A Combination Test Suite Generation Method Based on Adaptive Simulated Annealing Genetic Algorithm for Software Product Line Testing

Xin Zhao, Yu Li, Lianhui Liu, Jun Zheng, Yan Liu and Xiangdong He

ABSTRACT

A Based-Similarity Combination SPL Test Suite Generation Method can be used to compute coverage, replace poor test cases and generate high coverage test suite. Improve the way to compute mutation and make test suite generation adapt to the current coverage from great mutation, the optimization procedure can be more efficient. Meanwhile, adaptive simulated annealing algorithm combined with simplified GA algorithm, which can ensure local optimization accuracy and take the whole situation into global searching space accordingly, is a good choice for combinatorial optimization problems. Experiments with 6 feature model from SPLOT shows that the new test suite generation method can achieve smaller-size test suite with higher coverage.

INTRODUCTION

Most of the software errors are always caused by the interaction between the Software Product Line (SPL), e.g. [1], features in the software product line testing. And combinatorial testing for software product line features can help find features interaction problem early. For large-scale combination of SPL test, designers' subjectivity will cause some test case generations invalid or redundant, e.g. [2][3] In fact, it is impossible to execute the whole test suite including all combinations of all SPL features in a limited time.

Test suite generation is the key of software testing and the focus of concern in the study of combinatorial testing. SPL combination test suite generation is a typical combinatorial optimization problem. Many algorithms have been proposed to deal with combinatorial optimization problem, such as genetic algorithm, annealing algorithm, particle swarm optimization algorithm. Based on the SPL feature model, the simplified genetic algorithm can help generate test suite with
certain coverage criteria in the search space of effective set, e.g. [4]). In spite of
good local convergence, GA technology sometimes just gets local optimum
solution for complex combination optimization problems and multi-modal
problems, e.g.[9]. Adaptive simulated annealing algorithm can be very good control
of the overall situation, which can help find a better solution close to whole goal
state with stochastic search in the search space, after the optimizing direction
selective according to the given probability combined with the current temperature.
Proposed in the reference, e.g. [5], which gives an optimization method to generate a
combination test suite for SPL testing, the method can generate a test suite with
higher coverage compared with the random search method. But the condition that
the variability equal to 1 is to be constant is adverse to the searching efficiency in
the process of composition optimization.

This paper introduces a new method which is based on the similarity measure
between test cases, and combined with the adaptive simulated annealing algorithm
and simple genetic algorithm, the improvement is for the variation method in the
process of generating test suite. The new test suite generation method is based on
Adaptive Simulated Annealing Genetic Algorithm (ASAGA) which can be
applied in improving the method of mutation. At the same time with self-adaptive
adjusted mutation function instead of constant mutation, the new method helps
increase local optimization precision and optimization efficiency.

SPL TEST SUITE GENERATION BASED ON ASAGA

Combination Testing Feature Model Based on SPL

Combinatorial interaction testing technique can be used to locate most errors
caused by the interaction between features in SPL feature configuration testing, e.g.
[3]. Representative SPL feature configuration is of small-scale so that most
important test cases can be taken precedence on execution and cost-effective and
timely achievement testing is possible. SPL feature model with variability and
scalability is based on the SPL test requirements in reference, e.g. [4]. And many
combination testing methods have been applied to SPL feature models so far, e.g.
[5][6]. Considering search-based methods to search a good test suite in the search
space of the valid set which is generated from SPL feature models and constraints
between features, this paper shows SPL feature model definition as formula (1).

\[(fm, c) \iff \begin{cases} fm = \{ fm_1, fm_2, \ldots, fm_n \} \\ C = \{ c_1, c_2, \ldots, c_n \} \end{cases} \]  \tag{1}

The precondition for the combinatorial test suite generation method for SPL feature
model is a valid set satisfying all constraints between features, based on which this
paper emphasizes on an approximate optimization method to generate a test suite
randomly first and optimize the test suite continuously until the termination
condition is reached.
T-wise Coverage Criteria and Similarity Measure

T-wise coverage criteria

The performance of a test suite can be presented by its coverage rate directly. A test suite with high coverage rate covers most representative test cases, and these test cases can help testers to find software errors timely[8]. A reduced test suite with high coverage is small sized and can sufficiently cover the valid set space. With many testing coverage calculation methods proposed to measure the generated test suite, T-wise coverage calculation method is adopted in this paper which is raised by Christopher Henard in the paper, e.g. [4] and considers all the possible interactions with respect to the constraints of the SPL feature model[11]. Varying linearly with test case similarity, T-wise coverage of a test suite, is defined as the ratio as formula 2, where TCS is the generated test suite and valid_TCS is the valid set. And t=2 in this paper as formula (2).

\[
\text{Coverage}_{t}(\text{TCS}, \text{Valid}_t \text{TCS}) = \frac{\#(\text{Combset}_t(\text{TCS}) \cap \text{Combset}_t(\text{Valid}_t \text{TCS}))}{C_t'}
\]  

The fitness function based on similarity measure

Commonly used in composition researching area, Similarity measure can simplify the solving of the test suite coverage rate[11]. The similarity measure between test cases can help to effectively evaluate the coverage of test suite without calculating the t combinations. The greater distance between the test cases is, the better test suite coverage is, e.g. [5][6][11].

Test suite similarity with each other is the core of coverage calculation and simplifying calculating the coverage, it can be used as the evaluation criteria for selecting test cases in test suite generation, e.g. [11]such as JACCARD distance as formula (3), which shows the distance between sets and can be used to measure the similarity of SPL feature model [5].

\[
\text{Distance}_{jaccard}(\text{TC}_i, \text{TC}_j) = 1 - \frac{\#(\text{TC}_i \cap \text{TC}_j)}{\#(\text{TC}_i \cup \text{TC}_j)}
\]  

And based on similarity measurement, the further definition is for the fitness function to evaluate the test suite as formula (4).

\[
\text{Fitness}(\text{TC}_i) = \sum_{i<j} \text{Dissimilarity}(\text{TC}_i, \text{TC}_j)
\]

Test Suite Generation Based on ASAGA

The fitness function based on similarity measure

Continuous turnover in test suite will help obtain a approximate optimal solution. Measured by the T-wise coverage criterion, the validity and better performance of the final test suite can be verified.

Simplified genetic algorithm mutation method is mainly considering mutation factors in the process of test suite optimization. After internal sorting of the test suite, only the worst one is out and a new replaced. And only a better fitness
function, the replacement can be admitted. Based on the above method, two improvements can be taken into consideration in this paper.

(1) Combined with the adaptive simulated annealing algorithm, the new test suite without better fitness can be accepted for next optimizing in a small probability.

(2) Rather than constant mutation number of test cases in each optimization. The mutation number of test cases, new mutation function is introduced in this paper and with adaptively convergence to an ideal state as the test suite approaches better coverage.

**Test suite mutation based on ASAGA**

The adaptive combination test suite mutation method is presented in this paper. On the one hand, when updating test suite, it is worth considering to determine in a probability whether the new test suite with poor fitness function can replace the current one to improve optimization accuracy of test suite. On the other, picking the right mutation founded on the current test suite, the new method to get new test suite is more efficient.

(1) To optimize the random generated test suite constantly, whether new test cases generated randomly can replace poor test cases sorted by internal optimization depends on the obvious improvement of the test suite coverage after replacing. The replaced test suite can lead to a more high-quality test suite until stopping conditions are met, but can’t lead to the best or the approximate optimum test suite. The other hand, even with the slight performance of the test suite hit temporarily, replacement may lead to a test suite with better coverage.

(2) And despite of decreased performance after replacement slightly, the optimization algorithm for test suite can keep with whole situation beyond the only optimization solution to increase chance of getting the best test suite in a limited time. So, considering temporary performance sight degradation of the test suite on a very small scale, increase search space then with searching efficiency improved the global optimum solution can be found rather than a local optimum solution. Feasibility Assessment of the new solution of the poor coverage using simulated annealing algorithm, can avoid the local optimal solution, e.g. [10].

The simulated annealing algorithm proposed by S. Kirkpatrick, derived from the principle of solid annealing, e.g. [10], can be applied in combination of optimization problems. Starting at some initial temperature and with the decrease of temperature, the probability-sudden-jump character of SA can help the objective function gain the global optimal solution in ideal time as formula (5). And in this paper, the classical simulated annealing algorithm is used as the cooling method during optimization, e.g. [10].

\[ T(\lambda) = \frac{T_0}{\log(1 + \lambda)} \]  

(5)

Using the simplified genetic algorithm, only the new solution is better than the current, can the new solution replace the current solutions, but in this way the final solution can only converge to a local optimum. Metropolis criterion of SA is improved with similarity measurement of test suite as follows formula (6), to decide whether the new solutions can replace the old, e.g. [10].
\[
p(TCS \rightarrow TCS) = \begin{cases} 
1; \text{if } (\text{fitness}(TCS') \geq \text{fitness}(TCS)) \\
\frac{e^{(\text{fitness}(TCS') - \text{fitness}(TCS))/T(\lambda)}}{1}; \text{else}
\end{cases}
\]

(6)

So, two aspects deserve consideration in accepting a new solution. 1) By measuring the similarity between test cases, a new solution with better fitness function can be accepted. 2) Comprehensively considering test suite similarity and the current temperature, a new solution without better fitness can be accepted as search seed in a certain probability. And the initial temperature in experiments is \(T_0 = 100000.00\).

After internal sorting optimization, test cases with worse performance should go out of test suite. In fact, according to the T-wise coverage criteria, in a certain sized test suite arriving at certain coverage, some test cases are redundant and do not make a difference on the coverage. There is no improvement for test suite coverage with test case TC17 and TC18, TC19, TC20, TC21, and these redundant test cases can be eliminated to simplify the test suite. Compared with the simplified genetic algorithm whose mutation is constant to be 1, improving mutation by cutting more redundant test cases, and the new mutation method find the optimal solution faster.

For larger scale SPL feature model, a certain scale of test suite is needed to ensure good coverage and sight update doesn't work for the coverage. This actually reduces updating efficiency of the test suite, thus optimizing will require more time. And only a test case updated, can't test suite coverage become better if considering the similarity between test cases and the final test suite may be not only approximate global optimal. Therefore, in the process of updating the test suite, replacing more test cases, greater mutation can improve the optimization efficiency.

But more test cases eliminated may not mean better coverage. And more important is feasibility while keeping the efficiency. Considering the unpredictability of the initial test suite generated randomly, a significant degree of update will corrupt the current test suite, thus defeating the purpose of mutation and optimization. That means excessive mutation will increase searching cost and decrease the performance of the algorithms obviously. So, as follows formula (7), mutation number of test suite should be rolled back to a stable value to ensure smooth optimization in the post-optimization procedure.

\[
p_{\lambda \_mutate +1} = p_{\lambda \_mutate} \times e^{\text{con}_\lambda \_mutate}
\]

(7)

In summary, mutation of test suite optimization will affect the efficiency and performance of the algorithm, so it is necessary to select a suitable mutation probability. Mutation improvement includes two points. First, what is the right mutation. Second, how to find right mutation with all the factors that affect the efficiency of updating test suite. In this paper, the mutation function is shown as the formula 7. The ASAGA mutation method is completely shown in Fig. 1.
EMPIRICAL STUDY

Experiment Data and Experiment Schemes

Select 6 feature models on SPLOT to experiment and evaluate the algorithm in this paper. 3 small SPL feature models (aircraft-fm, Cellphone-fm, Graph-product-line-fm), 2 feature models with medium (smart_home_FM, arcade_game_pl_fm) and a feature model with larger scale (Eshop-fm) are given in Table 1.

Table 1. SPL Feature Models in the empirical study.

<table>
<thead>
<tr>
<th>SPL FM</th>
<th>features</th>
<th>valid pairs</th>
</tr>
</thead>
<tbody>
<tr>
<td>aircraft-fm</td>
<td>13</td>
<td>240</td>
</tr>
<tr>
<td>Graph-product-line-fm</td>
<td>20</td>
<td>499</td>
</tr>
<tr>
<td>Eshop-fm</td>
<td>287</td>
<td>147534</td>
</tr>
<tr>
<td>Cellphone-fm</td>
<td>11</td>
<td>151</td>
</tr>
<tr>
<td>smart_home_fm</td>
<td>35</td>
<td>1465</td>
</tr>
<tr>
<td>arcade_game_pl_fm</td>
<td>61</td>
<td>5209</td>
</tr>
</tbody>
</table>

The conducted experiments are performed on Windows with 8G RAM and Linux with 16G of RAM. Two experiment schemes for experiment results.

Scheme 1 Compare the average size of test suites with significant amount of coverage generated many times from the simplified genetic algorithm and which generated from the algorithm proposed in this paper.

Scheme 2 Compare the coverage and the average coverage of test suites with same size generated from the simplified genetic algorithm and the algorithm proposed in this paper.

Figure 1. TCS' mutation with ASAGA.
Experimental Results and Analysis

**Scheme 1** For this 3 SPL feature models (Cellphone-fm, smart-home-fm, arcade-game-pl-fm) the simplified GA algorithm and the algorithm in this paper can be used to generated test suites with small size and high coverage up to 100% according to 2-wise coverage criteria. The comparison of the TC’s generation with the simplified GA and the algorithm in this paper on the 3FMs after 30 times in the same time as Table 2.

<table>
<thead>
<tr>
<th>SPL FM</th>
<th>GA</th>
<th>ASAGA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cellphone-fm</td>
<td>7.8</td>
<td>7.5</td>
</tr>
<tr>
<td>smart_home-fm</td>
<td>13.4</td>
<td>13.23</td>
</tr>
<tr>
<td>arcade_game_pl_fm</td>
<td>41.26</td>
<td>39.19</td>
</tr>
</tbody>
</table>

The algorithm in this paper can help get a test suite with same coverage up to 100% and the test suite is smaller than the one generated from the simplified GA algorithm. Details show as Fig.2~Fig.4.

![Figure 2. Cellphone-fm’ generation Comparison.](image1)

![Figure 3. Smart-home-fm’ generation Comparison.](image2)

![Figure 4. Comparison of the arcade-game-pl-fm’ generation.](image3)
The comparison of the TC’s generation with the simplified algorithm and the algorithm as Table 3. And as Table 3 shows that test suite with same size generated from the algorithm ASAGA has higher coverage.

|                                 | aircraft-fm |                     | Graph-product-line-fm |
|                                 | Size=8      | Size=9              | Size=10                | Size=15                |
| ASAGA                           | 98.222      | 99.292              | 99.722                 | 99.218                 |

Table 3. Comparison of the TCs’ generation with GA on the 3 FMs.

| Eshop-fm                          |
|---|---|---|---|---|---|---|
| Size=70 | Size=75 | Size=80 | Size=85 | Size=90 | Size=95 | Size=100 |
| 99.997 | 99.998 | 99.998 | 99.999 | 99.999 | 99.999 | 100.000 |

CONCLUSION

The test suite generation method of SPL feature model that combines Adaptive Simulated Annealing Genetic Algorithm and Simplified Genetic Algorithm is presented in this paper, which is based on similarity measure to improve the searching efficiency and accuracy of the algorithm without the local optimum. Experiments performed on 6 feature models with different scale show that compared with the Simplified Genetic Algorithm, the Adaptive Simulated Annealing Genetic Algorithm is more intelligent, and the test suite with higher coverage is more compact.

For mutation calculation method proposed in this paper, and different parameters of SPL feature models, experimental results is different, there are two aspects considered in future work. Firstly, considering appropriate parameters of the feature model, enhance the algorithm searching ability. Secondly, using the coverage criteria with different strength, improve the optimization strategy in the global trend and local performance to get the test suite with higher coverage.

REFERENCES


