DYNAMIC POPULATION STRATEGY ASSISTED PARTICLE SWARM OPTIMIZATION IN MULTIOBJECTIVE EVOLUTIONARY ALGORITHM DESIGN

Haiming Lu
School of Electrical and Computer Engineering
Oklahoma State University, Stillwater, OK, 74078, USA

Abstract- In this research report, the author proposes two new evolutionary approaches to Multiobjective Optimization Problems (MOPs)— Dynamic Particle Swarm Optimization (DPSMO) and Dynamic Particle Swarm Evolutionary Algorithm (DPSEA). In DPSMO, instead of using genetic operators (e.g., crossover and mutation), the information sharing technique in Particle Swarm Optimization is applied to inform the entire population more accurate moving direction and speed than EA. Meanwhile, based on the dynamic population strategies, cell-based rank and density estimation and objective space compression strategy used in Dynamic Mutiobjective Evolutionary Algorithm (DMOEA), the DPSMO can also evolve to an approximately optimal population size while the population is approaching the true Pareto front. Moreover, to overcome DPSMO's difficulty in producing high-quality Pareto front, DPSEA is designed by combining both EA and PSO's information sharing techniques. By examining the selected performance measures on two test functions, DPSEA is found to be competitive with, or even superior to DMOEA and DPSMO in terms of keeping the diversity of the individuals along the trade-off surface, tending to extend the Pareto front to new areas and finding a well-approximated Pareto optimal front. Moreover, from the experimental results, DPSMO and DPSEA also show more promising performance in improving algorithm efficiency comparing to DMOEA.

I. INTRODUCTION

During the past decade, several Multiobjective Evolutionary Algorithms (MOEAs) have been proposed and applied in Multiobjective Optimization Problems (MOPs) [1]. These algorithms share the unique purpose—searching for a uniformly distributed, *near-optimal* and *well-extended* Pareto front for a given MOP. Therefore, from VEGA [2] to NSGA II [3], many advanced MOEAs had been developed and they all show their unique advantages in solving multiobjective optimization problems. However, their drawback is also obvious—need relatively longer time in producing high quality Pareto front comparing to the traditional optimization methods (i.e., linear weighting method). This low-efficiency problem is caused by EA's population-based information sharing technique and random variation characteristics, which cannot be overcome by evolutionary algorithm itself. Although in [4], a Dynamic Multiobjective Evolutionary Algorithm (DMOEA) proposed a promising way to improve the computation efficiency of MOEA by applying dynamic population strategies, it is still restricted by EA's intrinsic properties and will not escape from EA's major problems. Therefore, in order to improve the efficiency of an MOEA, we need to search for a clever technique to assist MOEA to achieve a high-quality Pareto

front with a faster convergence speed. In this report, Particle Swarm Optimization (PSO) is considered to be such a candidate technique.[1]

Particle Swarm Optimization (PSO) was first proposed by Kennedy and Eberhart [5] in 1995, which was inspired by the choreography of a bird flock. According to PSO, the behavior of each individual is affected by either the best local or the best global individual to help it fly through a hyperspace. Since an individual can only obtain useful information from the local or global optimal ones, it will not be disturbed by any other individuals, which may be helpful to force it converges to the best solution quickly. However, due to PSO's single-point-centered characteristic, it is difficult to locate the non-dominated points on the Pareto front since there will be more than one criteria exist to direct the velocity and position of an individual. Nevertheless, with certain modifications (i.e., Pareto ranking [6]+ niche sharing [7], neighborhood method [8]), PSO can be suitable to solve MOPs. By now, there are very few papers [9-12] found to extend PSO in solving MOPs, this research area is still in its beginning stage.

In this reprot, a Particle Swarm Optimization method is combined with a dynamic population strategy [4] to devise a Dynamic Particle Swarm Multiobjective Optimization (DPSMO) algorithm. In DPSMO, cell-based rank and density values of each individual are calculated and the positions of the global and local optimal points in terms of rank and density values are obtained for each individual. In particular, the information sharing mechanism of evolutionary algorithm (i.e., crossover and mutation) is replaced by PSO's one-way information sharing method to drive an individual from current position to the next more optimal one in decision space. From the experimental result, DPSMO is found to be more efficient than DMOEA whereas the decreasing of the quality of resulting Pareto front is the deficiency of DPSMO comparing to DMOEA. Therefore, another evolutionary algorithm— Dynamic Particle Swarm Evolutionary Algorithm (DPSEA) is also designed in order to improve both algorithm efficiency and the quality of the resulting Pareto front. In DPSEA, the information sharing mechanisms of EA and PSO are applied simultaneously and the experimental results implies that DPSMO and DPSEA are two promising approaches in solving complicated MOP to provide a relatively fast convergence speed and high quality Pareto front.

The remainder of this reprot is organized as follows. Section II reviews the basic procedure of Particle Swarm Optimization (PSO). Section III describes the key techniques used in Dynamic Multiobjective Evolutionary Algorithm (DMOEA), which include cell-based rank and density calculation scheme, population growing strategy and population declining strategy, and objective space compression strategy. In Section IV, a Dynamic Particle Swarm Optimization (DPSMO) algorithm is devised by combining the skeleton of DMOEA with PSO's information technique. Section V shows the experimental study of comparing the performances of DMOEA and DPSMO in terms of rank, density distance and C-measure on a selected benchmark problem. Based on the deficiencies and advantages of both algorithms, another algorithm—Dynamic Particle Swarm Evolutionary Algorithm (DPSEA), which integrates both EA and PSO's information sharing mechanisms, are proposed in Section VI. The analysis of

[1]This report is written based on the original project "Multiobjective Evolutionary Algorithms" sponsored by IEEE Neural Network Society (NNS) 2002 Student Summer Research Grant.

the experimental results of DMOEA, DPSMO and DPSEA on two test functions explain different characteristics of *efficiency* and *efficacy* of three algorithms. Finally, Section VI provides some concluding remarks along with pertinent observations.

II. Particle Swarm Optimization

Particle Swarm Optimization (PSO) was first proposed by Kennedy and Eberhart [5] in 1995. This technique was inspired by the choreography of a bird flock and can be seen as a distributed behavior algorithm that performs multidimensional search [13]. According to PSO, the behavior of each individual is affected by either the best local or the best global individual to help it fly through a hyperspace. Moreover, an individual can learn from its past experiences to adjust its flying speed and direction. Therefore, by observing the behavior of the flock and memorizing their flying histories, all the individuals in the swarm can quickly converge to *near-optimal* geographical positions with well-preserved population density distribution.

Normally, PSO is considered as an evolutionary computation approach in that it possesses many common characteristics used by evolutionary algorithms such as: 1) It is initialized with a population of random solutions. 2) It searches for the optimum by updating generations. 3) The adjustments of individuals are analogous to real value crossover operation in evolutionary algorithms. 4) Fitness evaluation is evaluated by objective functions.

However, unlike EAs, the updates of the individuals of PSO are not accomplished by random crossover or mutation of genes. In PSO, an equation is used to calculate the new velocity of each individual i at the jth dimension based on its current location $x(i,j)$, previous velocity ($V_p(i,j)$), previous location ($p_{best}(i,j)$), where the best fitness this individual has been achieved, and the population global location ($g_{best}(j)$), where the best fitness value the entire population has achieved. Therefore, the velocity updating equation is

$$V_p(i,j) = \omega V_p(i,j) + R_1(p_{best}(i,j) - x(i,j)) + R_2(g_{best}(j) - x(i,j)) \quad (1)$$

where ω is an inertia weight value [88] and R_1 and R_2 are two random numbers between 0 and 1. After the velocity is updated, the new location of ith individual at the jth dimension can be calculated as

$$x(i,j) = x(i,j) + V_p(i,j) \quad (2)$$

Comparing with evolutionary algorithms, PSO's information sharing mechanism is significantly different. In EAs, individuals share their information with each other by crossover and the whole population moves like one group towards an optimal point. In PSO, only $g_{best}(j)$ provides the information to other individuals to adjust their

speeds. It is a one-way information sharing mechanism [9]. The entire population follows the movement of the best individual and converges to a *near-optimal* solution quickly.

III. Dynamic Multiobjective Evolutionary Algorithm (DMOEA)

In order to improve MOEA's *efficiency* and *efficacy* and approximate a optimal population size for a given MOP, a Dynamic Multiobjective Evolutionary Algorithm (DMOEA) was proposed by Yen and Lu [4]. In DMOEA, instead of determining population size heuristically, a dynamic population growing and a population declining strategy are designed to help the algorithm allocates optimal number of nondomiated solutions for final resulting Pareto set. Moreover, there are several other novel techniques are devised in DMOEA to refine the quality of resulting Pareto front and reduce computational complexity. These key techniques are described as followings:

3.1 Cell-based Rank and Density Calculation Scheme

In DMOEA, the original *m*-dimensional objective space is divided into $K_1 \times K_2 \times \mathrm{K} \times K_M$ cells (i.e. grids) and then the center position of each cell will be obtained. First, two matrices are set up to store the rank and density values of each cell, which initially having all 1 and 0, respectively. Second, each individual of the initial population will search for its nearest cell center and identify this cell as its "home address" and consider the other individuals who share the same "home address" as its "family members". For each of these "homes", the number of "family members" who dwell in it will be counted and saved as the density value of the "home". In addition, the rank values of the cells that are dominated by any of these "homes" will be increased by the density values of those "homes". Third, when an offspring is generated and accepted, its "home address" can be easily located by following the second step, and the density value of its "home" will increase by one, and the rank values of the cells dominated by its "home" are increased by one. Meanwhile, if an old individual is removed, its "home" will be notified, and the density value of its "home" will be decreased by one and the rank values of the cells dominated by its "home" are decreased by one, correspondingly. Therefore, if the estimated objective space is approximately chosen that a new generated or a removed individual does not change the boundaries of the range of the current objective space, the size of each cell will not change, which means an individual's "home address" will never change if this individual is not removed. By this means, no matter how large the population size is, the computation effort for both matrices updating and fitness evaluation will not be affected. The proposed cell-based rank and density estimation design has provided an efficient way in applying a dynamic population size in evolutionary process.

3.2 Population Growing Strategy

Because exploring the cells with minimum rank values and maintaining these cells densities to a desired value are two converted objectives of DMOEA, crossover and mutation operations need to be devised to fit both of the purposes. The fitness assignment scheme of DMOEA is borrowed from RDGA [8]— the main population is equally divided into two subpopulations that are responsible for minimizing the rank value and maintaining the density value, respectively. If an offspring resulted from crossover or mutation is located in a cell with a better

fitness (a lower rank value or a lower density value) than its selected parents, it will be kept to the next generation; otherwise, it will not survive. As a result, this strategy will guarantee that a newborn individual will have a better fitness value than at least one of its parents, which helps DMOEA to cover all the unexplored cells in the objective space.

However, as DMOEA encourages an offspring to spawn in a sparse area thus those unexplored cells will have higher chances to be visited, it is expected that some offspring will tend to move toward an opposite direction to the true Pareto front when the cells close to the Pareto front are crowded. Obviously, these movements are harmful to the population to converge to the Pareto front. To prevent "harmful" offspring from surviving and affecting the evolutionary direction and speed, a *forbidden region* concept is proposed in the offspring-generating scheme for the density subpopulation, thereby preventing the "backward drifting" effect. As shown in Figure 1, the offspring located in the *forbidden region* will not survive in the next generation, and thus the selected parent will not be replaced.

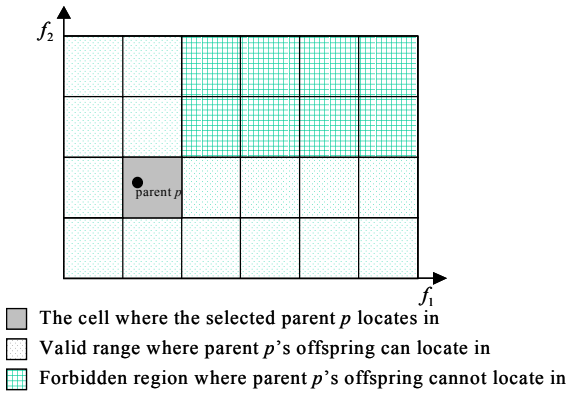


Figure 1 Illustration of the valid range and the forbidden region

3.3 Population Declining Strategy

In DMOEA, a population declining strategy is also designed to prevent the population size from growing excessively. According to this strategy, whether an individual will be removed or not depends on its cell rank and density values. Moreover, to ensure that each generated individual has enough lifespan to contribute its valuable schemas, an *age* value is also designed in DMOEA. For an individual in the initial population, its *age* value is assigned to be one, and its *age* will increase by one if the individual survives at the next generation. Based on this setting, an individual will not be eliminated if its *age* is less than a predetermined age threshold.

To ensure that an eliminated individual has a low fitness value, DMOEA periodically removes three types of individuals with different likelihood—the likelihood of removing the individuals with highest rank values, the likelihood of removing the dominated individuals in the most crowded cells, and the likelihood of removing non-dominated individuals from the most crowded cells after all the resulting cell rank values are 1. For any individual,

if its calculated likelihood is smaller than a number that is randomly generated between 0 and 1, the individual will be removed from the population.

3.4 Objective Space Compression Strategy

Although the cell-based rank and density calculation scheme discussed in Subsection 3.1 can significantly improve the efficiency of DMOEA during its evolutionary process, it cannot guarantee the accuracy of the resulting Pareto front since an individual's rank value is represented by the rank value of its "home address", not by its own dominance status. Because the size of the true Pareto front is generally unknown, the boundaries of the objective are usually selected to be very large, which may be far away from the true Pareto front, to ensure that the entire true Pareto front is covered by the estimated objective space. To address this problem, an objective space compression strategy is designed in DMOEA to adjust the size of the objective space and to make it suitable to search for the true Pareto front with a high precision. Before the objective space is compressed, three criteria are examined. They are: the maximum cell rank value of all the individuals is 1, the distance between the boundary of objective space and the boundary of current population in any dimension is greater than a pre-determined value, and the minimum age value of all the individuals is greater than the predefined age threshold. If all of three criteria are met, the objective space will be compressed. Therefore, by continuously compressing the objective space, the resulting non-dominated set can be tuned, and a more extended and homogenously distributed Pareto front can be obtained.

IV. DYNAMIC PARTICLE SWARM MULTIOBJECTIVE OPTIMIZATION

In this reprot, to tackle multiobjective optimization problems, PSO is devised with dynamic population strategies proposed in DMOEA [3]. In another word, DMOEA's crossover and mutation scheme is replaced by PSO's information sharing method in order to improve convergence speed. To prevent the degradation of the effectiveness and efficiency of the algorithm, the following strategies are applied in the new algorithm:

1. The genotype of each individual will be a real number instead of binary genes form.
2. For each individual, its genotype includes two types of velocity parameters—rank velocity and density velocity. On each dimension of the decision vector, an individual will be assigned with a rank and a density velocity.
3. Cell rank value of each individual is still calculated, all the individuals with rank value equal to 1 are the global best (rank) individuals. However, for any individual **A**, only those best (rank) individuals that dominate it will be considered as the candidates of **A**'s g_{best_rank}. If more than one candidates of g_{best_rank} exists, the one with lowest density value will be selected as the g_{best_rank} of individual **A**.
4. For any individual **A**, its local best (rank) individual p_{best_rank} is randomly selected from the individuals that are located in the same cell and dominate **A**. If there is no such kind of individual exists, p_{best_rank} will be individual **A** itself.

5. Cell density value of each individual is still calculated in objective space. For any individual **A**, its global best (density) individual g_{best_den} is the individual that has the lowest cell density value (except "forbidden region").
6. For an individual **A**, its local best (density) individual p_{best_den} is randomly selected from the individuals that are located in the same cell or neighbor cells (except "forbidden region") and has the lowest cell density value.
7. The entire population is equally and randomly divided into two subpopulations that are responsible for minimizing rank value and maintaining density value, respectively. For each subpopulation, one individual will be randomly selected from each cell (if the cell contains any individual that belongs to the concerned subpopulation). All the selected individuals are cloned and the location and velocity of the cloned version of each of these individuals will be updated in decision space based on Equation (1) and (2).
8. The original population and the updated cloned population will be shuffled and survive to the next generation.
9. Population declining strategy performs same as described in DMOEA.
10. Objective compression strategy performs same as described in DMOEA.

From the procedures of the proposed Dynamic Particle Swarm Multiobjective Optimization (DPSMO) algorithm, we can see that

1. At each generation, since there are more than one individuals have lowest rank or density values, the geography restrictions described in step 3 or 5 are necessary in order to assign an individual a specific global best target to follow. Otherwise, any nondominated individuals may affect an individual's new velocities, moving direction and location, which will disturb an individual's moving trajectory and slow down the convergence speed significantly.
2. To obtain a set of optimal solutions with uniform distribution, the population density value needs to be preserved as long as the nondominated solution set is refined. Therefore, in DPSMO, each individual has two types of velocities, rank velocity and density velocity, which will guarantee both Pareto optimality and uniform distribution of the final results will be achieved.
3. For an individual, Equations (1) and (2) update its velocities and locations on each dimension in the decision space. Indeed, this action implies a crossover operation among an individual, its local best and its global best. The newly updated individual can be considered as an offspring. For this reason, "population growing strategy" in DMOEA is not applied in DPSMO since an individual is supposed to know "where to go" before it moves in particle swarm. Instead of applying "population growing strategy", using simple offspring updating method based on Equation (9,1) and (9.2) will save significant running time spending in DMOEA on evaluating an offspring's fitness value and comparing with its parents.

4. As there may be more than one particles affect an individual's moving speed and direction, and most importantly, there are two types of velocities associated with each individual, the "cloning" method in step 7 implements an elitism scheme to keep the newly explored better-fitted individuals. This method is crucial for DPSMO because it guarantees the population converges to the correct direction.
5. Ill-fitted individuals will be removed based on "population declining strategy", thus the population size can be controlled and the population quality will be increased.

V. EXPERIMENTAL STUDY ON DPSMO

In order to validate the proposed DMOEA and compare its performance with DMOEA, a test function proposed in [] are tested DMOEA and DPSMO and each of the algorithms runs for 50 times to obtain the statistical results. For the test function, both DMOEA and DPSMO run with the initial population size equal to 2 and achieve an approximated desired population size dps. We use three indicators derived from the final generations of 50 runs to benchmark the comparison results via statistical Box plots. They are: average individual rank value, average individual density value and average individual distance. The reason that average individual distance is used as an indicator is because the rank is a relative value, which implies we cannot guarantee that the final population will be a true Pareto set even if all its individuals have rank values equal to 1s as shown in Figure 15. For this reason, we use the "final average individual distance" as the third indicator to show how far the non-dominated points on the resulting final Pareto front PF_{final} are away from the true Pareto front PF_{true}, where PF_{true} is known in *a priori* for the given test functions in this reprot. This indicator was originally introduced by Veldhuizen and Lamont [14], where the final individual distance G is defined as

$$G = \frac{(\sum_{i=1}^{N} d_i^2)^{1/2}}{N}, \tag{3}$$

where N is the number of individuals in PF_{final}, and d_i is the Euclidean distance between each of these individuals and a point on PF_{true} that is the closest to it. A result of $G = 0$ indicates the convergence $PF_{final} = PF_{true}$; any other value indicates PF_{final} deviates from PF_{true}.

Moreover, in order to compare the dominance relationship between two final populations resulted by two different MOEAs, the coverage of two sets (C value) [15] is measured to show how the final population of one algorithm dominate the final population of another algorithm. Function C maps the ordered pair (X_i, X_j) to the interval [0, 1], where X_i and X_j denote the final populations resulted from the algorithms *i* and *j*, respectively. The value $C(X_i, X_j) = 1$ means that all points in X_j are dominated by, or equal to, points in X_i. The opposite,

$C(X_i, X_j) = 0$, represents the situation when none of the points in X_j are covered by the set X_i. Note that both $C(X_i, X_j)$ and $C(X_j, X_i)$ need to be considered independently since they have distinct meanings.

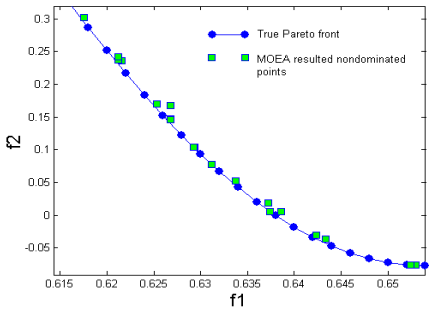


Figure 2 Difference between PF_{true} and PF_{final}

Therefore, four indicators represent qualitative measures that describe the quality of the final result of DMOEA and DPSMO— the average individual rank value shows the dominated relationship between different individuals, the average individual density value illustrates how good the population diversity is preserved, the average individual distance measures distance between PF_{final} and PF_{true}, which provides the quality of the resulting Pareto front, and the C value compares the domination relationship between DMOEA and DPSMO. All the values of the four indicators evaluated at the final generation are illustrated by Box plots to provide the statistical comparison results. Table 1 illustrates the key parameters used by DMOEA and DPSMO in simulation.

Table 1 Parameter setting settings for DMOEA and DPSMO

Common parameters	Chromosome length	Offspring generated per generation	Crossover rate	Number of individuals perform mutation	Number of binary bit flipped	PSO Weight value ω
DMOEA	$15 \times dec\_num$ *	10	0.7	1	1	-
DPSMO	-	-	-	-	-	0.5

* dec_num **represents number of decision variables**

Minimize $f_1(x)$ and $f_2(x)$, where

$$f_1(x) = 1 - e^{-4x_1} \sin^6(6\pi x_1) \tag{4}$$

$$f_2(x) = g(x)\left(1 - \frac{f_1(x)}{g(x)}\right)^2,$$

$$g(x) = 1 + 4\left(\sum_{i=2}^{5} x_i / 4\right)^{0.25},$$

subject to $0 \le x_i \le 1, \quad i = 1, \mathrm{K}, 6.$

Test function ***F1*** is proposed in [16], which is described in Equation (4). It has an MOP with a high-dimensional decision space and local Pareto fronts in objective space as shown in Figure 3. For a fair comparison,

the initial population, the age threshold, the population size per unit volume, ppv and the cell scales K_1 and K_2 are selected as 2, 10, 3, 50 and 50, respectively. Both algorithms run 50 times and the stopping generation is set to be 2,000. Figure 4 shows the true Pareto front, resulting Pareto front by DMOEA and DPSMO. The evolutionary trajectories of the population size and average individual rank, density and distance values over 50 runs by DMOEA and DPSMO are shown in Figures 5 (a) – (d), respectively.

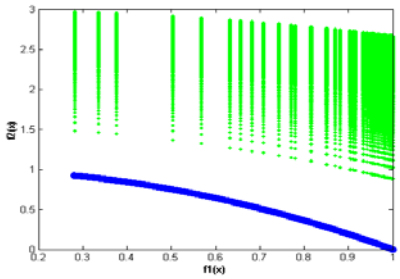


Figure 3 Objective space and Pareto front of *F1*

Form Figure 4, apparently, there are many final solutions resulting from DPSMO are dominated by those from DMOEA. This result can also be verified by Figure 5, which shows the final Pareto front of DMOEA is closer to the true Pareto front than that of DPSMO. This effect can partially explained by the intrinsic characteristics of function ***F1***'s local and global optimality—when the resulting Pareto front is getting closer to the true (global) Pareto front, both algorithms experience difficulty to yield better-fitted offspring. Moreover, for DPSMO, since only the global best individuals and local best individuals can provide the moving information to the entire population, it is more possible for DPSMO to stuck at a middle stage if all the current individuals are Pareto optimal and there is no even better-fitted g_{best_rank} is generated. This problem will hold DPSMO from locating true Pareto front, especially when the given MOP has more than one local Pareto fronts.

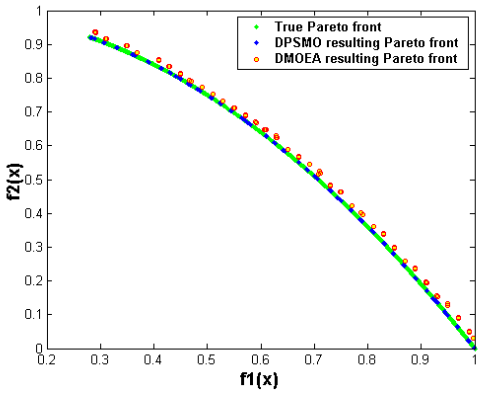
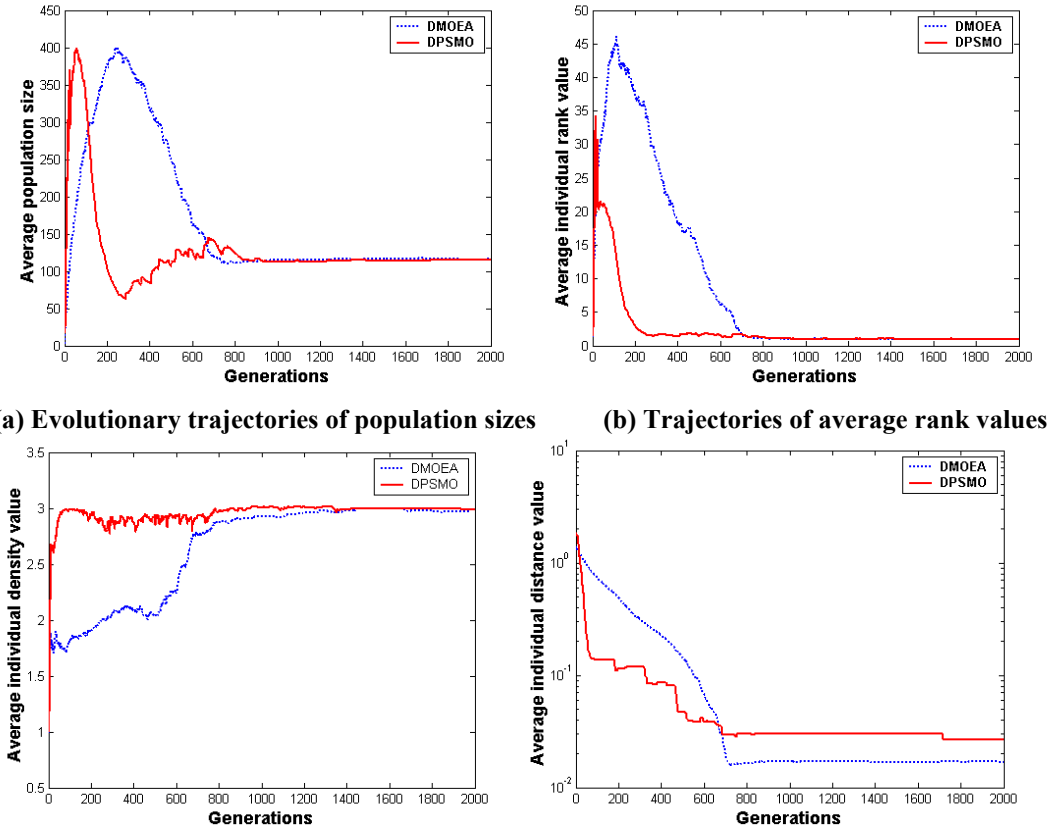


Figure 4 Resulting Pareto fronts by DMOEA and DPSMO on Function *F1*

Comparing to DPSMO, DMOEA does not have this problem because evolutionary algorithm applies a population-based information sharing mechanism. A better-fitted offspring can be generated by a crossover operation between any two individuals, no matter how good these parents are. However, from Figure 5 (a) – (c), we can see the advantage of DPSMO over DMOEA in that DPSMO produces much faster convergence speed. In

DPSMO, each particle knows its moving direction and how fast it should go if there exists another individual with better performance. Therefore, before it is trapped by a local Pareto front, the probability that an individual generates a better-fitted offspring by DPSMO is much higher than that of DMOEA. This characteristic will result in both less evaluation time and less generation numbers, which are the major reasons that DPSMO spends much less generations than DMOEA on Function ***F1*** in terms of converging entire population to a uniformly distributed Pareto front.



(a) Evolutionary trajectories of population sizes **(b) Trajectories of average rank values**

(c) Trajectories of average density value **(d) Trajectories of average distance values**

Figure 5 Evolutionary trajectories for the population size and the values of three indicators resulting by DMOEA and DPSMO on Function *F1*

VI. DYNAMIC PARTICLE SWARM EVOLUTIONARY ALGORITHM

Since both DMOEA and DPSMO have significant advantages and drawbacks, we can integrate particle swarm and evolutionary algorithm together in order to take advantages of both algorithms and improve the quality of the evolved solutions. In one aspect, evolutionary algorithm can help each individual share its information with any other individuals instead of only focusing on the best individuals. On the other hands, particle swarm can inform an individual which direction might be the best way to go and how fast its velocities should be. Therefore, inspired

by both algorithms, a Dynamic Particle Swarm Evolutionary Algorithm (DPSEA) is designed to improve *efficiency* and *efficacy* of evolutionary process.

7.1 Dynamic Particle Swarm Evolutionary Algorithm

The main skeleton of DPSEA is constructed based on DPSMO. Nevertheless, in addition to the location updating strategy of particle swarm, the individuals will perform crossover operation as well. At each generation, an offspring may be generated through two mechanisms—updating the location of a cloned individual or performing crossover between two selected parents. Population growing strategy will be used to determine if an offspring generated through crossover can survive to the next generation and population declining strategy is applied to remove an existing ill-fitted individual. Therefore, comparing to DPSMO, the only change in DPSEA is adding a crossover operation and a population growing strategy borrowed from DMOEA in both of rank and density subpopulations. By adding these two operations, the running interval for each generation may increase comparing to DMOEA and DPSMO because there are two information-sharing actions performed in DPSEA. However, this sacrificing will be worthy if these two actions can assist each other and find more valuable individuals than using only one information sharing action.

7.2 Comparison Study on DMOEA, DPSMO and DPSEA

To compare the performance of DPSEA with DMOEA and DPSMO, two benchmark problems— Function ***F1*** and ***F2*** are tested.

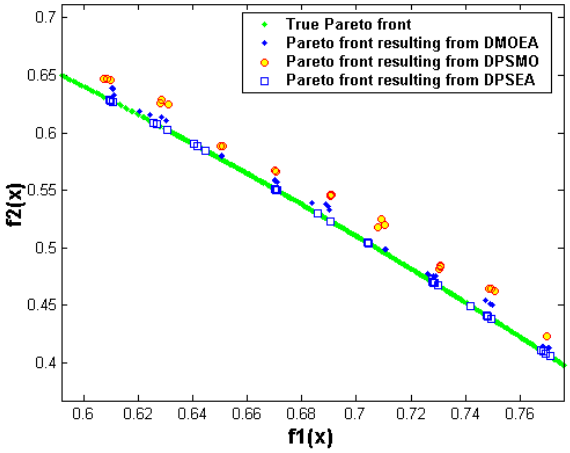


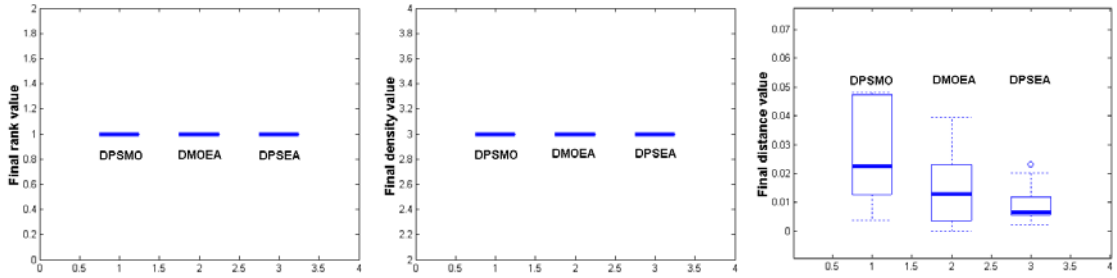
Figure 7 Resulting Pareto fronts by DMOEA, DPSMO and DPSEA on Function *F1*

7.2.1 Simulation on Function F1

Function ***F1*** is same with the test function used in Section VI. Figure 7 shows the zoomed sample of the true Pareto front and the resulting Pareto fronts by DMOEA, DPMO and DPSEA. Figures 8 (a) – (c) show the Box plots for the final rank, density and distance indicators over 50 runs, respectively. The performance measures of $C(X_i, X_j)$ for the comparison sets between algorithms i and j are shown in Figure 9, where algorithms 1 – 3

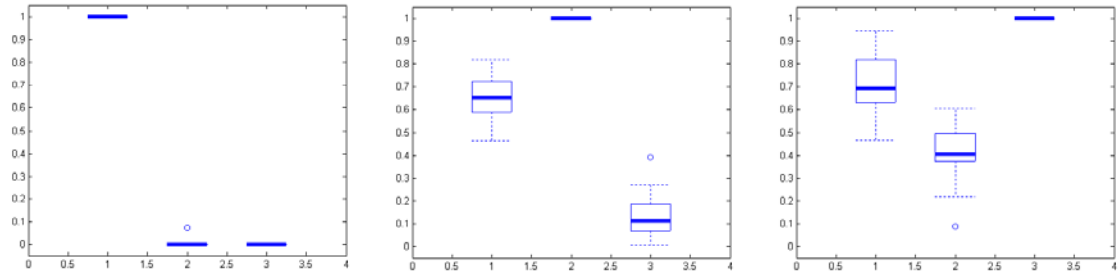
represent DPSMO, DMOEA and DPSEA, respectively. Moreover, the evolutionary trajectories of the population size and average individual rank, density and distance values over 50 runs by three algorithms are shown in Figures 10 (a) – (d), respectively.

From Figures 7 – 9, we can see that all three algorithms have the capability to converge to a Pareto front with rank value and density value equal to 1 and 3, respectively. However, from Figures 7 and 8(c), it is obvious that DPSEA's resulting Pareto front is closer to the true Pareto front than those produced by the other two algorithms. In addition, Figure 9 (c) shows that about 70% and 45% of final populations resulting from DPSMO and DMOEA are covered by DPSEA and 0% and 10% of population resulting from DPSEA are covered by DPSMO and DMOEA, respectively. This result proves that DPSEA produce better Pareto fronts than the other two algorithms in terms of finding *near-optimal*, *near-complete* and *uniformly distributed* Pareto front.



(a) final rank value **(b) final density value** **(c) final distance value**

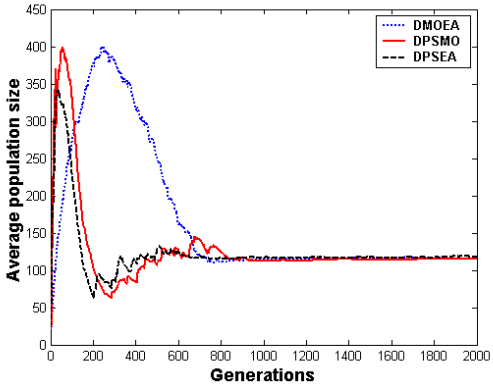
Figure 8 Box plots of three indicators on Function *F1*



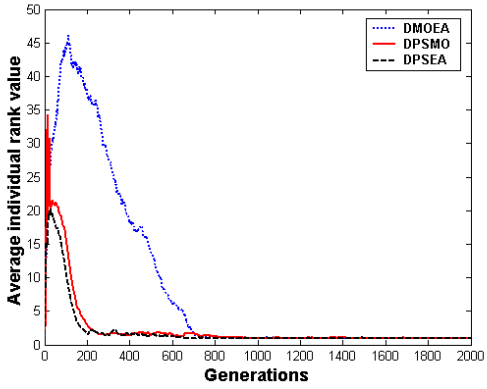
$C(X_1, X_{1-3})$ $C(X_2, X_{1-3})$ $C(X_3, X_{1-3})$

Figure 9 Box plots based on C measure on Function *F1*

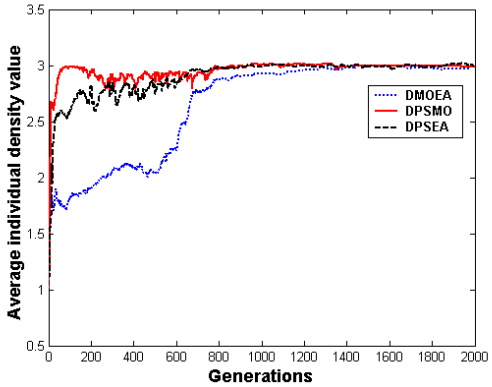
Form Figures 10 (a) – (d), it is observed that DPESA is even faster than DPSMO in terms of generation numbers to converge. This phenomenon shows that two information-sharing techniques can promote each other and help entire population converges relatively faster than any single one of them. When both of the techniques assist evolutionary process, it will be much easier for an individual to find a better-fitted offspring. These newly generated offspring will continuously approaching true Pareto front and making previously non-dominated individuals into dominated points, which will be eliminated by population declining strategy. This mechanism explains why DPSEA produces lowest distance value within smallest number of generations as shown in Figure 9(d).



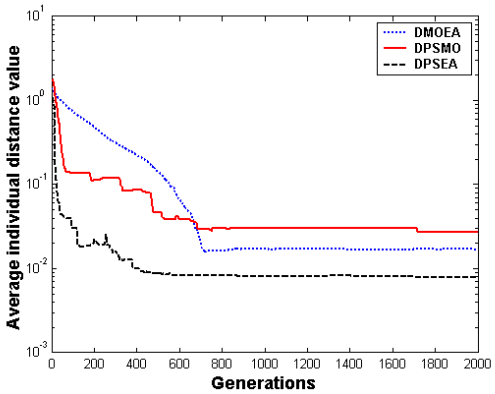
(a) Evolutionary trajectories of population sizes



(b) Trajectories of average rank values



(c) Trajectories of average density value



(d) Trajectories of average distance values

Figure 10 Evolutionary trajectories for the population size and the values of three indicators resulting by DMOEA and DPSMO and DPSEA on Function *F1*

Minimize $f_1(x)$, $f_2(x)$ and $f_3(x)$, where

$$f_1(x) = (1 + g(\mathbf{x}))\cos(\frac{\pi x_1}{2})\cos(\frac{\pi x_2}{2}), \quad (5)$$

$$f_2(x) = (1 + g(\mathbf{x}))\cos(\frac{\pi x_1}{2})\sin(\frac{\pi x_2}{2}),$$

$$f_3(x) = (1 + g(\mathbf{x}))\sin(\frac{\pi x_1}{2}),$$

$$g(\mathbf{x}) = \sum_{i=1}^{12}(x_i - 0.5)^2,$$

subject to $0 \le x_i \le 1, \quad i = 1, \mathrm{K}, 12.$

7.2.2 Simulation on Function *F2*

Function ***F2*** was first proposed in [17] as an MOP with high-dimensional decision and objective spaces. Meanwhile, the true Pareto front of ***F2*** is exactly the first quadrant of a unit sphere as shown in Figure 11. Since the mathematical expression of the true Pareto front is clearly defined as Equation (5), the distance between the final

and true Pareto front can be precisely calculated. The desired population size can be determined based on the ppv value and the grid scales K_1 - K_3.

For Function ***F2***, the initial population, the age threshold, the population size per unit volume, ppv, the cell scales K_1, K_2 and K_3 and stopping generations are selected as 2, 10, 3, 20, 20, 20 and 10,000, respectively. Meanwhile, for each test function, the final Pareto front, trajectories of population size, average rank, density and distance values, Box plots of final rank, density and distance values and *C* values resulting from all three algorithms are illustrated.

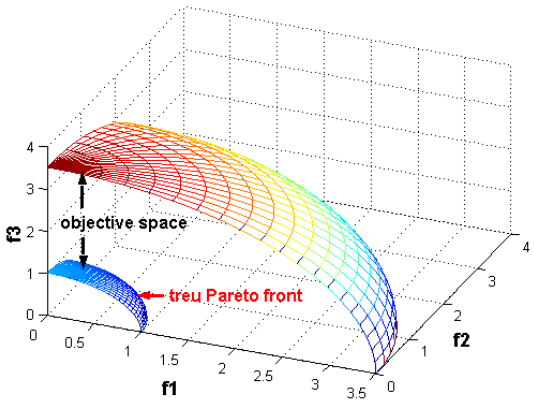
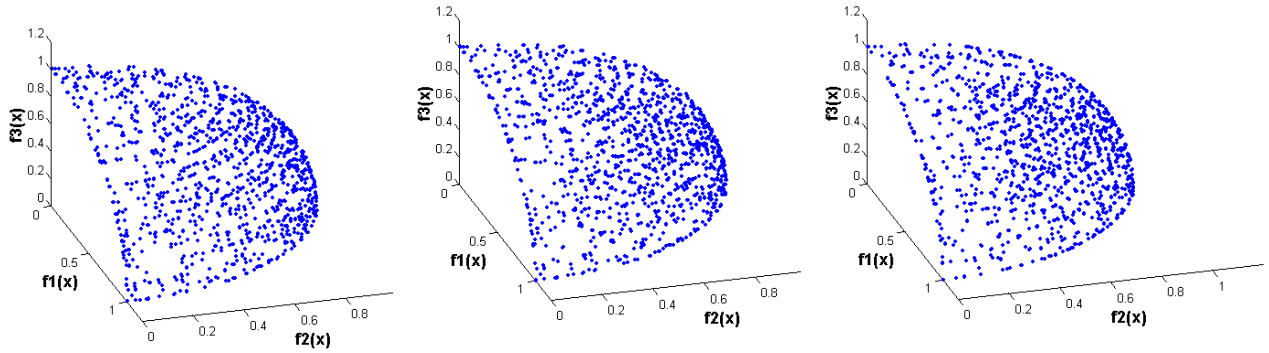


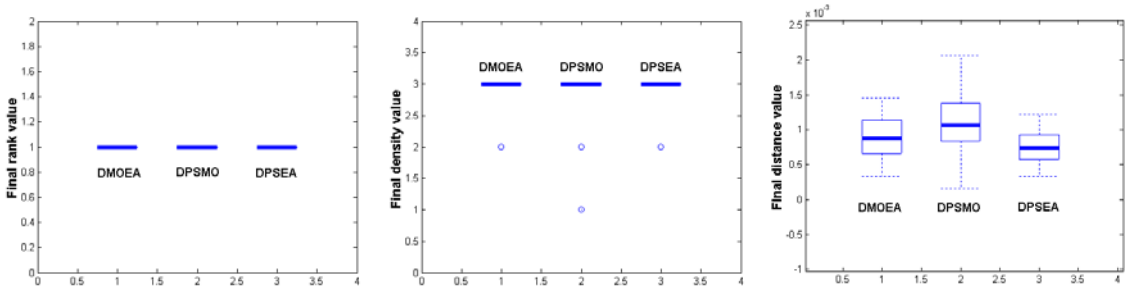
Figure 11 Objective space and Pareto front of *F2*



(a) DMOEA **(b) DPSMO** **(c) DPSEA**

Figure 12 Resulting Pareto fronts from DMOEA, DPSMO and DPSEA on Function *F2*

Figure 12(a) – (c) shows the resulting Pareto fronts by DMOEA, DPMO and DPSEA, respectively. Figures 13(a) – (c) show the Box plots for the final rank, density and distance indicators over 50 runs, respectively. The performance measures of $C(X_i, X_j)$ for the comparison sets between algorithms i and j are shown in Figure 14, where algorithms 1 – 3 represent DPSMO, DMOEA and DPSEA, respectively. Moreover, the evolutionary trajectories of the population size and average individual rank, density and distance values over 50 runs by three algorithms are shown in Figures 15 (a) – (d), respectively.



(a) final rank value **(b) final density value** **(c) final distance value**

Figure 13 Box plots of three indicators on Function *F2*

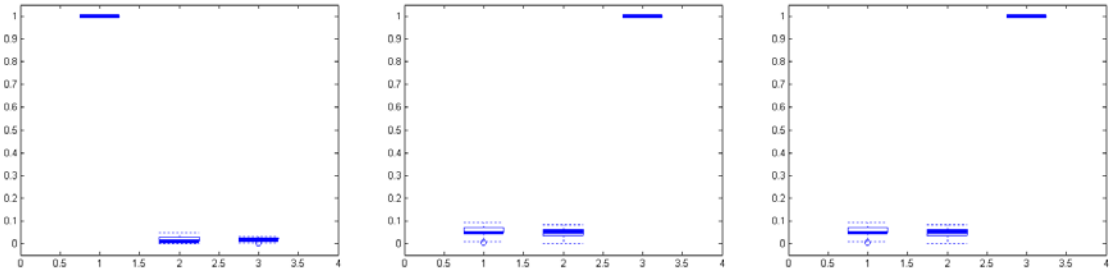
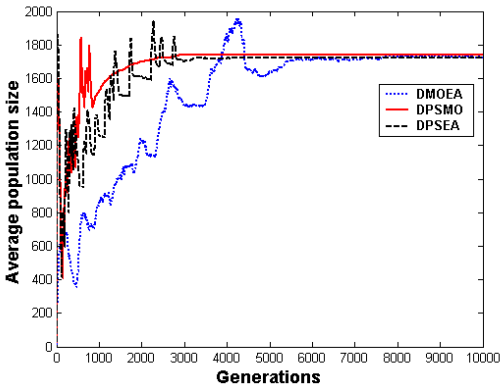


Figure 14 Box plots based on C measure on Function *F2*

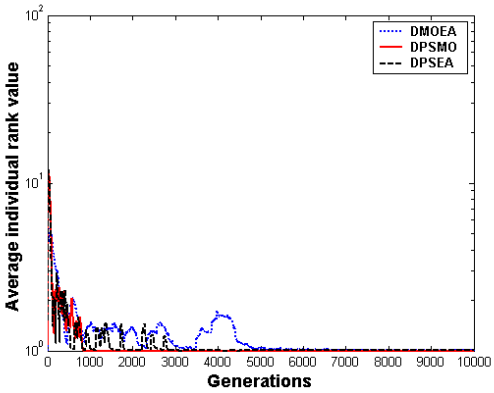
From Figures 12(a) – (c), we can see that all three algorithms result in completive Pareto fronts from their appearances. Meanwhile, from Figures 13 (a) – (c) and 14 (a) – (c), we cannot find significant differences from the indicators of final results from DMOEA and DPSMI. Since Function ***F2*** does not generate any local Pareto front, there will be no hindrance for DPSMO to locate true Pareto front. However, by applying two information-sharing techniques, DPSEA still shows its ability to approximate more accurate Pareto front than the other two algorithms as shown in Figure 13 (c) and 15 (d).

Furthermore, by examining the evolutionary trajectories as shown in Figures 15 (a) – (d), we can see remarkable difference among three algorithms in terms of convergence property. It only takes DPSMO less than 1,000 generations to converge and DMOEA and DPSEA needs about 7,000 and 3,500 generations, respectively. Table 2 shows the average running time per generation for each of three algorithms. The "CPUTIME" command from MATLAB (version 6.1) is used to measure the time elapsed for each algorithm implemented in MATLAB and a HP computer with dual 2-GHz processors and 1-GByte RAM is used for simulation. From Table 2, we can see that DPSMO runs faster than DMOEA and DPSEA at each generation. This advantage is contributed by DPSMO's population growing method, which does not evaluate newly generated offspring. DPSEA is a bit slower than DMOEA since it applies two information-sharing techniques at each generation. However, because the most time consuming parts are population declining strategy and objective compression strategy, which are used by all three algorithms, the difference of time consuming per generation for these algorithms is not remarkable. Therefore, from the above observations, it is clear that DPSMO has the fastest convergence speed since it spends least time on each generation and takes smallest number of generations to converge. Although DPSEA will spend a little longer time

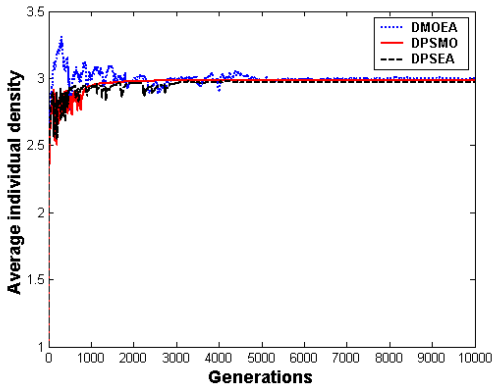
on each generation than DMOEA, the total time consuming of convergence for DPSEA is still significantly shorter than DMOEA since DPSEA takes much smaller number of generations to converge. In addition, we need to keep in mind that DPSEA will produce more accurately approximated Pareto front than the other two algorithms in terms of distance values and DPSMO may generate less competitive Pareto front, especially when there are local Pareto fronts exist for the given MOP. Nevertheless, combining particle swarm optimization with evolutionary algorithms provides a potential way to design an MOEA in solving real world MOPs that need less processing time to generate qualified Pareto fronts.



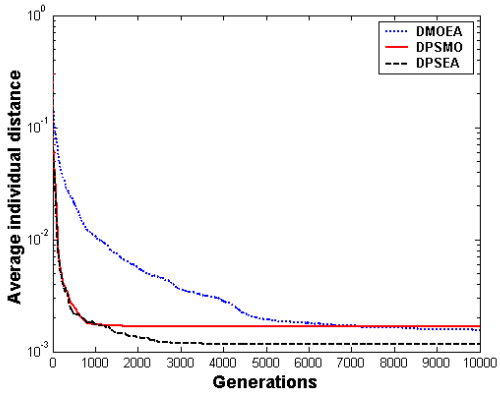
(a) Evolutionary trajectories of population sizes



(b) Trajectories of average rank values



(c) Trajectories of average density value



(d) Trajectories of average distance values

Figure 15 Evolutionary trajectories for the population size and the values of three indicators resulting by DMOEA and DPSMO and DPSEA on Function *F2*

Table 2 Comparison results of computation time of *F2* from DMOEA, DPSMO and DPSEA

	DMOEA	DPSMO	DPSEA
Time (sec)	**0.18**	**0.15**	**0.20**

VII. CONCLUSIONS

In this reprot, a new particle swarm optimization based multiobjective evolutionary algorithms—Dynamic Particle Swarm Multiobjective Optimization (DPSMO) algorithm is proposed. DPSMO can be characterized as a) efficiently guiding an individual's direction and speed in decision space, b) simplifying the computation complexity by using a cell-based rank and density fitness estimation scheme, c) effectively refining individual's rank and density values by objective compression strategy, d) adaptively increasing or decreasing population size based on rank and density status of current population and e) converging to an optimal (desired) population size by exploring all the non-dominated hyperareas with a fixed number of Pareto points. The first characteristic of DPSMO is contributed by PSO's information-sharing method, and the rest of characteristics are caused by the optimization techniques of Dynamic Multiobjective Evolutionary Algorithm (DMOEA). In the experimental study, DPSMO significantly improve the convergence speed comparing to DMOEA. However, the optimality of the resulting Pareto front by DPSMO is not as good as that of DMOEA. This deficiency of DPSMO is caused by PSO's one-way information sharing mechanism. In order to cope with DPSMO's deficiency, a Dynamic Particle Swarm Evolutionary Algorithm is devised. DPSEA includes both PSO and EA's information techniques and in the experimental results, it produces Pareto fronts with higher quality comparing to both DPSMO and DMOEA. Meanwhile, DPSEA also provides a faster convergence speed than DMOEA. Therefore, DPSMO and DMOEA can be potential candidates in solving real-time or on-line MOPs. However, as the test functions used in this reprot are still far from covering all the challenging characteristics of MOPs, a more profound study in applying DPSMO and DPSEA in dealing with real world MOPs is absolutely necessary in future work.

REFERENCES

[1] C. M. Fonseca and P. J. Fleming, "An overview of evolutionary algorithms in multiobjective optimization," *Evol. Comput.*, vol. 3, pp. 1-16, 1995.

[2] J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms," in *Proc. 1st Int'l Conf. Genetic Algorithms*, pp. 93-100, 1985.

[3] K. Deb, S. Agrawal, A. Pratap and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, pp. 182-197, 2002.

[4] G. G. Yen and H. Lu, "Dynamic population size in multiobjective evolutionary algorithm," in *Proc. 9th IEEE Cong. Evol. Comput.*, pp. 1648-1653, 2002.

[5] J. Kennedy and R.C. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Networks*, pp. 1942-1948, 1995.

[6] N. Srinivas and K. Deb, "Multi-Objective function optimization using non-dominated sorting genetic algorithms," *Evol. Comput.*, vol. 2, pp. 221-248, 1994.

[7] J. Horn, N. Nafpliotis and D. E. Goldberg, "A niched pareto genetic algorithm for multiobjetcive optimization," in *Proc. 1st IEEE Cong. Evolutionary Computation,* pp. 82-87, 1994.

[8] H. Lu and G. G. Yen, "Rank-density based multiobjective genetic algorithm," in *Proc. 9th IEEE Cong. Evol. Comput.*, pp. 944-949, 2002.

[9] X, Hu and R.C. Eberhart, "Multiobjective optimization using dynamic neighborhood particle swarm optimization," in *Proc. 9th IEEE Cong. Evol. Comput.*, pp. 1677-1681, 2002.

[10] J. Moore and R. Chapman, *Application of particle swarm to multiobjetcive optimization*, Department of Computer Science and Software Engineering, Auburn University, 1999.

[11] T. Ray, T. Kang and S. K. Chye, "Multiobjective design optimization by evolutionary algorithm," *Engineering Optimization*, 2002 (in Press).

[12] C.A.Coello and M. S. Lechuga, "MOPSO: A proposal for multiple objective particle swarm optimization," in *Proc. 9th IEEE Cong. Evol. Comput.*, pp. 1051-1056, 2002.

[13] J. Kennedy, "The particle swarm optimization: social adaptation of knowledge," in *Proc. 4th IEEE Cong. Evolutionary Computation,* pp. 303-308, 1997.

[14] D. A. Van Veldhuizen and G. B. Lamont, "On measuring multiobjective evolutionary algorithm performance," in *Proc. 7th IEEE Cong. Evol. Comput.*, pp. 204-211, 2000.

[15] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach," *IEEE Trans. Evol. Comput.*, vol. 3, pp. 257-271, 1999.

[16] K. Deb, "Multiobjective genetic algorithms: problem difficulties and construction of test problems," *Evol. Comput.*, vol. 7, pp. 205-230, 1999.

[17] K. Deb, L. Thiele, M. Laumanns and E. Zitzler, "Scalable multi-objectvie optimization test problems," in *Proc. 9th IEEE Cong. Evol. Comput.*, pp. 825-890, 2002.