Availability Optimization of Parallel-Series System by Evolutionary Computation

Mohamed Arezki Mellal
LMSS, Faculty of Engineering Sciences
M'Hamed Bougara University
Boumerdes, Algeria
e-mail: mellal.mohamed@gmail.com,
mellal.mohamed@univ-boumerdes.dz

Enrico Zio
Chair System Science and the Energy Challenge,
Fondation Electricité de France (EDF), CentraleSupélec,
Université Paris-Saclay, France
Politecnico di Milano, Milan, Italy

***Abstract*—This paper addresses the problem of availability optimization of a parallel-series system by using evolutionary techniques. Five techniques are considered, namely the cuckoo optimization algorithm, particle swarm optimization, flower pollination algorithm, differential evolution, and genetic algorithms. The integer values and constraints of cost are handled. The results regarding a system with ten subsystems are compared. It demonstrates that the cuckoo optimization algorithm performs best.**

Keywords-system availability; system cost; parallel-series system; optimization; evolutionary computation

I. INTRODUCTION

The availability of systems is crucial for production or service competitiveness. However, increasing the overall system availability comes with a cost. Recently, many works dealing with the optimization of RAMS+C problems (reliability, availability, maintainability, safety and cost) use evolutionary computation techniques. In [1], [2], the system cost under a fixed availability value has been optimized using the combination of genetic algorithms and Tabu search. The system and cost have been optimized by introducing various techniques, such as artificial immune [3], artificial bee colony [4], genetic algorithms [5], [6], and penalty guided stochastic fractal search [6], [7]. In [8], genetic algorithms have been applied to solve both objectives.

Although evolutionary optimization techniques have been shown to perform well in RAMS+C problems, in practice it can be difficult to decide which one to use. This paper aims to address the availability optimization problem of a parallel-series system. A system with ten subsystems is considered, under the constraints of cost and design. A comparison of five evolutionary computation techniques is offered. Penalty functions are introduced to handle the constraints. The remainder of the paper is organized as follows: Section II describes the problem; Section III presents the numerical case study; Section IV gives a brief description of the implemented evolutionary techniques; Section V gives the results with a discussion; finally, conclusions are given in Section VI.

II. AVAILABILITY OPTIMIZATION OF PARALLEL-SERIES SYSTEM

Design of parallel-series system (Figure 1) can be characterized by the following formulations [1], [2]:
System cost

$$C_S(n,\lambda,\mu)=\sum_{i=1}^{m}\left[\left(\alpha_i\left(\lambda_i\right)^{-\beta_i}+\mu_i mc_i\right)\left(n_i+\exp(n_i/4)\right)\right] \quad (1)$$

where $C_S(\bullet)$ is the total system cost, n_i is the number of identical redundant components in the ith subsystem, λ_i is the failure rate of the components in the ith subsystem, and μ_i is the repair rate of the components in the ith subsystem, m is the number of subsystems in the system. β_i and α_i are parameters representing physical features (shaping and scaling factors, respectively) of each component in subsystem i.

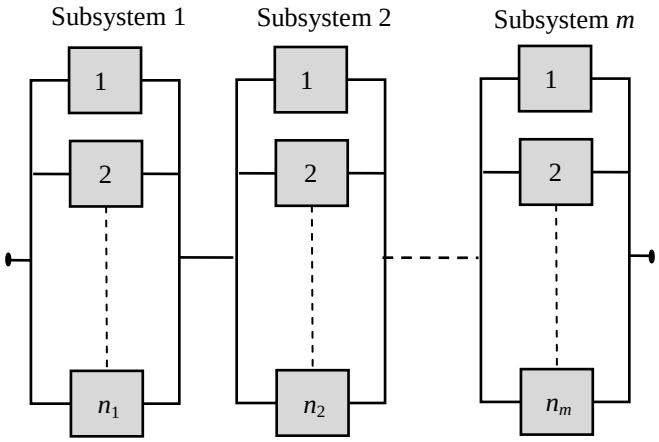

Figure 1. Parallel-series system.

Asymptotic system availability

$$A_S(n,\lambda,\mu)=\prod_{i=1}^{m}\left[1-\left(1-\frac{\mu_i}{\lambda_i+\mu_i}\right)^{n_i}\right] \quad (2)$$

System design configuration constraint of weight

$$\sum_{i=1}^{m} p_i (n_i)^2 \le D_1 \quad (3)$$

where p_i is the product of weight and volume per component in subsystem i and D_1 is the limit of constraint (3).

System design configuration constraint of the product of weight and volume

$$\sum_{i=1}^{m} w_i n_i \exp(n_i / 4) \le D_2 \quad (4)$$

where w_i is the weight of one component in subsystem i and D_2 is the limit of constraint (4).

The aim is to allocate the set of values (n_i, λ_i, μ_i), i=1, 2, …, m which satisfies the redundancy allocation in each subsystem of the system. The search for the optimum of (1) (minimum) and (2) (maximum), constrained by (3) and (4), is performed within:

$$\begin{aligned} & n_i \ge 1; n_i \in \mathbb{Z}^+ \\ & \lambda_i \in \left[\lambda_i^L, \lambda_i^U \right] \subset \Re^+ \\ & \mu_i \in \left[\mu_i^L, \mu_i^U \right] \subset \Re^+ \end{aligned} \quad (5)$$

where λ_i^L, μ_i^L, λ_i^U, μ_i^U are the lower and upper limits of the failure and repair rates, respectively.

III. Numerical Case Study

The system considered consists of ten subsystems connected in parallel-series configuration (Figure 1, with m=10). Table I reports the data of the system.

TABLE I. Data of the System

Subsystem i	α_i (10^{-5})	β_i	mc_i	p_i	w_i
1	1.25	1.5	500	2	6
2	2.70	1.5	500	4	9
3	8.10	1.5	500	3	7
4	4.50	1.5	500	2	6
5	1.90	1.5	500	4	8
6	3.55	1.5	500	2	5
7	2.45	1.5	500	4	3
8	6.30	1.5	500	3	9
9	1.80	1.5	500	2	7
10	5.25	1.5	500	2	5

In [1], [2], the authors fixed the system cost as an objective to be minimized in the case of a system with five subsystems. In the present paper, the goal is to maximize the system availability under the constraints of cost and system configurations. Therefore, according to Eqs. (1)–(5) and the data reported in Table I, the problem of this case study is:

$$\text{Maximize } A_S(n, \lambda, \mu) = \prod_{i=1}^{10} \left[1 - \left(1 - \frac{\mu_i}{\lambda_i + \mu_i} \right)^{n_i} \right] \quad (6)$$

subject to

$$\sum_{i=1}^{10} \left[\left(\alpha_i (\lambda_i)^{-\beta_i} + \mu_i mc_i \right) \left(n_i + \exp(n_i / 4) \right) \right] \le C_S^* \quad (7)$$

$$\sum_{i=1}^{10} p_i (n_i)^2 \le D_1 \quad (8)$$

$$\sum_{i=1}^{10} w_i n_i \exp(n_i / 4) \le D_2 \quad (9)$$

where

$n_i \ge 1$ ($n_i \in \mathbb{Z}^+$)
$\lambda_i \in [10^{-7}, 10^{-3}] \subset \mathbb{R}^+$, $\mu_i \in [32\times10^{-7}, 32\times10^{-3}] \subset \mathbb{R}^+$
$A_S \ge 0.9$
C_S^*=250; D_1=200; D_2=300, in arbitrary units.

IV. Evolutionary Computation Techniques

Five evolutionary computation techniques are applied: genetic algorithms, differential evolution, particle swarm optimization, cuckoo optimization algorithm, and flower pollination algorithm. Table II summarizes their main characteristics.

TABLE II. Characteristics of the Applied Techniques

Methods	Advantages	Limitations
GA	Robust, Flexibility	Premature convergence, Problem encoding, Requires high function evaluations
DE	Accurate, Effective,	No proof of convergence
PSO	Fast execution, Simple calculations	Velocity parameters could be difficult to hand
COA	Proper convergence, Simple parameters to be handled	Hard to handle integer variables
FPA	Good convergence rate	Parameters could be difficult to hand

The redundancy variables are rounded to the nearest integer values, while the constraints are handled using penalty functions [7], introduced in the formulation of the problem as follows:

$$Fitness\ value = A_S(n, \lambda, \mu) + \psi(n, \lambda, \mu) \quad (10)$$

with the penalty function

$$\psi(n, \lambda, \mu) = \sum_{h=1}^{M} \phi_h \cdot \max(0, g_h(n, \lambda, \mu))^2 \quad (11)$$

where ϕ_h is the penalty factor and M is the number of constraints (M=9).

A. Genetic Algorithms (GA)

The GA are based on the principles of the evolution of species [9], [10]. Table III contains the parameters and rules of the implemented GA.

TABLE III. GENETIC ALGORITHM PARAMETERS AND RULES.

Population size	20
Selection technique	Standard roulette
Mutation probability	10^{-3}
Crossover probability	1

B. *Differential Evolution (DE)*

The DE is a population-based algorithm similar to GA, but with more perturbations in the iterations of the population [11]–[13]. Table IV contains the parameters and rules of the implemented DE.

TABLE IV. DIFFERENTIAL EVOLUTION PARAMETERS AND RULES.

Population size	20
Scaling factor	0.2
Crossover rate	0.2

C. *Particle Swarm Optimization (PSO)*

The PSO is based on the behavior of swarms of fishes and birds [14], [15]. Table V contains the parameters and rules of the implemented PSO.

TABLE V. PARTICLE SWARM OPTIMIZATION PARAMETERS.

Swarm size	20
Inertia weight	1
Learning coefficient	2

D. *Cuckoo Optimzization Algorithm (COA)*

The COA is inspired from the processes of reproduction and migration of the cuckoos birds [16], [17]. Table VI contains the parameters and rules of the implemented COA.

TABLE VI. CUCKOO OPTIMIZATION ALGORITHM PARAMETERS.

Number of cuckoos	20
Motion coefficient	5
Radius coefficient	10^{-3}

E. *Flower Pollination Algorithm (FPA)*

The principles of the FPA is based on the pollination process of flowers [18]. The parameters of the implemented FPA are reported in Table VII.

TABLE VII. FLOWER POLLINATION ALGORITHM PARAMETERS.

Population size	20
Probability switch	0.8

The values of the parameters of the above techniques have been fixed based on experience and trial-and-error, in the attempt to get the best performance for each technique. The algorithms have been encoded using MATLAB 2017 and run on a personal computer with the following characteristics: Intel Core I3 of 2.53 GHz with 4 GB of RAM. The problem involves 30 decision variables, including 10 integer values.

V. RESULTS AND DISCUSSION

Table VIII summarizes the worst, average and best values of A_s found by each algorithm, whereas Table IX reports the optimal solutions, system cost C_s, standard deviation (SD) after 10 independent runs, number of function evaluations (NFE) required to find the best solution, and the CPU time.

TABLE VIII. RESULTS.

Method	Worst	Average	Best
COA	0.9697	0.9734	**0.9765**
GA	0.9053	0.9255	0.9589
FPA	0.9011	0.9267	0.9644
DE	0.9015	0.9350	0.9654
PSO	0.9240	0.9424	0.9649

From Table VIII, it can be observed that the best values of A_s found by each algorithm are: PSO (0.9649), DE (0.9654), FPA (0.9644), GA (0.9589), and COA (0.9765). Thus, the maximum A_s has been found by the COA (A_s=0.9765), whereas the minimum value 0.9589 has been provided by the GA. Furthermore, the worst and average values found by the COA are better. Figure 2 illustrates the best values of A_s found by each algorithm.

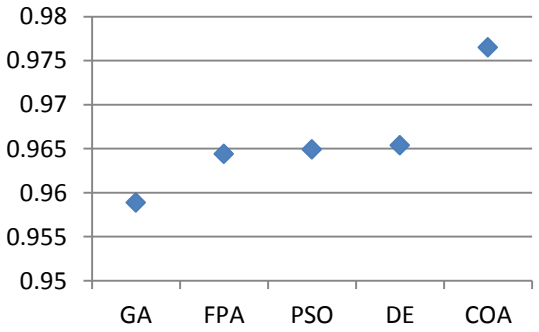

Figure 2. Best system availability provided by each evolutionary computation technique.

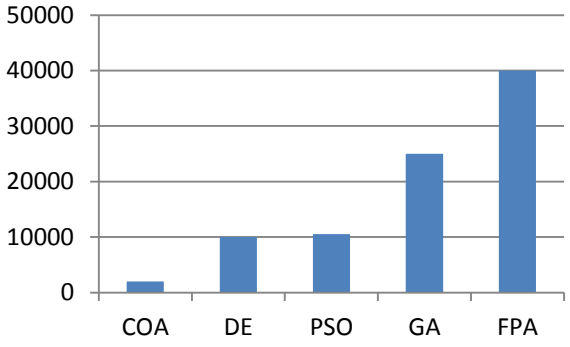

Figure 3. NFE required by each evolutionary computation technique.

TABLE IX. DETAILED RESULTS.

Method	n	λ (10^{-3})	μ (10^{-2})	C_s	A_s	NFE	SD	CPU (s)
COA	(3, 2, 2, 3, 2, 2, 3, 2, 3, 3)	(0.5435, 0.4903, 0.7522, 0.8219, 0.4606, 0.4313, 0.6612, 0.6649, 0.5661, 0.8780)	(0.4393, 0.8895, 1.0907, 0.5776, 0.8726, 0.7540, 0.4958, 1.1125, 0.4630, 0.6139)	249.9922	**0.9765**	**2,000**	**0.001884**	**1.5912**
GA	(3, 3, 2, 2, 3, 3, 2, 2, 2, 2)	(0.5978, 0.7117, 0.8648, 0.6510, 0.4545, 0.8658, 0.5126, 0.6194, 0.2048, 0.7043)	(0.3637, 0.4170, 1.0598, 0.9699, 0.3067, 0.6567, 0.8701, 0.9023, 1.7164, 0.9745)	248.5754	0.9589	25,000	0.0166	1.6536
FPA	(3, 2, 2, 2, 2, 2, 3, 3, 3, 2)	(0.5157, 0.7579, 0.4919, 0.6473, 0.5591, 0.9095, 0.4938, 0.8977, 0.7457, 0.6910)	(0.3410, 0.7964, 1.0277, 0.8878, 1.0974, 1.4357, 0.2876, 0.5501, 0.4852, 0.8708)	249.8303	0.9644	40,000	0.0181	5.8032
DE	(3, 2, 2, 1, 2, 1, 2, 3, 3, 2)	(0.4844, 0.3590, 0.7673, 0.4063, 0.5366, 0.7744, 0.6316, 0.9290, 0.6903, 0.5165)	(0.4681, 0.5971, 1.1225, 1.9211, 0.2822, 1.1092, 0.8182, 0.8442, 0.8668, 0.4274)	244.0932	0.9654	10,000	0.0181	3.66
PSO	(2, 2, 2, 2, 2, 2, 3, 2, 2, 2)	(0.3515, 0.5183, 0.7869, 0.5934, 0.5166, 0.5958, 0.7610, 0.7557, 0.5538, 0.6736)	(0.6501, 0.8490, 1.0842, 0.8974, 0.8375, 0.9218, 0.5122, 1.0620, 0.8899, 0.9773)	249.9994	0.9649	10,500	0.0128	8.5333

It can be also observed that the COA has required fewer number of function evaluations (2,000), consumed less CPU time (1.5912 s), and is more stable in term of standard deviation (0.001884) than the other techniques. Figure 3 summarizes the NFE values required by each technique. The above results reveal that the penalty function has been successfully implemented and the COA has outperformed the other applied evolutionary computation techniques.

VI. CONCLUSIONS

In this paper, the system availability of a parallel-series system has been addressed using five evolutionary computation techniques: genetic algorithm (GA), differential evolution (DE), particle swarm optimization (PSO), flower pollination algorithm (FPA), and the cuckoo optimization algorithm (COA). A case study involving 10 subsystems has been investigated and a penalty function has been implemented for handling the constraints. From the results it can be concluded that the COA has outperformed the other four techniques in terms of system availability, number of function evaluations, standard deviation, and CPU time. On the other hand the integer values of redundancy have been successfully handled in implementing the five techniques. Further works on other system configurations will be performed to confirm the results.

REFERENCES

[1] G. S. Liu, "Availability optimization for repairable parallel-series system by applying Tabu-GA combination method," in *10th IEEE International Conference on Industrial Informatics*, 2012, pp. 803–808.

[2] G. S. Liu, "Availability optimization for repairable n-stage standby system by applying Tabu-GA combination method," *Int. J. Model. Optim.*, vol. 3, no. 3, pp. 245–250, 2013.

[3] Y. C. Hsieh and P. S. You, "An effective immune based two-phase approach for the optimal reliability–redundancy allocation problem," *Appl. Math. Comput.*, vol. 218, no. 4, pp. 1297–1307, 2011.

[4] H. Garg, M. Rani, and S. P. Sharma, "An efficient two phase approach for solving reliability–redundancy allocation problem using artificial bee colony technique," *Comput. Oper. Res.*, vol. 40, no. 12, pp. 2961–2969, 2013.

[5] M. A. Mellal and E. J. Williams, "Large scale reliability-redundancy allocation optimization problem using three soft computing methods," in *Modeling and Simulation based Analysis in Reliability Engineering*, CRC Press Francis & Taylor, 2018, pp. 199–214.

[6] M. A. Mellal and E. Zio, "System reliability-redundancy allocation by evolutionary computation," in *2017 2nd International Conference on System Reliability and Safety*, 2017, pp. 15–19.

[7] M. A. Mellal and E. Zio, "A penalty guided stochastic fractal search approach for system reliability optimization," *Reliab. Eng. Syst. Saf.*, vol. 152, pp. 213–227, 2016.

[8] M. Marseguerra, E. Zio, and S. Martorell, "Basics of genetic algorithms optimization for RAMS applications," *Reliab. Eng. Syst. Saf.*, vol. 91, pp. 977–991, 2006.

[9] J. H. Holland, *Adaptation in Natural and Artificial Systems*, vol. Ann Arbor. 1975.

[10] G. Renner and A. Ekart, "Genetic algorithms in computer aided design," *Comput. Des.*, vol. 35, no. 8, pp. 709–726, 2003.

[11] R. Storn and K. Price, "Differential evolution – A simple and efficient adaptive scheme for global optimization over continuous spaces," Berkeley, CA, USA, 1995.

[12] E. Zio, L. R. Golea, and G. Sansavini, "Optimizing protections against cascades in network systems: A modified binary differential evolution algorithm," *Reliab. Eng. Syst. Saf.*, vol. 103, pp. 72–83, 2012.

[13] E. Zio and G. Viadana, "Optimization of the inspection intervals of a safety system in a nuclear power plant by Multi-Objective Differential Evolution (MODE)," *Reliab. Eng. Syst. Saf.*, vol. 96, no. 11, pp. 1552–1563, 2011.

[14] J. Kennedy and R. Eberhart, "Particle swarm optimization," *Neural Networks, 1995. Proceedings., IEEE Int. Conf.*, vol. 4, pp. 1942–1948 vol.4, 1995.

[15] M. A. Mellal and E. J. Williams, "A survey on ant colony optimization, particle swarm optimization, and cuckoo algorithms," in *Handbook of Research on Emergent Applications of Optimization Algorithms*, IGI Global., USA, 2018.

[16] R. Rajabioun, "Cuckoo optimization algorithm," *Appl. Soft Comput.*, vol. 11, no. 8, pp. 5508–5518, 2011.

[17] M. A. Mellal and E. J. Williams, "The cuckoo optimization algorithm and its applications," in *Handbook of Neural Computation*, Elsevier., 2017, pp. 269–277.

[18] X.-S. Yang, "Flower pollination algorithm for global optimization," *Unconv. Comput. Nat. Comput.*, vol. 7445, pp. 240–249, 2012.