Response to Reviewers

Reviewer #1: This paper proposes a diversity-driven TCG method DTester, which can maximize behavior exploration and minimize the test suite size while covering more server-side vulnerable paths. Firstly, the authors propose three diversity metrics to measure the underlying relationship between test cases. Secondly, a 3-dimensional weight graph is designed to model association among metrics, which provides fine-grained guidance for genetic algorithm to generate diverse test cases from client-side behavior model. Finally, the empirical evaluation on five web applications shows that DTester can efficiently and robustly generate better test suites than the state-of-the-art TCG method.
In general, this paper is easy to follow. Moreover, the target problem of test case generation for web applications is important and practical. However, this paper suffers from some problems that should be carefully addressed.

(1) In the introduction section, the authors need to reorganize the description logic when summarizing their work. In the text, the authors present three metrics, define the graph, and finally present the DTester.

Response:

Thanks for your comments.

Following your suggestion, we have rewritten the corresponding content about the summary of our work in Introduction of our revised manuscript. Specifically, we fisrtly proposed a novel diversity-driven test case generation method for testing web applications, DTester, which leverages a 3-dimensional weight graph ($\mathcal{G}_{3dw}$) to fully utilize different diversity metrics. We further introduced three diversity metrics to measure the underlying relationship between test cases from the viewpoints of user behavior, code logic, and test execution history.

(2) The viewing order of Fig. 1 is a bit awkward. It would be better for the authors to label the order of DTester execution in Fig. 1 for easy understanding. The focus of the work in this paper is Graph Building, which should be highlighted.

Response:

Thanks for your comments.

In our revised manuscript, we redrew the overview (i.e., Fig. 1) of our proposed approach DTester to improve the clarification. We believe that it will help readers better understand how DTester work and the role of the graph. We have also rewritten the corresponding context in Section "3.1-Overview".

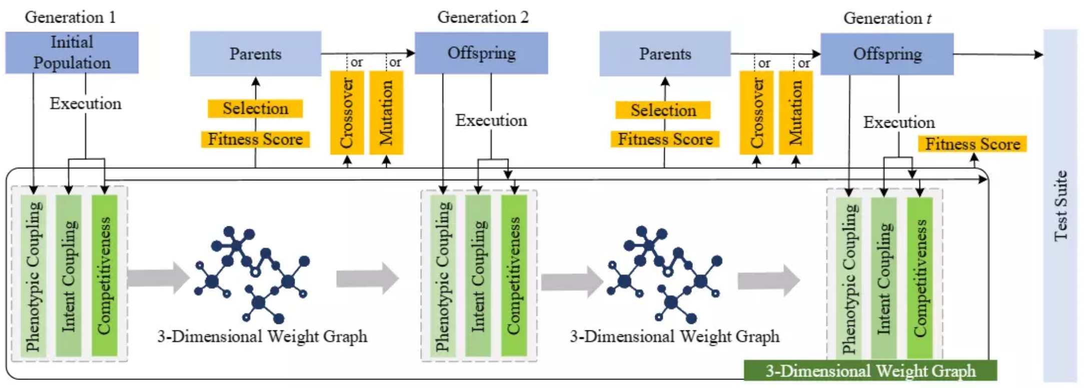


Fig. 1. The overview of DTester

(3) The Seq in Definition 1 has a textual error and there are two „ . Eq. 1 is less well-defined and has not been subsequently reused.

Response:

Thanks for your comments.

Following your comments, we revised the corresponding Definition (i.e., Definition 2 in the revised manuscript) to fix any textual errors and removed Eq. 1 to enhance clarity. Here is the revised content: *" Definition 2. (Test Path) A test path is a sequence of transitions with unspecified input values, denoted by* $S =< r, \ldots, e_i, e_{i+1}, \ldots, e_n >$ *, which satisfies the state reached by the transition* e_i *is the start state of the transition* e_{i+1} *."*

(4) What is the difference between T and t? Why is t used in Eq. 1 and T in the competitiveness description?

Response:

Thanks for your comments.

This is a typo. Both T and t represent the transition. We have updated all of them to "e" in the whole revised manuscript consistently. Moreover, all the authors have tried to review this paper and improved the quality of this paper.

(5) Intent coupling takes into account changes in business logic from a server-side perspective. Phenotypic coupling emphasizes user behavior diversity among test cases from the perspective of the client-side model. Do all test cases w_{ij} contain these four weight values?

Response:

Thanks for your comments.

For an edge between any two test cases I_i, I_j, there are four weight values. Even if the two test cases do not share any similarities in terms of client-side user behavior or server-side code logic, the corresponding weight value will be 1, indicating that there is no phenotypic coupling or intent coupling between both and showcasing a high diversity between them. We have also revised the corresponding context in section 3.2.2 of the revised manuscript to enhance the clarity of our paper.

(6) How to understand the title of section 3.3 " Diversity's Diversity"?

Response:

Thanks for your comments.

Following your comments, we updated the title of previous section 3.3 to "3-dimensional weight graph" (i.e., the section 3.2.2 of out revised manuscript) to more clearly highlight the focus of this section .

(7) How are the default values of parameters such as \alpha and \beta in Eq. 8 and I_{max} in Equation 11 in the configuration obtained? What is the effect of these parameters on the experimental results?

Response:

Thanks for your comments.

\alpha and \beta in our experiments were set as 0.5 respectively, suggesting that the diversity of client-side user behavior and server-side logic code are equally important for caclulating fitness of a test case. Following the previous research [49], the maximum length of a test case was set as 17 in this paper. Therefore, I_{max} were 17.

In the revised manuscript, we provided a detailed description of these parameters. We also provide more discussion of parameter choices in the Threats to Validity section: *"Besides, the choice of parameters poses an internal threat to our study that may impact the empirical results. The performance of our* *method may be affected by the configuration of the coefficient* α *and* β *. To mitigate this particular threat, we experimented with several configurations, ultimately finding that the recommended setting is* $\alpha = 0.5$ *and* $\beta = 0.5$ *."*

---References---

[49]Wang W, Guo X, Li Z, et al. Test case generation based on client-server of Web applications by memetic algorithm[C]//2019 IEEE 30th International Symposium on Software Reliability Engineering (ISSRE). IEEE, 2019: 206-216.

(8) How is dist(I_i,I_j) calculated in the section of metrics?

Response:

Thanks for your comments.

The dist(I_i,I_j) in the section of metrics is calculated using levenshtein distance [29]. We have added its reference in our revised manuscript.

---References---

[29] Levenshtein V I. Binary codes capable of correcting deletions, insertions, and reversals[C]//Soviet physics doklady. 1966, 10(8): 707-710.

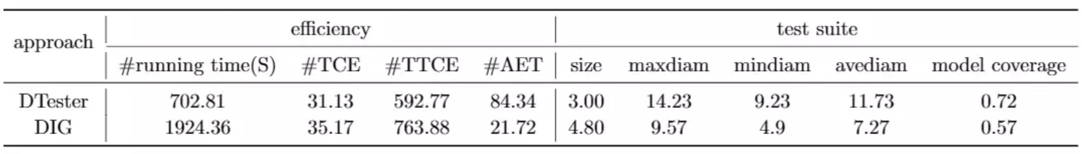
(9) In the related work, the authors mention that Base is the only one that, aside from us, takes into account both the client and server side to guide web test case generation. However, the authors do not introduce some other existing web test case generation methods, and the summary of related work is insufficient. In addition, DTester is compared with only one approach (Base approach) in the experimental section. The authors should compare with other state-of-the-art approaches mentioned in the related work, such as [23] and [30].

Response:

Thanks for your comments.

In our revised manuscript, we have added a new "Existing Work on Web Test Case Generation" subsection (i.e., Section 2.2.2) based on this comment, and highlighted how our work differs from previous work. In addition, we added one baseline method, DIG [12] (one state-of-the-art web test case generation technique utilizing diversity). Due to time constraints, we limited our comparison of DTester and DIG to a web application and discussed the results in the new subsection RQ5. Table 6 lists the performance results of the DTester and DIG. From Table 6, we can see that our proposed DTester outperforms DIG in terms of both test efficiency and test suite quality.

Table 6. Comparison of Efficiency and Test Suite between DTester and DIG on Addressbook



approach	efficiency				test suite				
	#running time(S)	#TCE	#TTCE	#AET	size	maxdiam	mindiam	avediam	model coverage
DTester	702.81	31.13	592.77	84.34	3.00	14.23	9.23	11.73	0.72
DIG	1924.36	35.17	763.88	21.72	4.80	9.57	4.9	7.27	0.57

---References---

[12] Biagiola, M., Stocco, A., Ricca, F., and Tonella, P., Diversity-based web test generation, in Proceedings of the ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/SIGSOFT (ACM, 2019), pp. 142–153.

(10) The authors should improve the quality of the figures.

Response:

Thanks for your comments.

Following your suggestions, we updated all the figures in our revised manuscript to improve the quality and clarity in accordance with the guidelines and standards. Specifically, we have addressed various aspects such as resolution, font, color, legend, and title to ensure that they meet the required standards and enhance the overall presentation of our work.

Reviewer #2: Paper Summary:This paper presents a test generation method based on genetic algorithm to generate test cases of web applications. The technique is designed to consider both the client model and the server-side code to maximize behavior exploration, minimize the size of the test suite, and cover more vulnerable paths on the server side. Three diversity metrics are proposed. Further, a 3-dimensional weight graph is designed to model association among these metrics, which provides fine-grained guidance for genetic algorithm to generate diverse test cases from client-side behavior model. Evaluation is conducted on five open source web applications and target evaluation targets efficiency, test suite size, diversity, and robustness.

Strengths:
- This paper proposes a method that simultaneously considers the client model, server code, and test execution history, and proposes corresponding diversity metrics.
- The diversity metrics are used to guide the exploration process of the genetic algorithm, which improves the effect of the generation algorithm and has some originality.

Weaknesses:
- There is no clear explanation of the relationship between test history and diversity, especially the relationship between the competitiveness metric proposed by test history and diversity, and the calculation of this metric is done by the transitions in the client model. It is unclear why it should be parallel to the Phenotypic Coupling metric proposed by the client model.

Response:

Thanks for your comments.

Intent coupling only focuses on calculating the diversity between two test cases within the current population, instead of evaluating their diversity with respect to previously executed test cases during population evolution. Unlike intent coupling, competitiveness evaluates the diversity of a single test case from the perspective of the testing execution history. Its goal is to find diverse test cases, which are farther from previously executed test cases, by analyzing the historical execution frequency of transitions composing a test case. Such test cases tend to involve less-frequented transitions or harder-to-access page states and are more likely to uncover undiscovered user behaviors or server-side logic code.

In our revised manuscript, we have provided a clear explanation of competitiveness and their relationship with intent coupling and phenotypic coupling to address your concerns in Section "3.2.1-Diversity Metrics". We believe that this additional explanation will help readers better understand these diversity metrics and their relationship with diversity.

- Both Abstract and Introduction mention the diversity metrics proposed by "user behavior, code logic, and test execution history", but in the contribution and method sections, they mention the diversity proposed by "client, server and test history dimensions". Should they be unified?

Response:

Thanks for your comments.

Following your suggestion, we have used "user behavior, code logic, and test execution history" consistently in the revised manuscript to enhance the clarity of the paper. Here is the revised content in the contribution and method sections:

*"**Three novel diversity metrics**, (phenotypic coupling, intent coupling, competitiveness), comprehensively assess the diversity relationship between web test cases from the dimensions of user behavior, code logic, and test execution history."*

"Phenotypic Coupling is focused on identifying the overlap in client-side user behavior simulated by two test cases."

"Intent Coupling is designed to assess the extent of overlap between the server-side logic code covered by two test cases."

- In Section 3.2, the formula formats of Phenotypic Coupling and Intent Coupling are inconsistent. For the definitions involving "||" symbols, no specific explanations are given, especially for |E1| and so on. How is the length of logic code measured by code lines or other methods? No explanation is given for "∩E1,E2" either. The relationship between the values of Definition 5 and Definition 6 and diversity is not given.

Response:

Thanks for your comments.

In the revised manuscript, we standardized the formula formats for phenotypic coupling and intent coupling in Equations 1 and 2, and also provided specific explanations for the symbols involved in Section "3.2.1-Diversity Metrics". Specifically, the symbol "||" represents the length of a set. E_1 refers to a set of vulnerable paths covered by the test case I_1. Because our goal is to cover vulnerable paths on the server side, we represent the testing intent of a test case by it covering vulnerable paths. "∩" is the intersection operation of two sets. "∩E1,E2" is the vulnerable paths shared by test cases I_1 and I_2.

Regarding the relationship between the values of intent coupling (i.e., previous Definition 5) and conditional intent coupling (i.e., previous Definition 6) and diversity, we utilized Definition 6, Equation 2, and the corresponding descriptions to provide a clear explanation in Section "3.2.1-Diversity Metrics". In particular, intent coupling focuses on the similarity of logic code between two test cases, while conditional intent coupling is more concerned with which test case contributes more to enhancing logic code diversity when there is an intent coupling between two test cases. The relationship between the values of both can be seen in Equation 2, whereas their relationship with diversity can be seen in Definition 6 and associated descriptions.

$$\begin{aligned}\widehat{I_1 I_2} &= \frac{1}{2} \times (\widehat{I_1|I_2} + \widehat{I_2|I_1}) \\ &= \frac{1}{2} \times (\frac{|LSC(S_1,S_2)|}{|S_1|} + \frac{|LSC(S_1,S_2)|}{|S_2|})\end{aligned} \tag{1}$$

$$\begin{aligned}\widetilde{I_1 I_2} &= \frac{1}{2} \times (\widetilde{I_1|I_2} + \widetilde{I_2|I_1}) \\ &= \frac{1}{2} \times (\frac{\cap_{E_1,E_2}}{|E_1|} + \frac{\cap_{E_1,E_2}}{|E_2|})\end{aligned} \tag{2}$$

- In Section 3.3, the specific calculation formula for the diversity2 of the subgraph is not given.

Response:

Thanks for your comments.

In section 3.2.2 of our revised manuscript, we have provided a more clear definition and specific calculation formula of diversity^2 to enhance the clarity. Here is the related revised content in section 3.2.2:

*"**Definition 9 (Diversity^2)** Given a group of test cases* $\mathcal{I} = \{I_1, I_2, \ldots, I_n\}$ *and multiple diversity metrics* $\mathcal{D} = \{D_1, D_2, \ldots, D_m\}$ *, diversity^2 accesses their comprehensive diversity under multiple diversity metric, as shown in Equation 6.*

$$D^2(\cup_{I_i \in \mathcal{I}}^{D_j \in \mathcal{D}}) = \sum_{I_i \in \mathcal{I}} \sum_{D_j \in \mathcal{D}} (\omega_j \times D_j(I_i)) \tag{6}$$

where ω_j *is the weight of the diversity metric* D_j *, and* $\sum_{j \in [1,m]} (\omega_j) = 1$ *."*

- Section 3.4 mentions: "G3dw can point out which test cases are overlapping or even redundant, and prioritize test cases according to weights." Does prioritize test cases use the weight of each test case's competitiveness or other metrcis? If it refers to the competitiveness, then it can be sorted by the rank definition given by Definition 10, rather than by the constructed G3dw.

Response:

Thanks for your comments.

In the revised manuscript, we have made necessary corrections to accurately depict our work and avoid misunderstandings caused by improper presentation in the section "3-dimensional weight graph". The G3dw incorporates three diversity metrics (i.e., phenotypic coupling, intent coupling, and competitiveness) to fully utilize diversity information from different dimensions. While conducting web testing, We did not use any of them separately to prioritize test cases but considered different diversity metrics together. We demonstrated how G3dw represents diversity information in different dimensions, using the example shown in Fig. 2. Additionally, we showcased the limitations and potential biases of relying solely on a one-dimensional diversity metric. Specifically, we displayed the rankings of test cases according to phenotypic coupling, intent coupling, and competitiveness respectively, which reveal differences among their rankings. This emphasizes the need for using G3dw to comprehensively evaluate test case diversity.

-There is a symbol error in formula 9 in Section 3.4, and there are typos in definition 9.

Response:

Thanks for your comments.

In our revised manuscript, we have replaced typos with "weight" in Definition 8 (i.e., previous Definition 9) and fixed the symbol error of Formula 7 (i.e., previous Formula 9) as follows :

$$F_{I_i}^{div} = 1 - \frac{\sum_{j \in \nabla} (\alpha \widehat{I_i|I_j} + \beta \widetilde{I_i|I_j})}{|\nabla|}, \quad \alpha + \beta = 1 \tag{7}$$

In addition, we have used Grammarly (www.grammarly.com) to check our manuscript. Moreover, all the authors in this paper have tried to remove typos in this paper as much as possible and improved the English quality of this paper.

-In Section 3.4, based on similarity recombination, the coupling similarity mentioned is not given a corresponding definition, and the relationship with the metrics proposed earlier is unclear.

Response:

Thanks for your comments.

In our revised manuscript, we have provided a clear definition (i.e., Definitions 5 and 6) for coupling similarity (i.e., phenotypic coupling and intent coupling). Furthermore, we have revised the content concerning recombination in the revised manuscript, carefully breaking it down into selection and crossover to provide an accurate depiction of our work. Specifically, phenotypic coupling determines the degree of structural similarity between two test cases by comparing their test paths in the client, while intent coupling determines the degree of functional similarity between two test cases by comparing their execution code on the server. By utilizing these similarity calculations, we can effectively select and cross test cases to generate new diverse test cases.

Here is the related revised content in the section "3.2.2-Genetic Operation":

*"**Selection** follows two principles to ensure the retention of good transitions and test diversity in our work: (1) Test cases with higher fitness are more likely to be selected as parents to preserve good transitions. (2) Two test cases that are further apart are more suitable as parents to generate diverse test cases and promote the exploration of unexplored areas of web applications. According to Equation 7, two test cases with high-fitness do not necessarily mean that they are furthest apart from each other, because fitness is calculated based on population. Therefore, the key insight in selection is crossing dissimilar high fitness test cases to produce new test cases that have larger differences from the parents. Specifically, we start by choosing a high-fitness test case as the paternal parent, and then select its farthest neighbor in the* $\mathcal{G}_{3dw}$ *based on the weight of edges (i.e., phenotypic coupling and intent coupling) to as the maternal parent. This selection process aims to lay the groundwork for generating high-quality offspring.*

***Crossover** is a probabilistic event determined by the crossover probability* P_c *. During the crossover operation, two selected test cases acting as parents will be input for processing. If the crossover occurs, a single-point crossover operation is executed to exchange transition fragments between parents, thereby creating two new test cases. Otherwise, no operation will be performed on the parents. In our work, the crossover also follows two principles to generate diverse test cases, as shown in Fig. 3: (1) crossover tends to exchange transition fragments starting from different transition points, which have the same start state, in two test cases. (2) crossover tends to generate new test cases that differ from their parents."*

-In formula 15 in Section 4.2, the denominator is not explained.

Response:

Thanks for your comments.

The denominator $\mathtt{C}_{|P|}^{2}$ is the number of test case pairs in the popculation *P*. This omission has been supplemented in the revised manuscript. Please refer to the relevant description of Formula 13 (i.e., previous Formula 15) for details.

Reviewer #3: I enjoyed reading the article "DTester: Diversity-driven Test Case Generation for Web Applications."

The authors propose three new diversity metrics (phenotypic coupling, intent coupling, and competitiveness) to gain more insights into a single test case and association among test cases.

They propose a 3-dimensional weight graph realizing the collaboration of several test diversity metrics.

They develop a novel web testing method DTester that covers both the client and server sides. In their testing method, they use the proposed model to dynamically adjust the parameters of the used genetic algorithm according to the test diversity.

They evaluate and compare their method with "Base" **(the most comparable baseline work)** and show considerable improvements in the efficiency, test suit size, diversity, and robustness.

The paper is very well-organized. The contributions are clearly stated. Related work is concisely explained. The figures, tables, equations, formula, and algorithms are appropriately explained.

There is no issue in referencing except a couple ones I have mentioned below.

The comparison, achievement, and conclusion seem valid and reliable.

I strongly recommend the article for publishing in the International Journal of Software Engineering and Knowledge Engineering (IJSEKE)

Please consider the following few suggestions:

-Section 1, the sentence ending with "...., costly and error-prone." may need a reference.
Response:
Thanks for your comments.
Following your suggestion, we have added the relevant reference in the "Introduction" section of our revised manuscript: *"However, such manual strategy is also labor-intensive, costly, and error-prone [57]."*

---References---
[57]Zheng Y, Liu Y, Xie X, et al. Automatic web testing using curiosity-driven reinforcement learning[C]//2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE). IEEE, 2021: 423-435.

-Section 1, the sentence beginning with "Among them, search-based software..." needs a reference right after "software."
Response:
Thanks for your comments.
In our revised manuscript, we have provided appropriate references to support the corresponding context: *"Among them, search-based software testing [27,49] has received significant attention due to its effectiveness for real-world systems."*

---References---
[27] Jan S, Panichella A, Arcuri A, et al. Automatic generation of tests to exploit XML injection vulnerabilities in web applications[J]. IEEE Transactions on Software Engineering, 2017, 45(4): 335-

362.

[49] Wang W, Guo X, Li Z, et al. Test case generation based on client-server of Web applications by memetic algorithm[C]//2019 IEEE 30th International Symposium on Software Reliability Engineering (ISSRE). IEEE, 2019: 206-216.

-Section 1, the sentence ending with "... which can reveal weak robustness." needs a reference.

Response:

Thanks for your comments.

In the revised manuscript, we have included the relevant references within their respective contexts: *"Furthermore, the efficiency of multiple searches for the same vulnerability may vary considerably due to the randomness nature of GA [50], which can reveal weak robustness [41]."*

---References---

[41] Jan S, Panichella A, Arcuri A, et al. Search-based multi-vulnerability testing of XML injections in web applications[J]. Empirical Software Engineering, 2019, 24: 3696-3729.

[50] Wang W, Wu S, Li Z, et al. Parallel evolutionary test case generation for web applications[J]. Information and Software Technology, 2023, 155: 107113.

-Section 2.1, the "Selenium" tool needs a reference or footnote.

Response:

Thanks for your comments.

In the revised manuscript, we have provided the corresponding footnote for Selenium as you commented: *"To automatically execute test cases, they are converted into test scripts using Selenium [a], a tool for web automation testing, and run in the browser.*

---Footnote---

a https://www.selenium.dev/selenium-ide/

-Section 4.1, the application phpaaCMS footnote does not contain the complete URL.

Response:

Thanks for your comments.

We have shared the application phpaaCMS used on our GitHub page. For more detailed information, please visit the following website: *https://github.com/Rose4948/DTester*