Optimised Realistic Test Input Generation

Mustafa Bozkurt and Mark Harman
{m.bozkurt,m.harman}@cs.ucl.ac.uk

CREST Centre, Department of Computer Science,
University College London.
Malet Place,
London WC1E 6BT, UK

Abstract. Generating realistic test data is a major problem for software testers. This problem is very severe for certain input types that are hard to generate automatically. Previously we proposed a novel automated solution to test data generation that exploits existing web services as sources of realistic test data. One of the issues for this research agenda is the ability to select/use services with high reliability and low cost. We formulate this as an optimisation problem and suggest the use of multi-objective optimisation approaches as a possible solution.

1 Introduction

In SOA, it is often possible to assume that there may exist many services that provide the same functionalities. The ability to chose a suitable service is one of the major advantages provided by SOA. This ability is provided by a well-known concept called Quality of Service (QoS) in SOA.

QoS requirements generally refer to non-functional qualities of services [2, 8]. According to Wan Ab. Rahman [8] the most common QoS requirements are service time, performance, reliability, execution price (price from here on), availability and security.

We have developed a tool called ATAM, that composes services to provide service mashups that are able to provide realistic test data to test other services. For example, to produce a real ISBN (one that corresponds to a real book) ATAM might compose a bookstore service with a source of likely author names. ATAM is able to generate arbitrary service compositions to find even more elaborate real-world test data.

Our motivation came from one of the case studies we used during the evaluation of ATAM [3]. In this case study, we generate ZIP codes by sequencing the Google search service along with two other services: a service, S_2, that finds IP addresses for domain names and another service, S_3, that provides the location information of IP addresses. The functionalities provided by these three services are very popular and there are many other web services and websites with these functionalities. We found 31 internet search websites, 13 websites providing the functionality of S_2 and 40 websites providing the functionality of S_3. Some of these websites such as Google, Bing and Yahoo already provide web services using REST and SOAP protocols.

In our case study scenario, assuming all the mentioned websites are also available as web services, our case study will have $31x13x40 = 16,120$ potential solutions for generating ZIP codes. This scenario is merely a simple illustrative case study. In a scenario where the solution consists of a 7-service sequence with only 15 candidate services from which to choose, the number of potential solutions would be $15^7 = 170,859,375$. Such search spaces are to be expected when using ATAM. ATAM's ability to discover all possible service compositions for generating the required test input, will tend to increase the size of candidate service set.

The current version of ATAM uses the discovered services in the order from the search results. As a result, ATAM does not always use the most suitable solution. The two main problems surrounding the service selection were:

1. ATAM requires the ability to select and use services with higher reliability.
2. ATAM requires the ability to select and use services with low price in order to reduce cost of test data generation.

Since ATAM uses services with semantic descriptions that include QoS parameters, the obvious solution to the two problems stated above is the use of QoS parameters in our test data generation process. We recognized this problem as an optimisation problem. In the test data generation process we want to use services with higher reliability to maximise the reliability while use low cost services to minimise the cost of generating test data.

The advantages of the proposed application of multi-objective optimisation to our test data generation approach are:

1. Reduced cost: Ability to select and use services with lower cost.
2. Increased efficiency: Ability to select and use services with high reliability.

The rest of this paper is organised as follows. Section 2 discusses existing test data generation approaches that use optimisation and reviews ATAM and the test generation process it uses in detail. Section 3 explains the proposed multi-objective optimisation for our approach. Section 4 concludes the paper.

2 Background

2.1 Test Data Generation and Optimisation

Multi-objective optimisation is not new to test data generation domain. Lakhotia et al. [4], Oster and Saglietti [5] and Pinto and Vergilio [6] already proposed multi-objective test data generation. These approaches mainly focus on structural coverage as the main objective and use additional objectives such as execution time, memory consumption and size of test set. There are other approaches such as Sagarna and Yao [7] where branch coverage is formulated as constraint optimisation problem. Similarly, Alshahwan and Harman [1] used a single objective search based testing approach to generate test data to achieve branch coverage of server side code in PHP. The use of multi-objective optimisations in test case selection has also been proposed. For example, Yoo and Harman [9] used objectives such as code coverage and execution cost in test case selection.

2.2 ATAM Service-Oriented Test Data Generator

ATAM is capable of generating realistic test data, such as ISBNs and ZIP codes, using existing semantic web services [3]. ATAM is capable of generating realistic test data for any semantic system. In order to generate test data, ATAM searches for services that can provide the data needed as test input. ATAM minimises the manual input required by using three benefits provided by semantic SOA:

1. Ability to discover services automatically.
2. Ability to invoke services dynamically.
3. Ability to discover relations between service inputs and outputs using ontological descriptions.

ATAM searches for services providing the required data regardless of their inputs. As a result, service search queries may return multiple individual services with different inputs but provide the required test inputs. In these cases ATAM provides the tester with multiple possible solutions.

In some cases, the required test input might be generated using multiple services. This is because ATAM not only searches for services that provide the required test input, but also performs the search process for previously discovered services (which provide the required test input) in order to automate the test data generation process. As a result, in situations where input data for the discovered services can be provided by other services these input-output dependent services form a 'service sequence'.

A search result might contain many services with the same input(s) and the output. In such a situation, performing the discovery process for all these discovered services requires running the same query several times. This is an unwanted side effect that increases the time and cost of service discovery. In order to prevent this, ATAM performs a grouping process where services with same inputs and outputs are grouped together. The definition of each service group is used in service sequences rather than the actual services themselves. As a result, a sequence forms a template solution for all potential solutions that can be constructed using one service from each of the service groups in this sequence.

ATAM is able to discover and provide many of the possible ways that a test input can be generated using the existing services. Some of the issues regarding having multiple options can be formulated as optimisation objectives for a multi-objective SBSE approach to optimised realistic test input generation.

3 Multi-objective Optimisation

In this section, we explain the two necessary elements that are required in the multi-objective optimisation approach we propose here: the objective function and genetic operators.

3.1 Objective Function

We introduce two different functions for the two QoS parameters we intend to use in the next version of ATAM: price and reliability. The reason for having two

different objective functions is caused by our perception of combined reliability in service sequences.

The objective function for the price parameter is straightforward. The function is the sum of the costs of all services in a required service sequence. In our approach we considered price as an objective rather than a constraint in order to allow the tester to explore all the possible solutions on the pareto-optimal front. The following is introduced as the objective function for minimising total cost of test data generation:

$$Minimize \sum_{i=1}^{n} p_{s_i}$$

where n is the total number of services in the selected sequence and p_{s_i} is price of the ith service.

The objective function for the reliability is not as straightforward. The reliability of a service sequence is as high as the combined reliability of services used. We introduced the concept of 'combined reliability' because the reliability of an individual service is not solely defined by the behaviour of that service in isolation. Rather, the reliability of a service S also depends on the reliability of the service sequence that generates the necessary input for S. Figure 1 illustrates an example solution and explains necessary concepts in reliability calculations.

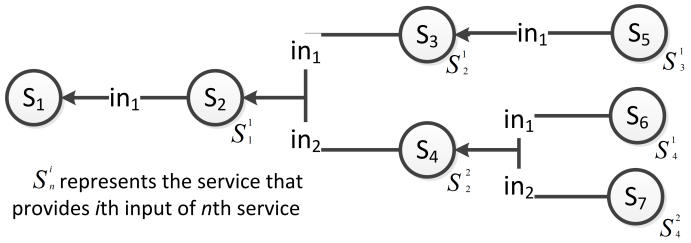

Fig. 1: Example service sequence scenario for generating realistic test data generation using services. In the figure, each node represents a service used in this sequence. In this scenario service S_1 is the service that provides the required test input. The services on the other end, such as S_5, S_6 and S_7, are services that either require tester input or automatically generated input using a predefined algorithm. Each edge in the figure represents an input required by the service that targeted by the edge. The source of the edge is the service that provides the required input. For example, in this scenario, the input for S_1 is provided by S_2.

We combine the two reliability parameters and formulated combined reliability (cr) of a service as:

$$cr(S_n) = r_{S_n} \times ir(S_n)$$

where $cr(S_n)$ is the combined reliability and r_{S_n} is the reliability of the service S_n and $ir(S_n)$ is the reliability function that calculates the combined reliability of the services that provide inputs for S_n.

The reliability calculation for inputs ($ir(S_n)$) varies based on the number of services providing the input. This is because in our approach a service in the sequence can get the required inputs in two possible manner:

case1 *From the tester or predefined algorithm*: In this case, the input reliability of the service is accepted 100% reliable (i.e. reliability score = 1.0). Services S_5, S_6 and S_7 in Figure 1 are examples to this case.

case2 *From some arbitrary number of services*: In this case, the input reliability of the service is equal to the lowest of the combined reliability of the services that provide its inputs. For example, service S_3 takes input from one service S_5 while S_4 takes input from two services S_6 and S_7 as depicted in Figure 1.

In the light of these definitions, we formulated our input reliability function to suit these two cases as follows:

$$ir(S_n) = \begin{cases} 1.0 & \text{if } S_n \text{ is case 1} \\ MIN(cr(S_n^1), cr(S_n^2), \dots, cr(S_n^{in(S_n)})) & \text{if } S_n \text{ is case 2} \end{cases}$$

where S_n^i is the service providing ith input for service S_n and $in(S_n)$ is the total number of inputs service S_n has.

The reliability score of a service sequence is equal to the combined reliability of the first service in the sequence (service at the highest level). In the light of the given combined reliability calculation, the objective function for maximising the total reliability is formulated as:

$$Maximise \;\; r_{S_1} \times ir(S_1)$$

3.2 Representation and Genetic Operators

After discovering all possible service sequences ATAM applies the optimisation process to each service sequence separately. For optimisation an initial population is generated using the discovered service groups that form the selected sequence. Each solution in the initial population is generated by randomly selecting a service from each service group. The definitions of the mutation and crossover operators are also needed for optimisation. ATAM was implemented in a way to facilitate these operators with grouping mechanism and sequence building. The mutation operator will replace a service with another service from the same group. The crossover operator will replace services that provide the input of a certain service in the current solution with service(s) from the other solution that provides the same input. Figure 2 illustrates these genetic operators.

4 Conclusion

In this paper, we proposed optimisation in the field of service-oriented test data generation. We focused on the cost of test data generation and the effectiveness of the test data generation approach as our two primary objectives. Other QoS parameters such as service response time can also be introduced as objectives to further improve our test data generation approach.

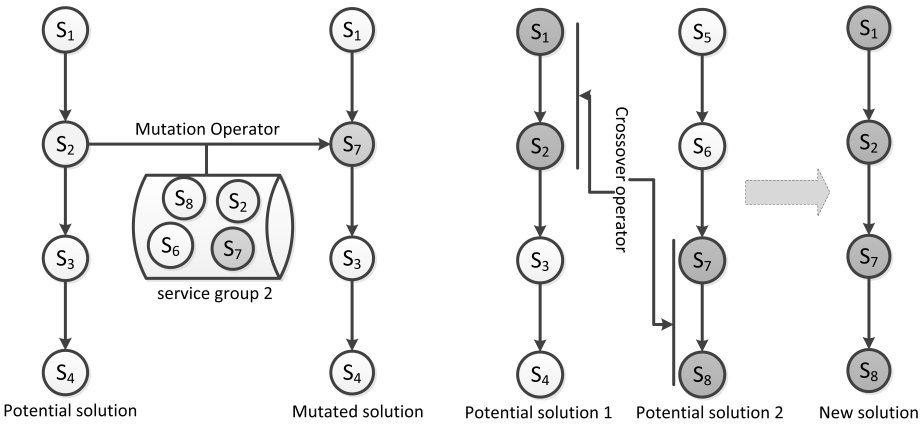

Fig. 2: The mutation and the crossover operators

References

1. Alshahwan, N., Harman, M.: Automated web application testing using search based software engineering. In: 26th IEEE/ACM International Conference on Automated Software Engineering (ASE 2011). Lawrence, Kansas, USA (November 2011), to Appear
2. Bozkurt, M., Harman, M., Hassoun, Y.: Testing web services: A survey. Tech. Rep. TR-10-01, Department of Computer Science, King's College London (January 2010)
3. Bozkurt, M., Harman, M.: Generating realistic test input using semantic web services. Tech. Rep. RN/11/17, University College London (July 2011)
4. Lakhotia, K., Harman, M., McMinn, P.: A multi-objective approach to search-based test data generation. In: GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation. pp. 1098–1105. ACM, London, United Kindom (July 2007)
5. Oster, N., Saglietti, F.: Automatic test data generation by multi-objective optimisation. In: Górski, J. (ed.) Computer Safety, Reliability, and Security, Lecture Notes in Computer Science, vol. 4166, pp. 426–438. Springer Berlin / Heidelberg (2006)
6. Pinto, G.H.L., Vergilio, S.R.: A multi-objective genetic algorithm to test data generation. In: ICTAI '10: Proceedings of the 22nd International Conference on Tools with Artificial Intelligence. vol. 1, pp. 129–134. IEEE Computer Society, Arras, France (October 2010)
7. Sagarna, R., Yao, X.: Handling constraints for search based software test data generation. In: ICST 2008: Proceedings of the 1st International Conference on Software Testing, Verification, and Validation Workshop. pp. 232–240. IEEE Computer Society, Lillehammer, Norway (April 2008)
8. Wan Ab. Rahman, W., Meziane, F.: Challenges to describe QoS requirements for web services quality prediction to support web services interoperability in electronic commerce. In: Proceedings of the 10th IBIMA Conference on Innovation and Knowledge Management in Business. vol. 4, pp. 50–58. International Business Information Management Association (IBIMA), Kuala Lumpur, Malaysia (June 2008)
9. Yoo, S., Harman, M.: Pareto efficient multi-objective test case selection. In: Proceedings of International Symposium on Software Testing and Analysis (ISSTA 2007). pp. 140–150. ACM Press, London, United Kingdom (July 2007)