The Performance of Penalty Methods on Tree-Seed Algorithm for Numerical Constrained Optimization Problems

Ahmet Cinar[1] and Mustafa Kiran[2]
[1]Department of Computer Engineering, Selçuk University, Turkey
[2]Department of Computer Engineering, Konya Technical University, Turkey

Abstract: *The constraints are the most important part of many optimization problems. The metaheuristic algorithms are designed for solving continuous unconstrained optimization problems initially. The constraint handling methods are integrated into these algorithms for solving constrained optimization problems. Penalty approaches are not only the simplest way but also as effective as other constraint handling techniques. In literature, there are many penalty approaches and these are grouped as static, dynamic and adaptive. In this study, we collect them and discuss the key benefits and drawbacks of these techniques. Tree-Seed Algorithm (TSA) is a recently developed metaheuristic algorithm, and in this study, nine different penalty approaches are integrated with the TSA. The performance of these approaches is analyzed on well-known thirteen constrained benchmark functions. The obtained results are compared with state-of-art algorithms like Differential Evolution (DE), Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC), and Genetic Algorithm (GA). The experimental results and comparisons show that TSA outperformed all of them on these benchmark functions.*

Keywords: *Constrained optimization, penalty functions, penalty approaches, tree-seed algorithm.*

Received January 3, 2019; accepted February 26, 2020
https://doi.org/10.34028/iajit/17/5/13

1. Introduction

Using penalty approaches for constrained optimization is not only the simplest way but also as effective as other constraint handling techniques. Penalty approaches convert constrained optimization problems to unconstrained optimization problems. The main idea is penalizing the unfeasible solutions thus the exploration process goes towards the feasible region. Penalty functions are divided into two separate groups in the literature. These are named as exterior and interior. Interior penalty approaches penalize the feasible solutions, and exterior penalty approaches penalize the infeasible solutions. The infeasible solutions are created easier than feasible solutions by evolutionary computation techniques. Therefore, researchers mostly interested in the exterior penalty approach. In this study, we also interested in these techniques. The significant thing is to tune the penalty factor. If we use a big penalty factor, then the search is tending to local optimums, otherwise, if we use small penalty factor, exploring the feasible region is needed very time-consuming process. Thus, if an efficient penalty factor is found, it is an important improvement to the constrained optimization problem area. Therefore, this subject constantly is studied in the literature. Schoenauer and Xanthakis [41] suggested a new method which handles constraints one by one. At initialization, a random population is created, then the algorithm tries to handle the first constraint. When the first constraint is satisfied, the algorithm deal with the second constraint. This process continues until all constraints are satisfied. This method is specific and some studies [35] show that this method does not solve any problems. Therefore, we did not use this technique in our study. Michalewicz and Attia [36] proposed Genocop II for solving constrained problems. Genocop II handles any type of constraints while Genocop I, the previous version, handles only linear constraints. Michalewicz [35] compared six methods on five test problems. These methods are given in [23, 24, 36, 39, 41] death penalty. Five problems that have different characteristics (linear, nonlinear, quadratic, and polynomial) are solved by these methods. Results show that there is no superior method for solving constrained optimization problems. Michalewicz and Schoenauer [37] prepared a survey on constrained optimization with evolutionary computation methods. In section 5.2 of the Handbook of Evolutionary Computation [6], Smith and Coit [42] discussed the penalty functions very deeply. Coello [16] introduces a self-adaptive penalty approach for constrained optimization. This approach is used not only the total constraint violation but also the number of unsatisfied constraints. These two properties are controlled by two weighting factors. These weighting factors are optimized with another population during the search process. This technique is also a co-evolution penalty approach. Hamida and

Schoenauer [22] improved ASCHEA [21] technique. ASCHEA consists of three main parts, the first is the population-based adaptive penalty function, the second is the mate feasible individuals by infeasible individuals, and the third is segregational selection similar to [19]. Kuri-Morales and Gutiérrez-García [28] integrated five different penalty approaches with the genetic algorithm. The experimental results are examined with statistical analysis information. Coello [15] prepared a comprehensive survey on constraint handling techniques. These techniques discussed deeply and compared positive and negative features. Farmani and Wright [20] proposed a self-adaptive fitness formulation for solving constrained optimization problems. This method requires no parameter tuning. Lemonge and Barbosa [29] proposed a parameterless adaptive penalty approach. This procedure is named as an Adaptive Penalty Method (APM) and it is used for solving structural and benchmark problems. Yeniay [47] collected penalty functions that are found in the literature. This work discusses the main advantages and drawbacks of penalty approaches. Yuchi and Kim [48] proposed a new method that divides the population into two groups. This method labels individuals as feasible or infeasible. After this phase, new children are created with feasible parents. The pipe network optimization problem is solved with a penalty adapting ant algorithm [1]. GA [44] and PSO [2] have properly solved the cost optimization problems that are a type of constrained optimization problems. Babaeizadeh and Ahmad [4] solved 24 constrained benchmark problems with the enhanced artificial bee colony algorithm. Mallipeddi and Suganthan [32] mention the no free lunch theorem and propose the Ensemble of Constraint Handling Techniques (ECHT). ECHT contains four different constraint handling techniques. The superiority of feasible solutions [19], Self-adaptive Penalty [46], ε–Constraint [45] and Stochastic Ranking [40] techniques are used for constraint handling. Liu *et al*. [31] added static and dynamic penalty approaches to the differential search algorithm. The dynamic penalty approach produced more quality solutions than the static penalty approach. Chehouri *et al*. [9] criticize the penalty approaches and suggest a new constraint handling mechanism named as Violation Constraint-Handling-VCH. VCH method is compared with penalty approaches in the literature. This is a parameter-free constraint-handling technique. The VCH is remarkably similar to Deb's rules which use in the work of Babalik *et al*. [5]. The differences are VCH takes account of a number of violated constraints and using elitism. Babalik *et al*. [5] integrated Deb's rules to the tree-seed algorithm for solving constrained optimization problems. Constrained TSA (CTSA) solved well-known thirteen constrained optimization benchmark problems and four engineering design problems. De Castro Rodrigues *et al*. [18] presented a constraint handling method whose name is Extended Balanced Ranking Method (E-BRM). E-BRM is a self-adaptive procedure. E-BRM creates two rank lists for feasible and infeasible solutions. These lists are unified during the exploration process. Metaheuristic algorithms have different inspiration sources like animal behaviors [33, 34], chemical reactions [3] and so on. The main inspiration of TSA [27] is the relationship between seeds and their seeds. TSA is proposed for solving low dimensional unconstrained continuous numerical optimization problems. Cinar and Kiran [10, 12] proposed the parallel version of TSA. Kiran [26] investigated the performance of TSA on constrained optimization which is an engineering design problem (the pressure vessel design problem). Cinar and Kiran [11] studied the effectiveness of search space limitation methods on TSA. TSA is modified for constrained optimization in [5], binary optimization in [13] and discrete optimization in [14].

The remainder of the paper is organized as follows: our study is presented and literature is given in the first section and the constrained optimization is explained in section 2. The basic TSA is introduced in section 3 and the detailed information about penalty approaches is given in section 4. The experimental setup, results and discussions are presented in section 5. Finally, the study is concluded, and a future direction is given in section 6.

2. Constrained Optimization

A Constrained Optimization Problem (COP) is usually defined as follows [31]:

$$Minimize\ z = f(x)$$
$$subject\ to \begin{cases} g_j(x) \le 0, for\ j = 1, \dots, q \\ h_j(x) = 0, for\ j = 1, \dots, m \end{cases} \quad (1)$$

Where $x = (x_1, x_2, \dots, x_n) \in R^n, l_i \le x_i \le u_i, i = 1, \dots, n$, q is the number of total inequality constraint and m is the number of total equality constraints. The objective function f(x) is defined on a search space, S, defined by $S = \{x = (x_1, x_2, \dots, x_n)^T \in R^n : l_i \le x_i \le u_i, i = 1, \dots, n\}$

Let F be the set which contains all those x ∈ S such that the inequality and equality constraints given by Equation (1) are fulfilled. The feasible region is denoted as F. The equality constraints can be transformed to inequality constraints given as follows:

$$\left|h_j(x)\right| \le \xi\ j = 1,2,3, \dots, m \quad (2)$$

ξ is set to 1e-4 in this study, m is the number of total equality constraints. Minimization problems are taken as basis in this article.

3. Tree-Seed Algorithm

TSA was presented by Kiran [27] for solving optimization problems. The relationship between trees

and their seeds are the main inspiration of TSA. Trees and seeds represent the potential solutions for optimization problems. Trees are created randomly in search space at the initialization phase. The number of trees or population is named as "stand size" in TSA. The number of seeds is analyzed in [27], and it is recommended as between 10% and 25% of stand size. Seeds are produced using Equations 3 or 4 for each tree at every generation.

$$Seeds(k) = Trees(i) + \alpha\big(BestTree - Trees(r)\big) \quad (3)$$

$$Seeds(k) = Trees(i) + \alpha\big(Trees(i) - Trees(r)\big) \quad (4)$$

where, Trees(i) is ith tree, Seeds(k) is kth seed of Trees(i), α is a uniformly distributed random number between -1 and 1, BestTree is the best tree obtained so far, Trees(r) is a random tree which is different from the Trees(i). Search Tendency (ST) parameter controls the selection of Equation (3) or Equation (4) ST has a value between 0 and 1. In the course of the iterations, a random number between 0 and 1 is produced and compared with the ST parameter. If this random number is smaller than ST, Equation (3) is used for seed creation, otherwise, Equation (4) is used for seed generation. Equation (3) provides exploitation, and Equation4 provides exploration in TSA. For detailed information for TSA, referenced works [10, 13, 27] can help researchers.

4. Penalty Approaches

Simply, if any constraint is violated, then the penalty value added (for minimization problems) to the objective function in the penalty approach. Three different types of penalty approaches are conducted. These are static, dynamic and adaptive. The general form of using penalty approaches in metaheuristic algorithms is given in Equation (5):

$$f(x) = o(x) + w \times p(x) \quad (5)$$

Where $f(x)$ is the fitness function, $o(x)$ is the objective function, $p(x)$ is the penalty function and w is the coefficient of the penalty. If a solution is feasible, then $p(x)=0$ otherwise, $p(x)$ is calculated as mentioned in the subsections. If the penalty values are constant during the iterations, this type of approaches is named as static, if the coefficient of penalty is changed via iterations, this type of approaches is named as dynamic, and if the evolution process feedbacks the coefficient of penalty, this type of approaches is named as adaptive.

4.1. Static Penalty Approaches the Constant Static Penalty Approach

The basic formulation of the constant static penalty approach is as follows:

$$p(x) = \sum_{i=1}^{m} C_i \delta_i, where \begin{cases} \delta_i = 1, constraint(i) > 0 \\ \delta_i = 0, constraint(i) \le 0 \end{cases} \quad (6)$$

Where C_i is the constant penalty value for constraint i. For simplicity, in this study, we use a unique constant value of all constraints. C is set as 10^9. This technique is named as TSA1 in this study.

- *Sum of the constraint violations approach*

The formulation of the sum of the constraint violations approach is as follows:

$$f(x) = o(x) + \sum_{i=1}^{m} |V_i| \delta_i, where \begin{cases} \delta_i = 1, constraint(i) > 0 \\ \delta_i = 0, constraint(i) \le 0 \end{cases} \quad (7)$$

where V_i is the violation amount of constraint i. This technique is named as TSA2 in this study.

- *Sum of the constraint violations squared approach*

The formulation of the sum of the constraint violations squared approach is as follows:

$$p(x) = \sum_{i=1}^{m} |V_i|^2 \delta_i, where \begin{cases} \delta_i = 1, constraint(i) > 0 \\ \delta_i = 0, constraint(i) \le 0 \end{cases} \quad (8)$$

Where $|V_i|^2$ is the squared violation amount of constraint i. This technique is named as TSA3 in this study.

- *Homaifar et al.'s [23] Static Penalty Approach*

Homaifar *et al*. [23] propose a static penalty approach. In this approach, a multi-stage penalty mechanism is included in the Genetic algorithm. This approach is problem-dependent because m(2s+1) (m is the number of constraints and s is the number of stages) parameters must be determined for the calculation. This is an arbitrary and time-consuming process. Thus, in this study, we did not use this approach in our experiments.

- *Morales and Quezada's Static Penalty Approach*

Morales and Quezada [38] propose a static penalty approach. This approach adds penalty value according to the violated constraint number.

$$p(x) = K - \sum_{i=1}^{s} \frac{K}{m} \quad (9)$$

Where K is a large constant (i.e., 10^9), s is the number of satisfied constraints, m is the number of constraints. This technique is named as TSA4 in this study.

4.2. Dynamic Penalty Approaches

4.2.1. Joines and Houck's Dynamic Penalty Approach

Joines and Houck [24] propose a dynamic penalty approach for the genetic algorithm. The main inspiration of this technique is simulated annealing and calculus-based penalty approach.

$$p(x) = (C \times k)^{\alpha} \times \sum_{i=1}^{m} g_i(x)^{\beta} \quad (10)$$

Where α, β, C are constant parameters. k is the current iteration number, m is the number of total constraints. In this study, we used these values as C=0.5, α=2 and β=2 as in [28]. This technique is named as TSA5 in this study.

4.2.2. Liu *et al.*'s Dynamic Penalty Approach

Liu *et al*. [31] proposed a dynamic penalty approach. This approach depends on two new parameters.

$$p(x) = 10^{1+e^{\frac{20\left(-g+\frac{G}{4}\right)}{G}}} \times \sum_{i=1}^{m} C_i \quad (11)$$

Where, g is the current iteration number, G is the total iteration number, θ_2 and θ_1 are the predefined lower and upper values of the power of 10. In this study, we used these values as $\theta_2=6$ and $\theta_1=2$ like as in Liu *et al*. [31] work. This technique is named as TSA6 in this study.

4.2.3. Kazarlis and Petridis's Dynamic Penalty Approach

Kazarlis and Petridis [25] propose a dynamic penalty approach and named it as Varying Fitness Function (VFF) technique. This dynamic penalty technique contains three crucial parameters. These parameters are A, B and w. A is the severity factor, B is the penalty threshold value and w is the weights of the constraints. The parameter definition process is a substantial problem-dependent. Therefore, in this study, we did not use this technique.

4.2.4. Carlson and Shonkwiler's Annealing Penalty Approach

Carlson and Shonkwiler [8] propose an annealing penalty approach. This technique works as follows:

$$p(x) = e^{\frac{-M}{\frac{1}{\sqrt{t}}}} \quad (12)$$

Where t is the last temperature used in the previous iteration, M is the total violation of constraints. Carlson and Shonkwiler [8] solved the groundwater management problem in their work. When we analyze this technique on standard benchmark minimization problems, it does not produce good solutions and most of the time it found local optimums. Because of this, in this study, we did not use this technique.

4.3. Adaptive Penalty Approaches

4.3.1. Ben Hadj-Alouane and Bean's Adaptive Penalty Approach

Ben Hadj-Alouane and Bean [7] propose an adaptive penalty approach that depends on a penalty value that changes through iterations.

$$p(x) + \lambda(t)\left[\sum_{i=1}^{q} g_i^2(x) + \sum_{j=q+1}^{m} |h_j(x)|\right] \quad (13)$$

$$\lambda(t+1) = \begin{cases} \left(\frac{1}{\beta_1}\right)\lambda(t) if Case1 \\ \beta_2\lambda(t) if Case2 \\ \lambda(t) otherwise \end{cases} \quad (14)$$

where t is the current iteration number, $\lambda(.)$ is the penalty factor, if best individuals in the last k generations are feasible, then Case # 1 occurs, if they are not feasible then Case # 2 occurs. In this method, k, β_1, and β_2 parameters must be carefully selected. This technique is the problem and the parameter-dependent technique. Therefore, in this study, we did not use this technique.

4.3.2. Smith et al.'s Adaptive Penalty Approach

Coit and Smith [17] and Smith and Tate [43] proposed an improved Near Feasibility Threshold (NFT) technique for constrained optimization. NFT uses a threshold value for determining the additional feasible area (NFT-infeasible region). The main formula of NFT is as follows:

$$p(x) = (BestF - Best) \times \left(\left(\frac{\Delta w_i}{NFT_W}\right)^K + \left(\frac{\Delta c_i}{NFT_C}\right)^K\right) \quad (15)$$

Where BestF non-penalized solution value of the best solution (maybe infeasible), Best is the best feasible solution, K is the severity factor, NFT_W is the weight of constraint, NFT_C is the cost of constraint, Δw_i is the weight of the ith solution, Δc_i is the cost of the ith solution. The main disadvantage of this formula is the (*BestF - Best*) part because if a premature convergence occurs BestF is equals to Best so the value of this part is zero. Therefore, the fitness value equals to objective function value and this method does not affect the solution. The second disadvantage is the value of (*BestF - Best*) is huge, then the penalty value is severe so the search does not continue efficiently.

This method includes problem-dependent variables such as NFT_W and NFT_C. Coit and Smith [17] proposed a dynamic NFT as follows:

$$NFT = \frac{NFT_0}{1+\lambda g} \quad (16)$$

Where NFT_O is the starting value, λ is a constant which assures the area between NFT_O and zero, g is the current iteration number. As you see, in the formula two new parameters should be determined for this adaptive NFT. Especially, the selection of the λ parameter very critical for convergence. The main aim is NFT not approach zero either too slowly or too quickly.

In this study, our aim is to analyze problem-independent penalty approaches. Therefore, we use this technique as follows:

$$p(x) = (BestF - Best) \times \left(\left(\frac{\Delta c_i}{\frac{1000}{1+g}}\right)^2\right) \quad (17)$$

This technique, which is given in Equation (17) is named as TSA7 in this study.

For determining the effect of adaptiveness, we change this formula as follows:

$$p(x) = (BestF - Best) \times \left(\left(\frac{\Delta c_i}{1000}\right)^2\right) \quad (18)$$

This technique which is given in Equation (18) is named as TSA8 in this study.

4.3.3. Tessema and Yen's Self Adaptive Penalty Approach

Tessema and Yen [46] propose a self-adaptive penalty approach. This approach has not special parameters. This technique is named as TSA9 in this study.

4.3.4. Farmani and Wright's Self Adaptive Penalty Approach

Farmani and Wright [20] proposed a self-adaptive penalty approach for constrained optimization problems. We could not reproduce the code of this approach via the article. Therefore, in this study, we did not use this technique.

4.3.5. Powell and Skolnick's Adaptive Penalty Approach

Powell and Skolnick [39] presented a method which mapped feasible solutions between -∞ and 1, and infeasible solutions between 1 and +∞. Bevcause this method is similar to the study of Deb [19], we did not use this technique in this study.

4.4. Proposed Method

The blueprint of the proposed method is given in Figure 1.

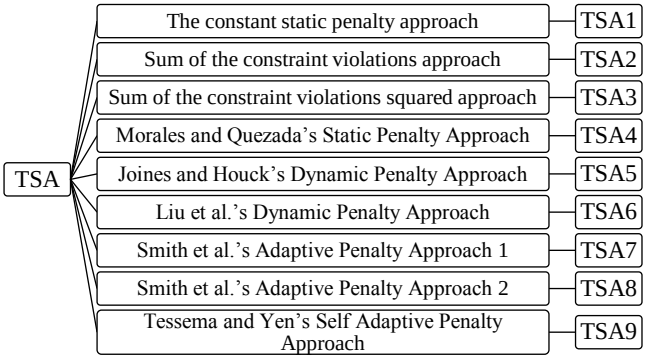

Figure 1. The blueprint of the proposed method.

In each version of TSA, the constrained handling technique is integrated with TSA, and it is applied to solve the constrained numeric benchmark functions.

5. Experiments

5.1. Experimental Setup

The performance investigations of penalty approaches on constrained optimization for tree-seed algorithm have been evaluated using a set of 13 benchmark functions [5]. This set is also named as CEC2006 test case and detailed information can be found in Liang *et al.'s* [30] work. This test set is a comprehensive benchmark suite because in this test set linear, quadratic, cubic, polynomial, and nonlinear functions are included. Babalik *et al.*'s [5] work gives some feedback about the peculiar parameters of TSA. Therefore, the stand size is taken as 20 and ST is taken as 0.2 in this study. The termination condition is the maximum number of function evaluations and it is set to 2.4E+5. The "mean" means "mean of the final obtained fitness functions of 30 different runs".

5.2. Results and Discussions

In the first experiment, 9 different penalty approaches are integrated into TSA and compared to each other. TSA1, TSA2, TSA3 and TSA4 are static penalty approaches, TSA5 and TSA6 dynamic penalty approaches, TSA7, TSA8 and TSA9 are adaptive penalty approaches. The mean results of 30 different runs are given in Table 1.

Table 1. The mean results of 30 different runs.

	Optimum	**TSA1**	**TSA2**	**TSA3**	**TSA4**
G01	-1.50E+01	-1.50E+01	-1.50E+01	-1.52E+01	1.00E+09
G02	-8.04E-01	-8.01E-01	-8.01E-01	-8.03E-01	1.00E+09
G03	-1.00E+00	6.67E+08	-1.00E+05	-9.99E+04	1.00E+09
G04	-3.07E+04	-3.07E+04	-3.22E+04	-3.22E+04	1.00E+09
G05	5.13E+03	2.93E+09	1.60E+03	5.11E+03	1.00E+09
G06	-6.96E+03	3.33E+07	-7.96E+03	-7.95E+03	1.00E+09
G07	2.43E+01	2.46E+01	2.08E+01	2.30E+01	1.00E+09
G08	-9.58E-02	-9.58E-02	-1.54E+03	-1.27E+03	1.00E+09
G09	6.81E+02	6.81E+02	6.79E+02	6.80E+02	1.00E+09
G10	7.05E+03	9.33E+08	2.10E+03	2.10E+03	1.00E+09
G11	7.50E-01	3.00E+08	7.50E-01	5.00E-01	1.00E+09
G12	-1.00E+00	-1.00E+00	-1.00E+00	-1.00E+00	1.00E+09
G13	5.40E-02	2.00E+09	2.83E-01	1.16E-01	1.00E+09
	TSA5	**TSA6**	**TSA7**	**TSA8**	**TSA9**
G01	-1.08E-01	-1.50E+01	-2.75E+02	-2.95E+02	-1.50E+01
G02	-8.01E-01	-8.00E-01	-2.48E-01	-2.54E-01	-8.01E-01
G03	-6.68E+04	-9.80E+04	-1.70E+04	-1.54E+04	-1.00E+05
G04	-3.21E+04	-3.20E+04	-3.21E+04	-3.21E+04	-3.22E+04
G05	8.95E+03	5.21E+03	2.99E+02	1.93E+01	3.20E+03
G06	-7.92E+03	-7.85E+03	-7.75E+03	-7.86E+03	-7.95E+03
G07	2.48E+01	2.45E+01	1.88E+02	8.23E+01	2.45E+01
G08	-1.17E+02	-9.58E-02	-1.15E+03	-1.27E+03	-1.47E+03
G09	6.81E+02	6.81E+02	7.04E+02	3.29E+02	6.81E+02
G10	2.56E+03	2.27E+03	3.35E+03	3.28E+03	2.10E+03
G11	2.03E-01	7.50E-01	9.94E-05	1.68E-05	7.50E-01
G12	-1.00E+00	-1.00E+00	-1.00E+00	-1.00E+00	-1.00E+00
G13	1.06E+00	8.98E+00	1.49E-07	1.91E-09	5.17E-01

For a detailed analysis, we added Table 2, which shows the percentage of the deviation from the optimum solution for the obtained mean results. In Table 2, if the percentage value is bigger than 100, we write N/A in this cell.

Table 2. The percentage of the deviation from the optimum solution for the obtained mean results.

	Optimum	TSA1	TSA2	TSA3	TSA4	TSA5	TSA6	TSA7	TSA8	TSA 9
G01	-1.50E+01	0.00	0.00	1.00	N/A	99.28	0.00	N/A	N/A	0.00
G02	-8.04E-01	0.35	0.34	0.13	N/A	0.38	0.48	69.20	68.34	0.28
G03	-1.00E+00	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
G04	-3.07E+04	0.00	5.05	5.04	N/A	4.58	4.21	4.71	4.74	5.05
G05	5.13E+03	N/A	68.79	0.30	N/A	74.62	1.62	94.17	99.62	37.58
G06	-6.96E+03	N/A	14.37	14.23	N/A	13.71	12.82	11.37	12.97	14.22
G07	2.43E+01	1.04	14.24	5.20	N/A	2.09	0.86	N/A	N/A	0.89
G08	-9.58E-02	0.00	N/A	N/A	N/A	N/A	0.00	N/A	N/A	N/A
G09	6.81E+02	0.00	0.21	0.05	N/A	0.00	0.00	3.43	51.71	0.00
G10	7.05E+03	N/A	70.19	70.19	N/A	63.73	67.85	52.46	53.43	70.18
G11	7.50E-01	N/A	0.01	33.35	N/A	72.93	0.04	99.99	100.00	0.01
G12	-1.00E+00	0.00	0.00	0.00	N/A	0.00	0.00	0.01	0.01	0.00
G13	5.40E-02	N/A	N/A	N/A	N/A	N/A	N/A	100.00	100.00	N/A

To review from a different perspective, the manual ranking was conducted and mean ranks are given in Table 3.

Table 3. The ranking results.

	Optimum	TSA1	TSA2	TSA3	TSA4	TSA5	TSA6	TSA7	TSA8	TSA9
G01	-1.50E+01	1	1	2	4	3	1	4	4	1
G02	-8.04E-01	4	3	1	9	5	6	8	7	2
G03	-1.00E+00	1	1	1	1	1	1	1	1	1
G04	-3.07E+04	1	7	6	8	3	2	4	5	7
G05	5.13E+03	8	4	1	8	5	2	6	7	3
G06	-6.96E+03	8	7	6	8	4	2	1	3	5
G07	2.43E+01	3	6	5	7	4	1	7	7	2
G08	-9.58E-02	1	2	2	2	2	1	2	2	2
G09	6.81E+02	1	3	2	6	1	1	4	5	1
G10	7.05E+03	7	6	6	7	3	4	1	2	5
G11	7.50E-01	7	1	3	7	4	2	5	6	1
G12	-1.00E+00	1	1	1	3	1	1	2	2	1
G13	5.40E-02	2	2	2	2	2	2	1	1	2
Total Ranks		45	44	38	72	38	26	46	52	33

According to Table 3, the best approach is TSA6, the second approach is TSA9. TSA6 is a dynamic penalty approach, and TSA9 is an adaptive penalty approach. At first view, dynamic and adaptive penalty approaches are better than static penalty approaches. TSA5 is another dynamic penalty approach, and it was third. TSA3 is a static penalty approach and the rank point same as TSA5. TSA4 is worst because TSA4 does not improve the starting solutions and at the initialization phase, TSA4 shows premature convergence characteristics on the test functions. The convergence characteristics of the TSA with penalty approaches for the G05 problem are presented in Figure 2. TSA1 and TSA4 could not solve the G05 problem so we do not add these approaches to the convergence graph. Only TSA6 and TSA3 converge properly for the G05 problem. The other approaches do not converge in a reasonable time violate the constraints, so they break the optimum line and trapped in local optimums. TSA6 uses the current iteration number, the total iteration number, and an exponential product of the constraint value. This dynamism puts TSA6 ahead of the others.

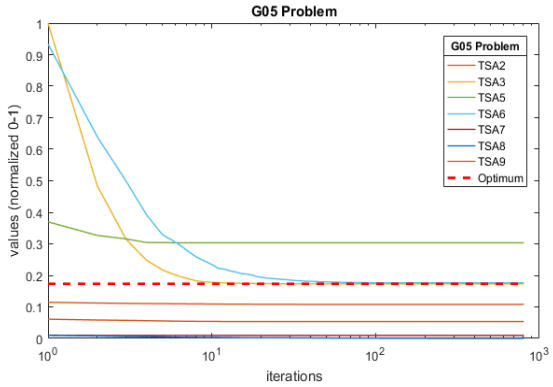

Figure 2. The convergence characteristics of the TSA penalty approaches for G05 problem.

In the second experiment, TSA6 is compared with state-of-art algorithms. The other results are directly taken from [5] and all conditions are the same for a fair comparison. The mean results of TSA6, CTSA, PSO, GA, DE and ABC algorithms are given in Table 4.

Table 4. The comparison results of TSA6, CTSA, PSO, GA, DE and ABC algorithms.

Problem	Optimum	ABC	PSO	GA	DE	CTSA	TSA6
G01	-1.50E+01	-1.50E+01	-1.06E+01	-1.42E+01	-1.42E+01	-1.50E+01	-1.50E+01
G02	8.04E-01	4.80E-01	4.04E-01	7.89E-01	6.66E-01	8.01E-01	8.00E-01
G03	1.00E+00	3.02E+00	1.17E+00	9.76E-01	1.17E+00	1.02E+00	9.80E+04
G04	-3.07E+04	-3.06E+04	-3.07E+04	-3.06E+04	-3.07E+04	-3.07E+04	-3.20E+04
G05	5.13E+03	5.12E+03	5.30E+03	N/A	5.33E+03	5.17E+03	5.21E+03
G06	-6.96E+03	-7.58E+03	-6.96E+03	-6.87E+03	-6.77E+03	-6.96E+03	-7.85E+03
G07	2.43E+01	2.91E+01	2.87E+01	3.50E+01	2.43E+01	2.45E+01	2.45E+01
G08	9.58E-02	6.53E+00	8.47E-02	9.58E-02	9.58E-02	9.58E-02	9.58E-02
G09	6.81E+02	6.84E+02	6.81E+02	6.92E+02	6.81E+02	6.81E+02	6.81E+02
G10	7.05E+03	7.26E+03	8.13E+03	1.00E+04	7.16E+03	7.12E+03	2.27E+03
G11	7.50E-01	7.17E-01	7.63E-01	7.50E-01	9.55E-01	8.00E-01	7.50E-01
G12	1.00E+00	1.00E+00	1.00E+00	1.00E+00	1.00E+00	1.00E+00	1.00E+00
G13	5.40E-02	9.55E-02	1.42E+00	N/A	9.49E-01	9.67E-01	8.98E+00

For a complete analysis, we added Table 5, which shows the percentage of the deviation from the optimum solution for the obtained mean results. In Table 5, if the percentage value is bigger than 100, we write N/A in this cell.

Table 5. The percentage of the deviation from the optimum solution for the mean results.

Problem	Optimum	ABC	PSO	GA	DE	CTSA	TSA6
G01	-1.50E+01	0.14	29.63	5.09	5.06	0.00	0.00
G02	8.04E-01	40.33	49.69	1.87	17.12	0.39	0.48
G03	1.00E+00	N/A	16.75	2.40	16.94	1.58	N/A
G04	-3.07E+04	0.18	0.01	0.24	0.00	0.00	4.21
G05	5.13E+03	0.22	3.35	N/A	3.95	0.89	1.62
G06	-6.96E+03	8.87	0.00	1.29	N/A	0.00	12.82
G07	2.43E+01	19.71	18.25	43.92	0.04	0.80	0.86
G08	9.58E-02	N/A	11.61	0.03	0.03	0.03	0.00
G09	6.81E+02	0.48	0.02	1.68	0.00	0.00	0.00
G10	7.05E+03	2.98	15.31	41.90	1.61	0.95	67.85
G11	7.50E-01	4.39	1.68	0.00	27.27	6.71	0.04
G12	1.00E+00	0.01	0.00	0.00	0.00	0.00	0.00
G13	5.40E-02	77.02	N/A	N/A	N/A	N/A	N/A

The ranking results are given in Table 6.

Table 6. The ranking results.

Problem	Optimum	ABC	PSO	GA	DE	CTSA	TSA6
G01	-1.50E+01	2	5	4	3	1	1
G02	8.04E-01	5	6	3	4	1	2
G03	1.00E+00	5	3	2	4	1	5
G04	-3.07E+04	3	2	4	1	1	5
G05	5.13E+03	1	4	6	5	2	3
G06	-6.96E+03	3	1	2	5	1	4
G07	2.43E+01	5	4	6	1	2	3
G08	9.58E-02	4	3	2	2	2	1
G09	6.81E+02	3	2	4	1	1	1
G10	7.05E+03	3	4	5	2	1	6
G11	7.50E-01	4	3	1	6	5	2
G12	1.00E+00	2	1	1	1	1	1
G13	5.40E-02	1	2	2	2	2	2
Total Ranks		41	40	42	37	21	36

According to Table 6, at first view, TSA6 is better than ABC, PSO, GA and DE, but CTSA is better than TSA6. These results show that TSA6 outperforms the state-of-art algorithms for constrained optimization. TSA integrated with the dynamic penalty approach of Liu *et al*. [31] and this approach is named as TSA6. In this study, TSA6 produces promising and comparable results on benchmark functions. TSA has two peculiar parameters. These are the number of seeds and search tendency. The number of seeds improves the exploitation of the algorithm and search tendency controls the balance between exploration and exploitation. Because of these mechanisms, the TSA outperforms the other algorithms.

6. Conclusions and Future Work

In this work, penalty approaches are used for solving constrained optimization problems with the TSA. Most of the optimization problems have constraints. Metaheuristic algorithms are produced by optimal and near-optimal solutions for constrained optimization problems. TSA is a population-based swarm intelligence algorithm. The penalty approaches are the simplest way to handle constraints. In literature, there are many penalty approaches. These approaches are grouped as static, dynamic and adaptive. The pros and cons of the penalty approaches are discussed in this study. Nine different penalty approaches are integrated TSA and experimental results are conducted on well-known benchmark functions. Experiments show that dynamic and adaptive penalty approaches are better than static penalty approaches. TSA integrated with the dynamic penalty approach of Liu *et al*. [31] and this approach is named as TSA6. TSA6 uses the current iteration number, the total iteration number, and an exponential product of the constraint value. This dynamism puts TSA6 ahead of the others. TSA6 solved G01, G08, G09 and G12 optimally. We also compared TSA variants wit state-of-art algorithms. The results show that, the TSA is better than ABC, PSO, GA and DE. In future works, we will study on new constraint handling methods for metaheuristic algorithms and especially on TSA. As future work, we will examine the sea lion optimization algorithm [33] and will apply the penalty approaches for improving the capabilities of this algorithm.

Acknowledgments

The first author wish to thank Scientific Research Projects Coordinatorship at Selcuk University and The Scientific and Technological Research Council of Turkey for their institutional supports.

References

[1] Afshar M., "Penalty Adapting Ant Algorithm: Application to Pipe Network Optimization," *Engineering Optimization*, vol. 40, no. 10, pp. 969-987, 2008.

[2] Altun A. and Şahman M., "Cost Optimization of Mixed Feeds with the Particle Swarm Optimization Method," *Neural Computing and Applications*, vol. 22, no. 2, pp. 383-390, 2013.

[3] Asmaran M., Sharieh A., and Mahafzah B., "Chemical Reaction Optimization Algorithm to Find Maximum Independent Set in a Graph," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 9, 2019.

[4] Babaeizadeh S. and Ahmad R., "Enhanced Constrained Artificial Bee Colony Algorithm for Optimization Problems," *The International Arab Journal of Information Technology*, vol. 14, no. 2, pp. 246-253, 2017.

[5] Babalik A., Cinar A., and Kiran M., "A Modification of Tree-Seed Algorithm Using Deb's Rules for Constrained Optimization," *Applied Soft Computing*, vol. 63, pp. 289-305, 2018.

[6] Bäck T., Fogel D., and Michalewicz Z., *Handbook of Evolutionary Computation*, Oxford University Press, 1997.

[7] Ben Hadj-Alouane A. and Bean J., "A Genetic Algorithm for the Multiple-Choice Integer Program," *Operations Research*, vol. 45, no. 1, pp. 92-101, 1997.

[8] Carlson S. and Shonkwiler R., "Annealing A Genetic Algorithm over Constraints," *in Proceedings of Systems, Man, and Cybernetics, IEEE International Conference on*, San Diego, pp. 3931-3936, 1998.

[9] Chehouri A., Younes R., Perron J., and Ilinca A., "A Constraint-Handling Technique for Genetic Algorithms Using A Violation Factor," *Journal of Computer Science*, vol. 12, no. 7, pp. 350-362, 2016.

[10] Cinar A. and Kiran M., "A Cuda-based Parallel Programming Approach to Tree-Seed Algorithm," MSc Thesis, Selcuk University, 2016.

[11] Cinar A. and Kiran M., "Boundary Conditions In Tree-Seed Algorithm: Analysis of The Success of Search Space Limitation Techniques In Tree-Seed Algorithm," *in Proceedings of International Conference on Computer Science and Engineering*, Antalya, pp. 571-576, 2017.

[12] Cinar A. and Kiran M., "A Parallel Version of Tree-Seed Algorithm (TSA) within CUDA Platform," *in Proceedings of International Scientific Conference on Applied Sciences*, At Antalya, pp. 174-178, 2016.

[13] Cinar A. and Kiran M., "Similarity and Logic Gate-Based Tree-Seed Algorithms for Binary Optimization," *Computers and Industrial Engineering*, vol. 115, no. pp. 631-646, 2018.

[14] Cinar A., Korkmaz S., and Kiran M., "A Discrete Tree-Seed Algorithm for Solving Symmetric Traveling Salesman Problem," *Engineering Science and Technology, an International Journal*, vol. 22, no. 6, pp. 1169-1200, 2019.

[15] Coello C., "Theoretical and Numerical Constraint-Handling Techniques Used With Evolutionary Algorithms: A Survey of the State of the Art," *Computer Methods in Applied Mechanics and Engineering*, vol. 191, no. 11, pp. 1245-1287, 2002.

[16] Coello C., "Use of A Self-Adaptive Penalty Approach for Engineering Optimization Problems," *Computers in Industry*, vol. 41, no. 2, pp. 113-127, 2000.

[17] Coit D. and Smith A., "Penalty Guided Genetic Search for Reliability Design Optimization," *Computers and Industrial Engineering*, vol. 30, no. 4, pp. 895-904, 1996.

[18] De Castro Rodrigues M., Guimarães S., and De Lima B., "E-BRM: A Constraint Handling Technique to Solve Optimization Problems With Evolutionary Algorithms," *Applied Soft Computing*, vol. 72, pp. 14-29, 2018.

[19] Deb K., "An Efficient Constraint Handling Method for Genetic Algorithms," *Computer Methods in Applied Mechanics and Engineering*, vol. 186, no. 2-4, pp. 311-338, 2000.

[20] Farmani R. and Wright J., "Self-Adaptive Fitness Formulation for Constrained Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 5, pp. 445-455, 2003.

[21] Hamida S. and Schoenauer M., "An Adaptive Algorithm for Constrained Optimization Problems," *in Proceedings of International Conference on Parallel Problem Solving from Nature*, Paris, pp. 529-538, 2000.

[22] Hamida S. and Schoenauer M., "ASCHEA: New Results Using Adaptive Segregational Constraint Handling," *in Proceedings of Evolutionary Computation, CEC'02*, Honolulu, pp. 884-889, 2002.

[23] Homaifar A., Qi C., and Lai S., "Constrained Optimization Via Genetic Algorithms," *Simulation*, vol. 62, no. 4, pp. 242-253, 1994.

[24] Joines J. and Houck C., "On The Use of Non-Stationary Penalty Functions To Solve Nonlinear Constrained Optimization Problems with GA's," *in Proceedings of Evolutionary Computation, IEEE World Congress on Computational Intelligence*, Orlando, pp. 579-584, 1994.

[25] Kazarlis S. and Petridis V., "Varying Fitness Functions in Genetic Algorithms: Studying the Rate of Increase of the Dynamic Penalty Terms," *in Proceedings of International Conference on Parallel Problem Solving from Nature*, Amsterdam, pp. 211-220, 1998.

[26] Kiran M., *in Intelligent and Evolutionary Systems*, Springer, 2016.

[27] Kiran M., "TSA: Tree-Seed Algorithm for Continuous Optimization," *Expert Systems with Applications*, vol. 42, no. 19, pp. 6686-6698, 2015.

[28] Kuri-Morales A. and Gutiérrez-García J., "Penalty Function Methods for Constrained Optimization With Genetic Algorithms: A Statistical Analysis," *in Proceedings of Mexican International Conference on Artificial Intelligence*, Yucatan, pp. 108-117, 2002.

[29] Lemonge A. and Barbosa H., "An Adaptive Penalty Scheme for Genetic Algorithms in Structural Optimization," *International Journal for Numerical Methods in Engineering*, vol. 59, no. 5, pp. 703-736, 2004.

[30] Liang J., Runarsson T., Mezura-Montes E., Clerc M., Suganthan P., Coello C., and Deb K., "Problem Definitions and Evaluation Criteria for The CEC 2006 Special Session on Constrained Real-Parameter Optimization," *Journal of Applied Mechanics*, vol. 41, no. 8, pp. 8-31, 2006.

[31] Liu J., Teo K., Wang X., and Wu C.,"An Exact Penalty Function-Based Differential Search Algorithm for Constrained Global Optimization," *Soft Computing*, vol. 20, no. 4, pp. 1305-1313, 2016.

[32] Mallipeddi R. and Suganthan P., "Differential Evolution With Ensemble of Constraint Handling Techniques for Solving CEC 2010 Benchmark Problemsm," *in Proceedings of Evolutionary Computation, IEEE Congress on*, Barcelona, pp. 1-8, 2010.

[33] Masadeh R., Mahafzah B., and Sharieh A., "Sea Lion Optimization Algorithm," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 5, pp. 388-395, 2019.

[34] Masadeh R., Sharieh A., and Mahafzah B., "Humpback whale Optimization Algorithm Based on Vocal Behavior for Task Scheduling in Cloud Computing," *International Journal of Advanced Science and Technology*, vol. 13, no. 3,

pp. 121-140, 2019.
[35] Michalewicz Z. "Genetic Algorithms, Numerical Optimization, and Constraints," *in Proceedings of 6th International Conference on Genetic Algorithms*, pp. 151-158, 1995.
[36] Michalewicz Z. and Attia N., "Evolutionary optimization of constrained problems," *in Proceedings of 3rd annual conference on Evolutionary Programming*, pp. 98-108, 1994.
[37] Michalewicz Z. and Schoenauer M., "Evolutionary Algorithms for Constrained Parameter Optimization Problems," *Evolutionary Computation*, vol. 4, no. 1, pp. 1-32, 1996.
[38] Morales A. and Quezada C., "A Universal Eclectic Genetic Algorithm for Constrained Optimization," *in Proceedings of 6th European Congress on Intelligent Techniques and Soft Computing*, pp. 518-522, 1998.
[39] Powell D. and Skolnick M., "Using Genetic Algorithms in Engineering Design Optimization with Non-Linear Constraints," *in Proceedings of 5th International Conference on Genetic Algorithms*, San Francisco, pp. 424-431, 1993.
[40] Runarsson T. and Yao X., "Stochastic Ranking for Constrained Evolutionary Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 3, pp. 284-294, 2000.
[41] Schoenauer M. and Xanthakis S., "Constrained GA optimization," *in Proceedings of the 5th International Conference on Genetic Algorithms* Urbana Champaign, pp. 573-580, 1993.
[42] Smith A. and Coit D., *C5.2 of Handbook of Evolutionary Computation in Handbook of Evolutionary Computation*, A Joint Publication of Oxford University Press and Institute of Physics Publishing, 1996.
[43] Smith A. and Tate D., "Genetic Optimization Using A Penalty Function," *in Proceedings of 5th International Conference on Genetic Algorithms*, San Francisco, pp. 499-505, 1993.
[44] Şahman, M., Çunkaş M., İnal Ş., İnal F., Coşkun B., and Taşkiran U., "Cost Optimization of Feed Mixes by Genetic Algorithms," *Advances in Engineering Software*, vol. 40, no. 10, pp. 965-974, 2009.
[45] Takahama T. and Sakai S., "Constrained Optimization by the E Constrained Differential Evolution with an Archive and Gradient-Based Mutation," *in Proceedings of IEEE Congress on Evolutionary Computation*, Barcelona, pp. 1-9, 2010.
[46] Tessema B. and Yen G., "A Self Adaptive Penalty Function Based Algorithm for Constrained Optimization," *in Proceedings of International Conference on Evolutionary Computation*, Vancouver, 246-253, 2006.
[47] Yeniay Ö., "Penalty Function Methods for Constrained Optimization with Genetic Algorithms," *Mathematical and computational Applications*, vol. 10, no. 1, pp. 45-56, 2005.
[48] Yuchi M. and Kim J., "Evolutionary Algorithm Using Feasibility-Based Grouping for Numerical Constrained Optimization Problems," *Applied Mathematics and Computation*, vol. 175, no. 2, pp. 1298-1319, 2006.

Ahmet Cinar was born in Turkey, in 1986. He graduated from the Department of Computer Engineering, Selçuk University, Turkey, in 2009. He received the M.S. degree from the Computer Engineering Department, Selcuk University in 2016. He is currently pursuing his Ph.D. at the Konya Technical University. He is a Research Staff at the Department of Computer Engineering, Selçuk University. His current research interests include swarm intelligence, nature-inspired algorithms, machine learning and artificial intelligence systems.

Mustafa Kiran received the B.S. and Ph.D. degrees in computer engineering from the Institute of Natural and Applied Sciences, Selcuk University, Konya, Turkey, in 2010 and 2014, respectively. He is currently an Associate Professor at the Computer Engineering Department, Konya Technical University. His current research interests include swarm intelligence, evolutionary algorithms, and their real-world applications.