Combining PSO and SSGA to support multiobjective decision making

B. N. Al-Matrafi[1], Abdallah A. Mousa[1,2], ,Abdallah A. Galal[1,3]

[1]Department of Mathematics and Statistics, Faculty of Sciences, Taif University, Taif, KSA
[2]Department of Basic Engineering Science, Faculty of Engineering, Menoufia University, Shibin Al Kom, Egypt
[3]Physics and Engineering Mathematics Department, Faculty of Engineering, Tanta University, Tanta, Egypt

Abstract— Particle swarm optimization (PSO) is a computational method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. In this paper, we present a hybrid algorithm combining particle swarm optimization (PSO) with steady state genetic algorithm (SSGA) for solving multiobjectuve decision making (MODM) problems. The methodology combines and extends the attractive features of both PSO and SSGA, where it is based on PSO to get approximate nondominated set of the problem followed by SSGA to improve the solution quality, where Steady State GA is an alternative to the engine in order to improve the spread of the solutions found so far. The results, provided by the proposed algorithm for engineering problems, are promising when compared with exiting traditional GA approach that is based on the partial replacement of the parent population, instead of the whole population. Then, in the second stage, rough set theory is adopted as local search well-known algorithms. Also, our results suggest that our algorithm is better applicable for solving real-world application problems.

Index Terms— multiobjective optimization., swarm optimization , rough set theory, steady state genetic algorithm

1 INTRODUCTION

In multiobjective optimization, several conflicting objectives have to be minimized simultaneously. Generally, no unique solution exists but a set of mathematically equally good solutions can be identified, by using the concept of Pareto optimality. Many applications of multiobjective optimization can be found in engineering [1–2], economics and finance [3], medicine [4-5], management and planning [6], etc. There exist many solution strategies to solve the multiobjective optimization problems. One of the basic approaches is the weighting method (see [7]), where one single-objective optimization problem is formed by weighting several objective functions. Similar problem has the ε-constraint method, introduced in [7].

Recently, there has been a boom in applying evolutionary algorithms to solve multiobjectuve optimization problems [8–11]. Evolutionary algorithms (EAs) are stochastic search methods that mimic the metaphor of natural biological evolution and/or the social behavior of species. The development of metaheuristic optimization theory has been flourishing. Many metaheuristic paradigms such as genetic algorithm [8,12,13], simulated annealing [14], tabu search [14,15], and the ant colony algorithm(ACO) [16] has become an interesting approach to solve many hard problems.

Recently particle swarm optimization PSO [11,17] have shown their efficacy in solving computationally intensive problems. Particle Swarm Optimization is an evolutionary computation technique, developed for optimization of continuous nonlinear, constrained and unconstrained, non differentiable multimodal functions [18]. PSO is inspired firstly by general artificial life, the same as bird flocking, fish schooling and social interaction behaviour of human and secondly by random search methods of evolutionary algorithm[19]. Animals, especially birds, fishes etc. always travel in a group without colliding, each member follows its group, adjust its position and velocity using the group information, because it reduces individual's effort for search of food, shelter etc. Particle swarm optimization is evolutionary technique similar to genetic algorithm because both are population based and are equally affective. Particle swarm optimization has better computational efficiency, i.e. it requires less memory space and lesser speed of CPU, it has less number of parameters to adjust. Genetic algorithm and other similar techniques (e.g. simulated annealing), work for discrete design variables, whereas particle swarm optimization work for discrete as well as analogue systems, because it is inherently continuous, does not need D/A or A/D conversion. Although for handling discrete design variables Particle swarm optimization needs some modification to be done in particle swarm optimization methods.

All GA's are based on the principles developed by John Holland in his book ''adaptation in natural and artificial systems [20]. Holland outlined the methods for successfully implementing population based adaptive optimizers. Holland's methods operate on the principle of survival of the fittest. In a computational sense, candidate solutions are assembled in a population and compared to one another, the weak die off and the strong are left to reproduce and mutate to produce better children. Binary encoded generational GA has been used extensively for optimization and has been shown to be a very versatile and robust method for optimization. A primary disadvantage of the binary coded GA comes from the fact that because all of the variables must be converted into a single bit string, the solution accuracy is dependant on the number of bits that can be used for the string. Because of the large ranges associated with many of the design parameters the smallest resolution for the binary GA is generally limited to about 1% of the solution space for complex problems while the real coded GA is only limited to a double precision number. Resolu-

tion is not a significant issue with a real coded GA because all of the variables remain real double precision variables. Dozier et al. [21], Unsal et al. [22] and Dozier et al.[23] demonstrated the ability for a real coded GA to achieve shorter run times as well as more accurate solutions for some applications. The key difference is that in the steady state GA for each generation only the worst performer is thrown out and replaced by a new member, whereas for the generational GA all of the members of the population are thrown out and replaced (expect in elitist mode when the best member remains in the next generation) using a similar tournament routine. For complex problems the steady state GA may not be as efficient as the generational GA because of its lack of diversity. For the steady state GA, once the survivors have had a chance to crossover (i.e. pass genetic material back and forth through their variables), the new member replacing the worst performer is run back through the objective function. This process continues, with the parents producing on average better offspring, until the maximum number of generations (user specified) is reached. There are proofs [24,25] which show why this process produces increasingly superior performers in a population, but a simplistic view is that a good parent mated with another good parent, is more likely to produce good offspring than two poor parents when mated. This is not to say that two good parents cannot produce poor performers. Rather, when two good performers exchange genes, statistically the resulting offspring have a higher chance of outperforming their parents. With the above concepts of PSO and GA and difficulties with the classical methods, we propose an enhanced PSO methodology by which a set of Pareto-optimal solutions will be found, thereby eliminating the need of any weight vector and the need of applying the method again and again. Instead of finding a single solution corresponding to a particular weight vector,

In this paper, we combine PSO with a steady state genetic algorithm SSGA. The proposed methodology combines and extends the attractive features of both PSO and SSGA, where it is based on PSO to get approximate nondominated set of the problem followed by SSGA to improve the solution quality. Then in the second stage, rough set theory is adopted as local search engine in order to improve the spread of the solutions found so far. The algorithm was tested on a set of engineering problems and the obtained performance is compared against approaches and state of-the-art approaches. Based on the results presented some conclusions are drawn and the future work is established.

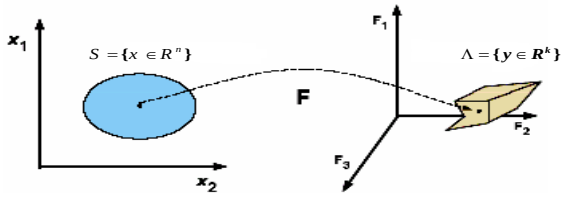
2. PROBLEM FORMULATION

A general multiobjective optimization problem is expressed [26] by

MOP :

$$Min \quad F(x) = (f_1(x), f_2(x), \ldots, f_m(x))^T$$
$$s.t. \quad x \in S$$
$$x = (x_1, x_2, \ldots, x_n)^T \tag{1}$$

Where $(f_1(x), f_2(x), \ldots, f_k(x))$ are the m objectives functions, $(x_1, x_2, \ldots, x_n)$ are the n optimization parameters, and $S \in R^n$ is the solution or parameter space. Obtainable objective vectors, $\{F(x) \mid x \in S\}$ are denoted by Λ, so $\{F : S \to \Lambda\}$, S is mapped by F onto Λ. This situation is represented in fig.1 for the case n=2, m=3.



"Decision variable space" "Objective function space"

Fig. 1: MOP evaluation n mapping.

Because F(x) is a vector, there is no unique solution to this problem; instead, the concept of nondominated (also called Pareto optimality) must be used to characterize the objectives. A nondominated solution is one in which an improvement in one objective requires a degradation of another (Fig.2).

Definition 1.(Pareto optimal solution)[27]: x^* is said to be a Pareto optimal solution of MOP if there exists no other feasible x (i.e., $x \in S$) such that, $f_j(x) \le f_j(x^*)$ for all $j = 1, 2, \ldots, k$ and $f_j(x) < f_j(x^*)$ for at least one objective function f_j.

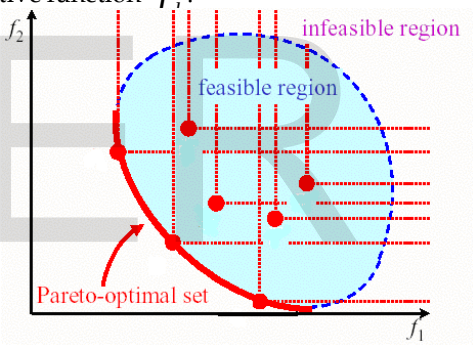


Fig. 2: The concept of Pareto optimality

3. USE OF ROUGH SETS IN MULTIOBJECTIVE OPTIMIZATION

For our hybrid approach we try to investigate the Pareto front using a Rough sets grid. To do this, we will use an initial approximate of the Pareto front (provided by any evolutionary algorithm) and will implement a grid in order to get more information about the true Pareto front [28].

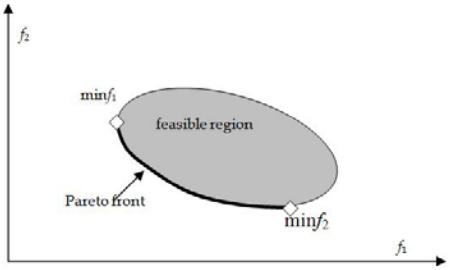


Fig. 3: nondominated solution (Pareto front) and dominated solutions

To create this grid, as an input we will have N feasible points divided in two sets: the nondominated points (NS) and the dominated ones (DS) (see figure 3). Using these two sets we want to create a grid to describe the set NS in order to intensify the search on it. This is, we want to describe the Pareto front in the decision variable space because we could easily use this information to generate more efficient points and then improve this initial Pareto set approximation. Figure 4 (taken from [28]) shows how information in objective function space can be translated into information in decision variable space through the use of a grid. We must note the importance of the DS sets as in a rough sets method, where the information comes from the description of the boundary of the two sets NS, DS. Then the more efficient points provided the better. However, it is also required to provide dominated points, since we need to estimate the boundary between being dominated and being nondominated. Once the information is computed we can simply generate more points in the "efficient side".

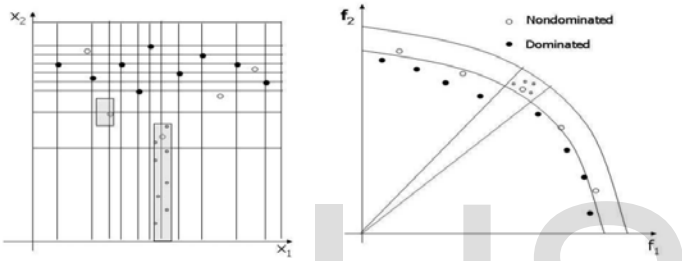


Fig. 4: Decision variable space (left) and objective function space (right)

4. PARTICLE SWARM OPTIMIZATION

PSO [18] is an evolutionary computation approach motivated by the simulation of social behaviour. Namely, each individual (agent) utilizes two important kinds of information in decision process. The first one is their own experience; that is, they have tried the choices and know which state has been better so far, and they know how good it was. The second one is other agent's experiences; that is, they have knowledge of how the other agents around them have performed. Namely, they know which choices their neighbors have found are most positive so far and how positive the best pattern of choices was. In the PSO system, each agent makes his decision according to his own experiences and other agent's experiences. The system initially has a population of random solutions. Each potential solution, called a particle (agent), is given a random velocity and is flown through the problem space. The agents have memory and each agent keeps track of its previous (local) best position (called the Pbest) and its corresponding fitness. There exist a number of Pbest for the respective agents in the swarm and the agent with greatest fitness is called the global best (Gbest) of the swarm. Each particle is treated as a point in a n-dimensional space. The i-th particle is represented as Xi = (xi1, xi2, . . . , xin). The best previous position of the i-th particle (Pbesti) that gives the best fitness value is represented as Pi = (pi1,pi2, . . , pin). The best particle among all the particles in the population is represented by Pg = (pg1,pg2, . . . , pgn). The velocity for particle i (i.e., the rate of the position change) is represented as Vi = (vi1, vi2, . . , vin).

The particles are manipulated according to the following equations (the superscripts denote the iteration):

$$v_i^{k+1} = wv_i^k + c_1 r_1\left(p_i - x_i^k\right) + c_2 r_2\left(p_g - x_i^k\right) \quad (2)$$

$$x_i^{k+1} = x_i^k + v_i^{k+1} \quad (3)$$

Where $i = 1,2,.....,N$, and N is the size of the population; w is the inertia weight; c1 and c2 are two positive constants, called the cognitive and social parameter respectively; r1 and r2 are random numbers uniformly distributed with in the range [0,1]. Equation (2) is used to determine the i-th particle's new velocity v_i^{k+1}, in each iteration, while equation (3) provides the new position of the i-th particle x_i^{k+1}, adding its new velocity v_i^{k+1}, to its current position x_i^k. Figure 2 shows the description of velocity and position updates of a particle for a two-dimensional parameter space.

5. THE PROPOSED APPROACH

The proposed methodology introduces a hybrid algorithm combining PSO and steady state genetic algorithm to improve the performance of each algorithm. Also, to improve the solution quality Local Search-Inspired Rough Sets is implemented as a neighborhood search engine, where it intends to explore the less-crowded area in the current archive to possibly obtain more nondominated solutions nearby. The proposed algorithm consists of two stages. The description diagram of the proposed algorithm is described as follows:

Stage 1: PSO

In this subsection, the procedure of PSO algorithm is described, which consists of the following steps:

Step 1: Initialize parameters for PSO, initialize randomly N particles with position $X_i^{t=0}$ with velocities $V_i^{t=0}$ where t is the time counter and $i = 1,....,N$.

Step 2: Identify the local set $\left(L_i^{t=0}\right)$ for each particle as $L_i^{t=0} = \left\{X_i^{t=0} \middle| \; i = 1,...,N\right\}$. Also, identify the local preferred element $\left(LP_i^{t=0} \subset L_i^{t=0}\right)$ of the i-th particle as $\forall \text{particle } i \;\; \exists LP_i^{t=0} = \left\{X_i^{t=0}\right\}$.

Step 3: Collect all local sets $\left\{L_i^{t=0}\right\} \forall i = 1,...,N$ in a pool C such that $C = \bigcup_{i=1}^{N} L_i^{t=0}$.

Step 4: Define a global set $G^{t=0} = ND(C)$, where $ND(\cdot)$ refers to the function which has the ability to detect all nondominated solutions.

Step 5: In the objective space, The distances between $X_i^{t=0} \; \forall i = 1,...,N$ and the members in $G^{t=0}$ are measured using the Euclidean distance, where the distance between any two d-dimensional points $\vec{x}_i$ and $\vec{x}_j$ is given by

$$d\left(\vec{x}_i, \vec{x}_j\right) = \left\|\vec{x}_i - \vec{x}_j\right\|_2 = \sqrt{\sum_{p=1}^{d}\left(x_{i,p} - x_{j,p}\right)^2} \qquad (4)$$

The nearest member in G^t to the i-th particle is defined as the global preferred element GP_i^t, For more details we refer the reader to[29].

Step 6: Set the external set $E^{t=0}$ equal to $G^{t=0}$, For more details we refer the reader to[29].

Step 7: Update particles: Update the velocity v_i^t and position x_i^t of each particle according to the following equations:

$$v_i^{t+1} = wv_i^t + c_1 r_1\left(LP_i^t - x_i^t\right) + c_2 r_2\left(GP_i^t - x_i^t\right) \qquad (5)$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \qquad (6)$$

Step 8: Evolution of particles: To restrict (control) the particle's velocity v_i^t, a modified constriction factor (i.e., dynamic constriction factor) is presented to keep the feasibility of the particles. e.g., Figure 5 shows the movement of the particle i through the search space without any control factor (dashed line) also with a modified constriction factor (Solid line). Where the particle i start at position x_i^t with velocity v_i^t in the feasible space, the new position x_i^{t+1} depends on velocity v_i^{t+1} making the particle to lose its feasibility, so we use a modified constriction factor χ as follows [29]:

$$\chi = 2\Big/\left|-2-\tau-\sqrt{\tau^2+\tau}\right| \qquad (7)$$

Where, τ is the age of the infeasible particle (i.e., How long it's still infeasible) and it is increased with the number of failed trial to keep the feasibility of the particle. The new modified positions of the particles are computed as:

$$\overline{x}_i^{t+1} = x_i^t + \chi\, v_i^{t+1} \qquad (8)$$

For each particle, the feasibility is checked, if it is infeasible, χ parameter is implemented to control its position and velocity (see figure 6).

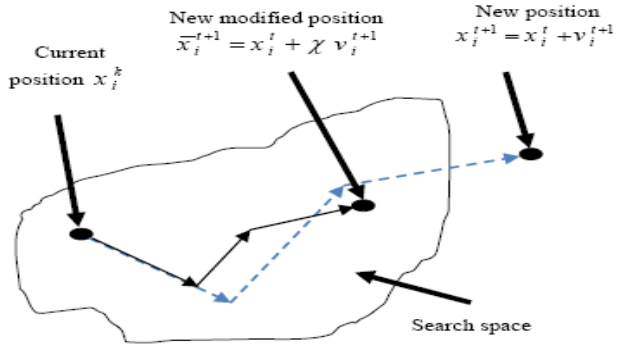


Fig. 5. The movement of the particle i through search space (taken from [29]).

Algorithm 1: Evolution of particles

input$\left(x_i^t, v_i^{t+1}, x_i^{t+1}\right)$
while
  x_i^{t+1} is unfeasible | number of trial not satisfie
  generate $\overline{x}_i^{t+1} = x_i^t + \chi\, v_i^{t+1}$
  $x_i^{t+1} = \overline{x}_i^{t+1}$
end
output$\left(v_i^{t+1}, x_i^{t+1}\right)$

Fig. 6. Evolution of particles.

Step 9: Update local set L_i^t : The new position of each particle x_i^{t+1} is added to L_i^t to form L_i^{t+1} which is updated according to algorithm 2 in figure 7.

Step 10: Update global set G : $G^{t+1} = ND\left(\bigcup_{i=1}^{N} L_i^{t+1}\right)$ which contain all nondominated solution of $\bigcup_{i=1}^{N} L_i^{t+1}$ (see figure 7).

Step 11: Update external set E^t : Copy the members of G^{t+1} to E^t and dominance criteria is applied to remove all dominated solution from E^t (i.e., each member of G^{t+1} has three probabilities as in algorithm3 in figure 8).

Step 12: Update local preferred element LP_i^{t+1} and global preferred element GP_i^{t+1} for each particle : In the objective space, The distances between $X_i^{t+1} \forall i = 1,\ldots,N$ and members in L_i^{t+1} are measured using equation (4). The nearest member in L_i^{t+1} to the i-th particle is defined as LP_i^{t+1}. Also, The distances between $X_i^{t+1} \forall i = 1,\ldots,N$ and the members in G^{t+1} are measured using equation (4). The nearest member in G^{t+1} to the i-th particle is defined as the global preferred GP_i^{t+1}.

Stage 2: Steady state genetic algorithm

Steady state genetic algorithm was implemented in such way that, two offspring are produced in each generation. Parents are selected to produce offspring and then a decision is made as to which individuals in the population to select for deletion to make room for the new offspring. A replacement/ deletion strategy defines which member of the population will be replaced by the new offspring. The main steps of the SSGA are summarized as follows

Algorithm 2: Update local set L_i^t

Input $\left(L_i^t, X_i^{t+1}\right)$

If $\exists X \in L_i^t \mid X \succ X_i^{t+1}$ then

$L_i^{t+1} = L_i^t$

Else if $\exists X \in L_i^t \wedge X_i^{t+1} \succ X$ the

$L_i^{t+1} = L_i^t \bigcup \left\{X_i^{t+1}\right\} / \{X\}$

Else if $\nexists X \in L_i^t \mid X \succ X_i^{t+1}$ then

$L_i^{t+1} = L_i^t \bigcup \left\{X_i^{t+1}\right\}$

End

Output $\left(L_i^{t+1}\right)$

Fig. 7.Update local set L_i^t

Algorithm 3: Update external set E^t

Input $\left(E^t, X \in G^{t+1}\right)$

If $\exists Y \in E^t \mid Y \succ X$ then

$E^t = E^t$

Else if $\exists Y \in E^t \wedge X \succ Y$ then

$E^t = E^t \bigcup \{X\} / \{Y\}$

Else if $\nexists Y \in E^t \mid Y \succ X$ then

$E^t = E^t \bigcup \{X\}$

End

Output $\left(E^t\right)$

Fig. 8.Update external set E^t

Step1 : Selection : This process selects two parents from the population ($x_i, x_j \;\&\; i, j = 1,2,...,m$). The mechanism for selecting the parents is based on a tournament selection. Tournament selection works by randomly drawing a number of individuals from the population to create a tournament. The winner from the tournament is the individuals with best fitness and is used as a parent. In this way all parents necessary for a reproduction operator are selected.

Step 2: Recombination Operators: Recombination is a process in which new individuals are generated by exchanging features of the selected parents with the intent of improving the fitness of the next generation. This process is sometimes called crossover. These new individuals are then subjected to mutation. There are a number of different ways in which the recombination operation can be implemented. The following describes the mechanisms of crossover and mutation.

Crossover: Once the parents are created, the crossover step is carried out by replacing the current value with a new one which produced stochastically with a probability proportional to the crossover probability. Suppose the crossover probability set by the system is p_C. Generating a random number $r \in [0,1]$, the crossover operation could be carried out only if $r < p_C$. In the operation of crossover, we first generate two parents at random from the population space. Suppose x_i and x_j are two initial values of the population and $\alpha \in [0,1]$ is a random number. The result of crossover operation x_i' and x_j' can be obtained by the following linear combination of x_i and x_j :

$$x_i' = \alpha . x_i + (1-\alpha) . x_j \qquad (9)$$

$$x_j' = (1-\alpha) . x_i(1) + \alpha . x_j$$

Mutation: Once the, the crossover is performed, the mutation step is carried out by replacing the current value with a new one which produced stochastically with a probability proportional to the mutation probability. Suppose the mutation probability set by the system is p_m .Generating a random number $r \in [0,1]$, the mutation operation is implemented only if $r < p_m$ .Suppose $x(j)$ will be transformed into $x'(j)$ after mutation as follows:

$$x'(j) = L(j) + \lambda * [U(j) - L(j)] \;,\; j = 1,2,..,n \qquad (10)$$

Where $\lambda \in [0,1]$ is a random number. Here L and U are the lower and upper bounds respectively.

Step 3: Replacement / deletion strategy

One can choose the replacement strategy (e.g., replacement of the worst, the oldest, or a randomly chosen individual). A widely used combination is to replace the worst individual only if the new individual is better. In the paper, this strategy will be suggested that the deletion of the worst individual only if the new individual is better based on the fitness.

Stage 3: Local Search inspired on rough sets theory

Stage 3 starts with initial approximate of the Pareto front (provided by the proposed algorithm in stage 1 &2) which noted as NS. Also all dominated solutions are marked as DS. It is worth remarking that NS can simply be a list of an approximated Pareto solutions [30,31,32]. From the set NS we choose NNS points previously unselected. If we do not have enough unselected points, we choose the rest randomly from the set DS. Next, we choose from the set DS, NDS points previously unselected (and in the same way if we do not have enough unselected points, we complete them in a random fashion)

these points will be used to approximate the boundary between the Pareto front and the rest of the feasible set in decision variable space. We store theses points in the set Items and perform rough sets iterations.

Range Initialization: for each decision variable i, we compute and sort (from smallest and highest) the different values it takes in the set Items. Then, for each decision variable, we have a set of $range_i$ values and combining all these sets we have a non-uniform grid in decision variable space.

Compute Atoms: we compute "NNS rectangular atoms" centered in the NNS efficient points selected. To build a rectangular atom associated to a nondominated point $x^e \in Items$ we compute the following upper and lower bounds for each decision variable i:

Lower Bound i: Middle point between x_i^e and the previous value in the set $range_i$

Upper Bound i: Middle point between x_i^e and the following value in the set $range_i$

If there are no pervious or subsequent values in $range_i$, we consider the absolute lower or upper bound of variable i. This setting lets the method to explore close to the feasible set boundaries.

Generate Offspring: inside each atom we randomly generate offspring new points. Each of these points is sent to the set NS as follows. The idea is that "new solutions are only accepted in the archive if they are not ε-dominated by any other element of the current archive". If a solution is accepted, all dominated solutions are removed. Algorithm 4 (Figure 9) shows the operator for ε-approximate Pareto set

Algorithm 4: Operator for archive update

1. $INPUT: A, x$
2. $if\ \exists x' \in A\ such\ that\ x' \succ x\ then$
3. $A' = A$
4. $else$
5. $D = \{x' \in A : x \succ x'\}$
6. $A' = A \cup \{x\} \setminus D$
7. $end\ if$
8. $Output: A'$

Fig.9. Operator for archive update

6. SIMULATION RESULTS

This paper intends to implement our proposed approach for solving multiobjective engineering design problems and present an optimal design of these Problems. The results are compared with another approach which solving these design problems to show the reliability of our approach and its ability for solving this kind of problems. In the following, three engineering component design problems are discussed and extensively studied, two-bar truss design, gear train design, and air-cored solenoid design [32,138]. The parameters of proposed approach are listed in table 1

Parameter	Value
PSO iteration	200
w	0.6
c1	2.8
c2	1.3
τ	15

Table 1: parameter setting

- **Two-Bar Truss Design**

This problem is originally studied using NSGA-II [33]. The truss shown in Figure 10 has to carry a certain load without elastic failure. Thus, in addition to the objective of designing the truss for minimum volume, there are additional objectives of minimizing stresses in each of the two members AC and BC. the following two-objective optimization problem were constructed for three variables y (vertical distance between B and C in m), x1 (length of AC in m) and x2 (length of BC in meter):

$$\text{Min} \quad f_1 = x_1\sqrt{16+y^2} + x_2\sqrt{1+y^2}$$

$$\text{Min} \quad f_2 = \max\left(\sigma_{AC}, \sigma_{BC}\right)$$

$$\text{subject to} \quad \max\left(\sigma_{AC}, \sigma_{BC}\right) \le 1\left(10^5\right)$$

$$1 \le y \le 3 \quad \text{and} \quad 0 \le x_1, x_2 \le 0.01$$

The stresses are calculated as follows:

$$\sigma_{AC} = \frac{20\sqrt{16+y^2}}{yx_1} \qquad \sigma_{BC} = \frac{80\sqrt{1+y^2}}{yx_2}$$

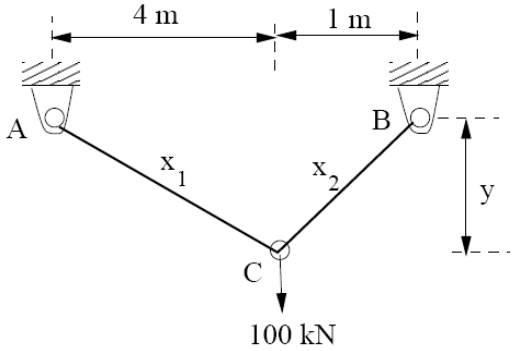


Fig. 10: The two-bar truss.

Figure 11 shows the optimized front found using the proposed method and NSGA-II. The solutions are spread by NSGA-II in the following range: (0.00407 m3, 99755 kPa) and (0.05304 m3, 8439 kPa), while by the proposed approach :(0.00406 m3, 99553.7 kPa) and (0.0554 m3, 8472.44 kPa), which indicates the power of proposed approach compared to NSGA-II. From the results shown below, we can see that our approach solutions are better than NSGA-II solutions, both in terms of closeness to the optimum front and also in their spread.

If minimization of stress is important, the proposed ap-

proach finds a solution with stress as low as 8312 kPa, where NSGA-II method has found a solution with minimum stress of 8439 kPa. On the other hand, If minimization of volume is important, the proposed approach finds a solution with volume as low as 0.004012 m3, where NSGA-II method has found a solution with minimum stress of 0.00407 m3. The following Table shows the best maximum stress and the best volume obtained by the proposed algorithm.

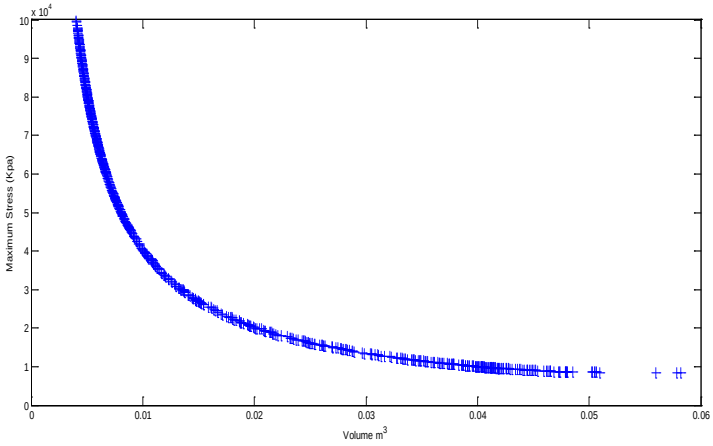


Figure 11: Optimized solutions obtained using the proposed approach (A) and NSGA (B) for the two-bar truss problem.

Table 5.1: The best maximum stress and best volume obtained by the proposed algorithm.

	f_1(Volume)	f_2(stress)
Min. Maximum stress	0.058	8312
Min. Volume	0.04012	99701

- **Gear Train Design**

A compound gear train is to be designed to achieve a specific gear ratio between the driver and driven shafts (Figure 12).

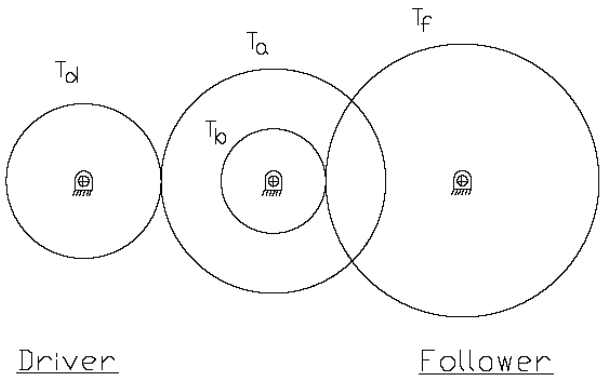


Figure. 12: A compound gear train.

The objective of the gear train design is to find the number of teeth in each of the four gears so as to minimize (i) the error between the obtained gear ratio and a required gear ratio of 1/6.931 and (ii) the maximum size of any of the four gears. Since the number of teeth must be integers, all four variables are strictly integers. By denoting the variable vector x=(x1, x2, x3, x4)= (Td, Tb, Ta, Tf), we write the two-objective optimization problem:

$$\text{Min} \qquad f_1 = \left[\frac{1}{6.931} - \frac{x_1 x_2}{x_3 x_4}\right]^2$$

$$\text{Min} \qquad f_2 = \max\left(x_1, x_2, x_3, x_4\right)$$

$$\text{subject to} \qquad 12 \le x_1, x_2, x_3, x_4 \le 60$$

Figure 13 shows the obtained optimized solutions by our approach. It can be deduced that the proposed algorithm finds comparable minimum of maximum size to NSGA-II.

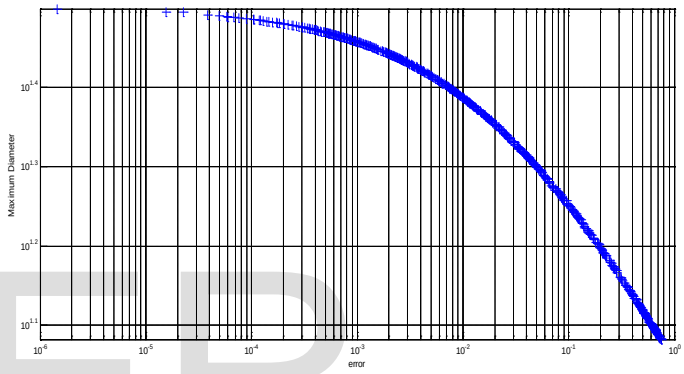


Figure 13: Optimized solutions obtained using the proposed approach (A) and NSGA (B) for the gear train design problem.

- **Shape Design of an Air-Cored Solenoid**

The multiobjective shape optimization of a coreless solenoid with rectangular cross section a × b and a mean radius c (see Figure 14) is tackled [34]. If the electric current is uniformly distributed over the cross section, it can be seen that if the number of turns (N) of the solenoid is given, then the inductance L[μH] can be approximated from:

$$L = \frac{31.49\left(a^2 N^2 / b\right)}{9 + 6\left(a/b\right) + 10\left(a/b\right)}$$

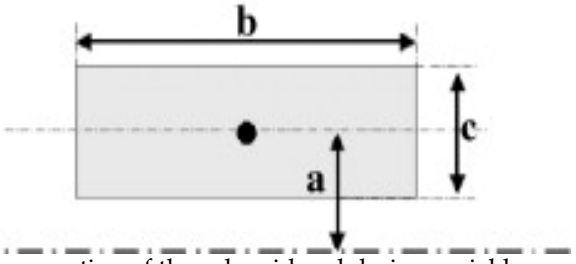


Fig. 14: Cross section of the solenoid and design variables.

This multiobjective design problem can then be formally defined in the following two objectives: maximize the inductance L(a, b, c) and minimize the volume V(a, b, c) for the given

length k1 = 10 m and k2 = 10−6 m2 of the current carrying wire. In order to simplify the analysis, two variables, a and b, are considered. Correspondingly, the computation of L and V are simplified, respectively, to

$$f_1 = \frac{31.49\left(k_1^{\,2}/4\pi^2 b\right)}{9+6\left(a/b\right)+5\left(k_1 k_2/\pi a b^2\right)}$$

$$f_2 = \frac{\pi a^2 b}{4}+\frac{k_1^{\,2}k_2^{\,2}}{4\pi a^2 b}+\frac{k_1 k_2}{2}$$

Now, the problem reads: maximize $f_1(a,b)$ and minimize $f_2(a,b)$ subject to

$$a > \sqrt{\frac{k_1 k_2}{4\pi b}}, \quad a \in [0,0.1], \quad b \in [0,0.3].$$

Despite the simplicity of formulae for both objective functions, the MOOP is not trivial and cannot be tackled analytically. The searched Pareto front using the proposed algorithm and using [34] are illustrated in Figure 15. Clearly, the proposed algorithm produces a better uniform sampling of the Pareto front for this application than that obtained by Wang et al. [34]. The results shows that a wide variety of optimal solutions have been obtained.

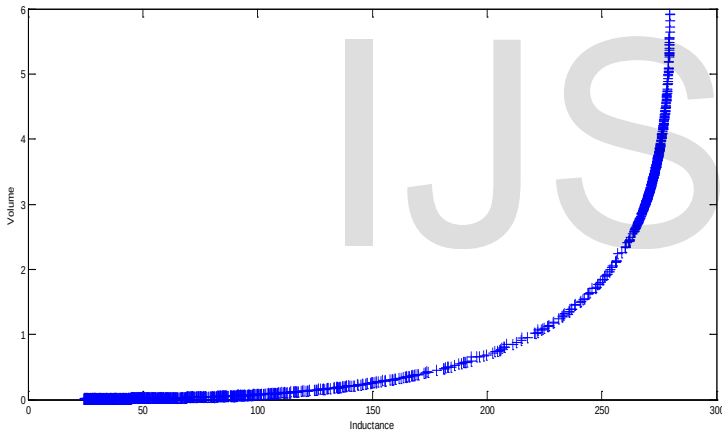


Fig. 15. Optimized solutions obtained using the proposed approach

7. Conclusion

Particle swarm optimization (PSO) is a computational method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. In this paper, we present a hybrid algorithm combining particle swarm optimization (PSO) with steady state genetic algorithm (SSGA) for solving multiobjectuve decision making (MODM) problems. The methodology combines and extends the attractive features of both PSO and SSGA, where it is based on PSO to get approximate nondominated set of the problem followed by SSGA to improve the solution quality. Then, in the second stage, rough set theory is adopted as local search engine in order to improve the spread of the solutions found so far. The results, provided by the proposed algorithm for engineering problems, are promising when compared with exiting well-known algorithms. Also, our results suggest that our algorithm is better applicable for solving real-world application problems. The main features of the proposed algorithm could be summarized as follows.

a) The proposed approach has been effectively applied to solve the MOP, with no limitation in handing higher-dimensional problems.
b) The proposed algorithm was able to find well distributed of the Pareto-optimal curve in the objective space.
c) The proposed algorithm keeps track of all the feasible solutions found during the optimization and therefore do not have any restrictions on the number of the Pareto-optimal solutions found.
d) The inclusion of local search inspired rough sets theory speeds-up the search process and also helps in obtaining a fine-grained value for the objective functions.
e) The success of our approach on most of the test problems not only provides confidence but also stress the importance of hybrid evolutionary algorithms in solving multiobjective optimization problems.
f) The reality of using the proposed approach to handle complex problems of realistic dimensions has been approved due to procedure simplicity.

For future work, we intend to test the algorithm on more complex real-world applications. Also, conduct research on the parameter adaptation of swarm optimization algorithms so as to improve the efficiency of such approaches which are very relevant for real- world scenarios.

References

[1] S. Jaganathan, S. Palaniswami, G. Maharaja Vignesh, R. Mithunraj, Applications of multi objective optimization to reactive power planning problem using ant colony algorithm, Eur. J. Sci. Res. 51 (2011) 241–253.
[2] F. Petterson, N. Chakraborti, S.B. Singh, Neural networks analysis of steel plate processing augmented by multi-objective genetic algorithms, Steel. Res. Int. 78 (2007) 890–898.
[3] R.E. Steuer, Y. Qi, M. Hirschberger, Portfolio selection in the presence of multiple criteria, in: C. Zopounidis, M. Doumpos, P.M. Pardalos (Eds.), Handbook of Financial Engineering, in: Springer Optimization and Its Applications, vol. 18, Springer, US, 2008, pp. 3–24.
[4] K.L. Kiran, S. Lakshminarayanan, Treatment planning of cancer dendritic cell therapy using multi-objective optimization, in: Proceedings of ADCHEM 2009, Istanbul, Turkey.
[5] A. Petrovski, J. McCall, B. Sudha, Multi-objective optimization of cancer chemotherapy using swarm intelligence, in: AISB 2009 Symposium on Adaptive and Emergent Behaviour and Complex Systems.
[6] R. Janssen, M. van Herwijnen, T.J. Stewart, J.C.J.H. Aerts, Multiobjective decision support for land-use planning, Environ. Plann. B 35 (2008) 740–756.
[7] Miettinen K. "Non-linear multiobjective optimization" Dordrecht: Kluwer Academic Publisher (2002).
[8] A.A.Mousa, 'Using genetic algorithm and TOPSIS technique for multiobjective transportation problem :ahybrid approach', International Journal of Computer Mathematics 87(13)(2010)3017–3029.
[9] Waiel. F. Abd El-Wahed, A.A.Mousa , M. A. El-Shorbagy, Integrating particle swarm optimization with genetic algorithms for solving nonlinear optimization problems, Journal of Computational and Applied Mathematics 235 (2011) 1446–1453.(Top 25 Hottest Articles, Oct. to Dec. 2010, Jan. to Mar. 2011, April to June 2011)

[10] A.A.Mousa, M.A.El-Shorbagy, WaielF.AbdEl-Wahed, Local search based hybrid particle swarm optimization for multiobjectuve optimization ,International Journal of Swarmand Evolutionary Computation3(2012)1–14.

[11] A.A.Mousa,I.M.El_Desoky,Stability of Pareto optimal allocation of land reclamation by multistage decision- based multipheromone ant colony optimization, Swarm and Evolutionary Computation(2013), http://dx.doi.org/10.1016/j.swevo.2013.06.003i

[12] Renata E. Onety, Roberto Tadei, Oriane M. Neto, Ricardo H.C. Takahashi , Multiobjective optimization of MPLS-IP networks with a variable neighborhood genetic algorithm , Applied Soft Computing, Volume 13, Issue 11, November 2013, Pages 4403-4412

[13] James N. Richardson, Guy Nordenson, Rebecca Laberenne, Rajan Filomeno Coelho, Sigrid Adriaenssens , Flexible optimum design of a bracing system for façade design using multiobjective Genetic Algorithms , Automation in Construction, Volume 32, July 2013, Pages 80-87

[14] H. Ahonen, A.G. de Alvarenga, A.R.S. Amaral , Simulated annealing and tabu search approaches for the Corridor Allocation Problem ,European Journal of Operational Research, Volume 232, Issue 1, 1 January 2014, Pages 221-233

[15] Chen-Fu Chen, Muh-Cherng Wu, Keng-Han Lin , Effect of solution representations on Tabu search in scheduling applications ,Computers & Operations Research, Volume 40, Issue 12, December 2013, Pages 2817-2825.

[16] Celal Özkale, Alpaslan Fığlalı , Evaluation of the multiobjective ant colony algorithm performances on biobjective quadratic assignment problems ,
Applied Mathematical Modelling, Volume 37, Issues 14–15, 1 August 2013, Pages 7822-7838

[17] Kun Fan, Weijia You, Yuanyuan Li , An effective modified binary particle swarm optimization (mBPSO) algorithm for multi-objective resource allocation problem (MORAP) ,Applied Mathematics and Computation, Volume 221, 15 September 2013, Pages 257-267

[18] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of the 1995 IEEE International Conference on Neural Networks, vol. 4, 1995, pp.1942–1948.

[19] X. Jie, X. Deyun, New metropolis coefficients of particle swarm optimization,in:IEEE, 2008.

[20] Holland, Adaptation in natural and artificial systems, The MIT Press; Reprint edition 1992 (originally published in 1975).

[21] [21] G. Dozier, H. Cunningham, W. Britt, F. Zhang, Distributed constraint satisfaction, restricted recombination, and hybrid genetic search, in: The Proceedings of the 2004 Genetic and Evolutionary Computation Conference (GECCO-2004), LNCS, Springer, Seattle, WA, 2004, pp. 1078–1087.

[22] E. Unsal, J.H. Dane, G.V. Dozier, A genetic algorithm for predicting pore geometry based on air permeability measurements, The Vadose Zone Journal 4 (2005) 389–397. Soil Science Society of America, Madison, WI.

[23] G. Dozier, A. Homaifar, E. Tunstel, D. Battle, An introduction to evolutionary computation, in: Intelligent Control Systems Using Soft Computing Methodologies, A. Zilouchian, M. Jamshidi (Eds.), CRC press, pp. 365–380, (Chapter 17).

[24] J.N. Siddal, Optimal Engineering Design: Principles and Applications, Mercell Dekker Inc., New York, NY, 1982.

[25] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, Introduction to algorithms, Second Edition. MIT Press and McGraw-Hill, 2001. ISBN 0-262-03293-7. Section 29.3: The simplex algorithm, pp. 790–804.

[26] Osman M.S., M.A.Abo-Sinna, and A.A. Mousa " IT-CEMOP: An Iterative Co-evolutionary Algorithm for Multiobjective Optimization Problem with Nonlinear Constraints" Journal of Applied Mathematics & Computation (AMC) 183, pp373-389, (2006).

[27] Osman M.S., M.A.Abo-Sinna, and A.A. Mousa " IT-CEMOP: An Iterative Co-evolutionary Algorithm for Multiobjective Optimization Problem with Nonlinear Constraints" Journal of Applied Mathematics & Computation (AMC) 183, pp373-389, (2006).

[28] Hernández Díaz, A. G., L. V. Santana-Quintero, C. A. Coello Coello, R. Caballero and J. Molina 2008) "Improving Multi-Objective Evolutionary Algorithms by using Rough Sets," in A. Ligeza, S. Reich, R. Schaefer and C. Cotta (eds.), Knowledge-Driven Computing: Knowledge Engineering and Intelligent Computations, Studies in Computational Intelligence, vol. 102, pp. 81:98.

[29] A.A.Mousa , M. A. El-Shorbagy , Waiel. F. Abd El-Wahed, Local search based hybrid particle swarm optimization for multiobjective optimization, International journal of Swarm and evolutionary computation, 3(2012),1:14. (Top 25 Hottest Articles, July to September 2012, October to December 2012, January to March 2013) promotion of assistant professor

[30] Ahmed A. EL- Sawy, Mohamed A. Hussein, EL-Sayed M. Zaki, A. A. Mousa, Local Search-Inspired Rough Sets for Improving Multiobjective Evolutionary Algorithm, Applied Mathematics, 2014, 5, 1993-2007.

[31] Mohamed A. Hussein, Ahmed A. EL-Sawy, EL-Sayed M. Zaki, A. A. Mousa, Genetic Algorithm and Rough Sets Based Hybrid Approach for Economic Environmental Dispatch of Power Systems, British Journal of Mathematics & Computer Science, ISSN: 2231-0851,Vol.: 4, Issue.: 20 (16-31 October)

[32] Ahmed A. El- sawy, Mohamed A. Hussein1, El-Sayed M. Zaki1, A. A. Mousa, Rough sests-inspired evolutionary algorithm for engineering multiobjective optimization problems, journal of global research in mathematical archives volume 1, no. 12, december 2013.

[33] Deb K., Pratap A., Moitra S.; (2000), "Mechanical Component Design for multi-objective using Elitist non-dominated sorting GA", KanGAL Report No. 200002.

[34] Wang Y., Yang S., Ni G., Ho S.L., and Liu Z.J.; (2004), "An emigration genetic algorithm and its application to multiobjective optimal designs of electromagnetic devices", IEEE Transactions On Magnetics 40(2): 1240-1243.