International Journal of Engineering & Technology, 7 (4.27) (2018) 30-37

SPC

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

A Study on the Effect of Local Neighbourhood Parameter towards the Performance of SAFIRO

Tasiransurini Ab Rahman[1], Zuwairie Ibrahim[2*], Nor Azlina Ab. Aziz[3], Nor Hidayati Abdul Aziz[4], Suad Khairi Mohammed[5], Badaruddin Muhammad[6] and Zulkifli Md Yusof[7]

[1]*Faculty of Manufacturing Engineering, Universiti Malaysia Pahang, 26600 Pekan, Pahang, Malaysia.*
[2]*Faculty of Electrical and Electronic, Universiti Tun Hussein Onn Malaysia, 86400 Johor, Malaysia.*
[3]*Faculty of Engineering and Technology, Multimedia University, 75450 Bukit Beruang, Melaka, Malaysia.*
[4]*Department of Electrical Engineering, University of Technology, Baghdad, Iraq.*
[5]*Faculty of Electrical and Electronics Engineering, Universiti Malaysia Pahang, 26600 Pekan, Pahang, Malaysia.*
**Corresponding author E-mail: zuwairie@ump.edu.my*

Abstract

Single-agent Finite Impulse Response Optimizer (SAFIRO) is a recently proposed metaheuristic optimization algorithm which adopted the procedure of the ultimate unbiased finite impulse response filter (UFIR) in state estimation. In SAFIRO, a random mutation with shrinking local neighborhood method is employed during measurement phase to balance the exploration and the exploitation process. Beta, β, is one of the parameters used in the local neighborhood to control the step size. In this study, the effect of β towards the performance of SAFIRO is observed by assigning the value of 1, 5, 10, 15, and 20. The best setting of β for SAFIRO is also determined. The CEC2014 Benchmark Test Suite is used to evaluate the SAFIRO performance with different β values. Results show that the performance of β is depending on the problems to be optimized. 17 out of 30 functions show the best performance of SAFIRO by setting $\beta = 10$. Statistical analysis using Friedman test and Holm post hoc test were performed to rank the performance. $\beta = 10$ has the highest rank where its performance is significantly better than other values, but equivalent to $\beta = 5$ and $\beta = 15$. Hence, it is recommended to tune the β for best performance, however, $\beta = 10$ is a good value to be used in SAFIRO for solving optimization problems.

Keywords: *Finite Impulse Response; Local Search Neighborhood; Metaheuristic; Optimization.*

1. Introduction

Single-agent metaheuristic algorithm is a single-solution based algorithm that used only one agent to find the optimal or the best near-optimal solution for a given optimization problem. A single-agent algorithm produces only one solution and improves the solution based on its current solution until the stopping criterion is reached. The entire optimization process of the single-agent algorithm is simpler and require a lesser number of function evaluation compared to multi-agent metaheuristic algorithm.

Among the classical single-agent metaheuristic algorithms are the Simulated Annealing (SA) [1], Tabu Search (TS) [2], Random Search (RS) [3], Pattern Search (PS) [4], Greedy Randomized Adaptive Search Procedures (GRASP) [5], Variable Neighborhood Search (VNS) [6], Guided Local Search (GLS) [7], and Iterated Local Search (ILS) [8].

Boussaid [9] stated that a basic single-agent algorithm tends to focus on exploitation, meanwhile, multi-agent algorithm biased towards exploration. Normally, a single-agent metaheuristic algorithm consists of local search-based metaheuristic [10]. Unlike global search that explores the search space, local search concentrates on finding a new solution among a neighborhood of its current solution or the best solution found so far [9]. The capability to balance of global search (diversification) and local search (intensification) is essential to ensure the efficiency of metaheuristic algorithm [11].

A local search is claimed as one of the successful approach for approximate algorithms [12]. The descent method is among the earliest and simplest method for local search [13] where the solution is chosen from the agent's neighbour. This simple method anyhow causes the agent to be easily trapped in a local optimal [14].

As single-agent metaheuristic algorithm has higher possibility to trap to the local optima [15], many researchers came out with strategies for local optimal avoidance. SA algorithm is among the first algorithms that proposed a strategy to escape from local optimal [16], which was introduced by Kirkpatrick et al. in 1983. SA was inspired by an analogy from thermodynamic systems whereby the material is heated with high temperature until its molten state is reached and then the temperature is gradually reduced until no further change is observed [1][17]. In this method, a random neighbour of the current candidate solution is compared at every iteration and replaced if it improves the current solution [18]. Otherwise, the current solution is accepted according to the probability where the probability is higher if the temperature is high and slowly lower as the temperature is getting lower [19][20]. The possibility to escape from local optima in SA is good due to the stochastic cooling factor [21]. However, there is still a possibility for SA to find the same local optimal again throughout the optimization process [14].

TS algorithm was developed by Glover in 1986 considering the use of local memory where the recent history of the search is memorized and stored in Tabu list, and prohibited to be revisited [19][22]. TS is biased to exploration if the size of Tabu list is increased, whereas it is biased to exploitation if the size of Tabu list

is reduced [14]. A local search is applied to move from the current solution to the improved solution in the neighborhood of the current solution. This process continues until the stopping criterion is met [20].

The algorithms such as the GRASP method, VNS, GLS, and ILS had been introduced with its respective local search strategies. Vortex Search (VS) [23], Mean-Variance Mapping Optimization (MVMO) [24], Simulated Raindrop (SRD) [17], and Single-solution Simulated Kalman Filter (SSKF) [25], are examples of modern single-agent metaheuristic algorithm which associated a local search approach in the respective algorithm.

Vortex Search (VS) algorithm was developed by Dogan and Olmez in 2013 [23]. It adopted the vortex flow of stirred fluids. The exploration and exploitation phases are balanced by using a vortex-like search method. For each iteration, radius decrement is done through a new adaptive step-size adjustment scheme using the inverse incomplete gamma function to improve the performance of the search process.

Single-solution Simulated Kalman Filter (SSKF) algorithm [25], developed by Aziz et.al in 2016 as another version of Simulated Kalman Filter (SKF) algorithm [26][27]. SSKF also employed a local search method in its search strategy, which mimics the prediction-measurement-estimation phases in Kalman filter to seek for the solution. An adaptive neighborhood mechanism is applied during prediction phase to predict the location of the candidate solution which is somewhere near the best-so-far solution.

Single-agent Finite Impulse Response Optimizer (SAFIRO), is a recently developed method for numerical optimization problems [28]. The search strategy of SAFIRO is inspired by the cycle of estimation procedure in ultimate unbiased finite impulse response (UFIR) filter. In state space model, besides Kalman filter, the UFIR filter is increasingly preferable to be used as an estimator for state estimation. In SAFIRO, the state estimation is considered as a solution to the optimization problem. The estimation in SAFIRO's sub-iteration is depending on the previous estimation, measurement and Kalman-like gain. SAFIRO employs a random mutation of ***X***_*best_so_far* and shrinking local neighborhood method to balance the exploration and exploitation process. The adaptive coefficient, β is one of the parameters used in shrinking local neighborhood method. This parameter is used in exponential decay equation to control the reduction of step-size, δ.

The aim of this paper is twofold. First, to observe the effect of β value towards the performance of SAFIRO. β = 1, 5, 10, 15, and 20; are selected to be used in the experiments and the comparison is based on the mean fitness value over 51 runs with 500,000 iterations of 50 problem dimension in CEC2014 Benchmark Test Suite. Second, to determine the best parameter setting of β for SAFIRO. Friedman and Holm post hoc tests show that β = 10 has highest ranking where its performance is significantly better than other values and on par with β = 5 and β = 15. Since 17 out of 30 functions show the best performance is achieved with β = 10, hence this setting is considered as the best set of β for SAFIRO.

The rest of this paper is organized as follows: Section 2 briefly introduces the SAFIRO. Section 3 presents the experimental setup. In section 4, the experimental results and discussion are presented. Finally, in Section 5, conclusions and future work are given.

2. The SAFIRO

SAFIRO is a newly developed metaheuristic algorithm for a numerical optimization problem. SAFIRO makes use of only one agent to solve an optimization problem by mimicking standard UFIR filter procedures; measurement and estimation. Only two parameters need to be initialized in SAFIRO; the horizon length, *N*, and the adaptive coefficient, β. The agent in SAFIRO makes an estimation of the optimal based on *N* recent measurements. Unlike a real UFIR filter that takes measurement values from the sensor, measurement in SAFIRO must be simulated by its own. The estimated position in the search space represents the solution of SAFIRO. The fitness of the estimated position is then evaluated based on an objective function. Figure 1 visualized the flow of SAFIRO. SAFIRO requires *N* initial measurements to begin the optimization process. Therefore, the value of *N* is defined during the initialization phase. In the original work, *N* is equal to 4. Four initial measurements, ***Y***(*t*), ***Y***(*t*-1), ***Y***(*t*-2), and ***Y***(*t*-3) are randomly generated and evaluated by using the fitness function of the problem to determine the initial ***X***_*best_so_far*. ***X***_*best_so_far* is the best-so-far solution. For a minimization problem, the value of initial ***X***_*best_so_far* is taken from the initial measurement that has the smallest fitness value. For a maximization problem, in contrast, the value of initial ***X***_*best_so_far* is taken from the initial measurement that has the largest fitness value.

After *N* random initial measurements are generated and the initial ***X***_*best_so_far* is determined during the initialization phase, SAFIRO's agent makes the next step, which is the measurement phase. For each iiteration, a new measurement is produced by using random mutation of ***X***_*best_so_far* and shrinking local neighborhood method. This approach encourages the exploration through mutation, and at the same time making a balance between the exploration and exploitation through shrinking local neighborhood. Every dimension of the problem to be optimized is associated to a random value between 0 to 1. Dimension, *d*, that has a random value ≤ 0.5 will assume the ***X***_*best_so_far* value as their measurement value as follows:

$$Y_d(t) = \boldsymbol{X}\_best\_so\_far_d \tag{1}$$

Meanwhile, dimensions with a random value > 0.5, will undergo the mutation process to produce a new candidate solution, which is conducted in a local neighborhood of ***X***_*best_so_far*. The measurement values for these respective dimensions is shown in (2),

$$Y_d(t) = \boldsymbol{X}\_best\_so\_far_d + rand(U[-\delta, \delta]) \tag{2}$$

where

$$\delta = e^{-\beta \times \frac{t}{T}} \times \frac{X_{max} - X_{min}}{2} \tag{3}$$

The local neighborhood radius is determined by the step size, δ, as in (3), where β is an adaptive coefficient value; *t* is the number of current iteration; *T* is the number of maximum iteration; $X_{\max}$ is the upper limit of search space, and $X_{\min}$ is the lower limit of search space. The exponential decay equation, $e^{-\beta \times \frac{t}{T}}$ is used to scale down the local neighborhood radius, while $(X_{\max} - X_{\min})/2$ ensure the maximum coverage of the search space. The adaptive step-size adjustment process allows the exploration behavior occurred at the beginning of steps and subsequently transform to the exploitation behavior towards the end of the steps [23]. As the iteration increases, the δ is reduced. Figure 2 shows plots of $e^{-\beta \times \frac{t}{T}}$ for different values of β. The value of β helps to control the reduction speed of neighborhood's size. A small β value causes a linear reduction, whereas a higher β value causes a faster convergence speed [25]. The convergence speed reflects the transition process from exploration to exploitation. Theoretically, β = 1 has a slower convergence speed, β = 10, 15, and 20 in contrast, have a faster convergence speed, whereas β = 5 has a moderate convergence speed.

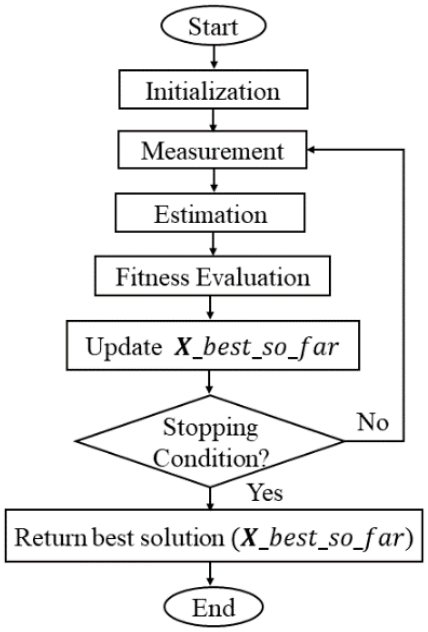

Fig. 1: SAFIRO algorithm [28].

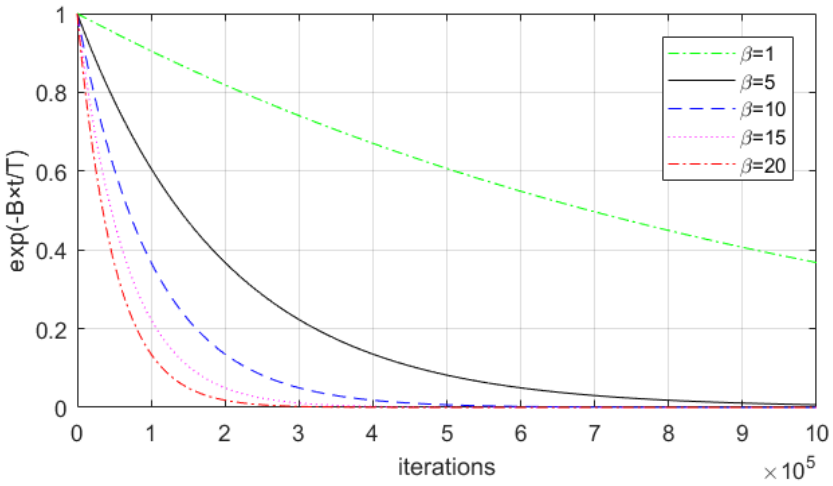

Fig. 2: The plot of δ with different β values.

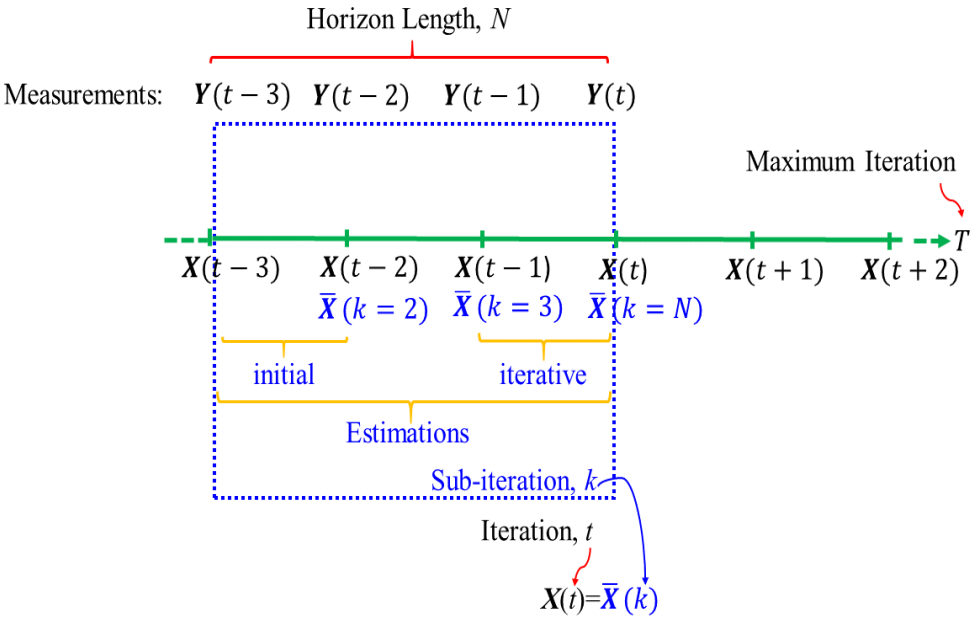

Fig. 3: Estimation stage in SAFIRO algorithm (N=4) [28].

Table 1: The CEC2014 benchmark test suite [29]

Types	No.	Functions	Ideal Fitness
Unimodal functions	1	Rotated High Conditioned Elliptic function	100
	2	Rotated Bent Cigar function	200
	3	Rotated Discus function	300
Simple multimodal functions	4	Shifted and Rotated Rosenbrock's function	400
	5	Shifted and Rotated Ackley's function	500
	6	Shifted and Rotated Weierstrass function	600
	7	Shifted and Rotated Griewank's function	700
	8	Shifted Rastrigin's function	800
	9	Shifted and Rotated Rastrigin's function	900
	10	Shifted Schwefel's function	1000
	11	Shifted and Rotated Schwefel's function	1100
	12	Shifted and Rotated Katsura function	1200
	13	Shifted and Rotated HappyCat function	1300
	14	Shifted and Rotated HGBat function	1400
	15	Shifted and Rotated Expanded Griewank's plus Rosenbrock's function	1500
	16	Shifted and Rotated Expanded Scaffer's F6 function	1600
Hybrid functions	17	Hybrid function 1 (N=3)	1700
	18	Hybrid function 2 (N=3)	1800
	19	Hybrid function 3 (N=4)	1900
	20	Hybrid function 4 (N=4)	2000
	21	Hybrid function 5 (N=5)	2100
	22	Hybrid function 6 (N=5)	2200
Composition functions	23	Composition function 1 (N=5)	2300
	24	Composition function 2 (N=3)	2400
	25	Composition function 3 (N=3)	2500
	26	Composition function 4 (N=5)	2600
	27	Composition function 5 (N=5)	2700
	28	Composition function 6 (N=5)	2800
	29	Composition function 7 (N=3)	2900
	30	Composition function 8 (N=3)	3000

After producing measurement values during the measurement phase, SAFIRO's agent conducts the next step, which is the estimation phase. During this stage, SAFIRO's solution is updated using four recent measurements correspond to its horizon length.

As depicted in Figure 3, the first two measurements ($\boldsymbol{Y}(t\text{-}3)$ and $\boldsymbol{Y}(t\text{-}2)$) are used for initial estimation whereas the other two measurements are used for iterative estimation ($\boldsymbol{Y}(t\text{-}1)$ and $\boldsymbol{Y}(t)$). Each iteration, t, consists of sub-iteration, k. The initial estimation, $\overline{\boldsymbol{X}}$ (k=2), is generated randomly between [lower limit, upper limit] of the first and the second point of the horizon. The iterative estimation, $\overline{\boldsymbol{X}}$ (k=3) until $\overline{\boldsymbol{X}}$ (k=N) are used to improve the solution by using (4) and (5),

$$\overline{\boldsymbol{X}}(k) = \overline{\boldsymbol{X}}(k-1) + K(k)(\boldsymbol{Y}(t-N+k) - \overline{\boldsymbol{X}}(k-1) \quad (4)$$

$$K(k) = \frac{1}{k} \quad (5)$$

where $\overline{\boldsymbol{X}}$ (k) is the estimated solution for current sub-iteration. The estimated solution is depending on the differences between current measurement, $\boldsymbol{Y}$ ($t - N + k$) and the previously estimated solution, $\overline{\boldsymbol{X}}$ (k - 1). The Kalman-like gain, $K(k)$, helps to improve the estimated solution. As the sub-iteration, k increases, the value of $K(k)$ is decreases. The sub-iteration process continues until the k=N. At the end of the sub-iteration, a better estimation is produced. The final updated solution of $\overline{\boldsymbol{X}}$ (k) is then assigned as $\boldsymbol{X}(t)$. Therefore, $\boldsymbol{X}(t)$ represents the estimated value for the corresponding iteration.

After finding the estimated solution, $\boldsymbol{X}(t)$, the evaluation phase is carried out to assess its fitness. The fitness of $\boldsymbol{X}(t)$ is then compared to the fitness of $\boldsymbol{X}\_best\_so\_far$. The $\boldsymbol{X}\_best\_so\_far$ is updated when a better solution is found. For minimization problem, $\boldsymbol{X}\_best\_so\_far$ is updated if fit($\boldsymbol{X}(t)$) < fit($\boldsymbol{X}\_best\_so\_far$), whereas for maximization problem, $\boldsymbol{X}\_best\_so\_far$ is updated if fit($\boldsymbol{X}(t)$) > fit($\boldsymbol{X}\_best\_so\_far$). Measurement and estimation phases continue until the stopping condition, which is the maximum iteration, T, is met. Then, the $\boldsymbol{X}\_best\_so\_far$ returns as the solution, to the given optimization problem.

3. Experimental Setup

The CEC2014 Benchmark Test Suite for single-objective optimization is employed to observe the performance of SAFIRO with different β values. This test suite comprises of 30 functions, which represent 30 real optimization problems. Every function has their own ideal fitness to represent the optimal or the global solution.

As shown in Table 1, the functions are categorized into 4 major groups; unimodal test suite, simple multimodal test suite, hybrid test suite, and composition test suite. The capability of exploitation process of an algorithm can be evaluated by solving unimodal functions, whereas, exploration capability can be evaluated by solving multimodal functions [30]. The capability for both exploration and exploitation, on the other hand, can be evaluated in parallel by solving the composition functions.

In this study, the value of β = 1, β = 5, β = 10, β =*15*, and β = 20 are chosen to be used in (3). β = 1 is assigned to allow a slower size reduction which represents slower transition between exploration phase to exploitation phase, β = 5, on the other hand, allows a moderate transition from the exploration phase to the exploitation phase, whereas β = 10, 15, and 20 allow a faster speed of the transition process from exploration phase to exploitation phase. The aims are to observe the effect of using different values of β towards SAFIRO performance and to determine which setting is the best for SAFIRO. Therefore, by using these values, SAFIRO need to solve 30 optimization problems in CEC2014 Benchmark Test Suite.

To provide a fair comparison, the problem dimension is set as 50 for all variants of SAFIRO, while the maximum iteration is set as 500,000. The stopping condition is set to be the maximum iteration. Each variant of SAFIRO is run 51 times on each of the function. The observation is based on the mean performance of this setting.

The Friedman test for the non-parametric test is then performed to rank these variants of SAFIRO with 5% significant level. The null hypothesis for Friedman test stated that the performances of all tested algorithms are equal, with no significant differences [31]. The performances are ranked statistically according to its mean fitness. The significant differences are then observed.

The Holm post hoc test with tolerance, α=0.05 is then performed to analyze the significant differences of SAFIRO with variants of β values. The null hypothesis of Holm is rejected if the statistical value is smaller than the p-value. In Holm test, the p-value corre-

sponds to their own null hypothesis, $H_1, H_2, H_3, .., H_l, .., H_k$, where the null hypothesis stated that the performance of algorithms are statistically equal [31]. All tests are performed by using the KEEL Software Tool, which can be downloaded through http://www.keel.es.

4. Result and Discussion

Table 2 shows the mean fitness and the mean error values obtained by SAFIRO in solving CEC2014 Benchmark Test Suite, with various values of β. The reading of β=10 is as presented in the original paper of SAFIRO [28]. As the problems in CEC2014 are minimization problems, the smaller reading indicates the better result.

The unimodal test suite is the first group of problems need to be solved by SAFIRO. This test suite consists of three optimization problems (Fn1, Fn2, Fn3) which is related to rotation problems. Fn1 and Fn2 are more difficult to handle because it involved a *quadratic ill-conditioned* property and a *smooth but narrow ridge* property, respectively [29]. The *one sensitive direction* property in Fn3, on the other hand, make it easier to be solved compared to Fn1 and Fn2. Results in Table 2 on the whole, shows that β = 10, is the best setting for solving all unimodal test suite of CEC'14. For Fn3, SAFIRO with β = 10 and β = 15 managed to produce the optimal solution of 300 with 0 mean error. These results indicate that SAFIRO needs a faster speed of the exploration phase compared to the exploitation phase in solving the *Rotated Discus* function. The graph of mean fitness for Fn3 can be shown in Figure 4. The lower reading indicates the better result.

Subsequently, the second group of problems need to be handled by SAFIRO is a simple multimodal test suite which has 13 functions (Fn4 until Fn16). These functions mostly consist of shifting and rotation problems. As tabulated in Table 2, β = 10 is the best setting value for the majority of functions in the simple multimodal test suite. The functions are Fn4, Fn6, Fn10, Fn12, Fn13, Fn15, and Fn16. On the other hand, β = 5 is the best setting for Fn8, Fn9, and Fn11. The readings show that SAFIRO needs moderate transition speed between exploration phase and exploitation phase in solving these three functions which comprise a huge number of local optima.

On the contrary, although Fn5 also has many local optima, SAFIRO requires a larger β value (β = 20) to obtain the best performance. As shown in Figure 5, the larger value of β shows better performance of SAFIRO in solving the *Shifted and Rotated Ackley's* function. Fn7 and Fn14, on the other hand, suite-well with β = 15 to produce the best solution. On top of that, with a suitable value of beta, SAFIRO shows a superior performance in solving this test suite with a very small value of mean error especially for Fn7, Fn12, Fn14, Fn13, Fn15, Fn6, Fn5, and Fn16. A very small value of the mean error indicates SAFIRO able to provide a near-optimal solution.

Next, the hybrid test suite (Fn17 until Fn22) which consist of either combination of several multimodal functions (Fn19, Fn21, and Fn22), or combination of unimodal functions with simple multimodal functions (Fn17, Fn18, and Fn20), make it more challenging to be handled. Table 3 shows the same trend with the simple multimodal test suite where in most functions, the best performances are obtained when β = 10. The functions are Fn17, Fn19, Fn21, and Fn22. The SAFIRO shows a very near-optimal solution for Fn19 with only 20.6 mean error. For Fn18, the best performance is recorded when a higher value of β (β = 20), applied. The graph of mean fitness for Fn18 is depicted in Figure 6. For Fn20, in contrast, the SAFIRO needs a lower value of β (β = 5), to give the best performance.

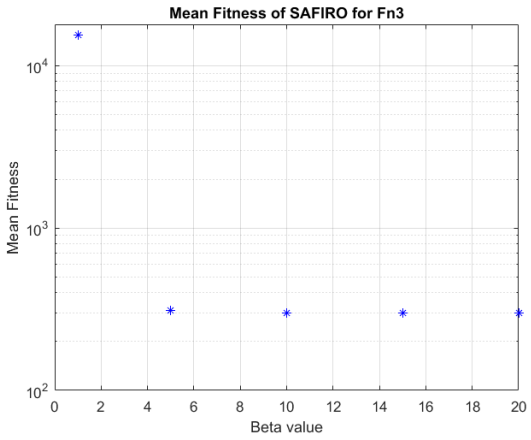

Fig. 4: Mean fitness of SAFIRO for Fn3.

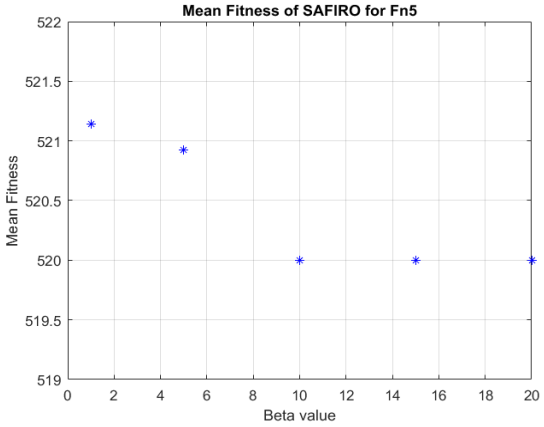

Fig. 5: Mean fitness of SAFIRO for Fn5.

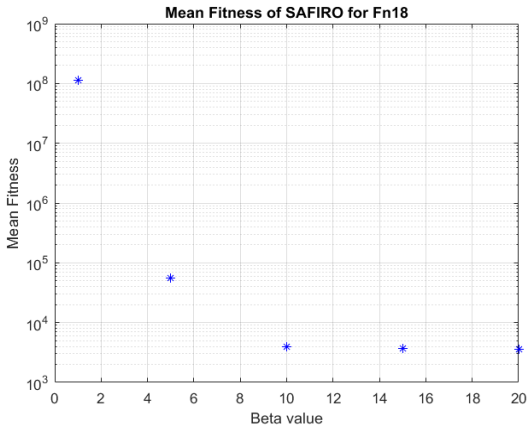

Fig. 6: Mean fitness of SAFIRO for Fn18.

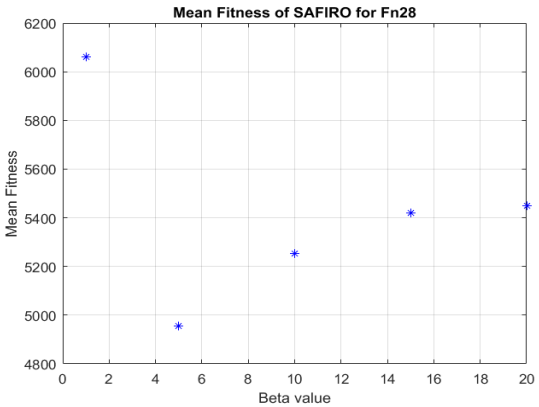

Fig. 7: Mean fitness of SAFIRO for Fn28.

Table 2: Mean fitness and mean error with different β values for unimodal and multimodal functions

Fn	Ideal Fitness		β = 1	β = 5	β = 10	β = 15	β = 20	The best β
1	100	Mean Fitness	9.71E+07	3.41E+06	**7.98E+05**	1.06E+06	1.26E+06	10
		Mean Error	9.71E+07	3.41E+06	7.98E+05	1.06E+06	1.26E+06	
2	200	Mean Fitness	4.23E+09	1.88E+06	**7695.60**	8461.01	1.04E+04	10
		Mean Error	4.23E+09	1.88E+06	7495.60	8261.01	1.02E+04	
3	300	Mean Fitness	1.55E+04	312.82	**300.00**	**300.00**	300.0097	10 & 15
		Mean Error	1.52E+04	12.82	0.00	0.00	0.01	
4	400	Mean Fitness	924.62	502.89	**488.72**	504.45	495.15	10
		Mean Error	524.62	102.89	88.72	104.45	95.15	
5	500	Mean Fitness	521.14	520.93	520.00	520.00003	**519.999997**	20
		Mean Error	21.14	20.93	20.00	20.00	20.00	
6	600	Mean Fitness	643.39	619.32	**619.03**	620.43	621.17	10
		Mean Error	43.39	19.32	19.03	20.43	21.17	
7	700	Mean Fitness	742.33	700.96	700.01	**700.0065**	700.008	15
		Mean Error	42.33	0.96	0.01	0.01	0.01	
8	800	Mean Fitness	1241.49	**973.34**	994.29	1006.09	1004.84	5
		Mean Error	441.49	173.34	194.29	206.09	204.84	
9	900	Mean Fitness	1361.21	**1079.78**	1095.90	1111.61	1114.57	5
		Mean Error	461.21	179.78	195.90	211.61	214.57	
10	1000	Mean Fitness	12804.77	5805.68	**5785.20**	6004.76	6060.21	10
		Mean Error	11804.77	4805.68	4785.20	5004.76	5060.21	
11	1100	Mean Fitness	13839.28	**6172.98**	6462.40	6497.61	6638.93	5
		Mean Error	12739.28	5072.98	5362.40	5397.61	5538.93	
12	1200	Mean Fitness	1203.36	1200.45	**1200.10**	1200.13	1200.16	10
		Mean Error	3.36	0.45	0.10	0.13	0.16	
13	1300	Mean Fitness	1300.78	1300.63	**1300.60**	1300.67	1300.68	10
		Mean Error	0.78	0.63	0.60	0.67	0.68	
14	1400	Mean Fitness	1402.93	1400.47	1400.50	**1400.45**	1400.54	15
		Mean Error	2.93	0.47	0.50	0.45	0.54	
15	1500	Mean Fitness	1607.99	1524.41	**1511.20**	1512.08	1511.82	10
		Mean Error	107.99	24.41	11.20	12.08	11.82	
16	1600	Mean Fitness	1621.63	1620.603	**1620.600**	1621.12	1620.86	10
		Mean Error	21.63	20.60	20.60	21.12	20.86	

The composition test suite (Fn23 until Fn30) is the last group of problems completed by SAFIRO in these simulation experiments. Compared to the other test suites, β = 5 is the best setting for the majority of the functions, where Fn24 until Fn28 show the best results with this setting. Figure 7 shows the graph of mean fitness for Fn28. For the rest functions which are Fn23, Fn29, and Fn30, the best results obtained when β = 10.

Overall, there are 17 out of 30 functions which show the best results with β = 10. Majority of the functions in the unimodal test suite, simple unimodal test suite, and hybrid test suite provides the best performance with these setting. It can be confirmed that none of the CEC2014 functions provided the best performance with β = 1. This means that the SAFIRO needs more exploitation in the search space in solving all functions in CEC2014 Benchmark Test Suite.

The performances of SAFIRO with different β values are then ranked by using the Friedman 1×4 statistical test. The average Friedman ranking of SAFIRO's performance with β = 1, β = 5, β = 10, β = 15, and β = 20 is tabulated in Table 4. According to the table, Friedman test ranks the SAFIRO with β = 10 the highest, followed by β = 15, β = 5, β = 20, and lastly β = 1.

Based on the Friedman test statistic of 75.353, distributed according to a chi-square distribution with 4 degrees of freedom, a significant difference exists between the compared values. To test which values is significantly better than the other, Holm post hoc test is performed. The results are shown in Table 5. From the table, Holm's procedure rejects those hypotheses that have unadjusted *p*-value equal or less than 0.01. Hence, SAFIRO with β=10 performs significantly better than β = 1 and β = 20 but has an equivalent performance with β = 5 and β = 15 in solving the CEC2014 Benchmark Test Suite.

5. Conclusion

It can be concluded that the effect of beta towards SAFIRO performance depends on the given function. Increasing the value of β helps to improve the performance of SAFIRO for Fn5 and Fn18. On the contrary, SAFIRO shows a better performance with a smaller value of β for Fn9, Fn11, Fn20, Fn24, Fn25, Fn27, and Fn28. The statistical tests show that β = 10 has the highest rank where this setting has outperformed β = 1 and β = 20 and has an equivalent performance with β = 5 and β = 15. As a conclusion, the user is recommended to tune the β for best performance. However, β = 10 is a good value to be used in SAFIRO for solving most optimization problems. This is based on the result of the mean fitness (17 out of 30 functions show the best results with β = 10) as well as the statistical tests. Hence, the initial setting in SAFIRO's original paper is considered as a robust value setting for SAFIRO. As a future work, an auto-tuning of parameters in SAFIRO is considered.

Acknowledgement

This research is financially supported by the Universiti Malaysia Pahang (UMP) internal research fund (RDU1703234). The authors would like to thank anonymous reviewers for their constructive comments, and Universiti Malaysia Pahang, Universiti Tun Hussein Onn Malaysia, MOE, and Multimedia University for their logistics support.

Table 3: Mean fitness and mean error with different β values for hybrid and composition functions

Fn	Ideal Fitness		β = 1	β = 5	β = 10	β = 15	β = 20	The best β
17	1700	Mean Fitness	4.78E+06	2.23E+05	**4.60E+04**	5.79E+04	7.83E+04	10
		Mean Error	4.77E+06	2.21E+05	4.43E+04	5.62E+04	7.66E+04	
18	1800	Mean Fitness	1.15E+08	5.48E+04	3.98E+03	3.70E+03	**3.58E+03**	20
		Mean Error	1.15E+08	5.30E+04	2.18E+03	1.90E+03	1.78E+03	
19	1900	Mean Fitness	1968.02	1922.74	**1920.60**	1924.70	1925.85	10
		Mean Error	68.02	22.74	20.60	24.70	25.85	
20	2000	Mean Fitness	4041.59	**2467.45**	2476.10	2499.59	2561.37	5
		Mean Error	2041.59	467.45	476.10	499.59	561.37	
21	2100	Mean Fitness	1.76E+06	2.07E+05	**6.11E+04**	6.75E+04	7.93E+04	10
		Mean Error	1.76E+06	2.04E+05	5.90E+04	6.54E+04	7.72E+04	
22	2200	Mean Fitness	3617.39	2993.56	**2920.90**	3007.73	3024.94	10
		Mean Error	1417.39	793.56	720.90	807.73	824.94	
23	2300	Mean Fitness	2709.71	2646.13	**2645.00**	2645.44	2646.12	10
		Mean Error	409.71	346.13	345.00	345.44	346.12	
24	2400	Mean Fitness	2722.67	**2676.81**	2678.70	2678.80	2679.70	5
		Mean Error	322.67	276.81	278.70	278.80	279.70	
25	2500	Mean Fitness	2733.23	**2711.88**	2712.50	2714.34	2714.89	5
		Mean Error	233.23	211.88	212.50	214.34	214.89	
26	2600	Mean Fitness	2782.51	**2757.37**	2778.70	2772.47	2790.16	5
		Mean Error	182.51	157.37	178.70	172.47	190.16	
27	2700	Mean Fitness	4090.92	**3496.66**	3519.20	3544.06	3587.06	5
		Mean Error	1390.92	796.66	819.20	844.06	887.06	
28	2800	Mean Fitness	6059.83	**4954.57**	5252.50	5421.25	5449.55	5
		Mean Error	3259.83	2154.57	2452.50	2621.25	2649.55	
29	2900	Mean Fitness	1.40E+07	3.78E+04	**2.89E+04**	3.03E+04	3.06E+04	10
		Mean Error	1.40E+07	3.49E+04	2.60E+04	2.74E+04	2.77E+04	
30	3000	Mean Fitness	2.21E+05	4.72E+04	**3.87E+04**	4.21E+04	4.34E+04	10
		Mean Error	2.18E+05	4.42E+04	3.57E+04	3.91E+04	4.04E+04	

Table 4: Average Friedman ranking of β

Algorithm	Ranking
β = 10	1.5833
β = 15	2.5833
β = 5	2.6
β = 20	3.2667
β = 1	4.9667

Table 5: Holm post Hoc result of β value for α = 0.05

β	p	Holm
β = 1 vs β = 10	0	0.005
β = 1 vs β = 15	0	0.005556
β = 1 vs β = 5	0	0.00625
β = 1 vs β = 20	0.000031	0.007143
β = 10 vs β = 20	0.000037	0.008333
β = 5 vs β = 10	0.012763	0.01
β = 10 vs β = 15	0.014306	0.0125
β = 15 vs β = 20	0.094166	0.016667
β = 5 vs β= 20	0.10247	0.025
β = 5 vs β= 15	0.967436	0.05

References

[1] Kirkpatrick S, Gelatt CD, & Vecchi MP (1983), Optimization by Simulated Annealing, *Science*, Vol. 220, No. 4598, pp. 671–680.

[2] Glover F (1989), Tabu Search - Part I, *ORSA J. Comput.*, Vol. 1, No. 3, pp. 190–206.

[3] Solis FJ & Wets RJB (1981), Minimization By Random Search Techniques, *Math. Oper. Res.*, Vol. 6, No. 1, pp. 19–30.

[4] Michael Lewis R & Torczon V (1996), Pattern Search Algorithms for Bound Constrained Minimization.

[5] Feo TA & Resende MGC (1995), Greedy Randomized Adaptive Search Procedures, *J. Glob. Optim.*, Vol. 6, pp. 109–133.

[6] Mladenovic N & Hansen P (1997), Variable Neighborhood Search, *Comput. Oper. Res.*, Vol. 24, No. 11, pp. 1097–1100.

[7] Voudouris C & Tsang E (1999), Guided Local Search and its Application to the Traveling Salesman Problem, *Eur. J. Oper. Res.*, Vol. 113, No. 2, pp. 469–499.

[8] Lourenço HR, Martin O, & Stützle T, A Beginner's Introduction to Iterated Local Search, *Proceeding 4th Metaheuristics Int. Conf., June 2014*, (2001), pp. 4-11.

[9] Boussaid I, Lepagnot J & Siarry P (2013), A Survey on Optimization Metaheuristics, *Inf. Sci. (Ny).*, Vol. 237, pp. 82–117.

[10] Yang XS (2015), *Recent Advances in Swarm Intelligence and Evolutionary Computation*, Springer International Publishing Switzerland.

[11] Yang XS, Deb S & Fong S (2014), Metaheuristic Algorithms: Optimal Balance of Intensification and Diversification, *Appl. Math. Inf. Sci.*, Vol. 8, No. 3, pp. 977–983.

[12] Dumitrescu I & Stützle T (2013), A Survey of Methods that Combine Local Search and Exact Algorithms, *Appl. Evol. Comput.*, pp. 57–68.

[13] Osman IH & Laporte G (1996), Metaheuristics: A Bibliography, *Ann. Oper. Res.*, Vol. 63, No. 5, pp. 511–623.

[14] Héliodore F, Amir N, Ismail B, Ouchraa S & Schmitt L (2017), *Metaheuristics for Intelligent Electrical Networks*, ISTE Ltd and John Wiley & Sons, Inc.

[15] Bertsimas D & Tsitsiklis J (1993), Simulated Annealing, *Statistical Science*, Vol. 8, No. 1. pp. 10–15.

[16] Blum C & Roli A (2003), Metaheuristics in Combinatorial Optimization : Overview and Conceptual Comparison, Vol. 35, No. 3, pp. 268–308.

[17] Ibrahim A, Rahnamayan S & Martin MV, Simulated Raindrop Algorithm for Global Optimization, *Canadian Conference on Electrical and Computer Engineering*, (2014), pp. 1–8.

[18] Idoumghar L, Melkemi M, Schott R, and Aouad MI (2011), Hybrid PSO-SA Type Algorithms for Multimodal Function Optimization and Reducing Energy Consumption in Embedded Systems, *Appl. Comput. Intell. Soft Comput.*, Vol. 2011, pp. 1–12.

[19] Sevaux M, Sörensen K & Glover F (2016), *Metaheuristics*, October 2016.

[20] Beheshti Z & Shamsuddin SM (2013), A Review of Population-based Meta-Heuristic Algorithms, *Int. J. Adv. Soft Comput. Appl.*, Vol. 5, No. 1, pp. 1–35.

[21] Saremi S, Mirjalili S & Lewis A (2017), Grasshopper Optimisation Algorithm: Theory and Application, *Adv. Eng. Softw.*, Vol. 105, pp. 30–47.

[22] Glover F & Kochenberger G (2003), *Handbook of Metaheuristics*, Kluwer Academic Publishers.

[23] Dogan B & Olmez T (2015), A New Metaheuristic for Numerical Function Optimization: Vortex Search Algorithm, *Inf. Sci. (Ny).*, Vol. 293, pp. 125–145.

[24] Rueda JL & Erlich I, MVMO for Bound Constrained Single-objective Computationally Expensive Numerical Optimization," *2015 IEEE Congress on Evolutionary Computation*, (2015), pp.

1011–1017.
[25] Abdul Aziz NH, Ibrahim Z, Ab. Aziz NA, Mohamad MS & Watada J (2018), Single-solution simulated Kalman filter algorithm for global optimisation problems, *Sadhana*, Vol. 123, No. 4, pp. 2333–2335.
[26] Ibrahim Z, Abdul Aziz NH, Ab. Aziz NA, Razali S, Shapiai MI, Nawawi SW & Mohamad MS (2015), A Kalman Filter Approach for Solving Unimodal Optimization Problems, *ICIC Express Letters*, Vol. 9, Issue 12, pp. 3415-3422.
[27] Ibrahim Z, Abdul Aziz NH, Ab. Aziz NA, Razali R & Mohamad MS (2016), Simulated Kalman Filter: A Novel Estimation-Based Metaheuristic Optimization Algorithm, *Advance Science Letters*, Vol. 22, pp. 2941-2946.
[28] Ab Rahman T, Ibrahim Z, Ab. Aziz NA, Zhao S & Abdul Aziz NH (2018), Single-Agent Finite Impulse Response Optimizer for Numerical Optimization Problems, *IEEE Access*, Vol. 6, pp. 9358-9374.
[29] Liang JJ, Qu BY & Suganthan PN (2013), Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization.
[30] Mirjalili S, Mirjalili SM & Lewis A (2014), Grey Wolf Optimizer, *Adv. Eng. Softw.*, Vol. 69, pp. 46-61.
[31] Ab. Aziz NA, Mubin M, Ibrahim Z & Nawawi SW (2015), Statistical Analysis for Swarm Intelligence - Simplified, *Int. J. Futur. Comput. Commun.*, Vol. 4, No. 3, pp. 193-197, 2015.