Introduction to Genetic Algorithm

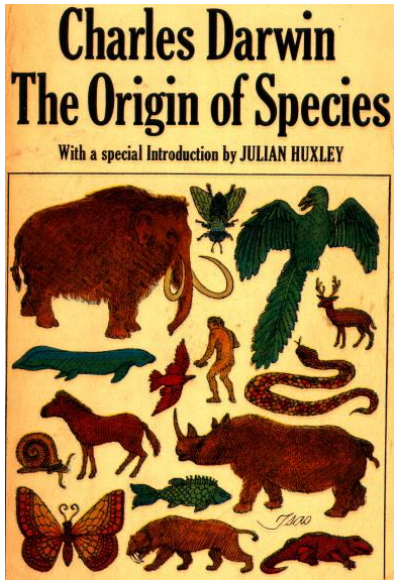
General Introduction to GAs

- Genetic algorithms (GAs) are a technique to

  solve problems which need optimization.

- GAs are a subclass of **Evolutionary Computing** and are random search algorithms.

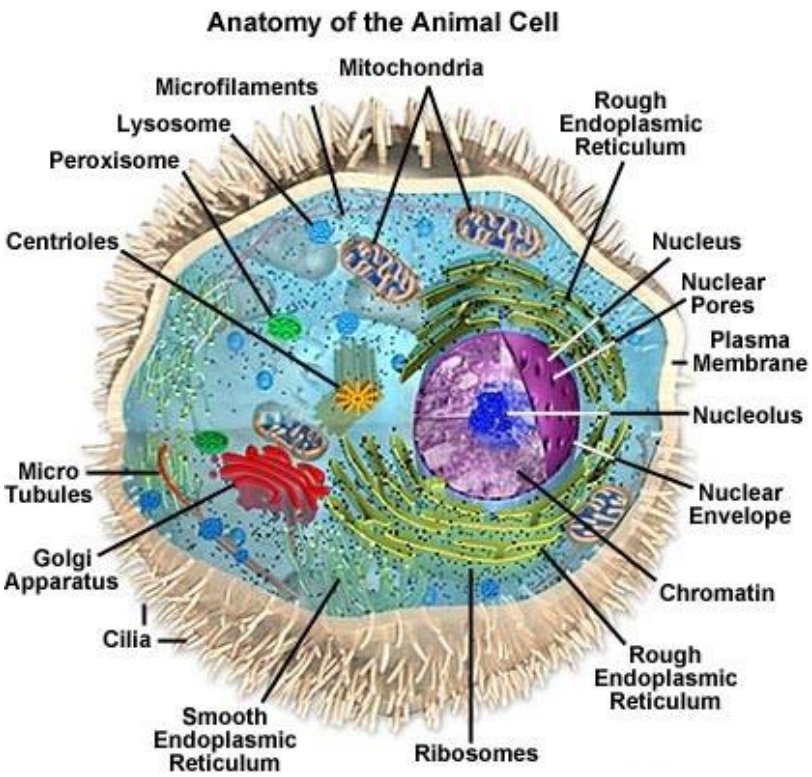
- GAs are based on
  Darwin's theory of evolution.

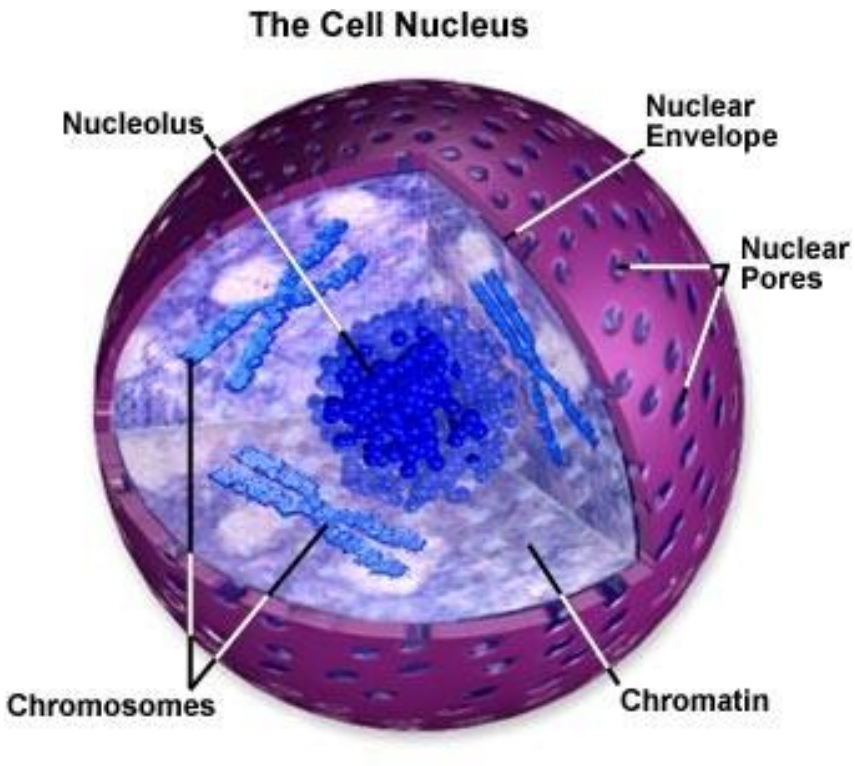


- History of GAs:
  - Evolutionary computing evolved in the 1960s.
  - GAs were created by John Holland in the mid-1970s.

Biological Background (1) – The Cell

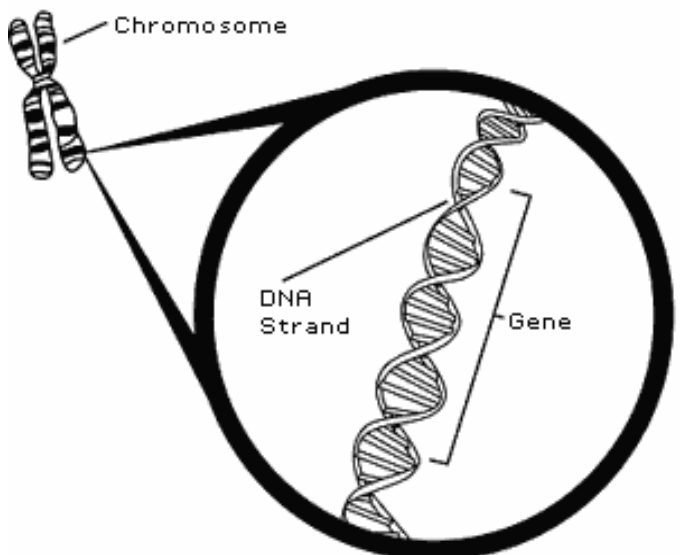
- The center of this all is the **cell nucleus.**
- The nucleus contains the genetic information.





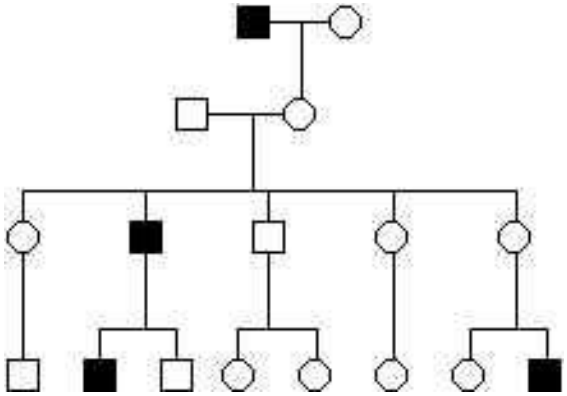
Biological Background (2) – Chromosomes

- Genetic information is stored in the **chromosomes.**
- Each chromosome is build of **DNA (**Deoxyribonucleic acid**).**
- Chromosomes in humans form pairs.
- There are 23 pairs.
- The chromosome is divided in parts: **genes.**
- Genes code for properties.
- The posibilities of the genes for one property is called: **allele.**
- Every gene has an unique position on the chromosome: **locus.**



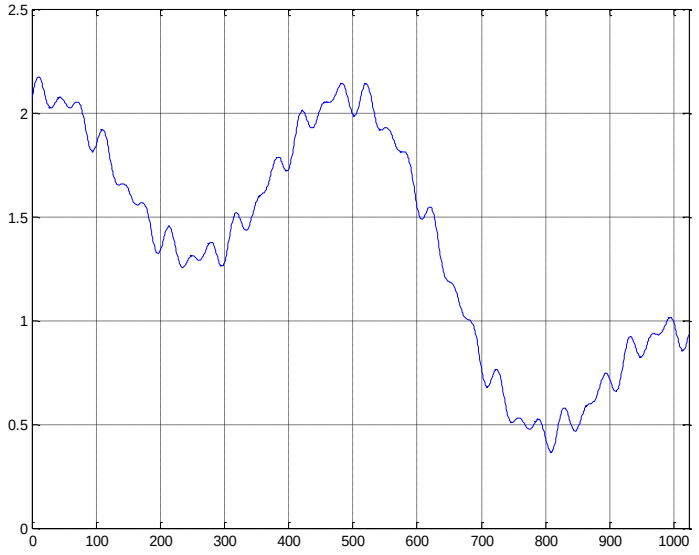
Biological Background (3) – Genetics

- The entire combination of genes is called **genotype.**
- A genotype develops into a **phenotype.**
- **Alleles** can be either dominant or recessive.
- Dominant alleles will always express from the genotype to the fenotype.
- Recessive alleles can survive in the population for many generations without being expressed.



Genetic Algorithm (1) – Search Space

- Most often one is looking for the best solution in a specific subset of solutions.
- This subset is called the **search space** (or state space).
- Every point in the search space is a possible solution.
- Therefore every point has a **fitness** value, depending on the problem definition.
- GAs are used to search the search space for the best solution, e.g. a minimum.
- Difficulties are the local minima and the starting point of the search.



Genetic Algorithm (2) – Basic Algorithm

- Starting with a subset of ***n*** randomly chosen solutions from the search space (i.e. chromosomes). This is the **population.**

- This population is used to produce a next **generation** of individuals by reproduction.

- Individuals with a higher **fitness** have more chance to reproduce (i.e. natural selection).

Comparison of Natural and GA Terminology

Natural	Genetic Algorithm
Chromosome	String
Gene	Feature or character
Allele	Feature value
Locus	String position
Genotype	Structure
Phenotype	Parameter set, a decoded structure

Genetic Algorithm (3) – Basic Algorithm

- Outline of the basic algorithm

0 START : Create random population of **n** chromosomes

1 FITNESS : Evaluate fitness **f(x)** of each chromosome in the population

2 NEW POPULATION

1 REPRODUCTION/SELECTION **:** Based on **f(x)**

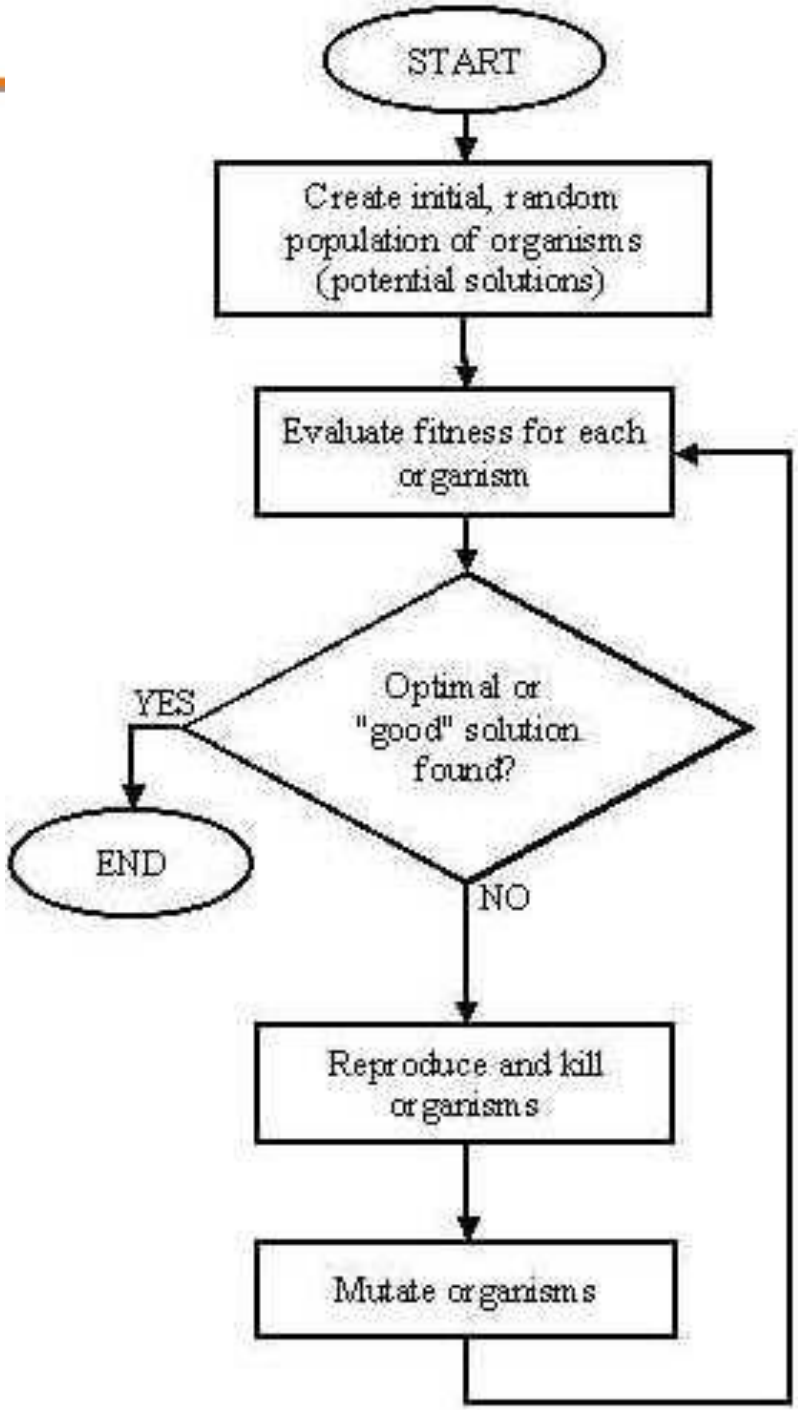
2 CROSS OVER : Cross-over chromosomes

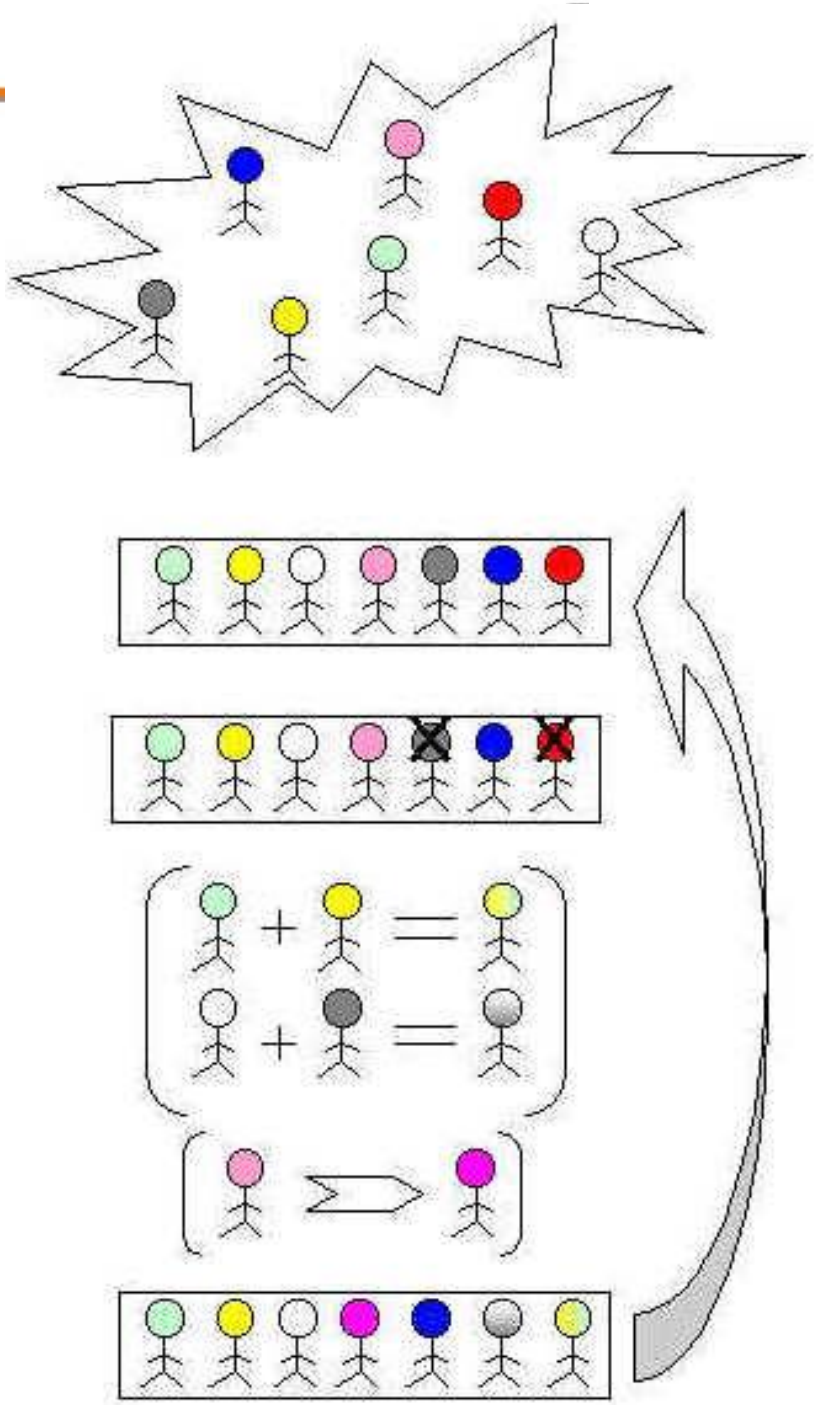
3 MUTATION **:** Mutate chromosomes

3 REPLACE : Replace old with new population: the new generation

4 TEST **:** Test problem criterium

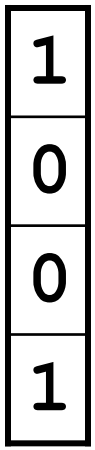
5 LOOP **:** Continue step 1 – 4 untill criterium is satisfied


START
Create initial, random population of organisms (potential solutions)
Evaluate fitness for each organism
Optimal or "good" solution found?
YES
END
NO
Reproduce and kill organisms
Mutate organisms



Genetic Algorithm – Coding

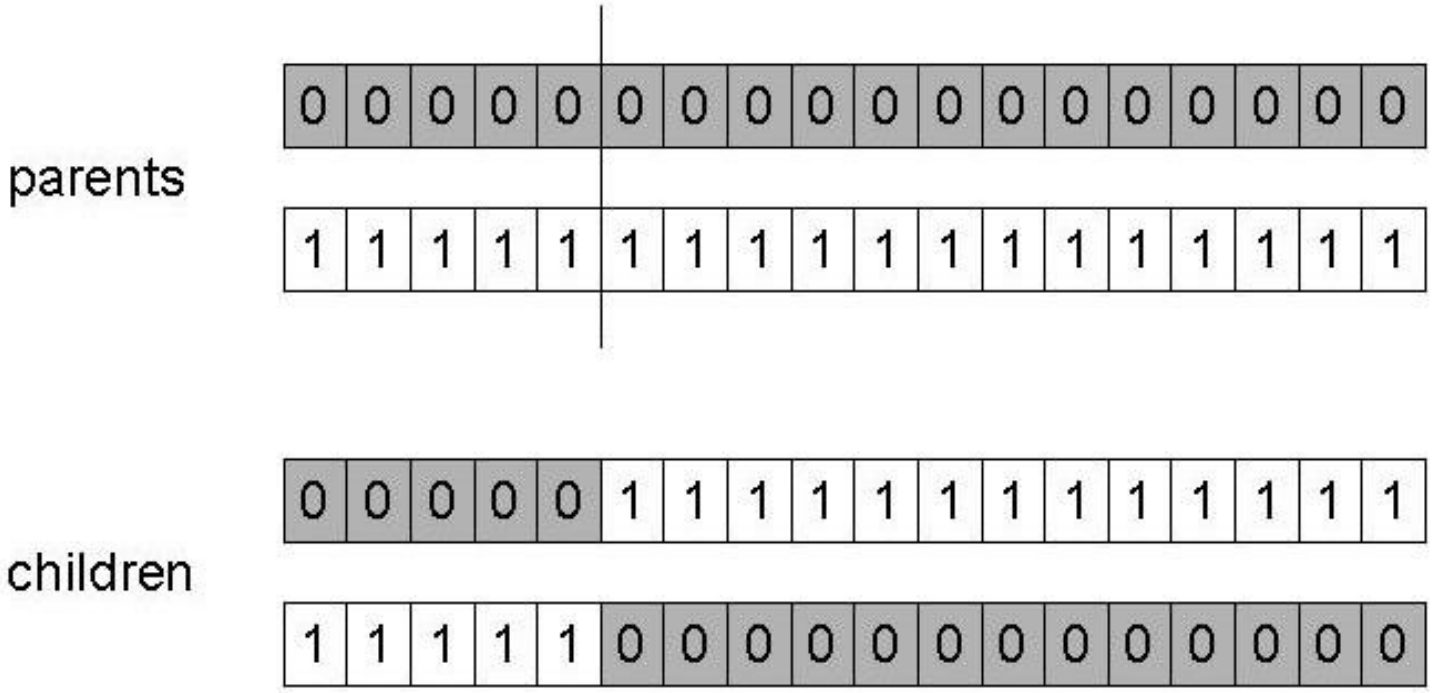
- Chromosomes are encoded by **bitstrings.**
- Every bitstring therefore is a solution but not necessarily the best solution.
- The way bitstrings can code differs from problem to problem.



Either sequence of on/off **or** the number 9

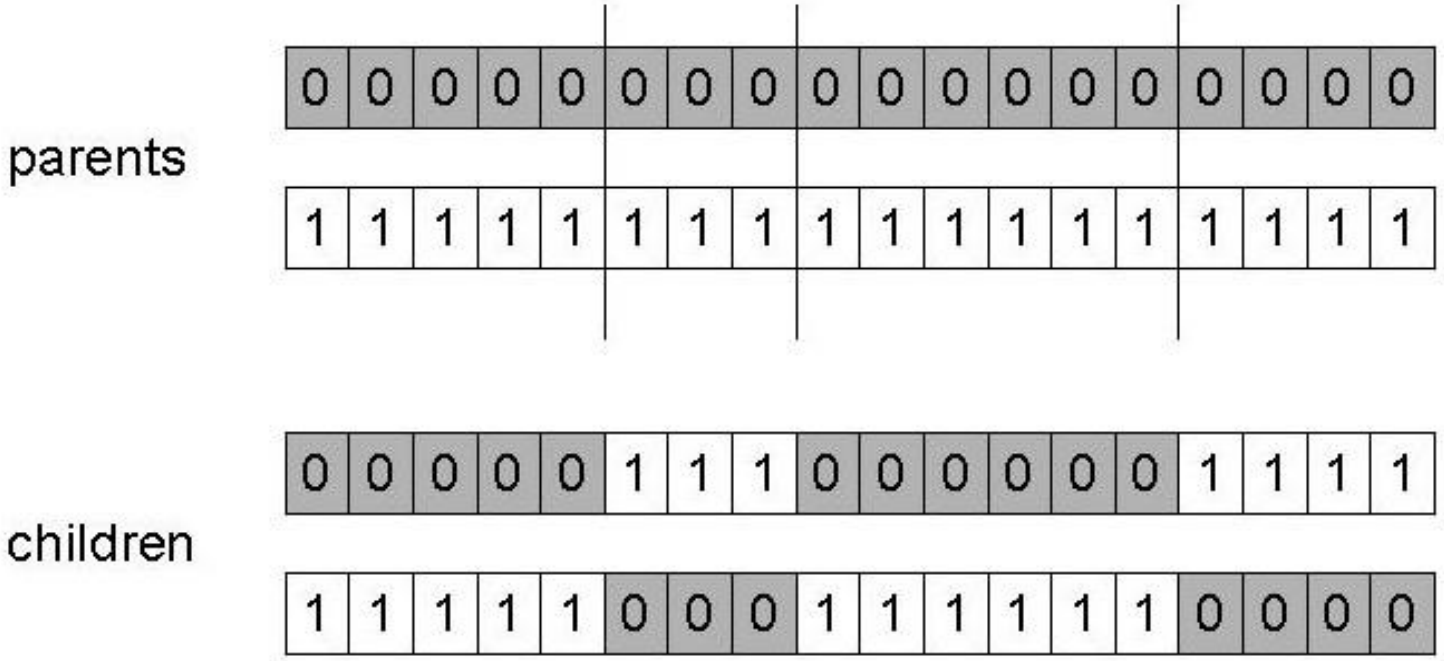
Genetic Algorithm – Crossover (Single Point)

- Choose a random point on the two parents.
- Split parents at this crossover point.
- Create children by exchanging tails.



Genetic Algorithm – Crossover (n Points)

- Choose n random crossover points.
- Split along those points.
- Glue parts, alternating between parents.
- Generalization of 1 point.



Genetic Algorithm – Uniform Crossover

Generate uniformly random number.

X1 = 0 1 1 0 0 0 1 0 1 0

X2 = 1 1 0 0 0 0 0 1 1 1

Uniformly generated = 1 0 0 0 0 0 1 0 0 0

As a result, the new population becomes,

X1 = 1 1 1 0 0 0 0 0 1 0

X2 = 0 1 0 0 0 0 1 1 1 1

Mutation

- Mutation is the addition of new characteristics into the population. With mutation, we randomly try to inject newer characteristics into the population. If these characteristics are good, they will survive and continue into the next population.

- Mutation is carried out on a very few individuals, because these are random characteristics that can drive the population anywhere.

- The amount of mutation to be carried out is specified by the mutation rate *rm, a number between* 0 and 1. This number is usually kept low to keep a limited mutation in the algorithm. We iteratethrough each entity in a solution in the population.

Uniform Mutation

- Uniform mutation is the most common mutation operation. In this operation, the value at the selected location is simply flipped. Thus if a 0 is present in the location, it changes to 1, and vice versa.

Gaussian Mutation

- Gaussian mutation happens in the case of the phenotype representation of the solution, where the solution is represented by simple numerals. In this type of mutation, we add a randomly generated number to the existing value at the selected location.

Genetic Algorithm – An Example

- Simple problem: max x^2 over {0, 1, …, 31}
- GA approach:
  - Representation: binary code, e.g. 01101 ↔ 13
  - Population size: 5
  - 1-point xover,
  - bitwise mutation
- One generational cycle performed manually is shown here.

Example : Selection

String no.	Initial population	x Value	Fitness $f(x) = x^2$
1	0 1 1 0 1	13	169
2	1 1 0 0 0	24	576
3	0 1 0 0 0	8	64
4	1 0 0 1 1	19	361

Example : Crossover

String no.	Mating pool	Crossover point	Offspring after xover	x Value	Fitness $f(x) = x^2$
1	0 1 1 0 | 1	4	0 1 1 0 0	12	144
2	1 1 0 0 | 0	4	1 1 0 0 1	25	625
2	1 1 | 0 0 0	2	1 1 0 1 1	27	729
4	1 0 | 0 1 1	2	1 0 0 0 0	16	256
Sum					1754
Average					439
Max					729

Example : Mutation

String no.	Offspring after xover	Offspring after mutation	x Value	Fitness $f(x) = x^2$
1	0 1 1 0 0	1 1 1 0 0	26	676
2	1 1 0 0 1	1 1 0 0 1	25	625
2	1 1 0 1 1	1 1 0 1 1	27	729
4	1 0 0 0 0	1 0 1 0 0	18	324
Sum				2354
Average				588.5
Max				729

Comparison of GA with Traditional Optimization Techniques

- GA works with the coding of solution set and not with the solution itself.
- GA uses population of solutions rather than a single solution for searching.
- GA uses fitness function for evaluation rather the derivatives.
- GA uses probabilistic transition and not deterministic rules.

Thank You!!!