Journal of Scientific & Industrial Research
Vol. 80, February 2021, pp. 129-136

Non-Dominated Sorting Social Group Optimization Algorithm for Multi-Objective Optimization

Anima Naik[1], Junali Jasmine Jena[2] and Suresh Chandra Satapathy[2]*
[1] K L Deemed to be University, Hyderabad, India, 500 075
[2]School of Computer Engineering, KIIT Deemed to be University, India, 751 024

Received 09 June 2020: Revised 20 December 2020; accepted 02 January 2021

In this paper, authors have proposed a posterior multi-objective optimization algorithm named Non-dominated Sorting Social Group Optimization (NSSGO) for multi-objective optimization. 'Non-dominated Sorting' is the technique of sorting the population into several non-domination levels and 'Crowding Distance' is a concept used for maintaining diversity among the current best solutions. The algorithm acquires the combined concept of both. The proposed algorithm was simulated on a set of multi-objective CEC 2009 functions and competitive results were obtained.

Keywords: Crowding distance, Posterior Approach, Pareto fronts, Ranks

Introduction

Various real-world problems possess multi-objective solutions but with some degree of conflict among them. Such problems are popularly termed as multi-objective optimization problems (MOPs). Solution set these problems are called Pareto optimal solutions.[1] Multi-objective optimization comprises two independent tasks i.e. searching task and decision-making task. Searching task intends to find Pareto optimal solutions which should have minimal difference to the solutions on the true Pareto front, whereas choosing the most appropriate solution from the searched pareto optimal solutions is the goal of the decision-making task.

In 1984, the multi-objective evolutionary algorithm (MOEA)[2] was proposed by Schaffer to solve multi-objective problems. To solve the same, V Pareto, in 1896, defined the optimum in multi-objective optimization. As compared to the classical techniques, Evolutionary computation provides Pareto optimal solutions in a more quick and efficient way. Since 1984, several MOEAs have been proposed such as niched Pareto genetic algorithm (NPGA)[3], niched Pareto genetic algorithm 2 (NPGA2)[4], non-dominated sorting genetic algorithm (NSGA)[5], nondominated sorting genetic algorithm-II (NSGA-II)[6], strength Pareto evolutionary algorithm (SPEA)[7], strength Pareto evolutionary algorithm 2 (SPEA2)[8], multiobjective particle swarm optimization (MOPSO)[9], multiobjective differential evolution (MODE)[10], Multi-objective optimization using self-adaptive differential evolution (MOSaDE)[11], voronoi-based estimation of distribution algorithm (VEDA)[12], regularity model-based multiobjective estimation of distribution algorithm (RM-MEDA)[12], and multiobjective evolutionary algorithm based on decomposition (MOEA-D).[13] This work focuses on developing a non-dominated sorting social group optimization (NSSGO) for MOPs in which population is sorted into several non-domination levels using the 'NSGA-II concept' and diversity among the current best solutions is maintained using the 'crowding distance concept'. The crowding distance is directly proportional to the fitness of the solution. It was simulated upon six unconstrained benchmark problems of CEC 2009(14) for performance evaluation and the better solutions were obtained as compared to MOPSO and MOEA-D. This non-dominated concept is utilized for solving multi-objective optimization problems of papers.[15–19] These problems also can be solved using our proposal, NSSGO algorithm and kept as future work. The rest of the paper contains the description of the NSSGO, then results and discussions followed by conclusions.

The Proposed NSSGO Algorithm

NSSGO algorithm is the multi-objective version of the Social Group Optimization algorithm[20–23] which is inspired from the interaction of the humans living in a society. Here the candidate solution is represented by a person and every person has some personality traits

Author for Correspondence
E-mail: suresh.satapathyfcs@kiit.ac.in

using which he finds the solution to a problem. These personality traits represent the dimension of the problem or in other words the number of design variables a problem has. There are two phases in the algorithm i.e. 'improving phase' and 'acquiring phase'. Every person has got his own traits using which he solves any problem. These traits represent the dimension of the candidate solution. Improving Phase makes every individual to learn from the 'best person' and Acquiring Phase makes the individuals interact among each other and learn from the better one as well as from the 'best person'. The original paper can be referred to for details.

Description of the NSSGO

NSSGO algorithm is a supplement of SGO algorithm consists of improving phase and acquiring phase for solving MOPs. It is a posterior approach. The following sections describe these methods.

Selection Operator

In NSSGO, the rank-based comparison is performed to choose the best solution as it affects not only the convergence capability of the algorithm but also the divergence of non-dominated solutions. Rank is assigned with NSGA-II and crowding distance techniques. Here, the non-dominated solutions obtained in the previous iteration are kept by an abounded external archive. To ensure that the chosen solutions occupy the low-populated region of the search space, we select g_{best} among non-dominated solutions with the value of the highest crowding distance.

Initially, N numbers of solutions (persons) are created as a random population, followed by ranking and sorting according to the non-dominance concept. A person who achieves 1st rank is considered as 'g_{best}' or the best person. If more than one person has the same rank, then a person with a greater 'crowding distance' value is considered. Hence it is ensured that the best person (g_{best}) is selected from the low-populated region and diversity is maintained. Then it is followed by the 'improving phase' and 'acquiring phase' of the algorithm. Always the selection of the best person (g_{best}) follows the same rank based technique as mentioned above.

Non-Dominated Sorting

Rank in Non-Dominated sorting technique is based on the dominance concepts. According to this concept a solution P_i is said to dominate other solution P_j if and only if P_i is no worse than P_j with respect to all objectives and P_i is strictly better than P_j in at least one objective. The P_i does not dominate P_j if it violates any of the two conditions.

Suppose there is a set of random solution called S. When no other solutions dominate a particular solution, it is called a *'non-dominated solution'* and those which are segregated in the first run are set to Rank 1 and deleted. Again the same procedure is repeated for the rest of the solutions in S, till all solutions get their rank and are in sorted order.

Crowding Distance Sorting

Crowding distance[6] is the technique used to estimate the density of other solutions that surrounds a particular solution i in the population.

In two situations, a crowding distance mechanism is needed. First, when there is non-dominance between the target vector and trial vector, the crowding distance of the two are evaluated. Evaluation is performed based on the externally archived non-dominated solutions. For next-generation, the new target vector is the less crowded one. Secondly, the solutions present in the most populated search space should be deleted, when the size of the external archive exceeds a specified size. The computation of crowding distance is performed by sorting the population, in ascending order of magnitude, according to each objective function value, followed by assigning the boundary solutions with infinite distance values. Boundary solutions for each objective function are the solutions with the smallest and largest function values. The absolute normalized difference in the function values of two adjacent solutions is calculated and assigned to all other intermediate solutions. The same calculation is performed for other objective functions. The sum of individual distance values of each objective is calculated and taken as overall crowding-distance. Normalization of objective functions is performed prior to the calculation of crowding distance.

Pseudo-Code for NSSGO Algorithm

The pseudo-code of NSSGO is summarized as follows:

Initialize N (Number of persons, i.e., population size), person P_i, Max_Fes (Maximum number of function evaluations), c (self-introspection parameter)

Assign lower and upper bounds of decision variables $P_{min}[d]$ and $P_{max}[d]$, d=1,2,....., D, where D is the number of decision variables.

Generate N random solutions using a uniform distribution.

Evaluate function values at these N solutions. Adopt non-dominated and crowding distance sorting to these N solutions and store them in the current archive.

t=0;

While (t<Max_FEs) select the best solution 'g_{best}' from the current archive for i=1:N// Improving phase generate a trial individual T_i=c×P_i+rand×(g_{best}−P_i) evaluate function value; t++; non-domination checking of trial individual T_i with the target individual P_i if (T_i dominate P_i) replace P_i by T_i else if (T_i and P_i non-dominates each other) select an individual randomly and replace P_i end if end for

Combine these N new solutions with current archive N solutions, then select N fittest solution using non-dominated and crowding distance sorting from these 2N solutions and store them in current archive select the best solution 'g_{best}' from the current archive for i=1:N// Acquiring phase select randomly an individual P_j different from the target individual P_i if (P_j dominates P_i) T_i = P_i +rand×($P_i - P_j$)+rand×(g_{best}−P_i); else T_i=P_i+rand×($P_j - P_i$)+rand×(g_{best}−P_i); end if evaluate function value; t++; non-domination checking of trial individual T_i with the target individual P_i if (T_i dominate P_i) replace P_i by T_i else if (T_i and P_i non-dominates each other) select randomly an individual and replace P_i end if end for

Combine these N new solutions with current archive N solutions, then select N fittest solution using non-dominated and crowding distance sorting from these 2N solutions and store them in current archive end while

Results and Discussion

The experimentation and demonstration of the Non-dominated Sorting Social Group Optimization (NSSGO) algorithm has been carried out in this paper. The analysis of the results showcase how well the NSSGO performs on multi-objective test problems. We have considered six standard multi-objective test problems proposed in CEC 2009. Out of which three are bi-objective, and three are tri-objective test problems with ten decision variables. These benchmark problems are provided in Table 1. The performance of the NSSGO algorithm has been compared to MOEA/D and MOPSO of literature.

Experimental Setup

We have set the common control parameters such as population size (pop_size) = 100, maximum number of function evaluation (Max_FEs) = 30,000 and number of archive point = 100 for all experiments. The other algorithmic specific parameters are considered as follows. The values assigned to the parameters have been taken from the original paper of SGO.[16]

Parameter values for MOPSO are as follows:

- c1 (represents the cognitive parameter) = 1
- c2 (represents the social parameter) = 2
- w (inertia weight) = 0.5
- α (grid inflation parameter) = 0.1
- β (leader selection pressure parameter) = 4
- γ (deletion selection pressure parameter) = 2
- n Grid (no. of grids in each dimension) =10

Parameter values for MOEA/D are as follows:

- N (No. of Sub-problems) = 100
- T (number of neighbors) = 0.1N = 10
- n_r (the maximal copies of a new child in update) = 0.01N = 1
- δ (the probability of selecting parents from the neighborhood) = 0.9
- CR =F (mutation rates) = 0.5
- ρ (distribution index) = 30

Parameter values for NSSGO:

- C (self-introspection parameter) = 0.2

The entire experiment is conducted in MATLAB 2016a in Windows 10 Environment, on a machine having an Intel Core i5 processor with 8 GB memory.

For estimation of convergence, IGD (Inverted Generational Distance)[24] has been used as a performance metric. For quantification and measurement of the coverage, SP (Spacing)[25] and MS (Maximum Spread)[26] have been used. The mathematical definition of IGD, SP, and MS are given below:

IGD = $\frac{\sqrt{\sum_{i=1}^{n} d_i^2}}{n}$, where 'n' is the solutions on the Pareto front and d_i is the value obtained by finding the Euclidean distance between ith true Pareto optimal solution and nearest obtained Pareto optimal solutions in objective space.

SP=$\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(\overline{d} - d_i)^2}$, where $\overline{d}$ signifies the average of all d_i , n is the number of obtained solutions on the Pareto-front and d_i = $\min_j(\sum_{k=1}^{M}\left|f_k^i - f_k^j\right|)$ for all i,j = 1,2,3,….,n and M is No. of objective functions.

Table 1 — Mathematical formulation of multiobjective problems

Name	Mathematical formulation, Bi-objective
UF1	$f_1 = x_1 + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1}[x_j - \sin(6\pi x_1 + \frac{j\pi}{n})]^2$, $f_2 = 1 - \sqrt{x} + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_2}[x_j - \sin(6\pi x_1 + \frac{j\pi}{n})]^2$
	$J_1 = \{j \mid j\ \mathit{is\ odd\ and}\ 2 \le j \le n\}, J_2 = \{j \mid j\ \mathit{is\ even\ and}\ 2 \le j \le n\}$
UF2	$f_1 = x_1 + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1} {y_j}^2$, $f_2 = 1 - \sqrt{x} + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_2} {y_j}^2$
	$J_1 = \{j \mid j\ \mathit{is\ odd\ and}\ 2 \le j \le n\}, J_2 = \{j \mid j\ \mathit{is\ even\ and}\ 2 \le j \le n\}$
	$y_j = \begin{cases} x_j - \left[0.3x_1^2\cos\left(24\pi x_1 + \frac{4j\pi}{n}\right) + 0.6x_1\right] cos\left(6\pi x_1 + \frac{j\pi}{n}\right)\ if\ j \in J_1 \\ x_j - \left[0.3x_1^2\cos\left(24\pi x_1 + \frac{4j\pi}{n}\right) + 0.6x_1\right] sin\left(6\pi x_1 + \frac{j\pi}{n}\right)\ if\ j \in J_2 \end{cases}$
UF3	$f_1 = x_1 + \frac{2}{\lvert J_1\rvert}(4\sum_{j\in J_1} {y_j}^2 - 2\prod_{j\in J_1}\cos\left(\frac{20y_{j\pi}}{\sqrt{j}}\right) + 2)$, $f_2 = \sqrt{x_1} + \frac{2}{\lvert J_2\rvert}(4\sum_{j\in J_2} {y_j}^2 - 2\prod_{j\in J_2}\cos\left(\frac{20y_{j\pi}}{\sqrt{j}}\right) + 2)$
	$J_1 = \{j \mid j\ \mathit{is\ odd\ and}\ 2 \le j \le n\}, J_2 = \{j \mid j\ \mathit{is\ even\ and}\ 2 \le j \le n\}, y_j = x_j - x_1^{0.5(1.0+\frac{3(j-2)}{n-2})}$ j=2,3,.......,n
	Mathematical formulation, Tri-objective
UF8	$f_1 = \cos(0.5x_1\pi)\cos(0.5x_2\pi) + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1}(x_j - 2x_2\sin(2\pi x_1 + \frac{j\pi}{n})^2)$
	$f_2 = \cos(0.5x_1\pi)\sin(0.5x_2\pi) + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_2}(x_j - 2x_2\sin\left(2\pi x_1 + \frac{j\pi}{n}\right))^2)$
	$f_3 = \sin(0.5x_1\pi) + \frac{2}{\lvert J_3\rvert}\sum_{j\in J_3}(x_j - 2x_2\sin(2\pi x_1 + \frac{j\pi}{n})^2)$
	$J_1 = \{j \mid 3 \le j \le n, and\ j-1\ is\ a\ multiplication\ of\ 3\}, J_2 = \{j \mid 3 \le j \le n, and\ j-2\ is\ a\ multiplication\ of\ 3\}$
	$J_3 = \{j \mid 3 \le j \le n, and\ j\ is\ a\ multiplication\ of\ 3\}$
UF9	$f_1 = 0.5[\max\{0, (1+\epsilon)(1-4(2x_1-1)^2)\} + 2x_1]x_2 + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1}(x_j - 2x_2\sin(2\pi x_1 + \frac{j\pi}{n})^2)$
	$f_2 = 0.5[\max\{0, (1+\epsilon)(1-4(2x_1-1)^2)\} + 2x_1]x_2 + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_2}(x_j - 2x_2\sin\left(2\pi x_1 + \frac{j\pi}{n}\right))^2)$
	$f_3 = 1 - x_2 + \frac{2}{\lvert J_3\rvert}\sum_{j\in J_3}(x_j - 2x_2\sin(2\pi x_1 + \frac{j\pi}{n})^2)$
	$J_1 = \{j \mid 3 \le j \le n, and\ j-1\ is\ a\ multiplication\ of\ 3\}, J_2 = \{j \mid 3 \le j \le n, and\ j-2\ is\ a\ multiplication\ of\ 3\}$
	$J_3 = \{j \mid 3 \le j \le n, and\ j\ is\ a\ multiplication\ of\ 3\}, \epsilon = 0.1$
UF10	$f_1 = \cos(0.5x_1\pi)\cos(0.5x_2\pi) + \frac{2}{\lvert J_1\rvert}\sum_{j\in J_1}[4{y_j}^2 - co\,s(8\pi y_j) + 1]$
	$f_2 = \cos(0.5x_1\pi)\sin(0.5x_2\pi) + \frac{2}{\lvert J_2\rvert}\sum_{j\in J_1}[4{y_j}^2 - co\,s(8\pi y_j) + 1]$
	$f_3 = \sin(0.5x_1\pi) + \frac{2}{\lvert J_3\rvert}\sum_{j\in J_1}[4{y_j}^2 - co\,s(8\pi y_j) + 1]$
	$J_1 = \{j \mid 3 \le j \le n, and\ j-1\ is\ a\ multiplication\ of\ 3\}, J_2 = \{j \mid 3 \le j \le n, and\ j-2\ is\ a\ multiplication\ of\ 3\}$
	$J_3 = \{j \mid 3 \le j \le n, and\ j\ is\ a\ multiplication\ of\ 3\}\ y_j = x_j - 2x_2\sin(2\pi x_1 + \frac{j\pi}{n})$, j = 3,4,.....,n

The maximum spread is given by

$$MS\ (\overline{PF}, PF) = \sqrt{\frac{\sum_{j=1}^{M}(\frac{\min(PF_{j,u},\overline{PF}_{j,u}) - \max(PF_{j,l},\overline{PF}_{j,l})}{\overline{PF}_{j,u} - \overline{PF}_{j,l}})^2}{M}}$$

where $PF_{j.l}$ and $PF_{j.u}$ are the minimum and maximum value of the jth objective in the obtained Pareto optimal solutions. $\overline{PF}_{j,u}$ and $\overline{PF}_{j,l}$ are the maximum and minimum value of jth objective in true Pareto optimal solutions. Measurement of coverage of optimal PF by obtained PF is performed by MS. Higher the value of MS, the larger the area of $\overline{PF}$ covered by PF.

IGD, SP, and MS performance metrics allow comparing the performance of the algorithms

NSSGO, MOPSO, and MOEA/D qualitatively and quantitatively both in parameter space and search space. Tables 2–4 shows the statistical results obtained by running the algorithms ten times on the test problems, and qualitative results are provided in Fig. 1.

Performance Analysis

Statistical results of the algorithms for IGD have been provided in Table 2. It shows the accuracy and convergence of an algorithm. Statistical results for SP have been given in Table 3, and Table 4 provides statistical results of the algorithms for MS that is the measure of coverage of an algorithm. From the tables, we have found out the following observations:

For UF1 Problem:

NSSGO finds the best of best performance and best of worse performance, whereas MOPSO finds the best of average performance with respect to IGD. MOEA/D finds best in all performances such as best, average, and worse performance with respect to SP. Similarly, MOPSO finds best in both best performance and average performance, and NSSGO finds best in worse performance to MS for the UF1 problem. The graphical representation of Pareto-optimal solutions obtained from each algorithm on UF1 is depicted in Fig. 1. Hence, it can be stated that the MOPSO algorithm provides better convergence and best coverage onUF1 in comparison to other algorithms.

For UF2 Problem:

MOEA/D finds the best of best performance, whereas NSSGO finds the best of average performance and best of worse performance with respect to IGD. NSSGO finds best in the best performance, best in average performance, and best in worse performance with respect to SP. Similarly, MOPSO finds best in both best performance and average performance, and NSSGO finds best in worse performance to MS for the UF2 problem. The

Table 2 — Statistical results for IGD on benchmark problems

	Result of MOPSO	Result of MOEA/D	Result of NSSGO	Result of MOPSO	Result of MOEA/D	Result of NSSGO
	UF1			UF2		
Best	0.0993	0.0963	0.0884	0.0852	0.0385	0.0447
Median	0.1497	0.1325	0.1942	0.1153	0.0571	0.0625
Worse	0.3477	0.4496	0.3436	0.1915	0.1506	0.1029
Mean	0.1675	0.1795	0.2006	0.1209	0.0824	0.0722
Std	0.0681	0.1053	0.0981	0.0298	0.0418	0.0186
	UF3			UF8		
Best	0.4389	0.3004	0.4085	0.2690	0.1558	0.1023
Median	0.7141	0.5850	0.5474	0.3495	0.2358	012941
Worse	1.1194	0.9261	0.6944	0.5315	0.3908	0.1247
Mean	0.7261	0.5945	0.5707	0.3789	0.2555	0.1957
	UF9			UF10		
Best	0.4050	0.1920	0.2765	1.9766	0.8789	0.4261
Median	0.5660	0.2359	0.3731	3.4551	1.1262	0.4387
Worse	0.7621	0.3244	0.4582	5.9429	1.6621	0.4411
Mean	0.5884	0.2549	0.3793	3.7770	1.2215	0.4374
Std	0.1173	0.0488	0.0584	1.2830	0.2236	0.0046

Table 3 — Statistical results for SP on benchmark problems

	Result of MOPSO	Result of MOEA/D	Result of NSSGO	Result of MOPSO	Result of MOEA/D	Result of NSSGO
		UF1			UF2	
Best	0.0153	2,5612E-04	0.0135	0.0372	0.0045	0.0038
Median	0.0526	8.5626e-04	0.0195	0.0544	0.0101	0.0074
Worse	0.1017	0.0082	0.0370	0.1469	0.0168	0.0139
Mean	0.0576	0.0022	0.0229	0.0749	0.0104	0.0091
Std	0.0289	0.0027	0.0080	0.0389	0.0039	0.0036
		UF3			UF8	
Best	8.5622e-04	1.0002e-04	0.0133	0.3537	0.0269	0.0095
Median	0.1271	2.0961e-04	0.0226	0.5987	0.0422	0.0321
Worse	0.3971	0.0078	0.0941	1.1303	0.0991	0.1692
Mean	0.1486	0.0017	0.0333	0.7304	0.0502	0.0493
Std	0.1262	0.0026	0.0252	0.2765	0.0210	0..0317
		UF9			UF10	
Best	0.5025	0.0191	0.1409	2.0817	8.9012e-04	0.0071
Median	0.7249	0.0426	0.2774	3.4407	0.0409	0.0079
Worse	1.4223	0.0700	0.6052	4.9718	0.1774	0.0110
Mean	0.8363	0.0139	0.3587	3.5655	0.0552	0.0084
Std	0.3323	0.0139	0.1770	1.0722	0.0515	0.0011

Table 4 — Statistical results for MS on benchmark problems

	UF1			UF2		
	Result of MOPSO	Result of MOEA/D	Result of NSSGO	Result of MOPSO	Result of MOEA/D	Result of NSSGO
Best	0.9969	0.8438	0.9843	0.9997	0.9423	0.9865
Median	0.9850	0.5773	0.9558	0.9928	0.7732	0.9762
Worse	0.6695	0.2375	0.8082	0.9090	0.4658	0.9293
Mean	0.9447	0.5201	0.9233	0.9747	0.7170	0.9652
Std	0.1023	0.2046	0.0582	0.0304	0.1724	0.0201
	UF3			UF8		
	Result of MOPSO	Result of MOEA/D	Result of NSSGO	Result of MOPSO	Result of MOGWO	Result of NSSGO
Best	0.7162	0.3688	0.9620	0.9995	0.9986	0.9999
Median	0.3877	0.3240	0.8566	0.9953	0.9410	0.9995
Worse	1.2911e-04	1.8857e-04	0.7796	0.9863	0.8076	0.9867
Mean	0.3370	0.2190	0.8615	0.9944	0.9121	0.9983
Std	0.2276	0.1500	0.0523	0.0037	0.0625	0.0045
	UF9			UF10		
	Result of MOPSO	Result of MOEA/D	Result of NSSGO	Result of MOPSO	Result of MOEA/D	Result of NSSGO
Best	0.9987	0.9503	0.9999	0.7908	0.6483	0.8165
Median	0.9945	0.8258	0.9997	0.6310	0.5078	0.8164
Worse	0.9732	0.6126	0.9894	0.2633	0.0447	0.8161
Mean	0.9920	0.8106	0.9981	0.5685	0.4556	0.8164
Std	0.0083	0.1006	0.0035	0.1584	0.1793	1.1064e-04

graphical representation of Pareto-optimal solutions obtained from each algorithm on UF2 is depicted in Fig. 1. Hence, it can be stated that the NSSGO algorithm provides better convergence on UF2 in comparison to all other algorithms.

For UF3 Poblem:

MOEA/D finds the best of best performance, whereas NSSGO finds the best of average performance and best of worse performance with respect to IGD. MOEA/D finds best in the best performance, best in average performance, and best in worse performance with respect to SP. Similarly, NSSGO finds best in the best performance, best in average performance and best in worse performance to MS for the UF3 problem. The graphical representation of Pareto-optimal solutions obtained from each algorithm on UF3 is depicted in Fig. 1. Hence, it can be stated that the NSSGO algorithm provides better convergence and best coverage on UF3 in comparison to all other algorithms.

For UF8 Problem:

NSSGO finds the best of best performance, best of average performance, and best of worse performance with respect to IGD. NSSGO finds best in the best performance, best in average performance, whereas MOEA/D finds best in worse performance with respect to SP. Similarly, NSSGO finds best in the best performance, best in average performance and best in worse performance to MS for the UF8 problem. The graphical representation of Pareto-optimal solutions obtained from each algorithm on UF8 is depicted in Fig. 1. Hence, it can be stated that the NSSGO algorithm provides better convergence and best coverage on UF8 in comparison to all other algorithms.

For UF9 Problem:

MOEA/D finds the best of best performance, best of average performance, and best of worse performance with respect to both in IGD and SP. Similarly, NSSGO finds best in the best performance, best in average performance and best in worse performance to MS for the UF9 problem. The graphical representation of Pareto-optimal solutions obtained from each algorithm on UF9 is depicted in Fig. 1. Hence, it can be stated that the MOEA/D algorithm provides better convergence on UF9 in comparison to other algorithms.

For UF10 Problem:

NSSGO finds the best of best performance, best of average performance, and best of worse performance with respect to all metrics such as IGD, SP, and MS. The graphical representation of Pareto-optimal solutions obtained from each algorithm on UF10 is depicted in Fig. 1. Hence, it can be stated that the NSSGO algorithm can provide superior convergence and best coverage on UF10 in comparison to all other algorithms.

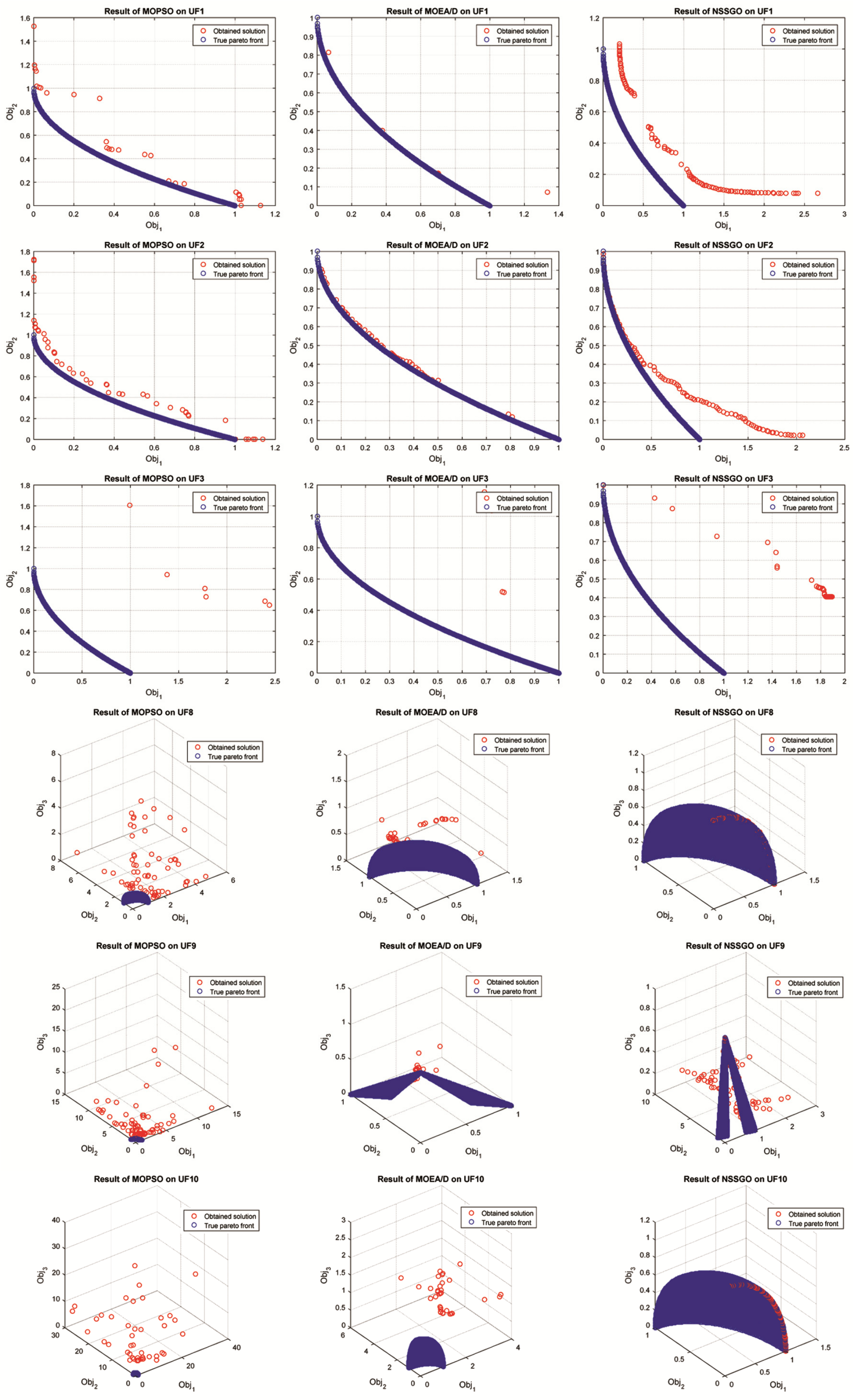

Fig. 1 — Obtained Pareto Optimal solution by MOPSO, MOEA/D, and NSSGO

So in overall, we have concluded that the NSSGO algorithm able to find superior convergence and best coverage in all four problems out of 6 six problems: 2 in bi-objective and 2 in tri-objective problems whereas MOPSO finds its superior performance in one bi-objective and MOEA/D finds its superior performance in one tri-objective problem.

Conclusions

The proposed NSSGO algorithm was evaluated using three bi-objective and three tri-objective optimization problems of CEC 2009 and found effective and efficient as compared to MOPSO and MOEA/D. The performance of the algorithm is measured using the IGD, SP, and MS metric. From the experimental results, we have concluded that the NSSGO algorithm is able to find superior convergence and best coverage in 4 problems out of 6 six problems. It can be summarized that NSSGO provides promising solutions to multi-objective optimization problems. As a further study, we shall evaluate the multi-objective optimization problems of CEC2015 using NSSGO algorithm and also we shall try to improve the NSSGO algorithm to solve a many-objective optimization problem.

References

1 Deb K, Multi-objective optimization using evolutionary algorithms (John Wiley & Sons) 2001.

2 Schaffer J D, Multiple objective optimizations with vector evaluated genetic algorithms, In Proc first Int conf genetic algorithms their appl (Lawrence Erlbaum Associates. Inc., Publishers) 1985.

3 Horn J, Nafpliotis N & Goldberg D E, A niched Pareto genetic algorithm for multiobjective optimization, In Proc first IEEE conf Evol Comput, IEEE world congress on computational intelligence (1994) 82−87.

4 Erickson M, Mayer A & Horn J, The niched Pareto genetic algorithm 2 applied to the design of groundwater remediation systems, In Int Conf Evol Multi-Criterion Optimization, Springer, Berlin Heidelberg, (2001) 681−695.

5 Srinivas N & Deb K, Muiltiobjective optimization using nondominated sorting in genetic algorithms, Evol Comput, **2(3)** (1994) 221−48.

6 Deb K, Pratap A, Agarwal S & Meyarivan T A, A fast and elitist multiobjective genetic algorithm, NSGA-II, *IEEE Trans Evol Comput*, **6(2)** (2002) 182−97.

7 Zitzler E & Thiele L, Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach, *IEEE Trans Evol Comput*, **3(4)** (1999) 257−271.

8 Zitzler E, Laumanns M & Thiele L, SPEA 2: improving the strength Pareto evolutionary algorithm, in Proc Intl Conf Evol Methods Design Optimisation Control with Appl Ind Problems, (2002) 95−100.

9 Coello C A, Pulido G T & Lechuga M S, Handling multiple objectives with particle swarm optimization, *IEEE Trans Evol Comput*, 8(**3**) (2004) 256−79.

10 Xue F, Sanderson A C & Graves R J, Pareto-based multi-objective differential evolution, in The 2003 Congress Evol Comput, CEC'03, **2** (2003) 862−869.

11 Huang V L, Zhao S Z, Mallipeddi R & Suganthan P N, Multi-objective optimization using self-adaptive differential evolution algorithm, in IEEE Congress Evol Comput (2009) 190−194.

12 Zhang Q, Zhou A & Jin Y, RM-MEDA: A regularity model-based multiobjective estimation of distribution algorithm, *IEEE Trans Evol Comput*, **12(1)** (2008) 41−63.

13 Zhang Q & Li H, MOEA/D: A multiobjective evolutionary algorithm based on decomposition, *IEEE Trans Evol Comput*, **11(6)** (2007) 712−31.

14 Zhang Q, Zhou A, Zhao S, Suganthan P N, Liu W & Tiwari S, Multiobjective optimization test instances for the CEC 2009 special session and competition, (2009).

15 Bao C, Xu L, Goodman E D & C Leilei, A novel non-dominated sorting algorithm for evolutionary multi-objective optimization, *J Comput Sci*, **23** (2017), 31−43, https://doi.org/10.1016/j.jocs.2017.09.015.

16 Zhang M, Wang L, Cui Z, Liu J, Du D & Guo W, Fast Nondominated Sorting Genetic Algorithm II with Lévy Distribution for Network Topology Optimization, Math Probl Eng, (2020) https://doi.org/10.1155/2020/3094941

17 Singh P & Chaudhary H, Optimal shape synthesis of a metallic flywheel using non-dominated sorting Jaya algorithm, Soft Comput, 24 (2020) 6623–6634. https://doi.org/10.1007/s00500-019-04302-x.

18 Esfe M H, Amiri M K & Bahiraei M, Optimizing thermophysical properties of nano fluids using response surface methodology and particle swarm optimization in a non-dominated sorting genetic algorithm, *J Taiwan Inst Chem Eng*, **103** (2019) 7−19

19 Liu Q, Tang R & Ren H, Optimizing multicast routing tree on application layer via an encoding-free non-dominated sorting genetic algorithm, Appl Intell, 50 (2020) 759–777. https://doi.org/10.1007/s10489-019-01547-9

20 Satapathy S C & Naik A, Social group optimization (SGO): a new population evolutionary optimization technique, Complex intell systems, 2(3) (2016) 173−203.

21 Naik A, Satapathy S C, Ashour A S & Dey N, Social group optimization for global optimization of multimodal functions and data clustering problems, Neural Comput Appl, 30(1) (2018) 271−87.

22 Naik A &, Satapathy S C, A comparative study of social group optimization with a few recent optimization algorithms, Complex intell systems (2020), https://doi.org/10.1007/s40747-020-00189-6

23 Naik A, Satapathy S C & Abraham A, Modified Social Group Optimization - a meta-heuristic algorithm to solve short-term hydrothermal scheduling, Appl Soft Comput, 95 (2020), https://doi.org/10.1016/j.asoc.2020.106524.

24 Sierra M R & Coello C A, Improving PSO-based multi-objective optimization using crowding, mutation and ∈-dominance, in International conference on evolutionary multi-criterion optimization, Springer Berlin Heidelberg, (2005) 505−519.

25 Coello C A & Lamont G B, Applications of multi-objective evolutionary algorithms, (World Scientific) 2004.

26 Zitzler E &Thiele L, Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach, *IEEE Trans Evol Comput*, **3(4)** (1999) 257−271.