[Applied Computing and Informatics 14 (2018) 107–115](http://dx.doi.org/10.1016/j.aci.2017.09.001)



Contents lists available at [ScienceDirect](http://www.sciencedirect.com/science/journal/22108327)

Applied Computing and Informatics

journal homepage: [www.sciencedirect.com](http://www.sciencedirect.com/)

An adaptive Cuckoo search algorithm for optimisation



M. Mareli [a](#_bookmark0),[⇑](#_bookmark2), B. Twala [b](#_bookmark1)

a *Department of Electrical and Electronics Engineering, University of Johannesburg, PO Box 542, Auckland Park 2006, South Africa*

b *Institute of Intelligent Systems, Department of Electrical and Electronics Engineering University of Johannesburg, PO Box 542, Auckland Park 2006, South Africa*

# a r t i c l e i n f o

*Article history:*

Received 18 May 2017

Revised 30 August 2017

Accepted 3 September 2017

Available online 6 September 2017

*Keywords:* Cuckoo search Optimisation Random walk Lévy distribution

Pareto distribution Test functions

# a b s t r a c t

Cuckoo search is one of many nature-inspired algorithms used extensively to solve optimisation prob- lems in different fields of engineering. It is a very effective in solving global optimisation because it is able to maintain balance between local and global random walks using switching parameter. The switching parameter for the original Cuckoo search algorithm is fixed at 25% and not enough studies have been done to assess the impact of dynamic switching parameter on the performance of Cuckoo search algo- rithm. This paper’s contribution is the development of three new Cuckoo search algorithms based on dynamically increasing switching parameters. The three new Cuckoo search algorithms are validated on ten mathematical test functions and their results compared to those of Cuckoo search algorithms with constant and dynamically decreasing switching parameters respectively. Finally, the simulations in this study indicate that, the Cuckoo search algorithm with exponentially increasing switching parameter out- performed the other Cuckoo search algorithms.

© 2017 The Authors. Production and hosting by Elsevier B.V. on behalf of King Saud University. This is an

open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

Contents

1. Introduction 107
2. Cuckoo search 108
   1. Cuckoo breeding behaviour 108
   2. Lévy flights 108
   3. Cuckoo search algorithm 108
   4. Some Cuckoo search algorithm improvements 109
      1. Step size improvements 109
      2. Other improvements 109
3. Proposed Cuckoo search 109
4. Results and discussions 110
   1. Simulation setup 110
   2. Characteristics of test functions 114
   3. Simulations results and discussions 114
5. Conclusions 114

Acknowledgements 114

References 114

\* Corresponding author.

*E-mail addresses:* [mmareli@hotmail.com](mailto:mmareli@hotmail.com) (M. Mareli), [btwala@uj.ac.za](mailto:btwala@uj.ac.za) (B. Twala).

Peer review under responsibility of King Saud University.

**Production and hosting by Elsevier**

1. Introduction

Optimisation plays an important role in solving different engi- neering problems. The goal of optimisation process is to determine

<http://dx.doi.org/10.1016/j.aci.2017.09.001>

2210-8327/© 2017 The Authors. Production and hosting by Elsevier B.V. on behalf of King Saud University.

This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

either a maximum or a minimum value of the problem being solved, generally known as the objective function [[1]](#_bookmark19). These prob- lems include but not limited to systems design, electricity network operation, electricity generation, wireless communications routing and minimisation of energy losses during electricity transmission. Proper validations of optimisation algorithms require assessment of computational time and convergence rate in addition to the accuracy to determine the minimum or maximum values [[2–8]](#_bookmark20).

Some researchers have innovated optimisation algorithms based on nature observations, these algorithms are known as nature-inspired algorithms. In [[9]](#_bookmark21) a bat-inspired algorithm was developed based on echolocation to sense distance between a bat and its surroundings. Particle swarm optimisation (PSO) was inno- vated after observing behaviour of fish and birds schooling [[10]](#_bookmark21). Differential evolution (DE) algorithm was created by Storm and Prince [[11]](#_bookmark21) based on population evolution using mutation, cross- over and selection operations. Ant and bee algorithms were devel- oped based on the foraging behaviour of ants and bees that use pheromone as a chemical messenger and the concentration of pheromone is regarded as indication of quality solutions to the problem being solved [[12]](#_bookmark21). Simulated annealing (SA) is based on the characteristics of the metal annealing process [[13]](#_bookmark21).

Cuckoo search (CS) algorithm is also a nature-inspired algo- rithm, based on brood reproductive strategy of cuckoo birds to increase their population [[14]](#_bookmark21). However, CS is more effective than other nature-inspired algorithms. In fact, DE, SA and PSO are spe- cial cases of CS algorithm, hence it is no surprising why CS algo- rithm outperforms them [[14]](#_bookmark21). In [[15]](#_bookmark21) CS algorithm outperformed DE algorithm in terms of convergence speed to reach optimum solution. In addition, CS algorithm was reported as being more computationally efficient than the PSO [[16]](#_bookmark21).

Baskan [[17]](#_bookmark21) used CS algorithm to minimise traffic congestion by improving the performance of transportation road networks. The objective function was defined as a total of travel time and invested cost of 16 link capacity expansions. Cuckoo search pro- duced best results when compared to other methods found in the literature. In [[18]](#_bookmark22) CS algorithm was used to maintain fault level and the voltage fluctuations within acceptable level, thus minimise real power losses in a smart grid.

The aim of this paper, is to develop and assess the performance of Cuckoo search algorithms with increasing switching parameter between local and global random walks as the number of iteration increases. The first CS algorithm has linearly increasing switching parameter. The second CS algorithm implements exponentially increasing switching parameter, and the third CS algorithm uses increasing power switching parameter.

The rest of this paper is organised as follows: Section [2](#_bookmark3) dis- cusses the Cuckoo bird’s breeding behaviour, Lévy flights and CS algorithm and some of CS algorithm improvements carried out by other scholars to make CS algorithm even more effective. Three new CS algorithms are proposed in Section [3](#_bookmark7). Simulation results and their interpretations are discussed in Section [4](#_bookmark12). Finally, the paper conclusions are presented in Section [5](#_bookmark17).

1. Cuckoo search
   1. *Cuckoo breeding behaviour*

The parasitic cuckoos are good in sporting nests where eggs have just been laid and their timing of laying eggs is very precise. They lay one egg in the host nest which will normally hatch quicker than the other eggs. When this happens, the foreign cuckoo would remove the non-hatched eggs from the nest by pushing the eggs out of the nest. This behaviour is aimed at reducing the prob- ability of the legitimate eggs from hatching. Furthermore, the for- eign cuckoo chick can gain access to more food by mimicking the call of the host chicks. There are times when the host cuckoo dis- covers that one of the eggs is foreign. In that case the cuckoo either gets rid of the egg or abandon the nest altogether and moves to build a new nest somewhere else [[19]](#_bookmark23).

* 1. *Lévy flights*

Lévy flights are random walks whose directions are random and their step lengths are derived from the Lévy distribution. These Lévy flights are performed by animals and insects and it is charac-

terised by series of straight flights followed by sudden 90*degree* turns. Compared to normal random walks, Lévy flights are more efficient in exploring large–scale search areas. That is mainly due to Lévy flights variances increases much faster than that of the nor- mal random walk. Lévy flights can reduce the number of optimisa- tion algorithms iterations by about 4 orders compared to normal random walk [[14]](#_bookmark21).

* 1. *Cuckoo search algorithm*

Cuckoo search algorithm is a nature-inspired algorithm devel- oped based on reproduction of cuckoo birds [[14]](#_bookmark21). While working with CS algorithms, it is important to associate potential solutions with cuckoo eggs. Cuckoos normally lay their fertilised eggs in other cuckoos’ nests with the hope of their off-springs being raised by proxy parents. There are times when the cuckoos discover that the eggs in their nests do not belong to them, in those cases the for- eign eggs are either thrown out of the nests or the whole nests are abandoned. The CS optimisation algorithm is basically based on the following three rules:

* Each cuckoo selects a nest randomly and lays one egg in it.
* The best nests with high quality of eggs will be carried over to the next generation.
* For a fixed number of nests, a host cuckoo can discover a foreign egg with a probability *pa* є [0,1]. In this case, the host cuckoo can

either throw the egg away or abandon the nest and build a new one somewhere else.

The last rule can be approximated by replacing a fraction *pa* of the n host nests with new nests (with new random solutions). The quality or fitness of a solution can simply be proportional to

the value of the objective function. From the implementation point of view, the representation that is followed is that each egg in a

Table 1

Local and Global random walk parameters.

Parameter Description

xt and xt Current positions selected by random permutation.

i k

x

Cuckoos are a family of birds with unique reproductive strategy

a Positive step size scaling factor.

more aggressive compared to other bird’s species. Some of cuckoo

t+1 i

Next position.

bird’s species like Ani and Guira lay eggs in communal nests; how- ever, they may remove others’ eggs to increase the hatching prob- ability of their own eggs. Other species use brood parasitism method of laying their eggs in the nests of other birds or host nests [[19]](#_bookmark23).

s Step size.

Entry-wise product of two vectors.

H Heavy-side function.

pa Used to switch between local and global random walks.

e Random number from uniform distribution.

L(s;k) Lévy distribution, used to define the step size of random walk



Objective function ( ), = ( 1, 2 ……. ) Generate initial population of n host nests (*i = 1, 2...n*) **While** (t < Max Generation) or (stop criteria)

Get a cuckoo (say *i)* randomly by Lévy distribution; Evaluate its quality/fitness ;

Choose a nest among n (say j) randomly; Evaluate its quality/fitness ;

**If** ( > )

Replace j by the new solution;

**End**

A fraction of ( ) of worse nests are abandoned and new ones are built at new locations via Lévy flights; Keep the best solutions (or nests with quality solutions); Rank the solutions and find the current best;

**End while**

Post processing

Fig. 1. Lévy based Cuckoo search pseudo code for a global optimisation.

nest represents a solution, and each cuckoo can lay only one egg (thus representing one solution). We can safely make no deference between an egg, a nest or a cuckoo. The aim is to use the new and potentially better solution (cuckoo egg) to replace a bad solution in the nest.

Cuckoo search algorithm is very effective for global optimisa- tion problems since it maintains a balance between local random walk and the global random walk. The balance between local and global random walks is controlled by a switching parameter

*pa* s[0, 1]. The local and global random walks are defined by Eqs.

[(1) and (2)](#_bookmark6), respectively. Their parameters are defined in [Table 1](#_bookmark4).

## xt+1 = xt + as H(pa — e) xt — xt (1)

i i j k

xt+1 = xt + aL(s,k) (2)

i i

The basic steps of the Cuckoo search algorithm based on three

rules can be summarised as pseudo code shown in [Fig. 1](#_bookmark5).

* 1. *Some Cuckoo search algorithm improvements*
     1. *Step size improvements*

This section discusses research studies done to improve the effi- ciency of Cuckoo search, these studies apply different probability distributions to determine the Cuckoo search random walk step sizes.

The first study was done by Zheng and Zhou [[20]](#_bookmark24), they used Gauss distribution instead of original Lévy distribution to deter- mine the random walk step size. When applied to find global min- imum values of 6 mathematics test functions, the Gauss CS performed better than the Lévy CS for all cases. Furthermore, the Gauss and Lévy CS algorithms were used to solve engineering design optimisation problem. The results further confirmed that Gauss CS is better than the Lévy CS in terms of higher convergence rate and the average generation was reduced from 20.15 to 13.95 for Gauss CS.

The rapid growing rate of documentation in the Internet space posed some challenges especially in the documentation retrievals

process. Zaw and Mon [[21]](#_bookmark25), solved this web document clustering by using a Gauss based CS algorithm. The algorithm was tested on 3 clusters and 300 documents. The results confirmed that Gauss CS algorithm outperformed Lévy CS algorithm. More specifically, the convergence rate of Gauss CS and Lévy CS are 120 and 160 iter- ations, respectively. The quality of clustering was determined by a combination of Precision and Recall, called F-measure where high F-measure indicated high accuracy. The Gauss CS algorithm and Lévy CS algorithm produced F-measure of 0.626 and 0.619, respectively.

Ho et al. [[22]](#_bookmark25) proposed CS algorithm using Gaussian and Cauchy distributions and applied them to solve economic emission load dispatch problem with multiple fuel options. The new versions of CS algorithms resulted in fewer parameters, fewer equations and shorter computational processes when compared to Lévy CS. In addition, the Gauss CS performed better than the Cauchy CS algo- rithm. The application of Gauss CS and Cauchy CS to short term hydrothermal scheduling with reservoir volume constraint was done by Nguyen et al. [[23]](#_bookmark25). In this study, however, Lévy CS pro- duced the best results with lowest minimum compared to Gauss and Cauchy CS algorithms. Furthermore, the Gauss CS algorithm average time was 1.47% more than the Lévy CS algorithm average time. While the Cauchy CS algorithm average time was 4.83% more than that of Lévy CS algorithm.

Roy et al. managed to improve CS by using Gamma distribution instead of original Lévy distribution. When tested on 6 mathemat- ical test functions, the Gamma based CS proved to be more accu- rate and efficient than the Lévy CS algorithm [[24]](#_bookmark25). The best performance was recorded for the Ackley test function for 1000

minimum valves of 1.0923exp (—15) and 2.22507exp (—308), iterations where Lévy and Gamma CS algorithms produced average respectively.

* + 1. *Other improvements*

In the original cuckoo search algorithm, switching parameter between local and global random walks is keep constant at 0.25 as Cuckoo search algorithm searches for global minima. There is limited research studies in CS algorithm efficiency improvement by dynamically changing the value of switching parameter. In

[[25]](#_bookmark25) the value of switching parameter was linearly decreased as CS algorithm was searching for global minimum value of the test function. Another CS improvement was achieved by selecting suit- able nest by using sorting function instead on permutation func- tion [[26]](#_bookmark25).

1. Proposed Cuckoo search

In the previous Section [2](#_bookmark3), CS efficiency improvements were accomplished by determining the CS random step size using other probability distribution functions other than the original Lévy dis- tribution. In this section three CS algorithms are proposed based on dynamic increasing switching parameter as the number of CS iter- ations increases. The new CS algorithms are defined as per Eqs. [(3)–](#_bookmark10)

[(5)](#_bookmark10) and their parameters are summarised in [Table 2](#_bookmark8).

Table 2

Dynamic Switching parameter definition.

Parameter Description

paCi Switching parameter for the current iteration.

paMax Maximum value of switching parameter.

Ci Current iteration.

Ti Set total number of iteration.

Exp Mathematical exponential function

Table 3

Different Cuckoo search algorithms based on changing switching parameter.

Table 6

CSLD results.

Cuckoo search

Description

Test Function Global minimum value

Ackley 9.67E-08

CSCo This Cuckoo search uses constant switching parameter. CSLD Cuckoo search using linear decreasing switching parameter. CSLI Cuckoo search using linear increasing switching parameter.

CSEI Cuckoo search using exponential increasing switching parameter.

CSPI Cuckoo search using power increasing switching parameter.

Table 4

Test functions modality and global minimum values.

Function Modality Global minimum value References Ackley Multimodal 0 [[29]](#_bookmark25)

Griewank Multimodal 0 [[27]](#_bookmark25)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Bohachevsky  De Jong | Multimodal  Unimodal | 0  0 | [[14]](#_bookmark21)  [[30]](#_bookmark25) | Test Function | Minimum value |
| Matyas | Unimodal | 0 | [[14]](#_bookmark21) | Ackley | 3.35E-08 |
| Zakharov | Unimodal | 0 | [[14]](#_bookmark21) | Griewank | 3.72E-08 |
| Goldstein-Prices | Multimodal | 3 | [[14]](#_bookmark21) | Bohachevsky | 0 |
| Rosenbrock | Unimodal | 0 | [[31]](#_bookmark26) | De Jong | 7.26E-18 |
| Easom | Unimodal | —1 | [[14]](#_bookmark21) | Matyas | 1.22E-46 |
| Michalewicz | Multimodal | —9.66 | [[32]](#_bookmark27) | Zakharov | 1.17–13 |
|  |  |  |  | Goldstein-Prices | 3 |
|  |  |  |  | Rosenbrock | 1.97E-01 |

Griewank 1.97E-06

Bohachevsky 0

De Jong 6.44E-18

Matyas 6.75E-46

Zakharov 2.90E-12

Goldstein-Prices 3

Rosenbrock 1.31E-01

Easom 1

Michalewicz —7.39

Table 7

CSLI results.

The first proposed CS algorithm uses switching parameter whose value is linearly increased as the number of CS iterations increases. The switching parameter is defined by Eq. [(3)](#_bookmark10).

## paCi = (paMax)\* (Ci/Ti) (3)

Eq. [(4)](#_bookmark11) corresponds to switching parameter that is exponen-

tially increased as iterations increases.

## paCi = (paMax)\* Exp(Ci/Ti) (4)

Eq. [(5)](#_bookmark13) represents switching parameter that is increased in a

power of three as iterations increases.

## paCi = (paMax)\* (Ci/Ti)3 (5)

1. Results and discussions

Easom 1

Michalewicz —7.01

Table 8

CSEI results.

|  |  |
| --- | --- |
| Test Function | Minimum value |
| Ackley | 5.94E-07 |
| Griewank | 2.83E-07 |
| Bohachevsky | 0 |
| De Jong | 5.83E-17 |
| Matyas | 1.04E-33 |
| Zakharov | 1.69E-07 |
| Goldstein-Prices | 3 |
| Rosenbrock | 2.19 |
| Easom | —1 |
| Michalewicz | —8.87 |

* 1. *Simulation setup*

Five Cuckoo search algorithms tabulated in [Table 3](#_bookmark9) were inves- tigated in MATLAB version 7.10.0.499 (R201a). The test environ- ment was FUJITSU laptop with the following specifications; RAM of 3.0 GB, CPU is Intel Celeron [900@2.2](mailto:900@2.2) Ghz and 32 bit windows 7 home Basic operating system.

The number of nests, n was set to 25 for all CS algorithms since any number between 15 and 40 is sufficient for most optimisation

Table 5

CSCo results.

Test Function Global minimum value

Ackley 1.10E-07

Griewank 1.47E-06

Table 10

Table 9

CSPI results.

Test Function Minimum value

Ackley 2.55E-08

Griewank 1.13E-07

Bohachevsky 0

De Jong 5.90E-18

Matyas 6.75E-50

Zakharov 4.00E-16

Goldstein-Prices 3

Rosenbrock 5.58E-01

Easom 1

Michalewicz —6.93

Bohachevsky 0

95% Confidence Interval (CI).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| De Jong  Matyas | 1.31E-17  6.60E-42 | CS | CI | CI Length |
| Zakharov | 9.79E-10 | CSCo | —2.48495 < m < 1.48495 | 3.9699 |
| Goldstein-Prices | 3 | CSLD | —2.401378 < m < 1.349378 | 3.750756 |
| Rosenbrock | 8.12E-01 | CSLI | —2.279265 < m < 1.316665 | 3.59593 |
| Easom | —1 | CSEI | —2.695006 < m < 1.81006 | 4.505066 |
| Michalewicz | —7.82 | CSPI | —2.2280448 < m < 1.3540448 | 3.5820896 |

8

CSCo CSLD CSLI CSEI CSPI

7

6

5

Function value

4

3

2

1

0

20 40 60 80 100 120 140 160 180 200 220

Iteration

Fig. 2. Cuckoo search algorithms average convergences for Ackley function.

0.9

0.8

0.7

Function value

0.6

0.5

0.4

0.3

0.2

0.1

0

50 100 150 200 250 300

CSCo CSLD CSLI CSEI CSPI

Iteration

Fig. 3. Cuckoo search algorithms average convergences for Griewank function.

5

CSCo CSLD CSLI CSEI CSPI

4

3

Function value

2

1

0

0 5 10 15 20

Iteration

Fig. 4. Cuckoo search algorithms average convergences for Bohachevsky function.

45

CSCo CSLD CSLI CSEI CSPI

40

35

30

Function value

25

20

15

10

5

0

0 10 20 30 40 50 60 70 80 90

Iteration

Fig. 5. Cuckoo search algorithms average convergences for De Jong function.

0.1

CSCo CSLD CSLI CSEI CSPI

0.08

Function value

0.06

0.04

0.02

0

-5 0 5 10 15 20 25

Iteration

Fig. 6. Cuckoo search algorithms average convergences for Matyas function.

200

180

160

140

Function value

120

100

80

60

40

20

0

CSCo CSLD CSLI CSEI CSPI

-10 0 10 20 30 40 50 60 70 80 90

Iteration

Fig. 7. Cuckoo search algorithms average convergences for Zakharov function.

problems [[27]](#_bookmark25). The step size scaling factor was set to 0.01 while the total of iterations was 1000 and the value of paMax was 0.25.

Due to the stochastic nature of CS algorithms, the start points for the algorithms are not the same and the paths followed are also

different. To address these differences, each CS algorithm was run twenty times for each of the ten test functions and the results were saved in Microsoft excel file. Using Microsoft Excel, statistical anal- ysis was performed to determine mean, standard deviation. Then

x104

3

CSCo CSLD CSLI CSEI CSPI

2.5

2

Function value

1.5

1

0.5

0

0 5 10 15 20 25 30 35 40 45 50

Iteration

Fig. 8. Cuckoo search algorithms average convergences for Rosenbrock function.

400

CSCo CSLD CSLI CSEI CSPI

350

300

Function value

250

200

150

100

50

0 -5 0 5 10 15 20 25 30

Iteration

Fig. 9. Cuckoo search algorithms average convergences for Goldstein-Prices function.

-0.1

-0.2

-0.3

Function value

-0.4

-0.5

-0.6

-0.7

-0.8

-0.9

-1

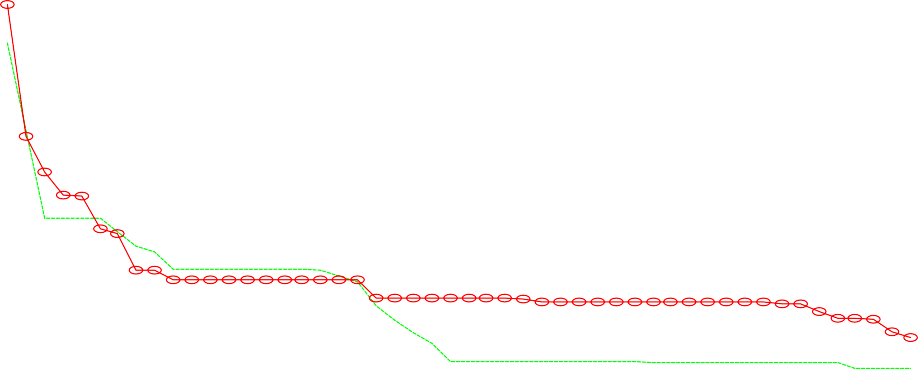
-5 0 5 10 15 20 25 30

CSCo CSLD CSLI CSEI CSPI

Iteration

Fig. 10. Cuckoo search algorithms average convergences for Easom function.

-1



CSCo CSLD CSLI CSEI CSPI

-2

-3

-4

Function value

-5

-6

-7

-8

-9

0 100 200 300 400 500 600 700 800 900 1000

Iteration

Fig. 11. Cuckoo search algorithms average convergences for Michalewicz function.

Table 11

CS convergence.

CS Version Plot Number of leading

CSLI 3, 6 and 10 3

CSEI 4, 9 and 11 3

CSCo 5 and 8 2

CSPI 7 1

CSLD 2 1

95% confidence interval (CI) for each CS algorithm was determined and the CS algorithm with larger CI is deemed as more efficient than the one with smaller CI interval.

* 1. *Characteristics of test functions*

The benchmark test functions used to validate the performance of each CS are tabulated in [Table 4](#_bookmark9). Test functions modality is defined as the number of peaks encountered in the function land- scape. These peaks can negatively impact the optimisation process when the optimisation algorithm gets trapped in the peaks [[28]](#_bookmark25). Some test functions are unimodal while others are multimodal.

* 1. *Simulations results and discussions*

The obtained global minimum values for each of CS algorithm tested are tabulated in [Tables 5–9](#_bookmark14). The corresponding 95% confi- dence intervals were calculated and tabulated in [Table 10](#_bookmark15).

From [Table 10](#_bookmark15), the CSEI has the longest confidence interval (4.505066) as per CI Length column. That means that CSEI is more effective in finding global minimum values of the sample to test functions used. The second most effective algorithm is CSCo with confidence interval of 3.9699, the original CS with constant switch- ing parameter. The least effective algorithm is CSLI with lowest confidence interval of 3.59593.

Cuckoo search convergence results depicted in [Figs. 2–11](#_bookmark16). Some iteration axis have been resized to reveal details.

The convergence results for [Figs. 2–11](#_bookmark16) are summarised in [Table 10](#_bookmark15). This is more of an indication as to how different CS algo- rithms converge to the global minimal point for each test function used (see [Table 11](#_bookmark18)).

1. Conclusions

This paper introduced nature-inspired optimisation algorithms and highlighted that CS algorithm is a more general and efficient algorithm when compared to PSO, DE and SA. Then some studies carried out to improve the CS efficiency were reviewed. It was found that most CS efficiency improvements were based on apply- ing different probability distribution functions (Cauchy, Gauss and Gamma) to determine the random walk step sizes. The contribu- tion of this paper is the investigation of dynamically increasing switching parameter in CS algorithms performance. Three CS algo- rithms using linear, exponential and power increasing switching parameters were developed and tested against the constant and linearly decreasing CS algorithms. The CS using exponential increasing (CSEI) switching parameter was found to be more effi- cient than other CS algorithms.

Acknowledgements

This work is supported by the Institute for Intelligent Systems at the University of Johannesburg in South Africa. We are very thankful of Prof Xin-She Yang for providing demo CS algorithm in the book Nature-Inspired Optimisation Algorithms. We would like to thank also anonymous reviewers for their valuable comments.

References

1. [S. Noureddine, An optimization approach for the satisfiability problem, Appl.](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0005) [Comput. Inform. 11 (1) (2015) 47–59](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0005).
2. [E. Belic, N. Lukac, K. Dezelak, B. Zalik, G. Stumberger, GPU-based online](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0010) [optimization of low voltage distribution network operation, IEEE Trans. Smart](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0010) [Grid 8 (3) (2017) 1460–1468](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0010).
3. [A. Koppel, B.M. Sadler, A. Ribeiro, Proximity without consensus in online](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0015) [multiagent optimization, IEEE Trans. Signal Process. 65 (12) (2017) 3062–](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0015)

[3077](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0015).

1. A. Platonov, Information theory and optimization of analog feedback communication systems, in: 2016 IEEE International Black Sea Conference on Communicatios and Networking, 2016.
2. [M. Tang, L. Gao, H. Pang, J. Huang, L. Sun, Optimizations and economics of](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0025) [crowdsourced mobile streaming, Fog Comput. Netw. (2017) 21–27](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0025).
3. [S. Das, J.R. Doppa, P.P. Pande, K. Chakrabarty, Design-space exploration and](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0030) [optimization of an energy-efficient and reliable 3-D small-world network-on-](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0030) [chip, IEEE Trans. Comp.-Aided Des. Integr. Circ. Syst. 36 (5) (2017) 719–732](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0030).
4. [Z. Yan, J. Fan, J. Wang, A collective neurodynamic approach to constrained](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0035) [global optimization, IEEE Trans. Neural Learn. Syst. 28 (5) (2017) 1206–1215](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0035).
5. [A.R. Parkinson, R.J. Balling, J.D. Hedengren, Optimization Methods for](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0040) [Engineering Design: Applications and Theory, fifth ed., Brigham Young](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0040) [University, 2013](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0040).
6. [X.S. Yang, A new metaheuristic bat-inspired algorithm, in:](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0045) [J.R. Gonzalez,](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0045) [D.A.](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0045) [Pelta,](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0045) [C. Cruz,](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0045) [G. Terrazas,](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0045) [N. Krasnogor (Eds.), Nature Inspired Cooperative](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0045) [Strategies for Optimization (NICSO 2010), vol. 284,](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0045) [Springer, Berlin, 2010, pp.](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0045) [65–74](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0045).
7. [D.P. Rini, S.M. Shamsuddin, S.S. Yuhaniz, Particle swarm optimization:](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0050) [technique, system and challanges, Int. J. Comput. Appl. 14 (1) (2011) 19–27](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0050).
8. [R. Storm, K. Prince, Differential evolution – a simple and efficient heuristic for](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0055) [global optimization over continuous spaces, J. Global Optim. 11 (4) (1997)](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0055) [341–359](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0055).
9. [M. Kefayat, A.L. Ara, S.N. Niaki, A hybrid of ant colony optimization and](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0060) [artificial bee colony algorithm for probabilistic optimial placement and sizing](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0060) [of distributed energy resources, Energy Convers. Manage. 92 (2015) 149–161](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0060).
10. [S. Kirkpatric, C.D. Gelatt, M.P. Vecchi, Optimization by simulated annealing,](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0065) [Science 220 (4598) (1983) 670–680](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0065).
11. [X.-S. Yang, Nature-Inspired Optimization Algorithms, first ed.,](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0070) [Elsevier,](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0070) [London, 2014](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0070).
12. M.I. Solihin, M.F. Zanil, Performance comparison of Cuckoo search and differential evolution algorithm for constrained optimization, in: Intrnational Enginering Research and Innovation Symposium (IRIS), vol. 160(1), 2016, pp. 1–7.
13. M.A. Adnan, M.A. Razzaque, A comparative study of particle swarm optimization and Cuckoo search techniques through problem – specific distance function, in: 2013 International Conference on Information and Communication Technology (ICoICT), Bandung, Indonesia, 2013.
14. [O. Baskan, Determining optimal link capacity expansions in road networks](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0085) [using Cuckoo search algorithm with Levy flights, J. Appl. Math. 2013 (2013) 1–](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0085) [11](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0085).
15. [W. Buaklee, K. Hongesombut, Optimal DG allocation in a smart distribution](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0090) [grid using Cuckoo search algorithm, ECTI Trans. Electr. Eng. Electron. Commun.](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0090) [11 (2) (2013) 16–22](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0090).
16. [X.S. Yang, S. Deb, Engeering optimization by Cuckoo search, Int. J. Math.](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0095) [Modell. Neumeric Opt. 1 (4) (2010) 330–343](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0095).
17. [H. Zheng, Y. Zhou, A novel cuckoo search algorithm based on Gauss](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0100) [distribution, J. Comput. Inf. Syst. 8 (10) (2012) 4193–4200](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0100).
18. [M.M. Zaw, E.E. Mon, Web document clustring using Gauss distribution based](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0105) [cuckoo search clustring algorithm, Int. J. Sci. Eng. Technol. Res. 3 (13) (2014)](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0105) [2945–2949](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0105).
19. [S.D. Ho, V.S. Vo, T.M. Le, T.T. Nguyen, Economic emission load dispatch with](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0110) [multiple fuel optings using cuckoo search algorithm with Gaussian and](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0110) [Cauchy distributions, Int. J. Energy, Inf. Commun. 5 (5) (2014) 39–54](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0110).
20. [T.T. Nguyen, D.N. Vo, B.H. Dinh, Cuckoo search algorithm using different](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0115) [distributions for short term hydrothermal scheduling with reservoir volume](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0115) [constraint, Int. J. Electr. Eng. Inf. 8 (1) (2016) 76–92](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0115).
21. S. Roy, A. Mallick, S.S. Chowdhury, S. Roy, A novel approach on cuckoo search algorithm using Gamma distribution,” in: Second International Conference on Electronics and Communication Systems, 2015.
22. [S.I. Tusiy, N. Shawkat, M.A. Ahmed, B. Panday, N. Sakib, Comparative analysis](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0125) [on improved Cuckoo search algorithm and artificial Bee colony algorithm on](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0125) [continouos optimization problems, Int. J. Adv. Res. Artif. Intell. 4 (2) (2015) 14–](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0125)

[19](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0125).

1. M. Tuba, M. Subotic, N. Stanarevic, Modified Cucko search algorithm for unconstrained optimization problems,” in: Proceedings of the European Computing Conference, 2011.
2. [X.-S. Yang, Test Problems in Optimization, in Engineering Optimization: An](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0135) [Introduction with Metaheuristic Applications, John Wiley & Sons, 2010](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0135).
3. [M. Jamil, X.S. Yang, A literature survey of banchmark functions for global](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0140) [optimization problems, Int. J. Math. Model. Numer. Opt. 4 (2) (2013) 150–194](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0140).
4. [J.M. Dieterich, B. Hartke, Empirical review of standard benchmark functions](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0145) [using evolutionary glabal optimization, Appl. Math. 3 (10) (2012) 1552–1564](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0145).
5. [N.A. Al-Madi, A.T. Khader, De Jong’s sphere model test for a Social-Based](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0150) [Genetic Algorithm (SBGA), Int. J. Comp. Sci. Netw. Security (IJCSNS) 8 (3)](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0150) [(2008) 179–185](http://refhub.elsevier.com/S2210-8327(17)30167-9/h0150).
6. N. Chase, M. Rademacher, E. Goodman, R. Averill and R. Siodhu, 2016. [Online]. Available: [http://www.yumpu.com/en/document/view/46603781/a-](http://www.yumpu.com/en/document/view/46603781/a-benchmark-study-of-optimization-search-algorithms-red-cedar-/7) [benchmark-study-of-optimization-search-algorithms-red-cedar-/7](http://www.yumpu.com/en/document/view/46603781/a-benchmark-study-of-optimization-search-algorithms-red-cedar-/7). (Accessed 04 November 2016).
7. M. Molga, C. Smutnicki, ‘‘Robert Marks,” Robert Marks, 2005. [Online]. Available: [http://www.robertmarks.orgClassespapers](http://www.robertmarks.orgClassespapers/). (Accessed 29 April 2017).