ПЕДАГОГІЧНА АКАДЕМІЯ:
НАУКОВІ ЗАПИСКИ

Інформаційно-комунікаційні технології в освіті

УДК 004.6:378.147; 37.091.3

DOI https://doi.org/10.5281/zenodo.17195759

Навчальна інфраструктура Big Data: локальний Proxmox-кластер для ETL і Spark SQL

Сіциліцин Юрій Олександрович

PhD, старший викладач кафедри інформатики і кібернетики Мелітопольського державного педагогічного університету імені Богдана Хмельницького, 69000, м. Запоріжжя, вул. Наукового містечка, 59, Україна, Sicylicyn_Yurij@mspu.edu.ua, https://orcid.org/0000-0002-3888-5575

Лубко Дмитро Вікторович

Кандидат технічних наук, доцент кафедри комп'ютерних наук Таврійського державного агротехнологічного університету імені Дмитра Моторного, 69600, м. Запоріжжя, вул. Жуковського, 66, Україна, di75ma@gmail.com, http://orcid.org/0000-0002-2506-4145

Прийнято: 12.09.2025 | Опубліковано: 23.09.2025

Анотація: *Стаття пропонує методично вивірений підхід до розгортання та педагогічної валідації локальної навчальної інфраструктури Big Data на базі Proxmox із кластером Hadoop/Spark для виконання лабораторних робіт на курсах опрацювання даних. На відміну від попередньої публікації автора, де було описано архітектуру віртуалізованого середовища та організацію доступу, у цій роботі зосереджено увагу на мінімальному відтворюваному протоколі експериментів, який безпосередньо пов'язує технічні метрики продуктивності зі зрозумілими педагогічними індикаторами якості навчального процесу.*

Запропонований протокол включає стандартизовану послідовність завдань (ETL → Spark SQL → аналіз стадій), фіксацію версій компонентів (операційна система, JDK, Hadoop, Spark), контроль вхідних даних і єдині інструкції для студентів. Для агрегування результатів обрано стійкі статистики – медіану та міжквартильний розмах (IQR), а також профілі стадій виконання, що дозволяє зменшити вплив поодиноких збоїв і забезпечити інтерпретованість показників у межах академічного заняття. Педагогічна валідація здійснюється через операціоналізовані індикатори: передбачуваність таймінгу пари (частка групи, що вкладається у відведений слот), прозорість артефактів (можливість перевірки ходу виконання за логами/ноутбуками/звітами), кількість технічних інцидентів, а також сприйняття інструкцій за короткою анкетою. Окремо розглядається порівняльна рамка «локальна інфраструктура vs хмара» в освітньому контексті: сумарні витрати на курс, стабільність і керованість виконання завдань, залежність від зовнішніх сервісів, вимоги до підтримки та доступність для студентів із різним рівнем підготовки. Емпіричні результати демонструють, що локальний кластер на Proxmox забезпечує кращу керованість і стабільніші часові характеристики типових завдань без втрати технічної репрезентативності, що важливо для планування і оцінювання навчальної діяльності. Практичний внесок роботи полягає у формалізації відтворюваного мінімуму Big Data-експериментів для аудиторної та змішаної форм, узгодженні технічних метрик із освітніми індикаторами та наданні інструкцій, придатних для масштабування курсу і міжкурсових порівнянь. Обмеження дослідження стосуються розміру кластера та набору завдань; подальша робота передбачає автоматизацію збору метрик, розширення корпусу даних і перевірку підходу на різних програмах підготовки.

Ключові слова: *Big Data, Hadoop, Spark, Proxmox, навчальна інфраструктура, відтворюваність, медіана, IQR, ETL, Spark SQL, педагогічні індикатори.*

ПЕДАГОГІЧНА АКАДЕМІЯ:
НАУКОВІ ЗАПИСКИ

Big Data training infrastructure: local Proxmox cluster for ETL and Spark SQL

Yurii Sitsylitsyn

PhD, Senior Lecturer at the Department of Informatics and Cybernetics, Bogdan Khmelnitsky Melitopol State Pedagogical University, 69000, Zaporizhzhia, Naukovogo Mistechka St., 59, Ukraine, Sicylicyn_Yurij@mspu.edu.ua, https://orcid.org/0000-0002-3888-5575

Lubko Dmytro

Candidate of Technical Sciences, Associate Professor, Department of Computer Science, Dmytro Motornyi Tavria State Agrotechnological University, 69600, Zaporizhzhia, str. Zhukovsky, 66, Ukraine, di75ma@gmail.com, http://orcid.org/0000-0002-2506-4145

Abstract: *The article proposes a methodologically grounded approach to deploying and pedagogically validating a local Big Data teaching infrastructure based on Proxmox with a Hadoop/Spark cluster for conducting laboratory work in data-processing courses. Unlike the author's previous publication, which described the architecture of the virtualized environment and access organization, this study focuses on a minimal reproducible experimental protocol that directly links technical performance metrics with clear educational indicators of instructional quality. The proposed protocol includes a standardized sequence of tasks (ETL → Spark SQL → stage analysis), version pinning of components (operating system, JDK, Hadoop, Spark), controlled input data, and unified student instructions. To aggregate results, we employ robust statistics – median and interquartile range (IQR) – as well as stage execution profiles, which reduces the impact of outliers and ensures interpretability of measurements within an academic class session. Pedagogical validation is carried out through operationalized indicators: predictability of class timing (the share of the group that completes work within the allotted slot), transparency of artifacts (the*

ability to verify progress via logs/notebooks/reports), number of technical incidents, and perceived clarity of instructions via a short survey. We also examine an education-oriented comparison framework of "local infrastructure vs cloud": total course costs, stability and controllability of task execution, dependence on external services, support requirements, and accessibility for students with varying levels of preparation. Empirical results show that a local Proxmox-based cluster provides better controllability and more stable time characteristics for typical tasks without sacrificing technical representativeness – an essential factor for planning and assessing learning activities. The practical contribution lies in formalizing a reproducible minimum of Big Data experiments for in-class and blended formats, aligning technical metrics with educational indicators, and providing instructions suitable for course scaling and cross-course comparisons. The study's limitations concern the cluster size and the set of tasks; future work includes automating metric collection, expanding the data corpus, and validating the approach across different academic programs.

Keywords: *Big Data, Hadoop, Spark, Proxmox, teaching infrastructure, reproducibility, median, IQR, ETL, Spark SQL, educational indicators.*

Постановка проблеми. У сучасних курсах з опрацювання даних необхідно поєднати технічну репрезентативність інструментів (Hadoop/Spark) з передбачуваністю й керованістю навчального процесу. Типові труднощі – неоднорідність студентських машин, нестабільність мережевих умов і залежність від сторонніх хмарних сервісів – призводять до збоїв у таймінгу занять і знижують відтворюваність результатів.

Попередня робота авторів [1] окреслила рішення на рівні віртуалізованої інфраструктури (Proxmox) та організації віддаленого доступу, що спростило адміністрування й підвищило контрольованість середовища. Водночас вона не надавала формалізованого протоколу навчальних експериментів і не пов'язувала технічні метрики з педагогічними індикаторами.

У цій статті проблема формулюється інакше: як забезпечити відтворюваний мінімум експериментів Big Data, придатний для щотижневих лабораторних робіт, та як напряму зв'язати технічні показники продуктивності зі сталістю таймінгу пари, зрозумілістю інструкцій і часткою студентів, які завершують завдання в межах регламенту.

Запропонований підхід спирається на локальний кластер Hadoop/Spark поверх Proxmox і мінімально достатній стек інструментів. Основна ідея – замкнути повний цикл «дані → обчислення → інтерпретація метрик» у межах керованого середовища з фіксацією версій і прозорими артефактами, щоб будь-яку лабораторну можна було відтворити та перевірити.

Окремим завданням є побудова порівняльної рамки «локальна інфраструктура vs хмара» не у виробничих, а саме в освітніх координатах: сумарні витрати на курс, стабільність виконання завдань у межах академічних слотів, доступність для студентів із різним рівнем технічної підготовки, та навантаження на викладача.

Аналіз останніх досліджень і публікацій. Після 2020 року у фаховій літературі чітко фіксується тренд на інтеграцію тем Big Data та хмарних обчислень у навчальні плани з інформатики й інженерії даних. Огляди та практичні кейси підкреслюють потребу створення навчальної програми як системи, де поряд із алгоритмічними темами вибудовується компетентність роботи з розподіленими платформами та інфраструктурою даних [2], а Apache Spark фактично слугує «ядром» навчальних сценаріїв завдяки уніфікації API (SQL/DataFrame/MLlib) і масштабованості від одного вузла до кластера [3]. На рівні інфраструктурних рішень активізувалося використання керованих хмарних сервісів (наприклад, Google Cloud Dataproc) для лабораторій і курсів, що спрощує старт, але переносить питання вартості й політик безпеки в центр прийняття рішень закладами освіти [4].

ISSN: 2786-9458

Паралельно розвивається напрям локальних віртуалізованих середовищ у ЗВО. Публікації демонструють, що Proxmox VE може бути основою для академічних хмар і мережевих лабораторій, забезпечуючи керованість, гнучке виділення ресурсів і відтворюваність конфігурацій [5, 6]. На методичному рівні дослідження застосування Hadoop/Spark у навчанні засвідчують зростання залученості студентів та кращу операціоналізацію практик обробки даних, однак наголошують на потребі чітких протоколів оцінювання й стандартних завдань (ETL, SQL-агрегації, типові бенчмарки) [7, 8, 9]. Окремі прикладні роботи показують релевантність Spark для міждисциплінарних кейсів (ризик-менеджмент, енергетика), що підсилює аргумент про цінність цієї платформи в навчальному процесі як інструмента наближеного до реального виробництва [4, 5].

Водночас у літературі чітко артикульовано проблему відтворюваності результатів у прикладних і навчальних дослідженнях: від заклику до стандартизації протоколів вимірювань до вимоги публікувати код/ноутбуки та забезпечувати повторюваність обчислень у гетерогенних середовищах [10 - 15]. Для освітніх кластерів це означає фіксацію мінімуму методичних правил: кількість повторів, узагальнення медіаною та IQR, чітка фіксація конфігурацій та версій. На цьому тлі перспективним виглядає підхід, що поєднує локальну інфраструктуру (Proxmox) з відтворюваними сценаріями Hadoop/Spark та прозорими процедурами оцінювання. Додатково, власна попередня робота авторів щодо розбудови університетської екосистеми на Proxmox задає емпіричне підґрунтя для переходу від загальноосвітніх сервісів до спеціалізованого кластерного середовища Big Data та його педагогічної валідації [1]. Таким чином, сучасний стан досліджень підтримує обраний нами вектор: баланс між зручністю хмари й керованістю локальних кластерів із наголосом на відтворюваність та методично обґрунтовані метрики.

Виділення невирішених раніше частин загальної проблеми. Попри значний прогрес у впровадженні Big Data-технологій в освіті, низка ключових аспектів залишається нерозв'язаною або опрацьованою фрагментарно. Розглянемо ці аспекти. З точки зору мінімального експериментального протоколу для освітніх кластерів, у літературі бракує узгоджених правил вимірювань (набір базових завдань, кількість повторів, способи узагальнення – медіана/IQR, фіксація версій ПЗ та конфігурацій). Через це результати курсів і лабораторій важко порівнювати між університетами, а накопичення доказової бази відбувається повільно. Розглядаючи аспект зв'язку технічних метрик із освітніми результатами слід зазначити, що більшість робіт оцінює продуктивність (час, масштабованість), але не поєднує її з педагогічними індикаторами: часткою студентів, що встигають у межах пари, стабільністю виконання, суб'єктивною зручністю, валідними інструментами опитування. Відсутній загальноприйнятий набір показників «техніка → навчальний ефект». Також не достатньо розкрити аспект методично обґрунтованого порівняння «локально vs хмара». Існують поодинокі оцінки зручності або вартості, однак відсутні контрольовані розробки, які одночасно враховують TCO, політики безпеки/даних, стійкість до зовнішніх збоїв (блекаути, обмеження доступу) і якісний досвід студентів у дистанційному форматі. Хотілось би докладніше дослідити викладання Big Data у контексті малих і переміщених університетів. Більшість кейсів виходить із припущення про стабільну мережеву інфраструктуру та доступ до хмарних бюджетів. Для невеликих і переміщених закладів, де «дано» – власний сервер і «не дано» – значні кошти на оренду хмари, немає типових рішень і рекомендацій щодо пріоритизації сервісів, планування навантажень і поетапної еволюції середовища. Також залишається недоопрацьованим баланс між оперативністю розгортання, прозорістю навчальних даних (логи, артефакти job-ів) та вимогами до захисту персональної інформації.

У цій роботі ми адресуємо зазначені прогалини, пропонуючи: відтворюваний «мінімум» експериментів (набір завдань, правило повторів, метрики медіана/IQR, фіксація версій); пов'язування технічних показників із освітніми індикаторами; відкриті артефакти (структури даних/скрипти аналізу); рамку порівняння локальної та хмарної траєкторій, релевантну для невеликих і переміщених університетів. Це створює основу для кумулятивних, зіставних і практично корисних результатів.

Формулювання цілей статті. Загальна мета статті це обґрунтувати та експериментально підтвердити доцільність локальної навчальної інфраструктури Big Data на базі Proxmox для підтримки курсів з обробки даних та аналітики, забезпечивши відтворювані лабораторії на стеку Hadoop/Spark за умов обмежених ресурсів переміщеного й невеликого університету. У цьому контексті розглянемо такі специфічні цілі статті як проєктування мінімально достатньої топології (1 master + 1..3 workers) і опису стандартизованих процесів розгортання HDFS/YARN та Spark із фіксацією версій і конфігурацій. Виміряти продуктивність типових навчальних завдань (WordCount, Spark SQL агрегати) на наборах 0.5–3.0 ГБ із 5-кратними повторами для 1/2/3 воркерів; узагальнити результати за медіаною та IQR; оцінити характер масштабування. Зібрати індикатори навчальної ефективності (частка завершення в межах пари; суб'єктивна зручність/стабільність) для студентів, що навчаються дистанційно та співвіднести їх із технічними метриками. Сформулювати рамку зіставлення локальної та хмарної траєкторій для освітніх курсів із урахуванням TCO, стійкості до зовнішніх збоїв та вимог безпеки. Виробити рекомендації щодо планування занять (обсяги даних, число воркерів, тривалість етапів, типові ризики та їхня профілактика).

В результаті плануємо отримати набір перевірених практик і метрик, які дозволяють надійно проводити Big Data-лабораторії локально та масштабувати під різні курси й умови.

Виклад основного матеріалу дослідження. Метою дослідження є побудова та валідація локальної навчально-дослідницької інфраструктури Big Data на базі платформи віртуалізації Proxmox VE з розгортанням стеку Hadoop/Spark для підтримки лабораторних робіт з обробки даних та аналітики. Дослідження спирається на три взаємопов'язані компоненти: (1) інженерний – архітектура та процес розгортання; (2) експериментальний – вимірювання продуктивності типових навчальних завдань за стандартизованим протоколом; (3) педагогічний – оцінка навчальної доцільності за допомогою емпіричних освітніх індикаторів (завершення робіт у межах пари, суб'єктивна зручність і стабільність).

Ключові вимоги до інфраструктури: керованість, відтворюваність, автономність (мінімальна залежність від зовнішніх сервісів), прозорість конфігурацій та обмежена вартість володіння. Додатковими практичними чинниками виступають наявність у закладу власного серверного обладнання, переміщений статус університету та обмежений бюджет на оренду хмарних сервісів, що стимулює пошук локальних рішень із прогнозованим.

Дослідна інфраструктура розгорнута на сервері з 16 vCPU (Intel Xeon E-2378 @ 2.60 GHz) та 126 GiB RAM. Диски розділено логічно: системні образи ВМ – на окремому томі, дані HDFS – на іншому томі для зменшення взаємного впливу I/O.

Використано Proxmox VE 8.x (KVM/QEMU). Переваги вибору Proxmox для навчального контексту [5]:

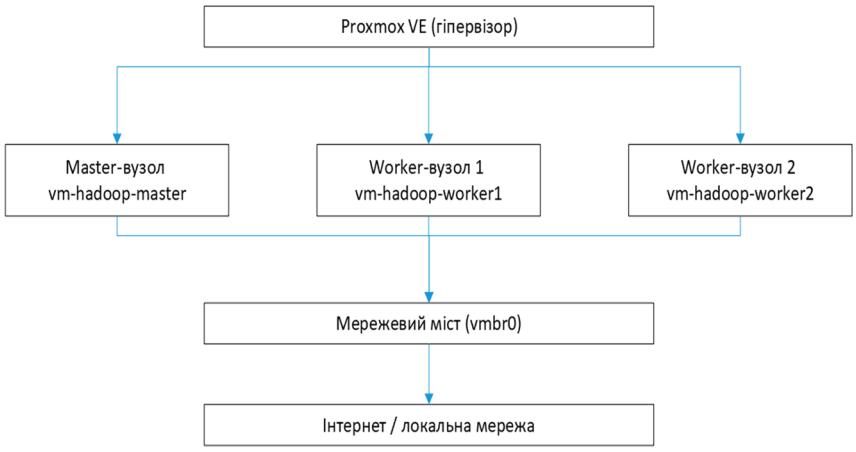
- просте адміністрування образів ВМ;
- швидке клонування та масштабування вузлів;
- вбудований моніторинг і зручне керування ресурсами;
- можливість ізоляції «навчальних» і «сервісних» навантажень на рівні різних storage pool'ів.

ISSN: 2786-9458

Мінімальна топологія – один вузол керування (Master) і змінна кількість обчислювальних вузлів (Workers = 1..3). На кожній ВМ – Ubuntu Server LTS, OpenJDK 11. Програмний стек: Hadoop 3.3.6 (HDFS, YARN), Apache Spark 3.4.1 (Scala/PySpark). HDFS забезпечує зберігання даних з реплікацією (фактор 2 для тестів). YARN відповідає за планування ресурсів та виконання контейнерів (NodeManager/ResourceManager). Spark працює в режимі --master yarn та --deploy-mode cluster. ВМ підключено до окремого віртуального бриджу Proxmox (vmbr-edu) з адресним простором, зарезервованим під навчальну мережу; це спрощує відлагодження та ізолює трафік лабораторій (рисунок 1).

Рисунок 1

Діаграма архітектури вузлів та мереж



Після розгортання було виконано форматування HDFS, перевірка стану NameNode/DataNode; старт ResourceManager/NodeManager, перевірка реєстрації контейнерів; тестові запуски spark-shell/PySpark і прикладу WordCount на невеликому файлі; інспекція Spark UI і YARN RM UI для підтвердження коректного розподілу стадій (Таблиця 1).

Таблиця 1

Структура кластера

Роль	Віртуальна машина	Основні служби
Master	vm-hadoop-master	NameNode, ResourceManager, Spark Driver
Worker #1	vm-hadoop-worker1	DataNode, NodeManager, Spark Executor
Worker #2	vm-hadoop-worker2	DataNode, NodeManager, Spark Executor
Worker #3	vm-hadoop-worker3	DataNode, NodeManager, Spark Executor

Для перевірки того, як локальний кластер поводиться в типових навчальних умовах, ми зібрали два зрозумілих і водночас показових сценарії. Перший - WordCount: класична «текстова аналітика», де Spark проходить повний шлях від читання великих текстових файлів у HDFS до підрахунку частот слів. Цей кейс зручний тим, що «оголює» інфраструктуру: помітно, як на час впливають реплікація в HDFS, мережеві передачі та shuffle між вузлами. Другий сценарій - Spark SQL-агрегати над простою CSV-таблицею events(id, category, value) із шістьма категоріями. Ми свідомо обрали базові операції COUNT/AVG/SUM BY category і вивантаження результатів у Parquet – саме такі дії студенти найчастіше виконують на перших лабораторних, а отже, результати легко інтерпретувати.

Щоб завдання залишалися реалістичними, для кожного сценарію ми підготували синтетичні набори даних на 0.5, 1.0 та 3.0 ГБ. Такий обсяг дає змогу відчути різницю між одним і кількома воркерами, не виходячи за межі аудиторного часу.

Далі, для кожної комбінації параметрів – тип завдання × розмір даних × кількість воркерів (1, 2 або 3) – ми запускаємо завдання п’ять разів. Тривалість фіксуємо у секундах (через /usr/bin/time та/або зі Spark UI). Після цього обчислюємо медіану (як стійку оцінку типового часу) і міжквартильний розмах (IQR) - він показує, наскільки коливаються результати від запуску до запуску.

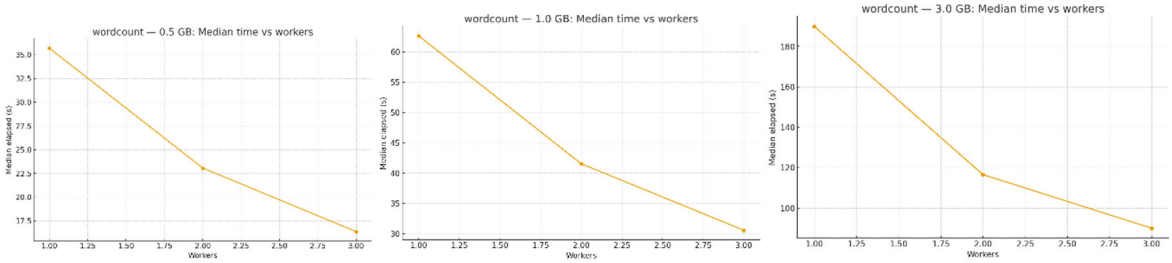
Для викладача це критично: медіана дає опору для планування, а IQR - уявлення про ризики «довгих» прогонів.

Тести розпочали з класичного завдання підрахунку слів WordCount. На 0.5 ГБ картина прозора: медіана часу падає з 35.70 с на одному воркері до 23.07 с на двох і 16.36 с на трьох (IQR стискається з 1.10 до 0.84). На 1.0 ГБ тенденція зберігається: 62.64 → 41.56 → 30.59 с (IQR 1.17 → 1.03 → 0.88). Навіть на 3.0 ГБ, де вплив диска й мережі найпомітніший, маємо відчутний спад: 189.90 → 116.59 → 90.06 с, а розкид між повторами падає з 10.82 до 1.71 секунд.

Це дає два практичні висновки. По-перше, прискорення сублінійне: перехід 1→3 воркери приносить близько 2.2× виграшу на малих і ≈2.1× - на середніх/великих наборах. По-друге, більше воркерів – це не лише швидше, а й стабільніше: IQR помітно зменшується, отже час виконання стає передбачуванішим. Для аудиторної роботи така передбачуваність є принципово важливою: вона дає змогу детально планувати тривалість етапів без непередбачуваних затримок. На графіках (рис. 2а–в) залежності мають випуклий характер: найбільший приріст продуктивності фіксується під час масштабування з 1 до 2 воркерів, тоді як подальше збільшення їх кількості забезпечує менший, маргінально спадний ефект.

Рисунок 2

Завдання підрахунку WordCount



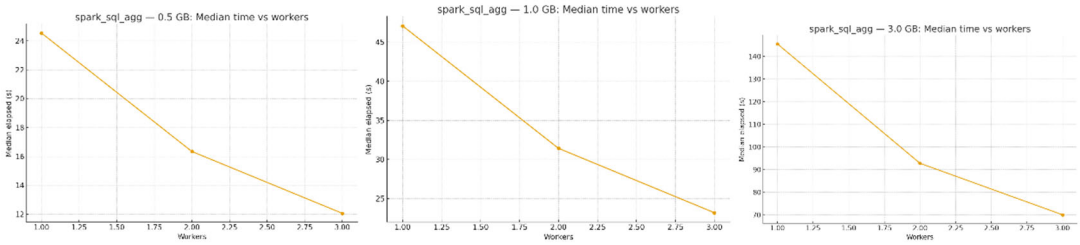
а) б) в)

У SQL-сценарії картина ще приємніша. На 0.5 ГБ медіани 24.54 → 16.34 → 12.06 с при IQR 1.06 → 0.64 → 0.59. На 1.0 ГБ – 47.08 → 31.42 → 23.19 с (IQR

1.76 → 1.33 → 1.44), на 3.0 ГБ – 145.50 → 92.73 → 69.96 с (IQR 2.26 → 3.27 → 2.81). Тобто аналітичні завдання на Spark SQL стабільно швидші за WordCount на тих самих обсягах – заслуга оптимізатора Catalyst і колонкових форматів. Важливо й інше: низький IQR (≈0.6–3.3 с) означає мінімум «довгих хвостів», які зазвичай вибивають заняття з ритму. На рисунку 3 а–в добре видно односпрямоване спадання медіан із додаванням воркерів.

Рисунок 3

Spark SQL-агрегати



а) б) в)

Якщо звести все в одну рамку, то на 1.0 ГБ маємо ≈2.05× прискорення для WordCount (62.64 → 30.59 с) і ≈2.03× – для Spark SQL (47.08 → 23.19 с). На 3.0 ГБ картина схожа: ≈2.11× (WordCount: 189.90 → 90.06 с) і ≈2.08× (Spark SQL: 145.50 → 69.96 с). Тобто горизонтальне масштабування 1→3 воркери дає стале ~2× скорочення часу в типових навчальних задачах і робить їх передбачуваними завдяки низькому IQR.

Отримані показники продуктивності засвідчують, що навіть на невеликому кластері можливо надійно планувати повний цикл ETL → SQL → збереження у межах одного академічного заняття. Завдання WordCount репрезентує більш ресурсоємний, «важкий» клас розподілених обчислень (виразний вплив I/O, реплікації HDFS і shuffle), тоді як Spark SQL демонструє швидкі інтерактивні агрегування з нижчими часовими витратами. У поєднанні ці сценарії формують керований навчальний модуль, у межах якого студенти спостерігають як ефект горизонтального масштабування, так і структурну логіку інфраструктури на

кількісно підтверджених даних; викладач, своєю чергою, оперує передбачуваним хронометражем.

Для перевірки віповідності «апаратних» метрик із навчальними результатами проведено пілотне анонімне опитування (n = 15) з мінімальним, інструментарієм: два бінарні показники (завершення роботи в межах заняття; інсталяція без критичних збоїв), чотири шкальні оцінки за Likert (зручність, стабільність, зрозумілість інструкцій, сприйнята продуктивність) та відкрите запитання щодо поліпшень. Підсумки узгоджуються з технічними вимірами: 66.7% респондентів завершили завдання в межах пари, 80.0 % не стикалися з критичними збоями під час інсталяції. Медіанні оцінки 4 за всіма шкалами при IQR ≈ 0–1 вказують на однорідне сприйняття інструкцій і стабільності середовища; вузький розмах для «зрозумілості» відображає коректність і достатність методичних матеріалів, а показники «стабільності» та «сприйнятої продуктивності» консистентні з низькою варіативністю часу виконання (IQR зменшується зі зростанням кількості воркерів). Якісні відгуки підтверджують кількісні дані: відзначено зручність запуску робіт через веб-інтерфейси та належну структурованість інструкцій; як напрями поліпшення названо розширення пулу прикладів SQL (зокрема, віконні функції) та підготовку стислої пам’ятки щодо типових помилок YARN і процедур відновлення.

З позиції організації заняття отримані часові оцінки для 1.0 ГБ і трьох воркерів можуть слугувати робочим нормативом: WordCount ≈ 30.6 с, Spark SQL ≈ 23.2 с для одного запуску. Це створює часовий резерв на інструктаж, підготовку даних та обговорення результатів. Раціональна послідовність дій включає: (i) імпорт та первинну очистку даних у HDFS; (ii) серію прогонів WordCount із варіюванням числа воркерів для демонстрації впливу конфігурації на часовий профіль; (iii) виконання блоку Spark SQL (групування, агрегати, запис у Parquet) із фіксацією часу відповіді; (iv) аналітичний «розбір» у Spark UI (стадії, кількість завдань, обсяги shuffle). Така структура забезпечує прозорість

причинно-наслідкових зв'язків (конфігурація → план виконання → час/стабільність) і водночас зберігає контроль над хронометражем завдяки низькій варіативності вимірювань.

Методично це означає перехід від разової демонстрації до формування стійких компетентностей. Студент засвоює не лише синтаксис API, а й операційну логіку системи: здатність локалізувати проблему за індикаторами UI/логів, коректно модифікувати конфігураційні файли та відновлювати працездатність сервісів. Відповідно, технічні метрики, педагогічні індикатори та якісні відгуки утворюють узгоджену картину: масштабування підвищує не тільки швидкодію, а й передбачуваність навчального процесу, що є ключовою умовою для планування й ефективної реалізації лабораторних занять у розподілених середовищах.

Виходячи з виконаних експериментів зробимо висновки локально проти хмари. Переваги локального підходу полягають у повному контролі над апаратно-програмним стеком, автономності функціонування за відсутності зовнішніх сервісів, передбачуваності продуктивності та нульових «погодинних» експлуатаційних витратах у процесі навчання. Крім того, локальний сценарій спрощує відтворюваність конфігурацій і протоколів вимірювань, що критично для навчальних досліджень. До обмежень належать необхідність системного адміністрування (оновлення, моніторинг, резервне копіювання), фізичні межі масштабування ресурсів та відповідальність за забезпечення відмовостійкості (резервування, тестування відновлення). Керовані платформи надають еластичне масштабування, інтеграції з суміжними інструментами (сховища, каталоги даних, CI/CD) та скорочують час первинного розгортання. Водночас вони створюють залежність від бюджетних обмежень (кредити/квоти), політик доступу та мережевої доступності; зростають ризики втрати контролю над даними й варіативності витрат у динаміці семестру. Виходячи з цього, пропонуємо дворівневу рамку: (і) базовий модуль реалізується локально на

 ISSN: 2786-9458

Proxmox (стабільність, відтворюваність, відсутність змінних витрат, чіткий контроль конфігурацій і оцінювання); (ii) поглиблений модуль передбачає «екскурс у хмару» з фіксованим бюджетом, заздалегідь підготованими скриптами та артефактами перенесення, що формує компетенції портованості між локальним і хмарним виконанням.

Вибірка опитування (n = 15) має пілотний характер і обмежену статистичну потужність. Заплановано розширення вибірки та стратифікацію за рівнем підготовки і формою навчання, а також повторні вимірювання для перевірки стійкості ефектів. Використані датасети не відтворюють усю різноманітність реальних профілів навантажень (ентропія, кореляції ознак, «перекоси» категорій). Для підвищення зовнішньої валідності передбачено поетапне введення відкритих реальних датасетів (CSV/Parquet) та сценаріїв MLlib із більш інтенсивним shuffle/join.

Оцінки масштабування отримано для малих кластерів; екстраполяція на більші N має здійснюватися з обережністю. Наступний етап включатиме експерименти на 4–5 воркерах, варіацію параметрів розміру партицій і моделювання високошафлових навантажень.

У поточному циклі не проводилися тести ін’єкції відмов (вивід із ладу NodeManager/DataNode, деградація мережі). Планується окремий модуль fault-injection із метриками часу відновлення та втрат продуктивності, що поглибить аналіз відмовостійкості навчального середовища.

За результатами досліджень розглянемо практичні рекомендації для викладачів.

Для академічної групи 12–20 осіб за конфігурації кластера «3 worker» доцільно закладати такий часовий бюджет: підготовка та завантаження даних до HDFS – 10–15 хв; виконання сценарію WordCount з варіюванням числа воркерів (демонстрація ефекту масштабування 1→3) – 15–20 хв; блок Spark SQL (групування, агрегування, запис у Parquet) – 20–25 хв; колективний аналіз у Spark

 ISSN: 2786-9458

UI/YARN RM UI (стадії, завдання, обсяг shuffle) – 10–15 хв. Така структура забезпечує відтворюваність та контрольований хронометраж.

Рекомендовано використовувати колонковий формат Parquet із увімкненою компресією, контролювати кількість партицій під час запису результатів, а також забезпечувати локальність даних за рахунок попереднього розподілу по DataNode. Це зменшує навантаження на мережу та обсяг shuffle, стабілізуючи час відгуку.

Перед початком заняття варто виконувати коротку програму smoke-тестів: перевірка станів служб (NameNode/DataNode, ResourceManager/NodeManager), тестовий запуск еталонного job'а, верифікація наявного дискового простору в HDFS. Така процедура підвищує операційну готовність і знижує ризик зривів.

Доцільні стислі чек-листи зі структурою навчального проєкту, базовими командами spark-submit, правилами доступу до вхідних/вихідних каталогів та «шпаргалка» типових помилок YARN (на кшталт insufficient memory, executor lost, permission denied) з короткими сценаріями відновлення.

Рекомендується фіксувати час виконання базового кейсу та збирати короткі рефлексії студентів (2–3 речення) щодо причинно-наслідкових зв'язків «конфігурація → план виконання → час». Це підвищує усвідомленість і формує дані для подальшого удосконалення курсу.

Висновки. Запропонована локальна інфраструктура Big Data на базі Proxmox (1 master + 1..3 workers; Hadoop 3.3.6; Spark 3.4.1) виявилася методично доцільною для проведення лабораторій з обробки даних і аналітики у закладах із обмеженими ресурсами. Кількісні експерименти засвідчили кероване горизонтальне масштабування з орієнтовним дворазовим скороченням часу виконання при переході від одного до трьох воркерів та водночас із низькою або такою, що зменшується, варіативністю (IQR). Такі характеристики роблять час відповіді передбачуваним і дозволяють у межах однієї пари планувати повний цикл ETL → SQL → збереження з подальшим аналізом стадій у Spark/YARN UI.

Педагогічні індикатори узгоджуються з технічними метриками: дві третини студентів завершують роботу в межах заняття, більшість не стикається з критичними збоями інсталяції, а медіанні оцінки зручності, стабільності, зрозумілості та сприйнятої продуктивності тримаються на рівні 4 за шкалою Лайкерта за мінімальної розбіжності оцінок. Це підтверджує, що стабільність і відтворюваність середовища безпосередньо транслюються у керованість навчального процесу.

Сформовано «мінімум відтворюваності» для освітніх кластерів: стандартизовані сценарії вимірювань, правило п’яти повторів з узагальненням медіаною та IQR, фіксація версій і конфігурацій, а також відкриті артефакти (шаблони, скрипти аналізу, інструктивні матеріали), придатні до оперативного впровадження в інших підрозділах. Практичні рекомендації охоплюють використання колонкових форматів даних (Parquet із компресією та партиціюванням), розвиток спостережуваності (UI-метрики, лічильники HDFS), а також базові DevOps-процедури (Ansible, smoke-тести). Доцільною виглядає дворівнева траєкторія курсу: базовий модуль – локально для стабільності й відтворюваності; поглиблений – з контрольованим «виходом у хмару» та фіксованим бюджетом для формування переносимості компетенцій.

Водночас результати отримано на малих конфігураціях і синтетичних наборах даних, що окреслює межі узагальнюваності. Подальша робота передбачає тести на більших кластерах, введення відкритих реальних датасетів, сценарії ін’єкції відмов для оцінки відмовостійкості та контрольоване порівняння з керованими хмарними сервісами з урахуванням повної вартості володіння й вимог безпеки. Загалом, локальна інфраструктура Big Data на Proxmox демонструє життєздатний баланс між технічною реалістичністю, стабільністю проведення занять і відтворюваністю науково-методичних результатів.

ISSN: 2786-9458

Список використаних джерел

1. Сіциліцин Ю. О., Лубко Д. В. Впровадження віртуалізованого середовища для лабораторних робіт з паралельного програмування. *Педагогічна Академія: наукові записки*. 2025. №17. URL: https://doi.org/10.5281/zenodo.15269510

2. Deb D., et al. Integrating Big Data and Cloud Computing Topics into the Computing Curricula: A Modular Approach. *Journal of Parallel and Distributed Computing.* 2021. Vol. 157. URL: https://doi.org/10.1016/j.jpdc.2021.07.012

3. Villegas-Ch. W., et al. Analysis of the State of Learning in University Students with the Use of a Hadoop Framework. *Future Internet.* 2021. Vol. 13(6). URL: https://doi.org/10.3390/fi13060140

4. Gkikas M. C., et al. A Risk Management Case Study Using Apache Spark. *Applied Sciences*. 2024. Vol. 14(22). URL: https://doi.org/10.3390/app142210112

5. Oleksiuk V., Oleksiuk O. The practice of developing the academic cloud using the Proxmox VE platform. *Educational Technology Quarterly*. 2021. Vol. 2021(4), p.p. 605–616. URL: https://doi.org/10.55056/etq.36

6. Baun C., Kappes M., Cocos H.-N., Koch M., Petrozziello M. The Virtual Computer Networks Lab: On the Design and Implementation of a Location Independent Networks Laboratory in Higher Education. *In Proceedings of the 17th International Conference on Computer Supported Education.* 2025. Vol. 1. p.p. 199-207. URL: https://doi.org/10.5220/0013199400003932

7. Yerra S. Reducing ETL processing time with SSIS optimizations for large-scale data pipelines. *International journal of data science and machine learning.* 2025. Vol. 05(1). p.p. 61–69. URL https://doi.org/10.1186/s12894-022-01059-8

8. Antolínez García, A. Hands-On Guide to Apache Spark 3: Build Scalable Computing Engines for Batch and Stream Data Processing. Apress, 2024. 403 p. URL: https://doi.org/10.1007/978-1-4842-9380-5

9. Ismail A., et al. Data science technology course: The design, assessment and development of the curriculum. *Education and Information Technologies*. 2023. Vol. 28. p.p. 1879–1901. URL: https://doi.org/10.1007/s10639-022-11558-8

10. Gutierrez A. C. Q., et al. Reproducibility and Scientific Integrity of Big Data Research in Urban Public Health and Digital Epidemiology: A Call to Action. *International Journal of Environmental Research and Public Health*. 2023. Vol. 20(2). URL: https://doi.org/10.3390/ijerph20021473

11. Di Rocco L., Ferraro Petrillo U., Palini F. Using software visualization to support the teaching of distributed programming. *The Journal of Supercomputing*. 2023. Vol. 79. p.p. 3974–3998. URL: https://doi.org/10.1007/s11227-022-04805-9

12. El-Sayed Ali H., Alham M. H., Ibrahim D. K. Big data resolving using Apache Spark for load forecasting and demand response in smart grid: a case study of Low Carbon London Project. *Journal of Big Data 11*. 2024. Vol. 59. URL: https://doi.org/10.1186/s40537-024-00909-6

13. International Brain Laboratory. Standardized and reproducible measurement of decision-making in mice. *eLife 10:e63711*. 2021. URL: https://doi.org/10.7554/eLife.63711

14. Laurinavichyute, A. K., Yadav H., Vasishth S. Share the code, not just the data: A case study of the reproducibility of articles published in the Journal of Memory and Language under the open data policy. *Journal of Memory and Language*. 2022. Vol. 125. URL: https://doi.org/10.1016/j.jml.2022.104332

15. Samuel S., Mietchen D. Computational reproducibility of Jupyter notebooks from biomedical publications. *GigaScience*. 2024. Vol. 13(5). URL: https://doi.org/10.1093/gigascience/giad113