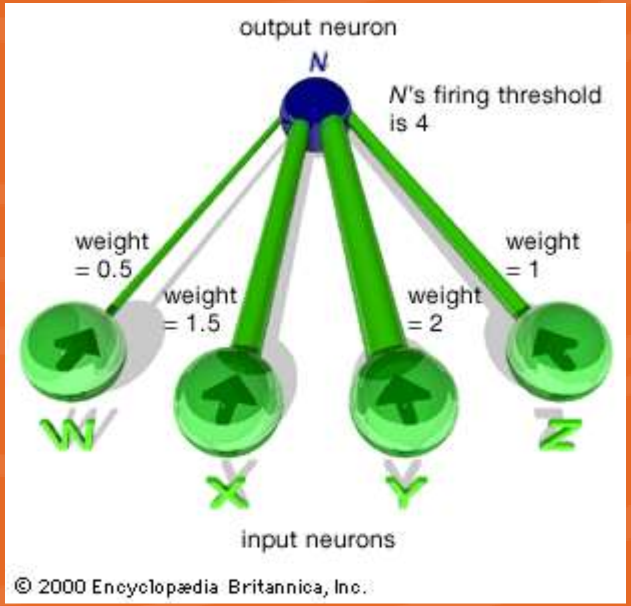
Machine Learning

Machine Learning

- ✓ Connectionist
- ✓ Perceptron learning.
- ✓ Social and Emergent Models of Learning
- ✓ The Genetic Algorithm
- ✓ Artificial Life and Society based Learning.

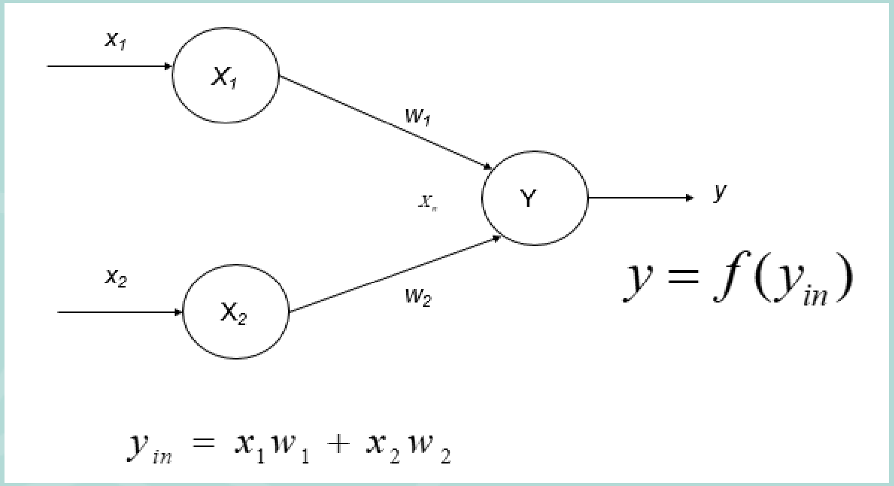
Connectionism

- An approach to AI that developed out of attempts to understand how the human brain works at the neural level and, in particular, how people learn and remember.
- This approach is sometimes referred to as neuron like computing.
- Connectionism is a movement in cognitive science that hopes to explain intellectual abilities using artificial neural networks

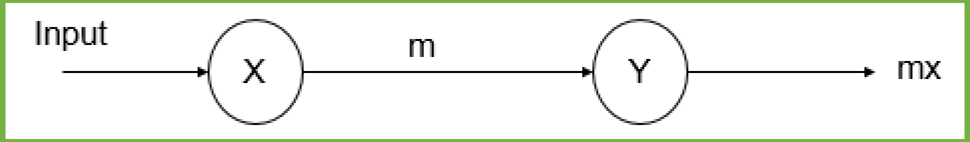


Fundamentals of Neural network

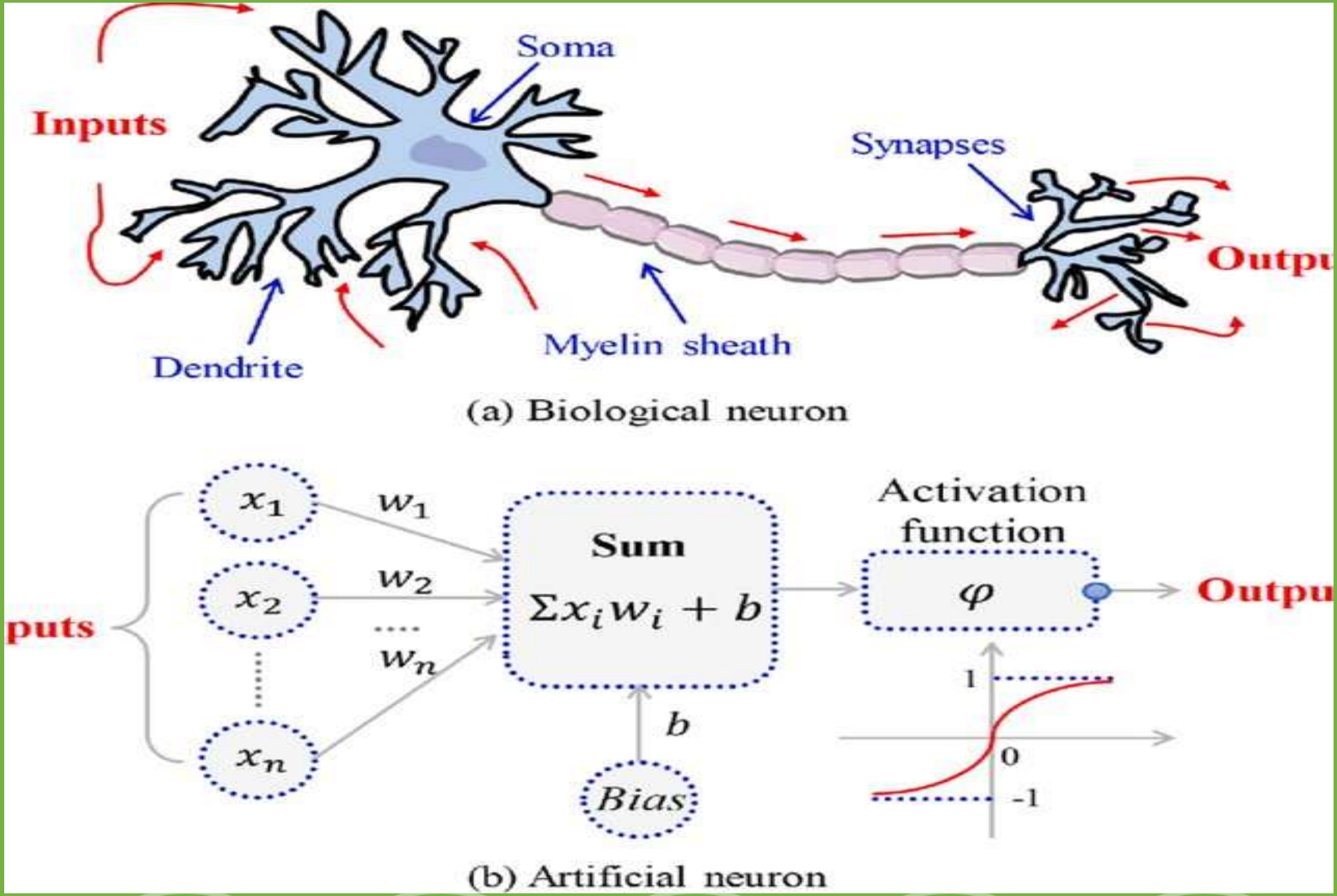
- NN is constructed and implemented to model the human brain.
- Performs tasks such as pattern-matching, classification, optimization function, approximation, vector quantization, and data clustering.
- ANN possesses many processing elements called nodes/neurons which operate in parallel.
- Neurons are connected with others by connection links.
- Each link is associated with weights that contain information about the input signal.
- Each neuron has an internal state of its own which is a function of the inputs that the neuron receives- Activation level



Neural net of pure linear eqn



Biological vs Artificial Neuron



(a) Biological neuron

(b) Artificial neuron

Biological Neuron

Has 3 parts

- Soma or cell body:- cell nucleus is located
- Dendrites:- nerve connected to cell body
- Axon: carries impulses of the neuron

End of axon splits into fine strands

Each strand terminates into a bulb-like organ called synapse

Electric impulses are passed between the synapse and dendrites

Synapses are of two types

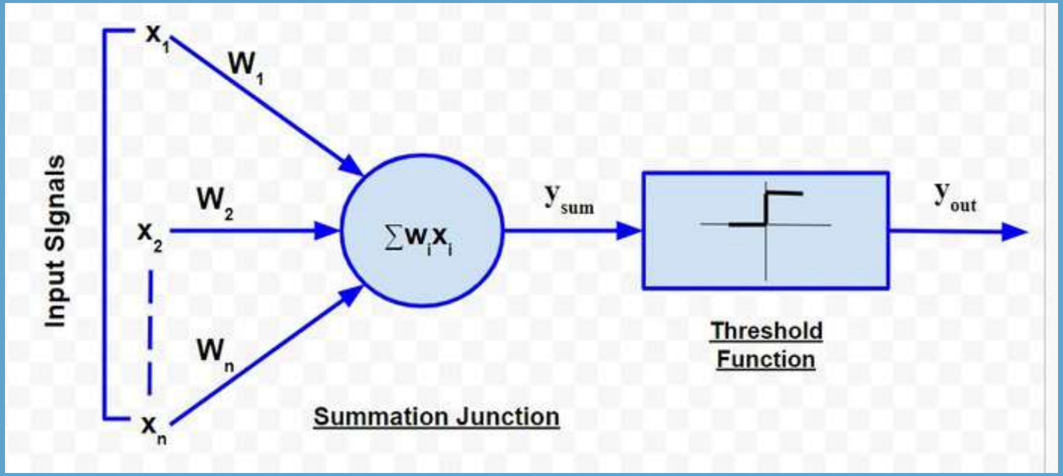
- Inhibitory:- impulses hinder the firing of the receiving cell
- Excitatory:- impulses cause the firing of the receiving cell

Neuron fires when the total of the weights to receive impulses exceeds the threshold value during the latent summation period

After carrying a pulse an axon fiber is in a state of complete non-excitability for a certain time called the refractory period.

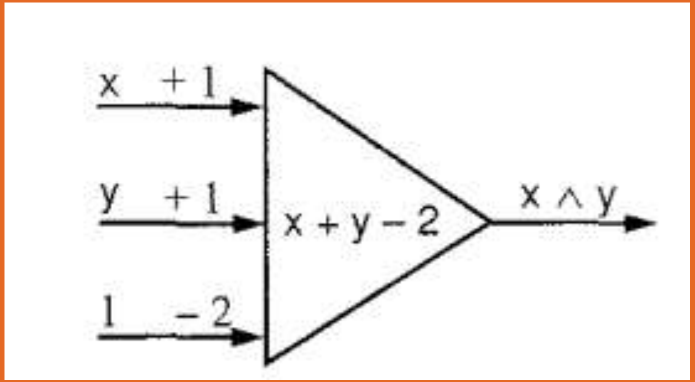
McCulloch-Pitts Neuron Model

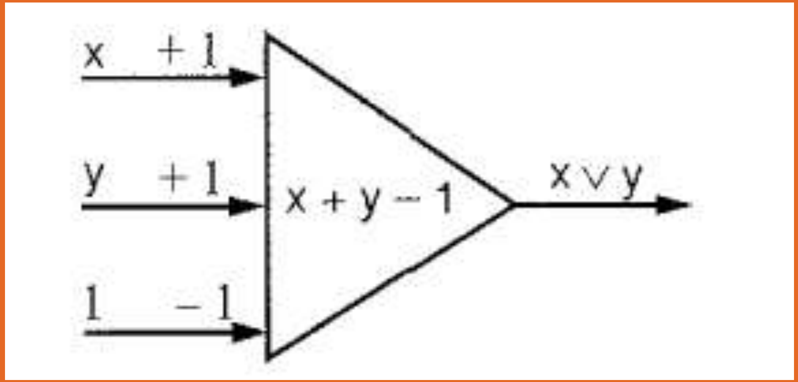
- The first computational model of a neuron was proposed by Warren MuCulloch (neuroscientist) and Walter Pitts (logician) in 1943.

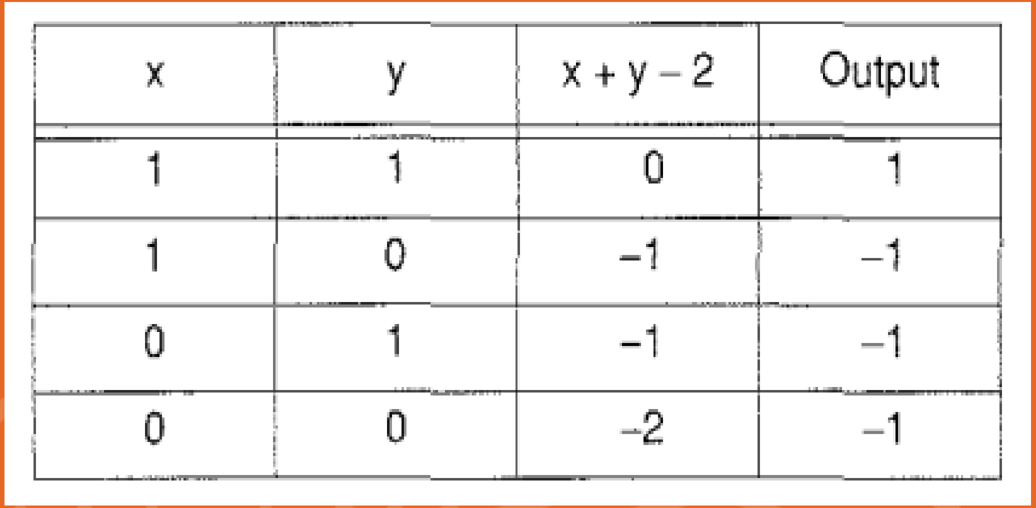


- Only 2 types of inputs - Excitatory and Inhibitory.
- The excitatory inputs have weights of positive magnitude and the inhibitory weights have weights of negative magnitude.
- The inputs of the McCulloch-Pitts neuron could be either 0 or 1.
- It has a threshold function as an activation function.
- So, the output signal yout is 1 if the input ysum is greater than or equal to a given threshold value, else 0.

McCulloch-Pitts Neuron Model for AND & OR





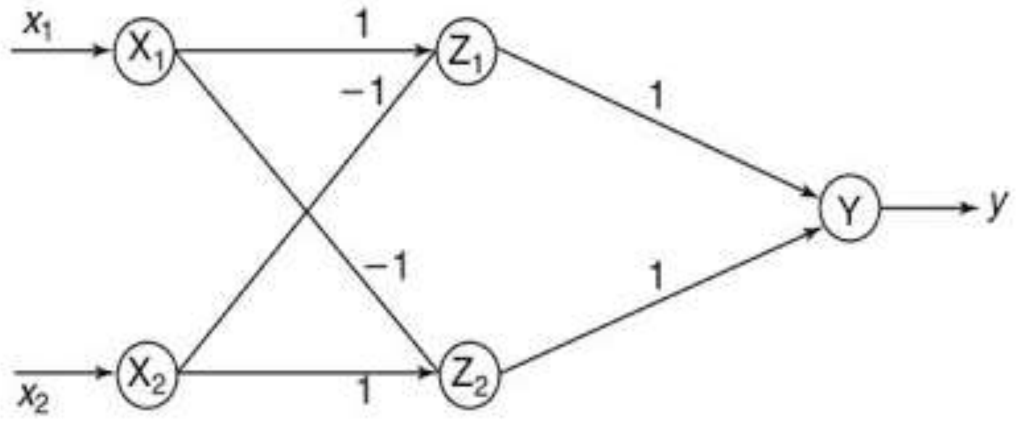


x	y	x + y − 2	Output
1	1	0	1
1	0	–1	–1
0	1	–1	–1
0	0	–2	–1

X	Y	X+Y-1	Output

Implement XOR function using McCulloch–Pitts Neuron

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0



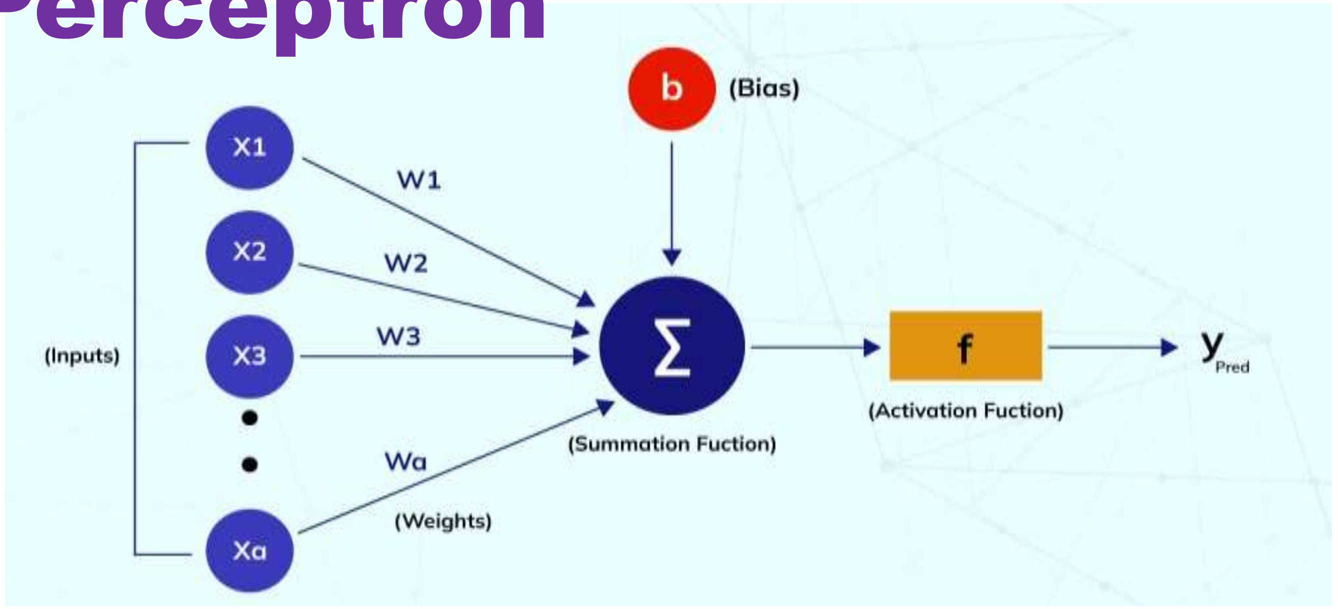
Features of McCulloch-Pitts model

- Allows binary 0,1 states only
- Operates under a discrete-time assumption
- Weights and the neurons’ thresholds are fixed in the model and no interaction among network neurons
- Just a primitive model

Perceptron

- Perceptron is the elementary unit of an Artificial Neural Network.
- It is a single-layer neural network linear or a Machine Learning algorithm used for supervised learning of various binary classifiers.
- It works as an artificial neuron to perform computations by learning elements and processing them for detecting the business intelligence and capabilities of the input data.
- A perceptron network is a group of simple logical statements that come together to create an array of complex logical statements, known as the neural network.

Perceptron



- Input Values: A set of values or a dataset for predicting the output value. They are also described as a dataset’s features and dataset.
- Weights: The real value of each feature is known as weight. It tells the importance of that feature in predicting the final value.
- Bias: The activation function is shifted towards the left or right using bias. You may understand it simply as the y-intercept in the line equation.
- Summation Function: The summation function binds the weights and inputs together. It is a function to find their sum.
- Activation Function: It introduces non-linearity in the perceptron model.

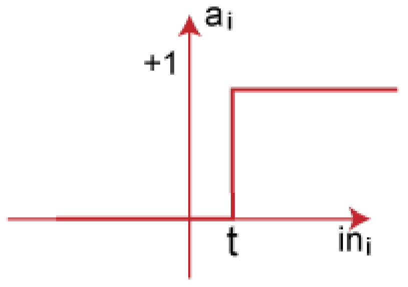
Weight & Bias

- Weights measure the importance of each feature in predicting output value.
- Features with values close to zero are said to have lesser weight or significance.
- These have less importance in the prediction
- High-weighted features have greater predictive power
- The weight can also be positive or negative.
- If the weight of a feature is positive then it has a direct relation with the target value,
- If it is negative then it has an inverse relationship with the target value.
- weight increases the speed of triggering an activation function

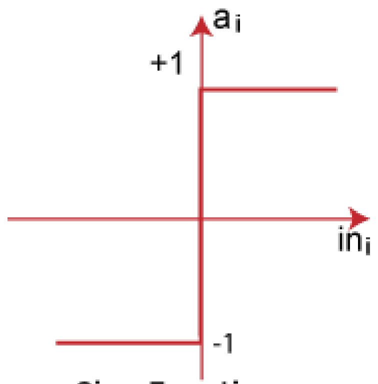
- Bias delays the trigger of the activation function.
- It acts like an intercept in a linear equation.
- Bias is a constant used to adjust the output and help the model to provide the best-fit output for the given data.

Activation Function

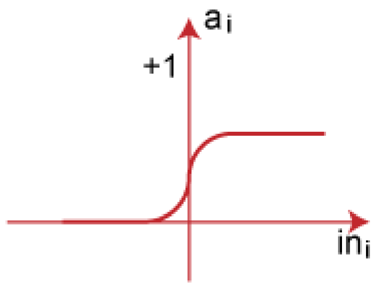
- These are the final and important components that help to determine whether the neuron will fire or not.
- Activation Function can be considered primarily as a step function.
- Types of Activation functions:
    - Sign function
    - Step function, and
    - Sigmoid function



Step Function



Sign Function



Sigmoid Function

How does Perceptron work?

- In the first step first, multiply all input values with corresponding weight values and then add them to determine the weighted sum.

Mathematically, we can calculate the weighted sum as follows:

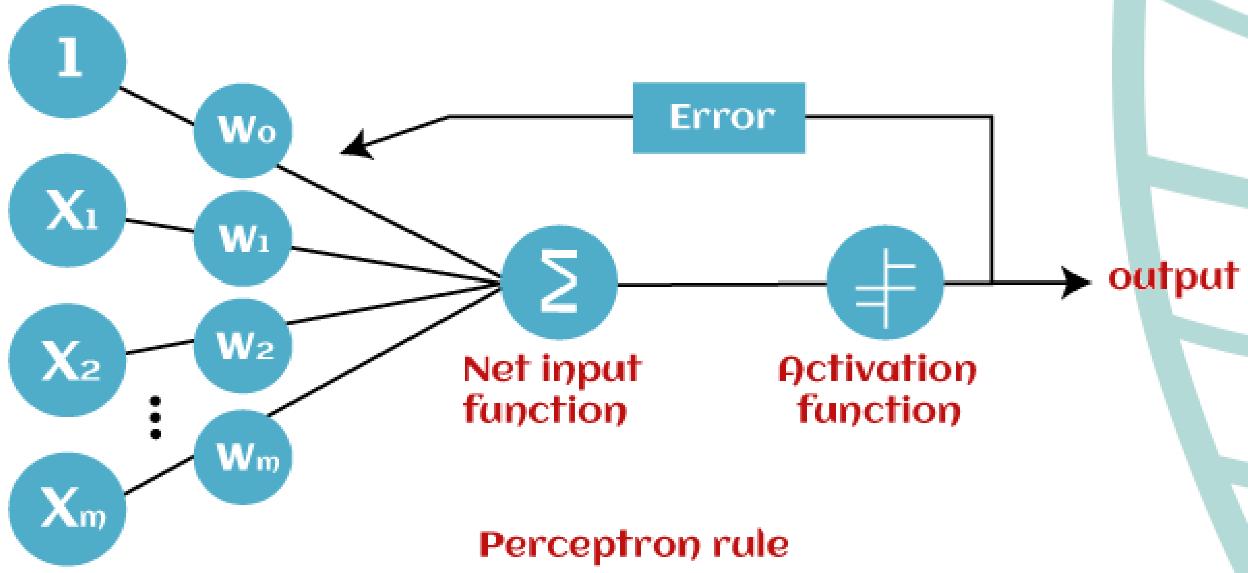
∑wi*xi = x1*w1 + x2*w2 +…wn*xn

Add a special term called **bias 'b'** to this weighted sum to improve the model's performance.

∑wi*xi + b

- In the second step, an activation function is applied with the above-mentioned weighted sum, which gives us output either in binary form or a continuous value as follows:

Y = f(∑wi*xi + b)



Types of Perceptron Models

Single-layer Perceptron Model

- consists feed-forward network and also includes a threshold transfer function inside the model.
- The main objective is to analyze the linearly separable objects with binary outcomes.
- After adding all inputs, if the total sum of all inputs is more than a pre-determined value, the model gets activated and shows the output value as +1.
- *Single-layer perceptron can learn only linearly separable patterns*

Types of Perceptron Models

Multi-layer Perceptron model

- a multi-layer perceptron model also has the same model structure but has a greater number of hidden layers.
- The multi-layer perceptron model is also known as the Backpropagation algorithm, which executes in two stages as follows:
  - Forward Stage: Activation functions start from the input layer in the forward stage and terminate on the output layer.
  - Backward Stage: In the backward stage, weight and bias values are modified as per the model's requirement. In this stage, the error between actual output and demanded originated backward on the output layer and ended on the input layer

Social and Emergent Models of Learning

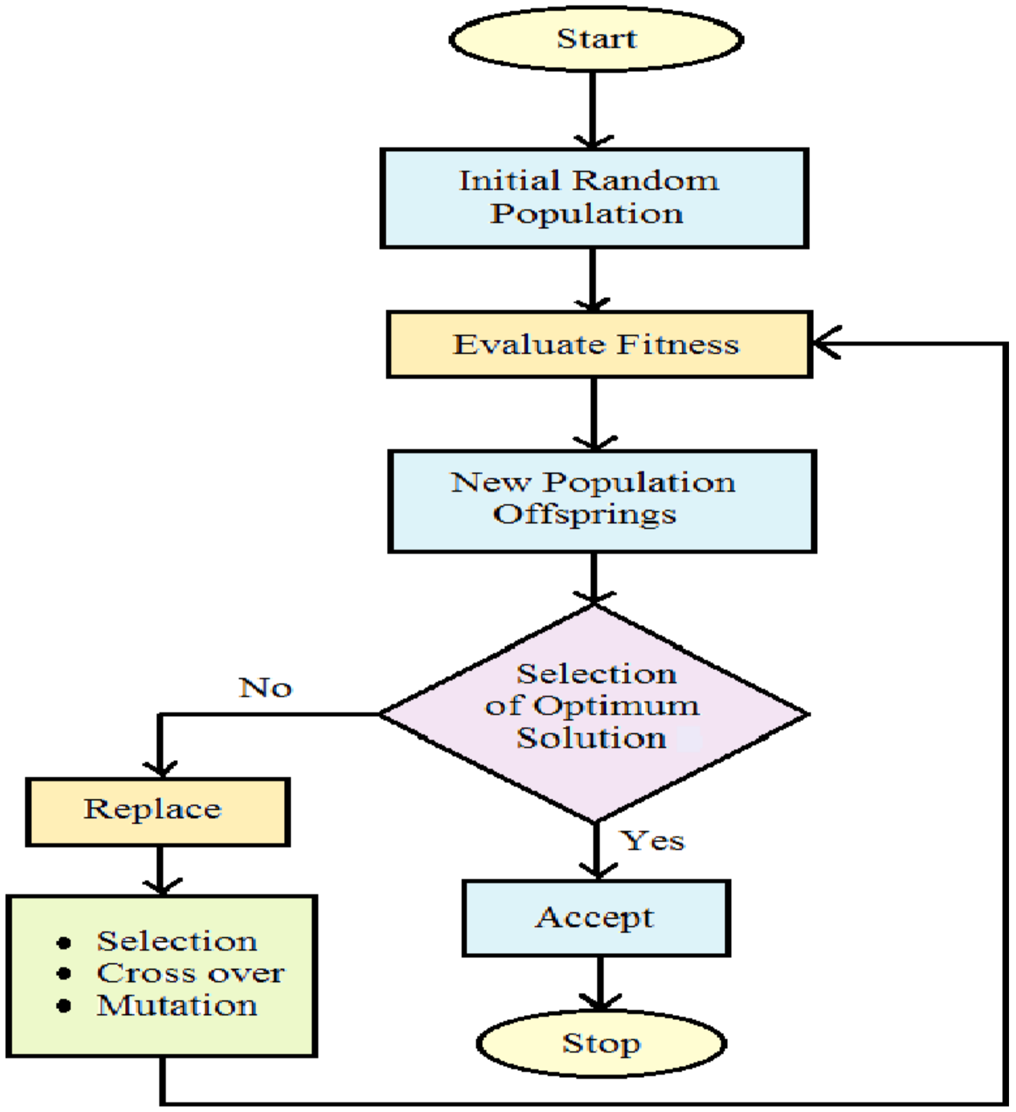
- Biological analogies influenced the design of machine learning algorithms.
- Processes underlying evolution
- Emergence of species in natural evolution
- Populations of embodied individuals

Genetic Algorithm

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.
- Global search heuristics.
- Class of evolutionary algorithms that use techniques inspired by evolutionary biology such as inheritance, mutation, selection, and crossover (also called recombination).

Genetic Algorithm

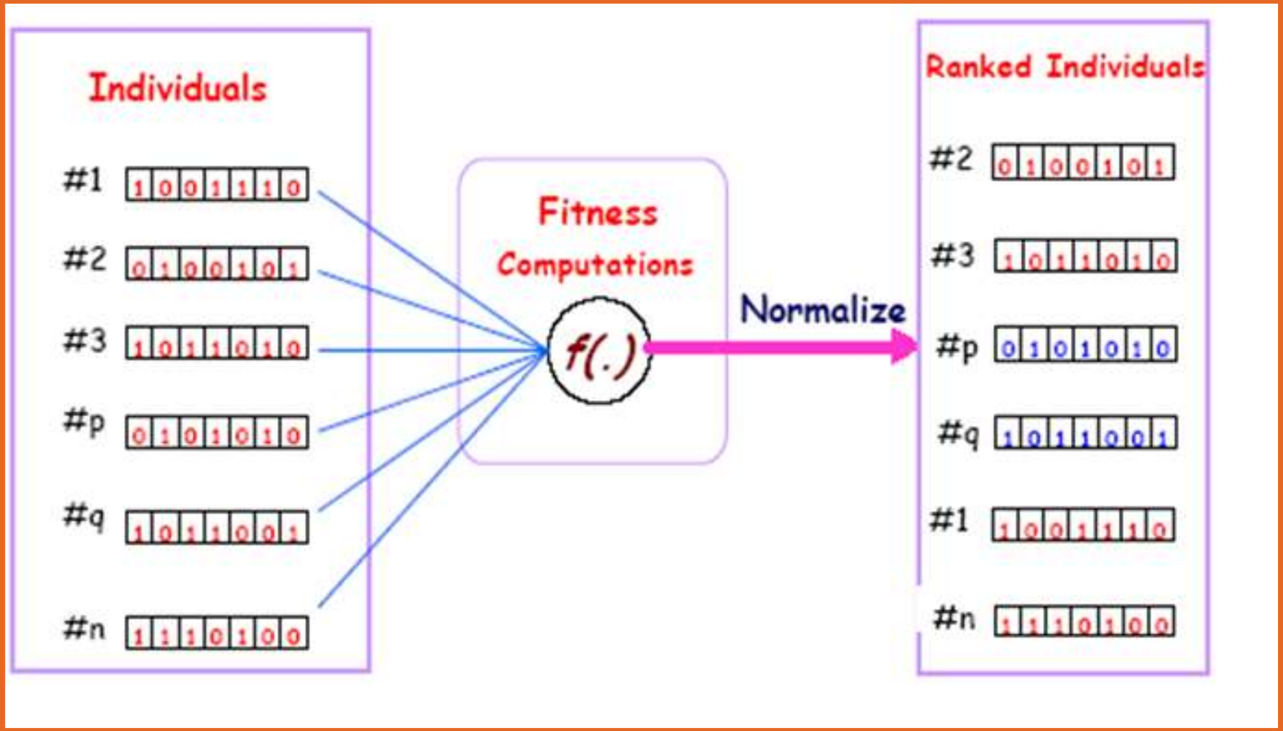

Start
Initial Random Population
Evaluate Fitness
New Population Offsprings
Selection of Optimum Solution
No
Yes
Replace
• Selection
• Cross over
• Mutation
Accept
Stop

What is GA?

- The evolution 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.
- The new population is then used in the next iteration of the algorithm.
- 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.

GA Requirements: Fitness

- The fitness function is defined over the genetic representation and measures the *quality* of the represented solution.
- The fitness function is always problem dependent.



General Algorithm for GA

- **Initialization**
- Initially many individual solutions are randomly generated to form an initial population.
- The population size depends on the nature of the problem, but typically contains several hundreds or thousands of possible solutions.
- Traditionally, the population is generated randomly, covering the entire range of possible solutions (the *search space*).
- Occasionally, the solutions may be "seeded" in areas where optimal solutions are likely to be found.

General Algorithm for GA

- **Selection**
- During each successive generation, a proportion of the existing population is selected to breed a new generation.
- Individual solutions are selected through a *fitness-based* process, where fitter solutions (as measured by a fitness function) are typically more likely to be selected.
- Certain selection methods rate the fitness of each solution and preferentially select the best solutions. Other methods rate only a random sample of the population, as this process may be very time-consuming.
- Most functions are stochastic and designed so that a small proportion of less fit solutions are selected. This helps keep the diversity of the population large, preventing premature convergence on poor solutions. Popular and well-studied selection methods include roulette wheel selection and tournament selection.

General Algorithm for GA

- **Reproduction**
- The next step is to generate a second generation population of solutions from those selected through genetic operators:
- crossover (also called recombination), and/or mutation.
- For each new solution to be produced, a pair of "parent" solutions is selected for breeding from the pool selected previously.
- By producing a "child" solution using the above methods of crossover and mutation, a new solution is created which typically shares many of the characteristics of its "parents".
- New parents are selected for each child, and the process continues until a new population of solutions of appropriate size is generated.

General Algorithm for GA

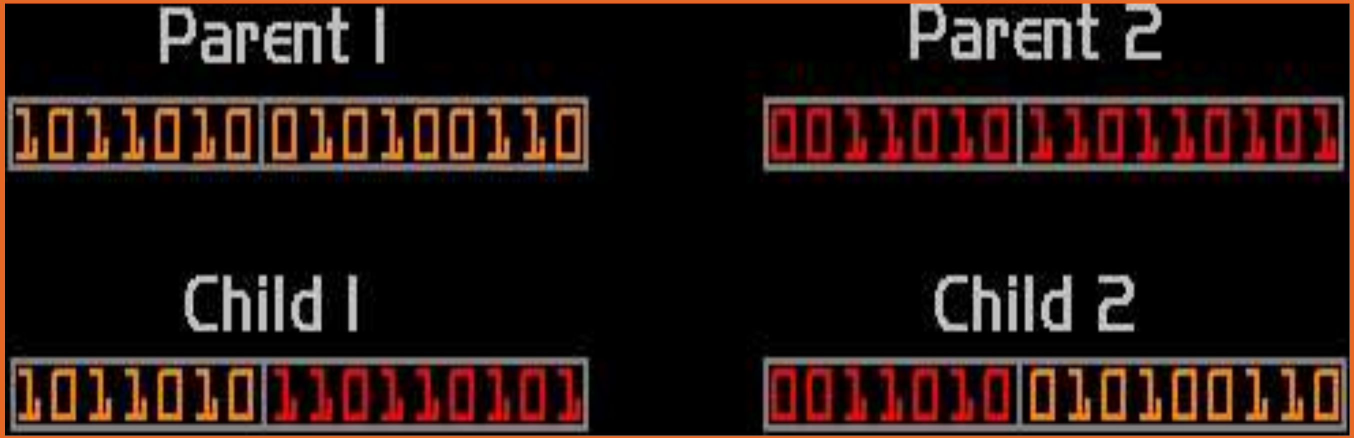
- These processes ultimately result in the next generation population of chromosomes that is different from the initial generation.

- Generally the average fitness will have increased by this procedure for the population, since only the best organisms from the first generation are selected for breeding, along with a small proportion of less fit solutions.

Crossover

- The most common type is single point crossover.
- In single point crossover, you choose a locus at which you swap the remaining alleles from on parent to the other.
- The children take one section of the chromosome from each parent.
- The point at which the chromosome is broken depends on the randomly selected crossover point.
- This particular method is called single point crossover because only one crossover point exists. Sometimes only child 1 or child 2 is created, but oftentimes both offspring are created and put into the new population.
- Crossover does not always occur, however. Sometimes, based on a set probability, no crossover occurs and the parents are copied directly to the new population. The probability of crossover occurring is usually 60% to 70%.

Crossover

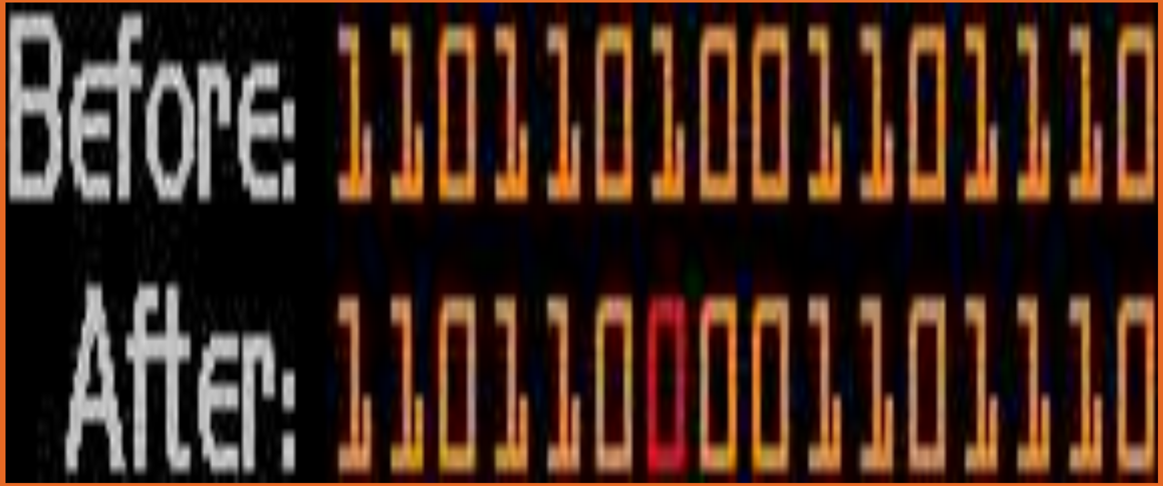


Parent 1
1011010 010100110
Parent 2
0011010 110110101
Child 1
1011010 110110101
Child 2
0011010 010100110

Mutation

- After selection and crossover, you now have a new population full of individuals.
- Some are directly copied, and others are produced by crossover.
- In order to ensure that the individuals are not all exactly the same, you allow for a small chance of mutation.
- You loop through all the alleles of all the individuals, and if that allele is selected for mutation, you can either change it by a small amount or replace it with a new value. The probability of mutation is usually between 1 and 2 tenths of a percent.
- Mutation is fairly simple. You just change the selected alleles based on what you feel is necessary and move on. Mutation is, however, vital to ensuring genetic diversity within the population.

Mutation


Before: 1101101001101110

After: 1101100001101110

General Algorithm for GA

- **Termination**
- This generational process is repeated until a termination condition has been reached.
- Common terminating conditions are:
  - A solution is found that satisfies minimum criteria
  - Fixed number of generations reached
  - Allocated budget (computation time/money) reached
  - The highest ranking solution's fitness is reaching or has reached a plateau such that successive iterations no longer produce better results
  - Manual inspection
  - Any Combinations of the above

GA Pseudo-code

```
Choose initial population
Evaluate the fitness of each individual in the population
Repeat
    Select best-ranking individuals to reproduce
    Breed new generation through crossover and mutation
    (genetic operations) and give birth to offspring
    Evaluate the individual fitnesses of the offspring
    Replace worst ranked part of population with offspring
Until <terminating condition>
```

A Simple Example

The Traveling Salesman Problem:

Find a tour of a given set of cities so that

- each city is visited only once
- the total distance traveled is minimized

Representation

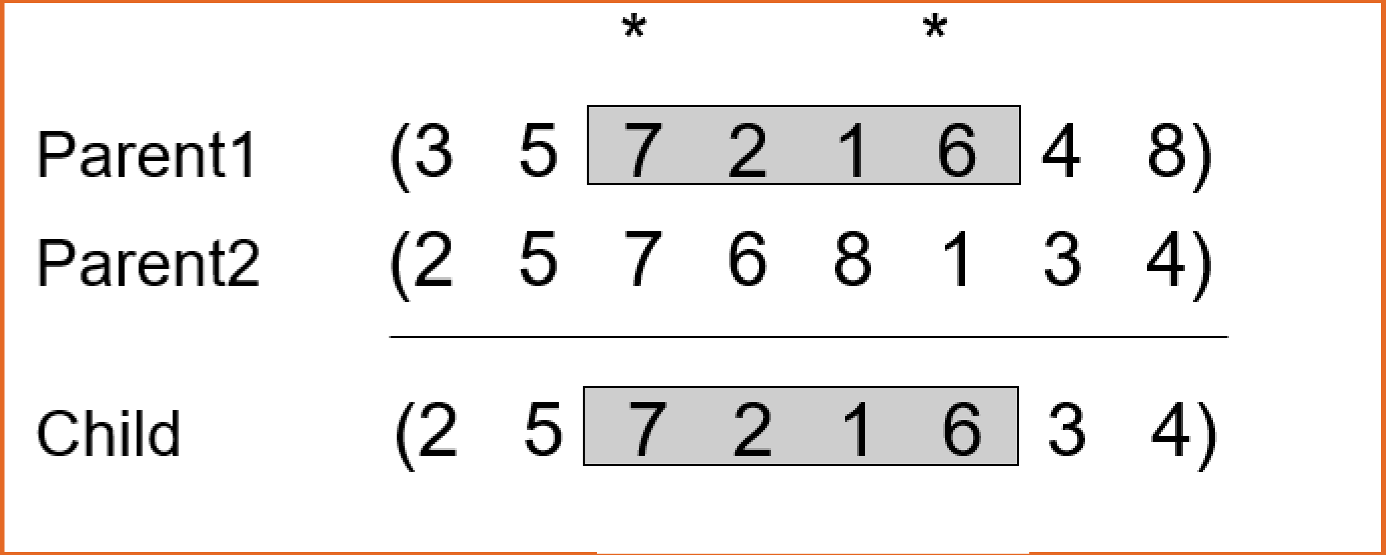
Representation is an ordered list of city numbers known as an *order-based* GA.

1) London	3) Dunedin	5) Beijing	7) Tokyo
2) Venice	4) Singapore	6) Phoenix	8) Victoria

```
CityList1    (3  5  7  2  1  6  4  8)
CityList2    (2  5  7  6  8  1  3  4)
```

Crossover

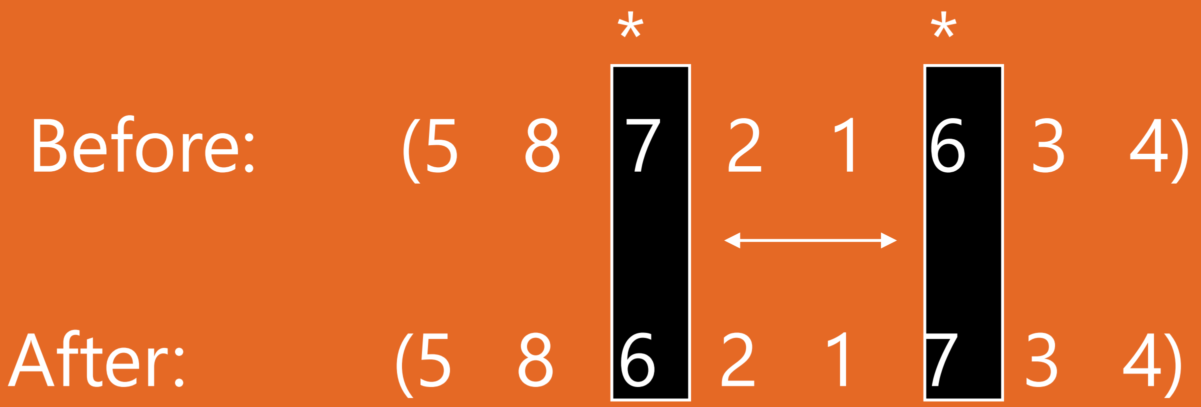
Crossover combines inversion and recombination:



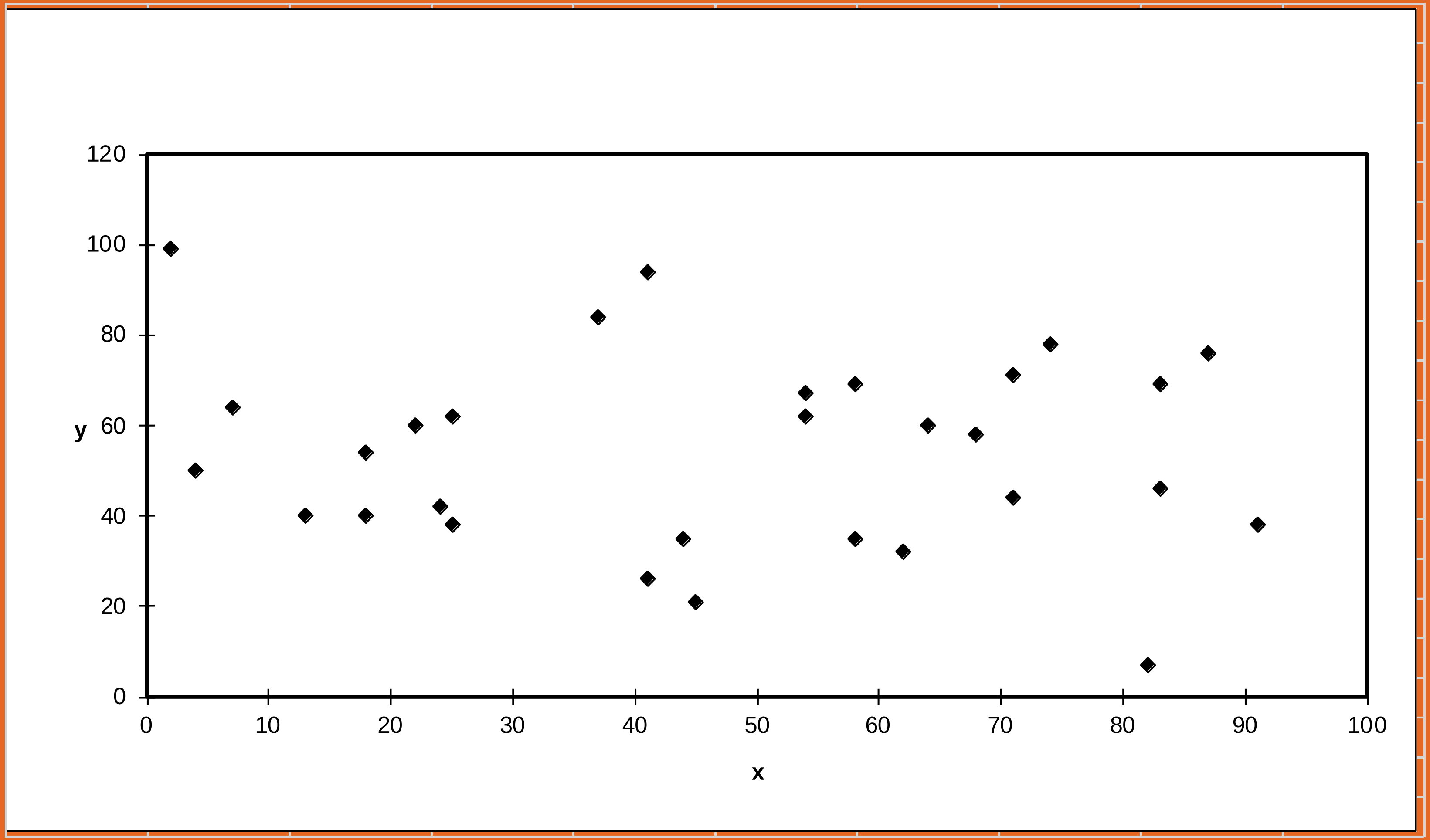
This operator is called the *Order1* crossover.

Mutation

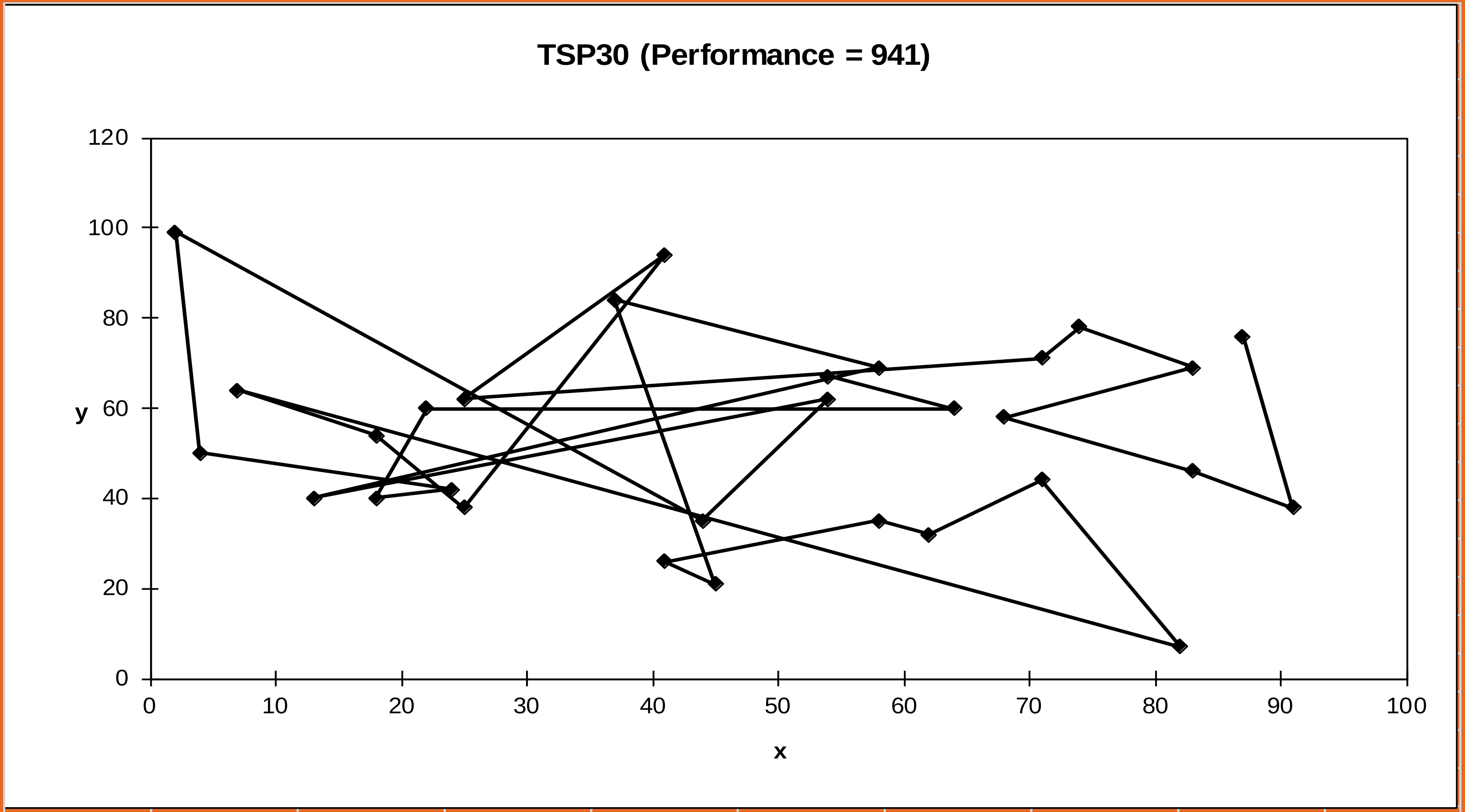
Mutation involves reordering of the list:



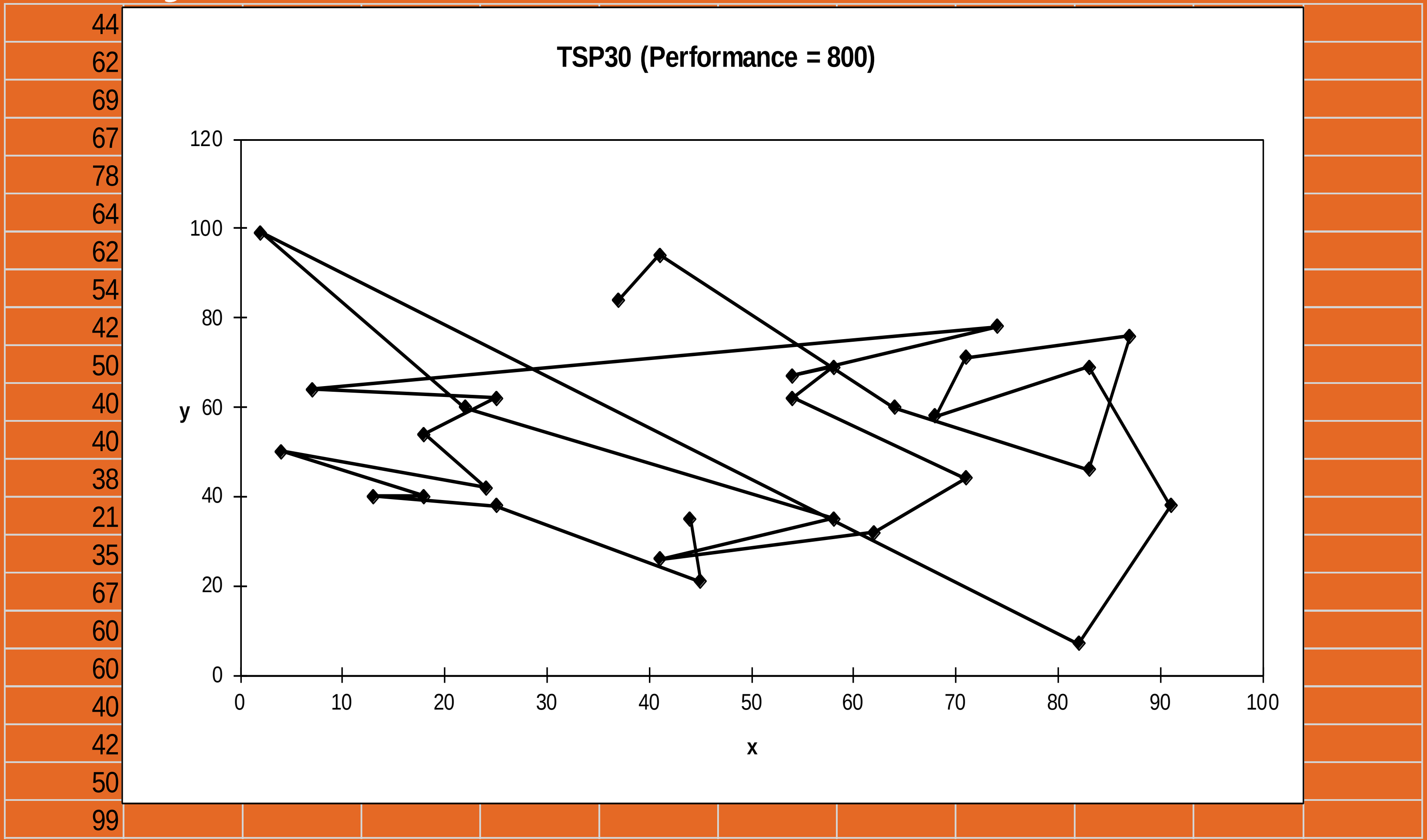
TSP Example: 30 Cities



Solution $_i$ (Distance = 941)

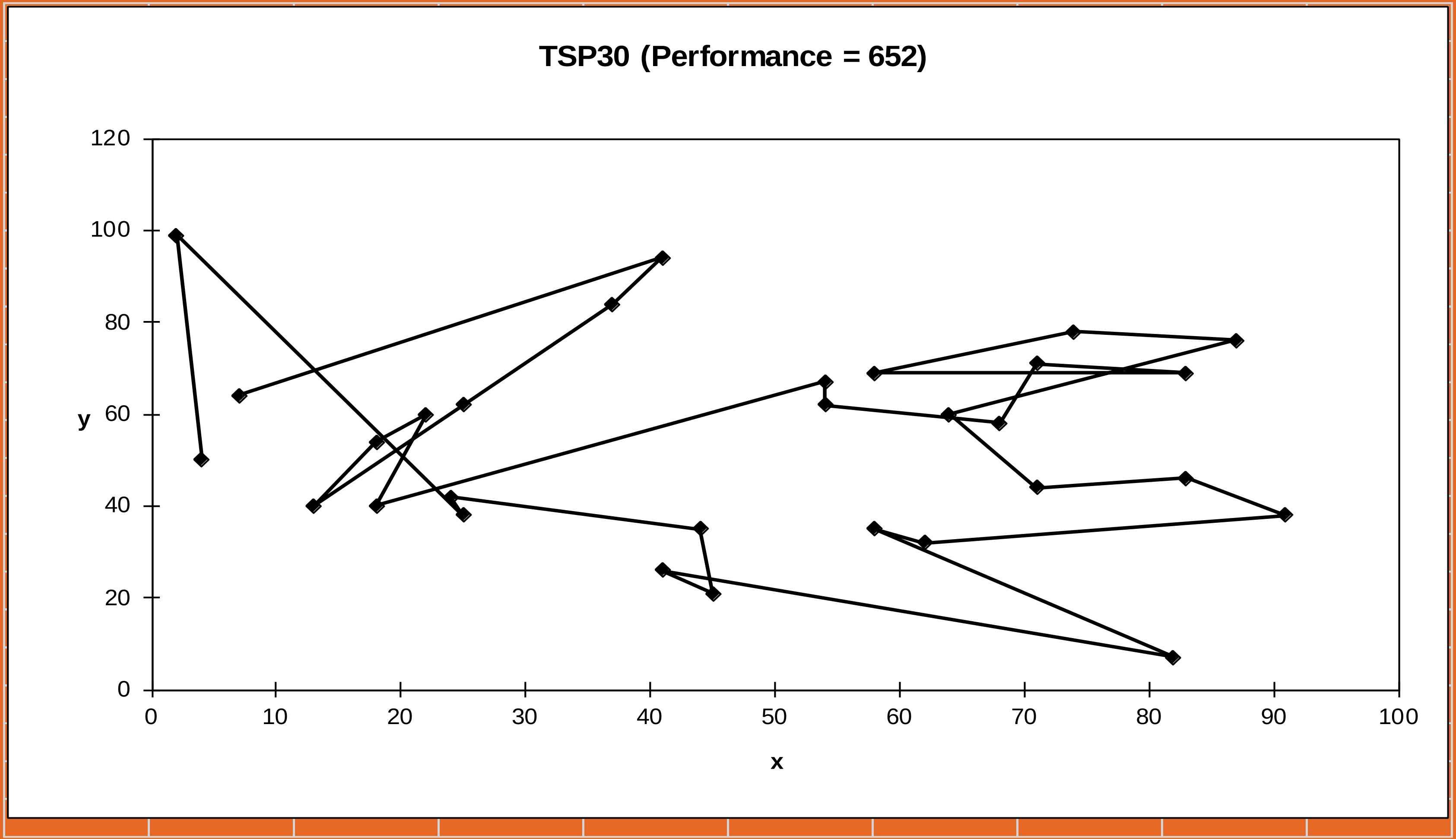


Solution $_j$(Distance = 800)



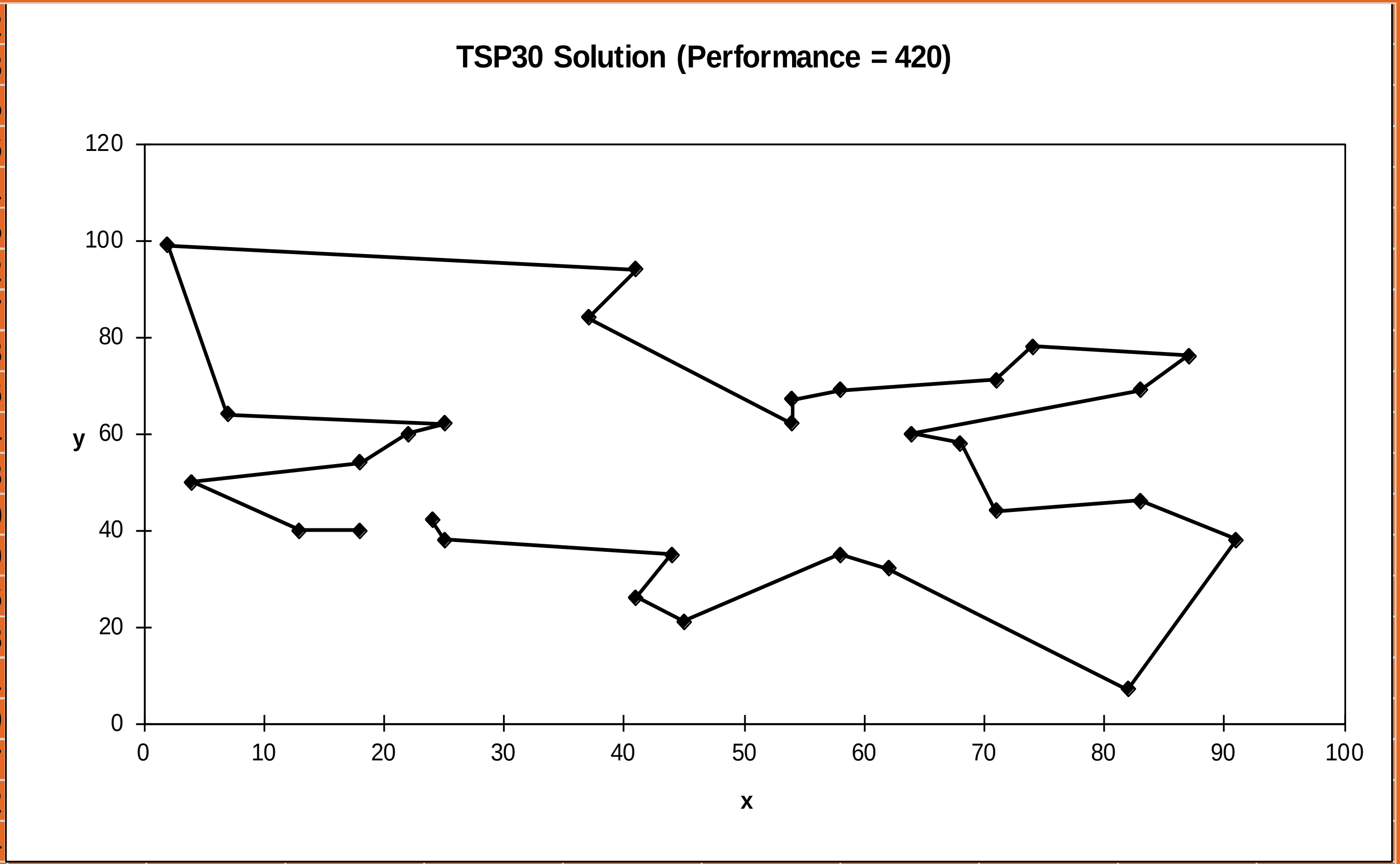
44
62
69
67
78
64
62
54
42
50
40
40
38
21
35
67
60
60
40
42
50
99

Solution $_k$(Distance = 652)

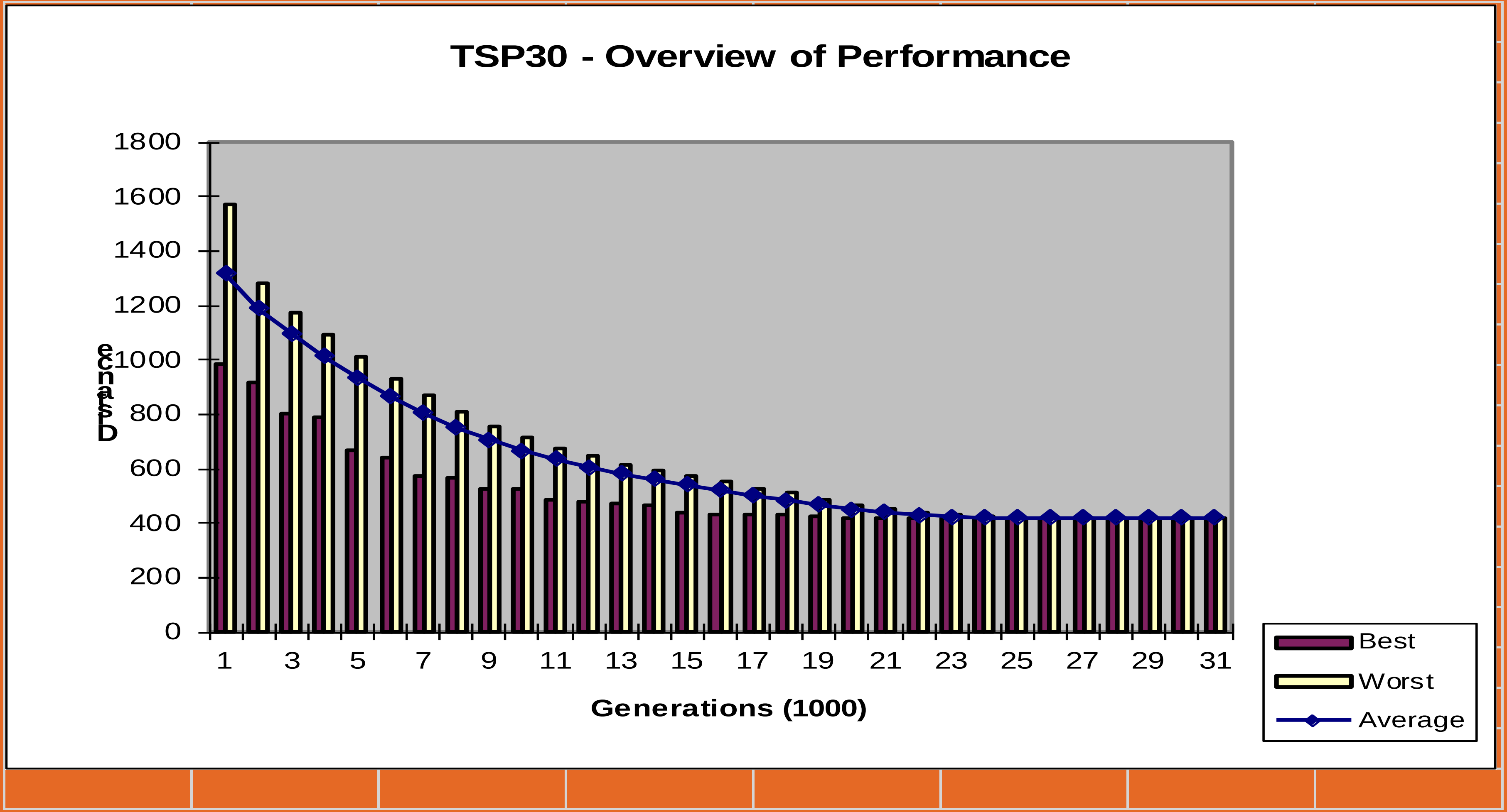


Best Solution (Distance = 420)

42
38
35
26
21
35
32
7
38
46
44
58
60
69
76
78
71
69
67
62
84
94



Overview of Performance



Artificial and Social Based Learning

- Focuses on creating synthetic systems that mimic or simulate life-like behavior found in biological organisms.
    - **Emergence:** ALife systems often exhibit emergent properties, where complex behavior arises from simple rules followed by individual agents.
    - **Evolutionary Algorithms:** Inspired by the process of natural selection and evolution involving genetic algorithms, etc
    - **Self-Organization:** ALife systems often demonstrate self-organization, where local interactions between agents give rise to global patterns or structures without central control.
    - **Cellular automata:** Cellular automata are discrete computational models composed of a grid of cells, each of which can be in a finite number of states

Artificial and Social Based Learning

- **Social-Based Learning in AI:** Involves designing systems that can learn from interactions with other agents or entities within a social context.
  - **Multi-Agent Systems (MAS):** MAS consist of multiple interacting agents, each with its own goals, knowledge, and capabilities. Agents can learn from each other through observation, communication, or collaboration.
  - **Reinforcement Learning with Social Learning:** In reinforcement learning, agents learn by interacting with an environment to maximize cumulative rewards. Social learning techniques extend this framework by allowing agents to learn from the experiences or policies of other agents in addition to their own.
  - **Imitation Learning:** Imitation learning involves learning from demonstrations provided by other agents or humans. This approach is particularly useful in scenarios where explicit reward signals may be sparse or difficult to define.
  - **Cultural Evolution:** Cultural evolution models how knowledge, behaviors, and norms spread and evolve within a population over time. In AI, cultural evolution can be simulated to study the emergence of complex behaviors or societal norms.