Engineering Optimization
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/geno20

A cuckoo search algorithm by Lévy flights for solving reliability redundancy allocation problems

Ehsan Valian[a] & Elham Valian[b]

[a] Faculty of Electrical and Computer Engineering, University of Sistan and Baluchestan, Iran

[b] Department of Mathematics, Islamic Azad University of Kermanshah Branch, Iran

Published online: 03 Dec 2012.

To cite this article: Ehsan Valian & Elham Valian (2013) A cuckoo search algorithm by Lévy flights for solving reliability redundancy allocation problems, Engineering Optimization, 45:11, 1273-1286, DOI: 10.1080/0305215X.2012.729055

To link to this article: http://dx.doi.org/10.1080/0305215X.2012.729055

Engineering Optimization, 2013
Vol. 45, No. 11, 1273–1286, http://dx.doi.org/10.1080/0305215X.2012.729055

A cuckoo search algorithm by Lévy flights for solving reliability redundancy allocation problems

Ehsan Valian[a]* and Elham Valian[b]

[a]*Faculty of Electrical and Computer Engineering, University of Sistan and Baluchestan, Iran;*
[b]*Department of Mathematics, Islamic Azad University of Kermanshah Branch, Iran*

(*Received 27 April 2012; final version received 20 August 2012*)

A new metaheuristic optimization algorithm, called cuckoo search (CS), was recently developed by Yang and Deb (2009, 2010). This article uses CS and Lévy flights to solve the reliability redundancy allocation problem. The redundancy allocation problem involves setting reliability objectives for components or subsystems in order to meet the resource consumption constraint, *e.g.* the total cost. The difficulties facing the redundancy allocation problem are to maintain feasibility with respect to three nonlinear constraints, namely, cost, weight and volume-related constraints. The redundancy allocation problems have been studied in the literature for decades, usually using mathematical programming or metaheuristic optimization algorithms. The performance of the algorithm is tested on five well-known reliability redundancy allocation problems and is compared with several well-known methods. Simulation results demonstrate that the optimal solutions obtained by CS are better than the best solutions obtained by other methods.

Keywords: cuckoo search algorithm; Lévy flight; reliability redundancy allocation problem

1. Introduction

System reliability optimization plays a vital role in real-world applications and has been studied for decades (Hikita *et al.* 1992, Hsieh *et al.* 1998, Gen and Kim 1999, Chen 2006, Salazar *et al.* 2006, Yin *et al.* 2007, Ramirez-Marquez 2008, Coelho 2009, Wu *et al.* 2010, Zou *et al.* 2010, 2011, Kanagaraj and Jawahar 2011). In order to be more competitive in the market, many designers have worked on improving the reliability of manufacturing systems or product components. Two main ways are usually used to attain higher system reliability. The first approach is to increase the reliability of system components. This may improve the system reliability to some degree; however, the required reliability enhancement may be never achievable even though the most currently reliable elements are used. The second approach is to use redundant components in different subsystems. Using this method increases cost, weight and volume. This approach is named the reliability redundancy allocation problem (Kuo and Prasad 2000).

Industrial engineering has been the source of many complex systems. A design engineer often tries to attain the highest reliability for such systems. The reliability problem is subject to several resource constraints such as cost, weight and volume, and can usually be formulated as a nonlinear programming problem.

*Corresponding author. Email: ehsanvalian@gmail.com

In order to cope with optimization problems arising in system reliability, important contributions have been made since 1970 (Prasad and Kuo 2000). To solve a category of reliability optimization problems with multiple-choice constraints, Chern and Jan (1986) developed a two-phase solution method. They presented a general model that can be stated as the problem of finding the optimum number of redundancies maximizing the system reliability. Prasad and Kuo (2000) offered a search method (P and K-algorithm) based on lexicographic order, and an upper bound on the objective function for solving redundancy allocation problems in coherent systems. The main advantages of the P and K-algorithm are its simplicity and applicability to a wide range of complex optimization problems arising in system reliability design. A penalty-guided artificial immune algorithm to solve mixed-integer reliability design problems was proposed in Chen (2006). To efficiently find the feasible optimal/near optimal solution, it can search over promising feasible and infeasible regions. Gen and Yun (2006) employed a soft computing approach to solving a variety of reliability optimization problems. To prevent the early convergence situation of its solution, this method combined rough search and local search techniques. Moreover, several optimization algorithms based on swarm intelligence, such as the particle swarm optimization algorithm (Coelho 2009, Elegbede 2005, Yin *et al.* 2007, Wu *et al.* 2010), genetic algorithm (Yokota *et al.* 1996, Gen and Kim 1999, Marseguerra *et al.* 2004, Painton and Campbell 1995), evolutionary algorithm (Aponte and Sanseverino 2007, Ramirez-Marquez 2008, Salazar *et al.* 2006), ant colony algorithm (Meziane *et al.* 2005) and harmony search algorithms (HS) (Zou *et al.* 2010, 2011) have been employed to solve reliability problems. Kuo (2007) reviewed recent advances in optimal reliability redundancy allocation problems and summarized several techniques.

The cuckoo search (CS) algorithm, a new metaheuristic approach, was developed recently by Yang and Deb (2009, 2010). The optimal solutions obtained by CS are far better than the best solutions obtained by an efficient particle swarm optimizer (Yang and Deb 2010). The metaheuristic optimization approach employing penalty-guided-based CS for the reliability redundancy allocation problems is proposed in this article.

The article is organized as follows. In Section 2, the procedure of the CS algorithm is briefly presented. In Section 3, four reliability redundancy allocation problems as well as a large-scale reliability redundancy allocation problem are introduced. In Section 4, a large number of numerical experiments are carried out to test the performance and effectiveness of the CS algorithm in solving complex reliability redundancy allocation problems. The article ends with some conclusions in Section 5.

2. Cuckoo search algorithm

In order to describe the CS clearly, the interesting breeding behaviour of certain cuckoo species is briefly reviewed. Then, the basic ideas and steps of the proposed algorithm will be outlined.

2.1. *Cuckoo breeding behaviour*

CS is an optimization algorithm developed by Xin-She Yang and Suash Deb in 2009. It was inspired by the brood parasitism of some cuckoo species by laying their eggs in the nests of host birds of other species. Some host birds can engage in direct conflict with the intruding cuckoos. For example, if a host bird discovers the eggs are not their own, it will either throw these alien eggs away or simply abandon its nest and build a new nest elsewhere. Some cuckoo species such as the New World brood-parasitic Tapera have evolved in such a way that female parasitic cuckoos are often very specialized in the mimicry in colours and pattern of the eggs of a few chosen host species .This reduces the probability of their eggs being abandoned and thus increases their reproductivity (Payne *et al.* 2005).

Furthermore, the timing of egg-laying of some species is also amazing. Parasitic cuckoos often choose a nest where the host bird has just laid its own eggs. In general, the cuckoo eggs hatch slightly earlier than the host eggs. Once the first cuckoo chick is hatched, the first instinctive action it will take is to evict the host eggs by blindly propelling the eggs out of the nest, which increases the cuckoo chick's share of food provided by the host bird (Payne *et al.* 2005). Studies also show that a cuckoo chick can mimic the call of host chicks to gain access to more feeding opportunities.

CS idealizes such breeding behaviour, and thus can be applied to various optimization problems. In addition, Yang and Deb (2009, 2010) discovered that the random-walk style search is better performed by Lévy flights rather than a simple random walk.

2.2. *Cuckoo search*

Each egg in a nest represents a solution, and a cuckoo egg represents a new solution. The aim is to use the new and potentially better solutions (cuckoos) to replace a not-so-good solution in the nests. In the simplest form, each nest has one egg. The algorithm can be extended to more complicated cases in which each nest has multiple eggs representing a set of solutions. CS is based on three idealized rules:

- Each cuckoo lays one egg at a time, and dumps it in a randomly chosen nest.
- The best nests with a high quality of eggs (solutions) will carry over to the next generations.
- The number of available host nests is fixed, and a host can discover an alien egg with probability $p_a \in [0, 1]$. In this case, the host bird can either throw the egg away or abandon the nest so as to build a completely new nest in a new location (Yang and Deb 2009).

For simplicity, this last assumption can be approximated by a fraction (P_a) of the n nests being replaced by new nests (with new random solutions at new locations). For a maximization problem, the quality or fitness of a solution can simply be proportional to the objective function. Other forms of fitness can be defined in a similar way to the fitness function in genetic algorithms (Yang and Deb 2009).

Based on these three rules, the basic steps of the CS can be summarized as the pseudo-code, as follows (Yang and Deb 2009):

```
begin
    Objective function $f(x), x = (x_1, \dots, x_d)^T$
    Generate initial population of
      $n$ host nests $x_i (i = 1, 2, \dots, n)$
    While ($t <$ MaxGeneration) or (stop criterion)
      Get a cuckoo randomly by Lévy flights
        evaluate its quality/fitness $F_i$
      Choose a nest among $n$ (say, $j$) randomly
      if ($F_i > F_j$),
        replace $j$ by the new solution;
      end if
      A fraction ($p_a$) of worse nests
            is abandoned and new ones are built;
        Keep the best solutions
           (or nests with quality solutions);
        Rank the solutions and find the current best
    end while
    Postprocess results and visualization
end
```

When generating new solutions $x(t+1)$ for, say cuckoo i, a Lévy flight is performed:

$$x_i(t+1) = x_i(t) + \alpha \oplus \text{ Lévy } (\lambda) \tag{1}$$

where $\alpha > 0$ is the step size, which should be related to the scale of the problem of interest. In most cases, $\alpha = O(1)$ could be used. The above equation is essentially the stochastic equation for random walk. The product $\oplus$ means entry-wise multiplications. This entry-wise product is similar to those used in PSO, but here the random walk via Lévy flight is more efficient in exploring the search space as its step length is much longer in the long run (Yang and Deb 2009). A Lévy flight is a random walk in which the step-lengths are distributed according to a heavy-tailed probability distribution. Specifically, the distribution used is a power law of the form:

$$\text{Lévy } u = t^{-\lambda}, 1 < \lambda \leq 3 \tag{2}$$

which has an infinite variance. Here, the consecutive jumps/steps of a cuckoo essentially form a random walk process which obeys a power-law step-length distribution with a heavy tail. It is worth pointing out that, in the real world, if a cuckoo's egg is very similar to a host's eggs, then this cuckoo's egg is less likely to be discovered, thus the fitness should be related to the difference in solutions. Therefore, it is a good idea to do a random walk in a biased way with some random step sizes (Yang and Deb 2009).

3. Case studies: reliability redundancy allocation problems

To evaluate the performance of the proposed approach on reliability redundancy allocation problems, five case studies are considered. They are a complex (bridge) system, a series system, a series–parallel system, an overspeed protection system and large-scale systems.

3.1. *Case study 1: A complex (bridge) system*

This case study (Hikita *et al.* 1992, Hsieh *et al.* 1998, Chen 2006, Coelho 2009, Wu *et al.* 2010, Zou *et al.* 2010, 2011, Kanagaraj and Jawahar 2011) is a nonlinear mixed-integer programming problem for a complex (bridge) system with five subsystems, as shown in Figure 1.

The problem is formulated as follows:

$$\begin{aligned} \text{Max} \quad f(r,n) = {} & R_1R_2 + R_3R_4 + R_1R_4R_5 + R_2R_3R_5 \\ & - R_1R_2R_3R_4 - R_1R_2R_3R_5 - R_1R_2R_4R_5 \\ & - R_1R_3R_4R_5 - R_2R_3R_4R_5 + 2R_1R_2R_3R_4R_5 \end{aligned}$$

s.t.

$$g_1(r,n) = \sum_{i=1}^{m} w_i v_i^2 n_i^2 - V \leq 0$$

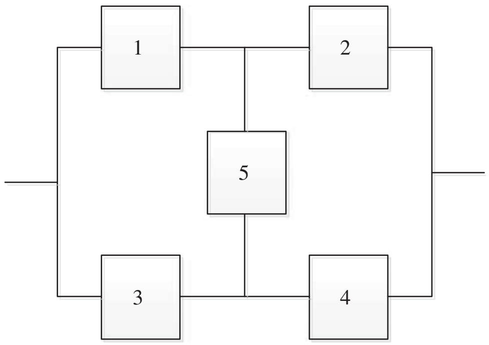

Figure 1. Schematic diagram of the complex (bridge) system.

Table 1. Data used in complex (bridge) and series systems.

i	$10^5\alpha_i$	β_i	$w_i v_i^2$	w_i	V	C	W
1	2.330	1.5	1	7	110	175	200
2	1.450	1.5	2	8			
3	0.541	1.5	3	8			
4	8.050	1.5	4	6			
5	1.950	1.5	2	9			

$$g_2(r,n) = \sum_{i=1}^{m} \alpha_i \left(-\frac{1000}{\ln(r_i)}\right)^{\beta_i} [n_i + \exp(0.25n_i)] - C \le 0$$

$$g_3(r,n) = \sum_{i=1}^{m} w_i n_i \exp(0.25n_i) - W \le 0$$

$$0 \le r_i \le 1, \quad n_i \in Z^+, \quad 1 \le i \le m$$

where m is the number of subsystems in the system. n_i, r_i and $q_i = 1 - r_i$ are the number of components, the reliability of each component, and the failure probability of each component in subsystem i, respectively. $R_i(n_i) = (1 - q_i)^{n_i}$ is the reliability of subsystem i, while $f(r,n)$ is the system reliability. For each component in subsystem i, w_i, v_i and c_i are the weight, volume and cost, respectively. V, C and W refer to the upper limit on the sum of the subsystems' products of volume and weight, the upper limit on the cost of the system, and the upper limit on the weight of the system, respectively. The parameters α_i and β_i are physical features of system components. Constraint $g_1(r,n)$ is a combination of weight, redundancy allocation and volume. $g_2(r,n)$ and $g_3(r,n)$ are constraints on the cost and weight. The input parameters of the complex (bridge) system are given in Table 1.

3.2. *Case study 2: A series system*

This case study (Kuo *et al.* 1978, Xu *et al.* 1990, Hikita *et al.* 1992, Hsieh *et al.* 1998, Chen 2006, Wu *et al.* 2010, Kanagaraj and Jawahar 2011) is a nonlinear mixed-integer programming problem for a series system with five subsystems, as shown in Figure 2.

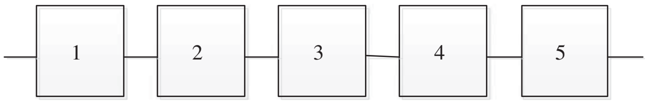

Figure 2. Schematic diagram of a series system.

The problem formulation is as follows:

$$\text{Max} \quad f(r,n) = \prod_{i=1}^{m} R_i(n_i)$$

s.t.

$$g_1(r,n) = \sum_{i=1}^{m} w_i v_i^2 n_i^2 - V \le 0$$

$$g_2(r,n) = \sum_{i=1}^{m} \alpha_i \left(-\frac{1000}{\ln(r_i)}\right)^{\beta_i} [n_i + \exp(0.25 n_i)] - C \le 0$$

$$g_3(r,n) = \sum_{i=1}^{m} w_i n_i \exp(0.25 n_i) - W \le 0$$

$$0 \le r_i \le 1, \quad n_i \in Z^+, \quad 1 \le i \le m$$

This case study has three nonlinear constraints, which are the same as those of the first case study. Also, the input parameters of the series system are the same as those of the complex (bridge) system.

3.3. *Case study 3: A series–parallel system*

This case study (Hikita *et al.* 1992, Hsieh *et al.* 1998, Chen 2006, Wu *et al.* 2010, Kanagaraj and Jawahar 2011) is a nonlinear mixed-integer programming problem for a series–parallel system with five subsystems, as shown in Figure 3.

The reliability function reported by Chen (2006) and Hsieh *et al.* (1998) was $f(r,n) = 1 - (1 - R_1R_2)(1 - (1 - R_3)(1 - R_4)R_5)$; in fact, this is wrong (Wu *et al.* 2010), and the right problem

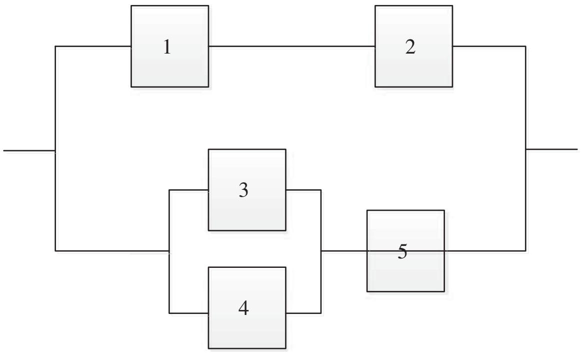

Figure 3. Schematic diagram of a series–parallel system.

formulation is as follows:

$$\begin{aligned}
\text{Max} \quad & f(r,n) = 1 - (1 - R_1R_2)(1 - (R_3 + R_4 - R_3R_4)R_5) \\
\text{s.t.} \quad & \\
& g_1(r,n) = \sum_{i=1}^{m} w_i v_i^2 n_i^2 - V \le 0 \\
& g_2(r,n) = \sum_{i=1}^{m} \alpha_i \left(-\frac{1000}{\ln(r_i)}\right)^{\beta_i} [n_i + \exp(0.25n_i)] - C \le 0 \\
& g_3(r,n) = \sum_{i=1}^{m} w_i n_i \exp(0.25n_i) - W \le 0 \\
& 0 \le r_i \le 1, \quad n_i \in Z^+, \quad 1 \le i \le m
\end{aligned}$$

This case study has the same nonlinear constraints as those of the first case study, but it has different input parameters, given in Table 2.

3.4. *Case study 4: An overspeed system for a gas turbine*

Overspeed detection is continuously provided by the electrical and mechanical systems. When an overspeed occurs, it is necessary to cut off the fuel supply. For this purpose, 4 control valves (V1–V4) must be closed. The control system is modelled as a four-stage series system, as shown in Figure 4. The objective is to determine an optimal level of r_i and n_i at each stage i so that the system reliability is maximized.

Table 2. Data used in series–parallel system.

i	$10^5\alpha_i$	β_i	$w_i v_i^2$	w_i	V	C	W
1	2.500	1.5	2	3.5	180	175	100
2	1.450	1.5	4	4			
3	0.541	1.5	5	4			
4	0.541	1.5	8	3			
5	2.100	1.5	4	4.5			

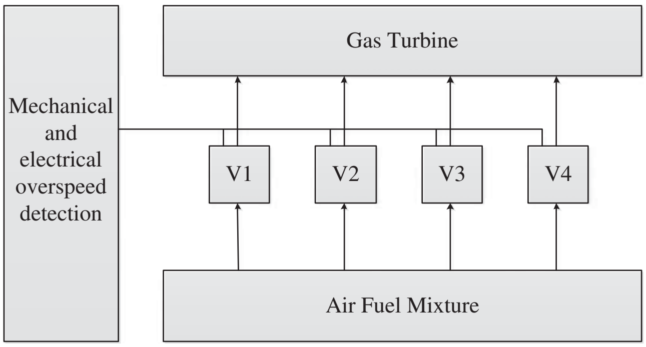

Figure 4. Schematic diagram of an overspeed system for a gas turbine.

Table 3. Data used in overspeed protection system.

i	$10^5\alpha_i$	β_i	v_i	w_i	V	C	W	T
1	1.0	1.5	1	6	250	400	500	1000 h
2	2.3	1.5	2	6				
3	0.3	1.5	3	8				
4	2.3	1.5	2	7				

This case study (Dhingra 1992, Yokota *et al.* 1996, Chen 2006, Coelho 2009, Wu *et al.* 2010, Zou *et al.* 2010, 2011, Kanagaraj and Jawahar 2011) is formulated as follows:

$$\text{Max} \quad f(r,n) = \prod_{i=1}^{m} [1-(1-r_i)^{n_i}]$$

s.t.

$$g_1(r,n) = \sum_{i=1}^{m} v_i n_i^2 - V \le 0$$

$$g_2(r,n) = \sum_{i=1}^{m} C(r_i)[n_i + \exp(0.25n_i)] - C \le 0$$

$$g_3(r,n) = \sum_{i=1}^{m} w_i n_i \exp(0.25n_i) - W \le 0$$

$$0.5 \le r_i \le 1-10^{-6}, \quad r_i \in R^+, \quad 1 \le n_i \le 10, n_i \in Z^+$$

where r_i, n_i, v_i and w_i refer to the reliability of each component, the number of redundant components, the product of weight and volume per element, and the weight of each component, all in stage i. The term $\exp(0.25n_i)$ accounts for the interconnecting hardware. The cost of each component with reliability r_i at subsystem i is given by $C(r_i) = \alpha_i \left(-\frac{T}{\ln(r_i)}\right)^{\beta_i}$. Parameters α_i and β_i are constants representing the physical characteristics of each component at stage i. T is the operating time during which the component must not fail. The input parameters defining the overspeed protection system are given in Table 3.

3.5. *Case study 5: Large-scale system reliability problem*

Prasad and Kuo (2000) developed the P and K-algorithm for solving the large-scale system reliability optimization design problem. Gen and Yun (2006) developed a soft computing approach to solve the same problem. The considered mathematical formulations are as follows:

$$\text{Max} \quad f(r,n) = \prod_{j=1}^{n} [1-(1-r_j)^{x_j}]$$

s.t.

$$g_1(r,n) = \sum_{j=1}^{n} \alpha_j x_j^2 - \left(1+\frac{\theta}{100}\right)\sum_{j=1}^{n} \alpha_j l_j^2 \le 0$$

$$g_2(r,n) = \sum_{j=1}^{n} \beta_j \exp\left(\frac{x_j}{2}\right) - \left(1+\frac{\theta}{100}\right)\sum_{j=1}^{n} \beta_j \exp\left(\frac{l_j}{2}\right) \le 0$$

Table 4. Data used in large-scale system reliability problem.

j	$1-r_j$	α_j	β_j	γ_j	δ_j	j	$1-r_j$	α_j	β_j	γ_j	δ_j
1	0.005	8	4	13	26	26	0.029	8	1	18	35
2	0.026	10	4	16	32	27	0.022	8	3	16	32
3	0.035	10	4	12	23	28	0.017	9	3	15	29
4	0.029	6	3	12	24	29	0.002	10	1	18	35
5	0.032	7	1	13	26	30	0.031	9	2	19	37
6	0.003	10	4	16	31	31	0.021	7	5	15	28
7	0.020	9	2	19	38	32	0.023	9	5	11	22
8	0.018	9	3	15	29	33	0.030	6	3	15	29
9	0.004	7	4	12	23	34	0.026	7	3	14	27
10	0.038	6	4	16	31	35	0.009	6	5	15	29
11	0.028	6	5	14	28	36	0.019	10	5	17	33
12	0.021	10	3	15	30	37	0.005	9	5	19	37
13	0.039	9	1	17	34	38	0.019	10	5	11	22
14	0.013	10	4	20	39	39	0.002	6	2	17	34
15	0.038	7	4	14	28	40	0.015	8	3	17	33
16	0.037	10	2	13	25	41	0.023	10	5	17	33
17	0.021	10	1	15	29	42	0.040	8	3	18	35
18	0.023	8	3	19	38	43	0.012	8	1	18	35
19	0.027	10	5	18	36	44	0.026	6	4	19	38
20	0.028	7	4	13	26	45	0.038	6	4	13	26
21	0.030	6	2	15	30	46	0.015	8	1	19	37
22	0.027	6	2	12	24	47	0.036	7	4	14	28
23	0.018	7	2	20	40	48	0.032	10	2	19	37
24	0.013	8	5	19	38	49	0.038	8	3	15	30
25	0.006	9	5	15	39	50	0.013	10	2	11	22

$$g_3(r,n) = \sum_{j=1}^{n} \gamma_j x_j - \left(1 + \frac{\theta}{100}\right) \sum_{j=1}^{n} \gamma_j l_j \leq 0$$

$$g_4(r,n) = \sum_{j=1}^{n} \sqrt{\delta_j} x_j - \left(1 + \frac{\theta}{100}\right) \sum_{j=1}^{n} \sqrt{\delta_j} l_j \leq 0$$

$$1 \leq x_j \leq 10, \quad j = 1, 2, \ldots, n.$$

Table 4 shows the design data of reliability systems for Case study 5. The component reliabilities are generated from the uniform distribution in [0.95,1.0]. The coefficients α_j, β_j, γ_j and δ_j are generated from uniform distributions in [6,10], [1,5], [11,20] and [21,40], respectively. Here, l_j represents the lower bound of x_j. The tolerance error θ, which implies 33% of the minimum requirement of each resource (l_j), is available for optimization. For the reliability system with n subsystems, the average minimum resource requirement is $\sum_{j=1}^{n} g_{ij}(l_j)$, $(i = 1, \ldots, 4)$, the average value of which is represented by $b_i = \left(1 + \frac{\theta}{100}\right) \sum_{j=1}^{n} g_{ij}(l_j)$, $(i = 1, \ldots, 4)$. In this way, the available system resources are set for a reliability system with 36, 38, 40, 42 and 50 subsystems, respectively, as shown in Table 5.

The best solutions obtained by the above two approaches (Prasad and Kuo 2000, Gen and Yun 2006) are reported in Table 6. Here, VTV represents the variables that have a value of 2 in optimum conditions and the other variables are equal to 1.

4. Experimental results, analysis and discussion

The parameters of the CS algorithm used for reliability redundancy allocation problems are shown in Table 7.

Table 5. Available system resources for each system in Case study 5.

n	i	1	2	3	4
36	b_i	391	257	738	1454
38	b_i	416	278	778	1532
40	b_i	435	289	823	1621
42	b_i	458	306	870	1712
50	b_i	543	352	1040	2048

Table 6. Optimization results of Case study 5 with different dimensions.

n	VTV in optimum	$f(x)$
36	$\{5, 10, 15, 21, 33\}$	0.519976
38	$\{10, 13, 15, 21, 33\}$	0.510989
40	$\{5, 10, 13, 15, 33\}$	0.503292
42	$\{4, 10, 11, 15, 21, 33\}$	0.479664
50	$\{4, 10, 15, 21, 33, 45, 47\}$	0.405390

Table 7. Parameters for CS.

Example	Population size	Number of iterations	Probability p_a	α
Case study 1	10	1000	0.25	1
Case study 2	10	1000	0.25	1
Case study 3	10	1000	0.25	1
Case study 4	10	1000	0.25	1
Case study 5-(36 DIM)	10	2000	0.25	1
Case study 5-(38 DIM)	10	2000	0.25	1
Case study 5-(40 DIM)	10	2500	0.25	1
Case study 5-(42 DIM)	10	2500	0.25	1
Case study 5-(50 DIM)	10	3000	0.25	1

Table 8. Simulation results after 50 runs.

Example	Best	Worst	Median	SD	NFOS	Execution time for 50-iteration (sec)
Case study 1	0.99988964	0.99968396	0.99986946	3.2703e-005	50	71
Case study 2	0.93168239	0.89905558	0.92903027	4.5025e-003	50	38
Case study 3	0.99997665	0.99984484	0.99996550	1.5389e-005	50	40
Case study 4	0.99995468	0.99991922	0.99995336	4.5576e-006	50	40
Case study 5-(36 DIM)	0.51997597	0.49949243	0.51718476	4.2094e-003	50	115
Case study 5-(38 DIM)	0.51098860	0.49467807	0.50801105	2.9530e-003	50	132
Case study 5-(40 DIM)	0.50599242	0.49604469	0.50184162	2.0457e-003	50	162
Case study 5-(42 DIM)	0.47966355	0.46659835	0.47421700	2.7735e-003	50	167
Case study 5-(50 DIM)	0.40695474	0.39205805	0.40347689	3.2395e-003	50	233

Table 8 includes the numerical experiments results of 50 independent runs of the proposed algorithm and all experiments are performed in MATLAB. The PC used is an INTEL32, X2, 3.0 GHz having 4 GB of memory. Moreover, execution times for all case studies are given in Table 8 for 50 iterations. Here, SD represents the standard deviation based on 50 converged objective function values while Median represents the average value of the objective function. NFOS represents the number of feasible optimal solutions found in 50 runs.

Table 9. Case study 1: Bridge (complex) system.

Parameter	Hikita *et al.* (1992)	Hsieh *et al.* (1998)	Chen (2006)	Coelho (2009)	Zou *et al.* (2010)	Wu *et al.* (2010)	CS
$f(r, n)$	0.9997894	0.99987916	0.99988921	0.99988957	0.99988962	0.99988963	0.99988964
n_1	3	3	3	3	3	3	3
n_2	3	3	3	3	3	3	3
n_3	2	3	3	2	2	2	2
n_4	3	3	3	4	4	4	4
n_5	2	1	1	1	1	1	1
r_1	0.814483	0.814090	0.812485	0.826678	0.82883148	0.82868361	0.82809404
r_2	0.821383	0.864614	0.867661	0.857172	0.85836789	0.85802567	0.85800449
r_3	0.896151	0.890291	0.861221	0.914629	0.91334996	0.91364616	0.91416292
r_4	0.713091	0.701190	0.713852	0.648918	0.64779451	0.64803407	0.64790779
r_5	0.814091	0.734731	0.756699	0.715290	0.70178737	0.70227595	0.70456598
MPI (%)	47.597341	8.672625	0.388122	0.063389	0.018120	0.009060	–

Table 10. Case study 2: Series system.

Parameter	Kuo *et al.* (1978)	Xu *et al.* (1990)	Hikita *et al.* (1992)	Hsieh *et al.* (1998)	Chen (2006)	Wu *et al.* (2010)	CS
$f(r, n)$	0.92975	0.931677	0.931363	0.931578	0.931678	0.931680	0.931682
n_1	3	3	3	3	3	3	3
n_2	2	2	2	2	2	2	2
n_3	2	2	2	2	2	2	2
n_4	3	3	3	3	3	3	3
n_5	2	3	3	3	3	3	3
r_1	0.77960	0.77939	0.777143	0.779427	0.779266	0.78037307	0.77941694
r_2	0.80065	0.87183	0.867514	0.869482	0.872513	0.87178343	0.87183328
r_3	0.90227	0.90288	0.896696	0.902674	0.902634	0.90240890	0.90288508
r_4	0.71044	0.71139	0.717739	0.714038	0.710648	0.71147356	0.71139387
r_5	0.85947	0.78779	0.793889	0.786896	0.788406	0.78738760	0.78780371
MPI (%)	2.750748	0.007904	0.465347	0.152583	0.006440	0.003513	–

Tables 9–12 compare the best results obtained in this article for four case studies with those provided by other methods reported in the literature (Kuo *et al.* 1978, Xu *et al.* 1990, Dhingra 1992, Hikita *et al.* 1992, Yokota *et al.* 1996, Hsieh *et al.* 1998, Chen 2006, Coelho 2009, Zou *et al.* 2010, 2011, Wu *et al.* 2010). Clearly, the solutions obtained by the CS algorithm are better than those given by the other methods. An improvement index is required to measure the improvement of the best solutions found by the proposed approach in comparison with those given by the other methods. This index, which has been called maximum possible improvement (MPI) (Ku and Prasad 2000), is as follows:

$$MPI(\%) = \left(\frac{f_{CS} - f_{other}}{1 - f_{other}}\right) \tag{3}$$

where f_{CS} represents the best system reliability obtained by the CS algorithm, while f_{other} denotes the best system reliability obtained by any other method. It should be noted that in high reliability applications, it is often difficult to obtain even very small improvements in reliability.

For the complex (bridge) system, as mentioned in Table 9, the best results reported by Hikita *et al.* (1992), Hsieh *et al.* (1998), Chen (2006), Coelho (2009), Zou *et al.* (2010) and Wu *et al.* (2010), are 0.9997894, 0.99987916, 0.99988921, 0.99988957, 0.99988962 and 0.99988963, respectively. The CS result is 47.597341%, 8.672625%, 0.388122%, 0.063389%, 0.018120% and 0.009060%, respectively, better than the aforementioned algorithms in terms of improvement indices.

Table 11. Case study 3: Series–parallel system.

Parameter	Hikita*et al.* (1992)	Hsieh *et al.* (1998)	Chen (2006)	Wu *et al.* (2010)	CS
$f(r,n)$	0.99996875	0.99997418	0.99997658	0.99997664	0.99997665
n_1	3	2	2	2	2
n_2	3	2	2	2	2
n_3	1	2	2	2	2
n_4	2	2	2	2	2
n_5	3	4	4	4	4
r_1	0.838193	0.785452	0.812485	0.81918526	0.81992709
r_2	0.855065	0.842998	0.843155	0.84366421	0.84526766
r_3	0.878859	0.885333	0.897385	0.89472992	0.89549155
r_4	0.911402	0.917958	0.894516	0.89537628	0.89544069
r_5	0.850355	0.870318	0.870590	0.86912724	0.86831878
MPI (%)	25.280000	9.566228	0.298890	0.042808	–

Table 12. Case study 4: Overspeed system.

Parameter	Dhingra (1992)	Yokota *et al.* (1996)	Chen (2006)	Coelho (2009)	Zou *et al.* (2010)	Wu *et al.* (2010)	CS
$f(r,n)$	0.99961	0.999468	0.999942	0.999953	0.99995467	0.99995467	0.99995468
n_1	6	3	5	5	5	5	5
n_2	6	6	5	6	6	6	5
n_3	3	3	5	4	4	4	4
n_4	5	5	5	5	5	5	6
r_1	0.81604	0.965593	0.903800	0.902231	0.90186194	0.90163164	0.90161460
r_2	0.80309	0.760592	0.874992	0.856325	0.84968407	0.84997020	0.88822337
r_3	0.98364	0.972646	0.919898	0.948145	0.94842696	0.94821828	0.94814103
r_4	0.80373	0.804660	0.890609	0.883156	0.88800590	0.88812885	0.84992090
MPI (%)	88.461523	91.541402	22.413812	4.255389	0.012186	0.000910	–

For the series system, Table 10 shows that the best results reported by Kuo *et al.* (1978), Xu *et al.* (1990), Hikita *et al.* (1992), Hsieh *et al.* (1998), Chen (2006) and Wu *et al.* (2010) are 0.92975, 0.931677, 0.931363, 0.931578, 0.931678 and 0.931680, respectively. The result given by the CS is better than six mentioned results. The improvement indices are 2.750748%, 0.007904%, 0.465347%, 0.152583%, 0.006440% and 0.003513%, respectively.

The same investigation has been done for a series–parallel system. The proposed algorithm results in 25.280000%, 9.566228%, 0.298890% and 0.042808% improvement compared with the best results reported by Hikita (1992), Hsieh (1998), Chen (2006) and Wu*et al.* (2010).

Considering an overspeed system for a gas turbine, Table 12 shows that the best results reported by Dhingra (1992), Yokota *et al.* (1996), Chen (2006), Coelho (2009), Zou *et al.* (2011) and Wu *et al.* (2010), are 0.99961, 0.999468, 0.999942, 0.999953, 0.99995467 and 0.99995467 respectively. The result provided by the CS is better than the above five results. The improvement indices are 88.461538%, 91.541353%, 22.413793%, 4.255319%, 0.012186% and 0.000910% respectively.

For Case study 5 the results of the CS algorithm are shown in Table 13. In addition, Table 14 indicates the comparison between CS and Wu algorithms (Wu et al. 2010) for Case study 5. It can be seen in Table 14 that the best results of CS algorithm and Wu (Wu et al. 2010) for n = 36, 38, 40 and 42 are the same, but for n = 50 the CS provides better results. Also, the worst, median and SD results given by the CS are better than those given by Wu for n = 36, 38, 40, 42 and 50 in Case study 5.

Table 13. Best results of Case study 5 with different dimensions by CS algorithm.

n	VTV in optimum	$f(x)$
36	$\{5, 10, 15, 21, 33\}$	0.51997597
38	$\{10, 13, 15, 21, 33\}$	0.51098860
40	$\{4, 10, 11, 21, 22, 33\}$	0.50599242
42	$\{4, 10, 11, 15, 21, 33\}$	0.47966355
50	$\{4, 10, 15, 21, 33, 42, 45\}$	0.40695474

Table 14. Comparison results for the large-scale system reliability problem.

Example	Algorithm	Best	Worst	Median	SD
P5-(36 DIM)	IPSO(Wu 2010)	**0.51997597**	0.49705368	0.50874873	5.8624e-003
	CS	**0.51997597**	**0.49949243**	**0.51718476**	**4.2094e-003**
P5-(38 DIM)	IPSO(Wu 2010)	**0.51098860**	0.48707780	0.50557752	5.3420e-003
	CS	**0.51098860**	**0.49467807**	**0.50801105**	**2.9530e-003**
P5-(40 DIM)	IPSO(Wu 2010)	**0.50599242**	0.48016779	0.49772492	5.4393e-003
	CS	**0.50599242**	**0.49604469**	**0.50184162**	**2.0457e-003**
P5-(42 DIM)	IPSO(Wu 2010)	**0.47966355**	0.45788422	0.47090324	4.6902e-003
	CS	**0.47966355**	**0.46659835**	**0.47421700**	**2.7735e-003**
P5-(50 DIM)	IPSO(Wu 2010)	0.40657143	0.36682410	0.39338827	6.5878e-003
	CS	**0.40695474**	**0.39205805**	**0.40347689**	**3.2395e-003**

According to the above comparisons, it can be concluded that the CS algorithm outperforms the other methods in the literature to find the best solutions for the given reliability redundancy allocation problems.

5. Conclusion

In this article, the cuckoo search by Lévy flights is proposed to solve five well-known reliability redundancy allocation problems. In these optimization problems, both the redundancy (number of redundant components) and the corresponding reliability of each component in each subsystem under multiple constraints are considered to be decided simultaneously. Experimental results showed that the CS algorithm led to the best solutions in comparison with other methods. Since the CS has a strong convergence and a high exploration capability of solution space, it is a promising algorithm to find optimal solutions for complex optimization problems.

References

Aponte, D.E.S. and Sanseverino, C.M.R., 2007. Solving advanced multi-objective robust designs by means of multiple objective evolutionary algorithms (MOEA): a reliability application. *Reliability Engineering and System Safety*, 92, 697–706.

Chen, T.C., 2006. IAs based approach for reliability redundancy allocation problems. *Applied Mathematics and Computation*, 182, 1556–1567.

Chern, M.S. and Jan, R.H., 1986. Reliability optimization problems with multiple constraints. *IEEE Transactions on Reliability*, 35, 431–436.

Coelho, L.S., 2009. An efficient particle swarm approach for mixed-integer programming in reliability–redundancy optimization applications. *Reliability Engineering and System Safety*, 94, 830–837.

Dhingra, A.K., 1992. Optimal apportionment of reliability & redundancy in series systems under multiple objectives. *IEEE Transactions on Reliability*, 41, 576–582.

Elegbede, C., 2005. Structural reliability assessment based on particles swarm optimization. *Structural Safety*, 27, 171–186.

Gen, M. and Kim, J.R., 1999. GA-based reliability design: state-of-the-art survey, *Computers and Industrial Engineering*, 37–, 151–155.

Gen, M. and Yun, Y.S., 2006. Soft computing approach for reliability optimization: state-of-the-art survey. *Reliability Engineering and System Safety*, 91, 1008–1026.

Hikita, M., Nakagawa, H., and Harihisa, H., 1992. Reliability optimization of systems by a surrogate constraints algorithm. *IEEE Transactions on Reliability*, 41, 473–480.

Hsieh, Y.C., Chen, T.C., and Bricker, D.L., 1998. Genetic algorithm for reliability design problems. *Microelectronics Reliability*, 38, 1599–1605.

Kanagaraj, G. and Jawahar, N., 2011. Simultaneous allocation of reliability & redundancy using minimum total cost of ownership approach. *Journal of Computational Applied Research in Mechanical Engineering*, 1, 1–16.

Kuo, W., 2007. Recent advances in optimal reliability allocation. *IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans*, 37, 143–156.

Kuo, W. and Prasad, V.R., 2000. An annotated overview of system-reliability optimization. *IEEE Transactions on Reliability,* 49, 176–187.

Kuo, W., Hwang, C.L., and Tillman, F.A., 1978. A note on heuristic methods in optimal system reliability. *IEEE Transactions on Reliability R*, 27, 320–324.

Marseguerra, M., Zio, E., and Podofillini, L., 2004. Optimal reliability/availability of uncertain systems via multi-objective genetic algorithms. *IEEE Transactions on Reliability*, 53, 424–434.

Meziane, R., *et al.*, 2005. Reliability optimization using ant colony algorithm under performance and cost constraints. *Electric Power Systems Research*, 76 (1–3), 1–8.

Painton, L., and Campbell, J., 1995. Genetic algorithms in optimization of system reliability. *IEEE Transactions on Reliability*, 44, 172–178.

Payne, R.B., Sorenson, M.D., and Klitz, K., 2005. *The cuckoos*. Oxford University Press.

Prasad, V.R. and Kuo, W., 2000. Reliability optimization of coherent systems. *IEEE Transactions on Reliability*, 49, 323–330.

Ramirez-Marquez, J.E., 2008. Port-of-entry safety via the reliability optimization of container inspection strategy through an evolutionary approach. *Reliability Engineering and System Safety*, 93, 1698–1709.

Salazar, D., Rocco, C.M., and Galvn, B.J., 2006. Optimization of constrained multipleobjective reliability problems using evolutionary algorithms. *Reliability Engineering and System Safety*, 91, 1057–1070.

Wu, P., *et al.*, 2010. An improved particle swarm optimization algorithm for reliability problems. *ISA Transactions*, 50, 71–81.

Xu, Z., Kuo, W., and Lin, H.H., 1990. Optimization limits in improving system reliability. *IEEE Transactions on Reliability*, 39, 51–60.

Yang, X.S. and Deb, S., 2009. Cuckoo search via Lévy flights. *In*: *Proceedings of world congress on nature & biologically inspired computing (NaBIC 2009),* 9–11 December 2009, Coimbatore, India. Piscataway, NJ: IEEE Press, 210–214.

Yang, X.S. and Deb, S., 2010. Engineering optimisation by cuckoo search. *International Journal of Mathematical Modelling and Numerical Optimisation*, 1, 330–343.

Yin, P.Y., *et al.*, 2007. Task allocation for maximizing reliability of a distributed system using hybrid particle swarm optimization. *Journal of Systems and Software*, 80, 724–735.

Yokota, T., Gen, M., and Li, H.H., 1996. Genetic algorithm for nonlinear mixed-integer programming problems and its application. *Computers and Industrial Engineering*, 30, 905–917.

Zou, D., *et al.*, 2010. A novel global harmony search algorithm for reliability problems. *Computers and Industrial Engineering*, 58, 307–316.

Zou, D., *et al.*, 2011. An effective global harmony search algorithm for reliability problems. *Expert Systems with Applications*, 38, 4642–4648.