**Football Fixtures (EPL Fixtures) Using Genetic Algorithms**

1. **Introduction:**

Genetic algorithms are a type of heuristic search algorithms that reflects the process of natural selection where the fittest individuals from the existing population are selected for reproduction to produce offspring of the next generation.

One of the application of Genetic Algorithm is **Scheduling** different events like University classes schedule, meeting fixtures schedule. We will study the use of GA for Football Fixtures, which is an NP complete problem and can be easily solved using Genetic Algorithm.

1. **Problem Statement:**

Implement Genetic Algorithm for Football Fixture problem. It is a scheduling problem which will create fixtures of the **English Premier League**. Input the team names, match locations and the start date, the algorithm will create fixtures for the league. Each team will have their respective home ground. Standard league rules will apply.

Here,

**Total number of matches in a tournament = ((Number of teams) \* (Number of teams - 1) \* (Number of Rounds)) / 2**

Following are the hard constrains used for creating the fixtures:

For example: Number of Rounds in the league: 2

1. A team cannot play with itself
2. Each team plays exactly 2(I.e. number of rounds) matches against each team in the league
3. Each team plays [total number of matches – number of rounds] matches in the league
4. Each team plays exactly one match at its own home ground and one match at opponent’s home ground. (One – home, One - away)
5. Two matches cannot take place on the same day and same location
6. A team cannot play 2 matches on the same day
7. **Implementation Details:**

Basic Algorithm for GA:

//add flowchart

1. Create initial population using population size
2. Compute fitness of each chromosome (fixtures of the league)
3. do{
4. Select the best fit individuals for reproduction

parents = selectParents(population[generation]);

1. Bread new individuals through crossover and mutation

population[generation + 1] = crossover(parents);

population[generation + 1] = mutate(population[generation + 1]);

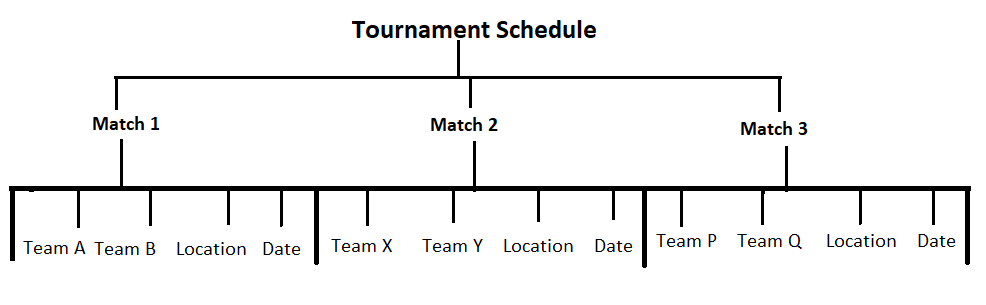
1. Evaluate fitness of new Individual

evaluatePopulation(population[generation] + 1);

1. Replace least fit population with new individuals
2. generation++;

} while( ! terminationCondition )

1. return the fittest indivisual



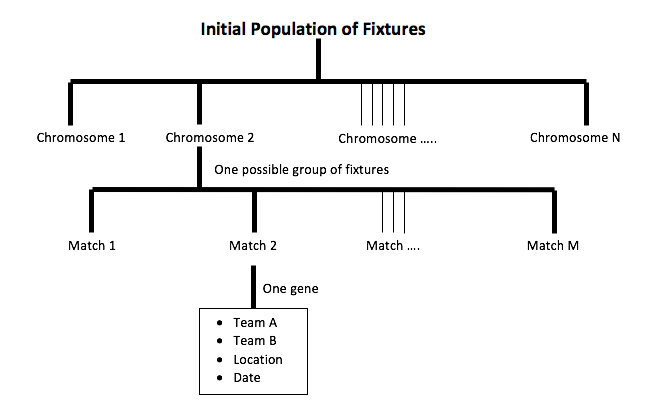


Figure 1: Basic layout of population, chromosome and gene

Population, Chromosome and Gene:

In genetic algorithms, a chromosome (also sometimes called a genotype) is a set of parameters which define a proposed solution to the problem that the genetic algorithm is trying to solve. The set of all solutions is known as the *population*.

In the case of football fixtures, a chromosome is considered one possible solution to the league fixtures scheduler. **A chromosome** consists of **all the matches** played in the league.

**A** **gene** is one element position of a chromosome. In this case, a gene represents **a match** which is played between 2 team at a location on a date.

Encoding is useful for efficient manipulation of the representation of a chromosome. Object of type – Match was encoded in the chromosome for better accessing and manipulation. Further, a Team object was encapsulated in the Match gene to represent the playing teams. POJO class for Team was created to encapsulate the team name, home ground.

To sum it up, a population will contain multiple chromosomes where each chromosome is a possible solution to the fixture problem.