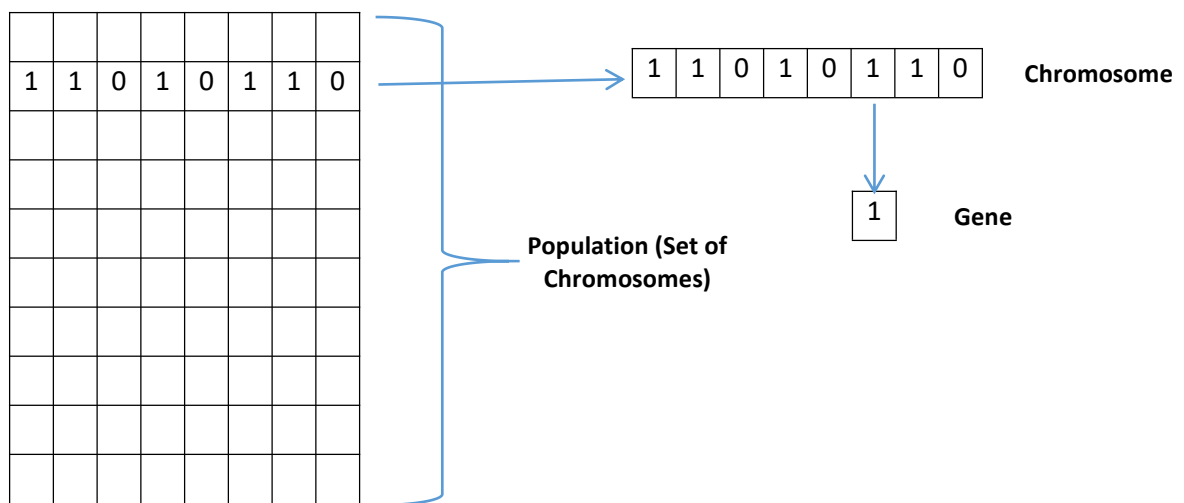# GENETIC ALGORITHM

Genetic Algorithm (**GA**) is a searched based optimization technique based on the principal of Genetics and Natural Selection. It is basically an optimization technique that mimic the process the natural evolution

In **GA**'s we have a pool or a population of possible solutions to the given problem. These solutions then undergo recombination and mutation (like in general genetics), producing new children, and the process is repeated over various generations. Each individual (or candidate solution) is assigned a fitness value (based on its objective function value) and the fitter individuals are given a higher chance to mate and yield more '**fitter**' individuals. This is line with the **Darwinian Theory** of **"Survival of the Fittest"**.

TERMINOLOGIES RELATED TO GENETIC ALGORITHM :-

- **Population :-** Subset of all the possible solutions to the given problem. The population for a **GA** is analogous to the population of human being.
  - **Chromosomes** :- A chromosome is one such solution to the given problem.
    - **Gene** :- A gene is one element position of a chromosome

Population (Set of Chromosomes)

Chromosome

Gene

GENETIC ALGORITHM OPERATORS & PARAMETERS

- ◆ **Selection** :- The process that determines which solutions are to be preserved and allowed to reproduce and which one deserves to die out. The function of the selection operation operator : identify the **good solutions** in a population, make multiple copies of the good solutions, eliminate the bad solutions from the population so that the multiple copies of good solutions can be placed in the population.

"Selects the best, discards the rest"

Some of the different techniques to implement selection :

> **Tournament Selection**
> **Roulette Wheel Selection**
> **Proportionate Selection** , etc

Now to identify the above mentioned *'good solutions'* we take help of the fitness function which is one of the parameter of the **Genetic Algorithm.**

✧ **Fitness Function** :- A fitness function value quantifies the optimality of a solution. The value is used to rank a particular solution against all the other solutions
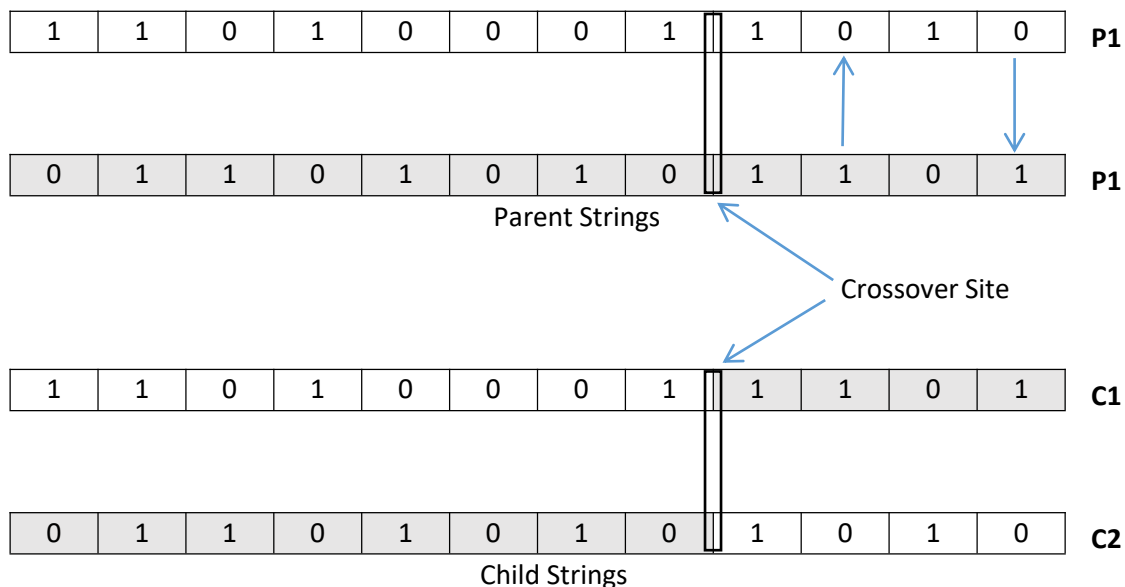
A fitness value is assigned to each solution depending on how close it is actually to the optimal solution of the problem.

◆ **Crossover** :- The crossover operator is analogous to reproduction and biological cross over. Basically, this operator is used to create new solutions from the existing solutions available in the mating pool after applying **Selection** operator.

Some of the different types of crossover:

> **One Point Crossover (Binary Crossover)**
> **Multi Point Crossover**
> **Uniform Crossover**

Below is an illustration of **Binary Crossover** :-

| 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | **P1** |

| 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | **P1** |

Parent Strings

Crossover Site

| 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | **C1** |

| 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | **C2** |

Child Strings

◆ **Mutation** :- Mutation is the occasional introduction of new features in to the solution strings of the population pool to maintain diversity in the population

Though crossover has the main responsibility to search for the optimal one, however mutation is also used for this purpose.

Before Mutation   **After Mutation**

Some of the different types of mutation:

➢ **Binary Mutation (Bit Flip Mutation)**
➢ **Swap Mutation**
➢ **Inversion Mutation**

Below is an illustration of **Binary Mutation (Bit Flip Mutation)** :-

The mutation operator changes a 1 to 0 or vice versa

1	1	0	1	0	0	0	1	1	1	0	1

Before Mutation

1	1	0	0	0	0	0	1	1	1	0	1

After Mutation

Thus a simple genetic algorithm (sga) :-

```
function sga()
{
        Initialize population;
        Calculate fitness_function;
        while (fitness_value != termination_criteria)
        {
                selection();
                crossover();
                mutation();
        }
}
```