1. Introduction

Making proper decision is the crucial component in every sphere of life from the ancient world to the modern civilization. Maintaining the notion of change in the nature of the world, the dependency in making judgement has gradually transited from primitive practises to logical reasoning based mathematical foundations.

Decision making is the process of identifying appropriate action from a collection of actions employable in a particular situation under some environmental stipulation. The nature of the real-world problem, closely associated with human needs and desires, determines the framework of decision problems. The problems of human behaviour and stability of wants were well documented by Edwards [126] and Georgescu-Roegen [158] in 1954. Luce and Raiffa [301] provided critical and systematic explanation of that concept in 1957.

In the context of decision making, mathematical programming (MP) is a strong area which made its debut in early seventeenth century. However, interest in programming problems didn’t sustain due to non-availability of mathematical framework.

Von Neumann [344] established strong mathematical foundation on programming problems in 1932 (published in 1937). The interest in this area was propagated among the researchers mainly in the field of Economics and Mathematical Sciences. Kontorovitch [240] established the potential use of MP to practical problems in 1939. The major contribution in this area of investigation was made by researchers of Operations Research and Management Science during ’50s to ’60s of the previous century. The recent development in the field of MP has been well documented in the book of Kumar and Bonnett [269] in 1991.

The area of MP mainly deals with the problems involving very large numbers of variables and constraints. Due to the high dimensionality of the problems reflecting increasing complexity of the modern day society, it becomes difficult to obtain optimal solutions for such large scale programming problems [423]. Most of the large scale programming problems arising in application almost always have a special feature. These are investigated in the field of Optimization.
1.1 Decision Making in Deterministic Environment

Optimization refers to finding one or more feasible solutions which corresponds to extreme values of one or more objectives [104, 105]. Finding the optimal solution for real-world problem is termed as decision making in the context of the problem. When the objectives, decision variables and associated constraints of the optimization problems are all deterministic (crisp in nature), finding the optimal solution is termed as decision making in deterministic environment. The need for finding such optimal solutions in a problem comes mostly from the extreme purpose of either designing a solution for minimum possible cost, or for maximum possible reliability, or others. Because of such extreme properties of optimal solutions, optimization methods are of great importance in practice, particularly in engineering design, scientific experiments and business decision-making.

In the context of optimization, there exists a search space \( V \) and a function \( g : V \rightarrow R, \)

\[
\begin{align*}
\text{where the problem is to find } & \arg \min_{v \in V} g, \quad \arg \max_{v \in V} g \\
\text{Here } v & \text{ is a vector of decision variables and } g \text{ is a function.}
\end{align*}
\]

Although specified here in an abstract way, this is nonetheless a problem with huge number of real-world applications.

Now, in the decision making horizon, the decision analysts are faced with two types of decision problems: single objective and multiobjective decision problems. When an optimization problem modelling a physical system involves only one objective function the task of finding the solution is called single objective optimization. In the second half previous century, most effort has been made in this field to understand, develop and apply single objective optimization methods.

When an optimization problem involves more than one objective function, the task of finding one or more optimum solutions is known as multiobjective optimization. In the parlance of management, such search and optimization problems are known as multiobjective decision making (MODM) [245, 246]. Since MODM involves multiple objectives, it is intuitive to realize that single objective decision making (SODM) is a degenerate case of MODM. Actually MODM is an extension and very useful generalization of more traditional SODM.
1.1.1 Single Objective Decision Making: A Brief Review

The mathematical model and its solution procedure for programming problems in the context of optimum allocation of resources were introduced in 1941 by Hitchcock [191]. Kantorovitch [241] first provided the Mathematical formulation of same type of problems in 1942. This was followed by the development of standard mathematical model for programming problems and a constructive solution approach, known as simplex method (published in the Monograph No.13 of the historic Cowels Commission Conference on "Linear Programming", University of Chicago, 1949, edited by Koopmans [260] in 1951) by G.B. Dantzig in 1947.

During 1950s, the methodological development of single-objective programming problems and their applications to practical problems were deeply studied by the pioneer researchers in the field during 1950s, and major contributions were made by Charnes et al. [79], Dantzig [94], Wagner [494], Fabin [131], Gass [149], Reinfeld and Vogel [403], and others.

Two different categories in the classification of the single-objective programming and planning environment are: Linear Programming (LP) and Nonlinear Programming (NLP).

Initially, the modeling aspects of MP were concerned mainly with LP. Then, the interest in NLP was developed in the context of modeling with nonlinear mathematical relationships in same economic problems.

The general model of a single-objective programming problem can be expressed as:

Find $X(x_1, x_2, ..., x_n)$ so as to

Optimize (Maximize / Minimize) $Z(X)$

Subject to $f_i(X) = \begin{cases} \geq & \text{if} \geq \\ = & \text{if} = \\ \leq & \text{if} \leq \end{cases} b_i, \quad i = 1, 2, \ldots, m.$ \quad (1.1)

$X \geq 0$

where $X$ is a vector of $n$ decision variables;

$f_i(X)$ is the $i$-th structural constraint;

$b_i$ is the right hand side value of the $i$-th structural constraint.
1.1.1.1 Linear Programming

LP is an integral and perhaps the most devoted part of MP in the broad area of studies of operations research and management sciences.

Kuhn and Tucker [267] first edited the LP problems (LPPs), in 1956. Frisch [143] discussed the multiplex method for solving LPPs in 1958. A comprehensive survey on the early development of the field of LP was presented by Riley and Gass [404] in the same year.

The extensive study in this field and methodological developments for different practical LPPs were made by Hadley [170], Dantzig [96], Hillier and Lieberman [190] and Beale [33]. Myers [339] provided a dual-simplex implementation of a constraint selection algorithm for LP.

Dantzig [96] in his monographs and Ignizio [216], prepared the literature on LP methods and its application to practical problems in the recent past.

Integer Programming (IP) and Dynamic Programming (DP) are two main study areas of LP. IP was first proposed by Dantzig [95]. The methodological developments in the field of IP was made by Gomory [165], Glover [161], Balinski [22] Geoffrion [157], Garfinkel and Nemhauser [148], Zionts [530], Smith and Pickard [457] and others. Taha [472] and Wolsey [502] presented dedicated book in IP in the past. The different aspects of IP have been well covered by Milano [330] in the recent past.

DP is concerned with multi-stage decision process in which a large problem is decomposed into a finite number of sub-problems, of which each contains a subset of the decision variables involved in the programming environment.

DP is based on Bellman’s “Principle of Optimality Theory” [35]. However, the seed of DP was inherent in the research work of Wald [491] on sequential decision theory. Bellman and Karush [36] presented a bibliography on the early development of DP in 1964. A comprehensive survey on DP has been made by Thomas and DaCosta [481] in 1979.
1.1.1.2 Nonlinear Programming

NLP is concerned with a class of MP problems in which the objective and / or environmental constraints are nonlinear in nature.

Kuhn-Tucker [267] research work on vector maximization problems in 1956 is a pioneer work that created interest among researchers in the field of NLP. Kuhn-Tucker [268] also proposed different aspects of NLP in their successive work in the same year.

The early part of the development of the field of NLP was extensively surveyed by Dorn [115] in 1963. The field of NLP is well covered in the text books prepared by Hadley [171], Mangasarian [310], Zangwill [519], Luenberger [302], Avriel [15], Bazaraa and Shetty [32], Gill et al. [159] and others. Ruszczynski [419] provided an extensive study in the field of NLP in 2006.

Fractional programming is an important area of study of MP problems, in the field of NLP.

The fractional programming deals with the optimization problems in which the objective functions are fractional in form. Fractional programming can be classified into two different categories, viz., linear fractional programming and nonlinear fractional programming, based on the linear and nonlinear nature of the numerator and /or denominator of the fractional objective function.

Isbell and Marlow [223] extensively discussed the concept of fractional programming in their work in 1956. The standard model of fractional programming was introduced by Charnes and Cooper [75] in which they proposed a constructive solution procedure. The fields of single-objective fractional programming and its different aspects have been explored by Bitran and Novaes [50], Mangasarian [311], and others.


1.1.2 Multiobjective Decision Making: A Brief Literature Survey

The methods and procedures to solve MODM problems are referred to as Multiobjective Programming (MOP). A set of objectives are involved in the decision environment in MODM.

In the decision making context, it is found that the objectives are incommensurable and they often conflict each other for optimizing them in the decision environment. It is one of the fastest growing fields of study in the area of multi criteria decision making (MCDM) in MP.

The term criteria are generally referred to the standards or rules of judgment for human needs and desires. In the framework of MCDM, a criterion often means evaluation schemes on the basis of which the effectiveness of performance of a decision system is measured.

The concept of making decisions with multiple criteria was introduced by Pareto [387] in 1906. The mathematical foundation of MCDM was presented by Koopmans [260] in 1951. The field of MCDM was studied extensively by the pioneer researchers [3, 4, 13, 51, 126, 217, 258, 259, 331, 394, 462] during mid 1950s to 1960s.

The extensive study on MCDM was made during the 1970s to 1980s of the past century. The significant methodological developments of MCDM were well documented in the books prepared by Cochrane and Zeleny [85], Hwang and Yoon [200], Nijkamp and Spronk [345], Roy and Vincke [416], Zeleny [520], Yu [514], Hwang and Lin [198], Tabucanon [470], Romero and Rehaman [410]. Further exploration of the field of MCDM has been surveyed by Bana e Costa and Vincke [25] and Stewart [466] during the last decade of the past century.

The current state-of-the-art of MCDM has been presented by Ballestero and Romero [23] in 1998.

Ehrgott and Gandibleux [127] presented a comprehensive survey on MCDM in the recent past.

Figueira et al. [133] presented a book on the state of the art surveys on MCDM in 2005. The potential use of MCDM analysis to natural resource management has been well documented in the book prepared by Hearth and Prato [186] in the recent past.
In the area of MCDM, decision making with multiplicity of objectives was introduced by Charnes and Cooper [72] in 1957. During 1960s, the field was studied by Charnes and Cooper [74], Bod [53], and Briskin [58].

A text book mainly concerned with MODM was then prepared by Johnsen [235] in 1968.

The significant methodological development of MOP in the field of MODM and applications to real life problems were made during the 1970s due to the pioneering contributions of the active researchers [130, 246, 415] in the field and widely circulated in the literature. The extensive studies in the field have been well documented in the books prepared by Haimes et al. [172], Wilhelm [501], Cohon [86], and Hemming [189] in the seventies of the past century. Hwang and Masud provided a comprehensive survey of the state of the art in MODM in their handbook [199].

The methodological extensions and refinements of MODM have been made in the books prepared by Goicoechea et al. [162], French et al. [142] and Sawaragi et al. [436] in the eighties of the past century.

The methodological aspect of a special class of MOP problems (MOPPs) has been discussed by Lewandowski et al. [283] in 1991. In 1999, Miettinen [329] has provided a book on comprehensive study on the nonlinear MODM problems. The field of nonlinear MOP has been further studied by Abo-Sinha and Amer [1], ElShafei [129], and others in the very recent past. The different aspects of MOP and planning have also been well documented in the book prepared by Cohon [87] in 2004.

The general MOP model for decision making can be presented as:

Find \( \mathbf{X}(x_1, x_2, \ldots, x_n) \) so as to

\[
\text{Optimize (Maximize / Minimize) } Z_j(\mathbf{X}), \quad j = 1, 2, \ldots, r
\]

Subject to \( f_i(\mathbf{X}) \begin{cases} \geq & \text{subject to } (1.2) \\ = & \text{b}, \\ \leq & \text{i = 1, 2, ..., m.} \\ \end{cases} \)

\( \mathbf{X} \geq 0 \)

where

\( \mathbf{X} \) is a vector of \( n \) decision variables;
$Z_j(X)$ is the $j$-th objective function;

$f_i(X)$ is the $i$-th structural constraint;

$b_i$ is the right hand side value of the $i$-th structural constraint.

Among all the MOP approaches developed so far in the field of MODM, the goal programming (GP) has appeared as a robust and promising tool for multiobjective decision analysis.

During the last 35 years, the study on GP was made extensively and widely circulated in the literature.

1.1.2.1 Goal Programming: A Brief Historical Development

GP is concerned with MODM problems in which the objectives that generally conflict each other to achieve their aspired target levels in the decision making environment.

The work of Koopmans [260] in the early 1950s contains the seeds of GP in the context of resource allocation planning. The conceptual framework of GP was first introduced by Charnes and Cooper [74] in 1961. A book on the early works of GP was presented by Ijiri [217] in 1965.

The significant development in the field of GP was made by Lee [279], Dyer [125], Dauer and Kruger [97], Charnes and Cooper [78], Hannan [175], Ignizio [207, 208], Arthur and Ravindran [14], Kluyver [256], and others during the 1970s.


The potential real-world applications of GP have been presented by Neely et al. [343], Hannan [176], Fisk [134], Kwak and Schniederjans [271], Buckley and Hayashi [60], and others.

During 1980s, the field of GP was extensively studied and pioneering contributions were made by Ignizio [209-215], Masud and Hwang [317], Cook [89], Gass [150], Crowder and Sposito [91], and others. The interesting methodological extensions of GP were well documented in the books prepared by Ignizio [214], Schniederjans [441]. The bibliographical works on GP had also been made by Zanakis and Gupta [517].
The application of GP to different real-life problems has been discussed by Chames and Storbeck [71], Zanakis and Smith [518], Dobbins et al. [114], Taylor et al. [479], Dryzan [116], Rehman and Romero [402], among others. The use of GP to practical problems was surveyed by Lin [287], Spronk [460] and Romero [406]. The application potential of GP had also been well documented in the book prepared by Romero and Rehman [410].

During the 1990s, the field of GP was explored further by the pioneer researchers: Bryson [59], Ignizio and Cavalier [216], Tamiz et al. [474, 475], Schnierderjan [442], Romero et al. [411], among others.

The literature on the methods of GP had been reviewed thoroughly and critically, classified systematically in the book prepared by Romero [407] in 1991. The new trends and applications of GP have also been discussed by Pant and Shah [386], Caballero et al. [62] among others.

Different aspects of GP formulation have been well documented in the books prepared by Michnik et al. [328] in 2002, Tanino et al. [477] in 2003. The use of GP to bank asset liability management has been presented in the book prepared by Kosmidou and Zopounidis [264] in 2004.

The various aspects of GP have been further studied by Mirrazavi et al. [334], Romero [408] among others in the recent past. In 2006, the application potential of GP to different real-life problems have been presented by Li et al. [286], Bal et al. [21], Leung et al. [282] among others.

However, further exploration of the field of GP is one of the current research problems from the viewpoint of its potential use to different real-life problems in the area of MODM.

1.1.2.2 Goal Programming Formulation: Overview

The formulation of the mathematical model of GP is based on the concept of goal satisficing philosophy, expound by Simon [452] in 1957 in the context of managerial decision-making.

A desired target value termed as aspiration levels (or goal level) is introduced to each of the objectives in GP instead of optimizing the objective function directly.
In general, an objective function is of the form:

\[ \text{Optimize (Maximize / Minimize) } Z(X), \]

Then, in conjunction with aspiration level, an objective function takes either of the forms:

\[ Z(X) \geq b, \]
\[ Z(X) = b \]

or \[ Z(X) \leq b \]

depending on the decision situation, where ‘b’ indicates aspiration level.

The forms of the above inequalities are the same as that of the structural (rigid) constraints in a programming problem.

The objectives with their aspiration levels along with the structural constraints can be presented in a unified form as:

\[ Z_j(X) \begin{cases} \geq b_j, \\ = b_j, \\ \leq b_j \end{cases}, \quad j = 1, 2, \ldots, r. \]

Traditionally, the above inequalities are termed as constraints. But in the literature of GP, (under the satisficing philosophy) the term goal, instead of constraint, is used. i.e., the above set of inequalities is said to be (inflexible) goal.

The conceptual and technical differences between constraint and goal can be defined as follows:

- A Constraint is a (temporarily) fixed requirement, which cannot be violated in any decision situation.
- A Goal is a (temporarily) fixed requirement (specific state), which is to be satisfied as closely as possible in a decision-making environment.

In GP, the worth mentionable criterion is:

- It is not only necessary and sufficient to satisfy a goal as a constraint but it is also necessary to satisfy a goal in the best possible way.

Now, in case of the conventional MP models, either slack or surplus variables (or artificial variables, when needed) are introduced to transform the given model into the standard form. But in GP, the logical variables which are termed as deviational variables (under- and over-deviational variables) are introduced to make the standard form. Since in the decision situation, the aspiration levels of the goals may exactly be
achieved, or either under-or over-achievement may occur, both of the under- and over-deviational variables are introduced to each of the goals. This makes the model flexible to achieve the goal levels in the best possible way in the decision making situation.

The goals with the inclusion of deviational variables appear as:

$$Z_j(X) + d_j^- - d_j^+ = b_j, \quad j = 1, 2, ..., r.$$ 

where $d_j^- (\geq 0)$ and $d_j^+ (\geq 0)$ represent the under- and over-deviational variables, respectively, with $d_j^- \cdot d_j^+ = 0$.

This form of a goal is called the flexible (Soft) goal.

Now in a decision making context, the objective of the decision maker (DM) is to minimize the (unwanted) deviational variables so as to achieve the desired aspiration levels of the goals to the extent possible.

In GP, the term ‘goal achievement function’ instead of the traditional use of ‘objective function’ is used. It seems to be more logical from the viewpoint of the question of achieving the aspiration levels of the goals in the best possible way.

In the Table 1.1, a summary of the conversion of inflexible goals into the flexible goals and deviational variables to be minimized in the achievement function is presented.

<table>
<thead>
<tr>
<th>Inflexible Goal</th>
<th>Flexible Goal (Soft Goal)</th>
<th>Contribution to the Achievement Function (Deviational variable(s) to be minimized)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Z_j(X) \geq b_j$</td>
<td>$Z_j(X) + d_j^- - d_j^+ = b_j$</td>
<td>$d_j^-$ This means that $d_j^-$ is to be minimized</td>
</tr>
<tr>
<td>$Z_j(X) = b_j$</td>
<td>$Z_j(X) + d_j^- - d_j^+ = b_j$</td>
<td>$d_j^- + d_j^+$ This means that both $d_j^-$ and $d_j^+$ are to be minimized</td>
</tr>
<tr>
<td>$Z_j(X) \leq b_j$</td>
<td>$Z_j(X) + d_j^- - d_j^+ = b_j$</td>
<td>$d_j^+$ This means that $d_j^+$ is to be minimized</td>
</tr>
</tbody>
</table>

In the field of GP, the two most prominent approaches for solving multiobjective decision analysis are: Minsum GP and Priority based GP.
1.1.2.3 Minsum Goal Programming

The GP model initially proposed by Charnes and Cooper [74] is actually the minsum GP model and it is the simplest form of GP.

In minsum GP approach, the goals are rank-ordered according to their importance of achieving the desired aspiration levels in the planning context. The (non-negative) numerical weights assigned to the goals actually represent the relative importance of achieving the goals to their respective (desired) target values.

In the solution process of minsum GP, the achievement of a goal to its aspired level to the extent possible with highest weight is considered first and then the achievement of a goal with next higher weight is considered, and so on. The achievement function in minsum GP is to minimize the sum of the weighted (unwanted) deviational variables.

The general form of minsum GP model is:

Find \( X(x_1, x_2, \ldots, x_n) \) so as to

Minimize \( Z = \sum_{j=1}^{r} (w_j^d d_j^- + w_j^d d_j^+) \)

and satisfy \( Z_j(X) + d_j^- - d_j^+ = b_j \)

\( d_j^-, d_j^+ \geq 0 \) with \( d_j^- = 0 \), \( d_j^+ = 0 \), \( j = 1, 2, \ldots, r \). \( (1.3) \)

where

\( Z_j(X) \) is the \( j \)-th goal constraint of the decision vector \( X \);

\( b_j \) is the aspiration level of the \( j \)-th goal;

and \( Z \) represents the achievement function consisting of the weighted deviational variables, where \( d_j^- (\geq 0) \) and \( d_j^+ (\geq 0) \) represent the under- and over-deviational variables of the \( j \)-th goal \( (j = 1, 2, \ldots, r) \) and \( w_j^{-} (\geq 0) \) and \( w_j^{+} (\geq 0) \) are the numerical weights of importance of achieving the goal in the decision situation.

1.1.2.4 Priority based Goal Programming

In the field of GP, the priority based GP is the most prominent and powerful technique for solving decision problems with multiple and conflicting goals in MODM environment.

The priority based GP model was first developed by Ijiri [217] in 1965 and further investigation (revision and extension) on priority based GP model has been made by Ignizio [206, 214], Lee [279], Steuer [462], and others and has been applied to a wide-range (variety) of real-life problems. Now a days, when people talk about GP, they generally refer to priority based GP.

In the priority based GP procedure, the goals are rank ordered according to their priorities for achievement of the respective aspiration levels in the decision making situation. The goals of equal importance are included at the same priority level. Since, the achievement of the desired aspiration levels of all the goals are rare due to limited resources, the differential weights are assigned to the goals at the same priority level according to their relative importance of achieving the respective target values.

In the priority based GP solution process, the goals at the highest priority level are considered first for achievement of their respective aspiration levels according to their relative importance of weights at that priority level. Then, the question of achievement of the goal at the next higher priority level is considered, and so on.

The general priority based GP model is:

Find $X(x_1, x_2, \ldots, x_n)$ so as to

Minimize $Z = [P_1(d^-, d^+), P_2(d^-, d^+), \ldots, P_l(d^-, d^+), \ldots, P_L(d^-, d^+)]$

and satisfy $Z_j(X) + d_j^- - d_j^+ = b_j$

$X \geq 0$

d_j^-, d_j^+ \geq 0$ with $d_j^- = 0; \quad j = 1, 2, \ldots, r.$ \hspace{1cm} (1.4)

where $Z$ is the L priority achievement function and where $P_l(d^-, d^+)$ is of the form:

$$P_l(d^-, d^+) = \sum_{j=1}^{r} (w_{j,l}^+ d_j^- + w_{j,l}^- d_j^+) \quad l = 1, 2, \ldots, L$$
$w_{j'}^+, w_{j'}^- \geq 0$ and $d_{j'}^-, d_{j'}^+ \geq 0$ with $d_{j'}^-, d_{j'}^+ = 0$, \( j = 1, 2, \ldots, r; \ L \leq r \).

where

- $Z_j(X)$ is the $j$-th goal constraint of the decision vector $X$;
- $b_j$ is the aspiration level of the $j$-th goal;
- $P_l (l = 1, 2, \ldots, L; L \leq r)$ is the $l$-th factor assigned to the set of objectives goals that are grouped together in the problem formulation, and the priority factors have the relationship:

\[ P_1 \gg P_2 \gg \ldots \gg P_l \gg \ldots \gg P_L. \]

where $\gg$ implies much greater than, i.e., the goals at the highest priority level ($P_1$) must be achieved to the extent possible before the set of goals at the next priority level ($P_2$) is considered, and so on.

All other notations are analogous to the \textit{minsum} GP model (1.3).

It is to be noted that the \textit{minsum} GP is actually a special case of \textit{priority based} GP where no priority preference is given to the goals, i.e., in \textit{minsum} GP, only the weight structure plays the key role in the decision making process.

A general structure of achievement function for a GP model has been presented by Romero [409]. Some of the important GP approaches for solving different multiobjective decision problems are:

- Interactive GP [125, 214, 453],
- Nonlinear GP [206, 420],
- Integer GP [214, 312],
- Fractional GP [175, 262].

However, GP has been successfully implemented to different real-world problems [271, 280, 313, 333, 343]. The potential use of GP to real-life problems have been surveyed by Lin [287] and Romero [406].

The advantage of using GP approach is that it offers an acceptable compromise or a satisficing solution in the crisp decision making environments.
1.1.3 Hierarchical Decision Making

In hierarchical decision situation, decision makers (DMs) are located at different hierarchical decision levels, each independently controlling a vector of decision variables for optimizing the individual objectives which often conflict each other in the decision making situation. In two successive levels of hierarchy, the DM in the higher level is termed as 'leader' and the DM in the lower level is termed as 'follower'. The decision of the leader is often affected by the reaction of the follower due to his / her dissatisfaction with the decision. As a consequence, decision deadlock often arises and the problem of distribution of proper decision powers to the DMs is encountered in a hierarchical decision making horizon.

The concept of hierarchical decision problem as a field of study in the area of mathematical programming was first suggested by Burton et al. [61] for solving decentralized planning problems of large decision making organizations.

Multilevel programming (MLP) as a special branch of study in the area of hierarchical decision problems, has been studied in the past by Charnes and Storbeck [71], Candler, Fortuny-Amat and McCarl [64], Bard and Falk [29], Anandalingam [9], Pal et al. [353] and others.

A general MLP problem (MLPP) can be presented as:

\[
\begin{align*}
\text{Max } Z_1(X) \\
\text{where, for given } X_1; \ X_2, X_3, \ldots, X_L \text{ solve} \\
\text{Max } Z_2(X) \\
\text{where, for given } X_1 \text{ and } X_2; \ X_3, \ldots, X_L \text{ solve} \\
\vdots \\
\text{Max } Z_L(X), \\
\text{subject to } f(X) \geq 0, \\
X \geq 0.
\end{align*}
\]

where
- \( X = (X_1, X_2, \ldots, X_L) \) is the decision vector of \( n \) decision variables.
- \( Z_\ell(X) \) \( (\ell = 1, 2, \ldots, L) \) represent the objective (performance) functions of DM located at \( \ell \)-th hierarchical level;
- \( X_\ell \in \mathbb{R}^{n_\ell}; \ \ell = 1, 2, \ldots, L \ (\bigcap_{\ell=1}^{L} X_\ell = \emptyset) \) are the decision vectors controlled by the DMs located at \( L \) different hierarchical levels.
Bilevel programming (BLP) is a special case of MLP for solving hierarchical decision problems. The concept of mathematical formulation of BLPPs was introduced separately by Fortuny-Amat et al. [141] and Candler et al. [65]. Thereafter, during '80s, problems of various versions of BLP as well as MLP in general were studied in [26, 28, 43, 44, 63, 84, 487, 497, 505] and the other researchers in the field from the view point of their potential use to different real-world problems. A multiobjective solution technique with post optimality analysis on the objective values based on the three compromise solutions: ideal point, threat point and ideal threat point have been introduced by Wen et al. [498].

A general BLP problem (BLPP) can be presented as:

\[
\begin{align*}
\text{Max } Z_1(X_1) \\
\text{where, for given } X_1, X_2 \text{ solve } \\
\text{Max } Z_2(X_2) \\
\text{subject to } f(X) \geq 0, \\
X \geq 0
\end{align*}
\]  

(1.6)

where

- \( X = (X_1, X_2) \) is the decision vector of \( n \) decision variables.
- \( Z_\ell(X) \) \((\ell = 1, 2)\) represent the objective (performance) functions of DMs located at \( \ell \)-th hierarchical level;
- \( X_\ell \in \mathbb{R}^{n_\ell}; \ell = 1, 2; \bigcap_{\ell=1}^{2} X_\ell = \Phi \) are the decision vectors controlled by the DMs located at \( L \) different hierarchical levels.
- \( f_\ell: \mathbb{R}^n \rightarrow \mathbb{R} \) \((\ell = 1, 2)\), where \( n = n_1 + n_2 \).
- \( f(X) \) is the set of system constraints.

The use of fuzzy approach to solve BLPP as well as MLPP have been investigated by Lai [272], Shih et al.[449], Shih and Lee [450], Pal et al. [353, 371]. The use of BLP in the area of fractional programming has been investigated [309, 382]. BLP as well as MLP are now growing rapidly from the viewpoint of their potential use to several decentralized planning problems.
It is to be noted that the conventional MOP approaches are the robust tool for making decisions. However the main weakness of the conventional MODM methodologies as well as the approaches for hierarchical decision is that the different types of parameters involved with the problems are crisp in nature, i.e., they are precisely defined. Actually, the parameters involved there are fixed at some values in an experimental and/or subjective manner through the DMs' understanding of the nature of the parameters. But, in most of the real-world decision situations, however, the possible values of the parameters are often only ambiguously known to the experts, that is, they are often ill-defined or vague in nature in actual practice.
1.2 Decision Making in Inexact Environment

In a real-world decision situation, inexactness of parameter values occurs due to imprecision in human judgments as well as inherent uncertain in nature of parameters involved with the problem.

In the real-world decision making situations, decision makers (DMs) are frequently faced with the three types of uncertainties - stochastic, fuzzy and interval valuedness, to setting off the parameter values.

In the probabilistically uncertain decision situation, the parameters involved with a problem are random in nature with certain probability distributions. In general probability is the measurement of degree of occurrence of an event.

In the context of making decisions, Bellman and Zadeh [37] quoted: “Much of the decision making in the real world takes place in an environment in which the goals, the constraints and the consequences of possible actions are not known precisely.”

That is, we as human, often find ourselves in a state of uncertainty. This happens due to imprecise or ambiguous nature of human thinking and judgments. Imprecision (neither deterministic nor stochastic) here is meant in the sense of vagueness rather than the knowledge about the value of a parameter as in tolerance analysis in statistics.

Fuzziness is the ambiguity or vagueness that can be found in the definition of a concept or the meaning of a word. Actually, everyone is involved with fuzziness and it is a kind of uncertainty that anyone can understand.

On the other hand, it has been observed in an imprecise decision making environment that setting of imprecise values to the parameters involved with the problem is not possible in the decision situation. In this approach, instead of assigning single parameter values (crisp or fuzzy), interval valued parameters are introduced depending on the decision making environment.

The different approaches addressed in making decisions in different uncertain environments are:

(a) Stochastic programming which is based on theory of probability.

(b) Fuzzy programming which is based on fuzzy set theory.

(c) Interval programming which is based on interval arithmetic.
These three different types of uncertainties are described in the following sections.

1.2.1 Stochastic Programming

The values of model parameters cannot be defined precisely due to inherent randomness of parameter, sometimes in real-world decision making problems. The stochastic programming (SP) based on the probability theory, was used to solve this type of optimization problem.

SP, in the area of uncertain optimization was actually introduced by Dantzig [93] in 1955. The field of SP was extensively studied and significant contributions were made by Kataoka [243], Contini [88], Prekopa [392] and others. The interesting methodological extensions of SP were well documented in the books prepared later by Dempster [109], Kall and Wallace [238], Liu [293] and others.

The application of SP to different real-life problems in the area of agricultural planning has been discussed by Tintner [482], Panne and Popp [489] among others. The application potential of SP had also been well documented in the book prepared by Vajda [488], Birge et al. [45], Marti [315], Gassman [151], and others. The use of SP in optimization problems with multiple criteria was investigated by Sahoo and Biswal [421].

The chance-constrained programming (CCP), as a special area of SP, was initially introduced by Charnes and Cooper [73] in 1959. Thereafter, CCP within the framework of GP (chance-constrained goal programming (CCGP)) was well documented in the works of Charnes and Cooper [76], Miller and Wagner [332], and others. The potential use of CCP in several real-world problems in the area of capital budgeting was extensively investigated by Kewon et al. [247, 248, 249], and De et al. [103].

In some real-world problems, decision making depends on information which is fuzzily as well as probabilistically uncertain. The constructive modeling aspect of programming problems under randomness and fuzziness was first introduced by Luhandjula [303] in 1983. The use of SP in fuzzy environment has been studied extensively in [227, 335, 336]. The methodological aspects studied in the area of stochastic linear programming have been surveyed by Luhandjula [308] in 2006.
The potential use of different real-life problems in the area of fuzzy SP have been investigated by Liu et al. [294], Iskander [226, 228] and others. A fuzzy satisficing method to solve LP problem with random coefficients in the objective function and/or with chance constraints has been developed by Sakawa et al. [424, 425].

Stochastic linear programming (SLP) with multiplicity of criteria in fuzzy environment was investigated by Leclercq [277], Hulsurkar et al. [197], Mohan and Nguyen [337] in the past. A stochastic GP model, concerned with the structure of mean-variance minimization, has been proposed by Ballestero [24] in 2001. Stochastic GP method, for solving water use planning, has been suggested by Bravo and Ganzalez [57] in 2009. The use of CCP and dependant chance programming (DCP) was extensively investigated by Liu [290, 292], Liu and Iwamura [296, 297]. Slowinski and Teghem provided a comparative study between stochastic and fuzzy approaches in [456].

In the field of SP, the two most useful models for solving multiobjective decision problems are:

(a) Chance-Constrained Programming (CCP) [73, 295],
(b) Dependent-Chance Programming (DCP) [290, 291, 299].

The above two models are most effective in dealing with two different types of randomness that may occur in SP problem. In first type, the probability distribution of the random parameters is known (e.g., the random variations in weather in agricultural applications) and the problem is to choose a decision vector $X$ which is optimal in some sense. In other case, the probability distribution of the model parameters are unknown, but only sample observation may be available. Then, unknown population parameters have to be estimated in order to incorporate them into the program to arrive at an optimal decision. The different branches of SP are presented in the following Figure 1.1.
Stochastic Programming

- Chance-constrained Programming (CCP)
- Dependent-chance programming (DCP)

Figure 1.1: Different SP models

It may be pointed out CCP, among different approaches is the most efficient and widely used technique in the field of SP.

Generally, it is not realistic to assume any set of decision satisfying all the constraints under all circumstances. Then, some constraints may be considered as chance, with minimum level of confidence. If some / all of the constraints are taken as probabilistic instead of deterministic, then the problem is called CCP problem.

A multiobjective CCP problem can be stated as:

Maximize: $Z^{(k)}(X) = \sum_{j=1}^{n} c_{j}^{(k)} x_{j}$, \hspace{1cm} k = 1, 2, ..., K

subject to $\Pr(\sum_{j=1}^{n} a_{ij} x_{j} \leq b_{i}) \geq 1 - \alpha_{i}$, \hspace{1cm} i = 1, 2, ..., m

$0 \leq x_{j} \leq x_{j}^{u}$, \hspace{1cm} j = 1, 2, ..., n

$\alpha_{i} \in (0,1), i = 1, 2, ..., m$ \hspace{1cm} (1.7)

where

- $a_{ij}$ and $b_{i}$ are any continuous random variables,
- $\alpha_{i}$ are specified probabilities and
- $x_{j}^{u}$ is the upper bound of the $j$-th decision variable.

'Pr' represents the probability measure of the specified event.
Alternatively, an CCP problem appears as:

\[
\text{Optimize } C^T \mathbf{X} \\
\text{subject to } \Pr [ a_i^T \mathbf{X} \leq b_i ] \geq 1 - \varepsilon_i, \quad i = 1, 2, \ldots, m
\]

\[
\begin{bmatrix}
\geq \\
= \\
\leq
\end{bmatrix}
\]

\[
A \mathbf{X} = b,
\]

\[
\mathbf{X} \geq 0,
\]

where, \( \mathbf{X} \) is decision vector of order \( n \times 1 \), \( a_i \) and \( b_i \) are random vector of order \( n \times 1 \). Here, at least one of \( a_i \) or \( b_i \) is random in nature and \( 1 - \varepsilon_i \) represents the minimum level of probability satisficing measure of the event \( a_i^T \mathbf{X} \leq b_i \) (i = 1, 2, ..., p). \( A \) is the coefficient matrix of order \( m \times n \) and \( b \) is resource vector of order \( m \times 1 \).

Since 1959, CCP has been applied to different real-world problems in [128, 145, 146, 168]. In most of the cases, chance constraints are transformed into their deterministic equivalent form [197] for model simplification and then the prominent deterministic tools are employed in the process of making decision.

Although, some methodological developments in the area of SP have been made in the past, but from the viewpoint of methodological extension as well as implementation of stochastic FP to practical MODM problems is at an early stage. However, the traditional probability cannot express the amount of ambiguity. As such the stochastic approach is not efficient in handling vague or imprecise data. Since the seminal paper on “Fuzzy Sets” by Zadeh [515] in 1965, there exists a convenient and powerful way of modeling vague data without having recourse to stochastic concepts.

Randomness and fuzziness differ in nature and fuzziness expresses much more everyday uncertainty than probability. By considering the imprecise or fuzzy nature of human judgments, the three basic concepts introduced by Bellman and Zadeh [37] are: fuzzy goal, fuzzy constraint and fuzzy decision; and further exploring the application of these concepts to decision making processes under fuzziness.

Actually, fuzzily described different activities involved with a decision problem, are represented by sets. The description and conception of such a set, which is called
subjective category, changes from a crisp set to a set with vague boundaries. The sets with such boundaries are called fuzzy sets.

The fuzzy programming (FP) approach for solving MODM problems is discussed in the following Section 1.2.2.

1.2.2 Fuzzy Programming

The fuzzy set theory (FST) has appeared as the most promising tool for dealing with human centric decision making problems under fuzziness.

The FST expresses fuzziness by means of the concept of the sets in the traditional sense. It provides a strict mathematical framework in which vague conceptual phenomenon can be precisely and rigorously studied. It can also be considered as a modeling language for describing fuzzy criteria, phenomena and fuzzy relations existing in real-life situations.

The FST, however, abandons the excluded-middle law and the law of contradiction of non fuzzy sets; and thus abandons the standard probability calculus altogether, since it gives up the De Morgan relations.

The potential applications of FST [515] to wide variety of MP problems studied by the researchers in different fields have been circulated in the literature. The practical use fuzzy sets in MP, the process analysis, interpretation and establishments of mathematical models are different for different types of decision problems.

Now, some formal definitions in the framework of FST are discussed in the following Section 1.2.2.1

1.2.2.1 Fuzzy Sets and Operations

Fuzzy Sets

The basic conceptual frame of a fuzzy set is that unlike classical set theory, the properties \( E \cap \bar{E} = \emptyset \) (Law of contradiction) and \( E \cup \bar{E} = U \) (Law of excluded middle) doesn't hold good. This means

\[ E \cap \bar{E} \neq \emptyset \text{ and } E \cup \bar{E} \neq U \text{ in a fuzzy set}. \]

(\( \bar{E} \) is the complement of \( E \)).

Let \( X \) be a set (a universal set) of objects denoted generically by \( x \). In FST, a ' ~ ' mark placed above a fuzzy set is used to distinguish between fuzzy and non-fuzzy sets. Then a fuzzy set \( \tilde{E} \) in \( X \) (called subset of \( X \)) is defined by a set of ordered pairs:
\[ \tilde{E} = \{(x, \mu_{\tilde{E}}(x)) \mid x \in X \}, \]
where \( \mu_{\tilde{E}}(x) \) represents the grade of membership of \( x \) in \( \tilde{E} \).

The function \( \mu_{\tilde{E}}(x) \) defines the degree to which the element \( x \) of the set \( X \) is included in \( \tilde{E} \). The degree of inclusion is also called degree of truth (also extent or grade).

The function \( \mu_{\tilde{E}}(x) \) may also be expressed as:

\[ \mu_{\tilde{E}} : X \rightarrow [0, 1]. \]

Thus the value of \( \mu_{\tilde{E}} \) nearer to unity implies the higher grade of membership of \( x \) in \( \tilde{E} \) and vice-versa. When \( \mu_{\tilde{E}} \) contains only two points 0 and 1, then \( \mu_{\tilde{E}} \) is identical to the characteristic function

\[ C_E : X \rightarrow \{0, 1\}, \]

where \( E \) is an ordinary crisp set and hence \( \tilde{E} \) is no longer a fuzzy set.

An ordinary set \( E \) is conventionally expressed as:

\[ E = \{ x \in X \mid C_E(x) = 1 \}, \]

through its characteristic function

\[ C_E(x) = \begin{cases} 1 & \text{if } x \in A, \\ 0 & \text{if } x \not\in A. \end{cases} \]

Thus membership function is an obvious extension of the idea of a characteristic function of an ordinary set, because it takes values not only 0 and 1, but also the values in between them.

Now, some basic operators for fuzzy sets are discussed as follows:

**Operations on Fuzzy Sets**

Several operators for fuzzy sets have been developed and widely circulated in the literature.

Some of the basic operations originally suggested by Zadeh [515] are presented as follows:

- **Union**

  The union of two fuzzy sets \( \tilde{E}_1 \) and \( \tilde{E}_2 \) on \( X \), denoted by \( \tilde{E}_1 \cup \tilde{E}_2 \), is defined by

  \[ \mu_{\tilde{E}_1 \cup \tilde{E}_2} = \max\{\mu_{\tilde{E}_1}(x), \mu_{\tilde{E}_2}(x)\}, \text{ for all } x \in X. \]
- **Intersection**
  The intersection of two fuzzy sets \( E_1 \) and \( E_2 \) on \( X \), denoted by \( E_1 \cap E_2 \), is defined by
  \[
  \mu_{E_1 \cap E_2} = \min\{\mu_{E_1}(x), \mu_{E_2}(x)\}, \quad \text{for all } x \in X.
  \]

- **Complement**
  The complement of \( E \) on \( X \), denoted by \( \overline{E} \), is defined by
  \[
  \mu_{\overline{E}}(x) = 1 - \mu_E(x), \quad \text{for all } x \in X.
  \]
  The above fuzzy set theoretic operations are basically useful in the conceptual framework of a decision-making process under fuzziness.

- **Equality**
  Two fuzzy sets \( E_1 \) and \( E_2 \) on \( X \) are said to be equal, denoted by \( E_1 = E_2 \), if and only if their membership values at each point \( x \in X \) are equal, i.e.,
  \[
  E_1 = E_2 \iff \mu_{E_1}(x) = \mu_{E_2}(x), \quad \text{for all } x \in X.
  \]

- **Inclusion**
  A fuzzy set \( E_1 \) is contained in the fuzzy set \( E_2 \) on \( X \), denoted by \( E_1 \subseteq E_2 \) if and only if the membership function values of all \( x \) belongs to \( E_1 \) is less than or equal to that of all \( x \) belongs to \( E_2 \), i.e.,
  \[
  E_1 \subseteq E_2 \iff \mu_{E_1}(x) \leq \mu_{E_2}(x), \quad \text{for all } x \in X.
  \]
  Also, a fuzzy set on \( X \) is empty, denoted by \( \emptyset \), if and only if \( \mu_E(x) = 0 \), for all \( x \in X \).
  As a matter of fact, the crisp set \( X \) can be viewed as a fuzzy set with membership function values \( \mu_X(x) = 1 \), for all \( x \in X \).

The fuzzy set-theoretic operations defined above are basically useful in the mathematical framework of a decision making process under fuzziness.

Several set-theoretic operations involving fuzzy sets proposed by Zadeh and many other fuzzy operations introduced by numerous researchers into the field of FST have been extensively included in the books prepared by Kaufmann [244], Gupta et al. [169], Dubois and Prade [118], Zimmermann [525], Kandle [239], Jones et al. [236],
Smithson [458], Klir and Folger [254], Novak [347], Klir and Yuan [255]. A comprehensive survey on fuzzy sets and its applications can be found in the book prepared by Klir [253] in 2000.

Now, the conceptual framework for decision making in fuzzy environment is introduced in the following section.

**Fuzzy Decision**

Let the set of possible alternatives be denoted by $X$ that contains the solution of a decision problem under consideration.

Then a fuzzy goal $\tilde{T}$ is a fuzzy set on $X$ characterized by the membership function

$$
\mu_{\tilde{T}} : X \rightarrow [0, 1].
$$

A fuzzy constraint $\tilde{C}$ is a fuzzy set on $X$ characterized by the membership function

$$
\mu_{\tilde{C}} : X \rightarrow [0, 1].
$$

Bellman and Zadeh [37] defined the fuzzy decision resulting from the goal $\tilde{T}$ and constraint $\tilde{C}$ as the intersection of $\tilde{T}$ and $\tilde{C}$ realizing that both the fuzzy goal and fuzzy constraint are desired to be satisfied simultaneously in making decision.

Explicitly, the fuzzy decision $\tilde{D}$ on $X$ can be expressed as:

$$
\tilde{D} = \tilde{T} \cap \tilde{C},
$$

and the characterization of it by a membership function can be presented as:

$$
\mu_{\tilde{D}}(x) = \min (\mu_{\tilde{T}}(x), \mu_{\tilde{C}}(x)), \ x \in X.
$$

The maximizing type decision is then defined as:

$$
\text{Maximize } \mu_{\tilde{D}}(X) = \max_{x \in X} \min (\mu_{\tilde{T}}(x), \mu_{\tilde{C}}(x)).
$$

In general, the fuzzy decision $\tilde{D}$ resulting from the $K$ fuzzy goals $\tilde{T}_1, \tilde{T}_2, \ldots, \tilde{T}_K$ and $M$ fuzzy constraints $\tilde{C}_1, \tilde{C}_2, \ldots, \tilde{C}_M$ is defined by

$$
\tilde{D} = \tilde{T}_1 \cap \tilde{T}_2 \cap \cdots \cap \tilde{T}_K \cap \tilde{C}_1 \cap \tilde{C}_2 \cap \cdots \cap \tilde{C}_M;
$$

and is characterized by the membership function

$$
\mu_{\tilde{D}}(x) = \min \left\{ \mu_{\tilde{T}_1}(x), \mu_{\tilde{T}_2}(x), \ldots, \mu_{\tilde{T}_K}(x), \mu_{\tilde{C}_1}(x), \mu_{\tilde{C}_2}(x), \ldots, \mu_{\tilde{C}_M}(x) \right\}, \ x \in X.
$$
Thus the maximizing type decision is defined as
\[
\text{Maximize } \mu_D(X)
\]
\[
\text{Minimize } \mu_{\tilde{T}_k}(x), \mu_{\tilde{C}_m}(x), \text{ for } k = 1, 2, \ldots, K, m = 1, 2, \ldots, M
\]
\[
\text{in the problem formulation process.}
\]
It is to be observed that in the above expression for decision \(\tilde{D}\), the goals \(\tilde{T}_k\), \(k = 1, 2, \ldots, K\) and constraints \(\tilde{C}_m\), \(m = 1, 2, \ldots, M\) are included in \(\tilde{D}\) in the same way. That is, there is no longer a difference between a fuzzy goal and a fuzzy constraint. However, other aggregation patterns for the fuzzy goals and fuzzy constraints may be worth considering and that depends on the decision situations.

The literature on the mathematical aspects of fuzzy decisions has been well documented in the books presented by Kickert [250], Foder et al. [135], Li et al. [284].

Now, the literature on applications of FST to different real-world decision making problems is briefly discussed in the following section.

1.2.2.2 Fuzzy Set Theory in Decision Making

The potential applications of FST to wide variety of MP problems studied by the researchers in different fields have been circulated in the literature.

Actually, the introduction of a classical mathematical framework, suggested by Bellman and Zadeh [37] in 1970 for decision making in fuzzy environment served as a point of departure for the study of methodological aspects of MP under fuzziness.

The term fuzzy MP (FMP) is the general term used in the area of study of Operational Research or Management Sciences [237]. But, the term fuzzy programming (FP) is widely used in the literature for the use of FST to various types of decision problems in the field of MP.

Now, the practical use fuzzy sets in MP, the process analysis, interpretation and establishments of mathematical models are different for different types of decision problems.

However, in a decision situation, two types of inexactness for establishment of a model are faced by the DM:

- DM's ambiguity of understanding the nature of the parameters in the problem formulation process.
- The fuzzy goals of the DM for the objective functions and the constraints.
The concept of FP for general MP problems was first proposed by Tanaka et al. [476] in 1974. More comprehensive surveys of the major FP approaches proposed through the mid-1980s can be found in [117, 237]. Empirical aspects of the FP approaches developed from mid-1980s to early 1990s can be found in different books, monographs, conference proceedings and peer reviewed journals in Zimmermann [524], Verdegay [490], Lai and Hwang [273], Dutta et al. [121, 122, 124], Rommelfanger and Slowinski [413].

In the programming and planning environment, the FP approaches for MP problems involving single-objective as well as multiple objectives have been studied in the past.

In the programming and planning environment, the FP approaches for MP problems involving single-objective as well as multiple objectives have been studied in the past.

A brief review of the methodological developments and applications of FP studied in the past for both types of decision problems are presented in the following sections.

1.2.2.3 Fuzzy Single Objective Programming

During 1980s, FP approaches for single-objective optimization problems have been studied extensively by the active researchers in the field.

The literature on the early developments of FP approaches to both the linear and nonlinear optimization problems has been reviewed thoroughly and classified systematically in the monographs prepared by Zimmermann [525, 526] and others.

The classical model of a single-objective programming, namely the LP, problem is of the form:

Find \( \mathbf{X} \) so as to

satisfy \( \text{Max } f(\mathbf{X}) = \mathbf{C X} \geq \mathbf{T} \)

subject to \( \mathbf{A X} \leq \mathbf{b} \),

\( \mathbf{X} \geq 0 \), \hspace{1cm} (1.9)

where \( \mathbf{C} \) and \( \mathbf{X}^T \in \mathbb{R}^n \), \( \mathbf{b}^T \in \mathbb{R}^m \), \( \mathbf{A} \) is an \( m \times n \) real matrix.
Now the notion of fuzzy goals introduced by Bellman and Zadeh [37] is applied for fuzzy set representation of the objective function and each of the structural constraints.

Then, the FP formulation of the problem (1.9) takes the form:

\[
\text{Find } X \text{ so as to}
\]
\[
\text{satisfy } C X \geq T,
\]
\[
\text{subject to } A X \leq b,
\]
\[
X \geq 0,
\]

where T is the aspiration level for achievement of the objective function; and \(\leq\) and \(\geq\), respectively, denote the fuzzified version of \(\leq\) and \(\geq\); and have the linguistic interpretations "essentially smaller than or equal" and "essentially greater than or equal", respectively in the sense of Zimmermann.

In model (1.10), it is to be observed that the objective function and the constraints are softening by fuzzifying them. Again, each soft constraint adds an additional objective to the problem, called fuzzy objective.

As a matter of fact, they can be presented in a unified form as:

\[
P X \leq Q,
\]
\[
X \geq 0,
\]

and, in general, it is termed as the fuzzy goal expression, where

\[
P = \begin{pmatrix} -C \\ A \end{pmatrix} \quad \text{and} \quad Q = \begin{pmatrix} -T \\ b \end{pmatrix}.
\]

In the field of FP, each of the goal expression in (1.11) is characterized by the corresponding membership function.

If the membership function of the i-th goal is represented by \(\mu_i(X)\), then the algebraic form of \(\mu_i(X)\) appears as [526] :

\[
\mu_i(X) = \begin{cases} 
1 & \text{if } (P X)_i \leq Q_i, \\
\frac{Q_i + t_i - (P X)_i}{t_i} & \text{if } Q_i < (P X)_i \leq Q_i + t_i, \\
0 & \text{if } (P X)_i > Q_i + t_i; \quad i = 1, 2, \ldots, m + 1
\end{cases}
\]

(1.12)
where $t_i$ is the allowable tolerance range of the $i$-th fuzzy goal level $D_i$.

An equivalent form of (1.12) is

$$
\mu_i(X) = \begin{cases} 
1 & \text{if } (P_X)_i \leq Q_i, \\
1 - \frac{(P_X)_i - Q_i}{t_i} & \text{if } Q_i < (P_X)_i \leq Q_i + t_i, \\
0 & \text{if } (P_X)_i > Q_i + t_i; \quad i = 1, 2, \ldots, m + 1
\end{cases}
$$

(1.13)

Now, following the fuzzy decision of Bellman and Zadeh [37], together with the defined membership functions in (1.13), the problem of finding the optimal decision is to determine $X^0$ such that

$$
\mu_D(X^0) = \max_{X \geq 0} \min_{i=1, \ldots, m+1} \left\{ \mu_i(X) \right\}
$$

(1.14)

Then, optimal decision can be obtained as:

$$
\mu_D(X^0) = \max_{X \geq 0} \min_{i=1, \ldots, m+1} \left\{ 1 - \frac{(P_X)_i - Q_i}{t_i} \right\}
$$

(1.15)

Introducing a new variable $\lambda$, which corresponds essentially to (1.15), the final decision problem appears as:

Find $X$ so as to

Maximize $\lambda$

subject to $\lambda t_i + (P_X)_i \leq Q_i + t_i, \quad i = 1, 2, \ldots, m + 1$

$\lambda \leq 1,$

$X, \lambda \geq 0.$

(1.16)

The problem in (1.16) matches with a single-objective programming model and can easily be solved by conventional single-objective programming methods.

The proposals for construction of membership functions have been extensively presented in [422, 455].

Now, in a decision making context, some of the constraints are found to be rigidly involved in the process of formulating the model. They can easily be incorporated in the framework of the model without involving any computational difficulty.

The literature on the methodological aspects and applications of fuzzy Multiobjective Programming (FMOP) are presented in the following Section 1.2.2.4.
1.2.2.4 Fuzzy Multiobjective Programming

FST has appeared as a promising tool to represent a MODM problem with fuzzy input data. The FP approach to MP problems with multiplicity of objectives was first proposed by Zimmermann [523] in 1978. Afterwards, the field of fuzzy MODM (FMODM) has grown tremendously both in theory and application due to the pioneering works of different researchers in the area of MP.

The literature on FMODM has been systemically classified and discussed by the pioneer authors [132, 239, 300, 422, 476, 526, 528] in the past. During the last decades, the methodological development of the field of FMODM has been extensively surveyed by Zimmermann [527], Luhandjula [307], Inuiguchi et. al. [219], Carlsson et al.[66]. However, most of the FMODM methods [81, 177, 304, 529] developed so far are mainly the extensions of Zimmermann's [523] modeling concept for multiobjective decision analysis.

An MODM problem with the fuzzy objectives and crisp constraints can be presented under the framework of FP called fuzzy goal programming (FGP) as:

Find \( \mathbf{X} \)
so as to satisfy

\[
\begin{align*}
\begin{array}{c}
Z_k(\mathbf{X}) \\
\geq
\end{array}
\begin{array}{c}
g_k, \ k = 1, 2, \ldots, K
\end{array}
\end{align*}
\]

subject to

\[
\begin{align*}
\begin{array}{c}
A \mathbf{X}
\end{array}
\begin{array}{c}
= \\
\leq
\end{array}
\begin{array}{c}
b
\end{array}
\end{align*}
\]

\( \mathbf{X} \geq 0 \),

where \( \mathbf{X} \) is the vector of decision variable, \( b \) is a constant vectors and \( A \) is a constant matrix, and where \( \leq, \geq \) and \( = \) indicates the fuzziness of the aspiration levels. The aspiration level \( g_k \) signifies that the DM will be satisfied even for a value smaller than \( g_k \) up to a certain tolerance limit and / or larger than the same or different tolerance limit, and that depends on the fuzziness of the objectives.
Construction of Membership Functions

Let $g_{lk}$ and $g_{uk}$ be the lower and upper tolerance limits, respectively, for achievement of the aspired level $g_k$ of the k-th fuzzy objective goal. Then the membership function, say $\mu_k(X)$, for the fuzzy goal $Z_k(X)$ can be characterized as follows:

For $\geq$ type of restriction, $\mu_k(X)$ takes the form

$$
\mu_k(X) = \begin{cases} 
1, & \text{if } Z_k(X) \geq g_k, \\
\frac{Z_k(X) - g_{lk}}{g_k - g_{lk}}, & \text{if } g_{lk} \leq Z_k(X) < g_k, \\
0, & \text{if } Z_k(X) < g_{lk}
\end{cases}
$$

(1.18)

For $\leq$ type of restriction, $\mu_k(X)$ becomes

$$
\mu_k(X) = \begin{cases} 
1, & \text{if } Z_k(X) \leq g_k, \\
\frac{g_{uk} - Z_k(X)}{g_{uk} - g_k}, & \text{if } g_k < Z_k(X) \leq g_{uk} \\
0, & \text{if } Z_k(X) > g_{uk}
\end{cases}
$$

(1.19)

For $\approx$ type of restriction, $\mu_k(X)$ can be expressed as:

$$
\mu_k(X) = \begin{cases} 
1, & \text{if } Z_k(X) = g_k \\
\frac{Z_k(X) - g_{lk}}{g_k - g_{lk}}, & \text{if } g_{lk} \leq Z_k(X) < g_k \\
\frac{g_{uk} - Z_k(X)}{g_{uk} - g_k}, & \text{if } g_k < Z_k(X) \leq g_{uk} \\
0, & \text{if } g_k(X) > g_{uk} \text{ or if } Z_k(X) < g_{lk}
\end{cases}
$$

(1.20)

Here, it is to be observed that the membership functions defined for $\geq$ and $\leq$ type of fuzzy goals are actually the particular forms of the membership function defined for $\approx$ type of fuzzy goals.
The schematic representation of linear membership functions in (1.18) and (1.19) are presented in Figures 1.2 and 1.3, respectively.

![Figure 1.2: The Membership Function of the k-th Fuzzy Goal in (1.18)](image1)

![Figure 1.3: The Membership Function of the k-th Fuzzy Goal in (1.19)](image2)

Now, achievement of the highest degree (unity) of a membership function means absolute achievement of the aspired level of the associated fuzzy goal. So, the membership goal corresponding to the k-th membership function with unity as the aspiration level can be presented as:

\[ \mu_k(X) + d_k^- - d_k^+ = 1, \]  

(1.21) 

\[ d_k^-, d_k^+ \geq 0 \text{ with } d_k^- \cdot d_k^+ = 0; \ k = 1, 2, \ldots, K. \]
Some nonlinear membership functions [423] are presented in the Figure 1.4 to Figure 1.6 below:

**Figure 1.4: Exponential membership function**

**Figure 1.5: Hyperbolic membership function**
Now, as in the case of conventional MODM approaches, two most efficient approaches for solving FMODM problems are:

Fuzzy GP (FGP) approach [341],
Interactive FP approach [132, 422],

The survey on the above two approaches and their applications have been well documented in the books prepared by Lai and Hwang [274] in 1994 as well as Slowinski ed. [455] in 1998.

However, between the above two multiobjective FP approaches, the literature on FGP has grown tremendously in the recent past from the viewpoint of its potential use to real-life problems. The FGP approach to MODM problems was first discussed by Narasimhan [341] in 1980. Narasimhan’s approach has been further extended by Hannan [178, 179], Narasimhan [342], and others. The FGP approach has been briefly reviewed by Ignizio [210] in 1982.

The FGP approach with different types is presented in the following Section 1.2.2.5.
1.2.2.5 Fuzzy Goal Programming

The modeling aspects of FGP within the framework of conventional GP have been studied by Hannan [179], Tiwari et al. [483, 484], and others. The various aspects of FGP formulation have also investigated by Rao et al. [395, 396], Martel et al. [314], Kim and Whang [252], Chen and Tsai [82]. In the above approaches individual achievement of each of the membership goals to its highest value to the extent possible unlike conventional GP method has been discussed by Pal and Moitra [374] in the recent past. However, the most prominent tool in the field of GP, the priority based GP, has not been so far widely circulated in the literature. However, FGP approaches developed in the past have been applied to different real-world problems by Pickens et al.[389], Sastry et al.[435], Bhattacharya et al.[42], Suzuki et al.[469], Lee et al. [278], Parra et al.[388].

**The Weighted FGP approach**

Under the framework of *minsum* GP, the equivalent FGP model of the problem (1.17), can be explicitly formulated as:

Find $X$ so as to

$$\text{Minimize } Z = \sum_{i=1}^{K} w_i^* d_i^*$$

and satisfy $\mu_k (X) + d_k^* - d_k^+ = 1$

subject to the given system constraints of the problem. (1.22)

$d_k^-, d_k^+ \geq 0$ with $d_k^- d_k^+ = 0$, $k = 1, 2, ..., K$

$w_i^*$ is the numerical weight associated with $d_k^-$ and represents the weight of importance of achieving the aspired level of the $k$-th goal.

Here, $w_i^*$ is presented by

$$w_i^* = \begin{cases} 
\frac{1}{(g_k - s_k)} & \text{for the defined } \mu_k \text{ in (1.18)} \\
\frac{1}{(s_{ak} - g_k)} & \text{for the defined } \mu_k \text{ in (1.19)} 
\end{cases}$$
**The Priority based FGP approach**

The priority based FGP model can be presented as:

Find \( X \) so as to

Minimize \( Z = [P_1(d^-), P_2(d^-), ..., P_L(d^-)] \)

and satisfy \( \mu_k(X) + d^-_k - d^+_k = 1 \),

subject to the given system constraints of the problem. \( (1.23) \)

where \( Z \) represents the vector of \( L \) priority achievement functions consisting of the under-deviational variables of the goals for minimizing them on the basis of the priorities of achieving the aspired levels of the associated goals, and \( d^-_k, d^+_k \) are the under- and over-deviational variables, respectively of the \( k \)-th goal. Also, the \( l \)-th priority factor \( P_l(d^-) \) is assigned to the set of commensurable goals that are grouped together in the problem formulation, and the priority factors have the relationship

\[ P_1 \gg P_2 \gg ... \gg P_l \gg ... \gg P_L, \]

where '\( \gg \)' implies much greater than i.e., the membership goals at the highest priority level (\( P_1 \)) are achieved to the extent possible before the set of membership goals at the next priority level (\( P_2 \)) is considered, and so forth. \( P_l(d^-) \) is a linear function of the weighted under-deviational variables at the \( l \)-th priority level, where \( P_l(d^-) \) is of the form:

\[ P_l(d^-) = \sum_{k=1}^{K} w_{l k}^d \cdot d^-_{l k}; \]

\[ w_{l k}^d, d^-_{l k} \geq 0, \quad k = 1, 2, ..., K; \quad l = 1, 2, ..., L; \quad L \leq K, \quad (1.24) \]

where \( d^-_{l k} \) is renamed for the actual deviational variable \( d^-_k \) to represent it at the \( l \)-th priority level, \( w_{l k}^d \) is the numerical weight associated with \( d^-_{l k} \) and represents the weight of importance of achieving the aspired level of the \( k \)-th goal relative to others which are grouped together at the \( l \)-th priority level. The values of \( w_{l k}^d \) (\( k = 1, 2, ..., K \)) are determined as:
\[ w_{ik} = \begin{cases} 
\frac{1}{(g_k - g_{wk})_l} & \text{for the defined } \mu_k \text{ in (1.18)} \\
\frac{1}{(g_{uk} - g_k)_l} & \text{for the defined } \mu_k \text{ in (1.19)}. 
\end{cases} \]

where \((g_k - g_{wk})_l\) and \((g_{uk} - g_k)_l\) are used to represent \((g_k - g_{lk})\), \((g_{uk} - g_k)\), respectively, at the l-th priority level.

It may be noted that if all the fuzzy goals are considered as equally important in a decision making context, the model will be the ‘Weighted FGP Model’ in the case of conventional GP model.

Again, if some of the system constraints are fuzzily described then the membership functions for then can also be defined in an analogous way and the corresponding membership goals can easily be incorporated in the framework of the above model.

In the FP formulation, it is to be observed that the inherent weakness of the conventional GP does not involve, because the deviations from the highest membership value are controlled by the stated tolerance zones.

### 1.2.3 Interval Programming

Interval programming (IvP) has appeared as one of the promising tools in the recent years for decision making problems in an imprecise environment where setting of imprecise values to the parameters involved with the problem is not possible in the decision situation. Inexactness of the model parameters are frequently involved in most of the decision making problems due to imprecise nature of human judgments. In inexact programming, instead of assigning single parameter values (crisp or fuzzy), interval valued parameters are introduced depending on the decision making environment.

The IvP approaches are actually based on the theory of interval arithmetic [338]. IvP was initially introduced by Moore in 1962 in his thesis work. The methodological aspects of interval programming have been studied by Berti [40, 41], Rokne and Lancaster [405] Soyster [459], Inuiguchi and Sakawa [221], Wu [503] extensively in the past. The interval programming approaches to decision problems within the frameworks GP have also been studied by Steuer [464] in the past.
Different procedures related to interval computation are well documented in the book presented by Alefeld and Herzberger [7].

IvP approach have been successfully applied to several real-world problems [201, 202, 203] like water resource planning, energy planning, investment decision making and corporate financial management, mobile robot path planning, portfolio selection. These problems are generally multiobjective in nature and they often conflict each other in decision making situation. In such a case DM feels more comfortable to specify interval rather than setting precise values to the parameters of the problem.

IvP approaches to decision problem in inexact environment have been deeply studied by Bitran [49] and Steuer [465] in the past. A comprehensive study to theoretical developments and potential use of interval analysis are well documented in the books prepared by Moore [338], Jaulin et al. [233], and others.

Two types of methodologies are used to solve IvP problems. The first one is based on satisficing philosophy of GP [452] and second one is based on the traditional method of optimization. GP approaches [206, 407] to IvP have been investigated by Inuiguchi and Kume [220] in 1991. The IvP approach with the use of penalty function has been studied by Romero and Vitoriano [412] in 1999. The IvP, with interval coefficients in objective functions have been investigated by Ishibuchi and Tanaka in [224]. In 1994, Tong proposed an approach [485] for solving IvP problems, where both the coefficients associated with objective function as well as constraints were in the form of interval. The solution approach with minimax regret criteria for obtaining the two types of solution, necessary and possibly optimal solutions have been investigated by Inuiguchi and Sakawa [221] in the past.

The use of different relations in interval arithmetic with consideration of midpoint and width of interval, different methodologies has been suggested in [70, 225, 340, 448]. The methodological developments of IvP made in the past had been surveyed by Oliviera and Anunes [349] in 2007.

However, methodological extension of IvP is still at an early stage from the view point of its use to different real-life problems. Again, IvP approaches to MOFPPs are yet to be circulated in the literature.
Some basic concepts of interval arithmetic are presented in the following Section 1.2.3.1.

1.2.3.1 Interval Arithmetic Preliminaries

Let the field of real numbers be denoted by \( \mathbb{R} \), and members of \( \mathbb{R} \) be denoted by lowercase letters. A subset \( \mathbb{R} \) of the form

\[
C = [c^L, c^U] = \{ t | c^L \leq t \leq c^U, t \in \mathbb{R} \},
\]

is called a closed interval where \( c^L, c^U \) denotes the lower and upper bound of the interval \( C \) on the real line \( \mathbb{R} \).

Real numbers \( c \in \mathbb{R} \) may be considered a special numbers \([c, c]\) and are referred to as point interval.

Let \( m(C) \) and \( w(C) \) be the midpoint and the width respectively of an interval \( C \).

Then, \( m(C) = \frac{1}{2} (c^L + c^U) \), \( w(C) = (c^U - c^L) \)

Now, the different operations are defined as follows [338]:

- **Scalar multiplication**
  
  The scalar multiplication of \( C \) is defined as:
  
  \[
  \lambda C = \begin{cases} 
  [\lambda c^L, \lambda c^U], & \lambda \geq 0 \\
  [\lambda c^U, \lambda c^L], & \lambda < 0 
  \end{cases}
  \]

- **Absolute value**
  
  Absolute value of \( A \) is defined as
  
  \[
  |C| = \begin{cases} 
  [c^L, c^U], & c^U \geq 0 \\
  [0, \max(-c^L, c^U)], & c^L < 0 < c^U \\
  [-c^U, -c^L], & c^U \leq 0 
  \end{cases}
  \]

- **Equality of two intervals**
  
  Two interval numbers \( C_1 = [a^L, a^U] \) and \( C_2 = [b^L, b^U] \) are called equal, that is \( C_1 = C_2 \) if and only if
  
  \[
  a^L = b^L, \quad a^U = b^U.
  \]

- **Arithmetic multiplication**
The binary operation ‘*’ can be defined between two interval numbers $C_1$ and $C_2$ as:

$$C_1 \ast C_2 = \{ x | x = a \ast b, a \in C_1, b \in C_2 \} = \{ x | x = a \ast b, a^L \leq a \leq a^U, b^L \leq b \leq b^U \}$$

‘*’ is designated as any of the operation of four conventional arithmetic operations.

- **Interval addition**
  Ordinary addition ‘+’ between two interval numbers $C_1$ and $C_2$ can be defined as:
  $$C_1 + C_2 = [a^L+b^L, a^U+b^U]$$

- **Interval subtraction**
  Subtraction between two intervals $A$ and $B$ can also be defined as
  $$C_1 - C_2 = [a^L-b^U, a^U-b^L]$$
  The ordinary subtraction ‘-’ defined above, sometimes, is termed as possible subtraction.
  Another binary operation ‘-’ is defined between interval numbers $A$ and $B$ as [220]:
  $$C_1 (-) C_2 = [a^L-b^L, a^U-b^U]$$
  This newly defined subtraction is called necessary subtraction.

- **Interval multiplication**
  Multiplication is defined as
  $$C_1 \cdot C_2 = [\min(a^Lb^L, a^Lb^U, a^Ub^L, a^Ub^U), \max(a^Lb^L, a^Lb^U, a^Ub^L, a^Ub^U)]$$

- **Interval division**
  Division between two interval numbers can be presented as:
  $$C_1/C_2 = [\min(a^L/b^L, a^L/b^U, a^U/b^L, a^U/b^U), \max(a^L/b^L, a^L/b^U, a^U/b^L, a^U/b^U)]$$, provided $b^L, b^U > 0$.

- **Interval Deviations**
  Two types of deviations, used in the field of IvP, are
  (a) Necessary Deviation
  (b) Possible Deviation.
The relevant descriptions related to the deviations mentioned above and methodologies developed for solving the problems are presented in the subsequent chapters.

1.2.3.2 An Overview of Interval Programming

In IvP, two different types of models are used. First type deals with the interval parameters appearing in the objective functions whereas in the second one, interval parameters in both the objective functions as well as constraints of the problem are taken into consideration.

The general model of IvP with interval valued objective function can be expressed as [485]:

\[
\text{Find } X(x_1, x_2, ..., x_n) \text{ so as to}
\]

\[
\text{Optimize } Z_r(X) = \sum_{j=1}^{n} [c_{j}^L, c_{j}^U] x_j, \quad r = 1, 2, ..., R
\]

subject to

\[
A X \begin{pmatrix}
\geq \\
\leq
\end{pmatrix} b,
\]

\[X \geq 0,\] (1.25)

where \( X \) is a decision vector of order \( n \times 1 \), \([c_{j}^L, c_{j}^U]\) (j = 1, 2, ..., n; r = 1, 2, ..., R) is closed interval, \( A \) is \( m \times n \) matrix, \( b \) is \( m \times 1 \) vector and superscripts \( L \) and \( U \) stand for lower and upper bounds of the coefficients respectively.

Again, interval coefficients in objective functions as well as constraints in multiobjective linear programming (MOLP) can be presented as [349]:

\[
\text{Find } X(x_1, x_2, ..., x_n) \text{ so as to}
\]

\[
\text{Optimize } Z_r(X) = \sum_{j=1}^{n} [c_{j}^L, c_{j}^U] x_j, \quad r = 1, 2, ..., R
\]

subject to \( \sum_{j=1}^{n} [a_{j}^L, a_{j}^U] x_j \leq [b_{i}^L, b_{i}^U], \quad i = 1, 2, ..., m. \) (1.26)
where $X$ is a decision vector of order $n \times 1$, $[c_{i}^{l}, c_{i}^{u}]$, $[b_{j}^{l}, b_{j}^{u}]$ ($j = 1, 2, \ldots, n$; $i = 1, 2, \ldots, m$; $r = 1, 2, \ldots, R$) are closed intervals. It is to be noted that the system constraints in (1.25) can be incorporated in the formulated problem (1.26).

### 1.2.4 The Genesis of Stochastic and Fuzzy Programming

In the recent years, the methods of multiobjective stochastic optimization problems have become increasingly important in searching for solutions of practical decision problems arising in economics, industry, healthcare, transportation, agriculture, military operations, and others.

Actually, in the real-life decision situations, it has been realized that all three of the probabilistic, fuzzy and interval-valued data are frequently involved, and the aspects of SP, FP and IvP need be taken into account in actual practice of modeling the problems and thereby taking proper decisions. However, consideration of both the aspects creates a great challenge for developing efficient solutions methods to deal with stochastic, fuzzy and interval-valued terms in actual practice.

Here, it may be mentioned that the fuzzification of objectives / constraints as well as randomization of various parameters with different probability distributions involved with the problems depends on the decision making environment.

The constructive modeling aspects on programming problems under randomness and fuzziness were first studied by Luhandjula [303] in 1983. Inuiguchi et al. [218], Li et al. [285], Katagiri et al. [242] presented their works in programming problems with randomness and fuzziness. The methodological development of fuzzy stochastic programming (FSP) approaches for solving LP problems has been surveyed by Luhandjula [308] in 2006. Now, since most of the real world problems are multiobjective in nature, FGP [483, 484] in the area of FMODM and as extension of conventional GP [206, 407] for multiobjective decision analysis has been studied deeply and applied to real-life problems. FSP approaches to multiobjective decision problems have been studied in the past.

The CCP, as a prominent tool in the field of SP, for solving FP problems with multiplicity of objectives has also been studied in the past [295, 297]. The use of IvP in the field of SP [381], FP [204, 393] and other optimization problems [378, 504] has also been studied in the recent past.
1.3 Global Solution Search Methods for Multiobjective Decision Making

In the decision making horizon, a large number of problems in engineering, science and management has been formulated as global optimization problems [52, 508] and throughout the centuries, many different approaches have been proposed and have been intensively studied and discussed for searching global optimal decisions. Most of the methods developed so far are path-oriented search methods. They start at an arbitrary point, apply some decision rules to obtain a new point as a local refinement one, and use again a random and/or deterministic decision whether to accept this point as new starting point or not. As a matter of fact, they are all designated as the 'local solution search methods' [136], and there is no criterion to decide whether a local solution is also the global solution. Further, the search space may be arbitrary complex and no further information of the objective function landscape is known besides the objective function values, i.e., there is no derivative or gradient information to apply classical decision rules. Therefore, conventional optimization methods that make use of derivatives, gradients and the like are, in general, not able to locate or identify the global optimum. Furthermore, local optimal solutions often prove insufficient for real-world decision problems.

A schematic presentation of how the local optimal solution often arises with use of a classical approach is given in the Figure 1.7.

Figure 1.7: A classical approach stuck in local optima 'p'

In the above context, volume-oriented search methods according to global reliability strategy, i.e., the methods for scanning the whole search space are of great demand in a decision making horizon.
Here, it may be pointed out that the optimum / near optimum as a most satisfactory one may be located anywhere in the search space, and no parts of the search region can be neglected. Again, in the case of arriving at a near optimal solution, it may be mentioned that convergence to the best is not an issue in business or in most walks of life; we are only concerned with searching better relative to others.

To cope with the situation of developing volume-oriented search methods, the Computational Intelligence (CI) based on principles of natural evolution has appeared as the promising area of study in the recent years.

Evolutionary Computation (EC) is the computation of human thought process in the field of CI by following the evolutionary principles of nature. EC is actually a field of simulating evolution and adaptation on a machine (a computer), occurring both in living and human cognitive systems.

There is intermediate search approach between heuristic and ECs, namely variable neighbourhood search approach [181]. In this approach, systematic change of neighbourhood within a possibly randomized local search algorithm yields a simple and effective function for combinatorial and global optimization.

In the field of EC, the population-based stochastic methods known as Evolutionary Algorithms (EAs) - inspired from the natural evolution - are widely used for global optimization problems. It is widely believed that the specific potential of EAs originates from their parallel search by means of entire populations [432].

EAs are actually adaptive algorithms and based on the most common evolutionary architecture in biological systems in the sense that they accumulate and use information to progressively improve their ability to solve the problems of interest.

The major characteristics of EAs are the maintenance of a set of solution points (called population) that are searched in parallel. Each point (individual solution) is evaluated according to the objective function (fitness function). Furthermore, a set of genetic operators (as in biological systems: selection, reproduction and mutation) is given that work on populations. Selection focuses the search to a “better” region of the search space by giving individuals with “better” fitness values a higher probability to be member of the next generation (loop iteration). On the other hand,
variation operators (reproduction and mutation) create new points in the search space.

Here, random mixing of the information of two or more individuals (via. crossover / recombination) and also random changes of a particular point (via mutations) that correspond to the neighborhood function of a traditional approach are considered.

There are currently three main avenues of research in simulated evolution: Evolution Strategies (ESs), Evolutionary Programming (EP), and Genetic Algorithms (GAs). Each method emphasizes a different facet of natural evolution. Evolution strategies emphasize behavioural changes at the level of individual. Evolutionary programming stresses behavioural change at the level of species. GAs stress chromosomal operators. Thus, the resulting evolutionary algorithms are based on the stochastic learning process within a population of individuals, each of which represents a search point in the space of potential solutions to a given problem $f(x)$.

ESs were developed in the 1960s at the Technical University of Berlin (TUB) in Germany by Schwefel [445] and Rechenberg [397]. They introduced a class of algorithms which imitate the principles of natural evolution for parameter optimization problem [398, 446]. The methodological developments in the study of ESs are well documented in [17, 18, 19, 325, 399, 447, 510]. ES systems use fixed-length, real valued string as representation. Each position in the string marks a separate behavioral trait. By concentrating on optimizing behavior, the representation and reproduction heuristics create objects that are behaviorally similar to their parents but they are not structurally similar [10]. Hashem et al. provided some real-world application of ESs in their work in the last decade of the previous century in [182, 183, 184].

EP is in many aspects similar to ESs, elaborated independently by L.J. Fogel et al. [139] in the early 1960s. The current state-of-the-art in EP by David B. Fogel [136, 137] started with its application to optimization problems and is termed as meta-EP [16, 138]. EP employs a model of evolution at a higher abstraction from both GAs and ESs. The methodological aspects of EP are well documented in the literature [17, 18, 80, 140, 509]. This approach models the reproductive relationship between a species behavior in successive generations.
1.3.1 A Brief History of Genetic Algorithms

Probably GAs are the best known EAs, receiving remarkable attention all over the world. GAs has been introduced by John Holland [192, 193] in 1970s. He used a population-based algorithm to evolve rules for a classifier system. These algorithms were applied to parameter optimization for the first time by De Jong [106], who laid down the foundations of this application technique. The extensive works in GAs are made available in the literature by Booker et al.[55], Davis [99, 101], DeJong [107], Goldberg [163], Grefenstette [166], Holland [194], Michalewicz et al.[322, 323, 326]. The extensive study in this field and methodological developments for GAs are well documented in the book prepared by Michalewicz [325], Goldberg [164], and others.

Genetics is the scientific study of genes and heredity—of how particular qualities or traits are transmitted from parents to offspring. Inspired by the Darwin’s principle of survival of fittest [348] and influenced by the population genetics of Mendel [320, 321], the biological perception of the existence of fitter one are simulated in GAs. The GA applications are widely considered as optimization tools in which the biological genetic models [496] are simulated in optimization algorithms.

GAs model a view of evolution as a search for structures, namely for organizations of genes which can be realized into organisms that perform well in a given environment. As a result, the object representation and reproduction operators of GAs emphasize the manipulation of structure independent, of the interpretation of the structure. In addition, the result of applying a reproduction heuristic in a GA is an object which shares structural similarity to the components that were generated from but which may have significantly different interpretation [10].

Numerous modifications of the original GA were applied to all fields of global optimization [16, 17, 108, 155, 486] in the past. The extensive literature of the application of GA in various related field of studies are available in the books of Holland [195], Davis [100, 102], Davidor [98], and others. The use of GA in optimization problems are investigated by Gen et al. [156, 288, 434] in the past.
1.3.1.1 The Basics of Genetic Algorithms

GAs are adaptive computational procedures [102, 164, 325] modelled on the mechanics of natural genetic systems. They express their ability by efficiently exploiting the historical information to speculate on new offspring with expected improved performance [164].

GA encode the parameters of the search space in structures called chromosomes (or strings). They execute it iteratively on a set of chromosomes, called population, with three basic operators selection / reproduction, crossover, mutation. They are different from normal optimization and search procedures in four ways:

- GAs work with the coding of the parameter set, not with the parameter themselves.
- GAs work simultaneously with multiple points, and not a single point.
- GAs search via sampling (a blind search)
- GAs search using stochastic operators, not deterministic rules.

Since GA works on a set of coded solutions, it has very little chance of getting stuck at a local optimum when used as an optimization technique. Again, the search space need not be continuous, and no auxiliary information, like derivative of the optimizing function is required. Moreover, the resolution of the possible search space is increased by operating on coded (possible) solutions and not on the solutions themselves.

The fundamental procedures of GAs can be summarized into steps as follows:

Step 1: Map a possible solution space into a genetic search space (individual representation), and define a fitness function by unifying an objective function and constraints in primary problems.

Step 2: Initialize a population of individuals.

Step 3: Evaluate each fitness value of individuals in the current population.

Step 4: If one of the termination conditions is satisfied, stop and return the best individual in all generations, and then change the individual into the possible solution. Otherwise, proceed to Step 5.

Step 5: Implement reproductive plan by using each fitness value of individuals and generate a new population.
Step 6: Create new individuals by mating current individuals with probability $P_c$ and replace the population before crossover with the population after crossover.

Step 7: Apply mutation to each bit of strings (individuals) with probability $P_m$. Then return to Step 3.

A schematic diagram of the basic structures of GA is shown in Figure 1.8.

![Diagram](Diagram.png)

**Figure 1.8: Basic steps of a genetic algorithm**

The evolution starts from a set of chromosomes (representing a potential solution set for the function to be optimized) and proceeds from generation to generation through genetic operations. Replacement of an old population with a new one is known as generation (or iteration) when generational replacement technique (where all the members of the old population are replaced with new ones) is used. GAs require only a suitable objective function, which is a mapping from the chromosomal space to the solution space, in order to evaluate the suitability or fitness of the derived solutions.

GAs are stochastic iterative algorithms, which cannot guarantee convergence; termination is hereby commonly triggered by reaching a maximum number of generations or by finding an acceptable solution or more sophisticated termination...
criteria indicating premature convergence. The algorithm of standard GA (SGA), which represents the basis of all variants of GA is presented in the following [5]:

Algorithm

Produce an initial population of individuals;
Evaluate the fitness of all individuals;
While termination condition not met do
    Select fitter individuals for reproduction and produce new individuals (crossover and mutation);
    Evaluate fitness of new individuals;
    Generate a new population by introducing new and fitter individuals and by deleting some old and weaker individuals;
End while

The special type of GA, which is the basis for theoretical GA research such as the well known schema theorem and accordingly the building block hypothesis, is also called the canonical GA (CGA).

A GA typically consists of the following components:

- A population of binary strings or coded possible solutions (biologically referred to as chromosomes)
- A mechanism to encode a possible solution (mostly as a binary string).
- Objectives function and associated fitness evaluation techniques.
- Selection / Reproduction procedure
- Genetic operators (crossover and mutation)
- Genetic parameters or probabilities to perform genetic operations

The components are discussed in details in the next subsection.

1.3.1.2 Genetic Operators and Terminologies

In the following, the fitness function, the main genetic operators, namely parent selection, crossover, mutation and the genetic parameters (namely probability of crossover $P_c$ and probability of mutation $P_m$) are to be described. Here, more
emphasize is given on a functional description of the principles rather than to give a complete overview of operator concepts [120].

**Fitness function**

It is chosen depending on the problem to be solved in such a way that the strings representing good points (possible solutions) in the search space has the high fitness values. This is the only information (also known as the pay-off information) that GAs use while searching for possible solutions.

**Selection**

Once a population has been generated and its fitness has been measured, the set of solutions, that are selected to be "mated" in a given generation, is produced. In the SGA, the probability that a chromosome of the current population is selected for reproduction is proportional to its fitness. There are many ways of accomplishing this selection. These include:

- **Roulette wheel Selection**

  Here, the expected number of descendant for an individual \( i \) is given as \( p_i = \frac{f_i}{\bar{f}} \)

  with \( f_i : S \to \mathbb{R}^+ \) denoting the fitness function and \( \bar{f} \) representing the average fitness of all individuals. Therefore, each individual of the population is represented by a space proportional to its fitness. By repeatedly spinning the wheel, individuals are chosen using random sampling with replacement.

- **Fitter codon selection**

  In this selection scheme, fitness of the portion of the string (codon) in its most significant position is taken into consideration instead of computing the fitness of entire string. This reduces the computational complexity.

Viz. in the following population,

```
111010000
111101010
010111010
101010010
```
Here, codons are determined by means of maximum occurrence of dominant value in the most significant bits. Codon length is the most significant bit to the position of first non-matching bits in the compared pair. E.g. the length is 4, when chromosome (i) and chromosome (ii) are compared. In the proposed selection scheme, it is not required to compute the strings from their most to least significant bits in order to convert them into corresponding decimal values and compare. The codon provides the fitter chromosome with less computational burden. E.g. in the above comparison, chromosome (ii) is fitter than chromosome (i), because all its entry inside the codon is dominant (one). However, chromosome (i) has one non-dominant (zero) value at its least position inside the codon. Similarly, if chromosome (iii) and (iv) are considered, codon with length 2 can decide the fitter one.

- **Linear-rank selection**
  In this context, the individuals of the population are ordered according to their fitness and copies are assigned in such a way that the best individual receives a pre-determined multiple of the number of copies the worst one receives [167]. On the one hand rank selection implicitly reduces the dominating effects of “super individuals” in populations (i.e. individuals that are assigned a significantly better fitness value than all the individuals), but on the other hand it warps the difference between close fitness values, thus increasing the selection pressure in stagnant populations. The rank based evaluation function can be defined as [230]:

\[
\text{eval}(V_i) = a(1-a)^{i-1}, \quad i = 1, 2, \ldots, \text{p\_size}.
\]

where, \( a \in (0, 1) \) in the genetic system be given.

- **Tournament selection**
  There are number of variants in this theme. The most common one is \( k \)-tournament selection where \( k \) individuals are selected from a population and the fittest individual of the \( k \) selected ones is considered for reproduction. In this variant, selection pressure can be scaled easily by choosing an appropriate number for \( k \).
Crossover

The main purpose of the crossover is to exchange information between randomly selected parent chromosomes by recombining parts of their corresponding strings. It recombines the genetic material of two parent chromosomes to produce offspring for the next generation. Conventional crossover technique for binary representation includes:

- **Single point crossover**

  A single random cut is made, producing two head sections and two tail sections. Two tail sections are then swapped to make two new individuals (chromosomes). This following Figure 1.9 systematically sketches this crossover method which is also called one-point crossover.

  ![Figure 1.9: Schematic display of a single point crossover](image)

  Consider the following example where the parts between the parents at crossover point between 3rd and 4th bit is swapped

<table>
<thead>
<tr>
<th>101</th>
<th>0011</th>
<th>1011001</th>
</tr>
</thead>
<tbody>
<tr>
<td>011</td>
<td>1001</td>
<td>0110011</td>
</tr>
</tbody>
</table>

- **Multiple point crossover**

  It is a natural extension of the single point crossover. In a n-point crossover there are n crossover points and substrings are swapped between the n points. According to researchers multiple point crossover is more suitable to combine
good features present in strings because it samples uniformly along the full-length of a chromosome [400]. At the same time multiple point crossover becomes more and more disruptive with an increasing number of crossover points. Decreasing the number of crossover points during the run of GA may reduce the computational complexity [401].

- **Uniform crossover**
  Given two parents, each gene in the offspring is created by copying the corresponding gene from one of the parents. The selection of the corresponding parent is undertaken via randomly generated crossover mask. At each index, the offspring gene is taken from the first parent if there is 1 in the mask at this index otherwise the gene is taken from the second parent [164].

**Mutation**

Mutations allow undirected jumps to slightly different areas of the search space. The basic mutation operator for binary coded problems is bit-wise mutation. Mutation occurs randomly and very rarely with a probability. For example, (0111001) becomes (0110001) due to flipping the fourth gene (from 1 to 0).

\[
\begin{array}{c}
0111001 \\
0110001
\end{array}
\]

After having generated a new generation of descendant (offspring) by crossover and mutation, the question arises that which of the new candidates should become member of next generations. In the context of evolution strategies, this fact determines the life span of the individuals and substantially influences the convergence behaviour of the algorithm.

**Genetic Parameters**

Both the crossover and mutations are performed stochastically. The genetic operators here imply the probability of crossover \(P_c\) and probability of mutation \(P_m\). The probability of crossover is chosen in such a way that recombination of potential string (highly fit chromosomes) increases without any disruption. Generally, the crossover probability \(P_c\) lies between 0.6 to 0.9 [164, 325]. Mutation occurs occasionally and probability of performing mutation operation is very low. Typically
the value of \( P_m \) lies between 0.001 to 0.01 [164, 325]. The importance of choosing the mutation probability value properly is discussed in [438].

Some biological terminologies related to GA are described below:

- All living organisms consist of cells containing the same set of one or more chromosomes, i.e., strings of DNA. A gene can be understood as an “encoder” of a characteristic, such as eye color. The different possibilities for a characteristic (e.g., brown, green, blue, gray) are called alleles. Each gene is located at a particular position (locus) on the chromosome.

- Most organisms have multiple chromosomes in each cell. The sum of all chromosomes, i.e., the complete collection of genetic material, is called the genome of the organism and the term genotype refers to the particular set of genes contained in a genome. Therefore, if two individuals have identical genomes, they are said to have same genotype.

- Organisms whose chromosomes are arranged in pairs are called diploid, whereas organisms with unpaired chromosomes are called haploid. In nature, most reproducing species are diploid. Humans for instance have 23 pairs of chromosomes in each somatic cell in their body. Recombination (crossover) occurs during reproduction in the following way.

- For producing a new child, the genes of the parents are combined to eventually form a new diploid set of chromosomes. Offspring are subject to mutation where elementary parts of DNA (nucleotides) are changed. The fitness of an organism (individual) is typically defined as its probability to reproduce, or as a function of the number of offspring the organism has produced.

To identify analogously with biological terminology, a table is provided below:

<table>
<thead>
<tr>
<th>Biological</th>
<th>GA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chromosome</td>
<td>String</td>
</tr>
<tr>
<td>Gene</td>
<td>Feature, Character</td>
</tr>
<tr>
<td>Allele</td>
<td>Feature value (0 or 1)</td>
</tr>
<tr>
<td>Locus</td>
<td>string position</td>
</tr>
<tr>
<td>Genotype</td>
<td>Encoded structure</td>
</tr>
<tr>
<td>Phenotype</td>
<td>Parameter set, Solution set, Decoded structure</td>
</tr>
<tr>
<td>Epistasis</td>
<td>Non-linearity</td>
</tr>
</tbody>
</table>
(a gene is epistatic if its presence suppresses the effect of a gene at another locus).

The schema theorem and the building block hypothesis are discussed in the following Section 1.3.1.3.

1.3.1.3 Schema Theorem and Building Block Hypothesis

It is to be realized that the GAs model a view of evolution as a search for structures (organizations of genes in Genotypes) [5].

The chromosomal structures (Genotypes) play the vital roles in searching the higher fitness values for survival through generations and thereby improving the quality of solution. The success of GAs is based on the efficient creation of fitter chromosomes from one generation to the next generation results in ideal solution. The ‘design rules’ for structural representation of chromosomes is based on the “Schema Theorem”, the Fundamental Theorem of Genetic Algorithm. A schema is a “template” (a form for Chromosome representation), which allow exploration of similarities among Chromosomes in the search space for generation of potential solution.

A schema is a string with fixed (0 and 1) and variable symbols, which represents all potential strings for searching solution. In binary string representation, the symbol alphabet \{0, 1, #\} is considered in schema construction, where \{#\} (don’t care) is a special wild card symbol (variable symbol) that matches both 0 and 1. As such, a schema can be defined as a special structured string of the space \{0, 1, #\}^l \ i.e., a string of length \ l involving 0 and 1 and the don’t care symbol #.

A schema represents all strings, which match it on all positions other than #. Essentially, every evaluated string actually gives partial information about the fitness of the set of possible schemata of which the string is a member.

For example, \ H=01\#1\# \ is a schema of length 5. We say that a string \ s \ belongs to the schema \ H \ if \ \ s \ and \ \ H \ coincide at all places where \ H \ is different from \ # \ (i.e, \ \ s \ matches the fixed 0 and 1 positions of \ H. For example, \ s = 01110 \ belongs to the schema \ H=01\#1\#. \ We will frequently identify a schema with the set of strings belonging to it.
So, for example,

\( H = 01\#1\# \) matches the four strings \{01010, 01011, 01110, 01111\}, where the defined 0 and 1 are the fixed positions in \( H \).

Schema \((\#, \#, \#, \#, \#)\) represents all strings of length 5.

It is clear that every schema matches exactly \( 2^r \) strings, where \( r \) is the number of don’t care symbols \# in a schema template.

Formally, if \( H = h_{i-1}, \ldots, h_0 \) and \( s = s_{r-1}, \ldots, s_0 \) then \( s \in H \) exactly when \( s_i = h_i \) whenever, \( h_i \neq \# \). It is to be noted that different schemata have different characteristics in the domain of interest for searching potential solution.

**Schema Properties**

The most important properties of schema are:

(i) The "defining length": The defining length \( (\delta) \) of a schema is the distance between the first and the last fixed string position.

\[
H_1 = (\#\#\#1\#1) \\
\text{and } H_2 = (\#1\#\#\#1) \text{ be the two schemata.}
\]

Then, \( \delta (H_1) = 7 - 5 = 2 \) and \( \delta (H_2) = 7 - 2 = 5 \). It is to be noted that the string \( s = 11111111 \), belongs to both \( H_1 \) and \( H_2 \), where as \( s' = 00001111 \) belongs to \( H_1 \), but \( s'' = 11110000 \) neither belongs to \( H_1 \) nor \( H_2 \).

Actually, \( \delta (H) \) of schema \( H \) defines the compactness of potential information contained in a schema \( H \). The notion of \( \delta (H) \) is useful in calculating the survival probability of the schema for a crossover.

(ii) The "order": The order of a schema, say schema \( H \), is denoted by \( 0(H) \), and defined by the number of fixed positions \( (0 \text{ and } 1 \text{ only}) \), the non-don’t care positions in the schema.

For example, \( 0(H) \) for :

\[
H_1 = (0\#11\#01) \text{ and } H_2 = (0\#\#\#01) \\
\text{are } o(H_1) = 5 \text{ and } o(H_2) = 3, \text{ respectively.}
\]

The notion of \( o(H) \) is useful in calculating the survival probability of the schema for a mutation.
Schema growth equation

The survival of schemata and their growth in the successive generations can be defined as follows.

Let \( m \) be the number of chromosomes of a population belonging to a particular schema at time \( t \), defined by \( m(H,t) \). Then, the expected number of chromosomes matched by \( H \) at time \( t+1 \) related to the same number at time \( t \) can be defined as:

\[
m(H,t+1) = m(H,t) \frac{f_H(t)}{f(t)}
\]

where \( f_H(t) \) is the average fitness value of the string representing schema \( H \), and \( f(t) \) is the average fitness value over all strings within the population.

From the above expression, it is to be followed that the number of strings in the population grows as the ratio of the fitness of the schema to the average fitness of the population. This means that, an "above average" schema receives an increasing number of strings in the next generation, and a "below average" schema receives a decreasing number of strings, and an average schema stays on the same level.

Now, it is to be realized that destruction of schema occurs for both the crossover and mutation operations.

For the schema \( H \), it is easy to see that the probability of destruction through crossover is equal to \( \delta(H)/(l-1) \) and the probability of survival is thus

\[
1 - \frac{\delta(H)}{l-1}
\]

When the crossover is applied with some probability \( p_c \), the probability of survival for \( H \) is then found as:

\[
p_c(H) = 1 - p_c \cdot \frac{\delta(H)}{l-1}
\]

Here, it can easily be realized that if a bad crossover site is selected, the schema \( H \) could still survive "accidentally" though the choice of the partner of the considered string.

Then, we have

\[
p_c(H) \geq 1 - p_c \cdot \frac{\delta(H)}{l-1}
\]
Here, it is important to note that a short defining length schema is less likely to be disrupted by a single point crossover operator.

As a matter of fact, it is found that

\[ m(H, t+1) \geq m(H, t) \frac{f_H(t)}{f(t)} \left( 1 - P_c \cdot \frac{\delta(H)}{1-1} \right) \]

Now, the effect of mutation to schema survival growth equation can be described as follows:

A mutation randomly changes bits from 0 to 1 or vice versa (for an example, the string \( s = 01110 \) belongs to the schema \( H = 0111\# \)).

By mutation, when the bit at the third position is flipped, the string \( s = 01010 \) is still belonged to \( H \), but if the bit at second position is flipped, the string \( s = 00110 \) does not belong to \( H \).

If the bit mutation probability is \( P_m \), then the probability of survival of a single bit is \( 1 - P_m \).

If \( o(H) \) be the order of schema \( H \), then the probability of surviving a mutation free certain schema \( H \) is \( P_m(H) = (1 - P_m)^{O(H)} \), which can be approximated by \( P_m(H) \approx (1 - P_m)^{O(H)} \) for \( P_m <\!\!\!< 1 \). Here, it is important to note that low-order schemata are less likely to be disrupted by a mutation operator.

Now, combining the effects of selection, crossover and mutation, the reproductive schema growth relation appears as:

\[ m(H, t+1) \geq m(H, t) \frac{f_H(t)}{f(t)} \left( 1 - P_c \cdot \frac{\delta(H)}{1-1} - o(H)P_m \right) \]

The result essentially indicates that the number of short schemata with low order and above average quality grows exponentially in subsequent generations of an GA.

**Building Blocks**

The high-fit, short-defining-length and low-order schemata are called Building Blocks. Now, the potential force behind the successful implementation of GAs can be defined as follows:

The success of an GA is based on the identification of building blocks in the search space. Building blocks are the partial solutions (genetic modules) of which good solutions are composed. Building blocks are exchanged between the individuals
by recombination. During optimization, GAs are able to identify the building blocks of an optimal solution, and to store them in the individuals within the population. The combination of several building blocks results in an ideal solution.

Then, the "Schema theorem" and "Building block hypothesis" are presented as follows:

Schema Theorem: Short-defining length, low-order, above-average schemata receive exponentially increasing trials in subsequent generations of GA.

Building Block Hypothesis: GA seeks near-optimal performance through the juxtaposition of short, low-order, high-performance schemata (called "Building Blocks").

1.3.1.4 Real-world Applications of Genetic Algorithms

As in the classical optimization, early researchers of evolutionary algorithms have also realized the need of how to deal with multiple objectives. G.E.P.Box [56] made hand-simulation of evolutionary operations for multiple objectives in 1957. He realized the necessity of using more than one objective in a design. Box was also concerned with the need for choosing an appropriate criterion as the objective function of the converted single objective optimization problem.

Fogel et al. [139] suggested and simulated a weighted approach in handling multiple objectives (or goals) in two different scenarios. With the task of finding a logic for transforming a sequence of input symbols to a sequence of output symbols, the investigators suggested a way to evaluate a solution for predicting the n-th symbol into the future (where n>0) with an evolution function $f_n$.


The first real implementation of multi-objective evolutionary algorithms (vector evaluated GA or VEGA) was suggested by Schaffer [437] in 1984. He modified the
simple GAs with selection, crossover, mutation by performing independent selection cycles according to each objective.

The use of GA in MODM problems are currently gaining significant attention from researchers in various field due to their effectiveness and robustness in searching for a set of global solutions. There have been many surveys on use of genetic techniques in Multiobjective Optimization problems by Fonseca and Fleming [144], Horn [196] and others.

The use of GA in engineering design is well documented in the book presented by Gen and Cheng [152, 153]. GA under the framework of MODM has been successfully implemented in optimal design [187], computer network [471], supply chain network [8].

GA, has been employed in hierarchical decision problems [318, 511] with the use of IvP studied by Pal et al.[363]. GA to CCP problems was studied by Ruiqing et al.[418], Pal and Gupta [355], Pal et al [369] and others.

GAs are used in different real-life applications with multiple criteria in areas of university time table planning [111], laminated ceramic composites optimization [34], job shop scheduling problem [147, 495], structural topology [473], transportation [234], allocation of customers in warehouse [522], and other domains.

The potential application of GA under the framework of MODM with different type of inexactness has been successfully investigated by the various researchers in the field. Sakawa et al.[426, 427, 430], Gen et al. [154], Kim et al.[251], Sasaki and Gen [434] presented the solution approach in fuzzy environment with the use of GA. GA has also been implemented in SP and CCP problems in [229, 230, 231]. The GA in IvP problems has been studied by Pal and Gupta [356] in the recent past. GA has also been successfully explored in problems with more than one type of inexactness by Pal and Gupta [360], Pal et al. [361]. In their work they have explored GA using FGP in the problems in which the parameters are stochastic in nature.
1.4 Outline of the Work

Chapter 2 presents GA based FGP solution method to MODM problems with fractional criteria. In the model formulation of the problem, first fractional objectives are transformed into fuzzy goals by defining the imprecise aspiration levels to each of them by employing the proposed GA. Then, the concept of membership functions in FST for measuring the degree of achievement of the fuzzy goals by defining the tolerance limits of them is introduced in the decision making context. In the executable priority based FGP model, the achievement of the highest membership values (unity) of the defined membership goals to the extent possible by minimizing the associated under-deviational variables on the basis of the priorities of achieving the goals is considered. In the solution process, the GA scheme is iteratively used to the FGP formulation by defining the fitness function and without linearizing the fractional membership goals to reach a satisfactory decision in the decision making environment. In the decision process, the notion of Euclidean distance function is used to perform the sensitivity analysis with the change of priorities and thereby to identify the appropriate priority structure under which the most satisfactory decision can be reached in the decision situation. The roulette-wheel selection, single point crossover and random mutation are used in the proposed GA.

In Chapter 3, the method presented in Chapter 2 has been explored to solve executable FGP model with weight structure. Here, a GP solution method based on GA for fuzzy multiobjective fractional programming problems is presented. The roulette-wheel selection, single point crossover and random mutation are used in the proposed GA. In the proposed solution method, first a genetically driven system is employed to develop the FGP framework of the problem. Then in the decision process, the proposed GA is used to achieve the highest membership values (unity) of the objectives to the extent possible on the basis of their relative weights of importance of achieving the fuzzy aspirations levels assigned to them in the decision making context. The effectiveness of the approach is illustrated by numerical examples studied previously.

In Chapter 4, GA based FGP procedure for modeling and solving BLPPs with fractional objectives in a hierarchical decision system. In the proposed approach, the individual optimal decision of each of the DMs are determined first by using the
proposed GA adopted in the process of solving the problem. Then the concept of membership functions in fuzzy sets for measuring the degree of satisfaction of the DMs regarding achievements of fuzzily described objective goals as well as the degree of optimality of the decision vector controlled by the upper-level DM are defined in the decision making context. In the decision process, the GA method is directly employed to the FGP formulation of the problem for achievement of the highest membership value (unity) of the defined membership goals of the problem. In the GA search process, the fitter codon selection scheme, two-point crossover and random mutation are adopted to reach a satisfactory solution in the decision making environment.

In Chapter 5, the use of GA to solve Interval-valued MOFP has been discussed. In the proposed approach, first the interval arithmetic technique is used to transform the fractional objectives with interval coefficients into the standard form of an interval programming problem with fractional criteria. Then, the redefined problem is converted into the conventional fractional goal objectives by using interval programming approach and then introducing under-and over-deviational variables to each of the objectives. In the model formulation of the problem, both the aspects of GP methodologies, minsum GP and minimax GP are taken into consideration to construct the interval function (achievement function) for accommodation within the ranges of the goal intervals specified in the decision situation where minimization of the regrets (deviations from the goal levels) to the extent possible within the decision environment is considered. In the solution process, GA approach is introduced directly into the GP framework of the proposed problem. In the proposed GA, the conventional roulette wheel selection scheme and arithmetic crossover are used for achievement of the goal levels in the solution space specified in the decision environment. The interval function defined for the achievement of the fractional goal objectives is considered the fitness function in the reproduction process of the proposed GA.

Chapter 6 is concerned with the multiobjective CCP problems. In the proposed approach, the individual optimal decision of each of the objectives are determined by using the GA scheme adopted in the process of solving the problem after converting the chance constraints into their deterministic equivalent. Then, the FGP model of
the problem is formulated by introducing the concept of tolerance membership functions in fuzzy sets. In the solution process, the GA method is employed to the FGP formulation of the problem for achievement of the highest membership value (unity) of the defined membership functions to the extent possible in the decision making environment.

In Chapter 7, the use of CCP problems has been explored again, but, unlike Chapter 6 the stochastic simulation approach has been used here in the solution search process instead of deterministic conversion. In the proposed approach, a stochastic simulation to the chance constraints having the continuous random parameters is introduced first to determine the candidate solutions in the decision making context. Then, in the model formulation, the fuzzy goal descriptions of the objective are defined by employing the proposed GA method. In the solution process, achievement of the membership goals of the defined fuzzy goals to the highest membership value (unity) by minimizing the associated under-deviational variables to the extent possible by using the GA scheme is taken into consideration.

In Chapter 8, the stochastic simulation approach discussed in Chapter 7 has been extended further in case of chance constrained fuzzy goal programming (CCFGP) problems. In the proposed approach, achievement of the membership functions defined for the fuzzily described chance constrained goal with specified probability as the aspiration levels to the highest degree (unity) to the extent possible is considered. In the model formulation of the problem, instead of converting the probabilistic membership function to their deterministic equivalent, the stochastic simulation technique is introduced to reach a feasible decision at each iterative step in the process of solving the problem. In the solution process, GA scheme with use of the proposed stochastic simulation approach is adopted to reach a most satisfactory decision in the uncertain decision making environment.

In Chapter 9, how the GA method can be efficiently used to FGP formulation of the problem for proper allocation of academic personnel to different teaching departments for enrichment of the academic activities of a university on a long-term basis in the academic planning horizon is presented. In the proposed model, the number of full-time teaching staff, part-time teaching staff and total pay-roll budget goals of each of the academic departments are described fuzzily. For smooth
functioning of academic activities, different ratios which are inherently fractional in nature are taken into consideration in the framework of the developed model of the problem. To expound the potential use of the approach, the case example of the University of Kalyani, West Bengal (W.B.), India is considered.

Chapter 10 provides the application potential of GA to solve long term land allocation problems for optimal cropping plan in the area of Agricultural planning. Here, utilization of total available land for cultivation, aspiration levels of the production of crops, expected profit from the farm as well as certain ratios in fractional form for crops production and profit achievement are fuzzily described in the decision making context. To illustrate the potential use of the approach, the case example of the Nadia district, West Bengal, India is considered. The obtained solution is compared with the existing cropping plan of the district as well as the solution of the FGP approach studied previously.

In Chapter 11, a case study incorporating the use of GA to priority based FGP model for patrolman power deployment problem in Traffic management system is presented. In this chapter, how the GA can be used in the process of solving FGP formulation of patrol manpower allocation problems to the road-segment areas of Metropolitan cities to deter traffic violations and accidents is presented.

The concluding remarks and the scope for future studies are discussed in Chapter 12. A sample pseudo code of GA is presented in the Appendix following the chapters. A brief index is annexed next and before the reprints.