3. Цмоць І. Г. Інформаційні технології та спеціалізовані засоби обробки сигналів і зображень у реальному часі / Цмоць І. Г. – Львів: УАД, 2005. – 227 с.

4. Параллельная обработка информации: в 5т. / АН УССР. Физ-мех. ин-т. Проблемно-ориентированные и специализированные средства обработки информации / А. И. Аксенов, В. В. Аристов, Е. Ю. Барзилович и др.; под ред. Б. Н. Малиновского и Грицика В. В. К.: Наукова думка, 1990. – 504 с.

5. Грушицкий Р. И. Проектирование систем на микросхемах программируемой логики / Грушицкий Р. И., Мурсаев А. Х., Угрюмов Е. П.. – СПб.: БХВ-Петербург, 2002. – 608 с.

6. Воеводин В. В. Параллельные вычисления / В. В. Воеводин, Вл. В. Воеводин. – СПб.: БХВ-Петербург, 2002. – 608 с.

7. Немнюгин С. А. Параллельное программирование для многопроцессорных систем / С. А. Немнюгин, О. Л. Стесик. – СПб.: БХВ – Петербург, 2002. – 400 с.

8. Касьянов В. Н. Графы в программировании: обработка, визуализация и применение / В. Н. Касьянов, В. А. Евстигнеев. – СПб.: БХВ – Петербург, 2003. – 1104 с.

Надійшла 21.3.2011 р.

УДК 004.891.3: 004.3

О.В. ПОМОРОВА, Т.О. ГОВОРУЩЕНКО, О.С. ОНИЩУК
Хмельницький національний університет

ОЦІНЮВАННЯ РЕЗУЛЬТАТІВ ПРОЕКТУВАННЯ ТА ПРОГНОЗУВАННЯ ХАРАКТЕРИСТИК ЯКОСТІ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

В статті показано місце нейромережного методу оцінювання результатів проектування та прогнозування характеристик якості програмного забезпечення (НМОП) в процесі розроблення програмного забезпечення за методологією Safety Case, обрано архітектуру штучної нейронної мережі для реалізації НМОП, описано методику навчання та результати функціювання нейромережної складової НМОП.

In the article the location of the artificial neural net's method of design results evaluation and software quality characteristics prediction (NMEP) is demonstrated in software development process for Safety Case methodology, ANN architecture for the NMEP realization is selected, ANN training methodology and results of neural net's component functioning is described

Ключові слова: програмне забезпечення (ПЗ), методологія Safety Case, метрики якості та складності програмного забезпечення, штучна нейронна мережа (ШНМ), нейромережний метод оцінювання результатів проектування та прогнозування характеристик якості програмного забезпечення (НМОП).

Вступ

Методологія Safety Case (Safety computer-aided software engineering – безпечна автоматизована програмна інженерія) розвивається вже більше 20 років [1– 4]. Первинним об'єктом методології Safety Case є мінімізація ризиків безпеки та комерційних ризиків програмного забезпечення шляхом побудови звіту, який повинен забезпечити докази, причини, аргументи та свідчення того, що програмне забезпечення (ПЗ) є безпечним, і всі вимоги до ПЗ належним чином виконані. Наразі ця методологія стала загальноприйнятою, однак рівень її автоматизації залишається низьким.

Процес створення безпечного ПЗ пов'язаний і залежить від значної кількості документів (вимоги, стандарти, специфікації), вихідного коду, методів оцінювання ПЗ та аналізу їх результатів, результатів тестування та ступеня їх документування. Рис. 1 [5] представляє узагальнену модель методології Safety Case.

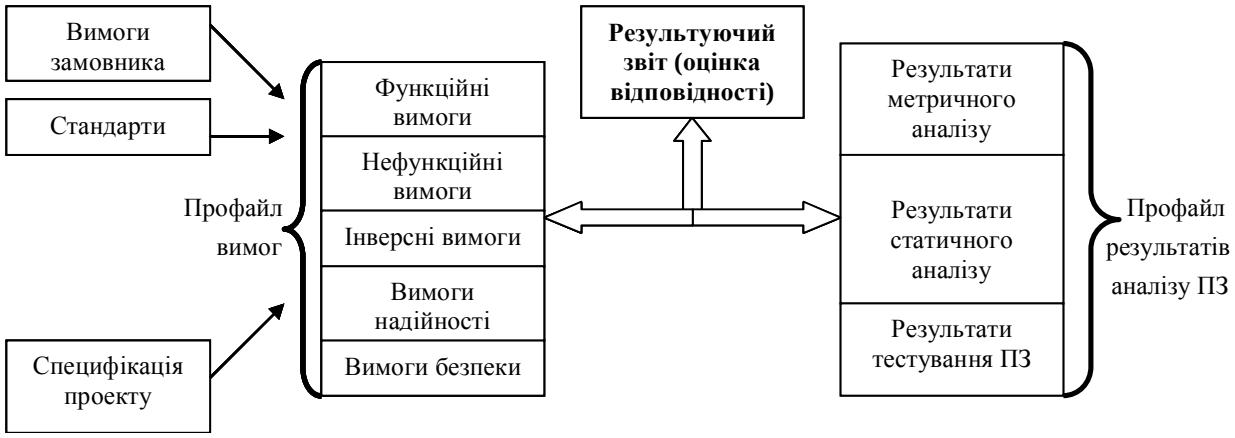

Рис. 1. Узагальнена модель методології Safety Case

Основні частини моделі Safety Case:

- профайл вимог до програмного забезпечення з врахуванням стандартів щодо розроблення ПЗ,

стандартів предметної галузі та вимог замовника – для формування такого профайлу аналізуються функційні, нефункційні та інверсні вимоги до ПЗ систем критичного застосування; досліджуються вимоги щодо надійності та безпеки ПЗ і оцінюється повнота всіх видів вимог;

- профайл результатів аналізу ПЗ – для формування такого профайлу досліджуються та аналізуються результати метричного аналізу, програмний код та результати тестування ПЗ;
- оцінка відповідності одержаного ПЗ (профайл результатів) висунутим до нього вимогам (профайл вимог).

Одним з основних засобів аналізу та оцінювання якості ПЗ є метричний аналіз. Метрика визначається як міра ступеня володіння властивістю, яка має числове значення [6]. Взагалі, метрика ПЗ – це міра, яка дозволяє одержати числове значення деякої властивості ПЗ або його специфікацій. Сучасна програмна індустрія накопичила велику кількість метрик, які оцінюють окремі виробничі та експлуатаційні властивості ПЗ. Однак прагнення їх універсальності, неврахування типу та області застосування розроблюваного ПЗ, ігнорування етапів життєвого циклу ПЗ та необгрунтоване їх використання в процедурах прийняття виробничих рішень істотно підірвало довіру розробників та користувачів ПЗ до метрик. Ці обставини вимагають ретельного відбору метрик для певного типу та області застосування розроблюваного ПЗ, врахування їх обмежень на різних етапах життєвого циклу ПЗ, встановлення порядку їх сумісного використання, накопичення та інтеграції різнорідної метричної інформації для прийняття своєчасних виробничих рішень.

Незважаючи на численні дослідження програмних метрик, в цій галузі залишається багато невирішених питань [7, 8]. Одним з таких питань є відсутність єдиних стандартів на метрики (створено більше тисячі метрик), тому кожен постачальник "вимірювальної" системи пропонує власні способи оцінки якості ПЗ і відповідні метрики. Складною також є задача інтерпретації значень метрик – для більшості користувачів як метрики, так і їх значення не є інформативними. На етапі проектування ПЗ основна увага при виборі проекту приділяється вартості, тривалості розроблення, репутації фірми-проектувальника та технологіям розроблення ПЗ. За статистичними даними [9], лише 1,5 % софтверних організацій намагаються оцінити якість процесів і готового продукту кількісно за допомогою метрик, і лише 0,5 % софтверних організацій намагаються покращити роботу, керуючись кількісними критеріями якості ПЗ з метою випуску бездефектних продуктів. Технологія вимірювання якості ще не досягла зрілості, оскільки лише 0,5 % софтверних організацій знаходяться на оптимізованому, зрілому рівні Capability Maturity Model (CMM) [7, 8]; основними параметрами при виборі варіанту реалізації ПЗ є вартість та тривалість розроблення і репутація фірми-проектувальника, але рішення, прийняті на основі цих параметрів, не завжди гарантують належну якість ПЗ.

Отже, методи оцінки якості ПЗ, особливо на етапі проектування, на сьогодні є суб'єктивно залежними, для знаходження значень метрик використовуються експертні вагові коефіцієнти; порівняння значень метрик поточних проектів з попередніми (постає проблема, що робити, якщо проект принципово новий); немає загальноприйнятої номенклатури метрик; відсутні точні значення метрик, з якими можна було б порівняти поточні одержані значення; при виборі проекту враховується не стільки інформація про якість проекту та майбутнього ПЗ, скільки інформація про тривалість, вартість, технології програмування та репутацію фірми-розробника.

Приймаючи до уваги результати аналізу методів оцінки ПЗ, можна зробити висновок, що перспективним напрямком досліджень є розроблення інтелектуальних систем, які аналізуватимуть і опрацьовуватимуть результати метричного аналізу етапу проектування та надаватимуть оцінку проекту та прогноз про характеристики ПЗ, що розробляється.

1. Оцінювання результатів проектування та прогнозування характеристик якості програмного забезпечення на основі метричного аналізу

На сьогодні, ПЗ є визначальною складовою багатьох систем, серед яких системи критичного застосування та спеціалізовані системи різноманітного призначення. Для зазначених систем наявність помилок та низька якість ПЗ загрожує катастрофами, які призводять до людських жертв, екологічних катаклізмів, економічних втрат. Проблема виявлення та усунення помилок загострюється по мірі збільшення складності ПЗ. Розвиток сучасних технологій розроблення ПЗ вимагає динамічного розвитку засобів оцінки якості ПЗ, причому вже на етапі проектування (з точки зору економічної та часової доцільності).

Сучасна індустрія ПЗ характеризується високою конкуренцією. Для успішної роботи на цьому ринку софтверна компанія повинна розробляти, впроваджувати та супроводжувати ПЗ швидко, вкладаючись в термін та із задовільною якістю. Тому багато софтверних компаній вкладають кошти в модернізацію процесів розроблення ПЗ, пам'ятаючи про те, що таке вкладення коштів обов'язково окупається.

У роботах [6– 8] були визначені метрики етапу проектування ПЗ з точними та прогнозованими значеннями та описані алгоритми їх обчислення.

В якості базових для побудови інтелектуального методу оцінювання результатів проектування та прогнозування характеристик якості програмного забезпечення оберемо 9 метрик етапу проектування ПЗ з точними значеннями та 15 метрик етапу проектування ПЗ з прогнозованими значеннями (таблиця 1). Інші метрики, описані в [6], є похідними від обраних базових метрик.

Таблиця 1

Метрики етапу проектування ПЗ з точними та прогнозованими значеннями

№ п/п	Метрики етапу проектування з точними значеннями		Метрики етапу проектування з прогнозованими значеннями	
	Метрики складності	Метрики якості	Метрики складності	Метрики якості
1	Метрика Чепіна	Метрика зв'язності	Очікувана LOC-оцінка	Прогнозований загальний час розроблення ПЗ (в робочих днях)
2	Метрика Джилба (абсолютна модульна складність програми)	Метрика зчеплення	Метрика Холстеда (складність програми)	Час виконання робіт процесу проектування ПЗ (в робочих днях)
3	Метрика Мак-Клура (загальна складність програмної системи)	Метрика звертання до глобальних змінних (наближена імовірність посилання довільного модуля на довільну глобальну змінну)	Метрика Маккейба (цикломатична складність ПЗ в цілому)	Очікувана вартість розроблення ПЗ (в гривнях)
4	Метрика Кафура (інформаційна складність модуля)	Час модифікації моделей (в робочих днях)	Метрика Джилба (відносна логічна складність програми)	Прогнозована вартість перевірки якості ПЗ
5		Загальна кількість знайдених помилок при інспектуванні моделей та прототипів модулів	Прогнозована кількість операторів програми	Прогнозована продуктивність розроблення ПЗ (в хвилинах на один рядок коду)
6			Прогнозована оцінка складності інтерфейсів ПЗ	Прогнозовані витрати на реалізацію програмного коду (в гривнях)
7				Прогнозований функційний розмір
8				Прогнозована оцінка трудовитрат за моделлю Боема (в людиномісяцях)
9				Прогнозована оцінка тривалості проекту за моделлю Боема (в місяцях)

В результаті опрацювання метрик, наведених в таблиці 1, за допомогою штучної нейронної мережі для матимемо оцінку результатів проектування та прогноз характеристик складності та якості ПЗ, що розробляється на основі того чи іншого проекту.

2. Нейромережний метод оцінювання результатів проектування та прогнозування характеристик якості програмного забезпечення (НМОП)

Для оцінювання результатів проектування та прогнозування характеристик якості ПЗ на основі опрацювання метрик етапу проектування з точними та прогнозованими значеннями використаємо штучну нейронну мережу (ШНМ), яка здійснює апрокисмацію метрик ПЗ етапу проектування та надає оцінку складності та якості проекту та прогноз характеристик складності та якості розроблюваного ПЗ.

Вхідними даними для ШНМ є множина метрик етапу проектування з точним значенням $TMP = \{tmp_a \mid a = 1..9\}$ та множина метрик етапу проектування з прогнозованим значенням $PMP = \{pmp_b \mid b = 1..15\}$.

Результатом роботи ШНМ є оцінки складності та якості проекту на основі точних метрик етапу проектування та прогноз про складність та якість майбутнього програмного забезпечення на основі точних та прогнозованих метрик етапу проектування (рис. 2).

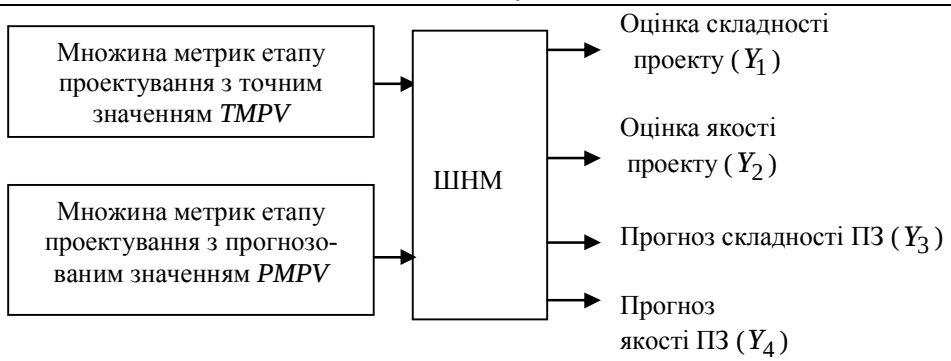

Рис. 2. Нейромережна складова НМОП

Вхідні дані для ШНМ подаються у вигляді множин $TMPV = \{tmpv_i \mid i = 1..9\}$, де $tmpv_i$ – кількісне значення i-ї метрики етапу проектування з точним значенням, якщо ця метрика задана для даного проекту, інакше нуль (0), та $PMPV = \{pmpv_j \mid j = 1..15\}$, де $pmpv_j$ – кількісне значення j-ї метрики етапу проектування з прогнозованим значенням, якщо ця метрика задана для даного проекту, інакше нуль (0).

Отже, ШНМ має 9 входів x' та 15 входів x, на входи x' подаються кількісні значення метрик етапу проектування з точним значенням, а на входи x подаються кількісні значення метрик етапу проектування з прогнозованим значенням. На вхід x'_i ($i = 1..9$) подається значення i-го елементу множини $TMPV$, на вхід x_j ($j = 1..15$) подається j-й елемент множини $PMPV$.

ШНМ опрацьовує набори вхідних векторів та видає 4 вихідних значення з діапазону [0, 1]: OQP – оцінка якості проекту, де 0 – проект неякісний, 1 – проект задовольняє вимоги замовника щодо якості; OSP – оцінка складності проекту, де 0 – проект складний для реалізації та передбачає високу вартість реалізації, 1 – проект простий для реалізації; $PQPZ$ - прогноз якості ПЗ, де 0 – майбутнє ПЗ буде неякісним, 1 – майбутнє ПЗ очікується високої якості; $PSPZ$ – прогноз складності ПЗ, де 0 – майбутнє ПЗ буде мати суттєву складність, 1 – майбутнє ПЗ очікується простим, причому дана характеристика передбачає не тільки простоту чи складність розроблюваного ПЗ з точки зору його обсягу, вартості та часу розробки, але й складність чи простоту супроводу, зручності використання (usability) та ефективність методів, обраних для вирішення задачі.

На основі аналізу 4-х одержаних результатів робиться висновок про якість і складність проекту та очікувану якість і складність розроблюваного за проектом програмного забезпечення.

3. Вибір архітектури нейромережі для реалізації нейромережної складової НМОП

У результаті аналізу відомих архітектур штучних нейронних мереж було обрано архітектуру для аналізу метрик етапу проектування ПЗ та прогнозу характеристик якості ПЗ.

Найбільш поширені архітектури ШНМ та їх особливості [10]:

1) багатошаровий персептрон – найпростіша і найбільш досліджена структура, якої достатньо для задачі аналізу метрик та прогнозування характеристик якості ПЗ;

2) радіальні базисні мережі (РБМ) вимагають великої кількості навчальних векторів (порядку 50000 навчальних векторів для мережі з 24 входами і 4 виходами). Для задачі аналізу метрик та прогнозування характеристик якості ПЗ важко зібрати таку кількість інформації по значеннях точних та прогнозованих метрик етапу проектування;

3) мережі GRNN (регресійні) – один з видів РБМ, використовуються для аналізу числових рядів. В задачі аналізу метрик та прогнозування характеристик якості ПЗ зв'язків, аналогічних числовому ряду, немає;

4) мережі PNN (імовірнісні) – один з видів РБМ, призначені для розв'язання імовірнісних задач, зокрема, задач класифікації – приналежність об'єкта до певного класу. Імовірнісна нейромережа має єдиний керуючий параметр навчання, значення якого обирає користувач, – - відхилення гаусової функції (параметр згладжування) PNN-мережі особливо корисні при пробних експериментах, наприклад, коли потрібно визначити, які вхідні параметри слід використовувати. В задачі аналізу метрик та прогнозування характеристик якості ПЗ немає потреби в класифікації;

5) карта Кохонена – призначена для кластеризації даних. Ця задача під час аналізу метрик та прогнозування характеристик якості ПЗ не вирішується;

6) мережі для класифікації вхідних векторів (LVQ) – використовуються для кластеризації і класифікації. Ці задачі під час аналізу метрик та прогнозування характеристик якості ПЗ не розглядаються;

7) мережі Елмана та Хопфілда – це мережі з динамічними оберненими зв'язками, орієнтовані на опрацювання динамічних моделей, що враховують передісторію процесів, які спостерігаються. Задача аналізу метрик та прогнозування характеристик якості ПЗ не містить обернених зв'язків і не вимагає врахування передісторії.

Для вирішення задачі аналізу метрик та прогнозування характеристик якості ПЗ оберемо

багатошаровий персептрон. При використанні нейромережі іншого типу для розв'язання цієї задачі її природа буде штучно спотворюватись, в результаті чого результати роботи ШНМ будуть невідповідними.

4. Визначення діапазонів значень вхідних векторів нейромережної складової НМОП

Перш ніж приступити до реалізації ШНМ, визначимо *діапазони значень її вхідних векторів*. Для цього накладемо певні обмеження на проекти та ПЗ, які будуть аналізуватись за допомогою НМОП. Для метрики Чепіна в якості максимальних для кожного модуля програми використовуватимемо наступні значення: 100 змінних для розрахунків і виведення, 100 модифікованих або створених в модулі змінних, 100 керуючих змінних та 100 невикористовуваних в модулі ("паразитних") змінних. При розробленні будь-якого модуля ПЗ не рекомендується вживати більше 100 змінних жодного вищеназваного типу [11, 12]. Тоді *метрика Чепіна* для одного модуля матиме максимальне значення $Q_m = 100 + 2 \cdot 100 + 3 \cdot 100 + 0{,}5 \cdot 100 = 650$ згідно формули, наведеної у [6]. Накладемо обмеження, що за допомогою НМОП аналізуватимуться проекти ПЗ, які мають не більше 50 модулів. Тоді максимальне значення метрики Чепіна для всього ПЗ становить $Q = 50 \cdot 650 = 32500$.

Для визначення максимального значення *метрики Джилба* вважатимемо, що за допомогою НМОП аналізуватимуться проекти ПЗ, які мають не більше 50 модулів. Якщо припустити, що кожен модуль ПЗ пов'язаний з іншими 49 модулями одним зв'язком, то максимальна кількість міжмодульних зв'язків (абсолютна модульна складність програми) становитиме 2450.

Враховуючи вищевикладені припущення щодо кількості модулів та кількості міжмодульних зв'язків, функція складності кожного з модулів для *метрики Мак-Клура* матиме значення:

$$M(P_1) = 0 \cdot 49 + 49 \cdot 49 = 2401$$
$$M(P_2) = 1 \cdot 49 + 48 \cdot 49 = 2401$$
$$M(P_3) = 2 \cdot 49 + 47 \cdot 49 = 2401$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$
$$M(P_{50}) = 49 \cdot 49 + 0 \cdot 49 = 2401$$

Тоді максимальне значення загальної складності програмної системи (метрика Мак-Клура) становить $M(P) = 50 * 2401 = 120050$.

Для одержання інформаційної складності модуля за *метрикою Кафура* вважатимемо, що в кожному модулі є не більше 50 процедур, які оновлюють структуру даних, не більше 50 процедур, які читають інформацію зі структури даних, та не більше 50 процедур, які читають і оновлюють структуру даних. Тоді за формулою, наведеною у [6], одержимо максимальне значення інформаційної складності одного модуля $I_m = 50 \cdot 50 + 50 \cdot 50 + 50 \cdot 50 + 50 \cdot (50 - 1) = 9950$. Оскільки введено обмеження, що аналізуються проекти ПЗ, які мають не більше 50 модулів, то максимальне значення інформаційної складності проекту становить $I = 50 * 9950 = 497500$.

Максимальне значення *метрики зв'язності* $CЗ = 10$ [6].

Максимальне значення *метрики зчеплення* $CЗ_ч = 9$ [6].

Для визначення максимального значення *метрики звертання до глобальних змінних* накладемо обмеження, що кожен модуль реально одержує доступ до глобальної змінної стільки разів, скільки може одержати такий доступ, тоді згідно формули, наведеної у [6], максимальне значення імовірності посилання довільного модуля на довільну глобальну змінну $Rup = 1$.

Для обчислення максимального часу модифікації моделей потрібно спочатку прорахувати максимальну прогнозовану тривалість проекту за моделлю Боема [6]. Накладемо припущення, що за допомогою НМОП можна аналізувати ПЗ, яке містить не більше 50000 рядків вихідного коду. Для визначення коефіцієнтів COCOMO [6] слід визначити, з яким типом проекту маємо справу (самостійним; жорстко вбудованим; не самостійним, але й не жорстко вбудованим). Щоб не накладати обмеження на тип проекту, для розрахунку максимальних значень оцінки трудовитрат та тривалості проекту оберемо максимальні значення коефіцієнтів COCOMO. Тоді *максимальна прогнозована оцінка трудовитрат за моделлю Боема* [6]: $Трудовитрати = 3{,}6 \cdot 50^{1{,}20} = 394$ (людиномісяців), а *максимальна прогнозована оцінка тривалості проекту за моделлю Боема* [6]: $Тривалість = 2{,}5 \cdot 394^{0{,}38} = 24$ (місяці) = 520 (робочих днів).

Якщо прийняти, що в цілому на розроблення програмного забезпечення на 50000 рядків коду – від постановки задачі до налагодження – потрібен максимальний час (520) днів, тоді на постановку задачі потрібно 52 дні (10 % часу), на проектування – 182 дні (35 % часу), на програмування – 182 дні (35 % часу) та на тестування, налагодження та перевірку якості – 104 дні (20 % часу) [13]. Якщо взяти до уваги, що під час проектування приблизно 25 % часу забирають побудова та модифікація моделей, то максимальне значення *часу модифікації моделей* – 46 днів.

Накладемо наступне обмеження, що максимальна кількість помилок моделей та прототипів одного модуля не повинна перевищувати 100, тоді максимальна *загальна кількість знайдених помилок при інспектуванні моделей та прототипів 50 модулів* складає 5000.

Максимальне значення *очікуваної LOC-оцінки* становить 50000 рядків коду, враховуючи

вищевикладені припущення.

Для розрахунку метрики Холстеда накладемо наступні обмеження: кількість унікальних операторів програми включно з іменами підпрограм (словник операторів) не перевищує 25000 (кожен другий рядок коду – це унікальний оператор), загальна кількість операндів програми не перевищує 50000 (в кожному рядку коду є один операнд), кількість унікальних операндів програми (словник операндів) не перевищує 400 (за вищенаведеними припущеннями). Тоді *максимальна складність програми за метрикою Холстеда* згідно з формулою, наведеною в [6], становить $HDiff = \frac{25000}{2} \cdot \frac{50000}{400} = 1562500$.

Максимальна *цикломатична складність Маккейба* за формулою, наведеною в [6]: $V(G) = 2450 - 50 + 2 = 2402$.

Вважатимемо, що кожен рядок програми – це логічний оператор або оператор циклу, тоді максимальна абсолютна логічна складність програми становить 50000, а максимальна *відносна логічна складність програми* в такому разі становить $CL = \frac{50000}{50000} = 1$.

Якщо припустити, що кожен рядок коду – це один оператор, тоді максимальна *прогнозована кількість операторів програми* становить 50000.

Для обчислення максимальної прогнозованої оцінки складності інтерфейсів ПЗ припустимо, що кожна змінна модуля передається по його інтерфейсу. Враховуючи припущення щодо кількості змінних в модулі, обчислимо *максимальну прогнозовану оцінку складності інтерфейсів ПЗ* за формулою, наведеною у [6]: $C = \frac{20000}{50 \cdot 400 \cdot K_{скл}} = \frac{1}{K_{скл}}$. Оскільки $K_{скл} = 1 + \sum_{i=1}^{n} K_i$, де K_i - приріст складності розроблюваної програми за *i*-ю характеристикою, *n* – кількість врахованих характеристик, то максимальна прогнозована оцінка складності інтерфейсів ПЗ становить 1.

Максимальним значенням *прогнозованого загального часу розроблення ПЗ* вважатимемо 520 робочих днів, а максимальним значенням *часу виконання робіт в процесі проектування ПЗ* вважатимемо 182 робочих дні.

За статистикою [13] *максимальною очікуваною вартістю розроблення ПЗ* в доларах США вважають кількість рядків вихідного коду, поділену навпіл. З врахуванням вищенакладених обмежень маємо 25000 доларів США, що приблизно становить 200000 грн. На тестування, налагодження та перевірку якості відводиться 20 % проектного часу і проектної вартості, тобто 40000 грн. Якщо взяти до уваги, що приблизно 50 % відведеного на даний етап часу та вартості забирає перевірка якості ПЗ [13], то максимальне значення *прогнозованої вартості перевірки якості ПЗ* – 20000 грн. На реалізацію програмного коду відводиться 35 % проектного часу і проектної вартості, тоді максимальне значення *прогнозованих витрат на реалізацію програмного коду* становить 70000 грн.

Продуктивність праці може вимірюватись кількістю часу, витраченого на одиницю продукції. Так, для розроблення ПЗ продуктивністю вважатимемо кількість часу, витрачену на 1 рядок коду. Тоді, враховуючи вищенаведені розрахунки тривалості проекту та обмеження щодо кількості рядків коду, максимальне значення *прогнозованої продуктивності розроблення ПЗ* при 8-годинному робочому дні становить $P = \frac{520 \cdot 8 \cdot 60}{50000} \approx 5$ хвилин.

Для розрахунку наближеного функційного розміру [6] вважатимемо, що кількість зовнішніх входів кожного модуля – 49, кількість зовнішніх виходів кожного модуля – 49, кількість зовнішніх запитів до кожного модуля – 49. Припустимо, що кожен модуль має максимум 50 внутрішніх логічних файлів та використовує 50 зовнішніх логічних файлів. Оскільки розраховуємо максимальне значення функційного розміру, то значення коефіцієнтів складності кожного фактору в програмному проекті обираємо максимальні [6]. Тоді максимальне значення наближеного функційного розміру згідно формули з [6] складає: $FP_{набл} = 49 \cdot 6 + 49 \cdot 7 + 49 \cdot 6 + 50 \cdot 15 + 50 \cdot 10 = 2181$. Для одержання максимального значення уточненого *функційного розміру* за формулою, наведеною у [6], оберемо ваги для 14 загальних характеристик проекту рівними 5 (максимальне значення), тоді: $FP = 2181 \cdot (0{,}65 + 0{,}01 \cdot (14 \cdot 5)) = 2945$.

5. Нейромережна складова НМОП

Архітектура нейромережної складової НМОП представлена на рис. 3.

ШНМ реалізовано у пакеті Matlab. Для створення шаблону ШНМ використовувалась функція *network*. Було визначено кількість вхідних векторів (*net.numInputs=4*), шарів (*net.numLayers=4*), елементів кожного вхідного вектора ШНМ: *net.inputs{1}.size=4; net.inputs{2}.size=5; net.inputs{3}.size=6; net.inputs{4}.size=9.*

Матриця зв'язності для зміщень: *net.biasConnect= [1; 1; 1; 1].*

Матриці зв'язності для входів, шарів, виходів та цілей задано наступним чином: *net.inputConnect = [1 1 1 1; 0 0 0 0; 0 0 0 0; 0 0 0 0]; net.layerConnect = [0 0 0 0; 1 0 0 0; 0 1 0 0; 0 0 1 0]; net.outputConnect = [0 0 0 1]; net.targetConnect = [0 0 0 1].*

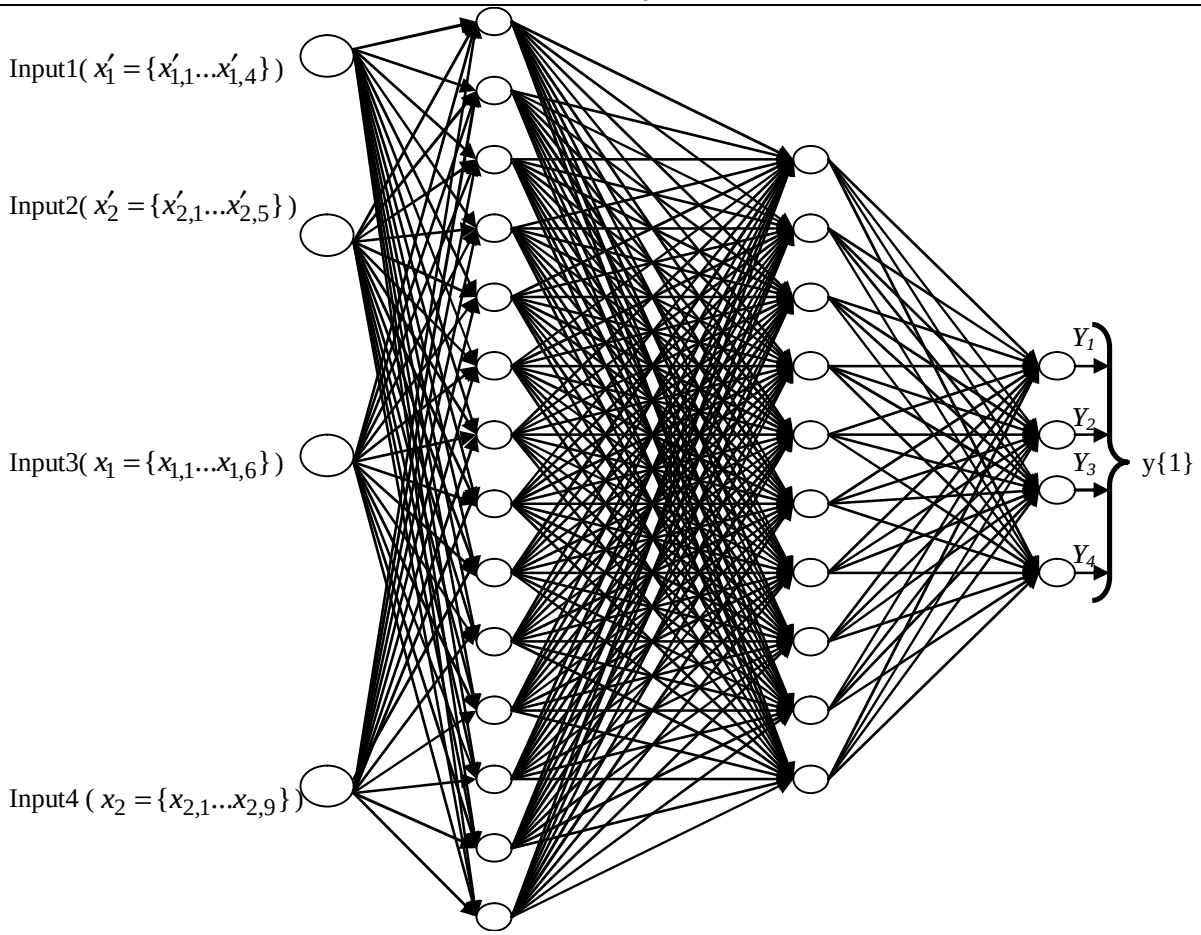

Рис. 3. Архітектура нейромережної складової НМОП

Діапазони значень вхідних векторів: *net.inputs{1}.range = [0 32500; 0 2450; 0 120050; 0 497500]; net.inputs{2}.range = [0 10; 0 9; 0 1; 0 46; 0 5000]; net.inputs{3}.range = [0 50000; 0 1562500; 0 2402; 0 1; 0 50000; 0 1]; net.inputs{4}.range = [0 520; 0 182; 0 200000; 0 20000; 0 5; 0 70000; 0 2945; 0 394; 0 24].*

Кількість нейронів у шарах ШНМ: *net.layers{1}.size=24; net.layers{2}.size=14; net.layers{3}.size=8; net.layers{4}.size=4.*

В якості функції ініціалізації для кожного шару ШНМ використано функцію Нгуена-Відроу (initnw) [10]. Для 1– 3-го шарів у якості активаційної функції обрано гіперболічний тангенс [10], для 4-го (вихідного) шару активаційною є порогова лінійна функція.

Оператор *gensim (net)* дає змогу одержати модель в пакеті Simulink (рис. 4-9).

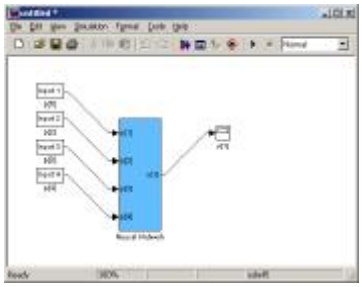

Рис. 4. Архітектура ШНМ в пакеті Simulink

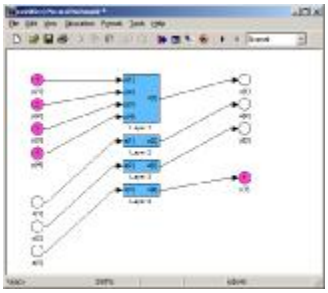

Рис. 5. Структурна схема шарів ШНМ в пакеті Simulink

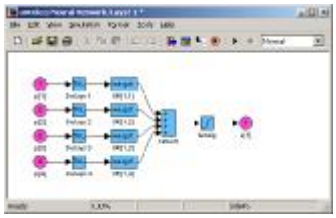

Рис. 6. Структурна схема першого шару ШНМ

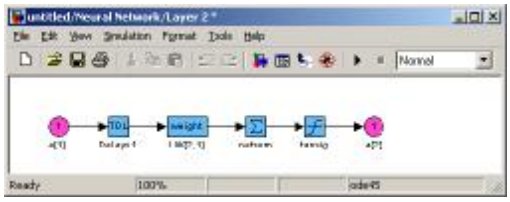

Рис. 7. Структурна схема другого шару ШНМ

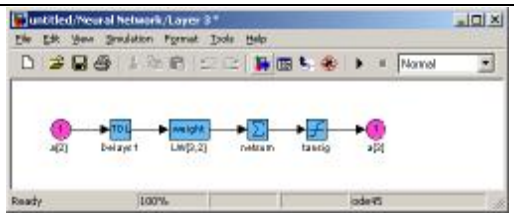

Рис. 8. Структурна схема третього шару ШНМ

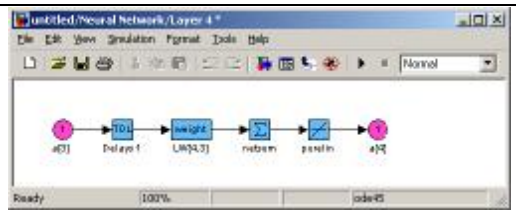

Рис. 9. Структурна схема четвертого шару ШНМ

6. Методика навчання ШНМ

Для навчання одержаної ШНМ послідовність навчальних векторів (навчальна вибірка) задано у вигляді:

c={ [32500; 0; 0; 0] [30875; 0; 0; 0] […]; – навчальні вектори для входу Input1 (x_1');

[0; 0; 0.15; 0; 0] [0; 0; 0.2; 0; 0] […]; – навчальні вектори для входу Input2 (x_2');

[3450; 0; 0; 0; 0; 0] [5900; 0; 0; 0; 0; 0] […]; – навчальні вектори для входу Input3 (x_1);

[0; 0; 0; 0; 1.7; 0; 0; 0; 0] [0; 0; 0; 0; 1.8; 0; 0; 0; 0] […]}; – навчальні вектори для входу Input4 (x_2).

Цільовий вектор визначено як m={ [0.05; 0.02; 0.01; 0.01] [0.1; 0.04; 0.02; 0.02] […]}.

Для вибору алгоритму навчання було проведено навчання ШНМ з навчальною вибіркою з 554 векторів за різними алгоритмами (рис. 10-13).

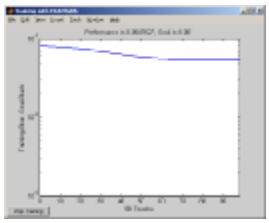

Рис. 10. Алгоритм градієнтного спуску з вибором параметра швидкості налагодження (GDA)

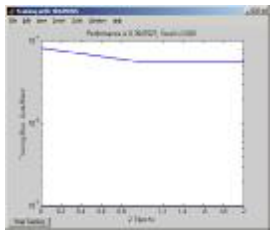

Рис. 11. Однокроковий алгоритм методу січної (OSS)

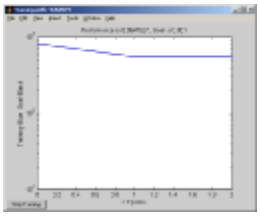

Рис. 12. Алгоритм Левенберга-Марквардта (LM)

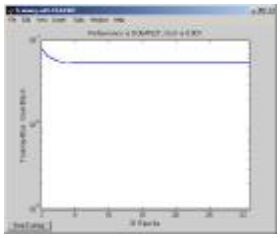

Рис. 13. Пороговий алгоритм оберненого поширення Rprop

Результати дослідження показали, що ШНМ навчається різними алгоритмами за різну кількість епох, але похибка навчання є приблизно однаковою для всіх алгоритмів навчання і у середньому складає 0.0547527.

7. Аналіз отриманих результатів

Для тестування ШНМ використовувалась тестова вибірка з 48 векторів. Процес навчання і тестування за різними алгоритмами навчання відображено на рис. 18– 21. На рисунках нижня крива відображає графік навчання, а верхня крива відображає графік тестування ШНМ.

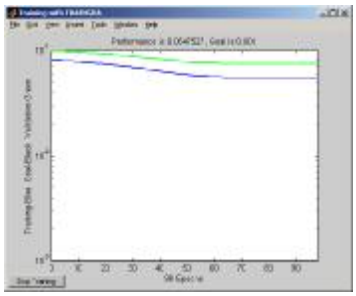

Рис. 18. Алгоритм градієнтного спуску з вибором параметра швидкості налагодження (GDA)

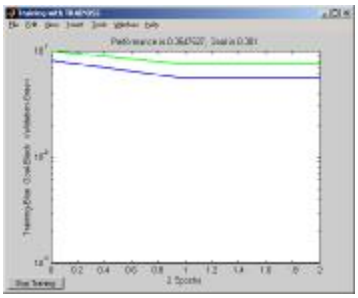

Рис. 19. Однокроковий алгоритм методу січної (OSS)

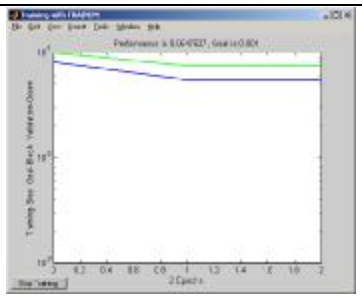

Рис. 20. Алгоритм Левенберга-Марквардта (LM)

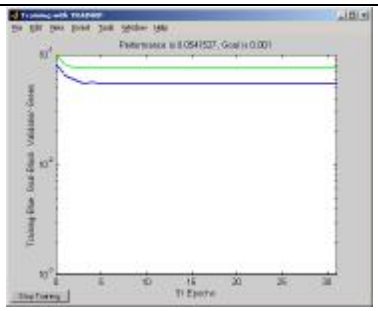

Рис. 21. Пороговий алгоритм оберненого поширення Rprop

Графіки навчання і тестування ШНМ показали, що мережа навчена з похибкою, яка обумовлюється недостатньою кількістю векторів навчальної вибірки.

Для розрахунку максимального об'єму навчальної вибірки використаємо формулу комбінаторики, яка дозволяє обчислити кількість сполучень без повторень: $C(m,n)=\frac{m!}{n!\cdot(m-n)!}=\frac{24!}{4!\cdot(24-4)!}=10626$, де m – кількість вхідних нейронів (входів) ШНМ ($m=24$); n – кількість вихідних нейронів (виходів) ШНМ ($n=4$). Отже, при наявності більше 10 тисяч навчальних векторів можна застосувати статистичні методи.

ШНМ використовується в умовах неповноти вхідної інформації. Для розрахунку необхідного об'єму навчальної вибірки для ШНМ, яку потрібно навчити з похибкою порядку 10^{-1} використаємо формулу [14]: $N>\frac{h\cdot g}{e_0}=\frac{24\cdot 24}{10^{-1}}=5760$, де g – кількість вхідних нейронів (входів) ШНМ ($g=24$); h – кількість нейронів прихованого шару ШНМ ($h=14+10=24$), e_0 – допустима похибка навчання ($e_0=10^{-1}$). Отже, 5760 векторів навчальної вибірки достатньо для того, щоб навчити ШНМ розпізнавати ці можливі ситуації з заданою точністю.

У якості прикладу використання запропонованого методу розглянемо результати дослідження метрик 4-х проектів, розроблених софтверною компанією "СТУ-Електронікс" м.Хмельницького (таблиця 2).

Таблиця 2

Опрацювання нейромережною складовою НМОП результатів метричного аналізу етапу проектування

№ п/п	Метрики складності з точними значеннями	Метрики якості з точними значеннями	Метрики складності з прогнозованими значеннями	Метрики якості з прогнозованими значеннями	Резуль-тат функці-ювання ШНМ
1	Метрика Чепіна – 1700 Метрика Джилба – 160	Метрика зв'язності – -10 Метрика зчеплення – 1	Очікувана LOC-оцінка – 3300 Метрика Холстеда – 73400 Метрика Джилба – 0,053	Прогнозований загальний час розроблення ПЗ – 27 Очікувана вартість розроблення ПЗ – 7800 Прогнозований функційний розмір – 120	Y_1=0,95 Y_2=1 Y_3=0,95 Y_4=0,96
2	Метрика Мак-Клура – 90000 Метрика Кафура – 376900 Метрика Чепіна – 24530	Метрика зв'язності – 3 Метрика зчеплення – 7 Метрика звертання до глобальних змінних – 0,71	Очікувана LOC-оцінка – 40135 Метрика Холстеда – 124928 Метрика Маккейба – 1903	Прогнозований загальний час розроблення ПЗ – 396 Очікувана вартість розроблення ПЗ – 152000 Прогнозований функційний розмір – 2220	Y_1=0,25 Y_2=0,3 Y_3=0,2 Y_4=0,26
3	Метрика Чепіна – 14538 Метрика Джилба – 1121	Метрика зв'язності – 7 Метрика зчеплення – 4	Очікувана LOC-оцінка – 25530 Метрика Холстеда – 781232 Метрика Джилба – 0,5	Прогнозований загальний час розроблення ПЗ – 218 Очікувана вартість розроблення ПЗ – 86100 Прогнозований функційний розмір – 1210 Прогнозована вартість перевірки якості ПЗ – 8800	Y_1=0,55 Y_2=0,6 Y_3=0,51 Y_4=0,57
4	Метрика Мак-Клура – 12000 Метрика Кафура – -64238 Метрика Чепіна – 3241	Метрика зв'язності – 9 Метрика зчеплення – 3 Метрика звертання до глобальних змінних – 0,089	Очікувана LOC-оцінка – 6240 Метрика Холстеда – 162251 Метрика Джилба – 0,12 Метрика Маккейба – 298	Прогнозований загальний час розроблення ПЗ – 69 Очікувана вартість розроблення ПЗ – 29300 Прогнозована оцінка трудовитрат за моделлю Боема – 60	Y_1=0,9 Y_2=0,92 Y_3=0,89 Y_4=0,86

Згідно з отриманими результатами: 1-й проект достатньо простий та має високу якість, майбутнє ПЗ очікується також простим та високої якості; 2-й проект достатньо складний та має низьку якість, майбутнє ПЗ очікується також достатньо складним та низької якості; 3-й проект має середню складність та якість, майбутнє ПЗ також очікується середньої складності та якості; 4-й проект є простим і має високу якість, майбутнє ПЗ також очікується достатньо простим і високоякісним.

Розглянемо вибір варіанту реалізації проекту на основі результатів ШНМ, вартості та часу розроблення (таблиця 3).

Таблиця 3

Характеристики варіантів реалізації проекту

№ варіанту	Характеристики Y_1, Y_2	Характеристики Y_3, Y_4	Вартість	Час розроблення
1	Y_1=0,51 Y_2=0,56	Y_3=0,60 Y_4=0,57	87000 грн	200 робочих днів
2	Y_1=0,32 Y_2=0,35	Y_3=0,38 Y_4=0,37	89000 грн	210 робочих днів

Характеристики варіантів реалізації з таблиці 3 свідчать, що обидва варіанти мають приблизно однакові вартість та час розроблення, але суттєво різні оцінки складності та якості проекту і прогнози щодо складності та якості розроблюваного ПЗ, тому на основі лише вартості та часу розроблення софтверна організація може прийняти хибний висновок щодо вибору варіанту реалізації проекту. Саме оцінки, надані ШНМ, допоможуть зробити вірний вибір і реалізовувати 1-й варіант, який має кращі показники складності та якості.

Висновки

З результатів аналізу методів метричної оцінки ПЗ випливає потреба і актуальність наукових досліджень в галузі оцінки та прогнозування якості ПЗ.

Основними параметрами при виборі варіанту реалізації ПЗ є вартість та тривалість розроблення і репутація фірми-проектувальника, але рішення, прийняті на основі цих параметрів, не завжди гарантують належну якість ПЗ. Прогнозовані оцінки характеристик якості розроблюваного ПЗ дають прогноз щодо якості реалізації конкретної версії проекту та дозволяють порівняти між собою різні версії проектів у ситуації, коли вартість і тривалість приблизно однакові. Запропонований підхід дає змогу прийняти мотивоване та обгрунтоване рішення щодо вибору проекту та його реалізації на основі не лише вартісних та часових характеристик, але й з врахуванням характеристик якості.

В ході дослідження автори виявили ряд невирішених задач: 1) проблема недостатності метричної інформації для нарощування обсягу навчальної та тестової вибірок; 2) необхідність розроблення метрик оцінки складності розроблюваного ПЗ з точки зору складності чи простоти його супроводу, зручності використання (usability) та ефективності методів, обраних для розв'язання задачі; 3) оптимізація архітектури ШНМ та вибір функції оцінки якості навчання ШНМ.

Література

1. Bishop P. A Methodology for Safety Case Development / P. Bishop. – 1998.

2. Kelly T. Arguing Safety – A systematic Approach to Managing Safety Cases / T. Kelly. – 1998.

3. Kelly T., Managing Complex Safety Cases. 11th Safety Critical System Symposium, 2003.

4. Górski J. Trust Case – a case for trustworthiness of IT infrastructures / J. Górski // Cyberspace Security and Defense: Research Issues, 2005.

5. Netkachova K. I. Safety Case Methodology: Architecting principles // Радіоелектронні і комп'ютерні системи – Харків: НАУ "ХАІ", 2010 – № 7, с.109-112

6. Поморова О.В., Говорущенко Т.О. Інтелектуальний метод оцінювання результатів проектування та прогнозування характеристик якості програмного забезпечення // Радіоелектронні і комп'ютерні системи – Харків: НАУ "ХАІ", 2010 – № 6, с.211-218

7. Поморова О.В., Говорущенко Т.О. Аналіз методів та засобів оцінки якості програмних систем // Радіоелектронні і комп'ютерні системи – Харків: НАУ "ХАІ", 2009 – № 6, с.148-158

8. Поморова О. В. Аналіз та опрацювання метрик якості програмного забезпечення на етапі проектування / О. В. Поморова, Т. О. Говорущенко, С. Я.Тарасек // Вісник Хмельницького національного університету – Хмельницький: ХНУ, 2010. – № 1, С. 54– 63

9. Липаев В. В. Выбор и оценивание характеристик качества программных средств: Методы и стандарты / Липаев В. В. – М.: Синтег, 2001 – 224 с.

10. Медведев В. С., Потемкин В.Г. Нейронные сети Matlab 6 / В. С. Медведев, В. Г. Потемкин; под общей ред. В. Г.Потемкина. – М.: Диалог-Мифи, 2002. – 496 с.

11. Кент Рейсдорф. Borland C++ Builder 3. Освой самостоятельно; [пер. с англ.]. – М.: ЗАО "Издательство БИНОМ", 1999. – 736 с.

12. Бобровский С. Delphi 5: учебный курс / Бобровский С. – СПб.: Изд-во "Питер", 2000. – 640 с.

13. Брауде Э. Технология разработки программного обеспечения / Брауде Э. – СПб.: Питер, 2004 –

655 с.

14. Уоссермен Ф. Нейрокомпьютерная техника: Теория и практика / Уоссермен Ф. – М: Мир, 1992. – 240 с.

Надійшла 20.3.2011 р.

УДК 621.391.24

М.В. ЗАХАРЧЕНКО, В.Й. КІЛЬДІШЕВ, С.В. ХОМИЧ, О.Г ПРИШЛЯК
Одеська Національна Академія зв'язку ім. О.С. Попова, м. Одеса

КОМПЕНСАЦІЯ НАДЛИШКОВОСТІ В БЛОКОВИХ КОРЕКТУЮЧИХ КОДАХ ЗА РАХУНОК ТАЙМЕРНИХ СИГНАЛІВ

Проведена оцінка ефективності компенсації затрат на перевірочні елементи в блокових коректуючих кодах за рахунок використання таймерних сигналів в бінарних каналах моделі Гільберта.

In the work was presented the assessment of the effectiveness the compensation costs oftesting elements in correcting block codes by using timer signals in a binary channel model Hilbert.

Ключові слова: Пропускна спроможність, спотворення, швидкість модуляції, таймерні сигнальні конструкції.

Постановка задачі

На даний момент часу актуальною є проблема старіння інформації, причиною якої є затримка повідомлень в процесі передачі інформації [1]. Однією із причин збільшення затримки повідомлень є збільшення числа додаткових коректуючих елементів в сигнальних конструкціях блокових кодів при підвищенні вимог до якості передачі [2].

Зрозуміло, що проблема компенсації надлишковості в каналах з базою $Б = \Delta F \cdot t_0 = 1$, в яких збільшення швидкості модуляції приводить до суттєвого збільшення між символьних завад, є актуальною. Методи зменшення часових витрат на передачу заданого об'єму інформації в бінарному каналі можливо розділити на два види:

- збільшення швидкості модуляції за рахунок зменшення відстані між суміжними значущими моментами модуляції (ЗММ), тобто використання наднайквістової швидкості передачі, аналізу цього методу якому присвячені роботи [3– 5];
- збільшення швидкості передачі інформації в одиницю часу за рахунок використання статистичних характеристик стану реальних каналів.

Аналіз публікацій стосовно даної проблеми. В роботі [4] для каналів з базою $Б = 1$ одержано аналітичну залежність між величиною дисперсії міжсимвольної завади s_z^2 характеристикою передавальної функції $H(w)$

$$s_z^2 = \frac{(1-p/\Omega)^2}{p\,l_0}\int_0^{\Omega}\sum_{p=-\infty}^{\infty}\frac{1}{1+\left[\frac{w-p2p}{2\,l_0}\right]^2}dw + \frac{1}{p\,l_0}\int_{\Omega}^{p}\sum_{p=-\infty}^{\infty}\frac{1}{1+\left[\frac{w-p2p}{2\,l_0}\right]^2}dw, \qquad (1)$$

де l_0 – середнє число перетинів нульового рівня за одиницю часу; Ω – смуга пропускання каналу;

w – циклічна частота коливання.

На рис. 1 [6] показані залежності характеристики завадостійкості h^2 при найквістовій швидкості і при різних коефіцієнтах перевищення її $m \in 1{,}1;\ 1{,}2;\ 1{,}4;\ 1{,}6$.

З іншого боку, збільшення швидкості передавання утворює можливість сформувати на тому самому часовому інтервалі кодового слова, більшу кількість реалізацій, тобто створити умови для збільшення пропускної здатності каналу:

$$C = m\left[1 + p_{\text{п}}\log_2 p_{\text{п}} + (1-p_{\text{п}})\log_2(1-p_{\text{п}})\right]. \qquad (2)$$

На рис. 2 наведено залежності $C = f(m)$ для значень h = 2; 3; 4; 5. Таким чином, показано, що зменшення тривалості елементарних сигналів за розрядно-цифрового кодування по відношенню до

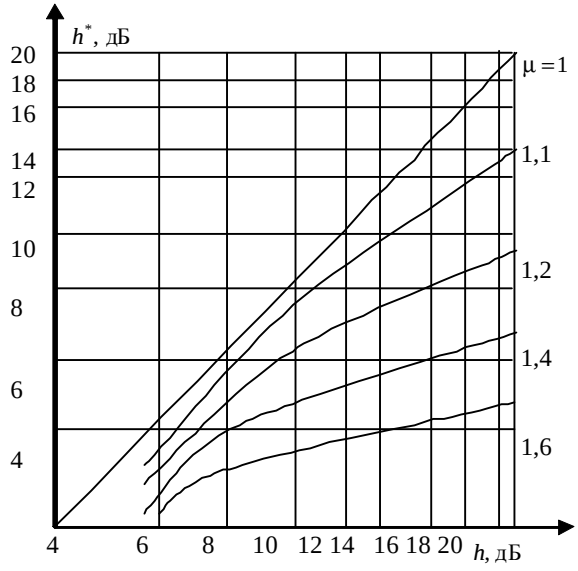

Рис. 1. Вплив флуктуаційного шуму на момент спрацьовування безінерційного реле