УДК 004.415.53

О. В. Блистів, В. П. Карашецький*

аспірант кафедри Комп'ютерних наук, Національний лісотехнічний університет України, Львів, Україна. https://orcid.org/0009-0001-4802-1164.

blystiv.o@nltu.lviv.ua

* доцент кафедри Інженерії програмного забезпечення, Національний лісотехнічний університет України, Львів, Україна. karashetskyy@nltu.edu.ua

АНАЛІЗ ТА ОПТИМІЗАЦІЯ ТЕХНІК ПОПАРНОГО ТЕСТУВАННЯ ТА ОРТОГОНАЛЬНИХ МАСИВІВ ДЛЯ ЕФЕКТИВНОГО ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ З ВИКОРИСТАННЯМ МЕТОДОЛОГІЇ ПРІОРИТИЗАЦІЇ ТЕСТ-КЕЙСІВ

Анотація. У даній роботі розглядаються та порівнюються дві поширені методики тестування програмного забезпечення: попарне тестування (англ. pairwise testing) та ортогональні масиви. Обидві техніки використовуються для зменшення кількості тестових сценаріїв при одночасному забезпеченні максимального покриття функціональності програмного забезпечення. Проаналізовано їхні переваги та недоліки у різних контекстах застосування. Крім того, запропоновано нову формулу для пріоритизації тест-кейсів (наборів вхідних значень), яка включає такі фактори, як ризик, історію дефектів, складність, покриття та бізнес-цінність. Ця формула дозволяє визначати пріоритет тестів, що сприяє більш раціональному використанню ресурсів тестування і покращує загальну ефективність процесу тестування.

Ключові слова: попарне тестування, ортогональні масиви, пріоритизація тест-кейсів, покриття, оптимізація тестування.

Вступ

Методології тестування програмного забезпечення відіграють важливу роль у забезпеченні якості та стабільності розроблюваних систем. Однак тестування усіх можливих комбінацій параметрів є занадто ресурсо затратним, особливо для складних систем з великою кількістю вхідних даних [3]. Використання попарного тестування та ортогональних масивів, розроблених для оптимізації цього процесу, надають можливість зменшити кількість необхідних тестів без втрати якості.

Водночас постає питання щодо того, які тестові сценарії мають бути виконані першочергово, що спонукає до розробки методик для пріоритизації тестів.

Актуальність

У сучасних умовах швидкої розробки програмного забезпечення необхідно не лише зменшити кількість тестових сценаріїв, але й правильно розподілити їх пріоритетність. Це дозволить зосередитися на тестах, які мають найбільший вплив на якість та стабільність програмного продукту. Вибір між методиками попарного тестування і ортогональних масивів та застосування нової формули для пріоритизації тест-кейсів дозволяють досягнути значного підвищення ефективності тестування.

Мета роботи

Мета роботи полягає в аналізі технік попарного тестування і ортогональних масивів та застосуванні для них нової методики для пріоритизації тест-кейсів з метою ефективного тестування програмного забезпечення.

Виклад основного матеріалу

Техніка попарного тестування полягає у перевірці всіх можливих пар комбінацій параметрів[1]. Це дозволяє суттєво зменшити кількість тестів у порівнянні з тестуванням усіх можливих комбінацій, забезпечуючи при цьому достатнє покриття. Основна його перевага полягає в його оптимальності для

систем із великою кількістю параметрів, оскільки значно скорочує кількість тестів, не знижуючи їх ефективність.

Методика використання ортогональних масивів для тестування забезпечує більш повне покриття можливих комбінацій параметрів, зокрема для багатовимірних взаємодій [2]. Ця техніка часто використовується для тестування складних систем із великою кількістю взаємопов'язаних параметрів, де просте попарне тестування не може гарантувати повного охоплення критичних взаємодій. Проте ортогональні масиви потребують більшої кількості тестів і можуть бути менш ефективними з погляду оптимізації часу та ресурсів.

Для більш раціонального використання тестових ресурсів пропонується формула (1) для визначення пріоритету тест-кейса:

$$\begin{aligned}\Pr iority(TC) = \omega_1 * Risk(TC) + \omega_2 * Defects(TC) + \omega_3 * Complexity(TC) + \\ \omega_4 * Coverage(TC) + \omega_5 * Bu\sin essValue(TC),\end{aligned} \quad (1)$$

де:

TC – тест-кейс;

$\Pr iority(TC)$ – пріоритет тест-кейса;

$Risk(TC)$ – ризик тестового сценарію (ймовірність виявлення дефекту);

$Defects(TC)$ – історія дефектів, пов'язаних із цим тестом або подібними сценаріями;

$Complexity(TC)$ – складність тестового сценарію (кількість параметрів або взаємодій, які тестуються);

$Coverage(TC)$ – покриття тест-кейса (кількість унікальних комбінацій параметрів, які він перевіряє);

$Bu\sin ess(TC)$ – бізнес-цінність або критичність сценарію для кінцевих користувачів або функцій системи;

$\omega_1, \omega_2, \omega_3, \omega_4, \omega_5,$ – вагові коефіцієнти, що визначають важливість кожного з факторів.

Вагові коефіцієнти $\omega_1, \omega_2, \omega_3, \omega_4, \omega_5$ задаються командою на основі аналізу, обговорень і попереднього досвіду. Важливо, щоб цей процес був

прозорим і базувався на об’єктивних даних, що підвищить якість оцінок і спростить прийняття рішень у тестуванні.

Дана формула дає можливість динамічно визначати, які тест-кейси мають бути виконані першочергово, базуючись на ключових метриках: ризик, історія дефектів, складність, покриття та бізнес-цінність. Такий підхід дозволяє оптимізувати час тестування, зосередившись на найбільш важливих та ризикованих зонах програмного забезпечення.

Висновки

Використання методик попарного тестування та ортогональних масивів у процесі тестування значно підвищує його ефективність за рахунок зменшення кількості тестових сценаріїв при збереженні належного рівня покриття. Запропоновано формулу для пріоритизації тест-кейсів на основі п’яти ключових метрик, яка дозволяє більш раціонально розподіляти тестові ресурси, підвищуючи загальну якість і швидкість виконання тестів. Вибір між техніками залежить від складності програмного забезпечення та специфіки завдань, що тестуються. Впровадження нової методології пріоритизації сприятиме кращій організації тестування та забезпеченню стабільності програмного забезпечення.

Список використаних літературних джерел

[1] Practical Usage of Pairwise Testing. (20.06.2024). Retrieved from: https://adequatica.medium.com/practical-usage-of-pairwise-testing-8a8d00f86b9b.

[2] What is Orthogonal Array Testing? | Why & How to Perform? (22.07.2024). Retrieved from: https://testsigma.com/blog/orthogonal-array-testing/.

[3] Muthu Ramachandran, Rogério Atem de Carvalho (2009). Handbook of Research on Software Engineering and Productivity Technologies: Implications of Globalization, 241-255. doi:10.4018/978-1-60566-731-7.