UTM UNIVERSITI TEKNOLOGI MALAYSIA
INTERNATIONAL JOURNAL OF INNOVATIVE COMPUTING
ISSN 2180-4370
Journal Homepage : https://ijic.utm.my/

Ontology-based Negative Selection Algorithm for Pairwise Testing

Arif Masyhur Mohamed Hatta[1] & Radziah Mohamad[2]
Faculty of Computing
Universiti Teknologi Malaysia, 81310 UTM JohorBahru,
Johor, Malaysia
Email: mmharif2@graduate.utm.my[1]; radziahm@utm.my[2]

Submitted: 17/1/2024. Revised edition: 28/3/2024. Accepted: 12/5/2024. Published online: 31/5/2024
DOI: https://doi.org/10.11113/ijic.v14n1.468

***Abstract*—Pairwise testing is one of the most widely used methods for testing combinations of parameters in software applications. The semantic relationships between parameters may not be taken into consideration by traditional pairwise testing methods, which may result in missing important test cases. It is the objective of this study to evaluate the possibility of improving software testing by combining combinatorial testing algorithms. The proposed technique uses the negative selection algorithm (NSA) approach to change detection and classify the problem domains. To reduce the number of test cases, the researcher combines the model developed using the ontology method for the semantic web with the NSA. In contrast, previous research used a variety of different types of model types when generating test cases. This method proposal proved effective in reducing the testing problem.**

***Keywords*—Pairwise Testing, NSA, Combinatorial Testing**

I. INTRODUCTION

The system testing process is an integral part of the System Development Life Cycle (SDLC), which is essential to the success of a system for the client and customer. Today, technology has rapidly changed and grown in tandem with the demands of society. Life has relied on technology for many years, and the Coronavirus (COVID-19) pandemic has made it even more important. It is often difficult for people to obtain what they require, but the web system, such as online shopping and delivery, has made everything easier and simpler without them having to leave their comfort zone. It is critical for a system that runs errands with important orders and puts people's lives at stake to ensure that they are safe when performing their duties. What can be done to ensure that people who use these technologies trust them daily? To demonstrate their reliability, software testing should be conducted.

There are three types of software testing: black box testing, grey box testing, and white box testing [1]. Unlike black box testing, which does not require any internal components of the system, such as the coding, black box testing emphasizes only the external functionality, as well as model-based testing. A white box test focuses on the internal defects and bugs of the system, such as the coding function, while a grey box test combines both black box and white box tests. The traditional method for generating test cases is exhaustive testing, which gives a large number of test cases, also known as complete testing. The current activity only involves black box testing with model-based testing.

As part of combinatorial testing, pairwise testing is an effective method of detecting actual test cases and helps to save time by testing all the test cases simultaneously. It is an explicit, formal specification of a shared conceptualization. Some classes and relationships represent the relationship between an organization and a person. This allows the model to capture the input-output behavior of the system by using the ontology concept.

II. RELATED WORKS

N Ramli *et al*. [2] proposed the metaheuristics search techniques for the Ant Colony Algorithm, which demonstrates it produces a different number of test cases depending on the test configuration. Using a Gravitational Search Algorithm (GSA), K. Htay *et al*. [3] demonstrated that the test case produced is the best and that all covered tuples are removed from the list to prove that the test case produced is the best.

The Flower Pollination Algorithm is proposed as the Flower Strategy approach, in which it mimics the pollination of the flower. This approach uses the Levy Flight equation to generate population globally. However, based on the final result, it is found that this approach cannot outperform other approaches in some covering arrays with values 313 and 513822 [4].

Ontologies can encode semantic constraints so that negative selection algorithms can filter out combinations that do not conform to the defined rules. In some instances, combinations might violate business rules, constraints, or logical relationships. As a result, logically valid test cases are generated. By utilizing ontology, a negative selection algorithm can identify and eliminate redundant or irrelevant combinations of parameters. The reduction in test cases improves testing. Several works have utilized ontology in software testing life cycle [5, 6]. However, the utilization of the ontology in pairwise testing is still scarce.

III. Experimental Data Sets

Two data sets are used to show the implementation of the ontology model applied with the NSA: the Pizza Option Menu System and the E-Travel System.

A. Pizza Option Menu

This data set is known to be used in creating the ontology model for the past researcher, the same with another approach to choosing the Pizza to purchase [7]. These options for the Pizza are considered as the parameters, including Pizza type, Pizza Crust, Toppings, Size, and Buy Option. Convert it to the Cumulative Array (CA), which will be counted as $2^3 3^2$ to show the difference in number value and parameters. This pizza option is part of the test considered a small version of test cases, but exhaustive testing can produce several test cases. The parameters for this case study are as follows:

1. Pizza type: Veggie lover, Hawaiian Chicken
2. Crust: Thin N Crispy, Classic, Pan
3. Toppings: Mushrooms, Spinach, chicken ham
4. Size: Small, Medium, Large
5. Buy option: Dine in, Take away

From the parameters and values, we can create some false conditions and maybe contain redundancy:

a) Veggie pizza type should not have chicken ham as the topping.
b) Hawaiian Pizza should not have mushrooms as its toppings.
c) Hawaiian Pizza should not have spinach cheese as the toppings.

B. E-Travel System

This case study has been used in past research and can be an enhanced version of the extension [8, 9]. This system is used to get the booking for traveling and planning vacations according to their needs, including flight booking, rental house or hotel, and transportation. The parameters for this system are quite significant, consisting of the Client, Agency Travel, Airline, Hotel, Rental car, Bank type, Travel Period, Travel type, Month of Travel, Payment type, Nationality, Destination, and Season. The large size of test cases creates the CA 3^{13} with the same value but different parameters. The case study parameters are stated as below:

1. Client: Ahmad, Sajad, Vahid
2. Agency Travel: Sfiran, Mayflower, Rakhsh
3. Airline: AirAsia, Mas, Malindo
4. Hotel: Golestan, Persian, Swiss-garden
5. Car rental:4-wheel car, Motorcycle, 2-wheel car
6. Bank: Mellat, Sepah, Pasargad
7. Travel Period: Days, Weeks, Months
8. Travel type: Individual, Couple, Family
9. Month of Travel: January, June, December
10. Payment Type: Cash, Online Banking, Credit card
11. Nationality: Malaysia, Indonesia, Thailand
12. Destination: Malaysia, India, Pakistan
13. Seasons: Spring, Winter, Summer

IV. Methodology

A. Ontology Construction

The first step is important for ontology creation [10] to determine the domain and scope of the research following the chosen data set. In this research, two different data sets are used, which are the Pizza option and the E-travel System. This first step can be created as a question about what ontology will be used and what kind of answer will be given in creating that ontology. For example, the domain of the first dataset is Pizza, and the concept and properties need to be considered in creating the ontology modeling. In the context of the Pizza option, it has five main concepts in the proposed ontology, which are Pizza Type, Pizza Crust, Pizza Toppings, Pizza Size, and Buy Option.

After completing the concept detail and the description, the next step is for the formation of the taxonomy arranged to create the class later. From this taxonomy, it is easier for others to understand the flow of the model and can be a reference for future works.

Axiom is a logical statement considered to be true for the relationship in ontology modeling. Every ontology has its line of axioms used to identify the specification as partial or complete about the class and properties.

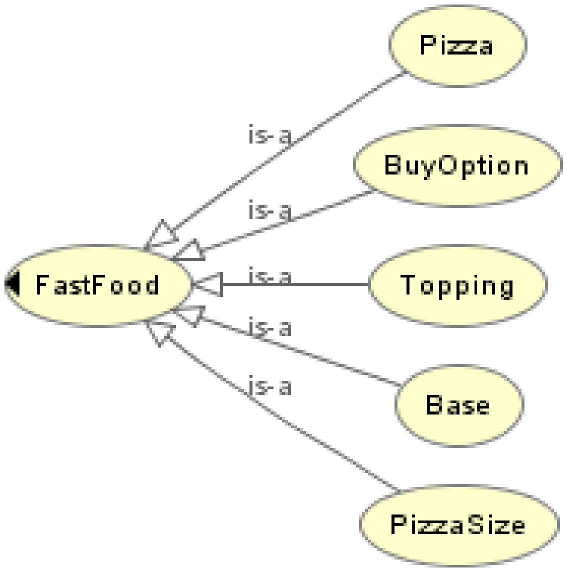

Fig. 1. Pizza ontology class and sub-class

B. *Integration of Ontology and NSA*

As part of this research, an application ontology model will be developed, and a new domain ontology model will be developed to integrate ontology with NSA in pairwise testing. Input parameters from the testing are regarded as SUT information, which will then be processed by a SUT modeling method using an ontology concept and creative output as the complete information of the SUT with parameter values and constraints. The identified parameter value and constraint completely convert into the ontology model and convert into the standard format of pairwise testing, with each parameter having a parameter value and a constraint for the SUT. This research will be evaluated by comparing it with previous work, either ontology or NSA since pairwise testing reduces the likelihood of redundancy in the test cases generated.

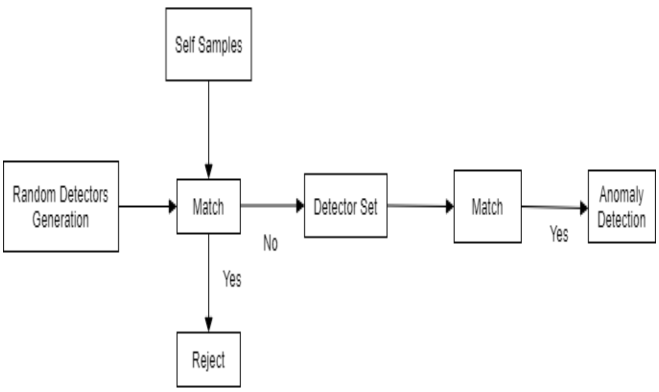

Fig. 2. NSA general flow

NSA-generated data will be matched against the self-samples and calculated. The detector will be rejected if it returns the same results as the self-sample since it already has the data stored. Data that does not match the samples will then be inserted into the detector, and any anomalies will be detected. For example, let the data $[a_1,a_2,\ldots,a_n]$ and $[b_1,b_2,\ldots,b_n]$ as the self-sample of the candidate detector and n is the data size. For this matching degree d, Euclidean distance is the best to use to calculate, as shown in equation 3.1. To get the comparison between this distance, using the detecting matching error, E, equation 3.2 with comparing the distance and present threshold, λ. The detector $[b_1,b_2,\ldots,b_n]$ does not match the self-sample $[a_1,a_2,\ldots,a_n]$ and will be included in the detector set if the error is more than 0. If the error is less than 0, it will be considered a match with the detector and rejected, considered redundancy. From this comparison, the existing anomaly can be detected if the $[a_1,a_2,\ldots,a_n]$ matches the new sample $[b_1,b_2,\ldots,b_n]$ [11].

V. RESULTS

The overall proposed approach by this research is illustrated in Fig. 3. Ontology model construction is the model used integrated for this proposed approach. The input parameter of the testing is considered the SUT information, then will be processed by the SUT modeling method chosen using the ontology concept and create output as the complete information of SUT with parameter-value and constraint. The identified parameter-value and constraint completely convert into the ontology model and the standard format of pairwise testing, with each parameter having a parameter with value for each of it and the constraint of the SUT. The modeling output is extracted into a dataset to be run with the testing part as a CSV file. The CSV file used in the next part, test case generation, will use the NSA to detect the similarity of the test cases. By applying the NSA [12], the K value determined to be used is five, which is the most suitable to avoid too much noise and loose coupling. The test is run at least ten times for each algorithm and test case to get the best overall performance.

After the SUT information in the model is extracted into the dataset, NSA is applied to measure new test cases generated using the two different algorithms to find the best outcome. Hamming distance is an algorithm that can calculate the similarity of the number of different bits in two strings that cover the pairwise testing coverage, as stated in Fig. 4. Euclidean distance is another algorithm used to calculate the similarity that calculates the square root of the sum squared differences, as indicated in Fig. 5.

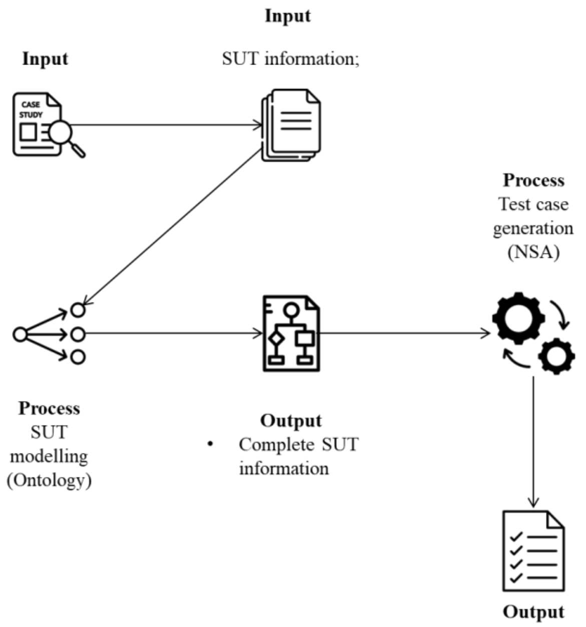

Fig 3. The proposed approach

Input:
1. System under test, S from modelling method algorithm;
   a. Parameter list, $P = (p_1, p_2, \ldots, p_n)$
   b. Instance, $I = (i_1, i_2, \ldots, i_n)$
   c. Constraint, $C = (c_1, c_2, \ldots, c_n)$ where C is the invalid combination of the I of different P

Output:
1. Final test cases $D = (d_1, d_2, \ldots, d_n)$

Begin:

Step 1: Let d be a set of empty test cases.
Step 2: Generate initial test case d.
Step 3: Check the constraint of instance and eliminate the invalid combination from the dataset.
Step 4: Calculate the test data invalid combination of I which is represented by Euclidean distance and can be calculated from Equation:

$$f_{aff}\,(d_i, i_{instance}) = \sqrt{\sum_{i=1}^{N}(d_i - i_{instance})^2}$$

Step 6: Remove the redundant test data. Check the distance $f_{aff}\,(d_i, x_{instance})$
If
$f_{aff}\,(d_i, i_{instance}) < \tau$ then swap the invalid I in $x_{instance}$
Else
output $x_{instance}$ to D
Step 7: Calculate the test data similarity with every test data d which is represented by Euclidean distance and can be calculated from Equation:

$$f_{aff}\,(d_i, x_{instance}) = \sqrt{\sum_{i=1}^{N}(d_i - x_{instance})^2}$$

Step 8: Repeat step 2 to 4 until d of 2 P reach the standard of pairwise testing
Step 9: Remove the redundant test data. Check the distance $f_{aff}\,(d_i, x_{instance})$
If
$f_{aff}\,(d_i, x_{instance}) < \tau$ then remove the new test data $x_{instance}$
Else
Add $x_{instance}$ to D

End

Repeat Step 5 to 6 until D reach the generate number of test set.

Fig. 4. Test Case Generation Algorithm using Euclidean Distance

Input:
1. System under test, S from modelling method algorithm;
   a. Parameter list, $P = (p_1, p_2, \ldots, p_n)$
   b. Instance, $I = (i_1, i_2, \ldots, i_n)$
   c. Constraint, $C = (c_1, c_2, \ldots, c_n)$ where C is the invalid combination of the I of different P

Output:
1. Final test cases $D = (d_1, d_2, \ldots, d_n)$

Begin:

Step 1: Let d be a set of empty test cases.
Step 2: Generate references R show the combination of I for 2 Parameter.
Step 3: Generate initial test case d randomly.
Step 4: Check the constraint of instance and eliminate the invalid combination from the dataset.
Step 5: **For** each $d \in D$ **do**
Step 6: Calculate the similarity of d_i in D, $\forall d_i \in D$ with c_i in C, $\forall c_i \in C$ using the hamming distance and can be calculated from equation:

$$HD(c_i, d_i) = \sum_{i=1}^{n}\overline{(c_i \oplus d_i)}$$

Step 7: Check the distance $HD(c_i, d_i)$;
If
$HD(c_i, d_i) < \tau$ then exchange i_i with other i_{i+1} swap with the valid I
Else
add d in D
Step 8: **End for**
Step 9: Calculate the test data similarity with every test data d which is represented by hamming distance and can be calculated from Equation:

$$HD(r_i, d_i) = \sum_{i=1}^{n}\overline{(r_i \oplus d_i)}$$

Step 10: Check the distance $HD(r_i, d_i)$;
If
$HD(r_i, d_i) < \tau$ then add d in D replace as a new D
Else
add d as d_{i+1} in D
Step 7: Repeat step 3 to 6 until D reach the coverage of full pairwise testing.

End

Fig. 5. Test Case Generation using Hamming Distance

VI. Discussion

Test case generation for this proposed approach showed that it could reduce the number of redundancies and eliminate unnecessary test cases and invalid combinations. The test case generation for exhaustive testing is no different from the previous research with a default value for case study one, which generated as many as 72 test cases. In comparison, the ontology and NSA produce 25 test cases. For case study two, the exhaustive test case number is 1594323 and is too large for the experiment to generate, which is only 28, representing the value. Using the NSA for the testing reduces the number of cases generated and the time taken for the testing phase. The number produced by the ontology modeling and NSA is reduced a bit, even though the performance is far from better when using the enhanced NSA.

Throughout the result from the previous section, the result displayed is different using different approaches and algorithms. The similarity can be reduced but does not perform much better than the past research, and the more significant test case might also reduce only a few amounts of redundancy. Test cases with a small number of instances and parameters can use the hamming distance approach, which gives better accuracy and is much more suitable for small test cases. In comparison, test cases with more parameters and instances can be tested to reduce the similarity using the Euclidean approach, which produces a much bigger accuracy than the hamming distance approach. Table 1 depicts the comparison.

TABLE 1. COMPARISON OF THE RELATED APPROACH

	Random/ Exhaustive	NSA	
		Without Ontology	With Ontology
Pizza Option	72	36	25
E-Travel System	100	28	19

VII. CONCLUSION AND FUTURE WORK

In conclusion, ontologies are essential in supporting negative selection algorithms in pairwise testing because they provide a structured, context-aware, and semantically rich foundation for the algorithm to operate effectively. By leveraging ontology, the negative selection algorithm can filter out irrelevant test cases, improve fault detection, ensure logical validity, and adapt to the unique semantics and constraints of the testing domain, ultimately enhancing the efficiency and effectiveness of pairwise testing.

ACKNOWLEDGMENT

We would like to thank Universiti Teknologi Malaysia for their full support for this research.

REFERENCES

[1] Mohd Ehmer Khan. (2010). Different forms of software testing techniques for finding errors. *IJCSI International Journal of Computer Science Issues*, *7*(3).

[2] Ramli, N., Othman, R. R., Hendradi, R., & Iszaidy, I. (2021). T-way test suite generation strategy based on ant colony algorithm to support T-way variable strength. *Journal of Physics: Conference Series, 1755*(1), 012034. Https://Doi.Org/10.1088/1742-6596/1755/1/012034

[3] Maung Htay, K., Razif Othman, R., & Amir, A. (2021). Utilization of gravitational search algorithm for combinatorial T-way testing. *Journal of Physics: Conference Series*, *1755*(1), 012007. Https://Doi.Org/10.1088/1742-6596/1755/1/012007.

[4] Nasser, A. B., Alsewari, A. A., Tairan, N. M. and Zamli, K. Z. (2017). Pairwise test data generation based on flower pollination algorithm. *Malaysia Journal of Computer Science*, *30*(3).

[5] Abdulhak, M. A. (2013). an ontology-based approach for test case management system using semantic technology. Doctoral Thesis. University of Malaya, Kuala Lumpur, Malaysia. Http://Studentsrepo.Um.Edu.My/5624/1/Mansoor-WHA060019_Phd_Thesis.Pdf.

[6] Franz, O., Bozic, J. and Li, Y. (2020). Ontology-based testing: an emerging paradigm for modeling and testing systems and software. *2020 IEEE International Conference On Software Testing, Verification and Validation Workshops (ICSTW),* 1-4. Https://Doi.Org/10.1109/ICSTW50294.2020.

[7] Alsewari, A. R., Zamli, K. Z., and Kazemi, B. A. (2015). Generating T-way test suite in the presence of constraints. *Journal of Engineering and Technology, 6*(2).

[8] Esfandyari, S., and Rafe V. (2018). A tuned version of genetic algorithm for efficient test suite generation in interactive T-way testing strategy. *Information and Software Technology*, *94*, 165-185.

[9] Sandin, E. V., & Mohamad, R. (2017). Enhanced classification tree method for modeling pairwise testing. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC),* *9*(3-4), 87-92. Https://Jtec.Utem.Edu.My/Jtec/Article/View/2923.

[10] Baclawski, K., Chan, E. S., Gawlick, D., Ghoneimy, A., Gross, K., Liu, Z. H., & Zhang, X. (2017). Framework for ontology-driven decision making. *Applied Ontology*, *12*(3-4), 245-273. Doi:10.3233/AO-170189

[11] Radziah Mohamad, Nor Azizah Sa'adon and Sherolwendy Sualim. (2018). Ontology of mutation testing for Java operators. *International Journal of Innovative Computing, 8*(2), 49-60.

[12] Zeng, J., Liu, X., Li, T. *et al*. (2009) A self-adaptive negative selection algorithm used for anomaly detection. *Prog Nat Sci, 19*, 261-266. https://doi.org/10.1016/j.pnsc.2008.06.008.