Голові разової спеціалізованої вченої ради
Національного технічного університету
«Дніпровська політехніка»
д.т.н., професору Михайлу АЛЕКСЄЄВУ

ВІДГУК

офіційного опонента на кваліфікаційну наукову працю (дисертацію) **Стародубського Ігоря Петровича** на тему «Методи та засоби переносимості програм та програмних систем на різні обчислювальні платформи», поданої на здобуття наукового ступеня доктора філософії за спеціальністю 122 «Комп'ютерні науки»

1. Актуальність теми дисертаційного дослідження

Сучасний розвиток комп'ютерних наук і програмної інженерії відбувається в умовах швидкого розширення спектра апаратних архітектур, операційних систем, середовищ виконання та моделей розгортання програмного забезпечення. Поряд із традиційними процесорними платформами загального призначення дедалі ширше застосовуються графічні процесори, спеціалізовані прискорювачі, периферійні обчислювальні вузли, вбудовані системи та хмарні інфраструктури. Унаслідок цього програмні системи мають зберігати функціональну коректність, прийнятні експлуатаційні характеристики та стабільність роботи в суттєво відмінних апаратно-програмних середовищах.

Забезпечення переносимості програмного забезпечення за таких умов не може обмежуватися лише використанням стандартизованих мов програмування, крос-компіляції, віртуалізації або формуванням незмінних контейнерних образів. Зазначені засоби створюють необхідну основу для міжплатформного виконання програм, однак переважно не враховують динамічних змін навантаження, ресурсних обмежень цільової платформи, конкуренції за обчислювальні ресурси та відмінностей у поведінці програмної системи під час її експлуатації.

Особливої значущості набуває проблема автоматизованої адаптації середовища виконання без внесення змін до прикладного коду. У хмарних, мультихмарних і периферійних інфраструктурах параметри процесорних ресурсів, пам’яті, кількості реплік, політик розміщення та масштабування мають коригуватися відповідно до поточного стану системи. Водночас такі зміни повинні здійснюватися узгоджено, без порушення доступності сервісів і без виникнення нестабільних коливань конфігурації.

Перспективним напрямом розв’язання цієї проблеми є поєднання контейнерних технологій, засобів оркестрації та методів машинного навчання, здатних оцінювати поточний стан програмної системи, прогнозувати наслідки можливих керуючих впливів і забезпечувати проактивне налаштування середовища виконання. Додаткові можливості підвищення міжплатформної ефективності створює адаптивна компіляція, яка дає змогу враховувати особливості цільової мікроархітектури ще на етапі формування машинного коду.

Отже, тема дисертації Стародубського Ігоря Петровича, спрямована на розроблення методів і програмних засобів підвищення переносимості програмних систем шляхом комплексного застосування адаптивної контейнеризації, машинного навчання та оптимізації параметрів компіляції, є актуальною. Вона відповідає сучасним тенденціям розвитку комп’ютерних наук, хмарних технологій, інтелектуального керування програмними системами та автоматизації процесів розгортання й експлуатації програмного забезпечення.

2. Загальна характеристика структури та змісту дисертації

Дисертаційна робота Стародубського І.П. складається зі вступу, чотирьох розділів, загальних висновків, списку використаних джерел і додатків. Загальний обсяг роботи становить 218 сторінок. Дисертація містить 17 рисунків, 9 таблиць, список із 115 використаних джерел та 10 додатків.

Структура дисертації загалом відповідає логіці проведеного дослідження. Виклад матеріалу побудовано за послідовністю від аналізу наявних підходів і формування теоретичних положень до розроблення методів, їх програмної

реалізації, експериментальної перевірки та оцінювання можливостей комплексного застосування.

У вступі обґрунтовано актуальність теми, сформульовано мету та задачі дослідження, визначено його об’єкт, предмет і методологічну основу. Наведено положення наукової новизни, охарактеризовано практичне значення результатів, особистий внесок здобувача, апробацію та публікаційну активність за темою роботи. Сформульовані задачі загалом узгоджуються з метою дисертації та послідовно розв’язуються у відповідних розділах.

У першому розділі проведено систематизацію теоретичних положень щодо переносимості програм і програмних систем. Розглянуто рівні абстракції переносимості, критерії її оцінювання, стандарти програмування, особливості цільових платформ, а також компіляційні, віртуалізаційні, емуляційні та контейнерні підходи. Позитивною рисою розділу є спроба розглядати переносимість не як ізольовану властивість вихідного коду, а як результат взаємодії програмного продукту, середовища виконання та апаратно-програмної платформи. Проведене зіставлення крос-компіляції, віртуалізації та контейнеризації дало змогу обґрунтувати необхідність комбінування цих засобів у складних гетерогенних середовищах.

Другий розділ містить основні теоретичні результати дисертаційного дослідження. Здобувачем запропоновано метод адаптивної контейнеризації з інтеграцією засобів штучного інтелекту. Контейнеризовану програмну систему представлено як керований динамічний об’єкт, стан якого визначається сукупністю експлуатаційних метрик, параметрів контейнерного середовища та профілю цільової платформи.

Для кількісного оцінювання міжплатформних відмінностей уведено функцію відхилення переносимості $D(t)$, яка характеризує віддаленість поточного стану системи від еталонного профілю або допустимого експлуатаційного інтервалу. На цій основі сформульовано критерії забезпечення переносимості, визначено множину допустимих керуючих впливів та побудовано замкнений цикл керування, що охоплює моніторинг, аналіз,

прогнозування, прийняття рішень і застосування змін до контейнерної конфігурації.

Важливим результатом розділу є розроблена багаторівнева архітектура, яка передбачає взаємодію компонентів збирання телеметрії, формування стану системи, машинного навчання, вибору керуючих дій та їх реалізації засобами оркестрації. Інтелектуальний модуль використовується не лише для оцінювання поточного значення $D(t)$, а й для прогнозування стану після застосування кожного допустимого керуючого впливу. Така організація надає адаптації проактивного характеру.

У третьому розділі наведено програмну реалізацію запропонованого методу та результати його експериментальної перевірки. Програмний прототип інтегровано з Kubernetes control plane та системою моніторингу Prometheus. Теоретичні складові методу отримали безпосереднє відображення у модулях збирання телеметрії, формування вектора ознак, машинного інференсу, прийняття рішень та застосування керуючих впливів.

Експериментальне дослідження проведено для платформ x86_64 cloud та ARM64 edge за рівномірного і стрибкоподібного навантаження. Запропонований метод порівнювався з базовим режимом, у якому використовувалися статичні параметри контейнерного середовища або типові механізми масштабування без урахування профілю платформи та цільової функції переносимості.

Отримані результати засвідчили помітне зменшення значень індикатора $D(t)$. Зокрема, для платформи x86_64 за рівномірного навантаження середнє значення $D(t)$ зменшилося з 0,42 до 0,18, а частка часу порушення встановленого критерію переносимості – з 0,28 до 0,04. Для платформи ARM64 за стрибкоподібного навантаження середнє значення показника скоротилося з 0,78 до 0,33, а частка часу перебування системи поза допустимим інтервалом – з 0,56 до 0,12. Зниження стандартного відхилення та 95-го перцентиля підтверджує, що запропонований метод впливає не лише на середній рівень відхилення, а й на стабільність часової поведінки системи.

Позитивною рисою третього розділу є також обговорення меж

застосовності методу. Здобувач ураховує його залежність від повноти телеметрії та історичних даних, затримки механізмів Kubernetes, орієнтацію на режим near real-time, можливе збільшення споживання ресурсів і необхідність подальшої інтеграції з політиками інформаційної безпеки.

У четвертому розділі досліджено адаптацію програмного забезпечення на етапі компіляції. Задачу вибору параметрів компілятора подано як багатокритеріальну дискретну оптимізаційну задачу, у якій ураховуються час виконання, енергоспоживання та використання пам’яті. Для автоматизованого пошуку конфігурацій застосовано генетичний алгоритм.

Експериментальну перевірку проведено на платформах ARM Cortex-A73, Intel Xeon Platinum та NVIDIA RTX 3080. Для ARM Cortex-A73 отримано зменшення часу виконання на 35%, енергоспоживання на 50% та використання пам’яті на 25%. Для Intel Xeon Platinum відповідні показники становили 30%, 35% і 35%, а для NVIDIA RTX 3080 – 25%, 35% і 24%. Наведені дані підтверджують залежність оптимальної конфігурації компілятора від характеристик цільової платформи.

У розділі також аргументовано доцільність спільного застосування адаптивної компіляції та адаптивної контейнеризації. Оптимізація машинного коду зменшує базову потребу програми в ресурсах, тоді як контейнерний контролер компенсує зміни середовища та навантаження під час виконання. Таке поєднання дає змогу здійснювати адаптацію на двох взаємопов’язаних рівнях - build-time та run-time.

У додатках наведено матеріали щодо публікацій і впровадження результатів, а також фрагменти програмного коду, які ілюструють реалізацію ключових складових запропонованого керуючого циклу.

Загалом зміст розділів є взаємопов’язаним, а одержані теоретичні та експериментальні результати спрямовані на досягнення сформульованої мети. Дисертація справляє враження завершеного дослідження, у якому теоретична формалізація доповнена програмною реалізацією та експериментальною перевіркою.

3. Ступінь обґрунтованості та достовірності наукових положень, висновків і рекомендацій

Методологічну основу дисертаційного дослідження становлять методи системного аналізу, програмної інженерії, математичного моделювання, теорії оптимізації, аналізу часових рядів, машинного навчання, контейнеризації та експериментального оцінювання програмних систем.

Обґрунтованість одержаних результатів забезпечується послідовним переходом від теоретичної постановки задачі до її алгоритмічного та програмного втілення. Формалізовані змінні стану, профіль платформи, функція відхилення переносимості, політика керування та множина допустимих впливів відображені у структурі програмного прототипу. Це дає змогу простежити відповідність між математичними положеннями та практичними механізмами їх реалізації.

Достовірність результатів підтверджується проведенням експериментів у гетерогенному контейнеризованому середовищі, порівнянням із базовими режимами керування, використанням кількісних показників та аналізом часової динаміки. Оцінювання середніх, максимальних значень, стандартного відхилення, перцентилів і тривалості порушення заданого критерію дає змогу комплексніше характеризувати ефективність запропонованого методу.

Метод адаптивної компіляції перевірено на трьох різних класах апаратних платформ. Використання показників часу виконання, енергоспоживання та пам’яті відповідає багатокритеріальному характеру поставленої оптимізаційної задачі. Наведений аналіз збіжності генетичного алгоритму додатково характеризує стабільність пошуку конфігурацій.

Висновки до розділів і загальні висновки переважно відповідають змісту виконаних досліджень. Одержані результати пройшли апробацію на наукових конференціях, висвітлені у публікаціях здобувача та впроваджені у діяльність підприємств і в освітній процес.

Таким чином, основні наукові положення дисертації є достатньо аргументованими, а застосовані методи дослідження загалом відповідають

характеру поставлених задач.

4. Зв’язок дисертації з науковими програмами, планами і темами

Дисертаційну роботу виконано відповідно до наукового напряму кафедри програмного забезпечення комп’ютерних систем Національного технічного університету «Дніпровська політехніка» в межах теми «Високопродуктивні багатопроцесорні системи: особливості конструювання, дослідження оцінок ефективності, застосування до розв’язування прикладних задач» (Е-340, номер державної реєстрації 0121U113718).

Тематика дисертації безпосередньо пов’язана з розробленням методів і програмних технологій створення багатоплатформних систем, підвищенням ефективності використання обчислювальних ресурсів та автоматизацією керування програмним забезпеченням у гетерогенних обчислювальних середовищах.

5. Наукова новизна та практичне значення одержаних результатів

Наукова новизна дисертаційного дослідження полягає насамперед у формуванні комплексного підходу до забезпечення переносимості, у межах якого вона розглядається не лише як статична властивість програмного коду або середовища розгортання, а як динамічно керована характеристика поведінки програмної системи.

До найбільш вагомих результатів роботи належать:

1. Розроблення методу адаптивної контейнеризації з інтеграцією машинного навчання, який передбачає автоматизоване оцінювання стану програмної системи, прогнозування наслідків керуючих впливів і динамічне налаштування контейнерного середовища відповідно до профілю платформи та характеру навантаження.

2. Формалізація контейнеризованої програмної системи як керованого динамічного об’єкта та введення кількісного індикатора D(t), що дає змогу оцінювати відхилення поточної поведінки системи від еталонного профілю або

допустимого експлуатаційного інтервалу.

3. Розроблення архітектури інтелектуального керування контейнеризованими програмними системами, яка поєднує засоби моніторингу, формування ознак, машинного інференсу, вибору керуючих дій та їх декларативного застосування через систему оркестрації.

4. Розроблення програмної технології прогнозування потреб програмної системи в ресурсах на підставі поточних і накопичених експлуатаційних даних, що забезпечує перехід від реактивного до проактивного керування.

5. Розроблення методу адаптивної компіляції на основі генетичного алгоритму, у якому вибір параметрів компілятора здійснюється з урахуванням кількох експлуатаційних критеріїв та характеристик цільової апаратної платформи.

6. Обґрунтування комплексного застосування адаптації на етапах компіляції та виконання, що забезпечує узгоджене зменшення базових ресурсних потреб програмного коду та компенсацію динамічних змін середовища.

Одержані результати розвивають методи забезпечення переносимості програмного забезпечення у гетерогенних середовищах і розширюють функціональні можливості контейнерних технологій завдяки включенню інтелектуального керуючого контуру.

Практичне значення результатів полягає у можливості застосування розроблених методів під час створення, розгортання та супроводу програмних систем у хмарних, мультихмарних, периферійних і вбудованих середовищах. Програмний прототип інтегрується зі стандартними засобами Kubernetes і Prometheus, що спрощує його включення до сучасних DevOps- та CI/CD-процесів.

Використання запропонованого підходу дає змогу зменшувати залежність від ручного налаштування контейнерів, автоматизувати вертикальне та горизонтальне масштабування, ураховувати специфіку апаратної платформи та підтримувати стабільність поведінки системи за змінного навантаження.

Практична цінність результатів підтверджується їх упровадженням у діяльність ТОВ НВП «Агропромавтоматизація», іноземного підприємства «SoftRequest LTD», Державного науково-виробничого підприємства «Ельдорадо», ТОВ «Біологічні активні добавки», а також в освітній процес Національного технічного університету «Дніпровська політехніка».

6. Повнота висвітлення результатів дисертації у наукових публікаціях

Основні положення та результати дисертаційного дослідження опубліковано у 17 наукових працях. Шість статей надруковано у наукових виданнях, включених до переліку наукових фахових видань України. Одна зі статей опублікована у виданні, що індексується у наукометричній базі Web of Science. Одинадцять праць представлено у збірниках матеріалів міжнародних наукових і науково-практичних конференцій.

Тематика публікацій охоплює адаптивну контейнеризацію, використання методів машинного навчання для оптимізації хмарних середовищ, адаптивну компіляцію, переносимість програмного коду C++, автоматизоване виявлення платформозалежних компонентів та оцінювання переносимості програм у гетерогенних системах.

Зміст опублікованих праць загалом відображає основні теоретичні й прикладні результати дисертації. У роботі наведено відомості про особистий внесок здобувача у спільні публікації. Кількість і спрямованість наукових праць відповідають характеру виконаного дослідження та забезпечують належну апробацію його результатів.

7. Зауваження та дискусійні положення

Позитивно оцінюючи дисертаційну роботу загалом, необхідно висловити такі зауваження та побажання.

1. У дисертації переносимість значною мірою оцінюється через відхилення експлуатаційних показників від еталонного профілю. Водночас такі показники, як затримка, використання процесора, пам’яті або пропускна здатність,

традиційно характеризують також продуктивність, ефективність і стабільність програмної системи. Тому доцільним було б детальніше обґрунтувати межі між поняттями переносимості, адаптивності та ефективності, а також чіткіше показати, за яких умов зменшення $D(t)$ слід інтерпретувати саме як підвищення переносимості.

2. Формулювання наукової новизни у вступі не повністю охоплює результати четвертого розділу. Розроблений метод адаптивної компіляції на основі генетичного алгоритму фактично є самостійним науковим результатом, однак у переліку положень наукової новизни він не отримав достатньо чіткого відображення. Доцільно було б узгодити формулювання наукової новизни, задач дослідження та загальних висновків із фактичним змістом четвертого розділу.

3. У функції відхилення переносимості $D(t)$ використовуються вагові коефіцієнти, еталонні значення, допустимі інтервали та порогові параметри ε, T і ρ. У роботі зазначено, що їх вибір залежить від вимог програмної системи та рівня сервісу, проте методику визначення цих параметрів висвітлено недостатньо докладно. Додатковий аналіз чутливості результатів до зміни вагових коефіцієнтів і порогів посилив би обґрунтованість запропонованої моделі.

4. Для оцінювання та прогнозування $D(t)$ передбачено застосування регресійних моделей, зокрема Random Forest, Gradient Boosting, лінійної регресії та неглибоких нейронних мереж. Водночас у дисертації бракує розгорнутого опису обсягу і структури навчальної вибірки, процедури поділу даних, налаштування гіперпараметрів та кількісних показників точності моделі, наприклад MAE, RMSE або R^2. Наведення таких даних дало б змогу повніше оцінити якість інтелектуального модуля та його здатність до узагальнення.

5. Експериментальну перевірку адаптивної контейнеризації проведено для двох платформних профілів і двох основних сценаріїв навантаження. Для ширшого підтвердження узагальнюваності результатів доцільно було б розширити експериментальну базу, включивши інші апаратні архітектури, класи прикладних програм, змішані навантаження та сценарії відмов. Крім того,

бажано чітко зазначити кількість повторних запусків для експериментів третього розділу та навести довірчі інтервали одержаних оцінок.

6. Як базовий режим використано статичні ресурсні обмеження та типові механізми autoscaling Kubernetes. Порівняння з сучасними інтелектуальними засобами керування ресурсами, зокрема прогнозними алгоритмами масштабування, Vertical Pod Autoscaler або підходами на основі навчання з підкріпленням, дало б змогу точніше визначити переваги запропонованого методу. Доцільним також є кількісне оцінювання накладних витрат самого контролера та економічного компромісу між стабілізацією переносимості й додатковим споживанням ресурсів.

Висловлені зауваження мають переважно рекомендаційний і дискусійний характер. Вони не заперечують достовірності основних результатів, не знижують наукової цінності дисертації та можуть розглядатися як напрями подальшого розвитку запропонованих методів.

Загальний висновок

Дисертаційна робота Стародубського Ігоря Петровича «Методи та засоби переносимості програм та програмних систем на різні обчислювальні платформи» є завершеною самостійною кваліфікаційною науковою працею, у якій розв’язано актуальну науково-прикладну задачу підвищення переносимості та автоматизації адаптації програмних систем до різних обчислювальних платформ.

У роботі розроблено метод адаптивної контейнеризації з інтеграцією машинного навчання, формалізовано програмну систему як керований динамічний об’єкт, запропоновано кількісний індикатор відхилення переносимості, побудовано архітектуру замкненого керуючого циклу та створено відповідний програмний прототип. Експериментально підтверджено здатність запропонованого методу зменшувати міжплатформні відмінності у поведінці програмної системи та стабілізувати її функціонування за змінного навантаження.

Окремим вагомим результатом є розроблення методу адаптивної компіляції на основі генетичного алгоритму та обґрунтування комплексного застосування адаптації на рівнях компіляції й виконання. Теоретичні положення, алгоритмічні рішення, програмна реалізація та експериментальні результати утворюють логічно пов’язану систему.

Основні результати дисертації достатньо повно представлено у наукових публікаціях, апробовано на наукових конференціях і впроваджено у практичну діяльність та освітній процес. Наукові положення й висновки є аргументованими, а практична спрямованість роботи відповідає сучасним потребам розроблення та експлуатації багатоплатформних програмних систем.

За актуальністю, обсягом виконаних досліджень, рівнем наукової новизни, обґрунтованістю одержаних результатів і практичним значенням дисертаційна робота відповідає вимогам «Порядку присудження ступеня доктора філософії та скасування рішення разової спеціалізованої вченої ради закладу вищої освіти, наукової установи про присудження ступеня доктора філософії», затвердженого постановою Кабінету Міністрів України від 12 січня 2022 р. № 44, а її автор, Стародубський Ігор Петрович, заслуговує на присудження ступеня доктора філософії за спеціальністю 122 «Комп’ютерні науки».

Офіційний опонент:

професор кафедри програмних систем і технологій
факультету інформаційних технологій
Київського національного університету
імені Тараса Шевченка,
доктор технічних наук, професор  Олександр ВОВНА