Swarm Intelligence based Soft Computing Techniques for the Solutions to Multiobjective Optimization Problems

Hifza Afaq[1] and Sanjay Saini[2]
Department of Physics & Computer Science, Dayalbagh Educational Institute
Agra 282005, India

Abstract
The multi objective optimization problems can be found in various fields such as finance, automobile design, aircraft design, path optimization etc. This paper reviews some of the existing literature on multi objective optimization problems and some of the existing Swarm Intelligence (SI) based techniques to solve these problems. The objective of this paper is to provide a starting point to the multi objective optimization problems to the reader.
Keywords: *Multiobjective optimization problem, Evolutionary Algorithm, Particle Swarm Optimization, Ant Colony Optimization.*

1. Introduction

Most real life problems are multi objective in nature. These objectives are conflicting (preventing simultaneous optimization) in general. It means that one objective is optimized at the cost of other objective. The multi objective optimization problems are difficult but realistic, because of their broad applicability, optimization problems have been studied by researchers with various backgrounds. This gives rise to a variety of strategies for solving such problems. There exists a vast literature on the methods to deal with multi objective optimization problems. It should be noted that every approach has its pros and cons, and no single best option is available to the solution seeker in the general case. There are two general approaches to multiple objective optimization problems. (i) Classical Approach (preference-based multi-objective optimization) (ii) Ideal Approach (Pareto optimal approach).

A simple method to solve a multi objective optimization problem would be to form a composite objective function as the weighted sum of the objectives, where a weight for an objective is proportional to the preference vector assigned to that particular objective. This method of scalarizing an objective vector into a single composite objective function converts the multi objective optimization problem into a single objective optimization problem. When such a composite objective function is optimized, in most cases it is possible to obtain one particular trade off solution. This procedure of handling multi objective optimization problems is simple but relatively subjective. This procedure is preference based multi objective optimization [19].

The second approach is to determine an entire set of solutions that are non-dominated with respect to each other. This set is known as Pareto optimal set. While moving from one Pareto solution to another, there is always a certain amount of sacrifice in one or more objectives to achieve a certain amount of gain in the other(s). Pareto optimal solution sets are often preferred to single solutions because they can be practical when considering real-life problems. The size of the Pareto set usually increases with the increase in the number of objectives [55].

It is important to note that the result obtained by preference-based strategy largely depends on the relative preference vector used in forming the composite function. A change in this preference vector will result in a (hopefully) different trade-off solution. Preference vector need not result in a trade-off optimal solution to all problems. Also, it is important to note that it is highly subjective and not so easy to find a relative preference vector. On the other hand, the ideal multi objective optimization procedure is less subjective. The main task in this approach is to find as much different trade-off solutions as possible. Once a well distributed set of trade off solutions is found then higher level information related to a problem is required to choose one solution from a set of already obtained set of trade off solutions. Thus, the fundamental difference in using the problem information in the two approaches is that, relative preference vector needs to be supplied without any knowledge of the possible consequences in the classical approach; where in the ideal approach, the problem information is used to choose one solution from the obtained set of trade-off solutions. The ideal approach, in this matter is more practical and less subjective [19].

2. Multi Objective Optimization Problems

The multi-objective optimization problem needs to pay attention to a number of conflicting objectives simultaneously. The goal in multi-objective optimization is to obtain a finite number of Pareto optimal solutions, instead of a single optimal solution. For example, the multi objective problem that simultaneously minimizes (without loss of generality)[1] objectives can be described as follows:

Minimize $(f_1(x), f_2(x), ..., f_i(x))$, subject to $x \in S$

Where, S denotes the feasible solution space. In order to solve a multi objective problem, some important concerns must be handled carefully. One of these concerns is how to optimize every objective functions at the same time. An important concept tied to multi objective optimization is *domination*. Deb, K. [19] defines domination by Definition 1.

Definition 1: A solution $x^{(1)}$ is said to dominate the other solution $x^{(2)}$, if both conditions 1 and 2 are true:

I. The solution $x^{(1)}$ is no worse than $x^{(2)}$ in all objectives, or $f_j(x^{(1)}) \ntriangleright f_j(x^{(2)})$ for all j=1,2,….,M.
II. The solution $x^{(1)}$ is strictly better than $x^{(2)}$ in at least one objective, or $f_j(x^{(1)}) \lhd f_j(x^{(2)})$ for at least one $j \in \{1,2,….,M\}$ [19].

A non-dominated set of solution is defined by Definition 2.

Definition 2: Among a set of solutions P, the non-dominated set of solutions P' are those that are not dominated by any member of the set P [19].

The set of non-dominated solutions is also known as the Pareto optimal set of solution. This set has all the possible good solutions for the solution seeker.

Multi objective optimization problems has been tackled through various approaches of which evolutionary computation, PSO and ACO are reviewed in the current text. PSO has been known to perform well for many static problems. However, many real-world problems are dynamic and it has been argued that EAs are potentially well-suited in the dynamic problems [73], [77], [83]. However, PSO techniques also hold promises for dynamic problems [5], [12]. Another approach which is gaining popularity for multi objective Optimization is Ant Colony Optimization technique [7], [21], [35]. Various fields where ACO has been applied and known to be working well are classical problems such as assignment problems, sequencing [35] and scheduling problems [7], graph coloring [39], discrete and continuous function optimization and path optimization problems [3], [23], [24], [31], [32], [33], [36], [72] etc. More recent applications include cell placement problems arising in circuit design, the design of communication networks [11], [22], problems in bioinformatics [62] etc.

[1] A maximization problem can be changed to a minimization problem by using duality principle.

3. Solution to Multiobjective Optimization Problems

3.1. Evolutionary Algorithm

Evolutionary algorithms (EA's) are often well-suited for optimization problems involving several, often conflicting objectives, i.e., multi objective optimization problems. EAs have been used in various fields such as an auto pilot design [8], vehicle routing problem, travelling salesman problem [30]. Brockhoff reviewed theoretical aspects of evolutionary multiobjective optimization in [9]. Potwin [71] reviewed the literature on evolutionary algorithm for vehicle routing problem which is a multi objective optimization problem. Abraham and Jain state that real world applications have several multiple conflicting objectives generally. They defined fundamental concepts of multiobjective optimization emphasizing the motivation and advantages of using evolutionary algorithms in [1].

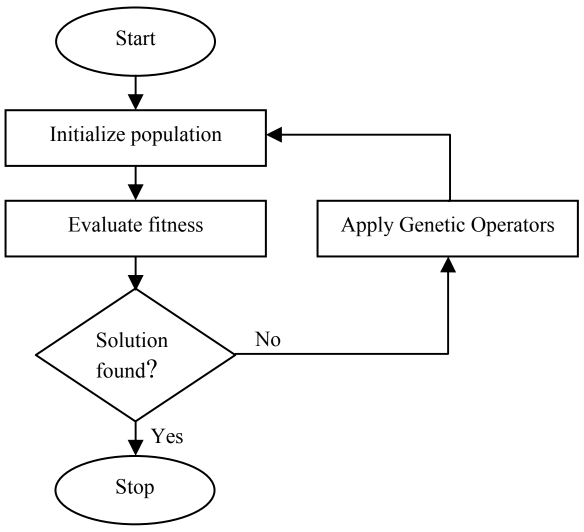

Fig. 1 Flow chart for EA.

Fieldsend [29] defined multi objective problems as being typically complex, with both a large number of parameters to be adjusted and several objectives to be optimized. Multi objective evolutionary algorithms (MOEAs) are a popular approach to confronting multi objective optimization problems by using evolutionary search techniques. The use of EAs as a tool of preference is due to such problems. EAs, which can maintain a population of solutions, are in addition able to explore several parts of the Pareto front simultaneously. Flow chart in Fig. 1 displays basic EA functionality.

Srinivas, N. and Deb, K. applied genetic algorithm in [77] to find the Pareto optimal set. Deb, K. describes the problem features that may cause a multi-objective genetic algorithm difficulty in converging to the true Pareto-optimal front in [18]. He constructed multiobjective test problems having features specific to multiobjective optimization enabling researchers to test their algorithms for specific aspects of multi-objective optimization. Deb, K. and Agrawal, S. [20] investigated the performance of simple tripartite GAs on a number of simple to complex test problems from a practical standpoint. For solving simple problems mutation operator plays an important role, although crossover operator can also be applied to solve these problems. When these two operators are applied alone they have two different working zones for population size. For complex problems involving massive multimodality and deception, crossover operator is the key search operator and performs reliably with an adequate population size. Based on this study, they recommended using of the crossover operator with an adequate population size.

Corne and Knowles [17] introduced Pareto Envelop based Selection Algorithm (PESA). Zitzler, Deb and Thiele [84] compared four multi objective EA's quantitatively. Also, they introduced an evolutionary approach to multi-criteria optimization, the Strength Pareto EA (SPEA) that combines several features of previous multi objective EA's in a unique manner to produce better results. Binh and Korn [4] presented a Multiobjective evolution strategy (MOBES) for solving multiobjective optimization problems subject to linear and non-linear constraints. The algorithm proposed by them maintains a set of feasible Pareto optimal solution in every generation.

Some researchers modified genetic algorithm to get better results for the problems to be tested. In this series Jian [48] employed a proposed genetic particle swarm optimization method (MGPSO) to solve the capacitated vehicle routing problems. He incorporated PSO with the genetic reproduction mechanisms (crossover and mutation). This algorithm employs an integer encoding and decoding representation. This method has been implemented to five well-known CVRP benchmarks. Prins developed a hybrid GA that is relatively simple but effective to solve the vehicle routing problem.

Jin, Okabe and Sendhoff [49] brought an idea that systematically changes the weights during evolution that leads the population to the Pareto front. They investigated two methods; one method is to assign a uniformly distributed random weight to each individual in the population in each generation. The other method is to change the weight periodically with the process of the evolution. They found in both cases that the population is able to approach the Pareto front, although it does not keep all the found Pareto solutions in the population. Zhang and Rockett [82] compared three evolutionary techniques (i) The Strength Pareto Evolutionary Algorithm (SPEA2), (ii) The Non-dominated Sorting Genetic Algorithm (NSGA-II) and (iii) The Pareto Converging Genetic Algorithm (PCGA).

3.2. Particle swarm techniques

a. Particle Swarm Optimization Algorithm

Heppner and Grenander, in 1990 [38] proposed that the small birds fly in coordination and display strong synchronization in turning, initiation of flight, and landing without any clear leader. They proposed that the synchronization of movement may be a by product of "rules" for movement followed by each bird in the flock. Simulation of a bird swarm was used to develop a particle swarm optimization (PSO) concept [51]. PSO was basically developed through simulation of bird flocking in two dimensional spaces. Searching procedures by PSO can be described as follows:

i. Each particle evaluates the function to maximize at each point it visits in spaces.
ii. Each particle remembers the best value it has found so far (pbest) and its co-ordinates.
iii. Each particle know the globally best position that one member of the flock had found, and its value global best (gbest).
iv. Using the co-ordinates of pbest and gbest, each particle calculates its new velocity as in Eq. (1) **Error! Reference source not found.**and the position of each particle is updated by Eq. (2).

$$\vec{v}_i(t+1)=\vec{v}_i(t)+\phi_1\otimes(\vec{p}_i-\vec{x}_i(t))+\phi_2\otimes(\vec{p}_g-\vec{x}_i(t)) \quad (1)$$

$$\vec{x}_i(t+1)=\vec{x}_i(t)+\vec{v}_i(t+1) \quad (2)$$

Here, $\phi_1 = c_1R_1 \text{ and } \phi_2 = c_2R_2;\ 0 < R_1 < 1;\ 0 < R_2 < 1.$

The basic particle swarm optimization algorithm is as follows:

```
i.    Randomly generate an initial swarm
ii.   Repeat for each particle i do
          a.  If $f(x_i) > f(p_i)$ then $p_i \leftarrow x_i$
          b.  $p_g = \max(p_{neighbours})$
          c.  update velocity
          d.  update position
          e.  end for
iii.  until termination criteria is met
```

Kennedy and Eberhart, in a later work [50] introduced the Discrete Binary version of the PSO algorithm. In this algorithm the velocity of particle is described by the Hamming distance between the particle at time t and t+l. they defined v_{id} for discrete space as the probability of bit x_{id} taking the value 1. So, the **Error! Reference source not found.** remained unaltered in the discrete version except that now p_{id} and x_{id} belongs to {0, 1} and are discrete. Secondly, V_{id}, since it is a probability, is constrained to the interval {0.0, 1.0}. The resulting change in position then is defined by the following rule:

```
If (rand() < $S(v_{id})$)
    then $x_{id} = 1$
else $x_{id} = 0$
```

where, the function S(v) is a sigmoid limiting transformation and rand() is a quasi-random number selected from a uniform distribution in [0.0, 1.0]. From the results obtained they [50] concluded that binary particle swarm implementation is capable of solving various problems very rapidly. Also, they found that the discrete version of the algorithm extends the capabilities of the continuous valued one and is able to optimize any function, continuous or discrete. Higashi and Iba [40] presented a particle swarm optimization with Gaussian mutation that combines the idea of the particle swarm with concepts from evolutionary algorithms. They tested and compared this model with the standard PSO and standard GA. The experiments conducted on unimodal and multimodal functions gave the better results when PSO with Gaussian mutation is used in comparison with standard GA and PSO. Khanesar, Teshnehlab, and Shoorehdeli [53] introduced a novel binary particle swarm optimization.

b. PSO Parameters

Many researchers and scientists focused their attention to different parameters used in PSO to produce better results. Pederson, M. E. H. explains in *Good Parameters for Particle Swarm Optimization* [69] that particle swarm optimization has a number of parameters that determine its behavior and efficacy in optimizing a given problem.

Shi and Eberhart [75] in late 90's pointed out that for different problems, there should be different balances between the local search ability (pbest) and global search ability (gbest). Considering this they introduced a parameter called inertia weight (w), in **Error! Reference source not found.** as given in **Error! Reference source not found.** where, **Error! Reference source not found.** remained unchanged.

$$\vec{v}_i(t+1) = w*\vec{v}_i(t) + \phi_1 \otimes (\vec{p}_i - \vec{x}_i(t)) + \phi_2 \otimes (\vec{p}_g - \vec{x}_i(t)) \quad (3)$$

Inertia weight introduced by [75] could be a positive constant or even a positive linear or nonlinear function of time. They performed various experiments to test the Schaffer's f6 function, a benchmark function, and concluded that the PSO with the inertia weight in the range [0.9, 1.2] on average will have a better performance. They also introduced the concept of time decreasing inertia weight to improve the PSO performance. Also, Shi and Eberhart [75] introduced the idea to build a fuzzy system to tune the inertia weight on line. Clerc and Kennedy [13] pointed out that the traditional versions of the algorithm have had some undesirable dynamical properties especially the particle's velocities needed to be limited in order to control their trajectories. In their study they [13] also analyzed the particle's trajectory as it moves in discrete time, then progresses to the view of it in continuous time. On the basis of this analysis they presented a generalized model of the algorithm, containing a set of coefficients to control the system's convergence tendencies. They [13] introduced the constriction factor χ that causes the convergence of the individual trajectory in the search space, and whose value is typically approximately 0.729.

$$\vec{v}_i(t+1) = \chi[\vec{v}_i(t) + U(0,\varphi_1) \otimes (\vec{p}_i - \vec{x}_i(t)) + U(0,\varphi_2) \otimes (\vec{p}_g - \vec{x}_i(t))] \quad (4)$$

Here, $\varphi = \varphi_1 + \varphi_2$. Clerc's analysis worked out using a condensed form of the formula given by Eq. (5).

$$\vec{v}_i(t+1) = \chi[\vec{v}_i(t) + \varphi.(\vec{p}_m - \vec{x}_i(t))] \quad (5)$$

In this model $\vec{p}_m = \dfrac{\varphi_1.\vec{p}_i + \varphi_2.\vec{p}_g}{\varphi_1 + \varphi_2}$. Mendes, Kennedy and Neves [59] proposed an alternate form of calculating $\vec{p}_m$ given by Eq. **Error! Reference source not found.**).

$$\vec{p}_m = \frac{\sum_{k \in N} w(k)\vec{\varphi}_k \otimes p_k}{\sum_{k \in N} w(k)\vec{\varphi}_k}; \quad \vec{\varphi}_k = \vec{u}\left[0, \frac{\varphi_{MAX}}{|N|}\right] \forall k \in N \quad (6)$$

Where N is the set of neighbors of the particle and $\vec{p}_k$ is the best position found by individual k [59]. In this model all the neighbors contribute to the velocity adjustment, the authors called it the fully informed PSO.

Adriansyah and Amin [2] proposed a variation of PSO model where inertia weight is sigmoid decreasing; they called it Sigmoid Decreasing Inertia Weight. Parsopolus and Vrahatis [67] conducted various experiments on the benchmark problems to yield useful conclusions regarding the effect of the parameters on the algorithm's performance.

c. Neighborhood topologies

Kennedy and Mendes [52] describes the particle swarm algorithm as a population of vectors whose trajectories oscillate around a region which is defined by each individual's previous best success and the success of some other particle. Various methods have been used to identify "some other particle" to influence the individual. In the original PSO algorithm the particle were influenced by its own position and the position of the particle that has found best value for the function. They used gbest and pbest topologies in combination to improve the performance of basic PSO and tested the algorithm on six standard benchmark functions keeping two factors in focus (i) Number of neighbors and (ii) Amount of clustering. The results suggested that the Von-Neumann topology (with and without self) did well where star and gbest (without self) were the worst [52]. Fig. 2 depicts different topological structures. The traditional particle swarm topology gbest as described by Mendes, Kennedy and Neves [59] treats the entire population as the individual's neighborhood; all particles are directly connected to the best solution in the population. Whereas, the neighborhood of individual in the pbest type sociometry is comprised of the adjacent members of the population array. It is the slowest and most indirect communication pattern. According to Kennedy and Mendes [52] it has been thought that the gbest type converges quickly on problem solutions but has a weakness for becoming trapped in local optima, while pbest populations are able to flow around local optima, as subpopulations explore different regions. They concluded that when distances between nodes are too short, and communication passes too quickly, the population converges very quickly to the best solution found in the early iterations. In this case the population will fail to explore outside of locally optimal regions. On the other hand, inhibiting communication too much results in inefficient allocation of trials, as individual particles wander clueless through the search space. Out of the neighborhood configurations they tested they recommended von Neumann sociometry that performed better than the standard ones on a suite of standard test problems. Mendes, R., Kennedy, J. and Neves, J. [60] introduced new ways that an individual particle can be influenced by its neighbors. Different topological structures are shown in

Fig. 2.

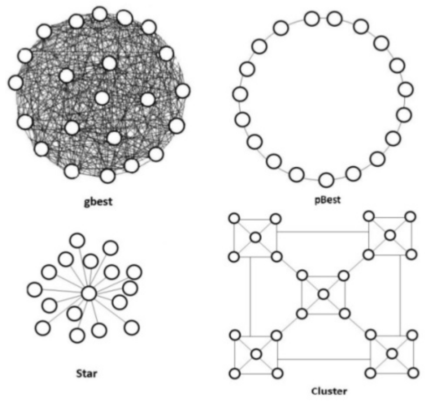

Fig. 2 Neighborhood Topologies.

Mendes, Kennedy and Neves designed and studied several sociometries [60] i.e., gbest, pbest, four clusters, Von-Neumann and pyramid (See

Fig.). Von-Neumann is a square lattice whose extremities connect as a torus; pyramid is a three dimensional wireframe triangle as described by Mendes, Kennedy and Neves [60]. They used two measure of performance (a) Best functional result found after the fixed number of iterations. (b) Number of iterations needed for the algorithm to meet the criteria and employed five standard test functions. They found that the individual is not only influenced by the best individual in the neighborhood but also that the best results do not take into consideration the experience of the individual. They concluded that the von-Neumann sociometry is a very good performer but the reason remained unanswered. Wang and Xiang [78] proposed a dynamically changing ring topology, in which particles have unidirectional connection with respect to their personal best fitness. Meanwhile, two strategies, namely the "Learn From Far and Better Ones" strategy and the "Centroid of Mass" strategy are used to enable certain particle to communicate with its neighbors. Experimental results on six benchmarks functions validate the effectiveness of the proposed algorithm [78]. Ghosh, Kundu, Kaushik, Das, Abraham, Panigrahi and Snasel [34] took first step towards the probabilistic analysis of the pbest PSO with variable random neighborhood topology by addressing issues like inter-particle interaction and probabilities of selection based on particle ranks. Hamdan [37] hybridized star, ring and Von-Neumann topologies to get better results. According to this algorithm the particle will compute its velocity according to all the topologies separately and will update its velocity according to the best suited topology. So, the particle will choose the best for itself. According to the author this algorithm performed better on nine standard test functions and has faster convergence speed in comparison to other strategies that exists.

d. Dynamic Environments

Many real world applications are dynamic that requires an optimization algorithm to have a property to continuously track the changing position of the goal. Carlisle and Dozier [10] proposed a method for adapting the particle swarm optimizer for dynamic environments. The process consists of causing each particle to reset its record of its best position as the environment changes, to avoid making direction and velocity decisions on the basis of outdated information. Two methods for initiating this process are examined: (i)Periodic resetting, based on the iteration count, and (ii)Triggered resetting, based on the magnitude of change in environment. The preliminary results suggest that these two modifications allow PSO to search in both static and dynamic environments. Hu and Eberhart [42] modified PSO to track changes in a dynamic system automatically. They tested different environment detection and response techniques on two benchmark functions. Also, they introduced re-randomization to respond to the dynamic changes. In another study they introduced an adaptive PSO [42] which automatically tracks various changes in a dynamic system. Blackwell and Branke [6] explored different variants of PSO that works well in the dynamic environments. Their main idea is to split the population of particles into a set of interacting swarms. These swarms interact locally by an exclusion parameter and globally through an anti convergence operator. The multi-swarm algorithm evaluated on multimodal dynamic moving peaks benchmark functions gives the results showing that the multi-swarm optimizer significantly outperforms previous approaches. Janson and Middendorf [46] compared dynamic variants of standard PSO and Hierarchical PSO (H-PSO) on different dynamic benchmark functions. Also, they proposed a hierarchical PSO, called Partitioned H-PSO (PH-PSO). In PH-PSO algorithm the hierarchy is partitioned into several sub-swarms for a limited number of generations after a change occurred. The test results show that H-PSO performs significantly better than PSO on all test functions and that the PH-PSO algorithms often perform best on multimodal functions where changes are not too severe.

In a later study on noisy and dynamic environment Janson and Middeendorf [47] studied Particle Swarm Optimization method for dynamic and noisy function optimization. PH-PSO maintains a hierarchy of particles that is partitioned into several sub-swarms for a limited number of generations after a change of the environment occurred. A standard method for metaheuristics to cope with noise is to use function re-evaluations. They proposed a method to reduce the number of necessary re-evaluations which uses the hierarchy to find a subset of particles for which re-evaluations are particularly important. They also presented a method to detect changes of the optimization function in the presence of noise.

e. Particle Swarm Optimization Algorithm for Multiobjective Optimization Problems

Particle Swarm Optimization Techniques have applications in a variety of area such as Clustering Analysis [12], Multiobjective Optimization Problems [14], [43], [44], [45], [63], [65], [81], Neural Network Training [58] etc. In this section we will focus our attention to the application of PSO to Multi-objective Optimization Problems. Parsopolus and Vrahatis presented first study of the Particle Swarm Optimization method in Multiobjective Optimization Problems in [68]. They studied the ability of PSO to detect Pareto Optimal points and capture the shape of the Pareto Front. They considered Weighted Aggregation technique with fixed or adaptive weights. Also, they adapted critical aspects of the VEGA approach for Multiobjective Optimization using

Genetic Algorithms to the PSO framework in order to develop a multi-swarm PSO that can cope effectively with multi objective problems.

Hu, Eberhart and Shi [43] presented a dynamic neighborhood particle swarm optimization algorithm for multiobjective optimization problems. They introduced an extended memory to store global Pareto optimal solutions to reduce computation time. Li [57] introduced a modified PSO, Non-dominated Sorting Particle Swarm Optimizer (NSPSO), for better multiobjective optimization. NSPSO extends the basic form of PSO by making a better use of particle's personal bests and offspring for more effective non-domination comparisons. Instead of a single comparison between a particle's personal best and its offspring, NSPSO compares all particle's personal bests and their offspring in the entire population. This proves to be effective in providing an appropriate selection pressure to propel the swarm population towards the Pareto-optimal front. By using the non-dominated sorting concept and two parameter-free niching methods, results and comparison with NSGA II show that NSPSO is highly competitive with existing evolutionary and PSO multiobjective algorithms.

Coello, Pulido and Lechuga [15] introduced the concept of Pareto dominance into particle swarm optimization in order to allow it to handle problems with several objective functions. This modified algorithm uses a secondary repository of particles that is later used by other particles to guide their own flight. They also incorporated a special mutation operator that enriches the exploratory capabilities of the algorithm. Persopolus, Tasoulis and Vrahatis [65] studied a parallel version of the Vector Evaluated Particle Swarm Optimization (VEPSO) method for multiobjective problems and performed experiments on well known and widely used test problems. The obtained results yields the superiority of VEPSO when compared with the corresponding results of the Vector Evaluated Genetic Algorithm approach. Hwang, Koo and Lee [45] proposed a Homogeneous Particle Swarm Optimizer (HPSO) for multi-objective optimization problems. They proposed one global repository concept for choosing pBest and gBest, this means that each particle has lost its own identity and treated simply as a member of social group. They tested various kinds of the multiobjective optimization problem that illustrated the successful result in finding a Pareto optimal set. Parsopolus and Vrahatis [66] described the state of art literature on multiobjective PSO algorithms. They distinguished two fundamental categories of algorithms, (a) approaches that exploit each objective function separately and (b) Pareto-based schemes. The exposition of the methods for each category is based on chronological ordering. According to Xiao-hua, Hong-yun and Li-cheng [79] it is very important to find a sufficient number of uniformly distributed and representative Pareto optimal solutions for multiobjective optimization problems. They constructed a model for particle swarm optimization, and proposed an intelligent particle swarm optimization (IPSO) for MO problems, on the basis of AER (agent-environment-rules) model, in which competition operator and clonal selection operator are designed to provide an appropriate selection pressure to propel the swarm population towards the Pareto-optimal front. The quantitative and qualitative comparisons indicate that the proposed approach is highly competitive and that can be considered as a viable alternative to solve multiobjective objective problems. Koppen and Veenhuis [56] introduced a approach to multi-objective particle swarm optimization. The approach is based on the Fuzzy-Pareto-Dominance (FPD) relation. FPD is a generic ranking scheme, where ranking values are mapped to element vectors of a set. These ranking values are directly computed from the element vectors of the set and can be used to perform rank operations (e.g. selecting the "largest") with the vectors within the given set. FPD can be seen as a paradigm or metaheuristic to formally expand single-objective optimization algorithms to multi-objective optimization algorithms, as long as such vector sets can be defined. This was already shown for the Standard Genetic Algorithm. They applied this concept to PSO, where a swarm of particles is maintained. The resulting PSO algorithm studied on a fundamental optimization problem (Pareto-Box-Problem) is shown to handle the case of a larger number of objectives, and shows properties similar to single-objective PSO. Kodru, Das and Welch described a PSO Nelder Mead Simplex hybrid multiobjective optimization algorithm [54] based on a numerical metric called ε-fuzzy dominance. In this approach, k-means algorithm is applied to divide the population into smaller sized clusters, while the position and velocity of each particle is updated using PSO, in every iteration. The Nelder Mead simplex algorithm is used separately within each cluster for added local search. The algorithm they proposed is shown to perform better than MOPSO on several test problems.

Zhang and Xue [81] proposed a dynamic sub-swarms multi-objective particle swarm optimization algorithm (DSMOPSO). Based on solution distribution of multi-objective optimization problems, it separates particles into multi subs warms, each of which adopts an improved clustering archiving technique, and operates PSO in a comparably independent way. Clustering eventually enhances the distribution quality of solutions. The selection of the closest particle to the gbest from archiving set and the developed pbest select mechanism increase the choice pressure. In the meantime, the dynamic set particle inertial weight being relevant to the number of dominating particles, effectively keeps the balance between the global search in the early stage and the local search in the later stage. Experiments show that this strategy yields good convergence and strong capacity to conserve the distribution of solutions, especially for the problems with non-continuous Pareto-optimal front. Elhossini and Areibi [27] proposed an efficient particle swarm optimization technique based on the strength Pareto approach to deal with multi objective optimization problem. The proposed modified particle swarm algorithm is used to build three hybrid EA-PSO algorithms to solve different multi-objective optimization problems. They tested these algorithm using seven benchmarks from the literature and compared the results to the strength Pareto evolutionary algorithm (SPEA2) and a competitive multi-objective PSO using several metrics. The proposed algorithm shows a slower convergence, compared to the other algorithms, but requires less CPU time. Combining PSO and evolutionary algorithms leads to superior hybrid algorithms that outperform SPEA2, the competitive multi-objective PSO, and the proposed strength Pareto PSO based on different metrics.

Minnebo in [61] explored various strategies for handling boundary constraints and guide selection. Additionally he explored the method to combine independent runs of the algorithm in order to obtain solutions of higher quality. Mostaghim and Teich [63] proposed a method using multi-objective particle swarm optimization to cover the Pareto optimal front. The method works in two phases. In phase 1 the goal is to obtain a good approximation of the Pareto-front. In a second run sub swarms are generated to cover the Pareto front. Omkar, Khandelwal, Ananth, Naik, Narayana and Gopalakrishnan [64] present a generic method for multi-objective design optimization of laminated composite components using a novel multi-objective optimization algorithm developed on the basis of the Quantum behaved Particle Swarm Optimization (QPSO) paradigm. QPSO is a co-variant of the popular Particle Swarm Optimization and has been developed and implemented successfully for the multi-objective design optimization of composites. The problem is formulated with multiple objectives of minimizing weight and the total cost of the composite component to achieve a specified strength. Fan and Chang [28] proposed a technique to use particle swarm optimization algorithms to solve multi-objective optimization problems. Their algorithm is based on the concept of Pareto dominance, as well as a state-of-the-art 'parallel' computing technique that intends to improve algorithmic effectiveness and efficiency simultaneously. The results indicate that the algorithm proposed is extremely competitive when compared with other MOEAs, being able to accurately, reliably and robustly approximate the true Pareto front in almost every tested case. Sierra and Coello [76] proposed a Multi-Objective Particle Swarm Optimizer, which is based on Pareto dominance and the use of a crowding factor to filter out the list of available leaders. They also proposed the use of different mutation operators which act on different subdivisions of the swarm. Also, the approach incorporates the e-dominance concept to fix the size of the set of final solutions produced by the algorithm. The results indicate that the proposed approach is highly competitive, being able to approximate the front even in cases where all the other PSO-based approaches fail.

f. Hybrid Particle Swarm Algorithm

Researchers and scientists hybridized PSO with various different existing evolutionary and fuzzy concepts. Existing literature on these include [41] in which Hongbo, Abraham and Zhang states, that for multimodal problems involving high dimensions, the PSO algorithm tends to suffer from premature convergence. Analysis of the behavior of the particle swarm model reveals that such premature convergence is mainly due to the decrease of velocity of particles in the search space that leads to a total implosion and ultimately fitness stagnation of the swarm. They introduced turbulence in the Particle Swarm Optimization algorithm to overcome the problem of stagnation. The algorithm uses a minimum velocity threshold to control the velocity of particles. The parameter, minimum velocity threshold of the particles is tuned adaptively by a fuzzy logic controller embedded in the TPSO algorithm, which is further called as Fuzzy Adaptive TPSO (FATPSO). The results of the experiments performed illustrates that the FATPSO could prevent premature convergence very effectively and it clearly outperforms SPSO and GA. Premlatha and Natarajan [70] presented the hybrid approaches of Particle Swarm Optimization with Genetic Algorithm (GA). They proposed modification strategies in PSO using GA. The experimental results are show that the proposed hybrid models outperform the standard PSO. Settles and Soule in [74] proposed a novel hybrid GA/PSO algorithm, Breeding Swarms, combining the strengths of particle swarm optimization with genetic algorithms. The hybrid algorithm combines the standard velocity and position update rules of Crossover (VPAC), incorporating the PSO velocity vector. The VPAC crossover operator actively disperses the population preventing premature convergence. The comparison between the hybrid algorithm and the standard GA and PSO models show that the hybrid algorithm is highly competitive, often outperforming both GA and PSO.

3.3. Ant Colony Optimization Techniques

Ant System (AS) can be applied to the optimization problems like traveling salesman problem [23], the quadratic assignment and job-shop scheduling [5]. The inspiring source of ACO is the foraging behavior of real ants. When searching for food, ants initially explore the area surrounding their nest in a random manner. As soon as an ant finds a food source, it evaluates the quantity and the quality of the food and carries some of it back to the nest. During the return trip, the ant deposits a chemical pheromone trail on the ground. The quantity of pheromone deposited, which may depend on the quantity and quality of the food, will guide other ants to the food source. The main principle behind these interactions is called stigmergy, or communication through the environment. An example is pheromone laying on trails followed by ants. Pheromone is a potent form of hormone that can be sensed by ants as they travel along trails. It attracts ants and therefore ants tend to follow trails that have high pheromone concentrations. This causes an autocatalytic reaction, i.e., one that is accelerated by itself. Ants attracted by the pheromone will lay more of the same on the same trail, causing even more ants to be attracted see Fig. 3. This characteristic makes swarm intelligence very attractive for routing networks, robotics and optimization. A number of extensions are proposed to the original ant algorithm. These algorithms performed better producing much improved results than the original ant algorithm. Deneubourg, Aron, Goss and Pasteels [21] explained that the workers of the Argentine ant (Iridomyrmex humilis) start to explore a chemically unmarked territory randomly. As the exploratory front

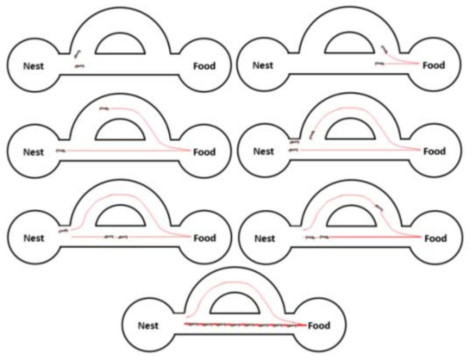

Fig. 3 Path Optimization by Ants.

advances, other explorers are recruited and a trail extends from it to the nest. Whereas recruitment trails are generally constructed between two points, these exploratory trails have no fixed destination, and strongly resemble the foraging patterns of army ants. A minimal model shows how the exploratory pattern may be generated by the individual worker's simple trail laying and following behavior, illustrating how complex collective structures in insect colonies may be based on self-organization. Colorni, Dorigo and Meniezzo [16] explored the implications that the study of ants behavior can have on problem solving and optimization. They introduced a distributed problem solving environment and proposed its use to search for a solution to the travelling salesman problem. Later, Dorigo, M. Maniezzo, V. and Colorni, A. proposed ant colony optimization based on this foraging behavior of the ants in [26]. The main characteristics of this model are positive feedback, distributed computation, and the use of a constructive greedy heuristic. The basic algorithm introduced by them [18] is given by following steps:

```
i. Set parameters, initialize the pheromone trails
ii. while (termination condition not met) do
    a. Construct ants solutions
    b. Apply local search
    c. Update pheromones
iii. end while
```

Dorigo and Gambardella in [24] applied ant colony optimization to solve travelling salesman problem. They proposed [24] an artificial ant colony capable of solving the traveling salesman problem (TSP). Ants of the artificial colony are able to generate successively shorter feasible tours by using information accumulated in the form of a pheromone trail deposited on the edges of the TSP graph. Computer simulations demonstrate that the artificial ant colony is capable of generating good solutions to both symmetric and asymmetric instances of the TSP. The method is an example, like simulated annealing, neural networks, and evolutionary computation, of the successful use of a natural metaphor to design an optimization algorithm. Di Caro and Dorigo in [22] introduced a distributed algorithm that is applied to the traveling salesman problem (TSP). In his algorithm, a set of cooperating agents called ants cooperate to find good solutions to TSPs. Ants cooperate using an indirect form of communication mediated by pheromone they deposit on the edges of the TSP graph while building solutions. The results show that ACS outperforms other nature inspired algorithms such as simulated annealing and evolutionary computation. They concluded comparing ACS-3-opt, a version of ACS augmented with a local search procedure, to some of the best performing algorithms for symmetric and asymmetric TSPs. Dorigo, Di Caro and Gambardella [25] presented algorithms for discrete optimization that took inspiration from the observation of ant colonies foraging behavior, and introduces the ant colony optimization metaheuristic. Guntsch and Middendorf [36] investigated strategies for pheromone modification of ant algorithms in reaction to the insertion/deletion of a city of Traveling Salesperson Problem instances. They proposed three strategies for pheromone diversification through equalization of the pheromone values on the edges. One strategy acts globally without consideration of the position of the inserted/deleted city. The other strategies perform pheromone modification only in the neighborhood of the inserted/deleted city, where neighborhood is defined differently for the two strategies [3]. The Probabilistic Traveling Salesman Problem (PTSP) is a TSP problem where each customer has a given probability of requiring a visit. The goal is to find an a priori tour of minimal expected length over all customers, with the strategy of visiting a random subset of customers in the same order as they appear in the a priori tour.

Gottlieb, Puchta and Solnon [35] described and compared several heuristic approaches for the car sequencing problem. Yaseen and AL-Salami [80] introduced ACO as a distributed algorithm and applied it to solve Traveling Salesman Problem.

4. Conclusion

Although there are many techniques in literature to solve multiobjective optimization problems such as Evolutionary Algorithms, Particle Swarm Optimization Technique, Ant Colony Optimization and the Hybrid Algorithms of all these techniques but the best technique to handle the multiobjective optimization problems is not yet clear. The result of different algorithms depends upon different parameters of the respective algorithms but it is to be discovered that exactly in which ways these factors are to be altered to get better result for different kind of problems and how will a solution seeker set the parameters in the chosen algorithm to get the best result in minimum number of iterations.

5. Future Scope

In this paper we reviewed various techniques to solve multiobjective optimization problems (MOOPs) such as Evolutionary Algorithms (EAs), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO) and their various versions. Firstly MOOPs are described and categorized into two general approaches viz., Classical Approach (preference-based multi-objective optimization) and Ideal Approach (Pareto optimal approach). Both of these techniques are explained briefly. Considering EAs, PSOs and ACOs as the possible efficient solutions to these problems, these techniques are also described and work done by various researchers in this area are listed as a literature review. Suitability of these three techniques to particular types of problems is also established. Thus, we found that there is no technique that can be categorized clearly as the best techniques to solve the MOOPs. Some techniques are better for one kind of problems while some others are better for other kind of MOOPs. Also, the performance of these algorithms depends on various parameters used in the respective algorithms, and the choice of values of these parameters also depends on the problem in hand.

Acknowledgement

Authors would like to thank the UGC-Special Assistance Program for the Department of Physics and Computer Science, Dayalbagh Educational Institute, for funding this research work.

References

[1] Abraham, A., & Jain, L. (2005). Evolutionary Multiobjective Optimization. Springer.
[2] Adriansyah, A., & Amin, S. H. Analytical and Empirical Study of Particle Swarm Optimization with a Sigmoid Decreasing Inertia Weight.
[3] Bianchi, L., Gambardella, L. M., & Dorigo, M. (2002). An Ant Colony Optimization Approach to the Probabilistic Traveling Salesman Problem. *Proceedings of PPSN-VII, Seventh International Conference on Parallel Problem Solving from Nature. 2439*, pp. 883-892. Berlin, Germany: Springer Verlag.
[4] Binh, T. T., & Korn, U. (1996). An evolution strategy for the multiobjective optimization.
[5] Blackwell, T. Particle Swarm Optimization in Dynamic Environments. *Evolutionary Computatation in Dynamic and Uncertain Environments , 51*.
[6] Blackwell, T., & Branke, J. (2004). Multi-swarm optimization in dynamic environments. *Applications of Evolutionary Computing , 3005*, 489-500.
[7] Blum, C. (2005). Beam-ACO: hybridizing ant colony optimization with beam search: an application to open shop scheduling. *Computers and Operations Research , 32* (6), 1565 - 1591 .
[8] Blumel, A. L., Hughes, E. J., & White, B. A. (2000). fuzzy autopilot design using a multi objective evolutionary algorithm.
[9] Brockhoff, D. (2009). *Theoretical Aspects of Evolutionary Multiobjective Optimization—A Review.* France.
[10] Carlisle, A., & Dozier, G. (2000). Adapting Particle Swarm Optimization to Dynamic Environments. *Proceedings of The International Conference on Artificial Intelligence* , (pp. 429-434).
[11] Cauvery, N. K., & Viswanatha, K. V. (2008). Enhanced Ant Colony Based Algorithm for Routing in Mobile Ad Hoc Network. *World Academy of Science, Engineering and Technology , 46*, 30-35.
[12] Chen, Y., & Ye, F. (2004). Particle swarm optimization algorithm and its application to clustering analysis. *In the*

Proceedings of IEEE International Conference on Networking, Sensing and Control, *2*, pp. 789-79.

[13] Clerc, M., & Kennedy, J. (2002). The Particle Swarm - Explosion, Stability, and Convergence in a Multidimensional Complex Space. *IEEE Transactions on Evolutionary Computation , 6*, 58-73.

[14] Coello, C. A., & Lamont, G. B. AN INTRODUCTION TO MULTI-OBJECTIVE EVOLUTIONARY ALGORITHMS AND THEIR APPLICATIONS.

[15] Coello, C. A., Pulido, G. T., & Lechuga, M. S. (2004). Handling multipe objectives with particle swarm optimization. *IEEE Transactions on Evolutionary Computation , 8*, 256-279.

[16] Colorni, A., Dorigo, M., & Maniezzo, V. (1991). Distributed Optimization by Ant Colonies. *PROCEEDINGS OF ECAL91 - EUROPEAN CONFERENCE ON ARTIFICIAL LIFE* (pp. 134-142). PARIS, FRANCE: ELSEVIER PUBLISHING.

[17] Corne, D., Knowles, J. D., & Oates, M. J. (2000). The Pareto Envelope-Based Selection Algorithm for Multi-objective Optimisation. *PPSN VI Proceedings of the 6th International Conference on Parallel Problem Solving from Nature.*

[18] Deb, K. (1999). Multi-objective Genetic Algorithms: Problem. *IEEE Evolutionary Computation , 7*, 205-230.

[19] Deb, K. (2005). *Multiobjective Optimization Using Evolutionary Algorithms.* Chichester: Jhon Wiley & SonsLtd.

[20] Deb, K., & Agrawal, S. *Understanding Interactions among Genetic Algorithm Parameters.* KanGAL Report.

[21] Deneubourg, J. L., Aron, S., Goss, S., & Pasteels, J. M. (1990). The self-organizing exploratory pattern of the argentine ant. *Journal of Insect Behavior* , 159-168.

[22] Di Caro, G., & Dorigo, M. (1998). Antnet: Distributed stigmergetic control communications networks. *Journal of Artificial Intelligence Research* , 317-365.

[23] Dorigo, M., & Gambardella, L. M. (1997). Ant colonies for the traveling salesman problem. *BioSystems , 43*, 73-81.

[24] Dorigo, M., & Gambardella, L. M. (1997). Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. *IEEE Transactions on Evolutionary Computation , 1* (1), 53-66.

[25] Dorigo, M., Caro, G. D., & Gambardella, L. M. (1999). Ant Algorithms for Discrete Optimization. *Artificial Life , vol. 5* (2), 137-172.

[26] Dorigo, M., Maniezzo, V., & Colorni, A. (1996). The Ant System: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics , 26* (1), 29-41.

[27] Elhossini, A., & Areibi, S. (2010). Strength Pareto Particle Swarm Optimization and Hybrid EA-PSO for Multi-Objective Optimization. *Evolutionary Computation , 18* (1), 127-156.

[28] Fan, S.-K. S., & Chang, J.-M. (2009). A parallel particle swarm optimization algorithm for multi-objective optimization problemms. *Engineering Optimization , 41* (7), 673 - 697.

[29] Filedsend, J. E. (2004). *Multi-objective particle swarm optimisation methods.* Technical Report, University of Exeter, Department of Computer Science.

[30] Fogel, D. B. (1988). *Biological Cybernetics* , 139-144.

[31] Gambardella, L. M., & Dorigo, M. (1995). Ant-Q: A Reinforcement Learning approach to the traveling salesman problem. *Twelfth International Conference on Machine Learning*, (pp. 252-260).

[32] Gambardella, L. M., & Dorigo, M. (1996). Solving Symmetric and Asymmetric TSPs by Ant Colonies. *Proceedings of the IEEE Conference on Evolutionary Computation.* Nagoya, Japan.

[33] Gambardella, L. M., Taillard, E., & Agazzi, G. (1999). MACS-VRPTW: A Multiple Ant Colony System for Vehicle Routing Problems with Time Windows. In M. D. D. Corne (Ed.), *New Ideas in Optimization* (pp. 63-76). United Kingdom: McGraw-Hill, UK.

[34] Ghosh, S., Kundu, D., Suresh, K., Das, S., Abraham, A., Panigrahi, B. K., et al. (2009). On Some Properties of the lbest Topology in Particle Swarm Optimization. *Proceedings of Ninth International Conference on Hybrid Intelligent Systems* (pp. 370–376). China: IEEE Computer Society Press.

[35] Gottlieb, J., Puchta, M., & Solnon, C. (2003). A study of greedy, local search, and ant colony optimization approaches for car sequencing problems. *In the Proceedings of the 2003 international conference on Applications of evolutionary computing* (pp. 246-257). Springer-Verlag Berlin, Heidelberg.

[36] Guntsch, M., & Middendorf, M. (2001). Pheromone Modification Strategies for Ant Algorithms Applied to Dynamic TSP. *Proceedings of EvoWorkshops 2001, Lake Como, Italy, Springer Verlag*, (pp. 213-222).

[37] Hamdan, S. A. (n.d.). Hybrid Particle Swarm Optimizer using multi-neighborhood topologies.

[38] Heppner, F., & Grenander, U. (1990). A stochastic nonlinear model for coordinated bird flocks. In S. Krasner (Ed.), *The Ubiquity of Chaos* (pp. 233-238).

[39] Hertz, A., & Zu, N. (n.d.). A New Ant Algorithm for Graph Coloring.

[40] Higashi, N., & Iba, H. (2003). Particle swarm optimization with Gaussian mutation. *warm Intelligence Symposium 2003.* Proceedings of the 2003 IEEE.

[41] Hongbo, L., Abraham, A., & Zhang, W. (2005). Fuzzy adaptive turbulent particle swarm optimization. *Proceedings of The Fifth Internantional Conference on Hybrid Intelligent System*, (p. 6 pp.).

[42] Hu, X., & Eberhart, R. C. (2002). Adaptive particle swarm optimization: detection and response to dynamic systems. *Proceedings of the IEEE Congress on Evolutionary Computation*, (pp. 1666-1670). Honolulu, Hawaii, USA.

[43] Hu, X., & Eberhart, R. C. (2002). Multiobjective optimization using dynamic neighborhood particle swarm optimization. *Proceedings of the IEEE Congress on Evolutionary Computation*, (pp. 1677-1681). Honolulu, Hawaii USA.

[44] Hu, X., Eberhart, R. C., & Shi, Y. (2003). Paerticle Swarm with Extended Memory for Multiobjective Optimisation. *Proceedings of the IEEE Swarm Intelligence Symposium 2003*, (pp. 193-197). Indianapolis, Indiana, USA.

[45] Hwang, S. K., Koo, K., & Lee, J. S. (2005). Homogeneous particle swarm optimizer for multi-objective optimization

problem. *International Conference on Artificial Intelligence and Machine Learning.* Cairo, Egypt.

[46] Janson, S., & Middendorf, M. (2004). A Hierarchical Particle Swarm Optimizer for Dynamic Optimization Problems. *Proc. Evoworkshops 2004: 1st European Workshop on Evolutionary Algorithms in Stochastic and Dynamic Environments* (pp. 513-524). Springer.

[47] Janson, S., & Middendorf, M. (2006). A hierarchical particle swarm optimizer for noisy and dynamic environments. *Genetic Programming and Evolvable Machines , 7* (4), 329-354.

[48] Jian, L. (2009). Solving capacitated vehicle routing problems via genetic particle swarm optimization. *Proceedings of the 3rd international conference on Intelligent information technology application*, (pp. 528-531). China.

[49] Jin, Y., Okabe, T., & Sendhoff, B. (2001). Adapting Weighted Aggregation for Multiobjective Evolution Strategies. In *Evolutionary Multi-Criterion Optimization* (pp. 96-110). Springer Verlag.

[50] Kennedy, J., & Eberhart, R. C. (1997). A discrete binary version of the particle swarm algorithm. *Proceedings of the 1997 Conference on Systems, Man, and Cybemetics* (pp. 4104-4109). Piscataway, NJ, USA: IEEE.

[51] Kennedy, J., & Eberhart, R. C. (1995). Particle Swarm Optimization. *In Proceedings of the Fourth IEEE Intrenational Confrence on Neural Networks* (pp. 1942-1948). Perth, Australia: IEEE Srvice Center.

[52] Kennedy, J., & Mendes, R. (2002). Population structure and particle swarm performance. *Proceedings of the 2002 Congress on Evolutionary Computation. 2*, pp. 1671-1676. IEEE Computer Society Washington, DC, USA.

[53] Khanesar, M. A., Teshnehlab, M., & Shoorehdeli, M. A. (2007). A Novel Binary Particle Swarm Optimization. *Proceedings of the 15th Mediterranean Conference on Control and Automation.* Athenes, Greece.

[54] Kodru, P., Das, S., & Welch, S. M. (2007). Multi-Objective and Hybrid PSO Using ε-Fuzzy Dominance. *Proceedings of the Genetic and Evolutionary Computing Conference*, (pp. 853-860). London, UK.

[55] Konak, A., Coit, D. W., & Smith, A. E. (2006). Multi-objective optimization using gnetic algorithms: A tutorial. *Reliability Engineering and system safety* .

[56] Koppen, M., & Veenhuis, C. (2006). Multi-objective particle swarm optimization by fuzzy-pareto-dominance meta-heuristic. *International Journal of Hybrid Intelligent Systems , 3* (4), 179-186.

[57] Li, X. (2003). A Non-dominated Sorting Particle Swarm Optimizer for Multiobjective Optimization. (E. C. al., Ed.) *Lecture Notes in Computer Science , 2723*, 37-48.

[58] Mendes, R., Cortez, P., Rocha, M., & Neves, J. (n.d.). Particle swarm for feedforward neural network training.

[59] Mendes, R., Kennedy, J., & Neves, J. (2005). The Fully Informed Particle Swarm: Simpler,Maybe Better. *IEEE TRANSACTIONS OF EVOLUTIONARY COMPUTATION , 1* (1).

[60] Mendes, R., Kennedy, J., & Neves, J. (2003). Watch Thy Neighbor or How the Swarm Can Learn From Its Environment. *Proceedings of Swarm Intelligence Symposium, IEEE* (pp. 88-94). IEEE.

[61] Minnebo, W. (n.d.). Multi-Objective Optimization using Particle Swarms.

[62] Moss, J., & Johnson, C. G. (2003). An ant colony algorithm for multiple sequence alignment in bioinformatics. (D. Pearson, N. Steele, & R. Albrecht, Eds.) *Springer Computer Series* , 182-186.

[63] Mostaghim, S., & Teich, J. (2004). Covering Pareto-optimal Fronts by Subswarms in Multi-objective Particle Swarm Optimization. *IEEE* , 1404-1411.

[64] Omkar, S. N., Khandelwal, R., Ananth, T. V., Naik, G., Narayana, & Gopalakrishnan, S. (2009). Quantum behaved Particle Swarm Optimization (QPSO) for multi-objective design optimization of composite structures. *Expert Systems with Applications , 36* (8), 11312-11322.

[65] Parsopolus, K. E., Tasoulis, D. K., & Vrahatis, M. N. (2004). Multiobjective optimization using parallel vector evaluated particle swarm optimization. *Proceedings the IASTED international conference on artificial intelligence and applications, as part of the 22nd IASTED international multi-conference on applied informatics.* Innsbruck Austria.

[66] Parsopolus, K. E., & Vrahatis, M. N. (2008). Multi-Objective Particles Swarm Optimization Approaches. In S. A. Lam Thu Bui (Ed.), *Multi-Objective Optimization in Computational Intelligence: Theory and Practice* (pp. 20-42). IGI Global.

[67] Parsopolus, K. E., & Vrahatis, M. N. (2007). Parameter selection and adaptation in Unified Particle Swarm. *Mathematical and Computer Modelling* , 198-213.

[68] Parsopolus, K. E., & Vrahatis, M. N. (2002). Particle Swarm Optimization Method in Multiobjective Problems. *Proceedings of the ACM 2002 Symposium on Applied Computing* (pp. 603-607). Madrid, Spain: ACM New York, NY, USA.

[69] Pederson, M. E. (2010). *Good Parameters for Particle Swarm Optimization.* Technical Report, Hvass Laboratories.

[70] Premalatha, K., & Natarajan, A. M. (2009). Hybrid PSO and GA for Global Maximization. *International Journal Open Problems Compt. Math. , 2* (4), 597-608.

[71] Ptowin, J. Y. (2007). *Evolutionary Algorithm for Vehicle Routing Problem.*

[72] Rizzoli, A. E., Montemanni, R., Lucibello, E., & Gambardella, L. M. (2007). Ant colony optimization for real-world vehicle routing problems. In *Swarm Intelligence* (Vol. 1, pp. 135-151). Springer Science.

[73] Sbalzarini, I. F., Muller, S., & Koumoutsakos, P. (2000). Multiobjective optimization using evolutionary algorithms. *In the Proceedings of CTR summer program 2000.* Standford University, Stanford.

[74] Settles, M., & Soule, T. (2005). Breeding Swarms: A GA/PSO Hybrid. *Proceeding of the Genetic and Evolutionary Computation Conference (GECCO-2005).* Washington D.C.: ASME press.

[75] Shi, Y., & Eberhart, R. C. (1998). A modified particle swarm optimizaer. *Proceedings of IEEE International Conference on Evolutionary Computation*, (pp. 69-73).

[76] Sierra, M. R., & Coello, C. A. Improving PSO-based Multi-Objective Optimization.

[77] Srinivas, N., & Deb, K. (1994). Multiobjective Optimization using Nondominated Sorting in Genetic Algorithms. *Evolutionary Computation , 2* (3), 221-248.
[78] Wang, Y.-X., & Xiang, Q.-L. (2008). Particle Swarms with Dynamic Ring Topology. *IEEE Congress on Evolutionary Computation (CEC 2008)*, (pp. 419-423).
[79] Xiao-hua, Z., Hong-yun, M., & Li-cheng, J. (2005). Intelligent particle swarm optimization in multiobjective optimization. *Congress on Evolutionary Computation. 1*, pp. 714 – 719. Edinburg, Scotland, UK: IEEE press.
[80] Yaseen, S. G., & AL-Slamy, N. M. (2008). Ant Colony Optimization. *IJCSNS International Journal of Computer Science and Network Security , 8* (6), 351-357.
[81] Zhang, Q., & Xue, S. (2007). An Improved Multi-Objective Particle Swarm Optimization Algorithm. *Lecture Notes in Computer Science , 4683*, pp. 372-381.
[82] Zhang, Y., & Rockett, P. (2007). A Comparison of three evolutionary strategies for multiobjective genetic programming. *Artificial Intelligence Review , 27* (2-3), 149-163.
[83] Zitzler, E., & Thiele, L. (1999). Multiobjective Evolutionary Algorithms:A Comparative Case Study and the Strength Pareto Approach. *IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, , 3* (4), 257-271.
[84] Zitzler, E., Deb, K., & Thiele, L. (2000). Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation , 8* (2), 173-195.