Efficient Hybrid Multi-Objective Evolutionary Algorithm

Tareq Abed Mohammed[†, ††], Oguz BAYAT[†], Osman N UÇAN[†]

[†]Collage of Engineering, Altinbas University, Istanbul, Turkey
[††]Collage of Science, Kirkuk University, Kirkuk, Iraq

Summary

In the artificial intelligence community the multi-objective optimization problem become very common and has been rapidly increasing attention. This significant is due to the fact that there is high number of real-world applications having optimization problems that include more than one objective function. As has been evident in the last ten years, the evolutionary algorithms are one of the best choices to solve multi-objective optimization problems. Although evolutionary algorithms are the most common approach to solve multi-objective optimization problems, there is still many issues and drawbacks that need solving and enhancing. In this paper a set of improved hybrid Memetic evolutionary algorithms are proposed to solve multi-objective optimization problems. The proposed algorithms enhance the performance of NSGA-II algorithm by using different new proposed and simple search schemes. Merging a simple and efficient search technique to NSGA-II significantly enhances the convergence ability and speed of the algorithm. To assess the performance of proposed algorithms, three multi-objective test problems are used from ZDT set. Our empirical results in this paper show that the proposed algorithms significantly enhance the NSGA-II algorithm performance in both diversity and convergence.

Keywords:

Evolutionary algorithms; Memetic algorithms; multi-objective optimization; high dimensional problems; hybrid algorithms

1. Introduction

Evolutionary algorithms (EAs) have proven to be a popular and useful choice when we need to get an estimated optimal solution for complex optimization problems [1]. The real-world optimization problems can be single objective and multi-objective and for each type there are different evolutionary algorithms to solve it. The high complexity of multi-objective optimization problems (MOPs) over the single objective optimization problems prevents the use of traditional single objective evolutionary algorithms in solving MOPs. Therefore, solving MOPs becomes a separated research field and has its own techniques, test problems and performance metrics. Recently the Multi-objective Optimization Evolutionary Algorithms (MOEA) become very important and widely used since there is many real world problems that include multi-objective functions such as telecommunication networks design problems [2], gas turbine combustion optimization problems [3], antenna design issues [4], scheduling optimization problems [5], and many other applications [6]. Recently, the MOPs again rose in many real world applications and many algorithms are proposed to solve such these problems as will be discussed in next sections. In other research work, researchers proposed to solve big data problems using multi-objective evolutionary algorithms [25] using different mechanisms such as fuzzy systems [23][24].

Formally, a general MOP can be defined as minimizing or maximizing a set of two or more objective functions where at least two of them must be in conflict with each other. Equation 1 shows the definition of MOP.

$$min/max\ F(x) = (f1(x), f2(x) \ldots , fk(x)) \ldots\ldots (1)$$

Subject to

$gi(x) < 0$, $i=1,2,3\ldots M$

$kj(x) = 0$, $j=1,2,3\ldots N$

The MOP consists of k objectives where k has to be greater than two and M is the number of equality constrains of the problem where N is the number of inequality constrains of the problem on the objective functions. The objective functions of the MOP can be linear or nonlinear, continuous or discrete and stationary or dynamic. In linear problems, the POF is linear due to the existence of linear functions in the objective functions of the MOP. In more complex scenario the POF of a problem may be discrete which increase the difficulty to solve such these problems. Recently the dynamic problems also become very important since there is many real world problems that may be dynamic in nature and continuously changing during the run of the evolutionary algorithm. In this study work we will focus only on the static problem where the proposed algorithms can be applicable on dynamic optimization problems after taking into account the dynamic environment conditions.

In order to solve MOP using genetic algorithms, MOEAs evolve a set of candidate solutions in order to find the optimal solutions (more than one solution) instead of finding a single optimal solution as in the single objective evolutionary algorithms. The collected best solution should be the best solutions and cover the entire true POF of the solved problem. After that the resulted set of best solutions is given to the decision maker who is responsible to select one of these solutions according to his

Manuscript received March 5, 2018
Manuscript revised March 20, 2018

requirements and plans. In literature POF is used to indicate the set of best solutions in the objective space where POS is used for best solutions in decision space [6]. In MOP solving approaches there is an additional difficulty when selecting best solutions in the population since there is more than one objective function (we cannot select only the highest value in maximization problem and the lowest value in minimization problem as in single objective problems).

Different approaches were developed to estimate the best solutions from the population set where the non-dominating sorting strategy is one of the best and popular sorting strategies that has been proposed. The non-dominating sorting algorithm extracts the best solutions from population by selecting the non-dominated solutions. The non-dominated solution is defined as the solution that is not dominated by any other solutions in the population. Using this approach in each dominance level, many solutions may be exist if they dominated by equal number of other candidate solutions. The first level (which is the POF) is considered to be the best solution set since this means that no other solution could dominate the solutions of that set.

The MOEA has to take into account two important tasks during the evolution process which are the convergence and diversity. The convergence means that the generated new solutions have to be close to the true Pareto-optimal front. Where the diversity means that the generated solutions should be distributed over all the POF and not gathered in clusters. In this paper, the MOEA is enhanced by merging the power of search algorithms which increase the convergence speed of the algorithm.

This paper is organized as follow; in the second section a short summary of the MOEAs is previewed. Our proposed algorithms and their steps are explained in section three. In section four, the experimental study and results discussion are given. Finally the paper will be concluded in section five.

2. Related work

The researchers give MOPs a high significant since they can be found in many applications. Therefore, in literature there are hundreds of papers published to address MOPs. In this section we will summarize the most important evolutionary algorithms used to solve MOPs.

A. The Vector Evaluated Genetic Algorithm (VEGA)

VEGA [7, 8] is one of the first algorithms that proposed to solve MOPs. The VEGA works by generating a separated sub population for each objective function. VEGA tries to guide the solutions toward local optima for each objective function separately. Each sub population of VEGA uses only the fitness function of their own objective for fitness assignment. After that a selection mechanism is used to select the best solutions and to generate the new mating pool. All sub populations are gathered in one population which is the next generation population.

B. The non-dominated sorting genetic algorithm (NSGA)

NSGA is a multi-objective algorithm developed and proposed in [9]. NSGA algorithm uses the concept of non-dominating sorting to sort the solutions and it has a good performance but it suffers from some problems such as the non-existence of elitism process and the high complexity of the algorithm. Because of these problems, Deb proposed his modified version of NSGA and names it as NSGA-II [10]. The Modified algorithm becomes one of the most popular algorithms in multi-objective domain. NSGA-II solved the two issues of NSGA algorithm and improved the implementation of NSGA to become less computational expensive. Because of the importance of this algorithm our proposed algorithms are developed based on it.
The NSGA-II is described briefly in the following steps:-
Step 1: Previous iteration parent and children populations are merged to obtain one new population with size 2N.

Step 2: The obtained population is sorted using non-domination sorting algorithm to give ranks for each solution where rank one indicates the best solutions.

Step 3: Transfer best solutions rank by rank starting with rank one to next generation population until we reach size N.

Step 4: Sort The last rank in the population (which if we add their solutions the size of the new population will be more than N) using crowding distance.

Step 5: Apply mutation and crossover operators and generate the new mating

Because of the high performance of NSGA-II, it is used to solve many real-world problems and to develop many other versions of this algorithm to solve stationary [11] and dynamic [12] optimization problems.

C. Strength Pareto Evolutionary Algorithm (SPEA)

SPEA is an efficient MOEA which introduced by Eckart Zitzler in [13]. This MOEA can be considered as a hybrid algorithm which integrates different MOEAs. The SPEA algorithm uses an external archive to gather the nondominated solutions that have been found previously. At each generation, the non-dominated solutions are copied to the archive. For each solution in the archive individuals, a strength value is computed as described in [14]. In SPEA, the fitness value of each individual from population is computed using the value of all archive nondominated solutions that dominate it. This new fitness assignment method takes into account the closeness of fitness values to the true POF and solutions distribution over the fitness space. This approach approves its effectiveness but sometimes it depends on the size of the archive of non-dominated solutions. In [15] the authors proposed an enhanced version of SPEA which is SPEA2. The new revised algorithm solves many issues of the first version and consequently the performance is enhanced.

D. Strength Pareto Evolutionary Algorithm (SPEA)

This algorithm is developed by modifying the normal genetic algorithm (GA) to deal with multiple objectives scenarios. This is done by adding the concept of Pareto domination solutions as in the previous algorithms to the selection operator, and after that a niching process is applied to spread the solutions of the population over the true Pareto optimal to make tradeoff between the objective functions [26]. The authors proved and demonstrated the ability of this algorithm to find and maintain a diverse "Pareto optimal solutions" using many real-world hydro systems problems and test problems.

In literature, there are still many MOEAs proposed such as the PAES [16], Niched-Pareto Genetic Algorithm (NPGA) [17] and Multiobjective Messy Genetic Algorithm (MOMGA) [18].Although there is a big number of proposed algorithms to solve MOP, many comparative experiments in MOEA research domain show that SPEA2 and NSGA-II are the most reliable and efficient algorithms and so they are widely used in comparing newly designed MOEAs.

3. Proposed Algorithms

As explained in previous section, NSGA-II algorithm is one of the best algorithms used to solve MOPs and because of this we selected this algorithm as a base of our new proposed algorithms. NSGA-II algorithm as many other MOEAs try to converges to true Pareto optimal front with keeping the solutions distributed as can as possible. Keeping the solutions distributed may slow the convergence process for NSGA-II algorithm. Our new proposed algorithms are developed mainly to enhance the NSGA-II algorithm convergence and keeping the solutions distributed during the evolution iterations. In order to increase the convergence speed of the NSGA-II algorithm an additional efficient search strategy is used to achieve long jumps toward the true Pareto optimal set.

E. The Proposed Search Algorithm

The proposed search strategy is described in Algorithm 1. Firstly, the value that will be added to each variable is computed. This value must be large in the early generations and then starts to decrease gradually while evolving. Therefore, it is multiplied by adapting factor which is 0.9 in our algorithm. Secondly this value is added or decreased from each variable in the considered solution. Finally the current solution, the solution results from adding the specified value and the solution results from decreasing the specified value are compared to determine which solution is better. Determining the best solution process is based on simple mechanism which is the sum of fitness function values. If the problem is minimization the solution with the minimum fitness value will be the best one.

Algorithm. 1. The steps of the proposed search Algorithm.

```
Begin
Nvar = getVaraiblesNum()  // get number of variables
S = GetOneSolution()        // select one solution from
population

// determine the value of increasing or decreasing each variable
For k = 0 to Nvar
      If (S[k] < ( MaxBound - S[k]))
               val = S[k]/2
      else
               val = (1-S[k])/2
      end if
end For

// decrease the value while evolving
val = val * 0.9

// increase the value and test
S[k] = S[k] + val
SInc = EvaluteOne(S)

// decrease the value and test
S[k] = S[k] – 2*val
SDec = EvaluteOne(S);


// select the best solution
Best = SelectBest(SInc, SDec, S)

end   // Program end
```

The proposed search algorithm is similar to the binary search since it search using two directions and the current position of each variable. The important thing is that if the algorithm could not find better value than the current then it keeps the considered variable without changing. This mechanism can effectively enhance the performance of the evolutionary algorithm without degrading the performance if we currently on the optimal solutions.

F. Combing Search Algorithm with NSGA-II

To get the best performance from the search algorithm we need to merge the search algorithm in a convenience way to not negatively affecting the NSGA-II performance. Algorithm 2 shows the steps of the proposed algorithm after adding the efficient search mechanism. As shown in the algorithm our proposed search mechanism is applied in the first step of each generation on the parent's individuals, after that best solutions are selected using non-domination strategy and crowding distance. In the last step, the crossover operator followed by mutation operator is applied to generate the new child population.

In this paper we designed four strategies to apply the efficient search mechanism in order to test different scheme and select the best one. The fore proposed scheme are different based on when and where to apply the search mechanism.

1) NSGA-II-M1: In the first version of proposed algorithm, the search algorithm is applied in each generation on all variables and all indivisuals in the population.

2) NSGA-II-M2: The second proposed memetic algorithm uses the search algorithm only on half of generations. So the search algorithm is applied on one generation and the next generation is performed without search mechanims.

3) NSGA-II-M3: In this version of proposed algorithm, the search algorithm is applied in each generation on all variables but using only the half indivisuals of the population. This strategy can gives the NSGA-II algorithm a chance to run 50% of population using its own mechanism which will avoid the full dependence on search algorithm.

Algorithm. 2. The steps of proposed memetic algorithm.

```
Begin
 Initialize Population (P)

for i = 1 to genNum do

//Apply the proposedSearch mechanism
fastSearchAlgorithm(Parent)

// Merge Parent and Child individuals in One 2*N Population
Pbig = MergeParentChild()

// Assign Rank (level) for each solution based on Pareto dominance
NonDominatingSort(Pbig)

// Add best solutions to next generation starting from the first Rank
Pnew = SelectBest(Pbig)

// Use crowding distance to select the remaining solutions from last
rank
Pnew = CrowdingDist(Pbig)

Pchild = TournamentSelection() //Binary Tournament Selection

Recombination(Pchild)
Mutation(Pchild)
end   // Generations end

end   // Program end
```

4) NSGA-II-AD: The last proposed memetic algorithm works adaptively and dynamic. In order to avoid degrading the best solutions on the population this algorithm applies the search mechanism only on the dominated solutions. Therefore the non-dominated solutions will remains without changing and solutions that have rank greater than 1 will be enhanced by the proposed search strategy. Using this mechanism the number of changed solutions is determined in each generation adaptively depending on the number of dominated solutions.

4. Experiments and discussions

In this section we will test the performance of proposed algorithms by performing a comprehensive comparison. The proposed algorithms will be compared with NSGA-II, using three test benchmarks.

G. Test Problems and Performance Metric

To evaluate the performance of the proposed algorithms, three multiobjective test problems were used. The three test problems are selected from ZDT test suite, (ZDT1 to ZDT3) [19]. The ZDT test suite is one of the most popular MOP test problems. These problems designed based on the principles of Deb in [20], the MOP in ZDT suite are constructed by using three functions, including a distribution function f1, a distance function g and a shape function h, function f1 is designated to test the ability of diversity maintain, function g is for testing the ability of convergence and finally function h for defining the shape of the PF [21]. The following equations define the three ZDT test problems:-

ZDT1:

$$\begin{aligned} f_1(x_1) &= x_1 \\ g(x_2,\ldots,x_m) &= 1+9\cdot\textstyle\sum_{i=2}^{m} x_i/(m-1) \\ h(f_1,g) &= 1-\sqrt{f_1/g} \end{aligned}$$

Where x ϵ [0,1] and POF can be generated when g(x) = 1.

ZDT2:

$$\begin{aligned} f_1(x_1) &= x_1 \\ g(x_2,\ldots,x_m) &= 1+9\cdot\textstyle\sum_{i=2}^{m} x_i/(m-1) \\ h(f_1,g) &= 1-(f_1/g)^2 \end{aligned}$$

Where x ϵ [0,1] and POF can be generated when g(x) = 1.

ZDT3:

$$\begin{aligned} f_1(x_1) &= x_1 \\ g(x_2,\ldots,x_m) &= 1+9\cdot\sum_{i=2}^{m} x_i/(m-1) \\ h(f_1,g) &= 1-\sqrt{f_1/g}-(f_1/g)\sin(10\pi f_1) \end{aligned}$$

Where x ϵ [0,1] and POF can be generated when g(x) = 1.

In our experiments, the Inverted Generational Distance (IGD) is mainly used [22] to test the performance of the proposed algorithms. The IGD metric can measure the diversity and convergence of the MOEAs. The IGD metric is computed for the final solutions of the population. The following equations define the IGD metric and show how it computed:-

$$\mathrm{IGD_t} = \frac{\sum_{t=1}^{|\mathrm{PF_t^*}|} \mathrm{d_i}}{|\mathrm{PF_t^*}|}$$

$$\mathrm{d_i} = \min_{k=1}^{|\mathrm{PF_t^*}|} \sqrt{\sum_{j=1}^{M} (f_j^{*(i)} - f_j^{(k)})^2}$$

Where the term $\mathrm{PF_t^*}$ indicates the optimal Pareto Front and M is the number of objective functions used in the problem. di is computed by using the Euclidean distance.

H. Results and Discussion

To test the performance of the proposed algorithms we implemented the proposed algorithms based on NSGA-II. The population size is selected to be 10, 20, 30 and 40. The crossover operator is fixed to be 0.7. The number of variables is selected to be 10 and two objectives are used in each test problem.

Table I, Table II and Table III show the performance comparison between the four proposed algorithms and The results of the three test problems show that the performance of NSGA-II is enhanced significantly when our efficient search algorithm applied on the half of the population and when it's applied on the bad solutions (the adaptive version). In addition, it is clear from results on tables that the NSGA-II-M1 and NSGA-II-M2 algorithms are still better than the NSGA-II in many cases. This result proves that adding a simple search algorithm on the evolutionary algorithms is very important and can be very helpful to guide the search mechanism of the evolutionary algorithms. In addition the usage of search algorithm in every generation may have negative effects on the diversity of solutions so it is preferable to restrict the use of the local search only in the needed situations as in the NSGA-II-M3 and NSGA-II-AD proposed algorithms.

In order to more investigate the performance of proposed algorithms Figure 2, Figure 3 and Figure 4 show the distribution of the solutions in last population (using 100 solutions in the population for each algorithm) after running 10, 20 and 30 generations and for three test problems ZDT1, ZDT2 and ZDT3 respectively. The figures compare the results of The NSGA-II algorithm and NSGA-II-AD which gains the

best performance among the other proposed algorithms. Firstly, Figure 1 shows the true POF of the three used test problems, as shown in the figure the first two problems are continuous and one problem is concave where the other one is convex. On the other hand, ZDT3 is a non-continuous POF problem as shown in Figure 1. From Figure2, Figure3 and Figure 4 we can see clearly that NSGA-II-AD is very fast in convergence process since the solutions started to be very close to the true Pareto optimal front only after 10 generations. On the other hand the NSGA-II algorithm performance after 10 generations is still not good and the solutions are far from the true POF. Figure 2, Figure 3 and Figure 4 also show that our

NSGA-II for ZDT1, ZDT2and ZDT3 problems respectively. For each experiment, the best result is marked by bold font. The values in tables represent the averages of 10 independent runs for each instance of the experiments. As shown in the results NSGA-II-M3 and NSGA-II-AD are the best two algorithms since they outperform the other algorithms in most of cases. The results also show that the NSGA-II algorithm without any searching algorithm is the worst algorithm. The NSGA-II-M3 algorithm which apply the search algorithm on half of population individuals outperforms the other algorithms in 9 cases out of 12, where NSGA-II-AD is the best algorithm in 8 cases.

Talbe I: The performance comparison between the proposed algorithms and NSGA-II algorithm for ZDT1 test problem

GEN	NSGA-II	NSGA-II-M1	NSGA-II-M2	NSGA-II-M3	NSGA-II-AD
10	0.1709	**0.0138**	0.0422	0.0223	**0.0141**
20	0.0329	0.0190	0.0151	**0.0102**	**0.0104**
30	0.0170	0.0147	0.0149	**0.0145**	0.0147
40	0.0193	0.0193	0.0199	**0.0191**	**0.0192**

Table II:The performance comparison between the proposed algorithms and NSGA-II algorithm for ZDT2 test problem

GEN	NSGA-II	NSGA-II-M1	NSGA-II-M2	NSGA-II-M3	NSGA-II-AD
10	0.5996	**0.3251**	0.5437	0.3880	0.4442
20	0.4082	0.2547	0.1984	**0.1034**	0.1775
30	0.0284	0.1538	0.0215	**0.0196**	**0.0189**
40	0.1577	0.1394	**0.0262**	0.0269	**0.0256**

Table III: The performance comparison between the proposed algorithms and NSGA-II algorithm for ZDT3 test problem

GEN	NSGA-II	NSGA-II-M1	NSGA-II-M2	NSGA-II-M3	NSGA-II-AD
10	0.1846	**0.0144**	0.0235	0.0184	**0.0147**
20	0.0272	0.0095	0.0088	**0.0086**	**0.0084**
30	0.0137	0.0117	**0.0113**	**0.0114**	**0.0112**
40	**0.0141**	**0.0143**	0.0144	**0.0143**	0.0144

proposed algorithm NSGA-II-AD can fasten the process of convergence significantly without degrading the diversity of solutions.

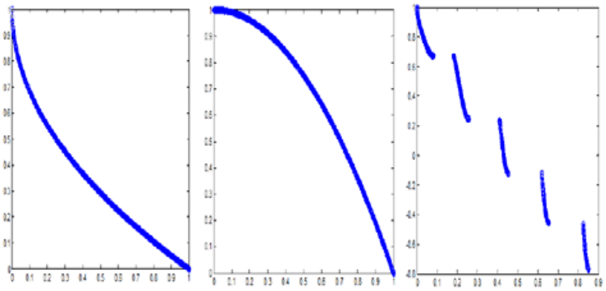

Fig. 1 The True POF distribution of ZDT1, ZDT2 and ZDT3 problems.

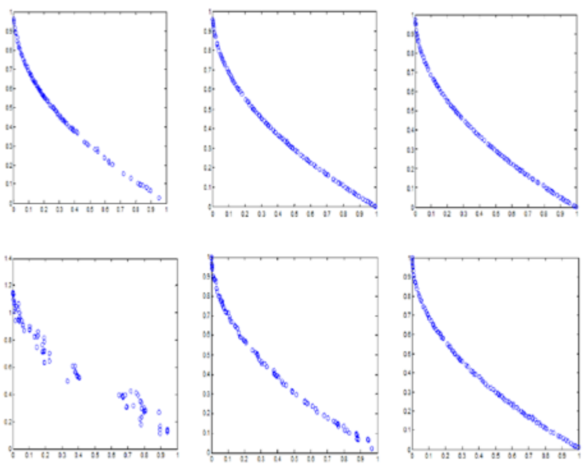

Fig. 2 The POF distribution of population after 10, 20 and 30 generations respectively of ZDT1 problem. The first column is for NSGA-II-AD and the second column is for NSGA-II algorithm

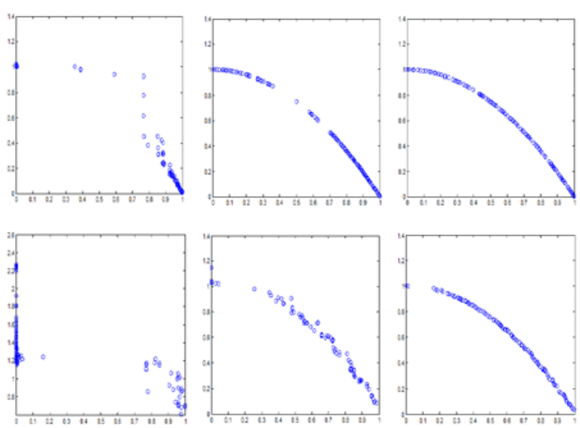

Fig. 3 The POF distribution of population after 10, 20 and 30 generations respectively of ZDT2 problem. The first column is for NSGA-II-AD and the second column is for NSGA-II algorithm

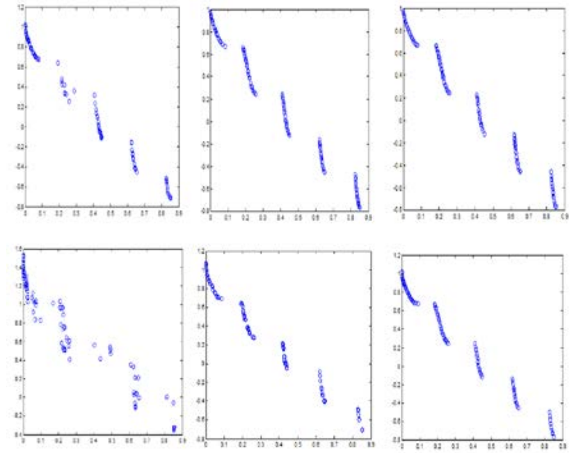

Fig. 4 The POF distribution of populaion after 10, 20 and 30 generations respectively of ZDT3 problem. The first column is for NSGA-II-AD and the second column is for NSGA-II algorithm

5. Conclusion

In this paper a set of new efficient and improved hybrid Memetic evolutionary algorithms are proposed to solve multi-objective optimization problems. The multi-objective optimization problems become very common and have been rapidly using in different domains.

The proposed algorithms in this paper enhance the performance of NSGA-II algorithm by using different search schemes. In this paper we designed four strategies to apply the efficient search mechanism in order to test different scheme and select the best one. The fore proposed scheme are different based on when and where to apply the search mechanism. In the first version of proposed algorithm, the search algorithm is applied in each generation on all variables and all indivisuals in the population. In the second proposed algorithm we used the search algorithm only on half of generations. So the search algorithm is applied on one generation and the next generation is performed without search mechanism. Version three of the proposed algorithm applies the search algorithm in each generation on all variables but using only the half indivisuals of the population. The last proposed algorithm works adaptively and dynamic. In order to avoid degrading the best solutions on the population this algorithm applies the search mechanism only on the dominated solutions. Therefore the non-dominated solutions will remains without changing

The results show that merging simple and efficient search techniques to NSGA-II significantly enhance the convergence ability to the set of optimal solutions. To assess the performance of proposed algorithms, three multi-objective test problems are used from ZDT set. Our empirical results show that the proposed algorithms significantly enhance the performance of NSGA-II algorithm and able to efficiently solve MOPs in less number of generations. The results also show that the proposed algorithm performs better than NSGA-II algorithm in both diversity and convergence.

References

[1] Y. Collette and P. Siarry, Multiobjective Optimization. Principles and Case Studies. Berlin, Germany: Springer-Verlag, 2003.

[2] S. Watanabe, T. Hiroyasu, and M. Miki, “Parallel evolutionary multicriterion optimization for mobile telecommunication networks optimization,” in EUROGEN 2001—Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems, Barcelona, Spain: EdsInternational Center for Numerical Methods in Engineering (CIMNE), Sept. 2001, pp. 167–172.

[3] L. Barone, L. While, and P. Hingston, “Designing crushers with a multiobjective evolutionary algorithm,” in Proc. Genetic and Evol. Comput.Conf., 2002, pp. 995–1002.

[4] N. V. Venkatarayalu and T. Ray, “Single and multi-objective design of Yagi-Uda antennas using computational intelligence,” in Proc. Congr. Evol. Comput., vol. 2, R. R. Ruhul Sarker, H. Abbass, K. C. Tan, B. McKay, D. Essam, and T. Gedeon, Eds., Dec. 2003, pp. 1237–1242.

[5] K. Shaw, A. L. Nortcliffe, M. Thompson, J. Love, P. J. Fleming, and C. M. Fonseca, “Assessing the performance of multiobjective genetic algorithms for optimization of a batch process scheduling problem,” in Proc. Congr. Evol. Comput., vol. 1, P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, and A. Zalzala, Eds., July 1999, pp. 37–45.

[6] Deb, Kalyanmoy, Karthik Sindhya, and Jussi Hakanen. "Multi-objective optimization." Decision Sciences: Theory and Practice. CRC Press, 2016. 145-184.

[7] J. D. Schaffer. Multiple Objective Optimization with Vector Evaluated Genetic Algorithms. In Genetic Algorithms and their Applications: Proceedings of the First International Conference on Genetic Algorithms, pages 93–100, Hillsdale, New Jersey, 1985. Lawrence Erlbaum.

[8] J. D. Schaffer and J. J. Grefenstette. Multiobjective Learning via Genetic Algorithms. In Proceedings of the 9th International Joint Conference on Artificial Intelligence (IJCAI-85), pages 593–595, Los Angeles, California, 1985. AAAI.

[9] Srinivas N. and Deb K.: Multiobjective optimization using nondominated sorting in genetic algorithms. IEEE Trans. Evol. comput., vol. 2, no. 3, pp. 221–248, 1994.

[10] Deb K., Pratap A., Agarwal S., and Meyarivan T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. comput., vol. 6, no. 2, pp. 182–197, Apr. 2002.

[11] N. Khan, D. E. Goldberg, and M. Pelikan. Multi-Objective Bayesian Optimization Algorithm. Technical Report 2002009, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, Urbana, Illinois, March 2002.

[12] S. Sahmoud and H. R. Topcuoglu, “A memory-based nsga-ii algorithm for dynamic multi-objective optimization problems,” in European Conference on the Applications of Evolutionary Computation. Springer, 2016, pp. 296–310.

[13] E. Zitzler and L. Thiele. Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. IEEE Transactions on Evolutionary Computation, 3(4):257–271, November 1999.

[14] C. M. Fonseca and P. J. Fleming. Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In S. Forrest, editor, Proceedings of the Fifth International Conference on Genetic Algorithms, pages 416–423, San Mateo, California, 1993. University of Illinois at Urbana- Champaign, Morgan Kaufmann Publishers.

[15] E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. In K. Giannakoglou, D. Tsahalis, J. Periaux, P. Papailou, and T. Fogarty, editors, EUROGEN 2001. Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems, pages 95–100, Athens, Greece, 2001.

[16] J. D. Knowles and D. W. Corne. Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. Evolutionary Computation, 8(2):149–172, 2000.

[17] J. Horn and N. Nafpliotis. Multiobjective Optimization using the Niched Pareto Genetic Algorithm. Technical Report IlliGAl Report 93005, University of Illinois at Urbana-Champaign, Urbana, Illinois, USA, 1993.

[18] D. A. Van Veldhuizen and G. B. Lamont. Multiobjective Optimization with Messy Genetic Algorithms. In Proceedings of the 2000 ACM Symposium on Applied Computing, pages 470–476, Villa Olmo, Como, Italy, 2000. ACM.

[19] E. Zitzler, K. Deb, and L. Thiele, “Comparison of multiobjective evolutionary algorithms: empirical results,” Evolutionary Computation, vol. 8, no. 2, pp. 173–195, 2000.

[20] K. Deb, “Multi-objective genetic algorithms: Problem difficulties and construction of test problems,” Evolutionary Computation, vol. 7, no. 3, pp. 205–230, 1999.

[21] Ran Cheng, Yaochu Jin, “Test Problems for Large-Scale Multiobjective and Many-Objective Optimization”, IEEE TRANSACTIONS ON CYBERNETICS.

[22] Zhou, Y. Jin, Q. Zhang, “A population prediction strategy for evolutionary dynamic multiobjective optimization,” IEEE Trans. Cybern., vol. 44, no. 1, pp. 40–53, 2014.

[23] Ferranti, Andrea, Francesco Marcelloni, and Armando Segatori. "A Multi-objective evolutionary fuzzy system for big data." Fuzzy Systems (FUZZ-IEEE), 2016 IEEE International Conference on. IEEE, 2016.

[24] Pulgar-Rubio, F., et al. "MEFASD-BD: Multi-objective evolutionary fuzzy algorithm for subgroup discovery in big data environments-A MapReduce solution." Knowledge-Based Systems 117 (2017): 70-78.

[25] M. Fazzolari, R. Alcalá, Y. Nojima, H. Ishibuchi, F. Herrera, "A review of the application of multi-objective evolutionary fuzzy systems: Current status and further directions", IEEE Transactions on Fuzzy Systems, vol. 21, no. 1, pp. 45-65, 2013.

[26] Horn, Jeffrey, Nicholas Nafpliotis, and David E. Goldberg. "A niched Pareto genetic algorithm for multiobjective optimization." Evolutionary Computation, 1994. IEEE

World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on. Ieee, 1994.

Tareq Abed Mohammed received B.Sc., degrees in Computer Science - College of Science,Kirkuk University, kirkuk, Iraq, in 2007 , and M.Sc degree from Cankaya University –Ankara – Turkey in 2012 .In 2014 I start to Ph.D. degree in Electronic and Computer Engineering at Altinbas University.Istanbul –Turkey. Since 2015- until now, he is teaching at graduate school of science and engineering GSSE at Altinbas University.

Oguz Bayat received the B.S. degree from Istanbul Technical University, Istanbul, Turkey, in 2000, and M.S degree from University of Hartford, CT, USA, in 2002, and a Ph.D. degree from Northeastern University, Boston, MA, USA, in 2006, all in electrical engineering. He completed the Executive Certificate Program in Technical Management and Leadership at Massachusetts Institute of Technology, Boston, MA, USA in 2009.

Osman N. Ucan received B.Sc., M.Sc. and Ph.D. degrees in Electronics and Communication Engineering, Istanbul Technical University in 1985, 1988 and 1995 respectively. He is currently working as Dean of Engineering and Natural Sciences at Altinbas University. His current research areas include information theory, jitter analysis, channel modeling, cellular neural network, random neural networks, wavelet, turbo coding and Markov Random Fields applications.