53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, 23 - 26 April 2012, Honolulu, Hawaii
SDM 2012 Student Papers Competition

Avoiding Premature Convergence in a Mixed-Discrete Particle Swarm Optimization (MDPSO) Algorithm

Souma Chowdhury* and Jie Zhang*
Rensselaer Polytechnic Institute, Troy, New York 12180
and
Achille Messac†
Syracuse University, Syracuse, NY, 13244

Over the past decade or so, Particle Swarm Optimization (PSO) has emerged to be one of most useful methodologies to address complex high dimensional optimization problems - it's popularity can be attributed to its ease of implementation, and fast convergence property (compared to other population based algorithms). However, a premature stagnation of candidate solutions has been long standing in the way of its wider application, particularly to constrained single-objective problems. This issue becomes all the more pronounced in the case of optimization problems that involve a mixture of continuous and discrete design variables. In this paper, a modification of the standard Particle Swarm Optimization (PSO) algorithm is presented, which can adequately address system constraints and deal with mixed-discrete variables. Continuous optimization, as in conventional PSO, is implemented as the primary search strategy; subsequently, the discrete variables are updated using a deterministic *nearest vertex approximation* criterion. This approach is expected to avoid the undesirable discrepancy in the rate of evolution of discrete and continuous variables. To address the issue of premature convergence, a new adaptive diversity-preservation technique is developed. This technique characterizes the population diversity at each iteration. The estimated diversity measure is then used to apply (i) a dynamic repulsion towards the globally best solution in the case of continuous variables, and (ii) a stochastic update of the discrete variables. For performance validation, the Mixed-Discrete PSO algorithm is successfully applied to a wide variety of standard test problems: (i) a set of 9 unconstrained problems, and (ii) a comprehensive set of 98 Mixed-Integer Nonlinear Programming (MINLP) problems.

Keywords: constraint, discrete variable, mixed-integer nonlinear programming (MINLP), Particle Swarm Optimization, population diversity

I. Introduction

A. An Overview of Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a stochastic optimization algorithm that imitates the dynamics of social behavior observed in nature. This algorithm was introduced by an Electrical Engineer, Russel C. Eberhart, and a Social Psychologist, James Kennedy.[1] The underlying philosophy of PSO and swarm intelligence can be found in the book by Kennedy et al.[2] PSO has emerged over the years to be one of

*Doctoral Student, Multidisciplinary Design and Optimization Laboratory, Department of Mechanical, Aerospace and Nuclear Engineering. AIAA Student Member
†Distinguished Professor and Department Chair. Department of Mechanical and Aerospace Engineering. AIAA Lifetime Fellow. Corresponding author. Email: messac@syr.edu

the most popular population-based heuristic optimization approaches. Several variations of PSO have been reported in the literature, and applied to diverse optimization problems in engineering, basic sciences and finance.[3] The modifications of the PSO algorithm presented in this paper are inspired by the optimization challenges, encountered in the authors' research in product family design[4] and wind farm optimization.[5] Both these optimization problems (defined as single-objective) involve complex multimodal criterion functions and a high dimensional system of mixed-discrete design variables. These problems generally require a large number of system-model evaluations; they also suffer from early premature convergence during optimization when a standard population-based algorithm is used.

In the case of **constrained single-objective optimization** problems, population-based algorithms, e.g., evolutionary and swarm-based optimization methods, often suffer from premature stagnation.[3] This undesirable property can be attributed to an excessive and mostly unopposed pressure of exploration or evolution. The simultaneous presence of continuous and discrete design variables that may experience differing rates of evolution further complicates the optimization scenario. In this paper, a new method is developed to both characterize and infuse diversity, adaptively, into the population of candidate solutions. This method is an evolution from earlier diversity-preservation methods reported in the PSO literature, which are later discussed in Section D. The PSO algorithm presented in this paper can address a mixture of discrete and continuous design variables. Two distinct yet coherent approaches are developed to address the diversity-preservation issues for discrete and continuous variables.

A comprehensive review of the background and the development of Particle Swarm Optimization based algorithms (till 2007) can be found in the chapter by Banks et al.[6] An extensive follow up review of the various attributes of PSO, and the applicability of PSO to different classes of optimization problems: unconstrained/constrained, combinatorial, and multicriteria optimization, can be found in the book chapter by Banks et al.[3] Brief surveys of reported variations of PSO that address the following critical optimization attributes: (i) mixed-discrete variables, (ii) population diversity preservation, and (iii) constraint handling, are provided in Sections B, D, and C, respectively.

B. Existing Mixed-Discrete Optimization Approaches

A significant amount of research has been done in developing algorithms for solving **Mixed-Integer Non-Linear Programming (MINLP)** problems. Most of these algorithms are gradient-based search techniques. Three major categories of gradient-based algorithms are (i) the branch and bound, (ii) the cutting plane, and (iii) the outer approximation algorithms. A list of these algorithms, related discussion, and bibliography can be found in the websites of MINLP World[7] and CMU-IBM Cyber-Infrastructure for MINLP.[8] These algorithms possess attractive numerical properties, namely (i) fast convergence, (ii) proof of optima, and (iii) an intrinsic ability to deal with constraints. However, gradient-based algorithms do not readily apply to the broad scope of engineering design problems that may involve highly nonlinear, non-smooth and multimodal criterion functions.

Among population-based optimization methods, binary Genetic Algorithms (GAs)[9,10] have been reported to be effective for discrete optimization. Binary GAs convert the design variables into binary strings. This process leads to an approximate discrete representation of the continuous variables. A population of candidate solutions, each represented by a binary string, evolve over generations, through the four stages: (i) fitness assignment, (ii) selection, (iii) crossover, and (iv) mutation. One of the most popular binary GAs is the bin-NSGA-II developed by Deb et al.[11] Genetic algorithms have been successfully implemented on MINLP problems, such as batch plant design.[12,13] Another class of discrete optimization algorithms, which belong to Ant Colony Optimization (ACO), have also been reported in the literature.[14,15] Applications of ACO-based algorithms to discrete optimization problems include vehicle routing, sequential ordering, and graph coloring. There exists in the literature a handful of variations of the PSO algorithm that can address discrete and/or integer variables. A summary of these variations of PSO is discussed in the following section.

C. Mixed-Discrete Particle Swarm Optimization: Principles and Objectives

This paper presents fundamental modifications to the original dynamics of PSO, with the aim to solve *highly constrained single-objective mixed-discrete optimization* problems. The development of this Mixed-Discrete PSO (MDPSO) is driven by the following specific objectives:

i. Develop an approximation technique that can address mixed-discrete design variables through continuous optimization;

ii. Include a constraint handling technique to deal with both equality and inequality constraints; and

iii. Formulate an explicit diversity preservation technique to avoid the stagnation of particles.

Efficient diversity preservation provides a conducive environment for the first and the second objectives. Hence, the third objective is considered to be the primary contribution of this paper. A method is formulated to characterize the existing diversity in the population and adjust the diversity parameter(s)/coefficient(s) at every iteration. **This approach provides a generalized adaptive regulation of the population diversity, which can be implemented in a majority of population-based optimization algorithms and is not restricted to PSO**. For example, the concerned diversity parameter can be (i) the *mutation probability* in genetic algorithms,[10] or (ii) the *time-varying acceleration coefficients* (TVAC) in PSO[16] or (iii) the *window-size of the hypercube operator* in Predator-Prey algorithms,[17] or (iv) the *random selection rate* in Ant Colony Optimization.[18]

A majority of the existing Mixed-Discrete PSO algorithms are hindered by the effects of differing rates of evolution of the continuous and discrete design variables. To avoid this limiting scenario, continuous optimization is applied as the primary search strategy for all variables, whether they are continuous or discrete. After the particles have moved to their new locations, the discrete component of the design vector for each particle is approximated to the nearest feasible discrete domain location. In this case, nearness is determined using the Euclidian distance in the discrete variable space. As a result, although the variables evolve through continuous search dynamics, system-function evaluations are performed only at the allowed discrete locations. This approach is partly similar to the strategy presented by Laskari et al.[19] A schematic of the proposed mixed-discrete optimization approach for each candidate solution is shown in Fig. 1.

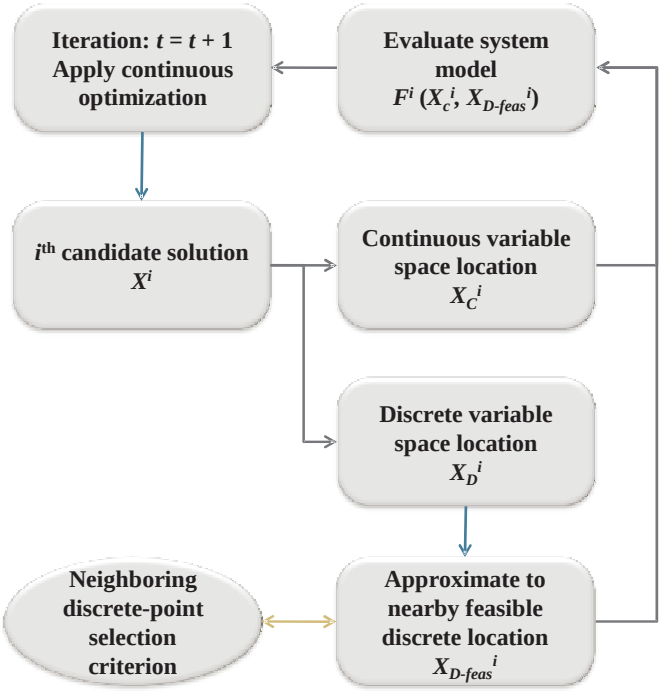


Figure 1. Process diagram of the generalized approach to MDNLO

Constraint handling in MDPSO is performed using the principle of constrained non-dominance that was introduced by Deb et al.[11] This method has been successfully implemented in the Non-dominated Sorting Genetic Algorithm-II,[11] Modified Predator-Prey algorithm,[17] and other standard evolutionary algorithms. The MDPSO algorithm involves a set of coefficients that regulate the inertia, the personal behavior, the social behavior, and the diversity preserving behavior of the particles. Parameter selection in PSO is far from trivial, as discussed in the previous section. However, detailed analysis of the selection of PSO parameters, and the

ensuing numerical behavior of the particle dynamics are not within the scope of this paper. In this paper, we specifically intend to provide

i. the detailed formulation of the Mixed-Discrete PSO algorithm,

ii. the underlying hypothesis supporting the proposed modifications, and

iii. the performance of this modified algorithm on a wide variety of test cases.

It is important to note that, considering the volume of interesting research in PSO reported in the literature over the past decade, other existing characteristic modifications might further advance the performance of the MDPSO algorithm. For validation purposes, the MDPSO algorithm is applied to (i) a set of standard unconstrained nonlinear optimization problems,[20,21] and (ii) a comprehensive set of MINLP problems.[22] In the next Section, the formulation of the **Mixed-Discrete Particle Swarm Optimization (MDPSO)** algorithm is presented. This formulation includes redefining the particle dynamics, addressing discrete variables in optimization, incorporating the constraint management technique, and developing the new diversity preservation technique. Results and subsequent discussions regarding the application of MDPSO to various standard test problems are provided in Section III.

II. Development of the Mixed-Discrete Particle Swarm Optimization (MDPSO)

A. Dynamics of Particle Swarm Optimization

1. Literature Survey: Dynamics of Particle Motion in PSO

A balance between exploration, exploitation, and population-diversity in PSO requires appropriate quantification of the PSO coefficients, or what is more popularly termed as *parameter selection*. One of the earliest strategies to balance exploration and exploitation was the introduction of the inertia weight.[6] Shi and Eberhart[23] investigated the influences of the inertia weight and the maximum velocity on the algorithm performance. Using numerical experiments, they proposed particular values (and/or range of values) for the inertia weight and the maximum velocity, and also suggested the application of time varying inertia weight to further improve the algorithm performance. Trelea[24] used standard results from dynamic systems theory to provide graphical parameter selection guidelines. The applications of control theory by Zhang et al.,[25] and chaotic number generation by Alatas et al.[26] are among the recently proposed methods used to establish parameter selection guidelines (for PSO).

2. Dynamics of Particle Motion in MDPSO

A general mixed-discrete single-objective constrained minimization problem involving m discrete variables and a total of n design variables can be expressed as

$$\begin{aligned} &\text{Min} \quad f(X) \\ &\text{subject to} \\ &\qquad g_j(X) \le 0, \quad j = 1, 2, ..., p \\ &\qquad h_k(X) = 0, \quad k = 1, 2, ..., q \\ &\text{where} \\ &\qquad X = \left[\begin{array}{ccccccc} x_1 & x_2 & \dots & x_m & x_{m+1} & \dots & x_n \end{array} \right] \end{aligned} \tag{1}$$

where p and q are the number of inequality and equality constraints, respectively. In Eq. 1, X is the design variable vector, where the first m variables are discrete and the next $n - m$ variables are continuous. To solve this optimization problem, the PSO algorithm is initialized with N random particles. To this end, the Sobol's quasirandom sequence generator[27] is applied. Sobol sequences use a base of two to form successively finer uniform partitions of the unit interval, and then reorder the coordinates in each dimension. The location of each particle in the swarm is updated using a velocity vector at each iteration; the velocity vector of a particle is variable, and is itself updated at every iteration. In the MDPSO algorithm, the velocity vector

update formula is redefined to allow for an explicit diversity preservation term. The modified dynamics of the particle motion can be represented as

$$\begin{aligned} X_i^{t+1} &= X_i^t + V_i^{t+1} \qquad \text{and} \\ V_i^{t+1} &= \alpha V_i^t + \beta_l r_1 \left(P_i - X_i^t\right) + \beta_g r_2 \left(P_g - X_i^t\right) \\ &\quad + \gamma_c r_3 \hat{V}_i^t \end{aligned} \tag{2}$$

where,

- X_i^t is the location of the i^{th} particle at the t^{th} iteration;
- r_1, r_2 and r_3 are random real numbers between 0 and 1;
- P_i is the best candidate solution found for the i^{th} particle;
- P_g is the best candidate solution for the entire population;
- α, β_l and β_g are the user defined coefficients that control the inertial, the exploitive, and the explorative attributes of the particle motion; and
- γ_c is the diversity preservation coefficient for continuous design variables.

The determination of the diversity preservation coefficient (γ_c) is discussed in Section D. The global (P_g) and the local best (P_i) solutions are updated at every iteration using the solution comparison principle. This solution comparison principle is based on the values of the corresponding criterion functions: objective function and constraint functions. This principle is discussed in Section C. The continuous update process (Eq. 2) is applied to *all* the design variables of a particle. Following this process, the discrete component of the design vector is updated to nearby feasible discrete locations. In this case, feasibility pertains to the constraints imposed by the discreteness of the variable space, and not to the system constraints.

B. Addressing Discrete Variables in PSO

1. Literature Survey: Discrete and Combinatorial PSO

Several variations of the PSO algorithm that can solve combinatorial optimization problems have been reported in the literature. Kennedy and Eberhart[28] presented one of the earliest modification of PSO to address binary variables. They defined the trajectories of the binary variables in terms of the change in the probability that a value of one or zero will be taken. Tasgetiren et al.[29] used construction/destruction operators to perturb the discrete component of the variable vector of a particle in solving a Traveling Salesman problem. A similar combinatorial-PSO concept was also developed and used by Jarboui et al[30] for resource-constrained project scheduling. These variations of the PSO algorithm provide efficient and robust performances typically for combinatorial optimization problems that are similar to the corresponding reported applications. A majority of these methods do not readily apply to the broad scope of mixed-discrete optimization that involves problems with: (i) *integers and/or real-valued discrete variables*, (ii) *non-uniformly spaced discrete variable values* (e.g., $x \in [1, 3, 100, 1000, \ldots]$) and (iii) *widely different sizes of the "set of feasible values" for the discrete variables* (e.g., $x_1 \in [0, 1]$ and $x_2 \in [1, 2, \ldots, 1000]$).

Kitayama et al.[31] developed a more generalized approach to address discrete variables using a penalty function - discrete variables are treated as continuous variables by penalizing at the intervals. However, the additional multimodal constraint in the penalty function-based approach may undesirably increase the complexity of the design problem. Singh et al.[32] presented an interesting approach to address discrete variables by manipulating the random operators in the particle-velocity update step. This approach can be very helpful in maintaining consistency in the rates of evolution of the continuous and the discrete variables. The needed *stochastic* and *mutually independent* attributes of the random operators that regulate the PSO dynamics are restricted in this approach.

2. Updating Discrete Design Variables in MDPSO

In a mixed-discrete optimization scenario, the design space can be divided into a continuous domain and a discrete domain, which correspond to the continuous and the discrete components of the design variable

vector, respectively. Following a continuous search PSO step (Eq. 2), the location of a particle in the discrete domain is defined by a local hypercube that is expressed as

$$\begin{aligned} &H_d = \left\{ \left(x_1^L, x_1^U\right),\ \left(x_2^L, x_2^U\right),\ \ldots,\ \left(x_m^L, x_m^U\right) \right\},\ \text{where} \\ &x_i^L \le x_i \le x_i^U, \quad \forall\ i = 1, 2, \ldots, m \end{aligned} \tag{3}$$

In Eq. 3, m is the number of discrete design variables, and x_i's denote the current location of the candidate solution in the discrete domain. The parameters x_i^L and x_i^U represent two consecutive feasible values of the i^{th} discrete variable that bound the local hypercube. The total number of vertices in the hypercube is equal to 2^m.

The values, x_i^L and x_i^U, can be obtained from the discrete vectors that need to be specified a priori for each discrete design variable. A relatively straight-forward criterion, called the Nearest Vertex Approximation (NVA), is developed to approximate the current discrete-domain location of the candidate solution to one of the vertices of its local hypercube, H_d (Eq. 3). The NVA approximates the discrete-domain location to the nearest vertex of the local hypercube (H_d), on the basis of the Euclidean distance. This approximation is represented by

$$\begin{aligned} &\tilde{X} = \left[\begin{array}{cccc} \tilde{x}_1 & \tilde{x}_2 & \cdots & \tilde{x}_m \end{array} \right], \\ &\tilde{x}_i = \begin{cases} x_i^L\ , & \text{if } \left|x_i - x_i^L\right| \le \left|x_i - x_i^U\right| \\ x_i^U\ , & \text{otherwise} \end{cases} \\ &\forall\ i = 1, 2, \ldots, m \end{aligned} \tag{4}$$

In Eq. 4, $\tilde{X}$ represents the approximated discrete-domain location (nearest hypercube vertex). Another criterion, related to the shortest normal distance between the latest velocity vector of the particle and the neighboring hypercube vertices, was also tested. However, this Shortest Normal Approximation (SNA)[33] was found to be computationally expensive, when applied to a wide range of test problems; hence NVA was selected for universal application in the MDPSO algorithm. An illustration of the NVA and the SNA in the case of a representative 2-D discrete domain are shown in Fig. 2.

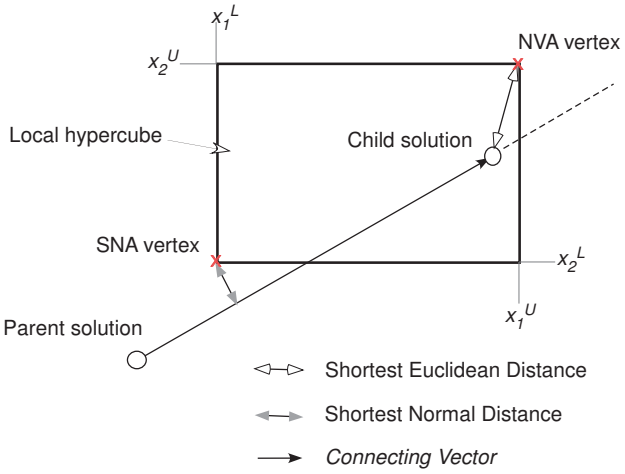


Figure 2. Illustration of the Nearest Vertex and Shortest Normal Approximations

This deterministic approximation seeks to retain the search characteristics of the continuous PSO dynamics, while ensuring that the system-model is evaluated only at the allowed discrete domain locations. Such an approximation strategy can be readily implemented in other non-gradient based continuous optimization algorithms, as a post process to the usual continuous search step at every iteration.

C. Constraint Handling in PSO

1. Literature Survey: Constraint Handling

The basic dynamics of PSO does not account for system constraints. Several variations of the PSO algorithm that incorporate a constraint handling capability have been proposed: (i) a straight-forward method of

considering only feasible particles for global and local best solutions,[34] (ii) the use of conventional dynamic penalty functions,[35] (iii) an effective bi-objective approach where the net constraint serves as the the second objective,[36] and (iv) the use of the efficient constrained non-dominance principles.[37] In this paper, we implement the rules of constrained non-dominance introduced by Deb et al.[10] *Interestingly, the constrained non-dominance principle can be perceived as an aspect of natural swarm intelligence: communication of information from particle to particle regarding whether they are beyond the feasible domain boundaries, and/or how far beyond they are.*

2. Solution Comparison and Constraint Handling in MDPSO

Solution comparison is essential in PSO at every iteration, to determine and update the global best for the population and the local best for each particle. The principle of constrained non-domination[11] is used to compare solutions. This principle has been successfully implemented in other major population-based optimization algorithms. According to this principle, solution-i is said to dominate solution-j if,

i. solution-i is feasible and solution-j is infeasible or,

ii. both solutions are infeasible and solution-i has a smaller net constraint violation than solution-j or,

iii. both solutions are feasible and solution-i weakly dominates solution-j.

In the case of a multi-objective problem, it is possible that none of the above scenarios apply, which implies that the solutions are non-dominated with respect to each other.

The net constraint violation $f_c(X)$ is determined by

$$f_c(X) = \sum_{j=1}^{p} max\left(\bar{g_j}, 0\right) + \sum_{k=1}^{q} max\left(\bar{h_k} - \epsilon, 0\right) \tag{5}$$

where $\bar{g_j}$ and $\bar{h_k}$ represent the normalized values of the j^{th} inequality constraint and k^{th} equality constraint, respectively. In Eq. 5, ϵ represents the tolerance specified to relax each equality constraint. The solution comparison approach in MDPSO favors feasibility over the objective function value. This approach has a tendency to drive solutions towards and into the feasible region during the initial iterations of the algorithm.[17,38] Throughout this initial phase, dominance scenarios I and II are prominently active. When a majority of the particles have moved into the feasible space, scenario III takes over; solution comparisons are then progressively determined by the magnitude of the objective function.

In the case of highly constrained single-objective problems, this solution comparison approach, together with the intrinsic swarm dynamics, can lead to an appreciable loss in diversity. This undesirable phenomenon occurs primarily during the feasibility-seeking process of optimization. To counter this limiting characteristic of the particle motion in the MDPSO, an explicit diversity preservation term, $\gamma_c r_3 \hat{V}_i^t$, is added to the velocity vector, as shown in Eq. 2.

D. Diversity Preservation in PSO

1. Literature Survey: Diversity Preservation

Preservation of the population diversity to avoid premature convergence has been a long standing challenge for PSO. Rapid swarm convergence, which is one of the key advantages of PSO over other population-based algorithms, can however lead to stagnation of particles in a small suboptimal region. Efficient and time-variant parameter selection has been traditionally used as an implicit method to avoid particle stagnation, thereby preserving population diversity. Over the years, the use of explicit diversity preservation techniques have proved to be more effective.[1] Krink et al.[39] introduced a collision-avoidance technique to prevent premature convergence. Particles coming within a defined vicinity of each other were allowed to bounce off; bouncing back along the old velocity vector (U-turn approach) was found to be most effective. Blackwell and Bentley[40] also developed a diversity preserving swarm based on a similar collision-avoidance concept. The collision avoidance schemes however require an intuitive specification of the threshold radius.

A more globally applicable approach was developed by Riget and Vesterstrom,[41] where the usual attraction phase was replaced by a repulsion phase, when the entire population diversity fell below a predefined threshold. In this case, the usual PSO location update formula is applied with the direction reversed. A

modification of the standard deviation of the particle locations was used as the measure of diversity. This measure, however, does not readily account for the combined effects of the distribution of the particles and the overall spread of the particles in the variable space. In other words, with their method,[41] infrequent extreme deviations i.e., a higher kurtosis (e.g., $[0, 0, 0, 0, 10, -10]$) may yield the same measure of diversity as frequent moderate deviations (e.g., $[5, 6, 7, -5, -6, -7]$), which is misleading. Other interesting methodologies to address population diversity include: (i) introduction of a predatory particle,[42] and (ii) introduction of the concept of negative entropy from thermodynamics.[43] *Nevertheless, the consideration of population diversity in a mixed-discrete/combinatorial optimization scenario (in PSO) has rarely been reported in the literature.*

2. Diversity Preservation in MDPSO

The first step in diversity preservation is to characterize and quantify the existing population diversity with respect to the design variable space. A consistent measure of diversity should simultaneously capture the *overall spread* and the *distribution* of solutions in the population. A variable space metric, similar to the performance metric (δ parameter) used to measure the spread of solutions along the computed Pareto front in the objective space,[11] would be an almost ideal choice for diversity characterization. However, the required determination of the *nearest-neighbor Euclidian distances* for every member of the population is likely to become computationally prohibitive in the case of high dimensional optimization problems. The diversity characterization developed in this paper

- seeks to effectively capture the two diversity attributes: overall spread and distribution of particles , and
- is computationally inexpensive to implement, if required, at every iteration.

Separate diversity metrics and diversity preservation mechanisms are formulated for continuous and discrete design variables. The diversity metrics and the corresponding diversity preservation coefficients, γ_c and γ_d, are estimated for the entire population at the start of an iteration. These diversity metrics are then updated using a common factor that seeks to account for the distribution of solutions. In the case of continuous design variables, the initial diversity metric is given by the normalized side length of the smallest hypercube that encloses all the particles. This metric is expressed as

$$D_c = \left(\prod_{i=m+1}^{n} \frac{x_i^{t,max} - x_i^{t,min}}{x_i^{max} - x_i^{min}} \right)^{\frac{1}{n-m}} \tag{6}$$

where $x_i^{t,max}$ and $x_i^{t,min}$ are, respectively, the maximum and the minimum values of the i^{th} design variable in the population at the t^{th} iteration; and x_i^{max} and x_i^{min}, respectively, represent the the specified upper and lower bounds of the i^{th} design variable.

A likely scenario is the presence of one or more outlier particles, when the majority of the particles are concentrated in a significantly smaller region. Occurrence of this scenario leads to an appreciable overestimation of the population diversity (D_c). To overcome this limiting scenario, as well as to account for the distribution of candidate solutions, the diversity metric is further modified. The number of solutions in a λ fraction of the then occupied variable space is first determined. This λ-fractional domain is formed around the global best candidate solution and considers both continuous and discrete variables. The boundaries of this domain are given by

$$\bar{x}_i^{t,max} = max \left[\begin{array}{l} x_i^{t,min} + \lambda \Delta x_i^t \ , \\ min \left(P_{g,i} + 0.5\lambda \Delta x_i^t, \ x_i^{t,max} \right) \end{array} \right], \quad \text{and}$$

$$\bar{x}_i^{t,min} = min \left[\begin{array}{l} x_i^{t,max} - \lambda \Delta x_i^t \ , \\ max \left(P_{g,i} - 0.5\lambda \Delta x_i^t, \ x_i^{t,min} \right) \end{array} \right]$$

$$\forall \ i = 1, 2, \ldots, n$$

where $\Delta x_i^t = x_i^{t,max} - x_i^{t,min}$; the parameters $\bar{x}_i^{t,max}$ and $\bar{x}_i^{t,min}$, respectively, represent the upper and lower bounds of the fractional domain for the design variable x_i; and $P_{g,i}$ is the i^{th} variable of the global best

solution. Using the evaluated "number of particles" in the fractional domain, the continuous diversity metric (enclosing-hypercube side length) is adjusted to better account for the distribution of solutions. The adjusted continuous diversity metric $\bar{D}_c$ is given by

$$\bar{D}_c = \left(\frac{N+1}{N_\lambda+1}\right)^{\frac{1}{n}} \times D_c \tag{7}$$

where N_λ is the number of particles in the λ-fractional domain.

The diversity coefficient, γ_c, for continuous variables is then defined as a function of the continuous diversity metric, which is given by

$$\begin{aligned} \gamma_c &= \gamma_{c0} \exp\left(\frac{-\bar{D}_c^2}{2\sigma_c^2}\right), \quad \text{where} \\ \sigma_c &= \frac{1}{\sqrt{2\ln\left(1/\gamma_{min}\right)}} \end{aligned} \tag{8}$$

and γ_{c0} and γ_{min} are specified constants. The order of magnitude of the diversity-scaling constant γ_{c0} should be one, or in other words comparable to that of the explorative coefficient, β_g. In the range 0 to 1 for $\bar{D}_c$, the diversity coefficient is a monotonically decreasing function. The nature of this function for different orders of magnitude of γ_{min} is shown in Fig. 3.

In the case of discrete design variables, the diversity is characterized independently for each variable. This approach is adopted because of the following two percepts:

i. The effective diversity in the i^{th} discrete variable depends on (1) the available number of feasible values for that variable and (2) the distribution of these feasible values.

ii. Diversity preservation in discrete variables should seek to avoid the stagnation of particles inside a local hypercube H_d.

The initial diversity metric (D_d) for discrete design variables is a vector of the *normalized discrete variable ranges* that span the current population. This metric is expressed as

$$D_{d,i} = \frac{x_i^{t,max} - x_i^{t,min}}{x_i^{max} - x_i^{min}}, \quad \forall\ i = 1, 2, \ldots, m \tag{9}$$

where $D_{d,i}$ is the component of the discrete diversity metric corresponding to the i^{th} discrete variable. Subsequently, in order to better account for the *distribution of solutions*, the discrete diversity metric is adjusted as

$$\bar{D}_{d,i} = \left(\frac{N+1}{N_\lambda+1}\right)^{\frac{1}{n}} \times D_{d,i} \tag{10}$$

where $\bar{D}_{d,i}$ is the adjusted discrete diversity metric. It is important to note how the parameter λ couples the diversity in continuous and discrete design variables. As a result, the diversity preservation mechanisms for continuous and discrete variables are expected to work in coherence with each other.

Diversity preservation for discrete variables is accomplished through modification of the discrete update process described in Section B. The otherwise deterministic approximation of the particle to a nearby feasible discrete location is replaced by a stochastic update process. This stochastic update gives a particle the opportunity to jump out of a local hypercube, thereby seeking to prevent the stagnation of the swarm's discrete component. **A vector of discrete-variable diversity coefficients, γ_d, is defined to further regulate the updating of discrete variables, in order to prevent their premature stagnation.** A random number (r_4) is generated between 0 and 1, and the stochastic update for the generic i^{th} discrete variable (x_i) of a particle is then applied using the following rules:

i. If r_4 is greater than the diversity coefficient $\gamma_{d,i}$, then update the discrete variable using Eq. 4.

ii. If r_4 is less than equal to $\gamma_{d,i}$, then randomly approximate x_i to either x_i^L or x_i^U (defined in Eq. 4).

The discrete-variable diversity coefficient, $\gamma_{d,i}$, that regulates the stochastic update rules is designed to adapt to the size of *the set of feasible values* for the i^{th} discrete variable. This approach avoids a false impression of considerable diversity, in the case of discrete variables that take a relatively small sized *set of feasible values*. The discrete diversity coefficient is defined as

$$\begin{aligned} \gamma_{d,i} &= \gamma_{d0} \exp\left(\frac{-\bar{D}_{d,i}^2}{2\sigma_{d,i}^2}\right), \quad \text{where} \\ \sigma_{d,i} &= \frac{1}{\sqrt{2\ln M_i}} \\ \forall\ i &= 1, 2, \ldots, m \end{aligned} \tag{11}$$

and where M_i represents the size of the *set of feasible values* for the i^{th} discrete variable, and γ_{d0} is a prescribed constant between 0 and 1. For any estimated value of the population diversity, a higher value of the prescribed parameter, γ_{d0}, makes the random update of the discrete domain location more likely.

It is important to note that while the continuous-variable diversity coefficient (γ_c) directly regulates the particle motion (in the location update step), the discrete-variable diversity coefficients ($\gamma_{d,i}$), control the updating of the discrete variables as a post-process (during the NVA application) in every pertinent iteration. In addition, the same value of γ_c is used for all design variables at a particular iteration, whereas a different value of $\gamma_{d,i}$ is used for each i^{th} discrete variable. An illustration of the discrete diversity coefficient for different sizes of the *set of feasible values* is shown in Fig. 3.

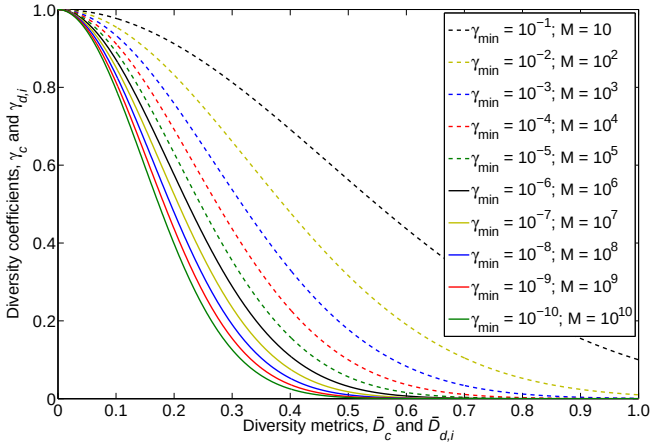


Figure 3. Variation of the diversity coefficients γ_c and $\gamma_{d,i}$ with the diversity metrics $\bar{D}_c$ and $\bar{D}_{d,i}$, respectively, illustrated at (i) different values of γ_{min} for continuous variables, and (ii) different sizes (M) of the feasible set for discrete variables, with $\gamma_{d0} = 1$

III. Numerical Experiments

To validate the Mixed-Discrete Particle Swarm Optimization (MDPSO) algorithm, we apply it to two different classes of single-objective optimization problems: (i) standard unconstrained problems, most of which are multimodal, and (ii) Mixed-Integer Non-Linear Programming (MINLP) problems. These two sets of numerical experiments are discussed in the following sub-sections. The values of the prescribed MDPSO parameters for the two sets of numerical experiments are given in Table 1.

A. Unconstrained Standard Optimization Problems

The new MDPSO algorithm is applied to a set of nine *standard unconstrained nonlinear optimization test problems with only continuous variables* to compare its performances with that of the basic PSO. For a majority of these test problems, the basic PSO is expected to offer a powerful solution. The MDPSO is specifically designed to address complex **constrained and/or mixed-discrete optimization** problems.

Table 1. User-defined constants in PSO

Parameter	Unconstrained problems	MINLP problems
α	0.5	0.5
β_g	1.4	1.4
β_l	1.4	1.4
$\gamma_c 0$	0.1,0.5,1.0	2.0
$\gamma_d 0$	-	0.7
γ_{min}	1.0e-10	1.0e-10
Population Size (N)	$10 \times n$	$10 \times n$
Fractional domain size ($\lambda \times N$)	$0.25 \times N$	$0.1 \times N$
Allowed number of function calls	10,000	50,000

With this set of numerical experiments, we investigate if the additional features in MDPSO, particularly those related to diversity preservation, introduces any unexpected characteristics. The first eight test problems have been taken from the list of sample single-objective optimization problems provided in the MATLAB Genetic and Evolutionary Algorithm Toolbox (GEATbx) Documentation.[20] The GEATbx problems were originally developed and reported by different prominent researchers from the design and optimization community. The last test problem from Table 2 (Miele-Cantrell function) has been taken from the paper by Miele and Cantrell.[21] Details of the standard unconstrained test problems are summarized in Table 2.

Table 2. Standard unconstrained optimization problems

Test problem	Function name	Number of variables	Complexity attribute
1	Rosenbrock's valley	2	long relatively flat valley
2	Rastrigin's function	2	highly multimodal
3	Schwefel's function	2	highly multimodal
4	Griewangk's function	2	highly multimodal
5	Ackley's Path function	2	highly multimodal
6	Michalewicz's function	10	flat regions and multimodal
7	Easom's function	2	mostly flat search space
8	Goldstein-Price's function	2	extensive flat region
9	Miele-Cantrell	4	multimodal

The MDPSO algorithm is applied to each test problem, using three different values of the diversity coefficient scaling constant: $\gamma_{c0} = 0.1, 0.5, 1.0$. Each test problem is run 10 times, with a particular γ_{c0} value, to compensate for the effects of the random operators on the overall algorithm performance. Results for the conventional PSO was obtained simply by specifying the diversity coefficient scaling constant, γ_{c0}, to be zero, while other basic PSO parameters were fixed at the same values as given in Table 1. The convergence histories from representative runs of the MDPSO and a representative run of the conventional PSO for the Miele Cantrell test function are shown in Fig. 4. The actual minimum of the objective function is known for each test problem listed in Table 3. Using the actual minimum objective function value, an additional algorithm termination criterion is specified, based on the normalized relative error (ε_f). This error is evaluated as

$$\varepsilon_f = \begin{cases} \dfrac{|f_{min}^{comp} - f_{min}^{act}|}{f_{min}^{act}}, & \text{if } f_{min}^{act} \neq 0 \\ |f_{min}^{comp} - f_{min}^{act}|, & \text{if } f_{min}^{act} = 0 \end{cases} \tag{12}$$

where f_{min}^{comp} and f_{min}^{act} are the computed minimum and the actual minimum of the objective function,

respectively. The algorithms are terminated when the relative error ε_f falls below 1.0e-10.

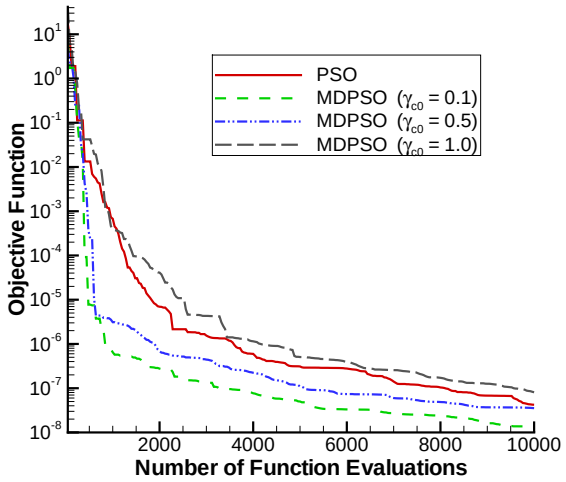
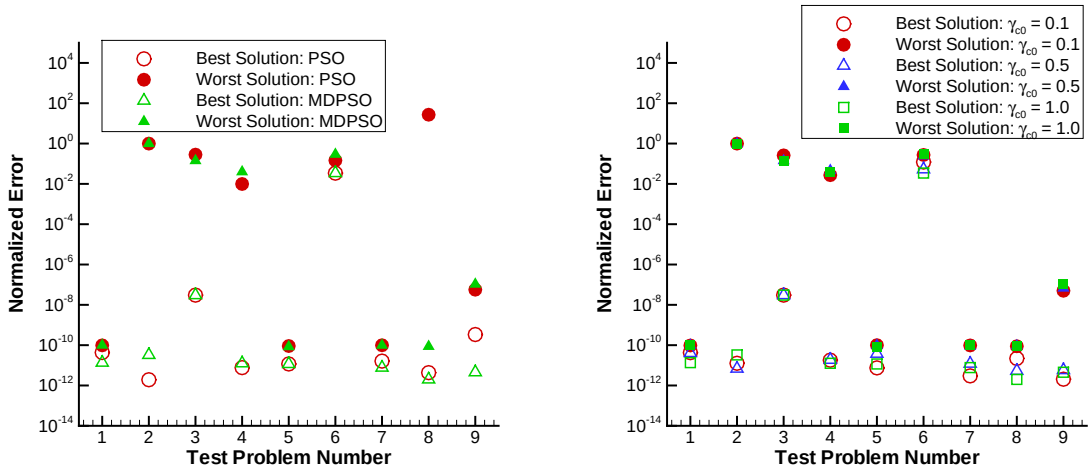


Figure 4. Convergence histories of the MDPSO and the conventional PSO for the Miele-Cantrell function

It can be observed from Fig. 4 that the algorithms perform very well for the multimodal Miele-Cantrell test function. With the diversity scaling constant equal to 0.1 (green dashed line), the rate of convergence of MDPSO is approximately twice that of the conventional PSO - the relative error ε_f reduces to 1.0e-07 in half the number of function calls. With the value of the diversity scaling constant equal to 1.0 (black dashed line), the MDPSO algorithm converges slightly slower than the conventional PSO algorithm. This phenomenon can be attributed to the increased reduction in the particle velocities towards the global optimum, caused by the introduction of a larger amount of population diversity among the particles. The normalized relative errors corresponding to the best and the worst optimized solutions among the 10 runs, obtained for each test problem by MDPSO and conventional PSO are shown in Figs. 5(a) and 5(b). Further details, regarding the performance of the MDPSO algorithm with the diversity scaling constant equal to 1.0, are provided in Table 3.



(a) Using PSO, and MDPSO with $\gamma_{c0} = 1.0$

(b) Using MDPSO with $\gamma_{c0} = 0.1, 0.5, 1.0$

Figure 5. Maximum and minimum values (among 10 runs) of the normalized relative error obtained for the standard unconstrained test problems

Figure 5(a) shows that the MDPSO algorithm performs as good as or better than the conventional PSO

Table 3. Performance of MDPSO (with $\gamma_{c0} = 1.0$) on the standard unconstrained test problems

Test Problem	Actual minimum	Best computed minimum	Worst computed minimum	Standard deviation of the computed minima
1	0.000E+00	1.359E-11	9.951E-11	3.274E-11
2	0.000E+00	3.283E-11	9.950E-01	5.138E-01
3	-8.380E+02	-8.380E+02	-7.195E+02	6.242E+01
4	0.000E+00	1.263E-11	3.946E-02	1.269E-02
5	0.000E+00	1.167E-11	8.373E-11	2.276E-11
6	-9.660E+00	-9.328E+00	-6.843E+00	8.119E-01
7	-1.000E+00	-1.000E+00	-1.000E+00	2.723E-11
8	3.000E+00	3.000E+00	3.000E+00	8.614E-11
9	0.000E+00	4.534E-12	1.054E-07	4.436E-08

algorithm for most of the standard unconstrained test problems, except for test problem 2. It is observed from Fig. 5(a) and Table 3 that, neither of the algorithms could provide satisfactory solutions for test problem 6, Michalewicz's function; this test function has extensive flat regions and is also multimodal.[20] A selective combination of the prescribed MDPSO parameters may successfully find the optimum for such complex nonlinear functions. A detailed parametric analysis of the sensitivity and the variation of the algorithm performance with respect to the prescribed constants is required for this purpose. Figure 5(b) illustrates that the performance of MDPSO is marginally sensitive to the specified value of the diversity scaling constant (γ_{c0}) in the case of these unconstrained continuous problems - the relative errors given by MDPSO for the three different values of the diversity scaling constant are close to each other. The standard deviation in the computed minima obtained from the 10 runs for each test problem (Table 3) is observed to be relatively small when compared to the corresponding actual minima. This observation further illustrates the consistency in the performance of the MDPSO algorithm.

B. Mixed-Integer Nonlinear Programming (MINLP) Problems

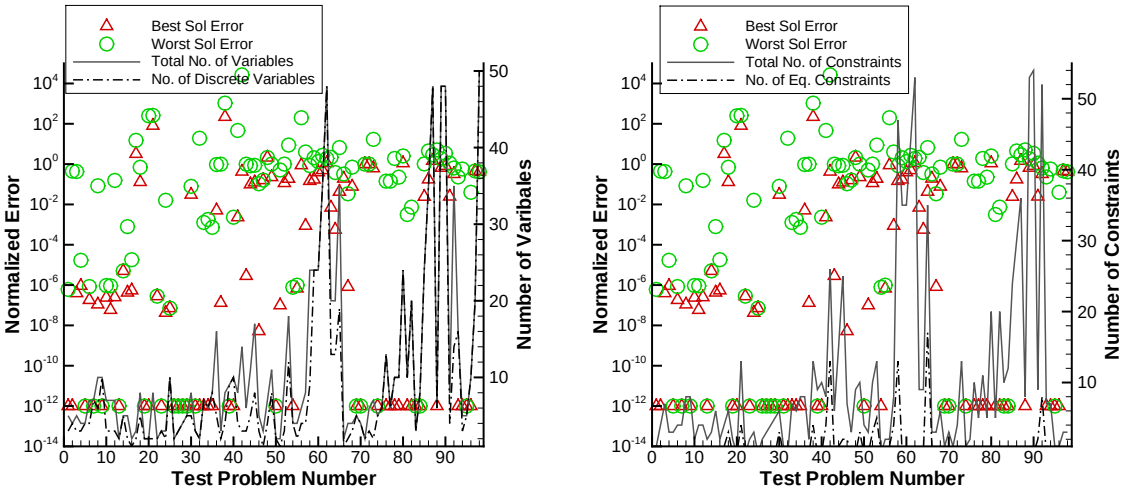
The MDPSO algorithm is applied to an extensive set of ninety-eight Mixed-Integer Non-Linear Programming (MINLP) test problems; these test problems were obtained from the comprehensive list of one hundred MINLP problems reported by Schittkowski.[22] The problems numbered 10 and 100 in the original list[22] have not been tested in this paper. A majority of these MINLP test problems belong to the GAMS Model Library MINLPlib,[44] and have been widely used to validate and compare optimization algorithms.[22] These MINLP test problems present a wide range of complexities:

- total number of design variables vary from 2 to 50;
- numbers of binary design variables and integer design variables vary from 0 to 16 and 0 to 50, respectively;
- total number of constraints (including equality and inequality) vary from 0 to 54;
- number of equality constraints vary from 0 to 17.

Similar to the previous set of numerical experiments, each MINLP test problem is run 10 times to compensate for the performance effects of the random operators in the algorithm. For each run, the algorithm is terminated when the best global solution does not improve by at least 1.0e-06 times its objective value in 10 consecutive iterations.

The normalized relative errors corresponding to the best and the worst solutions among the ten runs obtained for each test problem by MDPSO are illustrated in Figs. 6(a) and 6(b). Figs. 6(a) and 6(b) also show (i) the number of design variables and (ii) the number of constraints in each test problem, to provide insights into their influences on the performance of the algorithm. A histogram of the relative errors is

shown in Fig. 7. *It is helpful to note that, in the case of several MINLP test problems that comprise only discrete variables, a zero relative error is obtained through optimization; the zero error for each of these test problem runs is replaced by an artificial error value of 1.0e-12 in the figures to allow a logarithmic scale representation of the error.*



(a) Showing the *number of design variables* in each problem (b) Showing the *number of constraints* in each problem

Figure 6. Maximum and minimum values (among 10 runs) of the normalized relative error obtained by MDPSO for the MINLP test problems

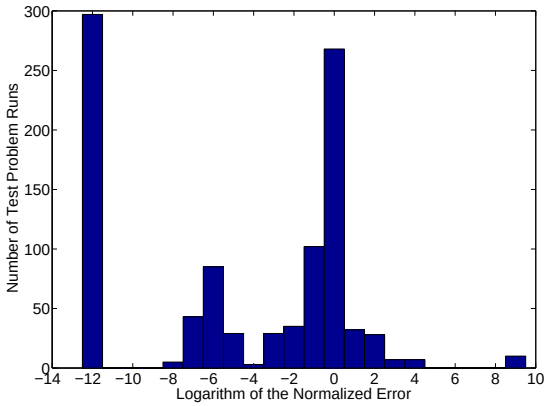


Figure 7. Histogram of the *order of magnitude* of the normalized relative error obtained by MDPSO for the MINLP test problems

From Figs. 6(a) and 6(b), it is expectedly observed that, a higher number of design variables and/or a higher number of constraints generally results in a higher error in the computed minimum. The numbers of design variables and constraints are two key attributes of the complexity of an optimization problem. Future advancements in Mixed-Discrete PSO should focus on reducing the sensitivity of the algorithm performance to the problem complexity attributes. The histogram in Fig. 7 illustrates the distribution of the relative error, on a logarithmic scale, for all the 980 test problem runs, which includes 10 runs for each problem. The same values of the prescribed algorithm parameters have been used for the entire set of MINLP problems. This specification is partly responsible for the high relative error of 10% or more for a significant number of test problem runs (as seen from Fig. 7). The set of MINLP test problems present a wide variety of non-linear criteria functions and problem complexities; better performance for individual problems can be obtained through appropriate alteration of the prescribed MDPSO parameter values based on the problem complexity.

Figure 8 illustrates the net constraint violation (Eq. 5) corresponding to the best and the worst solutions obtained by MDPSO. Figure 9 illustrates the number of function evaluations made by the MDPSO algorithm. Further details, regarding the performance of the MDPSO algorithm, are provided in Tables 4 for MINLP test problems 1 to 50, and 5 for MINLP test problems 51 to 98.

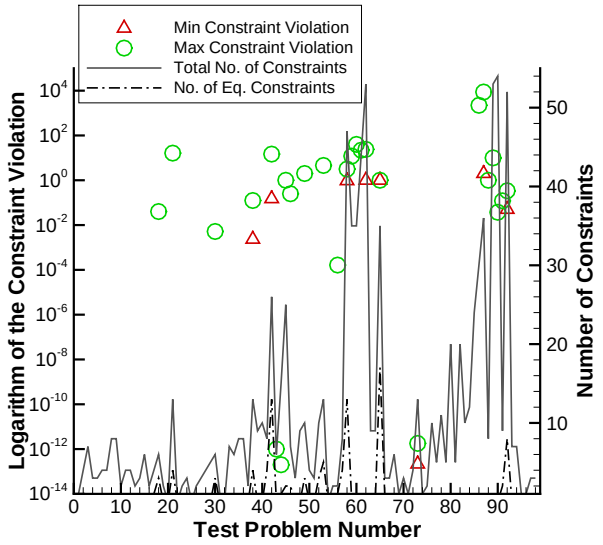


Figure 8. Net constraint violation in the best and the worst solutions obtained by MDPSO for the MINLP test problems

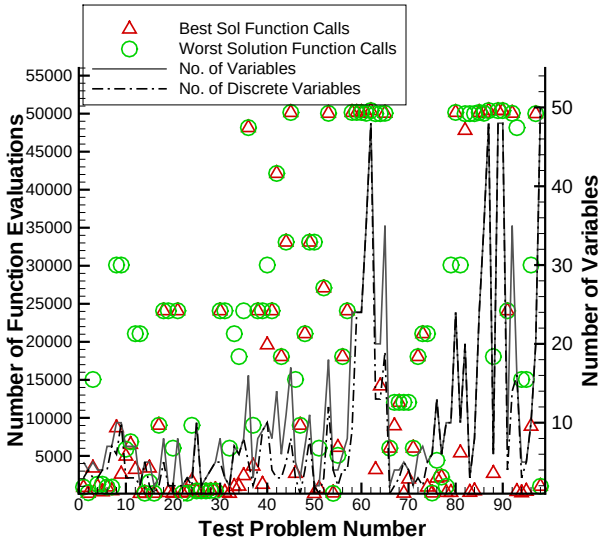


Figure 9. Number of function evaluations made by MDPSO for the MINLP test problems (corresponding to the best and the worst computed solutions)

Figure 8 and the feasibility success % for each problem, listed in Tables 4 and 5, show that the MDPSO algorithm successfully finds the feasible space in a majority of the MINLP test problems. The feasibility success is lower for the test problems that involve 20 or more constraint functions, as seen from Fig. 8. Expectedly, the standard deviations in the computed minima are observed to be higher in the case of the test problems for which optimization resulted in higher relative errors. For complex engineering design applications, the computational expense of optimization is generally dominated by the expense of system-model evaluations. Therefore, in addition to the ability to successfully find the feasible space, and subsequently the optimum design, the number of function evaluations invested in the process is also an important per-

Table 4. Performance of MDPSO on the MINLP test problems numbered 1 to 50

Test Problem	Feasibility success (%)	Actual minimum	Best computed minimum	Worst computed minimum	Standard deviation of the computed minima
1	100	-1.001E+04	-1.001E+04	-1.001E+04	4.216E-03
2	100	-2.000E+01	-2.000E+01	-2.900E+01	3.755E+00
3	100	3.500E+00	3.500E+00	5.001E+00	7.747E-01
4	100	-4.096E+01	-4.096E+01	-4.096E+01	2.268E-04
5	100	-3.800E+01	-3.800E+01	-4.096E+01	0.000E+00
6	100	6.949E+02	6.949E+02	6.949E+02	1.581E-04
7	100	7.000E+02	7.000E+02	6.949E+02	0.000E+00
8	100	3.722E+01	3.722E+01	4.035E+01	9.896E-01
9	100	4.300E+01	4.300E+01	4.035E+01	0.000E+00
10	100	1.000E+00	1.000E+00	1.000E+00	3.162E-07
11	100	-2.718E+00	-2.718E+00	-2.718E+00	7.379E-07
12	100	-8.980E+06	-8.980E+06	-7.589E+06	4.598E+05
13	100	-5.694E+01	-5.694E+01	-7.589E+06	0.000E+00
14	100	1.004E-01	1.004E-01	1.004E-01	1.463E-17
15	100	-3.067E+04	-3.067E+04	-3.069E+04	9.515E+00
16	100	-2.444E+00	-2.444E+00	-2.444E+00	1.588E-05
17	100	1.247E+01	5.277E+01	2.007E+02	5.980E+01
18	60	5.964E+00	6.738E+00	1.013E+01	1.071E+00
19	100	1.600E+01	1.600E+01	1.013E+01	0.000E+00
20	100	7.200E-01	7.200E-01	1.799E+02	6.860E+01
21	30	5.471E+00	4.467E+02	1.434E+03	2.937E+02
22	100	1.770E+00	1.770E+00	1.770E+00	2.341E-16
23	100	4.000E+00	4.000E+00	1.770E+00	0.000E+00
24	100	2.345E+01	2.345E+01	2.383E+01	1.196E-01
25	100	-4.313E+01	-4.313E+01	-4.313E+01	0.000E+00
26	100	-3.108E+02	-3.108E+02	-4.313E+01	5.992E-14
27	100	-4.310E+02	-4.310E+02	-4.313E+01	0.000E+00
28	100	-4.812E+02	-4.812E+02	-4.313E+01	5.992E-14
29	100	-5.852E+02	-5.852E+02	-4.313E+01	1.198E-13
30	20	-4.036E+04	-3.912E+04	-3.717E+04	6.201E+02
31	100	1.000E+00	1.000E+00	-3.717E+04	0.000E+00
32	100	7.031E-01	7.031E-01	1.420E+01	6.521E+00
33	100	-1.100E+03	-1.100E+03	-1.099E+03	6.763E-01
34	100	-7.784E+02	-7.784E+02	-7.770E+02	6.763E-01
35	100	-1.098E+03	-1.098E+03	-1.098E+03	3.373E-01
36	100	2.309E+02	2.321E+02	4.518E+02	6.811E+01
37	100	-5.685E+00	-5.685E+00	-1.616E-02	2.775E+00
38	0	6.058E+00	1.344E+03	6.456E+03	2.017E+03
39	100	-1.125E+03	-1.125E+03	6.456E+03	2.397E-13
40	100	-1.033E+03	-1.033E+03	-1.031E+03	7.589E-01
41	100	1.000E+00	1.002E+00	4.759E+01	1.506E+01
42	0	2.545E-01	1.460E-01	6.791E+03	2.170E+03
43	100	6.010E+00	6.010E+00	1.198E+01	2.289E+00
44	100	7.304E+01	8.029E+01	1.334E+02	1.852E+01
45	80	6.801E+01	7.642E+01	1.272E+02	1.905E+01
46	90	7.667E+00	7.667E+00	8.476E+00	3.815E-01
47	100	1.077E+00	1.250E+00	1.250E+00	0.000E+00
48	100	4.580E+00	1.431E+01	1.431E+01	0.000E+00
49	30	-9.435E-01	-7.294E-01	0.000E+00	3.242E-01
50	100	3.100E+01	3.100E+01	0.000E+00	0.000E+00

Table 5. Performance of MDPSO on the MINLP test problems numbered 51 to 98

Test Problem	Feasibility success (%)	Actual minimum	Best computed minimum	Worst computed minimum	Standard deviation of the computed minima
51	100	-1.700E+01	-1.700E+01	-8.333E+00	4.186E+00
52	100	-1.923E+00	-1.701E+00	0.000E+00	5.380E-01
53	20	8.462E-01	6.869E-01	8.272E+00	2.363E+00
54	100	1.000E+00	1.000E+00	1.000E+00	3.162E-07
55	100	1.363E+00	1.363E+00	1.363E+00	2.341E-16
56	70	1.000E+00	1.887E+00	2.010E+02	7.830E+01
57	100	1.000E+00	1.001E+00	4.994E+00	1.291E+00
58	0	1.446E+05	1.234E+05	2.240E+04	6.954E+04
59	30	1.960E+01	2.300E+01	6.000E+01	1.285E+01
60	80	8.600E+00	1.200E+01	2.000E+01	3.134E+00
61	90	1.030E+01	1.550E+01	3.850E+01	7.169E+00
62	0	1.630E+01	3.450E+01	4.750E+01	3.725E+00
63	100	1.219E+01	1.210E+01	3.848E+01	1.310E+01
64	100	5.315E+00	5.312E+00	7.341E+00	7.073E-01
65	0	-1.430E+00	-1.492E+00	-1.088E+01	2.788E+00
66	100	-2.460E+02	-1.988E+02	-1.665E+02	1.023E+01
67	100	7.198E+03	7.198E+03	7.440E+03	9.671E+01
68	100	2.824E+01	3.041E+01	4.814E+01	5.962E+00
69	100	2.810E+02	2.810E+02	4.814E+01	0.000E+00
70	100	2.000E+00	2.000E+00	4.814E+01	0.000E+00
71	100	-6.000E+00	0.000E+00	0.000E+00	0.000E+00
72	100	-4.574E+03	-8.000E+00	-8.000E+00	0.000E+00
73	90	-3.339E+02	-1.256E+02	5.312E+03	1.580E+03
74	100	-4.500E+03	-4.500E+03	5.312E+03	0.000E+00
75	100	-9.250E+00	-2.147E+10	-2.147E+10	0.000E+00
76	100	-7.000E+00	-7.000E+00	-8.000E+00	4.830E-01
77	100	-7.000E+00	-7.000E+00	-8.000E+00	3.162E-01
78	100	-1.100E+02	-1.100E+02	-3.200E+02	7.696E+01
79	100	4.710E+02	4.710E+02	5.770E+02	5.174E+01
80	100	-2.961E+04	3.371E+03	4.520E+04	1.368E+04
81	100	-1.281E+01	-1.281E+01	-1.277E+01	2.070E-02
82	100	-2.059E+01	-2.059E+01	-2.044E+01	4.367E-02
83	100	-8.050E+01	-8.050E+01	-2.044E+01	0.000E+00
84	100	2.300E+00	2.300E+00	-2.044E+01	4.681E-16
85	100	8.300E+00	8.500E+00	1.680E+01	2.628E+00
86	50	1.030E+01	1.210E+01	5.650E+01	1.746E+01
87	0	1.460E+01	3.590E+01	5.700E+01	6.867E+00
88	90	1.122E+05	1.122E+05	6.801E+05	1.777E+05
89	10	1.630E+01	2.710E+01	4.610E+01	5.964E+00
90	80	4.807E+01	8.768E+01	2.127E+02	3.385E+01
91	70	2.925E+00	2.997E+00	6.200E+00	1.332E+00
92	0	1.419E+01	1.879E+01	2.331E+01	1.578E+00
93	100	1.300E+01	1.300E+01	1.600E+01	9.487E-01
94	100	3.500E+00	3.500E+00	5.500E+00	8.564E-01
95	100	8.000E+00	8.000E+00	5.500E+00	0.000E+00
96	100	3.000E+02	3.000E+02	3.120E+02	3.777E+00
97	100	6.853E-01	1.000E+00	1.000E+00	0.000E+00
98	100	1.713E+00	1.000E+00	1.000E+00	0.000E+00

formance attribute for an optimization algorithm. It is observed from Fig. 9 that most of the MINLP test problems used 25,000 or less number of function evaluations to obtain a relative error smaller than 1.0e-06. The required number of function evaluations is observed to scale with the *variable-space dimensionality* of the problem. Problem specific assignment of the values of the prescribed MDPSO parameters are expected to (i) further advance the probability of finding the feasible space for highly constrained problems, and (i) reduce the required number of system-model evaluations.

IV. Concluding Remarks

In this paper, a modification of the Particle Swarm Optimization (PSO) algorithm is developed, which can simultaneously address system constraints and mixed-discrete variables for single-objective problems. Constraint handling is performed using the constrained non-dominance principle adopted from evolutionary algorithms. The conventional particle motion step at each iteration is followed by approximating the discrete component of the variable vector to a neighboring feasible discrete space location. This approach ensures that, the system model is always evaluated at *feasible discrete variable values* without deviating from the basic search characteristics of the standard particle motion. Stagnation of particles owing to the loss of population diversity has been one of the major limitations of the PSO algorithm; this undesirable search attribute becomes more pronounced in the case of *constrained single-objective mixed-discrete* problems. To address this issue, a new efficient technique is developed to characterize the population diversity at a concerned iteration. Subsequently, in the case of the continuous component of the design variable vector (for each solution), the estimated diversity measure is used to apply a dynamic repulsion towards the global best solution. At the same time, the estimated diversity measure is used apply a stochastic update of the discrete component of the design variable vector for each solution. The generalized diversity measure formulated in this paper can also be used for diversity preservation in other standard population-based optimization algorithms.

The Mixed-Discrete PSO (MDPSO) algorithm performs well, when applied to a set of standard unconstrained problems; a majority of these test functions are multimodal and/or involve extensive flat regions. In order to establish the true potential of the MDPSO algorithm, the algorithm is also be applied to a comprehensive set of ninety-eight MINLP problems. Satisfactory results are obtained, and the algorithm is observed to be particularly successful in driving candidate solutions into the feasible space. It was found that the performance of the MDPSO algorithm (with fixed prescribed parameter values), in terms of the resulting relative error and the computational expense, is sensitive to the number of design variables and the number of constraints. Problem specific assignment of the prescribed MDPSO parameter values can help in obtaining more accurate solutions for such a wide variety of mixed-discrete optimization problems. The MDPSO algorithm is also employed to perform wind farm design, where the selection of turbine-types to be installed (discrete) and the farm layout (continuous) are simultaneously optimized. A remarkable increase in the overall farm power generation (60%) is accomplished, which illustrates the potential of applying the MDPSO algorithm to real life mixed-discrete engineering design problems. Future research in Mixed-Discrete PSO should focus on a comprehensive parametric analysis of the sensitivity of the algorithm performance to the prescribed parameter values. The development of standard guidelines to specify prescribed parameters values based on problem complexity would further enhance the universal applicability of this class of mixed-discrete optimization algorithms.

V. Acknowledgements

Support from the National Science Foundation Awards CMMI-1100948, and CMMI-0946765 is gratefully acknowledged.

References

[1]Kennedy, J. and Eberhart, R. C., "Particle Swarm Optimization," *IEEE International Conference on Neural Networks*, No. IV, IEEE, Piscataway, NJ, USA, April 1995, pp. 1942–1948.

[2]Kennedy, J., Eberhart, R. C., and Shi, Y., *Swarm Intelligence*, Morgan Kaufmann, USA, 1st ed., April 2001.

[3]Banks, A., Vincent, J., and Anyakoha, C., "A review of particle swarm optimization. Part II: hybridisation, combinatorial, multicriteria and constrained optimization, and indicative applications," *Natural Computing*, Vol. 7, No. 1, 2008, pp. 109–124.

[4]Chowdhury, S., Messac, A., and Khire, R., "Comprehensive Product Platform Planning (CP^3) Framework," *ASME*

Journal of Mechanical Design, Special Issue on Designing Complex Engineered Systems in press, 2011.

[5]Chowdhury, S., Zhang, J., Messac, A., and Castillo, L., "Unrestricted Wind Farm Layout Optimization (UWFLO): Investigating Key Factors Influencing the Maximum Power Generation," *Renewable Energy*, Vol. 38, No. 1, February 2012, pp. 16–30.

[6]Banks, A., Vincent, J., and Anyakoha, C., "A review of particle swarm optimization. Part I: background and development," *Natural Computing*, Vol. 6, No. 4, 2007, pp. 467–484.

[7]GamsWorld, *MINLP World*, http://www.gamsworld.org/minlp/index.htm, 2010.

[8]CMU and IBM, *CMU-IBM Cyber-Infrastructure for MINLP*, http://www.minlp.org/index.php, 2010.

[9]Goldberg, D. E., *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley Professional, USA, 1st ed., January 1989.

[10]Deb, K., *Multi-Objective Optimization Using Evolutionary Algorithms*, Wiley, Chichester, UK, March 2009.

[11]Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T., "A Fast and Elitist Multi-objective Genetic Algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, Vol. 6, No. 2, April 2002, pp. 182–197.

[12]Ponsich, A., Azzaro-Pantel, C., Domenech, S., and Pibouleau, L., "Mixed-Integer Nonlinear Programming Optimization Strategies for Batch Plant Design Problems," *Industrial and Engineering chemistry Research*, Vol. 46, No. 3, 2007, pp. 854863.

[13]Ponsich, A., Azzaro-Pantel, C., Domenech, S., and Pibouleau, L., "Mixed-Integer Nonlinear Programming Optimization Strategies for Batch Plant Design Problems," *Chemical Engineering and Processing*, Vol. 47, 2008, pp. 420–434.

[14]Corne, D., dorigo, M., and Glover, F., *New Ideas in Optimisation (Advanced Topics in Computer Science)*, McGraw-Hill, USA, October 1999.

[15]Bonabeau, E., dorigo, M., and Theraulaz, G., *Swarm Intelligence: From Natural to Artificial Systems*, Oxford University Press, USA, 1st ed., September 1999.

[16]Ratnaweera, A., Halgamuge, S. K., and Watson, H. C., "Self-Organizing Hierarchial Particle Swarm Optimizer with Time-Varying Acceleration Coefficients," *IEEE Transactions on Evolutionary Computation*, Vol. 8, No. 3, June 2004, pp. 240–255.

[17]Chowdhury, S. and Dulikravich, G. S., "Improvements to Single-Objective Constrained Predator-Prey Evolutionary Optimization Algorithm," *Structural and Multidisciplinary Optimization*, Vol. 41, No. 4, April 2010, pp. 541–554.

[18]Nakamichi, Y. and Arita, T., "Diversity control in ant colony optimization," *Artificial Life and Robotics*, Vol. 7, No. 4, 2004, pp. 198–204.

[19]Laskari, E. C., Parsopoulos, K. E., and Vrahatis, M. N., "Particle Swarm optimization for Integer Progarmming," *IEEE 2002 Congress on Evolutionary Computation*, IEEE, Hawaii, USA, May 2002.

[20]Pohlheim, H., *GEATbx: Example Functions (single and multi-objective functions) 2 Parametric Optimization*, 2007.

[21]Miele, A. and Cantrell, J. W., "Study on a memory gradient method for the minimization of functions," *Journal of Optimization Theory and Applications*, Vol. 3, No. 6, 1969, pp. 459–470.

[22]Schittkowski, K., *A Collection of 100 Test Problems for Nonlinear Mixed-Integer Programming in Fortran: Users Guide*, Department of Computer Science, University of Bayreuth, Bayreuth, Germany, November 2009.

[23]Shi, Y. and Eberhart, R. C., "Parameter Selection in Particle Swarm Optimization," *Evolutionary Programming VII*, New York, USA, 1998, pp. 591–600.

[24]Trelea, I. C., "The particle swarm optimization algorithm: convergence analysis and parameter selection," *Information Procesing Letters*, Vol. 85, 2003, pp. 317325.

[25]Zhang, W., Li, H., Zhao, Q., and Wang, H., "Guidelines for Parameter Selection in Particle Swarm Optimization According to Control Theory," *Fifth International Conference on Natural Computation*, IEEE, Tianjin, China, August 2009.

[26]Alatas, B., Akin, E., and Ozer, A. B., "Chaos Solutions and Fractals," *Chaos embedded particle swarm optimization algorithms*, Vol. 40, 2009, pp. 1715–1734.

[27]Sobol, M., "Uniformly distributed sequences with an additional uniform property," *USSR Computational Mathematics and Mathematical Physics*, Vol. 16, 1976, pp. 236–242.

[28]Kennedy, J. and Eberhart, R. C., "A discrete binary version of the particle swarm algorithm," *IEEE International Conference on In Systems, Man, and Cybernetics*, Vol. 5, IEEE, 1997, pp. 4104–4108.

[29]Tasgetiren, M. F., Suganthan, P. N., and Pan, Q. Q., "A discrete particle swarm optimization algorithm for the generalized traveling salesman problem," *9th Annual conference on Genetic and evolutionary computation (GECCO 2007)*, New York, USA, 2007.

[30]Jarboui, B., Damak, N., Siarry, P., and Rebai, A., "Applied Mathematics and Computation," *A combinatorial particle swarm optimization for solving multi-mode resource-constrained project scheduling problems*, Vol. 195, 2008, pp. 299–308.

[31]Kitayama, S., Arakawa, M., and Yamazaki, K., "Structural and Multidisciplinary Optimization," *Penalty function approach for the mixed discrete nonlinear problems by particle swarm optimization*, Vol. 32, 2006, pp. 191–202.

[32]Singh, G., Grandhi, R. V., and Stargel, D. S., "Structural and Multidisciplinary Optimization," *Modified particle swarm optimization for a multimodal mixed-variable laser peening process*, Vol. 42, 2010, pp. 769782.

[33]Chowdhury, S., Messac, A., and Khire, R., "Developing a Non-gradient Based Mixed-Discrete Optimization Approach for Comprehensive Product Platform Planning (CP^3)," *13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference*, AIAA, Fort Worth, September 2010.

[34]Hu, X. and Eberhart, R. C., "Solving constrained nonlinear optimization problems with particle swarm optimization," *Sixth world multiconference on systemics, cybernetics and informatics (SCI),*, Orlando, FL, USA, 2002.

[35]Hu, X. and Eberhart, R. C., "Particle swarm method for constrained optimization problems," *Euro-international symposium on computational intelligence*, Kosice, Slovakia, 2002.

[36]Venter, G. and Haftka, R. T., "Structural and Multidisciplinary Optimization," *Constrained particle swarm optimization using a bi-objective formulation*, Vol. 40, 2009, pp. 65–76.

[37]Zavala, A. E. M., Diharce, E. R. V., and Aguirre, A. H., *Particle evolutionary swarm for design reliability optimization*, Vol. 3410, chap. Evolutionary multi-criterion optimization. Third international conference, EMO 2005, Springer, Guanajuato, Mexico, March 2005, p. 856869.

[38]Chowdhury, S., Dulikravich, G. S., and Moral, R. J., "Modified Predator-Prey Algorithm for Constrained and Unconstrained Multi-objective Optimisation," *International Journal of Mathematical Modelling and Numerical Optimisation*, Vol. 1, No. 1/2, 2009, pp. 1–38.

[39]Krink, T. and an J Riget, J. S. V., "Particle swarm optimisation with spatial particle extension," *2002 IEEE Congress on Evolutionary Computation*, Vol. 2, IEEE Computer Society, Honolulu, HI, USA, 2002.

[40]Blackwell, T. M. and Bentley, P., "Don't push me! Collision Avoiding Swarms," *2002 IEEE Congress on Evolutionary Computation*, Vol. 2, IEEE Computer Society, Honolulu, HI, USA, 2002.

[41]Riget, J. and Vesterstrom, J. S., "A Diversity-Guided Particle Swarm Optimizer the ARPSO," Tech. Rep. 2002-02, EVALife Project Group, Department of Computer Science, Aarhus Universitet, 2002.

[42]Silva, A., Neves, A., and Costa, E., "An empirical comparison of particle swarm and predator prey optimization," *13th Irish international conference on artificial intelligence and cognitive science*, Vol. 2464, Lmerick, Ireland, 2002, pp. 103–110.

[43]Xie, X. F. and Yang, W. J. Z. Z. L., "A dissipative particle swarm optimization," *2002 IEEE Congress on Evolutionary Computation*, Vol. 2, IEEE Computer Society, Honolulu, HI, USA, 2002, pp. 1456–1461.

[44]Bussieck, M. R., Drud, A. S., and Meeraus, A., "MINLPLib - A Collection of Test Models for Mixed-Integer Nonlinear Programming," Tech. rep., GAMS Development Corp, Washington, D.C. 20007, USA, 2007.