On The Use of Informed Initialization and Extreme Solutions Sub-population in Multiobjective Evolutionary Algorithms

Dipankar Dasgupta*, German Hernandez†, Andres Romero†, Deon Garrett*, Aishwarya Kaushal*, James Simien‡

* Department of Computer Science
University of Memphis
Memphis, TN, USA
Email: dasgupta@cs.memphis.edu, deong@acm.org, xxxxx@memphis.edu

† Department of Computer Science
National University of Columbia
Bogota, Columbia
Email: xxxxxxx@xxxxx.xxx, xxxxxxx@xxxxx.xxx

‡ Navy Personnel Research, Studies, and Technology
Millington, TN, USA
Email: james.simien@navy.mil

Abstract—This paper examines two strategies in order to improve the performance of multi-objective evolutionary algorithms when applied to problems with many objectives: informed initialization and extreme solutions sub-population. The informed initialization is the inclusion of approximations of extreme and internal points of the Pareto front in the initial population. These approximations, called informed initial solutions, are found using a high quality evolutionary or local search algorithm on single objective problems obtained by scalarizing the multiple goals into a single goal by the use of weight vectors. The extreme solutions sub-population is proposed here to keep the best approximations of the extreme points of the Pareto front at any point of the evolution, and the selection scheme is biased to give these solutions slightly higher chances of being selected. Experimental results applying these two strategies in continuous and combinatorial benchmark problems show that the diversity in the final solutions is improved, while preserving the proximity to the Pareto front. Some additional experiments that demonstrate how the number of initial informed solutions affects the performance are also presented.

I. INTRODUCTION

Over the past ten years, a large amount of research has been conducted on the use of evolutionary techniques to solve multiobjective optimization problems. The inherent use of a population of candidate solutions provides benefits unattained by other techniques when applied to problems with two or three objectives. In recent years some attention has been devoted to the experimental study of the performance of these evolutionary techniques in many objective problems. These are optimization problems with more than three conflicting objectives. The experimental studies conducted thus lead to the conclusion that the performance of the evolutionary techniques is severely hindered when the dimension of the objective space grows. A recent study [3] shows that for problems with more than ten objectives a purely random search may perform favorably when compared with an evolutionary technique.

It is well-known that for single objective problems the performance of evolutionary algorithms can often be improved through the introduction of a local search operator. These combinations of evolutionary and local search are known by the names of Memetic Algorithms (MAs) [24], Hybrid Evolutionary Algorithms [10], [11], [17], and Genetic Local Search (GLS) [23]. These hybrids of evolutionary and local search algorithms have been shown to provide state-of-the-art performance on a wide range of hard single objective combinatorial optimization problems [23], [21], [25] and also have shown to provide a good performance on multiobjective optimization problems [15], [16], [20], [17], [19], [22].

This paper proposes two modifications to improve the performance of multiobjective evolutionary algorithms through the use of a particular form of hybridization. A high performance single objective search algorithm is used to seed the initial population with solutions to a scalarized version of the problem. In this work, we consider two such search methods: a linear programming algorithm for the Sailor Assignment Problem as well as a high performance evolutionary algorithm for general optimization. In addition, these initial solutions are maintained in an external population throughout the evolutionary process and given additional mating chances. We show that these modifications provided significant performance improvements over conventional multiobjective evolutionary algorithms.

This paper is organized as follows: Section II introduces the MOEAs used in this work, NSGA-II and SPEA2, and gives a brief overview of hybrid algorithms that combine MOEAs and other search strategies; Section III describes the benchmark problems used in this study and introduces the concept of informed initialization and extreme solutions sub-population.
Sections IV and V present the experiments and results used in this study, and Section VI briefly discusses the conclusions and future work.

II. MOEAS AND HYBRIDIZED ALGORITHMS

MOEAs have two primary goals: to find "good" solutions (near the true Pareto optimal front), and to maintain diverse alternatives along the approximation to the Pareto front. Both NSGA-II and SPEA2, the algorithms considered in this work, achieve these goals by assigning fitness in such a way as to bias selection toward non-dominated solutions that do not resemble other known solutions.

A. NSGA-II

The Non-Dominated Sorting Genetic Algorithm II (NSGA-II) [5] uses the concept of a domination level to assign fitness. All non-dominated individuals are assigned level zero. Upon removing those individuals from the population, the newly non-dominated solutions are assigned level one. The process continues until all solutions have been assigned a non-domination level. Compared to SPEA, NSGA is able to maintain a better spread of solutions and convergence in the obtained non-dominated Pareto front. Like SPEA2, NSGA-II then employs a density estimation technique to estimate which solutions may be removed with the least impact on diversity.

B. SPEA2

SPEA2, the Strength Pareto Evolutionary Algorithm, was first published by Zitzler et al. in 2001 [28]. SPEA2 utilizes a concept called strength, a concept related to the number of other individuals a given solution dominates. The fitness of a dominated individual is computed based on the strengths of all other individuals that dominate it. The algorithm ensures that all non-dominated solutions have lower fitness values than any dominated solution, where minimization is the goal of the algorithm, thereby biasing selection toward non-dominated solutions. In addition, the distance to a solutions nearest neighbors is used to apply additional pressure toward individuals in sparsely populated regions of the Pareto front.

C. Hybrid Algorithms

Despite their popularity, pure evolutionary approaches often have some difficulty in finding truly optimal solutions, particularly in combinatorial optimization problems. The solutions obtained by these MOEAs are often clustered in a region of relatively low quality. Inclusion of a local search routine can help push solutions to improve both in their proximity to the Pareto front as well as improving the diversity of solutions maintained by the algorithm. In this work, we consider a linear programming algorithm as well as a high performance evolutionary algorithm, CHC, to generate the locally improved solutions that will participate in the evolutionary process.

Like any hybrid algorithm, there are a number of choices for how to integrate the different search algorithms into a single method. In this work, we describe a two-part process by which CHC is used to generate a small number of initial solutions to the problem, which are then used to seed the initial population of a conventional multiobjective evolutionary algorithm. Then during the run of the MOEA, these "informed" solutions are maintained and given additional opportunities to participate in the evolutionary process. This process consists of adding to the initial population of a MOEA some individuals which are very good approximations to 1) the extreme solutions of the Pareto front and 2) some few internal points on the Pareto front trying to guide the remaining individuals in the population towards the Pareto front.

CHC [9] is an evolutionary algorithm combining strongly elitist environment selection driving the population toward uniformity with a novel restart mechanism allowing the algorithm to successfully continue searching after it has converged. In addition, CHC utilizes a crossover operator designed to ensure that offspring are maximally distant from their parents. This combination of features has led to CHC performing very well on a wide area of numeric and combinatorial optimization problems. Algorithm 1 below briefly describes the steps of CHC.

Algorithm 1 CHC Algorithm

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$t ← 0$</td>
</tr>
<tr>
<td>2</td>
<td>Initialize ($Pa$, $ConvergenceCount$, $k$) { $Pa$: Population}</td>
</tr>
<tr>
<td>3</td>
<td>while not $EndingCondition (t, Pa)$ do</td>
</tr>
<tr>
<td>4</td>
<td>$Parents ← SelectionParents(Pa, ConvergenceCount)$</td>
</tr>
<tr>
<td>5</td>
<td>$Offspring ← HUX(Parents)$</td>
</tr>
<tr>
<td>6</td>
<td>Evaluate($Offspring$)</td>
</tr>
<tr>
<td>7</td>
<td>$Pn ← ElitistSelection (Offspring, Pa) { $Pn$: new population}$</td>
</tr>
<tr>
<td>8</td>
<td>if not modified ($Pa$, $Pn$) then</td>
</tr>
<tr>
<td>9</td>
<td>$ConvergenceCount ← ConvergenceCount − 1$</td>
</tr>
<tr>
<td>10</td>
<td>if $ConvergenceCount ≤ −k$ then</td>
</tr>
<tr>
<td>11</td>
<td>$Pn ← Restart(Pa)$</td>
</tr>
<tr>
<td>12</td>
<td>Initialize $ConvergenceCount$</td>
</tr>
<tr>
<td>13</td>
<td>end if</td>
</tr>
<tr>
<td>14</td>
<td>end if</td>
</tr>
<tr>
<td>15</td>
<td>$t ← t + 1$</td>
</tr>
<tr>
<td>16</td>
<td>$Pa ← Pn$</td>
</tr>
<tr>
<td>17</td>
<td>end while</td>
</tr>
</tbody>
</table>

In order to generate such initial solutions, the multiobjective problem is scalarized using particular weight vectors and solved using the CHC single objective genetic algorithm. The weight vectors are chosen so that each objective is considered in isolation, yield points at every extreme of the Pareto front. In addition, some number of additional points are chosen to push toward the interior regions of the front. The weight vectors for these interior points may be chosen randomly or systematically, e.g., each objective equally weighted. With these two types of vectors, it is possible to generate good approximations to be inserted into the initial population of NSGA-II; these approximations are generated by running the CHC algorithm using each one of the weight vectors.

The solutions generated using the CHC algorithm with the weight vectors are kept in a sub-population located in the main
The QAP problem can be stated as: A set of
\( n \) facilities or departments and a set of \( n \) locations are given. For each pair of facilities weight or flow is given; and for locations, distance is given. The goal is to assign all facilities to different locations. It is fundamental and very popular combinatorial optimization problem. The QAP can be described as follows: Given two \( n \times n \) matrices \( A \) and \( B \), find a permutation \( \pi \) that minimizes
\[
\min_{\pi} F(\pi) = \sum_{i=1}^{n} \sum_{j=1}^{n} A_{i,j} B_{\pi_i,\pi_j}. 
\]

Conventionally, the matrices \( A \) and \( B \) are called the distance and flow matrices, the terminology arising from the original formulation of QAP as a facilities layout problem. In a facility layout problem there are \( n \) number of departments or facilities to be assigned to \( n \) number of locations. It is because that sub-population contains very good approximations to the pareto front which must be preserved. Of course those solutions are not perfect and can be replaced from the sub-populations if in the remaining of the populations appears a better solution; that is, a solution that dominates one of the solutions in the sub-population will replace it.

The NSGA-II selection scheme was then modified to allow the solutions from the informed sub-population to recombine more often. The standard tournament selection procedure takes two solutions and first checks for dominance. If one of the solutions dominates the other, then that solution is selected; if the solutions are non-dominated, then the crowding distance is used to compare and select one of them. The modification proposed here adds an intermediate step, if the solutions are non-dominated then the one that is generated from the informed sub-population is selected. If no solution comes from the informed sub-population or if more than one solution is generated, then the crowding distance should be checked.

III. BENCHMARK PROBLEMS

A. DTLZ3

In [6], Deb et. al. introduced several scalable multiobjective test problems. Among the most challenging, DTLZ3 contains an exponential number of false Pareto fronts which can trap an optimization algorithm. By scalable, it is meant that the problems may be defined with arbitrary numbers of objectives and decision variables without losing their basic character. For the purposes of this study, we chose DTLZ3 with both six and 12 objectives, using the recommendations of [6] for all other parameters.

B. QAP

The Quadratic Assignment Problem (QAP) is one of the fundamental and very popular combinatorial optimization problems. It was first introduced by Koopmans and Beckmann in 1957 [2] to model the problem of locating the departments with material flow between them.

The QAP problem can be stated as: A set of \( n \) facilities or departments and a set of \( n \) locations are given. For each pair of facilities weight or flow is given; and for locations, distance is given. The goal is to assign all facilities to different locations in such a way that the sum of the distances multiplied by the corresponding flows is minimized. The QAP can be described as follows: Given two \( n \times n \) matrices \( A \) and \( B \), find a permutation \( \pi \) that minimizes
\[
\min_{\pi} F(\pi) = \sum_{i=1}^{n} \sum_{j=1}^{n} A_{i,j} B_{\pi_i,\pi_j}. 
\]

The mQAP models any sort of facilities layout problem in which the minimization of multiple simultaneous flows is required.

C. SAP

According to the United States Navy’s personnel policies, roughly every three years sailors serving on active duty are reassigned to a different job. As a result, at any given time there exists a sizable population of sailors to be reassigned to available jobs. Currently, more than 300,000 sailors serve in the Navy and more than 120,000 are reassigned each year [14]. Sailors must be reassigned in such a way as to satisfy their individual preferences and the needs of the Navy. Many factors go into determining a good set of assignments. The Navy must ensure that not only are the sailors and commanders satisfied with the assignment, but also the match must be implemented within a fixed budget. If the resulting cost is too expensive, then the Navy cannot adequately maintain its other priorities. However, if the assignment is overly focused on costs, it may be that the sailors are unhappy with their assigned jobs. This could lead to a decrease in morale, followed by decreased retention rates and other problems. Thus, the Navy’s goal is to identify sailor and job matches that maximize some overall criterion of desirability and is referred to as the Sailor Assignment Problem (SAP). In particular, the SAP involves...
finding a balanced range of solutions that maximize the total training score, the sailor preference score, and the commander preference score, and minimize the PCS cost.

In [13], genetic algorithms were tested on single objective versions of the problem and compared against results of the Gale-Shapley algorithm that solves in $O(n^2)$ time the stable marriage problem. In [12] it was shown that when NSGA-II [5] or SPEA2 [28] are applied to multiobjective instances of the sailor assignment problem (discussed later), it results in solution that lack adequate diversity and that combining the evolutionary algorithms with a rudimentary local search operator, i.e. an hybrid algorithm, achieves greatly improved diversity.

In [4] a different and more efficient hybrid algorithm was introduced for SAP combining NSGA-II with the Kuhn-Munkres algorithm. The Kuhn-Munkres algorithm solves linear assignment problems in $O(n^3)$ time and it was chosen due to fact that the single objective versions of SAP with small modifications are linear assignment problems. Because the Kuhn-Munkres algorithm does fully solve the SAP, one option is to simply apply the algorithm to a wide range of scalarizations of the problem yielding a good approximation to the true Pareto front. This approach, while yielding very good solutions, suffers from very long run times as the problem sizes are increased. Thus, one useful goal is to use metaheuristics to reduce the computation time required, while maintaining a reasonably good approximation to the front as found by the linear programming algorithm.

IV. EXPERIMENTS

In order to validate the performance of the proposed approach, experiments were carried out utilizing the jMetal platform [8] and using the scalable multiobjective test problems defined in [7]; specifically, the DTLZ3 problem has been tested in 6 and 12 dimensions. In addition, the approach was tested on the SAP with the linear programming initialization as well as the QAP using CHC for the local improvement. To measure the performance of the algorithms, the coverage and proximity metrics [1] were used. The metrics are briefly described below.

A. Proximity

The distance between the approximation set $S$ and the pareto front $P_F$ is defined as the average of the minimum distance between a solution and the pareto optimal front over each solution in $S$, as in [1]:

$$D_{S \rightarrow P_F}(S) = \frac{1}{|S|} \sum_{Z^0 \in S} \min_{Z^1 \in P_F} d(Z^0, Z^1),$$

where $d(Z^0, Z^1)$ is the euclidean distance between the two solutions. A small value for this indicator means that all the points in the approximation set are, on average, close to the true pareto front. An ideal value of 0 is obtained when all the points are actually in the pareto front.

B. Diversity

To measure the diversity of the approximation set the reverse of the proximity metric is used, as defined in [1]:

$$D_{P_F \rightarrow S}(S) = \frac{1}{|P_F|} \sum_{Z^1 \in P_F} \min_{Z^0 \in S} d(Z^0, Z^1).$$

V. RESULTS

A. NSGA-II with Kuhn Munkres

As described above, in [4], NSGA-II and SPEA2, with informed initialization were applied on large instances of SAP problem. To illustrate how the diversity is improved with informed initialization, figures 1, 2 and 3 show the experimental results for a small instance of SAP with 1,000 sailors and 1,100 jobs. Fig.1 shows a parallel plot for the objective values of the assignments found a complete sample of obtained solving the single objective versions corresponding to the 287 weight vectors imposed by discretizing the simplex by 0.1 in each dimension using the Kuhn-Munkres algorithm. In the parallel plots, each objective is mapped onto one “bar” of the graph, and a single Pareto set approximation consists of a number of solutions, drawn as lines connecting the solution’s values for each objective function respectively.

Obviously, including the informed initialization into NSGA-II provides significantly increased performance, and begins to approach the performance of the full Kuhn-Munkres solution while using dramatically fewer computational resources. In preliminary testing, the evolutionary methods produced a five to ten fold speedup over running the full linear programming algorithm for every weight vector [4]. However, this performance does come at the cost of solution quality in the interior
regions of the Pareto front. The full Kuhn-Munkres results show a marked curve along the bottom edge of the parallel plots which is not present in the results of NSGA-II with the Kuhn-Munkres initialization. This curve is created by the increase in solution quality of the average interior solution in the full Kuhn-Munkres results. Essentially, many of the interior “lines” in the parallel plot are lower in several dimensions simultaneously, leading to an increase in the number of lines near the bottom edge of the curve. Considering the hypervolume, shown in Table I provides another view of this effect.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>1000/1100</th>
<th>2000/2100</th>
<th>4000/4300</th>
</tr>
</thead>
<tbody>
<tr>
<td>LP</td>
<td>0.3393</td>
<td>0.2394</td>
<td>0.2928</td>
</tr>
<tr>
<td>NSGA-II</td>
<td>0.1362</td>
<td>0.0806</td>
<td>0.0696</td>
</tr>
<tr>
<td>NSGA-II+LP</td>
<td>0.2331</td>
<td>0.1137</td>
<td>0.1840</td>
</tr>
</tbody>
</table>

The proposed approach was tested using the DTLZ3 problem, on 6 and 12 objectives, performing 5 million function evaluations; Fig.4 and Fig.5 show proximity and diversity of the populations of solutions for the NSGA-II standard and NSGA-II with informed initialization of 7 solutions.

In Fig.4(a), the proximity indicator for the two algorithms (NSGA-II standard and NSGA-II with informed initialization) shows a decreasing value, which means that the two algorithms are moving all the non-dominated points toward the actual pareto front. Note the small value found at the beginning of the NSGA-II with informed initialization algorithm. This is because in the first generations, the set of non-dominated solutions consists only of the approximations obtained with CHC, which are solutions closely resembling the actual pareto front. Then, the indicator gets a higher value as the number of points in the non-dominated set gets higher and, on average, those points are farther from the pareto front than those of the initial solutions.

On the other hand, in Fig.4(b) the value for the coverage indicator is very good from the beginning in the NSGA-II with CHC.
with informed initialization algorithm, and, for the NSGA-II standard algorithm. This indicator is very high and gets a good value only after several generations. To understand this behavior, this indicator should be seen along with the proximity indicator: a low coverage with a high proximity value mean that the approximation set is on average far from the pareto front, but there are some feasible solutions very good which are close to the front. When the proximity value decreases, this signifies that the solutions are moving toward the pareto front and are covering it more effectively. This performance is obtained consistently in the NSGA-II with informed initialization algorithm, but is only reached after several (in this case almost 5,000) generations in the NSGA-II standard algorithm.

The proposed approach was also tested using the mQAP problem, performing 5 million function evaluations. Fig.6 shows the proximity and diversity of the populations of solutions for the NSGA-II standard and NSGA-II with informed initialization of seven solutions for increasing numbers of CHC evaluations. Table II shows the average final proximity and coverage for 30 runs, showing that the informed initialization has a positive impact increasing the diversity and proximity, but the extra time devoted with CHC to find this initial informed solutions show no additional impact in the performance.

Finally, Fig.7 shows proximity and diversity of the populations of solutions for the NSGA-II standard and NSGA-II with informed initialization as the relative search effort allocated to CHC is increased. There are two ways of incrementing the number of evaluations performed by CHC: one is running
CHC more times which gives more informed solutions in the initial populations of NSGA-II; the second way is to perform larger runs of CHC which gives the same number of informed solutions, but those solutions are optimistically closer to the actual pareto front. Table IV shows the different configurations tested. Table III shows the average final proximity and coverage for 30 runs, showing that if the number of initial informed solutions is increased the performance increases, meaning better proximity and diversity, but only if the number of total function evaluations does not exceed 50% of the total number of evaluations. Here, in Fig. 7 the different configurations are used to change the % of evaluations performed by CHC with a basis of 5 million evaluations in total.

VI. Conclusions

In this paper, we have shown that the performance in terms of proximity and coverage of multiobjective algorithms can be improved through the injection of solutions generated by a high performance single objective search algorithm on various test problems. In addition, the selection scheme for a conventional multiobjective evolutionary algorithm was modified to increase the likelihood of selecting the injected solutions. We show that the best results in terms of proximity and coverage are obtained from NSGA-II with informed initialization when compared to the standard NSGA-II and SPEA2 algorithms for DTLZ3 test problem on 12 objectives.

Many hybrid MOEAs operate by subjecting the individuals to local search throughout the evolutionary process. In this work, we consider the ability of a multiobjective evolutionary algorithm to achieve good results through the injection of good solutions into the initial population. While performance was certainly improved, there remain several open questions regarding the relative merits of various forms of hybridization. This work provides a preliminary study from which many of these questions may be addressed in future work.

REFERENCES

Fig. 8. Comparison of the proximity and coverage of NSGA-II, SPEA2 with NSGA-II with informed initialization


