

ПІДВИЩЕННЯ ДОСТОВІРНОСТІ ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

© Говорущенко Т., 2007

Метою дослідження є підвищення достовірності програмних продуктів за рахунок ідентифікації прихованих помилок через повторне тестування програмного забезпечення (ПЗ). Для досягнення мети розвинута концептуальна модель підвищення достовірності тестування ПЗ, розроблена категорійна модель повторного тестування ПЗ на базі штучних нейронних мереж (ШНМ), метод ідентифікації прихованих помилок ПЗ на основі ШНМ, удосконалено метод оцінки достовірності виявлення прихованих помилок ПЗ.

The goal of research is software product reliability increasing at the expense of hidden mistakes identification by repeated software testing. For goal achievement conceptual model of software testing reliability increasing was developed, category neuronet model of repeated software testing process was received, software hidden mistakes identification neuronet technique was developed, technique of evaluation of software hidden mistakes identification reliability was improved

Вступ

Надійність і достовірність роботи сучасних мікропроцесорних пристроїв, комп'ютерів та комп'ютерних систем є запорукою успішного розв'язання задач, що розв'язуються ними, та довготривалої їхньої експлуатації. Одним з основних засобів, за допомогою котрого досягається висока надійність правильного функціонування обчислювальної техніки та правильність розв'язання прикладних задач, є діагностування програмного забезпечення (ПЗ) на різних етапах його життєвого циклу, зокрема, на етапах проектування, розробки та експлуатації.

Тема діагностування програмного забезпечення одержала розвиток в працях відомих вітчизняних та іноземних вчених: Харченка В.С., Гуляєва В.А., Локазюка В.М., Савченка Ю.Г., Согомояна Е.С., Романкевича О.М., Кривулі Г.Ф., Дрозда О.В., Ліпаєва В.В., Майєрса Г., Канера Сема, Коммервіла І., Бейзера Б.

Аналіз програмного забезпечення як об'єкта діагностування дав можливість зробити такі висновки:

- 1) розвиток і впровадження нових архітектур та апаратна складність комп'ютерів і систем, побудованих на їхній базі, сьогодні випереджає розроблення методів і засобів діагностування системного та прикладного програмного забезпечення цих систем;
- 2) наявні діагностичні програми не завжди повністю враховують все вищі вимоги до розроблення програм, причини постійного ускладнення ПЗ та намагання користувачів розв'язувати за його допомогою важкоформалізовані та неформалізовані задачі;
- 3) низька якість окремих діагностичних програм знижує ефективність використання існуючого ПЗ комп'ютерних систем.

Однією з основних складових діагностування ПЗ є тестування як виявлення помилок у програмах. Його роль тим більше зростає в зв'язку з тим, що ПЗ сучасних комп'ютерних систем є доволі складним і не може бути бездефектним. Причиною невиявлення помилок у розроблюваному ПЗ швидше за все треба вважати недосконалість тестів, а не бездефектність програми.

З аналізу методів тестування ПЗ стає зрозумілим, що жоден з них не є універсальним і має певні недоліки.

Виявлення недоліків методів та засобів тестування ПЗ, зокрема ідентифікація прихованих помилок програмних продуктів, є актуальною задачею розвитку методів тестування ПЗ, що підвищує його достовірність.

Концепція повторного тестування програмного забезпечення

Прихованою помилкою назвемо будь-яку помилку ПЗ, що залишилась у програмному продукті після його діагностування під час розроблення та налагодження.

Основним тестуванням вважатимемо тестування дефектів (помилки), яке здійснюється на етапах розроблення та налагодження ПЗ і є частковою складовою цих процесів.

Повторним тестуванням вважатимемо тестування для виявлення прихованих помилок, яке здійснюється після розроблення та налагодження ПЗ і є окремим технологічним процесом.

Повторне тестування здійснюється на етапі вхідного контролю, який здійснює замовник, тобто допомагає замовнику оцінити якість тестування програмного забезпечення, яке приймається, і вказує на наявність в ньому прихованих помилок. Повторне тестування в паралельній каскадній моделі [1] наведено на рис. 1.



Рис. 1. Повторне тестування в паралельній каскадній моделі життєвого циклу програмного забезпечення

Відомий розподіл помилок за пріоритетами і категоріями [1]. Виконаємо уточнення розподілу помилок ПЗ щодо опису прихованих помилок [2, 3]. Всі приховані помилки розподілимо за видами на незначні (НПП), помірні (ППП), серйозні (СПП) та катастрофічні (КПП).

Незначними прихованими помилками (НПП) програмного забезпечення вважатимемо такі, що не впливають на дії користувача, програмний продукт з їхньою наявністю придатний для використання.

Помірними прихованими помилками (ППП) програмного забезпечення вважатимемо такі, що впливають на дії користувача. Програмний продукт з їхньою наявністю придатний для використання з частковою втратою функційності.

Серйозними прихованими помилками (СПП) програмного забезпечення вважатимемо такі, що призводять до помилкових результатів, внаслідок чого програмний продукт непридатний до використання.

Катастрофічними прихованими помилками (КПП) програмного забезпечення вважатимемо такі, що призводять до спотворення інформації (даних), внаслідок чого програмний продукт непридатний до використання і намагання його опрацювати призводить до відмови програмної системи.

Незначним прихованим помилкам присвоїмо найнижчий рівень категорійності – перший. Помірним прихованим помилкам присвоїмо, відповідно, рівень 2; серйозним – рівень 3. Найвищим рівнем вважатимемо катастрофічний – рівень 4.

Припустимо, що певна множина незначних помилок (**НПП'**) призводить до появи підмножини окремих типів помірних помилок (**ППП'**), які, своєю чергою, призводять до появи певної підмножини серйозних помилок (**СПП'**), а вони, відповідно, до катастрофічних помилок (**КПП'**).

Введемо поріг a_i допустимої кількості помилок і важливості помилок різних типів одного виду, при перевищенні якого необхідно здійснювати повторне тестування для виявлення прихованих помилок цього виду.

Категорійна модель повторного тестування

На основі запропонованої концепції підвищення достовірності тестування розробимо математичну модель повторного тестування, поклавши в її основу штучну нейронну мережу (ШНМ) типу прямонапрявленого перцептрона [2, 3].

Вибір апарату ШНМ мотивований тим, що штучні нейронні мережі дають можливість враховувати важливість (ваги) кожного типу неприхованих та прихованих помилок, пороги граничної кількості допустимих помилок кожної категорії, взаємний вплив прихованих помилок одних типів на помилки інших типів.

Задачею повторного тестування є визначення ваг впливу помилок різних типів однієї категорії на помилки іншої категорії, причиною яких є помилки попередніх категорій. Ця задача може бути розв'язана методом навчання ШНМ.

Відобразимо зазначений підхід узагальненою складною ШНМ, в якій структура багатошарового перцептрона типу MLP (multi-layer-perceptron) поєднується зі структурою простого перцептрона Розенблатта. Структура цієї ШНМ зображена на рис. 2.

Функціонал ШНМ Y_1 визначиться за формулою:

$$Y_1 = F_{S_1} \left(\sum_{i=1}^n x_i w_{x_i} - q_1 w_{q_1} \right).$$

Визначимо функціонал ШНМ Y_h :

$$Y_h = F_{S_h} \left(\sum_{i=1}^h \left(f(z^h) \cdot \left(\sum_{i=1}^h z_i^{h-1} w_{z_i^{h-1}, z_i^h} + \sum_{i=1}^{\ell} x_i^{z_i^h} w_{x_i^{z_i^h}, z_i^h} \right) \right) \cdot w_{z_i^h, S_h} \right) - q_h w_{q_h, S_h}$$

де $f(z^h)$ – активаційна функція нейронів прихованого шару h ; z_i^{h-1} – вихідне значення активності i -го нейрона прихованого шару ($h-1$); $w_{z_i^{h-1}, z_i^h}$ – ваговий коефіцієнт зв'язку між i -ми

нейронами прихованого шару ШНМ ($h-1$) і прихованого шару h ; $x^{z_i^h}$ – i -й вхід ШНМ, який безпосередньо пов'язаний з i -м нейроном шару h ; $w_{x^{z_i^h}, z_i^h}$ – ваговий коефіцієнт зв'язку між входом $x^{z_i^h}$ і i -м нейроном шару h ; $w_{z_i^h, S_h}$ – ваговий коефіцієнт зв'язку між i -м нейроном шару h і нейроном S_h .

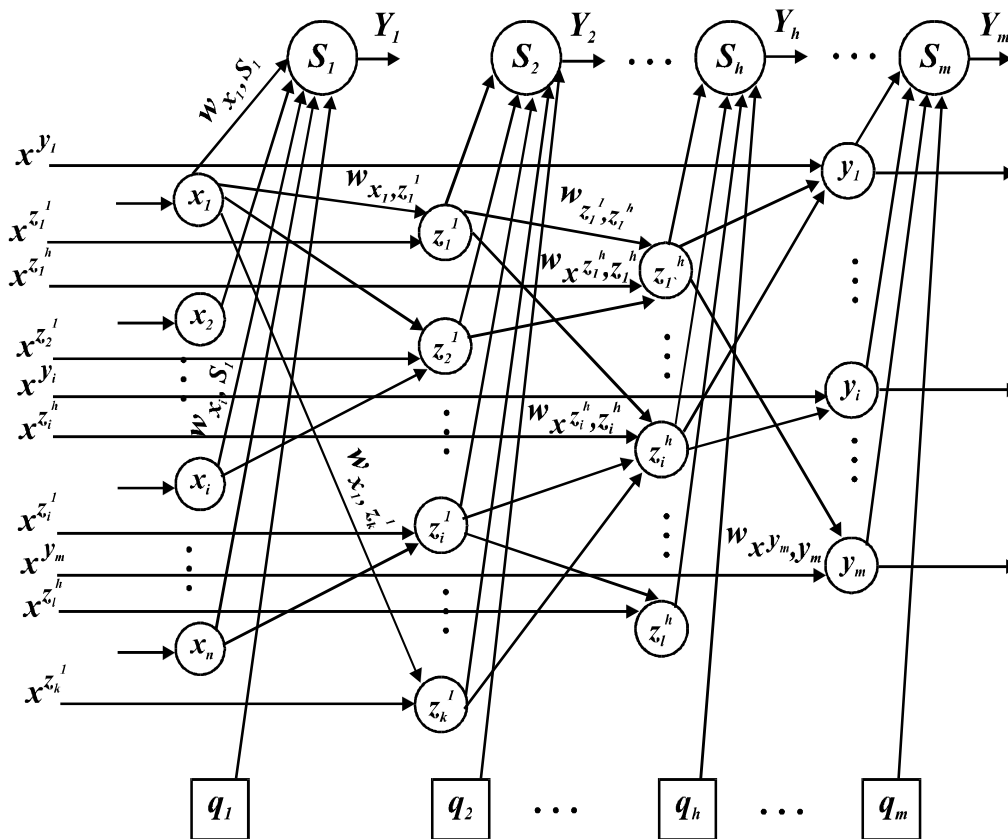


Рис. 2. Категорійна модель на базі ШНМ, що відображає зв'язок помилок ПЗ різних категорій

Активізаційною функцією нейронів прихованих (асоціативних) шарів є функція гіперболічного тангенсу. Активізаційною функцією нейронів ефекторних шарів є лінійна функція. Результати лінійної активізаційної функції нейронів ефекторних шарів лежать в інтервалі $[-1; 1]$. Нас цікавлять результати у вигляді 1 або 0 (відповідно є помилка i -го рівня категорійності чи немає), тому здійснюється перетворення (заокруглення) результатів:

$$Y_i = \begin{cases} 1, & \text{якщо } Y_i > 0; \\ 0, & \text{якщо } Y_i \leq 0. \end{cases}$$

Оцінка достовірності ідентифікації прихованих помилок програмного забезпечення

Із запропонованої моделі випливає, що при $Y_h > 0$ у програмі є помилки категорії, якій відповідає Y_h . Нехай без врахування впливу на початку у програмі було p_x помилок першого рівня категорійності, p_{z^1} – помилка другого рівня категорійності, ..., p_{z^h} – помилка h -го рівня категорійності. З урахуванням впливу помилок попереднього рівня категорійності на наступний помилка стало: p_x – першого рівня категорійності, $p_{z^1} + p_{z^h-1}$ – h -го рівня категорійності.

За критерій достовірності ідентифікації помилок ПЗ приймемо кількість виявлених помилок згідно із запропонованою моделлю.

Достовірність D ідентифікації прихованих помилок ПЗ за допомогою повторного тестування дорівнюватиме:

$$D = kn_1 p_x + kn_2 \frac{p_{z^1} + p_{x_i}}{p_{z^1}} + \dots + kn_h \frac{p_{z^h} + p_{z_i^{h-1}}}{p_{z^h}} + \dots,$$

де $KN = \{kn_h\}$ – множина коефіцієнтів нормування категорійності прихованих помилок.

Підвищення достовірності ідентифікації прихованих помилок дорівнюватиме $\Delta D = 1 - \frac{D'}{D}$, де D' – достовірність ідентифікації прихованих помилок ПЗ без урахування впливу прихованих помилок кожного попереднього рівня категорійності на помилки наступного рівня категорійності.

Для згаданих раніше чотирьох рівнів категорійності:

$$D = kn_1 \cdot p_x + kn_2 \cdot \frac{p_{z^1} + p_{x_i}}{p_{z^1}} + kn_3 \cdot \frac{p_{z^2} + p_{x_i^{z^1}}}{p_{z^2}} + kn_4 \cdot \frac{p_{z^3} + p_{x_i^{z^2}}}{p_{z^3}}.$$

З j -ї вибірки одержано значення величин для визначення достовірності повторного тестування (табл. 1).

Таблиця 1

Значення величин для визначення достовірності повторного тестування

№ експ.	p_x	p_{z^1}	p_{z^2}	p_{z^3}	p_{x_i}	$p_{x_i^{z^1}}$	$p_{x_i^{z^2}}$
1	28	16	10	4	6	4	2
2	32	20	10	6	8	6	2
3	46	28	14	8	10	6	4
4	50	30	18	10	12	8	2

Експертно присвоїмо такі значення коефіцієнтам нормування категорійності прихованих помилок: $kn_1 = 0,08$; $kn_2 = 0,22$; $kn_3 = 1,7$; $kn_4 = 8$.

Достовірність виявлення прихованих помилок при повторному тестуванні з j -ї вибірки подана на графіку (рис. 3).

Підвищення достовірності ідентифікації прихованих помилок дорівнюватиме:

$$\Delta D_1 = 1 - \frac{12,16}{16,923} = 0,28; \dots \Delta D_4 = 1 - \frac{13,92}{16,364} = 0,15$$

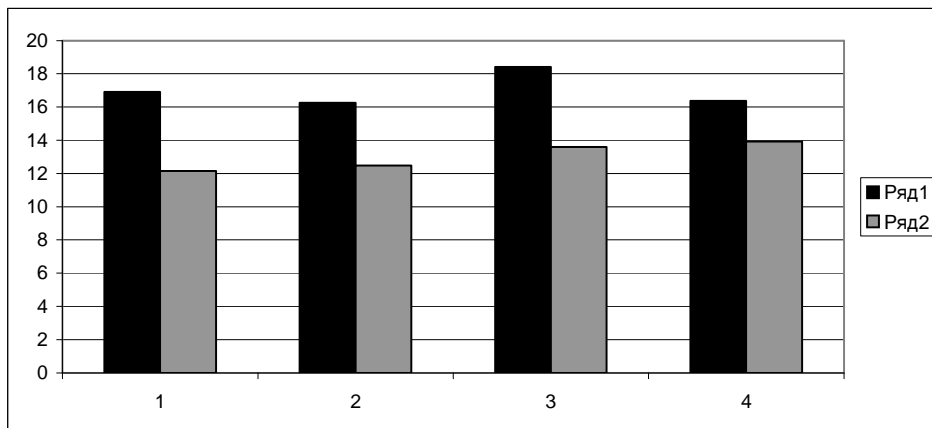


Рис.3. Гістограма підвищення достовірності

На гістограмі ряд 1 (чорний колір) показує достовірність D ідентифікації прихованих помилок при повторному тестуванні, а ряд 2 (сірий колір) – відображає достовірність D' ідентифікації прихованих помилок ПЗ без урахування впливу прихованих помилок кожного попереднього рівня категорійності на помилки наступного рівня категорійності.

Урахування впливу помилок попередніх рівнів категорійності підвищило достовірність виявлення прихованих помилок на 15–28%.

Модель та програмна реалізація ШНМ у пакеті Matlab

У пакеті Matlab було виконано програмну реалізацію моделі ШНМ [4, 5]. Структурну схему імітаційної моделі ШНМ в пакеті Matlab подано на рис. 4.

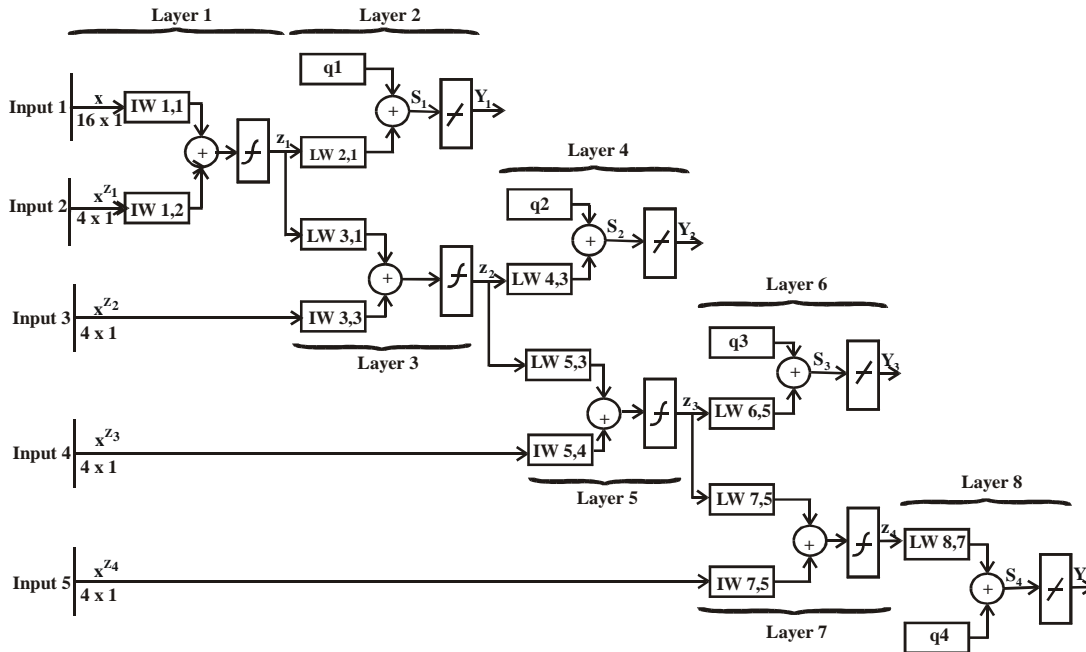


Рис. 4. Структурна схема імітаційної моделі ШНМ у пакеті Matlab

Оператор `gensim(net)` дає змогу одержати модель у пакеті Simulink (рис. 5, 6).

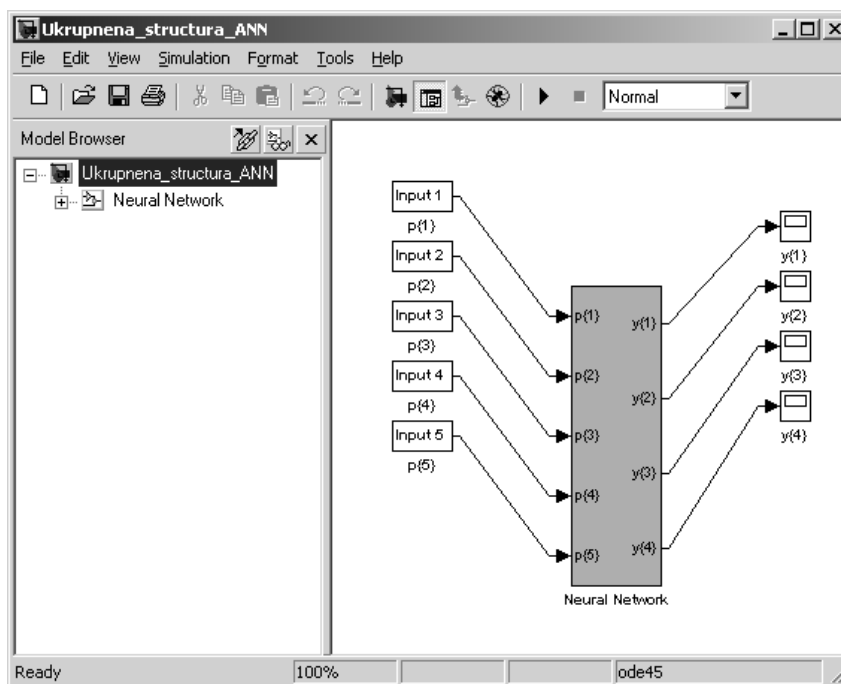


Рис. 5. Укрупнена структурна схема ШНМ у пакеті Simulink

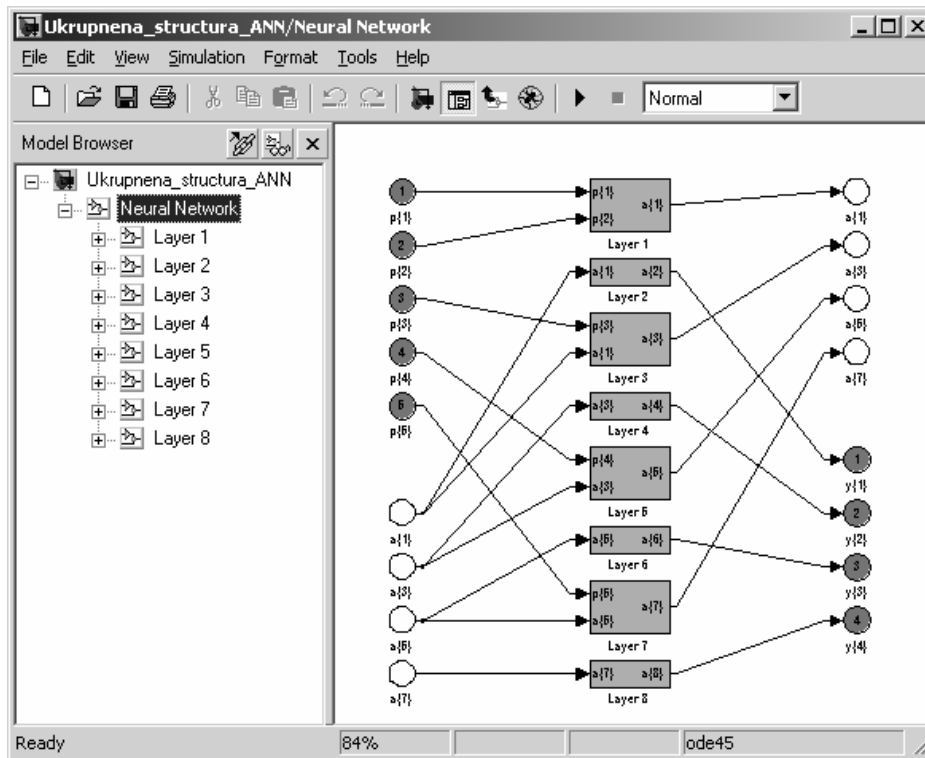


Рис. 6. Структурна схема шарів ШНМ у пакеті Simulink

Для вибору алгоритму навчання ШНМ та критерію оцінки якості навчання ШНМ досліджувалась при навчанні вибіркою з 2250 векторів різними алгоритмами з використанням різних критеріїв оцінки якості навчання [5, 6]. Для розрізнення значень вибірки, які відрізняються на порядок, було вирішено масштабувати навчальну вибірку за допомогою функції `premnmx`.

Максимальна похибка, якої було досягнуто при використанні комбінованого критерію якості навчання та масштабованої навчальної вибірки з 2250 векторів, становить 0.448359. Більшої точності навчання досягати не потрібно, оскільки виходи мережі, які містяться в інтервалі $[-1; 1]$ перетворюються для подання цілими значеннями 1 або 0 (є чи немає помилки i -го рівня категорійності відповідно).

Для тестування ШНМ було побудовано тестову вибірку з 200 векторів, яка також підлягала масштабуванню. Навчання і тестування [5, 6] різними алгоритмами навчання з використанням комбінованого критерію якості відображається на рис. 7, 8. На рисунках нижня крива відображає графік навчання, а верхня крива відображає графік тестування ШНМ.

Аналізуючи графіки навчання і тестування ШНМ, можна зробити висновок, що для навчання мережі найкраще підходить алгоритм навчання CGB на основі методу спряженого градієнта з оберненим поширенням і рестартами в модифікації Пауела–Біеле та його модифікації (алгоритм навчання Флетчера–Рівса або алгоритм навчання Полака–Рібейри) з використанням комбінованого критерію оцінки якості навчання.

Програмні засоби ідентифікації прихованих помилок програмного забезпечення

На рис.9 запропоновано структуру системи ідентифікації прихованих помилок програмного забезпечення [7], яка дає змогу користувачу на основі звіту про результати основного тестування одержати висновок про необхідність повторного тестування та про метод, яким рекомендується здійснювати повторне тестування.

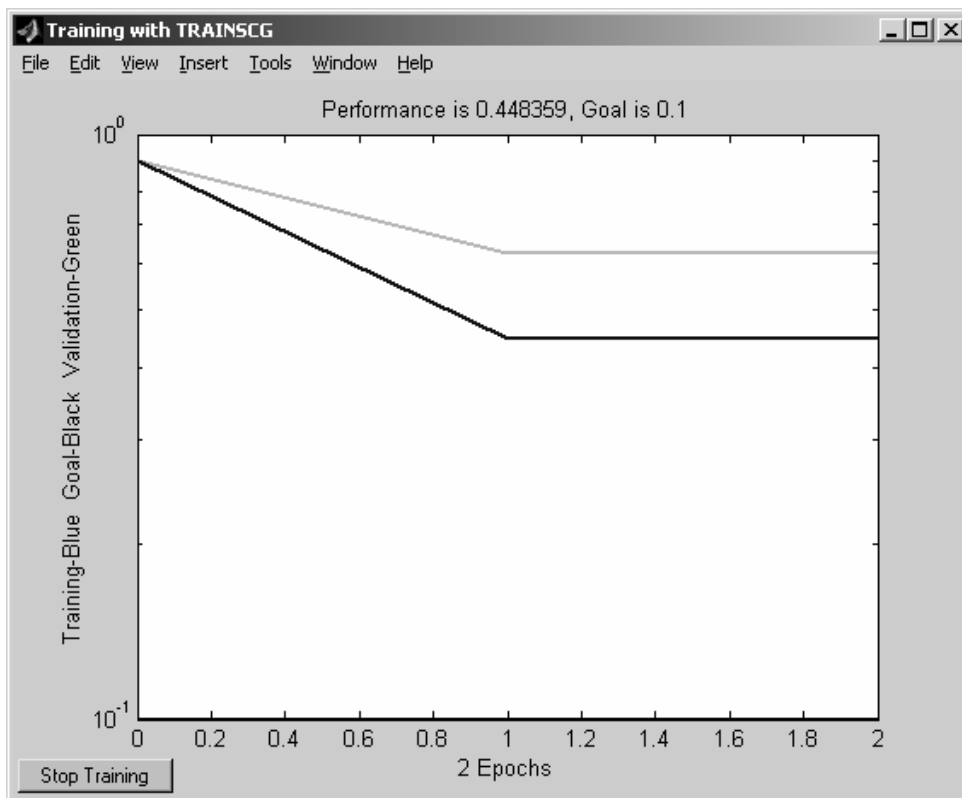


Рис. 7. Графіки навчання і тестування ШНМ за алгоритмом навчання SCG

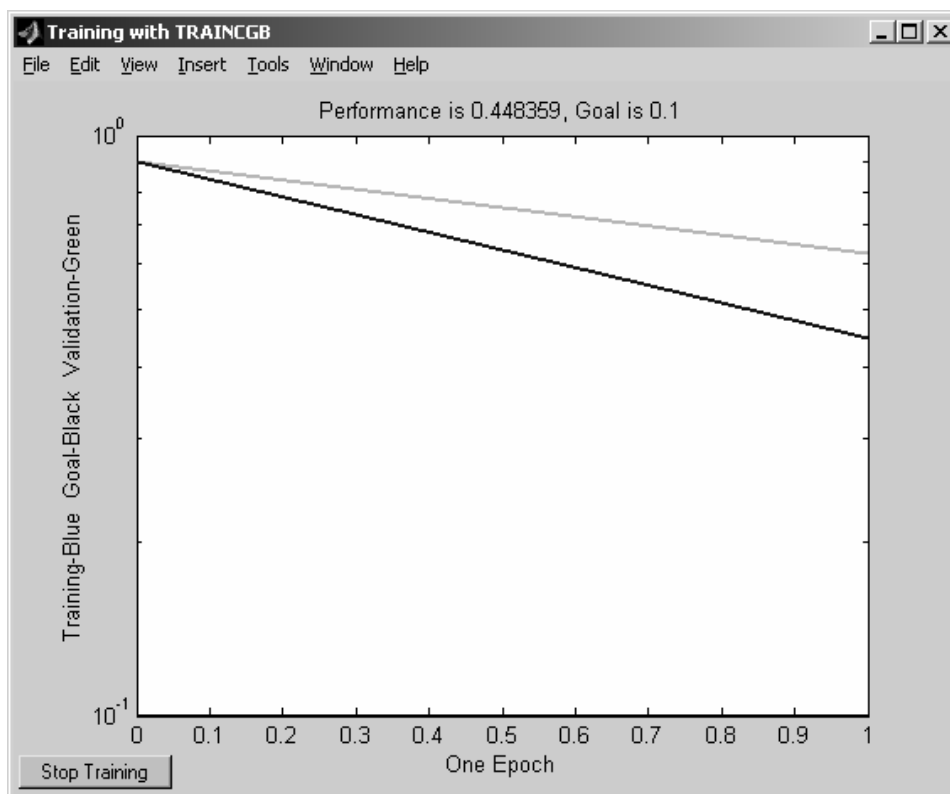


Рис. 8. Графіки навчання і тестування ШНМ за алгоритмом навчання CGB

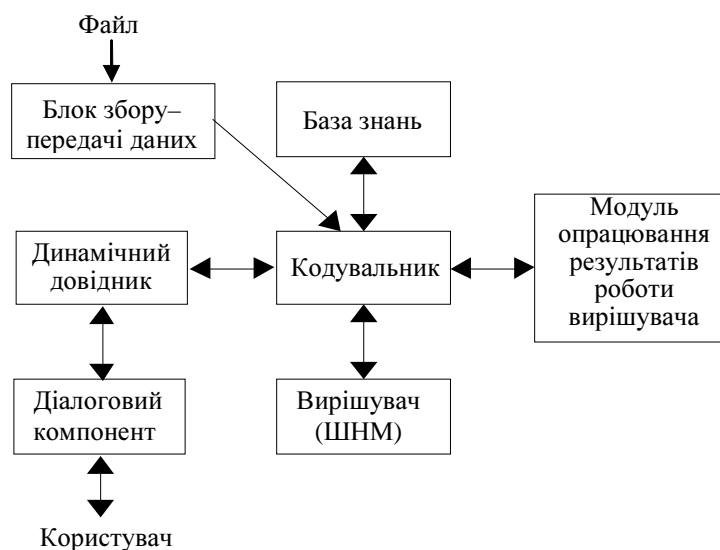


Рис. 9. Структурна схема системи ідентифікації прихованих помилок програмного забезпечення

Система ідентифікації прихованих помилок програмного забезпечення складається з таких компонентів:

1. Блок збирання – передавання даних – підключає наданий користувачем файл з результатами основного тестування, поданими у вигляді журналу “Метод тестування – Операція тестування – Тип виявленої помилки”.

2. Кодувальник – виконує перетворення вхідних даних з лінгвістичної форми подання в кількісну форму за допомогою таблиць 1–3 бази знань (додаток А), заповнення табл. 5 бази знань (додаток А) вхідними даними та формування вхідних векторів для модуля вирішувача. Кодувальник також перевіряє вхідні дані на правильність та достатність при формуванні вхідних векторів; якщо дані недостатні або вони неправильні, то кодувальник передає динамічному довідникові своє повідомлення з пропозицією сформулювати ще один файл такої самої форми, як і попередній, з додатковими результатами, що перетворюються в кількісну форму аналогічно даним основного файлу, після чого заносяться в базу знань. Здійснюється заповнення бази знань вихідними даними, перетворення результуючих векторів вирішувача з кількісної в лінгвістичну форму за допомогою табл. 4 бази знань (додаток А) та передавання їх модулю опрацювання результатів роботи вирішувача.

3. База знань містить таблиці присвоєння номерів методам і операціям основного тестування, типам виявлених помилок та присвоєння номерів рівням категорійності прихованих помилок; таблицю кількісного подання вхідних даних, в якій містяться вхідні дані, перетворені кодувальником в кількісну форму; таблицю текстового представлення результуючих векторів вирішувача (ШНМ), в якій подані результуючі вектори, перетворені кодувальником в лінгвістичну форму; таблиці відповідності методу основного тестування, операцій основного тестування, типів виявлених під час основного тестування помилок, відповідності між номером методів тестування ПЗ та рівнем категорійності прихованих помилок ПЗ, відповідності між операціями тестування ПЗ та рівнем категорійності прихованих помилок, на основі яких система формує висновок про метод, яким рекомендується здійснювати повторне тестування прикладного ПЗ, а також правила для формування висновку про необхідність та метод повторного тестування.

4. Вирішувач – штучна нейронна мережа, на входи якої подається інформація про методи і операції основного тестування та типи виявлених під час основного тестування помилок, а на виході одержується рівень категорійності прихованих помилок.

5. Модуль опрацювання результатів роботи вирішувача – на основі правил та таблиці результатів роботи вирішувача, взятих з бази знань, генерує висновок про необхідність та метод повторного тестування, який передається користувачу через кодувальник, динамічний довідник та діалоговий компонент.

6. Динамічний довідник – надає користувачу довідку про формат вхідного файлу, про відомі системі методи і операції основного тестування ПЗ, типи виявлених під час основного тестування помилок ПЗ, а також подає у наочній формі всі повідомлення будь-якого з компонентів системи.

7. Діалоговий компонент – візуалізує повідомлення динамічного довідника та видає їх користувачу в зрозумілій для сприйняття формі.

Запропонована система ідентифікації прихованих помилок програмного забезпечення дає змогу користувачу на основі звіту про результати основного тестування одержати висновок про необхідність повторного тестування, а саме: про наявність у програмному забезпеченні прихованих помилок та про метод, яким рекомендується здійснювати повторне тестування. У цій системі кодувальник виступає як інтерфейс між користувачем і системою, а динамічний довідник – як інтерфейс між користувачем та кодувальником.

Систему ідентифікації прихованих помилок програмного забезпечення було реалізовано в Borland C++ Builder 6.0 [7].

Після активізації системи користувачем на екрані з'являється таке вікно (рис. 10).

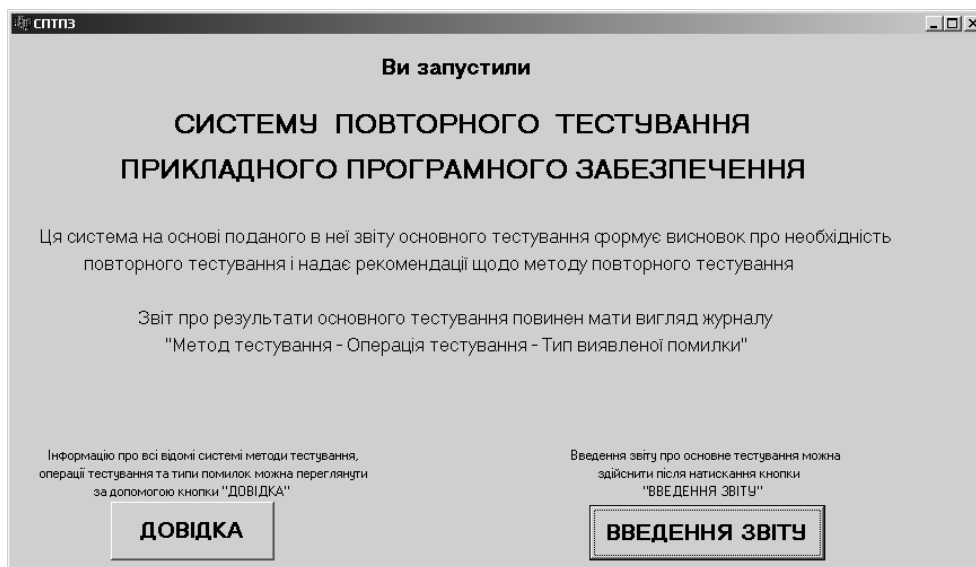


Рис. 10. Перше діалогове вікно системи ідентифікації прихованих помилок ПЗ

При натисканні в першому діалоговому вікні кнопки “ВВЕДЕННЯ ЗВІТУ” користувач має змогу подати по рядку звіт в систему повторного тестування прикладного програмного забезпечення (рис. 11).

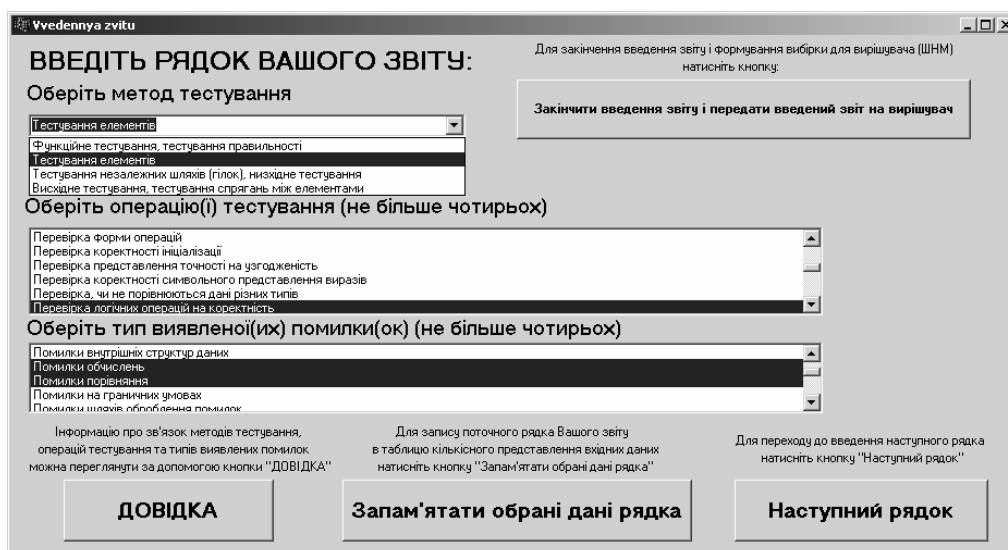


Рис. 11. Введення рядка звіту

Висновки

Дістала подальший розвиток концептуальна модель ідентифікації прихованих помилок програмного забезпечення, суть якої полягає у тому, що їхнє виявлення здійснюється після розроблення і налагодження ПЗ, де тестування програмного продукту здійснюється як часткова технологічна операція, тобто на впровадженому у життєвий цикл етапі повторного тестування, що може відповідати вхідному контролю.

Вперше одержано категорійну модель повторного тестування ПЗ на базі ШНМ, котра відрізняється від відомих тим, що в ній враховується вплив прихованих помилок різних типів попередньої категорії на виникнення помилок наступної категорії. Це дає можливість оцінити сумарний вплив помилок цієї категорії на якість ПЗ і зробити висновок щодо необхідності повторного тестування ПЗ.

Удосконалено метод оцінки достовірності ідентифікації прихованих помилок ПЗ. Досліджено, що підвищення достовірності буде тим більше, чим більше буде виявлено прихованих помилок, що впливають на виникнення помилок на наступному рівні категорійності. У розглянутому в статті прикладі підвищення достовірності становило від 15 до 28%.

1. Калбертсон Р., Браун К., Кобб Г. *Быстрое тестирование: Пер. с англ.* – М.: Издательский дом “Вильямс”, 2002. – 384 с.
2. Локажук В.М., Пантелєєва (Говоруценко) Т.О. Категорійна модель процесу повторного тестування дефектів програмного забезпечення // *Вісник Технологічного університету Поділля – Хмельницький: ТУП*, 2004. – Ч.1. – Т.1. – С. 53–58.
3. Lokazyuk V.M., Govoruschenko T.O. *Category Model of Process of Repeated Software Testing // Proceedings of the Third IEEE Workshop on Intelligent Data Acquisition and Advanced Computing Systems : Technology and Applications.* – Sofia, Bulgaria, 2005. – p. 241–245.
4. Локажук В.М., Поморова О.В., Говоруценко Т.О. Імітаційна модель системи повторного тестування програмного забезпечення // *Вісник ХНУ – Хмельницький: ХНУ*, 2006. – № 6. – С. 65–72.
5. Говоруценко Т.О. Дослідження моделі вирішувача системи повторного тестування прикладного програмного забезпечення // *Вісник Хмельницького національного університету – Хмельницький: ХНУ*, 2007 – №3. – Т.1. – С. 236–244.
6. Govoruschenko T.O. *Model of decision maker of repeated application software testing system // Радиоелектронні і комп’ютерні системи – Харків: НАУ “ХАІ”*, 2007 – № 7. – С. 191–198.
7. Говоруценко Т.О. Реалізація та функціонування системи повторного тестування прикладного програмного забезпечення // *Вісник Хмельницького національного університету – Хмельницький: ХНУ*, 2007 – №2. – Т.2. – С. 113–120.