A Firefly Dynamic Penalty Approach for Solving Engineering Design Problems

Rogério B. Francisco[*], M. Fernanda P. Costa[†] and Ana Maria A.C. Rocha[**]

[*]*Escola Superior de Tecnologia e Gestão de Felgueiras, 4610-156 Felgueiras, Portugal*
[†]*Centre of Mathematics, University of Minho, 4710-057 Braga, Portugal*
[**]*Algoritmi R&D Centre, School of Engineering, University of Minho, 4710-057 Braga, Portugal*

Abstract.
Firefly Algorithm is a recent swarm intelligence method, inspired by the social behavior of fireflies, based on their flashing and attraction characteristics [1, 2]. In this paper, we analyze the implementation of a dynamic penalty approach combined with the Firefly algorithm for solving constrained global optimization problems. In order to assess the applicability and performance of the proposed method, some benchmark problems from engineering design optimization are considered.

Keywords: Global optimization, Firefly Algorithm, Penalty function, Constrained optimization, Engineering Design Problems
PACS: 02.60.Pn

INTRODUCTION

The Firefly is a stochastic population-based global optimization algorithm developed to solve global optimization problems with simple bounds. The Firefly Algorithm (FA) is inspired by the collective behavior of fireflies, specifically in how they attract each other. Previous studies have demonstrated that the FA obtained good results, indicating its superiority over some bio-inspired methods [1, 2]. Due to the good results produced, the FA has been used in several applications [3, 4].

This paper aims to illustrate the behavior of a dynamic penalty framework for solving nonconvex constrained global optimization problems. To promote that a global optimal solution is obtained the subproblems are solved by a stochastic population-based global FA. The mathematical formulation of the problem to be addressed has the form:

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & g_l(x) \leq 0,\, l = 1,\ldots,m \\ & x \in \Omega, \end{array} \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ and $g : \mathbb{R}^n \to \mathbb{R}^m$ are nonlinear continuous functions, possibly nondifferentiable, and $\Omega = \{x \in \mathbb{R}^n : -\infty < l \leq x \leq u < \infty\}$ is a bounded feasible region. Problems with equality constraints $h(x) = 0$ can be reformulated in the above form as $h(x) - \varepsilon \leq 0$ and $-h(x) - \varepsilon \leq 0$, where ε is a small positive value. We do not assume that the objective function f and the constrained functions g_l are convex. Since the optimization problem may be a nonconvex one, many local minima may exist in the feasible region. The class of global optimization problems arises frequently in some engineering applications and derivative-free methods reveal to be the most appropriate to solve them [3, 4, 5, 6].

The most common approach for solving constrained optimization problems is based on penalty methods [7, 8, 9]. Most of these approaches seeks for the solution of problem (1) by replacing the constrained problem by a sequence of unconstrained subproblems, i.e., a sequence of penalty functions. A penalty function consists of the objective function f *plus* penalty terms (one for each constraint), which are positive when the constraints are violated and zero otherwise. The penalty terms are multiplied by some positive penalty parameter. By driving the penalty parameter to infinity, the constraint violations are penalized with increasing severity, thereby forcing the minimizer of the penalty function closer and closer to the feasible region of (1). Penalty methods are simple to implement and take advantage of the very powerful unconstrained minimization methods that have been developed recently. Different penalty functions have been suggested and can also be classified based on the way the penalty terms are added: death, static, dynamic, annealing and adaptive [5, 6, 10]. Although penalty functions are very simple and easy to implement, they often require several penalty parameters to be chosen heuristically by users. Most penalty parameters are problem dependent and need prior knowledge of the degree of constraint violation present in a problem. To address this concerning issue, a dynamic penalty function was suggested recently where the computation of the penalty parameter depends on the current iteration number [11]. In this paper we intend to analyze the performance of a dynamic penalty approach combined with the stochastic FA to solve constrained global optimization problems. Some preliminary results are presented when solving a benchmark set of engineering design problems.

The remainder of this paper is as follows. Firstly, a briefly description of the Firefly algorithm and the proposed penalty approach is presented. Then, the experimental results are reported and the, finally some remarks are provided.

FIREFLY ALGORITHM

The Firefly optimization algorithm was first introduced by [1, 2] and is based on the three main rules: i) all fireflies are unisex, meaning that all the elements of a population can attract each other; ii) the attractiveness between fireflies is proportional to their brightness. For any two flashing fireflies, the one with less bright will move towards the brighter one. If no one is brighter than a particular firefly, it moves randomly. Attractiveness is proportional to the brightness which decreases with increasing distance between fireflies; iii) the brightness or light intensity of a firefly is associated with the encoded objective function to be optimized. The main ideas of FA are related with the light intensity emitted by each firefly and the degree of attractiveness that is generated between two fireflies. The attractiveness β of the firefly i depends on the light intensity seen by an adjacent firefly j and on the distance between themselves and is given by $\beta = \beta_0 \exp\left(-\gamma \|x^i - x^j\|^2\right)$, where β_0 is the attraction parameter when the distance is zero. The movement of a firefly i towards another brighter firefly j is computed as follows:

$$x^i = x^i + \beta(x^j - x^i) + \alpha\varepsilon^i,$$

where the second term is due to the attraction, while the third term is a randomization with α being the randomization parameter. Here, ε^i is a vector of random numbers generated by a Uniform distribution. Note that the third term can be extended to other distributions, such as the Gaussian distribution or the Lévy distribution.

DYNAMIC PENALTY APPROACH

Most penalty approaches define a sequence of penalty functions, which are defined by adding to the objective function penalty terms, one for each constraint, multiplied by some positive penalty parameter. Each penalty function aims at penalizing infeasible solutions by increasing their fitness values proportionally to their level of constraint violation. Unfortunately, the penalty parameters required by the penalty function are often problem dependent and chosen by priori knowledge by users.

In the proposed penalty approach a dynamic penalty function is implemented, herein denoted by $\Phi_t(.)$, where the penalty parameters should vary dynamically along the search according to exogenous schedule, and is defined by:

$$\Phi_t(x) = f(x) + \mu(t) \sum_{l=1}^{m} (v_l(x))^{\theta(v_l(x))} \tag{2}$$

where $\mu(.)$ is a dynamically modified penalty parameter and $v_l(x) = \max\{0, g_l(x)\}$ is the level of constraints violation, for $l = 1, \ldots, m$ [10]. They suggested the penalty parameter update given by

$$\mu(t) = (C \times t)^a \tag{3}$$

where t represents the iteration number, and the constants have the values of $C = 0.5$ and $a = 2$. Another interesting and quite efficient rule, found in [11], is given by

$$\mu(t) = D \times t\sqrt{t}, \tag{4}$$

where the constant D is set to 1. Note that, the penalty parameter does not depend on the number of constraints although the pressure on infeasible solutions increases as t increases. See, examples of dynamic penalties in [11]. The power of the constraint violation, $\theta(.)$, is a constant function depending on the violation: $\theta(z) = 1$ if $z \leq 1$, and $\theta(z) = 2$, otherwise. It is worth pointing out that a quadratic penalty function can be obtained from (2), considering the power of the constraint violation, $\theta(.)$, as a constant function $\theta(.) = 2$.

The general framework for algorithms based on penalty functions is composed by an outer cycle where, at each iteration t, an unconstrained subproblem is solved for a fixed value of the penalty parameter. Our proposal for finding an approximate global minimizer of each subproblem (2), is based on the stochastic population-based FA.

NUMERICAL EXPERIMENTS

In order to evaluate the performance of the embedded Firefly algorithm in the dynamic penalty approach and assess its effectiveness eight benchmark engineering problems are used. The characteristics of the selected engineering design problems can be found in [12], where all of them have simple bounds and inequality constraints.

In the proposed firefly dynamic penalty approach, the movement of the firefly i towards a brighter firefly j considers the Lévy distribution in the computation of the randomization term,

$$x^i = x^i + y^i \text{ with } y^i = \beta(x^j - x^i) + \alpha L(x^1)\sigma_x^i,$$

where $L(x^1)$ is a random number from the Lévy distribution centered at x^1, the position of the brightest firefly, with an unitary standard deviation, and σ_x^i is a vector that represents the variation around x^1 (and based on real position x).

The condition used to stop the algorithm is the maximum number of outer iterations set to 15. Each subproblem is solved by FA for maximum number of 10 iterations. A population of 40 fireflies is used and each engineering design problem is solved 30 times. In the following, a small set of different penalty parameter updates is tested for some fixed constants in order to assess their efficiency in the proposed Firefly dynamic penalty approach. Thus, to update the penalty parameter, the formula (3) and (4) are used, for different values of the constants (C,a) and D, namely: $(C,a) = (0.5,2)$ and $(C,a) = (2,4)$ in (3) and $D = 10$ in (4). In the experiments, two dynamic penalty parameter updates in the context of the dynamic quadratic penalty function are also tested: $\mu(t) = t^t$ and $\mu(t)10^t$, where t is the iteration number.

Figure 1 plots the evolution of the penalty parameter value along the iterations. Figure 2 plots the performance profile for the best function values obtained concerning Firefly dynamic penalty approach when using the different penalty parameter updates depicted in Figure 1.

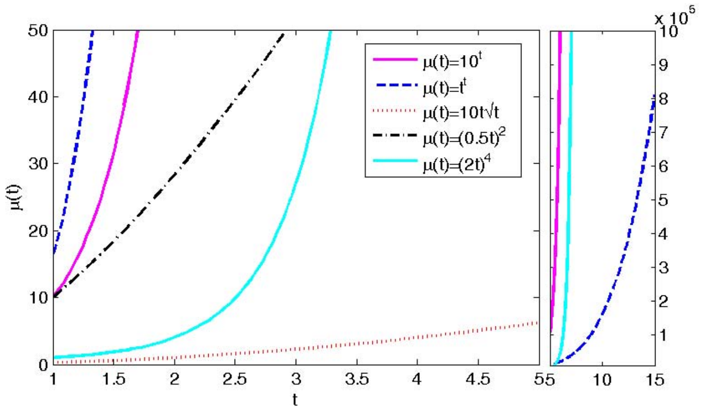

FIGURE 1. Penalty parameter updates.

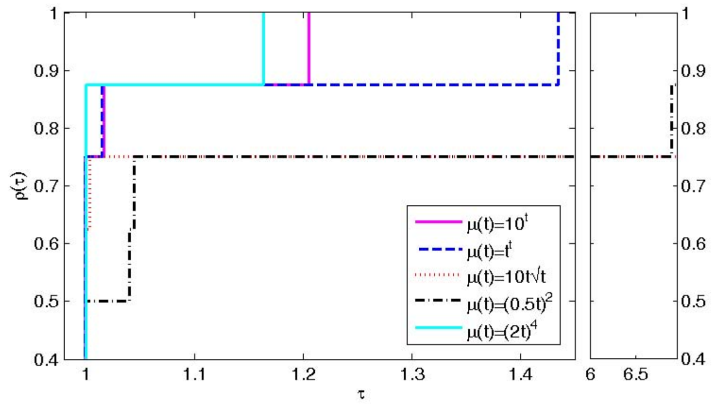

FIGURE 2. Performance profiles of the Firefly dynamic penalty approach when using different penalty parameter updates.

The profiles show that the best behavior of the Firefly dynamic penalty approach is obtained when using the penalty parameter update $\mu(t) = (2t)^4$, and outperform the two versions of the dynamic quadratic penalty function, as well as, the versions that use $\mu(t) = 10t\sqrt{t}$ and $\mu(t) = (0.5t)^2$. These last two versions are less robust than the others and this is due to the fact that these updates are only able to reach small values for the penalty parameters in 15 iterations of the outer cycle (see Figure 1). The results obtained by the Firefly dynamic penalty approach using the penalty parameter update $\mu(t) = (2t)^4$ are reported in Table 1, where in the first column the name of the problem is shown, followed by the dimension of the problem, n; the best objective function value, f_{best}; the worst objective function value, f_{worst}; and the standard deviation of the objective function values, SD; during the 30 runs. The remain columns refer to average values (over 30 runs) of the average number of function evaluations, Nfe, the average function values, f_{avg}, and a measure of the constraint violation, viol.

TABLE 1. Numerical results of the Firefly dynamic penalty approach using $\mu(t) = (2t)^4$.

Problem	n	f_{best}	f_{worst}	SD	Nfe	f_{avg}	viol
Vessel	4	6449.9342	7319.5209	2.1324E+02	22345	7100.1568	0.0000E+00
Brake	4	0.1274	0.1274	5.6460E-17	27424	0.1274	0.0000E+00
4-Bar	4	1400.0000	1400.0000	0.0000E+00	30198	1400.0000	0.0000E+00
Speed	7	2994.7223	3020.0636	5.6653E+00	21589	3000.0998	0.0000E+00
Tubular	2	26.5313	26.5366	1.0748E-03	21924	26.5317	0.0000E+00
Spring	3	0.0127	0.0154	6.1174E-04	21485	0.0134	0.0000E+00
3-Bar	2	263.8958	263.9895	1.9185E-02	22572	263.9042	0.0000E+00
Beam	4	3.2328	7.3987	1.0633E+00	22028	4.9674	0.0000E+00

The numerical results of this preliminary experiments allows us to conclude that proposed Firefly dynamic penalty approach using $\mu(t) = (2t)^4$ is effective in reaching the best solution known in the literature. It may be a promising and efficient strategy for solving engineering design problems.

CONCLUSION

This paper presents a dynamic penalty approach combined with the Firefly algorithm to solve constrained global optimization problems. To assess the performance of the herein proposed method a set of eight well–known engineering design problems is solved. Comparing with the two versions based on a dynamic quadratic penalty function approach, the Firefly dynamic penalty approach based on (2) with $\mu(t) = (2t)^4$ is by far the most efficient and effective. Other interesting and complex engineering problems, such as larger truss sizing and configuration optimization problems will be addressed in the future.

ACKNOWLEDGMENTS

The authors would like to thank the financial support from FCT (Fundação para a Ciência e Tecnologia, Portugal) in the scope of the projects: PEst-OE/MAT/UI0013/2014 and PEst-OE/EEI/UI0319/2014.

REFERENCES

1. X.-S. Yang, SAGA 2009, O. Watanabe, T. Zeugmann (Eds.), *L*ect. Notes Comput. Sc., **5792**, 2009, pp. 169–178.
2. X.-S. Yang, *Int. J. Bio-Inspired Computation*, **2(2)**, 78–84, (2010).
3. B. Basu and G. K. Mahanti, *Prog. Electromagn. Res.*, **32**, 169–190 (2011).
4. M.H. Horng, *Expert Syst. Appl.*, **39**, 1078–1091 (2012).
5. A.C.C. Lemonge and H.J.C. Barbosa, *Int. J. Numer. Meth. Eng.*, **59(5)**, 703–736 (2004).
6. B. Tessema and G.G Yen, *Proc. IEEE Congress on Evolutionary Computation*, Vancouver, Canada, 2006, pp. 246–253.
7. R. Courant, *Bull. Amer. Math. Soc.* **49**, 1–23 (1943).
8. C.W. Carroll, *Oper. Res.*, **184**, 9–16 (1961).
9. A. V. Fiacco and G. P. McCormick, *Manage. Sci.*, **12(11)**, 816–828 (1966).
10. J. Joines and C. Houck, *Proc. IEEE Congress on Evolutionary Computation*, Orlando, FL, 1994, pp. 579–584.
11. Y.G. Petalas, K.E. Parsopoulos and M.N. Vrahatis, *Ann. Oper. Res.*, **156**, 99–127, (2007).
12. A.M.A.C. Rocha, and E.M.G.P. Fernandes, *I*nt. J. Comput. Math., **86(10–11)**, 1932–1946 (2009).