

Genetic Algorithm

BY

DR. ANUPAM GHOSH

DATE: 30.11.2024

Limitations of the traditional optimization approaches

- ❑ Computationally expensive
 - ❑ For a discontinuous objective function, methods may fail.
 - ❑ Method may not be suitable for parallel computing.
 - ❑ Discrete (integer) variables are difficult to handle.
 - ❑ Methods may not necessarily adaptive.
-
- ❑ Evolutionary algorithms have been evolved to address the above mentioned limitations of solving optimization problems with traditional approaches.

Evolutionary Algorithms

The algorithms, which follow some biological and physical behaviors:

Biologic behaviors:

- ❑ Genetics and Evolution → Genetic Algorithms (GA)
- ❑ Behavior of ant colony → Ant Colony Optimization (ACO)
- ❑ Human nervous system → Artificial Neural Network (ANN)
- ❑ In addition to that there are some algorithms inspired by some physical behaviors:

Physical behaviors:

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

A brief account on evolution

Evolution : Natural Selection

Four primary premises:

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

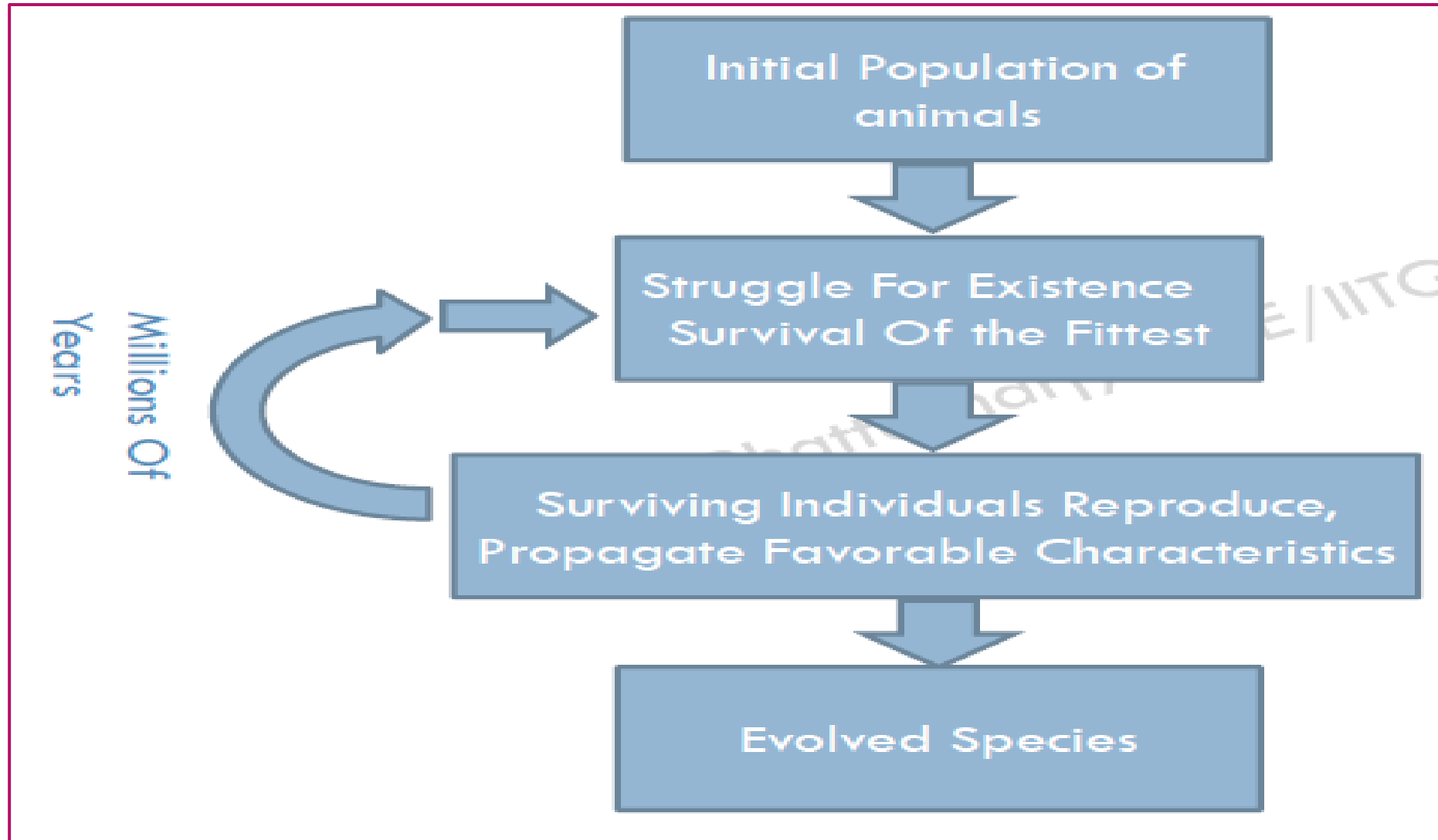
History of GAs

- As early as 1962, John Holland's work on adaptive systems laid the foundation for later developments.
- By the 1975, the publication of the book *Adaptation in Natural and Artificial Systems*, by Holland and his students and colleagues.
- early to mid-1980s, genetic algorithms were being applied to a broad range of subjects.
- In 1992 John Koza has used genetic algorithm to evolve programs to perform certain tasks. He called his method "genetic programming" (GP).

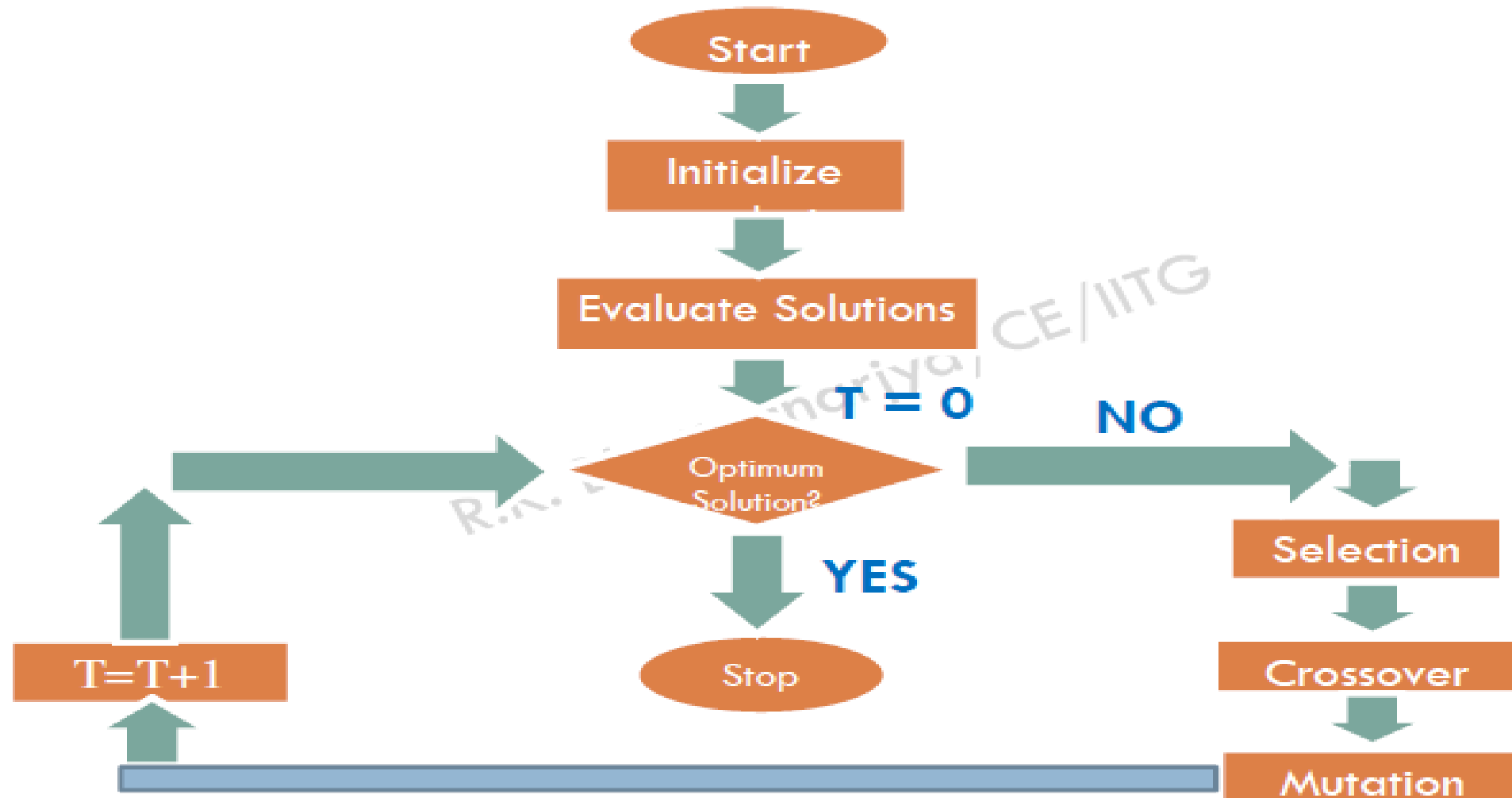
Introduction

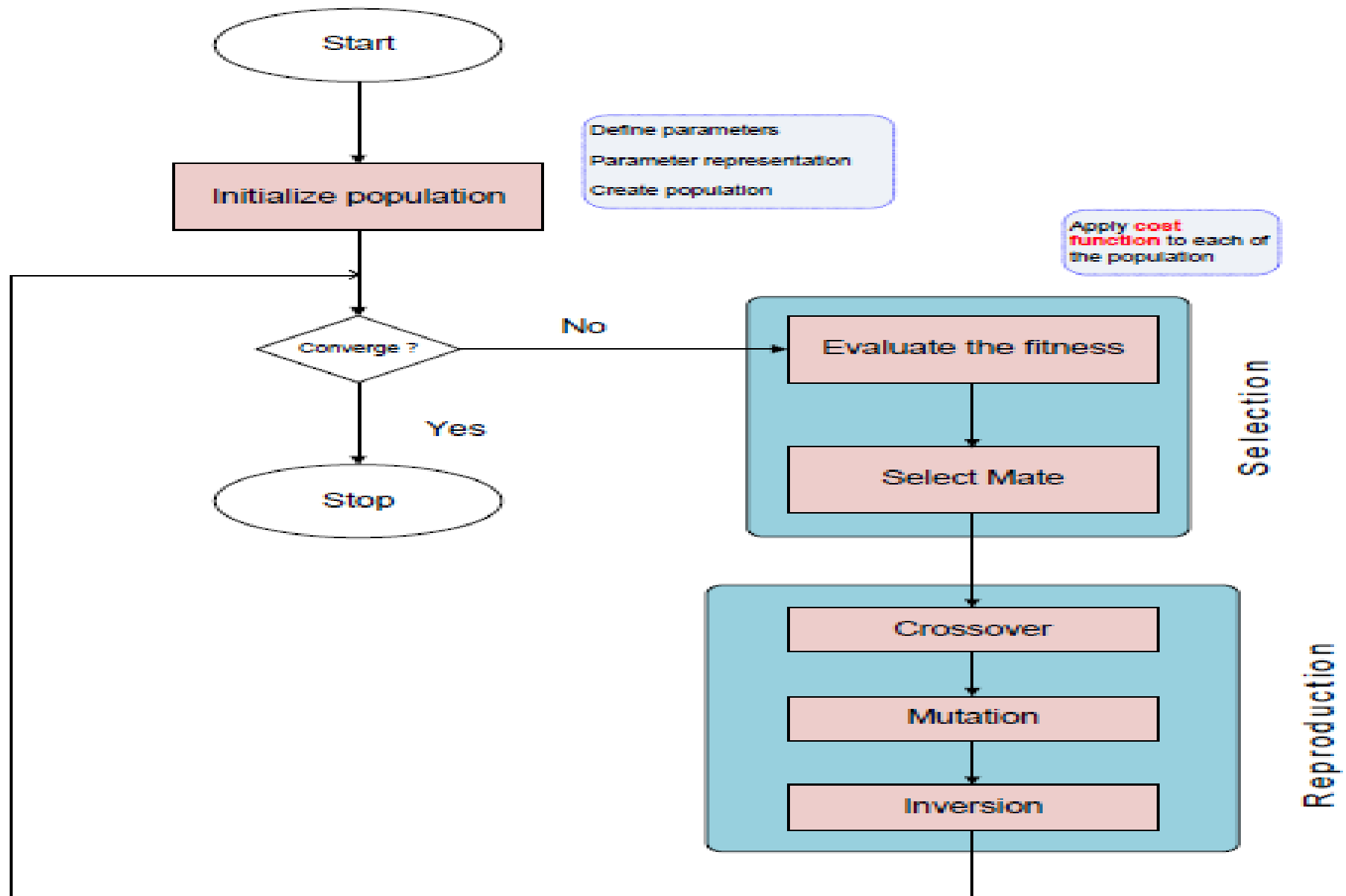
- ▶ Genetic Algorithms are the heuristic search and optimization techniques that mimic the process of natural evolution
- ▶ Genetic algorithms implement the optimization strategies by simulating evolution of species through natural selection
- ▶ GA Operators and Parameters:
 - ❑ Selection
 - ❑ Crossover
 - ❑ Mutation

Evolution of species



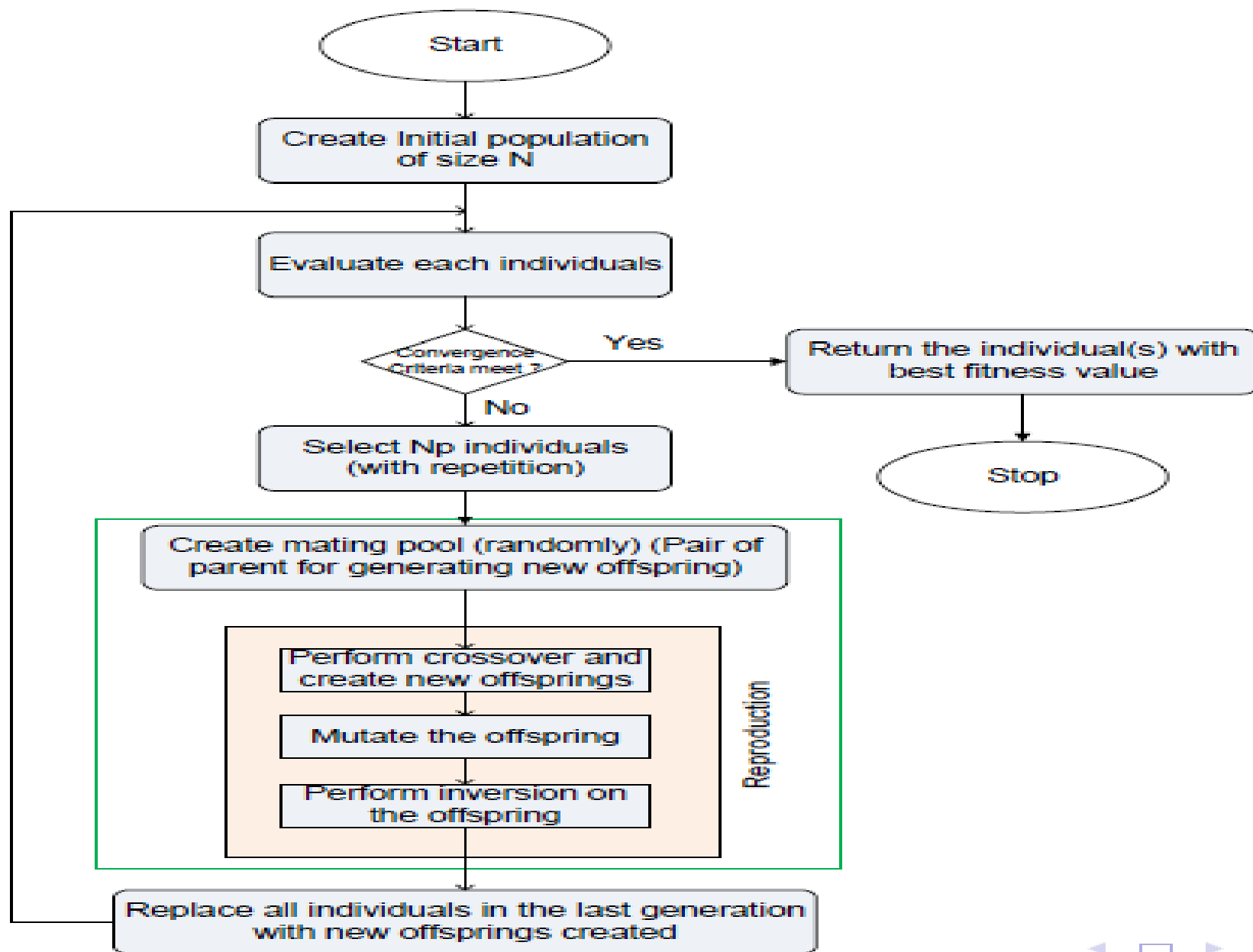
Simple Genetic Algorithms





GA Operators

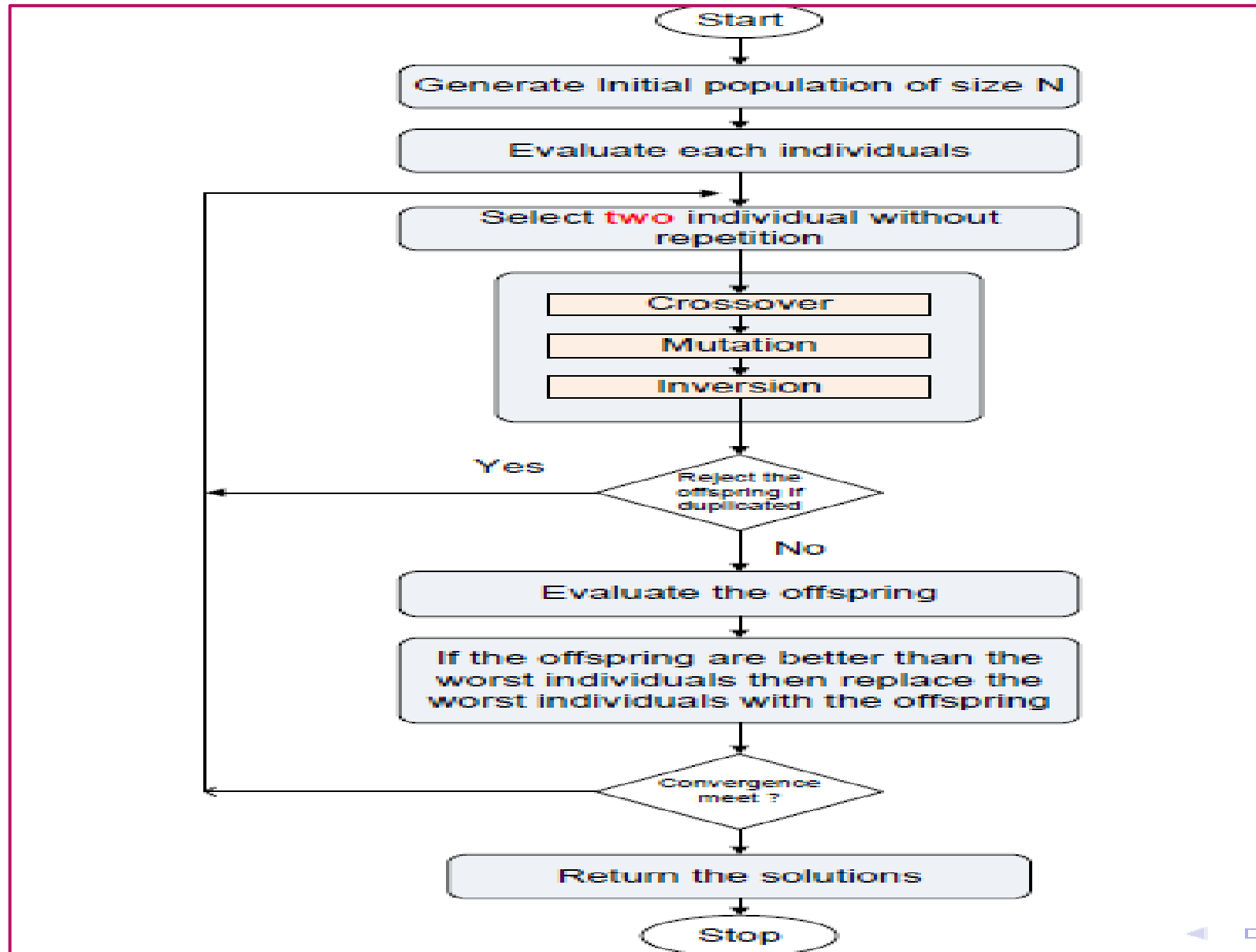
- ① **Encoding:** How to represent a solution to fit with GA framework.
- ② **Convergence:** How to decide the termination criterion.
- ③ **Mating pool:** How to generate next solutions.
- ④ **Fitness Evaluation:** How to evaluate a solution.
- ⑤ **Crossover:** How to make the diverse set of next solutions.
- ⑥ **Mutation:** To explore other solution(s).
- ⑦ **Inversion:** To move from one optima to other.



Simple GA features

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

Steady State Genetic Algorithm (SSGA)



SGA Features:

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

Limitations in SSGA:

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

Selection

- ▶ The process that determines which solutions are to be preserved and allowed to reproduce and which ones deserve to die out.
- ▶ The primary objective of the selection operator is to emphasize the good solutions and eliminate the bad solutions in a population while keeping the population size constant.
- ▶ “Selects the best, discards the rest”

Functions of Selection operator

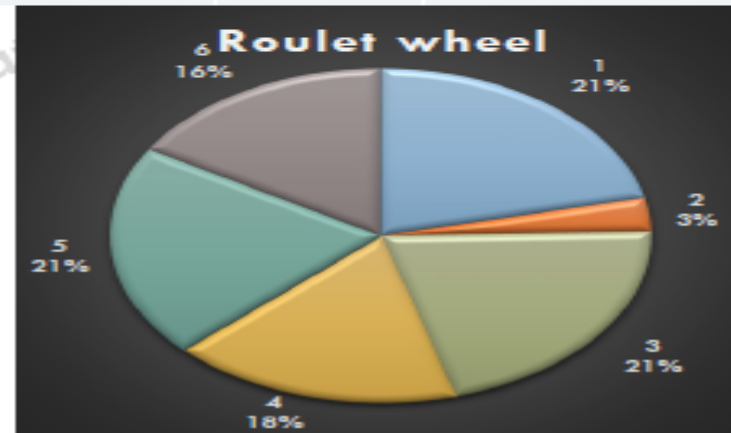
- ❑ Identify the good solutions in a population
- ❑ Make multiple copies of the good solutions
- ❑ Eliminate bad solutions from the population so that multiple copies of good solutions can be placed in the population
- ❑ There are different techniques to implement selection in Genetic Algorithms:
Tournament selection. Roulette wheel selection Proportionate selection Rank selection Steady state selection
- ❑ A fitness value can be assigned to evaluate the solutions
- ❑ A fitness function value quantifies the optimality of a solution.
- ❑ The value is used to rank a particular solution against all the other solutions
- ❑ A fitness value is assigned to each solution depending on how close it is actually to the optimal solution of the problem

Roulette wheel and proportionate selection

Parents are selected according to their fitness values

The better chromosomes have more chances to be selected

| Chrom # | Fitness | % of RW | EC | AC |
|---------|---------|---------|------|----|
| 1 | 50 | 26.88 | 1.61 | 2 |
| 2 | 6 | 3.47 | 0.19 | 0 |
| 3 | 36 | 20.81 | 1.16 | 1 |
| 4 | 30 | 17.34 | 0.97 | 1 |
| 5 | 36 | 20.81 | 1.16 | 1 |
| 6 | 28 | 16.18 | 0.90 | 1 |
| | 186 | 100.00 | 6 | 6 |



November 2013

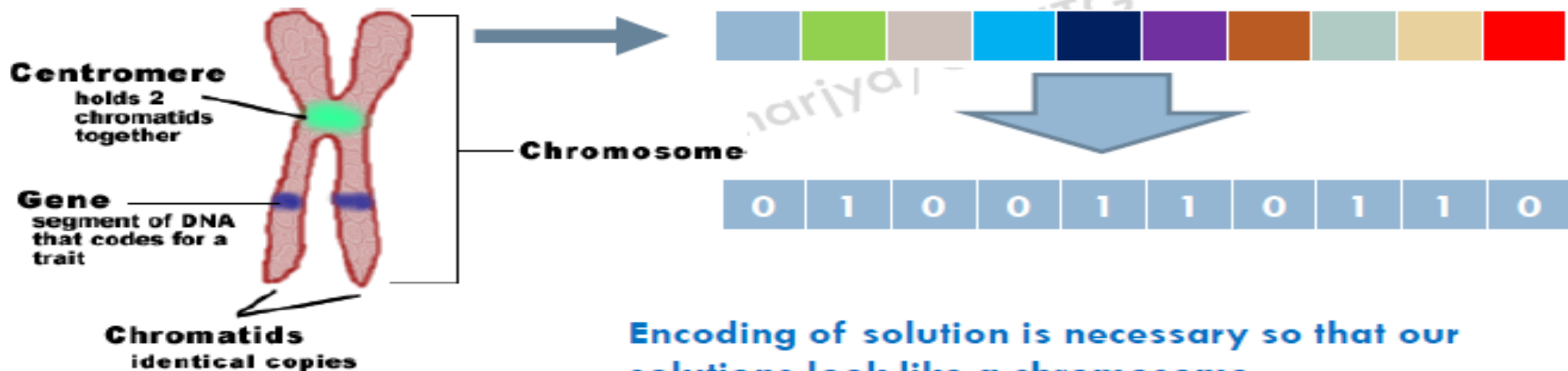
Tournament selection

- ▶ In tournament selection several tournaments are played among a few individuals.
- ▶ The individuals are chosen at random from the population.
- ▶ The winner of each tournament is selected for next generation.
- ▶ Selection pressure can be adjusted by changing the tournament size.
- ▶ Weak individuals have a smaller chance to be selected if tournament size is large.

How to implement crossover

The crossover operator is used to create new solutions from the existing solutions available in the mating pool after applying selection operator.

This operator exchanges the gene information between the solutions in the mating pool.



Encoding

- ▶ The process of representing a 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.

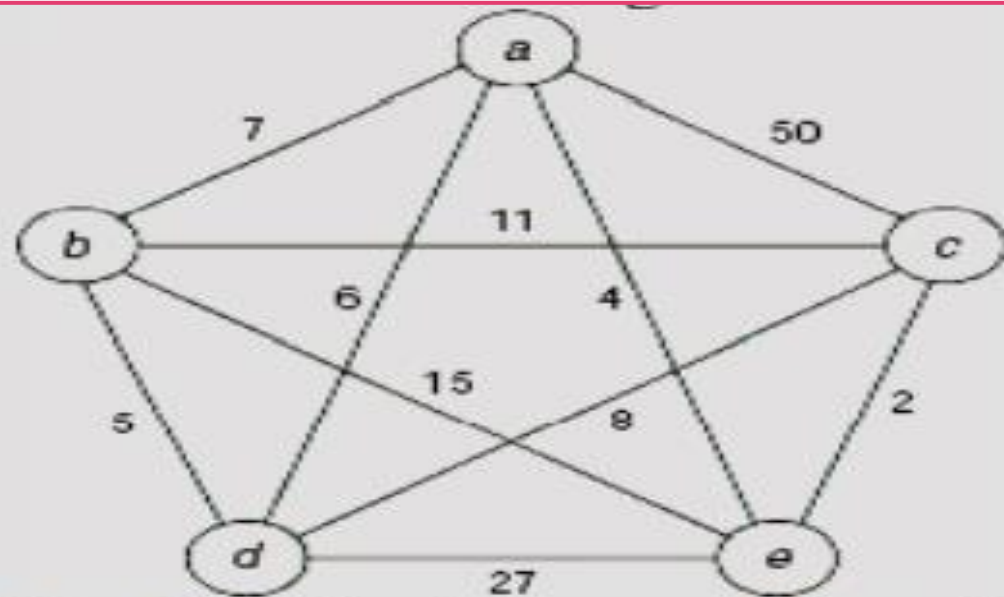
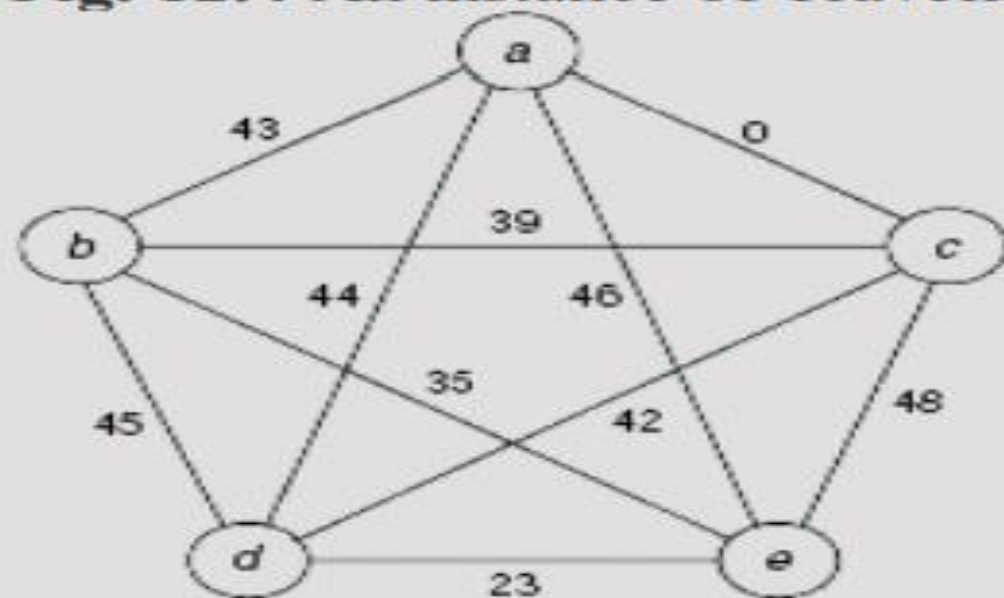
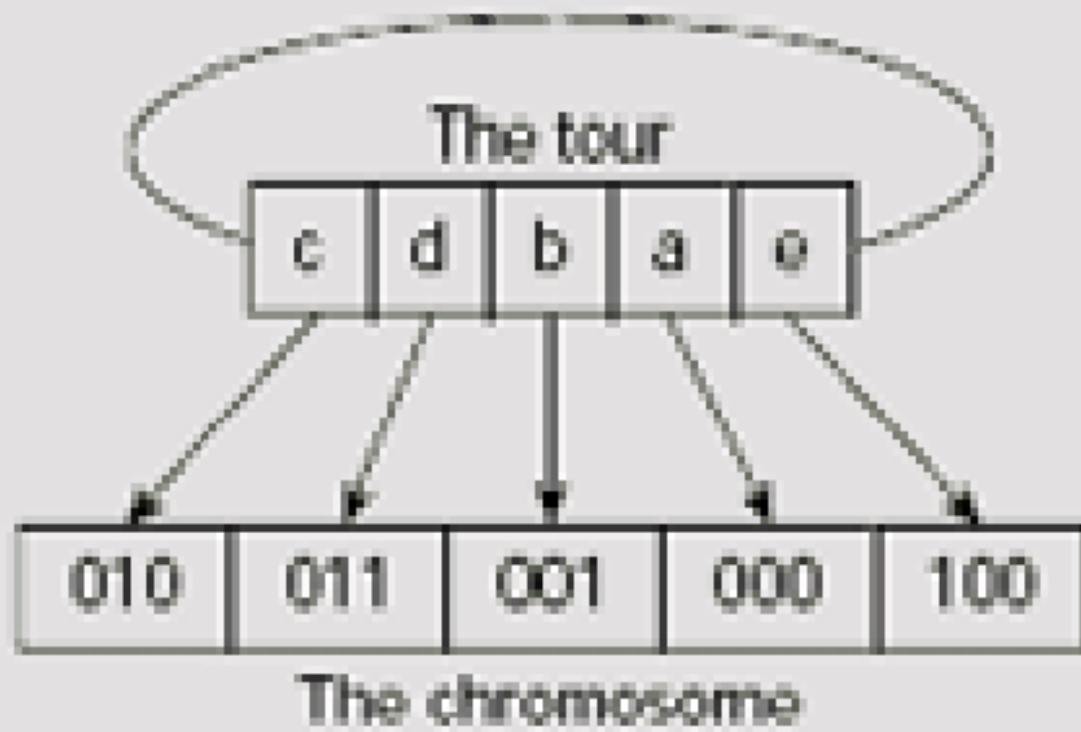


Fig. 12.4 An instance of Travelling Salesperson I



| # | Node | 3-bit Code |
|---|--------|------------|
| 0 | a | 000 |
| 1 | b | 001 |
| 2 | c | 010 |
| 3 | d | 011 |
| 4 | e | 100 |
| 5 | | 101 |
| 6 | Unused | 110 |
| 7 | | 111 |



$$\begin{aligned}\text{Reward} &= 42 + 45 + 43 \\ &\quad + 46 + 48 = 224\end{aligned}$$

chromosome $ch = 101\ 011\ 001\ 110\ 001$ will be interpreted as the tour $a \rightarrow d \rightarrow b \rightarrow c \rightarrow e$.

Table 12.1 Randomly Generated Initial Population

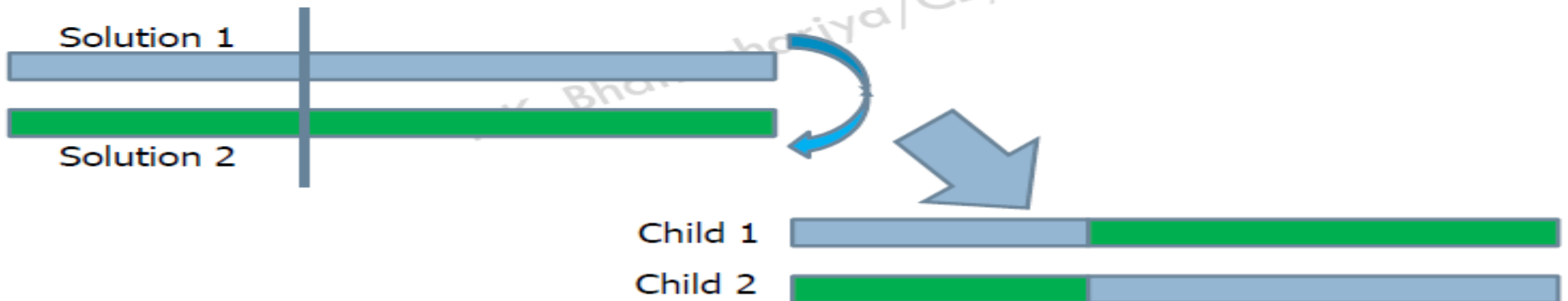
| # | Chromosome | Tour | Fitness |
|----|---------------------|-----------|---------|
| 1 | 011 101 111 011 001 | d-e-a-b-c | 193 |
| 2 | 010 100 001 101 110 | c-e-b-d-a | 172 |
| 3 | 100 100 001 111 010 | e-a-b-c-d | 193 |
| 4 | 011 100 110 001 101 | d-e-a-b-c | 193 |
| 5 | 110 011 101 110 001 | a-d-e-b-c | 141 |
| 6 | 010 100 010 011 011 | c-e-d-a-b | 197 |
| 7 | 100 100 011 111 110 | e-a-d-b-c | 222 |
| 8 | 010 101 011 011 001 | c-d-e-a-b | 193 |
| 9 | 100 110 101 011 110 | e-a-b-d-c | 224 |
| 10 | 111 011 101 100 100 | a-d-e-b-c | 141 |

Crossover operator

The most popular crossover selects any two solutions strings randomly from the mating pool and some portion of the strings is exchanged between the strings.

The selection point is selected randomly.

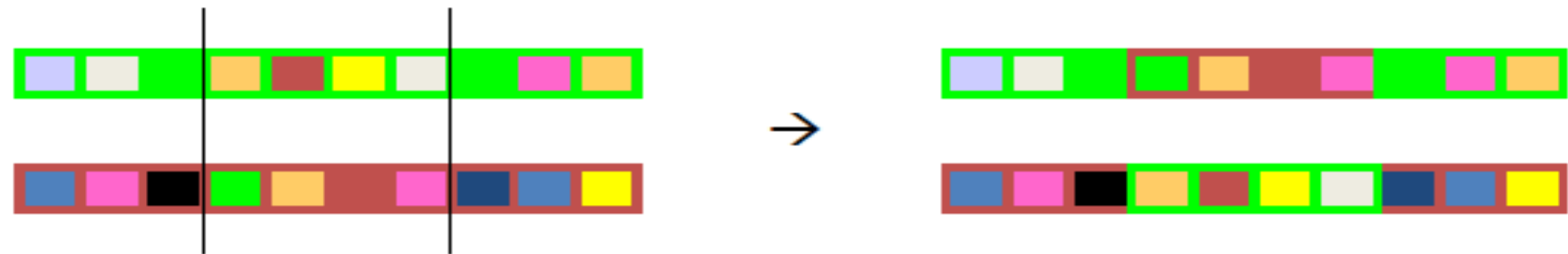
A probability of crossover is also introduced in order to give freedom to an individual solution string to determine whether the solution would go for crossover or not.

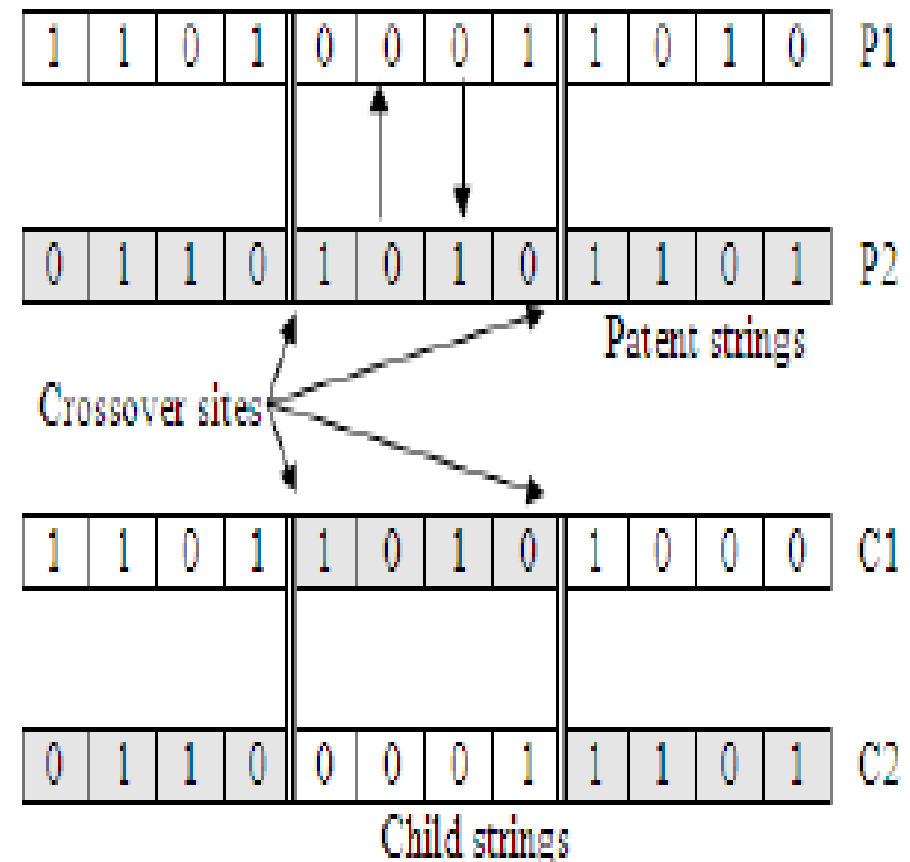
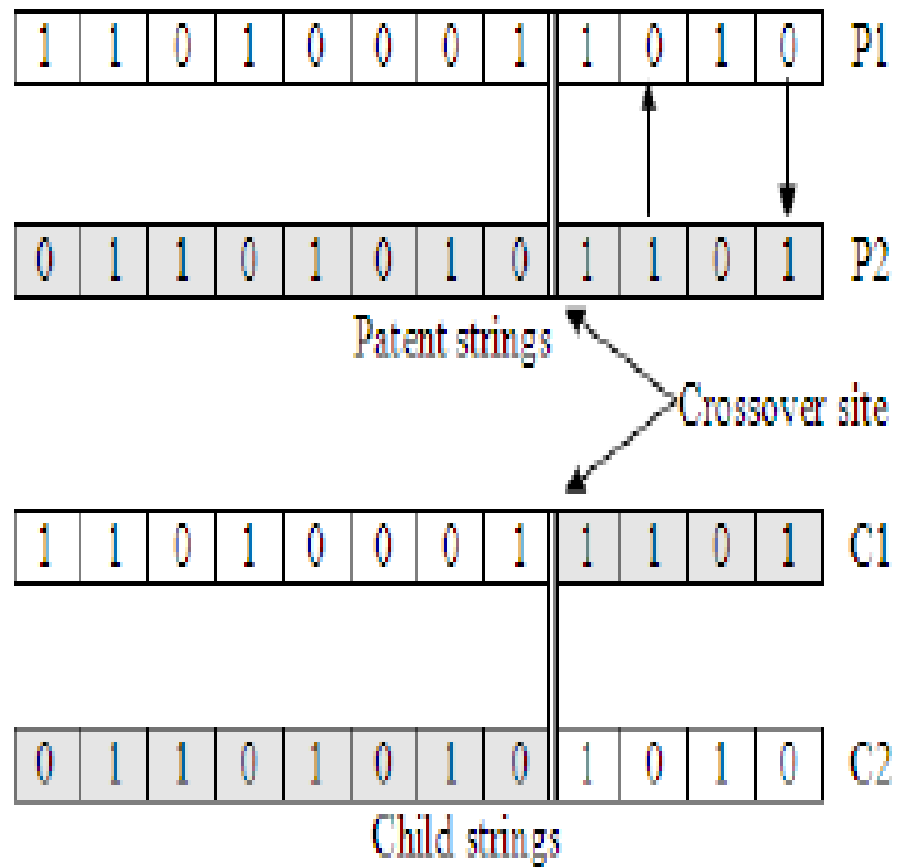


- Single point crossover



- Two point crossover (Multi point crossover)





Mutation operator

Mutation is the occasional introduction of new features in to the solution strings of the population pool to maintain diversity in the population.

Though crossover has the main responsibility to search for the optimal solution, mutation is also used for this purpose.



Before mutation



After mutation

Mutation operator changes a 1 to 0 or vice versa, with a mutation probability of .

The mutation probability is generally kept low for steady convergence.

A high value of mutation probability would search here and there like a random search technique.

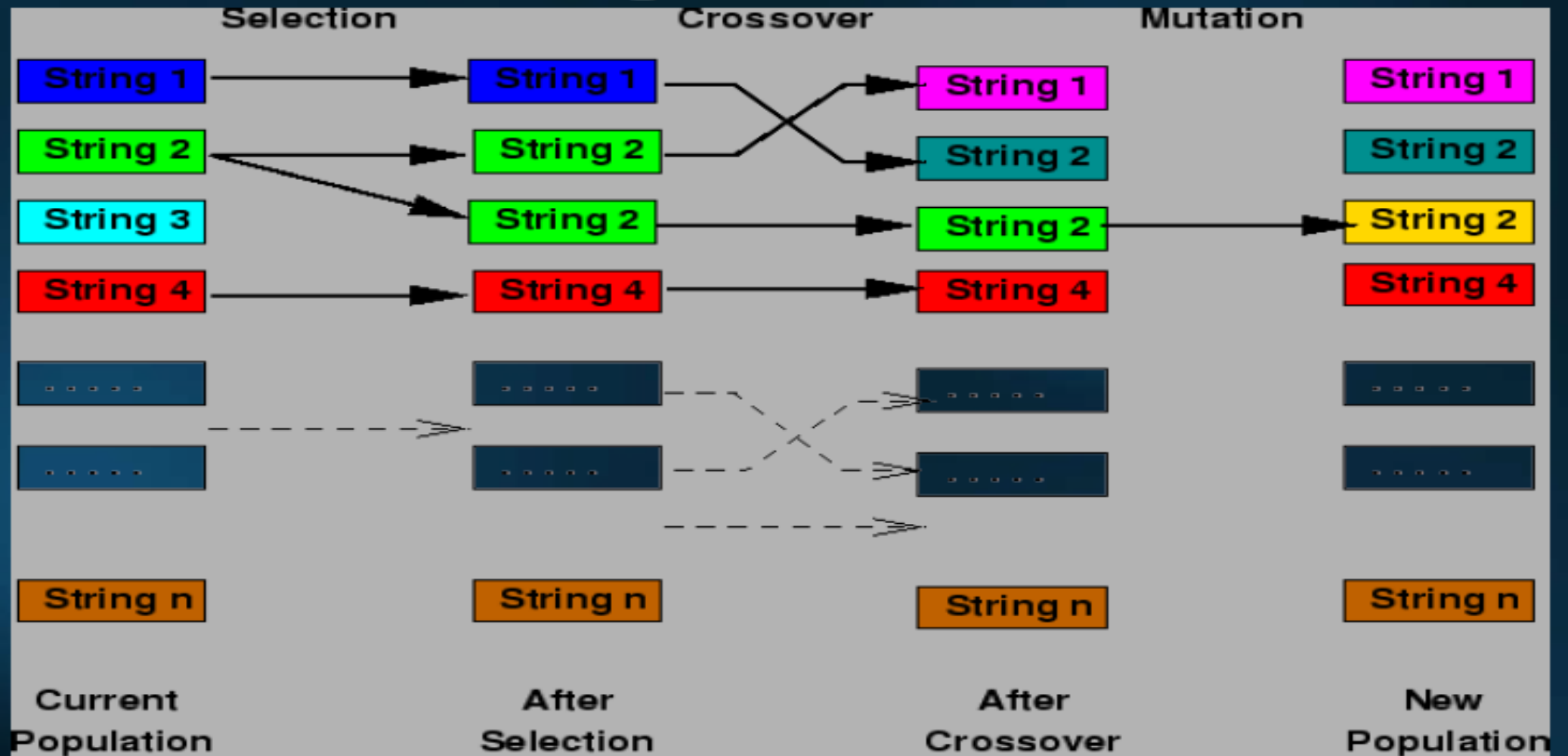
Elitism

Crossover and mutation may destroy the best solution of the population pool

Elitism is the preservation of few best solutions of the population pool

Elitism is defined in percentage or in number

Working Principle of Genetic Algorithm

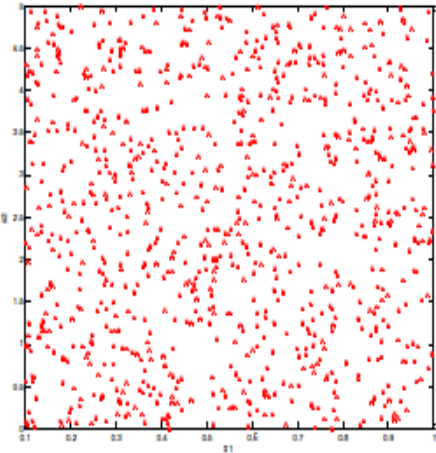


Multi-objective optimization

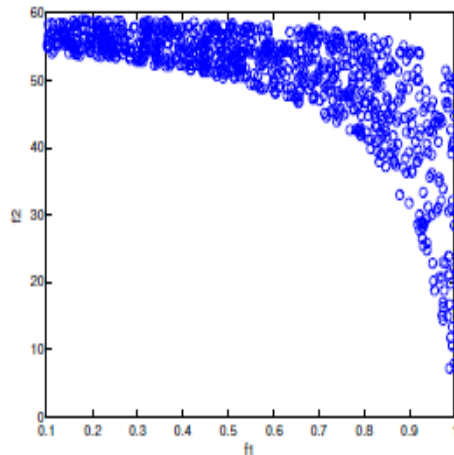
- ❑ More than one objectives
- ❑ Objectives are conflicting in nature
- ❑ Dealing with two search space
- ❑ Decision variable space
- ❑ Objective space
- ❑ Unique mapping between the objectives and often the mapping is non-linear
- ❑ Properties of the two search space are not similar
- ❑ Proximity of two solutions in one search space does not mean a proximity in other search space

Multiple objective genetic algorithm (MOGA)

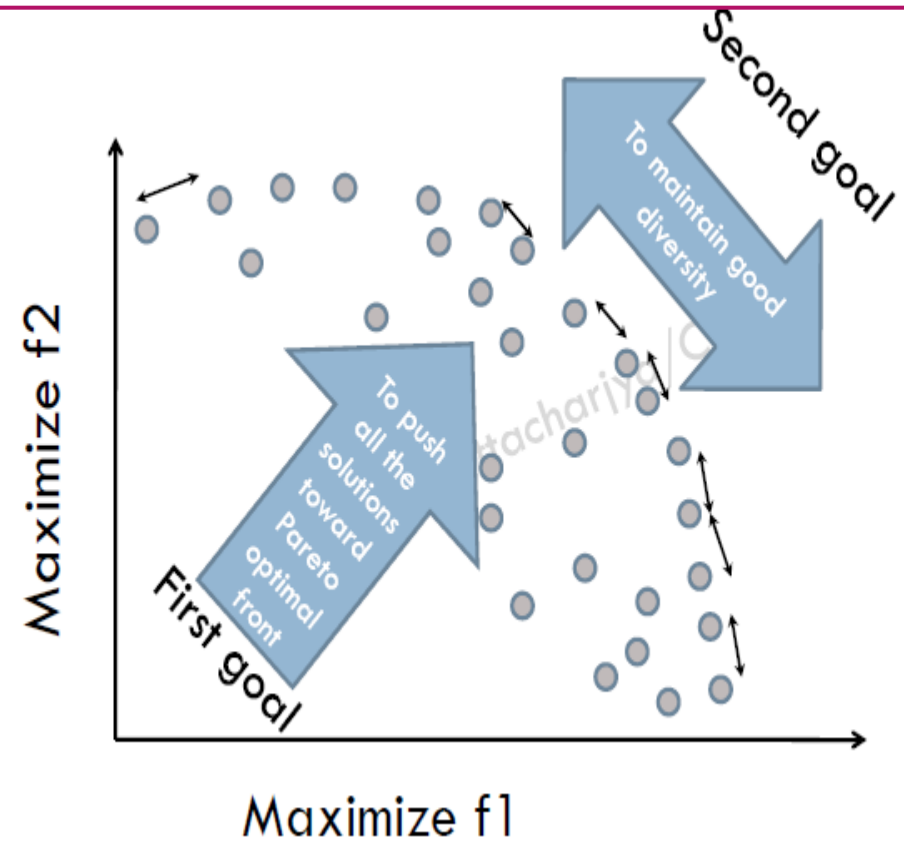
$$\begin{aligned} \text{Maximize } f_1 &= 1.1 - x_1 \\ \text{Maximize } f_2 &= 60 - \frac{1 + x_1}{x_2} \\ \text{Subject to } 0.1 &\leq x_1 \leq 1 \\ &0 \leq x_2 \leq 5 \end{aligned}$$



Solution space



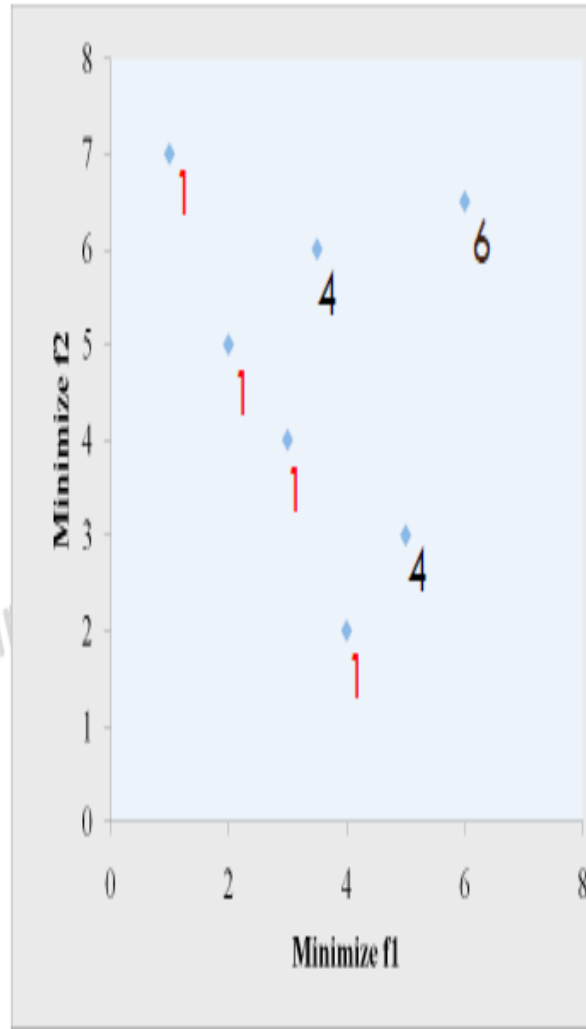
Objective space



Fonseca and Fleming (1993) first introduced multiple objective genetic algorithm (MOGA)

The assigned fitness value based on the non-dominated ranking.

The rank is assigned as $r_i = 1 + n_i$ where r_i is the ranking of the i^{th} solution and n_i is the number of solutions that dominate the solution.



□ Fonseca and Fleming (1993) maintain the diversity among the non-dominated solution using niching among the solution of same rank.

□ The normalize distance was calculated as,

$$d_{i,j} = \sqrt{\sum_{k=1}^M \left(\frac{f_k^i - f_k^j}{f_k^{\max} - f_k^{\min}} \right)^2}$$

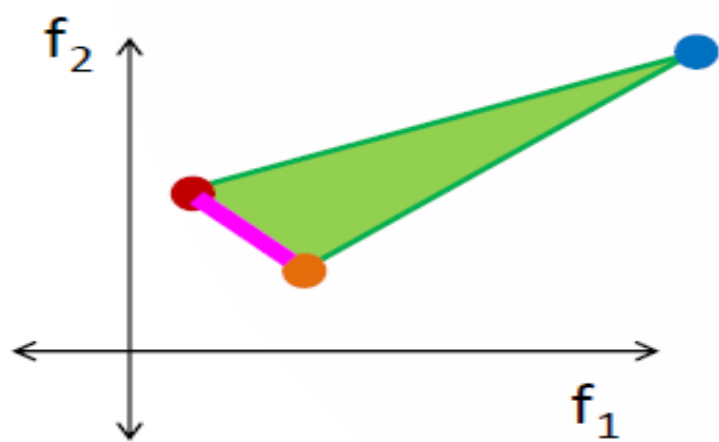
□ The niche count was calculated as,

$$nc_i = \sum_{j=1}^{\mu(r_i)} Sh(d_{ij})$$

Pareto Optimality

In the business example, we were trying to minimize time and cost.

Note that the orange point in criterion space is the lowest value of f_2 (time) and the red point is the lowest value of f_1 (cost). The edge between them is called the *Pareto Front*.



Any point on this front is considered “Pareto optimal”. By moving along the curve, you could minimize cost at the expense of time, or minimize time at the expense of cost, but you can not improve both at once.

The Pareto-optimal Front

The Pareto-optimal front is defined in terms of dominance relation between solution vectors. Assuming all the objective functions to be minimized, a solution vector \bar{x}_1 is said to dominate another solution vector \bar{x}_2 if all the objective functional values of \bar{x}_1 are less than or equal to those of \bar{x}_2 and there is at least one objective function for which \bar{x}_1 has a value which is strictly less than that of \bar{x}_2 .

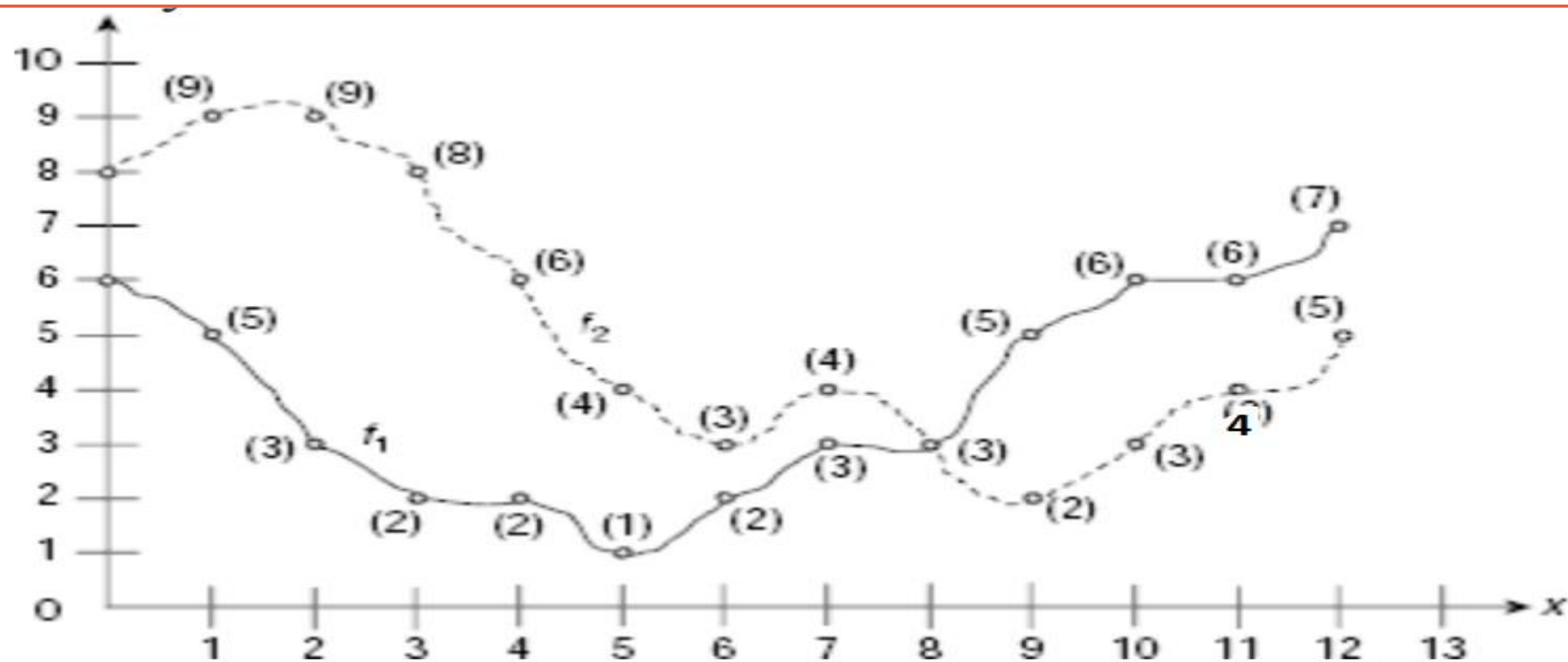
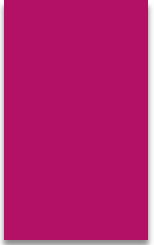


Fig. 17.3 Two objective functions to be minimized

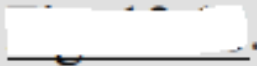




Definition 12.1 (*Dominance relation between solution vectors*) Let \vec{x} denote the vector of objective functions of a MOO problem with K objective functions. Without loss of generality we assume that the functions $f_1(\vec{x}), f_2(\vec{x}), \dots, f_K(\vec{x})$ all need to be minimized. We say that solution \vec{x}_1 dominates another solution \vec{x}_2 , written as $\vec{x}_1 \prec \vec{x}_2$, if and only if

$$f_i(\vec{x}_1) \leq f_i(\vec{x}_2) \quad \forall i \in \{1, 2, \dots, K\}, \text{ and}$$

$$\exists i \in \{1, 2, \dots, K\} \ni f_i(\vec{x}_1) < f_i(\vec{x}_2)$$

Example (Dominance relation between solution vectors)

Let us consider an MOO with two objective functions f_1 and f_2 both to be minimized. For simplicity we assume that the solution vector consists of a single parameter x . The shapes of f_1 and f_2 plotted against x are shown in . The search process produces sample solutions at $x \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}$. It is evident from  that the minima for f_1 and f_2 do not coincide. In fact f_1 attains its minimum value of $f_1(x) = 1$ at $x = 5$, while $f_2(x)$ attains its minimal value of 2 at $x = 9$.  shows positions of the objective function vector $f(x) = (f_1(x), f_2(x))$ corresponding to various values of $x \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}$. Now consider the points $e(3, 3)$ and $g(3, 4)$. Here e dominates g . Similarly g dominates i, j, k, l and m . However, $d(2, 6)$ and $e(3, 3)$ are non-dominating with respect to each other. In fact, the points in the set $\{a, b, c\}$ are mutually non-dominating. Similarly $\{d, e\}$, $\{f, g, h\}$, $\{i, j\}$, and $\{k, l, m\}$ are non-dominating sets of solutions.

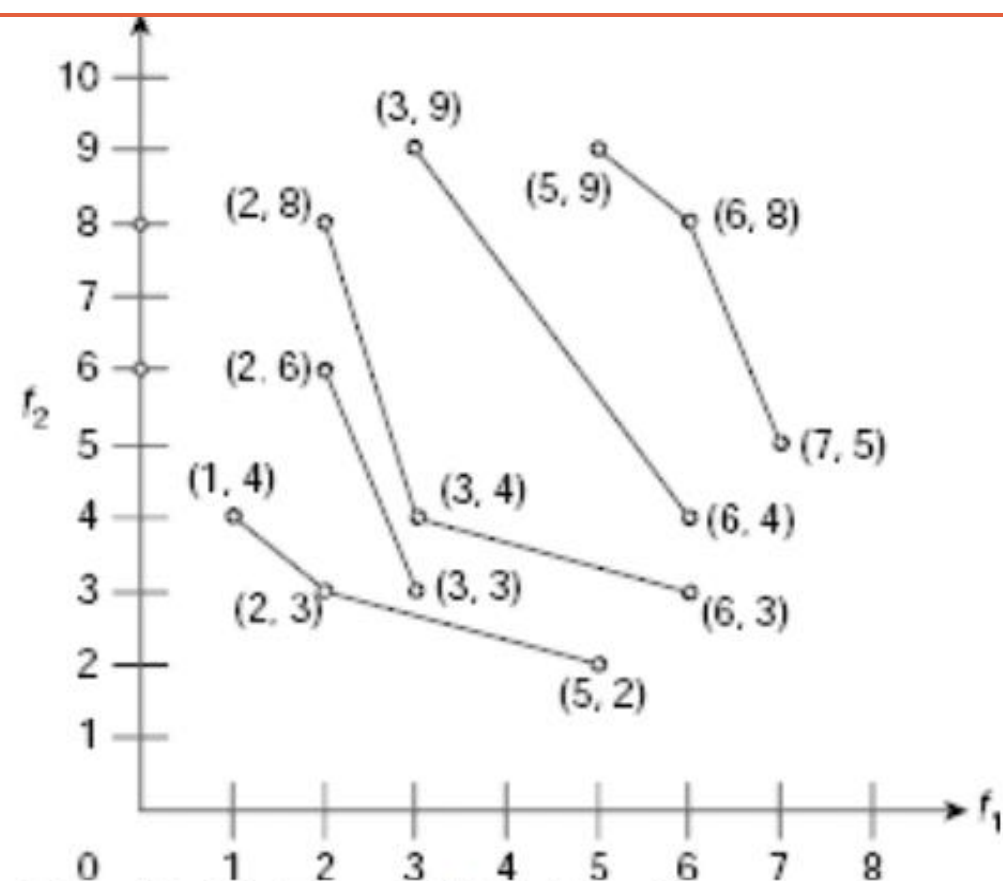


Fig. 12.19 Two-objective function vectors

Definition 12.2 (*Pareto-optimal Front*) Let P be a population of solutions to an MOO problem. The pareto-optimal front, F_{PO} is the set of all non-dominated candidates of P .

$$F_{PO} = \{\vec{x} \mid \neg \exists \vec{y} \ni (f(\vec{y}) \prec f(\vec{x}))\}$$

The pareto-ranking method described above is often improvised to obtain other ranking schemes. For example, Fonseca and Fleming (1993) followed a ranking method that penalizes solutions located in the regions of the objective function space which are dominated, or covered, by densely populated sections of the pareto-fronts. They used the formula

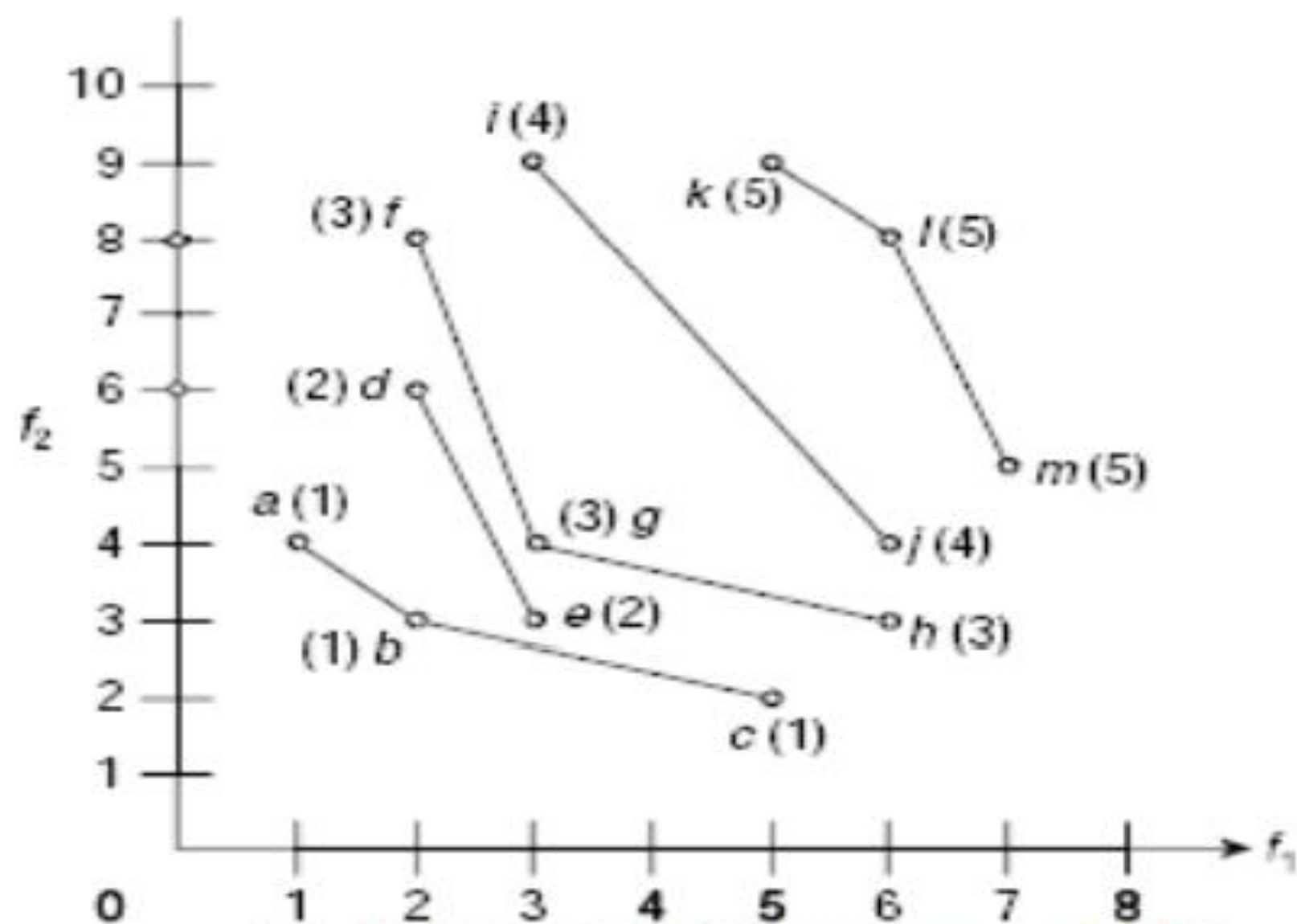
$$r(\vec{x}) = 1 + nd(\vec{x}) \quad (12.3)$$

Here $nd(\vec{x})$ is the number of solutions who dominate \vec{x} .

$$F_{P01} = \{a(1, 4), b(2, 3), c(5, 2)\}, F_{P02} = \{d(2, 6), e(3, 3)\},$$

$$F_{P03} = \{f(2, 8), g(3, 4), h(6, 3)\}, F_{P04} = \{i(3, 9), j(6, 4)\},$$

$$F_{P05} = \{k(5, 9), l(6, 8), m(7, 5)\}$$



Pareto-Ranking as per Goldberg's method

1. Between every pair of solutions $\vec{x}, \vec{y} \in P$, calculate the Euclidean distance $df(\vec{x}, \vec{y})$ in the normalized objective space using the formula

$$df(\vec{x}, \vec{y}) = \sqrt{\sum_{i=1}^K \left(\frac{f_i(\vec{x}) - f_i(\vec{y})}{f_i^{\max} - f_i^{\min}} \right)^2} \quad (12.6)$$

In Formula 12.6, f_i^{\max} and f_i^{\min} are the maximum and the minimum values of the objective function f_i observed so far in the search.

2. For each $\vec{x} \in P$, calculate the niche count $nc(\vec{x})$ using the formula

$$nc(\vec{x}) = \sum_{\substack{r(\vec{x})=r(\vec{y}) \\ df(\vec{x}, \vec{y}) \leq \sigma}} \frac{\sigma - df(\vec{x}, \vec{y})}{\sigma} \quad (12.7)$$

Here σ is the size of the niche. The value of σ is to be supplied by the user.

Fitness of a chromosome is initially calculated with the help of Formula 12.8 given below.

$$f(\vec{x}) = PopSize - \sum_{i=1}^{r(\vec{x})-1} n_i - \frac{n_{r(\vec{x})} - 1}{2} \quad (12.8)$$

In Formula 12.8, n_i is the size of the i^{th} pareto front, and $r(\vec{x})$ is the rank of \vec{x} .

Then *shared* fitness of an individual solution is computed using the niche count as per Formula 12.9.

$$f'(\vec{x}) \leftarrow \frac{f(\vec{x})}{nc(\vec{x})} \quad (12.9)$$

Finally, the *normalized* fitness values are calculated using the shared fitness with the help of Formula 12.10.

$$f''(\vec{x}) = \frac{f'(\vec{x}) \times n_{r(\vec{x})}}{\sum_{\substack{\vec{y} \in P \\ r(\vec{y}) = r(\vec{x})}} f'(\vec{y})} \times f(\vec{x}) \quad (12.10)$$

Procedure Multi-Objective-GA

- Step 1.** Generate the initial population randomly.
- Step 2.** Determine the pareto-optimal fronts $F_{\text{po1}}, F_{\text{po2}}, \dots, F_{\text{poK}}$.
- Step 3.** If stopping criteria is satisfied then Return the pareto-optimal front F_{po1} and **Stop**.
- Step 4.** For each solution $\bar{x} \in P$, evaluate the fitness as follows:
- Step 4.1.** Assign a rank $r(\bar{x})$ using Goldberg's method, or Formula 12.3 (Fonseca and Fleming), or Formula 12.4 (Lu and Yen).
 - Step 4.2.** Compute the basic fitness value using Formula 12.8.
 - Step 4.3.** Compute the shared fitness value using Formula 12.9.
 - Step 4.4.** Compute the normalized fitness value using Formula 12.10.
- Step 5.** Generate the mating pool MP from population P applying appropriate selection operator.
- Step 6.** Apply crossover and mutation operations on the chromosomes of the mating pool to produce the next generation P' of population from MP .
- Step 7.** Replace the old generation of population P by the new generation of population P' .
- Step 8.** Goto Step 2.

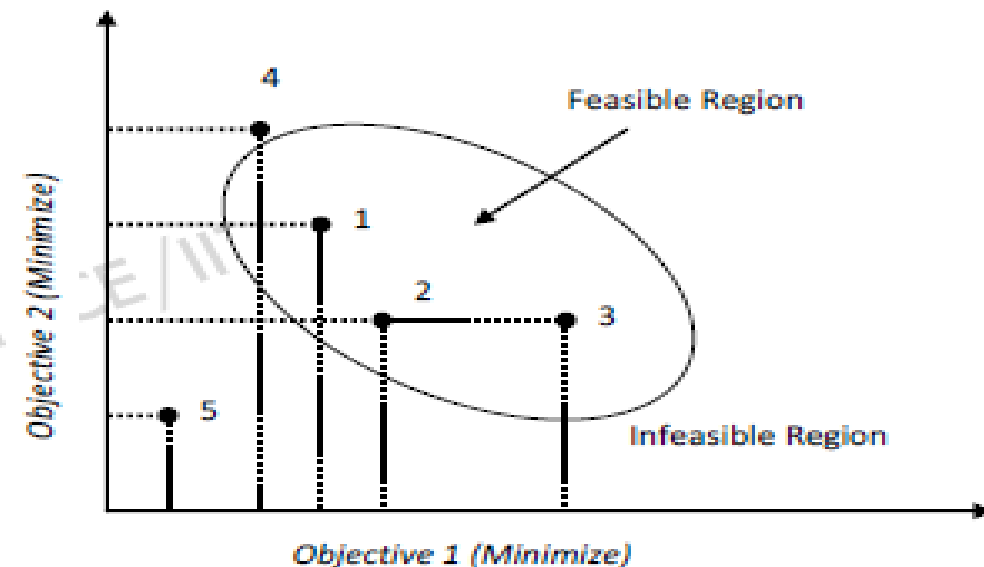
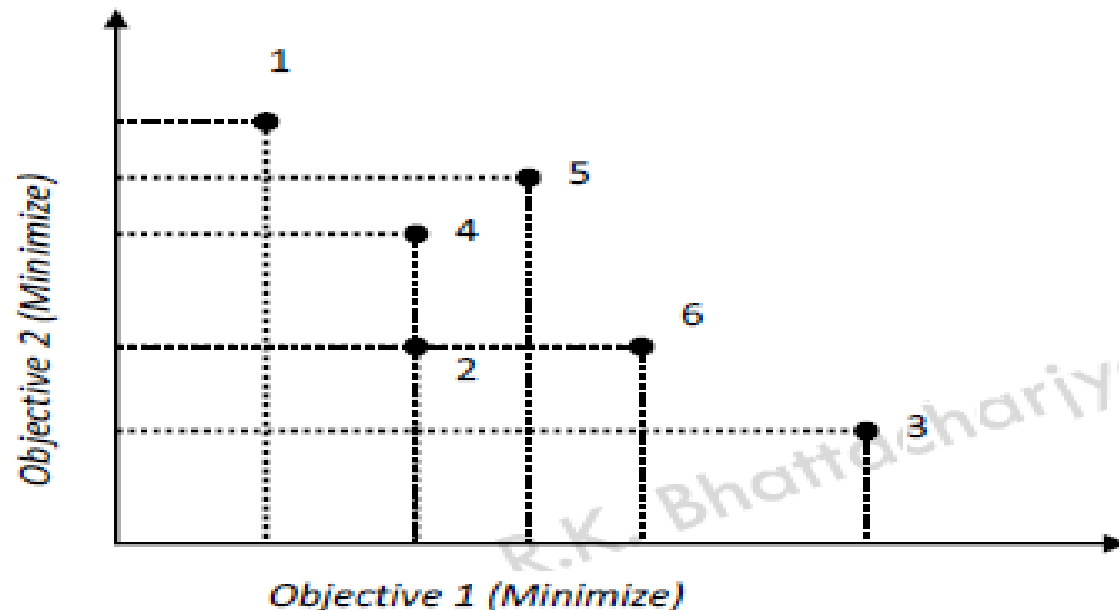
Non-dominated sorting

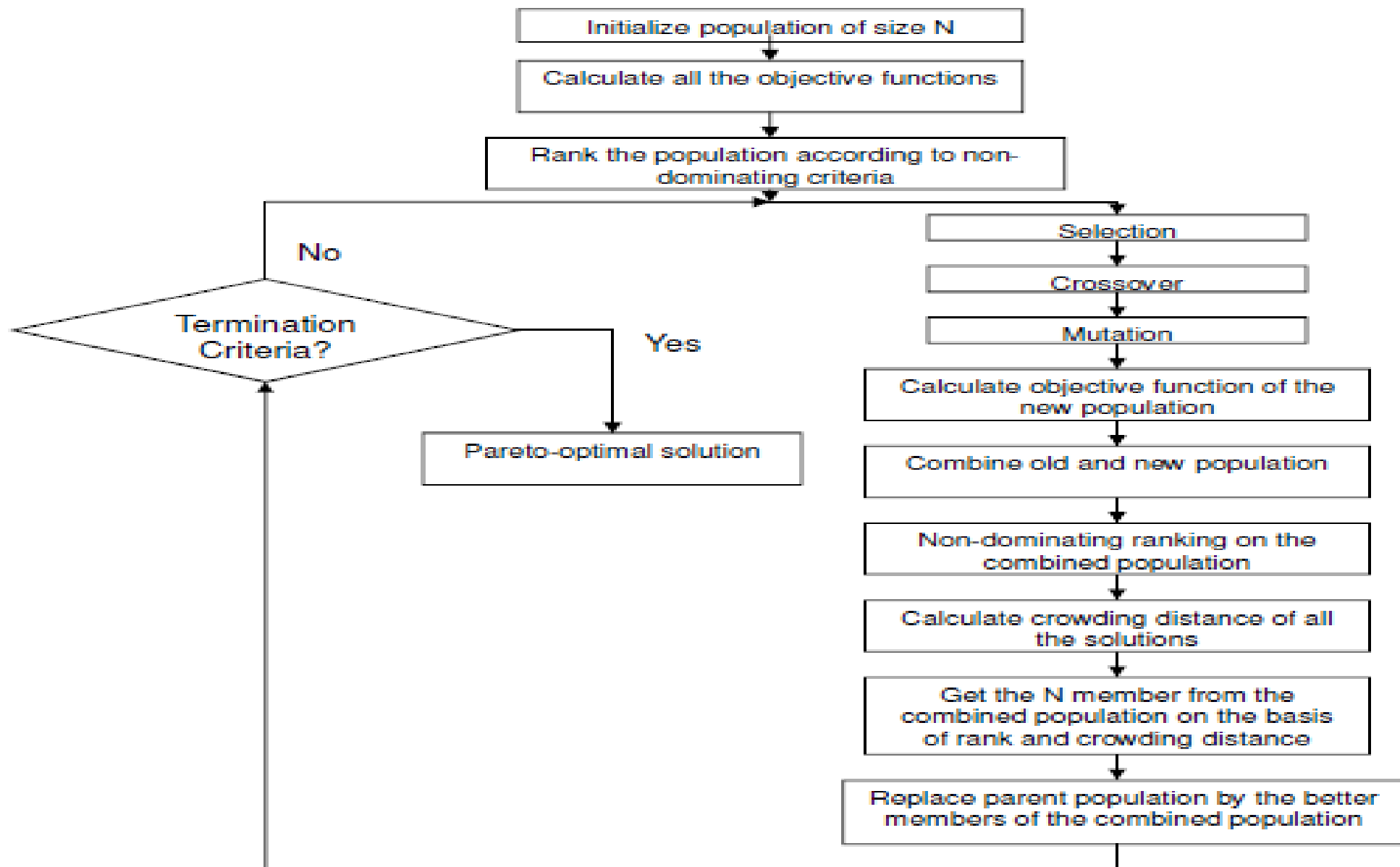
- Srinivas and Deb (1994) proposed NSGA
- The algorithm is based on the non-dominated sorting.
- The spread on the Pareto optimal front is maintained using sharing function


$$d_{i,j} = \sqrt{\sum_{k=1}^{P_1} \left(\frac{x_k^i - x_k^j}{x_k^{\max} - x_k^{\min}} \right)^2}$$

Non-dominated Sorting Genetic Algorithms:
NSGA II is an elitist non-dominated sorting Genetic Algorithm to solve multi-objective optimization problem developed by Prof. K. Deb and his student at IIT Kanpur.

It has been reported that NSGA II can converge to the global Pareto-optimal front and can maintain the diversity of population on the Pareto-optimal front







Genetic Algorithm Applications

| Domains | Application Types |
|----------------------------|--|
| Control | Gas pipeline, pole balancing, missile evasion, pursuit |
| Robotics | Trajectory planning |
| Signal Processing | Filter design |
| Game Playing | Poker, checker, prisoner's dilemma |
| Scheduling | Manufacturing facility, scheduling, resource allocation |
| Design | Semiconductor layout, aircraft design, keyboard configuration, communication networks |
| Combinatorial Optimisation | Set covering, travelling salesmen, routing, bin packing, graph coloring and partitioning |

Closure