Computational Intelligence

About This Subject

- Subject Code: **SCJ4553**
- Subject Name: **Computational Intelligence**
- Pre-Requisite: Linear Algebra, Data Structure, Probability and Statistics Computer programming

- Meeting Hours:

- Lecturer(s): AP. Dr. Siti Zaiton Mohd Hashim

sitizaiton@utm.my
019-7726248
Faculty of Computing
UTM Skudai

Learning Objectives

- To have in-depth understanding of Soft Computing/Computational Intelligence/Bio-Inspired Computing (SC/CI and BIC) techniques
- To investigate some common models and their applications in solving real-world problems
- To implement these techniques on selected case-studies
- To be alert of the recent progress made by the scientific and technical community in the broader field of SC/CI/BIC

Contents of Subject

- Overview of Artificial Intelligence
- Neural Network: Supervised and Unsupervised Learning
- **Evolutionary Computing: GA**
- Fuzzy Logic
- Rough Set

Technique 2: Evolutionary Computing (Genetic Algorithms with Hands-on simulation)

Contents

- Objective of this topic:
  - To understand the processes involved ie. GAs Basic flows – operator and parameters (roles, effects etc)
  - To be able to apply GAs in solving optimisation problems

Contents

Genetic Algorithms

- Introduction
- Simulation of Natural Evolution
- Genetic Algorithms : Mice & Cat Story
- Example 1 : Burger and Profit Problem
- Example 2 : Optimization of simple equation
- Example 3 : Optimization of complex equation
- Example 4 : The Traveling Salesman Problem
- Summary

Introduction: Can Evolution be Intelligent ?

- Intelligence can be defined as the capability of a system to adapt its behavior to ever-changing environment.

- Evolutionary computation simulates evolution on a computer. The result of such a simulation is a series of optimization algorithms, usually based on a simple set of rules.

Introduction…continue

- Optimization iteratively improves the quality of solutions until optimal or at least feasible, solution is found.

- The behavior of an individual organism is an inductive inference about some yet unknown aspects of its environment. If, over successive generations, the organism survives, we can say that this organism is capable of learning to predict changes in its environment

Introduction…continue

- The evolutionary approach is based on computational models of natural selection and genetics. We call the evolutionary computation, an umbrella term that combines genetic algorithms, evolutionary strategies and genetic programming.

Simulation of Natural Evolution

- Natural Evolution .. Neo-Darwinian paradigm … process of reproduction, mutation, competition and selection.
- The power to reproduce appears to be an essential property of life.
- The power to mutate is also guaranteed in any living organism that reproduces itself in a continuously changing environment.
- Processes of competition and selection normally take place in the natural world, where expanding populations of different species are limited by a finite space.

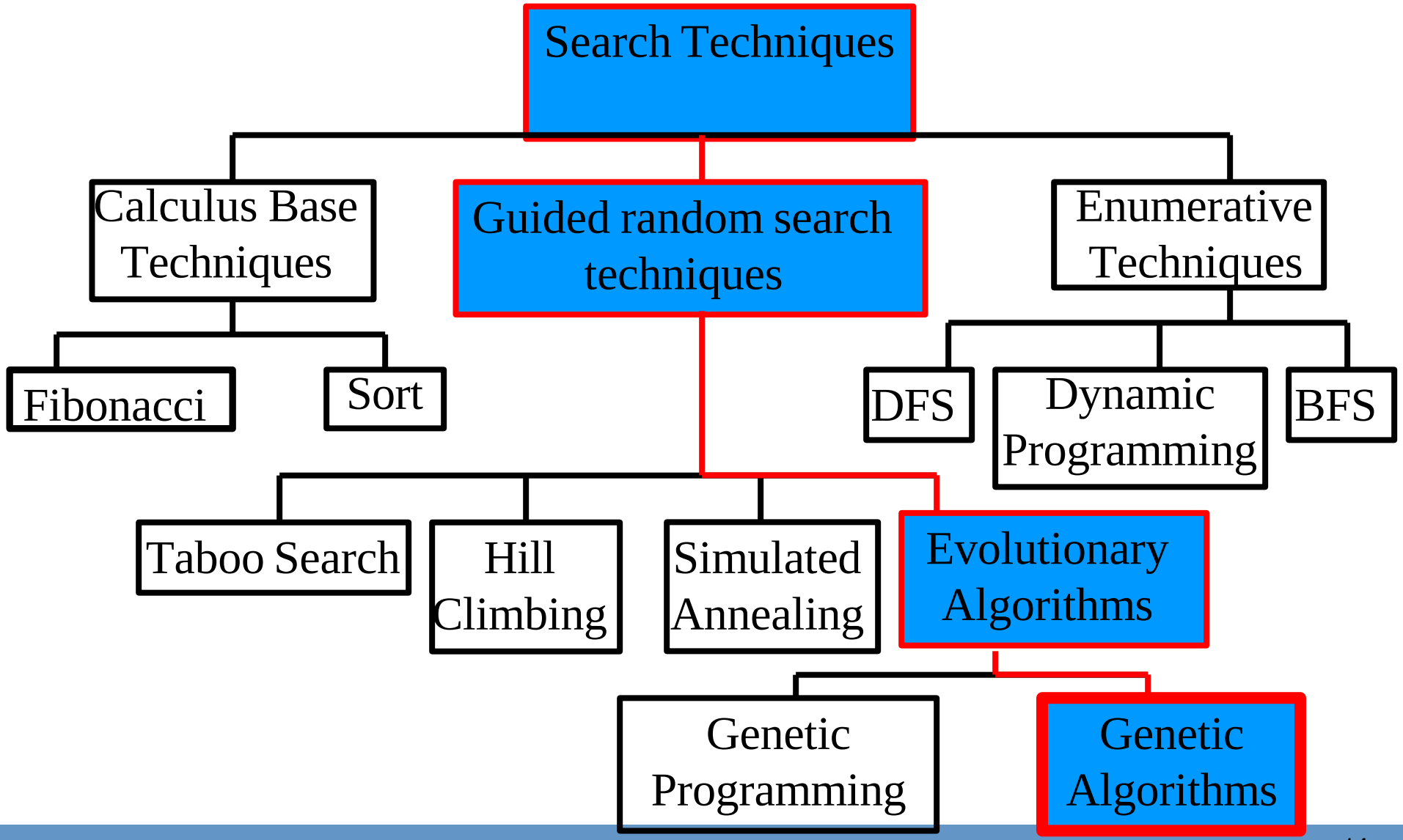
Simulation of Natural Evolution.....

- Evolution can be seen as a process leading to the maintenance of a population's ability to survive and reproduce in a specific environment. This ability is called evolutionary fitness.
- Evolutionary fitness can also be viewed as a measure of organism's ability to anticipate changes in its environment.
- The better an organism's fitness to the environment, the better its chances to survive

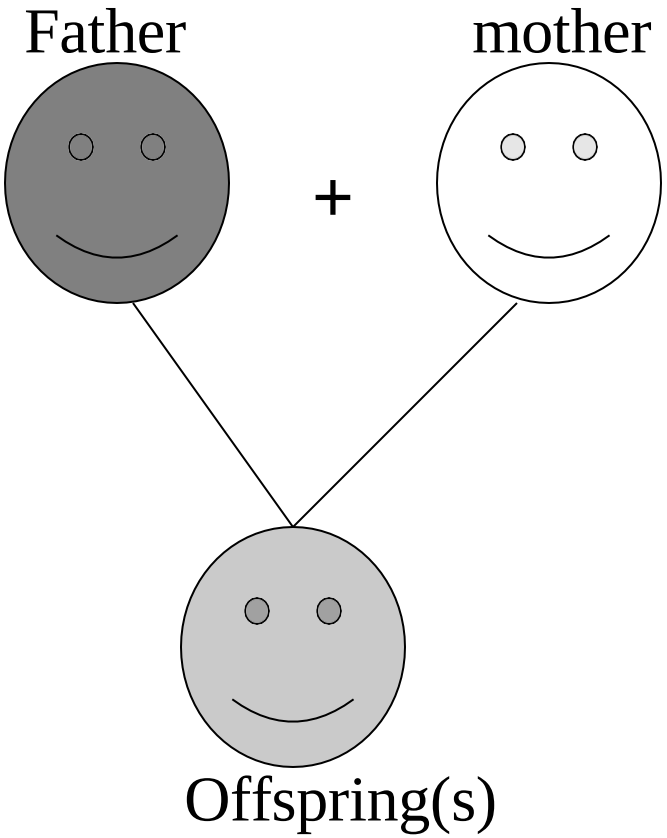
What is GA(s) again ?

- Inspired by biological evolution process
- A class of probabilistic optimization algorithms
- In General, GA is a guided search algorithm based on evolutionary process

Classes of search techniques



Evolutionary process !

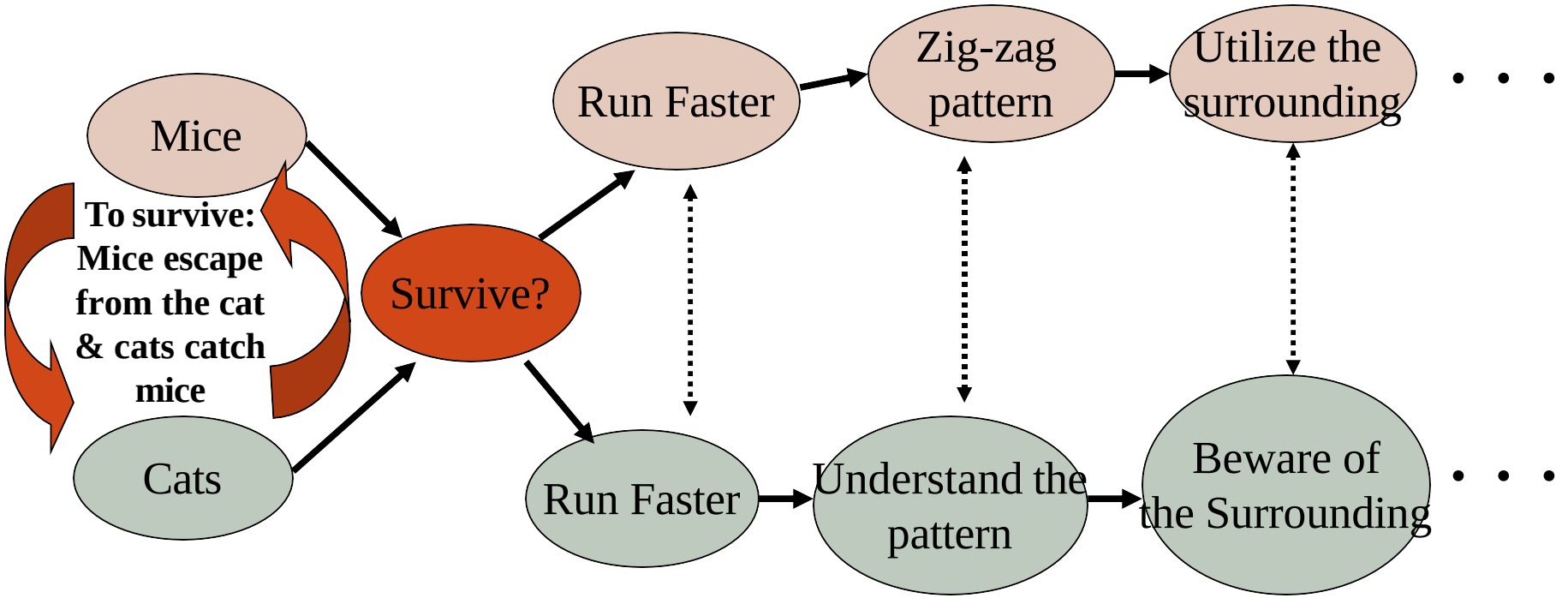


Example of evolutionary process

- Mice & Cats

Basically it is about the survival of the fittest …

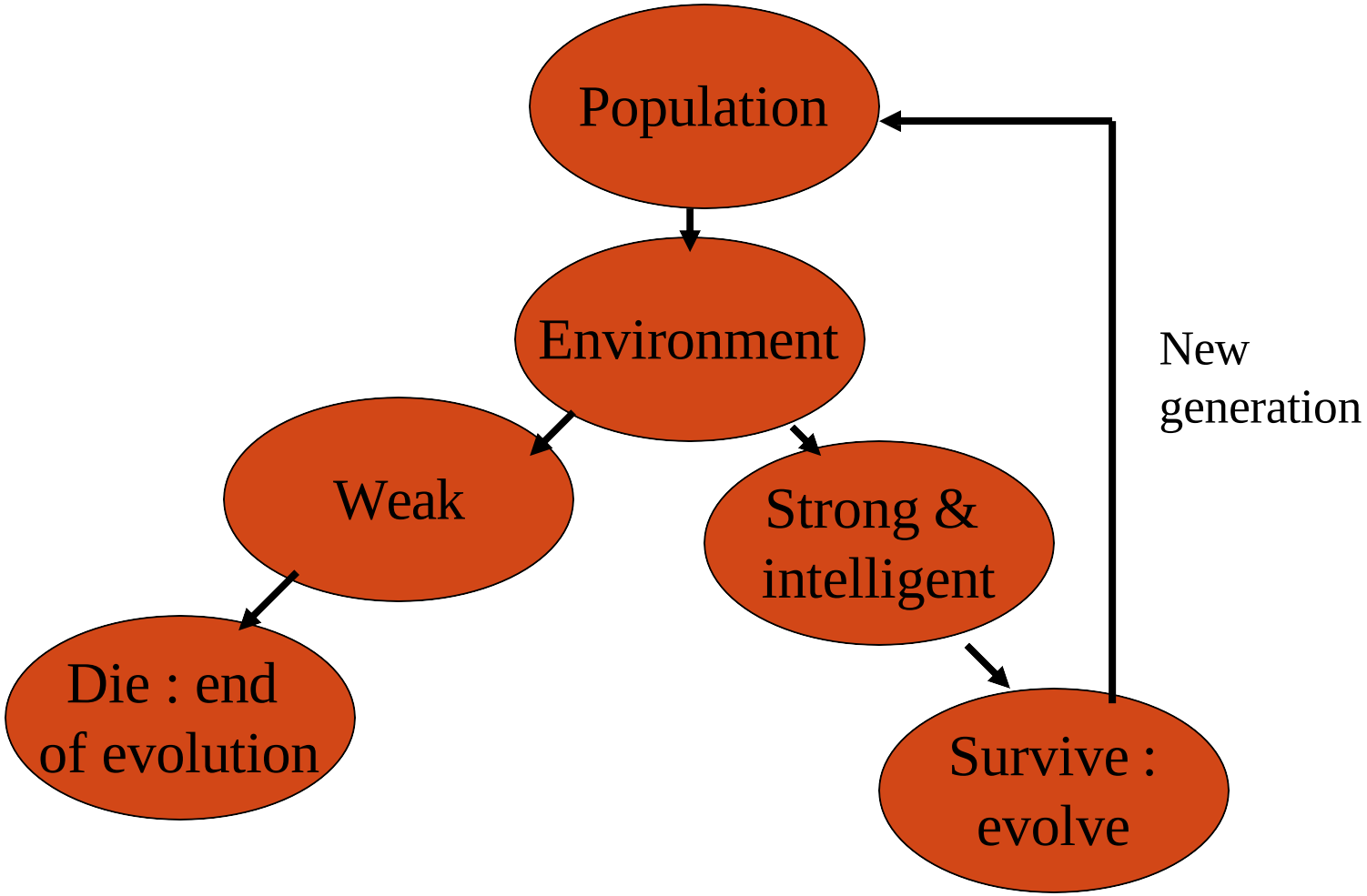
mice & cats : an evolutionary process



The Mice & Cats algorithm

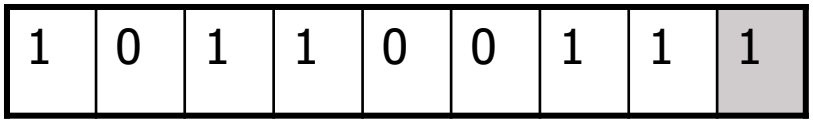
```
Mice & Cats want to survive
  While (mice != dead || cats != dead)
        if (speed(mice) > speed(cats) )
             mice = survive  else  cats = survive
        if (pattern(mice) > pattern(cats)
            mice = survive else cats = survive
        if  (surrounding(mice) > surrounding(cats)
           mice = survive else cats = survive
     *
     *
     *
endofwhile
```

General Evolutionary Process



About ..Genetic Algorithms

- In the early 1970's John Holland introduced the concept of genetic algorithms
- His aim was to make computer do what nature does. Holland was concerned with algorithms that manipulate strings of binary digits.
- Each artificial "chromosomes" consist of a number of "genes", and each gene is represented by 0 or 1



About ..Genetic Algorithms

- Nature (God's Creation) has an ability to adapt and learn without being told what to do. In other words, nature finds good chromosomes blindly. Basically GAs do the same. Two mechanisms link a GA to the problem it is solving: **encoding** and **evaluation**.

- The GA uses a **measure of fitness** of individual chromosomes to carry out reproduction. As reproduction takes place, the crossover operator exchanges parts of two single chromosomes, and the mutation operator changes the gene value in some randomly chosen location of the chromosome.

GA … a definition

A genetic algorithm maintains a population of candidate solutions for the problem at hand, and makes it evolve by iteratively applying a set of stochastic operators

Think: Reflection- How would you determine a university with selected smart students only using GA?chromosome? How to produce the smart ones?

GA Stochastic operators

- ***Selection*** replicates the most successful solutions found in a population at a rate proportional to their relative quality
- ***Recombination*** decomposes two distinct solutions and then randomly mixes their parts to form novel solutions
- ***Mutation*** randomly perturbs a candidate solution

Metaphor

Genetic Algorithm	Nature/Real Life
Optimization problem	Environment
Feasible solutions	Individuals living in that environment
Solutions quality (fitness function)	Individual's degree of adaptation to its surrounding environment

Metaphor (cont..)

Genetic Algorithm	Nature
A set of feasible solutions	A population of organisms (species)
Stochastic operators	Selection, recombination and mutation in nature's evolutionary process
Iteratively applying a set of stochastic operators on a set of feasible solutions	Evolution of populations to suit their environment

Basic Genetic Algorithm

- Step 1: Represent the problem variable domain as a chromosomes of a fixed length. Choose size of population *N*, probability of mutation *pm* and crossover *pc*
- Step 2: Define fitness function to measure performance of individual chromosomes.
- Step 3: Randomly generate initial population of chromosomes of sized *N.*
- Step 4: Calculate the fitness of each individual chromosomes.
- Step 5: Select a pair of "fit" chromosomes for mating.

Basic Genetic Algorithm

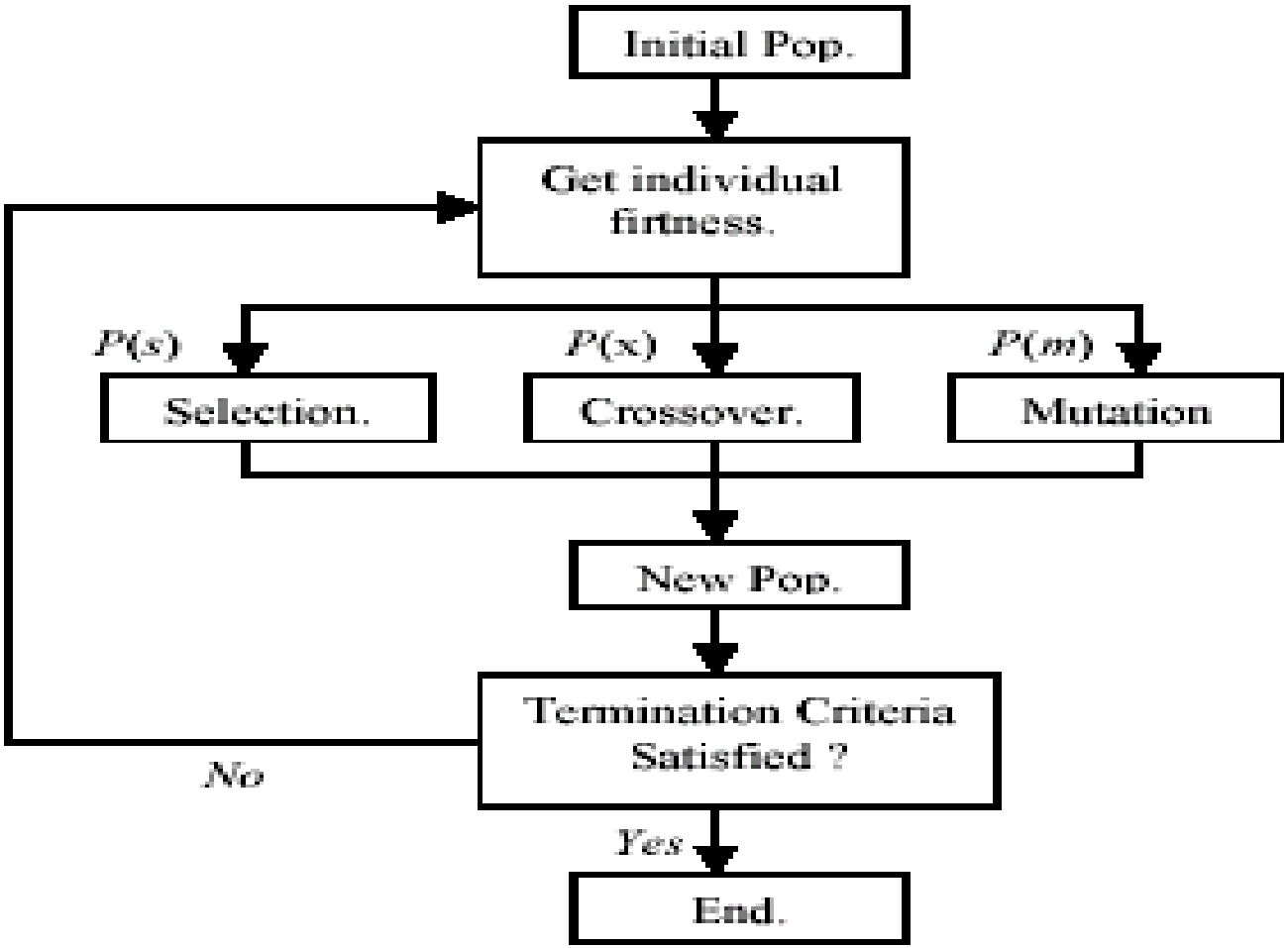
- Step 6: Create a pair of offspring chromosomes by applying crossover and mutation.
- Step 7: Place the created offspring chromosomes in the new population.
- Step 8: repeat *Step 5* until the size of the new population equal to size of initial population, *N*
- Step 9: Replace the initial (parent) chromosomes with the new (offspring) population.
- Step 10: Go to *Step 4* and repeat the process until the termination criterion satisfy.

Another way of looking at this…

- produce an initial population of individuals
- evaluate the fitness of all individuals
- **While** termination condition not met **do**
  - ➔select fitter individuals for reproduction
  - ➔recombine between individuals
  - ➔mutate individuals
  - ➔evaluate the fitness of the modified individuals
  - ➔generate a new population

  End while

Another way of looking at this…

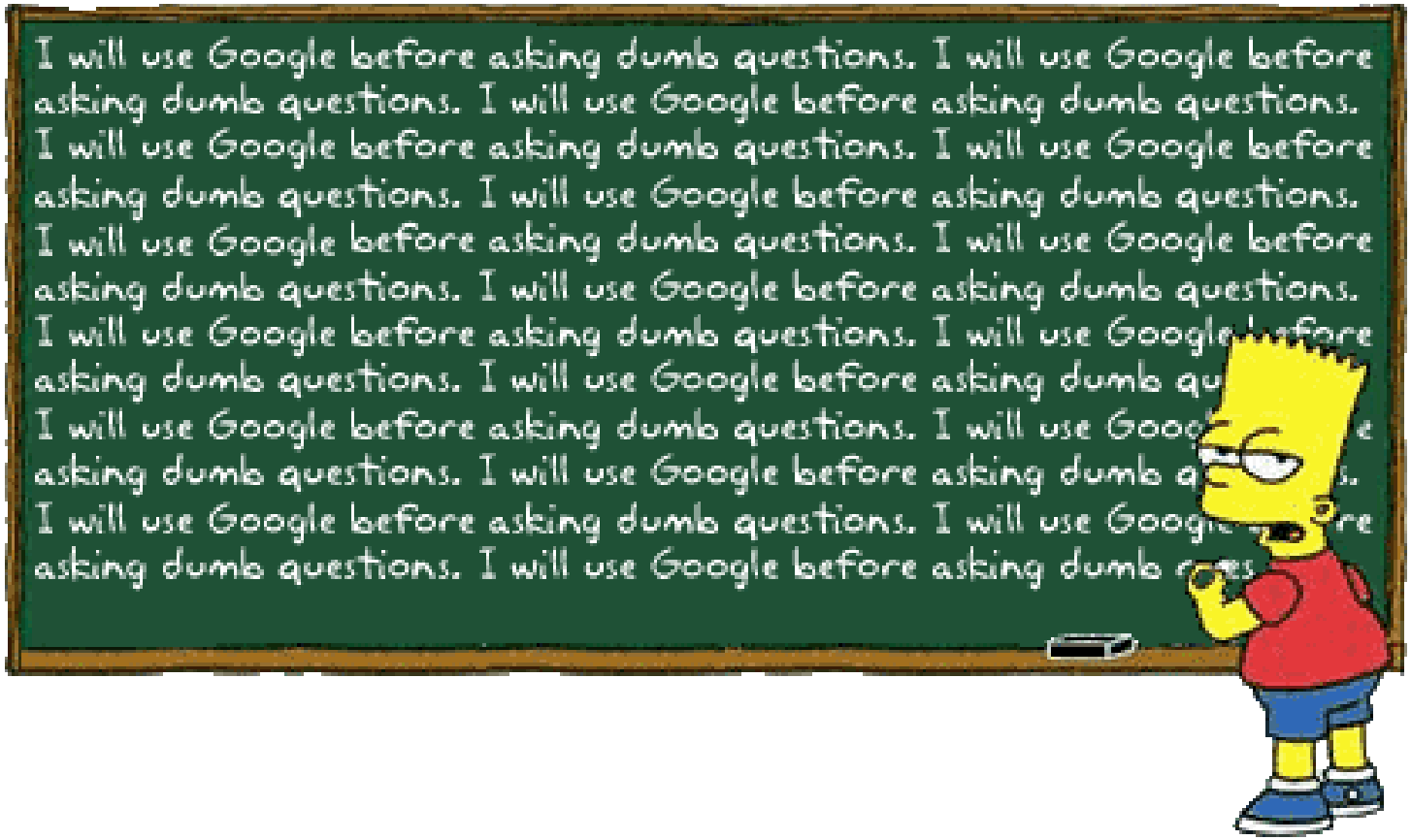


A simplified flow chart of GA process

GA Process

- GA represents an iterative process. Each iteration is called a **generation**. A typical number of generations for a simple GA can range from 50 to over 500. The entire set of generations is called a **run**.
- Because GAs use a **stochastic search** method, the fitness of a population may remain stable for a number of generations before a superior chromosome appears.
- A common practice is to terminate a GA after a specified number of generations and then examine the best chromosomes in the population. If no satisfactory solution is found, the GA is restarted.

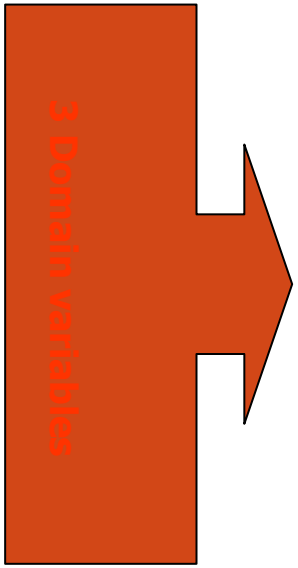
Question so far …



Example 1:
Burger and Profit Problem

The goal : To help the owner of a restaurant find the best combination of possible solutions (we will see in next slide) that produces the highest profit.

The Burger and Profit Problem: understanding the problem



- Decision To make: which is more profitable??:
  - Price : Should the price of the burger be RM1 or RM2?
  - Drink : Should coke or pepsi be served with the burger ?
  - Speed of Service : Should the restaurant provide

    slow, leisurely service by waiter with tuxedos or

    fast, snappy service by waiters in white polyester uniform ?

The Burger and Profit Problem: understanding the problem

- There are 3 decision variables each of which can assume one of two possible values.
- Assuming K is the possible value for the decision variable so K = 2.
- Assuming L is the length of string of possible strategy or combination so L = 3.
- The search space for this problem is $2^3 = 8$ possible strategies but we consider only 4 possible combinations.

The burger and profit Problem : Representation/encoding

- We can represent the problem (decision to make) in binary. i.e.
  - Price (0 represents RM1; 1 represents RM2)
  - Drink (0 represents Pepsi; 1 represents Cola)
  - Speed (0 represents Leisurely; 1 represents Fast)

Strategy	Price	Drink	Speed	Binary
1	RM1	Cola	Fast	011
2	RM1	Pepsi	Fast	001
3	RM2	Cola	Leisurely	110
4	RM1	Cola	Leisurely	010

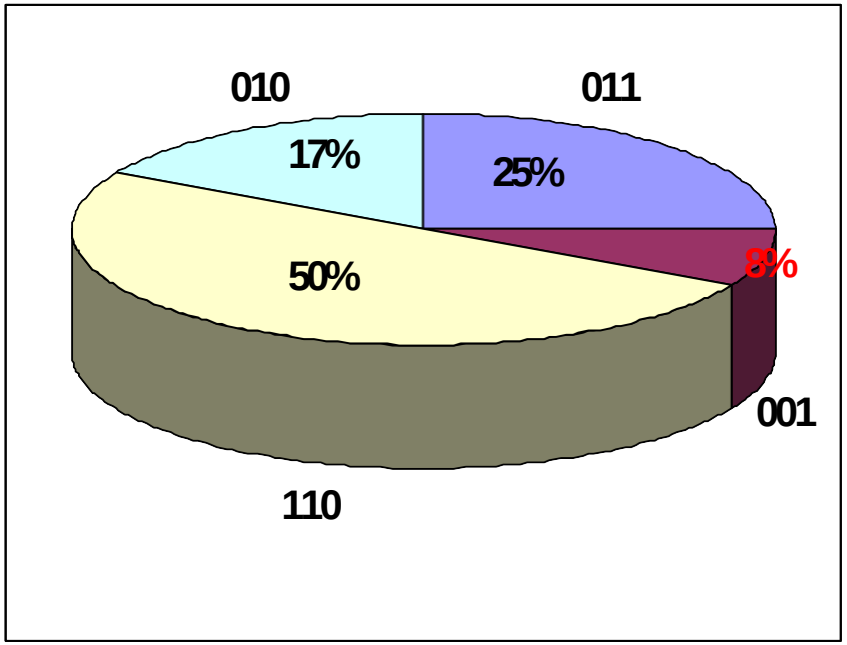
The burger and profit Problem : fitness function

- Fitness = Profit

Generation 0			
i	String X_i	Fitness $F(X_i)$	$\frac{f(X_i)}{\sum f(X_i)}$
1	011	3	.25
2	001	1	.08
3	110	6	.50
4	010	2	.17
Total		12	
Worst		1	
Average		3.00	
Best		6	

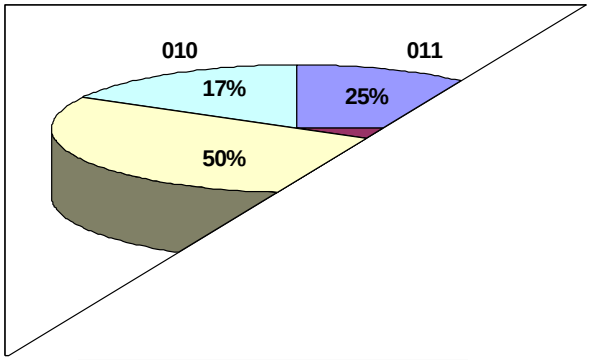
Fitness ratio

For Simplicity .. The fitness is evaluated based on the binary value



The burger and profit Problem : Selection/Reproduction

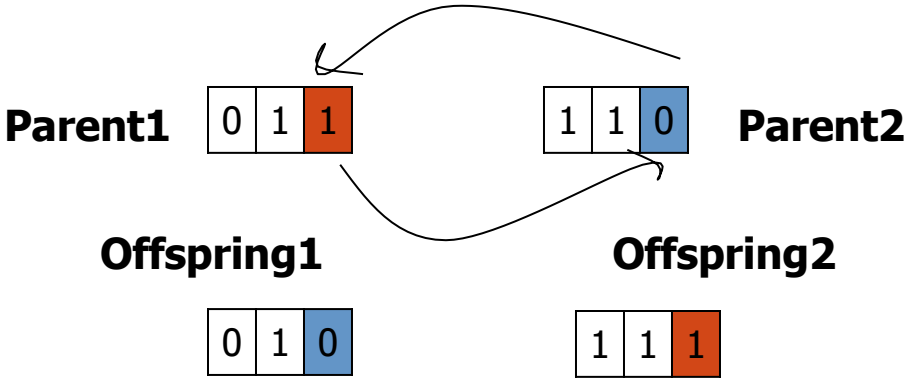
- Roulette Wheel Approach
  - 110 has possibility of 0.50 to reproduce so in the wheel it has 180 degree sector of being selected. In theory, it will be selected 2x in next generation.
  - 011 has possibility of 0.25 ie. ¼ chances to reproduce.
  - The same goes to the other string or individual.
  - BUT since GA is probabilistic, string 110 can be chosen 3X or even none in the next gen. The same applies to others.



Mating Pool Selected for Reproduction	
Mating Pool	Pool $F(X_i)$
011	3
110	6
110	6
010	2
Total	17
Worst	2
Average	4.25
Best	6

The burger and profit Problem : CrossOver

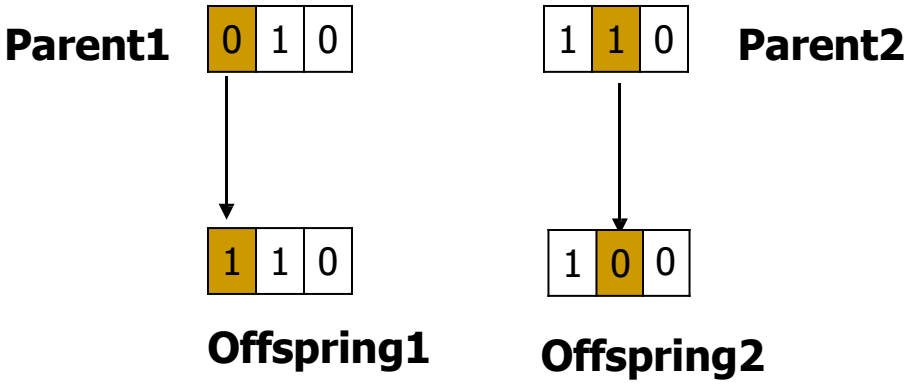
- Fixed Point Xover
  - Two parents are chosen based on theirs fitness i.e. 011 and 110 are selected.
  - Randomly select a point between 0 and L-1 (L = length of the string (3)).
  - Suppose the point is 2.



After Xover (Generation 1)		
Xover Point	String X_i	Fitness $F(X_i)$
2	111	7
2	010	2
-	110	6
-	010	2
Total		17
Worst		2
Average		4.25
Best		7

The burger and profit Problem: Mutation

- Mutation
  - Represents a change in the gene
  - Role to provide guarantee that the search algorithm is not trapped on a local optimum.
  - Parents are chosen randomly i.e. 010 and 110 are selected to be mutated
  - Randomly select a point between 0 and L-1 (L = length of the string=3).
  - Suppose the point is 0 and 1.



After Mutation (Generation 1)		
Mutation Point	String X_i	Fitness $F(X_i)$
-	111	7
1	110	6
2	100	4
-	010	2
Total		19
Worst		2
Average		4.75
Best		7

The burger and profit Problem : Result

Generation 0				Mating Pool Created After Reproduction		After Xover (Generation 1)			After Mutation (Generation 1)		
i	String X_i	Fitness $F(X_i)$	$\frac{f(X_i)}{\sum f(X_i)}$	Mating Pool	Pool $F(X_i)$	*Xover Point*	String X_i	Fitness $F(X_i)$	*Mutation Point*	String X_i	Fitness $F(X_i)$
1	011	3	.25	011	3	2	111	7	-	111	7
2	001	1	.08	110	6	2	010	2	1	110	6
3	110	6	.50	110	6	-	110	6	2	100	4
4	010	2	.17	010	2	-	010	2	-	010	2
Total		12			17			17			19
Worst		1			2			2			2
Average		3.00			4.25			4.25			4.75
Best		6			6			7			7

Example 2: Optimization of a function with one variable, x

The Task : Find the maximum value of the function ($15x - x\text{^}2$) where parameter x varies between 0 and 15. Assume for simplicity that x takes only integer values.

Answer these questions?

- How do we represent the problem in chromosomes like structure ?
- What should be the initial value of Num. Of Population *N, Xover Probability Pc , Mutation Probability Pm* and Num. Of generation *gen*
- What is the fitness function ?
- What kind of stochastic operators we want to use ?
- What will be the termination criterion ?

Recap..Steps in the GA development

1. Specify the problem, define constraints and optimum criteria;
2. Represent the problem domain as a chromosome;
3. Define a fitness function to evaluate the chromosome performance;
4. Construct the genetic operators;
5. Run the GA and tune its parameters.

Example 2: Solution Representation

- Since parameter x is between 0 – 15 and assumed to be integer value only, the solution can be represented as a string of "0" and "1" of size 4. Thus, chromosomes can be built with only four genes:

Integer (x)	Binary Code	Integer (x)	Binary Code	Integer (x)	Binary Code
1	0001	6	0110	11	1011
2	0010	7	0111	12	1100
3	0011	8	1000	13	1101
4	0100	9	1001	14	1110
5	0101	10	1010	15	1111

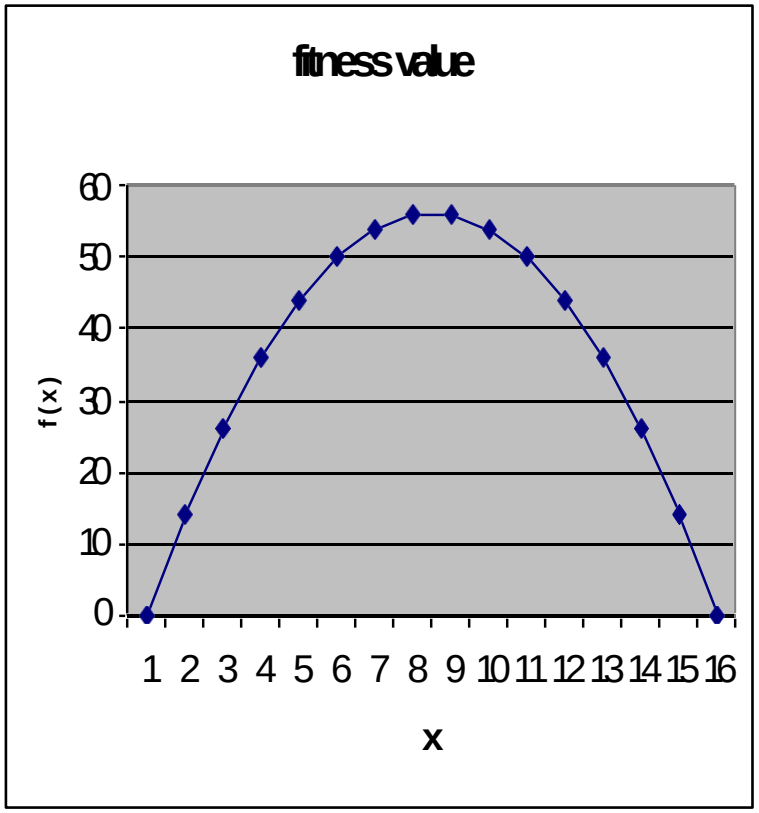
Example 2: Operator Parameters

- Number of Population, *N* , Probability of Mutation *Pm*, Cross over, *Pc* and number of generation are subject to be experimented.
- Rule of thumb:
  - *Pc should be big - Pm should be small – N should be medium (relative to the problems)*
- Let's choose
  - *N* = 6, *Pc = 0.7 and Pm = 0.01*

Example 2: The Fitness Function

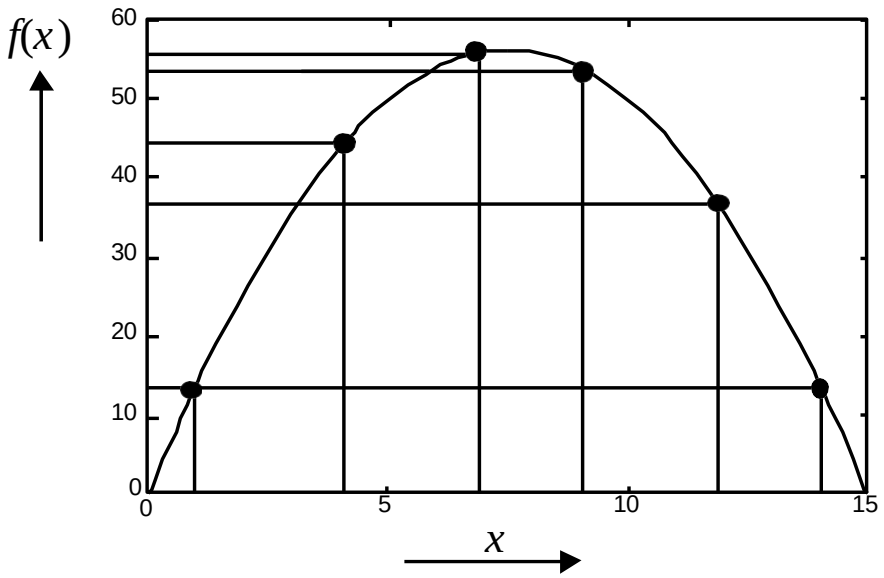
- Since we want to find the max of function (15x-x*x), the fitness function will be defined:

$$f(x) = 15x - x*x$$

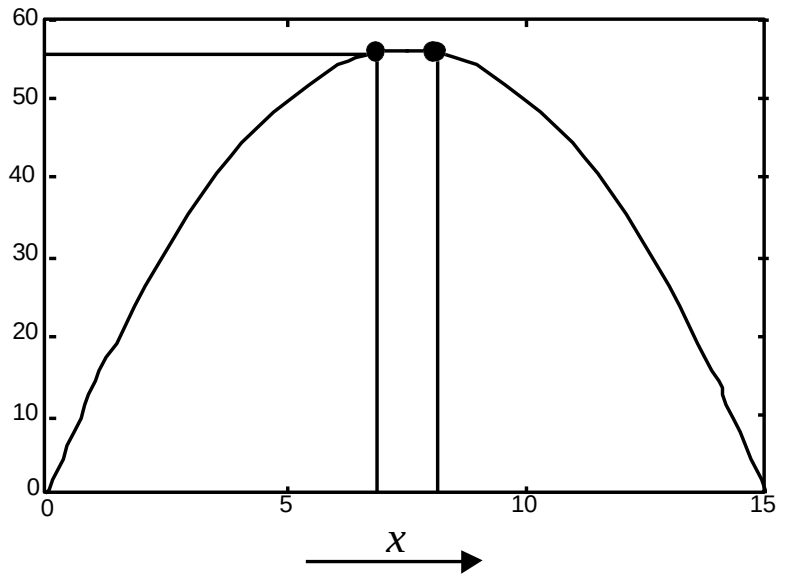


Example 2: The fitness function and chromosome locations

Chromosome label	*Chromosome string*	*Decoded integer*	*Chromosome fitness*
X1	1 1 0 0	12	
X2	0 1 0 0	4	
X3	0 0 0 1		
X4	1 1 1		
X5			
X6			



(*a*) Chromosome initial locations.



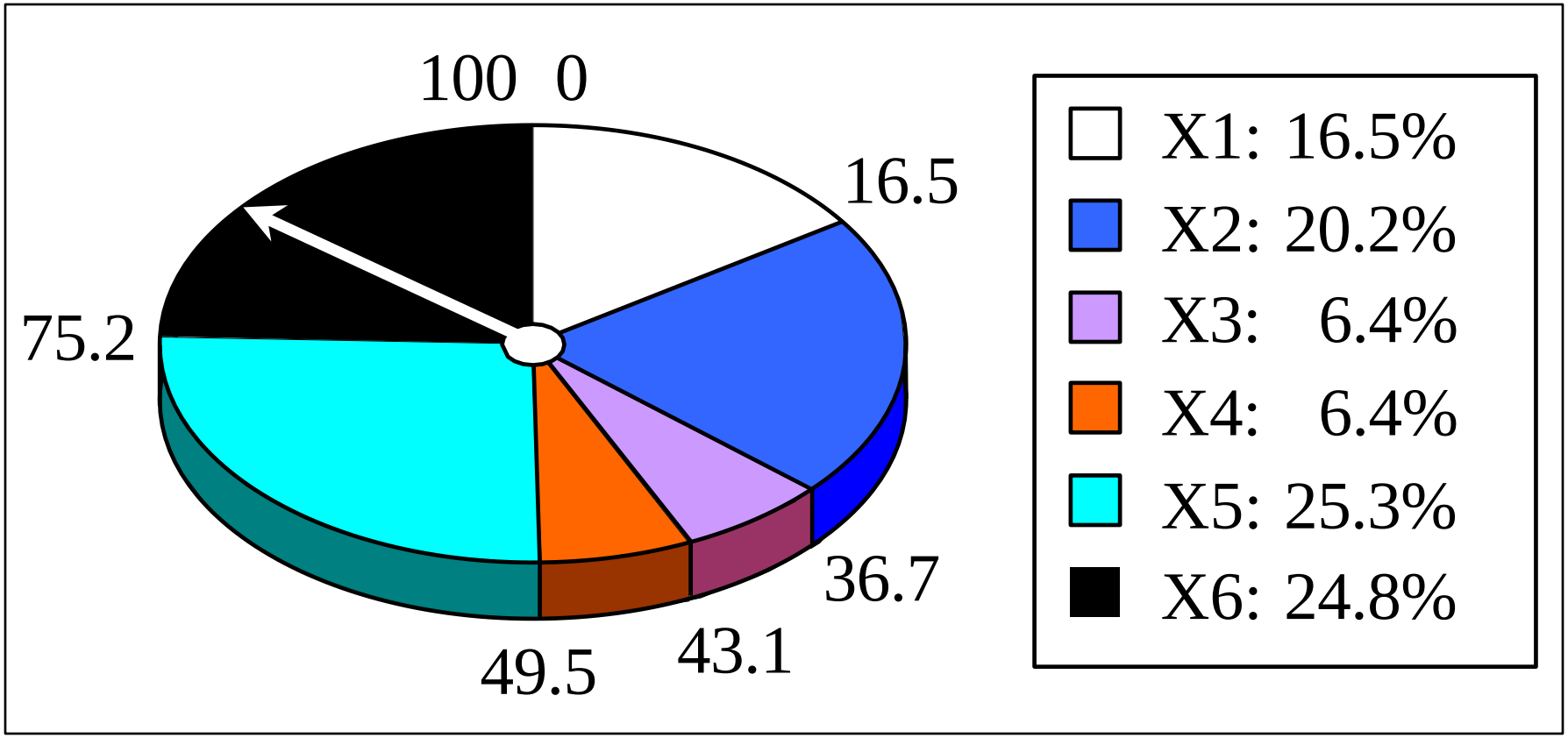
(*b*) Chromosome final locations.

Example 2: The fitness function and chromosome locations

- In natural selection, only the fittest species can survive, breed, and thereby pass their genes on to the next generation. GAs use a similar approach, but unlike nature, the size of the chromosome population remains unchanged from one generation to the next.
- The last column in Table shows the ratio of the individual chromosome's fitness to the population's total fitness. This ratio determines the chromosome's chance of being selected for mating. The chromosome's average fitness improves from one generation to the next.

Example 2: Roulette wheel selection

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



Example 2: Crossover operator

- In our example, we have an initial population of 6 chromosomes. Thus, to establish the same population in the next generation, the roulette wheel would be spun six times.

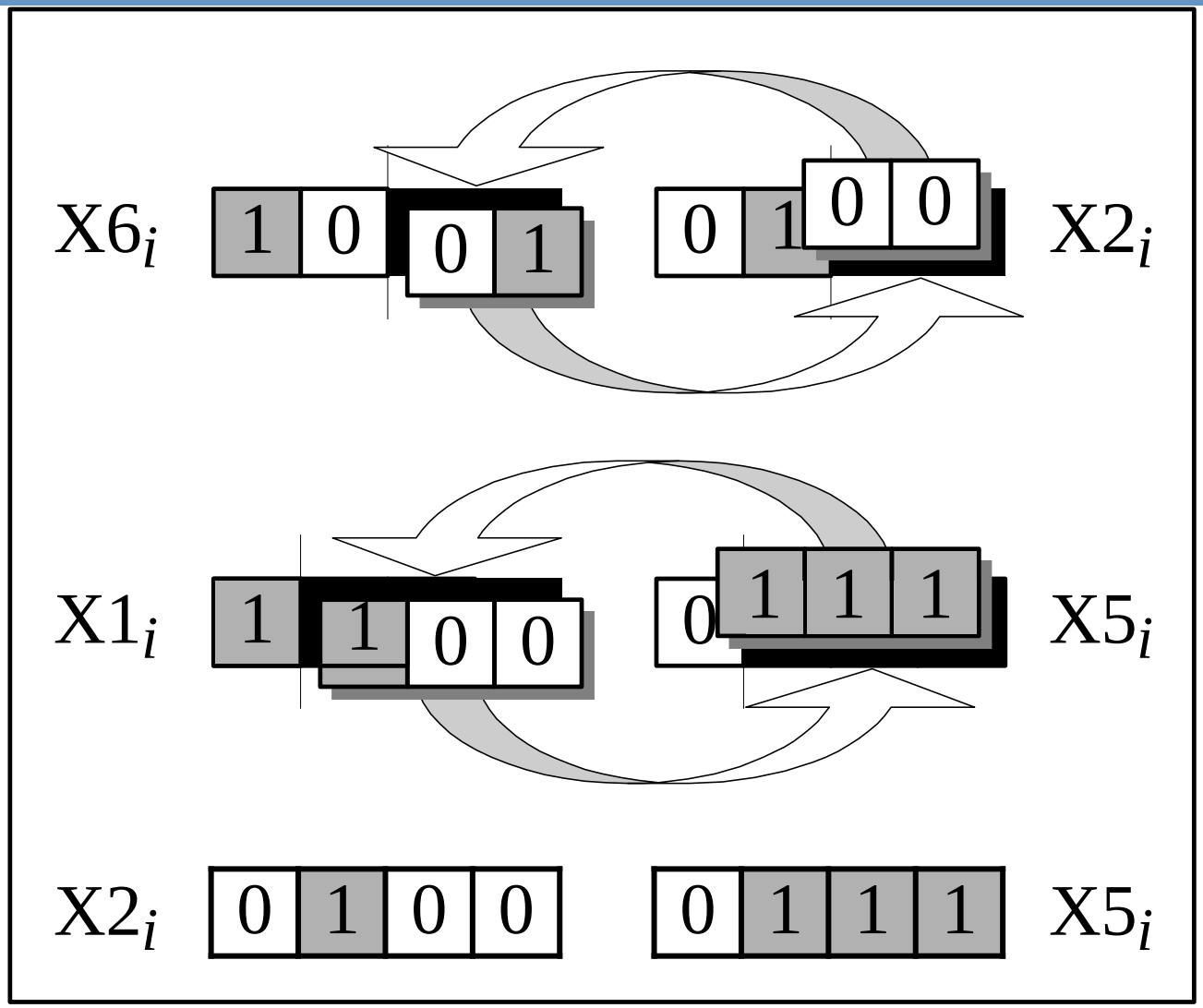
- Once a pair of parent chromosomes is selected, the **crossover** operator is applied.

Example 2: Crossover operator….

- First, the crossover operator randomly chooses a crossover point where two parent chromosomes "break", and then exchanges the chromosome parts after that point. As a result, two new offspring are created.

- If a pair of chromosomes does not cross over, then the chromosome cloning takes place, and the offspring are created as exact copies of each parent.

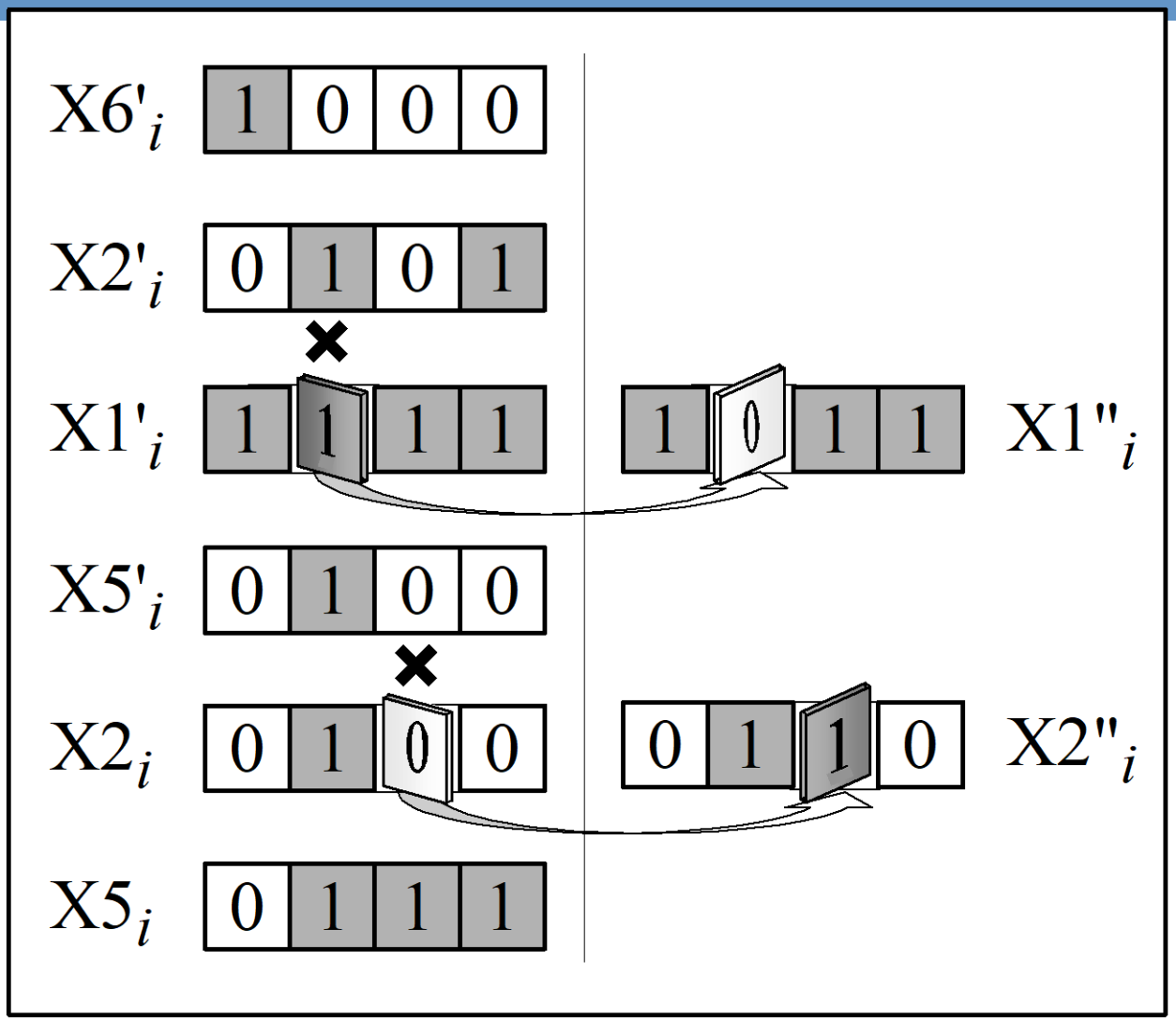
Example 2: Crossover



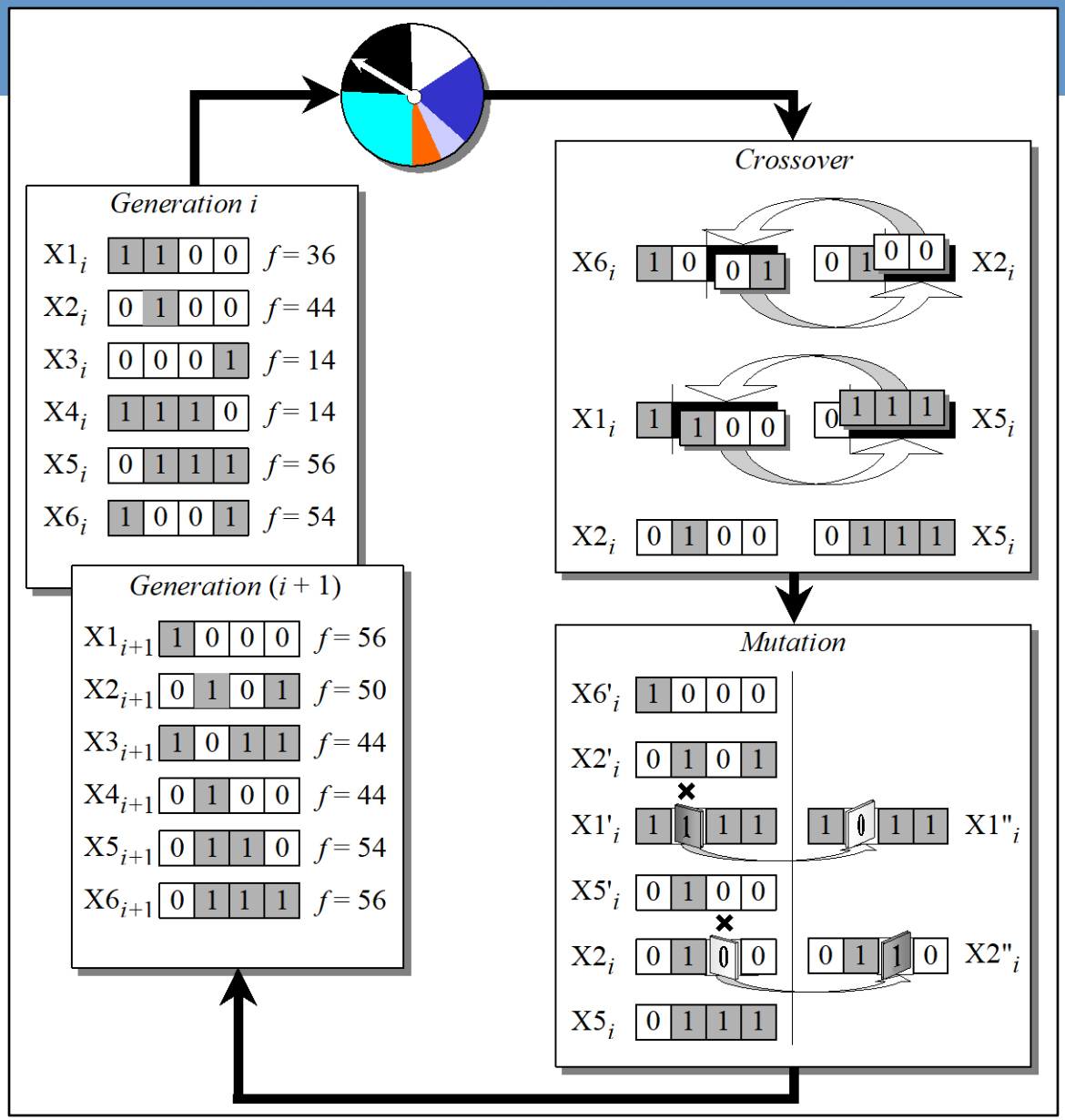
Example 2: Mutation operator

- Mutation represents a change in the gene.
- Mutation is a background operator. Its role is to provide a guarantee that the search algorithm is not trapped on a local optimum.
- The mutation operator flips a randomly selected gene in a chromosome.
- The mutation probability is quite small in nature, and is kept low for GAs, typically in the range between 0.001 and 0.01.

Example 2: Mutation



Example 2: The genetic algorithm cycle



Example 2: Stochastic Operators

- Selection / reproduction : Roulette Wheel
- Crossover : One point Xover – based on probability of Xover
- Mutation : Change gene based on probability of mutation

Example 2: Termination Criteria

- In most GAs application termination criterion will be number of generations. In this example, we will choose that criterion.

DEMO example 2

- Execute your Matlab program.
- Open file GA_1.m
- Evaluate the File ie. run it …

Summary of exercise 2

- The problem is simple and finite, i.e. we know the answer right away without using GA by looking at all possible solutions of parameter x =(0 to 15).

- This example however, shows how GA works in finding the optimum result by iteratively "evolve" in the environment.

Example 3: Optimization of two parameters x and y

- Find the maximum of the "peak" function of two variables:

$$f(x, y) = (1-x)^2 e^{-x^2-(y+1)^2} - (x - x^3 - y^3) e^{-x^2-y^2}$$

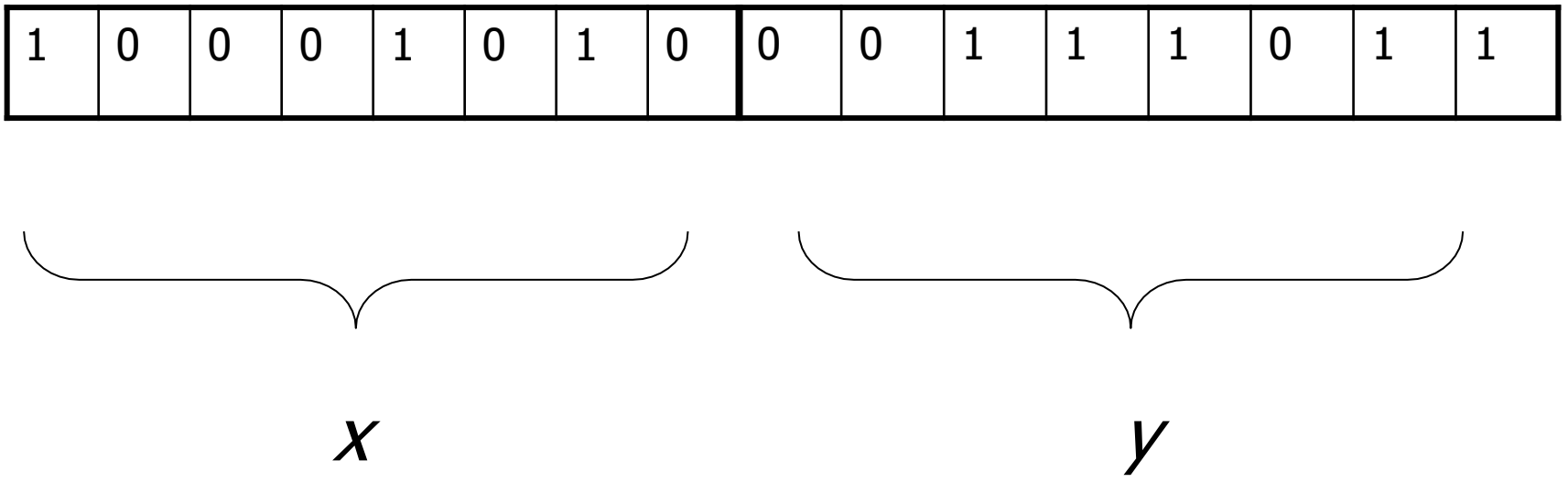
where parameters x and y vary between –3 and 3.

Answer these questions?

- How do we represent the problem in chromosomes like structure ?
- What is the fitness function ?
- What kind of stochastic operators we want to use ?
- What should be the initial value of Num. Of Population *N*, *Xover Probability Pc* , *Mutation Probability Pm* and Num. Of generation *gen*
- What will be the termination criterion ?

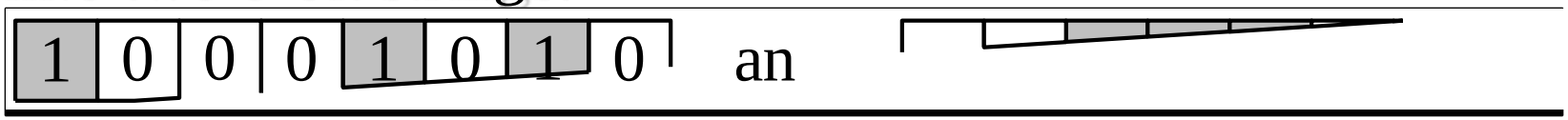
Example 3: Solution Representation

- The first step is to represent the problem variables as a chromosome
- Since parameter *x* and y is between -3 and +3, we can represent it in concatenated binary form of 8 bits for *x* and 8 bits for *y*



Example 3: Solution Representation

- We also choose the size of the chromosome population, for instance 6, and randomly generate an initial population.
- The next step is to calculate the fitness of each chromosome. This is done in two stages.
- First, a chromosome, that is a string of 16 bits, is partitioned into two 8-bit strings:



- Then these strings are converted from binary (base 2) to decimal (base 10):

$$(10001010)_2 = 1\times 2^7 + 0\times 2^6 + 0\times 2^5 + 0\times 2^4 + 1\times 2^3 + 0\times 2^2 + 1\times 2^1 + 0\times 2^0 = (138)_{10}$$

and

$$(00111011)_2 = 0\times 2^7 + 0\times 2^6 + 1\times 2^5 + 1\times 2^4 + 1\times 2^3 + 0\times 2^2 + 1\times 2^1 + 1\times 2^0 = (59)_{10}$$

Example 3: Solution Representation

- Now the range of integers that can be handled by 8-bits, that is the range from 0 to $(2^8 - 1)$, is mapped to the actual range of parameters *x* and *y*, that is the range from –3 to 3:

$$\frac{6}{256-1} = 0.0235294$$

- To obtain the actual value of *x* and *y*, multiply their decimal values by 0.0235294 and subtract 3 from the result

$$x = 138 * 0.0235294 - 3 = 0.2470588$$

$$y = 59 * 0.0235294 - 3 = -1.6117647$$

Example 3: The Fitness Function

- Using decoded values of *x* and *y* as inputs in the mathematical function, the GA calculates the fitness of each chromosome
- The fitness function would be the equation

$$f(x, y) = (1 - x)^2 e^{x^2 - (y+1)^2}$$

f(0.2470588, -1.6117647)= .....

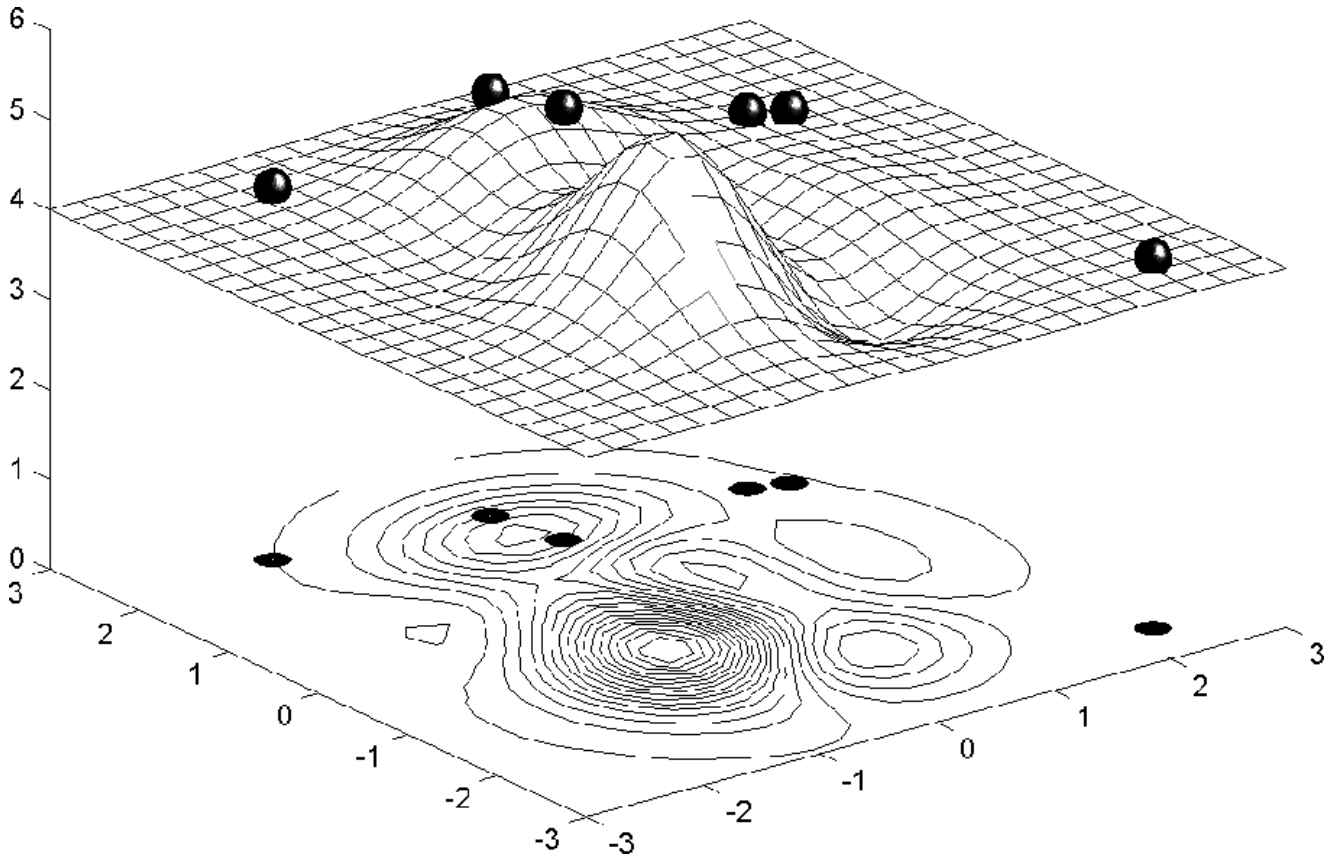
GA Process

- Using decoded values of *x* and *y* as inputs in the mathematical function, the GA calculates the fitness of each chromosome.
- To find the maximum of the "peak" function, we will use crossover with the probability equal to 0.7 and mutation with the probability equal to 0.001. As we mentioned earlier, a common practice in GAs is to specify the number of generations. Suppose the desired number of generations is 100. That is, the GA will create 100 generations of 6 chromosomes before stopping.

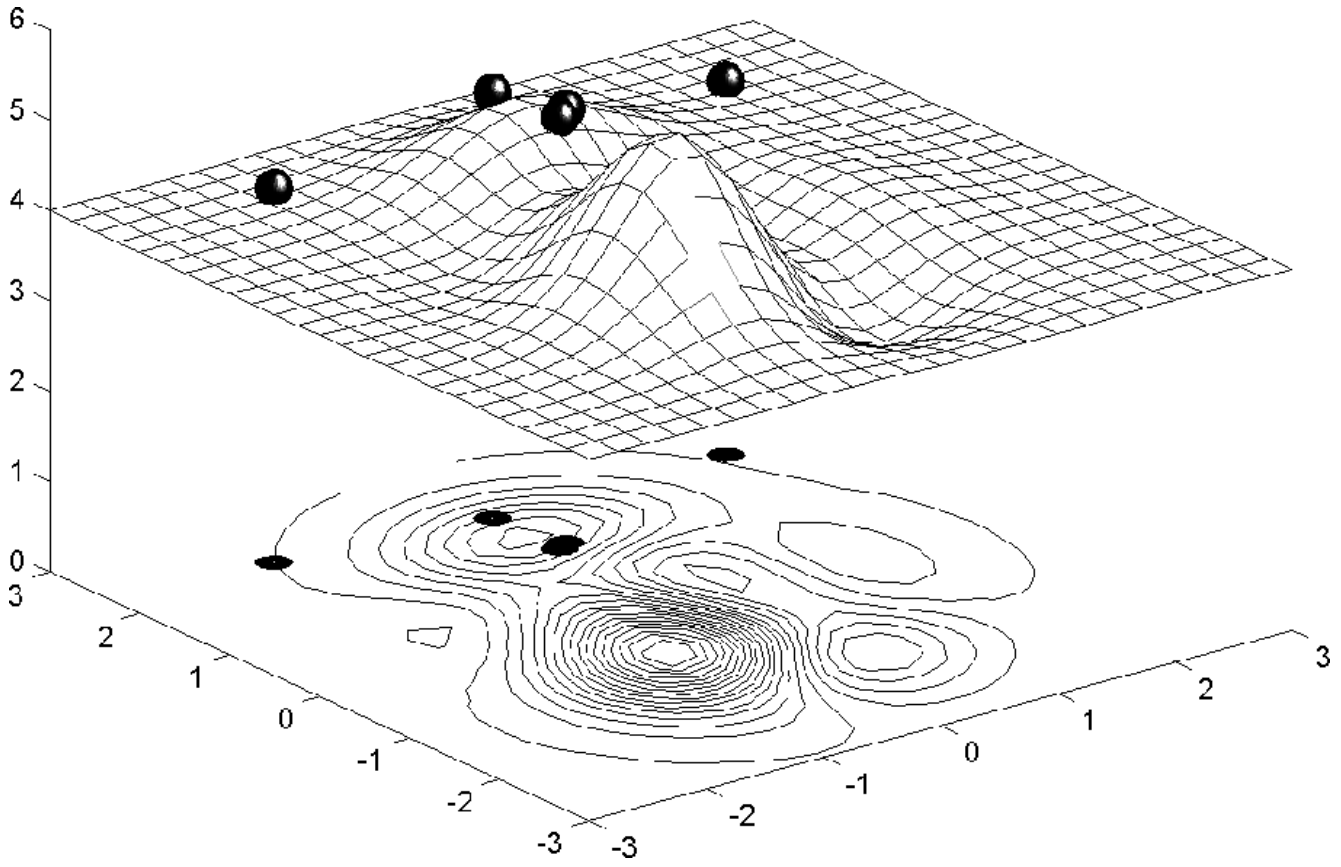
Recap..Steps in the GA development

1. Specify the problem, define constraints and optimum criteria;
2. Represent the problem domain as a chromosome;
3. Define a fitness function to evaluate the chromosome performance;
4. Construct the genetic operators;
5. Run the GA and tune its parameters.

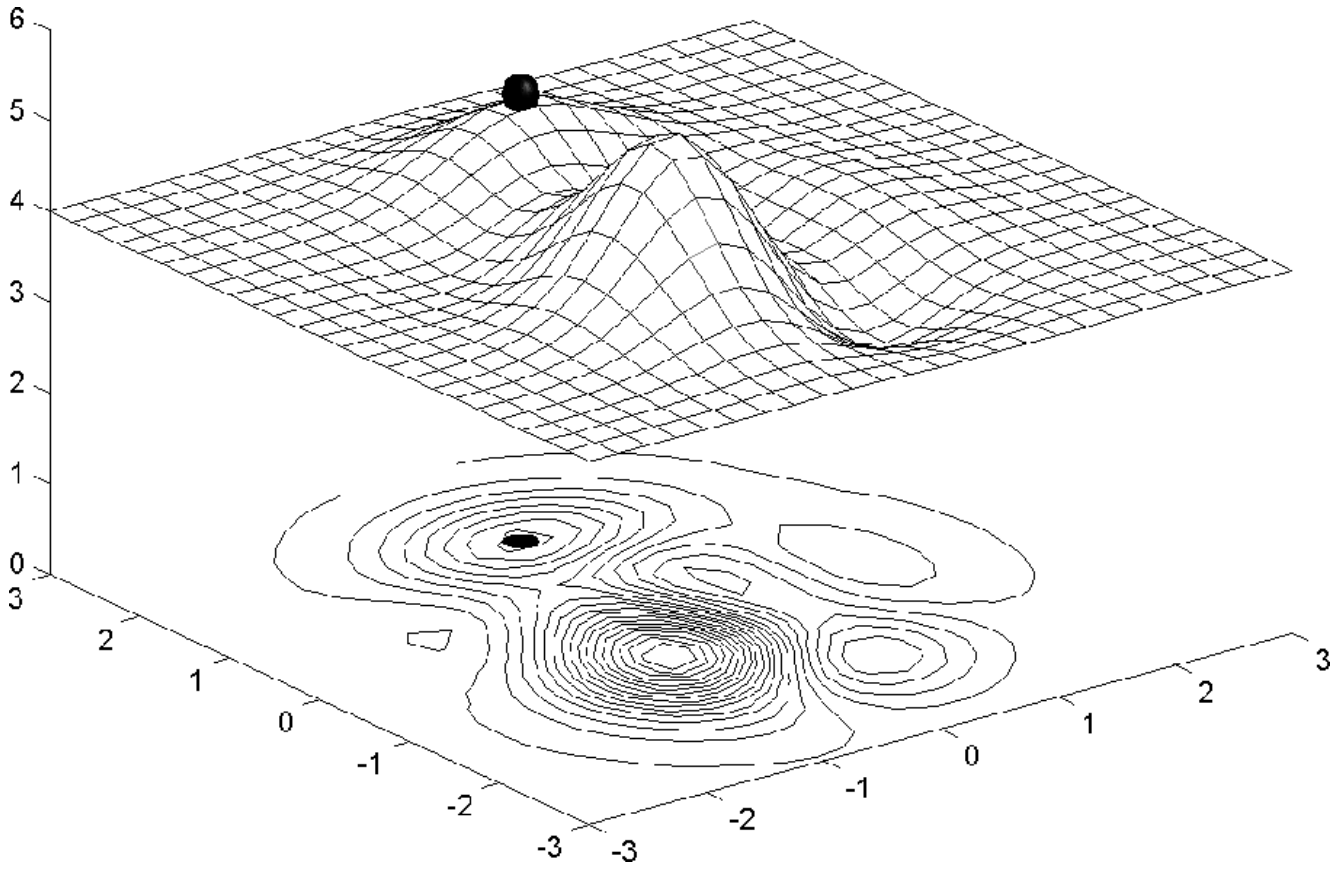
Chromosome locations on the surface of the "peak" function: initial population



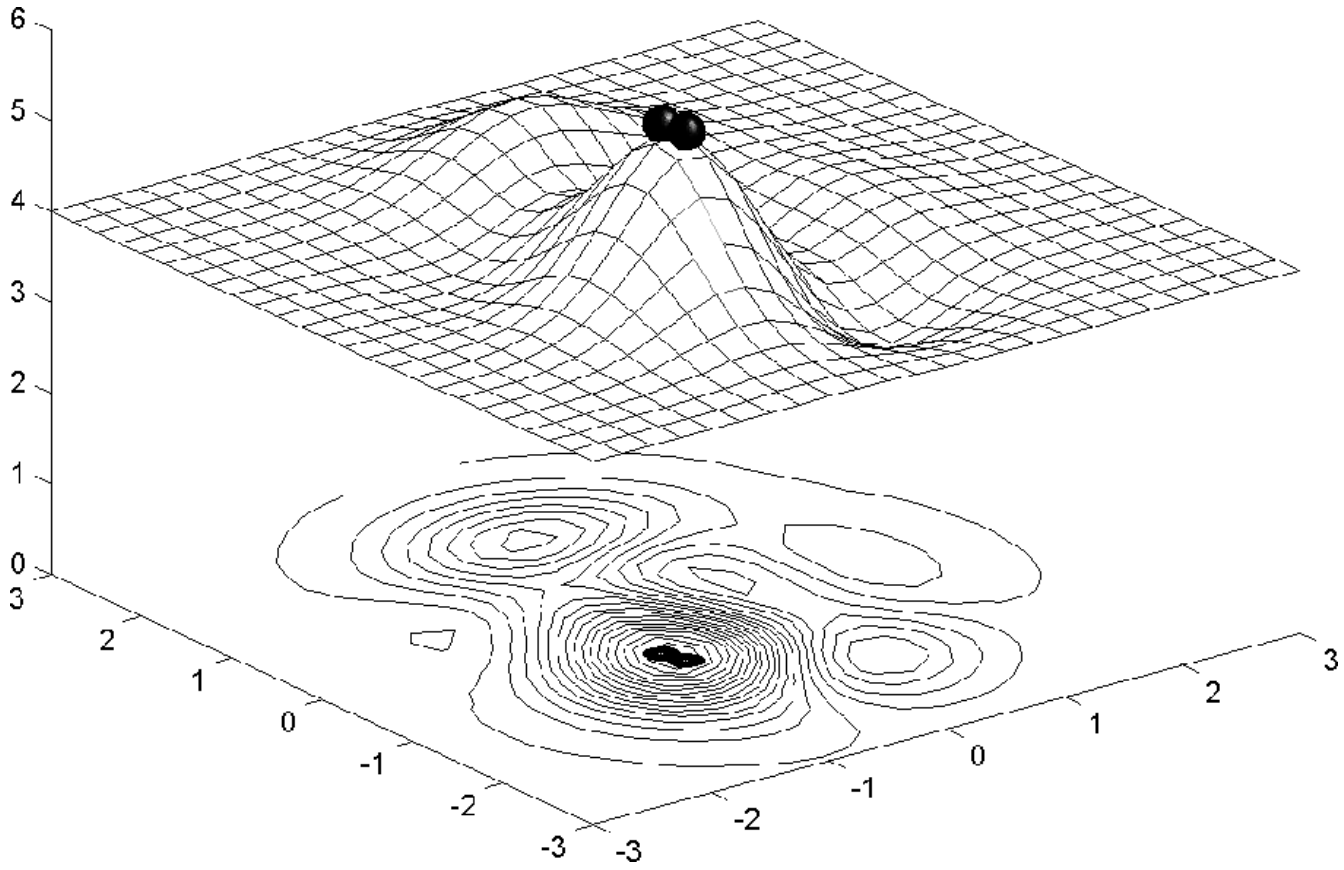
Chromosome locations on the surface of the "peak" function: first generation



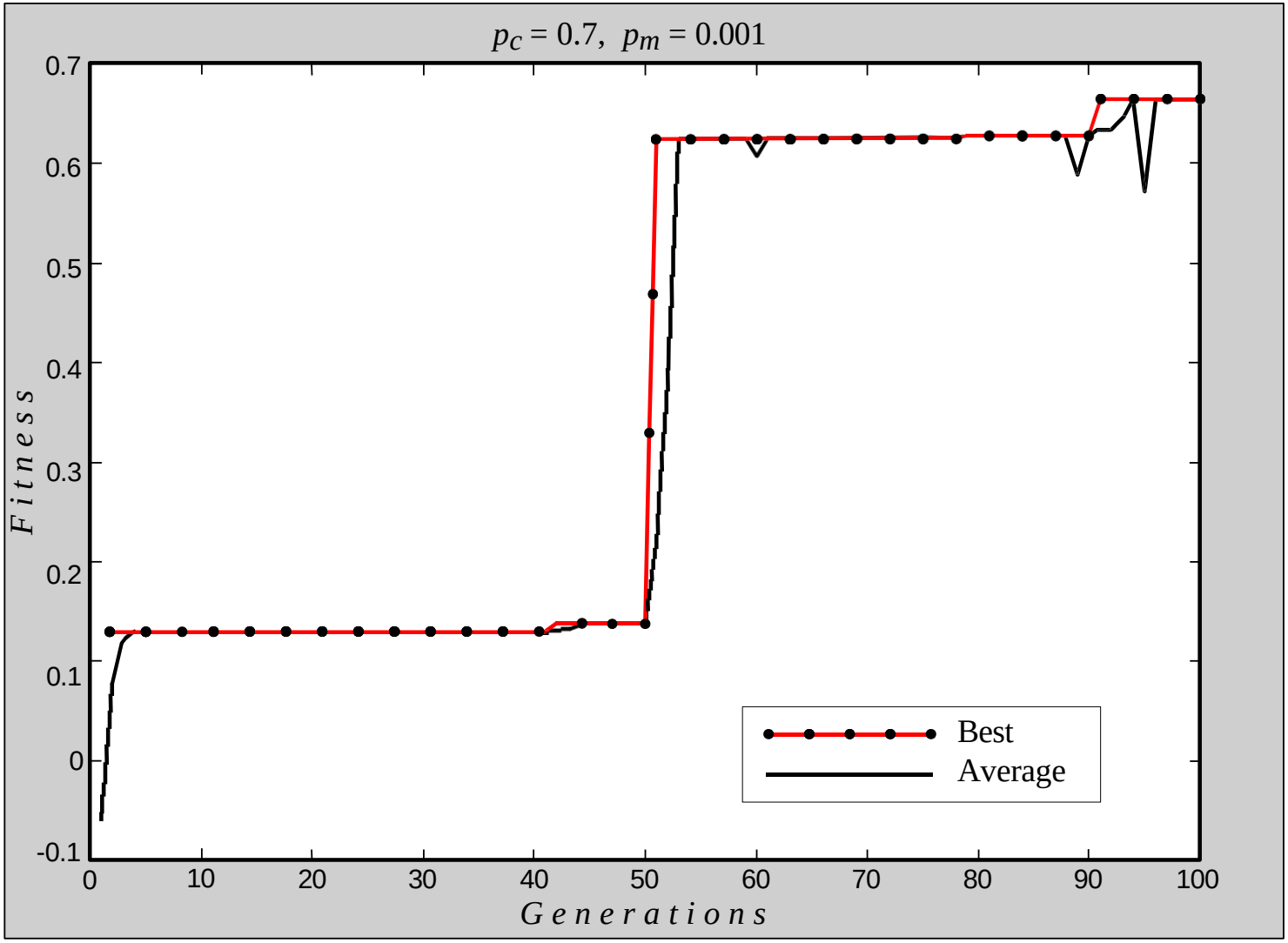
Chromosome locations on the surface of the "peak" function: local maximum



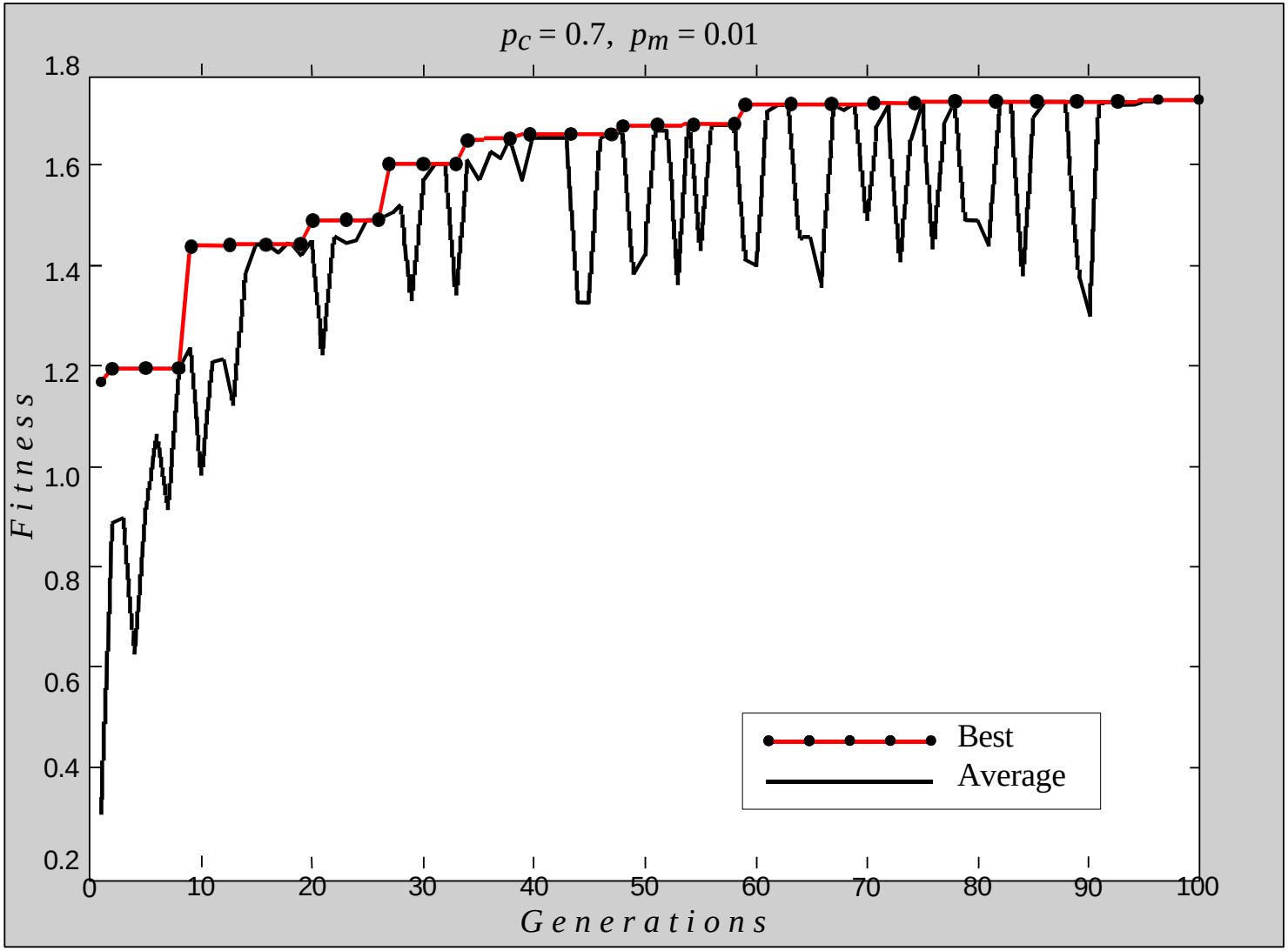
Chromosome locations on the surface of the "peak" function: global maximum



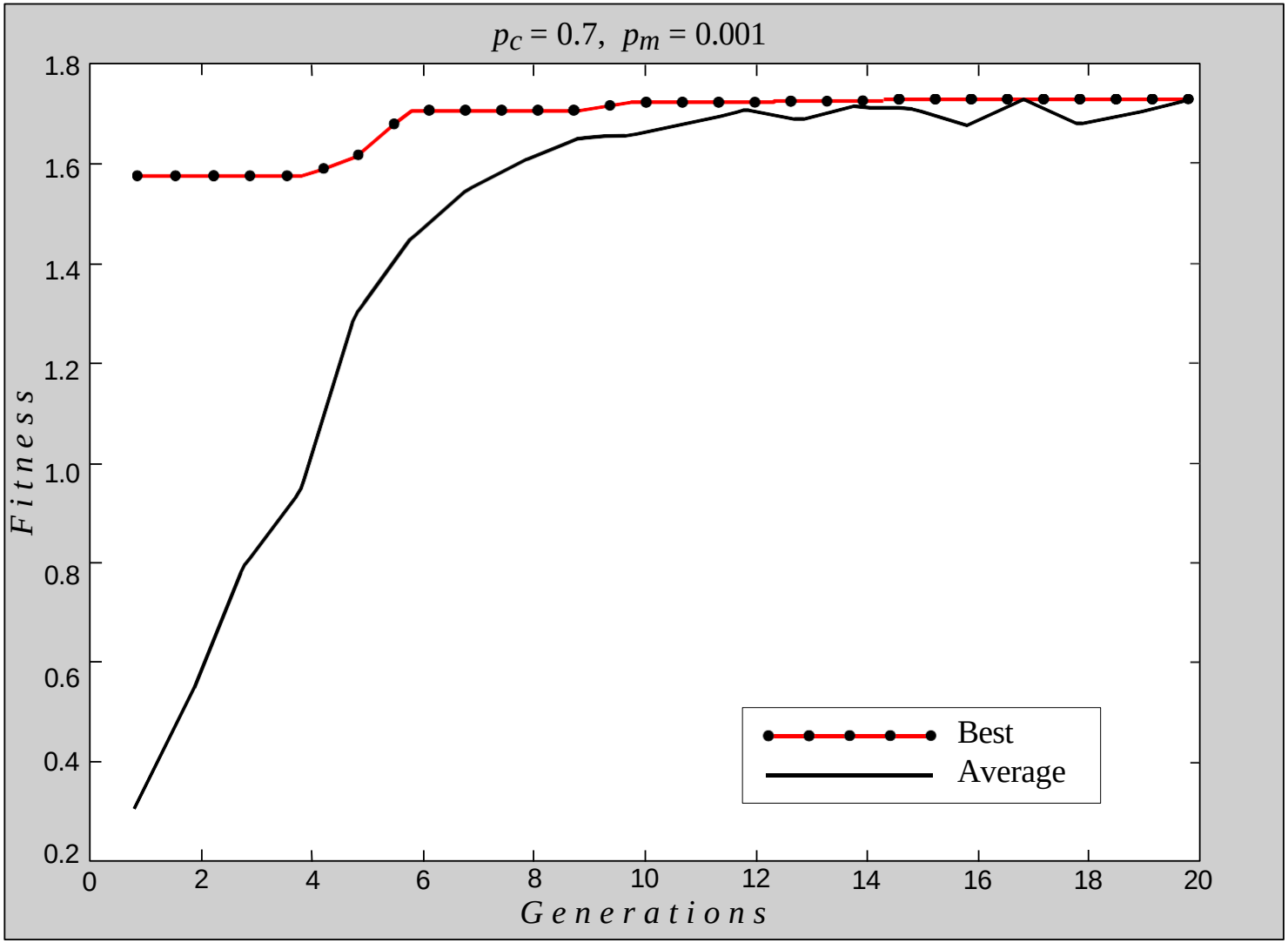
Performance graphs for 100 generations of 6 chromosomes: local maximum



Performance graphs for 100 generations of 6 chromosomes: global maximum



Performance graphs for 20 generations of 60 chromosomes



Example 3: Stochastic Operators

- SAME AS BEFORE…
- Selection / reproduction : Roulette Wheel
- Crossover : One point Xover – based on probability of Xover
- Mutation : Change gene based on probability of mutation

Example 3: Termination Criteria

- Num of generation … any other suggestion ?

DEMO example 3

- Execute your Matlab program.
- Open file GA_2.m
- Evaluate the Result ie. change the parameter and see the effects ..

Example 4: Traveling Salesman Problem (TSP)

The task : Given a finite number of cities, *N* and the cost of travel (or distance) between each pair of cities, we need to find the cheapest way (or shortest route) for visiting each city exactly once and returning to the start point.

Example 4: TSP Continue ..

- The TSP is represented in numerous transportation and logistics applications such as
  - Arranging routes for school buses to pick up children in a school district
  - Delivering meals to home-bound people
  - Scheduling stacker cranes in a warehouse
  - Planning truck routes to pick up parcel post and many other ..
  - A classic example of the TSP is the scheduling of a machine to drill holes in a circuit board.

Example 4: Answer these questions?

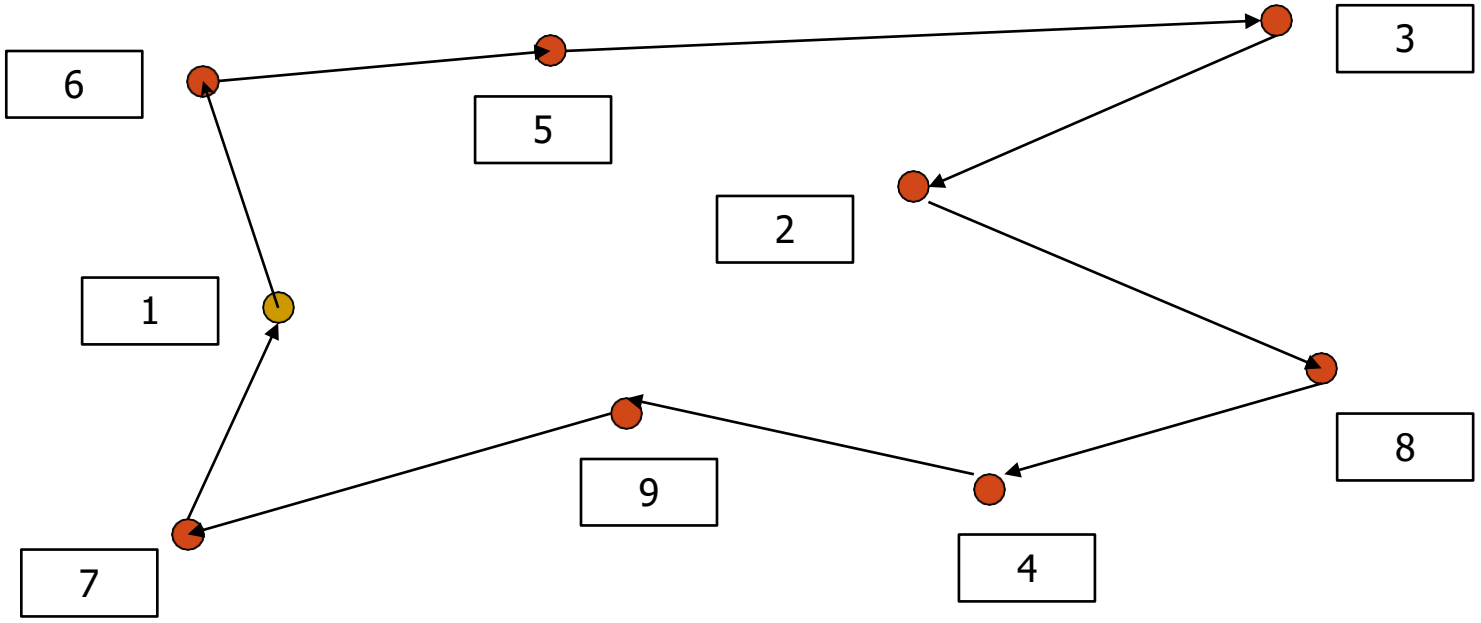
- How do we represent the problem in chromosomes like structure ?
- What should be the initial value of Num. Of Population *N, Xover Probability* P_c*, Mutation Probability* P_m and Num. Of generation *gen*
- What is the fitness function ?
- What kind of stochastic operators we want to use ?
- What will be the termination criterion ?

Example 4: Solution Representation

- We need to represent the route of the salesmen in chromosomes like structure – how ?
- An easy answer for that would be label the cities with alphabets or number and each possible route will represent the path.
- Example .. Suppose we have 9 cities, so the chromosomes would be like ..

1	6	5	3	2	8	4	9	7

TSP.. continue : the path representation



1	6	5	3	2	8	4	9	7

Example 4: Operator Parameters

- Subject to be experimented …same as before
- Number of Population, *N* , Probability of Mutation *Pm*, Cross over, *Pc* and number of generation are subject to be experimented.
- Rule of thumb:
  - *Pc should big - Pm should small – N should in medium size (relative to the problems)*

Example 4: The Fitness Function

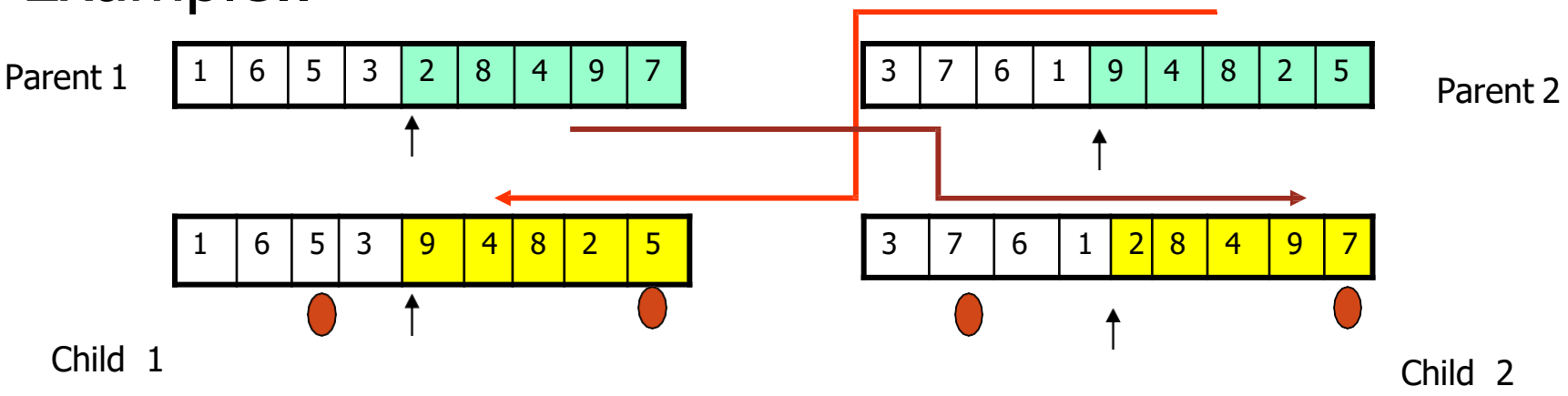
- If we want to find the shortest path, then distance between each pair of cities would be the fitness function ➔ the shorter the route the fitter the chromosomes
- If we want to find the optimal cost, then fare / expenses between each pair of cities would be the fitness function.

Example 4: Stochastic Operators

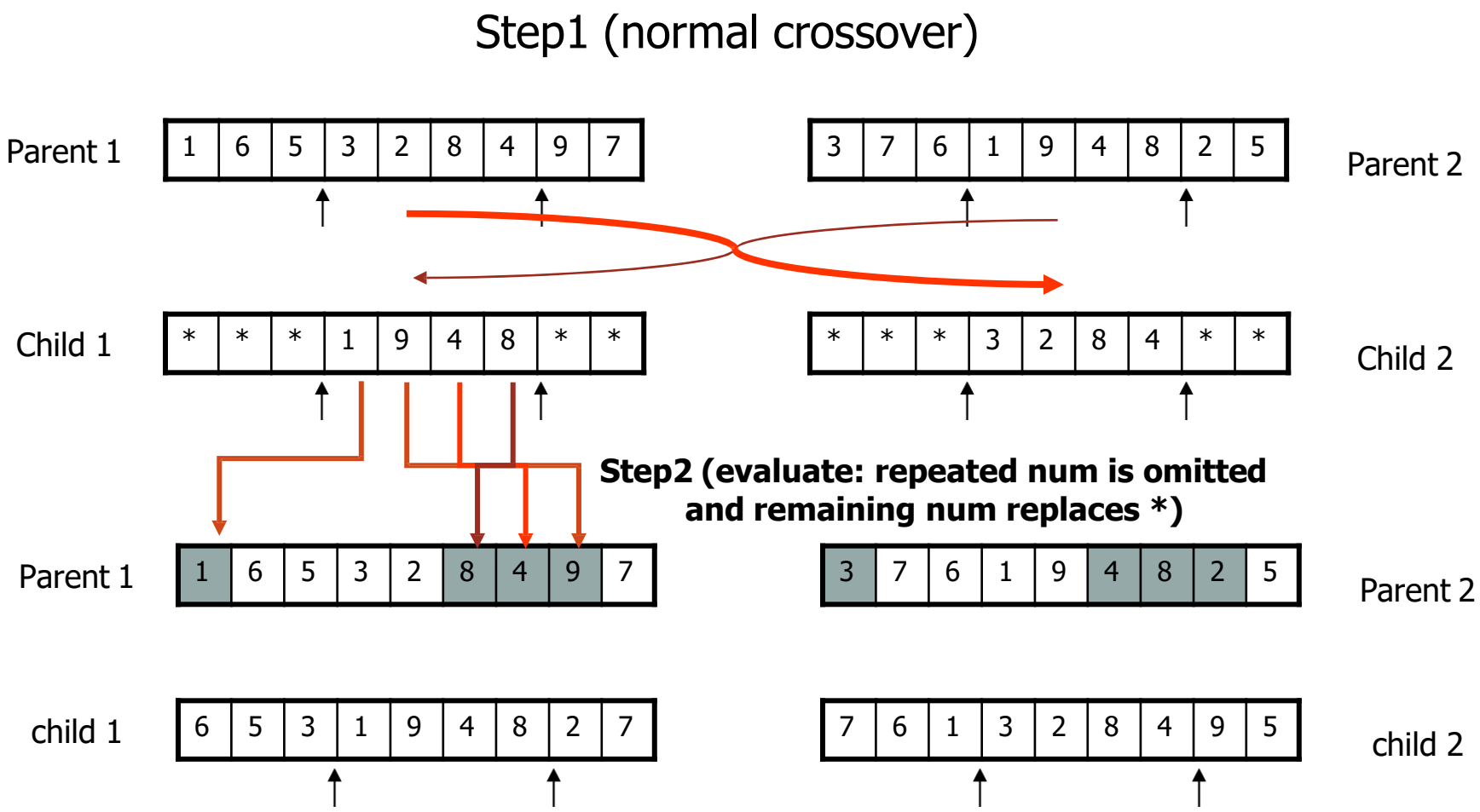
- SAME AS BEFORE…HOWEVER NOTE THAT IT IS NOT THAT EASY FOR MUTATION AND CROSSOVER ..

Example 4: CROSSOVER

- The crossover in its classical form cannot be directly applied to TSP. A simple exchange of parts between two parents would produce illegal routes containing duplicates and omissions – some cities would be visited twice while some others would not be visited at all
- Example..



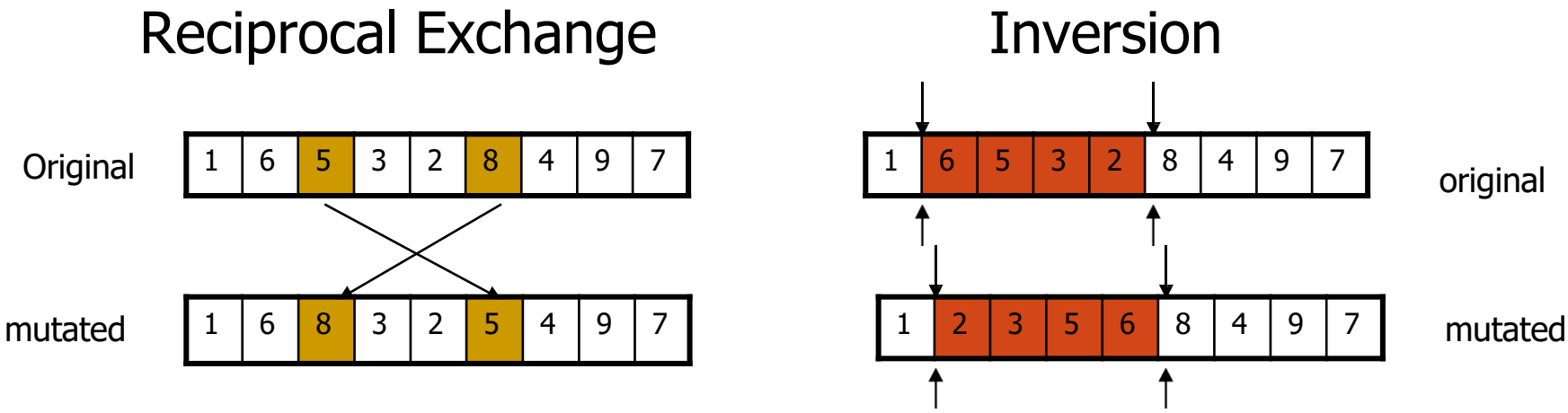
Example 4: CROSSOVER for TSP



Example 4: Mutation For TSP

- There are two types of mutation operators for TSP
  - Reciprocal exchange & inversion
- Reciprocal exchange operator simply swaps two randomly selected cities in the chromosome.
- Inversion operator selects two randomly point along the chromosome string and reverses the order of the cities between these points

Example 4: Mutation For TSP



Example 4: Termination Criteria

- Of course Num of generation again …
- Try with some numbers ie. 60 , 100, 120 etc observe the effect

DEMO example 4

- Execute your Matlab program.
- Open file GA_TSP.m
- Evaluate the File ie. change the parameter and see the effects ..

Summary of GA

- Developed: USA in the 1970's by Holland and his students
- A class of probabilistic optimization algorithms
- Inspired by the biological evolution process
- Uses concepts of "Natural Selection" and "Genetic Inheritance" (Darwin 1859)
- Typically applied to:
  - discrete optimization (ex. TSP)
- Attributed features:
  - good solver for combinatorial problems
- Special:
  - many variants, e.g., reproduction models, operators
  - formerly: *the* GA, nowdays: *a* GA, GA*s*

Summary : What you have learned ..

- The processes involved in GA .. – stochastic operators
- How to represent some problems into chromosomes like structure ..
- How to do Xover & mutation …
- Understand the effect of *N* ,*Pm* and *Pc* and also number of generation as termination criteria
- Basically how GAs works …

What we have not covered ..

- .. To name a few
  - We have not got into details on techniques in the stochastic operators.
  - Schemata theory to prove GA does work to solve NP-problems
  - Some other alternative to represent chromosomes like structure ie. bits of “0” & “1”, floating points, integers, rules, symbols etc.
  - ……

The End