International Joint Conference on Information and Communication Technology (IJcICT-2010), Bhubaneswar, 9th-10th January, 2010

Multiobjective Evolutionary Algorithms for Financial Portfolio Design

Sudhansu Kumar Mishra, Ganapati Panda, Sukadev Meher,Ritanjali Majhi
[1,3]Dept of ECE, NIT, Rourkela,Odisha, INDIA
[2]School of Electrical Science, IIT, Bhubaneswar, India
[4]Centre for Management Studies, NIT, Warangal, India-506004
Sudhansu.nit@gmail.com,ganapati.panda@gmail.com,smeher@nitrkl.ac.in,ritanjalimajhi@gmail.com,

Abstract— Efficient portfolio design is a principal challenge in modern computational finance. Optimization based on Markowitz two-objective mean-variance approach is computationally expensive for real financial world. Practical portfolio design introduces further complexity as it requires the optimization of multiple return and risk measures. Some of these measures are nonlinear and nonconvex. The problem of portfolio design is a standard problem in financial world and has received a lot of attention. Three well known multi-objective evolutionary algorithms i.e. Pareto envelope-based selection algorithm , Micro Genetic algorithm and Multiobjective particle swarm optimization has been applied for solving the bi-objective portfolio optimization problem which simultaneously maximize the return measures and minimize the risk measures. Performance comparison carried out by performing different numerical experiments. The approach has been tested on real-life portfolio with many assets. The results show that MOPSO outperforms the existing method for the considered test cases.

Index Terms—Evolutionary algorithms,Multiobjective optimization, Pareto optimal solutions, Global optimization, Crowding distance, Portfolio optimization

I. Introduction

Portfolio design is very complicated as it depends on many factors such as assets inter- relationships, preferences of the decision makers, resource allocation and several other factors. As a result, the decision maker has to take several issues into consideration. Almost all practical optimization problems, especially economical design optimization problems have a multiobjective nature much more frequently than a single objective one. In this work we suggest the use of multiobjective optimization algorithms for optimal weighting of assets as a portfolio optimization problem. Choosing an optimal portfolio weighting of assets, when their future rate of return is uncertain is seen as a problem of minimizing the uncertainty for a given level of the portfolio expected return. This uncertainty is called as risk and measured by standard deviation of the probability distribution of future return. The risk of a particular investment is not as important as its contribution to total portfolio risk. Combining a riskful investment with one carrying less risk it is possible to reduce the total risk associated to that portfolio. Selecting an optimal portfolio weighting of available assets is main aim of portfolio design problem. These issues are conflicting which makes the problem as a multi-objective one. In this paper we used different multiobjective algorithms like PESA, MicroGA and MOPSO for modeling the Pareto front and for optimizing the portfolio performance. The results obtained with these three algorithms are finally compared by performing different numerical experiments.

Section 2 outlines the multi-objective optimization formulation of portfolio optimization. In Section 3 some of the multi-objective evolutionary techniques used in this paper are dealt. Multi-objective optimization fundamental is presented in section 3. The Multiobjective particle swarm optimization (MOPSO) algorithm is explained in section 4. In Section 5, some well know multi-objective evolutionary techniques are briefly described. For comparing different multiobjective algorithms, different metric proposed by different authors in the literature are presented in section 6.

Simulation studies based on several numerical experiments are carried out in Section 7. The results in terms Pareto fronts between risk and return are shown in Section 8. Conclusions and further research work directions are discussed in the final section.

2. Statement of the problem

A portfolio p be consists of N number of assets. Selection of optimal weighting of assets (with specific volumes for each asset given by weights w_i)is to be found. The unconstrained portfolio optimization problem is given as minimizing the variance of the portfolio and maximizing the return of the portfolio shown in equation 1 and 2 respectively.

$$\rho_p{}^2 = \sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j \sigma_{ij} \tag{1}$$

$$\alpha_p = \sum_{i}^{N} w_i \mu_i \tag{2}$$

$$\sum_{i}^{N} w_i = 1 \tag{3}$$

$$0 \le w_i \le 1; \text{ and } i = 1,2...,N \tag{4}$$

μ_i the expected return of asset i, σ_{ij} the covariance between asset i and j and finally w_i are the decision variables giving the composition of the portfolio. ρ_p be the standard deviation of portfolio and α_p be the expected return of portfolio. 3 and 4 give the constraints for this portfolio optimization problem. This is a multiobjective optimization problem with two competing objectives
(i) minimize the total variance, denoting the risk associated with the portfolio (ii) maximize the return of the portfolio.

3. Multi-objective optimization

In a single-objective optimization problem, an optimal solution is the one which optimizes the objective with certain associated constraints. It is not possible to find a single solution for a multiobjective problem and due to the contradictory objectives a set of solutions is obtained. The general multi-objective minimization problem involves minimization of n objective functions:

$$\left\{ f_1\left(\bar{x}\right), f_2\left(\bar{x}\right)........., f_n\left(\bar{x}\right)\right\}$$

Where $n \ge 2$ (5)

The solution to this problem is more complex than the single-objective case, and the idea of Pareto-dominance is used to explain it. Consider first an objective function $\bar{F}\left(\bar{x}\right)$, where

$$\bar{F}\left(\bar{x}\right) = \left\{ f_1\left(\bar{x}\right), f_2\left(\bar{x}\right)........., f_n\left(\bar{x}\right)\right\} \tag{6}$$

A point $\bar{x}_1$ with an objective function vector $\bar{F}_1$, is said to dominate point $\bar{x}_2$, with an objective function vector $\bar{F}_2$, if no component of $\bar{F}_1$ is greater than its corresponding component in $\bar{F}_2$, and at least one component is smaller. Similarly, $\bar{x}_1$ is said to be Pareto-equivalent to $\bar{x}_2$ if some components of $\bar{F}_1$ are greater than $\bar{F}_2$ and some are smaller. Pareto-equivalent points represent a trade-off between the objective functions, and it is impossible to infer that one point is better than another Pareto equivalent point without introducing preferences or relative weighting of the objectives.

Therefore the solution to a multi-objective optimization problem is a set of vectors which are not dominated by any other vector, and which are Pareto- equivalent to each other. This paper describes a situation in which 31 assets are available and an optimal portfolio weighting of these assets are needed. We assume that interdependencies exist among these assets. Evolutionary algorithms are able to reserve a population of solutions and explore several parts of the Pareto front simultaneously.

4. Multi-objective particle swarm optimization

Observing bird flocks and fish schools, Kennedy and Eberhart [19] realized that an optimization problem can be formulated by mimicking this social behavior of a flock of birds flying across an area looking for food. This observation and inspiration by the social behavior exhibited by flocks of birds and schools of fish, resulted the invention of a novel optimization technique called particle swarm optimization (PSO). In this algorithm a population of individual solutions is developed during the optimization process. The parameters of these population members are adjusted with respect to their previous experience. This experience is derived from the particle as an individual and as a member of the entire population.

Particle swarm optimization algorithms optimize an objective function by conducting a population-based stochastic search. When utilizing stochastic search and optimization to handle multi-objective problems, two major issues should be considered in the algorithm design. First, the fitness evaluation strategy should be addressed so that the search can move towards the Pareto-optimal set. Second, the diversity of the population should be preserved to obtain a well-distributed Pareto-optimal front. In particle swarm optimization the population comprises potential solutions, called particles, which are a metaphor of birds in flocks. These particles are randomly initialized and freely fly across the multi-dimensional search space. During flight, each particle updates its velocity and position based on the best experience of its own and the entire population encountered thus far. The updating rule will steer the particle swarm to move toward the more promising region with higher objective value, and eventually all particles will accumulate around the optimum point. The main steps of particle swarm optimization can be summarized as.

Step 1: Initialization: The velocity and position of all particles are randomly set to fall into the pre-specified or allowed range.

Step 2: Velocity updating: At each iteration, the velocities of all particles are updated according to the following policy:

$$\vec{v}_i = w\vec{v}_i + c_1 r_1\left(\vec{p}_{i,best} - \vec{p}_i\right) + c_2 r_2\left(\vec{g}_{best} - \vec{p}_i\right)$$

where p_i and v_i are the position and velocity of particle i, respectively; $\vec{p}_{i,best}$ and $\vec{g}_{best}$ are the positions with the best objective value found so far by particle i and the entire population, respectively; w is the parameter controlling the dynamics of flying; r_1 and r_2 are random variables in the range [0,1]; c_1 and c_2 are weighting factors.

Step 3: Position updating: Between successive iterations, the positions of all particles are updated according to the following rule: $\vec{p}_i \leftarrow \vec{p}_i + \vec{v}_i$

Step 4: Memory updating: Update pbest and gbest when the corresponding conditions are met:

$$\vec{p}_{i,best} \leftarrow \vec{p}_i \text{ if } f\left(\vec{p}_{i,best}\right) < f\left(\vec{p}_i\right)$$

Where $f\left(\vec{x}\right)$ is the objective function to be minimized.

Step 5: Termination criteria: The algorithm repeats step 2 to step 4 until certain stopping rules are satisfied. Once terminated, the algorithm outputs the $\vec{g}_{best}$ and $f\left(\vec{g}_{best}\right)$ as its solution.

Different single and multi objective PSO algorithms have been proposed by many researcher. Two different techniques have been employed in the literature for gbest selection: (1) roulette wheel selection and (2) quantitative standards [4][5]. Hu and Eberhart[6] proposed a local lbest and a single pbest for each swarm member. Vector evaluated particle swarm optimizer (VEPSO) designates a swarm to each objective in the case of bi-objective problem and the velocities of each swarm are updated by the social leader of the other swarm[7]. Coello and Lechunga [8] applied a grid based selection scheme. Fieldsend and Singh proposed a data structure i.e. a dominated tree for local gbest selection for each individual[9]. Mostaghim and Teich introduced the sigma method to direct particles towards the front where sigma values were calculated for archive and the swarm in each iteration[10]. The gbest for every individual was chosen using the minimal sigma distance to the particle. Rey and Leiw [12] integrated the Pareto ranking scheme and PSO to handle multi objective problems in a way that a leader with a higher crowding radius value is more likely to be selected as a gbest. The improved PSO algorithm proposed by Jiang, Hu, Huang, and Wu used several sub-swarms to enhance both exploration and exploitation by sharing information gained by them. Liao, Tseng, and Luarn proposed a discrete version of PSO for flow shop scheduling problems[13].Niu, Zhu, He, and Wu proposed a cooperative PSO where the population comprises of a master swarm and several slave swarms[14]. In their proposed algorithm each slave swarm applied a single PSO to maintain the diversity of particles while the master swarm improved through its own experience and the experience of the slave swarms.

PSO differs from that of other population-based algorithms as it does not apply the filtering operations such as crossover or/and mutation, and the information is only socially shared by a gbest or lbest. The key point in MOPSO is deciding on which gbest or pbest to option for in order to direct the flight of a swarm member. The selection criteria for pbest is similar to that of PSO with the only difference being that the Pareto dominance determines the leader.

4.1 Multiobjective particle swarm optimization (MOPSO) algorithm

As the analogy of the proposed algorithm with the classic multi-objective evolutionary algorithms (MOEAs), a secondary population, the so-called Archive set, is maintained, which contains a representation of the non-dominated front among all solutions considered so far. The Archive set is used as external storage and updated at each generation.

Step 1: $P_1, A_1 =$ Initialization

Step 2: FOR $t = 1$ to N

```
A. $P_{t+1} = Generate(P_t, A_t)$
    for $j = 1$ TO POPULATIONSIZE
        $g_{j,t}$ = findgbest $(A_t, P_{j,t})$
        $P_{j,t+1}$ = Update Particle $(P_{j,t}, g_{j,t})$
        Evaluate $(P_{j,t+1})$
        $p_{j,t}$ = UpdatepBest $(P_{j,t+1})$
NEXT
B. $A_{t+1}$ = UpdateArchive $(P_{t+1}, A_t)$
C. $P_{t+1}$ = Mutation $(P_{t+1})$
NEXT
Step 3: OutputArchive$(A_{t+1})$
```

where, t denotes the generation index, P_t is the population, A_t is the Archive set at t-th generation, $g_{j,t}$ is the gbest of j-th particle, $p_{j,t}$ is the pbest of j-th particle, and $P_{j,t}$ is the j-th particle of P_t at t-th generation.

The jobs of different functions are as follow:

Initialization: It generates the initial population and copies all non-dominated solutions to the Archive set. The function Generate, generates the next generation population.

FindgBest : It selects gBest from A_t for $P_{j,t}$ adopting the Pareto-optimal solution search algorithm.

UpdateParticle: It updates the speed and position of $P_{j,t}$ using $g_{j,t}$ and $p_{j,t}$

Evaluate: This function evaluates the particles of population.

UpdateArchive: It inserts the non-dominated solutions of P_{t+1} to A_t and removes the superfluous particles from A_t.

OutputArchive : outputs the particles of the Archive set. The steps of the MOPSO algorithm are iteratively repeated until the maximum number of generations is reached.

5. Multiobjective Evolutionary Algorithms

Multi-objective evolutionary algorithms are a popular approach in dealing with problems which consider several objectives to optimize. In this paper work we compare the performance of three recently developed multiobjective evolutionary algorithms such as: Pareto Envelope-based Selection Algorithm, micro genetic algorithm and multiobjective particle swarm optimization for optimal weighting of assets in portfolio optimization problem.

In PESA mating selection procedure is based on a crowding measure. The crowding distance measurement is done over the archive members. Crowding strategy works by forming hyper-grid and it divides phenotype space into hyper-boxes. 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 archive which present in the same hyper box. Environmental selection criteria based on this crowding measure is used for each individuals from archive.

A micro genetic algorithm is a GA with a small population and a reinitialization process. First, a random population is generated and it feeds the population memory. The population memory is divided in two parts (i) a replaceable and (ii) a nonreplaceable portion. The nonreplaceable portion of the population memory never changes during the entire run and it provides the required diversity for the algorithm. But the replaceable portion of population memory experiences changes after each cycle of the microGA. The microGA uses three forms of elitism: (i) it retains nondominated solutions found within the internal cycle of the microGA (ii) it uses a replaceable memory whose contents is partially refreshed at certain intervals and (iii) it replaces the population by the nominal solutions produced i.e. the best solutions found after a full internal cycle of the microGA. This approach was proposed by Coello Coello[2].

5.1 PESA Algorithm

PESA has two parameters concerning population size i.e P_I (the size of the internal population IP) and PE (the maximum size of the archive or external population). It has one parameter concerning the hyper-grid crowding strategy.

The main steps in this algorithm are (i) Generate and evaluate each of an initial internal population (IP) of P_I chromosomes and initialize the external population (EP) to the empty set.(ii) Incorporate the non-dominated members of IP into EP.(iii) If a termination criterion has reached then stop, returning the set of chromosomes in EP as the result. Otherwise delete the current contents of IP and repeat the following until P_I new candidate solutions have been generated. With probability p_c , select two parameters from EP. Produce a single child via crossover and mutate the child. With probability $(1-p_c)$ select one parent and mutate it to produce a child. (iv) Repetition of the same process.

PESA has two parameters concerning population size i.e. p_I (the size of internal population, IP) and PE (the maximum size of external population EP) . It has one parameter concerning the hyper-grid crowding strategy.

```
1. Generate and evaluate each of an initial internal population (IP) of PI chromosomes.
2. Initialize the external population (EP) as empty set.
3. For $t = 1$ to Number of Generations
3.1. Incorporate the non-dominated members of IP into EP.
3.2. Delete the current content of IP.
3.3. Until obtain new solution of $p_I$.
3.3.1. Select two parents from EP with probability $p_c$
3.3.2. Recombination this two parents for obtaining one offspring
3.3.3. Mutate the offspring
3.3.4. Select one parent from IP with probability $(1-p_c)$
3.3.5. Mutate the parent to produce one offspring
3.3.6. Add the two obtained offspring into IP
4. Return to 3
```

5.2 Micro-GA Algorithm

```
1. Generate starting population $P$ of size $N$
2. Store its contents in the population memory $M$
3. Divide the population memory $M$ in two parts, a replaceable and a nonreplaceable part.
4. For $t = 1$ to Number of Generations
4.1. Get the initial population of micro-GA ( $P_i$ ) from $M$
4.2. Apply the binary tournament selection based on nondominance.
4.3. Apply tow point crossover and uniform mutation to the selected individual
4.4. Apply elitism (retain only one nondominated vector) and generate next generation
4.5. Until nominal convergence is reached copy two nondominated vector from $P_i$ to the external memory $E$ .
```

4.6. When E is full use adaptive grid.
4.7. Copy two nondominated vectors from P_i to M
5. Return to step 4

6. Comparison of Results

For performance comparison and quantitative assessment of a multiobjective optimization algorithm normally three issues are taken into consideration.
(i) Minimize the distance of the Pareto front produced by our algorithm with respect to the global Pareto front, assuming the location of global Pareto front. To address the issue Van Veldhuizen and Lamont [17] proposed generation distance (GD).

$$GD = \frac{\sqrt{\sum_{i=1}^{n} d_i^{\,2}}}{n} \tag{6}$$

Where n the number of vectors in the set of nondominated solutions is found so far and d_i is the Euclidean distance between each of these and the nearest member of the Pareto optimal set. If $GD = 0$ indicates all the elements generated are in the Pareto optimal set. Therefore, any other value will indicate how far the elements are from the global Pareto front.
(ii) Maximize the spread of solutions found so that the distribution of vectors will be smooth and uniform. Schott [16] introduced a metric i.e. Spacing (SP) metric for this propose which measures the range variance of neighboring vectors in the nondominated vectors. This metric is defined as:

$$S \overset{\Delta}{=} \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}\left(\bar{d} - d_i\right)^2} \tag{7}$$

Where $d_i = \min_j \left(\left| f_1^{\,i}\left(\vec{x}\right) - f_1^{\,j}\left(\vec{x}\right) \right| + \left| f_2^{\,i}\left(\vec{x}\right) - f_2^{\,j}\left(\vec{x}\right) \right| \right)$ (8)

$i, j = 1,2,...,n$

$\bar{d}$ = mean of all d_i and n is the number of nondominated vectors found so far. A value of zero for this metric indicates all members of the Pareto front currently available are equidistantly spaced.
(iii) Maximize the number of elements of the Pareto optimal set found. Error ratio (ER) metric was proposed by Van Veldhuizen [18] which addresses this issue. It indicate the percentage of solutions from the nondominated vectors that are not members of the true Pareto optimal set

$$ER = \frac{\sum_{i=1}^{n} e_i}{n} \tag{9}$$

Where n is the number of vectors in the current set of nondominated vectors available. if vector i is a member of the Pareto optimal set then $e_i = 0$ and $e_i = 1$ if vector i not a member of Pareto optimal set. $ER = 0$ indicates all the vectors generated by our algorithm belong to the Pareto optimal set of the problem.

For comparison of two non-dominated solution sets obtain by proposed multiobjective algorithm for efficient weighting of available assets, the following measures are computed:

1. S metric

The S metric proposed in [4] indicates the extent of objective space dominated by a given nondominated set A. If the S metric of a non dominated front f_1 is less than another front f_2 then f_1 is better than f_2. It has been proposed by Zitzler [4].

2. Δ metric

This metric called as spacing metric (Δ) measures how evenly the points in the approximation set are distributed in the objective space. This formulation introduced by K. Deb [1] is given by

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{N-1}\left|d_i - \bar{d}\right|}{d_f + d_l + (N-1)\bar{d}} \tag{10}$$

Where d_i be the Euclidean distance between consecutive solutions in the obtained nondominated set of solutions. $\bar{d}$ is the average of these distances. d_f and d_l are the Euclidean distance between the extreme solutions and the boundary solutions of the obtained non dominated set and N is the number of solutions from nondominated set. The low value for Δ indicate a better diversity and hence better is the algorithm.

3. C metric

Two sets of non dominated solutions are compared using C metric. The definition of C metric given in [4] for convergence of two sets A and B is given by:

$$C(A,B) = \frac{\left|\{b \in B \mid \exists a \in A : a \succ b\}\right|}{|B|} \tag{11}$$

7. Simulation Studies

In this section we describe the test problem used to compare the performance of PESA, MicroGA and MOPSO for optimal weighting of the available assets. In all case the objective number is 2. We have taken parameters of these algorithms such a way that it will be comparable. We run experiments on data from OR library that maintained by Prof. Beasley as a public benchmark data set and is derived from Heng Seng data set with 31 assets. The data can be found at http://people.brunel.ac.uk/~mastjjb/jeb/orlib/portinfo.html.

The PESA has internal population size of 100, external population size of 100 and number of gene is equal to number of assets. The number of generations is taken as 100, crossover is uniform having rate as 0.8. It has mutation rate of $1/L$, where L refers to the length of the chromosomic string that encodes the decision variables. The grid size i.e. the number of division per dimension is 10.

The microGA used an external memory of 100 individuals, a number of iterations to achieve nominal convergence , a population memory of 50 individuals. Percentage of nonreplaceable memory equals 0.05, a population size of four individuals and 25 subdivisions of the adaptive grid. It used a crossover rate of 0.9 and mutation rate was set to $1/L$ ($L =$ length of the chromosomic string). MOPSO used a population of 100 particles and a repository size of 100 particles. The mutation rate set to 0.5 and 30 divisions for the adaptive grid.

Table 1 The S and Δ metric

Algorithm	PESA	MicroGA	MOPSO
Metric S	0.0003095745	0.0000067874	0.0000003461
Metric Δ	0.8654128591	0.8227976192	0.561273589

Table I shows the S and Δ metrics obtained using all the three algorithms. It may be observed from the Table I that MOPSO performs better as its S and Δ metric values are less than those obtained by other two algorithms.

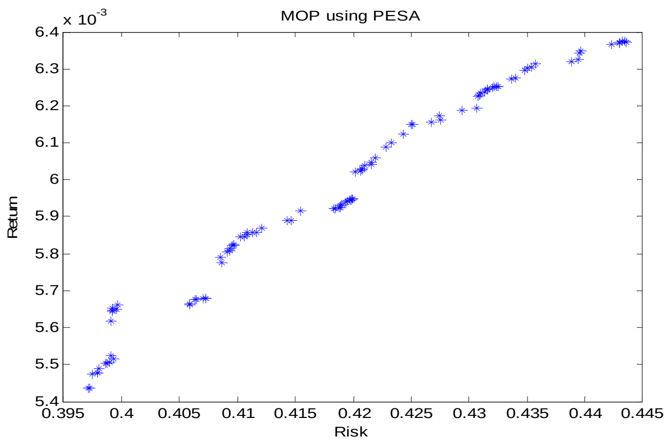
Table 2 The result obtained for C metric

	PESA	MicroGA	MOPSO
PESA	—	0.0000	0.0000
MicroGA	0.5579	—	0.0366
MOPSO	0.94627	.88534	—

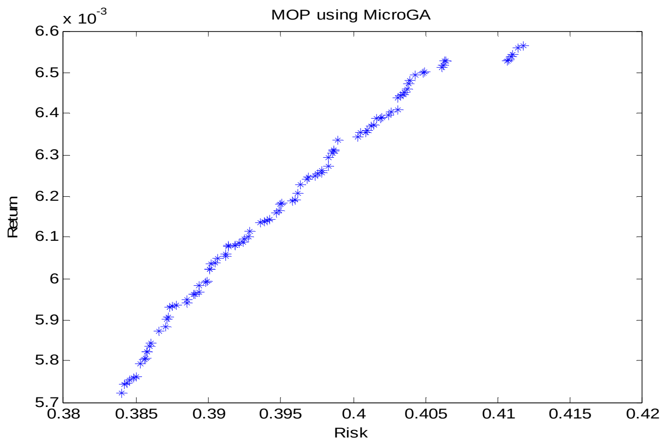
Table II demonstrates the results of C metric. A magnitude of 0.94627 on the third line, first column means almost all solutions from final populations obtained by MOPSO dominate the solutions obtained by PESA. The values 0 on first row means that no solution from the nondominated population obtained by MicroGA and by MOPSO is dominated by solutions from final populations obtained by PESA.

8. Convergence Characteristics

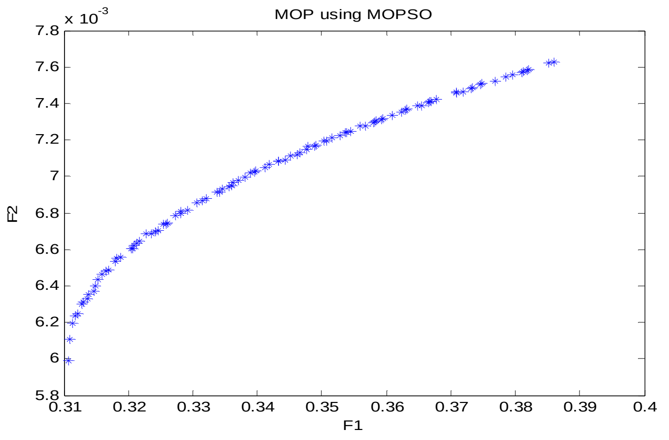
The Pareto front (between risk and return) attained by three algorithms are depicts in Figs (a)-(c):



(a) PESA



(b) MicroGA



(c) MOPSO

Fig 1. Plots of Pareto fronts achieved by three methods

9. Conclusions and Further work

The paper makes a comparative study of three multi-objective approaches PESA, MicroGA and MOPSO for solving portfolio optimization problem. Experimental results reveal that the MOPSO algorithm outperforms other two MOEA algorithms in the considered optimal weighting of assets as a portfolio design problem. Future work include introduction and incorporation of different operators for local search which allow better exploration and exploitation of the search space when applied to portfolio design problem.

References

[1] K. Deb, S. Agrawal, A. Pratab, and T. Meyarivan. A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGAII. Proceedings of the Parallel Problem Solving from Nature VI Conference, pages 849-858,Paris, France, 2000. Springer. Lecture Notes in Computer Science No.1917.

[2] Carlos C.H Coello Coello. A Comprehensive Survey of Evolutionary –based multiobjective optimization Techniques. Knowledge and Information System. An International Jornal,1(3): 269-308,Aug 1999.

[3] H. M. Markowitz. Portfolio Selection: efficient diversification of investments. John Wiley & Sons, 1959.

[4]E. Zitzler and L. Thiele. Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. IEEE Transactions on Evolutionary Computation, 3(4):257-271, 1999.

[5] Hu, X., Shi, Y., & Eberhart, R. C. (2002). Solving constrained nonlinear optimization problems with particle swarm optimization. In Proceedings of the sixth world multiconference on systemics, cybernetics and informatics 2002, Orlando, USA.

[6] Hu, X., & Eberhart, R. C., (2002). Multiobjective optimization using dynamic neighborhood particle swarm optimization. In Proceedings of the 2002 congress on evolutionary computation, part of the 2002 IEEE world congress on computational intelligence, Hawaii, May 12–17. IEEE Press.
[7]Parsopoulos, K. E., & Vrahatis, M. N. (2002). Particle swarm optimization method in5 multiobjective problems. In Proceedings of the 2002 ACM symposium on applied computing (pp. 603–607).

[8]Coello, C. A. C., & Lechunga, M. S. (2002). 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 (pp. 1051–1056), Hawaii, May 12–17, 2002. IEEE Press.

[9] Fieldsend, J. E., & Singh, S. (2002). 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) (pp. 37–44), Birmingham, UK, September 2–4.

[10] Fieldsend, J. E. (2004). Multi-objective particle swarm optimization methods. 1st March, Technical Report, Department of Computer Science, University of Exeter.

[11] Mostaghim, S., & Teich, J. (2003). Strategies for finding good local guides in multiobjective particle swarm optimization (mopso). In IEEE 2003 swarm intelligence symposium.

[12] Rey, T., & Leiw, K. M. (2002). A swarm metaphor for multiobjective design optimization. Engineering Optimization, 34(2), 141–153.

[13] Liao, C., Tseng, C., & Luarn, P. (2007). A discrete version of particle swarm optimization for flow shop scheduling problems. Computers and Operations Research, 34, 3099–3111.

[14] Niu, B., Zhu, Y., He, X., & Wu, H. (2007). MCPSO: A multi-swarm cooperative particle swarm optimizer. Applied Mathematics and Computation, 185, 1050–1062.

[15] Laura Diosan, A multi-objective evolutionary approach to the portfolio optimization problem, IEEE conference,CIMCA-IAWTIC'05.

[16] J. R. Schott, "Fault tolerant design using single and multicriteria genetic algorithm optimization," M.S. thesis, Dept. Aeronautics and Astronautics,Massachusetts Inst. Technol., Cambridge, MA, May 1995.

[17] D. A. Van Veldhuizen and G. B. Lamont, "Multiobjective evolutionary algorithm research: A history and analysis," Dept. Elec. Comput. Eng.,Graduate School of Eng., Air Force Inst. Technol., Wright PattersonAFB, OH, Tech. Rep. TR-98-03, 1998.

[18] D. A. Van Veldhuizen, "Multiobjective evolutionary algorithms: Classifications,analyzes, and new innovations," Ph.D. dissertation, Dept.Elec. Comput. Eng., Graduate School of Eng., Air Force Inst. Technol.Wright-Patterson AFB, OH, May 1999.

[19] Hu, X., Shi, Y., & Eberhart, R. C. (2004). Recent advances in particle swarm. In IEEE congress on evolutionary computation 2004, Portland, Oregon, USA.