Available online at www.sciencedirect.com
ScienceDirect
Information Sciences 177 (2007) 5033–5049
INFORMATION SCIENCES
AN INTERNATIONAL JOURNAL
www.elsevier.com/locate/ins

Multi-Objective Particle Swarm Optimization with time variant inertia and acceleration coefficients

Praveen Kumar Tripathi *, Sanghamitra Bandyopadhyay, Sankar Kumar Pal

Machine Intelligence Unit, Indian Statistical Institute, 203 B.T. Road, Kolkata 700108, India

Received 20 June 2006; received in revised form 16 June 2007; accepted 23 June 2007

Abstract

In this article we describe a novel Particle Swarm Optimization (PSO) approach to multi-objective optimization (MOO), called Time Variant Multi-Objective Particle Swarm Optimization (TV-MOPSO). TV-MOPSO is made adaptive in nature by allowing its vital parameters (viz., inertia weight and acceleration coefficients) to change with iterations. This adaptiveness helps the algorithm to explore the search space more efficiently. A new diversity parameter has been used to ensure sufficient diversity amongst the solutions of the non-dominated fronts, while retaining at the same time the convergence to the Pareto-optimal front. TV-MOPSO has been compared with some recently developed multi-objective PSO techniques and evolutionary algorithms for 11 function optimization problems, using different performance measures.

Keywords: Multi-objective optimization; Pareto dominance; Particle Swarm Optimization

1. Introduction

The multi-objective optimization (MOO) domain covers many real-life optimization problems. Often this task becomes challenging due to the inherent conflicting nature of the objectives to be optimized. Several computational intelligence based approaches, namely, evolutionary computation, swarm intelligence and artificial immune systems have been used for solving MOO problems. PSO and ant colony optimization methods belong to the swarm intelligence domain of computational intelligence. The population based nature of evolutionary techniques captures the different compromising solutions in the population simultaneously at each iteration. This fact has led to the considerable growth in multi-objective evolutionary algorithms (MOEA), from VEGA in [27], to the most recent techniques like NSGA-II [10], SPEA-2 [37] and PESA-II [8]. A population based swarm intelligence heuristic called Particle Swarm Optimization (PSO) was proposed in 1995

* Corresponding author. Tel.: +91 33 2575 3100; fax: +91 33 2578 3357.
E-mail address: praveen_r@isical.ac.in (P.K. Tripathi).

0020-0255/$ - see front matter
doi:10.1016/j.ins.2007.06.018

[16]. This is inspired by the flocking behavior of birds, which is very simple to simulate and has been found to be quite efficient in handling the single objective optimization (SOO) problems [12,32]. The simplicity and efficiency of PSO motivated researchers to apply it to the MOO problems since 2002. Some of these techniques can be found in [5,6,14,15,18,22,24,36].

In the present article we describe a multi-objective PSO, called Time Variant Multi-Objective Particle Swarm Optimization (TV-MOPSO), where the vital parameters of the PSO i.e., inertia and acceleration coefficients, are allowed to change with the iterations, making it capable of effectively handling optimization problems of different characteristics. The concept of having time varying parameters has been used in earlier works e.g., in genetic algorithms [23], PSO [2,3,34] etc, although most of these works dealt with SOO problems. In this article we incorporate it into the proposed multi-objective PSO. It is known that PSO suffers from the problem of premature convergence [20]. To overcome this problem, mutation operator similar to the ones suggested in [13,19] has been incorporated in TV-MOPSO.

In order to improve the diversity in the Pareto-optimal solutions, a novel parameter exploiting the nearest neighbor concept is used. This method for measuring diversity has an advantage that it needs no parameter specification, unlike the one in [6]. Note that diversity has earlier been incorporated in PSO using different approaches, namely the hyper-grid approach [6], σ-method with clustering [22] and NSGA-II based approach [18]. Both the hyper-grid and clustering based approaches for diversity are found to take significant computational time. In the former, the size of the hyper-grid needs to be specified a priori, and the performance depends on its proper choice. The measure adopted in this article is similar to the one in [18], though the way of computing the distance to the nearest neighbor is different. In order to demonstrate the effectiveness of the proposed diversity measure, TV-MOPSO has also been implemented with both the hyper-grid approach (TV-MOPSO-H) and the clustering approach (TV-MOPSO-C). Results indicate the superiority of the proposed scheme. Comparative study of TV-MOPSO with other multi-objective PSOs viz., σ-MOPSO [22], NSPSO [18] and MOPSO [6] and also other multi-objective evolutionary methods viz., NSGA-II [10] and PESA-II [8] has been conducted to establish its effectiveness for 11 test problems, using four qualitative measures and also visual displays of the Pareto front.

2. Multi-objective optimization

A general minimization problem of M objectives can be mathematically stated as: given $\vec{x} = [x_1, x_2, \ldots, x_d]$, where d is the dimension of the decision variable space,

$$\left.\begin{array}{ll} \text{Minimize :} & \vec{f}(\vec{x}) = [f_i(\vec{x}),\ \ i = 1, \ldots, M] \\ \text{subject to the constraints :} & g_j(\vec{x}) \leqslant 0,\ \ j = 1, 2, \ldots, J, \\ & h_k(\vec{x}) = 0,\ \ k = 1, 2, \ldots, K, \end{array}\right\} \tag{1}$$

where $f_i(\vec{x})$ is the ith objective function, $g_j(\vec{x})$ is the jth inequality constraint and $h_k(\vec{x})$ is the kth equality constraint. The MOO problem then reduces to finding an $\vec{x}$ such that $\vec{f}(\vec{x})$ is optimized. Since the notion of an optimum solution in MOO is different compared to the SOO, the concept of *Pareto dominance* is used for the evaluation of the solutions. This concept formulated by Vilfredo Pareto is defined as [7]:

A vector $\vec{u} = (u_1, u_2, \ldots, u_M)$ is said to dominate a vector $\vec{v} = (v_1, v_2, \ldots, v_M)$ (denoted by $\vec{u} \preceq \vec{v}$), for a multi-objective minimization problem, if and only if

$$\forall i \in \{i, \ldots, M\}, \quad u_i \leqslant v_i \wedge \exists i \in \{1, \ldots M\} : u_i < v_i, \tag{2}$$

where M is the dimension of the objective space.

A solution $\vec{u} \in U$, where U is the universe, is said to be *Pareto Optimal* if and only if there exists no other solution $\vec{v} \in U$, such that $\vec{u}$ is dominated by $\vec{v}$. Such solutions $(\vec{u})$ are called *non-dominated solutions*. The set of all such non-dominated solutions constitutes the *Pareto-Optimal Set* or *non-dominated set*.

3. Particle Swarm Optimization (PSO)

The concept of PSO is inspired by the flocking behavior of the birds. It was first proposed by Kennedy in 1995 [16]. Like evolutionary algorithms PSO is also a population based heuristic, where the population of the potential solutions is called a *swarm* and each individual solution within the *swarm*, is called a *particle*. In order to simulate the behavior of the swarm, each particle is allowed to fly towards the optimum solution.

Considering a d–dimensional search space, an ith particle is associated with the position attribute $\vec{X}_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,d})$, the velocity attribute $\vec{V}_i = (v_{i,1}, v_{i,2}, \ldots, v_{i,d})$ and the individual experience attribute $\vec{P}_i = (p_{i,1}, p_{i,2}, p_{i,d})$. The position attribute $(\vec{X}_i)$ signifies the position of the particle in the search space, whereas the velocity attribute $(\vec{V}_i)$ is responsible for imparting motion to it. The $\vec{P}_i$ parameter stores the position (coordinates) corresponding to the particle's best individual performance. Similarly the experience of whole of the swarm is captured in the index g, which corresponds to the particle with the best overall performance in the swarm. The movement of the particle towards the optimum solution is governed by updating its position and velocity attributes. The velocity and position update equations are given as

$$v_{i,j} = wv_{i,j} + c_1 r_1 (p_{i,j} - x_{i,j}) + c_2 r_2 (p_{g,j} - x_{i,j}), \tag{3}$$

$$x_{i,j} = x_{i,j} + v_{i,j}, \tag{4}$$

where $j = 1, \ldots, d$ and w, c_1, $c_2 \geqslant 0$. w is the inertia weight, c_1 and c_2 the acceleration coefficients, and r_1 and r_2 are random numbers, generated uniformly in the range $[0, 1]$, responsible for providing randomness to the flight of the swarm. The term $c_1 r_1 (p_{i,j} - x_{i,j})$ in Eq. (3) is called *cognition* term whereas the term $c_2 r_2 (p_{g,j} - x_{i,j})$ is called the *social* term. The *cognition* term takes into account only the particle's individual experience, whereas the *social* term signifies the interaction between the particles. The c_1 and c_2 values allow the particle to tune the cognition and the social terms respectively in the velocity update equation (Eq. (3)). A larger value of c_1 allows exploration, while a larger value of c_2 encourages exploitation.

In [4] the trajectories of the particles are theoretically analyzed and a constriction coefficient χ is introduced. This constriction coefficient limits the speed of the particles within bounds, without the need of any velocity clamping.

4. Related study

The simplicity and efficiency of PSO in solving the SOO problems [12,32], inspired its extension to the MOO problem domain. There have been several recent attempts to use PSO for MOO [5,6,14,15,18,22,24,36]. Some of these concepts have been surveyed briefly in this section.

The dynamic neighborhood PSO [15] has been given for two objective MOO problems only. This concept assumes a considerable degree of prior knowledge in terms of the test problem properties. Here instead of a single *gbest*, a local *lbest* is obtained for each swarm member, that is selected from the closest two swarm members. The closeness is considered in terms of one of the objectives, while the selection of the optimal solution from the closest two is based on the other objective. The selection of the objectives for obtaining the closest neighbors and local optima is usually based on the knowledge of the problem being considered for optimization. Usually the simpler objective is considered for closest members computation. A single *pbest* solution is maintained for each member that gets replaced by the present solution only if the present solution dominates the *pbest* solution.

In [24] two methods have been proposed to solve MOO using PSO. The first method uses *weighted aggregate approach*, whereas the second one is inspired by VEGA [27] called *Vector Evaluated Particle Swarm Optimizer* (*VEPSO*).

The Multi-Objective Particle Swarm Optimization Algorithm (*MOPSO*) in [5] maintains two archives, one for storing the globally non-dominated solutions, while the other for storing the individual best solutions attained by each particle. MOPSO uses method inspired by [17] for maintaining diversity. The fitness assigned

to each individual in the archive is computed on the basis of its density. This fitness is used in roulette wheel selection, to pick the *gbest* solution in velocity and position update Eqs. (3) and (4). In the local archive a solution gets replaced by the present solution, only if the former is dominated by the latter. In [6], authors improved the aforementioned MOPSO by incorporating a mutation operator. The mutation operator boosts the exploration capability of the MOPSO. The article also addressed the constraint handling problem with MOO.

In [14] the authors suggested that for the *gbest* of a particle in the swarm, its nearest dominating solution from the archive should be used. For the efficient computation of the nearest dominating solution from the global archive, the concept of *domination tree* has been given. The concept of *turbulence* was also incorporated.

In [22] authors have introduced a concept of σ, for selecting the best local guides. σ values are assigned to the members of the archive as well as that of the swarm. For a particle in the swarm, a particle from the archive with the closest value of σ is chosen as its local guide. The *turbulence* in the decision space has been used and the size of the archive has been kept constant by using clustering based truncation method. Due to the emphasis on the closeness in σ values, and hence even higher the selection pressure, premature convergence may result.

In [18] author has proposed *non-dominated sorting PSO* (*NSPSO*) using the non-dominated sorting of [10] in MOPSO. If the size of the swarm be N, then a combined population of size $2N$, comprising the swarm and its personal bests is first obtained. Then the non-dominated sorting of this population determines the particles of the next generation swarm. To ensure proper diversity amongst the solutions of the non-dominated solutions, two approaches namely *niche count* and *crowded distance* [10] methods are used. An archive containing the non-dominated solutions is also maintained, the best solutions from this archive in terms of diversity are selected as the global best for a particle.

The details of the aforementioned algorithms can be obtained from the respective references.

5. Time Variant Multi-Objective Particle Swarm Optimization (TV-MOPSO)

This section describes the proposed PSO approach to MOO problem. The motivation behind this concept is to attain better convergence to the Pareto-optimal front, while giving sufficient emphasis to the diversity consideration. In TV-MOPSO a new parameter has been incorporated that ensures the diversity of the solutions in the archive. The basic structure of TV-MOPSO is given in **Algorithm TV-MOPSO**. The different steps are described below in detail.

5.1. Initialization

In the initialization phase of TV-MOPSO, the swarm of size N_s is randomly generated. The individuals of the swarm are assigned random values for the coordinates, from the respective domains, for each dimension. Similarly the velocity is initialized to zero in each dimension. Initial value for the parameter $\vec{Pb}_i$ (*pbest*) is set to $\vec{X}_i$, where $\vec{X}_i$ is the ith particle.

Step 1 of TV-MOPSO takes care of the initialization. TV-MOPSO also maintains an archive for storing the best non-dominated solutions found in the flight of the particles in the swarm. In Step 1 of TV-MOPSO, the archive has been initialized to contain the non-dominated solutions from S_0 (swarm at t_0). The function *non_dominated*(S_0) returns the non-dominated solutions from the swarm S_0.

5.2. Update

The update step of TV-MOPSO deals with the simulation of the flight of the particles. Movement of a particle in the search space is mainly governed by its individual experience and by the experience of the group, with which it interacts directly. The flight of the particle is influenced by some vital parameters which are explained below:

Algorithm TV-MOPSO: $O_f = \text{TV-MOPSO}(N_s, N_a, C, d)$
/* N_s: size of the swarm, N_a: size of the archive, C: maximum number of iterations, d: the dimensions of the search space, O_f: the final output */

(1) $t = 0$, randomly initialize S_0, /*S_t: swarm at iteration t */
- initialize $x_{i,j}$, $\forall i$, $i \in \{1, \ldots, N_s\}$ and $\forall j$, $j \in \{1, \ldots, d\}$
/* $x_{i,j}$: the jth coordinate of the ith particle */
- initialize $v_{i,j}$, $\forall i$, $i \in \{1, \ldots, N_s\}$ and $\forall j$, $j \in \{1, \ldots, d\}$
/* $v_{i,j}$: the velocity of ith particle in jth dimension */
- $Pb_{i,j} \leftarrow x_{i,j}$, $\forall i$, $i \in \{1, \ldots, N_s\}$ and $\forall j$, $j \in \{1, \ldots, d\}$
/* $Pb_{i,j}$: the jth coordinate of the personal best of the ith particle */
- $A_0 \leftarrow non\_dominated(S_0)$, $l_0 = |A_0|$ /* returns the non-dominated solutions from the swarm*/
/* A_t: archive at iteration t */

(2) for $t = 1$ to $t = C$,
- for $i = 1$ to $i = N_s$ /* update the swarm S_t */
./* updating the velocity of each particle */
$Gb \leftarrow get\_gbest()$ /* returns the global best */
$Pb_i \leftarrow get\_pbest()$ /* returns the personal best */
$adjust\_parameters(w_t, c_{1t}, c_{2t})$
/* adjusts the parameters, w_t: the inertia coefficient, c_{1t}: the local acceleration coefficient, and c_{2t}: the global acceleration coefficient */
$v_{i,j} = w_t v_{i,j} + c_{1t} r_1 (Pb_{i,j} - x_{i,j}) + c_{2t} r_2 (Gb_j - x_{i,j})$
$\forall j$, $j \in \{1, \ldots, d\}$
./* updating coordinates */
$x_{i,j} = x_{i,j} + v_{i,j}$
$\forall j$, $j \in \{1, \ldots, d\}$
- /* updating the archive */
$A_t \leftarrow non\_dominated(S_t \cup A_t)$
.if $(l_t > N_a)$ $truncate\_archive()$
/*l_t: size of the archive */
- mutate (S_t) /* mutating the swarm */

(3) $O_f \leftarrow A_t$ and stop. /* returns the Pareto optimal front */

5.3. Personal Best Performance (pbest)

In PSO the individual experience of the particle is captured in the *pbest* attribute, that corresponds to the best performance attained so far by it in its flight. In TV-MOPSO, the present solution is compared with the *pbest* solution, and it replaces the latter only if it dominates that solution [6]. The function *get_pbest*() returns the personal best solution in the algorithm TV-MOPSO, in Step 2.

5.4. Global Best Performance (gbest)

The term *gbest* represents the best solution obtained by the swarm. Often the conflicting nature of the multiple objectives involved in MOO problems make the choice of a single optimum solution difficult. To resolve this problem, the concept of non-dominance is used and an archive of non-dominated solutions is maintained, from which a solution is picked up as the *gbest*. TV-MOPSO maintains an archive that has its maximum size limit. The selection of the *gbest* solution is done from the archive on the basis of the diversity of the solutions as in [6]. In TV-MOPSO the diversity measurement has been done using a novel concept. This concept is similar to the *crowding-distance* measure in [10]. The method is described below.

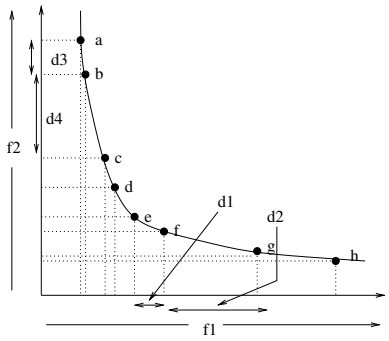

Fig. 1. TV-MOPSO diversity measure.

The computation of the density parameter (den_i) in TV-MOPSO is done amongst the solutions of the archive. The parameter den_i of a solution i is based on its distance to its nearest neighbor in the archive. It may so happen that two solutions may end up with the same value, as they are the closest to each other. As can be seen from Fig. 1 (representing two objective minimization problem), solutions e and f are closest to each other and hence will get the same value for the distance to their nearest neighbor, which may not reflect the real situation. To overcome such a scenario the nearest neighbor distance calculation is done in some order, such that the distance of a solution to its nearest neighbor which has not already been considered (in distance to nearest neighbor computation), is taken. To illustrate this consider Fig. 1. If the nearest neighbor distance is computed in the order a, b, c, d, e, f, g and h, then for the nearest neighbor of solution f, solution g should be considered, as its distance to solution e has been already considered (for nearest neighbor distance of solutions e). To compute this measure efficiently the approach suggested in [10] is used.

In order to compute den_i, solutions are sorted in non-descending order on the basis of different function values. In each such sorting, the normalized difference in function value to its immediate next neighbor is calculated. Finally all these values computed for a solution are added. This gives an estimate of the solutions' density on the front. Considering the scenario in Fig. 1, if the sorting is done on the basis of $f1$ then solution e will have f as its immediate next neighbor and $d1$ will be the corresponding distance along $f1$, while the solution f will have g as its immediate next neighbor and value $d2$ as the distance. Similarly considering $f2$ as a basis of sorting the solution b will have a as its neighbor with its distance $d3$ along $f2$, while the solution c will have $d4$ as its distance along $f2$ to its immediate next neighbor b.

Using the aforementioned method the density for all the solutions in the archive is obtained. Based on the density values as fitness, roulette wheel selection is done to pick a solution as the *gbest*. The higher fitness value signifies better solution. In TV-MOPSO the function *get_gbest*(), returns the *gbest* solution in Step 2, for the velocity update equation.

5.5. Inertia weight (w)

In order to improve the convergence of PSO, a strategy for the incorporation of *inertia weight* w is suggested in [28]. The parameter w, controls the influence of the previous velocity on the present velocity [29]. The higher values of w help in the global search for the optimal solution, while lower values help in the local search around the current search area. All the population based search techniques rely on global exploration and local exploitation in order to achieve good performance. Generally more exploration should be carried out in the initial stages when the algorithm has very little knowledge about the search space. In contrast more exploitation is needed in the later stages so that the algorithm is able to exploit the information it has gained so far.

Since the MOO problems often involve complex search space, the parameter w becomes very vital in multi-objective PSO algorithms. Therefore TV-MOPSO uses a time variant w as in [30]. The issue of adapting the PSO parameters has also been addressed in [2,3,34], for SOO problems. Here adaptation of w is introduced in

multi-objective PSO. The value of w_t is allowed to decrease linearly with iteration from w_1 to w_2. The value of inertia weight at iteration t, w_t is obtained as

$$w_t = (w_1 - w_2)\frac{\max\_t - t}{\max\_t} + w_2, \tag{5}$$

where max_t is the maximum number of iterations and t is the iteration number. The function *adjust_parameters*() in Step 2 of TV-MOPSO algorithm performs this task.

5.6. Acceleration coefficients (c_1 and c_2) and random parameters (r_1 and r_2)

In the velocity update Eq. (3) there are two important parameters c_1 and c_2, called the *acceleration coefficients*. c_1 is called the *cognitive acceleration coefficient*, while c_2 the *social acceleration coefficient*. Higher values of c_1 ensure larger deviation of the particle in the search space, while the higher values of c_2 signify the convergence to the present global best (*gbest*). To incorporate better compromise between the exploration and exploitation of the search space in PSO, time variant acceleration coefficients have been introduced in [26]. TV-MOPSO exploits this concept, ensuring better search for the Pareto-optimal solutions. c_1 has been allowed to decrease from its initial value of c_{1i} to c_{1f} while c_2 has been increased from c_{2i} to c_{2f} using the following equations as in [26]. The values of c_{1t} and c_{2t} are evaluated as follows:

$$c_{1t} = (c_{1f} - c_{1i})\frac{t}{\max\_t} + c_{1i}, \tag{6}$$

$$c_{2t} = (c_{2f} - c_{2i})\frac{t}{\max\_t} + c_{2i}. \tag{7}$$

The values of c_1 and c_2 gets updated with iterations in TV-MOPSO in the *adjust_parameters*() function at Step 2.

The random numbers r_1 and r_2 in Eq. (3) are generated independently for each dimension and for each particle.

5.7. Update archive

Since the selection of the *gbest* solution is done from the archive in TV-MOPSO, it plays a very vital role. At each iteration, the archive gets updated with the inclusion of the non-dominated solutions from the combined population of the swarm and the archive. If the size of the archive exceeds the maximum size limit (N_a), it is truncated using the diversity consideration. Thus the most sparsely spread N_a solutions are retained in the archive. The archive gets updated in TV-MOPSO at Step 2 using function *non_dominated*($S_t U\ A_t$). The function *truncate_archive*() is used to truncate the archive, if it exceeds the maximum limit.

5.8. Mutation

The single objective PSO algorithms have been found to show good convergence properties. However, for the multi-objective PSOs, this convergence is usually achieved at the cost of the diversity [6]. To allow the multi-objective PSO algorithm to explore the search space to a greater extent, while obtaining better diversity, a *mutation* operator has been used in TV-MOPSO, that is based on the one found in [19]. Similar mutation operator has been used in PSO for multi-modal function optimization in [13]. Given a particle, a randomly chosen variable (one of the coordinate of the particle), say g_k, is mutated as given below:

$$g_k' : \begin{cases} g_k + \Delta(t, \mathrm{UB} - g_k) & \text{if } flip = 0, \\ g_k - \Delta(t, g_k - \mathrm{LB}) & \text{if } flip = 1. \end{cases} \tag{8}$$

where *flip* denotes the random event of returning 0 or 1. UB denotes the upper limit of the variable g_k, while LB denotes the lower limit. The function Δ is defined as

$$\Delta(t, x) = x * \left(1 - r^{\left(1 - \frac{t}{\max\_t}\right)^b}\right), \tag{9}$$

where r is a random number generated in the range $[0, 1]$, max_t is the maximum number of iterations and t is the iteration number. The parameter b determines the degree of dependence of mutation on the iteration number. Mutation operation on the swarm is done by the function *mutate*(S_t) in TV-MOPSO at Step 2.

5.9. Termination

The algorithm terminates after executing a specified number of iterations. After termination, the archive contains the final non-dominated front. This is mentioned in Step 3 of TV-MOPSO.

5.10. Complexity consideration

Let the number of functions to be optimized be M, and the size of the archive and swarm be N and n, respectively. In TV-MOPSO the complexity is mainly influenced by the *non_dominated*() function and the diversity computation operation. Considering the archive update process involving *non_dominated*($S_t \cup A_t$), for checking a particle for its non dominance within $N + n$ particles, $M(N + n)$ comparisons are needed. Therefore the worst case complexity of this function will be $O(M(n + N)^2)$. Since the computation of density involves, sorting on the basis of each objective, it will have a complexity of $O(MN\log(N))$ (for archive only). Similarly the archive truncation that involves density consideration will have $O(M(n + N)\log(n + N))$ as its worst case complexity (considering the worst case with $n + N$ elements in the archive). Thus the overall worst case complexity of TV-MOPSO will be $O(M(n + N)^2)$. The computational complexities corresponding to NSGA-II, NSPSO, MOPSO, σ-MOPSO and PESA-II are $O(Mn^2)$, $O(Mn^2)$, $O(Mn^2)$, $O(M(n + N)^2)$ and $O(M(n + N)^2)$, respectively.

6. Experimental results

The effectiveness of TV-MOPSO has been demonstrated on various standard test problems. These problems have a known set of Pareto-optimal solutions and are characterized to test the algorithms on different aspects of their performance. TV-MOPSO has been compared with some MOEAs and MOPSOs. The MOEAs include NSGA-II and PESA-II, whereas the MOPSOs are MOPSO, σ-MOPSO and NSPSO. All these algorithms have been evaluated using the same test problems. The parameters used are given below:

- Population/swarm size 100 for NSGA-II and NSPSO, 10 for PESA-II (as suggested in [8]), 50 for MOPSO, σ-MOPSO and TV-MOPSO.
- Archive size 100 for PESA-II, σ-MOPSO, MOPSO and TV-MOPSO.
- Number of iterations 250 for NSGA-II and NSPSO, 2500 for PESA-II, and 500 for MOPSO, σ-MOPSO and TV-MOPSO (to keep the number of fitness function evaluations to 25000 for all the algorithms).
- Cross-over probability 0.9 (as suggested in [10]) for NSGA-II and PESA-II, no cross-over for MOPSOs.
- Hyper-grid size 32×32 for PESA-II (as suggested in [8]), TV-MOPSO-H.
- Mutation probability inversely proportional to the chromosome length (as suggested in [10]).
- Coding strategy binary for NSGA-II and PESA-II, while real encoding for MOPSO and TV-MOPSO (PSO naturally operates on real numbers).
- $c_{1i} = 2.5$, $c_{1f} = 0.5$, $c_{2i} = 0.5$, $c_{2f} = 2.5$ (as suggested in [26]) (for TV-MOPSO).
- $w_1 = 0.7$ and $w_2 = 0.4$ (as suggested in [31]) (for TV-MOPSO).
- $b = 5$ (as suggested in [19]).

For the evaluation of the performance of MOEAs, researchers have proposed many test problems. Veldhuizen [33], in his doctoral thesis has explored many such test problems. In this article 11 standard test problems have been used. Nine of these test problems (*SCH*1, *SCH*2, *FON*, *KUR*, *POL*, *ZDT*1, *ZDT*2, *ZDT*3, *ZDT*4 [9]) have two objectives, while the other two (*DLTZ*2 and *DLTZ*7 [11]) have three objectives. The performance of the algorithms is evaluated with respect to one or more of the four performance measures: the convergence measure Υ [10], diversity measure Δ [10], *purity* [1] and minimal-spacing (S_m), [1]. Results reported are the mean and the variance values over 20 simulations of the algorithms. It should be noted that Υ and *purity* eval-

uate the convergence of the non-dominated set to the true Pareto-front, whereas Δ and S_m measures reflect the diversity amongst the non-dominated solutions. Larger values of the purity measure, whereas smaller values of Υ, Δ and minimal spacing parameters signify better performance. The details of these measures may be found in the respective references.

7. Results

The proposed TV-MOPSO has been implemented with three different concepts of diversity namely, clustering, hyper-grid and proposed nearest neighbor based diversity concept, hence called TV-MOPSO-C, TV-MOPSO-H and TV-MOPSO, respectively. These different versions of TV-MOPSO, have been evaluated on some test problems, and their performance has been assessed using Υ and Δ measures. However, other performance measures may also be used. The results are reported in terms of the mean and variance of the performance measures in Table 1.

As can be clearly seen from the Table 1, Υ values signify that TV-MOPSO has resulted in better convergence, compared to its other versions (i.e., clustering and hyper-grid), for all the test problems. Similarly, the Δ measure values also highlight that TV-MOPSO is better in terms of diversity, when compared with its other versions, on all the test problems except *DLTZ*2 and *DLTZ*7. Although TV-MOPSO has not given

Table 1
Mean (M) and variance (Var) of Υ and Δ measures for the test problems

Algorithm	SCH1	SCH2	FON	ZDT1	ZDT2	ZDT3	ZDT4	DLTZ2	DLTZ7
Convergence measure Υ									
TV-MOPSO-C(M)	0.01718	0.01697	0.00238	0.00176	0.00134	0.00436	0.72587	0.04550	0.13705
(VAR)	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00107
TV-MOPSO-H(M)	0.01503	0.01087	0.00331	0.00171	0.00119	0.00436	0.98700	0.03681	0.10723
(VAR)	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.07830	0.00000	0. 00000
TV-MOPSO(M)	0.00858	0.00672	0.00142	0.00099	0.00083	0.00404	0.47240	0.03312	0.04953
(VAR)	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
Diversity measure Δ									
TV-MOPSO-C(M)	0.57957	1.52472	0.72652	0.43634	0.38876	0.63368	0.71070	0.52004	0.68734
(VAR)	0.02079	0.00253	0.00000	0.00257	0.00140	0.00243	0.00378	0.00205	0.02582
TV-MOPSO-H(M)	0.82414	1.45810	0.80070	0.68897	0.69676	0.79929	0.88009	0.70776	0.64902
(VAR)	0.01448	0.02623	0.00014	0.00426	0.00415	0.00148	0.00219	0.00051	0.04192
TV-MOPSO(M)	0.32935	1.00305	0.72401	0.33194	0.32565	0.54302	0.69551	0.60364	0.68486
(VAR)	0.00055	0.00105	0.00000	0.00066	0.00058	0.00022	0.01063	0.00072	0.00035

Table 2
Mean (M) and variance (Var) of Υ measure for the test problems

Algorithm	SCH1	SCH2	FON	ZDT1	ZDT2	ZDT3	ZDT4	DLTZ2	DLTZ7
NSGA-II(M)	0.01276	0.01644	0.00260	0.00430	0.00480	0.00900	2.76292	0.72775	0.04976
(VAR)	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	1.33900	0.20400	0.00000
PESA-II(M)	0.01687	0.00962	0.02002	0.00105	0.00074	0.00789	9.98254	0.03219	0.05139
(VAR)	0.00000	0.00000	0.00000	0.00000	0.00000	0.00011	20.13400	0.00000	0.00000
σ-MOPSO(M)	0.01364	0.01516	0.00125	0.01638	0.00584	0.10205	3.83344	0.03167	0.24494
(VAR)	0.00000	0.00000	0.00000	0.00048	0.00000	0.00238	1.87129	0.00000	0.01126
NSPSO(M)	0.01002	0.00854	0.00255	0.00642	0.00951	0.00491	4.95775	0.04938	0.05618
(VAR)	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	7.43601	0.00000	0.00019
MOPSO(M)	0.01148	0.01405	0.00122	0.00133	0.00089	0.00418	7.37429	0.82799	0.04986
(VAR)	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	5.48286	0.01133	0.00000
TV-MOPSO(M)	0.00858	0.00672	0.00142	0.00099	0.00083	0.00404	0.47240	0.03312	0.04953
(VAR)	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000

Table 3
Mean (M) and variance (Var) of Δ measure for the test problems

Algorithm	SCH1	SCH2	FON	ZDT1	ZDT2	ZDT3	ZDT4	DLTZ2	DLTZ7
NSGA-II(M)	0.52329	1.22823	0.76540	0.56931	0.59744	0.73817	0.83973	0.97599	0.89149
(VAR)	0.00738	0.06216	0.00023	0.00068	0.00254	0.00455	0.04149	0.00738	0.00052
PESA-II(M)	0.65025	1.24480	0.85217	0.84816	0.89292	1.22731	1.01136	0.74808	0.74791
(VAR)	0.01298	0.06092	0.00032	0.00287	0.00574	0.02925	0.00072	0.00093	0.00106
σ-MOPSO(M)	0.60937	1.47054	0.85813	0.39856	0.38927	0.76016	0.82842	0.60405	0.94113
(VAR)	0.02237	0.03025	0.00172	0.00731	0.00458	0.00349	0.00054	0.00194	0.00640
NSPSO(M)	0.39165	1.07167	0.78024	0.90695	0.92156	0.62072	0.96462	0.72988	0.73812
(VAR)	0.00494	0.04239	0.00013	0.00000	0.00012	0.00069	0.00156	0.00091	0.01002
MOPSO(M)	0.76097	1.43353	0.84943	0.68132	0.63922	0.83195	0.96194	0.74808	0.87375
(VAR)	0.016427	0.03504	0.00016	0.01335	0.00114	0.00892	0.00114	0.00093	0.08186
TV-MOPSO(M)	0.32935	1.00305	0.72401	0.33194	0.32565	0.54302	0.69551	0.60364	0.68486
(VAR)	0.00055	0.00105	0.00000	0.00066	0.00058	0.00022	0.01063	0.00072	0.00035

Table 4
Mean (M) and variance (Var) of *Purity* measure for the test problems

Algorithm	SCH1	SCH2	KUR	POL	FON	ZDT1	ZDT2	ZDT3	ZDT4	DLTZ2	DLTZ7
NSGA-II(M)	0.910	0.960	0.566	0.591	0.562	0.110	0.081	0.301	0.033	0.321	0.872
(VAR)	0.00000	0.00000	0.00223	0.00372	0.00324	0.01095	0.00296	0.01512	0.00439	0.00103	0.00187
PESA-II(M)	0.980	0.980	0.508	0.792	0.379	0.426	0.817	0.339	0.000	0.994	0.990
(VAR)	0.00600	0.00031	0.00424	0.00830	0.74566	0.01507	0.00518	0.04363	0. 00000	0.00000	0.00000
σ-MOPSO(M)	0.980	0.980	0.560	0.864	0.509	0.279	0.182	0.002	0.198	0.948	0.194
(VAR)	0.00000	0.00000	0.00142	0.00074	0.00234	0.07592	0.02609	0.00000	0. 17394	0.00053	0.18338
NSPSO(M)	0.980	0.910	0.571	0.646	0.494	0.891	0.550	0.609	0.048	0.576	0.802
(VAR)	0.00000	0.00790	0.00037	0.01609	0.00463	0.01234	0.05180	0.00470	0. 00640	0.06074	0.01247
MOPSO(M)	0.970	0.982	0.535	0.912	0.842	0.980	0.987	0.941	0.000	0.000	0.760
(VAR)	0.00000	0.00011	0.00222	0.00055	0.00111	0.00000	0.00000	0.00134	0. 00000	0.00000	0.01625
TV-MOPSO(M)	0.980	0.991	0.634	0.924	0.868	0.990	0.950	0.961	1.000	0.960	0.966
(VAR)	0.00000	0.00000	0.000716	0.00036	0.00053	0.00000	0.00165	0.00021	0. 00000	0.00018	0.00043

Table 5
Mean (M) and variance (Var) of S_m measure for the test problems

Algorithm	SCH1	SCH2	KUR	POL	FON	ZDT1	ZDT2	ZDT3	ZDT4	DLTZ2	DLTZ7
NSGA-II(M)	0.00983	0.05029	0.02430	0.07024	0.00870	0.01201	0.01169	0.04239	0.03570	0.11332	0.10489
(VAR)	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0. 00257	0.00019	0.00013
PESA-II(M)	0.02762	0.05629	0.02656	0.07392	0.01795	0.01562	0.01727	0.04233	0.06382	0.09117	0.10702
(VAR)	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0. 00773	0.00022	0.00000
σ-MOPSO(M)	0.01283	0.05194	0.02432	0.06957	0.02149	0.00985	0.01053	0.03635	0.01606	0.09378	0.25459
(VAR)	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0. 00015	0.00047	0.12227
NSPSO(M)	0.02174	0.05468	0.02634	0.06863	0.01171	0.01045	0.00975	0.03545	0.04789	0.10983	0.23935
(VAR)	0.00000	0.00012	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0. 00639	0.00079	0.09152
MOPSO(M)	0.01573	0.05086	0.02540	0.06498	0.01469	0.01691	0.01606	0.03536	0.04762	0.09080	0.10787
(VAR)	0.00000	0.00000	0.00000	0.00035	0.00000	0.00000	0.00000	0.00000	0. 00385	0.00020	0.00028
TV-MOPSO(M)	0.00757	0.04989	0.00072	0.06807	0.00660	0.00852	0.00811	0.03522	0.02142	0.09203	0.10422
(VAR)	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0. 00061	0.00018	0.00000

the best results in terms of diversity for all the test problems, it has given best performance in terms of convergence.

Tables 2 and 3 represent the Υ and Δ measures for the algorithms on nine test problems, respectively. These problems include seven two-objective problems and two three-objective problems. It can be seen that TV-MOPSO has better convergence on test problems *SCH*1, *SCH*2, *ZDT*1, *ZDT*3, *ZDT*4 and *DLTZ*7, whereas PESA-II, MOPSO and σ-MOPSO, have better convergence on *ZDT*2, *FON* and *DLTZ*2 test problems, respectively. Interestingly the values of the Δ measure in Table 3, show that TV-MOPSO is able to attain the best distribution of the solutions on the non-dominated front, consistently for all the test problems considered.

Tables 4 and 5 represent the purity (P_i) and minimal-spacing parameter (S_m), respectively, for the algorithms on the 11 test problems. The P_i values indicates that TV-MOPSO has better convergence on all the test

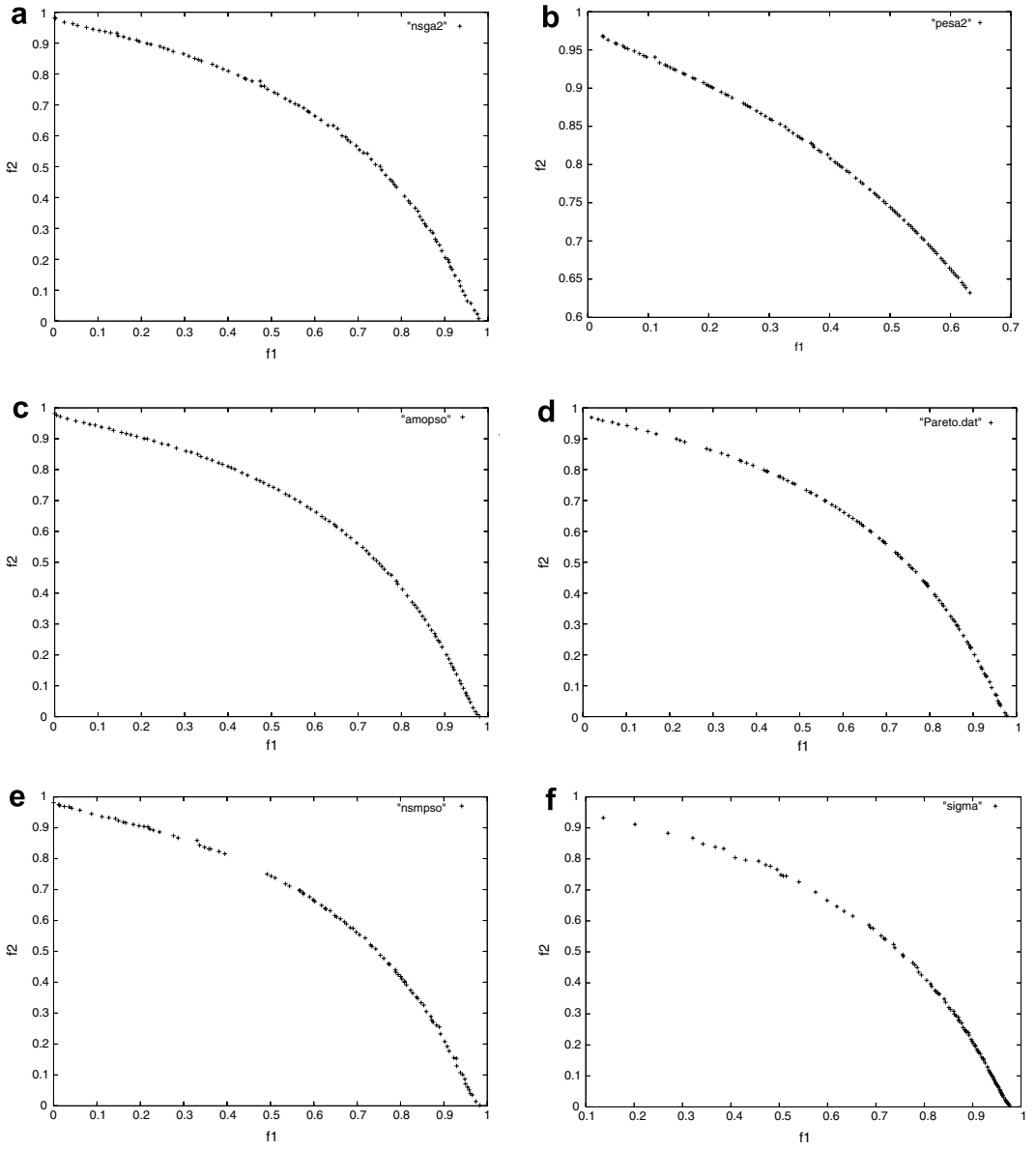

Fig. 2. Final fronts of (a) NSGA-II, (b) PESA2, (c) TV-MOPSO and (d) MOPSO, (e) NSPSO and (f) σ-MOPSO on FON.

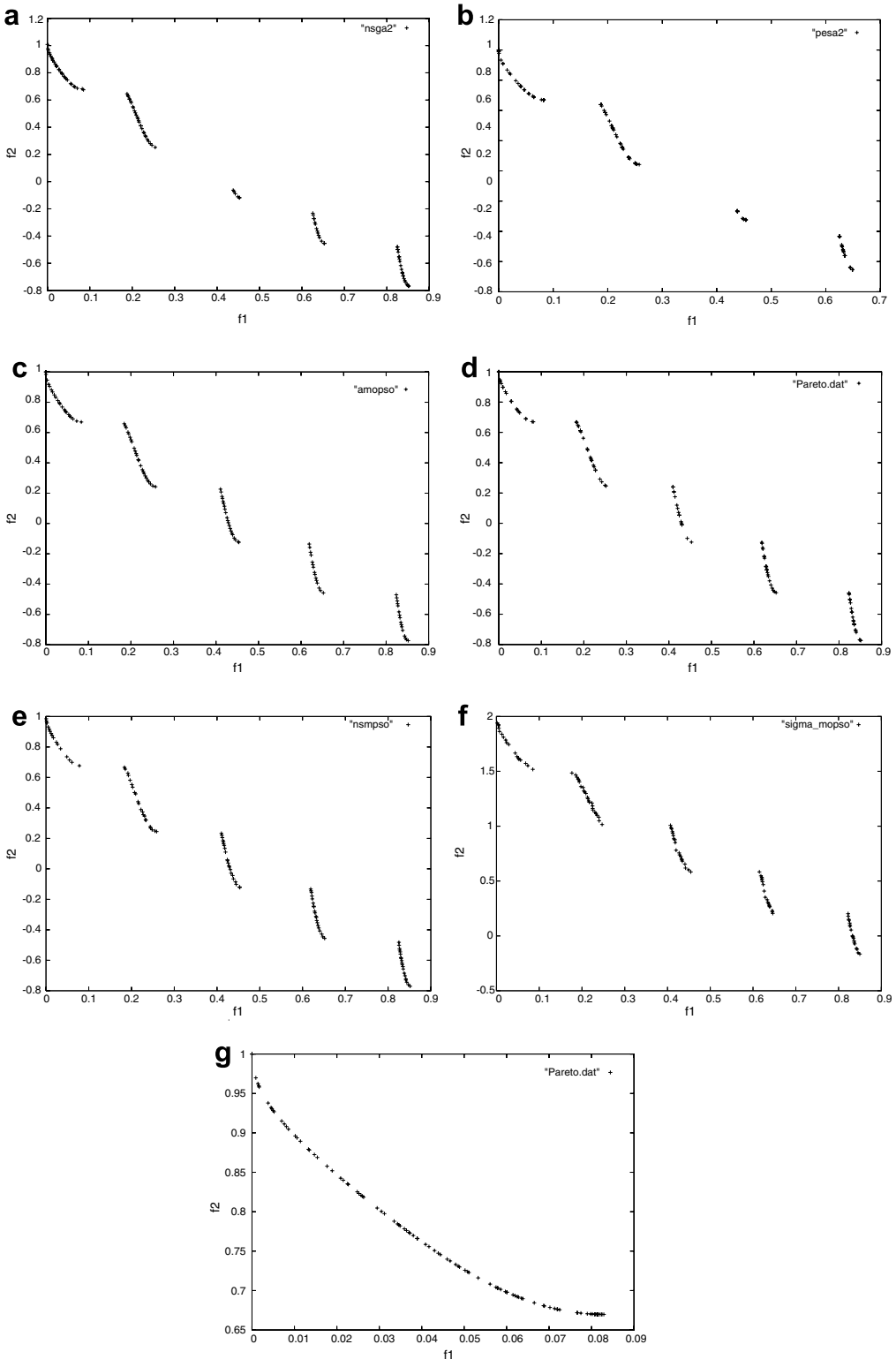

Fig. 3. Final fronts of (a) NSGA-II, (b) PESA2, (c) TV-MOPSO, (d) MOPSO, (e) NSPSO, (f) σ-MOPSO and (g) MOPSO (local convergence) on ZDT3.

problems except *ZDT*2, *DLTZ*2 and *DLTZ*7. However, it is second for *ZDT*2, *DLTZ*2 and *DLTZ*7 test problems, as compared to MOPSO and PESA-II, respectively. The minimal spacing parameter values, as shown in Table 5, indicate that TV-MOPSO has obtained better spread of the solutions on the Pareto-front for all the test problems except *POL*, *ZDT*4 and *DLTZ*2. However, it is second to MOPSO and σ-MOPSO on *POL* and *ZDT*4, respectively.

In order to demonstrate the distribution of the solutions on the final non-dominated front, we have considered *FON*, *ZDT*3 and *DLTZ*2 test problems as typical illustrations. Figs. 2–4 show the resultant non-dominated fronts corresponding to these test problems. Fig. 2 provides the non-dominated solutions returned by the six algorithms for the *FON* test problem. The poor performance of PESA-II and σ-MOPSO is clearly evident. Fig. 2b shows that the convergence obtained by PESA-II is not proper, as it has resulted in a different shape of the front compared to the original Pareto front. Similarly Fig. 2f shows that σ-MOPSO has resulted

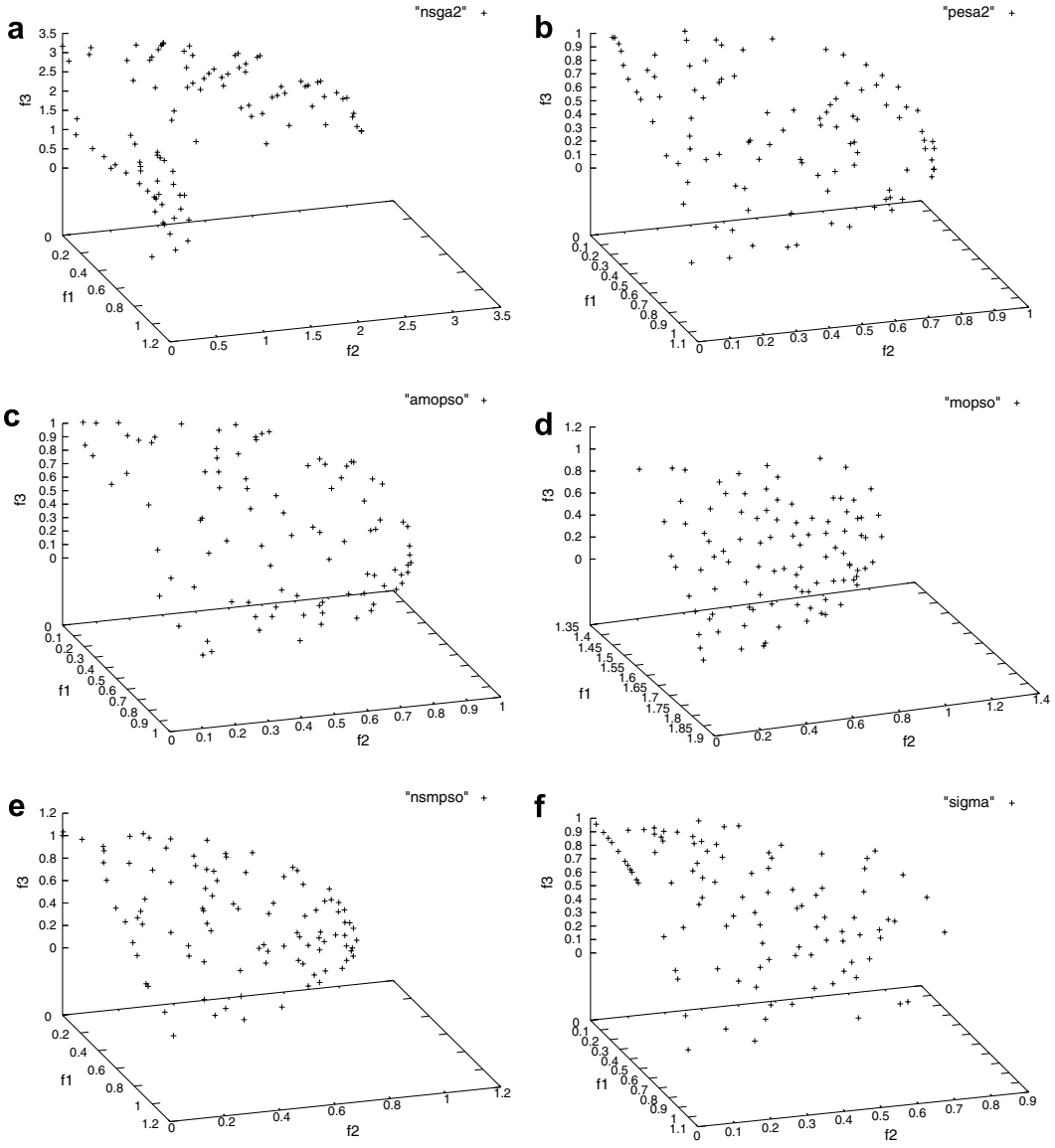

Fig. 4. Final Fronts of (a) NSGA-II, (b) PESA2, (c) TV-MOPSO, (d) MOPSO, (e) NSPSO and (f) σ-MOPSO on ZLTZ2.

in poor diversity amongst the solutions of the non-dominated set. Although, the results in Fig. 2a, d and e are better than the aforementioned results, the best result has been obtained by TV-MOPSO in Fig. 2c, in terms of convergence as well as diversity.

Similarly, Fig. 3 represents the final fronts obtained by the six algorithms for $ZDT3$ function. It can be seen that σ-MOPSO in Fig. 3f, has failed to converge to the true Pareto-front properly, as the highest value for function f_2 is found to be 2, whereas for the results obtained by the other algorithms this value is close to *1* (i.e., the desirable value). It should be noted that PESA-II in Fig. 3b has failed to obtain all the five disconnected Pareto-optimal fronts. Although, NSGA-II has been successful in obtaining these five fronts, one of those did not come out properly in Fig. 3a. MOPSO is found to be better than PESA-II and NSGA-II in this regard, but its solutions have poor spread on the front as clearly evident from Fig. 3(d). Moreover MOPSO is often found to converge to a local optimal front for this test function. Such an instance is shown in Fig. 3g, where MOPSO has been able to obtain only one front (not all the five) because of local optima problem. NSPSO has resulted in a very good convergence, as evident in Fig. 3e, but its diversity is not as good as that of TV-MOPSO. Compared to all these algorithms, TV-MOPSO in Fig. 3c has given better convergence and spread of the solutions on this test function.

The comparative performance of the above algorithms in terms of their performance in a three objective test problem, $DLTZ2$, can be seen in Fig. 4 which shows the final non-dominated fronts obtained by the algorithms. From Fig. 4a it can be seen that NSGA-II has failed considerably in attaining the non-dominated set properly in terms of both the convergence as well as diversity. MOPSO in Fig. 4(d) has failed to attain the full non-dominated set. Similarly σ-MOPSO in Fig. 4f could not attain the non-dominated set properly. Although NSPSO has resulted in better shape in Fig. 4e, its convergence is not as good as that of TV-MOPSO and PESA-II (Figs. 4c and b, respectively). Table 6 summarizes the relative performance of the six algorithms in terms of their ranks, that have been attributed to them, based on their performance with respect to Υ and Δ measures. The table has been derived from the performance in Tables 2 and 3. The entries in Table 6 signify the number of times an algorithm attained a particular rank on nine test functions, in terms of the Υ and Δ measures. As can be seen, TV-MOPSO performed the best in terms of Υ in six of the test problems, whereas PESA-II and MOPSO have performed best in one test problem each. Interestingly, TV-MOPSO has performed the best in terms of Δ measure for all the test problems considered. Similarly the summarized result corresponding to the *Purity* and S_m measures, as shown in Table 7, has been obtained from Tables 4 and 5, respectively. It is evident that TV-MOPSO has given the best convergence results, in terms of purity parameter, for eight of the 11 test problems considered. PESA-II has the best results in terms of purity for three test problems, whereas σ-MOPSO, NSPSO and MOPSO have the best result for one test problem each. Interestingly, for the test problem $SCH1$, TV-MOPSO, NSPSO, PESA-II and σ-MOPSO have attained the same value of purity, signifying the same extent of convergence. Similarly for diversity in terms of S_m measure also,

Table 6
Υ and Δ ranking table for the test problemsz

Algorithm	Rank 1	Rank 2	Rank 3	Rank 4	Rank 5	Rank 6
Υ Ranking						
NSGA-II	0	2	0	3	3	1
PESA-II	1	2	1	2	0	3
σ-MOPSO	1	1	1	0	3	3
NSPSO	0	2	1	3	2	1
MOPSO	1	1	4	1	1	1
TV-MOPSO	6	1	2	0	0	0
Δ Ranking						
NSGA-II	0	1	6	0	2	0
PESA-II	0	0	1	2	4	2
σ-MOPSO	0	4	0	2	0	3
NSPSO	0	4	2	0	1	2
MOPSO	0	1	0	5	2	1
TV-MOPSO	9	0	0	0	0	0

Table 7
Purity and S_m ranking table for the test problems

Algorithm	Rank 1	Rank 2	Rank 3	Rank 4	Rank 5	Rank 6
Purity ranking						
NSGA-II	0	0	4	2	2	3
PESA-II	3	0	2	3	1	2
σ-MOPSO	1	1	3	2	2	2
NSPSO	1	1	3	3	3	0
MOPSO	1	6	0	0	3	1
TV-MOPSO	8	3	0	0	0	0
S_m ranking						
NSGA-II	0	5	1	2	1	2
PESA-II	0	1	1	0	3	6
σ-MOPSO	1	1	3	4	0	2
NSPSO	0	1	4	0	6	0
MOPSO	2	1	1	5	1	1
TV-MOPSO	8	2	1	0	0	0

TV-MOPSO has performed the best in eight of the eleven test problems. MOPSO has given the best results in terms of S_m for two problems, while σ-MOPSO has that for one problem only.

In summary, TV-MOPSO has been consistently found to perform the best amongst all the multi-objective PSOs considered. Considering NSGA-II and PESA-II, it is found that PESA-II performs well in terms of Υ, purity and S_m. However, the performance of TV-MOPSO is similar to, sometimes better than, that of PESA-II, for the test problems considered in this study. Considering the computational complexity, it has been shown in Section 5.10 that TV-MOPSO is comparable with the algorithms studied in this article.

The better performance of the proposed algorithm TV-MOPSO may be attributed to all of its features i.e., mutation operator, diversity measure and the adaptive control parameters. Mutation operator has the role of better exploration of the search space that has been used in earlier works also [6]. It is well established that the performance of the PSO is dependent on the proper choice of the control parameters, which is usually problem specific. Our attempt in this article has been to let the parameters change with the iterations, emphasizing exploration in the initial stages whereas exploitation in the later stages, to overcome this problem. Similarly the better performance of the TV-MOPSO in terms of diversity can be attributed to the novel diversity measure and this has been established empirically with different versions of TV-MOPSO. In future, an exhaustive study regarding the relative contribution of each of the aforementioned features of TV-MOPSO needs to be conducted.

8. Conclusions and discussion

In the present article, a novel multi-objective PSO algorithm, called TV-MOPSO, has been presented. TV-MOPSO is adaptive in nature with respect to its inertia weight and acceleration coefficients. This adaptiveness enables it to attain a good balance between the exploration and the exploitation of the search space. A mutation operator has been incorporated in TV-MOPSO to resolve the problem of premature convergence to the local Pareto-optimal front (often observed in the multi-objective PSOs). An archive has been maintained to store the non-dominated solutions found during TV-MOPSO execution. The selection of the *gbest* solution is done from this archive, using the diversity consideration. The method for computing diversity of the solutions is based on a nearest neighbor concept. In order to evaluate the comparative performance of the aforementioned diversity measure, two other versions of TV-MOPSO, using clustering and hyper-grid for diversity incorporation, are also developed. The results have shown that the proposed approach for diversity preservation, is superior to the other two in terms of convergence and diversity.

The performance of TV-MOPSO is compared with some recently developed multi-objective PSO techniques and evolutionary algorithms, for 11 function optimization problems of two and three objectives. The results have been evaluated on the basis of several performance measures, characterizing the convergence to the Pareto-optimal front and the diversity in the non-dominated set of solutions. TV-MOPSO is found to be

good not only in approximating the Pareto-optimal front, but also in terms of diversity of the solutions on the front.

In future, the problem of MOO in uncertain environment [35] will be explored. Recently some novel concepts have been given for evolutionary computation in [21,25]. We intend to do the comparative evaluation of the proposed algorithm with these algorithms for MOO.

It may be noted that all the algorithms considered here need proper tuning of several parameters. Although in the present article these values have been set in accordance with the suggestions of the authors of the respective algorithms, an exhaustive study regarding the sensitivity of the algorithms to different parameters needs to be undertaken.

References

[1] S. Bandyopadhyay, S.K. Pal, B. Aruna, Multi-objective GAs quantitative indices and pattern classification, IEEE Transaction on Systems Man and Cybernetics – Part B: Cybernetics 34 (2004) 2088–2099.
[2] A. Carlisle, G. Dozier, Adapting particle swarm optimization to dynamic environments, in: Proceedings of International Conference on Artificial Intelligence (ICAI 2000), Las Vegas, Nevada, USA, 2000, pp. 429–434.
[3] A. Carlisle, G. Dozier, Tracking changing extrema with adaptive particle swarm optimizer. in: Proceedings of the 5th Biannual World Automation Congress, Orlando, Florida, USA, 2002, pp. 265–270.
[4] M. Clerc, J. Kennedy, The particle swarm: explosion stability and convergence in a multi-dimensional complex space, IEEE Transactions on Evolutionary Computation 6 (2002) 58–73.
[5] C.A.C. Coello, M.S. Lechuga, MOPSO: A proposal for multiple objective particle swarm optimization, in: Proceedings of the 2002 Congress on Evolutionary Computation part of the 2002 IEEE World Congress on Computational Intelligence, IEEE Press, Hawaii, 2002, pp. 1051–1056, May.
[6] C.A.C. Coello, G.T. Pulido, M.S. Lechuga, Handling multiple objectives with particle swarm optimization, IEEE Transactions on Evolutionary Computation 8 (3) (2004) 256–279.
[7] C.A.C. Coello, D.A.V. Veldhuizen, G.B. Lamount, Evolutionary Algorithms for Solving Multi-Objective Problems, Kluwer Academic Publishers, 2001.
[8] D.W. Corne, N.R. Jerram, J.D. Knowles, M.J. Oates, PESA-II: region-based selection in evolutionary multiobjective optimization, in: Proceedings of the Genetic and Evolutionary Computing Conference (GECCO-2001), Morgan Kaufman, 2001, pp. 283–290.
[9] K. Deb, Multi-Objective Optimization using Evolutionary Algorithms, John Wiley and Sons, USA, 2001.
[10] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multi-objective genetic algorithm: NSGA-II, IEEE Transactions On Evolutionary Computation 6 (2) (2002) 182–197.
[11] K. Deb, L. Thiele, M. Laumanns, E. Zitzler, Scalable Test Problems for Evolutionary Multi-Objective Optimization. Technical Report TIK-Technical Report No. 112, Institut fur Technische Informatik und Kommunikationsnetze,, ETH Zurich Gloriastrasse 35., ETH-Zentrum, CH-8092, Zurich, Switzerland, July 2001.
[12] A.P. Engelbrecht, Fundamentals of Computational Swarm Intelligence, John Wiley and Sons, USA, 2006, January.
[13] S.C. Esquivel, C.A.C. Coello, On the use of particle swarm optimization with multimodal functions, in: The 2003 Congress on Evolutionary Computation, 2003. CEC '03., vol. 2, 2003, pp. 1130– 1136.
[14] J.E. Fieldsend, S.Singh, A multi-objective algorithm based upon particle swarm optimization, an efficient data structure and turbulence, in: Proceedings of UK Workshop on Computational Intelligence (UKCI'02), vol. 2–4, Bermingham, UK, September 2002, pp. 37–44.
[15] X. Hu, R. Eberhart, Multiobjective Optimization Using Dynamic Neighbourhood Particle Swarm Optimization, in: Proceedings of the 2002 Congress on Evolutionary Computation part of the 2002 IEEE World Congress on Computational Intelligence, IEEE Press, Hawaii, 2002, May.
[16] J. Kennedy, R. Eberhart, Particle swarm optimization, in: IEEE International Conference Neural Networks, 1995, pp. 1942–1948.
[17] J.D. Knowles, D.W. Corne, Approximating the nondominated front using the Pareto archived evolution strategy, Evolutionary Computation 8 (2) (2000) 149–172.
[18] X. Li, A non-dominated sorting particle swarm optimizer for multi-objective optimization, in: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2003), Lecture Notes in Computer Science, vol. 2723, Springer, 2003, pp. 37–48.
[19] Z. Michalewicz, Genetic Algorithms + Data Structure = Evolution Programs, Springer-Verlag, 1992.
[20] C.K. Monson, K.D. Seppi, Adaptive diversity in PSO, in: Proceedings of the 8th Annual Conference on Genetic and evolutionary Computation GECCO'2006, ACM Press, New York, NY, USA, 2006, pp. 59–66.
[21] O. Montiel, O. Castillo, P. Melin, A.R. Daz, R. Seplveda, Human evolutionary model: a new approach to optimization, Information Sciences 177 (October) (2006) 2075–2098.
[22] S. Mostaghim, J. Teich, Strategies for Finding Good Local Guides in Multi-objective Particle Swarm Optimization (SIS'03), IEEE Service Center, Inidanapolis, Indiana, USA, 2003. April 26–33.
[23] S.K. Pal, S. Bandyopadhyay, C.A. Murthy, Genetic algorithms for generation of class boundaries, IEEE Transaction on Systems Man and Cybernetics – Part B: Cybernetics 28 (October) (1998) 816–828.
[24] K.E. Parsopoulos, M.N. Vrahatis, Particle swarm optimization method in multiobjective problems, in: Proceedings of the 2002 ACM Symposium on Applied Computing (SAC 2002), 2002, pp. 603–607.

[25] K. Penev, G. Littlefair, Free search – a comparative analysis, Information Sciences 172 (June) (2005) 173–193.
[26] A. Ratnaweera, S.K. Halgamuge, H.C. Watson, Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients, IEEE Transactions On Evolutionary Computation 8 (3) (2004) 240–255.
[27] J.D. Schaffer, Some experiments in machine learning using vector evaluated genetic algorithm, Ph.D. thesis, Vanderbilt University, Nashville,TN, 1984.
[28] Y. Shi, R.C. Eberhart, A modified particle swarm optimizer, in: IEEE Conference on Evolutionary Computation, 1998, pp. 69–73.
[29] Y. Shi, R.C. Eberhart, Parameter selection in particle swarm optimization. in: Evolutionary Programming VII: Proceedings of the Seventh Annual Conference on Evolutionary Programming, New York, 1998, pp. 591–600.
[30] Y. Shi, R.C. Eberhart, Empirical study of particle swarm optimization, in: IEEE International Congress on Evolutionary Computation, vol. 3, 1999, pp.101–106.
[31] F. van den Bergh. An analysis of particle swarm optimizers, Ph.D. thesis, Faculty of Natural and Agricultural Science, University of Pretoria, Pretoria, November 2001.
[32] F. van den Bergh, A.P. Engelbrecht, A study of particle swarm optimization particle trajectories, Information Sciences 176 (2006) 937–971.
[33] D.V. Veldhuizen, Multiobjective evolutionary algorithms: classifications, analysis and new innovations, Ph.D. thesis, Department of Electrical and Computer Engineering, Air Force Institute of Technology (1999), Dayton, 1999.
[34] T. Ying, Y.P. Yang, J.C. Zeng. An enhanced hybrid quadratic particle swarm optimization, in: Sixth International Conference on Intelligent Systems Design and Applications, 2006, ISDA '06, vol. 2, October 2006, pp. 980–985.
[35] L.A. Zadeh, Towards a generalized theory of uncertainity (GTU) – an outline, Information Sciences 172 (2005) 1–40.
[36] Y. Zhang, S. Huang. A novel multi-objective particle swarm optimization for buoys-arrangement design, in: Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT 2004), 2004, pp. 24–30.
[37] E. Zitzler, M. Laumanns, L. Thiele, SPEA2: Improving the strength Pareto evolutionary algorithm. technical report TIK-103, Computer Engineering and Network Laboratory (TIK), Swiss Fedral Institute of Technology (ETH), Gloriastrasse 35, CH-8092 Zurich, Swidzerland, May 2001.