Department of Business Technology

Progressively Interactive Evolutionary Multiobjective Optimization

Ankur Sinha

Aalto University

DOCTORAL DISSERTATIONS

Aalto University publication series
DOCTORAL DISSERTATIONS 17/2011

Progressively Interactive Evolutionary Multiobjective Optimization

Ankur Sinha

Aalto University
School of Economics
Department of Business Technology

Aalto University publication series
DOCTORAL DISSERTATIONS 17/2011

ISBN 978-952-60-4053-0 (pdf)
ISBN 978-952-60-4052-3 (printed)
ISSN-L 1799-4934
ISSN 1799-4942 (pdf)
ISSN 1799-4934 (printed)

Aalto Print
Helsinki 2011

Aalto University

Abstract

Aalto University, P.O. Box 11000, FI-00076 Aalto www.aalto.fi

Author
Ankur Sinha

Name of the doctoral dissertation
Progressively Interactive Evolutionary Multiobjective Optimization

Publisher Aalto University School of Economics

Unit Department of Business Technology

Series Aalto University publication series DOCTORAL DISSERTATIONS 17/2011

Field of research Decision Making and Optimization

Abstract

A complete optimization procedure for a multi-objective problem essentially comprises of search and decision making. Depending upon how the search and decision making task is integrated, algorithms can be classified into various categories. Following `a decision making after search′ approach, which is common with evolutionary multi-objective optimization algorithms, requires to produce all the possible alternatives before a decision can be taken. This, with the intricacies involved in producing the entire Pareto-front, is not a wise approach for high objective problems. Rather, for such kind of problems, the most preferred point on the front should be the target. In this study we propose and evaluate algorithms where search and decision making tasks work in tandem and the most preferred solution is the outcome. For the two tasks to work simultaneously, an interaction of the decision maker with the algorithm is necessary, therefore, preference information from the decision maker is accepted periodically by the algorithm and progress towards the most preferred point is made.

Two different progressively interactive procedures have been suggested in the dissertation which can be integrated with any existing evolutionary multi-objective optimization algorithm to improve its effectiveness in handling high objective problems by making it capable to accept preference information at the intermediate steps of the algorithm. A number of high objective un-constrained as well as constrained problems have been successfully solved using the procedures. One of the less explored and difficult domains, i.e., bilevel multi-objective optimization has also been targeted and a solution methodology has been proposed. Initially, the bilevel multi-objective optimization problem has been solved by developing a hybrid bilevel evolutionary multi-objective optimization algorithm. Thereafter, the progressively interactive procedure has been incorporated in the algorithm leading to an increased accuracy and savings in computational cost. The efficacy of using a progressively interactive approach for solving difficult multi-objective problems has, therefore, further been justified.

Keywords Evolutionary multi-objective optimization algorithms, multiple criteria decision-making, interactive multi-objective optimization algorithms, bilevel optimization

ISBN (printed) 978-952-60-4052-3 **ISBN (pdf)** 978-952-60-4053-0

ISSN-L 1799-4934 **ISSN (printed)** 1799-4934 **ISSN (pdf)** 1799-4942

Pages 135 **Location of publisher** Espoo **Location of printing** Helsinki **Year** 2011

Progressively Interactive Evolutionary Multi-objective Optimization

Ankur Sinha

Department of Business Technology
P.O. Box 21210, FI-00076
Aalto University School of Economics
Helsinki, Finland
Ankur.Sinha@aalto.fi

Abstract

A complete optimization procedure for a multi-objective problem essentially comprises of search and decision making. Depending upon how the search and decision making task is integrated, algorithms can be classified into various categories. Following 'a decision making after search' approach, which is common with evolutionary multi-objective optimization algorithms, requires to produce all the possible alternatives before a decision can be taken. This, with the intricacies involved in producing the entire Pareto-front, is not a wise approach for high objective problems. Rather, for such kind of problems, the most preferred point on the front should be the target. In this study we propose and evaluate algorithms where search and decision making tasks work in tandem and the most preferred solution is the outcome. For the two tasks to work simultaneously, an interaction of the decision maker with the algorithm is necessary, therefore, preference information from the decision maker is accepted periodically by the algorithm and progress towards the most preferred point is made.

Two different progressively interactive procedures have been suggested in the dissertation which can be integrated with any existing evolutionary multi-objective optimization algorithm to improve its effectiveness in handling high objective problems by making it capable to accept preference information at the intermediate steps of the algorithm. A number of high objective un-constrained as well as constrained problems have been successfully solved using the procedures. One of the less explored and difficult domains, i.e., bilevel multi-objective optimization has also been targeted and a solution methodology has been proposed. Initially, the bilevel multi-objective optimization problem has been solved by developing a hybrid bilevel evolutionary multi-objective optimization algorithm. Thereafter, the progressively interactive procedure has been incorporated in the algorithm leading to an increased accuracy and savings in computational cost. The efficacy of using a progressively interactive approach for solving difficult multi-objective problems has, therefore, further been justified.

General Keywords: Evolutionary multi-objective optimization algorithms, multiple criteria decision-making, interactive multi-objective optimization algorithms, bilevel optimization

Additional Keywords: Preference based multi-objective optimization, hybrid evolutionary algorithms, self-adaptive algorithm, sequential quadratic programming, algorithm development, test problem development

Acknowledgements

The dissertation has been completed at the Department of Business Technology, Aalto University School of Economics. The work has been performed under the supervision of Prof. Kalyanmoy Deb, Prof. Pekka Korhonen and Prof. Jyrki Wallenius. I would like to extend heartfelt gratitude and thanks to my supervisors for their inspiration and support which motivated me to start working on the dissertation with utmost enthusiasm and has been the driving force for successfully accomplishing the task. Their constant support and fruitful discussions encouraged new thoughts and ideas which were instrumental in executing the work. I consider myself fortunate to receive the best possible guidance and mentoring from the leading researchers in the field of Evolutionary Multi-objective Optimization and Multi Criteria Decision Making.

I thank the pre-examiners of this thesis, Prof. Juergen Branke and Prof. Lothar Thiele for their valuable feedback that helped in improving the dissertation. I would also thank all my colleagues at the department, in particular, Antti Saastamoinen, Pekka Malo and Oskar Ahlgren for providing a friendly work environment. I express my gratitude to the the department secretaries, Helena Knuuttila, Merja Mäkinen and Jutta Heino for their unconditional help and support.

I gracefully thank the graduate school on 'Systems Analysis, Decision Making and Risk Management', HSE foundation and the Academy of Finland for financially supporting this study. I would like to dedicate this thesis to my parents as I could not have accomplished the task successfully without their blessings. Finally, I wish to give a special mention to my wife, Parul, whose love, affection and support kept me rejuvenated even towards the completion of my dissertation.

Contents

Part I
Overview of the Dissertation

1 Introduction

Many real-world applications of multi-objective optimization involve a high number of objectives. Existing evolutionary multi-objective optimization algorithms [7, 34] have been applied to problems having multiple objectives for the task of finding a well-representative set of Pareto-optimal solutions [6, 4]. These methods have been successful in solving a wide variety of problems with two or three objectives. However, these methodologies tend to fail for high number of objectives (greater than three) [8, 22]. The major hindrances in handling high number of objectives relate to stagnation in search, increased dimensionality of Pareto-optimal front, large computational cost, and difficulty in visualization of the objective space. These difficulties are inherent to a multi-objective problem having a high number of dimensions and cannot be eliminated; rather, procedures to handle such difficulties need to be explored.

In many of the existing methodologies, preference information from the decision maker is utilized before the beginning of the search process or at the end of the search process to produce the optimal solution(s) in a multi-objective problem. Some approaches interact with the decision maker and iterate the process of elicitation and search until a satisfactory solution is found. However, not many studies have been performed where preference information is elicited during the search process and the information is utilized to progressively proceed towards the most preferred solution.

This dissertation is an effort towards development of progressively interactive procedures to handle difficult multi-objective problems, combining concepts from the fields of Evolutionary Multi-objective Optimization (EMO) and Multi Criteria Decision Making (MCDM). The fields of Evolutionary Multi-objective Optimization and Multi Criteria Decision Making have a common goal, but researchers have shown only lukewarm interest, until recently, in applying the principles of one field to the other. In the dissertation, emphasis has been placed on integration of methods and development of hybrid procedures that are helpful in the extension of the existing algorithms to handle challenging problems with multiple objectives. The utility of the procedures has also been shown on bilevel multi-objective optimization problems. The amalgamation of ideas has profound ramifications and addresses the challenges posed by multi-objective optimization problems.

The dissertation is composed of five papers which have been summarized in this introductory chapter. Before providing a summary, a short review of the basic concepts necessary to understand the papers will be given in the following sections.

1.1 Multi-objective Optimization

In a multi-objective optimization problem [30, 19, 6] there are two or more conflicting objectives which are supposed to be simultaneously optimized subject to a given set of constraints. These problems are commonly found in the fields of science, engineering, economics or any other field where optimal decisions are to be taken in the presence of trade-offs between two or more conflicting objectives. Usually such problems do not have a single solution which would simultaneously maximize/minimize each of the objectives; instead, there is a set of solutions which are optimal. These optimal solutions are called the Pareto-optimal solutions. A general multi-objective problem ($M \geq 2$) can be described as follows:

$$\begin{array}{ll} \text{Maximize} & \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_M(\mathbf{x})), \\ \text{subject to} & \mathbf{g}(\mathbf{x}) \geq \mathbf{0}, \mathbf{h}(\mathbf{x}) = \mathbf{0}, \\ & x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = 1, \ldots, n. \end{array} \tag{1.1}$$

In the above formulation, $\mathbf{x}$ represents the decision variable which lies in the decision space. The decision space is the search space represented by the constraints and variable bounds in a general multi-objective problem statement. The objective space $\mathbf{f}(\mathbf{x})$ is the image of the decision space under the objective function $\mathbf{f}$. In a single objective optimization ($M = 1$) problem the feasible set is completely ordered according to the objective function $\mathbf{f}(\mathbf{x}) = f_1(\mathbf{x})$, such that for solutions, $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ in the decision space, either $f_1(\mathbf{x}^{(1)}) \geq f_1(\mathbf{x}^{(2)})$ or $f_1(\mathbf{x}^{(2)}) \geq f_1(\mathbf{x}^{(1)})$. Therefore, for two solutions in the objective space there are two possibilities with respect to the $\geq$ relation.

However, when several objectives ($M \geq 2$) are involved, the feasible set is not necessarily completely ordered, but partially ordered. In multi-objective problems, for any two objective vectors, $\mathbf{f}(\mathbf{x}^{(1)})$ and $\mathbf{f}(\mathbf{x}^{(2)})$, the relations $=$, $>$ and $\geq$ can be extended as follows,

- $\mathbf{f}(\mathbf{x}^{(1)}) = \mathbf{f}(\mathbf{x}^{(2)})\ f_i(\mathbf{x}^{(1)}) = f_i(\mathbf{x}^{(2)}) :\Leftrightarrow i \in \{1, 2, \ldots, M\}$

- $\mathbf{f}(\mathbf{x}^{(1)}) \geq \mathbf{f}(\mathbf{x}^{(2)})\ f_i(\mathbf{x}^{(1)}) \geq f_i(\mathbf{x}^{(2)}) : \Leftrightarrow i \in \{1, 2, \ldots, M\}$
- $\mathbf{f}(\mathbf{x}^{(1)}) > \mathbf{f}(\mathbf{x}^{(2)}) \Leftrightarrow \mathbf{f}(\mathbf{x}^{(1)}) \geq \mathbf{f}(\mathbf{x}^{(2)}) \wedge \mathbf{f}(\mathbf{x}^{(1)}) \neq \mathbf{f}(\mathbf{x}^{(2)})$

While comparing the multi-objective scenario with the single objective case [5], in contrast we find that for two solutions in the objective space there are three possibilities with respect to the $\geq$ relation. These possibilities are: $\mathbf{f}(\mathbf{x}^{(1)}) \geq \mathbf{f}(\mathbf{x}^{(2)})$, $\mathbf{f}(\mathbf{x}^{(2)}) \geq \mathbf{f}(\mathbf{x}^{(1)})$ or $\mathbf{f}(\mathbf{x}^{(1)}) \ngeq \mathbf{f}(\mathbf{x}^{(2)}) \wedge \mathbf{f}(\mathbf{x}^{(2)}) \ngeq \mathbf{f}(\mathbf{x}^{(1)})$. If any of the first two possibilities are met, it allows to rank or order the solutions independent of any preference information (or a decision maker). On the other hand, if the first two possibilities are not met, the solutions cannot be ranked or ordered without incorporating preference information (or involving a decision maker). Drawing analogy from the above discussion, the relations $<$ and $\leq$ can be extended in a similar way.

1.2 Domination Concept and Optimality

1.2.1 Domination Concept

Based on the established binary relations for two vectors in the previous section, the following domination concept [14] can be constituted,

- $\mathbf{x}^{(1)}$ strongly dominates $\mathbf{x}^{(2)} \Leftrightarrow \mathbf{f}(\mathbf{x}^{(1)}) > \mathbf{f}(\mathbf{x}^{(2)})$,
- $\mathbf{x}^{(1)}$ weakly dominates $\mathbf{x}^{(2)} \Leftrightarrow \mathbf{f}(\mathbf{x}^{(1)}) \geq \mathbf{f}(\mathbf{x}^{(2)})$,
- $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ are non-dominated with respect to each other$\Leftrightarrow \mathbf{f}(\mathbf{x}^{(1)}) \ngeq \mathbf{f}(\mathbf{x}^{(2)}) \wedge \mathbf{f}(\mathbf{x}^{(2)}) \ngeq \mathbf{f}(\mathbf{x}^{(1)})$.

The above domination concept is also explained in Figure 1.1 for a two objective maximization case. In Figure 1.1 two shaded regions have been shown in reference to point A. The shaded region in the north-east corner (excluding the lines) is the region which strongly dominates point A, the shaded region in the south-west corner (excluding the lines) is strongly dominated by point A and the unshaded region is the non-dominated region. Therefore, point A strongly dominates point B, points A, E and D are non-dominated with respect to each other, and point A weakly dominates point C.

Most of the existing evolutionary multi-objective optimization algorithms use the domination principle to converge towards the optimal set of solutions. The concept allows us to order two decision vectors based on the corresponding objective vectors in the absence of any preference

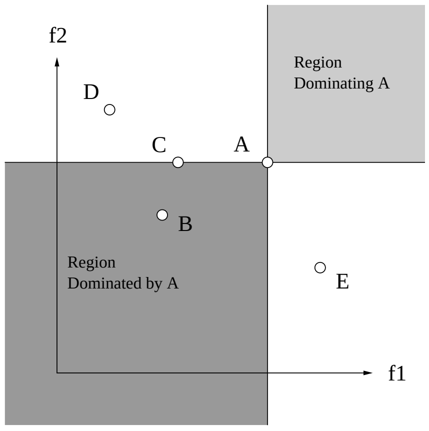

Figure 1.1: Explanation for the domination concept for a maximization problem where A strongly dominates B; A weakly dominates C; A, D and E are non-dominated.

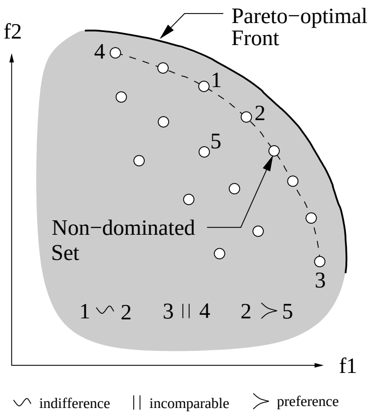

Figure 1.2: Explanation for the concept of a non-dominated set and a Pareto-optimal front. A hypothetical decision maker's preferences for binary pairs are also shown.

information. The algorithms which operate with a sparse set of solutions in the decision space and the corresponding images in the objective space usually give priority to a solution which dominates another solution. The solution which is not dominated with respect to any other solution in the sparse set is referred to as a non-dominated solution.

In case of a discrete set of solutions: the subset whose solutions are not dominated by any solution in the discrete set is referred to as the non-dominated set within the discrete set. The non-dominated set consists of the best solutions available and form a front called a non-dominated front. When the set in consideration is the entire search space, the resulting non-dominated set is referred as a Pareto-optimal set and the front is referred as the Pareto-optimal front. To formally define a Pareto-optimal set, consider a set $\mathbf{X}$, which constitutes the entire decision space with solutions $\mathbf{x} \in \mathbf{X}$. The subset $\mathbf{X}^* : \mathbf{X}^* \subset \mathbf{X}$, containing solutions $\mathbf{x}^*$, which are not dominated by any $\mathbf{x}$ in the entire decision space forms a Pareto-optimal set.

The concept of a Pareto-optimal front and a non-dominated front are illustrated in Figure 1.2. The shaded region in the figure represents $\mathbf{f}(\mathbf{x}) : \mathbf{x} \in \mathbf{X}$. It is the image in the objective space of the entire feasible region in the decision space. The bold curve represents the Pareto-optimal front for a maximization problem. Mathematically, this curve is $\mathbf{f}(\mathbf{x}^*) : \mathbf{x}^* \in \mathbf{X}^*$ which are all the optimal points for the two objective optimization problem. A number of points are also plotted in the figure, which constitute a finite set. Among this set of points, the points connected by broken lines are the points which are not dominated by any point in the finite set. Therefore, these points constitute a non-dominated set within the finite set. The other points which do not belong to the non-dominated set are dominated by at least one of the points in the non-dominated set.

In the field of Multi-Criteria Decision Making, the terminology slightly differs. For a given set of points in the objective space, the points which are not dominated by any other point belonging to the set are referred as non-dominated points, and their corresponding images in the decision space are referred as efficient. Based on the definition of weak and strong domination for a pair of points, the concept of weak efficiency and strong efficiency can be developed for a point within a set. A point $\mathbf{x}^* \in \mathbf{X}$, is weakly efficient if and only if there does not exist another $\mathbf{x} \in \mathbf{X}$ such that $f_i(\mathbf{x}) > f_i(\mathbf{x}^*)$ for $i \in \{1, 2, \ldots, M\}$. Weak efficiency should be distinguished from strong efficiency which states that a point $\mathbf{x}^* \in \mathbf{X}$, is strongly efficient if and only if there does not exist another $\mathbf{x} \in \mathbf{X}$ such that $f_i(\mathbf{x}) \geq f_i(\mathbf{x}^*)$ for all i and $f_i(\mathbf{x}) > f_i(\mathbf{x}^*)$ for at least one i.

The terminologies, efficiency and non-domination, are used differently in different fields. The researchers in the field of Data Envelopment Analysis tend to call the points in the objective space as efficient or inefficient. Some researchers prefer to call only the pareto-optimal points as efficient or non-dominated points. To avoid any confusion, we shall not be differentiating between efficiency and non-domination and the terminologies will be used only in reference to points belonging to a set. The two terminologies will be used synonymously for points in the objective space as well as the decision space, based on domination comparisons performed in the objective space. If the set in which domination comparisons are made, encompasses the entire feasible region in the objective space, then the efficient or non-dominated points for that set will be referred as pareto-optimal points.

In Figure 1.3, for a set of points $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$, the points $\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ are weakly efficient and the points $\{1, 2, 3, 4, 5\}$ are strongly efficient. Note that the set of all strongly efficient points is a subset of the set of all weakly efficient points. The point $\{10\}$ is inefficient as it is dominated by at least one other point in the set. It should be noted that the notion of efficiency arises while comparing points within a set. Here the set in consideration consists of 10 number of points with few as

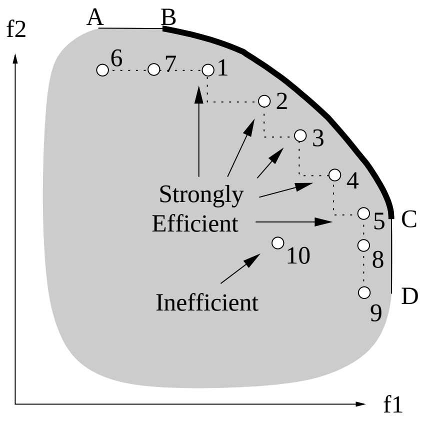

Figure 1.3: Explanation for efficiency

strongly efficient. The efficient points are not necessarily Pareto-optimal as it is obvious from the figure. The frontier ABCD represented in the figure is a weak Pareto-frontier and the subset BC shown in bold is a strong Pareto-frontier.

1.3 Decision Making

Even though there are multiple optimal solutions to a multi-objective problem, there is often just a single solution which is of interest to the decision maker; this is termed as the most preferred solution. Search and decision making are two intricacies [18] involved in handling any multi-objective problem. Search requires an intensive exploration in the decision space to get close to the optimal solutions; on the other hand, decision making is required to provide preference information for the non-dominated solutions in pursuance of the most preferred solution.

In a decision making context the solutions can be compared and ordered based on the preference information, though there can be situations where strict preference of one solution over the other is not obtained and the ordering is partial. For instance, consider two vectors, $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$, in the decision space having their images, $\mathbf{f}(\mathbf{x}^{(1)})$ and $\mathbf{f}(\mathbf{x}^{(2)})$, in the objective space. A preference structure can be defined using three binary relations $\succ$, $\sim$ and $\|$,

- $\mathbf{x}^{(1)} \succ \mathbf{x}^{(2)} \Leftrightarrow \mathbf{x}^{(1)}$ is preferred over $\mathbf{x}^{(2)}$,

- $\mathbf{x}^{(1)} \sim \mathbf{x}^{(2)} \Leftrightarrow \mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ are equally preferable,
- $\mathbf{x}^{(1)} \parallel \mathbf{x}^{(2)} \Leftrightarrow \mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ are incomparable,

where the preference relation, $\succ$, is asymmetric, the indifference relation, $\sim$, is reflexive and symmetric and the incomparability relation, $\parallel$, is irreflexive and symmetric. A weak preference $\succeq$ relation can be established as $\succeq = \succ \cup \sim$ such that,

- $\mathbf{x}^{(1)} \succeq \mathbf{x}^{(2)} \Leftrightarrow \mathbf{x}^{(1)}$ is either preferred over $\mathbf{x}^{(2)}$ or they are equally preferred.

As already mentioned, preference can easily be established for pairs where one solution dominates the other. However, for pairs which are non-dominated with respect to each other, a decision is required to establish a preference. The following are the inferences for preference choice which can be drawn from dominance:

- If $\mathbf{x}^{(1)}$ strongly dominates $\mathbf{x}^{(2)} \Rightarrow \mathbf{x}^{(1)} \succ \mathbf{x}^{(2)}$,
- If $\mathbf{x}^{(1)}$ weakly dominates $\mathbf{x}^{(2)} \Rightarrow \mathbf{x}^{(1)} \succeq \mathbf{x}^{(2)}$.

The binary relations $\succ$, $\sim$ and $\parallel$, are also explained in Figure 1.2 for a two objective case. Multiple points have been shown in the objective space and when comparisons are made between solutions in pairs then one of the binary relations will hold. For example, points 1 and 2 are close to each other; therefore, a decision maker may be indifferent between the two points. Points 3 and 4 lie on the extremes and are far away from each other; therefore, a decision maker may find such points incomparable. When points 2 and 5 are considered, a decision maker is not required as 2 dominates point 5; it can be directly inferred that a rational decision maker will prefer 2 over 5.

It is common to emulate a decision maker with an non-decreasing value function, $V(\mathbf{f}(\mathbf{x})) = V(f_1(\mathbf{x}), \ldots, f_M(\mathbf{x}))$, which is scalar in nature and assigns a value or a measure of satisfaction to each of the solution points. For two solutions, $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$,

- If $\mathbf{x}^{(1)} \succ \mathbf{x}^{(2)} \Leftrightarrow V(\mathbf{f}(\mathbf{x}^{(1)})) > V(\mathbf{f}(\mathbf{x}^{(2)}))$,
- If $\mathbf{x}^{(1)} \sim \mathbf{x}^{(2)} \Leftrightarrow V(\mathbf{f}(\mathbf{x}^{(1)})) = V(\mathbf{f}(\mathbf{x}^{(2)}))$.

1.4 Evolutionary Multi-objective Optimization (EMO) Algorithms

An evolutionary algorithm is a generic population based optimization algorithm which uses a mechanism inspired by biological evolution, i.e., selection, mutation, crossover and replacement. The common underlying idea behind an evolutionary technique is that, for a given population of individuals, the environmental pressure causes natural selection which leads to a rise in fitness of the population. A comprehensive discussion of the principles of an evolutionary algorithm can by found in [16, 24, 12, 1, 25]. In contrast to classical algorithms which iterate from one solution point to the other until termination, an evolutionary algorithm works with a population of solution points. Each iteration of an evolutionary algorithm results in an update of the previous population by eliminating inferior solution points and including the superior ones. In the terminology of evolutionary algorithms an iteration is commonly referred to as a generation and a solution point as an individual. A pseudo code for a generic evolutionary algorithm is provided next:

```
Step 1: Create a random initial population
Step 2: Evaluate the individuals in the population and assign fitness
Step 3: Repeat the generations until termination
    Sub-step 1: Select the most fit individuals (parents) from the population for reproduction
    Sub-step 2: Produce new individuals (offsprings) through Crossover and Mutation operators
    Sub-step 3: Evaluate the new individuals and assign fitness
    Sub-step 4: Replace low fitness members with high fitness members in the population
Step 4: Output
```

Along with the pseudo code presented above, a flowchart for a general evolutionary algorithm has also been presented in Figure 1.4. A pool of individuals is generated by randomly creating points in the search space which is called the population. Each member in the population is evaluated and assigned a fitness. For instance, while solving a single objective

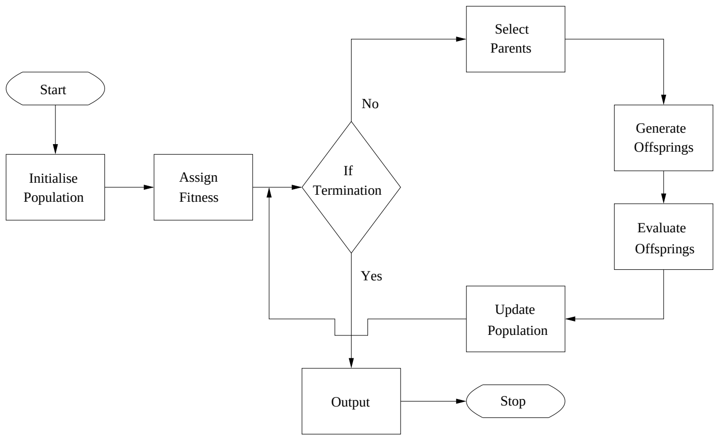

Figure 1.4: A flowchart for a general evolutionary algorithm

maximization problem, a solution point with a higher function value is better than a solution point with lower function value. Therefore, in such cases, the individual with higher function value is assigned a higher fitness. The function value can itself be treated as a fitness value in this case, or they can be transformed through a quality function to give the fitness measure. Similarly, for a multi-objective maximization problem a solution point which dominates another solution point is considered to be better. There is also a measure for crowdedness [6] which is used for individuals which cannot be ordered based on the domination principle. A multi-objective evolutionary procedure, therefore, assigns fitness to each of the solution points based on their superiority over other solutions points in terms of domination and crowdedness. Different algorithms use different approaches to assign fitness to an individual in a population. Once an initial population is generated and the fitness is assigned, few of the better candidates from the population are chosen as parents. Crossover and mutation is performed to generate new solutions. Crossover is an operator applied to two or more selected individuals and results in one or more new individuals. Mutation is applied to a single individual and results in one new individual. Executing crossover and mutation leads to offsprings that compete, based on their fitness, with the individuals in the population, for a place in the next generation. An iteration of this process leads to a rise in the average fitness of the population.

Using the described evolutionary framework, a number of algorithms have been developed which successfully solve a variety of optimization problems. Their strength is particularly observable in handling multi-objective optimization problems and generating the entire Pareto front. The aim of an evolutionary multi-objective optimization (EMO) algorithm is to produce solutions which are (ideally) Pareto-optimal and uniformly distributed over the entire Pareto-front so that a complete representation is provided. In the domain of EMO algorithms these aims are commonly referred to as convergence and diversity. The researchers in the EMO community have so far regarded an a posteriori approach to be an ideal approach where a representative set of Pareto-optimal solutions are found and then a decision maker is invited to select the most preferred point. The assertion is that only a decision maker who is well informed is in a position to take a right decision. A common belief is that decision making should be based on complete knowledge of the available alternatives; current research in the field of EMO algorithms has taken inspiration from this belief. Though the belief is true to a certain extent, there are inherent difficulties associated with producing the entire set of alternatives and performing decision making thereafter, which many a times renders the approach ineffective.

1.5 Integrating Search and Decision Making

Search and Decision Making can be combined in various ways to generate procedures which can be classified into three broad categories [19]. Each of the approaches to integrate the search and decision making will be discussed in the following sub-sections.

1.5.1 A posteriori Approach

In this approach, after a set of (approximate) Pareto-optimal solutions are obtained using an optimization algorithm, decision making is performed to find the most preferred solution. Figure 1.5 shows the process followed to arrive at the final solution which is most preferred to a decision maker. This approach is based on the assumption that a complete knowledge of all the alternatives helps in taking better decisions. The research in the field of evolutionary multi-objective optimization has been directed along this approach, where the aim is to produce all the possible alternatives for the decision maker to make a choice. The community has largely ignored decision making aspects, and has been striving towards producing all the possible optimal solutions.

There are enormous difficulties in finding the entire Pareto-optimal front for a high objective problem. Even if it is assumed than an algorithm can approximate the Pareto-optimal front for a high objective problem with a huge set of points, the herculean task of choosing the best point from the set still remains. For two and three objectives where the solutions in the objective space could be represented geometrically, making decisions might be easy (though even such an instance could be, in reality, a difficult task for a decision maker). Imagine a multi-objective problem with more than three objectives for which an evolutionary multi-objective algorithm is able to produce the entire front. The front is approximated with high accuracy and high number of points. Since a graphical representation is not possible for the Pareto-points, how is a decision maker going to choose the most preferred point? There are of course decision aids available, but the limited accuracy with which the final choice could be made using these aids, questions the purpose of producing the entire front with a high accuracy. Binary comparisons can be a solution to choose the best point out of a set, but this can only be utilized if the points are very few in number. Therefore, offering the entire set of Pareto-points should not be considered as a complete solution to the problem. However, the difficulties related to decision making have been realized by EMO researchers only after copious research has already gone towards producing the entire Pareto-front for many objective problems.

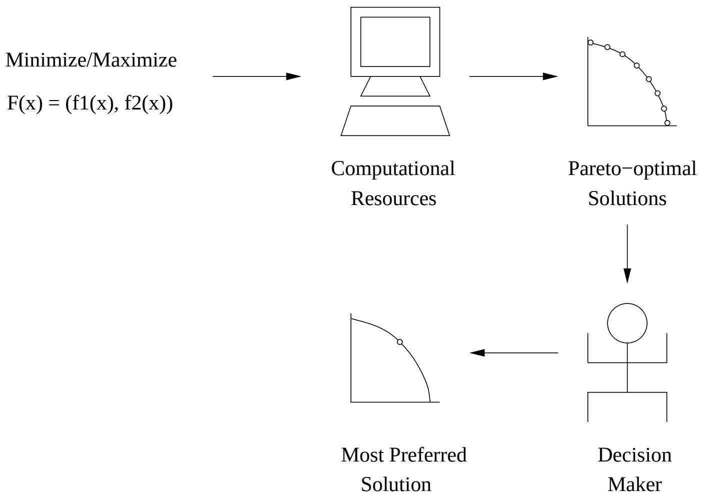

Figure 1.5: A posteriori approach.

1.5.2 A priori Approach

In this approach, decision making is performed before the start of the algorithm, then the optimization algorithm is executed by incorporating the preference rules, and the most preferred solution is identified. Figure 1.6 shows the process followed to arrive at the most preferred solution. This approach has been common among MCDM practitioners, who realized the complexities involved in decision making for such problems. Their approach to the problem is to ask simple questions from the decision maker before starting the search process. The initial queries usually include the direction of search, aspiration levels for the objectives, or preference information for one or more given pairs. After eliciting such information from the decision maker, the multi-objective problem is usually converted into a single objective problem. One of the early approaches, that is, Multi-Attribute Utility Theory (MAUT) [21] used the initial information from the decision maker to construct a utility function which reduced the problem to a single objective optimization problem. Scalarizing functions (for example, [32]) are also commonly used by the researchers in this field to convert a multi-objective problem into a single objective problem. Other techniques which are used to elicit information from a decision maker can be found in the review [23] on multi-criteria decision support.

Since information is elicited towards the beginning, the solution obtained after executing the algorithm is usually a satisfactory solution and may not be close to the most preferred solution. Moreover, the decision makers' preferences might be different for solutions close to the Pareto-optimal front and the initial inputs taken from them may not confirm it. Therefore, it will be difficult to get close to the actual solution which confirms to the requirements of the decision maker. The approach is also highly error prone as even slight deviations in providing preference information at the beginning may lead to entirely different solutions. To avoid the errors due to deviations, researchers in the EMO field used the approach in a slightly modified way. They produced multiple solutions in the region of interest to the decision maker [2, 11, 31, 17], instead of a single solution, therefore, giving choices to the decision maker at the end of the EMO search. However, researchers in the MCDM field recognized the enormous possibilities which could lead to erroneous results and therefore there exists a different school of thought which focusses on interactive approaches.

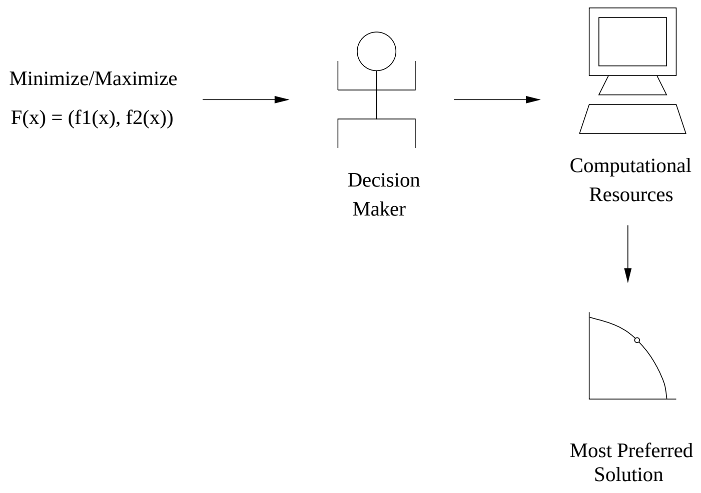

Figure 1.6: A priori approach.

1.5.3 Interactive Approach

In this approach, the decision maker interacts with the optimization algorithm and has multiple opportunities to provide preference information to the algorithm. The interaction between the decision maker and the optimization algorithm continues until a solution acceptable to the decision maker is obtained. The process is represented in Figure 1.7. Based on the type of interaction of the decision maker with the optimization algorithm, a variety of interactive approaches can exist. The dissertation discusses a special kind of an interactive approach referred as Progressively Interactive Approach.

Progressively Interactive Approach

A progressively interactive approach involves elicitation of preference information periodically from a decision maker. While the optimization algorithm is underway, preference information is taken at the intermediate steps of the algorithm, and the algorithm proceeds towards the most preferred point. This is a more effective integration of the search and decision making process, as both work simultaneously towards the exploration of the solution.

This approach overcomes the limitations of the previously discussed approaches as it allows actual interaction of the decision maker with the algorithm. The algorithm takes decision maker's preferences into account

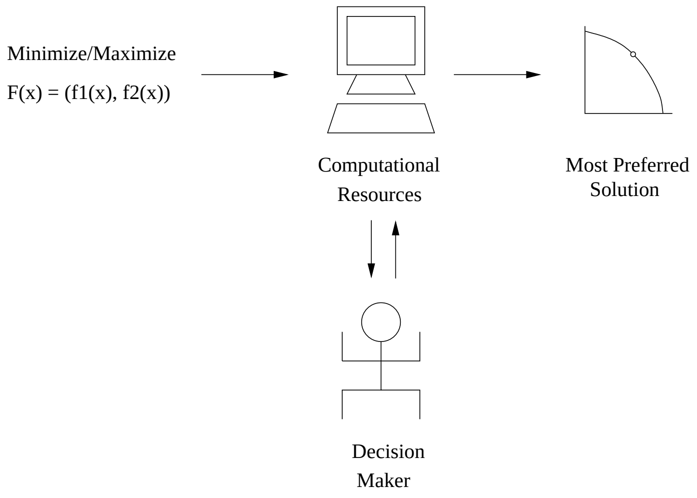

Figure 1.7: Interactive approach.

after every small step it takes towards the Pareto-optimal front. The approach offers a major advantage, as it allows the decision makers to change their preference structure as the algorithm progresses and more solutions are explored. Though the algorithm allows the decision maker to be seated in the driver's seat and have a greater control over the algorithm, it does not get mis-directed by a few errors which any human decision maker is prone to make. A progressively interactive approach with small step sizes (or frequent elicitation) is guaranteed to take a decision maker very close to the most preferred solution, as shown in the dissertation. The dissertation suggests two procedures which use a progressively interactive approach. In the first procedure, the decision maker value function is approximated after each step, and the second procedure constructs a polyhedral cone after each step. The progressively interactive approach is promising, as it avoids the drawbacks present in the other approaches. Some previous work which has been done in a similar vein in the MCDM field are [15, 33]. Little work [26, 13, 20, 3] on progressively interactive approach has been done in the field of EMO and calls for more contributions from researchers.

The iterations of a simple algorithm using a progressively interactive approach has been shown in the Figure 1.8. The decision maker is presented with a set of points and is expected to choose one of the points to start the search. The decision maker picks the point P_1. Based on answers, to the questions posed to the decision maker, a direction D_1 to perform the search is chosen. A scalarizing function is formulated based on the information and a search is performed with a fixed number of function evaluations (say n_f). This leads to a progress from point P_1 to P_2. At this instant, another decision maker call is made and more questions are asked from the decision maker. Based on the answers provided, the search direction is modified to D_2, and another scalarizing function is formulated based on point P_2 and new direction D_2. With n_f number of function evaluations, further progress is made from P_2 to P_3. Another decision maker call is executed and the process is repeated until no further progress is possible. In the figure it is shown that a satisfactory point is found in four decision maker calls. The point finally achieved by the algorithm is very close to the most preferred point. If the step size (P_1 to P_2, P_2 to P_3, P_3 to P_4 and P_4 to P_5 are the steps) of the algorithm is reduced, an even higher accuracy could be obtained, but with a higher number of decision maker calls. This procedure has potential to get close to the most preferred point which is, otherwise, difficult in other approaches.

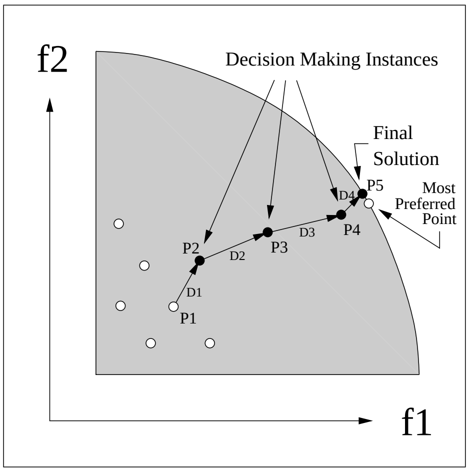

Figure 1.8: Progressively Interactive Approach.

1.6 Motivation

As already pointed out, the target of the EMO algorithms has been to find a set of well-converged and well diversified Pareto-optimal solutions. Once the optimization process is started there is no interaction involved with the decision maker until a set of representative Pareto solutions are found.

However, it is never guaranteed that a set of representative Pareto-optimal solutions can be obtained. There can be difficulties involved, for example, the algorithm is unable to converge to the optima, or the entire Pareto-front is not represented by the set of solutions. Though EMO procedures have shown their efficacy in solving multi-objective problems, they are not equipped to handle a high number of objectives. The challenges posed by high objective problems make the evolutionary multi-objective algorithms suffer in convergence as well as maintaining diversity. Moreover, the decision making task also becomes demanding when the non-dominated front cannot be represented geometrically. The difficulties necessitate coming up with procedures which can effectively handle the challenges offered by high objective optimization problems.

As there is just a single point which is most preferred to a decision maker, and finding the entire Pareto-optimal front has its own difficulties, there is motivation to aim for the most preferred solution by judiciously using search and decision making. The manual and the computational resources available can be effectively mobilized if the single point of interest is perpetuated as the target right from the start of the optimization process. It also alleviates the problems associated with generating the entire Pareto-optimal set. Therefore, it would be advisable to begin with the exploration along with inputs from a decision maker and advance towards the region or point of interest. Moreover, the conjugation is expected to find the most preferred solution with less computational expense and a high accuracy for difficult problems.

1.7 Summary of Research Papers

The dissertation consists of five papers which concern multi-objective optimization in general, and specifically deal with progressively interactive methods and bilevel optimization. The first paper is about a progressively interactive methodology which uses an implicitly defined value function and the second paper is an extension of the work. The third paper proposes a different progressively interactive methodology for the decision maker to interact with the algorithm and provide preferences. The fourth paper focusses on the less explored area of bilevel multi-objective optimization, and takes the domain a step forward by developing a generic evolutionary algorithm to handle the problem and also proposing test problems to evaluate the procedure. The fifth and final paper develops the previous paper by incorporating decision making in the suggested algorithm for bilevel multi-objective optimization. A short summary for each of the papers is provided in this section.

1.7.1 An Interactive Evolutionary Multi-Objective Optimization Method Based on Progressively Approximated Value Functions

The first paper [10] introduces a preference based optimization algorithm where decision making is incorporated in an evolutionary multi-objective search procedure. The algorithm requires preference based information to be elicited from the decision maker after every few generations. The decision maker is expected to order a given set of alternatives (five in number) according to preferences and the information is used to model a value function which emulates the decision maker and drives the algorithm towards more preferred solutions in the subsequent generations. A polynomial value function has been proposed which is quasi-concave in nature and is used to map the decision maker's preferences by optimally setting the parameters. The study suggests a simple optimization problem which is solved to determine the optimal parameters of the value function. The search of the evolutionary multi-objective optimization algorithm is focussed in the region of decision maker's interest by modifying the domination criteria based on the preference information. Further, a termination criterion based on preferences is also suggested. The methodology is evaluated on two to five objective unconstrained test problems, and the computational expense and decision maker calls required to arrive at the final solution are reported. The study also presents results obtained by emulating a decision maker who is prone to make errors while providing preference information.

1.7.2 Progressively Interactive Evolutionary Multi-Objective Optimization Method Using Generalized Polynomial Value Functions

The second paper [28] is an extension of the first paper where the polynomial value function has been augmented into a generalized polynomial value function. This equips the approach to fit a wider variety of quasi concave preference information. Further, the value function fitting procedure is tested on other commonly used value functions like the Cobb-Douglas value function and the CES value function, and the generality of the methodology is shown. Results are computed for constrained test problems up to five objectives. In this study the efficacy of the algorithm is also evaluated when the decision maker provides preference information in terms of partial ordering of the alternatives. In such cases the value function is generated by taking into account the indifference of the decision maker towards a pair of alternatives.

1.7.3 An Interactive Evolutionary Multi-Objective Optimization Method Based on Polyhedral Cones

The third paper [29] suggests a different progressively interactive algorithm by eliminating the requirement of a value function to progress towards the most preferred solution. This methodology accepts preference information in a different way during the intermediate steps of the algorithm and explores the region of interest. In this methodology the decision maker is expected to choose the best point from a provided set of alternatives. The number of alternatives shown to the decision maker is usually much higher than the number presented in the value function approach. However, in this case ordering of the points is not expected and the best alternative is chosen from a presented set using a visually interactive decision aid. The value function is replaced by a polyhedral cone which is constructed from a small set of points consisting of the best solution chosen by the decision maker and certain other solutions picked up by the algorithm. The polyhedral cone is used to modify the domination principle in order to focus more on the region of interest. The termination criterion is retained from the previous approach and the algorithm is once again evaluated on two to five objective test problems.

1.7.4 An Efficient and Accurate Solution Methodology for Bilevel Multi-Objective Programming Problems Using a Hybrid Evolutionary-Local-Search Algorithm

After successfully handling high objective problems in the previous papers, the fourth paper [9] deals with another challenging domain of multi-objective optimization, i.e. the bilevel multi-objective optimization problem. Bilevel optimization problems involve an upper and lower level of optimization tasks where an individual at the upper level can be feasible only if it is an optimal solution to a lower level problem. These problems are commonly found in practice but have not extensively been pursued by researchers primarily because of the complexity involved in handling them. Bilevel single objective optimization has received some attention but bilevel multi-objective optimization has largely been an untouched domain of optimization practitioners. In this study, the past key research efforts are highlighted and insights are drawn for solving such problems. The study discusses some of the intricate issues involved in handling bilevel multi-objective optimization problems. A number of test problems are also developed and a hybrid evolutionary-cum-local-search technique is proposed to handle the problems. The proposed solution methodology is made self adaptive such that the parameters of the algorithm need not be supplied by the user. All the test problems are two objective problems at both levels and the algorithm aims the entire front which leads to high number of function evaluations. The approach is once again a posteriori where decision making is performed after finding the entire front. Once a generic algorithm for handling bilevel multi-objective problem is available, in the next paper, it is augmented to interact with a decision maker and seek for the most preferred solution instead of the entire front.

1.7.5 Bilevel Multi-Objective Optimization Problem Solving Using Progressively Interactive EMO

In the fifth paper [27] the hybrid bilevel evolutionary multi-objective optimization algorithm has been extended to a progressively interactive algorithm such that the decision maker is able to interact during the search process and the most preferred solution could be obtained quickly and with much higher accuracy. The progressively interactive approach using the value function described in the first two papers is used in this algorithm at the upper level which allows decision maker preferences to be incorporated. Incorporating decision making at the upper level leads to six to ten times savings in function evaluations for all the considered test problems and is able to produce a solution much closer to the true solution. The algorithm, however, accepts decision maker preferences only at the upper level during the search process. Decision making at both levels during the search process opens an interesting area for researchers to pursue. Accepting preference information at both levels becomes a sophisticated problem, as it could lead to a conflict; decision maker at one of the levels should be given priority, or a mutually agreeable solution should be searched. This scenario has not been studied and does not fall in the realm of this dissertation.

References

[1] T. Bäck. Evolutionary Algorithms in Theory and Practice. New York: Oxford University Press, 1996.

[2] J. Branke and K. Deb. Integrating user preferences into evolutionary multi-objective optimization. In Y. Jin, editor, Knowledge Incorporation in Evolutionary Computation, pages 461–477. Hiedelberg, Germany: Springer, 2004.

[3] J. Branke, S. Greco, R. Slowinski, and P. Zielniewicz. Interactive evolutionary multiobjective optimization using robust ordinal regression. In Proceedings of the Fifth International Conference on Evolutionary Multi-Criterion Optimization (EMO-09), pages 554–568. Berlin: Springer-Verlag, 2009.

[4] C. A. C. Coello, D. A. VanVeldhuizen, and G. Lamont. Evolutionary Algorithms for Solving Multi-Objective Problems. Boston, MA: Kluwer, 2002.

[5] J. L. Cohon. Multicriteria programming: Brief review and application. In J. S. Gero, editor, Design Optimization, pages 163–191. New York: Academic Press, 1985.

[6] K. Deb. Multi-objective optimization using evolutionary algorithms. Chichester, UK: Wiley, 2001.

[7] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast and elitist multi-objective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation, 6(2):182–197, 2002.

[8] K. Deb and D. Saxena. Searching for Pareto-optimal solutions through dimensionality reduction for certain large-dimensional multi-objective optimization problems. In Proceedings of the World Congress on Computational Intelligence (WCCI-2006), pages 3352–3360, 2006.

[9] K. Deb and A. Sinha. An efficient and accurate solution methodology for bilevel multi-objective programming problems using a hybrid evolutionary-local-search algorithm. Evolutionary Computation Journal, 18(3):403–449, 2010.

[10] K. Deb, A. Sinha, P. Korhonen, and J. Wallenius. An interactive evolutionary multi-objective optimization method based on progressively approximated value functions. IEEE Transactions on Evolutionary Computation, 14(5):723–739, 2010.

[11] K. Deb, J. Sundar, N. Uday, and S. Chaudhuri. Reference point based multi-objective optimization using evolutionary algorithms. International Journal of Computational Intelligence Research (IJCIR), 2(6):273–286, 2006.

[12] D. B. Fogel. Evolutionary Computation. Piscataway, NY: IEEE Press, 1995.

[13] J. W. Fowler, E. S. Gel, M. Koksalan, P. Korhonen, J. L. Marquis, and J. Wallenius. Interactive evolutionary multi-objective optimization for quasi-concave preference functions.

[14] A. M. Geoffrion. Proper efficiency and theory of vector maximization. Journal of Mathematical Analysis and Applications, 22(3):618–630, 1968.

[15] A. M. Geoffrion, J. S. Dyer, and A. Feinberg. An interactive approach for multi-criterion optimization with an application to the operation of an academic department. Management Science, 19(4):357–368, 1972.

[16] D. E. Goldberg. Genetic Algorithms for Search, Optimization, and Machine Learning. Reading, MA: Addison-Wesley, 1989.

[17] G. W. Greenwood, X. Hu, and J. G. D'Ambrosio. Fitness functions for multiple objective optimization problems: Combining preferences with pareto rankings.

[18] J. Horn. Multicriterion decision making. In Handbook of Evolutionary Computation, pages F1.9:1–15. Bristol: Institute of Physics Publishing and New York: Oxford University Press, 1997.

[19] C.-L. Hwang and A. S. M. Masud. Multiple Objective Decision Making – Methods and Applications: A State-of-the-art Survey. Berlin: Springer-Verlag, 1979.

[20] A. Jaszkiewicz. Interactive multiobjective optimization with the pareto memetic algorithm. Foundations of Computing and Decision Sciences, 32(1):15–32, 2007.

[21] R. L. Keeney and H. Raiffa. Decisions with Multiple Objectives: Preferences and Value Tradeoffs. New York: Wiley, 1976.

[22] J. Knowles and D. Corne. Quantifying the effects of objective space dimension in evolutionary multiobjective optimization. In Proceedings of the Fourth International Conference on Evolutionary Multi-Criterion Optimization (EMO-2007), pages 757–771, 2007. (LNCS 4403).

[23] P. Korhonen, H. Moskowitz, and J. Wallenius. Multiple criteria decision support - a review. European Journal of Operational Research, 63:361–375, 1992.

[24] J. R. Koza. Genetic Programming : On the Programming of Computers by Means of Natural Selection. Cambridge, MA: MIT Press, 1992.

[25] M. Mitchell. Introduction to Genetic Algorithms. Ann Arbor, MI: MIT Press, 1996.

[26] S. Phelps and M. Koksalan. An interactive evolutionary metaheuristic for multiobjective combinatorial optimization. Management Science, 49(12):1726–1738, December 2003.

[27] A. Sinha. Bilevel multi-objective optimization problem solving using progressively interactive evolutionary algorithm. In Evolutionary Multi-Criterion Optimization (EMO-2011). In Press, 2011.

[28] A. Sinha, K. Deb, P. Korhonen, and J. Wallenius. Progressively interactive evolutionary multi-objective optimization method using generalized polynomial value functions. In 2010 IEEE Congress on Evolutionary Computation (CEC-2010), pages 1–8. IEEE Press, 2010.

[29] A. Sinha, P. Korhonen, J. Wallenius, and K. Deb. An interactive evolutionary multi-objective optimization method based on polyhedral cones. In 2010 Learning and Intelligent Optimization (LION-2010), pages 318–332. IEEE Press, 2010.

[30] R. E. Steuer. Multiple Criteria Optimization: Theory, Computation and Application. New York: Wiley, 1986.

[31] L. Thiele, K. Miettinen, P. Korhonen, and J. Molina. A preference-based interactive evolutionary algorithm for multi-objective optimization. Evolutionary Computation Journal, 17(3):411–436, 2009.

[32] A. P. Wierzbicki. The use of reference objectives in multiobjective optimization. In G. Fandel and T. Gal, editors, Multiple Criteria Decision Making Theory and Applications, pages 468–486. Berlin: Springer-Verlag, 1980.

[33] S. Zionts and J. Wallenius. An interactive programming method for solving the multiple criteria problem. Management Science, 22:656–663, 1976.

[34] E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the strength pareto evolutionary algorithm for multiobjective optimization. In K. C. Giannakoglou, D. T. Tsahalis, J. Périaux, K. D. Papailiou, and T. Fogarty, editors, Evolutionary Methods for Design Optimization and Control with Applications to Industrial Problems, pages 95–100, Athens, Greece, 2001. International Center for Numerical Methods in Engineering (Cmine).

Part II

List of Papers

1 An interactive evolutionary multi-objective optimization method based on progressively approximated value functions

K. Deb, A. Sinha, P. Korhonen, and J. Wallenius

IEEE Transactions on Evolutionary Computation, 14(5), 723-739. IEEE Press, 2010.

An Interactive Evolutionary Multi-Objective Optimization Method Based on Progressively Approximated Value Functions

Kalyanmoy Deb, Ankur Sinha, Pekka Korhonen and Jyrki Wallenius

***Abstract*— This paper suggests a preference based methodology, which incorporates an evolutionary multi-objective optimization algorithm to lead a decision-maker to the most preferred solution of her or his choice. The progress towards the most preferred solution is made by accepting preference based information progressively from the decision maker after every few generations of an evolutionary multi-objective optimization algorithm. This preference information is used to model a strictly monotone value function, which is used for the subsequent iterations of the EMO algorithm. In addition to the development of the value function which satisfies DM's preference information, the proposed progressively interactive EMO (PI-EMO-VF) approach utilizes the constructed value function in directing EMO algorithm's search to more preferred solutions. This is accomplished using a preference-based domination principle and utilizing a preference based termination criterion. Results on two to five-objective optimization problems using the progressively interactive NSGA-II approach shows the simplicity of the proposed approach and its future promise. A parametric study involving the algorithm's parameters reveals interesting insights of parameter interactions and indicates useful parameter values. A number of extensions to this study are also suggested.**

***Index Terms*— Evolutionary multi-objective optimization algorithms, multiple criteria decision-making, interactive multi-objective optimization algorithm, sequential quadratic programming, preference based multi-objective optimization.**

I. Introduction

In evolutionary multi-objective optimization (EMO), the target has usually been to find a set of well-converged and well-diversified Pareto-optimal solutions [1], [2]. Once an optimization run is started, usually no further information is elicited from the decision maker (DM). In an *aposteriori EMO approach*, after a set of approximate Pareto-optimal solutions has been found, preference information is elicited from a DM to choose the most preferred solution. As discussed elsewhere [3], [4], EMO procedures are not particularly applicable for handling a large number of objectives (practically, more than three). Firstly, the usual domination principle allows a majority of the population members to become non-dominated to each other, thereby not allowing much room for introducing new solutions in a finite population. This slows down the progress of an EMO algorithm. Secondly, the representation of a high-dimensional Pareto-optimal front requires an exponentially large number of points, thereby requiring a large population size in running an EMO procedure. Thirdly, the visualization of a high-dimensional front becomes a non-trivial task for decision-making purposes.

To alleviate the above problems associated with the aposteriori EMO approach, some EMO researchers have adopted a particular multiple criteria decision-making (MCDM) approach (*apriori approach*) and attempted to find a crowded set of Pareto-optimal points near the most preferred solution. Since the focus is now on finding a small region on the Pareto-optimal front, despite the high dimensionality of the problem, some of the difficulties mentioned above get alleviated and an EMO algorithm becomes suitable again. The cone-domination based EMO [5], biased niching based EMO [6], reference point based EMO approaches [7], [8], the reference direction based EMO [9], the light beam approach based EMO [10] are a few attempts in this direction. Also, Greenwood et al. [11] derived a linear value function from a given ranking of a few alternatives and then employed an EMO algorithm to find points which are preferred with respect to the constructed linear value function. In Greenwood's method, the preference information is used prior to employing the EMO algorithm, thus this qualifies as another apriori method. For a recent survey, see [12]. These studies have clearly shown that it is difficult for an EMO algorithm alone to find a good spread of solutions in 5 or 10-objective problems. When solutions around a specific Pareto-optimal point (or around a region) are the target, MCDM-based EMO approaches suggested in these studies can find satisfactory solutions. However, in these approaches, the decision maker interacts only at the beginning of an EMO run. The decision maker provides preference information such as one or more reference point(s), one or more reference directions, one or more light beam specifics, etc. An EMO algorithm then targets its population to converge near the specific solutions on the Pareto-optimal front.

The above MCDM-based EMO approaches can also be used in an iterative manner with a DM, similar to the way suggested elsewhere [13], [14]. In a *semi-interactive EMO approach*, some preference information (in terms of reference points, reference directions or other means) can be obtained from the

Kalyanmoy Deb is a Deva Raj Chair Professor at Indian Institute of Technology Kanpur, PIN 208016, India (deb@iitk.ac.in), Also an Adjunct Professor at Department of Business Technology, Aalto University School of Economics, PO Box 21210, FIN-00101, Helsinki, Finland (kalyanmoy.deb@hse.fi)

Ankur Sinha is a doctoral student at the Department of Business Technology, Aalto University School of Economics, PO Box 21210, FIN-00101, Helsinki, Finland (ankur.sinha@hse.fi)

Pekka Korhonen is a Professor at the Department of Business Technology, Aalto University School of Economics, PO Box 21210, FIN-00101, Helsinki, Finland (pekka.korhonen@hse.fi)

Jyrki Wallenius is a professor at the Department of Business Technology, Aalto University School of Economics, PO Box 21210, FIN-00101, Helsinki, Finland (jyrki.wallenius@hse.fi)

DM and an MCDM-based EMO algorithm can be employed to find a set of preferred Pareto-optimal solutions. Thereafter, a few representative preferred solutions can be shown to the DM and a second set of preference information in terms of new reference points or new reference directions can be obtained and a second MCDM-based EMO run can be made. This procedure can be continued till a satisfactory solution is found. This principle has been utilized with the reference direction [9] and light beam approaches [10] to solve some engineering design problems.

However, the integration of preference information within an EMO algorithm can be made in a more effective manner, as shown in a recent study [15]. Instead of keeping the DM waiting, to complete an EMO run (either to find a complete Pareto-optimal front in the aposteriori approach or to find a preferred set of Pareto-optimal solutions based on an MCDM principle in an apriori approach), the DM can be involved to periodically provide preference information as the EMO iterations are underway. This will be a less time-consuming and simultaneously more flexible approach than the previously suggested ones. In such a *progressively interactive EMO approach* using value functions (PI-EMO-VF), the DM is allowed to modify her/his preference structure as new solutions evolve. Since the DM gets more frequent chances to provide new information, the overall process is more DM-oriented. The DM may feel more in-charge and more involved in the overall optimization-cum-decision-making process.

In this paper, we have suggested a simplistic framework of a PI-EMO-VF approach based on a couple of earlier progressive multi-criterion decision-making approaches [16], [17]. Periodically, the DM is supplied with a handful of currently non-dominated points and is asked to rank the points from best to worst. From here on we refer to this instance as a 'DM call'. Based on this preference information, an optimization problem is formulated and solved to find a suitable value function, which optimally captures DM's preference information. From this iteration till the next DM call, the derived value function is utilized to drive the EMO algorithm in major ways: (i) in determining termination of the overall procedure and (ii) in modifying the domination principle, which directly affects EMO algorithm's convergence and diversity-preserving operators. The PI-EMO-VF concept is integrated with the well-known NSGA-II algorithm [18]. The working of the algorithm is demonstrated on four problems involving two to five objectives. A parameter sensitivity study is also performed to analyze the influence on working of the overall algorithm. Thereafter, the sensitivity of the proposed PI-NSGA-II-VF procedure on the inconsistencies in decision-maker responses is studied. Finally, a number of important and immediate future studies are listed and conclusions are drawn.

II. Past Studies on progressively interactive Methods

There exist a plethora of studies involving aposteriori and apriori EMO approaches. Most methodologies borrow the core decision-making idea from the MCDM literature and integrate it with an EMO algorithm. Since the focus of this study is not to discuss aposteriori or the apriori EMO approaches, but to concentrate on procedures requiring more frequent involvements of a DM with an EMO algorithm, we do not provide a review of aposteriori and apriori approaches, except to encourage the readers to look at a recent survey [12].

Towards the methodologies involving a progressive use of preference information by involving a decision-maker in an evolutionary multi-objective optimization framework, there are not many studies yet. Some recent studies periodically presented to the DM one or more pairs of alternative points found by an EMO algorithm and expected the DM to provide some preference information about the points. The information is then used to derive a weighted value function, which is linear. Phelps and Köksalan [19] optimized the constructed linearly weighted sum of objectives in subsequent iterations using an evolutionary algorithm. In their technique, if the actual value function is non-linear, the method may not be able to find a linear approximation and may generate an infeasible solution. This creates a need to reformulate the optimization problem by deleting constraints one at a time. Fowler et al. [20] have developed an interactive EMO approach based on the idea of using convex preference cones. They use such cones to partially order the population members and further use the order as the fitness function. They have tested their algorithm on multi-dimensional (upto 4 dimensions) knapsack problems. Jaszkiewicz [21] selected a set of linear value functions (based on weighted sum of objectives) from a set of randomly created linear value functions, conforming to the preference information supplied by the DM by pairwise comparisons. EMO algorithm's search is then continued with these selective weight vectors. Although the assumption of linear value functions facilitates a quick and easy determination of the value function representing DM's preference information, linear value functions have limitations in handling non-linear problems, particularly where the most preferred point lies on a non-convex part of the Pareto-optimal front. Nevertheless, each interactive EMO idea suggested in the above-mentioned studies remains as the main hallmark of these studies.

Branke et al. [15] implemented the GRIP [22] methodology in which the DM compares pairs of alternatives and the preference information thus obtained is used to find all possible compatible additive value functions (not necessarily linear). An EMO algorithm (NSGA-II) then used a preference-based dominance relationship and a preference-based diversity preserving operator to find new solutions for the next few generations. Their procedure recommended to make a single pair of comparison after every few generations in order to develop the preference structure. Since this procedure generates not enough preference information after every call of the DM, the EMO algorithm is likely to keep a wide variety of points from across the Pareto-optimal front in the population. The authors have demonstrated their procedure on a two-objective test problem. To obtain a narrow range of points close to the true preferred Pareto-optimal point, they had to call the DM at every generation of the EMO algorithm. It is not clear how the procedure will perform in higher objective problems, where dominance-based approaches are too slow and a reasonably high level of preference information would be needed to make a fast and focused search using an EMO algorithm. However, the use of preference information in EMO algorithm's operations remains a significant contribution of this study.

Korhonen, Moskowitz and Wallenius [16] suggested a progressive, interactive multi-objective optimization algorithm in which the DM is presented with a set of alternatives and is asked to make a set of binary comparisons of the alternatives. From this information, a linear programming problem is solved to identify a class of value functions in which the DM's preference information falls. They considered three classes of value functions for further processing: (i) linear, (ii) quasi-concave and (iii) no pre-assumed form. Based on this classification, a dominance structure is defined and either by search or from an existing sample of alternatives, the expected probabilities of finding new and better alternatives are determined. If there is a reasonable probability of finding better points, the algorithm is continued, otherwise the currently judged most preferred point is reported. An extension of this study [17] used a sampling based statistical procedure to compute expected probabilities of finding better solutions. It is clear that the algorithm is likely to perform better if the sampling procedure is replaced by an evolutionary multi-objective optimization algorithm for finding new points. After every decision-making event, an EMO algorithm can be employed for a few generations to find a better population of points, if available. Motivated by this study and recognizing the need for a simple interactive preference-based approach involving a DM in an EMO framework, we launch this particular study.

III. Proposed Progressively Interactive EMO Using Value Functions (PI-EMO-VF)

In this section, we propose an interactive EMO algorithm, where an approximate value function is generated progressively after every few generations. Here, we study optimization problems of the following form:

$$\begin{array}{ll}\text{Maximize} & \{f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_M(\mathbf{x})\}, \\ \text{subject to} & \mathbf{x} \in \mathcal{S},\end{array} \tag{1}$$

where $\mathbf{x}$ is a solution vector, $\mathcal{S}$ denotes the feasible search space and $f_i(\mathbf{x})$ is the i-th objective function. Usually, the objective functions are in conflict with each other.

A standard EMO algorithm (such as NSGA-II [18], SPEA2 [23] and others) works with a population of points in each iteration. A sparsed set of non-dominated points is preferred in a population so that the algorithm progresses towards the Pareto-optimal front and aims at finding a representative set over the entire front. However, in our proposed approach, we are interested in utilizing DM's preference information repeatedly as the algorithm progresses and in directing the search on the corresponding preferred region of the Pareto-optimal front iteratively.

For this purpose, after every τ generations of an EMO algorithm, we provide the decision-maker with η (≥ 2) well-sparsed non-dominated solutions from the current set of non-dominated points and expect the decision-maker to provide a complete or partial preference information about superiority or indifference of one solution over the other. In an ideal situation, the DM can provide a complete ranking (from best to worst) of these solutions, but partial preference information is also allowed. In the event that the DM is not able to declare any preferences, the algorithm has the back-tracking ability in search of new and preferred solutions. With the given preference information, we then construct a strictly increasing polynomial value function. The construction procedure involves solving a single-objective optimization problem. Till the next τ generations, we use the constructed value function to direct the search for additional such preferred solutions. A termination condition is also set up based on the expected progress, which can be made with respect to the constructed value function. In the following, we provide a step-by-step procedure of the proposed progressively interactive EMO using value function (PI-EMO-VF) methodology:

Step 1: Initialize a population Par_0 and set iteration counter $t = 0$. Set $Par_{\text{old}} = Par_0$. Domination of one solution over another is defined based on the usual definition of dominance [24] and an EMO algorithm is executed for τ iterations. The value of t is incremented by one after each iteration.

Step 2: If ($t \mod \tau = 0$), cluster the current non-dominated front to choose η widely distributed points; otherwise, proceed to Step 5.

Step 3: Obtain decision-maker's preferences on η points. If the DM is unable to declare a single preferred point in all pairwise comparisons, set $Par_t = Par_{\text{old}}$ and proceed to Step 5 with usual domination principle in EMO operators; otherwise set $Par_{\text{old}} = Par_t$ and proceed with the following operations. Construct a value function $V(\mathbf{f})$ from this information by solving an optimization problem (VFOP), described in Section III-A. If no feasible value function is found satisfying all DM's preference information, proceed to Step 5 and use the usual domination principle in EMO operators.

Step 4: A termination check (described in Section III-B) is performed based on the expected improvement in solutions from the currently judged best solution based on the value function $V(\mathbf{f})$. If the expected improvement is not significant (with respect to a parameter d_s, defined later), the algorithm is terminated and the current best solution is chosen as the final outcome.

Step 5: The parent population Par_t is used to create a new offspring population Off_t by using a modified domination principle (discussed in Section III-C) based on the current value function $V(\mathbf{f})$ and EMO algorithm's search operators.

Step 6: Populations Par_t and Off_t are used to determine a new population Par_{t+1} using the current value function and EMO algorithm's diversity preserving operator. The iteration counter is incremented as $t \leftarrow t + 1$ and the algorithm proceeds to Step 2.

The above is a generic progressively interactive PI-EMO-VF procedure, which can be combined with any existing EMO algorithm in Step 1 and subsequently in Steps 5 and 6. The PI-EMO-VF algorithm expects the user to set a value of τ, η and d_s.

In Step 2, points in the best non-dominated front are considered and the k-mean clustering algorithm [1], [23] can be used to identify η well-diversified points in the objective space. Other multi-criteria decision making methodologies [25] of selecting points from a set of non-dominated points may also be used.

We now provide the details for the specific procedures used in this study for Steps 3 to 6.

A. Step 3: Decision Maker's Preference Information and Construction of a Polynomial Value Function

At an instance of a DM call, η points are presented to the DM. The DM is expected to provide some preference information. One of the usual ways of providing such information is to make pairwise comparisons of given points and suggest one of the two scenarios: (i) a solution is more preferred over the other or (ii) both solutions are incomparable. Based on such preference statements, it is expected that for some pairs (i,j) of points, the i-th point is found to be preferred over the j-th point, thereby establishing $P_i \succ P_j$ and for some pairs (i,j), they are incomparable, establishing ($P_i \equiv P_j$). If the DM is unable to provide a clear preference information ($P_i \succ P_j$) for all pairs, it means that the current set of trade-off solutions are too diverse or the DM is not satisfied with any of these solutions. The algorithm then back-tracks to the previous population for which the DM could make a decisive action and instead of using the modified domination principle, the usual domination principle is used to advance the search process. However, in general, it is expected that the DM is able to establish at least one pair satisfying $P_i \succ P_j$. Thus, at the end of DM's preference elicitation task, usually we are likely to have at least one point which lies in the best category and at least one point which lies in the second-best category. In the 'complete ranking' situation, the DM may provide a complete ranking of η solutions (say, P_1 being the best, P_2 being the second-best and so on till P_η being the least preferred point).

Given such preference information, the task is to construct a polynomial value function satisfying the given preference structure. A similar task has been performed for linear utility functions elsewhere [26], [16]. Here, we construct a simple mathematical value function to capture the given preference information of η points.

1) Polynomial Value Function for Two Objectives: A value function is formed based on preference information provided by the decision maker. We first describe the procedure for two objectives and then present the procedure for the generic case. The structure of the value function is fixed as follows:

$$\begin{array}{lll} & V(f_1,f_2) = (f_1 + k_1 f_2 + l_1)(f_2 + k_2 f_1 + l_2), & \\ \text{where} & f_1, f_2 & \text{are the objective functions} \\ \text{and} & k_1, k_2, l_1, l_2 & \text{are the value function parameters} \end{array} \tag{2}$$

The value function V, for two objectives shown above, is considered to be the product of two linear functions $S_1 : \mathbf{R}^2 \to \mathbf{R}$ and $S_2 : \mathbf{R}^2 \to \mathbf{R}$. The parameters k_1, k_2, l_1 and l_2 are unknown and must be determined from the preference information concerning η points supplied by the decision-maker (DM). For this purpose, we solve the following optimization problem (VFOP):

$$\begin{array}{ll} \text{Maximize} & \epsilon, \\ \text{subject to} & V \text{ is non-negative at every point } P_i, \\ & V \text{ is strictly increasing at every point } P_i, \\ & V(P_i) - V(P_j) \geq \epsilon, \quad \text{for all } (i,j) \text{ pairs} \\ & \qquad \text{satisfying } P_i \succ P_j, \\ & |V(P_i) - V(P_j)| \leq \delta_V, \quad \text{for all } (i,j) \text{ pairs} \\ & \qquad \text{satisfying } P_i \equiv P_j. \end{array} \tag{3}$$

The first two sets of constraints ensure that the derived value function is non-negative and strictly increasing at all η points. The value function can always be shifted by adding a constant term to the function. Without loss of generality, we construct a value function which assigns a positive value to all the data points. These conditions satisfy the quasi-concavity of the value function – a desired property suggested in the economics literature [27]. This property in fact corresponds with the convex towards the origin indifference contours. The third and fourth sets of constraints ensure that the preference order supplied by the decision maker is maintained for respective pairs. In order to implement the first two constraint sets, we first sketch the value function $V(f_1,f_2)$ with the desired properties of being non-negative and strictly increasing. Figure 1 shows a pair of straight lines represented by $V(f_1,f_2) = 0$ at which either (or both) of the two terms S_1 or S_2 is zero. However, if

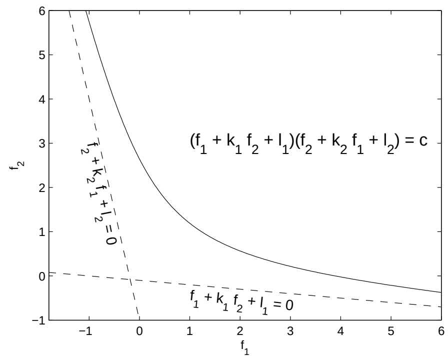

Fig. 1. The proposed value function.

the chosen points P_i ($i = 1,\ldots,\eta$) are such that both S_1 and S_2 are non-negative at these points, the first set of constraints will be satisfied. A generic iso-value curve for which $S_m > 0$ (for $m = 1,2$) is also depicted in the figure. Thus, the first set of constraints can be satisfied by simply considering $S_m \geq 0$ for $m = 1,2$. To impose strictly increasing nature of the value function at the chosen points, we can use $\partial V/\partial f_i \geq 0$ for both objectives. For the two-objective case, these two conditions yield $S_2 + k_2 S_1 \geq 0$ and $k_1 S_2 + S_1 \geq 0$.

The fourth constraint set takes into account all pairs of incomparable points. For such pairs of points, we would like to restrict the absolute difference between their value function values to be within a small range (δ_V). To eliminate having another parameter, we may like to use $\delta_V = 0.1\epsilon$, such that it is at most 10% of the maximum difference in value functions between $\succ$-class of points.

A little thought will reveal that the above optimization problem attempts to find a value function for which the minimum difference in the value function values between the ordered pairs of points is maximum. Considering all the expressions, we have the following optimization problem:

$$\begin{array}{ll} \text{Maximize} & \epsilon, \\ \text{subject to} & S_m(P_i) \geq 0, \quad i = 1, 2, \ldots, \eta, \text{ and } m = 1, 2, \\ & S_2(P_i) + k_2 S_1(P_i) \geq 0, \quad i = 1, 2, \ldots, \eta, \\ & k_1 S_2(P_i) + S_1(P_i) \geq 0, \quad i = 1, 2, \ldots, \eta, \\ & V(P_i) - V(P_j) \geq \epsilon, \quad \text{for all } (i, j) \text{ pairs} \\ & \qquad \text{satisfying } P_i \succ P_j, \\ & |V(P_i) - V(P_j)| \leq \delta_V, \quad \text{for all } (i, j) \text{ pairs} \\ & \qquad \text{satisfying } P_i \equiv P_j. \end{array} \tag{4}$$

Figure 2 considers five ($\eta = 5$) hypothetical points ($P_1 = (3.5, 3.7)$, $P_2 = (2.6, 4.0)$, $P_3 = (5.9, 2.2)$, $P_4 = (0.0, 6.0)$, and $P_5 = (15.0, 0.5)$) and a complete ranking of the points (P_1 being best and P_5 being worst). Due to a complete ranking, we do not have the fourth constraint set. The solution to the above optimization problem results in a value function, the contours (iso-utility curves) of which are drawn in the figure. The value function obtained after the optimization is as follows:

$$V(f_1, f_2) = (f_1 + 4.3229)(f_2 + 0.9401).$$

The asymptotes of this value function are parallel to f_1 and f_2 axes. The optimized value of ϵ is 2.0991. It is interesting to note the preference order and other restrictions are maintained by the obtained value function.

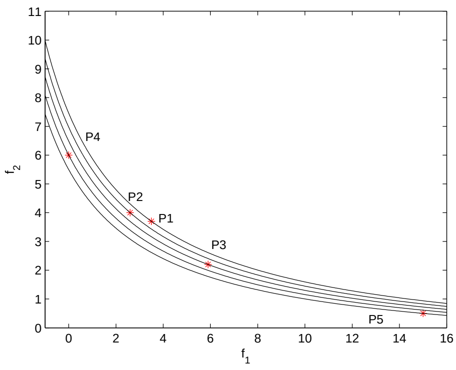

Fig. 2. Value function found by optimization.

Interestingly, if the DM provides a different preference information: P_1 is preferred over P_2, P_2 is preferred over P_3 and no preference exists among P_3, P_4 and P_5, a different value function will be obtained. We re-optimize the resulting problem with the above preference information on the same set of five points used in Figure 2 and obtain the following value function:

$$V(f_1, f_2) = (f_1 + 5.9355)(f_2 + 1.6613).$$

Figure 3 shows the corresponding value function contours. The contour makes a clear distinction between solutions in pairs P_1-P_2 and P_2-P_3 (within an optimized value ϵ), however, there is no distinction among P_3, P_4 and P_5 (with 0.1ϵ), to establish the given preference structure. Since a value function maintaining a difference (ϵ) between points in pairs P_1-P_2 and P_2-P_3 is needed and a maximum gap of 10% of ϵ is needed, a somewhat greater ϵ value to that found in the previous case is obtained here. The optimized ϵ value is found to be 2.2645 in this case.

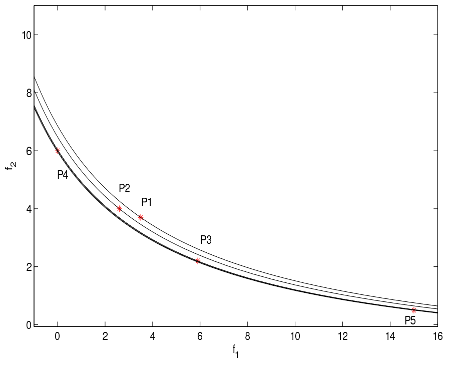

Fig. 3. Revised value function with a different preference information.

2) Polynomial Value Function for M Objectives: The above suggested methodology can be applied to any number of objectives. For a general M objective problem the value function can be written as follows:

$$\begin{array}{rl} V(\mathbf{f}) = & (f_1 + k_{11} f_2 + k_{12} f_3 + \ldots + k_{1(M-1)} f_M + l_1) \times \\ & (f_2 + k_{21} f_3 + k_{22} f_4 + \ldots + k_{2(M-1)} f_1 + l_2) \times \\ & \ldots \\ & (f_M + k_{M1} f_1 + k_{M2} f_4 + \ldots + k_{M(M-1)} f_{M-1} + l_M) \end{array} \tag{5}$$

The above value function can be expressed more elegantly as follows:

$$V(\mathbf{f}) = \prod_{i=1}^{M} \left(\sum_{j=1}^{M} \left[K_{ij} f_j + K_{i(M+1)} \right] \right). \tag{6}$$

Since each term in the value function can be normalized, we can introduce an additional constraint $\sum_{j=1}^{M} K_{ij} = 1$ for each term denoted by i. As discussed below, $K_{ij} \geq 0$ for $j \leq M$ and for each i, however $K_{i(M+1)}$ can take any sign. In the remainder of the paper, we follow the value function definition given in equation 5.

In the formulation it should be noted that the subscripts of the objective functions change in a cyclic manner as we move from one product term to the next. The number of parameters in the value function is M^2. The optimization problem formulation for the value function suggested above contains $M^2 + 1$ variables (k_{ij} and l_i). The variable ϵ is to be maximized. The second set of constraints (strictly increasing property of V) will introduce non-linearity. To avoid this, we simplify the above constraints by restricting the strictly increasing property of each term S_k, instead of V itself. The resulting constraints then become $k_{ij} \geq 0$ for all i and j combinations. The optimization problem (VFOP) to determine the parameters of the value function can thus be generalized as follows:

$$\begin{array}{ll} \text{Maximize} & \epsilon, \\ \text{subject to} & S_m(P_i) \geq 0, \quad i = 1, \ldots, \eta \text{ and } m = 1, \ldots, M, \\ & k_{ij} \geq 0, \quad i = 1, \ldots, M, \text{ and } j = 1, \ldots, (M-1), \\ & V(P_i) - V(P_j) \geq \epsilon, \quad \text{for all } (i,j) \text{ pairs} \\ & \qquad \text{satisfying } P_i \succ P_j, \\ & \qquad \text{combinations satisfying } i < j, \\ & |V(P_i) - V(P_j)| \leq \delta_V, \quad \text{for all } (i,j) \text{ pairs} \\ & \qquad \text{satisfying } P_i \equiv P_j. \end{array} \tag{7}$$

In the above problem, the objective function and the first two constraint sets are linear, however the third and fourth constraint sets are polynomial in terms of the problem variables. There are a total of $M\eta + M(M-1)$ linear constraints. However, the number of polynomial constraints depends on the number of pairs for which the preference information is provided by the DM. For a 10-objective ($M = 10$) problem having $\eta = 5$ chosen points, the above problem has 101 variables, 140 linear constraints, and at most 10 ($\binom{5}{2}$) polynomial constraints. Since majority of the constraints are linear, we suggest using a sequential quadratic programming (SQP) algorithm to solve the above problem. The non-differentiability of the fourth constraint set can be handled by converting each constraint ($|g(\mathbf{x})| \leq \delta_V$) into two constraints ($g(\mathbf{x}) \geq -\delta_V$ and $g(\mathbf{x}) \leq \delta_V$)). In all our problems, we did not consider the cases involving the fourth constraint and leave such considerations for a later study.

B. Step 4: Termination Criterion

Once the value function V is determined, the EMO algorithm is driven by it in the next τ generations. The value function V can also be used for determining whether the overall optimization procedure should be terminated or not. To implement the idea we identify the best and second-best points P_1 and P_2 from the given set of η points based on the preference information. In the event of more than one point in each of the top two categories (best and second-best classes) which can happen when the '$\equiv$'-class exists, we choose P_1 and P_2 as the points having highest value function value in each category, respectively.

The constructed value function can provide information about whether any new point P is better than the current best solution (P_1) with respect to the value function. Thus, if we perform a single-objective search along the gradient of the value function (or ∇V) from P_1, we expect to create solutions more preferred than P_1. We can use this principle to develop a termination criterion.

We solve the following achievement scalarizing function (ASF) problem [28] for $P_1 = \mathbf{z}^b$:

$$\begin{array}{ll} \text{Maximize} & \left(\min_{i=1}^{M} \frac{f_i(\mathbf{x}) - z_i^b}{\frac{\partial V}{\partial f_i}} \right) + \rho \sum_{j=1}^{M} \frac{f_j(\mathbf{x}) - z_j^b}{\frac{\partial V}{\partial f_j}}. \\ \text{subject to} & \mathbf{x} \in \mathcal{S}. \end{array} \tag{8}$$

Here, $\mathcal{S}$ denotes the feasible decision variable space of the original problem. The second term with a small ρ ($= 10^{-10}$ is used here) prevents the solution from converging to a weak Pareto-optimal point. Any single-objective optimization method can be used for solving the above problem and the intermediate solutions ($\mathbf{z}^{(i)}, \ i = 1, 2, \ldots$) can be recorded. If at any intermediate point, the Euclidean distance between $\mathbf{z}^{(i)}$ from P_1 is larger than a termination parameter d_s, we stop the ASF optimization task and continue with the EMO algorithm. In this case, we replace P_1 with $\mathbf{z}^{(i)}$. Figure 4 depicts this scenario. On the other hand, if at the end of the SQP run, the final SQP solution (say, $\mathbf{z}^T$) is not greater than d_s distance away from P_1, we terminate the EMO algorithm and declare $\mathbf{z}^T$ as the final preferred solution. This situation indicates that based on the current value function, there does not exist any solution in the search space which will provide a significantly better solution than P_1. Hence, we can terminate the optimization run. Figure 5 shows such a situation, warranting a termination of the PI-EMO-VF procedure.

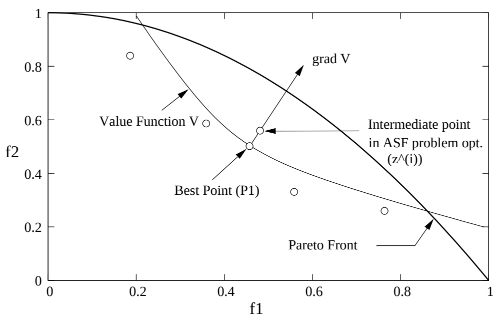

Fig. 4. Local search, when far away from the front, finds a better point more than distance d_s away from the best point. Hence, no termination of the P-EMO.

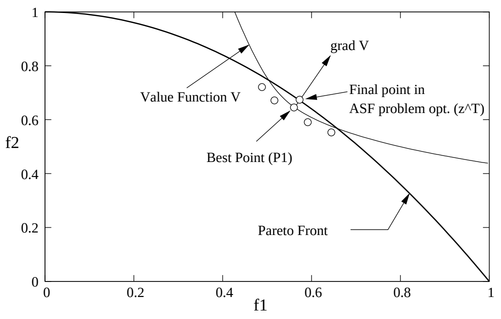

Fig. 5. Local search terminates within distance d_s from the best point. Hence, the P-EMO is terminated.

C. Steps 5 and 6: Modified Domination Principle

The utility function V can also be used to modify the domination principle in order to emphasize and create preferred solutions.

Let us assume that the value function from the most recent decision-making interaction is V. The value function value

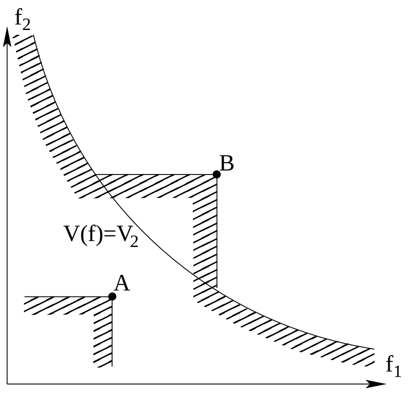

Fig. 6. Dominated regions of two points A and B using the modified definition.

for the second-best member (P_2 defined in the previous subsection) from the set of η points given to the DM is V_2. Then, any two feasible solutions $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ can be compared with their objective function values by using the following modified domination criteria:

1. If both solutions have a value function value *less* than V_2, then the two points are compared based on the usual dominance principle.
2. If both solutions have a value function value *more* than V_2, then the two points are compared based on the usual dominance principle.
3. If one has value function value more than V_2 and the other has value function value less than V_2, then the former dominates the latter.

Figure 6 illustrates regions dominated by points A and B. The value function contour having a value V_2 is shown by the curved line. Point A lies in the region in which the value function is smaller than V_2. The region dominated by point A is shaded. This dominated area is identical to that which can be obtained using the usual domination principle. However, point B lies in the region in which the value function is larger than V_2. For this point, the dominated region is different from that which would be obtained using the usual domination principle. In addition to the usual region of dominance, the dominated region includes all points which have a smaller value function value than V_2.

We now discuss the reason for choosing the baseline value function value at P_2 (as opposed to at P_1) for defining the modified dominance criterion above. While providing preference information on η points given to the DM, the DM has the knowledge of η points. Consider the scenario in Figure 7, in which point $\mathbf{z}^*$ may lie between P_1 and P_2. If the value function at P_1 is considered as the baseline value for domination, the most preferred point $\mathbf{z}^*$ will get dominated by points like P_1. In higher objective problems, the most preferred point may lie elsewhere and considering V_2 may also be too stringent. To be more conservative, $V(P_\eta)$ can be considered as the baseline value in the modified domination criterion.

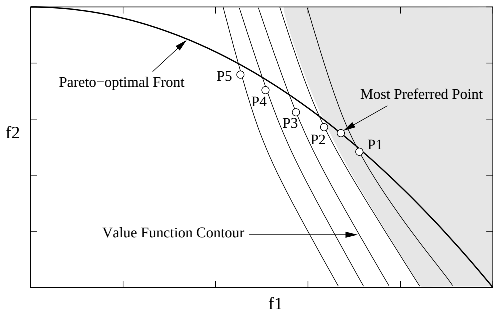

Fig. 7. A scenario in which final preferred point may lie between $P1$ and P_2 for a two-objective problem.

The above modified domination principle can be used in both steps 5 and 6 for creating the new population Off_t and for selecting the new population Par_{t+1}.

Although we do not handle constrained problems in this study, the above modified domination principle can be extended for handling constraints. As defined in [18], when both solutions under consideration for a domination check are *feasible*, the above domination principle can simply be used to establish dominance of one over the other. However, if one point is feasible and the other is not, the feasible solution can be declared as dominating the other. Finally, if both points are infeasible, the one having smaller overall constraint violation may be declared as dominating the other. We defer consideration of a constrained PI-EMO-VF to a later study.

IV. PI-NSGA-II-VF PROCEDURE

In the PI-NSGA-II-VF procedure, the first τ generations are performed according to the usual NSGA-II algorithm [18]. Thereafter, we modify the NSGA-II algorithm by using the modified domination principle (discussed in Section III-C) in the elite-preserving operator and also in the tournament selection for creating the offspring population. We also use a different recombination operator in this study. After a child solution $\mathbf{x}^C$ is created by the SBX (recombination) operator [29], two randomly selected population members $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ are chosen and a small fraction of the difference vector is added to the child solution (similar in principle to a differential evolution operator [30]), as follows:

$$\mathbf{x}^C = \mathbf{x}^C + 0.1\left(\mathbf{x}^{(1)} - \mathbf{x}^{(2)}\right). \tag{9}$$

The crowding distance operator of NSGA-II has been replaced with k-means clustering for maintaining diversity among solutions of the same non-dominated front to make the diversity preservation more meaningful for problems having more than two objectives.

The success of EMO algorithms to find a set of diverse trade-off solutions for two and three-objective problems is due to an appropriate balance between their diversity maintaining operators (for example, crowding distance, clustering, or other mechanisms) and their emphasis of non-dominated solutions [1]. When preference information is to be implemented in an EMO, the search focus has to shift more towards emphasizing currently preferred solutions, as the target becomes finding a single preferred solution at the end. If a proper balance between these *exploring* and *exploiting* mechanisms are not maintained, the resulting preference-based EMO procedure may not work well and may end up either in a premature convergence to a sub-optimal solution or in a random-like search behavior. By modifying the domination principle with preference information, we have emphasized preferred solutions. By using a modified recombination operator for child creation and a clustering operator, instead of crowding distance operator, for a better diversity preservation, we have attempted to make a balance with the enhanced selection pressure towards the preferred solutions. Simulation results of the next section demonstrates this aspect on a number of problems.

The value function optimization problem is solved using the SQP code of KNITRO software [31]. The termination is set if the Karush-Kuhn-Tucker (KKT) error measure computed within KNITRO is less than or equal to 10^{-6}.

For termination check (discussed in Section III-B), we also use the SQP code of KNITRO software and the SQP algorithm is terminated (if not terminated due to d_s distance check from P_1 discussed earlier) when the KKT error measure is less than or equal to 10^{-6}.

V. Results

In this section, we present the results of the PI-NSGA-II-VF procedure on two, three, and five objective test problems. ZDT1 and DTLZ2 test problems are adapted to create maximization problems. In all simulations, we have used the following parameter values:

1) Number of points given to the DM for preference information: $\eta = 5$.
2) Number of generations between two consecutive DM calls: $\tau = 5$.
3) Termination parameter: $d_s = 0.01$.
4) Crossover probability and the distribution index for the SBX operator: $p_c = 0.9$ and $\eta_c = 15$.
5) Mutation probability: $p_m = 0$.
6) Population size: $N = 10M$, where M is the number of objectives.

In the optimization of the VFOP problem (given in equation 5), we restrict the bounds of parameters as follows: $0 \le (k_1, k_2) \le 1000$ and $-1000 \le (l_1, l_2) \le 1000$. In the next section, we perform a parametric study with some of the above parameters. Here, we present the test problems and results obtained with the above setting.

A. Two-Objective Test Problem

Problem 1 is adapted from ZDT1 and has 30 variables [32].

$$\begin{array}{rl} \text{Maximize} & \mathbf{f}(\mathbf{x}) = \left\{ \begin{array}{c} x_1 \\ \frac{10-\sqrt{x_1 g(\mathbf{x})}}{g(\mathbf{x})} \end{array} \right\}, \\ \text{where} & g(\mathbf{x}) = 1 + \frac{9}{29}\sum_{i=2}^{30} x_i, \\ & 0 \le x_i \le 1, \quad \text{for } i = 1, 2, \ldots, 30, \end{array} \tag{10}$$

The Pareto-optimal front is given by $f_2 = 10 - \sqrt{f_1}$ and is shown in Figure 8. The solutions are $\mathbf{x}_i = 0$ for $i = 2, 3, \ldots, 30$ and $x_1 \in [0, 1]$.

This maximization problem has a non-convex front, therefore if the decision maker is not interested in the end points, the value function has to be non-linear. A linear value function will always lead to the end points of the front. In our simulations, we assume a particular value function which acts as a representative of the DM, but the information is not explicitly used in creating new solutions by the operators of the PI-NSGA-II-VF procedure. In such cases, the most preferred point $\mathbf{z}^*$ can be determined from the chosen value function beforehand, thereby enabling us to compare our obtained point with $\mathbf{z}^*$.

In our study, we assume the following non-linear value function (which acts as a DM in providing a complete ranking of η solutions at every τ generations):

$$V(f_1, f_2) = \frac{1}{(f_1 - 0.35)^2 + (f_2 - 9.6)^2}. \tag{11}$$

This value function gives the most preferred solution as $\mathbf{z}^* = (0.25, 9.50)$. The contours of this value function are shown in Figure 8. Since a DM-emulated value function is used to decide on preference of one point to the other in pairwise comparisons, we shall have complete ranking information of all η points in our study. Thus, we shall not have the fourth set of constraints in determining the value function, as given in equation 5. In a future study, we shall consider partial preference information and its effect on the constructed value function.

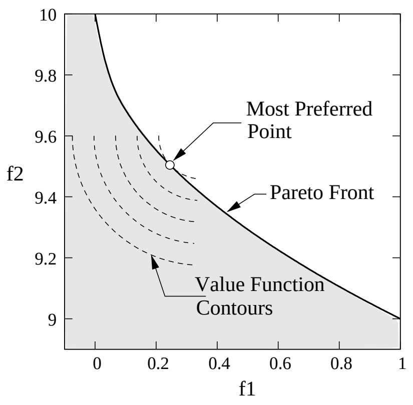

Fig. 8. Contours of the chosen value function (acts as a DM) and the most preferred point corresponding to the value function.

Table I presents the best, median and worst of 21 different PI-NSGA-II-VF simulations (each starting with a different initial population). The performance (accuracy measure) is computed based on the Euclidean distance of each optimized point with $\mathbf{z}^*$. Note that this accuracy measure is different from the termination criterion used in the PI-NSGA-II-VF procedure. Table II shows minimum, median and maximum accuracy, the number of overall function evaluations, and

TABLE I

FINAL SOLUTIONS OBTAINED BY PI-NSGA-II-VF FOR THE MODIFIED ZDT1 PROBLEM.

	$\mathbf{z}^*$	Best	Median	Worst
f_1	0.2500	0.2498	0.2461	0.2713
f_2	9.5000	9.5002	9.5038	9.4791

the number of DM calls recorded in the 21 runs. The table indicates that the proposed PI-NSGA-II-VF procedure is able to find a solution close to the final preferred solution. Although the overall number of function evaluations depend on the initial population, for a 30-variable problem these numbers are reasonable.

TABLE II

DISTANCE OF OBTAINED SOLUTION FROM THE MOST PREFERRED SOLUTION, FUNCTION EVALUATIONS, AND THE NUMBER OF DM CALLS REQUIRED BY THE PI-NSGA-II-VF FOR THE MODIFIED ZDT1 PROBLEM.

	Minimum	Median	Maximum
Accuracy	0.0001	0.0062	0.0197
Func. Evals.	5,408	7,372	11,809
# of DM Calls	14	19	30

We now show the working of the PI-NSGA-II-VF approach for a particular run, which required 14 DM calls before termination. Figure 9 shows the value functions optimized after various DM calls. The first DM call was made after generation 5. Five chosen points (P1 to P5 shown in shaded circles) from the non-dominated solutions at generation 5 are shown in the figure. The best and second-best points are close to each other. The strictly increasing requirement of the value function imposed in the optimization process creates an almost linear value function as an optimum choice in this case. The corresponding parameter values of the value function are: ($k_1 = 998.189$, $k_2 = 0.049$, $l_1 = 369.532$, and $l_2 = 137.170$). The value functions are drawn at the second-best point. After five more generations, the DM is called to provide preference information the second time. The corresponding value function drawn at the second-best point is shown in the figure. Five points used for preference ranking are shown as diamonds. The figure shows how the PI-NSGA-II-VF procedure finds better and better points and how progressively the DM calls enable the overall procedure to find refined value functions. Eventually, at the 14th DM call, all five solutions come very close to $\mathbf{z}^*$ and the algorithm terminates with the imposed $d_s = 0.01$ condition. The optimal parameter values fixing the value functions at various DM calls are shown in Table III. Although no pattern in these parameter values is observed from one DM call to another, every value function thus obtained is strictly increasing and maximizes the maximum difference in value function values between any two chosen points. However, the NSGA-II algorithm with these value functions in five subsequent generations seems to guide the best point towards the most preferred point ($\mathbf{z}^*$) progressively.

Figure 10 shows the value functions from the 10th DM call onwards. In this figure, the value functions are drawn

TABLE III

OPTIMAL PARAMETER VALUES DETERMINING THE VALUE FUNCTION AND CORRESPONDING BEST POINT AT VARIOUS DM CALLS.

DM Call	k_1	k_2	l_1	l_2	$P_1 = (f_1, f_2)$
#1	998.189	0.049	369.532	137.170	(0.223, 2.600)
#2	999.998	19.699	114.161	359.199	(0.078, 2.898)
#3	821.797	0.003	-15.116	770.050	(0.260, 4.321)
#4	1000.000	440.133	87.366	393.896	(0.282, 4.706)
#6	804.650	0.033	-99.871	567.481	(0.332, 6.525)
#8	807.395	105.691	-30.880	365.454	(0.344, 8.066)
#10	403.750	49.007	-30.667	290.960	(0.254, 9.259)
#14	0.007	0.006	-0.308	-9.488	(0.251, 9.499)

at the second-best point (shown with a diamond) and the corresponding best point is also shown by a shaded circle. It is interesting to observe how the value functions get modified with generations and how the modified value functions help find better non-dominated solutions progressively with the help of modified domination and NSGA-II operators. The final point obtained by the PI-NSGA-II-VF is $(f_1, f_2) = (0.251, 9.499)$, which is very close to the most preferred point $\mathbf{z}^* = (0.25, 9.5)$ corresponding to the optimum of the DM-emulated value function given in equation 11.

B. Three-Objective Test Problem

The DTLZ2 test problem [33] is scalable to number of objectives. In the three-objective case, all points (objective vectors) are bounded by two spherical surfaces in the first octant. In the case of minimizing all objectives, the inner surface (close to the origin) becomes the Pareto-optimal front. But here, we maximize each objective of the DTLZ2 problem. Thus, the outer spherical surface becomes the corresponding Pareto-optimal front. An M-objective DTLZ2 problem for maximization is given as follows:

$$\begin{array}{l}\text{Maximize} \quad \mathbf{f}(\mathbf{x}) = \\ \left\{\begin{array}{l} (1.0+g(\mathbf{x}))\cos(\frac{\pi}{2}x_1)\cos(\frac{\pi}{2}x_2)\cdots\cos(\frac{\pi}{2}x_{M-1}) \\ (1.0+g(\mathbf{x}))\cos(\frac{\pi}{2}x_1)\cos(\frac{\pi}{2}x_2)\cdots\sin(\frac{\pi}{2}x_{M-1}) \\ \vdots \\ (1.0+g(\mathbf{x}))\cos(\frac{\pi}{2}x_1)\sin(\frac{\pi}{2}x_2) \\ (1.0+g(\mathbf{x}))\sin(\frac{\pi}{2}x_1) \end{array}\right\}, \\ \text{subject to} \quad 0 \le x_i \le 1, \quad \text{for } i = 1, \ldots, 12, \\ \quad\quad \text{where} \quad g(\mathbf{x}) = \sum_{i=3}^{12}(x_i - 0.5)^2. \end{array} \tag{12}$$

The Pareto-optimal front for a three-objective DTLZ2 problem is shown in Figure 11. The points (objective vectors) on the Pareto-optimal front follow the relation: $f_1^2 + f_2^2 + f_3^2 = 3.5^2$. The decision variable values correspond to $x_1 \in [0,1]$, $x_2 \in [0,1]$ and $x_i = 0$ or 1 for $i = 3, 4, \ldots, 12$.

To test the working of PI-NSGA-II-VF on this problem, we have replaced the decision maker by using a linear value function (emulating the DM), as follows:

$$V(f_1, f_2, f_3) = 1.25f_1 + 1.50f_2 + 2.9047f_3. \tag{13}$$

This value function produces the most preferred solution on the Pareto-optimal front as $\mathbf{z}^* = (1.25, 1.50, 2.9047)$.

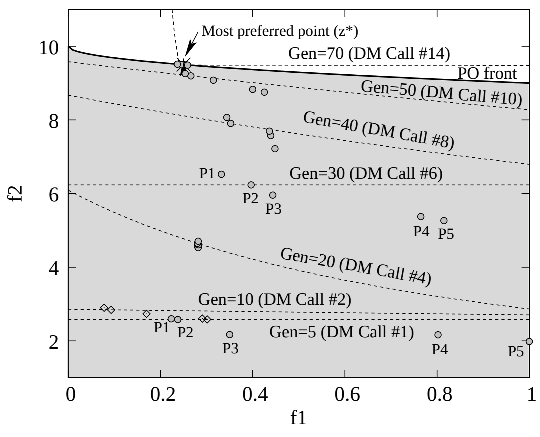

Fig. 9. Evolution of value functions after successive DM calls.

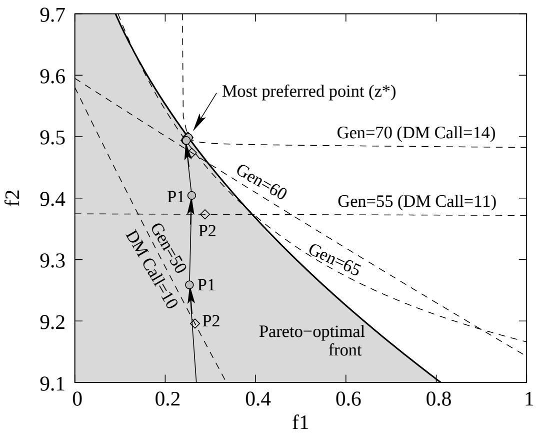

Fig. 10. Value functions near the most preferred point.

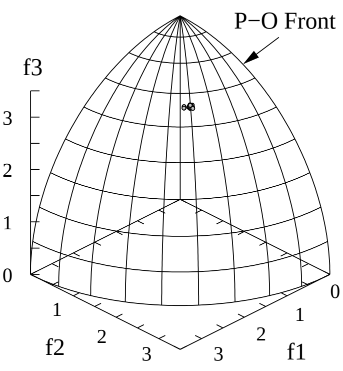

Fig. 11. Final population members after termination of the algorithm for three-objective modified DTLZ2 problem. The complete Pareto-optimal surface is marked as ‘P-O front’.

The PI-NSGA-II-VF is run with $N = 10 \times 3$ or 30 population members 21 times, each time with a different random initial population. In terms of the accuracy measure from $\mathbf{z}^*$, Table IV presents the minimum, median and worst performing runs. Table V shows the accuracy, number of overall function evaluations and number of DM calls needed by the procedure. It is clear that the obtained points are close to the most preferred point $\mathbf{z}^*$. Figure 11 shows the population at the final generation of a typical PI-NSGA-II-VF run.

TABLE IV
FINAL SOLUTIONS OBTAINED BY PI-NSGA-II-VF FOR THE THREE-OBJECTIVE MODIFIED DTLZ2 PROBLEM.

	$\mathbf{z}^*$	Best	Median	Worst
f_1	1.2500	1.2459	1.2197	1.3178
f_2	1.5000	1.5050	1.4888	1.4755
f_3	2.9047	2.9039	2.9233	2.8873

TABLE V
DISTANCE OF OBTAINED SOLUTION FROM THE MOST PREFERRED SOLUTION, NUMBER OF FUNCTION EVALUATIONS, AND NUMBER OF DM CALLS REQUIRED BY PI-NSGA-II-VF ON THE THREE-OBJECTIVE MODIFIED DTLZ2 PROBLEM.

	Minimum	Median	Maximum
Accuracy	0.0008	0.0115	0.0434
Func. Evals.	4,200	6,222	8,982
# of DM Calls	17	25	36

C. Five-Objective Test Problem

We now consider the five-objective ($M = 5$) version of the DTLZ2 problem described in the previous subsection. The Pareto-optimal front is described as $f_1^2 + f_2^2 + f_3^2 + f_4^2 + f_5^2 = 3.5^2$.

For this problem, we choose a non-linear DM-emulated value function, as follows:

$$V(\mathbf{f}) = 1/\sum_{i=1}^{5}(f_i - a_i)^2, \qquad (14)$$

where $\mathbf{a} = (1.1, 1.21, 1.43, 1.76, 2.6468)^T$. This value function produces the most preferred point as $\mathbf{z}^* = (1.0, 1.1, 1.3, 1.6, 2.4062)$.

Table VI presents the obtained solutions by PI-NSGA-II-VF with 50 population members. Table VII shows the

TABLE VI
FINAL OBJECTIVE VALUES OBTAINED FROM PI-NSGA-II-VF FOR THE FIVE-OBJECTIVE MODIFIED DTLZ2 PROBLEM.

	$\mathbf{z}^*$	Best	Median	Worst
f_1	1.0000	0.9931	0.9785	0.9455
f_2	1.1000	1.1382	1.0502	1.1467
f_3	1.3000	1.3005	1.3382	1.3208
f_4	1.6000	1.5855	1.5947	1.6349
f_5	2.4062	2.4007	2.4199	2.3714

accuracy measure, the number of overall function evaluations, and the number of DM calls. Although the points close to the most preferred point are obtained in each run, the higher dimensionality of the problem requires more function evaluations and DM calls compared to two and three-objective test problems. However, the above results are obtained for a strict termination criterion with $d_s = 0.01$. Smaller number of DM calls and evaluations are expected if this termination criterion is relaxed. We discuss these matters in the next section. It is worth mentioning that the application of a usual EMO (including the original NSGA-II) is reported to face difficulties in converging to the *entire* five-dimensional Pareto-optimal front with an identical number of function evaluations in an earlier study [34]. However, since our target is one particular preferred point on the Pareto-optimal front (dictated by a sequence of preference information provided by the DM), our proposed PI-EMO-VF is able to find a near Pareto-optimal solution for the same five-objective optimization problem. This ability of an EMO to handle problems with a large number of objectives demonstrated in this study remains as an evidence of an advantage of using preference information within an EMO.

TABLE VII
DISTANCE OF OBTAINED SOLUTION FROM THE MOST PREFERRED SOLUTION, FUNCTION EVALUATIONS, AND THE NUMBER OF DM CALLS REQUIRED BY PI-NSGA-II-VF FOR THE FIVE-OBJECTIVE MODIFIED DTLZ2 PROBLEM.

	minimum	median	maximum
Accuracy	0.0084	0.0240	0.0902
# of Function Eval.	23,126	27,202	41,871
# of DM Calls	57	67	102

D. Five-Objective Test Problem with an Intermediate Change in Decision

Next, we use the same five-objective DTLZ2 problem, but simulate a rather realistic scenario in which the DM changes his/her preference information while the optimization process is on. We use three DM-emulated functions of type given in equation 14 with the parameters shown in Table VIII. The first DM-emulated function is used to provide preference information for the first 10 DM calls, thereafter for some reason the DM changes his/her mind and the second function is used to provide preference information. Again, after 10 more DM calls, a third DM-emulated function is used. The parameters are quite different from each other, meaning that the corresponding Pareto-optimal solutions lie on different parts of the Pareto-optimal front (shown in the table). Such a simulation will demonstrate the flexibility of the proposed procedure in its ability to find the Pareto-optimal solutions corresponding to the third DM-emulated function, although being distracted by the initial influence of different functions.

TABLE VIII
PARAMETERS OF VALUE FUNCTIONS AND CORRESPONDING DM CALLS IN WHICH THEY ARE USED. CORRESPONDING PARETO-OPTIMAL SOLUTIONS ARE SHOWN.

DM calls	$\mathbf{a}$	Corresponding P-O Solution
1–10	$(2.75, 1.65, 1.10, 1.32, 1.26)$	$(2.50, 1.50, 1.00, 1.20, 1.14)$
11–20	$(1.87, 1.54, 1.82, 1.16, 3.18)$	$(1.70, 1.40, 1.65, 1.05, 2.89)$
21–terminate	$(1.10, 1.21, 1.43, 1.76, 2.65)$	$(1.00, 1.10, 1.30, 1.60, 2.40)$

We use an identical parameter setting as used in the previous problem. Figure 12 shows the value of the three DM-emulated functions for the most preferred solution after every DM call. It is clear from the figure that during the first 10 DM calls, the corresponding DM-emulated function value for the most preferred solution steadily increases, as if the DM were interested in maximizing this value function. The fact that the first DM-emulated function value after the 11th call reduces, indicates that the algorithm is able to respond to the new function. Thereafter, again after 21st DM call, the third DM-emulated function is active and the preferred solutions get tuned to this new value function. Eventually, the proposed method is able to converge to the Pareto-optimal solution corresponding to the third DM-emulated function.

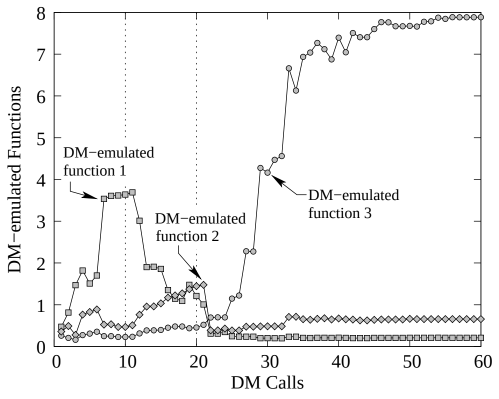

Fig. 12. Values of three DM-emulated functions corresponding to the most preferred solution in each DM call.

VI. PARAMETRIC STUDY

Besides the usual parameters associated with an evolutionary algorithm, such as population size, crossover and mutation probabilities and indices, tournament size etc., in the proposed PI-NSGA-II-VF we have introduced a few additional parameters which may effect the accuracy and number of DM calls. They are the number of points used in obtaining DM's preference information (η), the number of generations between DM calls (τ), termination parameter (d_s), KKT error limit for terminating SQP algorithm in value function optimization and in single-objective optimization used for the termination check, and the parameter ρ used in the ASF function optimization. Of these parameters, the first three have shown to have an effect on the chosen performance measures — accuracy, the number of overall function evaluations, and the number of

DM calls. As mentioned earlier, the parameter η is directly related to the maximum number of pairwise comparisons a DM would like to do in a single DM call. Of course, if more points can be compared, a more appropriate value function can be obtained. However, based on a maximum of 10 pairwise comparisons per DM call, we restrict $\eta = 5$ in this study and do not do a parametric study with this parameter. Thus, in this section, we study the effect of two parameters (τ and d_s), while keeping the other PI-NSGA-II-VF parameters identical to that mentioned in the previous section. In each case, we use the same three test problems.

A. Effect of Frequency of DM Calls (τ)

First we study the effect of τ by considering four different values: 2, 5, 10 and 20 generations. The parameter d_s is kept fixed to 0.01. To investigate the dependence of the performance of the procedure on the initial population, in each case, we run PI-NSGA-II-VF from 21 different initial random populations and plot the best, median and worst performance measures.

We plot three different performance measures — accuracy, number of DM calls and number of function evaluations obtained for the modified ZDT1 problem in Figure 13. It is

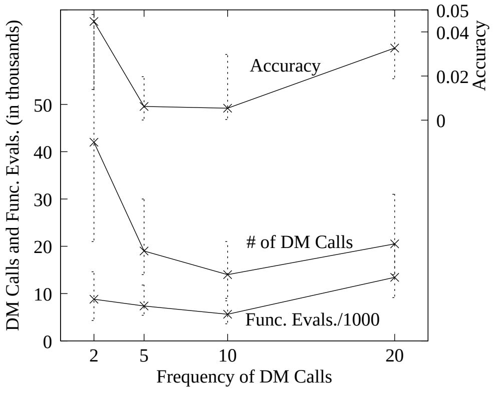

Fig. 13. Three performance measures on modified ZDT1 problem for different τ values.

interesting to note that all three median performance measures are best for $\tau = 10$, although $\tau = 5$ also results in a similar accuracy and the number of DM calls. A small value of τ means that DM calls are to be made more frequently. Clearly, this results in higher number of DM calls, as evident from the figure. Frequent DM calls result in more single-objective optimization runs for termination check, thereby increasing the number of overall function evaluations. On the other hand, a large value of τ captures too little preference information to focus the search near the most preferred point, thereby causing a large number of generations to satisfy termination conditions and a large number of DM calls.

Figure 14 shows the same three performance measures on the three-objective modified DTLZ2 problem. For this problem, the number of DM calls is similar for $\tau = 5$ and 10 generations, whereas accuracy and the number of function evaluations are better for $\tau = 5$ generations. Once again, too small or too large τ is found to be detrimental.

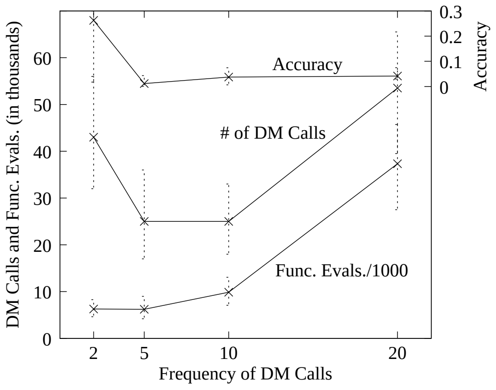

Fig. 14. Three performance measures on three-objective modified DTLZ2 problem for different τ values.

For the five-objective modified DTLZ2 problem, $\tau = 5$ produces optimal median performance on the number of DM calls and accuracy (Figure 15). However, the overall function evaluations is smaller with smaller τ.

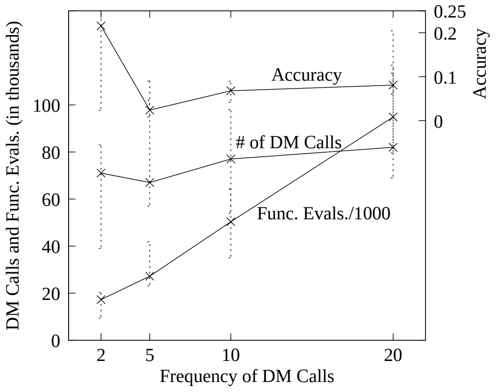

Fig. 15. Three performance measures on five-objective modified DTLZ2 problem for different τ values.

Based on these simulation studies on two, three and five-objective optimization problems, one can conclude that a value of τ within 5 to 10 generations is better in terms of an overall performance of the PI-NSGA-II-VF procedure. This range of τ provides a good convergence accuracy, requires less function evaluations, and less DM calls to converge near the most preferred point.

B. Effect of Termination Parameter d_s

Next, we investigate the effect of the termination parameter d_s on the three performance measures on all three problems. In this study, we fix $\eta = 5$ and $\tau = 5$. Figure 16 shows the positive correlation between accuracy and d_s. As d_s is increased (meaning a relaxed termination), the obtained accuracy (distance from $\mathbf{z}^*$) gets worse. Interestingly, the associated variation in obtained accuracy over number of runs also gets worse. The flip side of increasing d_s is that the number of function evaluations reduces, as a comparatively lesser number of generations are now needed to satisfy the termination condition. Similarly, the number of DM calls also reduces with an increase in d_s.

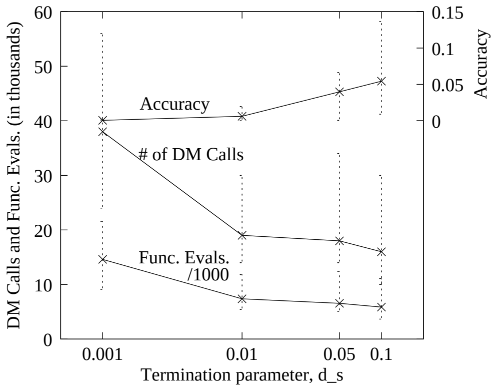

Fig. 16. Three performance measures on modified ZDT1 problem for different d_s values.

Similar observations are made for three-objective and five-objective modified DTLZ2 problem, as evident from Figures 17 and 18, respectively.

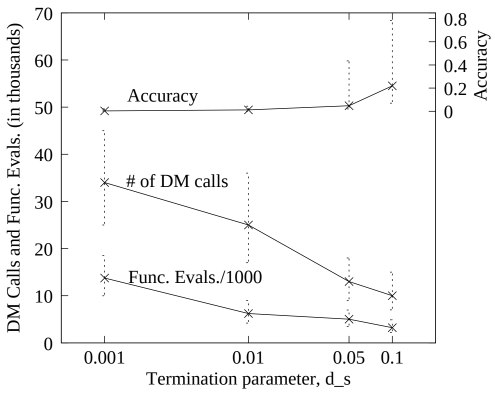

Fig. 17. Three performance measures on three-objective modified DTLZ2 problem for different d_s values.

These results clearly reveal the behavior of our proposed algorithm with regard to the choice of d_s. Unlike in the parametric study of τ, we find a monotonic variation in performance measures with d_s, however with a trade-off between accuracy and the number of DM calls (or, the number of function evaluations). This indicates that d_s need not be

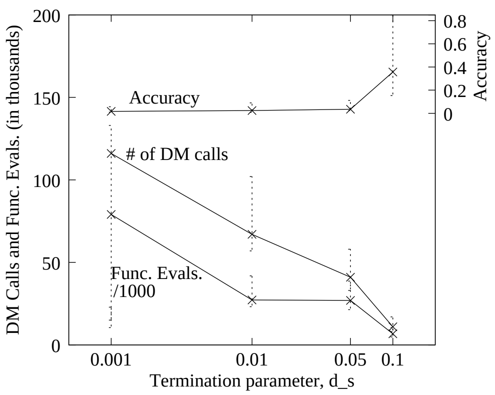

Fig. 18. Three performance measures on modified five-objective modified DTLZ2 problem for different d_s values.

chosen as an arbitrarily small value. If approximate solutions are acceptable, they can be achieved with a smaller number of function evaluations and DM calls. Figure 19 shows the trade-off of these quantities for the modified three and five-objective DTLZ2 problems. The nature of the trade-off between accuracy and the number of DM calls indicates that $d_s = 0.05$ makes a good compromise between these performance indicators for these problems. A smaller d_s setting calls for substantially more DM calls, and a larger d_s setting, although reduces the number of DM calls, makes a substantially large deviation from the most preferred solution.

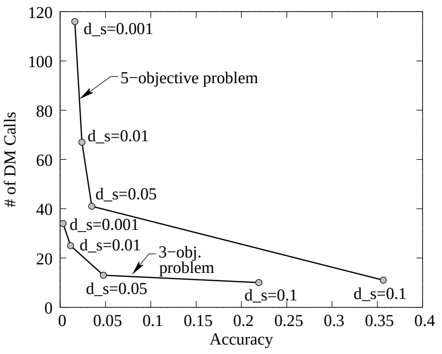

Fig. 19. Trade-off between accuracy and the number of DM calls for the modified three and five-objective DTLZ problems.

VII. Random Error in Preference Information

In the above simulations, a mathematical value function is used to emulate the preference information to be given by a DM. However, in practice, the DM is a human being. There is bound to be some level of inconsistencies in providing preference information. To simulate the effect of this factor, we consider a DM-emulating value function which is stochastic in nature.

A linear value function similar to the one used before is chosen, but the coefficients of the value function are made stochastic. The stochasticity is reduced with the increase in the number of generations. This has been done to simulate a realistic DM who is likely to make errors during the start of the algorithm when he/she is in the process of learning his/her preferences. Later, the decision maker is likely to make more consistent decisions. A successful convergence of an algorithm in this case verifies that the algorithm does not get misdirected by inconsistent preference based information during the beginning of the run.

The DM-emulated value function used for the three-objective modified DTLZ2 problem is as follows:

$$\begin{aligned} V(f_1,f_2,f_3) &= \text{noise}(1.25,\sigma)f_1 + \text{noise}(1.50,\sigma)f_2 \\ &\quad +\text{noise}(2.9047,\sigma)f_3, \end{aligned} \tag{15}$$

where σ is set as $\exp(-t/10)$ (t is the generation counter) and $\text{noise}(\mu,\sigma)$ refers to a random normal distribution with a mean μ and standard deviation σ. This setting ensures that the standard deviation of the noise around the mean reduces as the number of generations of the algorithm increases. With $\sigma = 0$, this value function gives the most preferred point as $\mathbf{z}^* = (1.25, 1.50, 2.9047)$. At the first instance of DM calls (that is, at $t = \tau = 5$ generations), $\sigma = \exp(-0.5) = 0.606$, meaning a significantly different value function than what is required for the algorithm to converge to the most preferred point.

Table IX shows the best, median and worst points obtained by the PI-NSGA-II-VF procedure with $\eta = 5$, $\tau = 5$, $d_s = 0.01$ and other parameter values used in Section V-B. Again, 21 different runs were performed from different initial random populations. As clearly shown in Table X, the accuracy for the best and median runs is good, despite the large stochasticities involved in the early stages of the optimization process. Although the number of function evaluations and the number of DM calls are 20 to 40% more compared to that in the deterministic DM-emulated value function case (Table IV), the accuracy of the final point is good. This indicates that the final point is close to the most preferred solution for the deterministic case.

TABLE IX
FINAL SOLUTIONS OBTAINED BY PI-NSGA-II-VF FOR THE THREE-OBJECTIVE MODIFIED DTLZ2 PROBLEM WITH A STOCHASTIC DM-EMULATED VALUE FUNCTION.

	$\mathbf{z}^*$	Best	Median	Worst
f_1	1.2500	1.2555	1.2695	1.2902
f_2	1.5000	1.5105	1.5205	1.6437
f_3	2.9047	2.8969	2.8856	2.8078

TABLE X
DISTANCE OF OBTAINED SOLUTION FROM THE MOST PREFERRED SOLUTION, FUNCTION EVALUATIONS, AND THE NUMBER OF DM CALLS BY PI-NSGA-II-VF FOR THE THREE-OBJECTIVE MODIFIED DTLZ2 PROBLEM WITH A STOCHASTIC DM-EMULATED VALUE FUNCTION.

	Minimum	Median	Maximum
Accuracy	0.0142	0.0342	0.1779
Func. Evals.	5,841	7,608	9,663
# of DM Calls	24	31	39

Next, we apply the PI-NSGA-II-VF procedure to the five-objective modified DTLZ2 problem with the following stochastic value function to emulate the DM:

$$\begin{aligned} V(\mathbf{f}) &= \text{noise}(1.0,\sigma)f_1 + \text{noise}(1.1,\sigma)f_2 + \text{noise}(1.3,\sigma)f_3 \\ &\quad +\text{noise}(1.6,\sigma)f_4 + \text{noise}(2.4062,\sigma)f_5. \end{aligned} \tag{16}$$

The best, median, and worst points obtained by PI-NSGA-II-VF are shown in Table XI. As shown by the performance measures in Table XII, despite somewhat larger function evaluations and number of DM calls, final points obtained by PI-NSGA-II-VF are reasonably close to the most preferred point obtained for the deterministic version of the DM-emulated value function.

TABLE XI
FINAL SOLUTIONS OBTAINED BY PI-NSGA-II-VF FOR THE FIVE-OBJECTIVE MODIFIED DTLZ2 PROBLEM WITH A STOCHASTIC DM-EMULATED VALUE FUNCTION.

	$\mathbf{z}^*$	Best	Median	Worst
f_1	1.0000	1.0103	1.0875	1.0557
f_2	1.1000	1.1171	1.1495	1.2394
f_3	1.3000	1.3037	1.3525	1.4915
f_4	1.6000	1.6025	1.5942	1.4977
f_5	2.4062	2.4140	2.4125	2.4889

TABLE XII
DISTANCE OF OBTAINED SOLUTION FROM THE MOST PREFERRED SOLUTION, FUNCTION EVALUATIONS, AND THE NUMBER OF DM CALLS BY PI-NSGA-II-VF FOR THE FIVE-OBJECTIVE MODIFIED DTLZ2 PROBLEM WITH A STOCHASTIC DM-EMULATED VALUE FUNCTION.

	Minimum	Median	Maximum
Accuracy	0.0219	0.1137	0.2766
Func. Evals.	33,653	39,264	52,564
# of DM Calls	72	87	136

1) Effect of Extent of Stochasticity: In the above study, we used a noise factor on the coefficients of the DM-emulated value function, given as a function of generation counter t as follows: $\sigma = \exp(-t/10)$. As discussed above, at the first DM call with $\tau = 5$, this has an effect of having a standard deviation of 0.606 on each objective. We now investigate the effect of increasing the standard deviation by modifying the σ term as follows:

$$\sigma = s\exp(-t/10), \tag{17}$$

where s is a stochasticity factor. For $s = 1$, we have an identical stochastic effect as in the previous subsection. By using a larger value of s, we can simulate a situation with more inconsistencies in the decision-making process. We use four different values of s: 1, 5, 10 and 100.

With a large value of s, it is expected that the DM-emulated value function provides a different ranking of η points than an ideal ranking (which would have been obtained without the stochasticity effect). We count the number of times the ranking of top three points is different from the ideal ranking of the same three points and tabulate it in Table XIII for a typical run. The corresponding function evaluations and accuracy in the final optimized point from $\mathbf{z}^*$ are also shown in the table. An increase in stochasticity in the decision-making process requires more DM calls and more function evaluations to achieve an acceptable accuracy and termination. Importantly, since all runs are terminated with $d_s = 0.01$ condition, despite large stochasticities involved in the beginning of PI-NSGA-II-VF runs, the algorithm is able to find a point close to the most preferred Pareto-optimal point corresponding to the deterministic version of the DM-emulated value function.

TABLE XIII
EFFECT OF STOCHASTIC FACTOR ON PERFORMANCE MEASURES FOR THREE-OBJECTIVE MODIFIED DTLZ2 PROBLEM.

s	Incorrect ranking	Total DM Calls	Func. Evals.	Accuracy
1	12	20	6,786	0.0399
5	17	23	7,528	0.0437
10	19	28	8,572	0.0498
100	23	31	9,176	0.0512

VIII. Extensions of Current Study

This study has suggested a simple yet elegant methodology by which the DM's preferences can be incorporated with an EMO algorithm so that the final target is not a complete Pareto-optimal set (as is usual in an EMO application), but a single preferred solution on the Pareto-optimal set. The ideas suggested can be extended in a number of different ways, which we discuss in the following paragraphs.

- *Incomparable class and constrained problems:* In this study, we have not simulated the case in which the DM judges some of the η points to be incomparable. Although our optimization problem formulation (equation 5) considers this situation and we have demonstrated its use in constructing the respective value function in Figure 3, a complete study is needed to implement the idea with the proposed PI-EMO-VF procedure and particularly in analyzing the effect of δ_V in the development of the value function. Since some cases may occur, in which a value function satisfying all DM's preferences is not possible, this study will also test the specific part (Step 3) of the proposed PI-EMO-VF algorithm.
  Moreover, we have not tested constrained optimization problems in this study. The modified constrained domination principle should be used and tested on some challenging problems.
- *Other value functions:* In this study, we have restricted the value function to be of certain form (equation 5). Other more generic value function structures can also be considered. Our suggested value function construction procedure results in strictly increasing functions. However, a more generic non-concave value function may be obtained by using different conditions in the optimization problem formulation.
- *Robust value functions:* The optimization problem for deriving the value function can include a robustness consideration, in which the insensitivity of the value function coefficients in producing an identical ranking of η points can be ensured. This would be a different way of handling inconsistencies in decision-making.
- *Other termination conditions:* Our proposed PI-EMO-VF algorithm terminates when there does not exist a far away point with a better value function value than the currently judged preferred point. Although this indicates somewhat the probability of creating better preferred points than the currently judged preferred point, other termination indicators are certainly possible. In this direction, instead of terminating based on Euclidean distance between the two points, the difference in value function values can be checked.
- *Reduction in DM calls:* One outcome of the parametric study is that by fixing a relaxed termination criterion (relatively larger value of d_s), the number of DM calls can be reduced. However, there are other extensions to this study which may also reduce the number of DM calls. The basic operators in the suggested algorithm can be extended so that the modified procedure requires a reduced number of overall DM calls. The issue of having more points in each DM call, thereby reducing the overall number of DM calls to achieve a comparable accuracy will constitute an important study. Instead of keeping a fixed interval of τ generations for each DM call, DM call interval can be varied (or self-adapted) based on the extent of improvement achieved from the previous value function. Varying the number of points (η) in each DM call in a self-adaptive manner would be another important task. Since the points early on in the PI-EMO-VF procedure are not expected to be close to the Pareto-optimal front, the number of DM calls and points per call can be made small. Thereafter, when the procedure approaches the Pareto-optimal front, more points can be included per DM call and the frequency of DM calls can be controlled by the observed rate of improvement of the performance of the procedure. Also, it would be an interesting study to ascertain the effect of cumulating the preference information from one decision call to the next and use it in approximating the value function.
- *Fixed budget of DM calls:* In this study, we have kept a termination criterion which is related to the extent of improvements in currently judged preferred solution. We then recorded the number of DM calls which were needed until the termination criteria was met. However, a comparative study, in which different algorithms are compared for a fixed number of DM calls may be performed.
- *Value function based recombination and mutation operators:* In this study, we have modified the domination principles to emphasize points which have better value function value. However, EMO algorithm's recombination and mutation operators can also be modified based

on developed value function. For example, restricting one of the top two currently judged preferred solutions as one parent in the SBX operator may help generate better preferred solutions.
- *PI-EMO-VF with other EMO algorithms:* In this study, we have integrated the preference information in NSGA-II algorithm. A natural extension of this study would be to incorporate the preference handling approach with other popular EMO methodologies, such as SPEA2 [23], PESA [35], and others.

IX. Conclusions

In this paper, we have suggested a simple preference based evolutionary multi-objective optimization (PI-EMO-VF) procedure, which iteratively finds new solutions by using an EMO algorithm that progressively sends a representative set of trade-off solutions to a DM for obtaining a complete or partial preference ranking. DM's preference information has been used in the following three ways in developing the new algorithm:

- Firstly, a strictly increasing value function is derived by solving an optimization problem, which maximizes the value function value between ranked points.
- Secondly, the resulting value function is then utilized to redefine the domination principle between the points. The modified domination principle is used to drive the EMO search.
- Thirdly, the resulting value function is used to design a termination criterion for the PI-EMO-VF algorithm by executing a single-objective search along the gradient direction of the value function.

The above generic preference based EMO approach has been implemented with the NSGA-II procedure. The PI-NSGA-II-VF procedure has then been applied to three different test-problems involving two, three and five objectives. By using a DM-emulated utility function, we have shown that the PI-NSGA-II-VF is capable of finding the most preferred solution corresponding to the emulated utility function. A parametric study on the additional parameters has clearly indicated optimal parameter settings. Finally, to simulate inconsistencies, which may arise in providing preference information was simulated by considering a stochastic value function with a noise effect reducing over time. Even in such cases, the PI-NSGA-II-VF has been able to come closer to the most preferred point corresponding to the deterministic version of the DM-emulated value function.

Combining the ideas from EMO algorithms and multiple criteria decision making (MCDM) seems an encouraging direction for future research in multi-objective optimization. In this paper, we have suggested one particular integration of DM's direct preference information into an EMO algorithm. The method is generic and the obtained results indicate that it is a promising approach. More emphasis must now be placed for developing pragmatic hybrid algorithms for multi-objective optimization and decision-making.

Acknowledgments

Kalyanmoy Deb wishes to thank the Academy of Finland and the Foundation of the Helsinki School of Economics for their support under the FiDiPro programme. Pekka Korhonen and Jyrki Wallenius also wish to acknowledge the financial support from the Academy of Finland (Grant No. 12198).

References

[1] K. Deb, *Multi-objective optimization using evolutionary algorithms*. Chichester, UK: Wiley, 2001.

[2] C. A. C. Coello, D. A. VanVeldhuizen, and G. Lamont, *Evolutionary Algorithms for Solving Multi-Objective Problems*. Boston, MA: Kluwer, 2002.

[3] K. Deb and D. Saxena, "Searching for Pareto-optimal solutions through dimensionality reduction for certain large-dimensional multi-objective optimization problems," in *Proceedings of the World Congress on Computational Intelligence (WCCI-2006)*, 2006, pp. 3352–3360.

[4] J. Knowles and D. Corne, "Quantifying the effects of objective space dimension in evolutionary multiobjective optimization," in *Proceedings of the Fourth International Conference on Evolutionary Multi-Criterion Optimization (EMO-2007)*, 2007, pp. 757–771, (LNCS 4403).

[5] J. Branke, T. Kauβler, and H. Schmeck, "Guidance in evolutionary multi-objective optimization," *Advances in Engineering Software*, vol. 32, pp. 499–507, 2001.

[6] J. Branke and K. Deb, "Integrating user preferences into evolutionary multi-objective optimization," in *Knowledge Incorporation in Evolutionary Computation*, Y. Jin, Ed. Heidelberg, Germany: Springer, 2004, pp. 461–477.

[7] K. Deb, J. Sundar, N. Uday, and S. Chaudhuri, "Reference point based multi-objective optimization using evolutionary algorithms," *International Journal of Computational Intelligence Research (IJCIR)*, vol. 2, no. 6, pp. 273–286, 2006.

[8] L. Thiele, K. Miettinen, P. Korhonen, and J. Molina, "A preference-based interactive evolutionary algorithm for multi-objective optimization," *Evolutionary Computation Journal*, vol. 17, no. 3, pp. 411–436, 2009.

[9] K. Deb and A. Kumar, "Interactive evolutionary multi-objective optimization and decision-making using reference direction method," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2007)*. New York: The Association of Computing Machinery (ACM), 2007, pp. 781–788.

[10] ——, "Light beam search based multi-objective optimization using evolutionary algorithms," in *Proceedings of the Congress on Evolutionary Computation (CEC-07)*, 2007, pp. 2125–2132.

[11] G. W. Greenwood, X. Hu, and J. G. D'Ambrosio, "Fitness functions for multiple objective optimization problems: Combining preferences with pareto rankings," in *Foundations of Genetic Algorithms (FOGA)*. San Mateo: Morgan Kauffman, 1996, pp. 437–455.

[12] A. Jaszkiewicz and J. Branke, "Interactive multiobjective evolutionary algorithms," in *Multiobjective optimization – Interactive and Evolutionary Approaches (LNCS volume 5252)*, J. Branke, K. Deb, K. Miettinen, and R. Slowinski, Eds. Heidelberg: Springer, 2008, pp. 179–193.

[13] P. Korhonen and J. Laakso, "A visual interactive method for solving the multiple criteria problem," *European Journal of Operational Reseaech*, vol. 24, pp. 277–287, 1986.

[14] P. Korhonen and G. Y. Yu, "A reference direction approach to multiple objective quadratic-linear programming," *European Journal of Operational Research*, vol. 102, pp. 601–610, 1997.

[15] J. Branke, S. Greco, R. Slowinski, and P. Zielniewicz, "Interactive evolutionary multiobjective optimization using robust ordinal regression," in *Proceedings of the Fifth International Conference on Evolutionary Multi-Criterion Optimization (EMO-09)*. Berlin: Springer-Verlag, 2009, pp. 554–568.

[16] P. Korhonen, H. Moskowitz, and J. Wallenius, "A progressive algorithm for modeling and solving multiple-criteria decision problems," *Operations Research*, vol. 34, no. 5, pp. 726–731, 1986.

[17] P. Korhonen, H. Moskowitz, P. Salminen, and J. Wallenius, "Further developments and tests of a progressive algorithm for multiple criteria decision making," *Operations Research*, vol. 41, no. 6, pp. 1033–1045, 1993.

[18] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast and elitist multi-objective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[19] S. Phelps and M. Koksalan, “An interactive evolutionary metaheuristic for multiobjective combinatorial optimization,” *Management Science*, vol. 49, no. 12, pp. 1726–1738, December 2003.

[20] J. W. Fowler, E. S. Gel, M. Koksalan, P. Korhonen, J. L. Marquis, and J. Wallenius, “Interactive evolutionary multi-objective optimization for quasi-concave preference functions,” 2009, submitted to European Journal of Operational Research.

[21] A. Jaszkiewicz, “Interactive multiobjective optimization with the pareto memetic algorithm,” *Foundations of Computing and Decision Sciences*, vol. 32, no. 1, pp. 15–32, 2007.

[22] J. Figueira, S. Greco, and R. Slowinski, “Building a set of additive value functions reprsenting a reference preorder and intensities of preference: GRIP method,” *European Journal of Operational Research*, vol. 195, no. 2, pp. 460–486, 2009.

[23] E. Zitzler, M. Laumanns, and L. Thiele, “SPEA2: Improving the strength pareto evolutionary algorithm for multiobjective optimization,” in *Evolutionary Methods for Design Optimization and Control with Applications to Industrial Problems*, K. C. Giannakoglou, D. T. Tsahalis, J. Périaux, K. D. Papailiou, and T. Fogarty, Eds. Athens, Greece: International Center for Numerical Methods in Engineering (CIMNE), 2001, pp. 95–100.

[24] A. M. Geoffrion, “Proper efficiency and theory of vector maximization,” *Journal of Mathematical Analysis and Applications*, vol. 22, no. 3, pp. 618–630, 1968.

[25] P. Korhonen and J. Karaivanova, “An algorithm for projecting a reference direction onto the nondominated set of given points,” *IEEE Trans. on Systems, Man and Cybernetics–Part A: Systems and Humans*, vol. 29, no. 5, pp. 429–435, 1999.

[26] S. Zionts and J. Wallenius, “An interactive programming method for solving the multiple criteria problem,” *Management Science*, vol. 22, pp. 656–663, 1976.

[27] A. Mas-Colell, M. D. Whinston, and J. R. Green, *Microeconomic Theory*. New York: Oxford University Press, 1995.

[28] A. P. Wierzbicki, “The use of reference objectives in multiobjective optimization,” in *Multiple Criteria Decision Making Theory and Applications*, G. Fandel and T. Gal, Eds. Berlin: Springer-Verlag, 1980, pp. 468–486.

[29] K. Deb and R. B. Agrawal, “Simulated binary crossover for continuous search space,” *Complex Systems*, vol. 9, no. 2, pp. 115–148, 1995.

[30] K. V. Price, R. Storn, and J. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*. Berlin: Springer-Verlag, 2005.

[31] R. H. Byrd, J. Nocedal, and R. A. Waltz, *KNITRO: An integrated package for nonlinear optimization*. Springer-Verlag, 2006, pp. 35–59.

[32] E. Zitzler, K. Deb, and L. Thiele, “Comparison of multiobjective evolutionary algorithms: Empirical results,” *Evolutionary Computation Journal*, vol. 8, no. 2, pp. 125–148, 2000.

[33] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, “Scalable test problems for evolutionary multi-objective optimization,” in *Evolutionary Multi-objective Optimization*, A. Abraham, L. Jain, and R. Goldberg, Eds. London: Springer-Verlag, 2005, pp. 105–145.

[34] D. Saxena and K. Deb, “Trading on infeasibility by exploiting constraint’s criticality through multi-objectivization: A system design perspective,” in *Proceedings of the Congress on Evolutionary Computation (CEC-2007)*, in press.

[35] D. W. Corne, J. D. Knowles, and M. Oates, “The Pareto envelope-based selection algorithm for multiobjective optimization,” in *Proceedings of the Sixth International Conference on Parallel Problem Solving from Nature VI (PPSN-VI)*, 2000, pp. 839–848.

2 Progressively interactive evolutionary multi-objective optimization method using generalized polynomial value functions

A. Sinha, K. Deb, P. Korhonen, and J. Wallenius

In Proceedings of the 2010 IEEE Congress on Evolutionary Computation (CEC-2010), pages 1-8. IEEE Press, 2010.

Progressively Interactive Evolutionary Multi-Objective Optimization Method Using Generalized Polynomial Value Functions

Ankur Sinha, Kalyanmoy Deb, Pekka Korhonen and Jyrki Wallenius

***Abstract*— This paper advances and evaluates a recently proposed progressively interactive evolutionary multi-objective optimization algorithm. The algorithm uses preference information from the decision maker during the intermediate generations of the EMO and produces the most preferred solution on the Pareto-optimal front. The progress towards the Pareto-optimal front is made by approximating decision maker's value function. In this paper, a generalized polynomial value function has been proposed and the procedure to fit the value function to the decision maker's preference information has been described. The generality of the procedure of fitting a value function to the decision maker's preferences has been shown by using other existing value functions from the literature. The proposed generic polynomial value function has been incorporated in the PI-EMO-VF algorithm to efficiently approximate the decision maker's value function. The paper then evaluates the performance of the PI-EMO-VF algorithm on three and five objective test problems with constraints. It also evaluates the efficacy of the procedure in producing the most preferred solution when the decision maker is unable to provide perfect information, i.e., the decision maker finds certain pairs of solutions in the objective space to be incomparable. Results have been presented for three and five objective constrained test problems using the procedure.**

***Index Terms*— Evolutionary multi-objective optimization algorithm, multiple criteria decision-making, interactive multi-objective optimization algorithm, value/utility function, preference based multi-objective optimization.**

I. Introduction

The efficacy of the progressively interactive evolutionary multi-objective algorithms in handling high objective problems with a relatively less number of function evaluations has been shown recently [1], [2]. These methods are able to find the most preferred solution on the Pareto-optimal front, with the intervention of the decision maker in the intermediate generations of the algorithm. The PI-EMO-VF [1] procedure has been used in this study. The simplicity of the procedure incorporating decision making with EMO and its ability to solve high objective test problems makes the approach quite promising when EMO algorithms tend to suffer [3], [4] in such problems. In this paper, a generalized polynomial value function has been proposed which can be used to fit any preference information (quasi-concave) from the decision maker. This generalized polynomial value function has been incorporated in the PI-EMO-VF algorithm to approximate the preferences of the decision maker during the intermediate steps of the algorithm.

Very often in practice, whenever preferences are elicited from a decision maker about a pair of points (solutions in objective space), the decision maker finds the points in the pair as incomparable. Translating this decision of a decision maker to a value function may mean either of the two things. Firstly, that the two points lie on the same indifference curve, the probability of which is very less, and secondly, that the difference in the values of the two points is too small for the decision maker to perceive. The PI-EMO-VF algorithm while approximating the decision maker's value function takes such kind of preference information into account. Though this idea has already been mooted by the authors in their previous study but the algorithm has not been evaluated for any such test cases.

In this paper we make an attempt to evaluate the algorithm's performance with high objective constrained test problems. Further, we also test the performance when the algorithm is provided with information about some pairs being incomparable, instead of a strict preference of one member over the other. We begin the paper with some of the past studies done in the area of progressively interactive evolutionary multi-objective optimization algorithms. Thereafter we propose the generalized version of the polynomial value function. The value function fitting procedure to the decision maker's preferences has also been shown for two commonly used value functions in the economics literature. Finally, the results for the performance of the PI-EMO-VF procedure on constrained test problems have been produced and discussed.

II. Past Studies on Progressively Interactive Methods

Not many studies are yet available in the direction of a progressive use of preference information provided by the decision maker during the intermediate steps of the algorithm. Some recent studies periodically presented to the

Ankur Sinha is a doctoral student at the Department of Business Technology, Aalto University School of Economics, PO Box 21210, FIN-00101, Helsinki, Finland (ankur.sinha@hse.fi)

Kalyanmoy Deb is a Deva Raj Chair Professor at Indian Institute of Technology Kanpur, PIN 208016, India (deb@iitk.ac.in), Also a Professor at Department of Business Technology, Aalto University School of Economics, PO Box 21210, FIN-00101, Helsinki, Finland (kalyanmoy.deb@hse.fi)

Pekka Korhonen is a Professor at the Department of Business Technology, Aalto University School of Economics, PO Box 21210, FIN-00101, Helsinki, Finland (pekka.korhonen@hse.fi)

Jyrki Wallenius is a professor at the Department of Business Technology, Aalto University School of Economics, PO Box 21210, FIN-00101, Helsinki, Finland (jyrki.wallenius@hse.fi)

DM one or more pairs of alternative points found by an EMO algorithm and expected the DM to provide some preference information about the points. Some of the work in this direction has been done by Phelps and Köksalan [10], Fowler et al. [15], Jaszkiewicz [11], Branke et al. [6] and Korhonen et al. [7], [8].

In a couple of recent studies [1], [2], the authors have proposed a progressively interactive approach where the information from the decision maker is elicited and used to direct the search of the EMO in a preferred region. The first study fits a quasi-concave value function to the preferences provided by the decision maker and then uses it to drive the EMO. The second study uses the preference information from the decision maker to construct a polyhedral cone which is again used to drive the EMO procedure towards the region of interest.

A. Approximating Decision Maker's Preference Information with a Value Function

Information from the decision maker is usually elicited in the the form of his/her preferences. The DM (decision maker) is required to compare certain number of points in the objective space. The points are presented to the DM and pairwise comparisons of the given points result in either a solution being more preferred over the other or the solutions being incomparable. Based on such preference statements, a partial ordering of the points is done. If $P_k, k \in \{1,\ldots,\eta\}$ represents a set of η points in the decision space then for a given pair (i,j) the i-th point is either preferred over the j-th point ($P_i \succ P_j$), or they are incomparable ($P_i \equiv P_j$). This information is used to fit a value function which matches the DM's preferences. A number of available value functions can be chosen from the literature and the preference information can be fitted. Here, we describe the preference fitting task with three different value functions. The first value function is the CES [16] value function and the second one is the Cobb-Douglas [16] value function. These two value functions are commonly used in economics literature. The Cobb-Douglas value function is a special form of the CES value function. As the two value functions have a limited number of parameters, they can be used to fit only a certain class of convex preferences. In this study we propose a generalized polynomial value function which can be used to fit any kind of convex preference information. A special form of this value function has been suggested in an earlier study by Deb, Sinha, Korhonen and Wallenius [1]. They used this special form of the polynomial value function in the PI-EMO-VF procedure. In this section we discuss the process of fitting preference information to any value function and then incorporate the generalized value function in the PI-EMO-VF procedure in the later part of the paper.

1) CES Value Function:

$$\begin{array}{ll}
V(f_1,f_2,\ldots,f_M) = (\sum_{i=1}^{M}\alpha_i f_i^{\rho})^{\frac{1}{\rho}}, & \\
\text{such that} & \\
\quad \alpha_i \geq 0, \quad i=1,\ldots,m & \\
\quad \sum_{i=1}^{M}\alpha_i = 1 & \\
\text{where} \quad f_i \quad \text{are the objective functions} & \\
\text{and} \quad \rho,\alpha_i \quad \text{are the value function parameters} &
\end{array} \qquad (1)$$

2) Cobb-Douglas Value Function:

$$\begin{array}{l}
V(f_1,f_2,\ldots,f_M) = \prod_{i=1}^{M} f_i^{\alpha_i}, \\
\text{such that} \\
\quad \alpha_i \geq 0, \quad i=1,\ldots,m \\
\quad \sum_{i=1}^{M}\alpha_i = 1 \\
\text{where} \quad f_i \quad \text{are the objective functions} \\
\text{and} \quad \alpha_i \quad \text{are the value function parameters}
\end{array} \qquad (2)$$

3) Polynomial Value Function: A generalized polynomial value function has been suggested which can be utilized to fit any number of preference information by choosing a higher degree polynomial.

$$\begin{array}{lll}
\multicolumn{3}{l}{V(f_1,f_2,\ldots,f_M) = \prod_{j=1}^{p}\sum_{i=1}^{M}(\alpha_{ij}f_i+\beta_j)} \\
\multicolumn{3}{l}{\text{such that}} \\
\multicolumn{3}{l}{\quad 0 = 1-\sum_{i=0}^{M}\alpha_{ij}, \quad j=1,\ldots,p} \\
\multicolumn{3}{l}{\quad S_j = \sum_{i=1}^{M}(\alpha_{ij}f_i+\beta_j) > 0, \quad j=1,\ldots,p} \\
\multicolumn{3}{l}{\quad 0 \leq \alpha_{ij} \leq 1, \quad j=1,\ldots,p} \\
\text{where} & f_i & \text{are the objective functions} \\
 & \alpha_{ij},\beta_i,p & \text{are the value function parameters} \\
\text{and} & S_j & \text{are the linear product terms in} \\
 & & \text{the value function}
\end{array} \qquad (3)$$

A special form of this value function suggested by Deb, Sinha, Korhonen and Wallenius [1] used $p = M$, where M is the number of objectives. Choosing a value of $p = M$ makes the shape of the value function easily deductible with each product term, $S_j, j = 1,\ldots,M$, representing an asymptote (a hyper-plane). However, any positive integer value of p can be chosen. More the number of parameters in the value function, more is the flexibility and any type of quasi-concave indifference curve can be fitted by increasing the value of p. Once the preference information is given, the task is to figure out the parameters of the value function which capture the preference information optimally. Next, we frame the optimization problem which needs to be solved to figure out the value function parameters.

4) Value Function Optimization: Following is a generic approach which could be used to fit any value function to the preference information provided by the decision maker. In the equations, V represents the value function being used and P is a vector of objectives. $V(P)$ represents a scalar assigned to the objective vector P such that the scalar represents the utility/value of the objective vector. The optimization problem attempts to find such parameters for the value function for which the minimum difference in the value function values between the ordered pairs of points is maximum.

$$
\begin{array}{rl}
\text{Maximize} & \epsilon, \\
\text{subject to} & \text{Value Function Constraints} \\
& V(P_i) - V(P_j) \geq \epsilon, \quad \text{for all } (i,j) \text{ pairs} \\
& \qquad\qquad \text{satisfying } P_i \succ P_j, \\
& |V(P_i) - V(P_j)| \leq \delta_V, \quad \text{for all } (i,j) \text{ pairs} \\
& \qquad\qquad \text{satisfying } P_i \equiv P_j.
\end{array} \tag{4}
$$

The first set of constraints ensure that the constraints specified in the construction of the value function are met. It includes variable bounds for the value function parameters or any other equality or inequality constraints which need to be satisfied to obtain feasible parameters for the value function. The second and third set of constraints ensure that the preference order specified by the decision maker is maintained for each pair. The second set ensures the domination of one point over the other such that the difference in their values is at least ϵ. The third constraint set takes into account all pairs of incomparable points. For such pairs of points, we would like to restrict the absolute difference between their value function values to be within a small range (δ_V). $\delta_V = 0.1\epsilon$ has been used in the entire study to avoid introducing another parameter in the optimization problem. It is noteworthy that the value function optimization task is considered to be successful, with all the preference orders satisfied, only if a positive value of ϵ is obtained by solving the above problem. A non-positive value of ϵ means that the chosen value function is unable to fit the preference information provided by the decision maker.

Now, we consider a case where five ($\eta = 5$) hypothetical points ($P_1 = (3.6, 3.9)$, $P_2 = (2.5, 4.1)$, $P_3 = (5.5, 2.5)$, $P_4 = (0.5, 5.2)$, and $P_5 = (6.9, 1.8)$) are presented to a decision maker. Let us say that the decision maker provides strict preference of one point over the other and a complete ranking of points is obtained with P_1 being the best, $P2$ being the second best, $P3$ being the third best, $P4$ being the fourth best and P_5 being the worst. Due to a complete ranking, third constraint set will not exist while framing the value function optimization problem. Trying to fit a linear value function to the preferences reveals that there does not exist any such linear function which could take all the preference information into account. Next we resort to non-linear value functions. CES value function, Cobb-Douglas value function as well as the Polynomial value function are found to satisfy all the preference constraints. Figure 1, 3 and 5 represent the indifference curves corresponding to the respective value functions and the points.

Next we consider a case where the decision maker finds a pair or pairs incomparable. The points mentioned in the previous paragraph have been used again. From the preference information provided by the decision maker a partial ordering of the points is done with P_1 being the best, $P2$ being the second best, $P3$ being the third best and $P4, P5$ being the worst. In this case the decision maker finds the pair $(P4, P5)$ incomparable. This introduces the third set of constraints as well in the value function optimization problem. Figure 2, 4 and 6 represent the points and the indifference curves corresponding to the respective value functions for the case where the DM finds a pair incomparable.

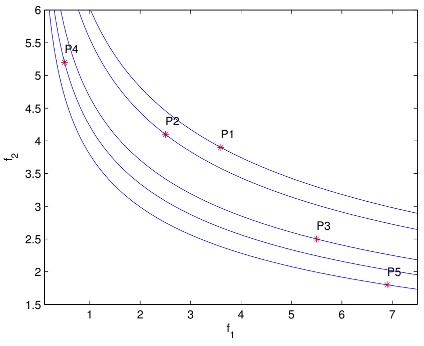

Fig. 1. CES value function for $P1 \succ P2 \succ P3 \succ P4 \succ P5$. The equation for the value function is $V(f_1, f_2) = 0.28 f_1^{0.13} + 0.72 f_2^{0.13}$

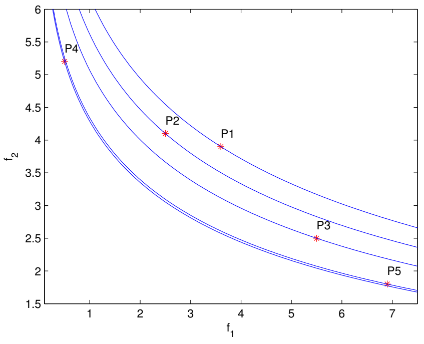

Fig. 2. CES value function for $P1 \succ P2 \succ P3 \succ P4 \equiv P5$. The equation for the value function is $V(f_1, f_2) = 0.31 f_1^{0.26} + 0.69 f_2^{0.26}$

III. Progressively Interactive EMO Using Value Functions (PI-EMO-VF)

A progressively interactive EMO [1] was proposed by Deb, Sinha, Korhonen and Wallenius where an approximate value function is generated after every τ generations of the EMO. Ideas from the multiple criteria decision making (MCDM) were combined with NSGA-II [9] to implement the procedure. The decision maker is provided with η well sparse non-dominated solutions from the current population of non-dominated points of the EMO algorithm. The DM is then expected to provide a complete or partial preference information about the superiority or indifference of one solution over the other. Based on the DM preferences the parameters of a strictly monotone value function are determined by solving the value function optimization problem discussed above.

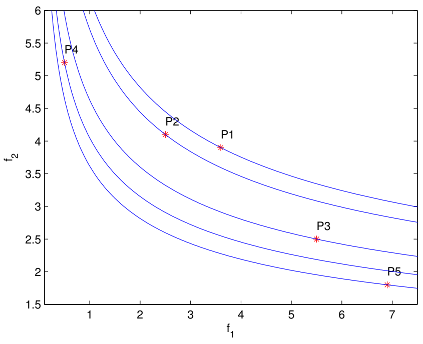

Fig. 3. Cobb-Douglas value function for $P1 \succ P2 \succ P3 \succ P4 \succ P5$. The equation for the value function is $V(f_1, f_2) = f_1^{0.27} f_2^{0.73}$

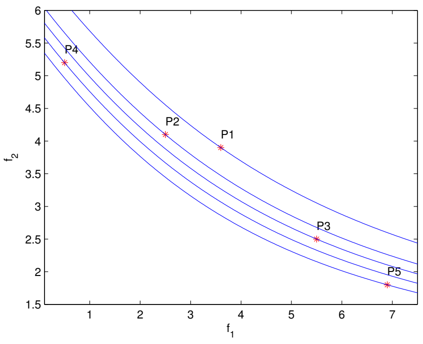

Fig. 5. Polynomial value function for $P1 \succ P2 \succ P3 \succ P4 \succ P5$. The equation for the value function is $V(f_1, f_2) = f_2(f_1 + 0.54f_2 - 0.54)$

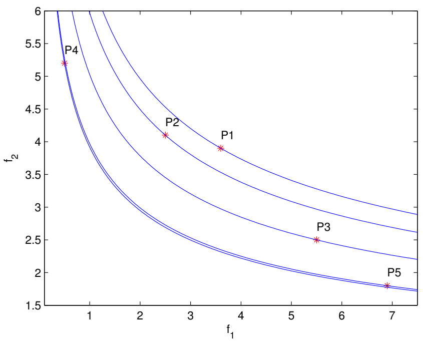

Fig. 4. Cobb-Douglas value function for $P1 \succ P2 \succ P3 \succ P4 \equiv P5$. The equation for the value function is $V(f_1, f_2) = f_1^{0.29} f_2^{0.71}$

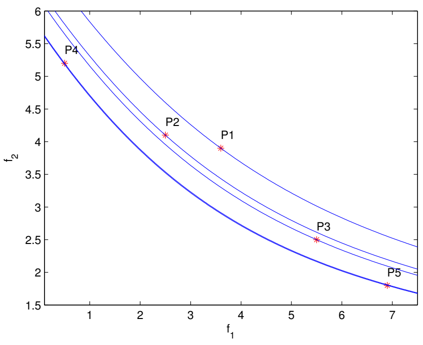

Fig. 6. Polynomial value function for $P1 \succ P2 \succ P3 \succ P4 \equiv P5$. The equation for the value function is $V(f_1, f_2) = f_2(f_1 + 0.49f_2 - 0.49)$

The determined value function is used to guide the search of the EMO towards the region of interest. A local search based termination criteria was also proposed in the study. The termination condition is set-up based on the expected progress which can be made with respect to the constructed value function.

In this study we replace the previously used polynomial value function with a generalized polynomial value function. To begin with, a polynomial value function with $p = 1$ is optimized. In case the optimization process is unsuccessful with a negative ϵ, the value of p is incremented by 1. This is done until a value function is found which is able to fit the preference information. This ensured that any information received by the decision maker always gets fitted with a value function. This makes the algorithm more efficient eliminating cases where a value function cannot be fitted to the DM preferences.

In the previous study two, three and five objective unconstrained test problems were successfully solved using the algorithm. The algorithm was able to produce solutions close to the most preferred point with a very high accuracy. A decision maker (DM) was replaced with a DM emulated value function which was used to provide preference information during the progress of the algorithm. Though the possibility of the decision maker's indifference towards a pair of solutions was discussed but the algorithm was evaluated only with a DM emulated value function which provides perfect ordering of the points. In the following part of this paper we evaluate the efficacy of the algorithm on three and five objective test problems with constraints. We also evaluate, how does the efficiency of the algorithm change in case the decision maker is unable to provide perfectly ordered set of points i.e. he/she finds some of the pairs incomparable.

IV. Results

In this section, we present the results of the PI-NSGA-II-VF procedure on three, and five objective test problems. DTLZ8 and DTLZ9 test problems are adapted to create maximization problems. In all simulations, we have used the following parameter values:

1) Number of points given to the DM for preference information: $\eta = 5$.
2) Number of generations between two consecutive DM calls: $\tau = 5$.
3) Termination parameter: $d_s = 0.01, 0.1$.
4) Crossover probability and the distribution index for the SBX [13] operator: $p_c = 0.9$ and $\eta_c = 15$.
5) Mutation probability: $p_m = 0$.
6) Population size: $N = 10M$, where M is the number of objectives.

Now, we present the test problems and results obtained with the above setting.

A. Three-Objective Test Problem

The DTLZ8 [14] test problem is scalable to any number of objectives. Here, we consider a three objective DTLZ8 problem. As suggested in [14] the number of variables has been chosen as 30 ($n = 10M$). The original problem is a minimization problem but since we wish to maximize the objectives, a negative sign has been used before both the objectives to turn the problem into a maximization problem. The description of the test problem for M number of objectives and n number of variables is as follows:

$$\begin{array}{ll} \text{Maximize} & f_j(\mathbf{x}) = -\frac{1}{\lfloor \frac{n}{M} \rfloor} \sum_{i=\lfloor (j-1)\frac{n}{M} \rfloor}^{\lfloor j\frac{n}{M} \rfloor} x_i, \\ & \quad j = 1, 2, \dots, M, \\ \text{subject to} & g_j(\mathbf{x}) = f_M(\mathbf{x}) + 4f_j(\mathbf{x}) - 1 \geq 0, \\ & \quad j = 1, 2, \dots, M-1, \\ & g_M(\mathbf{x}) = 2f_M(\mathbf{x}) + \\ & \quad \min_{\substack{i,j=1 \\ i \neq j}}^{M-1} [f_i(\mathbf{x}) + f_j(\mathbf{x})] - 1 \geq 0, \\ & 0 \leq x_i \leq 1, \quad \text{for } i = 1, \dots, n, \end{array} \tag{5}$$

For an M objective test problem the number of constraints are equal to $M-1$. The Pareto-optimal front is a combination of a straight line and a hyper plane. The straight line is the intersection of first $M-1$ constraints with $f_1 = f_2 = \ldots = f_{M-1}$ and the hyper-plane is represented by the plane g_M. The Pareto-optimal front for a three-objective DTLZ8 problem is shown in Figure 7.

To test the working of PI-NSGA-II-VF on this problem, we have replaced the decision maker by using a non-linear value function (emulating the DM), as follows:

$$V(\mathbf{f}) = 1 / \sum_{i=1}^{3} (f_i - a_i)^2, \tag{6}$$

where $\mathbf{a} = (0.0, 0.0, -0.775)^T$. This value function produces the most preferred solution on the Pareto-optimal front as $\mathbf{z}^* = (-0.05, -0.05, -0.80)$.

The PI-NSGA-II-VF is run with $N = 10 \times 3$ or 30 population members 21 times, each time with a different random initial population. In terms of the accuracy measure from $\mathbf{z}^*$, Table I presents the best, median and worst performing runs. Table II shows the accuracy, number of overall function evaluations and number of DM calls needed by the procedure. It is clear that the obtained points are close to the most preferred point $\mathbf{z}^*$.

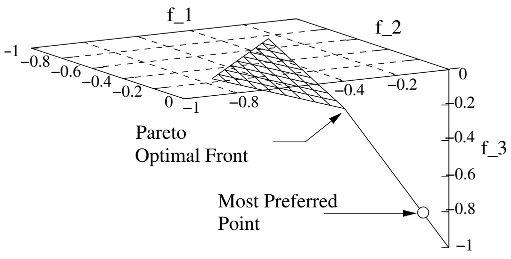

Fig. 7. Final population members after termination of the algorithm for three-objective modified DTLZ8 problem.

TABLE I
FINAL SOLUTIONS OBTAINED BY PI-NSGA-II-VF FOR THE THREE-OBJECTIVE DTLZ8 PROBLEM.

	$\mathbf{z}^*$	Best	Median	Worst
f_1	-0.0500	-0.0501	-0.0503	-0.0512
f_2	-0.0500	-0.0501	-0.0504	-0.0514
f_3	-0.8000	-0.7996	-0.7988	-0.7952

TABLE II
DISTANCE OF OBTAINED SOLUTION FROM THE MOST PREFERRED SOLUTION, NUMBER OF FUNCTION EVALUATIONS, AND NUMBER OF DM CALLS REQUIRED BY PI-NSGA-II-VF ON THE THREE-OBJECTIVE DTLZ8 PROBLEM. $d_s = 0.01$

	Minimum	Median	Maximum
Accuracy	0.0004	0.0013	0.0051
# of Function Eval.	11274	13636	15264
# of DM Calls	24	29	36

In Table III, the termination parameter d_s has been relaxed to 0.1. This leads to reduction in function evaluations as well as reduction in the number of DM-calls. This reduction comes with a trade-off and the accuracy in turn becomes low. The d_s parameter offers a flexibility to go for high number of DM-calls if a very high accuracy is desirable or reduce the number of DM-calls and be satisfied with an adequate accuracy. A low value of d_s leads to a high accuracy with an increased number of DM-calls. In a future study, we plan to eliminate this parameter d_s with a fixed number of DM-calls, i.e., instead of fixing this parameter d_s, the user of the algorithm will have to provide an input indicating the maximum number of times the decision maker can be called to provide preference information.

B. Five-Objective Test Problem

We now consider the five-objective ($M = 5$) version of the DTLZ9 [14] problem. This is also a scalable problem and the number of variables have been suggested to be chosen

TABLE III

DISTANCE OF OBTAINED SOLUTION FROM THE MOST PREFERRED SOLUTION, NUMBER OF FUNCTION EVALUATIONS, AND NUMBER OF DM CALLS REQUIRED BY PI-NSGA-II-VF ON THE THREE-OBJECTIVE DTLZ8 PROBLEM. $d_s = 0.1$.

	Minimum	Median	Maximum
Accuracy	0.0074	0.0230	0.0819
# of Function Eval.	3706	4625	4665
# of DM Calls	8	11	18

as $n = 10M$. So the five objective test problem which we consider here has 50 number of decision variables. The original problem once again is a minimization problem but since we wish to maximize the objectives a negative sign has been used before each of the objectives to turn the problem into a maximization problem. The description of the test problem for M number of objectives and n number of variables is as follows:

$$\begin{array}{rl} \text{Maximize} & f_j(\mathbf{x}) = -\sum_{i=\lfloor (j-1)\frac{n}{M} \rfloor}^{\lfloor j \frac{n}{M} \rfloor} x_i^{0.1}, \quad j = 1, 2, \ldots, M, \\ \text{subject to} & g_j(\mathbf{x}) = f_M^2(\mathbf{x}) + f_j^2(\mathbf{x}) - 1 \geq 0, \\ & \quad j = 1, 2, \ldots, M-1, \\ & 0 \leq x_i \leq 1, \quad \text{for } i = 1, \ldots, n, \end{array} \tag{7}$$

For this test problem, the Pareto-optimal front is a curve with $f_1 = f_2 = \ldots = f_{M-1}$. The Pareto-optimal curve lies on the intersection of all $M-1$ constraints. A two dimensional plot of the Pareto-optimal front with f_M and any other objective represents a circular arc of radius 1. The Pareto-optimal front for a five-objective DTLZ9 problem is shown in Figure 8 with objectives f_1 and f_5. The other objective values (f_2, f_3, f_4) are equal to f_1.

For this problem, we choose a non-linear DM-emulated value function, as follows:

$$V(\mathbf{f}) = 1/\sum_{i=1}^{5}(f_i - a_i)^2, \tag{8}$$

where $\mathbf{a} = (-0.175, -0.175, -0.175, -0.175, -0.4899)^T$. This value function produces the most preferred point as $\mathbf{z}^* = (-0.2, -0.2, -0.2, -0.2, -0.9798)$.

Table IV presents the obtained solutions by PI-NSGA-II-VF with 50 population members. Table V shows the accuracy

TABLE IV

FINAL OBJECTIVE VALUES OBTAINED FROM PI-NSGA-II-VF FOR THE FIVE-OBJECTIVE DTLZ9 PROBLEM.

	$\mathbf{z}^*$	Best	Median	Worst
f_1	-0.2000	-0.2012	-0.2023	-0.2610
f_2	-0.2000	-0.2002	-0.2008	-0.2408
f_3	-0.2000	-0.2008	-0.2024	-0.2111
f_4	-0.2000	-0.2005	-0.2004	-0.2007
f_5	-0.9798	-0.9798	-0.9797	-0.9797

measure, the number of overall function evaluations, and

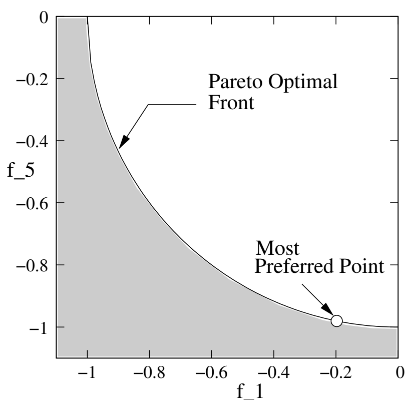

Fig. 8. Final population members after termination of the algorithm for five-objective DTLZ9 problem.

TABLE V

DISTANCE OF OBTAINED SOLUTION FROM THE MOST PREFERRED SOLUTION, FUNCTION EVALUATIONS, AND THE NUMBER OF DM CALLS REQUIRED BY PI-NSGA-II-VF FOR THE FIVE-OBJECTIVE DTLZ9 PROBLEM. $d_s = 0.01$.

	minimum	median	maximum
Accuracy	0.0015	0.0034	0.0742
# of Function Eval.	25452	29035	38043
# of DM Calls	59	63	89

the number of DM calls. Although the points close to the most preferred point are obtained in each run, the higher dimensionality of the problem requires more function evaluations and DM calls compared to the three-objective test problem. However, the above results are obtained for a strict termination criterion with $d_s = 0.01$. Smaller number of DM calls and evaluations are expected if this termination criterion is relaxed. In Table VI, the termination parameter d_s has been relaxed to 0.1. Once again we find that this leads to reduction in function evaluations as well as reduction in the number of DM-calls. The accuracy also becomes low.

TABLE VI

DISTANCE OF OBTAINED SOLUTION FROM THE MOST PREFERRED SOLUTION, FUNCTION EVALUATIONS, AND THE NUMBER OF DM CALLS REQUIRED BY PI-NSGA-II-VF FOR THE FIVE-OBJECTIVE DTLZ9 PROBLEM. $d_s = 0.1$

	minimum	median	maximum
Accuracy	0.0284	0.0673	1.3836
# of Function Eval.	8201	9273	12806
# of DM Calls	19	24	33

It is worth mentioning that the application of a usual EMO (including the original NSGA-II) is reported to face difficulties in converging to the *entire* five-dimensional Pareto-optimal front with an identical number of function evaluations in an earlier study [14]. However, our target is a single preferred point on the Pareto-optimal front (dictated by a sequence of preference information provided by the DM), our procedure is able to find a near Pareto-optimal solution for the five-objective optimization problem. The results demonstrate the increased efficacy of an EMO when preference information is used in the intermediate generations.

V. Performance in Absence of Perfect Information

Next, we consider a case where the decision maker is unable to provide strict preference for some of the pairs in the η set of solutions given to him. In this case the algorithm does not have the complete ranking of the η points, instead some of the members are equally good as the other. In order to simulate a decision maker who is unable to provide preferences for some of the pairs, we again consider a DM-emulated value function which provides strict preference. From this value function, the values of each of the η points are calculated. Once we have all the values, the difference between the minimum and the maximum value is taken. Let us call the difference to be D. Now, if for any given pair the absolute difference between the value functions is less than or equal to $\alpha D, \alpha < 1$, then the decision maker is said to be unable to make a comparison between those two points. To illustrate the procedure, let us consider a set of η points $P_1, P_2, \ldots, P_\eta$. Let the DM-emulated value function be V_{DM}. The values for each of the η points are calculated and the points are ranked as $P_1 \succeq P_2 \succeq \ldots \succeq P_\eta$ such that $V_{DM}(P_1) \geq V_{DM}(P_2) \geq \ldots \geq V_{DM}(P_\eta)$. This means that the difference between the maximum and the minimum value is $D = V_{DM}(P_1) - V_{DM}(P_\eta)$. For any given pair (P_i, P_j) if the absolute difference in the values is found to be less than αD, then the decision maker marks that pair as incomparable.

An interesting case occurs for three points P_i, P_j, P_k, when $V_{DM}(P_i) - V_{DM}(P_j) \leq \alpha D$, $V_{DM}(P_j) - V_{DM}(P_k) \leq \alpha D$ but $V_{DM}(P_i) - V_{DM}(P_k) > \alpha D$. In this case, when the decision maker is asked to compare (P_i, P_j), he/she concludes $(P_i \equiv P_j)$ and when he/she is asked to compare (P_j, P_k), he/she concludes that $(P_j \equiv P_k)$. Transitivity should imply that $(P_i \equiv P_k)$ but such a decision maker would defy transitivity by concluding $(P_i \succ P_k)$ as $V_{DM}(P_i) - V_{DM}(P_k) > \alpha D$. However, such a decision maker will always be transitive in case of the following preferences i.e., $P_l \succ P_m \succ P_n$ implies $P_l \succ P_n$ and $P_l \succeq P_m \succeq P_n$ implies $P_l \succeq P_n$.

The procedure mentioned above has been used to emulate a decision maker who finds some of the pairs incomparable. The decision maker is also intransitive when it comes to the equivalence relation. Once the information is elicited from such a decision maker, it is given as input to the algorithm which tries to fit an approximated value function to the preference information and uses it in the subsequent τ generations. Now, we present the results for three objective DTLZ8 and five objective DTLZ9 test problems where α has been varied as $\{0, 0.10, 0.15, 0.20, 0.25\}$. Figure 9 and 10 show the change in accuracy, number of function evaluations and number of DM calls as α increases. With increase in inconclusiveness i.e., with more and more pairs being assigned as incomparable, the number of function evaluations and the number of DM calls are found to increase. The accuracy also gets affected and becomes poorer as α increases.

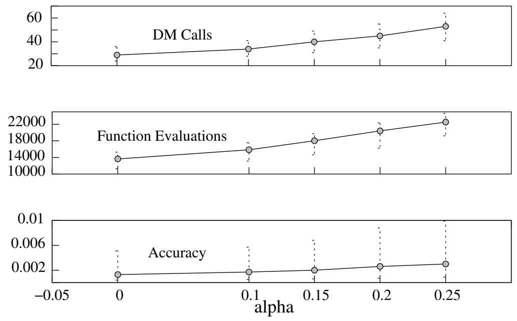

Fig. 9. Change in accuracy, function evaluations and DM calls with increase in α for three objective DTLZ-8 problem.

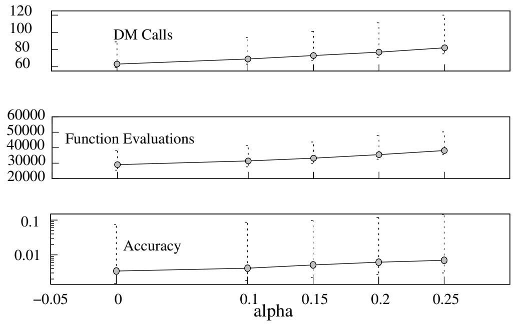

Fig. 10. Change in accuracy, function evaluations and DM calls with increase in α for five objective DTLZ-9 problem.

VI. Conclusions

In this paper, we have described a technique to fit value functions to the preferences of the decision maker. To fit the preference information, a couple of commonly used value functions have been picked up from the literature. With a limited number of parameters in these value functions, it is not possible to fit all kinds of convex preference information, hence, we proposed a generic polynomial value function whose complexity can be increased by adding more number of product terms and any kind of convex preferences can be fitted. The value function optimization technique allows to figure out the best parameters for the value function. The PI-EMO-VF algorithm used a special form of the generic polynomial value function which has been replaced. The algorithm now uses a generic polynomial value function.

The PI-EMO-VF algorithm has been tested with a three and a five objective constrained test problem. The algorithm is able to successfully find the most preferred point in these high objective difficult test problems. A technique is also suggested to emulate a decision maker who is inconclusive with some of the pairs given for preference information. Such a decision maker finds the points incomparable if the utility of the two points are quite close. The algorithm has been further evaluated for such a decision maker and the results have been presented. It is found that the algorithm is able to efficaciously respond to the preferences from the decision maker and gets close to the most preferred solution. Though, the accuracy gets hampered with an increasing inconclusiveness of the decision maker.

Incorporating the aspects of decision making in the intermediate steps of EMO algorithms appears to be a promising approach to handle high objective optimization problems. This paper is an attempt by the authors to evaluate the idea and demonstrate the effectiveness of the hybrid procedure.

VII. Acknowledgements

Authors wish to thank the Academy of Finland (Grant Number: 121980) for supporting this study.

References

[1] K. Deb, A. Sinha, P. Korhonen and J. Wallenius, “An Interactive Evolutionary Multi-Objective Optimization Method Based on Progressively Approximated Value Functions,” *IEEE Transactions on Evolutionary Computation*, 2010.

[2] A. Sinha, K. Deb, P. Korhonen and J. Wallenius, “An Interactive Evolutionary Multi-Objective Optimization Method Based on Polyhedral Cones,” in *Proceedings of Learning and Intelligent Optimization Conference (LION-2010)*, 2010.

[3] J. Knowles and D. Corne, “Quantifying the effects of objective space dimension in evolutionary multiobjective optimization,” in *Proceedings of the Fourth International Conference on Evolutionary Multi-Criterion Optimization (EMO-2007)*, 2007, pp. 757–771, (LNCS 4403).

[4] K. Deb and D. Saxena, “Searching for Pareto-optimal solutions through dimensionality reduction for certain large-dimensional multi-objective optimization problems,” in *Proceedings of the World Congress on Computational Intelligence (WCCI-2006)*, 2006, pp. 3352–3360.

[5] A. Jaszkiewicz and J. Branke, “Interactive multiobjective evolutionary algorithms,” in *Multiobjective optimization – Interactive and Evolutionary Approaches (LNCS volume 5252)*, J. Branke, K. Deb, K. Miettinen, and R. Slowinski, Eds. Heidelberg: Springer, 2008, pp. 179–193.

[6] J. Branke, S. Greco, R. Slowinski, and P. Zielniewicz, “Interactive evolutionary multiobjective optimization using robust ordinal regression,” in *Proceedings of the Fifth International Conference on Evolutionary Multi-Criterion Optimization (EMO-09)*. Berlin: Springer-Verlag, 2009, pp. 554–568.

[7] P. Korhonen, H. Moskowitz, and J. Wallenius, “A progressive algorithm for modeling and solving multiple-criteria decision problems,” *Operations Research*, vol. 34, no. 5, pp. 726–731, 1986.

[8] P. Korhonen, H. Moskowitz, P. Salminen, and J. Wallenius, “Further developments and tests of a progressive algorithm for multiple criteria decision making,” *Operations Research*, vol. 41, no. 6, pp. 1033–1045, 1993.

[9] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, “A fast and elitist multi-objective genetic algorithm: NSGA-II,” *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[10] S. Phelps and M. Koksalan, “An interactive evolutionary metaheuristic for multiobjective combinatorial optimization,” *Management Science*, vol. 49, no. 12, pp. 1726–1738, December 2003.

[11] A. Jaszkiewicz, “Interactive multiobjective optimization with the pareto memetic algorithm,” *Foundations of Computing and Decision Sciences*, vol. 32, no. 1, pp. 15–32, 2007.

[12] A. P. Wierzbicki, “The use of reference objectives in multiobjective optimization,” in *Multiple Criteria Decision Making Theory and Applications*, G. Fandel and T. Gal, Eds. Berlin: Springer-Verlag, 1980, pp. 468–486.

[13] K. Deb and R. B. Agrawal, “Simulated binary crossover for continuous search space,” *Complex Systems*, vol. 9, no. 2, pp. 115–148, 1995.

[14] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, “Scalable test problems for evolutionary multi-objective optimization,” in *Evolutionary Multiobjective Optimization*, A. Abraham, L. Jain, and R. Goldberg, Eds. London: Springer-Verlag, 2005, pp. 105–145.

[15] J. W. Fowler, E. S. Gel, M. Koksalan, P. Korhonen, J. L. Marquis and J. Wallenius “Interactive Evolutionary Multi-Objective Optimization for Quasi-Concave Preference Functions,” in *Submitted to European Journal of Operational Research*, 2009.

[16] A. Mas-Colell, M. D. Whinston and J. R. Green, *Microeconomic Theory*. New York: Oxford University Press, 1995.

3 An interactive evolutionary multi-objective optimization method based on polyhedral cones

A. Sinha, P. Korhonen, J. Wallenius, and K. Deb

In Proceedings of the 2010 Learning and Intelligent Optimization Conference (LION-2010), pages 318-332. IEEE Press, 2010.

An Interactive Evolutionary Multi-Objective Optimization Method Based on Polyhedral Cones

Ankur Sinha, Pekka Korhonen, Jyrki Wallenius and Kalyanmoy Deb[**]

Department of Business Technology
Aalto University School of Economics
PO Box 21220, 00076 Aalto, Helsinki, Finland
{Firstname.Lastname}@hse.fi

Abstract. This paper suggests a preference based methodology, where the information provided by the decision maker in the intermediate runs of an evolutionary multi-objective optimization algorithm is used to construct a polyhedral cone. This polyhedral cone is used to eliminate a part of the search space and conduct a more focussed search. The domination principle is modified, to look for better solutions lying in the region of interest. The search is terminated by using a local search based termination criterion. Results have been presented on two to five objective problems and the efficacy of the procedure has been tested.

Keywords: Evolutionary multi-objective optimization, multiple criteria decision-making, interactive multi-objective optimization, sequential quadratic programming, preference based multi-objective optimization.

1 Introduction

Most of the existing evolutionary multi-objective optimization (EMO) algorithms aim to find a set of well-converged and well-diversified Pareto-optimal solutions [1, 2]. As discussed elsewhere [3, 5], finding the entire set of Pareto-optimal solutions has its own intricacies. Firstly, the usual domination principle allows a majority of the population members to become non-dominated, thereby not allowing much room for introducing new solutions in a finite population. This slows down the progress of an EMO algorithm. Secondly, the representation of a high-dimensional Pareto-optimal front requires an exponentially large number of points, thereby requiring a large population size in running an EMO procedure. Thirdly, the visualization of a high-dimensional front becomes a non-trivial task for decision-making purposes.

In most of the existing EMO algorithms the decision maker is usually not involved during the optimization process. The decision maker is called only at the end of the optimization run after a set of approximate Pareto-optimal solutions has been found. The decision making process is then executed by choosing the most preferred solution from the set of approximate Pareto-optimal solutions obtained. This approach is called

[**] Also Department of Mechanical Engineering, Indian Institute of Technology Kanpur, PIN 208016, India (deb@iitk.ac.in).

the *aposteriori approach*. This approach makes it necessary for the algorithm to produce the entire set of approximate Pareto-optimal solutions so that a decision making process can be executed.

Some EMO researchers adopt a particular multiple criteria decision-making (MCDM) approach (*apriori approach*) to avoid the problems associated with finding the entire front. In such an approach the entire Pareto-optimal set is not aimed at rather a crowded set of Pareto-optimal solutions near the most preferred solution is targeted. In this approach the decision maker interacts at the beginning of an EMO run. The cone-domination based EMO [6, 1], biased niching based EMO [7], reference point based EMO approaches [8, 9], the reference direction based EMO [10], the light beam approach based EMO [11] are a few attempts in this direction.

In a *semi-interactive EMO approach*, the decision maker is involved iteratively [13, 14] in the optimization process. Some preference information (in terms of reference points or reference directions or others) is accepted from the decision maker and an MCDM-based EMO algorithm is employed to find a set of preferred Pareto-optimal solutions. Thereafter, a few representative preferred solutions are shown to the DM and a second set of preference information in terms of new reference points or new reference directions is obtained and a second MCDM-based EMO run is made. This procedure is continued till a satisfactory solution is found.

However, the decision maker could be integrated with the optimization run of an EMO algorithm in a much more effective way, as shown in recent studies [4, 15]. These approaches require progressive interaction with the decision maker during the intermediate generations of the optimization process to converge towards the most preferred solution. Such a *progressively interactive EMO approach* (PI-EMO), allows the decision maker to modify her/his preference structure as new solutions evolve, thus making the process more DM-oriented.

This paper discusses a simple PI-EMO where the decision maker is provided with a set of points perodically and asked to pick the most preferred solution from the set. Each time the decision maker is asked to make a choice of the most preferred solution, we call the instance as a ‘DM call’. With the information obtained from the decision maker a polyhedral cone is constructed and the domination principle is modified. The obtained polyhedral cone is further utilized to figure out a direction in which a local search is performed to determine the termination of the PI-EMO algorithm. The PI-EMO concept has been integrated with the NSGA-II algorithm [18] and the working of the algorithm has been demonstrated on three test problems having two, three and five objectives. A parametric study of the algorithm has also been done to determine the overall working of the algorithm.

2 Past Studies on Progressively Interactive Methods

Towards the methodologies involving a progressive use of preference information by involving a decision-maker in an evolutionary multi-objective optimization framework, there are not many studies yet. Some recent studies periodically presented to the DM one or more pairs of alternative points found by an EMO algorithm and expected the DM to provide some preference information about the points. Some of the work in this direction has been done by Phelps and Köksalan [20], Fowler et al. [29], Jaszkiewicz [21], Branke et al. [15] and Korhonen, Moskowitz and Wallenius [16].

In a recent study [4], the authors have proposed a progressively interactive approach (PI-EMO-VF) where the information from the decision maker is used to arrive at a quasi-concave value function which is used to drive the EMO. The approach based on DM interaction, generates a rank order of a set of points. In the same paper a local search based termination criterion has been proposed, the efficacy of which motivates us to use the same termination criterion for this study as well.

3 Progressively Interactive EMO Based on Polyhedral Cones (PI-EMO-PC)

In this section, we propose a progressively interactive EMO algorithm (PI-EMO-PC), where a polyhedral cone is used to modify the domination criteria of an EMO and drives it towards a single most preferred point on an M objective maximization problem. The polyhedral cone is arrived at using the preference information provided by the decision maker. The principle may be integrated with any standard EMO algorithm (such as NSGA-II [18], SPEA2 [22] and others) which works with a population of points in each iteration and prefers a sparsed set of non-dominated points in a population. The integration of the above principle with an EMO algorithm modifies the working of the algorithm and helps in finding the most preferred solution instead of the entire Pareto-optimal set.

For this purpose, after every τ generations of an EMO algorithm, we provide the decision-maker with an archive set containing non-dominated solutions and expect him/her to choose the best solution. The decision maker picks the best solution from the archive set of non-dominated solutions using an advanced selection technique known as VIMDA [30]. It is a visual interactive method which uses the reference point technique to allow the decision maker to select the best point from a set of non-dominated points. Its efficacy could be well demonstrated for high dimensional objective vectors when a geometrical representation is not possible. Once the best solution has been picked, the end points of the non-dominated front from the current parent population are chosen as the other points. It is noteworthy that the end points of the front are the points with best function value in one of the objectives. Such points are exactly M in number. This provides us with η ($= M + 1$, M is the number of objectives) points which is used to construct M different hyperplanes i.e., sides of a polyhedral cone. (described in Section 3.1).

The modified domination criterion is used until the next τ generations. The suggested termination criterion also uses the information from the polyhedral cone to decide when to terminate the algorithm. The following gives a step-by-step procedure of the proposed progressively interactive EMO (PI-EMO) methodology:

Step 1: Initialize a population Par_0 and set iteration counter $t = 0$. Initialize archive set A [1]. Domination of one solution over another is defined based on the usual definition of dominance [23] and an EMO algorithm is executed for τ iterations

[1] The initial size of the archive is $|A| = 0$. The maximum size the archive can have is $|A|^{max}$.

(generations). The value of t is incremented by one after each iteration. At each iteration of the EMO all the feasible non-dominated solutions found in the population are added to the archive A. The archive, at each iteration, is updated by removing the non-dominated solutions. If the archive size exceeds $|A|^{max}$, k-mean clustering is used to keep the diversed set of $|A|^{max}$ clusters and rest of the solutions are deleted.

Step 2: If ($t \mod \tau = 0$), the DM chooses the best solution A_t^{best}, from the archive A_t using VIMDA. Choose the end points from the non-dominated front of the current parent population Par_t as rest of the solutions. This makes the chosen solution count as $\eta = M + 1$: otherwise, proceed to Step 5.

Step 3: Construct the sides of the polyhedral cone from the chosen set of points, described in Section 3.1.

Step 4: A termination check (described in Section 3.2) is performed based on the expected improvement in solutions from the currently judged best solution. If the expected improvement is not significant (with respect to a parameter d_s), the algorithm is terminated and the current best solution is chosen as the final outcome.

Step 5: The parent population Par_t is used to create a new population Off_t (offsprings) by using a modified domination principle (discussed in Section 3.3) based on the current polyhedral cone and EMO algorithm's search operators.

Step 6: Populations Par_t and Off_t are used to determine a new population Par_{t+1} using the polyhedral cone and EMO algorithm's diversity preserving operator. The iteration counter is incremented as $t \leftarrow t + 1$ and the algorithm proceeds to Step 2.

The above is a generic progressively interactive procedure, which can be combined with any existing EMO algorithm in Step 1 and subsequently in Steps 5 and 6. The PI-EMO algorithm expects the user to set a value for τ, d_s and $|A|^{max}$.

We now provide the details for the specific procedures used in this study for Steps 3 to 6.

3.1 Step 3: Polyhedral Cone

At an instance of a DM call, η members need to be selected to create M different hyperplanes which form the sides of the polyhedral cone. The decision maker is asked to choose the best solution from the archive set and the end points (M in number) of the non-dominated front in the parent population are selected as other members. Therefore, for M number of objectives we have $\eta = M + 1$. Now, $M + 1$ number of different hyperplanes can be constructed using $M + 1$ points in an M-dimensional hyperspace. From the set of $M+1$ hyperplanes the hyperplane not containing the best point from the archive is removed which leaves us with remaining M hyperplanes. Since all the points are non-dominated with respect to each other, the normals to all the M hyperplanes will have positive direction cosines. The M planes together form a polyhedral cone with the best member from the archive as the vertex. Each hyperplane represents one of the sides of the polyhedral cone.

For instance figure 1 and figure 2 show the polyhedral cones in two and three dimensions. The equation of each hyperplane can be written as $P_i(f_1, \ldots, f_M) = 0, i \in \{1, \ldots, M\}$. If a given point $(f_1^{(1)}, \ldots, f_M^{(1)})$, in the objective space has $P_i >$

$0 \,\forall\, i \in \{1, \ldots, M\}$, then the point lies inside the polyhedral cone otherwise it lies outside. In figure 1, the shaded region has $P_i < 0$ for at least one $i \in \{1, \ldots, M\}$ and the unshaded region has $P_i > 0 \,\forall\, i \in \{1, \ldots, M\}$. In figure 2, the shaded polyhedral cone represents $P_i < 0 \,\forall\, i \in \{1, \ldots, M\}$ and the unshaded polyhedral cone represents $P_i > 0 \,\forall\, i \in \{1, \ldots, M\}$. The region outside the two cones has $P_i < 0$ for at least one $i \in \{1, \ldots, M\}$

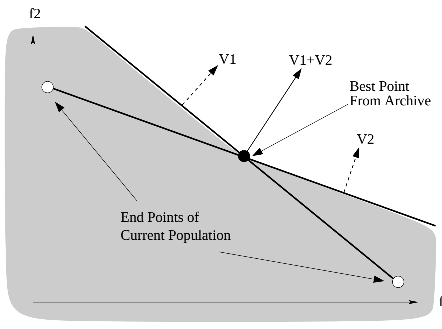

Fig. 1. Cone in two dimensions.

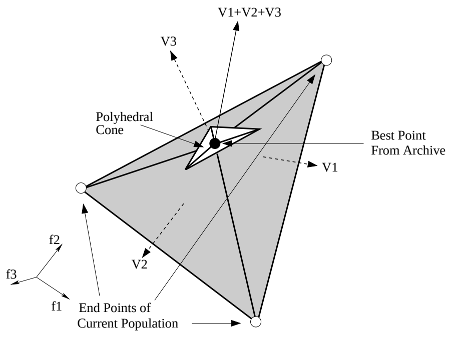

Fig. 2. Polyhedral cone in three dimensions.

3.2 Step 4: Termination Criterion

Once the polyhedral cone is determined, it provides an idea for a search direction. The normal unit vectors ($\hat{V_i}$) of all the M hyperplanes can be summed up to get a search direction ($\boldsymbol{W} = \sum_{i=1}^{n} \hat{V_i}$). $\hat{W}$ is the unit vector along $\boldsymbol{W}$ and $W_i \;\; \forall \;\; i \in \{1, \ldots, M\}$ represents the direction cosines of the vector $\boldsymbol{W}$. This direction has been used to determine if the optimization process should be terminated or not. To implement this idea we perform a single-objective search along the identified direction.

We solve the following achievement scalarizing function (ASF) problem [24] for the best point from the archive $A_t^{best} = \mathbf{z}^b$:

$$\begin{aligned} &\text{Maximize } \left(\min_{i=1}^{M} \frac{f_i(\mathbf{x}) - z_i^b}{W_i} \right) + \rho \sum_{j=1}^{M} \frac{f_j(\mathbf{x}) - z_j^b}{W_j}. \\ &\text{subject to } \mathbf{x} \in \mathcal{S}. \end{aligned} \tag{1}$$

Here $\mathcal{S}$ denotes the feasible decision variable space of the original problem. The second term with a small ρ ($= 10^{-10}$ is suggested) prevents the solution from converging to a weak Pareto-optimal point. Any single-objective optimization method can be used for solving the above problem and the intermediate solutions ($\mathbf{z}^{(i)}, \; i = 1, 2, \ldots$) can be recorded. If at any intermediate point, the Euclidean distance between $\mathbf{z}^{(i)}$ from A_t^{best} is larger than a termination parameter d_s, we stop the ASF optimization task and continue with the EMO algorithm. In this case, we replace A_t^{best} with $\mathbf{z}^{(i)}$ in the archive set, and update the archive set A_t, by deleting the dominated members. A_t^{best} replaces the member closest to it in the parent population Par_t. Figure 3 depicts this scenario. On the other hand, if at the end of the SQP run, the final SQP solution (say,

$\mathbf{z}^T$) is not greater than d_s distance away from A_t^{best}, we terminate the EMO algorithm and declare $\mathbf{z}^T$ as the final preferred solution. This situation indicates that based on the search direction, there does not exist any solution in the search space which will provide a significantly better solution than A_t^{best}. Hence, we can terminate the optimization run. Figure 4 shows such a situation, warranting a termination of the PI-EMO procedure.

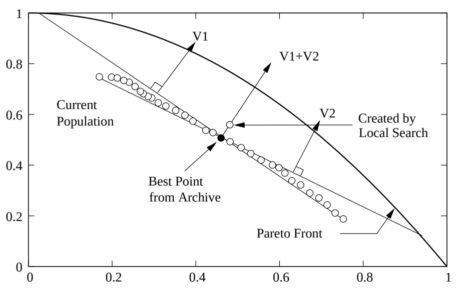

Fig. 3. Local search, when far away from the front, finds a better point more than distance d_s away from the best point. Hence, no termination of the P-EMO.

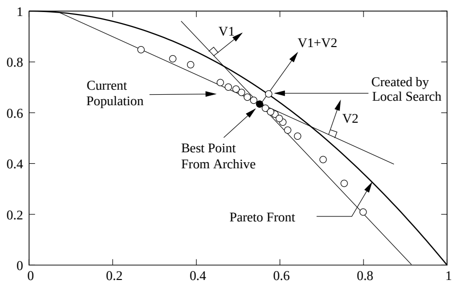

Fig. 4. Local search terminates within distance d_s from the best point. Hence, the P-EMO is terminated.

3.3 Steps 5 and 6: Modified Domination Principle

The polyhedral cone has been used to modify the domination principle in order to emphasize and create preferred solutions.

Let us assume that the polyhedral cone from the most recent decision-making interaction is represented by a set of hyperplanes $P_i(f_1, \ldots, f_M) = 0$ for $i \in \{1, \ldots, M\}$. Then, any two feasible solutions $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ can be compared with their objective function values by using the following modified domination criteria:

1. If for both the solutions, $P_i(f_1, \ldots, f_M) > 0 \quad \forall \quad i \in \{1, \ldots, M\}$, then the two points are compared based on the usual dominance principle.
2. If for both the solutions, $P_i(f_1, \ldots, f_M) < 0$ for at least one $i \in \{1, \ldots, M\}$, then the two points are compared based on the usual dominance principle.
3. If one solution has $P_i(f_1, \ldots, f_M) > 0 \quad \forall \quad i \in \{1, \ldots, M\}$, and the other solution has $P_i(f_1, \ldots, f_M) < 0$ for at least one $i \in \{1, \ldots, M\}$, then the former dominates the latter.

Figure 5 illustrates the region dominated by two points A and B. The cone formed by the linear equations have been shown. The point A lies in the region in which $P_i(f_1, \ldots, f_M) < 0$ for at least one $i \in \{1, \ldots, M\}$. The region dominated by point A is shaded. This dominated area is identical to that which can be obtained using the usual domination principle. However, point B lies in the region $P_i(f_1, \ldots, f_M) > 0$ for $i \in \{1, \ldots, M\}$. For this point, the dominated region is different from that which would be obtained using the usual domination principle. In addition to the usual region of dominance, the dominated region includes all points which have $P_i(f_1, \ldots, f_M) < 0$ for at least one $i \in \{1, \ldots, M\}$.

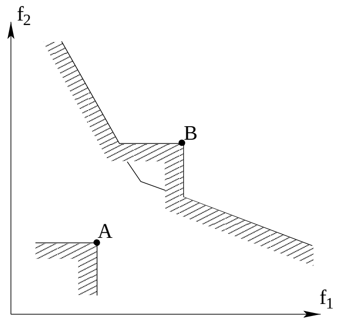

Fig. 5. Dominated regions of two points A and B using the modified definition.

Although we do not handle constrained problems in this study, the above modified domination principle can be extended for handling constraints. As defined in [18], when both solutions under consideration for a domination check are *feasible*, the above domination principle can simply be used to establish dominance of one over the other. However, if one point is feasible and the other is not, the feasible solution can be declared as dominating the other. Finally, if both points are infeasible, the one having smaller overall constraint violation may be declared as dominating the other. We defer consideration of a constrained PI-EMO to a later study.

4 PI-NSGA-II-PC Procedure

In the PI-NSGA-II-PC procedure, the first τ generations are performed according to the usual NSGA-II algorithm [18]. Thereafter, we modify the NSGA-II algorithm by using the modified domination principle (discussed in Section 3.3) in the elite-preserving operator and also in the tournament selection for creating the offspring population. We also use a different recombination operator in this study. After a child solution $\mathbf{x}^C$ is created by the SBX (recombination) operator [25], two randomly selected population members $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ are chosen and a small fraction of the difference vector is added to the child solution (similar in principle to a differential evolution operator [26]), as follows:

$$\mathbf{x}^C = \mathbf{x}^C + 0.1\left(\mathbf{x}^{(1)} - \mathbf{x}^{(2)}\right). \tag{2}$$

The crowding distance operator of NSGA-II has been replaced with k-mean clustering for maintaining diversity among solutions of the same non-dominated front.

An archive A is maintained which contains all the non-dominated members found in the current as well as the previous iterations of the optimization run. The maximum size the archive can have is $|A|^{max}$. This makes sure that none of the non-dominated solutions generated is lost even if the decision maker makes an error while providing preference information.

For termination check (discussed in Section 3.2), we use the SQP code of KNITRO [27] software to solve the single objective optimization problem and the SQP algorithm is terminated (if not terminated due to d_s distance check from A_t^{best} discussed earlier) when the KKT error measure is less than or equal to 10^{-6}.

5 Results

In this section we present the results of the PI-NSGA-II procedure on two, three, and five objective test problems. ZDT1 and DTLZ2 test problems are adapted to create maximization problems. In all simulations, we have used the following parameter values:

1. Number of generations between two consecutive DM calls: $\tau = 5$.

2. Termination parameter: $d_s = 0.01$ and $d_s = 0.1$.
3. Crossover probability and the distribution index for the SBX operator: $p_c = 0.9$ and $\eta_c = 15$.
4. Mutation probability: $p_m = 0.1$.
5. Population size: $N = 10M$, where M is the number of objectives.
6. Maximum Archive Size: $A^{max} = 10N$, where N is the population size.

In the next section, we perform a parametric study with some of the above parameters. Here, we present the test problems and results obtained with the above setting.

5.1 Two-Objective Test Problem

Problem 1 is adapted from ZDT1 and has 30 variables.

$$\begin{aligned} \text{Maximize } \mathbf{f}(\mathbf{x}) &= \left\{ \begin{array}{c} x_1 \\ \frac{10-\sqrt{x_1 g(\mathbf{x})}}{g(\mathbf{x})} \end{array} \right\}, \\ \text{where } g(\mathbf{x}) &= 1 + \tfrac{9}{29}\textstyle\sum_{i=2}^{30} x_i, \\ 0 \le x_i \le 1, &\quad \text{for } i = 1, 2, \ldots, 30, \end{aligned} \tag{3}$$

The Pareto-optimal front is given by $f_2 = 10 - \sqrt{f_1}$ and is shown in Figure 6. The solutions are $\mathbf{x}_i = 0$ for $i = 2, 3, \ldots, 30$ and $x_1 \in [0, 1]$.

This maximization problem has a non-convex front. In order to emulate the decision maker, in our simulations, we assume a particular value function which acts as a representative of the DM, but the information is not explicitly used in creating new solutions by the operators of the PI-NSGA-II procedure. In such cases, the most preferred point $\mathbf{z}^*$ can be determined from the chosen value function beforehand, thereby enabling us to compare our obtained point with $\mathbf{z}^*$.

In our study, we assume the following non-linear value function (which acts as a DM) in finding the best point from the archive at every τ generations:

$$V(f_1, f_2) = \frac{1}{(f_1 - 0.35)^2 + (f_2 - 9.6)^2}. \tag{4}$$

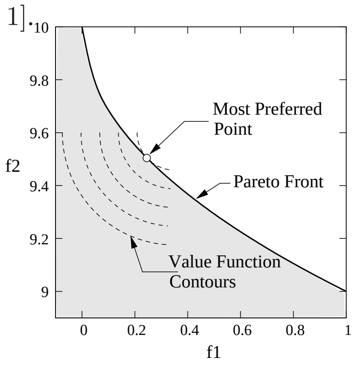

Fig. 6. Contours of the chosen value function (acts as a DM) and the most preferred point corresponding to the value function.

This value function gives the most preferred solution as $\mathbf{z}^* = (0.25, 9.50)$. The contours of this value function are shown in Figure 6.

Table 1 presents the best, median and worst of 21 different PI-NSGA-II simulations (each starting with a different initial population). The performance (accuracy measure) is computed based on the Euclidean distance of each optimized point with $\mathbf{z}^*$. Note that this accuracy measure is different from the termination criterion used in the PI-NSGA-II procedure. Results have been presented for two values of the termination criteria d_s. As expected, when the termination criteria is relaxed from 0.01 to 0.1, the

Table 1. Final solutions obtained by PI-NSGA-II for the modified ZDT1 problem.

		$d_s = 0.01$			$d_s = 0.1$		
	$\mathbf{z}^*$	Best	Median	Worst	Best	Median	Worst
f_1	0.2500	0.2482	0.2466	0.2401	0.2616	0.2733	0.3355
f_2	9.5000	9.5018	9.5034	9.5101	9.4885	9.4772	9.4208

accuracy reduces, the number of function evaluations as well as the number of DM calls also reduce. Table 2 shows minimum, median and maximum accuracy, the number of overall function evaluations, and the number of DM calls recorded in the 21 runs. The table indicates that the proposed PI-NSGA-II procedure is able to find a solution close to the final preferred solution.

Table 2. Distance of obtained solution from the most preferred solution, function evaluations, and the number of DM calls required by the PI-NSGA-II for the modified ZDT1 problem.

	$d_s = 0.01$			$d_s = 0.1$		
	Minimum	Median	Maximum	Minimum	Median	Maximum
Accuracy	0.0020	0.0048	0.0142	0.0170	0.0326	0.0726
Func. Evals.	5680	7698	11202	4159	6052	11176
# of DM Calls	15	20	29	10	14	25

5.2 Three-Objective Test Problem

The DTLZ2 test problem [28] is scalable to any number of objectives. In the three-objective case, all points (objective vectors) are bounded by two spherical surfaces in the first octant. In the case of minimizing all objectives, the inner surface (close to the origin) becomes the Pareto-optimal front. But here, we maximize each objective of the DTLZ2 problem. Thus, the outer spherical surface becomes the corresponding Pareto-optimal front. An M-objective DTLZ2 problem for maximization is given as follows:

$$\begin{array}{l}\text{Maximize } \mathbf{f}(\mathbf{x}) = \\ \left\{\begin{array}{l}(1.0+g(\mathbf{x}))\cos(\frac{\pi}{2}x_1)\cos(\frac{\pi}{2}x_2)\cdots\cos(\frac{\pi}{2}x_{M-1}) \\ (1.0+g(\mathbf{x}))\cos(\frac{\pi}{2}x_1)\cos(\frac{\pi}{2}x_2)\cdots\sin(\frac{\pi}{2}x_{M-1}) \\ \vdots \\ (1.0+g(\mathbf{x}))\cos(\frac{\pi}{2}x_1)\sin(\frac{\pi}{2}x_2) \\ (1.0+g(\mathbf{x}))\sin(\frac{\pi}{2}x_1)\end{array}\right\}, \\ \text{subject to } 0 \le x_i \le 1, \quad \text{for } i = 1,\ldots,12, \\ \quad \text{where } g(\mathbf{x}) = \sum_{i=3}^{12}(x_i - 0.5)^2.\end{array} \qquad (5)$$

The Pareto-optimal front for a three-objective DTLZ2 problem is shown in Figure 8. The points (objective vectors) on the Pareto-optimal front follow the relation: $f_1^2 +$

$f_2^2 + f_3^2 = 3.5^2$. The decision variable values correspond to $x_1 \in [0,1]$, $x_2 \in [0,1]$ and $x_i = 0$ or 1 for $i = 3, 4, \ldots, 12$.

To test the working of PI-NSGA-II on this problem, we have replaced the decision maker by using a linear value function (emulating the DM), as follows:

$$V(f_1, f_2, f_3) = 1.25 f_1 + 1.50 f_2 + 2.9047 f_3. \tag{6}$$

This value function produces the most preferred solution on the Pareto-optimal front as $\mathbf{z}^* = (1.25, 1.50, 2.9047)$.

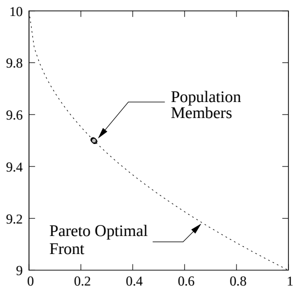

Fig. 7. Final population members after termination of the algorithm for two-objective modified ZDT1 problem.

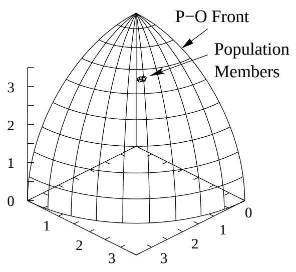

Fig. 8. Final population members after termination of the algorithm for three-objective modified DTLZ2 problem.

The PI-NSGA-II is run with $N = 10 \times 3$ or 30 population members 21 times, each time with a different random initial population. In terms of the accuracy measure from $\mathbf{z}^*$, Table 3 presents the best, median and worst performing runs. Results have been presented for two values of parameter d_s. Table 4 shows the accuracy, number of overall function evaluations and number of DM calls needed by the procedure. It is clear that the obtained points are close to the most preferred point $\mathbf{z}^*$. Figure 8 shows the population at the final generation of a typical PI-NSGA-II run.

Table 3. Final solutions obtained by PI-NSGA-II for the three-objective modified DTLZ2 problem.

		$d_s = 0.01$			$d_s = 0.1$		
	$\mathbf{z}^*$	Best	Median	Worst	Best	Median	Worst
f_1	1.2500	1.2474	1.2444	1.2367	1.2388	1.2159	1.1434
f_2	1.5000	1.4971	1.4956	1.4835	1.5445	1.5912	1.7482
f_3	2.9047	2.9074	2.9094	2.9189	2.8861	2.8705	2.8083

Table 4. Distance of obtained solution from the most preferred solution, number of function evaluations, and number of DM calls required by PI-NSGA-II on the three-objective modified DTLZ2 problem.

	$d_s = 0.01$			$d_s = 0.1$		
	Minimum	Median	Maximum	Minimum	Median	Maximum
Accuracy	0.0048	0.0085	0.0255	0.0495	0.1032	0.2868
Func. Evals.	4125	6514	8227	2577	3544	5223
# of DM Calls	14	22	34	8	10	14

5.3 Five-Objective Test Problem

We now consider the five-objective ($M = 5$) version of the DTLZ2 problem described in the previous subsection. The Pareto-optimal front is described as $f_1^2 + f_2^2 + f_3^2 + f_4^2 + f_5^2 = 3.5^2$.

For this problem, we choose a non-linear DM-emulated value function, as follows:

$$V(\mathbf{f}) = \frac{1}{(f_1 - 1.1)^2 + (f_2 - 1.21)^2 + (f_3 - 1.43)^2 + (f_4 - 1.76)^2 + (f_5 - 2.6468)^2} \quad (7)$$

This value function produces the most preferred point as $\mathbf{z}^* = (1.0, 1.1, 1.3, 1.6, 2.4062)$.

Table 5 presents the obtained solutions by PI-NSGA-II with 50 population members. Once again, we present the results for two different values of the termination parameter d_s. Table 6 shows the accuracy measure, the number of overall function evaluations, and the number of DM calls. Although the points close to the most preferred point are obtained in each run, the higher dimensionality of the problem requires more function evaluations and DM calls compared to two and three-objective test problems. When, the above results are computed for a strict termination criterion with $d_s = 0.01$, we observe a very high number of DM calls. However, with a relaxed value of $d_s = 0.1$ a much smaller number of DM calls and evaluations are required.

Table 5. Final objective values obtained from PI-NSGA-II for the five-objective modified DTLZ2 problem.

		$d_s = 0.01$			$d_s = 0.1$		
	$\mathbf{z}^*$	Best	Median	Worst	Best	Median	Worst
f_1	1.0000	0.9915	0.9721	0.8919	1.0220	1.0368	1.0893
f_2	1.1000	1.1041	1.1112	1.1373	1.1130	1.1324	1.2136
f_3	1.3000	1.2963	1.2942	1.2881	1.3072	1.3155	1.3382
f_4	1.6000	1.5986	1.5966	1.5918	1.6115	1.6346	1.7164
f_5	2.4062	2.4108	2.4179	2.4430	2.3793	2.3432	2.2031

It is worth mentioning that the application of an EMO (including NSGA-II) will face difficulties in converging to the five-dimensional Pareto-optimal front with an identical number of function evaluations.

Table 6. Distance of obtained solution from the most preferred solution, function evaluations, and the number of DM calls required by PI-NSGA-II for the five-objective modified DTLZ2 problem.

	$d_s = 0.01$			$d_s = 0.1$		
	Minimum	Median	Maximum	Minimum	Median	Maximum
Accuracy	0.0112	0.0329	0.1210	0.0395	0.0884	0.2777
Func. Evals.	20272	29298	37776	5083	6872	9919
# of DM Calls	51	69	96	9	12	17

6 Parametric Study

Besides the usual parameters associated with an evolutionary algorithm, such as population size, crossover and mutation probabilities and indices, tournament size etc., in the proposed PI-NSGA-II we have introduced a few additional parameters which may effect the accuracy and number of DM calls. They are the number of generations between DM calls (τ), termination parameter (d_s), maximum archive size ($|A|^{max}$), KKT error limit for terminating SQP algorithm in single-objective optimization used for the termination check, and the parameter ρ used in the ASF function optimization. Of these parameters, the first two have shown to have an effect on the chosen performance measures — accuracy, the number of overall function evaluations, and the number of DM calls.

A parametric study for d_s has not been done in this section as results for two different values of d_s have already been presented in the previous section. The results show an expected behavior, that is, a strict d_s provides higher accuracy and requires a larger number of DM calls and function evaluations, a relaxed d_s provides lower accuracy and requires less number of DM calls and function evaluations.

Thus, in this section, we study the effect of the parameter τ, while keeping $d_s = 0.01$ and the other PI-NSGA-II parameters identical to that mentioned in the previous section. Here, we use the two objective ZDT1 and three objective DTLZ2 test problems.

6.1 Effect of Frequency of DM Calls (τ)

We study the effect of τ by considering four different values: 2, 5, 10 and 20 generations. The parameter d_s is kept fixed at 0.01. To investigate the dependence of the performance of the procedure on the initial population, in each case, we run PI-NSGA-II from 21 different initial random populations and plot the best, median and worst performance measures.

We plot three different performance measures — accuracy, number of DM calls and number of function evaluations obtained for the modified ZDT1 problem in Figure 9. It is interesting to note that all three median performance measures are best for $\tau = 5$. A small value of τ means that DM calls are to be made more frequently. Clearly, this results in higher number of DM calls, as evident from the figure. Frequent DM calls result in more single-objective optimization runs for termination check, thereby increasing the number of overall function evaluations. On the other hand, a large value of τ captures too little information from the DM to focus the search near the most

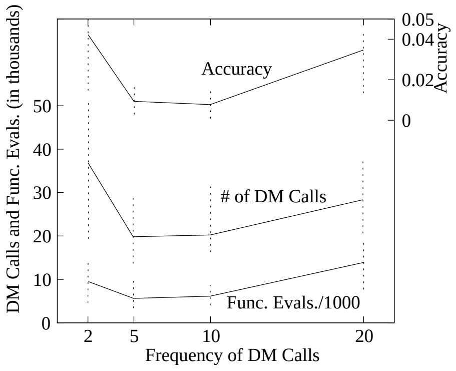

Fig. 9. Three performance measures on modified ZDT1 problem for different τ values.

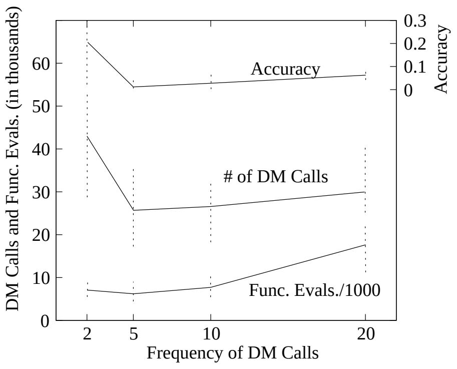

Fig. 10. Three performance measures on three-objective modified DTLZ2 problem for different τ values.

preferred point, thereby causing a large number of generations to satisfy termination conditions and a large number of DM calls.

Figure 10 shows the same three performance measures on the three-objective modified DTLZ2 problem. For this problem, the number of DM calls is minimum for $\tau = 5$ and accuracy and the number of function evaluations are also better for $\tau = 5$ generations. Once again, too small or too large τ is found to be detrimental.

Based on these simulation studies on two and three-objective optimization problems, one can conclude that a value of τ close to 5 generations is better in terms of an overall performance of the PI-NSGA-II procedure. This value of τ provides a good convergence accuracy, requires less function evaluations, and less DM calls to converge near the most preferred point.

7 Conclusions

In this paper, we have proposed a preference based evolutionary multi-objective optimization (PI-EMO) procedure which uses a polyhedral cone to modify domination. It accepts preference information from the decision maker in terms of the best solution from the archive set. The preference information from the decision maker and information from the non-dominated set of the parent population of the evolutionary algorithm have been used together to construct a polyhedral cone. Progressive information from the population of the evolutionary algorithm as well as the decision maker is used to modify the polyhedral cone after every few iterations. This approach helps in approaching towards the most preferred point on the Pareto-front by focussing the search on the region of interest.

The direction provided by the cone has been used to develop a termination criterion for the algorithm. The procedure has then been applied to three different test-problems involving two, three and five objectives. The procedure has been successful in finding the most preferred solution corresponding to the DM-emulated utility function. A parametric study has also been performed to determine the optimal settings. The parametric study gives an insight about the trade-off between the number of calls to the decision maker, number of function evaluations and the accuracy of the solution obtained.

8 Acknowledgements

Authors wish to thank the Academy of Finland (Grant Number: 121980) for their support of this study.

References

1. K. Deb, *Multi-objective optimization using evolutionary algorithms*. Chichester, UK: Wiley, 2001.
2. C. A. C. Coello, D. A. VanVeldhuizen, and G. Lamont, *Evolutionary Algorithms for Solving Multi-Objective Problems*. Boston, MA: Kluwer, 2002.
3. K. Deb and D. Saxena, “Searching for Pareto-optimal solutions through dimensionality reduction for certain large-dimensional multi-objective optimization problems,” in *Proceedings of the World Congress on Computational Intelligence (WCCI-2006)*, 2006, pp. 3352–3360.
4. K. Deb, A. Sinha, P. Korhonen and J. Wallenius, “An Interactive Evolutionary Multi-Objective Optimization Method Based on Progressively Approximated Value Functions,” in *Technical Report Kangal Report No. 2009005, Kanpur, India: Department of Mechanical Engineering, Indian Institute of Technology Kanpur. http://www.iitk.ac.in/kangal/pub.htm*
5. J. Knowles and D. Corne, “Quantifying the effects of objective space dimension in evolutionary multiobjective optimization,” in *Proceedings of the Fourth International Conference on Evolutionary Multi-Criterion Optimization (EMO-2007)*, 2007, pp. 757–771, (LNCS 4403).
6. J. Branke, T. Kaussler, and H. Schmeck, “Guidance in evolutionary multi-objective optimization,” *Advances in Engineering Software*, vol. 32, pp. 499–507, 2001.
7. J. Branke and K. Deb, “Integrating user preferences into evolutionary multi-objective optimization,” in *Knowledge Incorporation in Evolutionary Computation*, Y. Jin, Ed. Heidelberg, Germany: Springer, 2004, pp. 461–477.
8. K. Deb, J. Sundar, N. Uday, and S. Chaudhuri, “Reference point based multi-objective optimization using evolutionary algorithms,” *International Journal of Computational Intelligence Research (IJCIR)*, vol. 2, no. 6, pp. 273–286, 2006.
9. L. Thiele, K. Miettinen, P. Korhonen, and J. Molina, “A preference-based interactive evolutionary algorithm for multiobjective optimization,” *Evolutionary Computation Journal*, in press.
10. K. Deb and A. Kumar, “Interactive evolutionary multi-objective optimization and decision-making using reference direction method,” in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2007)*. New York: The Association of Computing Machinery (ACM), 2007, pp. 781–788.
11. ——, “Light beam search based multi-objective optimization using evolutionary algorithms,” in *Proceedings of the Congress on Evolutionary Computation (CEC-07)*, 2007, pp. 2125–2132.
12. A. Jaszkiewicz and J. Branke, “Interactive multiobjective evolutionary algorithms,” in *Multiobjective optimization – Interactive and Evolutionary Approaches (LNCS volume 5252)*, J. Branke, K. Deb, K. Miettinen, and R. Slowinski, Eds. Heidelberg: Springer, 2008, pp. 179–193.

13. P. Korhonen and J. Laakso, “A visual interactive method for solving the multiple criteria problem,” *European Journal of Operational Reseaech*, vol. 24, pp. 277–287, 1986.
14. P. Korhonen and G. Y. Yu, “A reference direction approach to multiple objective quadratic-linear programming,” *European Journal of Operational Reseaech*, vol. 102, pp. 601–610, 1997.
15. J. Branke, S. Greco, R. Slowinski, and P. Zielniewicz, “Interactive evolutionary multiobjective optimization using robust ordinal regression,” in *Proceedings of the Fifth International Conference on Evolutionary Multi-Criterion Optimization (EMO-09)*. Berlin: Springer-Verlag, 2009, pp. 554–568.
16. P. Korhonen, H. Moskowitz, and J. Wallenius, “A progressive algorithm for modeling and solving multiple-criteria decision problems,” *Operations Research*, vol. 34, no. 5, pp. 726–731, 1986.
17. P. Korhonen, H. Moskowitz, P. Salminen, and J. Wallenius, “Further developments and tests of a progressive algorithm for multiple criteria decision making,” *Operations Research*, vol. 41, no. 6, pp. 1033–1045, 1993.
18. K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, “A fast and elitist multi-objective genetic algorithm: NSGA-II,” *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.
19. G. W. Greenwood, X. Hu, and J. G. D’Ambrosio, “Fitness functions for multiple objective optimization problems: Combining preferences with pareto rankings,” in *Foundations of Genetic Algorithms (FOGA)*. San Mateo: Morgan Kauffman, 1996, pp. 437–455.
20. S. Phelps and M. Koksalan, “An interactive evolutionary metaheuristic for multiobjective combinatorial optimization,” *Management Science*, vol. 49, no. 12, pp. 1726–1738, December 2003.
21. A. Jaszkiewicz, “Interactive multiobjective optimization with the pareto memetic algorithm,” *Foundations of Computing and Decision Sciences*, vol. 32, no. 1, pp. 15–32, 2007.
22. E. Zitzler, M. Laumanns, and L. Thiele, “SPEA2: Improving the strength pareto evolutionary algorithm for multiobjective optimization,” in *Evolutionary Methods for Design Optimization and Control with Applications to Industrial Problems*, K. C. Giannakoglou, D. T. Tsahalis, J. Périaux, K. D. Papailiou, and T. Fogarty, Eds. Athens, Greece: International Center for Numerical Methods in Engineering (CIMNE), 2001, pp. 95–100.
23. K. Miettinen, *Nonlinear Multiobjective Optimization*. Boston: Kluwer, 1999.
24. A. P. Wierzbicki, “The use of reference objectives in multiobjective optimization,” in *Multiple Criteria Decision Making Theory and Applications*, G. Fandel and T. Gal, Eds. Berlin: Springer-Verlag, 1980, pp. 468–486.
25. K. Deb and R. B. Agrawal, “Simulated binary crossover for continuous search space,” *Complex Systems*, vol. 9, no. 2, pp. 115–148, 1995.
26. K. V. Price, R. Storn, and J. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*. Berlin: Springer-Verlag, 2005.
27. R. H. Byrd, J. Nocedal, and R. A. Waltz, *KNITRO: An integrated package for nonlinear optimization*. Springer-Verlag, 2006, pp. 35–59.
28. K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, “Scalable test problems for evolutionary multi-objective optimization,” in *Evolutionary Multiobjective Optimization*, A. Abraham, L. Jain, and R. Goldberg, Eds. London: Springer-Verlag, 2005, pp. 105–145.
29. J. W. Fowler, E. S. Gel, M. Koksalan, P. Korhonen, J. L. Marquis and J. Wallenius “Interactive Evolutionary Multi-Objective Optimization for Quasi-Concave Preference Functions,” *Submitted to European Journal of Operational Research*, 2009.
30. P. Korhonen and J. Karaivanova “An Algorithm for Projecting a Reference Direction onto the Nondominated Set of Given Points,” *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 29, pp. 429–435, 1999.

4 An efficient and accurate solution methodology for bilevel multi-objective programming problems using a hybrid evolutionary-local-search algorithm

K. Deb and A. Sinha

Evolutionary Computation Journal, 18(3), 403-449. MIT Press, 2010.

An Efficient and Accurate Solution Methodology for Bilevel Multi-Objective Programming Problems Using a Hybrid Evolutionary-Local-Search Algorithm

Kalyanmoy Deb deb@iitk.ac.in
Department of Mechanical Engineering, Indian Institute of Technology Kanpur, PIN 208016, India, and Department of Business Technology, Helsinki School of Economics, PO Box 1210, FIN-101, Helsinki, Finland

Ankur Sinha ankur.sinha@hse.fi
Department of Business Technology, Helsinki School of Economics, PO Box 1210, FIN-101, Helsinki, Finland

Abstract

Bilevel optimization problems involve two optimization tasks (upper and lower level), in which every feasible upper level solution must correspond to an optimal solution to a lower level optimization problem. These problems commonly appear in many practical problem solving tasks including optimal control, process optimization, game-playing strategy developments, transportation problems, and others. However, they are commonly converted into a single level optimization problem by using an approximate solution procedure to replace the lower level optimization task. Although there exists a number of theoretical, numerical, and evolutionary optimization studies involving single-objective bilevel programming problems, there does not exist too many studies in the context of multiple conflicting objectives in each level of a bilevel programming problem. In this paper, we address certain intricate issues related to solving multi-objective bilevel programming problems, present challenging test problems, and propose a viable and hybrid evolutionary-cum-local-search based algorithm as a solution methodology. The hybrid approach performs better than a number of existing methodologies and scales well up to 40-variable difficult test problems used in this study. The population sizing and termination criteria are made self-adaptive, so that no additional parameters need to be supplied by the user. The study clearly indicates a clear niche of evolutionary algorithms in solving such difficult problems of practical importance compared to their usual solution by a computationally expensive nested procedure. The study opens up many issues related to multi-objective bilevel programming and hopefully this study will motivate EMO and other researchers to pay more attention to this important and difficult problem solving activity.

Keywords

Bilevel optimization, evolutionary multi-objective optimization, NSGA-II, test problem development, problem difficulties, hybrid evolutionary algorithms, self-adaptive algorithm, sequential quadratic programming.

1 Introduction

Optimization problems are usually considered to have a single level consisting of one or more objective functions which are to be optimized, and constraints, if present, must be satisfied (Reklaitis *et al.*, 1983; Rao, 1984). Optimization problems come in many different forms and complexities involving the type and size of the variable vector, objective and constraint functions, nature of problem parameters, modalities of objective functions, interactions among objectives, extent of feasible search region, computational burden and inherent noise in evaluating solutions, etc. (Deb, 2001). While these factors are keeping optimization researchers and practitioners busy in devising efficient solution procedures, the practice always seems to have more to offer than what the researchers have been able to comprehend and implement in the realm of optimization studies.

Bilevel programming problems have two levels of optimization problems – upper and lower levels (Colson *et al.*, 2007; Vicente and Calamai, 2004). In the upper level optimization task, a solution, in addition to satisfying its own constraints, must also be an optimal solution to another optimization problem, called the lower level optimization problem. Although the concept is intriguing, bilevel programming problems commonly appear in many practical optimization problems (Bard, 1998). Thinking simply, the bilevel scenario occurs when a solution in an (upper level) optimization task must be a physically or a functionally acceptable solution, such as being a stable solution or being a solution in equilibrium or being a solution which must satisfy certain conservation principles, etc. Satisfaction of one or more of these conditions can then be posed as another (lower level) optimization task. However, often in practice (Bianco *et al.*, 2009; Dempe, 2002; Pakala, 1993), such problems are not usually treated as bilevel programming problems, instead some approximate methodologies are used to replace the lower level problem. In many scenarios it is observed that approximate solution methodologies are not available or practically and functionally unacceptable. Ideally such problems involving an assurance of a physically or functionally viable solution must be posed as bilevel programming problems and solved.

Bilevel programming problems involving a single objective function in upper and lower levels have received some attention from theory (Dempe *et al.*, 2006), algorithm development and application (Alexandrov and Dennis, 1994; Vicente and Calamai, 2004), and even using evolutionary algorithms (Yin, 2000; Wang *et al.*, 2008). However, apart from a few recent studies (Eichfelder, 2007, 2008; Halter and Mostaghim, 2006; Shi and Xia, 2001) and our recent evolutionary multi-objective optimization (EMO) studies (Deb and Sinha, 2009a,b; Sinha and Deb, 2009), multi-objective bilevel programming studies are scarce in both classical and evolutionary optimization fields. The lack of interests for handling multiple conflicting objectives in a bilevel programming context is not due to lack of practical problems, but more due to the need for searching and storing multiple trade-off lower level solutions for a single upper level solution and due to the complex interactions which upper and lower level optimization tasks can provide. In this paper, we make a closer look at the intricacies of multi-objective bilevel programming problems, present a set of difficult test problems by using an extended version of our earlier proposed test problem construction procedure, and propose and evaluate a hybrid EMO-cum-local-search bilevel programming algorithm (H-BLEMO).

In the remainder of this paper, we briefly outline a generic multi-objective bilevel optimization problem and then provide an overview of existing studies both on single and multi-objective bilevel programming. Past evolutionary methods are particularly highlighted. Thereafter, we list a number of existing multi-objective bilevel test problems and then discuss an extension of our recent suggestion. The proposed hybrid and self-adaptive bilevel evolutionary multi-objective optimization algorithm (H-BLEMO) is then described in detail by providing a step-by-step procedure. Simulation results on eight different problems are shown. A comparison of the performance of the proposed algorithm with our previously-proposed methodology and with a nested optimization strategy is made. The test problem construction procedure allowed us to create problems which exhibit conflicting goals between lower and upper level optimization tasks. The use of local search and self-adaptive methodologies enabled us to solve such difficult problems using H-BLEMO, whereas these same problems are found to be unsolvable by our earlier methods. Further, a scalability study of the proposed algorithm is made by considering different problem sizes ranging from 10 to 40 decision variables. Finally, conclusions of the study are presented.

2 Multi-Objective Bilevel Optimization Problems

A multi-objective bilevel optimization problem has two levels of multi-objective optimization problems such that a feasible solution of the upper level problem must be a member of the Pareto-optimal set of the lower level optimization problem. A general multi-objective bilevel optimization problem can be described as follows:

$$\begin{array}{rl} \text{Minimize}_{(\mathbf{x}_u,\mathbf{x}_l)} & \mathbf{F}(\mathbf{x}) = (F_1(\mathbf{x}),\ldots,F_M(\mathbf{x}))\,, \\ \text{subject to} & \mathbf{x}_l \in \text{argmin}_{(\mathbf{x}_l)} \left\{ \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}),\ldots,f_m(\mathbf{x})) \middle| \mathbf{g}(\mathbf{x}) \geq \mathbf{0}, \mathbf{h}(\mathbf{x}) = \mathbf{0} \right\}, \\ & \mathbf{G}(\mathbf{x}) \geq \mathbf{0}, \mathbf{H}(\mathbf{x}) = \mathbf{0}, \\ & x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = 1,\ldots,n. \end{array} \tag{1}$$

In the above formulation, $F_1(\mathbf{x}),\ldots,F_M(\mathbf{x})$ are upper level objective functions and $f_1(\mathbf{x}),\ldots,f_m(\mathbf{x})$ are lower level objective functions. The functions $\mathbf{g}(\mathbf{x})$ and $\mathbf{h}(\mathbf{x})$ determine the feasible space for the lower level problem. The decision vector is $\mathbf{x}$ which comprises of two smaller vectors $\mathbf{x}_u$ and $\mathbf{x}_l$, such that $\mathbf{x} = (\mathbf{x}_u, \mathbf{x}_l)$. It should be noted that the lower level optimization problem is optimized only with respect to the variables $\mathbf{x}_l$ and the variables $\mathbf{x}_u$ act as fixed parameters for the problem. Therefore, the solution set of the lower level problem can be represented as a function of $\mathbf{x}_u$, or as $\mathbf{x}_l^*(\mathbf{x}_u)$. This means that the upper level variables ($\mathbf{x}_u$), act as a parameter to the lower level problem and hence the lower level optimal solutions ($\mathbf{x}_l^*$) are a function of the upper level vector $\mathbf{x}_u$. The functions $\mathbf{G}(\mathbf{x})$ and $\mathbf{H}(\mathbf{x})$ along with the Pareto-optimality to the lower level problem determine the feasible space for the upper level optimization problem. Both sets $\mathbf{x}_l$ and $\mathbf{x}_u$ are decision variables for the upper level problem.

2.1 Practical Importance

Bilevel multi-objective optimization problems arise from hierarchical problems in practice in which a strategy for solving the overall system depends on optimal strategies of solving a number of subsystems. Let us consider two different examples to illustrate these problems.

Many engineering design problems in practice involve an upper level optimization problem requiring that a feasible solution to the problem must satisfy certain physical conditions, such as satisfying a network flow balance or satisfying stability conditions or satisfying some equilibrium conditions. If simplified mathematical equations for such conditions are easily available, often they are directly used as constraints and the lower level optimization task is avoided. But in many problems establishing whether a solution is stable or in equilibrium can be established by ensuring that the solution is an optimal solution to a derived optimization problem. Such a derived optimization problem can then be formulated as a lower level problem in a bilevel optimization problem. A common source of bilevel problems is in chemical process optimization problems, in which the upper level problem optimizes the overall cost and quality of product, whereas the lower level optimization problem optimizes error measures indicating how closely the process adheres to different theoretical process conditions, such as mass balance equations, cracking, or distillation principles (Dempe, 2002).

The bilevel problems are also similar in principle to the Stackelberg games (Fudenberg and Tirole, 1993; Wang and Periaux, 2001) in which a leader makes the first move and a follower then maximizes its move considering the leader's move. The leader has an advantage in that it can control the game by making its move in a way so as to maximize its own gain knowing that the follower will always maximize its own gain. For an example (Zhang *et al.*, 2007), a company CEO (leader) may be interested in maximizing net profits and quality of products, whereas heads of branches (followers) may maximize their own net profit and worker satisfaction. The CEO knows that for each of his/her strategy, the heads of branches will optimize their own objectives. The CEO must then adjust his/her own decision variables so that CEO's own objectives are maximized. Stackelberg's game model and its solutions are used in many different problem domains, including engineering design (Pakala, 1993), security applications (Paruchuri *et al.*, 2008), and others.

3 Existing Classical and Evolutionary Methodologies

The importance of solving bilevel optimization problems, particularly problems having a single objective in each level, has been recognized amply in the optimization literature. The research has been focused in both theoretical and algorithmic aspects. However, there has been a lukewarm interest in handling bilevel problems having multiple conflicting objectives in any or both levels. Here we provide a brief description of the main research outcomes so far in both single and multi-objective bilevel optimization areas.

3.1 Theoretical Developments

Several studies exist in determining the optimality conditions for a upper level solution. The difficulty arises due to the existence of another optimization problem as a hard constraint to the upper level problem. Usually the Karush-Kuhn-Tucker (KKT) conditions of the lower level optimization problems are first written and used as constraints in formulating the KKT conditions of the upper level problem, involving second derivatives of the lower level objectives and constraints as the necessary conditions of the upper level problem. However, as discussed in Dempe *et al.* (2006), although KKT optimality conditions can be written mathematically, the presence of many lower level Lagrange multipliers and an abstract term involving coderivatives makes the procedure difficult to be applied in practice.

Fliege and Vicente (2006) suggested a mapping concept in which a bilevel single-objective optimization problem (one objective each in upper and lower level problems) can be converted to an equivalent four-objective optimization problem with a special cone dominance concept. Although the idea may apparently be extended for bilevel multi-objective optimization problems, no such suggestion with an exact mathematical formulation is made yet. Moreover, derivatives of original objectives are involved in the problem formulation, thereby making the approach limited to only differentiable problems.

3.2 Algorithmic Developments

One simple algorithm for solving bilevel optimization problems using a point-by-point approach would be to directly treat the lower level problem as a hard constraint. Every solution ($\mathbf{x} = (\mathbf{x}_u, \mathbf{x}_l)$) must be sent to the lower level problem as an initial point and an optimization algorithm can then be employed to find the optimal solution $\mathbf{x}_l^*$ of the lower level optimization problem. Then, the original solution $\mathbf{x}$ of the upper level problem must be repaired as $(\mathbf{x}_u, \mathbf{x}_l^*)$. The employment of a lower level optimizer within the upper level optimizer for every upper level solution makes the overall search a *nested* optimization procedure, which may be computationally an expensive task. Moreover, if this idea is to be extended for multiple conflicting objectives in the lower level, for every upper level solution, multiple Pareto-optimal solutions for the lower level problem need to be found and stored by a suitable multi-objective optimizer.

Another idea (Herskovits *et al.*, 2000; Bianco *et al.*, 2009) of handling the lower level optimization problem having differentiable objectives and constraints is to include the explicit KKT conditions of the lower level optimization problem directly as constraints to the upper level problem. This will then involve Lagrange multipliers of the lower level optimization problem as additional variables to the upper level problem. As KKT points need not always be optimum points, further conditions must have to be included to ensure the optimality of lower level problem. For multi-objective bilevel problems, corresponding multi-objective KKT formulations need to be used, thereby involving further Lagrange multipliers and optimality conditions as constraints to the upper level problem.

Despite these apparent difficulties, there exist some useful studies, including reviews on bilevel programming (Colson *et al.*, 2007; Vicente and Calamai, 2004), test problem generators (Calamai and Vicente, 1994), nested bilevel linear programming (Gaur and Arora, 2008), and applications (Fampa *et al.*, 2008; Abass, 2005; Koh, 2007), mostly in the realm of single-objective bilevel optimization.

Recent studies by Eichfelder (2007, 2008) concentrated on handling multi-objective bilevel problems using classical methods. While the lower level problem uses a numerical optimization technique, the upper level problem is handled using an adaptive exhaustive search method, thereby making the overall procedure computationally expensive for large-scale problems. This method uses the nested optimization strategy to find and store multiple Pareto-optimal solutions for each of finitely-many upper level variable vectors.

Another study by Shi and Xia (2001) transformed a multi-objective bilevel programming problem into a bilevel ϵ-constraint approach in both levels by keeping one of the objective functions and converting remaining objectives to constraints. The ϵ values for constraints were supplied by the decision-maker as different levels of 'satisfactoriness'. Further, the lower-level single-objective constrained optimization problem was replaced by equivalent KKT conditions and a variable metric optimization method was used to solve the resulting problem.

Certainly, more efforts are needed to devise effective classical methods for multi-objective bilevel optimization, particularly to handle the upper level optimization task in a more coordinated way with the lower level optimization task.

3.3 Evolutionary Methods

Several researchers have proposed evolutionary algorithm based approaches in solving single-objective bilevel optimization problems. As early as in 1994, Mathieu *et al.* (1994) proposed a GA-based approach for solving bilevel linear programming problems having a single objective in each level. The lower level problem was solved using a standard linear programming method, whereas the upper level was solved using a GA. Thus, this early GA study used a nested optimization strategy, which may be computationally too expensive to extend for nonlinear and large-scale problems. Yin (2000) proposed another GA based nested approach in which the lower level problem was solved using the Frank-Wolfe gradient based linearized optimization method and claimed to solve non-convex bilevel optimization problems better than an existing classical method. Oduguwa and Roy (2002) suggested a coevolutionary GA approach in which two different populations are used to handle variable vectors $\mathbf{x}_u$ and $\mathbf{x}_l$ independently. Thereafter, a linking procedure is used to cross-talk between the populations. For single-objective bilevel optimization problems, the final outcome is usually a single optimal solution in each level. The proposed coevolutionary approach is viable for finding corresponding single solution in $\mathbf{x}_u$ and $\mathbf{x}_l$ spaces. But in handling multi-objective bilevel programming problems, multiple solutions corresponding to each upper level solution must be found and maintained during the coevolutionary process. It is not clear how such a coevolutionary algorithm can be designed effectively for handling multi-objective bilevel optimization problems. We do not address this issue in this paper, but recognize that Oduguwa and Roy's study (2002) was the first to suggest a coevolutionary procedure for single-objective bilevel optimization problems. Since 2005, a surge in research in this area can be found in algorithm development mostly using the nested approach and the explicit KKT conditions of the lower level problem, and in various application areas (Hecheng and Wang, 2007; Li and Wang, 2007; Dimitriou *et al.*, 2008; Yin, 2000; Mathieu *et al.*, 1994; Sun *et al.*, 2006; Wang *et al.*, 2007; Koh, 2007; Wang *et al.*, 2005, 2008).

Li *et al.* (2006) proposed particle swarm optimization (PSO) based procedures for both lower and upper levels, but instead of using a nested approach, they proposed a serial application of upper and lower levels iteratively. This idea is applicable in solving single-objective problems in each level due to the sole target of finding a single optimal solution. As discussed above, in the presence of multiple conflicting objectives in each level, multiple solutions need to be found and preserved for each upper level solution and then a serial application of upper and lower level optimization does not make sense for multi-objective bilevel optimization. Halter and Mostaghim (2006) also used PSO on both levels, but since the lower level problem in their application problem was linear, they used a specialized linear multi-objective PSO algorithm and used an overall nested optimization strategy at the upper level.

Recently, we have proposed a number of EMO algorithms through conference publications (Deb and Sinha, 2009a,b; Sinha and Deb, 2009) using NSGA-II to solve both level problems in a synchronous manner. First, our methodologies were generic so that they can be used to linear/nonlinear, convex/non-convex, differentiable/non-differentiable and single/multi-objective problems at both levels. Second, our methodologies did not use the nested approach, nor did they use a serial approach, but employed a structured intertwined evolution of upper and lower level populations. But they were computationally demanding. However, these initial studies made us understand the complex intricacies by which both level problems can influence each other. Based on this experience, in this paper, we suggest a less-structural, self-adaptive, computationally fast, and a hybrid evolutionary algorithm coupled with a local search procedure for handling multi-objective bilevel programming problems.

Bilevel programming problems, particularly with multiple conflicting objectives, should have been paid more attention than what has been made so far. As more and more studies are performed, the algorithms must have to be tested and compared against each other. This process needs an adequate number of test problems with tunable difficulties. In the next section, we suggest a generic procedure for developing test problems and suggest a test suite.

4 Test Problems for Multi-Objective Bilevel Programming

There does not exist any systematic past study in developing test problems for multi-objective bilevel programming. However, Eichfelder (2007) used a number of test problems in her study, in which some problems were not analyzed for their exact Pareto-optimal fronts. Here, we first describe some of these existing test problems (we refer to them as 'TP' here) and then discuss an extension of our recently proposed test problem development procedure (Deb and Sinha, 2009a) for any number of objectives and scenarios. The principle is used to generate five test problems (we refer to them as 'DS' here).

4.1 Problem TP1

Problem TP1 has a total of three variables with $\mathbf{x}_l = (x_1, x_2)^T$ and $\mathbf{x}_u = (y)$ (Eichfelder, 2007):

$$\begin{array}{ll} \text{Minimize} & \mathbf{F}(\mathbf{x}) = \begin{pmatrix} x_1 - y \\ x_2 \end{pmatrix}, \\ \text{subject to} & (x_1, x_2) \in \text{argmin}_{(x_1,x_2)} \left\{ \mathbf{f}(\mathbf{x}) = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \middle| g_1(\mathbf{x}) = y^2 - x_1^2 - x_2^2 \geq 0 \right\}, \\ & G_1(\mathbf{x}) = 1 + x_1 + x_2 \geq 0, \\ & -1 \leq x_1, x_2 \leq 1, \quad 0 \leq y \leq 1. \end{array} \tag{2}$$

The Pareto-optimal set for the lower-level optimization task for a fixed y is the bottom-left quarter of the circle: $\{(x_1, x_2) \in \mathbb{R}^2 \mid x_1^2 + x_2^2 = y^2, x_1 \leq 0, x_2 \leq 0\}$. The linear constraint in the upper level optimization task does not allow the entire quarter circle to be feasible for some y. Thus, at most a couple of points from the quarter circle belongs to the Pareto-optimal set of the overall problem. Eichfelder (2007) reported the following Pareto-optimal solutions for this problem:

$$\mathbf{x}^* = \left\{ (x_1, x_2, y) \in \mathbb{R}^3 \quad \middle| \quad x_1 = -1 - x_2, x_2 = -\frac{1}{2} \pm \frac{1}{4}\sqrt{8y^2 - 4}, y \in \left[\frac{1}{\sqrt{2}}, 1\right] \right\}. \tag{3}$$

Figure 1 shows the Pareto-optimal front for the problem TP1. Lower level Pareto-optimal fronts of some representative y values are also shown in the figure using dashed lines, indicating that at most two such Pareto-optimal solutions (such as points B and C for $y = 0.9$) of a lower level optimization problem become the candidate Pareto-optimal solutions of the upper level problem.

4.2 Problem TP2

We created a simplistic bilevel two-objective optimization problem elsewhere (Deb and Sinha, 2009b), having $\mathbf{x}_l = (x_1, \ldots, x_K)$ and $\mathbf{x}_u = (y)$:

$$\begin{array}{l} \text{Minimize} \quad \mathbf{F}(\mathbf{x}) = \begin{pmatrix} (x_1 - 1)^2 + \sum_{i=2}^{K} x_i^2 + y^2 \\ (x_1 - 1)^2 + \sum_{i=2}^{K} x_i^2 + (y - 1)^2 \end{pmatrix}, \\ \text{subject to} \\ (x_1, x_2, \ldots, x_K) \in \text{argmin}_{(x_1,x_2,\ldots,x_K)} \left\{ \mathbf{f}(\mathbf{x}) = \begin{pmatrix} x_1^2 + \sum_{i=2}^{K} x_i^2 \\ (x_1 - y)^2 + \sum_{i=2}^{K} x_i^2 \end{pmatrix} \right\}, \\ -1 \leq (x_1, x_2, \ldots, x_K, y) \leq 2. \end{array} \tag{4}$$

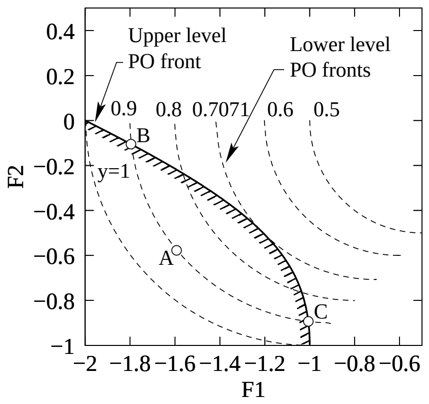

Figure 1: Pareto-optimal fronts of upper level (complete problem) and some representative lower level optimization tasks are shown for problem TP1.

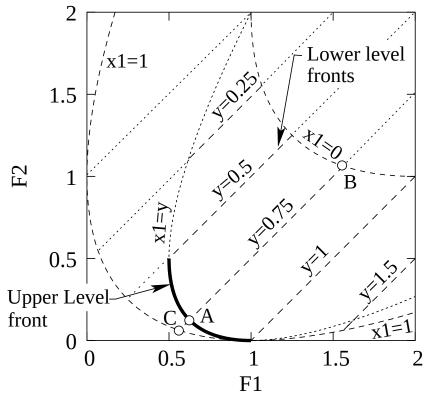

Figure 2: Pareto-optimal front of the upper level problem for problem TP2 with $x_i = 0$ for $i = 2, \ldots, K$.

For a fixed value of y, the Pareto-optimal solutions of the lower level optimization problem are given as follows: $\{\mathbf{x}_l \in \mathbb{R}^K | x_1 \in [0, y], x_i = 0, \text{ for } i = 2, \ldots, K\}$. However, for the upper level problem, Pareto-optimal solutions correspond to following conditions: $\{\mathbf{x} \in \mathbb{R}^{K+1} | x_1 = y, x_i = 0, \text{ for } i = 2, \ldots, K, y \in [0.5, 1.0]\}$. If an algorithm fails to find the true Pareto-optimal solutions of the lower level problem and ends up finding a solution below the '$x_1 = y$' curve in Figure 2 (such as solution C), it can potentially dominate a true Pareto-optimal point (such as point A) thereby making the task of finding true Pareto-optimal solutions a difficult task. We use $K = 14$ here, so that the total number of variables is 15 in this problem.

4.3 Test Problem TP3

This problem is taken from (Eichfelder, 2007):

$$
\begin{array}{l}
\text{Minimize} \quad \mathbf{F}(\mathbf{x}) = \left(\begin{array}{c} x_1 + x_2^2 + y + \sin^2(x_1 + y) \\ \cos(x_2)(0.1 + y)(\exp(-\frac{x_1}{0.1+x_2}) \end{array} \right), \\
\text{subject to} \\
(x_1, x_2) \in \left\{ \begin{array}{l} \text{argmin}_{(x_1,x_2)} \mathbf{f}(\mathbf{x}) = \left(\begin{array}{c} \frac{(x_1-2)^2+(x_2-1)^2}{4} + \frac{x_2 y + (5-y_1)^2}{16} + \sin(\frac{x_2}{10}) \\ \frac{x_1^2 + (x_2-6)^4 - 2x_1 y_1 - (5-y_1)^2}{80} \end{array} \right) \Bigg| \\ g_1(\mathbf{x}) = x_2 - x_1^2 \geq 0 \\ g_2(\mathbf{x}) = 10 - 5x_1^2 - x_2 \geq 0 \\ g_3(\mathbf{x}) = 5 - \frac{y}{6} - x_2 \geq 0 \\ g_4(\mathbf{x}) = x_1 \geq 0 \end{array} \right\}, \\
G_1(\mathbf{x}) \equiv 16 - (x_1 - 0.5)^2 - (x_2 - 5)^2 - (y - 5)^2 \geq 0, \\
0 \leq x_1, x_2, y \leq 10.
\end{array}
\quad (5)
$$

For this problem, the exact Pareto-optimal front of the lower or the upper level optimization problem are not derived mathematically. For this reason, we do not consider this problem any further here. Some results using our earlier BLEMO procedure can be found elsewhere (Deb and Sinha, 2009b).

4.4 Test Problem TP4

The next problem comes from a company scenario dealing with the management decisions between a CEO (leader) and heads of branches (follower) (Zhang *et al.*, 2007). Although fuzziness in the coefficients were used in the original study, here we present the deterministic version of the problem:

$$
\begin{array}{ll}
\text{Maximize} \quad \mathbf{F}(\mathbf{x},\mathbf{y}) = & \text{subject to } (\mathbf{x}) \in \text{argmin}_{(\mathbf{x})} \\
\left(\begin{array}{l} (1,9)(y_1,y_2)^T \\ \quad +(10,1,3)(x_1,x_2,x_3)^T \\ (9,2)(y_1,y_2)^T \\ \quad +(2,7,4)(x_1,x_2,x_3)^T \end{array}\right), &
\left\{\begin{array}{l} \mathbf{f}(\mathbf{x}) = \left(\begin{array}{l} (4,6)(y_1,y_2)^T + (7,4,8)(x_1,x_2,x_3)^T \\ (6,4)(y_1,y_2)^T + (8,7,4)(x_1,x_2,x_3)^T \end{array}\right) \Bigg| \\ g1 = (3,-9)(y_1,y_2)^T + (-9,-4,0)(x_1,x_2,x_3)^T \le 61 \\ g2 = (5,9)(y_1,y_2)^T + (10,-1,-2)(x_1,x_2,x_3)^T \le 924 \\ g3 = (3,-3)(y_1,y_2)^T + (0,1,5)(x_1,x_2,x_3)^T \le 420 \end{array}\right\}, \\
G1 = (3,9)(y_1,y_2)^T + (9,5,3)(x_1,x_2,x_3)^T \le 1039, & \\
G2 = (-4,-1)(y_1,y_2)^T + (3,-3,2)(x_1,x_2,x_3)^T \le 94, & \\
x_1, x_2, y_1, y_2, y_3 \ge 0. &
\end{array}
\tag{6}
$$

4.5 Development of Tunable Test Problems

Bilevel multi-objective optimization problems are different from single-level multi-objective optimization problems in that the Pareto-optimality of a lower level multi-objective optimization problem is a requirement to the upper level problem. Thus, while developing a bilevel multi-objective test problem, we should have ways to test an algorithm's ability to handle complexities in both lower and upper level problems independently and additionally their interactions. Further, the test problems should be such that we would have a precise knowledge about the exact location (and relationships) of Pareto-optimal points. Thinking along these lines, we outline a number of desired properties in a bilevel multi-objective test problem:

1. *Exact location of Pareto-optimal solutions in both lower and upper level problems are possible to be established.* This will facilitate a user to evaluate the performance of an algorithm easily by comparing the obtained solutions with the exact Pareto-optimal solutions.

2. *Problems are scalable with respect to number of variables.* This will allow a user to investigate whether the proposed algorithm scales well with number of variables in both lower and upper levels.

3. *Problems are scalable with respect to number of objectives in both lower and upper levels.* This will enable a user to evaluate whether the proposed algorithm scales well with the number of objectives in each level.

4. *Lower level problems are difficult to solve to Pareto-optimality.* If the lower level Pareto-optimal front is not found exactly, the corresponding upper level solution are not feasible. Therefore, these problems will test an algorithm's ability to converge to the correct Pareto-optimal front. Here, ideas can be borrowed from single-level EMO test problems (Deb *et al.*, 2005) to construct difficult lower level optimization problems. The shape (convex, non-convex, disjointedness and multi-modality) of the Pareto-optimal front will also play an important role in this respect.

5. *There exists a conflict between lower and upper level problem solving tasks.* For two solutions $\mathbf{x}$ and $\mathbf{y}$ of which $\mathbf{x}$ is Pareto-optimal and $\mathbf{y}$ is a dominated solution in

the lower level, solution $\mathbf{y}$ can be better than solution $\mathbf{x}$ in the upper level. Due to these discrepancies, these problems will cause a conflict in converging to the appropriate Pareto-optimal front in both lower and upper level optimization tasks.

6. *Extension to higher level optimization problems is possible.* Although our focus here is for bilevel problems only, test problems scalable to three or higher levels would be interesting, as there may exist some practical problems formulated in three or higher levels. On the other hand, it will also be ideal to have bilevel test problems which will degenerate to challenging single level test problems, if a single objective function is chosen for each level.

7. *Different lower level problems may contribute differently to the upper level front in terms of their extent of representative solutions on the upper level Pareto-optimal front.* These test problems will test an algorithm's ability to emphasis different lower level problems differently in order to find a well-distributed set of Pareto-optimal solutions at the upper level.

8. *Test problems must include constraints at both levels.* This will allow algorithms to be tested for their ability to handle constraints in both lower and upper level optimization problems.

Different principles are possible to construct test problems following the above guidelines. Here, we present a generalized version of a recently proposed procedure (Deb and Sinha, 2009a).

4.5.1 A Multi-Objective Bilevel Test Problem Construction Procedure

We suggest a test problem construction procedure for a bilevel problem having M and m objectives in the upper and lower level, respectively. The procedure needs at most three functional forms and is described below:

Step 1: First, a parametric trade-off function $\Phi_U : \mathbb{R}^{M-1} \rightarrow \mathbb{R}^M$ which determines a trade-off frontier $(v_1(\mathbf{u}), \ldots, v_M(\mathbf{u}))$ on the $\mathbf{F}$-space as a function of $(M-1)$ parameters $\mathbf{u}$ (can be considered as a subset of $\mathbf{x}_u$) is chosen. Figure 3 shows such a v_1-v_2 relationship on a two-objective bilevel problem.

Step 2: Next, for every point $\mathbf{v}$ on the Φ_U-frontier, a $(M-1)$-dimensional envelope $(U_1(\mathbf{t}), \ldots, U_M(\mathbf{t}))^{\mathbf{v}})$ on the $\mathbf{F}$-space as a function of $\mathbf{t}$ (having $(M-1)$ parameters) is chosen. The non-dominated part of the agglomerate envelope $\cup_{\mathbf{v}} \cup_{\mathbf{t}} \left[(v_1(\mathbf{u}) + U_1(\mathbf{t})^{\mathbf{v}}), \ldots, (v_M(\mathbf{u}) + U_M(\mathbf{t})^{\mathbf{v}})\right]$ constitutes the overall upper level Pareto-optimal front. Figure 3 indicates this upper level Pareto-optimal front and some specific Pareto-optimal points (marked with bigger circles) derived from specific $\mathbf{v}$-vectors.

Step 3: Next, for every point $\mathbf{v}$ on the Φ_U-frontier, a mapping function $\Phi_L : \mathbb{R}^{M-1} \rightarrow \mathbb{R}^{m-1}$ which maps every $\mathbf{v}$-point from the $\mathbf{U}$-frontier to the lower level Pareto-optimal front $(f_1^*(\mathbf{s}), \ldots, f_m^*(\mathbf{s}))^{\mathbf{v}}$ is chosen. Here, s is a $(m-1)$-dimensional vector and can be considered as a subset of $\mathbf{x}_l$. Figure 3 shows this mapping The envelope A′C′B′ (a circle in the figure) is mapped to the lower level Pareto-optimal frontier ACB (inlet figure on top).

Step 4: After these three functions are defined, the lower level problem can be constructed by using a bottom-up procedure adopted in Deb *et al.* (2005) through additional terms arising from other lower level decision variables: $f_j(\mathbf{x}_l) = f_j^*(\mathbf{s}) +$

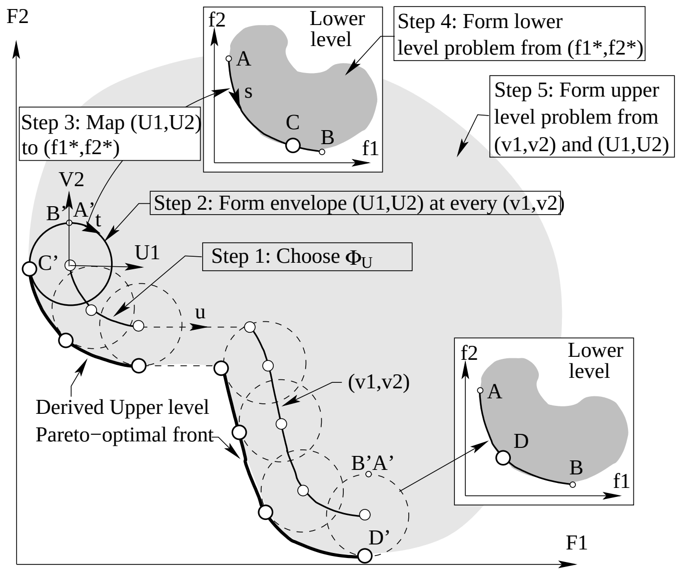
F2
f2
Lower level
A
s
C
B
f1
Step 4: Form lower level problem from (f1*,f2*)
Step 5: Form upper level problem from (v1,v2) and (U1,U2)
Step 3: Map (U1,U2) to (f1*,f2*)
V2
B' A'
t
Step 2: Form envelope (U1,U2) at every (v1,v2)
C'
U1
Step 1: Choose Φ_U
u
f2
Lower level
A
D
B
f1
(v1,v2)
Derived Upper level Pareto−optimal front
B'A'
D'
F1

Figure 3: A multi-objective bilevel test problem construction procedure is illustrated through two objectives in both upper and lower levels.

$e_j(\mathbf{x}_l \backslash \mathbf{s})$ with $e_j \geq 0$. The task of the lower level optimization task would be to make the e_j term zero for each objective. The term e_j can be made complex (multi-modal, non-linear, or large-dimensional) to make the convergence to the lower level Pareto-optimal front difficult by an optimization algorithm.

Step 5: Finally, the upper level objectives can be formed from u_j functions by including additional terms from other upper level decision variables. An additive form is as follows: $F_j(\mathbf{x}) = u_j(\mathbf{u}) + E_j(\mathbf{x}_u \backslash \mathbf{u})$ with $E_j \geq 0$. Like the e_j term, the term E_j can also be made complex for an algorithm to properly converge to the upper level Pareto-optimal front.

Step 6: Additionally, a number of linked terms $l_j(\mathbf{x}_u \backslash \mathbf{u}, \mathbf{x}_l \backslash \mathbf{s})$ and $L_j(\mathbf{x}_u \backslash \mathbf{u}, \mathbf{x}_l \backslash \mathbf{s})$ (non-negative terms) involving remaining $\mathbf{x}_u$ (without $\mathbf{u}$) and $\mathbf{x}_l$ (without $\mathbf{s}$) variables can be added to both lower and upper level problems, respectively, to make sure a proper coordination between lower and upper level optimization tasks is needed to converge to the respective Pareto-optimal fronts.

Another interesting yet a difficult scenario can be created with the linked terms. An identical link term can be added to the lower level problem, but subtracted from the the upper level problem. Thus, an effort to reduce the value of the linked term will make an improvement in the lower level, whereas it will cause a deterioration in the upper level. This will create a conflict in the working of both levels of optimization. The following two-objective test problems are constructed using the above procedure.

4.5.2 Problem DS1

This problem has $2K$ variables with K real-valued variables each for lower and upper levels. Since in this study we consider bilevel optimization problems having $m = M = 2$, the vectors $\mathbf{u}$, $\mathbf{t}$ and $\mathbf{s}$ all have a single element. Following three parametric functions are used for DS1:

Step 1: Here, one of the upper level variables y_1 is chosen as $\mathbf{u}$. The mapping Φ_U is chosen as follows: $v_1 = (1 + r) - \cos(\pi y_1)$ and $v_2 = (1 + r) - \sin(\pi y_1)$ (r is a user-supplied constant). Depending on the value of y_1, the v_1-v_2 point lies on the quarter of a circle of radius one and center at $(1+r, 1+r)$ in the $\mathbf{F}$-space, as shown in Figure 4.

Step 2: For every (v_1, v_2) point on the $\mathbf{F}$-space, the following one-dimensional (t) envelope is chosen: $v_1(y_1) = -r\cos\left(\gamma\frac{\pi}{2}\frac{t}{y_1}\right)$ and $v_2(y_1) = -r\sin\left(\gamma\frac{\pi}{2}\frac{t}{y_1}\right)$, where $t \in [0, y_1]$ and γ (=1) is a constant. The envelope is a quarter of a circle of radius r and center located at (v_1, v_2) point. Although for each (v_1, v_2) point, the entire envelope (quarter circle) is a non-dominated front, when all (v_1, v_2) points are considered, only one point from each envelope qualifies to be on the upper level Pareto-optimal front. Thus, the Pareto-optimal front for the upper level problem lies on the quarter of a circle having radius $(1 + r)$ and center at $(1 + r, 1 + r)$ on the $\mathbf{F}$-space, as indicated in the figure.

Step 3: Each U_1-U_2 point is then mapped to a (f_1^*, f_2^*) point by the following mapping: $f_1^* = s^2$, $f_2^* = (s - y_1)^2$, where $s = t$ is assumed.

Step 4: We do not use any function l_j here.

Step 5: But, we use $E_j = (y_j - \frac{j-1}{2})^2$ for $j = 2, \ldots, K$ in the upper level problem. This will ensure that $y_j = (j-1)/2$ (for $j = 2, \ldots, K$) will correspond to the upper level Pareto-optimal front.

Step 6: Different l_j and L_j terms are used with $(K-1)$ remaining upper and lower level variables such that all lower level Pareto-optimal solutions must satisfy $x_i = y_i$ for $i = 2, \ldots, K$.

The complete DS1 problem is given below:

$$
\begin{array}{l}
\text{Minimize} \quad \mathbf{F}(\mathbf{y}, \mathbf{x}) = \\
\left(\begin{array}{c}
(1 + r - \cos(\alpha\pi y_1)) + \sum_{j=2}^{K}(y_j - \frac{j-1}{2})^2 \\
+\tau\sum_{i=2}^{K}(x_i - y_i)^2 - r\cos\left(\gamma\frac{\pi}{2}\frac{x_1}{y_1}\right) \\
(1 + r - \sin(\alpha\pi y_1)) + \sum_{j=2}^{K}(y_j - \frac{j-1}{2})^2 \\
+\tau\sum_{i=2}^{K}(x_i - y_i)^2 - r\sin\left(\gamma\frac{\pi}{2}\frac{x_1}{y_1}\right)
\end{array}\right), \\
-K \le x_i \le K, \quad \text{for } i = 1, \ldots, K, \\
1 \le y_1 \le 4, \quad -K \le y_j \le K, \quad j = 2, \ldots, K.
\end{array}
\begin{array}{l}
\text{subject to} \quad (\mathbf{x}) \in \text{argmin}_{(\mathbf{x})}\mathbf{f}(\mathbf{x}) = \\
\left\{\left(\begin{array}{l}
x_1^2 + \sum_{i=2}^{K}(x_i - y_i)^2 \\
\quad + \sum_{i=2}^{K} 10(1 - \cos(\frac{\pi}{K}(x_i - y_i))) \\
\sum_{i=1}^{K}(x_i - y_i)^2 \\
\quad + \sum_{i=2}^{K} 10|\sin(\frac{\pi}{K}(x_i - y_i)|
\end{array}\right)\right\},
\end{array}
\tag{7}
$$

For this test problem, we suggest $K = 10$ (overall 20 variables), $r = 0.1$, $\alpha = 1$, $\gamma = 1$, and $\tau = 1$. Since $\tau = 1$ is used, for every lower level Pareto-optimal point, $x_i = y_i$ for $i = 2, \ldots, K$ and both l_j and L_j terms are zero, thereby making an agreement between this relationship between optimal values of x_i and y_i variables in both levels.

An interesting scenario happens when $\tau = -1$ is set. If any $\mathbf{x}$ is not Pareto-optimal to a lower level problem (meaning $x_i \neq y_i$ for $i = 2, \ldots, K$), a positive quantity is

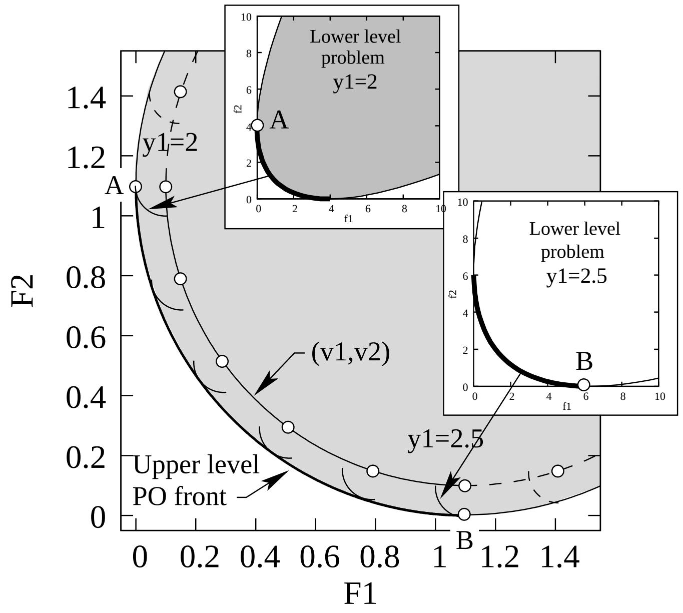

Figure 4: Pareto-optimal front for problem DS1.

deducted from the upper level objectives from each L_j term, thereby indicating that this point will dominate some true Pareto-optimal points of the upper level problem. In fact, such points are infeasible to the upper level problem due to their non-optimality property at the lower level problem and if allowed to exist in the upper level, they will dominate the true upper level Pareto-optimal front. Thus, this problem with $\tau = -1$ will be difficult to solve, in general, compared to problems with $\tau = 1$.

The problem DS1 is likely to provide following difficulties to a bilevel optimization algorithm:

- Lower level problem has multi-modalities, thereby making the lower level problem difficult to solve to Pareto-optimality.

- The problem is scalable to any even number of variables (by adjusting K).

- By choosing $\tau = -1$ in the linked term, a conflict in the working of lower and upper level problems can be introduced.

- By adjusting α, a small fraction of y_1 values can be made responsible for the upper level Pareto-optimal front, thereby making an algorithm difficult to locate the true Pareto-optimal points.

- By adjusting γ, only a part of the U_1-U_2 envelope (and thereby only a part of the f_1^*-f_2^* front) can be made responsible for the upper level Pareto-optimal front.

4.6 Problem DS2

The next problem uses Φ_U parametric function which causes a few discrete values of y_1 to determine the upper level Pareto-optimal front. The Φ_U mapping function is chosen as follows and is shown in Figure 5.

$$
\begin{aligned}
v_1(y_1) &= \begin{cases} \cos(0.2\pi)y_1 + \sin(0.2\pi)\sqrt{|0.02\sin(5\pi y_1)|}, \\ \quad \text{for } 0 \le y_1 \le 1, \\ y_1 - (1-\cos(0.2\pi)), \quad y_1 > 1 \end{cases} \\
v_2(y_1) &= \begin{cases} -\sin(0.2\pi)y_1 + \cos(0.2\pi)\sqrt{|0.02\sin(5\pi y_1)|}, \\ \quad \text{for } 0 \le y_1 \le 1, \\ 0.1(y_1-1) - \sin(0.2\pi), \quad \text{for } y_1 > 1. \end{cases}
\end{aligned} \tag{8}
$$

The U_1-U_2 parametric function is identical to that used in DS1, but here we use $\gamma = 4$. This will cause the U_1-U_2 envelope to be a complete circle, as shown by dashed lines in the figure. The f_1^*-f_2^* mapping is chosen identical to that in DS1. Again, the term e_j is not considered here and a multimodal E_j term is used. Different linked terms l_j and L_j compared to those used in DS1 are used here. The overall problem is given as follows:

$$
\begin{aligned}
&\text{Minimize} \quad \mathbf{F}(\mathbf{x},\mathbf{y}) = \\
&\begin{pmatrix} v_1(y_1) + \sum_{j=2}^{K}\left[y_j^2 + 10(1-\cos(\frac{\pi}{K}y_i))\right] \\ +\tau\sum_{i=2}^{K}(x_i-y_i)^2 - r\cos\left(\gamma\frac{\pi}{2}\frac{x_1}{y_1}\right) \\ v_2(y_1) + \sum_{j=2}^{K}\left[y_j^2 + 10(1-\cos(\frac{\pi}{K}y_i))\right] \\ +\tau\sum_{i=2}^{K}(x_i-y_i)^2 - r\sin\left(\gamma\frac{\pi}{2}\frac{x_1}{y_1}\right) \end{pmatrix}, \quad \begin{array}{l} \text{subject to} \quad (\mathbf{x}) \in \mathbf{f}(\mathbf{x}) = \\ \quad \operatorname{argmin}_{(\mathbf{x})} \left\{ \begin{pmatrix} x_1^2 + \sum_{i=2}^{K}(x_i-y_i)^2 \\ \sum_{i=1}^{K} i(x_i-y_i)^2 \end{pmatrix} \right\}, \end{array} \\
&-K \le x_i \le K, \quad i = 1,\ldots,K, \\
&0.001 \le y_1 \le K, \quad -K \le y_j \le K, \ j = 2,\ldots,K,
\end{aligned} \tag{9}
$$

Due to the use of periodic terms in v_1 and v_2 functions, the upper level Pareto-optimal front corresponds to only six discrete values of y_1 (=0.001, 0.2, 0.4, 0.6, 0.8 and 1), despite y_1 taking any real value within $[0.001, K]$. We suggest using $r = 0.25$ here. This problem

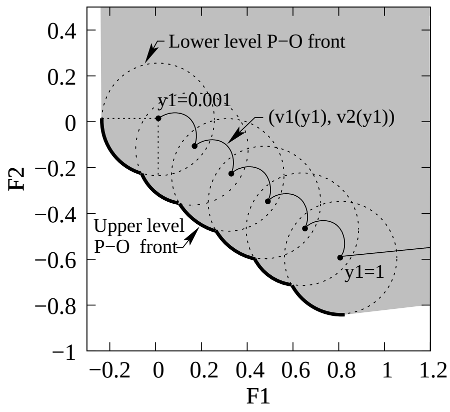

Figure 5: Pareto-optimal front for problem DS2.

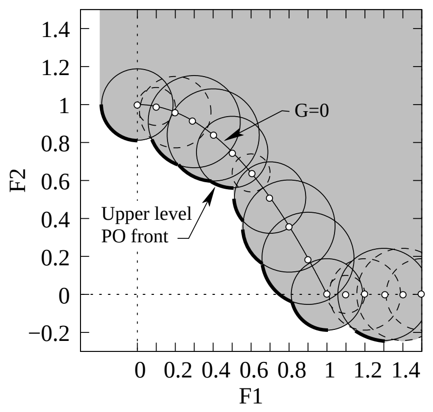

Figure 6: Pareto-optimal front for problem DS3.

has following specific properties:

- The upper level problem has multi-modalities, thereby causing an algorithm difficulty in finding the upper level Pareto-optimal front.
- With $\tau = -1$, the conflict between upper and lower level problems can be introduced, as in DS1.
- The dimension of both upper and lower level problems can be increased by increasing K.
- The parameter γ can be adjusted to cause a small proportion of lower level Pareto-optimal points to be responsible for the upper level Pareto-optimal front.

4.7 Problem DS3

In this problem, the $\Phi_U 2$ and the f_1^*-f_2^*-frontiers lie on constraints and thus are not defined parametrically as in DS1 and DS2 here, but are defined directly as a function of problem variables. Since a constraint function determines the lower level Pareto-optimal front, Φ_U and Φ_L-frontiers are defined with M variables. The linked terms ($l_j = (x_i - y_i)^2$ for $i = 3, \ldots, K$) are included. We use $L_j = \tau l_j$. Like before, we do not use any e_j term, but use a E_j term in the upper level problem. The variable y_1 is considered to be discrete, thereby causing only a few y_1 values to represent the upper level Pareto-optimal front. The overall problem is given below:

$$
\begin{array}{l}
\text{Minimize} \quad \mathbf{F}(\mathbf{x},\mathbf{y}) = \\
\left(\begin{array}{l} y_1 + \sum_{j=3}^{K}(y_j - j/2)^2 + \tau \sum_{i=3}^{K}(x_i - y_i)^2 - R(y_1)\cos(4\tan^{-1}\left(\frac{y_2 - x_2}{y_1 - x_1}\right) \\ y_2 + \sum_{j=3}^{K}(y_j - j/2)^2 + \tau \sum_{i=3}^{K}(x_i - y_i)^2 - R(y_1)\sin(4\tan^{-1}\left(\frac{y_2 - x_2}{y_1 - x_1}\right) \end{array} \right), \\
\text{subject to} \quad (\mathbf{x}) \in \text{argmin}_{(\mathbf{x})} \\
\quad \left\{ \mathbf{f}(\mathbf{x}) = \left(\begin{array}{l} x_1 + \sum_{i=3}^{K}(x_i - y_i)^2 \\ x_2 + \sum_{i=3}^{K}(x_i - y_i)^2 \end{array} \right) \middle| g_1(\mathbf{x}) = (x_1 - y_1)^2 + (x_2 - y_2)^2 \le r^2 \right\}, \\
G(\mathbf{y}) = y_2 - (1 - y_1^2) \ge 0, \\
-K \le x_i \le K, \quad \text{for } i = 1, \ldots, K, \quad 0 \le y_j \le K, \quad \text{for } j = 1, \ldots, K, \\
y_1 \text{ is a multiple of } 0.1.
\end{array}
\tag{10}
$$

Here we suggest a periodically changing radius: $R(y_1) = 0.1 + 0.15|\sin(2\pi(y_1 - 0.1)|$ and use $r = 0.2$. For the upper level Pareto-optimal points, $y_i = j/2$ for $j \le 3$. The variables y_1 and y_2 take values satisfying constraint $G(\mathbf{y}) = 0$. For each such combination, variables x_1 and x_2 lie on the third quadrant of a circle of radius r and center at (y_1, y_2) in the $\mathbf{F}$-space. Notice in Figure 6, how lower level Pareto-optimal solutions for $y_1 = 0.1$ and 0.2 (shown in dashed lines in the figure) mapped to corresponding circles in the upper level problem get dominated by that for $y_1 = 0$ and 0.3. Following properties are observed for this problem:

- The Pareto-optimal fronts for both lower and upper level lie on constraint boundaries, thereby requiring good constraint handling strategies to solve both problems optimally.
- Not all lower level Pareto-optimal solutions qualify as upper level Pareto-optimal solutions.
- Every lower level front has an unequal contribution to the upper level Pareto-optimal front.
- By choosing $\tau = -1$, conflict between two levels of optimization can be introduced.

4.8 Problem DS4

In this problem, the v_1-v_2 relationship is linear ($v_1 = 2 - y_1$, $v_2 = 2(y_1 - 1)$), spanning in the first quadrant of $\mathbf{F}$-space. The mapping U_1-U_2 is not considered here. For every (v_1, v_2) point, the following relationship is chosen for the lower level Pareto-optimal front: $f_1^* + f_2^* = y_1$. Additional terms having a minimum value of one are multiplied to form the lower and upper level search spaces. This problem has $K + L + 1$ variables, which are all real-valued:

$$\begin{array}{ll} \text{Minimize} \quad \mathbf{F}(\mathbf{x}, \mathbf{y}) = & \text{subject to} \quad (\mathbf{x}) \in \text{argmin}_{(\mathbf{x})} \mathbf{f}(\mathbf{x}) = \\ \left(\begin{array}{l} (1 - x_1)(1 + \sum_{j=2}^{K} x_j^2) y_1 \\ x_1 (1 + \sum_{j=2}^{K} x_j^2) y_1 \end{array} \right), & \left\{ \left(\begin{array}{l} (1 - x_1)(1 + \sum_{j=K+1}^{K+L} x_j^2) y_1 \\ x_1 (1 + \sum_{j=K+1}^{K+L} x_j^2) y_1 \end{array} \right) \right\}, \\ G_1(\mathbf{x}) = (1 - x_1) y_1 + \frac{1}{2} x_1 y_1 - 1 \geq 0, & \\ -1 \leq x_1 \leq 1, \quad 1 \leq y_1 \leq 2, & \\ -(K + L) \leq x_i \leq (K + L), \; i = 2, \ldots, (K + L). & \end{array} \tag{11}$$

The upper level Pareto-optimal front is formed with $x_i = 0$ for all $i = 2, \ldots, (K + L)$ and $x_1 = 2(1 - 1/y_1)$ and $y_1 \in [1, 2]$. This problem has following properties:

- By increasing K and L, the problem complexity in converging to the appropriate lower and upper level fronts can be increased.
- Only one Pareto-optimal point from each participating lower level problem qualifies to be on the upper level front.

For our study here, we choose $K = 5$ and $L = 4$ (an overall 10-variable problem).

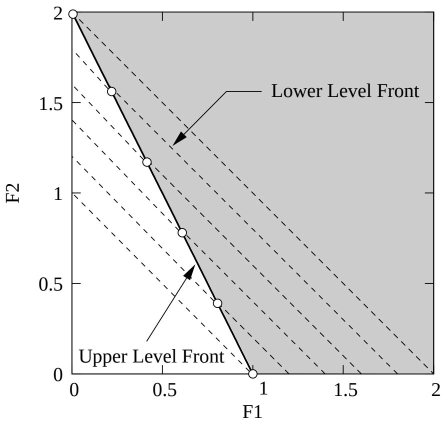

Figure 7: Pareto-optimal front for problem DS4.

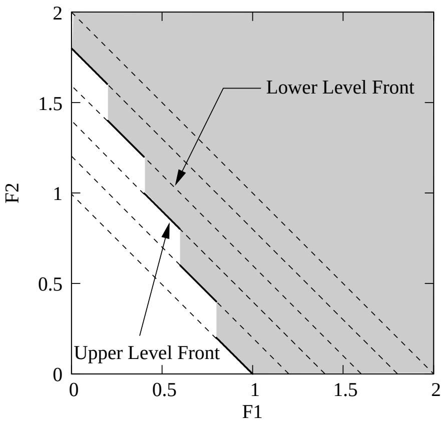

Figure 8: Pareto-optimal front for problem DS5.

4.9 Problem DS5

This problem is similar to problem DS4 except that the upper level Pareto-optimal front is constructed from multiple points from a few lower level Pareto-optimal fronts. There are $K + L + 1$ real-valued variables in this problem as well:

$$\begin{array}{ll}
\text{Minimize}\quad \mathbf{F}(\mathbf{x},\mathbf{y}) = & \text{subject to}\quad (\mathbf{x}) \in \text{argmin}_{(\mathbf{x})}\mathbf{f}(\mathbf{x}) = \\
\left(\begin{array}{l} (1-x_1)(1+\sum_{j=2}^{K} x_j^2)y_1 \\ x_1(1+\sum_{j=2}^{K} x_j^2)y_1 \end{array}\right), & \left\{\left(\begin{array}{l} (1-x_1)(1+\sum_{j=K+1}^{K+L} x_j^2)y_1 \\ x_1(1+\sum_{j=K+1}^{K+L} x_j^2)y_1 \end{array}\right)\right\}, \\
\multicolumn{2}{l}{G_1(\mathbf{x}) = (1-x_1)y_1 + \frac{1}{2}x_1y_1 - 2 + \frac{1}{5}\left[5(1-x_1)y_1 + 0.2\right] \ge 0,\ [\cdot] \text{ denotes greatest int. function,}} \\
\multicolumn{2}{l}{-1 \le x_1 \le 1, \quad 1 \le y_1 \le 2,} \\
\multicolumn{2}{l}{-(K+L) \le x_i \le (K+L),\ i = 2,\ldots,(K+L).}
\end{array} \tag{12}$$

For the upper level Pareto-optimal front, $x_i = 0$ for $i = 2,\ldots,(K+L)$, $x_1 \in [2(1 - 1/y_1), 2(1 - 0.9/y_1)]$, $y_1 \in \{1, 1.2, 1.4, 1.6, 1.8\}$ (Figure 8). For this test problem we have chosen $K = 5$ and $L = 4$ (an overall 10-variable problem). This problem has similar difficulties as in DS4, except that only a finite number of y_1 qualifies at the upper level Pareto-optimal front and that a consecutive set of lower level Pareto-optimal solutions now qualify to be on the upper level Pareto-optimal front.

5 Hybrid Bilevel Evolutionary Multi-Objective Optimization (H-BLEMO) Algorithm

The proposed hybrid BLEMO procedure is motivated from our previously suggested algorithms (Deb and Sinha, 2009a,b), but differs in many different fundamental ways. Before we describe the differences, we first outline the proposed hybrid procedure.

A sketch of the population structure is shown in Figure 9. The initial population

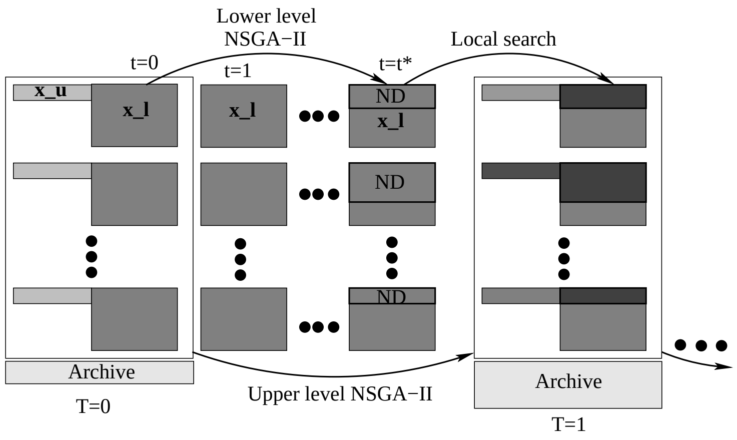

Figure 9: A sketch of the proposed bilevel optimization algorithm.

(marked with upper level generation counter $T = 0$ of size N_u) has a subpopulation of lower level variable set $\mathbf{x}_l$ for each upper level variable set $\mathbf{x}_u$. Initially the subpopulation size ($N_l^{(0)}$) is kept identical for each $\mathbf{x}_u$ variable set, but it is allowed to change adaptively with generation T. Initially, an empty archive A_0 is created. For each $\mathbf{x}_u$, we perform a lower level NSGA-II operation on the corresponding subpopulation having variables $\mathbf{x}_l$ alone, not till the true lower level Pareto-optimal front is found, but only till a small number of generations at which the specified lower level termination criterion (discussed in subsection 5.2) is satisfied. Thereafter, a local search is performed on a few rank-one lower level solutions until the local search termination criterion is met (discussed in Step 3 in subsection 5.3). The archive is maintained at the upper level containing solution vectors $(\mathbf{x}_{u_a}, \mathbf{x}_{l_a})$, which are optimal at the lower level and non-dominated at the upper level. The solutions in the archive are updated after every lower level NSGA-II call. The members of the lower level population undergoing a local search are lower level optimal solutions and hence are assigned an 'optimality tag'. These local searched solutions ($\mathbf{x}_l$) are then combined with corresponding $\mathbf{x}_u$ variables and become eligible to enter the archive if it is non-dominated when compared to the existing members of the archive. The dominated members in the archive are then eliminated. The solutions obtained from the lower level ($\mathbf{x}_l$) are combined with corresponding $\mathbf{x}_u$ variables and are processed by the upper level NSGA-II operators to create a new upper level population. This process is continued till an upper level termination criterion (described in subsection 5.2) is satisfied.

To make the proposed algorithm computationally faster, we have used two different strategies: (i) for every upper level variable vector $\mathbf{x}_u$, we do not completely solve the lower level multi-objective optimization problem, thereby not making our approach a nested procedure, and (ii) the subpopulation size and number of generations for a lower level NSGA-II simulation are computed adaptively based on the relative location of $\mathbf{x}_u$ compared to archive solutions, thereby making the overall procedure more less parametric and more computationally efficient in terms of overall function evaluations. However, before we present a detailed step-by-step procedure, we discuss the automatic update procedure of population size and termination criteria of the lower level NSGA-II.

5.1 Update of Population Sizes

The upper level population size N_u is kept fixed and is chosen to be proportional to the number of variables. However, the subpopulation size (N_l) for each lower level NSGA-II is sized in a self-adaptive manner. Here we describe the procedure.

In a lower level problem, $\mathbf{x}_l$ is updated by a modified NSGA-II procedure and the corresponding $\mathbf{x}_u$ is kept fixed throughout. Initially, The population size of each lower level NSGA-II ($N_l^{(0)}$) is set depending upon the dimension of lower and upper level variables ($|\mathbf{x}_l|$ and $|\mathbf{x}_u|$, respectively). The number of lower level subpopulations ($n_s^{(0)}$) signifies the number of independent population members for $\mathbf{x}_u$ in a population. Our intention is to set the population sizes ($n_s^{(0)}$ and $N_l^{(0)}$) for $\mathbf{x}_u$ and $\mathbf{x}_l$ proportionately to their dimensions, yielding

$$\frac{n_s^{(0)}}{N_l^{(0)}} = \frac{|\mathbf{x}_u|}{|\mathbf{x}_l|}. \tag{13}$$

Noting also that $n_s^{(0)} N_l^{(0)} = N_u$, we obtain the following sizing equations:

$$n_s^{(0)} = \sqrt{\frac{|\mathbf{x}_u|}{|\mathbf{x}_l|} N_u}, \tag{14}$$

$$N_l^{(0)} = \sqrt{\frac{|\mathbf{x}_l|}{|\mathbf{x}_u|} N_u}. \tag{15}$$

For an equal number of lower and upper level variables, $n_s^{(0)} = N_l^{(0)} = \sqrt{N_u}$. The above values are set for the initial population only, but are allowed to get modified thereafter in a self-adaptive manner by directly relating the location of the corresponding $\mathbf{x}_u$ variable vector from the points in the archive in the variable space. As shown in Figure 10, first the maximum Euclidean distance (δ_U) in the $\mathbf{x}_u$-space among the members of the archive is computed. Then, the Euclidean distance (δ_u) between the current

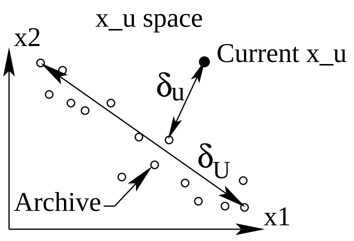

Figure 10: Computation of δ_u and δ_U.

$\mathbf{x}_u$ vector and the closest archive member is computed. The subpopulation size N_l is then set proportional to the ratio of δ_u and δ_U as follows:

$$N_l = (\text{round})\frac{\delta_u}{\delta_U} N_l^{(0)}. \tag{16}$$

To eliminate the cases with too low or too large population sizes, N_l is restricted between four (due to the need of two binary tournament selection operations to choose two parent solutions for a single recombination event in the NSGA-II) and $N_l^{(0)}$. If the current $\mathbf{x}_u$ variable vector is far away from the archive members, a large number of generations must have to be spent in the corresponding lower level NSGA-II, as dictated by equation (16).

5.2 Termination Criteria

In a bilevel optimization, it is clear that the lower level optimization must have to be run more often than the upper level optimization, as the former task acts as a constraint to the upper level task. Thus, any judicial and efficient efforts in terminating a lower level optimization can make a substantial saving in the overall computational effort. For this purpose, we first gauge the difficulty of solving all lower level problems by observing the change in their *hypervolume* measures only in the initial generation ($T = 0$) of the upper level optimization.

The maximum ($H^{\max}$) and minimum ($H^{\min}$) hypervolume is calculated from the lower level non-dominated set (with a reference point constructed from the worst objective values of the set) in every τ generations of a lower level run. The H_l-metric is then computed as follows:

$$H_l = \frac{H_l^{\max} - H_l^{\min}}{H_l^{\max} + H_l^{\min}}. \tag{17}$$

If $H_l \leq \epsilon_l$ (a threshold parameter) is encountered, indicating that an adequate convergence in the hypervolume measure is obtained, the lower level NSGA-II simulation is terminated. The number of lower level generations needed to meet the above criterion is calculated for each subpopulation during the initial generation ($T = 0$) of the upper level NSGA-II and an average (denoted here as $t_l^{\max}$) is computed. Thereafter, no subsequent lower level NSGA-II simulations are allowed to proceed beyond t_l generations (derived from $t_l^{\max}$, as calculated below) or the above $H_l \leq \epsilon_l$ is satisfied. We bound the limiting generation (t_l) to be proportional to the distance of current $\mathbf{x}_u$ from the archive, as follows:

$$t_l = (\mathrm{int})\ \frac{\delta_u}{\delta_U} t_l^{\max}. \tag{18}$$

For terminating the upper level NSGA-II, the normalized change in hypervolume measure H_u of the upper level population (as in equation (17) except that the hypervolume measure is computed in the upper level objective space) is computed in every τ consecutive generations. When $H_u \leq \epsilon_u$ (a threshold parameter) is obtained, the overall algorithm is terminated. We have used $\tau = 10$, $\epsilon_l = 0.1$ (for a quick termination) and $\epsilon_u = 0.0001$ (for a reliable convergence of the upper level problem) for all problems in this study.

Now, we are ready to describe the overall algorithm for a typical generation in a step-by-step format.

5.3 Step-by-Step Procedure

At the start of the upper level NSGA-II generation T, we have a population P_T of size N_u. Every population member has the following quantities computed from the previous iteration: (i) a non-dominated rank ND_u corresponding to $\mathbf{F}$ and $\mathbf{G}$, (ii) a crowding distance value CD_u corresponding to $\mathbf{F}$, (iii) a non-dominated rank ND_l corresponding to $\mathbf{f}$ and $\mathbf{g}$, and (iv) a crowding distance value CD_l using $\mathbf{f}$. In addition to these quantities, for the members stored in the archive A_T, we have also computed (v) a crowding distance value CD_a corresponding to $\mathbf{F}$ and (vi) a non-dominated rank ND_a corresponding to $\mathbf{F}$ and $\mathbf{G}$.

Step 1a: Creation of new $\mathbf{x}_u$: We apply two binary tournament selection operations on members ($\mathbf{x} = (\mathbf{x}_u, \mathbf{x}_l)$) of P_T using ND_u and CD_u lexicographically. Also, we apply two binary tournament selections on the archive population A_T using ND_a and CD_a lexicographically. Of the four selected members, two participate in the recombination operator based on stochastic events. The members from A_T participate as parents with a probability of $\frac{|A_T|}{|A_T|+|P_T|}$, otherwise the members from P_T become the parents for recombination. The upper level variable vectors $\mathbf{x}_u$ of the two selected parents are then recombined using the SBX operator (Deb and Agrawal, 1995) to obtain two new vectors of which one is chosen for further processing at random. The chosen vector is then mutated by the polynomial mutation operator (Deb, 2001) to obtain a child vector (say, $\mathbf{x}_u^{(1)}$).

Step 1b: Creation of new $\mathbf{x}_l$: First, the population size ($N_l(\mathbf{x}_u^{(1)})$) for the child solution $\mathbf{x}_u^{(1)}$ is determined by equation (16). The creation of $\mathbf{x}_l$ depends on how close the new variable set $\mathbf{x}_u^{(1)}$ is compared to the current archive, A_T. If $N_l = N_l^{(0)}$ (indicating that the $\mathbf{x}_u$ is away from the archive members), new lower level variable vectors $\mathbf{x}_l^{(i)}$ (for $i = 1, \ldots, N_l(\mathbf{x}_u^{(1)})$) are created by applying selection-recombination-mutation operations on members of P_T and A_T. Here, a parent member is chosen from A_T with a probability $\frac{|A_T|}{|A_T|+|P_T|}$, otherwise a member from P_T is chosen at random. A total of $N_l(\mathbf{x}_u^{(1)})$ child solutions are created by concatenating upper and lower level variable vectors together, as follows: $c_i = (\mathbf{x}_u^{(1)}, \mathbf{x}_l^{(i)})$ for $i = 1, \ldots, N_l(\mathbf{x}_u^{(1)})$. Thus, for the new upper level variable vector $\mathbf{x}_u^{(1)}$, a subpopulation of $N_l(\mathbf{x}_u^{(1)})$ lower level variable vectors are created by genetic operations from P_T and A_T.

However, if the lower level population size ($N_l(\mathbf{x}_u^{(1)})$) is less than $N_l^{(0)}$ (indicating that the variable set $\mathbf{x}_u$ is close to the archive members), a different strategy is used. First, a specific archive member (say, $\mathbf{x}_u^{(a)}$) closest to $\mathbf{x}_u^{(1)}$ is identified. Instead of creating new lower level variable vectors, $N_l(\mathbf{x}_u^{(1)})$ vectors are chosen from the subpopulation to which $\mathbf{x}_u^{(a)}$ belongs. Complete child solutions are created by concatenating upper and lower level variables vectors together. If however the previous subpopulation does not have $N_l(\mathbf{x}_u^{(1)})$ members, the remaining slots are filled by creating new child solutions by the procedure of the previous paragraph.

Step 2: Lower level NSGA-II: For each subpopulation, we now perform a NSGA-II procedure using lower level objectives ($\mathbf{f}$) and constraints ($\mathbf{g}$) for t_l generations (equation (18)). It is important to reiterate that in each lower level NSGA-II run, the upper level variable vector $\mathbf{x}_u$ is not changed. The selection process is different from that in the usual NSGA-II procedure. If the subpopulation has no member in the current archive A_T, the parent solutions are chosen as usual by the binary tournament selection using ND_l and CD_l lexicographically. If, however, the subpopulation has a member or members which already exist in the archive, only these solutions are used in the binary tournament selection. This is done to emphasize already-found good solutions. The mutation operator is applied as usual. After the lower level NSGA-II simulation is performed on a subpopulation, the members are sorted according to the constrained non-domination level (Deb *et al.*, 2002) and are assigned their non-dominated rank (ND_l) and crowding distance value (CD_l) based on lower level objectives ($\mathbf{f}$) and lower level constraints ($\mathbf{g}$).

Step 3: Local search: The local search operator is employed next to provide us with a solution which is guaranteed to be on a locally Pareto-optimal front. Since the local search operator can be expensive, we use this operator sparingly. We apply the local search operator to good solutions having the following properties: (i) it is a non-dominated solution in the lower level having $ND_l = 1$, (ii) it is a non-dominated solution in the upper level having $ND_u = 1$, and (iii) it does not get dominated by any current archive member, or it is located at a distance less than $\delta_U N_l / N_l^{(0)}$ from any of the current archive members. In the local search procedure, the achievement scalarizing function problem (Wierzbicki, 1980) formulated at the current NSGA-II solution ($\mathbf{x}_l$) with $z_j = f_j(\mathbf{x}_l)$ is solved:

$$\begin{array}{rl} \text{Minimize}_{\mathbf{p}} & \max_{j=1}^{m} \frac{f_j(\mathbf{p})-z_j}{f_j^{\max}-f_j^{\min}} + \rho \sum_{j=1}^{m} \frac{f_j(\mathbf{p})-z_j}{f_j^{\max}-f_j^{\min}}, \\ \text{subject to} & \mathbf{p} \in \mathcal{S}_l, \end{array} \tag{19}$$

where $\mathcal{S}_l$ is the feasible search space for the lower level problem. The minimum and maximum function values are taken from the NSGA-II minimum and maximum function values of the current generation. The optimal solution $\mathbf{p}^*$ to the above problem is guaranteed to be a Pareto-optimal solution to the lower level problem (Miettinen, 1999). Here, we use $\rho = 10^{-6}$, which prohibits the local search to converge to a weak Pareto-optimal solution. We use a popular software KNITRO (Byrd *et al.*, 2006) (which employs a sequential quadratic programming (SQP) algorithm) to solve the above single objective optimization problem. The KNITRO software terminates when a solution satisfies the Karush-Kuhn-Tucker (KKT) conditions (Reklaitis *et al.*, 1983) with a pre-specified error limit. We fix this error limit to 10^{-2} in all problems of this study here. The solutions which meet this KKT satisfaction criterion are assigned an 'optimal tag' for further processing. For handling non-differentiable problems, a non-gradient, adaptive step-size based hill-climbing procedure (Nolle, 2006) can be used.

Step 4: Updating the archive: The optimally tagged members, if feasible with respect to the upper level constraints ($\mathbf{G}$), are then compared with the current archive members. If these members are non-dominated when compared to the members of the archive, they become eligible to be added into the archive. The dominated members in the archive are also eliminated, thus the archive always keeps non-dominated solutions. We limit the size of archive to $10N_u$. If and when more members are to be entered in the archive, the archive size is maintained to the above limit by eliminating extra members using the crowding distance (CD_a) measure.

Step 5: Formation of the combined population: Steps 1 to 4 are repeated until the population Q_T is filled with newly created solutions. Each member of Q_T is now evaluated with $\mathbf{F}$ and $\mathbf{G}$. Populations P_T and Q_T are combined together to form R_T. The combined population R_T is then ranked according to constrained non-domination (Deb *et al.*, 2002) based on upper level objectives ($\mathbf{F}$) and upper level constraints ($\mathbf{G}$). Solutions are thus, assigned a non-dominated rank (ND_u) and members within an identical non-dominated rank are assigned a crowding distance (CD_u) computed in the $\mathbf{F}$-space.

Step 7: Choosing half the population: From the combined population R_T of size $2N_u$, half of its members are retained in this step. First, the members of rank $ND_u = 1$ are considered. From them, solutions having $ND_l = 1$ are noted one by one in the order of reducing crowding distance CD_u. For each such solution, the entire N_l subpopulation from its source population (either P_T or Q_T) are copied in an intermediate population S_T. If a subpopulation is already copied in S_T and a future solution from the same subpopulation is found to have $ND_u = ND_l = 1$, the subpopulation is not copied again. When all members of $ND_u = 1$ are considered, a similar consideration is continued with $ND_u = 2$ and so on till S_T has N_u population members.

Step 6: Upgrading old lower level subpopulations: Each subpopulation of S_T which are not created in the current generation are modified using the lower level NSGA-II procedure (Step 2) applied with $\mathbf{f}$ and $\mathbf{g}$. This step helps progress each lower level population towards their individual Pareto-optimal frontiers.

The final population is renamed as P_{T+1}. This marks the end of one generation of the overall H-BLEMO algorithm.

5.4 Algorithmic Complexity

With self-adaptive operations to update population sizes and number of generations, it becomes difficult to compute an exact number of function evaluations (FE) needed in the proposed H-BLEMO algorithm. However, using the maximum allowable values of these parameters, we estimate that the worst case function evaluations is $N_u(2T_u + 1)(t_l^{\max} + 1) + FE_{\text{LS}}$. Here T_u is the number of upper level generations and FE_{LS} is the total function evaluations used by the local search (LS) algorithm. Lower level

NSGA-II is able to bring the members close to the Pareto-optimal front which requires a relatively small number of function evaluations required by the local search operator. Moreover, towards the end of a simulation, most upper level solutions are close to the archive, thereby requiring a much smaller number of function evaluations than that used in the above expression. As an evidence to this fact, Figure 22 can be referred for a variation in N_l and t_l for two test problems of this study.

However, it is important to note that the computations needed in the local search may be substantial and any effort to reduce the computational effort will be useful. In this regard, the choice of the local search algorithm and the chosen KKT error limit for terminating the local search will play an important role. Also, the termination parameter for lower level NSGA-II run (parameter ϵ_l) may also make an important contribution. For both these parameters, we have used reasonable values (KKT error threshold of 0.01 and $\epsilon_l = 0.1$) for a good balance between accuracy and computational efficiency. In section 6.9, we shall discuss more about these issues by empirical evidence of independent computations needed in the local search, in the lower level NSGA-II, and in the upper level NSGA-II.

5.5 Review of the Drawbacks in Earlier Approaches

The proposed self-adaptive and hybrid bilevel procedure is quite different from our earlier rather rigid and computationally expensive BLEMO procedures (Deb and Sinha, 2009a,b). We carry out a review of the drawbacks in the earlier approaches and elucidate how these drawbacks have been tackled in H-BLEMO.

1. The previous approaches did not perform a local search at the lower level NSGA-II and hence did not guarantee a lower level solution to be optimal. The methods relied on the solutions provided by the EMO at the lower level. The H-BLEMO procedure incorporates a local search to the best non-dominated lower level NSGA-II solutions so that an indication of distance and direction of lower Pareto-optimal front from the current solution can be obtained. Moreover, in later generations, the use of local search in the lower level optimization guarantees convergence to the locally Pareto-optimal front, thereby satisfying the principle of bilevel programming which requires that the final archive solutions are true lower level Pareto-optimal solutions.

2. The earlier approaches were rigid in the sense that every time it made a call to the lower level NSGA-II, a fixed population size and number of generations were used for the lower level run. This was done irrespective of the solutions being close to the front. The approaches did not have a measure to estimate the proximity of the solutions from the lower level Pareto-optimal front which made it necessary to run the lower level NSGA-II with a sufficient number of generations and with an adequate population size to obtain near Pareto-optimal solutions. Moreover in the absence of local search, it is necessary to have sufficient population size and number of generations otherwise the run would provide solutions which are not close to the true Pareto-optimal front at the lower level and hence infeasible at the upper level. This led to unnecessary lower level evaluations which has been avoided in the H-BLEMO. The H-BLEMO procedure is self-adaptive in nature. This allows each lower level NSGA-II to use a different population size and run for a different number of generations adaptively depending on the proximity of the upper level variable vector to current archive solutions. This should allow the H-BLEMO procedure to allocate function evaluations as and where needed.

3. The previous approaches had a fixed population size for the lower level. This made the number of upper level variable vectors fixed from one generation to another for these approaches, whereas in H-BLEMO, the use of a variable population size for different lower level NSGA-IIs establishes that the upper level population can hold varying number of upper level variable vectors. This allows the H-BLEMO procedure to provide a varying importance between the extent of upper and lower level optimizations, as may be demanded by a problem.

4. The earlier procedures did not have a algorithmically sensible termination criteria for the upper or the lower level NSGA-II and had to run for a fixed number of generations. In H-BLEMO a hypervolume based termination criteria has been used where both lower and upper level terminate, based on the dynamic performance of the algorithms. The criteria ensures the termination of each NSGA-II whenever the corresponding non-dominated front has stabilized, thereby avoiding unnecessary function evaluations.

6 Results on Test Problems

We use the following standard NSGA-II parameter values in both lower and upper levels on all problems of this study: Crossover probability of 0.9, distribution index for SBX of 15, mutation probability of 0.1, distribution index for the polynomial mutation of 20. The upper level population size is set proportional to the total number of variables (n): $N_u = 20n$. As described in the algorithm, all other parameters including the lower level population size, termination criteria are all set in a self-adaptive manner during the optimization run. In all cases, we have used 21 different simulations starting from different initial populations and show the 0, 50, and 100% attainment surfaces (Fonseca and Fleming, 1996) to describe the robustness of the proposed procedure.

6.1 Problem TP1

This problem has three variables (one for upper level and two for lower level). Thus, we use $N_u = 60$ population members. Figure 11 shows the final archive members of a single run on the upper level objective space. It can be observed that our proposed procedure is able to find solutions on the true Pareto-optimal front. Conditions for the exact Pareto-optimal solutions are given in equation (3). For each of the obtained solutions, we use the upper level variable (y) value to compute lower level optimal variable (x_1^* and x_2^*) values using the exact conditions for Pareto-optimality and compare the values with H-BLEMO solutions. The average error $\sum_{i=1}^{2}(x_i - x_i^*)^2/2$ for each obtained solution is computed and then averaged over all archive solutions. This error value for our H-BLEMO is found to be 7.0318×10^{-5}, whereas the same error value computed for the solutions reported with our earlier algorithm (Deb and Sinha, 2009a) is found to be 2.2180×10^{-3}. The closer adherence to optimal variable values with H-BLEMO indicates a better performance of our hybrid algorithm. The minimum, median and worst number of function evaluations needed over 21 different runs of H-BLEMO are 567,848, 643,753, and 716,780, respectively. Although for a three-variable problem, these numbers may seem too large, one needs to realize that the bilevel problems have one optimization algorithm nested into another and Pareto-optimality for the lower level problem is essential for an upper level solution. In comparison, our previous algorithm (Sinha and Deb, 2009) required 1,030,316, 1,047,769 and 1,065,935 function evaluations for best, median and worst performing runs. Despite using about 40% less function evaluations, our H-BLEMO also finds more accurate solutions.

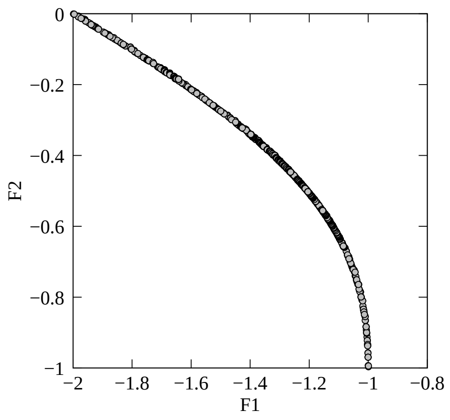

Figure 11: Final archive solutions for problem TP1.

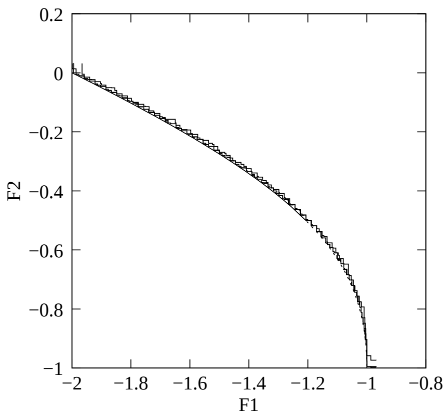

Figure 12: Attainment surfaces (0%, 50% and 100%) for problem TP1 from 21 runs.

The attainment surfaces obtained for the archive solutions over 21 runs are shown in Figure 12. All three surfaces are so close to each other that they are difficult to be distinguished from one another. This indicates the robustness of the procedure. The hypervolume values are computed after normalizing the upper level objective values by their minimum and maximum values. The hypervolumes for the 0%, 50% and 100% attainment surfaces are 0.3583, 0.3678 and 0.3700, respectively. The difference in the hypervolume value over 21 runs is only about 3%.

In order to investigate the effect of N_u on the performance of the algorithm, next, we use different N_u values but maintain an identical termination criteria. Figure 13 shows the function evaluations needed for lower (including the local search) and upper level optimization tasks for different N_u values ranging from 40 to 200. It is clear from the figure that a population size of $N_u = 60$ (which we used in Figures 11 and 12) performs the best, on an average, in both lower and upper level optimization tasks.

6.2 Problem TP2

The second test problem has $n = 15$ variables. Thus, we use $N_u = 300$. Figure 14 shows the final archive population of a typical run. The attainment surface plot in Figure 15 shows that the proposed algorithm is fairly robust in all 21 different simulations. The algorithm finds an almost an identical front close to the true Pareto-optimal front in multiple runs. The hypervolumes for the obtained attainment surfaces are 0.8561, 0.8582 and 0.8589, making a maximum difference of 0.3% only. A comparison of our current local search based algorithm with our previously proposed BLEMO procedure (Sinha and Deb, 2009) in terms of an error measure (as discussed for problem TP1) from the exact Pareto-optimal solutions indicates a smaller error for our current approach. Despite the use of 15 variables here, as opposed to 7 variables used in the previous study, the error in the current approach is 7.920×10^{-6}, compared to 11.158×10^{-6} in the previous study. In terms of the median performance, H-BLEMO requires 338,232 function evaluations for the 15-variable problem, as opposed to 771,404 function evaluations needed by our previous approach (Sinha and Deb, 2009) on a seven-variable version of the problem.

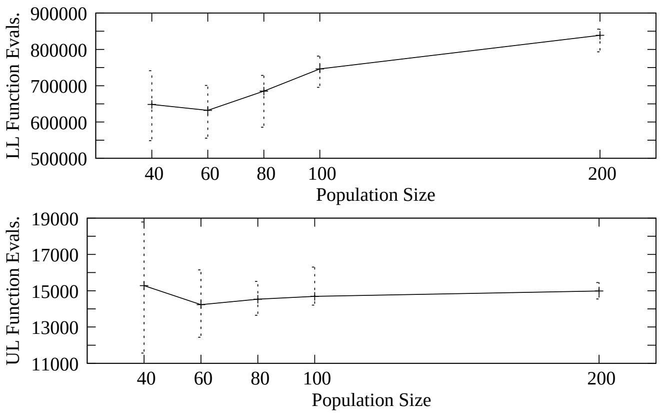

Figure 13: Average lower and upper level function evaluations with different population sizes (N_u) for problem TP1. The average has been taken for 21 runs. On an average, $N_u = 60$ is found to perform the best.

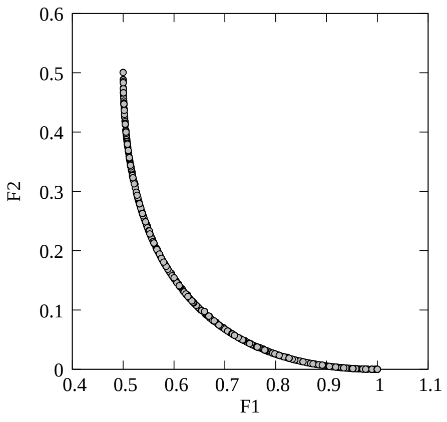

Figure 14: Final archive solutions for problem TP2.

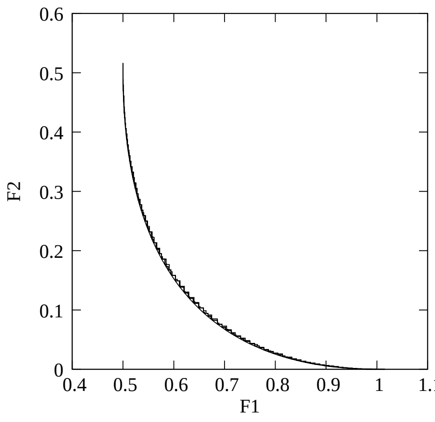

Figure 15: Attainment surfaces (0%, 50% and 100%) for problem 2 from 21 runs for problem TP2.

Interestingly the function evaluation plots (Figure 16) for the lower and upper level tasks indicate that the proposed $N_u = 20n$ (300 in this case) works the best in terms of achieving a similar performance with the smallest number of function evaluations.

For brevity and space limitations, we do not show results on TP3, but similar results are found for this problem as well. But, we show results on TP4 to highlight a comparison of H-BLEMO with a previous study.

6.3 Problem TP4

This problem is linear and has five variables in total. Thus, we use $N_u = 100$. Figure 17 shows the final archive which follows a linear relationship among upper level objec-

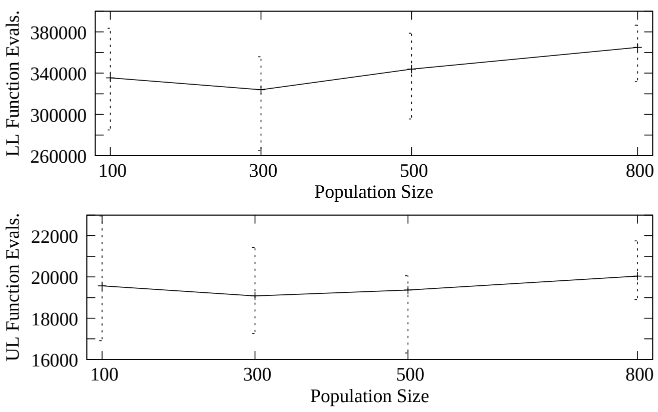

Figure 16: Average lower and upper level function evaluations with different population sizes for problem TP2 indicates $N_u = 300$ is the best choice. 21 runs are performed in each case.

tives. This multi-objective bilevel problem was solved in the original study (Zhang *et al.*, 2007) by converting two objectives into a single objective by the weighted-sum approach. The reported solution is marked on the figure with a star. It is interesting to note that this solution is one of the extreme solutions of our obtained front. For a problem having a linear Pareto-optimal front, the solution of a weighted-sum optimization approach is usually one of the extreme points. To this account, this result signifies the accuracy and efficiency of the H-BLEMO procedure.

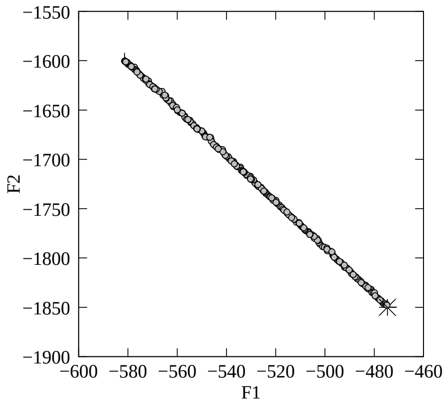

Figure 17: Final archive solutions for problem TP4. The point marked with a star is taken from Zhang *et al.* (2007).

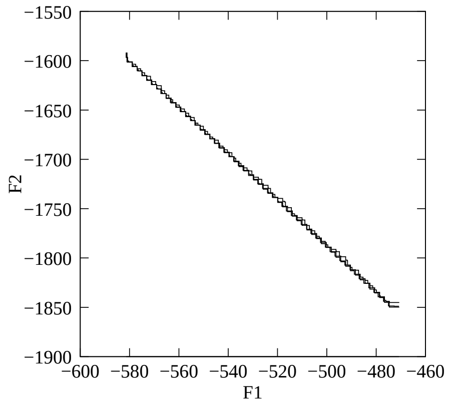

Figure 18: Attainment surfaces (0%, 50% and 100%) for problem TP4 from 21 runs.

Figure 18 shows three attainment surface plots which are close to each other, like they are in the previous two problems. The hypervolumes for the obtained attainment surfaces are 0.5239, 0.5255 and 0.5260, with a maximum difference of 0.4% only, indicating the robustness of our procedure.

6.4 Problem DS1

This problem has 10 upper and 10 lower level variables. Thus, an overall population of size $N_u = 400$ is used here. For this problem, we first consider $\tau = 1$. Figure 19 shows the obtained archive solutions for a typical run. It is worth mentioning here that this problem was possible to be solved up to only six variables (in total) by our earlier fixed BLEMO approach (Deb and Sinha, 2009a). But here with our hybrid and self-adaptive approach, we are able to solve 20-variable version of the problem. Later, we present results with 40 variables as well for this problem.

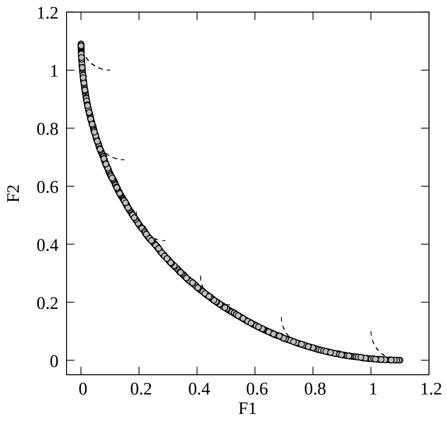

Figure 19: Final archive solutions for problem DS1.

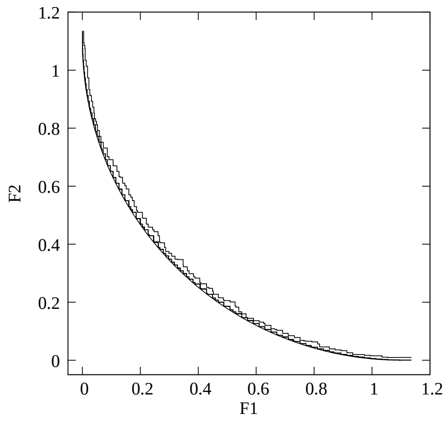

Figure 20: Attainment surfaces (0%, 50% and 100%) for problem DS1 from 21 runs.

The attainment surface plots in Figure 20 further show the robustness of the proposed algorithm. The hypervolumes for the obtained attainment surfaces are 0.7812, 0.7984 and 0.7992, with a maximum difference of about 2%. The parametric study with N_u in Figure 21 shows that $N_u = 400$ is the best choice in terms of achieving a similar performance on the hypervolume measure using the smallest number of function evaluations, thereby supporting our setting $N_u = 20n$.

Since this problem is more difficult compared to the previous problems (TP1, TP2, and TP4), we investigate the effect of self-adaptive changes in lower level NSGA-II parameters (N_l and t_l) with the generation counter. In the left side plot of Figure 22, we show the average value of these two parameters for every upper level generation (T) from a typical simulation run. It is interesting to note that, starting with an average of 20 members in each lower level subpopulation, the number reduces with generation counter, meaning that smaller population sizes are needed for later lower level simulations. The average subpopulation size reduces to its lower permissible value of four in 32 generations. Occasional increase in average N_l indicates that an upper level variable vector may have been found near a previously undiscovered region of the Pareto-optimal front. The hybrid algorithm increases its lower level population to explore the region better in such occasions.

The variation of number of lower level generations (t_l) before termination also follows a similar trend, except that at the end only 3-9 generations are required to fulfill

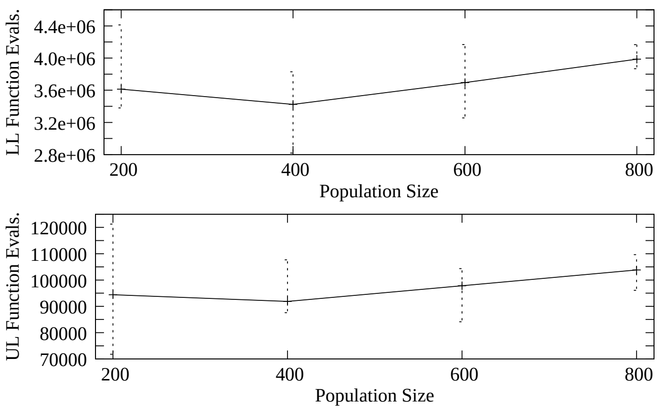

Figure 21: Average lower and upper level function evaluations with different population sizes for problem DS1 indicates an optimal population size of $N_u = 400$. The average has been taken for 21 runs.

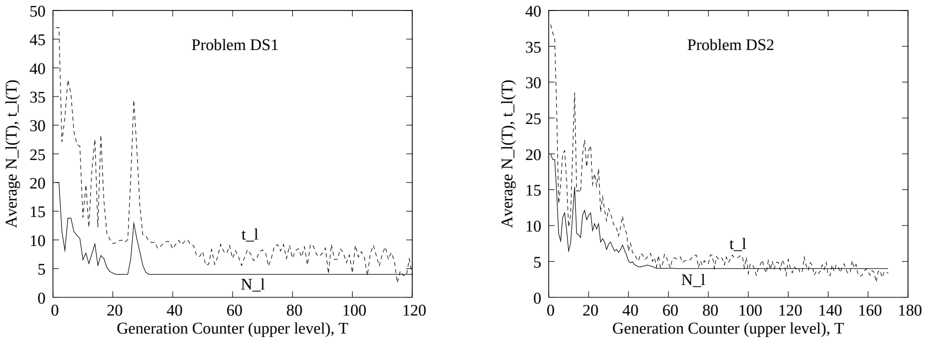

Figure 22: Variation of N_l and t_l with upper level generation for problems DS1 (left figure) and DS2 (right figure). The algorithm adapts these two important parameters automatically.

the lower level termination condition, although initially as large as 47 generations were needed. Interestingly, whenever there is a surge in N_l, a similar surge is noticed for t_l as well. This supports our argument about possible discovery of new and isolated $\mathbf{x}_u$ variable vectors occasionally. Our previous implementation (Deb and Sinha, 2009a) used predefined fixed values for N_l and t_l throughout, thereby requiring a unnecessarily large computational effort. In our current hybrid algorithm, we reduce the computational burden by using the self-adaptive updates of the two most crucial parameters involving computations in the lower level optimization task.

6.5 Problem DS2

This problem also has 20 variables in total, thereby motivating us to use $N_u = 400$. Here too, we use $\tau = 1$ at first and postpone a discussion on the difficult version of the problem with $\tau = -1$ later. Figure 23 shows the final archive solutions from a typical run, indicating the efficiency of our procedure. Our earlier BLEMO algorithm (Deb and Sinha, 2009a) could only solve at most a four-variable version of this problem. The hypervolumes for the obtained attainment surfaces are 0.5776, 0.6542 and 0.6629,

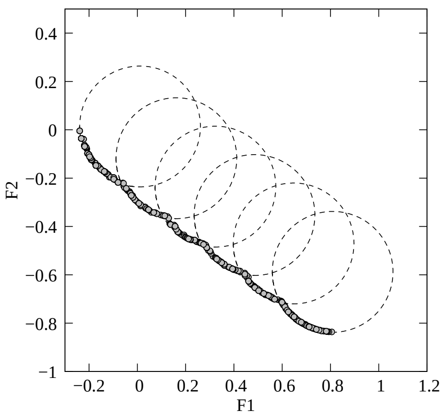

Figure 23: Final archive solutions for problem DS2.

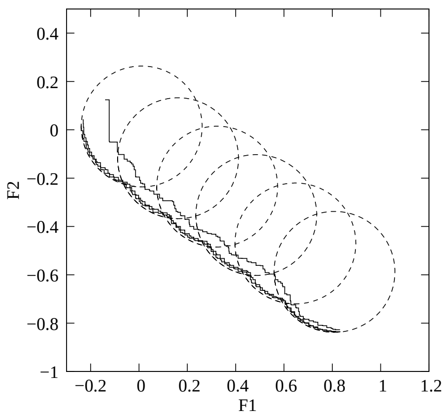

Figure 24: Attainment surfaces (0%, 50%, 75% and 100%) for problem DS2 from 21 runs.

respectively. The 0%, 50%, and 75% attainment surfaces (Figure 24) are very close to each other, indicating that at least 75% of the runs are close to each other. The gap between 75% and 100% attainment surfaces indicates that a few solutions are found to be not so close to the true Pareto-optimal frontier in this problem.

Figure 25 confirms that $N_u = 400$ is the best choice of N_u to achieve a similar hypervolume measure with the smallest number of overall function evaluations. The right side plot of Figure 22 shows that N_l and t_l starts with large values but drops to small values adaptively to make the optimization process efficient and computationally fast.

6.6 Problem DS3

This problem has 20 variables. Thus, we have used $N_u = 400$. Figure 26 shows the final archive population and Figure 27 shows corresponding attainment surface plot. Our earlier algorithm was able to solve a maximum of eight-variable version of this problem. The hypervolumes for the attainment surfaces are 0.5528, 0.5705 and 0.5759, respectively. All 21 runs find the entire Pareto-optimal front with a maximum difference in hypervolume value of about 4%.

6.7 Problem DS4

This problem is considered for 10 variables; thus we use $N_u = 200$. Figures 28 and 29 show the archive population and the attainment surface for this problem. The hypervolumes for the obtained attainment surfaces are 0.5077, 0.5241 and 0.5264, respectively. The maximum difference in hypervolume measures in 21 runs is about 3.6% only, indicating the robustness of the proposed procedure.

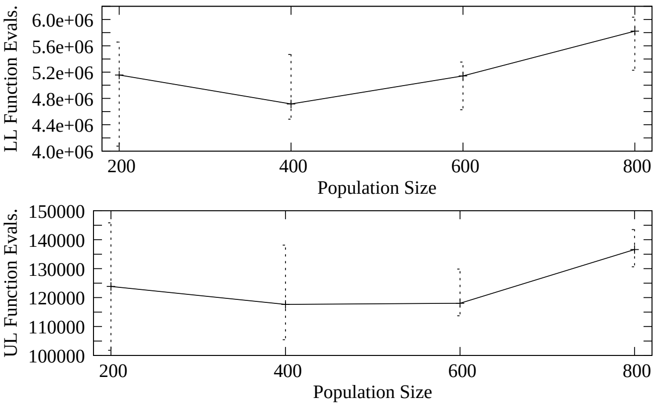

Figure 25: Average lower and upper level function evaluations with different population sizes for problem DS2 indicates $N_u = 400$ is the best choice. 21 runs are performed in each case.

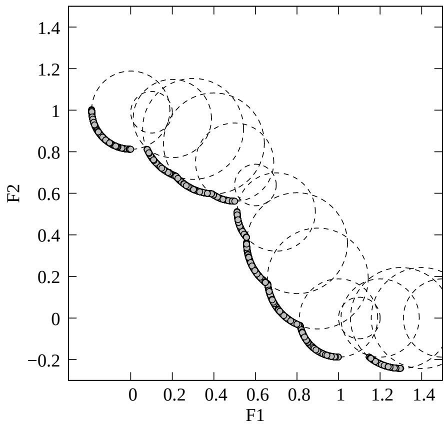

Figure 26: Final archive solutions for problem DS3.

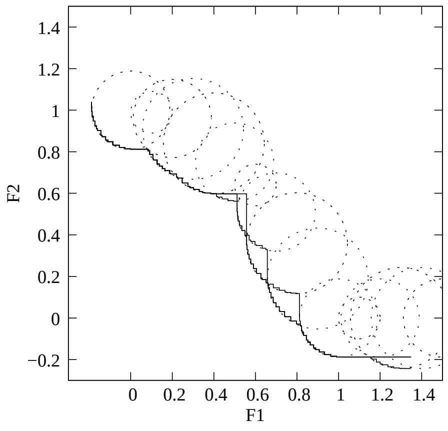

Figure 27: Attainment surfaces (0%, 50% and 100%) for problem DS3 from 21 runs.

6.8 Problem DS5

This problem is also considered with 10 variables. We have used $N_u = 200$. Figure 30 shows the final archive population for a typical run. Figure 31 shows the corresponding attainment surface plots, which are very close to each other indicating the efficacy of the procedure. The hypervolumes for the obtained attainment surfaces are 0.5216, 0.5281 and 0.5308, respectively. The difference in hypervolume is only 1.7% for this problem.

6.9 Computational Efforts

Next, we investigate two aspects related to the computational issues. First, we record the total function evaluations needed by the overall algorithm to achieve the specified

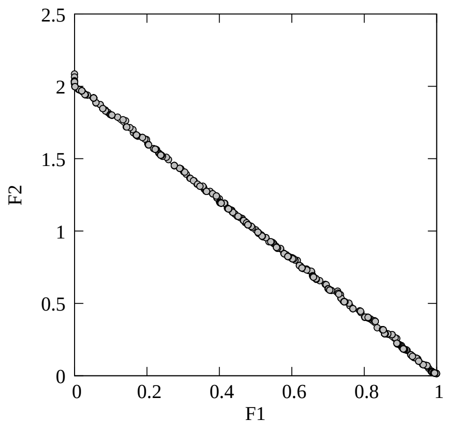

Figure 28: Final archive solutions for problem DS4.

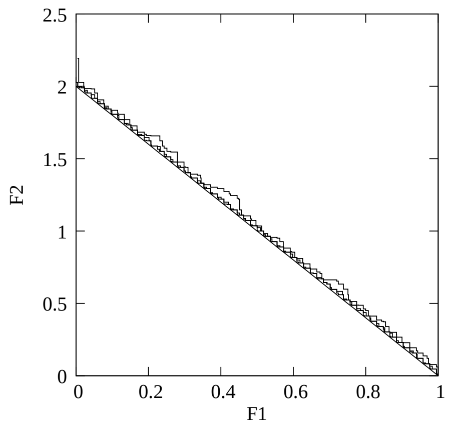

Figure 29: Attainment surfaces (0%, 50% and 100%) for problem DS4 from 21 runs.

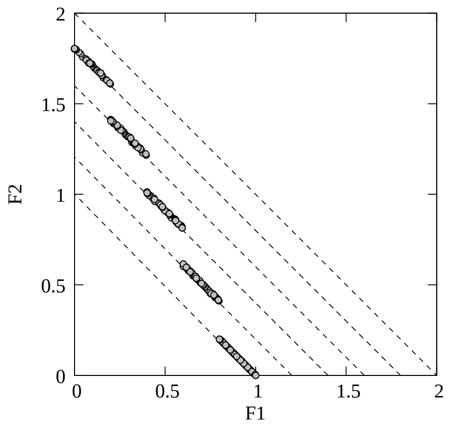

Figure 30: Final archive solutions for problem DS5.

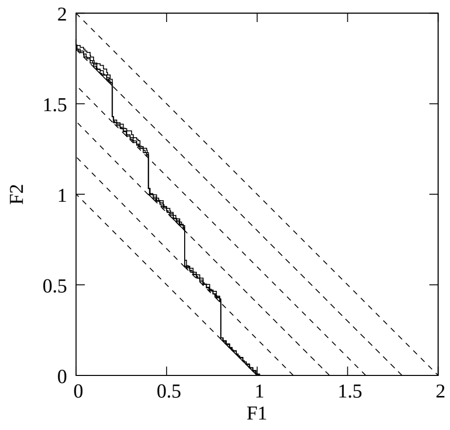

Figure 31: Attainment surfaces (0%, 50% and 100%) for problem DS5 from 21 runs.

termination criterion ($\epsilon_u = 0.0001$) and the same needed exclusively for the lower level optimization task, which includes the local search. Table 1 shows these values for the best, median and worst of 21 simulation runs for all eight problems. It is clear from the table that the most of the computational efforts are spent in the lower level solution evaluations. Despite our efforts being different from a nested algorithm in not solving a lower level problem all the way for every upper level solution, the nature of bilevel programming problem demands that the lower level optimization task must be emphasized. The use of archive in sizing lower level subpopulations in a self-adaptive manner and the use of a coarse terminating condition for lower level optimization task enabled our algorithm to use comparatively smaller number of function evaluations than that would be needed in a nested algorithm. We compare the function evalua-

Table 1: Total function evaluations for the upper and lower level (21 runs). The lower level function evaluations include the evaluations of local search as well.

Pr. No.	#	Best		Median		Worst	
	var.	Total LL FE	Total UL FE	Total LL FE	Total UL FE	Total LL FE	Total UL FE
TP1	3	555,410	12,438	629,590	14,163	70,0634	16,146
TP2	15	264,675	17,262	319,499	18,733	355,904	21,429
TP4	5	1,624,658	37,218	1,927,022	39,960	2,036,208	41,570
DS1	20	2,819,770	87,582	3,423,544	91,852	3,829,812	107,659
DS2	20	4,484,580	105,439	4,695,352	116,605	5,467,633	138,107
DS3	20	3,970,411	112,560	4,725,596	118,848	5,265,074	125,438
DS4	10	1,356,598	38,127	1,435,344	53,548	1,675,422	59,047
DS5	10	1,666,953	47,127	1,791,511	56,725	2,197,470	71,246

tions of H-BLEMO with a nested bilevel optimization algorithm later in Subsection 9 to illustrate this fact.

To investigate the computational efforts needed in the local search operator, we record the average function evaluations in the local search procedure and in the overall lower level optimization task. Table 2 presents the results. The local search efforts

Table 2: Function evaluations needed by the local search.

Pr. No.	Avg. Local search FE	FE per call of local search	Avg. Lower level FE	% Local search FE
TP1	91,382.2	19.89	640,696.1	0.14
TP2	232,426.9	108.59	324,093.8	0.72
TP4	504,084.2	43.93	1,940,248.9	0.26
DS1	1,996,769.6	77.90	3,744,071.8	0.53
DS2	2,620,456.7	97.79	5,039,594.9	0.52
DS3	2,867,432.3	75.43	4,841,572.7	0.59
DS4	1,020,262.2	60.24	1,472,883.7	0.69
DS5	1,427,828.6	68.56	1,893,837.5	0.75

vary from problem to problem. However, it contributes to more than half the computations in the lower level optimization task in most problems. Thus, putting both tables together, we conclude that the effort of the local search is about 50% to the overall computational efforts of the proposed H-BLEMO algorithm. Of course, this quantity depends on the chosen termination criteria for the lower level, upper level, and the local search optimization tasks. Nevertheless, the termination conditions chosen for this study seemed to be adequate for the overall algorithm to work on all the problems of this paper and an effort of about 50% for achieving guaranteed convergence (with respect to satisfying KKT conditions) by the local search may seem a reasonable proposition. However, future efforts may be spent on devising quicker local search and lower level optimization tasks for reducing the overall computational effort. As we find next, the local search operator may have a bigger role to play than simply guaranteeing convergence to a locally Pareto-optimal frontier.

7 Introducing Further Difficulties in DS1 and DS2

In this section, we use $\tau = -1$ in 20-variable DS1 and DS2 problems (refer to equations 7 and 9, respectively). As discussed earlier, this choice makes a conflicting scenario in the working principles between upper and lower level optimization tasks. To demonstrate the difficulties caused by this setting, we consider three different algorithms: (i) A1: H-BLEMO procedure with local search, (ii) A2: H-BLEMO procedure without local search, and (iii) A3: BLEMO procedure (Deb and Sinha, 2009a). All parameter values are the same as before and are maintained for all three algorithms. To compare the performances, we first identify the nadir point in each problem and then compute the true hypervolume measure H^* by computing 10,000 well-distributed Pareto-optimal points. Thereafter, we record a normalized hypervolume measure $DH(T) = (H(T) - H^*)/H^*$ at each upper level generation T from the hypervolume ($H(T)$) computed using the true nadir point as the reference point. Note that if the $DH(T)$ value reaches zero, the corresponding algorithm can be said to reach the true Pareto-optimal front. A negative value of $DH(T)$ indicates that the obtained set of solutions lie above the true Pareto-optimal front so that the hypervolume $H(T)$ is smaller than H^*. This is an usual scenario of a multi-objective optimization run, where solutions are usually worse than the true Pareto-optimal front in the beginning of a run (having negative $DH(T)$ values) and then the solutions come closer to the Pareto-optimal front with generation.

However, in the case of $\tau = -1$ for both DS1 and DS2 problems, if the lower level problem is unable to find the true Pareto-optimal points, the corresponding solutions lie much below the true Pareto-optimal front in the upper level objective space. Thus, the hypervolume measure $H(T)$ in the upper level objective space will be very large, but this may be construed as a case in which the obtained solutions are infeasible. Since $H(T)$ will be larger than H^* in this case, the $DH(T)$ value will be positive.

Figures 32 shows the $DH(T)$ measure for all three algorithms with generation counter T for problem DS1. It is clear that only our proposed hybrid algorithm (A1) is able to find the true Pareto-optimal front, by approaching a $DH(T)$ value of zero from an initial negative value. Other two algorithms get stuck to a large positive $DH(T)$ value, indicating that the obtained set of solutions lie in the infeasible region beneath the true Pareto-optimal front in the upper level objective space. Our hybrid algorithm without the local search (A2) is somewhat better than our previous algorithm (A3). A similar performance is also observed for problem DS2, shown in Figure 33. These scenarios clearly indicate the importance of using the local search approach in the lower level optimization task.

Having shown the importance of the local search and self-adaptive update of NSGA-II parameters for the lower level optimization task, we now investigate the proposed H-BLEMO algorithm's extent of sensitivity to the parameter τ. We compare the function evaluations for both DS1 and DS2 problems with $\tau = +1$ and $\tau = -1$ in Table 3. It is interesting to note that both problems are harder with $\tau = -1$, but the H-BLEMO algorithm is not overly sensitive to the difficulty caused by the discrepancy in search directions in lower and upper level problems achieved with $\tau = -1$. For the median performance, DS1 and DS2 require an increase in overall function evaluations of 5.1% and 5.6%, respectively. However, an absence of local search or our previous BLEMO algorithm is unable to solve the $\tau = -1$ version of both problems (Figures 32 and 33).

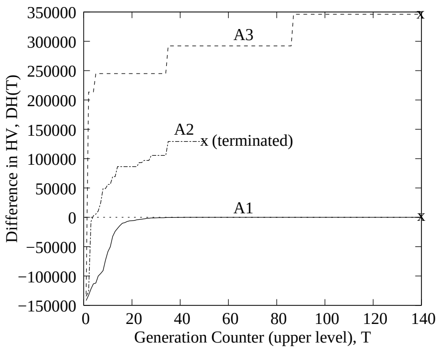

Figure 32: Difference in hypervolume from ideal $DH(T)$ with upper level generation counter T for problem DS1 using three algorithms. Only algorithm A1 (H-BLEMO) reaches the Pareto-optimal front by making $DH(T) = 0$.

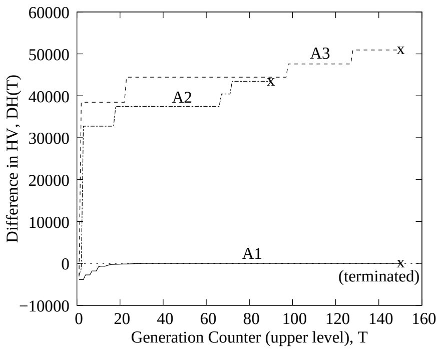

Figure 33: Difference in hypervolume from ideal $DH(T)$ with upper level generation counter T for problem DS2 using three algorithm. Only algorithm A1 (H-BLEMO) reaches the Pareto-optimal front by making $DH(T) = 0$.

Table 3: Comparison of function evaluations for $\tau = -1$ and $\tau = +1$ cases with the H-BLEMO algorithm.

Prob. No.	Best		Median		Worst	
	Total LL FE	Total UL FE	Total LL FE	Total UL FE	Total LL FE	Total UL FE
DS1 ($\tau = +1$)	2,819,770	87,582	3,423,544	91,852	3,829,812	107,659
DS1 ($\tau = -1$)	3,139,381	92,624	3,597,090	98,934	4,087,557	113,430
DS2 ($\tau = +1$)	4,484,580	105,439	4,695,352	116,605	5,467,633	138,107
DS2 ($\tau = -1$)	4,796,131	112,563	4,958,593	122,413	5,731,016	144,428

8 Scalability Study

In this section, we consider DS1 and DS2 (with $\tau = 1$) and show the scalability of our proposed procedure up to 40 variables. For this purpose, we consider four different variable sizes: $n = 10$, 20, 30 and 40. Based on parametric studies performed on these problems in section 6, we set $N_u = 20n$. All other parameters are automatically set in a self-adaptive manner during the course of a simulation, as before.

Figure 34 shows the variation of function evaluations for obtaining a fixed termination criterion on normalized hypervolume measure ($H_u < 0.0001$) calculated using the upper level objective values for problem DS1. Since the vertical axis is plotted in a logarithmic scale and the relationship is found to be sub-linear, the hybrid methodology performs better than an exponential algorithm. The break-up of computations needed in the local search, lower level NSGA-II and upper level NSGA-II indicate that majority of the computations is spent in the lower level optimization task. This is an important insight to the working of the proposed H-BLEMO algorithm and suggests that further efforts must be put in making the lower level optimization more computationally efficient. Figure 35 shows the similar outcome for problem DS2, but a comparison with that for problem DS1 indicates that DS2 is more difficult to be solved with an increase

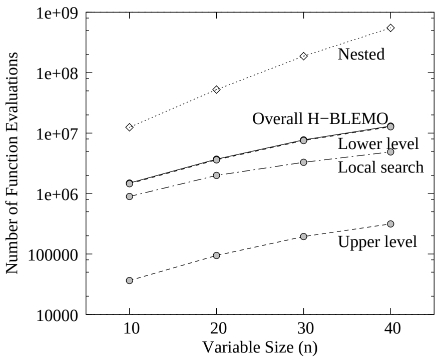

Figure 34: Variation of function evaluations with problem size n for DS1.

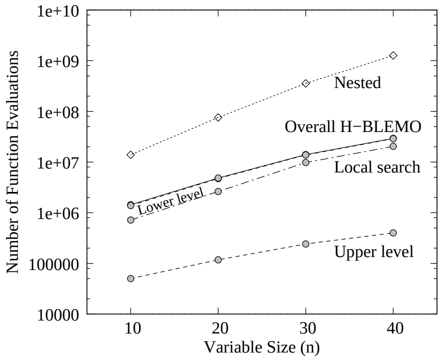

Figure 35: Variation of function evaluations with problem size n for DS2.

in problem size than DS1.

9 Comparison with a Nested Algorithm

We have argued before that by allowing lower level and upper level NSGA-IIs to proceed partially in tandem, we have created a computationally efficient and accurate algorithm which progresses towards the true Pareto-optimal front on a number of difficult problems (Section 6). The algorithm is even found to converge in problems in which there is a conflict between upper and lower level problems (Section 7). The proposed algorithm is also found to solve scaled-up problems up to 40 real-parameter variables (Section 8). In this section, we compare the proposed H-BLEMO algorithm with an efficient yet nested bilevel optimization algorithm using the NSGA-II-cum-local-search procedure. This algorithm uses a fixed population structure, but for every $\mathbf{x}_u$, the lower level optimization is terminated by performing a local search to all non-dominated solutions of the final lower level NSGA-II population. The termination criterion for lower and upper level NSGA-IIs and that for the local search procedure are identical to that in H-BLEMO algorithm. Since for every $\mathbf{x}_u$, we find a set of well-converged and well-distributed lower level Pareto-optimal solutions, this approach is truly a nested bilevel optimization procedure.

For the simulation with this nested algorithm on DS1 and DS2 problems ($\tau = 1$), we use $N_u = 400$. To make a fair comparison, we use the same subpopulation size N_l as that was used in the very first iteration of our H-BLEMO algorithm using equation (15). The number of generations for the lower level NSGA-II is kept fixed for all upper level generations to that computed by equation (14) in the initial generation. Similar archiving strategy and other NSGA-II parameter values are used as before. Table 4 shows the comparison of overall function evaluations needed by the nested algorithm and by the hybrid BLEMO algorithm. The table shows that for both problems, the nested algorithm takes at least one order of magnitude of more function evaluations to find a set of solutions having an identical hypervolume measure. The difference between our proposed algorithm and the nested procedure widens with an increase in number of decision variables. The median number of function evaluations are also plotted in Figures 34 and 35. The computational efficacy of our proposed hybrid approach and

Table 4: Comparison of function evaluations needed by a nested algorithm and by H-BLEMO on problems DS1 and DS2. Results from 21 runs are summarized.

Problem DS1						
n	Algo.	Median			Min. overall	Max. overall
		Lower FE	Upper FE	Overall FE	FE	FE
10	Nested	12,124,083	354,114	12,478,197	11,733,871	14,547,725
10	Hybrid	1,454,194	36,315	1,490,509	1,437,038	1,535,329
20	Nested	51,142,994	1,349,335	52,492,329	42,291,810	62,525,401
20	Hybrid	3,612,711	94,409	3,707,120	2,907,352	3,937,471
30	Nested	182,881,535	4,727,534	187,609,069	184,128,609	218,164,646
30	Hybrid	7,527,677	194,324	7,722,001	6,458,856	8,726,543
40	Nested	538,064,283	13,397,967	551,462,250	445,897,063	587,385,335
40	Hybrid	12,744,092	313,861	13,057,953	10,666,017	15,146,652

Problem DS2						
n	Algo.	Median			Min. overall	Max. overall
		Lower FE	Upper FE	Overall FE	FE	FE
10	Nested	13,408,837	473,208	13,882,045	11,952,650	15,550,144
10	Hybrid	1,386,258	50,122	1,436,380	1,152,015	1,655,821
20	Nested	74,016,721	1,780,882	75,797,603	71,988,726	90,575,216
20	Hybrid	4,716,205	117,632	4,833,837	4,590,019	5,605,740
30	Nested	349,242,956	5,973,849	355,216,805	316,279,784	391,648,693
30	Hybrid	13,770,098	241,474	14,011,572	14,000,057	15,385,316
40	Nested	1,248,848,767	17,046,212	1,265,894,979	1,102,945,724	1,366,734,137
40	Hybrid	28,870,856	399,316	29,270,172	24,725,683	30,135,983

difference of our approach from a nested approach are clearly evident from these plots.

10 Conclusions

Bilevel programming problems appear commonly in practice, however due to complications associated in solving them, often they are treated as single-level optimization problems by adopting approximate solution principles for the lower level problem. Although single-objective bilevel programming problems are studied extensively, there does not seem to be enough emphasis for multi-objective bilevel optimization studies. This paper has made a significant step in presenting past key research efforts, identifying insights for solving such problems, suggesting scalable test problems, and implementing a viable hybrid evolutionary-cum-local-search algorithm. The proposed algorithm is also self-adaptive, allowing an automatic update of the key parameters from generation to generation. Simulation results on eight different multi-objective bilevel programming problems and their variants, and a systematic overall analysis amply demonstrate the usefulness of the proposed approach. Importantly, due to the multi-solution nature of the problem and intertwined interactions between both levels of optimization, this study helps to showcase the importance of evolutionary algorithms in solving such complex problems.

The study of multi-objective bilevel problem solving methodologies elevates every aspect of an optimization effort at a higher level, thereby making them interesting and challenging to pursue. Although formulation of theoretical optimality conditions is possible and has been suggested, viable methodologies to implement them in practice are challenging. Although a nested implementation of lower level optimization from the upper level is an easy fix-up and has been attempted by many researchers, suitable practical algorithms coupling the two levels of optimizations in a computationally efficient manner is not easy and definitely challenging. Developing hybrid bilevel optimization algorithms involving various optimization techniques, such as evolutionary, classical, mathematical, simulated annealing methods etc., are possible in both levels independently or synergistically, allowing a plethora of implementational opportunities. This paper has demonstrated one such implementation involving evolutionary algorithms, a classical local search method, and a mathematical optimality condition for termination, but certainly many other ideas are possible and must be pursued urgently.

Every upper level Pareto-optimal solution comes from a lower level Pareto-optimal solution set. Thus, a decision making technique for choosing a single preferred solution in such scenarios must involve both upper and lower level objective space considerations, which may require and give birth to new and interactive multiple criterion decision making (MCDM) methodologies. Finally, a successful implementation and understanding of multi-objective bilevel programming tasks should motivate us to understand and develop higher level (say, three or four-level) optimization algorithms, which should also be of great interest to computational science due to the hierarchical nature of systems approach often followed in complex computational problem solving tasks today.

Acknowledgments

Authors wish to thank Academy of Finland and Foundation of Helsinki School of Economics (under grant 118319) for their support of this study.

References

Abass, S. A. (2005). Bilevel programming approach applied to the flow shop scheduling problem under fuzziness. *Computational Management Science*, **4**(4), 279–293.

Alexandrov, N. and Dennis, J. E. (1994). Algorithms for bilevel optimization. In *AIAA/NASA/USAF/ISSMO Symposium on Multidisciplinary Analyis and Optimization*, pages 810–816.

Bard, J. F. (1998). *Practical Bilevel Optimization: Algorithms and Applications*. The Netherlands: Kluwer.

Bianco, L., Caramia, M., and Giordani, S. (2009). A bilevel flow model for hazmat transportation network design. *Transportation Research. Part C: Emerging technologies*, **17**(2), 175–196.

Byrd, R. H., Nocedal, J., and Waltz, R. A. (2006). *KNITRO: An integrated package for nonlinear optimization*, pages 35–59. Springer-Verlag.

Calamai, P. H. and Vicente, L. N. (1994). Generating quadratic bilevel programming test problems. *ACM Trans. Math. Software*, **20**(1), 103–119.

Colson, B., Marcotte, P., and Savard, G. (2007). An overview of bilevel optimization. *Annals of Operational Research*, **153**, 235–256.

Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms*. Wiley, Chichester, UK.

Deb, K. and Agrawal, R. B. (1995). Simulated binary crossover for continuous search space. *Complex Systems*, **9**(2), 115–148.

Deb, K. and Sinha, A. (2009a). Constructing test problems for bilevel evolutionary multi-objective optimization. In *Proceedings of the Congress on Evolutionary Computation (CEC-2009)*. Piscataway, NJ: IEEE Press. (Also KanGAL Report No. 2008010).

Deb, K. and Sinha, A. (2009b). Solving bilevel multi-objective optimization problems using evolutionary algorithms. In *Proceedings of Evolutionary Multi-Criterion Optimization (EMO-2009)*, pages 110–124. Heidelberg: Springer.

Deb, K., Agrawal, S., Pratap, A., and Meyarivan, T. (2002). A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, **6**(2), 182–197.

Deb, K., Thiele, L., Laumanns, M., and Zitzler, E. (2005). Scalable test problems for evolutionary multi-objective optimization. In A. Abraham, L. Jain, and R. Goldberg, editors, *Evolutionary Multiobjective Optimization*, pages 105–145. London: Springer-Verlag.

Dempe, S. (2002). *Foundations of bilevel programming*. Kluwer, Dordrecht.

Dempe, S., Dutta, J., and Lohse, S. (2006). Optimality conditions for bilevel programming problems. *Optimization*, **55**(5-6), 505–524.

Dimitriou, L., Tsekeris, T., and Stathopoulos, A. (2008). Genetic computation of road network design and pricing Stackelberg games with multi-class users. In *Proceedings of EvoWorkshops*, pages 669–678. Also LNCS 4974.

Eichfelder, G. (2007). Solving nonlinear multiobjective bilevel optimization problems with coupled upper level constraints. Technical Report Preprint No. 320, Preprint-Series of the Institute of Applied Mathematics, Univ. Erlangen-Nürnberg, Germany.

Eichfelder, G. (2008). Multiobjective bilevel optimization. *Mathematical Programming*. DOI 10.1007/s10107-008-0259-0.

Fampa, M., Barroso, L. A., Candal, D., and Simonetti, L. (2008). Bilevel optimization applied to strategic pricing in competitive electricity markets. *Comput. Optim. Appl.*, **39**, 121–142.

Fliege, J. and Vicente, L. N. (2006). Multicriteria approach to bilevel optimization. *Journal of Optimization Theory and Applications*, **131**(2), 209–225.

Fonseca, C. M. and Fleming, P. J. (1996). On the performance assessment and comparison of stochastic multiobjective optimizers. In H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature (PPSN IV)*, pages 584–593. Berlin: Springer. Also available as Lecture Notes in Computer Science 1141.

Fudenberg, D. and Tirole, J. (1993). *Game theory*. MIT Press.

Gaur, A. and Arora, S. R. (2008). Multi-level multi-attribute multi-objective integer linear programming problem. *AMO-Advanced Modeling and Optimization*, **10**(2), 297–322.

Halter, W. and Mostaghim, S. (2006). Bilevel optimization of multi-component chemical systems using particle swarm optimization. In *Proceedings of World Congress on Computational Intelligence (WCCI-2006)*, pages 1240–1247.

Hecheng, L. and Wang, Y. (2007). A genetic algorithm for solving a special class of nonlinear bilevel programming problems. In *7th international conference on Computational Science, Part IV: ICCS 2007*, pages 1159–1162. Also LNCS 4490.

Herskovits, J., Leontiev, A., Dias, G., and Santos, G. (2000). Contact shape optimization: A bilevel programming approach. *Struct. Multidisc. Optimization*, **20**, 214–221.

Koh, A. (2007). Solving transportation bi-level programs with differential evolution. In *2007 IEEE Congress on Evolutionary Computation (CEC-2007)*, pages 2243–2250. IEEE Press.

Li, H. and Wang, Y. (2007). A hybrid genetic algorithm for solving nonlinear bilevel programming problems based on the simplex method. In *Third International Conference on Natural Computation (ICNC 2007)*, pages 91–95.

Li, X., Tian, P., and Min, X. (2006). A hierarchical particle swarm optimization for solving bilevel programming problems. In *Proceedings of Artificial Intelligence and Soft Computing (ICAISC 2006)*, pages 1169–1178. Also LNAI 4029.

Mathieu, R., Pittard, L., and Anandalingam, G. (1994). Genetic algorithm based approach to bi-level linear programming. *Operations Research*, **28**(1), 1–21.

Miettinen, K. (1999). *Nonlinear Multiobjective Optimization*. Kluwer, Boston.

Nolle, L. (2006). On a hill-climbing algorithm with adaptive step size: Towards a control parameter-less black-box optimization algorithm. In B. Reusch, editor, *Computational Intelligence, Theory and Applications*, pages 587–596. Berlin: Springer-Verlag.

Oduguwa, V. and Roy, R. (2002). Bi-level optimisation using genetic algorithm. In *Proceedings of the 2002 IEEE International Conference on Artificial Intelligence Systems (ICAIS-02)*, pages 322–327.

Pakala, R. R. (1993). *A Study on Applications of Stackelberg Game Strategies in Concurrent Design Models*. Master's thesis, Department of Mechanical Engineering: University of Houston.

Paruchuri, P., Pearce, J. P., Marecki, J., Tambe, M., Ordonez, F., and Kraus, S. (2008). Efficient algorithms to solve bayesian Stackelberg games for security applications. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence*, pages 1559–1562.

Rao, S. S. (1984). *Optimization: Theory and Applications*. Wiley, New York.

Reklaitis, G. V., Ravindran, A., and Ragsdell, K. M. (1983). *Engineering Optimization Methods and Applications*. New York : Wiley.

Shi, X. and Xia, H. S. (2001). Model and interactive algorithm of bi-level multi-objective decision-making with multiple interconnected decision makers. *Journal of Multi-Criteria Decision Analysis*, **10**(1), 27–34.

Sinha, A. and Deb, K. (2009). Towards understanding evolutionary bilevel multi-objective optimization algorithm. Technical Report Proceedings of the IFAC Workshop on Control Applications of Optimization (6-8 May, 2009, Jyväskylä, Finland), Kanpur, Indian Institute of Technology, India. (Also KanGAL Report No. 2008006).

Sun, D., Benekohal, R. F., and Waller, S. T. (2006). Bi-level programming formulation and heuristic solution approach for dynamic traffic signal optimization. *Computer-Aided Civil and Infrastructure Engineering*, **21**(5), 321–333.

Vicente, L. N. and Calamai, P. H. (2004). Bilevel and multilevel programming: A bibliography review. *Journal of Global Optimization*, **5**(3), 291–306.

Wang, G., Wan, Z., Wang, X., and Lv, Y. (2008). Genetic algorithm based on simplex method for solving linear-quadratic bilevel programming problem. *Comput. Math. Appl.*, **56**(10), 2550–2555.

Wang, G.-M., Wang, X.-J., Wan, Z.-P., and Jia, S.-H. (2007). An adaptive genetic algorithm for solving bilevel linear programming problem. *Applied Mathematics and Mechanics*, **28**(12), 1605–1612.

Wang, J. F. and Periaux, J. (2001). Multi-point optimization using gas and Nash/Stackelberg games for high lift multi-airfoil design in aerodynamics. In *Proceedings of the 2001 Congress on Evolutionary Computation (CEC-2001)*, pages 552–559.

Wang, Y., Jiao, Y.-C., and Li, H. (2005). An evolutionary algorithm for solving nonlinear bilevel programming based on a new constraint-handling scheme. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, **35**(2), 221–232.

Wierzbicki, A. P. (1980). The use of reference objectives in multiobjective optimization. In G. Fandel and T. Gal, editors, *Multiple Criteria Decision Making Theory and Applications*, pages 468–486. Berlin: Springer-Verlag.

Yin, Y. (2000). Genetic algorithm based approach for bilevel programming models. *Journal of Transportation Engineering*, **126**(2), 115–120.

Zhang, G., Liu, J., and Dillon, T. (2007). Decntralized multi-objective bilevel decision making with fuzzy demands. *Knowledge-Based Systems*, **20**, 495–507.

5 Bilevel multi-objective optimization problem solving using progressively interactive EMO

A. Sinha

In Proceedings of the Sixth International Conference on Evolutionary Multi-criterion Optimization (EMO-2011), In Press, 2011.

Bilevel Multi-Objective Optimization Problem Solving Using Progressively Interactive EMO

Ankur Sinha Ankur.Sinha@aalto.fi
Department of Business Technology, Aalto University School of Economics
PO Box 21210, FIN-00076 Aalto, Helsinki, Finland

Abstract

Bilevel multi-objective optimization problems are known to be highly complex optimization tasks which require every feasible upper-level solution to satisfy optimality of a lower-level optimization problem. Multi-objective bilevel problems are commonly found in practice and high computation cost needed to solve such problems motivates to use multi-criterion decision making ideas to efficiently handle such problems. Multi-objective bilevel problems have been previously handled using an evolutionary multi-objective optimization (EMO) algorithm where the entire Pareto set is produced. In order to save the computational expense, a progressively interactive EMO for bilevel problems has been presented where preference information from the decision maker at the upper level of the bilevel problem is used to guide the algorithm towards the most preferred solution (a single solution point). The procedure has been evaluated on a set of five DS test problems suggested by Deb and Sinha. A comparison for the number of function evaluations has been done with a recently suggested Hybrid Bilevel Evolutionary Multi-objective Optimization algorithm which produces the entire upper level Pareto-front for a bilevel problem.

Keywords

Genetic algorithms, evolutionary algorithms, bilevel optimization, multi-objective optimization, evolutionary programming, multi-criteria decision making, hybrid evolutionary algorithms, sequential quadratic programming.

1 Introduction

Bilevel programming problems are often found in practice [25] where the feasibility of an upper level solution is decided by a lower level optimization problem. The qualification for an upper level solution to be feasible is that it should be an optimal candidate from a lower level optimization problem. This requirement consequentially makes a bilevel problem very difficult to handle. Multiple objectives at both the levels of a bilevel problem further adds to the complexity. Because of difficulty in searching and defining optimal solutions for bilevel multi-objective optimization problems [11], not many solution methodologies to such problems have been explored. One of the recent advances made in this direction is by Deb and Sinha [9] where the entire Pareto set at the upper level of the bilevel multi-objective problem is explored. The method, though successful in handling complex bilevel multi-objective test problems, is computationally expensive and requires high function evaluations, particularly at the lower level. High computational expense associated to such problems provides a motivation to explore a different solution methodology.

Concepts from a Progressively Interactive Evolutionary Multi-objective Optimization algorithm (PI-EMO-VF) [10] has been integrated with the Hybrid Bilevel Evolutionary Multi-objective Optimization algorithm (HBLEMO) [9] in this paper. In the suggested methodology, preference information from the decision maker at the upper level is used to direct the search towards the most preferred solution. Incorporating preferences from the decision maker in the optimization run makes the search process more efficient in terms of function evaluations as well as accuracy. The integrated methodology proposed in this paper, interacts with the decision maker after every few generations of an evolutionary algorithm and is different from an a posteriori approach, as it explores only the most preferred point. An a posteriori approach like the HBLEMO and other evolutionary multi-objective optimization algorithms [5, 26] produce the entire efficient frontier as the final solution and then a decision maker is asked to pick up the most preferred point. However, an a posteriori approach is not a viable methodology for problems which are computationally expensive and/or involve high number of objectives (more than three) where EMOs tend to suffer in convergence as well as maintaining diversity.

In this paper, the bilevel multi-objective problem has been described initially and then the integrated procedure, Progressively Interactive Hybrid Bilevel Evolutionary Multi-objective Optimization (PI-HBLEMO) algorithm, has been discussed. The performance of the PI-HBLEMO algorithm has been shown on a set of five DS test problems [9, 6] and a comparison for the savings in computational cost has been done with a posteriori HBLEMO approach.

2 Recent Studies

In the context of bilevel single-objective optimization problems a number of studies exist, including some useful reviews [3, 21], test problem generators [2], and some evolutionary algorithm (EA) studies [18, 17, 24, 16, 23]. Stackelberg games [13, 22], which have been widely studied, are also in principle similar to a single-objective bilevel problem. However, not many studies can be found in case of bilevel multi-objective optimization problems. The bilevel multi-objective problems have not received much attention, either from the classical researchers or from the researchers in the evolutionary community.

Eichfelder [12] worked on a classical approach on handling multi-objective bilevel problems, but the nature of the approach made it limited to handle only problems with few decision variables. Halter et al. [15] used a particle swarm optimization (PSO) procedure at both the levels of the bilevel multi-objective problem but the application problems they used had linearity in the lower level. A specialized linear multi-objective PSO algorithm was used at the lower level, and a nested strategy was utilized at the upper level.

Recently, Deb and Sinha have proposed a Hybrid Bilevel Evolutionary Multi-objective Optimization algorithm (HBLEMO) [9] using NSGA-II to solve both level problems in a synchronous manner. Former versions of the HBLEMO algorithm can be found in the conference publications [6, 8, 19, 7]. The work in this paper extends the HBLEMO algorithm by allowing the decision maker to interact with the algorithm.

3 Multi-Objective Bilevel Optimization Problems

In a multi-objective bilevel optimization problem there are two levels of multi-objective optimization tasks. A solution is considered feasible at the upper level only if it is a Pareto-optimal member to a lower level optimization problem [9]. A generic multi-objective bilevel optimization problem can be described as follows. In the formulation there are M number of objectives at the upper level and m number of objectives at the lower level:

$$\begin{aligned} \text{Minimize}_{(\mathbf{x}_u,\mathbf{x}_l)}\ & \mathbf{F}(\mathbf{x}) = (F_1(\mathbf{x}), \ldots, F_M(\mathbf{x})), \\ \text{subject to}\ & \mathbf{x}_l \in \text{argmin}_{(\mathbf{x}_l)} \left\{ \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x})) \,\middle|\, \mathbf{g}(\mathbf{x}) \geq \mathbf{0}, \mathbf{h}(\mathbf{x}) = \mathbf{0} \right\}, \\ & \mathbf{G}(\mathbf{x}) \geq \mathbf{0}, \mathbf{H}(\mathbf{x}) = \mathbf{0}, \\ & x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = 1, \ldots, n. \end{aligned} \tag{1}$$

In the above formulation, $F_1(\mathbf{x}), \ldots, F_M(\mathbf{x})$ are upper level objective functions which are M in number and $f_1(\mathbf{x}), \ldots, f_m(\mathbf{x})$ are lower level objective functions which are m in number.

The constraint functions $\mathbf{g}(\mathbf{x})$ and $\mathbf{h}(\mathbf{x})$ determine the feasible space for the lower level problem. The decision vector, $\mathbf{x}$, contains the variables to be optimized at the upper level. It is composed of two smaller vectors $\mathbf{x}_u$ and $\mathbf{x}_l$, such that $\mathbf{x} = (\mathbf{x}_u, \mathbf{x}_l)$. While solving the lower level problem, it is important to note that the lower level problem is optimized with respect to the variables $\mathbf{x}_l$ and the variables $\mathbf{x}_u$ act as fixed parameters for the problem. Therefore, the Pareto-optimal solution set to the lower level problem can be represented as $\mathbf{x}_l^*(\mathbf{x}_u)$. This representation means that the upper level variables $\mathbf{x}_u$, act as a parameter to the lower level problem and hence the lower level optimal solutions $\mathbf{x}_l^*$ are a function of the upper level vector $\mathbf{x}_u$. The functions $\mathbf{G}(\mathbf{x})$ and $\mathbf{H}(\mathbf{x})$ define the feasibility of a solution at the upper level along with the Pareto-optimality condition to the lower level problem.

4 Progressively Interactive Hybrid Bilevel Evolutionary Multi-objective Optimization Algorithm (PI-HBLEMO)

In this section, the changes made to the Hybrid Bilevel Evolutionary Multi-objective Optimization (HBLEMO) [9] algorithm have been stated. The major change made to the HBLEMO algorithm is in the domination criteria. The other change which has been made is in the termination criteria.

The Progressively Interactive EMO using Value Function (PI-EMO-VF) [10] is a generic procedure which can be integrated with any Evolutionary Multi-objective Optimization (EMO) algorithm. In this section we integrate the procedure at the upper level execution of the HBLEMO algorithm.

After every τ upper level generations of the HBLEMO algorithm, the decision-maker is provided with η (≥ 2) well-sparse non-dominated solutions from the upper level set of non-dominated points. The decision-maker is expected to provide a complete or partial preference information about superiority of one solution over the other, or indifference towards the two solutions. In an ideal situation, the DM can provide a complete ranking (from best to worst) of these solutions, but partial preference information is also allowed. With the given preference information, a strictly increasing polynomial value function is constructed. The value function construction procedure involves solving another single-objective optimization problem. Till the next τ upper level generations, the constructed value function is used to direct the search towards additional preferred solutions.

The termination condition used in the HBLEMO algorithm is based on hypervolume. In the modified PI-HBLEMO algorithm the search is for the most preferred point and not for a pareto optimal front, therefore, the hypervolume based termination criteria can no longer be used. The hypervolume based termination criteria at the upper level has been replaced with a criteria based on distance of an improved solution from the best solutions in the previous generations.

In the following, we specify the steps required to blend the HBLEMO algorithm within the PI-EMO-VF framework and then discuss the termination criteria.

1. Set a counter $t = 0$. Execute the HBLEMO algorithm with the usual definition of dominance [14] at the upper level for τ generations. Increment the value of t by one after each generation.
2. If t is perfectly divisible by τ, then use the k-mean clustering approach ([4, 26]) to choose η diversified points from the non-dominated solutions in the archive; otherwise, proceed to Step 5.
3. Elicitate the preferences of the decision-maker on the chosen η points. Construct a value function $V(\mathbf{f})$, emulating the decision maker preferences, from the information. The value function is constructed by solving an optimization problem (VFOP), described in Section 4.1. If a feasible value function is not found which satisfies all DM's preferences then proceed to Step 5 and use the usual domination principle in HBLEMO operators.

4. Check for termination. The termination check (described in Section 4.2) is based on the distance of the current best solution from the previous best solutions and requires a parameter ϵ_u. If the algorithm terminates, the current best point is chosen as the final solution.
5. An offspring population at the upper level is produced from the parent population at the upper level using a modified domination principle (discussed in Section 4.3) and HBLEMO algorithm's search operators.
6. The parent and the offspring populations are used to create a new parent population for the next generation using the modified domination based on the current value function and other HBLEMO algorithm's operators. The iteration counter is incremented as $t \leftarrow t + 1$ and the algorithm proceeds to Step 2.

The parameters used in the PI-HBLEMO algorithm are τ, η and ϵ_u.

4.1 Step 3: Elicitation of Preference Information and Construction of a Polynomial Value Function

Whenever a DM call is made, a set of η points are presented to the decision maker (DM). The preference information from the decision maker is accepted in the form of pairwise comparisons for each pair in the set of η points. A pairwise comparison of a give pair could lead to three possibilities, the first being that one solution is preferred over the other, the second being that the decision maker is indifferent to both the solutions and the third being that the two solutions are incomparable. Based on such preference information from a decision maker, for a given pair (i, j), if i-th point is preferred over j-th point, then $P_i \succ P_j$, if the decision maker is indifferent to the two solutions then it establishes that $P_i \equiv P_j$. There can be situations such that the decision maker finds a given pair of points as incomparable and in such a case the incomparable points are dropped from the list of η points. If the decision maker is not able to provide preference information for any of the given solution points then algorithm moves back to the previous population where the decision maker was able to take a decisive action, and uses the usual domination instead of modified domination principle to proceed the search process. But such a scenario where no preference is established by a decision maker is rare, and it is likely to have at least one point which is better than another point. Once preference information is available, the task is to construct a polynomial value function which satisfies the preference statements of the decision maker.

Polynomial Value Function for Two Objectives

A polynomial value function is constructed based on the preference information provided by the decision maker. The parameters of the polynomial value function are optimally adjusted such that the preference statements of the decision maker are satisfied. We describe the procedure for two objectives as all the test problems considered in this paper have two objectives. The value function procedure described below is valid for a maximization problem therefore we convert the test problems used in this paper into a maximization problem while implementing the value function procedure. However, while reporting the results for the test problems they are converted back to minimization problems.

$$\begin{aligned} &V(F_1, F_2) = (F_1 + k_1 F_2 + l_1)(F_2 + k_2 F_1 + l_2), \\ &\text{where } F_1, F_2 \quad \text{are the objective values} \\ &\text{and } \quad k_1, k_2, l_1, l_2 \text{ are the value function parameters} \end{aligned} \tag{2}$$

The description of the two objective value function has been taken from [10]. In the above equations it can been seen that the value function V, for two objectives, is represented as a product of two linear functions $S_1 : \mathbf{R}^2 \rightarrow \mathbf{R}$ and $S_2 : \mathbf{R}^2 \rightarrow \mathbf{R}$. [1] The parameters in this

[1] A generalized version of the polynomial value function can be found in [20].

value function which are required to be determined optimally from the preference statements of the decision maker are k_1, k_2, l_1 and l_2. Following is the value function optimization problem (VFOP) which should be solved with the value function parameters (k_1, k_2, l_1 and l_2) as variables. The optimal solution to the VFOP assigns optimal values to the value function parameters. The above problem is a simple single objective optimization problem which can be solved using any single objective optimizer. In this paper the problem has been solved using a sequential quadratic programming (SQP) procedure from the KNITRO [1] software.

$$\begin{aligned}
&\text{Maximize } \epsilon, \\
&\text{subject to } V \text{ is non-negative at every point } P_i, \\
&\qquad V \text{ is strictly increasing at every point } P_i, \\
&\qquad V(P_i) - V(P_j) \geq \epsilon, \quad \text{for all } (i,j) \text{ pairs} \\
&\qquad\qquad \text{satisfying } P_i \succ P_j, \\
&\qquad |V(P_i) - V(P_j)| \leq \delta_V, \quad \text{for all } (i,j) \text{ pairs} \\
&\qquad\qquad \text{satisfying } P_i \equiv P_j.
\end{aligned} \tag{3}$$

The above optimization problem adjusts the value function parameters in such a way that the minimum difference in the value function values for the ordered pairs of points is maximum.

4.2 Termination Criterion

Distance of the current best point is computed from the best points in the previous generations. In the simulations performed, the distance is computed from the current best point to the best points in the previous 10 generations and if each of the computed distances $\delta_u(i), i \in \{1, 2, \ldots, 10\}$ is found to be less than ϵ_u then the algorithm is terminated. A value of $\epsilon_u = 0.1$ has been chosen for the simulations done in this paper.

4.3 Modified Domination Principle

In this sub-section we define the modified domination principle proposed in [10]. The value function V is used to modify the usual domination principle so that more focussed search can be performed in the region of interest to the decision maker. Let $V(F_1, F_2)$ be the value function for a two objective case. The parameters for this value function are optimally determined from the VFOP. For the given η points, the value function assigns a value to each point. Let the values be $V_1, V_2, \ldots, V_\eta$ in the descending order. Now any two feasible solutions ($\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$) can be compared with their objective function values by using the following modified domination criteria:

1. If both points have a value function value *less* than V_2, then the two points are compared based on the usual dominance principle.
2. If both points have a value function value *more* than V_2, then the two points are compared based on the usual dominance principle.
3. If one point has a value function value more than V_2 and the other point has a value function value less than V_2, then the former dominates the latter.

The modified domination principle has been explained through Figure 1 which illustrates regions dominated by two points A and B. Let us consider that the second best point from a given set of η points has a value V_2. The function $V(F) = V_2$ represents a contour which has been shown by a curved line [2]. The first point A has a value V_A which is smaller than V_2 and the region dominated by A is shaded in the figure. The region dominated by A is identical to what can be obtained using the usual domination principle. The second point B has a value V_B

[2] The reason for using the contour corresponding to the second best point can be found in [10]

which is larger than V_2, and, the region dominated by this point is once again shaded. It can be observed that this point no longer follows the usual domination principle. In addition to usual region of dominance this point dominates all the points having a smaller value function value than V_2.

The above modified domination principle can easily be extended for handling constraints as in [5]. When two solutions under consideration for a dominance check are feasible, then the above modified domination principle should be used. If one solution is feasible and the other is infeasible, then the feasible solution is considered as dominating the other. If both the solutions are found to be infeasible then the one with smaller overall feasibility violation (sum of all constraint violations) is considered to be dominating the other solution.

5 Parameter Setting

In the next section, results of the PI-HBLEMO and the HBLEMO procedure on the set of DS test problems ([9]) have been presented. In all simulations, we have used the following parameter values for PI-HBLEMO:

1. Number of points given to the DM for preference information: $\eta = 5$.
2. Number of generations between two consecutive DM calls: $\tau = 5$.
3. Termination parameter: $\epsilon_u = 0.1$.
4. Crossover probability and the distribution index for the SBX operator: $p_c = 0.9$ and $\eta_c = 15$.
5. Mutation probability and the distribution index for polynomial mutation: $p_m = 0.1$ and $\eta_m = 20$.
6. Population size: $N = 40$

6 Results

In this section, results have been presented on a set of 5 DS test problems. All the test problems have two objectives at both the levels. A point, $(F_1^{(b)}, F_2^{(b)})$, on the Pareto-front of the upper level is assumed as the most preferred point and then a DM emulated value function is selected which assigns a maximum value to the most preferred point. The value function selected is $V(F_1, F_2) = \frac{1}{1+(F_1-F_1^{(b)})^2+(F_2-F_2^{(b)})^2}$. It is noteworthy that the value function selected to emulate a decision maker is a simple distance function and therefore has circles as indifference curves which is not a true representative of a rational decision maker. A circular indifference curve may lead to assignment of equal values to a pair of points where one dominates the other. For a pair of points it may also lead assignment of higher value to a point dominated by the other. However, only non-dominated set of points are presented to a decision maker, therefore, such discrepancies are avoided and the chosen value function is able to emulate a decision maker by assigning higher value to the point closest to the most preferred point and lower value to others.

The DS test problems are minimization problems and the progressively interactive procedure using value function works only on problems to be maximized. Therefore, the procedure has been executed by converting the test problems into a maximization problem by putting a negative sign before each of the objectives. However, the final results have once again been converted and the solution to the minimization problem has been presented. The upper level and lower level function evaluations have been reported for each of the test problems. A comparison has been made between the HBLEMO algorithm and PI-HBLEMO procedure in the tables 1, 2, 3, 4 and 5. The tables show the savings in function evaluations which could be achieved moving from an a posteriori approach to a progressively interactive approach. The total lower level function evaluations (Total LL FE) and the total upper level function evaluations (Total UL FE) are presented separately. Table 6 shows the accuracy achieved and the number of DM calls required to get close to the most preferred point. The accuracy is the Euclidean distance of the best point achieved by the algorithm from the most preferred point. The most preferred point has been represented on the Pareto-optimal fronts of the test problems.

6.1 Problem DS1

Problem DS1 has been taken from [9]. A point on the Pareto-optimal front of the test problem is chosen as the most-preferred point and then the PI-HBLEMO algorithm is executed to obtain a solution close to the most preferred point. This problem has $2K$ variables with K real-valued variables each for lower and upper levels. The complete DS1 problem is given below:

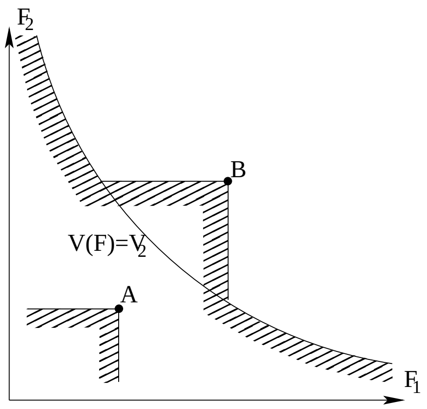

Fig. 1. Dominated regions of two points A and B using the modified definition. Taken from [10].

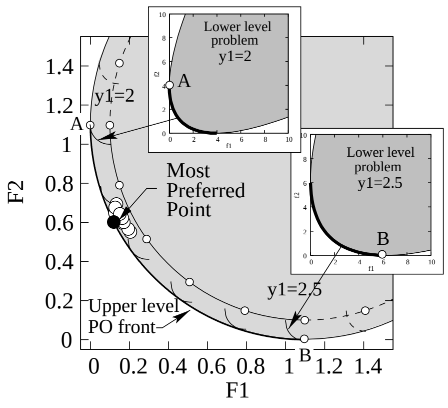

Fig. 2. Pareto-optimal front for problem DS1. Final parent population members have been shown close to the most preferred point.

Figure 2 shows the Pareto-optimal front for the test problem, and the most-preferred solution is marked on the front. The final population members from a particular run of the PI-HBLEMO algorithm are also shown. Table 1 presents the function evaluations required to arrive at the best solution using PI-HBLEMO and also the function evaluations required to achieve an approximated Pareto-frontier using the HBELMO algorithm. The third row in the table presents the ratio of function evaluations using HBLEMO and PI-HBLEMO.

$$
\begin{array}{l}
\text{Minimize} \quad \mathbf{F}(\mathbf{x},\mathbf{y}) = \qquad\qquad \text{subject to} \quad (\mathbf{x}) \in \text{argmin}_{(\mathbf{x})} \mathbf{f}(\mathbf{x}) = \\
\left(\begin{array}{c}
(1+r-\cos(\alpha\pi y_1)) + \sum_{j=2}^{K}(y_j - \frac{j-1}{2})^2 \\
+\tau\sum_{i=2}^{K}(x_i-y_i)^2 - r\cos\left(\gamma\frac{\pi}{2}\frac{x_1}{y_1}\right) \\
(1+r-\sin(\alpha\pi y_1)) + \sum_{j=2}^{K}(y_j - \frac{j-1}{2})^2 \\
+\tau\sum_{i=2}^{K}(x_i-y_i)^2 - r\sin\left(\gamma\frac{\pi}{2}\frac{x_1}{y_1}\right)
\end{array}\right),
\left\{\left(\begin{array}{c}
x_1^2 + \sum_{i=2}^{K}(x_i-y_i)^2 \\
+\sum_{i=2}^{K} 10(1-\cos(\frac{\pi}{K}(x_i-y_i))) \\
\sum_{i=1}^{K}(x_i-y_i)^2 \\
+\sum_{i=2}^{K} 10|\sin(\frac{\pi}{K}(x_i-y_i)|
\end{array}\right)\right\}, \\
-K \le x_i \le K, \quad \text{for } i = 1,\ldots,K, \\
1 \le y_1 \le 4, \quad -K \le y_j \le K, \quad j = 2,\ldots,K.
\end{array}
\tag{4}
$$

For this test problem, $K = 10$ (overall 20 variables), $r = 0.1$, $\alpha = 1$, $\gamma = 1$, and $\tau = 1$ has been used.

Table 1. Total function evaluations for the upper and lower level (21 runs) for DS1.

Algo. 1, Algo. 2, Savings	Best		Median		Worst	
	Total LL FE	Total UL FE	Total LL FE	Total UL FE	Total LL FE	Total UL FE
HBLEMO	2,819,770	87,582	3,423,544	91,852	3,829,812	107,659
PI-HBLEMO	329,412	12,509	383,720	12,791	430,273	10,907
$\frac{HBLEMO}{PI-HBLEMO}$	8.56	7.00	8.92	7.18	8.90	9.87

6.2 Problem DS2

Problem DS2 has been taken from [9]. A point on the Pareto-optimal front of the test problem is chosen as the most-preferred point and then the PI-HBLEMO algorithm is executed to obtain a solution close to the most preferred point. This problem uses discrete values of y_1 to determine the upper level Pareto-optimal front. The overall problem is given as follows:

$$\begin{aligned} u_1(y_1) &= \begin{cases} \cos(0.2\pi)y_1 + \sin(0.2\pi)\sqrt{|0.02\sin(5\pi y_1)|}, \\ \quad \text{for } 0 \le y_1 \le 1, \\ y_1 - (1-\cos(0.2\pi)), \quad y_1 > 1 \end{cases} \\ u_2(y_1) &= \begin{cases} -\sin(0.2\pi)y_1 + \cos(0.2\pi)\sqrt{|0.02\sin(5\pi y_1)|}, \\ \quad \text{for } 0 \le y_1 \le 1, \\ 0.1(y_1 - 1) - \sin(0.2\pi), \quad \text{for } y_1 > 1. \end{cases} \end{aligned} \tag{5}$$

$$\begin{aligned} &\text{Minimize} \quad \mathbf{F}(\mathbf{x},\mathbf{y}) = \\ &\begin{pmatrix} u_1(y_1) + \sum_{j=2}^{K}\left[y_j^2 + 10(1-\cos(\frac{\pi}{K}y_i))\right] \\ +\tau\sum_{i=2}^{K}(x_i - y_i)^2 - r\cos\left(\gamma\frac{\pi}{2}\frac{x_1}{y_1}\right) \\ u_2(y_1) + \sum_{j=2}^{K}\left[y_j^2 + 10(1-\cos(\frac{\pi}{K}y_i))\right] \\ +\tau\sum_{i=2}^{K}(x_i - y_i)^2 - r\sin\left(\gamma\frac{\pi}{2}\frac{x_1}{y_1}\right) \end{pmatrix}, \ \text{subject to} \quad (\mathbf{x}) \in \mathbf{f}(\mathbf{x}) = \text{argmin}_{(\mathbf{x})}\left\{\begin{pmatrix} x_1^2 + \sum_{i=2}^{K}(x_i - y_i)^2 \\ \sum_{i=1}^{K} i(x_i - y_i)^2 \end{pmatrix}\right\}, \\ &-K \le x_i \le K, \quad i = 1,\ldots,K, \\ &0.001 \le y_1 \le K, \quad -K \le y_j \le K,\ j = 2,\ldots,K, \end{aligned} \tag{6}$$

Due to the use of periodic terms in u_1 and u_2 functions, the upper level Pareto-optimal front corresponds to only six discrete values of y_1 (=0.001, 0.2, 0.4, 0.6, 0.8 and 1). $r = 0.25$ has been used.

In this test problem the upper level problem has multi-modalities, thereby causing an algorithm difficulty in finding the upper level Pareto-optimal front. A value of $\tau = -1$ has been used, which introduces a conflict between upper and lower level problems. The results have been produced for 20 variables test problem.

The Pareto-optimal front for this test problem is shown in Figure 3. The most-preferred solution is marked on the Pareto-front along with the final population members obtained from a particular run of the PI-HBLEMO algorithm. Table 2 presents the function evaluations required to arrive at the best solution using PI-HBLEMO. The function evaluations required to achieve an approximated Pareto-frontier using the HBELMO algorithm is also reported.

6.3 Problem DS3

Problem DS3 has been taken from [9]. A point on the Pareto-optimal front of the test problem is chosen as the most-preferred point and then the PI-HBLEMO algorithm is executed to obtain a solution close to the most preferred point. In this test problem, the variable y_1 is considered

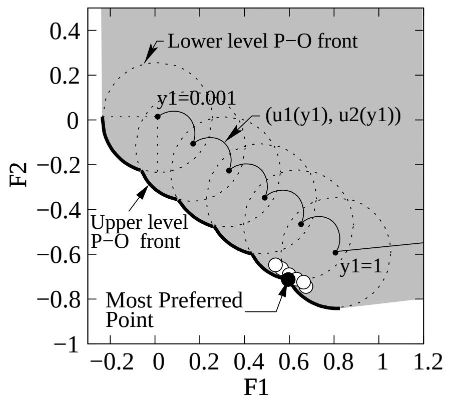

Fig. 3. Pareto-optimal front for problem DS2. Final parent population members have been shown close to the most preferred point.

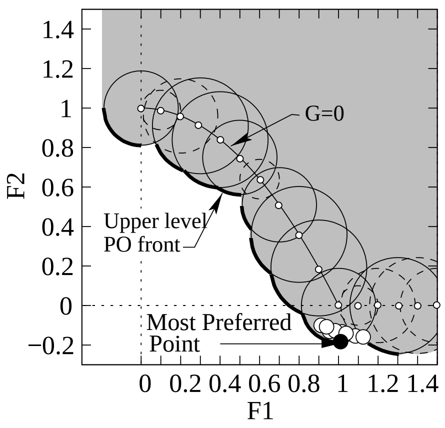

Fig. 4. Pareto-optimal front for problem DS3. Final parent population members have been shown close to the most preferred point.

Table 2. Total function evaluations for the upper and lower level (21 runs) for DS2.

Algo. 1, Algo. 2, Savings	Best		Median		Worst	
	Total LL FE	Total UL FE	Total LL FE	Total UL FE	Total LL FE	Total UL FE
HBLEMO	4,796,131	112,563	4,958,593	122,413	5,731,016	144,428
PI-HBLEMO	509,681	14,785	640,857	14,535	811,588	15,967
$\frac{HBLEMO}{PI-HBLEMO}$	9.41	7.61	7.74	8.42	7.06	9.05

to be discrete, thereby causing only a few y_1 values to represent the upper level Pareto-optimal front. The overall problem is given below:

$$\begin{aligned}
&\text{Minimize} \quad \mathbf{F}(\mathbf{x},\mathbf{y}) = \\
&\begin{pmatrix} y_1 + \sum_{j=3}^{K}(y_j - j/2)^2 + \tau \sum_{i=3}^{K}(x_i - y_i)^2 - R(y_1)\cos(4\tan^{-1}\left(\frac{y_2-x_2}{y_1-x_1}\right) \\ y_2 + \sum_{j=3}^{K}(y_j - j/2)^2 + \tau \sum_{i=3}^{K}(x_i - y_i)^2 - R(y_1)\sin(4\tan^{-1}\left(\frac{y_2-x_2}{y_1-x_1}\right) \end{pmatrix}, \\
&\text{subject to} \quad (\mathbf{x}) \in \text{argmin}_{(\mathbf{x})} \\
&\quad \left\{ \mathbf{f}(\mathbf{x}) = \begin{pmatrix} x_1 + \sum_{i=3}^{K}(x_i - y_i)^2 \\ x_2 + \sum_{i=3}^{K}(x_i - y_i)^2 \end{pmatrix} \middle| g_1(\mathbf{x}) = (x_1 - y_1)^2 + (x_2 - y_2)^2 \le r^2 \right\}, \\
&G(\mathbf{y}) = y_2 - (1 - y_1^2) \ge 0, \\
&-K \le x_i \le K, \quad \text{for } i = 1,\dots,K, \quad 0 \le y_j \le K, \quad \text{for } j = 1,\dots,K, \\
&y_1 \text{ is a multiple of 0.1.}
\end{aligned} \tag{7}$$

Here a periodically changing radius has been used: $R(y_1) = 0.1 + 0.15|\sin(2\pi(y_1 - 0.1)|$ and use $r = 0.2$. For the upper level Pareto-optimal points, $y_i = j/2$ for $j \le 3$. The variables y_1 and y_2 take values satisfying constraint $G(\mathbf{y}) = 0$. For each such combination, variables x_1 and x_2 lie on the third quadrant of a circle of radius r and center at (y_1, y_2) in the $\mathbf{F}$-space. For this test problem, the Pareto-optimal fronts for both lower and upper level lie on constraint boundaries, thereby requiring good constraint handling strategies to solve the problem adequately. $\tau = 1$ has been used for this test problem with 20 number of variables.

Table 3. Total function evaluations for the upper and lower level (21 runs) for DS3.

Algo. 1, Algo. 2, Savings	Best		Median		Worst	
	Total LL FE	Total UL FE	Total LL FE	Total UL FE	Total LL FE	Total UL FE
HBLEMO	3,970,411	112,560	4,725,596	118,848	5,265,074	125,438
PI-HBLEMO	475,600	11,412	595,609	16,693	759,040	16,637
$\frac{HBLEMO}{PI-HBLEMO}$	8.35	9.86	7.93	7.12	6.94	7.54

The Pareto-front, most-preferred point and the final population members from a particular run are shown in Figure 4. Table 3 presents the function evaluations required by PI-HBLEMO to produce the final solution and the function evaluations required by HBELMO to produce an approximate Pareto-front.

6.4 Problem DS4

Problem DS4 has been taken from [9]. A point on the Pareto-optimal front of the test problem is chosen as the most-preferred point and then the PI-HBLEMO algorithm is executed to obtain a solution close to the most preferred point. This problem has $K + L + 1$ variables, which are all real-valued:

$$\begin{array}{ll} \text{Minimize} \quad \mathbf{F}(\mathbf{x}, \mathbf{y}) = & \text{subject to} \quad (\mathbf{x}) \in \text{argmin}_{(\mathbf{x})} \mathbf{f}(\mathbf{x}) = \\ \begin{pmatrix} (1 - x_1)(1 + \sum_{j=2}^{K} x_j^2) y_1 \\ x_1 (1 + \sum_{j=2}^{K} x_j^2) y_1 \end{pmatrix}, & \left\{ \begin{pmatrix} (1 - x_1)(1 + \sum_{j=K+1}^{K+L} x_j^2) y_1 \\ x_1 (1 + \sum_{j=K+1}^{K+L} x_j^2) y_1 \end{pmatrix} \right\}, \\ G_1(\mathbf{x}) = (1 - x_1) y_1 + \frac{1}{2} x_1 y_1 - 1 \geq 0, & \\ -1 \leq x_1 \leq 1, \quad 1 \leq y_1 \leq 2, & \\ -(K + L) \leq x_i \leq (K + L), \ i = 2, \ldots, (K + L). & \end{array} \tag{8}$$

The upper level Pareto-optimal front is formed with $x_i = 0$ for all $i = 2, \ldots, (K + L)$ and $x_1 = 2(1-1/y_1)$ and $y_1 \in [1, 2]$. By increasing K and L, the problem complexity in converging to the appropriate lower and upper level fronts can be increased. Only one Pareto-optimal point from each participating lower level problem qualifies to be on the upper level front. For our study here, we choose $K = 5$ and $L = 4$ (an overall 10-variable problem).

Table 4. Total function evaluations for the upper and lower level (21 runs) for DS4.

Algo. 1, Algo. 2, Savings	Best		Median		Worst	
	Total LL FE	Total UL FE	Total LL FE	Total UL FE	Total LL FE	Total UL FE
HBLEMO	1,356,598	38,127	1,435,344	53,548	1,675,422	59,047
PI-HBLEMO	149,214	5,038	161,463	8,123	199,880	8,712
$\frac{HBLEMO}{PI-HBLEMO}$	9.09	7.57	8.89	6.59	8.38	6.78

The Pareto-front, most-preferred point and the final population members from a particular run are shown in Figure 5. Table 4 presents the function evaluations required by PI-HBLEMO to produce the final solution and the function evaluations required by HBELMO to produce an approximate Pareto-front.

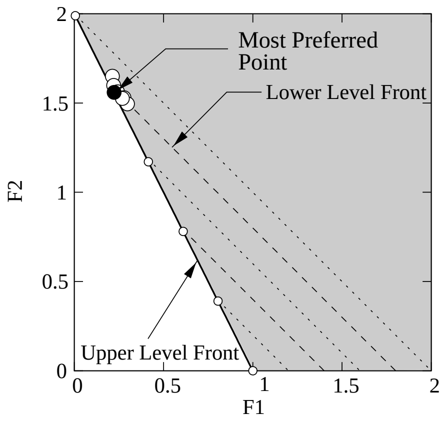

Fig. 5. Pareto-optimal front for problem DS4. Final parent population members have been shown close to the most preferred point.

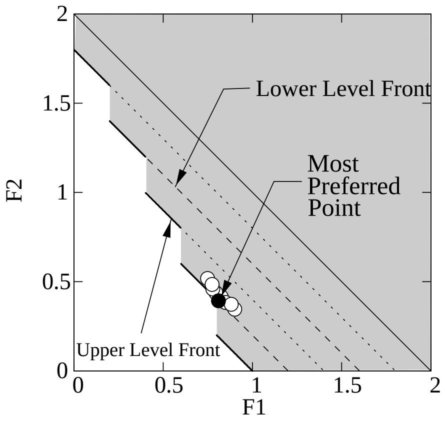

Fig. 6. Pareto-optimal front for problem DS5. Final parent population members have been shown close to the most preferred point.

6.5 Problem DS5

Problem DS5 has been taken from [9]. A point on the Pareto-optimal front of the test problem is chosen as the most-preferred point and then the PI-HBLEMO algorithm is executed to obtain a solution close to the most preferred point. This problem is similar to problem DS4 except that the upper level Pareto-optimal front is constructed from multiple points from a few lower level Pareto-optimal fronts. There are $K + L + 1$ real-valued variables in this problem as well:

$$\begin{array}{ll}
\text{Minimize} \quad \mathbf{F}(\mathbf{x},\mathbf{y}) = & \text{subject to} \quad (\mathbf{x}) \in \text{argmin}_{(\mathbf{x})} \mathbf{f}(\mathbf{x}) = \\
\begin{pmatrix} (1-x_1)(1+\sum_{j=2}^{K} x_j^2)y_1 \\ x_1(1+\sum_{j=2}^{K} x_j^2)y_1 \end{pmatrix}, & \left\{ \begin{pmatrix} (1-x_1)(1+\sum_{j=K+1}^{K+L} x_j^2)y_1 \\ x_1(1+\sum_{j=K+1}^{K+L} x_j^2)y_1 \end{pmatrix} \right\}, \\
G_1(\mathbf{x}) = (1-x_1)y_1 + \frac{1}{2}x_1y_1 - 2 + \frac{1}{5}\left[5(1-x_1)y_1 + 0.2\right] \geq 0, & \\
[\cdot] \text{ denotes greatest integer function,} & \\
-1 \leq x_1 \leq 1, \quad 1 \leq y_1 \leq 2, & \\
-(K+L) \leq x_i \leq (K+L), \; i = 2, \ldots, (K+L). &
\end{array} \tag{9}$$

For the upper level Pareto-optimal front, $x_i = 0$ for $i = 2, \ldots, (K+L)$, $x_1 \in [2(1-1/y_1), 2(1-0.9/y_1)]$, $y_1 \in \{1, 1.2, 1.4, 1.6, 1.8\}$ (Figure 6). For this test problem we have chosen $K = 5$ and $L = 4$ (an overall 10-variable problem). This problem has similar difficulties as in DS4, except that only a finite number of y_1 qualifies at the upper level Pareto-optimal front and that a consecutive set of lower level Pareto-optimal solutions now qualify to be on the upper level Pareto-optimal front.

The Pareto-front, most-preferred point and the final population members from a particular run are shown in Figure 6. Table 5 presents the function evaluations required by PI-HBLEMO to produce the final solution and the function evaluations required by HBELMO to produce an approximate Pareto-front.

7 Accuracy and DM calls

Table 6 represents the accuracy achieved and the number of decision maker calls required while using the PI-HBLEMO procedure. In the above test problems the most preferred point which

Table 5. Total function evaluations for the upper and lower level (21 runs) for DS5.

Algo. 1, Algo. 2, Savings	Best		Median		Worst	
	Total LL FE	Total UL FE	Total LL FE	Total UL FE	Total LL FE	Total UL FE
HBLEMO	1,666,953	47,127	1,791,511	56,725	2,197,470	71,246
PI-HBLEMO	168,670	5,105	279,568	6,269	304,243	9,114
$\frac{HBLEMO}{PI-HBLEMO}$	9.88	9.23	6.41	9.05	7.22	7.82

the algorithm is seeking is pre-decided and the value function emulating the decision maker is constructed. When the algorithm terminates it provides the best achieved point. The accuracy measure is the Euclidean distance between the best point achieved and the most preferred point. It can be observed from the results of the PI-HBLEMO procedure that preference information from the decision maker leads to a high accuracy (Table 6) as well as huge savings (Table 1,2,3,4,5) in function evaluations. Producing the entire front using the HBLEMO procedure has its own merits but it comes with a cost of huge function evaluations and there can be instances when the entire set of close Pareto-optimal solutions will be difficult to achieve even after high number of evaluations. The accuracy achieved using the HBLEMO procedure has been reported in the brackets; the final choice made from a set of close Pareto-optimal solutions will lead to a poorer accuracy than a progressively interactive approach.

Table 6. Accuracy and the number of decision maker calls for the PI-HBLEMO runs (21 runs). The distance of the closest point to the most preferred point achieved from the HBLEMO algorithm has been provided in the brackets.

		Best	Median	Worst
DS1	Accuracy	0.0426 (0.1203)	0.0888 (0.2788)	0.1188 (0.4162)
	DM Calls	12	13	29
DS2	Accuracy	0.0281 (0.0729)	0.0804 (0.4289)	0.1405 (0.7997)
	DM Calls	12	15	25
DS3	Accuracy	0.0498 (0.0968)	0.0918 (0.3169)	0.1789 (0.6609)
	DM Calls	7	17	22
DS4	Accuracy	0.0282 (0.0621)	0.0968 (0.0981)	0.1992 (0.5667)
	DM Calls	7	15	23
DS5	Accuracy	0.0233 (0.1023)	0.0994 (0.1877)	0.1946 (0.8946)
	DM Calls	7	14	22

8 Conclusions

There are not many approaches yet to handle multi-objective bilevel problems. The complexity involved in solving a bilevel multi-objective problem has deterred researchers, keeping the area unexplored. The Hybrid Bilevel Evolutionary Multi-objective Optimization Algorithm is one of the successful procedures towards handling a general bilevel multi-objective problem. However, the procedure involves heavy computation, particularly at the lower level, to produce the entire Pareto-optimal set of solutions at the upper level.

In this paper, the Hybrid Bilevel Evolutionary Multi-objective Optimization Algorithm has been blended with a progressively interactive technique. An evaluation of the Progressively Interactive HBLEMO (PI-HBLEMO) technique against the HBLEMO procedure shows an improvement in terms of function evaluations as well as accuracy. The savings in function evaluations at the lower as well as upper level is in the range of six to ten times. This is a significant improvement particularly for cases where a function evaluation is computationally very expensive. Moreover, for problems where EMOs tend to suffer in converging towards the front, a progressively interactive approach provides a viable solution to such problems and leads to a higher accuracy. Therefore, an integration of the progressively interactive procedure with an EMO algorithm offers a dual advantage of reduced function evaluations and increased accuracy. Such kind of an integrated procedure, with EMO algorithm's parallel search and concepts from the field of MCDM, can be a useful tool to efficiently handle difficult multi-objective problems. The power of such an amalgamation has been shown in this study with its successful application on the challenging domain of multi-objective bilevel problems.

9 Acknowledgements

The author would like to thank Prof. Kalyanmoy Deb, Prof. Pekka Korhonen and Prof. Jyrki Wallenius for their useful feedback and discussions which led to this work.

Bibliography

[1] R. H. Byrd, J. Nocedal, and R. A. Waltz. *KNITRO: An integrated package for nonlinear optimization*, pages 35–59. Springer-Verlag, 2006.

[2] P. H. Calamai and L. N. Vicente. Generating quadratic bilevel programming test problems. *ACM Trans. Math. Software*, 20(1):103–119, 1994.

[3] B. Colson, P. Marcotte, and G. Savard. An overview of bilevel optimization. *Annals of Operational Research*, 153:235–256, 2007.

[4] K. Deb. *Multi-objective optimization using evolutionary algorithms*. Chichester, UK: Wiley, 2001.

[5] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[6] K. Deb and A. Sinha. Constructing test problems for bilevel evolutionary multi-objective optimization. In *2009 IEEE Congress on Evolutionary Computation (CEC-2009)*, pages 1153–1160. IEEE Press, 2009.

[7] K. Deb and A. Sinha. An evolutionary approach for bilevel multi-objective problems. In *Cutting-Edge Research Topics on Multiple Criteria Decision Making, Communications in Computer and Information Science*, volume 35, pages 17–24. Berlin, Germany: Springer, 2009.

[8] K. Deb and A. Sinha. Solving bilevel multi-objective optimization problems using evolutionary algorithms. In *Evolutionary Multi-Criterion Optimization (EMO-2009)*, pages 110–124. Berlin, Germany: Springer-Verlag, 2009.

[9] K. Deb and A. Sinha. An efficient and accurate solution methodology for bilevel multi-objective programming problems using a hybrid evolutionary-local-search algorithm. *Evolutionary Computation Journal*, 18(3):403–449, 2010.

[10] K. Deb, A. Sinha, P. Korhonen, and J. Wallenius. An interactive evolutionary multi-objective optimization method based on progressively approximated value functions. *IEEE Transactions on Evolutionary Computation*, 14(5):723–739, 2010.

[11] S. Dempe, J. Dutta, and S. Lohse. Optimality conditions for bilevel programming problems. *Optimization*, 55(5âĂŞ6):505âĂŞ–524, 2006.

[12] G. Eichfelder. Solving nonlinear multiobjective bilevel optimization problems with coupled upper level constraints. Technical Report Preprint No. 320, Preprint-Series of the Institute of Applied Mathematics, Univ. Erlangen-NÃijrnberg, Germany, 2007.

[13] D. Fudenberg and J. Tirole. *Game theory*. MIT Press, 1993.

[14] A. M. Geoffrion. Proper efficiency and theory of vector maximization. *Journal of Mathematical Analysis and Applications*, 22(3):618–630, 1968.

[15] W. Halter and S. Mostaghim. Bilevel optimization of multi-component chemical systems using particle swarm optimization. In *Proceedings of World Congress on Computational Intelligence (WCCI-2006)*, pages 1240–1247, 2006.

[16] A. Koh. Solving transportation bi-level programs with differential evolution. In *2007 IEEE Congress on Evolutionary Computation (CEC-2007)*, pages 2243–2250. IEEE Press, 2007.

[17] R. Mathieu, L. Pittard, and G. Anandalingam. Genetic algorithm based approach to bi-level linear programming. *Operations Research*, 28(1):1–21, 1994.

[18] V. Oduguwa and R. Roy. Bi-level optimization using genetic algorithm. In *Proceedings of the 2002 IEEE International Conference on Artificial Intelligence Systems (ICAISâĂŹ02)*, pages 322–327, 2002.

Aalto-DD 17/2011

9 789526 040523

ISBN: 978-952-60-4053-0 (pdf)
ISBN: 978-952-60-4052-3
ISSN-L: 1799-4934
ISSN: 1799-4942 (pdf)
ISSN: 1799-4934

Aalto University
School of Economics
Department of Business Technology
aalto.fi