Swarm and Evolutionary Computation 16 (2014) 38–51

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

journal homepage: www.elsevier.com/locate/swevo

Regular Paper

# A comparative performance assessment of a set of multiobjective algorithms for constrained portfolio assets selection

Sudhansu Kumar Mishra [a,*], Ganapati Panda [b], Ritanjali Majhi [c]

[a] Department of EEE, Birla Institute of Technology, Mesra, India
[b] School of Electrical Sciences, Indian Institute of Technology, Bhubaneswar, India
[c] School of Management, National Institute of Technology, Warangal, India

ARTICLE INFO

Article history:
Received 28 November 2011
Received in revised form
14 December 2013
Accepted 1 January 2014
Available online 16 January 2014

Keywords:
Portfolio assets selection
Multiobjective optimization
Efficient frontier
Non-dominated sorting
Particle swarm optimization
Cardinality constraint

ABSTRACT

This paper addresses a realistic portfolio assets selection problem as a multiobjective optimization one, considering the budget, floor, ceiling and cardinality as constraints. A novel multiobjective optimization algorithm, namely the non-dominated sorting multiobjective particle swarm optimization (NS-MOPSO), has been proposed and employed efficiently to solve this important problem. The performance of the proposed algorithm is compared with four multiobjective evolution algorithms (MOEAs), based on non-dominated sorting, and one MOEA algorithm based on decomposition (MOEA/D). The performance results obtained from the study are also compared with those of single objective evolutionary algorithms, such as the genetic algorithm (GA), tabu search (TS), simulated annealing (SA) and particle swarm optimization (PSO). The comparisons of the performance include three error measures, four performance metrics, the Pareto front and computational time. A nonparametric statistical analysis, using the Sign test and Wilcoxon signed rank test, is also performed, to demonstrate the superiority of the NS-MOPSO algorithm. On examining the performance metrics, it is observed that the proposed NS-MOPSO approach is capable of identifying good Pareto solutions, maintaining adequate diversity. The proposed algorithm is also applied to different cardinality constraint conditions, for six different market indices, such as the Hang-Seng in Hong Kong, DAX 100 in Germany, FTSE 100 in UK, S&P 100 in USA, Nikkei 225 in Japan, and BSE-500 in India.



## 1. Introduction

The problem of portfolio assets selection has always been a challenging task for researchers, investors and fund managers. Markowitz set up a quantitative framework for the selection of assets in a portfolio [1,2]. In this framework, the percentage of each available asset is selected in such a way that the total profit of the portfolio is maximized, while the total risk is minimized simultaneously. The sets of portfolios of assets that yield the minimum risk for a given level of return from the efficient frontier. The optimal solution for the standard form of the Markowitz portfolio asset selection problem, which is classified as a quadratic programming model, can be obtained through exact methods, such as active set methods, interior point techniques, etc.

However, portfolio assets' selection is very complicated, as it depends on many factors such as the preferences of the decision makers, resource allocation, growth in sales, liquidity, total turnover, dividend and several other factors. Some authors have also added some practical constraints such as floor, ceiling, and cardinality to the Markowitz model that make it more realistic. The inclusion of these constraints to the portfolio assets selection problem makes it intractable even for small instances. With these constraints, the problem is a mixed integer programming with quadratic objective functions. The traditional optimization methods used to solve this problem get trapped in local minima solutions. To overcome this problem, different efficient heuristic methods are developed.

An overview of the literature on the application of meta-heuristics to the portfolio selection problem has been discussed in [3]. These methods consist of simulated annealing (SA) [4], tabu search (TS) and the genetic algorithm (GA) [5]. Tunchan [6] has applied the PSO technique to solve cardinality constrained portfolios, and the results are compared with those of the GA, TS and SA. Improved PSO algorithms have been proposed by Gao and Chu [7] for the portfolio selection problem with transaction costs. The PSO algorithm has been applied to solve the constrained portfolio selection problem, with bounds on holdings (minimum buy in threshold and maximum limit in combination), cardinality, minimum transaction lots and sector capitalization constraint [8]. Hanhong et al. [9] applied the PSO technique to solve different restricted and unrestricted risky investment portfolios, and compared it with the GA.

* Corresponding author.
E-mail addresses: sudhansu.nit@gmail.com, sudhansu.nit@yahoo.co.in (S.K. Mishra), ganapati.panda@gmail.com (G. Panda), ritanjalimajhi@gmail.com (R. Majhi).

2210-6502/$ - see front matter © 2014 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.swevo.2014.01.001

The portfolio assets selection problem is intrinsically a multi-objective problem having conflicting objectives, i.e., risk and returns. But, in the above studies, the problem has been viewed as a single objective optimization problem, by considering the overall objective as a weighted sum of two objectives. Such a formulation yields multiple solutions, by suitably varying the associated weights. But the selection of the appropriate weights to get an optimal solution is a difficult task. Moreover, it requires several runs to obtain multiple solutions. To overcome these shortcomings, many researchers have applied multiobjective evolutionary algorithms (MOEAs) to solve the problem. One of the main advantages of a MOEA is that it gives a set of possible solutions in a single run, called as a Pareto optimal solution, in a reasonable amount of time [10,11]. The Pareto ant colony optimization (PACO) has been introduced for solving the portfolio selection problem [11] and its performance been compared with other heuristic approaches (i.e., Pareto simulated annealing, and the non-dominated sorting genetic algorithm) by means of computational experiments with random instances.

The portfolio assets' selection problem with many practical constraints is reported by many researchers [12–16]. Mishra et al. [12,13] have applied MOEAs to solve the portfolio assets selection problem with only budget constraint. The literature survey reveals that the cardinality constraint has been addressed by using the hybrid local search in MOEA [14]. The floor, ceiling and cardinality constraints are addressed using MOEAs by some authors [15,16]. All these aforementioned studies lack generality and in-depth analysis, in examining how the presence of these constraints affects the decision of the portfolio manager. Hence, a solution to the portfolio assets selection problem, satisfying a set of constraints, is a challenging one for researchers. In the proposed work, the combined presence of practical constraints such as the budget, floor, ceiling and cardinality is considered, to make the portfolio assets selection problem more realistic.

In the present study, the portfolio assets selection problem is formulated as a multiobjective minimization problem with four practical constraints, and is solved by using the proposed non-dominated sorting multiobjective particle swarm optimization (NS-MOPSO) algorithm. Some peer MOEAs based on non-dominated sorting, such as the Pareto envelope-based selection algorithm-II (PESA-II) [17], strength Pareto evolutionary algorithm 2 (SPEA 2) [18], non-dominated sorting genetic algorithm-II (NSGA-II) [19], and the two-*lbests* based multiobjective particle swarm optimizer (2LB-MOPSO) [20], have been applied to the problem. One MOEA algorithm based on decomposition (MOEA/D) [21] has also been applied to the same problem by formulating the portfolio asset selection problem as a multiobjective maximization problem. The performance of these MOEAs is evaluated, using four statistical metrics such as generation distance, spacing, diversity and convergence metrics. Two nonparametric statistical tests for the pairwise comparison of MOEAs are also demonstrated. The performances of these MOEAs are also compared with four those of single objective optimization algorithms such as the GA, TS, SA and PSO, using the mean Euclidean distance, variance of return error and mean return error.

The rest of the paper is organized as follows. The multiobjective optimization is presented in a concise manner in Section 2. Different multiobjective evolutionary algorithms' frameworks, and the proposed non-dominated sorting multiobjective particle swarm optimization (NS-MOPSO) are discussed in Section 3. In Section 4, the portfolio assets selection problem and its multiobjective formulation are described. Four performance metrics for assessing the performance of MOEAs are discussed in Section 5. Section 5.5 provides the experimental results of the present study. Finally, the conclusion of the investigation is presented, and further possible extension of the work is outlined in Section 6.

## 2. Multiobjective optimization: basic concepts and overview

Multiobjective optimization deals with the simultaneous optimization of multiple objective functions, which are conflicting in nature. A multiobjective optimization problem (MOP) is defined as the problem of computing a vector of decision variables that satisfies the constraints and optimizes a vector function, whose elements represent the objective functions. The generalized multiobjective minimization problem may be formulated [28] as

$$\text{Minimize} \quad f(\vec{x}) = (f_1(\vec{x}), f_2(\vec{x}), \ldots, f_M(\vec{x})) \tag{1}$$

Subject to constraints:

$$g_j(\vec{x}) \geq 0, \quad j = 1, 2, 3 \ldots, J \tag{2}$$

$$h_k(\vec{x}) = 0, \quad k = 1, 2, 3 \ldots, K \tag{3}$$

where $\vec{x}$ represents a vector of decision variables $\vec{x} = \{x_1, x_2, \ldots, x_N\}^T$

The search space is limited by

$$x_i^L \leq x_i \leq x_i^U, \quad i = 1, 2, 3 \ldots, N \tag{4}$$

$x_i^L$ and $x_i^U$ represent the lower and upper acceptable values respectively for the variable $x_i$. $N$ represents the number of decision variables and $M$ represents the number of objective functions. Any solution vector $\vec{u} = \{u_1, u_2, \ldots u_K\}^T$ is said to dominate over $\vec{v} = \{v_1, v_2, \ldots, v_k\}^T$ if and only if

$$\left.\begin{array}{ll} f_i(\vec{u}) \leq f_i(\vec{v}) & \forall i \in \{1, 2, \ldots, M\} \\ f_i(\vec{u}) < f_i(\vec{v}) & \exists i \in \{1, 2, \ldots, M\} \end{array}\right\} \tag{5}$$

Those solutions which are not dominated by other solutions for a given set are considered as non-dominated solutions. The front obtained by mapping these non-dominated solutions into objective space is called the Pareto optimal front (*POF*)

$$POF = f(\vec{x}) = \{(f_1(\vec{x}), f_2(\vec{x}), \ldots, f_k(\vec{x})) | \vec{x} \in p\} \tag{6}$$

where $p$ is the set of obtained non-dominated particles.

The generalized concept of the Pareto front was introduced by Pareto in 1986 [25]. The pioneering work in the practical application of the genetic algorithm to MOP is the vector evaluated genetic algorithm (VEGA) [26]. For similar applications the PESA-II [17], SPEA 2 [18], NSGA-II [19], MOEA/D [21] algorithms have been proposed by many authors. In the recent past, the heuristic approach based on particle swarm optimization has been introduced by Coello et al. [28] to solve multiobjective problems. Some other variants of multiobjective particle swarm optimization techniques, such as the TV-MOPSO [30], FCPSO [31] and 2LB-MOPSO [20] have been suggested to solve the MOP. The PSO is used in the MOEA/D framework where each particle is responsible for solving one subproblem [32]. Multiobjective evolutionary algorithms based on the summation of normalized objective values and diversified selection (SNOV-DS) for solving MOP are proposed by Qu and Suganthan [23]. Following these algorithms a variant of the multiobjective optimization algorithm using particle swarm optimization, called as non-dominated sorting particle swarm optimization (NS-MOPSO), has been proposed and employed to solve the portfolio assets selection problem.

## 3. Multiobjective evolutionary algorithms' frameworks

According to algorithmic frameworks, the MOEAs may be categorized as non-dominated sorting based, decomposition based, memetic, convolution-based and indicator-based [27]. In this paper, five MOEAs algorithms based on non-dominated sorting and one algorithm based on decomposition (MOEA/D) are applied to the portfolio assets selection problem.

### 3.1. Non-dominated sorting based MOEAs

A majority of the MOEAs in both research and application areas are Pareto-dominance based, which are mostly the same as those of NSGA-II [19]. All such algorithms involve two populations of individuals. The first population or archive/external population is used to retain the *best* solutions found during the search. The second population is the normal population of individuals, sometimes used to simply store the offspring population and at other times to take part in the reproduction process. The archive is updated by the *best* individuals, based on information from both the populations, and hence, elitism is ensured.

In these algorithms, a selection operator based on Pareto domination and a reproduction operator are used. The operator of the MOEAs guides the population iteratively towards the non-dominated regions, by preserving the diversity to get the Pareto optimal set. The evaluate operator leads the population convergence towards the efficient frontier, and helps to preserve the diversity of solutions along the efficient frontier. Both goals are achieved by assigning a rank and a density value to each solution. The MOEAs provide first priority to non-dominance, and second priority to diversity. However, the methods by which they achieve these two fundamental goals differ. The main difference between the algorithms lies in their fitness assignment techniques. The popular fitness assignment strategies are alternating objectives-based fitness assignments such as the VEGA [26] and domination-based fitness assignments, such as SPEA 2 [18], NSGA-II [19], etc.

In this paper, the authors have applied five different non-dominated sorting based algorithms, such as PESA-II [17], SPEA 2 [18], NSGA-II [19], 2LB-MOPSO [20] and the proposed NS-MOPSO one. The 2LB-MOPSO algorithm uses two local bests instead of one personal best and one global best, to lead each particle. The two local bests are selected to be close to each other, and help to enhance the local search ability of the algorithm.

#### 3.1.1. The PESA-II algorithm

Corne et al. [17] have proposed the Pareto envelope-based selection algorithm for solving the multiobjective optimization problem. In this algorithm, the newly generated solutions $B_t$ are incorporated into the archive one by one. A candidate child from the newly generated solutions enters the archive when it is non-dominated within $B_t$, or it is not dominated by any current member of the archive. If the addition of a solution renders the archive over-full, then a mating selection is carried out by employing the crowding measure. The crowding distance measurement is done over the archive members. Each individual in the archive is associated with a particular hyper-box. It has a squeeze factor, which is equal to the number of other individuals from the archives, which are present in the same hyper-box. The environmental selection criterion is based on this crowding measure, and used for each individual from the archive. The PESA-II algorithm is proposed in [17] by incorporating region based selection, and shows improved performance over PESA.

#### 3.1.2. The SPEA 2 algorithm

In SPEA 2, mating selection is used, which is based on the fitness measure, and it uses the binary tournament operator [18]. It emphasizes non-dominated individuals by using a technique, which combines the dominance count and dominance rank method. Each individual is assigned a raw fitness value that specifies the number of individuals it dominates, and also the number of individuals by which it is dominated. The density information is incorporated into the raw fitness, by adding a value which is equal to the inverse of the *k*th smallest Euclidean distance to the *k*th nearest neighbor, plus two. The archive updation is performed according to the fitness values associated with each of the individuals in the archive. Then, the updated operator returns all the non-dominated individuals from the combined set of archive and the current pool. There are two possibilities; if the archive size is less than the pre-established size, it is completed with dominated individuals from the current pool; otherwise, some individuals are removed from the archive using the truncation operator. This operator is based on the distance of an individual from its nearest neighbor.

#### 3.1.3. NSGA-II algorithm

Dev and Pratab [19] have proposed the NSGA-II for solving MOPs. The NSGA-II algorithm starts from a random population and utilizes some operators, for uniform covering of the Pareto set. The NSGA-II algorithm for multi-criteria optimization contains three main operators: (i) a non-dominated sorting, (ii) density estimation, and (iii) a crowded comparison. To guide the individuals towards the efficient frontier, the dominance depth method is adopted by the NSGA-II. It classifies the solutions in several layers, based on the position of fronts containing the individuals. The crowding distance mechanism is employed to preserve the diversity of solutions which calculates the volume of the hyper-rectangle defined by the two nearest neighbors. Based on these values, the update operator returns the best individuals from the combination of the archive and the population. Individuals with a lower rank and higher crowding distance would fill the archive. The three main characteristics of NSGA-II are (i) the non-dominated sorting algorithm has a lower computational complexity than its predecessor, the NSGA. The maximum computational complexity of the NSGA-II algorithm is $O(mN^2)$, where $N$ is the population size and $m$ is the number of objectives; (ii) Elitism is maintained, and (iii) no sharing parameter needs to be chosen, because sharing is replaced by the crowded-comparison to reduce computations.

#### 3.1.4. The 2 LB-MOPSO algorithms

In this paper, another most recently proposed evolutionary MO algorithm called the Two-*lbests* based multiobjective particle swarm optimizer (2LB-MOPSO) [20], for solving the portfolio optimization problem, has been applied. This algorithm uses two local bests instead of one personal best and one global best, to lead each particle. In order to select the first lbest for a particle, an objective is first randomly selected, followed by a random selection of a bin of the chosen objective. Within this bin, the archived member with the lowest front number and with the highest crowding distance is selected as the first *lbest*. The second *lbest* is selected from a neighboring non-empty bin, with the lower front number and the smallest Euclidean distance in the parameter space to the first lbest. As each particle's velocity is adjusted by the two *lbests* from two neighboring bins, the flight of each particle will be in the direction of the positions of the two *lbests,* and orient to improve upon the current solutions. A pair of *lbests* is assigned to a particle, and the number of iterations the particle fails to contribute a solution to the archive is counted. If the count exceeds a predefined threshold, the particle is re-assigned to another pair of *lbests.* The two local bests are close to each other, and help to enhance the local search ability of the algorithm.

#### 3.1.5. Proposed non-dominated sorting multiobjective particle swarm (NS-MOPSO): description of the algorithm

Kennedy and Eberhart [22] realized that an optimization problem can be formulated by mimicking the social behavior of a flock of birds, flying across an area looking for food. This observation and inspiration resulted in the invention of a novel optimization technique called particle swarm optimization (PSO). In PSO [22] each solution is represented by a particle and the $i$th particle is given by $X_i=(x_{i1}, x_{i2}, x_{i3}, ..., x_{id})$, where $d$ is the dimension of the search space. The $i$th particle of the swarm population has its best position $P_i=(p_{i1}, p_{i2}, ..., p_{id})$ that yields the highest fitness value. The global best position $P_g=(p_{g1}, p_{g2}, ..., p_{gd})$ is the position of the best particle that gives the best fitness value in the entire population. $V_i=(v_{i1}, v_{i2}, ..., v_{id})$ is the current velocity of $i$th the particle. Particles communicate with each other and for a fully connected topology, the position and velocity of each particle in the next iteration are mathematically expressed as

$$V_{id}(t)=wv_{id}(t-1)+C_1r_1(p_{id}-x_{id})(t-1)+C_2r_2(p_{gd}-x_{id})(t-1) \quad (7)$$

$$x_{id}(t)=x_{id}(t-1)+\chi v_{id}(t) \quad (8)$$

where $d=1,2,...,D$ and $i=1,2,...,N$. The size of the swarm population is $N$. $\chi$ is a constriction factor which controls and constricts the velocity magnitude. $w$ is the inertia weight parameter to control the exploration or exploitation in the search space. It can be a positive constant, or a linear or nonlinear function of time [22]. $r_1$ and $r_2$ are two random values, called as acceleration constants within the range [0, 1].

In classical PSO, each particle tries to maximize the food substance obtained, by moving across the multi-dimensional search space, by updating its velocity and position. It is the only objective that governs the search process. But in the course of moving, it may face constraints like favorable temperature conditions, and it is expected that the swarm should not move to a region of unfavorable temperature. If the temperature constraint is incorporated by adding a penalty function to the actual nutrient concentration, then the approach leads to a single objective constraint optimization. Instead, the food concentration and favorable temperature can be considered as two separate objectives. The individual particle tries to optimize these two objectives simultaneously, and this can be applied to the multiobjective optimization problem.

Coello et al. [28] extended the PSO to MOPSO, in order to deal with the multiobjective problem. In our proposed NS-MOPSO, the concept of non-dominated sorting is incorporated in MOPSO, satisfying both the objectives and the constraints. Those swarms whose locations represent non-dominated solutions are classified as the optimal Pareto front 1 (OPF1) and the remaining swarms are classified as higher OPFs. In this way, the complete population is ranked, based on the Pareto-dominance criteria. The locations in

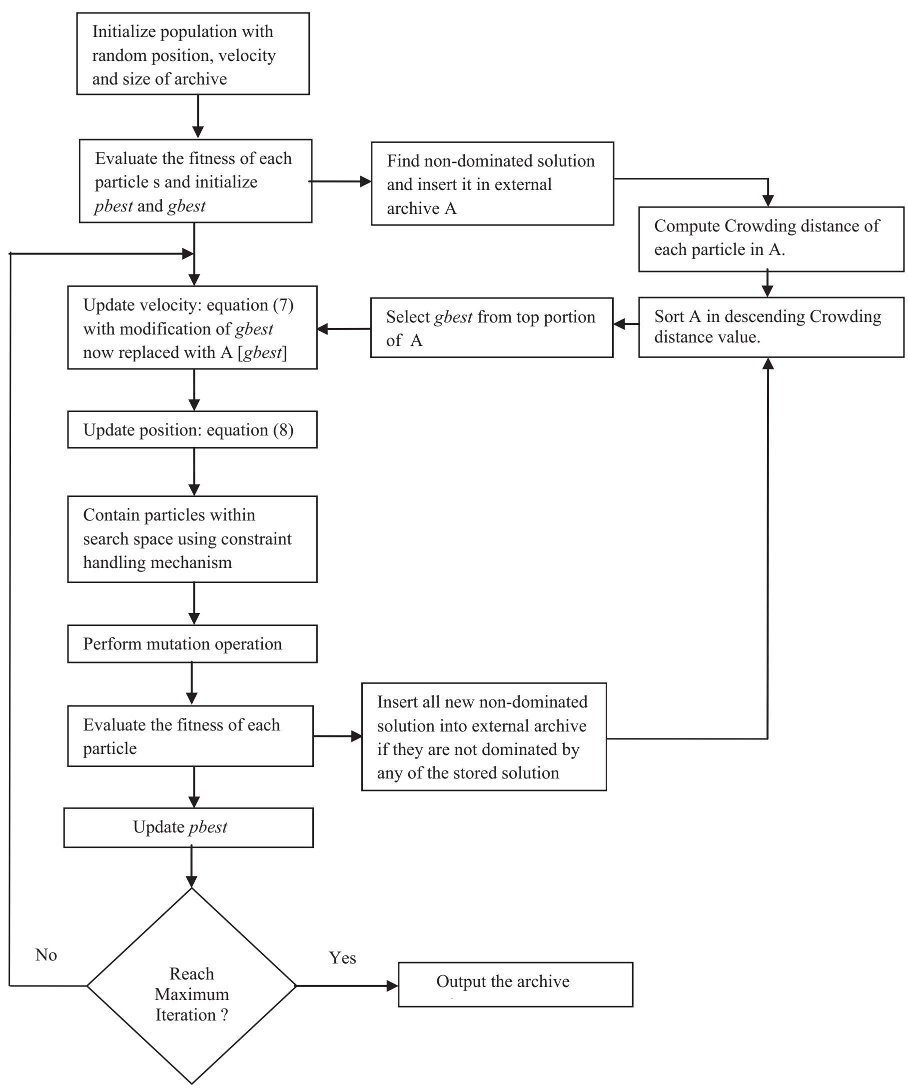


**Fig. 1.** Flow chat of NS-MOPSO algorithm.

the lower OPF1 are rich in food, and the locations of higher OPFs are poor in food content. Each particle updates its velocity and position based on its own best experience and that of the particles with lower OPF. The updating rule enables the particles to move toward the lower optimal Pareto front.

The constraint handling is carried out, based on the approach given by Deb et al. [19]. In this approach, the normalized sum of constraint violations for all individuals is calculated. Then, the individuals are classified according to the overall constraint violation. In between any two individuals, if the overall violation of both of them is zero, then the ordinary ranking assignment is applied. Otherwise, the individual with the lower (or null) overall violation dominates the other one. In this study of NS-MOPSO based portfolio assets selection, the position of each particle represents a weight vector associated with different assets. The two fitness functions (risk and return) evaluate the fitness value for each particle. A flow chat of the proposed algorithm is shown in Fig. 1.

The pseudo code of the NS-MOPSO algorithm can be summarized in the following steps.

*Step* 1: Initialization of parameters

- $N$: Population size and store the population in a list *PSOList*.
- $X_i$: The current position of the $i$th particle within a specified variable range.
- $V_i$: The current velocity of the $i$th particle within the specified decision variable range. It has the probability of 0.5 being specified in a different direction.

The personal best position Pi is set to X.

- $V_{UPP}$ and $V_{LOW}$: Upper and lower bounds of the decision variable range.
- *MaxIterations*: Maximum number of iterations.

*Step* 2: Evaluate each particle in the population.
*Step* 3: Iteration count loop:$t=t+1$.
*Step* 4: Identify particles that give non-dominated solutions in the population and store them in a list *NonDomPSOList*.
*Step* 5: Calculate the crowding distance value for each particle.
*Step* 6: Resort the *NonDomPSOList* according to the crowding distance values.
*Step* 7: Number of particles: $i=i+1$ (step through *PSOList*).

- Select randomly a global best $P_g$ for the $i$th particle from a specified top part (e.g. top 5%) of the sorted *NonDomPSOList.*
- Calculate the new velocity $V_i$ and the new $X_i$ based on Eqs. (7) and (8) respectively.
- Add the $i$th particles $P_i$ and the new $X_i$ to a temporary population, stored in *NextPopList.* At this stage the $P_i$ and $X_i$ coexist and the size of *NextPopList* is $2N$.

*Step* 8: If $i<N$, go to the next particle $(i+1)$ (step 7).
*Step* 9: Identify particles that give non-dominated solutions from *NextPopList* and store them in *NonDomPSOList*. Particles other than non-dominated ones from *NextPopList* are stored in a list *NextPopListRest*.
*Step* 10: Empty *PSOList* for the next iteration step.
*Step* 11: Select random members of *NonDomPSOList* and add them to the *PSOList* (not to exceed number of particles ($N$)). Assign the rest of *NonDomPSOList* as *NonDomPSOListRest*.
*Step* 12: If *PSOList* size < number of particles ($N$).

- Identify non-dominated particles from *NonDomPSOListRest*, and store them in *NextNonDomList.*
- Add member of *NextNonDomList* to *PSOList.*
- If still the *PSO List* size $<N$, copy *NextPopListRest* to *NextPopListRestCopy*, then vacate *NextPopListRest.*
- Assign the vacant *NextPOPListRest* with the remaining particles, other than non-dominated ones from *NextPopListRestCopy.*

*Step* 13: If *PSOlistsize* < number of particles ($N$), go to (step 12).
*Step* 14: If $t<$ *MaxIterations*, go to the next iteration (step 3)

### 3.2. Decomposition based MOEAs

The decomposition MOEA (MOEA/D) [21] is another form of algorithm. In this approach, the multiobjective optimization problem is decomposed into a number of scalar objective optimization problems (SOPs). The objective of each SOP, called a subproblem, is a weighted aggregation of the individual objectives.

The MOEA/D decomposes the multiobjective optimization problem into $N$ scalar optimization subproblems. It solves these subproblems simultaneously, by evolving a population of solutions. At each generation, the population is composed of the best solution found so far for each subproblem. The neighborhood relations among these subproblems are defined, based on the distances between their aggregation weight vectors. A subproblem is a neighbor of another subproblem, if the weight of the first is close to that of the other. Each subproblem is optimized in the MOEA/D by using information mainly from its neighboring subproblems. In this case, each individual subproblem keeps one solution in its memory, which could be the best solution found so far for the subproblem. The MOEA/D provides flexibility of incorporating any decomposition approach, into its framework for solving the MOPs.

The MOEA/D optimizes $N$ scalar optimization problems, rather than directly solving an MOP as a whole. Therefore, it employs scalar optimization methods as each solution is associated with a scalar optimization problem.

#### 3.2.1. Decomposition based MOEAs using particle swarm

The issues of fitness assignment and diversity maintenance are easier to handle in the framework of MOEA/D. Several improvements on MOEA/D have been reported in [33] and have been applied to a number of application areas [34,35]. In their works, the authors have applied one decomposition based MOEA algorithm (MOEA/D), in which the objective of each subproblem has been optimized, using the heuristic algorithm. In this paper, the authors have used one variant of the (MOEA/D) algorithm, where an individual objective is optimized using the particle swarm optimization for designing a decomposition-based multiobjective evolutionary algorithm (MOEA/D).

### 3.3. Constraint handling in MOEAs

Coello et al. [28] classified the constraints handling methods into five categories: (1) penalty functions, (2) special representations and operators, (3) repair algorithms, (4) separate objectives and constraints, and (5) hybrid methods. The past, present and future aspects of constraint handling in nature-inspired numerical optimization have been described by Coello [29]. A constraint dominance concept has been introduced by Deb et al. [19] to handle constraints in multiobjective problems. A solution $x$ dominates a solution $y$ if (i) $x$ is feasible and $y$ is infeasible, (ii) both are infeasible, and $x$ has a less constraint violation than $y$, or (iii) both are feasible and $x$ dominates $y$. The solutions are ranked using the non-constraint-dominated method, while the superiors are selected to evolve. The handling of different practical constraints in the portfolio assets selection problem is explained in Section 6.

Although MOEAs have been investigated in depth within the context of unconstrained and bound constrained MOPs, the general constraints encountered in real-world problems have not been considered. Typically, the search space $\Omega$ of a constrained MOP [27] can be formulated as follows:

$$\Omega=\begin{cases} g_i(x)=g_j(x_1,x_2,...,x_n)\le 0, & j=1,2,...,J \\ h_k(x)=h_k(x_1,x_2,...,x_n)=0, & k=1,2,...,K \\ x_i^L\le x_i\le x_i^U & i=1,2,...,n \end{cases} \tag{9}$$

where $g_j(x)$ and $h_k(x)$ are inequality and equality constraint functions, respectively. Generally, equality constraints are transformed into inequality forms, and then combined with the inequality constraints using

$$G_j(x) = \begin{cases} \max\{g_j(x), 0\}, & j = 1, 2, \ldots, J \\ \max\{|h_j - J(x)| - \delta, 0\}, & j = J+1, J+2, \ldots, J+K \end{cases} \quad (10)$$

where $\delta$ is a tolerance parameter for the equality constraints. Due to the presence of constraints, the search space is partitioned into feasible and infeasible regions.

## 4. Portfolio assets selection problem

The main objective of portfolio assets selection is the maximization of returns and minimization of risk. In addition to these two objectives, other restrictions may be present, called as constraints. In the Markowitz model [2] for portfolio selection, variance is used as a measure of risk which is mathematically expressed as

$$\sigma_p^2 = \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij} \quad (11)$$

where $\sigma_{ij}$ is the covariance between assets $i$ and $j$,

$\sigma_p^2$ is the variance of portfolios, and $N$ denotes the number of assets available, $w_i$ and $w_j$ (weighting of asset) are the proportions of the portfolios held in assets $i$ and $j$ respectively.

The portfolio return is represented as

$$r_p = \sum_{i=1}^{N} w_i r_i \quad (12)$$

$r_i$ is the expected return of the asset $i$.

$r_p$ is the expected return of the portfolio.

$$\text{Subject to} \quad : \sum_{i=1}^{N} w_i = 1 \quad (13)$$

$$a_i z_i \le w_i \le b_i z_i, \quad (14)$$

where

$$0 \le a_i \le 1 \quad (15)$$

$$0 \le b_i \le 1 \quad (16)$$

and

$$z_i = \begin{cases} 1 & \text{for } w_i > 0 \\ 0 & \text{otherwise} \end{cases} \quad (17)$$

$$\sum_{i=1}^{N} z_i = K \quad (18)$$

Eq. (13) shows the budget constraint which ensures that the sum of the weights associated with each asset is equal to one, i.e., all the available money is invested in the portfolio. $a_i$ is the floor constraint, and it is the lowest limit on the proportion of any asset that can be held in a single portfolio. It prevents excessive administrative costs for very small holdings, which have an insignificant influence on the performance of the portfolio. $b_i$ is the ceiling constraint and is the maximum limit on the proportion of any asset that can be held in a single portfolio. It prevents the excessive exposure to any portfolio, which is part of the institutional diversification policy.

The decision variable $z_i$ is 1 or 0 depending upon whether an asset $i (i = 1, 2, \ldots, N)$ is held or not respectively. Eq. (14) ensures that if any asset $i$ is selected ($z_i = 1$), its proportion $w_i$ must lie between $a_i$ and $b_i$; otherwise $w_i$ is zero ($z_i = 0$). Eq. (18) ensures that exactly $K$ assets (cardinality constraint) of $N$ available assets are held in a single portfolio.

Hence, with the presence of two objectives as shown in Eqs. (11) and (12), and the constraints shown in Eqs. (13)–(18), the portfolio problem becomes a multiobjective optimization problem, and the sole aim is to find all non-dominated sets of solutions. This multiobjective portfolio assets selection problem is solved, with single objective evolutionary algorithms (SOEAs)-based techniques by many researchers [5–9]. Most of these approaches consider the overall objectives as a weighted sum of the two objectives, which can be expressed mathematically as

$$\text{Minimize} \quad V = \lambda[\sigma_p^2] - (1-\lambda)[r_p]$$

$$= \lambda\left[\sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j \sigma_{ij}\right] - (1-\lambda)\left[\sum_{i=1}^{N} w_i r_i\right] \quad (19)$$

The only objective is to minimize '$V$'. By repeatedly varying the parameter value $\lambda$, and solving a sequence of optimization problems (for each $\lambda$), the efficient portfolios from the minimum variance portfolio ($\lambda = 1$) to the maximum return portfolio ($\lambda = 0$) can be found. Hence, such a formulation yields non-dominated solutions, by suitably varying the $\lambda$ factor from 0 to 1 with a small increment, viz., 0.01 or 0.02. The main advantage of these approaches is that it reduces the multiobjective problem to a scalar optimization problem, and then, any single objective metaheuristics algorithm can be applied.

However, solving this multiobjective problem with these SOEAs methods requires the repeated use of an optimization technique to find one single solution on the efficient frontier per run. Hence, it is a time consuming process to get the entire Pareto front. Furthermore, a uniform set of $\lambda$ does not guarantee a uniformly distributed set of efficient points [10]. The diversity of solutions along the efficient frontier is of immense importance, since certain trade-off portfolios of interest may be missed, if they are concentrated in a small area of the efficient frontier. One more shortfall of this approach is that it cannot find all efficient points as shown in [5]. In addition, if practical constraints are considered the problem becomes extremely difficult to solve, by using this method. To overcome these shortcomings the portfolio assets selection problem is solved, by applying multiobjective evolutionary algorithms (MOEAs), by suitably formulating it as a multiobjective optimization problem. These MOEAs give a set of possible solutions in a single run, called as a Pareto optimal solution.

### 4.1. *Multiobjective formulation of portfolio assets selection*

The portfolio assets selection is formulated as a multiobjective optimization problem for non-dominated sorting based MOEAs, and decomposition based MOEAs, as dealt within the following subsections.

#### 4.1.1. *Formulation for non-dominated sorting based MOEAs*

The multiobjective portfolio assets selection problem can be solved by MOEAs based on non-dominated sorting, which do not combine the two objectives to obtain the Pareto optimal solution set. Here, the two objectives are taken individually, and an attempt is made to optimize both simultaneously.

The main objective is to maximize the return, $r_p$ and minimize the risk, $\sigma_p^2$. The proposed NS-MOPSO are suitably oriented in such a way as to minimize the two objectives. To express both the objectives in minimization form, the first objective $r_p$ is expressed as $-r_p$. In addition to these objectives, different practical constraints mentioned in Eqs. (13)–(18) are also considered. Accordingly, the portfolio problem is expressed as

$$\textit{Minimize } \sigma_p^2 \textit{ and } -r_p \textit{ simultaneously considering all constraints} \quad (20)$$

Hence, in the presence of these multiple objectives and constraints, the problem becomes a complex multiobjective minimization problem. By solving this Eq. (20), a set of efficient solutions, called the efficient frontier, is obtained. This is a curve lying between the global minimum risk portfolio and the maximum return portfolio. In the present investigation, this efficient frontier is termed as the Pareto front.

#### 4.1.2. Formulation for decomposition based MOEAs (MOEA/D)

The MOEA/D provides the flexibility of using any decomposition approach in its framework for solving the MOPs. These approaches include the weighted sum approach, Tchebycheff approach, and the Boundary intersection approach [21]. If the weighted sum approach is applied to the MOEA/D algorithm, it considers a convex combination of different objectives. Mathematically it is expressed as

$$\text{Maximize} \quad g^{ws}\ (w/\lambda) = \sum_{i=1}^{m} \lambda_i f_i(w) \quad (21)$$

subject to $x \in \Omega$where $\lambda = (\lambda_1, ..., \lambda_m)^T$ is the weight vector, i.e. $\lambda_i \geq 0$ for all $i = 1, ..., m$ and

$$\sum_{i=1}^{m} \lambda_i = 1 \quad (22)$$

The symbol $\lambda$ is a coefficient vector in the objective function and $x$ is the variable to be optimized. Different weight vectors of $\lambda$ are used in the above scalar optimization problem to generate a set of different Pareto optimal vectors.

In the portfolio assets selection problem, the number of objectives $m$ is two, i.e., risk and return. For applying the MOEA/D, the portfolio assets selection problem can be expressed as

$$\text{Maximize} \quad g^{ws}(w/\lambda) = \sum_{i=1}^{2} \lambda_i f_i(w) \quad (23)$$

where $\lambda_i \geq 0$ for all $i = 1, 2$ and $\sum_{i=1}^{2} \lambda_i = 1$, subject to $x \in \Omega$, $\lambda$ is a coefficient vector of the objective function, and $x$ is the variable to be optimized. The two functions $f_1(x)$ and $f_2(x)$ are to be maximized. To generate a set of different Pareto optimal vectors, one can use different weight vectors $\lambda$ in the above scalar optimization problem. In a single run, a set values of $\lambda$ is utilized, and using the neighborhood concept the complete set of solutions on the Pareto front is obtained.

Since the objective is to maximize the return $r_p$ and minimize the risk $\sigma_p^2$, the risk $\sigma_p^2$ may be expressed in maximization form as $-\sigma_p^2$. In addition to these objectives, different practical constraints mentioned in Eqs. (13)–(18) are also considered. Accordingly the portfolio problem is expressed as

$$\textit{Maximize} - \sigma_p^2 \textit{ and } r_p \textit{ considering all constraint} \quad (24)$$

Hence, in the presence of this multiple objectives and constraints, the problem becomes a multiobjective maximization problem. A set of Pareto optimal solutions is obtained by solving Eq. (24) in a single run.

## 5. Simulation study

The algorithms are simulated in a MATLAB environment, and are run on a PC with Intel Core 2 Duo 3.0 GHz with 4 GB RAM. The portfolio assets selection problem is solved using NS-MOPSO. The results thus obtained are compared with the corresponding results achieved using the other five MOEAs (PESA-II, SPEA 2, NSGA-II, 2LB-MOPSO and MOEA/D) as well as the results obtained, using the four single objective evolutionary algorithms (SOEAs), such as the PSO, GA, TS and SA identical to those dealt within [6].

### 5.1. Solution representation, encoding and constraints satisfaction

The hybrid representation proposed by Streichert et al. [38] has been implemented, which seems to be more appropriate for portfolio assets selection. In order to have a fair comparison, the same solution representation is carried out for all the algorithms. In the hybrid representation, two vectors are used for defining a portfolio: a binary vector that specifies whether a particular asset participates in the portfolio, and a real-valued vector used to compute the proportions of the budget invested in the assets:

$$\Delta = \{z_1, \ldots, z_n\}, \quad z_i = \{0, 1\}, \quad i = 1, \ldots, n.$$
$$W = \{w_1, \ldots, w_n\}, \quad 0 \leq w_i \leq 1, \quad i = 1, \ldots, n. \quad (25)$$

Before the objective values are computed, the repair algorithm is performed in order to find the portfolio $x$ associated with the above encoding. First, if the number of assets in the portfolio, i.e., the number of 1′s in $\Delta$ of Eq. (25), overcomes the maximum allowable ones, then those assets that have the minimum weight in $W$ are deleted (by changing its value from 1 to 0 in $\Delta$). In this way, the portfolio satisfies the cardinality constraints.

To meet the constraint, the simplest strategy is to normalize the weights, so that the total weights will meet the budget constraint, i.e., the sum of the weights equal to one. This is mathematically defined as

$$W_{adjusted} = \frac{w_i.z_i}{\sum_{i=1}^{n} w_i.z_i} \quad (26)$$

If both the floor and the ceiling constraints are included, then the weight values are to lie within a specific range. Hence, the simple strategy of normalizing the total weights to one, so as to meet the budget constraint is no longer applicable, as the normalized weights might not be within the limits. Hence, the fitness evaluation for the proposed representation as in Eq. (26) has to be modified. The modified fitness evaluation has been initialized with an empty portfolio, where the assets are added iteratively. The combination of floor and ceiling constraints can be divided into three different cases.

*Case* 1: If both floor and ceiling constraints are present, then the weights vector represented by Eq. (26) needs to be adjusted as

$$W_{adjusted} = a_i.z_i + \frac{w_i.z_i}{\sum_{i=1}^{n} w_i.z_i}\left(b_i.z_i - \sum_{i=1}^{n} a_i.z_i\right), \quad i = 1, \ldots, n. \quad (27)$$

*Case* 2: If the weight has to be adjusted only for the floor constraint, and there is no restriction on the upper limit (ceiling

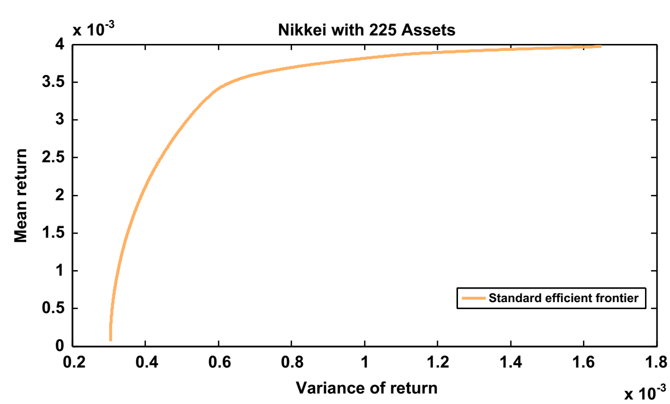


**Fig. 2.** Plots of standard efficient frontier for Nikkei 225 stock indices.

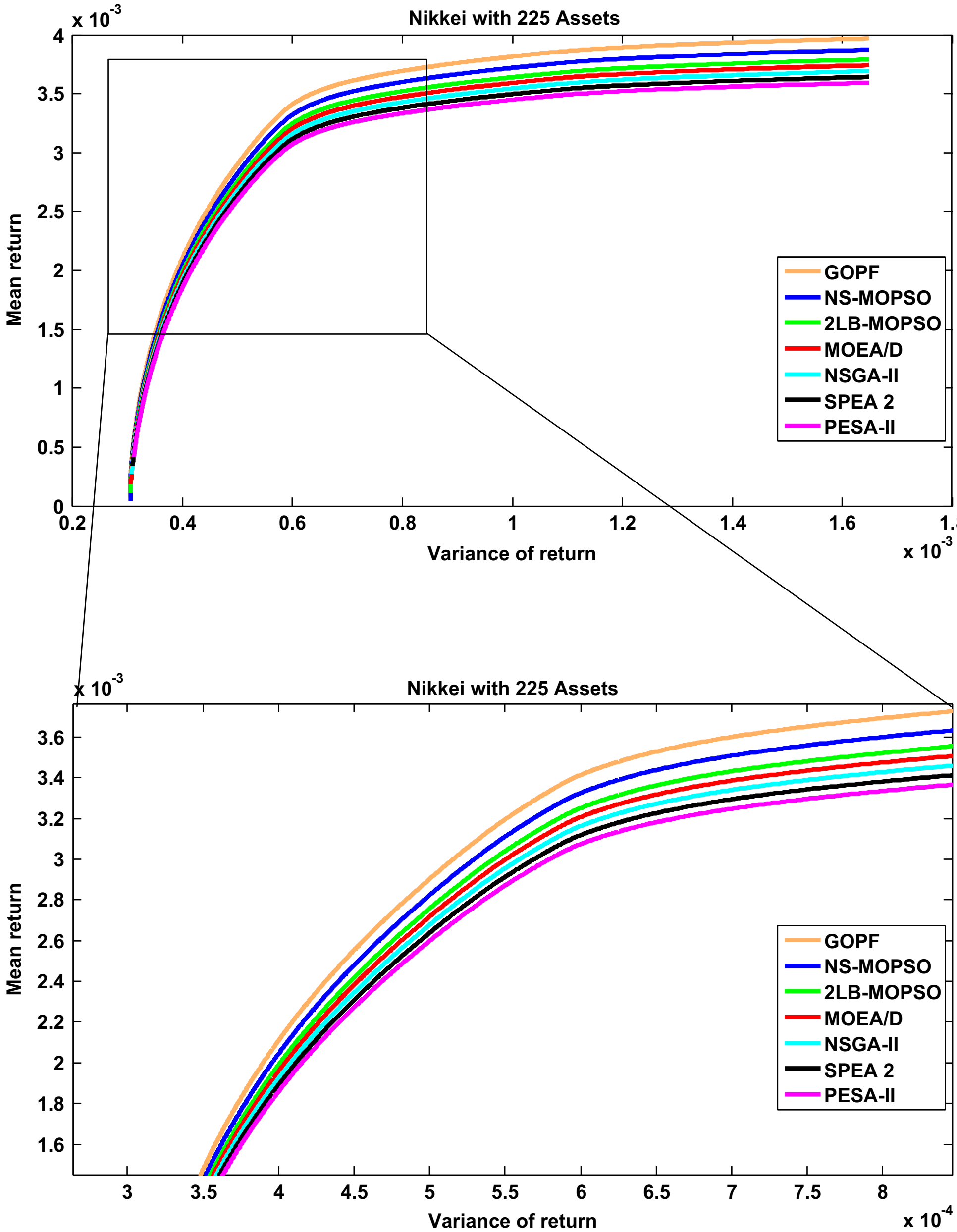


**Fig. 3.** Global optimal Pareto front and MOEAs efficient frontiers for Nikkei 225 stock indices.

**Table 1**
Experimental results of all algorithms to five markets.

| Index | Assets | Error | GA | TS | SA | PSO | PESA-II | SPEA 2 | NSGA-II | 2LB-MOPSO | MOEA/D | NS-MOPSO |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Hang-Seng | 31 | Mean Euclidian distance | 0.004 | 0.004 | 0.004 | 0.0049 | 0.0044 | 0.0042 | 0.004 | 0.004 | 0.004 | 0.004 |
| | | Variance of return error | 1.6441 | 1.6578 | 1.6628 | 2.2421 | 1.5233 | 1.4877 | 1.3266 | 1.2789 | 1.2766 | 1.284 |
| | | Mean return error (%) | 0.6072 | 0.6107 | 0.6238 | 0.7427 | 0.762 | 0.6899 | 0.6472 | 0.601 | 0.6015 | 0.6021 |
| DAX 100 | 85 | Mean Euclidian distance | 0.0076 | 0.0082 | 0.0078 | 0.009 | 0.0098 | 0.0084 | 0.0077 | 0.0074 | 0.0075 | 0.0075 |
| | | Variance of return error | 7.218 | 9.039 | 8.5485 | 6.8588 | 9.2819 | 8.2432 | 7.1211 | 6.5056 | 6.6078 | 6.7543 |
| | | Mean return error (%) | 1.2791 | 1.9078 | 1.2817 | 1.5885 | 2.2212 | 1.5922 | 1.2634 | 1.2566 | 1.2601 | 1.2671 |
| FTSE 100 | 89 | Mean Euclidian distance | 0.002 | 0.0021 | 0.0021 | 0.0022 | 0.0024 | 0.0022 | 0.0021 | 0.0018 | 0.0019 | 0.0019 |
| | | Variance of return error | 2.866 | 4.0123 | 3.8205 | 3.0596 | 5.2381 | 3.7652 | 2.9871 | 2.7899 | 2.8022 | 2.812 |
| | | Mean return error (%) | 0.3277 | 0.3298 | 0.3304 | 0.364 | 0.4023 | 0.3652 | 0.3329 | 0.32 | 0.322 | 0.325 |
| S&P 100 | 98 | Mean Euclidian distance | 0.0041 | 0.0041 | 0.0041 | 0.0052 | 0.0056 | 0.0049 | 0.0042 | 0.0041 | 0.0041 | 0.004 |
| | | Variance of return error | 3.4802 | 5.7139 | 5.4247 | 3.9136 | 7.0122 | 5.4323 | 3.7629 | 3.4588 | 3.4677 | 3.4763 |
| | | Mean return error (%) | 1.2258 | 0.7125 | 0.8416 | 1.404 | 2.4232 | 1.2109 | 0.7321 | 0.7001 | 0.702 | 0.7021 |
| Nikkei | 225 | Mean Euclidian distance | 0.0093 | 0.001 | 0.001 | 0.0019 | 0.0101 | 0.0032 | 0.001 | 0.0009 | 0.0009 | 0.0008 |
| | | Variance of return error | 1.2056 | 1.2431 | 1.2017 | 2.4274 | 3.0986 | 2.0421 | 1.1232 | 1.1138 | 1.1189 | 0.9876 |
| | | Mean return error (%) | 5.3266 | 0.427 | 0.4126 | 0.7997 | 1.2314 | 0.8654 | 0.4325 | 0.4011 | 0.411 | 0.3276 |

constraint), then the adjusted portfolio weights are computed using Eq. (28)

$$W_{adjusted} = a_i.z_i + \frac{w_i.z_i}{\sum_{i=1}^{n} w_i.z_i}\left(1 - \sum_{i=1}^{n} a_i.z_i\right), \quad i = 1, \ldots, n. \quad (28)$$

*Case* 3: If the weight has to be adjusted to the ceiling constraint, and there would be no restriction on lower limit (floor constraint). Then the adjusted portfolio weights are computed using the following equation:

$$W_{adjusted} = b_i.z_i - \frac{w_i.z_i}{\sum_{i=1}^{n} w_i.z_i}(b_i.z_i), \quad i = 1, \ldots, n. \quad (29)$$

### 5.2. Parameters used in the simulation of MOEAs

The conceptual framework for parameter tuning of different evolutionary algorithms is presented in [24]. For all the six MOEAs, the population size and number of generations are taken as 100 and 10,000 respectively. In the NS-MOPSO, 2LB-MOPSO and MOEA/D, the position of each particle represents a weight vector associated with different assets. For three MOEAs based on the genetic algorithm, such as such as PESA-II, SPEA 2 and NSGA-II, one chromosome represents one set of weights of assets, and each gene represents the weight of one asset. The dimensions of the search space depend on the number of assets of the stock. After several experiments with different parameters, the final parameters of the fine-tuned algorithms are as follow.

*PESA-II*: The internal and external population size is taken as 50, uniform crossover is taken having a rate of 0.8. It has a mutation rate of $1/L$, where $L$ refers to the length of the chromosome string that encodes the decision variables. The grid size, i.e., the number of divisions per dimension is set at 10.

*SPEA* 2: The crossover is taken as uniform. The crossover and mutation rate are taken as 0.8 and 0.05 respectively. The archive size is fixed at 50.

*NSGA-II*: The uniform crossover rate and mutation rate are taken as 0.8 and 0.05 respectively.

*NS-MOPSO*: Velocity has the probability of 0.5 being specified in a different direction. The upper and lower bounds of the decision variable range $V_{UPP}$ and $V_{LOW}$ are fixed at 0.06 and 0.5 respectively. The parameter $w = 0.862$ and $C_1 = C_2 = 2.05$.

*2LB-MOPSO*: The parameter $w = 0.862$, $C_1 = C_2 = 2.05$. Each objective function range in the external archive is divided into a number of bins, i.e.,*n_bin* and it is set to 10.

*MOEA/D*: Each subproblem of MOEA/D has been optimized, using the particle swarm optimization. The parameter $w = 0.862$ and $C_1 = C_2 = 2.05$.

### 5.3. Nonparametric statistical tests for the comparison of algorithms

The interest in a nonparametric statistical analysis has grown recently, for comparing evolutionary and swarm intelligence algorithms [40]. Pairwise comparisons are the simplest kind of statistical tests which can be applied within the framework of an experimental study. Such tests compare the performance of two algorithms, when applied to a common set of problems. In the

**Table 2**
The performance evaluation metrics for different MOEAs.

| Algorithm | PESA-II | SPEA 2 | NSGA-II | 2LB-MOPSO | MOEA/D | NS-MOPSO |
|---|---|---|---|---|---|---|
| *S* | | | | | | |
| Max | 3.21E−5 | 7.43E−6 | 6.54E−6 | 5.12E−6 | 5.93E−6 | 5.22E−6 |
| Min | 1.87E−5 | 5.23E−6 | 3.98E−6 | 2.58E−6 | 2.51E−6 | 2.33E−6 |
| Avg. | 2.33E−5 | 6.36E−6 | 4.74E−6 | 3.53E−6 | 3.62E−6 | 3.48E−6 |
| Std. | 0.58E−5 | 1.58E−6 | 1.53E−6 | 0.82E−6 | 0.87E−6 | 0.85E−6 |
| *GD* | | | | | | |
| Max | 2.54E−2 | 2.01E−3 | 7.23E−4 | 2.71E−4 | 2.63E−4 | 2.16E−4 |
| Min | 1.01E−2 | 0.89E−3 | 5.23E−4 | 1.04E−4 | 1.05E−4 | 1.02E−4 |
| Avg. | 1.76E−2 | 1.02E−3 | 6.72E−4 | 1.86E−4 | 1.76E−4 | 1.45E−4 |
| Std. | 0.42E−2 | 0.28E−3 | 1.48E−4 | 0.32E−4 | 0.57E−4 | 0.36E−4 |
| *Δ* | | | | | | |
| Max | 6.78E−1 | 4.34E−1 | 3.34E−1 | 2.52E−1 | 2.54E−1 | 2.45E−1 |
| Min | 4.23E−1 | 2.89E−1 | 1.89E−1 | 1.02E−1 | 1.20E−1 | 0.99E−1 |
| Avg. | 5.93E−1 | 3.86E−1 | 2.96E−1 | 1.42E−1 | 1.45E−1 | 1.33E−1 |
| Std. | 1.48E−1 | 0.93E−1 | 0.78E−1 | 0.44E−1 | 0.46E−1 | 0.43E−1 |

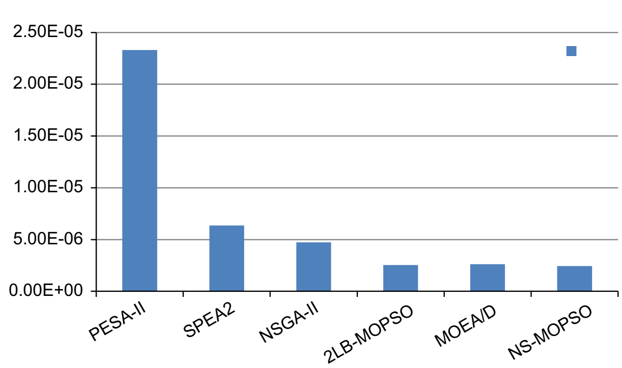


**Fig. 4.** Average value of *S* metric for MOEAs algorithms.

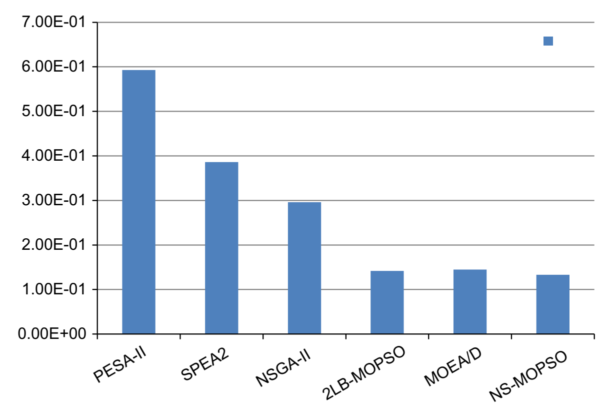


**Fig. 5.** Average value of *Δ* metric for MOEAs algorithms.

**Table 3**
The results obtained for *C* metric for different MOEAs.

| | PESA-II | SPEA 2 | NSGA-II | 2LB-MOPSO | MOEA/D | NS-MOPSO |
|---|---|---|---|---|---|---|
| PESA-II | – | 0.3810 | 0.2230 | 0.1881 | 0.1987 | 0.1880 |
| SPEA-II | 0.7280 | – | 0.3280 | 0.2478 | 0.2612 | 0.2380 |
| NSGA-II | 0.8530 | 0.7620 | – | 0.3444 | 0.3510 | 0.3430 |
| 2LB-MOPSO | 0.9080 | 0.8010 | 0.7810 | – | 0.4521 | 0.4412 |
| MOEA/D | 0.9010 | 0.7980 | 0.7710 | 0.4744 | – | 0.4510 |
| NS-MOPSO | 0.9090 | 0.8020 | 0.7800 | 0.4434 | 0.4680 | – |

**Table 4**
Comparison of CPU time in seconds among different markets using MOEAs.

| Algorithms | | PESA-II | SPEA-II | NSGA-II | 2LB-MOPSO | P-MOEA/D | NS-MOPSO |
|---|---|---|---|---|---|---|---|
| CPU time | Hang-Seng | 685 | 708 | 675 | 673 | 443 | 671 |
| | DAX 100 | 1606 | 1653 | 1586 | 1570 | 1033 | 1566 |
| | FTSE 100 | 1621 | 1669 | 1601 | 1585 | 1048 | 1582 |
| | S&P-100 | 1641 | 1680 | 1617 | 1600 | 1070 | 1602 |
| | Nikkie-225 | 4820 | 4960 | 4760 | 4720 | 3100 | 4700 |

present case, the Sign test and Wilcoxon signed rank test [40] are carried out to compare the performance pairwise. In the simulation work, the two tests are carried out by comparing all the MOEAs algorithms with the NS-MOPSO algorithms. The Sign test requires counting the number of wins achieved, either by the NS-MOPSO or by the comparison algorithm. The Wilcoxon signed rank test is analogous to the paired *t*-test in nonparametric statistical procedure [40]. The aim of the Wilcoxon signed rank test is to detect the difference between the behaviors of two algorithms.

### 5.4. Performance measure metrics

The four different metrics defined in the sequel are used during the investigation for measuring the performance quality.

*Generation distance* (*GD*): It estimates the distance of elements of the non-dominated vectors found from the standard efficient frontier, and is mathematically [36,37] expressed as

$$GD = \frac{\sqrt{\sum_{i=1}^{n} d_i^2}}{n} \tag{30}$$

where $n$ is the number of vectors in the set of obtained non-dominated solutions. $d_i$ is the Euclidean distance between each of these, and the nearest member of the standard efficient frontier. If $GD=0$, all the candidate solutions are in the standard efficient frontier. The smaller the value of *GD*, the closer the solution to the standard efficient frontier.

*Spacing* (S): It measures the spread of candidate solutions throughout the non-dominated vectors found. This metric [36] is mathematically expressed as

$$S \triangleq \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(\bar{d}-d_i)^2} \tag{31}$$

where

$$d_i = \min_j(|f_1{}^i(\vec{x}) - f_1{}^j(\vec{x})| + |f_2{}^i(\vec{x}) - f_2{}^j(\vec{x})|) \quad \text{and} \quad i,j = 1,2,\ldots,n$$

$\bar{d}$=mean of all $d_i$ and $n$ is the number of non-dominated vectors found so far.

A value of zero for this metric indicates that all members of the Pareto front currently available are equidistantly spaced.

*Diversity metric* ($\Delta$): This metric measures the extent of the spread, i.e., how evenly the points are distributed among the approximation sets in objective space. This metric does not require any standard efficient frontier, and has a relation with the Euclidean distance between solutions. It is defined [19] as

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{N-1}|d_i - \bar{d}|}{d_f + d_l + (n-1)\bar{d}} \tag{32}$$

where $d_i$ is the Euclidean distance between consecutive solutions in the obtained non-dominated set of solutions. $\bar{d}$ is the average of these distances $d_i$. $d_f$ and $d_l$ are the Euclidean distance between the extreme solutions and the boundary solutions of the obtained non-dominated set. $n$ is the number of solutions from the non-dominated set. The low value of $_\Delta$ indicates a better diversity of the non-dominated solution. Its value for the most widely and uniformly spread out set of non-dominated solutions is zero.

*Convergence metric* ($C$): This metric compares the quality of two non-dominated sets. This matrix is computed without taking the standard efficient frontier into consideration. Let $A$ and $B$ be two different sets of non-dominated solutions; then, the $C$ metric [36] is mathematically expressed as

$$C(A,B) = \frac{|\{b \in B | \exists a \in A : ab\}|}{|B|} \tag{33}$$

where $a$ and $b$ are candidate solutions of sets $A$ and $B$ respectively. The function $C$ maps the ordered pair $(A,B)$ to the interval [0, 1]. If $C(A,B)=1$, all the candidate solutions in $B$ are dominated by at least one solution in $A$. Similarly if $C(A,B)=0$, no candidate solutions in $B$ is dominated by any solution in $A$.

### 5.5. Experimental results

All the experiments have been simulated with a set of benchmark data available online, and obtained from OR-Library being maintained by Prof. Beasley [39]. The same test data have also been used in [6]. The data corresponds to the weekly prices between March 1992 and September 1997 from different well known indices, such as the Hang-Seng in Hong Kong, DAX 100 in Germany, FTSE 100 in UK, S&P 100 in USA, and Nikkei 225 in Japan. The numbers of different assets (stocks) for the above benchmark indices are 31, 85, 89, 98 and 225 respectively. In the PORT-1 to PORT-5 data sets, the returns of the individual assets and the correlation between assets are given for these five indices respectively. The covariance between the assets, evaluated from the correlation matrix, is used for calculating the risk of the portfolio.

The data for the global optimal Pareto fronts (GOPF), also called as the standard efficient front for each of these five stock indices,

**Table 6**
Critical values for the two-tailed sign test at $\alpha=0.05$ and $\alpha=0.1$ using $S$ metric as a winning parameter.

| NS-MOPSO | PESA-II | SPEA-II | NSGA-II | 2LB-MOPSO | MOEA/D |
|---|---|---|---|---|---|
| Wins (+) | 21 | 20 | 17 | 13 | 14 |
| Loses (−) | 4 | 5 | 8 | 12 | 11 |
| Detected differences | $\alpha=0.05$ | $\alpha=0.05$ | $\alpha=0.01$ | – | – |

**Table 7**
Critical values for the two-tailed sign test at $\alpha=0.05$ and $\alpha=0.1$ using $\Delta$ metric as a winning parameter.

| NS-MOPSO | PESA-II | SPEA-II | NSGA-II | 2LB-MOPSO | MOEA/D |
|---|---|---|---|---|---|
| Wins (+) | 19 | 18 | 17 | 16 | 13 |
| Loses (−) | 6 | 7 | 8 | 9 | 12 |
| Detected differences | $\alpha=0.05$ | $\alpha=0.05$ | $\alpha=0.01$ | – | – |

**Table 8**
Critical values for the two-tailed sign test at $\alpha=0.05$ and $\alpha=0.1$ using *GD* metric as a winning parameter.

| NS-MOPSO | PESA-II | SPEA-II | NSGA-II | 2LB-MOPSO | MOEA/D |
|---|---|---|---|---|---|
| Wins (+) | 20 | 19 | 17 | 15 | 14 |
| Loses (−) | 5 | 6 | 8 | 10 | 11 |
| Detected differences | $\alpha=0.05$ | $\alpha=0.05$ | $\alpha=0.01$ | – | – |

**Table 5**
Critical values for the two-tailed sign test at $\alpha=0.05$ and $\alpha=0.1$.

| No. of cases | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $\alpha=0.05$ | 5 | 6 | 7 | 7 | 8 | 9 | 9 | 10 | 10 | 11 | 12 | 12 | 13 | 13 | 14 | 15 | 15 | 16 | 17 | 18 | 18 |
| $\alpha=0.1$ | 5 | 6 | 6 | 7 | 7 | 8 | 9 | 9 | 10 | 10 | 11 | 12 | 12 | 13 | 13 | 14 | 14 | 15 | 16 | 16 | 17 |

are available in files PORTEF-1 to PORTEF-5. The data corresponds to the trade-off points between the risk (variance of return) and the return (mean return). The GOPF is plotted by joining these points. These trade-off points have been calculated by a rigorous method, and are called as global optimal points, as they are the best possible trade-off points between the risk and the return. The GOPF corresponding to the Nikkei 225 stock indices is depicted in Fig. 2 which shows the trade-off between the risk (variance of return) and the return (mean return).

The proposed NS-MOPSO algorithm and five other MOEAs are applied to the Nikkei 225 stock indices. The population sizes of all these MOEAs are taken as 100. Hence, the final solutions of each algorithm are 100 trade-off points between the risk and the return. Hence, there are 100 possible sets of participation of assets and proportions of the budget invested in them. The Pareto front is obtained by plotting these 100 trade-off points. Each individual point is one solution to the problem. The Pareto front obtained by all these algorithms and the GOPF are shown in Fig. 3. It is evident that the proposed NS-MOPSO algorithm is capable of providing better solutions in comparison to the other five algorithms, as its Pareto front is closer to the GOPF.

The effects of the budget, floor, ceiling and cardinality constraints have been analyzed by examining the resultant Pareto front achieved. The theoretical implementation of constraint is that it limits the portfolio size, and hence, influences the level of risk and return. The cardinality constraint is taken as $K=10$, the floor constraint has been set at $\varepsilon=0.01$ and the ceiling constraint is fixed at $\delta=1$ with all the available money that has to be invested. For this constraint condition, the performance of all the MOEAs has been compared with the results obtained, using single objective evolutionary algorithms (SOEAs) such as the GA, TS, SA and PSO as given in [6]. The results of the five benchmark stock indices for all the algorithms are listed in Table 1. The experimental results of these error measures reveal that the performance of the 2LB-MOPSO and MOEA/D algorithms are almost comparable to, and better than that of all other SOEAs as well as MOEAs for the Hang-Seng, DAX 100, FTSE 100, and S&P 100 benchmark indices, having 31, 85, 89, and 98 different assets. However, when the number of assets is more, as in the case of Nikkei 225, the proposed NS-MOPSO algorithm performs the best, in terms of all error measures.

Further, the performance of all the six MOEAs is evaluated, using three different performance metrics, such as $S$,$GD$ and $\Delta$ metrics. Each algorithm is applied to the Nikkei 225 market for 25 independent runs. The parameters of the algorithms are the same as in the previous cases. The maximum, minimum, average and standard deviation values of $S$, $GD$ and $\Delta$ metrics for 25 independent runs are calculated, and shown in Table 2. The standard deviations of the three metrics obtained by the 2LB-MOPSO algorithm are the smallest, which indicate better consistency compared to the other algorithms. From the experimental results it is clear that the proposed NS-MOPSO algorithm performs the best, in terms of the mean value of metrics among all MOEAs. The mean value of metrics $S$ and $\Delta$ for the different MOEAs in graphical form is shown in Figs. 4 and 5. The convergence $C$ metrics for all the MOEAs are demonstrated in Table 3. It clearly shows that most of the solutions obtained by 2LB-MOPSO, MOEA/D and NS-MOPSO dominate the solutions obtained by the other three MOEAs.

The computational time is also evaluated for each algorithm based on the same hardware platform. The CPU times of all the algorithms for different stock indices are shown in Table 4, which indicates that the decomposition based MOEAs (MOEA/D) take much less time, compared to others. Among all the algorithms, the SPEA 2 takes the maximum time.

The nonparametric statistical tests, such as the Sign test and Wilcoxon signed rank test, are demonstrated for the pairwise comparisons of the proposed algorithms with any other MOEAs. The critical number of wins needed to achieve both $\alpha=0.05$ and $\alpha=0.1$ levels of significance is shown in Table 5. An algorithm is

**Table 9**
Wilcoxon signed test using $S$ metric as a winning parameter and applying different MOEAs to Nikkie 225 market indices.

| Comparison | $R^+$ | $R^-$ | $z$ | Asymp. sig (2-tailed) | Exact sig. (2-tailed) | Exact sig. (1-tailed) | Point of probability |
|---|---|---|---|---|---|---|---|
| NS-MOPSO with PESA-II | 252 | 73 | −2.410 | 0.016 | 0.014 | 0.007 | 0.000 |
| NS-MOPSO with SPEA-II | 231 | 94 | −1.845 | 0.065 | 0.065 | 0.033 | 0.001 |
| NS-MOPSO with NSGA-II | 211.5 | 113.5 | −1.319 | 0.187 | 0.193 | 0.096 | 0.002 |
| NS-MOPSO with 2LB-MOPSO | 160 | 165 | −0.067 | 0.946 | 0.953 | 0.476 | 0.005 |
| NS-MOPSO with MOEA/D | 168 | 152 | −0.148 | 0.882 | 0.810 | 0.445 | 0.005 |

**Table 10**
Wilcoxon signed test using $\Delta$ metric as a winning parameter and applying different MOEAs to Nikkie 225 market indices.

| Comparison | $R^+$ | $R^-$ | $z$ | Asymp. sig (2-tailed) | Exact sig. (2-tailed) | Exact sig. (1-tailed) | Point of probability |
|---|---|---|---|---|---|---|---|
| NS-MOPSO with PESA-II | 214 | 111 | −1.387 | 0.165 | 0.170 | 0.085 | 0.002 |
| NS-MOPSO with SPEA-II | 196 | 129 | −0.902 | 0.367 | 0.377 | 0.188 | 0.004 |
| NS-MOPSO with NSGA-II | 194 | 131 | −0.848 | 0.396 | 0.407 | 0.203 | 0.004 |
| NS-MOPSO with 2LB-MOPSO | 191.5 | 133.5 | −0.781 | 0.435 | 0.445 | 0.223 | 0.004 |
| NS-MOPSO with MOEA/D | 186.5 | 168.5 | −0.162 | 0.872 | 0.879 | 0.440 | 0.005 |

**Table 11**
Wilcoxon signed test using $GD$ metric as a winning parameter and applying different MOEAs to Nikkie 225 market indices.

| Comparison | $R^+$ | $R^-$ | $z$ | Asymp. sig (2-tailed) | Exact sig. (2-tailed) | Exact sig. (1-tailed) | Point of probability |
|---|---|---|---|---|---|---|---|
| NS-MOPSO with PESA-II | 242 | 83 | −2.142 | 0.032 | 0.031 | 0.015 | 0.001 |
| NS-MOPSO with SPEA-II | 229 | 96 | −1.791 | 0.073 | 0.074 | 0.037 | 0.001 |
| NS-MOPSO with NSGA-II | 192 | 133 | −0.794 | 0.427 | 0.437 | 0.219 | 0.004 |
| NS-MOPSO with 2LB-MOPSO | 185 | 139 | −0.619 | 0.539 | 0.546 | 0.273 | 0.004 |
| NS-MOPSO with MOEA/D | 168 | 157 | −0.148 | 0.882 | 0.890 | 0.445 | 0.005 |

significantly better than another, if it performs better on at least the cases presented in each row. All the MOEAs are applied to the Nikkie 225 market indices. The results of the Sign test for pairwise comparisons among the proposed NS-MOPSO and other algorithms, while taking the *S* metric as the wining parameter (i.e. lower value of *S* means win) are shown in Table 6. From the results it is clear that the NS-MOPSO shows significant improvement over the PESA-II and SPEA 2 algorithms with a level of significance $\alpha=0.05$, and over NSGA-II, with a level of significance $\alpha=0.1$. Similarly, for the other metrics *GD* and $\Delta$, the results of the Sign test are listed in Tables 7 and 8. The Wilcoxon signed rank test is carried out by calculating $R^+$ and $R^-$, and then using the well-known statistical software package SPSS. Table 9 shows the $R^+$,$R^-$, *z*, Asymp. sig. (2-tailed), Exact sig. (2-tailed), Exact sig. (1-tailed) and point of probability computed for all the pairwise comparisons, with the NS-MOPSO considering the *S* metric as the winning parameter, and applying it to the Nikkie 225 market indices. The results of the Wilcoxon signed rank test for the other two metrics *GD* and $\Delta$ are reported in Tables 10 and 11.

The presence of different cardinality constraints *K* is also studied in this work. The Pareto fronts obtained by applying the NS-MOPSO for the Nikkei 225 data set having different cardinalities are presented in Fig. 5. The value of *K* is set at 20, and is increased to 180 in steps of 20. The portfolio manager has the option to make a trade-off between the risk and the returns for different values of *K*. The average and standard deviation values of the various performance metrics are shown in Table 12. It is observed that when *K* increases, these metric values also increase. Table 13 lists the results of the convergence metric '*C*'. It shows that the final solutions obtained at $K=20$ dominate the solutions obtained at *K* more than 20. The CPU time for various values of *K* is shown in Table 14. It reveals that the computation time increases with the increase in the value of *K*. From Fig. 6 it is seen that the Pareto front becomes shorter with an increase in the *K* value. Hence, the proposed algorithm is able to obtain a near optimal solution efficiently, by investing lower number of assets, i.e., approximately 10% of the available assets. The Pareto front of NS-MOPSO is also calculated for other stock indices, for different *K*, and are depicted in Figs. 7–10.

The proposed algorithm is also applied to the BSE-500 (Bombay Stock Exchange) of India [41]. The raw weekly prices of 50 stocks (assets) from 500 stocks are collected. The time series of weekly returns is calculated mathematically from the weekly prices for each stock. The expected return is also computed by calculating the mean of the past returns. The individual risk of each stock and the risk between each pair of stocks are obtained, from the variance and covariance of the time series of return. After calculating the return and risk for each assets as well as the pair of assets, the portfolio assets selection task is carried out by using our proposed NS-MOPSO and other MOEAs. The Pareto fronts obtained by all these algorithms are shown in Fig. 11. The presence of the cardinality constraints *K* is also tested, by applying the proposed algorithm to the BSE stock exchange. The floor constraint has been set at $\varepsilon=0.01$, the ceiling constraint is fixed at $\delta=1$, and with no budget constraint, all the available money can be invested. For this constraint condition, in the analysis, the value of *K* is varied from 5 to 40, with a step size of 5. The Pareto fronts obtained for these cardinality conditions are depicted in Fig. 12.

## 6. Conclusions and further work

A novel multiobjective PSO algorithm, the NS-MOPSO, has been applied to realistic portfolio assets selection problems, with the budget, floor, ceiling and cardinality constraints, by formulating it as a multiobjective optimization problem. The performance of the proposed approach is compared with that of four single objective evolutionary algorithms, such as the genetic algorithm (GA), tabu

**Table 12**
Comparison of various performance metrics at different cardinality constraints.

| Cardinality constraint | Metrics values | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *S* | | | | $\Delta$ | | | | *GD* | | | |
| | Max. | Min. | Avg. | Std. dev. | Max. | Min. | Avg. | Std. dev. | Max. | Min. | Avg. | Std. dev. |
| $K=0$ | 5.21E−6 | 2.32E−6 | 3.43E−6 | 0.85E−6 | 2.54E−1 | 1.87E−1 | 2.21E−1 | 0.67E−1 | 2.11E−4 | 1.10E−4 | 1.45E−4 | 0.36E−4 |
| $K=20$ | 7.21E−6 | 3.45E−6 | 5.64E−6 | 1.41E−6 | 4.43E−1 | 3.21E−1 | 3.76E−1 | 0.88E−1 | 3.42E−4 | 1.98E−4 | 2.45E−4 | 0.82E−4 |
| $K=40$ | 9.21E−6 | 5.32E−6 | 7.77E−6 | 2.12E−6 | 4.54E−1 | 2.99E−1 | 3.94E−1 | 0.98E−1 | 4.23E−4 | 2.98E−4 | 3.42E−4 | 1.02E−4 |
| $K=60$ | 5.32E−5 | 3.45E−5 | 4.43E−5 | 1.15E−5 | 5.15E−1 | 3.98E−1 | 4.54E−1 | 1.18E−1 | 5.24E−4 | 3.32E−4 | 4.15E−4 | 1.32E−4 |
| $K=80$ | 8.76E−5 | 5.68E−5 | 7.65E−5 | 1.92E−5 | 6.76E−1 | 4.99E−1 | 5.78E−1 | 1.45E−1 | 3.42E−3 | 1.89E−3 | 2.22E−3 | 0.55E−3 |
| $K=100$ | 9.21E−5 | 7.65E−5 | 8.88E−5 | 2.52E−5 | 7.43E−1 | 5.76E−1 | 6.68E−1 | 1.67E−1 | 4.23E−3 | 2.98E−3 | 3.88E−3 | 0.97E−3 |
| $K=120$ | 7.23E−4 | 3.21E−4 | 5.21E−4 | 1.32E−4 | 8.76E−1 | 6.55E−1 | 7.88E−1 | 1.94E−1 | 5.56E−3 | 3.98E−3 | 4.54E−3 | 1.36E−3 |
| $K=140$ | 9.76E−4 | 7.65E−4 | 8.45E−4 | 2.12E−4 | 9.87E−1 | 7.65E−1 | 8.64E−1 | 2.12E−1 | 2.23E−2 | 1.10E−2 | 1.42E−2 | 0.33E−2 |
| $K=160$ | 3.21E−3 | 1.98E−3 | 2.65E−3 | 0.89E−3 | 10.1E−1 | 8.7E−1 | 9.53E−1 | 2.88E−1 | 3.80E−2 | 1.86E−2 | 2.02E−2 | 0.52E−2 |
| $K=180$ | 4.32E−3 | 2.32E−3 | 3.12E−3 | 0.91E−3 | 10.4E−1 | 7.98E−1 | 9.99E−1 | 2.98E−1 | 4.53E−2 | 2.67E−2 | 3.12E−2 | 1.01E−2 |

**Table 13**
Comparison of results of convergence (*C*) metric for different cardinality constraints.

| Cardinality constraint | $K=20$ | $K=40$ | $K=60$ | $K=80$ | $K=100$ | $K=120$ | $K=140$ | $K=160$ | $K=180$ |
|---|---|---|---|---|---|---|---|---|---|
| $K=20$ | – | 0.2610 | 0.3400 | 0.4660 | 0.5970 | 0.6580 | 0.7200 | 0.7810 | 0.8600 |
| $K=40$ | 0.0890 | – | 0.3020 | 0.4220 | 0.5690 | 0.6260 | 0.7060 | 0.7670 | 0.8420 |
| $K=60$ | 0.0840 | 0.2420 | – | 0.3840 | 0.5322 | 0.5860 | 0.6810 | 0.7420 | 0.8280 |
| $K=80$ | 0.0810 | 0.2250 | 0.2620 | – | 0.5020 | 0.5590 | 0.6640 | 0.7220 | 0.8020 |
| $K=100$ | 0.0770 | 0.2040 | 0.2420 | 0.3680 | – | 0.5220 | 0.6430 | 0.7040 | 0.7840 |
| $K=120$ | 0.0740 | 0.1880 | 0.2240 | 0.3440 | 0.4740 | – | 0.6210 | 0.6810 | 0.7600 |
| $K=140$ | 0.0710 | 0.1560 | 0.1990 | 0.3260 | 0.4420 | 0.4920 | – | 0.6620 | 0.7380 |
| $K=160$ | 0.0670 | 0.1250 | 0.1640 | 0.3010 | 0.4170 | 0.4480 | 0.5920 | – | 0.7040 |
| $K=180$ | 0.0590 | 0.1080 | 0.1280 | 0.2790 | 0.03820 | 0.4200 | 0.5680 | 0.6390 | – |

**Table 14**
Comparison of CPU time in seconds.

| Number of cardinalities | $K=20$ | $K=40$ | $K=60$ | $K=80$ | $K=100$ | $K=120$ | $K=140$ | $K=160$ | $K=180$ |
|---|---|---|---|---|---|---|---|---|---|
| CPU time in second | 4910 | 5320 | 5730 | 6190 | 6680 | 7020 | 7490 | 7830 | 8440 |

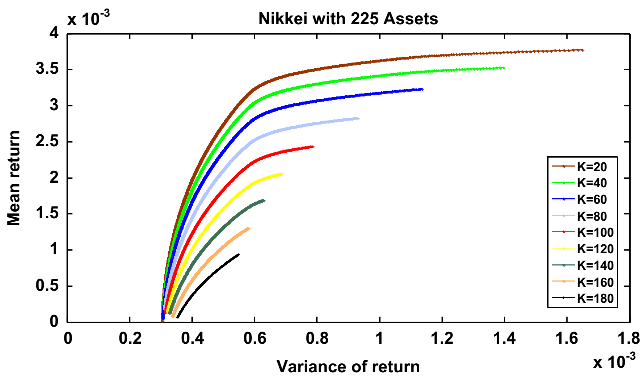


**Fig. 6.** NS-MOPSO efficient frontier for different cardinalities for Nikkei 225 data.

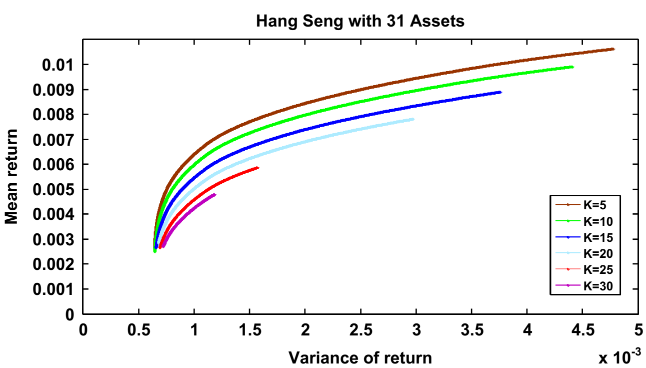


**Fig. 7.** NS-MOPSO efficient frontier for different cardinalities for Hang-Sang data.

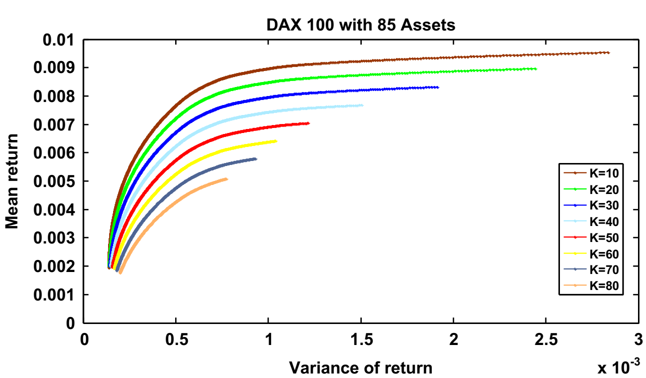


**Fig. 8.** NS-MOPSO efficient frontier for different cardinalities for DAX 100 data.

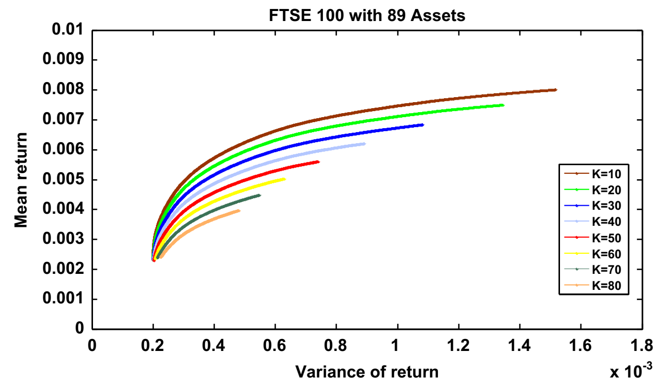


**Fig. 9.** NS-MOPSO efficient frontier for different cardinalities for FTSE 100 data.

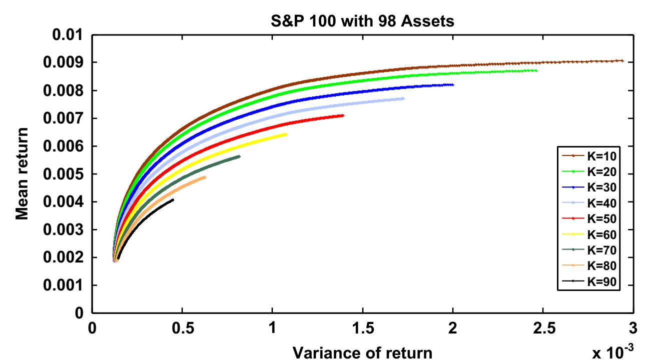


**Fig. 10.** NS-MOPSO efficient frontier for different cardinalities for S & P 100 data.

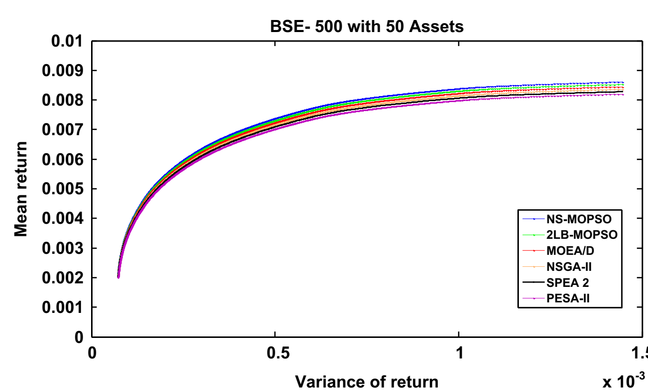


**Fig. 11.** MOEAs efficient frontiers for BSE-500 stock indices.

search (TS), simulated annealing (SA), particle swarm optimization (PSO) and a set of competitive multiobjective evolutionary algorithms (MOEAs) having non-dominated sorting based or decomposition-based frameworks. The comparisons include the evaluation of three error measures, four performance metrics, the Pareto front and computational time. The Sign test and Wilcoxon signed rank test are also performed to establish the superiority of the NS-MOPSO over others. The simulation results demonstrate the significant improvement of the proposed one over the PESA-II and SPEA 2 algorithms with a level of significance $\alpha=0.05$, and over NSGA-II, with a level of significance $\alpha=0.1$.

The MOEAs are applied to six different market indices, such as the Hang-Seng in Hong Kong, DAX 100 in Germany, FTSE 100 in UK, S&P 100 in USA, Nikkei 225 in Japan and BSE-500 in India. The computational results of these markets exhibit the improved performance of the proposed NS-MOPSO algorithm, and hence, the proposed method has been proven to be a good candidate for solving the constrained portfolio assets selection problem. From the simulation results it is clear that the investor does not have to invest money on all the available assets; rather he can invest in a fewer assets, i.e., approximately 10% of the available assets, to explore a wide risk-return area. The portfolio manager has the

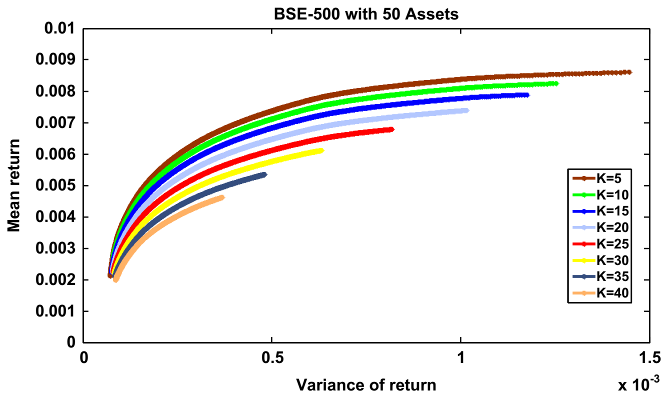


**Fig. 12.** NS-MOPSO efficient frontier for different cardinalities for BSE-500 data.

option to make a trade-off between risk and return for different cardinality constraints, to decide on the portfolio according to his requirement.

Future research work on the topic includes the incorporation of advanced local search operators into the proposed algorithm model, which is expected to allow better exploration and exploitation of the search space. To assess the strengths and weaknesses of non-dominated sorting based or decomposition based MOEAs frameworks, further investigation is needed. The performance of proposed NS-MOPSO algorithm can also be evaluated by considering other real-world constraints, like round-lot, turnover and trading. The same multiobjective optimization algorithm can also be applied to other financial applications, such as asset allocation, risk management, option pricing etc.

## References


[1] H.M. Markowitz, Portfolio selection, J. Financ. 7 (1952) 77–791.
[2] H.M. Markowitz, Portfolio selection: efficient diversification of investments, Yale University Press, John Wiley & Sons, New York, 1991.
[3] G.D.I. Tollo, A. Roli, Metaheuristics for the portfolio selection problem, Int. J. Oper. Res. 5 (1) (2008) 13–35.
[4] Y. Crama, M. Schyns, Simulated annealing for complex portfolio selection problems, Eur. J. Oper. Res. 150 (3) (2003) 546–571.
[5] T.J. Chang, N. Meade, J.E. Beasley, Y.M. Sharaiha, Heuristics for cardinality constrained portfolio assets selection, Comput. Oper. Res. 27 (13) (2000) 1271–1302.
[6] Tunchan Cura, Particle swarm optimization approach to portfolio assets selection, Nonlinear Anal.: Real World Appl. 10 (4) (2009) 2396–2406.
[7] J. Gao, Z. Chu., A new particle swarm optimization based on MATLAB for portfolio selection problem, Int. J. Model. Identif. Control 9 (1 and 2) (2010) 206–211.
[8] Hamid Reza Golmakani, Mehrshad Fazel, Constrained portfolio selection using particle swarm optimization, Expert Syst. Appl. 38 (7) (2011) 8327–8335.
[9] Hanhong Zhu, Y.i. Wang, Kesheng Wang, Y.u.n. Chen, Particle swarm optimization (PSO) for theconstrained portfolio assets selection problem, Expert Syst. Appl. 38 (8) (2011) 10161–10169.
[10] A. Mukherjee, R. Biswas, K. Deb, A.P. Mathur., Multi-objective evolutionary algorithms for the risk–return trade-off in bank loan management, Int. Trans. Oper. Res. 9 (2002) 583–597.
[11] K. Doerner, W. Gutjahr, R. Hartl, C. Strauss, C. Stummer, Pareto ant colony optimization: a metaheuristic approach to multiobjective portfolio selection, Ann. Oper. Res. 131 (2004) 79–99.
[12] S.K. Mishra, G. Panda, S. Meher, Multiobjective particle swarm optimization approach to portfolio assets selection, in: IEEE World Congress on Nature and Biologically Inspired Computing (NaBIC), Coimbatore, India, 2009, pp. 1612–1615.
[13] S.K. Mishra, G. Panda, S. Meher, R. Majhi, Comparative performance evaluation of multiobjective optimization algorithm for portfolio management, Int. J. Comput. Vis. Robotics. 1 (2) (2010) 236–247 (Inderscience Publisher).
[14] H. Keller, D. Maringer, Optimization of cardinality constrained portfolios with a hybrid local search algorithm, OR Spectr. 25 (4) (2004) 481–495.
[15] R. Armananzas, J.A. Lozano,A multiobjective approach to the portfolio assets selection problem, in: IEEE World Congress on Evolutionary Computation, vol. 2, 2005, pp. 1388–1395.
[16] S.C. Chiam, A.Al. Mamunand Low Y.L. A realistic approach to evolutionary multiobjective portfolio assets selection, in: IEEE Congress on Evolutionary Computation (CEC), 2007, pp. 204–211.
[17] D.W. Corne, N.R. Jerram,J. Knowles. M.J. Oates, PESA-II: region-based selection in evolutionary multiobjective optimization, in: Proceedings of the Genetic and Evolutionary Computation Conference, San Francisco, CA 2001.
[18] E. Zitzler, M. Laumanns, L. Thiele, SPEA2: improving the strength Pareto evolutionary algorithm. Technical Report 103, Gloriastrasse 35, CH-8092 Zurich, Switzerland, 2001.
[19] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197.
[20] S.Z. Zhao, P.N. Suganthan, Two-lbests based multi-objective particle swarm optimizer, Eng. Optim. 43 (1) (2011) 1–17, http://dx.doi.org/10.1080/03052151003686716.
[21] Q. Zhang, H. Li, MOEA/D: a multi-objective evolutionary algorithm based on decomposition, IEEE Trans. Evol. Comput. 11 (6) (2007) 712–731.
[22] R.C. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in: Sixth International Symposium on Micromachine and Human Science, 1995; pp. 39–43.
[23] B.Y. Qu, P.N. Suganthan, Multi-objective evolutionary algorithms based on the summation of normalized objectives and diversified selection, Inf. Sci. 180 (17) (2010) 3170–3181.
[24] A.E. Eiben, S.K. Smit, Parameter tuning for configuring and analyzing evolutionary algorithms, Swarm Evol. Comput. 1 (1) (2011) 19–31.
[25] Vilfredo Pareto, D. Cours, Economie Politique. Vols. I and II. F. Rouge, Lausanne 1986.
[26] J.D. Schaffer, Multiple objective optimizations with vector evaluated genetic algorithms, in: Genetic Algorithms and Their Applications: Proceedings of the International Conference on Genetic Algorithm, Lawrence Erlbaum 1985, pp. 93–100.
[27] A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, P.N. Suganthan, Q. Zhang, Multiobjective evolutionary algorithms: a survey of the state-of-the-art, Swarm Evol. Comput. 1 (1) (2011) 32–49.
[28] C.A.C. Coello, T.P. Gregorio., S.L. Maximino., Handling multiple objectives with particle swarm optimization, IEEE Trans. Evol. Comput. 8 (3) (2004) 256–279.
[29] E. Mezura-Montes, C.A. Coello Coello, Constraint-handling in nature-inspired numerical optimization: past, present and future, Swarm Evol. Comput. 1 (4) (2011) 173–194.
[30] P.K. Tripathy, S. Bandyopadhyay, S.K. Pal, Multi-objective particle swarm optimization with time variant inertia and acceleration coefficients, Inf. Sci. (NY) 177 (22) (2007) 5033–5049.
[31] S. Agrawal., B.K. Panigrahi, M.K. Tiwari, Multiobjective particle swarm algorithm with fuzzy clustering for electrical power dispatch, IEEE Trans. Evolut. Comput. 12 (5) (2008) 529–541.
[32] N.A. Moubayed, A. Petrovski, J. McCall, A novel smart multi-objective particle swarm optimization using decomposition, in: Parallel Problem Solving from Nature, PPSN XI, Lecture Notes in Computer Science, vol. 6239, 2010, pp. 1–10.
[33] H. Li, Q. Zhang, Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II, IEEE Trans. Evol. Comput. 13 (2) (2009) 284–302.
[34] P.C. Chang, S.H. Chen, Q. Zhang, J.L. Lin, MOEA/D for flowshop scheduling problems, in: IEEE Congress on Evolutionary Computation, CEC, 2008, pp. 1433–1438.
[35] S. Pal, B. Qu, S. Das, P.N. Suganthan, Optimal synthesis of linear antenna arrays with multi-objective differential evolution, Prog. Electromag. Res. B 21 (2010) 87–111.
[36] E. Zitzler, Evolutionary algorithms for multiobjective optimization: methods and applications (Doctoral dissertation ETH 13398), SwissFederal Institute of Technology (ETH), Zurich, Switzerland, 1999.
[37] E. Zitzler, L. Thiele, M. Laumanns, C.M. Fonseca, V.G. da Fonseca, Performance assessment of multiobjective optimizers: an analysis and review, IEEE Trans. Evolut. Comput. 7 (2) (2003) 117–132.
[38] F. Streichert, H..Ulmer, A. Zell, Evaluating a hybrid encoding and three crossover operators on the constrained portfolio selection problem, in: Proceedings of the Congress on Evolutionary Computation, Portland, USA, vol. 1, 2004b, pp. 932–939.
[39] 〈http://people.brunel.ac.uk/~mastjjb/jeb/orlib/portinfo.html〉.
[40] J. Derrac, S. García, D. Molina, F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, Swarm Evolut. Comput. 1 (1) (2011) 3–18.
[41] 〈http://in.finance.yahoo.com/q/hp?s=%5EBSESN〉.