Genetic Algorithm – Team 5

Introduction:

The Genetic Algorithm, a biologically inspired algorithm is an optimization meta-heuristic algorithm, is inspired by Charles Darwin’s theory of natural evolution. Genetic algorithms came from the research of John Holland, at the University of Michigan, in 1960 but were not popular until the ‘90s. Their main purpose is to be used to solve problems where deterministic algorithms are too costly. These algorithms are "**bio-inspired**" because they mimic living creatures' fitting in their environment for survival [1].

This algorithm reflects the process of **natural selection** where the fittest individuals are selected for reproduction to produce offspring of the next generation [2]. The solutions of this algorithm mimic a population that has evolved through several generations where each organism in the population is defined by a chromosome (or, sometimes, a pair of chromosomes), which encode information determining an organism’s behavior (genetics), and in turn how well an organism can adapt to its environment, survive, and generate offspring (natural selection). This information is the single solution to the problem that is optimized, also probing different areas of the cost function’s landscape at the same time [3].

The core idea of the Genetic Algorithm is to encode a solution as a chromosome and assign each chromosome to an organism. Then derive how fit the corresponding organism is for its environment, based on how good a solution is. Fitness, in turn, is a proxy for the chances of an organism to reproduce and pass its genetic material to the next generation through its offspring. Like in nature, the alpha-individuals of a population, the stronger (or faster, or yet smarter) ones, tend to have a higher chance of surviving longer and finding a mate. The resulting cross-product after mutation is declared to be the best solution for the problem that is being optimized [3]. In the industry, genetic algorithms are used when traditional ways are not efficient enough.

[1] History of Genetic Algorithms from <https://www.codingame.com/playgrounds/334/genetic-algorithms/history>

[2] Introduction to Genetic Algorithms — Including Example Code by Vijini Mallawaarchchi from <https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3>

[3] Genetic algorithms: Biologically inspired, fast-converging optimization by Marcello La Rocca from <https://freecontent.manning.com/genetic-algorithms-biologically-inspired-fast-converging-optimization/>

<https://www.generativedesign.org/02-deeper-dive/02-04_genetic-algorithms/02-04-01_what-is-a-genetic-algorithm>

Algorithm description: To get a fast, clear, simple optimization algorithm, we will implement a genetic algorithm that can find approximated, near-optimal solutions to the knapsack problem.

To completely define an optimization algorithm, we need to specify the following components:

* How to encode a solution-Chromosomes, for GA
* Transitional Operators-Crossover and mutations
* A way to measure cost-Fitness function
* How system evolves-Generations and natural selection

Chromosomes: Chromosomes encode a solution to the problem we need to solve. The 0-1 knapsack is the perfect example to discuss the original genetic algorithm, because a solution to this problem can be encoded exactly as a binary string.

For each item, a zero means we leave it on Earth, while a 1 means we add it to the crate going to Mars.

Two different solutions (or phenotypes, in biological terms) can be represented by two chromosomes (or genotypes). Their difference boils down to the difference between the two bit-strings.

It’s worth noting that for this example we have a 1:1 mapping between genotype and phenotype: a solution (the phenotype) is uniquely identified by its representation (the genotype) and so two organisms with the same genotype will translate to the same solution and in turn to the same fitness value.

In the context of our 0-1 knapsack example, if two organisms have the same genotype, both solutions will add the same set of goods added to the crate to Mars.

We also need to make one more thing clear: the actual representation of chromosomes depends on the actual, specific problem we are solving. In this case, on the specific instance of 0-1 knapsack (although a generic representation for all 0-1 knapsack can and should be designed).

Demonstration:

***Demonstration***

Genetic algorithms are adaptive heuristic search algorithms, they are a part of evolutionary computing, which rapidly growing area of artificial intelligence & soft computing.

Basically, the algorithm is inspired by Darwin’s theory about evolution. It is an intelligent random search technique. It is more suitable for optimization problems.

Genetic Algorithms are most used in **optimization problems and search problems** based on random heuristics. The idea consists of a simulation of natural selection. The initial population will evolve through (1) emergent variations derived from the crossover between fittest and (2) through mutations

Below is the example demonstrating how the genetic algorithm method works

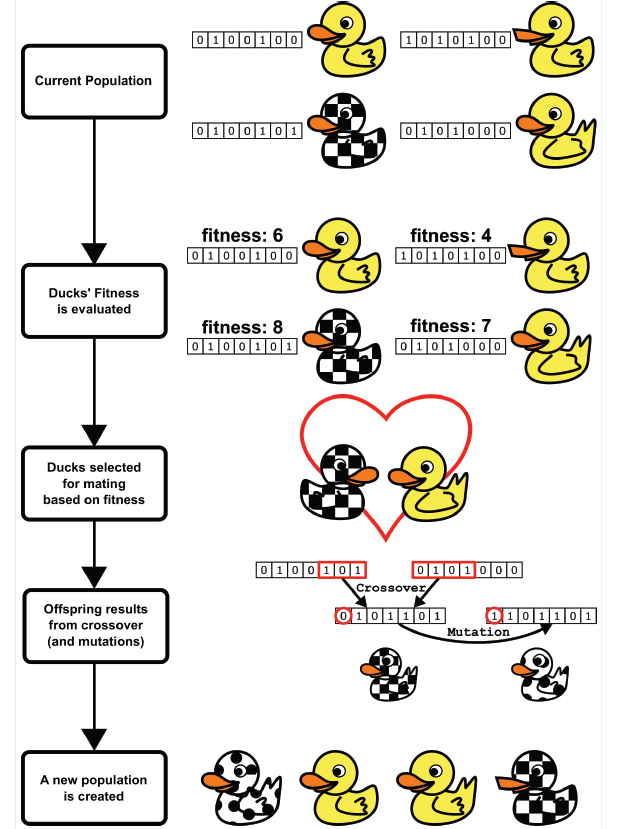


Fig ….... Natural selection in genetic algorithms, in an analogy with the biological process

The Natural selection in genetic algorithm involves 5 important steps:

1. Current population
2. Fitness evaluation
3. Selection based on fitness
4. Results formed from crossover and mutations
5. New population

***1. Current population:***

In the above example, the population of ducks shows a few unique features, from the shape of their beak to their color/pattern, to the shape of wings and tails. If you look closely, you should be able to spot them and to figure out how these changes are encoded in their chromosomes.

As discussed, each feature in individuals’ phenotype is determined by a difference in their genotype, a1 that becomes a0, or vice versa (or maybe a few bits that need to flip together, or even, if we move past binary strings, a gene assigned with a different real number).

***2.Fitness evaluation:***

To perform selection, each duck ‘s fitness needs to be evaluated. This can be as easy as feeding each chromosome to a function or as complicated as running a simulation for hours (as in the predator-prey setting described) and seeing which individuals survived longer or performed a real-world task better.

***3.Selection based on fitness:***

Organisms with higher fitness will have greater chances of being chosen to pass their genes to the next generation. This is most crucial part, because without this “meritocracy,” no optimization would be performed.

***4. Results formed from crossover and mutations:***

Once the two individuals are selected, genetic material of two ducks are recombined. At this stage crossover and mutations happen in their own section, in figure …., little duckling takes its feathering pattern and color from one parent and its tail shape from the other. Then a further mutation kicks in changing the feathering again from a checkerboard pattern to a polka dots pattern.

***5. New population:***

The new population, formed repeating this process many times, should show a larger presence of those genes/features associated with high-fitness individuals in the initial population.

***Advantages and Disadvantages of Genetic Algorithm***

Advantages:

1. Faster than other algorithms.
2. Easier. If vector representation of individual is right, solution could be found without a deep analysis work

Disadvantages:

1. The random heuristics sometimes don’t find the optimum.
2. It is not a complete algorithm. Sometimes it can get stuck with a local maximum problem. Nevertheless, crossover operation helps to mitigate it, although this implies more iterations.

**Conclusion:**

The demonstration above provides an overview of the implementation of the Genetic Algorithm to solve the 0-1 knapsack problem. We have seen the components involved to define an optimization algorithm, explanation on Chromosomes, Natural Selection, and Fitness, and how it is implemented to achieve the best solution.

From the results, we can conclude that the best possible solution for the knapsack problem is to pack wheat flour, rice, and beans that maximize the total amount of calories. In the case of a larger dataset where the problem is to pack hundreds of possible items, we can use the implementation of the genetic algorithm combined with Graph Theory to implement the methods for crossover, mutations, and random initialization that we discussed in this case study.

Some practical applications of Genetic algorithms include (1) designing composite materials and aerodynamic shapes for [race cars](http://www.sae.org/technical/papers/2003-01-1327) that can return combinations of best materials and best engineering to provide faster, lighter, more fuel-efficient, and safer vehicles (2) designing multi-purpose robots (3) developing dynamic and anticipatory routing of circuits for [telecommunications networks](https://en.wikipedia.org/wiki/Telecommunications_network) (4) TSP can be used to plan the most efficient routes and schedules for travel planners, traffic routers and even shipping companies (5) GA-assisted computer modeling of [finance and investment strategies](http://www.wiley.com/WileyCDA/WileyTitle/productCd-0471576794.html).