International Journal of
Operations Research

International Journal of Operations Research Vol. 9, No. 1, 44–52 (2012)

A Mixed-Integer Hybrid Differential Evolution Method for Multi-Objective Reliability Problems

C.M. Liu[1,*] and J.L. Wu[1]

[1]Department of Industrial Engineering and Systems Management, Feng Chia University,

Taichung, Taiwan - 407

Received August 2011; *Revised February 2012*; *Accepted March 2012*

Abstract— In practice, several conflict objectives are very often considered in formulating reliability problems. The inclusion of several objectives always makes models and solution methods more complicated and complex. In this study, a modified hybrid differential evolution method (MHDEM) is proposed for solving multi-object mixed-integer nonlinear reliability problems. The proposed approach combines the min-max Pareto method and a repaired differential evolution method to find a Pareto solution. A penalty function is also considered in this approach for preventing infeasibility. An aircraft engine protection design problem is formulated and test problems are generated to run the developed method for the computational performance. The computational results suggest that the developed methods perform satisfactorily in terms of CPU times and solution quality.

Keywords— Multi-objective mixed-integer nonlinear program, Hybrid differential evolution method, Aircraft engine protection system.

1. INTRODUCTION

Serial-parallel systems with redundant are very often applied to reinforce the function of reliability problems. The serial-parallel problem can be categorized as at least two different types: the active redundant model and the standby redundant model. The active redundant model uses several parallel components in which each component operates actively. The active redundant system will operate well only if a specific number of components within this system operates normally. The standby redundant model also adopts several parallel components. However, the entire number of components is required to operate normally in order to make sure that the whole system can operate functionally. When some of parallel components fail, the standby components can operate using switch devices. If the failure rate of switching devices is excluded, the system reliability is usually higher in the standby redundant model than in the active redundant model. Although the system reliability for the standby redundant model is high, this model could cause additional cost, weight, and weight when more switch devices are added in the standby redundant model. If the failure rate for switching devices is included, the whole system reliability should be recalculated using the relationships among the system reliability, the contact reliability of switching devices, and the conditional dynamic/static system reliability. In order to cope with these problems, a multi-objective mathematical programming is very often applied for designing such reliability systems.

For practical design considerations, the reliability problem might become more complicated when several conflict goals are considered. Sakawa (1998) developed a multi-objective reliability optimization method to solve the optimal reliability design of large-scale series-parallel systems and used a surrogate worth tradeoff method for multi-objective models of reliability allocation problems. Inagaki *et al.* (1988) used an interactive optimal design with minimal cost and weight, and maximal system reliability. Gen *et al.* (1989) utilized a multi-objective programming method to solve optimal reliability for large-scale series-parallel systems. Misra and Sharma (1991) devised a meta-heuristic method and an effective reliability design tool to solve integer programming problems and utilized a multi-objective programming method to solve multi-objective redundancy optimization problems, respectively. Prasad and Raghavachari (1998) proposed a heuristic method to solve optimal component allocation of series-parallel networks and utilized an optimal allocation of inter-exchangeable component method to solve the optimal component allocation of series-parallel and parallel-series systems, respectively. Levitin and Lisniaski (2003) formulated a series-parallel multi-state system considering the choice of system elements in order to achieve a desired level of system survivability. They applied a genetic algorithm to solve this structure optimization problem. Liang and Smith (2004) applied an ant colony meta-heuristic optimization method to solve the redundancy allocation problem. Their results suggest that the developed ant colony meta-heuristic method is competitive with the best-known heuristics for redundancy

* Corresponding author's email: cmliu@fcu.edu.tw

allocation problems. Chen and You (2005) presented a penalty-guided immune algorithms-based approach for solving an integer nonlinear redundant reliability design problem. Their computational results show the proposed method is better than or as well as the previously best-known solutions.

Recently, Chang and Wu (2006) investigated the optimal multi-objective planning of large-scale passive harmonic filters for a multibus system under abundant harmonic current sources using the hybrid differential evolution method. The migrant and accelerating operations embedded in the hybrid differential evolution method were used to overcome traps of local optimal solutions and problems of time consumption. Massa *et al.* (2006) applied an hybrid differential evolution method to contemporarily determinate the weights of the subarrays and the group membership of the elements. Yuan *et al.* (2009) proposed a novel hybrid method to solve dynamic economic dispatch problems with valve-point effects, by integrating an improved differential evolution with the Shor's *r*-algorithm. The feasibility and effectiveness of their proposed hybrid method was demonstrated and shown that the proposed method was capable of yielding higher quality solutions. Zhang *et al.* (2009) proposed a novel reactive power optimization method based on hybrid differential evaluation algorithm for solving reactive power optimization problems. Their results show that the proposed method possesses good convergence performance, good robustness and high calculation accuracy. Qian *et al.* (2008) proposed a hybrid algorithm based on the differential evolution method for solving a permutation flow-shop scheduling problem. Qian *et al.* (2008) proposed a memetic algorithm based on the differential evolution method for a multi-objective job shop scheduling problem. Lu *et al.* (2010) proposed an adaptive hybrid differential evolution algorithm for solving a dynamic economic dispatch problem. Niknam (2009) proposed a hybrid algorithm for multi-objective distribution feeder reconfiguration problem. Li (2009) applied a hybrid differential evolution method for practical engineering problems. Those study results indicate that the hybrid differential evolution method might be a promising approach for solving multi-objective nonlinear programs.

The motivation of this study is to modify the hybrid differential evolution method for efficiently solving multi-objective reliability problems. A modified mix-integer hybrid differential evolution method is proposed for solving multi-objective reliability problems. The proposed approach initially applies the concept of min-max method and then utilizes a heuristic method, named repaired differential evolution method, to find a Pareto solution. An aircraft engine protection design problem is used to test the developed methods for the computational performance.

2. HYBRID DIFFERENTIAL EVOLUTION METHOD

2.1 Modified Hybrid Differential Evolution Method

The hybrid differential evolution method is based on the concept of genetic algorithm, which can be applied to solve the non-convex mixed-integer nonlinear optimization problem. Chiou and Wang (1998) applied this method to solve an optimal control problem of a bioprocess system. Lin *et al.* (1999) used this method to obtain solutions for mixed-integer nonlinear optimization problem. Costa and Olivera (2001) utilized similar approach to the solution of mixed-integer nonlinear programming problems. Babu and Angira (2002) applied similar differential evolution approach for global of mixed-integer nonlinear problem. For the multi-objective mixed-integer nonlinear problem, the hybrid evolution method may be an appropriate solution method.

In this study, the hybrid differential evolution method is modified to solve a multi-objective mixed integer nonlinear reliability problem using the concept of min-max method and a heuristic method. The basic procedures for the proposed modified hybrid differential evolution method consist of initialization, mutation, crossover, and reproduction. The detailed steps for the proposed method are given follows.

Step 0: Population Initialization

In this step N_p decision variables (x, y) are used for devising a searching heuristic. Those N_p decision variables are served as chromosomes. That means there are N_P chromosomes in the Gth generation, (x_i^G, y_i^G) , $i = 1,2,\ldots,N_P$, where $(x_i, y_i) = (x_{1i}\ldots x_{ji}\ldots x_{n_c i}, x_{1i}\ldots x_{ji}\ldots x_{n_I i})$ is the chromosome, and x_{ji} and y_{ji} are genes with their values within bounded intervals and coded with real and integer numbers. N_P parents are randomly generated using the following formula:

$$(x_i^0, y_i^0) = \mathrm{rand}(x_i^{ini}, y_i^{ini}), i = 1,2,\ldots,N_P , \tag{1}$$

where $\left(x_i, y_i\right) = \left(x_{1i}\ldots x_{ji}\ldots x_{n_c i}, y_{1i}\ldots y_{ji}\ldots y_{n_I i}\right)$ is defined as a chromosome and the value for $\left(x_i^{ini}, y_i^{ini}\right)$ should be within the upper and lower limits.

Step 1: Mutation

One basic element in the mutation step is the generation of difference vector. The difference vector is used to generate chromosomes for next generation or offspring from parent chromosomes using the following two equations:

$$u_i^G = x_{r_1}^G + \rho x_m \left[x_{r_2}^G - x_{r_3}^G\right] \tag{2}$$

$$v_i^G = y_{r_1}^G + \rho y_m \left[y_{r_2}^G - y_{r_3}^G\right] \tag{3}$$

$$r_1, r_2, r_3 \in \{1, 2, \ldots, N_P\}$$

where r_1, r_2, r_3 are exclusive integer random variables, and the mutation factors ρx_m and ρy_m are constant and $\in [0,2]$.

Step 2: Crossover

In the crossover step, if the difference between populations is too little, the generated offspring (u_i^{G+1}, v_i^{G+1}) will be quickly converged and no much improvement can be gained. Hence in order to increase the difference between offspring chromosomes, binomial distribution is utilized. The crossover between parent chromosome and offspring chromosome is operated under binomial distribution. The crossover operation can be performed using the following formula:

$$u_{ji}^G = \begin{cases} x_{ji}^G, & \text{if a random number} > \rho_c \\ u_{ji}^G, & \text{otherwise}, j = 1, \ldots, n_c; i = 1, \ldots, N_P \end{cases} \tag{4}$$

$$v_{ji}^G = \begin{cases} y_{ji}^G, & \text{if a random number} > \rho_c \\ v_{ji}^G, & \text{otherwise}, j = 1, \ldots, n_c; i = 1, \ldots, N_P \end{cases} \tag{5}$$

where ρ_c is a crossover factor and $\in [0,1]$.

Step 3: Fitness Value Calculation

Compute fitness values for each of N_P parent chromosomes (x_i^G, y_i^G) and N_P offspring chromosomes (u_i^G, v_i^G) using the objective function.

Step 4: Selection and Reproduction

Check the fitness value for each pair of parent chromosome (x_i^G, y_i^G) and offspring chromosome (u_i^G, v_i^G). If the offspring chromosomes are better than the parent chromosomes, then keep this offspring chromosomes. Otherwise, replace the offspring chromosomes with the parent chromosomes. That is,

$$(u_i^G, v_i^G) = \begin{cases} (x_i^G, y_i^G), \text{ if } \Phi(u_i^G, v_i^G) > \Phi(x_i^G, y_i^G), \; i = 1, \ldots, N_p \\ (u_i^G, v_i^G), \text{ otherwise, if } \Phi(u_i^G, v_i^G) \le \Phi(x_i^G, y_i^G), \; i = 1, \ldots, N_p \end{cases} \tag{6}$$

$$\Phi(u_b^G, v_b^G) = \min\{\Phi(u_i^G, v_i^G)\}, i = 1, 2, \ldots, N_P \tag{7}$$

where $\Phi(u_b^G, v_b^G)$ is the best fitness value and chromosome (u_b^G, v_b^G) is the best solution for this iteration. Then $\Phi(u_b^G, v_b^G)$ and (u_b^G, v_b^G) are kept for further improvement.

Step 5: Feasibility Test

Update the new solution as follows:

$$(x_i^{G+1}, y_i^{G+1}) = (u_i^G, v_i^G), i = 1, 2, \ldots, N_p \tag{8}$$

If the i^{th} chromosome is not feasible, then use the following formula to reproduce a feasible chromosome:

$$(u_k^G, v_k^G) = \begin{cases} \text{rand}(x^{ini}, y^{ini}) \\ \text{feasible}(u_w, v_w), \text{ otherwise}, R \ge r, k \ne w \end{cases} \tag{9}$$

where R is the reproduction number and r is the maximum number of reproduction.

Step 6: Stopping Criteria

The iteration will be terminated if either of the following criteria is met: (1) the number of iterations exceeds some specific maximum number and (2) the fitness value cannot be improved within certain number of iterations.

2.2 Min-Max Pareto Solution Method

Suppose we have a multi-objective mixed-integer nonlinear programming problem.

Minimize $f_1(x,y), f_2(x,y), \ldots, f_m(x,y)$ (10)
subject to $x \in \Omega$. (11)

where $f_i(x,y), i = 1, 2\ldots, m$ are m nonlinear objective function, x is real variable, y is integer variable, and Ω the feasible region. This problem is equivalent to the following min-max programming problem.

Minimize $\max\{Z_1, Z_2, \ldots, Z_m\}$
subject to $x \in \Omega$.

Let $f_1^{\min}$ be the minimal value of $f_1\left(x,y\right)$, $f_2^{\min}$ be the minimal value of $f_2\left(x,y\right)$,..., and $f_m^{\min}$ be the minimal value of $f_m\left(x,y\right)$. Also, we let

$$Z_1 = \frac{|f_1 - f_1^{\min}|}{f_1},$$
$$Z_2 = \frac{|f_2 - f_2^{\min}|}{f_2}, \tag{12}$$
$$\ldots,$$
$$Z_m = \frac{|f_m - f_m^{\min}|}{f_m}.$$

Then we can convert the multi-objective programming problem into a single objective problem.

Minimize Y
subject to $Z_1 \le Y$, $Z_2 \le Y$, ..., $Z_m \le Y$, and $x \in \Omega$.

3. IMPLEMENTATION

3.1 Multi-Objective Problem for Engine Protection System

The developed approach is implemented using a design problem of over-speed protection system in turbofan engine. The mission of this design problem aims to provide a reliable protection system during an over-speed operation of turbofan engines. The interior of a turbofan engine overlaps and combines with many components. The malfunction of some electronic parts might cause the breakdown of turbofan engines. The application of series-parallel with redundancy systems is very often to prevent the breakdown of electronic control systems and turbofan engines when some components are failure. In such a series-parallel with redundancy system, several identical and parallel components are arrayed in each stage. While the use of series-parallel with redundancy system can be used to increase the system reliability, more complexity, weight, volume, or cost may inevitably add to the design system.

Figure 1 displays a functional block diagram for an over-speed protection system that is installed on the turbofan engine. This protection system consists of one electronic control valve and three mechanical valves, which provide over-speed protection for the turbofan engine in a continuous way. Once the speed of turbofan engine is too fast, those four valves should be shut down immediately. Therefore, those valves should be arrayed in four stages as series-parallel systems and in each stage the failure rate of components is assumed to be known with some values. Figure 2 displays a reliability block diagram for the turbofan engine protection system. The reliability block diagram can be served as a framework for showing the logic design of reliability and components in each stage for any series-parallel systems. For instance, the decision variables R_i and n_i in stage i are to be determined for optimizing system reliability and system cost.

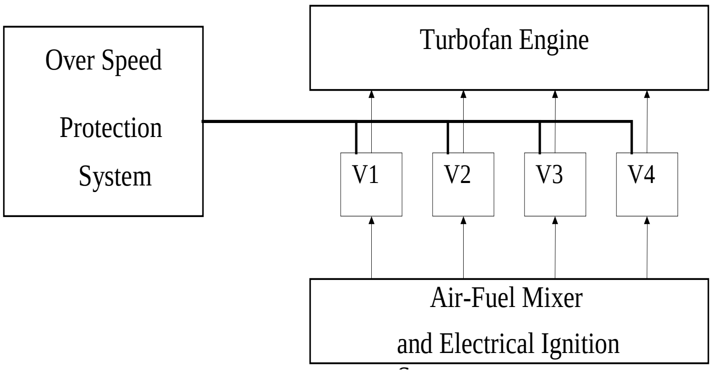

Figure 1. Functional diagram for an over-speed protection system of turbofan engine

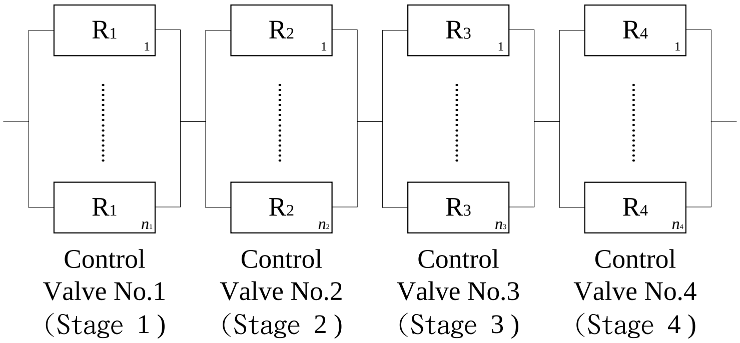

Figure 2. Reliability blocks for the protection system

During the modeling process, at least two objectives are considered in this problem. First, the overall system reliability is maximized. Second, the overall system cost is minimized. Also, several constrained design criteria, such as minimum requirements for system reliability, system cost, system volume, and system weight, are considered in this model. In order to develop a reliability modeling design for the series-parallel with redundancy problem, we define the following decision variables and parameters.

Decision variable:

R_i = represent the component reliability for the *i-th* stage；
n_i = represent the number of components for the *i-th* stage;
f_1 = represent the overall system reliability objective value;
f_2 = represent the overall system cost objective value.

Parameter:

$C_i(R_i)$ = represent the component unit cost with reliability R_i for the *i-th* stage;
w_i = represent each component weight for the *i-th* stage;
v_i = represent each component volume for the *i-th* stage;
R = represent the lower limit for the overall system reliability;
C = represent the upper limit for the overall system cost;
W = represent the upper limit for the system weight;
V = represent the upper limit for the system volume;
N = represent the number of stages in the design system;
n_{high} = represent the upper limit for the number of components for each stage;
n_{low} = represent the minimum requirement for the number of components for each stage;
R_{high} = represent the upper limit for the reliability of components for each stage;
R_{low} = represent the minimum requirement for the reliability of components for each stage.

Then a multi-objective reliability design model may be given as follows.

$$\text{Maximize} \quad f_1 = \prod_{i=1}^{N}\left[1-\left(1-R_i\right)^{n_i}\right] \tag{13}$$

$$\text{Minimize} \quad f_2 = \sum_{i=1}^{N} C_i\left(R_i\right) = \sum_{i=1}^{N} \frac{\alpha_i}{\lambda_i^{\beta_i}} = \sum_{i=1}^{N} \alpha_i \left\{\frac{-t}{\ln(R_i)}\right\}^{\beta_i} \tag{14}$$

$$\text{subject to} \quad \sum_{i=1}^{N} w_i\, n_i \exp(n_i / N) \le W \tag{15}$$

$$\sum_{i=1}^{N} v_i n_i^{\,2} \le V \tag{16}$$

$$n_{low} \leqq n_i \leqq n_{high}, \text{ integer, } i = 1,\ldots,N \tag{17}$$

$$R_{low} \leqq R_i \leqq R_{high},\ R_i \in R^n.\ i = 1,\ldots,N \tag{18}$$

The objective function (13) is used to maximize the overall system reliability for the series-parallel systems with redundancy problem, while the objective function (14) is used to minimize the overall system cost. Constraint (15) is used to set the upper limit for the system weight. Constraint (16) is used to set the upper limit for the system volume. Constraint (17) denotes the range of reliability for each component in each stage and constraint (18) is used to specify the allowable range of component number for each stage. The developed reliability design model is one type of multi-objective mixed integer nonlinear programming problem. This is a NP-hard problem.

3.2 Solution Procedure

The proposed approach is applied to the multi-objective mixed-integer nonlinear aircraft engine protection problem. The entire solution method combines the developed modified hybrid differential evolution method (MHDEM) and the min-max Pareto solution method. The detailed procedures are the following.

Step 1. Separate the multi-objective problem into two single objective problems, max f_1 and min f_2.

Step 2. Solve these two single objective problems using MHDEM and let $f_1^{\max}$ .and $f_2^{\min}$ be the solution, respectively.

Step 3. Formulate the following min-max problem.

$$\text{Minimize max} \quad \left\{ z_1 = \frac{\left|f_1\left(R,n\right) - f_1^{\max}\right|}{f_1^{\max}},\ z_2 = \frac{\left|f_2\left(R,n\right) - f_2^{\min}\right|}{f_2^{\min}} \right\} \tag{19}$$

$$\text{subject to} \quad \sum_{i=1}^{N} w_i\, n_i \exp(n_i / N) \le W \tag{20}$$

$$\sum_{i=1}^{N} v_i n_i^{\,2} \le V \tag{21}$$

$$n_{low} \leqq n_i \leqq n_{high}, \text{ integer, } i = 1,\ldots,N \tag{22}$$

$$R_{low} \leqq R_i \leqq R_{high},\ R_i \in R^n.\ i = 1,\ldots,N \tag{23}$$

And then convert into a single objective problem.

Minimize y

subjective to $z_1 \le y$

$$z_2 \le y$$

$$\sum_{i=1}^{N} w_i\, n_i \exp(n_i / N) \le W$$

$$\sum_{i=1}^{N} v_i n_i^{\,2} \le V$$

$$n_{low} \leqq n_i \leqq n_{high}, \text{ integer, } i = 1,\ldots,N$$

$$R_{low} \leqq R_i \leqq R_{high},\ R_i \in R^n.\ i = 1,\ldots,N$$

Step 4. Apply MHDEM to solve the min-max problem.

When MHDEM is applied to solve the mixed-integer nonlinear aircraft engine protection problem, a penalty function can be considered to use for converting the constrained problem into a non-constrained problem. In this way, we can maintain all solutions are feasible during the computation process.

3.3 Computational Results

To test the computational efficiency and solution quality for the developed MHDEM, 5 test problems from the aircraft engine protection system are formulated. These 5 test problems are solved by the MHDEM with and without a penalty function, respectively. The penalty function used in this computation is:

$$P\left(R,n\right) = r_1\{\max[0, g(R,n)]\}^2 + r_2\{\max[0, h(R,n)]\}^2 \tag{24}$$

and the resulting composite functions are given as:

$$F_1(R,n) = f_1(R,n) + \lambda_1 P(R,n) \tag{25}$$

$$F_2(R,n) = f_2(R,n) + \lambda_2 P(R,n) \tag{26}$$

where $g(R,n)$ is the weight constraint, $h(R,n)$ is the volume constrain, and $r_1, r_2, \lambda_1, \lambda_2$ are penalty parameters. When the MHDEM is applied, the population size is 50, the maximum number of iterations is 500, and the crossover factor is 0.3.

Table 1 displays the computational result using MHDEM with the penalty function. When MHDEM is used to solve single objective problems, the CPU times increase steadily as the number of variables increases, while the objective values appear satisfactorily. The CPU times in second are between 1.8 and 4.8. The maximum values (system reliability) are all greater than 0.96 and the minimum values (system cost) are between 21.3 and 274.9. When MHDEM is used to solve the multi-objective problem, the CPU times are a little longer and the trade-off Pareto solutions are getting worse significantly. The CPU times in second are between 3.2 and 10.8. The maximum values (system reliability) are between 0.87 and 0.40 and the minimum values (system cost) are between 46.0 and 703.6.

Table 2 displays the computational results using MHDEM without the penalty function. When single objective problems are solved, the CPU times are larger for MHDEM without the penalty function and the objective values are worse. The CPU times in second are between 2.6 and 11.1. The maximum values (system reliability) are all greater than 0.92 and the minimum values (system cost) are between 28.4 and 338.0. When the multi-objective problem is solved, the CPU times are smaller for MHDEM without the penalty function and the trade-off Pareto solutions are better. The CPU times in second are between 2.5 and 8.7. The maximum values (system reliability) are between 0.93 and 0.54 and the minimum values (system cost) are between 52.7 and 525.9. This result suggests that MHDEM with the penalty function is better for solving a single objective problem while MHDEM without the penalty function is better for the multi-objective problem. One possible reason is when the number of constraints is getting larger, the use of penalty function may take more CPU times for computation and provide less quality for Pareto solutions.

Table 1. Computational results for the test problems using MHDEM with penalty function

	Separate Single Optimum					Pareto Optimum		
	2 Constraints					6 Constraints		
Problem No	No of Var	Maximum	CPU in Sec	Minimum	CPU in Sec	Maximal Trade-off value	Minimal Trade-off Value	CPU in Sec
1	8	0.999999	1.828	21.3789	2.125	0.877879	46.0033	3.25
2	16	0.999912	2.641	55.3807	3.141	0.871006	106.675	5.281
3	24	0.999497	3.484	139.430	4.171	0.644156	244.467	7.156
4	32	0.965829	3.938	209.058	5.028	0.428206	455.397	8.922
5	40	0.959762	4.755	274.924	6.125	0.405606	703.663	10.843

Table 2. Computational results for the test problems using MHDEM without penalty function

	Separate Single Optimum					Pareto Optimum		
	2 Constraints					6 Constraints		
Problem No	No of Var	Maximum	CPU in Sec	Minimum	CPU in Sec	Maximal Trade-Off Value	Minimal Trade-Off Value	CPU in Sec
1	8	0.999999	2.281	28.4062	2.672	0.93179	52.7129	2.563
2	16	0.999989	3.297	60.4368	4.468	0.926401	128.788	4.672
3	24	0.998763	4.656	120.903	6.984	0.841393	191.674	5.928
4	32	0.972507	7.062	263.782	8.594	0.792419	419.69	8.219
5	40	0.921197	10.703	338.011	11.11	0.548533	525.962	8.733

4. CONCLUSIONS

A modified mixed-integer hybrid differential evolution method (MHDEM) with and without a penalty function is proposed for solving multi-objective mixed-integer nonlinear aircraft engine protection problems. The proposed method combines the min-max Pareto method and a hybrid differential evolution method with the penalty function and mixed coding. A set of test problems is used to test the computational performance. The computational results obtained suggest that the developed MHDEM performs satisfactorily in terms of CPU time and solution quality. Particularly, the developed MHDEM with the penalty function is better for solving a single objective problem while MHDEM without the penalty function works good for the multi-objective problem. When the number of constraints is getting larger, MHDEM without the penalty function may perform better.

REFERENCES

1. Babu, B.V. and Angira, R. (2002). A differential evolution approach for global of MINLP problems. *Proceedings of 4th Asia-Pacific Conference on Simulated Evolution and Learning*, Singapore, pp. 880-884.
2. Chang, Y.P. and Wu, C.J. (2005). Optimal multi-objective planning of large-scale passive harmonic filter using hybrid differential evolution method considering parameter and loading uncertainty. *IEEE Transactions on Power Delivery*, 20(1): 408-416.
3. Chen, T.C. and You, P.S. (2005). Immune algorithms-based approach for redundant reliability problems with multiple component choices. *Computers in Industry*, 56: 195-205.
4. Chiou, J.P. and Wang, F.S. (1998). A hybrid method of different evolution with application to optimal control problem of a bioprocess system. *Proceedings of IEEE World Congress on Computational Intelligence*, pp. 4-9.
5. Costa, L. and Olivera, P. (2001). Evolutionary algorithms approach to the solution of mixed integer non-linear programming problems. *Computers and Chemical Engineering*, 25: 257-266.
6. Gen, M., Ida, K., Sasaki, M. and Lee, J.U. (1989). Algorithm for solving large-scale 0-1 goal programming and its application to reliability optimization problem. *Computers and Industrial Engineering*, 17: 525-530.
7. Inagaki, T., Inoue, K., and Akashi, H. (1988). Interactive optimization of system reliability under multiple objectives. *IEEE Transactions on Reliability*, 27: 264-267.
8. Levitin, G. and Lisnianski, A. (2003). Optimizing survivability of vulnerable series-parallel multi-state systems. *Reliability Engineering and System Safety*, 79: 319-331.
9. Li, J., 2009. A hybrid differential evolution method for practical engineering problems. *Proceedings of 2009 IITA International Conference on Control, Automation and Systems Engineering*, pp. 54-57.
10. Liang, Y.C. and Smith, A.E. (2004). An ant colony optimization algorithm for the redundancy allocation problem. *IEEE Transactions on Reliability*, 53: 417-423.
11. Lin, Y.C., Lin, C.Y. and Chen, T.H. (1999). A hybrid method of evolutionary algorithms for mixed-integer nonlinear optimization problems. *IEEE–Evolutionary Computation*, 3: 2161-2166.
12. Lu, Y., Zhou, J., Qin, H., Li, Y., and Zhang, Y. (2010). An adaptive hybrid differential evolution algorithm for dynamic economic dispatch with valve-point effects. *Expert Systems with Applications*, 37: 4842–4849.
13. Massa, A., Pastorino, M., and Randazzo, A. (2006). Optimization of the directive of a monopulse antenna with a subarray weighting by a hybrid differential evolution method. *IEEE on Antennas and Wireless Propagation Letters*, 5(1): 155-158.
14. Misra, K.B. and Sharma, U. (1991). An efficient approach for multiple criteria redundancy optimization problems. *Microelectronic Reliability*, 31: 303-321.
15. Niknam, T. (2009). A new hybrid algorithm for multi-objective distribution feeder reconfiguration. *International Journal of Cybernetics and Systems*, 40(6): 508-527.
16. Prasad, V.R. and Raghavachari, M. (1998). Optimal allocation of interchangeable components to series-parallel reliability system. *IEEE Transactions on Reliability*, 47: 255-260.
17. Qian, B., Wang, L., Huang, D.X., and Wang, X. (2008). Scheduling multi-objective job shops using a memetic algorithm based on differential evolution. *International Journal of Advanced Manufacturing Technology*, 35: 1014–1027.

18. Qian, B., Wang, L., Hu, R., Wang, W.L., Huang, D.X., and Wang, X. (2008). A hybrid differential evolution method for permutation flow-shop scheduling. *International Journal of Advanced Manufacturing Technology*, 38: 757–777.
19. Sakawa, M. (1988). Multi-objective optimization by the surrogate worth trade-off method. *IEEE Transactions on Reliability,* R-27: 311-314.
20. Yuan, X., Wang, L., Zhang, Y., and Yuan, Y. (2009). A hybrid differential evolution method for dynamic economic dispatch with valve-point effects. *Expert Systems with Applications*, 36(2): 4042-4048.
21. Zhang, F.T., Song, J.H., Jian, L. and Cheng, X.L. (2007). A hybrid differential evolution method for optimal reactive power optimization. *Journal of Power System Technology*, 31(5): 234-241.