Revista de Matemática: Teoría y Aplicaciones 2017 **24**(1) : 157–188
cimpa – ucr issn: 1409-2433 (Print), 2215-3373 (Online)

SOLVING ENGINEERING OPTIMIZATION PROBLEMS WITH TABU/SCATTER SEARCH

RESOLVIENDO PROBLEMAS DE OPTIMIZACIÓN EN INGENIERÍA CON BÚSQUEDA TABÚ/DISPERSA

Ricardo P. Beausoleil[*]

Received: 14/Oct/2014; Revised: 8/Aug/2016; Accepted: 12/Aug/2016

[*]Departmento de Matemática Interdisciplinaria, Instituto de Cibernética Matemática y Física. La Habana, Cuba. E-Mail: rbeausol@icimaf.cu, rbeausol0531@gmail.com

Abstract

This paper introduces an adaptation of a multiobjective tabu/scatter search to deal with nonlinear discrete, mixed-integer constrained engineering optimization problems. The problem is reduced to a bi-objective problem (the objective function and the constraint violation function). This approach eliminates the use of penalties for constraint handling. Its performance was proved with different standard engineering optimization problems, including mathematical function minimization and structural engineering. The results show that the proposed method performs well in terms of efficiency and robustness.

Keywords: multiple objectives; metaheuristics; engineering optimization.

Resumen

Este artículo introduce una adaptación de una Búsqueda Tabú/ Dispersa multiobjetivo para problemas de ingeniería con restricciones no lineales con valores discretos y enteros-mixtos. El problema es reducido a un problema bi-objetivo, (la función objetivo y la función de violación de las restricciones). Este enfoque elimina el uso de penalidades para manipular las restricciones. El desempeño del algoritmo fue probado con diferentes problemas conocidos de la ingeniería, incluyendo algunas funciones de optimización matemática e ingeniería estructural. Los resultados muestran que el método propuesto trabaja bien en términos de eficiencia y robustez.

Palabras clave: múltiples objetivos; metaheurísticas; optimización en la ingeniería.

Mathematics Subject Classification: 90C27.

1 Introduction

Many engineering applications, such as structural optimization and engineering design, involve difficult optimization problems that must be solved efficiently and effectively. The nature of these applications usually need to be constrained in specific parts of the search space that are delimited by linear and/or nonlinear constraints.

Different exact as approximation algorithms have been developed for solving constrained optimization problems. Exact approaches such as sequential quadratic programming methods and generalized reduced gradient methods [35], [12], [7] are inflexible to adapt the solution algorithm to a given problem. On the other hand, approximation optimization algorithms such as the evolutionary algorithm proposed by Fogel et al. [14], De Jong [10], Koza [25] and the genetic algorithm (GA) proposed by Holland [22] and Goldberg [19], human behaviour (e.g., tabu search proposed by Glover [16]), and the physical annealing process (e.g., simulated annealing proposed by Kirkpatrick *et al.* [24]) have been successfully applied for tackling constrained optimization problems during the the last four decades. Recently, Geem *et al.* [15] developed a new harmony search (HS) meta-heuristic algorithm that was conceptualized using the musical process of searching for a perfect state of harmony. These different approaches have been applied to engineering optimization problems. Engineering design is one of the scientific fields in which constrained optimization problems frequently arise. These types of problems normally have mixed (continuous, integer and discrete) design variables, nonlinear objective functions and nonlinear constrains. Constrains are very important in engineering design problems, they are usually imposed in the statement of the problems and sometimes are very hard to satisfy, which makes the search difficult and inefficient.

The organization of the paper is as follows. Related work is presented in section 2. In the Section 3 the problem formulation and sets are presented. In the Section 4 the parameter settings are presented. Computational experiments are presented in Section 5. Results in Section 6. Section 7 contains conclusions. In the appendix all problems are presented.

2 Related work

Engineering optimization problems has been tackled by different approaches. S. Akhtar *et al.* developed a socio-behavioural simulation model [1], M.G. Sahab *et al.* a hybrid genetic algorithm for structural optimization problems [31], K.S. Lee and Z.W. Geem a harmony search [26], K.E. Parsopoulos and M.N. Vrahatis a particle swarm optimization [28], K.H. Raj *et al.* an evolutionary computational technique [29], Z.W. Geem *et al.* [15] a harmony search, M. Mahdavi *et al.* an improved harmony search algorithm [27], A.R. Yildiz a hybrid a Taguchi-harmony search algorithm [36], also, M. Fesanghary *et al.* used a hybrid harmony search algorithm [13], L.C. Cagnina *et al.* a simple constrained particle swarm [6], A. Kaveh and S. Talatahari a hybrid particle swarm and ant colony optimization [23].

We compare our approach with all these different approaches. To our knowledge, this is one of the first attempts at applying the tabu/scatter search technique to engineering optimization problems.

3 Problem formulation and sets

We are interested to find

$$x = (x_1, x_2, \ldots, x_n, y_{n+1}, y_{n+2}, \ldots, y_{n+l})$$

such that

$$\texttt{Minimize}\ \{f_0 : f_0 = f(x)\}$$

under the following constrains: $\Im = \{x : g_j(x) \leq 0, j = \{1, 2, \ldots, m\}\}$, where $x_i \in X_i \subset \Re$, $y_{n+k} \in Y_k \subset Z$ and Z the integer set, where $i \in \{1, 2, \ldots, n\}$, $k \in \{1, 2, \ldots l\}$.

The problem is again formulated as follows:

$$\texttt{Minimize}\ \{f_0, f_1\}$$

where

$$f_1 = \sum_{j=1}^{m} \max(0, g_j(x))$$

consequently, we apply a Pareto-base multiobjective tabu/scatter search.

During the development of our text we use the following notation:

S	a set of trial solutions, from which all other sets derive.
P	an approximate Pareto set, containing all non-inferior solutions of S.
R	a diverse set of solutions subset of P, called the reference set.
D	a set, consisting of considered duplicated solutions.
C	a selected set of the set R.
Ω	a set of generator points, created from a given set C.
b	the larger size of the trial solutions set.
$b1$	the larger size of the reference set.
$b2$	the larger size of the selected set.

Constrained mixed integer tabu/scatter search

This is a hybrid method that, in the tabu search phase creates restrictions to prevent moves toward solutions that are "too close" to previously visited solutions. A sequential fan candidate list strategy is used to explore solution neighbourhoods. A weighted linear function is used to aggregate the objective function values. The weights used are modified in a way that a sufficient variety of points can be generated. These solutions are later used as reference points in the scatter phase.

In order to guide the search more quickly to better regions, as in MOSS-II (see [4]) we embed, in the tabu phase, linear combinations by joining the new solutions with point of current set of efficient points. The proposed algorithm incorporates a new strategy in the scatter phase to improve the search toward the nondominated front. New points are generated by re-starting from the tabu search phase, taking as starting points solutions of the current efficient solutions set.

The following update to our TS/SS (MOSS) strategy is performed:

1. The algorithm works with continuous variables rounding the integers variables to the closest integers.

2. Frequency memory is used to control the deterministic selection of the sub-ranges where the variables take values. The sub-ranges are sorted in ascending order taking into account the frequency distribution of the visited sub-ranges, choosing the less visited sub-range in a diversification strategy and the most visited sub-range to intensify the search.

3. A new strategy to change the value of the radius of the ball in the decision space is introduced.

4. The Kramer Choice Function is changed by a new choice method.

5. A modified strategy to create subsets of new solutions to be combined is introduced.

6. A new procedure to change the bounds of the sub-ranges is introduced.

7. In order to restart, we choose starting points using the choice method proposed.

3.1 The reference set

The reference set R is a subset of the trial solution set S that consists of an approximation to the Pareto-optimal set.

Let $P \subseteq S \backslash D \cup R$, consists of a subset of current efficient solutions. If $|P| > b1$, then we use the max-min criterion and a parameter ϵ as measure of the closeness, to obtain a diversified collection of solutions $P^{'}$, that is, set one first element of P into $P^{'}$, then let $x \in P$ maximizes the minimum distance $d(f(x), f(y(i)))$ for $i \leq |P^{'}|$ (d is the Euclidean distance) and hold the condition $d(f(x), f(y(i))) \geq \epsilon,\ y(i)$ pertaining to the non-empty set $P^{'}$ then, when $P^{'} = b1$ set $P = P^{'}$.

3.2 Critical event design

The critical even design has been introduced within the context of our MOSS algorithm [3], it is based on a design that monitors the current solutions in R and in the trial solutions set S to control the proximity of the points in these sets. The elements considered in the design are the values of the objectives and the decision variables.

Let $B(p, \xi)$ be a set of points within distance ξ from p, $\xi > 0$. We call $B(p, \xi)$ a ball with center p and radius ξ. Then, for any point $p \in f[S \cup R]$ we define a ball $B(p, \rho)$ with $0 < \rho \leq 1$, and for any point $p^{'} \in S \cup R$ a ball $B(p^{'}, \delta)$ with $0 < \delta \leq 1$.

We say that two points are near ("duplicated"), if a trial solution satisfies a full "critical condition". A "critical condition" is "full" if it is satisfied and the trial solution belongs to $B(p^{'}, \delta)$ defined on $S \cup R$. The "critical condition" is satisfied if the image of the trial solution pertains to $B(p, \rho)$ defined on $f[S \cup R]$.

In this new approach we define a new strategy to give values to parameter δ, it is explained in 4.8. This design can be seen as tabu-distance restrictions that permits dynamically diversify and intensify the search.

3.3 Tabu search phase

We use a Multi-Start TS as a generator of diverse solutions. This approach can be seen as a sequence of tabu searches where each TS has its own starting point, recency memory, and aspiration threshold; they all share the frequency memory to bias the search to unvisited or less visited regions and also to a most visited regions.

3.3.1 Candidate list strategy and move

As in our MOSS, we use a simplified version of a sequential fan strategy as a candidate list strategy. The sequential fan generates p best alternative moves at a given step, and then creates a fan of solution streams, one for each alternative. The best available moves for each stream are again examined, and only the p best moves overall provide the p new streams at the next step. In our case, taking $p = 1$, we have in each step one stream and a fan of solutions denoted by fan is considered.

In this approach we propose moves that consist of changing one or at most five selected variables. We use a frequency memory to select the move-variables. The range of variables $U_i - L_i$ are split into sub-ranges $u_{ij} - l_{ij}$ of equal size, j represents the index of the sub-ranges, $j \in \{1, 2 \ldots, s\}$. The move $x^{'} = m(x)$ is defined as follows:

1. if a new best solution was reached in a previous iteration then, set $x^{''}$ equal to the best solution found; in other case $x_i^{''} = l_{ij} + \alpha(u_{ij} - l_{ij})$ where $\alpha = itabu/fan$ and $itabu$ is the current iteration value in the tabu phase.

2. set $x_i^{'} = x_i + \exp(-\alpha)(x_i^{''} - x_i)$.

Now we explain how to transit to a new solution. Our implementation uses as move attribute variables that change their values as result of the move. The change is represented by a difference of values $z_k^* - f_k(x^{'})$ $(\forall k = 0, 1)$, $x^{'} = m(x)$, x is a current solution and Z^* is a reference solution (or aspiration level) $Z^* = (z_0^*, z_1^*)$, denoted by *aspiration_level* in our pseudo_code. Without lost generality, let us assume that every criteria is minimized. Let $\Delta f(x^{'}) = (\Delta f_0(x^{'}), \Delta f_1(x^{'}))$ where $\Delta f_k(x^{'}) = z_k^* - f_k(x^{'})$, $k \in \{0, 1\}$.

Let E the set of efficient moves and D the set of deficient moves, where a deficient move is a move that not satisfies the aspiration level, in otherwise the move is efficient.

Definition 3.1 *The best efficient move* m^* *is defined as a move such that [*$m \in E(x) : c^* = \min\{c(x^{'}), x^{'} = m(x)\}$*] where* $c\,(.)$ *is the cost function defined for the problem.*

In each iteration if $E(x) \neq \emptyset$ the solution transit to $x^{'} = m^*(x)$ such that $m^* \in E$, in other case $x^{'} = m^*(x)$ and $m^* \in D$.

A move is efficient if $(\exists \Delta f_k(x^{'}) \geq 0)$ *or* $(\forall k \in \{0, 1\}[\Delta f_k(x^{'}) = 0])$ qualifying to be introduced in S, otherwise is deficient.

The point Z^* is updated by $z_k^* = \min f_k(x^{'}) \forall k \in \{0, 1\}$, $x^{'} \in S$.

In order to measure the quality of the solution we propose to use in our tabu search approach an additive function U with weighting coefficients $\lambda_k \geq 0$ $(k = 0, 1)$, representing the relative importance of the objectives. Each component in the weight vector is set according to the objective function values. The influence is given by an exponential function $\exp(-s_k)$, where s_k is obtained as $s_k = \left|(z_k^* - f_k(x^{'}))\right| / |z_k^*|$, $\lambda_k = 2 - \exp(-s_k)$ $(k \in \{0, 1\})$, then

$$U(x^{'}) = \begin{cases} (1 - \theta) \sum_{k=0,1} \lambda_k \Delta f_k(x^{'}) & \text{if } \textit{residence}_{ij} > T_i \\ \sum_{k=0,1} \lambda_i \Delta f_k(x^{'}) & \text{otherwise} \end{cases}$$

where *residence*$_{ij}$ is the number of time that the sub-range j has been visited by the variable with index i, the parameter $\theta = \frac{\textit{freq}}{\textit{freqtotal}}$, $freq$ is an addition of the entries of type $residence$ associated to the selected variables and sub-ranges that hold the condition *residence*$_{ij} > T_i$ and *freqtotal* $= \sum_i freq_i$. We would have for each variable a threshold $T_i = \max\{1, \textit{round}\ (\sum_{i=1}^{s} \textit{freq}_i/s)\}$, where *round* is the closest integer.

Notice that, if $\Delta f_k(x^{'})$ is positive then, the term produces an inducement, in otherwise a penalty. If $z_k^* = 0$ we take account only the numerator of s_k.

Candidate denotes the procedure to create the neighbourhood of the current solution x, using the candidate list and the move that have been described above.

Candidate

```
1. for $i = 1$ to *fan*
2.   Choose the variables
3.   $x^{'}$ = m($x$)
4.   if aspiration_level and not is tabu then set $x^{'}$ into $S$
5. endfor
6. Choose $x^* \in S$
```

3.3.2 Tabu restrictions

Tabu restrictions are imposed to prevent moves that bring the values of variables "too close" to values they held previously [17].
The implementation of this rule is as follows: the variable $x^{'}$ is excluded from falling inside the line interval bounded by $x - w(x^{'} - x)$ and $x + w(x^{'} - x)$, when a move from x to $x^{'}$ is executed, where $1 \geq w > 0$.

3.3.3 Tabu routine

The Tabu routine creates new solution vectors taking account the aspiration level function, the additive function value and a penalty function that modifies the value of the additive function. Tabu restrictions are used to guide the search. The parameters *nonefficientmove* identifies the maximum number of time that a non-efficient move is accepted, *efficientmove* identifies if a move has generated a solution accepted as trial solution and introduced in the set S.

Tabu Routine

```
1. *nonefficientmove* $= 0$
2. while *nonefficientmove* $< b_3$
3.   *efficientmove* = false
4.   $x^*$ = Candidate
5.   Update the tabu list
6.   Set $x = x^*$
```

```
7.  if $efficientmove$ then
8.      $nonefficientmove = 0$
9.      $movefive$ = false
10. else
11.     $nonefficientmove = nonefficientmove + 1$
12.     $movefive$ = true
13. endwhile
```

3.3.4 Multi-start tabu procedure

In our implementation the parameter *numcomb* denotes the number of tabu iterations necessary to apply linear combination with a set of parameters $\{wc_i : i = 1, \ldots, 10\}$, *startpoints* is the number of initial points, *delta* (δ) is the parameter that measure the proximity of two points in the decision space, and *movefive* identifies if one or at most five variables are moves. In this version, we introduce a new mechanism to deal with integer optimization problems, where *integers* identifies that the solving problem have integer variables.

MultiStartTabu Procedure

```
1.  $icomb = 1,\ itabu = 0,\ \rho = 0.2,\ movefive =$true
2.  repeat
3.    Choose the subranges
4.    $itabu = itabu + 1$
5.    Reset tabu list
6.    Set the reference point
7.    Tabu Routine
8.    if $itabu = icomb * numcomb$ then Linear Combination
9.    $icomb = icomb + 1$
10.   if $integers$ then round the integer variables of $x \in S$
11.   Translate R to S
12.   Choose the nondominated points of $S$
13.   Update R
14. until $itabu = startpoints$
```

3.4 Method choice

The method begins calculating the score for each solution. The next step is to sort the solutions in ascending order taking into account the score calculated. The third step is to choose a subset of this ordered solutions.

Here, $y_1^{**},\ y_1^*$ denote the maximum and the minimum value of the function $f_1(x^i)$ for all solution x^i in the set R.

1. Set the decision matrix $DM = \| y_{ij} \|$ where $y_{ij} = f_j(x^i)$, x^i belongs to a reference set of solutions R and $f_j(x^i)$ is the value of the objective j $(j = 0, 1)$ for every $x^i \in R$ $(i = 1, 2, \ldots, |R|)$.
2. Calculate the score of each solution of DM as following: $score_i = |(y_{i1} - y_1^*)/(y_1^{**} - y_1^*)|$.
3. Set in increasing order, taking into account the score calculated, a collection of b_2 solutions from R into C.

3.5 Choosing the best solution

Assume we minimize the objective function, and we choose a set $C \subset R$ applying our choice method explained above. Let $f_0^* = \min\{f_0(x)|f_1(x) = 0, x \in C\}$ and $f_1^* = \min\{f_1(x)|f_1(x) > 0, x \in C\}$ then, the best solution is denoted by f^* and defined as

$$f^* = \begin{cases} f_0^* & \text{if } \exists\ x \in C : f_1(x) = 0 \\ f_1^* & \text{otherwise.} \end{cases}$$

Therefore, the current best solution is the best f^* reached so far.

The following routine identifies this process. The parameter *newsol* identifies if an improved solution was reached in the current iteration of the algorithm, *cardC* is the cardinal of C, $x^i \in C$, *bestsol* is a record containing all attributes of the best solution, *prevnewsol* denote the best solution in the previous global iteration.

Choice_procedure

```
  newsol= false
  Choose solutions/*to apply the choice method*/
  for i = 1 to cardC
    if ($f_0(x^i) <$ bestsol.$f_0$) and (bestsol.$f_0 = 0$)
    or ($f_1(x^i) <$ bestsol.$f_1$) and (bestsol.$f_1 > 0$)
      then prevnewsol.x $=$ bestsol.x
      prevnewsol.f $=$ bestsol.f
      bestsol.x $= x^i$
      bestsol.f $= f(x^i)$
    newsol = true
  endfor
```

3.6 Scatter search phase

This phase is designed to intensify the search around the best solution found so far. This is done by creating new trial solutions which consist of weighted linear combination of a new set of points pertaining to the set Ω (defined below).

3.6.1 Generating trial points

In the scatter search phase the intensification scheme is based on a projection method [18] that fix some variables to move and the others free to change. In this case, a set Ω is created and a new set of trial solutions are generated as follows: To create a set of vectors around the best solution x^*, that we call *generator points*, we define the set $\Omega = \{(y^{'}, x^*) | y^{'} = \theta_1 x^* + \theta_2 y\}$ where $y \in C$ and $y^{'}$ is a *generator point*.

Now we explain the strategy to create new trial solutions: a memory that records the four best feasible solutions more recently achieved is defined, then the variables whose values have not changed (the consistent variables) are fixed and the rest are free to change (we consider that no change in the values of the variables has occurred, if the variation between the considered variables is less than 0.1). Then, for all pair $(y^{'}, x^*) \in \Omega$ a new trial point $x^{'}$ is generated as weighted linear combination of $(y^{'}, x^*)$, that is, $x^{'} = y^{'} + \gamma(x^* - y^{'})$, where $\gamma = \alpha \exp(-|x^* - y^{'}|)$ and $\alpha = \alpha_0 + h\Delta$ $(h = 1, \ldots, \widehat{s})$, Δ is the step of variation. Notice that if $y^{'}$ is close to x^* then $\gamma \longrightarrow \alpha$ in otherwise $\gamma \longrightarrow 0$. Therefore, $S = \Big\{x^{'} : x^{'} = y^{'} + \gamma(x^* - y^{'}) \forall (y^{'}, x^*) \in \Omega\Big\}$.

3.6.2 Scatter procedure

We use the following parameters: $iscatter$ denote the current iteration in the scatter phase and $maxi$ the maximum number of scatter iteration.

```
**Sc**atter Procedure
1. *iscatter* $= 0$
2. $\rho = 1$
3. while *maxi* $>$ *iscatter* or *newsol* do
4.    Choose a subset $C$ of efficient solutions from $R$
5.    Generate trial points from $\Omega$ and put into $S$
6.    if *integers* then round the integer variables of $x \in S$
7.    *iscatter* $=$ *iscatter* $+ 1$
8. end of while
```

3.7 Starting points

The StartingPoints procedure generates systematically the seed points to be used by tabu phase. Here the parameter *start_points* denote the number of starting points.

StartingPoints procedure

```
1. $start\_points = initial\_points$
2. for $i = 1$ to $start\_points$
3.     $rr_i = (i-1)/(start\_points - 1)$
4.     $s^i = L_i + rr_i * (U_i - L_i)$
5. endfor
```

The SelecNewPoints procedure is used to rebuild the initial set of reference points and restart the search from tabu phase.

SelectNewPoints procedure

```
1. $start\_points = cardC$
2. for $i = 1$ to $start\_points$
3.     $s^i = x^i : x^i \in C$
4. endfor
```

3.8 Changing the radius of the ball

The parameter *delta* denote the radius of the balls in the decision space, *newsol* is true if in the current scatter iteration a new improved solution is reached, false in otherwise. The user parameter *delta* is initialized and halving its current value when *bestsol* has not improved, where *deltamax*, *deltamin* are the corresponding ones upper and lower bounds.

1. if *newsol* then *delta* = (*deltamin-delta*)*0.5.

2. else *delta* = (*deltamin-delta*)*0.5.

3.9 Changing the bounds of the sub-ranges

The maximum and the minimum value of each variable for the current selected set of non-dominated solutions are calculated. Then, for each variable defined in the range $[L_i, U_i]$ the lower bound of the first sub-range is set equal to L_i, and the upper bound of the last sub-range is set equal to U_i. Next, the upper bound of the first sub-range is set equal to the minimum value of the corresponding variable, and the lower bound of the last sub-range is equal to the maximum of the corresponding variable also. A number of equal-sized sub-ranges are formed between these two boundaries.

This mechanism permits intensify the search in the region defined by the selected current efficient solutions. Nevertheless, the extreme sub-ranges (the first and the last sub-ranges) permit to escape of the narrow region.

3.10 Cutoff rule

Here, *iglob* denote the number of global iterations, that is, the number of times that the two phases (tabu, scatter) have occurred, and *maxiglob* is the largest global iteration: if $(\mathit{deltamax} - \mathit{delta} \leq 1E - 15)$ or $(\mathit{bestsol.f} - \mathit{prevnewsol} \leq 1E - 5)$ or $(\mathit{iglob} = \mathit{maxiglob})$ then stop.

3.11 General scheme of the new TS/SS

We present a general description of our new version of our TS/SS [3] that we call "MITS" approach for nonlinear mixed integer optimization problems.

1. Generate systematically the seed points
2. Multi-Star Tabu Search to generate diverse efficient solutions
3. Choose a subset of efficient solutions and identify the best solution so far
4. Scatter Search to generate new efficient solutions
5. Choose a subset of efficient solutions and identify the best solution so far
6. Change the radius of the ball in the decision space
7. Change the bounds of the sub-ranges
8. Restart to generate new efficient points
9. If the cutoff rule holds then stop otherwise go to 2

4 Parameter settings

A set of seven standard problems and three variants of these, for a total of ten problems, was chosen to evaluate the performance of our proposed approach. Numerous previous numerical experiments have been done in order to find proper values of these parameters.

4.1 Global parameters

The approach uses in the tabu and scatter phases, the following parameters: *s*=12 number of sub-ranges, $b = 450$ largest size of the trial solutions set S, $b1 = 200$ largest size of the reference set R, *initial_points=100* numbers of initial starting points, *maxiglob*=100 number of maximum global iterations, $\mathit{delta} \in [\mathit{deltamin}, \mathit{deltamax}]$ radius of the ball in the decision space, $\mathit{deltamin} = 0.1 \times 10^{-5}$ and $\mathit{deltamax} = 1$, $\epsilon = 0.01$ parameter to distribute the solutions in the set P.

4.2 Tabu phase parameters

Our approach uses in the tabu phase the following parameters: $\mathit{fan} = 60$, $\mathit{numcomb} = \mathit{starpoints}/15$ number of tabu iteration to apply linear combination, $b_3 = 3$ maximum number of tabu iteration without efficient move, $\rho \in [0.1 \times 10^{-5}, 0.2]$ radius of the ball in the objective space, $wc \in \{\frac{1}{2}, \frac{1}{3}, \frac{2}{3}, \frac{3}{4}, \frac{4}{5}, \frac{9}{10}, \frac{7}{6}, \frac{6}{5}, -\frac{7}{6}, -\frac{6}{5}\}$ parameters for linear combination, *w*=0.01 tabu distance.

4.3 Scatter phase parameters

In the scatter phase the following parameters are use: $b_2 = 20$ largest size of the selected set C, $(\theta_1, \theta_2) \in \{(0.8, \pm 0.2), (0.9, \pm 0.2), (1.1, \pm 0.2), (1.2, \pm 0.2), (0.8, 0), (1.2, 0)\}$ the weights to create the generator points, $\rho \in [0.1 \times 10^{-5}, 1]$ radius of the ball in the objective space, $\alpha_0 = 1$, $\Delta = 0.1$ and $\widehat{s} = 6$ parameters to generator points.

5 Computational experiments

In this section we show the performance of the new version of our MOSS described above on a number of single constrained mixed integer engineering problems. The experiments were ran in a AMD Phenom II $\times 4$ 940 Processor (3.00 GHz) 32-bit operating system, and programmed in DELFI-6.

5.1 Contribution of the scatter phase

To evaluate the contribution of the scatter phase was selected a subset of the engineering problems ran in our computational experiment. In these experiments, the only main factor is the proposed approach with two levels: tabu phase and the tabu/scatter phases. Both the two proposed phases shared the same cut-off rule of our approach.

Table 1: Contribution of the scatter phase. Problems: C = Cantilever, P = Pressure Vessel (6 inequalities), Si = Speed Reducer (integer-discrete variables), Sc = Speed Reducer(continuous-integer variables).

Problem	TS	TS/SS
C	1.26397	1.15080
P	31789.82978	7197.73412
Si	3397.54157	2922.43527
Sc	6308.59874	2996.39519

Table 1 shows that the introduction of the scatter phase produces a reduction in the value of the objective functions as follows: for the Cantilever problem 8.95%, for the Pressure Vessel 77.36%, for the Speed Reducer (integer-discrete variables) 13.98%, and Speed Reducer (continuous-integer variables) 52.5%. Is evident that the scatter phase contributes significantly in the quality of the solution.

5.2 Convergence study

To evaluate the convergence of the approach the same subset of the above problems were chosen. The following figures show the trend of the objective function of the chosen problems. We can see that the proposed approach converges quickly, and in all cases a feasible solution is obtained after the first iteration. The objective function of each chosen problem decreases monotonically, and a good approximation is obtained very quickly.

5.3 Examples

First we show an example taken from MATLAB and solved by a gradient technique implemented in MATLAB-7.

We wish optimize the following problem:

$$\min_x f(x) = \exp(x_1)(4x_1^2 + 2x_2^2 + 4x_1x_2 + 2x_2 + 1)$$

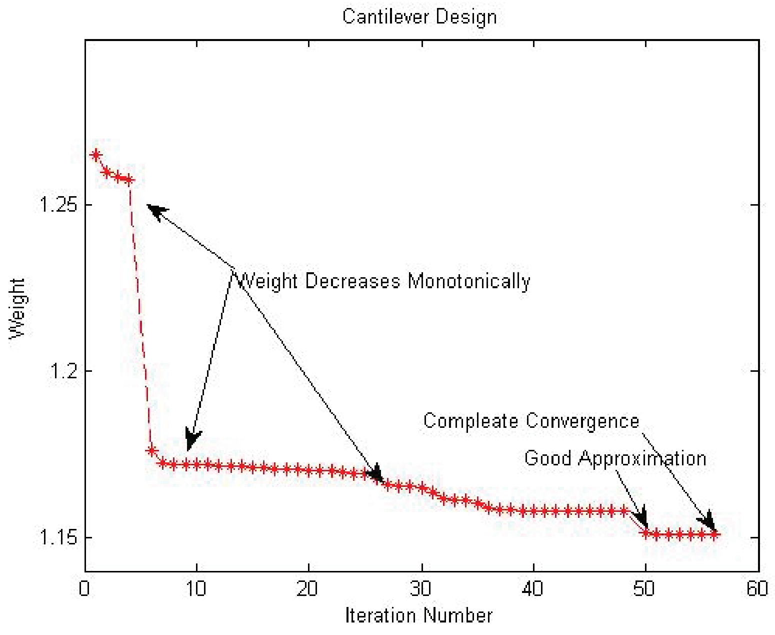
s.t.

$$x_1x_2 - x_1 - x_2 \leq -1.5$$

$$x_1x_2 \geq -10$$

$$-100 \leq x_1 \leq 10$$

$$0 \leq x_2 \leq 100.$$



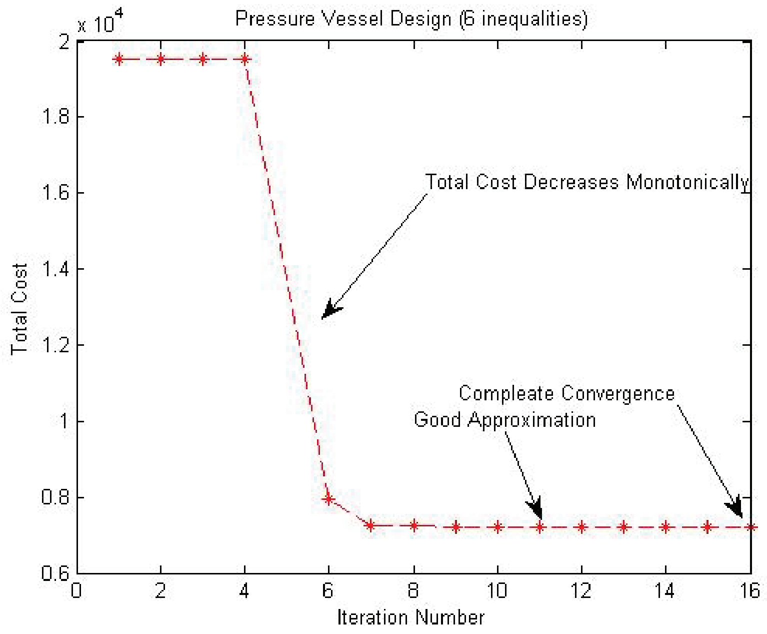
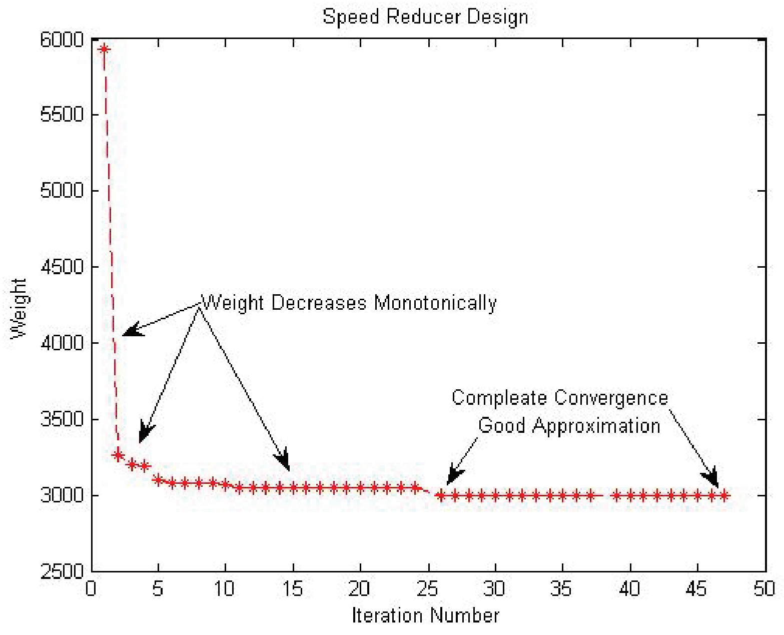


Figure 1: For the Cantilever problem a good approximation is reached after 50 iterations and a minimum after 56 iterations. For the Pressure Vessel problem a good approximation is reached after 11 iterations and a minimum after 15 iterations.



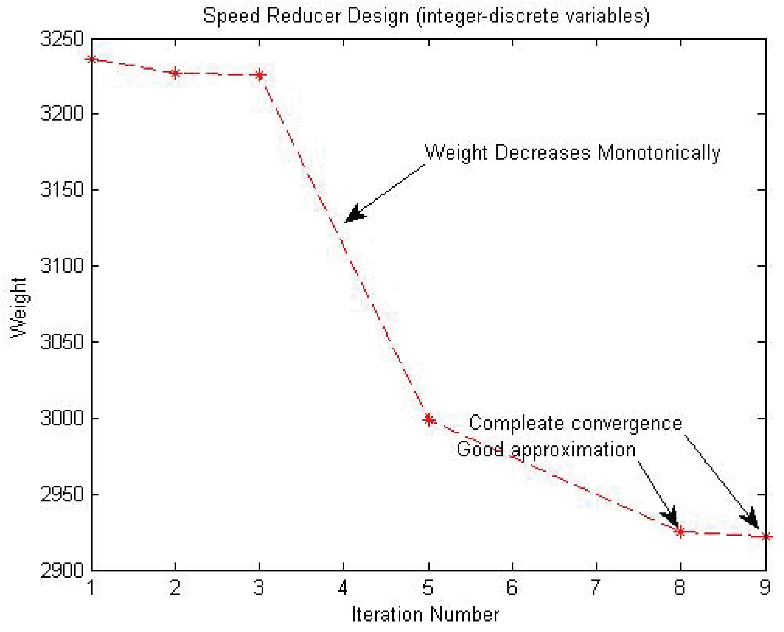


Figure 2: For the Speed Reducer (continuous-integer variables) problem a good approximation is reached after 27 iterations and a minimum after 47 iterations. For the Speed Reducer (integer-discrete variables) problem a good approximation is reached after 8 iterations and a minimum after 9 iterations.

The result reached by MATLAB-7 is $f(x^*) = 0.023698$ our result is $f(x^*) = 0.02355037$, $x^* = (-9.54740502507853481, 1.04740502510698826)$.

5.4 Engineering optimization problems

A set of 10 engineering design optimization problems was chosen to evaluate the performance of our proposed algorithm. We performed 5 independent runs per problem, one run for each fixed initial value of the radius of the ball defined in the decision space. Let $delta_0$ denote the initial radius, $delta_0 \in \{0.3, 0.35, 0.4, 0.45, 0.5\}$. The best results achieved were compared with respect to the best results reported in the specialized literature. The value averages here does not make sense because the best value reached will be always achieved with the fixed $delta_0$. The following tables show the comparison with other approaches. For homogeneity, in our reported solutions only at most the first six decimals are represented in the tables. More details about the test problems and the reached results may be consulted in the appendix at the end of this paper.

Table 2: Best results for the Cantilever design. The initial value $delta_0$ was fixed to 0.35. We compare our approach with the following methods: Generalized Convex Approximation Method GCA(I) and GCA(II) [8].

	Best solution found		
Variables	MITS	CGA(I)	CGA(II)
x_1	5.808324	6.01	6.01
x_2	2.882334	5.304	5.304
x_3	4.215829	4.49	4.49
x_4	3.446026	3.498	3.498
x_5	2.089881	2.15	2.15
$f(x)$	1.1508	1.34	1.34

Table 3: Best results for the Two-bar Truss design. The initial value $delta_0$ was fixed to 0.3. We compare our approach with the following methods: Generalized Convex Approximation Method GCA(I) and GCA(II) [8].

	Best solution found		
Variables	MITS	CGA(I)	CGA(II)
x_1	1.412742	1.41	1.41
x_2	0.374721	0.377	0.377
$f(x)$	1.50867	1.51	1.51

Table 4: Best results for the Three-bar Truss design. The initial value $delta_0$ was fixed to 0.4. We compare our approach with the following methods: Evolutionary Computational Technique (ECT) [29], Hernendez [21], Ray and Saini [30].

	Best solution found			
Variables	MITS	[29]	[21]	[30]
x_1	0.792714	0.789	0.788	0.795
x_2	0.396942	0.405	0.408	0.395
$f(x)$	263.9077	263.896	263.900	264.30

Table 5: Best results for the Welded Bean design. The initial value $delta_0$ was fixed to 0.45. The character (*) means infeasible solution. With our calculations and for these reported solutions, in the Cagnina *et al.* method [6] the constraints $\{g2 = -0.0927002, g3 = -1e-06, g7 = -0.0559377\}$ do not hold, $\{g1 = -0.064633, g2 = -0.1037782, g7 = -0.7577933\}$ are violated in the Fesanghary *et al.* [13].

	Best solution found			
Var.	MITS	[6]	[13]	[27]
x_1	0.205864	0.205730	0.20572	0.20573
x_2	3.463344	3.470489	3.47060	3.47049
x_3	9.047746	9.036624	9.03682	9.03662
x_4	0.205864	0.205729	0.20572	0.20573
$f(x)$	1.727036	1.724852*	1.7248*	1.724855

Table 6: Best results for the Tension/Compression Spring design. The initial value $delta_0$ was fixed to 0.3. The character (*) means infeasible solution. With our calculations and for these reported solutions, in the Cagnina *et al.* method [6] the constraint $\{g2 = 2.1812280341e-05\}$ does not hold, the constraint $\{g2 = 0.013670727\}$ is violated in the Mahdavi *et al.* [27] and the corresponding function value $f(x) = 0.01288$.

	Best solution found			
Var.	MITS	[6]	[27]	[2]
x_1	0.050447	0.051690	0.051154	0.05339
x_2	0.327464	0.356750	0.349871	0.39918
x_3	13.239984	11.287126	12.07643	9.18540
$f(x)$	0.012700	0.012665*	0.01267*	0.01273

Table 7: Best results for the Pressure Vessel design (four inequalities). The initial value $delta_0$ was fixed to 0.4. The character (*) means infeasible solution. With our calculations and for these reported solutions, (**) in the Mahdavi *et al.* [27] the variable x_4 is out of range, and in A. Kaveh and S. Talataharib [23] the constraint $\{g_1 = 9.88238e - 05\}$ is violated.

	Best solution found		
Var.	MITS	[27]	[23]
x_1	0.875	0.75	0.8125
x_2	0.4375	0.375	0.4375
x_3	45.336672	38.86010	42.103566
x_4	140.25502	221.36553	176.57322
$f(x)$	6090.5393	5849.76169**	6059.0925*

Table 8: Best results for the Pressure Vessel design (six inequalities). The initial value $delta_0$ was fixed to 0.45. Our approach is compared with Wu and Chow [34], Sandgren [32], Lee and Geem [26].

	Best solution found			
Var.	MITS	[34]	[32]	[26]
x_1	1.125	1.125	1.125	1.125
x_2	0.625	0.625	0.625	0.625
x_3	58.29007	58.1978	48.97	58.2789
x_4	43.69312	44.2930	106.72	43.7549
$f(x)$	7197.734	7207.494	7980.894	7198.433

Table 9: Best results for the Speed Reducer design (continuous-integer variables). The initial value $delta_0$ was fixed to 0.5. The character (*) means infeasible solution. With our calculations and for these reported solutions, in the Cagnina *et al.* method [6] the constraint $\{g5 = 5.9646629715e-07, g6 = 1.3037925261e-07\}$ do not hold, and in the Lee and Geem [26] the constraint $\{g6 = 1.3037925261e-07\}$ is violated.

	Best solution found		
Var.	MITS	[6]	[26]
x_1	3.500026	3.500000	3.500000
x_2	0.700005	0.700000	0.700000
x_3	17.0	17.0	17.0
x_4	7.300229	7.300000	7.300000
x_5	7.800022	7.800000	7.800000
x_6	3.350215	3.350214	3.350215
x_7	5.286699	5.286683	5.286683
$f(x)$	2996.39519	2996.34816*	2996.34816*

Table 10: Best results for the Speed Reducer design (integer-discrete variables). The initial value $delta_0$ was fixed to 0.4. The character (*) means infeasible solution. With our calculations and for these reported solutions, in the Ching-Long Su and Shutan Hsieh [9] the constraint $\{g5 = 0.21147567552\}$ is violated. Also, we compare with Singiresu *et al.* method [33].

	Best solution found			
Var.	MITS	[33]	[9]	Remarks
x_1	3.3	3.5	3.5	discrete
x_2	0.7	0.7	0.7	discrete
x_3	17.0	17.0	17.0	integer
x_4	7.3	7.3	7.3	discrete
x_5	7.8	7.8	7.8	discrete
x_6	3.36	3.36	3.35	discrete
x_7	5.29	5.29	5.29	discrete
$f(x)$	2922.4352	3000.83	2998.27*	

Table 11: Computational time in seconds for the solved problems. Problems: C = Cantilever, Tw = Two-bar Truss, Th = Three-bar Truss, Tc = Tension/Compression Spring, Pf = Pressure Vessel(four inequalities), Ps = Pressure Vessel(six inequalities), Sc = Speed Reducer(continuous-integer variables), Si = Speed Reducer(integer-discrete variables).

Problem	Time (secs)
C	231
Tw	30
Th	138
Tc	56
Pf	77
Ps	45
Sc	196
Si	10

6 Results

From Table 2 to Table 11 the results obtained by our approach and the comparison with other approaches are shown. For the MATLAB problem the result is 0.63% lightly better than the best solution found by the method of gradient employed by MATLAB-7. For the Cantilever design problem the result is 14.1% better than the best result found. For the Two-bar Truss design problem the result is 0.09% lightly better than the best result found. The result for the Three-bar Truss design problem is 0.004% lightly worse than the best result found by Raj *et al.* (ECT) [29]. The result obtained for the Welded Beem design by our approach is 0.13% worse than the result reported by Mahdavi, Fesanghary and Damangir [27], the other results are infeasible solutions with our calculations. For the Tension/Compression Spring design the result is 0.23% lightly better than the best result given by Arora [2], and the other results are infeasible solutions. Pressure Vessel design (four inequalities) problem presents the better result, because with our calculation and for the results reported by the other researchers, these are infeasible solutions. For Pressure Vessel design (6 inequalities) the approximation was 0.01% lightly better than the best solution found by Lee and Geem [26]. Our approach for the Speed Reducer design (continuous-integer variables) problem shows the better solution because the other are infeasible solutions with our calculation. The result for Speed Reducer design (integer-discrete variables) is 2.6% better than result reported by Singiresu *et al.* [33].

In general our algorithm obtains good approximations in an average time of 120.3 seconds.

7 Conclusions

Our approach seems to be an available strategy to solve non-linear constrained mixed-integer optimization problems. New mechanisms were incorporated in our TS/SS permitting to solve efficiently engineering design problems. From the practical point of view, the user can change the user-parameter to initialize the radius of the ball in the decision space and obtain different results. We can conclude that our approach is competitive with the state-of-the-art algorithms for constrained engineering optimization problems.

References

[1] Akhtar, S.; Tai, K.; Ray, T. (2002) "A socio-behavioural simulation model for engineering design optimization", *Eng. Opt.* **34**: 341–354.

[2] Arora, J.S. (1989) *Introduction to Optimum Design*. McGraw-Hill, New York.

[3] Beausoleil, R. (2006) "MOSS Multiobjective scatter search applied to non-linear multiple criteria optimization", *European Journal of Operational Research* **169**: 426–449.

[4] Beausoleil, R. (2008) "MOSS-II Tabu/Scatter Search for nonlinear multiobjective optimization", in: Z. Michalewicz & P. Siarry (Eds.) *Advance in Metaheuristics for Hard Problems. Chapter 3. Natural Computing Series*, Springer Verlag.

[5] Belegundu, A.D. (1982) "A study of mathematical programming method for structured optimization", R.I. Dept. of Civil and Environment Engineering of Iowa, Iowa.

[6] Cagnina, L.C.; Esquivel, S.; Coello, C.A. (2008) "Solving engineering optimization problems with the simple constrained particle swarm optimizer", *Informatica* **32**: 319–326.

[7] Chen, T.Y.; Cheng, Y.L. (2008) "Global optimization using hybrid approach", *WSEAS Transactions on Mathematics* **7**: 254–262.

[8] Chickermane, Chang, H.; Gea, H. (1996) "Structural Optimization using a New Local approximation", *International Journal for Numerical Method in Engineering Method* **39**: 829–846.

[9] Ching-Long, S.; Shutan, H. (2008) *Latest Trends on Computers* (Volume II).

[10] De Jong, K. (1975) *Analysis of the Behaviour of a Class of Genetic Adaptive Systems*. Ph.D. Thesis, University of Michigan, Ann Arbor MI.

[11] Deb, K. (1991) "Optimal design of a weld beam via genetic algorithm", *AIAA Journal* **29**(11): 2013–2015.

[12] Ettaouil, M.; Loqman C. (2008) "Constraint satisfaction problems solved by semidefinite relaxations", *WSEAS Transactions on Computers* **7**: 951–961.

[13] Fesanghary, M.; Mahdavi, M.; Minary-Jolandan, M.; Alizadeh, Y. (2008) "Hybridizing harmony search algorithm with sequential quadratic programming for engineering optimization problems", *Computer Methods in Applied Mechanics and Engineering* **197**: 3080–3091.

[14] Fogel, L.J.; Owens, A.J.: Walsh, M.J. (1966) *Artificial Intelligence Through Simulated Evolution*. John Wiley, Chichester UK.

[15] Geem, Z.W.; Kim, J.H.; Loganathan, G.V. (2001) "A new heuristic optimization algorithm: harmony search", *Simulation* **76**(2): 60–68.

[16] Glover, F. (1977) "Heuristic for integer programming using surrogate constraints", *Decision Sci.* **8**(1): 156–166.

[17] Glover, F. (1994) "Tabu search for nonlinear and parametric optimization (with links to genetic algorithms", *Discrete Applied Mathematics* **40**: 231–255.

[18] Glover, F. (2005) "Adaptive memory projection methods for integer programming", in: Rego C. & Alidee B. (Eds.) *Metaheuristics Optimization Via Memory and Evolution: Tabu Search and Scatter Search*. Kluwer Academic Publishers.

[19] Goldberg, D.E. (1989) "Genetic Algorithms in Search, Optimization and Machine Learning". Addison Wesley, Boston MA.

[20] Golinski, J. (1973) "An adaptive optimization system applied to machine synthesis", *Mech. Mach. Theory* **8**(4): 419–436.

[21] Hernández, S. (1994) "Multi-objective structural optimisation", in: S. Kodiyalam & M. Saxena (Eds.) *Geometry and Optimisation Techniques for Structural Design*, *Elsevier Applied Science*: 341–363.

[22] Holland, J.H. (1975) *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor MI.

[23] Kaveh, A.; Talatahari, S. (2009) "Engineering optimization with hybrid particle swarm and ant colony optimization", *Asian Journal of Civil Engineering (Building and Housing)* **10**(6): 611–628.

[24] Kirkpatrick, S.; Gelatt, D.; Vecchi, M.P. (1983) "Optimization by simulated annealing", *Science* **220**: 671–680.

[25] Koza, J.R. (1990) "Genetic programming: a paradigm for genetically breeding populations of computer programs to solve problems", Rep. No. STAN-CS-90-1314, Stanford University, Palo Alto CA.

[26] Lee, K.S.; Geem, Z.W. (2005) "A new meta-heuristic algorithm for continuous engineering optimization: Harmony search theory and practice", *Computer Methods in Applied Mechanics and Engineering* **194**: 3902–3933.

[27] Mahdavi, M.; Fesanghary, M.; Damangir, E. (2007) "An Improved Harmony Search Algorithm for Solving Optimization Problems", *Applied Mathematics and Computation* **188**: 1567–1579.

[28] Parsopoulos, K.E.; Vrahatis, M.N. (2005) "Particle swarm optimization for solving constrained engineering optimization problems", in: L. Wang, K. Chen, & Y.S. Ong (Eds.) *ICNC 2005, LNCS 3612*: 582–591.

[29] Raj, K.H.; Sharma, R.S.; Mishra, G.S.; Dua, A.; Patvardhan, C. (2005) "An evolutionary computational technique for constrained optimisation in engineering design", IE (I) *Journal.MC* **86**.

[30] Ray, T.; Saini, P. (2001) "Engineering design optimisation using a swarm with intelligent information sharing among individuals", *Engineering Optimisation* **33**: 735–748.

[31] Sahab, M.G.; Toropov, V.V.; Ashour, A.F. (2004) "A hybrid genetic algorithm for structural optimization probems", *Asian Journal of Civil Engineering (Building and Housing)* **5**(3-4): 121–143.

[32] Sandgren, E. (1990) "Nonlinear integer and discrete programming in mechanical design optimization", *J. Mech. Des. ASME* **112**: 223–229.

[33] Singiresu, S.; Ying, X. (2005) *Journal of Mechanical Design* **127**(6): 1100–1112.

[34] Wu, S.J.; Chow, P.T. (1995) "Genetic algorithms for nonlinear mixed discrete-integer optimization problems via meta-genetic parameter optimization", *Engineering Optimisation* **24**: 137–159.

[35] Yeniay, O. (2005) "A comparative study on optimization methods for the constrained nonlinear programming problems", *Mathematical Problems in Engineering Hindawi Publishing Corporation*: 165–173.

[36] Yildiz, A.R. (2008) "Hybrid Taguchi-harmony search algorithm for solving engineering optimization problems", *International Journal of Industrial Engineering* **15**(3): 286–293.

A Appendix

A.1 Cantiliver design problem [8]

The Cantilever beam is made of five elements, each having a hollow cross-section with constant thickness. The beam is rigidly supported as shown, and three is an external vertical force acting at the free end of the cantilever. The weight of the beam is to be minimized while assigning an upper limit on the vertical displacement of the free end.

The design variables are the heights (or widths) x_i of the cross-section of the each element. The lower bounds on the these design variables are very small and the upper bounds very large so they do not become active in the problem. The problem is formulated as follows:

$$Minimize\ \ f(x) = 0.0624(x_1 + x_2 + x_3 + x_4 + x_5)$$

s.t.

$$g_1(x) = \frac{61}{x_1^3} + \frac{37}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} \leq 1.0$$

$$1 \leq x_i \leq 10, i \in \{1, 2, 3, 4, 5\}.$$

Solution found:

$$x^* = (5.80832436167656592, 2.88233457568314051, 4.21582930749505342,$$

$$3.44602689729287517, 2.08988145846961546).$$

$$F^* = 1.150805547878516.$$

A.2 Two-bar truss design problem [8]

The two-bar truss problem consist of two design variables: a sizing variable x_1 which is the cross-sectional area of the bars and the configuration variable x_2 representing half the distance between the lower nodes. An external force, $|F| = 200kN$, $F_y = 8F_x$, acts on node 3 and the objective is to minimize the weight of the truss while keeping the tensile or compressive stress in each bar below $100N/mm^2$. The problem is formulated in closed form as:

$$Minimize\ \ f(x) = x_1\sqrt{1+x_2^2}$$

s.t.

$$g_1(x) = 0.124\sqrt{1+x_2^2}\left(\frac{8}{x_1}+\frac{1}{x_1x_2}\right) \leq 1.0\ (bar1),$$

$$g_2(x) = 0.124\sqrt{1+x_2^2}\left(\frac{8}{x_1}-\frac{1}{x_1x_2}\right) \leq 1.0\ (bar2),$$

$$0.2 \leq x_1 \leq 4.0,\ \ 0.1 \leq x_2 \leq 1.6.$$

Solution found:

$$x^* = (1.41274204233180889, 0.37472108515071976)$$

$$F^* = 1.508670852887466.$$

A.3 Three-bar truss design problem [29]

The three-bar truss problem consist of two design variables: The volume of the truss structure is to be minimized subject to stress constraints. The problem is formulated as:

$$Minimize\ \ f(x) = \left(2\sqrt{2}x_1 + x_2\right\}L$$

s.t.

$$g_1(x) = \left(\frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2}\right)P \leq 2,$$

$$g_2(x) = \left(\frac{1}{x_1 + \sqrt{2}x_2}\right)P \leq 2,$$

$$g_3(x) = \left(\frac{2}{\sqrt{2}x_1^2 + 2x_1x_2}\right)P \leq 2,$$

where $0 \leq x_1 \leq 1$ and $0 \leq x_2 \leq 1$. The other constants are $L = 100$cm, $P = 2kN/cm^2$.
Solution found:

$$x^* = (0.79271422810570653, 0.39694263279557871).$$

$$F^* = 263.90770577419977.$$

A.4 Welded beam design problem [11]

A welded beam design optimization problem, which is often used for the evaluation of optimization methods, is used to illustrate the implementation procedure of the proposed approach for solving optimization problems. The beam has a length of 14 in. and P=6,000 lb force is applied at the end of the beam. The welded beam is designed for minimum cost subject to constraints on shear stress, bending stress in the beam, buckling load on the bar, end deflection of the beam, and side constraints. The design variables are thickness of the weld $h(x_1)$, length of the weld $l(x_2)$, width of the beam $t(x_3)$, and thickness of the beam $b(x_4)$. The mathematical model of the welded beam optimization problem is defined as

$$Minimize\ \ f_w(x) = 1.1047x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2)$$

s.t.

$$\begin{gathered}
g_1(x) = 13,600 - \tau(x) \geq 0,\\
g_2(x) = 30,000 - \sigma(x) \geq 0,\\
g_3(x) = x_4 - x_1 \geq 0,\\
g_4(x) = 0.10471(x_1^2) - 0.04811x_3x_4(14.0 + x_2) + 5.0 \geq 0,\\
g_5(x) = x_1 - 0.125 \geq 0,\\
g_6(x) = 0.25 - \delta(x) \geq 0,\\
g_7(x) = P_c(x) - 6,000 \geq 0,\\
0.1 \leq x_1,\ x_2 \leq 5\\
0.1 \leq x_3,\ x_4 \leq 10.
\end{gathered}$$

The terms $\tau(x)$, $\sigma(x)$, $P_c(x)$, $\delta(x)$ are given below

$$\tau(x) = \sqrt{(\tau')^2 + (2\tau'\tau'')\frac{x_2}{2R} + (\tau'')^2}$$

$$\tau'(x) = \frac{6000}{\sqrt{2}x_1x_2}$$

$$\tau''(x) = \frac{6000(14 + \frac{x_2}{2})\sqrt{0.25(x_2^2) + ((x_1 + x_3)/2)^2}}{2[x_1x_2\sqrt{2}(x_2^2/12 + 0.25(x_1 + x_3)^2)]}$$

$$\sigma(x) = \frac{504,000}{x_3^2x_4}$$

$$\delta(x) = \frac{65,856,000}{(30 \times 10^6)x_4x_3^3}$$

$$P_c(x) = \frac{4.013(30 \times 10^6)\sqrt{\frac{x_3^2x_4^6}{36}}}{196}\left(1 - \frac{x_3\sqrt{\frac{30\times10^6}{4(12\times10^6)}}}{28}\right).$$

Solution found:

$$x^* = (0.20586359479354222, 3.46334453602819113,$$

$$9.04774674254588592, 0.20586428001618971).$$

$$F^* = 1.727036254666027.$$

A.5 Weight tension/compression spring problem [2] [5]

This problem minimizes the weight of a tension/compression spring, subject to constraints of minimum deflection, shear stress, surge frequency, and limits on outside diameter and on design variables. There are three design variables: the wire diameter x_1, the mean coil diameter x_2, and the number of active coils x_3. The mathematical formulation of this problem is:

$$Minimize\ \ f(x) = (x_3 + 2)x_2x_1^2$$

s.t.

$$g_1(x) = 1 - \frac{x_2^3x_3}{71785x_4^4} \leq 0,$$

$$g_2(x) = \frac{4x_2^2 - x_1x_2}{12566(x_2x_1^3 - x_1^4)} + \frac{1}{5108x_1^2} - 1 \leq 0,$$

$$g_3(x) = 1 - \frac{140.45x_1}{x_2^2x_3} \leq 0,$$

$$g_4(x) = \frac{x_2 + x_1}{1.5} - 1 \leq 0,$$

Solution found:

$$x^* = (0.05044713178541634, 0.32746441361099429, 13.23998350856038107)$$

$$F^* = 0.012700521857.$$

A.6 Pressure vessel design (six inequalities)

The pressure vessel design, was previously analysed by Sandgren [32] who first proposed this problem to minimize the total cost of the material, forming and welding of a cylindrical vessel. There are four design variables: x_1 (Ts, shell thickness), x_2 (Th, spherical head thickness), x_3 (R, radius of cylindrical shell) and x_4 (L, shell length). Ts (=x_1) and Th (=x_2) are integer multipliers of 0.0625 in. In accordance with the available thickness of rolled steel plates, and R (=x_3) and L (=x_4) have continuous values of $40 \leq R \leq 80$in. and $20 \leq L \leq 60$in., respectively. The mathematical formulation of the optimization problem is as follows:

$$Minimize\ \ f(x) = 0.6224x_1x_2x_3 + 1.7781x_2x_3^3 + 3.1611x_1^2x_4 + 19.84x_1^2x_3$$

s.t.

$$g_1(x) = 0.0193x_3 - x_1 \leq 0,$$

$$g_2(x) = 0.00954x_3 - x_2 \leq 0,$$

$$g_3(x) = 750.0 \times 1728.0 - \pi x_3^2 x_4 - \frac{4}{3}\pi x_3^2 \leq 0,$$

$$g_4(x) = x_4 - 240.0 \leq 0,$$

$$g_5(x) = 1.1 - x_1 \leq 0,$$

$$g_6(x) = 0.6 - x_2 \leq 0,$$

Solution found:

$$x^* = (1.125, 0.625, 58.2900704783923345, 43.6931234586562753)$$

$$F^* = 7197.73412633523851.$$

A.7 Pressure vessel design (four inequalities)

Another variation of this problem, that has two inequalities minus (g_5 and g_6 are eliminated) and the following bounds $1\times 0.0625 \leq x1, x2 \leq 99\times 0.0625, 10.0 \leq x3, x4 \leq 200.0$ has been solved by others researchers.
Solution found by our approach:

$$x^* = (0.875, 0.4375, 45.3366721064070408, 140.255022911949085).$$

$$F^* = 6090.53937693476024.$$

A.8 Speed reducer design (continuous-integer variables)

The design of the speed reducer [20], is considered with the face width x_1, module of teeth x_2, number of teeth on pinion x_3, length of the first shaft between bearings x_4, length of the second shaft between bearings x_5, diameter of the first shaft x_6, and diameter of the first shaft x_7 (all variables continuous except x_3 that is integer). The weight of the speed reducer is to be minimized subject to constraints on bending stress of the gear teeth, surface stress, transverse deflections of the shafts and stresses in the shaft. The problem is formulated as follows:

$$Minimize\ \ f(x) = 0.7854x_1x_1^2(3.3333x_3^2 + 14.9334x_3 - 43.0934)$$

$$-1.508x_1(x_6^2 + x_7^2) + 7.4777(x_6^3 + x_7^3) + 0.7854(x_4x_6^2 + x_5x_7^2)$$

s.t.

$$g_1(x) = \frac{27}{x_1x_2^2x_3} - 1 \leq 0,$$

$$g_2(x) = \frac{397.5}{x_1x_2^2x_3^2} - 1 \leq 0,$$

$$g_3(x) = \frac{1.93x_4^3}{x_2x_3x_6^4} - 1 \leq 0,$$

$$g_4(x) = \frac{1.93x_5^3}{x_2x_3x_7^4} - 1 \leq 0,$$

$$g_5(x) = \frac{1.0}{110.0x_6^3}\sqrt{\left(\frac{745.0x_4}{x_2x_3}\right)^2 + 16.9\times 10^6} - 1 \leq 0,$$

$$g_6(x) = \frac{1.0}{85.0x_7^3}\sqrt{\left(\frac{745.0x_5}{x_2x_3}\right)^2 + 157.5\times 10^6} - 1 \leq 0,$$

$$g_7(x) = \frac{x_2x_3}{40} - 1 \leq 0,$$

$$g_8(x) = \frac{5x_2}{x_1} - 1 \leq 0,$$

$$g_9(x) = \frac{x_1}{12x_2} - 1 \leq 0,$$

$$g_{10}(x) = \frac{1.5x_6 + 1.9}{x_4} - 1 \leq 0,$$

$$g_{11}(x) = \frac{1.1x_7 + 1.9}{x_5} - 1 \leq 0,$$

wuth $2.6 \leq x_1 \leq 3.6$, $0.7 \leq x_2 \leq 0.8$, $17 \leq x_3 \leq 28$, $7.3 \leq x_4 \leq 8.3$, $7.8 \leq x_5 \leq 8.3$, $2.9 \leq x_6 \leq 3.9$, $5.0 \leq x_7 \leq 5.5$.

Solution found by our approach:

$$x^* = (3.50002615416866586, 0.70000523059661887, 17, 7.30022922985589972,$$

$$7.8000228842193966, 3.35021507672250302, 5.28669973187709912).$$

$$F^* = 2996.3951944729081.$$

A.9 Speed reducer design (integer-discrete variables)

Another variation of this problem with the variables defined as follows: x_1, x_2, x_4, and x_5 must be integral multiples of 0.1. x_6 and x_7 must be integral multiples of 0.01, and x_3 must be an integer.

Solution found by our approach:

$$x^* = (3.3, 0.7, 17.0, 7.3, 7.8, 3.36, 5.29).$$

$$F^* = 2922.43527186608.$$