SMPSO: A New PSO-based Metaheuristic for Multi-objective Optimization

A.J. Nebro, J.J. Durillo, J. García-Nieto, C.A. Coello Coello, F. Luna and E. Alba

Abstract—**In this work, we present a new multi-objective particle swarm optimization algorithm (PSO) characterized by the use of a strategy to limit the velocity of the particles. The proposed approach, called Speed-constrained Multi-objective PSO (SMPSO) allows to produce new effective particle positions in those cases in which the velocity becomes too high. Other features of SMPSO include the use of polynomial mutation as a turbulence factor and an external archive to store the non-dominated solutions found during the search. Our proposed approach is compared with respect to five multi-objective metaheuristics representative of the state-of-the-art in the area. For the comparison, two different criteria are adopted: the quality of the resulting approximation sets and the convergence speed to the Pareto front. The experiments carried out indicate that SMPSO obtains remarkable results in terms of both, accuracy and speed.**

I. Introduction

Particle Swarm Opmitization (PSO) is a bio-inspired metaheuristic mimicking the social behavior of bird flocking or fish schooling [8] which has become very popular to solve multi-objective optimization problems. Since the first attempt proposed by Moore and Chapman in 1999 to extend it to multi-objective optimization [10], more than thirty different proposals of Multi-Objective Optimization PSOs (MOPSOs) have been reported in the specialized literature [15].

In [5], we analyzed the performance of six MOPSOs representative of the state-of-the-art and concluded that all of them are unable to solve some multi-frontal problems satisfactorily (e.g., ZDT4). We studied this issue in more depth, and found that the velocity of the particles in these algorithms can become too high, resulting in erratic movements towards the upper and lower limits of the positions of the particles. This is an example of the so-called "swarm explosion" [1], and it can be prevented by using a velocity constriction mechanism [1]. Thus, taking OMOPSO (the most salient algorithm from the MOPSOs studied in [5]) as our starting point, we developed a new algorithm called SMPSO (Speed-constrained Multi-objective PSO), which incorporates a velocity constriction procedure. The preliminary experiments carried out by the authors, showed that SMPSO could solve the problems where the other MOPSOs had difficulties. SMPSO was also compared with respect to NSGA-II [3] and OMOPSO [14], achieving competitive results.

In this paper our motivation is twofold. First, we want to compare SMPSO with five state-of-the-art multi-objective optimization algorithms in a typical study consisting in assessing the performance of the techniques by applying three quality indicators (additive espilon, spread, and hypervolume) after 25,000 function evaluations. The selected algorithms are: two genetic algorithms, NSGA-II [3] and SPEA2 [17]; a scatter search approach, AbYSS [13]; a cellular genetic algorithm, MOCell [12], and OMOPSO [14]. Our second goal is to study the convergence speed of SMPSO to determine how fast it is compared with the five aforementioned algorithms. We follow the approach taken in [11], in which a stopping condition based on achieving the 98% of the hypervolume of the true Pareto front is adopted. The remainder of this paper is structured as follows. Section II describes the SMPSO algorithm. The next section includes a brief description of each algorithm considered in this study. In Section IV, we analyze the obtained resuls. Finally, Section V presents our conclusions and some possible paths for future work.

II. Description of our SMPSO

In this section, we describe our approach detailing the velocity constriction mechanism, the pseudocode of SMPSO, and the differences with respect to OMOPSO (the algorithm which SMPSO is based on).

A. Velocity Constriction Approach

In a PSO algorithm, each potential solution to the problem is called *particle* and the population of solutions is called *swarm*. A basic PSO updates the particle $\vec{x}_i$ at the generation t with the formula:

$$\vec{x}_i(t) = \vec{x}_i(t-1) + \vec{v}_i(t) \quad (1)$$

where the factor $\vec{v}_i(t)$ is known as velocity and is given by

$$\vec{v}_i(t) = w \cdot \vec{v}_i(t-1) + C_1 \cdot r_1 \cdot (\vec{x}_{p_i} - \vec{x}_i) + C_2 \cdot r_2 \cdot (\vec{x}_{g_i} - \vec{x}_i) \quad (2)$$

In this formula, $\vec{x}_{p_i}$ is the best solution that $\vec{x}_i$ has viewed, $\vec{x}_{g_i}$ is the best particle (also known as the *leader*) that the entire swarm has viewed, w is the inertia weight of the particle and controls the trade-off between global and local experience, r_1

A.J. Nebro, J.J Durillo, J. García-Nieto, F. Luna, and E. Alba are with the Departamento de Lenguajes y Ciencias de la Computación, University of Málaga, Spain (e-mail: {antonio,durillo,gnieto,flv,eat}@lcc.uma.es). Carlos A. Coello Coello is with the Department of Computer Science, CINVESTAV-IPN, Mexico (e-mail: ccoello@cs.cinvestav.mx). He is also affiliated to the UMI-LAFMIA 3175 CNRS.

This work has been partially funded by the "Consejería de Innovación, Ciencia y Empresa", Junta de Andalucía under contract P07-TIC-03044 DIRICOM project, `http://diricom.lcc.uma.es`.

J.J. Durillo is supported by grant AP-2006-03349 from the Spanish Ministry of Education and Science. F. Luna acknowledges support from the Spanish Ministry of Education and Science and FEDER under contract TIN2005-08818-C04-01 (the OPLINK project).

and r_2 are two uniformly distributed random numbers in the range $[0, 1]$, and C_1 and C_2 are specific parameters which control the effect of the personal and global best particles.

In order to control the particle's velocity, instead of using upper and lower parameter values which limit the step size of the velocity, we have adopted a *constriction coefficient* (Eq. 3) obtained from the constriction factor χ originally developed by Clerc and Kennedy (Eq. 2) in [1].

$$\chi = \frac{2}{2 - \varphi - \sqrt{\varphi^2 - 4\varphi}} \tag{3}$$

where

$$\varphi = \begin{cases} C_1 + C_2 & \text{if } C_1 + C_2 > 4 \\ 1 & \text{if } C_1 + C_2 \leq 4 \end{cases} \tag{4}$$

In addition, we introduce a mechanism in such a way that the accumulated velocity of each variable j (in each particle) is further bounded by means of the following *velocity constriction* equation:

$$v_{i,j}(t) = \begin{cases} delta_j & \text{if } v_{i,j}(t) > delta_j \\ -delta_j & \text{if } v_{i,j}(t) \leq -delta_j \\ v_{i,j}(t) & \text{otherwise} \end{cases} \tag{5}$$

where

$$delta_j = \frac{(upper\_limit_j - lower\_limit_j)}{2} \tag{6}$$

Summarizing the procedure, the velocity of the particles are calculated according to Eq. 2; the resulting velocity is then multiplied by the constriction factor (Eq. 3) and the resulting value is constrained by using Eq. 5.

To illustrate the effect of the adopted velocity constriction scheme, we include in Fig. 1 (left) the trace of the velocity of the second variable in one uniformly random chosen particle in OMOPSO when facing the solution of ZDT4 in 250 iterations [5]. We can observe that the velocity values alternate from very high to very low values in some points of the execution. Let us note that the limits of the second variable in ZDT4 are $[-5, +5]$, and the velocity takes values higher than ± 20. As a consequence, the position of the particle variable takes limit values, which does not contribute to the search. Fig. 1 (right) depicts the same trace in SMPSO, where we can observe that the velocity is constrained within $[-5, +5]$, and thus, the particle is effectively moving through the search space.

B. Pseudocode of the Proposed Algorithm

Algorithm 1 shows the pseudocode of SMPSO. It starts by initializing the swarm (Line 1), which includes the position, velocity, and p (individual best) of the particles. The leaders archive is initialized with the non-dominated solutions in the swarm (Line 2). Then, the main loop of the algorithm is executed for a maximum number of iterations. The velocities and positions of the particles are calculated first (Lines 5 and 6) and a mutation operator is applied with a given probability (Line 7). The resulting particles are evaluated (Line 8) and both the particle's memory and the leaders archive are updated (Lines 9 and 10). The algorithm returns the leaders archive as the approximation set found (Line 13).

Algorithm 1 Pseudocode of our proposed SMPSO

```
1: initializeSwarm()
2: initializeLeadersArchive()
3: generation = 0
4: while generation < maxGenerations do
5:    computeSpeed() // Eqs. 2 - 6
6:    updatePosition() // Eq. 1
7:    mutation() // Turbulence
8:    evaluation()
9:    updateLeadersArchive()
10:   updateParticlesMemory()
11:   generation ++
12: end while
13: returnLeadersArchive()
```

Given that the leaders archive can become full, we use the crowding distance of NSGA-II to decide which particles must remain in it. As turbulence operator, we have chosen the polynomial mutation operator [2]. To choose the *pbest* particle to apply Eq. 2, we take two solutions from the leaders archive randomly and the one having the largest crowding distance to its nearest neighbors in the archive is selected.

C. Differences with respect to OMOPSO

Given that OMOPSO is the departure point for our SMPSO, it is worth mentioning the actual differences between the two algorithms. In the study presented in [5], both algorithms differed only in the velocity constriction mechanism and in the ranges of the values that C_1 and C_2 may take. In OMOPSO, they are random numbers in the range $[1.5, 2.0]$. However, in order to apply Eq. 4, in SMPSO the range was changed to $[1.5, 2.5]$ because in the former case the value of Eq. 4 is always 1.

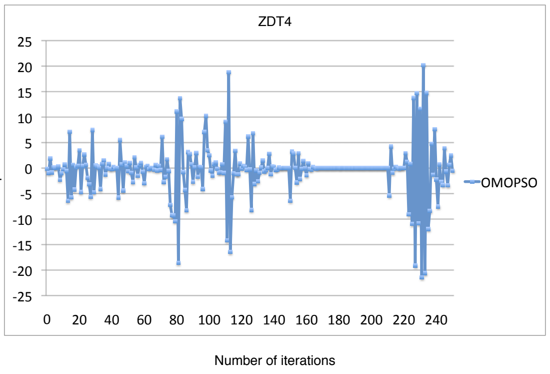
After applying Eq. 1, OMOPSO checks whether the resulting positions are out of the bounds of the variables of the problem. In that case, the positions are assigned the corresponding upper or lower bound value; additionally, the direction of the velocity is reversed by multiplying it by -1.0. We have conducted some preliminary experiments with SMPSO in this sense but, instead of reversing, reducing the velocity by multiplying it by values between 0.1 and 0.001. We have achieved slightly better results using a value of 0.001.

The last difference is related to the use of mutation operators. OMOPSO applies a combination of uniform and non-uniform mutation to the particle swarm (uniform mutation to the first 30% of the swarm, non-uniform to the next 30%, and no mutation to the rest of the particles). In SMPSO, a polynomial mutation [2] is applied to the 15% of the particles.

III. Description of the Evaluated Algorithms

In this section, we briefly describe the five algorithms that we have considered to evaluate the behavior of our SMPSO. We have used the implementation of these algorithms provided by the jMetal framework [6].

The NSGA-II algorithm was proposed by Deb *et al.* [3]. It is a genetic algorithm based on obtaining a new population from the original one by applying the typical genetic operators



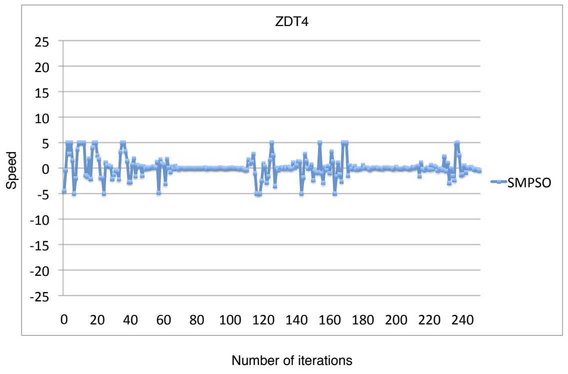


Fig. 1. Velocity value of the second variable of OMOPSO (left) and SMPSO (right) when solving ZDT4.

(selection, crossover, and mutation); then, the individuals in the two populations are sorted according to their rank, and the best solutions are chosen to create a new population. In case of having to select some individuals with the same rank, a density estimation based on measuring the crowding distance to the surrounding individuals belonging to the same rank is used to get the most promising solutions.

SPEA2 was proposed by Zitler *et al.* in [17]. In this algorithm, each individual has a fitness value that is the sum of its strength raw fitness plus a density estimation. The algorithm applies the selection, crossover, and mutation operators to fill an archive of individuals; then, the non-dominated individuals of both the original population and the archive are copied into a new population. If the number of non-dominated individuals is greater than the population size, a truncation operator based on calculating the distances to the k-th nearest neighbor is used. This way, the individuals having the minimum distance to any other individual are chosen.

OMOPSO (Optimized MOPSO, Coello *et al.* [14]) is a multi-objective particle swarm optimization algorithm whose main features include the use of an external archive based on the crowding distance from NSGA-II to filter out leader solutions and the use of mutation operators to accelerate the convergence of the swarm. OMOPSO has also an archive to store the best solutions found during the search. This archive makes use of the concept of ϵ-dominance to limit the number of solutions stored. We consider here a variant of OMOPSO consisting in considering the population containing the leader solutions instead of this archive.

MOCell (Nebro *et al.* [12]) is a cellular genetic algorithm (cGA). Like OMOPSO, it includes an external archive to store the non-dominated solutions found so far. This archive makes use of the crowding distance of NSGA-II to keep diversity. We have used here an asynchronous version of MOCell, called aMOCell4 in [12].

AbYSS is an adaptation of the *scatter search* metaheuristic to the multi-objective domain proposed by Nebro *et al.* in [13]. This algorithm uses an external archive similar to the one employed by OMOPSO and MOCell. The algorithm incorporates operators of the evolutionary algorithms domain, including polynomial mutation and simulated binary crossover in the improvement and solution combination methods, respectively.

TABLE I
PARAMETERIZATION (L = INDIVIDUAL LENGTH)

Parameterization used in NSGA-II [3]	
Population Size	100 individuals
Selection of Parents	binary tournament + binary tournament
Recombination	simulated binary, $p_c = 0.9$
Mutation	polynomial, $p_m = 1.0/L$
Parameterization used in SPEA2 [17]	
Population Size	100 individuals
Selection of Parents	binary tournament + binary tournament
Recombination	simulated binary, $p_c = 0.9$
Mutation	polynomial, $p_m = 1.0/L$
Parameterization used in OMOPSO [14]	
Swarm size	100 particles
Mutation	uniform + non-uniform
Leaders Size	100
Parameterization used in AbYSS [13]	
Population Size	20 individuals
Reference Set Size	10 + 10
Recombination	simulated binary, $p_c = 1.0$
Mutation (local search)	polynomial, $p_m = 1.0/L$
Archive Size	100 individuals
Parameterization used in MOCell [12]	
Population Size	100 individuals (10×10)
Neighborhood	1-hop neighbours (8 surrounding solutions)
Selection of Parents	binary tournament + binary tournament
Recombination	simulated binary, $p_c = 0.9$
Mutation	polynomial, $p_m = 1.0/L$
Archive Size	100 individuals
Parameterization used in SMPSO	
Swarm size	100 particles
Mutation	polynomial, $p_m = 1.0/L$
Archive Size	100 individuals

IV. Experimentation

The benchmarking problems chosen to evaluate the six algorithms have been the ZDT (Zitzler-Deb-Thiele) [16] and DTLZ (Deb-Thiele-Laumanns-Zitzler) [4] test suites. The DTLZ problems have been used with their bi-objective formulation. For assessing the performance of the algorithms, we have considered three quality indicators: additive unary epsilon indicator ($I^1_{\epsilon+}$) [9], spread (Δ) [3], and hypervolume (HV) [18]. The two first indicators measure, respectively, the convergence and the diversity of the resulting Pareto fronts, while the last one measures both convergence and diversity. To measure the convergence speed, the algorithms are executed until reaching a maximum of one million function evaluations, and they stop when they find an approximation set whose hypervolume is equal or greater than the 98% of the hypervolume of the true Pareto front of the problem being solved [11]. It is considered that an algorithm succeeded when it stops before performing one million evaluations.

We describe next the parameter settings of the algorithms and the experimentation methodology adopted.

A. Parameterization

We have chosen a set of parameter settings to guarantee a fair comparison among the algorithms. All GAs (NSGA-II, SPEA2, and MOCell) use an internal population of size equal to 100; the size of the archive is also 100 in SPEA2, OMOPSO, AbYSS, MOCell, and SMPSO. OMOPSO and SMPSO have been configured with 100 particles. For AbYSS, the population and the reference set have a size of 20 solutions.

In the GAs, we have used SBX and polynomial mutation [2] as the operators for the crossover and mutation operators, respectively. The distribution indexes for both operators are $\eta_c = 20$ and $\eta_m = 20$, respectively. The crossover probability is $p_c = 0.9$ and the mutation probability is $p_m = 1/L$, where L is the number of decision variables. AbYSS uses polynomial mutation in the improvement method and SBX in the solution combination method. OMOPSO applies a combination of uniform and non-uniform mutation, while SMPSO uses polynomial mutation, as commented in Section II-C. A summary of the parameter settings is included in Table I.

B. Methodology

To assess the search capabilities of the algorithms, we have performed 100 independent runs of each experiment, and we have obtained the median, $\tilde{x}$, and interquartile range, IQR, as measures of location (or central tendency) and statistical dispersion, respectively. Since we are dealing with stochastic algorithms and we want to provide the results with statistical confidence, we have also included a testing phase which allows us to perform a multiple comparison of samples [7]. We have used the `multcompare` function provided by Matlab$^{\copyright}$ for that purpose. We always consider a confidence level of 95% (i.e., significance level of 5% or *p*-value below 0.05) in the statistical tests. For the sake of a better understanding, the best result for each problem has a gray colored background and the second best one has a clearer grey background.

C. Evaluation

In this section, we analyze first the quality of the obtained Pareto fronts after 25,000 function evaluations, to further discuss the convergence speed results.

1) Quality of the Approximated Fronts: Table II includes the $I_{\epsilon}^{1}+$ values of the resulting approximated fronts computed by all the algorithms. The grey colored background in the SMPSO column clearly shows that this new proposal can be considered as the algorithm that has produced the fronts closest to the true Pareto front. Indeed, SMPSO has reached the lowest (best) values in nine out of twelve problems, and it has obtained the second best $I_{\epsilon}^{1}+$ values in the other three problems. The values yielded by MOCell make this algorithm the second best performer concerning this indicator. An additional interesting fact that can be drawn from Table II emerges from the particular comparison between SMPSO and OMOPSO, our new proposal and its predecessor. This comparison, points out that the velocity constriction mechanism has endowed SMPSO with improved search capabilities that has allowed for a better convergence towards the true Pareto front (it has always obtained better indicator values than OMOPSO). This occurs specially in the ZDT4, DTLZ1, and DTLZ3 problems, where the resulting fronts of SMPSO have reached $I_{\epsilon}^{1}+$ values that are several orders of magnitude lower that those of OMOPSO. The problems in which no statistical confidence can be assured in the $I_{\epsilon}^{1}+$ indicator by the tests carried out are shown in Table III. We can observe that most differences between each pair of algorithms have been significant according to this indicator. As to the comparison between OMOPSO and SMPSO, the tests cannot assure differences in problems ZDT6 and DTLZ6. The results of the Δ indicator are presented in Table IV. The values show that SMPSO has been the algorithm that better distributed the solutions along the Pareto front, reaching the best (lowest) indicator values in seven out of the twelve studied problems. MOCell has been again the second best algorithm in terms of the solution spread out (three lowest values). The comparison between SMPSO and OMOPSO is of particular interest here as well, since the improvements are noticeable. Indeed, the indicator values of SMPSO have always been lower than those of OMOPSO being one order of magnitude lower in problems ZDT1, ZDT4, and DTLZ1. It is therefore clear that the velocity constriction mechanism has also allowed SMPSO to obtain better distributed approximation sets.

Table V summarizes the results of the statistical tests for the Δ indicator. As with the $I_{\epsilon}^{1}+$ indicator, it includes the problems in which no statistical differences can be found. In this case, results have also had statistical confidence in most cases, except between AbYSS and MOCell: considering this indicator, statistical differences between these two algorithms do not exist in seven out of the twelve problems evaluated.

The last indicator used to assess the quality of the resulting Pareto fronts is the Hypervolume (see Table VI). The "•' symbols in the table mean that all the non-dominated solutions of the obtained fronts have been so far away from the true Pareto front. These types of solutions have to be no longer considered to compute the HV values because the results would be otherwise unreliable. As a measure of both convergence and diversity, the reached HV values have confirmed the obtained values by the two previous indicators: the algorithms with better values in $I_{\epsilon}^{1}+$ and Δ are also the ones with better values in HV. Thus, SMPSO has been the best algorithm also with respect to this indicator, and has obtained the best (higher) values in eight out of the twelve problems evaluated. MOCell has been the second best algorithm: it has obtained the best value in two out of the twelve problems evaluated and the second best value in four problems. As to the comparison

TABLE II
MEDIAN AND INTERQUARTILE RANGE OF THE EPSILON INDICATOR ($I^1_{\epsilon+}$)

	NSGA-II	SPEA2	OMOPSO	AbYSS	MOCell	SMPSO
Problem	$\tilde{x}_{IQR}$	$\tilde{x}_{IQR}$	$\tilde{x}_{IQR}$	$\tilde{x}_{IQR}$	$\tilde{x}_{IQR}$	$\tilde{x}_{IQR}$
ZDT1	$1.37e-02_{3.0e-03}$	$8.69e-03_{1.1e-03}$	$6.36e-03_{5.1e-04}$	$7.72e-03_{1.8e-03}$	$6.23e-03_{4.1e-04}$	$5.39e-03_{2.6e-04}$
ZDT2	$1.28e-02_{2.3e-03}$	$8.73e-03_{1.4e-03}$	$6.19e-03_{5.4e-04}$	$7.10e-03_{1.6e-03}$	$5.57e-03_{3.0e-04}$	$5.33e-03_{1.7e-04}$
ZDT3	$8.13e-03_{1.9e-03}$	$9.72e-03_{1.9e-03}$	$1.32e-02_{7.7e-03}$	$6.10e-03_{3.1e-01}$	$5.66e-03_{7.5e-04}$	$5.10e-03_{7.3e-04}$
ZDT4	$1.49e-02_{3.0e-03}$	$3.42e-02_{7.9e-02}$	$5.79e+00_{4.3e+00}$	$1.14e-02_{4.2e-03}$	$8.17e-03_{2.3e-03}$	$6.02e-03_{4.3e-04}$
ZDT6	$1.47e-02_{2.8e-03}$	$2.42e-02_{5.2e-03}$	$4.65e-03_{4.2e-04}$	$5.06e-03_{3.9e-04}$	$6.53e-03_{5.6e-04}$	$4.43e-03_{3.0e-04}$
DTLZ1	$7.13e-03_{1.6e-03}$	$5.89e-03_{2.8e-03}$	$1.92e+01_{1.1e+01}$	$5.85e-03_{5.5e-03}$	$4.02e-03_{1.5e-03}$	$2.97e-03_{2.0e-04}$
DTLZ2	$1.11e-02_{2.7e-03}$	$7.34e-03_{1.1e-03}$	$6.72e-03_{9.1e-04}$	$5.39e-03_{4.6e-04}$	$5.09e-03_{2.8e-04}$	$5.17e-03_{2.6e-04}$
DTLZ3	$1.04e+00_{1.2e+00}$	$2.28e+00_{1.9e+00}$	$8.86e+01_{9.5e+01}$	$1.66e+00_{1.6e+00}$	$7.91e-01_{1.0e+00}$	$5.39e-03_{8.5e-04}$
DTLZ4	$1.13e-02_{9.9e-01}$	$7.66e-03_{9.9e-01}$	$3.18e-02_{1.0e-02}$	$5.39e-03_{3.0e-04}$	$5.74e-03_{9.9e-01}$	$5.39e-03_{3.6e-04}$
DTLZ5	$1.05e-02_{2.5e-03}$	$7.47e-03_{1.2e-03}$	$6.62e-03_{8.9e-04}$	$5.36e-03_{5.2e-04}$	$5.08e-03_{3.2e-04}$	$5.24e-03_{3.0e-04}$
DTLZ6	$4.39e-02_{3.4e-02}$	$3.03e-01_{5.3e-02}$	$5.36e-03_{4.8e-04}$	$9.50e-02_{4.7e-02}$	$4.16e-02_{3.8e-02}$	$5.08e-03_{2.5e-04}$
DTLZ7	$1.04e-02_{2.8e-03}$	$9.09e-03_{1.4e-03}$	$7.13e-03_{6.8e-04}$	$5.51e-03_{9.6e-04}$	$5.19e-03_{1.0e-03}$	$4.95e-03_{2.8e-04}$

TABLE III
NON-SUCCESSFUL STATISTICAL TEST FOR THE EPSILON INDICATOR ($I^1_{\epsilon+}$)

NSGA-II	ZDT3, ZDT4 DTLZ1, DTLZ4, DTLZ7	- DTLZ4	ZDT3 DTLZ3	- DTLZ6	- -
SPEA2		ZDT3 DTLZ2	ZDT1 DTLZ1, DTLZ3	- -	- -
OMOPSO			- DTLZ7	ZDT1 -	ZDT6 DTLZ6
AbYSS				- DTLZ2, DTLZ7	- DTLZ2, DTLZ4, DTLZ5
MOCell					- DTLZ2, DTLZ5
	SPEA2	OMOPSO	AbYSS	MOCell	SMPSO

TABLE IV
MEDIAN AND INTERQUARTILE RANGE OF THE SPREAD INDICATOR

	NSGA-II	SPEA2	OMOPSO	AbYSS	MOCell	SMPSO
Problem	$\tilde{x}_{IQR}$	$\tilde{x}_{IQR}$	$\tilde{x}_{IQR}$	$\tilde{x}_{IQR}$	$\tilde{x}_{IQR}$	$\tilde{x}_{IQR}$
ZDT1	$3.70e-01_{4.2e-02}$	$1.52e-01_{2.2e-02}$	$1.00e-01_{1.4e-02}$	$1.05e-01_{2.0e-02}$	$7.64e-02_{1.3e-02}$	$7.34e-02_{1.7e-02}$
ZDT2	$3.81e-01_{4.7e-02}$	$1.55e-01_{2.7e-02}$	$9.45e-02_{1.8e-02}$	$1.07e-01_{1.8e-02}$	$7.67e-02_{1.4e-02}$	$7.14e-02_{1.5e-02}$
ZDT3	$7.47e-01_{1.8e-02}$	$7.10e-01_{7.5e-03}$	$7.35e-01_{5.2e-02}$	$7.09e-01_{9.7e-03}$	$7.04e-01_{6.2e-03}$	$7.05e-01_{6.3e-03}$
ZDT4	$4.02e-01_{5.8e-02}$	$2.72e-01_{1.6e-01}$	$8.78e-01_{5.2e-02}$	$1.27e-01_{3.5e-02}$	$1.10e-01_{2.8e-02}$	$9.14e-02_{1.7e-02}$
ZDT6	$3.56e-01_{3.6e-02}$	$2.28e-01_{2.5e-02}$	$8.78e-02_{1.2e+00}$	$8.99e-02_{1.4e-02}$	$9.33e-02_{1.3e-02}$	$7.02e-02_{4.4e-02}$
DTLZ1	$4.03e-01_{6.1e-02}$	$1.81e-01_{9.8e-02}$	$7.77e-01_{1.1e-01}$	$1.40e-01_{1.7e-01}$	$1.05e-01_{3.6e-02}$	$6.88e-02_{1.3e-02}$
DTLZ2	$3.84e-01_{3.8e-02}$	$1.48e-01_{1.6e-02}$	$1.81e-01_{2.3e-02}$	$1.09e-01_{1.9e-02}$	$1.08e-01_{1.7e-02}$	$1.28e-01_{1.8e-02}$
DTLZ3	$9.53e-01_{1.6e-01}$	$1.07e+00_{1.6e-01}$	$7.90e-01_{1.1e-01}$	$7.55e-01_{4.5e-01}$	$7.45e-01_{5.5e-01}$	$1.35e-01_{3.1e-02}$
DTLZ4	$3.95e-01_{6.4e-01}$	$1.48e-01_{8.6e-01}$	$6.77e-01_{7.9e-02}$	$1.08e-01_{1.8e-02}$	$1.23e-01_{9.0e-01}$	$1.14e-01_{1.9e-02}$
DTLZ5	$3.79e-01_{4.0e-02}$	$1.50e-01_{1.9e-02}$	$1.77e-01_{2.6e-02}$	$1.10e-01_{2.0e-02}$	$1.09e-01_{1.7e-02}$	$1.27e-01_{2.0e-02}$
DTLZ6	$8.64e-01_{3.0e-01}$	$8.25e-01_{9.3e-02}$	$1.18e-01_{1.7e-02}$	$2.31e-01_{6.3e-02}$	$1.50e-01_{4.3e-02}$	$1.10e-01_{2.0e-02}$
DTLZ7	$6.23e-01_{2.5e-02}$	$5.44e-01_{1.3e-02}$	$5.21e-01_{6.8e-03}$	$5.19e-01_{1.3e-03}$	$5.19e-01_{2.9e-02}$	$5.19e-01_{5.1e-04}$

TABLE V
NON-SUCCESSFUL STATISTICAL TEST FOR THE SPREAD INDICATOR.

NSGA-II	- DTLZ4, DTLZ6	ZDT3 DTLZ4	- -	- -	- -
SPEA2		- -	ZDT3 DTLZ1	- -	- -
OMOPSO			ZDT1 DTLZ3, DTLZ7	ZDT6 DTLZ3, DTLZ7	- DTLZ6, DTLZ7
AbYSS				ZDT3, ZDT4, ZDT6 DTLZ2, DTLZ3, DTLZ5, DTLZ7	ZDT3, ZDT6 DTLZ4
MOCell					ZDT1, ZDT2, ZDT3, ZDT6 -
	SPEA2	OMOPSO	AbYSS	MOCell	SMPSO

between SMPSO and OMOPSO, the last one has been better than the former in all the evaluated problems except for DTLZ6, in which OMOPSO has yielded the best value. The results of the statistical tests for the HV are presented in Table VII. Proceeding as in the two previous indicators, we have included the problems in which the statistical tests have not been successful. Focusing on the differences between SMPSO and the others algorithms, we observe that the tests have shown statistical confidence in most cases.

Summarizing this section, we can state that the velocity constriction mechanism allows SMPSO to be the best algorithm in the context of the problems, quality indicators, and parameterizations considered in our study. It is also noticeable that neither NSGA-II nor SPEA2 have produced approximated

TABLE VI

MEDIAN AND INTERQUARTILE RANGE OF THE HYPERVOLUME INDICATOR

Problem	NSGA-II $\tilde{x}_{IQR}$	SPEA2 $\tilde{x}_{IQR}$	OMOPSO $\tilde{x}_{IQR}$	AbYSS $\tilde{x}_{IQR}$	MOCell $\tilde{x}_{IQR}$	SMPSO $\tilde{x}_{IQR}$
ZDT1	$6.59e-01_{4.4e-04}$	$6.60e-01_{3.9e-04}$	$6.61e-01_{1.5e-04}$	$6.61e-01_{3.2e-04}$	$6.61e-01_{2.5e-04}$	$6.62e-01_{5.3e-05}$
ZDT2	$3.26e-01_{4.3e-04}$	$3.26e-01_{8.1e-04}$	$3.28e-01_{2.5e-04}$	$3.28e-01_{2.8e-04}$	$3.28e-01_{4.3e-04}$	$3.29e-01_{4.7e-05}$
ZDT3	$5.15e-01_{2.3e-04}$	$5.14e-01_{3.6e-04}$	$5.10e-01_{3.8e-03}$	$5.16e-01_{3.5e-03}$	$5.15e-01_{3.1e-04}$	$5.16e-01_{1.2e-04}$
ZDT4	$6.56e-01_{4.5e-03}$	$6.51e-01_{1.2e-02}$	●	$6.55e-01_{6.0e-03}$	$6.59e-01_{3.0e-03}$	$6.61e-01_{1.6e-04}$
ZDT6	$3.88e-01_{2.3e-03}$	$3.79e-01_{3.6e-03}$	$4.01e-01_{1.5e-04}$	$4.00e-01_{1.9e-04}$	$3.97e-01_{1.1e-03}$	$4.01e-01_{7.9e-05}$
DTLZ1	$4.88e-01_{5.5e-03}$	$4.89e-01_{6.2e-03}$	●	$4.86e-01_{1.7e-02}$	$4.91e-01_{3.8e-03}$	$4.94e-01_{1.6e-04}$
DTLZ2	$2.11e-01_{3.1e-04}$	$2.12e-01_{1.7e-04}$	$2.10e-01_{4.5e-04}$	$2.12e-01_{6.5e-05}$	$2.12e-01_{4.5e-05}$	$2.12e-01_{1.3e-04}$
DTLZ3	●	●	●	●	$0.00e+00_{1.7e-01}$	$2.12e-01_{2.8e-04}$
DTLZ4	$2.09e-01_{2.1e-01}$	$2.10e-01_{2.1e-01}$	$1.96e-01_{6.1e-03}$	$2.11e-01_{5.9e-05}$	$2.11e-01_{2.1e-01}$	$2.10e-01_{1.3e-04}$
DTLZ5	$2.11e-01_{3.5e-04}$	$2.12e-01_{1.7e-04}$	$2.11e-01_{5.4e-04}$	$2.12e-01_{6.8e-05}$	$2.12e-01_{3.1e-05}$	$2.12e-01_{1.3e-04}$
DTLZ6	$1.75e-01_{3.6e-02}$	$9.02e-03_{1.4e-02}$	$2.12e-01_{4.4e-05}$	$1.11e-01_{4.1e-02}$	$1.61e-01_{4.2e-02}$	$2.12e-01_{8.4e-05}$
DTLZ7	$3.33e-01_{2.1e-04}$	$3.34e-01_{2.2e-04}$	$3.34e-01_{3.2e-04}$	$3.34e-01_{7.8e-05}$	$3.34e-01_{9.5e-05}$	$3.34e-01_{3.1e-05}$

TABLE VII

NON-SUCCESSFUL STATISTICAL TEST FOR THE HYPERVOLUME INDICATOR

NSGA-II	ZDT2, ZDT4 DTLZ1, DTLZ4, DTLZ7	DTLZ4 -	ZDT3 DTLZ1	ZDT3 DTLZ6	-
SPEA2		DTLZ7 -	DTLZ1 -	- -	- -
OMOPSO			ZDT1 ZDT2	DTLZ7 -	ZDT6 -
AbYSS				ZDT2, ZDT3 DTLZ2, DTLZ5	- -
MOCell					DLTZ4 -
	SPEA2	OMOPSO	AbYSS	MOCell	SMPSO

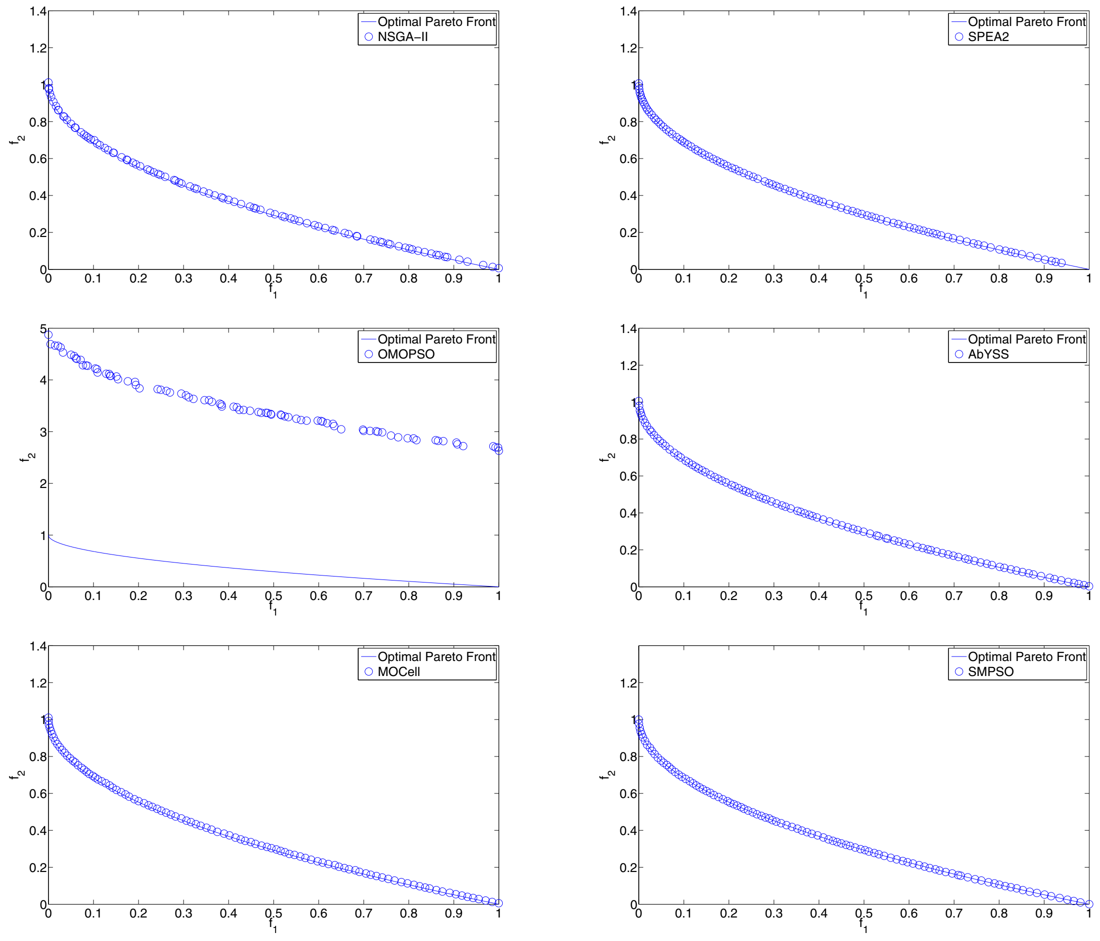


Fig. 2. The ZDT4 problem solved using the compared algorithms: NSGA-II, SPEA2, OMOPSO, AbYSS, MOCell, and SMPSO.

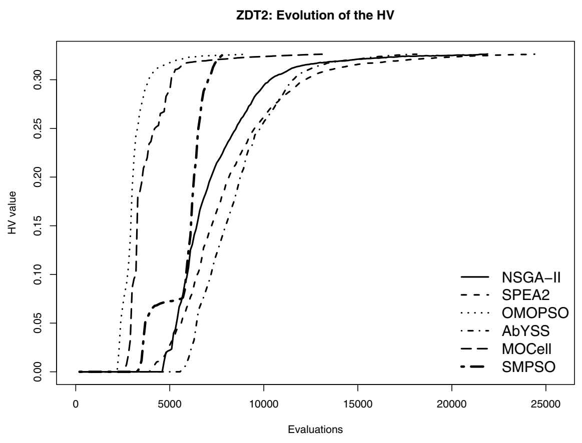
Pareto fronts whose quality indicators are the best or the second best in any of the evaluated problems.

To illustrate the performance of the six algorithms, we have included in Fig. 2 the obtained approximations to the optimal Pareto front on the ZDT4 problem. As we can observe, those obtained by SMPSO, MOCell, and AbYSS are the best choices in terms of convergence and spread to the optimal front, whereas the solution set obtained by OMOPSO has not converged and thus it is the worst approximation. As to the solutions of NSGA-II and SPEA2, both have converged towards the Pareto front but the distribution of points is not as good as in SMPSO, MOCell, and AbYSS.

2) Convergence Speed: Table VIII includes, for each multi-objective problem, the median and the IQR of the number of evaluations required by all the algorithms to reach 98% the HV value of the true Pareto front. The "•" symbol in some cells means that the given algorithm has been not able to reach such HV value in some (if not all) of the independent runs performed.

The first conclusion that can be drawn from the values in Table VIII is that the new proposal, SMPSO, is always either the fastest (in five out of the twelve problems) or the second fastest (the remaining seven problems) algorithm in reaching the target HV value. SMPSO has never failed at meeting the convergence criterion (i.e., never required more than $1,000,000$ evaluations to reach 98% of the true Pareto front), while the others fail in at least one problem. This fact clearly shows the enhanced exploration capabilities of SMPSO and its robustness for solving the studied problems. Of particular relevance is the comparison of OMOPSO and SMPSO, since the former has not been able to approximate fronts with the target HV in three problems (ZDT4, DTLZ1, and DTLZ3), whereas the latter can effectively achieve it. The results also show that, under this experimental setup, the two most well-known algorithms in the literature, NSGA-II and SPEA2, are outperformed by new proposals such as AbYSS, MOCell, and particularly SMPSO. Indeed, both NSGA-II and SPEA2 require a number of function evaluation that is one order of magnitude greater than the fastest algorithms for eight out of the twelve considered problems. Table IX summarizes the results of the statistical tests considering the number of evaluations required by the different solvers. We see that the differences between SMPSO and the other algorithms are statistically significant in most of the problems evaluated. As to the comparison between SMPSO and OMOPSO, no differences exist only in three out of the twelve evaluated problems according to the significance levels considered.

Although the results reported in this section are relevant, it is worth showing how the HV evolves during the execution of the algorithms. We have included in Fig. 3 the HV values of the algorithms when solving the ZDT2 and DTLZ7 problems, recorded every 100 function evaluations. Focusing on OMOPSO and SMPSO, the fastest algorithms on these problems, we can observe that OMOPSO is the fastest technique in finding approximation sets having a HV greater than 0; however, SMPSO arrives first to 98% of the HV of the



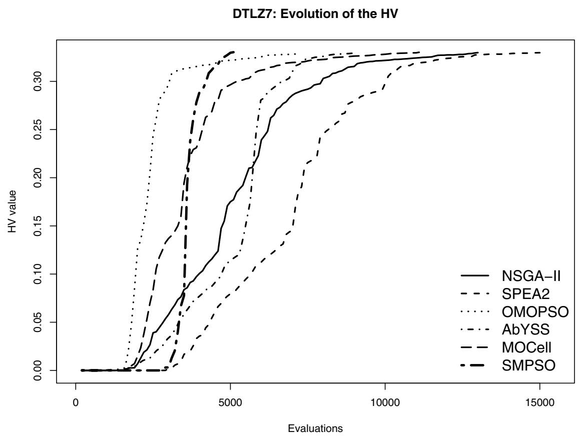


Fig. 3. Evolution of the HV during the different generations carried out in the problems ZDT2 (top) and DTLZ7 (bottom).

Pareto front. This means that using a lower percent of the HV as stopping condition, OMOPSO would be the fastest solver. It is also interesting to observe the behavior of SMPSO: in ZDT2 there is a region where the HV is stalled in a fixed value (between 2500 and 6000 evaluations), but then it converges rapidly. Similarly, SMPSO is among the slowest techniques in yielding a HV greater than zero, but then it arrives to the stopping condition in a few hundreds of evaluations. This suggests lines for further research, in which the behavior of the algorithms during their search process could be studied. We could envision, for example, an algorithm behaving like OMOPSO at the beginning of the search and then commuting to SMPSO, with the idea of combining the best of them.

V. Conclusions and Future Work

We have described SMPSO, a new multi-objective PSO algorithm which incorporates a velocity constriction mechanism. By using it, the maximum velocity of the particles is limited with the aim of enhancing the search capability of the technique. The new proposal has been evaluated using two benchmark families, ZDT and DTLZ, and it has been compared against five state-of-the-art multi-objective optimization algorithms: NSGA-II, SPEA2, OMOPSO, AbYSS,

TABLE VIII
MEDIAN AND IQR OF THE NUMBER OF EVALUATIONS REQUIRED BY THE SOLVERS TO REACH 98% THE HV OF THE TRUE PARETO FRONT.

Problem	NSGA-II $\tilde{x}_{IQR}$	SPEA2 $\tilde{x}_{IQR}$	OMOPSO $\tilde{x}_{IQR}$	AbYSS $\tilde{x}_{IQR}$	MOCell $\tilde{x}_{IQR}$	SMPSO $\tilde{x}_{IQR}$
ZDT1	1.435e+04 $_{8.0e+02}$	1.600e+04 $_{1.1e+03}$	6.800e+03 $_{2.0e+03}$	1.370e+04 $_{1.6e+03}$	1.300e+04 $_{1.2e+03}$	7.500e+03 $_{3.0e+03}$
ZDT2	2.430e+04 $_{1.8e+03}$	2.480e+04 $_{1.9e+03}$	8.900e+03 $_{3.6e+03}$	1.710e+04 $_{2.8e+03}$	1.170e+04 $_{4.0e+03}$	8.200e+03 $_{3.4e+03}$
ZDT3	1.270e+04 $_{9.0e+02}$	1.520e+04 $_{1.0e+03}$	9.850e+03 $_{2.7e+03}$	1.270e+04 $_{2.0e+03}$	1.300e+04 $_{1.3e+03}$	1.160e+04 $_{5.4e+03}$
ZDT4	2.130e+04 $_{5.0e+03}$	2.520e+04 $_{6.0e+03}$	●	2.285e+04 $_{1.1e+04}$	1.635e+04 $_{5.0e+03}$	4.700e+03 $_{1.2e+03}$
ZDT6	2.880e+04 $_{1.2e+03}$	3.335e+04 $_{1.0e+03}$	2.800e+03 $_{1.5e+03}$	1.560e+04 $_{1.2e+03}$	2.090e+04 $_{1.3e+03}$	3.750e+03 $_{1.3e+03}$
DTLZ1	2.515e+04 $_{9.4e+03}$	2.400e+04 $_{7.5e+03}$	●	2.375e+04 $_{1.2e+04}$	2.015e+04 $_{7.7e+03}$	5.300e+03 $_{2.0e+03}$
DTLZ2	8.100e+03 $_{1.2e+03}$	7.400e+03 $_{8.0e+02}$	8.200e+03 $_{3.1e+03}$	4.700e+03 $_{9.0e+02}$	5.600e+03 $_{9.0e+02}$	4.800e+03 $_{1.3e+03}$
DTLZ3	1.180e+05 $_{5.7e+04}$	1.000e+05 $_{3.0e+04}$	●	1.194e+05 $_{7.5e+04}$	6.735e+04 $_{2.3e+04}$	8.500e+03 $_{4.2e+03}$
DTLZ4	8.500e+03 $_{1.4e+03}$	7.800e+03 $_{5.0e+05}$	1.255e+04 $_{3.8e+03}$	4.800e+03 $_{7.5e+02}$	●	5.400e+03 $_{1.4e+03}$
DTLZ5	7.950e+03 $_{1.1e+03}$	7.500e+03 $_{7.0e+02}$	8.450e+03 $_{2.9e+03}$	4.650e+03 $_{8.0e+02}$	5.800e+03 $_{8.5e+02}$	5.250e+03 $_{1.4e+03}$
DTLZ6	●	●	4.100e+03 $_{1.5e+03}$	●	●	8.150e+03 $_{5.1e+03}$
DTLZ7	1.360e+04 $_{1.0e+03}$	1.585e+04 $_{1.1e+03}$	6.150e+03 $_{2.6e+03}$	1.060e+04 $_{1.7e+03}$	1.110e+04 $_{1.6e+05}$	5.500e+03 $_{2.4e+03}$

TABLE IX
NON-SUCCESSFUL STATISTICAL TEST FOR THE NUMBER OF EVALUATIONS REQUIRED BY THE SOLVERS TO REACH THE 98% THE HV OF THE TRUE PARETO FRONT

NSGA-II	ZDT2 DTLZ1, DTLZ3, DTLZ4, DTLZ6	- DTLZ4	ZDT1, ZDT3, ZDT4 DTLZ1, DTLZ3	ZDT3 DTLZ4, DTLZ7	ZDT3 -
SPEA2		- DTLZ6	ZDT4 DTLZ1, DTLZ3	- DTLZ4, DTLZ6	- -
OMOPSO			- -	- DTLZ4	ZDT1, ZDT2 DTLZ7
AbYSS				ZDT3 DTLZ6, DTLZ7	ZDT3 DTLZ2, DTLZ4
MOCell					ZDT3 -
	SPEA2	OMOPSO	AbYSS	MOCell	SMPSO

and MOCell. The results have shown that SMPSO overcomes the limitations of the algorithms it has been compared with. Indeed, in the context of the experiments carried out, it is the most salient technique in terms of the quality of the approximations to the Pareto front found, and it is also the fastest converging towards the Pareto front in most of the studied problems. As part of our future work, we plan to study new schemes for updating the velocity of the particles, and to apply SMPSO to other benchmarks composed of rotated problems as well as of problems with more than two objectives.

REFERENCES

[1] M. Clerc and J. Kennedy. The particle swarm - explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation*, 6(1):58–73, 2002.

[2] K. Deb. *Multi-Objective Optimization Using Evolutionary Algorithms*. John Wiley & Sons, 2001.

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

[4] 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, 2001.

[5] J.J. Durillo, J. García Nieto, A.J. Nebro, C.A. Coello Coello, F. Luna, and E. Alba. Multi-objective particle swarm optimizers: An experimental comparison. In *5th International Conference on Evolutionary Multi-Criterion Optimization (EMO'2009)*. Springer, 2009. (accepted).

[6] J.J. Durillo, A.J. Nebro, F. Luna, B. Dorronsoro, and E. Alba. jMetal: A Java Framework for Developing Multi-Objective Optimization Metaheuristics. Technical Report ITI-2006-10, Departamento de Lenguajes y Ciencias de la Computación, University of Málaga, E.T.S.I. Informática, Campus de Teatinos, December 2006.

[7] Y. Hochberg and A. C. Tamhane. *Multiple Comparison Procedures*. Wiley, 1987.

[8] J. Kennedy and R.C. Eberhart. Particle swarm optimization. In *Proceedings of IEEE International Conference on Neural Networks*, pages 1942–1948, 1995.

[9] J. Knowles, L. Thiele, and E. Zitzler. A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers. Technical Report 214, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, 2006.

[10] J. Moore and R. Chapman. Application of particle swarm to multiobjective optimization. Technical report, Departament of Computer Science and Software Engineering, Auburn University, 1999.

[11] A.J. Nebro, J.J. Durillo, C.A. Coello Coello, F. Luna, and E. Alba. Design issues in a study of convergence speed in multi-objective metaheuristics. In G. Rudolph et al., editor, *Parallel Problem Solving from Nature - PPSN X*, volume 5199 of *Lecture Notes in Computer Science*, pages 763–772. Springer, 2008.

[12] A.J. Nebro, J.J. Durillo, F. Luna, B. Dorronsoro, and E. Alba. Design issues in a multiobjective cellular genetic algorithm. In S. Obayashi et al., editor, *Evolutionary Multi-Criterion Optimization. 4th International Conference, EMO 2007*, volume 4403 of *Lecture Notes in Computer Science*, pages 126–140. Springer, 2007.

[13] A.J. Nebro, F. Luna, E. Alba, B. Dorronsoro, J.J. Durillo, and A. Beham. AbYSS: Adapting Scatter Search to Multiobjective Optimization. *IEEE Transactions on Evolutionary Computation*, 12(4), August 2008.

[14] M. Reyes Sierra and C. A. Coello Coello. Improving PSO-Based Multi-objective Optimization Using Crowding, Mutation and ϵ-Dominance. In *Evolutionary Multi-Criterion Optimization (EMO 2005)*, LNCS 3410, pages 505–519, 2005.

[15] M. Reyes-Sierra and C. A. Coello Coello. Multi-Objective Particle Swarm Optimizers: A Survey of the State-of-the-Art. *International Journal of Computational Intelligence Research*, 2(3):287–308, 2006.

[16] E. Zitzler, K. Deb, and L. Thiele. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, 8(2):173–195, 2000.

[17] E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. In K. Giannakoglou et al., editor, *EUROGEN 2001*, pages 95–100, Athens, Greece, 2002.

[18] 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, 1999.