s) for each individual **genome**.

Indi 1	O	O	I	I	O	I	O	I	O	I	I	O	I	I	O	O	I

f(1) = 9

EA:

Fitness value

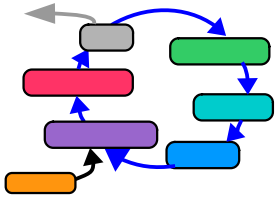


The **fitness function f** calculates the fitness value f(s) for each individual **genome**.

Indi 1	O O I I O I O I O I I O I I O O I	f(1) = 9
Indi 2	O I I O O O I I I I O O I I I O O	f(2) = 9
Indi 3	I O I I O I I I O I I I I I O O I	f(3) = 12
Indi 4	O I I O I O I I O O I O I O I O I	f(4) = 9
Indi 5	O O O I I I O O O I I I O O O I I	f(5) = 8
Indi 6	O I O I I O O I I O O I O O O I I	f(6) = 8
Indi 7	O I O I O I O I O I O I O I O I O	f(7) = 8
Indi 8	O O O I O O O O O I I O O I O O I	f(8) = 5

EA:

Fitness value



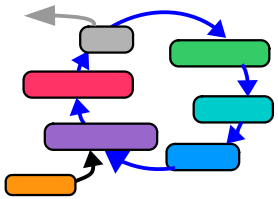
The **fitness function f** calculates the fitness value f(s) for each individual **genome**.

Indi 1	O	O	I	I	O	I	O	I	O	I	I	O	I	I	O	O	I

$f'(1) = 2$

EA:

Fitness value

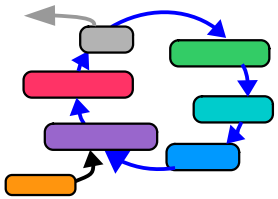


The **fitness function f** calculates the fitness value f(s) for each individual **genome**.

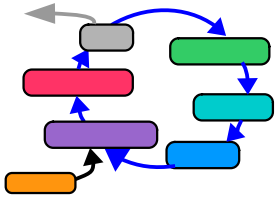
Indi 1	O O I I O I O I O I I O I I O O I	f'(1) = 2
Indi 2	O I I O O O I I I I O O I I I O O	f'(2) = 4
Indi 3	I O I I O I I I O I I I I I O O I	f'(3) = 5
Indi 4	O I I O I O I I O O I O I O I O I	f'(4) = 2
Indi 5	O O O I I I O O O I I I O O O I I	f'(5) = 3
Indi 6	O I O I I O O I I O O I O O O I I	f'(6) = 2
Indi 7	O I O I O I O I O I O I O I O I O	f'(7) = 1
Indi 8	O O O I O O O O O I I O O I O O I	f'(8) = 2

EA:

Population

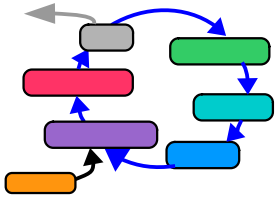


EA: Population



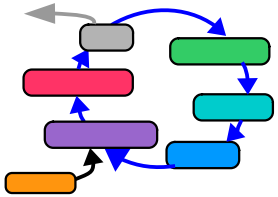
The EAs, as multi-hypothesis approach are working with a complete **population** of individuals.

EA: Population



The EAs, as multi-hypothesis approach are working with a complete **population** of individuals.
Typically, the number of hypothesis is equal to the number **P** of individuals; **P** is called the size of the population.

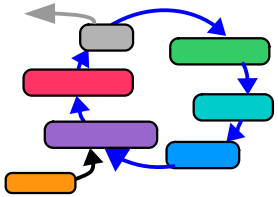
EA: Population



The EAs, as multi-hypothesis approach are working with a complete **population** of individuals.
Typically, the number of hypothesis is equal to the number **P** of individuals; **P** is called the size of the population.

In general, it is a good advise, to keep the size of the population fixed; but it is possible to change **P**.

EA: Population



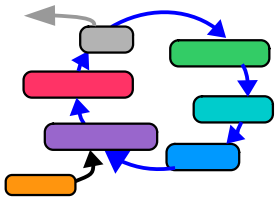
The EAs, as multi-hypothesis approach are working with a complete **population** of individuals.
Typically, the number of hypothesis is equal to the number **P** of individuals; **P** is called the size of the population.

In general, it is a good advise, to keep the size of the population fixed; but it is possible to change **P**.

Special approaches and extensions of EAs divide the population into several **sub-populations**, that may be treated differently.

EA:

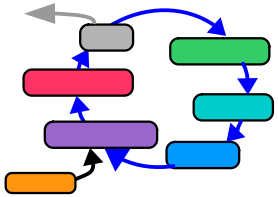
Population



How large shall the size of the **population** be?

EA:

Population

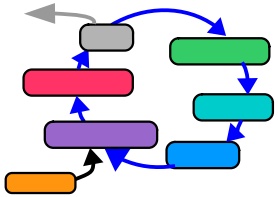


How large shall the size of the **population** be?

There are several aspects that influence the choice of **P:**

- enough richness, diversity of hypothesis
- costs for evaluating the individuals
- computing costs for the entire algorithm

EA: Population



How large shall the size of the **population** be?

There are several aspects that influence the choice of **P:**

- enough richness, diversity of hypothesis
- costs for evaluating the individuals
- computing costs for the entire algorithm

- keep it simple, keep it small

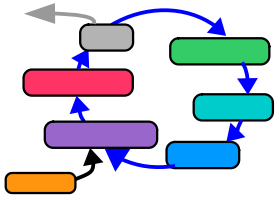
P = 50 … 1000 (simulations)

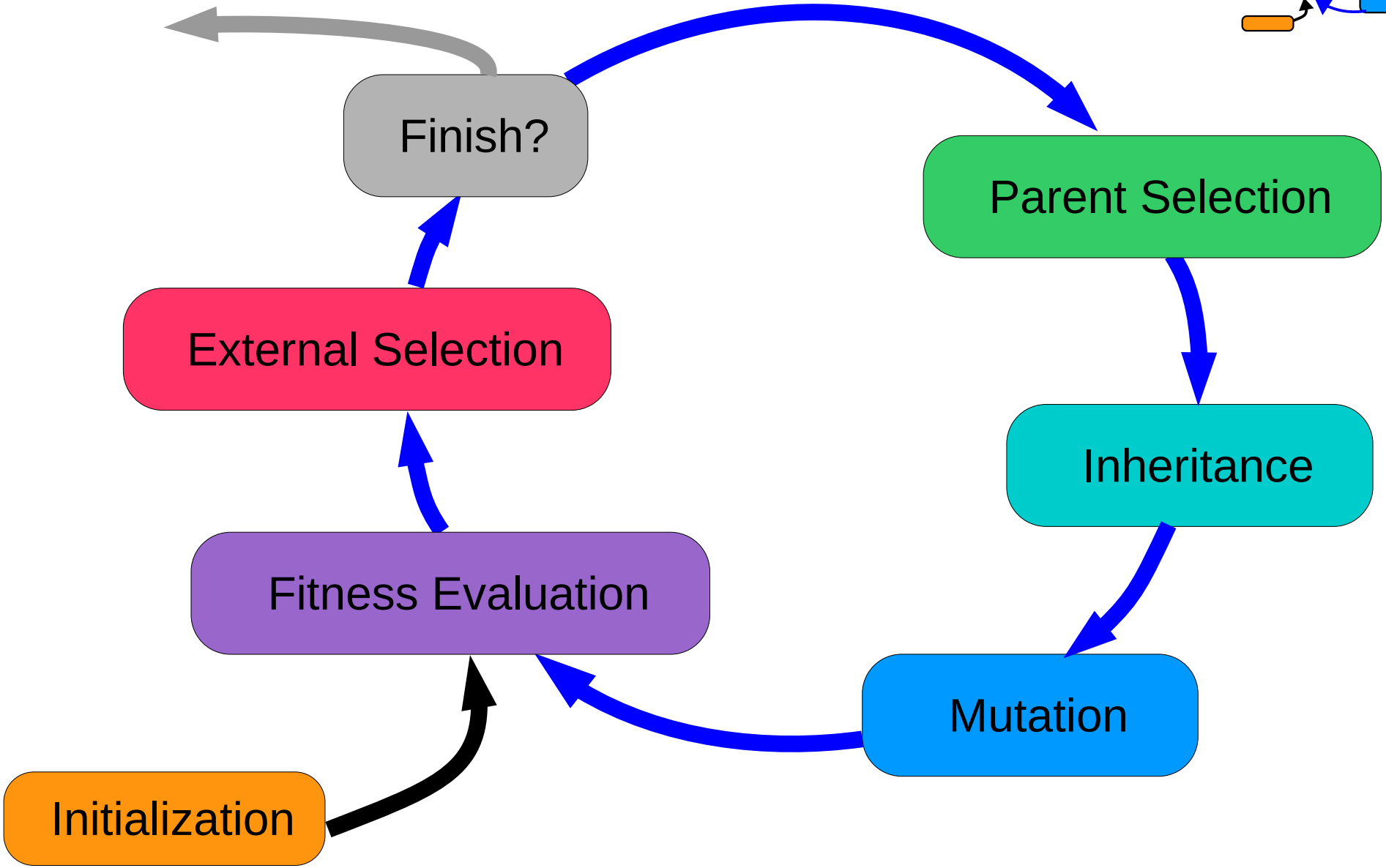
P = 5 ... 50 (real world experiments)

Overview:

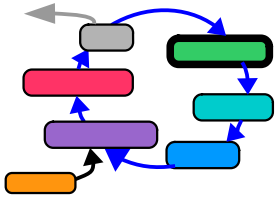
- Evolutionary Computation
- Historic Remarks
- Different Approaches
- Idea of Evolutionary Algorithms (EA)
- EA Steps
  - Individual, Genome, Fitness, Population
  - **Parent selection**
  - Inheritance
  - Mutation
  - Fitness evaluation
  - External selection

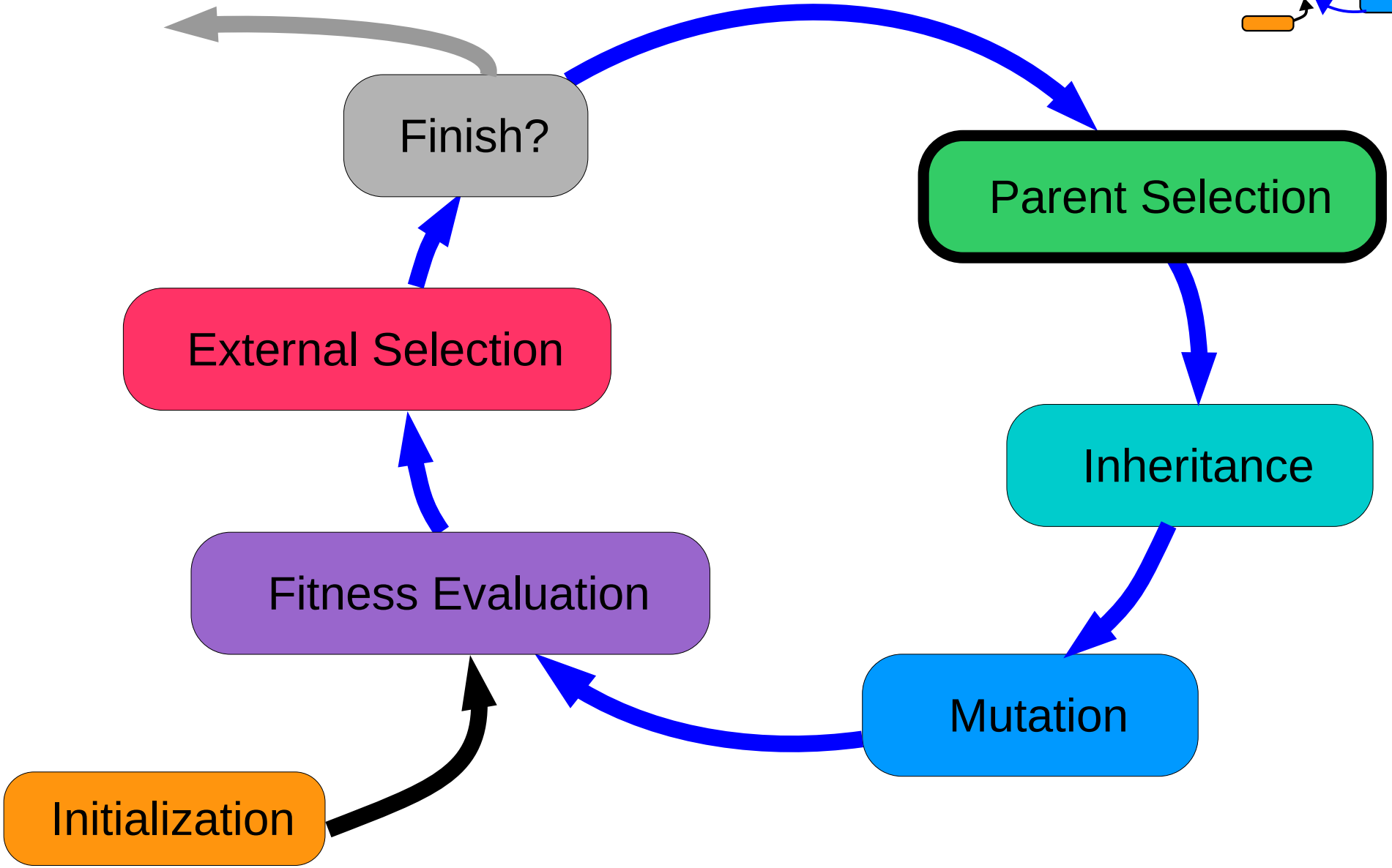
EA cycle of operation:



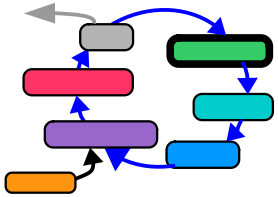


EA cycle of operation:





EA: Parent Selection



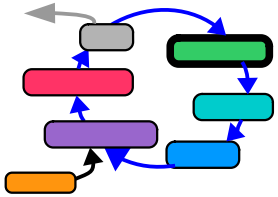
The μ individuals that survived the external selection process, are the pool to select the **parents** from.

Those **parents** selected, will have to generate λ offspring.

For every inheritance process, a number of k individuals (parents) are selected from the pool of possible parents for **mating + recombination**.

EA:

Parent Selection

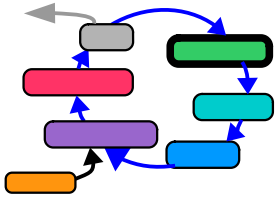


Out of the μ remaining individuals the **k-**parents are chosen for the inheritance step to generate λ offspring.

pool of μ possible parents

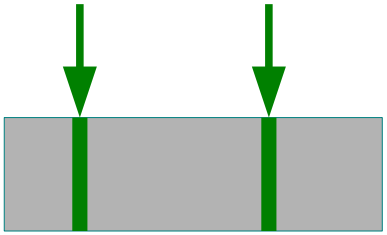
EA:

Parent Selection



Out of the μ remaining individuals the **k-**parents are chosen for the inheritance step to generate λ offspring.

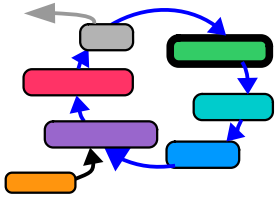
pool of μ possible parents



parent selection

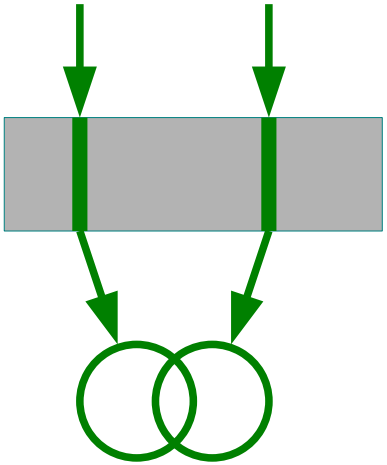
EA:

Parent Selection



Out of the μ remaining individuals the **k-**parents are chosen for the inheritance step to generate λ offspring.

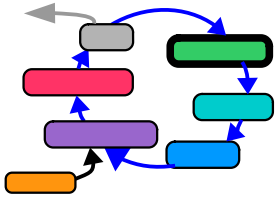
pool of μ possible parents



parent selection

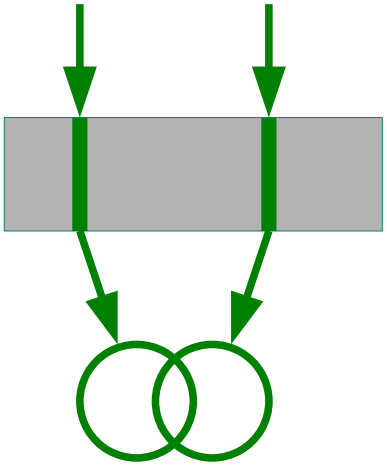
mating, recombination

EA: Parent Selection



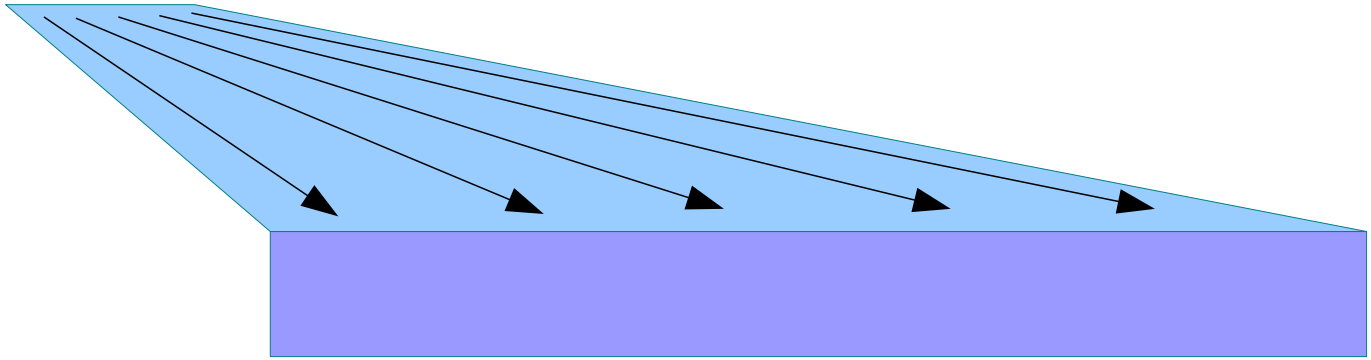
Out of the μ remaining individuals the **k-**parents are chosen for the inheritance step to generate λ offspring.

pool of μ possible parents



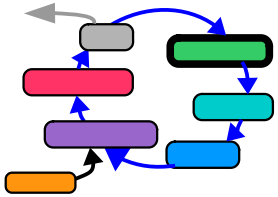
parent selection

mating, recombination



λ offspring inheritance

EA: Parent Selection



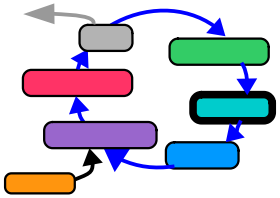
Common strategies for the **parent selection** are:

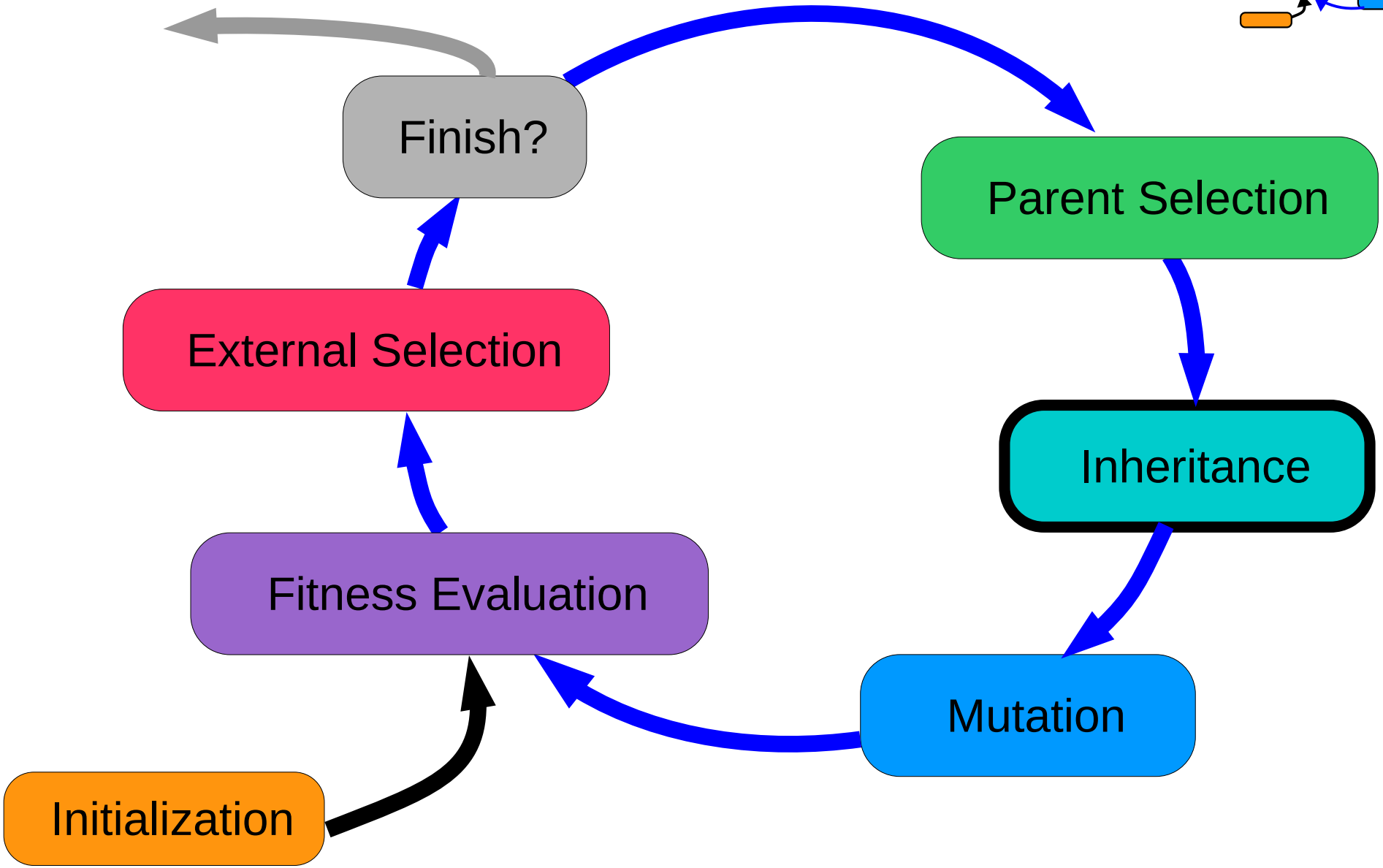
- random choice
- schedule based, e.g. round robin
- fitness based elitism:
  - fitness proportional choice
  - rank proportional choice
- fitness based stochastic:
  - fitness proportionate, probabilistic choice
  - rank proportionate, probabilistic choice
- combinations of the above

Overview:

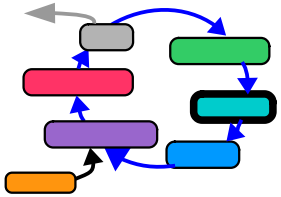
- Evolutionary Computation
- Historic Remarks
- Different Approaches
- Idea of Evolutionary Algorithms (EA)
- EA Steps
  - Individual, Genome, Fitness, Population
  - Parent selection
  - **Inheritance**
  - Mutation
  - Fitness evaluation
  - External selection

EA cycle of operation:





EA: Inheritance



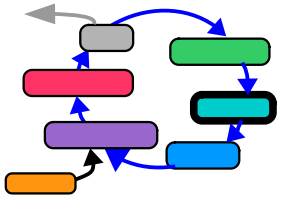
Inheritance is the principle of transporting some of the information from the previous generation (parents), into the next generation, by generating the **offspring** out of the **parents genomes** .

Thus, generating offspring means to create new individuals with new genomes.

Inheritance in the context of Evolutionary Algorithms means to use the genomes of those parents that have been selected.

There is a wide variety of possible ways to generate offspring.

EA: Inheritance



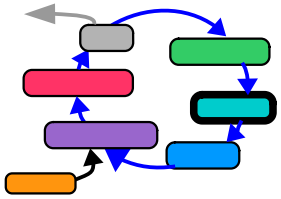
The most popular mechanism to implement the **inheritance** principle is to use k=2 parents and to **combine** their genomes (mostly called recombination).

The most popular mechanism to implement **recombination** of 2 genomes is **cross over:**

Inheritance:

- **recombination**
  - by **cross over**
  - (other recombination operators)
- *(other inheritance mechanism)*
  - (other inheritance operators)
  - ...

EA: Inheritance

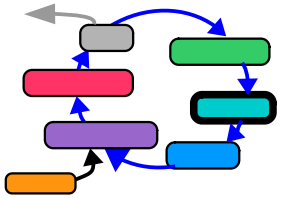


Recombination by **1-point cross over**:

Select two parents, **A** and **B** and **recombine** their genomes
by taking one part form parent **A**
and the other part from parent **B**

Choose a **random point C** where to split the genomes,
and recombine the four remaining fractions to build new individuals, containing partial information from both parents.

EA: Inheritance

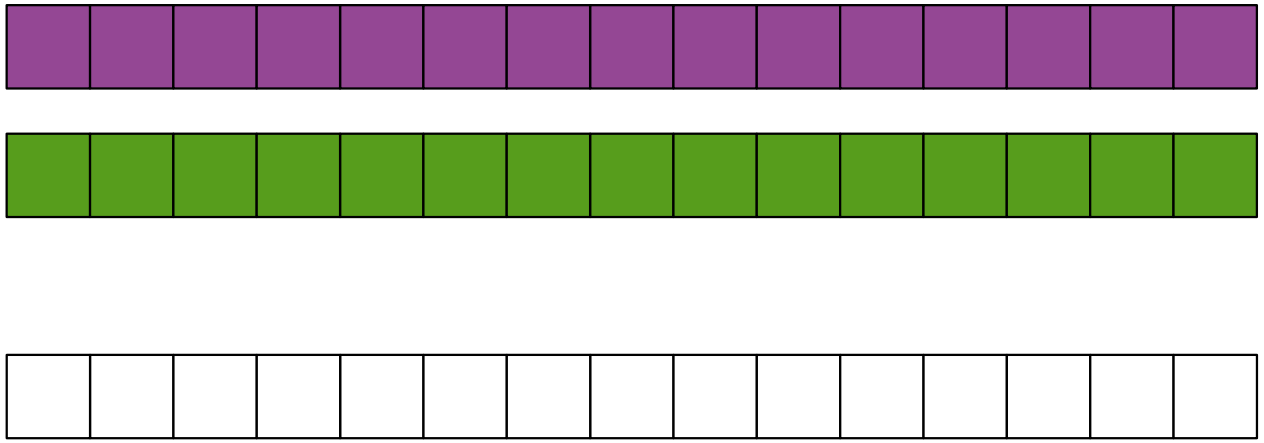


Recombination by **1-point cross over**:

Parent A:

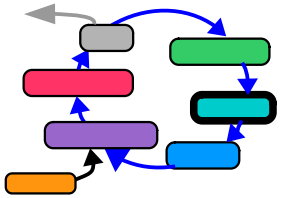
Parent B:

Child 1:

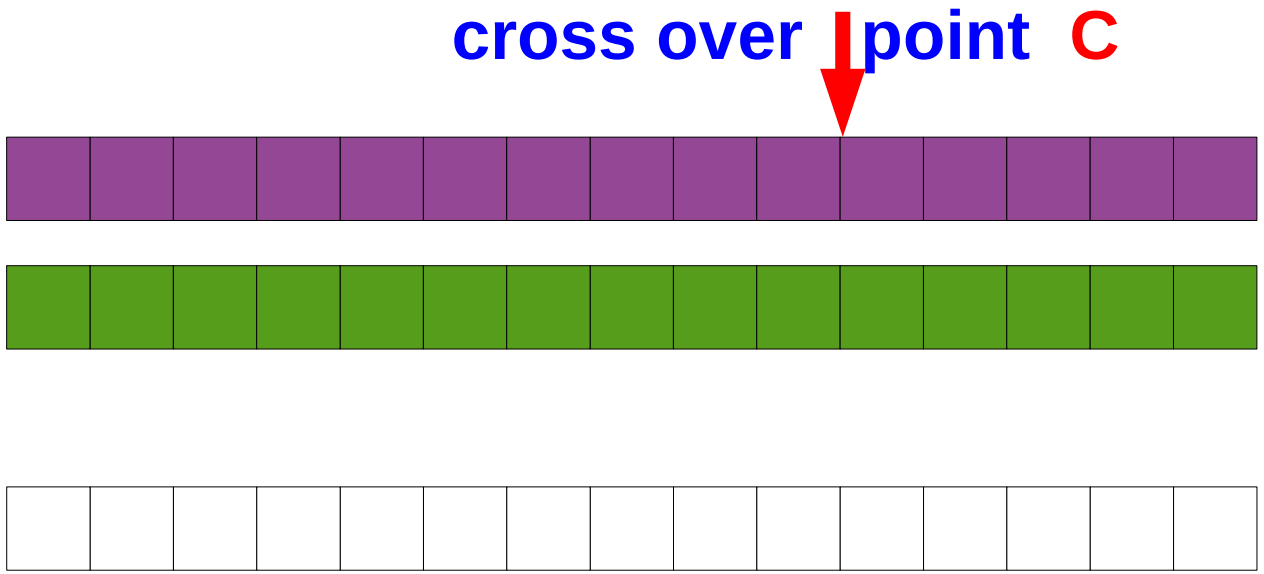


?

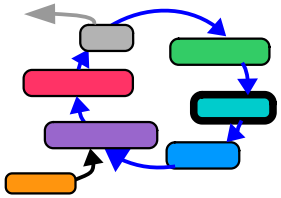
EA: Inheritance



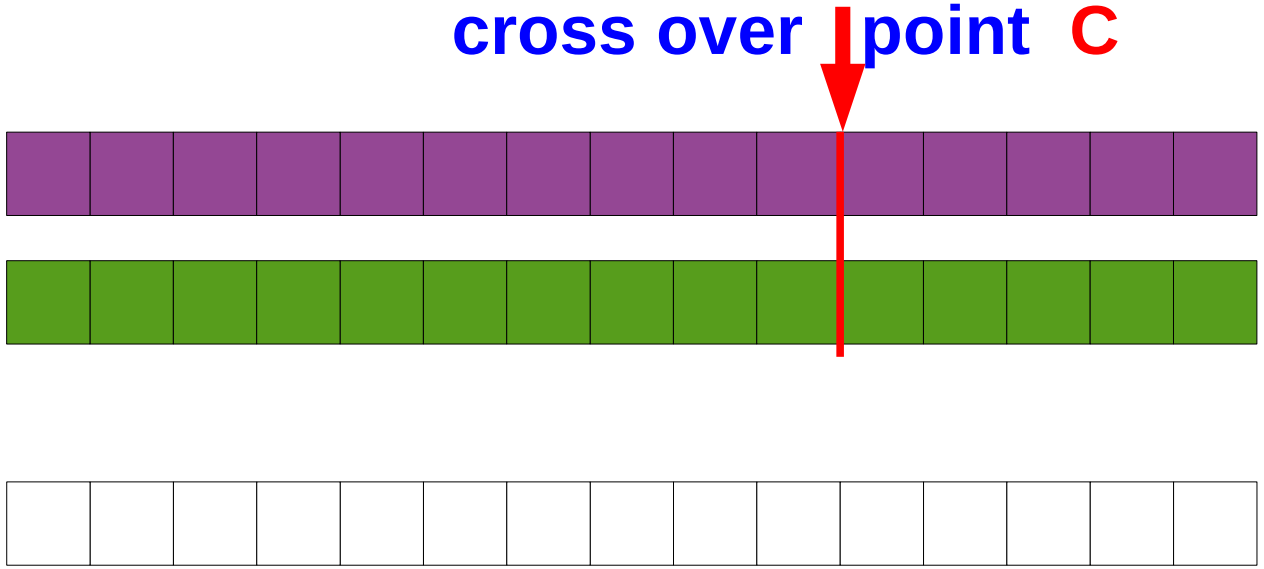
Recombination by **1-point cross over**:



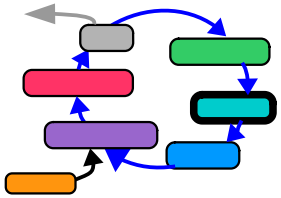
EA: Inheritance



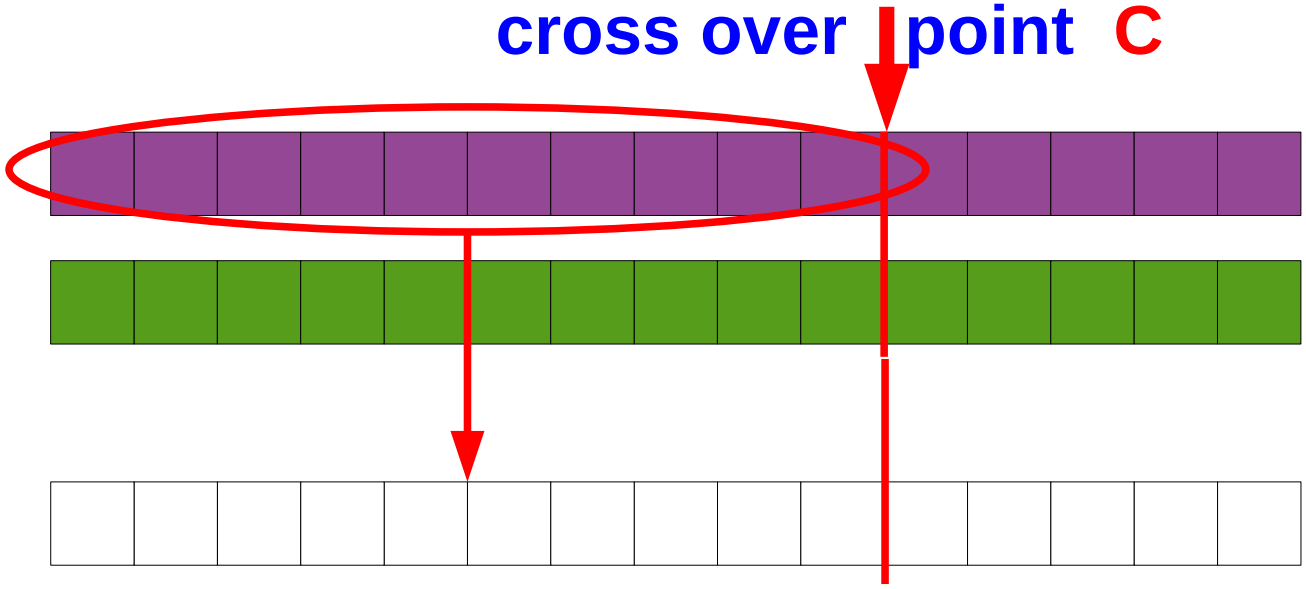
Recombination by **1-point cross over**:



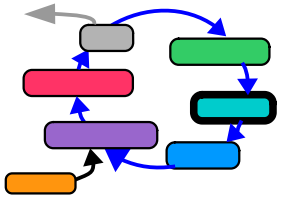
EA: Inheritance



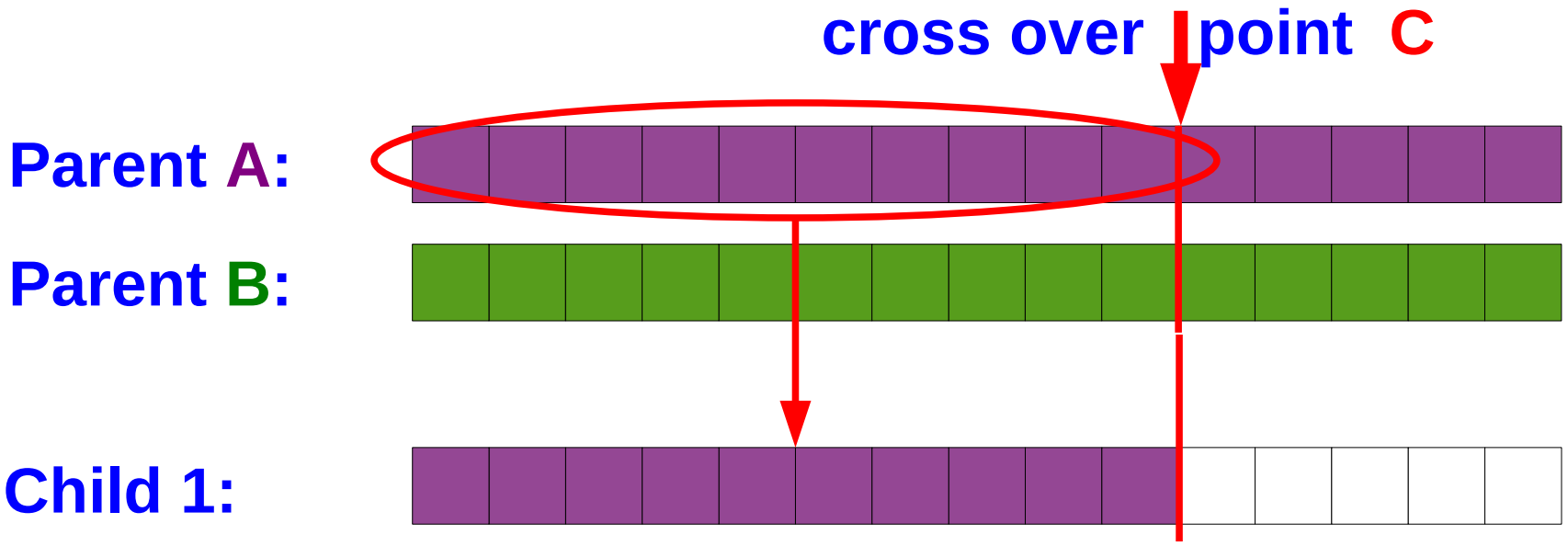
Recombination by **1-point cross over**:



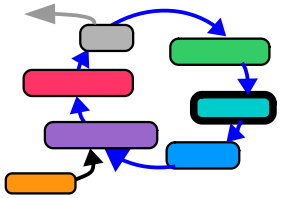
EA: Inheritance



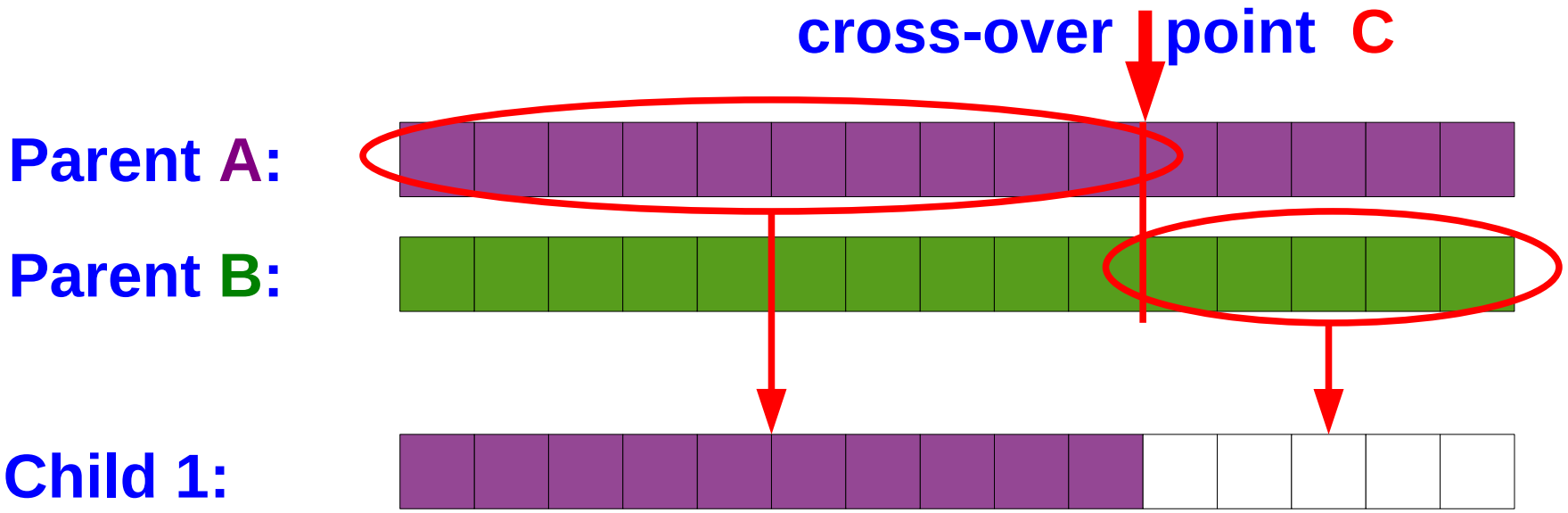
Recombination by **1-point cross over**:



EA: Inheritance

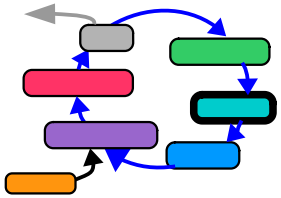


Recombination by **1-point cross-over**:

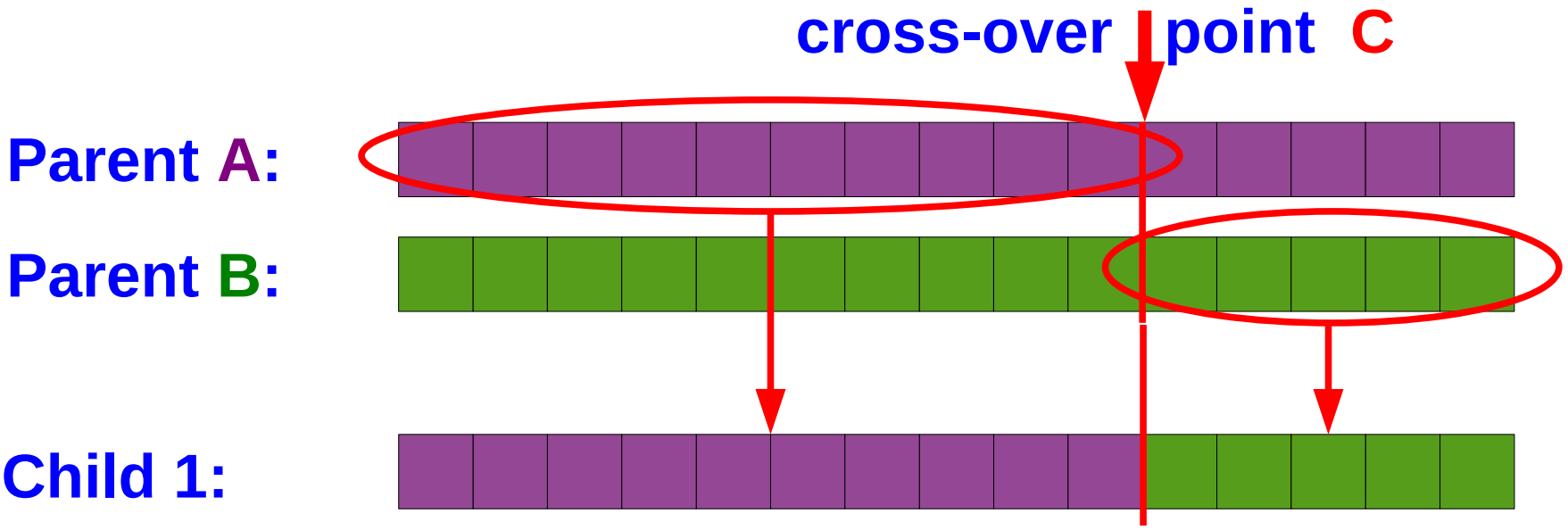


EA:

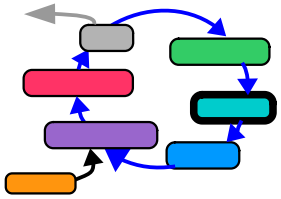
Inheritance



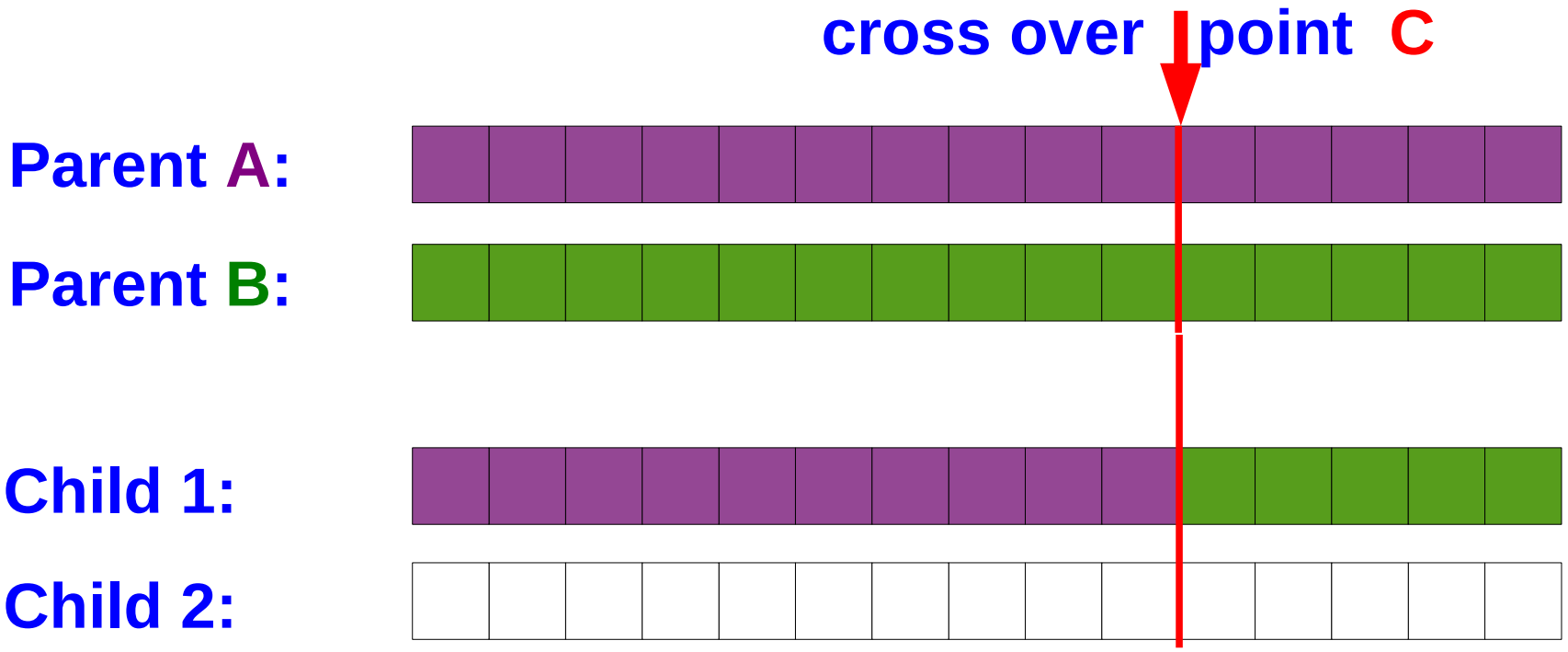
Recombination by 1-point cross-over:



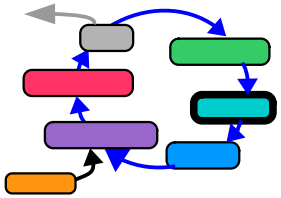
EA: Inheritance



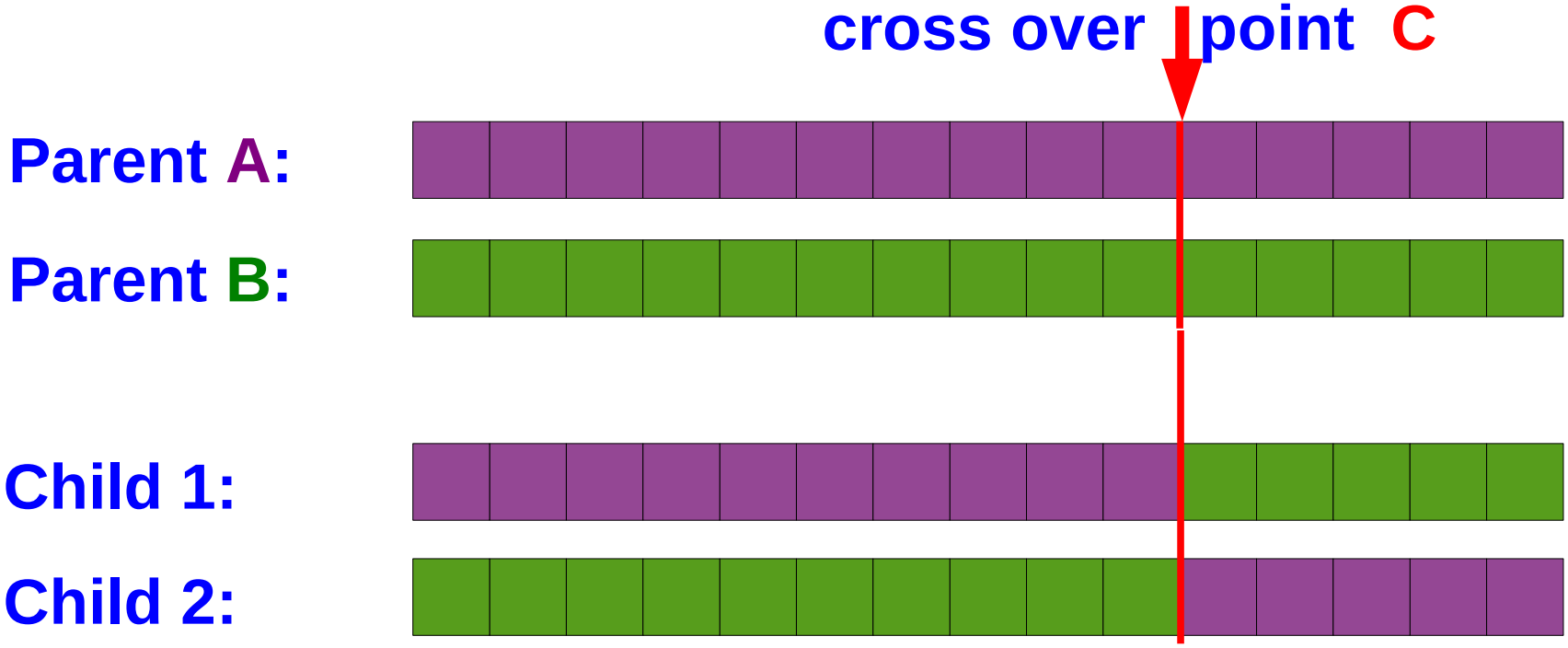
Recombination by **1-point cross over**:



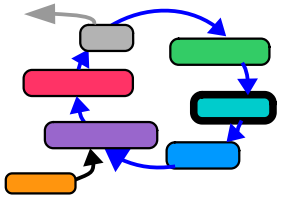
EA: Inheritance



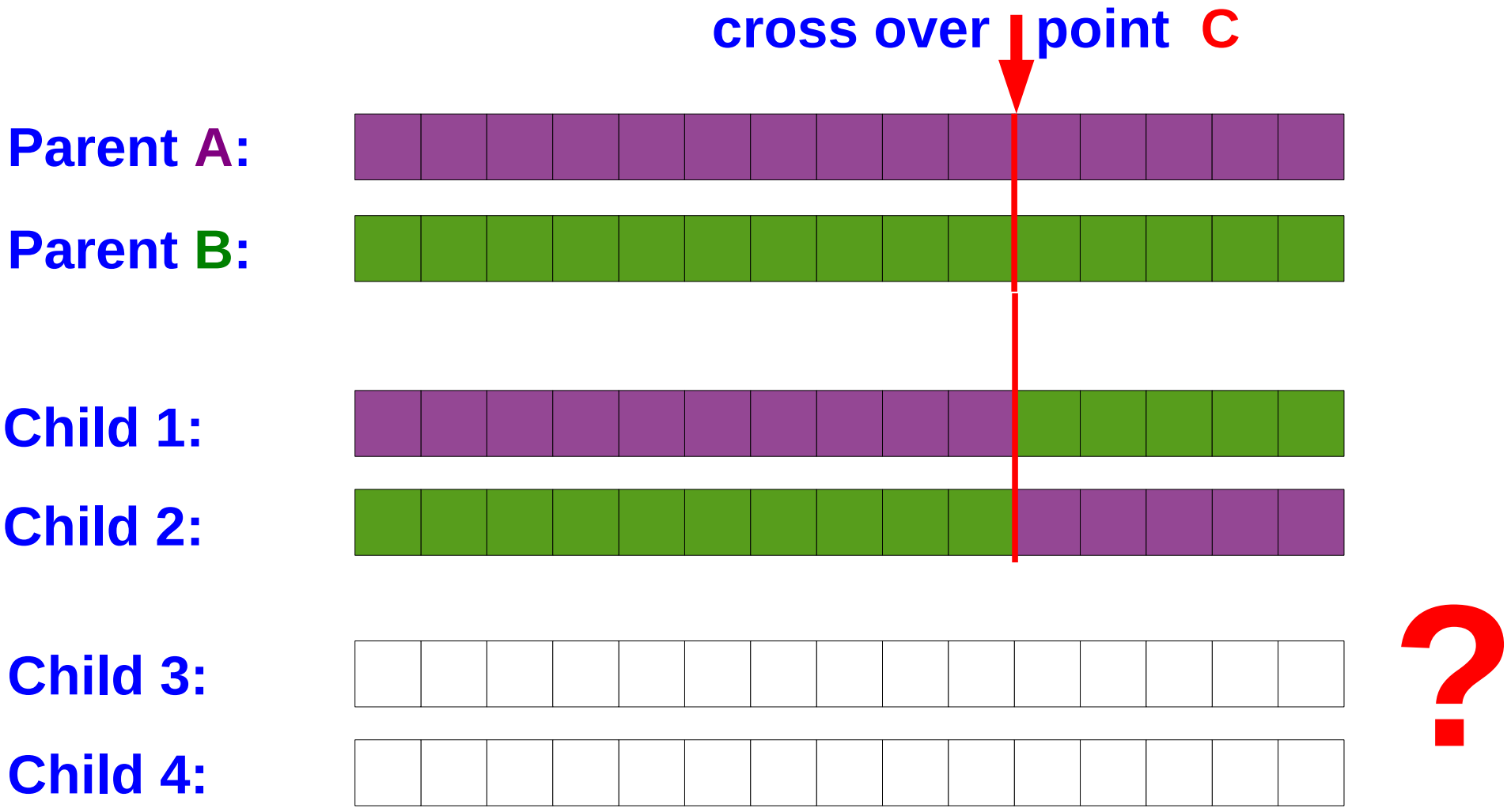
Recombination by **1-point cross over**:



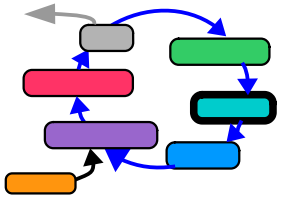
EA: Inheritance



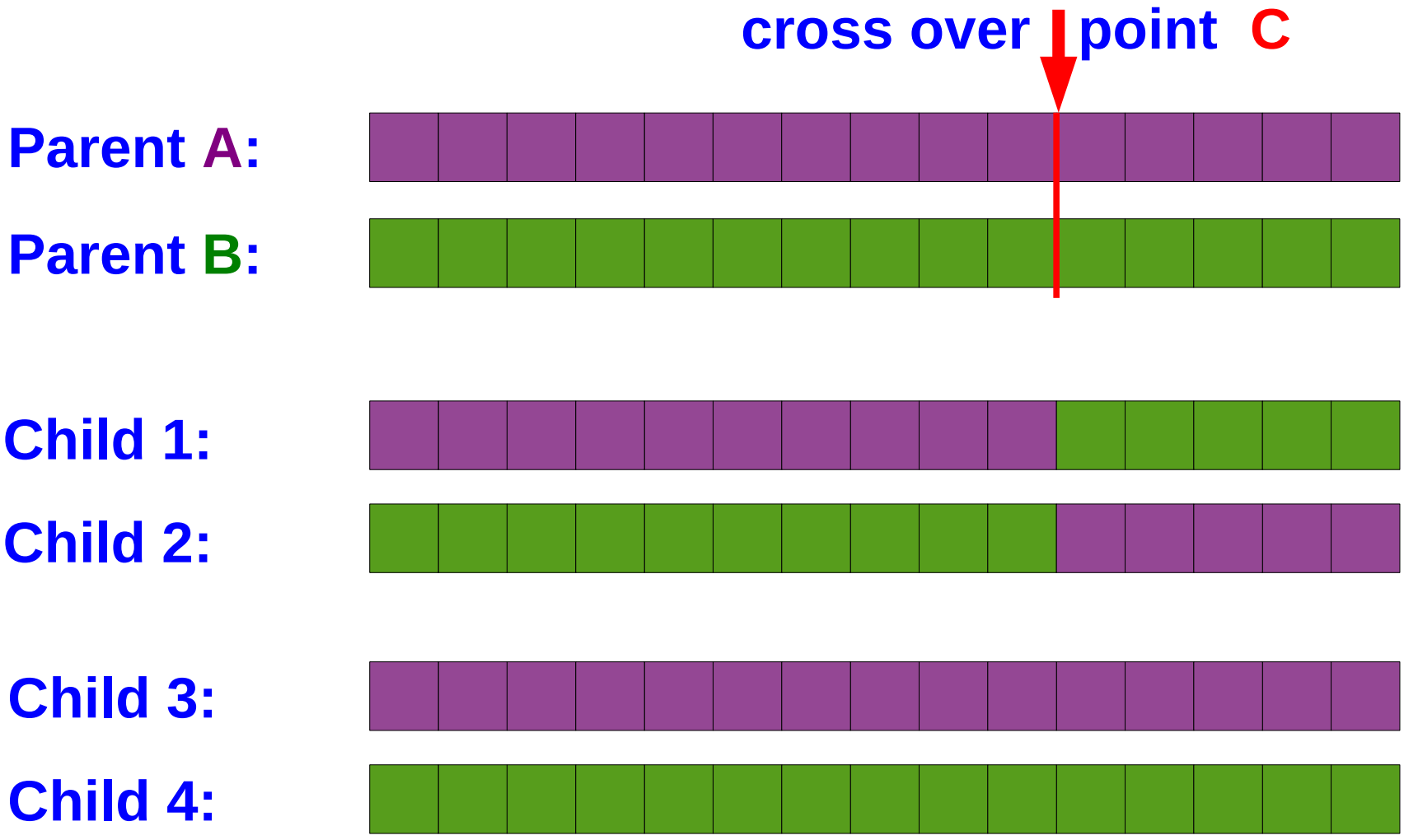
Recombination by 1-point cross over:



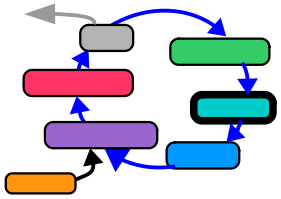
EA: Inheritance



Recombination by 1-point cross over:



EA: Inheritance



k=2 and k>2, recombination with k parents

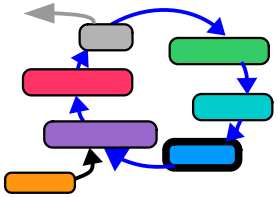
Several ways to **combine** (or **re-combine**) the genomes of the k parents have been proposed.
Some of them rely on a special structure of the genome.

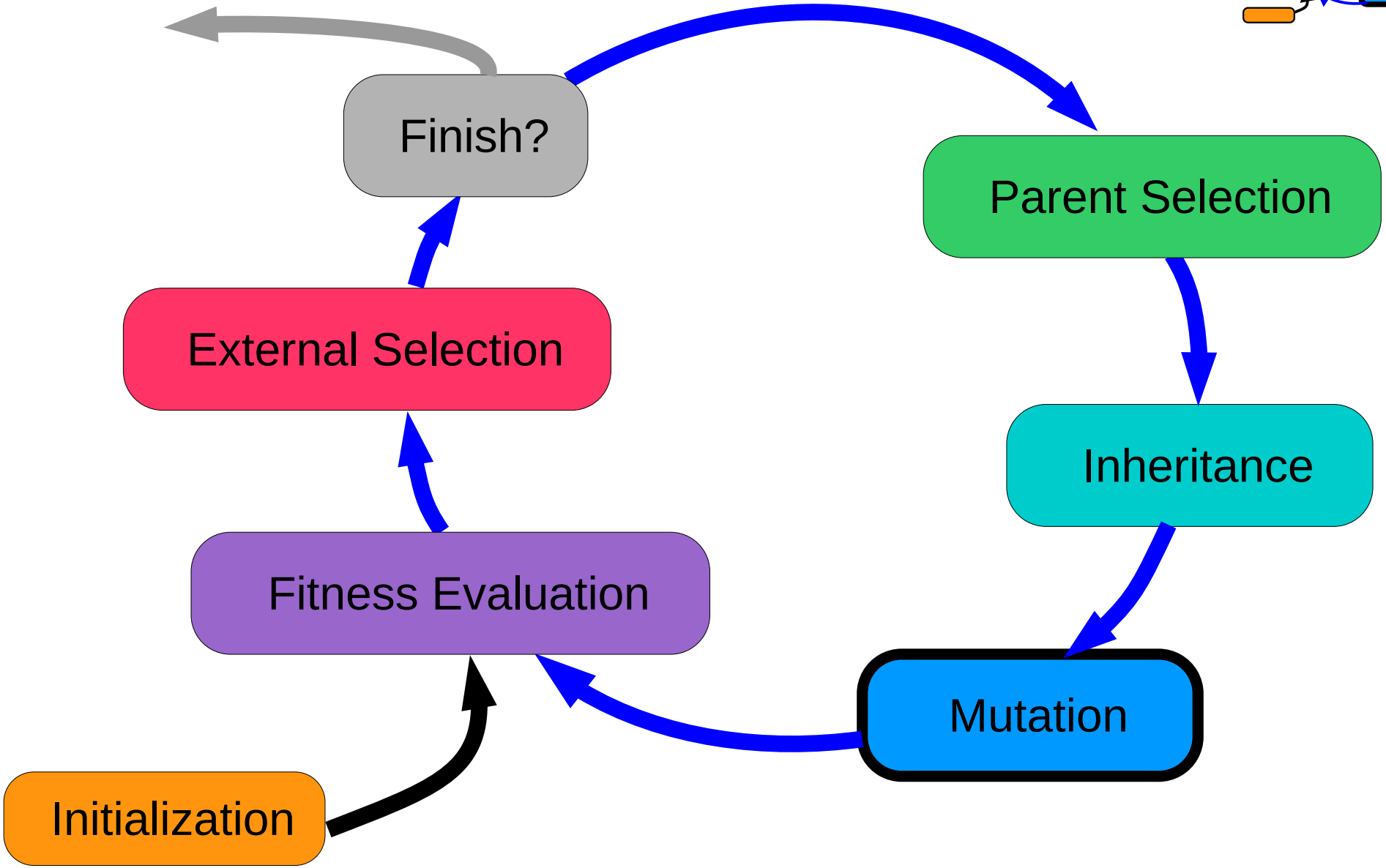
- **N-point cross over** (the most popular)
- **mean-values,** one component, all components
- **max, min, sum, difference** of values
- **union, intersection** of **sets** of items
- **concatenate** (strings)
- **...**

Overview:

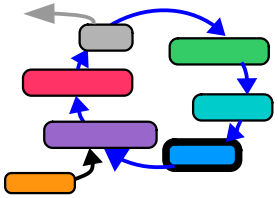
- Evolutionary Computation
- Historic Remarks
- Different Approaches
- Idea of Evolutionary Algorithms (EA)
- EA Steps
  - Individual, Genome, Fitness, Population
  - Parent selection
  - Inheritance
  - **Mutation**
  - Fitness evaluation
  - External selection

EA cycle of operation:





EA: Mutation

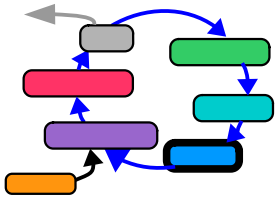


The job of the **mutation** step is to maintain the **diversity** of the genomes within the population:

Different positions in search space sample the fitness function at different points.

Changing the genome by **mutation** is implementing the **exploration** principle of **stochastic optimization.**

EA: Mutation



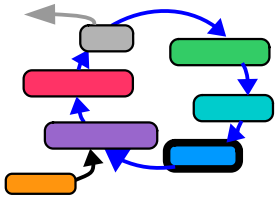
Depending on the different way the information is coded in the genome, the layout of the **mutation** step is to be designed differently.

Some common mutation operators are:

- **Binary genome**: a bit-flip
- **Vector of parameters**: change of value
- **Set of items**: replace a single item
- **String of characters**: replace a single character
- **Sequence**: change order within sequence
- **Other:** other method

Special codings of the genome will require that the **mutation** is paying respect to this coding.

EA: Mutation



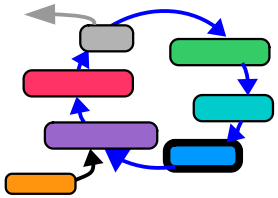
The **mutation** for a binary genome is typically implemented as one or more **bit-flip**s:

Method 1:

Choose a random position **r** in the genome and flip that bit.
This yields a 100% chance that the genome is changed;
the resulting Hamming distance is always exactly 1.

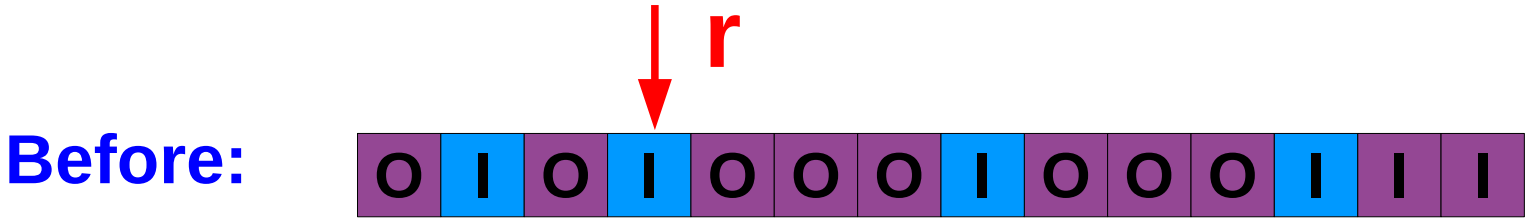
Before:

EA: Mutation

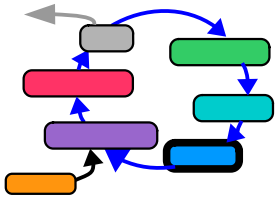


The **mutation** for a binary genome is typically implemented as one or more **bit-flip**s:

Method 1:
Choose a random position **r** in the genome and flip that bit.
This yields a 100% chance that the genome is changed;
the resulting Hamming distance is always exactly 1.

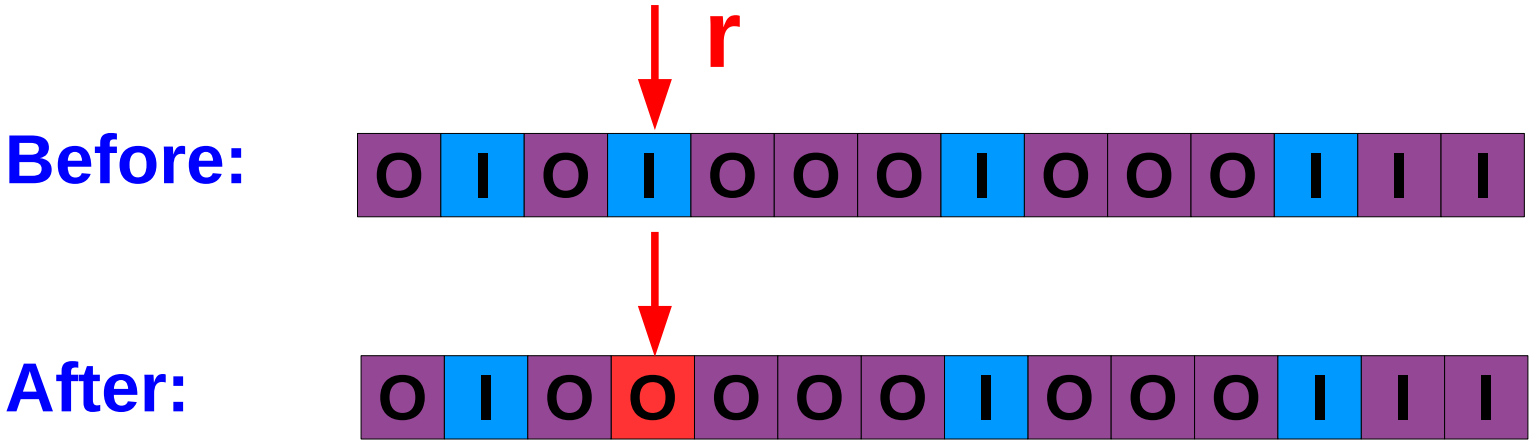


EA: Mutation

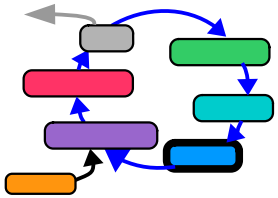


The **mutation** for a binary genome is typically implemented as one or more **bit-flip**s:

Method 1:
Choose a random position **r** in the genome and flip that bit.
This yields a 100% chance that the genome is changed;
the resulting Hamming distance is always exactly 1.



EA: Mutation



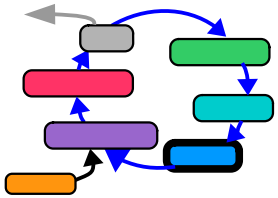
The **mutation** for a binary genome is typically implemented as one or more **bit-flip**s:

Method 2:
Go through all positions in the genome, and **flip** each **bit** with a given **bit-flip**-probability ω .
There is a chance, that all bits are flipped,
and a chance that none is flipped.

Before:

I	I	O	I	I	I	I	O	I	O	O	O	I	O

EA: Mutation

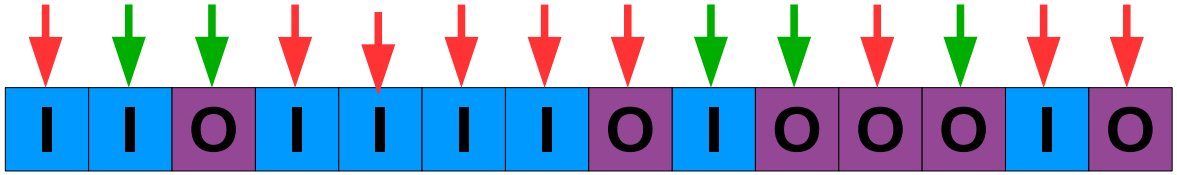


The **mutation** for a binary genome is typically implemented as one or more **bit-flip**s:

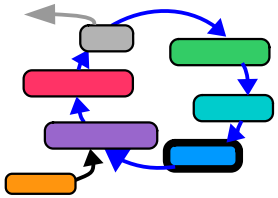
Method 2:
Go through all positions in the genome, and **flip** each **bit** with a given **bit-flip**-probability ω .
There is a chance, that all bits are flipped, and a chance that none is flipped.

$0.0 < \omega < 1.0$

Before:



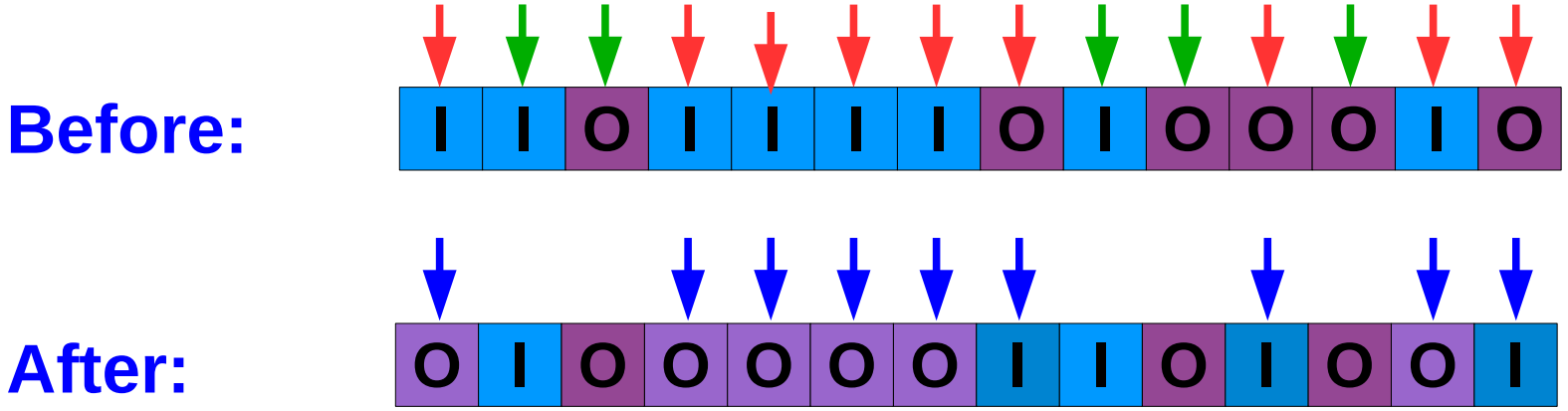
EA: Mutation



The **mutation** for a binary genome is typically implemented as one or more **bit-flip**s:

Method 2:
Go through all positions in the genome, and **flip** each **bit** with a given **bit-flip**-probability ω .
There is a chance, that all bits are flipped, and a chance that none is flipped.

$0.0 < \omega < 1.0$



EA: Mutation & Inheritance

It is possible, that the inheritance and the mutation process yield invalid genomes, which represent hypotheses **s** from outside the allowed search space **S**.

Those illegal genomes would stress the optimization process in an unwanted way.

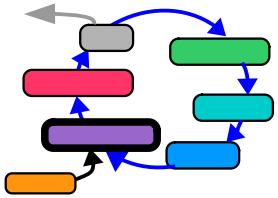
Therefore, it is a good advise, to take precautions against illegal genomes (if possible, and economic):

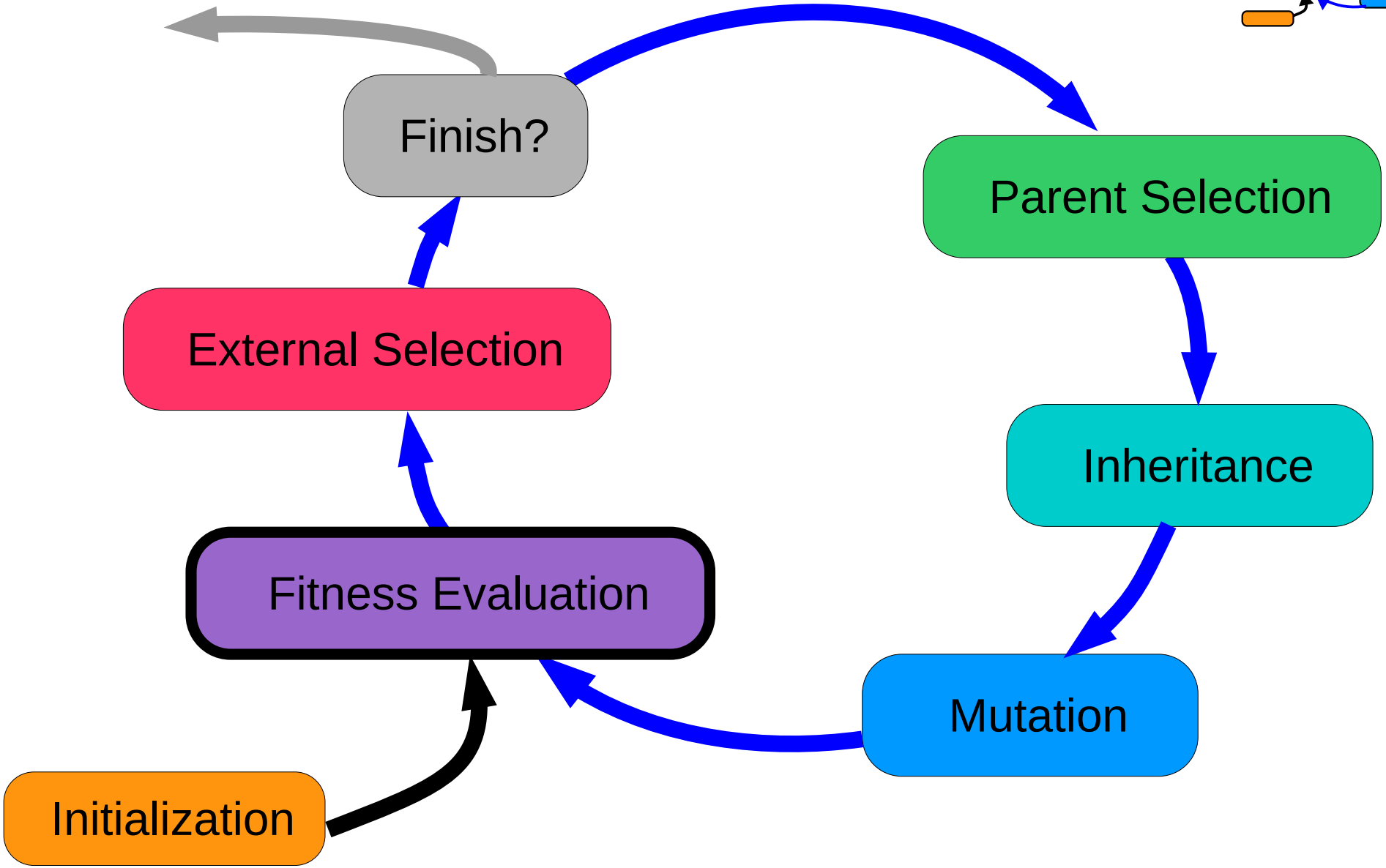
- shape the structure of the genome,
- shape the structure of the inheritance,
- shape the structure of the mutation.

Overview:

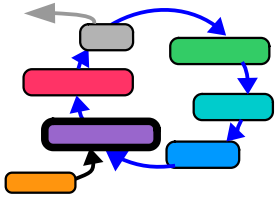
- Evolutionary Computation
- Historic Remarks
- Different Approaches
- Idea of Evolutionary Algorithms (EA)
- EA Steps
  - Individual, Genome, Fitness, Population
  - Parent selection
  - Inheritance
  - Mutation
  - **Fitness evaluation**
  - External selection

EA cycle of operation:



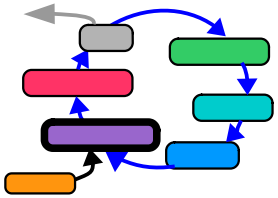


EA: Fitness Evaluation



The **fitness** of each individual is evaluated with respect to the **fitness-function** (most case identical to the objective function). For Evolutionary Algorithms, it is usual to have a **fitness** function that is to be maximized.

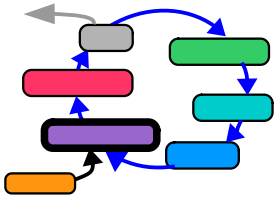
EA: Fitness Evaluation



The **fitness** of each individual is evaluated with respect to the **fitness-function** (most case identical to the objective function). For Evolutionary Algorithms, it is usual to have a **fitness** function that is to be maximized.

If the **fitness evaluation** is expensive, it is a good idea to process only the new individuals/genomes from the inheritance step, including those individuals/genomes that have changed during mutation.

EA: Fitness Evaluation



The **fitness $f(g_p)$** of each individual **p** is evaluated with respect to the application using the given **fitness-function f(g)**.

Genome g_p

p=P

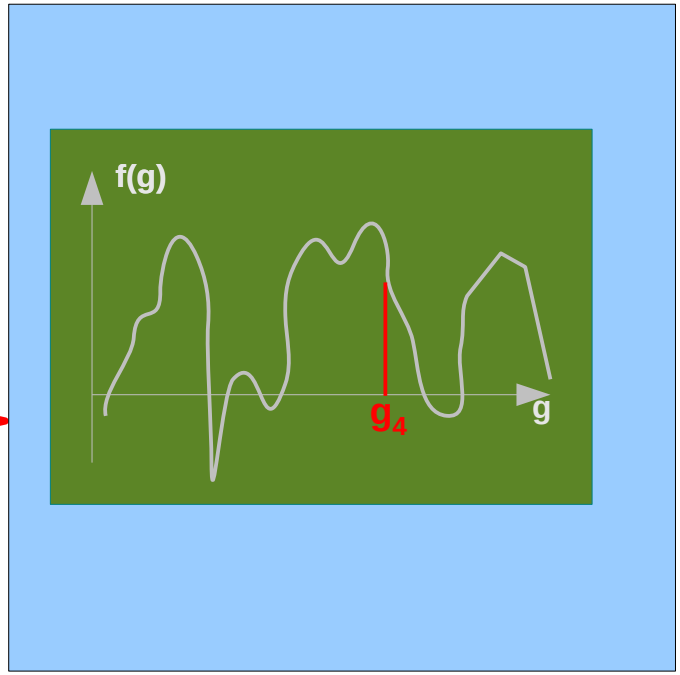
p=5

p=4

p=3

p=2

p=1



Fitness Value $f(g_p)$

$f(g_p)$ =

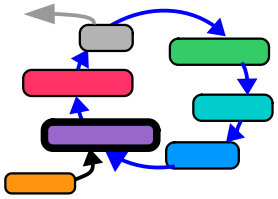
$f(g_4)$ = ???

$f(g_3)$ = 0.228

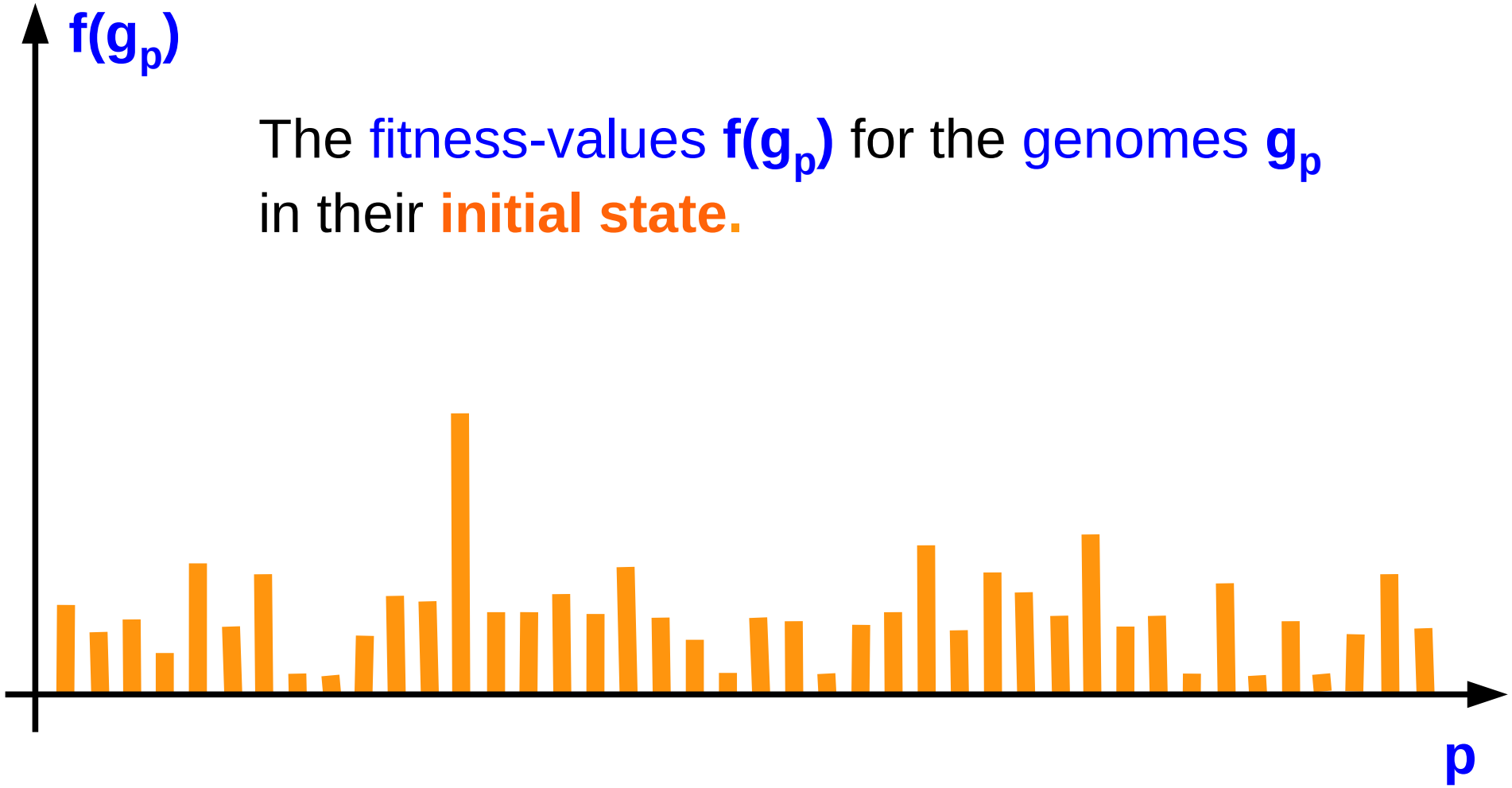
$f(g_2)$ = - 171.0

$f(g_1)$ = 0.5

EA: Fitness Evaluation

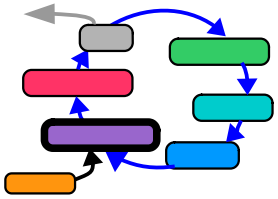


Distribution of **fitness values** over the population.

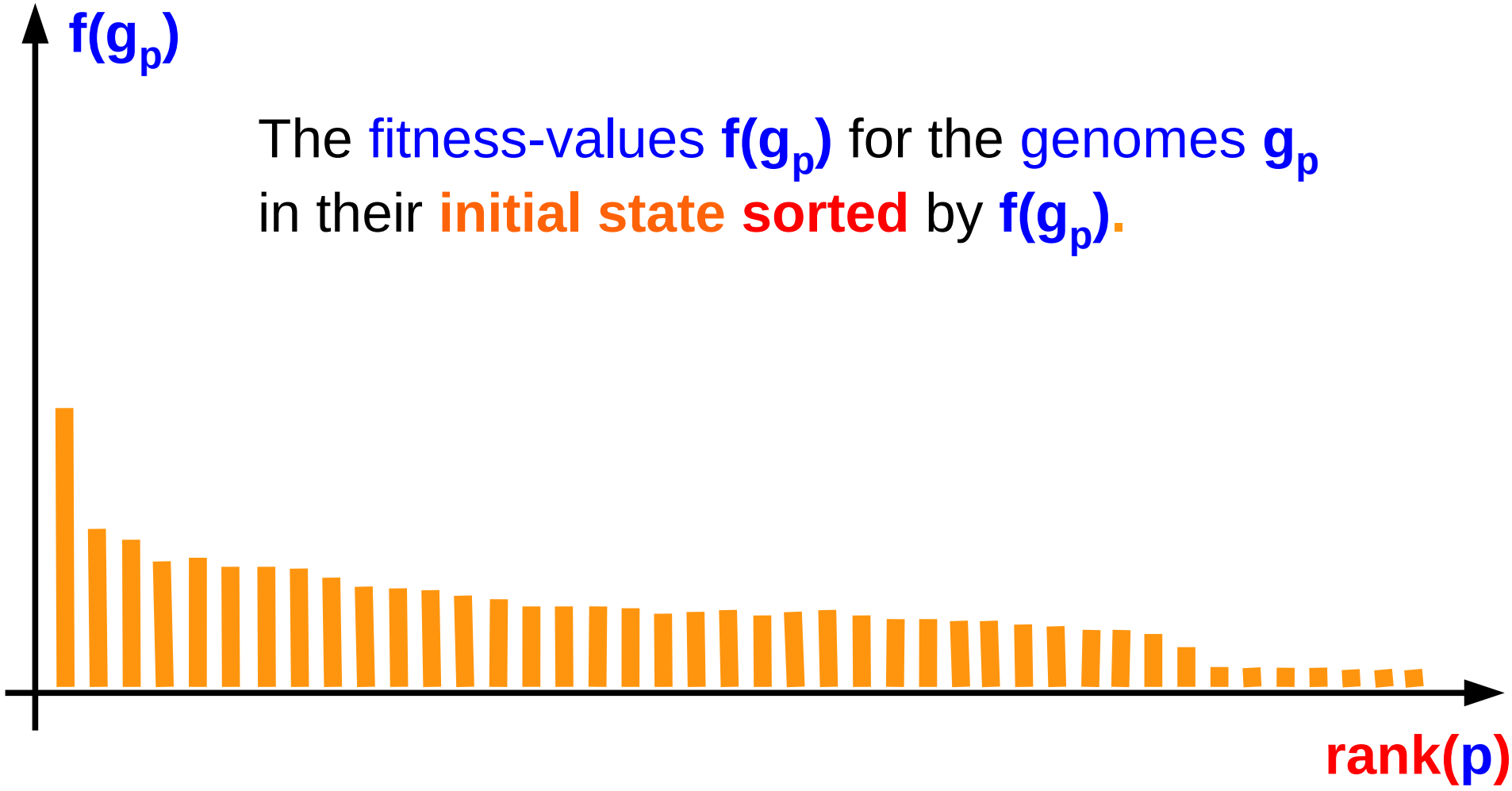


The fitness-values $\mathbf{f(g_p)}$ for the genomes $\mathbf{g_p}$ in their **initial state.**

EA: Fitness Evaluation

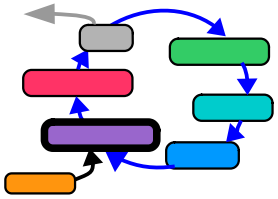


Distribution of **fitness values** over the population.

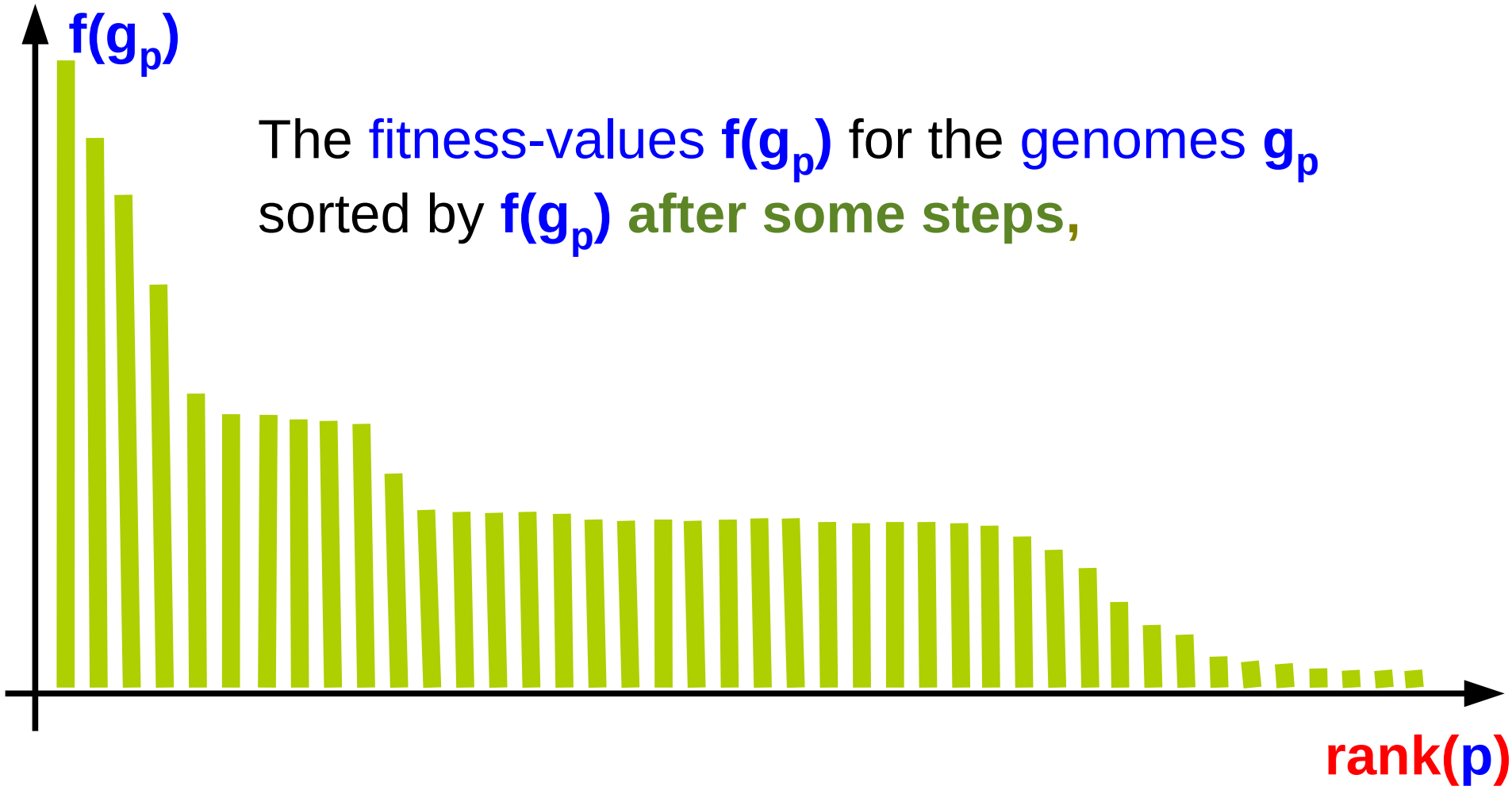


The fitness-values $\mathbf{f(g_p)}$ for the genomes $\mathbf{g_p}$ in their **initial state sorted** by $\mathbf{f(g_p)}$.

EA: Fitness Evaluation

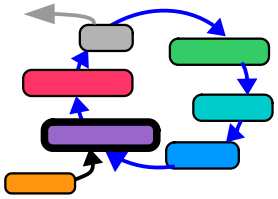


Distribution of **fitness values** over the population.

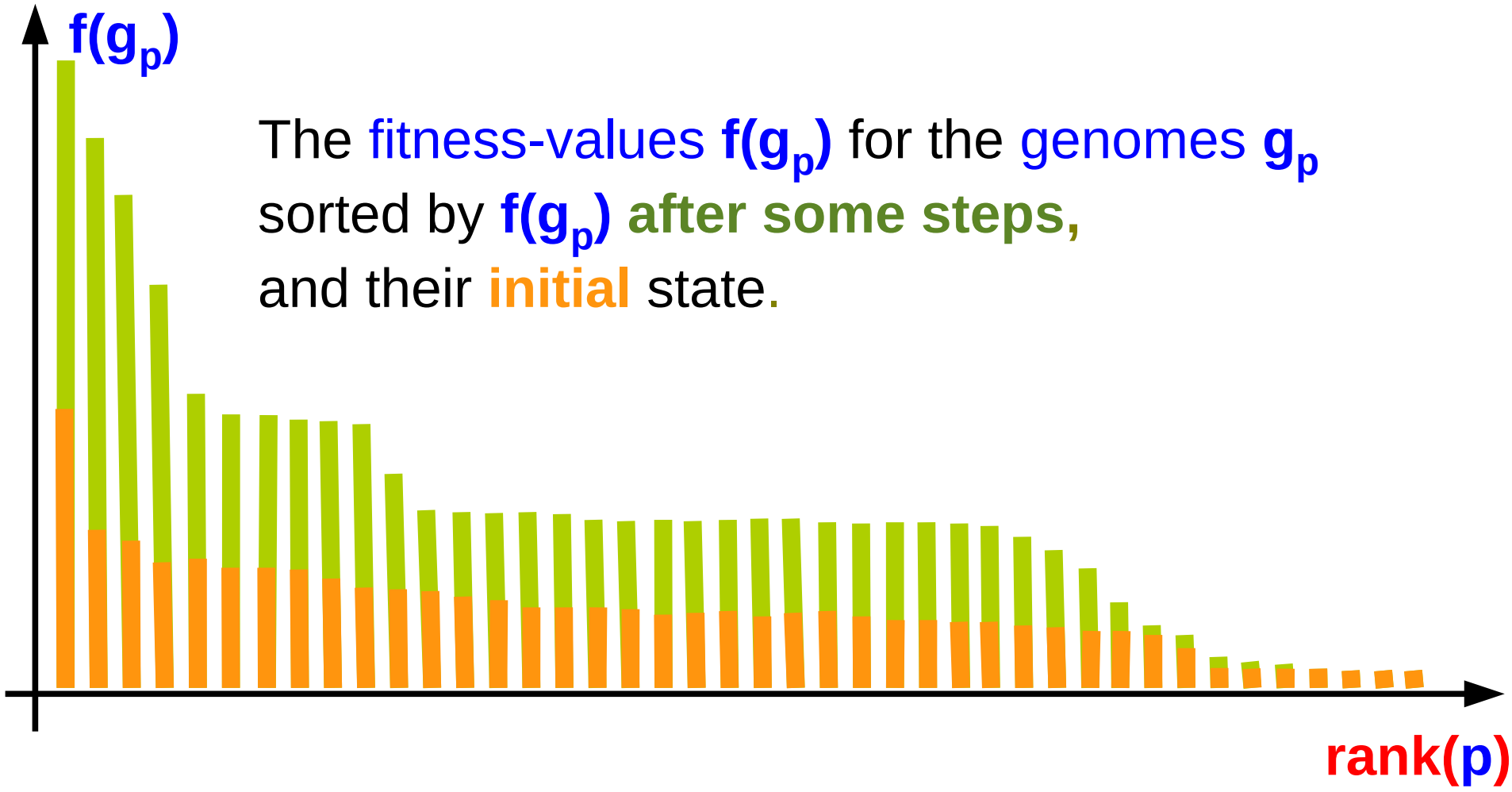


The fitness-values $\mathbf{f(g_p)}$ for the genomes $\mathbf{g_p}$ sorted by $\mathbf{f(g_p)}$ **after some steps,**

EA: Fitness Evaluation



Distribution of **fitness values** over the population.

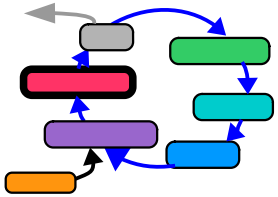


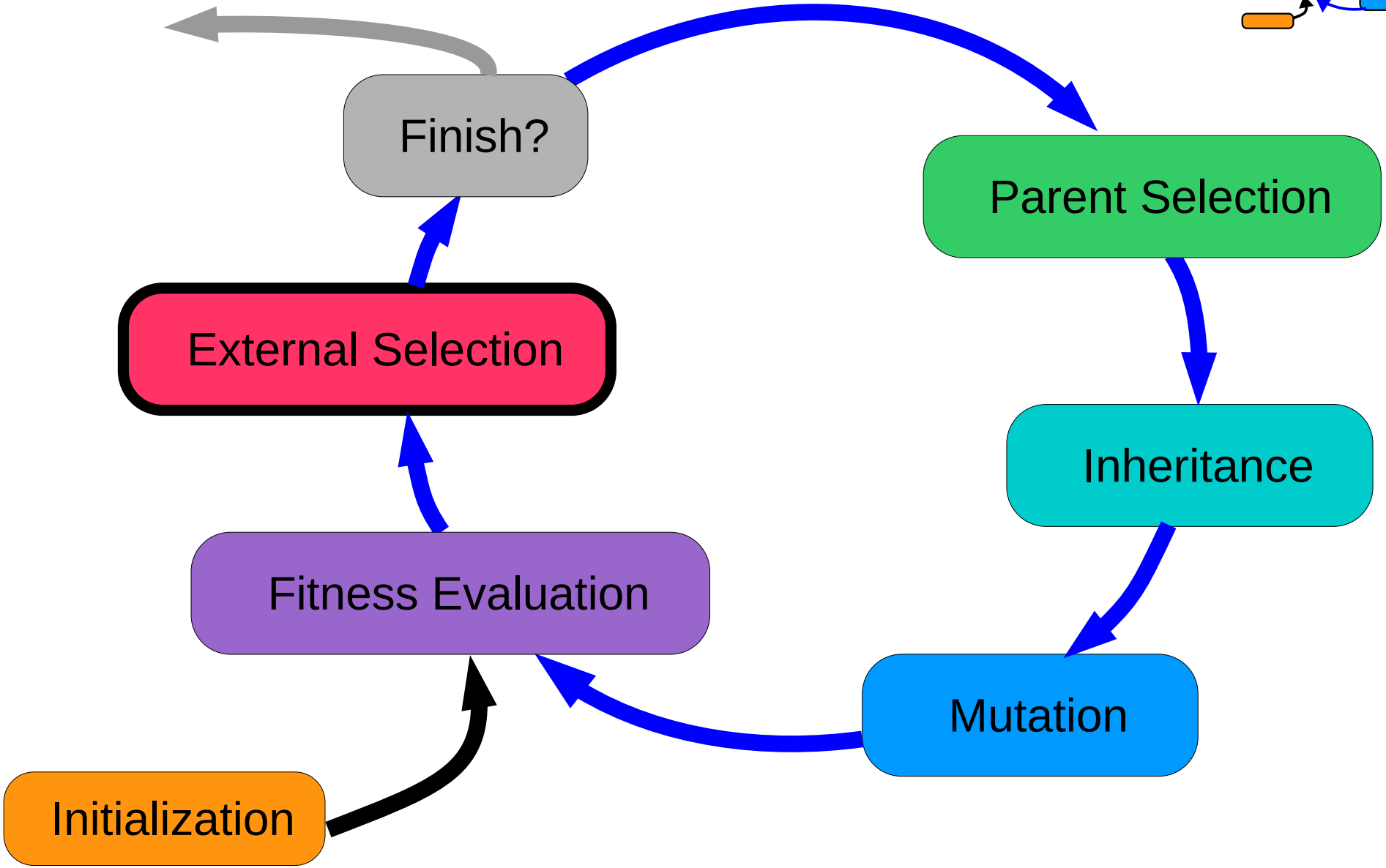
The fitness-values $\mathbf{f(g_p)}$ for the genomes $\mathbf{g_p}$ sorted by $\mathbf{f(g_p)}$ **after some steps,** and their **initial** state.

Overview:

- Evolutionary Computation
- Historic Remarks
- Different Approaches
- Idea of Evolutionary Algorithms (EA)
- EA Steps
  - Individual, Genome, Fitness, Population
  - Parent selection
  - Inheritance
  - Mutation
  - Fitness evaluation
  - **External selection**

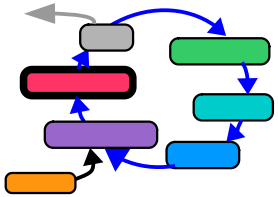
EA cycle of operation:





EA:

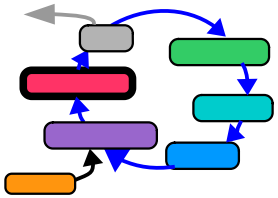
External Selection



The purpose of the **selection process** is to keep the „Best“, and discard the „Losers“.
The **selection** is based on the achieved fitness, which is obtained by the fitness function, quasi from **external** to the EA.

The μ individuals that survive the **external selection** process, will be the pool for the parents of the next generation. Later on, from these μ parents the λ offspring will be generated through the inheritance process.

EA: External Selection



The two questions for the **external selection** are:

How many (μ) of the individuals shall survive?

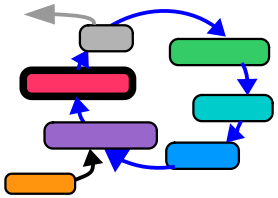
and

Which are the ones to keep?

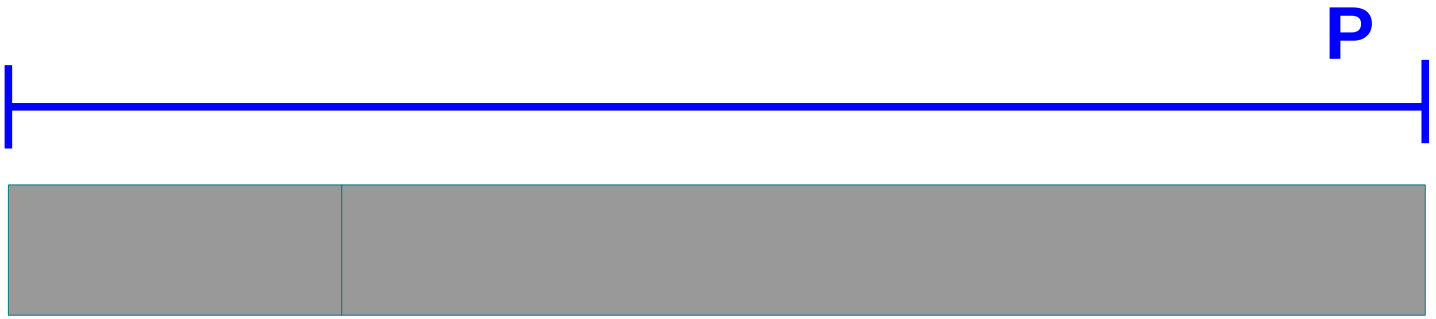
A common way is to keep the population constant to **P** individuals.
discard λ by external selection
and keep μ as pool for the parents,
then generate λ offspring for the next generation.

EA:

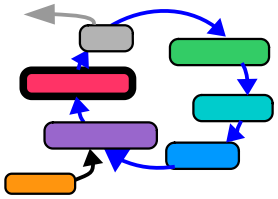
External Selection



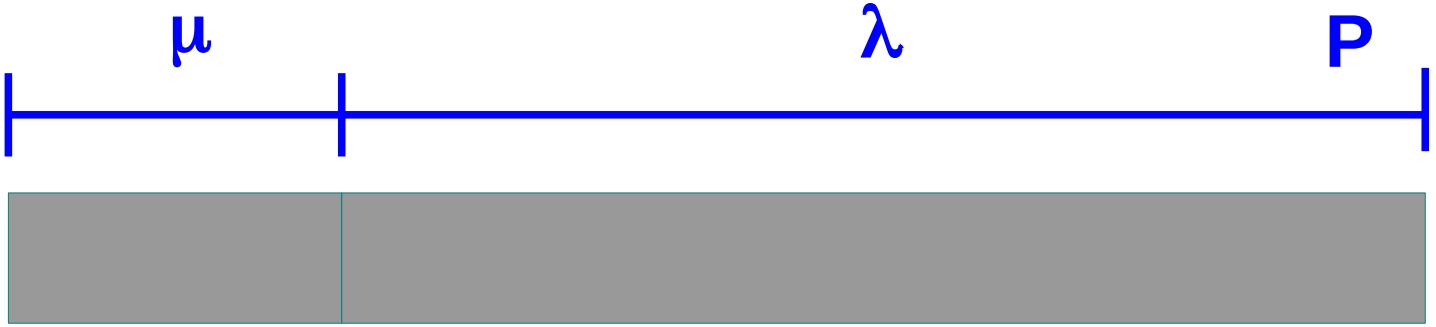
Have **P**, keep μ, generate λ offspring



EA: External Selection

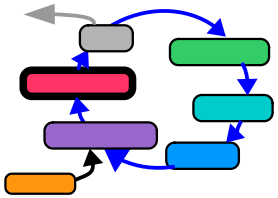


Have **P**, keep **μ**, generate **λ** offspring

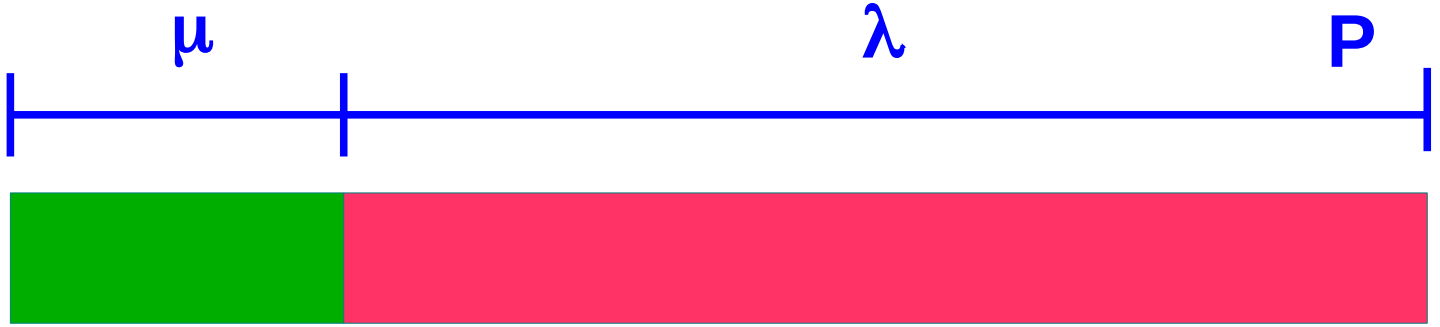


EA:

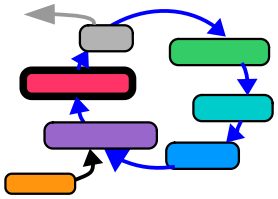
External Selection



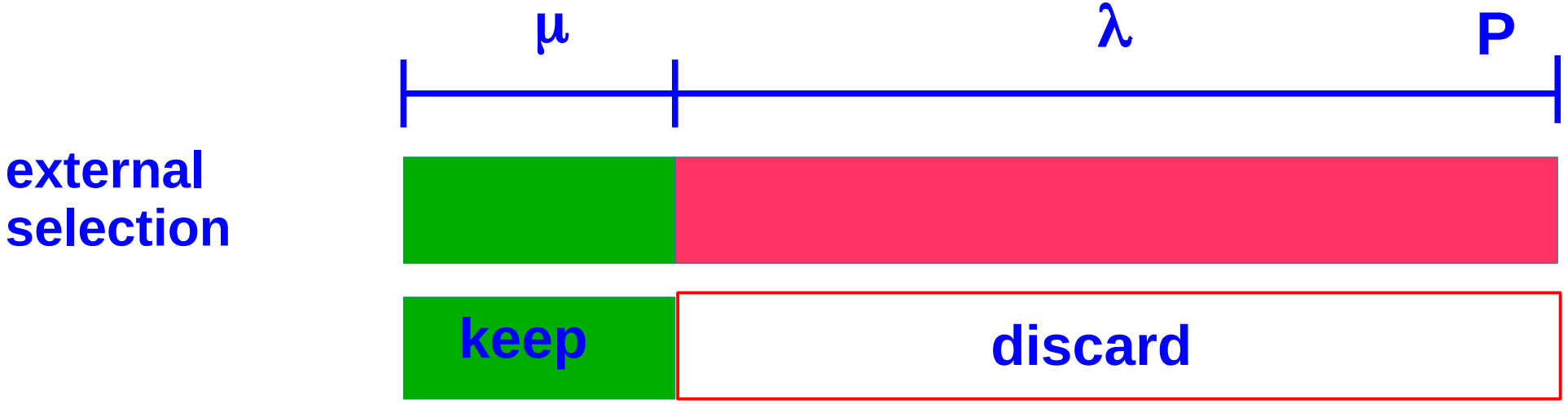
Have **P**, keep **μ**, generate **λ** offspring



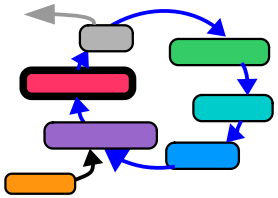
EA: External Selection



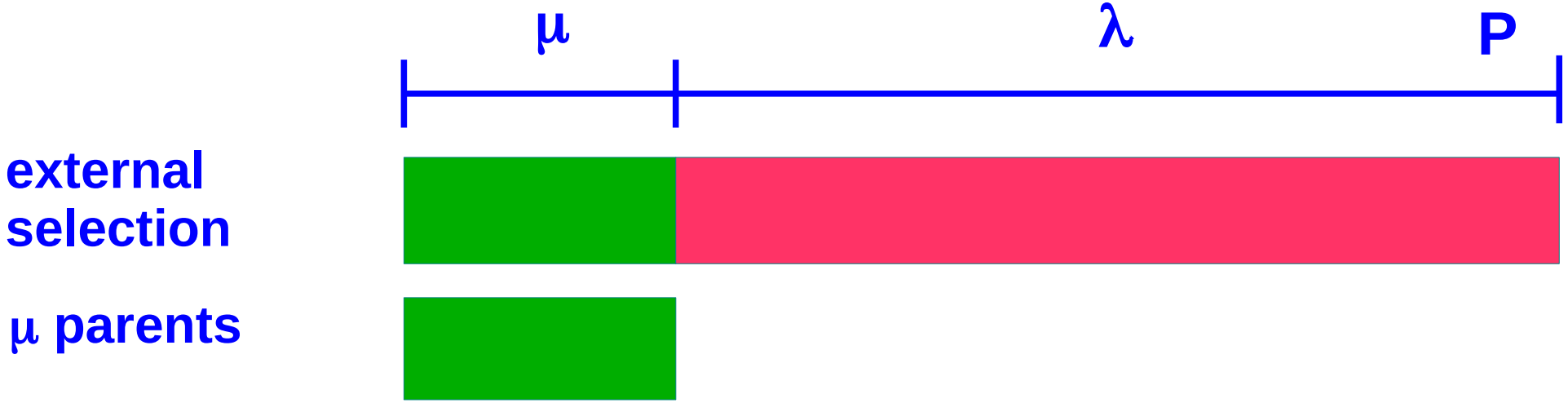
Have **P**, keep **μ**, generate **λ** offspring



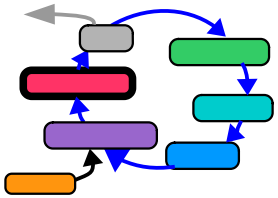
EA: External Selection



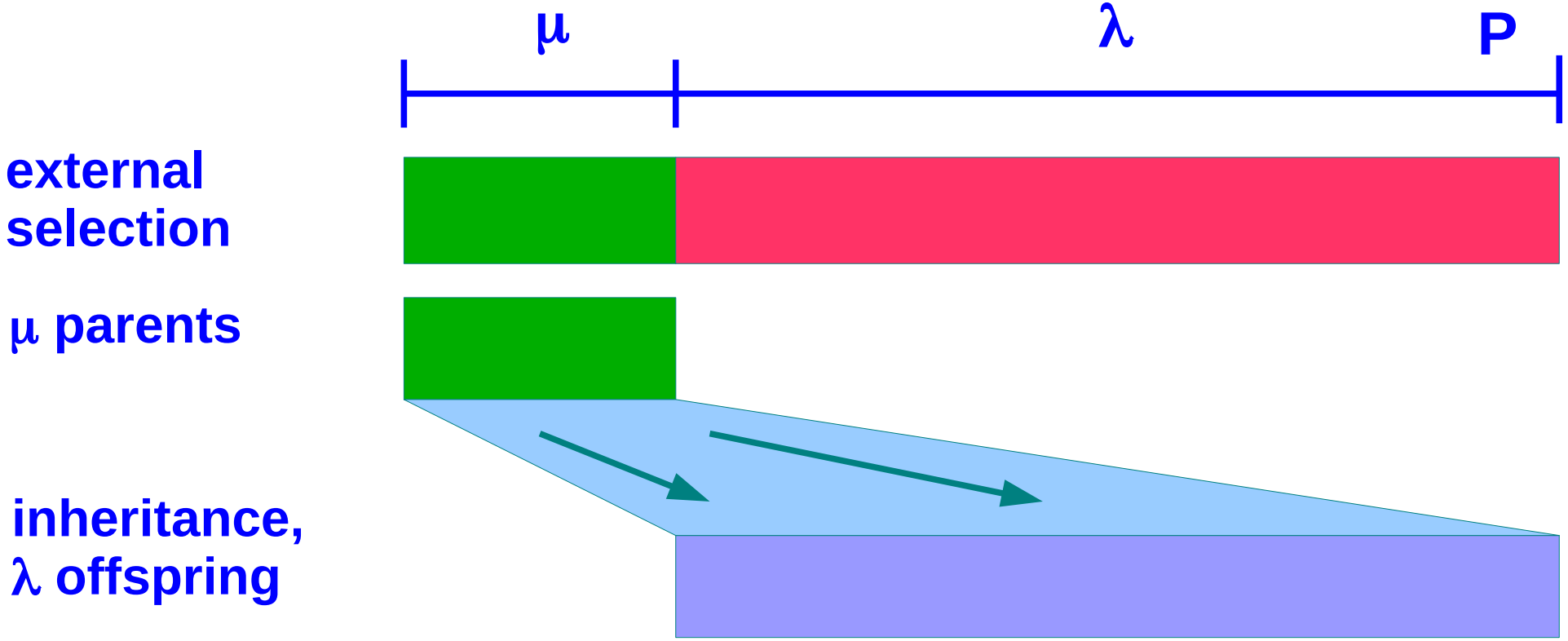
Have **P**, keep **μ**, generate **λ** offspring



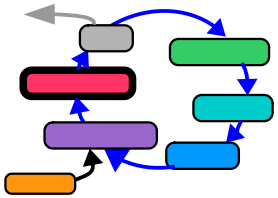
EA: External Selection



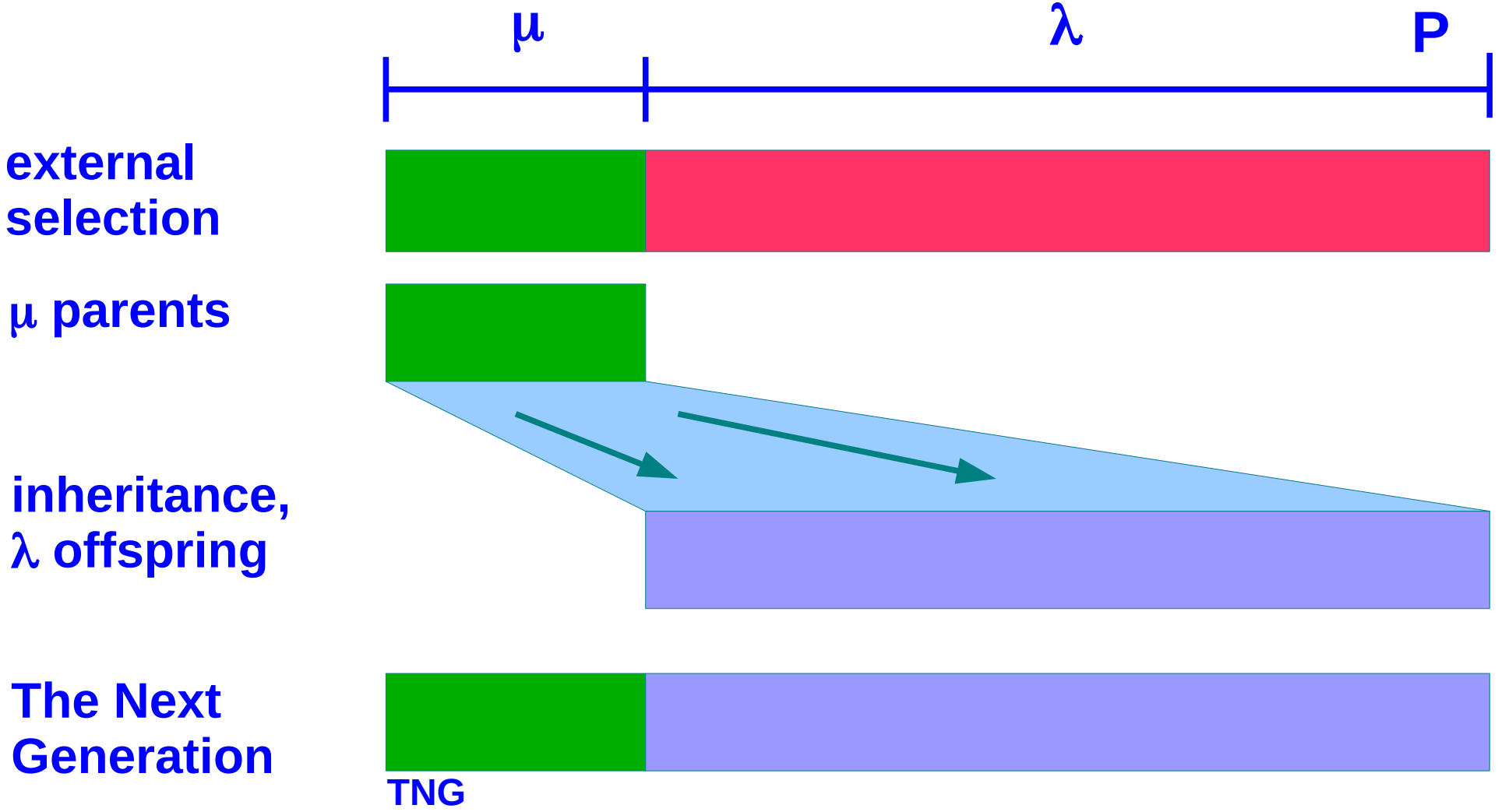
Have **P**, keep **μ**, generate **λ** offspring



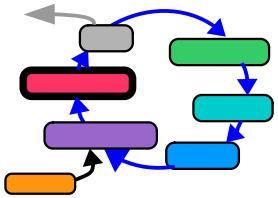
EA: External Selection



Have **P**, keep **μ**, generate **λ** offspring



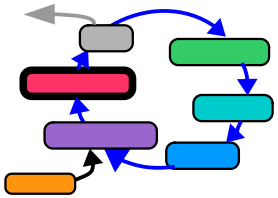
EA: External Selection



Common strategies for the **external selection** are:

- random choice
- schedule based, e.g. round robin
- fitness based elitism:
  - fitness proportional choice
  - rank proportional choice
- fitness based stochastic:
  - fitness proportionate, probabilistic choice
  - rank proportionate, probabilistic choice
- combinations of the above

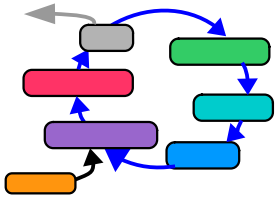
EA: External Selection

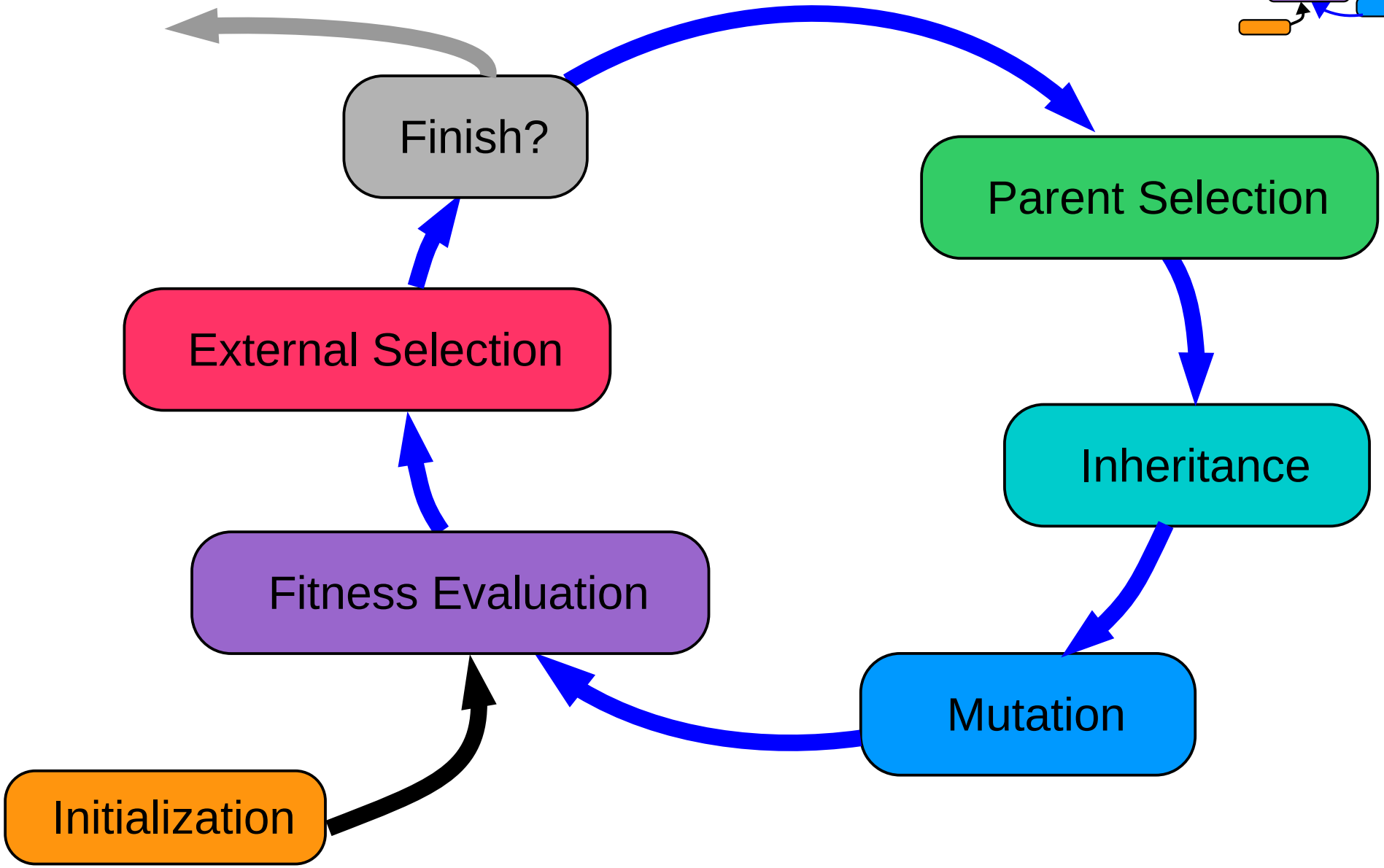


There are different ideas and principles behind the selection of the μ possible parents from the population, with special pros and cons.

- **deterministic** or **stochastic** selection
- **fitness dependent** selection
- **fitness proportional** selection
- **rank-based** selection
- **tournament** selection
- **life-time based** selection
- **combinations** of the above + more

EA cycle of operations:





Artificial Life
Summer 2025

Learning from Nature
Evolutionary Algorithms
Optimization Inspired by Biology

Master Computer Science [MA-INF 4201]
Mon 14:15 – 15:45, HSZ, HS-2

Dr. Nils Goerke, Autonomous Intelligent Systems,
Department of Computer Science, University of Bonn

Artificial Life
Summer 2025

Learning from Nature

Evolutionary Algorithms

Optimization Inspired by Biology

Thank you for attending the lecture

Dr. Nils Goerke, Autonomous Intelligent Systems,
Department of Computer Science, University of Bonn