Journal of Information & Optimization Sciences
ISSN 0252-2667 (Print), ISSN 2169-0103 (Online)
Vol. 47 (2026), No. 1, pp. 185–205
DOI : 10.47974/JIOS-1512

Two-phase gradient based search method for global optimum of constrained mixed integer programming problems using a tunneling function & comparison

V. K. Srivastava *
Vinit K. Srivastava †
Ottawa Hospital Research Institute (OHRI)
501 Smyth Rd, Box 511
Ottawa, ON, K1H 8L6
Canada

Abstract

In this paper, a gradient-based optimization approach with two stages is proposed that is designed to efficiently solve mixed-integer programming problems with linear and nonlinear objective functions and constraints. To find a viable local minimum in the first step, a gradient method with hemstitching modifications is used, which guarantees that the trajectory is redirected into the feasible region whenever constraint(s) violation take place. The purpose of the second phase is to identify another point on the opposite side of the barrier where the function value is approximately the same, by constructing a tunneling function at the current local minimum. This newly discovered point serves as the starting vector for the next cycle, which repeats the process of searching for the global optimum. The iterative procedure is repeated until no further reduction in the objective function value is observed. Several benchmark problems involving mixed-integer variables are tested. The gradient method involving 3^n enumeration that the author previously proposed is compared with the results from this new optimization technique, with a minor change made in the first phase of the previous published method. The results are compared with other published results.

Subject Classification: *90C11, 90C30, 90C59.*

Keywords: *Mixed integer problems, Nonlinear optimization methods, Heuristic methods.*

† *ORCID-ID:* **0009-0007-9417-1802**

* *E-mail:* **vijayasrivastava@gmail.com (Corresponding Author)**
† *E-mail:* **vsrivastava@ohri.ca**

1. Introduction

Mixed-integer nonlinear problems (MINLP) are widely encountered in various scientific and engineering disciplines. These include applications to solve complex MIP problems a) in medical science where neural network in conjunction with optimization method is used to classify tumors and identify cancer stages and in hospitals to schedule surgeries that are performed in multiple operating rooms, b) in engineering to solve design problems such as redundancy allocation problems, design of chemical reactors and scheduling of production lines and c) in operations research and transportation problems. These problems are mostly non-convex and thus difficult to solve for the global optimal solution. Over the years, metaheuristic approaches have proven to be valuable tools for addressing a wide range of optimization challenges and are gaining recognition as practical alternatives in scenarios constrained by time and computational resources. In this paper, some suggested important classes of nonconvex mixed integer linear and nonlinear programming test problem results are presented.

An important type of nonlinear optimization problem is known as MINLP, where some of the decision variables are integers. The branch-and-bound (B&B) technique developed by Land and Doig[1] to solve problems of integer linear programming was extended to solve MINLP problems. The bender decomposition technique was developed to solve mixed-integer programming problems by simultaneously addressing continuous and discrete variables [2–4]. In the last few decades, several heuristic algorithms have been reported to solve nonlinear optimization problems where both the objective function and constraints exhibit nonlinearity. Another hybrid conjugate gradient projection method for solving large-scale nonlinear monotone equations has been proposed [5].

Genetic algorithms (GA) do not rely on assumptions of traditional optimization techniques, such as continuity, derivative existence, or unimodality [6]. The population of candidate solutions is initially generated—either heuristically or randomly—and selection, crossover, and mutation operations are then applied iteratively and sequentially to evolve the population into a new generation of improved solutions [6–14]. In Evolutionary Programming (EP) the solution to optimization problem is iteratively improved to arrive at the appropriate solution. The algorithm is used to solve highly nonlinear and nonconvex problems. Using probabilistic transition rules of the Gaussian relationship between parents and offspring, the EP algorithm’s global search keeps track of a population of potential solutions at each iteration. This allows the algorithm to navigate the search space's hills and valleys increasing the likelihood of identifying a solution with the best possible objective function value. The MINLP problem has been solved effectively using a variety of EP-based algorithms [15–18].

Eberhart and Kennedy [19–32] pioneered Particle Swarm Optimization (PSO), a distinct population-based technique inspired by social behavior patterns, and showed its effectiveness in solving optimization problems. In order to accomplish this, the particles search the solution domain for extreme values of the function being optimized, either the highest or the lowest. They also look for the maximum or minimum of a given function. Due to its quick convergence, this metaheuristic search algorithm is widely used to solve a variety of optimization problems. Nevertheless, the original PSO has a tendency to converge prematurely to suboptimal solutions, especially when dealing with high-complexity optimization tasks [8]. Exploration and exploitation are the two main phases of the meta-heuristic search process that must be balanced [19]. In the exploration phase, various solutions within the search space are found, and in the exploitation phase, the search process finds the optimal solution in the neighborhood of the best solutions, thereby utilizing the information found. Numerous methods such as parameter adaptation [9, 10], modified neighborhood structure [8,11,12], and modified learning strategies [11-32] were introduced to enhance the exploration and exploitation searches of PSO for solving different types of optimization problems that effectively still remains as an on-going research interest since it was introduced in the mid-1990s.

Various broadly applicable solvers have emerged that employ convex relaxation strategies to handle nonconvex constraints in MINLPs. Tools such as αBB, BARON, and GLOP exemplify this class of methods [33–35]. In addition, global solutions to MINLPs are now attainable via branch-and-bound techniques, which have been incorporated into advanced frameworks like SCIP (Solving Constraint Integer Programs) [36–39], available under an open-source license. GAMS comes with the latest version of SCIP and commercial solvers, such as BARON [40], Lindo API [41], and Octeract [42,43], which use solver CPLEX 22.1 for LP/MIP/QP/QCP problems. It also offers facilities for solver comparisons.

A stepwise minimization framework composed of two sequential stages was utilized in earlier work addressing constrained integer programming, as noted in [44–48]. In the initial stage, a variant of the steepest descent technique is applied, guiding the search along the function's gradient within allowable bounds to locate a nearby minimum. If a constraint is violated, a correction, called hemstitching, is started to relocate the solution to a feasible coordinate $\mathbf{X}_k$ [49]. This approach entails making small, perpendicular inward movements away from the violated constraint. All of these changes are limited to grid points that are determined by discretizing each element of the n-dimensional decision vector using integers. Following correction, the search proceeds along a new trajectory derived by combining the objective's gradient with those of the constraints active at $\mathbf{X}_k$ [50]. The cycle restarts from the latest feasible candidate, and the process repeats until no further meaningful reduction in the objective function is achievable within that phase.

Typically, the local minimum would be found in the same valley as the initial vector during the first phase. When working with continuous or integer domains, numerical search approaches generally have this issue. The open literature has reported on a number of well-documented methods to improve the chances of capturing a global minimum. But due to the discretization process, there are now more local minimums, which exacerbates the issue. The second phase is used to get around the issue of becoming trapped in local minima. In this stage, a directional enumeration strategy involving 3^n steps [44] is applied using the basis vectors. This approach has shown strong performance in progressing from the earlier-identified local solution toward a globally optimal point, even when that solution lies close to the boundary between feasible and slightly infeasible zones (within a tolerance of $\in<10^{-3}$). The best optimum vector is found by applying the 3^n enumeration to find the minimum function; $\mathbf{X}^*$ is selected as the desired outcome.

Although the approach detailed in [44–47], with minor enhancements introduced in this work, has demonstrated high efficiency in handling nonlinear integer optimization challenges, its runtime tends to scale significantly with an increase in the problem dimension, n. In real-world scenarios, processing time can become impractical as n grows larger. A breakdown of the computation time across both phases revealed that the effort needed for the 3^n enumeration expands at an exponential rate relative to n.

In this paper, a new approach to address the shortcoming of handling problems with large n values is presented. It involves substituting the second phase, the 3^n enumeration phase, with a tunneling method that was initially documented in [51]. The method has been tailored to fit the requirements of this study and involves constructing a tunneling function, which is approximately solved by moving in the direction of its gradient.

A reference point is placed at the previously found local minimum. During the search for the tunneling function's root, if any constraint is breached, a corrective hemstitching technique is applied to guide the solution back into the permissible region. The starting point ($\mathbf{x}^o$) for the first iterative cycle is determined by taking the solution to the tunneling function minimization, $\mathbf{x} \neq \mathbf{x}^*$. From there, the search for the global optimum is carried out. When no more improvement is made or the search returns to a minimum that was previously encountered, the iterative process comes to an end.

Problem Description

Consider a general minimization scenario, which can be formulated in the following manner:

Find $\boldsymbol{X}^*$ that minimizes	$f(\boldsymbol{X})$
Where	$f(\boldsymbol{X}) \in \mathbb{Z}$
	$\boldsymbol{X_L} \leq \boldsymbol{X} \leq \boldsymbol{X_H}$
Subject to	$g_j(\boldsymbol{X}) \leq b_j$ for $j = 1,2,3,\ldots\ldots m$

Let $\boldsymbol{X}$ represent the design vector of dimension n, with $\boldsymbol{X_L}$ and $\boldsymbol{X_H}$ indicating its lower and upper limits, respectively. The related constraint functions $g_j(X)$ are not necessarily satisfied only because the bounds on $\boldsymbol{X}$ are satisfied. Moreover, in the space bounded by these bounds and the constraint functions, the function f($\boldsymbol{X}$) can reach a local minimum, which is also the global minimum, f($\boldsymbol{X}$*).

Algorithm Structure Overview

The proposed optimization strategy is executed in iterative cycles, each involving a minimization phase followed by a tunneling phase. During the minimization step, the goal is to identify a local minimum within the valley where the search begins. To keep the search trajectory within the allowable domain, the algorithm moves in the direction of the function's gradient. Step size is modified dynamically depending on how the function value changes, and when greater precision is needed, it uses a quadratic model to approximate an appropriate step size [44]. If a constraint is breached, the procedure adjusts the search path using a vector $\boldsymbol{Q}$, which is computed as the normalized sum of gradients for the p constraints violated: $g_1(X)$,... $g_p(X)$ [48]. The i^{th} component of vector ***Q*** is given by the following formula:

$$q_i = \sum_{k=1}^{p} \frac{K_k \left(\frac{\partial g_k}{\partial x_i} \right)}{\sqrt{\sum_{j=1}^{n} \left(\frac{\partial g_k}{\partial x_j} \right)^2}}$$

where the following coefficients, K_1,..., K_p, are assumed to be proportionate to the degree of constraint violation:

$$K_k = \frac{(g_k - b_k)}{\sum_{j=1}^{p} (g_j - b_j)}$$

The direction vector ***Q*** below indicates the path along which the algorithm advances into the feasible domain using incremental steps of size ℓ (where ℓ (≤1)), starting with an initial step length λ_o:

$$X_k = X_o + \sum_{j=0}^{k} [(\lambda_o + j * \ell) * Q_j]$$

During each step of the algorithm, the direction vector ***Q*** is recalculated. The initial point X_o, which begins outside the permissible solution space, is expected to enter the feasible region after a sequence of k^{th} moves. Once feasibility is restored, the algorithm proceeds along a direction ***P***, defined by the combination of the normalized gradient of the objective function and the gradients of any constraints previously breached—each evaluated anew at the feasible location [3]:

$$P = \frac{p}{\| p \|}$$

where, p is given by

$$p = \frac{\nabla f}{\| \nabla f \|} + \sum_{j=1}^{m} \frac{\nabla g_j}{\| \nabla g_j \|}$$

To advance the search in the direction ***P***, the algorithm utilizes a refined step size estimation technique as detailed in [44]. If the search trajectory exits the feasible space again, a corrective mechanism—previously described as the hemstitching method—is employed to steer the solution back into the allowable region.

When the objective function shows no additional decrease, or a previously encountered minimum resurfaces, the search process is terminated for that phase. The minimum obtained at this point is then employed to initiate the tunneling procedure, which aims to identify a new starting position for examining a different segment of the solution landscape. The mathematical expression for the tunneling function applied in this context is provided below [50]:

$$T(X) = \frac{f(X) - f(X^*)}{[(X - X^*)^T (X - X^*)]^\eta}$$

where $\eta \geq 1$. A point located on the same level in the neighboring valley is taken as the origin, or zero, of the tunneling function, denoted as X^0. This point, X^0, can be determined by applying a conventional root-finding method or by minimizing *T(X)* along its gradient until $|T(\mathbf{X}^*) - T(\mathbf{X}^0)| \leq \varepsilon$, in which ε is a small parameter. The value of x_2^0, representing the root of the subsequent tunneling function, is established, starting from the local minimum x_1^* [44]:

$$T(x) = \frac{f(x) - f(x_1^*)}{[(x - x_1^*)^T (x - x_1^*)]^\eta}, \ \eta \geq 1$$

In contrast to function minimization phase I, the focus here is to determine a point x_2^0 where the function value $|f(x_2^0) - f(x_1^*)|| \leq \delta$ where δ is specified positive number which ideally should be selected to be close to zero. $T(x)$ is treated as an unconstrained function during the process of moving along the gradient of the tunneling function, where the variables are not discretized, and a corrective hemstitching procedure is applied if the search moves into the infeasible region. The descent is terminated if $T(x_i) \approx T(x_{i-1})$ and the point x_2^0 found and search for the global minimum restarts using x_2^0 as new starting vector for the minimization phase I for the next cycle where the local minima x_2^* will have function value $f(x_2^0) \approx f(x_1^*)$ and consequently $f(x_2^*) \leq f(x_1^*)$. The tunneling function $T(X)$ is constructed at x_2^* and start the search for its zero x_3^0 followed by search for the next local minimum x_3^* using x_3^* as the next starting vector where, $f(x_3^*) \leq f(x_2^*)$. At the start of tunneling phase for each cycle, the exponent η is chosen to be 1 and $x = x^* + \varepsilon$ where ε is a random vector of very small magnitude, typically $\|\varepsilon\| = 10^{-2}$. During the iterative process of searching for the constrained global minimum, the collection of all local minima encountered in each cycle of the search algorithm forms a non-increasing sequence $f_1 \geq f_2 \geq \ldots \geq f_k$, where f_k represents the global minimum f *. The search is stopped when a predetermined number of iterations fail to produce an improvement in function value.

Numerical Examples

The algorithm was implemented in Fortran using double precision and tested on a system equipped with an AMD Athlon processor (1.8GHz) and 512 MB of RAM. It was applied to evaluate several benchmark optimization problems drawn from previously published studies. The test problems for $N \leq 12$ and one test problem for $N=15$ that were resolved and compared by both approaches using the proposed tunneling algorithm are shown, and a synopsis of the computational outcomes is provided below:

Test Problem #1 Pressure Vessel [52]

This optimization task focuses on the design of a compressed air storage tank with a required minimum volume of 750 ft³ and a working pressure of 3,000 psi. The objective is to lower the overall cost including welding material costs. The shell thickness (x_1), head thickness (x_2), inner radius (x_3), and cylindrical section length (x_4) are the main design factors. With the restriction that they be multiples of 0.0625 inches, variables x_1 and x_2 acquire discrete values. The issue is stated as a challenge to minimize costs

$$f(X) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3, x \in \Re^4$$

subject to:

$$g_1(X) = -x_1 + 0.0193x_3 \leq 0$$

$$g_2(X) = -x_2 + 0.00954x_3 \leq 0$$

$$g_3(X) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1,296,000 \leq 0$$

$$g_4(X) = x_4 - 240 \leq 0$$

Tunneling algorithm is applied to solve the above problem and the results are compared with two phase 3^n enumerative algorithm [45] and published results [52]. The initial vector (20, 10, 6, 172) is used to solve this problem. The starting step size is λ_o =0.000153, and a small increment of ℓ

= 0.00001 is added to the initial size λ_o to ensure that the feasible region is returned to when the constraint is violated during Phase I. This provides a minimum of (0.8125, 0.4375, 42.0958, 176.676) and function 6060.186. During the second phase of 3^n enumeration about this minimum provides an optimum solution of (0.8125, 0.4375, 42.0883, 176.669) with minimum function of 6058.760. Now restarting the search again using this optimum as initial starting vector, a lower optimum (0.25, 0.125, 10, 176.471) with function value 399.696 found during first Phase and a global optimum (0.25, 0.125, 10, 64.1116) with minimum value 147.0692 found after 3^n enumeration about this minimum is located that remained unchanged with different starting vectors. However, in the proposed in the tunneling algorithm, for chosen value of $\|\varepsilon\| = 10^{-2}$ and h=1, the suboptimum reached during Phase I (0.25, 0.125, 11.49280, 55.7381) with a minimum function value of 154.31184. The optimum reached after several iterations through tunneling Phase II provided no improvement in the feasible region. The results from the proposed Tunneling algorithm required a CPU time of 0.187 sec which is less than the 81.515 seconds required during the 3^n enumeration search. The reported results by both 3^n enumeration and the Tunneling algorithm are superior to the result reported in the reference paper [52] and has not yet been reported elsewhere.

Test Problem #2 Speed Reducer design optimization problem

This problem pertains to the minimization of weight reducers [52] while adhering to constraints regarding the bending stress of gear teeth, transverse deflections of the shaft, surface stress, and stresses of the shaft. This is seven variables mixed integer variables where all variables are continuous except variable x_3 that is discrete.

Minimize:

$$f(\mathbf{x}) = 0\ .7854 x_1 x_2^{\ 2}(3.3333 x_3^{\ 2} + \ 14.9334 x_3 - 43.0934) - 1.508 x_1$$
$$(x_6^{\ 2} + x_7^{\ 2}) + 7.4777(x_6^{\ 3} + x_7^{\ 3}) + 0.7854(x_4 x_6^{\ 2} + x_5 x_7^{\ 2})$$

Subject to:

$$g_1(\bar{X}) = \frac{27}{x_1 x_2^2 x_3} - 1 \le 0, \qquad g_2(\bar{X}) = \frac{397.5}{x_1 x_2^2 x_3^2} - 1 \le 0,$$

$$g_3(\bar{X}) = \frac{1.93x_4^{\ 3}}{x_2x_3x_6^2} - 1 \le 0, \qquad g_4(\bar{X}) = \frac{1.93x_5^3}{x_2x_3x_7^4} - 1 \le 0,$$

$$g_5(\bar{X}) = \frac{1.0}{110x_6^3}\sqrt{\left(\frac{745x_4}{x_2x_3}\right)^2 + 16.9\times 10^6} - 1 \le 0,$$

$$g_6(\bar{X}) = \frac{1.0}{85x_7^3}\sqrt{\left(\frac{745x_5}{x_2x_3}\right)^2 + 157.5\times 10^6} - 1 \le 0,$$

$$g_7(\bar{X}) = \frac{x_2x_3}{40} - 1 \le 0, \qquad g_8(\bar{X}) = \frac{5x_2}{x_1} - 1 \le 0,$$

$$g_9(\bar{X}) = \frac{x_1}{12x_2} - 1 \le 0, \qquad g_{10}(\bar{X}) = \frac{1.5x_6 + 1.9}{x_4} - 1 \le 0$$

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

Both 3^n enumeration and the proposed tunneling algorithm provide far better results with less computational cost than those reported in [52]. The best optimum from second phase of 3^n enumeration [44] algorithm is (3.50964, 0. 673466, 17, 7.29964, 7.79893, 2.88957, 4.96897) with minimum cost 2589.94 with computation time 1.094 sec. The optimum from the tunneling algorithm is (3.59659, 0.700, 17, 7.30, 8.00, 3.00, 5.00) with minimum cost 2584.0977 with computation time 0.750 sec.

Test Problem #3 Optimization problem for tension/compression spring design

The goal of this problem [52] is to reduce the weight of a spring that operates under both tension and compression, while satisfying constraints related to surge frequency, deflection limits, shear stress, and outer diameter, along with specified bounds on the design variables. The formulation includes two continuous variables— x_1 for wire diameter and x_2 for mean coil diameter—as well as one discrete variable, x_3, representing the number of active coils. The formal problem statement is presented below:

Minimize: $f(\bar{X}) = (x_3 + 2)x_2x_1^2$

Subject to:

$$g_1(\bar{X}) = 1 - \frac{x_2^3 x_3}{7178 x_1^4} \le 0, \qquad g_2(\bar{X}) = \frac{4x_2^2 - x_1 x_2}{12566 x_2 x_1^3 - x_1^4} + \frac{1}{5108 x_1^2} \le 1,$$

$$g_3(\bar{X}) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \le 0, \qquad g_4(\bar{X}) = \frac{x_1 + x_2}{1.5} \le 1$$

With $0.05 \le x_1 \le 2.0,\ 0.25 \le x_2 \le 1.3\ \&\ 2.0 \le x_3 \le 15.0$

The first phase of 3^n enumeration optimization method result is (0.0478112, 0.233081,14) with function value 0.0085248. After enumeration about this optimum a final optimum (0.0478112, 0.233081, 3) with lower function value 0.002664 was reached within 0.831 sec. The proposed tunneling optimization provides optimum (0.0500, 0.250, 8) with function value 0.00625 within 0.093. These results from both optimization methods reported here are better than reported [52] best solution (0.05169, 0.356750, 11.287126) with function value 0.012665.

Other Mixed Integer Problems Solved

In this section a general class of test problems are considered that utilize linear or non-linear objective function and constraints.

a) Mixed-Integer Nonlinear Convex Problem [53]

The mixed integer nonlinear convex programming problem (MINLP) proposed by Grossman in report [53] is defined as follows:

Minimize: $Z = y_1 + 1.5y_2 + 0.5y_3 + x_1^2 + x_2^2$

Subject to: $(x_1 - 2)^2 - x_2 \le 0$

$$x_1 - 2y_1 \ge 0$$
$$x_1 - x_2 - 4(1 - y_2) \le 0$$
$$x_1 - (1 - y_1) \ge 0$$
$$x_2 - y_2 \ge 0$$
$$x_1 + x_2 \ge 3y_3$$
$$y_1 + y_2 + y_3 \ge 1$$

The optimum solution shown in [53] is $y_1,\ y_3 = 0,\ y_2 = 1,\ x_1 = 1,$ $x_2 = 1,\ Z = 3.5$ which is based on extended cutting plane method for solving convex MINLP problems outlined in [53]. The sub-optimal solution obtained at the end of Phase I of the enumerative algorithm developed [45] is (2.50036, 2.26342, 0 , 0, 1) with function value 11.8748377 and now 3^n enumeration carried out about this sub-optimal vector gave an optimal solution of (1.500 ,1.500 ,0 ,0 ,1) with minimum function of 5.00. This solution is inferior to the reported result shown above. To enhance the outcome of the enumerative method it was necessary to modify baseline starting vector of 3^n enumeration algorithm [45] to take into account the minimum found in Phase I in the infeasible region where the 4^{th} constraint was violated by order of 10^{-5} and the new found Phase I sub-optimum is (0.999937, 1.07154, 0, 1, 0) with function value 3.648077. Now 3^n enumeration carried out using this solution of Phase I provided the desired final solution (1.0, 1.0, 0, 1, 0) with function value 3.500.

2) Modified mixed-integer Nonlinear Problem [54]

A modest extension to MINLP problem has been proposed in [54] that deals with optimal positioning of a new product in multi-attribute space as follows:

Minimize:

$$z = 5y_1 + 6y_2 + 8y_3 + 10x_1 - 7x_6 - 18\ln(x_2 + 1) - 19.2\ln(x_1 - x_2 + 1) + 10$$

Subject to: $0.8\ln(x_2 + 1) + 0.96\ln(x_1 - x_2 + 1) - 0.8x_6 \geq 0$

$$x_2 - x_1 \leq 0$$

$$x_2 - Uy_1 \leq 0$$

$$x_1 - x_2 - Uy_2 \leq 0$$

$$\ln(x_2 + 1) + 1.2\ln(x_1 - x_2 + 1) - x_6 - Uy_3 \geq -2$$

and provides $y_1 + y_2 \leq 1$

$$y = \in (0,1)^3, a \leq x \leq b, x = (x, : j = 1,2,6) \in R^3$$

$$a^T = (0,0,0), b^T = (2,2,1), U = 2$$

The 3^N enumerative procedure was initiated with starting vector (2, 2, 1, 0, 1, 0) and lower and upper limits on the variables as specified. A first phase feasible sub-optimum solution (1.99886, 0, 1, 0, 1, 0) with function value 7.902527 was reached after several iterations about which 3^N enumeration is carried out iteratively until there is no improvement in function minimum value. An optimum solution (1.30098, 0, 1, 0, 1, 0) with function value 6.009759 was reached in 14.89 seconds and is in agreement with reported result in [54].

The Tunneling algorithm converges to (1.30160, 0, 1, 0, 1, 0) where function value is 6.01079.

The above problem has been modified in [55] by introducing a linear form of the objective function; additionally, the variables **y** were combined to **x**:

$$\min f(x) = 10x_1 - 17x_2 - 5x_4 + 6x_5 + 8x_6$$

Subject to:

$$g_1(x) = 0.8\ln(x_2+1) + 0.96\ln(x_1 - x_2 + 1) - 0.8x_3 \geq 0$$
$$g_2(x) = \ln(x_2+1) + 1.2\ln(x_1 - x_2 + 1) - x_3 - 2x_6 + 2 \geq 0$$
$$x_1 - x_2 \geq 0, 2x_4 - x_2 \geq 0 \quad 2x_5 - x_1 + x_2 \geq 0, 1 - x_4 - x_5 \geq 0$$
$$x_1 - x_2 \geq 0, 2x_4 - x_2 \geq 0$$

The original problem is partitioned into objective function, linear constraints, variables and bounds and are used to solve as Mixed Integer Optimization problem. In this case x_1, x_2, x_3 and x_6 are bounded and x_4, x_5 and x_6 are binary. The intractable constraints $g_1(x) \geq 0$, $g_2(x) \geq 0$ are substituted by tractable disjunctive approximations, as described in [55], and the problem is reformulated into a MILO model that is compatible with commercial solvers. The solution, obtained through CPLEX, offers a result that is close to feasibility and near-optimality, with an objective function value of -7.693. This MILO solution is successfully repaired by the PGD method to provide feasible and locally optimal solution (0.699, 0.699, 0.530, 1, 0, 0) with objective value -7.021. This problem was solved using 3^N enumerative procedure that was initiated with infeasible starting vector (2.5, 2.5, 0, 1, 0, 0) and lower and upper limits on the variables as specified. A first phase feasible sub-optimum solution (1.99886, 0, 1, 0, 1, 0) with function value 7.902527 was reached after several iterations about which 3^N enumeration is carried iteratively until no improvement in function minimum value of -6.3855215 can be achieved and the optimum vector is (1.20795, 1.20795, 0.79206, 1, 0, 0). The tunneling algorithm yields a sub-optimal solution (1.10768, 0, 0.89452, 0, 1, 0) with function value 1.8699.

3) Electrical Wire Plant Upgrading MINLP Test Problem

The operating cost (in thousands) is calculated based on the number of machines of each option, denoted as y_i, and the proportion of annual machine time, represented by x_{ij}, allocated to producing wire size j, on machines of option i (where $i = 1, 5$; $j = 1, 2$):

This is a large variable mixed integer variable problem consisting of 5 integer variables and 10 continuous variables, described in [56] that deals with the manufacturing of two types of wire: covered wire and bare wire of various sizes with some produced by modifying existing machines while others require the purchase of new machines. The problem is minimizing the plant's running cost so that plant's projected demand as well as schedule to produce each type of wire are met.

Minimize:

$$P = 30y_1 + 33.6x_{11} + 32.88x_{12} + 50y_2 + 47.4x_{21} + 47.04x_{22} + 80y_3 + 58.8x_{31} + 57x_{32} + 100y_4 + 62.4x_{41} + 54.7x_{42} + 140y_5 + 86.4\, x_{51} + 82.8x_{52}$$

Two different types of constraints need to be met. The first type is based on the expected sales of each type and size of wire, as indicated by the following four constraints:

$$5880x_{11} + 8820x_{21} - 7200x_{31} - 9600x_{41} \geq 3000 \text{ (BARE1)}$$

$$4704x_{12} + 8232x_{22} - 6000x_{32} - 7800x_{42} \geq 2000 \text{ (BARE2)}$$

Regarding size 1 and 2 covered wires:

$$6984x_{31} + 9312x_{41} + 9312x_{51} \geq 14000 \text{ (COVERED1)}$$

$$5820x_{32} + 7566x_{42} + 6984x_{52} \geq 10000 \text{ (COVERED2)}$$

Furthermore, the sizes of all wires is dependent on the quantiy of machine type i.

$$y_i \geq x_{i1} + x_{i2}, i = 1, \ldots .5$$

The constraints of the settings of the machine

$$y_1 \leq 2$$
$$y_3 + y_4 \leq 1$$

The above MILP problem was solved by the proposed two phase tunneling algorithm provides suboptimal solution (0.4421, 0.3996429, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.5399156, 1.4595826, 1, 0, 0, 0, 3) with function 731.86430422 from gradient search phase and finally an optimal solution (0.470835, 0.305874, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.4398, 1.45947, 1, 0.0, 0.0, 0.0, 3) with function value 721.0955648 provided by tunneling algorithm. This solution is slightly better than the solution (0.51020, 0.42717, 0.0, 0.0, 0. 0, 0.0, 0.0, 0.0, 1.50344, 1.43184, 1, 0, 0, 0, 3) with a function value 729.57606 shown in [56].

Conclusion

This paper proposes to enhance the 3^n enumerative method for mixed integer nonlinear problems. It entails performing a 3^n enumeration about two sub-optimal vectors found in phase I: one in the feasible region and the other in the infeasible region near the constraint boundary (i.e., $\varepsilon < 10^{-3}$). The best optimum that satisfies all constraints is chosen as global optimum solution. A Tunneling algorithm is implemented to solve large variable MINL problems. The computation performance of both algorithms are illustrated on set of five reported problems and the last large fifteen-variable problem is solved using Tunneling algorithm provided a solution better than the one reported. Both of the two proposed algorithms provide better optimum solutions for the first three test problems than the one reported.

Acknowledgment

The work was done independently by the authors with no funding from any organization.

References

[1] H. Land and A. G. Doig, "An automatic method for solving discrete programming problems," *Econometrica*, vol. 28, no. 3, pp. 497-520 (1960). [Online]. Available: https://doi.org/10.2307/1910129

[2] A. M. Geoffrion, "Generalized Benders decomposition," *J. Optim.*, vol. 10, pp. 237-260 (1972).

[3] C. A. Floudas, A. Aggarwal, and A. R. Ciric, "Global optimum search for nonconvex NLP and MINLP problems," *Computers and Chemical Engineering*, vol. 13, no. 10, pp. 1117-1132 (1989). [Online]. Available: https://doi.org/10.1007/BF00934810

[4] A. Aggarwal and C. Floudas, "A decomposition approach for global optimum search in QP, NLP, and MINLP problems," *Computer Science* (1990). [Online]. Available: https://doi.org/10.1007/BF00934810

[5] K. Mompati and P. Kaelo, "An efficient hybrid conjugate gradient-based projection method for convex constrained nonlinear monotone equations," *Journal of Interdisciplinary Mathematics*, vol. 22, no. 6, pp. 1031-1050 (2019). [Online]. Available: DOI: 10.1080/09720502.2019.1700889

[6] J. E. Baker, "Adaptive selection methods for genetic algorithm," in *Proceedings of the First International Conference on Genetic Algorithms* (1985). [Online]. Available: https://doi.org/dl.acm.org/doi/10.5555/645511.657075

[7] D. E. Goldberg and K. Deb, "Foundations of genetic algorithms: A comparative analysis of selection schemes used in genetic algorithms," in *1st ed., pp. 69-93*, Elsevier (1991). [Online]. Available: https://doi.org/10.1016/B9780080506845500082

[8] L. D. Whitley, "A genetic algorithm tutorial," *Statistics and Computing*, vol. 4, no. 2, pp. 65–85, Jun. (1994). [Online]. Available: https://doi.org/10.1007/BF00175354

[9] B. S. Cheung, A. Langevin, and H. Delmaire, "Coupling genetic algorithm with a grid search method to solve mixed integer nonlinear programming problems," *Computers and Mathematics with Applications*, vol. 34, no. 12, pp. 13-23 (1997). [Online]. Available: www.sciencedirect.com/science/article/pii/S0898122197002290

[10] K. Deep, K. P. Singh, M. Kansal, and C. Mohan, "A real coded genetic algorithm for solving integer and mixed integer optimization problems," *Applied Mathematics and Computation*, vol. 212, no. 2, pp. 505-518 (2009).

[11] K. Deb, "An efficient constraint handling method for genetic algorithms," *Computer Methods in Applied Mechanics and Engineering*, vol. 186, no. 2, pp. 311–338 (2000).

[12] Z. Hua and F. Huang, "An effective genetic algorithm approach to large scale mixed integer programming problems," *Applied Mathematics and Computation*, vol. 174, no. 2, pp. 897-909 (2006).

[13] D. Goldberg, *Genetic Algorithms: Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, MA (1989).

[14] M. Thakur, S. S. Meghwani, and H. Jalota, "A modified real coded genetic algorithm for constrained optimization," *Applied Mathematics and Computation*, vol. 23, pp. 292-317 (2014).

[15] M. Crepinsek, M. Mernik, and S. H. Liu, "Analysis of exploration and exploitation in evolutionary algorithms by ancestry trees," *International Journal of Innovative Computing and Applications*, vol. 3, no. 1, pp. 11-19 (2011).

[16] L. Costa and P. Oliveira, "Evolutionary algorithms approach to the solution of mixed integer non-linear programming problems," *Computers and Chemical Engineering*, vol. 25, no. 2, pp. 257-266 (2011).

[17] Y. Lin, "Mixed-integer constrained optimization based on memetic algorithm," *Journal of Applied Research and Technology*, vol. 11, no. 2, pp. 242-250 (2013).

[18] T. Back, D. B. Fogel, and Z. Michalewicz, *Handbook of Evolutionary Computation*, Computational Intelligence Library, Oxford University Press, Inc., New York, NY, USA (1997). [Online]. Available: https://doi.org/. ISBN 0-7503-0392-1 and 0-7503-0895-8

[19] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Network (ICNN)*, vol. 4, pp. 1942-1948 (1995).

[20] J. Kennedy and R. C. Eberhart, "The particle swarm: Social adaptation in information-processing systems," in *New Ideas in Optimization*, pp. 379-387, McGraw-Hill, London, UK (1999).

[21] W. H. Lim, "A self-adaptive topologically connected-based particle swarm optimization," *IEEE Access*, vol. 6, pp. 65347-65366 (2018).

[22] L. C. Cagnina and S. C. Esquivel, "Solving engineering optimization problems with the simple constrained particle swarm optimizer," *Informatics*, vol. 32, no. 2, pp. 319-328 (2008).

[23] L. C. Cagnina, S. Esquivel, and R. Gallard, "Particle swarm optimization for sequencing problems: A case study," in *Proc. IEEE Congress on Evolutionary Computation (CEC)*, pp. 536-541 (2004). [Online]. Available: https://doi.org/10.1109/CEC2004.1330903.

[24] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *IEEE International Conference on Evolutionary Computation Proceedings: IEEE World Congress on Computational Intelligence (Cat. No.98TH8360)*, pp. 69-73 (1998). [Online]. Available: https://doi.org/10.1109/ICEC.1998.699146

[25] A. A. Karim, N. M. Isa, and W. H. Lim, "Hovering swarm particle swarm optimization," *IEEE Access*, vol. 9, pp. 115719-115749 (2021). [Online]. Available: https://doi.org/10.1109/ACCESS.2021.3106062

[26] M. A. De Oca, T. Stutzle, M. Birattari, and M. Dorigo, "Frankenstein's PSO: A composite particle swarm optimization algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 1120-1132 (2009). [Online]. Available: https://doi.org/10.1109/TEVC.2009.2021465

[27] Y. Fukuyama, "Fundamentals of particle swarm optimization," in *Modern Heuristic Optimization Techniques: Theory and Applications to Power Systems*, Hoboken, NJ, USA, Wiley-IEEE Press (2008). [Online]. Available: https://doi.org/10.1002/9780470225868.ch4.25

[28] Q. Chen and K. Yasuda, "Particle swarm optimization based global descent method," *Special issue on Electronics, Information and Systems / Special Issue Letter*, vol. 26, pp. 1558 (2009). [Online]. Available: https://doi.org/10.1587/elex.6.1558

[29] M. Mavrovouniotis, C. Li, and S. Yang, "A survey of swarm intelligence for dynamic optimization: Algorithms and applications," *Swarm and Evolutionary Computation*, vol. 33, pp. 1–17 (2017). [Online]. Available: https://doi.org/10.1016/j.swevo.2016.12.005

[30] F. Yang, P. Wang, Y. Zhang, L. Zheng, and J. Lu, "Survey of swarm intelligence optimization algorithms," in 2017 IEEE International Conference on Unmanned Systems (ICUS), pp. 544-549 (2017).

[31] F. Wang, H. Zhang, and A. Zhou, "A particle swarm optimization algorithm for mixed-variable optimization problems," *Swarm and Evolutionary Computation*, vol. 60, 100879 (2021).

[32] T. M. Shami, S. Mirjalili, Y. Al-Eryani, K. Daoudi, S. Izadi, and L. Abualiga, "Velocity pausing particle swarm optimization: A novel variant for global optimization," *Neural Computing and Applications*, Advance online publication (2023). [Online]. Available: doi.org/10.1007/s00521-022-08179-0.

[33] C. S. Adjiman and C. A. Floudas, "Rigorous convex under estimators for general twice-differentiable problems," *Journal of Global Optimization*, vol. 9, no. 1, pp. 23–40 (1996). [Online]. Available: doi:10.1007/BF00121749

[34] N. V. Sahinidis, "BARON: A general purpose global optimization software package," *Journal of Global Optimization*, vol. 8, no. 2, pp. 201-204 (1996).

[35] E. D. Smith and C. C. Pantelides, "A symbolic reformulation/spatial branch-and-bound algorithm for the global optimisation of noncon-

vex MINLPs," *Computers & Chemical Engineering*, vol. 23, no. 4–5, pp. 457–478 (1999). doi: 10.1016/s0098-1354(98)00286-5.

[36] P. Belotti, C. Kirches, S. Leyffer, J. Linderoth, J. Luedtke, and A. Mahajan, "Mixed-integer nonlinear optimization," *Acta Numerica*, vol. 22, pp. 1–131 (2013). doi: 10.1017/S0962492913000032.

[37] S. Vigerske and A. Gleixner, "SCIP: Global optimization of mixed-integer nonlinear programs in a branch-and-cut framework," *Optimization Methods and Software*, vol. 33, no. 3, pp. 563–593 (2017). doi: 10.1080/10556788.2017.1335312.

[38] A. Gleixner, L. Eier, T. Gally, G. Gamrath, P. Gemander, R. L. Gottwald, G. Hendel, C. Hojny, T. Koch, M. Miltenberger, B. Müller, M. E. Pfetsch, C. Puchert, D. Rehfeldt, F. Schlösser, F. Serrano, Y. Shinano, J. M. Viernickel, S. Vigerske, D. Weninger, J. T. Witt, and J. Witzig, "The SCIP Optimization Suite 5.0," ZIB Report 17-61, Zuse Institute Berlin (2017). urn:nbn:de:0297-zib-66297.

[39] G. Gamrath, D. Anderson, K. Bestuzheva, W.-K. Chen, L. Eier, M. Gasse, P. Gemander, A. Gleixner, R. L. Gottwald, K. Halbig, G. Hendel, C. Hojny, T. Koch, P. L. Bodic, S. J. Maher, F. Matter, M. Miltenberger, E. Mühmer, B. Müller, M. E. Pfetsch, F. Schlösser, F. Serrano, Y. Shinano, C. Taw_k, S. Vigerske, F. Wegscheider, D. Weninger, and J. Witzig, "The SCIP Optimization Suite 7.0," ZIB Report 20-10, Zuse Institute Berlin (2020). urn:nbn:de:0297-zib-78023.

[40] A. Khajavirad and N. V. Sahinidis, "A hybrid LP/NLP paradigm for global optimization relaxations," *Mathematical Programming Computation*, vol. 10, no. 3, pp. 383–421 (2018). doi: 10.1007/s12532-018-0138-5.

[41] Y. Lin and L. Schrage, "The global solver in the LINDO API," *Optimization Methods & Software*, vol. 24, no. 4–5, pp. 657–668 (2009). doi: 10.1080/10556780902753221.

[42] G. Kocis and I. Grossmann, "Computational experience with DICOPT: Solving MINLP problems in process systems engineering," *Computers & Chemical Engineering*, vol. 13, no. 3, pp. 307–315 (1989). doi: 10.1016/0098-1354(89)850082.

[43] C. Buskens and D. Wassel, "The ESA NLP solver WORHP," in *Modeling and Optimization in Space Engineering*, G. Fasano and J. D. Pinter, Eds., New York, NY: Springer, pp. 85–110 (2013). doi: 10.1007/978-1-4614-4469-5_4.

[44] V. K. Srivastava and A. Fahim, "A two phase optimization procedure for integer programming problems," *Computers & Mathematics with Applications*, vol. 42, no. 12, pp. 1475–1485 (2001). doi: 10.1016/S0898-1221(01)00264-4.

[45] V. K. Srivastava and A. Fahim, "An optimization method for solving mixed discrete-continuous programming problems," *Computers & Mathematics with Applications*, vol. 53, no. 12, pp. 1481–1491 (2007). doi: 10.1016/j.camwa.2006.09.007.

[46] V. K. Srivastava and D. Spinello, "A two-phase combined gradient-tunneling based algorithm for constrained integer programming problems," *Journal of Information & Optimization Sciences*, vol. 36, no. 4, pp. 339–365 (2015). doi: 10.1080/02522667.2014.932092.

[47] V. K. Srivastava and D. Spinello, "Efficiency of robust heuristic gradient based enumerative and tunneling algorithms for constrained integer programming problems," *World Academy of Science, Engineering and Technology International Journal of Mathematical and Computational Sciences*, vol. 11, no. 5, pp. 619–624 (2017).

[48] V. K. Srivastava and D. Spinello, "Multi-objective optimization of combined system reliability and redundancy allocation problem," *World Academy of Science, Engineering and Technology International Journal of Mathematical and Computational Sciences*, vol. 13, no. 8 (2019).

[49] S. M. Roberts and H. I. Lyvers, "The gradient method in process control," *Industrial and Engineering Chemistry*, vol. 53, no. 1, pp. 877–882 (1961).

[50] W. R. Klingman and D. M. Himmelblau, "Nonlinear programming with the aid of a multiple-gradient summation technique," *Journal of the Association for Computing Machinery*, vol. 11, no. 4, pp. 400–415 (1964).

[51] A. V. Levy and A. Montalvo, "The tunneling algorithm for the global minimization of functions," *SIAM Journal on Scientific and Statistical Computing*, vol. 6, no. 1, pp. 15–29 (1985). doi: 10.1137/0906003.

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

[53] I. E. Grossmann, "Overview of mixed-integer nonlinear programming," *Computers and Chemical Engineering*, vol. 29, pp. 2823–2835 (2005).

[54] M. A. Duran and I. E. Grossmann, "An outer approximation algorithm for a class of mixed-integer nonlinear programs," *Mathematical Programming*, vol. 36, pp. 307–339 (1986).

[55] B. Öztürk, "Global and robust optimization for engineering design," Ph.D. dissertation, Massachusetts Institute of Technology (2022).

[56] J. W. Sitopu, H. Mawengkang, and R. S. Lubis, "An improved search approach for solving non-convex mixed-integer nonlinear programming problems," in *4th International Conference on Operational Research (InteriOR)*, IOP Conf. Series: Materials Science and Engineering, vol. 300, pp. 012022 (2018). doi: 10.1088/1757-899X/300/1/012022.

Received June, 2023