УДК 004.052.42 **DOI:** 10.31673/2412-9070.2026.028108

А. П. САМОЙЛЕНКО, аспірант;
ORCID: 0009-0004-6353-1877
І. С. ЩЕРБИНА, канд. техн. наук, доцент,
ORCID: 0009-0004-8373-7522
Державний університет інформаційно-комунікаційних технологій, Київ

НАПРЯМИ ПІДВИЩЕННЯ ЕФЕКТИВНОСТІ МЕТОДУ CBUA В УМОВАХ ДИНАМІЧНОГО НАБОРУ ТЕСТІВ

У контексті зростання складності сучасного програмного забезпечення, забезпечення його надійності та якості залишається одним з ключових завдань інженерії програмного забезпечення. Одним із перспективних напрямів є використання мутаційного тестування, що дає змогу оцінити якість тестових наборів шляхом створення модифікованих версій програми - мутантів. Проте застосування цього підходу в умовах динамічного набору тестів, коли тестові випадки створюються або доповнюються під час виконання, унеможливлює використання низки традиційних оптимізацій.

У зв'язку зі значною обчислювальною вартістю мутаційного тестування було запропоновано широкий спектр методів оптимізації, спрямованих на зменшення витрат при збереженні здатності до виявлення помилок. Окрему групу становлять стратегії оптимізації, пов'язані з генерацією, виконанням та аналізом мутантів, до яких належать підходи, що не належать до інших категорій. Саме в цій групі знаходяться методи, що отримують результати мутаційного тестування без фактичного виконання мутантів. У контексті динамічних тестових наборів особливу перспективність демонструють підходи, здатні зберігати точність оцінювання за умови зміни тестів.

Метою даного дослідження є систематизація та аналіз сучасних підходів до підвищення ефективності методів прогнозування результатів мутаційного тестування. Особливу увагу приділено аналізу стратегій підвищення ефективності таких методів, зокрема вибору релевантних ознак для алгоритмів машинного навчання, побудові адекватних моделей мутаційного тестування та виявленню загроз валідності результатів.

У результаті дослідження виокремлено групу ознак, що є перспективними для вдосконалення стратегій оптимізацій, ідентифіковано основні загрози валідності експериментів, а також визначено напрями подальших досліджень. Зокрема, рекомендовано дослідження історичних в контексті методів прогнозування результатів мутаційного тестування та використання більш ефективних статичних і динамічних ознак, а також побудову більш точних математичних моделей, що враховують часові характеристики для підвищення загальної ефективності даних методів.

Ключові слова: якість програмного забезпечення, тестування програмного забезпечення, мутаційне тестування, прогнозування результатів тестування.

Вступ

У сучасних умовах зростаючої складності програмних систем забезпечення їхньої надійності та якості залишається однією з ключових задач у галузі інженерії програмного забезпечення. Застосування мутаційного тестування в умовах динамічного набору тестів, коли тестові випадки доповнюються під час виконання, зумовлює неможливість використання ряду традиційних оптимізацій [1]. Такий підхід був застосований у роботах [2], [3], де було відмічено, що основними викликами в результаті реалізації є значні обчислювальні витрати.

Тому, для масштабування таких систем, як Mu2 [2], [3], було запропоновано низку підходів до зменшення обчислювальної складності, зокрема використання методу PMT

(Predicted Mutation Testing) [4], що дає змогу прогнозувати результати мутаційного тестування без безпосереднього виконання мутантів. Однак, використання методу CBUA (Coverage-Based Unsupervised Approach) [5] може мати більшу ефективність для вирішення даної задачі, оскільки даний метод не знижує мутаційний рахунок при додаванні нових тестових випадків.

Аналіз останніх досліджень і публікацій

Мутаційне тестування - це техніка оцінювання здатності наборів тестів виявляти помилки у заданій програмі. Мутація - це синтаксична зміна в коді програми, яка може призводити до семантичної зміни. Нова версія програми, що включає мутацію або мутації є мутантом.

Коли тест розрізняє поведінку мутованої версії від поведінки оригінальної програми, то такий мутант є виявлений, в іншому випадку – живий [6]. У такому випадку, прогнозування результатів мутаційного тестування можна звести до задачі бінарної класифікації. Такі методи, як PMT створюють класифікаційну модель, що використовує ознаки, які можна отримати, або статично, або з мінімальними обчислювальним витратами. Важливим етапом у цьому процесі є відбір ознак, під час якого розглядається завдання вибору найбільш інформативної та значущої підмножини змінних із загального набору даних. Метою є покращення якості моделі шляхом усунення зайвої або корельованої інформації, яка може спричинити перенавчання. Якісний відбір ознак є критичним для побудови ефективних і точних моделей прогнозування.

За природою отримання ознак виділяють три основні категорії: динамічні, статичні та історичні:

1. Динамічні ознаки (Dynamic features). Це ті ознаки, які отримуються в процесі виконання коду. Тобто, вони збираються під час виконання програми або тестування. Прикладом таких ознак є - кількість виконань мутованого виразу, кількість тестів, що покривають даний вираз. Такі ознаки мають найбільший вплив на ефективність прогнозування.

У роботі [7] для запропонованого підходу з позиції аналізу важливості ознак, numExecuted (значення того, скільки разів даний мутант був виконаний) має потенціал забезпечити вищу прогностичну здатність моделі. Для прогнозування CBUA використовує тільки одну ознаку, що формується на основі даних з покриття коду - numTestCover - кількість тестових випадків, що покривають мутований вираз. Аналогічно використання numExecuted може бути більш ефективним, ніж використання numTestCover, оскільки така ознака містить більше інформації про мутантів під час тестування. Зокрема, numTestCover лише фіксує факт покриття мутанта певною кількістю тестів, не враховуючи частоту виконань, тоді як numExecuted враховує як кількість тестів, так і загальну кількість виконань мутанта. Це дозволяє отримати більш глибоке уявлення про інтенсивність його використання у тестових сценаріях, що безпосередньо пов’язано з ймовірністю виявлення мутанта та, відповідно, з точністю моделі прогнозування.

2. Статичні ознаки (Static features). Це характеристики, які можна зібрати без виконання програми. Їх можна визначити, аналізуючи вихідний код або взявши інші статичні властивості програми. Такі ознаки можуть включати різноманітні метрики, такі як розмір коду (LOC), складність, зв'язаність та інші. У порівнянні з динамічними ознаками, статичні ознаки можуть бути менш затратними за часом та ресурсами, оскільки вони не вимагають виконання програми. Дослідження моделі прогнозування результатів мутаційного тестування [8] виявило, що при використанні тільки статичних ознак модель досягає такого ж результату з невеликим зниженням точності, у порівнянні з комбінованим підходом - з додаванням також і покриттям виразів. При застосуванні в PMT [4, 9] - найбільш значущими ознаками є ознака стійкості пакету до змін [5], тип мутатора (MutatorClass) [9] та метрика Холстеда на рівні методу [7]. У дослідженнях [4, 9] також відмічено, що більш високорівневі метрики (наприклад, пакетного рівня) є більш важливими, ніж низькорівневі метрики (рівня класу або методу).

3. Історичні ознаки. Вказують на характеристики, які враховують аспекти історії, зазвичай пов'язані з минулим використанням або обслуговуванням системи, або даних. Ці ознаки можуть включати в себе інформацію про попередні стани, зміни, виправлення помилок, оновлення та інші події, які сталися в минулому. Історія може включати в себе дані про зміни в коді,

коміти в системі контролю версій і т.д. Наразі, існують методи та моделі прогнозування результатів мутаційного тестування, що використовують, як динамічні так і статичні ознаки, але відсутні такі, що використовують історичні ознаки.

Метою даного дослідження є систематизація та аналіз існуючих підходів для підвищення ефективності методів прогнозування результатів мутаційного тестування, з використанням їх для методу CBUA з урахуванням динамічного набору тестів.

Результати дослідження

CBUA - це підхід на основі покриття без нагляду, який використовує інформацію про покриття коду для оцінки ймовірності виживання мутантів. Він моделює процес мутаційного тестування за допомогою розподілу Бернуллі, де вірогідність виявлення мутанта залежить від кількості тестів, що покривають дані синтаксичні зміни. Мутант є покритим, коли вираз, на основі якого він був створений, виконується принаймні одним тестом із тестового набору.

Такі ознаки як numTestCover та numTestExec - розглядають тільки кількісні характеристики взаємодії набору тестів з мутантом, однак вони не враховують часові характеристики тестів і мутантів. Використання розподілу Пуассона в CBUA дозволяє побудувати гнучкішу та інформативнішу модель, яка враховує не лише факт покриття, але й частоту та інтенсивність взаємодії тестів з мутантом.

На відміну від підходів, що використовують навчання з учителем, наприклад, метод випадкових лісів, які потребують великої кількості розмічених даних і припущення про однорідність розподілу даних у навчальній і тестовій вибірках, CBUA уникає цих обмежень. Це дозволяє зменшити витрати, пов'язані з процесом навчання, які залишаються високими у традиційних підходах через потребу у проведенні повноцінного мутаційного тестування. Аналогічно до PMT, CBUA дозволяє виявляти потенційно живих мутантів, допомагаючи розробникам підвищувати якість тестових наборів. Зокрема, CBUA демонструє конкурентоспроможність із PMT за наявності великих обсягів навчальних даних і перевершує його в умовах обмеженості ресурсів.

Однак, у роботі [7] розглядається вплив непокритих мутантів, як загроза валідності для методу PMT. Наявність великої кількості непокритих мутантів може спотворювати результати прогнозування, знижуючи загальну точність моделі до ефективності випадкового вгадування, так як такі мутанти завжди є невиявленими. Додатково, даний вплив поширюється і на важливість ознак [7, 10]. Вірогідність виявлення мутанта в методі CBUA вираховується, наступним чином:

$$prob = 1 - p^{(x+1)}, \quad (1)$$

де p - значення початкової ймовірності, що виражається як відношення непокритих мутантів до загальної кількості мутантів:

$$p = \frac{X}{N} \quad (2)$$

Дана загроза валідності також впливає і на CBUA, оскільки кількість непокритих мутантів буде дорівнювати нулю і як результат значення початкової вірогідності також буде дорівнювати нулю. Тому, при використані даної моделі на наборі мутантів, при умові, що кожен з них є покритим, необхідно використовувати іншу методику розрахунку значення p. Для вирішення цієї проблеми можна замінити співвідношення мутантів метриками складності програмного забезпечення, зокрема метрикою Холстеда або метрикою Маккейба.

У співвідношенні (1), вираз $\log_2(x + 1)$ є функцією, що описує явище зниження ефективності тестів у процесі виявлення дефектів. У роботі [5] дане співвідношення (1), було сформоване на результатах дослідження [11]. Однак, за методологією цієї роботи [11], формування дослідних зразків здійснювалося шляхом випадкового вибору підмножини методів, які було відібрано з тестових наборів досліджуваних проєктів. Таким чином, вплив даних, що створені за допомогою методів автоматичної генерації тестів може бути іншим і відповідно закономірність між ефективністю тестів до їх числа також може бути іншою. Також недоліком даної

логарифмічної залежності є відсутність індивідуальних параметрів, що дозволяють апроксимувати криву залежності відповідно до кожної програми, що тестується індивідуально.

Ще однією загальною загрозою валідності для методів прогнозування результатів мутаційного тестування є застосування різних інструментів для проведення тестування, а також використання даних методів за допомогою імплементації на інших мовах програмування. Зміна мови програмування часто супроводжується використанням альтернативних наборів мутаційних операторів, що безпосередньо впливає на природу створюваних мутантів, результати мутаційного тестування та, як наслідок, на ефективність побудованих моделей прогнозування. Ця проблема була детально проаналізована у дослідженні [10], де автори здійснили відтворення оригінального експерименту [4] та додатково реалізували метод PMT для мови програмування C. Такий підхід дозволив оцінити вплив мовних особливостей на точність та узагальненість моделей прогнозування, продемонструвавши відмінності в результатах залежно від контексту застосування. Однак у дослідженні [10] використовувався тільки один проєкт з використанням мови програмування C, що є не достатнім доказом того, що метод PMT здатний ефективно використовуватися в інших проєктах.

У роботі [12] було з'ясовано, що різниця при оцінюванні між різними інструментами за допомогою мутаційного рахунку є незначною. Однак набори мутантів рідко узгоджуються між собою щодо значень мутаційних рахунків. Це означає, що проведені дослідження значною мірою залежать від використаного інструменту. Крім того, деякі інструменти можуть вважати тестовий набір ефективним за одним критерієм, тоді як інші - ні, що робить проблематичним використання будь-якого цільового мутаційного рахунку як орієнтиру для тестування.

Використання кількості мутантів в якості метрики для оцінки обчислювальної вартості мутаційного тестування є потенційною загрозою валідності. Незважаючи на те, що існує кореляція між кількістю мутантів та швидкістю їх виконання, це не означає, що зменшення мутантів на визначений відсоток призводить до відповідного зменшення обчислювальної вартості. У роботі [13] були досліджені відмінності між вимірюванням кількості мутантів та вимірюванням часу виконання для оцінки зменшення обчислювальної вартості у мутаційному тестуванні. В результаті було виявлено, що з 75 проаналізованих робіт, лише 15 враховують час виконання, тоді як 54 з них є вразливими до зазначеної загрози. Таким чином, автори наводять емпіричні результати, які демонструють, що дана загроза валідності має суттєвий вплив на наукові висновки та інтерпретацію ефективності методів оптимізації мутаційного тестування.

У контексті використання методу CBUA, застосування статистичних розподілів, таких як розподіл Пуассона, які враховують часові характеристики процесу тестування, дозволяє побудувати більш реалістичну модель обчислювальних витрат. Це, в свою чергу, сприяє підвищенню точності оцінки ефективності мутаційного тестування та зменшує ризики, пов'язані з хибними уявленнями про вартість оптимізації на основі лише кількісних показників мутантів.

Висновки

У межах проведеного дослідження було здійснено систематизацію та аналіз сучасних підходів до підвищення ефективності методів прогнозування результатів мутаційного тестування. Зокрема, було виокремлено групу ознак, що є перспективними для подальшого вдосконалення моделей прогнозування; ідентифіковано основні загрози валідності експериментальних досліджень та окреслено підходи до моделювання мутаційного тестування; визначено ключові напрямки подальших досліджень.

Разом з тим, для суттєвого покращення ефективності таких методів необхідно провести додаткові дослідження щодо впливу низки факторів на результати прогнозування. Серед них варто відзначити: перевірку методів у різних середовищах виконання (у тому числі імплементацію моделей на різних мовах програмування), використання більш точних математичних моделей, інтеграцію додаткових інформативних ознак, таких як numTestExec, дослідження потенціалу історичних ознак, а також подальший аналіз впливу виявлених загроз валідності на точність прогнозування.

Внесок авторів

Антон САМОЙЛЕНКО - аналіз джерел, теоретичні засади дослідження; Ірина ЩЕРБИНА - концептуалізація, методологія.

Декларація про штучний інтелект

Автор не використовував штучний інтелект при створенні матеріалів статті.

Конфлікт інтересів

Автор заявляє про відсутність конфлікту інтересів та підтверджує, що під час підготовки цієї роботи не існувало жодних комерційних, фінансових чи інших взаємовідносин, які могли б бути розцінені як такі, що здатні вплинути на результати дослідження або їх інтерпретацію. Робота виконана відповідно до принципів академічної доброчесності, етичниих норм проведення наукових досліджень та вимог редакційної політики щодо запобігання конфлікту інтересів.

Список використаної літератури

1. Gopinath, R., Görz, P., Groce, A. Mutation Analysis: Answering the Fuzzing Challenge. Preprints, 2022, Article 2201.11303. https://doi.org/10.48550/arXiv.2201.11303

2. Laybourn, I. μ2: using mutation analysis to guide mutation-based fuzzing. Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: Companion Proceedings (ICSE '22), 331-333, 2022. https://doi.org/10.1145/3510454.3522682

3. Vikram, V., Laybourn, I., Li, A., Nair, N., O'Brien, K., Sanna, R., Padhye, R. Guiding Greybox Fuzzing with Mutation Testing. Proceedings of the 32nd ACM SIGSOFT International Symposium on Software Testing and Analysis, 929–941, 2023. https://doi.org/10.1145/3597926.3598107

4. Zhang, J., Zhang, L., Harman, M., Hao, D., Jia, Y., Zhang, L. Predictive Mutation Testing. IEEE Transactions on Software Engineering, 45(9), 898-918, 2019. https://doi.org/10.1109/TSE.2018.2809496

5. Zhang, P., Li, Y., Ma, W., Yang, Y., Chen, L., Lu, H., Zhou, Y., Xu, B. CBUA: A probabilistic, predictive, and practical approach for evaluating test suite effectiveness. IEEE Transactions on Software Engineering, 48(3), 1067–1096, 2020. https://doi.org/10.1109/TSE.2020.3010361

6. Papadakis, M., Kintis, M., Zhang, J., Jia, Y., Le Traon, Y., Harman, M. Mutation Testing Advances: An Analysis and Survey. Advances in Computers, 112, 275-378, 2019. https://doi.org/10.1016/bs.adcom.2018.03.015

7. Aghamohammadi, A., Mirian-Hosseinabadi, S.-H. The Threat to the Validity of Predictive Mutation Testing: The Impact of Uncovered Mutants. Preprints, 2020, Article 2005.11532. https://arxiv.org/abs/2005.11532

8. Gil, Y., Ma'ayan, D. Better Prediction of Mutation Score. Preprints, 2021, Article 14905032. https://doi.org/10.36227/techrxiv.14905032.v1

9. Mao, D., Chen, L., Zhang, L. An Extensive Study on Cross-Project Predictive Mutation Testing. 12th IEEE Conference on Software Testing, Validation and Verification (ICST), 160-171, 2019. https://doi.org/10.1109/ICST.2019.00025

10. Duque-Torres, A., Doliashvili, N., Pfahl, D., Ramler, R. Predicting Survived and Killed Mutants. IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW), 274-283, 2020. https://doi.org/10.1109/ICSTW50294.2020.00053

11. Inozemtseva, L., Holmes, R. Coverage is not strongly correlated with test suite effectiveness. Proceedings of the 36th International Conference on Software Engineering, 435-445, 2014. https://doi.org/10.1145/2568225.2568271

12. Gopinath, R., Ahmed, I., Alipour, M. A., Jensen, C., Groce, A. Does choice of mutation tool matter? Software Quality Journal, 25(3), 871–920, 2017. https://doi.org/10.1007/s11219-016-9317-7

13. Guizzo, G., Sarro, F., Harman, M. Cost measures matter for mutation testing study validity. Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, 1127-1139, 2020. https://doi.org/10.1145/3368089.3409742

A. Samoylenko, I. Shcherbyna

DIRECTIONS FOR IMPROVING THE EFFECTIVENESS OF THE CBUA METHOD WITH DYNAMIC TEST SUITES

In the context of increasing complexity in modern software systems, ensuring their reliability and quality remains one of the key challenges in software engineering. One of the promising directions is the use of mutation testing, which allows evaluating the quality of test suites by creating modified versions of the program — mutants. However, applying this approach under conditions of a dynamic test suite, where test cases are generated or extended during execution, makes it impossible to use a number of traditional optimizations. In response to these challenges, approaches for predicting mutation testing outcomes without actually executing the mutants have been proposed, including Predictive Mutation Testing (PMT). At the same time, the Coverage-Based Unsupervised Approach (CBUA), which maintains mutation score when the test suite changes, proves potentially more effective in such conditions.

The aim of this study is to systematize and analyze modern approaches to improving the effectiveness of methods for predicting mutation testing outcomes, with a focus on their applicability within the CBUA method. Special attention is given to analyzing strategies for enhancing the effectiveness of such methods, including the selection of relevant features for machine learning algorithms, the construction of adequate models of mutation testing, and the identification of threats to the validity of the results.

As a result of the study, a group of features promising for improving prediction models was identified, major threats to the validity of experiments were outlined, and directions for further research were defined. In particular, it is recommended to explore historical features in the context of mutation testing prediction methods, to utilize more effective static and dynamic features, and to develop more accurate mathematical models that take temporal characteristics into account to improve the overall effectiveness of the CBUA method.

Keywords: software quality, software testing, mutation testing, test outcome prediction.

Надійшла до редакції: 03.03.2026
Прийнята до друку: 21.04.2026
Опубліковано: 27.04.2026