Hindawi
Journal of Optimization
Volume 2021, Article ID 6385713, 9 pages
https://doi.org/10.1155/2021/6385713

Research Article

Optimization for the Redundancy Allocation Problem of Reliability Using an Improved Particle Swarm Optimization Algorithm

H. Marouani [1,2]

[1]*College of Engineering, Muzahimiyah Branch, King Saud University, P.O. Box 2454, Riyadh 11451, Saudi Arabia*
[2]*University of Monastir, LGM, ENIM, Avenue Ibn-Eljazzar, Monastir 5019, Tunisia*

Correspondence should be addressed to H. Marouani; hmarouani@ksu.edu.sa

Received 19 September 2021; Revised 2 November 2021; Accepted 5 November 2021; Published 23 November 2021

Academic Editor: Sorin Mihai Grad

This paper presents an enhanced and improved particle swarm optimization (PSO) approach to overcome reliability-redundancy allocation problems in series, series-parallel, and complex systems. The problems mentioned above can be solved by increasing the overall system reliability and minimizing the system cost, weight, and volume. To achieve this with these nonlinear constraints, an approach is developed based on PSO. In particular, the inertia and acceleration coefficients of the classical particle swarm algorithm are improved by considering a normal distribution for the coefficients. The new expressions can enhance the global search ability in the initial stage, restrain premature convergence, and enable the algorithm to focus on the local fine search in the later stage, and this can enhance the perfection of the optimization process. Illustrative examples are provided as proof of the efficiency and effectiveness of the proposed approach. Results show that the overall system reliability is far better when compared with that of some approaches developed in previous studies for all three tested cases.

1. Introduction

With the continuous advance of technology and increasing complexity of industrial systems, it has become imperative for all production processes to perform adequately during their designed life cycle. However, errors do occur and can be linked to the human factors either in the processing, in the utilization, improper storage facilities, and poor maintenance or several other environment-related factors [1]. Therefore, in last three decades, system reliability becomes an important issue in improving the performance of any industrial system. Reliability is defined as the probability of achieving a set functionality goal or product function to successfully achieve these goals within a timeframe and in a controlled environment. Two approaches have been commonly used by designers to achieve the desired system reliability. The first is increasing the component reliability and the second is dividing the system into multiple subsystems and using redundant components with the same or less reliability for different subsystems. The first approach is quite expensive (i.e., higher reliability increases the component cost), and the required reliability improvement may fail to be realized even when the most reliable components are used. The second approach involves using a combination of optimal redundant components. Although the reliability of the entire system is improved accordingly, the associated cost, weight, and volume may also be affected. The problem of maximizing system reliability through the selection of redundancy and component reliability is called a reliability-redundancy allocation problem (RRAP).

The RRAP is a nondeterministic polynomial-time hard problem whose solution cannot be realized by direct, indirect, or mixed search approaches because of the discrete search space; that is, the optimum numbers of components are integers, and the components' reliability is floating. Recently, many deterministic methods were tried out, such as the metaheuristic [2–4], evolutionary algorithms [5], bee/ant colony optimization [6, 7], Tabu search [8], particle

swarm optimization (PSO) [9, 10], artificial neural networks [11], artificial immune system [12], fuzzy system [13, 14], cuckoo search algorithm [15–17], reduced gradient method [18], branch and bound method [19], integer programming [20], dynamic programming [21], and even some biogeography-based optimizations (BBOs) [22]. The whole idea of there being various ways proves that the efficiency and computing time can still be improved, and each of these techniques has its advantages and limitations. For example, dynamic programming is only applicable where there are decomposable objective functions and constraints. To accommodate several constraints within a dynamic programming formulation, Lagrangian multiplier for the weight/volume/cost constraints or a surrogate constraint combining cost, volume, and weight into one is utilized. Beeand colony optimization are more adequate for difficult combinatorial problems. The required memory rapidly goes high with the RRAP size when using the bound and branch approach. Genetic algorithms (GAs) are dysfunctional where an exact solution is required but are the finest when reaching a global region. Cuckoo search algorithm has been a recent discovery in 2009 [17]. The number of cuckoo search algorithm parameters seems to be less than GA and particle swarm optimization. BBO is also a new entrant in the RRAP optimization problem [22]. BBO has certain features collaborating with those of the GA and PSO which means that BBO can share information between solutions. However, BBO does not have crossover step, its set of solutions is maintained and improved from one generation to the next by migrating, and the solutions of BBO are changed directly via migration from other solutions, i.e., BBO solutions have common characteristics with other solutions. In a bid to consider the uncertainty of some RRAP parameters in real-life cases, mathematical models of such problems can be developed in the fuzzy environment. In these cases, it is almost impossible to get the exact optimal solution, and the optimization problem is to try and develop a solution which is close to a specified single objective. Finding an acceptable solution set requires solving the multiobjective fuzzy optimization problem repeatedly.

PSO is a stochastic global optimization technique inspired by social behavior of bird flocking or fish schooling. It is commonly applied in the optimization problem, such as in the RRAP [23–26]. The algorithm works by starting a group randomly over a specified search area where every bird/fish (i.e., particle) fly from a certain position at a certain velocity with the objective to find the global best position after this is done repeatedly. To reach this global best position and at each iteration, each particle adjusts its velocity based on its momentum and the influence of its previous best position and the best position of the other partakers. Literature studies [23–29] concluded that using only the PSO is, in some cases, not sufficient to obtain the ideal solution. Indeed, classical/traditional PSO algorithm has flaws: it leads to premature convergence phenomenon, which cannot perform a good global search and fall into the local solution convergence situation, and then it becomes hard to adapt to complex nonlinear optimization problems. Due to this, some authors used hybrid approaches [30–32], where the PSO is combined with the genetic algorithm to enhance the effectiveness.

The purpose of this work is to develop an efficient PSO-based approach in order to fix its deficiencies. In our proposed option, the inertia coefficient, velocity, and position updating are described by considering a normal distribution which can realize the multistep hopping of particles in the search space to meet the objective of improving search efficiency. The improved PSO is tested for solving the RRAP in series, series-parallel, and bridge systems. The system reliability is maximized with the nonlinear system constraints relative to volume, weight, and system cost. Series, series-parallel, and bridge systems are taken into consideration and optimized to assess the effectiveness and performance of our advanced procedure. All computational results are compared with those from the literature and obtained using the simulated annealing algorithm, biogeography-based optimization, surrogate constraint algorithm, cuckoo search algorithm, classical PSO, and hybrid GA-PSO algorithm. In conclusion, the suggested IPSO outperforms the simulated annealing technique, the traditional PSO algorithm, and the upgraded GA-PSO algorithm in all three system configurations (i.e., series, series-parallel, and bridge systems). Recent strategies such as biogeography-based optimization and the cuckoo search algorithm, on the contrary, appear to be more efficient, albeit the difference is minor.

The remaining of the paper is displayed as follows. The RRAP is described in Section 2; Section 3 introduces the classical and our improved PSO approach; numerical examples and computational results are depicted in Section 4; and conclusions are summarized in Section 5.

2. Redundancy Optimization of the Reliability Problem

Industrial systems can be generally structured in four types: series, parallel, series-parallel, and complex (bridge) systems. Figure 1 shows the systems analyzed in this study. Each subsystem is composed of x parallel components (redundant) with similar features in matters pertaining to weight, cost, volume, and reliability.

For each series, series-parallel, and bridge system, the reliabilities are defined, respectively, as

$$\begin{aligned} R_{\text{system}} &= R_1R_2R_3R_4, \\ R_{\text{system}} &= 1-\left(1-R_1R_2\right)\left(1-\left(1-R_3\right)\left(1-R_4\right)R_5\right), \\ R_{\text{system}} &= R_1R_2+R_3R_4+R_1R_4R_5-R_1R_2R_3R_4-R_1R_2R_3R_5-R_1R_3R_4R_5-R_1R_2R_4R_5+2R_1R_2R_3R_4R_5, \end{aligned} \tag{1}$$

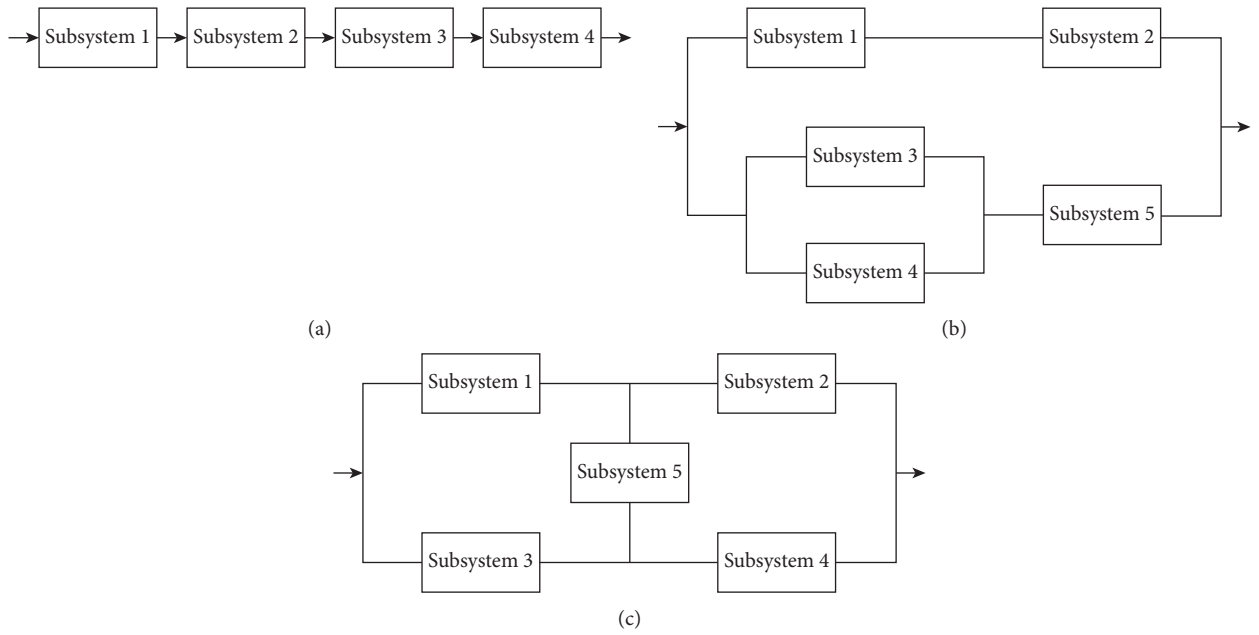

Figure 1: Representation of the (a) series system, (b) series-parallel system, and (c) bridge system.

where R_i is the reliability of the i^{th} parallel subsystem, as defined by

$$R_i = 1 - \left(1 - r_i\right)^{x_i}, \tag{2}$$

where x_i is the redundancy allocation of the i^{th} subsystem and r_i is the individual component reliability of this subsystem. In all the following configurations, we consider x_i as a discrete integer that ranges from 1 to 10 and r_i as a real number that ranges from 0.5 to $1–10^{-6}$.

The aim of the RRAP is to improve the reliability of the entire system under a specified controlled environment based on the cost, weight, and volume of the system. The system cost function $f_{\text{cost}}^{\text{system}}$, which depends on the number of components per subsystem, the number of subsystems, and the reliability of the components, can be defined as

$$f_{\text{cost}}^{\text{system}} = \sum_{i=1}^{n} f_{\text{cost}}^{\text{sybsystem } i} = \sum_{i=1}^{n} \alpha_i \left(\frac{-1000}{\ln r_i}\right)^{\beta_j} \left(x_i + e^{x_i/4}\right), \tag{3}$$

where n is the number of subsystems and α_i and β_i are the parameters of the cost function of subsystems. The value 1000 in the equation is the mean time between failures (i.e., the operating time during which the component must not fail).

Figure 2 shows the evolution of the cost function of the reliability in the cases of having 1 and 10 redundant components for one subsystem. The associated cost exponentially increases with reliability and the number of redundancy allocations. Figure 3 shows the evolution of the cost function of redundancy allocation for two levels of reliability (i.e., 0.9 and 0.99), where the cost ratio is 33.94 (for $\beta = 1.5$).

The weight function of the system depends on the number of components per subsystem and the number of subsystems and can be defined as

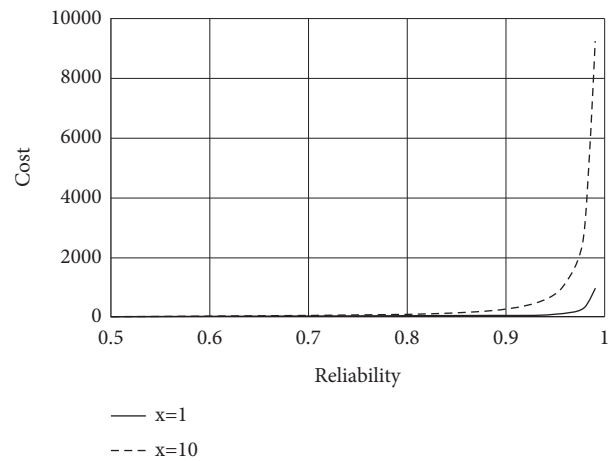

Figure 2: Evolution of the cost function of reliability ($\alpha = 2.3\ 10^{-5}$ and $\beta = 1.5$).

$$f_{\text{weight}}^{\text{system}} = \sum_{i=1}^{n} f_{\text{weight}}^{\text{sybsystem } i} = \sum_{i=1}^{n} w_i \cdot x_i \cdot e^{x_i/4}, \tag{4}$$

where w_i is the weight function parameter.

Early studies used a range of 3 to 9 for the weight function parameter. The weight is found to increase exponentially with the reliability and the amount of redundancy allocation (Figure 4).

The system volume function also depends on the number of components per subsystem and the number of subsystems, and it can be defined as

$$f_{\text{volume}}^{\text{system}} = \sum_{i=1}^{n} f_{\text{volume}}^{\text{sybsystem } i} = \sum_{i=1}^{n} v_i \cdot x_i^2, \tag{5}$$

where v_i is the volume function parameter.

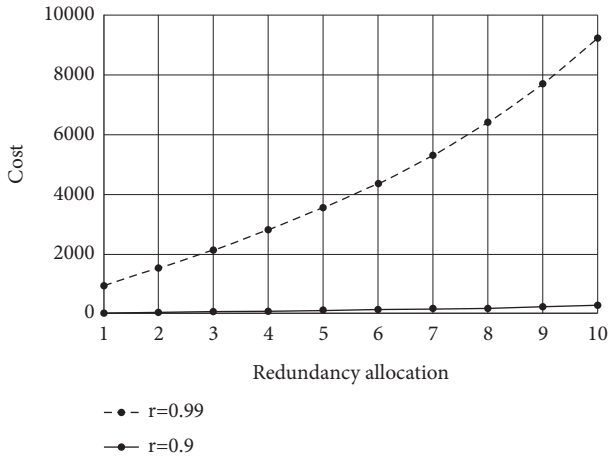

Figure 3: Evolution of the cost function of redundancy allocation ($\alpha = 2.3\ 10^{-5}$ and $\beta = 1.5$).

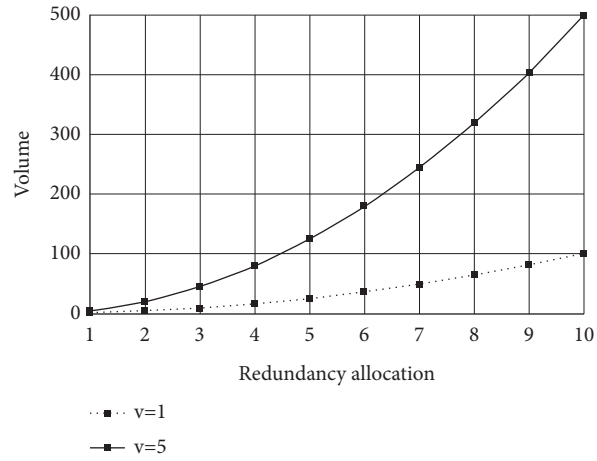

Figure 5: Evolution of the volume function of redundancy allocation (w= 3).

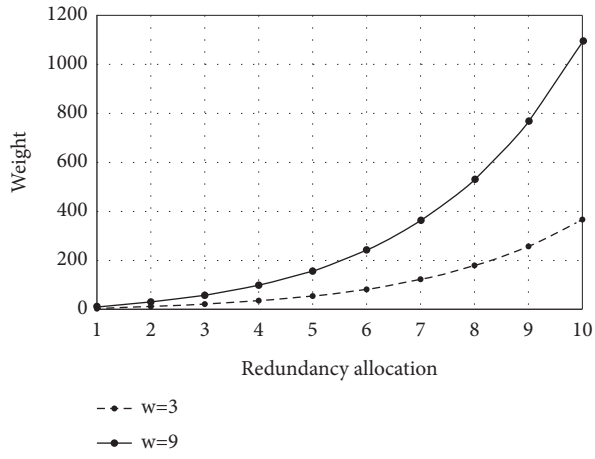

Figure 4: Evolution of the weight function of redundancy allocation.

Figure 5 shows the evolution of the subsystem volume function of redundant components.

In summary, the general RRAP can be expressed in the following formulation, in which the reliability is considered as the objective function.

$$\begin{cases} \max R_{\text{system}}, \\ s.t \begin{cases} f_{\text{cost}}^{\text{system}} \le C, \\ f_{\text{weight}}^{\text{system}} \le W, \\ f_{\text{volume}}^{\text{system}} \le V, \end{cases} \end{cases} \tag{6}$$

where C, W, and V are the upper bounds of the cost, weight, and volume of the system, respectively.

3. Improved Particle Swarm Optimization Algorithm

To solve the three proposed systems, an improved PSO (IPSO) approach was developed. The classical PSO algorithm can easily become premature and fall into the local extremum and is difficult to functionalize in complex nonlinear problems. This study introduces a new formulation of the inertia coefficient, acceleration coefficient, velocity, and position updating to improve the accuracy and fitness of the technique [33].

The PSO implementation involves defining the "workspace" by setting the maximum and minimum variables, population size n, maximum number of iterations $i_{\max}$, and other constants. A generation of the initial position and velocity is done randomly for each particle. Then, for each iteration, the inertia weight, position, and velocity are updated.

The inertia weight is a key factor in the PSO. A large inertia coefficient can improve the global search ability of the algorithm and avoid premature convergence caused by falling into the local extremum. On the contrary, a small inertia coefficient value can enable accurate search in the space and improve the convergence accuracy. To avoid falling into the local extremum and improve the diversity of particles, we introduce a normal model of the inertia weight ψ that is described by

$$\psi = \begin{cases} \eta_1 e^{-(f - f_{\text{aver}})^2 / 2\sigma^2} & \text{if } f \ge \overline{f}, \\ \eta_2 \left(\psi_{\max} - \dfrac{f - f_{\min}}{f_{\max} - f_{\min}} (\psi_{\max} - \psi_{\min}) \right) & \text{if } f < \overline{f}, \end{cases} \tag{7}$$

where $\psi_{\max}$ is the maximum inertia coefficient of the particle swarm; $\psi_{\min}$ is the minimum inertia coefficient of the particle swarm; f is the fitness value of the particle; $f_{\max}$ and $f_{\min}$ are, respectively, the maximum and minimum fitness values of the particle swarm; f_{aver} is the average fitness value of the particle swarm; and η_1, η_2, and σ are constants in the range of 0 to 1.

As the inertia coefficient decreases gradually, the algorithm switches from the initial generalized search to the local fine search in the later stage. For the classical approach, the inertia coefficient is simply defined as

$$\psi = \psi_{\max} - \frac{(\psi_{\max} - \psi_{\min})i}{i_{\max}}, \tag{8}$$

where $i_{\max}$ and i are the maximum iteration and number of iterations, respectively.

The acceleration coefficients (i.e., the self-acceleration c_1 and global acceleration c_2) can determine the influence of the particle self-cognition on the particle trajectory and reflect the degree of information exchange between particles in the swarms. More specifically, c_1 and c_2 are a representation of the acceleration weights of particles advancing toward their own extremums and global extremums, respectively. In classical PSO, they are all considered as constants. However, in our improved approach, they are calculated as follows:

$$\begin{cases} c_1 = c_1^{\text{start}} + \left(c_1^{\text{end}} - c_1^{\text{start}}\right)\sin(\psi), \\ c_2 = c_2^{\text{start}} + \left(c_2^{\text{end}} - c_2^{\text{start}}\right)\cos(\psi), \end{cases} \tag{9}$$

where c_1^{start}, c_2^{start}, c_1^{end}, and c_2^{end} are the initial and termination values of the acceleration coefficients, respectively.

Setting a larger global acceleration coefficient and smaller self-acceleration coefficient at the beginning of the PSO algorithm can lead to a stronger social learning ability of the particle and weaker self-learning ability, which is beneficial to strengthening the global search ability. However, defining a smaller global acceleration coefficient and larger self-acceleration coefficient at the later stage of the PSO algorithm brings about stronger self-learning ability for the particle and weaker social learning ability, which is beneficial to local fine search that converges to the global optimal solution with higher precision.

In the PSO, the particle velocity and position are updated in each iteration as follows:

$$v_{i,j}^{t+1} = \psi \cdot v_{i,j}^{t} + c_1 r_1^t \left(BP_{i,j}^t - x_{i,j}^t\right) + c_2 r_2^t \left(BG_j^t - x_{i,j}^t\right), \tag{10}$$

$$x_{i,j}^{t+1} = x_{i,j}^{t} + v_{i,j}^{t+1}, \tag{11}$$

where t represents the current number of iterations, i represents the particle number, and j represents the j^{th} dimension of the particle; r_1 and r_2 are random real numbers in the range of 0 to 1.

The parameter $BP_{i,j}^t$ is the best previous position of the particle called "the best personal position," whereas BG_j^t is the best position obtained from the population called "the best global position." The initial best personal position is considered as the initial position of the particle, and the initial best global position is set as the initial position of the particle with the best fitness in the particle swarm.

The term $\psi \cdot v_{i,j}^t$ represents the momentum part of the particle that reflects the inertia of the particle motion and its ability to maintain its previous velocity; $c_1 r_1^t (BP_{i,j}^t - x_{i,j}^t)$ represents the self-recognition of the particle, which is an indicator of the memory of the particle's own historical experience, and indicates that the particle has the tendency to approach its best historical position; and $c_2 r_2^t (BG_j^t - x_{i,j}^t)$ is the social consciousness part of the particle, which reflects the collective historical experiences of cooperation and knowledge common among particles, and indicates that the particle tends to approach the best historical position of the group or neighborhood. If the particle position or velocity exceeds the defined boundary range in the repetitive process, then the boundary value is utilized.

Compared to the classical PSO approach, equation (10) can be written as

$$v_{i,j}^{t+1} = w \cdot v_{i,j}^{t} + c_1 r_1^t \left(BP_{i,j}^t - x_{i,j}^t\right) + c_2 r_2^t \left(BG_j^t - x_{i,j}^t\right), \tag{12}$$

where c_1 and c_2 are constants and inertia weight w is a constant or can be a linear function of an iterative value.

The multiple objectives of the redundancy allocation problem are related with the cost, volume, and weight. A penalty function P is implemented to take into account the constraint functions of the system reliability. Based on recent research, we use the following expression [24, 34]:

$$P = R_{\text{system}} \cdot \left(\min \left\{ \frac{C}{f_{\text{cost}}^{\text{system}}}; \frac{W}{f_{\text{weight}}^{\text{system}}}; \frac{V}{f_{\text{volume}}^{\text{system}}} \right\} \right)^3, \tag{13}$$

where the fitness function f is defined as

$$f = \Gamma - P, \tag{14}$$

where Γ is a sufficiently large positive number.

The diagrammatic representation of the improved PSO algorithm is depicted in Figure 6.

4. Numerical Results

The values of the parameters for series, series-parallel, and bridge problems are given in Tables 1–3, respectively, (i.e., number of subsystems; component cost, volume, and weight values; and maximum value for the cost, volume, and weight) as stipulated in the literature [30, 32]. The IPSO method is coded in MATLAB R2019b (MathWorks), and the program was run on an i7-4700MQ @ 2.46 Hz Intel® Core™ processor with 8 GB of Random Access Memory. So as to measure the stochastic discrepancy, 20 runs were made for each system, which involves 20 different initial positions and initial velocities in the search domain for the particles. Herein, the best solutions are reported.

IPSO algorithm starts with defining the boundaries of the workspace by setting the maximum and minimum limits of the variables and initializing parameters, such as the population size n (50), the maximum and minimum inertia weights ($\psi_{\max} = 10$ and $\psi_{\min} = 1$), and the maximum number of iterations $i_{\max}$ (200), as described in the previous section and in Figure 6.

To evaluate the IPSO performance, we report the best, worst, median, and standard deviation (SD) in Table 4. SD is expressed as

$$S\,D = \sqrt{\frac{\sum_{i=1}^{20} (R_i - \overline{R})^2}{19}}, \tag{15}$$

where $\overline{R}$ is the system reliability average and R_i is the calculated reliability at iteration i.

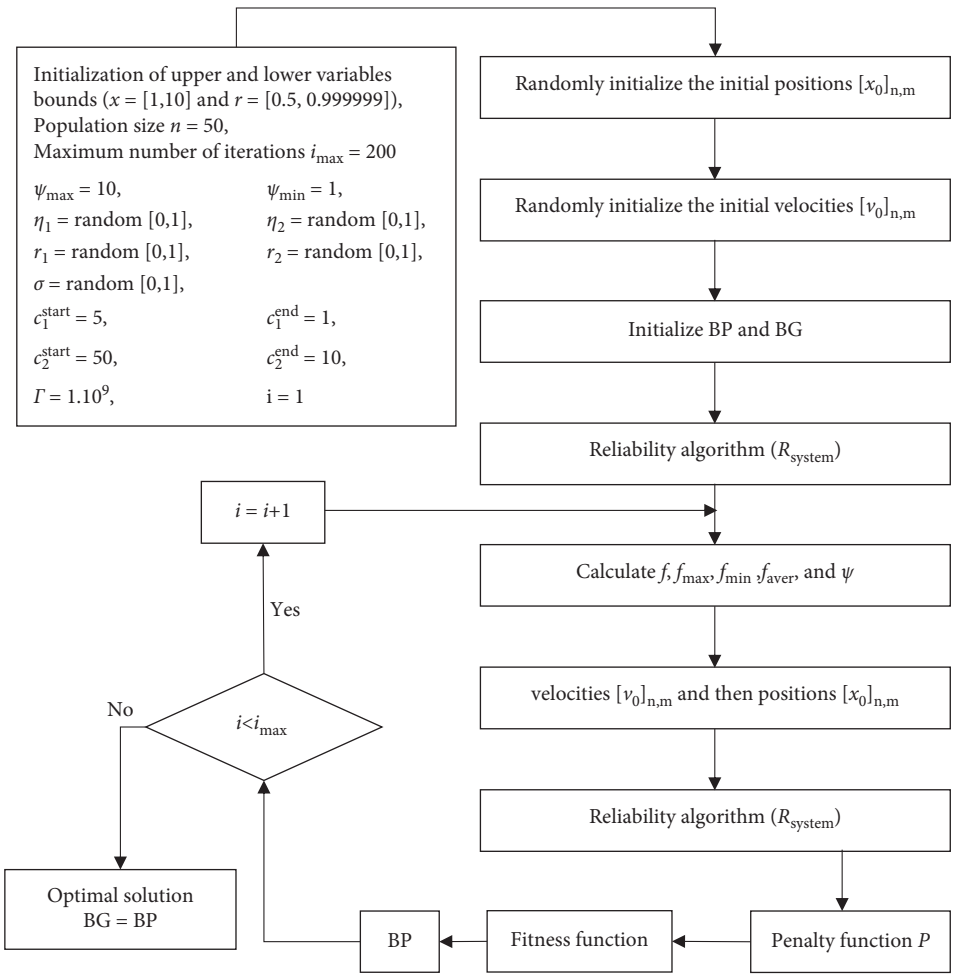

Figure 6: Flowchart of the improved particle swarm optimization algorithm.

Table 1: Input parameters of the series system.

Subsystem	α_i ($\times 10^{-5}$)	β_i	v_i	w_i	V	C	W
1	1.0	1.5	1	6	250	400	500
2	2.3		2	6			
3	0.3		3	8			
4	2.3		2	7			

Table 2: Input parameters of the series-parallel system.

Subsystem	α_i ($\times 10^{-5}$)	β_i	v_i	w_i	V	C	W
1	2.50	1.5	2	3.5	180	175	100
2	1.45		4	4			
3	0.54		5	4			
4	0.54		8	3.5			
5	2.10		4	4.5			

Table 3: Input parameters of the bridge system.

Subsystem	α_i ($\times 10^{-5}$)	β_i	v_i	w_i	V	C	W
1	2.33	1.5	1	7	110	175	200
2	1.45		2	8			
3	0.54		3	8			
4	8.05		4	6			
5	1.95		2	9			

Table 4: IPSO performances.

	Best	Worst	Median	SD (10^{-10})
Series system	0.99995469	0.99986601	0.99990384	6.430
Series-parallel system	0.99312499	0.99303183	0.99308784	9.537
Bridge system	0.98935571	0.98926162	0.98930717	8.268

Table 5: Series system: result comparison between the IPSO and algorithms from the literature.

	Simulated annealing algorithm [36]	PSO algorithm [20]	Improved GA-PSO algorithm [32]	Our IPSO algorithm
R_{system}	0.99946617	0.99995326	0.99995467	0.99995469
n_1	3	5	5	5
r_1	0.965593	0.902231	0.902231	0.901602
n_2	6	6	5	5
r_2	0.760592	0.856325	0.856325	0.888261
n_3	3	4	4	4
r_3	0.972646	0.948145	0.948145	0.948121
n_4	5	5	6	6
r_4	0.804660	0.883156	0.883156	0.849928
$f_{\text{cost}}^{\text{system}}$	466.3	399.0	400.0	400.0
$f_{\text{weight}}^{\text{system}}$	145.8	226.0	226	226.0
$f_{\text{volume}}^{\text{system}}$	158	195	195	195

Table 6: Series-parallel system: result comparison between the IPSO and algorithms from the literature.

	Simulated annealing algorithm [36]	Surrogate constraint algorithm [35]	Improved GA-PSO algorithm [32]	Biogeography-based optimization [22]	Our IPSO algorithm
R_{system}	0.94397376	0.99273890	0.94426072	0.99997664	0.99312499
n_1	2	3	2	2	3
r_1	0.812161	0.838193	0.819640	0.819658	0.880335
n_2	2	3	2	2	3
r_2	0.853346	0.855065	0.845091	0.844910	0.827082
n_3	2	1	2	2	2
r_3	0.897597	0.878859	0.895482	0.895487	0.873112
n_4	2	2	2	2	3
r_4	0.900710	0.911402	0.895517	0.895514	0.915546
n_5	4	3	4	4	2
r_5	0.866316	0.850355	0.868430	0.868468	0.701152
$f_{\text{cost}}^{\text{system}}$	175.0	175.0	175.0	175.0	174.6
$f_{\text{weight}}^{\text{system}}$	98.4	92.9	98.4	98.4	97.9
$f_{\text{volume}}^{\text{system}}$	140	127	140	140	162

Table 7: Complex system: result comparison between the IPSO and algorithms from the literature.

	Simulated annealing algorithm [36]	PSO algorithm [20]	Improved GA-PSO algorithm [32]	Biogeography-based optimization [22]	Cuckoo search algorithm [15]	IPSO algorithm
R_{system}	0.98214990	0.98913303	0.98919325	0.99988963	0.99988963	0.98935571
n_1	3	3	3	3	3	3
r_1	0.807263	0.826678	0.828134	0.828060	0.827855	0.807263
n_2	3	3	3	3	3	3
r_2	0.868116	0.857172	0.857831	0.858040	0.857626	0.868435
n_3	3	2	2	2	2	3
r_3	0.872862	0.914629	0.914192	0.914148	0.914752	0.872862
n_4	3	4	4	4	4	3
r_4	0.712673	0.648918	0.648069	0.647968	0.648217	0.723673
n_5	1	1	1	1	1	1
r_5	0.751034	0.715290	0.704476	0.704204	0.702670	0.5
$f_{\text{cost}}^{\text{system}}$	174.9	175	175	175	175	173.1
$f_{\text{weight}}^{\text{system}}$	195.7	198.4	198.4	198.5	198.5	195.7
$f_{\text{volume}}^{\text{system}}$	92	105	105	105	105	92

The results obtained using the algorithms proposed in the literature are taken into comparison with those from our proposed method, as summarized in Tables 5–7. These tables contain values of the system reliability, component number, and reliability of each subsystem, cost, weight, and volume of the system. These values correspond to the best solution out of 20 tests. It is realized that the IPSO solution demonstrates sometimes better performance than that reported in the previous works.

For the series system, the reliability of the IPSO system is 0.99995469 as compared to 0.99995467 in the previous study [32]. Note that Ha and Kuo [4] found an acceptable overall system reliability, but the cost constraint was altered in their research. For the series-parallel system, the reliability of the IPSO system is 0.99312499 as compared to 0.94426072–0.99997664 in the previous study [22, 32, 35, 36]. IPSO shows better performance over the simulated annealing algorithm, surrogate constraint algorithm, and improved GA-PSO algorithm. However, biogeography-based optimization seems more efficient (7% improvement than the IPSO result). Finally, for the complex system, the reliability of the IPSO system is 0.98935571 and is compared to 0.9821499–0.99988963 in the previous studies [15, 20, 22, 32, 36]. IPSO shows its superiority to the simulated annealing algorithm, surrogate constraint algorithm, and improved GA-PSO algorithm. However, biogeography-based optimization and cuckoo search algorithm seem more efficient (1% improvement than the IPSO result).

In conclusion, the proposed IPSO is proven to provide better results in all three system configurations compared to the simulated annealing algorithm, classical PSO algorithm, and improved GA-PSO algorithm. However, recent techniques such as biogeography-based optimization and cuckoo search algorithm seem to be more efficient, although the improvement is quite minute.

5. Conclusion

This paper reports an enhanced approach based on PSO to solve multiobjective optimization applied to the RRAP. Series, series-parallel, and complex problems are considered, and the conclusions are compared to those of previous research. The purpose of the proposed approach is to improve the system reliability, subject to cost, volume, and weight constraints. To improve computational efficiency, an improved PSO approach is applied to search the solution space more efficiently. The inertia and acceleration coefficients of the algorithm are improved by employing a normal distribution for the coefficients. This was found to enhance the global search ability in the initial stage of the algorithm, restrain the premature convergence of the algorithm, and enable the algorithm to focus on the local fine search in the later stage, by which the optimization precision is improved and not the case for classical PSO. Case studies show that the solutions found by the IPSO are better or close to reported in the literature (i.e., simulated annealing algorithm, PSO algorithm, improved GA-PSO algorithm, biogeography-based optimization, and cuckoo search algorithm). Nowadays, even the simplest of developments in optimization for the redundancy allocation problem of reliability is often hard to come by. The IPSO algorithm proved to be a promising next generation of PSO or hybrid PSO approaches.

Data Availability

The data used to support the findings of this study are available upon request to the author.

Conflicts of Interest

The author declares that there are no conflicts of interest.

References

[1] R. Niwas and H. Garg, "An approach for analyzing the reliability and profit of an industrial system based on the cost free warranty policy," *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, vol. 40, 2018.

[2] X. Huang, F. P. A. Coolen, and T. Coolen-Maturi, "A heuristic survival signature based approach for reliability-redundancy allocation," *Reliability Engineering & System Safety*, vol. 185, pp. 511–517, 2019.

[3] M. Agarwal and R. Gupta, "Penalty function approach in heuristic algorithms for constrained redundancy reliability optimization," *IEEE Transactions on Reliability*, vol. 54, no. 3, pp. 549–558, 2005.

[4] C. Ha and W. Kuo, "Multi-path heuristic for redundancy allocation: the tree heuristic," *IEEE Transactions on Reliability*, vol. 55, no. 1, pp. 37–43, 2006.

[5] T. J. Hsieh, "Hierarchical redundancy allocation for multilevel reliability systems employing a bacterial-inspired evolutionary algorithm," *Information Sciences*, vol. 288, pp. 174–193, 2014.

[6] H. Garg, M. Rani, and S. P. Sharma, "An efficient two phase approach for solving reliability-redundancy allocation problem using artificial bee colony technique," *Computers & Operations Research*, vol. 40, no. 12, pp. 2961–2969, 2013.

[7] M. Agarwal and V. K. Sharma, "Ant colony approach to constrained redundancy optimization in binary systems," *Applied Mathematical Modelling*, vol. 34, no. 4, pp. 992–1003, 2010.

[8] M. Ouzineb, M. Nourelfath, and M. Gendreau, "Tabu search for the redundancy allocation problem of homogenous series-parallel multi-state systems," *Reliability Engineering & System Safety*, vol. 93, no. 8, pp. 1257–1272, 2008.

[9] Z. Ouyang, Y. Liu, S. J. Ruan, and T. Jiang, "An improved particle swarm optimization algorithm for reliability-redundancy allocation problem with mixed redundancy strategy and heterogeneous components," *Reliability Engineering & System Safety*, vol. 181, pp. 62–74, 2019.

[10] H. Garg and S. P. Sharma, "Multi-objective reliability-redundancy allocation problem using particle swarm optimization," *Computers & Industrial Engineering*, vol. 64, no. 1, pp. 247–255, 2013.

[11] V. V. Vinod and S. Ghose, "Neural network optimization for redundancy allocation," *Microelectronics Reliability*, vol. 34, no. 1, pp. 115–123, 1994.

[12] T. C. Chen, "IAs based approach for reliability redundancy allocation problems," *Applied Mathematics and Computation*, vol. 182, no. 2, pp. 1556–1567, 2006.

[13] H. Garg, M. Rani, S. P. Sharma, and Y. Vishwakarma, "Bi-objective optimization of the reliability-redundancy allocation

problem for series-parallel system," *Journal of Manufacturing Systems*, vol. 33, no. 3, pp. 335–347, 2014.

[14] Y. Hu, Y. Zhang, and D. Gong, "Multiobjective particle swarm optimization for feature selection with fuzzy cost," *IEEE Transactions on Cybernetics*, vol. 51, no. 2, pp. 874–888, 2021.

[15] H. Garg, "An approach for solving constrained reliability-redundancy allocation problems using cuckoo search algorithm," *Beni-Suef University Journal of Basic and Applied Sciences*, vol. 4, no. 1, pp. 14–25, 2015.

[16] H. Shokri-Ghaleh, A. Alfi, S. Ebadollahi, A. Mohammad Shahri, and S. Ranjbaran, "Unequal limit cuckoo optimization algorithm applied for optimal design of nonlinear field calibration problem of a triaxial accelerometer," *Measurement*, vol. 164, Article ID 107963, 2020.

[17] X. S. Yang and S. Deb, "Engineering optimisation by cuckoo search," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 1, no. 4, p. 330, 2010.

[18] C. L. Hwang, F. A. Tillman, and W. Kuo, "Reliability optimization by generalized lagrangian-function and reduced-gradient methods," *IEEE Transactions on Reliability*, vol. 28, no. 4, 1979.

[19] C. Ha and W. Kuo, "Reliability redundancy allocation: an improved realization for nonconvex nonlinear programming problems," *European Journal of Operational Research*, vol. 171, no. 1, pp. 24–38, 2006.

[20] L. S. Coelho, "An efficient particle swarm approach for mixed-integer programming in reliability-redundancy optimization applications," *Reliability Engineering & System Safety*, vol. 94, no. 4, pp. 830–837, 2009.

[21] A. Yalaoui, E. Châtelet, and C. Chu, "A new dynamic programming method for reliability & redundancy allocation in a parallel-series system," *IEEE Transactions on Reliability*, vol. 54, no. 2, pp. 254–261, 2005.

[22] H. Garg, "An efficient biogeography based optimization algorithm for solving reliability optimization problems," *Swarm and Evolutionary Computation*, vol. 24, pp. 1–10, 2015.

[23] X. Kong, L. Gao, H. Ouyang, and S. Li, "Solving the redundancy allocation problem with multiple strategy choices using a new simplified particle swarm optimization," *Reliability Engineering & System Safety*, vol. 144, pp. 147–158, 2015.

[24] C. L. Huang, "A particle-based simplified swarm optimization algorithm for reliability redundancy allocation problems," *Reliability Engineering & System Safety*, vol. 142, pp. 221–230, 2015.

[25] C. M. Lai and W. C. Yeh, "Two-stage simplified swarm optimization for the redundancy allocation problem in a multi-state bridge system," *Reliability Engineering & System Safety*, vol. 156, pp. 148–158, 2016.

[26] Y. Marinakis, M. Marinaki, and G. Dounias, "A hybrid particle swarm optimization algorithm for the vehicle routing problem," *Engineering Applications of Artificial Intelligence*, vol. 23, no. 4, pp. 463–472, 2010.

[27] E. S. Alaviyan Shahri, A. Alfi, and J. A. Tenreiro Machado, "Fractional fixed-structure H ∞ controller design using augmented Lagrangian particle swarm optimization with fractional order velocity," *Applied Soft Computing Journal*, vol. 77, 2019.

[28] X. F. Song, Y. Zhang, Y. N. Guo, X. Y. Sun, and Y. L. Wang, "Variable-size cooperative coevolutionary particle swarm optimization for feature selection on high-dimensional data," *IEEE Transactions on Evolutionary Computation*, vol. 24, 2020.

[29] X. Ji, Y. Zhang, D. Gong, and X. Sun, "Dual-surrogate-assisted cooperative particle swarm optimization for expensive multimodal problems," *IEEE Transactions on Evolutionary Computation*, vol. 25, 2021.

[30] L. Sahoo, A. Banerjee, A. K. Bhunia, and S. Chattopadhyay, "An efficient GA-PSO approach for solving mixed-integer nonlinear programming problem in reliability optimization," *Swarm and Evolutionary Computation*, vol. 19, pp. 43–51, 2014.

[31] N. Beji, B. Jarboui, M. Eddaly, and H. Chabchoub, "A hybrid particle swarm optimization algorithm for the redundancy allocation problem," *Journal of Computer Science*, vol. 1, no. 3, pp. 159–167, 2010.

[32] M. Sheikhalishahi, V. Ebrahimipour, H. Shiri, H. Zaman, and M. Jeihoonian, "A hybrid GA-PSO approach for reliability optimization in redundancy allocation problem," *The International Journal of Advanced Manufacturing Technology*, vol. 68, no. 1-4, pp. 317–338, 2013.

[33] Y. Li, X. Yao, and M. Liu, "Multiobjective optimization of cloud manufacturing service composition with improved particle swarm optimization algorithm," *Mathematical Problems in Engineering*, vol. 2020, Article ID 9186023, 17 pages, 2020.

[34] W. C. Yeh and T. J. Hsieh, "Solving reliability redundancy allocation problems using an artificial bee colony algorithm," *Computers & Operations Research*, vol. 38, no. 11, pp. 1465–1473, 2011.

[35] M. Hikita, Y. Nakagawa, K. Nakashima, and H. Narihisa, "Reliability optimization of systems by a surrogate-constraints algorithm," *IEEE Transactions on Reliability*, vol. 41, no. 3, pp. 473–480, 1992.

[36] H. G. Kim, C. O. Bae, and D. J. Park, "Reliability-redundancy optimization using simulated annealing algorithms," *Journal of Quality in Maintenance Engineering*, vol. 12, no. 4, pp. 354–363, 2006.