Multi-Objective Test Case Prioritization based on Epistatic Particle Swarm Optimization

Jiaze Sun\textsuperscript{a,b,*}, Jingmin Chen\textsuperscript{a}, and Gang Wang\textsuperscript{a}

\textsuperscript{a}School of Computer Science and Technology, Xi’an University of Posts and Telecommunications, Xi’an, 710121, China
\textsuperscript{b}Shaansi Key Laboratory of Network Data Analysis and Intelligent Processing, Xi’an University of Posts and Telecommunications, Xi’an, 710121, China

Abstract

To address the Multi-Objective Test Case Prioritization (MOTCP) problem, an Epistatic Particle Swarm Optimization (EPSO) algorithm is presented. The epistasis in biology is introduced into the new algorithm, and the particles are updated based on the crossover of Epistatic Test Case Segment (ETS) in the test case sequence. The average coverage percentage of program entity and effective execution time of the test case sequence are set as two objective fitness functions in EPSO. The experiment selects four typical open12 source projects as benchmark programs. We adopted Average Percentage of Branch Coverage (APBC) and Effective Execution Time (EET) as objective fitness. The four classical Java testing projects results show that the EPSO is more effective and more diverse than single-point PSO and order PSO. The EPSO algorithm efficiently solves the MOTCP problem by promoting early detection of software defects and reducing software testing costs in regression testing.

Keywords: multi-objective test case prioritization; epistatic test case segment; particle swarm optimization; average percentage of branch coverage

(Submitted on July 8, 2018; Revised on August 10, 2018; Accepted on September 12, 2018)

© 2018 Totem Publisher, Inc. All rights reserved.

1. Introduction

In the software evolution process, Test Case Prioritization (TCP) technology is a pragmatic and high-efficiency regression testing technique. To enhance the testing efficiency in the regression testing, the test cases are sorted by the test targets, which contributes to the early detection of defects and the reduction of testing costs [1]. In the sophisticated industrial software testing, many testing factors, such as testing cost, testing time, and code changes, should be considered in regression testing design [2]. The traditional optimization technique of reordering test cases for a single test objective has not been able to meet the multi-objective requirements of industrial software testing [3]. Therefore, Multi-Objective Test Case Prioritization (MOTCP) technology is significant in regression testing [4].

The MOTCP problem is a typical combinatorial optimization problem, which is NP-hard [5]. There are generally conflicting relations among multiple objectives in MOTCP. The classical deterministic approach for solving the MOTCP will result in the combinatorial explosion when the scale increases, so there is not an effective deterministic method to solve them in practice. Recent researches demonstrate that the NP-hard problems can be approximately and effectively settled by metaheuristic or heuristic optimization algorithms [6-7]. Literature [8] proposed a fast non-dominated sorting genetic algorithm II with elite strategy for MOTCP problem with fast speed and good convergence. However, the algorithm operations in genetic algorithm including selection, crossover, and mutation are relatively sophisticated, based on which the efficiency of the test suite prioritization is significantly decreasing with the increase in the scale of the population. Literature [5] presented a Multi-Objective PSO (MOPSO) for solving multi-objective software testing optimization problems, such as TCP and test case minimization [9-10], but the convergence of MOPSO and the diversity of solutions are unsatisfactory [11].

To enhance the efficiency of the test suite prioritization, literature [12] presented an improved particle swarm optimization with order crossover or single point crossover, which refers to the crossover operator of the genetic algorithm.
To strengthen the original characters of individuals in the iteration process, literature [13] introduced the concept of Epistasis Test Case Segment (ETS) of the test case sequence to the genetic algorithm (GA) to work out the problem of single objective TCP. Literature [14] introduced a pheromone updating method based on ETS into the Ant Colony Optimization (ACO) to increase convergence speed and global convergence ability.

Because the epistatic gene segment can express the original character of the prioritization solution and the epistatic gene segment exists in the test case sequence, the ETS should play a decisive role in the fitness function of the evolution algorithms [7, 13]. The Particle Swarm Optimization (PSO) algorithm is representative of SI algorithms. In this paper, we apply the epistatic crossover method to improve the PSO algorithm. Therefore, a novel PSO algorithm based on epistatic crossover operation, called Epistatic PSO (EPSO), is presented to work out the MOTCP problem. The epistasis crossover method is used to redesign the speed and location updated approaches to obtain wider particle distribution and better fitness values in the PSO algorithm. The main contributions of this paper are as follows:

1) A novel algorithm associating biological evolution with the Multi-Objective Optimization Problem (MOOP) is presented, where epistasis test case segment of the test case sequence is well established for the MOTCP.

2) The ETS guided crossover strategy is adopted for MOPSO algorithm, enhancing the effectiveness and diversity of the MOPSO algorithm.

3) The classical four benchmarks testing projects are carried out to quantitatively evaluate the effectiveness and diversity of the proposed novel algorithm. The empirical experiments verified that the proposed novel algorithm is more excellent than the order PSO and the single-point PSO.

2. Multi-Objective Test Case Prioritization

2.1. Test Case Prioritization

To obtain the optimal test case prioritizations that can accomplish the testing targets as soon as possible, the TCP technique is diffusely adopted in regression testing [15]. The MOTCP problem is formalized as follows [16]:

Given: a test suite, $S$, the full permutations set of $S$, $FS$; the set of $m$ objectives, $f_i$, $i=1, 2, \ldots, m$, which are from $FS$ to real number. The MOTCP problem is to obtain a Pareto optimal permutation set $S^* \subset FS$ in the case of $m$ objectives.

In the MOOP, a non-dominated, also called Pareto optimal, solution set is often obtained [17]. We suppose that the smaller value of the fitness function is better and there exists two permutations $S_i, S_j \subset FS$. If at least one objective meets $f_i(S_i) < f_i(S_j)$ and another objective function meets $f_j(S_i) \geq f_j(S_j)$, $i, j=1, 2, \ldots, m$, then $S_i$ and $S_j$ are non-dominated. Otherwise, if the objective functions meet $f_i(S_i) < f_i(S_j)$ and $f_j(S_i) \leq f_j(S_j)$, $i, j=1, 2, \ldots, m$, permutation $S_i$ dominates permutation $S_j$.

2.2. Epistasis Test Case Segment

Because epistasis dominance can depict the interaction relations between genes [2], it is introduced into the GA encoding to improve the evolution process [14]. Based on the binary or real number encoding, the individuals with some values in the special gene locations can usually fulfill a better fitness than the common. Therefore, those rules should be applied to search the essential gene locations. There are epistatic gene segments in the test case sequences, called Epistasis Test Case Segments (ETS), and the definition of ETS [7, 13] in the test case sequence is depicted as follows:

Given: a test suite, $S$, a full permutation of the test case. The ETS is a subsegment of full permutation that first reaches the best respective objective fitness. The subsegment is represented by the index range from the first index of the permutation array to the last index. Because ETS is defined based on multiple objectives, the optimal solution is Pareto.

2.3. Multi-Objective Fitness Functions

The testing targets are various with the varying requirements in the actual testing project. Regression testing is always limited by the testing adequacy and testing time, so this paper chooses the coverage-based metric, which is to be maximized, and the execution time, which is to be minimized as the optimization objective. Coverage-based metrics have many kinds,
such as branch coverage, decision coverage, and path coverage. For instance, because of the branch coverage of every test case, the Average Percentage of Branch Coverage (APBC) is defined as formula 1 to evaluate the effectiveness of test case ordering.

\[
APBC = 1 - \frac{T_1 + T_2 + \cdots + T_m}{nm} + \frac{1}{2n}
\]

In Equation (1), \(n\) is the number of test cases, \(m\) is the number of branches in testing software, and \(T_i\) represents the priority sequence number of the execution test case that first covers the branch \(i\). The Effective Execution Time (EET) [14] is another broadly adopted constraint optimization objective to evaluate the time cost of the regression testing, stated as Equation (2):

\[
EET = \sum_{i=1}^{m} E_i
\]

Where \(m_i\) is the number of test cases that firstly fulfils the highest branch coverage in the test case prioritization sequence and \(E_i\) is the execution time of test case \(i\).

3. EPSO for MOTCP

3.1. Modularization Quality

The PSO algorithm is a swarm intelligence algorithm stimulated by the behaviours of bird flocking for optimal location through the interaction of plenty particles. The PSO algorithm is simplified and justified to have effective optimization performance. PSO has been successfully applied to single-objective and multi-objective optimization problems. The objective function comparison considers Pareto dominance relation when the locations of the particles are updating, and non-dominated solutions are saved to close to the Pareto front in the multi-objective search space. By guiding the iteration operations, the Epistatic Test Case Segment is introduced to improve the effectiveness and efficiency of PSO for the MOTCP problem. The basic processing of the EPSO algorithm for the MOTCP problem is as follows:

<table>
<thead>
<tr>
<th>Algorithm 1 The basic flow of EPSO algorithm for MOTCP</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong> the full permutations set of regressing test cases; multi-objective functions; the max number of iterations.</td>
</tr>
<tr>
<td><strong>Output:</strong> the Pareto optimal permutation set.</td>
</tr>
<tr>
<td>1. Initialize the swarm and external archive.</td>
</tr>
<tr>
<td>2. for each particle (l_i) in the swarm do:</td>
</tr>
<tr>
<td>3. update the particle personal best location (P_i)</td>
</tr>
<tr>
<td>4. select a particle from the external archive as the global best particle (P_g)</td>
</tr>
<tr>
<td>5. update the velocity and the location of the particle (l_i) based on the (P_i) and (P_g).</td>
</tr>
<tr>
<td>6. update the external archive and calculate the fitness for each particle based on the multi-objective functions.</td>
</tr>
<tr>
<td>7. If the max iteration number is not satisfied, then go to step 2.</td>
</tr>
<tr>
<td>8. Report the results of external archive, that is the Pareto optimal permutation set</td>
</tr>
<tr>
<td>9. end</td>
</tr>
</tbody>
</table>

When the PSO algorithm solves the multi-objective problems, there are three key algorithmic design aspects:

1) Encoding and initialization of particle: How to encode the particle on behalf of the permutation of the test case set?

2) Selection of the best particle: How to choose the best particle from the external solutions archive set that are all non-dominated with each other? How to update the particles in the external archive to keep the representations of the previous swarm searching history?

3) Updating of the new particle location: How to promote diversity through the updating of locations to create new particle location?
3.2. Encoding and Initialization of Particle

Suppose the sorted test case set has n test cases, each test case is numbered with an integer between 1 and n, and the permutation of test case numbers is used as the particle coding. Each particle represents a sorting scheme of test case executions. The particle swarm represents the feasible solution set for the MOTCP problem, and each particle represents a sort scheme. The current location and velocity of every particle are initialized with random permutation of the regressing test cases. The current location of the i\textsuperscript{th} particle \(p_i\) is set as the individual local best location \(p_i\). The multi-objective fitnesses of each particle are calculated. External archive is initialized with the non-dominant permutation set from the swarm, and the global best \(p_g\) is randomly chosen from the external archive.

3.3. Choosing the Best Particle and Updating the External Achieve

When the PSO is used to solve the MOTCP problem, the choosing strategy of the global best particle and the local best particle plays an important role in the particle flights. Compare the current particle location with the history personal best location in the objective function values. If the present particle can dominate or is non-dominated about the history personal best location, then the present particle is set as the particle personal best location. Otherwise, the personal best location stays the same. To keep the ergodicity of the solution in the searching space, the global best particle is randomly selected from the external archive.

The external archive is introduced to store the good particles in the previous swarm. When the new particle can dominate any of the particles within the external archive or is non-dominated about the contents of the archive, it can enter the external archive. It is obvious that the dominated particle in the external archive must be deleted. When the particle number in the external archive exceeds the bounded scale, the redundant particle will be moved from the archive to keep the size of the external archive bounded.

3.4. Updating of Particle Velocity and Location

As the core operation of evolutionary mechanism of genetic algorithm, crossover plays a very crucial role in the process of the evolutionary optimization algorithm. In the MOTCP problem, the crossover operation is introduced into the updating of the particle location in particle swarm optimization. Literature [11] adopted single point crossover and order crossover to solve the MOTCP, which is called single point crossover PSO (SPSO) and order crossover PSO (OPSO). When the single crossover point is outside the epistasis test case segment, the adaptability of the offspring will be like some parent, so the single crossover mostly decreases the convergence speed of the algorithm. The offspring particles is likely to only partly succeed the good gene of the parent to the epistasis test case segment in order crossover iterations. The crossover operation has many kinds. When the crossover is a single point crossover, in the construction offspring individual, the genes of the epistasis test case may come from some part of the back genes from the father individual that is not exactly the internal genes from the part of the father individual epistasis test case. In this way of crossing, it may just inherit a part of the good genes from the father individual. The offspring ETS genes using order crossover mostly are derived from the later genes, not from the whole ETS in parents. Therefore, the offspring is unlikely to succeed to better gene segments within ETS from the parent [11].

Because there exists an epistasis genomic segment in the test case sequence to express the original character of the test case executing sequences, the epistasis test case segment has a decisive impact on the objective function fitness values. Therefore, the epistasis crossover operation is introduced to generate the offspring in the PSO for the MOTCP problem. The basic epistasis crossover operation is shown in Algorithm 2:

**Algorithm 2** The basic flow of epistasis crossover operation.

**Input:** parent individuals \(A_1\) and \(A_2\).

**Output:** the offspring individuals \(B_1\) and \(B_2\).

1. Crossing points \(k_i\) and \(k_j\) are randomly generated, \(k_i, k_j \in \{0, \ldots, n\}\), where \(n\) is the test case number in the regressing test.
2. The genes between two crossing points on the parent individual \(A_1\) is directly copied into the corresponding location of the offspring \(B_1\) as a gene segment.
3. Traversing the parent individual \(A_1\) and excluding the gene between the two crossing points on the parental individual \(A_0\), the remaining gene sequence is \(S_1\).
4. The genes in \(S_1\) sequentially fill the locations from the first empty location to the end on the offspring \(B_1\), and then the offspring individual \(B_1\) is completely formed.
5. The offspring individual \(B_2\) is similarly generated to the offspring individual \(B_1\).
6. end
The particle location is changed according to its personal best location and the swarm global location based on the epistasis crossover. The updating equations of the velocity and location are as follows:

\[ v_i(k + 1) = p_i(k) \ominus p_g(k) \]  
\[ v_i(k + 1) = v_i(k + 1) \oplus v_i(k) \]  
\[ x_i(k + 1) = x_i(k) \oplus v_i(k + 1) \]

\( k \) denotes the number of iterations, and \( \ominus \) denotes the epistasis crossover operation.

For the two generations of offspring, if two sub-generation individuals do not satisfy the non-dominated relationship, the better offspring will be selected as the crossover result. If two descendants satisfy the non-dominated relationship, choose one of the offspring individuals as the cross result. If the offspring generated by the crossover operation meet the dominated relationship, the dominating offspring will be chosen as the epistasis crossover result. Otherwise, if those of the offspring satisfy the non-dominated relationship, one of the offspring can be selected as the crossover result.

In the \( k^{th} \) updating iteration, the \( i^{th} \) particle crosses his own history optimal solution \( p_i(k) \) and the swarm global optimal solution \( p_g(k) \) by applying the epistasis crossover operation to generate the velocity increment \( v_i(k + 1) \) at step \( k+1 \), as shown in Equation (3). The velocity \( v_i(k + 1) \) of the \( i \)-th particle at step \( k+1 \) is obtained through epistasis crossing of the velocity increment \( v_i(k + 1) \) and the velocity \( v_i(k) \), as shown in Equation (4). Consequently, the location \( x_i(k + 1) \) of the \( i^{th} \) particle at step \( k+1 \) is calculated through epistasis crossing of the velocity \( v_i(k + 1) \) at step \( k+1 \) and the velocity \( v_i(k) \) at step \( k \), as shown in Equation (5).

4. Empirical Evaluation

The primary goal of empirical study is to search a better algorithm for multi-objective test case prioritization. We implement the algorithm described in Section 3 and measure its effectiveness and stability. We intend to investigate the following questions in this paper:

- **RQ1**: How does the solution distribution of the EPSO differ under different iteration times?
- **RQ2**: How do different swarm sizes affect the Pareto optimal solutions of the EPSO?
- **RQ3**: How does EPSO compare with SPSO and OPSO in terms of effectiveness and diversity?

4.1. Experiment Setup

In the experiments, we used four classical open sources Java projects [18] that have been widely used in software testing research, and they are described in Table 1. All experiments were performed on Windows 7 32 bit using a 2.10 GHz Intel Core 3 processor, 6 GB of main memory, and JDK1.7.0_45.

<table>
<thead>
<tr>
<th>Test program</th>
<th>Number of branches</th>
<th>Test program information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jasmine-maven</td>
<td>381</td>
<td>The Maven plug-in of the JavaScript unit test framework Jasmine</td>
</tr>
<tr>
<td>Jeep-simple</td>
<td>431</td>
<td>Java library for parsing the command line options</td>
</tr>
<tr>
<td>La4j</td>
<td>145</td>
<td>The library of linear algebra provides the primitives and algorithms of linear algebra</td>
</tr>
<tr>
<td>Scribe-java</td>
<td>115</td>
<td>An open source OAuth library developed with Java</td>
</tr>
</tbody>
</table>

4.2. Experiments and Results

4.2.1. Solution Distribution for Different Iteration Times

To observe the optimal solution distribution of the EPSO algorithm under the different iteration times, the fitness values of APBC and EET in the first 200 iterations of the four test objects were recorded. The solution distribution under different
iteration times is shown in Figure 1. In this figure, the particle swarm size is 100, and the number of iteration is 50, 100, 150, and 200. The horizontal axis denotes the Effective Execution Time (EET) of the test case sequence, and the time unit is milliseconds (MS). The vertical axis indicates the Average Percentage of Branch Coverage (APBC) of test case. Every point in the picture represents an optimal non-dominated solution. From Figure 1, we can see that the non-dominated solution set is getting better and better with the iteration times increasing. Moreover, when the iteration times exceed 150 times, the non-dominated solution set no longer changes significantly, and the non-dominated solution tends towards the optimal set.

**Figure 1. Solution distribution under different iteration times**

### 4.2.2. Solution Distribution for Different Swarm Sizes

When the iteration number of the EPSO algorithm is 100, the distribution of the non-dominated solution set for four test projects with different swarm sizes such as 30, 50, 80, 100, and 150 is shown in Figure 2. From Figure 2, we can see that with an increase in particle swarm size, the solution points of the test projects Jasmine, La4j, and Scribe-java shift to the upper left area of the coordinate system, which means that the non-dominated set will become better. As the swarm size upgrading dose helps diversify more and better new particles, the diversity of the swarm increases quickly. However, for test project Jopt-simple, the non-dominated set did not change significantly. We find that the EPSO algorithm obtains the optimal solutions when the swarm size is 30. Consequently, the upgrading of the swarm size is limited to a certain extent under certain iteration times.

**Figure 2. Solution distribution under different swarm sizes**

### 4.2.3. Solution Distribution of Different Algorithms

The SPSO algorithm, OPSO algorithm, and EPSO algorithm were used to solve MOTCP problem in our experiments, and
we compare them using the solution fitness values distribution of four test projects. The swarm size and iteration times are all 100. The distribution of the non-dominated solution set obtained by 30 tests of the SPSO algorithm, OPSO algorithm, and EPSO algorithm for four test projects are shown in Figure 3. As shown in Figure 3, the EPSO algorithm can generally get higher APBC values than the SPSO algorithm and OPSO algorithm under the same effective execution time, which shows that the EPSO algorithm is more effective. On the other hand, the spread range of the solution obtained by the EPSO algorithm is also wider than SPSO and OPSO, which shows that the diversity searchability of the EPSO algorithm is better.

Figure 3. Solution distribution of different algorithms

In summary, the EPSO algorithm can not only obtain a more effective solution than the SPSO algorithm and OPSO algorithm, but also ensure the diversity of the non-dominated solutions, which provides an efficient approach for the MOTCP problem.

5. Conclusion

To address the MOTCP problem, the epistatic theory is introduced into the traditional PSO algorithm, and the epistasis test case section is used to guide the test case prioritization. Experimental results show that the EPSO algorithm for the MOTCP is superior to the SPSO algorithm and the OPSO algorithm. The optimal non-dominated solutions have a wide range of distribution and higher fitness values. In future work, we intend to explore the following aspects. First, we intend to use more test projects of different scales to analyze the advantages and disadvantages of different techniques for the MOTCP. Second, we intend to use more evaluation objectives to measure the effectiveness of prioritization results. Finally, we plan to find more gene segments that reflect the nature of the test case sequences.

Acknowledgment

This work is supported by the special fund for Key Discipline Construction of General Institutions of Higher Learning from Shaanxi Province, the Industrial Research Project of Shaanxi Province (2017GY-076), and the Innovation Fund of Xi’an University of Posts and Telecommunications (CXJJ2017064).

References

Jiaze Sun received his Ph.D. from the School of Information Science and Technology at Northwest University in 2015. He is currently an associate professor in the School of Computer Science and Technology at Xi’an University of Posts and Telecommunications (XUPT). His interests include swarm intelligence optimization algorithms, software testing, and data mining.

Jingmin Chen is a Master’s student in the School of Computer Science and Technology at Xi’an University of Posts and Telecommunications. Her research interests include swarm intelligence optimization algorithm and software testing.

Gang Wang is a Master’s student in the School of Computer Science and Technology at Xi’an University of Posts and Telecommunications. His research interests include data mining and software testing.