

# Software Test Resource Allocation Based on Adaptive Operator Selection

Wenjie Chang and Baolong Guo\*

Institute of Intelligent Control and Image Engineering of Xidian University, Xi'an 710071 China

\*Corresponding author

**Abstract**—With the rapid development of computer software, making the complexity of the software system increased dramatically, the inevitable cost of testing resources is also increasing. Under the conditions of limited resources, how to better find the balance between resource consumption and reliability obtains more and more people's attention. Optimization of test resource allocation (OTRAPs) involves finding the optimal reliability, cost, etc. Therefore, the traditional test resource allocation optimization problem is a multi-objective optimization problem. In recent years, many effective algorithms, such as MOEA/D[15] and other famous algorithms, and achieved good results. However, the shortcomings of these algorithms are the use of a single operator and fixed neighborhood size, although the operator is not adapted to each search stage and small neighborhood size can accelerate convergence and big neighborhood size can get rid of the local optimal solution, The algorithm (DS-MAB-MOEA/D) we proposed based on the Multi-armed Bandit principle to adaptively select the excellent operator and neighborhood size in the pool during the different stages of evolution. Considering the retention of Pareto set's diversity, this paper embeds distance sorting algorithm in DS-MAB-MOEA/D algorithm and applies it to 16-module system, Experiments show that the algorithm is better than the MOEA / D algorithm.

**Keywords**—computer software; software test resource allocation; multi-objective optimization; multi-armed bandit; moea/d

## I. INTRODUCTION

Ariane 5 rocket development costs up to 8 billion US dollars, but it exploded in 40 seconds after the launch, the cause of the accident is a BUG in positioning software. From this we can see how important the software testing is. But, software testing is also time and resource consuming. For example, When Microsoft developed Internet Explorer 4.0 version, the development phase took six months, while the latter part of the bug recovery took nearly eight months. With problem how to improve the reliability of a software system, Ohtera and Yamada et al. systematically describe test resource allocation problems (OTRAPs) [1]. Considering the high consumption of human and material resources in the testing stage, how to allocate resources effectively under the condition of limited resources to get higher reliability and lower resource consumption, which seems to be a multi-objective optimization problem. Usually a software system has a lot of different modules, in order to improve software reliability and reduce resource consumption, each module should be allocated a reasonable test resources. Hou et al. demonstrate that a

reasonable optimization scheme can effectively improve the reliability of software systems and reduce resource consumption[4]. In recent years, there are many excellent algorithms have been proposed, such as: MOEA / D multi-objective evolutionary algorithm, the algorithm get a good application in the field of resource allocation [2], [3],[9],[10] but the disadvantage of these algorithm is very sensitive to the choice of the operator to solve the problem of resource allocation . Faced a multi-objective optimization problem, a single operator is often not suitable for the entire search process, hence the realization of dynamic operator selection is an urgent problem to be solved. And the uniformity of Pareto optimal front hardly can be guaranteed by the operator selection method simply. In this paper, the crowding sorting method is used to ensure the uniformity of the optimal Pareto front.

## II. PROBLEM DESCRIPTION

In order to prove the reliability of the algorithm, complex 16-module system are applied to the proposed algorithm. The resources mentioned in the resource cost of the software testing phase include: manpower, available time, and total budget. In order to combine manpower and available time, we refer to the product of total number of people and the working hours of each person as total test time. We first test the cost and reliability as two optimization goals, and then time consuming as a third optimization goals.

The work we do is based on the above assumptions:

- Each module is independent of each other, whether in the development stage or in the unit testing phase.
- During the software testing phase, for all modules, the errors are randomly caused by the remaining errors in the software.
- The number of failures of any module  $i$  satisfies the non-homogeneous Poisson process distribution (NHPP) whose mean value is  $m_i(t)$  or the failure strength function  $\lambda_i(t) = dm_i(t) / dt$ .
- The total amount of available resources is a fixed value  $T$ .

Thus, we describe the software test resource allocation problem by assigning reasonable test resources to each module in the case where the total amount of test resources is a fixed

value T, resulting in acceptable software reliability, minimizing test cost and minimized resource consumption.

### III. SERIES AND PARALLEL SYSTEM DESCRIPTION

Typical serial-parallel system is very common in the actual software system, the general series and parallel system is a combination of a number of parallel systems and series system, the following out of the n parallel modules and m series modules in series and parallel system diagram:

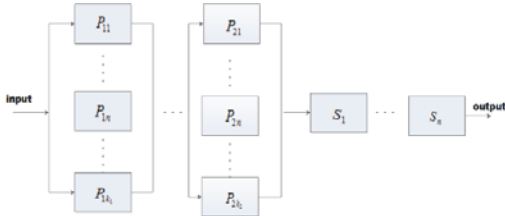


FIGURE 1. TYPICAL STRUCTURE OF A SERIES-PARALLEL MODULAR SOFTWARE SYSTEM

Based on the SRGMs model, the mean function  $m_i(t)$  and the fault sensitivity function  $\lambda_i(t)$  are:

$$m(t) = a(1 - e^{-bt}) \quad (1)$$

$$\lambda(t) = dm(t) / dt = abe^{-bt} \quad (2)$$

Where a is the total number of failures in the software before the test, and b is the effective sensitivity parameter for each failure.

Hence the run-time reliability function:

$$R_{op}(x | T) = \exp(-\lambda(T)x) = \exp(-abxe^{-bT}) \quad (3)$$

The reliability of the series-parallel system can be described by the following function:

$$R(x | T) = \prod_{i=1}^n (1 - \prod_{j=1}^{k_i} [1 - R_{li}(x | T_{li})]) \prod_{j=1}^m R_j(x | T_j) \quad (4)$$

Where  $T_i$  represents the test time allocated to the i module,  $1 - \prod_{j=1}^{k_i} [1 - R_{li}(x | T_{li})]$  represents the reliability function of the lth parallel module, then  $\prod_{i=1}^n (1 - \prod_{j=1}^{k_i} [1 - R_{li}(x | T_{li})])$  represents the reliability function of n parallel modules,  $\prod_{j=1}^m R_j(x | T_j)$  represents the reliability function of m series modules.

The total test cost of the series-parallel system can be described as follows:

$$C = \sum_{i=1}^n \sum_{j=1}^{k_i} C_{li}(R_{li}) + \sum_{j=1}^m C_j(R_j) \quad (5)$$

Where  $\sum_{i=1}^n \sum_{j=1}^{k_i} C_{li}(R_{li})$  indicates the total test overhead for n parallel modules,  $\sum_{j=1}^m C_j(R_j)$  indicates the total test test for m series modules.

### IV. ADPTIVE OPERATOR AND NEIGHBORHOOD SIZE SELECTION BASED MOEA/D FOR MULTI-OBJECTIVES OTRAPS

In the past few decades, many evolutionary algorithms for multi-objective optimization have been proposed. Schaffer's proposed Vector Assessment Genetic Algorithm (VEGA) [5] is the first multi-objective evolutionary algorithm. After that, Pareto Archive Evolutionary Strategy (PAES)[6], and enhanced version of the non-dominated sorting genetic algorithm (NSGA-II)[7]. An important problem with evolutionary algorithm is parameter setting and selection of evolutionary operators and neighborhood size, none of the evolutionary operators can be applied to each stage of evolution. The common drawback of these algorithms is the use of a single operator throughout the evolution. so we propose an algorithm based on MAB to select operator and PM to select neighborhood size during the search process, considering the uniformity of the Pareto optimal set, we embed the distance sorting algorithm into the DS-MAB-MOEA/D algorithm.

#### A. Adaptive Operator Selection of Multi-Arms Bandit

The choice of the operator is actually an instance of EvE, and we should choose as many as possible which perform good (exploration), but at the same time we should give chances to the poorly performing operators because these poorly performing operators are likely in the future search stage will have a good performance (exploration), we regard each operator as a separate arm, in order to get better improvement of fitness, we must choose the best arm in real time.

The dynamic adaptive operator selection (AOS) problem[10] mainly includes two aspects: credit allocation and operator selection:

##### 1) Credit Allocation:

The traditional credit allocation method is based on the increase of the simple improvement of fitness value, because the pre-period improvement of fitness value is significantly higher than the late stage during the evolution, so this will not be conducive to the robustness of the algorithm. Therefore, the algorithm uses the adaptive value improvement rate (FIR)[14] to allocate the operator's credit. FIR can be described as:

$$FIR_{i,t} = \frac{pf_{i,t} - cf_{i,t}}{pf_{i,t}} \quad (8)$$

Where  $pf_{i,t}$  indicates the parental fitness value,  $cf_{i,t}$  indicates offspring fitness value.

The Reward of the  $i$ -th operator is the sum of the FIR values of the operator in the current sliding window, Then introduce the decaying factor[14]  $D \in [0,1]$ , so that the performance of the operator can have more opportunities to be selected.

$$Decay_i = D^{Rank_i} \times Reward_i \quad (9)$$

The credit allocation function can be described as:

$$FRR_{i,t} = \frac{Decay_i}{\sum_{j=1}^K Decay_j} \quad (10)$$

## 2) Operator Selection

In this paper, we use the MAB algorithm selection operator based on UCB algorithm[12]. The  $t$ -time operator selection function can be described as:

$$\hat{q}_{i,t} + C \sqrt{\frac{2 \times \ln \sum_{j=1}^K n_{j,t}}{n_{i,t}}} \quad (11)$$

Where  $\hat{q}_{i,t}$  is the empirical quality of the  $i$ th operator,  $n_{i,t}$  is the number of times the operator has been selected throughout the search process, and  $C$  is a control factor between exploitation and exploration.

## B. Adaptive Neighborhood Size Selection of Probability Match

The size of the subproblem set is  $N$ . According to the weight vector, find the  $T$  nearest neighbor subproblems for each sub-problem  $i$ , then form a neighborhood set of the sub-problem. we use the information of the neighborhood set of sub-problem  $i$  to optimize the sub-problem  $i$ . Neighborhood size pool  $NS_S = \{N_1, N_2, \dots, N_k\}$  contains  $k$  candidates. This  $k$  candidates are different and sorted in ascending order, and this  $k$  candidate has the same initial selection probability

$P_c$ , and  $p_c = 1/k$ . The algorithm uses probability matching to select the neighborhood size. The small neighborhood size can make the search process converge rapidly and large neighborhood size can make the algorithm get rid of the local optimality. The neighborhood size selection probability uses the following update methods[16]:

$$P_{c,G} = \frac{S_{c,g}}{\sum_{c=1}^C S_{c,g}} \quad (12)$$

$$S_{c,g} = \frac{\sum_{g=G-LP}^{G-1} W_{-S_{c,g}}}{\sum_{g=G-LP}^{G-1} W_{c,g}} + \varepsilon, c = 1, \dots, C; G > LP$$

Where

Where  $LP$  is the learning cycle,  $S_{c,g}$  denotes the ratio of solutions to the next generation produced by the  $c$ -th neighborhood population in the previous learning cycle.  $W_{c,g}$  is the total solutions produced by  $c$ -th neighborhood population in the previous learning cycle.  $W_{-S_{c,g}}$  is the number of the solutions which succeed in entering the next generation. The  $\varepsilon$  is to avoid zero probability.

## C. Parameters Setting of DS-MAB-MOEA/D and MOEA/D

Assumes that there are 23 people to test the 16-module serial-parallel system, the total working time of each person is 1000 hours, so the total available test time is 50000 hours, The parameters of the 16 module test system refer[17], The parameters of the DS-MAB-MOEA/D algorithm are set as follows:

- The control parameter for DE is:  $CR = 1.0$ ;  $F = 0.5$ [11], The control parameters for the polynomial mutation are set to:  $\eta = 20$ ,  $p_m = 1/n$ [13].  $K = 0.5$ [10];
- For both reliability and test cost two - objective function optimization, the population of solutions is set to 600. For the reliability, test cost and resource consumption three target optimization, the population of solutions is set to 1000;
- The neighborhood size pool = {20,40,60,80},  $p_c = 1/k$ ,  $k=4$ ,  $LP=50$ ;
- Maximum number of solutions replaced by each new solution:  $nr = 2$ ;
- Probability with regard to selecting  $p$ :  $\delta = 0.9$  as in [11];
- Control parameters in MAB:
- Scaling factor:  $C = 5.0$ .
- Size of sliding window:  $W = 0.5 \times N$ .
- Decaying factor:  $D = 1.0$ .
- Number of independent runs and maximum number of function satisfactions: 30 independent runs are conducted for each algorithm on each test instance. The maximum number of function evaluations is fixed to be 300000;

About MOEA / D algorithm parameters settings, in addition to no set on the MAB, the other is completely consistent with DS-MAB-MOEA/D.

## V. COMPARISON OF DS-MAB/MOEA/D AND MOEA/D BOTH USED ON 16-MODULES SYSTEM

To prove the validity of this article, we choose C [15] as the index parameter and simulate 30 times. Compared with the classical MOEA / D algorithm, and both algorithms have the same population size and number of iterations. Table 1 is the result of C parameters for comparison of DS-MAB-MOEA / D and MOEA / D algorithm which are used on cost, reliability and test time these three-objective optimization function. The simulation results for the DS-MAB-MOEA / D Pareto solution, which is dominates the Pareto solution of MOEA / D, are highlighted in bold.

TABLE I. C(A,B)PERFORMANCE INDEX PARAMETER OF TRI-OBJECTIVE PROBLEM

Number of simulations	C(A,B)	C(B,A)
1	<b>0.99</b>	0.003
2	<b>0.85</b>	0.04
3	<b>0.81</b>	0.006
4	<b>0.99</b>	0.003
5	0.12	0.5
6	<b>0.776</b>	0.03
7	<b>0.733</b>	0.03
8	0.02	0.86
9	<b>0.92</b>	0.006
10	<b>0.83</b>	0.01
11	<b>0.77</b>	0.04
12	<b>0.97</b>	0.003
13	<b>0.6</b>	0.003
14	<b>0.89</b>	0.02
15	<b>0.72</b>	0.02
16	<b>0.98</b>	0.006
17	<b>0.87</b>	0.006
18	<b>0.80</b>	0.03
19	<b>0.966</b>	0.006
20	0.11	0.54
21	<b>0.94</b>	0.01
22	<b>0.97</b>	0.006
23	<b>0.89</b>	0.01
24	<b>0.99</b>	0.006
25	<b>0.97</b>	0.003
26	0.006	0.93
27	<b>0.536</b>	0.006
28	<b>0.85</b>	0.02
29	<b>0.94</b>	0.01
30	<b>0.98</b>	0.006

From Table I we can see that the DS-MAB-MOEA / D algorithm is used in the 16-module system to solve the three optimization objective problem and the pareto solutions obtained 86.7% dominates the MOEA / D algorithm and only the 26th simulation results acts poor. The results show that the DS-MAB-MOEA / D algorithm is more likely to find lower test time consumption and lower test cost with the same reliability.

From Fig. II and Fig. III, we can see that DS-MAB-MOEA / D algorithm and MOEA / D algorithm are respectively applied to 16-module system to solve test cost and reliability these two optimization objective problem. The performance of the algorithm DS-MAB-MOEA / D is only slightly better than the MOEA / D algorithm in the test cost value and the distribution of the function. The reason is that the two-

objective optimization problem using the MOEA / D algorithm has been able to get enough good pareto front.

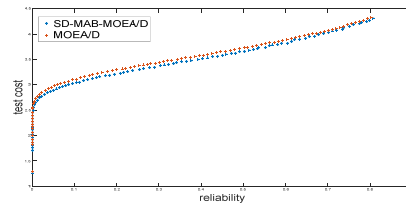


FIGURE II. SD-MAB-MOEA/D AND MOEA/D USED ON 16-MODULES SERIES AND PARALLEL SYSTEM WITH BIO-OBJECTIVES

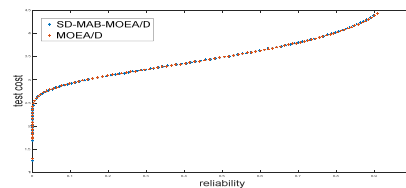


FIGURE III. SD-MAB-MOEA/D AND MOEA/D USED ON 16-MODULES SERIES AND PARALLEL SYSTEM WITH BIO-OBJECTIVES

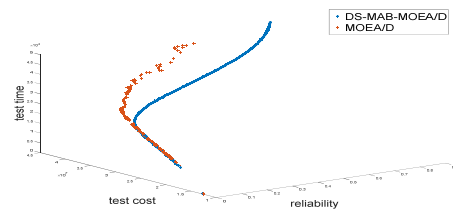


FIGURE IV. SD-MAB-MOEA/D AND MOEA/D USED ON 16-MODULES SERIES AND PARALLEL SYSTEM WITH TRI-OBJECTIVES

From Fig. IV and Fig. V, we can see that SD-MAB-MOEA / D algorithm is obviously better the MOEA / D algorithm in both function fitness and pareto frontal distribution when the total resource consumption is added as the third objective to solve, as seen in Table 1, 86.7% of the pareto solutions obtained by the SD-MAB-MOEA / D algorithm outperformed the pareto solutions obtained by the MOEA / D algorithm.

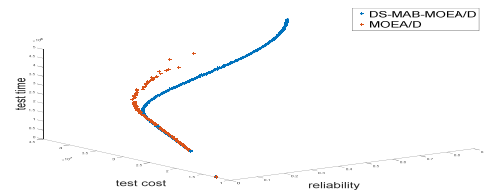


FIGURE V. SD-MAB-MOEA/D AND MOEA/D USED ON 16-MODULES SERIES AND PARALLEL SYSTEM WITH TRI-OBJECTIVES

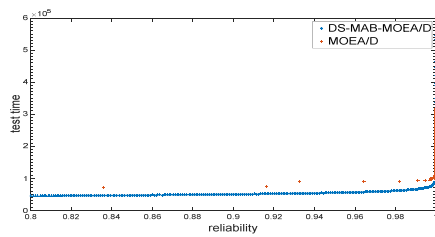


FIGURE VI. SD-MAB-MOEA/D AND MOEA/D USED ON 16-MODULES SERIES AND PARALLEL SYSTEM WITH TRI-OBJECTIVES WHEN RELIABILITY AS CONSTRAINT FUNCTION

When we choose the reliability greater than or equal to 0.8 as the limit, From Figure 6 we can see that using SD-MAB-MOEA / D to allocate test resources, the software reliability can reach 0.98 when the test time consumes 80000h. In this paper, we set the maximum workable time is 50000h, when the time consumption is 50000h, reliability achieves 0.915 by SD-MAB-MOEA / D algorithm. But the MOEA / D algorithm's pareto frontier converges to a reliability of 1 with 100000h time consuming. In Figure 7, we can also see that the MOEA / D algorithm's pareto frontier converges to a reliability of 1 with the resource consumption is  $8e + 7$ ; Therefore, the SD-MAB-MOEA / D algorithm can get satisfactory results with limited resources and reasonable the reliability requirements.

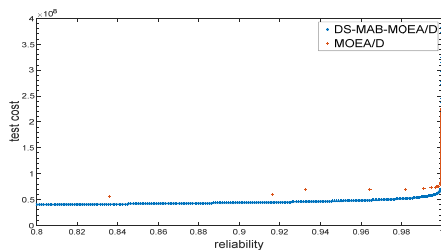


FIGURE VII. SD-MAB-MOEA/D AND MOEA/D USED ON 16-MODULES SERIES AND PARALLEL SYSTEM WITH TRI-OBJECTIVES WHEN RELIABILITY AS CONSTRAINT FUNCTION

## VI. CONCLUSION

In this paper, a hybrid DS-MAB-MOEA / D algorithm is proposed to solve the problem of multi-objectives problem in software test resource allocation. The MAB algorithm is used to solve the problem of using the single operator in the whole search process by using the traditional evolutionary algorithm. The problem of Pareto Front inequality in the process of test resource allocation is solved by DS algorithm. By applying the proposed algorithm to the complex 16-module series-parallel systems, the significant effectiveness of the algorithm is proved.

## ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China under Grants No. 61571346.

## REFERENCES

- [1] H. Ohtera and S. Yamada, "Optimal allocation and control problems for software testing resource," *IEEE Transactions on Reliability*, vol. 39, no. 2, pp. 171–176, 1990.
- [2] D. W. Coit and A. Smith, "Reliability optimization of series-parallel systems using a genetic algorithm," *IEEE Transactions on Reliability*, vol. 45, no. 2, pp. 254–260, 1996.
- [3] Y. S. Dai, M. Xie, K. L. Poh, and B. Yang, "Optimal testing-resource allocation with genetic algorithm for modular software systems," *The Journal of Systems and Software*, vol. 66, pp. 47–55, 2003.
- [4] R. H. Huo, S. Y. Kuo, and Y. P. Chang, "Efficient allocation of testing resources for software module testing based on the hyper-geometric distribution software reliability growth model," *IEEE Transactions on Reliability*, vol. 45, no. 4, pp. 541–549, 1996.
- [5] J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms," in *Genetic Algorithms and Their Applications: Proceedings of the First International Conference on Genetic Algorithms*, Pittsburgh, PA, USA, July 24–26, 1985, pp. 93–100.
- [6] J. D. Knowles and D. W. Corne, "Approximating the nondominated front using the pareto archived evolution strategy," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 2, pp. 149–172, 2000.
- [7] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.
- [8] Z. Wang, K. Tang, and X. Yao, "Multi-Objective Approaches to Optimal Testing Resource Allocation in Modular Software Systems," *IEEE Transactions on Reliability*, VOL. 59, NO. 3, pp. 563–575, 2010.
- [9] Y. S. Dai and X. L. Wang, "Optimal resource allocation on grid systems for maximizing service reliability using a genetic algorithms," *Reliability Engineering and System Safety*, vol. 91, no. 9, pp. 1071–1082, 2006.
- [10] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 398–417, Apr. 2009.
- [11] Q. Zhang, W. Liu, and H. Li, "The performance of a new version of MOEA/D on CEC09 unconstrained MOP test instances," in *Proc. IEEE Congr. Evol. Comput.*, May 2009, pp. 203–208.
- [12] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Mach. Learning*, vol. 47, nos. 2–3, pp. 235–256, May 2002.
- [13] H. Li and Q. Zhang, "Multiobjective optimization problems with complicated pareto sets, MOEA/D and NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 12, no. 2, pp. 284–302, Apr. 2009.
- [14] K. Li and S. Kwong, "Adaptive Operator Selection With Bandits for a Multiobjective Evolutionary Algorithm Based on Decomposition," *IEEE Trans. Evol. Comput.*, vol. 18, no. 1, pp. 114–130, Feb 2014.
- [15] Zhang Q, Li H, MOEA/D: a multi-objective evolutionary algorithm based on decomposition [J]. *IEEE Transactions on Evolutionary Computation*, 2007, 11(6): 712–731.
- [16] Liu Zhijun, Gao Yakui, Variable Neighborhood Multi - Objective Adaptive Differential Evolution Algorithm. *Control Theory & Applications*. vol. 31, no. 11, Nov. 2014.
- [17] Yu Shuaishuai, Fei Dong, Optimal Testing Resource Allocation for Modular Software Systems based on Multi-objective Evolutionary Algorithms with Effective Local Search Strategy. *IEEE Workshop on Memetic Computing (MC)*, 2013.