DOI: https://doi.org/10.25140/2411-5363-2026-1(43)-149-160
УДК 004.94

***Тарас Павлович Бивойно*[1], *Павло Георгійович Бивойно*[2]**

[1]старший викладач кафедри інформаційних технологій та програмної інженерії
Національний університет «Чернігівська політехніка» (Чернігів, Україна)
E-mail: tbivoyno@gmail.com. **ORCID:** http://orcid.org/0000-0001-6914-5441. **ResearcherID**: B-7478-2017

[2] кандидат технічних наук, доцент кафедри інформаційних та комп'ютерних систем
Національний університет «Чернігівська політехніка» (Чернігів, Україна)
E-mail: p.g.byvoino@gmail.com. **ORCID**: https://orcid.org/0000-0001-8145-8459. **ResearcherID:** R-7447-2016

АРХІТЕКТУРА ТА ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДІВ ОПТИМІЗАЦІЇ ПАРАМЕТРІВ ФУНКЦІЙ РЕГРЕСІЇ (В УМОВАХ ЛОКАЛЬНИХ ЕКСТРЕМУМІВ) З ВИКОРИСТАННЯМ ГІБРИДНОГО ПІДХОДУ

Представлена в статті інформація є науково-методичного характеру. Представлено підсистему, яка інтегрована у середовище імітаційного моделювання, що використовується для вивчення перехідної поведінки систем масового обслуговування (СМО) і поєднує модельні експерименти як для ідентифікації параметрів, так і для аналізу впливу конфігурацій СМО на перехідну поведінку. Архітектура підсистеми передбачає використання різних методів числової оптимізації, адаптованих до математичних моделей. Використовується гібридний підхід, що дозволяє подолати проблему локальних екстремумів, властиву деяким моделям. Доступ до всіх методів забезпечується через єдиний інтерфейс, який забезпечує уніфікацію взаємодії з підсистемою та масштабування системи для нових функцій регресії.

***Ключові слова**: оцінювання параметрів регресії; числова ідентифікація параметрів; імітаційне моделювання; системи масового обслуговування (СМО); гібридний алгоритм оптимізації; об'єктно-орієнтована архітектура програмного забезпечення.*

Рис.: 6. Бібл.: 8.

Актуальність теми дослідження. Підсистема, що розглядається, є складовою частиною комп'ютерного середовища імітаційного моделювання для визначення параметрів перехідної поведінки систем масового обслуговування (СМО). Перехідна поведінка СМО мало досліджена. Водночас це питання потребує вирішення, оскільки актуальним є завдання розрахунку потужності СМО для успішної обробки короткострокових потоків заявок, з використанням гібридного підходу, зокрема, кількості засобів ППО для успішної протидії повітряним атакам.

Постановка проблеми. У комп'ютерному середовищі імітаційного моделювання для визначення параметрів перехідної поведінки СМО є декілька підсистем, в яких потрібно знаходити функції регресії для представлення і подальшого використання результатів імітаційних експериментів. Ці функції регресії суттєво відрізняються між собою і знаходження їх оптимальних параметрів потребує застосування різних методів оптимізації. Створення багатофункціональної підсистеми пошуку параметрів функції регресії (БППФР) з використанням гібридного підходу дозволить удосконалити архітектуру комп'ютерного середовища, підвищити її надійність і спростити налаштування.

Аналіз останніх досліджень і публікацій. Питання визначення параметрів функцій регресії для представлення експериментальних даних достатньо добре вивчено. Ці питання розглядаються як у класичних монографіях[1], так і в навчальних посібниках для студентів [2]. Для обробки експериментальних даних існують сучасні, достатньо ефективні інструменти. Для вирішення простих задач можна використовувати табличний процесор MS Excel [3]. Більш складні завдання допомагає вирішувати Mathcad [4]. Але проблема полягає в тім, що такий інструмент потрібно вбудувати у спеціалізовану систему імітаційного моделювання. Тому доводиться створювати програмну реалізацію такого інструменту. Це завдання вимагає використання певних алгоритмів пошуку оптимальних рішень. Вирішенню цієї сторони проблеми як загалом, так окремих нестандартних випадків присвячено багато публікацій, зокрема [5-7] і результати цих досліджень можна використовувати для реалізації БППФР. Про спроби підійти до побудови подібної підсистеми є інформація в роботі [8], де розглядався проєкт, у складі якого було розроблено

дві окремих підсистеми. Одна використовувалася для визначення параметрів функції регресії для перехідного процесу для розміру черги СМО, інша для обробки результатів одно факторних багаторівневих експериментів.

Виділення недосліджених частин загальної проблеми. Проведений аналіз показує, що кожна окрема складова нашої проблеми добре досліджена та існують достатньо ефективні інструменти для обробки експериментальних даних. У роботі [8] цю проблему було вирішено частково. Проте нам потрібно об'єднання всіх складових проблеми в єдиному програмному комплексі, який можна буде вбудувати в середовище імітаційного моделювання для дослідження перехідної поведінки, а можливо і в інші системи.

Метою роботи є розробка архітектури і практична реалізація БППППФР на мові Java для середовища дослідження перехідної поведінки СМО засобами імітаційного моделювання.

Виклад основного матеріалу. Щоб визначитися з вимогами до програмних засобів, як мають бути реалізовані у підсистемі, розглянемо математичні моделі, які будуть використовуватися в середовищі імітаційного моделювання.

Моделі перехідної поведінки СМО. Перехідну поведінку СМО можна представити перехідним процесом для розміру черги. Для простої неперевантаженої СМО можна використовувати модель, яку було запропоновано в роботі [8]. Це експоненційний процес, що описується формулою (1).

$$q(t) = q_{\text{ст}}(1 - e^{\frac{-t}{tau}}), \quad (1)$$

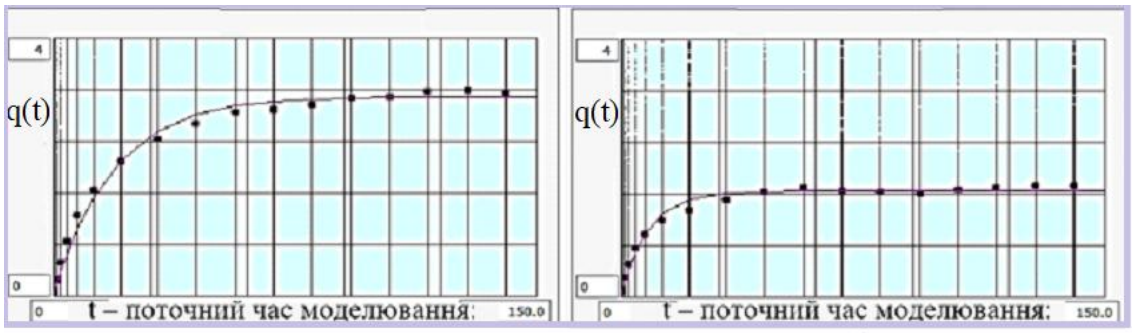
де q(t) – функція зміни середньої довжини черги в часі;

t – поточний час моделювання;

$q_{\text{ст}}$ – стале значення середньої довжини черги;

tau – стала часу (параметр, від якого залежить час переходу середньої довжини черги до сталого значення та тривалість перехідного процесу).

Модель характеризують два параметри – стале значення середньої довжини черги $q_{\text{ст}}$ та стала часу tau. Така модель добре описує перехідний процес у черзі не тільки для марківської СМО, але і для інших СМО. На рис. 1 можна побачити, як така модель узгоджується з експериментальними даними для СМО M/M/1/∞ та СМО M/G/1/∞ з коефіцієнтом завантаження 0,8.



а б

Рис. 1. Перехідна поведінка для СМО з коефіцієнтом завантаження 0,8: а – СМО M/M/1/∞; б – СМО M/G/1/∞

Джерело: розроблено авторами.

Якщо система перевантажена, то сталий режим для середньої довжини черги наступити не може. Черга буде безперервно зростати, але швидкість зростання з часом прийде до сталого значення. У такому разі перехідним процесом можна вважати процес зміни швидкості зростання черги від нульового до сталого значення. Можна припустити, що швидкість зміни розміру такої черги також буде змінюватися експоненційно відповідно

до формули 1, поки не досягне сталого значення. Експерименти підтвердили це припущення. У цьому випадку математичну модель перехідного процесу для середньої довжини черги можна представити формулою (2), яку отримано шляхом інтегрування правої частини формули 1.

$$q(t) = v_{\text{ст}} * (t - tau * e^{\frac{-t}{tau}}), \quad (2)$$

де q(t) – функція зміни середньої довжини черги в часі;

t – поточний час моделювання;

$v_{\text{ст}}$ – стале значення швидкості зміни середньої довжини черги;

tau – стала часу (параметр, від якого залежить час переходу швидкості зміни середньої довжини черги до сталого значення).

Тут також маємо два параметри – $v_{\text{ст}}$ і tau.

На рис. 2, *а* показано представлення експериментальних даних зміни черги в перевантаженій СМО M/M/1/∞ за допомогою експоненціальної моделі.

Таку модель можна також використовувати і для оцінки перехідних процесів на виході неперевантажених СМО. Для цього на виході системи можна додати чергу, яка буде приймати опрацьовані заявки. Розмір такої черги буде постійно зростати, але в цьому процесі зростання також можна виділити перехідний процес для швидкості зростання.

Виходячи з того, що нас цікавить не вигляд перехідного процесу, а тільки його параметри, то математичну модель перехідного процесу у перевантаженій СМО можна представити у вигляді двох прямих ліній, рис. 2, *б*.

Відповідна залежність описується формулою (3).

$$q(t) = \begin{cases} 0, & \text{якщо } t \le tau \\ v_{\text{ст}} * (t - tau), & \text{якщо } t > tau \end{cases}, \quad (3)$$

де q(t) – функція зміни середньої довжини черги в часі;

t – поточний час моделювання;

$v_{\text{ст}}$ – стале значення швидкості зміни середньої довжини черги;

tau – час, впродовж якого можна вважати, що довжина черги змінюється не суттєво.

Тут також маємо два параметри – $v_{\text{ст}}$ і tau. Останній може використовуватися для оцінки тривалості перехідного процесу.

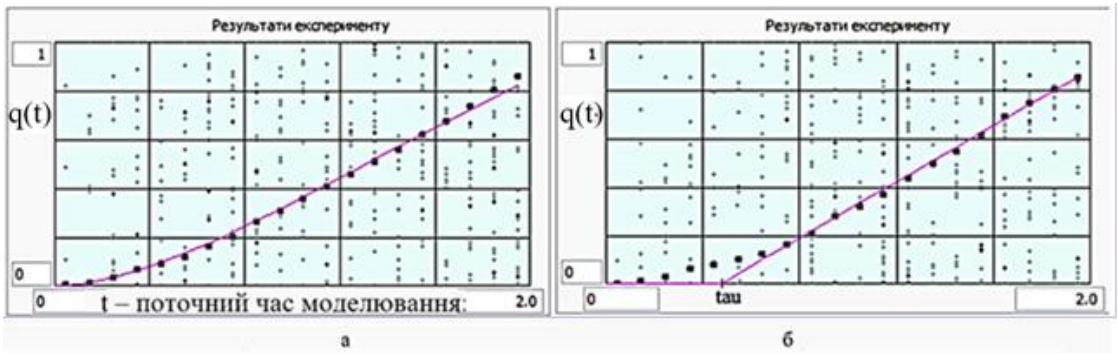


Рис.2. Характер середнього розміру черги в перевантаженій СМО (дрібними точками показані значення розміру черг в окремих реалізаціях процесу) а – експоненційна модель; б – лінеаризована модель

Джерело: розроблено авторами.

Якщо заявки, до появи в черзі, проходять попередньо декілька фаз обслуговування. то процес накопичення заявок в таких чергах затримується на початковій стадії. Для таких черг процес зміни середнього значення черги може виглядати так, як показано на

рисунку 3. Тобто спочатку середня довжина черги майже не змінюється, потім починається процес накопичення заявок і в решті решт настає сталий режим.

Для таких черг можна застосувати математичну модель, яка буде комбінацією лінеаризованої та експоненційної моделей, формула (4).

$$q(t) = \begin{cases} 0, & \text{якщо } t \le dely \\ q_{\text{ст}} * \left(1 - e^{\frac{-(t-delay)}{tau}}\right), & \text{якщо } t > dely \end{cases}, \quad (4)$$

де q(t) – функція зміни середньої довжини черги в часі;

t – поточний час моделювання;

v – стале значення швидкості зміни середньої довжини черги;

delay – час, коли можна вважати, що довжина черги змінюється не суттєво.

tau – стала часу (параметр, від якого залежить час переходу середньої довжини черги до сталого значення,

У цьому випадку тривалість перехідного процесу буде визначатися параметрами tau та delay.

Рис. 3. Процес зміни середньої довжини черги на останніх фазах обробки в багатофазній СМО

Джерело: розроблено авторами.

Моделі залежності параметрів СМО від коефіцієнта завантаження. Система імітаційного моделювання, для якої розробляється БППППФР, дозволяє проводити серії багаторівневих одно факторних експериментів для визначення впливу коефіцієнта завантаження на параметри системи. Такими параметрами є середня довжина черги у сталому режимі та стала часу перехідного процесу, від якої залежить його тривалість.

На рис. 4 показані результати експериментів по визначенню цих залежностей для простої марківської СМО M/M/1/∞.

Для середньої довжини черги у сталому режимі СМО M/M/1/∞ відома теоретична залежність, яка описується формулою (5), у якій коефіцієнт k = 1.

$$\bar{q} = k\frac{\omega^2}{1-\omega}, \quad (5)$$

де $\bar{q}$ – середня довжина черги;

ω – коефіцієнт завантаження системи.

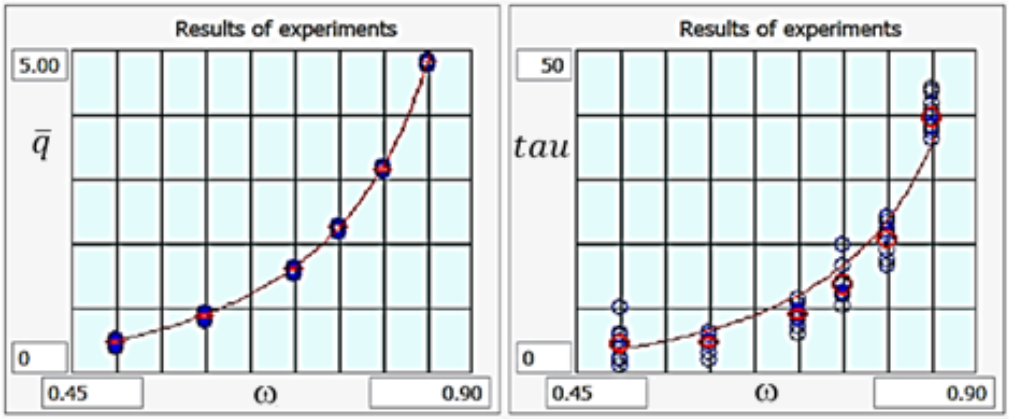
Що стосується залежності сталої часу перехідного процесу від коефіцієнта завантаження, то для неї теоретичних результатів не існує. Функція регресії у вигляді формули (5), яка була знайдена для даних, що представлені на рис. 4, *б*, виявилася неадекватною. А для пошуку адекватної функції регресії треба мати можливість експериментувати з різними формулами. Зокрема розглядались формули (6), (7), (8).

$$tau = k_1\frac{\omega^2}{1-\omega}e^{\left(\frac{k_2}{1-\omega^2}\right)}, \quad (6)$$

$$tau = k_1 \frac{\omega^2}{1-\omega} - e^{-k_2(\omega - k_3)^2}, \quad (7)$$

$$tau = k_1 \frac{\omega^{k_2}}{1-\sqrt{\omega}} + k_3, \quad (8)$$

де tau – стала часу перехідного процесу;
ω – коефіцієнт завантаження системи;
k_1, k_2, k_3 – коефіцієнти, значення яких потрібно знайти.



а б

Рис. 4. Залежність параметрів СМО від коефіцієнта завантаження: а – середня довжина черги; б – стала часу перехідного процесу

Джерело: розроблено авторами.

Отримання параметрів функції регресії для перехідного процесу. Завдання пошуку функції регресії для даних, що отримані в результаті експериментів, полягає у знаходженні таких параметрів вибраної математичної моделі, при яких лінія регресії буде проходити якомога ближче до експериментальних точок. Тобто задача полягає у пошуку оптимальних параметрів математичної моделі для наявних експериментальних даних. Для вирішення цього завдання потрібно визначитися з цільовою функцією та алгоритмом пошуку оптимуму.

Вибір цільової функції. Цільова функція є критерієм оцінки близькості лінії регресії до експериментальних точок. Значення цільової функції залежить від значень параметрів і має збільшуватися, якщо лінія регресії віддаляється від експериментальних точок, і зменшуватися при її наближенні. Найкращими значеннями параметрів будуть такі, при яких цільова функція має мінімальне значення.

Найбільш популярною цільовою функцією є така, що представляється у вигляді суми квадратів відхилень експериментальних точок від відповідних значень лінії регресії, вираз (9). Популярність цієї функції пояснюється тим, що вона є диференційованою і дозволяє в деяких випадках знайти аналітичне рішення шляхом вирішення системи рівнянь, де часткові похідні за параметрами функції регресії прирівнюються до нуля.

$$\sum_{i=1}^{n} (q_i^{\text{регр}} - q_i^{\text{експ}})^2 \rightarrow min \quad (9)$$

$$\sum_{i=1}^{n} |q_i^{\text{регр}} - q_i^{\text{експ}}| \rightarrow min \quad (10)$$

де $q_i^{\text{експ}}$ – ордината експериментальної точки,
$q_i^{\text{регр}}$ – ордината відповідної точки на лінії регресії.

Якщо ж для пошуку екстремуму будуть застосовуватися чисельні методи пошуку, то вигляд цільової функції не накладає суттєвих обмежень на реалізацію алгоритму пошуку оптимальних значень коефіцієнтів функції регресії. Тому в цьому випадку можна

застосувати і вираз 10, де мінімізується сума абсолютних відхилень експериментальних даних від лінії регресії. У БППППФР ми реалізуємо можливість вибору з наведених варіантів цільової функції у тих випадках, коли це можливо.

Алгоритми пошуку. Усі розглянуті математичні моделі для перехідних процесів та більшість моделей для залежності параметрів перехідного процесу від коефіцієнта завантаження представлені формулами, які не дозволяють отримати оптимальні значення коефіцієнтів функції регресії методом найменших квадратів, тому що після взяття похідних за невідомими коефіцієнтами отримуємо систему трансцендентних рівнянь, яку неможливо вирішити аналітично. З іншого боку, для всіх розглянутих моделей можна знайти коефіцієнти функції регресії чисельними методами. Розглянемо деякі з них.

Алгоритм координатного спуску. Такий алгоритм достатньо просто реалізується. Перед початком пошуку слід задати точку у просторі невідомих коефіцієнтів, з якої почнеться пошук. Часто точка береться навмання в допустимих межах для значень фактору. Далі алгоритм починає шукати мінімум цільової функції, шляхом зміни лише однієї координати. Після визначення мінімуму по цій координаті починається пошук мінімуму по наступній координаті і т. ін. Після того, як значення цільової функції перестануть покращуватися при переході від однієї координати до іншої, пошук закінчується. У випадку моделі, що представлена формулою 2.3 пошук будемо проводити спочатку по сталому рівню перехідного процесу ($q_{ст}$), потім по його тривалості (tau), потім знов по сталому рівню і т. д. Таким чином, реалізація пошук по декількох координатах потребує реалізації пошуку по одній координаті.

Алгоритм пошуку в напрямку градієнта. Так само як і в попередньому випадку, алгоритм потребує налаштування базової точки в просторі пошуку, з якої почнеться пошук. Далі складається матриця планування повного або дробового факторного експерименту на двох рівнях. Наприклад, для моделі перехідного процесу за формулою (4) матриця планування експериментів буде мати вигляд, який представлено в табл. 1.

Таблиця 1 – Матриця планування експериментів для отримання координат вектора градієнта для моделі перехідного процесу за формулою 4

Номер експерименту	k1 ($q_{ст}$)	k2 (tau)	k3 (dely)
1	+	+	-
2	+	-	+
3	-	+	+
4	-	-	-

Джерело: Розроблено авторами

Відповідно до цього плану проводиться серія експериментів. Отриману в результаті проведених експериментів інформацію використовують для обчислення координат вектора градієнта, який визначає напрям найбільшого збільшення (зменшення) цільової функції.

У напрямку, заданому цим вектором, проводять одно факторний експеримент, визначаючи оптимальне значення функції відгуку на лінії градієнта.

Точка, отримана таким чином, береться за базову точку, і для неї знову планується серія експериментів на двох рівнях.

Якщо значення функції відгуку в усіх точках плану виявилися гіршими, ніж в базовій точці, то це означає що, оптимум досягнутий. В іншому випадку проводиться нова серія експериментів по визначенню напрямку найбільшої зміни функції відгуку.

Алгоритм пошуку екстремуму по одній координаті. Розглянуті алгоритми переводять пошук екстремуму в багатовимірному просторі у пошук екстремуму по одній координаті. Тож тепер треба визначитися з алгоритмом реалізації такого пошуку.

Найбільш відомими методами пошуку екстремуму по одній координаті є методи дихотомії, чисел Фібоначчі, золотого перетину. Найпростіший з них – це метод дихотомії, але він, у порівнянні з двома іншими потребує більшої кількості тестів для прийняття рішення. Основна перевага методів Фібоначчі та золотого перетину полягає в тому, що на кожному кроці використовуються дані попереднього кроку, що скорочує число експериментів. Методи золотого перетину практично не відрізняються по ефективності, але у випадку дискретизації процесу пошуку, числа Фібоначчі використовувати зручніше. Тому будемо використовувати метод чисел Фібоначчі.

Для вирішення поставленого завдання пошуку оптимального значення цільової функції, перш за все, задається точність, з якою потрібно визначити невідомий параметр. Цю величину називатимемо мінімальним кроком пошуку. Потім визначаються границі області пошуку, всередині якої знаходиться екстремум. Далі починається сам пошук.

Перший етап алгоритму пошуку границь – це визначення напрямку руху. Обчислюється значення цільової функції в стартовій точці і на відстані кроку від неї. Якщо значення цільової функції збільшується, то крок робиться в протилежному напрямку. Якщо і тут значення цільової функції збільшується, то пошук закінчено. В іншому випадку стає відомим напрямок пошуку, який задається знаком кроку зміни параметра.

Другий етап полягає у визначенні другої границі області пошуку. Відстань від початкової точки до точки, де проводиться експеримент, поступово збільшується з кожним новим кроком пропорційно числам Фібоначчі, а ліву границю переносять у точку, де був попередній експеримент. Етап закінчується, коли цільова функція починає зростати. Важливо, що інтервали між точками у визначеній області, де проводилися експерименти, відповідають послідовності чисел Фібоначчі.

Останній етап реалізує пошук точки екстремуму всередині визначеної області. Область пошуку поступово звужується, але її ширина і відстані між точками, де проводяться експерименти, у будь-якому випадку дорівнюють числу Фібоначчі. Поки відстань між точками, в яких проводяться експерименти, більше одиниці, процес повторюється. В іншому випадку в області пошуку обирається точка з мінімальним значенням цільової функції, яка і буде точкою оптимуму.

Метод найменших квадратів. Цей метод можна застосувати для моделей, які представляють собою лінійну суперпозицію функцій фактора з ваговими коефіцієнтами, які є невідомими коефіцієнтами функції регресії, формула 11.

$$f = a1*\varphi1(x)+a2*\varphi2(x)+...+aN*\varphi N(x), \quad (11)$$

де $\varphi1(x), \varphi2(x), ..., \varphi N(x)$ – вибрані довільні функції від значень фактора, $a1, a2, ..., aN$ – невідомі коефіцієнти що визначають вигляд лінії регресії.

Серед розглянутих моделей такою є тільки формула 5. В цьому випадку у програмі достатньо запрограмувати розрахунки цих функцій та застосувати стандартні програми вирішення систем лінійних рівнянь.

Програмна реалізація підсистеми. Розглянуті алгоритми пошуку було реалізовано мовою програмування Java. Для стандартизації їх використання було створено інтерфейс IFinderRegresParm, який визначає протокол спілкування з об’єктами, які забезпечують пошук оптимальних параметрів математичних моделей перехідного процесу, незалежно від конкретної реалізації алгоритму пошуку коефіцієнтів та їх кількості. Для реалізації цього інтерфейсу було розроблено ієрархію класів, яку представлено на рисунку 5.

Інтерфейс IFinderRegresParm передбачає реалізацію методів, які вирішують наступні завдання:

– передачу сукупності даних про результати експериментів через методи setFactorArray(Double[]) та setValueArray(Double[]). Для цих масивів передбачені також гетери;

– доступ до результатів надає метод getParmArray(), який повертає оптимальні значення параметрів у вигляді масиву;

– доступ до методу funcRegres(double, double[]), що дозволяє розрахувати значення функції перехідного процесу для заданого значення фактора і масиву параметрів функції регресії;

– виведення переліку параметрів регресії у вигляді тексту за допомогою методу parametersAsString().

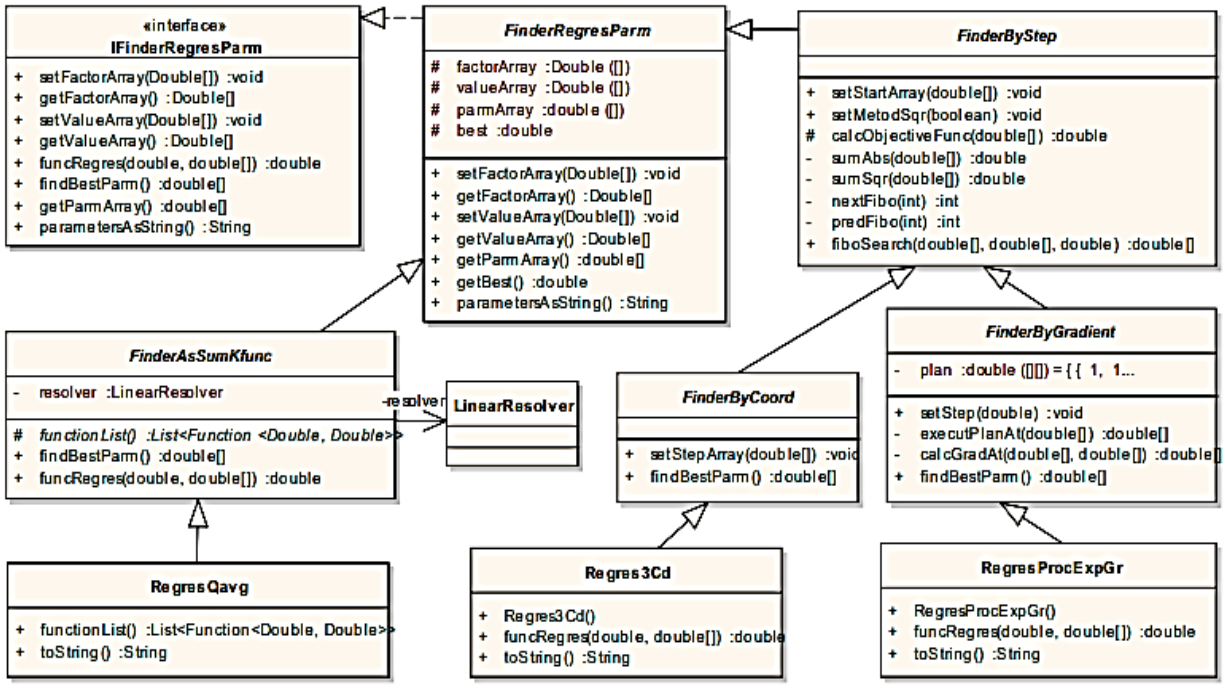


Рис. 5. Діаграма класів підсистеми БППППФР

Джерело: розроблено авторами.

Абстрактний клас FinderRegresParm містить поля для збереження вихідних експериментальних даних, результатів їх обробки та реалізації методів доступу до цих полів. Поле factorArray містить масив значень фактора, поле valueArray містить значення функції відгуку. Поле parmArray містить масив параметрів функції регресії, а поле best – оптимальне значення функції регресії. У класі також реалізовано метод parametersAsString(), в якому використовується метод getParmArray().

Абстрактний клас FinderAsSumKfunc реалізує загальні питання знаходження оптимальних параметрів функції регресії, якщо вона має вигляд формули 11. Абстрактний метод functionList<Function <Double,Double>> повертає список функцій *φ1(x), φ2(x), …, φN(x)*, які мають бути представлені у вигляді лямбда функцій і має бути реалізованим у класі спадкоємці. Клас також реалізує метод funcRegres(), в якому розраховуються значення функції регресії за формулою 11 з використанням функцій, що повертає метод functionList(). Метод findBestParm() формує і вирішує систему лінійних рівнянь для пошуку оптимальних коефіцієнтів функції регресії. Для розв’язку системи рівнянь використовується клас LinearResolver.

Клас RegresQavg успадковує клас FinderAsSumKfunc і являє собою конкретну реалізацію інтерфейсу IFinderRegresParm. Він дозволяє знайти оптимальний параметр функції регресії для залежності середньої довжини черги від коефіцієнта завантаження, яка визначається формулою 5. Цю формулу реалізовано у методі functionList(). Метод toString() повертає текстове представлення цієї формули.

Абстрактний клас FinderByStep реалізує загальні питання знаходження оптимальних параметрів функції регресії за допомогою чисельних методів пошуку, які не залежать від алгоритму пошуку.

Метод setStartArray(double[]) задає масив з координати стартової точки пошуку. Розмір цього масиву визначає кількість параметрів функції 11.

Метод setMetodSqr(boolean) дозволяє вибрати варіант цільової функції. Якщо параметр методу дорівнює true, то цільовою функцією буде сума квадратів, інакше сума модулів відхилень експериментальних точок від відповідних значень лінії регресії. Відповідні розрахунки реалізовано в приватних методах sumAbs(double[]) та sumSqr(double[]).

Сама цільова функція обчислюється за допомогою метода calcObjectiveFunc(double[]), куди передаються поточні значення параметрів функції регресії.

Метод fiboSearch(double[], double[], double) використовується для пошуку екстремуму цільової функції в одновимірному просторі, згідно з алгоритмом, який було описано вище. Перший параметр методу задає координати стартової точки. Другий параметр задає координати вектора градієнта, який визначає напрямок пошуку. Третій параметр задає початковий крок пошуку. Метод повертає масив з координатами точки оптимуму в заданому напрямку. Маніпуляції з числами Фібоначчі, які використовуються в алгоритмі, забезпечують приватні методи nextFibo(int) та predFibo(int).

Абстрактний клас FinderByGradient є спадкоємцем класу FinderByStep і реалізує градієнтний метод пошуку оптимальних параметрів функції регресії.

Метод setStep (double) визначає розмір мінімального кроку пошуку в напрямку градієнта, який зберігається у полі step.

Метод findBestParm() є реалізацією відповідного методу інтерфейсу IFinderRegres(). В методі реалізовано алгоритм пошуку екстремуму градієнтним методом, відповідно до алгоритму, описаного вище. Для пошуку в напрямку градієнта використовується метод fiboSearch() суперкласу FinderByStep.

Метод executePlan(double[]) розраховує значення цільової функції в точках плану проведення багатофакторного експерименту на двох рівнях. План проведення експерименту зберігається в полі plan.

Метод findGradient(double[]) розраховує координати вектора градієнта для заданої точки.

Клас RegresProcExpGR – це один із низки класів, що успадковують клас FinderByGradient і являють собою конкретні реалізації інтерфейсу IFinderRegresParm. Клас забезпечує пошук оптимальних параметрів функції регресії для перехідного процесу, яка визначається формулою 1. Цю формулу реалізовано у методі funcRegres(). У конструкторі класу налаштовуються усталені значення координат початкової точки пошуку та значення розміру кроку пошуку. Метод toString() повертає текстове представлення цієї формули.

Абстрактний клас FinderByCoord є спадкоємцем класу FinderByStep і реалізує пошук оптимальних параметрів функції регресії методом координатного спуску. Метод setStepArray(double[]) визначає значення розмірів кроків пошуку за координатами, які зберігаються у масиві stepArray. Метод findBestParm() є реалізацією відповідного методу інтерфейсу IFinderRegres(). В методі реалізовано алгоритм пошуку екстремуму методом координатного спуску. Для пошуку по кожній координаті використовується метод fiboSearch() суперкласу FinderByStep. Координата, по якій проводиться пошук, визначається наявністю одиниці в одному з елементів масиву градієнта.

Клас Regres3Cd являє собою одну із конкретних реалізацій інтерфейсу IFinderRegresParm, яка отримана шляхом успадкування класу FinderByCoord. Він дозво-

ляє знайти оптимальні параметри функції регресії для залежності сталої часу перехідного процесу від коефіцієнта завантаження системи , яка визначається формулою 8. Цю формулу реалізовано у методі funcRegres(). У конструкторі класу налаштовуються усталені значення координат початкової точки пошуку та значень розміру кроків пошуку по координатам. Метод toString() повертає текстове представлення цієї формули.

Проблема локальних екстремумів. В процесі пошуку адекватної лінії регресії для залежності параметрів перехідного процесу від коефіцієнта завантаження з'ясувалося, що для деяких функцій регресії можна отримати різні значення оптимальних параметрів, якщо змінювати координати стартової точки. Зокрема це відбувалося з пошуком параметрів лінії регресії, яка визначалася формулою (8). Для вирішення цієї проблеми було створено клас Regres3CdGr, який напряму успадковував клас FinderByStep і використовував класи Regres3Cd і Regres3Gr, які відповідно успадковували класи FinderByCoord та FinderByGradient.

Відповідна діаграма класів наведена на рис. 6.

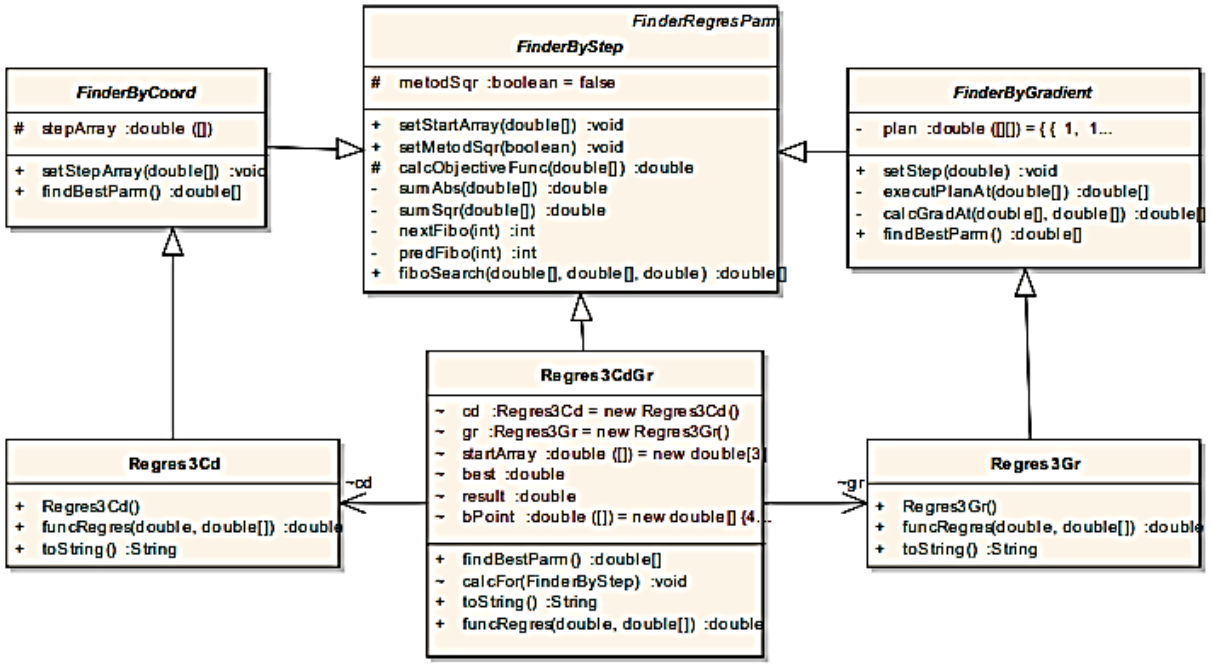


Рис. 6. Діаграма класів що реалізують пошук у випадку цільової функції з багатьма локальними екстремумами

Джерело: розроблено авторами.

Гібридний метод цього класу findBestParm поєднує методи координатного та градієнтного спуску з принципами генетичного алгоритму. У методі реалізовано прямий перебір координат стартових точок в діапазоні від 0.25 до 1.75 значень координат базової точки з кроком 0.1 значення координати. У кожній з точок пошук проводився як методом координатного спуску, так і градієнтним методом. У процесі перебору знаходилася точка, у якій значення цільової функції було найменшим. Після завершення циклу перебору знайдена точка ставала базовою і процес повторювався, поки новий цикл перебору давав кращий результат. Експерименти показали, що вибір початкової точки не впливав на результати пошуку параметрів функції регресії за допомогою цього класу. Попри велику кількість звертань до методу findBestParm (близько 7000 за один цикл перебору) пошук закінчувався достатньо швидко, у межах секунди.

Тестування класів. У класах FinderAsSumKfunk, FinderByCoord та FinderByGradient є методи main(), які дозволяють протестувати роботу об'єктів класів. У цих методах створюється об'єкт анонімного класу, який реалізує деяку функцію регресії. Далі генерується масив факторів і за допомогою створеного об'єкта генерується масив значень функції відгуку. Після цього задається деяка стартова точка й запускається процес пошуку. Результати тестування виводяться на консоль.

Висновки. У роботі запропоновано архітектуру багатофункціональної підсистеми оцінювання параметрів функції регресії, яку реалізовано мовою програмування Java. Ця підсистема призначена для інтеграції в систему дослідження перехідної поведінки СМО засобами імітаційного моделювання.

Зокрема, проаналізовано різновиди математичних моделей для функцій регресії, які можуть використовуватися в такій системі, а також розглянуто методи та алгоритми пошуку оптимальних параметрів цих функцій.

Архітектура підсистеми побудована як багаторівнева ієрархія класів із застосуванням єдиного інтерфейсу. Кінцевими (конкретними) класами цієї ієрархії є класи, що визначають специфічну модель функції регресії та відповідний метод пошуку.

Використання БППППФР у системі імітаційного моделювання дозволило спростити її архітектуру та підтвердило працездатність запропонованого підходу. Підсистема також може бути успішно використана в інших програмних платформах імітаційного моделювання.

Список використаних джерел

1. Cox, D. R., & Reid, N. (2000). *The theory of the design of experiments*. Chapman & Hall/CRC.

2. Літнарович, Р. М. (2011). *Побудова і дослідження математичної моделі за джерелами експериментальних даних методами регресійного аналізу : навчальний посібник*. МЕГУ.

3. Горват, А. А., Молнар, О. О., & Міньковіч, В. В. (2019). *Методи обробки експериментальних даних з використанням MS Excel: Навчальний посібник*. Видавництво УжНУ «Говерла».

4. Maxfield, B. (2006). *Engineering with Mathcad: Using Mathcad to create and organize your engineering calculations*. Butterworth-Heinemann.

5. Кісельова, О. М., & Шевельова, А. Є. (2008). *Чисельні методи оптимізації: Навчальний посібник*. Вид-во ДНУ.

6. Касіцький, О. В. (2012). Метод покоординатного спуску з евристикою середньозваженого напрямку. *Наукові вісті НТУУ «КПІ»*, (1(81)), 69–75.

7. Sebastian, R. (2016). *An overview of gradient descent optimization algorithms*. arXiv. https://arxiv.org/abs/1609.04747.

8. Бивойно, П. Г., Бивойно, Т. П., & Сисоєва, М. М. (2011). Дослідження перехідних процесів для середньої довжини черги в системах масового обслуговування. *Вісник Чернігівського державного технологічного університету. Серія «Технічні науки»,* (1(47)), 235–241.

References

1. Sokh, D. R., & Reid, N. (2000). *The Theory of the Design of Experiments*. Chapman & Hall / CRC.

2. Litnarovych, R. M. (2011). *Pobudova i doslidzhennia matematychnoi modeli za dzherelamy eksperymentalnykh danykh metodamy rehresiinoho analizu [Construction and study of a mathematical model based on experimental data sources using regression analysis methods: A study guide]*. MEHU.

3. Horvat, A. A., Molnar, O. O., & Minkovych, V. V. (2019). Metody obrobky eksperymentalnykh danykh z vykorystanniam MS Excel [Methods of experimental data processing using MS Excel: A study guide]. Uzhhorod: Vydavnytstvo UzhNU "Hoverla". (in Ukrainian).

4. Maxfield, B. (2006). *Engineering with Mathcad: Using Mathcad to create and organize your engineering calculations*. Butterworth-Heinemann.

5. Kiselova, O. M., & Shevelova, A. Ie. (2008). *Chyselni metody optymizatsii [Numerical optimization methods: A study guide]*. DNU Publishing House.

6. Kasitskyi, O. V. (2012). Metod pokoordynatnoho spusku z evrystykoiu serednozvazhenoho napriamku [Coordinate-by-coordinate descent method with weighted average direction heuristics]. *Naukovi visti NTUU «KPI» – Scientific News of NTUU "KPI"*, (1(81)), 69-75. (in Ukrainian).

7. Ruder, S. (2016). An overview of gradient descent optimization algorithms. arixv:1609.04747. Accessed 05 Feb 2020.

8. Byvoino, P. H., Byvoino, T. P., & Sysoieva, M. M. (2011). Doslidzhennia perekhidnykh protsesiv dlia serednoi dovzhyny cherhy v systemakh masovoho obsluhovuvannia [Study of transient behavior for average queue length in queueing systems]. *Visnyk Chernihivskoho derzhavnoho tekhnolohichnoho universytetu. Seriia «Tekhnichni nauky» – Bulletin of Chernihiv State Technological University. Series "Technical Sciences"*, (1(47)), 235–241.

Дата першого надходження статті до видання: 19.12.2025
Дата прийняття статті до друку після рецензування: 08.01.2026

UDC 004.94

***Taras Byvoino*[1], *Pavel Byvoino*[2]**

[1]Senior Lecturer, Department of Information Technologies and Software Engineering
Chernihiv Polytechnic National University (Chernihiv, Ukraine)
E-mail: tbivoyno@gmail.com. **ORCID:** http://orcid.org/0000-0001-6914-5441. **ResearcherID**: B-7478-2017

[2]PhD in Technical Sciences, Associate Professor, Department of Information and Computer Systems
Chernihiv Polytechnic National University (Chernihiv, Ukraine)
E-mail: p.g.byvoino@gmail.com. **ORCID**: https://orcid.org/0000-0001-8145-8459. **ResearcherID:** R-7447-2016

ARCHITECTURE AND SOFTWARE IMPLEMENTATION OF METHODS FOR OPTIMIZATION OF PARAMETERS OF REGRESSION FUNCTIONS (IN CONDITIONS OF LOCAL EXTREMES) USING A HYBRID APPROACH

Presented in the article information is of scientific and methodical character. This paper presents the development of a subsystem designed for regression parameter estimation. The subsystem is an integral part of a simulation environment used to study transient behavior in Queueing Systems (QS).

While the problem of determining regression function parameters for experimental data representation is well-established and supported by numerous tools, integrating such functionality into a specialized, multifunctional simulation system remains a challenge. Such integration is essential for systems that combine simulation experiments for both parameter identification and analyzing the impact of QS configurations on transient behavior .

The authors consider mathematical models with several unknown parameters to represent simulation results; however, in most cases, the least squares method cannot be employed due to the specific nature of the regression functions.

The proposed architecture system incorporates various numerical optimization methods tailored to the selected mathematical model. In particular, a novel specific method was implemented, combining coordinate and gradient descent techniques with genetic algorithm principles. This hybrid approach enables overcoming the problem of local extrema inherent in some models.

Access to all methods for estimating optimal regression parameters is provided through a unified interface that establishes a consistent protocol for interaction and ensures the extensibility of the system for new regression functions. To support this architecture, an appropriate class hierarchy was developed. The concrete classes within this hierarchy define specific regression models and their respective search algorithms.

The integration of the developed subsystem into the simulation environment for studying transient processes in QS has simplified the system architecture and confirmed its practical viability. Given its design, the subsystem can be effectively utilized in other simulation platforms.

***Keywords**: regression parameter estimation; numerical parameter identification; simulation modeling; queueing systems (QS); hybrid optimization algorithm; object-oriented software architecture.*

Fig.: 6. References: 8.

Бивойно Т. П., Бивойно П. Г. Архітектура та програмна реалізація методів оптимізації параметрів функцій регресії (в умовах локальних екстремумів) з використанням гібридного підходу. *Технічні науки та технології,* 2026. № 1(43). С. 149-160. DOI: https://doi.org/10.25140/2411-5363-2026-1(43)-149-160.