Hybrid Cuckoo Search for constraint engineering design optimization problems

Uroš Mlakar
University of Maribor
Smetanova 17
Maribor, Slovenia
uros.mlakar@um.si

ABSTRACT

This paper investigates, how the hybrid self-adaptive Cuckoo Search algorithm (HSA-CS) behaves, when confronted with constraint engineering design optimization problems. These problems are commonly found in literature, namely: welded beam, pressure vessel design, speed reducer, and spring design. The obtained results are compared to those found in literature, where the HSA-CS achieved better or comparable results. Based on results, we can conclude, that the HSA-CS is suitable for use in real-life engineering applications.

Keywords

design optimization, hybridization, cuckoo search, self-adaptation

1. INTRODUCTION

There is a increasing rate in the research community for developing new constrained optimization algorithms. Therefore suitable problems must be used, to show the effectiveness, efficiency and convergence of these new algorithms. Such problems are usually mathematical problems like the CEC competition problems, but also engineering design optimization problems are adopted in the specialized literature. Many researchers have studied these problems, by applying a wide range of different optimization methods such as Quadratic Programming [5], Simulated Annealing [12], Genetic Algorithms [9], and Swarm Intelligence [2, 3, 6, 7, 1]. The listed algorithms are among the most used in the literature. The design optimization problems usually have a non-linear objective function and constraints, while the design variables are often a combination of discrete and continuous. The hardest part of finding the optimal solution for these problems is directly related to the constraints, which are imposed on the problem.

Since SI based algorithms are getting a lot of attention in the past couple of years, our aim was to test the behaviour of a novel hybrid self-adaptive Cuckoo Search [8] (HSA-CS) on the design optimization problems. The rest of the paper is organized as follows. In Section 2 some of related work, mostly that of SI-based algorithms is presented. Section 3 is dedicated to Cuckoo Search (CS) and the used self-adaptive hybrid Cuckoo Search algorithm, where an emphasis is on describing the main differences from the original CS. Section 4 deals with describing the design optimization problems, and then in Section 5 the obtained results are presented. In Section 6 the results obtained by the HSA-CS is compared to those in the literature, and the paper is concluded in Section 7.

2. RELATED WORK

Since the HSA-CS belongs to the SI-based algorithms, it would only be reasonable to review the literature from this point of view. Akay and Karaboga [1] presented an artificial bee colony (ABC) algorithm, with a very simple constraint handling method. This method is biased to choose feasible solutions rather than those, which are infeasible. Gandomi et al. [7] use a bat algorithm for solving constraint optimization problems. Their results indicate that their method obtained better results, compared to those in literature. Another ABC algorithm was proposed by Brajevic and Tuba [3]. The upgraded ABC algorithm enhances fine-tuning characteristics of the modification rate parameter and employs modified scout bee phase of the ABC algorithm. Baykasoglu and Ozsoydan [2] presented an adaptive firefly, enhanced with chaos mechanisms. The adaptivity is focused on on the search mechanism and adaptive parameter settings. They report that some best results found in literature, were improved with their method. Bulatović [4] applied the improved cuckoo search (ICS) for solving constrained engineering problems, which produces better results than the original cuckoo search (CS). Their improvements lie in the dynamic changing of the parameters of probability and step size. Yang et al. [11] utilized a multi-objective CS (MOCS) for the beam design problem and disc brake problems. They conclude that the proposed MOCS is efficient on problems with complex constraints.

3. CUCKOO SEARCH

Cuckoo search is a stochastic population-based optimization algorithm proposed by Yang and Deb in 2009 [10]. It belongs in the SI-based algorithm family, and it is inspired by the natural behaviour of some cuckoo species in nature. To trap the behavior of cuckoos in nature and adapt it to be suitable for using as a computer program the authors [10] idealized three rules:

- Each cuckoo lays one egg, and dumps it in a randomly chosen nest,
- Nests with high-quality egg, will be carried over to the next generations,
- Any egg laid by a cuckoo, may be discovered by the host bird with a probability of $p_a \in (0, 1)$. When an egg is discovered, the host bird may get rid of it or simply abandon the nest and build a new one.

Each solution in the population of the cuckoo search algorithm corresponding to a cuckoo nest, represents the position of the egg in the search space. This position can be mathematically defined:

$$\mathbf{x}_i = \{x_{i,j}\}, \ for\ i = 1, \ldots, Np\ and\ j = 1, \ldots, D, \quad (1)$$

where Np represents the population size, and D the dimension of the problem to be solved.

Generating new solutions in the CS is done by executing a random walk, with the use of the Levy flight distribution:

$$\mathbf{x}_i = \mathbf{x}_i + \alpha L(s, \lambda). \quad (2)$$

The term $L(s, \lambda)$ determines the characteristic scale, and $\alpha > 0$ denotes the scaling factor of the step size s.

3.1 Hybrid self-adaptive Cuckoo Search

According to [8] the CS was modified by adding the following mechanisms: balancing of the exploration strategies within the CS, self-adaptation of the parameters, and population reduction. The used exploration employed by the HSA-CS are:

- random long distance exploration,
- stochastic short-distance exploration, and
- stochastic moderate-distance exploration.

The listed strategies have an impact on how the trial solution will be generated. The random long distance exploration is implemented as the abandon operator. The second strategy improves the current solution by using a local random walk, with the help of Levy flights (Eq. 2). The last strategy is borrowed from the DE algorithm. Additionally the last strategy adds a crossover operation to the CS algorithm. These execution of these strategies is controlled by a single parameter.

As was stated all parameters are fully self-adaptive, except the starting population size, which must be experimentally defined. Additionally the strategy balancing probability, the abandon rate, and the elitist parameter (controls whether a random of best solution is taken as the basis trial vector calculation) are determined by the user. Lastly the population reduction is implemented by using a simple linear reduction.

It was proven by the authors of the HSA-CS, that the biggest impact on the results has the inclusion of multiple strategies, than followed by self-adaptation. Population reduction did not have a big impact on the results. For more information about HSA-CS readers are referred to [8].

4. CONSTRAINED DESIGN OPTIMIZATION PROBLEMS

The following design optimization problems have been used in this study: welding beam, pressure vessel design, spring design, and speed reducer design. The used problems are thoroughly presented and formally defined in the remainder of this section.

4.1 Welding beam

The goal of this problem is to design a welded beam subject to minimum cost, subject to some constraints. The problem consists of four design variables, with the objective is to find the minimum fabrication cost, with constraints of shear stress τ, bending stress σ, buckling load P_c, and end deflection on the beam δ. The mathematical model can be formulated as follows:

$$f(\mathbf{x}) = 1.10471x_1^2x_2 + 0.04811 * x_3x_4(14 + x_2), \quad (3)$$

subject to:

$$g_0 : \tau - 13600 \leq 0,\ g_1 : \sigma - 30000 \leq 0,\ g_2 : x_1 - x_4 \leq 0,$$
$$g_3 : 0.10471x_1^2{+}(0.04811x_3x_4(14{+}x_2)){-}5 \leq 0,\ g_4 : 0.125{-}x_1 \leq 0,$$
$$g_5 : \delta - 0.25 \leq 0,\ g_6 : 6000 - P_c \leq 0, \quad (4)$$

where

$$\tau = \sqrt{\tau_1^2 + 2\tau_1\tau_2\frac{x_2}{2R} + \tau_2^2},\ \tau_1 = \frac{6000}{\sqrt{2}x_1x_2},\ \tau_2 = \frac{MR}{J},$$
$$M = 6000(14 + \frac{x_2}{2}),\ R = \sqrt{\frac{x_2^2}{4} + (\frac{x_1 + x_3}{2})^2},$$
$$J = 2(\sqrt{2}x_1x_2(\frac{x_2^2}{12}) + (\frac{x_1 + x_3}{2})^2),\ \sigma = \frac{504 * 10e^3}{x_4x_3^2},$$
$$\delta = \frac{65856 * 10e^3}{3 * 10e^6x_4x_3^3},$$
$$P_c = \frac{(12.039 * 10e^6\sqrt{(x_3^2x_4^6)/36)}}{196}(1 - \frac{x_3\sqrt{\frac{3*10e^6}{48*10e^6}}}{28.0}) \quad (5)$$

The design variables are bounded as: $0.1 \leq x_2, x_3 \leq 10$, and $0.1 \leq x_1, x_4 \leq 2$.

4.2 Pressure vessel design

The idea of this problem is designing a compressed air storage design, with a working pressure of 1000 psi and and minimum volume of 750 ft^3. The problem is described using four variables, which represent shell thickness, spherical head thickness, radius and length of the shell. The objective of the problem is minimizing the manfuacturing cost of the pressure vessel, and can be formulated as:

$$f(\mathbf{x}) = 0.6224x_1x_3x_4{+}1.7781x_2x_3^2{+}3.1661x_1^2x_4{+}19.84x_1^2x_3, \quad (6)$$

subject to

$$g_0 : -x_1 + 0.0193x_3 \leq 0,\ g_1 : -x_2 + 0.00954x_3 \leq 0,$$
$$g_2 : -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \leq 0,\ g_3 : x_4 - 240 \leq 0. \quad (7)$$

The bounds of the design variables are: $0.0625 \leq x_1, x_2 \leq 99 * 0.0625$, and $10 \leq x_3, x_4 \leq 200$.

4.3 Spring design

The spring design optimization problem deals with an optimal design of a tension spring. The problem consists of three variables, which are the number of spring coils, winding diameter, and wire diameter. The objective is to minimize the weight of the spring, subject to minimum deflection, surge frequency, shear stress, and limits on the outside diameter. Mathematically it can be formulated as:

$$f(\mathbf{x}) = (x_3 + 2) * x_1^2 * x_2, \tag{8}$$

subject to the following constraints:

$$g_0 : 1 - \frac{x_2^3 x_3}{71785 x_1^4} \leq 0,\ g_1 : \frac{4x_2^2 - x_2 x_3}{12566(x_2 x_3^3 - x_3^4)} + \frac{1}{5108 x_3^2} - 1 \leq 0,$$
$$g_2 : 1 - \frac{140.45 x_1}{x_2^2 x_3} \leq 0, g_3 : (x_1 + x_2) - 1.5 \leq 0 \tag{9}$$

The search space of design variables are limited as: $0.05 \leq x_1 \leq 2$, $0.25 \leq x_2 \leq 1.3$, and $2 \leq x_3 \leq 15$.

4.4 Speed reducer

This problem deals with a minimum weight design of a speed reducer, subject to bending stress of the gear teeth, surface stress, stresses in the shafts, and transverse deflections of the shafts. This problem is formulated with seven design variables, using the following mathematical definition:

$$f(\mathbf{x}) = 0.7854 x_1 x_2^2 (3.3333 x_3^2 + 14.9334 x_3 - 43.0934) - 1.508 x_1 (x_6^2 + x_7^2) + 7.477(x_6^3 + x_7^3) + 0.7854(x_4 x_6^2 + x_5 x_7^2), \tag{10}$$

subject to:

$$g_0 : \frac{27.0}{x_1 x_2^2 x_3} - 1.0 \leq 0,\ g_1 : \frac{397.5}{x_1 x_2^2 x_3^2} - 1.0 \leq 0,$$
$$g_2 : \frac{1.93 x_4^3}{x_2 x_3 x_6^4} - 1.0 \leq 0,\ g_3 : \frac{1.93 x_5^3}{x_2 x_3 x_7^4} - 1.0 \leq 0,$$
$$g_4 : \frac{\sqrt{(\frac{745 x_4}{x_2 x_3})^2 + 16.9 * 10e^6)}}{(110 x_6^3)} - 1.0 \leq 0,$$
$$g_5 : \frac{\sqrt{(\frac{745 x_5}{x_2 x_3})^2 + 157.5 * 10e^6)}}{(85 x_7^3)} - 1.0 \leq 0,$$
$$g_6 : \frac{x_2 x_3}{40} - 1 \leq 0,\ g_7 : \frac{5 x_2}{x_1} - 1 \leq 0,\ g_8 : \frac{x_1}{12 x_2} - 1 \leq 0,$$
$$g_9 : \frac{1.5 x_6 + 1.9}{x_4} - 1 \leq 0,\ g_{10} : \frac{1.1 x_7 + 1.9}{x_5} - 1.0 \leq 0. \tag{11}$$

The search of the design variables is defined as:
$(2.6, 0.7, 17.0, 7.3, 7.8, 2.9, 5.0)^T \leq \mathbf{x} \leq (3.6, 0.8, 28.0, 8.3, 8.3, 3.9, 5.5)^T$

5. RESULTS

The HCS-SA was applied to solve the design optimization problems, which were described in the previous section. To provide for a fair comparison with the literature, the number of function evaluations was set to 50000, as advised in [2], where the authors determined, that such a number is an average value for function evaluations found in literature. The parameters of HSA-CS were set according to the authors in [8], while the population size was varied as: $Np = 30$ for welded beam, spring design, and speed reducer, while for pressure vessel $Np = 50$. Each experiment was replicated 50 times, thus the results reported here are the average of those runs.

Table 1 holds the results of the experiments. For each problem the minimum (min), maximum (max), mean, median (md), and standard deviation (std) values are reported.

The results in Table 1 indicate that the HSA-CS was able to find the same solution for the welded beam and speed reducer problems in all 50 runs of the algorithm. On the contrary, the HSA-CS had trouble in converging towards a single solution.

6. DISCUSSION

In this section we analyze the results from our experiments and compare them to those found in the literature. For this purpose a Table 2 is provided, where results from literature are gathered. It can be seen, that the HSA-CS achieved competitive results if not better results on all test optimization problems. For the welded beam problem our method and the method in [2] converged to a single solution, whereas other methods were not as successful. It is also hard to say, which method performed the best, since the findings in other papers are reported only to 6 digits. On the pressure vessel problem, the HSA-CS achieved similar results as for the welded beam problem. Based on the mean value the only competitive method was again the one proposed in [2]. HSA-CS acheived the best results for the speed reducer. Again, like for the welded beam, the results were unanimous, converging to a single solution, which was also the smallest based on mean value. Good results were also obtained on the spring design problem, were the HSA-CS had the smallest std value over the 50 runs, while obtaining good results based on the mean value. We can conclude the HSA-CS would be suitable for use in real-life constraint optimization problems.

7. CONCLUSION

This paper investigated the recently proposed HSA-CS algorithm, on four well known engineering design optimization problems with constraints. The problems at hand were: welded beam, pressure vessel design, speed reducer design, and spring design. The obtained results were compared to the some state-of-the-art methods, were the HSA-CS performed very well, thus we can conclude it would be suitable for use in real-life engineering applications.

8. REFERENCES

[1] Bahriye Akay and Dervis Karaboga. Artificial bee colony algorithm for large-scale problems and engineering design optimization. *Journal of Intelligent Manufacturing*, 23(4):1001–1014, 2012.

[2] Adil Baykasoğlu and Fehmi Burcin Ozsoydan. Adaptive firefly algorithm with chaos for mechanical design optimization problems. *Applied Soft Computing*, 36:152–164, 2015.

Table 1: Results obtained by the HSA-CS on the four design optimization problems.

Problem	min	max	mean	md	std
Welded beam	1.724852309	1.724852309	1.724852309	1.724852309	0
Pressure vessel	6059.714	6410.087	6095.32	6059.714	88.270
Speed reducer	2996.218	2996.218	2996.218	2996.218	0
Spring design	0.01266523	0.0129087	0.01269661	0.01267089	5.402618 $*10e^{-5}$

Table 2: Comparison with state-of-the-art.

Welded beam				
Method	min	max	mean	std
Gandomi et al [7]	1.7312	2.3455793	1.8786560	0.2677989
Akay et al.[1]	1.724852	–	1.741913	0.031
Brajevic et al. [3]	1.724852	–	1.724853	0.0000017
Baykasoglu et al. [2]	1.724852	1.724852	1.724852	0
This study	1.724852309	1.724852309	1.724852309	0
Pressure vessel				
Method	min	max	mean	std
Gandomi et al [6]	6059.714	6495.347	6447.736	502.693
Akay et al.[1]	6059.714736	–	6245.308144	205.00
Brajevic et al. [3]	6059.714335	–	6192.116211	204
Baykasoglu et al. [2]	6059.71427196	6090.52614259	6064.33605261	11.28785324
This study	6059.714	6410.087	6095.32	88.270
Speed reducer				
Method	min	max	mean	std
Gandomi et al [6]	3000.9810	3009	3007.1997	4.9634
Akay et al.[1]	2997.058412	–	2997.058412	–
Brajevic et al. [3]	2994.471066	–	2994.471072	0.00000598
Baykasoglu et al. [2]	2996.372698	2996.669016	2996.514874	0.09
This study	2996.218	2996.218	2996.218	0
Spring design				
Method	min	max	mean	std
Gandomi et al [7]	0.01266522	0.0168954	0.01350052	0.001420272
Akay et al.[1]	0.012665	–	0.012709	0.012813
Brajevic et al. [3]	0012665	–	0.012683	0.00000331
Baykasoglu et al. [2]	0.0126653049	0.0000128058	0.0126770446	0.0127116883
This study	0.01266523	0.0129087	0.01269661	$5.402618*10e^{-5}$

[3] Ivona Brajevic and Milan Tuba. An upgraded artificial bee colony (abc) algorithm for constrained optimization problems. *Journal of Intelligent Manufacturing*, 24(4):729–740, 2013.

[4] Radovan R Bulatović, Goran Bošković, Mile M Savković, and Milomir M Gašić. Improved cuckoo search (ics) algorthm for constrained optimization problems. *Latin American Journal of Solids and Structures*, 11(8):1349–1362, 2014.

[5] JZ Cha and RW Mayne. Optimization with discrete variables via recursive quadratic programming: Part 2—algorithm and results. *Journal of Mechanisms, Transmissions, and Automation in Design*, 111(1):130–136, 1989.

[6] Amir Hossein Gandomi, Xin-She Yang, and Amir Hossein Alavi. Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. *Engineering with computers*, 29(1):17–35, 2013.

[7] Amir Hossein Gandomi, Xin-She Yang, Amir Hossein Alavi, and Siamak Talatahari. Bat algorithm for constrained optimization tasks. *Neural Computing and Applications*, 22(6):1239–1255, 2013.

[8] Uroš Mlakar, Iztok Fister Jr., and Iztok Fister. Hybrid self-adaptive cuckoo search for global optimization. *Swarm and Evolutionary Computation*, 29:47 – 72, 2016.

[9] Shyue-Jian Wu and Pei-Tse Chow. Genetic algorithms for nonlinear mixed discrete-integer optimization problems via meta-genetic parameter optimization. *Engineering Optimization+ A35*, 24(2):137–159, 1995.

[10] Xin-She Yang and Suash Deb. Cuckoo search via lévy flights. In *Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on*, pages 210–214. IEEE, 2009.

[11] Xin-She Yang and Suash Deb. Multiobjective cuckoo search for design optimization. *Computers & Operations Research*, 40(6):1616 – 1624, 2013. Emergent Nature Inspired Algorithms for Multi-Objective Optimization.

[12] Chun Zhang and Hsu-Pin Wang. Mixed-discrete nonlinear optimization with simulated annealing. *Engineering Optimization*, 21(4):277–291, 1993.