International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:12, No:11, 2018

An Enhanced Particle Swarm Optimization Algorithm for Multiobjective Problems

Houda Abadlia, Nadia Smairi, Khaled Ghedira

Abstract**—**Multiobjective Particle Swarm Optimization (MOPSO) has shown an effective performance for solving test functions and real-world optimization problems. However, this method has a premature convergence problem, which may lead to lack of diversity. In order to improve its performance, this paper presents a hybrid approach which embedded the MOPSO into the island model and integrated a local search technique, Variable Neighborhood Search, to enhance the diversity into the swarm. Experiments on two series of test functions have shown the effectiveness of the proposed approach. A comparison with other evolutionary algorithms shows that the proposed approach presented a good performance in solving multiobjective optimization problems.

Keywords**—**Particle swarm optimization, migration, variable neighborhood search, multiobjective optimization

I. Introduction

MULTIOBJECTIVE optimization problems have more than one independent objective that must be optimized at once. There is a set of the so-called non-dominated solutions (those which better satisfy each of the functions) rather than just one, which is called the Pareto set. The graphical representation of this Pareto-optimal set of solutions is called the Pareto front. Thus, the main issue of a good multi-objective algorithm particularly for the metaheuristics is to obtain the maximum number of non-dominated solutions, having a high diversity and spread, and being as closer to the optimal set as possible [1].

Particle swarm optimization (PSO) is an evolutionary computation technique bio-inspired by the social behavior of species, such as a birds flock or a fish school. This algorithm is based on a particle's population to find solutions through hyper dimensional search space. The change of the particle's position is based on the socio-psychological tendency of particles to emulate the success of other particles. Each particle has an associated velocity vector which drives the optimization process and reflects the socially exchanged information. Although there has been considerable research conducted on PSO in order to solve multiobjective optimization problem (MOP) and to improve the convergence and diversity of the approximate Pareto front further still remains an issue that needs to be considered. An improved MOPSO has been developed in this paper. Our aim is to improve both diversity and convergence by combining island model with MOPSO and by using a local search technique in order to obtain a good balance between the exploitation and exploration of the search space.

The paper is structured as follows: the concept of PSO and the island models are briefly reviewed in Section II. In Section III, a multiobjective PSO is proposed to improve diversity and convergence to the true Pareto front. In Section IV, the results of the experiments and the analysis are shown. Conclusions are given in Section V.

II. Preliminary

A. PSO

PSO is a metaheuristic proposed in 1995 by Kennedy and Eberhart [2]. PSO is a population-based stochastic approach for solving continuous and discrete optimization problems. The concept under PSO is to emulate the social interaction behavior of birds flock and fish school. It utilizes a population of particles that fly through the problem hyperspace with given velocities. At each iteration, the velocities of the particles are stochastically adjusted according to the influence of its best solution and of the best solution of its neighbors, then computes a new point to be evaluated. The displacement of a particle is influenced by three components:

- Physical component: the particle tends to keep its current direction of displacement;
- Cognitive component: the particle tends to move towards the best explored site until now;
- Social component: the particle tends to rely on the experience of its congeners, then moves towards the best site already explored by its neighbors.

The displacement of each particle in the search space is based on its current position and the update of its velocity. Let $x_i(t)$ be the position of the particle p_i at the time step *t*. The particle position p_i is modified by the addition of the velocity $v_i(t)$ of the current position:

$$x_i(t) = x_i(t-1) + v_i(t) \quad (1)$$

Each particle in the swarm, changes its velocity according to two essential information. The first information is related to its personal experience, which is the best position found by the particle during the search process, which is noted pbest. The second information is the best position found by the whole swarm. This information is obtained from the knowledge of how the other particles performed their searches.

The principle change of the velocity is defined as follows:

$$v_i(t) = wv_i(t-1) + r_1c_1\left(x_{pbest_i} - x_i(t)\right) + r_2c_2\left(x_{gbest} - x_i(t)\right) \quad (2)$$

Houda Abadlia is with the National School of Computer Science, University of Manouba, Tunisia (e-mail: houdaabadlia@gmail.com).

The $v_i(t)$ is the velocity of the i-th particle ($i \in 1,2,\ldots,s$) of the d-th dimension where: c_1, c_2 are the learning factors that will be fixed throughout the whole process, called acceleration coefficients, r_1, r_2 are two random numbers in the range [0,1] selected uniformly for each dimension at each iteration, $v_i(t)$ is the physical component, $r_1 c_1 \left(x_{pbest_i} - x_i(t)\right)$ is the cognitive component, where $c_1$controls the cognitive behavior of the particle, and $r_2 c_2 \left(x_{gbest} - x_i(t)\right)$ is the social component, where c_2 controls the social behavior of the particle.

Algorithm 1. The standard PSO algorithm
1: Calculate the fitness function of each particle.
2: Update x_{pbest_i} and x_{gbest}.
3: Update the position (1) and velocity (2) of each particle
4: If the stop criterion is not met, go to (2), else x_{gbest} is the best position

Due to the drawback of PSO mentioned, though MOPSO has good global optimization performance, the optimization efficiency of MOPSO sometimes obviously decreases near Pareto solutions. To solve the premature convergence issue, many effective particles updating strategies are incorporated into MOPSO to improve the diversity in the Pareto-optimal solutions, such as the hyper-grid approach [3], σ-method with clustering [4] and Sigma method [5]. On the other hand, the hybridization approach with other methods used to get a balance between exploitation and exploration. For example, the authors in [6] presented a co-evolutionary PSO algorithm associating with the artificial immune principle. In the proposed algorithm, the whole swarm was divided into two kinds of sub-swarms consisting of one elite subpopulation and several normal subpopulations. The best individual of each normal subpopulation will be memorized into the elite subpopulation during the evolution process.

The authors in [7] proposed a hybridization approach, which is a combination of PSO and Differential Evolution (DE-PSO). They use a random displacement strategy to increase the exploration capacity and at the same time to accelerate the algorithm convergence, by using operators of DE algorithm. So, three strategies have been used: DE Updating Strategy (DEUS), Random Updating Strategy (RUS) and PSO Updating Strategy (PSOUS). The author in [8] presented an algorithm based on a master-slave model, in which a population consists of one master swarm and several slave swarms. The slave swarms execute a single PSO independently to maintain the diversity of the particles, while the master swarm evolves based on its own knowledge and on the knowledge of the slave swarms. Also, the authors in [9] proposed a hybrid approach between Speed-constrained Multi-objective Particle Swarm Optimization (SMPSO) algorithm and the concept of tabu search in order to remedy to the premature convergence and to improve the exploitation in search space.

B. Island Models

Islands Model (IM) is a multi-population approach used by evolutionary algorithms. It is inspired by the punctuated equilibrium model. The first studies of the IM connected with the genetic algorithms showed a high efficiency and an equivalence between the exploitation and the exploration in the search space. In general, IM divides the population into subpopulations. Each population is considered an island and is isolated from other islands during the evolutionary process. The islands evolve independently for a while, and then, a process of migration is carried out by the islands which can exchange candidate solutions.

This migration process has the advantage of promoting or encouraging the development of subpopulations by introducing new characteristics through the exchange of candidate solutions between the islands. The fundamental aspect of IM is the collaboration between the islands, it is carried out by a periodic migration process. This process uses a strategy for which candidate solutions are transferred from one subpopulation to another. Migration allows subpopulations to help others by introducing new information in the form of migratory points.

There are several parameters specifying an island model, like:

- Islands number: represents the number of subpopulations in the model;
- Migration topology: represents the communication structure of the model;
- Migratory frequency: determines how often migration occurs;
- Migratory rate: determines how many individuals migrate from a subpopulation to another;
- Synchronization type: represents the type of synchronization for the migration process (synchronous or asynchronous);
- Migratory policy: defines which individuals will be removed and replaced when migration occurs;
- Migratory flow: defines the path of the emigrants inside the communication structure.

In the literature, several EA-based methods have integrated the island model concepts in order to improve the population diversity during the search and establish a balance between exploration and exploitation, Island based Genetic Algorithm [10], Island based Differential Evolution [11], [12]. Moreover, the parameters sensitivity of island models such as islands number, migration interval and migration frequency have been studied to show their effect on the convergence and their optimal values to improve the algorithms performance [13].

Most of time the island models are using a static way by introducing their parameters such as the immigrant replacement policy, the topology of the communication among subpopulations, and the synchronous or asynchronous nature of the connection among subpopulations. The authors proposed a dynamic model to improve its performance [14], [15].

III. Proposed Approach

In order to improve the MOPSO algorithm, we propose to combine it with the island model [9] in order to maintain diversity by using a set of parameters. Then, we use Variable Neighborhood Search (VNS) technique [16] as a local approach for its simplicity to parameterize and its effectiveness. For these reasons, this technique is chosen as a local mechanism to enhance the exploitation in each island where it is applied for every pbest solution of each particle in the sub-swarm.

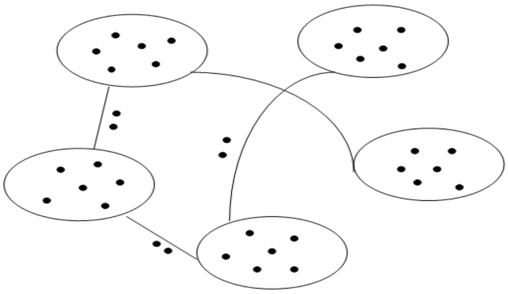

Fig. 1 IMOPSO algorithm structure

Step 1. The model sub-divided the swarm into a set of swarms connected by T topology. In the initialization phase of each sub-swarm, the particles are distributed randomly in the search space for each dimension. Also, the velocity is initialized in each sub-swarm too. Initial value for the pbest_i is set to the particle position. Another set named archive can be defined in order to store the obtained non-dominated solutions. Due to the presence of an archive, the best solutions are preserved during iterations.

Step 2. In the update phase, each particle i has a position defined by $x_i = x_{i1}, x_{i2}, \dots, x_{id}, \dots, x_{iD}$ and a velocity defined by $v_i = (v_{i1}, v_{i2}, \dots, v_{id}, \dots v_{iD})$. Its movement is stochastically adjusted according to the influence of its best solution and of the best solution of its neighbors. In fact, the velocity and position of each particle i are updated using the two equations (1) and (2).

Step 3. Update pbest and gbest: In the PSO algorithm, the personnel experience of the particle is captured in the pbest attribute, which corresponds to the best performance attained so far by it in its movement. In our approach, we use the following strategy for every current solution, we choose a set of neighborhoods $\mathrm{N_k}, \mathrm{k} = 1, \dots, \mathrm{k_{max}}$ generated randomly. Then, we compare the current solution of the i-th particle to the pbest solution of each particle of the chosen neighbor: if the pbest of the current solution dominates it, we kept it, otherwise it is replaced by the pbest of the considered neighbor.

To choose the leader, select the gbest solution using a binary tournament based on the crowding value of the leaders. The maximum size of the leaders set is fixed equal to the size of the swarm. After each iteration, the leaders archive, and its crowding values are updated. If the size of leaders archive is larger than the maximum allowable size, only the best leaders are retained based on their crowding value. The rest of the leaders are eliminated.

Algorithm 2. The Detailed Procedure of A Personal Guide

For the d-th dimension of the i-th particle and j-th particle,
Choose a random solution
while N_k, *k=1,...,* k_{max}
Generate a *pbest*
solution of the i-th particle at random from the k-th neighborhood.
a_{k+1} = *pbest-i th*
a_k = *pbest-j th*
if a_{k+1} dominated by a_k
select a_k and replaces the a_{k+1} and get a new pbest solution otherwise continue with the another *k-th* neighbor;
$k \leftarrow k + 1$.
End While
End For

Step 4. Mutation Operator: Two mutation operators: Polynomial mutation is applied to each particle in the sub-swarm.

Algorithm 3. Island-MOPSO

Randomly initialize the islands (sub-swarms) connected by T topology
While the sub-swarm has not converged and the maximum iterations number has not been reached.
Evaluate the fitness of each particle in the sub-swarm
Update pbest using VNS and gbest
Perform mutation operation
Perform migration operation
End While
Send the best solutions to the other island neighbor
Replace the randomly chosen solutions
Update the archive
End

Step 5. Migration Operator: Migration is determined by Random Ring topology T which is totally dynamic and it is defined randomly each time the migration has to occur. The islands connect to each other as a unidirectional-connected graph. The edge between each two islands represented the feasible path between an island and its neighboring island. When the migration frequency is reached, in our case it is defined by the iterations number $\mathrm{U_i}$, the migration rate occurs by determining the number of migrant particles to send and receive between the islands based on migration topology. The migration policy $\mathrm{R_m}$ is another process in migration responsible for selecting the migrant particles to be exchanged among islands. The random-random migration policy is used to send the random solutions from one island to its neighboring island by means of replacing the random particles.

IV. Experimental Test and Results

The proposed approach is compared to several multiobjective evolutionary algorithms to test its performance: OMOPSO [17], SMPSO [18], and genetic algorithm NSGA II [19]. We selected problems out of two well-known benchmark suites, namely the ZDT [20] and the DTLZ [21] problems.

These problems cover various features like concave geometries, disconnected Pareto-fronts, biased and multimodal problems (Table I). The problem dimensions were chosen as to create an interesting complexity level between 3 and 30 decisions variables. The algorithms are run for a population size of 100, the archive size is 100 solutions. The iterations number is equal to 500 iterations for PSO algorithms (the equivalent of the stopping criterion for NSGA II algorithm is a maximum evaluations number of the objective function equal to 50,000). We analyze the results quality of Pareto fronts after 30 independent runs of each function.

The parameters related to island model are intensively studied to measure their effect on the convergence and diversity of the proposed approach. The number of islands used with various {I_m= 2, 5, 10} values has been treated in the three cases with random ring topology. The other parameters have been also studied such as the migration frequency represented by the iteration number= 100 iterations and the migration rate (R_m)= 10% and the migration policy is random-random strategy. Also, the neighbor size used by VNS mechanism is = 3 in our experiments.

TABLE I
PROPERTIES OF THE TEST FUNCTIONS

Test functions	Objectives number	Proprieties
ZDT1	2	Convex
ZDT2	2	concave
ZDT3	2	Discontinue fronts, multimodal
ZDT4	2	Convex, multimodal
ZDT6	2	Concave, multimodal biased
DTLZ1	3	Linear, multimodal
DTLZ2	3	Concave
DTLZ3	3	Concave, multimodal
DTLZ4	3	Concave, biased
DTLZ5	3	Uni-modal
DTLZ6	3	Uni-modal
DTLZ7	3	Discontinue fronts, multimodal

TABLE II
RESULTS OF HV INDICATOR (I_M = 5)

Test functions		IMOPSO	OMOPS	SMPSO	NSGA II	p-value
	Best	2.98 e-1	2.82 e-1	3.45 e-1	4.74e-1	
ZDT1	Average	**4.95e-1**	5.71e -1	5.12 e-1	8.14 e-1	+
	Worst	6.53 e-1	6.7 4e-1	6.32 e-1	8.23 e-1	
	Best	2.58 e-1	2.67 e-1	3.21 e-1	7.02 e-2	
ZDT2	Average	**2.23e-1**	3.28 e-1	3.27 e-1	2.36 e-1	+
	Worst	4.1 e-1	4.57 e-1	4.2 e-1	4.51 e-1	
	Best	4.36 e-1	4.87 e-1	5.22 e-1	8.69 e-2	
ZDT3	Average	**4.98 e-1**	5.21 e-1	5.75 e-1	6.14 e -1	+
	Worst	6.24 e-1	7.26 e-1	6.32 e-1	6.75 e-1	
	Best	---	---	4.54 e-2	6.45 e-2	
ZDT4	Average	0.00	0.0	6.23 e-2	3.87 e-1	-
	Worst	---	---	6.85 e-2	4.43 e-1	
	Best	3.28 e-1	3.42 e-1	2.77 e-1	---	
ZDT6	Average	**3.52 e-1**	3.97 e-1	3.61 e-1	0.00	+
	Worst	4.56 e-1	4.72 e-1	4.36 e-1	---	
	Best	2.87 e-1	2.99 e-1	2.63 e-1	---	
DTLZ1	Average	**3.11e-1**	3.22 e-1	4.01 e-1	0.00	+
	Worst	3.95 e-1	4.73 e-1	4.82 e-1	---	
	Best	6.51 e-2	6.56 e-2	6.88 e-2	7.25 e-2	
DTLZ2	Average	**1.99e-1**	2.12e -1	3.27 e-1	3.83 e-1	+
	Worst	2.59 e-1	3.41 e-1	3.64 e-1	4.17 e-1	
	Best	--	--	1.98 e-1	--	
DTLZ3	Average	--	0.00	2.02 e-1	0.00	-
	Worst	--	--	3.61 e-1	--	
	Best	1.47e-1	1.91 e-1	1.97 e-1	3.68 e-1	
DTLZ4	Average	**1.94 e-1**	2.09 e-1	2.41 e-1	4.45 e-1	+
	Worst	3.26 e-1	3.52 e-1	3.24 e-1	4.89 e-1	
	Best	1.73 e-1	1.94 e-1	2.00 e.1	2.82 e-1	
DTLZ5	Average	**2.1 e-1**	2.18e-1	2.23 e-1	3.12 e-1	+
	Worst	2.84 e-1	3.25 e-1	2.93 e-1	4.34 e-1	
	Best	8.76 e-1	1.87 e-1	9.25 e-2	8.54 e-2	
DTLZ6	Average	**2.03 e-1**	2.12 e-1	6.21 e-1	1.58 e-1	+
	Worst	2.79 e-1	3.89 e-1	6.82 e-1	2.36 e-1	
	Best	2.84 e-1	2.86 e-1	2.96 e-1	3.24 e-1	
DTLZ7	Average	**3.25e-1**	3.36 e-1	2.37 e-1	3.35e-1	+
	Worst	3.92 e-1	4.06 e-1	2.88 e-1	3.96 e-1	

In order to compare the algorithms and since the evaluation of a multi-objective approach requires metrics and indicators in different scopes (such as diversity, distribution, convergence), we have used:

- Hypervolume (HV) [21]: This metric measures the hypervolume of the portion of the objective space that is weakly dominated by an approximation set A and is to be maximized. Here, we consider the hypervolume difference to a reference set R where smaller values correspond to higher quality.
- Spread (Spr) [22]: This indicator measures the extent of spread of a set of non-dominated solutions. It considers the Euclidean distance between consecutive solutions on average and extreme distances. Smaller values correspond to higher quality.
- Epsilon indicator ($I^1_{\epsilon+}$) [23]: This indicator is a measure of the smallest distance that would be necessary to translate every solution in a PS so that it dominates the optimal PF of the problem. It depends on the solutions range of values, but smaller values are better.

There is one indicator related to convergence (Epsilon), one to diversity (Spread), and one to the both convergence and diversity (Hypervolume). To compute these metrics and indicators, we have used the jMetal software.

Furthermore, to give a better analysis of the results, a statistical test is required in order to provide confident results. This included a testing phase which allows us to perform a multiple comparison of samples. We have used the multi-comparative function provided by MATLAB for that purpose. We always put down a confidence level of 95% (i.e., significance level of 5% or p-value below 0.05) in the statistical tests. Successful tests are marked with "+"symbols in the last column in all the tables containing the results; conversely, "− "means that no statistical confidence was found (p-value > 0.05).

TABLE III
RESULTS OF EPSILON INDICATOR (I_M = 5)

Test functions		IMOPSO	OMOPSO	SMPSO	NSGA II	p-value
	Best	4.74 e-3	6.05 e-3	1.46 e-2	5.32 e-3	
ZDT1	Average	**4.96 e-3**	6.42 e-3	1.82 e-2	5.72 e-3	+
	Worst	6.83 e-3	8.62 e-2	4.47 e-1	3.65 e-2	
	Best	4.67 e-3	5.84 e-3	4.68 e-3	4.84 e-3	
ZDT2	Average	**5.32 e-3**	6.14e-3	5.35 e-3	5.66 e-3	+
	Worst	7.47 e-3	9.32 e-3	8.56 e-3	8.96 e-3	
	Best	3.81 e-3	7.53 e-3	4.23 e-3	4.85 e-3	
ZDT3	Average	**4.44e-3**	1.32 e-2	5. e-3	6.12 e-3	+
	Worst	6.39 e-3	2.26 e-2	6.34 e-3	3.78 e-2	
	Best	4.21	5.23	7.09 e-1	6.42 e-3	
ZDT4	Average	4.43	5.64	7.51 e-1	**7.88e-3**	+
	Worst	4.98	6.3	8.49 e-1	4.31 e-2	
	Best	3.84 e-3	4.16 e-3	4.22 e-3	4.63 e-3	
ZDT6	Average	**4.38 e-3**	4.58 e-3	4.6 e-3	4.92 e-3	+
	Worst	6.15 e-2	6.47 e-2	6.29 e-2	7.39 e-2	
	Best	1.43	1.65	2.76 e-1	3.32 e-1	
DTLZ1	Average	1.89	1.92	3.25 e-1	**3.72 e-3**	+
	Worst	1.95	1.98	8.48 e-1	4.58 e-2	
	Best	2.84 e-3	4.43 e-3	7.25 e-2	4.23 e-3	
DTLZ2	Average	**5.23 e-3**	6.72 e-3	1.41 e-1	5.83 e-3	+
	Worst	3.86 e-2	4.35 e-2	1.74 e-1	4.53 e-2	
	Best	4.87	8.52	4.24 e-1	4.55 e-3	
DTLZ3	Average	8.61	8.73	7.59 e-1	6.57 e-3	+
	Worst	8.96	8.98	8.42 e-1	8.69 e-3	
	Best	1.97 e-2	2.84 e-2	8.36 e-2	3.68 e-2	
DTLZ4	Average	**2.74e-2**	3.23e-2	1.45 e-1	5.72 e-2	+
	Worst	3.34 e-2	4.39 e-2	2.39 e-1	6.83 e-2	
	Best	3.67 e-3	3.74 e-2	4.26 e-3	4.59 e-3	
DTLZ5	Average	**4.44e-3**	6.54 e-2	4.98 e-3	5.34 e-3	+
	Worst	3.65 e-2	7.67 e-2	3.57 e-2	6.41 e-2	
	Best	3.81 e-3	4.74 e-3	4.13 e3	4.25 e-3	
DTLZ6	Average	**4.23 e-3**	5.31e-3	4.65 e-3	5.23 e-3	+
	Worst	5.82 e-3	6.26 e-3	6.37 e-3	6.52 e-3	
	Best	4.35 e-3	5.43 e-3	5.59 e-3	4.87 e-3	
DTLZ7	Average	**4.92e-3**	7.12e-3	7.41 e-3	5.54 e-3	+
	Worst	6.78 e-3	5.47 e-2	6.67 e-2	6.22 e-2	

Table II shows the average, best and worst values of hypervolume for over 30 independent runs obtained by all of the four algorithms. For further analysis, the ANOVA test is performed to evaluate the significant difference between the samples. The values in bold show the best result found for each test function using the different methods. It can estimate that the proposed method IMOPSO has achieved the best performance in hypervolume for nine among the 12 test functions. We can observe that the proposed approach gives the best results for the distribution solutions in Pareto fronts and it is superior to OMOPSO for all 12 of the test functions, superior to SMPSO expect in DTLZ1 and DTLZ3, and finally superior to NSGAII expect in ZDT6 and DTLZ1.

Table III shows the average, best and worst values of $I^1_{\epsilon+}$ indicator, the modified IMOPSO provides good results for the following problems: ZDT1, ZDT2, ZDT3, ZDT6, DTLZ2, DTLZ4, DTLZ5, DTLZD6 and DTLZ7 which shows that the convergence is assured by using a local search technique: VNS for updating the pbest of each particle. It can be seen clearly that OMOPSO has some difficulties in solving ZDT4, DTLZ1 and DTLZ3 at the convergence level but the IMOPSO which is based on several islands could decrease these difficulties. Here, NSGAII shows a successful result to solve those problems compared to PSO algorithms.

TABLE IV
RESULTS OF SPREAD INDICATOR ($I_M = 5$)

Test functions		IMOPSO	OMOPSO	SMPSO	NSGA II	p-value
	Best	5.36 e-3	5.26 e-2	5.41 e-2	5.23e-2	
ZDT1	Average	**6.43e-3**	6.85e-2	6.76e-2	6.78e-2	+
	Worst	8.54 e-3	4.47 e-2	4.25 e-1	3.28e-1	
	Best	3.62 e-3	5.56 e-2	5.33 e-2	5.72e-2	
ZDT2	Average	**5.37e-3**	6.42e-2	5.84e-2	6.24e-2	+
	Worst	4.78 e-2	6.84 e-2	6.13 e-2	8.85e-2	
	Best	2.56 e-2	1.66 e-1	2.46 e-1	7.22e-1	
ZDT3	Average	**3.16e-2**	3.38e-1	3.47e-1	7.71e-1	+
	Worst	2.09 e-1	4.73 e-1	4.65 e-1	8.37e-1	
	Best	5.22 e-2	6.58 e-2	8.28 e-2	2.15e-1	
ZDT4	Average	**7.52e-2**	7.76e-2	9.1 e-2	2.81e-1	+
	Worst	2.42 e-1	2.47 e-1	3.38 e-1	3.79e-1	
	Best	4.57 e-1	5.37 e-1	4.28 e-1	4.73e-1	
ZDT6	Average	**6.72e-1**	6.81e-1	6.25e-1	6.94e-1	+
	Worst	6.89 e-1	7.26 e-1	6.97 e-1	7.59e-1	
	Best	4.23 e-1	4.89 e-1	4.34 e-1	5.52e-1	
DTLZ1	Average	**5.26 e-1**	5.69e-1	6.54e-1	6.78e-1	+
	Worst	5.86 e-1	6.93 e-1	7.31 e-1	7.46e-1	
	Best	3.34 e-1	4.28 e-1	5.37 e-1	5.33e-1	
DTLZ2	Average	**5.63e-1**	5.72e-1	6.42e-1	6.23e-1	+
	Worst	6.78 e-1	7.14 e-1	7.05 e-1	7.58e-1	
	Best	5.35 e-1	6.17 e-1	6.52 e-1	7.24e-1	
DTLZ3	Average	**7.29e-1**	7.37e-1	7.56e-1	8.21e-1	+
	Worst	8.42 e-1	8.58 e-1	8.2 e-1	8.87e-1	
	Best	4.21 e-1	5.78 e-1	4.69 e-1	5.2 e-1	
DTLZ4	Average	**5.67 e-1**	6.54 e-1	5.59e-1	6.75e-1	+
	Worst	6.36 e-1	7.84 e-1	7.21 e-1	7.44e-1	
	Best	9.06 e-2	1.54e-1	1.92 e-1	1.25e-1	
DTLZ5	Average	**1.51 e-1**	2.25e-1	2.35e-1	2.84e-1	+
	Worst	3.17 e-1	3.58 e-1	3.77 e-1	3.49e-1	
	Best	8.68 e-2	1.12 e-1	1.83 e-1	4.21e-1	
DTLZ6	Average	**1.19 e-1**	1.57 e-1	2.76e-1	5.07e-1	+
	Worst	3. 72e-1	3.65 e-1	4.38 e-1	6.27e-1	
	Best	4.53 e-1	5.31 e-1	5.43 e-1	5.33e-1	
DTLZ7	Average	**5.01 e-1**	5.86 e-1	6.31e-1	7.22e-1	+
	Worst	6.24 e-1	6.48 e-1	7.19 e-1	7.79e-1	

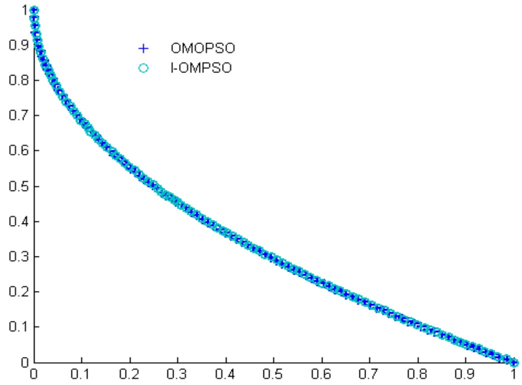

Fig. 2 Pareto fonts for ZDT1 test function

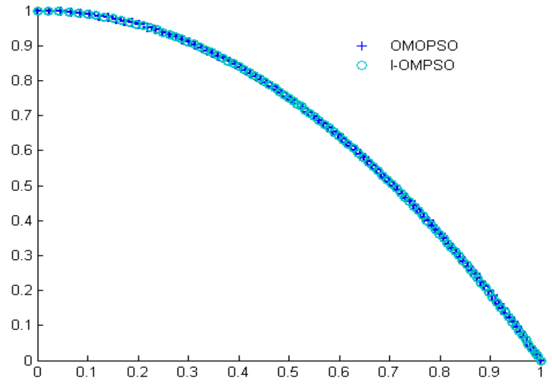

Fig. 3 Pareto fonts for ZDT2 test function

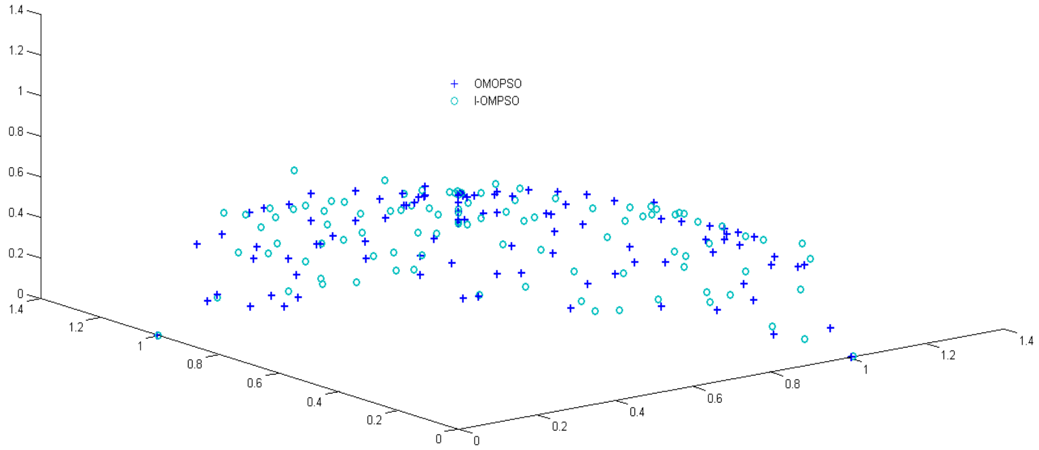

Fig. 4 Pareto fronts for DTLZ2 test function

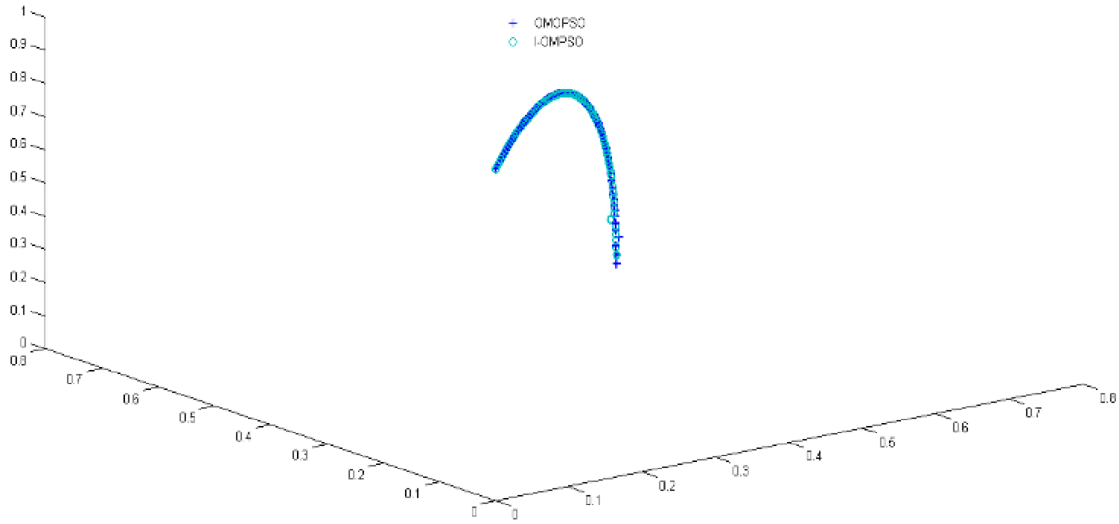

Fig. 5 Pareto fronts for DTLZ6 test function

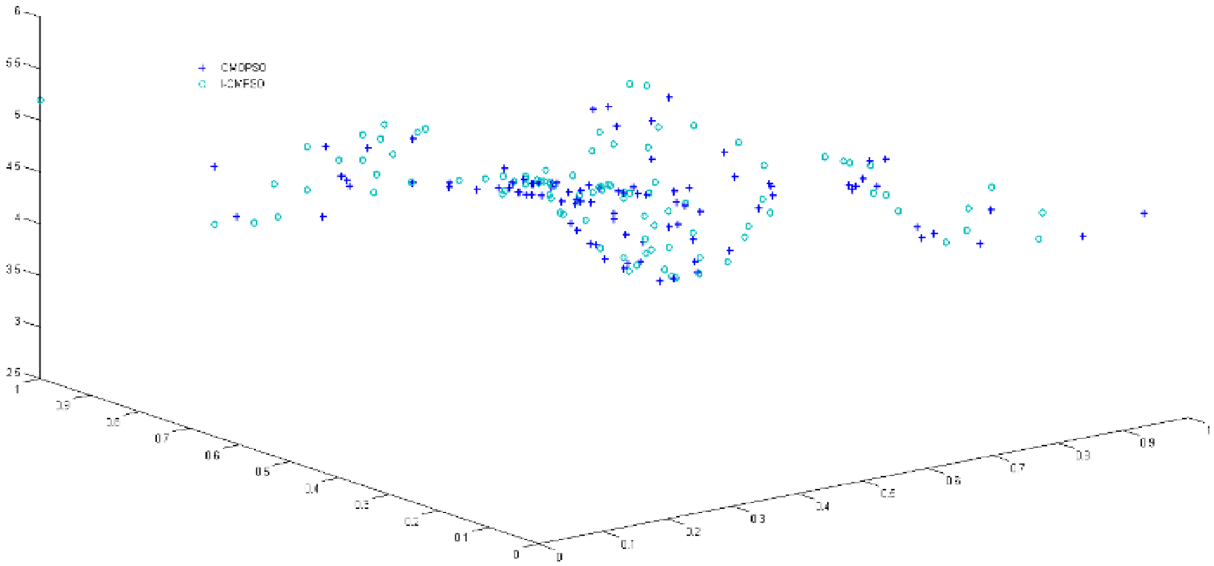

Fig. 6 Pareto fronts for DTLZ7 test function

Table IV shows the results of average, best and worst values for the Spread indicator where the smallest values represent the best results. The fronts found for the most test functions show that modified IMOPSO is the best in terms of non-dominated solution distribution along the Pareto front. Indeed, it gives smaller values, thus showing better quality of the Spread indicator. It can be observed that the use of migration operator could improve the particles diffusion over the search space. Furthermore, the impact of using the island models parameters such as the random ring topology, the frequency rate and migration rate yield a higher amount of good solutions at the level of Pareto fronts. Also, the use of VNS technique allows the good exploitation into each island based on the neighborhood for each best local solutions of the particles. The first conclusion is a big achievement for the proposed approach by using island model and a local search technique to improve the MOPSO diversity.

Looking at the tables, it can be noticed that the proposed approach and the comparative algorithms have significant influence on the values of every metric and indicator, since the p value is always much smaller than 0.05. Moreover, they have significant influence at a confidence level.

To illustrate the working of the IOMPSO and OMOPSO algorithms, we have included in Figs. 2-6 the obtained approximations to the optimal Pareto front on the ZDT1, ZDT2, DTLZ2, DTLZ6 and DTLZ7 test functions. It can be seen from the observations of the approximate Pareto fronts found by IOMPSO improved in term of diversity. The choice of VNS is done in order to maintain the intensification into each island, also the migration operator could be a good mechanism to improve the diversity.

V. Conclusions

In this paper, we presented an approach for the multiobjective PSO algorithm which consists to combine it with the island models strategy and a local search technique in order to get a better spread into the swarm. The island models concepts have been embedded into the MOPSO algorithm. The particles are divided into several sub-swarms called islands. After such generations, the migration process occurs using random-ring topology and a random-random migration policy is performed in order to exchange the leaders among the islands. On other hands, the used of VNS mechanism for the pbest solution could maintain a good exploitation of the search space. Experiments on a series of ZDT and DTLZ test functions have been conducted to compare the proposed method with several state-of-the-art MOPSO algorithms and NSGA II. The results show that the proposed approach gives a better result in almost test functions.

References

[1] CA. Coello, DA. Veldhuizen and GB. Lamont. Evolutionary algorithms for solving multi-objective problems. IEEE, 2002.

[2] J. Kennedy and R. C. Eberhart. Particle Swarm Optimization. Proceedings of the IEEE International Conference on Neural Networks, IEEE Press. pp. 1942-1948, Perth, Australia, 1995.

[3] C. A. Coello Coello, G. T. Pulido, M. S. Lechuga, Handling multiple objectives with particle swarm optimization, IEEE Trans. Evol. Comput. 8, pp 256–279, 2004.

[4] P. K. Tripathi, S. Bandyopadhya and S. K. Pal. Adaptive Multi-objective Particle Swarm Optimization Algorithm. IEEE Congress on Evolutionary Computation,2007.

[5] J. Teich Mostaghim. Strategies for finding good local guides in multi-objective particle swarm optimization (MOPSO), in: IEEE 2003 Swarm Intelligence Symposium, 2003.

[6] Z.-H. Liu, J. Zhang, S.-W. Zhou, X.-H. Li, and K. Liu, "Coevolutionary particle swarm optimization using AIS and its application in multiparameter estimation of PMSM," IEEE Transactions on Cybernetics, vol. 43, no. 6, pp. 1921–1935, 2013.

[7] Z-H. Zhan and J. Zhang, Discrete Particle Swarm Optimization for Multiple Destination Routing Problems, EvoWorkshops, LNCS 5484, Springer, 2009, pp. 117-122.

[8] K. E. Parspoulos. Parallel cooperative micro-particle swarm optimization: A master salve model. Journal of Applied Soft Computing, volume12, pp 3552-3579, 2012.

[9] H. Abadlia, N. Smairi and K. Ghedira. A new proposal for a multi-objective technique using SMPSO and Tabu Search. 15thIEEE/ACIS

International Conference on Computer and Information Science, pp 1-6, Japan, 2016.
[10] H. T. T. Thein. Island model based differential evolution algorithm for neural network training. Advances in Computer Science: An International Journal, 3(1), 2014.
[11] R. Michel and M. Middendorf. An island model based ant system with look ahead for the shortest super sequence problem. In Parallel problem solving from nature PPSN V (pp. 692–701). Springer, 1998.
[12] M. Tomassini. Spatially structured evolutionnary algorithms: Artificial evolution in space time. Secaus, NJ, USA: Spring-Verlag New York, 2005.
[13] F. Lardeux and A. Goeffon. A Dynamic Island-Based Genetic Algorithms Framework. SEAL '10: 156-165, 2010.
[14] C. Candan, A. Goeffon, F. Lardeux and F. Saubien. A Dynamic Island Model for Adaptive Operator Selection. GECCO'12, 2012.
[15] P. Hansen and N. Mladenovic. An introduction to variable neighborhood search. Springer, 1999.
[16] M. Reyes Sierra and C. A. Coello Coello. Improving PSO-Based Multi-objective Optimization Using Crowding, Mutation and ϙ-Dominance. In Evolutionary MultiCriterion Optimization (EMO 2005), LNCS 3410, pages 505–519, 2005.
[17] J. J. Durillo, J. García-Nieto, A. J. Nebro, C. A. C. Coello, F. Luna and E. Alba. Multi-Objective Particle Swarm Optimizers: An Experimental Comparison. 5th International Conference, Nantes, France, pp.495-509, 2009.
[18] K. Deb, S. Agarwal, A. Pratap, and T. Meyarivan, A fast and elitist multi-objective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6(2), pp. 182–197, 2002.
[19] E. Zitzler, K. Deb, and L. Thiele. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. Evolutionary Computation, 8(2):173–195, 2000.
[20] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable Test Problems for Evolutionary Multiobjective Optimization. In A. Abraham, L. Jain, and R. Goldberg, editors, Evolutionary Multiobjective Optimization. Theoretical Advances and Applications, pages 105–145. Springer, 2005.
[21] K. Deb. Multi-objective optimization using evolutionary algorithms. Wiley, Hoboken, 2001.
[22] Knowles J, Thiele L, Zitzler E. A tutorial on the performance assessment of stochastic multiobjective optimizers. Tech. Rep. 214, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, 2006.
[23] J. J. Durillo, A. J. Nebro and E. Alba. jMetal framework for multiobjective optimization: design and architecture. In: IEEE conference on evolutionary computation CEC-2010, pp 4138– 4325.