https://doi.org/10.31891/2307-5732-2025-359-92
УДК 004.41

ГРИГОР'ЄВА ТЕТЯНА
Міжнародний гуманітарний університет
https://orcid.org/0000-0001-8345-0020
e-mail: tig15090808@gmail.com
РУСУ ОЛЕКСАНДР
Міжнародний гуманітарний університет
https://orcid.org/0000-0002-7315-2537
e-mail: shurusu@ukr.net
ГОРБАЧОВ ВІКТОР
Міжнародний гуманітарний університет
https://orcid.org/0000-0002-6668-6864
e-mail: physmgu@gmail.com
КРОХМАЛЬ МИКОЛА
Міжнародний гуманітарний університет
e-mail: physonat@gmail.com

БАГАТОВИМІРНИЙ ПІДХІД В ЗАДАЧАХ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

У статті досліджується ефективність та доцільність застосування багатовимірного тензорного підходу у задачах розробки програмного забезпечення. Проаналізовано потенціал тензорних методів для моделювання складних багатовимірних залежностей, які є характерними для сучасних IT-систем. Обґрунтовано, що на відміну від традиційних векторно-матричних моделей, які часто призводять до втрати інформації про багатовимірну взаємодію, використання тензорних представлень складних взаємодій дозволяє в процесі розробки зберігати структуру й контекст не тільки елементів самого коду, але і між множиною параметрів програмного продукту, такими як зміни, авторство, метрики, тестові сценарії та час роботи програми. Розглянуто прикладні аспекти використання тензорних декомпозицій (CP, Tucker, Tensor Train) у задачах забезпечення якості, виявлення дефектів, оптимізації тестування та аналізу програмних залежностей. Також окреслено перспективи інтеграції тензорних моделей у середовища CI/CD, системи автоматизованого тестування, інструменти моніторингу технічного боргу та аналізу архітектури програмного забезпечення. Сформульовано рекомендації щодо впровадження багатовимірних технологій у практику програмної інженерії як основи для побудови самонавчальних, адаптивних та масштабованих систем підтримки розробки.

Ключові слова: *багатовимірний підхід, тензорний метод, оптимізація коду, аналіз залежностей, тестування, інтелектуальні системи розробки.*

HRYHORIEVA TETYANA, RUSU OLEKSANDR, GORBACHEV VICTOR, KROKHMAL MAKSYM
International Humanitarian University

MULTIDIMENSIONAL APPROACH TO SOFTWARE DEVELOPMENT PROBLEMS

The effectiveness and feasibility of using a multidimensional tensor approach in software development tasks are investigated. Within the framework of the multidimensional approach, the potential of tensor methods for modeling complex multidimensional dependencies characteristic of software of modern IT systems is analyzed. Traditional models based on the use of scalar, vector and matrix representations allow to effectively solve a wide range of tasks, such as algorithm analysis, testing, profiling and verification of program code. However, when it comes to multidimensional dependencies, such as simultaneous interaction between users, program modules, execution time and configuration parameters, matrix models become limited. They are unable to represent the complex structure of the system, which leads to the loss of information about the multidimensional interaction of product elements, reduces the accuracy of forecasting and complicates the detection of hidden patterns. It is substantiated that the use of tensor representations of complex interactions allows the development process to preserve the structure and context not only of the elements of the code itself, but also the relations between a set of parameters of the software product, such as changes, authorship, metrics, test scenarios and program execution time. The applied aspects of using various tensor decompositions (CP, Tucker, Tensor Train) in the tasks of quality assurance, defect detection, testing optimization and software dependency analysis are considered. The prospects for integrating tensor models into CI/CD environments, automated testing systems, technical debt monitoring tools and software architecture analysis are also outlined. Recommendations are formulated for the implementation of multidimensional technologies in software engineering practice as a basis for building self-learning, adaptive and scalable development support systems.

Keywords: *multidimensional approach, tensor method, code optimization, dependency analysis, testing, intelligent development systems.*

Стаття надійшла до редакції / Received 22.10.2025
Прийнята до друку / Accepted 15.11.2025

Постановка проблеми

В сучасних умовах стрімкого зростання обсягів даних та ускладнення структури інформаційних систем, розробка програмного забезпечення потребує ефективних інструментів для моделювання, оптимізації та аналізу багатовимірних залежностей. Одним із перспективних напрямків є використання тензорного підходу, який дозволяє працювати з багатовимірними структурами даних. Традиційні моделі, засновані на використанні скалярних, векторних і матричних уявлень, дозволяють ефективно вирішувати широкий спектр задач, таких як аналіз алгоритмів, тестування, профілювання та верифікація програмного коду. Однак, коли йдеться про багатовимірні залежності, такі як одночасна взаємодія між користувачами, модулями програм, часом виконання та параметрами конфігурації, матричні моделі стають обмеженими. Вони не здатні репрезентувати комплексну структуру системи, що знижує точність прогнозування та ускладнює виявлення прихованих закономірностей.

Сучасна розробка програмного забезпечення характеризується високим ступенем складності, глибокою інтеграцією численних підсистем, паралельною участю багатьох розробників та безперервними змінами у функціональності. За таких умов зростає потреба в інструментах, що дозволяють ефективно аналізувати, прогнозувати й оптимізувати якість програмного забезпечення, а також забезпечувати адаптивне управління процесами розробки та підтримки. Одночасно, у життєвому циклі програмного продукту накопичуються великі обсяги гетерогенних даних: зміни коду, коміти, тестові звіти, логи виконання, метрики складності, дефекти, пов'язані зі змінами, участь розробників, релізні графіки тощо. Ці дані природним чином мають багатовимірну структуру, але здебільшого опрацьовуються в обмежених векторних або табличних формах, внаслідок чого втрачається важлива контекстуальна інформація.

Аналіз досліджень та публікацій

Традиційні методи аналізу та оптимізації у сфері програмної інженерії (наприклад, лінійна регресія, класифікація, дерева рішень або метрики якості коду) демонструють ефективність лише за умови наявності одновимірних або двовимірних вхідних даних і не враховують складні міжмодальні залежності, що виникають між типом змін, історією комітів, компонентною архітектурою, активністю команди та показниками дефектності [1,2]. Ці обмеження ускладнюють автоматичне виявлення дефектів, планування тестування, прогнозування ризиків або прийняття управлінських рішень на основі метрик.

Останніми роками тензорні методи набувають все більшої популярності в задачах розробки програмного забезпечення, зокрема в контексті аналізу багатовимірних даних, оптимізації архітектури, тестування та машинного навчання. Їхня здатність моделювати складні багатовимірні залежності робить їх потужним інструментом у сучасній програмній інженерії [3,4]. Разом з тим залишається багато відкритих питань, що потребують подальших досліджень.

Застосування тензорних методів у задачах розробки програмного забезпечення перебуває на стику математичного моделювання, аналізу багатовимірних даних та програмної інженерії.

Формулювання цілей статті

Метою роботи є: дослідження ефективності та доцільності використання тензорних методів у задачах розробки програмного забезпечення. Зокрема, розглядається потенціал тензорного підходу для моделювання складних багатовимірних залежностей, характерних для сучасних IT-систем, а також аналізуються переваги тензорних представлень порівняно з традиційними векторно-матричними структурами в контексті забезпечення якості, прогнозування дефектів, управління зв'язками, оптимізації тестування й автоматизації супроводу програмного коду. У межах дослідження окреслюються перспективи впровадження тензорних технологій у сучасні середовища розробки, тестування і підтримки програмного забезпечення.

Виклад основного матеріалу

Моделювання складних багатовимірних залежностей. Тензорний підхід дозволяє моделювати дані у вигляді тензорів – багатовимірних масивів, де кожен вимір (мода) може відповідати певному аспекту системи: функціональному блоку, метриці ефективності, середовищу виконання, типу користувача тощо. Такий підхід надає змогу не лише краще структурувати дані, а й застосовувати потужні методи тензорної декомпозиції, що дозволяє зменшити розмірність без втрати важливої інформації, виявити латентні фактори, що впливають на продуктивність чи помилки, а також будувати точніші моделі поведінки програмних систем.

Відтак постає актуальна науково-практична задача з розробки математичних моделей та інструментів, які дозволяють одночасно працювати з багатовимірними залежностями, зберігаючи повноту інформації про взаємодію між численними параметрами системи. Тензорні методи, як узагальнення скалярних, векторних і матричних структур, надають саме таку можливість. Їх застосування дозволяє виконувати факторизацію або класифікацію для виявлення латентних закономірностей. Це відкриває нові можливості у завданнях виявлення дефектів, визначення технічного боргу, оптимізації регресійного тестування, керування навантаженням у командах розробників та автоматичного рекомендаційного аналізу.

Тензорний підхід має потужний потенціал для моделювання складних багатовимірних залежностей, які є типовими для сучасних IT-систем, оскільки він дозволяє зберігати та аналізувати взаємозв'язки між численними параметрами у спільному математичному просторі. У розробці програмного забезпечення дані дуже нечасто є лінійними або двовимірними – більшість інформації має глибоку структурну й контекстуальну прив'язку: зміни в коді залежать від часу, автора, типів завдань, впливають на результати тестування, взаємодіють із іншими компонентами та мають історію. Усі ці аспекти утворюють природно багатовимірну структуру. Наприклад, тензор $X \in R^m \times R^t \times R^a \times R^d$, де m – модулі коду, t – часові інтервали, a – автори, d – метрики (розмір, складність, частота змін), може одночасно відображати, як змінюється складність певного модуля протягом часу, хто ці зміни вносив, і з якими дефектами вони корелюють. У двовимірному аналізі таку залежність довелося б розглядати окремими частинами, втрачаючи їхню спільну взаємодію.

У задачах виявлення дефектів тензорна модель може виявити комбінації авторів, змін і тестових результатів, що в сукупності призводять до високої імовірності помилки, навіть якщо жоден з окремих факторів не є критичним. У тестуванні тензори дозволяють моделювати взаємозалежність тестових сценаріїв, конфігурацій середовища, версій програмного забезпечення і часу запуску.

Потенціал тензорного підходу особливо проявляється в таких ключових напрямках, як збереження латентних багатовимірних патернів; аналіз взаємодій між елементами у понад двох вимірах (наприклад, «код–людина–час»); виявлення аномалій, які неможливо виявити в ізольованих векторах або матрицях; інтеграція з глибоким навчанням і графовими представленнями для побудови гібридних моделей.

Таким чином, тензорні методи не лише здатні моделювати складні багатовимірні залежності, а й дозволяють зробити їх об'єктами прогнозування, оптимізації та автоматизованого контролю. Це робить їх цінним інструментом для IT-систем, де одночасно взаємодіє багато динамічних параметрів. Однак, впровадження тензорного підходу в програмну інженерію стикається з низкою викликів, зокрема: необхідністю коректного формування багатовимірних представлень, вибором оптимального типу тензорної декомпозиції (CP, Tucker, TT тощо) [5], обчислювальною складністю при великій розмірності, а також потребою в адаптації існуючих інструментів до специфіки життєвого циклу програмного забезпечення. Це визначає актуальність подальших досліджень у цьому напрямі, спрямованих на інтеграцію тензорних моделей у задачі автоматизованої підтримки якості, аналітики змісту, прогнозування ризиків та адаптивного управління процесами розробки.

Синтез тензорної алгебри, глибинного навчання та теорії графів у задачах розробки програмного забезпечення. Інтенсивний розвиток програмної інженерії супроводжується зростанням складності архітектур програмних систем, обсягів супровідної інформації та багаторівневої взаємодії між її елементами. У відповідь на ці виклики формуються інтегративні підходи, що поєднують інструменти з різних галузей обчислювальної науки. Одним із найбільш перспективних напрямів є синтез тензорної алгебри, глибинного навчання та теорії графів для побудови інтелектуальних систем аналізу, моделювання та оптимізації процесів розробки програмного забезпечення.

Тензорна алгебра забезпечує математичну основу для представлення багатовимірних даних, що є типовими у програмній інженерії. Векторні ембедінги модулів, логів, тестів, сценаріїв змін можуть бути організовані у багаторозмірні тензори, які зберігають структурні та функціональні залежності між об'єктами. Наприклад, тензорна модель типу $X \in R^m \times R^k \times R^a \times R^d$, де m – модулі коду, k – коміти, a – автори, d – метрики, дозволяє здійснювати комплексний аналіз еволюції коду.

Глибинне навчання, у свою чергу, забезпечує адаптивність та узагальнення в умовах великих і складних даних. Використання згорткових нейронних мереж (CNN) або трансформерів для аналізу ембедінгів коду вже продемонструвало ефективність у задачах автогенерації, рефакторингу, виявлення дефектів. Тензорні представлення можуть бути використані як вхід до таких моделей, а Tensor Train-декомпозиція – для стискання їхніх параметрів без втрати точності.

Теорія графів, зокрема графи викликів, залежностей, успадкування чи контролю потоку, забезпечує формалізацію логічної структури програмного забезпечення. Однак, класичні графові методи часто не враховують багатовимірну природу атрибутів вузлів і ребер. В цьому випадку тензори виступають як узагальнення графів у вигляді гіперграфів або багатовимірних взаємодій. Наприклад, Graph Neural Networks (GNN) [6], інтегровані з тензорним аналізом, дозволяють моделювати семантичні залежності між об'єктами, що мають часові та структурні аспекти.

Синтез зазначених підходів дозволяє будувати глибокі гібридні архітектури. Наприклад, системи, де графові репрезентації коду конвертуються у тензорні представлення, які далі обробляються трансформерами для виявлення аномалій чи прогнозування боргу. Це особливо ефективно в умовах великого розподіленого середовища розробки, де потрібне моделювання складної динаміки змін. Перспективними напрямками такого синтезу є такі задачі, як побудова багатовимірних репрезентацій версійного контролю з графовою структурою залежностей; мультиагентні моделі розробників із урахуванням їхніх ролей, вкладень і взаємодії у часово-просторовому тензорному просторі; автоматичне генерування тестів або сценаріїв виправлення на основі тензорних знань про архітектурні шаблони; стискання й оптимізація глибоких моделей для edge-середовищ розробки за допомогою тензорних факторизацій.

Таким чином, міждисциплінарний синтез тензорної алгебри, глибинного навчання та теорії графів формує потужну парадигму для побудови гнучких, структурно обізнаних, самонавчальних систем у розробці програмного забезпечення, які здатні адаптуватися до складності сучасних IT-продуктів і прискорити перехід до автономної інженерії.

Інтеграція тензорних моделей з інструментами CI/CD та DevOps. У сучасних програмно-інженерних практиках системи CI/CD (Continuous Integration/Continuous Delivery) та DevOps забезпечують безперервне розгортання, тестування й оновлення програмного забезпечення [7,8]. У таких динамічних середовищах виникає потреба не лише в автоматизації рутинних процесів, але й у глибокій багатовимірній аналітиці – виявленні патернів змін, оцінці ризиків, прогнозуванні дефектів і адаптивному плануванні. Інтеграція тензорних моделей є ефективним і перспективним напрямком для розширення інтелектуальної функціональності DevOps-інфраструктури.

Тензорні представлення дозволяють структурувати та аналізувати дані, що надходять із різних етапів CI/CD-конвеєра: зміни в коді (коміти), метадані про авторів, результати автоматизованих тестів, характеристики середовища виконання, лог-файли, метрики продуктивності, тривалість побудов, тощо. Наприклад, моделі типу $X \in R^v \times R^m \times R^a \times R^r$, де v – версії, m – модулі, a – автори, r – тестові результати, дозволяють проводити глобальний аналіз якості змін у часово-просторовому контексті.

Висновки

В статті показано, що тензорні моделі дозволяють цілісно охоплювати багатовимірні залежності, які виникають між модулями коду, метриками якості, результатами тестування, історією змін та активністю розробників. Аналіз прикладних сценаріїв підтвердив доцільність використання CP- і Tucker- декомпозицій, особливо у задачах виявлення дефектів, прогнозування технічного боргу, оптимізації тестових стратегій та

автоматизованого аудиту залежностей. Застосування тензорного підходу сприяє підвищенню точності прогнозів, зменшенню обчислювального навантаження та забезпеченню структурно обґрунтованої підтримки прийняття рішень у складних проектах програмної інженерії.

Разом з тим виявлено низку відкритих питань, що визначають напрями подальших наукових досліджень. По-перше, потребують подальшого розвитку алгоритми тензорного аналізу для їх адаптації до потокової обробки даних та онлайн-оновлення моделей у реальному часі. По-друге, перспективною є розробка домен-специфічних тензорних форматів для представлення вихідного коду, змін, логів і звітів про тестування. По-третє, актуальним завданням є формалізація критеріїв якості тензорних моделей у контексті програмної інженерії, зокрема стабільності факторизацій, чутливості до змін і інтерпретованості результатів.

Рекомендовано зосередити майбутні дослідження на інтеграції тензорних моделей з інструментами CI/CD та DevOps, побудові інтелектуальних панелей розробника на основі тензорної аналітики, а також створенні відкритих багатовимірних репозиторіїв коду для навчання моделей машинного навчання. Важливим є також міждисциплінарний підхід, який передбачає синтез тензорної алгебри, глибинного навчання, теорії графів і лінгвістичного аналізу коду. Такий напрям забезпечить формування нового покоління аналітичних інструментів для розробки програмного забезпечення, що відповідають вимогам масштабованості, гнучкості й інтелектуалізації в умовах динамічних IT-екосистем.

Література

1. Långström S. An Investigative Study of Testing Strategy and Test Case Creation in a Hardware-Software Co-design Environment Using Software Product Line Theory: degree project in computer science and engineering, second cycle / Stina Långström. – Stockholm, – 2021. – 88 p. URL: http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-305219.

2. Haripriya T. Statistical Learning Methods for Predictive Analytics in Engineering Management / T. Haripriya, D. P. Mankame, B. Patil, et al. // Nanotechnology Perceptions. – 2024. – V. 20. – No. S14. – P. 834–847. DOI: https://doi.org/10.62441/nano-ntp.vi.2855.

3. Wang D. High-dimensional low-rank tensor autoregressive time series modeling / D. Wang, Y. Zheng, G. Li // Journal of Econometrics. – 2024. – V. 238. – No. 1. – P. 105544. DOI: https://doi.org/10.1016/j.jeconom.2023.105544.

4. Xia D. Effective tensor sketching via sparsification / D. Xia, M. Yuan // IEEE Transactions on Information Theory. – 2021. – V. 67. – No. 2. – P. 1356–1369. DOI: https://doi.org/10.48550/arXiv.1710.11298.

5. Bhatt V. Tucker decomposition and applications / V. Bhatt, S. Kumar, S. Saini // International Conference on Technological Advancements in Materials Science and Manufacturing. – 2021. – V. 46. Part 20. – P. 10787–10792. DOI: https://doi.org/10.1016/j.matpr.2021.01.676.

6. Khemani B. A review of graph neural networks: concepts, architectures, techniques, challenges, datasets, applications, and future directions / B. Khemani, S. Patil, K. Kotecha, et al. // Journal of Big Data. – 2024. – V. 11. – No. 18. – P. 1–43. DOI: https://doi.org/10.1186/s40537-023-00876-4.

7. Rostami Mazrae P. On the usage, co-usage and migration of CI/CD tools: A qualitative analysis / P. Rostami Mazrae, T. Mens, M. Golzadeh, et al. // Empirical Software Engineering. – 2023. – V. 28. – No. 2. – P. 52–76. DOI: https://doi.org/10.1007/s10664-022-10285-5.

8. Faustino J. DevOps Benefits: A Systematic Literature Review / J. Faustino, D. Adriano, R. Amaro, et al. // Journal of Software: Practice and Experience. – 2022. – V. 52. – No. 9. – P. 1905–1926. DOI: https://doi.org/10.1002/spe.3096.

References

1. Långström S. An Investigative Study of Testing Strategy and Test Case Creation in a Hardware-Software Co-design Environment Using Software Product Line Theory: degree project in computer science and engineering, second cycle / Stina Långström. – Stockholm, – 2021. – 88 p. URL: http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-305219.

2. Haripriya T. Statistical Learning Methods for Predictive Analytics in Engineering Management / T. Haripriya, D. P. Mankame, B. Patil, et al. // Nanotechnology Perceptions. – 2024. – V. 20. – No. S14. – P. 834–847. DOI: https://doi.org/10.62441/nano-ntp.vi.2855.

3. Wang D. High-dimensional low-rank tensor autoregressive time series modeling / D. Wang, Y. Zheng, G. Li // Journal of Econometrics. – 2024. – V. 238. – No. 1. – P. 105544. DOI: https://doi.org/10.1016/j.jeconom.2023.105544.

4. Xia D. Effective tensor sketching via sparsification / D. Xia, M. Yuan // IEEE Transactions on Information Theory. – 2021. – V. 67. – No. 2. – P. 1356–1369. DOI: https://doi.org/10.48550/arXiv.1710.11298.

5. Bhatt V. Tucker decomposition and applications / V. Bhatt, S. Kumar, S. Saini // International Conference on Technological Advancements in Materials Science and Manufacturing. – 2021. – V. 46. Part 20. – P. 10787–10792. DOI: https://doi.org/10.1016/j.matpr.2021.01.676.

6. Khemani B. A review of graph neural networks: concepts, architectures, techniques, challenges, datasets, applications, and future directions / B. Khemani, S. Patil, K. Kotecha, et al. // Journal of Big Data. – 2024. – V. 11. – No. 18. – P. 1–43. DOI: https://doi.org/10.1186/s40537-023-00876-4.

7. Rostami Mazrae P. On the usage, co-usage and migration of CI/CD tools: A qualitative analysis / P. Rostami Mazrae, T. Mens, M. Golzadeh, et al. // Empirical Software Engineering. – 2023. – V. 28. – No. 2. – P. 52–76. DOI: https://doi.org/10.1007/s10664-022-10285-5.

8. Faustino J. DevOps Benefits: A Systematic Literature Review / J. Faustino, D. Adriano, R. Amaro, et al. // Journal of Software: Practice and Experience. – 2022. – V. 52. – No. 9. – P. 1905–1926. DOI: https://doi.org/10.1002/spe.3096.