Soft Computing

Prof. Snehalika

School of Electrical Engineering
KIIT Deemed to be University, Bhubaneswar - 751024

Module 4
Optimization

Chapter 14

Genetic Algorithm

Background of Genetic Algorithm

Firs time introduced by Prof. John Holland (of Michigan University, USA, 1965). But, the first article on GA was published in 1975.

Principles of GA based on two fundamental biological processes:

-
- **Genetics:** Gregor Johan Mendel (1865)

  Evolution: Charles Darwin (1875)

Limitations of the traditional optimization approaches

Limitations:

- Computationally expensive.
- For a discontinuous objective function, methods may fail. Method
- may not be suitable for parallel computing.
- Discrete (integer) variables are difficult to handle. Methods
- may not necessarily adaptive.

Evolutionary algorithms have been evolved to address the above mentioned limitations of solving optimization problems with traditional approaches.

Evolutionary Algorithms

The algorithms, which follow some biological and physical behaviors:

Biologic behaviors:

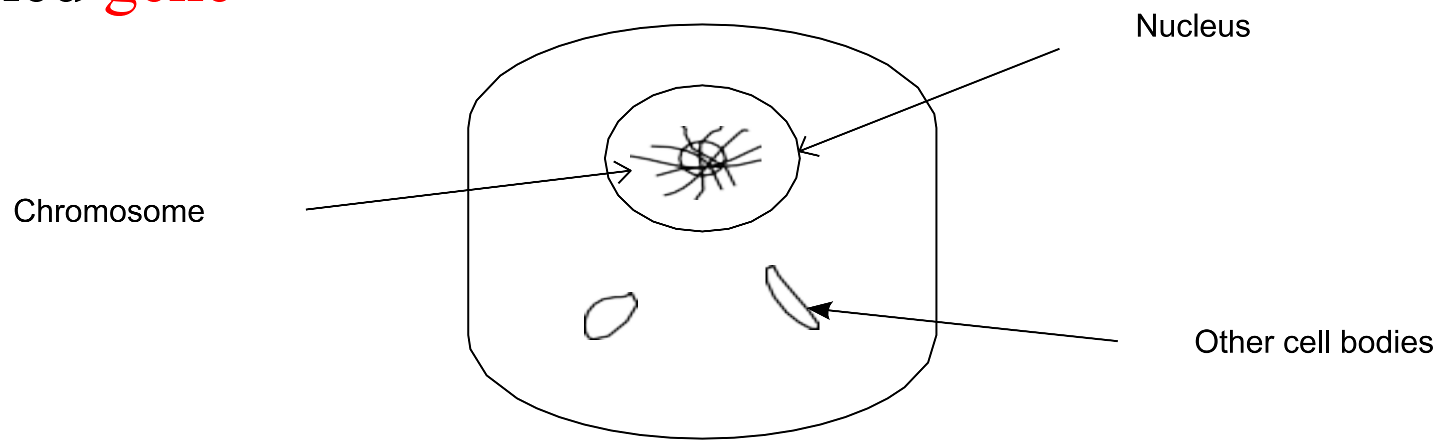
- Genetics and Evolution –> Genetic Algorithms (GA)
- Behavior of ant colony –> Ant Colony Optimization (ACO)
- Human nervous system –> Artificial Neural Network (ANN)

Physical behaviors:

- Annealing process –> Simulated Annealing (SA)
- Swarming of particle –> Particle Swarming Optimization (PSO)
- Learning –> Fuzzy Logic (FL)

A brief account on genetics

The basic building blocks in living bodies are cells. Each cell carries the basic unit of heredity, called gene



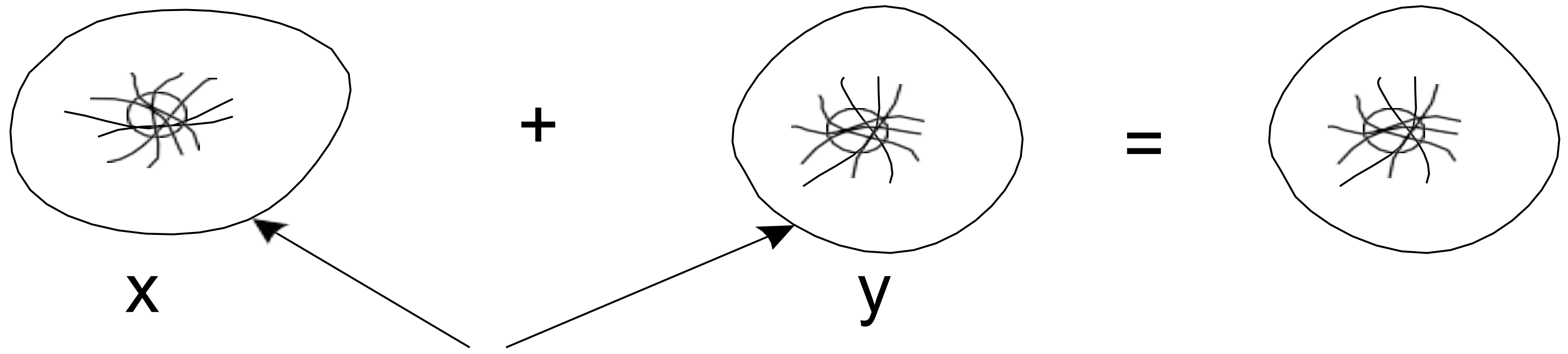
- For a particular species, number of chromosomes is fixed.

Examples

- Mosquito: 6
- Frogs: 26
- Human: 46
- Goldfish: 94 etc.

A brief account on genetics

Reproduction



Organism's cell :
Cell division

gamete

diploid

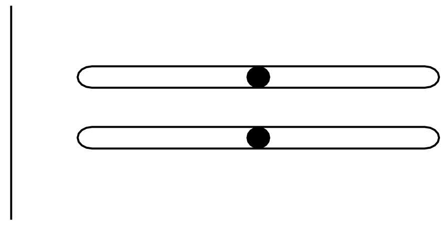
haploid (Reproductive cell has half the number of chromosomes)

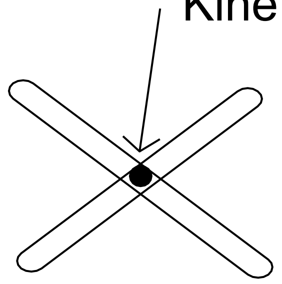
diploid Each chromosome from both haploids are combined to have full numbers

A brief account on genetics

Cross-Over

Information from two different organism's body cells





Random crossover points makes infinite diversities

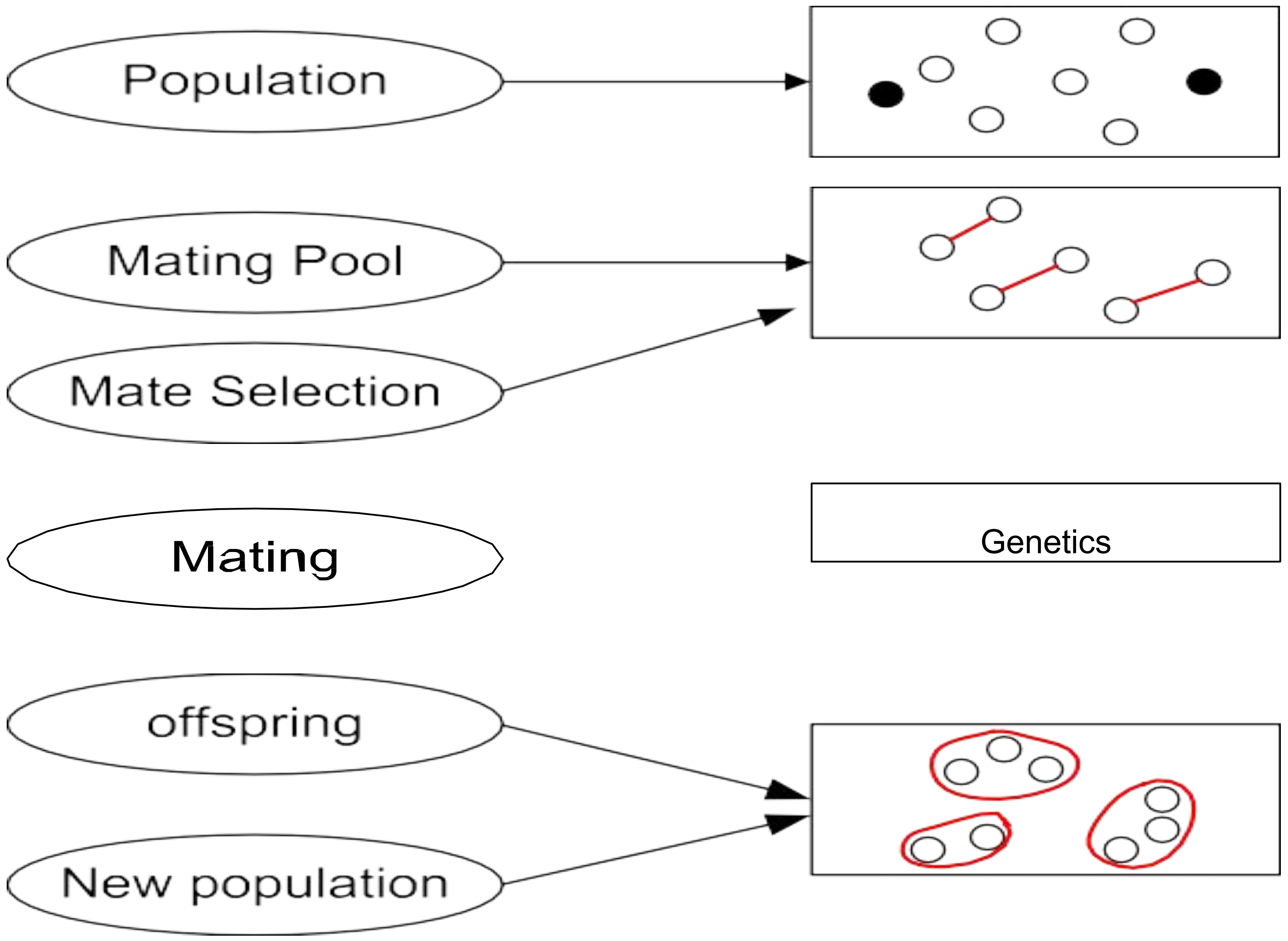
Combined into so that diversity in information is possible

A brief account on evolution

Evolution : Natural Selection

Four primary premises:

1. ***Information propagation:*** An offspring has many of its characteristics of its parents (i.e. information passes from parent to its offspring). **[Heredity]**
2. ***Population diversity:*** Variation in characteristics in the next generation. **[Diversity]**
3. ***Survival for exitence:*** Only a small percentage of the offspring produced survive to adulthood. **[Selection]**
4. ***Survival of the best:*** Offspring survived depends on their inherited characteristics. **[Ranking]**

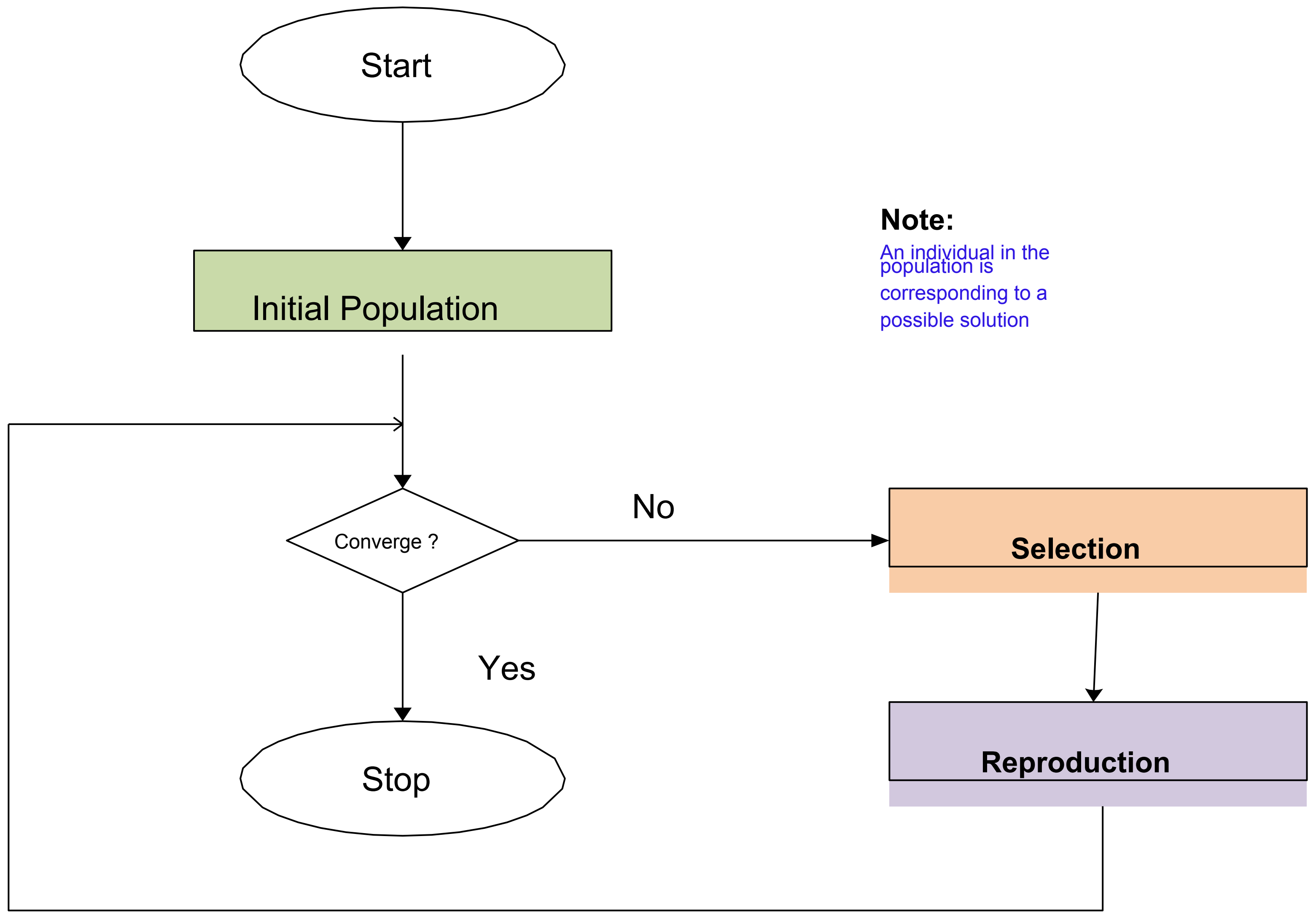

Population
Mating Pool
Mate Selection
Mating
Genetics
offspring
New population

Working of Genetic Algorithm

Definition of GA:

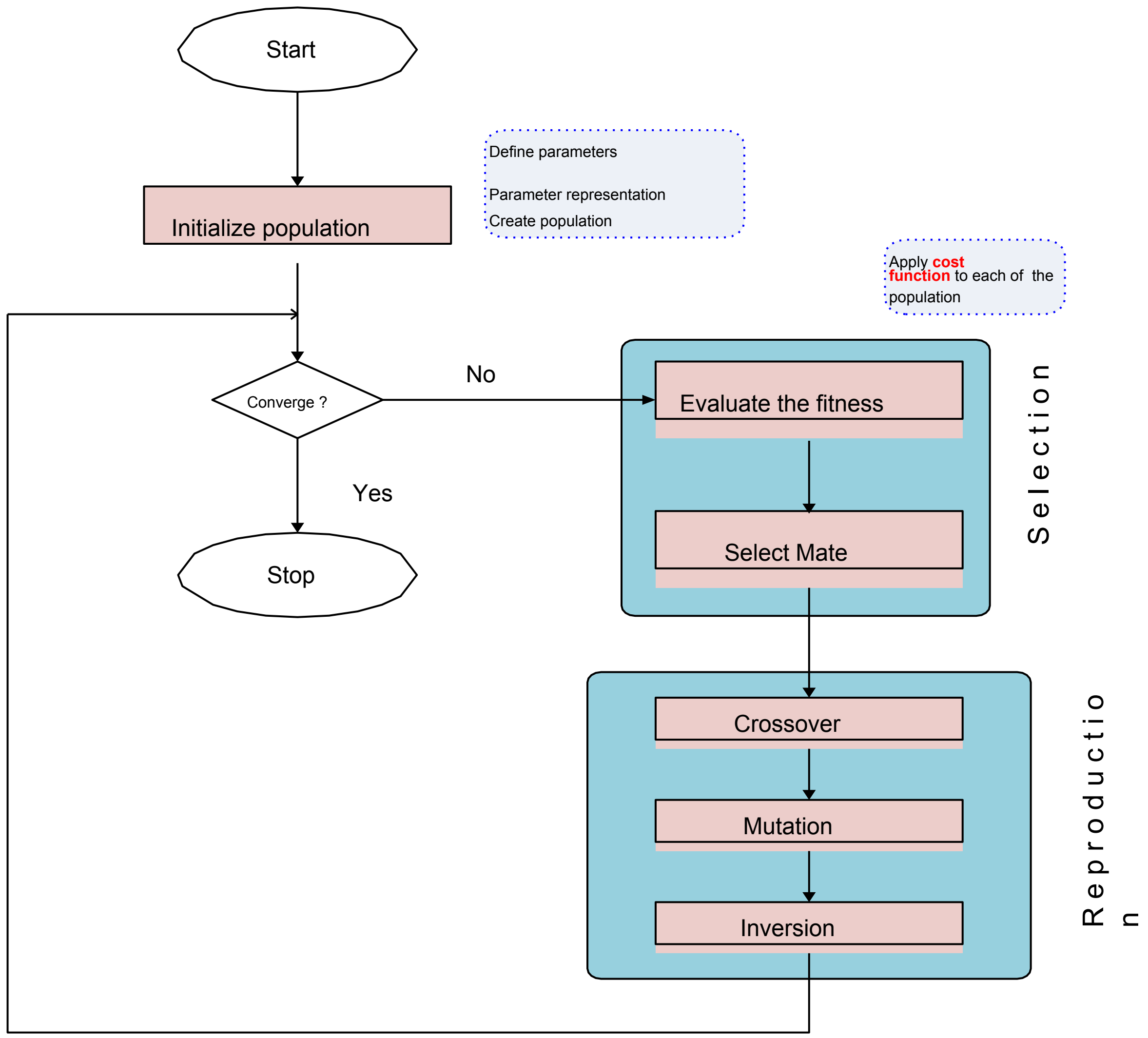
Genetic algorithm is a population-based probabilistic search and optimization techniques, which works based on the mechanisms of *natural genetics* and *natural evoluation*.

Framework of GA



Note:
An individual in the population is corresponding to a possible solution

Framework of GA: A detail view



Working of Genetic Algorithm

Note:

1. GA is an iterative process. It is a
2. searching technique.
3. Working cycle with / without convergence.
4. Solution is not necessarily guranteed. Usually, terminated with a local optima.

Optimization problem solving with GA

For the optimization problem, identify the following:

- Objective function(s)
- Constraint(s)
- Input parameters
- Fitness evaluation (it may be algorithm or mathematical formula)
- Encoding
- Decoding

GA Operators

In fact, a GA implementation involved with the realization of the following operations.

Encoding: How to represent a solution to fit with GA framework.

1

Convergence: How to decide the termination criterion.

Mating pool: How to generate next solutions.

Fitness Evaluation: How to evaluate a solution.

Crossover: How to make the diverse set of next solutions.

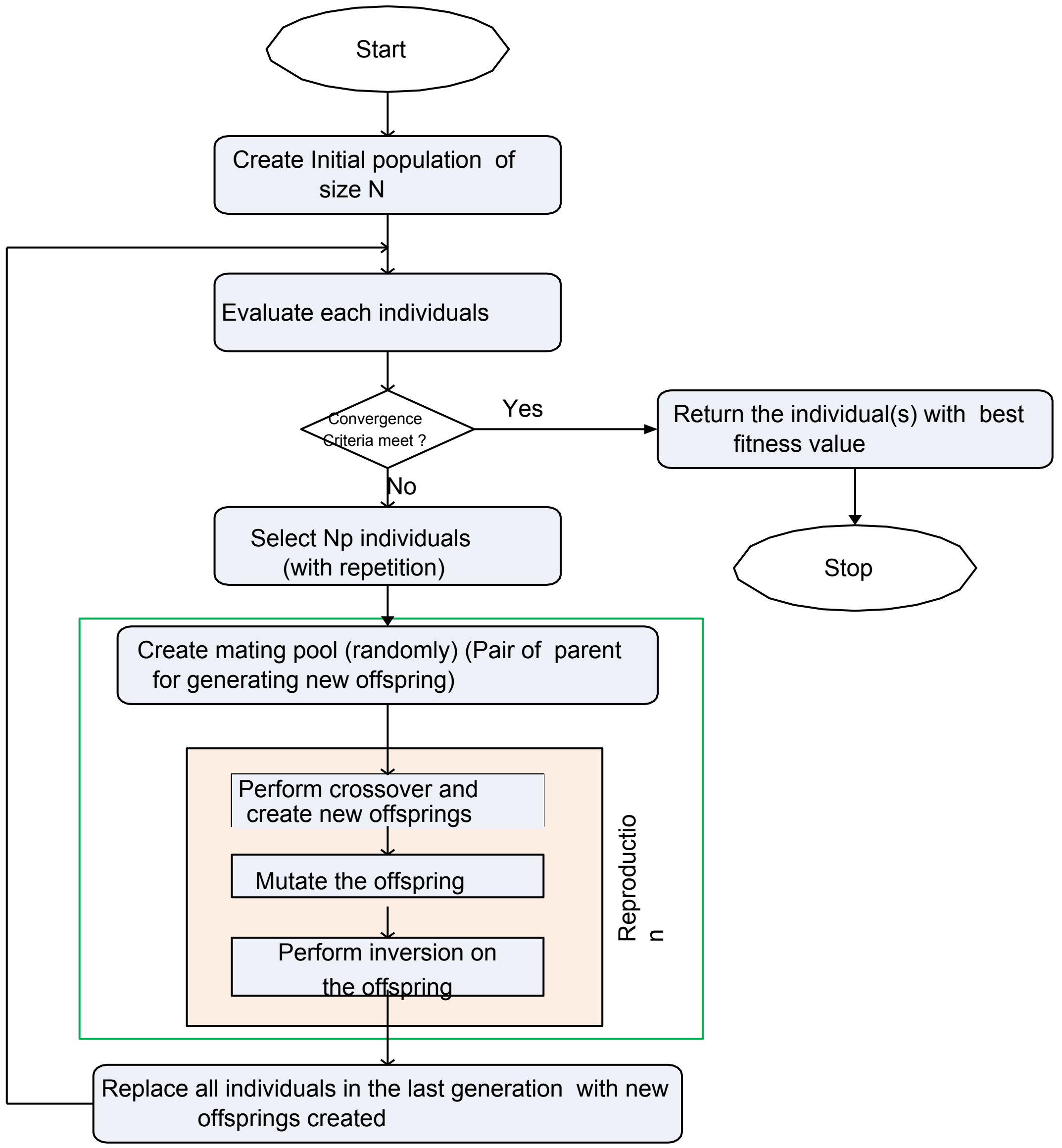
Mutation: To explore other solution(s).

Inversion: To move from one optima to other.

Different GA Strategies

- Simple Genetic Algorithm (SGA)
- Steady State Genetic Algorithm (SSGA) Messy
- Genetic Algorithm (MGA)

Simple GA


Start
Create Initial population of size N
Evaluate each individuals
Convergence Criteria meet ?
Yes
Return the individual(s) with best fitness value
Stop
No
Select Np individuals (with repetition)
Create mating pool (randomly) (Pair of parent for generating new offspring)
Perform crossover and create new offsprings
Mutate the offspring
Perform inversion on the offspring
Reproduction
Replace all individuals in the last generation with new offsprings created

Important parameters involved in Simple GA

SGA Parameters

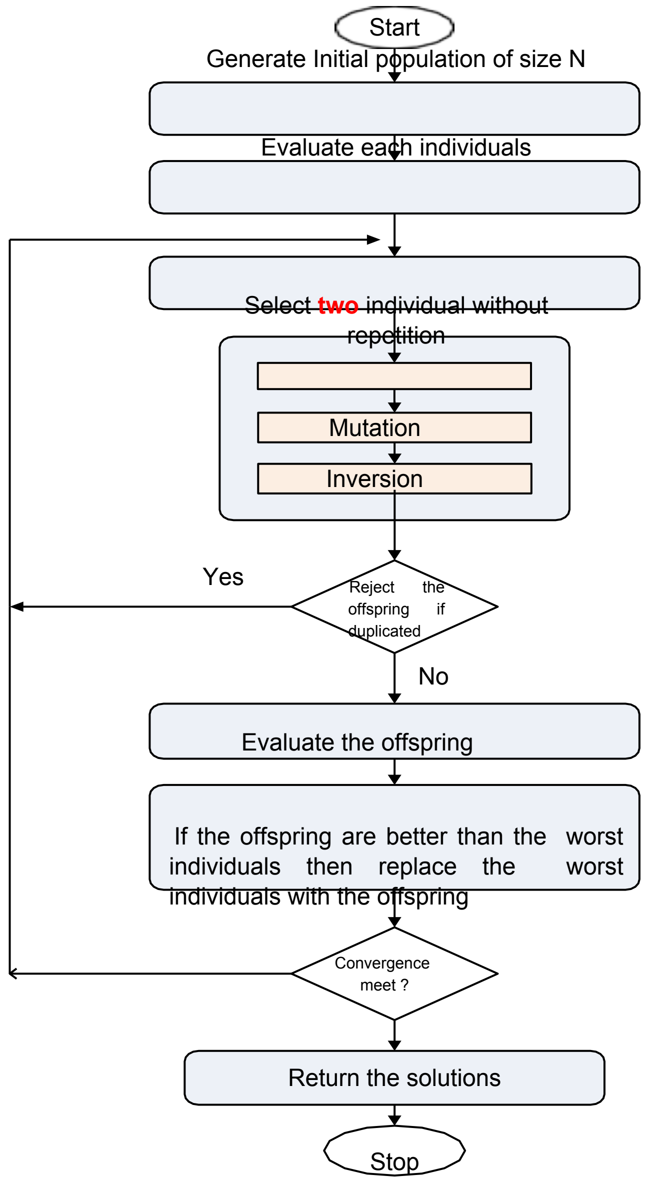
- Initial population size : N
- Size of mating pool, N_p : $N_p = p\% of N$
- Convergence threshold δ
- Mutation μ
- Inversion η
- Crossover ρ

Salient features in SGA

Simple GA features:

- Have overlapping generation (Only fraction of individuals are replaced).
- Computationally expensive.
- Good when initial population size is large. In general,
- gives better results.
- Selection is biased toward more highly fit individuals; Hence, the average fitness (of overall population) is expected to increase in succession.
- The best individual may appear in any iteration.

Steady State Genetic Algorithm (SSGA)


Start
Generate Initial population of size N
Evaluate each individuals
Select two individual without repetition
Mutation
Inversion
Reject the offspring if duplicated
Yes
No
Evaluate the offspring
If the offspring are better than the worst individuals then replace the worst individuals with the offspring
Convergence meet ?
Return the solutions
Stop

Salient features in Steady-state GA

SGA Features:

- Generation gap is small.
  Only two offspring are produced in one generation.
- It is applicable when
  - Population size is small
  - Chromosomes are of longer length
  - Evaluation operation is less computationally expensive (compare to duplicate checking)

Salient features in Steady-state GA

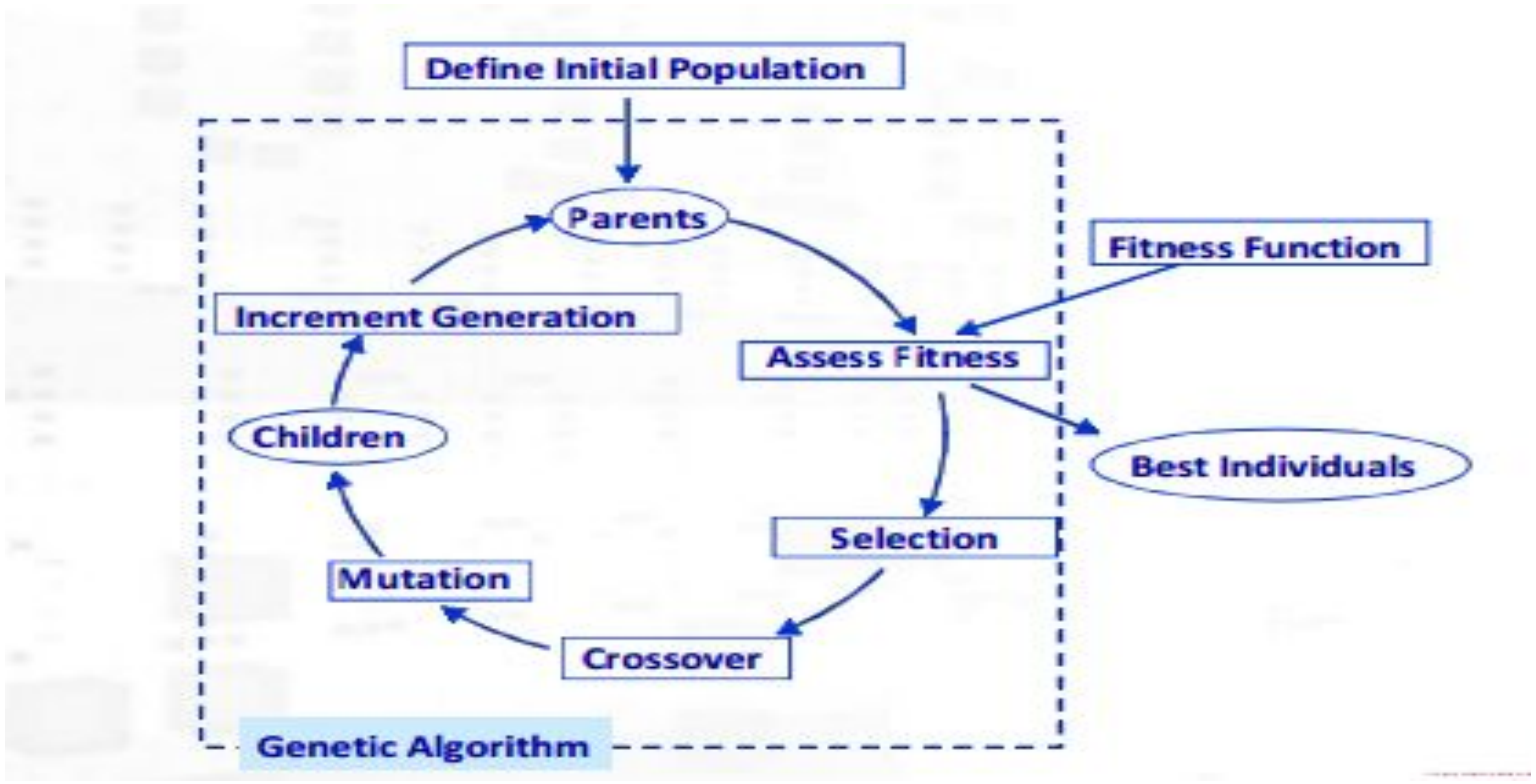
Limitations in SSGA:

- There is a chance of stuck at local optima, if crossover/mutation/inversion is not strong enough to diversify the population).
- Premature convergence may result.
- It is susceptible to stagnation. Inferiors are neglected or removed and keeps making more trials for very long period of time without any gain (i.e. long period of localized search).

The Genetic Algorithm (cont.)

- Provide efficient, effective techniques for optimization and machine learning applications
- Widely-used today in business, scientific and engineering circles

Genetic Algorithm: A Nature Inspired Technique



Simple Genetic Algorithm

```
{
    initialize population;
    evaluate population;
    while TerminationCriteriaNotSatisfied
    {
        select parents for reproduction;
        perform recombination and mutation;
        evaluate population;
    }
}
```

Another form of GA

„ **Initialization:** Start initial population of solutions, e.g.

- randomly. Evaluate the fitness for each individual.
- **Iteration:**
  1. Select some members of population to become parents.
  2. Cross genetic material of parents in a crossover operation. Mutation can occur in some genes.
  3. Take care of infeasible solutions, by making them feasible.
  4. Evaluate fitness of new members.
- „ **Stopping rule:** stop using fixed number of iterations, fixed number of iterations without improvement, etc.

Selection

- Select pairs randomly
- Fitter individual wins

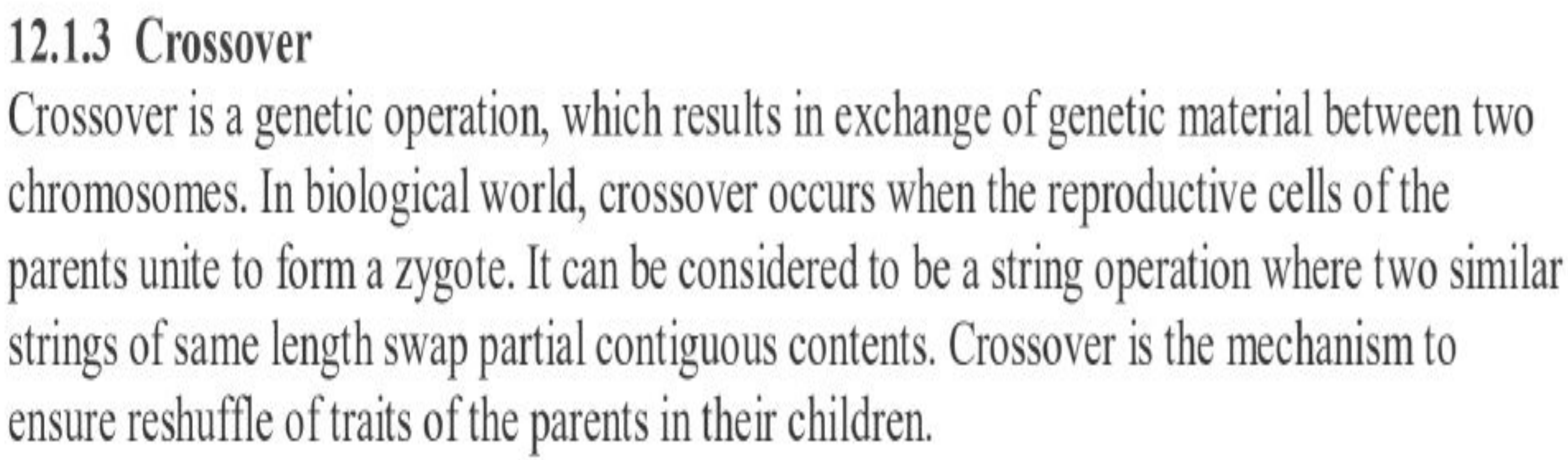
□ deterministic

probabilistic

constant probability of winning

„ probability of winning depends on fitness

Crossover

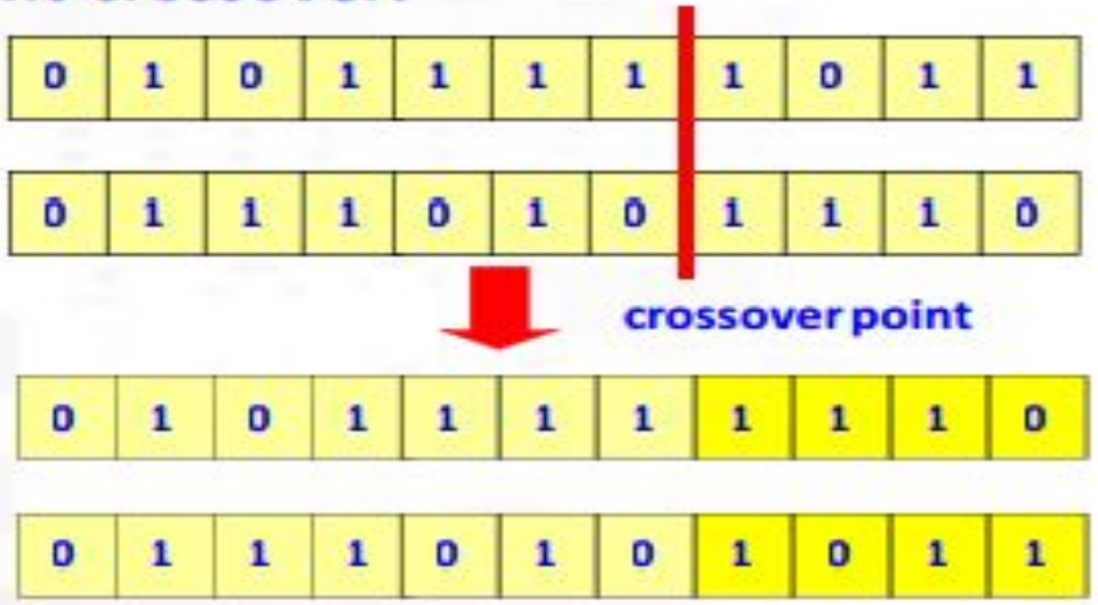


12.1.3 Crossover

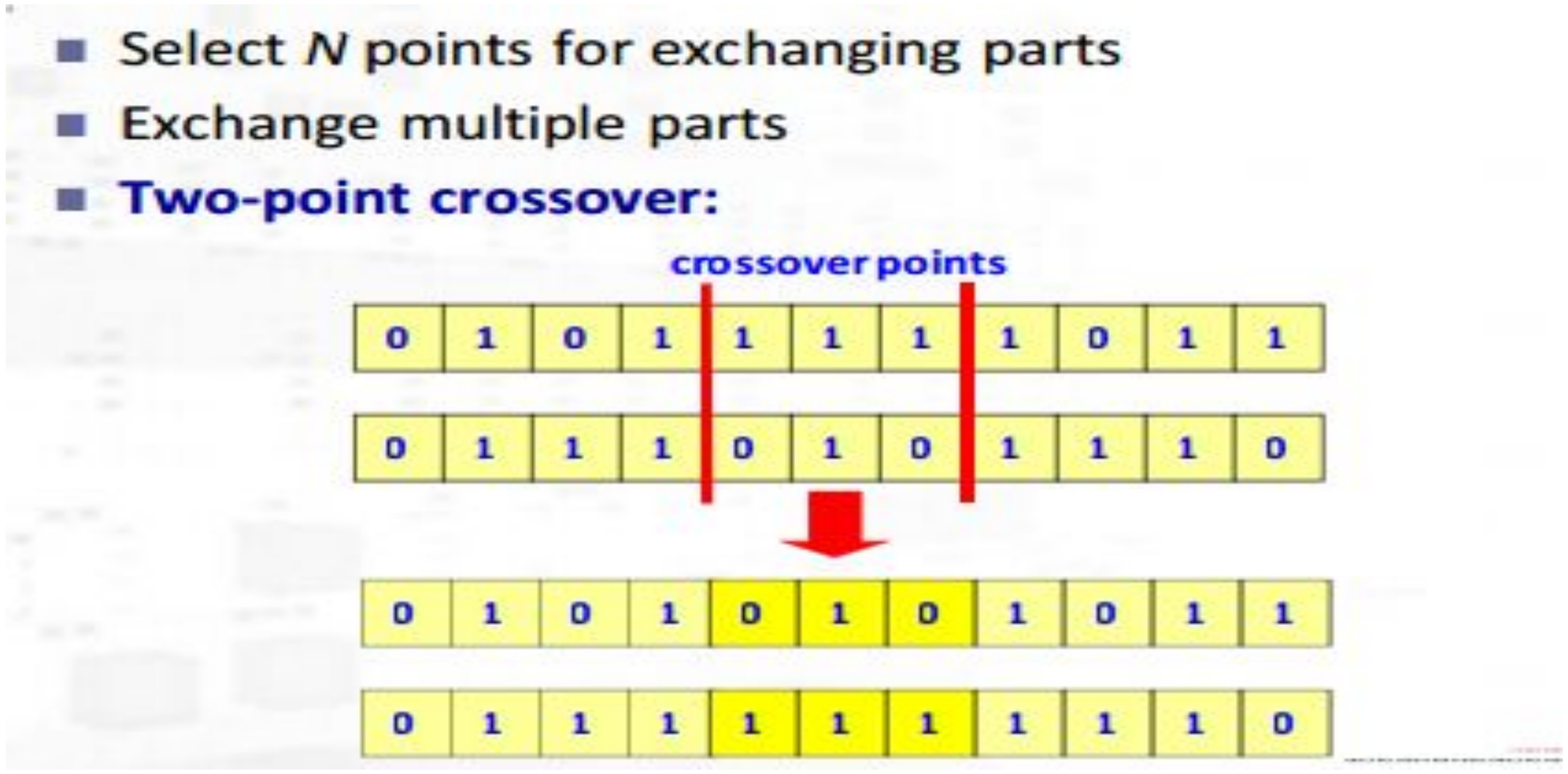
Crossover is a genetic operation, which results in exchange of genetic material between two chromosomes. In biological world, crossover occurs when the reproductive cells of the parents unite to form a zygote. It can be considered to be a string operation where two similar strings of same length swap partial contiguous contents. Crossover is the mechanism to ensure reshuffle of traits of the parents in their children.

Crossover

- Exchange parts of chromosome with a crossover probability (p_c is about 0.8)
- Select crossover points randomly
- **One-point crossover:**

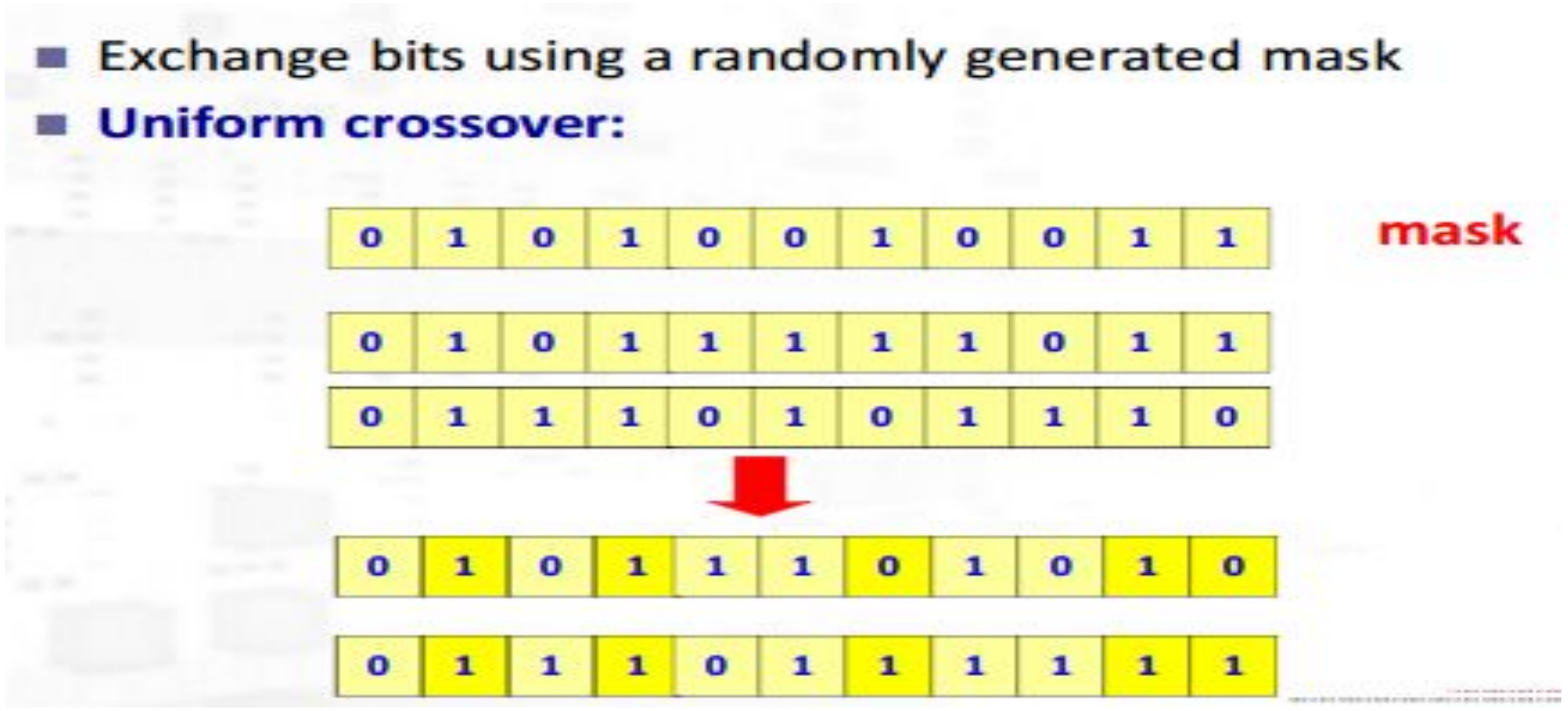


N-point Crossover



- Select *N* points for exchanging parts
- Exchange multiple parts
- **Two-point crossover:**

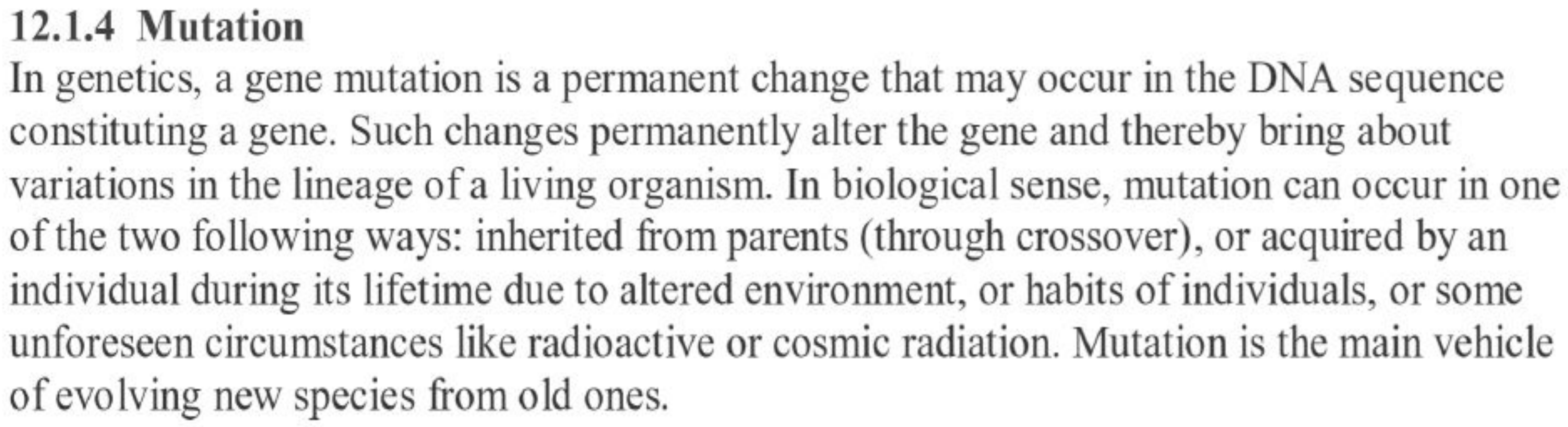
Uniform Crossover



- Exchange bits using a randomly generated mask
- **Uniform crossover:**

0	1	0	1	0	0	1	0	0	1	1	mask
0	1	0	1	1	1	1	1	0	1	1	
0	1	1	1	0	1	0	1	1	1	0	
0	1	0	1	1	1	0	1	0	1	0	
0	1	1	1	0	1	1	1	1	1	1	

Mutation

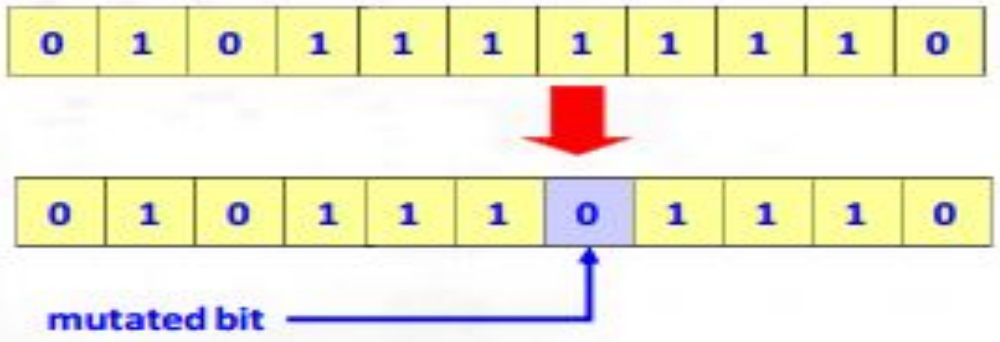


12.1.4 Mutation

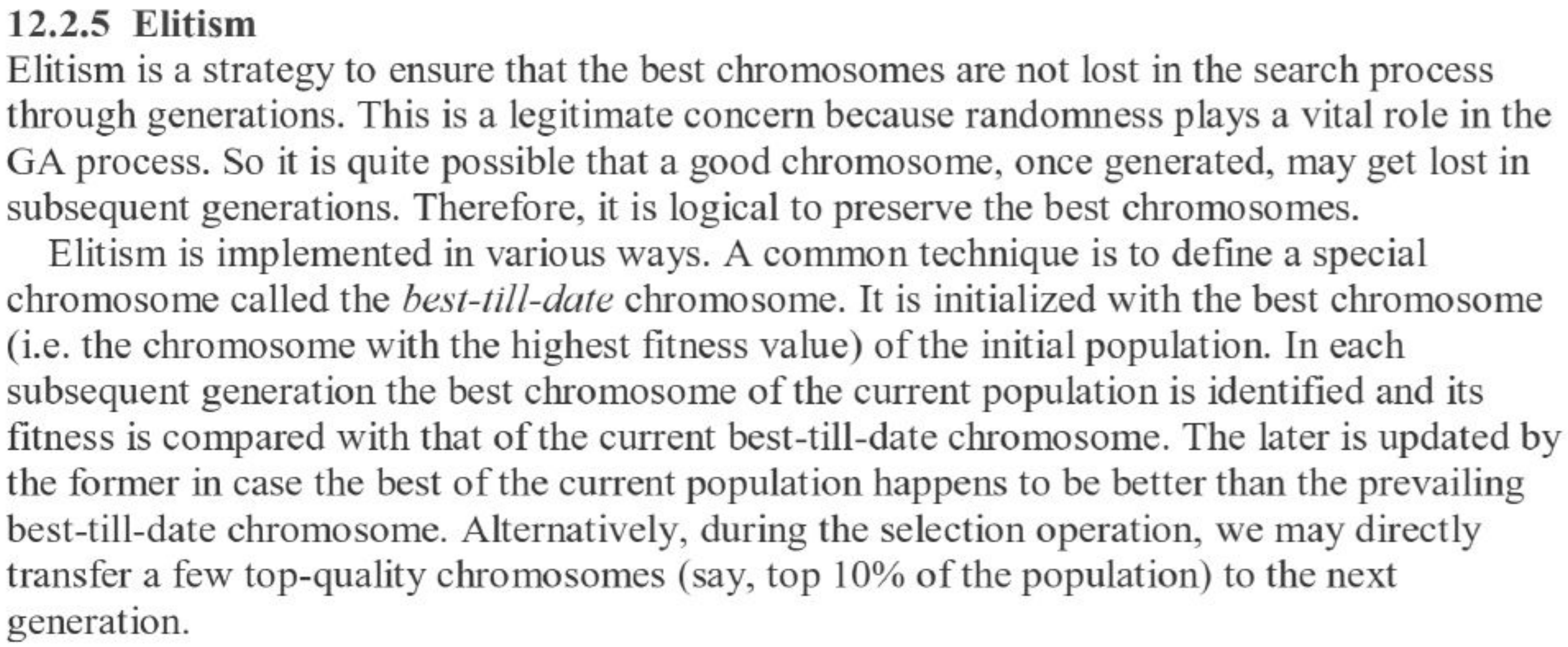
In genetics, a gene mutation is a permanent change that may occur in the DNA sequence constituting a gene. Such changes permanently alter the gene and thereby bring about variations in the lineage of a living organism. In biological sense, mutation can occur in one of the two following ways: inherited from parents (through crossover), or acquired by an individual during its lifetime due to altered environment, or habits of individuals, or some unforeseen circumstances like radioactive or cosmic radiation. Mutation is the main vehicle of evolving new species from old ones.

Mutation

- Crossover is used to search the solution space
- Mutation is needed to escape from local optima
- Introduces genetic diversity
- Mutation is rare (p_m is about 0.01)
- **Uniform mutation:**



Elitism

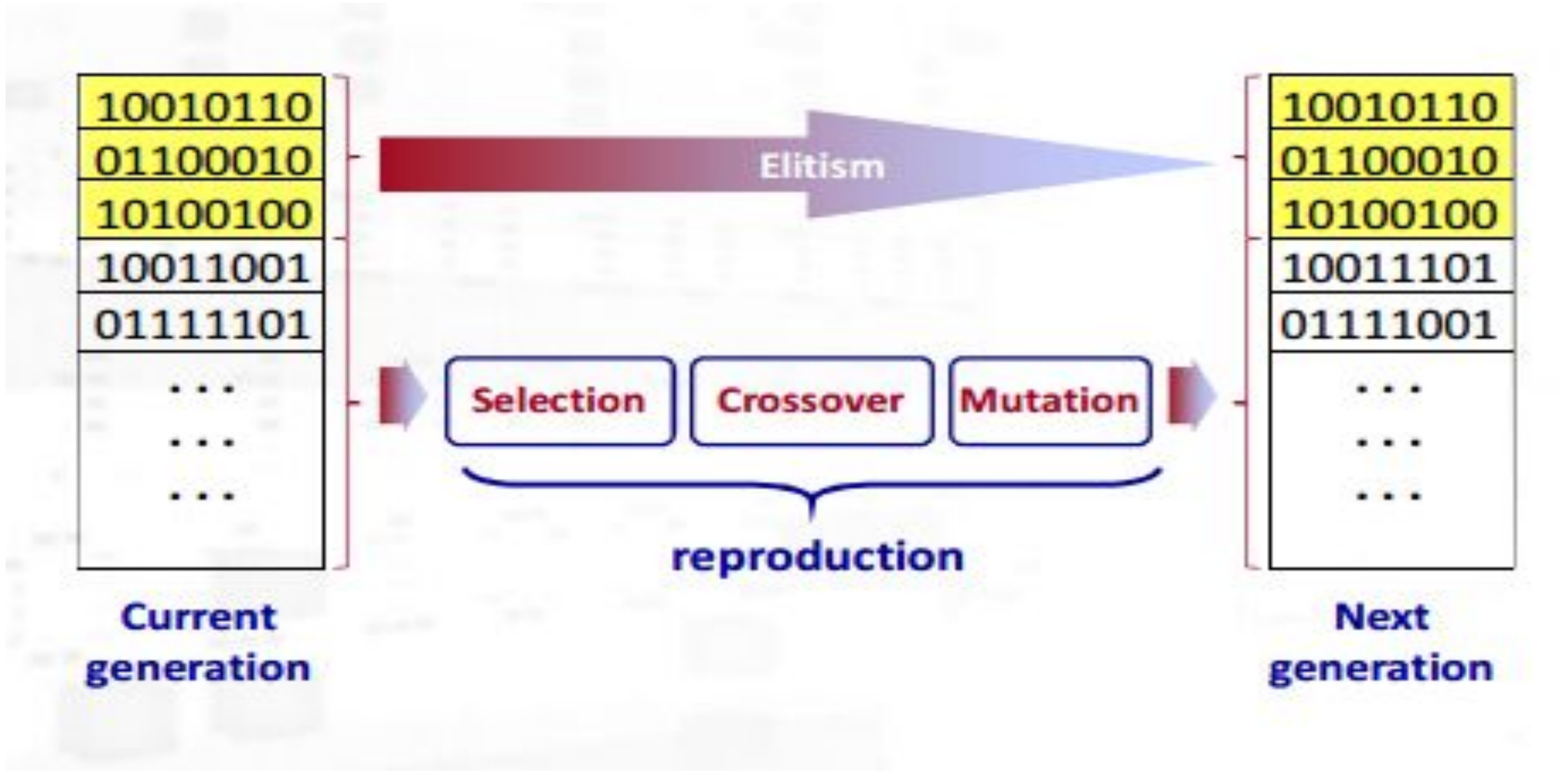


12.2.5 Elitism

Elitism is a strategy to ensure that the best chromosomes are not lost in the search process through generations. This is a legitimate concern because randomness plays a vital role in the GA process. So it is quite possible that a good chromosome, once generated, may get lost in subsequent generations. Therefore, it is logical to preserve the best chromosomes.

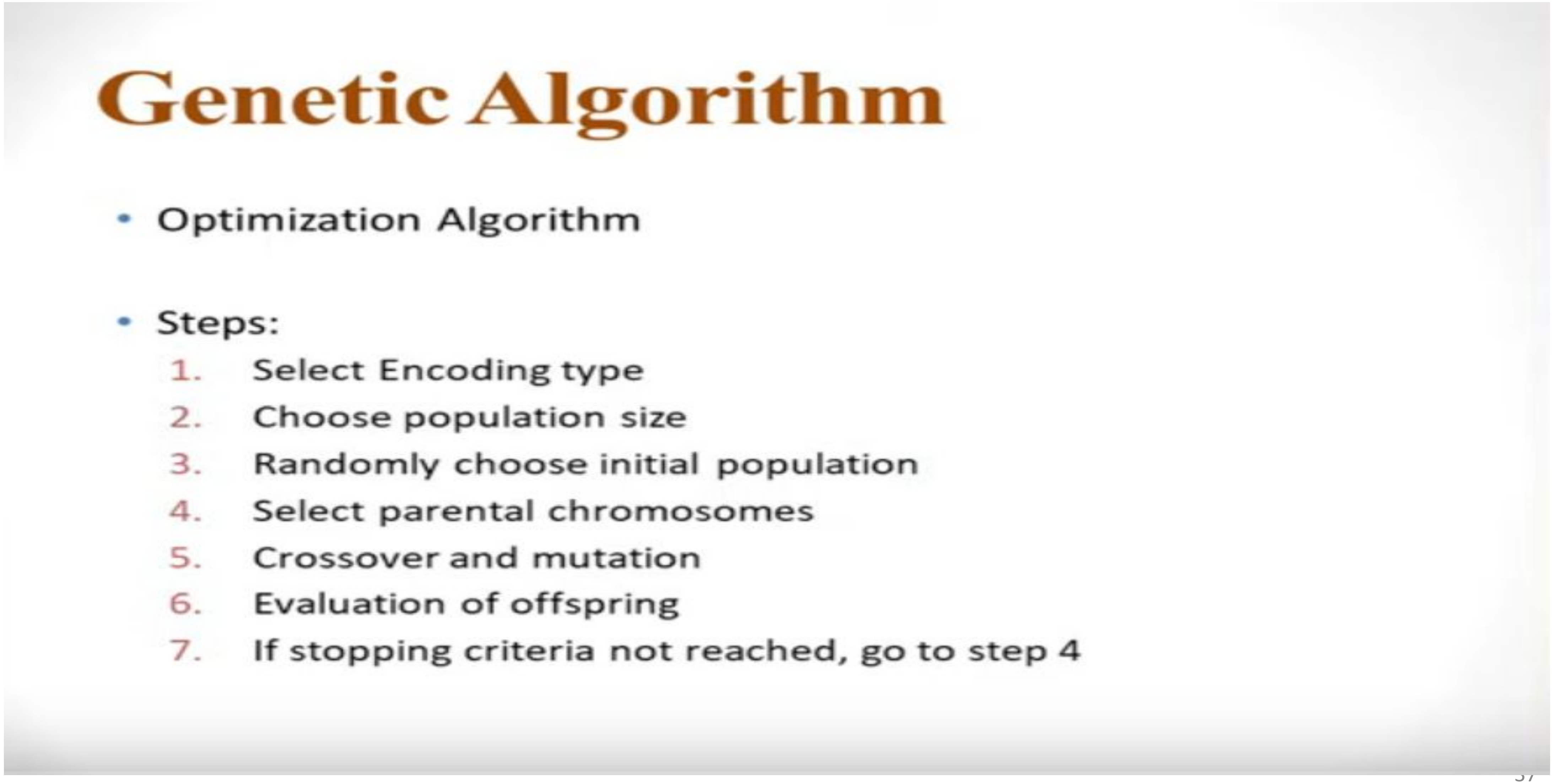
Elitism is implemented in various ways. A common technique is to define a special chromosome called the *best-till-date* chromosome. It is initialized with the best chromosome (i.e. the chromosome with the highest fitness value) of the initial population. In each subsequent generation the best chromosome of the current population is identified and its fitness is compared with that of the current best-till-date chromosome. The later is updated by the former in case the best of the current population happens to be better than the prevailing best-till-date chromosome. Alternatively, during the selection operation, we may directly transfer a few top-quality chromosomes (say, top 10% of the population) to the next generation.

Elitism



10010110
01100010
10100100
10011001
01111101
. . .
. . .
. . .
Current generation
Elitism
Selection
Crossover
Mutation
reproduction
10010110
01100010
10100100
10011101
01111001
. . .
. . .
. . .
Next generation

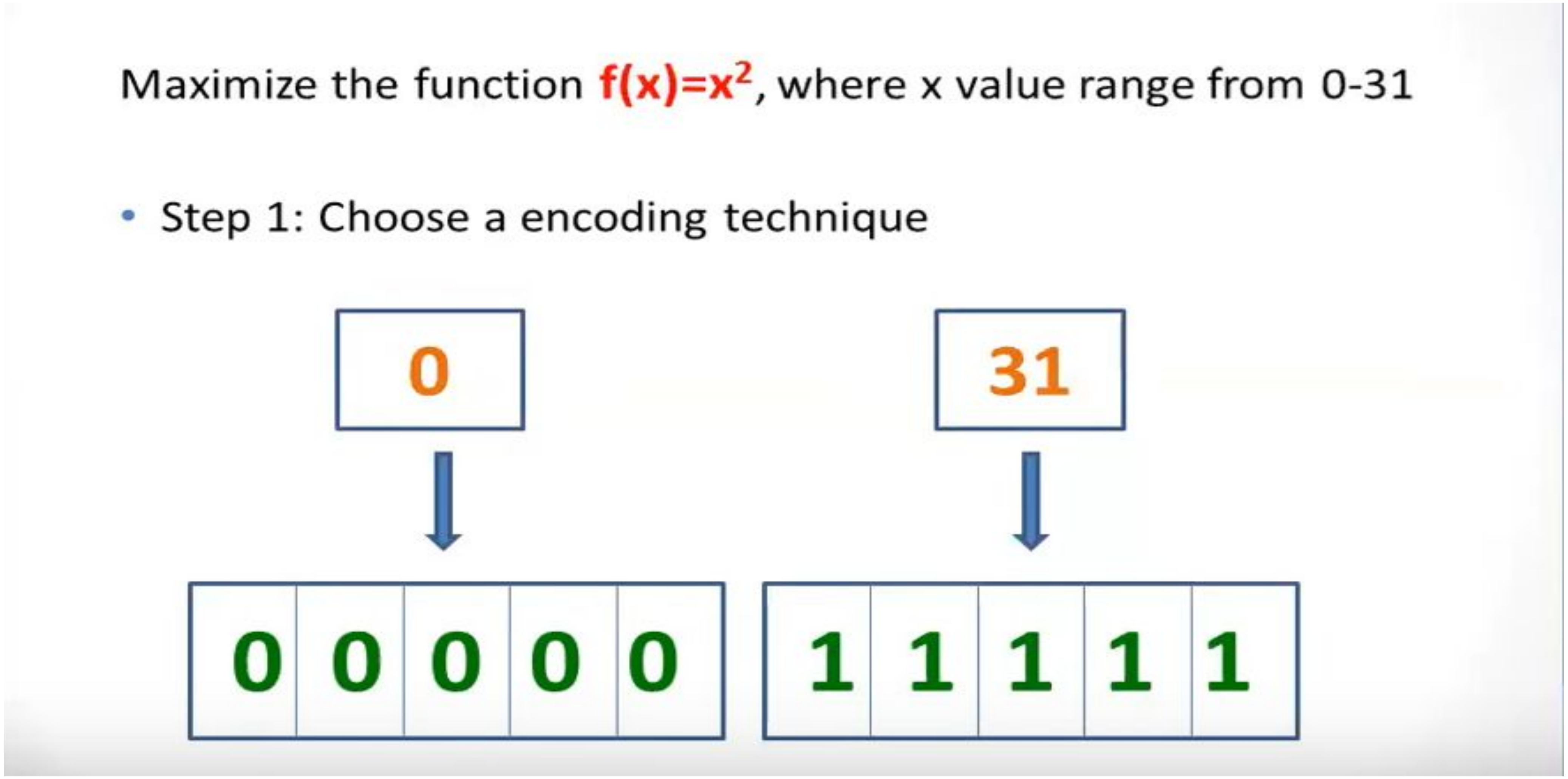
Steps of GA



Genetic Algorithm

- Optimization Algorithm
- Steps:
  1. Select Encoding type
  2. Choose population size
  3. Randomly choose initial population
  4. Select parental chromosomes
  5. Crossover and mutation
  6. Evaluation of offspring
  7. If stopping criteria not reached, go to step 4

Example of GA

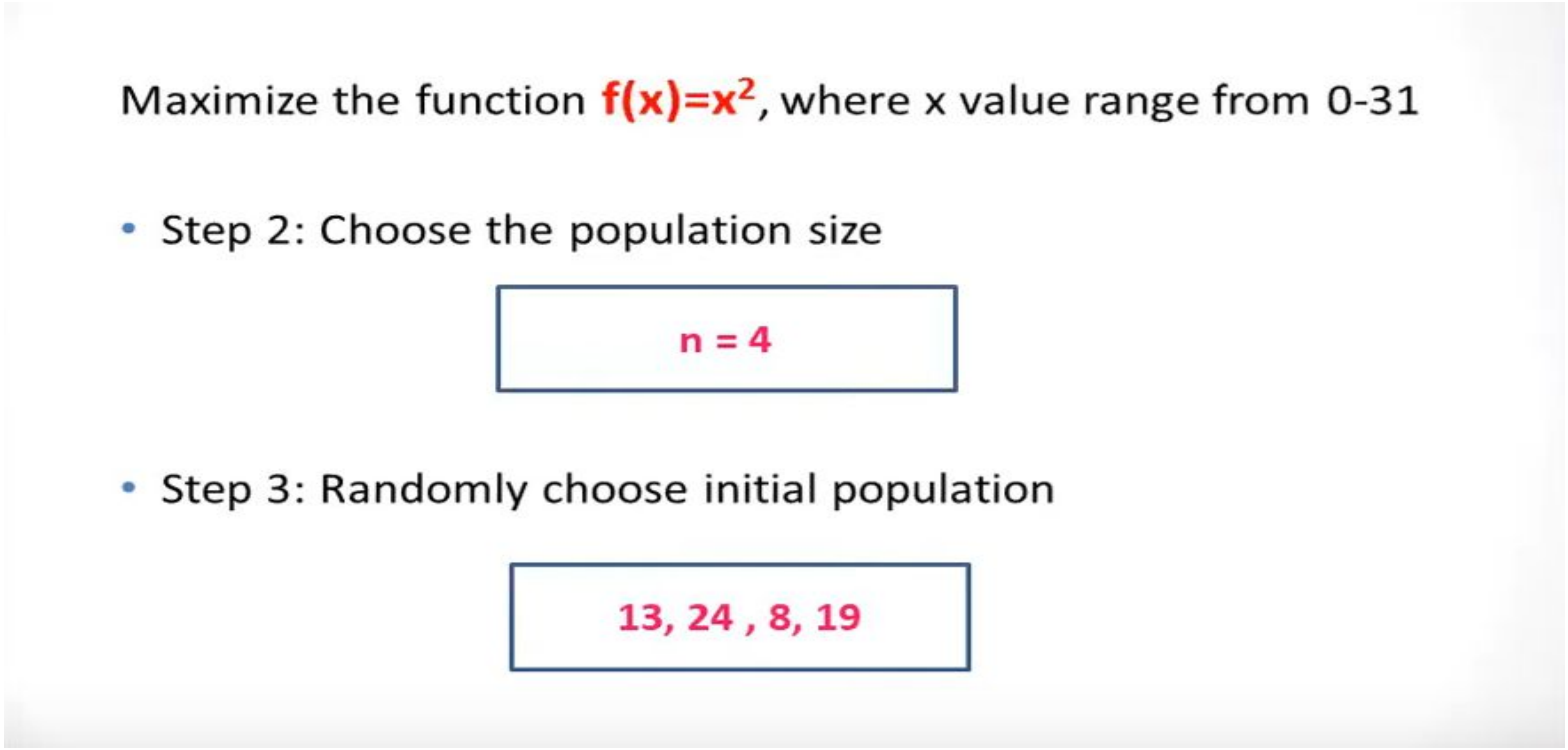


Maximize the function $f(x)=x^2$, where x value range from 0-31

- Step 1: Choose a encoding technique

0
31
0 0 0 0 0
1 1 1 1 1

Example of GA



Maximize the function $f(x)=x^2$, where x value range from 0-31

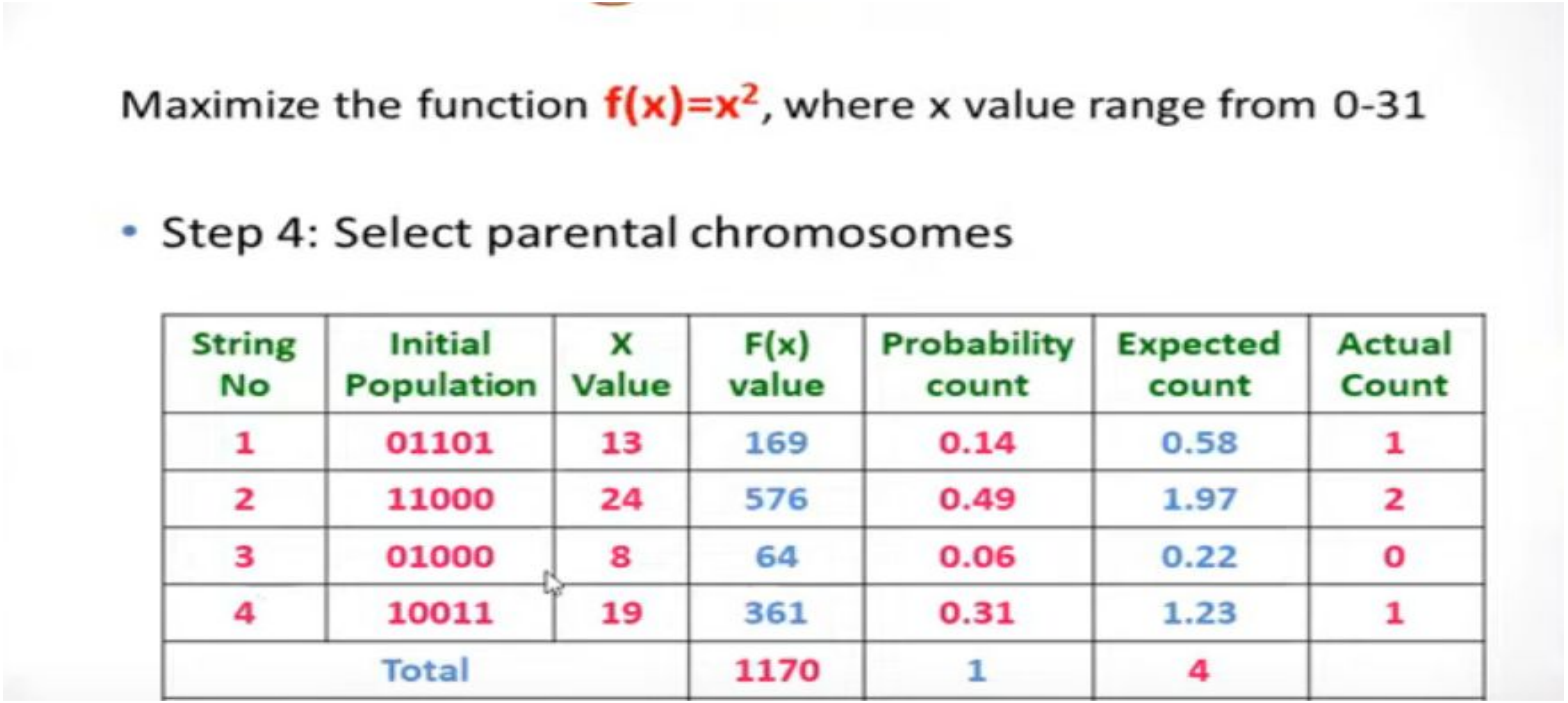
- Step 2: Choose the population size

$$\boxed{n = 4}$$

- Step 3: Randomly choose initial population

$$\boxed{13, 24, 8, 19}$$

Example of GA

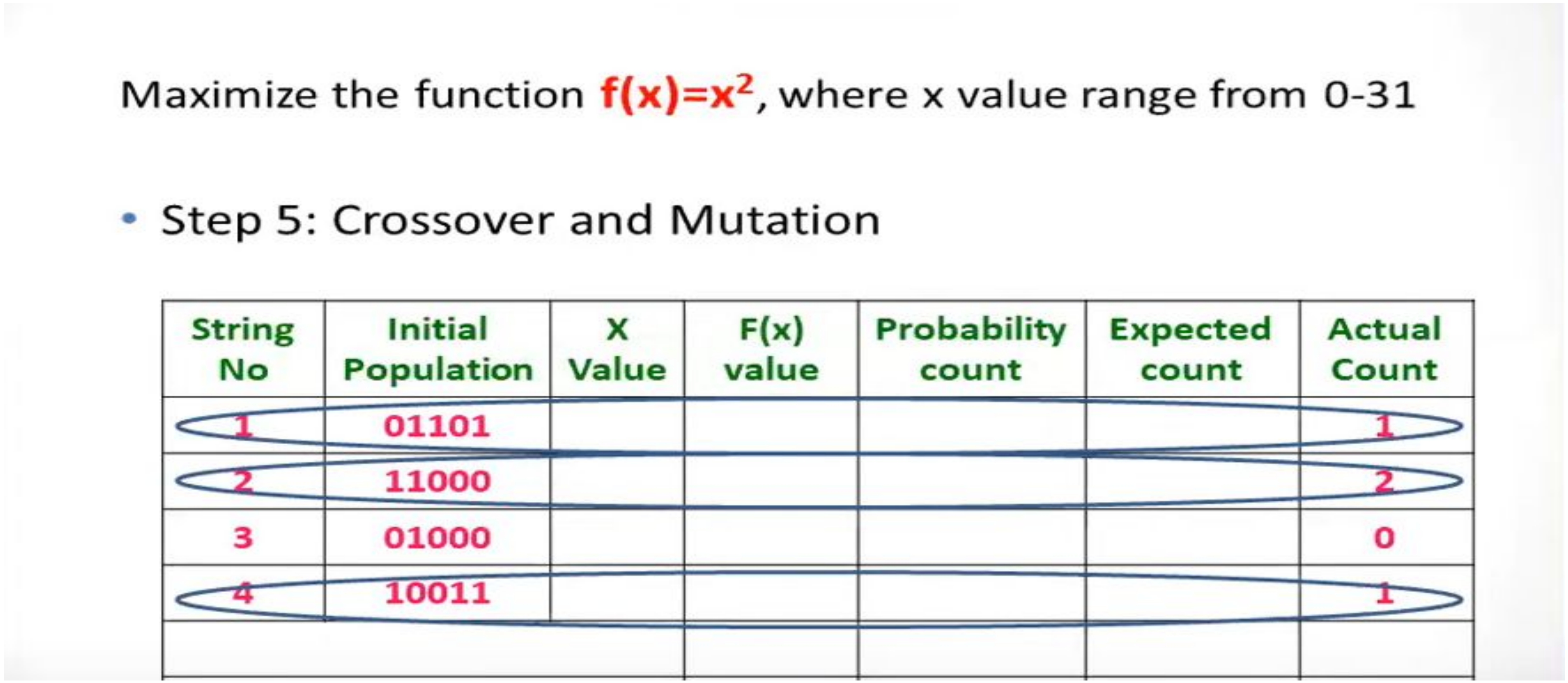


Maximize the function $f(x)=x^2$, where x value range from 0-31

- Step 4: Select parental chromosomes

String No	Initial Population	X Value	F(x) value	Probability count	Expected count	Actual Count
1	01101	13	169	0.14	0.58	1
2	11000	24	576	0.49	1.97	2
3	01000	8	64	0.06	0.22	0
4	10011	19	361	0.31	1.23	1
Total			1170	1	4	

Example of GA

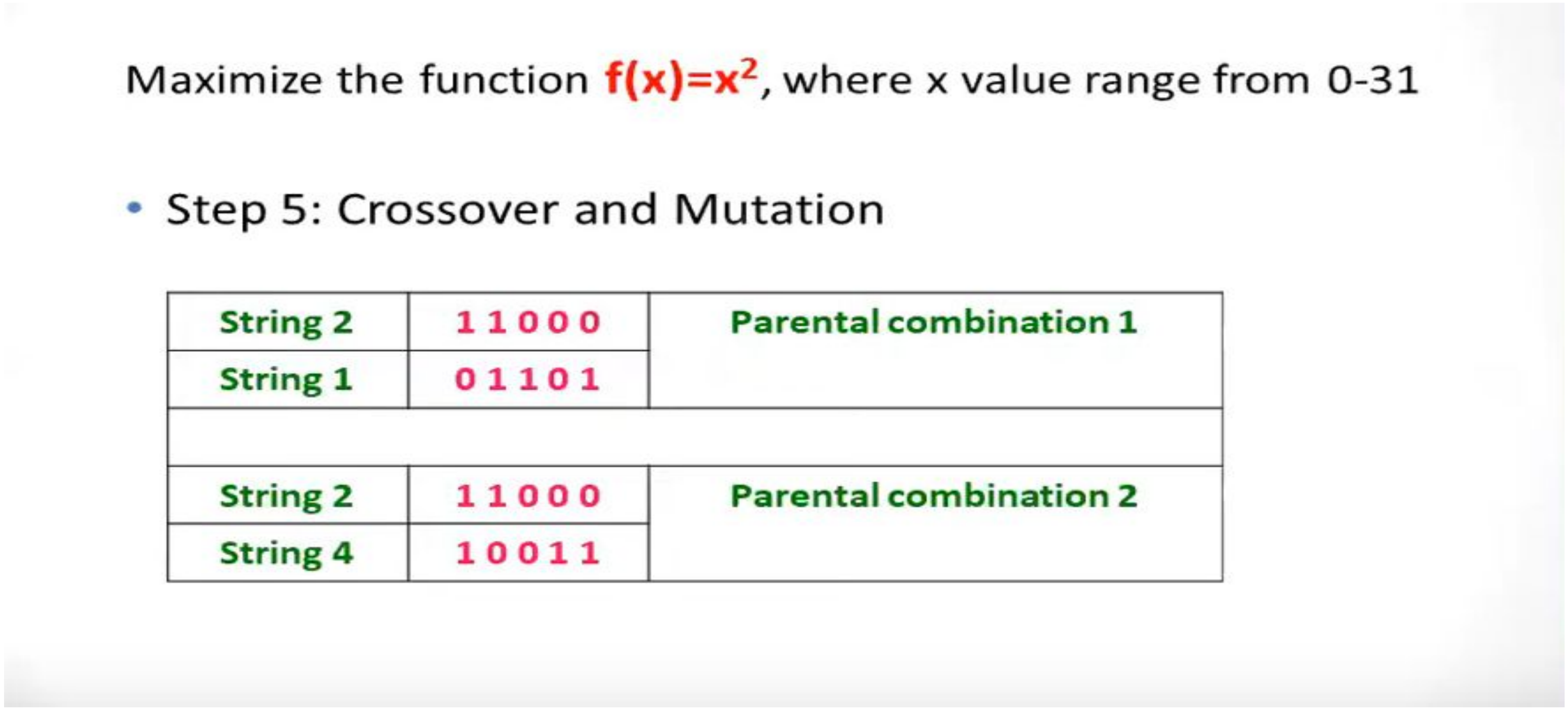


Maximize the function $f(x)=x^2$, where x value range from 0-31

- Step 5: Crossover and Mutation

String No	Initial Population	X Value	F(x) value	Probability count	Expected count	Actual Count
1	01101					1
2	11000					2
3	01000					0
4	10011					1

Example of GA

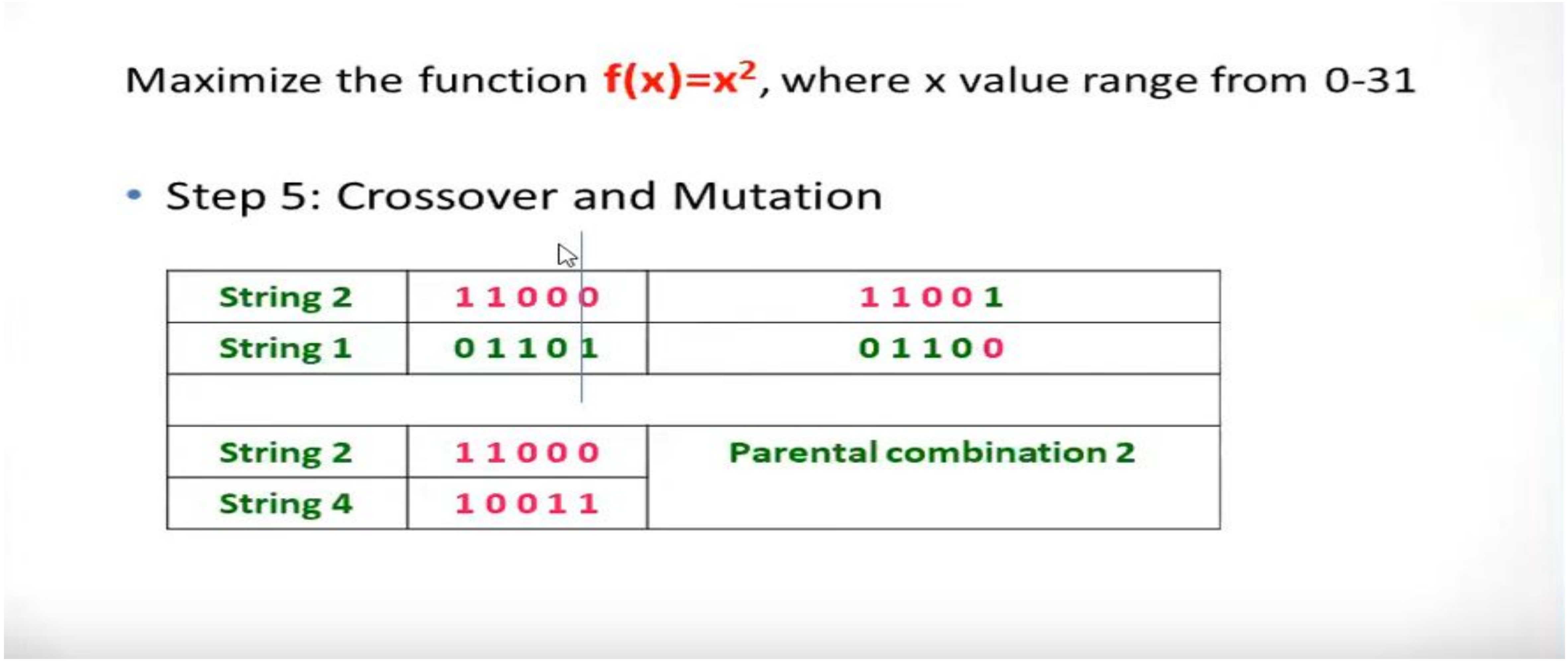


Maximize the function $f(x)=x^2$, where x value range from 0-31

- Step 5: Crossover and Mutation

String 2	1 1 0 0 0	Parental combination 1
String 1	0 1 1 0 1	
String 2	1 1 0 0 0	Parental combination 2
String 4	1 0 0 1 1	

Example of GA

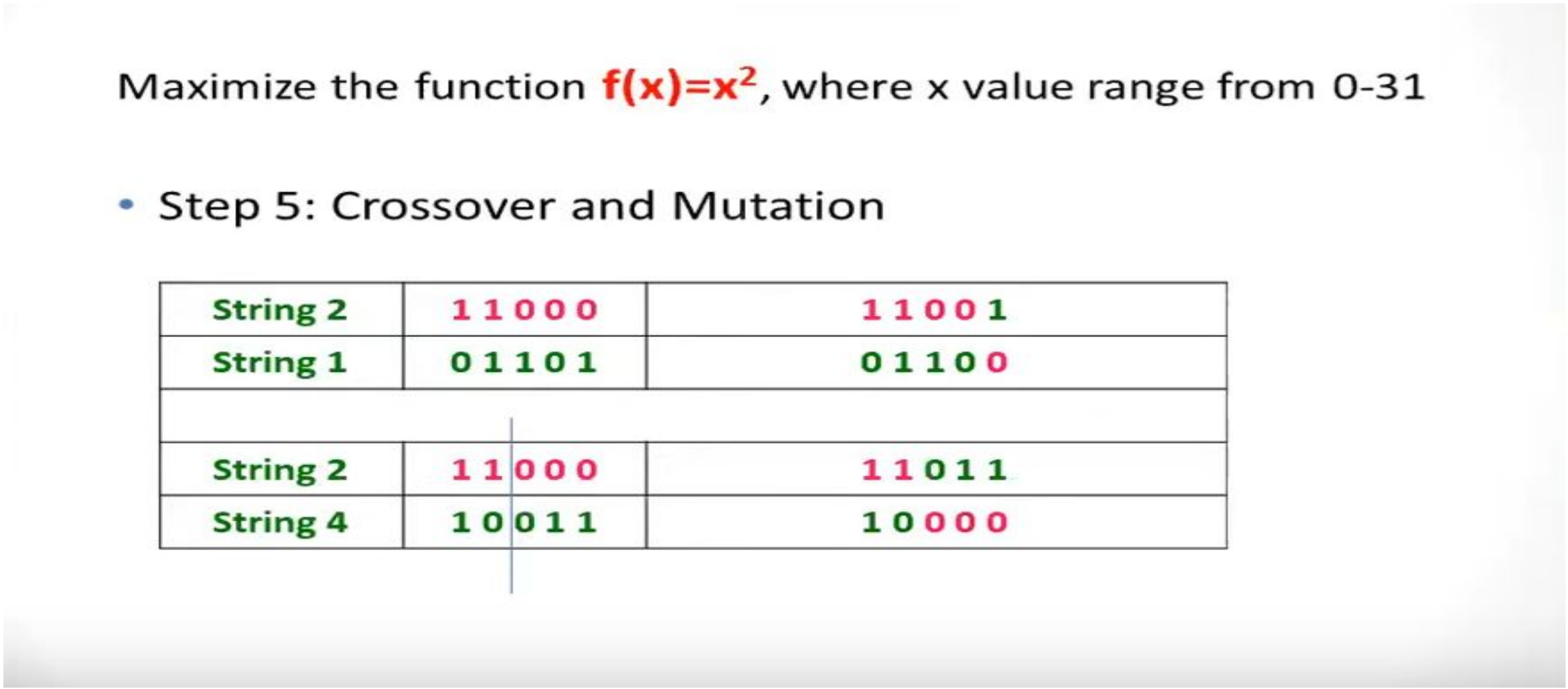


Maximize the function $f(x)=x^2$, where x value range from 0-31

- Step 5: Crossover and Mutation

String 2	1 1 0 0 0	1 1 0 0 1
String 1	0 1 1 0 1	0 1 1 0 0
String 2	1 1 0 0 0	Parental combination 2
String 4	1 0 0 1 1	

Example of GA

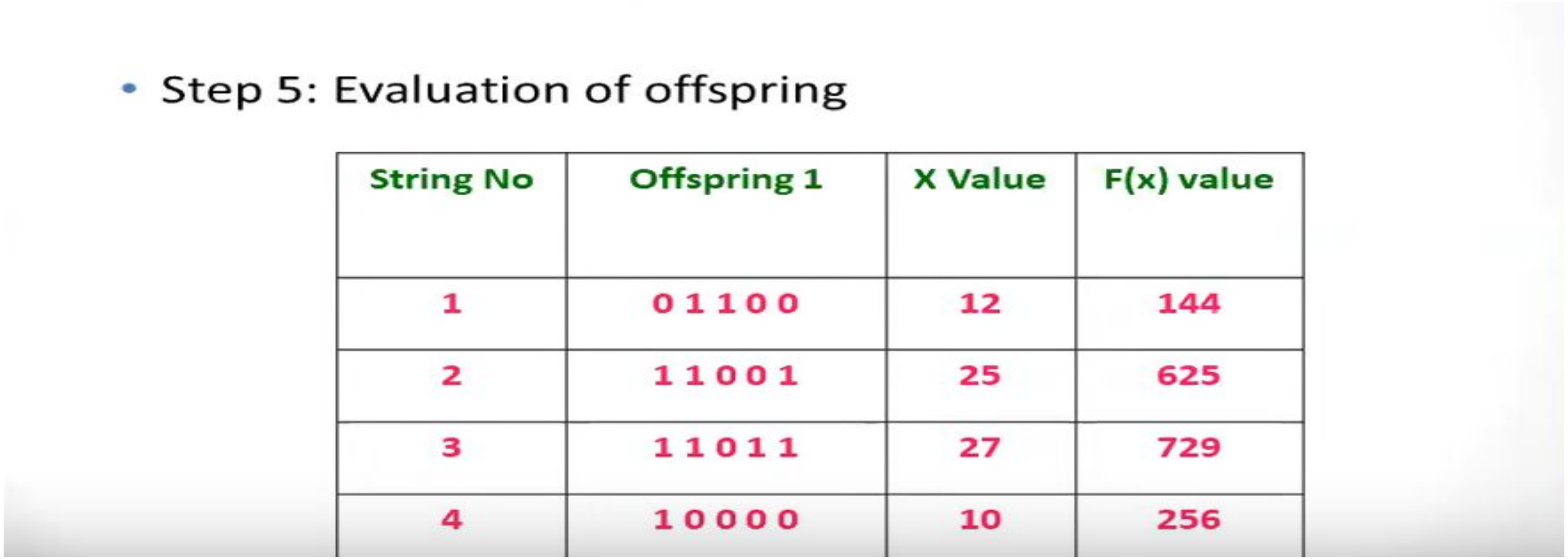


Maximize the function $f(x)=x^2$, where x value range from 0-31

- Step 5: Crossover and Mutation

String 2	1 1 0 0 0	1 1 0 0 1
String 1	0 1 1 0 1	0 1 1 0 0
String 2	1 1 0 0 0	1 1 0 1 1
String 4	1 0 0 1 1	1 0 0 0 0

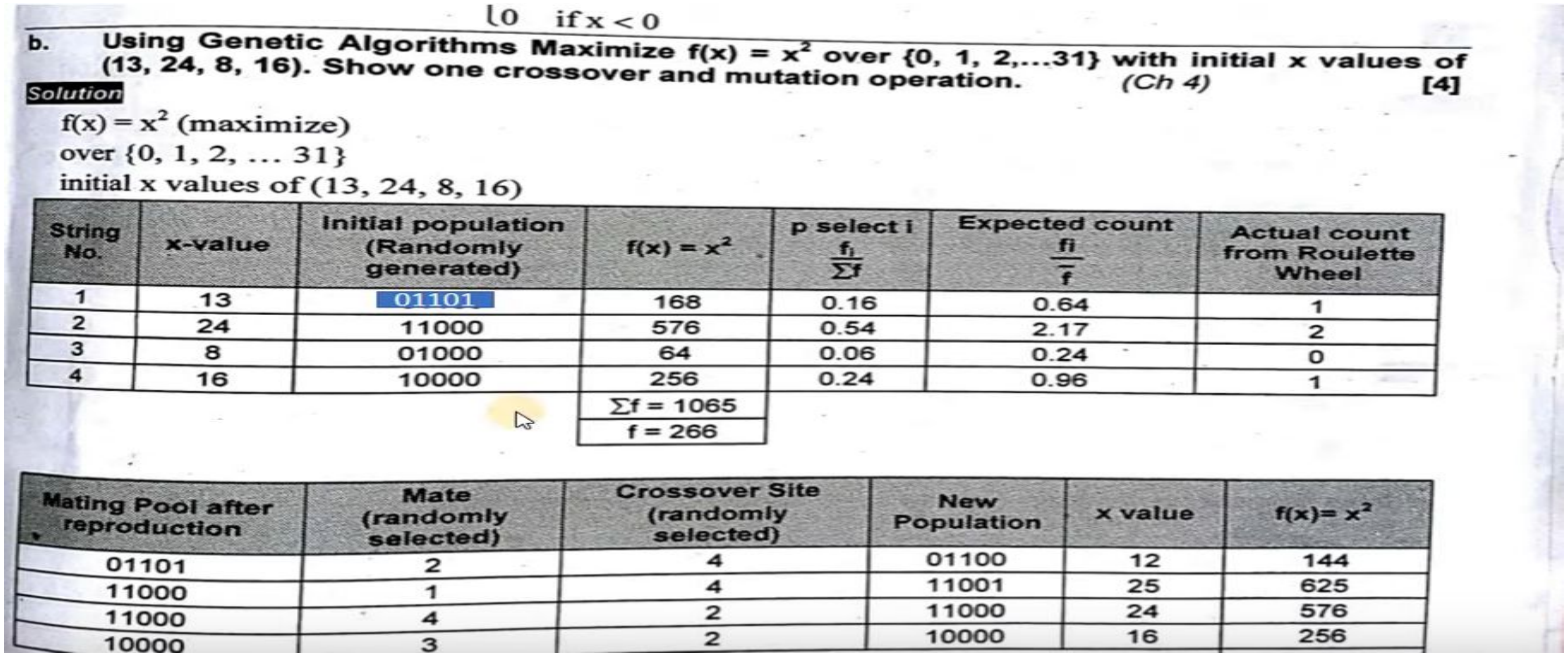
Example of GA



- Step 5: Evaluation of offspring

String No	Offspring 1	X Value	F(x) value
1	0 1 1 0 0	12	144
2	1 1 0 0 1	25	625
3	1 1 0 1 1	27	729
4	1 0 0 0 0	10	256

Example of GA



$0 \quad \text{if } x < 0$

b. Using Genetic Algorithms Maximize $f(x) = x^2$ over {0, 1, 2,...31} with initial x values of (13, 24, 8, 16). Show one crossover and mutation operation. *(Ch 4)* **[4]**

Solution

$f(x) = x^2$ (maximize)

over {0, 1, 2, ... 31}

initial x values of (13, 24, 8, 16)

String No.	x-value	Initial population (Randomly generated)	$f(x) = x^2$	p select i $\frac{f_i}{\Sigma f}$	Expected count $\frac{fi}{\bar{f}}$	Actual count from Roulette Wheel
1	13	01101	168	0.16	0.64	1
2	24	11000	576	0.54	2.17	2
3	8	01000	64	0.06	0.24	0
4	16	10000	256	0.24	0.96	1
			$\Sigma f = 1065$			
			$\bar{f} = 266$			

Mating Pool after reproduction	Mate (randomly selected)	Crossover Site (randomly selected)	New Population	x value	$f(x)= x^2$
01101	2	4	01100	12	144
11000	1	4	11001	25	625
11000	4	2	11000	24	576
10000	3	2	10000	16	256

Previous year Question

1. What is elitism in GA? Give an example.

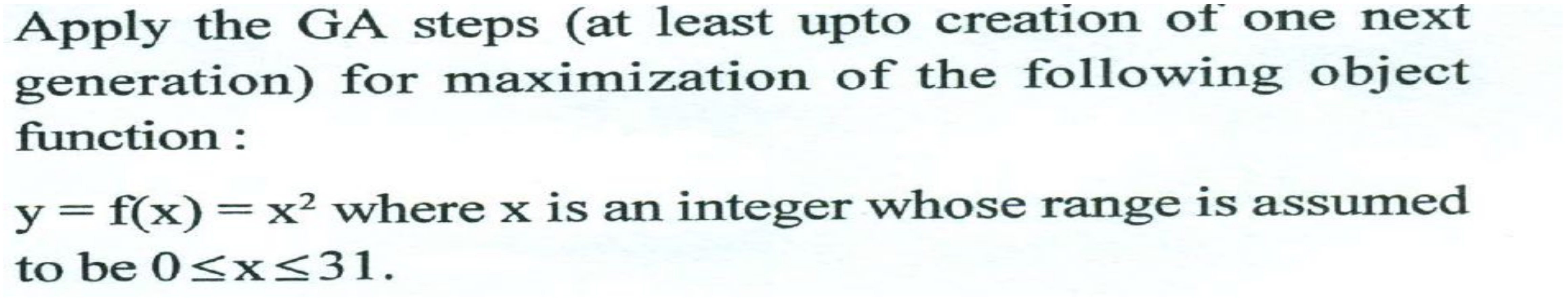
2. Explain the steps of Genetic Algorithm(GA). Carry out at least two iterations of GA for the following optimization problem: Maximize $f(x)= 15x-x^2$ where $0<=x<=15$

   (Assume crossover probability as 100% and mutation probability almost 0%)

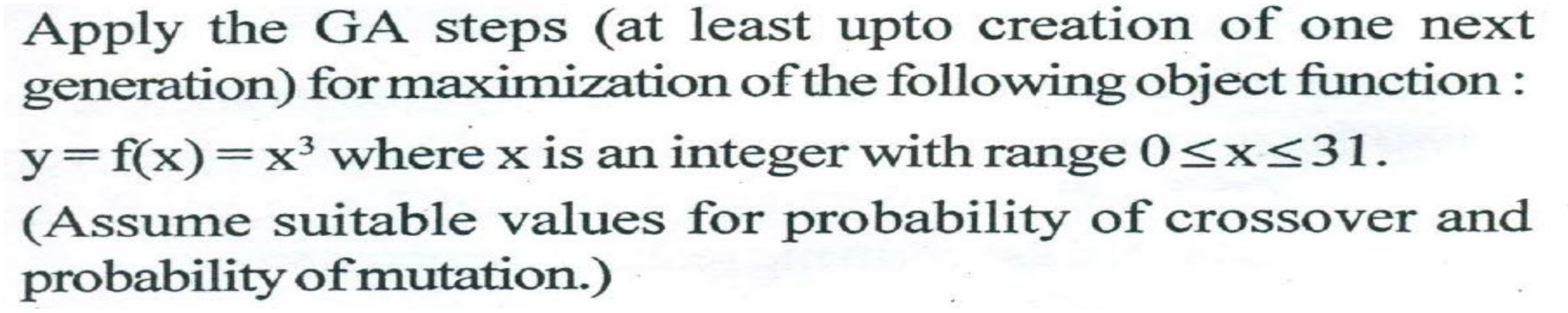
3. With some individual examples, explain the following terms associated with GA:

   Chromosome, Population(Gene pool), Encoding scheme, Fitness evaluation, selection, crossover, mutation.

Previous year Question


Apply the GA steps (at least upto creation of one next generation) for maximization of the following object function :

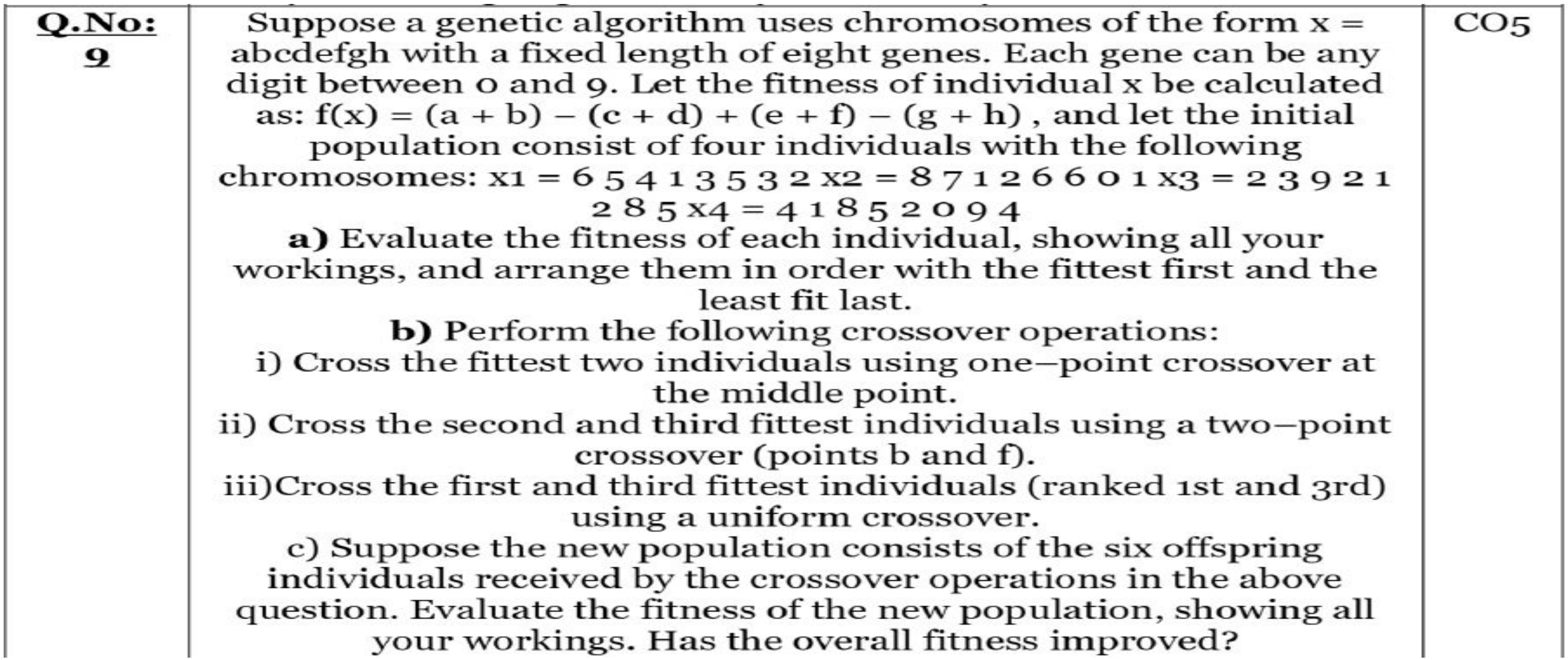
$y = f(x) = x^2$ where x is an integer whose range is assumed to be $0 \leq x \leq 31$.


Apply the GA steps (at least upto creation of one next generation) for maximization of the following object function :

$y = f(x) = x^3$ where x is an integer with range $0 \leq x \leq 31$.

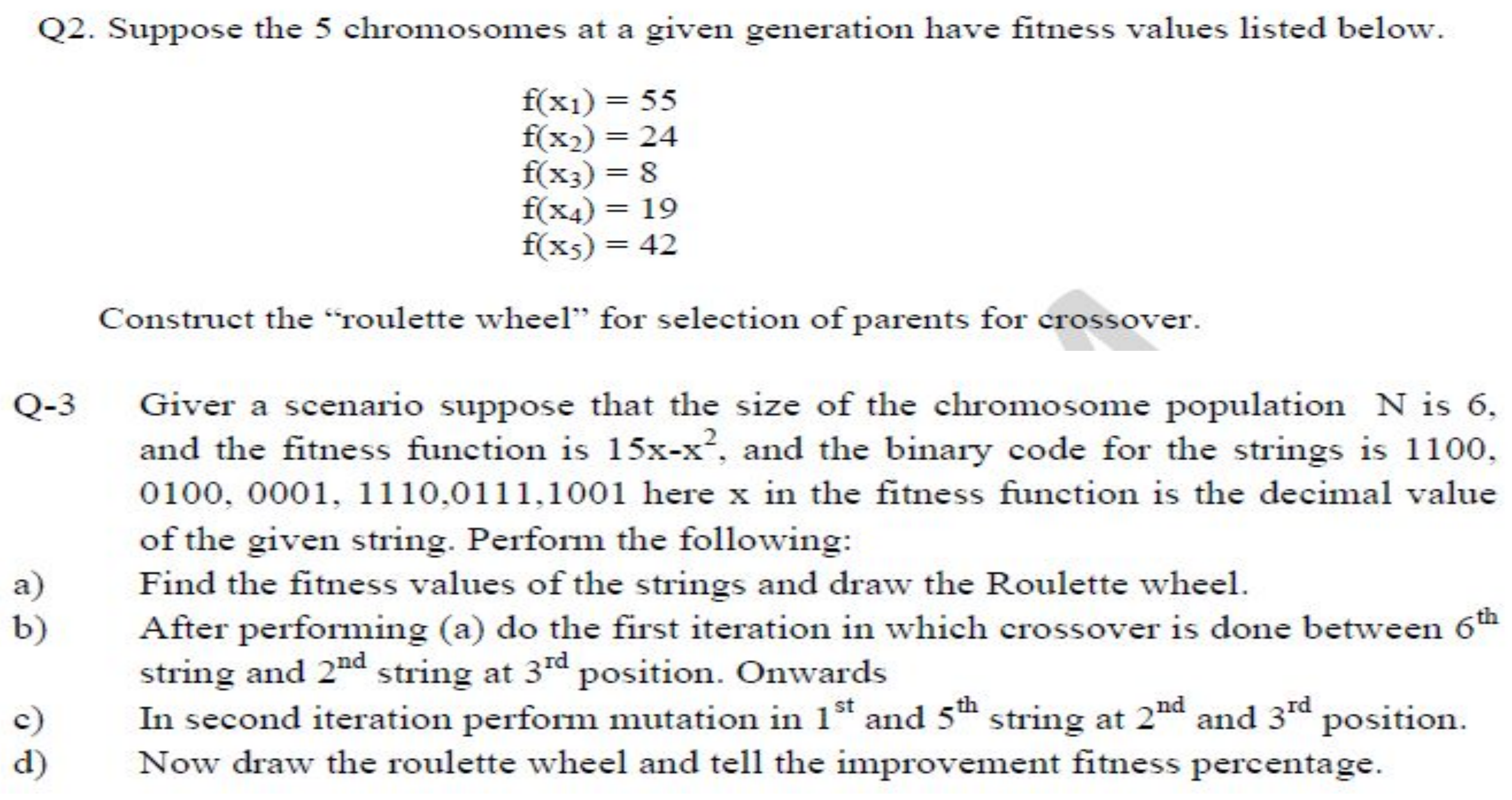
(Assume suitable values for probability of crossover and probability of mutation.)

Practice Question



Q.No: 9	Suppose a genetic algorithm uses chromosomes of the form x = abcdefgh with a fixed length of eight genes. Each gene can be any digit between 0 and 9. Let the fitness of individual x be calculated as: f(x) = (a + b) – (c + d) + (e + f) – (g + h) , and let the initial population consist of four individuals with the following chromosomes: x1 = 6 5 4 1 3 5 3 2 x2 = 8 7 1 2 6 6 0 1 x3 = 2 3 9 2 1 2 8 5 x4 = 4 1 8 5 2 0 9 4 **a)** Evaluate the fitness of each individual, showing all your workings, and arrange them in order with the fittest first and the least fit last. **b)** Perform the following crossover operations: i) Cross the fittest two individuals using one–point crossover at the middle point. ii) Cross the second and third fittest individuals using a two–point crossover (points b and f). iii)Cross the first and third fittest individuals (ranked 1st and 3rd) using a uniform crossover. c) Suppose the new population consists of the six offspring individuals received by the crossover operations in the above question. Evaluate the fitness of the new population, showing all your workings. Has the overall fitness improved?	CO5

Practice Question



Q2. Suppose the 5 chromosomes at a given generation have fitness values listed below.

$f(x_1) = 55$
$f(x_2) = 24$
$f(x_3) = 8$
$f(x_4) = 19$
$f(x_5) = 42$

Construct the "roulette wheel" for selection of parents for crossover.

Q-3 Giver a scenario suppose that the size of the chromosome population N is 6, and the fitness function is $15x-x^2$, and the binary code for the strings is 1100, 0100, 0001, 1110,0111,1001 here x in the fitness function is the decimal value of the given string. Perform the following:

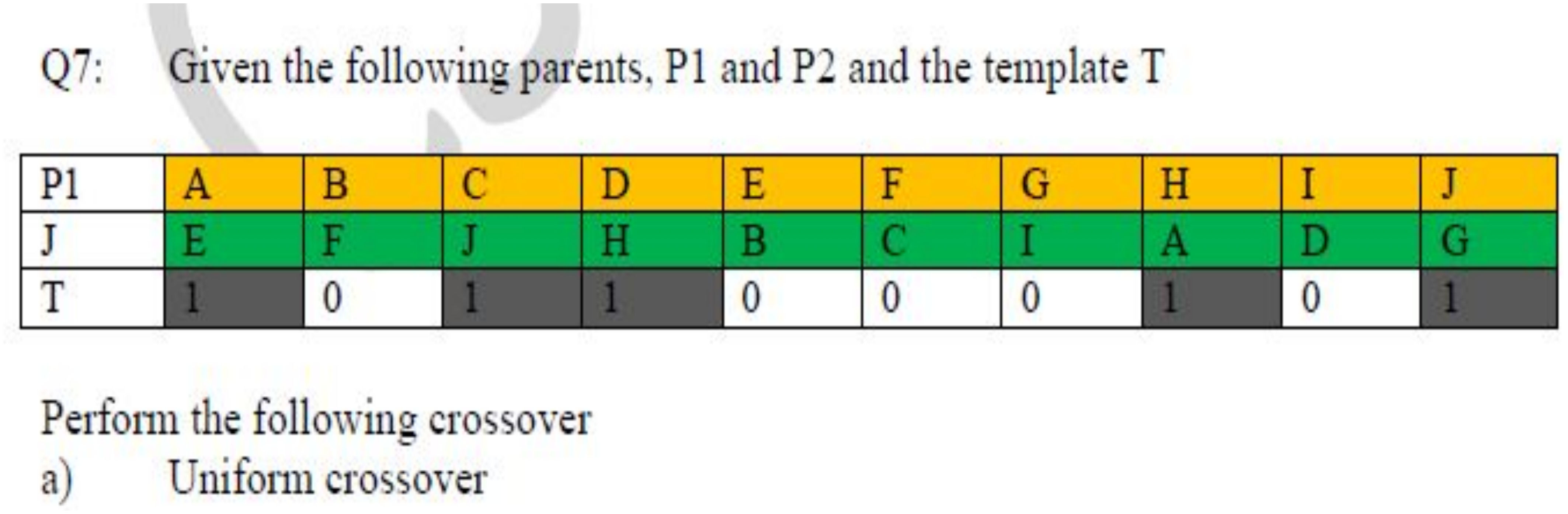
a) Find the fitness values of the strings and draw the Roulette wheel.

b) After performing (a) do the first iteration in which crossover is done between 6th string and 2nd string at 3rd position. Onwards

c) In second iteration perform mutation in 1st and 5th string at 2nd and 3rd position.

d) Now draw the roulette wheel and tell the improvement fitness percentage.

Practice Question



Q7: Given the following parents, P1 and P2 and the template T

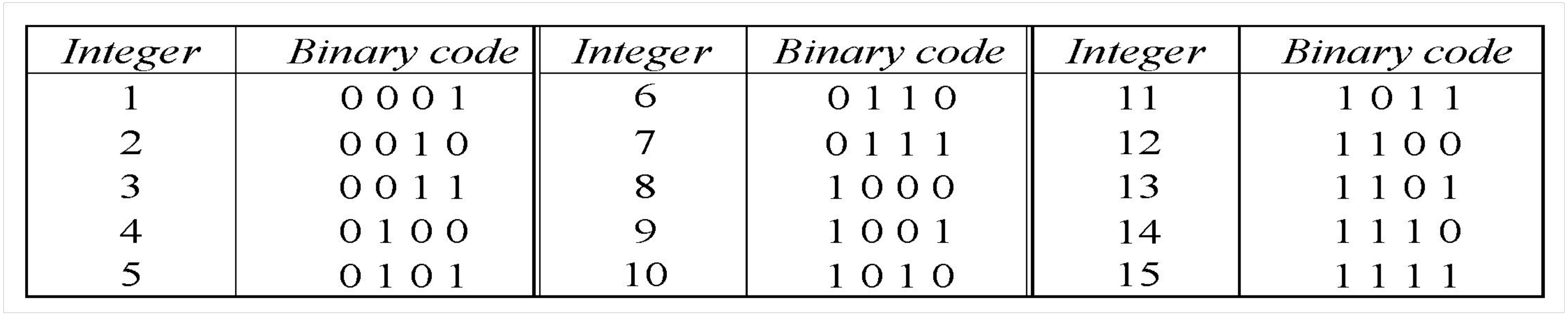
P1	A	B	C	D	E	F	G	H	I	J
J	E	F	J	H	B	C	I	A	D	G
T	1	0	1	1	0	0	0	1	0	1

Perform the following crossover

a) Uniform crossover

Genetic algorithms: case study

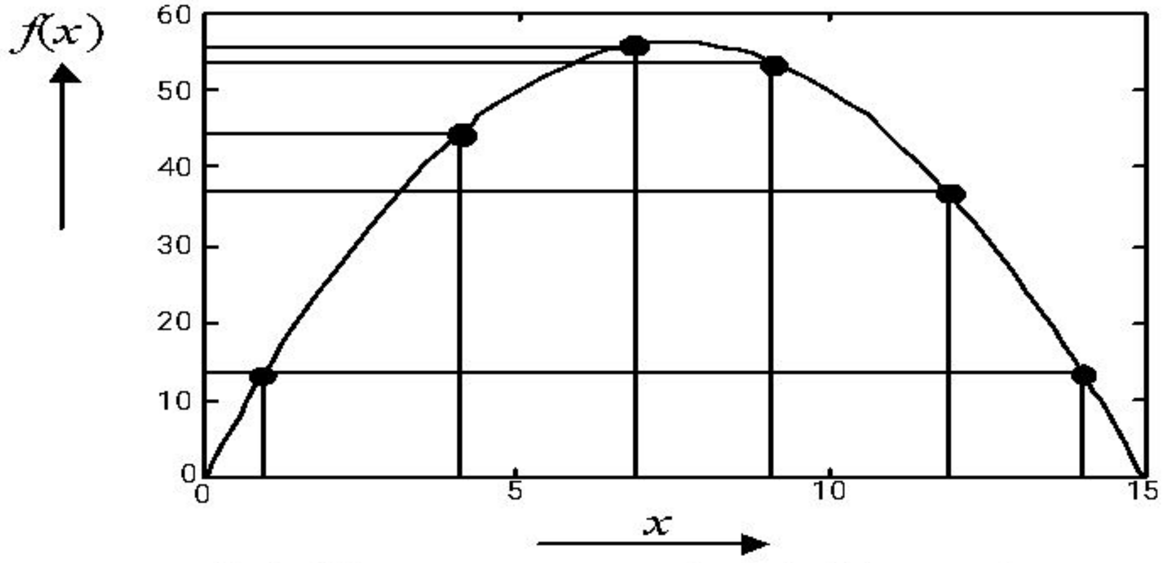
A simple example will help us to understand how a GA works. Let us find the maximum value of the function ($15x - x^2$) where parameter x varies between 0 and 15. For simplicity, we may assume that x takes only integer values. Thus, chromosomes can be built with only four genes:



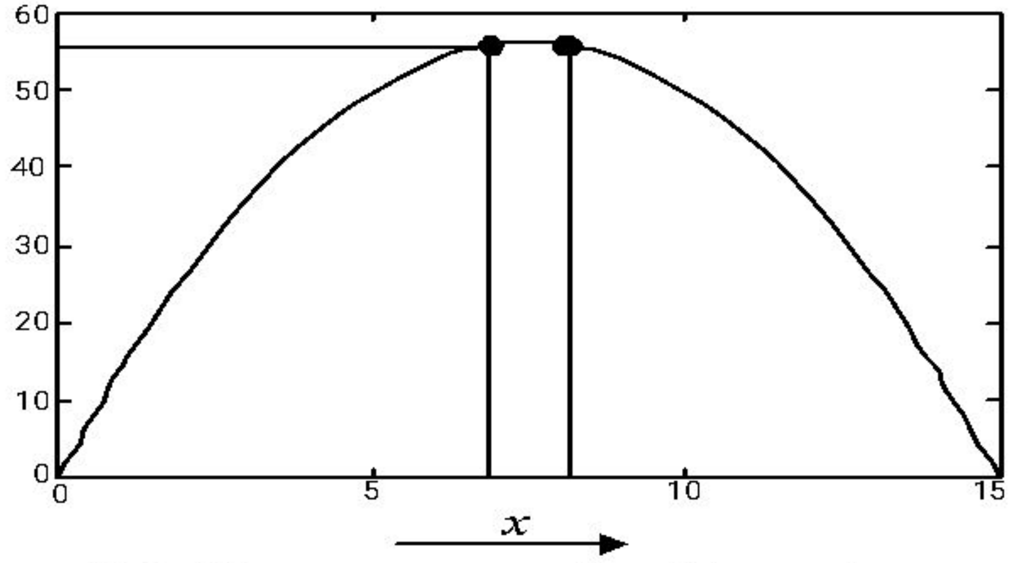
Integer	*Binary code*	*Integer*	*Binary code*	*Integer*	*Binary code*
1	0 0 0 1	6	0 1 1 0	11	1 0 1 1
2	0 0 1 0	7	0 1 1 1	12	1 1 0 0
3	0 0 1 1	8	1 0 0 0	13	1 1 0 1
4	0 1 0 0	9	1 0 0 1	14	1 1 1 0
5	0 1 0 1	10	1 0 1 0	15	1 1 1 1

The fitness function and chromosome locations

Chromosome label	*Chromosome string*	*Decoded integer*	*Chromosome fitness*	*Fitness ratio, %*
X1	1 1 0 0	12	36	16.5
X2	0 1 0 0	4	44	20.2
X3	0 0 0 1	1	14	6.4
X4	1 1 1 0	14	14	6.4
X5	0 1 1 1	7	56	25.7
X6	1 0 0 1	9	54	24.8



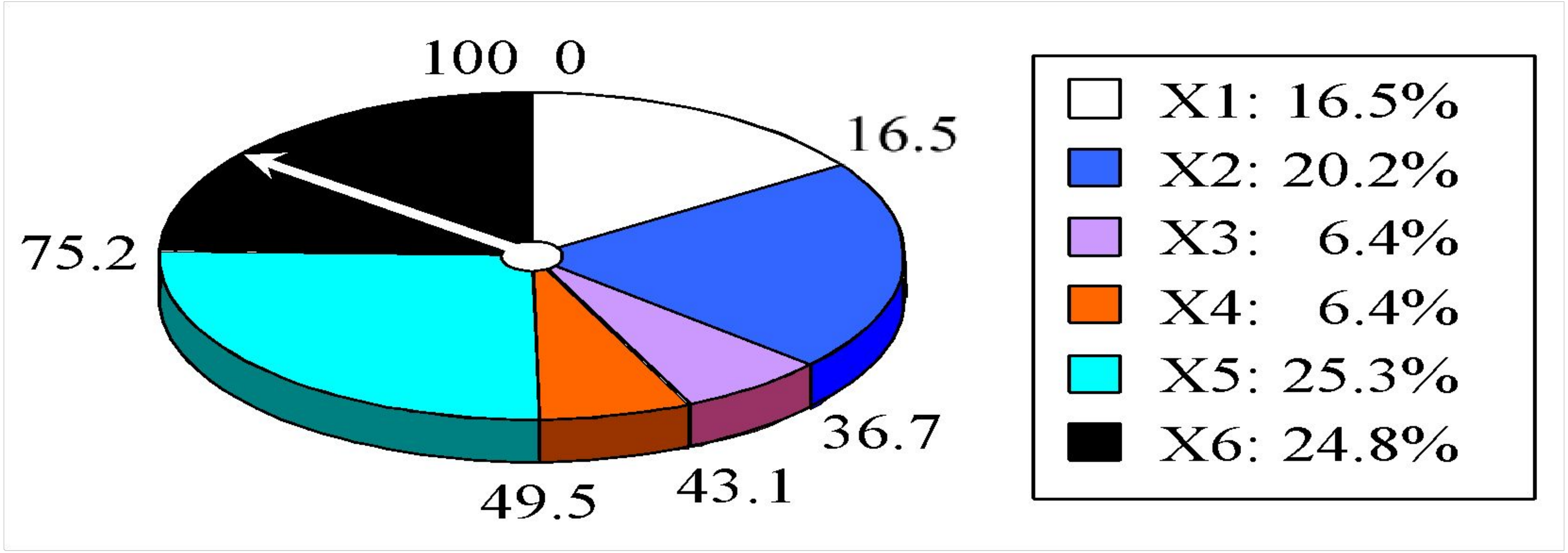
(*a*) Chromosome initial locations.



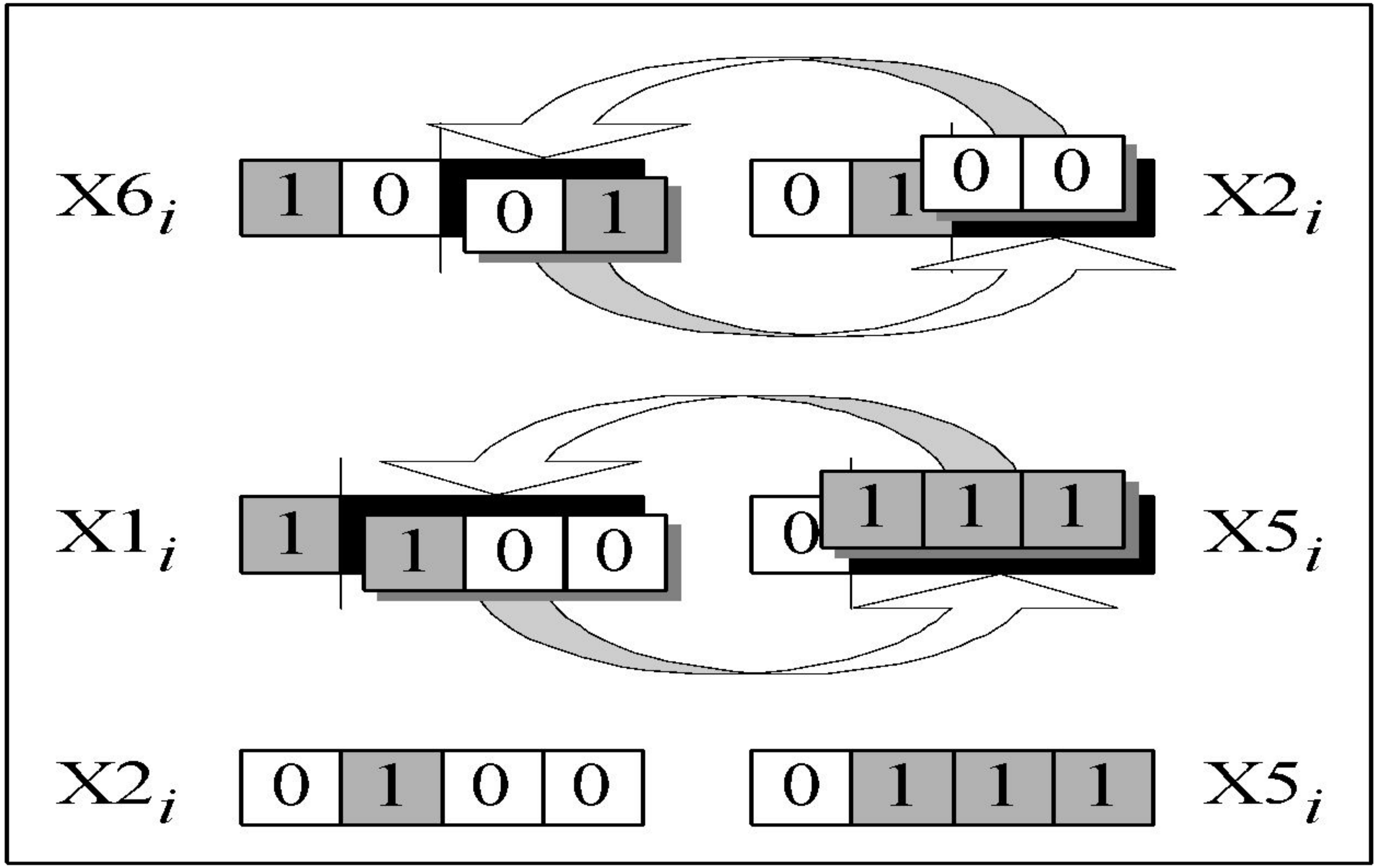
(*b*) Chromosome final locations.

Roulette wheel selection

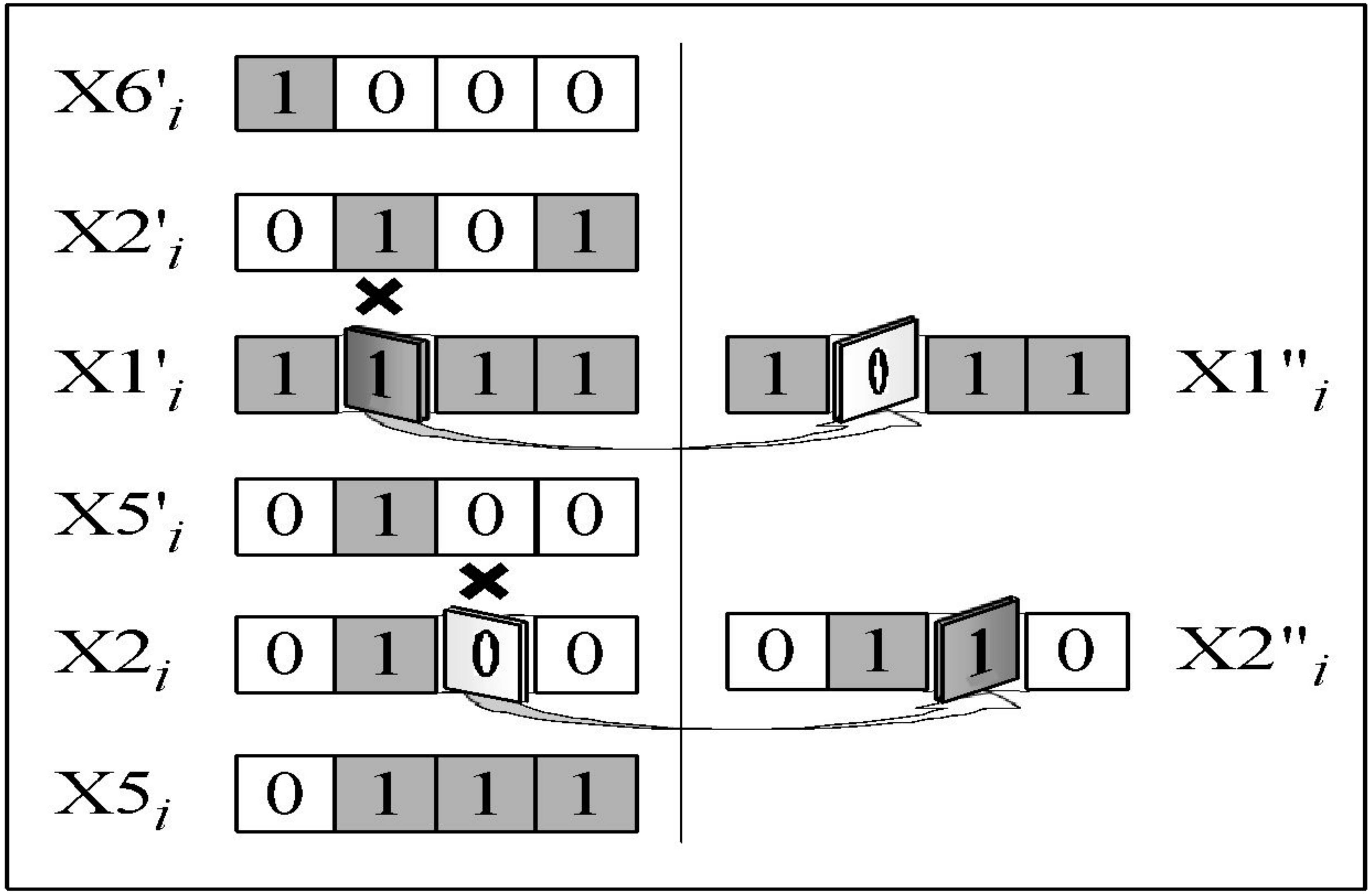
The most commonly used chromosome selection techniques is the **roulette wheel selection**.



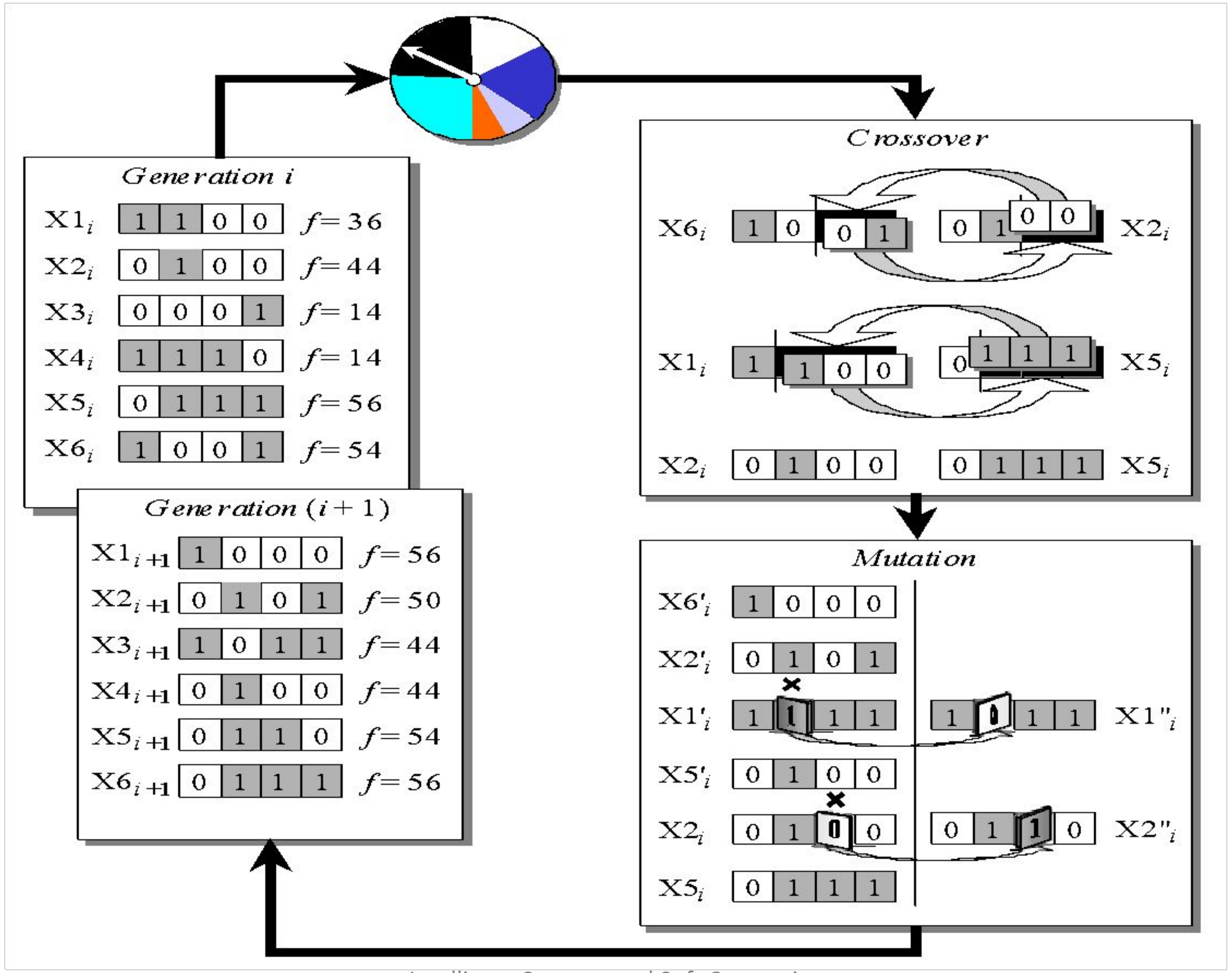
Crossover



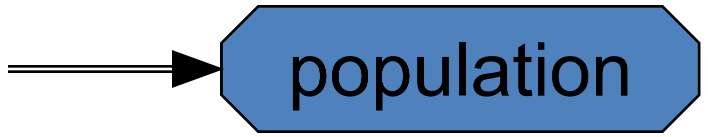
Mutation


X6'$_i$
1 0 0 0
X2'$_i$
0 1 0 1
X1'$_i$
1 1 1 1
X5'$_i$
0 1 0 0
X2$_i$
0 1 0 0
X5$_i$
0 1 1 1
1 0 1 1
X1"$_i$
0 1 1 0
X2"$_i$

The genetic algorithm cycle



Population



Chromosomes could be:

– Bit strings	(0101 ... 1100)
– Real numbers	(43.2 -33.1 ... 0.0 89.2)
– Permutations of element	(E11 E3 E7 ... E1 E15)
– Lists of rules	(R1 R2 R3 ... R22 R23)
– Program elements	(genetic programming)
– ... any data structure ...	

Issues for GA Practitioners

- Choosing basic implementation issues:
  - representation
  - population size, mutation rate, ...
  - selection, deletion policies
  - crossover, mutation operators
- Termination Criteria
- Performance, scalability
- Solution is only as good as the evaluation function (often hardest part)

Benefits of Genetic Algorithms

- Concept is easy to understand
- Modular, separate from application
- Supports multi-objective optimization
- Good for “noisy” environments
- Always an answer; answer gets better with time
- Inherently parallel; easily distributed

Benefits of Genetic Algorithms (cont.)

- Many ways to speed up and improve a GA-based application as knowledge about problem domain is gained
- Easy to exploit previous or alternate solutions
- Flexible building blocks for hybrid applications
- Substantial history and range of use

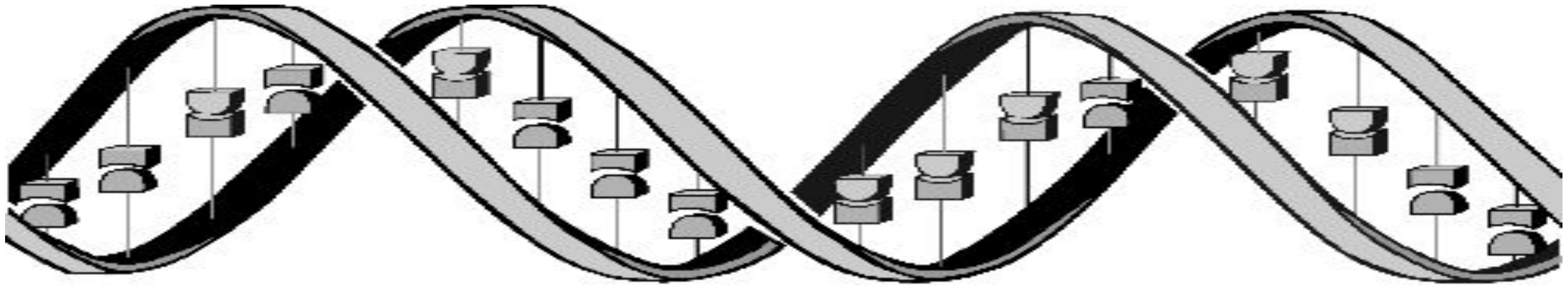
When to Use a GA

- Alternate solutions are too slow or overly complicated
- Need an exploratory tool to examine new approaches
- Problem is similar to one that has already been successfully solved by using a GA
- Want to hybridize with an existing solution
- Benefits of the GA technology meet key problem requirements

Some GA Application Types

Domain	Application Types
Control	gas pipeline, pole balancing, missile evasion, pursuit
Design	semiconductor layout, aircraft design, keyboard configuration, communication networks
Scheduling	manufacturing, facility scheduling, resource allocation
Robotics	trajectory planning
Machine Learning	designing neural networks, improving classification algorithms, classifier systems
Signal Processing	filter design
Game Playing	poker, checkers, prisoner's dilemma
Combinatorial Optimization	set covering, travelling salesman, routing, bin packing, graph colouring and partitioning

Conclusions



Question: ‘If GAs are so smart, why ain’t they rich?’

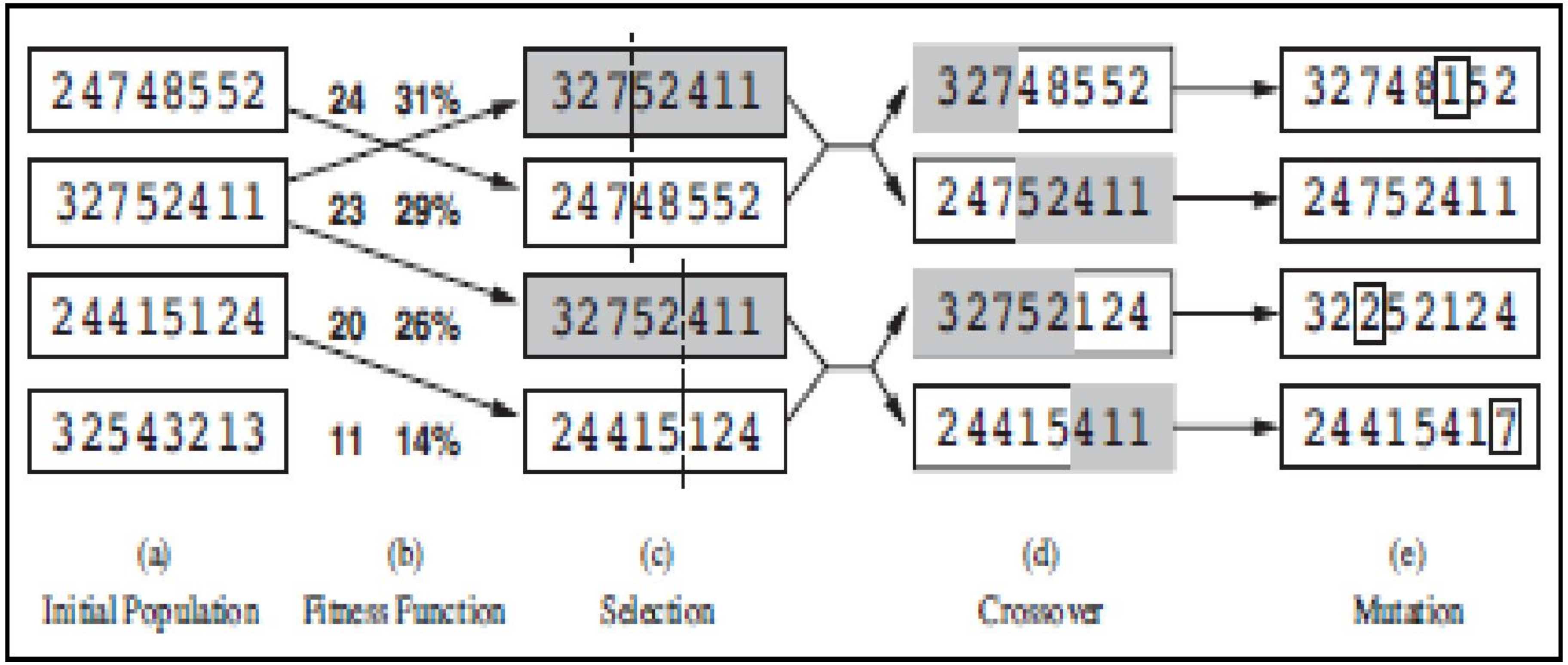
Answer: ‘Genetic algorithms ***are*** *rich - rich in application across a large and growing number of disciplines.’*

- David E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*

Genetic Algorithms

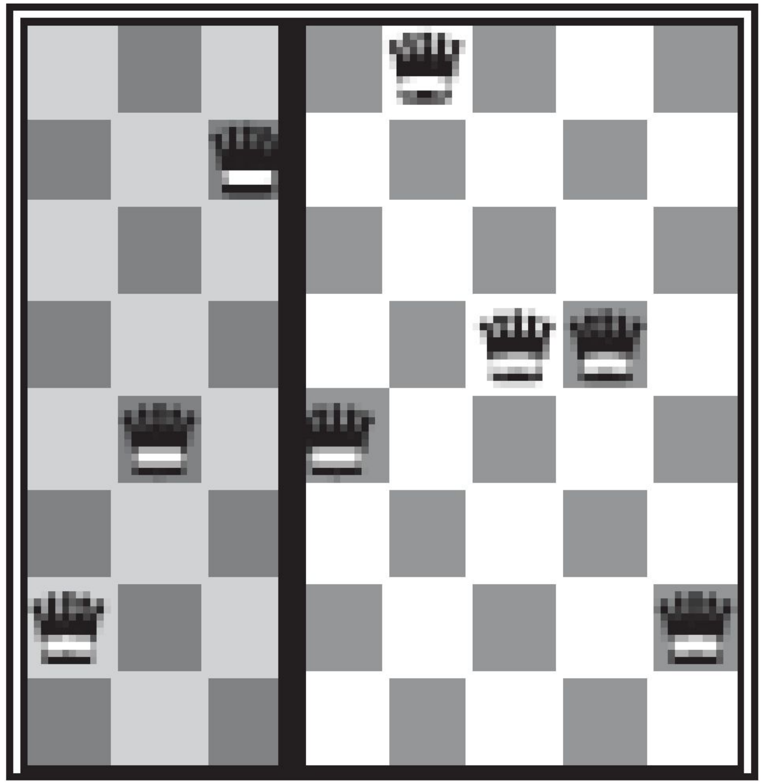
- A **genetic algorithm** (or **GA**) is a variant of *stochastic beam search* in which successor states are generated by combining *two* parent states rather than by modifying a single state.
- The analogy to natural selection is the same as in stochastic beam search, except that now we are dealing with mimicking the natural real life reproduction rather than asexual reproduction.
- Like beam searches, GAs begin with a set of k randomly generated states, called the **population**. Each state, or **individual**, is represented as a string over a finite alphabet—most commonly, a string of 0s and 1s.
- For example, an 8-queens state must specify the positions of 8 queens, each in a column of 8 squares, and so requires $8 \times \log_2 8$ bits.
- Alternatively, the state could be represented as 8 digits, each in the range from 1 to 8.
- *These two encodings behave differently.*

Genetic Algorithms (contd…)

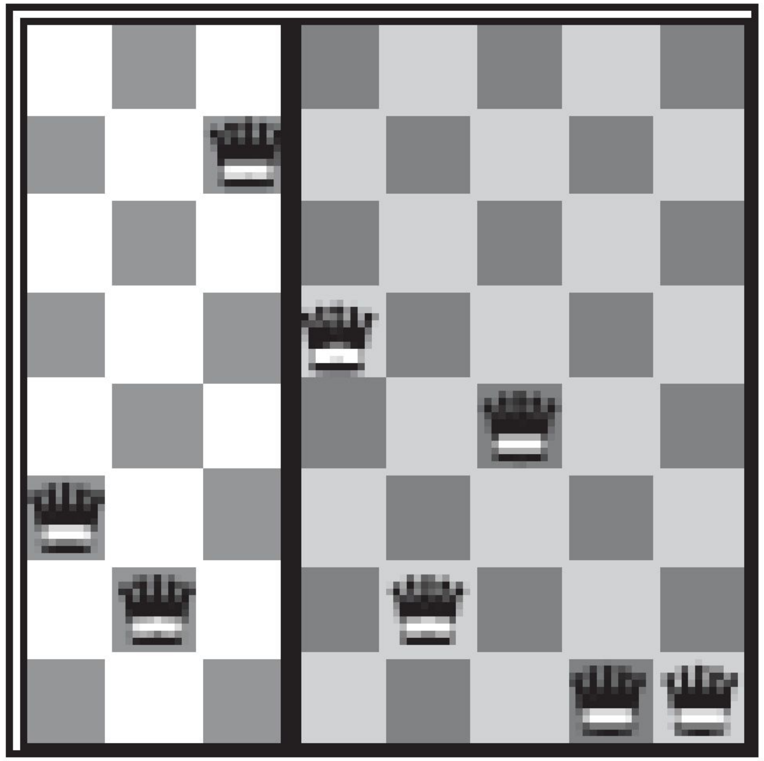


(Figure shows a population of four 8-digit strings representing 8-queens states)

Genetic Algorithms (contd…)



2 4 7 4 8 5 5 2
(**24** *non-attacking pairs of queens)*



3 2 7 5 2 4 1 1
(**23** *non-attacking pairs of queens)*

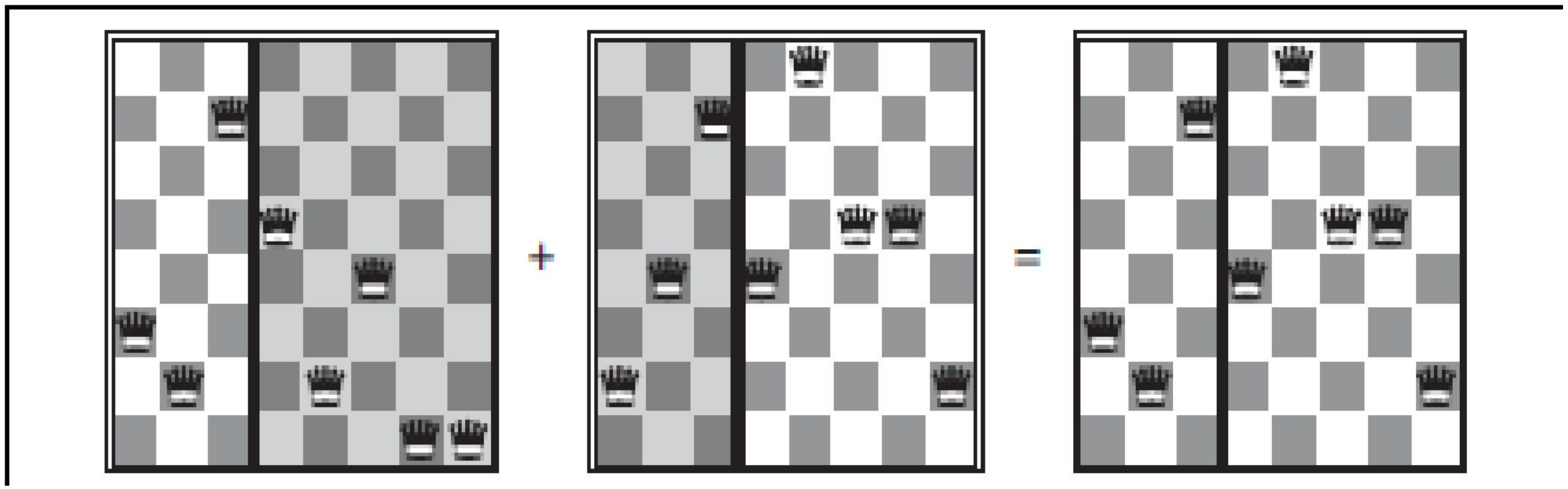
Genetic Algorithms (contd…)

- **(a)** shows a population of four 8-digit strings representing 8-queens states.
- The production of the next generation of states is shown in Figures **(b)–(e).**
- In **(b)**, each state is rated by the objective function, or (in GA terminology) the **fitness function**. A fitness function should return higher values for better states.
- So, for **the 8-queens problem**, we use the ***number of nonattacking pairs of queens***, which has a value of **28** for a solution.
- The values of the given four states are **24, 23, 20,** and **11**.
- In this particular variant of the genetic algorithm, the probability of being chosen for reproducing is directly proportional to the fitness score, and the percentages are shown next to the raw scores.

Genetic Algorithms (contd…)

- In Figure **(c)**, two pairs are selected at random for reproduction, in accordance with the probabilities in (b). It is to be noticed that *one individual is selected twice and one not at all.*
- For each pair to be mated, a **crossover** point is chosen randomly from the positions in the string. In Figure, *the crossover points are after the third digit in the first pair and after the fifth digit in the second pair.*
- In Figure **(d)**, the offspring themselves are created by crossing over the parent strings at the crossover point. For example, the *first child* of the first pair gets the first three digits from the first parent and the remaining digits from the second parent, whereas the *second child* gets the first three digits from the second parent and the rest from the first parent. The 8-queens states involved in this reproduction step are shown in the next figure.

Genetic Algorithms (contd…)



(The 8-queens states corresponding to the first two parents in Figure **(c)** and the first offspring in Figure **(d)**. The shaded columns are lost in the crossover step and the unshaded columns are retained.)

Genetic Algorithms (contd…)

- The example shows that when two parent states are quite different, the crossover operation can produce a state that is a long way from either parent state.
- It is often the case that the population is quite diverse early on in the process, so crossover (like *simulated annealing*) frequently takes large steps in the state space early in the search process and smaller steps later on when most individuals are quite similar.
- Finally, in Figure **(e)**, each location is subject to random **mutation** with a small independent probability.
- One digit was mutated in the first, third, and fourth offspring.
- In the 8-queens problem, this corresponds to choosing a queen at random and moving it to a random square in its column.

Genetic Algorithms (contd…)

- **function** GENETIC-ALGORITHM(population, FITNESS-FN) **returns** an individual
- **inputs**: population, a set of individuals
- FITNESS-FN, a function that measures the fitness of an individual
- **repeat**
- new population ←empty set
- **for** i = 1 **to** SIZE(population) **do**
- x ←RANDOM-SELECTION(population, FITNESS-FN)
- y ←RANDOM-SELECTION(population, FITNESS-FN)
- child ←REPRODUCE(x , y)
- **if** (small random probability) **then** child ←MUTATE(child)
- add child to new population
- population ←new population
- **until** some individual is fit enough, or enough time has elapsed
- **return** the best individual in population, according to FITNESS-FN

- **function** REPRODUCE(x , y) **returns** an individual
- **inputs**: x , y, parent individuals
- n←LENGTH(x); c←random number from 1 to n
- **return** APPEND(SUBSTRING(x, 1, c), SUBSTRING(y, c + 1, n))

(**A genetic algorithm**. The algorithm is the same as the one diagrammed in Figure, with one variation: in this more popular version, each mating of two parents produces only one offspring, not two.)