*J. Basic. Appl. Sci. Res.*, 2(11)11388-11401, 2012

ISSN 2090-4304
**Journal of Basic and Applied Scientific Research**
*www.textroad.com*

# GMGA *VS*MGA and PSO Methods for Functional Constrained Optimization Problems

**Mohammad Yadollahi, Majid Moavenian**

Department of Mechanics Engineering, University of Ferdowsi, Mashhad, Iran

## ABSTRACT

A typical GA solves unconstrained optimization problems, so that a traditional GA method was presented in which penalty functions have been utilized as the selection criterion of surviving individuals to apply GA for constrained problems. Sensitivity of the convergence properties of this method to the penalty parameter makes the try for a recently new method on title of "Modified Genetic Algorithm (MGA)", with a special selection criterion to overcome this sensitivity difficulty; but this new method is not general to apply in any optimization problem; thus, in this paper, a method has been proposed to generalize the recent modified GA. Furthermore, in this method, the cost of iteration has been minimized by selecting a suitable termination check. In this paper, it also has been tried to compare the mentioned generalized modified GA with the particle swarm optimization (PSO) method. The last section of the article consists of application of this method in some famous optimization problems and a MDO (multidisciplinary design optimization) in aerospace field.
**KEYWORDS**: Genetic Algorithm; Functional Constraint; Optimization; PSO

## 1. INTRODUCTION

In the design of today's increasingly complex engineering systems, the designer is increasingly dependent on computationally expensive computer analysis and simulation codes [1]. Physics, Biology, Economy or Sociology often have to deal with the classical problem of optimization [2].

The apparent billion years triumph of biological evolution has inspired computer scientists to implement, analyze, and utilize similar methods to solve e.g. optimization problems that are known or seem to be computationally hard and/or complex. This has led to the study of genetic algorithms (GA) and other evolutionary methods in global optimization and related engineering fields[3].

Genetic Algorithm(GA) is inspired by the mechanism of natural selection which is coined as "survival of the fittest". The basic principles of GA were first proposed by Holland, now GA has become an important research field of multiple disciplines[4].

John Holland is often considered the inventor of Genetic Algorithms. The name genetic algorithm originates from biology, because in the method populations are generated, best ones are selected, crossover in genes and mutation occurs. Genetic algorithms can be described as being global search methods. They are optimization algorithms, which do not require information about derivatives(gradients)[5].

Perhaps the most attractive features of genetic algorithms are on one hand their simplicity, which makes it easy to implement and tailor them to practical problems and on the other and their ability to solve hard problems. The other prominent feature is that GAs are general. They are quite independent of the details of the object problems.

GAs operate on a population of potential solutions applying the principle of survival of the fittest to produce successively better approximations to a solution[6].

Genetic algorithms are widely used for unconstrained optimization problems with general agreement about the fundamentals of the implementation. But their application to constrained optimization problems remain sun settled. However, many optimization problems, especially those in engineering design, are highly constrained. For such problems, it is necessary to find global optima not violating any constraint. In general, constraints may be linear or non-linear, and equality or inequality in type[7].

Traditionally, external penalty functions have been used to convert a constrained optimization problem into an unconstrained problem for GA-based optimization. This approach requires the somewhat arbitrary selection of penalty draw-down coefficients. [8]Convergence of this conventional method is

***Corresponding Author:** Mohammad Yadollahi, Department of Mechanics Engineering, University of Ferdowsi, Mashhad, Iran. Tel: +98-915-519-4360 Email: mohammad.yadollahi@gmail.com

sensitive to the penalty parameter. If the penalty parameter is substantially larger than its optimum value, which cannot be estimated a priori, the individuals tend to prematurely converge on a local minimum. On the other hand, if the penalty parameter is smaller than its optimum value, the individuals probably cannot converge on a feasible region. Thus, in a method called *modified GA (MGA)*, some remedies have been applied in the selection process to eliminate the difficulty of the penalty function method for GAs[9].

Nevertheless, the mentioned modified GA is not general to cover any problem with any number of optimization variables. Thus, in the method proposed in this paper, which has been called *generalized modified GA (GMGA)*, some modifications have been applied in comparison with that modified GA to cover any optimization problem. Furthermore, in this method, accuracy of the problem is on control and the cost of GA iteration can be minimized as it is possible.

A functional constraint is an implicit or explicit function of one or more optimization variables that restricts the valuation of the optimization variables.

Through application to some famous optimization problems, validation of the proposed method in this study is demonstrated. Furthermore, the proposed method is applied to a MDO application, and the effectiveness of the proposed method for a practical problem is confirmed.

In this paper, the mentioned GMGA is compared with the particle swarm optimization (PSO) method.

## 1- Genetic Algorithms

The limited utility of standard search algorithms in our problem of interest led to the development of a random 'genetic' type search[10].

Genetic Algorithm (GA), which borrows the idea of Darwinian principle of natural selection, is a powerful global search and optimization method.

Genetic algorithms for optimization are non-derivative, non-descent, random-search procedures for functional minimization, and their algorithmic structure is based on biological concepts[11].

### *1-1- Basic Form of Genetic Algorithm*

The several kinds of evolutionary algorithms are distinguished by the problems to which they are applied, the codings by which their chromosomes represent candidate solutions, the operators they apply to those chromosomes, and how they perform and use selection. Genetic algorithms(GAs) are most often applied to problems of combinatorial optimization, whose solutions they encode as strings of symbols[12].

A simple genetic algorithm composes of five operators: evaluation, selection, crossover, mutation, and sampling. The *evaluation* operator assigns a fitting score to a "gene" (represented by a vector of real numbers or binary ones) in the population. Superior gene should be granted better fitting score than inferior counterparts[13].

*t := 0;*

*Compute initial population* $B_0$*;*

**WHILE***stopping condition not fulfilled***DO**
**BEGIN**
*Selectindividuals for reproduction;*
*Createoffsprings by crossing individuals;*
*Eventuallymutate some individuals;*
*Computenew generation*
**END**

As obvious from the above algorithm, the transition from one generation to the next consists of four basic components:

**Selection:** Mechanism for selecting individuals (strings) for reproduction according to their fitness (objective function value)[14]. This operator is crucial for the performance of the GA, since it may lead the algorithm to premature convergence and limited search scope (or genetic diversity) by repeatedly choosing very strong individuals with similar genetic code[15].

**Crossover:** Method of merging the genetic information of two individuals; if the coding is chosen properly, two good parents produce good children.

**Mutation:** In real evolution, the genetic material can by changed randomly by erroneous reproduction or other deformations of genes, e.g. by gamma radiation. In genetic algorithms, mutation can berealized as a random deformation of the strings with a certain probability. The positive effect is preservation of genetic diversity and, asan effect, that local maxima can be avoided.

**Sampling:** Procedure which computes a new generation from the previous one and its off springs.
The four above stages differ from a kind of GA to another, so that there are several kinds of GA. The examples for *crossover* part are as: *Uniform crossover*[16], *One-Point crossover* and *Two-Point crossover*, and also *selection* part is as: *tournament selection*. However, GAs are classified in two main category: *Real-Coded* and *Binary-Coded* GA. In the initial formulation of *genetic algorithms*(GAs), the search space solutions are coded using the binary alphabet; however, other coding types, such as real coding, have also been taken into account to deal with the representation of the problem[17]. Real-coded genetic algorithms (RCGAs) attract attention as numerical optimization methods for nonlinear systems[18].

*1-2- Conventional GA with Adaptive Penalty Functions* [8]

Most applications of genetic algorithms (GAs) in handling constraints use a straightforward penalty function method[19].

The genetic algorithm (GA) has been receiving increasing use as a global search and optimization methodology, and GA applications now extend to aerospace optimization problems.

The GA is well suited to unconstrained optimization, yet most "real-world" engineering design problems involve constrained optimization. To remedy this, it has been common practice to use external penalty functions to transform a constrained objective function into an unconstrained fitness function as bellow:

$$f(X) = \varphi(X) + \sum_{i=1}^{N_{CON}} P_i(X) \qquad (1)$$

Nomenclature

c : penalty draw-down coefficient $f(X)$: fitness function n: generation number $N_{CON}$ :number of constraints $P(X)$: penalty function $X$: design variable vector $\varphi(X)$: objective function $\sigma$ : standard deviation of fitness $\sigma^2$: variance of fitness $g(X)$: constraint function , $g(X) \geq 0$

1-2-1 Penalty Function Forms

When penalty functions were first applied to genetic algorithms for constrained optimization, the external quadratic penalty function was used because of its tradition of use with calculus-based unconstrained optimization techniques. This was used as one of three penalty function forms investigated in this study and provided the baseline approach. This penalty function is generally expressed as:

$$P_i = c_i \left[\max(0, g_i(X))\right]^2 \qquad (2)$$

The continuous derivative requirement of calculus-based methods does not exist in the GA-based optimization approach, so the second penalty function form investigated in this study was an external linear penalty function. This form is simply:

$$P_i = c_i \left[\max(0, g_i(X))\right] \qquad (3)$$

Further, the GA has no requirement for objective function continuity. Because of this, an external step-linear functional form was the also investigated in this study. This form uses the draw-down coefficient to determine the step size and is expressed as:

$$P_i = \begin{cases} 0 & if \quad g_i(X) \leq 0 \\ c_i \left[1 + g_i(X)\right] & else \end{cases} \qquad (4)$$

1-2-2 Draw-Down Coefficient Strategies

Choosing the draw-down coefficient values, $c_i$ , for a penalty function is often arbitrary. A fundamental tradeoff to be considered when using a penalty function lies in the proper choice of the draw-down coefficient. A small coefficient will impose a smaller penalty than a large coefficient for the same magnitude of constraint violation. In the GA, a large penalty (resulting in a poor fitness) can quickly eliminate infeasible solutions from the search. These infeasible solutions may contain building blocks, or schema,1 that are key elements of the optimal solution; therefore, it may be beneficial to allow infeasible designs to survive in pursuit of the optimum. Conversely, using a small draw-down coefficient may allow the survival of infeasible designs to the

extent that the population converges at an infeasible point as the optimal fitness solution. Clearly, a compromise must be struck between these two extremes. The goal of an adaptive penalty function is to change the value of the draw-down coefficient during the search allowing exploration of infeasible regions to find optimal building blocks, while preserving the feasibility of the final solution.

Two basic forms of draw-down coefficient strategies have been identified in a previous study: generation number-based and population fitness-based. These strategies have been amended and used for this work.

First, a constant draw-down coefficient provided a reference to the traditional approach.

$$c = 2 \quad \textit{(constant valued)} \tag{5}$$

The generation number-based strategies increase the value of the draw-down coefficient with successive generations. These strategies are expressed as:

$$c = n \quad \textit{(linear increase)} \tag{6}$$

$$c = n^2 \quad \textit{(quadratic increase)} \tag{7}$$

$$c = 2^n \quad \textit{(exponential increase)} \tag{8}$$

Fitness-based strategies are meant to increase the penalty coefficient when the population fit nesses are diverse, causing the population to move toward an optimal feasible design; and to decrease the coefficient when the population begins to become homogeneous, allowing some infeasible designs with important design information to survive. These forms use the standard deviation and variance of the population's fitness values and are:

$$c = \sigma \quad \textit{(standard deviation)} \tag{9}$$

$$c = \sigma^2 \quad \textit{(variance)} \tag{10}$$

These six strategies for draw-down coefficients were applied to two mathematical test functions and an engineering design problem. Performance of the different strategies were measured in terms of the best feasible solutions discovered by the GA and in terms of computational cost as measured by the number of generations required to reach a stopping criterion.

*1-3- Modified Genetic Algorithm*[9]

The real-coded GA, in which the variables vector X (all the optimization variables gathered in one vector called variables vector) is regarded as the gene of each individual, is used in this method. The algorithm of this approach is outlined as follows.

1-3-1 Initialization

The initial $N_P$ population of the variables is prepared at random. Each variable is determined by the uniform random numbers.

1-3-2 Crossover

Multi parental unimodal normal distribution crossover (UNDX-m) is used as the crossover model. This model achieves an efficient global search. The algorithm of UNDX-m is as follows: $(m+2)$ parents $X_P^{(1)},\ldots,X_P^{(m+2)}$ are selected at random. The median point of the first $(m+1)$ parents is defined as $X_G$ ; i.e..

$$X_G = \frac{1}{m+1}\sum_{j=1}^{m+1} X_P^{(j)} \tag{11}$$

The difference vectors of each parent are defined as

$$\hat{d}^{(j)} = X_P^{(j)} - X_G \quad , \qquad (j = 1,\ldots,m+2) \tag{12}$$

Let $\hat{D}$ be the length of the component of $\hat{d}^{(m+2)}$ orthogonal to $\hat{d}^{(1)},\ldots,\hat{d}^{(m)}$. Moreover, let $\hat{e}^{(1)},\ldots,\hat{e}^{(n-m)}$ be the orthonormal basis of the subspace orthogonal to $\hat{d}^{(1)},\ldots,\hat{d}^{(m)}$. Generate children $X_C^{(i)}\ (i=1,\ldots,N_C)$ by the equation

$$X_C^{(i)} = X_G + \sum_{j=1}^{m} w_j\,\hat{d}^{(j)} + \hat{D}\sum_{j=1}^{n-m} v_j\,\hat{e}^{(j)},(n\geq 3) \qquad (13)$$

Where *n* is the length of the variables vector X and $w_j, v_j$ are random numbers that conform to the normal distribution with 0 mean and variance of $\sigma_w^2, \sigma_v^2$ respectively. $\sigma_w, \sigma_v$ are specified by the value determined by following equations. Note that *m* is valued by the user depending on the problem and the value of *n* (*m* must be less than *n*).

$$\sigma_w = \frac{1}{\sqrt{m}} \quad , \quad \sigma_v = \sqrt{\frac{3(m+1)}{2(n-m)(m+2)}} \qquad (14)$$

1-3-3 Selection

Select the surviving individuals from $(m+2)$ parents and $N_C$ children based on some criteria. In this step of the GA, a fitness function is usually used as a criterion of selection. In order to handle the constraints, a penalty function has been used frequently as part of the fitness function, of the form

$$F_r(X) = F(X) + rE(X) \qquad (15)$$

$$E(X) = \sum_{i=1}^{m_E} |G_i(X)| + \sum_{i=1}^{m_I} \max[0, H_i(X)] \qquad (16)$$

where $G(X)=0$ (with $m_E$ dimension) and $H(X)\leq 0$ (with $m_I$ dimension) are the functional constraint vectors of the problem, furthermore and $E(X)$ denote the penalty parameter and the constraint error, respectively. Because the optimal value of *r* cannot be estimated a priori, *r* is usually specified as an arbitrarily large value so that the global minimum of the penalty function (15) becomes that of the original constrained problem. However, the individuals tend to converge on a local minimum in the case of large *r*, because the contribution of the objective function to the penalty function with large *r* is small, speciallyin the early stages of evolution (i.e., when the constraint error of each individual is large).

*Step 1*

Rank the generated $N_C$ children in ascending order on the penalty function. Set the rank parameter $i=1$

*Step 2*

Carry out *the* selection with respect to the $i$ th child $X_C^{(i)}$: if there are some parents $X_P^{(j)}$ that satisfy both

$$F\left(X_C^{(i)}\right) \leq F\left(X_P^{(j)}\right) \qquad (17)$$

$$F_r\left(X_C^{(i)}\right) \leq F_r\left(X_P^{(j)}\right) \qquad (18)$$

Replace the nearest parent $X_P^{(n)}$ with $X_C^{(i)}$,[ i.e., $\left\| X_P^{(n)} - X_C^{(i)} \right\|_2$ is minimum among the parents that satisfy both requirements (17) and (18) ]. If the replacement was performed,or $i=N_C$, or there are no parents that satisfy (18), go to step 3. Otherwise, set $i \rightarrow i+1$ and repeat step 2.

*Step **3***

Newly rank the children in ascending order on the objective function and set the rank parameter $k=1$.

*Step **4***

Carry out the selection with respect to the $k$ th child $X_C^{(k)}$: if there are some parents $X_P^{(j)}$ that satisfy

$$F_r\left(X_C^{(k)}\right) \le F_r\left(X_P^{(j)}\right) \tag{19}$$

replace the nearest parent $X_P^{(n)}$ with $X_C^{(k)}$. If the replacement was performed, or $k = N_C$, go to termination check step. Otherwise, set $k \to k+1$ and repeat step 4.

1-3-4 Termination Check

If the generation number (crossover and selection introduced above, correspond to one generation) equals the specified number $N_G$, terminate the algorithm. Otherwise return to crossover step. (End)

No mutation scheme is performed in the above GA, because UNDX-m crossover also plays the role of mutation, i.e., random perturbation of the problem variable vector.

## 2- Generalized Modified GA

The method introduced in thesection (1-3) is applied for optimization problems with more than three variables because of its kind of crossover. The proposed method in this section is general for any kind of optimization problem. In this method some theory correction has been occurred comparing with the modified GA mentioned above.

### *2-1 Initialization*

This part of the proposed method is the same as modified GA, presented in section (2-3-1), i.e., in this method, the initial $N_P$ population of the variables is also prepared at random.

### *2-2 Crossover*

If the number of optimization variables was more than three, the UNDX-m model of crossover is used; otherwise, for problems with less than three optimization variables, a UNDX model is applied. Therefore, a multi selectable crossover is proposed to cover the optimization problems with any number of variables.

*UNDX Crossover*

When optimizing function has epistasis among parameters the UNDX generates children near the linesegment connecting two parents so that the children lie on the valley where the parents locate (Fig.1). The mathematical description can be written as follows: where $c_1$ and $c_2$ are children, $p_1$ and $p_2$ are parents and *m* is the middle point of parents.

$$c_1 = m + z_1 e_1 + \sum_{k=2}^{l} z_k e_k \,, c_2 = m - z_1 e_1 - \sum_{k=2}^{l} z_k e_k \tag{20}$$

$$z_1 \sim N\left(0, {\sigma_1}^2\right) \quad , \quad z_k \sim N\left(0, {\sigma_2}^2\right)\left(k = 2, \ldots, l\right) \quad , \quad \sigma_1 = \alpha\, d_1 \quad , \quad \sigma_2 = \beta\, d_2 / \sqrt{l}$$

$$m = \left(p_1 + p_2\right)/2 \tag{21}$$

$$e_1 = \left(p_2 - p_1\right) / \left| p_2 - p_1 \right| \qquad , e_i \perp e_j \; i, j = 0, \ldots, n_{param} \; ; i \neq j \tag{22}$$

$d_1$ is the distance between parents and $d_2$ isthe distance between the third parent $p_3$ (randomly selected)and the line connecting $p_1$ to $p_2$. $z_1 \sim N\left(0, {\sigma_1}^2\right)$ and $z_k \sim N\left(0, {\sigma_2}^2\right)\left(k = 2, \ldots, l\right)$ are normal distributed random numbers. $\alpha$ and $\beta$ are constants given by the user.

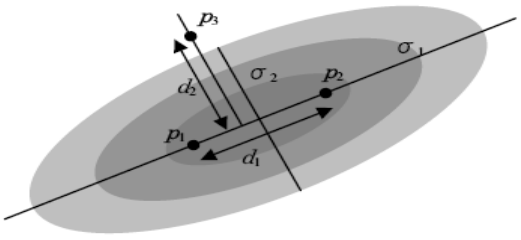


Figure1:UNDX Crossover

***2-3** Selection*

In the modified GA, the surviving individuals are selected from $(m+2)$ parents and $N_C$ children based on its selection criteria, but in this proposed method, surviving individuals have been selected from $N_P$ parents and $N_C$ children based on the same selection criteria as that of the modified GA, i.e., all the surviving parents of each generation in the population have been participated with their children in the selection. By this, all the surviving individuals in the population are given the chance of selection, so that the possibility of creating feasible individuals with the best value of fitness becomes greater than before.

***2-4** Termination Check*

In the modified GA, by monitoring the results of a trial run, the maximum generation number $N_G$ was determined as the value for which further improvement of the average of the penalty function could be regarded as negligible. For getting an appropriate $N_G$, the problem must be pre-optimized and this requires to have enough information on problem and to spend more cost. In this proposed method, the termination criterion is the differential error between the optimal point of present generation and that of the previous one, i.e., this error must be less than an accuracy value given by user. By this technique, the termination of algorithm is on control, i.e., the optimum can be achieved with problem required accuracy. Furthermore, this termination criterion makes the cost of algorithm iteration be decreased.

## 3- Optimization through Particle Swam Intelligence[20-23]

Particle Swarm Optimization (PSO) is a population based stochastic optimization technique developed by Dr. Eberhart and Dr. Kennedy in 1995, inspired by social behavior of bird flocking or fish schooling.
PSO shares many similarities with evolutionary computation techniques such as Genetic Algorithms (GA). The system is initialized with a population of random solutions and searches for optima by updating generations. However, unlike GA, PSO has no evolution operators such as crossover and mutation. In PSO, the potential solutions, called particles, fly through the problem space by following the current optimum particles.

Each particle keeps track of its coordinates in the problem space which are associated with the best solution (fitness) it has achieved so far. (The fitness value is also stored.) This value is called pbest. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the neighbors of the particle. This location is called lbest. When a particle takes all the population as its topological neighbors, the best value is a global best and is called gbest.

The particle swarm optimization concept consists of, at each time step, changing the velocity of (accelerating) each particle toward its pbest and lbest locations (local version of PSO). Acceleration is weighted by a random term, with separate random numbers being generated for acceleration toward pbest and lbest locations.

In past several years, PSO has been successfully applied in many research and application areas. It is demonstrated that PSO gets better results in a faster, cheaper way compared with other methods.

Another reason that PSO is attractive is that there are few parameters to adjust. One version, with slight variations, works well in a wide variety of applications. Particle swarm optimization has been used for approaches that can be used across a wide range of applications, as well as for specific applications focused on a specific requirement.

Vik+1 = wVik +c1 rand1(…) x (pbesti-sik) + c2 rand2(…) x (gbest-sik) ….. (23)

where,
vik : velocity of agent i at iteration k,
w: weighting function,
cj : weighting factor,
rand : uniformly distributed random number between 0 and 1,
sik : current position of agent i at iteration k,
pbesti : pbest of agent i,
gbest: gbest of the group.

The following weighting function is usually utilized in (23)

w = wMax-[(wMax-wMin) x iter]/maxIter (24)

where

wMax= initial weight,
wMin = final weight,
maxIter = maximum iteration number,
iter = current iteration number.

$$sik+1 = sik + Vik+1 \quad (25)$$

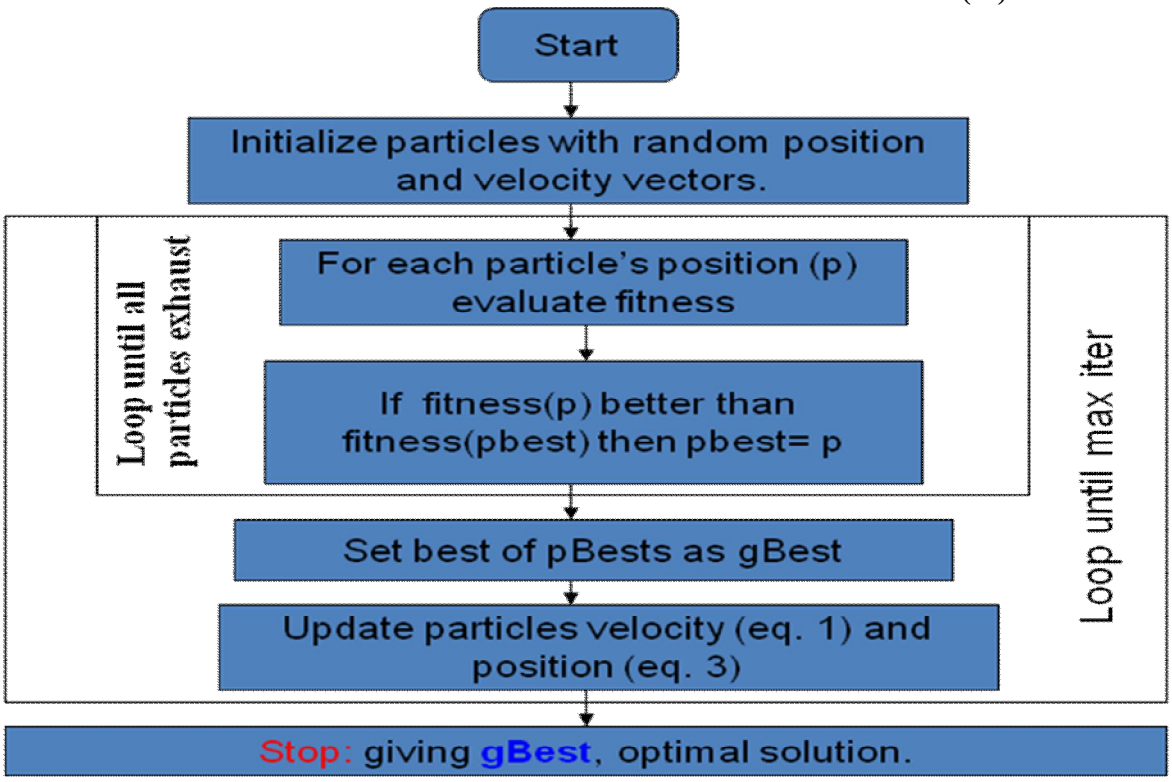


Figure 2 : PSO Algorithm Flow Chart

## 4- Application Examples

In this section, some optimization problems have been presented which use the "Generalized Modified GA" method as the optimization operator.

### *4-1- One-dimensional Optimization Problem*

*A Constrained Problem*

A simple one-dimensional math problem was studied, because the solution for this problem is easy to determine and visualize. The problem was simply:

$$min\ f_1(x) = x^2$$
$$subject\ to\ x \geq 2$$

*Solution Based on Conventional GA with Adaptive Penalty Functions:*

Table 1

| **Penalty Function Type** → | **External Quadratic** | | **External Linear** | | **External Step-Linear** | |
|---|---|---|---|---|---|---|
| Draw-Down Strategy ↓ | *Optimal* $f_1$ | *Optimal* $x$ | *Optimal* $f_1$ | *Optimal* $x$ | *Optimal* $f_1$ | *Optimal* $x$ |
| Constant | 4.0157 | 2.0039 | 6.5092 | 2.5513 | 4.0157 | 2.0039 |
| Generation Linear Increase | 5.3672 | 2.3167 | 4.0157 | 2.0039 | 4.0157 | 2.0039 |
| Generation Quadratic Increase | 4.1739 | 2.0430 | 4.9238 | 2.2190 | 4.0157 | 2.0039 |
| Generation Exponential Increase | 4.0944 | 2.0235 | 4.0157 | 2.0039 | 4.0157 | 2.0039 |
| Fitness-Based Standard Deviation | 4.0157 | 2.0039 | 4.0157 | 2.0039 | 4.1739 | 2.0430 |
| Fitness-Based Variance | 4.3352 | 2.0821 | 4.0944 | 2.0235 | 4.0157 | 2.0039 |

*Solution Based on Generalized Modified GA:*

Table 2

| Optimal $f_1$ | Optimal $x$ | Accuracy | Maximum Generation Number |
|---|---|---|---|
| 4.108943 | 2.027053 | $10^{-2}$ | 12 |
| 4.009286 | 2.002320 | $10^{-3}$ | 29 |
| 4.000019 | 2.000005 | $10^{-5}$ | 74 |
| 4.000001 | 2.000000 | $10^{-6}$ *Maximum Accuracy* | 176 |

*4-2- Two-dimensional Optimization Problem*

*An Unconstrained Problem*

Find the maximum of the following function.

$$f_2 : [-10,10]^2 \to R$$

$$(x, y) \to \frac{1-\sin^2(\sqrt{x^2+y^2})}{1+0.001.(x^2+y^2)}$$

As one can see easily from the plot in Fig. 3, the function has a global maximum in 0 and a lot of local maxima.

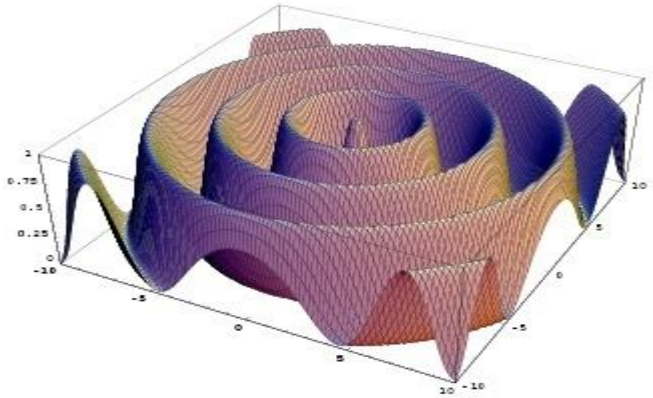

Figure 3. A surface plot of the function $f_2$

*Solution Based on Generalized Modified GA:*

Table 3

| **Optimal $f_2$** | **Optimal $x$** | **Optimal $y$** | **Accuracy** | **Maximum Generation Number** |
|---|---|---|---|---|
| 0.9982519 | $4.144014^{*}10^{-2}$ | $-5.483898^{*}10^{-3}$ | $10^{-3}$ | 148 |
| 0.9997131 | $1.609740^{*}10^{-2}$ | $5.249945^{*}10^{-3}$ | $10^{-4}$ | 461 |
| 0.9999635 | $5.676770^{*}10^{-3}$ | $2.069079^{*}10^{-3}$ | $10^{-5}$ | 594 |
| 1.000000 | $1.402405^{*}10^{-4}$ | $-8.068331^{*}10^{-5}$ | $10^{-8}$ *Maximum Accuracy* | 786 |

*A Constrained Problem*

This two-dimensional mathematical optimization problem is configured as next example. This multimodal objective function would be challenging to solve with a calculus-based method, but is well suited for the genetic algorithm. The problem is:

$$Min\ f_3(x) = x_1^2 + x_2^2 + 25\left(\sin^2 x_1 + \sin^2 x_2\right)$$

$$Subject\ to\ x_1^2 + x_2 \geq 2$$

*Solution Based on Conventional GA with Adaptive Penalty Functions:*

Table 4

| **Penalty Function Type →** | **External Quadratic** | | | **External Linear** | | | **External Step-Linear** | | |
|---|---|---|---|---|---|---|---|---|---|
| Draw-Down Strategy ↓ | *Optimal* $f_3$ | *Optimal* $x_1$ | *Optimal* $x_2$ | *Optimal* $f_3$ | *Optimal* $x_1$ | *Optimal* $x_2$ | *Optimal* $f_3$ | *Optimal* $x_1$ | *Optimal* $x_2$ |
| Constant | 9.6901 | -3.0863 | 0.0583 | 9.5929 | 2.9942 | -0.0583 | 9.6298 | -0.0154 | 3.0925 |
| Generation Linear Increase | 9.6427 | -2.9881 | 0.0706 | 9.6455 | -3.0065 | -0.0768 | 10.0915 | -0.0645 | 2.8775 |
| Generation Quadratic Increase | 9.5145 | -3.0433 | 0.0215 | 9.6506 | -3.0556 | -0.0706 | 9.6269 | -2.9451 | 0.0031 |
| Generation Exponential Increase | 9.6269 | -2.9451 | 0.0031 | 9.5114 | -3.0310 | 0.0276 | 9.5191 | -3.0126 | -0.0338 |
| Fitness-Based Standard Deviation | 9.5819 | -0.0031 | 3.0802 | 9.5212 | -0.0338 | 3.0310 | 9.5816 | -0.0522 | 3.0495 |
| Fitness-Based Variance | 9.6876 | -0.0215 | 2.9328 | 9.6212 | -2.9881 | 0.0645 | 9.9417 | 2.9328 | 0.1013 |

*Solution Based on Generalized Modified GA:*

Table 5

| Optimal $f_3$ | Optimal $x_1$ | Optimal $x_2$ | Accuracy | | Maximum Generation Number |
|---|---|---|---|---|---|
| 9.494057 | -3.032859 | $7.366068^{*}10^{-3}$ | $10^{-2}$ | | 150 |
| 9.492272 | $-9.016127^{*}10^{-4}$ | 3.006921 | $10^{-3}$ | | 250 |
| 9.488236 | $-5.253371^{*}10^{-4}$ | 3.020720 | $10^{-4}$ | | 300 |
| 9.488197 | $-4.223884^{*}10^{-5}$ | 3.019490 | $10^{-6}$ | *Maximum Accuracy* | 451 |

*4-3- MDO Application*

The design optimization problem, in which the objective function is affected byseveral engineering subjects through design process, is called multidisciplinary design optimization (MDO). In this mood, subsystems are designed so that the main system is optimized with respect to the given criterion. In this kind of problems, it is not necessary to optimize each subsystem with respect to its criterion, but it is just enough to optimize total system based on a special criterion [24].

For an aerospace application of the proposed method, the MDO of a flying vehicle is studied in this section. At first, this vehicle must be designed to fly on a trajectory so that its final height must be more than a given value (it is the problem functional constraint, because the flying height is an implicit function of design variables); then, the problem is to optimize the initial mass of vehicle so that it flies on an optimal trajectory.

4-3-1-Problem Formulation [25]

Figure 4 outlines the disciplines and data flow in this paper. A set of values is inserted into the variables and the disciplines are analyzed sequentially. Any analysis module, not only calculates the intermediate variable and passes them to other disciplines, but also computes function values of the equality and inequality constraints which should be satisfied. Problem inputs are demonstrated in figure 3 and consist of three following groups.

*Technical Data and Technological Constraints*

Disciplines input parameters that don't change during design cycle, are considered as technical data. Technological constraints are requirements about technologies to be used (for industrial reasons especially).In the other word, available technologies apply some limitation in design. Rocket motor length, diameter, case strength and material specifications, propellant characteristics and propellant loading density are some technical data and technological constrains.

*Mission Requirements*

These requirements consist of payload mass to be inserted into a given main mission orbit, injection accuracy, limitation on loads encountered by satellite, operational constraints such as range safety and launch azimuth.

*Initial **Point***

Initial values for design variables generate initial point. Based on optimization algorithm selection, needs for feasible initial point may be neglected.

4-3-2- Design Variables

The variables represent geometric shapes of the vehicles (diameter of each stage, $dm_1, dm_2$), propulsion performance(output mass rate of motors of each stage, $\dot{m}_1, \dot{m}_2$), and parameters of optimized flight trajectory(maximum angle of attack during first stage maneuver $\alpha_m$, final pitch angle $\theta_e$).

The design variables and their feasible range are shown in following table:

Table6 : Design Variables

| Design Variable | Description | Feasible Minimum | Feasible Maximum |
|---|---|---|---|
| (deg.) $x_1 = \alpha_m$ | maximum angle of attack in first maneuver | 4 | 6 |
| (deg.) $x_2 = \theta_e$ | terminal value of pitch angle | 35 | 45 |
| (m) $x_3 = dm_1$ | diameter of first stage motor | 1 | 1.2 |
| (m) $x_4 = dm_2$ | diameter of second stage motor | 0.8 | 1 |
| $\left(Kg/s\right) x_5 = \dot{m}_1$ | mass rate of first motor | 168 | 176 |
| $\left(Kg/s\right) x_6 = \dot{m}_2$ | mass rate of second motor | 18 | 24 |

*Subject to (flying height)* $h \geq h_{Constraint}$ ( $h_{Constraint}$ = 44 Km)

4-3-3- Analysis Modules

To formulate design in a MDO problem form, four disciplines such as propulsion, weight, aerodynamic and trajectory, must be considered.

*Propulsion*

The propulsion analysis consists of Rap SRMD analysis code which was developed by MDO Laboratory. This code uses performance prediction equations and calculates principal design parameters. Input parameters include motor diameter, propellant properties, combustion thermo-chemical characteristic, thrust and burning time. Motor design algorithm scans chamber pressure and reference design height for minimum rocket motor weight. The code computes propellant weight, specific impulse, nozzle exhaust velocity, port and throat area, propellant burning area and motor dimensions.

*Weight Modules*

The vehicle weight is broken down into following major subsystems: Propellant, Motor case, Nozzle, Inter stage structure, Payload adapter, Guidance set, and payload.

Propulsion module calculates weight of propellant. Inter stage structure, payload adapter, and guidance set are estimated by statistical curve fitted Mass Estimate Relationships (MERs). Weight of motor case was calculated using special pressure vessels codes.

*Aerodynamics*

The external vehicle mold lines are not allowed to change. Semi empirical equations were used for aerodynamic coefficients computation. These methods developed from databases of analyses, flight tests and wind tunnel data for vehicles that were either flying or existing.

Overall, there is good agreement for simple conventional ELV configuration. Good agreement is not expected for exotic configurations or revolutionary concepts due to the fact that Semi empirical codes are based on empirical methods.

The modeling program generates tabulated aerodynamics coefficients. These tables present coefficient values relative to Mach number, Reynolds number and angle of attack. The trajectory module interpolates these data. The effect of aerodynamics shape and coefficients in system level variables has been considered.

*Trajectory Analysis*

This study implements a three-degree-of-freedom(3DOF) trajectory analysis. State variables are velocity, flight path angle, range, altitude and mass. Control variables are maximum angle of attack during first maneuver and optimized pitch program parameters. Generally, the trajectory analysis computes state variable profiles by integrating equations of motion with given control variable profiles and examines satisfaction ratings of constraint conditions during the flight. One of the characteristics in this study is to optimize the flight trajectory with other design variables, as already described [25].

By combining four discipline codes, described above, according to figure 4, a homogenous design environment has been achieved.

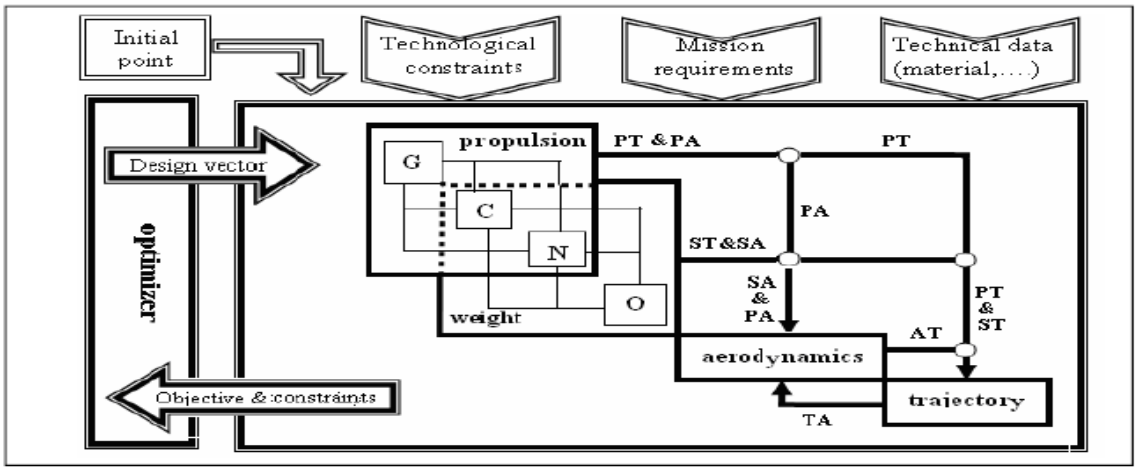


Figure4 : MDO Problem Formulation

*Optimal Solution Based on Conventional GA with Adaptive Penalty Functions:*

Table 7

| **Penalty Function Type** $\rightarrow$ | **External Quadratic** | | | | | | |
|---|---|---|---|---|---|---|---|
| Draw-Down Strategy $\downarrow$ | *Optimal Initial Mass* | *Optimal* $X_1$ | *Optimal* $X_2$ | *Optimal* $X_3$ | *Optimal* $X_4$ | *Optimal* $X_5$ | *Optimal* $X_6$ |
| Constant | 15535.0713 | 5.253177 | 35.00 | 1.015640 | 0.837537 | 168.007828 | 18.00 |
| Generation Linear Increase | 15535.0713 | 5.253177 | 35.00 | 1.015640 | 0.837537 | 168.007828 | 18.00 |
| Generation Quadratic Increase | 15535.0713 | 5.253177 | 35.00 | 1.015640 | 0.837537 | 168.007828 | 18.00 |
| Generation Exponential Increase | 15535.0713 | 5.253177 | 35.00 | 1.015640 | 0.837537 | 168.007828 | 18.00 |
| Fitness-Based Standard Deviation | 15560.8975 | 5.345064 | 35.879765 | 1.018768 | 0.800000 | 168.703812 | 18.046921 |
| Fitness-Based Variance | 15550.6191 | 5.259042 | 36.573803 | 1.020332 | 0.935484 | 168.00 | 18.193548 |
| | External Linear | | | | | | |
| Constant | 15535.0713 | 5.253177 | 35.00 | 1.015640 | 0.837537 | 168.007828 | 18.00 |
| Generation Linear Increase | 15535.0713 | 5.253177 | 35.00 | 1.015640 | 0.837537 | 168.007828 | 18.00 |
| Generation Quadratic Increase | 15535.0713 | 5.253177 | 35.00 | 1.015640 | 0.837537 | 168.007828 | 18.00 |
| Generation Exponential Increase | 15535.0713 | 5.253177 | 35.00 | 1.015640 | 0.837537 | 168.007828 | 18.00 |
| Fitness-Based Standard Deviation | 15550.6191 | 5.259042 | 36.573803 | 1.020332 | 0.935484 | 168.00 | 18.193548 |
| Fitness-Based Variance | 15550.6191 | 5.259042 | 36.573803 | 1.020332 | 0.935484 | 168.00 | 18.193548 |
| | External Step-Linear | | | | | | |
| Constant | 15535.0713 | 5.253177 | 35.00 | 1.015640 | 0.837537 | 168.007828 | 18.00 |
| Generation Linear Increase | 15535.0713 | 5.253177 | 35.00 | 1.015640 | 0.837537 | 168.007828 | 18.00 |
| Generation Quadratic Increase | 15535.0713 | 5.253177 | 35.00 | 1.015640 | 0.837537 | 168.007828 | 18.00 |
| Generation Exponential Increase | 15535.0713 | 5.253177 | 35.00 | 1.015640 | 0.837537 | 168.007828 | 18.00 |
| Fitness-Based Standard Deviation | 15550.6191 | 5.259042 | 36.573803 | 1.020332 | 0.935484 | 168.00 | 18.193548 |
| Fitness-Based Variance | 15550.6191 | 5.259042 | 36.573803 | 1.020332 | 0.935484 | 168.00 | 18.193548 |

*Optimal Solution Based on Generalized Modified GA & PSO :*

Table8 : Optimal Results of MDO

| Method | Optimal Initial Mass | Optimal $X_1$ | Optimal $X_2$ | Optimal $X_3$ | Optimal $X_4$ | Optimal $X_5$ | Optimal $X_6$ | Flight Height [Km] |
|---|---|---|---|---|---|---|---|---|
| GMGA | 15534.19 | 5.252387 | 35.00 | 1.015640 | 0.837537 | 168.007828 | 18.00 | 44.14 |
| PSO | 15536.00 | 5.30 | 34.50 | 1.10 | 0.81 | 167.90 | 18.30 | 44.00 |

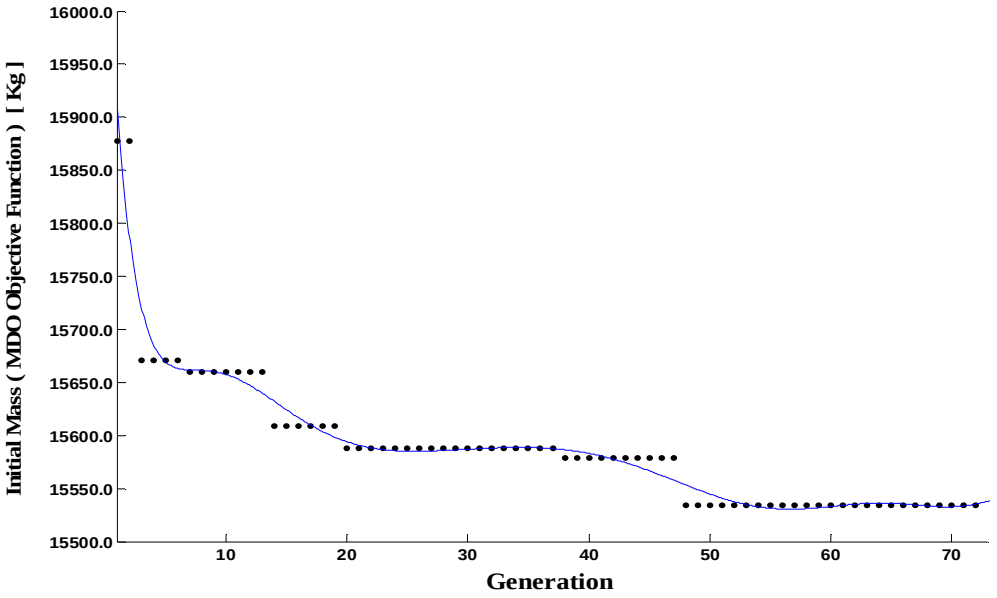


Figure 5 : Convergence Process of Proposed GA in MDO Application

## 5- Conclusion

Through application to one and two-dimensional optimization problems and a MDO application with six optimization variables, it was demonstrated that the generalized modified GA, proposed in this article, could be used for any kind of optimization problem with any number of variable. As it was observed in the constrained and unconstrained problems solved in sections (4-1) and (4-2), this proposed GA could be suitable to optimize any constrained and unconstrained problem with any value of accuracy. Furthermore, this method specially is useful for problems with functional constraints (i.e., constraints that areimplicit or explicit function of one or more optimization variables). Controllability and generalization of this optimization method are its main properties. Rate of convergence in this method depends on constraint type and value of accuracy given by user.

The sensitivity of convergence properties of conventional method,which is based on penalty functions, to penalty parameters makes the generalized modified GA operate better than that conventional method.

PSO shares many similarities with evolutionary computation techniques such as Genetic Algorithms (GA), but unlike GA, PSO has no evolution operators such as crossover and mutation. PSO like MGA is weak to cover the problems with several numbers of variables, unlike GMGA.

## REFERENCES

1. K. Hacker & K. Lewis, “Robust Design Through the Use of a Hybrid Genetic Algorithm”, *ASME Members, University at Buffalo*
2. Jean-Philippe Rennard, Ph.D., “Introduction to genetic algorithms, Genetic Algorithm Viewer: Demonstration of a Genetic Algorithm”, *May 2000*
3. Jarmo T. Alander, “Genetic Algorithms – an introduction”, *Department of Information Technology and Production Economics, University of Vaasa, PL 700 , 65101 Vaasa*

4. Z. Du, M. Ding, Sanli Li, Shuyou Li, M. Wu & J. Zhu, "Massively Parallel SPMD Algorithm for Cluster Computing-Combining Genetic Algorithm with Uphill", *Tsinghua University, Beijing, China*
5. J.-S.R. Jang, C.-T Sun & E. Mizutani, "Genetic Algorithms", *Examples from Neuro-Fuzzy and SoftComputing,*
6. A.J. Chipperfield & P.J. Fleming, "The MATLAB Genetic Algorithm Toolbox"
7. Suzan E. Carlson & R. Shonkwiler, "Annealing a Genetic Algorithm over Constraints",*University of Virginia, Charlottesville, Virginia&School of Mathematics, GeorgiaInstitute of Technology, Atlanta*
8. William A. Crossley & Edwin A. Williams, "A Study of Adaptive Penalty Functions for Constrained Genetic Algorithm – BasedOptimization", *Purdue University, West Lafayette, IN 47907-1282*
9. N. Yokoyama & S. Suzuki, "Modified Genetic Algorithm for Constrained Trajectory Optimization", *University of Tokyo, Tokyo 113-8656, Japan, Journal of "Guidance, Control, and Dynamics", Vol.28, No.1, January-February 2005*
10. C. Formoso, "Genetic Search Algorithm for Large Problems", *Division of Child Support, Olympia, WA*
11. Daniel D. Moerder & Bandu N. Pamadi, "Constrained Minimization of Smooth Functions Using a Genetic Algorithm",*LangleyResearchCenter, Hampton, Virginia*
12. Bryant A. JULSTROM, "Codings and Operators in Two Genetic Algorithms for the Leaf-Constrained Minimum Spanning Tree Problem",*Department of Computer Science, St. CloudStateUniversity, St. Cloud, USA*
13. M. Zhu & Kai-min Kevin Chang, "A Real-Coded Genetic Algorithm Approach to Data Segmentation Problem",*University of Waterloo, Faculty of Mathematics*
14. U. Bodenhofer, "Genetic Algorithms : Theory and Applications", *Lecture Notes, Third Edition – Winter 2003/2004*
15. J. Sanchez-Velazco & John A. Bullinaria, "Gendered Selection Strategies in Genetic Algorithms for Optimization", *School of Computer Science, University of Birmingham, Birmingham, UK*
16. "An Overview of Genetic Algorithms : Part 2, Research Topics" , *University Computing, 1993, 15(4) 170-181*
    *D. Beasley: Department of Computing Mathematics, University of Wales Collage of Cardiff, Cardiff, CF2 4YN, UK*
    *D. R. Bull: Department of Electrical and Electronic Engineering, University of Bristol, Bristol, BS8 1TR, UK*
    *R. R. Martin: Department of Computing Mathematics, University of Wales Collage of Cardiff, Cardiff, CF2 4YN, UK*
17. F. Herrera, M. Lozano & A. M. Sanchez, "Hybrid Crossover Operators for Real-Coded Genetic Algorithms: An Experimental Study", *Department of Computer Science and Artificial Intelligence, University of Granada, Granada, & University of Vigo, Orense, Spain*
18. T. Ueda, N. Koga & M. Okamoto, "Efficient Numerical Optimization Technique Based on Real-Coded Genetic Algorithm",*School of Bio-resource and Bio-environmental Sciences & Department of Biochemical Engineering and Science, Kyushu University, Japan*
19. K. Deb & S. Agrawal, "A Niched-Penalty Approach for Constraint Handling in Genetic Algorithms",*Kanpur Genetic Algorithms Lab. (Kan GAL), Dep. of Mechanical Engineering, Indian Institute of Technology Kanpur.*
20. Maurice Clerc, "Particle Swarm Optimization", *book, First published in Great Britain and the United States in 2006 by ISTE Ltd*
21. Poli, R. (2008). "Analysis of the publications on the applications of particle swarm optimisation" . *Journal of Artificial Evolution and Applications 2008*
22. Shi, Y. and Eberhart, R. C.,"Parameter selection in particle swarm optimization. Evolutionary Programming",*VII: Proc. EP 98 pp. 591-600. Springer-Verlag, New York, 1998*
23. Yin, P., Glover, F., Laguna, M., & Zhu, J.,"A Complementary Cyber Swarm Algorithm",*International Journal of Swarm Intelligence Research (IJSIR), (2011)*
24. Douglas C. Montgomery, "Design and Analysis of Experiments", *John Wiley & Sons ING , 1997-2001*
25. J. Roshanian, M. Ebrahimi & J. Jodei, "An Automated Approach to Multidisciplinary System Design Optimization of Small Solid Propellant Launch Vehicles", *MDO Laboratory, Department of Aerospace Engineering, K. N. Toosi University of Technology, Tehran, Iran*