Inter-Structure and Intra-Structure Similarity of Use Case Diagram using Greedy Graph Edit Distance

By Evi Triandini

WORD COUNT 4573
TIME SUBMITTED 09-FEB-2023 01:38PM
PAPER ID 96426509

Inter-Structure and Intra-Structure Similarity of Use Case Diagram using Greedy Graph Edit Distance

Fatimatus Zulfa
Department of Informatic
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
fatimatus15@mhs.if.its.ac.id

Daniel Oranova Siahaan
Department of Informatic
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
daniel@its.ac.id

Reza Fauzan
Department of Informatic & Department of Electrical Engineering
Institut Teknologi Sepuluh Nopember
Politeknik Negeri Banjarmasin
Surabaya, Banjarmasin, Indonesia
reza.fauzan@poliban.ac.id

Evi Triandini
Department of Information System
Institut Teknologi dan Bisnis STIKOM Bali
Denpasar, Indonesia
evi@stikom-bali.ac.id

2020 2nd International Conference on Cybernetics and Intelligent System (ICORIS) | 978-1-7281-7257-6/20/$31.00 ©2020 IEEE | DOI: 10.1109/ICORIS50180.2020.9320840

***Abstract*—Use case diagram is one of the behaviour diagrams in UML diagrams. It models the basic needs of users and how users and systems interact to achieve them. Each use case diagram forms a structure which abides a set of standard notations and grammar. Therefore, we can measure the structural similarity between the two use case diagrams. Measuring the similarity of two use case diagram structures can help in the process of reusing diagrams, detecting clones, and assessing learning results. This study proposes an advanced method to measure the structural similarity between the two use case diagrams. Each use case diagram is translated into a graph. The graph represents the inter-structure and intra-structure of a diagram. The graph inter-structure models the interaction between the main use cases and their actors. In comparison, the graph intra-structure models the detailed structure descriptions of every main use case. The similarity between the two represented graph calculates using Greedy Graph Edit Distance. Our experimental result shows that the proposed graph can measure the similarity of the use case structural diagram. The Greedy Graph Edit Distance also can calculate the similarity other UML diagrams for the specific purpose.**

Keywords–graph edit distance, graph similarity, greedy algorithm, structural similarity, use case diagram.

I. INTRODUCTION

Use case diagram is a series of actions performed by the system, users represented by actors, or other systems that interact with the system being modelled [1]. The use case diagram is one of the behavioural diagrams in the UML diagram. The use case diagram component consists of actors, use cases, and relations. Actors are the roles played by objects outside the system that interact directly with the system. Use cases are a series of actions that can be taken by the system and can be observed, which usually applies to one or more actors or other stakeholders of the system. Relationships are relations between actors, between use cases, and between actors and use cases.

Each use case diagram has a standard structure. So, we can measure the structural similarity between the two use case diagrams. Measuring the similarity of the use case diagram structure can help in the reuse diagram process [2], [3], clone detection [4], and similarity assessment [5].

Previous research, there have been several methods to calculate the similarity of use case diagrams. However, the main focus of these methods is to be able to reuse an existing system to be applied to a new approach using structural aspects of the diagram [2], [6]. Other studies have proposed an algorithm (SAFS3) to classify use case analyses to obtain semantic similarity. The research aims to measure the acceptance of artwork or trends in the market for future needs [7]. To efficiently meet user requirements for software, developers can reuse the use cases contained in collections. That means it can save time and money for further development. In that study, a structural equation was made between the requirements of the use case request and the use case contained in the collection [3].

In the study of Al-Khiaty [5] proposed Simulated Annealing to match UML class diagrams based on lexical, internal, environmental similarity, and combinations of the whole. It is done to reuse UML class diagram. The researcher also conducted a UML diagram class matching study to reuse the diagram using the Greedy Algorithm. It is based on lexical, internal, environmental similarity, and a combination of all [8]. Al-Khiaty [9] also proposed a Hybridized Greedy-Genetic algorithm to match UML class diagrams. The method is applied using a case study of fifty UML class diagrams and its performance with the previous algorithm in terms of matching accuracy and convergence time.

Other studies measure the similarity of use cases, also carried out similarity measurements of class diagrams [10] and similarity of sequence diagram [11] for reusing software design diagrams. The use case diagram is one of the UML diagrams needed in software design because it illustrates the functional requirements of a system [12]. Therefore, the use case diagram becomes a critical learning topic in education, especially design subjects, such as Software Engineering. At present, many learning systems are carried out with digital methods such as e-learning. E-learning provides many learning features, an automatic assessment feature in the form of questions with multiple choice answers, a list of choices, fields, and descriptions. The method used as an automated assessment of essay answers applied to e-learning uses the Latent Semantic Analysis or LSA method [13]. In other studies using the N-gram method as feature extraction and cosine similarity to make automatic corrections to essay answers on e-learning [14]. In the subject of Software Engineering, there is the other form of answers, and it is use

case diagram format. Therefore, we need a method that can calculate the similarity between the two use case diagrams.

In previous studies, the measurement of similarity of use case diagrams only paid attention to the lexical information and ignored the use case diagram [15]. The lexical information used for the similarity calculation includes actor name, use case name, relationship name, and relationship type. The research was conducted to be able to reuse software design. If there are examples of input use case diagrams and two use case diagrams in the repository with the same lexical information and one of the two structural diagrams is the same as the input diagram. By calculating semantic similarity, both diagrams from the repository will appear. However, the use case diagram may be similar to the input use case diagram required for project development, including the structural use case diagram. At this point, if the similarity calculation includes a structural similarity to the use case diagram, the use case diagram that does not have structural similarity will not appear from the repository. There may be other cases where two diagrams do not have the same lexical information, even though the structure of the two diagrams is similar. If you only use semantic similarity calculations, the two diagrams will be considered different. Though it could be that the structure of the diagram can be used.

TABLE I. GRAPH ELEMENT OF USE CASE DIAGRAM

No.	Element	Name	Tag
1	Vertex	Actor vertex	*va*
2	Vertex	Use case vertex	*vu*
3	Edge	Association edge	e_1
4	Edge	Generalization edge	e_2
5	Edge	Include edge	e_3
6	Edge	Extend edge	e_4

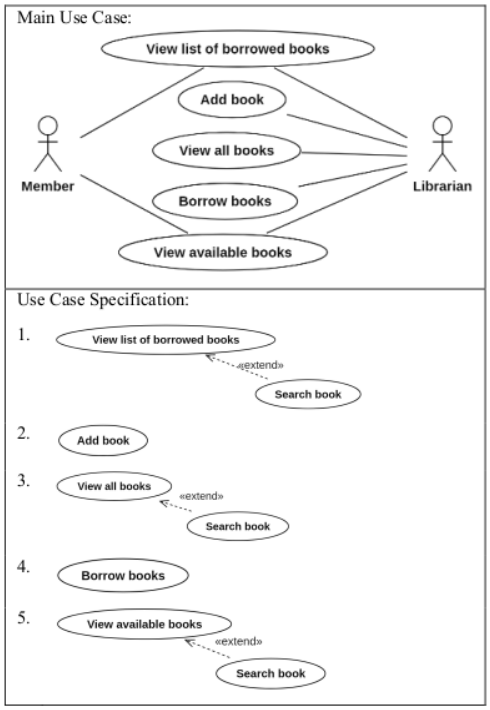


Fig. 1. Use case diagram (UCD) 1 of the Library.

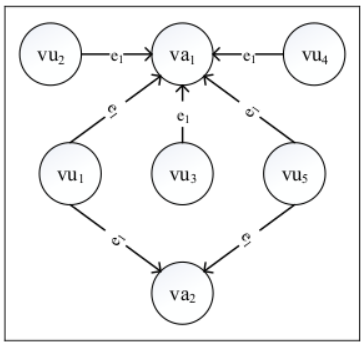


Fig. 2. Inter-structure of use case diagram 1.

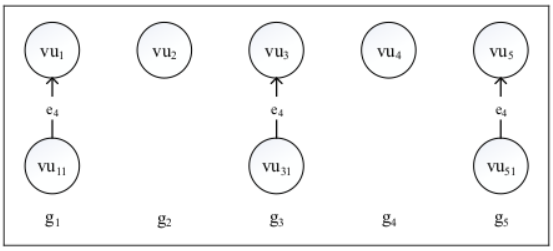


Fig. 3. Intra-structure of use case diagram 1.

This study proposes a method for calculating the similarity between the two use case diagrams that model the same system. It can facilitate the party in giving subjects based on the level of similarity of use case diagrams by students with the answer key. In this research, the measurement of the similarity of use case diagrams is carried out focusing on similarity-based on structural diagrams. And in the future, for maximum results, measurements can be made by combining two aspects of the diagram, namely semantic and structural. Structural measurements are carried out by modelling the use case diagram into a graph and calculating it by combining the Graph Edit Distance (GED) [16], [17] method, and the Greedy algorithm [18], [19].

II. RESEARCH METHOD

A. *Graph Translation of Use Case Diagram*

Before the similarity is calculated, the use case diagram must be translated into graphs. The graph used is a directed graph. Each graph has vertices and edges. The components contained in a graph can be seen in Table 1. The graph has two types of vertices, namely the actor vertex (va) and the use case vertex (vu) node. Edge is useful for connecting between vertices. The proposed edges on the graph are association edge (e1), generalization edge (e2), include edge (e3), and extend edge (e4).

The graph structure is divided into two structures, namely inter-structure and intra-structure. Inter-structure is the main use case structure of a use case diagram. Intra-structure is the detailed structure of the main use case based on the use case specification.

Inter-structure is the main use case of a project. Based on Fig. 1. the resulting intra-structure is Fig. 2. The inter-structure graph has two actor vertices and five use cases vertices. Actor vertex consists of "Librarian" as va1 and "Member" as va2.

The use case vertex consists of "view list of borrowed books" as vu1, "add book" as vu2, "view all books" as vu3, "borrow books" as vu4, and "view available books" as vu5. Each vertex actor is connected to the use case vertex using the association relation as e1.

As we know, every main use case has some use case specification. Therefore, every main use case might have a more detailed structure. The more detailed structure is intra-structure. The number of intra-structures is the same as the number of use cases in the main use case. Therefore, intra-structure can have more than one graph. Based on Fig. 1, the result of translation in intra-structure is Fig. 3. Vertex uc1 has one use case vertex with extended relations. Vertex uc2 and uc4 have no additional structure. Vertex uc3 has one use case vertex with extended relations. Vertex uc5 has one use case vertex with extended relations.

B. Greedy Graph Edit Distance

Greedy Graph Edit Distance (Greedy GED) is used to calculate the similarity between two graphs. The calculation process begins with building a cost matrix for each vertex pa Cost matrix C is a matrix that contains a collection of costs to conve the first graph into the second graph. Following the basic concept of GED is to change the first graph into the second graph with specific treatment. The treatments applied are substitute, add, and delete. Each treatment is worth one cost. Cost (c) is the number of steps needed to change the first node to the second node or to change the first edge to the second edge. The cost matrix has the size m + n, where m is the number of vertices in the first graph, and n is the number of vertices in the second graph. The resulting cost matrix consists of four quadrants. The first quadrant is the costs used to substitute all vertices in the first graph to all vertices in the second graph. The second quadrant is the costs used to delete all vertices from the first graph. The third quadrant is the costs used to add all vertices to the second graph. The fourth quadrant is zero. The cost matrix used as shown in Fig. 4.

$$C = \left[\begin{array}{cccc|cccc} c_{11} & c_{12} & \cdots & c_{1m} & c_{1\varepsilon} & \infty & \cdots & \infty \\ c_{21} & c_{22} & \cdots & c_{2m} & \infty & c_{2\varepsilon} & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \ddots & \infty \\ c_{n1} & c_{n2} & \cdots & c_{nm} & \infty & \cdots & \infty & c_{n\varepsilon} \\ \hline c_{\varepsilon 1} & \infty & \cdots & \infty & 0 & 0 & \cdots & 0 \\ \infty & c_{\varepsilon 2} & \cdots & \vdots & 0 & 0 & \cdots & \vdots \\ \vdots & \ddots & \ddots & \infty & \vdots & \ddots & \ddots & 0 \\ \infty & \cdots & \infty & c_{\varepsilon m} & 0 & \cdots & 0 & 0 \end{array}\right]$$

Fig. 4. The Cost Matrix

From the matrix above, the minimum cost will be sought based on the lowest total value of rows and columns in all combinations. The total combination required is

$$(m + n)! \qquad (1)$$

A standard computer will be difficult to calculate all of these combinations. To speed up the process, we use the greedy algorithm to find the minimum cost combination. The purpose of adding the greedy algorithm to the se ction of permutations is to find the minimum cost needed to convert the first graph into the second graph. The greedy algorithm looks for the first minimum value found to be used as a solution. The number of calculations needed to find the minimum cost in a GED using the Greedy algorithm is m2, where m is more than or equal to n. The greedy algorithm that we apply has several rules. First, if m is more than n, then the search for the minimum cost in the second quadrant cost matrix is done first. Second, if m is less than n, then the search for the minimum cost in the third quadrant cost matrix is done first. Third, if m is equal to n and the number of edges is the same, then the first minimum cost search in the first quadrant cost matrix is done first. Fourth, if m is equal to n and the number of edges is different, then the search for the first minimum cost in the second quadrant cost matrix is done first. The next minimum cost search is carried out in the fourth quadrant. Next, the selection of minimum costs is made by selecting the minimum cost of the remaining costs in the cost matrix.

C. *Structural Similarity Measurement*

Calculation of structural similarity in the use case (*strucUCD*) diagram consists of inter-structure similarity (*interSim*) and intra-structure similarity (*intraSim*). Each component of similarity has its weights, namely $w1$ d w_2. Equation (1) is the *StrucUCD* calculation between the use case diagram d_1 and the use case diagram d_2.

$$strucUCD(d_1, d_2) = w_1 \; x \; interSim(d_1, d_2) + w_2 \; x \; intraSim(d_1, d_2) \qquad (2)$$

InterSim calculations are calculated using Greedy GED by comparing the inter-structure of use case diagram d_1 and the use case diagram d_2. Equation (2) ows how to calculate the similarity of inter-structures from the use case diagram d_1 and the use case diagram d_2.

$$interSim(d_1, d_2) = greedyGED(d_1, d_2) \qquad (3)$$

IntraSim cannot be calculated using only Greedy GED. Intra-structure consists of several graphs in each diagram. Therefore, all the graphs from the use case diagram d_1 will be calculated using all the graphs from the use case diagram d_2. Then, the si larity value that has been collected will be recalculated using the greedy algorithm to find the optimal similarity value between all the graphs from use case diagram d_1 with all the graphs from use case diagram d_2. Equation (3) explains how to calculate *intraSim* between use case diagram d_1 and the use case diagram d_2.

$$intraSim(d_1, d_2) = greedy\left(\sum_{i=1}^{|G_1|} \sum_{j=1}^{|G_2|} greedyGED(g_i, g_j)\right) \qquad (4)$$

where G_1 is a collection of graphs from use case diagram d_1 and G_2 is a collection of graphs from use case diagram d_2.

III. Experimental Result

In this sect n, we use a simple case to show that the representation of a use case diagram into a graph can be calculated for its structural similarity. We calculate the similarity of use case diagrams in Fig. 1 as UCDI and use the case diagram in Fig. 5. as UCD2. Both use case diagrams in library information systems. However, the structure of the diagram is different.

Before we can measure inter-structure and intra-structure similarity, diagrammatic representation must be made into graphs. Graph representation of inter-structural diagram 1 can

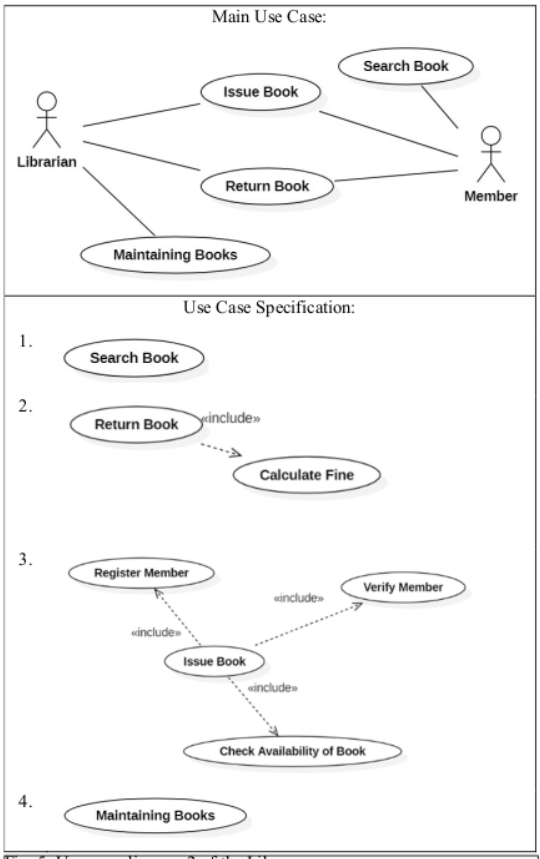
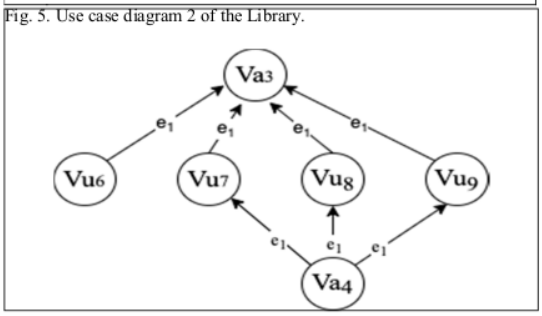


Fig. 5. Use case diagram 2 of the Library.



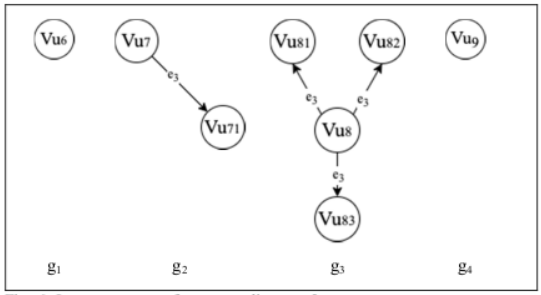


Fig. 6. Intra-structure of use case diagram 2.

be seen in Fig. 5. It can be seen that to translation the main use case into a graph using the rules that have been described.

Graph representation of intra-structural diagram 1 can be seen in Fig. 6. In the example of diagram 2, it has four structural intervals obtained from the use case specification. The use case diagram is changed into four graphs.

After the translation of the diagram to the graph, the cost value will be calculated using a cost matrix. Cost values are obtained using the rules of the Greedy GED method. The first rule of choosing a minimum cost is as follows.

- If m > n, then do the first minimum cost search on the Quadrant II cost matrix.
- If m <n, then search for the first minimum cost on the Quadrant III cost matrix.
- If m = n with the same number of vertices and the edges of the two are different, then do the first minimum cost search on the quadrant I cost matrix.
- If m = n with the same number of vertices and edges, search for the first minimum cost in the Quadrant II cost matrix.

The second rule for selecting minimum cost, the search is carried out in Quadrant 4. It is because, in quadrant four, it consists of cost 0, where cost 0 is the minimum cost. Next, the selection of minimum costs is made by selecting the minimum cost of the remaining costs in the cost matrix.

For the detailed calculation of structural similarity between the two case diagrams, it can be divided into two, inter-structure and intra-structure as follows.

A. Measurement of Inter-structure Similarity

The cost C matrix obtained from graph 1 is the UCD1 representation, and graph 2 is the representation of UCD2 on the inter-structure component as shown in Fig. 7.

$$C = \left[\begin{array}{cccccc|ccccccc} 1 & 6 & 6 & 6 & 6 & 2 & 1 & \infty & \infty & \infty & \infty & \infty & \infty \\ 5 & 0 & 1 & 1 & 1 & 4 & \infty & 2 & \infty & \infty & \infty & \infty & \infty \\ 5 & 1 & 0 & 0 & 0 & 4 & \infty & \infty & 2 & \infty & \infty & \infty & \infty \\ 5 & 1 & 0 & 0 & 0 & 4 & \infty & \infty & \infty & 3 & \infty & \infty & \infty \\ 5 & 0 & 1 & 1 & 1 & 4 & \infty & \infty & \infty & \infty & 2 & \infty & \infty \\ 5 & 1 & 0 & 0 & 0 & 4 & \infty & \infty & \infty & \infty & \infty & 3 & \infty \\ 2 & 3 & 3 & 3 & 3 & 1 & \infty & \infty & \infty & \infty & \infty & \infty & 3 \\ \hline 5 & \infty & \infty & \infty & \infty & \infty & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \infty & 2 & \infty & \infty & \infty & \infty & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \infty & \infty & 3 & \infty & \infty & \infty & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \infty & \infty & \infty & 3 & \infty & \infty & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \infty & \infty & \infty & \infty & 3 & \infty & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \infty & \infty & \infty & \infty & \infty & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right]$$

Fig. 7. The Cost C Matrix inter-structure component

From the cost matrix above, to make the first cost selection using the Greedy GED method, a minimum cost value in quadrant II is searched because of m> n. So that gets one as the initial cost. The second cost search is performed in Quadrant IV, and a cost value of 0 is obtained. Furthermore, the search is performed until each minimum cost value is obtained from different rows and columns. So that the minimum cost is 2, 0, 0, 0, 0, 4, 1, 0, 0, 0, 0, 0, 0. The total cost in the calculation of inter-structural similarity is obtained by the sum of all the minimum costs. It is equal to 7. It means that the value of inter-structure or intra-sim similarity cost (d1, d2) is 7.

After obtaining the value of inter-structure cost between the two use case diagrams, a calculation is performed to get

the inter-structure similarity between the two use case diagrams. The inter-structure similarity value between the two use case diagrams reduces the cost value obtained by the highest possible cost value from the calculation. Then the results are divided by the highest possible cost value from the calculation. If the cost value of two use case diagrams is 0, the use case diagrams are similar. The higher the cost value in the similarity calculation, the lower the similarity value between the two use case diagrams. From the case calculation, the main use case diagram 1, and the main use case diagram 2 show that the similarity value is 82%. The highest possible cost value is 39.

B. *Measurement of Intra-structure Similarity*

The cost C matrix obtained from graph 1 is a representation of the specifications of the use of "View Available Books" (Vu5) cases. Graph 2 is a representation of the specification of use cases "Return Book" (Vu7) on the intra-structural components as follows.

From the cost matrix above, we use Greedy GED to make the selection of the initial cost. We look for the minimum cost value in quadrant II because m = n, the number of vertices and edges of the two is the same. So that gets one as the initial cost. The second cost search is performed in Quadrant IV, and a cost value of 0 is obtained. Furthermore, the search is performed until each minimum cost value is obtained from different rows and columns. To get a minimum cost of 0, 1, 1, 0. To get the total cost in the calculation of inter-structure similarity, the sum of all the minimum costs obtained is equal to 2.

The above is one of the cost value calculations using the cost matrix in intra-structure. To get the overall intra-structural cost value, calculate the entire pair of graphs from each diagram. Choose the smallest cost of each pair between graphs. The calculation of the value of intra-structure cost of the use case specification diagram 1 and the use case specification diagram 2 are 2. By measuring the use case diagram in terms of the structure of the diagram, the limitations of each actor can be explained. If the measurement only uses lexical information, the similarity measurement results from two examples of use cases of "View Available Books" and "Return Book" will get a similarity result of 20%. But when the measurements pay attention to the structure of the diagram, the similarity value of the two use cases can reach 67%. From these results, we can see that by observing the similarity of structural diagrams, the similarity between the two diagrams can be more accurate.

After obtaining the value of intra-structure cost between the two use case diagrams, a calculation is performed to get the value of intra-structure similarity between the two use case diagrams. The intra-structural similarity value between the two use case diagrams is done by reducing the cost value obtained by the highest possible cost value from the calculation. Then the results are divided by the highest possible cost value from the calculation. If the cost value of two use case diagrams is 0, the use case diagrams are similar. The higher the cost value in the similarity calculation, the lower the similarity value between the two use case diagrams. From the case of calculation of the similarity of use case specification diagram 1 and use case specification diagram 2, the result of similarity value is 67%, from the highest possible cost value that is 6.

C. *Calculation of Similarity*

A calculation is made to find the similarity between the inter-structure and intra-structure with the specified weights. The weights used in the similarity measurement are 0.5 each. From the example, the similarity values are 82% inter-structure and 67% intra-structure, respectively. We get a similarity measurement result of 75%.

IV. Conclusion

This study proposes a structural translation of the use case diagram into a graph. The graph structure is divided into two types, namely inter-structure and intra-structure. The proposed graph structure can represent a use case diagram. Inter-structure is the main use case structure of a use case diagram. Intra-structure is the detailed structure of the main use case based on the use case specification. By dividing the graph structure into inter-structural and intra-structural, each actor and use case between diagrams can be measured in detail the cost value.

Their interests are distinguished based on the weight they have. Weight determination can be determined based on the measurement requirements. Measurement needs can be used to reuse diagrams, detect clones, or conduct similarity assessments. Greedy GED is used as a tool to calculate between two graphs that have been built. A greedy GED is expected to accelerate the calculation process than a conventional GED.

For future study, we would like to experiment with a bigger dataset to identify the most suitable weights. The weights would improve the performance of the proposed method in terms of reliability. Thus, Greedy GED can accelerate conventional GED processes. However, it needs further testing related to the accuracy and reliability of Greedy GED on every variable.

Acknowledgement

This research was funded by the Ministry of Research and Technology/National Research and Innovation Agency of the Republic of Indonesia. This research is a collaboration amongst Institut Teknologi Sepuluh Nopember, Politeknik Negeri Banjarmasin, and Institut Teknologi dan Bisnis STIKOM Bali.

References

[1] G. Spurrier and H. Topi, "Resolving the pedagogical disconnect between user stories and use cases in systems analysis and design textbooks," AMCIS Proceeding, 2018.

[2] Z. Yuan, L. Yan, and Z. Ma, "Structural similarity measure between UML class diagrams based on UCG," Requir. Eng., pp. 1–17, 2019.

[3] A. Udomchaiporn, N. Prompoon, and P. Kanongchaiyos, "Software requirements retrieval using use case terms and structure similarity computation," in 2006 13th Asia Pacific Software Engineering Conference (APSEC'06), 2006, pp. 113–120.

[4] H. Störrle, "Towards clone detection in UML domain models," Softw. Syst. Model., vol. 12, no. 2, pp. 307–329, 2013.

[5] M. A. Al-Khiaty and M. Ahmed, "Similarity assessment of UML class diagrams using simulated annealing," in 2014 IEEE 5th International Conference on Software Engineering and Service Science, 2014, pp. 19–23.

[6] A. Adamu and W. M. N. W. Zainon, "Multiview similarity assessment technique of UML diagrams," Procedia Comput. Sci., vol. 124, pp. 311–318, 2017.

[7] N. H. N. D. de Silva, "SAFS3 algorithm: Frequency statistic and semantic similarity based semantic classification use case," in 2015 Fifteenth International Conference on Advances in ICT for Emerging

Regions (ICTer), 2015, pp. 77–83.
[8] M. A. Al-Khiaty and M. Ahmed, "Similarity assessment of UML class diagrams using a greedy algorithm," in 2014 International Computer Science and Engineering Conference (ICSEC), 2014, pp. 228–233.
[9] M. A. R. Al-Khiaty and M. Ahmed, "Matching UML class diagrams using a Hybridized Greedy-Genetic algorithm," Int. Sci. Tech. Conf. Comput. Sci. Inf. Technol., vol. 1, pp. 161–166, 2017, doi: 10.1109/STC-CSIT.2017.8098759.
[10] R. Fauzan, D. Siahaan, S. Rochimah, and E. Triandini, "Class diagram similarity measurement: a different approach," in 2018 3rd International Conference on Information Technology, Information System and Electrical Engineering (ICITISEE), 2018, pp. 215–219.
[11] E. Triandini, R. Fauzan, D. O. Siahaan, and S. Rochimah, "sequence diagram similarity measurement: a different approach," in 2019 16th International Joint Conference on Computer Science and Software Engineering (JCSSE), 2019, pp. 348–351.
[12] D. Kulak and E. Guiney, "Use cases: requirements in context," Addison-Wesley, 2012.
[13] E. S. Pramukantoro and M. A. Fauzi, "Comparative analysis of string similarity and corpus-based similarity for automatic essay scoring system on e-learning gamification," in 2016 International Conference on Advanced Computer Science and Information Systems (ICACSIS), 2016, pp. 149–155.
[14] M. A. Fauzi, D. C. Utomo, B. D. Setiawan, and E. S. Pramukantoro, "Automatic essay scoring system using N-gram and cosine similarity for gamification based E-learning," in Proceedings of the International Conference on Advances in Image Processing, 2017, pp. 151–155.
[15] R. Fauzan, D. Siahaan, S. Rochimah, and E. Triandini, "Use case diagram similarity measurement: a new approach," in 2019 12th International Conference on Information & Communication Technology and System (ICTS), 2019, pp. 3–7.
[16] P. Čech, "Matching UML class models using graph edit distance, "Expert Syst. Appl.," vol. 130, pp. 206–224, 2019, doi: 10.1016/j.eswa.2019.04.008.
[17] K. Riesen, M. Ferrer, and H. Bunke, "Approximate graph edit distance in quadratic time," IEEE/ACM Trans. Comput. Biol. Bioinforma., vol. 13, no. 9, 2015, doi: 10.1109/TCBB.2015.2478463.
[18] A. Fischer, K. Riesen, and H. Bunke, "Improved quadratic time approximation of graph edit distance by combining Hausdorff matching and greedy assignment", Pattern Recognit. Lett., vol. 87, pp. 55–62, 2017, doi: 10.1016/j.patrec.2016.06.014.
[19] K. Riesen, M. Ferrer, R. Dornberger, and H. Bunke, "Greedy graph edit distance," in International Workshop on Machine Learning and Data Mining in Pattern Recognition, 2015, pp. 3–16.

Inter-Structure and Intra-Structure Similarity of Use Case Diagram using Greedy Graph Edit Distance

ORIGINALITY REPORT

9%
SIMILARITY INDEX

PRIMARY SOURCES

1	www.insightsociety.org Internet	85 words — 2%
2	repository.its.ac.id Internet	80 words — 2%
3	repository.poliupg.ac.id Internet	80 words — 2%
4	pdfs.semanticscholar.org Internet	63 words — 2%
5	www.semanticscholar.org Internet	61 words — 2%

EXCLUDE QUOTES	ON	EXCLUDE SOURCES	< 2%
EXCLUDE BIBLIOGRAPHY	ON	EXCLUDE MATCHES	OFF