COMP 473 / COMP 6731
Student Id: 40083939
Student Name: Apoorv Semwal
Mail-Id: a_semwal@encs.concordia.ca

Assignment 2: Obtain an optimal Project Selection to achieve maximum revenue using Genetic algorithm.

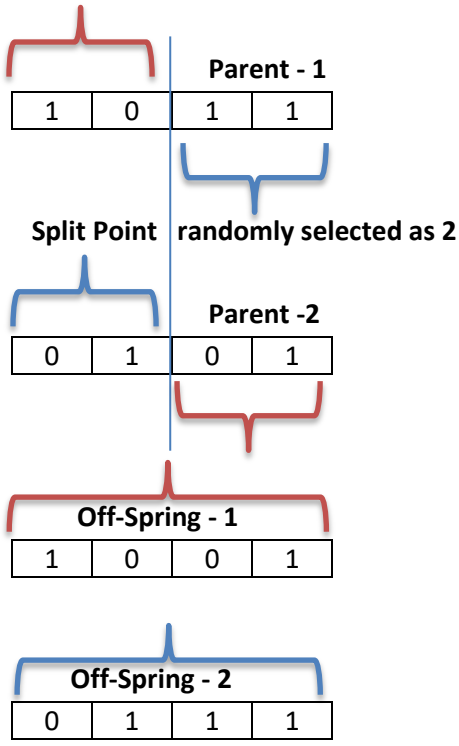
A. The **encoding** of the problem: As was explained in the hints section, problem was encoded like this:

Since there were **4 Projects,**
Each solution – Chromosome – would be represented by a **4-Bit binary sequence**, where a **"1" would mean Project is selected** and **"0" means it is not**.
Starting from left to right each bit represents one project in an ascending order, i.e.
Left-Most Bit is Project-1 while **Right-Most Bit is Project-4**.

For instance: **Chromosome** - **1011 would mean :** Project 1 – Selected, Project 2 – Not Selected, Project 3 – Selected, Project 4 – Selected.

B. **Crossover mechanism**:

1. **Two parents** are selected at Random based on "**Roulette Algorithm**".
2. A **random number** is generated to be taken as a **split point** for doing crossover.
3. Crossover is done to obtain two new Off-springs and both new off-springs are validated as per the constraints given. Only the ones satisfying the constraints are added to new generation.
4. For instance:

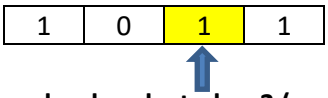


C. **Mutation Mechanism**: How frequently mutation should be done is **controlled by a Mutation Rate** which is given as a **Hyper-Parameter**. **Single bit** mutation is used.

1. **A Cross-overed-Offspring** is selected.
2. A **random number** is generated to be taken as an **index for mutation (Changing "0" to "1" and vice-versa).**
3. Mutation is done to obtain a new Off-Spring which is then validated as per the constraints given. Only if it satisfies the constraints it is added to the new generation.

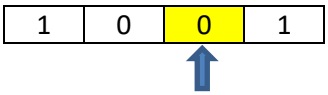
For instance:

Crossovered-Offspring



Mutation Index is randomly selected as 3 (assuming index starts from 1)

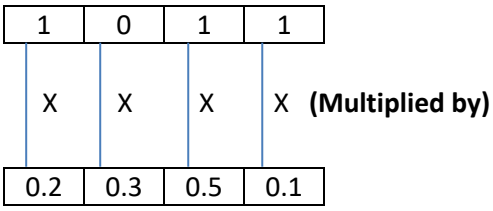
Mutated Off-Spring



D. **Fitness Function Used**: Fitness function at the crux of any Genetic Algorithm Problem provides us a numerical and quantifiable measure to find how close any individual Chromosome is to the desired solution.

For our scenario here the fitness function is pretty straightforward. As our **ultimate objective is to maximize the revenue, keeping in mind the given constraints** we Simply multiply and sum up the revenue values of individual projects if at all they are selected. In terms of their encoding in a Chromosome it is like this:

Chromosome



Revenue linked to each project (in ascending order)

Sum up all the individual Products: (0.2*1) + (0.3*0) + (0.5*1) + (0.1*1) = **0.8** - total revenue being taken as the **Fitness Value of this particular Chromosome**.

In general as was suggested in the hints:

Fitness function: maximize $f(x_1, x_2, x_3, x_4) = 0.2x_1 + 0.3x_2 + 0.5x_3 + 0.1x_4$.

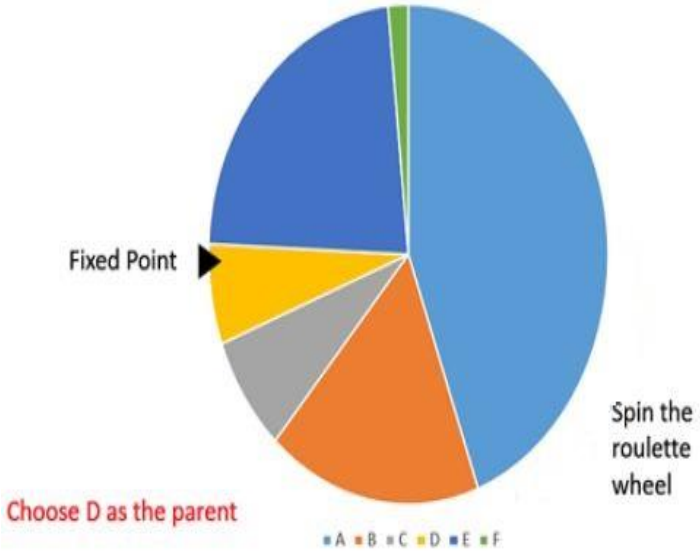
E. **Selection Methodologies Used**:

1. For Selecting Parents for Crossover "**Roulette Algorithm"** has been used.
   Roulette algorithm: Basic idea is to **assign fitness value to every chromosome** in the population and the probability of any chromosome being picked up for Crossover is directly proportional to its fitness value, i.e. **higher the fitness value higher is the probability of being picked up for Crossover**.

For instance: Let there be 4 Chromosomes A, B, C, D with fitness values as given:
A – 0.4
B – 0.1
C – 0.3
D – 0.2
So **A should have a 40% chance of being picked up as a parent, B -10%, C – 30% and D – 20%.**
Implementation – Generate a random number "R" between 1 and 100. If:
1<= R <=40 then **Choose A**.
41<= R <=50 then **Choose B**.
51<= R <=80 then **Choose C**.
81<= R <=100 then **Choose D**.

***Note – if sum of fitness values as shown above is not equal to 1 then we need to normalize these values by dividing each fitness value with total sum of fitness values. We do this normalization to make sum of probabilities = 1.**

Sample :



Chromosome	Fitness Value
A	8.2
B	3.2
C	1.4
D	1.2
E	4.2
F	0.3

2. **Elitism** has been used to achieve an early convergence: Elitism states that the **fittest member of every generation is carry forwarded to the next generation as it is**.
3. Selection of a chromosome in any generation is contingent upon that chromosome being able to **satisfy the capital constraints for every year, which is :**

Constraints: $0.5x_1 + 1.0x_2 + 1.5x_3 + 0.1x_4 \leq 3.1; 0.3x_1 + 0.8x_2 + 1.5x_3 + 0.4x_4 \leq 2.5; 0.2x_1 + 0.2x_2 + 0.3x_3 + 0.1x_4 \leq 0.4.$

F. Stopping Criteria : Generally when we are running any Genetic Algorithm there are two possible scenarios:

1. **We already know the target solution** and we code our algorithm to stop when the known target solution has appeared in any of the generations.

E.g.: Finding out if a particular phrase/word can be made using set of letters given. As in this case we already know the phrase or the word we need to find, we simply keep generating the new populations with the given set of letters until our word/phrase appears in any generation.

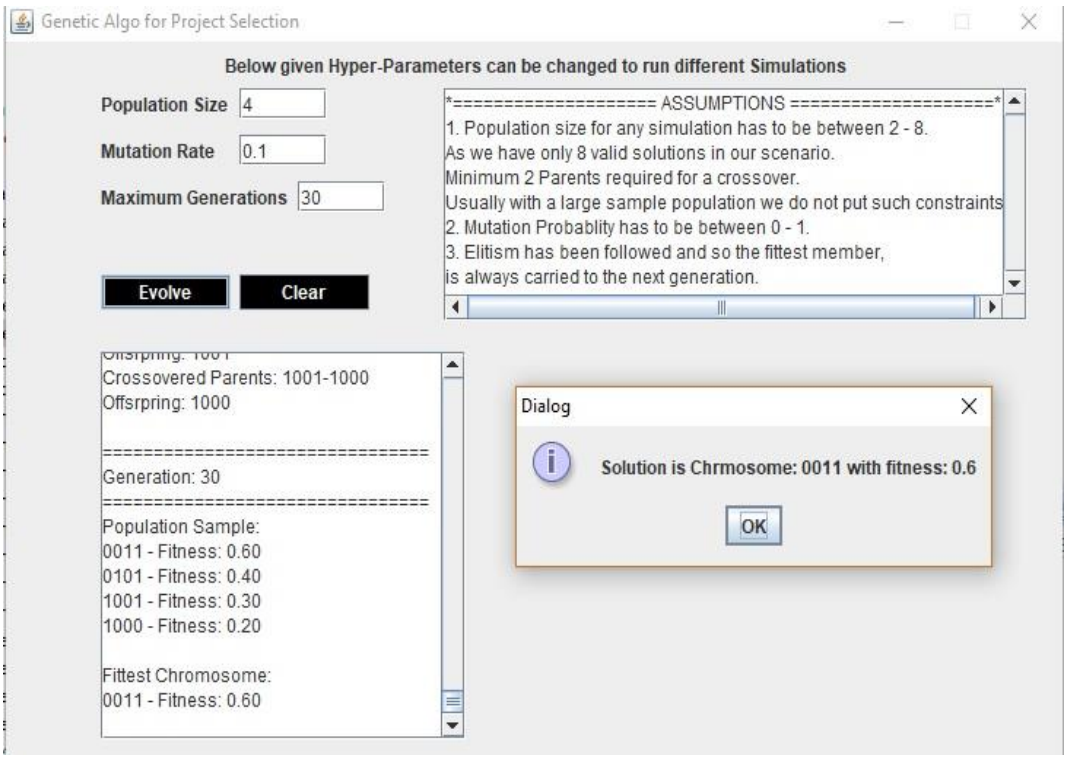
2. **We do not know the target solution** but we **expect our algorithm to converge to an optimal solution** which **may be the best** or **at least close to best**.
   Our assignment falls into this second category.
   Even though the Sample Space was very small for us (just 16 members), we have taken A **Maximum Generation count** as the **stopping criteria**.
   This "**Maximum Generation count" has been given as a hyper-parameter to run different simulations and compare various results.**

G. Assumptions:

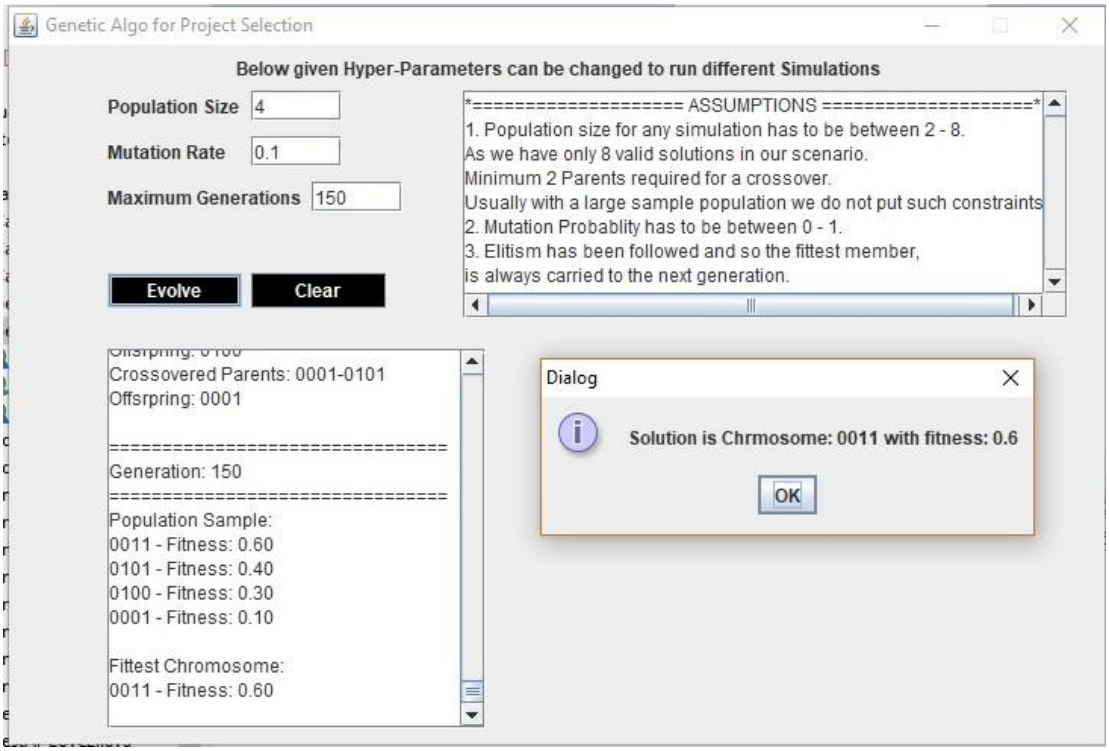
1. **Population size** for any simulation has to be between **2 and 8**. As we have **only 8 valid solutions** in our scenario and a **minimum of 2 parents are required for crossover**. **Usually with a large sample population we do not put such constraints but since we had a limited number of valid solutions we had to put this check.**
2. **Mutation Probability** has to be between 0 - 1.
3. **Elitism** has been followed and so the fittest member is always carried to the next generation.

H. Screenshots for some random simulations:

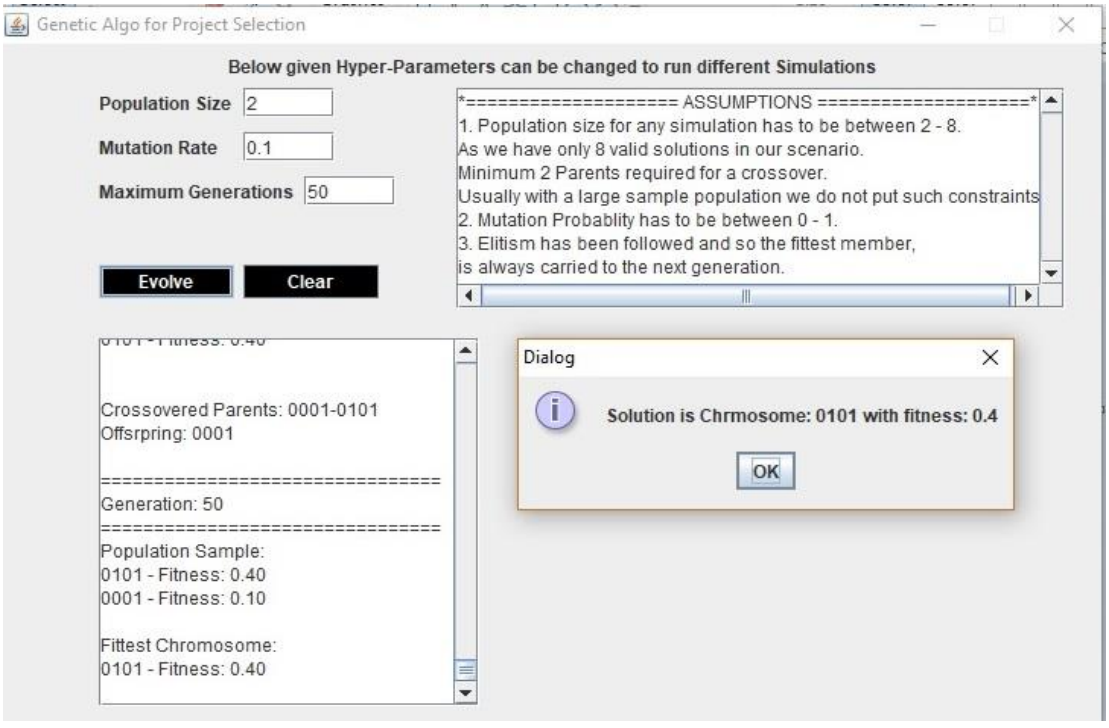
1. Max Generation 30



2. **With Max Generation as 150**



3. **With Population size 2 we got a maximum fitness 0.4 which was lesser than previous simulations where we had got 0.6**



4. Another value for Maximum Fitness as 0.5 with 30 Generations and Population size 3.