

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ОДЕСЬКА ПОЛІТЕХНІКА»
МІНІСТЕРСТВА ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ОДЕСЬКА ПОЛІТЕХНІКА»
МІНІСТЕРСТВА ОСВІТИ І НАУКИ УКРАЇНИ

Кваліфікаційна наукова
праця на правах рукопису

ГРИБНЯК СЕРГІЙ СЕРГІЙОВИЧ

УДК 004.75

ДИСЕРТАЦІЯ

**МОДЕЛІ ТА МЕТОДИ ПІДВИЩЕННЯ ЕФЕКТИВНОСТІ ОБРОБКИ
ТРАНЗАКЦІЙ У РОЗПОДІЛЕНИХ РЕЄСТРАХ**

Спеціальність 122 – Комп'ютерні науки
Галузь знань – Інформаційні технології

Подається на здобуття наукового ступеня доктора філософії

Дисертація містить результати власних досліджень. Використання ідей,
результатів та текстів інших авторів мають посилання на відповідне джерело.

_____ С.С. Грибняк

Науковий керівник Дмитришин Дмитро Володимирович
доктор технічних наук, професор

Одеса – 2023

АНОТАЦІЯ

Грибняк С.С. Моделі та методи підвищення ефективності обробки транзакцій у розподілених реєстрах. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії за спеціальністю 122 – Комп'ютерні науки. – Національний університет «Одеська політехніка», МОН України, Одеса, 2023.

У **вступі** показано актуальність та особливості застосування технології розподілених реєстрів (Distributed Ledger Technology (DLT)) на основі блокчейну, яка стає все більш популярною завдяки прозорим транзакціям та відсутністю посередників або центральних керуючих органів.

Але незважаючи на потребу в застосуванні та на чисельні дослідження в галузі DLT, їх широкому розповсюдженню заважає недостатня ефективність існуючих рішень. Тому підвищення ефективності застосовуваних рішень є актуальною науково-технічною задачею, якій присвячено дане дослідження.

Визначено об'єкт, предмет, задачі та методи дослідження; наведено наукову новизну та практичне значення отриманих результатів; висвітлено особистий внесок здобувача.

В **першому розділі** дисертаційної роботи проведено аналіз існуючих технологій DLT, встановлені їх переваги та недоліки. Показано, що відомі системи, побудовані на DLT, мають ряд серйозних недоліків. Розподілена система повинна мати механізм масштабування для адаптації до зміни робочого навантаження у дуже широких межах. У той же час швидкість обробки даних у відомих популярних системах Bitcoin та Ethereum невисока – ці системи обробляють приблизно 7 та 15 транзакцій за секунду (tps) перша та до 119 tps друга. Ці показники незрівнянні з традиційними централізованими системами, що обробляють тисячі транзакцій на секунду.

Тобто виникає актуальне науково-практичне завдання підвищення

ефективності обробки транзакцій в розподілених реєстрах, що передбачає поєднання високої швидкості обробки транзакцій, високої масштабованості системи та достатньої криптозахищеності від маніпуляцій та атак.

Для вирішення цього завдання запропоновано шляхи підвищення ефективності обробки транзакцій та масштабованості розподіленого реєстру на основі двошарової моделі розподіленого реєстру.

У другому розділі дисертаційної роботи досліджено шляхи підвищення ефективності обробки транзакцій та масштабованості розподіленого реєстру.

Для прискорення обробки транзакцій з високим рівнем масштабованості запропоновано двошарову модель системи (рис. 1).

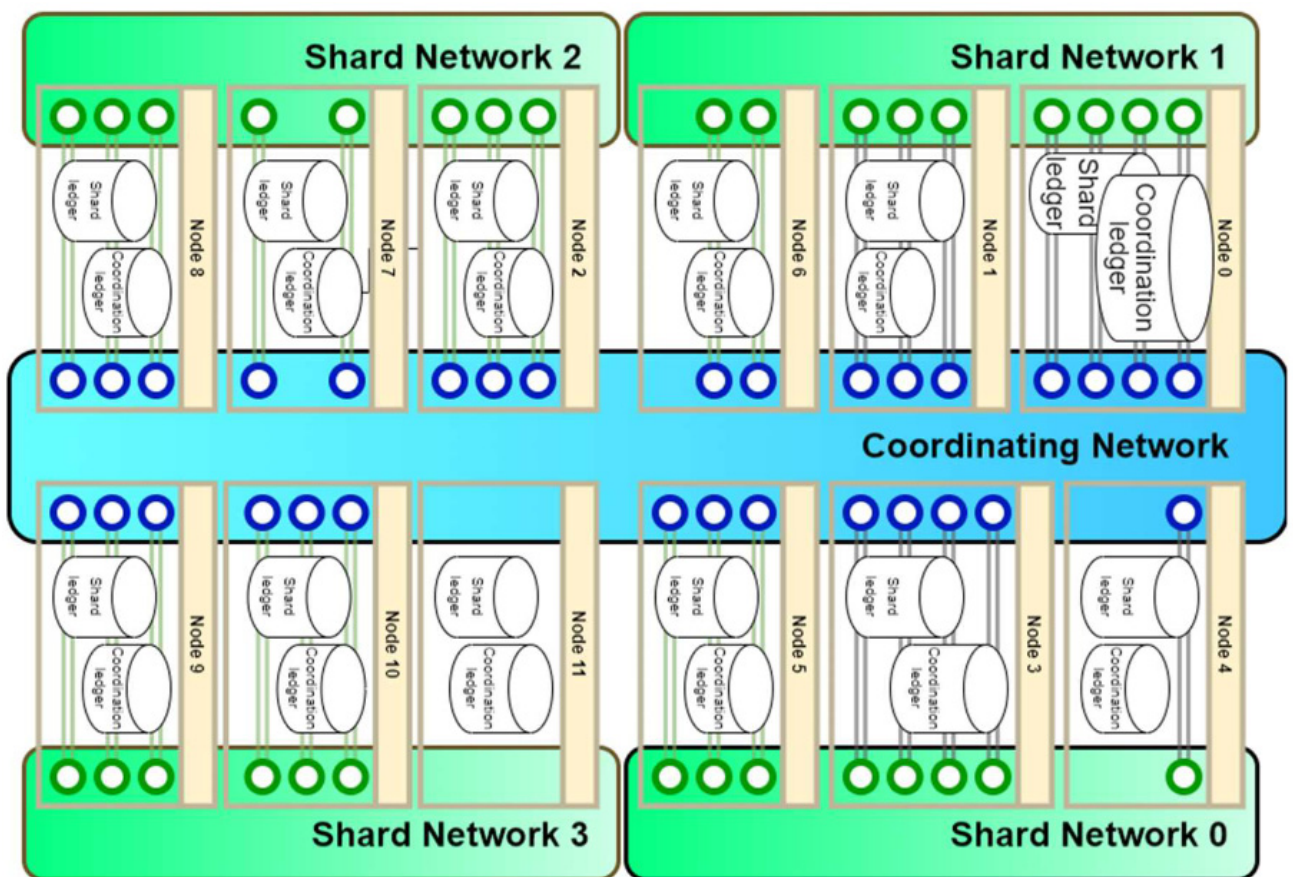


Рис. 1. Двошарова модель системи розподіленого реєстру

Основна мережа, яка призначена для формування та розповсюдження блоків, виконана за технологією BlockDAG (шар 1). Операції, пов'язані з верифікацією блоків на підставі протоколу консенсусу, винесені до координаційної мережі, що працює за технологією традиційного консенсусу

(шар 2). Згідно із запропонованою моделлю всі вузли, що входять до системи і називаються Worker'ами, отримують дві ролі – Утворювачів блоків в основній мережі та Координаторів у координаційній мережі. Таким чином розділені функції шарів мережі. В основній мережі виконуються функції формування блоків та їх поширення, що потребують операцій із об'ємними блоками. Побудова основної мережі за технологією BlockDAG суттєво прискорює процес попередньої обробки транзакцій за рахунок синхронного формування блоків. У координаційну мережу винесено операції з формування консенсусу відповідно до протоколу Proof of Stake (PoS), що вимагають великої кількості пересилань відносно коротких повідомлень. Тому робота координаційної мережі організована відповідно до традиційного блокчейн-протоколу, який виконується простіше, ніж BlockDAG.

Сформульовано перший пункт наукової новизни – вперше розроблено двошарову модель розподіленого реєстру, що поєднує в собі основну мережу, побудовану за технологією BlockDAG, та координаційну мережу, що базується на традиційній блокчейн технології, що дозволило підвищити швидкість обробки транзакцій покращило масштабованість системи.

Введено поняття хребетного блоку слота. Хребетний блок – це блок слота, якому притаманні наступні властивості:

- 1) має найбільшу висоту,
- 2) якщо таких блоків кілька, то береться блок, який посилається на більшу кількість попередніх блоків,
- 3) якщо таких блоків кілька, то береться той, у якого значення функції перемішування від хешу менше.

Оскільки кожен блок зберігає дерево Меркла, фактично хребетний блок зберігає в «упакованому» вигляді все «минуле» блоків даного слота. На цій підставі побудовано модель скороченого обміну інформацією між основною мережею та координаційною мережею, що зменшує інформаційне навантаження на систему в цілому та підвищує швидкість обробки транзакцій. Розроблено методику застосування обміну інформацією BlockDAG мережі з

координаційною мережею на підставі моделі хребетних блоків.

Сформульовано другий пункт наукової новизни – набула подальшого розвитку модель скороченого обміну інформацією між BlockDAG мережею та координаційною мережею на основі поняття хребетного блоку, що дозволило скоротити обсяг трафіку між мережею BlockDAG та координаційною мережею і тим самим зменшити навантаження на систему загалом та підвищити швидкість обробки транзакцій.

На підставі двох запропонованих моделей сформульовано узгоджену послідовність виконання операцій в основній та координаційній мережах.

Проведено дослідження шляхом статистичного моделювання можливостей скороченої процедури ймовірнісного консенсусу у двошаровій моделі. Встановлено, що якщо початкова частина ланцюжка хребетних блоків повторилася k разів і за ці блоки проголосували більше половини атестаторів – учасників відповідних слотів, то цей ланцюжок є кандидатом на фіналізацію. Показано, що при $k=3$ дане впорядкування не зміниться з ймовірністю щонайменше 0,9. В результаті прогноз можливої фіналізації може бути отриманий за $3 \cdot 4 = 12$ секунд, в той час як очікування повного консенсусу становить 4 хвилини 16 секунд.

Сформульовано третій пункт наукової новизни – вперше розроблений метод прискореного PoS консенсусу в двошаровій моделі розподіленого реєстру на основі k -кратної перевірки повторення початкової частини ланцюжка хребетних блоків і голосування за ці блоки більше половини учасників відповідних слотів, що дозволило зменшити час отримання рішення до декількох секунд із ймовірністю, не меншою 0,9.

У **третьому розділі** дисертаційної роботи показано, що при формуванні PoS консенсусу квазівипадковим чином у кожному слоті епохи відбувається формування складу комітетів та ролей координаторів у них (утворювач блоків, атестатор, агрегатор). Далі атестатор підписує останній, на його думку, правильний блок у слоті, а агрегатор поєднує результати атестації в один мультіпідписом. При цьому не виключені атаки маніпуляцією як на

формування складу комітетів, так і на вибір зі списку координаторів атестатора або агрегатора. Тому важливим засобом захисту від таких можливих маніпуляцій є розробка методу квазівипадкового перемішування довільної числової множини, що реально гарантує захищеність від подібних маніпуляцій. Розділ присвячений дослідженню можливості застосування у комп'ютерній криптографії найпростіших положень теорії нелінійних дискретних динамічних систем, основною характеристикою яких є хаотичність.

Досліджено можливість побудови якісних функцій перемішування на основі суперпозиції найпростіших лінійних та нелінійних відображень. Для лінійного відображення пропонується використовувати оператор перестановки, а для нелінійного – найбільш просте відображення, класичний q-Tent:

$$f_q(x) = 2(q/2 - |x - q/2|) = \begin{cases} 2x, x \leq q/2 \\ 2(q - x), x > q/2 \end{cases}, [0, q] \xrightarrow{f_q} [0, q].$$

Проведено кореляційний аналіз побудованих відображень, аналіз їх чутливості, аналіз середніх довжин циклів цих відображень, розглянуто можливості узагальнення алгоритмів побудови функцій перемішування та застосування їх до захисту від маніпуляцій при квазівипадковому виборі. Підтверджено очікувані хороші дифузійні властивості цих функцій перемішування. Досліджена функція перемішування має наступні переваги: відсутність колізій; простота застосовуваних математичних операцій; злом її можливий тільки шляхом повного перебору.

Розроблено метод перемішування даних за допомогою бієктивного відображення, яке є суперпозицією лінійного та найпростішого нелінійного відображення q-Tent. Запропоновані функції перемішуванні дозволили здійснити процес якісного перемішування при формуванні PoS консенсусу, і тим самим захистити процес обробки транзакцій у розподілених реєстрах від можливих маніпуляцій шляхом суттєвого підвищення криптостійкості.

Сформульовано четвертий пункт наукової новизни: отримав подальший розвиток метод перемішування даних за допомогою бієктивного відображення, яке є суперпозицією лінійного та найпростішого нелінійного відображення q-

Tent, що дозволило здійснити процес якісного перемішування при формуванні PoS консенсусу.

У четвертому розділі дисертаційної роботи розроблено експериментальну децентралізовану платформу Waterfall.

Розроблена платформа Waterfall успадковує основні принципи Ethereum 2.0 та розвиває їх. Завдяки розробленим моделям та методам платформа збільшує такі показники ефективності:

1. Висока швидкість обробки транзакцій при достатньому рівні масштабованості – масштабовані в системі блокові структури на основі DAG дозволяють одночасно публікувати кілька блоків. Це формує DAG та забезпечує завершеність всіх транзакцій за умови, що блоки не конфліктують один з одним. Тому Waterfall може одночасно обробляти об'єм транзакцій порівняний з тим що оброблюють Visa, MasterCard та UnionPay, але на децентралізованому рівні навіть у години пік. Згідно з останніми проміжними лабораторними тестами, система могла обробляти 3600 транзакцій в секунду. Для порівняння зазначимо, що Visa обробляє близько 1700 транзакцій на секунду.

2. Забезпечується висока захищеність платформи від можливих маніпуляцій при використанні операцій квазівипадкового вибору при здійсненні процедури консенсусу PoS за рахунок застосування розробленого методу перемішування.

Крім того, платформа Waterfall забезпечує наступні переваги, порівняно з аналогічними технічними рішеннями:

1. Низькі комісії за транзакції – архітектура спроектована таким чином, щоб підтримувати мінімальні комісії у різних сценаріях. Протокол динамічно масштабується зі зростанням навантаження на мережу. У той час, як продуктивність всієї системи збільшується, в тому самому слоті одночасно публікується більше блоків, а транзакційні збори знижуються в міру масштабування системи. Це знижує кількість операцій у пулі транзакцій навіть у години пік.

2. Низький фінансовий поріг входу – вузол із 6 Worker'ів коштує приблизно 1 920 доларів США.

3. Платформа обслуговує вбудовані токени – випуск та обслуговування tokenів (включно з NFT) не потребують спеціальних смарт-контрактів, а виконуються за допомогою звичайних транзакцій, що значно знижує накладні витрати. Крім того, це робить їх використання більш доступним для широкого кола користувачів.

4. Динамічні налаштування – платформа має механізми динамічної адаптації параметрів системи залежно від ситуації, що змінюється. зокрема, час слота, оптимальна кількість Worker'ів та деякі інші параметри налаштовуються автоматично.

Таким чином, платформа Waterfall забезпечує сприятливе середовище для надання та споживання широкого спектру послуг із ведення бізнесу та соціальної діяльності у зручному форматі загальнодоступного децентралізованого реєстру. Платформа призначена для обслуговування транзакцій з різними токенами, включаючи NFT, обслуговування смарт-контрактів і розробки розподілених додатків. Тестування системи показало, що вона може обробляти в середньому 2234 транзакції за секунду при досить високому рівні масштабованості.

Таким чином, підтверджено спроможність та правильність технічних рішень, заснованих на результатах дисертаційного дослідження.

Розроблені в роботі методи та інструментальні засоби отримали впровадження у діяльності Комунального підприємства «Теплопостачання міста Одеси» та знайшли відображення у навчальному процесі та науковій діяльності Національного університету «Одеська політехніка».

Ключові слова: розподілені системи, бази даних, технологія децентралізованих реєстрів, методи обробки інформації, операції з транзакціями, криптозахищеність, масштабованість, блокчейн-технології.

СПИСОК ОПУБЛІКОВАНИХ ПРАЦЬ ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Наукові праці, в яких опубліковані основні наукові результати дисертації:

1. Grybniak S., Dmitrishin D. Basic principles for constructing mixing functions based on the simplest linear and nonlinear mappings. *Proceedings of Odessa Polytechnic University*. 2022. № 2(66). P. 100–109. DOI: 10.15276/opu.2.66.2022.12. (Реєстр наукових фахових видань України, категорія «Б»)

2. Mazurok I., Leonchyk Y., Grybniak S., Nashyvan O., Masalskyi R. An incentives system for decentralized DAG-based platforms. *Applied Aspects of Information Technology*. 2022. Vol. 5, № 3. P. 196–207. DOI: <https://doi.org/10.15276/aait.05.2022.13>. (Реєстр наукових фахових видань України, категорія «Б»)

3. Mazurok I., Leonchyk Y., Grybniak S., Vorokhta A., Nashyvan O. Multi-objective optimization of committee selection for hierarchical byzantine fault tolerance-based consensus protocols. *Herald of Advanced Information Technology*. 2023. Vol. 6, № 1. P. 39–53. DOI: <https://doi.org/10.15276/hait.06.2023.3>. (Реєстр наукових фахових видань України, категорія «Б»)

4. Грибняк С. С. Двошарова модель масштабованого розподіленого децентралізованого реєстру. *Наукові праці Вінницького національного технічного університету*. 2023. № 2. С. 120–127. (Реєстр наукових фахових видань України, категорія «Б»)

5. Грибняк С. С. Розробка платформи децентралізованого реєстру з покращеними характеристиками. *Інформатика та математичні методи в моделюванні*. 2023. Т. 13, № 1-2, С. 48–55. DOI: 10.15276/imms.v13.no1-2.48. (Реєстр наукових фахових видань України, категорія «Б»).

Наукові праці, які засвідчують апробацію матеріалів дисертації:

6. Decentralized platforms: Goals, challenges, and solutions / S. Grybniak, Y. Leonchyk, R. Masalskyi, I. Mazurok, O. Nashyvan, R. Shanin. *2022 IEEE 7th Forum*

on Research and Technologies for Society and Industry Innovation (RTSI).24-26 August 2022. Paris, France. 2022. P. 62–67. DOI: <https://doi.org/10.1109/RTSI55261.2022.9905225>. Decentralized platforms express.pdf. (Indexed in SCOPUS)

7. Waterfall: A Scalable Distributed Ledger Technology / S. Grybniak, D. Dmytryshyn, Y. Leonchyk, I. Mazurok, O. Nashyvan, R. Shanin. In 2022 IEEE 1st Global Emerging Technology Blockchain Forum: *Blockchain & Beyond(iGETblockchain)*. November 2022. Irvine, CA, USA. 2022. P. 1–6. DOI: <https://doi.org/10.1109/iGETblockchain56591.2022.10087112>. Scalable Distributed Ledger express.pdf. (Indexed in SCOPUS)

8. Subnetworks in BlockDAG / O. Antonenko, S. Grybniak, D. Guzey, O. Nashyvan and R. Shanin. 2022 IEEE 1st Global Emerging Technology *Blockchain Forum: Blockchain& Beyond (iGETblockchain)*. November 2022. Irvine, CA, USA. 2022. P. 1–6. DOI: <https://doi.org/10.1109/iGETblockchain56591.2022.10087101>. Subnet works in BlockDAG (Final Version).pdf

9. Waterfall: Salto Collazo. Tokenomics / S. Grybniak, Y. Leonchyk, R. Masalskyi, I. Mazurok O. Nashyvan. 2022 IEEE International Conference on Blockchain, *Smart Healthcare and Emerging Technologies (SmartBlock4Health)*. 24-25 October 2022. Bucharest, Romania. 2022. P. 1-6. DOI: <https://doi.org/10.1109/SmartBlock4Health56071.2022.10034521>. Waterfall-Salto-Collazo express.pdf. (Indexed in SCOPUS)

10. Recurring Payments on EVM-based Platforms / S. Grybniak, N. Goga, O. Nashyvan, R. Mihai, I. Mazurok, Y. Leonchyk, G. Datta, O. F. Ozkul, C. V. Marian. 2022 IEEE 1st Global Emerging Technology Blockchain Forum: *Blockchain& Beyond (iGETblockchain)*. 07-11 November 2022. Irvine, CA, USA. 2022. P. 1–6. DOI: <https://doi.org/10.1109/iGETblockchain56591.2022.10087077>. Recurring Payments express.pdf. (Indexed in SCOPUS)

11. Blockchain-Enabled Economic Transactions: Recurring Financial Accruals and Payments / R. Mihai, O.F. Ozkul, G. Datta, S. Grybniak, C.V. Marian. 2022 IEEE 1st Global Emerging Technology Blockchain Forum: *Blockchain and Beyond*,

(*iGETblockchain*). 07-11 November 2022. Irvine, CA, USA. 2022.DOI: 10.1109/iGETblockchain56591.2022.10087074. (Indexed in SCOPUS)

12. Waterfall: Gozalandia. Distributed Protocol with Fast Finality and Proven Safety and Liveness / S. Grybniak, Y. Leonchyk, I. Mazurok, O. Nashyvan, R. Shanin. *IET Blockchain*. 2023. P. 1–12. DOI: <https://doi.org/10.1049/blc2.12023>.

13. Simulation Modelling of the Consensus Based on the Gozalandia / A. Vorokhta, I. Mazurok, Y. Leonchyk, S. Grybniak, and Y. Strakhov. *Adaptive Learning Management Technologies*. Kyiv. 2022. P. 31–33.

14. An Incentive System for Decentralized DAG-based Platforms / S. Grybniak, Y. Leonchyk, R. Masalskyi, I. Mazurok, O. Nashyvan. *IEEE Blockchain, Tech Brief*. 2022. URL: <https://blockchain.ieee.org/images/files/pdf/techbriefs-2022-q4/an-incentive-system-for-decentralized-dag-based-platforms.pdf>.

ABSTRACT

Grybniak S.S. Models and Methods for Improving Transaction Processing Efficiency in Distributed Ledgers. – Qualification scientific work in the form of manuscript.

Thesis for the Degree of Doctor of Philosophy in Computer Science, Specialization 122 – Computer Science. – Odessa Polytechnic National University, Ministry of Education and Science of Ukraine, Odesa, 2023.

The **introduction** shows the relevance and characteristics of Distributed Ledger Technology (DLT) based on blockchain. This technology is becoming increasingly popular due to its secure and transparent transactions, as well as its elimination of intermediaries and central governing authorities. Despite the growing demand for applications and research efforts in the field of DLT, its widespread adoption is hindered by the low efficiency of existing solutions. Therefore, enhancing the efficiency of applied solutions remains a proper scientific and technical challenge, to which this research is dedicated. The object, subject, tasks, and methods of the research have been defined. The scientific novelty and practical significance of the obtained results have been presented. The researcher's contribution has also been highlighted.

In the **first section** of the dissertation, an analysis of existing DLT technologies is conducted, identifying their advantages and disadvantages. It is demonstrated that well-known systems built on DLT suffer from several significant drawbacks. A distributed system should have a scalability mechanism to adapt to changing workloads within wide boundaries. However, the data processing speed in popular systems like Bitcoin and Ethereum is slow, processing approximately 7 and 15 transactions per second (tps) first one and up to 119 tps the second one, respectively. These figures are incomparable to traditional centralized systems, which can handle thousands of transactions per second.

There exist scientific and practical challenges to enhancing the efficiency of transaction processing in distributed ledgers, which requires combining high

transaction processing speeds, system scalability, and sufficient crypto security to prevent manipulations and attacks. The proposed ways to improve the efficiency of transaction processing and scalability of the distributed ledger are based on a two-layer model of the distributed ledger.

In the **second section** of the dissertation work, ways to improve the efficiency of transaction processing and scalability of the distributed ledger are investigated.

To accelerate transaction processing with high scalability, a two-layer model for the system is proposed (Fig. 1).

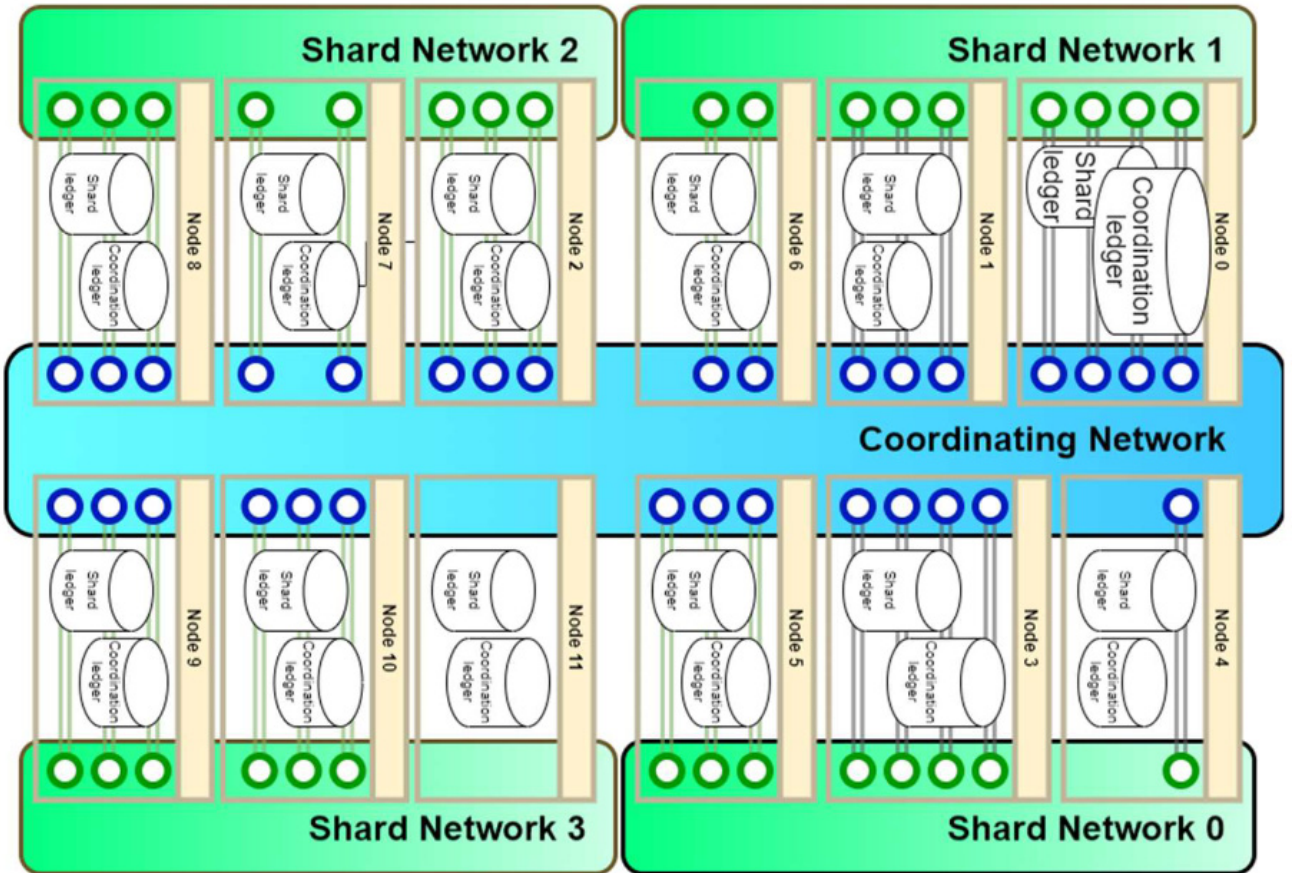


Fig. 1. Two-layer model of a distributed ledger

The main network is responsible for block formation and propagation, implemented using the BlockDAG technology (Layer 1). Operations related to block verification based on the consensus protocol are delegated to the coordination network, which operates using traditional consensus technology (Layer 2). According to the proposed model, all nodes in the system, called Workers, take on two roles – Block Producers in the main network and Coordinators in the coordination network.

With this approach, the functions of the network layers are separated. In the main network, block formation and propagation functions are executed, which require operations involving long blocks. Building the main network using BlockDAG technology significantly accelerates the process of pre-processing transactions due to asynchronous block formation. In the coordination network, operations related to forming consensus according to the Proof of Stake protocol are performed, which demand a large number of short message transmissions. Therefore, the operation of the coordination network is organized according to a traditional blockchain protocol, which is simpler than BlockDAG.

The first point of scientific novelty is formulated as follows: for the first time a two-layer model of a distributed ledger has been developed, which combines a main network built using BlockDAG technology and a coordination network based on traditional blockchain technology. This allows for an increased transaction processing speed while significantly scaling the system.

The concept of a “slot skeletal block” is introduced. A skeletal block is a slot block with the following properties:

- 1) It has the highest height,
- 2) If there are multiple such blocks, the one that references a larger number of previous blocks is chosen,
- 3) If there are multiple such blocks, the one with a lower hash value resulting from the mixing function is selected.

Since each block stores a Merkle tree, the skeletal block in fact stores in a “compressed” form all the “past” blocks of the same slot. Based on this, a model of efficient information exchange between the main network and the coordination network is built, reducing overall information overhead on the system and increasing transaction processing speed. A methodology for applying BlockDAG network information exchange with the coordination network is developed, based on the skeletal block model.

The second point of scientific novelty is formulated as follows: the model of efficient information exchange between the BlockDAG network and the coordination

network is further developed based on the concept of a skeletal block. This development leads to a reduction in the volume of traffic between the BlockDAG network and the coordination network, thereby decreasing the overall system load and increasing transaction processing speed.

Based on the two proposed models, a coordinated sequence of operations has been formulated in the main and coordination networks.

The research was conducted through statistical modeling of the possibilities of a shortened procedure for probabilistic consensus in a two-layer model. It has been established that if the initial part of the chain of skeletal blocks is repeated k times, and if more than half the attesters – participants of the corresponding slots – vote for these blocks, then this chain becomes a candidate for finalization. It has been shown that when $k = 3$, this ordering will not change with a probability of at least 0.9. As a result, a forecast of possible finalization can be obtained in $3 \cdot 4 = 12$ seconds, while the expected time for full consensus is 4 minutes and 16 seconds.

The third point of scientific novelty is formulated as follows: the first was developed as a method of accelerated PoS consensus in a two-layer model of a distributed ledger, based on k -fold verification of the initial part of the chain of skeletal blocks, and voting by more than half of the participants in the corresponding slots for those blocks. This allows for the time to obtain a decision to be reduced to just a few seconds, with a probability of at least 0.9.

In the **third section** of the dissertation, it is shown that in the formation of the PoS consensus, the composition of committees and coordinator roles (block proposer, attester, aggregator) is quasi-randomly determined in each epoch slot. Next, the attester signs the last valid (according to their judgment) block in the aggregator slot, combining the attestation results with a single signature. However, attacks on both the formation of committee compositions and the selection of coordinators as attesters and aggregators cannot be ruled out. Therefore, an important means of protection against such potential manipulations is the development of a method for quasi-random shuffling of an arbitrary numerical set, which can effectively guarantee protection against such manipulations. This chapter is dedicated to investigating the

possibility of applying the simplest principles of nonlinear discrete dynamic systems theory, characterized by chaos, in computer cryptography.

Research on the possibility of constructing qualitative mixing functions based on the superposition of the simplest linear and nonlinear transformations is investigated. As a linear transformation, the permutation operator is proposed to be used, and as a nonlinear transformation, the simplest classical q-Tent mapping is suggested:

$$f_q(x) = 2(q/2 - |x - q/2|) = \begin{cases} 2x, x \leq q/2 \\ 2(q - x), x > q/2 \end{cases}, \quad [0, q] \xrightarrow{f_q} [0, q].$$

Correlation analysis of the constructed mappings, sensitivity analysis, and analysis of the average cycle lengths of these mappings are conducted. The possibility of generalizing the algorithms for constructing mixing functions and their application for protection against manipulations in quasi-random selection is explored. The expected good diffusion properties of these mixing functions are confirmed. The investigated mixing function has the following advantages: absence of collisions; simplicity of the applied mathematical operations; its breaking is only possible through complete overrun.

A data mixing method using bijective mapping has been developed, which is a superposition of a linear and the simplest nonlinear mapping q-Tent. This method enables the process of qualitative mixing during the formation of PoS consensus, thereby protecting transaction processing in distributed ledgers from possible manipulations by significantly enhancing crypto security.

The fourth point of scientific novelty is formulated as follows: the data mixing method using a bijective mapping, which is a superposition of a linear and the simplest nonlinear q-Tent mapping, receives further development, enabling the process of qualitative shuffle during the formation of the PoS consensus.

In the **fourth section** of the dissertation, an experimental decentralized platform called "Waterfall" has been developed. The developed Waterfall platform inherits the main principles of Ethereum 2.0 and further expands them. Thanks to the developed models and methods, the platform increases the following efficiency indicators:

1. High transaction processing speed with sufficient scalability – Waterfall achieves scalability through block structures based on a directed acyclic graph (DAG). This allows multiple blocks to be published simultaneously, forming a DAG and ensuring the completion of all transactions, as long as the blocks do not conflict with each other. As a result, Waterfall can handle Visa, MasterCard, and Union Pay transactions simultaneously on a decentralized level, even during peak hours. According to recent intermediate laboratory tests, the system was able to process 3600 transactions per second. For comparison, Visa processes approximately 1700 transactions per second.

2. The platform ensures high security against potential manipulations by using operations of quasi-random selection during the Proof of Stake (PoS) consensus procedure, achieved through the application of the developed shuffling method.

In addition to the afore-mentioned advantages, the Waterfall platform provides the following benefits compared to similar technical solutions:

1. Low transaction fees – The architecture is designed to maintain minimal fees across various scenarios. The protocol dynamically scales with an increasing network load. As the overall system's performance improves, more blocks are simultaneously published in the same slot, leading to reduced transaction fees as the system scales. This reduces the number of operations in the transaction pool, even during peak hours.

2. Low financial entry barrier – A node with 6 Workers costs approximately 1,920 US dollars.

3. The platform serves built-in tokens – output and servicing tokens (including NFTs) do not require special smart contracts but are executed through regular transactions, significantly reducing overhead costs. Additionally, this makes their usage more accessible to a wide range of users.

4. Dynamic configuration – The platform incorporates mechanisms for dynamically adapting system parameters based on changing situations, such as the slot time, optimal number of Workers, and certain other parameters that are automatically adjusted as needed.

Thus, the Waterfall platform provides a favorable environment for the provision and consumption of a wide range of services in business and social activities through the convenient format of a publicly accessible decentralized ledger. The platform is designed to support transactions with various tokens, including NFTs, servicing smart contracts, and developing distributed applications. System testing has shown that Waterfall can process an average of 2.234 transactions per second at a high enough level of scalability.

As a result, the capability and correctness of the technical solutions based on the findings of the dissertation research have been confirmed.

The developed methods and tools in the research have been implemented in the activities of the Municipal enterprise “Heat supply of the city of Odesa” (Odesa). They have also been incorporated into the educational process and scientific activities of the National University "Odessa Polytechnic".

Key Words: distributed systems, databases, distributed ledger technology, information processing methods, transaction processing, crypto-security, scalability, blockchain technologies.

LIST OF PUBLISHED PAPERS ON THE TOPICS OF THE DISSERTATION

Papers in which the main scientific results of the dissertation are published:

1. Grybniak, S., & Dmitrishin, D. (2022). Basic principles for constructing mixing functions based on the simplest linear and nonlinear mappings. *Proceedings of Odessa Polytechnic University*, 2(66), 100–109. DOI: 10.15276/opu.2.66.2022.12. (Register of scientific and professional publications of Ukraine, category "B")
2. Mazurok, I., Leonchyk, Y., Grybniak, S., Nashyvan, O., & Masalskyi, R. (2022). An incentives system for decentralized DAG-based platforms. *Applied Aspects of Information Technology*, 5, 3, 196–207. DOI: <https://doi.org/10.15276/aait.05.2022.13>. (Register of scientific and professional publications of Ukraine, category "B")
3. Mazurok, I., Leonchyk, Y., Grybniak, S., Vorokhta, A., & Nashyvan, O. (2023). Multi-objective optimization of committee selection for hierarchical byzantine fault tolerance-based consensus protocols. *Herald of Advanced Information Technology*, 6, 1, 39–53. DOI: <https://doi.org/10.15276/hait.06.2023.3>. (Register of scientific and professional publications of Ukraine, category "B")
4. Grybniak, S.S. (2023). Two-layer model of scalable distributed decentralized registry. *Scientific works of Vinnytsia National Technical University*, 2, 120–127. (Register of scientific and professional publications of Ukraine, category "B")
5. Grybniak, S.S. (2023). Development of a decentralized registry platform with improved characteristics. *Informatics and Mathematical Methods in Simulation*, 13, 1-2, 48–55. DOI: 10.15276/imms.v13.no1-2.48. (Register of scientific and professional publications of Ukraine, category "B").

Papers certifying the approbation of the dissertation materials:

6. Grybniak, S., Leonchyk, Y., Masalskyi, R., Mazurok, I., Nashyvan, O., & Shanin, R. (2022). Decentralized platforms: Goals, challenges, and solutions. *2022 IEEE 7th Forum on Research and Technologies for Society and Industry Innovation*

(RTSI),24-26 August 2022, Paris, France, 62–67. DOI: <https://doi.org/10.1109/RTSI55261.2022.9905225>. Decentralized platforms express.pdf. (Indexed in SCOPUS)

7. Grybniak, S., Dmytryshyn, D., Leonchyk, Y., Mazurok, I., Nashyvan, O., & Shanin, R. (2022). Waterfall: A Scalable Distributed Ledger Technology. *In 2022 IEEE 1st Global Emerging Technology Blockchain Forum: Blockchain & Beyond (iGETblockchain)*, November 2022, Irvine, CA, USA, 1–6. DOI: <https://doi.org/10.1109/iGETblockchain56591.2022.10087112>. Scalable Distributed Ledger express.pdf. (Indexed in SCOPUS)

8. Antonenko, O., Grybniak, S., Guzey, D., Nashyvan, O., & Shanin, R. (2022). Subnetworks in BlockDAG. *2022 IEEE 1st Global Emerging Technology Blockchain Forum: Blockchain & Beyond (iGETblockchain)*, November 2022, Irvine, CA, USA, 1–6. DOI: <https://doi.org/10.1109/iGETblockchain56591.2022.10087101>. Subnet works in BlockDAG (Final Version).pdf

9. Grybniak, S., Leonchyk, Y., Masalskyi, R., Mazurok, I., & Nashyvan, O. (2022). Waterfall: Salto Collazo. Tokenomics. *2022 IEEE International Conference on Blockchain, Smart Healthcare and Emerging Technologies (SmartBlock4Health)*,24-25 October 2022, Bucharest, Romania, 1–6. DOI: <https://doi.org/10.1109/SmartBlock4Health56071.2022.10034521>. Waterfall-Salto-Collazo express.pdf. (Indexed in SCOPUS)

10. Grybniak, S., Goga, N., Nashyvan, O., Mihai, R., Mazurok, I., Leonchyk, Y., Datta, G., Ozkul, O. F., & Marian, C. V. (2022). Recurring Payments on EVM-based Platforms. *2022 IEEE 1st Global Emerging Technology Blockchain Forum: Blockchain & Beyond (iGETblockchain)*, 07-11 November 2022, Irvine, CA, USA, 1–6. DOI: <https://doi.org/10.1109/iGETblockchain56591.2022.10087077>. Recurring Payments express.pdf. (Indexed in SCOPUS)

11. Mihai, R., Ozkul, O.F., Datta, G., Grybniak, S., & Marian, C.V. (2022).Blockchain-Enabled Economic Transactions: Recurring Financial Accruals and Payments. *2022 IEEE 1st Global Emerging Technology Blockchain Forum: Blockchain and Beyond, (iGETblockchain)*, 07-11 November 2022, Irvine, CA, USA.

DOI: 10.1109/iGETblockchain56591.2022.10087074. (Indexed in SCOPUS)

12. Grybniak, S., Leonchyk, Y., Mazurok, I., Nashyvan, O., & Shanin, R. (2023). Waterfall: Gozalandia. Distributed Protocol with Fast Finality and Proven Safety and Liveness. *IET Blockchain*, 1–12. DOI: <https://doi.org/10.1049/blc2.12023>.

13. Vorokhta, A., Mazurok, I., Leonchyk, Y., Grybniak, S., & Strakhov, Y. (2022). Simulation Modelling of the Consensus Based on the Gozalandia. *Adaptive Learning Management Technologies*, Kyiv, 31–33.

14. Grybniak, S., Leonchyk, Y., Masalskyi, R., Mazurok, I., Nashyvan, O. (2022). An Incentive System for Decentralized DAG-based Platforms. *IEEE Blockchain, Tech Breif*. URL: <https://blockchain.ieee.org/images/files/pdf/techbriefs-2022-q4/an-incentive-system-for-decentralized-dag-based-platforms.pdf>.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ.....	25
ВСТУП.....	26
РОЗДІЛ 1 АНАЛІЗ ПРОБЛЕМИ ПІДВИЩЕННЯ ЕФЕКТИВНОСТІ ВИКОРИСТАННЯ ТЕХНОЛОГІЇ РОЗПОДІЛЕНИХ РЕЄСТРІВ	31
1.1 Формування технології розподілених реєстрів.....	31
1.2 Консенсус та роль технології розподілених реєстрів	34
1.2.1 Доказ роботи Proof of Work (PoW).....	34
1.2.2 Доказ частки володіння Proof of Stake (PoS).....	36
1.2.3 Делегований доказ частки Delegated Proof of Stake (DPoS)	37
1.2.4 Алгоритм практичної візантійської стійкості до відмови Practical Byzantine Fault Tolerance (PBFT)	38
1.3 Практичне застосування технології децентралізованих реєстрів	39
1.4 Покоління технології розподілених реєстрів	41
1.4.1 Bitcoin як перше покоління DLT	41
1.4.2 Ethereum 1.0 як друге покоління DLT.....	43
1.4.3 Ethereum 2.0 як третє покоління DLT	45
1.4.4 BlockDAG як четверте покоління Blockchain	49
1.5 Висновки до першого розділу.....	50
РОЗДІЛ 2 РОЗРОБКА ТА ДОСЛІДЖЕННЯ ШЛЯХІВ ПІДВИЩЕННЯ ШВИДКОСТІ ОБРОБКИ ТРАНЗАКЦІЙ І МАСШТАБОВАНOSTІ	52
2.1 Розробка двошарової моделі розподіленого реєстру	52
2.2 Розробка моделі скороченого обміну інформацією між основною мережею та координаційною мережею на базі поняття скелетних блоків.....	54
2.2.1 Формалізація поняття скелетного блоку	54
2.2.2 Методика застосування обміну інформацією на основі моделі скелетних блоків	58
2.3 Робота двошарової мережі на основі запропонованих моделей	58
2.3.1 Функціонування координаційної мережі.....	58

2.3.2 Функціонування BlockDAG мережі	61
2.4 Розробка методу ймовірнісного консенсусу	62
2.4.1 Недоліки повного консенсусу	62
2.4.2 Статистичне моделювання ймовірнісних процесів під час атестації	64
2.4.3 Метод ймовірнісного консенсусу	67
2.5 Дослідження шляхів підвищення стійкості протоколу PoS	68
2.6 Висновки до другого розділу	81
РОЗДІЛ 3 ДОСЛІДЖЕННЯ БАЗОВИХ ПРИНЦИПІВ ПОБУДОВИ	
ФУНКЦІЙ ПЕРЕМІШУВАННЯ НА ОСНОВІ НАЙПРОСТІШИХ	
ЛІНІЙНИХ І НЕЛІНІЙНИХ ВІДОБРАЖЕНЬ	
3.1 Аналіз потенційних вразливостей при виконанні протоколу	
Proof of Stake у координаційній мережі	84
3.2 Постановка задачі побудови функцій перемішування	85
3.3 Розробка методу перемішування	89
3.4 Аналіз показників якості функції перемішування	91
3.4.1 Кореляційний аналіз функції перемішування	91
3.4.2 Аналіз чутливості функції перемішування	92
3.4.3 Аналіз довжин періодів функції перемішування	95
3.5 Метод квазівипадкового перемішування	
довільної числової множини для захищеності	
від можливих маніпуляцій	98
3.6 Ілюстративний приклад із перемішуванням зображень	98
3.7 Висновки до третього розділу	101
РОЗДІЛ 4 РОЗРОБКА ТА ВИПРОБУВАННЯ ЕКСПЕРИМЕНТАЛЬНОЇ	
ДЕЦЕНТРАЛІЗОВАНОЇ ПЛАТФОРМИ WATERFALL НА ОСНОВІ	
ЗАПРОПОНОВАНИХ МОДЕЛЕЙ І МЕТОДІВ	
4.1 Загальні положення	103
4.2 Блок-схема програмного продукту	103
4.3 Схема вузлів та їх взаємодії	108
4.4 Основні структури даних	110

4.5 Реалізація та випробування платформи Waterfall.....	123
4.6 Висновки до четвертого розділу	125
ЗАГАЛЬНІ ВИСНОВКИ	127
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	129
ДОДАТОК А Список публікацій здобувача за темою дисертації	141
ДОДАТОК Б Довідка про використання результатів дисертаційної роботи у науково-дослідницькій діяльності Національного університету «Одеська політехніка»	144
ДОДАТОК В Довідка про впровадження результатів дисертаційної роботи в навчальний процес Національного університету «Одеська політехніка»	145
ДОДАТОК Д Довідка про впровадження результатів дисертаційної роботи у Комунальному підприємстві «Теплопостачання міста Одеси»	146

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

BFT – Byzantine Fault Tolerance, візантійська відмовостійкість

DAG – directed acyclic graph, спрямований ациклічний граф

DApps – Decentralized Applications, децентралізовані додатки

DeFi – Decentralized Finance, децентралізовані фінанси

DEX – Decentralized Exchanges, децентралізовані біржі

DLT – Distributed Ledger Technology, технологія розподілених реєстрів

DPoS – Delegated Proof of Stake, делегований доказ частки володіння

ETH 2.0 – Ethereum 2.0

EVM – Ethereum Virtual Machine, віртуальна машина Ethereum

IoT – Internet of Things, інтернет речей

NFT – non-fungible token, невзаємозамінний токен

P2P – peer-to-peer, однорангова (пірінгова) мережа–

PoW – Proof of Work , доказ роботи

PoS – Proof of Stake, доказ частки володіння

PBFT – Practical Byzantine Fault Tolerance, практична візантійська стійкість до відмови

ВСТУП

Актуальність теми зумовлена досягненнями останніх років в області технології розподілених реєстрів (Distributed Ledger Technology, DLT), яка є сьогодні однією з революційних технологій у розробці розподілених систем обробки даних. Першим і сьогодні класичним окремим випадком DLT стала технологія блокчейн, а також розроблена в подальшому на її основі криптовалюта Bitcoin. За останнє десятиліття коло застосування блокчейнів та інших DLT швидко розширюється. DLT стають все більш популярними завдяки прозорим транзакціям та відсутністю посередників або центральних керуючих органів. Технологія блокчейн може використовуватись практично у всіх сферах діяльності. Сюди належать насамперед різні фінансові послуги та платіжні системи, медицина, підтримка Інтернету речей (IoT) та багато інших галузей.

Але незважаючи на чисельні дослідження в галузі DLT, їх широкому розповсюдженню заважає недостатня ефективність існуючих рішень щодо швидкості обробки транзакцій при високих масштабуванні та криптозахищеності. Тому підвищення ефективності застосованих рішень є актуальною науково-технічною задачею, якій присвячено дане дослідження.

Метою дослідження є підвищення ефективності розподілених шляхом розробки моделей та методів обробки транзакцій у розподілених реєстрах.

Для досягнення мети дослідження необхідно вирішити наступні завдання:

- проаналізувати існуючі технології, встановити їх переваги та недоліки;
- розробити двошарову модель розподіленого реєстру, що поєднує в собі основну мережу, побудовану за технологією BlockDAG, та координаційну мережу, що базується на традиційній блокчейн технології;
- розробити модель прискореного впорядкування блоків DAG з урахуванням концепції хребетних блоків;
- удосконалити метод формування ймовірнісного Proof of Stake (PoS) консенсусу в рамках запропонованої двошарової моделі;
- розробити метод квазівипадкового перемішування довільної числової

множини для криптозахищеності від можливих маніпуляцій при формуванні PoS консенсусу;

– експериментально підтвердити підвищену ефективність розроблених моделей та методів при створенні розподіленого реєстру.

Об'єктом дослідження є процеси обробки транзакцій у розподілених реєстрах.

Предметом дослідження є моделі та методи підвищення швидкості обробки транзакцій при високих масштабованості та криптозахищеності.

Методи дослідження. Для вирішення поставлених задач використана теорія розподілених систем та методи дискретної математики при розробці двошарової моделі розподіленого реєстру та моделі прискореного впорядкування блоків DAG; теорія прийняття рішень та експертного оцінювання при розробці методу формування ймовірнісного консенсусу; теорія чисел та теорія ймовірності при розробці методу квазівипадкового перемішування довільної числової множини; методи математичного та імітаційного моделювання при апробації запропонованих рішень.

Наукова новизна отриманих результатів. Основний науковий результат полягає у розв'язанні важливої науково-практичної задачі підвищення ефективності розподілених систем шляхом розробки моделей та методів обробки транзакцій у розподілених реєстрах для широкої низки практичних застосувань.

У рамках виконаних досліджень отримано такі наукові результати:

–*вперше* розроблено двошарову модель розподіленого реєстру, що поєднує в собі основну мережу, побудовану за технологією BlockDAG, та координаційну мережу, що базується на традиційній блокчейн технології, що дозволило підвищити швидкість обробки транзакцій при суттєвому масштабуванні;

–*набула подальшого розвитку* модель скороченого обміну інформацією між BlockDAG мережею та координаційною мережею на основі поняття хребетного блоку, що дозволило скоротити обсяг трафіку між мережею

BlockDAG та координаційною мережею і тим самим зменшити навантаження на систему загалом та підвищити швидкість обробки транзакцій;

–вперше розроблений метод прискореного PoS консенсусу в двошаровій моделі розподіленого реєстру на основі k -кратної перевірки повторення початкової частини ланцюжка хребетних блоків і голосування за ці блоки більше половини учасників відповідних слотів, що дозволило зменшити час отримання рішення до декількох секунд із ймовірністю, не меншою 0,9;

–отримав подальший розвиток метод перемішування даних за допомогою бієктивного відображення, яке є суперпозицією лінійного та найпростішого нелінійного відображення q -Tent, що дозволило здійснити процес якісного перемішування при формуванні PoS консенсусу.

Практичне значення одержаних результатів. Розроблені в дисертаційній роботі моделі і методи були використані для обробки транзакцій, підвищення масштабованості та криптостійкості розподілених реєстрів, що дозволило підвищити ефективність розподілених систем для широкої низки практичних застосувань.

Розроблено методику визначення оптимального розміру комітету та розподілу Worker'ів залежно від їх загальної кількості в мережі та очікуваної помилкової частки.

Розроблено платформу Waterfall, що успадковує основні принципи Ethereum 2.0 та розвиває їх. Завдяки розробленим моделям та методам платформа має більшу ефективність за низкою показників, що дозволяє її рекомендувати до широкого кола прикладних завдань, де використовуються розподілені реєстри.

Розроблені в роботі методи та інструментальні засоби отримали впровадження у діяльності Комунального підприємства «Теплопостачання міста Одеси» та знайшли відображення у навчальному процесі та науковій діяльності Національного університету «Одеська політехніка» при виконанні проекту «Фундаментальні та поточні дослідження та розробки для Проекту Водоспад, технології розподілених обчислень з відкритим вихідним кодом»,

який розроблявся організацією Digital Clever Solutions Inc., США.

Особистий внесок здобувача. Основні наукові положення, висновки і рекомендації, що містяться в дисертації та виносяться на захист, отримано здобувачем особисто в період з 2022 по 2023 рік. Список публікацій здобувача за темою дисертації наведено в додатку А. У науковій праці [1], внесок здобувача полягає в удосконаленні методу перемішування даних за допомогою бієктивного відображення для якісного перемішування при формуванні PoS консенсусу, а також у виконанні числових експериментів; у науковій праці [2] – запропоновано використання BlockDAG для прискорення транзакцій; у науковій праці [3] – запропоновано основні принципи формування ймовірнісного консенсусу; у статті [4] – запропоновані двошарова модель розподіленого реєстру, модель використання хребетних блоків та її використання для прискорення обміну даними між основною і координаційною мережею та метод оптимістичного ймовірнісного консенсусу; в статті [5] – представлені отримані автором результати практичної реалізації розроблених моделей і методів у вигляді децентралізованої платформи Waterfall; в роботах [6 – 8] – розроблені принципи підвищення ефективності децентралізованих реєстрів на основі BlockDAG; в роботах [9 – 14] – запропоновані різні практичні застосування децентралізованих BlockDAG платформ.

Апробація результатів дисертації. Основні положення і практичні результати дисертаційної роботи доповідалися та одержали схвалення на таких конференціях: 7-й форум з IEEE досліджень і технологій для суспільства та промислових інновацій (Париж, Франція, 2022 рік), 1-й глобальний IEEE блокчейн-форум з нових технологій (м. Ірвін, США, 2022 рік), Міжнародна конференція IEEE з блокчейну, розумної охорони здоров'я та нових технологій (SmartBlock4Health) (м. Бухарест, Румунія, 2022 рік), Міжнародна конференція «Адаптивні технології управління навчанням ATL-2022» (Київ, 2022 рік).

Публікації. Основні результати дисертаційної роботи викладено в 14 публікаціях з них: 5 статей у наукових фахових виданнях України з технічних наук, 9 публікацій у зарубіжних наукових періодичних виданнях з напряду, з

якого підготовлено дисертацію, і працях і матеріалах наукових конференцій, 4 з яких проіндексовано наукометричною базою SCOPUS.

Структура і обсяг роботи. Дисертація складається з вступу, чотирьох розділів, висновків, списку використаних джерел і додатків. Повний обсяг дисертації складає 146 сторінку, в тому числі: 103 сторінок основного тексту, 39 рисунків і 3 таблиці, список використаних джерел з 115 найменувань і 4 додатків.

РОЗДІЛ 1

АНАЛІЗ ПРОБЛЕМИ ПІДВИЩЕННЯ ЕФЕКТИВНОСТІ ВИКОРИСТАННЯ
ТЕХНОЛОГІЇ РОЗПОДІЛЕНИХ РЕЄСТРІВ

1.1 Формування технології розподілених реєстрів

Технологія розподілених реєстрів (Distributed Ledger Technology , DLT) [1 – 4] (що має більше сьогодні популярну назву блокчейн-технологія) – це комплексна інформаційна технологія. Окремі компоненти цієї технології були сформовані раніше, але тільки спільне їх застосування дозволило отримати «вибуховий» синергетичний ефект. До цих технологій належать, перш за все:

1. Однорангові або пірінгові (P2P) мережі [5,6];
2. Ідея криптовалюти Девіда Чаума [8];
3. Дерево Меркла [10, 11].

Створено сотні різних програм обміну даними, необхідні користувачам по всьому світу. Однак зберігання їх на центральному сервері дуже обмежувало можливості їх передачі, адже пропускна спроможність одного сервера, як правило, невелика або обмежена. Це призвело до появи пірінгових або однорангових (peer-to-peer, P2P) мереж [5, 6], суть яких полягає в рівноправності учасників. У таких мережах відсутній виділений сервер і кожен рівноправний вузол – пир (peer) є як клієнтом, так і сервером. Отже, навіть якщо деякі комп'ютери/вузли вийдуть з ладу, мережа та зв'язок все одно працюватимуть. Обчислювальні ресурси збалансовано між учасниками мережі [7]. Станом на 2009 рік відеотрафік через пірінгові мережі становив одну третину всього трафіку інтернету.

Ідея криптовалюти з'явилася вже у 1980-х роках, коли її представив Девід Чаум [8]. Обновлюючись на принципах створення анонімної та безпечної системи цифрового голосування, Чаум розробив криптографічний метод, який називався сліпими підписами, який дозволяв використовувати псевдонім при обміні даними. Сліпі підписи використовували систему асиметричної

криптографії, в якій кожен користувач має загальновідомий відкритий ключ і секретний закритий ключ. Сліпі підписи дозволяли підписувати дані у цифровому вигляді за допомогою закритого ключа. Дані з цифровим підписом можна було використовувати для доведення того, що дані були отримані від певного власника, хоча відомий був тільки відкритий ключ цього власника. Тому не було необхідності знати реальну особу відправника, щоб підтвердити справжність відправника повідомленням. Чаум створив систему анонімних електронних платежів, згодом заснувавши компанію DigiCash в 1989 [9], яка розробила криптовалюту eCash, орієнтовану на анонімність та неможливість відстеження платежів за допомогою сліпих підписів. eCash був розроблений для забезпечення мікроплатежів через криптографічно захищені файли, у яких зберігалася певна грошова вартість та які генерувалися в обмін на фіатні (випущені державами) гроші. Отже, клієнтам необхідно було спочатку переказати кошти зі свого відповідного банківського рахунку на свій рахунок eCash. Після цього банк створював зашифровані файли, в яких зберігалася певна значення, яке також зберігалася на комп'ютері клієнта eCash. Щоб уникнути проблеми подвійного витрачання, концепція Чаума покладала на банки відповідальність за перевірку того, що електронні гроші ще не були витрачені в інших транзакціях. Однак концепція Чаума, як і раніше, залежала від посередників, таких як довірені органи, що беруть в ній участь. eCash тестувалась у Сполучених Штатах як система мікроплатежів з 1995 по 1998 рік. Пізніше PayPal уможливив негайний обмін активами від людини до людини і від покупця до продавця, і ідеї Чаума про анонімність платежів втратили актуальність.

Дерево Меркла – це структура даних, заснована на хеш-функціях і представлена Ральфом Мерклом [10, 11]. Дерева Меркла схожі структурою на перевернуте дерево, що складається з вузлів, з'єднаних вершинами. Існує два типи вузлів: листові вузли та нелістові вузли. Як правило, кожен нелістовий вузол n_p з'єднаний з двома окремими дочірніми вузлами n_{c1} і n_{c2} , де n_{c1} і

n_{C2} можуть бути як листовими, і нелістовими вузлами. Відповідний батьківський вузол n_p позначений хеш-значенням h_p , яке є хешованою конкатенацією хеш-значень його дочірніх елементів h_{C1} , h_{C2} , а саме $h_p = \text{hash}(h_{C1} + h_{C2})$. У нижній частині Древа Меркла, кожен листовий вузол є хеш-значенням набору даних. Листові вузли пов'язані з дочірніми вузлами. Таким чином, хеш-значення вузлів ітеративно об'єднуються і хешуються, починаючи з листових вузлів і продовжуючи до кореня дерева Меркла, який називається кореневим хешем. Кореневий хеш можна навіть використовувати для перевірки цілісності кількох блоків даних і великих структур даних, які представлені окремими хеш-значеннями, які потрібні кінцевим вузлам. В даний час дерева Меркла в основному використовуються в однорангових мережах, таких як мережа Bitcoin та Tor [11]. Одна з переваг такої структури даних полягає в тому, що немає необхідності знати все дерево Меркла, щоб перевірити цілісність блоку даних. На рис. 1.1, наприклад, цілісність блоку даних DB2 може бути перевірена негайно – якщо дерево вже містить $h1$ і $h6$ – шляхом хешування блоку даних та ітеративного об'єднання результату з хеш $h1$, а потім $h6$ і, нарешті, порівнянням результату з коренем хеша. Таким чином, термінальні пристрої, розмір сховища яких обмежений можуть перевіряти цілісність блоків даних, не виконуючи повну реплікацію реєстру.

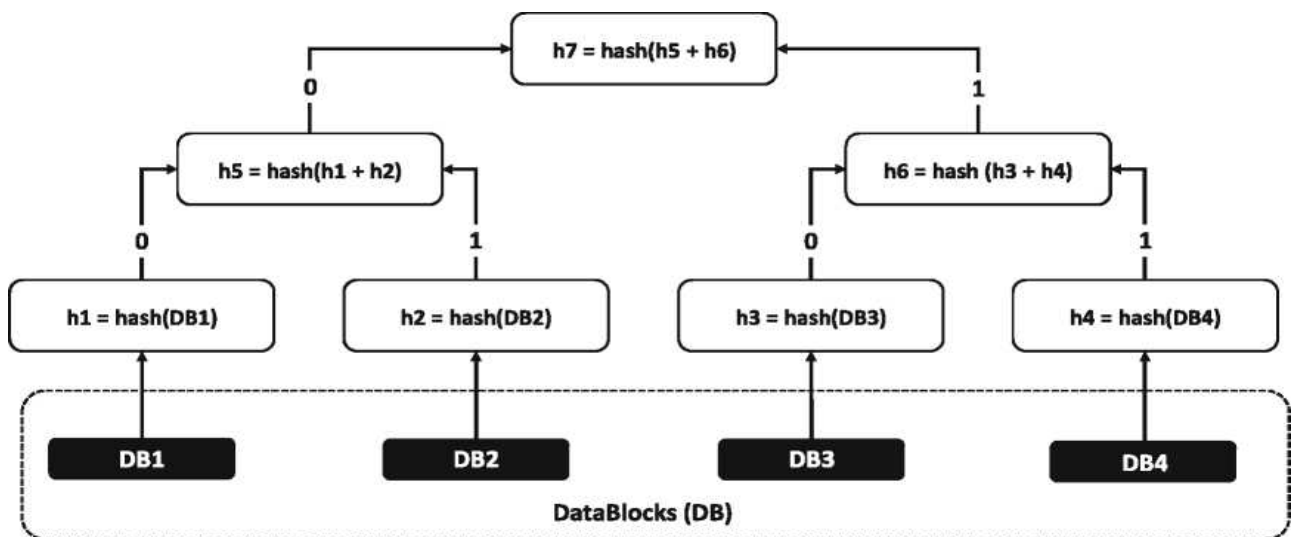


Рис. 1.1. Приклад бінарного дерева Меркла

1.2 Консенсус та роль технології розподілених реєстрів

На відміну від традиційної централізованої бази даних, розподілений реєстр представляє собою однорангову децентралізовану мережу, в яку потенційно може додати свій внесок будь-який учасник мережі. Для успішного функціонування такої розподіленої мережі необхідно схвалення зміни її стану усіма учасниками, враховуючи потенційну наявність великої множини вузлів: всі вони повинні узгодити стан мережі, щоб вона працювала належним чином.

Механізм консенсусу дозволяє користувачам та програмам координувати свої дії в розподіленій мережі. Тобто механізм консенсусу – це процес, за допомогою якого мережа надійно та автоматично визначає, який з представлених блоків учасників, що містить останні транзакції, буде доданий до реєстру [12, 13] .

Роль алгоритмів (або як їх ще називають, протоколів) консенсусу полягає у досягненні рівня відказостійкості мережі, побудованої на взаємодії вузлів (пристроїв , з'єднаних з іншими пристроями як частина комп'ютерної мережі) за рахунок забезпечення узгодженості між усіма учасниками мережі про поточний стан даних, навіть коли деякі ноди виходять з ладу. Це означає, що, якщо здійснена транзакція, то алгоритм консенсусу почне функціонувати для її верифікації, а саме обмінюватися внесеними даними в розподіленій мережі для перевірки того, чи може чи це дія мати місце . Далі розглянуто основні алгоритми консенсусу [14]:

- доказ роботи Proof of Work (PoW);
- доказ частки володіння Proof of Stake (PoS);
- делегований доказ частки Delegated Proof of Stake (DPoS);
- алгоритм практичної візантійської стійкості до відмови Practical Byzantine Fault Tolerance (PBFT).

1.2.1 Доказ роботи Proof of Work

Proof of Work – це перший алгоритм блокчейна , представлений у мережі

блокчейнів [15]. Ідея Proof of Work була запропонована задовго до виникнення блокчейну [16,17], вже пізніше Сатоші Накамото застосував її у статті про биткойнах у 2008 році. Алгоритм PoW працює, вимагаючи від вузлів у мережі вирішення складної математичної задачі для створення наступного блоку та перевірки легітимності транзакцій у мережі. Алгоритм PoW наведено на рис. 1.2.

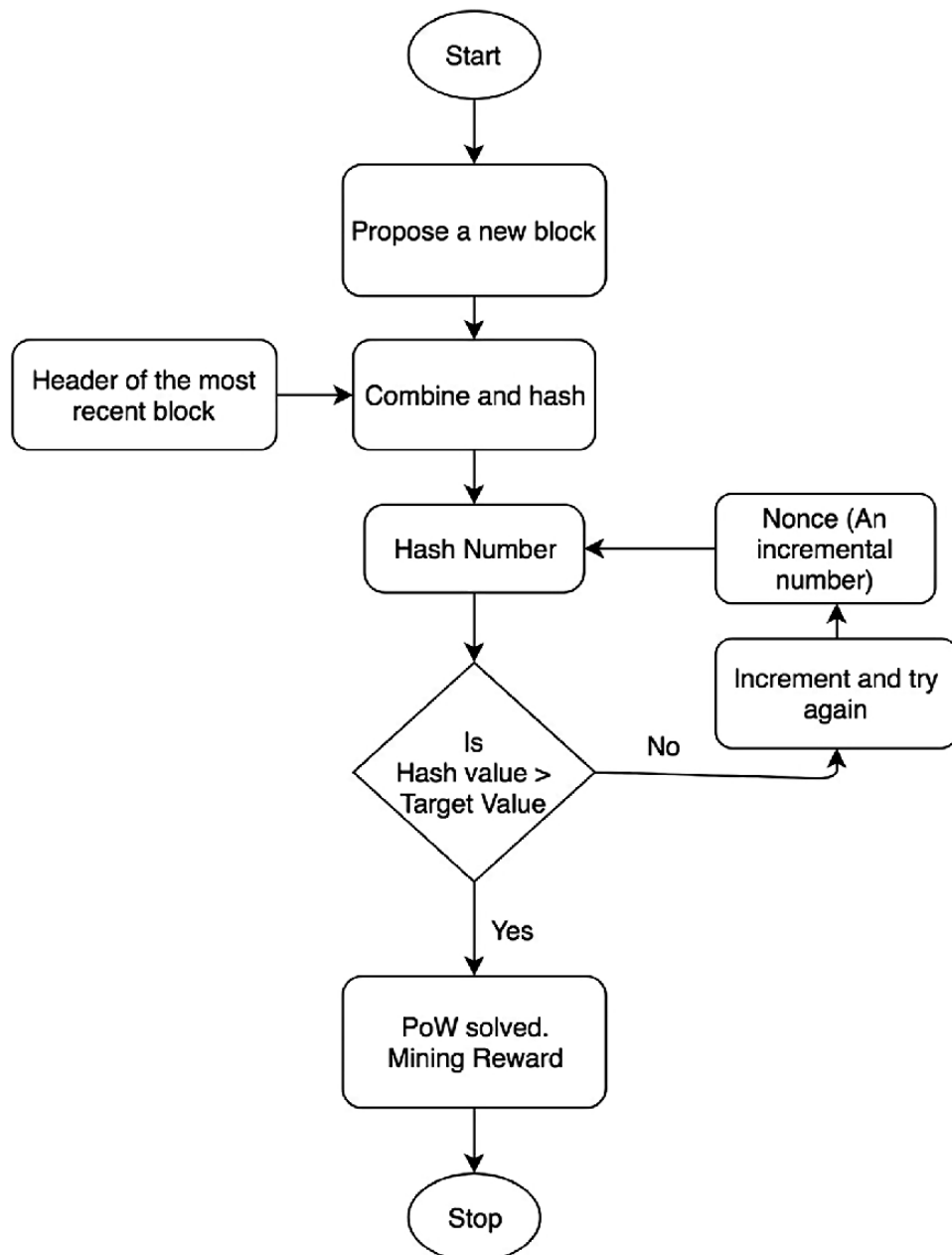


Рис. 1.2. Алгоритм функціонування Proof of Work [13, 18]

Математичне рішення виконується за допомогою хеш-функції [19]. Усі вузли змагаються за те, щоб першими знайти рішення методом перебору, що

потребує величезної кількості спроб. Перший учасник, який знайшов рішення, може мати право створити новий блок, і після перевірки блок додається до реєстру. Вузли, що беруть участь у обчисленнях, називаються майнерами, а процес розв'язання завдання називається майнінгом [20]. Цей процес також застосовується для створення нових вузлів даних у блокчейні або при синхронізації мережевого обладнання, щоб забезпечити узгодженість всього консенсусу.

Перевагою алгоритму Proof of Work є його висока безпека та значна ступінь децентралізації.

Однак його основним недоліком є більше споживання енергії та ресурсів. Майнерам потрібні великі обчислювальні потужності, щоб знайти рішення складного математичного завдання з погляду хешування мільярдів адрес. Це призводить до серйозних витрат ресурсів (гроші, енергія, простір, обладнання). Крім того процес досягнення Proof of Work консенсусу займає багато часу. Майнери повинні вивчити велике кількість значень *nonce* (рис. 1.2), щоб знайти правильне рішення проблеми, яку необхідно вирішити для видобутку блоку [20, 21].

1.2.2 Доказ частки володіння Proof of Stake

Щоб вирішити проблему великого споживання обчислювальної потужності PoW, в 2012 р. запропоновано алгоритм консенсусу Proof of Stake (PoS) [22]. Процес PoS відрізняється від PoW тим, що учасникам протоколу PoS не потрібно вирішувати математичне завдання для досягнення консенсусу. Їм потрібно використовувати накопичену криптовалюту як ставку для досягнення консенсусу. Схема алгоритму PoS представлена рис. 1.3.

Автор нового блоку вибирається з пулу користувачів, які поставили певну кількість криптовалюти, і ніхто з користувачів не може заздалегідь передбачити його дії. PoS ефективно вирішує проблему розтрати ресурсів у PoW.

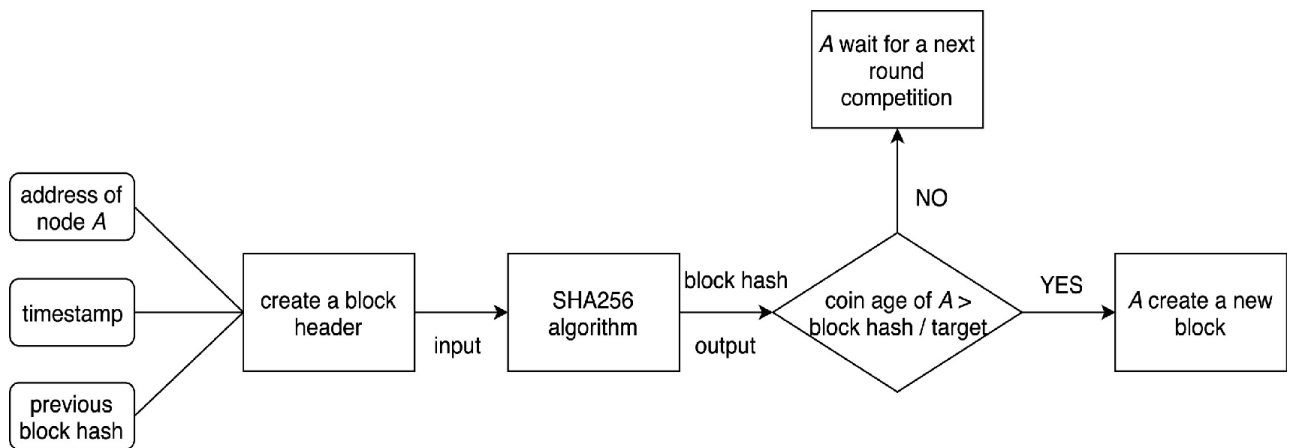


Рис. 1.3. Схема функціонування алгоритму Proof of Stake [13, 23]

Зі зростанням вартості блокчейну підвищується безпека блокчейну, що використовує PoS. Зловмисникам необхідно накопичити велику кількість монет і зберігати їх протягом тривалого часу, щоб атакувати блокчейн. Це також значно збільшує складність атаки. Хоча цей метод знижує втрати обчислювальної потужності, він може мати ризик монополії, що призводить до тенденції до централізації системи, дозволяючи зловмисникам мати чітку мету для атаки, ризикуючи безпекою [24].

1.2.3 Делегований доказ частки Delegated Proof of Stake

Алгоритм було запропоновано в [25]. Він пропонує поліпшити безпеку PoS, покладаючись на зацікавлені сторони, які проголосували за вибір виробників блоків. Відповідно до [26], DPoS, дозволяє власнику активувати певну кількість вузлів і задіяти їх для верифікації та обліку. У блокчейні з DPoS кожен вузол може вибирати верифікаторів з урахуванням своєї частки. Алгоритм DPoS показано на рис. 1.4.

Кількість верифікаторів N вибирається таким чином, щоб не менше 50% зацікавлених сторін із правом голосу вважали децентралізацію достатньою. Верифікатори повинні забезпечити достатній час перебування в мережі онлайн. DPoS максимально використовує голоси зацікавлених сторін для справедливого та демократичного досягнення консенсусу [26].

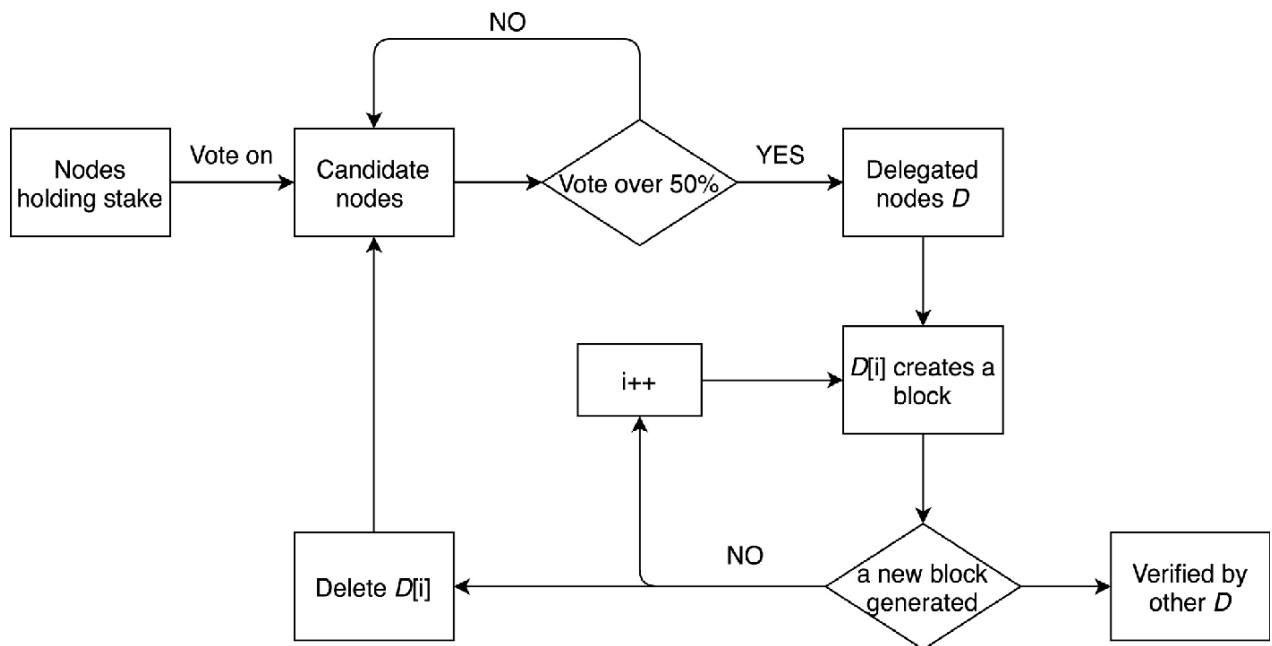


Рис. 1.4. Схема функціонування алгоритму DPOS [26]

1.2.4 Алгоритм практичної візантійської стійкості до відмови Practical Byzantine Fault Tolerance (PBFT)

Алгоритм PBFT було запропоновано у 1999 році, для початкового вирішення завдання візантійських генералів [27]. Алгоритм PBFT спрямований забезпечення практичної реплікації кінцевого автомата усунення візантійської помилки [28]. Максимальна кількість шкідливих або вузлів мережі f , що відмовил, не може перевищувати $1/3$ від загальної кількості вузлів n , отже, $n > 3f + 1$ буде потрібно для PBFT консенсусу. Процес алгоритму PBFT поділено на 3 етапи: попередня підготовка, підготовка та фіксація [29]. Алгоритм PBFT показано на рис.1.5.

Відповідно до [25] енергоефективність та висока пропускна спроможність розглядаються як переваги DPOS, а деякі моменти, такі як мала кількість або відсутність параметрів, доступних для масштабованості, та можливі затримки, оскільки мережа повинна чекати на голосування всіх вузлів, зазначені як недоліки DPOS.

Згідно [31] на даний час існує понад 70 алгоритмів консенсусу для розподілених платформ. Найбільш перспективним є алгоритм Delegated Proof of Stake [32].

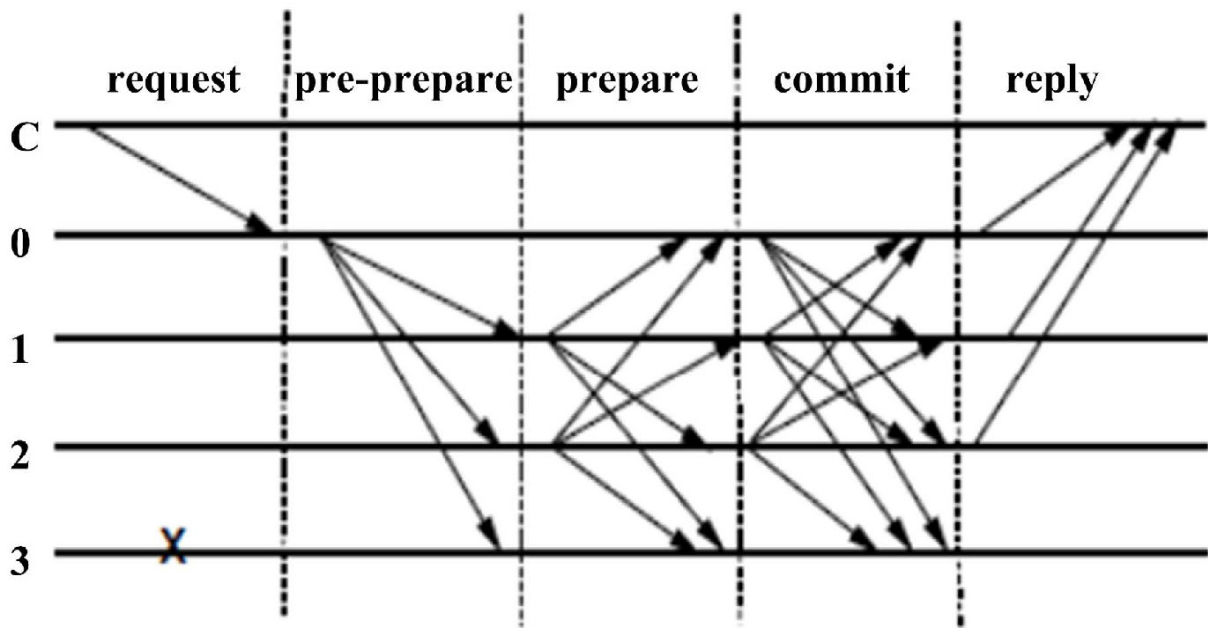


Рис. 1.5. Схема функціонування алгоритму DPoS [30]

1.3 Практичне застосування технології децентралізованих реєстрів

Хоча фактично технологія децентралізованих реєстрів була практично реалізована Сатоші Накамото в 2009 р. як перша криптовалютна система, блокчейн за майже 25 років свого існування став революційною технологією, що торкнулася різних областей життя суспільства [33]. На початкових етапах еволюції блокчейна основними напрямками його розвитку були фінансові застосування. Сюди передусім належать численні криптовалюти [34-36]. Згідно аналізу Forbes на сьогоднішній день існує, як повідомляє CoinMarketCap, близько 23000 криптовалют із загальною ринковою капіталізацією в 1,1 трильйона доларів [37]. Це досить багато з огляду на те, що Bitcoin був запущений тільки в 2009 році.

З переходом до наступного покоління Blockchain 2.0 у вигляді глобальної децентралізованої платформи Ethereum з'явилися нові можливості застосування DLT за рахунок побудови смарт-контрактів та децентралізованих додатків. Це своє чергу сформувало як окрему прикладну фінансову галузь децентралізовані фінанси (Decentralized Finance, DeFi). Сьогодні під DeFi розуміють аналоги традиційних фінансових інструментів, реалізованих в децентралізованій архітектурі [38]. DeFi існують у вигляді загальнодоступних сервісів, вони

зазвичай є проектами з відкритим вихідним кодом і найчастіше засновані на смарт-контрактах. До DeFi відносяться децентралізовані біржі (DEX), ринки передбачень та протоколи випуску різних токенів. У роботі [39] DeFi сприймається як екосистема. Особливості та проблеми DeFi детально розглянуті у [40, 41, 42]. Коректна робота всієї DeFi екосистеми забезпечується численними децентралізованими додатками (Decentralized Applications , DApps) [43, 44].

Крім фінансових застосувань DLT знаходять застосування у різних галузях бізнесу та соціальних сферах [45]. Дуже інтенсивно розвивається напрям практичного застосування DLT, такий як інтернет речей (Internet of Things, IoT) [46 – 49].

Серед інших застосувань DLT слід зазначити механізми електронного голосування [59, 51]. В останні роки DLT широко застосовуються в медицині та охороні здоров'я [52 – 54]. Інші багаточисельні перспективні застосування DLT в бізнесі, індустрії та соціальній сфері розглянуто в [55 – 67].

На рис. 1.6 наведено розглянуті області практичного застосування DLT.

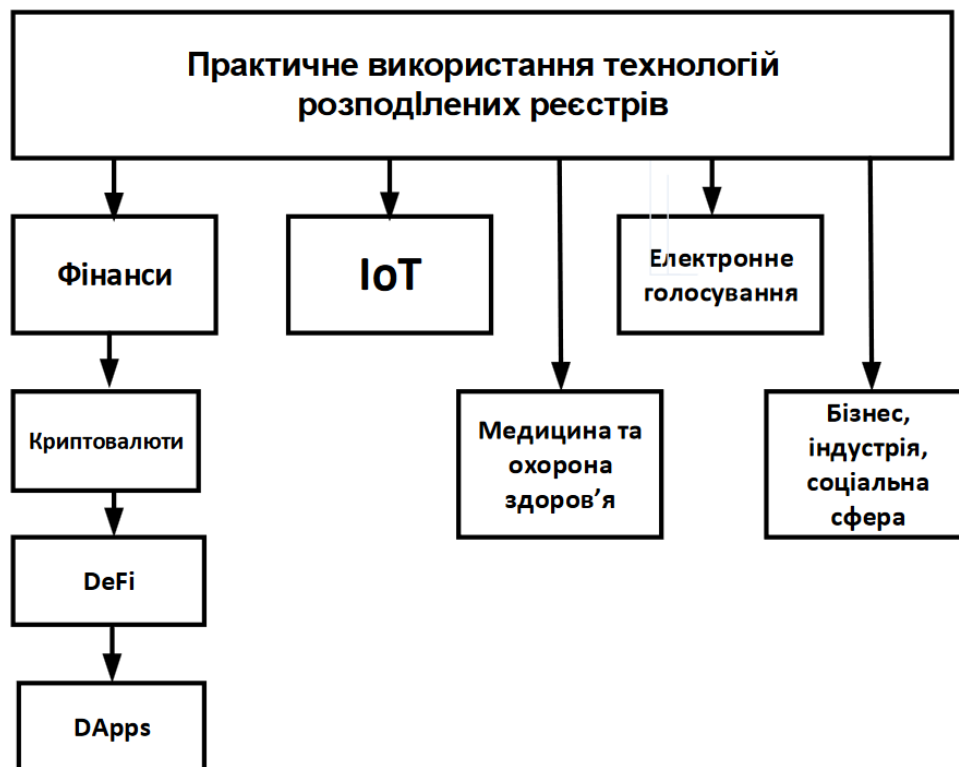


Рис.1.6. Практичне використання технологій децентралізованих реєстрів

Зазначимо, що для всіх перелічених застосувань DLT поряд з позитивними властивостями блокчейн-технологій спостерігається недостатня швидкість обробки транзакцій та проблеми, що виникають при високих рівнях масштабування. Це викликає інтенсивний пошук нових ідейних, алгоритмічних та технічних рішень для їх подолання.

1.4 Покоління технології розподілених реєстрів

У розділі 1.4 проаналізовані чотири покоління DLT (рис. 1.7):

- криптовалюти Bitcoin;
- DNL у вигляді платформи Ethereum 1.0;
- DNL у вигляді платформи Ethereum 2.0;
- систем на основі BlockDAG.

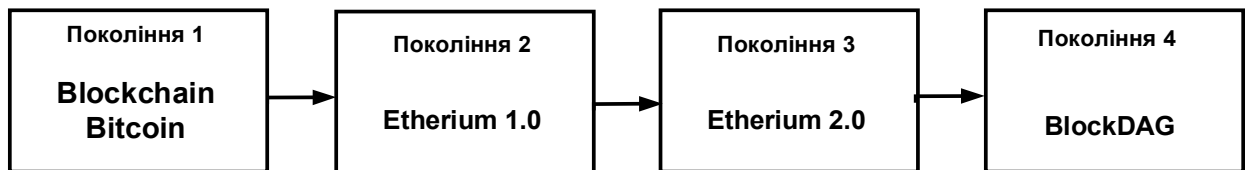


Рис. 1.7. 4 покоління DLT

1.4.1 Bitcoin як перше покоління DLT

Bitcoin як перше покоління Blockchain було представлено роботою анонімного автора, присвяченої криптовалюти Bitcoin [68]. Блокчейн є ланцюжком послідовно з'єднаних блоків записаних хешованих транзакцій. Приклад роботи блокчейна в Bitcoin-мережі представлений на рис. 1.8 [69].

Блок є набір хронологічних записів здійснених транзакцій. Кожен блок пов'язаний з попереднім блоком за допомогою хешу останнього блоку показника, відомого як батьківський блок.

Першим блоком блокчейна є «блок генезису», який не має батьківських блоків. Обчислювальні пристрої користувачів мережі блокчейн відомі як «вузли». Базовий будівельний блок блокчейну, який передає ресурс між однорангові вузли в мережі блокчейна, називається транзакція. Деякі з

конкретних вузлів виконують процес перевірки блоку перед додаванням будь-якої транзакції та блоку структури блокчейна. Ці вузли називаються майнерами. Протокол консенсусу – це набір правил, що забезпечують одностайну згоду операцій блокчейну. Таким чином, вузол, транзакція, блок, ланцюжок, майнер та процес консенсусу є компонентами архітектури блокчейну. Блок є набір хронологічного запису досконалих, але з підтверджених наборів транзакцій. Заголовок блоку та тіло блоку є двома частинами блоку (рис. 1.8). Хеш кореня дерева Меркла – це хеш, розрахований для транзакцій у блоці для швидкого тестування транзакцій. Часова мітка представляє позначку поточного часу в секундах для забезпечення прозорості блокчейна. Сегмент тіла блоку має лічильник транзакцій для відстеження кількості транзакцій у блоці.

Contents of a Block		
Size	Field	Description
32 bit	Block Size	Block Size following this field
640 bit	Block	Several fields form the block
8-72 bit	Transaction Counter	How many transactions follow
Variable bytes	Transactions	The transactions recorded in this
Contents of a Block Header		
Size	Field	Description
32 bit	Version	Track software protocol upgrades
256 bit	Previous Block	A point to the hash of the previous
256 bit	Merkle Root	A hash root of the Merkle tree of
32 bit	Timestamp	Block creation time
32 bit	Difficulty Target	The PoW algorithm
32 bit	Nonce	A counter track proof of work

Рис. 1.8. Архітектура блоку та заголовок блоку

Робочий механізм блокчейну. Будь-який вузол може ініціювати транзакцію із системи блокчейну, використовуючи цифровий підпис і транслюючи його до мережі. У цей момент вузол повинен вибрати та схвалити згенеровані транзакції згідно деяким визначеним правилам. Потім він

включається до блоку після перевірки та підтвердження транзакції деякими майнерами. Вузли, які виконують обчислювальну перевірку та перевірку транзакцій та блоків, відомі як майнери. Майнер має вирішити обчислювально складну головоломку і витратити частину своєї зареєстрованої обчислювальної потужності для формування блоку. Майнер, який першим вирішить обчислювально-складну головоломку, може створити новий блок. Потім усі вузли в мережі блокчейну перевіряють новий блок за допомогою протоколу консенсусу для досягнення згоди з деяких питань. Потім знову створений блок додається до існуючого найдовшого ланцюжка. На рис. 1.9 [69] показано роботу блокчейна в Bitcoin.

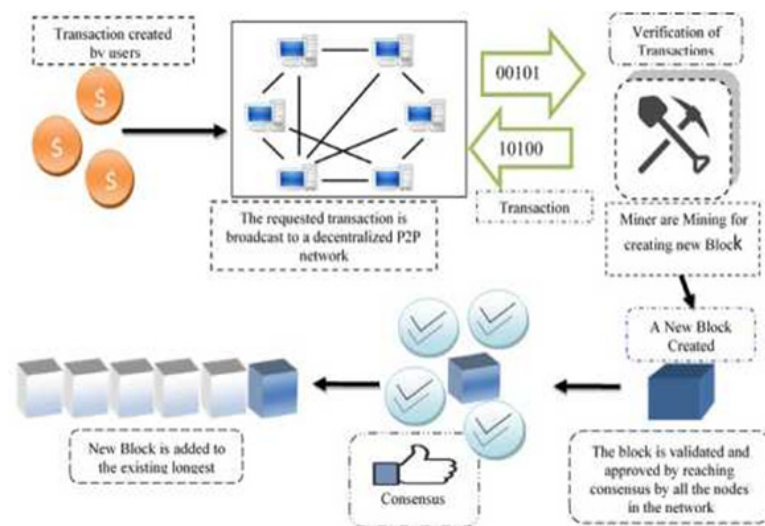


Рис. 1.9. Робота Blockchain у Bitcoin

На сьогоднішній день ринкова капіталізація Bitcoin (BTC), побудованого на технології Blockchain першого покоління, становить 570,6 мільярдів доларів США, що становить домінування цієї криптовалюти 46.7% [70].

1.4.2 Ethereum 1.0 як друге покоління DLT

DLT імплементована у вигляді платформи Ethereum 1.0 прийнято вважати другим поколінням блокчейна. Концептуально Ethereum 1.0 було сформульовано В. Бутеріним у 2014 р. [71]. У 2015 р. була випущена перша версія платформи Ethereum (Ethereum 1.0) під назвою Frontier. Відмінні риси платформи:

– Можливість виконання смарт-контрактів. Смарт-контракт – це програма, яка автоматично виконує прописані в коді транзакції при виникненні відповідних умов. Концепція смарт-контрактів з'явилася 1994 року, коли автор [72] дійшов висновку, що з допомогою децентралізованого реєстру можна укласти контракти, що автоматично виконуються. Однак на практиці реалізувати цю ідею вийшло лише у 2008 році, завдяки появі технології блокчейн у криптовалюті Bitcoin. Однак повністю реалізувати концепцію смарт-контрактів на Bitcoin не вийшло, оскільки творці платформи обмежили програмування смарт-контрактів з метою безпеки. Активний розвиток smart-contracts отримали лише з появою Ethereum 1.0.

Особливості Ethereum 1.0:

– Віртуальна машина Ethereum (EVM, Ethereum Virtual Machine) – це стекове середовище, що реалізує смарт-контракт. EVM виконує байткод контракту, отриманий компіляцією вихідного коду контракту, представленого мовою високого рівня. Машина EVM реалізована у вигляді програми, екземпляр якої знаходиться на будь-якому вузлі мережі Ethereum, що має повну функціональність. EVM є тьюрінг -повним автоматом, і має у своєму розпорядженні набір з приблизно 140 команд.

– Наявність мови програмування Solidity. Solidity – об'єктно-орієнтована повна по Тьюрингу мова програмування для реалізації смарт-контрактів. Solidity має Java-подібну структуру і дає можливість програмування розподілених додатків DApps [73].

– Криптовалюта Ether (ETH). Інший розрахунковою одиницею на платформі Ethereum є «газ» (gas) – розрахункова одиниця в EVM , що застосовується при розрахунку комісії за транзакцію .

Ethereum 1.0 використовував консенсус PoW. Майнери групують транзакції до нових блоків приблизно кожні 12 секунд. Блокчейн Ethereum опрацьовує 12 транзакцій на секунду [74, 75].

1.4.3 Ethereum 2.0 як третє покоління DLT

Ethereum 2.0 – це планована послідовність змін у протоколі Ethereum , спрямована на покращення його загальної продуктивності [76]. Якщо створення Ethereum започаткувало нову еру експериментів у фінансових додатках, то Ethereum 2.0 прагне продовжити цей прогрес, змінюючи дизайн мережі. Це передбачає перехід на новий механізм консенсусу, який керуватиме тим, як транзакції затверджуються мережею та поширюється її криптовалюта .

15 вересня 2022 р. стався по-справжньому історичний момент для Ethereum. Стався так званий The Merge – масштабне оновлення Ethereum 1.0 до Ethereum 2.0. Версія Ethereum 2.0 міститиме три основні оновлення блокчейну Ethereum [77, 78]:

1. Консенсус PoS

Відповідно до моделі PoS Ethereum механізм системи голосування містить такі компоненти. По-перше, великий пул учасників, кожен з яких становить до 32 ЕТН, необхідні для активації . Цей пул не голосує за блоки безпосередньо – він поділений на комітети, члени яких випадково обираються з-поміж членів ширшого пулу. Причина поділу на комітети полягає в тому, що не кожен учасник може брати участь у голосуванні за кожен блок – інакше блокчейн містив би надто багато даних про голоси, що унеможливило б масштабування. Крім того, комітети дозволяють збирати дані про голосування до контрольованих фрагментів даних. Відповідно до специфікацій Eth2 цільова кількість учасників у кожному комітеті – 128 (бажаний мінімум). Підписи голосів можна об'єднувати, що дозволяє зменшити необхідний розмір блоку та забезпечити масштабування мережі. Комітети обираються випадковим чином.

Блоки та блоки-контрольні точки. Один з кожних 32 блоків називається блок-контрольна точка (чекпойнт, checkpoint), часовий проміжок між контрольними точками називається епохою (epoch). Фактично поняття епохи було запропоновано протоколі Ouroboros [79]. У межах кожної епохи є 32 слоти (інтервалу) тривалістю 12 секунд, протягом яких можна пропонувати блоки. Таким чином, кожна епоха має 32 слоти для 32 комітетів. Після завершення

кожної епохи члени Комітету змінюються місцями. Один член комітету має монопольне право пропонувати блок протягом інтервалу 12 секунд, а інші члени можуть голосувати за блок. Це голосування називається атестацією. Процес схематично представлений на рис. 1.10 [80].

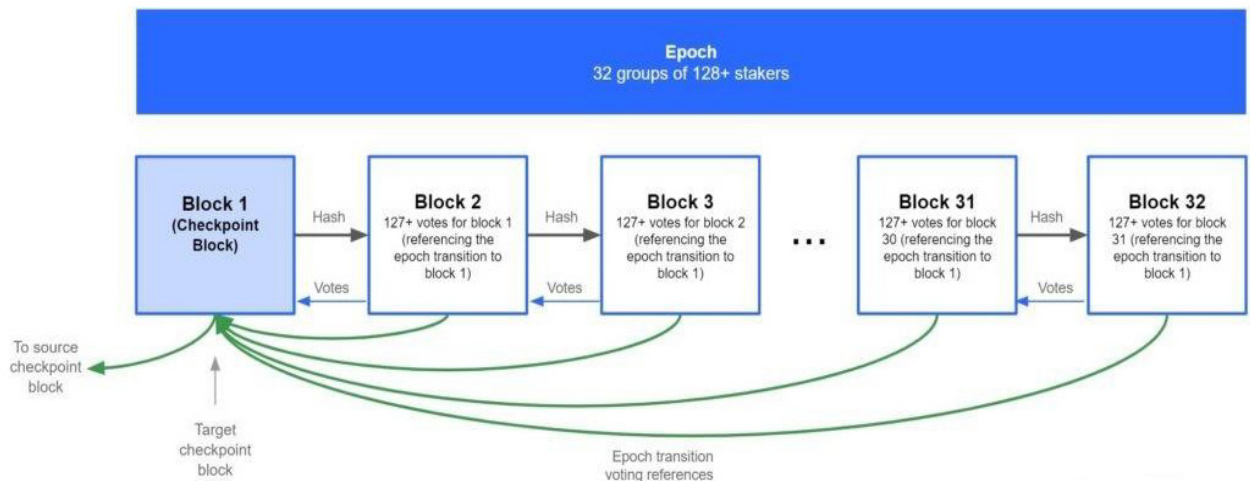


Рис.1.10. Призначення вузлів до комітетів у Beacon Chain
(передбачається, що на слот припадає один комітет)

2. Шардинг Ethereum [80]

Шардинг – це поділ мережі на автономні блокчейни (шарди), кожен зі своїм «міні-блокчейном». Із введенням шардингу нодам не доведеться обробляти кожну транзакцію в реальному часу. Це дозволить збільшити швидкість обробки транзакцій та збільшити пропускну здатність мережі. Спочатку розгорнуто експериментальна мережу з 64 шардами . Кожна частина або сегмент (шард) діє як власний блокчейн і дозволяє призначеним для них валідаторам зберігати дані, обробляти транзакції та додавати нові блоки у свій конкретний ланцюжок сегментів. Це означає, що шард-вузли містять певну підмножину блокчейна Ethereum і більше не потрібно зберігати всю історію Ethereum. Схему процесу наведено на рис. 1.11 [80].

3. ETH 2.0 Beacon Chain [80].

Beacon Chain є фактично другим координуючим шаром для основної мережі, в яку винесено здійснення протоколу PoS (рис.1.12) . Кожен шард буде

пов'язаний з Beacon Chain, де валідатори будуть по черзі створювати та перевіряти нові блоки сегментів, перш ніж додавати їх до решти мережі.

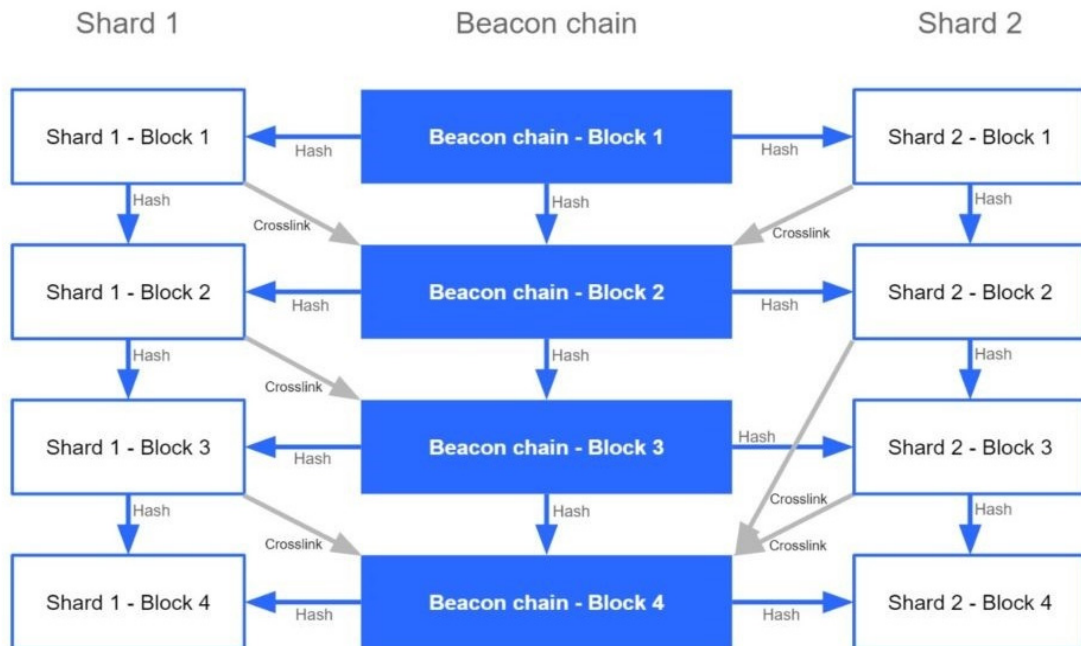


Рис.1.11. Шардинг в Ethereum 2.0 (показано два шарди) [80]

Consensus layer

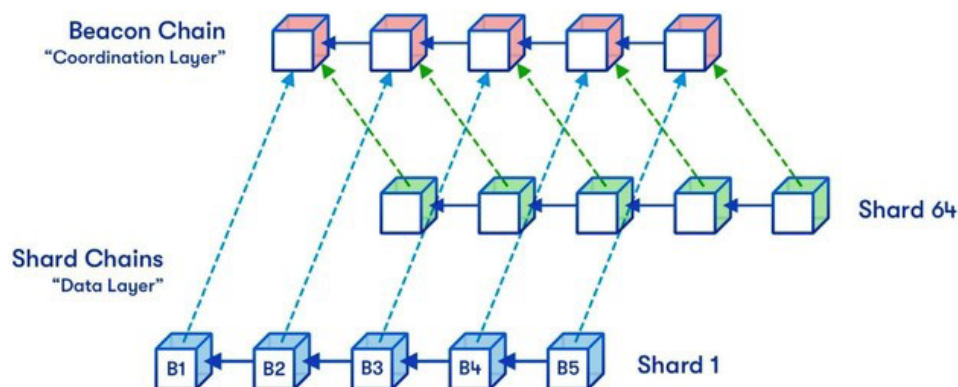


Рис. 1.12. Двошарова побудова Ethereum 2.0

Фіналізація в Ethereum 2. 0. [80] Коли члени комітету голосують за блок, вони голосують не лише за певну пропозицію блоку, але вони також повинні послатися на конкретний історичний блок – чекпойнт і проголосувати за нього. Саме цей механізм дозволяє гарантувати, що процес голосування стабільний. Тому фактично відбуваються дві процедури голосування за моделлю докази частки володіння, одна всередині іншої. Рис. 1.13 ілюструє, як відбуваються два

типи голосування і в яких блоках можуть зберігатися ці голоси. Голосування та посилення на умови ефективною комунікації (при припущенні, що на один слот припадає один комітет) відбувається таким чином. Блок може стати «підтвердженим», якщо поверх його створено чекпойнт-блок, та в каталозі всіх комітетів у рамках однієї епохи на цей чекпойнт у своєму голосуванні посиляються більше двох третин членів комітету. Блок може стати «підтвердженим» не раніше, ніж після закінчення двох третин епохи. Наступна стадія – фіналізація (рис.1.13) [80]. Блок фіналізується, коли блокчейн містить два підтверджені блоки після нього. Тому в більшості випадків, коли поріг у вигляді двох третин голосуючих досягнуто достатньо швидко, завдяки ефективним каналам комунікації, користувачу потрібно чекати підтвердження протягом однієї епохи (6,4 хвилини), а фіналізації – протягом двох епох (12,8 хвилин).

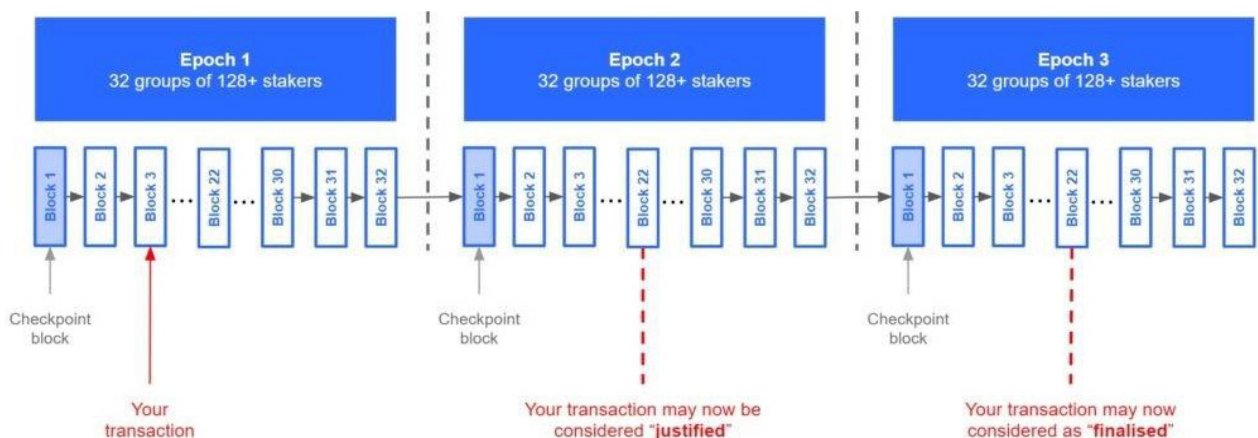


Рис. 1.13. Процес фіналізації в Ethereum 2.0 [80]

Переваги та недоліки Ethereum 2.0

Ethereum 2.0 може забезпечити мережі такі переваги, як:

- зниження споживання електроенергії – на 99,5%;
- відсутність вимог до обладнання;
- достатньо висока надійність завдяки системі валідаторів, які отримують фінансовий стимул;
- поліпшена децентралізація (при майнінгу значна частина потужностей була зосереджена у корпораціях стейкерів).

Недоліки:

- Ethereum 2.0 виключно складний. У системі з численними комітетами, шардами та різними типами голосування високий ризик збоїв та затримок у реалізації оновлень;

- очікувані зниження комісій та багаторазове прискорення транзакцій поки що не підтверджені.

1.4.4 BlockDAG як четверте покоління DLT

На відміну від блокчейну, як технології DLT з лінійним ланцюжком послідовними блоків, існують технології, в яких блоки/ транзакції утворюють структури, що відомі в теорії як спрямовані ациклічні графи (directed acyclic graph, DAG), оскільки вони можуть (а іноді й повинні) посилатися на кілька попередніх блоків/транзакцій [81 – 83]. У цьому випадку транзакції створюються безпосередньо учасниками мережі, а не спеціальними вузлами (майнерами чи ін.). Деякі системи накладають додаткові обмеження на посилення, дозволяючи, як і в блокчейні, тільки останні відомі блоки/ транзакції, на які потрібно посилатися .

У всіх системах на основі DAG існує проблема лінеаризації (впорядкування), оскільки посилальна структура DAG в загальному випадку є частково упорядкованою множиною. У такому разі необхідно створення певного упорядкування блоків порядку деякими додатковими діями [84] .

DAG не є новою концепцією або технологією, і це безумовно не механізм консенсусу; це математична структура, що виникла багато століть тому [85]. Технічно DAG – це граф зі спрямованими ребрами і без циклів (тобто не існує шляхів від вершини назад до себе). У контексті розподілених реєстрів BlockDAG – це DAG, вершини якого представляють блоки, а спрямовані ребра – посилення блоків на їх попередників.

Концепція застосування DAG в DLT. та підтип який має назву BlockDAG, може вважатися DLT четвертого покоління [86]. Прикладами blockDAG'ів є

SPECTER та PHANTOM [87,88]. Кожен з цих протоколів визначає алгоритм, який буде топологічне упорядкування блоків DAG, повторює DAG у цьому упорядкуванні та усуває транзакції, що конфліктують з попередніми.

Переваги спрямованого ациклічного графа [89, 90]:

1. Масштабованість. Основною причиною розвитку DAG стала неможливість блокчейна справлятися з великими навантаженнями та завантаженістю мережі. Пропускна здатність систем на основі DAG може теоретично досягати 1 млн та більше транзакцій в секунду.

2. Нижчі комісії. Масштабованість системи призводить до зниження комісій за транзакції.

За концепцією DAG побудовано платформи IOTA [91], Byteball [92], Hashgraph [93], Nano [94] та ін.

Таким чином, незважаючи на широку затребуваність та на чисельні дослідження та розробки в області технологій DLT, при впровадженні стикаються з недостатньою ефективністю існуючих рішень щодо швидкості обробки транзакцій при високих масштабуванні та криптозахищеності. Тому в даній роботі запропоновано ідею підвищення ефективності технології блокчейну третього покоління Ethereum 2 з урахуванням переваг мережі Block DAG, тобто ідею створення двошарової моделі розподіленого реєстру, що поєднує в собі основну мережу, побудовану за технологією Block DAG, та координаційну мережу, що базується на традиційній блокчейн технології.

1.5 Висновки до першого розділу

У першому розділі дисертаційної роботи проведено аналіз існуючих технологій DLT, встановлені їх переваги та недоліки. Показано, що відомі системи, побудовані на DLT, мають ряд серйозних недоліків. Розподілена система повинна мати механізм масштабування для адаптації до зміни навантаження у дуже широких межах. У той же час швидкість обробки даних у відомих популярних системах Bitcoin та Ethereum невисока – ці системи

обробляють приблизно 7 та 15 транзакцій за секунду (tps). Ці показники незрівнянні з традиційними централізованими системами, що обробляють тисячі транзакцій на секунду. Тобто виникає актуальне науково-практичне завдання підвищення ефективності обробки транзакцій у розподілених регістрах, що передбачає поєднання високої швидкості обробки транзакцій, високої масштабованості системи та достатньої криптозахищеності від маніпуляцій та атак.

РОЗДІЛ 2

РОЗРОБКА ТА ДОСЛІДЖЕННЯ ШЛЯХІВ ПІДВИЩЕННЯ ШВИДКОСТІ
ОБРОБКИ ТРАНЗАКЦІЙ І МАСШТАБОВАНOSTІ

2.1 Розробка двошарової моделі розподіленого реєстру

Для прискорення обробки транзакцій за високого рівня масштабованості запропоновано двошарову модель побудови системи. Основна мережа, призначена для формування та розповсюдження блоків, виконана за технологією BlockDAG (шар 1). Операції, пов'язані з верифікацією блоків на підставі протоколу консенсусу, винесені до координаційної мережі, що працює за технологією традиційного консенсусу (шар 2).

Як і в системі Ethereum [91] для обробки транзакцій застосовано дискретну часову шкалу з деякими модифікаціями. Слоти – мінімальні часові інтервали на які розбито лінію часу функціонування мережі. Слоти використовуються для ідентифікації такту протоколу. Слоти поєднуються в епохи. Епохи призначені для підбиття проміжних результатів функціонування мережі. Декілька послідовних «епох» називаються «ера». Кількість епох для ери визначається – кількість слотів, що дорівнює кількості учасників на момент закінчення попередньої ери. У ньому число слотів округляється до цілого числа епох.

У досліджуваній моделі слот має тривалість 4 с, епоха складається з 32 слотів.

Основним технологічним структурним елементом мережі є вузол (node, нода). Вузлом є зареєстрований сервер у мережі, що зберігає всі відповідні записи у вигляді реєстру. На кожному вузлі може бути розгорнуто певну кількість логічних структурних елементів, які умовно називатимемо Workers (Працівники), їх облікові записи мають необхідні дані для участі в протоколі консенсусу PoS. Кожен Worker після включення до системи складається з двох компонентів з незалежними адресами – Утворювач блоків (Creator) та Координатор (Coordinator) (рис. 2.1).

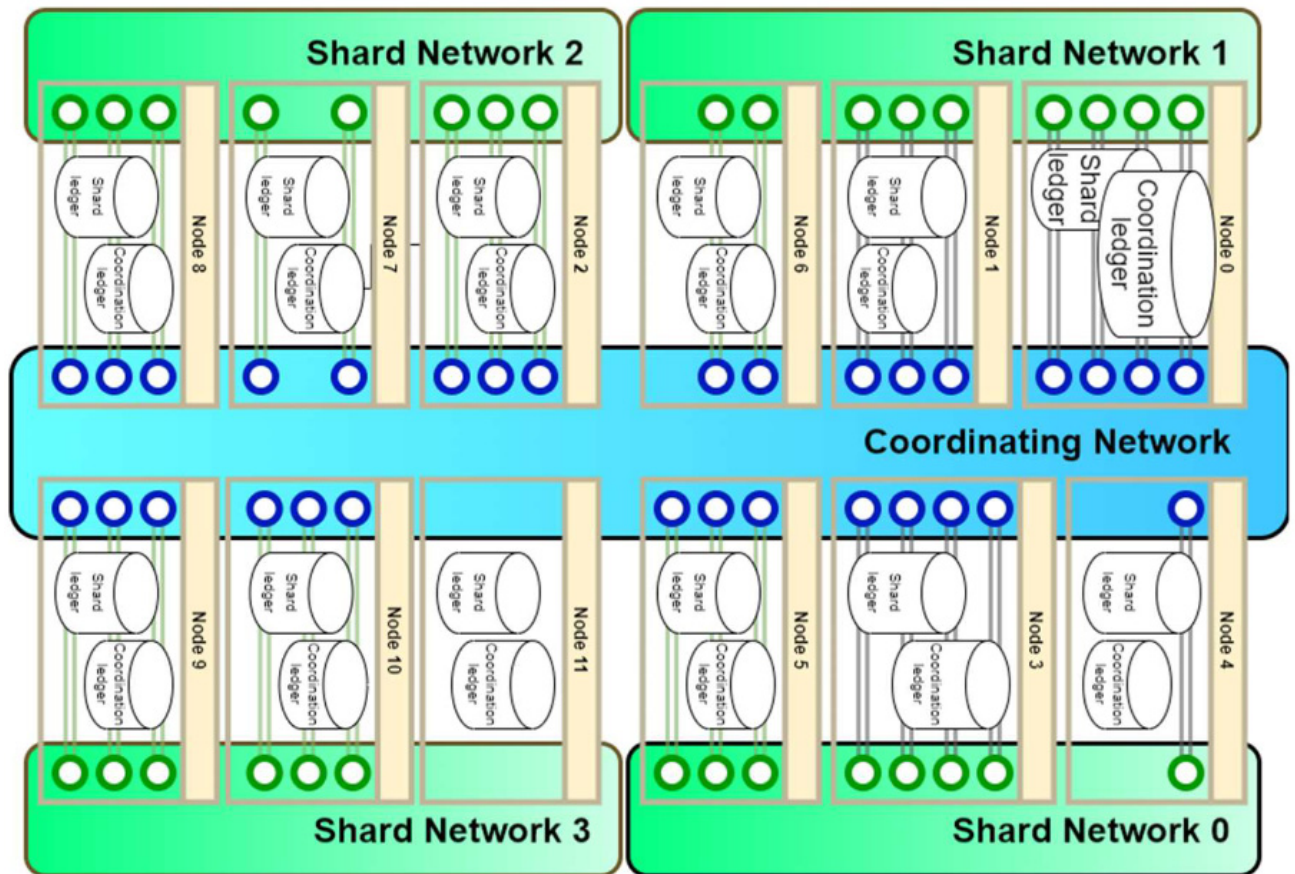


Рис. 2.1. Двошарова модель системи розподіленого реєстру

Функції обробки транзакцій розподілені між шарами системи таким чином. В основній мережі виконуються такі основні операції.

1. Пошук інших вузлів та підключення до них.
 2. Прийом транзакцій від користувачів, розміщення в пул транзакцій та пересилання транзакцій далі по мережі;
 3. Визначення методу перемішування, хто з утворювачів створює блок у кожному слоті.
 4. Видалення транзакцій з пулу, додавання до блоку, відправлення даного блоку іншим учасникам мережі.
 5. Передача своєму координуючому вузлу порядку блоків, що входять в цей вузол.
 6. Отримання від свого координуючого вузла порядку блоків для фіналізації.
- Під фіналізацією (або остаточністю) розуміється процес, після завершення якого

транзакція в мережі може вважатися остаточною і не існує ризику фальсифікації (зміни) транзакції або блоку в упорядкованому ланцюжку.

7. Збереження історії блоків та транзакцій у вузлах реєстру.

8. Участь у синхронізації блоків та транзакцій.

Основні операції, що виконуються в координаційній мережі, наступні:

1. Визначення в кожному слоті епохи складу комітетів Proof of Stake та ролей координаторів у них (утворювач, атестатор, агрегатор).

2. Прийом результатів атестації з інших вузлів і передачі її через мережу.

3. Додавання утворювачем до блоку атестацій, раніше не доданих до блоку.

4. Об'єднання сформованих агрегаторами атестацій в один мультипідпис.

5. Згідно з отриманими атестаціями та алгоритмом консенсусу формування ланцюжка блоків, що підлягають фіналізації; відправка фінального ланцюжка блоків у свій вузол основної мережі для фіналізації.

6. Синхронізація результату консенсусу з основною мережею.

7. Зберігання історії блоків та атестацій.

8. Збереження стану координаторів (баланси, статус) у загальному стані мережі.

2.2 Розробка моделі скороченого обміну інформацією між основною мережею та координаційною мережею на базі поняття хребетних блоків

2.2.1 Формалізація поняття хребетного блоку

Орієнтований граф або оргграф G є впорядкованою парою (V, E) , де $V = V(G)$ – непорожня множина вершин, а $E = E(G)$ – множина впорядкованих пар (a, b) – ребер графу [92]. Перша вершина упорядкованої пари називається початком ребра, а друга – кінцем; разом вони утворюють спрямовані ребра. Ребро направлено від початку до його кінця. Якщо існує ребро від a до b , то ми говоритимемо, що b є «дочірнім» a , а a є «батьківським» по відношенню до b . Позначимо це відношення (a, b) як $a \rightarrow b$ або $b \leftarrow a$ у випадку, якщо існує ребро від a до b . Шляхом або орієнтованим шляхом з a_k називається a_0 граф

P_{a_k, a_0} , вершини якого можна перенумерувати таким чином, що

$$V(P) = \{a_1, \dots, a_k\}, k \geq 1,$$

$$V(P) = \{a_0, a_1, \dots, a_k\}, k \geq 1,$$

$$\text{та } E(P) = \{a_k a_{k-1}, \dots, a_1 a_0\}.$$

Як вершини ациклічного графа ми будемо розглядати блоки, створенні в BlockDAG мережі. Ребра визначатимуться за допомогою посилань блоку на попередні блоки. Таким чином, у роботі ми розглядатимемо орієнтовані ациклічні графи, для яких існує вершина b_G , така, що для будь-якого $b \in V(G)$ існує шлях $P_{b, b_G} \subset E(G)$, блок b_G називатиметься генезис-блоком, його створення запускає роботу мережі.

Для множини X позначимо через $|X|$ – кількість елементів у множині X . Для $b \in V(G)$ позначимо через $\deg^+(b)$ кількість ребер, що виходять з вершини b , $\deg^+(b) = |\{(b, a) : (b, a) \in E(G)\}|$ і через $\deg^-(b)$ – кількість ребер, що входять у вершину b , $\deg^-(b) = |\{(a, b) : (a, b) \in E(G)\}|$.

Якщо $\deg^-(b) = 0$, то будемо говорити, що $b \in \text{tips}(G)$.

Оскільки час роботи BlockDAG мережі ділиться на слоти, то до звичайної структури орієнтованого ациклічного графа додається інформація про час створення блоку номер слота, в якому був створений блок. Для блоку, b створеного в n -му слоті будемо писати $b \in \text{Slot}(n)$. За визначенням вважаємо, що $b_G \in \text{Slot}(0)$. Це єдиний блок нульового слота.

Очевидно, що у добре синхронізованій мережі

$$b \in \text{Slot}(n) \Leftrightarrow \max_{P_{b, b_G}} |P_{b, b_G}| = n.$$

Будемо говорити, що блок a належить минулому блоку b (у графі G) і писати $a \in \text{past}(b, G)$, якщо існує шлях $P_{b, a} \subseteq G$ з блоку b до блоку a .

Побудуємо порядок в слоті орієнтованого ациклічного графа зі слотами. Нехай $b_1, b_2 \in \text{Slot}(n)$. Говоритимемо, що блок b_1 передує блоку b_2 і писати

$b_1 <_{sl} b_2$ якщо:

1. $|\text{past}(b_1, G)| > |\text{past}(b_2, G)|$, або
2. Якщо $|\text{past}(b_1, G)| = |\text{past}(b_2, G)|$, то $\deg^+(b_1) > \deg^+(b_2)$, або
3. Якщо $|\text{past}(b_1, G)| = |\text{past}(b_2, G)|$ та $\deg^+(b_1) = \deg^+(b_2)$, то $\text{hash}(b_1) < \text{hash}(b_2)$ (рис. 2.1).

Визначення 1. Блок $b \in \text{Slot}(n)$ називатимемо хребетним, якщо він передусе решті блоків слота, тобто, якщо

$$\forall a \in \text{Slot}(n) (b <_{sl} a).$$

На рис. 2.2 та 2.3 показані визначені хребетні блоки.

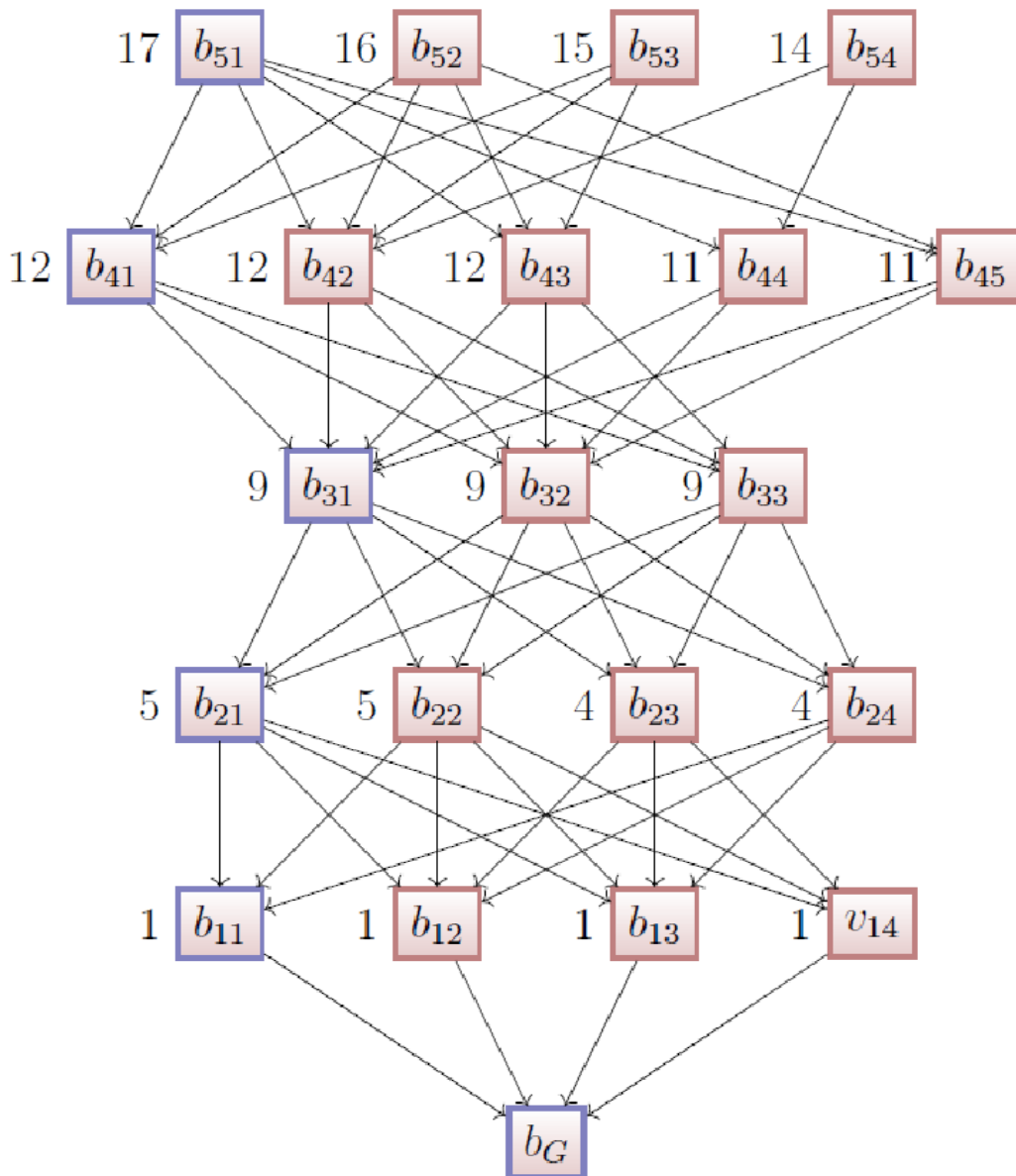


Рис. 2.2. Упорядкування блоків у слотах з індексами $|\text{past}(*, *)|$ у DAG мережі BlockDAG (хребетні блоки показані з синім кольором)

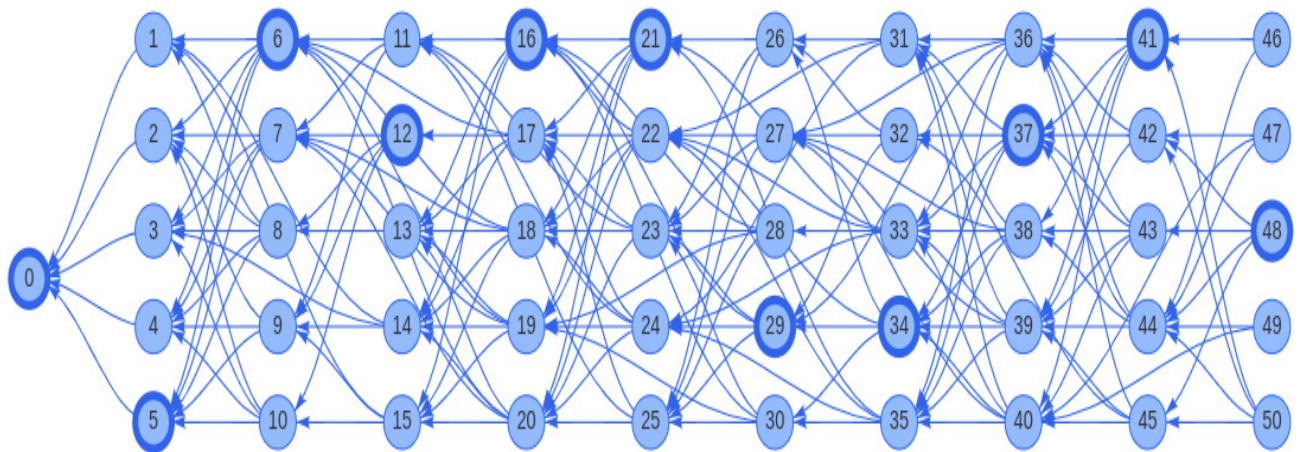


Рис. 2.3. Структура DAG створена за допомогою моделювання (вершини, виділені жирним шрифтом, є хребетними блоками)

У BlockDAG кожен блок посиляється (спрямованими ребрами) на попередні попередні утворені (батьківські) блоки. Іноді всю множину батьківських блоків називають «минулим» (past) даного блоку.

Висотою блоку називатимемо кількість всіх батьківських блоків, на які посиляється даний блок.

Визначення 2. Хребетний блок – це блок слота, який:

- 1) має найбільшу висоту,
- 2) якщо таких блоків декілька, то береться блок, який посиляється на більшу кількість попередніх блоків,
- 3) якщо таких блоків декілька, то береться той, у якого значення функції перемішування від хешу менше.

Оскільки кожен блок зберігає дерево Меркла, фактично хребетний блок зберігає в «упакованому» вигляді все «минуле» блоків даного слота. Тому можна передавати до координаційної мережі не всі блоки слота, а лише хребтні блоки. Це дозволяє зменшити трафік між мережею BlockDAG та координаційною мережею в n разів, де n – кількість блоків, утворених у слоті.

2.2.2 Методика застосування обміну інформацією на основі моделі хребетних блоків

Сформулюємо методику застосування обміну інформацією BlockDAG мережі з координаційною мережею на основі моделі хребетних блоків.

Крок 1. Визначається хребетний блок кожного нефіналізованого слота, вони впорядковуються по порядку їхнього прямування від меншого номера слота до більшого.

Крок 2. Вибирається перший хребетний блок.

Крок 3. Беруться нефіналізовані батьківські блоки цього блоку та впорядковуються за висотою (чим більше, тим раніше йде); якщо висота однакова, то блоки впорядковуються за кількістю посилань на попередні блоки (батьківські); якщо кількість посилань однакова, то блоки впорядковуються за найменшим хеш-кодом. Цей блок вставляється у список перед хребетним блоком.

4. Рекурсивно повторюється пункт 3 для вкладених батьківських блоків до того часу, доки прийдемо до того, що це батьківські блоки або вже фіналізовані, або у упорядкованому списку. Батьківські блоки вставляються до списку перед обраним блоком.

Крок 5. Повторюються кроки 3–4 всім хребетним блокам по порядку. Внаслідок цього з'являється впорядкована послідовність всіх блоків DAG.

2.3 Робота двошарової мережі на основі запропонованих моделей

2.3.1 Функціонування координаційної мережі

Дві запропоновані моделі дозволили сконструювати та промодельовати функціонування двошарової мережі із прискореною обробкою транзакцій. Мережа працює шляхом виконання наступних послідовних дій.

1) ведення реєстру утворювачів блоків – вузлів у мережі, які заблокували певну кількість монет для участі у мережі. Заблокована сума фіксована і однакова всім творцям блоків. Набір всіх творців позначатиметься як Cr ;

- 2) розподіл утворювачів–координаторів по комітетах та слотах;
- 3) призначення ролей Cr : утворювач, атестатор, агрегатор ;
- 4) отримання інформації про хребетні блоки хребта з BlockDAG;
- 5) голосування за хребетні блоки та дію над блоком відповідно до результатів голосування.

Передбачається, що всі члени координаційної мережі можуть аутентифікувати повідомлення друг від друга. Також передбачається, що всі повідомлення в кінцевому підсумку доставляються всім чесним учасникам і вони можуть синхронізуватися за час, що не перевищує T_{sync} . Вузли мережі, які не отримали своєчасне повідомлення, вважаються несправним.

Робота координаційної мережі починається із створення блоку генезису, який містить вихідний список творців блоків. Координаційна мережа працює за такою схемою.

- 1) На початку кожної епохи список комітетів N_C до наступної епохи генерується випадково для кожного слота.
- 2) На початку кожного слота агрегатори цих комітетів вибираються випадково з членів комітету.
- 3) На початку кожного слота випадково вибираються вузли від членів комітету до створення блоків.
- 4) На початку кожного слота кожен вузол координаційної мережі відправляє на підключені вузли BlockDAG мережну інформацію про епоху, слот, список творців блоків та інформацію про фіналізовані блоки мережі BlockDAG.
- 5) На початку кожного слота кожен вузол координаційної мережі отримує інформацію про хребетні блоки від підключеного до нього мережного вузла BlockDAG.

Фіналізований хребетний блок n -го слота позначатиметься як f_n . За визначенням припустимо $f_0 = b_G$, де b_G – блок генезису.

Опишемо процедуру створення блоку координаційної мережі,

припускаючи, що фіналізований блок хребта f_n вже обраний. Позначимо через $\sum_{n+s-1}$ список блоків до $(n+s-1)$ -го слота координаційної мережі та позначимо через B_i список блоків $(n+i)$ -го слота мережі BlockDAG.

1. Кожен член комітету надсилає повідомлення іншому комітету учасників зі списком видимих хребетних блоків, що розпочалися в $(n+1)$ слотах блок попереднього стану та підпис

$$m_{cr,1} = ((b_1, b_2, \dots, b_s), \sigma, cr),$$

де $b_i \in B_i$, $cr \in Cr$ – підпис та $\sigma \in \sum_{n+s-1}$ обирається відповідно до наступного правила:

а) якщо вилок немає, це останній коректно створений блок координаційної мережі;

б) якщо є вилок і найдовший ланцюжок, то це останній блок із найдовшого ланцюжка;

в) якщо є вилок та ланцюжки, породжені ними тієї ж довжини, то це раніший з останніх блоків вилок ланцюга;

2. Нехай послідовність $b = (b_1, b_2, \dots, b_k)$ є початок деякої послідовності $b = (b_1, b_2, \dots, b_s)$, опублікованої деяким утворювачем з r . Тоді вважаємо, що послідовність b отримує голос координатора з r .

В результаті виконання попереднього пункту можливі два варіанти:

а) Існує послідовність $b = (b_1, b_2, \dots, b_k)$, яка отримала більше $2/3$ голосів;

б) Жодна послідовність не набрала більше $2/3$ голосів. В цьому разі вважається, що $b = \emptyset$;

3. Кожен член комітету, який підтримує b , надсилає повідомлення $m_{cr,2} = (\bar{b}, \sigma, cr)$ іншим членам комісії;

4. Агрегатор Комітету формує повідомлення $m_{agr} = (\bar{b}, \sigma, (cr_1, \dots, cr_i))$ та відправляє його іншим агрегаторам комітетів;

5. Якщо більше $2/3$ комітетів отримують непусті послідовності, кожен агрегатор надсилає повідомлення $(\bar{m}, cr_i)(m, vl)$ іншому агрегатору. В іншому

випадку блок не публікується.

Перейдемо до правил прийому блоку до координаційної мережі:

1. Якщо правильно створений блок координаційної мережі отримано пізніше наступного слота, то він відправляється далі і додається в ланцюжок. У цьому випадку говоритимемо, що блок отримано вчасно;
2. Якщо правильно створений блок координаційної мережі виходить через один слот, він вирушає далі і додається до листа очікування. У цьому випадку ми говоритимемо, що блок отримано із запізненням;
3. Якщо блок, що належить аркушу очікування, посиляється на блок, створений вчасно чи із затримкою, він додається в ланцюг;
4. Якщо блок координаційної мережі отримано пізніше, ніж 2 слоти після створення, він видаляється і далі не відправляється.

2.3.2 Функціонування BlockDAG мережі

Перейдемо до опису того, як працює мережа BlockDAG.

На початку роботи мережі створюється BlockDAG блок генезісу, що містить службову інформацію про мережу. Блоки в мережі BlockDAG повинні містити хеші блоків, які не належать одному і тому ж слоту, в якому створюється блок і на які ніхто не посиляється в зоні видимості вузла при створенні блоку (немає блоків, що містять хеш цих блоків) та упорядкованому списку транзакцій.

Опишемо процедуру додавання нового блоку у вузол BlockDAG мережі. Коли отримано новий блок, вузол BlockDAG мережі перевіряє, чи був блок створено правильно:

1. Правильність слота створення та вузла (чи може вузол створити блок у цьому слоті);
2. Вміст блоку вірний чи ні;
3. Якщо посилання на попередні блоки вірні, перевіряється, що:
 - а) немає жодних посилань на кілька блоків, створених тим же вузлом у тому ж слоті;

б) відсутні посилання на некоректно створені блоки.

Опишемо процедуру впорядкування блоків у мережі BlockDAG.

Нехай задана послідовність $f_0, \dots, f_n$ фіналізованих хребетних блоків у графі G . Визначимо послідовність підграфів G_n графа G , що розширюється, пов'язану з цією послідовністю. Припустимо за визначенням, що G_0 – граф $V(G_0) = \{b_G\}$ і $E(G_0) = \emptyset$. Визначимо граничний граф G_n наступним чином:

$$V(G_n) = V(G_{n-1}) \cup \text{past } f_n \cup \{f_n\},$$

для будь-яких $a_1, a_2 \in V(G_n)$ ($a_1, a_2 \in E(G_n)$) і тоді, коли $(a_1, a_2) \in E(G)$.

Побудуємо лінійний порядок на G_n (рис. 2.4):

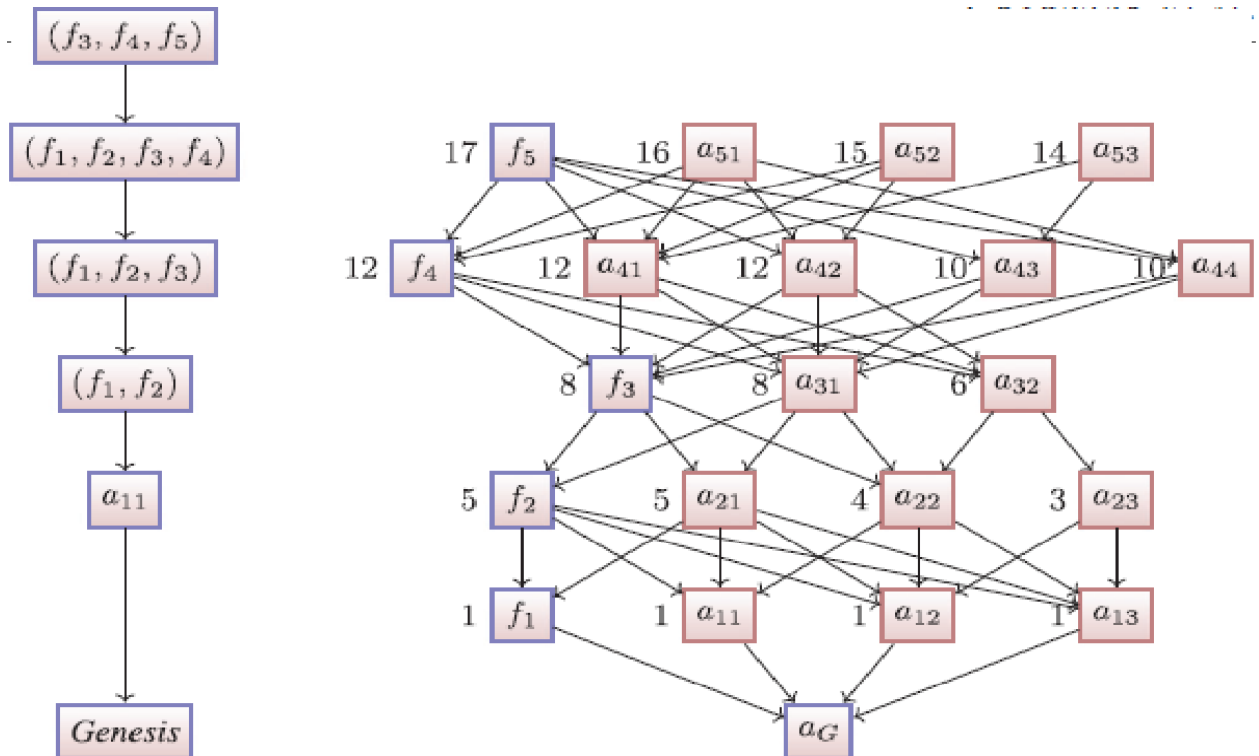


Рис. 2.4. Структура мереж Coordinating та BlockDAG з побудованим лінійним порядком. Фіналізовані хребетні блоки відзначені синім кольором

2.4 Розробка методу ймовірнісного консенсусу

2.4.1 Недоліки повного консенсусу

DLT може бути використані в децентралізованій системою цифрового обліку(наприклад блокчейн в Bitcoin). Безпека системи переважно залежить від

можливості вузлів завершувати транзакції через узгоджений протокол. Фіналізація – це процес підтвердження транзакцій та додавання їх до реєстру блокчейну, що забезпечує достовірність та незворотність транзакцій. Механізм консенсусу визначає, яким чином досягається фіналізація. Оскільки технологія блокчейна продовжує розвиватись, можуть бути розроблені нові методи консенсусу для покращення процесу фіналізації. Мережа Bitcoin, найвідоміший блокчейн у світі, має ймовірнісну концепцію завершеності: чим більше блоків додається до реєстру блокчейну після транзакції, тим менша ймовірність того, що транзакція буде скасована. Наприклад, вважається, що можливість скасування транзакції після шести доданих блоків становить менше 0,1%.

У блокчейні Ethereum 2.0 остаточний статус транзакцій визначається з урахуванням епох [91]. В даний час процес фіналізації займає близько 2 – 3 епох або 13 – 18 хвилин. Причина такої тривалості полягає в тому, що потрібно знайти баланс між децентралізацією, безпекою, загальним навантаженням на мережу, очікуваннями клієнтів і т.д. Це вважається придатним для багатьох варіантів використання в різних галузях, включаючи індустрію децентралізованих фінансів (DeFi). Однак у деяких бізнес-сценаріях, коли відбувається багато транзакцій і відносно невеликі суми, середній час очікування близько 15 хвилин може виявитися неприйнятним. Таким чином, є вагомі аргументи на користь фіналізації після одного слоту замість епох.

У двошаровій моделі, запропонованій у підрозділі 2.1, при повному консенсусі кожен атестатор протягом ери голосує за перший валідний блок попередньої ери. Якщо блок набрав більше $\frac{2}{3}$ голосів усіх атестаторів, то блок вважається фіналізованим, а, отже, ланцюжок хребетних блоків BlockDAG мережі від минулого фіналізованого блоку теж вважається фіналізованим і мережа, що відправляється в BlockDAG, остаточно затверджена. Якщо в епосі 32 слоти, а слот триває 4 секунди, транзакція пройде через $2 \cdot 32 \cdot 4 = 256$ секунд дорівнює 4 хвилинам 16 секундам. Як і в Ethereum, для деяких ситуацій потрібно знизити час фіналізації,

Одним із можливих рішень цієї проблеми є розробка протоколу

оптимістичного консенсусу, який забезпечує задану ймовірнісну завершеність, що визначається набором параметрів системи. Такий підхід дозволяє значно прискорити доопрацювання слотів, а згодом і самих блоків. Однак для підтримки масштабованості та продуктивності мережі необхідно йти на компроміс у питаннях безпеки.

2.4.2 Статистичне моделювання ймовірнісних процесів під час атестації

У процесі роботи координаційної мережі всі атестатори блокчейна діляться на 32 множини, що не перетинаються, відповідно до слотів, в яких вони працюють. У кожному слоті один із атестаторів має ексклюзивне право пропонувати блок, що містить список блоків DAG з транзакціями, які підлягають доопрацюванню, а інші атестатори з цього слота або наступних слотів можуть голосувати на підтримку цього блоку. У цьому підрозділі досліджено метод, згідно з яким слот вважатиметься таким, що проголосував позитивно.

Деякі атестати з якихось причин можуть не відправити свій голос за прийняття результатів слота. Такі учасники, що не голосують, становлять певну частку f від загального числа і випадковим чином розподіляються по слотам епох. Далі розглядатимемо лише збої, що сталися під час голосування. Таким чином, кожен атестатор може проводити валідацію один раз на епоху, а частка таких атестаторів обмежена значенням f . Консенсус PoS, який забезпечує завершення епохи, вимагає правила кваліфікованої більшості $f < 1/3$.

Пропонується така ймовірна оптимізація слотів. Кількість голосів, що підтримують результати, можна розглядати як випадкову величину, що має біноміальний розподіл з параметрами атестаторів n на слот і ймовірністю успіху $p=1-f$. На рис. 2.5 як приклад показані ймовірності позитивного числа голосів при $n=64$ та $p=2/3$.

Задамо граничне значення, при якому по слот прийматиметься позитивне рішення, $t=1/2$. При цьому ймовірність того, що чесні (не хибні) атестатори Y наберуть достатню кількість голосів, дорівнює

$$p(Y > t * n) = 1 - F_p(t * n) \approx 0,9957,$$

де F_p – біноміальна кумулятивна функція розподілу.

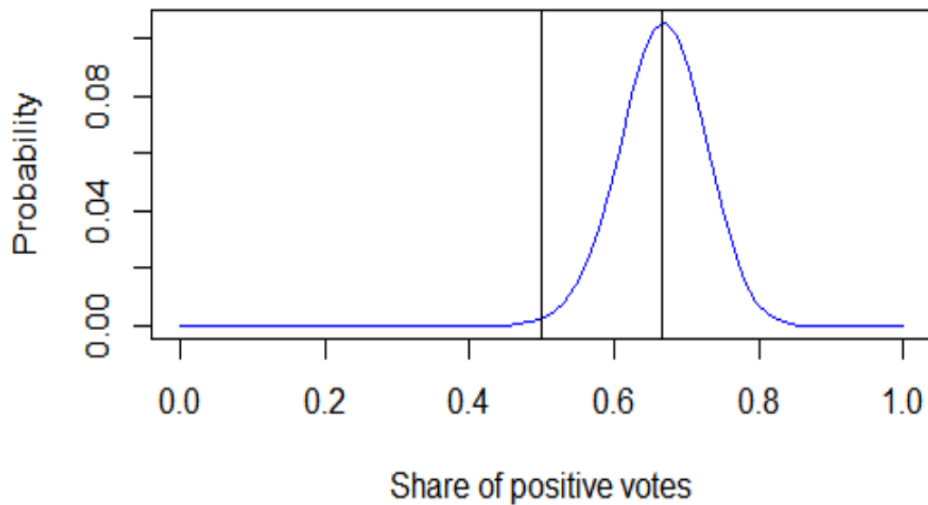


Рис. 2.5 Біноміальна функція щільності ймовірності з $n = 64$ та $p = 2/3$

Ця ймовірність збільшуватиметься зі збільшенням n . Наприклад, якщо кількість атестаторів на слот подвоїти, тобто. покласти $n=128$, воно перевищить 0,9999. З іншого боку, поріг $t=1/2$ не дозволить підтримувати кілька рішень у слоті, що призвело б до нерозв'язної суперечності в системі. Однак збільшення значення t є недоцільним. Наприклад, ймовірність того, що чесні виборці досягнуть «переважної більшості» (з $t = 2/3$) дорівнює 0,5235, що в середньому становить 16,75 слотів за епоху.

З практичної точки зору також цікаво оцінити ймовірність досягнення більшості шляхом змови зловмисних атестаторів в межах одного слота. Як найбільш несприятливий випадок можна припустити, що всі зловмисні учасники (частка яких $f < 1/3$) перебувають у змові. Очевидно, це додаткова подія до вищезгаданого, якщо знехтувати випадком, коли обидві групи (зловмисна та чесна) отримують однакову кількість голосів.

Розміри, розглянуті вище, стосуються випадку, коли частка всіх несправних мережевих атестаторів f залишається постійною по слотах. Однак на практиці при неповторній вибірці (32 вибірки, що не перетинаються) значення f може змінюватись, хоча й незначно, за досить великої загальної

кількості голосуючих. Для тестування було збудовано модель голосування за умови $f = 0,333$, яка імітує роботу координаційної мережі з різною кількістю n виборців на слот для одного мільйона епох. Результати, наведені в таблиці 2.1, відповідають теоретичним.

Таблиця 2.1. Середні значення слотів, які проголосували позитивно

n	64	128	256	2048	8192
$t = 1/2$	31.885448	31,998338	31,999999	32.000000	32.000000
$t = 2/3$	16.932351	15.898061	16.575013	16.355582	16.801351

На рис. 2.6 наведено загальний випадок для фіксованого числа атестаторів $n = 256$, коли параметри f та t змінюються в широких межах. При інших значеннях n форма 3D графіка залишається незмінною, але зі збільшенням кількості голосуючих стрибок на графіку поверхні стає різкішим. Результати моделювання підтверджують доцільність вибору мажоритарного правила за $t = 1/2$.

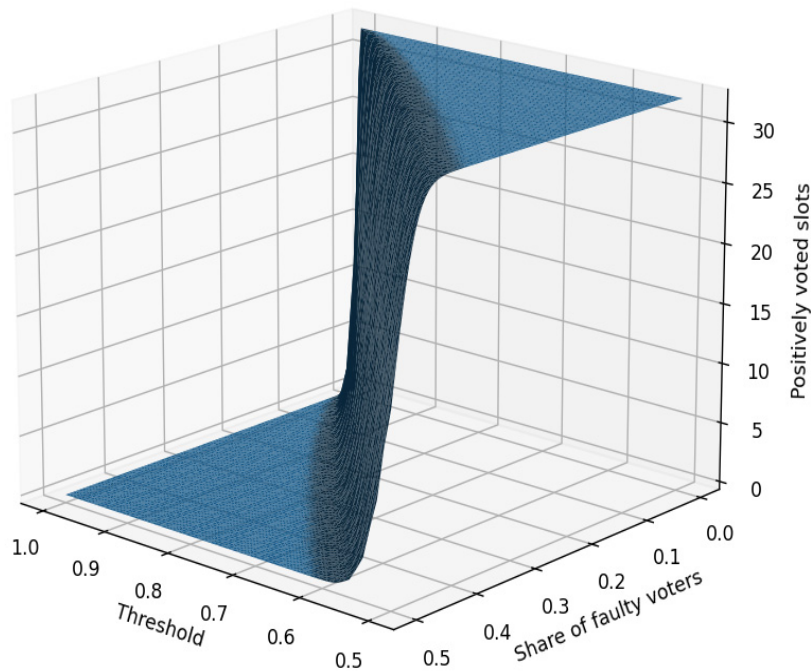


Рис. 2.6. Середні значення слотів, які проголосували позитивно,
з $n = 256$ атестаторами

Далі припустимо, що всі нечесні атестатори знаходяться в змові, і прикладають узгоджені зусилля, щоб доопрацювати слот, який не буде підтриманий переважною більшістю при голосуванні цілої епохи. Атака починається, коли лідером слота стає атестатор із числа змовників, який публікує свою пропозицію щодо доопрацювання. Для прийняття цієї пропозиції (оптимістичного доопрацювання) необхідно, щоб змовники згодом змогли отримати контроль над необхідною кількістю слотів k швидше, ніж чесні вузли, які підтримають «правильну» пропозицію, що конкурує.

Тут, як і вище, моделювалася різна кількість атестаторів на слот, які діють протягом мільйона епох. Частки епох, коли група змовників з $f = 1/3$ змогла дійти помилково–оптимістичного консенсусу, представлені в табл. 2.2.

Таблиця 2.2. Частки епох із помилковим оптимістичним консенсусом

n	64	128
$k = 1$	0,029812	0,000439
$k = 2$	0,000314	—
$k = 3$	0,000002	—

При інших значеннях n наведених у табл. 2.1, хибно–оптимістичного консенсусу не спостерігалось. У всіх випадках середня кількість оптимістично завершених блоків у координаційній мережі становила 20 – 21 на епоху, насамперед через нечесних лідерів (змовників), які виробляють помилкові блоки, які не можуть бути прийняті чесними атестаторами. Тому можна рекомендувати $k=2$, оскільки це дозволить блокам швидко досягати оптимістичної фіналізації із високим значенням достовірності.

2.4.3 Метод ймовірнісного консенсусу

Крок 1. Атестатори слота в попередньому слоті обмінюються хребетними

блоками, які бачать у BlockDAG. Автор координуючого блоку використовує цю інформацію, щоб вказати оптимальний ланцюжок хребетних блоків та підвищити шанси на те, що за нього проголосує більшість.

Крок 2. Атестатори голосують не лише за перший блок попередньої епохи, але й за останній блок координуючої мережі (в ідеальному варіанті за блок, створений у цьому ж слоті), в якому міститься ланцюжок хребетних блоків, яких він бачить у своєму BlockDAG вузлі та згоден із послідовністю.

Крок 3. Якщо початкова частина ланцюжка хребетних блоків повторилася k разів і за ці блоки проголосували більше половини учасників відповідних слотів, то цей ланцюжок є кандидатом на фіналізацію та відправляється до BlockDAG вузол на попередню фіналізацію.

Крок 4. Після закінчення ланцюжка хребетних блоків у мережі BlockDAG вона записується в блок координуючої мережі, і дані хребетні блоки вже не пропонуються далі як кандидати.

Моделювання даного методу формування консенсусу спільно з методом перемішування, викладеному нижче в розділі 3, показало, що при $k = 3$ дане впорядкування не зміниться з ймовірністю не менше 0,9. В результаті прогноз можливої фіналізації може бути отриманий за $3 \times 4 = 12$ секунд.

2.5. Дослідження шляхів підвищення стійкості протоколу PoS

В основу дослідження покладено завдання візантійської стійкості мережі. Візантійська стійкість (BFT) є властивістю системи, здатної протистояти класу відмов, що виникають через відоме завдання візантійських генералів [27,28]. BFT алгоритм забезпечує у розподіленій мережі досягнення консенсусу у разі, коли деякі вузли в мережі не відповідають або відповідають невірною інформацією. Метою механізму BFT є захист від збоїв системи шляхом використання колективного ухвалення рішень, мета якого зводиться до зменшення впливу несправних вузлів.

Нехай n – кількість координаторів (всіх учасників протоколу); c –

кількість комітетів на слот; m – кількість членів комітету . Якщо прийняти $n = 32$ mc , то виконується умова, що всі координатори беруть участь у роботі комітету один раз в епоху (як зазначено вище, вважаємо, що в кожній епосі 32 слоти). При створенні блоку спочатку учасники комітету обмінюються повідомленнями протоколу BFT, а потім агрегатори також обмінюються повідомленнями протоколу BFT.

Таким чином, загальна кількість повідомлень дорівнює:

$$M(c) = 2m(m - 1)c + 2c(c - 1) = 2 \left[\frac{n(m-1)}{32} - c(c-1) \right]. \quad (2.1)$$

При фіксованому n функція M залежить тільки від c , оскільки $m = n/(32c)$.

Збільшення частки чесних членів комітету. Однією з найважливіших властивостей протоколу BFT є те, що система складається з $3f+1$ вузлів, де f – максимальна кількість несправних вузлів, що вона може обробити [28]. Отже, згідно з протоколом, більше $2/3$ членів мають бути справними для належної роботи кожного комітету, і понад $2/3$ керівників комітету (агрегаторів) мають бути справними та повинні мати можливість подавати свій комітет на заключному етапі. Іншими словами, рішення може бути ухвалене як усередині комітету членами комітету, так і між комітетами (їх керівниками або агрегаторами) лише за наявності більше $2/3$ голосів за таке рішення. І тут відсоток більшості (*Share*), необхідний прийняття рішення, буде найменшим, якщо число утворювачів можна записати як $3f+1$, тобто. при поділенні цього числа на 3 залишок дорівнює 1. Приклади того, як *Share* змінюється в залежності від числа членів комітету (*Members*), наведено на рис. 2.7.

Із залежностей, що наведені вище, видно, що зі збільшенням кількості учасників комітету коливання відсотка більшості зменшуються. Тому, виходячи з цієї особливості протоколу BFT, ми за інших рівних умов рекомендуватимемо вибирати кількість комітетів як ціле число виду $3f+1$.

Розглянемо питання про значення c і m , при яких кількість повідомлень M , що передаються, буде мінімальною для зменшення загального навантаження на систему. Додатковою умовою є те, що m та c приймають лише цілі значення.

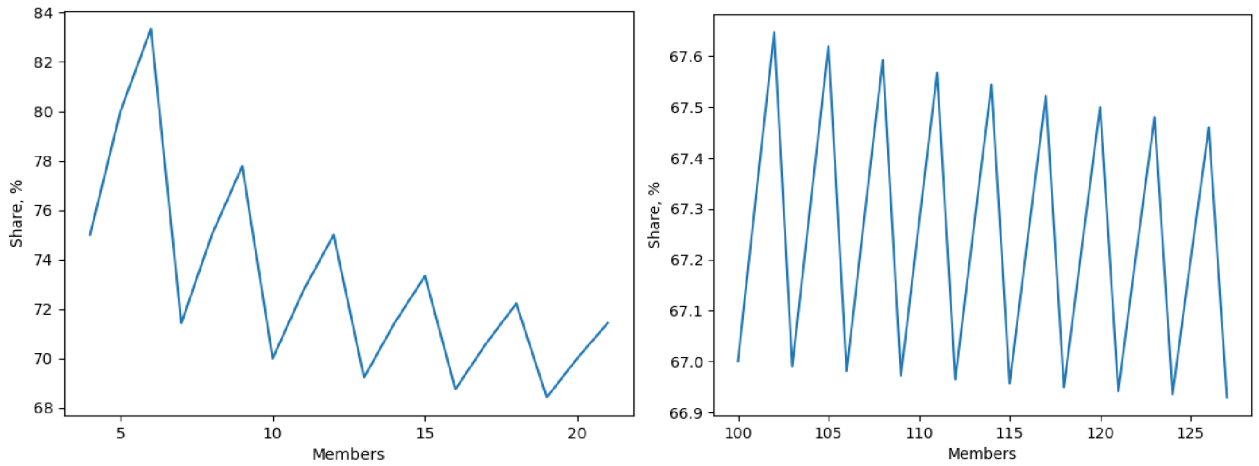


Рис. 2.7. Залежність відсотка більшості від членів комітету

Завдання можна звести до мінімізації функції (нехтуємо членом, що не містить c):

$$M'(c) = \frac{n^2}{1024c} + c^2 - c, \quad c \in \left[4, \frac{n}{128}\right]. \quad (2.2)$$

Тут через обмеження децентралізованого протоколу BFT на мінімальну кількість учасників ми припускаємо, що $c > 4$ і $m > 4$. З останньої нерівності, зокрема, випливає, що $c \leq \frac{n}{128}$. Крім того, ми припускаємо, що кількість координаторів досить велика.

Відкидаючи останній член M' , наближене значення точки мінімуму c_0 можна обчислити аналітично:

$$c_0 \approx \tilde{c}_0 = \sqrt[3]{\frac{n^2}{2048}}. \quad (2.3)$$

Можна отримати точнішу відповідь, ніж (2.3), з допомогою чисельних методів оптимізації чи інших математичних прийомів. Однак за досить великої кількості координаторів відповідь для наближеного і більш точного рішення буде однаковою. Рисунок 2.8 ілюструє випадок для $n=8192$.

Через обчислення ми отримали, що $c_0=32$. Сині та червоні лінії зображують безперервний та дискретний випадки числа членів комітету відповідно для порівняння.

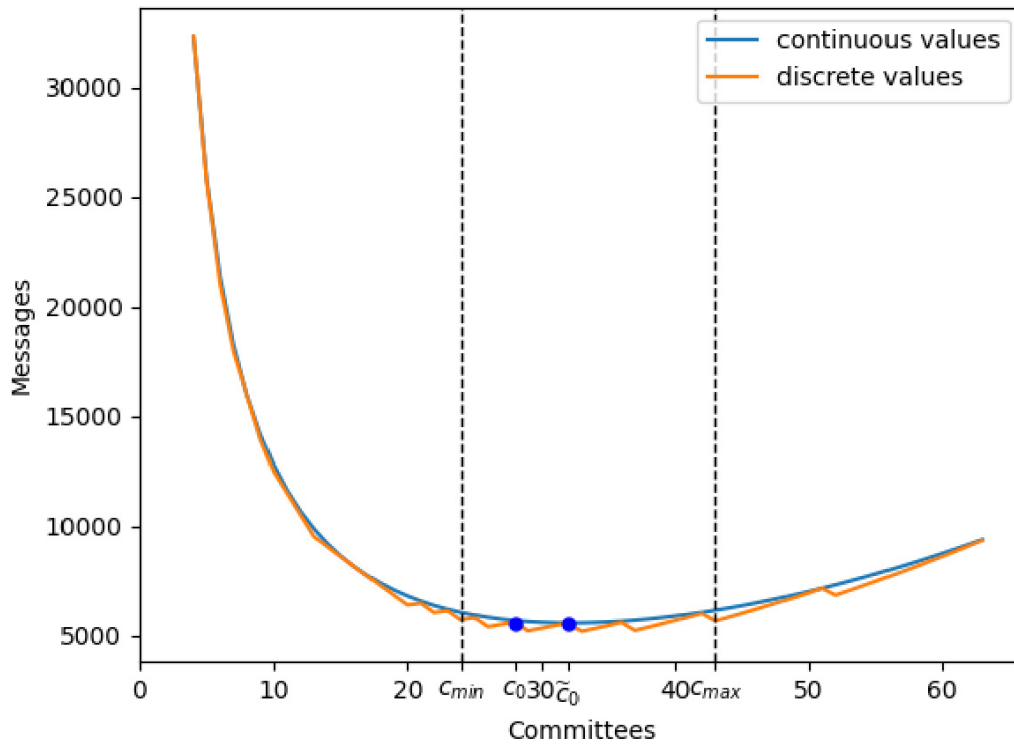


Рис. 2.8. Залежність кількості повідомлень, що пересилаються, від кількості комітетів

Дискретний випадок відповідає реальному, у якому кількість членів комітету є цілим числом і $32\text{ ms} \leq n$. Проте для знаходження мінімуму краще використовувати безперервний варіант. Зазначимо, що будь-яке значення з діапазону $[c_{\min}, c_{\max}]$ є практично прийнятним, оскільки збільшення кількості повідомлень при зростанні з цього інтервалу незначно (на 10%).

На основі властивостей протоколу BFT, можна рекомендувати вибирати кількість комітетів як ціле число виду $3f + 1$, найближче до c_0 .

Розглянемо питання мінімізації кількості несправних слотів. У зв'язку з тим, що деякі з Координаторів можуть виявитися несправними, необхідно з'ясувати, як це вплине на рішення у комітетах та остаточне рішення щодо виробництва блоків. У цьому випадку необхідно оцінити, яка максимальна частка несправних координаторів є прийнятною без зупинки або значного затягування процесу прийняття рішення. Завдання ставиться так. Для заданого числа координаторів знайти таку кількість комітетів і визначити число їх членів, при якому середня кількість несправних слотів в епоху $F(c)$ буде в

певному сенсі мінімальною. Несправними слотами вважаємо ті, в яких неможливо було прийняти блок. Очевидно, що кількість несправних слотів залежатиме від частки несправних координаторів та їхнього розподілу всередині комітетів. Завдання полягає в тому, щоб дати методику побудови розподілу, яка у середньому (у більшості випадків) даватиме менше несправних слотів, ніж інші розподіли.

Оптимізація розміру комітету. З допомогою імітаційного моделювання встановлено, що критичне значення загальної кількості несправних координаторів, у якому немає великої затримки виготовлення блоків, становить 20 %. У більшості випадків найкращі результати (найменше середнє число F) при фіксованій кількості Координаторів були досягнуті при зменшенні кількості комітетів та відповідному збільшенні їх членів (рис. 2.9).

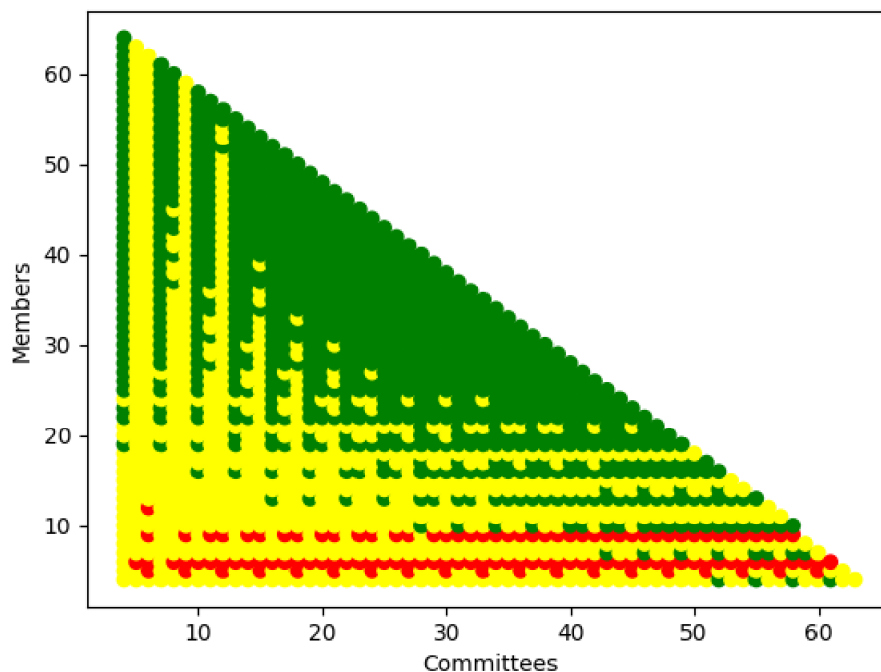


Рис. 2.9. Залежність кількості несправних слотів (F) від кількості комітетів та членів комітетів

Зелені крапки на рис. 2.9 позначають кількість отриманих несправних слотів до 11, жовті – від 11 до 21, а червоні – від 22. Однак, якщо кількість Комітету має вигляд $3f + 1$, найкращі результати спостерігаються навіть за малих кількості учасників (див. нижній правий кут на рис. 2.9).

На рисунку 2.10 показана кількість несправних слотів на епоху (F) з довірчим інтервалом 95% залежно від частки несправних координаторів у системі. На рис. 2.10 показано кількість несправних слотів на епоху (F) з довірчим інтервалом 95% залежно від частки несправних координаторів у системі. З рис. 2.10 видно, що з високих значення частки несправних координаторів (приблизно понад 20%) кількість комітетів має зменшуватися з відповідним зменшенням кількості членів комітету, і навпаки.

Алгоритм розподілу. Розглянемо випадок із часткою несправних координаторів менше 20%. Тоді суть алгоритму полягає в тому, щоб вибрати найменшу можливу кількість комітетів виду $3f + 1$, з числом членів комітету також виду $3f + 1$. При цьому кількість має бути не більше 127 у зв'язку з технічними обмеженнями на попередній аналіз і має давати невелику кількість несправних слотів.

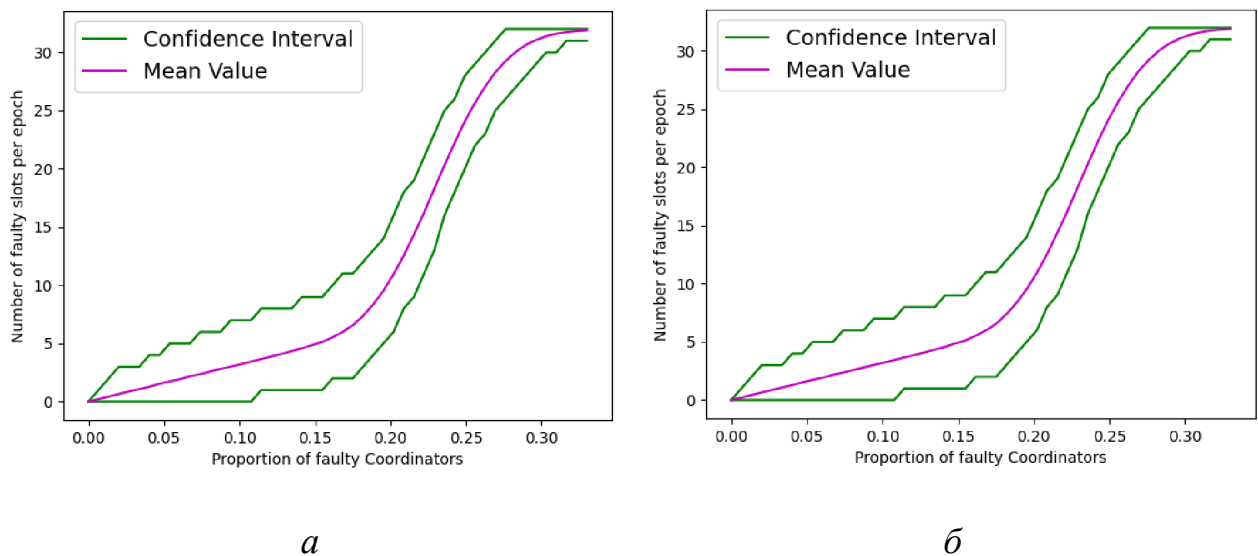


Рис. 2.10. Залежність кількості несправних слотів від частки несправних координаторів (a – 4 комітети, b – 64 комітети)

Щоб це сталося, покладемо:

n – загальна кількість координаторів у мережі;

$m_{\max} = 127$ – максимально можлива кількість членів Комітету.

Далі у формулах всі змінні – цілі позитивні числа, а результат при діленні округляється в меншу сторону.

1. Мінімальна кількість комітетів, до яких можуть бути включені всі члени:

$$c_{\min} = \frac{n-1}{32m_{\max}} + 1.$$

Як і раніше, вважаємо, що в епосі 32 слоти і всі координатори повинні брати участь у роботі комітетів один раз на епоху.

2. Кількість комітетів у формі $3f+1$:

$$c = 3 \left\lceil \frac{c_{\min} - 2}{3} + 1 \right\rceil + 1,$$

де число комітетів c – це мінімальна величина у формі $3f + 1$, більша, ніж $c_{\min}$.

Ця кількість комітетів використовуватиметься у кожному слоті.

3. Середня кількість членів комітету:

$$m_{\text{mean}} = \frac{n}{32c}.$$

4. Мінімальна кількість членів комітету у формі $3f + 1$: 22, де $m_{\min}$ – максимальна кількість членів у формі $3f + 1$, що не перевищує середню кількість членів. Інакше кажучи, у кожному з комітетів є координатори у кількості $m_{\min}$.

5. Кількість координаторів, які не закріплені за комітетами:

$$m_{\text{rest}} = n - 32m_{\min}c.$$

6. Кількість трійок, сформованих з цих координаторів :

$$t = \frac{m_{\text{rest}}}{3}.$$

7. Кількість Координаторів, що не входять до трійки:

$$m_{\text{next}} = m_{\text{rest}} \bmod(3).$$

8. Розподіл трійок за комітетами.

Випадок 1. Кількість перших слотів епохи, коли всі комітети будуть збільшені на 3 координатора:

$$s_{first} = t / c .$$

В наступному слоті кількість комітетів збільшується на 3 координатори:

$$c_{next} = t \bmod(c)$$

Також у цьому слоті можна збільшити ще один із комітетів іншими координаторами *nextMembers* .

Повинна виконуватись така рівність:

$$n = 32m_{mean}c + 3s_{first} + 3c_{next} + m_{next} .$$

Випадок 2. Мінімальна кількість комітетів з кожного слота, які отримають 3 додаткові координатори

$$c_{full} = \frac{t}{32} .$$

Крім того, певну кількість слотів має отримати ще по одному Комітету зі збільшеною кількістю Координаторів на 3 (наприклад, на початку епохи):

$$s_{extra} = t \bmod(32) .$$

Як і вище, ще один з комітетів може бути збільшений, наприклад, наступного слота координаторами *nextMembers*. Повинна виконуватись рівність:

$$n = 32m_{min}c + 96c_{full} + 3s_{extra} + m_{next} .$$

Результат:

– кількість комітетів у кожному слоті – мінімально можливе значення у формі $3f+1$;

– у всіх комітетах кількість членів (також у формі $3f + 1$) становить $m_{min} + 3 m_{min}$, за винятком, можливо, одного комітету, в якому на 1 або 2 члени більше ніж m_{min} ;

– кількість комітетів із «збільшенням» на 3 члени складає t . Такі комітети можуть розподілятися по слотах різними способами.

Щоб проілюструвати викладену методику, розглянемо такий приклад. З технічної точки зору, деякі проблеми виникають, коли кількість Worker'ів має занадто мало дільників, наприклад, якщо це просте число. Для прикладу нехай

у системі буде 8191 координаторів. Мінімальна кількість комітетів $c_{\min} = 3$.

Мінімальна кількість комітетів у формі $3f + 1$, яка не менша, ніж $\minCommittees$: $c = 4$. Кожен комітет матиме в середньому: $m_{\text{mean}} = 63$. Тоді, представляючи число членів комітету як $3f + 1$, отримуємо:

$m_{\min} = 61$. Кількість координаторів, які не увійшли до складу комітетів:

$m_{\text{rest}} = 383$. Для того, щоб додати їх до інших комітетів, не порушуючи форми $3f + 1$, решту координаторів ділимо на трійки.

$t = 127$. У результаті $m_{\text{next}} = 2$.

Далі розглянемо 2 стратегії розподілу решти m_{rest} Координаторів по комітетам.

Стратегія 1. Визначити кількість слотів епохи, коли всі комітети матимуть кількість координаторів, рівну $m_{\min} + 3$:

$$s_{\text{first}} = 31.$$

У наступному слоті збільшуємо комітети на 3 координатори:

$$c_{\text{next}} = 3.$$

У четвертому комітеті до цього слоту додаємо інших координаторів.

Стратегія 2. Визначити мінімальну кількість комітетів у кожному слоті, за якого всі вони матимуть кількість координаторів $m_{\min} + 3$:

$$c_{\text{full}} = 3.$$

Підраховуємо кількість слотів, які отримують ще один комітет, кожен із збільшенням кількості координаторів на 3:

$$s_{\text{extra}} = 31.$$

Проведено імітаційне моделювання з метою з'ясувати відмінності в результатах двох випадків розподілу координаторів, описаних вище.

На рис. 2.11 наведено залежність кількості несправних слотів від кількості Координаторів для відсотка несправних координаторів 20%.

З рисунку 2.11 видно, що середня кількість несправних слотів не перевищує 13 при відсотку несправних Координаторів 20%.

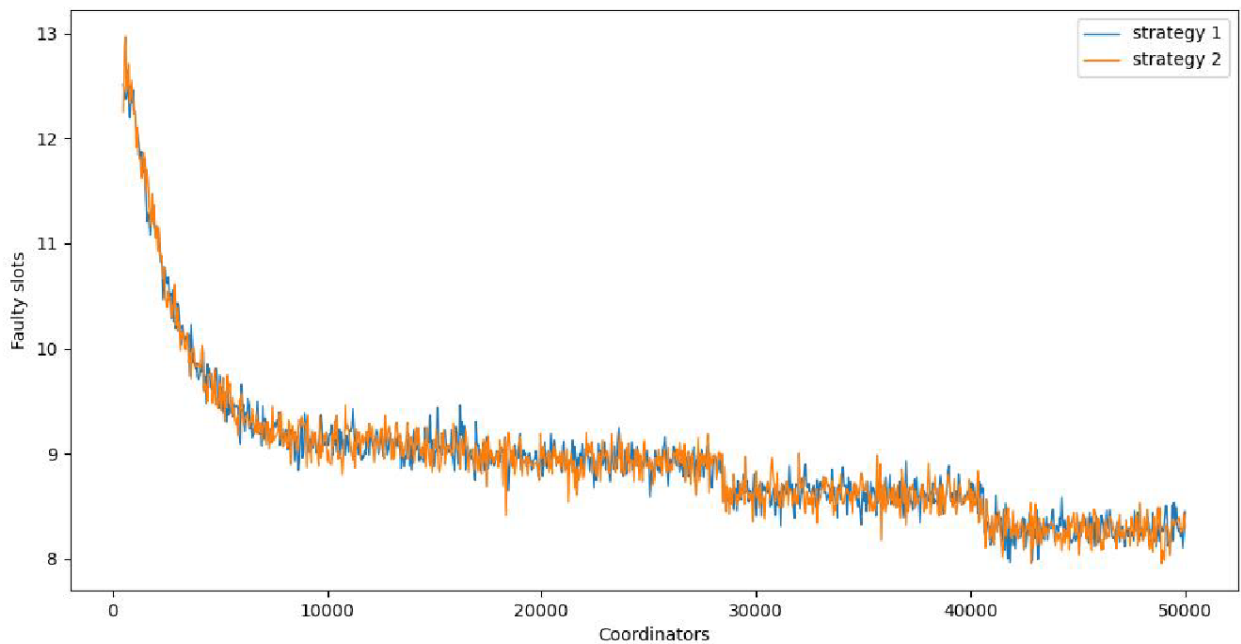


Рис. 2.11. Залежність кількості несправних слотів від кількості координаторів

Далі розглянуто мінімізацію непродуктивного сервісного трафіку.

Питання розподіл координатора для мінімізації кількості несправних слотів $F(c)$ розглянуто вище. Однак при цьому слід зчитувати та надсилати кількість повідомлень (X.1), по можливості зводячи його до мінімуму. Отже, при оптимізації протоколу консенсусу ми маємо два цільові критерії мінімізації – кількість відправлених службових повідомлень $M(c)$ та математичне очікування кількості помилкових слотів в епоху $F(c)$.

Зроблено спробу уникнути евристичних зважених згорткових підходів до вирішення цієї багатокритеріальної задачі оптимізації, щоб отримати суворіше рішення. Тому розглянуто поняття корисної інформації, яка безпосередньо корелює із кількістю отриманих блоків. У свою чергу ця кількість блоків може бути представлена математичним очікуванням кількості блоків, прийнятих за епоху.

Зазначимо, що навіть якщо в якомусь слоті рішення не ухвалювалося, службова інформація надсилалася в повному обсязі для досягнення консенсусу. Таким чином, слід розрахувати відношення обсягу службового трафіку

протягом доби до очікуваної кількості успішних консенсусів (тобто справних слотів). Це співвідношення покаже очікуваний службовий трафік на успішний блок:

$$P(c) = \frac{M(c)}{32 - F(c)}. \quad (2.4)$$

Очевидно, що чим нижче це значення, тим вища ефективність протоколу консенсусу. Таким чином, ми можемо розглядати це завдання цілісної оптимізації як однокритеріальне:

$$\min_c P(c) = P(c^*).$$

Для очікуваного числа Worker'ів розмірність завдання невелика. Це дозволяє визначити оптимальне рішення шляхом моделювання варіантів розподілу. Розглянемо приклад багатокритеріальної задачі, описаної вище. Нехай $n = 8192$, а відсоток несправних координаторів становить 20%. Побудуємо залежність (2.4) очікуваного службового трафіку на успішний блок від кількості комітетів із зазначеними вище параметрами (рис. 2.12).

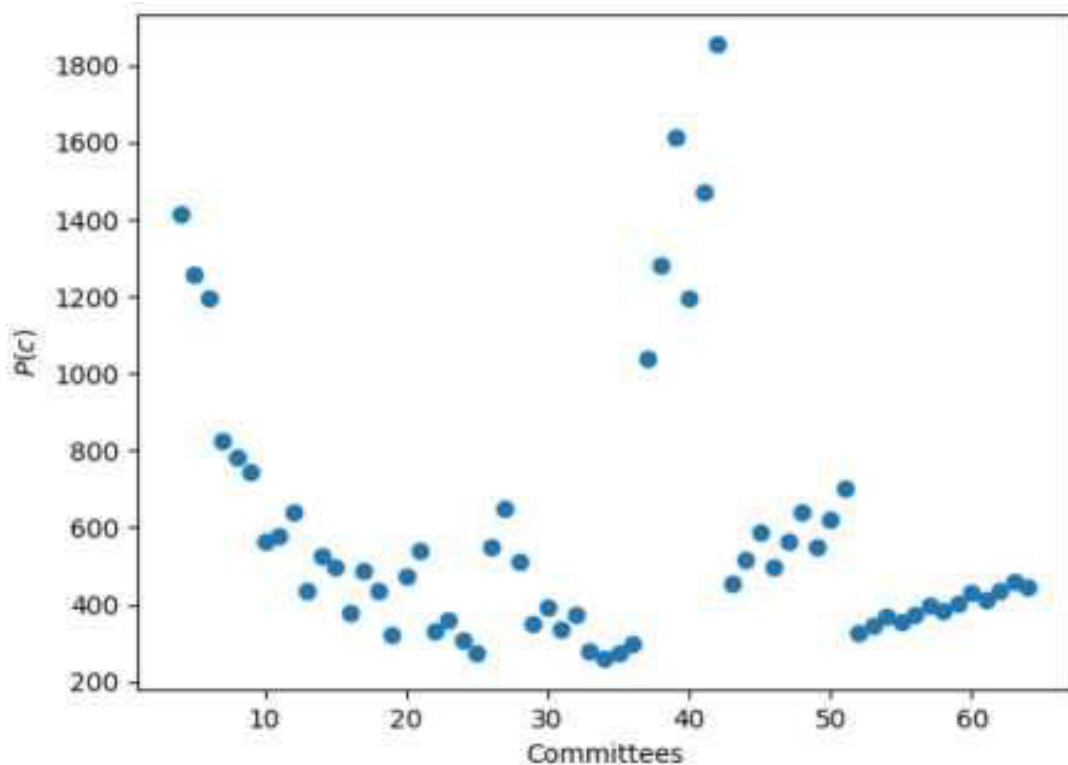


Рис. 2.12. Залежність очікуваного службового трафіку на успішний блок (P) від кількості комітетів

В результаті імітаційних експериментів було отримано, що в точці мінімуму кількість комітетів дорівнює $c^* = 34$, а кількість членів комітетів дорівнює 7. Такий результат легко пояснити тим, що і кількість комітетів, і кількість членів мають форму $3 + 1$, а кількість повідомлень, як показано вище, в цей момент близька до мінімуму.

Важливо, що у реальних сценаріях кількість і частка несправних вузлів можуть змінюватися з часом. У таких випадках співвідношення, отримані вище, все ще можуть використовуватися для визначення оптимальних розмірів комітету та членів комітету, які дають мінімальне значення функції. Крім того, наведена методика оптимізації може застосовуватися до широкого спектру консенсусних протоколів, у яких використовуються випадково обрані комітети для підпису блоків та перевірки. Його адаптованість та універсальність роблять її цінним інструментом для дослідників та розробників, що працюють у галузі розподілених систем.

Модель BlockDAG дозволяє розмішувати кілька Worker'ів на одному фізичному вузлі (сервері). Такий загальний хостинг творців блоків ще більше підвищує ефективність використання ресурсів сервера. Однак через ймовірнісного розподілу ролей Worker'ів усередині слота може виникнути ситуація, коли в якості виробників блоків одночасно призначаються кілька Worker'ів одного фізичного сервера (назвемо їх близнюками), в одному слоті основної мережі, що має структуру DAG. У цьому пропорційно зростає обчислювальна навантаження сервер. В результаті відповідні блоки можуть з'явитися із запізненням. Затримка створення та розподілення блоків не має вирішального значення для консенсусу мережі. Однак, такі сплески навантаження та відповідні затримки можуть вплинути на миттєву ефективність мережі, що небажано.

Було побудовано імітаційну модель для оцінки ймовірностей таких небажаних подій. Результати експериментів, наведені у таблиці 2.3.

Рядки таблиці відповідають кількості мережеских вузлів (серверів), а стовпці – кількості робочих процесів кожному вузлі. Оскільки ймовірність

загрози швидко зменшується зі збільшенням кількості вузлів, ми розглянули лише випадки, коли є від 32 до 1024 вузлів. Три блоки таблиці відповідають виникненню 2, 3 та 4 близнюків відповідно. Імовірність появи понад 4 близнюків мізерно мала порівняно з внеском 2 близнюків (див. табл. 2.3). Очевидно, що найімовірніша ситуація, коли лише два Worker'и на вузлі одночасно створюють блоки. Чим більше таких вузлів, тим менше за інших рівних умов ймовірність такої події одного конкретного вузла. У той же час, розгортання потужних серверів з великою кількістю Worker'ів значно збільшує ймовірність небажаної події. Однак із практичної точки зору таке збільшення можна вважати прийнятним, адже збільшуючи кількість Worker'ів на 16, ми отримуємо лише дворазове збільшення ймовірності.

Таблиця 2.3 Імовірність кількості близнюків в залежності від кількості вузлів і Worker'ів на вузол

Workers	2 twins					3 twins				4 twins			
	2	4	8	16	32	4	8	16	32	4	8	16	32
32	9.5e-02	1.4e-01	1.6e-01	1.7e-01	1.7e-01	1.5e-03	2.5e-03	3.1e-03	3.5e-03	3.0e-06	1.3e-05	2.1e-05	2.5e-05
64	4.7e-02	7.0e-02	8.0e-02	8.6e-02	8.8e-02	3.7e-04	6.4e-04	7.9e-04	8.8e-04	3.7e-07	1.6e-06	2.5e-06	3.2e-06
128	2.4e-02	3.5e-02	4.1e-02	4.3e-02	4.5e-02	9.2e-05	1.6e-04	2.0e-04	2.2e-04	4.8e-08	1.7e-07	3.3e-07	3.9e-07
256	1.2e-02	1.8e-02	2.0e-02	2.2e-02	2.3e-02	2.3e-05	4.0e-05	5.0e-05	5.6e-05	8.2e-09	2.7e-08	4.2e-08	5.2e-08
512	5.9e-03	8.8e-03	1.0e-02	1.1e-02	1.1e-02	5.8e-06	1.0e-05	1.3e-05	1.4e-05	2.0e-10	3.2e-09	5.7e-09	6.2e-09
1024	2.9e-03	4.4e-03	5.1e-03	5.5e-03	5.9e-03	1.4e-06	2.5e-06	3.2e-06	3.9e-06	5.9e-11	4.9e-10	7.3e-10	8.5e-10
Nodes 1													

На рис. 2.13 представлений підсумковий графік залежності математичного очікування сплесків обчислювального навантаження на сервер в залежності від кількості серверів і розгорнутих на них Worker'ів.

При фіксованій невеликій кількості Worker'ів краще робити багато легких серверів з невеликою кількістю воркерів. Зокрема, для 1024 Worker'ів ймовірність розриву навантаження варіюється від 0,6 до 17%.

Нижнє значення відповідає 512 легким серверам всього із двох Worker'ів. При цьому якщо кількість серверів досягає 500 і більше, ймовірність подвійного сплеску обчислювального навантаження стає близько одного відсотка для будь-якої кількості Worker'ів.

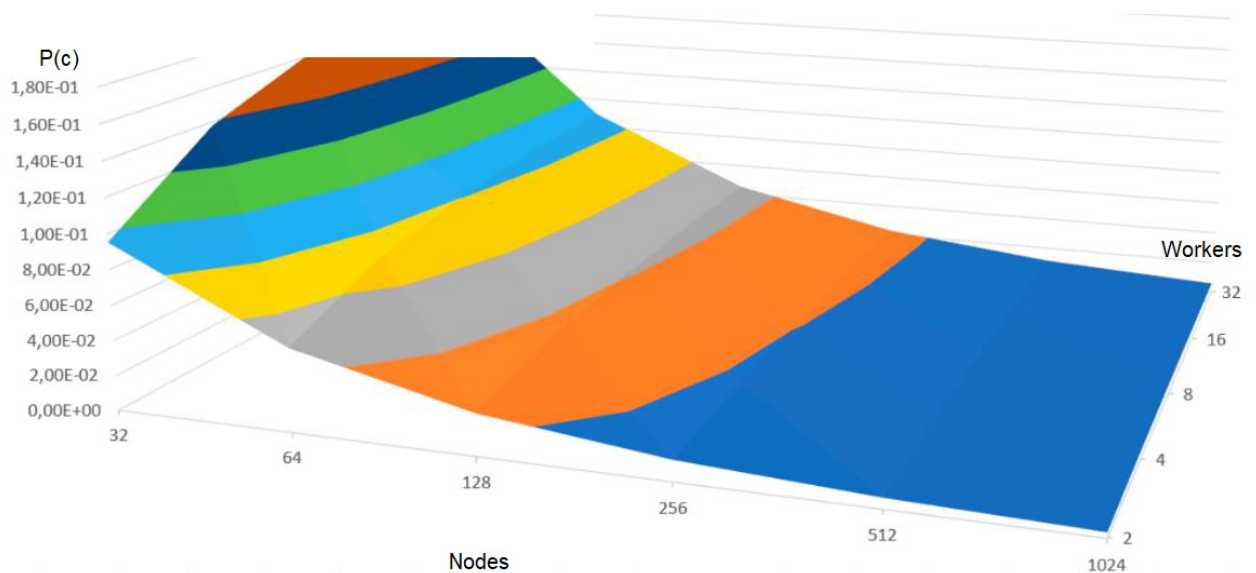


Рис. 2.13. Загальна ймовірність сплесків навантаження на вузли в залежності від кількості вузлів та кількість Worker'ів на вузол

Далі зі зростанням кількості серверів ймовірність сплеску пропорційно зменшується. Тому на початковому етапі зростання двошарової мережі доцільно використовувати малопотужні сервери з невеликою кількістю розгорнутих Worker'ів. Це дозволяє широко використовувати недорогі хмарні рішення і навіть малопотужні комп'ютери, підвищуючи при цьому рівень децентралізації.

2.6 Висновки до другого розділу

У другому розділі дисертаційної роботи досліджено шляхи підвищення швидкості обробки транзакцій та масштабованості розподіленого реєстру.

Для прискорення обробки транзакцій з високим рівнем масштабованості запропоновано двошарову модель побудови системи. Основна мережа, призначена для формування та розповсюдження блоків, виконана за технологією BlockDAG (шар 1). Операції, пов'язані з верифікацією блоків на підставі протоколу консенсусу, винесені до координаційної мережі, що працює за технологією традиційного консенсусу (шар 2). Згідно із запропонованою моделлю всі вузли, що входять до системи і звані Worker'ами, отримують дві

ролі – Утворювачів блоків в основній мережі та Координаторів у координаційній мережі. Таким чином розділені функції шарів мережі. В основній мережі виконуються функції формування блоків та їх поширення, що потребують операцій із довгими блоками. Побудова основної мережі за технологією BlockDAG суттєво прискорює процес попередньої обробки транзакцій за рахунок асинхронного формування блоків. У координаційну мережу винесено операції з формування консенсусу відповідно до протоколу Proof of Stake, що вимагають великої кількості пересилань відносно коротких повідомлень. Тому робота координаційної мережі організована відповідно до традиційного блокчейн-протоколу, який виконується простіше, ніж BlockDAG.

Сформульовано перший пункт наукової новизни:

– Вперше розроблено двохарову модель розподіленого реєстру, що поєднує в собі основну мережу, побудовану за технологією BlockDAG, та координаційну мережу, що базується на традиційній блокчейн технології, що дозволило підвищити швидкість обробки транзакцій при суттєвому масштабуванні системи.

Введено поняття хребетного блоку слота. Хребетний блок – це блок слота, який:

- 1) має найбільшу висоту,
- 2) якщо таких блоків кілька, то береться блок, який посилається на більшу кількість попередніх блоків,
- 3) якщо таких блоків кілька, то береться той, у якого значення функції перемішування від хешу менше.

Оскільки кожен блок зберігає дерево Меркла, фактично хребетний блок зберігає в «упакованому» вигляді все «минуле» блоків даного слоту. На цій підставі побудовано модель скороченого обміну інформацією між основною мережею та координаційною мережею, що зменшує інформаційне навантаження на систему в цілому та підвищує швидкість обробки транзакцій. Розроблено методику застосування обміну інформацією BlockDAG мережі з координаційною мережею на підставі моделі хребетних блоків.

Сформульовано другий пункт наукової новизни:

– Набула подальшого розвитку модель скороченого обміну інформацією між BlockDAG мережею та координаційною мережею на основі поняття хребетного блоку, що дозволило скоротити обсяг трафіку між мережею BlockDAG та координаційною мережею і тим самим зменшити навантаження на систему загалом та підвищити швидкість обробки транзакцій.

На підставі двох запропонованих моделей сформульовано узгоджену послідовність виконання операцій в основній та координаційній мережах.

Проведено дослідження шляхом статистичного моделювання можливостей скороченої процедури ймовірнісного консенсусу у двошаровій моделі. Встановлено, що якщо початкова частина ланцюжка хребетних блоків повторилася k разів і за ці блоки проголосували більше половини атестаторів – учасників відповідних слотів, то цей ланцюжок є кандидатом на фіналізацію. Показано, що при $k = 3$ дане впорядкування не зміниться з ймовірністю щонайменше 0,9. В результаті прогноз можливої фіналізації може бути отриманий за $3 \cdot 4 = 12$ секунд, в той час як очікування повного консенсусу становить 4 хвилини 16 секунд.

Сформульовано третій пункт наукової новизни: Вперше розроблений метод прискореного PoS консенсусу в двошаровій моделі розподіленого реєстру на основі k -кратної перевірки повторення початкової частини ланцюжка хребетних блоків і голосування за ці блоки більше половини учасників відповідних слотів, що дозволило зменшити час отримання рішення до декількох секунд із ймовірністю, не меншою 0,9.

Розроблена методика визначення оптимального розміру комітету та розподілу Worker'ів залежно від їх загальної кількості в мережі та очікуваної помилкової частки. Проведені імітаційні експерименти показали, що оптимальна кількість як комітетів, так і членів відповідає схемі $3f + 1$, що відповідає мінімальній кількості необхідних повідомлень. Отримані результати моделювання показали значне покращення продуктивності системи та зниження навантаження на мережні вузли.

РОЗДІЛ 3

ДОСЛІДЖЕННЯ БАЗОВИХ ПРИНЦИПІВ ПОБУДОВИ ФУНКЦІЙ
ПЕРЕМІШУВАННЯ НА ОСНОВІ НАЙПРОСТІШИХ ЛІНІЙНИХ І
НЕЛІНІЙНИХ ВІДОБРАЖЕНЬ

3.1 Аналіз потенційних вразливостей при виконанні протоколу Proof of Stake у координаційній мережі

При формуванні PoS консенсусу квазівипадковим чином у кожному слоті епохи відбувається формування складу комітетів та ролей координаторів у них (утворювач блоків, атестатор, агрегатор). Далі атестатор підписує останній, на його думку, валідний блок у слоті агрегатора, що поєднує результати атестації одним підписом. При цьому не виключені атаки маніпуляцією двох видів – як на формування складу комітетів, так і на вибір зі списку координаторів як атестатора, так і агрегатора. Тому важливим засобом захисту від таких можливих маніпуляцій є розробка методу квазівипадкового перемішування довільної числової множини, що реально гарантує захищеність від подібних маніпуляцій. Крім того, ще в класичній монографії [97], а пізніше в [98] були відзначені серйозні недоліки в широко застосовуваних комп'ютерних алгоритмах псевдовипадкового вибору. Як перший, так і другий види атак можуть розглядатися як потенційні вразливості, що не виключають можливих маніпуляцій при псевдовипадковому виборі. Тому важливим засобом захисту від таких можливих маніпуляцій є розробка ефективного методу квазівипадкового перемішування довільної числової множини, що реально гарантує підвищення криптозахищеності системи. Розділ присвячений дослідженню можливості застосування у комп'ютерній криптографії найпростіших положень теорії нелінійних дискретних динамічних систем, основною характеристикою яких є хаотичність. Основна проблема використання хаотичних систем у комп'ютерних обчисленнях полягає в тому, що кількість усіх станів у комп'ютері скінченне, отже, комп'ютерні моделі хаосу є лише апроксимацією справжньої хаотичної поведінки, а кожна

траєкторія апроксимованої системи є періодичною. З математичної точки зору шифрування в інформаційних системах є процесом перетворення простору кінцевих повідомлень, що аналогічно фазовому простору в теорії динамічних систем. Функція перемішування здійснює таке шифрування. Основні вимоги до функції перемішування – відсутність колізій, тобто бієктивність відображення, хороші дифузійні властивості, і, крім того, зворотне перетворення має бути не складнішим за пряме.

3.2 Постановка задачі побудови функцій перемішування

Криптографічні функції використовуються у завданнях забезпечення інформаційної безпеки. Існує велика кількість різноманітних функцій, які застосовуються в криптографії, що засновані на різних алгоритмах їх побудови. Необхідність реалізації захисту в системах з обмеженими обчислювальними ресурсами призводить до необхідності розробки нових криптографічних методів та алгоритмів. Основними властивостями функцій у них є властивості розсіювання і перемішування [99].

З математичної точки зору шифрування в інформаційних системах часто представляється як реалізація різних перетворень $G: X \rightarrow X$ на множині кінцевих повідомлень $X = \{x_1, \dots, x_k\}$. Вважаємо, що $x_1, \dots, x_k$ різні m -значні цілі позитивні числа, що кодують можливі повідомлення, де $k < 10^m$ та $x_j < 10^m$, $j = 1, \dots, k$. При цьому перетворення G повинні мати властивість відсутності колізій, тобто $G(x_j) = G(x_i) \Leftrightarrow x_j = x_i$. Крім того, зворотне перетворення має бути не складніше прямого. Якщо $k = 10^m - 1$, то функція $G(x)$ називається функцією повного перемішування чи функцією перестановки. В цьому випадку $X = \{1, 2, \dots, 10^m - 1\}$.

Практична реалізація функцій перемішування при великих значеннях m вимагає подолання значних складнощів, які пов'язані з описом та завданням параметрів перемішування, розробкою математичного та програмного

забезпечення, розвитку нових математичних моделей та методів.

Одним із найпоширеніших математичних методів, що використовуються для побудови функцій перемішування, є методи нелінійної динаміки [100-103]. Це можна пояснити тим, що у криптографії, й у нелінійних системах здійснюється нелінійне перетворення простору вихідної інформації (фазового простору). З одного боку, це перетворення повністю визначено, з іншого боку, воно має аперіодичну ергодичність і надзвичайну чутливість до початкових умов [104, 105], що дозволяє згенерувати псевдовипадкову хаотичну послідовність, яка має характеристики, що збігаються з основними вимогами криптографії. Структура динамічних систем із дискретним часом відносно проста, комп'ютерна реалізація зручна, швидкість розрахунків висока. Проте, значення елементів хаотичної послідовності у фазовому просторі розподілені нерівномірно, тобто недостатньо перемішуються, що спричиняє ризики зниження безпеки [106].

З іншого боку, функції перестановки, що засновані на лінійних операторах (можливо, з додатковою операцією порівняння по модулю), елементи фазового простору добре перемішують [107]. Вперше такі функції перестановки сконструював і почав використовувати Л. Хілл [108,109]. Основними перевагами шифру Хілла є стійкість до повного перебору, навіть за відносно невеликих розмірів ключа, простота та швидкість застосування. Однак є істотний недолік, притаманний всім криптографічним функціям з вираженими властивостями лінійних операцій: вразливість до атак по обраному відкритому тексту. У зв'язку з цим для збільшення криптостійкості лінійні шифри повинні бути доповнені будь якими нелінійними операціями. Комбінування лінійних операцій, як у шифрі Хілла, з нелінійними призвело до створення підстановочно-перестановочних мереж (наприклад, мережа Фейстеля [110]). В даний час методи конструювання функцій перемішування, в яких використовуються суперпозиції лінійних та нелінійних операцій, як і раніше, активно використовуються [111].

Основне завдання, яке розв'язується в цьому розділі – доведення можливості та побудова функцій перемішування на основі найпростіших лінійних і нелінійних відображень. Зокрема, в якості лінійного відображення пропонується використовувати оператор перестановки, а нелінійного – найпростіше нелінійне відображення, класичний Tent [112]. Іншими завданнями є проведення кореляційного аналізу побудованих відображень, аналізу чутливості, аналізу середніх довжин циклів цих відображень, розгляду можливості узагальнення алгоритмів побудови функцій перемішування та застосування їх до шифрування образів.

Далі буде використовуватися класичне відображення Tent [112], яке задається функцією

$$f(x) = 2(1/2 - |x - 1/2|) = \begin{cases} 2x, x \leq 1/2, \\ 2(1-x), x > 1/2, \end{cases} [0,1] \xrightarrow{f} [0,1],$$

та відображення q -Tent (q - непарне число)

$$f_q(x) = 2(q/2 - |x - q/2|) = \begin{cases} 2x, x \leq q/2 \\ 2(q-x), x > q/2, \end{cases} [0,q] \xrightarrow{f_q} [0,q].$$

Ці відображення мають ідентичні властивості (відрізняються одне від одного розтягуванням/стисненням), пов'язані між собою співвідношеннями:

$$f_1(x) = f(x),$$

$$f_q(x) = q f(x/q).$$

Відзначимо деякі важливі властивості відображення Tent.

Властивість 1. Якщо $\frac{p_1}{q} = f\left(\frac{p}{q}\right)$ (q - непарне число), то однозначно

відновлюється чисельник p , тобто, $p = \frac{p_1}{2}$, якщо $\frac{p_1}{2}$ - парне число, або $p = q - \frac{p_1}{2}$,

якщо $\frac{p_1}{2}$ - непарне число.

Отже, якщо p парне число, то прообраз однозначно відновлюється за образом.

Властивість 2. Множина правильних дробів з парними чисельниками та непарними знаменниками інваріантна щодо відображення Tent; на цій множині відображення Tent оборотне.

Властивість 3. Якщо p_1 и p_2 парні числа, то $f_q(p_1) = f_q(p_2)$ тоді і тільки тоді, коли $p_1 = p_2$.

Як буде показано далі, використання відображення $f_q(x)$ зручніше відображення $f(x)$, по-перше, з обчислювальної точки зору, по-друге, дріб $\frac{p}{q}$ може бути скоротним, а для майбутнього методу перемішування таке скорочення не бажано.

Також зазначимо, що метод вимагає, щоб обчислювальні процедури з використанням відображення Tent проводилися без округлень, а отже, виключно зі звичайними дробами. Справа в тому, що хаотична система не може бути реалізована на комп'ютері через скінченність числа станів. Кожен наступний стан системи не повинен збігатися з попереднім станом траєкторії. В іншому випадку (наприклад, в результаті округлення) траєкторія (послідовність станів системи) перетворюється на циклічну орбіту або вироджується на нерухому точку, як у відображенні Tent.

Для підтвердження цього здійснено моделювання з використанням пакету EXEL.

Розглянемо послідовність, яку згенеровано відображенням Tent $\{f^{(n)}(x_0)\}$, де $f^{(1)}(x) = f(x)$, $f^{(k)}(x) = f(f^{(k-1)}(x))$.

Візьмемо початкову точку $x_0 = 2/3$. Тоді теоретично повинно бути $f^{(k)}(x_0) = x_0$, $k = 1, 2, \dots$ Однак обчислення дають інші результати: $f^{(53)}(x_0) = 1$, $f^{(54)}(x_0) = 0$. Перше враження, що проблема виникла через те, що число $x_0 = 2/3$ не представляється кінцевим десятковим дробом. Тоді виберемо $x_0 = 0.4$. Теоретично повинно бути $f^{(1)}(x_0) = 0.8$, $f^{(2)}(x_0) = 0.4$ тощо. Однак в

результаті обчислень знову одержимо $f^{(53)}(x_0)=1$, $f^{(54)}(x_0)=0$. Такі самі результати отримаємо і при обчисленнях у пакеті MAPLE (і в інших подібних пакетах), якщо задаємо $x_0 = HFloat(evalf(2/3))$ або $x_0 = HFloat(0.4)$. Для будь-яких інших початкових значень x_0 , знову будемо одержувати $f^{(54)}(x_0)=0$.

Причина таких некоректних обчислень пояснюється тим, що число з відрізка $[0, 1]$ подається в комп'ютері в двійковій системі обчислення кінцевою сумою чисел виду $\alpha_j/2^j$ ($\alpha_j \in \{0,1\}$), тобто є двоїково-раціональним числом, яке дорівнює $A/2^m$ (A – ціле та $0 \leq A \leq 2^m$). Число m є двійковий порядок. Но тоді $f(A/2^m) = A_1/2^{m-1}$ (A_1 – ціле та $0 \leq A_1 \leq 2^{m-1}$), $f^{(2)}(A/2^m) = A_2/2^{m-2}, \dots$, $f^{(m+1)}(A/2^m) = 0$. На комп'ютерах, де проводилися обчислення ітерованої послідовності, максимальний порядок дорівнював 53.

Якщо відображення q -Tent використовується на підмножині парних натуральних чисел, то оператор перестановки визначається на множині векторів з компонентами, що складаються з натуральних однозначних чисел. Лінійний оператор перестановки визначається матрицею перестановки, що складається з нулів і одиниць, причому в кожному рядку і кожному стовпці знаходиться одна одиниця. Цей оператор оборотний, причому зворотна матриця дорівнює транспонованій вхідній матриці. Таким чином, і оператор q -Tent, і оператор перестановки взаємно однозначно відображають вищезгадані множини в себе.

3.3 Розробка методу перемішування

Введемо такі позначення. Нехай маємо множину кінцевих повідомлень X . Нехай повідомлення – це деяке число $x \in X$, $x < 10^m$. Представимо це повідомлення у вигляді:

$$x = \xi_1 10^{m-1} + \xi_2 10^{m-2} + \dots + \xi_{m-1} 10 + \xi_m, \quad \xi_j \in \{0,1,\dots,9\}, \quad j = 1,\dots,m.$$

Еквівалентний запис $x = [\xi_1 \ \xi_2 \ \dots \ \xi_{m-1} \ \xi_m]$.

Числу $x = [\xi_1 \ \xi_2 \dots \xi_{m-1} \ \xi_m]$ відповідає вектор $\bar{x} = \{\xi_1, \xi_2, \dots, \xi_{m-1}, \xi_m\}$. Ця відповідність взаємно однозначна. Позначимо її як $\bar{x} = \phi(x)$, та обернену відповідність $x = \phi^{-1}(\bar{x})$. Введемо оператор перестановки $\pi_M : X \rightarrow X$,

$$\pi_M(x) = \phi^{-1}(M \bar{x}) = \phi^{-1}(M \phi(x)),$$

де M – матриця перестановки розмірності $m \times m$. Очевидно, що обернений оператор – також оператор перестановки: $\pi_{M^*}(x) = \phi^{-1}(M^* \phi(x))$, де M^* – транспонована до M матриця.

На основі нелінійних операторів $f_q(x)$, $f_q^{(T)}(x)$ введемо нелінійні оператори $\frac{1}{2}f_q(2x)$, $\frac{1}{2}f_q^{(T)}(2x)$, де $q = 2 \cdot 10^m - 1$. Ці оператори відображають взаємно однозначно X в X . Оберненими до них операторами відповідно є $\frac{1}{2}(f_q)^{-1}(2x)$, $\frac{1}{2}(f_q^{(T)})^{-1}(2x)$.

Метод перемішування визначається виразом $y = G(x, T, M)$, де функція перемішування

$$G(x, T, M) = \phi^{-1} \left(M \phi \left(\frac{1}{2} f_q^{(T)}(2x) \right) \right) = \pi_M \left(\frac{1}{2} f_q^{(T)}(2x) \right).$$

Зворотна функція

$$G^{-1}(x, T, M) = \frac{1}{2} (f_q^{(T)})^{-1} \left(2 \phi^{-1} (M^* \phi(x)) \right) = \frac{1}{2} (f_q^{(T)})^{-1} (2 \pi_{M^*}(x))$$

також є функція перемішування.

Функція перемішування може розглядатися як функція шифрування, в якій параметр T і матриця M фактично є ключами. Зазначимо, що матрицю M можна задавати послідовністю чисел, де порядковий номер знаходження числа означає номер строки матриці, а само число співпадає з номером стовбця, де знаходиться одиниця.

Звичайно, що суперпозиція функцій перемішування є функція перемішування, тобто $G(G(x, T_1, M_1), T_2, M_2)$, яка залежить від двох параметрів T_1 і T_2 , і двох матриць M_1 і M_2 , тобто криптостійкість при суперпозиції має

суттєво підвищитися.

Для підтвердження цього проведемо аналіз показників якості функції перемішування.

3.4. Аналіз показників якості функції перемішування

3.4.1 Кореляційний аналіз функції перемішування

Для проведення кореляційного аналізу здійснено експериментальний тест.

$$\text{Нехай } m=4, \quad T_1=372, \quad T_2=251, \quad M_1 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}, \quad M_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Матриці M_1 та M_2 кодуються послідовностями: $\{3, 4, 2, 1\}$, $\{1, 3, 2, 4\}$.

Розглянуто три функції перемішування:

$$G_1(x) = \frac{1}{2} f_q^{(T_1)}(2x); \quad G_2(x) = G(x, T_1, M_1); \quad G_3(x) = G(G(x, T_1, M_1), T_2, M_2),$$

де $q = 2 \cdot 10^4 - 1 = 19999$.

У першій функції використовується виключно нелінійне перетворення q -Tent; у другій – додається лінійне перетворення перемішування; у третій – використовується суперпозиція попередньої функції з різними значеннями параметрів. Для кожної функції $G_k(x)$ побудовані відображення

$$x = \left\{ \frac{i}{10000} \right\}_{i=1}^{9999} \rightarrow y = \left\{ \frac{1}{10000} G_k(i) \right\}_{i=1}^{9999}, \quad k=1,2,3, \quad \text{та} \quad \text{проаналізовано}$$

статистичний показник лінійної залежності, а саме коефіцієнт кореляції:

$$\sigma_{xy}^{(k)} = \frac{1}{9999} \frac{\sum_{i=1}^{9999} \left(\frac{i}{10000} - \bar{X} \right) (y_i - \bar{Y})}{\sigma_x \sigma_y},$$

$$\text{де } \bar{X} = \bar{Y} = \frac{1}{9999} \sum_{i=1}^{9999} \frac{i}{10000} = \frac{1}{9999} \sum_{i=1}^{9999} \frac{1}{10000} G_k(i) = 0.5.$$

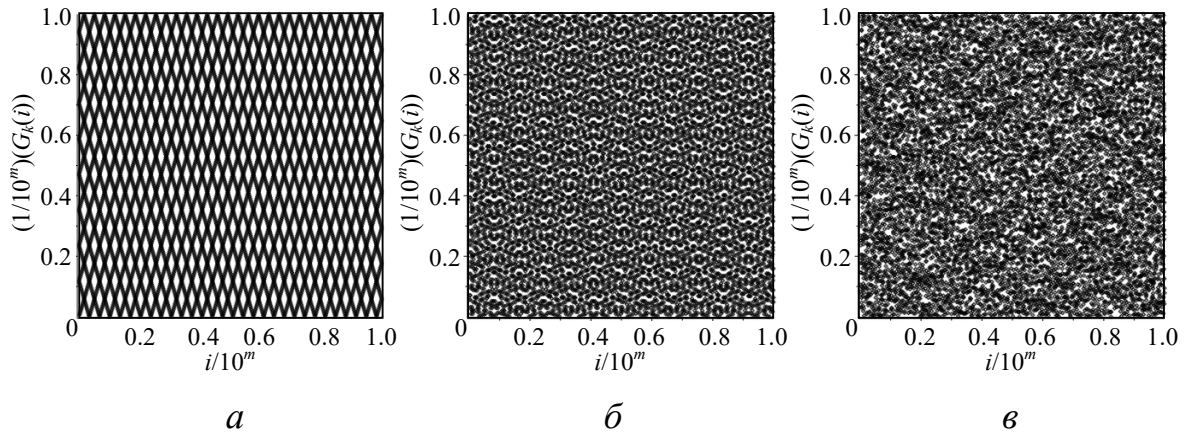


Рис.3.1. Графіки коефіцієнта кореляції $\left\{ \frac{i}{10000} \right\}_{i=1}^{9999} \rightarrow \left\{ \frac{1}{10000} G_k(i) \right\}_{i=1}^{9999}$ при різних k : $a - k = 1$, $б - k = 2$, $в - k = 3$

Обчислюємо $\sigma_y^{(k)} = 0.28864626$, $k = 1, 2, 3$, тоді коефіцієнти кореляції

$\sigma_{xy}^{(1)} = -0.00014894...$, $\sigma_{xy}^{(2)} = -0.00014533...$, $\sigma_{xy}^{(3)} = 0.00039123...$. Можна зробити

висновок, що не має лінійної залежності між послідовностями

$x = \left\{ \frac{i}{10000} \right\}_{i=1}^{9999}$ та $y = \left\{ \frac{1}{10000} G_k(i) \right\}_{i=1}^{9999}$, проте істотну залежність (нелінійну) у

перших двох випадках можна спостерігати на графіках (рис. 3.1).

3.4.2 Аналіз чутливості функції перемішування

Нехай є два повідомлення $x_0, y_0 \in X$, цифрові представлення яких відрізняються лише одним символом. Розглянуто послідовні перетворення цих повідомлень під дією функції перемішування.

$$G_k(x): x_{n+1} = G_k(x_n), y_{n+1} = G_k(y_n), k = 1, 2, 3, n = 0, 1, 2 \dots$$

Завданням було обчислити середнє та середньоквадратичне відхилення послідовності $\{x_i - y_i\}_{i=1}^{\infty}$. Розглянемо середнє послідовності $\{x_i - y_i\}$, тобто

функцію $S_n = \frac{1}{n+1} \sum_{i=0}^n (x_i - y_i)$. Рекурентне співвідношення має вигляд:

$$S_n = \frac{n}{n+1} S_{n-1} + \frac{1}{n+1} (x_n - y_n), n = 0, 1, 2 \dots, S_0 = x_0 - y_0.$$

Нехай, наприклад, $x_0 = [0001] = 1$, $y_0 = [0002] = 2$. Для кожної функції $G_k(x)$ були обчислені елементи послідовності $\{S_i\}_{i=1}^{\infty}$ (рис. 3.2 – 3.4)

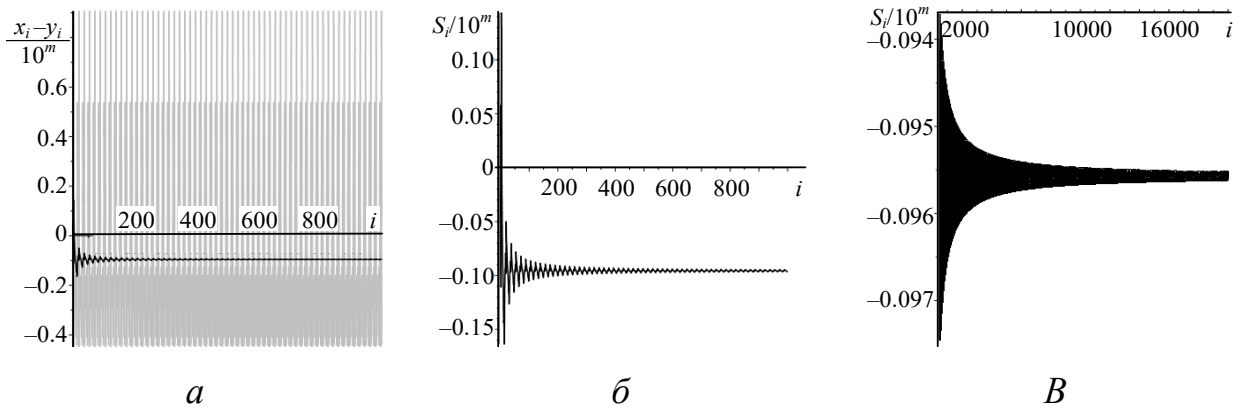


Рис. 3.2. Графіки відображень: $a - \left\{ \frac{x_i - y_i}{10^m} \right\}_{i=1}^{1000}$ (сірий колір); $б - \left\{ \frac{S_i}{10^m} \right\}_{i=1}^{1000}$;

$$B - \left\{ \frac{S_i}{10^m} \right\}_{i=500}^{20000} \text{ для функції перемішування } G_1(x)$$

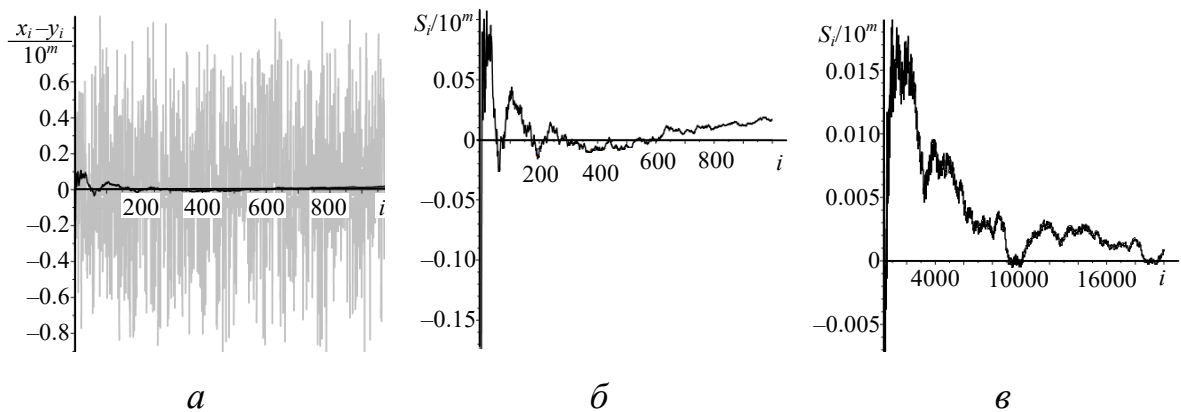


Рис. 3.3. Графіки відображень: $a - \left\{ \frac{x_i - y_i}{10^m} \right\}_{i=1}^{1000}$ (сірий колір); $а, б - \left\{ \frac{S_i}{10^m} \right\}_{i=1}^{1000}$;

$$в - \left\{ \frac{S_i}{10^m} \right\}_{i=500}^{20000} \text{ для функції перемішування } G_2(x)$$

Обчислені дисперсія та середньоквадратичне відхилення послідовності $\{x_i - y_i\}$, тобто функція $D_n = \frac{1}{n} \sum_{i=0}^n (x_i - y_i - S_i)^2$ и $\sigma_n = \sqrt{D_n}$.

Рекурентне співвідношення має вигляд: $D_n = \frac{n-1}{n} D_{n-1} + \frac{1}{n} (x_n - y_n - S_n)^2$, $D_0 = 0$. Для тих же початкових значень $x_0 = [0001]$, $y_0 = [0002]$ та для кожної функції $G_k(x)$ обчислені елементи послідовності $\{\sigma_i\}_{i=1}^{\infty}$ и (рис. 3.5, 3.6).

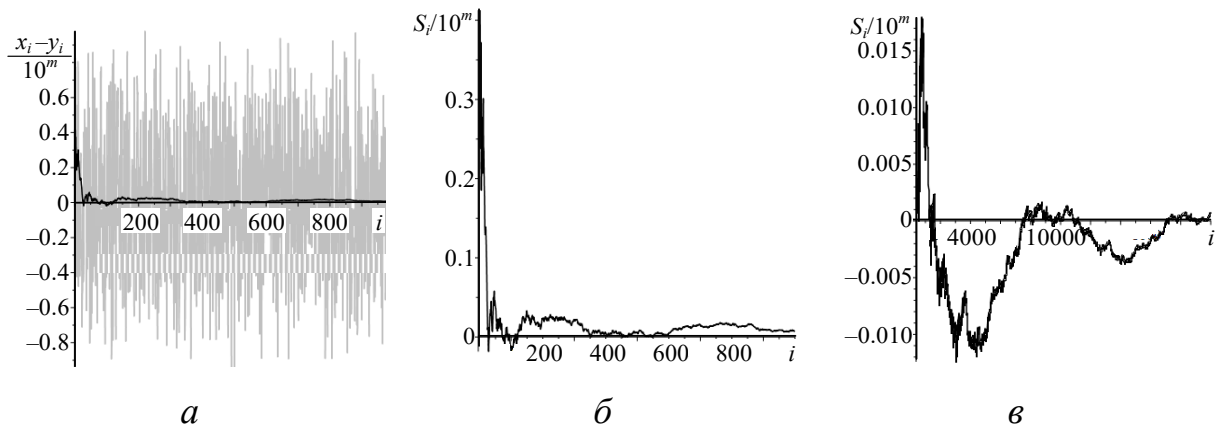


Рис. 3.4. Графіки відображень: $a - \left\{ \frac{x_i - y_i}{10^m} \right\}_{i=1}^{1000}$ (сірий колір); $б - \left\{ \frac{S_i}{10^m} \right\}_{i=1}^{1000}$; $в - \left\{ \frac{S_i}{10^m} \right\}_{i=500}^{20000}$ для функції перемішування $G_3(x)$

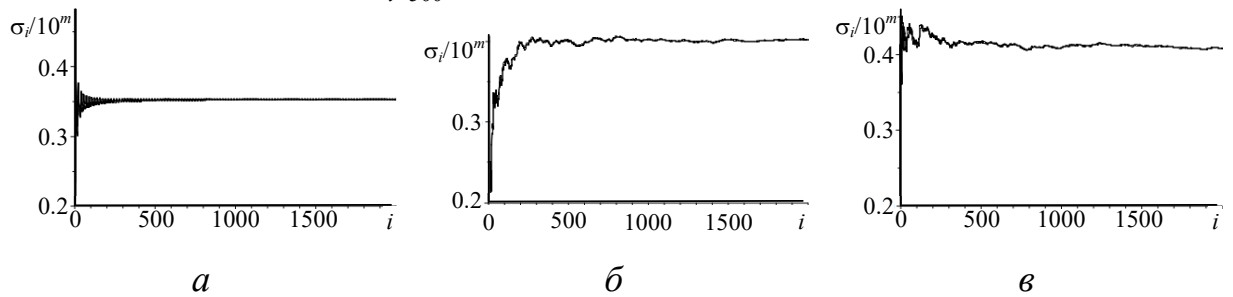


Рис. 3.5. Графіки відображень $\left\{ \frac{\sigma_i}{10^m} \right\}_{i=0}^{2000}$ для функції перемішування $G_k(x)$:

$a - k=1, б - k=2, в - k=3$

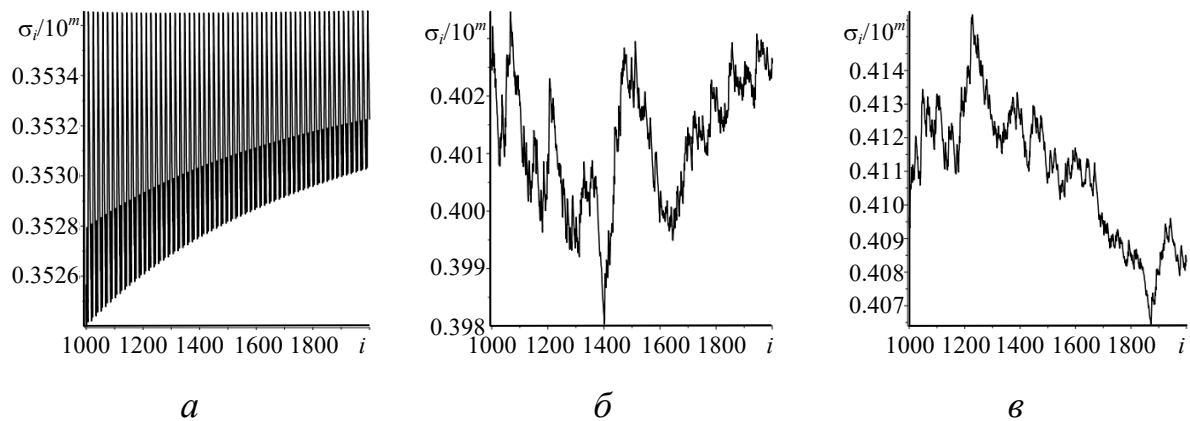


Рис. 3.6. Графіки відображень $\left\{ \frac{\sigma_i}{10^m} \right\}_{i=1000}^{2000}$ для функції перемішування $G_k(x)$:

$a - k=1, б - k=2, в - k=3$

Аналогічні результати можна отримати, якщо вибрати інші близькі значення чисел, які відповідають вихідним повідомленням. За результатами

експериментального аналізу можна дійти до висновку, що функція $G_1(x) = \frac{1}{2} f_q^{(T)}(2x)$ перемішує погано, чітко спостерігається збереження залежності між цифровими значеннями повідомлень у процесі перемішування, коли початкові значення були близькі. Середні різниці і середньоквадратичні відхилення прагнуть до константи зі зростанням числа перемішування. І, навпаки, для функцій перемішування $G_2(x)$, $G_3(x)$ середні різниці та середньоквадратичні відхилення поводяться без вираженого тренду (рис. 3.6).

3.4.3 Аналіз довжин періодів функції перемішування

Функція перемішування $G(x)$ відображає скінчену множину X в себе. Це означає, що для будь-якого $x \in X$ ітераційна послідовність $\{G^{(i)}(x)\}_{i=0}^{\infty}$ (вважається, $G^{(0)}(x) = x$) буде періодичною з періодом не більше ніж $10^m - 1$.

При побудові функцій перемішування важливо знати, як розбивається множина X на підмножини, що визначаються періодичними орбітами. Важливо знати кількість цих множин та його розміри, тобто мінімальну, середню та максимальну довжину періодичної орбіти, кількість орбіт. Наявність великої кількості орбіт із короткою довжиною небезпечна, оскільки це веде до утворення патернів. Краще – наявність однієї довгої орбіти, що проходить майже через весь простір X . Також може бути задовільним і ансамбль однотипних орбіт середніх довжин.

Розглянемо знову функцію $G_3(x) = G(G(x, T_1, M_1), T_2, M_2)$ з введеними значеннями параметрів T_1 , T_2 , M_1 , M_2 . Поставимо задачу: знайти всі підмножини множини X , що визначаються періодичними орбітами.

Чисельний аналіз показує, що відображення $G_3(x)$ має три нерухомі точки, чотири періодичні орбіти довжини 2, по одній орбіті довжин 3, 4, 6, 16, 17, 19, 319, 9604. В таблиці 3.1 наведено точки, що породжують відповідні періодичні орбіти.

Таблиця 3.1. Точки, що породжують відповідні періодичні орбіти

Період орбіт	1	2	3	4	6	16	17	19	319	9604
Точки, що породжують відповідні періодичні орбіти	6234 9666 9964	2122 5427 6111 9653	5913	1737	0109	0037	1192	0089	0016	0001

Зазначимо, що дві найбільш довгі орбіти містять $9604+319=9923$ точок, що становить більше 99% від загальної кількості точок множини X . Такий розподіл циклів відображення свідчить про добрі дифузійні властивості функції перемішування, яка побудована на найпростіших базових принципах. На рис.

3.7 наведено графіки функцій $\left\{ \frac{1}{9999} G_3^{(i)}(x) \right\}_{i=0}^{9999}$ для різних початкових точок.

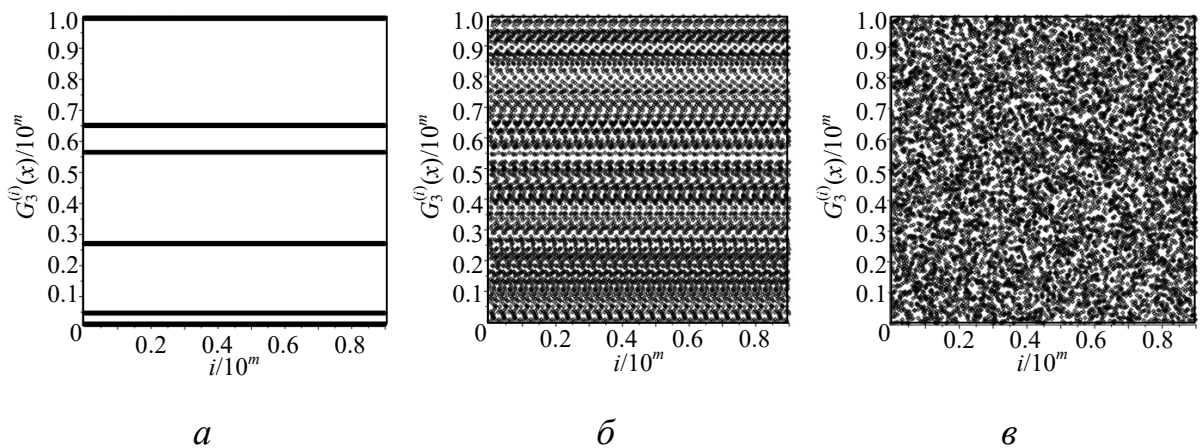


Рис. 3.7. Графіки відображень $\left\{ \frac{1}{9999} G_3^{(i)}(x) \right\}_{i=0}^{9999}$: а – $x = [0109]$ ($x = \frac{109}{9999}$,

$T = 6$), б – $x = [0016]$ ($x = \frac{16}{9999}$, $T = 319$), в – $x = [0001]$ ($T = 9604$)

Розглянемо варіанти можливої модифікації відображення перемішування. При побудові функції перемішування

$$G(x, T, M) = \phi^{-1} \left(M \phi \left(\frac{1}{2} f_q^{(T)}(2x) \right) \right) = \pi_M \left(\frac{1}{2} f_q^{(T)}(2x) \right)$$

використовувалась матриця перестановки M розмірності $m \times m$. Таких матриць існує $m!$. Відмітимо, що при $m \geq 25$ $m! > 10^m$, а при $m \geq 7$ $(m!)^2 > 10^m$. Таким чином, вже при $m \geq 25$ для злому функції перестановки знадобиться груба сила, тобто повний перебір варіантів. Для злому суперпозиції двох функцій перемішування повний перебір буде потрібний вже для $m \geq 7$. Тим не менш, властивості лінійного оператора, який застосовується для перемішування, можна ще посилити, використовуючи алгоритм Хілла, а саме, замість матриці перестановки брати будь-яку $m \times m$ матрицю, стежачи лише, щоб значення її визначника не ділилося на 2 чи 5. Нехай A - така матриця. Тоді оператор

$$\pi_A(x) = \phi^{-1}(A\bar{x}(\text{mod } 10)) = \phi^{-1}(A\phi(x)(\text{mod } 10))$$

відображає X в X взаємно однозначно [113]. Відповідних матриць існує більше, ніж $10^{m(m-1)}$. З одного боку як елементи матриці A можна використовувати тільки однозначні числа (оскільки обчислення йдуть по модулю 10), з іншого боку треба відкидати матриці з визначниками рівними нулю, парними або кратними п'яти. Відмітимо, що $M\bar{x}(\text{mod } 10) = M\bar{x}$, тому алгоритм з модифікацією по Хіллу є узагальненням викладеного вище методу.

Інший шлях модифікації методу перемішування пов'язаний із розбиттям вихідного повідомлення на блоки та застосуванням різних функцій перемішування до різних блоків. Причому після першого перемішування можна провести друге розбиття і перемішати, що узагальнює застосування операції суперпозиції функцій перемішування. Наприклад, нехай $m = 16$. Спочатку розбиваємо повідомлення на блоки 4-8-4 та застосовуємо три функції перемішування до кожного блоку. Потім отримане повідомлення розбиваємо на блоки 8-8 і застосовуємо знову нові функції перемішування до кожного блоку. Усього можливих варіантів використання матриць перестановок існує $(4!)^2 (8!)^3 > 10^{16}$, тобто для злому оператора перемішування, побудованого за допомогою функцій перемішування, необхідний повний перебір варіантів.

3.5 Метод квазівипадкового перемішування довільної числової множини для захищеності від можливих маніпуляцій

Результати проведених досліджень дозволяють сформулювати наступний метод квазівипадкового перемішування довільної числової множини для захищеності від можливих маніпуляцій.

1. З хеш-образів списку координаторів формується простий нумерований список шляхом довільного впорядкування. Елементи цього списку є числовою множиною, що підлягає перемішування.

Для кожного числа T з цієї множини послідовно виконуються кроки 2 – 7.

2. Початкове число множиться на два.

3. До отриманого числа T разів застосовується перетворення q -Tent с параметром $q = 2 \cdot 10^m - 1$.

4. Знову отримане число ділиться на два.

5. Це число перетворюється на вектор.

6. Матриця перестановки M множиться на цей вектор

7. Отриманий вектор перетворюється на число.

8. Перемішування закінчено.

3.6 Ілюстративний приклад із перемішуванням зображень

Застосуватиме функцію перемішування для шифрування зображень. Візьмемо класичне зображення «Lena» (рис.8-а), що складається з 561×2^{10} пікселів [114]. Побудуємо 561×2^{10} матрицю, що кодує зображення. Елементи матриці обчислюються з точністю до 16 десяткових символів. Округлимо елементи матриці до двох знаків після коми, отримавши, тим самим урізане зображення (рис. 8-б). Потім округлимо елементи матриці до одного знаку після коми (рис. 3.8-с).

У разі, який зображений нижче (рис. 3.8 в) помітна істотна втрата інформації, тобто якість зображення значно погіршилося.

*a**б**в*

Рис. 3.8. Зображення «Lena»: *a* – оригінальне, *б* – урізане (до двох десяткових знаків), *в* – урізане (до одного десяткового знаку)

Помножимо елементи цих матриць на 100 і представимо їх у вигляді одновимірних масивів, що містять $561 \cdot 2^{10}$ двозначних чисел, або у вигляді масивів, які містять $561 \cdot 2^{11}$ однозначних чисел. Розіб'ємо масиви на блоки, які що складаються з восьми чисел. Таких блоків існує $561 \cdot 2^8$ у кожному масиві. До кожного елементу блоку, тобто числа з вісьмома знаками, застосуємо функцію перемішування. Отримаємо нову множину блоків двох масивів, з котрих складемо матриці, і виведемо зображення, які кодуються цими матрицями. Отримаємо два перемішані зображення урізаних (до двох десяткових знаків і до одного десяткового знаку) зображень «Ліна» (рис. 3.9-а, 3.10-а).

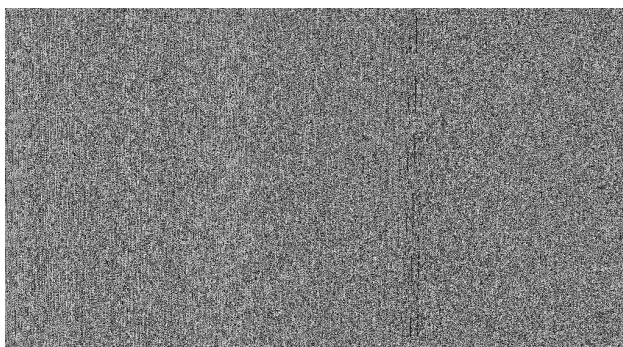
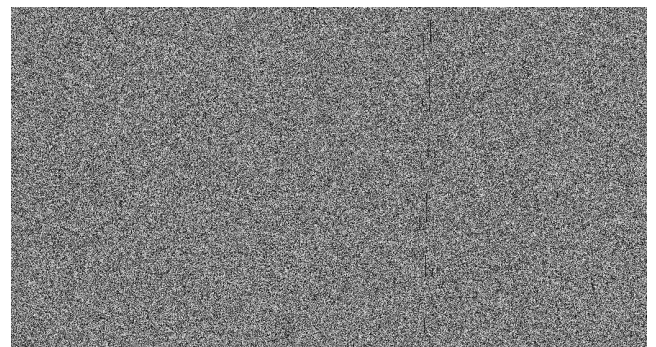
*a**б*

Рис. 3.9. Перемішування зображення «Lena» (урізаного до двох десяткових знаків): *a* – з розбиттям на блоки по 8 елементів;
б – з розбиттям на блоки по 24 елементи

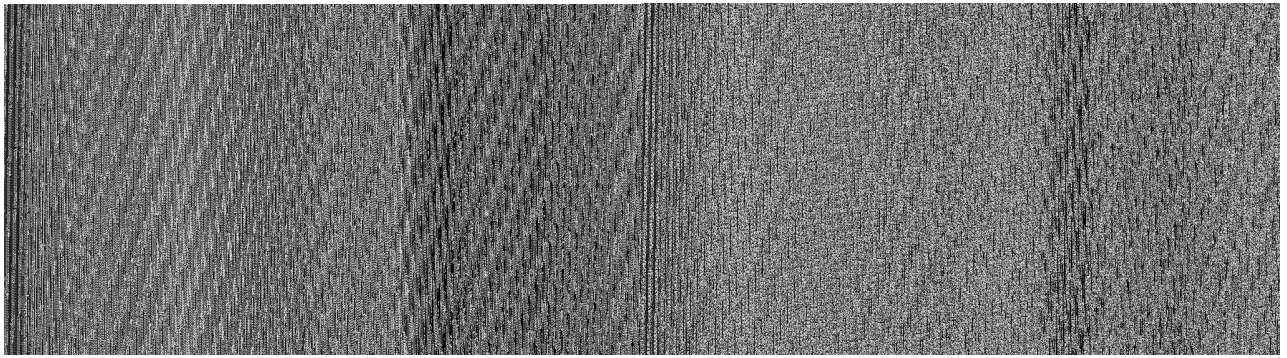
*a**б*

Рис. 3.10. Перемішування зображення «Lena» (урізаного до одного десяткового знаку): *a* – з розбиттям на блоки по 8 елементів;
б – з розбиттям на блоки по 24 елементи

До кожного блоку обох масивів була застосована та сама функція перемішування. Параметри функції перемішування: $T_1 = 373$, $T_2 = 251$, $m_1 = \{2, 4, 1, 3, 8, 7, 6, 5\}$, $m_2 = \{8, 3, 7, 2, 6, 1, 4, 5\}$. Матриці перемішування M_1 та M_2 кодуються послідовностями m_1 , m_2 (порядковий номер елементу послідовності визначає номер рядка, а значення цього елементу номер стовпця, на перетин яких стоїть одиниця в матриці перестановки). Ці перемішані зображення оборотні без втрати інформації, тобто зворотні зображення збігаються з вихідними урізаними зображеннями.

Щоб покращити якість перемішування, розіб'ємо масиви на блоки, що складаються з двадцяти чотирьох чисел. Таких блоків $187 \cdot 2^8$ у кожному масиві. Кожному елементу блоку (з двадцяти чотирьох знаків), застосуємо функцію перемішування з параметрами: $T_1 = 573$, $T_2 = 751$,

$$m_1 = \{13, 24, 20, 19, 9, 10, 2, 4, 23, 22, 1, 21, 16, 3, 18, 17, 8, 7, 15, 14, 6, 5, 12, 11\},$$

$$m_2 = \{16, 17, 8, 9, 18, 3, 10, 7, 14, 2, 23, 6, 11, 12, 1, 22, 21, 4, 13, 24, 5, 20, 19, 15\}.$$

На рисунках 3.9, 3.10 можна бачити, як покращується якість перемішування зі зростанням розмірів блоків розбиття масивів, що відповідають матрицям, що кодують вихідне зображення.

3.7 Висновки до третього розділу

У третьому розділі досліджено можливість застосування найпростіших положень теорії динамічних систем для захисту протоколу PoS від можливих маніпуляцій під час вирішення завдань псевдовипадкового вибору у координаційній мережі. Показано, що основна проблема використання хаотичних систем у комп'ютерних обчисленнях полягає в тому, що кількість різних станів у комп'ютері кінцеве. Тому комп'ютерні моделі хаосу є лише апроксимацією справжньої хаотичної поведінки. Така апроксимація може передавати властивість вихідної системи лише на початкових ітераціях, через чутливість системи до початкових умов траєкторію вихідної та апроксимованої систем розходяться дуже швидко, при цьому кожна траєкторія апроксимованої системи стає періодичною.

Проте використання дифузійних властивостей нелінійних динамічних систем у просторах з кінцевим числом станів виявляється можливим. Більше того, вже за допомогою найпростішого нелінійного відображення Tent можна побудувати функції перемішування, що взаємно однозначно відображають простір станів у себе, з хорошими дифузійними властивостями. При цьому використовувалися суперпозиції нелінійного відображення Tent і лінійного відображення перестановки (у більш загальному випадку відображення Хілла). Для побудованих функцій були проведені кореляційний аналіз, аналіз чутливості, а також найбільш значущий аналіз для просторів з кінцевим числом станів, аналіз довжин періодів циклів, що розбивають простір станів на підмножини, що не перетинаються. Підтверджено очікувані хороші дифузійні властивості цих функцій перемішування. Досліджена функція перемішування має наступні переваги:

- відсутність колізій,
- простота застосовуваних математичних операцій,
- злом її можливий тільки шляхом повного перебору.

Отримано наступний пункт наукової новизни:

– Отримав подальший розвиток метод перемішування даних за допомогою бієктивного відображення, яке є суперпозицією лінійного та найпростішого нелінійного відображення q-Tent, що дозволило здійснити процес якісного перемішування при формуванні PoS консенсусу, і тим самим захистити процес обробки транзакцій у розподілених реєстрах від можливих маніпуляцій.

РОЗДІЛ 4

РОЗРОБКА ТА ВИПРОБУВАННЯ ЕКСПЕРИМЕНТАЛЬНОЇ
ДЕЦЕНТРАЛІЗОВАНОЇ ПЛАТФОРМИ WATERFALL НА ОСНОВІ
ЗАПРОПОНОВАНИХ МОДЕЛЕЙ І МЕТОДІВ

4.1 Загальні положення

На підставі запропонованих у ході дослідження моделей та методів розроблено експериментальну децентралізовану платформу розподіленого реєстру Waterfall [115].

Основні властивості системи.

Мова програмування – Golang.

Архітектура – двошарова, загальна мережа – BlockDAG, координаційна мережа у початковій ітерації – Blockchain.

Протокол консенсусу – Proof of Stake удосконалений.

Функції, що виконуються – обслуговування транзакцій зі створенням і виявленням токенів (NFT), обслуговування смарт-контрактів, розробка розподілених додатків dApps.

4.2 Блок-схема програмного продукту

На рисунку 4.1 наведено блок-схему програмного продукту Waterfall. Нижче наведено її розшифрування.

1. Init Coordinator genesis

Відбувається ініціалізація координуючої мережі, загальні параметри всіх вузлів задані в генезис-блоці.

2. Init BlockDag genesis

Відбувається ініціалізація мережі BlockDag, загальні параметри для всіх вузлів задані в генезис-блоці.

3. Init?

Якщо ініціалізація не пройшла успішно, то виникає помилка.

4.Init?

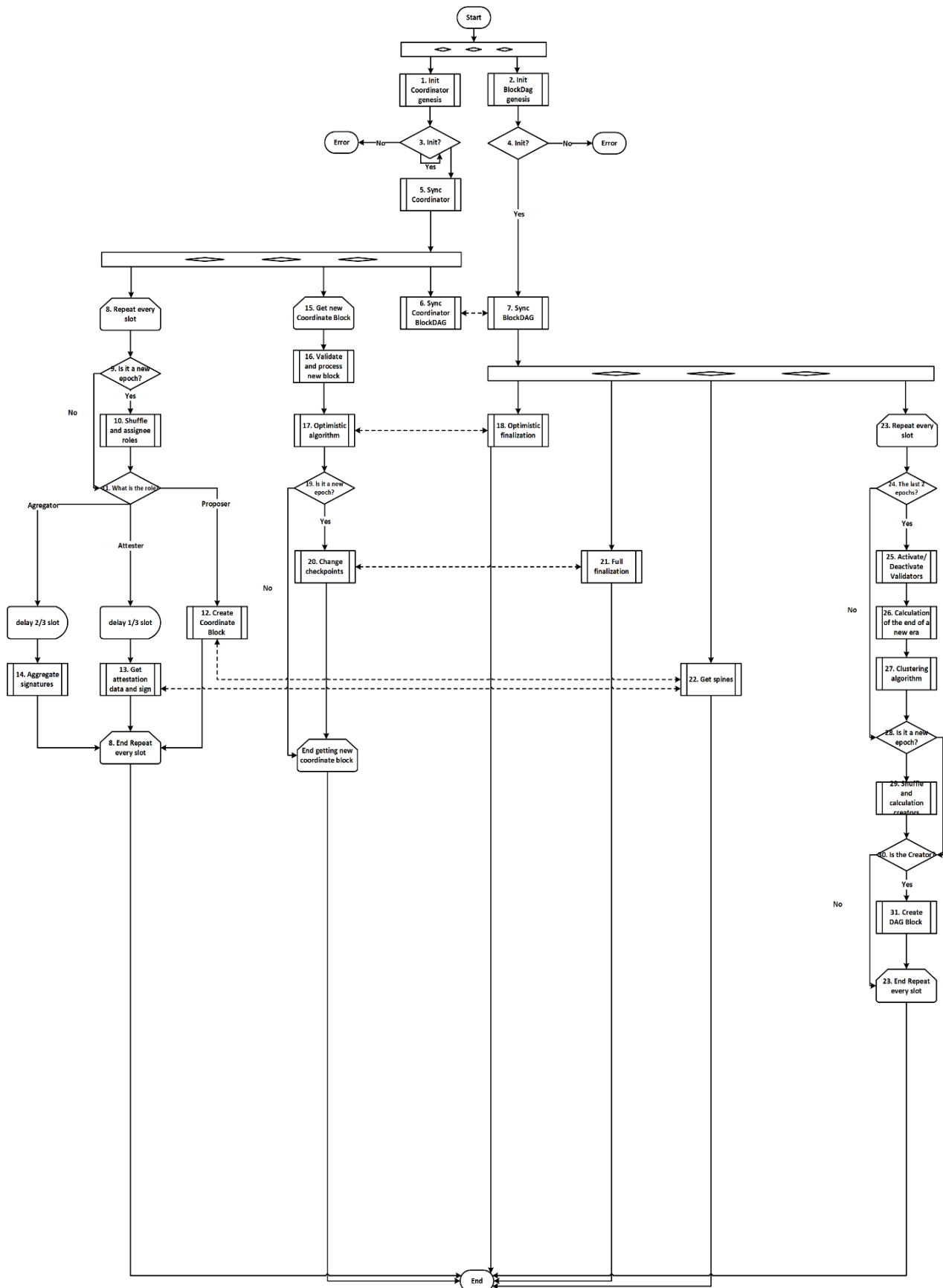


Рис. 4.1. Блок-схема програмного продукту Waterfall

Якщо ініціалізація не пройшла успішно, то виникає помилка.

5. Sync Coordinator

Координуючий вузол підключається до завантажувального вузла і отримує список інших вузлів, починає підключатися, завантажувати та перевіряти блоки.

6. Sync Coordinator BlockDAG

Після того, як координуючий вузол синхронізований, в окремому потоці він починає відправляти в BlockDAG для процедури консенсусу по порядку хребетні блоки кожного фінального чекпоінту, починаючи з генезис-блоку.

7. Sync BlockDAG

BlockDAG вузол підключається до завантажувального вузла та отримує список інших вузлів. Отримавши від координуючого вузла порядок хребетних блоків, завантажує їх та їх батьківські блоки з інших вузлів. Після того, як усі блоки завантажені, запускається процедура фіналізації (застосування транзакцій та розрахунку стану), згідно з алгоритмом упорядкування.

8. Repeat every slot

В окремій процедурі таймер запускається через кожні 4 секунди.

9. Is it a new epoch?

Визначається, чи настає нова епоха через кожні 32 слоти.

10. Shuffle and assignee roles

Запускається алгоритм перемішування, який на епоху визначає склади комітетів для кожного слоту та ролі координаторів у них.

11. What is the role?

У поточному слоті перевіряємо роль координаторів кожного вузла.

12. Create Coordinate Block

Якщо роль координатора - творець блоку, то вузол запитує у BlockDAG вузла порядок хребетних блоків, застосовує алгоритм визначення батьківського блоку, додає його до блоку атестації та агрегації, виконує алгоритм оптимістичної фіналізації та на підставі всіх даних та розрахунків розраховує новий стан, хеш якого додається до блоку та відправляється до мережі.

13. Get attestation data and sign

Якщо роль координатора - заствідува, то через 1/3 слота вузол запитує порядок хребетних блоків, щоб знайти оптимальний координуючий блок, в якому валідний для вузла порядок хребетних блоків, підписує та відправляє до підмережі комітету. Затримка потрібна для того, щоб можна було підписати блок, створений саме в цьому слоті.

14. Aggregate signatures

Якщо роль координатора - атестатор, то через 2/3 слота атестатор агрегує всі підписи, які він отримав від свого комітету, і відправляє в мережу агрегований підпис. Це потрібно для того, щоб зменшити кількість інформації, що пересилається по мережі.

15. Get new Coordinate Block

В окремому потоці вузол отримує нові блоки, створені цим чи іншими вузлами.

16. Validate and process new block

Новий блок перевіряється, на основі даних, що містяться в блоці, розраховується стан блоку.

17. Optimistic algorithm

Під час розрахунку стану застосовується оптимістичний алгоритм, результатом якого є порядок блоку хребта для оптимістичної фіналізації.

18. Optimistic finalization

BlockDAG вузол отримує порядок хребетних блоків, перевіряє, чи є у нього ці блоки і всі його батьківські блоки, якщо є, то виконується алгоритм упорядкування предків хребетних блоків. Цей порядок послідовно фіналізується.

19. Is it a new epoch?

Перевіряється, чи настала нова епоха.

20. Change checkpoints

При зміні епохи, можливо, зміниться і фіналізований чекпойнт, якщо за нього проголосували більше 2/3 координаторів. У цей час оновлюється баланс

валідаторів. У BlockDAG вузол відправляється фіналізований порядок хребетних блоків від попереднього чекпойнту до поточного.

21. Full finalization

Після отримання від координуючої ноди фіналізований порядок блоків хребта виконується перехід від попереднього чекпойнту до поточного. Вузол перевіряє, чи є у нього ці блоки та всі його батьківські блоки, якщо є, то виконується алгоритм упорядкування батьків хребетних блоків. Цей порядок послідовно фіналізується і не може змінюватися.

22. Get spines

Хребетний блок надходить на вхід номер слота і повертає блоки хребта від цього слота до останнього.

23. Repeat every slot.

В окремій процедурі таймер запускається через кожні 4 секунди.

24. Is it the last 2 epochs of the era?

Перевіряється, чи не настала передостання епоха поточної ери

25. Activate/Deactivate Validators

Активуються та деактивуються валідатори, щоб отримати список валідаторів, які будуть працювати в новій ері.

26. Calculation of the end of a new era

На підставі списку валідаторів розраховується епоха, коли нова ера закінчиться.

27. Clustering algorithm

На підставі зібраної статистики розраховується список активних валідаторів.

28. Is it a new epoch?

Визначається, чи настала нова епоха.

29. Shuffle and calculation creators

Використовується алгоритм перемішування, щоб зі списку валідаторів епохи визначити, хто у кожному слоті епохи створюватиме блоки.

30. Is the Creator?

Визначаємо, чи може валідатор в цьому слоті створити блок.

31. Create DAG Block

Беруться транзакції з пулу своєї групи, додаються до блоку та відправляються по мережі.

У наступних підрозділах описано ключові моменти програмної реалізації платформи.

4.3. Схема вузлів та їх взаємодії

На рисунку 4.2 наведено схему вузлів та їх взаємодії.

Кожен вузол Waterfall складається з наступного програмного забезпечення:

1. Coordinator-beacon-node - це програма, яка відповідає за синхронізацію роботи в координуючій мережі. Зберігає інформацію в своїй базі даних.

2. BlockDAG-node – програма, яка відповідає за створення/обмін блоками і транзакціями в мережі BlockDAG. Зберігає інформацію в базі даних. Якщо на цьому вузлі запущено Worker, то вам потрібно імпортувати ключі валідатора в сховище ключів.

3. Coordinator-Validator – якщо на вузлі працює worker, то потрібно запустити цю програму, з її допомогою імпортувати ключі координатора в сховище ключів. Ця програма підключається до маяка-вузла координатора і підписує дані, які вона отримує від нього, відповідно до ролі координатора.

Етапи запуску вузла:

1. Коли запускається Coordinator-beacon-node, програма підключається до публічного Coordinator-bootnode-node, щоб отримати список адрес інших Coordinator-beacon-node.

2. Отримавши список адрес інших Coordinator-beacon-node, програма починає підключатися до них і встановлювати з'єднання.

3. Після встановлення достатньої кількості з'єднань Coordinator-beacon-node починає синхронізацію.

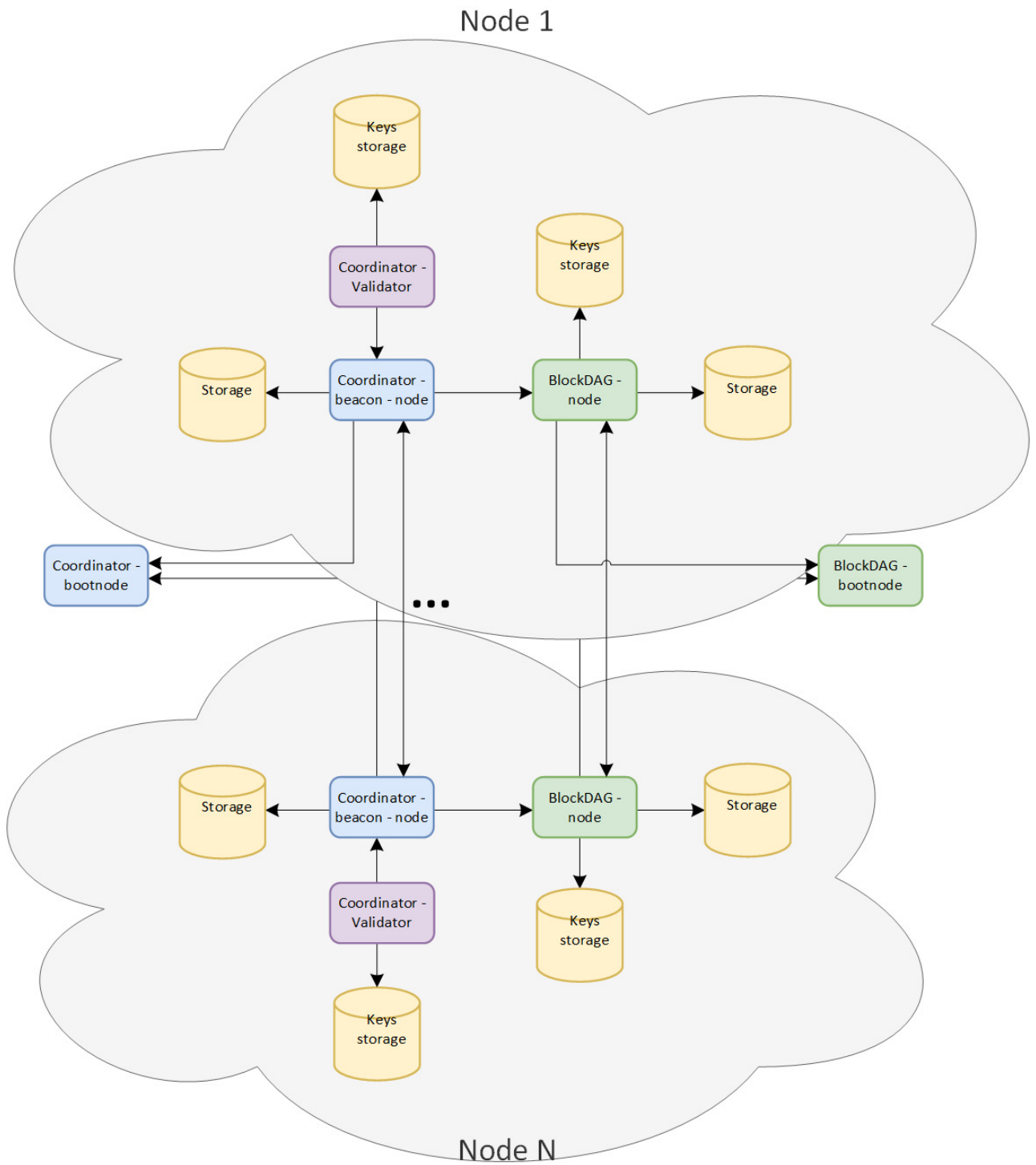


Рис. 4.2. Схема вузлів та їх взаємодії

4. При запуску BlockDAG-node, програма спочатку підключається до публічного BlockDAG-bootnode, щоб отримати список адрес інших BlockDAG-вузлів.

5. Отримавши список адрес інших BlockDAG-вузлів, програма починає

підключатися до них і встановлювати з'єднання.

6. Після синхронізації Coordinator-beacon-node програма підключається до його BlockDAG-вузлу і запускає процедуру синхронізації BlockDAG-вузла.

7. Після синхронізації Coordinator-beacon-node і BlockDAG-вузла, Координатор-валідатор підключається до свого Coordinator-beacon-node і починає працювати. При цьому Validator в вузлі BlockDAG також починає свою роботу.

4.4 Основні структури даних

Структури даних у вузлах BlockDAG.

Транзакція. Будь-яка зміна стану мережі здійснюється за допомогою транзакції. Транзакції використовуються для передачі монет, створення смарт-контрактів і виконання методів смарт-контрактів.

Транзакція складається з наступної структури:

```
type Tx struct {
    Nonce    uint64      // nonce of sender account
    GasPrice *big.Int    // wei per gas
    Gas      uint64      // gas limit
    To       *common.Address `rlp:"nil"` // nil means contract or easy-token
    creation
    Value    *big.Int    // wei amount
    Data     []byte      // contract invocation input data
    V, R, S  *big.Int    // signature values type Tx struct {
},
```

де:

Nonce – індекс транзакції у відправника і зазвичай він збільшується для послідовного виконання, якщо транзакція ще не була додана в блок, то можна відправити іншу транзакцію з тим же Nonce і тоді вона буде замінена новою і в блок потрапить нова транзакція;

GasPrice – орієнтовна вартість кількості ресурсів Gas, яку бажає заплатити користувач;

Gas – кількість ресурсів, які необхідно витратити для виконання цієї транзакції, для транзакції передачі це 21000, для контрактної транзакції це кількість операцій, виконаних віртуальною машиною, і обсяг даних, які необхідно записати;

To – адреса одержувача монет, порожнє поле, якщо створюється контракт, адреса договору, якщо контрактний метод виконується, адреса токена, якщо операція виконується над токеном;

Value – сума, яка передається адресату або договору;

Data – дані, які містять код контракту або токена;

V, R, S – підпис транзакції, за якою можна визначити відправника транзакції.

Блоки. Транзакції об'єднуються в блоки. Якщо транзакція змінює стан мережі, то блок фіксує цей стан.

Блок складається з двох частин:

Header – де вказана основна інформація про блок,

Body – де містяться блокові транзакції.

Заголовок має наступну структуру:

```
type Header struct {
```

```
//Base fields (set while create)
```

```
ParentHashes common.HashArray `json: "parentHashes" gencodec: "required"``
```

```
Slot uint64 `json: "slot" gencodec: "required"``
```

```
Era uint64 `json: "era" gencodec: "required"``
```

```
Height uint64 `json: "height" gencodec: "required"``
```

```
Coinbase common. Address `json: "miner" gencodec: "required"``
```

```
TxHash common.Hash `json: "transactionsRoot" gencodec: "required"``
```

```
BodyHash common. Hash `json: "bodyRoot" gencodec: "required"``
```

```
GasLimit uint64 `json: "gasLimit" gencodec: "required"``
```

```
Time uint64 `json: "timestamp" gencodec: "required"``
```

```

Extra []byte `json: "extraData" gencodec: "required"`
//Last finalized block fields (set while create)
CpHash common.Hash `json: "cpHash" gencodec: "required"`
CpNumber uint64 `json: "cpNumber" gencodec: "required"`
CpBaseFee *big.Int `json: "cpBaseFeePerGas" gencodec: "required"`
CpRoot common.Hash `json: "cpStateRoot" gencodec: "required"`
CpReceiptHash common.Hash `json: "cpReceiptsRoot" gencodec: "required"`
CpGasUsed uint64 `json: "cpGasUsed" gencodec: "required"`
CpBloom Bloom `json: "cpLogsBloom" gencodec: "required"`
//State fields (set while finalize)
// BaseFee was added by EIP-1559 and is ignored in legacy headers.
BaseFee *big.Int `json: "baseFeePerGas" rlp: "optional"`
Number *uint64 `json: "number" rlp: "optional"`
Root common.Hash `json: "stateRoot" rlp: "optional"`
ReceiptHash common.Hash `json: "receiptsRoot" rlp: "optional"`
GasUsed uint64 `json: "gasUsed" rlp: "optional"`
Bloom Bloom `json: "logsBloom" rlp: "optional"`
},

```

де:

ParentHashes – хеші блоків, які є підказками на момент створення блоку;

Slot – слот, в якому був створений блок;

Era – ім'я, в якому був створений блок;

Height – кількість блоків у минулому, якщо рекурсивно зібрати всі блоки за допомогою ParentHashes;

Coinbase – валідатор, який створив блок;

TxHash – корінь дерева транзакцій Merkle, які містяться в блоці, щоб перевірити, щоб транзакції, які зберігаються в тілі блоку, відповідали Заголовку блоку;

BodyHash – хеш вмісту Тіла поточного блоку, для перевірки відповідності даних, що зберігаються в Тілі блоку, Заголовку блоку;

GasLimit – інформація про те, скільки газу можна витратити якомога більше, щоб виконати всі транзакції, що зберігаються в блоці;

Time – час створення блоку;

CpHash – хеш блоку останньої фіналізованої контрольної точки, на момент створення поточного блоку (інформація про контрольну точку потрібна в блоці для узгодження стану завершеного блоку);

CpNumber – номер блоку кінцевої контрольної точки;

CpRoot - кореневий стан завершеного блоку контрольних точок;

CpReceiptHash – хеш результату виконаних транзакцій у блоці фіналізованої контрольної точки;

CpGasUsed – кількість ресурсів Gas, використаного при виконанні всіх транзакцій завершеного блоку контрольних-пропускних пунктів;

BaseFee – базова комісія за транзакцію;

Number – номер поточного блоку після доопрацювання (цей параметр не використовується при хешуванні блоку, так як він не існує на момент створення блоку);

Root – стан поточного блоку після доопрацювання (цей параметр не використовується при хешуванні блоку, так як його немає в момент створення блоку);

ReceiptHash – хеш результату транзакцій, здійснених в блоці після фіналізації (цей параметр не використовується при хешуванні блоку, так як його немає в момент створення блоку);

GasUsed – скільки газу використовується при виконанні всіх транзакцій блоку після доопрацювання (цей параметр не використовується при хешуванні блоку, так як його не існує на момент створення блоку).

BlockDAG. Для того щоб прискорити процес розрахунку Height, а також при оптимізації алгоритму упорядкування блоків, використовується структура BlockDAG:

```
type BlockDAG struct {
    Hash common.Hash
```

```

Height uint64
Slot uint64
CpHash common.Hash
CpHeight uint64
// ordered non-finalized ancestors hashes
OrderedAncestorsHashes common.HashArray
},

```

де:

Hash – хеш блоку;
 Height – висота блоку;
 Slot – слот, в якому був створений блок;
 CpHash – хеш контрольної точки, зазначеної в блоці;
 CpHeight – висота контрольної точки, задана в блоці;
 OrderedAncestorsHashes – хеші блоків, які знаходяться в минулому блоці з блоку контрольних точок.

Під час створення блоку, ParentHashes вказують хеші блоків, на які не посилалися інші блоки на даний момент, для полегшення цього процесу існує таблиця цих блоків, яка оновлюється під час створення або отримання нового блоку – type Tips map[common.Hash]*BlockDAG

Завершення. Кожен слот один раз в епоху в координуючий вузол відправляє інформацію своєму вузлу BlockDAG про:

- оптимістичне доопрацювання,
- повне доопрацювання,
- активація/деактивація валідаторів.

Для цього запроваджено структуру Checkpoint

```

type Checkpoint struct {
Epoch uint64
FinEpoch uint64
Root common.Hash
Spine common.Hash

```

},

де:

- Epoch – епоха, в яку був створений координуючий блок (контрольно-пропускний пункт);
- FinEpoch – епоха, в яку був повністю завершений координаційний блок (контрольно-пропускний пункт);
- Root – корінь стану координуючого блоку (контрольної точки);
- Spine – останній доопрацьований хребетний блок, який фіксується в стані координуючого блоку (контрольно-пропускної точки).

Для передачі повідомлення про активацію/деактивацію валідаторів з координуючої мережі використовується наступна структура:

```
type ValidatorSyncOp uint64
const (
    Activate ValidatorSyncOp = iota
    Deactivate
    UpdateBalance
)
type ValidatorSync struct {
    InitTxHash common.Hash
    OpType ValidatorSyncOp
    ProcEpoch uint64
    Index uint64
    Creator common.Address
    Amount *big.Int
    TxHash *common.Hash
},
```

де:

- InitTxHash – хеш транзакції, яка ініціювала дію
- OpType – тип операції
- ProcEpoch – епоха, не раніше якої дана операція має виконуватися

- Index – номер операції
- Creator – валідатор з яким провадиться дія
- Amount – суму, яку потрібно поповнити на баланс валідатора
- TxHash – хеш транзакції, яка оновлює стан у BlockDAG вузлі

Для синхронізації між вузлом координатора та вузлом BlockDAG, наступна:

```
type SyncMode uint8
const (
    NoSync SyncMode = iota
    MainSync
    HeadSync
)
type FinalizationParams struct {
    Spines    common.HashArray `json:"spines"`
    BaseSpine *common.Hash    `json:"baseSpine"`
    Checkpoint *Checkpoint    `json:"checkpoint"`
    ValSyncData []*ValidatorSync `json:"valSyncData"`
    SyncMode  SyncMode
}
```

де:

- Spines – хребетні блоки, за оптимістичним алгоритмом з чекпойнт;
- BaseSpine – останній завершений хребетних блок;
- Checkpoint – остання завершена контрольно-пропускна точка;
- ValSyncData – інформація про операції валідатора, які з'явилися з попередньої контрольної точки;
- SyncMode – тип синхронізації, NoSync - звичайний режим, працює кожен слот, MainSync - синхронізація по контрольним точкам, HeadSync - синхронізація від останньої контрольної точки до останнього відомого блока

Координуючий вузол. Консенсус в координаційній мережі досягається

через атестації. Всі координатори діляться на комітети, а ті, в свою чергу, належать до слоту. Координатор має ролі в цьому комітеті:

- творець блоку,
- атестатор блоку,
- атестаційний агрегатор.

Атестація. Атестатор підписує і відправляє в підмережу комісії атестацію на блок, згідно алгоритму пошуку останнього допустимого блоку. Основні структури наступні:

```
type Checkpoint struct {
    Epoch github_com_prysmaticlabs_eth2_types.Epoch
    Root []byte
},
```

де:

- Epoch – епоха чекпойнта,
- Root – корінь стейту блока чекпойнта.

```
type AttestationData struct {
    Slot github_com_prysmaticlabs_eth2_types.Slot
    CommitteeIndex github_com_prysmaticlabs_eth2_types.CommitteeIndex
    BeaconBlockRoot []byte
    Source *Checkpoint
    Target *Checkpoint
},
```

де:

- Slot – слот, в якому працює атестатор;
- CommitteeIndex – індекс комітету в слоті, так як в слоті може бути кілька комітетів;
- BeaconBlockRoot – хеш блоку, в якому лежать хребетні блоки, з яким погоджується атестатор;
- Source – голосування за останній чекпойнт, який валидатор вважає валідним;

– Target – голосування за перший блок в попередній епосі.

```
type Attestation struct {
    AggregationBits github_com_prysmaticlabs_go_bitfield.Bitlist
    Data *AttestationData
    Signature []byte,
},
```

де:

– AggregationBits – для атестатора по суті називається індексом, який необхідний для агрегації підписів, а агрегатор вказує індекси всіх сертифікатів, підписи, які він агрегував;

– Data – дані, за які проголосував атестатор;

– Signature – для Атестатора це його підпис, для агрегатів – це агрегований підпис всіх атестаторів, зазначений в AggregationBits.

BlockDAG. У координуючій мережі існує консенсус щодо впорядкування блоків BlockDAG. Для цього сугестивний блок додає наступну інформацію:

```
type BlockDAG1Data struct {
    DepositRoot []byte
    DepositCount uint64
    BlockHash []byte
    Candidates []byte
},
```

де:

– DepositRoot – корінь дерева Меркла, розрахований для депозитів;

– DepositCount – кількість відправлених депозитів;

– BlockHash – BlockDAG блока, для якого бралась інформація про депозити;

– Candidates – останні хребетні блоки від чекпойнта, які не ввійшли в фіналізований ланцюжок та префікс.

Slashings. За порушення роботи в мережі накладаються штрафи, той хто перший виявляє порушення, отримує за це винагороду. На даний момент є 2

порушення:

1. Пропонуючий надіслав 2 різних блоки. Для цього в блок I додається наступна інформація:

```
type BeaconBlockHeader struct {
    Slot github_com_prysmaticlabs_eth2_types.Slot
    ProposerIndex github_com_prysmaticlabs_eth2_types.ValidatorIndex
    ParentRoot []byte
    StateRoot []byte
    BodyRoot []byte
},
```

де:

- Slot – слот, в якому створювався блок;
- ProposerIndex – координатор, який створив блок;
- ParentRoot – батьківський блок, на який посилається поточний;
- StateRoot – хеш стейта блоку;
- BodyRoot – хеш вмісту блоку.

Підписаний Header блоку створювачем блоку

```
type SignedBeaconBlockHeader struct {
    Header *BeaconBlockHeader
    Signature []byte
},
```

де:

- Header – найменування блоку;
- Signature – підпис координатора, який створив блок.

Доказ того, що координатор відправив 2 блоки:

```
type ProposerSlashing struct {
    Header_1 *SignedBeaconBlockHeader
    Header_2 *SignedBeaconBlockHeader
},
```

де:

- Header_1 – підписаний заголовок одного блоку;
- Header_2 – підписаний заголовок другого блоку.

2. Атестатор відправив 2 різних голоси, для цього в блок добавляється наступна інформація:

Атестація та підпис

```
type IndexedAttestation struct {
    AttestingIndices []uint64
    Data *AttestationData
    Signature []byte
},
```

де:

– AttestingIndices – індекс координатора в загальному списку координаторів;

- Data – дані про те, що відправив атестатор;
- Signature – підпис атестатору.

Доказ про те, що координатор відправив 2 атестації:

```
type AttesterSlashing struct {
    Attestation_1 *IndexedAttestation
    Attestation_2 *IndexedAttestation
},
```

де:

- Attestation_1 – підписана перша атестація;
- Attestation_2 – підписана друга атестація.

Coordinators. В BlockDaG вузлі Worker відправляє транзакції депозиту та виходу, після того як транзакція фіналізується в BlockDaG мережі дані транзакції ловяться координуючим вузлом, та він блок, що запропонований, добавляє в блок.

Для додавання депозиту використовуються структури:

```
type Deposit_Data struct {
    PublicKey []byte
```

```

CreatorAddress []byte
WithdrawalCredentials []byte
Amount uint64
Signature []byte
};
type Deposit struct {
Proof [][]byte
Data *Deposit_Data
},

```

де:

- PublicKey – громадський ключ координатора;
- CreatorAddress – адреса валидатору;
- WithdrawalCredentials – адреса ключа, який може ініціювати процедуру

деактивації та забрати стейк та заробіток;

- Amount – сума поповнення стейку;
- Signature – підпис координатора.

```

type VoluntaryExit struct {
Epoch github_com_prysmaticlabs_eth2_types.Epoch
ValidatorIndex github_com_prysmaticlabs_eth2_types.ValidatorIndex
}
type SignedVoluntaryExit struct {
Exit *VoluntaryExit
Signature []byte
}

```

Блоки. Як і в BlockDAGt вузлі, для зменшення передачі інформації інформація про блок представлена в декількох частинах:

- BeaconBlock – коротка інформація про блок,
- SignedBeaconBlock – підпис блоку,
- BeaconBlockBody - повна інформація про блок.

BeaconBlock є такою структурою:

```

type BeaconBlock struct {
    Slot github_com_prysmaticlabs_eth2_types.Slot
    ProposerIndex github_com_prysmaticlabs_eth2_types.ValidatorIndex
    ParentRoot []byte
    StateRoot []byte
    Body *BeaconBlockBody
},

```

де:

- Slot – слот, в якому було створено блок;
- ProposerIndex – індекс координатора, який створив блок;
- ParentRoot – батьківський блок, на який посилається поточний блок;
- StateRoot – хеш стейта блоку;
- Body – додаткова інформація про блок.

SignedBeaconBlock є такою структурою:

```

type SignedBeaconBlock struct {
    Block *BeaconBlock
    Signature []byte
},

```

де:

- Block – блок, який підписано;
- Signature – підпис координатора, який створив блок;
- BeaconBlockBody – містить основну інформацію про блок:

```

type BeaconBlockBody struct {
    RandaoReveal []byte
    BlockDAG1Data *BlockDAG1Data
    Graffiti []byte
    ProposerSlashings []*ProposerSlashing
    AttesterSlashings []*AttesterSlashing
    Attestations []*Attestation
    Deposits []*Deposit
}

```

VoluntaryExits []*SignedVoluntaryExit

},

де:

- BlockDAG1Data – інформація про стан BlockDAG мережі;
- ProposerSlashings – доказ про порушення того, хто пропонує блок;
- AttesterSlashings – доказ про порушення атестаторів;
- Attestations – атестації, які не були раніш додані в блок;
- Deposits – нові депозити, які появились в BlockDAG мережі;
- VoluntaryExits – нові запити на вихід, які з'явилися в BlockDAG мережі.

4.5 Реалізація та випробування платформи Waterfall

На рис. 4.3 приведено одну з екранних форм платформ Waterfall .

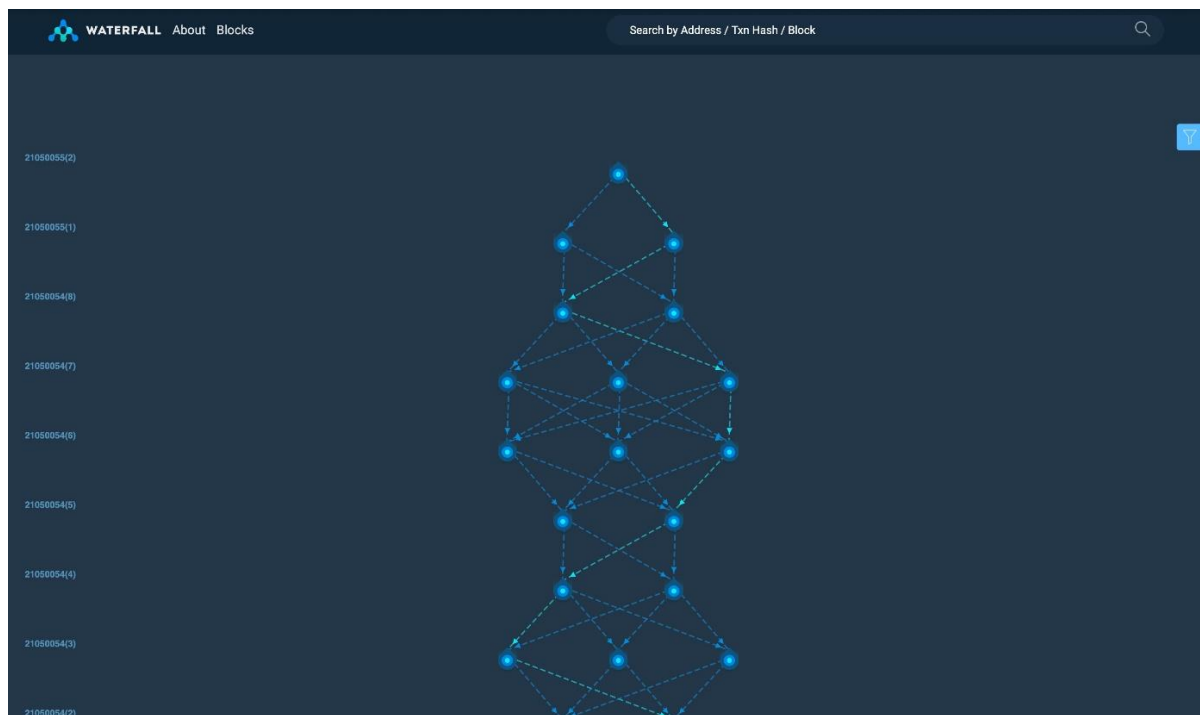


Рис. 4.3. Знімок екрану результуючого лінійно-топологічного упорядкування BlockDAG

Проведені експерименти системи щодо навантаження. Тестову мережу було сформовано на базі серверів Amazon Elastic Compute Cloud. Тестова

мережа працювала на 64 екземплярах t3.small (два ядра ЦП і 2 ГБ пам'яті) Amazon EC2. В ході експериментів був згенерований пул приблизно з 100 000 транзакцій та зафіксований час, за який остання з них буде записана в реєстр. Були зроблені заміри масштабованості. Середня швидкість складала 2234 tps. Модифікована версія Waterfall обробляла більш 3600 транзакцій в секунду при загрузці ЦП менш 20%.

На рис.4.4 приведено один з екранів з результатами тестування.

WATERFALL
STOP WAR IN UKRAINE
DONATE UKRAINIAN ARMY

Testing Results > DAPPs > Testnet Tools > Documentation > Become a validator

Testnet 3 Test 9

For the test, we used Amazon EC2 T3 Instances. With the following server characteristics:

Name	vCPUs	Memory (GiB)	Baseline Performance/vCPU	CPU Credits earned/hr	Network burst bandwidth (Gbps)	EBS burst bandwidth (Mbps)
t3a.small	2	2.0	20%	24	5	Up to 2,085

Algorithm of actions during testing:

1. Stop creating blocks in the tested network.
2. Generates ~96,000 transactions.
3. Start 64 nodes.
4. Fix the time in which all these transactions were added and accepted into blocks, i.e. executed.

Let's analyze the line from the test results below:

Block number: 35144
Number of transactions: 10000
Age(slot): 20594179(6)
Link to detailed information: <https://explorer3.waterfall.network/block/35144>

Block number: 35145
Number of transactions: 10000
Age(slot): 20594179(6)
Link to detailed information: <https://explorer3.waterfall.network/block/35145>

Block number: 35146
Number of transactions: 10000

Рис. 4.4. Приклад екрану з результатами тестування з навантаженням

Поточна реалізація технології Waterfall сумісна з EVM, що полегшує використання смарт-контрактів. Смарт-контракти для Ethereum можна копіювати та вставляти, при цьому вони успішно працюють у тестовій мережі Waterfall. Експерименти показали, що для імпорту більшої кількості децентралізованих додатків може знадобитися незначна зміна їхнього коду.

Безпека реалізації системи підтримується тими самими криптографічними протоколами, які добре зарекомендували себе мережі Ethereum.

Хеш-функції – основна мережа – Кессак-256/SHA3, координаційна мережа – SHA256.

Цифрові підписи для формування ключів – вузли основної мережі – ECDSA (Elliptic Curve Digital Signature Algorithm) secp256k1, вузли координаційної мережі – підпис BLS (Boneh-Lynn-Shacham). Встановлено рекомендовану конфігурацію для запуску вузла:

- число ЦП - 2;
- оперативна пам'ять - 4 Гб;– SSD – 80 Гб ;
- Інтернет трафік – 400 Гб / міс.

4.6 Висновки до розділу 4

Розроблено експериментальну платформу Waterfall.

Розроблена платформа Waterfall успадковує основні принципи Ethereum 2.0 та розвиває їх. Завдяки розробленим моделям та методам платформа збільшує такі показники ефективності:

1. Висока швидкість обробки транзакцій при достатньому рівні масштабованості - масштабовані в системі блокові структури на основі DAG дозволяють одночасно публікувати кілька блоків. Це формує DAG та забезпечує завершеність всіх транзакцій за умови, що блоки не конфліктують один з одним. Тому Waterfall може одночасно обробляти карти Visa, MasterCard та Union Pay на децентралізованому рівні навіть у години пік. Згідно з останніми проміжними лабораторними тестами, система могла обробляти 3600 транзакцій в секунду. Для порівняння зазначимо, що Visa обробляє близько 1700 транзакцій на секунду.

2. Забезпечується висока захищеність платформи від можливих маніпуляцій при використанні операцій квазівипадкового вибору при здійсненні процедури консенсусу PoS за рахунок застосування розробленого методу перемішування.

Крім того, платформа Waterfall забезпечує наступні переваги, порівняно з аналогічними технічними рішеннями:

1. Низькі комісії за транзакції – архітектура спроектована таким чином,

щоб підтримувати мінімальні комісії у різних сценаріях. Протокол динамічно масштабується зі зростанням навантаження на мережу. У той час, як продуктивність всієї системи збільшується, в тому самому слоті одночасно публікується більше блоків, а транзакційні збори знижуються в міру масштабування системи. Це знижує кількість операцій у пулі транзакцій навіть у години пік.

2. Низький фінансовий поріг входу.

3. Платформа обслуговує вбудовані токени — випуск та обслуговування tokenів (включно з NFT) не потребують спеціальних смарт-контрактів, а виконуються за допомогою звичайних транзакцій, що значно знижує накладні витрати. Крім того, це робить їх використання більш доступним для широкого кола користувачів.

4. Динамічна настройка – платформа має механізми динамічної адаптації параметрів системи залежно від ситуації, що змінюється. Зокрема, час слоту, оптимальна кількість Worker'ів та деякі інші параметри налаштовуються автоматично.

Таким чином, платформа Waterfall забезпечує сприятливе середовище для надання та споживання широкого спектру послуг для ведення бізнесу та соціальної діяльності у зручному форматі загальнодоступного децентралізованого реєстру. Платформа призначена для обслуговування транзакцій з різними токенами, включаючи NFT, обслуговування смарт-контрактів і розробки розподілених додатків. Тестування системи показало, що вона може обробляти в середньому 2234 транзакції за секунду при досить високому рівні масштабованості.

Таким чином, підтверджено спроможність та правильність технічних рішень, заснованих на результатах дисертаційного дослідження.

ЗАГАЛЬНІ ВИСНОВКИ

Дисертаційне дослідження присвячено вирішенню актуальної науково-технічної задачі – підвищенню ефективності розподілених реєстрів шляхом розробки моделей та методів обробки транзакцій у них для широкої низки практичних застосувань.

У рамках виконаних досліджень отримано такі наукові та практичні результати.

1. Розроблено двошарову модель розподіленого реєстру, що поєднує в собі основну мережу, побудовану за технологією BlockDAG, та координаційну мережу, що базується на традиційній блокчейн технології, що дозволило підвищити швидкість обробки транзакцій при суттєвому масштабуванні.

2. Запропоновано модель скороченого обміну інформацією між BlockDAG мережею та координаційною мережею на основі поняття хребетного блоку, що дозволило скоротити обсяг трафіку між мережею BlockDAG та координаційною мережею і тим самим зменшити навантаження на систему загалом та підвищити швидкість обробки транзакцій. Розроблена система з обробляє до 3600 транзакцій в секунду (в порівняння з системою Visa, яка обробляє близько 1700 транзакцій на секунду);

3. Розроблено метод прискореного PoS консенсусу в двошаровій моделі розподіленого реєстру на основі k -кратної перевірки повторення початкової частини ланцюжка хребетних блоків і голосування за ці блоки більше половини учасників відповідних слотів, що дозволило зменшити час отримання рішення до декількох секунд із ймовірністю, не меншою 0,9. Метод забезпечив високу захищеність платформи від можливих маніпуляцій;

4. Запропоновано метод перемішування даних за допомогою бієктивного відображення, яке є суперпозицією лінійного та найпростішого нелінійного відображення q -Tent, що дозволило здійснити процес якісного перемішування при формуванні PoS консенсусу.

5. Розроблено методику визначення оптимального розміру комітету та

розподілу Worker'ів залежно від їх загальної кількості в мережі та очікуваної помилкової частки. Проведені імітаційні експерименти показали, що оптимальна кількість як комітетів, так і членів відповідає схемі $3f + 1$, що відповідає мінімальній кількості необхідних повідомлень. Отримані результати моделювання показали значне покращення продуктивності системи та зниження навантаження на мережні вузли.

6. Розроблено експериментальну платформу Waterfall. Розроблена платформа Waterfall успадковує основні принципи Ethereum 2.0 та розвиває їх. Платформа Waterfall забезпечує сприятливе середовище для надання та споживання широкого спектру послуг для ведення бізнесу та соціальної діяльності у зручному форматі загальнодоступного децентралізованого реєстру. Платформа призначена для обслуговування транзакцій з різними токенами, включаючи NFT, обслуговування смарт-контрактів і розробки розподілених додатків. Тестування системи показало, що вона може обробляти в середньому 2234 транзакції за секунду при досить високому рівні масштабованості.

7. Розроблені в роботі методи та інструментальні засоби отримали впровадження у діяльності Комунального підприємства «Теплопостачання міста Одеси» та знайшли відображення у навчальному процесі та науковій діяльності Національного університету «Одеська політехніка».

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Sunyaev A. Distributed Ledger Technology. *Internet Computing*. 2020. P. 265–299.
2. Fundamentals of Blockchain and Distributed Ledger Technology (DLT) / Roselin G. Leema, R. Rajmohan, S. Usharani, K. Kiruba, P. Manjubala. *Recent Trends in Blockchain for Information Systems Security and Privacy*. 2021. CRC Press, P. 5–41.
3. Miles Gietzmann, Francesco Grossetti. Blockchain and other distributed ledger technologies: Where is the accounting? *Journal of Accounting and Public Policy*. 2021. Vol. 40, Is. 5. 106881.
4. Perdana A., Robb A., Balachandran V., Rohde F. Distributed Ledger Technology: Its Evolutionary Path and the Road Ahead. *Information & Management*. 2021. Vol. 58, Is. 3. 103316.
5. Куперштейн Л.М., Кренцін М.Д. Аналіз тенденцій розвитку пірингових мереж. *Вісник Хмельницького національного університету*. 2021. №4, (299). С. 25–29.
6. Secure Peer-to-peer Networks for Trusted Collaboration. URL: <https://core.ac.uk/download/pdf/192229218.pdf>. (дата звернення: 15.07.2023).
7. Balfe S., Lakhani A. D., Paterson K. G. Trusted computing: providing security for peer-to-peer networks. *Fifth IEEE International Conference on Peer-to-Peer Computing (P2P'05)*. 2005. P. 117–124.
8. Chaum D. L. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*. 1981. 24 (2): 84–90.
9. Chaum D. Blind signatures for untraceable payments. *Advances in cryptology*. / Eds. Chaum D., Rivest R.L., Sherman A.T. Springer, Boston, MA, 1983. P. 199–203.
10. Merkle R. C. A Digital Signature Based on a Conventional Encryption Function. *Advances in Cryptology – CRYPTO '87. Lecture Notes in Computer Science*. 1988. Vol. 293. P. 369–378. DOI: https://doi.org/10.1007/3-540-48184-2_32.

11. Liu H., Luo X., Liu H., Xia X. Merkle Tree: A Fundamental Component of Blockchains. *2021 International Conference on Electronic Information Engineering and Computer Science (EIECS)*. Changchun, China, 2021. P. 556–561.
12. Bach L. M., Mihaljevicand B., Zagar M. Comparative analysis of blockchain consensus algorithms. *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. Opatija, Croatia, 2018. P. 1545–1550, DOI: 10.23919/MIPRO.2018.8400278.
13. A review on consensus algorithm of blockchain / D. Mingxiao, M. Xiaofeng, Z. Zhe, W. Xiangwei and C. Qijun. *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. Banff, AB, Canada, 2017. P. 2567–2572, doi: 10.1109/SMC.2017.8123011.
14. Yusoff J., Mohamad Z., Anuar M. A Review: Consensus Algorithms on Blockchain. *Journal of Computer and Communications*. 2022. 10, 37–50. DOI: <https://doi.org/10.4236/jcc.2022.109003>.
15. Anwar H. Consensus Algorithms: The Root of Blockchain Technology. *101 Blockchains*. 2018. URL: <https://101blockchains.com/consensus-algorithms-blockchain/>. (дата звернення: 15.03.2023).
16. Dwork C., Naor M. Pricing via Processing or Combatting Junk Mail / Ed. Brickell E.F. *Advances in Cryptology — CRYPTO' 92. CRYPTO 1992. Lecture Notes in Computer Science*. 1993. Vol. 740. P. 139–147. DOI: https://doi.org/10.1007/3-540-48071-4_10.
17. Dwork C., Naor M., Wee H. Pebbling and Proofs of Work / Ed. Shoup V. *Advances in Cryptology – CRYPTO 2005. CRYPTO 2005. Lecture Notes in Computer Science*. 2005. Vol. 3621. P. 37–54. DOI: https://doi.org/10.1007/11535218_3.
18. Ghimire S., Selvaraj H. A Survey on Bitcoin Cryptocurrency and Its Mining. *2018 26th International Conference on Systems Engineering*. Sydney, 18-20 December 2018. 2019. P. 1–6. DOI: <https://doi.org/10.1109/ICSENG.2018.8638208>.
19. Salimitari M., Chatterjee M. A Survey on Consensus Protocols in Blockchain for IoT Networks. 2018. URL: <http://arxiv.org/abs/1809.05613>. (дата звернення: 15.03.2023).

20. Know Everything About Blockchain Proof of Work (PoW). *101 Blockchains*. 2019. URL: <https://101blockchains.com/blockchain-proof-of-work/>. (дата звернення: 15.03.2023).
21. Tschorsch F., Scheuermann B. Bitcoin and Beyond: A Technical Survey on Decentralized Digital Currencies. *IEEE Communications Surveys & Tutorials*. 2016. 18. P. 2084–2123. DOI: <https://doi.org/10.1109/COMST.2016.2535718>.
22. King S., Nadal S. PPCoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake. 2012. URL: <https://decred.org/research/king2012.pdf>. (дата звернення: 15.03.2023).
23. Zhang S., Lee J.-H. Analysis of the Main Consensus Protocols of Blockchain. *ICT Express*. 2020. 6. P. 93–97. DOI: <https://doi.org/10.1016/j.icte.2019.08.001>.
24. Fu X., Wang H., Shi P. A Survey of Blockchain Consensus Algorithms: Mechanism, Design and Applications. *Science China Information Sciences*. 2021. 64. ArticleID: 121101. DOI: <https://doi.org/10.1007/s11432-019-2790-1>.
25. Delegated Proof-of-Stake (DPOS). *Ciberexplosion*. 2014. URL: <https://www.geeksforgeeks.org/delegated-proof-of-stake/>.
26. Zhang S., Lee J.-H. Analysis of the Main Consensus Protocols of Blockchain. *ICT Express*. 2020. 6. P. 93–97. DOI: <https://doi.org/10.1016/j.icte.2019.08.001>.
27. Lamport L., Shostak R., Pease M. The Byzantine Generals Problem. *Dr. Dobbs's Journal*. 2008. 33. P. 30–36.
28. Alsunaidi S.J., Alhaidari F.A. A Survey of Consensus Algorithms forBlockchain Technology. *2019 International Conference on Computer and Information Sciences*. Sakaka, 3-4 April 2019. 2019. P. 1–6. DOI: <https://doi.org/10.1109/ICCISci.2019.8716424>.
29. Castro M. Practical Byzantine Fault Tolerance. *Third Symposium on Operating Systems Design and Implementation*. New Orleans, 22-25 February 1999. 2001. P. 1–14. URL: <http://pmg.csail.mit.edu/papers/osdi99.pdf>.
30. Du M., Ma X., Zhang Z., Wang X., Chen Q. A Review on Consensus

Algorithm of Blockchain. *2017 IEEE International Conference on Systems, Man, and Cybernetics*. Banff, 5-8 October 2017. 2017. P. 2567–2572. DOI: <https://doi.org/10.1109/SMC.2017.8123011>.

31. Кравченко П., Скрябін Б., Дубініна О. Блокчейн і децентралізовані системи : навч. посібник у 3 ч. Харків : ПРОМАРТ, 2019. Ч. 1. 452 с.

32. Skh Saad S. M., Raja Mohd Radzi R. Z. Comparative Review of the Blockchain Consensus Algorithm Between Proof of Stake (POS) and Delegated Proof of Stake (DPOS). *International Journal of Innovative Computing*. 2020. 10(2). DOI: <https://doi.org/10.11113/ijic.v10n2.272>.

33. Distributed ledger technology: Applications and implications / Maull R., Godsiff P., Mulligan C., Brown A., Kewell B. *Strategic Change*. 2017. 26(5). P. 481–489. DOI: <https://doi.org/10.1002/jsc.2148>.

34. Hairudin A., Sifat I. M., Mohamad A., Yusof Y. Cryptocurrencies: A survey on acceptance, governance and market dynamics. *International Journal of Finance & Economics*. 2020. 27(4). P. 4633–4659. DOI: <https://doi.org/10.1002/ijfe.2392>.

35. Can Inci, Rachel Lagasse. Cryptocurrencies: applications and investment opportunities. *Journal of Capital Markets Studies*. 2019. Vol. 3, No. 2. P. 98–112. DOI: <https://doi.org/10.1108/JCMS-05-2019-0032>.

36. Tredinnick L. Cryptocurrencies and the blockchain. *Business Information Review*. 2019. 36(1). P. 39–44.

37. How Many Cryptocurrencies Are There? *Forbes Advisor*. URL: <https://www.forbes.com/advisor/investing/cryptocurrency/different-types-of-cryptocurrencies/#:~:text=How%20Many%20Cryptocurrencies%20Are%20There,market%20capitalization%20of%20%241.1%20trillion>.

38. Chohan, Usman W. Decentralized Finance (DeFi): An Emergent Alternative Financial Architecture. *Critical Blockchain Research Initiative (CBRI) Working Papers*. 2021. P. 3–11. URL: <https://ssrn.com/abstract=3791921>.

39. Salami I. Challenges and Approaches to Regulating Decentralized Finance. *AJIL Unbound*. 2021. 115. P. 425–429. DOI: 10.1017/aju.2021.66.

40. DeFi-ning DeFi: Challenges & Pathway / H. Amler, L. Eckey, S. Faust, M. Kaiser, P. Sandner and B. Schlosser. 2021 3rd Conference on Blockchain Research. *Applications for Innovative Networks and Services (BRAINS)*. Paris, France. 2021. P. 181–184. DOI: 10.1109/BRAINS52497.2021.9569795.
41. A. Smorodin. Predictive control methods in tasks of searching saddle points. *Proceedings of Odessa Polytechnic University: Scientific, science and technology collected articles*. Odesa, 2020. Issue 3(62). P. 80–90. DOI: 10.15276/opu.3.62.2020.10.
42. Abdulhakeem S., Hu Q. Powered by Blockchain Technology, DeFi (Decentralized Finance) Strives to Increase Financial Inclusion of the Unbanked by Reshaping the World Financial System. *Modern Economy*. 2021. 12. P. 1–16.
43. Ghosh B., Kazouz H., Umar Z. Do Automated Market Makers in DeFi Ecosystem Exhibit Time-Varying Connectedness during Stressed Events? *J. Risk Financial Manag.* 2023. 16(5). 259.
44. Доценко О. В. Архітектура децентралізованих додатків у складі екосистеми DeFi. *Соціальна економіка*. 2022. 64. P. 117–124.
45. L. Besançon, C. F. Da Silva, P. Ghodous, J. -P. Gelas. A Blockchain Ontology for DApps Development. *IEEE Access*. 2022. Vol. 10. P. 49905–49933.
46. Blockchain technology applications: a literature review / Meiriño M. J., Méxas M. P., Faria A. V. et al. *Brazilian Journal of Operations & Production Management*. 2019. Vol. 16, No. 4. P. 672–684. URL: <https://bjopm.emnuvens.com.br/bjopm/article/view/585>.
47. Novo O. Blockchain Meets IoT: an Architecture for Scalable Access Management in IoT. *IEEE Internet of Things Journal*. 2018. Vol. 5, No. 2. P. 1184–1195.
48. Pradip Kumar Sharma, Saurabh Singh, Young-Sik Jeong, Jong Hyuk Park. DistBlockNet: A Distributed Blockchains-Based Secure SDN Architecture for IoT Networks. *IEEE Communications Magazine*. 2017. Vol. 55, No. 9. P. 78–85. DOI: 10.1109/MCOM.2017.1700041.
49. Zhang Y., Wen J. The IoT electric business model: Using blockchain

technology for the internet of things. *Peer-to-Peer Networking and Applications*. 2017. Vol. 10, No. 4. P. 983–994.

50. Jafar U., Aziz MJA, Shukur Z. Blockchain for Electronic Voting System-Review and Open Research Challenges. *Sensors (Basel)*. 2021. 21(17). 5874. DOI: 10.3390/s21175874.

51. Syada Tasmia Alvi, Mohammed Nasir Uddin, Linta Islam, Sajib Ahamed, DVTChain: A blockchain-based decentralized mechanism to ensure the security of digital voting system. *Journal of King Saud University - Computer and Information Sciences*. 2022. Vol. 34, Is. 9. P. 6855–6871. DOI: 10.1016/j.jksuci.2022.06.014.

52. Blockchain technology applications in healthcare: An overview / Abid Haleem, Mohd Javaid, Ravi Pratap Singh, Rajiv Suman, Shanay Rab. *International Journal of Intelligent Networks*. 2021. Vol. 2. P. 130–139. DOI: 10.1016/j.ijin.2021.09.005.

53. Applications of Blockchain in the Medical Field: Narrative Review / Xie Y, Zhang J, Wang H, Liu P, Liu S, Huo T, Duan YY, Dong Z, Lu L, Ye Z. *J Med Internet Res*. 2021. 23(10). e28613. DOI: 10.2196/28613.

54. Blockchain, consent and present for medical research / Porsdam Mann S., Savulescu J., Ravaud P., et al. *Journal of Medical Ethics*. 2021. 47. P. 244–250. DOI: 10.1136/medethics-2019-105963.

55. Taherdoost H., Madanchian M. Blockchain-Based New Business Models: A Systematic Review. *Electronics*. 2023. 12. 1479. DOI: <https://doi.org/10.3390/electronics12061479>.

56. Erceg A., Damoska Sekuloska J., Kelić I. Blockchain in the tourism industry—A Review of the situation in Croatia and Macedonia. *Informatics*. 2020. 7(1). 5. DOI: <https://doi.org/10.3390/informatics7010005>.

57. Hanna Haponenko, Vladyslav Vasylenko, Olena Yevtushenko, Irina Shamara. Problems and prospects of implementation of blockchain technologies in the tourist industry of Ukraine. *The Journal of V. N. Karazin Kharkiv National University. Series: International Relations. Economics. Country Studies. Tourism*. 2021. (14). P. 142–153.

58. Manski S. Building the blockchain world: Technological commonwealth or just more of the same? *Strategic Change*. 2017. 26(5). P. 511–522. DOI: <https://doi.org/10.1002/jsc.2151>.
59. White G.R. Future applications of blockchain in business and management: A Delphi study. *Strategic Change*. 2017. 26. P. 439–451.
60. Taherdoost, H. A Critical Review of Blockchain Acceptance Models—Blockchain Technology Adoption Frameworks and Applications. *Computers*. 2022. 11(2). P. 24–29. DOI: <https://doi.org/10.3390/computers11020024>.
61. Chen Y., Bellavitis C. Blockchain disruption and decentralized finance: The rise of decentralized business models. *J. Bus. Ventur. Insights*. 2020. 13. e00151. DOI: <https://doi.org/10.1016/j.jbvi.2019.e00151>.
62. Viriyasitavat W., Xu L.D., Bi Z., Pungpapong V. Blockchain and Internet of Things for Modern Business Process in Digital Economy—The State of the Art. *IEEE Trans. Comput. Soc. Syst.* 2019. 6. P. 1420–1432. DOI: 10.1109/TCSS.2019.2919325.
63. Blockchain, business and the fourth industrial revolution: Whence, whither, wherefore and how? / Kimani D., Adams K., Attah-Boakyie R., Ullah S. Frecknall-Hughes J., Kim J. *Technol. Forecast. Soc. Chang.* 2020. 161. 120254. DOI: 10.1016/j.techfore.2020.120254.
64. Brilliantova V., Thurner T.W. Blockchain and the future of energy. *Technol. Soc.* 2019. 57. P. 38–45. DOI: 10.1177/0975425319832392.
65. Kundu D. Blockchain and trust in a smart city. *Environ. Urban. ASIA*. 2019. 10. 31–43.
66. Lage O., Saiz-Santos M., Zarzuelo J.M. Decentralized platform economy: Emerging blockchain-based decentralized platform business models. *Electron. Mark.* 2022. 32. P. 1707–1723. DOI: 10.1007/s12525-022-00586-4.
67. Blockchain: A paradigm shift in business practices / Kizildag M., Dogru T., Zhang T., Mody M.A., Altin M., Ozturk A.B., Ozdemir O. *Int. J. Contemp. Hosp. Manag.* 2020. Vol. 32 (3). P. 953–975. DOI: <https://doi.org/10.1108/IJCHM-12-2018-0958>.

68. Satoshi Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System. URL: <https://bitcoin.org/bitcoin.pdf>. (дата звернення: 15.03.2023).
69. A comparative study on consensus mechanism with security threats and future scopes: Blockchain / A.K. Yadav, K. Singh, A.H. Amin et al. *Computer Communications*. 2023. 201. P. 102–115. DOI: <https://doi.org/10.1016/j.comcom.2023.01.018>.
70. Global Cryptocurrency Market Cap Charts. *CoinGecko*. URL: <https://www.coingecko.com/en/global-charts>.
71. Wood G. Ethereum: A secure decentralized generalized transaction ledger. *Git Hubrepository*. 2019. 39 p. URL: <https://ethereum.github.io/yellowpaper/paper.pdf>. (дата звернення: 15.03.2023).
72. Nick Szabo. Smart Contracts. 1994. URL: <http://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart.contracts.html>. (дата звернення: 15.03.2023).
73. Chris Dannen. Introducing Ethereum and Solidity. Springer, 2017. 185 p.
74. Що таке смарт-контракт? *Bankchart*. URL: https://bankchart.com.ua/finansoviy_gid/investitsiyi/statti/scho_take_smart_kontrakt_. (дата звернення: 15.03.2023).
75. Що таке Ethereum 2.0? *Kraken*. URL: <https://www.kraken.com/uk-ua/learn/ethereum-2-0>. (дата звернення: 15.03.2023).
76. Kendrick Lau. Ethereum 2.0. An Introduction. *CRYPTO.COM*. 2020. URL: https://assets.ctfassets.net/hfgyig42jimx/7j3AVp5aCnx2Ct2P6T6kY/e8991d5da1972d5d760233868f237609/Crypto.com_Macro_Report_-_Ethereum_2.0.pdf. – (дата звернення: 15.03.2023).
77. Kapengut, Elie, and Bruce Mizrach. 2023. "An Event Study of the Ethereum Transition to Proof-of-Stake" *Commodities* 2, No. 2: 96-110.
78. Said A., Janjua M.U., Hassan S., Muzammal Z., Saleem T., Thaipsisutikul T., Tuarob S., Nawaz R. 2021. Detailed analysis of Ethereum network on transaction behavior, community structure and link prediction. *Peer J. Computer Science*. 7:e815 <https://doi.org/10.7717/peerj-cs.815>

79. Kiayias A., Russell A., David B., Oliynykov R. Ouroboros: A Provably Secure Proof-of-Stake Blockchain Protocol. *Advances in Cryptology – CRYPTO 2017*. 2017. Vol. 10401. P. 357–388.

80. Что такое Ethereum 2.0? *Forklog*. 2020. URL: <https://forklog.com/cryptorium/chto-takoe-ethereum-2-0>. (дата звернення: 15.03.2023).

81. Benčićand F. M., Žarko I. Podnar. Distributed Ledger Technology: Blockchain Compared to Directed Acyclic Graph. *IEEE 38th International Conference on Distributed Computing Systems*. 2018. P. 1569–1570. DOI: 10.1109/ICDCS.2018.00171.

82. Pervez H., Muneeb M., Irfanand M. U., Haq I. U. A Comparative Analysis of DAG-Based Blockchain Architectures. *2018 12th International Conference on Open Source Systems and Technologies (ICOSST)*. 2018. P. 27–34. DOI: 10.1109/ICOSST.2018.8632193.

83. Wang Q., Yu J., Chen S., Xiang Y. SoK: Diving into DAG-based Blockchain Systems. *Cornell University*. 2022. arXiv:2012.06128.

84. Tian H., Lin H., Zhang F. Design a Proof of Stake Based Directed Acyclic Graph Chain. *Frontiers in Cyber Security. FCS 2020*. Communications in Computer and Information Science. Springer, Singapore. 2020. Vol. 1286. P. 150–165. DOI: 10.1007/978-981-15-9739-8_13.

85. Robin J. Wilson. Introduction to Graph Theory. Addison Wesley Longman Limited, 1998. 180 p.

86. Alexandra Carrillo. An introduction to the blockDAG paradigm. 2018. URL: <https://ancapalex.medium.com/an-introduction-to-the-blockdag-paradigm-50027f44facb>. (дата звернення: 15.03.2023).

87. Sompolinsky Y., Lewenberg Y., Zohar A. SPECTRE: A Fast and Scalable Cryptocurrency Protocol. *IACR Cryptol. ePrintArch*. 2016. 1159. URL: <https://eprint.iacr.org/2016/1159>. (дата звернення: 15.03.2023).

88. SompolinskyY., Zohar A. PHANTOM: A Scalable BlockDAG Protocol. *IACR Cryptol. ePrint Arch*. 2018. 104.

89. Ключка Я.О., Шматко О.В. Порівняння технології блокчейн і спрямованого ациклічного графа при зберіганні і обробці даних в розподіленому реєстрі. *Вісник Національного технічного університету «ХПІ»*. Серія: Системний аналіз, управління та інформаційні технології. 2020. 1 (3). С. 106–116. DOI: doi.org/10.20998/2079-0023.2020.01.18.

90. Advanced Blockchain AG. DAG based DLT Network. URL: https://www.advancedblockchain.com/docs/DAG_ADVANCED_BLOCKCHAIN_AG.pdf. (дата звернення: 15.03.2023).

91. Kusmierz B. The first glance at the simulation of the Tangle: discrete model. IOTA Found. *White Paper*. 2017. P. 1–10. URL: https://assets.ctfassets.net/r1dr6vzfxhev/2ZO5XxwehymSMsgusUE6YG/f15f4571500a64b7741963df5312c7e7/The_First_Glance_of_the_Simulation_Tangle_-_Discrete_Model_v0.1.pdf. (дата звернення: 15.03.2023).

92. Anton Churymov. Byteball: A decentralized system for storage and transfer of value. *Byteball*. 2016. URL: <https://obyte.org/Byteball.pdf>. (дата звернення: 15.03.2023).

93. Leemon Baird. The swirlds hashgraph consensus algorithm: Fair, fast, byzantine fault tolerance. *Swirlds, Inc. Technical Report SWIRLDS-TR-2016*. 2016. 1. URL: www.swirlds.com/downloads/SWIRLDS-TR-2016-01.pdf. (дата звернення: 15.03.2023).

94. Colin LeMahieu. Nano: A feeless distributed crypto currency network. *Computer Science*. 2018. P. 1–8. URL: <https://www.exodus.com/assets/docs/nano-whitepaper.pdf>. (дата звернення: 15.03.2023).

95. Detailed analysis of Ethereum network on transaction behavior, community structure and link prediction / Said A., Janjua M.U., Hassan S., Muzammal Z., Saleem T., Thaipisutikul T., Tuarob S., Nawaz R. *PeerJ Computer Science*. 2021. DOI: <https://doi.org/10.7717/peerj-cs.815>.

96. Claude Berge. *The Theory of Graphs and Its Application*. Greenwood Press, 2002. 247 p.

97. Knuth D.E. *The Art of Computer Programming. Seminumerical*

Algorithms. Reading, Massachusetts: Addison-Wesley, 1997. Volume 2. 762 p.

98. L'Ecuyer Pierre. Software for uniform random number generation: Distinguishing the good and the bad. *Winter Simulation Conference Proceedings*. 2001. Vol. 1. P. 95–105. DOI: 10.1109/WSC.2001.977250.

99. Shannon C. E. Communication Theory of Secrecy Systems. *The Bell System Technical Journal*. 1949. Vol. 28, Iss. 4. P. 656–715.

100. A Novel Asymmetric Hyperchaotic Image Encryption Scheme Based on Elliptic Curve Cryptography / Liang H., Zhang G., Hou W., Huang P., Liu B., Li S. *Appl. Sci*. 2021. 11(12). 5691. DOI: 10.3390/app11125691.

101. Talhaoui M.Z., Wang X., Midoun M.A. A new one-dimensional cosine polynomial chaotic map and its use in image encryption. *Vis. Comput*. 2021. 37(08). DOI:10.1007/s00371-020-01822-8.

102. A Novel Hybrid Secure Image Encryption Based on Julia Set of Fractals and 3D Lorenz Chaotic Map / Masood F., Ahmad J., Shah S.A., Jamal S.S., Hussain I. *Entropy*. 2020. 22(3). 274. DOI: 10.3390/e22030274.

103. Alvarez G. Some basic cryptographic requirements for chaos-based cryptosystems. *International J. of Bifurcation and Chaos*. 2006. Vol. 16 (8). P. 2129–2151.

104. Devaney R. L. An Introduction to Chaotic Dynamical Systems. New York : Addison-Wesley Publ. Co., Second edition, 1993. 336 p.

105. Ott E., Grebodgi C., Yorke J.A. Controlling chaos. *Phys. Rev. Lett*. 1990. 64. P. 1196–1199.

106. Biham E. Cryptanalysis of the Chaotic-Map Cryptosystem. Advances in Cryptology. EUROCRYPT'91; ed. D.W. Davies. LNCS 547. Berlin : Springer-Verlag, 1991. 532 p.

107. Bauer F.L. Friedrich Ludwig . Decrypted Secrets: Methods and Maxims of Cryptology, 4th edition. New York : Springer, 2006. 474 p.

108. Hill L. S. Cryptography in an Algebraic Alphabet. *The American Mathematical Monthly*. 1929. Vol. 36, No. 6. P. 306–312.

109. Hill L. S. Concerning Certain Linear Transformation Apparatus of

Cryptography. *The American Mathematical Monthly*. 1931. Vol. 37. P. 135–154.

110. Feistel H. Cryptography and Computer Privacy. *Scientific American*. 1973. Vol. 228, No. 5. P. 15–23.

111. Teh J. S., Alawida M., Sii Y. C. Implementation and practical problems of chaos-based cryptography revisited. *Journal of Information Security and Applications*. 2020. Vol. 50. Article 102421.

112. Devaney R. L. An Introduction to Chaotic Dynamical Systems. CRC Press, 2nd Edition. 1989. 360 p.

113. Al-shameri W.F., Mahiub M.A. Some dynamical properties of the family of tent maps. *International Journal of Mathematical Analysis*. 2013. 7. P. 1433–1449.

114. The Lenna Story – www.lenna.org. URL: <http://www.lenna.org/>. (дата звернення: 15.03.2023).

115. Waterfall – масштабована платформа смарт-контрактів на основі DAG. URL: <https://www.waterfall.network/> (дата звернення: 15.03.2023).

СПИСОК ОПУБЛІКОВАНИХ ПРАЦЬ ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Наукові праці, в яких опубліковані основні наукові результати дисертації:

1. Grybniak S., Dmitrishin D. Basic principles for constructing mixing functions based on the simplest linear and nonlinear mappings. *Proceedings of Odessa Polytechnic University*. 2022. № 2(66). P. 100–109. DOI: 10.15276/opu.2.66.2022.12. (Реєстр наукових фахових видань України, категорія «Б»)

2. Mazurok I., Leonchyk Y., Grybniak S., Nashyvan O., Masalskyi R. An incentive system for decentralized DAG-based platforms. *Applied Aspects of Information Technology*. 2022. Vol. 5, № 3. P. 196–207. DOI: <https://doi.org/10.15276/aait.05.2022.13>. (Реєстр наукових фахових видань України, категорія «Б»)

3. Mazurok I., Leonchyk Y., Grybniak S., Vorokhta A., Nashyvan O. Multi-objective optimization of committee selection for hierarchical byzantine fault tolerance-based consensus protocols. *Herald of Advanced Information Technology*. 2023. Vol. 6, № 1. P. 39–53. DOI: <https://doi.org/10.15276/hait.06.2023.3>. (Реєстр наукових фахових видань України, категорія «Б»)

4. Грибняк С. С. Двошарова модель масштабованого розподіленого децентралізованого реєстру. *Наукові праці Вінницького національного технічного університету*. 2023. № 2. С. 120–127. (Реєстр наукових фахових видань України, категорія «Б»)

5. Грибняк С. С. Розробка платформи децентралізованого реєстру з покращеними характеристиками. *Інформатика та математичні методи в*

модельованні. 2023. Т. 13, №1-2, С. 48–55. DOI: 10.15276/imms.v13.no1-2.48. (Реєстр наукових фахових видань України, категорія «Б»).

Наукові праці, які засвідчують апробацію матеріалів дисертації:

6. Decentralized platforms: Goals, challenges, and solutions / S. Grybniak, Y. Leonchyk, R. Masalskyi, I. Mazurok, O. Nashyvan, R. Shanin. *2022 IEEE 7th Forum on Research and Technologies for Society and Industry Innovation (RTSI)*. 24-26 August 2022. Paris, France. 2022. P. 62–67. DOI: <https://doi.org/10.1109/RTSI55261.2022.9905225>. Decentralized platforms express.pdf. (Indexed in SCOPUS)

7. Waterfall: A Scalable Distributed Ledger Technology / S. Grybniak, D. Dmytryshyn, Y. Leonchyk, I. Mazurok, O. Nashyvan, R. Shanin. In *2022 IEEE 1st Global Emerging Technology Blockchain Forum: Blockchain & Beyond (iGETblockchain)*. November 2022. Irvine, CA, USA. 2022. P. 1–6. DOI: <https://doi.org/10.1109/iGETblockchain56591.2022.10087112>. Scalable Distributed Ledger express.pdf. (Indexed in SCOPUS)

8. Subnetworks in BlockDAG / O. Antonenko, S. Grybniak, D. Guzey, O. Nashyvan and R. Shanin. *2022 IEEE 1st Global Emerging Technology Blockchain Forum: Blockchain & Beyond (iGETblockchain)*. November 2022. Irvine, CA, USA. 2022. P. 1–6. DOI: <https://doi.org/10.1109/iGETblockchain56591.2022.10087101>. Subnet works in BlockDAG (Final Version).pdf

9. Waterfall: Salto Collazo. Tokenomics / S. Grybniak, Y. Leonchyk, R. Masalskyi, I. Mazurok O. Nashyvan. *2022 IEEE International Conference on Blockchain, Smart Healthcare and Emerging Technologies (SmartBlock4Health)*. 24-25 October 2022. Bucharest, Romania. 2022. P. 1-6. DOI: <https://doi.org/10.1109/SmartBlock4Health56071.2022.10034521>. Waterfall-Salto-Collazo express.pdf. (Indexed in SCOPUS)

10. Recurring Payments on EVM-based Platforms / S. Grybniak, N. Goga, O. Nashyvan, R. Mihai, I. Mazurok, Y. Leonchyk, G. Datta, O. F. Ozkul, C. V. Marian. *2022 IEEE 1st Global Emerging Technology Blockchain Forum: Blockchain & Beyond (iGETblockchain)*. 07-11 November 2022. Irvine, CA, USA. 2022. P. 1–6.

DOI: <https://doi.org/10.1109/iGETblockchain56591.2022.10087077>. Recurring Payments express.pdf. (Indexed in SCOPUS)

11. Blockchain-Enabled Economic Transactions: Recurring Financial Accruals and Payments / R. Mihai, O.F. Ozkul, G. Datta, S. Grybniak, C.V. Marian. 2022 IEEE 1st Global Emerging Technology Blockchain Forum: *Blockchain and Beyond, (iGETblockchain)*. 07-11 November 2022. Irvine, CA, USA. 2022. DOI: 10.1109/iGETblockchain56591.2022.10087074. (Indexed in SCOPUS)

12. Waterfall: Gozalandia. Distributed Protocol with Fast Finality and Proven Safety and Liveness / S. Grybniak, Y. Leonchyk, I. Mazurok, O. Nashyvan, R. Shanin. *IET Blockchain*. 2023. P. 1–12. DOI: <https://doi.org/10.1049/blc2.12023>.

13. Simulation Modelling of the Consensus Based on the Gozalandia / A. Vorokhta, I. Mazurok, Y. Leonchyk, S. Grybniak, and Y. Strakhov. *Adaptive Learning Management Technologies*. Kyiv. 2022. P. 31–33.

14. An Incentive System for Decentralized DAG-based Platforms / S. Grybniak, Y. Leonchyk, R. Masalskyi, I. Mazurok, O. Nashyvan. *IEEE Blockchain, Tech Brief*. 2022. URL: <https://blockchain.ieee.org/images/files/pdf/techbriefs-2022-q4/an-incentive-system-for-decentralized-dag-based-platforms.pdf>.

Довідка

Додаток Б

про використання результатів дисертаційної роботи
Грибняка Сергія Сергійовича
«Моделі та методи підвищення ефективності обробки
транзакцій у розподілених реєстрах»
у науково-дослідницької діяльності Національного університету «Одеська
політехніка»

Довідка видана в тому, що у науково-дослідницькій діяльності **Національного університету «Одеська політехніка»** використані наступні наукові результати, отримані у дисертаційній роботі *Грибняка Сергія Сергійовича*:

двошарову модель розподіленого реєстру, що поєднує в собі основну мережу, побудовану за технологією BlockDAG, та координаційну мережу, що базується на традиційній блокчейн технології, що дозволило підвищити швидкість обробки транзакцій при суттєвому масштабуванні системи.

Дисертацію виконано згідно тематичних планів НДР Національного університету «Одеська політехніка» за період 2020 - 2023 рр.

Грибняк Сергій Сергійович був відповідальним виконавцем проекту «Фундаментальні та поточні дослідження та розробки для Проекту Водоспад, технології розподілених обчислень з відкритим вихідним кодом», який фінансувався організацією Digital Clever Solutions Inc., США (Грантовий договір від 23.07.2021 № 01/210721).

Ректор, д.т.н., професор



Г.О. Оборський
Г.О. Оборський

Керівник НДР,
д.т.н., професор

С.Г. Антошук

С.Г. Антошук

ДОВІДКА

про впровадження результатів дисертаційної роботи
Грибняка Сергія Сергійовича
«Моделі та методи підвищення ефективності обробки
транзакцій у розподілених реєстрах»
у навчальний процес Національного університету «Одеська політехніка»

Довідка видана у результаті того, що в програмах, навчально-методичних матеріалах та курсах лекцій з дисциплін, що вивчаються на кафедрі Інформаційних систем Інституту комп'ютерних систем магістрам спеціальності 122 «Комп'ютерні науки» – «Безпечні інформаційні системи та блокчейн-технології» та бакалаврами спеціальності 122 «Комп'ютерні науки» – «Технології захисту інформації».

В навчальному процесі використовуються такі наукові результати, отримані у дисертації *Грибняка Сергія Сергійовича*:

– дисципліна «Безпечні інформаційні системи та блокчейн-технології»: двошарову модель розподіленого реєстру, що поєднує в собі основну мережу, побудовану за технологією BlockDAG, та координаційну мережу – Тема «Технології децентралізованих реєстрів».

– дисципліна «Технології захисту інформації»: метод прискореного оптимістичного консенсусу та метод перемішування даних за допомогою бієктивного відображення – Тема «Основи блокчейну».

Використані результати дисертації *Грибняка Сергія Сергійовича* «Моделі та методи підвищення ефективності обробки транзакцій у розподілених реєстрах» свідчать про глибоке опрацювання наукової літератури за тематикою дисертації. Висновки та пропозиції відзначаються науковою новизною, можливістю реалізації для завдань проектування децентралізованих реєстрів, тому істотно поліпшують матеріал означених дисциплін і сприяють підвищенню якості підготовки фахівців в галузі комп'ютерних наук.

Ректор, д.т.н., професор

Голова методичної ради ІКС
к.т.н.,



Г.О. Оборський

Н.П. Волкова

4.05.2023



ОДЕСЬКА МІСЬКА РАДА
Комунальне підприємство
«ТЕПЛОПОСТАЧАННЯ міста ОДЕСИ»

вул. Балківська, 1-Б, м.Одеса, Україна. 65029, ЄДРПОУ 34674102
тел.: (048) 705-62-18, факс: (048) 705-62-48 email: office@teplo.od.ua; www.teplo.od.ua

№ 01-18/566 від 10.08.2023 року
на № від року

ДОВІДКА

про впровадження результатів дисертаційної роботи аспіранта
Національного університету «Одеська політехніка»

Грибняка Сергія Сергійовича

«Моделі та методи підвищення ефективності обробки транзакцій у
розподілених реєстрах»

Довідка видана в тому, що у діяльності Комунального підприємства «Теплопостачання міста Одеси» використані результати, що отримані у дисертаційній роботі аспірантом кафедри прикладної математики та інформаційних технологій Національного університету "Одеська політехніка" МОН України *Грибняком Сергієм Сергійовичем*, прийнято до впровадження рекомендації щодо підвищення швидкості обробки повідомлень у розподілених базах даних в умовах високої масштабованості і криптозахищеності.

Запропоновані *Грибняком Сергієм Сергійовичем* науково-технічні рішення дозволили підвищити ефективність розподілених баз даних при реалізації концепції систем розподілу теплопостачання по об'єктах критичної інфраструктури.

Очікуваний ефект має такі складові:

- скорочення часу на прийняття рішень;
- зменшення споживаного трафіку мережі;
- підвищення захищеності системи від зовнішнього втручання.

Директор



Ганна ПОЗДНЯКОВА