Genetic Algorithms

Benchmark

Computational Intelligence 2023/2024

Luca Catalano – S308658
Claudio Savelli – S317680
Florentin-Cristian Udrea – S319029

INTRODUCTION

Problem Introduction

1 Task

Optimizing the value of an ignore fitness function on an array of bits.

2 Objective

Minimizing the number of fitness calls.

3 Method to Use

Genetic Algorithms. Search Heuristic inspired by the process of natural selection.

4 Constraints

None if not exploiting the problem on the specific solution.

Introduction Overview

Parent Selection

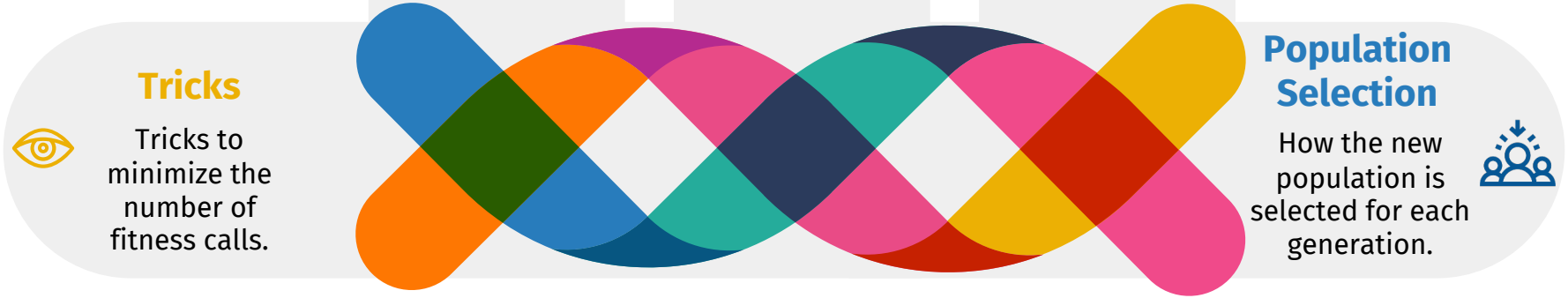
How parents are selected at each generation.

Reproduce Functions

How new genes are generated.

Mutations

Random changes in the population.



Tricks

Tricks to minimize the number of fitness calls.

Population Selection

How the new population is selected for each generation.

Parent Selection

Better with a lot of individuals

TOURNAMENT

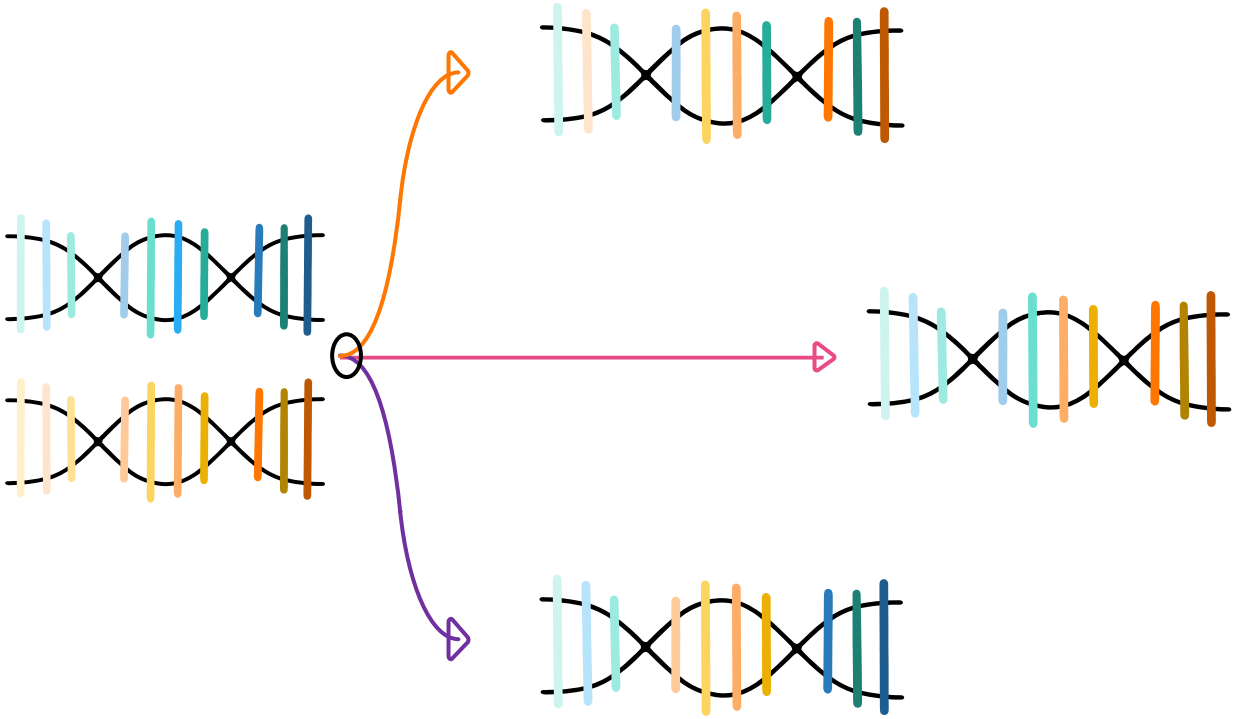
For each Generation, the parents are selected through tournaments where they compete based on their fitness.

Better with few individuals

ROULETTE

Each portion of a roulette wheel is assigned to an individual in the population based on their fitness. Individuals with higher fitness have larger portions.

Reproduce Function



Uniform
For each gene, choose one of the parents and select its gene of the same position.

One-Cut
A gene point is randomly selected. In a part, the gene of one of the two parents is copied, and in the second the other.

Two-Cuts
Same as One-Cut but two points are selected.

Mutation

01

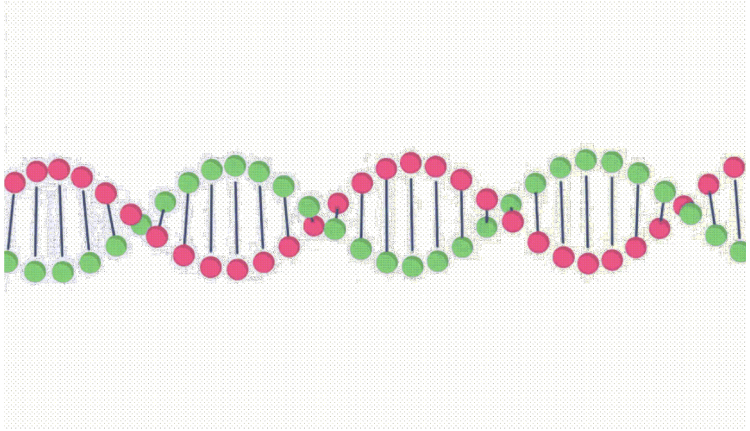
ONE-BIT-FLIP

Randomly change one single bit.

02

THREE-BIT-FLIP

Randomly change three bits.



Evolutionary strategies

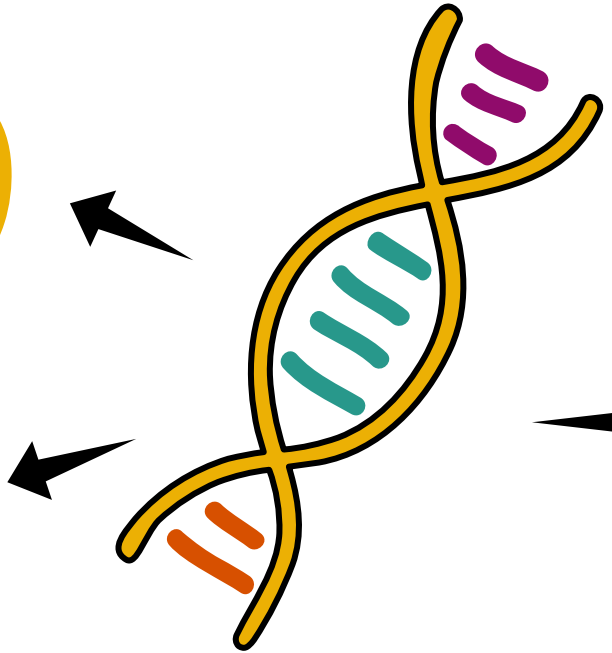
Trick to minimize fitness calls

EARLY STOPPING

If nothing gets better for a while, stop simulation

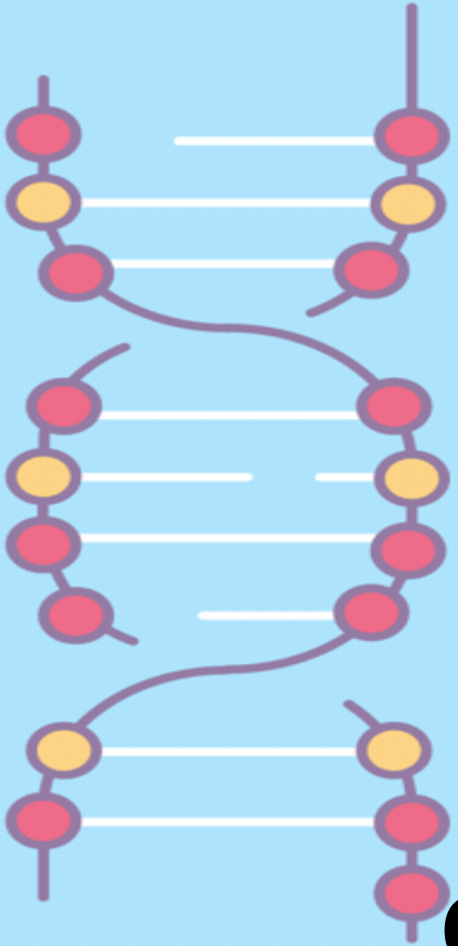
MEMOIZATION

Individuals' fitness is stored and reused



DYNAMIC MUTATION

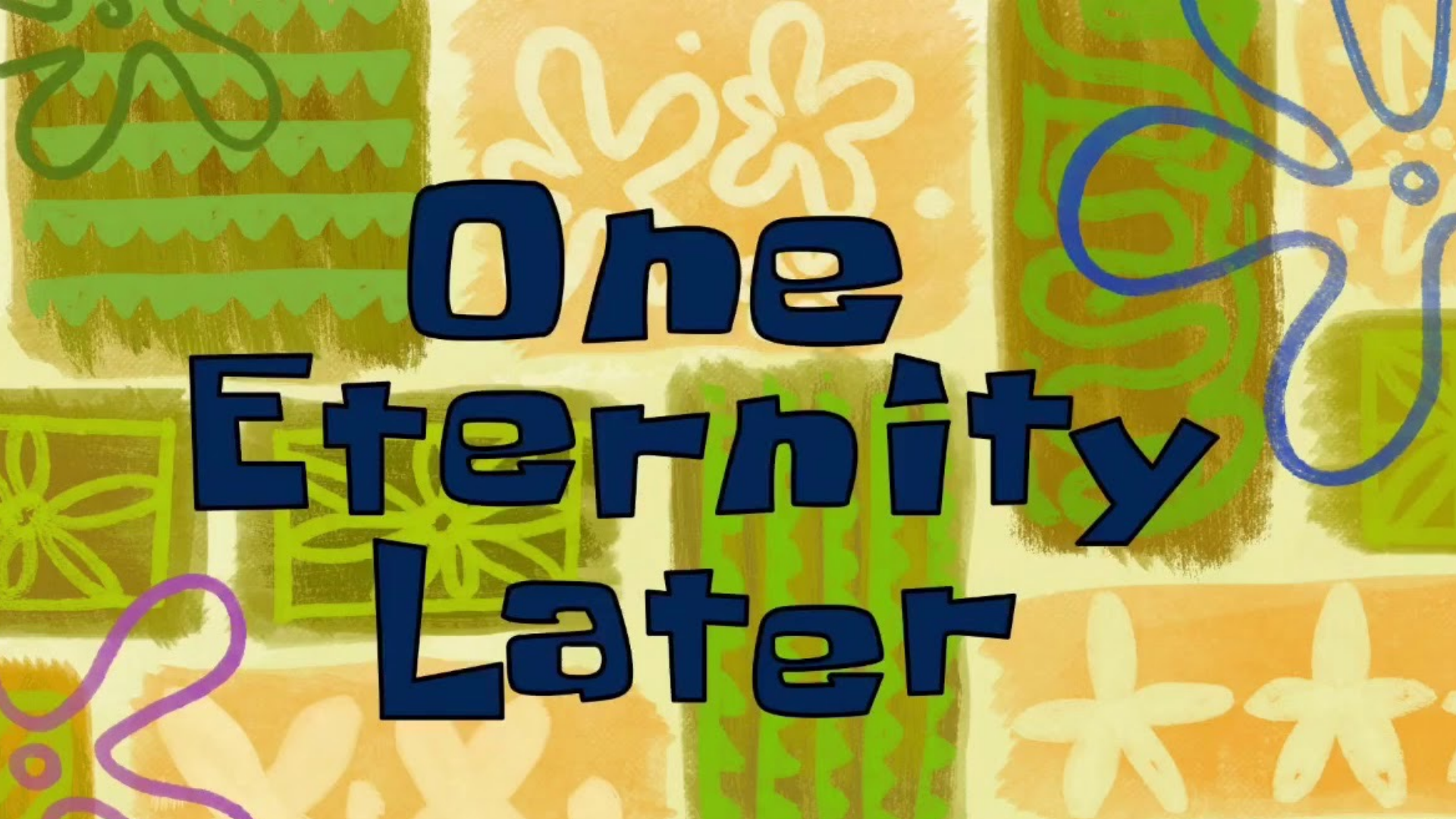
Individuals with similar parents have higher probability to mutate



OUR SOLUTIONS

Baseline

- We try all possible combinations of the methods.
- We evaluate our results with 10 iterations on each experiments.
  We published the mean values.
- Hyperparameter:
  - MU = 15
  - LAMBDA = 30
  - MUTATION_PROB = 0.2
  - DYNAMIC_MUTATION_PROB = True
  - LENGTH_SOLUTION = 1_000
  - NUMBER_GENERATIONS = 3_000
  - COOLDOWN_TIME = 100


One
Eternity
Later

Baseline

				Problem 1		Problem 2		Problem 5		Problem 10	
STRATEGY	PARENT SELECTION	MUTATION	REPRODUCTION	SCORE	CALLS	SCORE	CALLS	SCORE	CALLS	SCORE	CALLS
Comma	Roulette	One single bit	Uniform crossover	0.999	14771	0.512	7038	0.361	9916	0.215	19588
Comma	Roulette	One single bit	One cut	0.995	18574	0.520	6849	0.341	13047	0.208	21881
Comma	Roulette	One single bit	Two cut	0.999	6480	0.788	9218	0.429	7454	0.324	5828
Comma	Roulette	3 bit	Uniform crossover	0.988	15770	0.518	9170	0.451	5024	0.358	7052
Comma	Roulette	3 bit	One cut	0.944	18239	0.519	7003	0.456	6469	0.300	6696
Comma	Roulette	3 bit	Two cut	0.993	8761	0.706	5984	0.467	6986	0.3413	7334
Comma	Torunament	One single bit	Uniform crossover	0.999	15240	0.518	7505	0.282	3182	0.170	3320
Comma	Torunament	One single bit	One cut	0.997	22151	0.515	7271	0.270	3176	0.172	12037
Comma	Torunament	One single bit	Two cut	0.999	8698	0.729	7826	0.386	6092	0.324	7226
Comma	Torunament	3 bit	Uniform crossover	0.991	16358	0.493	8744	0.228	3236	0.209	3218
Comma	Torunament	3 bit	One cut	0.940	18645	0.513	8052	0.276	3318	0.239	3665
Comma	Torunament	3 bit	Two cut	0.992	10705	0.691	9746	0.375	5816	0.371	8486
Plus	Roulette	One single bit	Uniform crossover	1.0	11372	0.561	3471	0.293	12338	0.208	18163
Plus	Roulette	One single bit	One cut	0.999	14080	0.529	5106	0.338	12670	0.188	22267
Plus	Roulette	One single bit	Two cut	1.0	5213	0.843	14273	0.411	15367	0.317	11821
Plus	Roulette	3 bit	Uniform crossover	0.996	16628	0.778	15241	0.543	65352	0.277	67285
Plus	Roulette	3 bit	One cut	0.992	25869	0.572	5758	0.526	74580	0.291	68960
Plus	Roulette	3 bit	Two cut	0.998	8952	0.768	13075	0.449	8303	0.305	4422
Plus	Torunament	One single bit	Uniform crossover	1.0	13495	0.891	33415	0.423	15736	0.263	13852
Plus	Torunament	One single bit	One cut	1.0	18762	0.895	43254	0.266	19080	0.231	21508
Plus	Torunament	One single bit	Two cut	1.0	6260	0.999	23795	0.419	6450	0.302	6599
Plus	Torunament	3 bit	Uniform crossover	0.999	15831	0.799	32837	0.484	46579	0.309	51759
Plus	Torunament	3 bit	One cut	0.999	30925	0.880	52846	0.533	83469	0.266	58839
Plus	Torunament	3 bit	Two cut	0.999	10689	0.995	31199	0.391	7752	0.305	6406

more than 1000 tests

Analysis

- Given that each generation has a low mu (number of parents), as expected, the **Roulette** method seems to work better than the Tournament method.
- While dynamically selecting which offspring to mutate, in general applying the same mutation throughout the genetic algorithm results in the mutation going better on a **single-bit** (more exploitative) than on three-bits (more explorative).
- Generally, **Two-Cuts reproduction** gave better results in all problems.
- Generally, the **'Plus'** strategy achieves higher fitness values, while 'Comma' achieves convergence in fewer fitness calls.

We will use this **highlighted** configuration for the next experiments

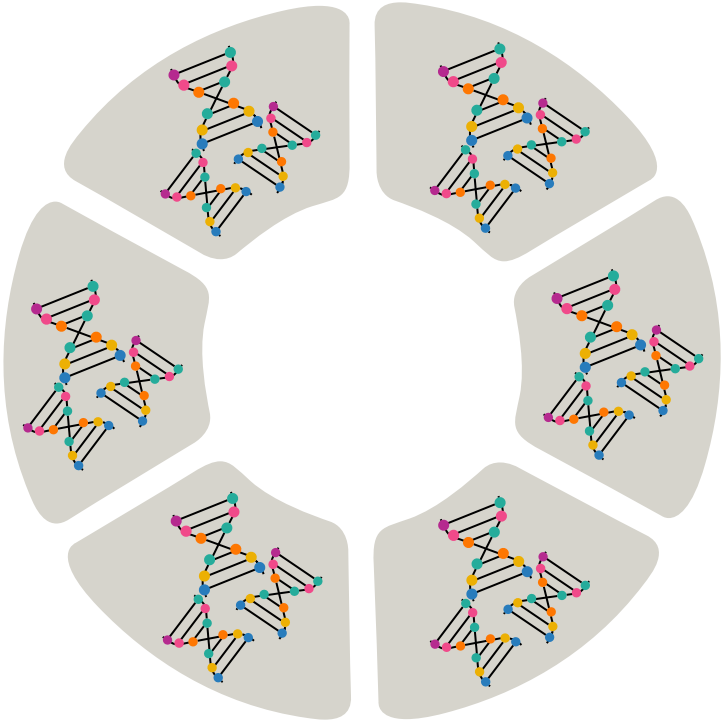
Baseline – Best results

Problem 1		Problem 2	
Score	Calls	Score	Calls
1.0	5’213	0.999	23’795
Plus, Roulette,1 single-bit, Two-cuts		Plus, Tournament, 1 single bit, Two-cuts	

Problem 5		Problem 10	
Score	Calls	Score	Calls
0.543	65’352	0.371	8’486
Plus, Roulette, 3-bits, Uniform		Comma, Tournament, 3-bits, Two-cuts	

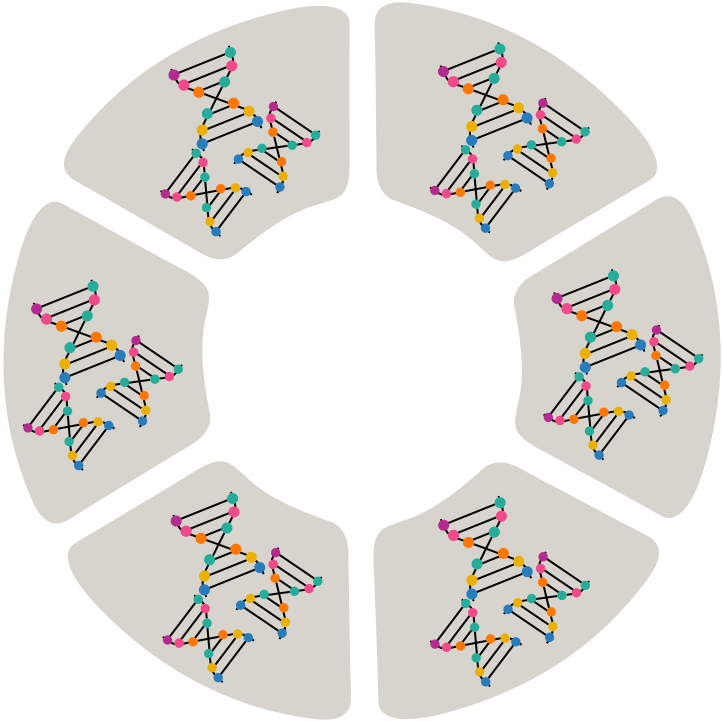
Islands

- Islands Model refers to **partitioning** the population **into subgroups (islands)** that evolve separately.
- Periodic migration or exchange of individuals between islands fosters **diversity** and **information exchange.**



Islands

- What is the process for migrating individuals?
  - Radom migration
  - Ring topology migration
  - Fitness based
- At what point should individuals be migrated?
  - Fixed generation
  - Fitness based



Islands

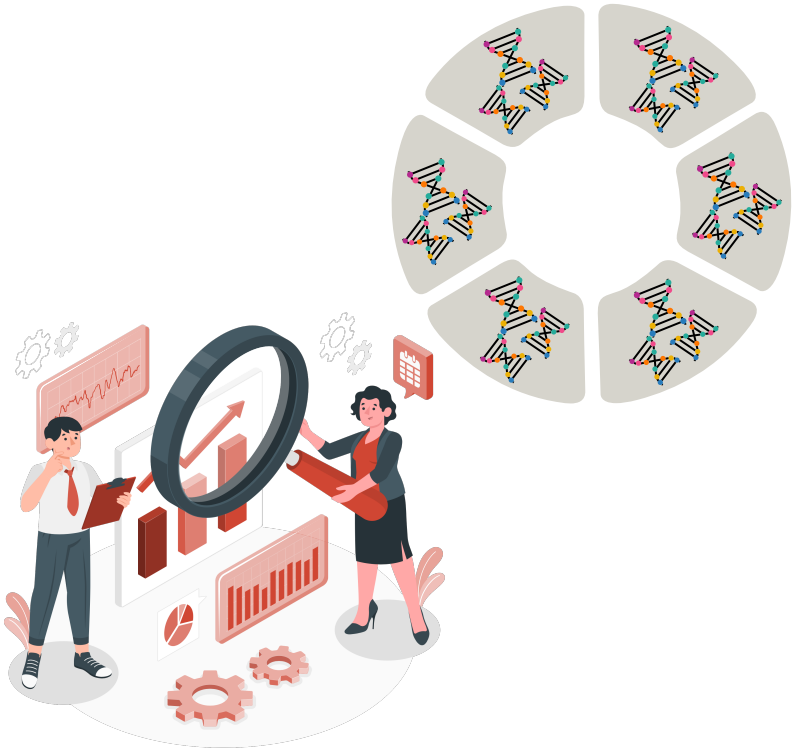
			Problem 1		Problem 2		Problem 5		Problem 10	
ISLAND STRATEGY	WHEN TO SWAP	MIGRATION	SCORE	CALLS	SCORE	CALLS	SCORE	CALLS	SCORE	CALLS
Base	Fixed Generation	Random Swap	1.0	23'596.26	0.992	87'507.43	0.345	37'537.77	0.340	34'720.7
Base	Fixed Generation	Ring Topology	1.0	31'325.4	0.9758	154'722.03	0.480	42'044.23	0.333	36'512.1
Base	Fixed Generation	Best Fitness	1.0	33'409.56	0.959	160'831.06	0.476	34'824.8	0.317	32070.7
Base	Fitness Based	Random Swap	1.0	22'255.46	0.985	99'106.9	0.493	38'394.96	0.356	42'583.8
Base	Fitness Based	Ring Topology	1.0	22'894.96	0.967	162'121.8	0.48539	36'711.3	0.351	45'035.7
Base	Fitness Based	Best Fitness	1.0	22'840.26	0.862	115'951.26	0.507	33'822.43	0.353	39'653.3

Analysis

- Island methods ask for more fitness evaluations, but yield better results in general
- Island method is more robust and results have less variance between runs
- **Fitness based** outperforms fixed generation
- There is not much difference between migration strategies. We choose **Random Swap** for the next experiment

We will use this **highlighted** configuration for the next experiments

Islands: results

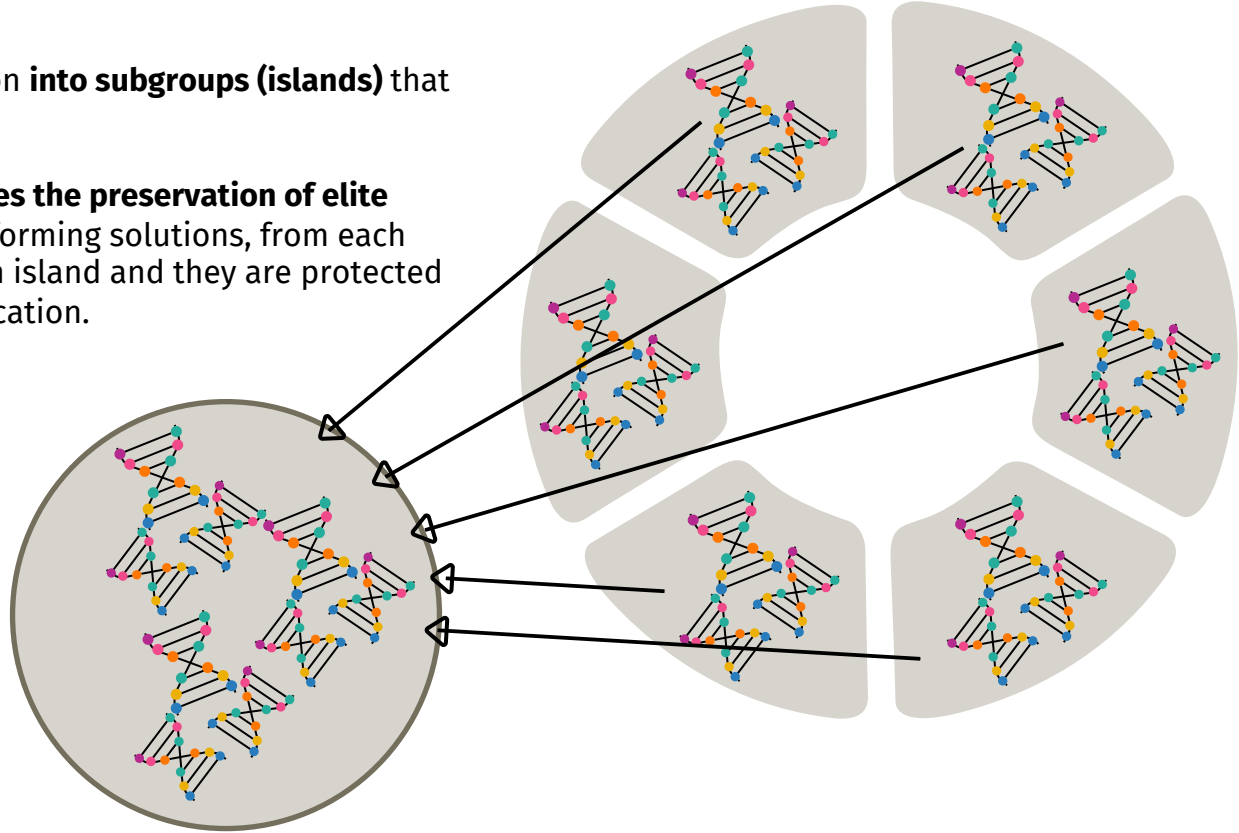


Problem 1		Problem 2	
Score	Calls	Score	Calls
1.0	22’225	0.992	87’507
Random Swap, Fitness based		Random Swap, Fixed Generation	

Problem 5		Problem 10	
Score	Calls	Score	Calls
0.507	33’822	0.356	42’583
Best Fitness, Fitness based		Random Swap, Fitness based	

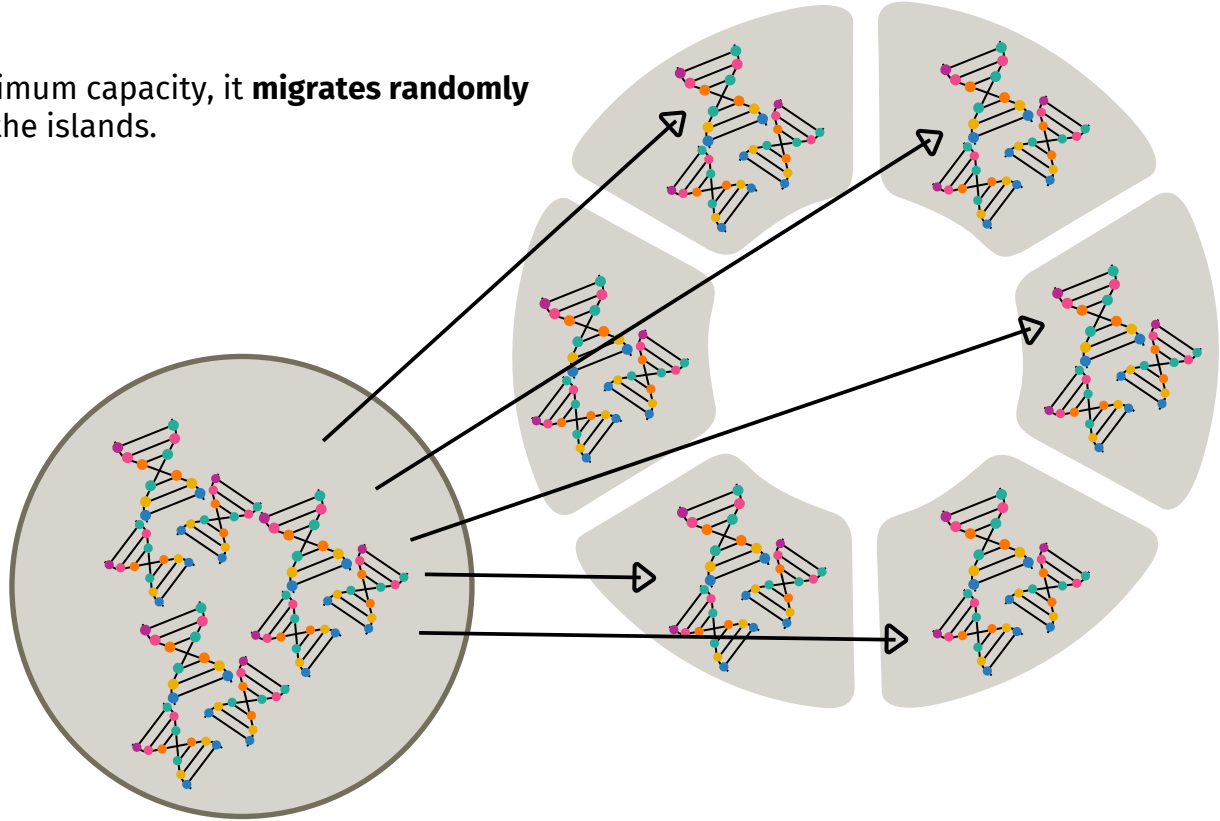
Valhalla + Islands

- **Partition** of the population **into subgroups (islands)** that evolve separately.
- Valhalla method **prioritizes the preservation of elite individuals**, the best-performing solutions, from each generation and from each island and they are protected from alteration or modification.

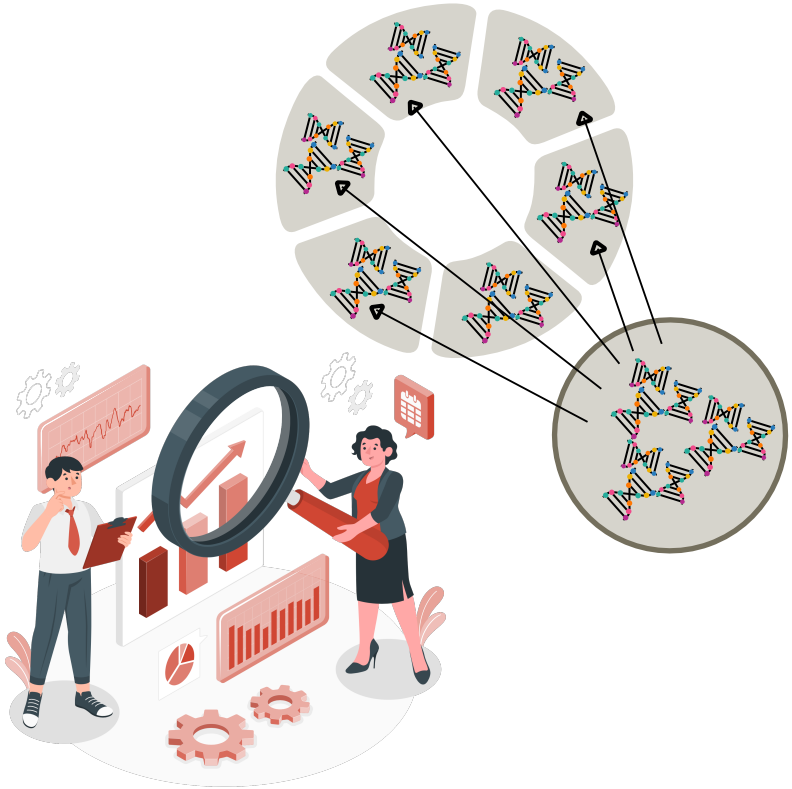


Valhalla + Islands

- When Valhalla exceed maximum capacity, it **migrates randomly some of his individuals** in the islands.



Valhalla + Islands: results



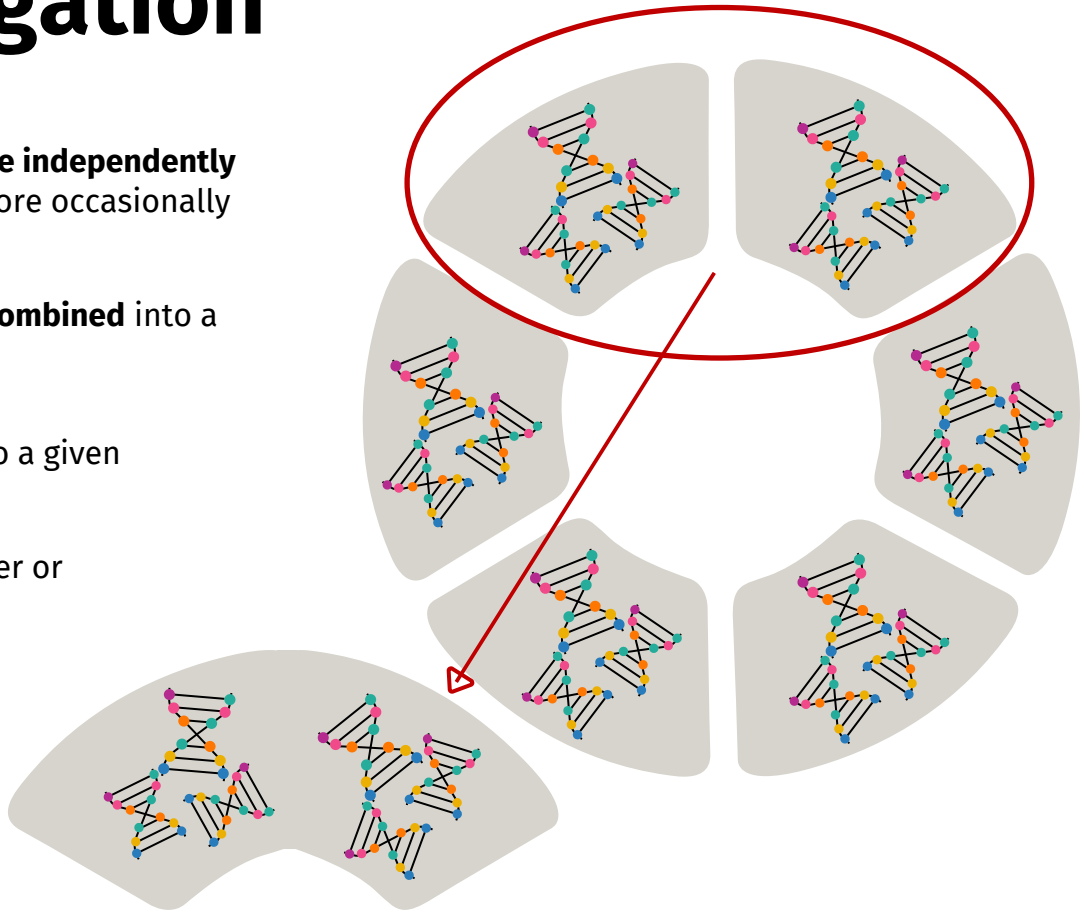
Problem 1		Problem 2	
Score	Calls	Score	Calls
1.0	27’052	0.997	107’739

Problem 5		Problem 10	
Score	Calls	Score	Calls
0.495	30’052	0.345	34’369

Islands Segregation

- Multiple **subpopulations (islands) evolve independently** for a certain number of generations before occasionally exchanging information.
- **The islands most similar in fitness are combined** into a single island.

  When?
  - Fitness changes are less or equal to a given threshold.
  - The number of generation is greater or equal a given threshold.



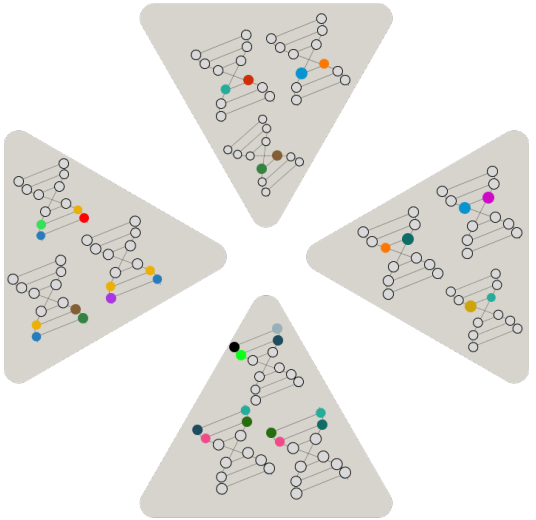
Islands Segregation: results

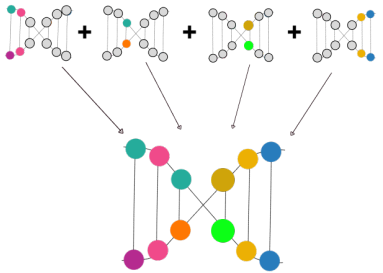
Problem 1		Problem 2	
Score	Calls	Score	Calls
1.0	22’941	0.989	100'672

Problem 5		Problem 10	
Score	Calls	Score	Calls
0.527	65’083	0.375	62’032

Expert Islands

- Each island is formed by individuals of smaller size than the actual solution needed
- The islands do not communicate in this scenario
- The island optimizes individuals that are used as subpart of the solution
- Each individual has size SOLUTION_SIZE/NUM_ISLANDS
- The solution is represented by the concatenation of the best agents of each island





Expert Islands: results

Problem 1		Problem 2	
Score	Calls	Score	Calls
0.612	17’592	0.589	33’882

Problem 5		Problem 10	
Score	Calls	Score	Calls
0.453	36’734	0.326	28’360

Hyperparameters Optimization

Some hyperparameter tuning combinations:

MU = 15 LAMBDA = 30 COOLDOWN_TIME = 20 NUM_ISLANDS = 50

			Problem 1		Problem 2		Problem 5		Problem 10	
ISLAND STRATEGY	WHEN TO SWAP	MIGRATION	SCORE	CALLS	SCORE	CALLS	SCORE	CALLS	SCORE	CALLS
Expert islands	-	-	0.87	34896.86	0.833	34786.0	0.74	34938.93	0.775	34856.0

MU = 5 LAMBDA = 15 COOLDOWN_TIME = 20 NUM_ISLANDS = 100

			Problem 1		Problem 2		Problem 5		Problem 10	
ISLAND STRATEGY	WHEN TO SWAP	MIGRATION	SCORE	CALLS	SCORE	CALLS	SCORE	CALLS	SCORE	CALLS
Expert islands	-	-	1.0	16732.73	1.0	19462.13	1.0	18400.26	0.996	16953.56

MU = 3 LAMBDA = 10 COOLDOWN_TIME = 5 NUM_ISLANDS = 250

			Problem 1		Problem 2		Problem 5		Problem 10	
ISLAND STRATEGY	WHEN TO SWAP	MIGRATION	SCORE	CALLS	SCORE	CALLS	SCORE	CALLS	SCORE	CALLS
Expert islands	-	-	1.0	2490.06	1.0	2564.86	1.0	2536.93	1.0	2514.9

Expert Islands: results

MU = 3 - LAMBDA = 10 - COOLDOWN_TIME = 5 - NUM_ISLANDS = 250

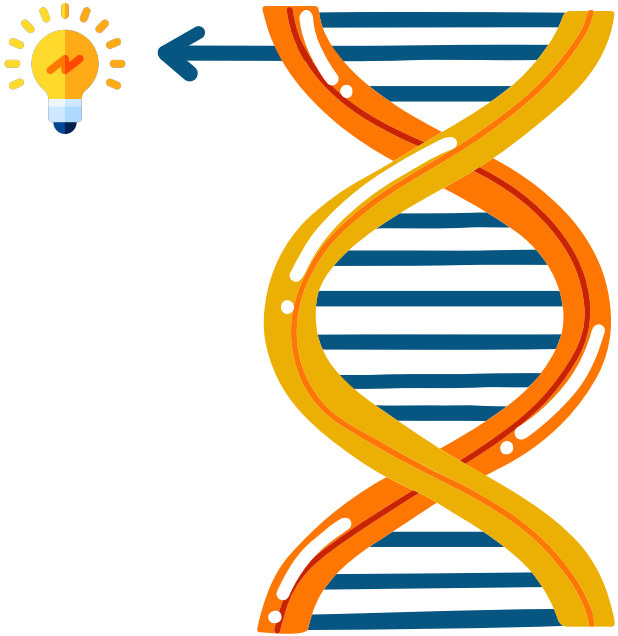
Problem 1		Problem 2	
Score	Calls	Score	Calls
1.0	**2’490**	**1.0**	**2564**

Problem 5		Problem 10	
Score	Calls	Score	Calls
1.0	**2’536**	**1.0**	**2’514**

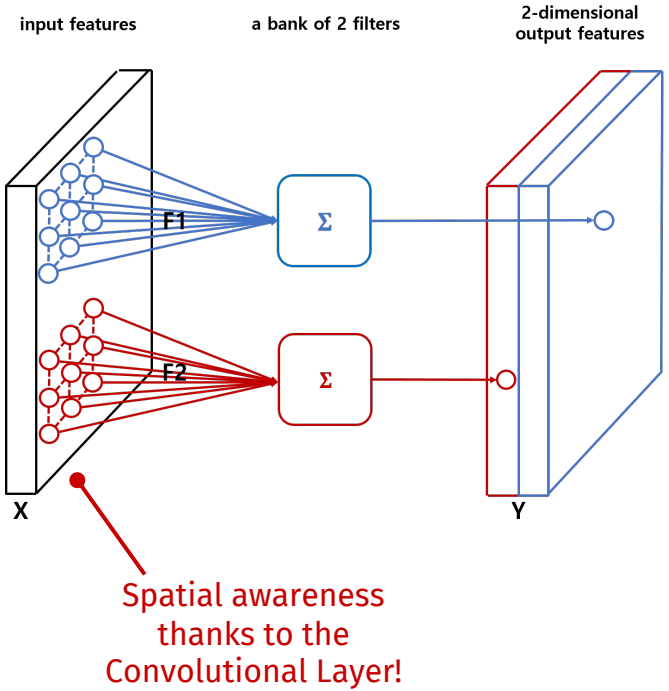
Neural Network to evaluate fitness

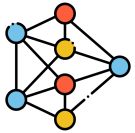
IDEA

To **minimize the number of calls to the fitness function**, it might be useful to find a method of proxying the function!



Neural Network to evaluate fitness





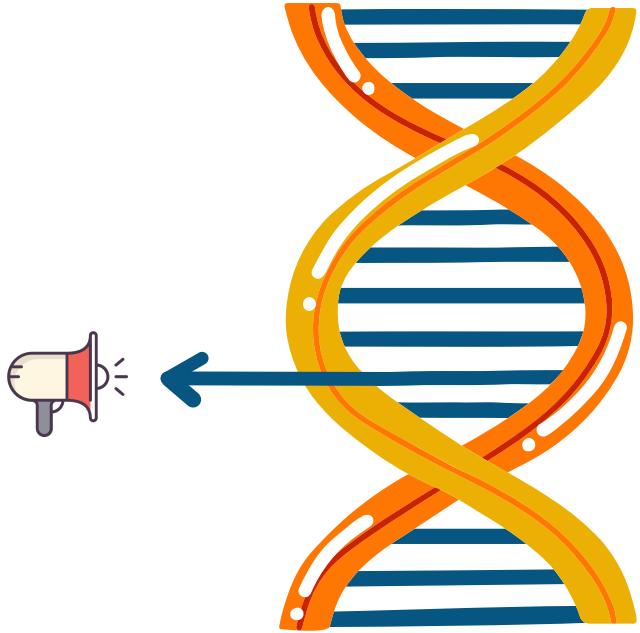
STRUCTURE

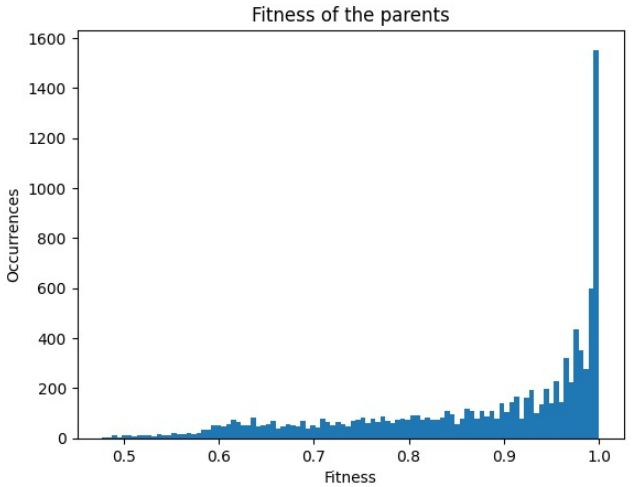
Various neural network configurations were tested. The one that gave the most satisfactory results was the one consisting only of a simple **convolutional layer + fully connected layer**.

Neural Network to evaluate fitness

PROBLEMS

During the genetic algorithm, the **distribution** of data to train our model is strongly **unbalanced**. Furthermore, it is difficult for the model to evaluate genes that have a **higher fitness** than that obtained during the training phase.





What's next

Test on new problems

Test the developed benchmark on new problems, to see if it is indeed applicable to other similar problems.

Extend the Benchmark

Due to time constraints, it was not possible to implement many other methods, parent selections, reproductions, or mutations. It might be interesting to extend the benchmark!

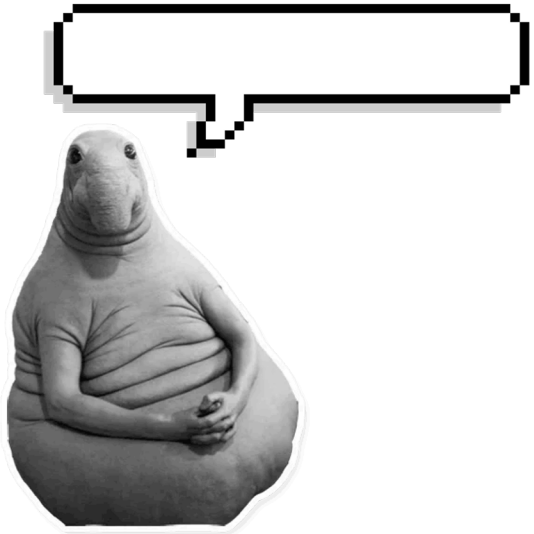
Formalize the Benchmark

Formalize the functions and methods used, with a more explanatory README on how each part of the code works, to turn the resulting benchmark into a library open to all!

Improve NN in the loop

Train the neural network several times during the genetic algorithm, once the imbalance problem has been solved, avoiding most of the exploitation phase calls.

Thank you!



SCAN ME

Luca Catalano – S308658
Claudio Savelli – S317680
Florentin-Cristian Udrea – S319029