Unit 4

Syllabus

Neuro-Fuzzy Systems:
Neuro-fuzzy modeling,
Neuro-fuzzy control.
Genetic Algorithms- Simple GA, crossover and mutation, genetic algorithms in search and optimization.

Introduction to Neuro-Fuzzy Systems

Neuro-Fuzzy Systems

- The combination of **fuzzy logic** and **neural network** technology is called "**Neuro-Fuzzy**" and combines the advantages of the two technologies.

Neuro-Fuzzy Systems (contd.)

Neural Network

Good for learning.

- Supervised leaning
- Unsupervised learning
- Reinforcement learning

Not good for human to interpret its internal representation.

+

Fuzzy Logic

Human reasoning scheme.

- Readable Fuzzy rules
- Interpretable

Fuzzy rules and membership functions are subjective.

Neuro-Fuzzy Systems (contd.)

Neural Network + Fuzzy Logic

A *neuro-fuzzy system* is a *fuzzy system* that uses a *learning algorithm* derived from or inspired by *neural network* theory to determine its *parameters* by processing data samples.

Neuro-Fuzzy Systems (contd.)

fuzzy sets and
fuzzy rules

A *neuro-fuzzy system* is a *fuzzy system* that uses a *learning algorithm* derived from or inspired by *neural network* theory to determine its *parameters* by processing data samples.

Combining Neural and Fuzzy

- The key benefit of fuzzy logic is, that it lets you **describe desired system behavior with simple "if-then" relations**. In many applications, this gets you a simpler solution in less design time.
- In addition, you can use all available engineering know-how to optimize the performance directly. While this is certainly the beauty of fuzzy logic, it at the same time is its major limitation.
- In many applications, knowledge that describes desired system behavior is contained in data sets. Here, the designer has to derive the "if-then" rules from the data sets manually, which requires a major effort with large data sets.
- When data sets contain knowledge about the system to be designed, a neural net promises a solution as **it can train itself from the data sets**.
- However, only few commercial applications of neural nets exist. This is a contrast to fuzzy logic, which is a very common design technique in Asia and Europe.

Sparse use of neural nets in applications

The sparse use of neural nets in applications is due to a number of reasons.

- First, neural net solutions remain a "black box". You can neither interpret what causes a certain behavior nor can you modify a neural net manually to change a certain behavior.
- Second, neural nets require prohibitive (too expensive) computational effort for most mass-market products.
- Third, selection of the appropriate net model and setting the parameters of the learning algorithm is still a "black art" and requires long experience.
- Of the aforementioned reasons, the lack of an easy way to verify and optimize a neural net solution is probably the major limitation.

Combining Neural and Fuzzy

- In simple words, both neural nets and fuzzy logic are powerful design techniques that have its strengths and weaknesses.
- Neural nets **can learn from data sets** while **fuzzy logic solutions are easy to verify and optimize**.
- If you look at these properties in a portfolio, the idea becomes obvious, that a clever combination of the two technologies delivers best of both worlds.
- Combine the **explicit knowledge representation of fuzzy logic with the learning power of neural nets, and you get NeuroFuzzy.**

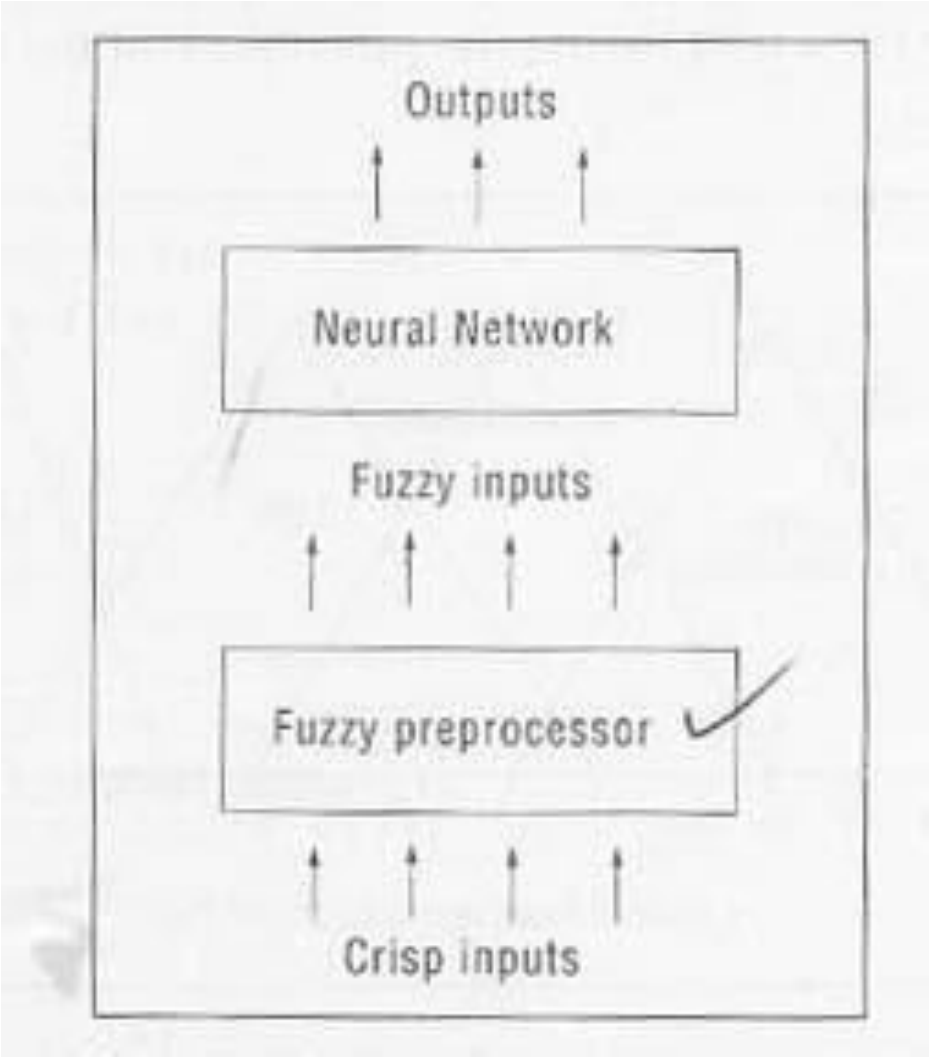
Training Fuzzy Logic Systems with NeuroFuzzy

- Many alternative ways of integrating neural nets and fuzzy logic have been proposed in the scientific literature.
- The first artificial neural net implementation dates over 50 years back.
- Since then, most research dealt with learning techniques and algorithms.
- One major milestone in the development of neural net technology was the invention of the so-called error back propagation algorithm about ten years ago.

Neuro-Fuzzy systems

- There are a number of ways fuzzy logic can be used with neural networks.
- Perhaps the simplest way to use a fuzzifier function to preprocess or post process data for a neural network.
- In the figure, neural network has a preprocessing fuzzifier that converts data into fuzzy data for application to a neural network.

A neural network with fuzzy preprocessor



Neuro-fuzzy approach

- Presently, the neuro-fuzzy approach is becoming one of the major areas of interest because it gets the benefits of neural networks as well as of fuzzy logic systems and it removes the individual disadvantages by combining them on the common features.
- Different architectures of neuro-fuzzy system have been investigated by number of researchers such as Lin (1994), Medsker (1995) and Jana (1996).
- These architectures have been applied in many applications especially in the process control.

Neuro-fuzzy approach

- Ever since fuzzy systems were applied in industrial applications, developers know that the construction of a well performing fuzzy system is not always easy.
- The **problem of finding appropriate membership functions and fuzzy rules in often a tiring process of trial and error.**
- Therefore the **idea of applying learning algorithms to fuzzy was considered early approaches to so call adaptive or self organizing fuzzy controllers can be found in literature.**
- These kind of adaptive models usually use knowledge based methods. However, another possibility of learning parameters of fuzzy systems is given by neural networks.
- The learning capabilities of neural networks made them a prime target for a combination with fuzzy systems in order to automate or support the process of developing a fuzzy system for a given task.
- The neuro-fuzzy approaches were considered mainly in the domain of neuro-fuzzy control, but today the approach is more general. Neuro-fuzzy systems are applied in various domains, e.g., control, data analysis, decision support, etc.

Modern neuro-fuzzy systems

- Modern neuro-fuzzy systems are usually represented as a **multilayer feed forward neural network**, but fuzzification of other neural network architectures are also considered, for example **self-organizing feature maps**.
- In neuro-fuzzy models, connection weights, a propagation and activation function comes from common neural networks.

Learning by Error Back Propagation

The error back propagation algorithm soon became the standard for most neural net implementation due to its high performance.

- First, it selects one of the examples of the training data set.
- Second, it computes the neural net output values for the current training examples' inputs.
- Then, it compares these output values to the desired output value of the training example.
- The difference, called error, determines which neuron in the net shall be modified and how.
- The mathematical mapping of the error back into the neurons of the net is called error back propagation.

Learning by Error Back Propagation

- If the error back propagation algorithm is so powerful, why not use it to train fuzzy logic systems too?
- Alas, this is not straightforward.
- To determine which neuron has what influence, the error back propagation algorithm differentiates the transfer functions of the neurons.
- One problem here is that the standard fuzzy logic inference cannot be differentiated.

Learning by Error Back Propagation

- To solve these problems, some **neuro-fuzzy development tools** use extended fuzzy logic inference methods.
- The most common approach is to use so-called **Fuzzy Associative Memories (**FAMs).
- A FAM is a fuzzy logic rule with an associated weight. A mathematical framework exists that maps FAMs to neurons in a neural net.
- This enables the use of a modified error back propagation algorithm with fuzzy logic.
- As a user of NeuroFuzzy tools, you do not need to worry about the details of the algorithm.
- Today's NeuroFuzzy tools work as an "intelligent" assistant with your design.
- They help you to generate and optimize membership functions and rule bases from sample data.

Different approaches

- Although there are a lot of different approaches, we want to restrict them term neuro-fuzzy to systems which display the following properties:

1. A neuro-fuzzy system is a fuzzy system that is trained by a **learning algorithm usually derived from neural network theory**. The (heuristic) learning procedure operates on local information, and causes only local modifications in the underlying fuzzy system. The learning process is not knowledge based, but data driven.

Different approaches

2. A neuro-fuzzy system can be viewed as a special **3-layer feed forward neural** network.

- The first layer represents input variables, the middle (hidden) layer represents fuzzy rules and the third layer represents output variables.
- Fuzzy sets are encoded as (fuzzy) connection weights. Some neuro-fuzzy models use more than 3 layers, and encode fuzzy sets as activation functions.
- In this case, it is usually possible to transform them into 3 layer architecture.
- This view of fuzzy systems illustrates the data flow within the system and its parallel nature.

Different approaches

3. A neuro-fuzzy system can always (i.e., during and after learning) be interpreted as a **system of fuzzy rules**.
4. The learning procedure of neuro-fuzzy systems takes the similar properties of the underlying fuzzy system in to account. This results in constraints on the possible modifications of the system's parameters.

Different approaches

5. A neuro-fuzzy system approximates an n-dimensional (unknown) function that is partially given by the training data. The fuzzy rules encoded within the system represent vague samples, and can be viewed as vague prototypes of the training data. A neuro-fuzzy system should not be seen as a kind of (fuzzy) expert system, and it has nothing to do with fuzzy logic in the narrow sense.

Definition

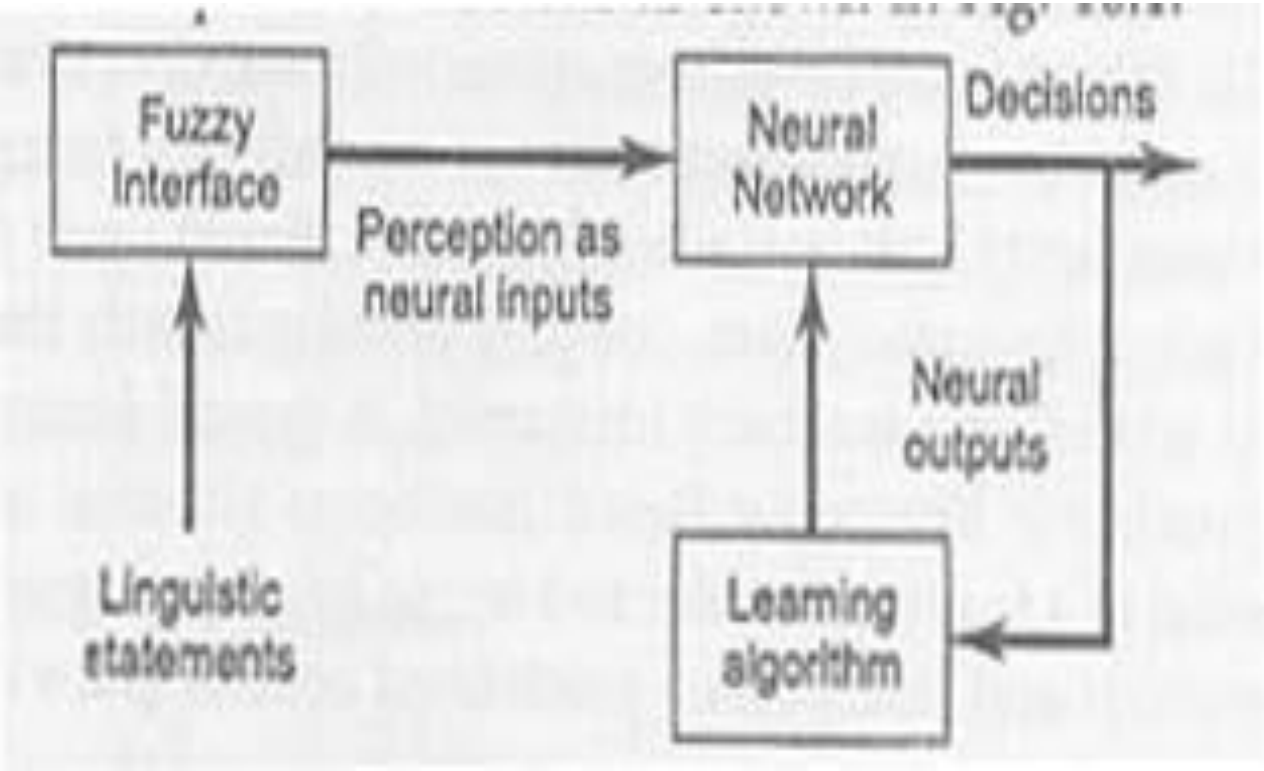
- Based on the above point, we consider **neuro-fuzzy as a technique to derive a fuzzy system from data, or to enhance it by learning from examples.**
- The exact implementation of the neuro-fuzzy model does not matter.
- It is possible to use a neural network to learn certain parameters of a fuzzy system, like using a self-organizing feature map to find fuzzy rules(cooperative models), or to view a fuzzy system as a special neural network and to apply a learning algorithm directly (hybrid models).

Models of neuro-fuzzy systems

Two possible models of neuro-fuzzy systems are:

1. In response to linguistic statements, the fuzzy interface block provides an input vector to a multi-layer neural network. The neural network can be adapted (trained) to yield desired command outputs or decisions as shown in fig. below.

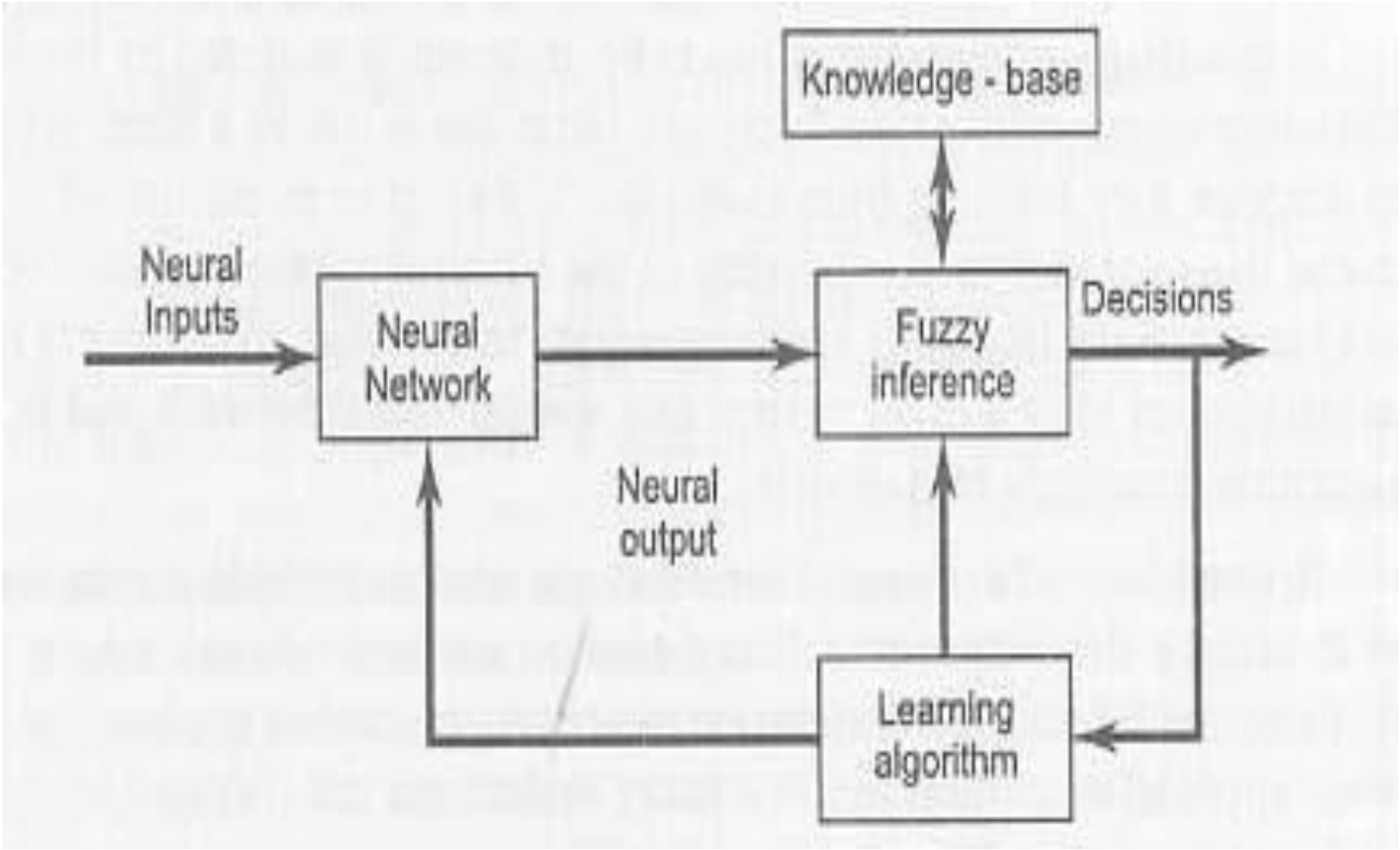
The first model of neuro-fuzzy system



Models of neuro-fuzzy systems

2. A multi-layered neural network drives the fuzzy inference mechanism (Fig.below).
   - Neural networks are used to tune membership functions of fuzzy systems that are employed as decision –making systems for controlling equipment.
   - Although fuzzy logic can encode expert knowledge directly using rules with linguistic labels, it usually takes a lot of time to design and tune the membership functions which quantitatively define these linguistic labels.
   - Neural network learning techniques can automate this process and substantially reduce development time and cost while improving performance.

The second model of neuro-fuzzy system



Conclusion

- In theory, neural networks, and fuzzy systems are equivalent in that they are convertible, yet in practice each has its own advantages and disadvantages.
- For neural networks, **the knowledge is automatically acquired by the back-propagation algorithm, but the learning process is relatively slow and analysis of the trained network is difficult.**
- Neither is it possible to extract structural knowledge from the trained neural network, nor can we integrate special information about the problem into the neural network in order to simplify the learning procedure.

Conclusion (contd.)

- Fuzzy systems are more favorable in that their behavior can be explained based on fuzzy rules and thus their performance can be adjusted by tuning the rules.
- But since, in general, knowledge acquisition is difficult and also the universe of discourse of each input variable needs to be divided in to several intervals, applications of fuzzy systems are restricted to the fields where expert knowledge is available and the number of input variable is small.

Genetic algorithm

Genetics

- Genetics is a science which deals with the transfer of biological information from generation to generation.
- Genetics deals with the physical and chemical nature of these information's itself.
- Geneticists are concerned with whys and how's of these transfer of biological information, which is the basis for certain differences and similarities that are recognized in a group of living organisms.
- What is the source of genetic variations?
- How are difference distributed in populations?
- Why not all variations among living things however are inherited?
- All these are concern with genetics.

Genetics

- The mechanism of genetics is entirely based on gene.
- The gene concept however, had been implicit in model's visualization of a physical element or factor that acts as the foundation of development of a trait.
- He first postulated the existence of genes from their end effects, as expressed in altered characteristics.
- The word "allelmorph", shortened to "allele" is used to identify the member of paired genes that control different alternative traits.
- The gene is characterized as an individual unit of genetic mechanism. **Genes replicate themselves and reproduce chromosomes, cells and organisms of their own kind.**
- Gene is the part of chromosome. Some chromosomal genes work together, each making a small contribution to height, weight or intelligence, etc.
- Genes not only have a basic role in the origin and life of individual organisms, but they also, through variation in gene frequencies, cause change in populations.

What is GA

- A **genetic algorithm** (or **GA**) is a search technique used in computing to find true or approximate solutions to optimization and search problems.
- Genetic algorithms are categorized as global search heuristics.
- Genetic algorithms are a particular class of evolutionary algorithms that use techniques inspired by evolutionary biology such as inheritance, mutation, selection, and crossover (also called recombination).

What is GA

- The evolution usually starts from a population of randomly generated individuals and happens in generations.

- In each generation, the fitness of every individual in the population is evaluated, multiple individuals are selected from the current population (based on their fitness), and modified (recombined and possibly mutated) to form a new population.

What is GA

- The new population is then used in the next iteration of the algorithm.
- Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population.
- If the algorithm has terminated due to a maximum number of generations, a satisfactory solution may or may not have been reached.

Evolution

- Here's a very oversimplified description of how evolution works in biology
- Organisms (animals or plants) produce a number of offspring which are almost, but not entirely, like themselves
  - Variation may be due to mutation (random changes)
  - Variation may be due to sexual reproduction (offspring have some characteristics from each parent)
- Some of these offspring may survive to produce offspring of their own—some won't
  - The "better adapted" offspring are more likely to survive
  - Over time, later generations become better and better adapted
- **Genetic algorithms use this same process to "evolve" better programs**

Genetic Algorithms

- Proposed by John Holland (1975)
- A Genetic Algorithm (GA) is a search technique used in computing to find exact or approximate solutions to optimization and search problems.
- GA uses techniques inspired by evolutionary biology such as inheritance, mutation, selection, and crossover.

Genetic Algorithms

- Genetic algorithms are computerized search and optimization algorithms based on the mechanics of natural genetics and natural selection
  - Derived by biologists
  - The offspring have certain desirable characteristics

Mathematical terminology	**Equivalent biological terminology**
Parameter	Gene
Set of parameters	Chromosome
Objective function evaluation	Fitness value
Population of individuals	Set of solutions

Genetic Algorithms

- The beginnings of genetic algorithms can be traced back to the early 1950s when several biologists used computers for simulations of biological systems (Goldberg and Holland 1989).
- However, the work done in late 1960s and early 1970s at the University of Michigan under the guidance of John Holland led to genetic algorithms as they are known today.
- GA vocabulary is being borrowed from natural genetics.
- **The idea behind genetic algorithms is to do what nature does.**
- Genetic algorithms (GAs) are stochastic algorithms whose search methods inspired from phenomena found in living nature.

Genetic Algorithms

- The phenomena incorporated so far in GA models include phenomena of natural selection as there are selection and the production of variation by means of recombination and mutation, and rarely inversion, diploid and others.
- Most Genetic algorithms work with one large panmictic population, i.e. in the recombination step each individual may potentially choose any other individual from the population as a mate.

GA operators

- Then GA operators are performed to obtain the new child offspring; the operators are:

i. Encoding
ii. Selection
iii. Crossover,
iv. Mutation,
v. Survival of fittest

Encoding

The process of representing the solution in the form of a **string** that conveys the necessary information.

- Just as in a chromosome, each gene controls a particular characteristic of the individual, similarly, each bit in the string represents a characteristic of the solution.

Encoding Methods

- **Binary Encoding –** Most common method of encoding. Chromosomes are strings of 1s and 0s and each position in the chromosome represents a particular characteristic of the problem.

Chromosome A	10110010110011100101
Chromosome B	11111110000000011111

Encoding Methods (contd.)

- **Permutation Encoding –** Useful in ordering problems such as the Traveling Salesman Problem (TSP). Example. In TSP, every chromosome is a string of numbers, each of which represents a city to be visited.

Chromosome A	1 5 3 2 6 4 7 9 8
Chromosome B	8 5 6 7 2 3 1 4 9

Encoding Methods (contd.)

- **Value Encoding –** Used in problems where complicated values, such as real numbers, are used and where binary encoding would not suffice.

  Good for some problems, but *often necessary to develop some specific crossover and mutation techniques for these chromosomes.*

Chromosome A	1.235 5.323 0.454 2.321 2.454
Chromosome B	(left), (back), (left), (right), (forward)

Selection

- As in natural surroundings it holds on average: **“the better the parents, the better the offsprings” and “the offspring is similar to the parents”.**
- Therefore, it is on the one hand desirable to choose the fittest individuals more often, but on the other hand not too often, because otherwise the diversity of the search space decreases.
- GA researchers have developed a variety of selection algorithms to make it more efficient and robust.

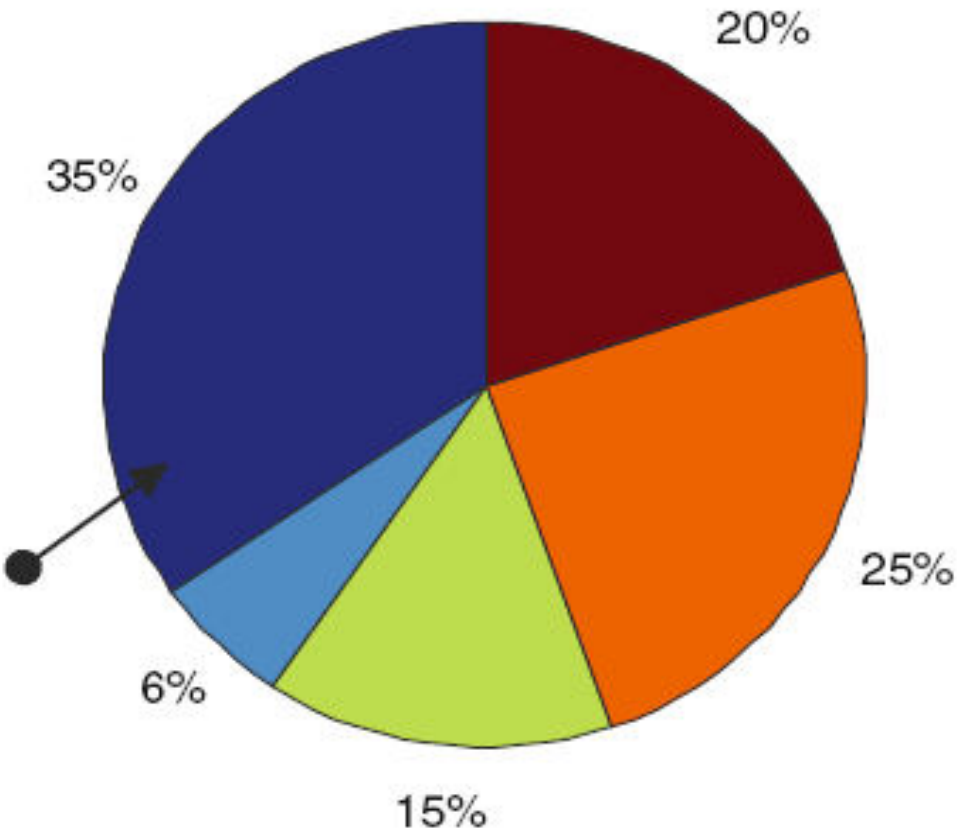
Selection algorithm

- **Roulette Wheel Selection**
- **Tournament Selection**
- **Linear Rank Selection**

Roulette Wheel Selection

- In roulette selection chromosomes are selected based on their fitness relative to all other chromosomes in the population.
- One disadvantage of using roulette wheel is that its selective pressure reduces as population converges upon a solution, which reduces the convergence rate and may not allow finding the better solutions.

Roulette Wheel Selection



Tournament Selection

- In this process of selection, one parent is selected by randomly comparing other individual in the same population and select with the best fitness.
- To select the second parent the same process is repeated.
- It is most popular selection method due to its simplicity.

Linear Rank Selection

- In this method the individuals are ordered according to their fitness values.
- The individuals of highest fitness are kept on the top and worst on the bottom.
- Then each individual in the population is assigned a subjective fitness based on linear ranking function as

f(r)=(popsize-rank)(max-min)/(Popsize-1)+min

where popsize – population size

rank – rank in the current population

max, min – maximum and minimum subjective fitness determined by the user.

- Now this subjective fitness value is assigned to the individual and the selection is done on the basis of roulette wheel spinning.
- In this selection process the selective pressure is constant and does not change with generation to generation.
- However, in this process, it is necessary to sort the population according to their fitness values and the individuals of same fitness will not have the same probability of being selected.

Crossover

- Obviously, selection alone can not generate better offsprings.
- To produce better new off springs a crossover operator is required. A crossover operator can be termed loosely as recombination or slice-exchange-merge operator.
- The most common type of crossover operator mentioned above is called single point crossover.
- In this operation select two parents and randomly selects a point between two genes to cut both chromosomes into two parts.
- This point is called crossover point.
- In crossover operation combine the first part of first parent and second part of second parent to get first offspring.
- Similarly, combine the first part of second parent and second part of first parent to get second offspring.
- These offsprings belong to the next population.

Crossover

The crossover operator has three distinct sub steps:

a) Slice each of the parent strings in two substrings.

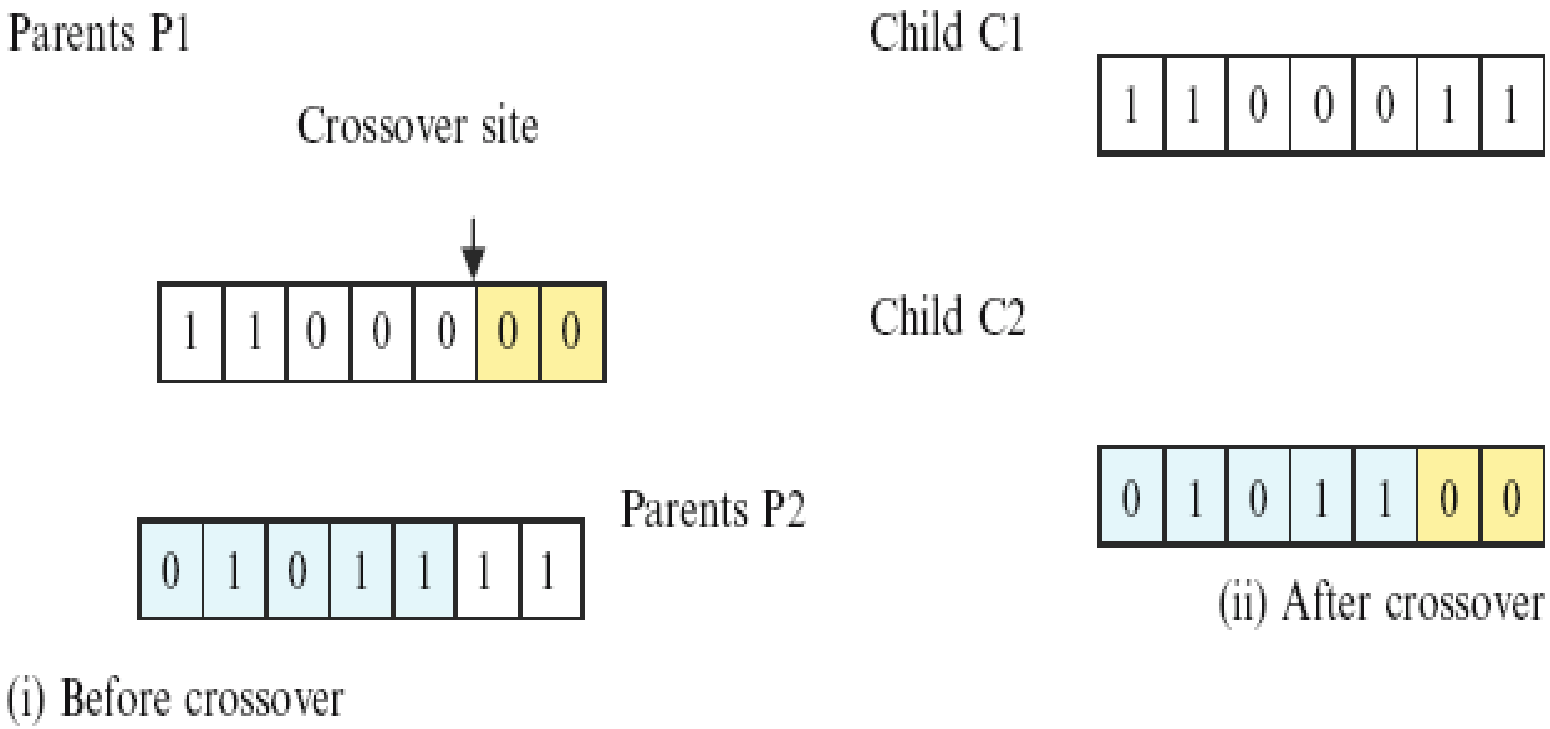
b) Exchange a pair of corresponding substrings of parents.

c) Merge the two respective substrings to form offsprings.

Example

- For example, suppose following two binary strings are mated together and undergoes the crossover operation.
- The strings are 1100000, 0101111.
- By a random choice the crossover site is fixed at 3 which is shown by a vertical bar.
- Then the effect of cross over will be as shown in below figure.

Single point cross over operation with two strings



Example

- To increase the speed of convergence of GA, the population is divided from the middle and two halves (subgroups) are used in group cross over as shown in Fig. below.

Single point Group Cross over operation

Group of Parents P1

1	1	1	0	1
0	1	0	0	0
0	0	1	1	0
1	1	1	0	0

Group of Parents P2

1	0	1	1	1
1	1	0	0	1
0	1	1	0	0
1	0	1	0	1

(i) Before Crossover

Group of Children C1

1	1	1	1	1
0	1	0	0	1
0	0	1	0	0
1	1	1	0	1

Group of Children C2

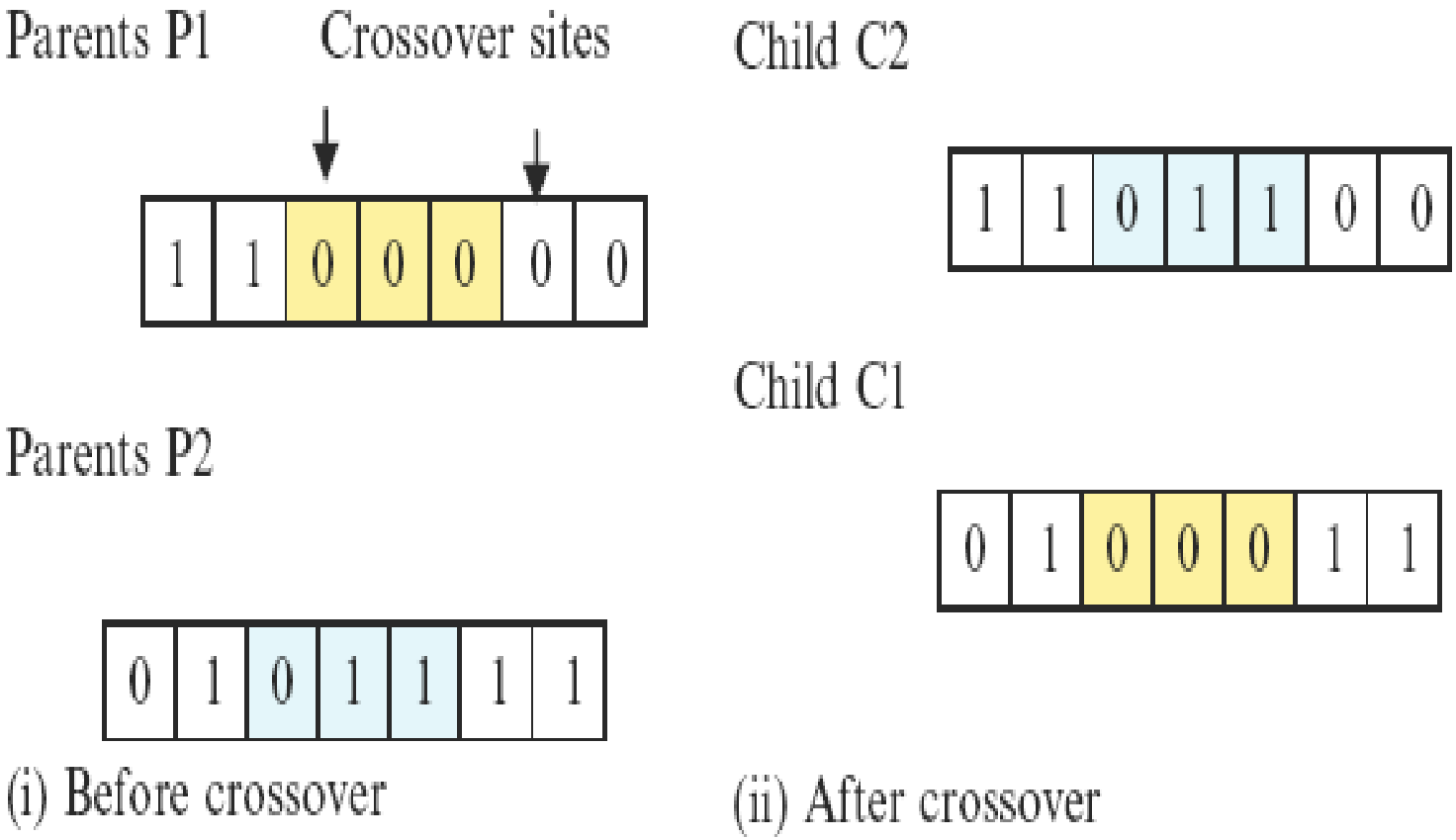
1	0	1	0	1
1	1	0	0	0
0	1	1	1	0
1	0	1	0	0

(ii) After Crossover

Example

- Another type of crossover is multi-point cross over, in which two or more than two sites have been selected and exchange have been done as illustrated in Fig. below.

Multipoint crossover



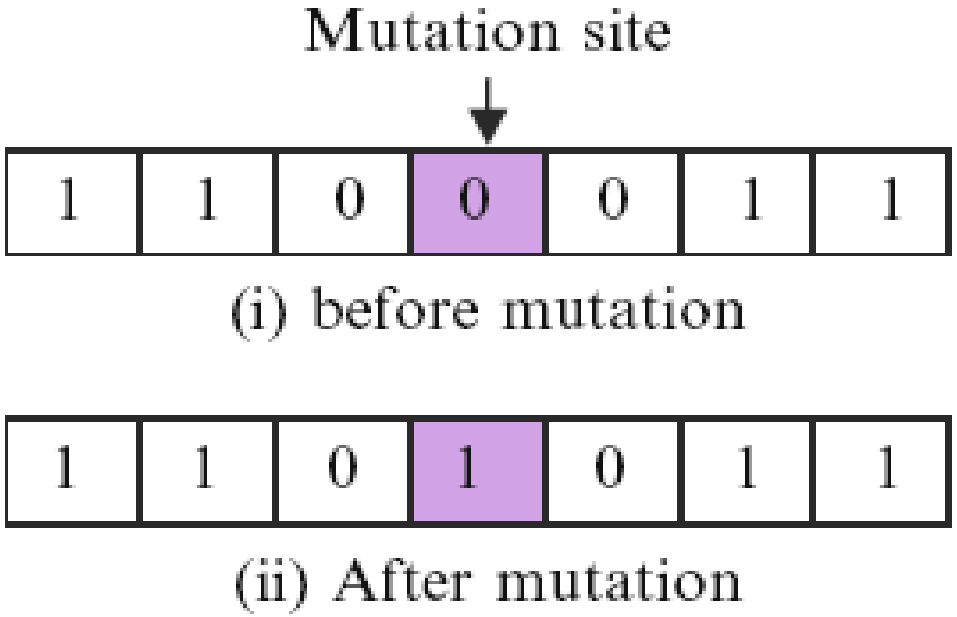
Mutation

- The newly created individuals have no new inheritance information and the number of alleles is constantly decreasing.
- This process results in the contraction of the population to one point, which is only wished at the end of the convergence process, after the population works in a very promising part of the search space.
- Diversity is necessary to search a big part of the search space.
- It is one goal of the learning algorithm to search always in regions not viewed before.
- Therefore, it is necessary to enlarge the information contained in the population.

Mutation

- One way to achieve this goal is mutation.
- The mutation operator M (chromosome) **selects a gene of that chromosome** and changes the allele by an amount called the mutation variance (mv), this happens with a mutation frequency (mf).
- The parameter mutation variance and mutation frequency have a major influence on the quality of learning algorithms.
- For binary coded GAs mutation is equivalent of flipping a bit at any particular position.
- Since, mutation is to be used sparingly its probability is very low.
- The mutation operation may be shown as in Fig.below. The group mutation and multipoint mutation may also be performed to improve the results.

Single point mutation operation



Survival of Fittest

- Further more we only accept an offspring as a new member of population, if it differ enough from the other individuals, that means here its fitness differ from all other individuals at least by some significant amount.
- After accepting a new individual we remove one of the worst individual (i.e. its fitness value is quite low) from the population in order to hold the population size constant.

Advantages Of GAs

- **Global Search Methods:** GAs search for the function optimum starting from a *population of points* of the function domain, not a single one. This characteristic suggests that GAs are global search methods. They can, in fact, climb many peaks in parallel, reducing the probability of finding local minima, which is one of the drawbacks of traditional optimization methods.

Advantages of GAs (contd.)

- **Blind Search Methods:** GAs only use the information about the *objective function*. They do not require knowledge of the first derivative or any other auxiliary information, allowing a number of problems to be solved without the need to formulate restrictive assumptions. For this reason, GAs are often called blind search methods.

Advantages of GAs (contd.)

- **GAs use probabilistic transition rules** during iterations, unlike the traditional methods that use fixed transition rules.

This makes them more robust and applicable to a large range of problems.

Advantages of GAs (contd.)

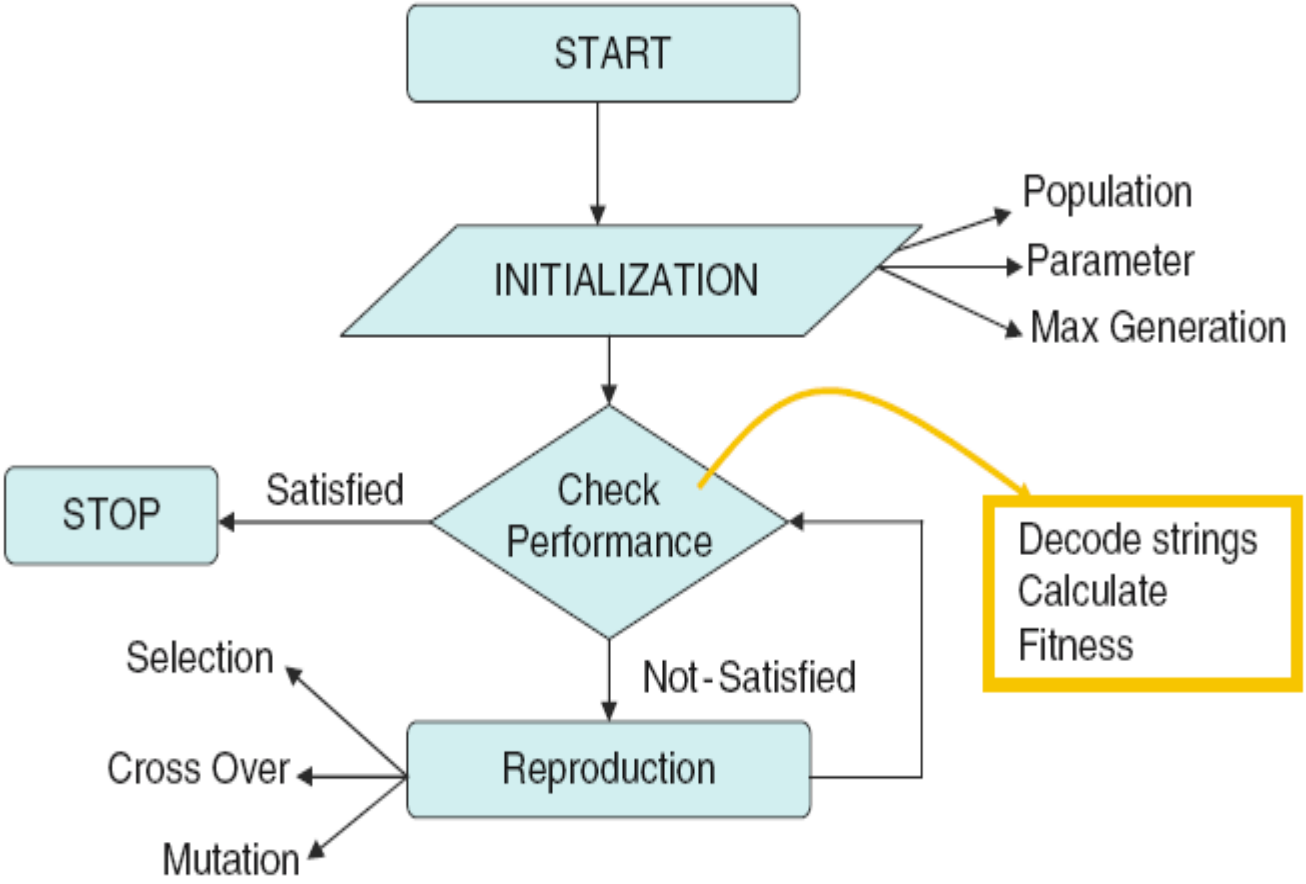
- **GAs can be easily used in *parallel machines-*** Since in real-world design optimization problems, most computational time is spent in evaluating a solution, with multiple processors all solutions in a population can be evaluated in a distributed manner. This reduces the overall computational time substantially.

Main Components of GA

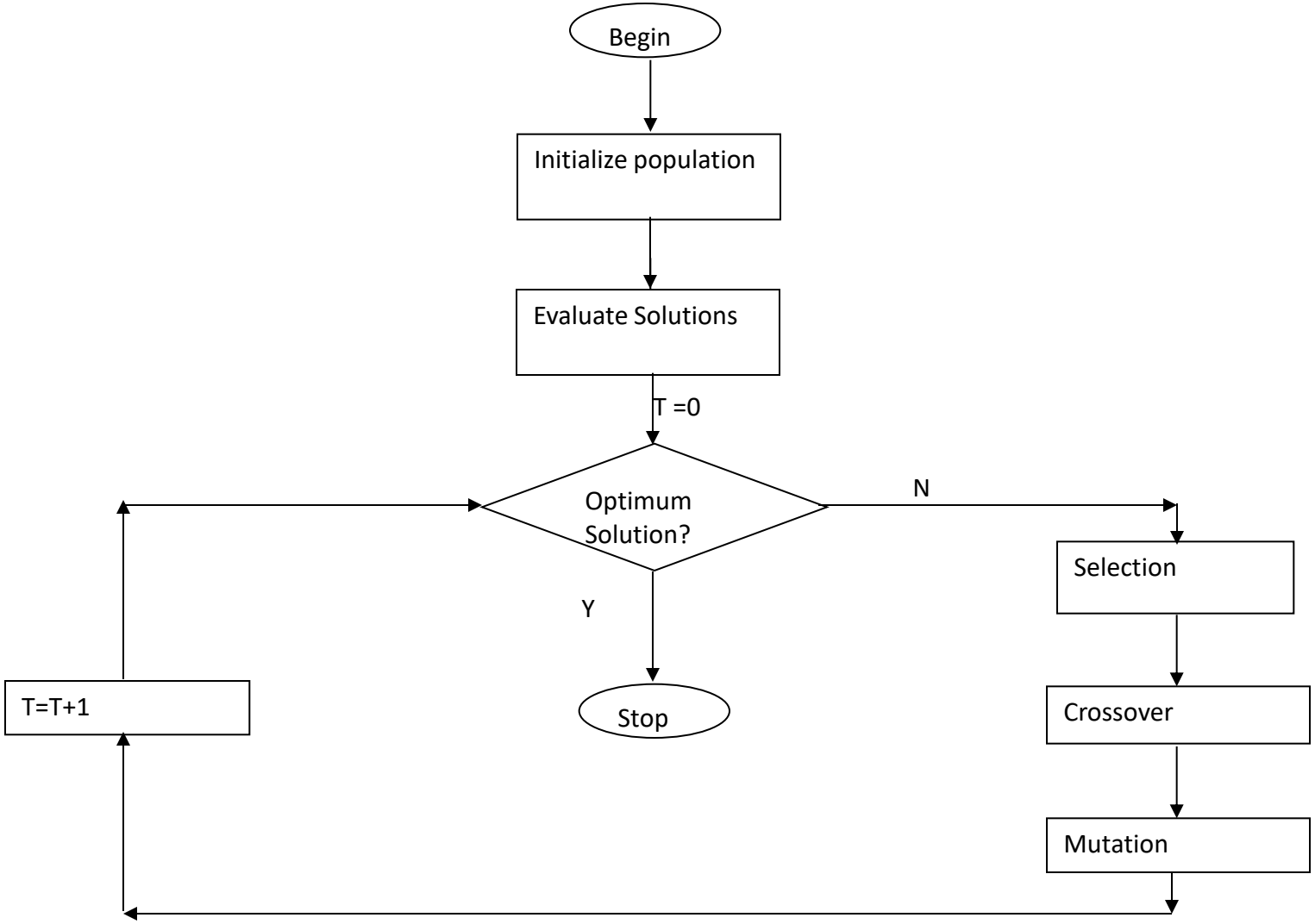
A GA for a particular problem must have the following five components:

1. A genetic representation for potential solutions to the problem (Coding).
2. A way to create an initial population of potential solution.
3. To evaluate rank of a solution define an objective function.
4. To alter the composition of offspring's define genetic operators.
5. Define GA parameters like population size, probabilities of genetic operators, etc.

Flow chart of simple genetic algorithm



Working Mechanism Of GAs



Applications of Genetic Algorithms

GA is not only used for solving optimization problems, but there are number of GA applications as mentioned below:

1. Industrial design by parameterization
2. Scheduling problems such as manufacturing, facility scheduling, allocation of resources, etc.
3. System design
4. Time series prediction
5. Data base mining
6. Control system
7. Artificial life system
8. Various medical applications, such as image segmentation and modeling
9. Combinatorial optimization problems like travelling sales man problem, routing, bin packing, graph partitioning and colouring.
10. Trajectory planning of robots
11. Game playing like chase playing, prisoner's dilemma, etc.
12. Resource allocation problem
13. Graph colouring and partitioning, etc.

Genetic Algorithms To Solve The Traveling Salesman Problem (TSP)

The Problem

The **Traveling Salesman Problem** is defined as:

‘We are given a set of cities and a symmetric distance matrix that indicates the cost of travel from each city to every other city.

The goal is to find **the shortest circular tour,** visiting every city exactly once, so as to **minimize the total travel cost**, which includes the cost of traveling from the last city back to the first city’.

Encoding

- I represent every city with an integer .

- Consider 6 Indian cities –

  Mumbai, Nagpur , Calcutta, Delhi , Bangalore and Chennai and assign a number to each.

Mumbai → 1
Nagpur → 2
Calcutta → 3
Delhi → 4
Bangalore → 5
Chennai → 6

Encoding (contd.)

- Thus a path would be represented as a sequence of integers from 1 to 6.

- The path [1 2 3 4 5 6] represents a path from Mumbai to Nagpur, Nagpur to Calcutta, Calcutta to Delhi, Delhi to Bangalore, Bangalore to Chennai, and finally from Chennai to Mumbai.

- This is an example of **Permutation Encoding** as the position of the elements determines the fitness of the solution.

Fitness Function

- The fitness function will be the total cost of the tour represented by each chromosome.

- This can be calculated as the sum of the distances traversed in each travel segment.

The Lesser The Sum, The Fitter The Solution Represented By That Chromosome.

Distance/Cost Matrix For TSP

	1	2	3	4	5	6
1	0	863	1987	1407	998	1369
2	863	0	1124	1012	1049	1083
3	1987	1124	0	1461	1881	1676
4	1407	1012	1461	0	2061	2095
5	998	1049	1881	2061	0	331
6	1369	1083	1676	2095	331	0

Cost matrix for six city example.

Distances in Kilometers

Fitness Function (contd.)

- So, for a chromosome [4 1 3 2 5 6], the total cost of travel or fitness will be calculated as shown below

- Fitness = 1407 + 1987 + 1124 + 1049 + 331+ 2095
  = 7993 kms.

- Since our objective is to Minimize the distance, the lesser the total distance, the fitter the solution.

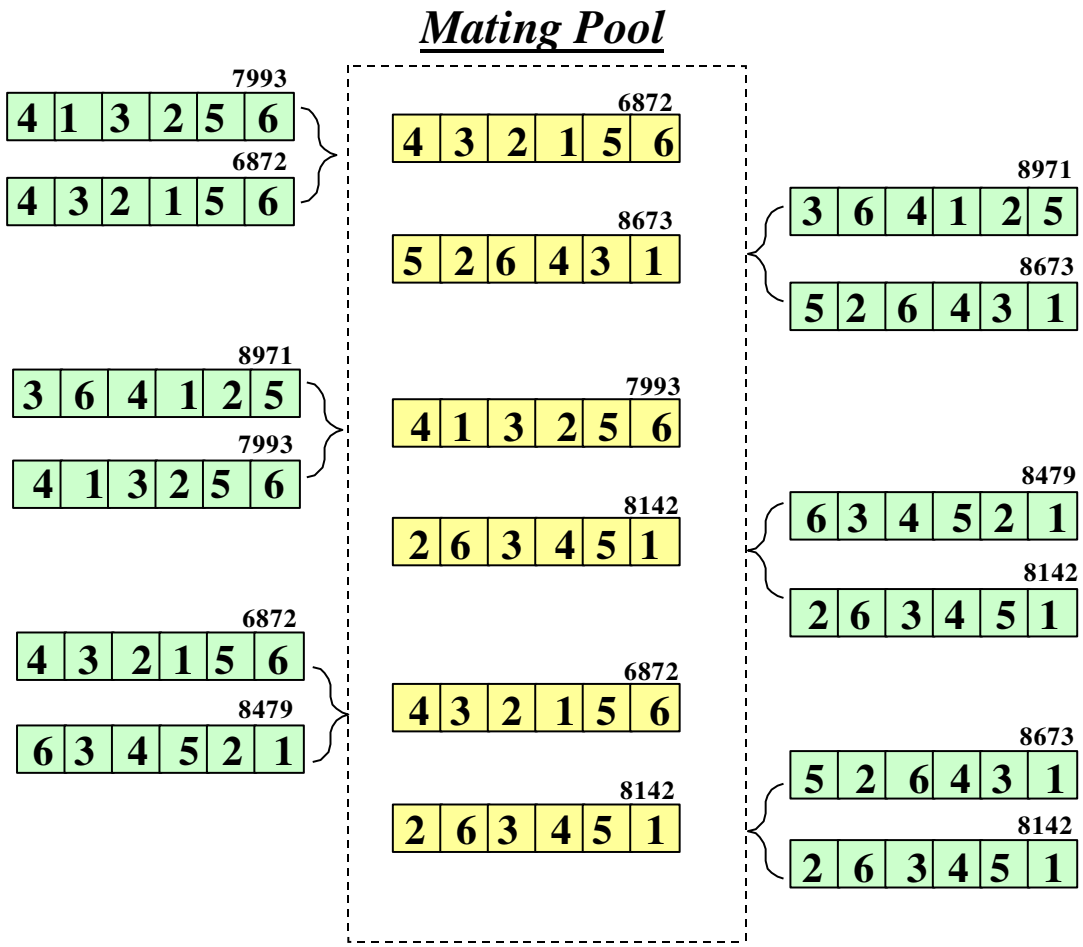
Selection Operator

I used ***Tournament Selection***.

As the name suggests *tournaments* are played between two solutions and the better solution is chosen and placed in the *mating pool*.

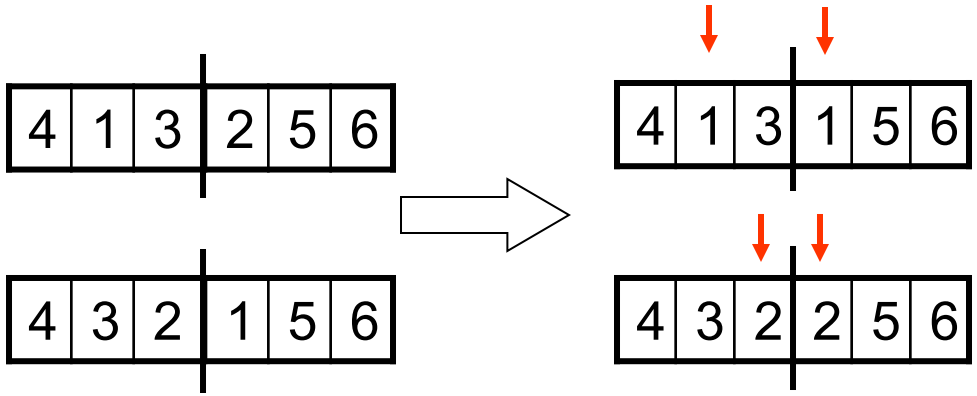
Two other solutions are picked again and another slot in the *mating pool* is filled up with the better solution.

Tournament Selection (contd.)



Why we cannot use single-point crossover:

- Single point crossover method randomly selects a crossover point in the string and swaps the substrings.
- This may produce some invalid offsprings as shown below.



Crossover Operator

- I used the *Enhanced Edge Recombination* operator (T.Starkweather, et al, *'A Comparison of Genetic Sequencing Operators,* International Conference of GAs, 1991) .

- This operator is different from other genetic sequencing operators in that it emphasizes *adjacency information* instead of the order or position of items in the sequence.

- The algorithm for the Edge-Recombination Operator involves constructing an Edge Table first.

Edge Table

The *Edge Table* is an *adjacency table* that lists links *into* and *out of* a city found in the two parent sequences.

If an item is already in the edge table and we are trying to insert it again, that element of a sequence must be a *common edge* and is represented by inverting it's sign.

Finding The Edge Table

Parent 1	**4**	**1**	**3**	**2**	**5**	**6**
Parent 2	**4**	**3**	**2**	**1**	**5**	**6**

1	4	3	2	5
2	-3	5	1	
3	1	-2	4	
4	-6	1	3	
5	1	2	-6	
6	-5	-4		

Enhanced Edge Recombination Algorithm

1. Choose the initial city from one of the two parent tours. (It can be chosen randomly as according to criteria outlined in *step 4*). This is the *current city.*

2. Remove all occurrences of the *current city* from the left hand side of the edge table.(These can be found by referring to the edge-list for the *current city*).

3. If the *current city* has entries in it's edge-list, go to s*tep 4* otherwise go to *step 5*.

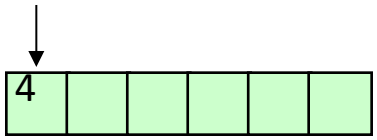
4. Determine which of the cities in the edge-list of the *current city* has the fewest entries in it's own edge-list. The city with fewest entries becomes the *current city*. In case a negative integer is present, it is given preference. Ties are broken randomly. Go to *step 2.*

5. If there are no remaining *unvisited* cities*,* then *stop.* Otherwise, randomly choose an *unvisited* city and go to *step 2.*

Example Of Enhanced Edge Recombination Operator

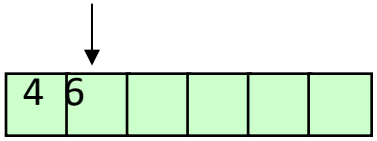
Step 1

1	4	3	2	5
2	-3	5	1	
3	1	-2	4	
4	-6	1	3	
5	3	2	-6	
6	-5	-4		



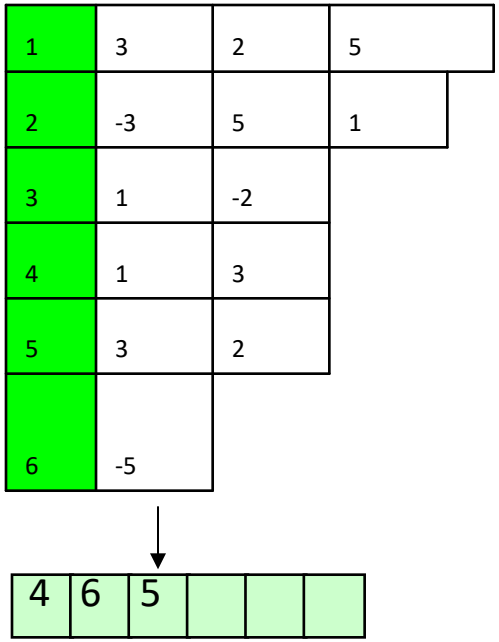
Step 2

1	3	2	5
2	-3	5	1
3	1	-2	
4	-6	1	3
5	3	2	-6
6	-5		

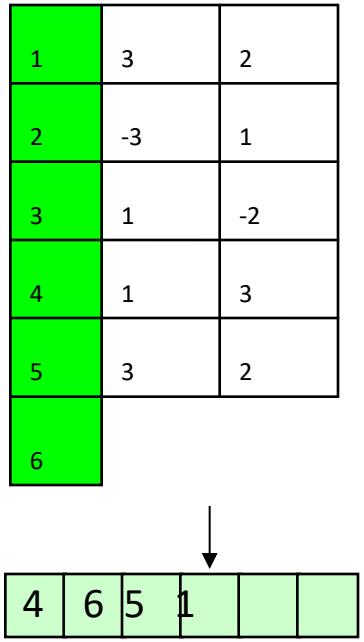


Example Of Enhanced Edge Recombination Operator (contd.)

Step 3

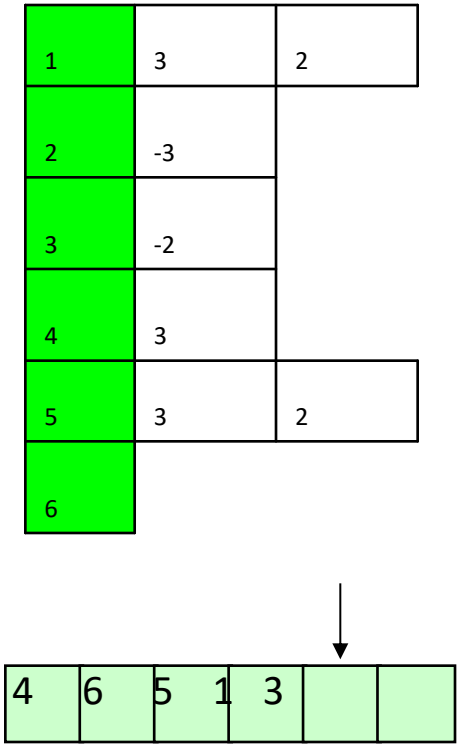


Step 4

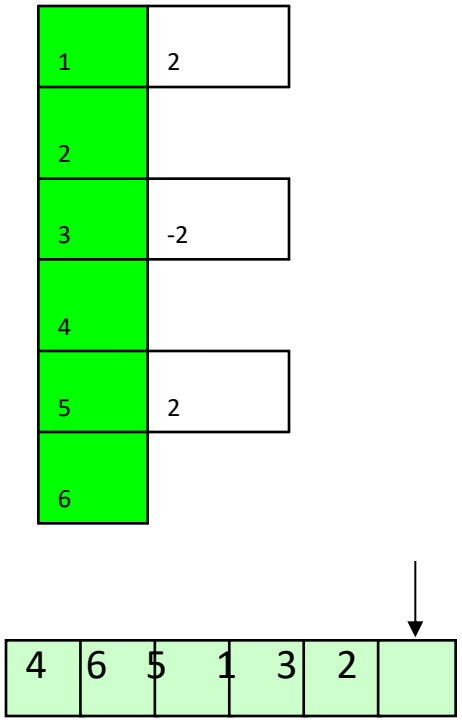


Example Of Enhanced Edge Recombination Operator (contd.)

Step 5

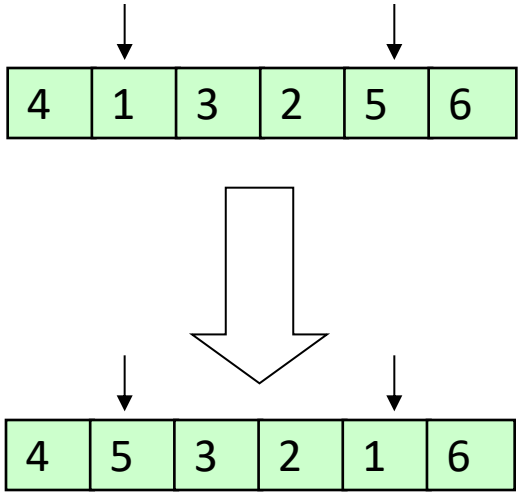


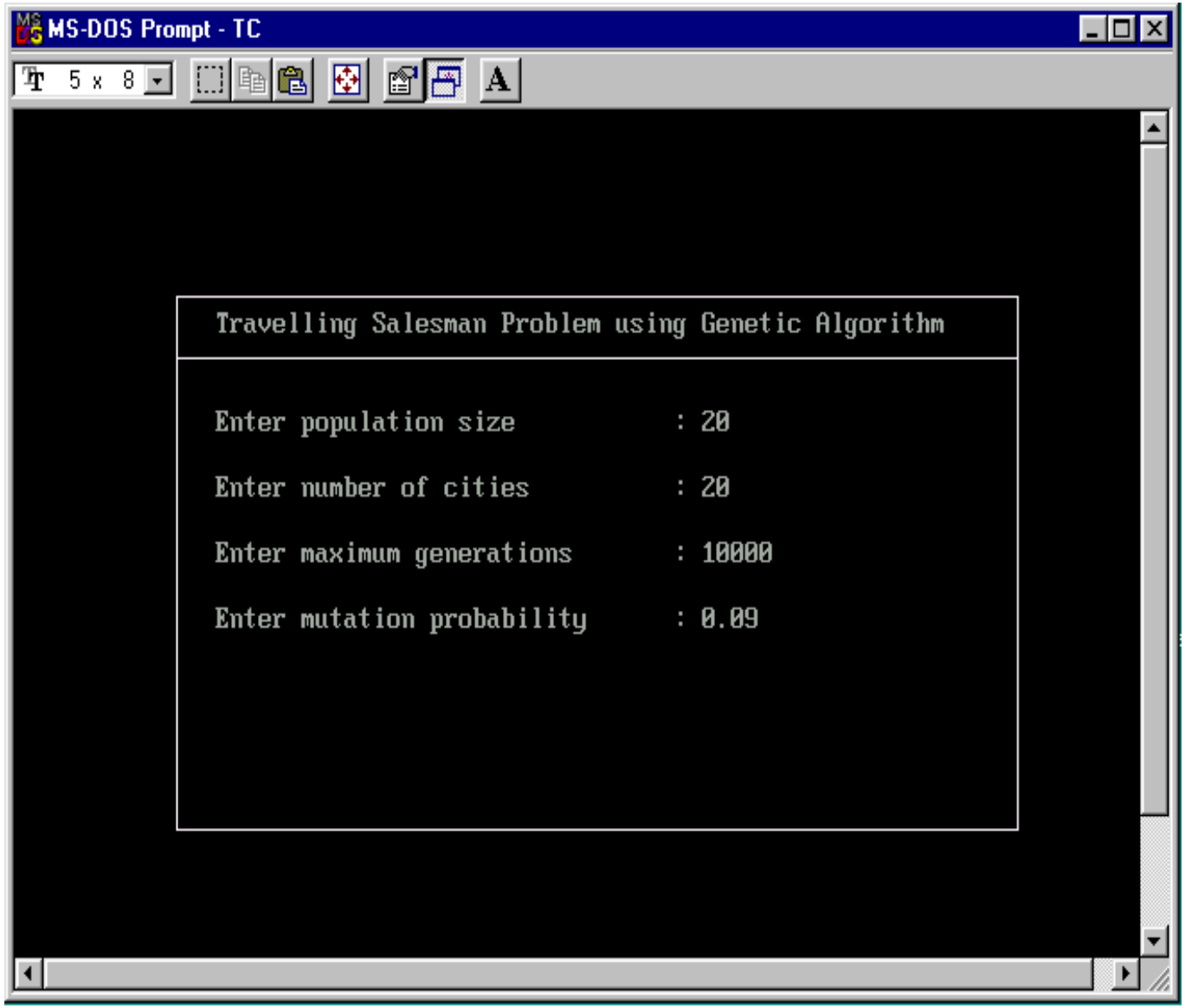
Step 6



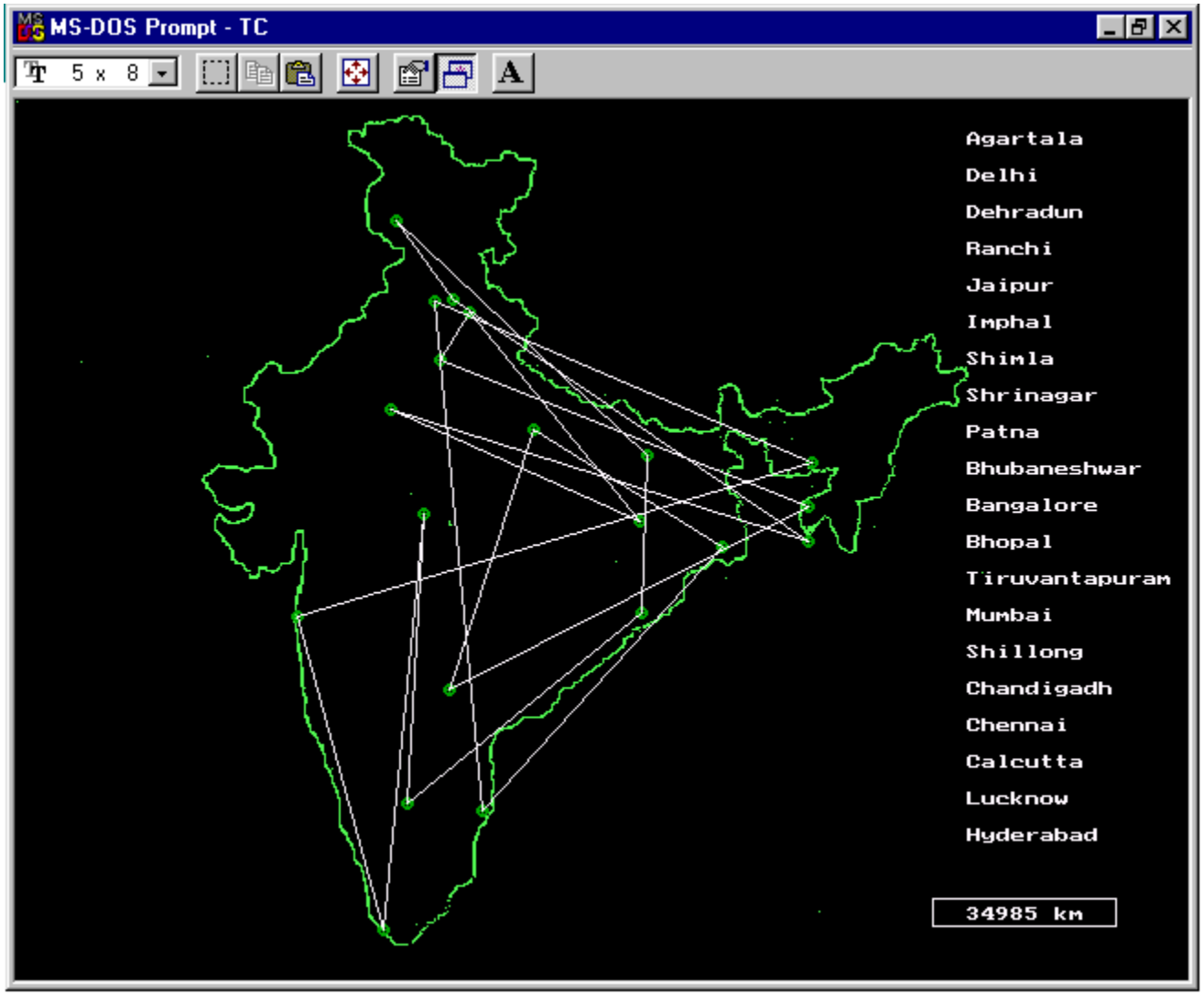
Mutation Operator

- The mutation operator induces a change in the solution, so as to maintain diversity in the population and prevent *Premature Convergence*.
- In our project, we mutate the string by randomly selecting any two cities and interchanging their positions in the solution, thus giving rise to a new tour.



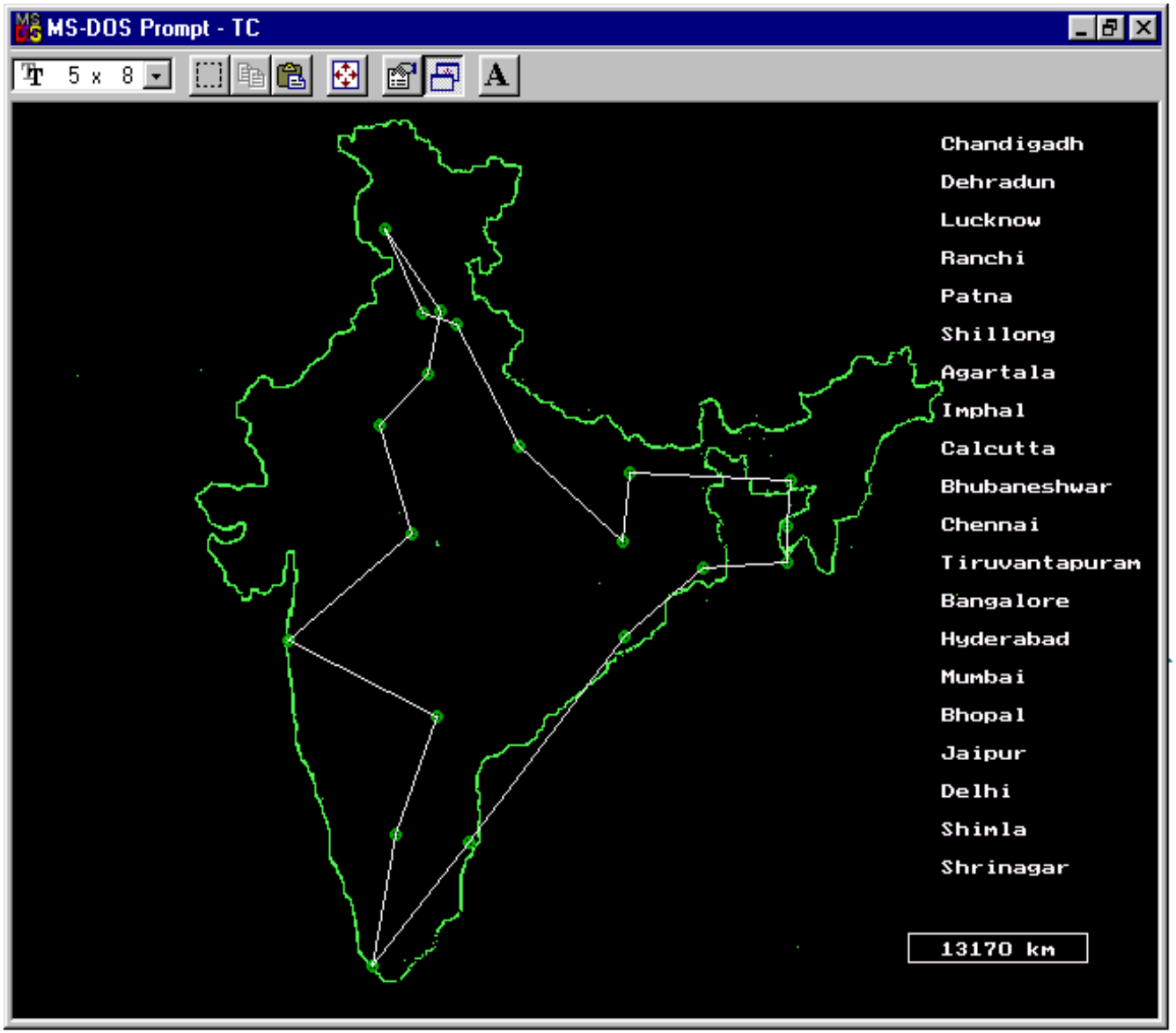


Input To Program



Initial Output For 20 cities : Distance=34985 km

Initial Population



Final Output For 20 cities : Distance=13170 km
Generation 4786

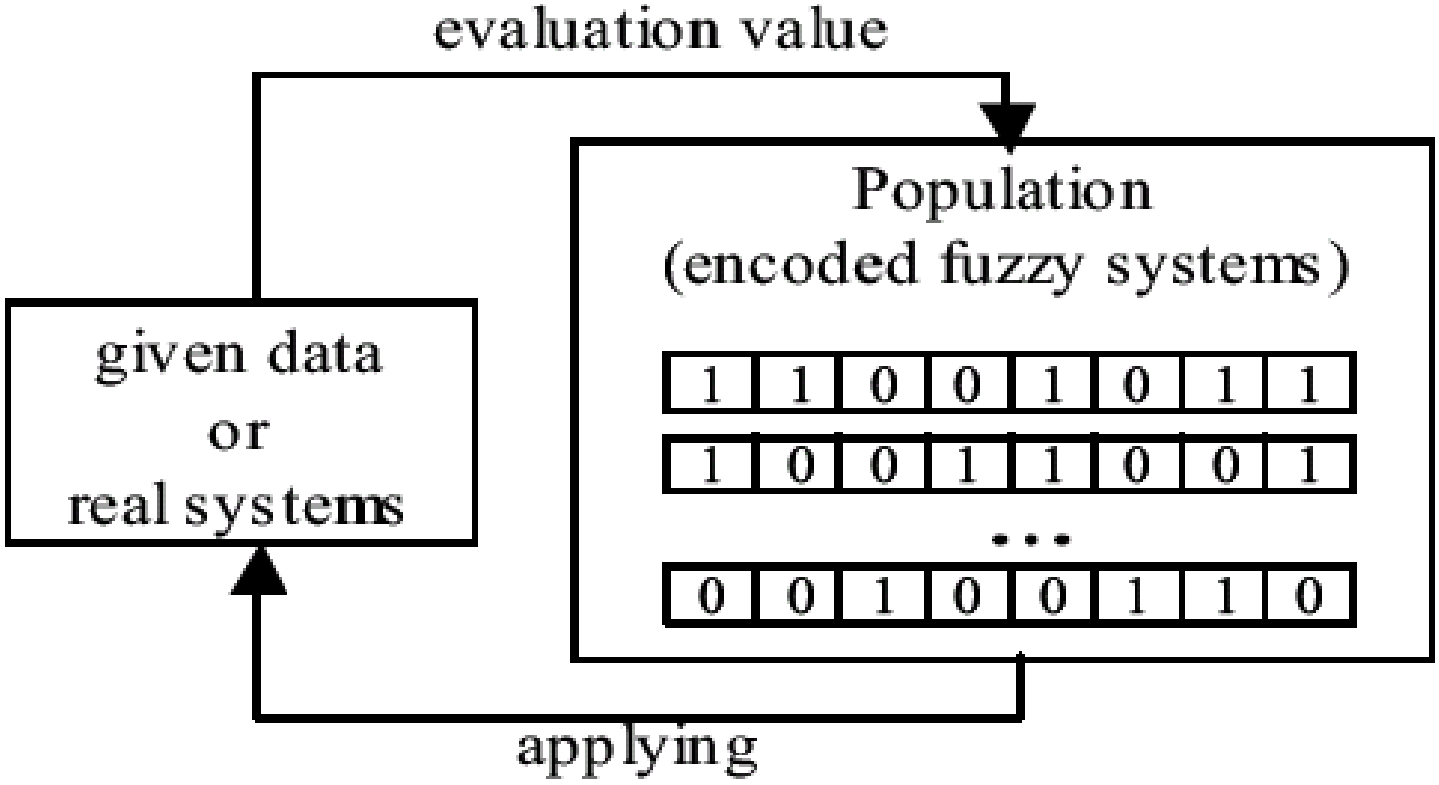
Fuzzy-genetic algorithm

Fusion with Genetic Algorithms

- Identifying fuzzy systems with genetic algorithms
- Controlling parameters of genetic algorithms with fuzzy systems

Identifying fuzzy systems with genetic algorithms

- Schematic diagram of identifying FSs with GAs



Identifying fuzzy systems with genetic algorithms

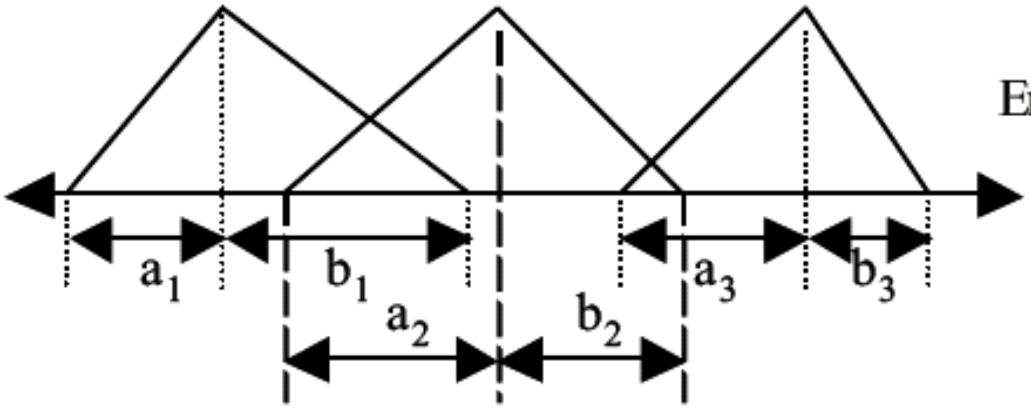
- Tuning an existing fuzzy system
- Building a fuzzy system with genetic algorithm

Tuning an existing fuzzy system

- Four fuzzy rules:

IF X is I_1 THEN Y is O_1
IF X is I_2 THEN Y is O_2
IF X is I_3 THEN Y is O_3
IF X is I_4 THEN Y is O_4

Encoding ⇨ $O_1 O_2 O_3 O_4$



Encoding ⇨ | a_1 | b_1 | a_2 | b_2 | a_3 | b_3 |

Building a fuzzy system with genetic algorithm

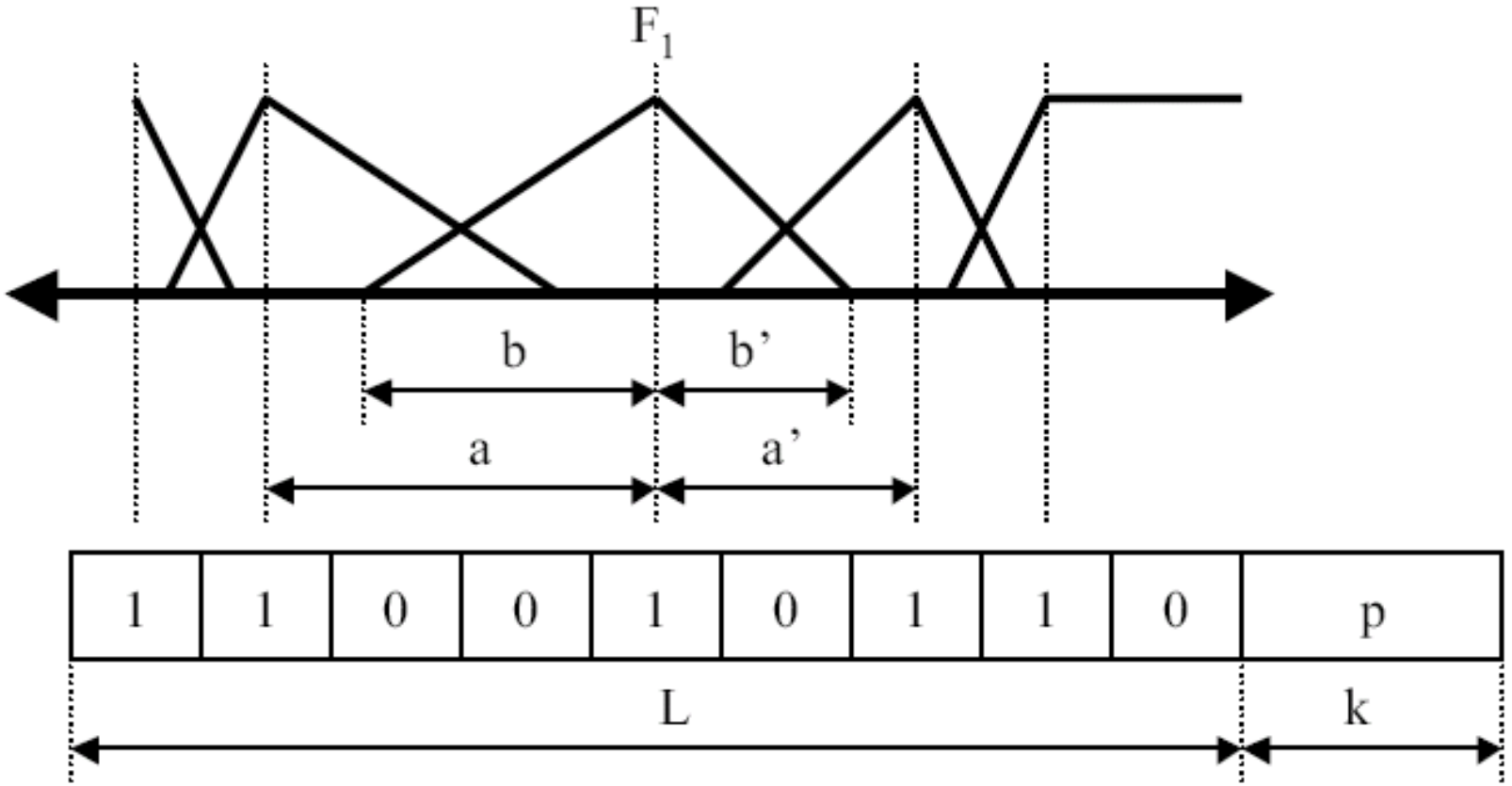
- This method do not need an existing fuzzy system. This approach determines all the parameters of a fuzzy system by genetic algorithms without any priori knowledge.
- Thus, the chromosomes used in this method usually include most of the parameters such as the number and membership functions of linguistic terms.
- So, it is very important how to effectively represent those parameters because a long chromosome means a wide search space.

Building a fuzzy system with genetic algorithm

- If a search space is wide, we cannot expect a good optimization result.
- So, most researches make restrictions; for example, some fix the number of linguistic terms or restrict the shape and position of membership functions.

Building a fuzzy system with genetic algorithm

$$n \times (L + k)$$



Building a fuzzy system with genetic algorithm

- Determination of consequent parts

$$R_1 : \text{IF } x_1 \text{ is } A_{11} \text{ and } x_2 \text{ is } A_{12} \text{ THEN } y = C_1$$
$$R_2 : \text{IF } x_1 \text{ is } A_{21} \text{ and } x_2 \text{ is } A_{22} \text{ THEN } y = C_2$$
$$\vdots$$
$$R_{n_r} : \text{IF } x_1 \text{ is } A_{n_r 1} \text{ and } x_2 \text{ is } A_{n_r 2} \text{ THEN } y = C_{n_r}$$

$$(x_{11}, x_{12}, y_1)(x_{21}, x_{22}, y_2), \cdots, (x_{n_s 1}, x_{n_s 2}, y_{n_s})$$

$$y_i' = \frac{\sum_{k=1}^{n_r} \mu_k(x_{i1}, x_{i2}) \cdot C_k}{\sum_{k=1}^{n_r} \mu_k(x_{i1}, x_{i2})}$$

Building a fuzzy system with genetic algorithm

$$y_i' = a_{1i}C_1 + a_{2i}C_2 + \cdots + a_{n_r i}C_{n_r}$$

$$a_{ji} = \frac{\mu_j(x_{i1}, x_{i2})}{\sum_{k=1}^{n_r} \mu_k(x_{i1}, x_{i2})}$$

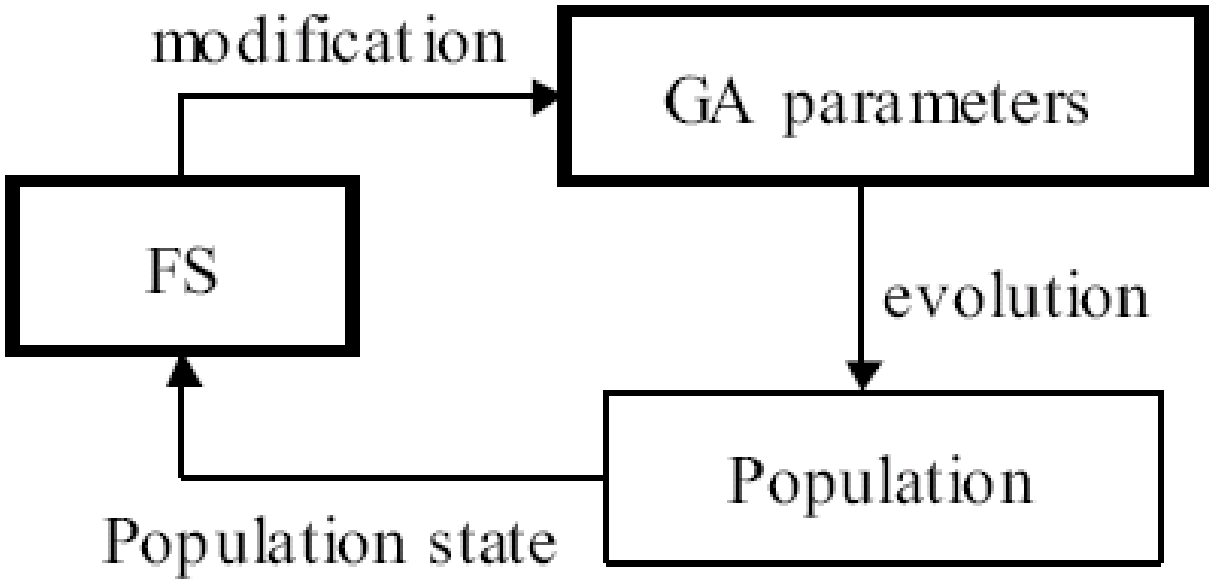
$$y_1' = a_{11}C_1 + a_{21}C_2 + \cdots + a_{n_r 1}C_{n_r}$$

$$y_2' = a_{12}C_1 + a_{22}C_2 + \cdots + a_{n_r 2}C_{n_r}$$

$$\vdots$$

$$y_{n_s}' = a_{1n_s}C_1 + a_{2n_s}C_2 + \cdots + a_{n_r n_s}C_{n_r}$$

Controlling parameters of genetic algorithms with fuzzy systems



IF average fitness is high **THEN** population size should be increased.
IF best fitness is not improved **THEN** mutation rate should be increased.

Fuzzy logic application in Medicine

Introduction

- Medicine is one field in which the applicability of fuzzy set theory was recognized quite early, in the mid-1970s.
- Within this field, it is the uncertainty found in the **process of diagnosis of disease** that has most frequently been the focus of applications of fuzzy set theory.

Introduction (cond…)

- With the increased volume of information available to physicians from new medical technologies, the process of classifying different sets of symptoms under a single name and determining appropriate therapeutic action becomes increasingly difficult.
- A single disease may manifest itself quite differently in different patients and at different disease stages.
- Furthermore, a single symptom may be indicative of several different diseases, and the presence of several diseases in a single patient may disrupt the expected symptom pattern of any one them.

Introduction (cond…)

- The best and most useful descriptions of disease entities often use linguistic terms that are irreducibly vague. For example, hepatitis is characterized by the statement:

  "Total proteins are *usually normal*, albumin is *decreased*, α-globulins are *slightly decreased*, β-globulins are *slightly decreased*, and γ-globulins are *increased*",

- Where the linguistic terms printed in italics are inherently vague.
- Although medical knowledge concerning the symptom-disease relationship constitutes one source of imprecision and uncertainty in the diagnostic process, the knowledge concerning the state of patient constitutes another.

Introduction (cond…)

- The physician generally gathers knowledge about the patient from the **past history**, **physical examination**, **laboratory test results**, and other **investigative procedures such as X-rays and ultrasonic's**.
- The knowledge provided by each of these sources carries with it varying degrees of uncertainty.
- The past history offered by the patient may be subjective, exaggerated, underestimated, or incomplete.
- Mistakes may be made in the physical examination, and symptoms may be overlooked.

Introduction (cond…)

- The measurements provided by laboratory tests are often of limited precision, and the exact borderline between normal and pathological is often unclear.
- X-rays and other similar procedures require correct interpretation of the results.
- Thus, **the state and symptoms of the patient can be known by the physician with only a limited degree of precision**.
- In the face of the uncertainty concerning the observed symptoms of the patient as well as the uncertainty concerning the relation of the symptoms to a disease entity, it is nevertheless crucial that the physician determine the diagnostic label that will entail the appropriate therapeutic regimen.

Introduction (cond…)

- The desire to better understand and teach this difficult and important process of medical diagnosis has prompted attempts to model it with the use of fuzzy sets.
- These models vary in the degree to which they attempt to deal with different complicating aspects of medical diagnosis disease stages, relations between diseases themselves, and the stages of hypothesis formation, preliminary diagnosis, and final diagnosis within the diagnostic process itself.
- These models also form the basis for computerized medical expert systems, which are usually designed to aid the physician in the diagnosis of some specified category of diseases.

Introduction (cond…)

- The fuzzy set framework has been utilized in several different approaches to modeling the diagnostic process.
- In the approach formulated by Sanchez [1979], the physician's medical knowledge is represented as a fuzzy relation between symptoms and diseases.
- Thus, given the fuzzy set A of the symptoms observed in the patient and the fuzzy relation R representing the medical knowledge that relates the symptoms in set S to the diseases.
- In set D, then the fuzzy set B of the possible diseases of the patient can be inferred by means of the compositional rule of inference.

Introduction (cond…)

$$B = A \circ R$$

or

$$B(d) = \max_{s \in S} [\min(A(s), R(s, d))]$$

for each $d \in D$. The membership grades of observed symptoms in fuzzy set A may represent the degree of possibility of the presence of the symptom or its severity. The membership grades in fuzzy set B denote the degree of possibility with which we can attach each relevant diagnostic label to the patient. The fuzzy relation R of medical knowledge should constitute the greatest relation such that given the fuzzy relation Q on the set P of patients and S of symptoms and the fuzzy relation T on the sets P of patients and D of diseases, then

$$T = Q \circ R.$$

Fuzzy sets and fuzzy relations involved in medical diagnosis