Indonesian Journal of Electrical Engineering and Computer Science
Vol. 13, No. 1, January 2019, pp. 162~169
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v13.i1.pp162-169

Leaders and followers algorithm for constrained non-linear optimization

Helen Yuliana Angmalisang, Syaiful Anam, Sobri Abusini
Brawijaya University, Jl. Veteran, Malang 65145, East Java, Indonesia

Article Info

Article history:

Received Jul 11, 2018
Revised Nov 18, 2018
Accepted Nov 30, 2018

Keywords:

Constraint
Leaders and followers
Metaheuristics
Non-linear optimization

ABSTRACT

Leaders and Followers algorithm was a novel metaheuristics proposed by Yasser Gonzalez-Fernandez and Stephen Chen. In solving unconstrained optimization, it performed better exploration than other well-known metaheuristics, e.g. Genetic Algorithm, Particle Swarm Optimization and Differential Evolution. Therefore, it performed well in multi-modal problems. In this paper, Leaders and Followers was modified for constrained non-linear optimization. Several well-known benchmark problems for constrained optimization were used to evaluate the proposed algorithm. The result of the evaluation showed that the proposed algorithm consistently and successfully found the optimal solution of low dimensional constrained optimization problems and high dimensional optimization with high number of linear inequality constraint only. Moreover, the proposed algorithm had difficulty in solving high dimensional optimization problem with non-linear constraints and any problem which has more than one equality constraint. In the comparison with other metaheuristics, Leaders and Followers had better performance in overall benchmark problems.

Corresponding Author:

Syaiful Anam,
Brawijaya University,
Jl. Veteran, Malang 65145, East Java, Indonesia.
Email: syaiful@ub.ac.id

1. INTRODUCTION

Nowadays, optimization plays an important role in various fields of real-world, e.g. engineering, finance, transportation and operational research [1]. There are many kinds of optimization problems. One of them is constrained non-linear optimization. An optimization problem is classified as constrained optimization if the objective function is minimized or maximized under given constraints [2]. Constrained non-linear optimization is defined as a constrained optimization problem where its objective function or at least one of the constraints is non-linear function [3]. In real life, constrained optimization problem may be found very often because many required resources are not always unlimited.

Metaheuristics have been widely implemented for solving many kinds of optimization problems, including constrained non-linear optimization. In solving optimization problem, metaheuristics search solution randomly and by trial and error. They are not like deterministic methods which require initial guess [4] and mathematical requirements, e.g. gradient or continuous functions [5]. They only require objective function and the searching domain in solving problems [6]. Moreover, they relatively need cheaper computation cost than the deterministic ones.

Since 1960s, metaheuristics have been rapidly developed [6]. Some of the famous metaheuristics are Genetic Algorithm (GA), Particle Swarm Optimization (PSO) and Differential Evolution (DE). They have been widely implemented in various optimization problems. However, they have a same disadvantage, i.e. easy to fall into local optima [7-9] or tend to prematurely converge [9, 10]. As the consequence, they often fail to approach the optimal solution. Therefore, it is necessary to find a metaheuristics that can perform better in solving optimization problems.

In [9], it is stated that the main cause of premature convergence in these well-known algorithms is the direct comparison of newly discovered solutions with the current best-known solution. Therefore, Gonzales-Fernandez and Chen [9] proposed a novel metaheuristics named Leaders and Followers (LaF) which avoids this kind of comparison. In [9], LaF is better in solving unconstrained non-linear optimization than PSO and DE. It is able to explore better so that it can perform better in muti-modal optimization problems. Moreover, LaF is simple and does not need any parameter, so it may save computation time because there is no need to estimate any parameter. However, in [9] there is no any discussion about boundary constraint handling, even though there is a possibility that some new solutions created by the operator in LaF is outside the searching space.

There are some methods to deal with the boundary constraint violations. Some of them are re-initialization and clamping (bring back the solution to the peak value). In [11], it is proven that clamping method is more effective than re-initialization method. It can improve the solution much better than the re-initialization method. Therefore, it can be used in LaF to handle the boundary constraint violation.

For solving constrained optimization problem, a metaheuristic should be modified using a constraint-handling technique. There are various constraint-handling techniques. The most widely used technique is penalty function [12]. It modifies the objective function by adding a penalty function. This technique has been being used with various metaheuristics, both the old and the new ones. In [13], Harmony Search (HS) algorithm was modified using death penalty, static penalty and a new penalty function technique, named two stage penalty function. In [14], static penalty and feasibility rules method were used with Firefly Algorithm (FA) for constrained optimization. In [15], static penalty technique was also combined with a novel metaheuristic, named Bacterial-inspired Algorithm, for constrained optimization. Static penalty and dynamic penalty function were also used with an emerging metaheuristic, named Cohort Intelligence (CI) for constrained optimization [16]. In [17], Differential Search (DS) algorithm is developed for constrained optimization with static and dynamic penalty function.

In this study, LaF is implemented for solving constrained optimization problem using static penalty function for handling the constraints and clamping method [11] for handling the boundary constraint violation. After being modified, the proposed algorithm was evaluated using several well-known benchmark problems. Then, the evaluation results of the proposed algorithm are compared with other metaheuristics [13, 14], [16-18]. Section 2 introduces the proposed algorithm. Section 3 is the research method. Section 4 presents and discusses the results. Then, the conclusions are given in Section 5.

2. THE PROPOSED ALGORITHM

Leaders and Followers (LaF) algorithm uses two different populations, i.e. *Leaders* and *Followers*. *Followers* is assigned to explore some new sub-regions of the searching space that have local optima (which called attraction basin), whereas *Leaders* is assigned to store promising solutions which may be a global optimum. In this algorithm, there is no comparison of new discovered solutions and a best current solution. This kind of comparison is avoided to prevent premature convergence. Algorithm 1 is the pseudocode of Leaders and Followers algorithm.

There is possibility that *Trial* is formed outside the searching space. To handle the boundary constraint violation, this study uses clamping method by bringing the solution to the boundary value [11]. The algorithm is modified by adding the conditionals on line 15-16. If the position of *Trial* is not in the searching space, the position is moved to the boundary. To handle the constraints, this algorithm uses penalty technique. This technique is the most widely constraint-handling technique. It transforms constrained problem into unconstrained problem. The objective function is modified by adding penalty function. The general form of penalty function is as follows.

$$F(\vec{x}) = f(\vec{x}) + \sum M \times \max[0, g(\vec{x})] \times a + \sum M \times (|h(\vec{x})| - \varepsilon) \times b$$

$F(\vec{x})$ is modified objective function, $f(\vec{x})$ is original objective function of constrained optimization problem, M is penalty factor which should be large enough for minimization problems, $g(\vec{x})$ is original inequality constraints, $h(\vec{x})$ is original equality constraints, a and b are both constants and ε is error tolerance.

3. RESEARCH METHOD

The proposed algorithm was evaluated using well-known benchmark problems for constrained optimization (f_1-f_{13}). Table 1 is the summary of the benchmark problem where ρ is the ratio of the feasible search space size and the entire search space, LI is the number of linear inequality constraint, NI is the number of non-linear inequality constraint, NE is the number of nonlinear equality constraint and a is the number of active constraint. Table 2 presents the details of problem. Each optimization problem was evaluated in twenty-five independent runs with various population size, $n = 10, 25, 50, 100$. The algorithm was stopped if there was no better solution found in 5000 iterations in a row or the algorithm had been run in 600 seconds. The proposed algorithm uses static penalty factor and parameters, i.e. $M = 50{,}000, a = 1, b = 1$ and $\varepsilon = 0.0001$. If the proposed algorithm meets difficulty to reach the optimal solution of a test function, the algorithm will be evaluated with a bigger population size and longer computation time limit.

Algorithm 1. Pseudocode of Leaders and Followers Algorithm

```
s = number of decision variables
n = population size
lb(j) = lower bound of j-th decision variables
ub(j) = upper bound of j-th decision variables
L = initialize Leaders with n uniform random vectors
F = initialize Followers with n uniform random vectors
repeat
    for i = 1:n do
        indl = round(rand*n)
        indf = round(rand*n)
        leader = L(indl,:)
        follower = F(indf,:)
        for j = 1:s do
            trial(j) = follower(j) + rand*2*(leader(j) – follower(j))
            if trial(j) < lb(j) then trial(j) = lb(j)
                if trial(j) > ub(j) then trial(j) = ub(j)
        end for
        if f(trial) < f(follower) then F(indf,:) = trial(:)
    end for
    if median(f(F)) < median(f(L)) then
        Lnew(1) = an element of L or F which has the best fitness
        for i = 2:n do
            leader = pick an element of L randomly
            follower = pick an element of F randomly
            if f(leader) < f(follower) then Lnew(i) = leader
                else Lnew(i) = follower
        end
        F = reinitialize Followers uniformly
    end if
until the termination criterion is satisfied
```

Table 1. Summary of the Benchmark Problems

Test Function	Optimal Solution	Dimension	Type of f	ρ (%)	LI	NI	NE	a
f_1	-15	13	Quadratic	0.0003	9	0	0	6
f_2	-0.8036191	20	Nonlinear	99.9962	0	2	0	1
f_3	-1.0005001	10	Polynomial	0.0002	0	0	1	1
f_4	-30665.539	5	Quadratic	26.9089	0	6	0	2
f_5	5126.496714	4	Cubic	0.0000	2	0	3	3
f_6	-6961.813876	2	Cubic	0.0065	0	2	0	2
f_7	24.30620907	10	Quadratic	0.0001	3	5	0	6
f_8	-0.09582504	2	Nonlinear	0.8484	0	2	0	0
f_9	680.6300574	7	Polynomial	0.5319	0	4	0	2
f_{10}	7049.2480205	8	Linear	0.0005	3	3	0	6
f_{11}	0.7499	2	Quadratic	0.0099	0	0	1	1
f_{12}	-1	3	Quadratic	4.7452	0	1	0	0
f_{13}	0.053941514	5	Exponential	0.0000	0	0	3	3

Table 2. Details of the Benchmark Problems

	Objective Function	Constraints	Bounds
$\boldsymbol{f_1}$	$f(\vec{x}) = 5\sum_{i=1}^{4} x_i - 5\sum_{i=1}^{4} {x_i}^2 - \sum_{i=5}^{13} x_i$	$g_1(\vec{x}) = 2x_1 + 2x_2 + x_{10} + x_{11} - 10 \le 0$ $g_2(\vec{x}) = 2x_1 + 2x_3 + x_{10} + x_{12} - 10 \le 0$ $g_3(\vec{x}) = 2x_2 + 2x_2 + x_{11} + x_{12} - 10 \le 0$ $g_4(\vec{x}) = -8x_1 + x_{10} \le 0$ $g_5(\vec{x}) = -8x_2 + x_{11} \le 0$ $g_6(\vec{x}) = -8x_3 + x_{12} \le 0$ $g_7(\vec{x}) = -2x_4 - x_5 + x_{10} \le 0$ $g_8(\vec{x}) = -2x_6 - x_7 + x_{11} \le 0$ $g_9(\vec{x}) = -2x_8 - x_9 + x_{12} \le 0$	L = (0, 0, …, 0) U = (1, 1, 1, 1, 1, 1, 1, 1, 1, 100, 100, 100, 1)
$\boldsymbol{f_2}$	$f(\vec{x}) = -\left\lvert \frac{\sum_{i=1}^{s} \cos(x_i)^4 - 2\prod_{i=1}^{s} \cos(x_i)^2}{\sqrt{\sum_{i=1}^{s} i{x_i}^2}} \right\rvert$ $s = 20$	$g_1(\vec{x}) = 0.75 - \prod_{i=1}^{s} x_i \le 0$ $g_2(\vec{x}) = \sum_{i=1}^{s} x_i - 7.5s \le 0$	L = 0; U = 10;
$\boldsymbol{f_3}$	$f(\vec{x}) = -(\sqrt{s})^s \prod_{i=1}^{s} x_i$ $s = 10$	$h_1(\vec{x}) = \sum_{i=1}^{s} {x_i}^2 - 1 = 0$	L = 0; U = 1;
$\boldsymbol{f_4}$	$f(\vec{x}) = 5.3578547x_3^2 + 0.8356891x_1x_5 + 37.293239x_1 - 40792.141$	$g_1(\vec{x}) = 85.334407 + 0.0056858x_2x_5 + 0.0006262x_1x_4 - 0.0022053x_3x_5 - 92 \le 0$ $g_2(\vec{x}) = -85.334407 - 0.0056858x_2x_5 - 0.0006262x_1x_4 + 0.0022053x_3x_5 \le 0$ $g_3(\vec{x}) = 80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 + 0.0021813x_3^2 - 110 \le 0$ $g_4(\vec{x}) = -80.51249 - 0.0071317x_2x_5 - 0.0029955x_1x_2 - 0.0021813x_3^2 + 90 \le 0$ $g_5(\vec{x}) = 9.300961 + 0.0047026x_3x_5 + 0.0012547x_1x_3 + 0.0019085x_3x_4 - 25 \le 0$ $g_6(\vec{x}) = -9.300961 - 0.0047026x_3x_5 - 0.0012547x_1x_3 - 0.0019085x_3x_4 + 20 \le 0$	L = (78, 33, 27, 27, 27) U = (102, 45, 45, 45, 45)
$\boldsymbol{f_5}$	$f(\vec{x}) = 3x_1 + 0.000001x_1^3 + 2x_2 + \left(\frac{0.000002}{3}\right)x_2^3$	$g_1(\vec{x}) = -x_4 + x_3 - 0.55 \le 0$ $g_2(\vec{x}) = -x_3 + x_4 - 0.55 \le 0$ $h_3(\vec{x}) = 1000\sin(-x_3 - 0.25) + 1000\sin(-x_4 - 0.25) + 894.8 - x_1 = 0$ $h_4(\vec{x}) = 1000\sin(x_3 - 0.25) + 1000\sin(x_3 - x_4 - 0.25) + 894.8 - x_2 = 0$ $h_5(\vec{x}) = 1000\sin(x_4 - 0.25) + 1000\sin(x_4 - x_3 - 0.25) + 1294.8 = 0$	L = (0, 0, -0.55, -0.55) U = (1200, 1200, 0.55, 0.55)
$\boldsymbol{f_6}$	$f(\vec{x}) = (x_1 - 10)^3 + (x_2 - 20)^3$	$g_1(\vec{x}) = -(x_1 - 5)^2 - (x_2 - 5)^2 + 100 \le 0$ $g_2(\vec{x}) = (x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \le 0$	L = (13, 0) U = (100, 100)
$\boldsymbol{f_7}$	$f(\vec{x}) = x_1^2 + x_2^3 + x_1x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 + 4(x_4 - 5)^2 + (x_5 - 3)^2 + 2(x_6 - 1)^2 + 5x_7^2 + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45$	$g_1(\vec{x}) = -105 + 4x_1 + 5x_2 - 3x_7 + 9x_8 \le 0$ $g_2(\vec{x}) = 10x_1 - 8x_2 - 17x_7 + 2x_8 \le 0$ $g_3(\vec{x}) = -8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12 \le 0$ $g_4(\vec{x}) = 3(x_1 - 2)^2 + 4(x_2 - 3)^2 + 2x_3^2 - 7x_4 - 120 \le 0$ $g_5(\vec{x}) = 5x_1^2 + 8x_2 + (x_3 - 6)^2 - 2x_4 - 40 \le 0$ $g_6(\vec{x}) = x_1^2 + 2(x_2 - 2)^2 - 2x_1x_2 + 14x_5 - 6x_6 \le 0$ $g_7(\vec{x})\ 0.5(x_1 - 8)^2 + 2(x_2 - 4)^2 + 3x_5^2 - x_6 - 30 \le 0$ $g_8(\vec{x}) = -3x_1 + 6x_2 + 12(x_9 - 8)^2 - 7x_{10} \le 0$	L = -10 U = 10
$\boldsymbol{f_8}$	$f(\vec{x}) = -\frac{(\sin(2\pi x_1))^3 \sin(2\pi x_2)}{x_1^3(x_1 + x_2)}$	$g_1(\vec{x}) = -x_1^2 - x_2 + 1 \le 0$ $g_2(\vec{x}) = 1 - x_1 + (x_2 - 4)^2 \le 0$	L = 0 U = 10
$\boldsymbol{f_9}$	$f(\vec{x}) = (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 + 3(x_4 - 11)^2 + 10_5^6 + 7x_6^2 + x_7^4 - 4x_6x_7 - 10x_6 - 8x_7$	$g_1(\vec{x}) = -127 + 2x_1^2 + 3x_2^4 + x_3 + 4x_4^2 + 5x_5 \le 0$ $g_2(\vec{x}) = -282 + 7x_1 + 3x_2 + 10x_3^2 + x_4 - x_5 \le 0$ $g_3(\vec{x}) = -196 + 23x_1 + x_2^2 + 6x_6^2 - 8x_7 \le 0$ $g_4(\vec{x}) = 4x_1^2 + x_2^2 - 3x_1x_2 + 2x_3^2 + 5x_6 - 11x_7 \le 0$	L = -10 U = 10
$\boldsymbol{f_1}$	$f(\vec{x}) = x_1 + x_2 + x_3$	$g_1(\vec{x}) = -1 + 0.0025(x_4 + x_6) \le 0$ $g_2(\vec{x}) = -1 + 0.0025(x_5 + x_7 - x_4) \le 0$ $g_3(\vec{x}) = -1 + 0.01(x_8 - x_5) \le 0$ $g_4(\vec{x}) = -x_1x_6 + 833.33252x_4 + 100x_1 - 83333.333 \le 0$ $g_5(\vec{x}) = -x_2x_7 + 1250x_5 + x_2x_4 - 1250x_4 \le 0$	L = (100, 1000, 1000, 10, 10, 10, 10, 10) U = (10000, 10000,

	Objective Function	Constraints	Bounds
		$g_6(\vec{x}) = -x_3x_8 + 1250000 + x_3x_5 - 2500x_5 \leq 0$	10000, 1000, 1000, 1000, 1000, 1000)
$\boldsymbol{f_1}$	$f(\vec{x}) = x_1^2 + (x_2 - 1)^2$	$h(\vec{x}) = x_2 - x_1^2 = 0$	L = -1 U = 1
$\boldsymbol{f_1}$	$f(\vec{x}) = -(100 - (x_1 - 5)^2 - (x_2 - 5)^2 - (x_3 - 5)^2)/100$	$g(\vec{x}) = (x_1 - p)^2 + (x_2 - q)^2 + (x_3 - r)^2 - 0.0625 \leq 0$	L = 0 U = 10
$\boldsymbol{f_1}$	$f(\vec{x}) = e^{x_1x_2x_3x_4x_5}$	$h_1(\vec{x}) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 - 10 = 0$ $h_2(\vec{x}) = x_2x_3 - 5x_4x_5 = 0$ $h_3(\vec{x}) = x_1^3 + x_2^3 + 1 = 0$	L = -2.3 U = 2.3

4. RESULTS AND DISCUSSION

Table 3 and 4 presents the evaluation results with population size (n) = 10, 25, 50 and 100. The algorithm obtains the optimal solution for $f_1, f_2, f_4, f_6, f_8, f_9, f_{11}$ and f_{12}. The standard deviation of all obtained solutions for f_4, f_6, f_8 and g_{12} approaches zero. This means that in all runs, the algorithm consistently obtains optimal solutions for these problems. Table 1 shows that all of these problems (f_4, f_6, f_8 and f_{12}) are low dimensional ($s \leq 5$) and have no equality constraints.

Table 3. The Results Obtained by the Proposed Algorithm for $f_1 - f_7$

Problem			f_1	f_2	f_3	f_4	f_5	f_6	f_7
Optimal Solution			-15	-0.8036	-1.0005	-30666	5126.5	-6961.8	24.3062
Obtained Solution	n = 10	Best	**-15**	**-0.8036**	-1.0002	**-30666**	5126.8	**-6961.8**	24.3558
		Mean	-12.9843	-0.6849	-1.0002	**-30666**	5264.9	**-6961.8**	25.4199
		Worst	-10.1094	-0.5702	-1.0002	**-30666**	5693.3	**-6961.8**	27.4359
		Std	1.4923	0.0685	2E-06	4E-06	209.63	4.5E-11	0.7485
	n = 25	Best	**-15**	-0.7881	-1.0002	**-30666**	5134.0	**-6961.8**	24.3188
		Mean	-14.0844	-0.7319	-0.9835	**-30666**	5239.4	**-6961.8**	24.7202
		Worst	-11.8281	-0.6057	-0.8326	**-30666**	5790.5	**-6961.8**	26.0405
		Std	0.9483	0.0452	0.0530	7E-10	198.89	2.3E-12	0.3933
	n = 50	Best	**-15**	**-0.8036**	-1.0003	**-30666**	5128.6	**-6961.8**	24.3156
		Mean	-14.4737	-0.7645	-0.9538	**-30666**	5253.1	**-6961.8**	24.6504
		Worst	-11.2812	-0.5744	-0.7191	**-30666**	5673.6	**-6961.8**	25.077
		Std	1.0533	0.0497	0.1006	1E-11	164.81	0	0.2545
	n = 100	Best	**-15**	**-0.8036**	-1.0002	**-30666**	5126.7	**-6961.8**	24.3084
		Mean	**-15**	-0.7897	-0.9521	**-30666**	5327.2	**-6961.8**	24.4549
		Worst	**-15**	-0.7692	-0.8074	**-30666**	5714.9	**-6961.8**	24.8242
		Std	7.25E-16	0.009	0.0611	6E-12	224.09	0	0.1303

Std = Standard Deviation

Table 4. The Results Obtained by the Proposed Algorithm for $f_8 - f_{13}$

Problem			f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}
Optimal Solution			-0.0958	680.6301	7049.2	0.7499	-1	0.0539
Obtained Solution	n = 10	Best	**-0.0958**	680.6333	7095.9	**0.7499**	**-1**	0.1789
		Mean	**-0.0958**	680.6518	7897.8	**0.7499**	**-1**	1.0785
		Worst	**-0.0958**	680.6736	10911	**0.7499**	**-1**	4.9511
		Std	4.01E-18	1.03E-02	868.6	9.3E-8	0	1.5194
	n = 25	Best	**-0.0958**	680.6324	7049.5	**0.7499**	**-1**	0.0865
		Mean	**-0.0958**	680.6360	7468.2	**0.7499**	**-1**	1.0323
		Worst	**-0.0958**	680.6423	8076.3	**0.7499**	**-1**	5.0551
		Std	4.91E-18	2.80E-03	298.6	6.5E-8	0	1.4771
	n = 50	Best	**-0.0958**	680.6305	7114.1	**0.7499**	**-1**	0.0730
		Mean	**-0.0958**	680.6319	7298.3	0.7502	**-1**	1.6426
		Worst	**-0.0958**	680.6339	7547.2	0.7529	**-1**	11.0996
		Std	8.96E-18	1.00E-03	118.2	9.6E-4	0	2.3141
	n = 100	Best	**-0.0958**	**680.6301**	7081.5	**0.7499**	**-1**	0.6148
		Mean	**-0.0958**	680.6307	7239.8	0.7500	**-1**	0.9625
		Worst	**-0.0958**	680.6321	7469	0.7505	**-1**	3.1843
		Std	8.96E-18	4.59E-04	111.7	2.0E-4	0	0.7877

For f_1 which is a high dimensional problem ($s = 13$), the obtained errors is quite high when the population size (n) = 10, 25 and 50, but when the population size (n) = 100, the errors approaches zero. The algorithm successfully obtains the optimal result in each run when the population size (n) = 100, although the number of dimension and inequality constraint in f_1 is higher than f_7, f_9 and f_{10}. Table 1 shows that the difference of f_1 and f_7, f_9, f_{10} except the dimensionality is the type of inequality constraints in the problem. f_1 has only linear constraints, unlike f_7, f_9 and f_{10} which have nonlinear constraints. The algorithm tends to have difficulty in solving the optimization problems with equality constraint(s) (f_3, f_5 and f_{13}), except f_{11}. In f_{11}, the proposed algorithm consistently approaches the optimal solution when n = 10 and 25. Table 1 shows that the difference of f_{11} and the others is it is low dimensional and has only one equality constraint. Moreover, the algorithm tends to find difficulty in solving the high dimensional optimization problems with nonlinear inequality constraints only (f_2, f_7, f_9 and f_{10}).

When the proposed algorithm is evaluated on f_2, f_7, f_9 and f_{10} with a big population size, e.g. n = 2000, and the same termination criterion, the obtained solutions are much better and the standard deviations are much smaller even though the computational time limit is same, e.g. 600 seconds (Table 5). When the time limit is longer, i.e. 1200 seconds, Table 5 shows that LaF does not obtain better solutions, except on f_2. Thus, in solving the optimization problems with high dimensional optimization problems with nonlinear inequality constraints only, LaF requires a big population size ($n \geq 2000$).

Table 5. The Results Obtained by the Proposed Algorithm for High Dimensional Optimization Problems with inequality constraints only (f_2, f_7, f_9 and f_{10}) when n = 2,000

Time Limit	600s				1200s			
Problem	f_2	f_7	f_9	f_{10}	f_2	f_7	f_9	f_{10}
Best	-0.8035	24.3114	680.6303	7.14E+03	-0.8036	24.3085	680.6303	7.09E+03
Mean	-0.8035	24.3168	680.6305	7.19E+03	-0.8036	24.3199	680.6305	7.17E+03
Worst	-0.8034	24.3349	680.6308	7.24E+03	-0.8036	24.3597	680.6311	7.29E+03
Std	2.8E-05	0.0067	1.88E-04	33.7939	5.5E-06	0.0154	2.34E-04	69.3378

Table 6 presents the comparison of solutions obtained by the proposed algorithm and other metaheuristics, i.e. Harmony Search with two stage penalty function (HS) [13], Firefly Algorithm with combination of static penalty and feasibility rules (FA) [14], Cohort Intelligence (CI) with static penalty (SCI) and dynamic penalty (DCI) [16], Differential Search with static penalty (SDS) and dynamic penalty (DDS) [17] and Musical Composition Method (MCM) [18]. The proposed algorithm obtains the smallest values of best, mean, worst and standard deviation values in this comparison on f_1, f_3, f_4, f_6, f_9 and f_{12}. It means that LaF is better and more consistent or stable than the other metaheuristics in solving these problems. In the other problems (f_2, f_5, f_7, f_{10} and f_{13}), except f_8 and f_{11}, LaF is still not competitive compared to the other metaheuristics, since it has difficulties in solving high dimensional optimization problem with non-linear constraints and any problem which has more than one equality constraint. In f_8 and f_{11}, LaF obtains the known optimal solutions, but SDS on f_8, FA and MCM on f_{11} apparently obtain better solutions than the optimal solutions that have been known so far. However, in overall, LaF is more competitive than the other metaheuristics.

5. CONCLUSION

Based on the result and analysis, it is concluded that Leaders and Followers (LaF) algorithm can be implemented to solve constrained non-linear optimization problems. With small population size, i.e. $n \leq 10$, LaF consistently and successfully find the optimal solution of low dimensional ($s \leq 5$) optimization problems with inequality constraints only and the low dimensional ($s \leq 2$) problem with only one equality constraint. With population size, i.e. $n \geq 100$, LaF can optimally solve any high dimensional constrained non-linear optimization problem that has high number of linear inequality constraints and no non-linear constraint. LaF has difficulty in solving high dimensional optimization problem with non-linear constraints and any problem which has more than one equality constraint.

In the comparison with other metaheuristics, LaF has better performance in overall benchmark problems. It should also be noted that the constraint-handling method used in the proposed algorithm is only the classical static penalty function and the LaF algorithm used in this study is the original one. It means that there is a big possibility to use some better constraint-handling method or to modify the original LaF algorithm in order to obtain much better performance in the further studies.

Table 6. Comparison of Solutions Obtained by the Proposed Algorithm and other Metaheuristics

		LaF	HS [13]	FA [14]	SCI [16]	DCI [16]	SDS [17]	DDS [17]	MCM [18]
f_1	Best	**-15**	-14.999	NA	-14.997	**-15**	**-15**	**-15**	**-15**
	Mean	**-15**	-14.959	NA	NA	-14.9	-14.8	-12.3	NA
	Worst	**-15**	-14.893	NA	NA	-13	-6	-13	NA
	Std	**7.25E-16**	0.0229	NA	0.1982	4.5E-01	2.5567	0.0181	0.1473
f_2	Best	**-0.8036**	-0.7255	NA	**-0.8036**	**-0.8036**	**-0.8036**	-0.8035	**-0.8036**
	Mean	-0.7897	-0.7009	NA	NA	-0.7864	**-0.7920**	-0.7880	NA
	Worst	-0.7692	-0.6543	NA	NA	-0.7395	-0.7729	**-0.7743**	NA
	Std	0.009	0.0397	NA	0.0361	0.02	0.0009	**0.0007**	0.0253
f_3	Best	**-1.0002**	-1.0000	NA	-1.0013	-0.9999	NA	NA	-0.9997
	Mean	**-1.0002**	-0.988	NA	NA	-0.9839	NA	NA	NA
	Worst	**-1.0002**	-0.951	NA	NA	-0.7395	NA	NA	NA
	Std	**2.00E-06**	0.0137	NA	0.0011	5.0E-02	NA	NA	0.0008
f_4	Best	**-30666**	-30665	-30665	**-30666**	-30665	**-30666**	**-30666**	**-30666**
	Mean	**-30666**	-30582	-30665	NA	-30665	-30662	**-30666**	NA
	Worst	**-30666**	-30405	-30664	NA	-30665	-30599	**-30666**	NA
	Std	**6.00E-12**	24.2567	0.4755	0.045	4.9E-03	1.4968	0.1204	16.175
f_5	Best	5128.6	**5112.3**	NA	5119.1	4232.6	5131.3	5131.3	5121.2
	Mean	5253.1	**5115.2**	NA	NA	4896.6	5557.3	5745.1	NA
	Worst	5673.6	**5125.3**	NA	NA	5612.5	6112.2	6112.2	NA
	Std	164.81	**1.25**	NA	40.42	3.9E+02	43.56	40.64	42.19
f_6	Best	**-6961.8**	-6961.6	-6960.5	**-6961.8**	**-6961.8**	**-6961.8**	**-6961.8**	**-6961.8**
	Mean	**-6961.8**	-6961.3	-6956.6	NA	**-6961.8**	5.9E+13	1.8E+6	NA
	Worst	**-6961.8**	-6960.9	-6953.5	NA	**-6961.8**	2.4E+15	3.6E+7	NA
	Std	**0**	0.2443	2.1928	1.5E-05	**0**	1.1E+14	8.1E+6	3.8E-07
f_7	Best	24.3084	24.552	24.3805	**24.3044**	24.3281	24.3302	24.315	24.3506
	Mean	24.4549	27.612	24.4705	NA	24.4677	**24.341**	24.7153	NA
	Worst	24.8242	31.231	**24.6024**	NA	24.987	25.5169	25.5336	NA
	Std	0.1303	1.6545	0.0597	0.2216	0.18	0.0382	**0.0306**	0.2135
f_8	Best	-0.0958	-0.0958	-0.0958	-0.0958	-0.0958	**-0.0959**	-0.0958	-0.0958
	Mean	-0.0958	-0.0807	-0.0958	NA	-0.0958	**-0.0959**	-0.0958	NA
	Worst	-0.0958	-0.0761	-0.0958	NA	-0.0958	**-0.0959**	-0.0958	NA
	Std	4.01E-18	0.0136	2.88E-06	-1.1E-12	2.1E−12	**0**	**0**	6.2E-08
f_9	Best	680.6301	680.656	680.8463	680.6726	684.1806	**680.63**	**680.63**	680.6738
	Mean	**680.6307**	680.742	681.0415	NA	684.1996	680.7093	680.7132	NA
	Worst	**680.6321**	680.779	681.2603	NA	684.2519	680.9682	681.1324	NA
	Std	**4.59E-04**	0.0725	0.15336	0.2598	1.66E-02	0.0082	0.0011	0.2882
f_{10}	Best	7081.5	7082.6	NA	**7051.8**	8648.2	7058.19	7056.76	7051.9
	Mean	7239.8	**7110.2**	NA	NA	9286.5	7297.595	7350.35	NA
	Worst	7469	**7110.3**	NA	NA	11745.2	7621.005	7846.79	NA
	Std	111.7	**2.0854**	NA	11.5586	8.6E+02	16.581	20.042	15.3881
f_{11}	Best	0.7499	0.749	0.7490	0.7497	0.7501	-0.7499	-0.7499	**0.7489**
	Mean	0.7499	0.749	**0.7490**	NA	0.7798	-0.8457	-0.7731	NA
	Worst	0.7499	0.749	**0.7490**	NA	0.8801	-1	-1	NA
	Std	**6.50E-08**	3.0E-06	3.42E-06	0.0013	3.5E-02	0.0116	0.006	0.0011
f_{12}	Best	**-1**	-0.9909	-0.99995	**-1**	**-1**	**-1**	**-1**	**-1**
	Mean	**-1**	-0.9525	-0.99995	NA	**-1**	**-1**	**-1**	NA
	Worst	**-1**	-0.8913	-0.99995	NA	**-1**	**-1**	**-1**	NA
	Std	**0**	0.0888	7.78E-07	1.6E-12	1.4E−09	**0**	**0**	0.0025
f_{13}	Best	0.1789	**0.0571**	NA	NA	NA	NA	NA	NA
	Mean	1.0785	**0.0595**	NA	NA	NA	NA	NA	NA
	Worst	4.9511	**0.0703**	NA	NA	NA	NA	NA	NA
	Std	1.5194	**0.0726**	NA	NA	NA	NA	NA	NA

NA = Not Available
Std = Standard Deviation

REFERENCES

[1] Bashath S, Ismail AR. Comparison of Swarm Intelligence Algorithms for High Dimensional Optimization Problems. *Indonesian Journal of Electrical Engineering and Computer Science*. 2018; 11(1): 300-307.

[2] Abd-El-Waheda WF, Mousa AA, El-Shorbagy MA. Integrating Particle Swarm Optimization with Genetic Algorithms for Solving Nonlinear Optimization Problems. *Journal of Computational and Applied Mathematics*. 2011; 235: 1446-1453.

[3] Griva I, Nash SG, Ariela S. Linear and Nonlinear Optimization. Second Edition. Philadelphia: Society for Industrial Applied Mathematics. 2009: 14.

[4] Nayak V, Suthar H, Gadit J. Implementation of Artificial Bee Colony Algorithm. *IAES International Journal of Artificial Intelligence*. 2012; 1(3): 112-120.
[5] Garg H. A Hybrid PSO-GA Algorithm for Constrained Optimization. *Applied Mathematics and Computation*. 2016; 274: 292-305.
[6] Rajpurohit J, Sharma TK, Abraham A, Vaishali. Glossary of Metaheuristic Algorithm. *International Journal of Computer Information Systems and Industrial Management Applications.* 2017; 9: 181-205.
[7] Weidong J, Jun Z. PSO Algorithm Based on Accumulation Effect and Mutation. *TELKOMNIKA Indonesian Journal of Electrical Engineering*. 2013; 11(12): 7344-7350.
[8] Anam K, Al-Jumaily A. Optimized Kernel Extreme Learning Machine for Myoelectric Pattern Recognition. *International Journal of Electrical and Computer Engineering*. 2018; 8(1): 483-496.
[9] Gonzalez-Fernandez YG, Chen S. *Leaders and Followers – A New Metaheuristic to Avoid the Bias of Accumulated Information.* IEEE Congress on Evolutionary Computation. Sendai. 2015: 776-783.
[10] Liu J, Cai Z, Liu J. *Premature Convergence in Genetic Algorithm: Analysis and Prevention Based on Chaos Operator.* Proceedings of 3rd World Congress on Intelligent Control and Automation. Hefei. 2000: 495–499.
[11] Kaur A, Kaur M. Dealing with Boundary Constraint Violations in Particle Swarm Optimization with Aging Leader and Challengers (ALC-PSO). *International Journal of Computer Applications.* 2015; 121(11): 13-19.
[12] Mezura-Montes E, Coello Coello CA. Constraint-handling in Nature-Inspired Numerical Optimization: Past, Present and Future. *Swarm and Evolutionary Computation.* 2011; 1(4): 173-194.
[13] Zhang B, Duan J, Sang H, Li J, Yan H. *A New Penalty Function Method for Constrained Optimization Using Harmony Search Algorithm*. IEEE Congress on Evolutionary Computation. Beijing. 2014: 853-859.
[14] Deshpande AM, Phatnani GM. *Constraint Handling in Firefly Algorithm*. IEEE International Conference on Cybernetics (CYBCO). Lausanne. 2013; 186-190.
[15] Niu B, Wang J, Wang H. Bacterial-inspired Algorithms for Solving Constrained Optimization Problems. *Neurocomputing*. 2015; 148: 54-62.
[16] Kulkarni O, Kulkarni N, Kulkarni AJ, Kakandikar G. Constrained Cohort Intelligence using Static and Dynamic Penalty Function Approach for Mechanical Components Design. *International Journal of Parallel, Emergent and Distributed Systems.* 2016. DOI: https://doi.org/10.1080/17445760.2016.1242728
[17] Liu J, Teo KL, Wang X, Wu C. An Exact Penalty Function-based Differential Search Algorithm for Constrained Global Optimization. *Soft Computing*. 2016; 20(4): 1305-1313.
[18] Mora-Gutiérrez RA, Ramírez-Rodríguez J, Rincón-García EA, Ponsich A, Herrera O, Lara-Velázquez P. Adaptation of the Musical Composition Method for Solving Constrained Optimization Problems. Soft Computing. 2014; 18(10): 1931-1948.