Center-Point-Based Simulated Annealing

Ali Esmailzadeh, *IEEE Member*, Shahryar Rahnamayan, *IEEE Member*

***Abstract*— The S-metaheuristic algorithms work with a single candidate-solution during the search process. That is why they are prone to be trapped in local optima. Many research has being conducted to speed up and also minimize their premature convergence. The *Center-point Sampling* was introduced by Rahnamayan and Wang in 2008. Based on their experiments, it has shown increase in probability of closeness of the unique point in the center of the search space, to an unknown solution, as the dimensionality of the problem increases. It means, the center is an exceptional point to be used as initial point, specially during solving large-scale black-box problems. In this paper, we investigate this phenomena on Simulated Annealing (SA). The purpose is to accelerate the convergence speed of the algorithm by using the center point as an initial point for SA algorithm. This modified version, called Center-Point-Based SA (CSA), is a very simple and effective idea to enhance SA. The experimental verifications are provided on seven shifted large-scale (i.e., D=300) benchmark functions to show improvements achieved by the CSA algorithm. Using the shifted version of the functions ensures there is no bias towards the center, and so towards CSA algorithm. The results confirm that CSA outperforms parent SA algorithm in overall.**

***Index Terms*— S-Metaheuristic, Simulated Annealing, Center-based Sampling, Large Scale Optimization.**

I. Introduction

The S-metaheuristic algorithms, unlike P-metaheuristic algorithms, are single-solution-based algorithms which consider a single candidate-solution in the search space to solve a problem. That is why, they are more prone to getting trapped in local optima, specially on multi-modal and complicated landscapes, as a result, they converge prematurely. Many research has been conducted to improve S-metaheuristic algorithms such that they do not get trapped easily or can escape. Therefore, the initial candidate-solution which is used to solve a problem in a S-metaheuristic algorithm is very important, as well, it is important to guide the candidate through the landscape such that it does not get trapped.

The Simulated Annealing (SA) algorithm was introduced by Kirkpatrick et al. in 1983 [7], as well, by Cerny [8], in two independent works. It is a S-metaheuristic algorithm that starts from an initial candidate-solution and by generating and selecting neighbors, it keeps moving on the landscape until it finds the solution. There has been some research done to improve the neighbor generation of SA. The neighbor generation of SA algorithm was modified in an algorithm introduced by Ventresca et al. [1], called *Opposition-Based Simulated Annealing* (OSA). It was based on the *Opposition-Based Learning* (OBL) concept proposed by Tizhoosh in 2005 [2]. For OSA [1], when a new neighbor is generated, the corresponding *opposite* neighbor is considered and the fitter point survives as the new *current* point. It was shown that OSA outperformed SA on majority of tested benchmark functions.

An improved and efficient SA algorithm was introduced and utilized by Ji-Yang [3], in which it modifies the search process such that it is no longer memory-less process. By adding memory to the search process of original SA, the improved algorithm keeps track of which neighbors have already been searched, and it will not search those again, similar to the Tabu Search (TS) algorithm. This addition to SA improves the efficiency and solution found of the algorithm.

An *adaptive simulated annealing* (ASA) was introduced by Youhua et al. [4], in which the temperature decrease (i.e., cooling schedule) of the SA algorithm is adaptive based on the current results of the algorithm.

In another work, Xinchao introduced *Simulated annealing algorithm with adaptive neighborhood* [5], which the neighborhood coverage is adaptive. During exploration, the coverage area is wider and covers the entire search space, but during exploitation it gets smaller adaptively.

In this paper, instead of working on the neighbor generation component of SA, we define a new and simple strategy for the initializing starting point of SA, but this idea can be investigated on any other S-metaheuristic algorithm. We verify our proposed approach by conducting experiments on seven shifted large scale benchmark functions. We have decided to test the method on shifted benchmark functions, so that there is no bias towards the center as well as the proposed algorithm.

The rest of this paper is organized as follows: in Section II, the concept of Center-Point Sampling is reviewed and discussed. The proposed Center-Point-Based Simulated Annealing is presented in Section III. The experimental results and analysis are given in Section IV. Finally, the work is concluded in Section V.

II. Center-Point Sampling: A Review

In this section, we review and explain the *Center-Point* sampling. It was introduced by Rahnamayan and Wang in 2009 [11]. They investigated the closeness of points in a search space from an unknown solution (black-box problem). They measured the Euclidean distances of the points to the unknown solution for the different dimensions. They utilized Monte-Carlo simulation by dividing the $[a,b]$ search space interval into partitions of 10^{-3} step-sizes, to represent a fixed

Faculty of Engineering and Applied Science, University of Ontario Institute of Technology (UOIT), 2000 Simcoe Street North, Oshawa, Ontario, L1H 7K4, Canada (phone: 1-(905)-721-8668 ext. 3843, fax: 1-(905)-721-3178, emails: (corresponding author) ali.esmailzadeh@uoit.ca; shahryar.rahnamayan@uoit.ca).

2012 25th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)
978-1-4673-1433-6/12/$31.00 ©2012 IEEE

point in a given calculation. For each fixed-point, x, in each dimension $(1, 2, 3, ..., D)$, they repeat the following steps 10^6 times (i.e. trials) in order to get measurable results:

1) Generate a uniform-random point, r, and a random *unknown-solution*, s, in the search space [a,b].
2) Measure the Euclidean distance of the fixed-point and the uniform-random point, from the unknown-solution.
3) Depending on which distance measure is smaller, the appropriate distance variables are updated for calculating the average distance and probability of closeness, at the end of the 10^6 trials.

By Monte-Carlo simulations, they have found that the probability of points being closer to an *unknown solution* (in comparison to uniformly generated points over the entire space) is much greater towards the center of the search space. Given the interval of search space as [a,b], shown in Fig. 1, the center of the interval is formulated by Eq. 1 and Eq. 2, for dimensions 1 and n, as follows:

For 1D:

$$c = \frac{(a+b)}{2} \tag{1}$$

For n-D:

$$c_i = \frac{(a_i+b_i)}{2} \tag{2}$$

where $i = 1, ..., D$ and D is the dimension of the problem.

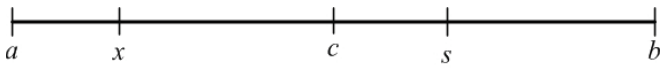

Fig. 1. The visual illustration (in 1D) of uniform-random point, x, and the unknown-solution, s, in the interval [a,b], where c indicates corresponding center of the search space, c=(a+b)/2.

The authors observed that as the candidate-solutions gets closer to the center of the search space, the probability of closeness to the unknown solution increases drastically. Furthermore, they examined this phenomenon to solve high-dimensional problems [11]. Interestingly, as the dimensionality of the problem increases, the probability of closeness to the solution improves as well. They experimented this concept for large dimensions of 100, 200, 500 and 1000. Even for low dimension of one, the probability of closeness to the solution is greater around the center-point. The probability graph is shown in Fig. 2.

Furthermore, the matchable improvements near the center-point can be seen with regards to the average distance of candidate-solutions to an unknown solution. In the Fig. 3, it can be seen that as the dimensionality of the problem increases, the average distance of candidate-solutions near and on the center-point decreases to very low values, close to 0.

The probability of closeness of center-point to an unknown solution has the highest value, as shown in Fig. 4.

In this section, we will try to intuitively explain the findings of [11]. The simulation results in [11], as reviewed in this section, indicate that the center of a search space is, on average, the closest to the unknown-solution. Even

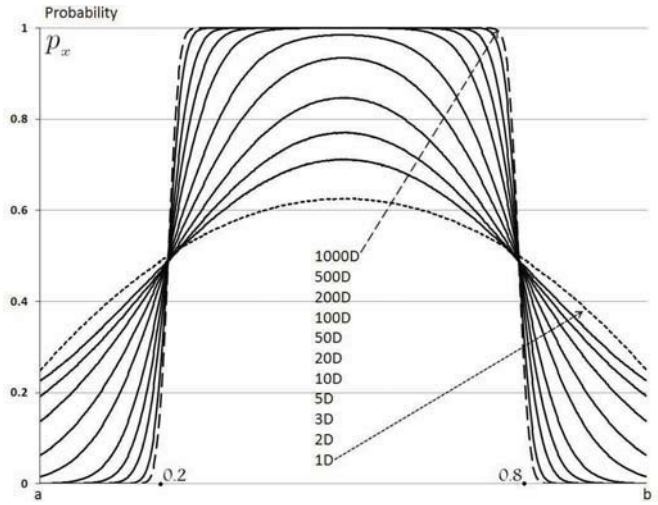

Fig. 2. The graphs of Monte-Carlo simulations which present the probability of closeness of candidate-solution to an unknown solution in the interval [a,b], for different dimensions [11].

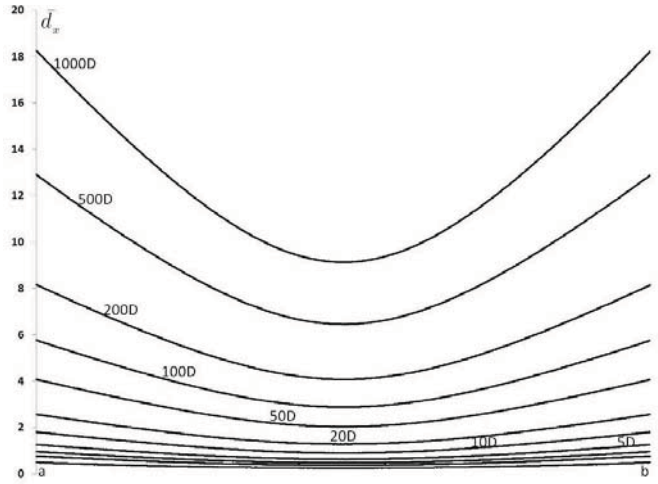

Fig. 3. The graphs of Monte-Carlo simulations which present the average distance of candidate solution to an unknown solution in the interval [a,b], for different high dimensions [11].

though this is a novel concept and it has been justified by Monte-Carlo simulations, it still has to be justified intuitively. According to Fig. 1, c is the middle of the search space, which has divided the entire search interval of [a,b] into equal (symmetric) sub-intervals of [a,c] and [c,b]. The candidate-solution x, and unknown solution s, can each be in different sub-intervals, or they can both be in the same sub-interval. The chances of either of the previous cases are 50% since we are dealing with uniform-random. For the former case, since c is in between x and s, no matter which sub-interval s belongs to, the Euclidean distance of x and c to s is $|x - s| \geq |c - s|$. Therefore, on average, in 50% of the times, c is always closer to s, than x is to s. Therefore, on 50% of the times, c is closer to the unknown solution. For the rest of the probability, where x and s are in the same sub-interval, then x and c are competing together for

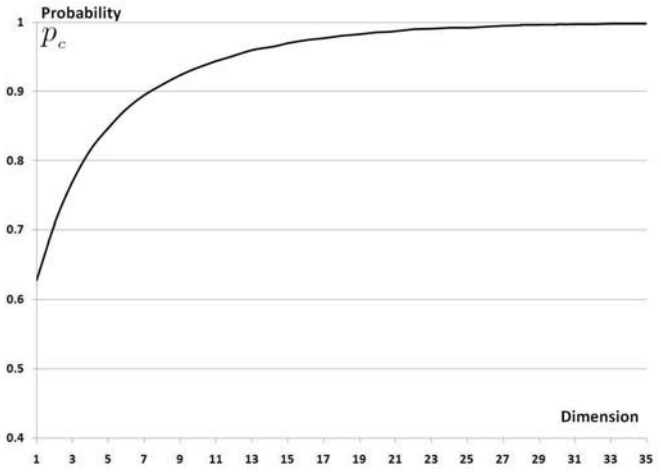

Fig. 4. The graphs of Monte-Carlo simulations which present the probability of closeness of center-point to an unknown solution, for dimensions of 1 to 35 [11].

closeness to s. Therefore, any probability of c closer to s in this scenario, along with the 50% chance in the first scenario, will only help increasing the chances of c being closer to s, in overall. Furthermore, the same probability of closeness of c to s can be applied for the other dimensions of a D dimensional problem. Therefore, over all the dimensions, the probability of closeness of c will be improved. That is why as the dimensionality of the problem increases, the probability of closeness of center increases as well, as shown in Fig. 2.

As indicated by the authors in [11], *Center-point* is a unique point in the search space; therefore, it can be easily utilized for S-metaheuristic algorithms.

In the next section, we will propose a modified SA algorithm that combines Center-point concept and SA, in order to investigate the effectiveness of Center-point concept in solving large-scale optimization problems.

III. Center-Point-Based Simulated Annealing (CSA)

In this section, we propose a simple modification to the SA algorithm in term of the initial starting point of the algorithm. As discussed in Section II, the Center-point is a unique point, which has the highest probability of being closer to the unknown solution, than any other random point generated in the entire search space. In general, when there is no any priori-knowledge about the problem, then start from a uniform random point.

In this paper, we aim to utilize the Center-point concept on a S-metaheuristic algorithm in order to accelerate its convergence speed. The implementation of Center-point concept in SA will be made at the initialization step. Therefore, in the initial estimate phase of SA where a random guess about the starting point on the search space has to be made, we simply utilize Center-point as the initial starting point for the search. In CSA, the initial starting point for the algorithm to solve the problem will be the point c of the search space, according to Eq. 2, with respect to each dimension.

We aim to not change the classical SA algorithm components for the center-point version (CSA). We do not utilize the *Center-point* concept during neighbor generation of SA. Simply, instead of generating a uniform random point on search space as an initial point for SA, we pick the center of the search space. The rest of the SA search process is then carried out followed with the center-point initial candidate. The complexity of the CSA algorithm is the same as its parent, SA algorithm because $O(1)$ is the complexity for finding the center of the search space.

Although this is a very simple idea, the intention of the current work is to examine the effect of using initial Center-point, to solve large-scale problems. As mentioned in Section II, for large-scale problems Center-point showed to have very high probability of being closer to an unknown solution. Therefore, we intend to test the CSA method with its parent (SA) on large-scale problems. To the best of our knowledge, no research work has tried the center point, as oppose to uniform-random point, as the initial starting point in the SA algorithm. We aimed at trying this combination in order to verify the effect of center-point, on a single-solution-based algorithm.

IV. Experimental Verifications

This section gives detailed description of the experiments performed to verify the effectiveness of CSA

A. Control Parameter Settings

All the common parameters of SA and CSA are set to the same values as below, in order to have a fair comparison.

- New Solution Generator: $Point_{current}$+$rand$(1,D)/10
- Starting Temperature, T_{max}=300
- Stop Temperature, T_{Stop}=$1E-8$
- Cooling Schedule, T=0.95×T
- Maximum Consecutive Rejections, Max_{Reject}=1000
- Max Success, 20
- Max Tries, 300

B. Benchmark Functions

The following benchmark functions are bound constrained high-dimensional benchmark functions for minimization as provided by the *CEC'2008 Special Session on Large Scale Global Optimization* [10]. The benchmark functions used in this paper are the ones used in [9] for large-scale problems. The functions $f1$ and $f2$ are unimodal and the rest of the functions are multimodal problems. Their complete list with more details are presented in Appendix A. All functions are randomly shifted over the search space to not have any kind of favor for the center point of the search space. Therefore, for none of the functions, the solution is located at the center.

C. Simulation Strategy

Similar to other research papers [1], for all conducted experiments, trials are repeated 250 times per function. Mean and standard deviation of the best fitness values are reported. However, the re-sampling and thresholding techniques [12] are not applied in this paper.

D. Simulation Results

The numerical results for SA and CSA algorithms on seven benchmark functions are summarized in Table I.

TABLE I

MEAN ± (STANDARD DEVIATION) OF THE BEST FITNESS VALUE OF SA COMPARED TO CSA, FOR D=300. THE BEST RESULT FOR EACH FUNCTION IS HIGHLIGHTED IN **BOLDFACE**.

F	SA	CSA
f_1	$966.214 \pm (1454.404)$	**241.756 ± (18.133)**
f_2	$195.730 \pm (56.980)$	**190.450 ± (1.765)**
f_3	$0 \pm (0)$	$0 \pm (0)$
f_4	$5032.840 \pm (340.945)$	**5031.222 ± (130.252)**
f_5	$62.584 \pm (64.521)$	**32.278 ± (0.804)**
f_6	**21.579** $\pm (0.396)$	$21.667 \pm$ **(0.390)**
f_7	**300.804** $\pm (11.540)$	$301.378 \pm$ **(3.499)**

E. Results Analysis

As indicated in the Table I, the SA and CSA algorithms performed the same best value of 0 for function f_3. As seen, the CSA algorithm has outperformed SA on 4 out of 7 functions, f_1, f_2, f_4 and f_5. By observing the high standard deviations for functions f_1 and f_4 by SA, it could be an indication that SA gets trapped in local optima for those functions. On only 2 functions out of 7, SA performed better than CSA, i.e., f_6 and f_7. The improvements for those functions by SA are not by a large value, and considering the better standard deviation values of CSA for f_6 and f_7, both SA and CSA algorithms performed nearly the same on those two functions.

Another contribution of CSA and its improvement to the SA algorithm can be seen by the standard deviation (STD) values of CSA compared to SA, for 6 out of 7 functions. As it can be observed, the STD values of CSA are much lower than SA. This indicates a higher consistency of results by CSA, compared to SA. The higher consistency of CSA means lower behavior fluctuation and so more accurate results. The lower STD values and higher consistency of results of CSA can be due to the fact that instead of starting from a random point each time, we always start from a fix point, the center. This has shown to be a good starting point in previous experiments and has yield better and more consistent results, compared to a uniform-random point. In overall, CSA algorithm presents promising results for solving large scale problems.

V. CONCLUDING REMARKS

The S-metaheuristic algorithms such as Simulated Annealing (SA) starts from one candidate-solution as the initial starting point which usually is a uniformly generated random point in the search space. Moreover, S-metaheuristic algorithms are prone to get trapped in local optima on difficult multi-modal landscapes. According to the Monte-Carlo simulations the center point is a unique point at the center of the search space which has a very high probability of closeness to an unknown solution, comparing to a uniform random point in the entire search space. The probability of closeness increases as the dimensionality of a problem increases. This paper, introduced *Center-Point-Based Simulated Annealing* (CSA), which utilizes the *center-point* to accelerate SA algorithm. For CSA algorithm, simply, we use the center point of the search space, which is a unique point, as the initial starting point for the SA algorithm; then the original SA algorithm steps are executed. Although this is a very simple idea, but it has shown to be effective and better performing than its parent algorithm. The experimental verifications carried out in this paper were on seven shifted large-scale benchmark functions, for dimension of D=300. The purpose of testing on shifted functions was to ensure that there is no bias towards the center of the search space, and that the optimum solution can be randomly placed anywhere in the space. Since *Center-point* concept was shown by other research as an exceptional feature for higher dimensions, we decided to test CSA on large-scale problems. The results verify that on four out of seven functions, CSA outperforms SA, and for one of the functions, both SA and CSA performed the same. Furthermore, the STD values of CSA on all seven functions were much lower than those from SA. This indicates that CSA has more consistency and less fluctuation than SA. These positive results indicate that for solving large-scale problems, CSA is the winner algorithm.

REFERENCES

[1] M. Ventresca, H.R. Tizhoosh, *Simulated Annealing with Opposite Neighbors*, IEEE Symposium on Foundations of Computational Intelligence (FOCI'07), Honolulu, pp. 186-192, 2007.

[2] H.R. Tizhoosh, *Opposition-Based Learning: A New Scheme for Machine Intelligence*, Int. Conf. on Computational Intelligence for Modelling Control and Automation (CIMCA'2005), Vienna, Austria, Vol. I, pp. 695-701, 2005.

[3] Q. Ji-Yang, *Application of Improved SimulatedAnnealing Algorithm in Facility Layout Design*, Proceedings of the 29th Chinese Control Conference (CCC'10), Beijing, China, pp. 5224-5227, July 2010.

[4] W. Youhua, Y. Weili, Z. Guansheng, *Adaptive Simulated Annealing for the Optimal Design of Electromagnetic Devices*, IEEE Transactions On Magnetics, Berlin, Germany, Vol. 32, No. 3, pp. 1214-1217, May 1996.

[5] Z. Xinchao, *Simulated Annealing Algorithm With Adaptive Neighborhood*, Journal of Applied Soft Computing, In Press.

[6] E.-G. Talbi, "Initial Population," in *METAHEURISTICS: FROM DESIGN TO IMPLEMENTATION*, Hoboken, New Jersey: John Wiley & Sons, 2009, pp. 126-133.

[7] S. Kirkpatrick, C.D. Gelatt, M. P. Vecchi, *Optimization by Simulated Annealing: Quantitative Studies*, Journal of Statistical Physics, Vol. 34, No. 5 and 6, pp. 975-986, 1984.

[8] V. Cerny, *Thermodynamical Approach to the Traveling Salesman Problem: An Efficient Simulation Algorithm*, Journal Of Optimization Theory And Applications, Vol. 45, No. 1, pp. 41-51, 1985.

[9] S. Rahnamayan, G. Gary Wang, *Solving Large Scale Optimization Problems by Opposition-Based Differential Evolution (ODE)*, World Scientific and Engineering Academy and Society, Transactions on Computers, Volume 7, Issue 10, Oct. 2008, pp. 1792-1804.

[10] K. Tang, X. Yao, P. N. Suganthan, C. Mac- Nish, Y. P. Chen, C. M. Chen, Z. Yang, *Benchmark Functions for the CEC2008 Special Session and Competition on Large Scale Global Optimization*, Technical Report, Nature Inspired Computation and Applications Laboratory, USTC, China, http://nical.ustc.edu.cn/cec08ss.php, 2007.

[11] S. Rahnamayan, G. Gary Wang, *Center-Based Sampling for Population-Based Algorithms*, IEEE Congress on Evolutionary Computation (CEC'09), pp. 933-938, May 2009.

[12] Yaochu Jin and Jürgen Branke, *Evolutionary Optimization in Uncertain Environments- A Survey*, IEEE Transactions on Evolutionary Computatioon, Vol. 9, No. 3, pp. 303-317, June 2005.