Available online at www.sciencedirect.com
ScienceDirect
SoftwareX 5 (2016) 37–43
SoftwareX
www.elsevier.com/locate/softx

Water cycle algorithm: A detailed standard code

Ali Sadollah[a], Hadi Eskandar[b], Ho Min Lee[a], Do Guen Yoo[a], Joong Hoon Kim[a,*]

[a] School of Civil, Environmental and Architectural Engineering, Korea University, 136-713, Seoul, South Korea
[b] Faculty of Engineering, University of Semnan, Semnan, Iran

Received 22 May 2015; received in revised form 24 December 2015; accepted 21 March 2016

Abstract

Inspired by the observation of the water cycle process and movements of rivers and streams toward the sea, a population-based metaheuristic algorithm, the water cycle algorithm (WCA) has recently been proposed. Lately, an increasing number of WCA applications have appeared and the WCA has been utilized in different optimization fields. This paper provides detailed open source code for the WCA, of which the performance and efficiency has been demonstrated for solving optimization problems. The WCA has an interesting and simple concept and this paper aims to use its source code to provide a step-by-step explanation of the process it follows.

Keywords: Metaheuristic algorithms; Water cycle algorithm; Global optimization

Required metadata

Current code version

C1	Current code version	v1.0
C2	Permanent link to code/repository used of this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-15-00035
C3	Legal Code License	MATLAB Site-License (Korea University, License Number: 1081614)
C4	Code versioning system used	Word Press
C5	Software code languages, tools, and services used	MATLAB R2015a
C6	Compilation requirements, operating environments & dependencies	MATLAB, Windows, Mac OS X, Linux and experimental, Android/iOS support
C7	If available Link to developer documentation/manual	http://www.ali-sadollah.com/water-cycle-algorithm-wca
C8	Support email for questions	ali_sadollah@yahoo.com

1. Introduction

The water cycle process, also known as the hydrological or the H_2O cycle, explains the unceasing movement of water on, above, and below the surface of the earth. It consists of several phases such as evaporation, precipitation, and surface run-off [1]. As we observe in nature, streams flow into rivers and rivers flow into the sea. Finally, all the rivers and/or streams end up in the sea, the most downhill (low-altitude) place in the world [2].

Therefore, similar to a metaheuristic swarm optimization algorithm, this phenomenon lends itself to finding a global optimal solution or a near-optimal solution via effective exploration and exploitation. Inspired by this observation, the water cycle algorithm (WCA) has been developed as a new metaheuristic algorithm [3].

One of the advantages of the WCA is the lower number of insensitive user parameters it requires, which means that the WCA can address a wide range of optimization problems using the fixed user defined parameters.

* Corresponding author. Tel.: +82 02 290 3316.
E-mail address: jaykim@korea.ac.kr (J.H. Kim).

http://dx.doi.org/10.1016/j.softx.2016.03.001

Over the last few years, the WCA has been successfully applied to several varieties of optimization problems such as water resources, civil engineering, and mathematics [4,5].

This paper aims to extend the application of WCA to additional optimization problems by making the source code of the algorithm available. This would assist users to improve or modify the current version of the WCA code. A unified code for the WCA is also offered.

The remainder of this paper is organized as follows. Section 2 presents the motivations and the significance of distributing the unified code of the WCA. The standard WCA along with its detailed processes is given in Section 3. The main source code of the WCA, written in MATLAB, is provided in Section 4. An illustrative example and its link to the WCA are presented in Section 5 to provide a comparison with other optimizers. The impact of the source code of the form in which the WCA is used on research activities is given in Section 6. Finally, a summary of this paper appears in Section 7.

2. Motivation and significance

The source code, which is available online, is capable of addressing various optimization problems arising in many different fields of study. This availability is expected to assist readers to save time in accessing the source code and to provide them with a fair comparison with other optimizers.

Some examples of WCA contributions are finding more optimal values in terms of cost (i.e., cheaper structures) [4,6] and weight (i.e., lighter products) [7].

Detailed explanations regarding the source code appear in Section 4. Lately, the WCA has gained more attention and by making its source code publicly available, the chance of other researchers using this optimizer in their applications is increased. Regarding its realistic application, the open source code of WCA can be implemented in robot path planning problems, because it can act as optimization software for finding the least distance to a destination point by a robot, and it can also be considered as one of the alternatives for the optimization toolbox used in MATLAB.

3. Water cycle algorithm: Idea, motivation, and design

The WCA mimics the flow of rivers and streams toward the sea and was derived by observing the water cycle process. Let us assume that there are some rain or precipitation phenomena. An initial population of design variables (i.e., population of streams) is randomly generated after the raining process. The best individual (i.e., the best stream), classified in terms of having the minimum cost function (for minimization problems), is chosen as the sea [3].

Then, a number of good streams (i.e., cost function values close to the current best record) are chosen as rivers, whereas the remaining streams flow into the rivers and the sea.

Starting the optimization algorithm requires the generation of an initial population representing a matrix of streams of size $N_{pop} \times D$, where D is the dimension. Hence, this matrix, which is generated randomly, is given as (the rows and column represent the population size (N_{pop}) and the number of design variables, D, respectively):

$$\textit{Total population} = \begin{bmatrix} Sea \\ River_1 \\ River_2 \\ River_3 \\ \vdots \\ Stream_{Nsr+1} \\ Stream_{Nsr+2} \\ Stream_{Nsr+3} \\ \vdots \\ Stream_{N_{pop}} \end{bmatrix} = \begin{bmatrix} x_1^1 & x_2^1 & x_3^1 & \cdots & x_D^1 \\ x_1^2 & x_2^2 & x_3^2 & \cdots & x_D^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_1^{N_{pop}} & x_2^{N_{pop}} & x_3^{N_{pop}} & \cdots & x_D^{N_{pop}} \end{bmatrix}. \tag{1}$$

In the first step, N_{pop} streams are created. Then, a number of best individuals N_{sr} (minimum values) are selected as the sea and rivers. The stream which has the minimum value (objective function) among the others is considered as the sea. In fact, N_{sr} is the summation of the number of rivers (which is defined by the user) and a single sea. The rest of the population (N_{stream}) are considered as streams flowing into the rivers or may alternatively flow directly into the sea.

Depending on the magnitude of the flow, each river absorbs water from streams. Hence, the amount of water entering a river and/or the sea varies from stream to stream. In addition, rivers flow to the sea, which is the most downhill location. The designated

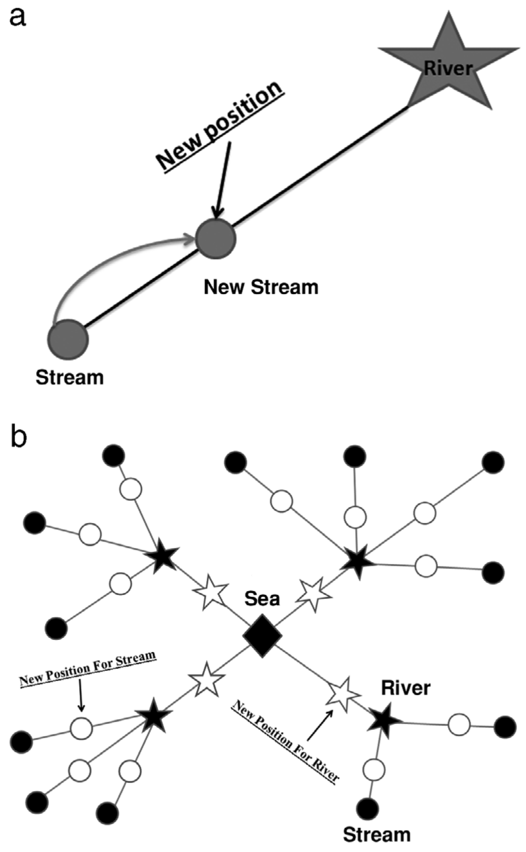

Fig. 1. Schematic illustration of (a) streams flowing into a specific river; (b) the WCA optimization process.

streams for each river and the sea are calculated using the following Eq. [8]:

$$NS_n = round\left\{\left|\frac{Cost_n - Cost_{N_{sr}+1}}{\sum_{n=1}^{N_{sr}} C_n}\right| \times N_{Streams}\right\}, \quad n = 1, 2, 3, \ldots, N_{sr}, \tag{2}$$

where NS_n is the number of streams which flow into the specific rivers and the sea. Fig. 1a shows a schematic view of a stream flowing towards a specific river along their connecting line.

For the exploitation phase of the WCA, new positions for streams and rivers have been suggested as follows [3]:

$$\vec{X}_{Stream}(t+1) = \vec{X}_{Stream}(t) + rand \times C \times (\vec{X}_{Sea}(t) - \vec{X}_{Stream}(t)) \tag{3}$$

$$\vec{X}_{Stream}(t+1) = \vec{X}_{Stream}(t) + rand \times C \times (\vec{X}_{River}(t) - \vec{X}_{Stream}(t)) \tag{4}$$

$$\vec{X}_{River}(t+1) = \vec{X}_{River}(t) + rand \times C \times (\vec{X}_{Sea}(t) - \vec{X}_{River}(t)), \tag{5}$$

where t is an iteration index, $1 < C < 2$, and the best value for C may be chosen as 2, and *rand* is a uniformly distributed random number between zero and one. Eqs. (3) and (4) are for streams which flow into the sea and their corresponding rivers, respectively. If the solution given by a stream is more optimal than that of its connecting river, the positions of the river and stream are exchanged (i.e., the stream becomes a river and the river becomes a stream). A similar exchange can be performed for a river and the sea.

The evaporation process operator is also introduced to avoid premature (immature) convergence to local optima (exploitation phase) [3]. Basically, evaporation causes sea water to evaporate as rivers/streams flow into the sea. This leads to new precipitation. Therefore, we have to check whether the river/stream is sufficiently close to the sea to enable the evaporation process to occur. For that purpose, the following criterion is utilized for the evaporation condition between a river and the sea:

$$\text{if } \left\|\vec{X}^t_{Sea} - \vec{X}^t_{River_j}\right\| < d_{\max} \quad \text{or} \quad rand < 0.1 \quad j = 1, 2, 3, \ldots, N_{sr} - 1$$

Perform raining process by uniform random search,
end

where $d_{\max}$ is a small number close to zero. After evaporation, the raining process is applied and new streams are formed in different locations (similar to mutation in the GAs). Indeed, the evaporation operator is responsible for the exploration phase in the WCA. Uniform random search is used to specify the new locations of the newly formed streams:

A large value for $d_{\max}$ prevents additional searches and small values encourage the search intensity near the sea. Therefore, $d_{\max}$ controls the search intensity near the sea (i.e., best obtained solution). The value of $d_{\max}$ adaptively decreases as follows [8]:

$$d_{\max}(t+1) = d_{\max}(t) - \frac{d_{\max}(t)}{Max.\ Iteration} \qquad t = 1, 2, 3, \ldots, Max\_Iteration. \tag{6}$$

The development of the WCA optimization process is illustrated in Fig. 1b where the circles, stars, and the diamond correspond to streams, rivers, and the sea, respectively. The white (empty) shapes denote the new positions occupied by streams and rivers.

4. Software description

The source code given in this section is written in MATLAB. The main parts of the WCA source code such as the movements of streams, rivers, and the sea, and the evaporation condition have been included in this section. The WCA creates a uniform random initial population. Based on the objective function value (e.g., cost/fitness), the generated population is sorted and individuals are named as the sea, rivers, and streams accordingly. Afterwards, as mentioned in Section 3, the sea and rivers depend on their intensity of flow (objective function value) can absorb streams as can be seen in Eq. (2). The notations *F_best*, *obj_river*, and *obj_stream* correspond to the best objective function obtained so far (i.e., sea), and the objective functions for the rivers and streams, respectively.

The main loop of WCA is as follows. First, we assume streams flow into the sea, and if they find more optimal positions compared with the sea, their positions will be exchanged. These processes have been coded and provided in Lines 1–10.

```
1. for j=1:NS(1)
2.    stream(j,:)=stream(j,:)+2.*rand(1).*(sea-stream(j,:));
3.    stream(j,:)=min(stream(j,:),UB);
4.    stream(j,:)=max(stream(j,:),LB);
5.    obj_stream(j)=objective_function(stream(j,:));
6.       if obj_stream(j)<F_best
7.         new_sea=stream(j,:);stream(j,:)=sea;sea=new_sea;
8.         F_obj=obj_stream(j);obj_stream(j)=F_best;F_best=F_obj;
9.      end
10. end
```

where LB and UB are the lower and upper bounds of a given problem, respectively. Afterward, streams that have been assigned to specific rivers flow to their corresponding rivers (Lines 11–31). In case of improvement in terms of the objective function value (i.e., cost/fitness), the position of a stream and its corresponding river switches (see Lines 17–23). In addition, we should ensure that our solution is updated as the best obtained solution so far (sea). Therefore, Lines 24–28 compare the goodness of the new river with the sea as follows:

```
11. for k=1:Nsr-1
12.    for j=1:NB(k)
13.         stream(j+sum(NS(1:k)),:)=stream(j+sum(NS(1:k)),:)+…
            2.*rand(1,nvars).*(river(k,:)-stream(j+sum(NS(1:k)),:));
14.         stream(j+sum(NS(1:k)),:)=min(stream(j+sum(NS(1:k)),:),UB);
15.         stream(j+sum(NS(1:k)),:)=max(stream(j+sum(NS(1:k)),:),LB);
16.         obj_stream(j+sum(NS(1:k)))=objective_function(stream(j+sum(NS(1:k)),:))
17.         if obj_stream(j+sum(NS(1:k)))<obj_river(k)
18.             new_river=stream(j+sum(NS(1:k)),:);
19.             stream(j+sum(NS(1:k)),:)=river(k,:);
20.             river(k,:)=new_river;
21.             obj_riv=obj_stream(j+sum(NS(1:k)));
22.             obj_stream(j+sum(NS(1:k)))=obj_river(k);
23.             obj_river(k)=obj_riv;
24.                 if obj_river(k)<F_best
25.                     new_sea=river(k,:);river(k,:)=sea;sea=new_sea;
26.                     new_Fbest=obj_river(k);obj_river(k)=F_best;
27.                     F_best=new_Fbest;
28.                 end
29.         end
30.     end
31. end
```

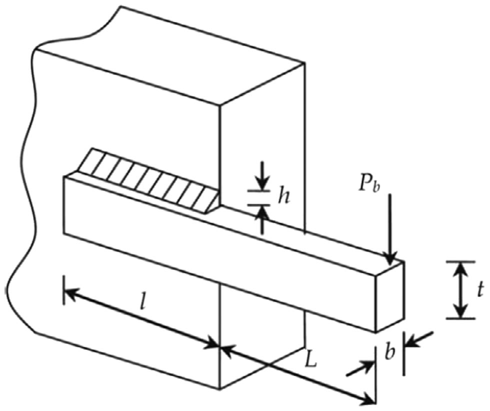

Fig. 2. Schematic view of welded beam problem.

In the last movement phase, rivers flow to the sea as shown in Lines 32–41. If a newly generated river is found to be more optimal than the sea, their roles will be switched (see Lines 37–40):

```
32. for j=1:Nsr-1
33.    river(j,:)=river(j,:)+2.*rand(1,nvars).*(sea-river(j,:));
34.    river(j,:)=min(river(j,:),UB);
35.    river(j,:)=max(river(j,:),LB);
36.    obj_river(j)=objective_function(river(j,:));
37.    if obj_river(j)<F_best
38.        new_sea=river(j,:);river(j,:)=sea;sea=new_sea;
39.        new_Fbest=obj_river(j);obj_river(j)=F_best;F_best=new_Fbest;
40.    end
41. end
```

Lines 42–53 are considered as the evaporation operator for both rivers (see Lines 42–48) and streams (see Line 49–53).

```
42. for k=1:Nsr-1
43.    if ((norm(river(k,:)-sea)<dmax) || rand<0.1)
44.       for j=1:NB(k)
45.          stream(j+sum(NS(1:k)),:)=LB+rand(1,nvars).*(UB-LB);
46.       end
47.    end
48. end
49. for j=1:NS(1)
50.    if ((norm(stream(j,:)-sea)<dmax))
51.          stream(j,:)=LB+rand(1,nvars).*(UB-LB);
52.    end
53. End
```

At Line 53, one iteration of WCA finishes and this process continues until the maximum number of iterations is reached.

5. An illustrative example

An engineering optimization problem, the so-called welded beam design problem, which is often used as a benchmark problem, is considered in this paper [9]. In this problem, a welded beam is designed with the aim of minimizing its cost, subject to constraints on shear stress (τ), bending stress (σ) in the beam, buckling load on the bar (P_b), end deflection of the beam (δ), and side constraints. This requires four design variables as shown in Fig. 2: $h(x_1)$, $l(x_2)$, $t(x_3)$, and $b(x_4)$.

In order to apply the reported problem on WCA, first, objective function which is minimization and all constraints should be defined and coded in separate *m* file in MATLAB. Therefore, lines 54–56 correspond to the cost function of the welded beam design problem as follows:

```
54. function Objective=func(x)
55.    Objective=1.10471*x(1)^2*x(2)+0.04811*x(3)*x(4)*(14+x(2));
56. End
```

Accordingly, for the constraints, Lines 57–66 belong to subjected constraints as given in the following:

```
57. function c=const(x)
58.    R=sqrt((x(2)^2)/4+((x(1)+x(3))/2)^2);
59.    P=6000;L=14;E=30e6;G=12e6;
60.     Px=((4.013*E*sqrt((x(3)^2*x(4)^6)/36))/(L^2))*(1-x(3)/(2*L))*sqrt(E/(4*G)));
61.    tap=P/(sqrt(2)*x(1)*x(2));Q=P*(L+(x(2)/2));
62.    J=2*(sqrt(2)*x(1)*x(2)*((x(2)^2/12)+((x(1)+x(3))/2)^2));tapp=(Q*R)/J;
63.    ta=sqrt(tap^2+((2*tap*tapp*x(2))/(2*R))+tapp^2);
64.    sigma=(6*P*L)/(x(4)*(x(3)^2));delta=(4*P*(L^3))/(E*(x(3)^3)*x(4));
65.    c=[ta-13600;sigma-30000;x(1)-x(4);0.125-x(1);delta-0.25;P-Px;
        0.10471*x(1)^2+0.04811*x(3)*x(4)*(14+x(2))-5];
66. end
```

For constraint handling approach, one can use any type of strategy for satisfying the subjected constraints such as penalty function technique. However, for this problem authors used the direct method (i.e., 4 simple rules) in this paper [4]. Talking about the connection among objective and constraints functions, Line 67 shows a general command for lunching the WCA source code. Therefore, for the considered problem, as can be seen in Line 68, WCA along with its initial parameters is prepared for optimizing our problem of interest (NFEs stand for the maximum of functional evaluations).

```
67. [X,Cost]=WCA(@func,@const,LB,UB,D,Npop,N_sr,d_max,NFEs)
68. [X,Cost]=WCA(@func,@const,[0.1,0.1,0.1,0.1],[2,10,10,2],4,50,4,1e-5,30000)
```

where *X* and *Cost* (considered as the outputs) are optimum values for design variables and the optimized cost obtained after optimization using the WCA, respectively. The optimization methods previously applied to this problem include genetic algorithms (GAs) [10], a cultural algorithm with evolutionary process (CAEP) [11], hybrid Nelder–Mead simplex search and particle swarm optimization (NM–PSO) [12], modified GA (MGAs) [13], society and civilization (SC) [14], differential evolution (DE) [15], and PSO-DE [16]. The WCA, based on the mentioned source code in Section 4, was applied to solve this design problem for 30 independent runs. The results of the comparison of the statistical optimization are listed in Table 1.

Among the previously reported results, the best solution was obtained using the WCA with an objective function value of $f(x) = 1.7235$ with fewer NFEs (i.e., 30,000). The statistical optimization results obtained by the WCA were superior to the results obtained by the other considered algorithms when considering that the WCA required a smaller number of NFEs as shown in Table 1.

6. Impact

The ability to produce better optimal solutions and the advantages of the WCA (simple in terms of coding and implementation) make it an attractive optimization method for researchers who work in optimization related areas. Recently, Haddad et al. [17] utilized the WCA to optimize the operation of reservoir systems. The optimization results they obtained demonstrated the high efficiency and reliability of the WCA in solving reservoir operation problems. Lenin et al. [18] used the WCA for detecting optimum reactive power dispatch problems. They claimed to have applied the WCA to standard IEEE 30 bus test systems and their simulation results clearly show the superior performance of the WCA in decreasing real power loss.

Jabbar and Zainudin [19] applied the WCA to attribute reduction problems in rough set theory. Based on their findings, it has been shown that the WCA performed equally well or even better than other methods for detecting optimal attribute selection.

Guney and Basbug [20] proposed an improved version of the WCA, named the quantized WCA (QWCA) and applied it to solve antenna array pattern synthesis. Yet another improved version of the WCA was recently developed by Sadollah et al. [8], who proposed a modification for WCA using different evaporation rates for rivers. Evaporation rate-based WCA (ER-WCA) was applied to a number of unconstrained and constrained optimization problems and compared with the standard WCA [8]. Moreover, the WCA has been applied to the sizing optimization of sandwich panels with prismatic cores with the goal of designing ultra-light weight sandwich panels [7]. The WCA was also applied to optimize the cost design of water distribution systems [21].

Ramzanpour and Abdi [22,23] applied the WCA for solving economic load dispatch problems among power plants considering different concepts. Lately, Ashouri and Hosseini [24] have utilized the WCA for dynamic economic load dispatch problems in the operation of power systems. Baghipour et al. [25] used the WCA to find the optimal number, location, and size of multiple types of distributed generation units in a distribution system. The attained optimization results demonstrate successful application of the WCA for distributed generation units.

In mathematics, approximate solutions of ordinary differential equations (ODEs) were obtained by using the WCA to efficiently solve nonlinear ODEs along with other optimization methods [5]. The approximate solutions that were obtained using the WCA were superior to the results of other optimization methods [5].

In civil engineering, the WCA has been implemented to reduce the cost of truss structures. Comparative studies showed the superiority of the WCA over other considered methods [4,6]. More recently, Sadollah et al. proposed a multi-objective version of WCA (MOWCA) for solving unconstrained and constrained multi-objective optimization problems [26,27]. Various performance

Table 1
Comparison of the statistical optimization results obtained from different optimizers for the welded beam design problem.

Methods	Worst solution	Average solution	Best solution	SD[a]	NFEs
GAs [10]	1.9934	1.7926	1.7282	7.47E–02	80,000
CAEP [11]	3.1797	1.9718	1.7248	4.43E–01	50,020
NM–PSO [12]	1.7333	1.7263	1.7247	3.50E–03	80,000
MGA [13]	1.9950	1.9190	1.8245	5.37E–02	N/A[b]
SC [14]	6.3996	3.0025	2.3854	9.60E–01	33,095
DE [15]	1.8241	1.7681	1.7334	2.21E–02	204,800
PSO-DE [16]	1.7248	1.7248	1.7248	6.70E–16	66,600
WCA	1.8133	1.7337	1.7235	2.80E–02	30,000

[a] Standard deviation.
[b] Not available.

metrics were used for evaluating the efficiency of MOWCA against other optimizers. The optimization results obtained indicated that MOWCA outperformed the other state-of-the-art metaheuristic methods.

7. Conclusions

This paper provides and considers the open source code of the water cycle algorithm (WCA) in detail. The original principle of WCA was inspired by nature and the water cycle process. In addition, this paper also presents explanations of the standard WCA and its applications. The provided source code is easy to implement, is based on a simple concept, and is efficient for solving a wide range of optimization problems.

Acknowledgment

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIP) (NRF-2013R1A2A1A01013886).

References

[1] Strahler AN. Geol Soc Am Bull 1952;63:923–38.
[2] David S. The water cycle, illustrations by John Yates. New York: Thomson Learning; 1993.
[3] Eskandar H, Sadollah A, Bahreininejad A, Hamdi M. Comput Struct 2012;110–111:151–66.
[4] Sadollah A, Eskandar H, Bahreininejad A, Kim JH. Comput Struct 2015;149:1–16.
[5] Sadollah A, Eskandar H, Yoo DG, Kim JH. Eng Appl Artif Intel 2015;40:117–32.
[6] Eskandar H, Sadollah A, Bahreininejad A. Int J Optim Civil Eng 2013;3(1):115–29.
[7] Sadollah A, Eskandar H, Kim JH, Yoo DG. Sizing optimization of sandwich panels having prismatic core using water cycle algorithm. In: 4th global congress on intelligent systems 2013 (GCIS 2013). Hong Kong: IEEE; 2013. p. 325–8.
[8] Sadollah A, Eskandar H, Bahreininejad A, Kim JH. Appl Soft Comput 2015;30:58–71.
[9] Coello CAC. Comput Ind 2000;41:113–27.
[10] Coello CAC, Mezura Montes E. Adv Eng Inf 2002;16:193–203.
[11] Coello CAC, Becerra RL. Eng Optim 2004;36:219–36.
[12] Zahara E, Kao YT. Expert Syst Appl 2009;36:3880–6.
[13] Coello CAC. Civ Eng Environ Syst 2000;17:319–46.
[14] Ray T, Liew KM. IEEE Trans Evol Comput 2003;7:386–96.
[15] Lampinen J. IEEE Trans Evol Comput 2002;1468–73.
[16] Liu H, Cai Z, Wang Y. Appl Soft Comput 2010;10:629–40.
[17] Haddad O, Moravej M, Loáiciga H. J Irrig Drain Eng 2015;141(5):04014064. http://dx.doi.org/10.1061/(ASCE)IR.1943-4774.0000832.
[18] Lenin K, Ravindranath Reddy B, Surya Kalavathi M. J Eng Technol Res 2014;2(2):1–11.
[19] Jabbar A, Zainudin S. J Theor Appl Inform Technol 2014;61(1):107–17.
[20] Guney K, Basbug S. Int J RF Microw C E 2014;25(1):21–9. http://dx.doi.org/10.1002/mmce.20819.
[21] Sadollah A, Yoo DG, Yazdi J, Kim JH, Choi Y. In: 11th int. conf. hydroinformatics, (HIC 2014), CUNY Academic Works, New York, USA: New York City; 2014.
[22] Ramzanpour M, Abdi H. Int J Smart Elec Eng 2013;2(4):201–8.
[23] Ramzanpour M, Abdi H. J Adv Comput Res 2014;5(3):69–84.
[24] Ashouri M, Hosseini SM. Int J Inform Eng Electron Bus (IJIEEB) 2014;6(4):12–9. http://dx.doi.org/10.5815/ijieeb.
[25] Baghipour R, Hosseini SM, Boor Z. Int J Mechatronics Electr Comput Technol (IJMEC) 2014;4(11):430–54.
[26] Sadollah A, Eskandar H, Kim JH, Bahreininejad A. Soft Comput. http://dx.doi.org/10.1007/s00500-014-1424-4.
[27] Sadollah A, Eskandar H, Kim JH. Appl Soft Comput 2015;27:279–98.