IIT KHARAGPUR

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Introduction to Soft Computing

GA Operator: Encoding-I

Debasis Samanta

Department of Computer Science and Engineering

IIT KHARAGPUR

GA Operators

Following are the GA operators in Genetic Algorithms.

1) Encoding
2) Convergence test
3) Mating pool
4) Fitness Evaluation
5) Crossover
6) Mutation
7) Inversion

Debasis Samanta
CSE
IIT Kharagpur

Encoding Operation

1) Encoding
2) Convergence test
3) Mating pool
4) Fitness Evaluation
5) Crossover
6) Mutation
7) Inversion

Debasis Samanta
CSE
IIT Kharagpur

Different Encoding Schemes

- **Different GA's**
  - Simple Genetic Algorithm (SGA)
  - Steady State Genetic Algorithm (SSGA)
  - Messy Genetic Algorithm (MGA)
- **Encoding Schemes**
  - Binary encoding
  - Real value encoding
  - Order encoding
  - Tree encoding

Debasis Samanta
CSE
IIT Kharagpur

Different Encoding Schemes

Often, GAs are specified according to the encoding scheme it follows.

For example:

- Encoding Scheme
- Binary encoding –> Binary Coded GA or simply **Binary GA**
- Real value encoding –> Real Coded GA or simply **Real GA**
- Order encoding –> **Order GA** (also called as **Permuted GA**)
- Tree encoding

Encoding Schemes in GA

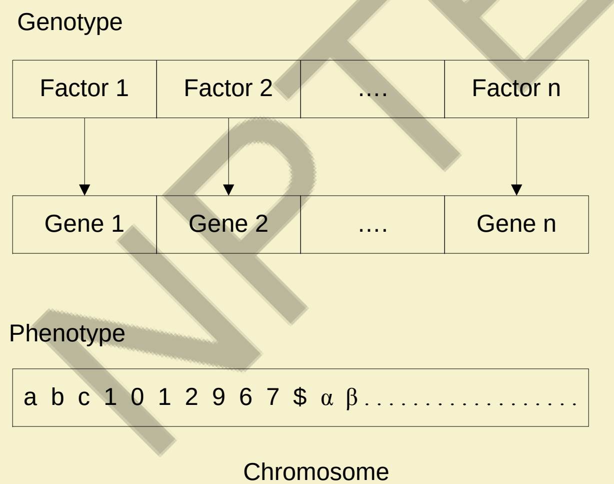
Genetic Algorithm uses metaphor consisting of two distinct elements :

1) Individual
2) Population

An individual is a single solution while a population is a set of individuals at an instant of searching process.

Individual Representation :Phenotype and Genotype

- An individual is defined by a chromosome. A chromosome stores genetic information (called phenotype) for an individual.
- Here, a chromosome is expressed in terms of factors defining a problem.



Individual Representation :Phenotype and Genotype

Note :

- A gene is the GA's representation of a single factor (i.e. a design parameter), which has a domain of values (continuous, discontinuous, discrete etc.) symbol, numbering, etc.

- In GA, there is a mapping from genotype to phenotype. This eventually decides the performance (namely speed and accuracy) of the problem solving.

Encoding techniques

There are many ways of encoding:

1) **Binary encoding:** Representing a gene in terms of bits (0s and 1s).
2) **Real value encoding:** Representing a gene in terms of values or symbols or string.
3) **Permutation (or Order) encoding:** Representing a sequence of elements.
4) **Tree encoding:** Representing in the form of a tree of objects.

Binary Encoding

In this encoding scheme, a gene or chromosome is represented by a string (fixed or variable length) of binary bits (0's and 1's)

A :	0 1 1 0 0 1 0 1 0 1 0 1 0 1 1 1 1 0	Individual 1
B :	0 0 1 0 1 0 1 1 1 0 1 0 1 0 1 0 0 0	Individual 2

Example: 0-1 Knapsack problem

- There are n items, each item has its own cost (c_i) and weight (w_i).
- There is a knapsack of total capacity w.
- The problem is to take as much items as possible but not exceeding the capacity of the knapsack.

This is an optimization problem and can be better described as follows.

Maximize

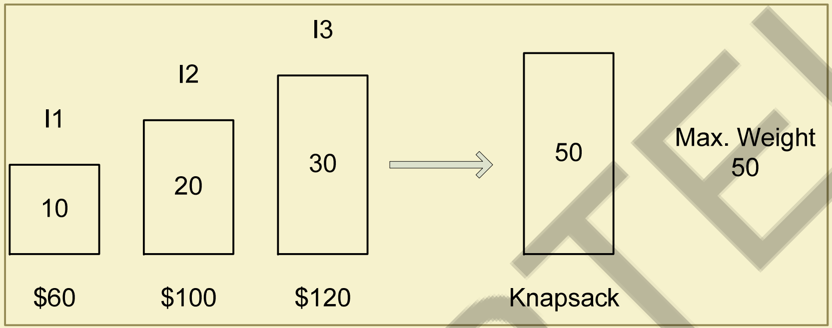
$$\sum_i c_i \times w_i \times x_i$$

Subject to

$$\sum x_i \times w_i \leq W \qquad \text{where } x_i \in [0 \ldots \ldots \ldots 1]$$

Example: 0-1 Knapsack problem

Consider the fallowing, an instance of the $0-1$ Knapsack problem.



Brute force approach to solve the above can be stated as follows:

1) Select at least one item

   [10], [20], [30], [10, 20], [10, 30], [20, 30], [10, 20, 30]

2) So, for n-items, are there are 2^{n-1} trials.

3) 0 – means item not included and 1 – means item included

   [100], [010], [011], [110], [101], [011], [111]

Example: 0-1 Knapsack problem

The encoding for the 0-1 Knapsack, problem, in general, for n items set would look as follows.

Genotype :

1	2	3	4	. . . . .	n-1	n
				. . . . .		

Phenotype :

0 1 0 1 1 0 1 0 1 0 1 0 1. . . . . .1 0 1

A binary string of n-bits

Few more examples

Example 1 :

Minimize :

$$f\ (x) = \frac{x^2}{2} + \frac{125}{x}$$

where $0 \leq x \leq 15$ and x is any discrete integer value.

Genotype :

x

Phenotype :

0 1 1 0 1

A binary string of 5-bits

Few more examples

Example 2 :

Maximize :

$$f\ (x, y) = x^3 - x^2y + xy^2 + y^3$$

subject to:

$$x + y \leq 10$$

and

$$1 \leq x \leq 10$$
$$-10 \leq y \leq 10$$

Genotype :

x	y

Phenotype :

0 1 1 0 1	1 1 0 0 1

Two binary string of 5-bits each

Pros and cons of Binary encoding scheme

- **Limitations:**
  1) Needs an effort to convert into binary from
  2) Accuracy depends on the binary representation
- **Advantages:**
  1) Since operations with binary representation is faster, it provide a faster implementations of all GA operators and hence the execution of GAs.
  2) Any optimization problem has it binary-coded GA implementation

Real value encoding

- The real-coded GA is most suitable for optimization in a continuous search space.
- Uses the direct representations of the design parameters.
- Thus, avoids any intermediate encoding and decoding steps.

Genotype :

x	**y**

Phenotype :

5.28	**-475.36**

Real-value representation

Real value encoding with binary codes

Methodology: Step 1 [Deciding the precision]

For any continuous design variable x such that $X_L \leq x \leq X_U$, and if ϵ is the precision required, then string length n should be equal to

$$n = log_2\left(\frac{X_U - X_L}{\epsilon}\right)$$

Equivalently,

$$\epsilon = \left(\frac{X_U - X_L}{2^n}\right)$$

In general

$\epsilon = [0 \ldots \ldots 1]$. It is also called, Obtainable accuracy

Note:

If ϵ = 0.5, then 4.05 or 4.49 ≡ 4 and 4.50 or 4.99 ≡ 4.5 and so on.

Real value encoding: Illustration 1

1) **Example 1:**

   $1 \leq x \leq 16, n = 6$. What is the accuracy?

$$\epsilon = \frac{16-1}{2^6} = \frac{15}{64} = 0.249 \approx 0.25$$

2) **Example 2:**

   What is the obtainable accuracy, for the binary representation for a variable X in the range $20.1 \leq X \leq 45.6$ with 8-bits?

3) **Example 3:**

   In the above case, what is the binary representation of $X = 34.35$?

Real value encoding with binary codes

1) Methodology: Step 2[Obtaining the binary representation]

Once, we know the length of binary string for an obtainable accuracy (i.e. precision), then we can have the following mapping relation from a real value X to its binary equivalent decoded value X_B, which is given by

$$X = X_L + \frac{X_U - X_L}{2^n - 1} \times X_B$$

where $X_B =$ Decoded value of a binary string,
n is the number of bits in the representation,
$X_L = 0\ 0\ 0\ 0\ 0\ 0 \ldots \ldots \ldots 0$ and $X_U = 1\ 1\ 1\ 1\ 1\ 1 \ldots \ldots \ldots 1$
are the decoded values of the binary representation of the lower and upper values of X.

Real value encoding: Illustration 2

Example:

Suppose, $X_L = 2$ and $X_U = 17$ are the two extreme decoded values of a variable x.

$n = 4$ is the number of binary bits in the representation for x.

$X_B = 10(= 1\ 0\ 1\ 0)$ is a decoded value for a given x.

What is the value of $x\ =?$ and its binary representation??

Here, $x = 2 + \frac{17-2}{2^4-1} \times 10 = 12$

Binary representation of $x = 1\ 1\ 0\ 0$

Thank You!!

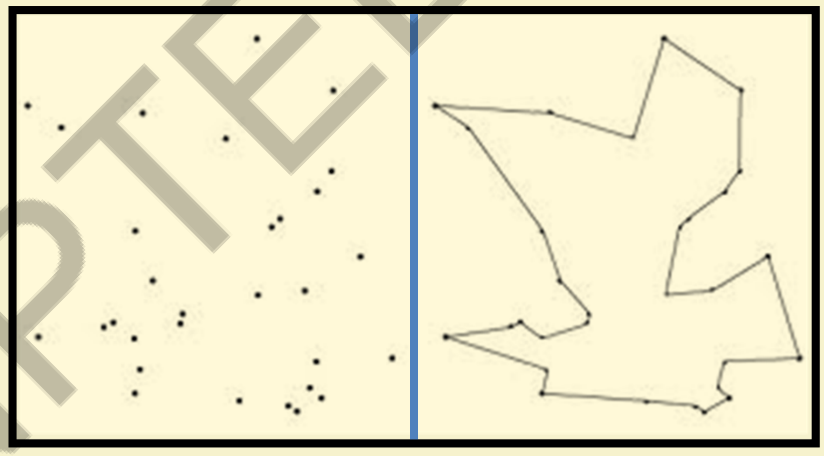
Introduction to Soft Computing

GA Operator: Encoding-II

Debasis Samanta
Department of Computer Science and Engineering
IIT KHARAGPUR

Order Encoding

Let us have a look into the following instance of the Traveling Salesman Problem (TSP).



TSP

- Visit all the cities
- One city once only
- Starting and ending city is the same

How we can formally define the TSP?

Order Encoding for TSP

Understanding the TSP:

There is a cost of visiting a city from another city and hence the total cost of visiting all the cities but exactly once (except the starting city).

Objective function:

To find a tour (i.e. a simple cycle covering all the cities) with a minimum cost involved.

Constraints:

1) All cities must be visited.
2) There will be only one occurrence of each city (except the starting city).

Design parameters:

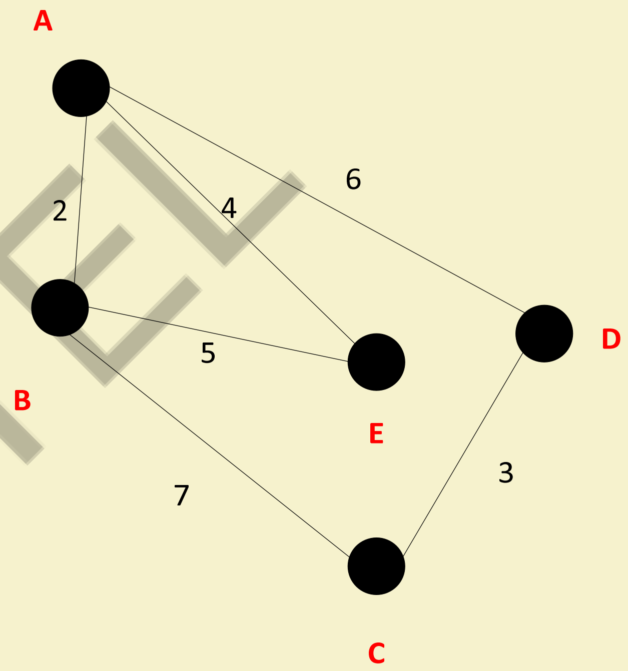
1) Euclidean distance may be taken as the measurement of the cost, otherwise, if it is specified explicitly.
2) The above stated information are the design variables in this case.

We are to search for the best path out of $n!$ possible paths.

A small instance of the TSP

d	A	B	C	D	E
A	0	2	∝	6	4
B	2	0	7	∝	5
C	∝	7	0	3	1
D	6	∝	3	0	∝
E	4	5	1	∝	0

d= Distance matrix



Connectivity among cities

Defining the TSP

Minimizing

$$cost = \sum_{i=0}^{n-2} d(c_i, c_{i+1}) + d(c_{n-1}, c_0)$$

Subject to

$$P = [c_0, c_1, c_2, c_3, \ldots \ldots , c_{n-1}]$$

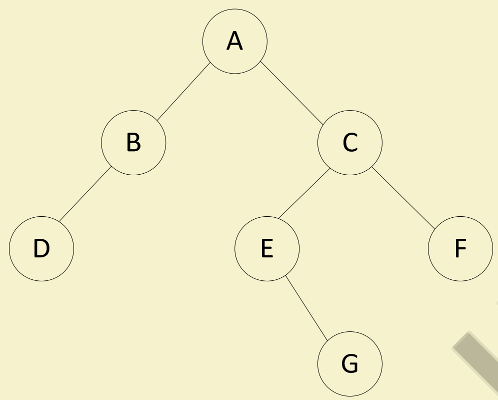
where $c_i \in X$;
Here, P is an ordered collection of cities and $c_i \neq c_j$ such that $\forall i, j = 0,1, \ldots \ldots , n-1$

Note: P represents a possible tour with the starting cities as c_0.

and
$X = x_1, x_2, x_3, \ldots \ldots , x_n$, set of n number of cities,
$d(x_i, x_j)$ is the distance between any two cities x_i and x_j.

Tree encoding

In this encoding scheme, a solution is encoded in the form of a binary tree.



A binary tree

D B A E G C F (In-order)

(T_L R T_R)

A B D C E G F (Pre-order)

(R T_L T_R)

D B G E F C A (Post-order)

(T_L T_R R)

Three compact representation

IIT KHARAGPUR

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Introduction to Soft Computing

Genetic Operator: Selection-I

Prof. Debasis Samanta

Department of Computer science & Engineering

IIT Kharagpur

Important GA Operations

✓ **Encoding**

- **Fitness evaluation and Selection**
- Mating pool
- Crossover
- Mutation
- Inversion
- Convergence test

Topics to be discussed..

- **GA Selection and Selection operations**
- **Fitness Evaluation**
- **Selection Schemes in GAs**
    - Canonical selection
    - Roulette Wheel selection
    - Rank-based selection
    - Tournament selection
    - Steady-state selection

GA Selection

- After deciding an encoding scheme, the second important thing is how to perform selection from a set of population, that is, how to choose the individuals in the population that will create offspring for the next generation and how many offspring each will create.

- The purpose of selection is, of course, to emphasize **fittest** individuals in the population in hopes that their offspring will in turn have even higher fitness.

Selection operation in GAs

- Selection is the process of creating the population for next generation from the current generation.
    - To generate new population: Breeding in GA
        - Create a mating pool
        - Select a pair
        - Reproduce

Fitness evaluation:

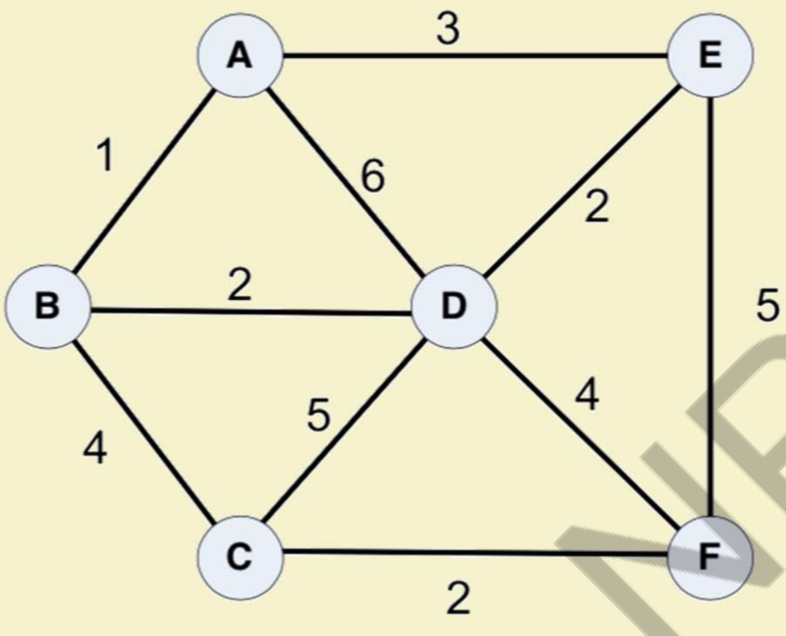
- **In GA, there is a need to create next generation**
  - The next generation should be such that it is toward the (global) optimum solution
  - Random population generation may not be a wiser strategy
  - Better strategy follows the biological process: **Selection**
- **Selection involves:**
  - Survival of the fittest
  - Struggle for the existence
- **Fitness evaluation is to evaluate the survivability of each individual in the current population**

Fitness evaluation:

How to evaluate the fitness of an individual?

- A simplest strategy could be to take the confidence of the value(s) of the objective function(s)
  - Simple, if there is a single objective function
  - But, needs a different treatment if there are two or more objective functions
    - They may be in different scales
    - All of them may not be same significant level in the fitness calculation. . . etc.

An example:



P1:	C	B	A	D	F	E	$\boxed{11}$
P2:	A	B	D	C	E	F	$\boxed{19}$
P3:	A	C	B	F	E	D	$\boxed{16}$
P4:	F	C	D	B	E	A	$\boxed{12}$
P5:	C	F	D	A	B	E	$\boxed{10}$

Selection Schemes in GAs:

Different strategies are known for the selection:

- **Canonical selection** (also called proportionate-based selection)
- **Roulette Wheel selection** (also called proportionate-based selection)
- **Rank-based selection** (also called as ordinal-based selection)
- **Tournament selection**
- **Steady-state selection**
- Boltzman selection

Canonical selection:

- In this techniques, fitness is defined for the $i-th$ individual as follows.

$$fitness(i) = \frac{f_i}{\bar{F}}$$

where f_i is the evaluation associated with the $i-th$ individual in the population.

- $\bar{F}$ is the average evaluation of all individuals in the population size N and is defined as follows.

$$\bar{F} = \frac{\sum_{i=1}^{N} f_i}{N}$$

Canonical selection:

- In an iteration, we calculate $\frac{f_i}{\bar{F}}$ for all individuals in the current population.
- In Canonical selection, the probability that individuals in the current population are copied and placed in the mating pool is proportional to their fitness.

Note :

- Here, the size of the mating pool is $p\% \times N$, for some p.
- Convergence rate depends on p.

Roulette-Wheel selection:

- In this scheme, the probability for an individual is being selected in the mating pool is considered to be proportional to its fitness.

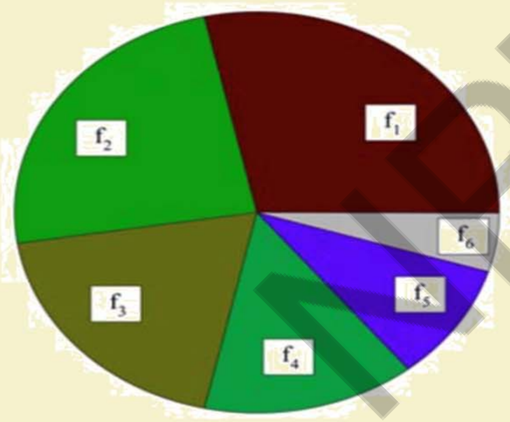
- It is implemented with the help of a wheel as shown in the next slide.

Roulette-Wheel selection:

$i \longrightarrow f_i$

$j \longrightarrow f_j$

$fi > fj \Rightarrow i$ (Preferable than j)



[Wheel Game (Rotate and See the Pointer)]

Roulette-Wheel selection mechanism:

- The top surface area of the wheel is divided into N parts in proportion to the fitness values $f_1, f_2, f_3 \cdots f_N$.
- The wheel is rotated in a particular direction (either clockwise or anticlockwise) and a fixed pointer is used to indicate the winning area, when it stops rotation.
- A particular sub-area representing a GA-Solution is selected to be winner probabilistically and the probability that the $i - th$ area will be declared as

$$p_i = \frac{f_i}{\sum_{i=1}^{N} f_i}$$

Roulette-Wheel selection mechanism:

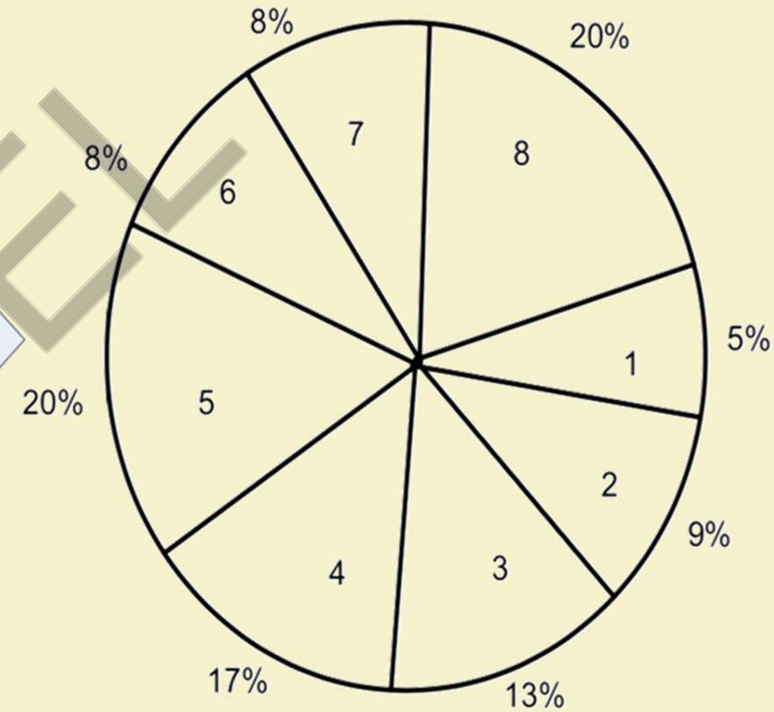
- In other words, the individual having higher fitness value is likely to be selected more.
- The wheel is rotated for N_p times (where $N_p = p\%N$, for some p) and each time, only one area is identified by the pointer to be the winner.

Note :

- Here, an individual may be selected more than once.
- Convergence rate is fast.

Roulette-Wheel selection mechanism: An Example

Individual	Fitness value	p_i
1	1.01	**0.05**
2	2.11	**0.09**
3	3.11	**0.13**
4	4.01	**0.17**
5	4.66	**0.20**
6	1.91	**0.08**
7	1.93	**0.08**
8	4.51	**0.20**



Roulette-Wheel selection : Implementation

Input: A Population of size N with their fitness values
Output: A mating pool of size N_p

Steps:

1) Compute $p_i = \frac{f_i}{\sum_{i=1}^{N} f_i}, \forall i = 1,2 \cdots N$
2) Calculate the cumulative probability for each of the individual starting from the top of the list, that is
$$P_i = \sum_{j=1}^{i} p_j, \text{ for all } j = 1,2 \cdots N$$
3) Generate a random number say r between 0 and 1.
4) Select the $j-th$ individual such that $P_{j-1} < r \leq P_j$
5) Repeat Step 3-4 to select N_p individuals.
6) End

Roulette-Wheel selection: Example

The probability that $i-th$ individual will be pointed is $p_i = \frac{f_i}{\sum_{i=1}^{N} f_i}$

Example:

Individual	p_i	P_i	r	T
1	0.05	0.05	0.26	I
2	0.09	0.14	0.04	I
3	0.13	0.27	0.48	II
4	0.17	0.44	0.43	I
5	0.20	0.64	0.09	II
6	0.08	0.72	0.30	
7	0.08	0.80	0.61	
8	0.20	1.0	0.89	I

p_i = Probability of an individual

r = Random Number between 0..1

P_i = Cumulative Probability

T=Tally count of selection

Roulette-Wheel selection

Following are the point to be noted:

1) The bottom-most individual in the population has a cumulative probability $P_N = 1$
2) Cumulative probability of any individual lies between 0 and 1
3) The $i-th$ individual in the population represents the cumulative probability from $P_{i\ \_\ 1}\ to\ Pi$
4) The top-most individual represents the cumulative probability values between 0 and p_1

Roulette-Wheel selection

Following are the point to be noted:

5) It may be checked that the selection is consistent with the expected count $E_i = N \times pi$ for the $i - th$ individual.

Does the selection is sensitive to ordering, say in ascending order of their fitness values?

Drawback in Roulette-Wheel selection

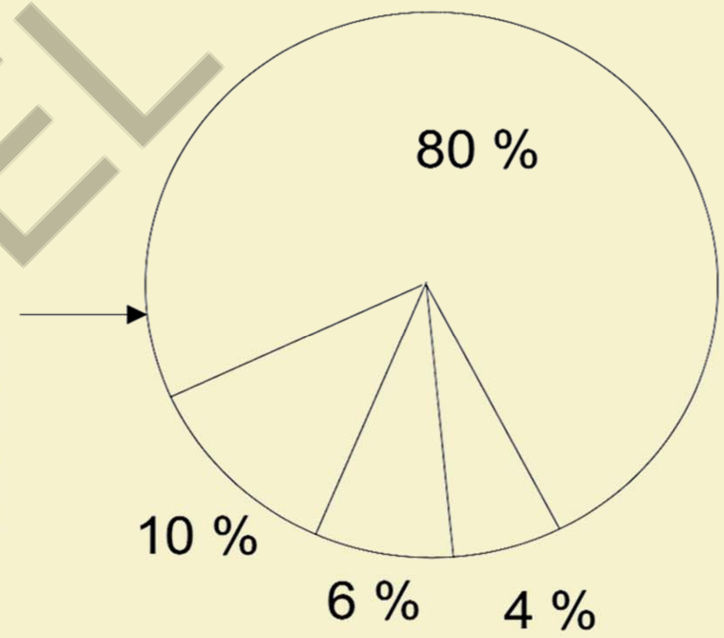
- Suppose, there are only four binary strings in a population, whose fitness values are $f_1, f_2, f_3 \ and \ f_4$.
- Their values 80%, 10%, 6% and 4%, respectively.

What is the expected count of selecting f_3, f_4, f_2 or f_1?

Problem with Roulette-Wheel selection scheme

The limitations in the Roulette-Wheel selection scheme can be better illustrated with the following figure.

Individual (i)	Fitness (fi)	RW (Area)
1	0.4	80 %
2	0.05	10 %
3	0.03	6 %
4	0.02	4 %



Problem with Roulette-Wheel selection scheme

- The observation is that the individual with higher fitness values will guard the other to be selected for mating.

- This leads to a **lesser diversity** and hence fewer scope toward exploring the alternative solution and also **premature convergence** or early convergence with **local optimal solution**.

Thank You!!

IIT KHARAGPUR

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Introduction to Soft Computing

Genetic Operator: Selection-II

Prof. Debasis Samanta

Department of Computer science & Engineering

IIT Kharagpur

Topics to be discussed..

- **Selection Schemes in GAs**
    - ✓ Canonical selection
    - ✓ Roulette Wheel selection
    - Rank-based selection
    - Tournament selection
    - Steady-state selection

Rank-based selection

- To overcome the problem with Roulette-Wheel selection, a rank-based selection scheme has been proposed.
- The process in rank selection consists of two steps.
  1. Individuals are arranged in an ascending order of their fitness values. The individual, which has the lowest value of fitness is assigned rank 1, and other individuals are ranked accordingly.

  2. The proportionate based selection scheme is then followed based on the assigned rank.

Rank-based selection

Note:

- The % area to be occupied by a particular individual i, is given by

$$\frac{r_i}{\sum_{i=1}^{N} r_i}$$

  where r_i indicates the rank of the $i - th$ individual.

- Two or more individuals with the same fitness values should have the same rank.

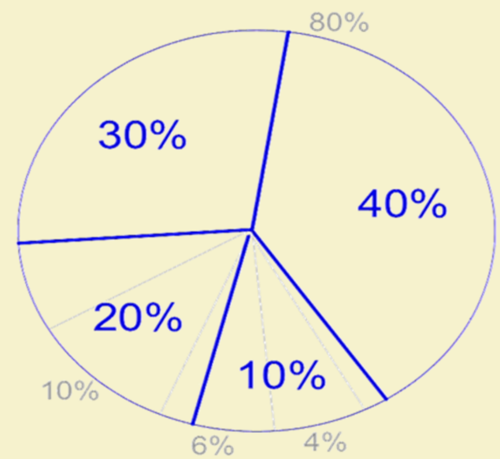
Rank-based selection: Example

- Continuing with the population of 4 individuals with fitness values:

$$f_1 = 0.40, f_2 = 0.05, f_3 = 0.03 \; and \; f_4 = 0.02.$$

- Their proportionate area on the wheel are: 80%, 10%, 6% and 4%
- Their ranks are shown in the following figure.

Individual (i)	Fitness (fi)	RW (Area)	Rank	RS (Area)
1	0.4	80 %	4	40 %
2	0.05	10 %	3	30 %
3	0.03	6 %	2	20 %
4	0.02	4 %	1	10 %



It is evident that expectation counts have been improved compared to Roulette-Wheel selection.

Rank-based selection: Implementation

Input: A Population of size N with their fitness values
Output: A mating pool of size N_p

Steps:

1) Arrange all individuals in ascending order of their fitness value.
2) Rank the individuals according to their position in the order, that is, the worst will have rank 1, the next rank 2 and best will have rank N.
3) Apply the Roulette-Wheel selection but based on their assigned ranks. For example, the probability p_i of the i-th individual would be

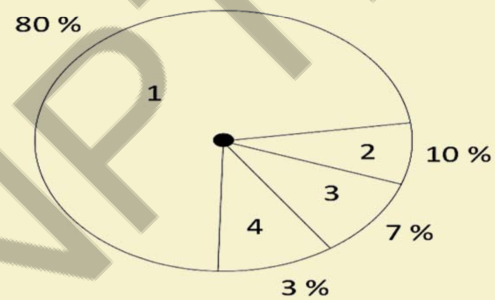
$$p_i = \frac{r_i}{\sum_{j=1}^{i} r_j}$$

4) Stop

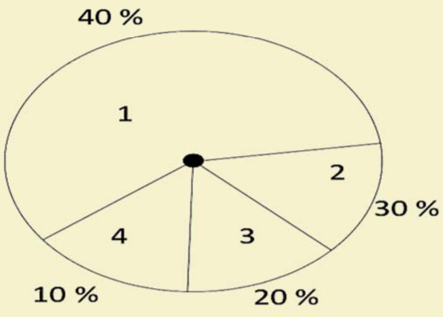
Comparing Rank-based selection with Roulette-Wheel selection

Individual	% Area	f_i	Rank (r_i)	% Area
1	80 %	0.4	4	40 %
2	10 %	0.05	3	30 %
3	7 %	0.03	2	20 %
4	4 %	0.02	1	10 %

A rank-based selection is expected to perform better than the Roulette-Wheel selection, in general.



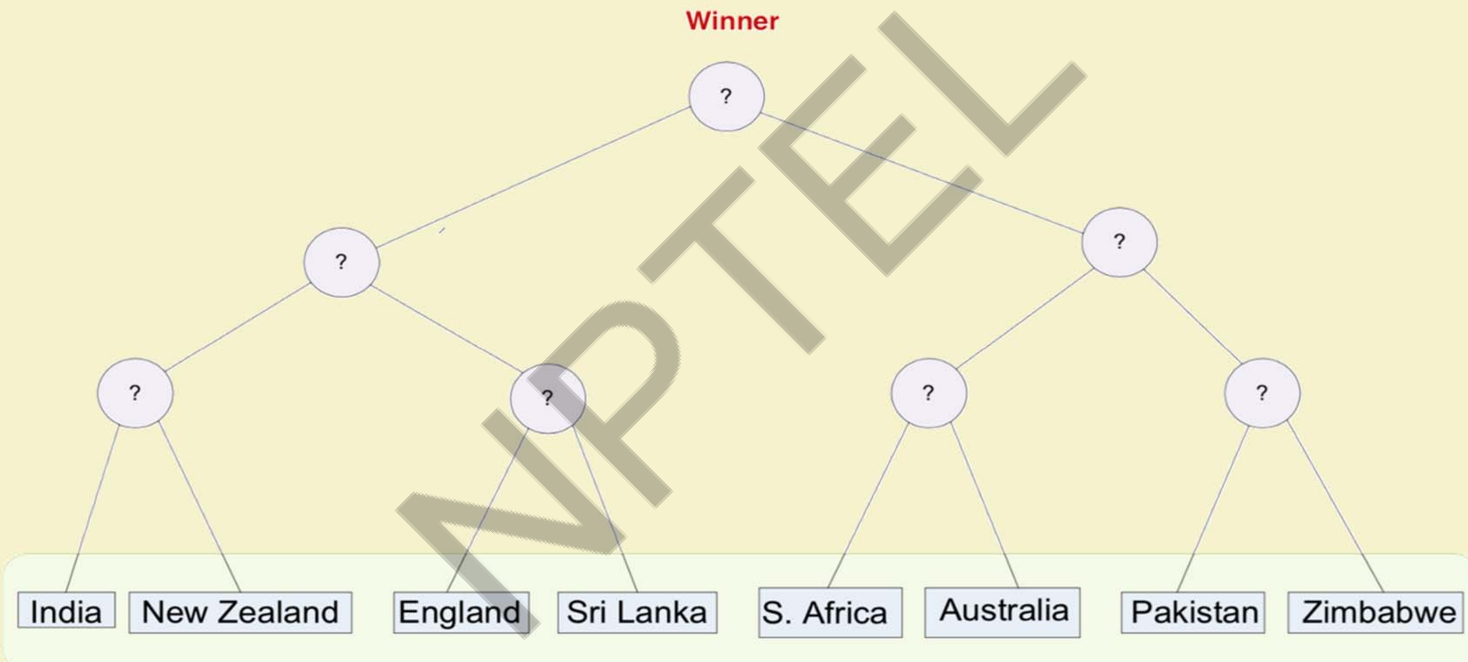
Roulette-Wheel based on proportionate-based selection



Roulette-Wheel based on ordinal-based selection

Tournament Selection

Who will win the match in this tournament?



Tournament selection

1) In this scheme, we select the tournament size n at random.
2) We pick n individuals from the population, at random and determine the best one in terms of their fitness values.
3) The best individual is copied into the mating pool.
4) Thus, in this scheme only one individual is selected per tournament and N_p tournaments are to be played to make the size of mating pool equals to N_p.

Tournament selection : Implementation

Input: A Population of size N with their fitness values
Output: A mating pool of size N_p $(N_p \leq N)$

Steps:

1) Select N_U individuals at random $(NU \leq N)$.
2) Out of N_U individuals, choose the individual with highest fitness value as the winner.
3) Add the winner to the mating pool, which is initially empty.
4) Repeat Steps 1-3 until the mating pool contains N_p individuals
5) Stop

Tournament selection : Example

$$N = 8,\ N_U = 2,\ N_p = 8$$

Input :

Individual	1	2	3	4	5	6	7	8
Fintess	1.0	2.1	3.1	4.0	4.6	1.9	1.8	4.5

Output :

Trial	Individuals	Selected
1	2, 4	4
2	3, 8	8
3	1, 3	3
4	4, 5	5
5	1, 6	6
6	1, 2	2
7	4, 2	4
8	8, 3	8

If the fitness values of two individuals are same, than there is a tie in the match!! So, what to do????

Tournament selection

Note:

- Here, there is a chance for a good individual to be copied into the mating pool more than once.

- This techniques founds to be computationally more faster than both Roulette-Wheel and Rank-based selection scheme.

Tournament selection

Note:

There are different twists can be made into the basic Tournament selection scheme:

- Frequency of N_U = small value (2, 3), moderate 50 % of N and large $N_U \approx N$.
- Once an individual is selected for a mating pool, it can be discarded from the current population, thus disallowing the repetition in selecting an individual more than once.
- Replace the worst individual in the mating pool with those are not winners in any trials.

Steady-State selection algorithm

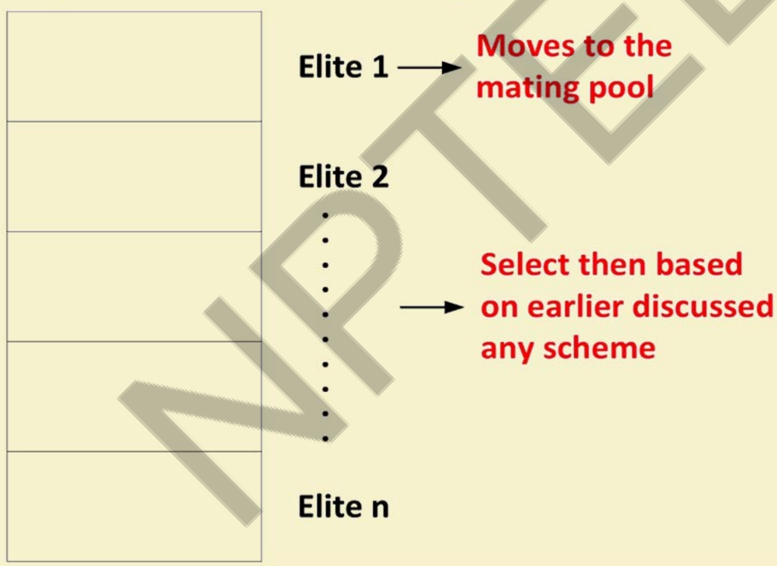
Steps:

1. N_U individuals are selected at random.
2. N_U individuals with the worst fitness values are replaced with N_U individuals selected in Step 1 and are added into the mating pool.

This completes the selection procedure for one iteration. Repeat the iteration until the mating pool of desired size is obtained.

Elitisms

- In this scheme, an elite class (in terms of fitness) is identified first in a population of strings.
- It is then directly copied into the next generation to ensure their presence.



Comparing selection schemes

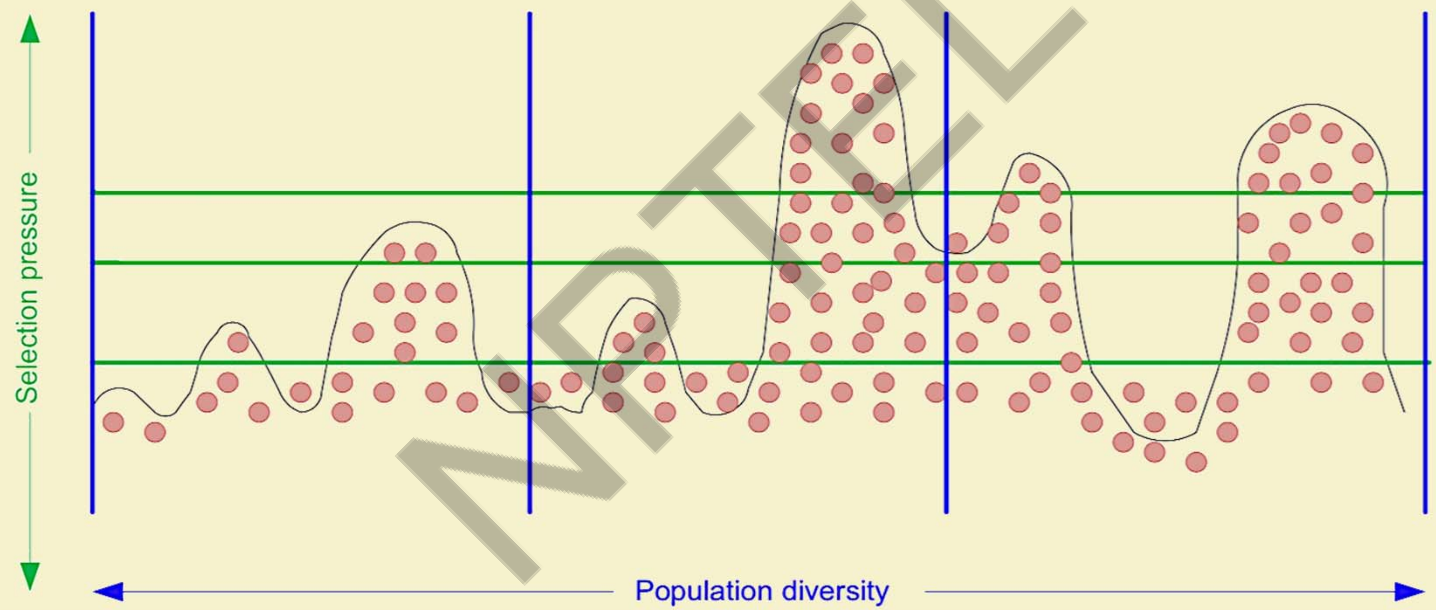
- Usually, a selection scheme follows Darwin's principle of **"Survival of the fittest"**.
- In other words, a selection strategy in GA is a process that favours the selection of better individuals in the population for the matting pool (so that better genes are inherited to the new offspring) and hence search leads to the global optima.

There are two issues to decide the effectiveness of any selection scheme.

- Population diversity
- Selection pressure

Analyzing a selection schemes

- More population diversity means more exploration
- Higher selection pressure means lesser exploitation



Effectiveness of any selection scheme

Population diversity

- This is similar to the **concept of exploration.**
- The population diversity means that the genes from the already discovered good individuals are exploited while permitting the new area of search space continue to be explored.

Selection pressure

- This is similar to the **concept of exploitation.**
- It is defined as the degree to which the better individuals are favoured.

Effectiveness of any selection scheme

These two factors are inversely related to each other in the sense that if the selection pressure increases, the population diversity decrease and vice-versa. Thus,

If selection pressure is HIGH

1. The search focuses only on good individuals (in terms of fitness) at the moment.
2. It loses the population diversity.
3. Higher rate of convergence. Often leads to pre-mature convergence of the solution to a sub-optimal solution.

Effectiveness of any selection scheme

If the selection pressure is LOW

1. May not be able to drive the search properly and consequently the stagnation may occurs.
2. The convergence rate is low and GA takes unnecessary long time to find optimal solution.
3. Accuracy of solution increases (as more genes are usually explored in the search).

Analysis of different selection strategies

Selection Scheme	Population Diversity	Selection Pressure
Roulette-wheel selection (It works fine when fitness values are informally distributed)	• Low Population Diversity - Pre-mature convergence - Less Accuracy in solution	• It is with high selection pressure - Stagnation of Search
Rank Selection (It works fine when fitness values are not necessarily uniformly distributed)	• Favors a high population diversity - Slow rate of convergence	• Selection pressure is low - Explore more solutions
Tournament Selection (It works fine when population are with very diversified fitness values)	• Population diversity is moderate - Ends up with a moderate rate of convergence	• It provides very high selection pressure - better exploration of search space
Steady-state Selection	• Population diversity is decreases gradually as the generation advances	• Selection pressure is too low. - Convergence rate is too slow

Fine tuning a selection operator : Generation Gap

- The generation gap is defined as the proportion of individuals in the population, which are replaced in each generation, i.e.

$$G_p = \frac{p}{N}$$

Where N is the population size and p is the number of individuals that will be replaced.

Note that in steady-state selection with $N_u = 2$ ($p = 2$) and hence $G_p \approx 0$ for a large population whereas other selection schemes has $G_p \approx 1$

Fine tuning a selection operator : Generation Gap

To make the G_p a large value, several strategies may be adopted.

- Selection of individuals according to their fitness and replacement at random
- Selection of individuals at random and replacement according to the inverse of their fitness values.
- Selection of both parents and replacement of according to fitness or inverse fitness.

Thank You!!

IIT KHARAGPUR

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Introduction to Soft Computing

GA Operators : Crossover-I

Prof. Debasis Samanta

Department of Computer Science & Engineering

IIT Kharagpur

Important GA Operations

- ✓ Encoding
- ✓ Fitness evaluation and Selection
- Reproduction
  - Crossover
  - Mutation
  - Inversion
  - Convergence test

Reproduction in Genetic Algorithm

Reproduction:

- **Crossover**
- Mutation
- Inversion

These genetic operators varies from one encoding scheme to another.

- Binary coded GAs
- Real-coded GAs
- Tree-coded GAs

Mating Pool: Prior to crossover operation

- A mating pair (each pair consists of two strings) are selected at random. Thus, if the size of mating pool is N, then $\frac{N}{2}$ mating pairs are to be formed. [Random Mating]
- The pairs are checked, whether they will participate in reproduction or not by tossing a coin, whose probability being p_c . If p_c is head, then the parent will participate in reproduction. Otherwise, they will remain intact in the population.

Note :

Generally, $p_c = 1.0$, so that almost all the parents can participate in production.

Crossover operation

Once, a pool of mating pair are selected, they undergo through crossover operations.

- In crossover, there is an exchange of properties between two parents and as a result of which **two** offspring solutions are produced.
- The crossover point(s) (also called k-point(s)) **is(are)** decided using a random number generator generating integer(s) in between 1 and L, where L is the length of the chromosome.
- Then we perform exchange of gene values with respect to the k-point(s)

There are many exchange mechanisms and hence crossover strategies.

Crossover Techniques in Binary Coded GA

Debasis Samanta
CSE
IIT Kharagpur

Crossover operations in Binary-coded GAs

There exists a large number of crossover schemes, few important of them are listed in the following.

- Single point crossover
- Two-point crossover
- Multi-point crossover (also called n-point crossover)
- Uniform crossover (UX)
- Half-uniform crossover (HUX)
- Shuffle crossover
- Matrix crossover (Tow-dimensional crossover)
- Three-parent crossover

Single Point Crossover

Single point crossover

1) Here, we select the K-point lying between 1 and L. Let it be k.
2) A single crossover point at k on both parent's strings is selected.
3) All data beyond that point in either string is swapped between the two parents.
4) The resulting strings are the chromosomes of the offspring produced.

Single point crossover: Illustration

Before Crossover

Parent 1 :	0	1	1	0	0	0	1	0
Parent 2 :	1	0	1	0	1	1	0	0

Two diploid from a mating pair

Crossover Point - k

Select crossover points randomly

Offspring 1:	0	1	1	0	1	1	0	0
Offspring 2:	1	0	1	0	0	0	1	0

Two diploid for two new offspring is produced

After Crossver

Two-Point Crossover

Two-point crossover

1) In this scheme, we select two different crossover points k_1 and k_2 lying between 1 and L at random such that $k_1 \neq k_2$.

2) The middle parts are swapped between the two strings.

3) Alternatively, left and right parts also can be swapped.

Two-point crossover: Illustration

Before Crossover

Parent 1 :	0	1	1	0	0	0	1	0
Parent 2 :	1	0	1	0	1	1	0	0

Crossover Point k_1

Crossover Point k_2

Select two crossover points randomly

Offspring 1:	0	1	1	0	1	0	1	0
Offspring 2:	1	0	1	0	0	1	0	0

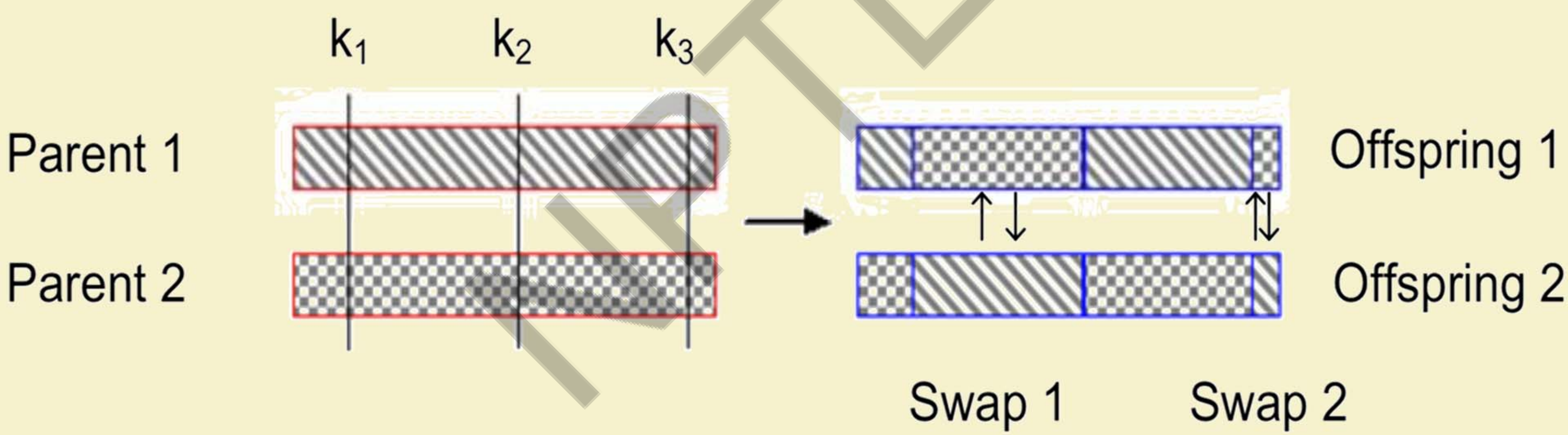
After Crossver

Multi-Point Crossover

Debasis Samanta
CSE
IIT Kharagpur

Multi-point crossover

1) In case of multi-point crossover, a number of crossover points are selected along the length of the string, at random.
2) The bits lying between alternate pairs of sites are then swapped.



Uniform Crossover (UX)

Debasis Samanta
CSE
IIT Kharagpur

Uniform Crossover (UX)

- Uniform crossover is a more general version of the multi-point crossover.
- In this scheme, at each bit position of the parent string, we toss a coin (with a certain probability p_s) to determine whether there will be swap of the bits or not.
- The two bits are then swapped or remain unaltered, accordingly.

Uniform crossover (UX): Illustration

Before crossover

Parent 1 :	1	1	0	0	0	1	0	1	1	0	0	1
Parent 2 :	0	1	1	0	0	1	1	1	0	1	0	1
Coin tossing:	1	0	0	1	1	1	0	1	1	0	0	1

After crossover

Offspring 1:	1	1	1	0	0	1	1	1	1	1	0	1
Offspring 2:	0	1	0	0	0	1	0	1	0	0	0	1

Rule: If the toss is 0 than swap the bits between P1 and P2

Uniform crossover with crossover mask

- Here, each gene is created in the offspring by copying the corresponding gene from one or the other parent chosen according to a random generated binary crossover mask of the same length as the chromosome.
- Where there is a 1 in the mask, the gene is copied from the first parent.
- Where there is a 0 in the mask, the gene is copied from the second parent.
- The reverse is followed to create another offspring.

Uniform crossover with crossover mask: Illustration

Before Crossover

Parent 1 :	1	1	0	0	0	1	0	1
Parent 2 :	0	1	1	0	0	1	1	1
Mask	1	0	0	1	1	1	0	1
Offspring 1:	1	1	1	0	0	1	1	1
Offspring 2:	0	1	0	0	0	1	0	1

Offspring 1: When there is a 1 in the mask, the gene is copied from Parent 1 else from Parent 2.

Offspring 2: When there is a 1 in the mask, the gene is copied from Parent 2 else from Parent 1.

After Crossver

Half-Uniform Crossover (HUX)

Half-uniform crossover (HUX)

- In the half uniform crossover scheme, exactly half of the **non-matching bits** are swapped.

1) Calculate the Hamming distance (the number of differing bits) between the given parents.
2) This number is then divided by two.
3) The resulting number is how many of the bits that do not match between the two parents will be swapped but probabilistically.
4) Choose the locations of these half numbers (with some strategies, say coin tossing) and swap them.

Half-uniform crossover: Illustration

Before crossover

Parent 1 :	1	1	0	0	0	0	1	0
Parent 2 :	1	0	0	1	1	0	1	1
Tossing:		1		0	1			1
Offspring 1:	1	0	0	0	1	0	1	1
Offspring 2:	1	1	0	1	0	0	1	0

After crossver

Here, Hamming distance is 4

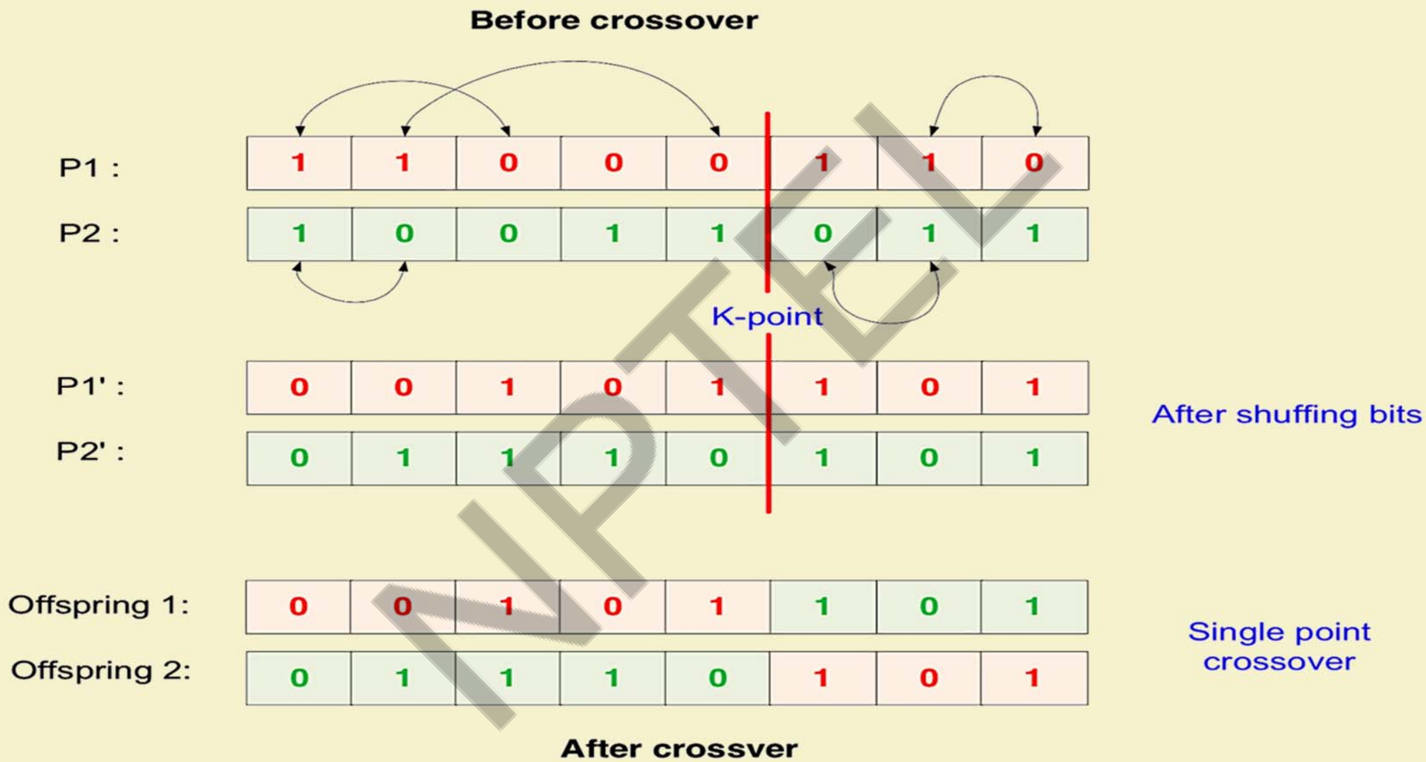
If toss is 1, then swap the bits else remain as it is

Shuffle Crossover

Shuffle crossover

- A single crossover point is selected. It divides a chromosome into two parts called **schema**.
- In both parents, genes are shuffled in each schema. Follow some strategy for shuffling bits.
- Schemas are exchanged to create offspring (as in single-point crossover)

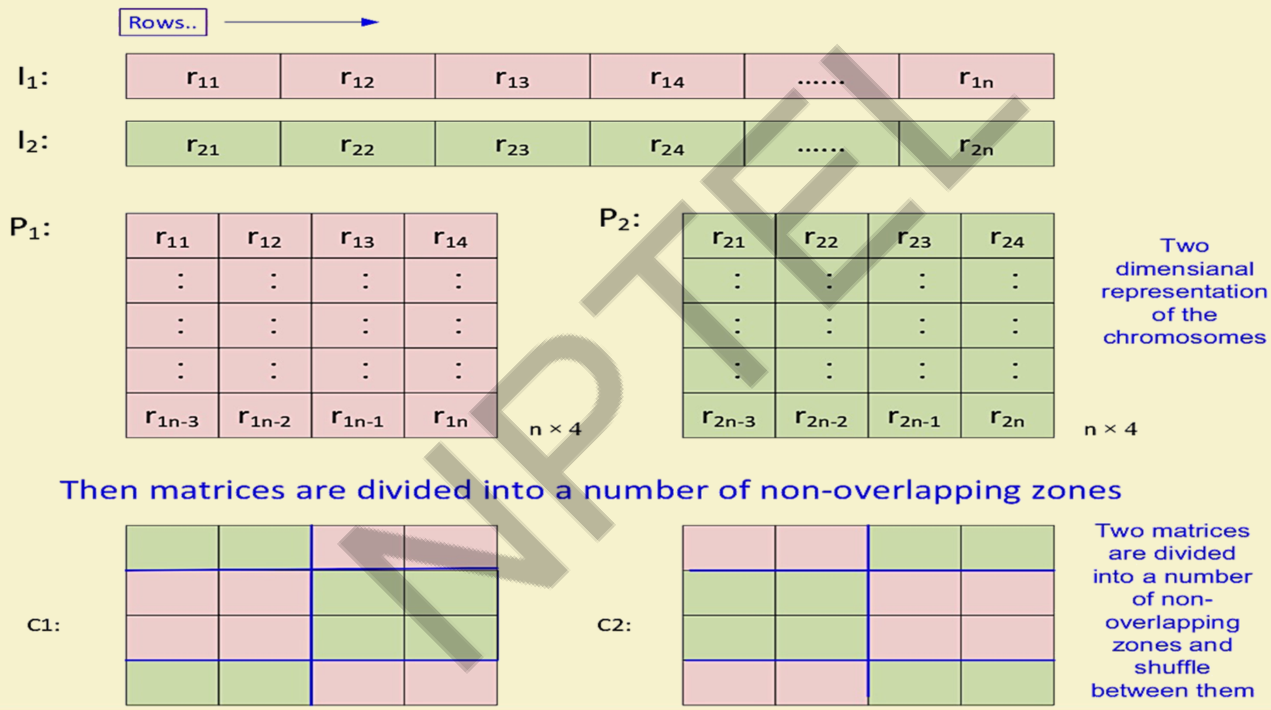
Shuffle crossover: Illustration



Matrix Crossover

Matrix crossover

The matrix crossover strategy is explained with the following illustration.



Three-Parent Crossover

Three parent crossover

- In this technique, three parents are randomly chosen from the current population.
- Each bit of the first parent is compared with the bit of the second parent.
- If both are the same, the bit is taken for the offspring.
- Otherwise, the bit from the third parent is taken for the offspring.

Three parent crossover: Illustration

P1:	1	1	0	1	0	0	0	1
P2:	0	1	1	0	1	0	0	1
P3:	0	1	1	0	1	1	0	1

C1:	0	1	1	0	1	0	0	1

Note: Sometime, the third parent can be taken as the crossover mask.

Comments on the binary crossover techniques

- **Non-uniform variation:** It cannot combine all possible schemas (i.e. building blocks)

  For example : it cannot in general combine instances of

  1 1 * * * * * 1

  and

  *** * * * 1 1 * ***

  to form an instance of

  1 1 * * 1 1 * 1.

- **Positional bias:** The schemas that can be created or destroyed by a crossover depends strongly on the location of the bits in the chromosomes.

Comments on the binary crossover techniques

- **End-point bias:** It is also observed that single-point crossover treats some loci preferentially, that is, the segments exchanged between the two parents always contain the end points of the strings.
- **Hamming cliff problem:**
  - A one-bit change can make a large (or a small) jump.
  - A multi-bits can make a small (or a large gap).

For example, **1000** ⇒ **0111**
(Here, Hamming distance = 4, but distance between phenotype is 1)

Similarly, **0000** ⇒ **1000**
(Here, Hamming distance = 1, but distance between phenotype is 8)

Comments on the binary crossover techniques

- To reduce the positional bias and end-point bias, two-point crossover and multi-point crossover schemes have been evolved.
- In contrast, UX and HUX distribute the patterns in parent chromosomes largely resulting too much deflections in the offspring.
- To avoid binary code related problem, **gray coding** can be used.
- In summary, binary coding is the simplest encoding and its crossover techniques are fastest compared to the crossover techniques in other GA encoding schemes.

Thank You!!