ЯКІСТЬ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

УДК 004.052

Яковина В.С. , Федасюк Д.В.
Національний університет "Львівська політехніка"

УДОСКОНАЛЕНА ПРОЦЕДУРА ПРОГНОЗУВАННЯ ВІДМОВ ПРОГРАМНИХ ЗАСОБІВ НА ОСНОВІ МОДЕЛІ НАДІЙНОСТІ З ІНДЕКСОМ СКЛАДНОСТІ

Робота присвячена дослідженню шляхів удосконалення процесу прийняття рішення стосовно досягнення заданого рівня надійності програмних засобів за рахунок підвищення точності побудови кумулятивного профілю відмов програмного засобу на ранніх стадіях тестування. Показано, що використання регресійного аналізу на основі функції заданого вигляду дає можливість підвищити точність прогнозу відмов на 3–5%, або ж скоротити процес тестування програмного засобу (зі збереженням точності прогнозу відмов) на 20–40%.

Работа посвящена исследованию путей усовершенствования процесса принятия решений касательно достижения заданного уровня надежности программных средств за счет повышения точности построения кумулятивного профиля отказов программного средства на ранних стадиях тестирования. Показано, что использование регрессионного анализа на основании функции заданного вида позволяет улучшить точность прогноза отказов на 3–5 %, либо же сократить процесс тестирования программного средства (с сохранением точности прогноза отказов) на 20–40 %.

The paper is devoted to studying the ways to improve the decision-making concerning reaching an acceptability reliable software system. The task is reached by improvement the accuracy of cumulative failure profile building at the beginning of testing phase. It is shown that using the regression analysis with given function allows to improve the failures prediction accuracy by 3–5 %, or to reduce the software testing time (keeping the same accuracy) by 20–40 %.

Ключові слова: *надійність програмних засобів, прогнозування, кумулятивний профіль відмов, регресійний аналіз.*

Вступ

Надійність програмних засобів (ПЗ) є одним з найважливіших атрибутів їх якості. Розроблення моделей надійності ПЗ, які відповідають сучасному етапу розвитку індустрії, а також прикладних засобів, що формалізують використання таких моделей на практиці та надають підтримку в прийнятті рішень стосовно процесу розробки ПЗ заданого рівня надійності є актуальною науковою проблемою.

Одним з основних напрямків у галузі оцінювання і забезпечення надійності програмного забезпечення є аналіз процесу виникнення відмов у ПЗ і розробка математичних моделей, призначених для оцінки показників надійності [1]. За останні десятиліття було запропоновано багато аналітичних моделей для вирішення проблеми вимірювання надійності ПЗ. Ці підходи базуються головним чином на історії спостереження відмов ПЗ і можуть бути

класифіковані згідно з процесом дослідження наступним чином [2]: моделі на основі часу між відмовами, моделі на основі кількості відмов, моделі на основі висівання помилок, моделі на основі області вхідних даних.

Імовірнісні моделі надійності застосовують як для визначення показників надійності ПЗ згідно ДСТУ 2860-94 [3], так і для оцінювання метрик надійності ПЗ, якими, згідно стандарту IEEE 982 "Стандартний словник IEEE з вимірювань при виробництві надійного ПЗ" [4], серед інших можуть бути:

- кумулятивний профіль відмов;
- середній час виявлення *K* помилок;
- остаточна кількість помилок;
- частота відмов.

Зокрема кумулятивний профіль відмов використовують для прогнозування надійності ПЗ; оцінювання додаткового часу тестування, потрібного для досягнення заданого рівня надійності системи; визначення модулів та підсистем, які вимагають додаткового тестування тощо [4].

Однак процес визначення метрик надійності та прийняття рішень стосовно тривалості і вартості етапу тестування ПЗ все ще вимагає значних затрат, оскільки не може бути здійснений до досягнення критерію достатності тестування [5].

Метою цієї роботи є удосконалення процесу прийняття рішення стосовно досягнення заданого рівня надійності за рахунок підвищення точності прогнозування кількості відмов програмного засобу на ранніх етапах тестування.

Аналіз сучасного стану досліджень

Якість програмної системи зазвичай залежить від того, скільки часу займає тестування та які методології тестування застосовуються. З одного боку, чим більше часу люди витратять на тестування, тим більше помилок буде виправлено, і тим більш надійним буде програмне забезпечення; однак, вартість тестування ПЗ також зросте. З іншого боку, якщо на тестування витрачено дуже мало часу, вартість ПЗ може бути зменшена, але користувачі ризикують отримати ненадійне ПЗ [6]. Це також підвищить вартість програмного продукту під час експлуатації, адже значно дорожче виправити помилку на етапі експлуатації, ніж на етапі тестування. Отже, важливою проблемою інженерії програмного забезпечення є визначення оптимального моменту для припинення тестування і випуску (релізу) продукту.

У [7] наведено огляд і обговорення декількох узагальнених моделей затрат на розробку програмного продукту, що базуються на функціях надійності ПЗ (імовірності безвідмовної роботи), що задаються на основі неоднорідного пуассонового процесу. Використання таких моделей дає можливість ефективно спланувати ресурси для забезпечення вчасного та ефективного уведення в експлуатацію програмного продукту; визначити чи програмний продукт є достатньо надійним для релізу; отримати менеджерам чи розробникам підтримку в процесі прийняття рішень про перехід від поточного етапу тестування до випуску продукту. Все це забезпечується за рахунок визначення оптимальних політик тестування та випуску програмних систем на основі моделей затрат. Окрім вартостей, що входять до традиційних моделей затрат, моделі затрат, що розглядаються у [7], містять у собі такі параметри як вартість тестування, вартість відлагодження під час тестування, вартість відлагодження під час гарантійного терміну, та вартість ризиків програмної відмови. Ці моделі можуть бути використані для реалістичної оцінки повної вартості ПЗ для таких галузей, як телекомунікації, системи масового обслуговування, та вбудованих систем реального часу.

На практиці широко використовують моделі надійності ПЗ на основі кількості відмов. Предметом дослідження моделей цього класу є кількість відмов у визначеному часовому інтервалі, а не час між відмовами. По мірі усунення помилок з системи, очікується, що кількість відмов в одиницю часу спадатиме. Вважають, що кількість відмов відповідає відомому стохастичному процесу з дискретним або неперервним параметром потоку відмов, що залежить від часу. Параметри функції параметру потоку відмов можна оцінити на основі спостережених значень кількості відмов чи з часів між відмовами. Було запропоновано багато моделей, що описують такі явища. В основу більшості з цих моделей покладено розподіл Пуассона, параметри якого мають різний вигляд для різних моделей, оскільки використання такого розподілу випадкових величин добре зарекомендувало себе в

багатьох областях, де основна зацікавленість полягає в кількості подій [2].

Основним припущенням моделей на основі неоднорідного пуассонового розподілу є те, що $\mu(t)$ – кількість відмов ПЗ на проміжку часу $(0;t]$ – має пуассонів розподіл, а $\{\mu(t), t \geq 0\}$ відповідає неоднорідному пуассоновому процесу. Нехай $\lambda(t) = E[\mu(t)]$ є математичним сподіванням величини $\mu(t)$, або середньою функцією пуассонового процесу. Різні моделі надійності ПЗ на основі неоднорідного пуассонового процесу відрізняються виглядом функції $\lambda(t)$ [8].

Нещодавно була розроблена модель надійності ПЗ з індексом складності [9], яка відноситься до моделей на основі неоднорідного пуассонового процесу; та встановлено інтервали для індексу складності програмного продукту [10], які дають змогу класифікувати програмні засоби за їх складністю.

В цій моделі пропонується наступний вигляд середньої функції пуассонового процесу [9], яка має фізичний зміст параметру потоку відмов ПЗ [11]:

$$\lambda(t) = \alpha\beta^{s+1}t^{s}\exp(-\beta t). \quad (1)$$

де α – коефіцієнт, що визначає загальну кількість помилок в ПЗ, β – коефіцієнт, що характеризує загальну тривалість процесу виявлення помилок, s – індекс складності програмного засобу, що узагальнює S-подібну модель [12].

Для такої середньої функції процесу кумулятивна кількість відмов має вигляд:

$$\mu(t) = \int_0^t \lambda(\tau)d\tau = \alpha\left[-\beta^{s}t^{s}e^{-\beta t} + s\Gamma_{\beta t}(s)\right], \quad (2)$$

де $\Gamma_z(p) = \int_0^z t^{p-1}e^{-t}dt$, $(\mathrm{Re}\, p > 0)$, – неповна гама-функція.

Загальна кількість відмов ПЗ визначається кумулятивною функцією при $t \to \infty$:

$$\mu(\infty) = \alpha s\Gamma(s), \quad (3)$$

де $\Gamma(s)$ – гама-функція.

Таким чином, аналітичний вигляд моделі з показником складності дозволяє узагальнити вираз для загальної кількості відмов системи, яка залежить від величини та складності ПЗ і визначається параметрами моделі.

Опис експериментів

З метою коректного порівняння результатів використання моделі надійності з індексом складності в цій роботі використано результати експериментів, описаних в роботі [8], стосовно яких було зроблено висновок, щодо невідповідності їх розподілу пуассоновому, а відповідно моделі, в основі яких лежить такий розподіл, не достатньо коректно описують поведінку експериментальних даних [8].

Програма Space, використана в першому експерименті, широко використовувалась в дослідженнях, присвячених тестуванню ПЗ [13, 14]. В цьому експерименті в програму Space було внесено 38 помилок, після чого програма проходила цикл тестування [8]. Тестовим пулом (вхідною множиною) даних служили 13498 окремих тестових випадків, кожен з яких був файлом ADL. Вихідна програма Space без внесених помилок служила в якості еталону. Будь-яка відмінність між вихідними значеннями тестової та еталонної програм була індикатором відмови, спричиненої принаймні однією помилкою. Серед 38 внесених помилок дві не виявлялись жодним тестовим випадком з даного тестового пулу [8].

Програма Space пройшла процес автоматизованого випадкового тестування з використанням платформи SRATE (Software Reliability Analysis, Testing, and Evaluation), розробленої авторами роботи [8]. На початку програма містила 38 внесених помилок. Після відмови, видаляли одну і тільки одну помилку, що спричинила цю відмову. Процес тестування зупинявся після 1200-го тесту. Такий процес автори [8] називають "випробуванням". Таким чином, в кожному випробуванні було виконано 1200 тестів, незалежно від кількості відмов. Така процедура імітувала поширений випадок, коли процес тестування ПЗ припиняється після використання заданої кількості ресурсів тестування.

З метою набору достовірної статистики, для такого тестування було проведено 40 випробувань. Оскільки різні користувачі можуть використовувати ПЗ з різним операційним профілем, автори [2] випадково генерували різний тестовий профіль для кожного випробування. В кожному випробуванні тестові випадки вибирались один за одним з відповідного тестового

профілю, який міг мати або не мати однорідний розподіл імовірностей [8].

В якості об'єкту дослідження в другому експерименті використовувався програмний засіб під назвою SESD (Software Environment for Software Data collection) [8]. Функціональність цього засобу полягала в граматичному аналізі, що використовується в середовищах програмування для збирання програмних даних. Цей програмний засіб містив 3559 рядків коду мовою C++, серед яких 3179 – виконуваного коду. Вхідними даними для засобу SESD є будь-яка програма мовою C. Для такого входу засіб створює п'ять вихідних значень: кількість рядків коду; загальна кількість використання операторів; загальна кількість використання операндів; кількість унікальних використань операторів; кількість унікальних використань операндів. Засіб SESD був реалізований одним програмістом [8], а потім був предметом незалежного тестування. Під час процесу тестування було виявлено і задокументовано 28 помилок.

Для отримання експериментальних даних виявлені 28 помилок були повторно уведені в програму SESD. Після цього програма пройшла процес автоматизованого випадкового тестування з використанням тієї ж платформи SRATE. Програма SESD без уведених 28 помилок використовувалась в якості еталонної для оцінювання коректності отриманих результатів. Будь-яке відхилення між результатами виконання тестового випадку (test case) досліджуваною програмою та результатами виконання тестового випадку еталонною програмою розцінювалось як помилка програми. Одна і тільки одна помилка, що спричиняла відмову вилучалась з програми після виявлення відмови. Множина тестів містила 5477 тестових випадків, в якій один тестовий випадок відповідав одній програмі мовою C [8]. Ці програми мовою C були завантажені з мережі Інтернет. Під час тестування було виявлено, що 2 з 28 помилок не виявлялись на даній множині тестів. Оскільки в реальних умовах індустрії програмного забезпечення процес тестування може бути припинено до виявлення усіх помилок, в цьому експерименті кожен раунд тестування припинявся після виявлення 25 помилок, незалежно, скільки часу це займало. Було проведено 40 раундів тестування. Для кожного раунду створювався відповідний профіль тестування. Цей профіль являв собою деяку випадкову послідовність (з однорідним розподілом) вхідних файлів з множини тестів, які подавались на вхід програми SESD [8].

В якості часової шкали використовувалась кількість ітерацій (тестових випадків), як універсальна міра, що не залежить від зовнішніх факторів та умов тестування [2, 8]. Для використання розробленої моделі, експериментальні дані з роботи [8], які представляли у вигляді номерів ітерацій, на яких було отримано відмови, були розбиті на інтервали по 10 ітерацій, для яких рахувалась кількість відмов на відповідному інтервалі, після чого будувалась кумулятивна функція відмов.

Для досліджень, пов'язаних з визначенням критерію достатності процесу тестування, вхідні дані додатково опрацьовувались наступним чином:

- раунд тестування розбивали на фази, в даному випадку по 50 ітерацій;
- після кожних 50 ітерацій тестування на інтервалі $(0, t_i]$ отримували методом максимальної правдоподібності точкові оцінки параметрів моделі $\hat{\alpha}$, $\hat{\beta}$ та $\hat{s}$;
- після отримання точкових оцінок параметрів моделі на усьому інтервалі тестування отримували значення прогнозу кількості відмов згідно (3).

Такі дії мали на меті імітувати використання моделі надійності на етапі тестування проектів з розробки програмного забезпечення, де проводиться певна кількість тестувань, після чого кожного разу приймається рішення про продовження чи достатність етапу тестування.

Залежність точкових оцінок параметрів моделі надійності від профілю тестування

Для опису поведінки моделі з показником складності ПЗ при використанні на різних наборах тестових профілів було проведено дослідження параметрів моделі для різних раундів тестування та їх статистичний опис. На рис. 1 та 2 зображено залежності параметрів α (рис. 1) та s (рис. 2) моделі з показником складності від тривалості процесу тестування для різних тестових профілів.

Як видно з цих рисунків, усі залежності для різних раундів тестування є близькими між собою і проявляють однакові якісні

залежності з різким стрибком в околі t = 500, що підтверджує припущення про ефективність використання цих параметрів для побудови критерію достатності процесу тестування та визначення точки переходу вибірки вхідних даних до пуассонового розподілу [5].

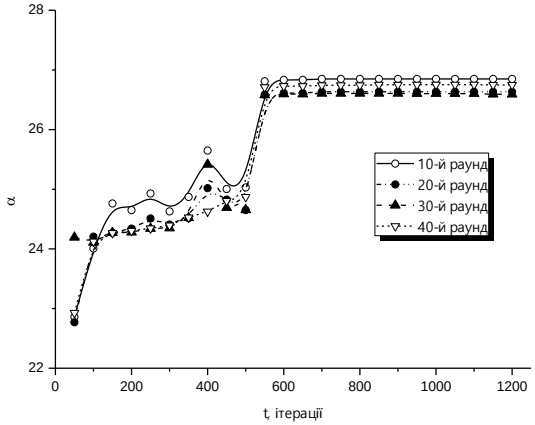

Рис. 1. Залежність параметру α моделі з показником складності від тривалості процесу тестування для різних тестових профілів.

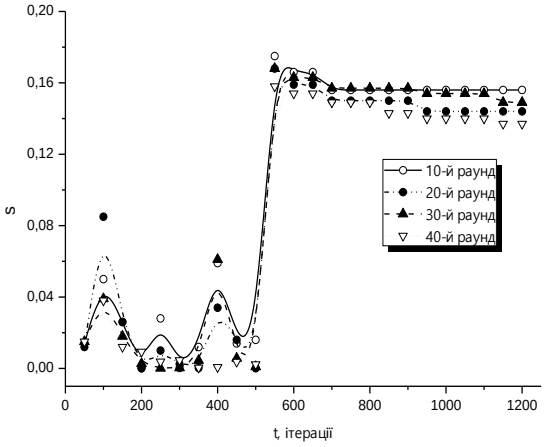

Рис. 2. Залежність параметру s моделі з показником складності від тривалості процесу тестування для різних тестових профілів.

Для кількісної характеристики однорідності розподілу точкових оцінок параметрів моделі з показником складності розраховано статистичні характеристики такого розподілу (табл. 1). Основною характеристикою в цьому випадку є структурна характеристика вибірки – медіана. Вона практично визначає структуру вибіркових даних і визначається через ці дані. Цей показник має статус основного або головного при асиметричному розподілі даних, причому, у випадку асиметрії розподілу медіана бере на себе роль середнього значення, крім того, медіана вважається найбільш стійкою характеристикою вибірки, а тому може бути основою для критерію оптимального розподілу даних в інтервалах. Як бачимо, в нашому випадку значення медіани вибірки кожного параметра практично співпадає з середнім значенням точкової оцінки кожного параметра, що підтверджує статистичну однорідність даних. Крім того розмах між мінімальним та максимальним значенням точкової оцінки кожного параметра не перевищує 15 % (а у випадку параметру α – 1 %) з дуже низькими значеннями дисперсії та стандартного відхилення, що також свідчить про однорідність вибірки.

Таблиця 1
Статистичні характеристики розподілу точкових оцінок параметрів моделі для різних вхідних наборів тестових даних

Параметр	α	β	s	$\alpha s\Gamma(s)$
Найбільше значення	26,849	0,015	0,156	25,08
Найменше значення	26,592	0,013	0,137	24,82
Розмах	0,257	0,002	0,019	0,26
Медіана	26,691	0,014	0,147	24,95
Середнє арифметичне значення	26,706	0,014	0,147	24,95
Дисперсія	0,014	$6{,}7\cdot10^{-7}$	$6{,}4\cdot10^{-5}$	0,013
Стандартне відхилення	0,12	$8{,}2\cdot10^{-4}$	$8{,}0\cdot10^{-3}$	0,11

Оскільки, середнє вибірки є незміщеною ефективною і правдивою оцінкою для математичного сподівання випадкової величини та враховуючи близькість значень медіани і середнього арифметичного, можна зробити висновок, що дані є статистично однорідні. Це дозволяє використовувати модель надійності програмного забезпечення з показником складності для прогнозування помилок різного типу.

Прогнозування відмов на основі критерію достатності процесу тестування

Експеримент 1.

Емпіричні дані першого експерименту адекватно описуються моделлю надійності з показником складності (2) про що свідчить близьке до одиниці значення коефіцієнту детермінації (R^2=0,999), та низьке значення середньої квадратичної похибки апроксимації – 0,05, при цьому така похибка для інших моделей надійності (S-подібної чи Goel–Okumoto) не була меншою за 0,1.

Точкові оцінки параметрів моделі з показником складності для першого експерименту становили $\hat{\alpha}$ =36,7, $\hat{\beta}$ =0,0086 та $\hat{s}$ =0,085. Розрахована за рівнянням (3) загальна кількість відмов ПЗ з використанням отриманих для усього етапу тестування параметрів моделі становить 35,1, тоді як в цей ПЗ було впроваджено 38 помилок.

Для визначення критерію достатності процесу тестування [5] будували залежності параметрів моделі надійності з показником складності в залежності від кількості ітерацій, на яких було припинено тестування для емпіричних даних першого експерименту з роботи [8]. Побудова таких залежностей відповідає практиці проведення тестування, коли здійснюється певна ітерація (цикл тестування) після якої оцінюється якість програмного продукту. Потрібно після кожного такого циклу описувати отримані результати моделлю надійності, а параметри моделі зображати графічно чи в таблиці.

В [5] показано відмінності у поведінці залежності значення параметра s від тривалості процесу тестування до та після переходу до пуассонового розподілу кількості відмов. Таку поведінку залежності $s(t)$ пояснено тим, що показник складності ПЗ є основним параметром, що визначає форму і функцію розподілу, а відповідно і густину розподілу імовірностей випадкової величини, яка у нашому випадку визначається кількістю відмов. На пізніх етапах тестування ПЗ, коли взаємозалежні помилки виявлені та усунені, а кількість відмов відповідає пуассоновому розподілу, параметр s вже практично не змінюється, а змінюються в основному кількісні характеристики (параметри α та β), що дало можливість формалізувати критерій достатності процесу тестування ПЗ таким чином [5]:

$$\frac{ds(t)}{dt} \xrightarrow[t\to\infty]{} 0 . \qquad (4)$$

Залежність функції (4), отриману шляхом чисельного диференціювання показника складності для першого ПЗ, наведено на рис. 3. Така залежність наочно ілюструє критерій достатності процесу тестування ПЗ і показує різку зміну форми кривої при переході експериментальних даних до пуассонового розподілу.

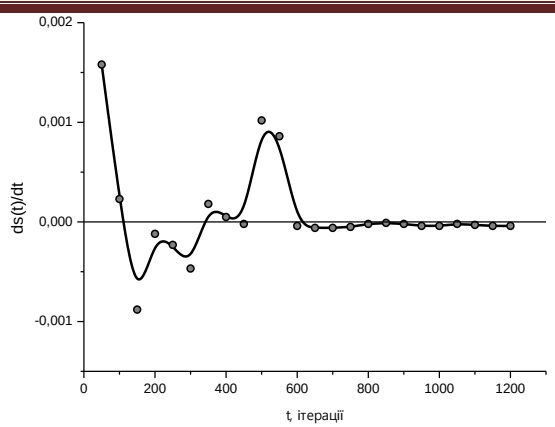

Рис. 3. Залежність критерію достатності процесу тестування ПЗ від тривалості тестування для першого експерименту.

Експеримент 2.

У випадку даних другого експерименту за статистичними характеристиками якості опису експериментальних даних різними моделями надійності можна зробити висновок, що модель з показником складності більш адекватно описує реальні експериментальні дані порівняно з традиційними S-подібною моделлю та моделлю Goel–Okumoto. Так, в цьому випадку найгірші статистичні показники якості опису експериментальних даних були у випадку використання моделі Goel–Okumoto – коефіцієнт детермінації є найменшим (0,974), а середня квадратична похибка найбільша (0,137). Модель з показником складності за середньою квадратичною похибкою суттєво (більш ніж на 20%) переважає S-подібну модель (0,080 проти 0,105), в той час як за значенням коефіцієнта детермінації незначно поступається S-подібній моделі (0,989 проти 0,995).

Точкові оцінки параметрів моделі з показником складності для другого експерименту становили $\hat{\alpha}$ =26,8, $\hat{\beta}$ =0,013 та $\hat{s}$ =0,137. Розрахована за рівнянням (3) загальна кількість відмов ПЗ з використанням отриманих для усього етапу тестування параметрів моделі становить 25,1, тоді як в цей ПЗ було впроваджено 28 помилок, з яких дві неможливо було виявити під час тестування.

Залежність критерію достатності процесу тестування (4) для даних другого експерименту наведено на рис. 4.

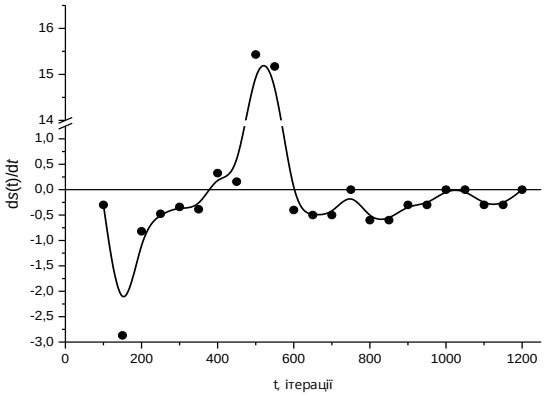

Рис. 4. Залежність критерію достатності процесу тестування ПЗ від тривалості тестування для другого експерименту.

Так само, як і у випадку першого експерименту, залежність показника складності ПЗ від тривалості процесу тестування виявляє чітку особливість, яку і було покладено в основу критерію достатності процесу тестування. Ця особливість полягає в тому, що при $t \geq 550$ (що знову таки відповідає переходу до пуассонового розподілу відмов) значення s наближається до постійної величини, на відміну від такої залежності при $t < 500$. Як видно з рис. 4, залежність критерію достатності процесу тестування показує наближення цього критерію до нуля при переході експериментальних даних до пуассонового розподілу.

Удосконалена процедура прогнозування відмов та аналіз отриманих результатів

З метою отримання інформації про поведінку прогнозованої загальної кількості дефектів в програмному продукті (див. рівняння (3)) було досліджено її залежність від тривалості етапу тестування. Для цього здійснено нелінійний регресійний аналіз критеріальної змінної μ_∞, отриманої з рівняння (3) відносно предикторів T_i, які означають час припинення тестування, а точкові оцінки параметрів моделі $\hat{\alpha}, \hat{\beta}, \hat{s}$, використані в рівнянні (3) для отримання відповідного значення $\mu_\infty = f(T_i)$, отримують методом максимальної правдоподібності [9] на інтервалі $(0; T_i]$. В якості рівняння регресії було обрано вираз вейбулівського типу:

$$\mu_\infty = A \cdot \left(1 - e^{-(k \cdot (T_i - T_c))^d}\right). \quad (5)$$

З виразу (4) можна отримати прогнозоване значення кількості відмов програмного продукту за припущення, що тривалість процесу тестування була безмежною ($T_i \to \infty$). Як видно з (5) це значення дорівнює значенню параметра регресії A. Таким чином, отримавши методом найменших квадратів параметри рівняння регресії, за значенням параметру A можна отримати уточнений прогноз граничної кількості відмов програмного продукту.

Графік залежності $\mu_\infty = f(T_i)$ для обох експериментів та відповідні лінії регресії наведено на рис. 5. Як видно з цього рисунку розкид значень прогнозу є більшим для першого експерименту, тоді як у випадку другого експерименту прогноз є практично стабільним починаючи приблизно з 200 ітерацій тестування. Величина зміни прогнозу зі збільшенням тривалості процесу тестування, імовірно, залежить від складності програмного засобу, однак остаточне вияснення цього питання потребує подальших досліджень.

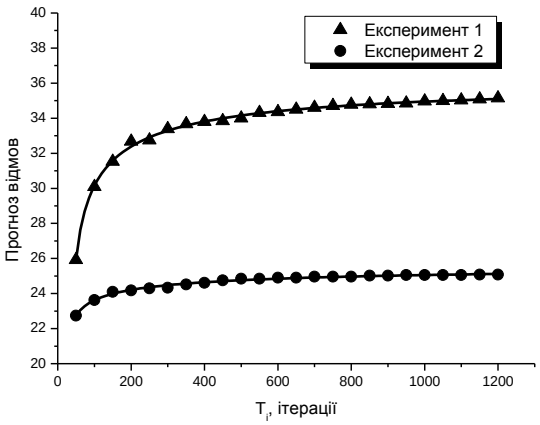

Рис. 5. Прогноз загальної кількості дефектів за моделлю з індексом складності в залежності від тривалості процесу тестування та лінія регресії за рівнянням (4) для першого і другого експериментів.

Після отримання чисельних значень параметрів рівняння регресії для визначення якості опису експериментальних точок рівнянням регресії було розраховано коефіцієнт детермінації (R^2) експериментальних значень та значень, отриманих з рівняння (5). Значення цього коефіцієнту та прогнозовані значення граничної кількості відмов за рівняннями (3) та (5) наведені в табл. 2 та 3 для першого та другого експериментів відповідно.

Як видно з табл. 2, використання описаної процедури та рівняння (5) дає змогу покращити прогноз на 2–3 %, або ж скоротити тривалість (а відповідно і витрати) тестування в даному прикладі вдвічі – з 1200 до 600 ітерацій з отриманням практично однакового прогнозу (35,1 дефекту).

Таблиця 2

Результати прогнозування кількості дефектів ПЗ для експерименту 1

Час, ітерацій	Прогноз за (3)	Прогноз за (5)	R^2	Різниця прогнозу, %
300	33,4	34,6	0,992	3,7
400	33,8	35,0	0,996	3,5
500	34,0	34,6	0,997	1,7
600	34,4	35,1	0,997	2,1
700	34,6	35,4	0,997	2,4
800	34,8	35,7	0,997	2,7
900	34,8	35,7	0,997	2,6
1000	35,0	35,7	0,997	2,2
1100	35,0	35,8	0,998	2,1
1200	35,1	35,8	0,998	2,0

Згідно з [5] для цього експерименту критерій достатності тестування, а отже мінімальна тривалість процесу тестування, досягався щонайменше при 650 ітераціях тестування. Однак у випадку такої мінімальної тривалості процесу тестування модель з індексом складності [9] прогнозує кількість дефектів в програмі на рівні 34–34,5, а використання запропонованого підходу на основі регресійного аналізу вже на цьому етапі дає прогноз на рівні 35,0–35,5 (див. табл. 2). З іншого боку, при проведенні повного циклу тестування, описаного в [8], використання моделі з індексом складності дає граничне значення біля 35 дефектів, тоді як використання регресійного аналізу прогнозів моделі – 36 дефектів, що ближче до реального значення.

Таблиця 3

Результати прогнозування кількості дефектів ПЗ для експерименту 2

Час, ітерацій	Прогноз за (3)	Прогноз за (5)	R^2	Різниця прогнозу, %
300	24,3	24,3	0,994	0,0
400	24,6	26,5	0,984	7,6
500	24,8	30,5	0,982	22,8
600	24,9	30,6	0,986	22,8
700	25,0	29,9	0,989	19,8
800	25,0	30,3	0,989	21,3
900	25,0	27,5	0,989	9,9
1000	25,1	26,5	0,990	5,7
1100	25,1	25,9	0,989	3,3
1200	25,1	25,7	0,990	2,4

Аналіз даних табл. 3 не такий однозначний, як у випадку першого експерименту (особливо у випадках, коли тестування припинялось після 500–800 ітерацій), однак також дозволяє зробити висновок про те, що використання запропонованої процедури регресійного аналізу з використанням виразу (5), покращує прогноз загальної кількості відмов ПЗ на 2–5 %, або ж дозволяє скоротити тривалість процесу тестування в даному випадку практично на третину – з 1200 до 900 ітерацій.

Висновки

Показано можливість удосконалення процесу прийняття рішення стосовно досягнення заданого рівня надійності ПЗ за рахунок підвищення точності прогнозування кількості відмов на ранніх етапах тестування.

Запропоновано процедуру на основі нелінійного регресійного аналізу критеріальної змінної, що має зміст прогнозованої кількості відмов, відносно предикторів, які означають тривалість тестування. В якості рівняння регресії було обрано вираз вейбулівського типу, з параметрів якого можна отримати новий, уточнений прогноз загальної кількості відмов досліджуваного програмного засобу.

Верифікація запропонованої процедури здійснювалась на основі ряду експериментів, що імітували використання моделі надійності на етапі тестування проектів з розробки програмного забезпечення, та використовували результати тестування реальних програмних продуктів.

Використання описаної процедури дає змогу покращити точність прогнозу на 3–5 %, або ж скоротити тривалість (а відповідно і витрати) тестування в середньому на 30–40%.

Список використаних джерел

1. Основи надійності цифрових систем [Текст] : підручник / В.С. Харченко, В.Я. Жихарев, В.М. Ілюшко, В.А. Краснобаєв, П.М. Куліков, І.В. Лисенко, М.В. Нечипорук, Г.М. Тимонькін. – Харків: Нац. аерокосм. ун-т "Харк. авіац. інт-т", 2004. – 573 с.

2. Goel A.L. Software reliability models: assumptions, limitations, and applicability / A.L. Goel // IEEE Transactions on software

engineering. – 1985. – Vol. SE-11, No 12. – P. 1411–1423.

3. ДСТУ 2860-94. Надійність техніки. Терміни та визначення. – Введ. 1996–01–01. – К. : Держстандарт України, 1994. – 33 с.

4. Std 982.1-1988. IEEE Standard Dictionary of Measures to Produce Reliable Software. – Published by The Institute of Electrical and Electronics Engineers, NY, USA, 1999. – 38 pages.

5. Яковина В.С. Критерій достатності процесу тестування програмного забезпечення / Яковина В.С., Сенів М.М., Чабанюк Я.М., Федасюк Д.В., Хімка У.Т. // Вісник Національного університету "Львівська політехніка" Комп'ютерні науки та інформаційні технології. – 2010. – № 672. – С. 346–358.

6. Zhang X. A software cost model with error removal times and risk costs / X. Zhang, H. Pham // International Journal of Systems Science. – 1998. – Vol. 29, No 4. – P. 435–442.

7. Pham H. System software reliability // Springer-Verlag London Limited, 2006. – 440 p.

8. Cai K.-Y. Does software reliability growth behavior follow a non-homogeneous Poisson process / K.-Y. Cai, D.-B. Hu, Ch.-G. Bai, H. Hu, T. Jing // Information and Software Technology. – 2008. – Vol. 50. – P. 1232–1247.

9. Чабанюк Я.М. Побудова і дослідження моделі надійності програмного забезпечення з індексом величини проекту / Я.М. Чабанюк, В.С. Яковина, Д.В. Федасюк, М.М. Сенів, У.Т. Хімка // Інженерія програмного забезпечення. – 2010. – № 1. – С. 24–29.

10. Яковина В.С. Моделювання параметра потоку відмов програмного забезпечення та визначення діапазонів показника його складності / В.С. Яковина // Вісник Національного університету "Львівська політехніка". Комп’ютерні системи та мережі. – 2014. – № 806. – С. 296–302.

11. Половко А.М. Основы теории надежности / Половко А.М., Гуров С.В. – СПб.: БХВ-Петербург, 2008. – 704 с.

12. Yamada S. S-shaped reliability growth modeling for software error detection / Yamada S., Ohba M., Osaki S. // IEEE Transactions on Reliability. – 1983. – Vol. R-32, No. 5. – P. 475–478.

13. Vokolos F.I. Empirical evaluation of the textual differencing regression testing technique / F.I. Vokolos, P.G. Frankl // International Conference on Software Maintenance : Proceedings. – 1998. – P. 44–53.

14. Rothermel G. Prioritizing test cases for regression testing / G. Rothermel, R.H. Untch, C. Chu, M.J. Harrold // IEEE Transactions on Software Engineering. – 2001. – Vol. 27 (10). – P. 929–948.

Відомості про авторів:

Яковина Віталій Степанович – доцент, кафедра програмного забезпечення, Національний університет "Львівська політехніка"; кандидат фізико-математичних наук. Наукові інтереси: надійність та безпека програмного забезпечення.

E-mail: yakovyna@lp.edu.ua

Федасюк Дмитро Васильович – проректор, Національний університет "Львівська політехніка", кафедра програмного забезпечення; доктор технічних наук. Наукові інтереси: автоматизація теплового проектування мікроелектронних систем, технології створення програмного забезпечення.

E-mail: fedasyuk@lp.edu.ua