МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

КУЧКОВСЬКИЙ ВОЛОДИМИР ВОЛОДИМИРОВИЧ

На правах рукопису

УДК 004.652

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ БЛОКЧЕЙН ДЛЯ ОПРАЦЮВАННЯ ВЕЛИКИХ ДАНИХ

122 – комп’ютерні науки

Дисертація на здобуття наукового ступеня
доктора філософії

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

_____/В.В. Кучковський/

Науковий керівник

Шаховська Наталія Богданівна

доктор технічних наук, професор

Львів – 2022

АНОТАЦІЯ

Кучковський В.В. Інформаційна технологія блокчейн для опрацювання великих даних. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії за спеціальністю 122 “Комп’ютерні науки”. – Національний університет «Львівська політехніка», Львів, 2022.

Зміст анотації. Дисертація присвячена забезпеченню цілісності даних та скороченню часу доступу до даних на основі розроблення інформаційної технології блокчейн для Великих даних. Через свій специфічний характер запису даних блокчейн добре підходить для тих завдань, де потрібна висока достовірність і незмінність інформації. Наприклад, у сфері інформаційної безпеки. Технологія розподіленого реєстру забезпечує цілісність та достовірність даних, а завдяки відсутності єдиної точки відмови – стабільність роботи інформаційних систем. Блокчейн може вирішити проблему довіри до даних, і навіть надати можливість універсального обміну даними.

У першому розділі «Аналіз технології блокчейн для опрацювання великих даних» здійснено аналіз технології блокчейн, описані математичні методи, які лежать в основі блокчейну, проаналізовано та описано, що собою представляють великі дані. На підставі проведеного аналізу відомих рішень встановлено, що існуючі технології опрацювання даних, які використовують бази даних, не можуть бути використані у блокчейн через неможливість забезпечення прийнятного часу доступу до даних та їх опрацювання. Поставлена задача, яку необхідно розв’язати методом дослідження та практикою.

У другому розділі «Розроблення методів послідовного опрацювання даних» було здійснено аналіз методів хешування даних. Розроблено модель великих даних у блокчейн та методи додавання даних у блокчейн. Розроблено

метод оцінки якості запису у блокчейн. Проаналізовано алгоритми консенсуса, необхідних для роботи з блокчейном.

У третьому розділі «Розроблення алгоритмів опрацювання великих даних у блокчейн» було розроблено алгоритм визначення блоків-сиріт, алгоритм кодування-декодування даних у тразакції, алгоритм запису даних у блокчейн, а також метод перевірки якості внесених даних. Розроблено алгоритм кодування та декодування даних у транзакції для наступного запису їх у блокчейн, розроблений базовий алгоритм для перевірки якості внесених даних. Усі алгоритми було зображено за допомогою блок-схем.

У четвертому розділі «Розроблення архітектури та апробація результатів» представлено розробку архітектури системи, обґрунтовано вибір технологій для реалізації системи. Також визначено функціонал системи та засоби, які забезпечать стабільну роботу системи. Розроблено протокол обміну даними. Здійснено аналіз отриманих результатів.

Основні наукові результати дисертації опубліковано в 10 працях, зокрема: 2 статті – у виданнях, що індексуються в наукометричних базах даних (одна стаття в журналі з Q2), 4 статті – у фахових виданнях України, 4 публікації – у збірниках наукових праць конференцій.

Ключові слова: блокчейн, хеш-функція, база даних, блок-сирота, цілісність даних.

ABSTRACT

Kuchkovskyy V. Blockchain information technology for big data processing. – Qualifying scientific work on the rights of the manuscript.

Dissertation for the degree of Doctor of Philosophy in the specialty 122 "Computer Science". – National University «Lviv Polytechnic», Lviv, 2022.

Abstract content. The dissertation is dedicated to ensuring data integrity and reducing data access time based on the development of blockchain information technology for Big Data. Due to its specific nature of data recording, blockchain is well suited for tasks that require high reliability and immutability of information. For example, in the field of information security. Distributed registry technology ensures the integrity and reliability of data, and due to the absence of a single point of failure, the stability of information systems. Blockchain can solve the problem of data trust, and even enable universal data exchange.

In the first chapter, "Analysis of blockchain technology for processing big data", the analysis of blockchain technology is carried out, the mathematical methods underlying the blockchain are described, and what big data is analyzed and described. On the basis of the analysis of known solutions, it was established that existing data processing technologies using databases cannot be used in the blockchain due to the impossibility of ensuring an acceptable time for data access and processing. A task has been set, which must be solved by the method of research and practice.

In the second chapter, "Development of methods for sequential processing of data", an analysis of data hashing methods was carried out. A model of big data in the blockchain and methods of adding data to the blockchain have been developed. A method for assessing the quality of recording in the blockchain has been developed. Consensus algorithms necessary for working with the blockchain were analyzed.

In the third section, "Development of algorithms for processing big data in the blockchain", an algorithm for determining orphan blocks, an algorithm for encoding and decoding data in a transaction, an algorithm for recording data in the blockchain, as well as a method for checking the quality of entered data were developed. An algorithm for encoding and decoding data in a transaction for subsequent recording in the blockchain has been developed, and a basic algorithm for checking the quality of entered data has been developed. All algorithms were depicted using flowcharts.

The fourth chapter "Development of the architecture and testing of the results" presents the development of the system architecture, substantiates the choice of technologies for the implementation of the system. The functionality of the system and means that will ensure stable operation of the system are also defined. A data exchange protocol has been developed. Analysis of the obtained results was carried out.

The main scientific results of the dissertation were published in 10 works, in particular: 2 articles – in publications indexed in scientometric databases (one article in a journal from Q2), 4 articles – in specialized publications of Ukraine, 4 publications – in collections of scientific works of conferences.

Keywords: blockchain, hash function, database, orphan block, data integrity.

СПИСОК ПУБЛІКАЦІЙ ЗДОБУВАЧА ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Статті в наукометричних базах даних:

1. Lytvyn, V., Vysotska, V., Kuchkovskiy, V., Bobyk, I., Malanchuk, O., Ryshkovets, Y., ... & Panasyuk, V. Development of the system to integrate and generate content considering the cryptocurrent needs of users // Eastern-European Journal of Enterprise Technologies. – 2019. – Vol. 1, No 2. – P. 18-39. (Q2)
2. Kuchkovskiy, V., Andrunyk, V., Krylyshyn, M., Chyrun, L., Vysotskyi, A., Chyrun, S., ... & Brodovska, I. Application of Online Marketing Methods and SEO Technologies for Web Resources Analysis within the Region // In Computational Linguistics and Intelligent Systems (COLINS 2021) 5th International Conference, Lviv, 22–23 April 2021, CEUR workshop proceedings. Aachen, CEUR-WS. org. – 2021. – Vol. 2870. – P. 1652-1693.

Статті у фахових виданнях України:

3. Кучковський В. В., Шаховська Н. Б. Блокчейн як база-даних, його використання, опис розумних контрактів і майбутній потенціал // Моделювання та інформаційні технології. – 2019. – Вип. 87. – С. 109–116.
4. Кучковський В. В. Алгоритми консенсуса блокчейн систем // Вісник Хмельницького національного університету. Серія: Технічні науки. – 2021. – № 3 (297). – С. 30–33.
5. Литвин В. В., Висоцька В. А., Кучковський В. В., Дуткевич С. Ю., Наум О. Метод інтеграції та управління контентом мережі інформаційних ресурсів туризму згідно з потребами користувача // Вісник Національного університету “Львівська політехніка”. Серія: Інформаційні системи та мережі. – 2018. – № 901. – С. 22–36.

6. Литвин В. В., Висоцька В. А., Кучковський В. В., Оливко Р. М. Архітектура інформаційної системи інтеграції та формування контенту про криптовалюти на основі аналізу діяльності бірж // Вісник Національного університету “Львівська політехніка”. Серія: Інформаційні системи та мережі. – 2018. – № 901. – С. 43–60.

Матеріали конференцій:

7. Lytvyn, V., Kuchkovskiy, V., Vysotska, V., Markiv, O., & Pabyrivskyy, V. Architecture of system for content integration and formation based on cryptographic consumer needs // IEEE 13th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT), Lviv, 11-14 September 2018. – 2018. – Vol. 1. – P. 391-395. IEEE
8. Kuchkovskiy, V., & Shakhovska, N. Information technology of Blockchain: Database, smart contracts, architecture // IEEE 14th International Conference on Computer Sciences and Information Technologies (CSIT), Lviv, 17-20 September 2019. – 2019. – Vol. 2. – P. 55-59. IEEE
9. Кучковський В. В. Алгоритми консенсусу // Trends in science and practice of today : abstracts of XXVIII International scientific and practical conference (Ankara, Turkey; June 01 – 04, 2021). – 2021. – С. 502–505.
10. Кучковський В. В. Змішані алгоритми консенсусу // Trends in science and practice of today : abstracts of XXVIII International scientific and practical conference (Ankara, Turkey; June 01 – 04, 2021). – 2021. – С. 506–507.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

АС – автоматизована система.

ПІР – клієнт, що бере участь у роздачі.

ІС – інформаційна система.

ПЗ – програмне забезпечення.

ПП – програмний продукт.

БД – база даних.

ASIC – інтегральна схема для специфічного застосування.

FPGA – програмована користувачем вентильна матриця.

GPU – графічний процесор.

CPU – центральний процесор.

RAM – оперативна пам'ять.

SEED – багатослівний термін.

P2P – пірінгова мережа.

Роздача – процес поширення файлу за протоколом BitTorrent.

Хеш – функція, що перетворює вхідні дані будь-якого розміру в дані фіксованого розміру.

Вступ

Актуальність теми. Великі дані (Big Data) і технології розподіленого реєстру (блокчейн, blockchain) є одними з найпопулярніших IT-тем. Можливості їх впровадження у кожну прикладну сферу, від банківської галузі до медицини, обговорюються на конференціях усіх рівнів.

У великих даних через різнорідність джерел даних виникає потреба в скороченні часу доступу до даних, що можна вирішити через блокчейн.

Насамперед слід підкреслити важливу відмінність технологій розподіленого реєстру та великих даних: Big Data передбачає інтеграцію інформації з різних джерел, тоді як у блокчейні, навпаки, копії інформаційних ланцюжків зберігаються на безлічі різних комп'ютерів. Децентралізоване зберігання та послідовний характер запису даних у блокчейн обумовлює досить низьку швидкість їх зчитування. Концепція Big Data має на увазі швидку обробку величезних масивів інформації, яку блокчейн може забезпечити.

Незважаючи на вагомі відмінності у технологіях Великих даних, їх поєднання може дати суттєві переваги, зокрема, скорочення часу доступу до даних, незважаючи на тип джерела, а також підвищення якості даних через зниження дублікатів та суперечностей у записах даних.

Дані, які потрапили до блокчейну, залишаються там назавжди. Тому застосовувати цю технологію доцільно лише для тих задач, де необхідне постійне зберігання незмінної інформації, а також навіть застарілої і тієї, що вже не використовується. Незмінність і достовірність інформаційних ланцюжків блокчейна стане в нагоді при організації автоматичного архіву операцій з даними, зокрема, для запису відомостей в залежності від різних галузей, де можна його використовувати. Аналогічно blockchain дозволяє отримувати докладні аналітичні дані про ланцюги поставок та споживання, щоб

відстежувати та контролювати продукцію при транспортуванні, наприклад, втрати ваги внаслідок усихання та випаровування деяких видів товарів.

Подібним чином поєднання Big Data і блокчейну можна використовувати в охороні здоров'я, щоб важливі дані про здоров'я клієнтів медустанов були максимально захищені, незмінні, перевірені і неможливі до будь-яких маніпуляцій. За допомогою блокчейну медичні установи зможуть обмінюватися достовірними відомостями зі роботодавцями, органами правосуддя, страховими компаніями, науковими установами та іншими організаціями, які потребують медичної інформації. Для таких галузей, як медицина та фармацевтика, є необхідність збереження усієї історії зміни параметрів об’єкта – пацієнтів, медикаментів тощо.

Отже, задача розроблення інформаційної технології блокчейн для опрацювання Великих даних є актуальною.

Зв'язок роботи з науковими програмами, планами і темами. Дисертація виконувалася відповідно до пріоритетних напрямків науково-дослідних робіт Національного університету “Львівська політехніка”, відповідно до координаційних планів Міністерства освіти і науки України. Зокрема, в рамках наукових досліджень, виконуваних відповідно до держбюджетної роботи кафедри систем штучного інтелекту «Інформаційна технологія формування психофізичного портрету в умовах стресових ситуацій» (№ держ. реєстру 0119U002257).

Метою дослідження є підвищення якості даних та скорочення часу доступу до даних на основі розроблення інформаційної технології блокчейн для Великих даних. Для досягнення поставленої мети в роботі потрібно розв’язати такі задачі:

1. провести аналіз сучасного стану інформаційних технологій в галузі Великих даних та застосування блокчейну;

2. розробити модель Великих даних у блокчейні;
3. розробити метод запису Великих даних у блокчейн;
4. розробити методи опрацювання Великих даних у блокчейні для зменшення часу доступу до даних;
5. удосконалити метод визначення блоків-сиріт для підвищення якості даних;
6. розробити алгоритми та програмні засоби апробації запропонованих методів та моделі.

Об'єкт дослідження – процеси опрацювання Великих даних у блокчейні.

Предмет дослідження – методи, моделі та інформаційні технології для опрацювання Великих даних у блокчейні.

Методи дослідження. У процесі розробки моделі Великих даних у блокчейн використано математичну статистику, теорію множин. Для побудови алгоритмів запису у блокчейн використано теорію інформації, хеш-функції, теорію адгоритмів. Для побудови методу перевірки якості внесених даних використано теорію кодування.

Наукова новизна одержаних результатів полягає в тому, що:

– **вперше** розроблено інформаційну модель Великих даних у блокчейні, де різнотипові записи з джерел Великі дані подані як послідовності заголовків, метаданих та даних, а також адреси зв’язків, що дозволяє зберігати дані різної структури та зменшувати супепречливість даних через використання додаткової хеш-функції, підтримуючи цим цілісність даних;

– **уперше** розроблено метод перевірки якості внесених даних, який полягає в пошуку унікальних контрольних сум та обчисленні хешів блоків, а також будується на основі інформаційної моделі Великих даних у блокчейн, що дає змогу забезпечити безутратне внесення даних та гарантує їх цілісність;

– **отримав подальший розвиток** метод визначення блоків-сиріт на основі марковських ланцюгів, шляхом додавання таких блоків як нової одиниці в ланцюжку блоків та врахування часу затримки у прийнятті блоків вузлами. Це дає змогу забезпечити цілісність даних та усунути надмірність у них;

– **удосконалено** метод запису даних у блокчейн шляхом включення додаткових параметрів у заголовок блоку, підпису попереднього блоку та перевірки наявності блоку-предка. Це дає змогу гарантувати, що попередній блок не був змінений.

Практичне значення одержаних результатів. Практична цінність роботи полягає у доведенні отриманих наукових результатів до конкретних технологій, методик, алгоритмів та програмних продуктів. Розроблена модель Великих даних у блокчейн та її реалізація дає змогу зберігати дані такого розміру, що певні реляційні бази даних, наприклад MySQL, навіть на потужних серверах, не змогли б виконати операції CRUD (Create, Update, Delete) в час, зазначений у вимогах якостей Кодда (12 с).

Розроблено алгоритм кодування та декодування даних у транзакції, який базується на перетворенні будь-яких даних з різними структурами в hex значення за запропонованою функцією strToHex для кодування та функції hexToStr для декодування.

На основі методу запису в блокчейн розроблено алгоритм запису транзакцій у блокчейн. Таким чином, маючи транзакцію, можна записати в неї будь-які дані, та записати їх назавжди в блокчейн. Експериментально визначено, що навантаженість на сервер знижується утричі у випадку збереження даних у блокчейні порівняно з реляційною базою даних.

Розроблено алгоритм перевірки якості внесених даних, який базується на аналізі наявності дублікатів, суперечливих даних та даних з пропусками. Це

дало змогу зменшити кількість запитів на анонсер майже вдвічі. Також це дає змогу усунути проблему з keep-timeout та знизити можливість втрати даних.

Розроблено архітектуру Великих даних у блокчейн для медицини, яка використана в інформаційній системі перевірки ліків, упровадженій в лікарні швидкої допомоги м. Львова. Розроблену систему можна використовувати, як конструктор для оптимізації даних, які є незмінними, наприклад логи для статистики тощо.

Особистий внесок здобувача. Основні положення та результати дисертаційної роботи одержані автором самостійно. Особисто здобувачеві належать наступні наукові результати: розроблено метод запису даних у блокчейн для криптовалют [1], розроблено метод перевірки даних, записаних у блокчейн [2], розроблено модель Великих даних у блокчейн [3], розроблено метод інтеграції даних [5, 6], розроблено архітектуру даних системи [7, 8].

Апробація результатів дисертації. У результаті дисертації розроблена і запроваджена ідея використання блокчейну для високонавантажувальних проектів з великими даними. Ця робота представляє собою дослідження з оптимізації високонавантажувальної системи під блокчейн технологію. Роботу апробовано на Міжнародній науково-практичній конференції «Комп'ютерні науки та інформаційні технології» (2018, 2019), Міжнародній науково-практичній конференції «Перспективи в науці і практиці сьогодення» (2021).

Публікації. За результатами виконаних досліджень опубліковано 10 наукових праць, із них 2 статті – у виданнях, що індексуються в наукометричних базах даних (одна стаття в журналі з Q2), 4 статті – у фахових виданнях України, 4 публікації – у збірниках наукових праць конференцій.

Структура і обсяг дисертації. Дисертаційна робота викладена на 156 сторінках та складається із змісту, переліку скорочень, вступу, чотирьох

основних розділів, в яких містяться 43 рисунків, списку використаних джерел із 101 найменування та 9 додатків.

РОЗДІЛ 1. АНАЛІЗ ТЕХНОЛОГІЇ БЛОКЧЕЙН ДЛЯ ОПРАЦЮВАННЯ ВЕЛИКИХ ДАНИХ

У розділі здійснено аналіз технології блокчейн, описані математичні методи, які лежать в основі блокчейну, проаналізовано та описано, що собою представляють великі дані.

1.1. Аналіз технології блокчейн

Blockchain – розподілена децентралізована мережа, яка забезпечує незмінність, конфіденційність, безпеку та прозорість. Не існує центрального органу для перевірки транзакцій, проте кожна транзакція в блокчейні вважається повністю захищеною та перевіреною.

Наше життя нерозривно пов'язане з грошима, даними і документами. Через це нам доводиться зв'язуватися з різноманітними посередниками, котрі видають нам ці гроші, документи і дані, перевіряють їх, засвідчують їхню справжність, видають копії, перевіряють достовірність копії і так далі.

Причому нас змушують довіряти цим посередникам, хоча раз у раз несумлінні банкіри ховаються з грошима клієнтів, нотаріуси заднім числом підробляють заповіти і договори, співробітники державних органів і комерційних організацій користуються службовим становищем у зловмисних цілях.

Технологія blockchain якраз і вирішує всі ці проблеми.

Blockchain – ланцюжок блоків даних, де кожен блок пов'язаний з попереднім. Блок містить у собі набір записів. А нові блоки завжди додаються строго в кінець ланцюжка [1][2][3][4][5].

Система побудована на трьох принципах:

- розподіленість;
- відкритість;

- захищеність.

Усі користувачі blockchain утворюють собою мережу серверів (вузлів), на кожному з яких зберігається копія даних blockchain. Зазвичай це повна копія всіх блоків, але, в принципі, можна зберігати лише потрібні на конкретному комп'ютері дані.

Завдяки цьому зупинити, виключити або зламати blockchain практично неможливо, оскільки для цього треба вимкнути або зламати всі вузли [6][7][8]. Поки є хоч один користувач, blockchain буде існувати. Кожен новий користувач розширює і зміцнює цю мережу. Причому всі вузли рівноправні, там немає організаторів, модераторів, контролерів і менеджерів. Кожен відповідає за себе сам.

Усі дані блоки в blockchain і їх вміст є завжди відкритими та для усіх доступними. Можна легко прочитати будь-який блок і побачити всі записи в ньому. Також можна подивитися повний ланцюжок і відстежити зміну інформації. Таким чином, усі дані в blockchain легко перевіряються, з цього випливає, що не обов'язково довіряти іншим учасникам мережі, адже самому завжди можна їх перевірити і отримати гарантовано достовірну відповідь.

Для захисту даних в blockchain використовуються різноманітні види шифрування. Завдяки цьому можна отримати здавалося б несумісне – відкритість і достовірність даних при повній недовірі до решти учасників, а також можливо, навіть при їх зловмисному умислі [9][10][11].

Надійність і захищеність тримається на криптографічних ключах, за допомогою яких можна перевіряти правдивість і коректність даних.

Криптографічні ключі розраховуються за допомогою односторонніх хеш-функцій. Маючи ключ, неможливо дізнатися про вхідні дані, а також неможливо використати інший набір даних, що дає такий самий ключ.

Тому при втраті ключа, втрачається і можливість використання цих даних. Це приведено в таблиці нижче, на даних стрічках було використано алгоритм хешування sha256.

Таблиця 1.1 – Вхідні дані і ключ.

Вхідні дані	Ключ
Blockchain	625da44e4eaf58d61cf048d168aa6f5e492dea166d8bb54ec06c30de07db57e1
Blockchain.	d70efa3d07ce636c2843270e3a67094e60889d5fbd6829a113b14716ee1c0c4f
Blockchain1	0fc6e34f6899f5e2ca06688e49bb42cc104a45d5bb86c55eafe8c7d588204a48
Blockchain)	afab0a78635c83245a27ae3e41286a72507202b5d790de37a89b433a899c3815

Усі дані blockchain зберігаються на вузлах користувачів blockchain-мережі. Усі користувачі мережі мають рівні права і обов’язки, узагалі кажучи, можуть робити все, що завгодно, в тому числі безуспішно намагатися обманути інших користувачів та намагатись змінити ланцюг блоків. Заборонити цього їм ніхто не може, тому що всі знаходяться в рівних умовах, мають рівні права, і в однаковій мірі можуть виконувати чи навіть порушувати свої обов'язки.

Таким чином, користувачеві blockchain-мережі не потрібні ніякі посередники, наприклад, банки, державні органи, аудитори, контролери, страхові компанії або реєстратори. Адже тут немає в кого просити дозволу, жодна зручно зайнята позиція не дає ніяких додаткових прав або можливостей, нічий авторитет не захищає вас більше, ніж ви самі себе можете захистити.

Увійшовши в blockchain-мережу, користувач підключається до інших комп'ютерів мережі для того, щоб обмінюватися з ними такими даними: блоками і транзакціями. Важливо, що ця мережа ніяк не прив'язана до географії, тобто користувач з Києва може одночасно підключитися до користувачів з Парижа, Лондона і Токіо. І це, до речі, також захищає його від будь-яких регіональних особливостей.

Отримавши нові дані, кожен користувач перевіряє їх коректність, і, переконавшись у достовірності, зберігає їх у себе, а також передає далі по мережі. Таким чином, у мережі може курсувати два види даних – підроблені та справжні – які поширюються, відповідно, зловмисними і добропорядними учасниками. Кожен з добропорядних учасників, виявивши підроблені дані, далі їх не передає. У результаті підроблені дані блукають тільки між зловмисними учасниками, а добропорядні учасники обмінюються лише коректними даними [12][13][14].

Учасники мережі діляться на дві групи: звичайні користувачі, які створюють транзакції з даними, і майнери, які видобувають блоки для наступного їх запису в блокчейн. Справа в тому, що створення блоку – це дуже ресурсомісткий і складний процес, ось тому далеко не всі можуть і хочуть цим займатися.

Звичайні користувачі створюють і поширюють по мережі записи, наприклад, “користувач з ключем А переводить 10 монет користувачеві з ключем Б” або “людина з ключем А відправила 100 монет з ключем Х”. Як ви бачите, всі записи відкриті, але зашифровані. Якщо ви знаєте ключ адреси відправника, то ви можете дізнатися входи і виходи, і чи знаходяться монети на даній адресі, але не дізнаєтеся ім'я відправника, якщо відправник сам не вирішив розмістити ці дані у відкритому вигляді. Причому в однієї людини може бути кілька ключів, тому, навіть знаючи ключ А, не можна дізнатися скільки всього монет.

Далі в системі майнери збирають записи, перевіряють їх і записують в блоки, а потім розсилають ці блоки по мережі. Після чого звичайні користувачі отримують блоки та зберігають їх у себе, щоб можна було коректно створювати свої і достовірно перевіряти чужі нові записи.

Поки новий запис не внесений у жоден блок, запис не вважається достовірним і висить в мемпулі. Будь-який учасник мережі може використовувати його тільки на свій страх і ризик, оскільки є ймовірність, що це некоректний запис або навіть підроблений, є ймовірність, що його можуть скасувати, тобто відкинути ланцюг з цим записом, а сам запис помітити як “сироту”. Тому зазвичай учасники просто пересилають новий запис, щоб він рано чи пізно дійшов до майнера, який включить його в блок. І лише тоді, коли запис збережений в блоці, можна бути впевненим, що запис перевірений, коректний і скасувати його вже неможливо. Ось так і функціонує blockchain.

Це досягається завдяки наявності консенсусного протоколу, який є основною частиною будь-якої мережі Blockchain.

Алгоритм консенсусу – це процедура, за допомогою якої всі однолітки мережі Blockchain досягають спільної згоди щодо поточного стану розподіленої книги [15][16][17]. Таким чином, консенсусні алгоритми досягають надійності в мережі Blockchain та встановлюють довіру між невідомими однолітками у розподіленому обчислювальному середовищі. По суті, консенсус-протокол є гарантом, що кожен новий блок з даними, який додається до блокчейну, є єдиною версією істини, яка є узгодженою усіма вузлами в блокчейні. Протокол Blockchain консенсусу складається з деяких конкретних цілей, а саме з таких як досягнення згоди, співпраця, рівні права для кожного вузла та обов’язкова участь кожного вузла в процесі консенсусу. Звідси виходить, що алгоритм консенсусу спрямований на пошук спільної згоди, яка є виграшем для всієї мережі.

Сам blockchain складається з блоків і найцікавіше те, як ті блоки генеруються при видобуванні (mining) [18][19][20]. Блок в blockchain складається з заголовка і тіла (рис. 1.1).

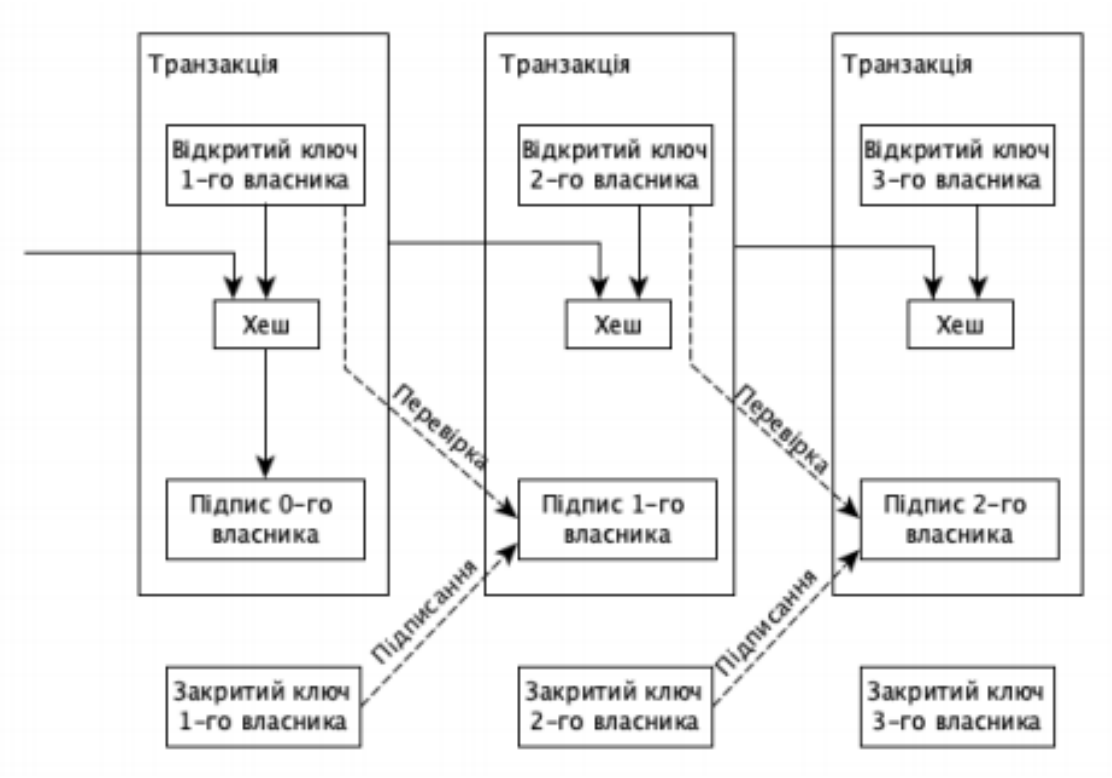


Рисунок 1.1 – Загальна схема структури ланцюга блоків

Тіло блоку – це просто список записів. Блоки в blockchain пов'язані за допомогою ключів, оскільки в заголовку кожного блоку зберігається ключ попереднього блоку. Це дуже важливе і разом з тим технічно грамотне рішення забезпечує захищеність blockchain.

Ключ кожного блоку розрахований на дані всього блоку і ключі попереднього блоку. А це означає, що в ключі будь-якого блоку закодовані не тільки записи цього блоку, але й всі попередні блоки. Усяка, навіть незнача зміна даних в будь-якому блоці викликає повну зміну його ключа, що, у свою чергу, вимагає зміни ключів усіх наступних блоків.

Таким чином, бачачи весь blockchain і ключі, можна легко перевірити коректність будь-яких даних, зокрема, впевнитися, чи вірна послідовність блоків, чи немає пропущеного блоку – або, може бути, у середину ланцюга вставлений новий блок, а також – чи відповідає ключ блоку збереженим у ньому даних. Простіше кажучи, завжди можна перевірити дані в блоці.

Ключ блоку повинен задовільняти важливі правила безпеки, що встановлює рівень захищеності мережі, який, до речі, може змінюватися у міру її зростання. Наприклад, у цифровій криптовалюті Bitcoin ключі перших блоків починалися з десяти нулів – і це встановлює ступінь складності створення нового блоку: як підробленого блоку, так і абсолютно коректного нового блоку [21][22][23].

Майнер – це такий же користувач blockchain-мережі, як і всі інші. Але, окрім перевірки і поширення даних, він ще займається виготовленням нових блоків.

Отримавши нові записи від інших учасників мережі, ці записи перебувають в мемпулі (memory pool), майнер збирає їх разом, формує заголовок майбутнього блоку і розраховує ключ блоку. Припустимо, після першого розрахунку ключ вийшов ось таким:

“5dfc885b7cea8e290d6a9f244f8883419f5c69df53df3d04d749a2cfa50d70e1”.

Однак, за правилами, ключ повинен починатися з десяти нулів. Щоб змінився ключ, необхідно змінити вихідні дані. Для цього в заголовку блоку передбачено спеціальне поле, яке називається nonce. При першому розрахунку воно дорівнює 0. Тому майнер змінює значення на 1 і знову розраховує ключ. Тепер він повністю змінився і дорівнює:

“176998421a40d91428d7433c3d9ede5648a08e961c5e31425830284da1028b0e”.

Тобто знову починається не з нулів. Тоді майнер збільшує nonce до 2 і перераховує ключ. Щоб знайти відповідне значення ключа, майнеру доводиться робити трильйони перерахунків. І коли відповідний ключ знайдений, майнер

зберігає блок і відправляє його іншим учасникам мережі. Тепер усі записи в блоці підтверджені і захищені ключем, який вельми нелегко підробити. Причому, як вищезазначено, у ключі блоку закодований і ключ попереднього блоку, який тепер підробити вже просто нереально.

Хитрість майнінга ключів полягає ще й у тому, що цей процес не має прогресу. Простіше кажучи, неважливо, коли ви почали шукати ключ, скільки записів входить у блок, скільки часу ви вже шукаєте, скільки ключів ви вже перебрали – ймовірність знаходження ключа на будь-якій ітерації завжди однакова. А це означає, що ви не можете зробити перерахунок, ні купити ключі, ні створити склад ключів – все це безглуздо, адже всі учасники рівноправні. І в кожного майнера є тільки одна можливість отримати відповідний ключ (рис. 1.2) [24][25][26].

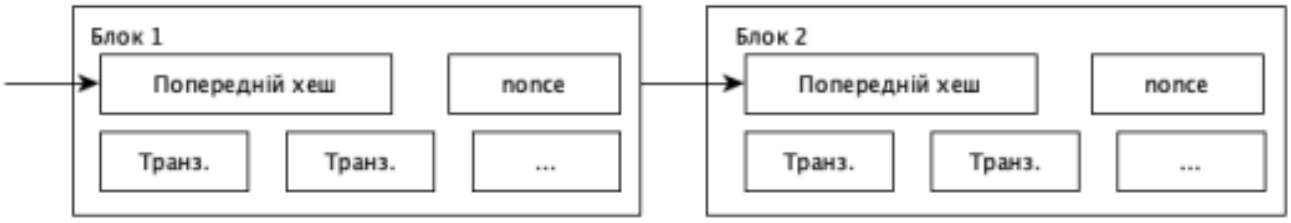


Рисунок 1.2 – Загальна схема структури ланцюга блоків

За кожний створений блок майнери отримують плату. Хто першим знайшов ключ, той створив блок і заробив, тому система виділяє нагороду. Решта не отримали нічого.

Таким чином, хитромудра процедура розрахунку ключів, звичайно, ускладнює створення блоку, але ще більше вона ускладнює створення підроблених блоків, роблячи це майже неможливим.

1.2. Поняття Великих даних

Великі дані – це різноманітні дані, які надходять з постійно зростаючою швидкістю і обсяг яких постійно зростає. Таким чином, три основні властивості великих даних – це різноманітність, висока швидкість надходження і великий обсяг [27][28][29].

Аналітика великих даних – це процес аналізу великих і складних джерел даних для виявлення тенденцій, моделей поведінки клієнтів і ринкових переваг, що допомагає приймати більш ефективні бізнес-рішення. Складність аналізу великих даних вимагає нових аналітичних інструментів, таких як аналітика прогнозів, машинне навчання, потокова аналітика, і такі методи, як аналіз в базі даних і в кластері.

Визначаються великі дані зазвичай чотирма V [30][31][32]:

- обсяг (Volume): великі обсяги даних;
- різноманітність (Variety): багато різних форм даних, неструктурованих і структурованих;
- швидкість (Velocity): частота вхідних даних;
- правдивість (Veracity): достовірність даних.

До вже перелічених факторів іноді ще додають:

- складність (Value): різний його рівень, наприклад, відомості про користувачів соцмереж та інформацію про транзакції в банківській системі;
- змінність (Variability): сплески і спади даних, які вимагають певних технологій для обробки.

Окрім величезного обсягу даних, складність зібраних даних створює проблеми в управлінні даними, їх інтеграції та аналізі. Але компанії, що об'єднують неструктуровані джерела даних, такі як вміст соціальних мереж, з

існуючими структурованими даними, такими як транзакції, можуть додавати контекст і генерувати нові, а часто і багатші ідеї [33][34][35].

Крім того, великі дані описують підвищену швидкість вхідних даних, що надходять з різноманітних джерел, таких як датчики, мобільні пристрої, веб-потоки кліків і транзакції, що призводить до необхідності аналітики в реальному часі. Організації, які можуть отримати вигоду від того, що відбувається зараз, щоб запобігти відмові обладнання, рекомендувати предмет для покупки, виявити шахрайство з кредитними картками і багато іншого, швидко стають лідерами в своїх галузях.

Нарешті, великі дані відносяться до високого ступеня точності даних, точності та достовірності. Це не означає, що всі дані повинні бути ретельно відібраними і чистими, тому що більш складні джерела даних, такі як соціальні мережі, можуть привести до нового розуміння з певним аналізом. Але важливо, щоб організації знали якість, точність та достовірність даних, що використовуються для формування розуміння та прийняття рішень.

В основі аналітики лежить перетворення даних у проникливі дії, які підвищують цінність для організації. Але зростання структурованих і неструктурованих даних, також відомих як великі дані, радикально змінило функцію аналітики.

Хоча великі дані розширили можливості, доступні для бізнесу, вони також створили більше проблем для збору, зберігання та доступу до інформації.

Цей революційний зсув висуває значні новітні вимоги до зберігання даних і ставить нові завдання для аналітичного програмного забезпечення. Це також створює потужні можливості для виявлення та реалізації сучасних стратегій для розвитку конкурентної переваги.

Реалізація цих можливостей вимагає двох речей: технологічного потенціалу для збору і зберігання великих даних, а також нових інструментів для перетворення даних у розуміння і, в кінцевому підсумку, у їх цінність.

Аналітика великих даних може об'єднувати дані в стані спокою (традиційні структуровані дані) з даними в русі (дані в реальному часі), щоб виявляти можливості й використовувати їх у даний момент [36][37][38].

Великі дані вже стали реальністю для більшості компаній, а обсяг і величезна складність великих даних можуть здатися приголомшливими. Компанії повинні працювати над тим, щоб осмислити і створити можливості як для даних у стані спокою, так і для даних у русі.

Великі дані являють собою розширений набір нових потужних можливостей. Хоча точно невідомо, як саме великі дані будуть використовуватися через рік, три або п'ять років, необхідність перетворити їх на конкурентну перевагу означає, що жодна компанія не може дозволити собі чекати щодо вжиття заходів.

Отримання вигоди від великих інвестицій у дані вимагає здатності ефективно використовувати їх. Пошук фрагментів інформації, які підвищують рентабельність інвестицій для організації, схожий на пошук голки в стосі сіна, тому багато компаній повідомляють про низький коефіцієнт окупності інвестицій у великі дані.

Щоб реалізувати потенціал епохи великих даних і мінімізувати її ризики, підприємствам потрібна уніфікована архітектура даних, а також програмне забезпечення для аналізу та візуалізації даних.

Переваги рішення для аналізу великих даних [39][40][41]:

- Платформи для аналізу великих даних, що дозволяє користувачам виявляти невидимі тенденції і закономірності у великих і складних наборах даних, які сприяють більш швидкій ідентифікації стратегічних можливостей і загроз.
- Єдиний погляд на бізнес – завдяки уніфікованій архітектурі даних, надає компаніям дуже узгоджене і всеосяжне вікно даних, що

підвищує ефективність прийняття рішень і дозволяє користувачам працювати з найточнішою і своєчасною інформацією.

- Найшвидший час для дій – аналітика великих даних підвищує продуктивність прийняття рішень, дозволяючи будь-якому співробітнику компанії передбачити ситуації і можливості, поставити актуальні і своєчасні запитання та отримувати відповіді, які ведуть до рішучих дій.

Розглянемо основні можливості аналітики великих даних:

- Розширені статистичні та машинного навчання розрахунки – це інструменти наявних наук про дані (Data of Science) та статистичні обчислення, які беруть великі обсяги історичних даних і використовують їх для отримання нових знань і пошуку закономірностей. Машинне навчання допомагає створювати і вивчати потужні алгоритми, які можуть поліпшити бізнес-процеси і підвищити цінність бізнесу.
- Потокова аналітика – за допомогою неї можна автоматизувати дії в режимі реального часу, застосовуючи аналітичні та прогнозовані моделі до оперативних даних. Використовуючи середовище візуальної розробки для швидкого створення та розгортання потокових додатків, можна дозволити операційним системам оцінювати дані, надсилати сповіщення та швидко виконувати дії для своєчасного прийняття рішень відповідно до контексту.
- Візуалізація даних – необхідна проста статистика і вбудовані коннектори даних, які дозволяють швидко імпортувати дані в інтуїтивно зрозумілі панелі моніторингу. Це дозволяє надати своїм бізнес-користувачам можливість аналізувати великі джерела даних, приймати рішення, засновані на реальних даних, і постійно

використовувати інформаційні панелі, що відповідають потребам бізнесу.

- Віртуалізація даних – ця функція забезпечує сучасний рівень даних, який дозволяє користувачам отримувати доступ, об'єднувати, перетворювати і доставляти набори даних з неймовірною швидкістю і економічністю [42][43][44]. Завдяки віртуалізації даних користувачі отримують не тільки актуальні, але і найостанніші дані, які легко знайти, використовувати і зрозуміти.
- Управління активами даних – дозволяє забезпечувати постійний доступ, доставку, управління та безпеку даних відповідно до вимог організації, використовуючи такі інструменти, як управління основними даними, віртуалізація даних, каталог даних, а також підготовка та обробка даних самообслуговування.
- Виявлення даних самообслуговування – це велике аналітичне рішення для даних дозволяє користувачам у всій організації досліджувати дані і отримувати відповіді без необхідності спеціалізованого, поглибленого моделювання даних. Це зменшує залежність від IT та виділених ресурсів бізнес-аналітики (BI) і значно прискорює процес прийняття рішень.

Загальні джерела даних для аналітики великих даних [45][46][47]:

- Платформи великих даних;
- Транзакційні дані;
- IoT дані датчиків;
- Соціальні медіа;
- Інтернет / Онлайн дані;
- Дані мобільних пристроїв;
- Історичні дані та дані в реальному часі;

- Торгова точка (PoS);
- Геопозиційні дані;
- Текстові дані.

Завдяки таким великим наборам даних можна: визначати повний спектр клієнта, забезпечувати захист від шахрайства, оптимізовувати ціни, визначати ефективність роботи, розробляти рекомендаційні сервіси, аналізувати соціальні мережі, прогнозувати погоду тощо.

Процес збору великих даних. Технології і сам процес збору даних називають дата майнінг (data mining). У процесі збору система може знаходити петабайти інформації, яка потім буде оброблена методами інтелектуального аналізу, що виявляє закономірності. До них відносять нейронні мережі, алгоритми кластеризації, алгоритми виявлення асоціативних зв'язків між подіями, дерева рішень, і деякі методи machine learning.

Коротко процес збору та обробки інформації виглядає так:

- аналітична програма отримує завдання;
- система збирає потрібну інформацію, одночасно готуючи її: видаляє нерелевантну, очищає від сміття, декодує;
- вибирається модель або алгоритм для аналізу;
- програма вчиться алгоритму і аналізує знайдені закономірності.

Найчастіше дані зберігають в data lake “озері даних”. При цьому зберігають у різних форматах і ступенях структурованості:

- рядки і колонки з БД-структурні;
- CSV, XML, JSON-файли, логи – напівструктуровані;
- документи, поштові повідомлення, неструктуровані pdf файли;
- аудіо, відео та бінарні зображення.

Для зберігання та обробки даної інформації в data lake використовують такі інструменти, як:

Hadoop – платформа управління даними, яка містить один або кілька кластерів. Дана платформа використовується для обробки, зберігання та аналізу великих обсягів нереляційних даних: файлів журналів (логів), записів інтернет-трафіку, даних IoT датчиків, об'єктів JSON формату, повідомлень у соцмережах, зображень тощо. Якщо простими словами, то це програма для кластеризації даних. Також це набір утиліт, бібліотек і фреймворків, які служать для розробки і виконання розподілених програм. Вони працюють на кластерах багатьох вузлів.

Storm – розроблений на Clojure фреймворк для обробки інформації в реальному часі.

Data lake – "Озеро даних" може включати в себе і програмну платформу, наприклад, Hadoop, і кластери серверів зберігання та обробки даних, і засоби інтеграції з джерелами й споживачами інформації та системи підготовки даних, управління й іноді інструментів машинного навчання. Також "озеро даних" можна масштабувати до тисячі серверів без зупинки кластера.

Коли дані отримані та збережені, їх потрібно проаналізувати і представити в зрозумілому вигляді: графіки, таблиці, зображення або готові алгоритми. Через обсяг і складність у обробці традиційні способи не підходять. З великими даними необхідно:

- обробляти весь масив даних;
- шукати кореляції по всьому масиву (в тому числі приховані);
- обробляти і аналізувати інформацію в реальному часі.

Тому для роботи з big data розроблені окремі технології, такі як [48][49][50].

MapReduce – фреймворк для паралельних обчислень дуже великих наборів даних (до декількох петабайт). Розроблений Google (2004 рік).

NoSQL – Not Only SQL. Допомагає працювати із розрізненими даними, розв’язує проблеми масштабованості і доступності за допомогою атомарності і узгодженості даних.

1.3. Математичні методи, які лежать в основі блокчейн

Математичні методи, які лежать в основі блокчейна, – це еліптичні криві, ECDSA і ключі [51][52][53].

Фундаментальною частиною блокчейна є алгоритми криптографії. Один з таких алгоритмів – це ECDSA – Elliptic Curve Digital Signature Algorithm, що використовує еліптичні криві (elliptic curve) і кінцеві поля (finite field) для підпису даних, і розроблений для того, щоб третя сторона мала можливість підтвердити автентичність підпису, виключивши можливість її підробки. Для підпису ECDSA і верифікації використовуються різні процедури, що складаються з декількох арифметичних операцій.

Нехай маємо деяке поле K, еліптична крива над цим полем – це кубічна крива над алгебраїчним замиканням поля K, що задається рівнянням третього степеня з коефіцієнтами з поля K і «точкою на нескінченності». Одними з таких форм еліптичних кривих є криві Вейерштрасса [54][55][56].

Алгебраїчно кожна така крива може бути представлена як рівняння виду: $y^2 = x^3 + ax + b$. У мережі криптовалюти Bitcoin, коефіцієнти формули еліптичної кривої: $a = 0$ і $b = 7$, графік функції приймає наступний вигляд:

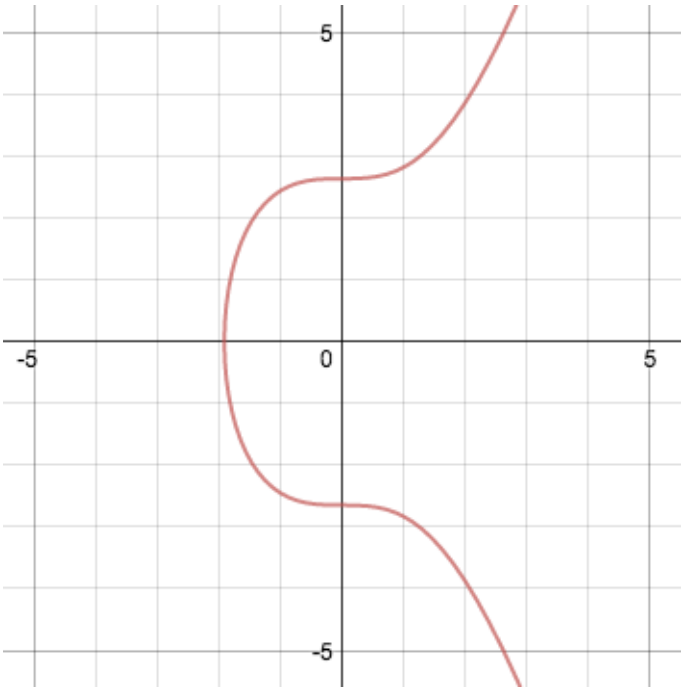


Рисунок 1.3 – Еліптична крива в Bitcoin

Еліптичні криві мають декілька цікавих властивостей, наприклад, невертикальна лінія, яка перетинає дві недотичні точки на кривій, перетинає третю точку на кривій. Сумою двох точок на даній кривій P + Q називають точку R, яка є відображенням точки -R (побудованої шляхом продовження прямої (P, Q) до перетину з кривою) щодо осі X [57][58][59].

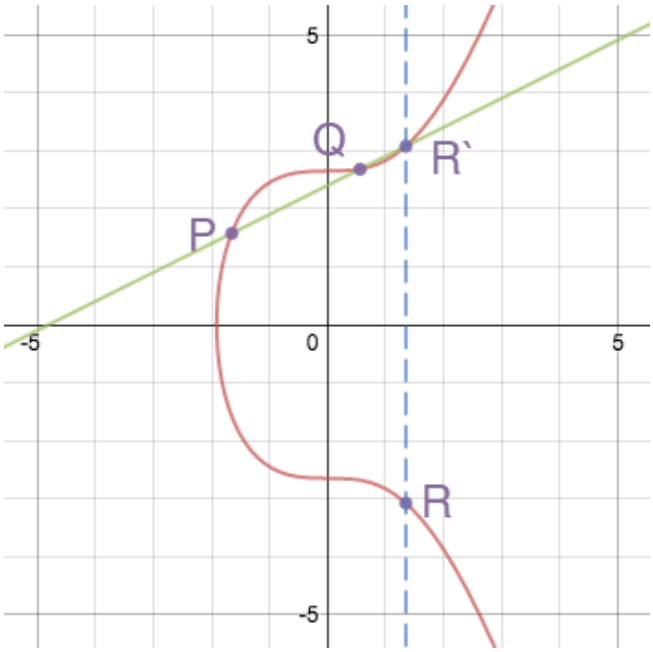


Рисунок 1.4 – Еліптична крива в Bitcoin

Якщо, для прикладу, провести пряму через дві точки, що мають координати виду P (a, b) і Q (a, -b), то дана пряма буде паралельна осі ординат. У цьому випадку не буде третьої точки перетину, для того щоб розв'язати цю проблему, вводиться так звана точка на нескінченності (point of infinity), що позначається як O. Тому, якщо перетин відсутній, рівняння приймає такий вигляд: P + Q = O.

Якщо ми хочемо накласти точку саму на себе (подвоїти її), то в цьому випадку просто проводиться дотична до точки Q. Отримана точка перетину відображається симетрично щодо осі X.

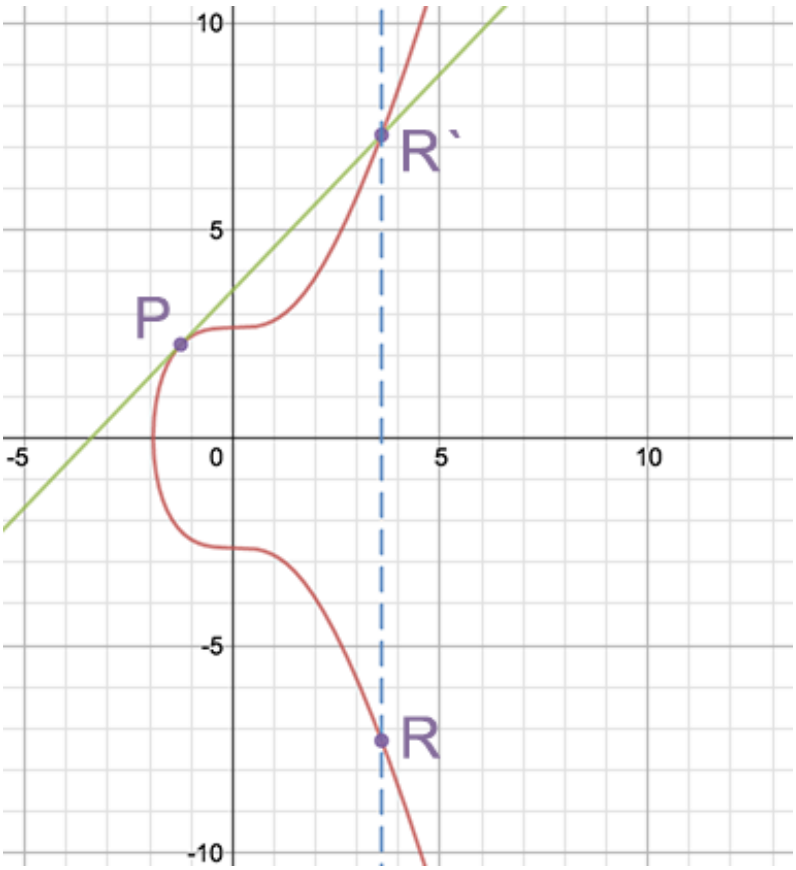


Рисунок 1.5 – Еліптична крива в Bitcoin

Разом ці дві операції використовуються для скалярного множення, R = aP, визначеного додаванням точки P самої до себе. Наприклад:

R = 7P

R = P + (P + (P + (P + (P + (P + P)))))

Процес скалярного множення зазвичай можна спростити за допомогою комбінації операцій додавання і подвоєння точок. Для прикладу:

R = 7P

R = P + 6P

R = P + 2 (3P)

R = P + 2 (P + 2P)

Тут 7P було розбито на два етапи подвоєння точок і два етапи додавання точок.

Еліптична крива над кінцевим полем, в еліптичній криптографії використовується така ж крива, є розтягнута над деяким кінцевим полем. Кінцеве поле в контексті еліптичної криптографії можна представити як зумовлений набір додатніх чисел, у якому повинен отримуватися результат кожного обчислення.

$y^2 = x^3 + ax + b \pmod p$

Наприклад, 9 mod 7 = 2. Тут ми маємо кінцеве поле від 0 до 6, і всі операції по модулю 7, над яким би числом вони не здійснювалися, дадуть результат, що потрапляє в цей діапазон.

Усі названі вище властивості (додавання, множення, крапка в нескінченності) для такої функції залишаються дійсними, хоча графік цієї кривої не буде схожим на еліптичну криву. Еліптична крива монети біткойна, $y^2 = x^3 + 7$, визначена на кінцевому полі по модулю 67, виглядає наступним чином:

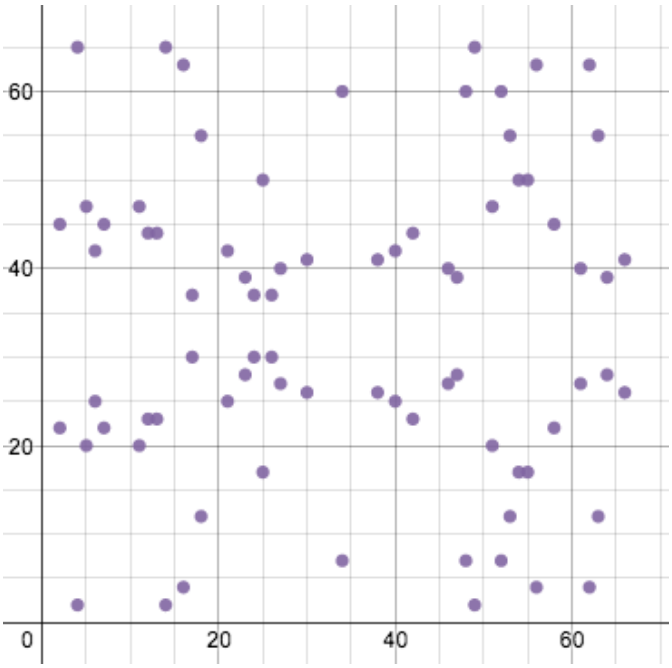


Рисунок 1.6 – Еліптична крива в Bitcoin по модулю 67

Це множина точок, у яких всі значення x і y є цілими числами між 0 і 66. Прямі лінії, намальовані на цьому графіку, тепер будуть ніби "обертатися" навколо поля, як тільки досягнуть бар'єру 67, і продовжаться з іншого його кінця, зберігаючи колишній нахил, але зі зрушенням. Наприклад, додавання точок (2, 22) і (6, 25) у цьому конкретному випадку виглядає так:

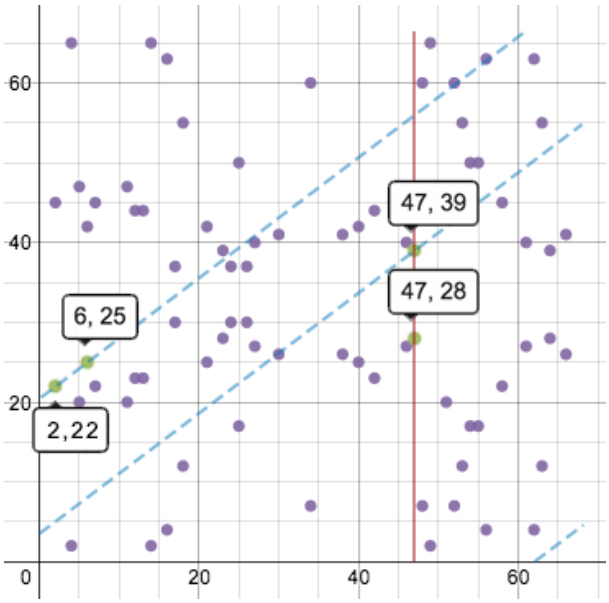


Рисунок 1.7 – Еліптична крива в Bitcoin по модулю 67

Такий протокол, як біткойн, вибирає набір параметрів для еліптичної кривої і її представлення в кінцевому полі, який фіксований для всіх користувачів протоколу. Параметри включають використовуване рівняння, просте значення по модулю поля і базову точку, яка потрапляє на криву. Порядок базової точки, який не вибирається довільно, а є функцією інших параметрів, можна представити графічно як кількість разів, коли точка може бути додана до самої себе, поки її нахил не стане нескінченним або не буде вертикальною лінією. Базова точка вибирається таким чином, щоб порядок був великим простим числом.

Біткойн використовує дуже великі числа для своєї базової точки, простого числа по модулю і порядку. Насправді всі практичні програми ECDSA використовують величезні значення. Безпека алгоритму залежить від того, що ці значення великі і, отже, непрактичні для методу грубої сили або зворотного проєктування.

Візьмемо для прикладу Bitcoin, рівняння еліптичної кривої: $y^2 = x^3 + 7$.

Простий модуль буде: $2^{256} – 2^{32} – 2^9 – 2^8 – 2^7 – 2^6 – 2^4 – 1$ = FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFC2F

Отже, базова точка буде: 04 **79BE667E F9DCBBAC 55A06295 CE870B07 029BFCDB 2DCE28D9 59F2815B 16F81798** 483ADA77 26A3C465 5DA4FBFC 0E1108A8 FD17B448 A6855419 9C47D08F FB10D4B8

Жирним шрифтом виділена координата X в шістнадцятковому записі. За нею відразу слідує координата Y.

Ось порядок, який з цього виходить: FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE BAAEDCE6 AF48A03B BFD25E8C D0364141

Ця конкретна реалізація називається secp256k1, яка є і частиною сімейства рішень еліптичних кривих над кінцевими полями, цей набір

параметрів є частиною сімейства стандартів SEC (Standards for Efficient Cryptography), запропонованих для використання в криптографії. У біткоїна еліптична крива secp256k1 використовується спільно з алгоритмом цифрового підпису ECDSA. У ECDSA секретний ключ складається з випадкового числа між одиницею і значенням порядку. Відкритий ключ формується на основі секретного ключа: останній множиться на значення базової точки і виходить відкритий ключ, у якому будуть міститись транзакції блокчейна. Рівняння має наступний вигляд:

*Відкритий ключ = секретний ключ * G.*

Це демонструє, що максимальна кількість секретних ключів (отже, біткоїн-адрес) – звичайна, і дорівнює порядку. Однак порядок є неймовірно великим числом, а саме 2^256 степені, тому випадково або навмисно підібрати секретний ключ іншого користувача зі сучасними технологіями нереально.

У безперервному полі ми можемо нанести дотичну лінію і визначити відкритий ключ на графіку, але є деякі рівняння, які виконують те саме в контексті скінчених полів. Точкове додавання p + q для знаходження r визначається по компонентах наступним чином:

$c = (q_y - p_y) / (q_x - p_x)$

$r_x = c^2 - p_x - q_x$

$r_y = c (p_x - r_x) - p_y$

Скалярний добуток p, щоб знайти r, виглядає так:

$c = (3p_x^2 + a) / 2p_y$

$r_x = c^2 - 2p_x$

$r_y = c (p_x - r_x) - p_y$

На практиці обчислення відкритого ключа розбивається на ряд операцій добутку і складання точок, починаючи з базової точки.

Розглянемо приклад із оберненої сторони, використовуючи невеликі числа, щоб отримати уявлення про те, як створюються і використовуються ключі при підписанні і перевірці. Параметри, які ми будемо використовувати, такі:

Рівняння: $y^2 = x^3 + 7$ (тобто a = 0 і b = 7)

Число по модулю: 67

Базова точка: (2, 22)

Порядок: 79

Закритий ключ: 2

У першу чергу знайдемо відкритий ключ. Оскільки ми вибрали найпростіший з можливих закритих ключів зі значенням рівним 2, для цього буде потрібна тільки операція подвоєння однієї точки з базової точки. Розрахунок подаємо наступним чином:

c = (3 * 22 + 0) / (2 * 22) мод 67

c = (3 * 4) / (44) мод 67

c = 12/44 mod 67

Виходить, що відкритий ключ: 0.27272727272.

Але, результат має бути цілим числом. Тому потрібно помножити на обернену величину, яку простір не дозволяє нам визначити тут. У цьому випадку доведеться довіритися нам на даний момент, що:

$44^{-1} = 32$

Обчислюємо:

c = 12 * 32 mod 67

c = 49

$r_x = (49^2 – 2 * 2) \bmod 67$

$r_x = 52$

$r_y = (49 * (2 – 52) – 22) \bmod 67$

$r_y = 7$

Таким чином виходить, що відкритий ключ відповідає пункту (52, 7) в координатах. Усе це працює для закритого ключа 2!

Обчислення відкритого ключа виконується за допомогою таких самих операцій подвоєння і складання точок. Це тривіальне завдання, яке звичайний персональний комп'ютер або смартфон може вирішувати за мілісекунди. Але обернену задачу (отримання секретного ключа з публічного) є проблемою дискретного логарифмування, яка вважається і є на даний момент обчислювально складною. Кращі відомі алгоритми її вирішення, на зразок ро Полларда, мають експоненціальну складність. Для secp256k1, щоб вирішити цю задачу, потрібно близько зробити близько 2^256 степені операцій, що потребує часу обчислення на звичайному комп'ютері у порівнянні з часом існування Всесвіту. Тому перехід від закритого ключа до відкритого ключа за задумом є подорожжю в один кінець.

Як і у випадку зі закритим ключем, відкритий ключ зазвичай представлений шістнадцятковим рядком. У нестисненому відкритому ключі два 256-розрядних числа, що представляють координати x і y, просто склеєні в один довгий рядок. Можна скористатися симетрією еліптичної кривої для створення стисненого відкритого ключа, зберігаючи тільки значення x і визначаючи, на якій половині кривої знаходиться точка. З цієї часткової інформації можна відновити обидві координати.

1.4. Постановка задачі

Суть дисертаційного дослідження полягає в розробленні інформаційної системи з використанням технології блокчейн для опрацювання Великих даних.

На підставі проведеного аналізу відомих рішень встановлено, що існуючі технології опрацювання даних, що використовують бази даних, не можуть бути використані в блокчейн через неможливість забезпечення прийнятного часу доступу до даних та їх опрацювання (Таблиця 1.2, Таблиця 1.3).

Таблиця 1.2. Порівняльний аналіз сучасного стану застосування блокчейн

Застосування	**Призначення**	**Переваги**	**Недоліки**
Вибори	Інформація про голосування людей	Накопичення даних, менша можливість фальсифікації даних, доступні для відкритого доступу	Людський фактор є недоліком даної системи, фальсифікації можливі через збої системи
Логістика	Запис логістичних маршрутів, відслідковування посилок	Заміна стандартних баз даних, побудова маршрутів на основі блокчейну	При виникненні помилки, дані залишаться назавжди помилковими.
Транзакційні дані	Логування історії пересилання коштів	Історія транзакцій від початку до останнього тимчасового власника коштів. Пошук злочинців за допомогою визначення всього ланцюга	Не повна анонімність в передачі транзакцій, при визначенні ланцюга можна встановити всіх користувачів, котрі передавали кошти
Медицина	Запису захворювань пацієнтів	Спрощує процес підтвердження факту	Людський фактор, через нього можуть бути помилки в історіях захворювань
IoT дані датчиків	Запис статичних даних з датчиків	Висока доступність та швидкий запис даних з мінімальними втратами	При величезній кількості даних, можливі проблеми з масштабуванням даних, через ріст в арифметичній прогресії

Геопозиційні дані	Запис геопозиції з телефонів	Висока доступність, швидкий запис та вибірка даних, незмінність	При неправильній передачі даних або збоях, назавжди буде записана невірна інформація
Нерухомість	Записи всіх власників нерухомості	Відкритий реєстр, можна дізнатися про ланцюг всіх власників нерухомості	Порушення анонімності

Таблиця 1.3. Аналіз існуючих методів блокчейн

Алгоритм	**Призначення**	**Переваги**	**Недоліки**
Адреси з еліптичної кривої Діффі-Хелмана-Меркле (ECDHM)	Встановлює спільний секрет між двома сторонами	Може бути використаний для приватного обміну повідомленнями через загальнодоступну мережу. Відправник і приймач обмінюються один з одним адресами ECDHM, а потім використовують спільний секрет для отримання анонімних адрес в мережі блокчейну	Приховує лише ідентичність користувача, а не його загальну транзакційну діяльність, де ці адреси фігурують у якості відправника чи отримувача коштів

змішувачі	Група користувачів може об'єднати власні виплати в одне угрупування, відстежуючи свої позички в окремій інстанції – приватній книзі, що надається їм як інструмент для виплат	Будь-який користувач блокчейну, може побачити сплачені суми разом з одержувачами, але теоретично неможливо простежити особу, яка ініціювала платіж	Забезпечують лише незначний захист від Сібіл-атаки та атаки відмови в обслуговуванні. Ще більш критичним є той факт, що прихованою приватною книгою обліку змішувача повинен управляти деякий центральний орган, тобто треба довіряти третій стороні для "змішування" транзакцій.
Криптовалюта Monero	Використання схеми "кільцевого підпису" та генерування одноразових "невидимих адрес" для кожної транзакці	Використовуючи приватний ключ, підпис 16 групи підтверджує, що один підписувач з фіксованої групи схвалив транзакцію, не розкриваючи його ідентичності	Можливість побудови кільцевої атаки та аналізу простежуваності транзакційної діяльності користувачів

Тому необхідно розробити модель Великих даних у блокчейн та забезпечити розроблення методів додавання та пошуку даних із забезпеченням їх цілісності.

Також у дисертаційній роботі необхідно розробити систему, яка працює як конструктор, що дасть змогу інтегрувати рішення в різні галузі для опрацювання великих даних та забезпечити збереження даних в інформаційну технологію блокчейн. Великі дані, які були використані в дисертаційній роботі, як приклад роботи системи і її алгоритмів та методів, подано зі сторони інформаційних технологій, а саме для торрент-трекера.

1.5. Висновки

У даному розділі проведений аналіз технології блокчейн, подано поняття великих даних, описані методи, які лежать в основі блокчейн технології, поставлена задача, яку необхідно розв’язати методом дослідження та практикою.

Визначено слабкі сторони існуючих методів опрацювання даних у блокчейн, які б завадили зберіганню Великих даних у блокчейн.

Проаналізовано математичне підгрунтя блокчейн, що дало змогу здійснити постановку задачі.

На підставі проведеного аналізу відомих рішень встановлено, що існуючі технології опрацювання даних, що використовують бази даних, не можуть бути використані у блокчейн через неможливість забезпечення прийнятного часу доступу до даних та їх опрацювання.

Результати розділу опубліковано у таких наукових роботах [1, 2].

РОЗДІЛ 2. РОЗРОБЛЕННЯ ІНФОРМАЦІЙНОЇ МОДЕЛІ ВЕЛИКИХ ДАНИХ ТА ЇХ ПОСЛІДОВНОГО ДОДАВАННЯ

У розділі здійснено аналіз методів хешування даних для визначення проблем втрати якості даних. Розроблено модель великих даних у блокчейн та методи додавання даних у блокчейн. Розроблення методу перевірки якості внесених даних.

2.1. Аналіз методів хешування даних

Хешування створено для виконання багатьох задач. В основнову використовується хешування для порівняння даних, тобто якщо взяти різні дані, захешувати їх, то вихідний хеш завжди буде різним. Але іноді бувають випадки, коли різні дані дають однаковий хеш-код, це називається колізіями. Колізії відіграють велику роль у якості хеш-функцій.

Колізії є частими у Великих даних через те, що дані надходять з різних джерел, їх структури відрізняються, швидкість надходження даних надзвичайно велика.

Серед багатьох хеш-функцій можна виділити криптографічно стійкі, які використовуються в криптографії.

Визначити стійку криптографічну функцію можна через неможливість з вихідних даних отримати вхідні дані та через стійкість до колізій.

Методи хешування даних базуються на однобічних хеш-функціях. Деякі функції можливо генерувати тільки використовуючи CPU, деякі через специфічний алгоритм працюють лише для GPU, також є варіанти, коли одна і та сама функція може працювати і на CPU, і на GPU. Існує також така річ ASIC – це інтегральна схема, яка застосовується для конкретної хеш-функції.

На даний момент є такі сучасні хеш-функції, які використовуються в блокчейн системах:

allium – це поєднання декількох алгоритмів хешування, таких як Lyra2 в якості основного компонента: BLAKE-256, SHA-3/Keccak, Lyra2 з використанням сітки 8 на 8, CubeHash-256, повторення Lyra2, Skein-256 і, нарешті, Grøstl-256. Даний мікс використовується в блокчейнах: Garlicoin, Tuxcoin, Globaltoken (Allium).

argon2 – функція формування ключа. Використовує повідомлення від 0 до $2^{32} – 1$. Також необхідно вказати сіль з розміром відповідно від 8 до $2^{32} – 1$. Даний алгоритм не використовується в блокчейнах, а використовуються його похідні модифікації. Сама хеш-функція працює ось так: починає хешувати повідомлення в алгоритмі Blake2b, результат записується в блоки пам’яті, потім блоки пам’яті перетворюються з використанням стискання, на виході отримуємо ключ, Tag.

У роботі *використано алгоритм хешування* на основі Argon2. Схема роботи розробленої хеш-функції подана на рис. 2.1.

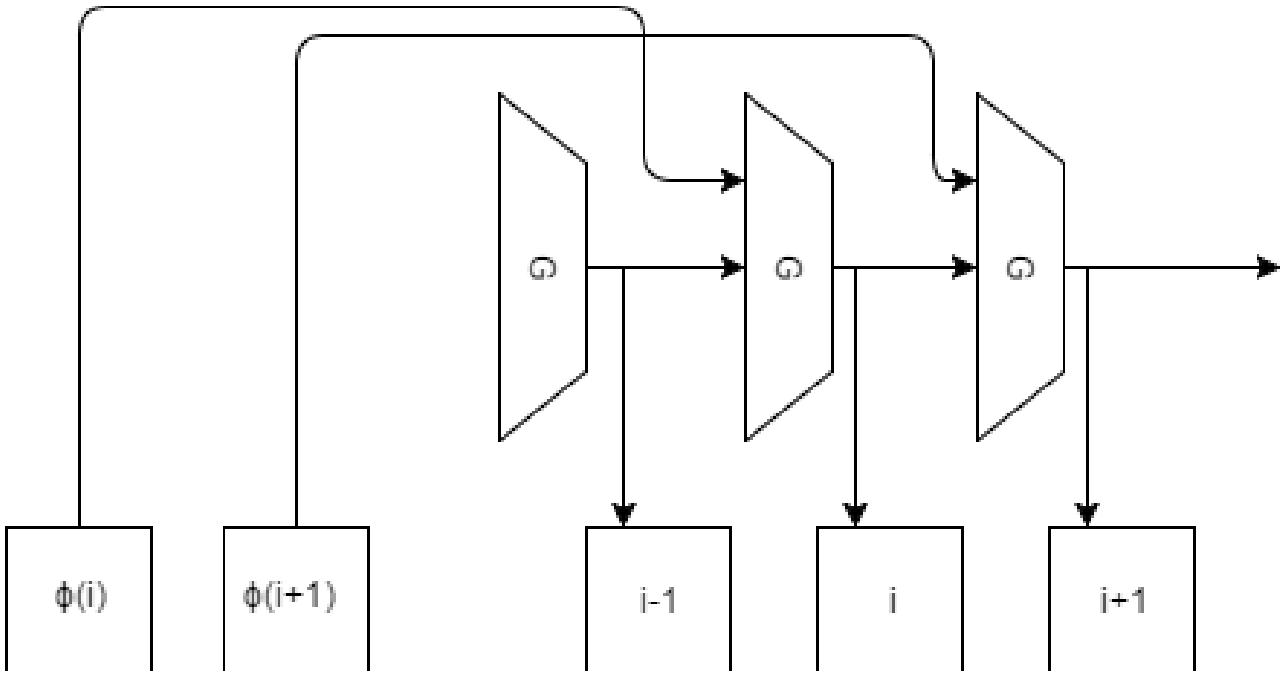


Рисунок 2.1 – Схема роботи хеш-функції Argon2

argon2d – оптимізований argon2 для захисту інформаційних систем та цифрових валют. Даний алгоритм не схильний до атак сторонніми каналами. Атаки діляться на 3 типи: контроль над обчислювальним процесом (активний і пасивний), способи доступу до системи (агресивні, неагресивні, напівагресивні), за методом аналізу (прості і різносторонні). Дана хеш-функція не використовується в блокчейнах.

argon2d250 – варіація argon2d, яка використовується в блокчейнах таких, як Credits, Zumy.

argon2d500 – варіація argon2d, яка використовується в блокчейнах таких, як Dynamic.

argon2d4096 – також варіація argon2d, який використовується в блокчейнах таких, як Argentum, Unitus.

Різниця між argon2d250, argon2d500 і argon2d4096 тільки в об’ємі використовуваної пам’яті, відповідно 250 = 256 KB, 500 = 512 KB, 4096 = 4 Mb.

argon2i – оптимізований argon2 для захисту від атак побічними каналами. Він звертається до масиву пам'яті незалежно від паролю порядку. У блокчейнах використовується в основному його модифікація argon2id, а найпопулярніший блокчейн, який використовує цю хеш-функцію: Arionum.

argon2id – модифікація argon2i для захисту від trade-off атак, який працює повільніше ніж argon2d через те, що дана хеш-функція робить декілька проходів в пам’яті. Приклад таких блокчейнів: Aquachain.

blake – дана функція побудована на трьох компонентах: режим ітерації HAIFA, яка забезпечує стійкість до атак другого роду, внутрішня структура local wide-pipe, яка забезпечує захист від колізій, а також алгоритм стиснення для BLAKE є модифікованою версією добре паралелізованого потокового шифру ChaCha, чия безпека ретельно проаналізована.

HAIFA – ітеративний метод побудови криптографічних хеш-функцій, який є покращеною структурою Меркла — Дамгора.

Потоковий шифр *ChaCha* використовує перемішування за один раунд для покращення криптостійкості. Основний блок системи працює так. Кожна операція змінює одне зі слів. Зміни відбуваються циклічно «на зворотний бік», починаючи з 0-го слова. Чергуючи операції складання та побітової суми разом із зрушенням, складається слово з попереднім.

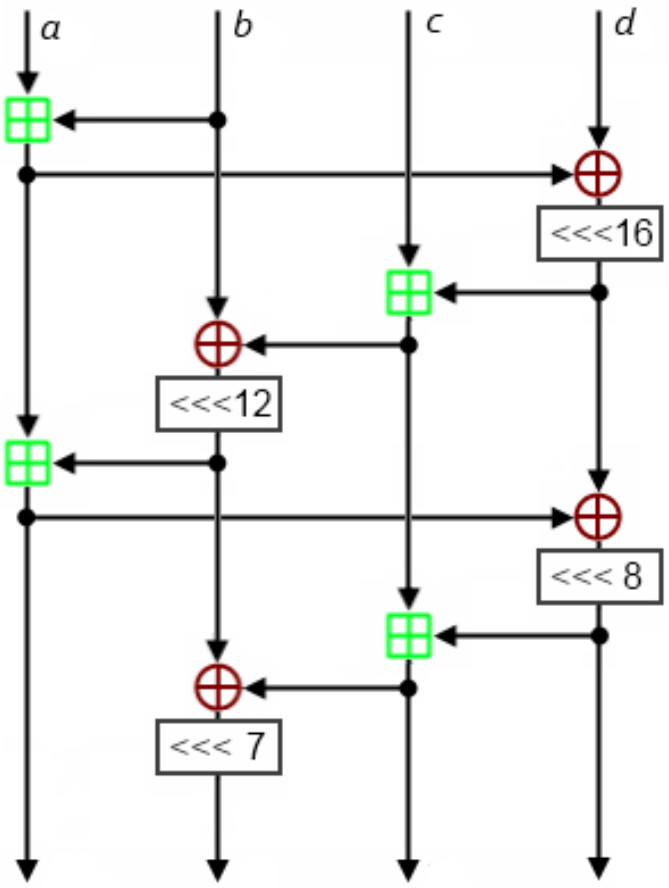


Рисунок 2.2 – Схема роботи ChaCha, функції quarterround(a, b, c, d)

Але зазначена функція не використовується в Blake, використовується її модифікація з додаванням констант до повідомлень і зрушенням напрямку.

Існує два варіанти функції: Blake-256 і Blake-512. Дані хеш-функції використовується для багатьох блокчейнів.

blake2b – Blake-512 оперує з 64-бітними словами і повертає 64-байтний хеш. Дана хеш-функція використовується в таких блокчейнах, як-от BitcoinX, Cash2, Hyperspace, Sia Classic Sia Classic, Siacoin, Title Network, SiaPrime Coin

blake2s – Blake-256 працює з 32-бітними словами і повертає хеш розміру 32-байти. ЇЇ використовують такі блокчейни як: InfoCoin, NevaCoin, Shield, TajCoin, Verge, Volta, Volvox.

Різниця між blake2b і blake2s в кількості раундів і бітності.

bmw – Blue Midnight Wish криптографічна хеш-функція (хф) з виходом у n біт, де n=224, 256, 384 або 512. Дані хеш-функції, призначені для створення «відбитків» повідомлень виробленої бітової довжини. Призначені в різних додатках або компонентах, пов'язаних із захистом інформації.

c11 – цей алгоритм є клоном до x11 алгоритму, алгоритм спеціально розроблений для протидії ASIC.

cryptonight – алгоритм, створений для високої конфіденційності, побудований на основі CryptoNote алгоритму. В CryptoNote час обчислення в більшій степені залежить від швидкості вільного доступу до пам'яті, ніж від швидкості виконання звичайних простих математичних операцій. Цей алгоритм включає в себе: і алгоритм Keccak і функцію губки, містить в собі буфер розміром 2 МБ, для якого виконується вільний доступ до читання та запису, використовуються 64-бітні операції множення, обрахунки раундів шифрування AES, та додаткові хеш-функції: BLAKE, Grøstl, JH, Skein. Плюсом даного алгоритму є висока конфіденційність.

groestl – ітеративна криптографічна хеш-функція. Функція стискання Grøstl складається з двох фіксованих 2n-бітових перестановок P і Q, структура яких запозичена у шифрі AES. Зокрема, використовується такий самий S-блок. Вихідна стрічка хеш-функції може мати довжину від 8 до 512 біт із кроком 8 біт. Варіант, що повертає n-біт, називається Grøstl-n.

equihash – ґрунтується на математичній функції, в основі якої закладений «парадокс дня народження». Парадокс характеризується таким чином: у колективі є 23 людини, з імовірністю 50% є дві людини, народжені протягом одного дня, а збільшення колективу до 60 і більшої кількості людей збільшується до 99%. Дана теорія розраховується по математичній формулі.

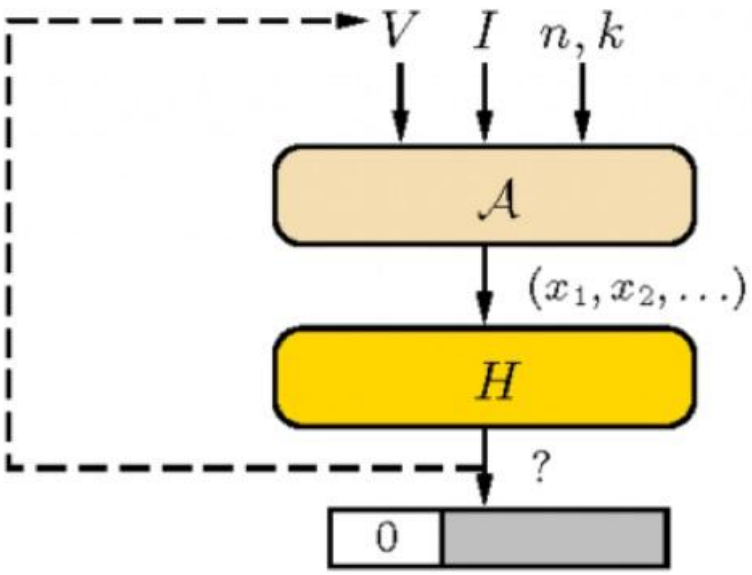


Рисунок 2.3 – Приклад математичної формули Equihash

Для цифрових монет та блокчейнів, що працюють на Equihash, для знахождения вірного хеша використовується узагальнене значення парадоксу, і його можна перефразувати таким чином: ймовірність знаходження правильного хеша майнерами рівня 2 на рівні N, розділеного на 2.

hmq1725 – означає «Highly Modified Quark1725», де 1725 є 17 алгоритмів і 25 хеш-раундів. Щоб забезпечити найвищий рівень безпеки, він використовує 17 алгоритмів, які хешуються 25 разів.

Також HMQ1725 не дуже стійкий до ASIC, але в даний час, згідно з дослідженнями, немає ASIC або FPGA для HMQ1725. Перш ніж вони потраплять на ринок, є шанс видобути монети алгоритму HMQ1725 за допомогою CPU та GPU.

keccak – алгоритм хешування, який побудований на змінній розрядності. Тобто основний принцип алгоритму є принцип криптографічної губки.

lbry – спеціальний алгоритм blockchain використовується для захисту даних, що містяться в транзакціях, які включають metadata of claims and publisher identities.

luffa – криптографічний алгоритм (родина алгоритмів) хешування змінної розрядності. Luffa є варіантом функції губки, криптостійкість якої ґрунтується лише на випадковості основної перестановки. На відміну від оригінальної функції губки, Luffa використовує множину паралельних перестановок та функції інжекції повідомлень.

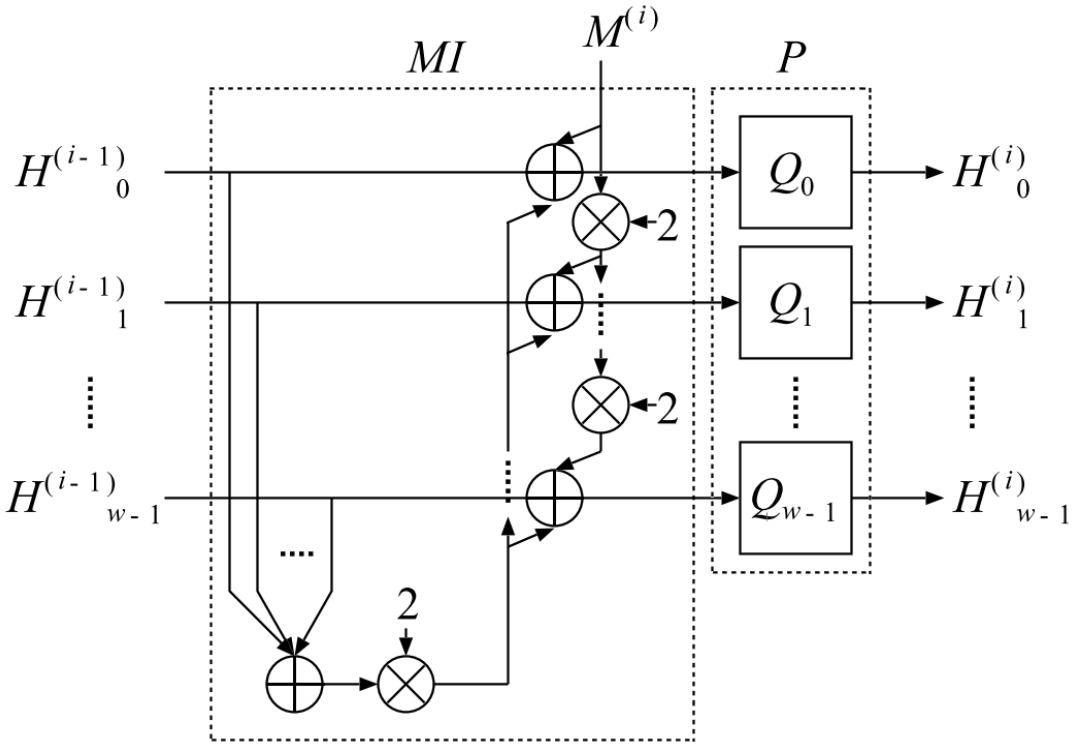


Рисунок 2.4 – Схема раундів функції luffa

lyra2 – це криптографічна хеш-функція, яка також може використовуватися, як функція формування ключа.

neoscrypt – є нащадком оригінального алгоритму Scrypt. NeoScrypt використовує 2 протоколи:

- Salsa20, що складається з 20 раундів (Salsa20/20);
- ChaCha20, також виконує 20 різних раундів (ChaCha20/20).

Вони обидва можуть бути налаштовані як для послідовного, так і для паралельного шифрування:

- для послідовного режиму NeoScrypt використовуватиме (N+3)*r*128 байтів обсягу пам'яті, що відповідає 32,75 Кбіт;
- у паралельному режимі – (2*N+3)*r*128 байтів, тобто 64,75Kb на один епізод.

Конфігурація за замовченням NeoScrypt має такі параметри: 128, 2, 1.

nist5 – це алгоритм майнінгу Proof-of-Work. Цей алгоритм побудований як комбінація алгоритмів хешування. На основі 5 алгоритмів: Keccak, Skein, Blake, Groestl, JH. Ця хеш-функція, орієнтована на пам'ять і спочатку була стійка до ASIC. Проте після 2018 року алгоритм втратив цю можливість. Його . можна видобувати з використанням апаратного забезпечення споживчого класу, такого як досить дешеві процесори та графічні процесори. Однак він не такий стійкий до ASIC, як популярні алгоритми з великим обсягом пам'яті, такі як CryptoNight від Bytecoin, Ethash від Ethereum або Equihash від Zcash.

pentablake – цей алгоритм використовує хеш-функцію 5x Blake 512.

phi1612 – даний алгоритм побудований на основі міксу алгоритмів: Skein, JH, Cubehash, Fungue, Gost і Echo.

phi2 – це покращена версія Phi1612. Створена вона була через збільшення кількості ASIC та FPGA пристроїв. Дані пристрої є потужними обчислювальними машинами, що виконують роль відеокарт у класичному майнінгу.

pluck – побудований на основі SHA256 з великим навантаженням на обчислювання через оперативну пам'ять та центральний процесор.

Polytimos – алгоритм, який включає шість послідовних хеш-функцій: skein, shabal, echo, luffa, fugue, gost.

power2b - алгоритм майнінгу лише на CPU, котрий є модифікацією алгоритму yespower, де змінена функція sha256 на blake2b.

quark – алгоритм для майнінгу криптовалюти, заснований на однорівневій хеш-функції, що складається з 9 рівнів шифрування за шістьма різними криптографічними алгоритмами. Quark невибагливий до великих обсягів оперативної пам'яті пристрою. Також у цьому алгоритмі вбудований високий 64-бітовий захист від хакерських атак. Особливість даного алгоритму є те, що він використуває 6 хеш-функцій, таких як Grostl, Blue Midnight Wish, Keccak, JH, Skein, Blake.

scrypt – криптографічна функція формування ключа, де за основу береться пароль. Засновані на паролі функції формування ключа (функція виведення ключа на основі пароля, PBKDF) зазвичай розробляються згідно вимоги до часу обчислення (за порядку величини — сотні мілісекунд). При використанні потрібно вирахувати подібну функцію один раз (наприклад, при аутентифікації), і таке час від часу допустимо.

Принцип роботи алгоритма скрипта полягає в тому, що він штучно ускладнює підбір варіантів для вирішення криптографічних завдань, наповнених його «шумом». Цей шум – випадково сгенеровані числа, до яких алгоритм Scrypt звертається, збільшуючи час роботи.

Якщо скрипт перевіряє ключ користувача, таке заміщення буде практично незамітним. Але коли код намагаються зламати методом перебору, скрипт це ускладнює: в сумі всі операції займають дуже багато часу.

Для будь-якого блокчейна, побудованого на Scrypt, означає, що майнінг потребує великої кількості учасників у мережі, кожен із яких буде виконувати частину роботи.

Сама функція:

scrypt (P, S, N, r, p, dkLen) = MFcryptHMAC SHA256,SMixr (P, S, N, p, dkLen),

де N, r, p — параметри складності знаходження функції.

Дуже багато блокчейнів створені саме на цьому алгоритмі, так як він один з найбільш популярних на даний час.

shavite3 – криптографічна хеш-функція, створена на поєднанні компонентів AES з фреймворком HAIFA. Ця хеш-функція використовує такі криптографічні примітиви, як мережа Фейстеля та конструкцію Девіса-Мейєра. Особливістю даної хеш-функції є ітерації функцій стиснення і отримана хеш-функція за допомогою алгоритму HAIFA, який дозволяє отримати хеш довільної довжини, що не перевищує 512 біт, де підтримка солі і функція стиснення розроблена з використанням відомих та добре вивчених компонентів: мережі Фейстеля, раундових функцій AES та регістрів зсуву з лінійним зворотним зв'язком.

skein – алгоритм хешування змінної розрядності. Хеш-функція виконана як універсальний криптографічний примітив на основі блочного шифрування Threefish, працюючого в режимі UBI-хешування.

skunk – алгоритм складається з чотирьох різних функцій хешування та одного балансувальника. Функції були об'єднані для створення цього унікального алгоритму, серед них містяться Skein, CubeHash, Fugue та GOST-Streebo.

tribus – алгоритм, який складається з трьох функцій хешування, як-от: JH, Keccak і Echo.

whirlpool – криптографічна хеш-функція, яка хешує вхідне повідомлення 2^{256}, а вихідний хеш на 512 бітів.

x11 – алгоритм ланцюгового хешування x11 використовує послідовність з одинадцяти наукових алгоритмів хешування (Blake, BMW, Groestl, JH, Keccak, Skein, Luffa, Cubehash, Shavite, Simd, Echo) для підтвердження роботи. Ціль даного механізму полягає в тому, щоб розподіл криптовалюти в результаті майнінгу був справедливим, як і на початкових етапах майнінгу Bitcoin. X11 розроблявся для того, щоб зробити ASIC-майнінг складно реалізованим і щоб унеможливити централізацію потужностей майнінгу. До початку 2016 року ASIC-пристрої для алгоритму X11 все ж таки були розроблені і стали становити значну частину хешрейту мережі, проте не призвели до того рівня централізації, яка існує в Біткоїні.

X11 – назва механізму ланцюгового доказу роботи (Proof-of-work), який було введено у криптовалюту Dash. В алгоритмі X11 є підхід, який використовується в криптовалюті Quark, де використовується ланцюгове хешування з додаванням додаткової глибини і складності за рахунок збільшення хешів. При цьому X11 відрізняється від Quark тим, що раунди хешей визначаються заздалегідь, місце вибору випадковим чином.

Алгоритм X11 використовує кілька раундів з 11 різних хеш-функцій, що робить цей алгоритм одним із найбезпечніших і найскладніших криптографічних хеш-функцій, які використовуються в сучасних криптовалютах. Варто зазначити, що назва X11 ніяк не пов'язана із системою з відкритим вихідним кодом, поширеною в UNIX-подібних операційних системах.

x11evo – особливість даного алгоритма, що він є форком x11, а також на ньому є можливість працювати як з CPU, так і з GPU.

x12 – алгоритм клон x11, в якому додана функція хешування: Hamsi.

x13 – алгоритм побудований на алгоритмі x12, в якому лише додана функція хешування: Fugue.

x14 – продовження алгоритму x13 з новою функцією Shabal.

x15 – модифікований алгоритм x14, де додаткова функція виступає як Whirlpool.

x16r – похідний алгоритм з x15, який складається з хеш-функцій: BLAKE, BMW, Groestl, JH, Keccak, Skein, Luffa, Cubehash, Shavite, Simd, Echo, Hamsi, Fugue, Shabal, Whirlpool, Loselose і Djb2. Особливістю даного алгоритму є:

xevan – алгоритм хешування, розроблений як унікальна комбінація подвійного алгоритму X17 зі 128-бітними заголовками. Алгоритм Xevan був розроблений та вперше використаний у криптовалюті BitSend (BSD). Пізніше розробники інших альткоїнів почали впроваджувати алгоритм Xevan у свою криптовалюту, оскільки Xevan є стійким до майнінгу на ASIC, енергоефективним та стабільним.

yescrypt – алгоритм стандарту формування ключа з урахуванням пароля. Yescrypt застосовує повільні криптографічні операції до паролю та salt, створюючи ключ, який підходить для виконання шифрування або зберігання для перевірки пароля в майбутньому. За основу даного алгоритму взятий алгоритм scrypt. Особливістю даного алгоритму є те, що майнінг на CPU швидше працює, ніж на GPU.

yescryptR8..32 – алгоритм форк з yescrypt, де R збільшує розмір блоків, з якими працює алгоритм, і таким чином збільшує використання пам'яті.

yespower - це форк yescrypt, орієнтований на доказ роботи (PoW). У той час як yescrypt є функцією виведення ключа на основі пароля (KDF) і схемою хешування паролів, і, таким чином, призначений для обробки паролів, yespower призначений для обробки пробних вводів, таких як заголовки блоків (включаючи одноразові номери) у блокчейнах на основі PoW.

Сама по собі yespower не є повною системою підтвердження роботи. Поскільки, у випадку використання блокчейну, значення, що повертається yespower, має бути перевірене на те, чи чисельно не перевищує поточну ціль блокчейну (що пов’язано зі складністю майнінгу), або спробу підтвердження (виклик yespower) потрібно повторити (за допомогою іншого одноразового номера) доти, поки умова остаточно не буде виконана (дозволяючи видобути новий блок).

yespowerr16 – це більш ускладнений алгоритм yespower, створений для майнінгу CPU і протидії використання GPU.

yespower-b2b – алгоритм yespower + алгоритм power2b.

Приблизно така кількість різноманітних функцій використовується в блокчейн технологіях.

Для вибору кращого алгоритму для Blockchain системи був проведений глобальний стрес-тест різних алгоритмів на сервері: Ryzen 9 3900 128GB RAM 2x 1TB NVME. Процесор має 12 ядер і 24 потоки і на ньому доступні більшість інструкцій для виконання хеш-функцій, таблиця порівнянь міститься в додатку 1.

Також є алгоритми консенсуса в блокчейн системах. Алгоритм консенсусу блокчейна – це спосіб, завдяки якому децентралізовані ноди мережі досягають згоди про поточний стан даних у всіх блоках.

Один з головних алгоритмів консенсусу є PoW (Proof-of-Work), доказ роботи. Найімовірніше, що саме завдяки біткоїну, алгоритм консенсусу PoW є найбільш відомим способом підтвердження транзакцій у блокчейні. Основна ідея полягає в тому, щоб вузли блокчейн мережі, що підтверджують транзакції, виконували досить складну обчислювальну роботу (прорахунок алгоритму), причому результат має бути легко і швидко перевіреним іншими вузлами мережі.

Перший вузол, який повністю провів усі необхідні обчислення, отримує винагороду від блокчейн мережі. Усі вузли борються між собою, нарощуючи ємність обчислювальних ресурсів, щоб виявитися таким чином, першим вузлом, який отримав винагороду.

Недоліком цього алгоритму є вагомі енергетичні витрати: велика кількість вузлів, які виконують обчислення, але в реальності тільки один, той хто перший проведе успішну роботу, той і отримує винагороду.

Згаданий вище алгоритм використовує Bitcoin та його похідні форки, такі як Litecoin, Dash тощо.

Наступним з популярних алгоритмів консенсусу є PoS (Proof-of-Stake), доказ стейкінгу. Його суть полягає в тому, що творцем наступного блоку в ланцюжку блоків вибирається вузол, який володіє великим балансом – кількістю ресурсів. Наприклад, у криптовалютах створює блок той, у кого на балансі величезна кількість монет. Саме за створення блоку вузол винагороду не отримує, але винагорода виплачується за проведення транзакції.

Можливі два варіанти таких вузлів: вузол з найбільшою кількістю монет, або вузол, який був запущений раніше за всі інші, тобто чим довше вузол працює, тим є більша вірогідність того, що він підтвердить і створить блок, який внесе в блокчейн. Такі блокчейни як Stellar, Decred, Qtum, NEO використовують даний консенсус. Також є монети, які використовують декілька варіантів разом, один з таких яскравих прикладів – це Dash, там і PoW, і PoS одночасно.

Цей вид консенсусу має більше плюсів, ніж мінусів. Одним з таких плюсів є істотне зниження споживання електроенергії для генерації ланцюга порівняно з PoW методом. Але є і недолік такого підходу: з нього простіше робити атаки на мережу. Для створення атаки з подвійною розтратою (Double-spending) необхідно сконцентрувати більше 50% від загальної кількості всіх

монет, при захопленні такого вузла (node) можливо порушувати сам ланцюг і навіть зруйнувати його.

Для боротьби з такими діями, був розроблений консенсус DPoS (Delegated Proof-of-Stake), в основі якого лежить PoS. Одна з особливостей такого алгоритму консенсусу, порівняно з Proof-Of-Stake, є те, що блоки підписують вибрані представники. Власники найбільших балансів вибирають своїх представників, кожен з яких отримує право підписувати блоки в блокчейн мережі. Кожен хто володіє одним або більше відсотками від усіх голосів, потрапляє до Ради. Зі сформованої "ради директорів" по колу вибирається наступний представник, який і підпише наступний блок. У тому випадку, якщо з якої-небудь причини представник пропустив свою чергу в підписанні, він позбавляється делегованих голосів і залишає "раду директорів", після чого на його місце вибирається наступним той кандидат, котрий найбільше підходить.

Власники балансів, делегуючи свої голоси, жодним чином не втрачають над ними контролю, так як в будь-який момент можуть їх відкликати у свого представника. Приклад блокчейнів, де є такий вид консенсусу, – це EOS.

Є ще один варіант модифікації PoS консенсусу, такий як LPoS (Leased Proof-of-Stake). На даний момент цей консенсус використовує не так багато криптовалют, одна з популярних криптовалют WAVES. У рамках цього алгоритму консенсусу, будь-який користувач має можливість передавати свій баланс в оренду вузлам, які займаються майнінгом, а за це майнінг-вузли діляться частиною прибутку з ними. Таким чином, даний алгоритм консенсусу дозволяє отримати дохід від майнінгової діяльності, не виконуючи самого майнінга.

Третій з таких алгоритмів консенсусу є PoC (Proof-of-Capacity), доказ пропускної здатності. Цей консенсус використовує тільки один альткоїн Burstcoin. Особливість PoC полягає в таких принципах:

- кожен майнер обчислює досить великий обсяг даних, який записується на дискову підсистему (жорсткий диск, хмарні системи зберігання) вузла. Такий початковий набір даних в PoC називається "ділянка";

- для кожного нового блоку в блокчейні, майнер читає невеликий набір даних (1/4096, що приблизно становить 0.024%) від свого загального збереженого обсягу і повертає величину пройденого часу в секундах з моменту створення останнього блоку, після якого майнер зможе створити новий блок;

- майнер, який отримав мінімальний час дедлайну, підписує блок і отримує винагороду за транзакції.

Четвертий вид консенсусу PoI (Proof-of-Importance), доказ важливості. Алгоритм консенсусу, що використовується блокчейном NEM. Важливість кожного користувача в мережі NEM визначається кількістю коштів, наявних у нього на балансі, і кількістю проведених транзакцій з/на його гаманець. На відміну від більш звичного PoS, який враховує тільки баланс наявних коштів у користувача, PoI враховує як кількість коштів, так і активність користувача в блокчейн мережі. Завдяки такому підходу залучається більше користувачів не просто тримати кошти у себе на рахунку, а й активно їх використовувати.

П’ятий вид консенсусу PoA (Proof-of-Activity), доказ активності. Його суть полягає в тому, що автори алгоритму PoA спробували об'єднати два найбільш популярних алгоритми, такі як Proof-of-Work і Proof-of-Stake, з метою збільшення рівня захисту від потенційно можливих атак – це 51% attack, Denial-of-Service attacks (DoS).

Принцип роботи такого алгоритму наступний:

1. Кожен видобувач блокчейн-мережі пробує згенерувати заголовок порожнього блоку, який включає в себе хеш попереднього блоку, публічну адресу майнера, індекс поточного блоку в блокчейні і nonce число.
2. Після генерації заголовка порожнього блоку відповідає поточним вимогам складності, вузол розсилає цей заголовок у блокчейн мережу.
3. Усі вузли мережі розглядають заголовок такого блоку, як дані отримані від псевдовипадкових власників. Використовуючи хеш розісланого заголовка блоку і хеш попереднього блоку + N пресетів з використанням алгоритму follow-the-satoshi вибираються стейкхолдери.
4. Кожен стейкхолдер, що знаходиться в онлайні, перевіряє отриманий, порожній заголовок блоку на його коректність. Під час перевірки кожен, хто отримав заголовок, перевіряє чи є він одним з перших N-1 стейкхолдерів "щасливчиків" цього блоку, в цьому випадку підписує заголовок порожнього блоку своїм секретним ключем і відправляє його в блокчейн мережу.
5. Коли n-й стейкхолдер бачить, що він повинен стати підписантом цього блоку, він, на додаток до заголовка порожнього блоку, додає блок з включеними транзакціями (кількість включених транзакцій вибирає сам), всі підписи N-1 від інших стейкхолдерів та підписує блок.
6. Стейкхолдер N розсилає новий, підготовлений блок. Вузли отримують цей блок, переконуються в його законності і додають цей блок у блокчейн.

7. Премія за транзакції, яку отримав N-стейкхолдер, розподіляється між майнером і N стейкхолдерами "везунчиками".

Одна з монет, яка використовує даний консенсус, є Decred.

Шостий вид консенсусу PoAuthority (Proof-of-Authority), доказ авторитетності. Цей вид консенсусу використовує одна з популярних монет, як VeChain. PoA алгоритм консенсусу відрізняється від інших алгоритмів тим, що для своєї роботи йому не потрібно мати взагалі ніякого майнінга, як у випадку з PoW або POS. У блокчейн мережі, що базується на PoAuthority, всі транзакції і блоки перевіряються за допомогою схвалених акаунтів (валідаторів). Проведення транзакцій і створення блоків проходить в автоматичному режимі за допомогою обчислювальних потужностей валідатора.

Позитивним моментом даного алгоритму є відсутність майнінгу і, як наслідок, величезне зниження витрат на його обслуговування.

Негативний момент використання даного алгоритму: валідатори приводять до централізації мережі. Це робить мережу вразливою, так як сама суть блокчейну – це децентралізований ланцюг блоків. У приватних мережах, для вузькоспрямованих задач ймовірно це має сенс, так як там не потрібно децентралізації як такої.

Сьомий вид алгоритму консенсусу Proof-of-capacity або Proof-of-space (PoSpace) – метод захисту криптовалют, заснований на використанні вільного місця на пристрої зберігання файлів заздалегідь виділеного користувачами. Метод є менш енерговитратним способом у порівнянні з PoW. Прикладом монет на такому консенсусі є Chia. За допомогою відкритого ключа генерується файл певного розміру, в якому генеруються усі можливі ключі. Далі вказаний файл використовується для пошуку наступних блоків. Майнінг на жорсткому диску відрізняється тим, що для виконання роботи не потрібно купівлі дорогих відеокарт, блоків живлення великої потужності і материнських плат з величезним числом роз'ємів.

Алгоритм Proof of Space дуже схожий на алгоритм консенсусу Proof of Work, тільки в ньому замість обчислювальної потужності використовується обсяг пам'яті для зберігання.

Восьмий і заключний алгоритм консенсусу є PoB (Proof-of-Burn), доказ спалювання. Цей алгоритм є популярний у монетах, де є величезна кількість монет, тобто велика емісія. Він відбувається наступним чином: майнер відправляє монети на випадкову адресу з випадкового згенерованого хешу, витратити кошти з цієї адреси практично неможливо, так як ймовірність підібрати до нього ключі прагне до нуля. За таке спалювання монет майнер отримує постійний шанс знайти PoB блок і отримати за нього нагороду. Шанси на майнінг збільшуються при збільшенні кількості спалених монет. Економічно цей процес спалювання монет можна уявити, як купівлю бурової установки для майнгінгу. Природно, що такий алгоритм має сенс використання тільки на пізніх етапах існування тієї чи іншої криптовалюти, тоді коли є що "спалювати". Сам алгоритм не є популярним, його використовують дуже мало монет. Однією з таких монет була Slimcoin.

Як було зазначено, в кожного консенсусу є позитивні і негативні моменти. Щоб обійти дану проблему, необхідно комбінувати алгоритми. Одною з таких популярних комбінацій є PoS + PoW. Це є актуально, коли Proof-of-Work консенсус упирається в складність мережі, комбінування допомагає добувати блоки зразу двома способами і збільшувати кількість користувачів криптовалюти. Одні майнять за допомогою апаратури і знаходять блоки по певному алгоритму, а інші запускають основні вузли (master node) з великою кількістю монет і теж добувають блоки.

Метою гібридних систем Proof of Work і Proof of Stake є виявлення переваг відповідних підходів і їх використання для врівноваження слабких сторін один одного. Decred – це одна з небагатьох криптовалют, яка

використовує PoW і PoS в їх початкових формах і об'єднує їх разом для створення багатофакторного або гібридного консенсус механізму.

"Монети мастернод" в певному сенсі також є гібридами тому, що вони мають один з компонентів, що пов'язаний з Proof of Work, який виконує ту ж роль, що і в Біткойн, і додаткову роль для спеціальних вузлів. Як правило, існує обов'язкова вимога, щоб ці спеціальні вузли тримали певну кількість валюти, щоб продемонструвати, що їм можна довіряти, і вони діють в найкращих інтересах мережі, що за аналогією має відношення до Proof of Stake.

Це непоганий варіант, так як пропонує на вибір різні альтернативи. Хороший приклад такої монети – це Dash. Дана монета має алгоритм X11, його дуже складно майнити, так як сам алгоритм використовує ланцюжок з 11 алгоритмів типу криптографічної хеш-функції для доведення виконання роботи. Сам алгоритм X11 придумав головний розробник Dash з метою ускладнити використання спеціалізованого обладнання для майнінгу. Для того було в монеті введена альтернатива як мастерноди. На адресу відправляються 1000 монет і вона реєструється в мережі мастернода. Мастерноди використовуються для приватної відправки монет через змішування (неможливість відслідкувати відправника).

Можна ще для прикладу навести монету NIX. Алгоритм генерації монет Lyra2REv2. Цей алгоритм якісно обробляється на GPU. Тобто, майнери майнять монету на відеокартах та збільшують емісію. Спочатку був консенсус PoW (для початкової генерації монет в обіг), а потім вся криптовалюта перейшла в консенсус LPoS, коли майнінг вже неможливий, а тільки стейкінг через мастерноди.

На даний момент на ринку представлено понад 8000 різноманітних криптовалют, які працюють на неоднакових алгоритмах консенсусу, на відмінних алгоритмах генерації монет, та з абсолютно різними функціями.

2.2. Розроблення алгоритму хешування

На основі порівнянь 82 функцій хешування, було вибрано алгоритм хешування **scrypt** та зроблена його модифікація шляхом комбінації з іншим алгоритмом yescrypt для покращення захисту даних. Для побудови самого блокчейна був використаний консенсус Proof of Work, коли емісію монет для майбутніх транзакцій створюють майнери.

Кроки алгоритму хешування, розробленого у роботі:

1) штучно ускладнюється підбір варіантів для вирішення криптографічних завдань, наповнених його «шумом» (випадково згенеровані числа, до яких алгоритм Scrypt звертається, збільшуючи час роботи).

2) якщо скрипт перевіряє ключ користувача, таке заміщення буде практично непомітним, проте, коли код намагаються зламати методом перебору, скрипт це ускладнює: в сумі всі операції займають дуже багато часу.

3) застосування функції:

scrypt (P, S, N, r, p, dkLen) = *MFcryptHMAC SHA256,SMixr* (P, S, N, p, dkLen), де N, r, p — параметри складності знаходження функції (рис. 2.5).

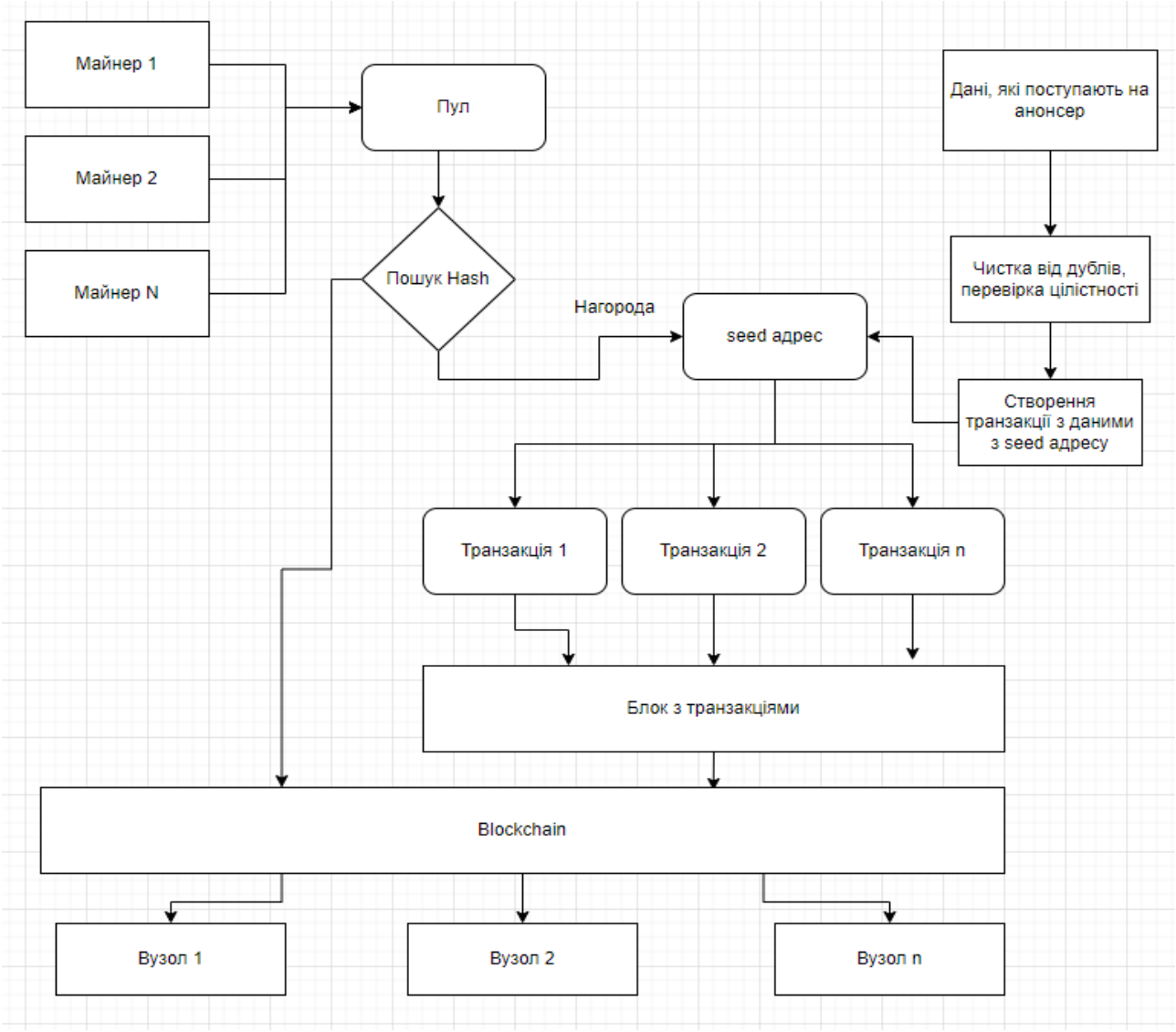


Рисунок 2.5 – Алгоритм опрацювання даних у блокчейн

Даний хеш-алгоритм можна майнити на CPU, GPU, ASIC схемах і час генерації 1 блока рівний 1 хв. Розмір блока рівний 100 МБ, максимальний розмір транзакцій дорівнює 100 MB за хвилину. Використовуєтться консенсус PoW, Proof of Work, коли для емісії необхідні майнери. Розроблений алгоритм достатньо швидкий для підтвердження транзакцій і пошуку нових блоків, тому має більше плюсів, ніж мінусів.

2.3. Розроблення інформаційної моделі Великих даних та їх послідовного опрацювання

Великі дані в блокчейн – це послідовні дані, подані у великому обсязі, які відправляють клієнти на анонсер, що підключені до мережі:

Block::=<*BlockHead*, *Data*>,

де *BlockHead* гарантує цілісність *Data*.

BlockHead::=<*PreviousBlockHash*, *DataHash*, *Moment*, *Difficulty*, *Timestamp*>

PreviousBlockHash – це хеш-значення попереднього блоку заголовку. *DataHash* – хеш-значення *Data* у цьому блоці. *Moment* – значення, яке визначається шляхом випадкової перевірки.

$$Zero\ (Hash\ (PreviuosBlockHash \parallel DataHash \parallel Timestamp \parallel Moment)) \geq Difficulty,$$

де *Zero*(▪) – функція, яка повертає кількість лівих послідовних нулів у вхідному рядку в байтах; *Difficulty* – ціле число, яке визначає вимогу щодо кількості послідовних нулів у голові результату хешування. Наприклад, Difficulty=3 означає, що перших три байти вихідного хешу дорівнюють нулеві, отже, перших 3 байти з *Hash*(▪) є нулями. *Timestamp* – час поточного блоку.

Data::=<*Metadata*, *BData*>,

де *Metadata* – кортеж дескриптора для *BData*, *BData* – обов’язковий кортеж з записом даних. Приклад запису для відображення неструктурованимх даних поданий у Таблиці 2.1.

Таблиця 2.1 – типи даних, які отримує анонсер з клієнта.

Ключ hash для ідентифікації роздачі	info_hash
IP адреса в форматі ip2long	info_ip
Інформація про порт клієнта	info_port
Інформація про піра	info_peer
Унікальний хеш ключа, IP, порта в md5	info_md5
Унікальний хеш ключа, IP, порта в sha1	info_sha1
Інформація про роздане, в байтах	info_upload
Інформація про скачане, в байтах	info_download
Залишилось скачати, в байтах	info_left
Мітка часу останнього онлайну піра	info_update
Мітка часу зі зміщенням на 1 годину	info_expire
Інформація про IP, звичайний формат	info_ipv46

Metadata складається з таких даних як:

$$\textbf{Metadata}::=<LSource, DataType>,$$

де *LSource* – джерела даних, а *DataType* – мітка для структурованих, напівструктурованих чи неструктурованих даних.

$$PreviousBlockHash::=Hash(BlockHead),$$

де *BlockHead* – заголовок з попереднього блоку. Тобто хеш-значення попереднього заголовку вбудовується в наступний блок. Його також можна розглядати як ланку двох сусідніх заголовків.

$$DataHash::=Hash\ (BData_1 \parallel BData_2 \parallel \ldots \parallel BData_n).$$

Отже, проблема, яка вирішується за допомогою побудованої моделі – це наявність такої великої кількості даних, що стандартні реляційні бази даних не

витримують навантаження. Для прикладу в роботі було обрано базу даних в MySQL, оскільки така СУБД – одна з найчастіших СУБД для інтернет-застосувань. Проводилися дослідження і в PostreSQL, суттєвої різниці не спостерігалось.

Середня статистика за 1 годину роботи з базою MySQL, яка тестувалася в роботі, зображена на рисунку 2.6.

Запитань із моменту запуску: 47,571,820,059

ø за годину: 13,551,837

ø за хвилину: 225,864

ø за секунду: 3,764

Параметри	#	ø за годину	%
select	34,006 M	9,687.2 k	71.48
insert	7,020 M	1,999.7 k	14.76
update	6,546 M	1,864.9 k	13.76
set option	4,968	1.4	<0.01
truncate	2,974	0.8	<0.01
alter table	644	0.2	<0.01
show tables	361	0.1	<0.01
show variables	330	0.1	<0.01
drop table	329	0.1	<0.01
create table	327	0.1	<0.01
unlock tables	323	0.1	<0.01
lock tables	323	0.1	<0.01
change db	226	0.1	<0.01
show fields	91	<0.1	<0.01
show master status	83	<0.1	<0.01

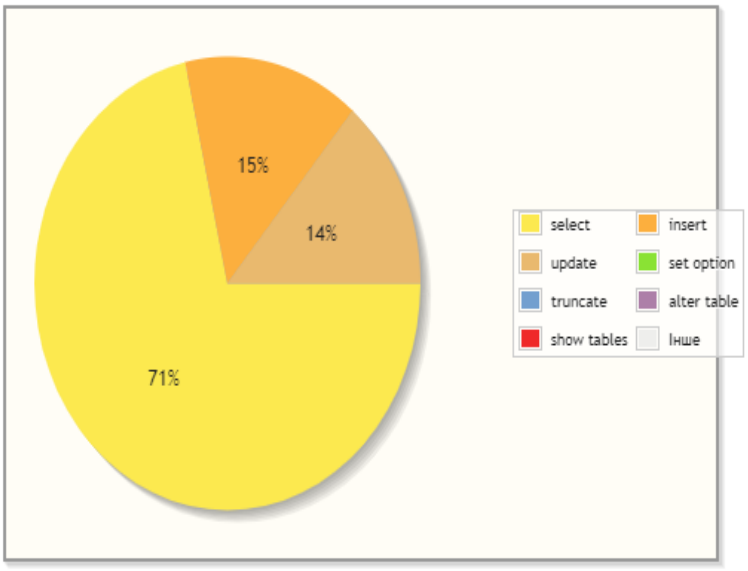


Рисунок 2.6 – Приклад кількості запитів в статистиці MySQL сервера

У середньому здійснювалося 13 мільйонів запитів за 1 годину, з них 14.76% – це запити на вставлення нових даних і 13.76% - на перезапис існуючих.

Отже, виходячи зі статистики з анонсера, здійснено 1.36 мільярдів запитів за 24 години (рис. 2.7).

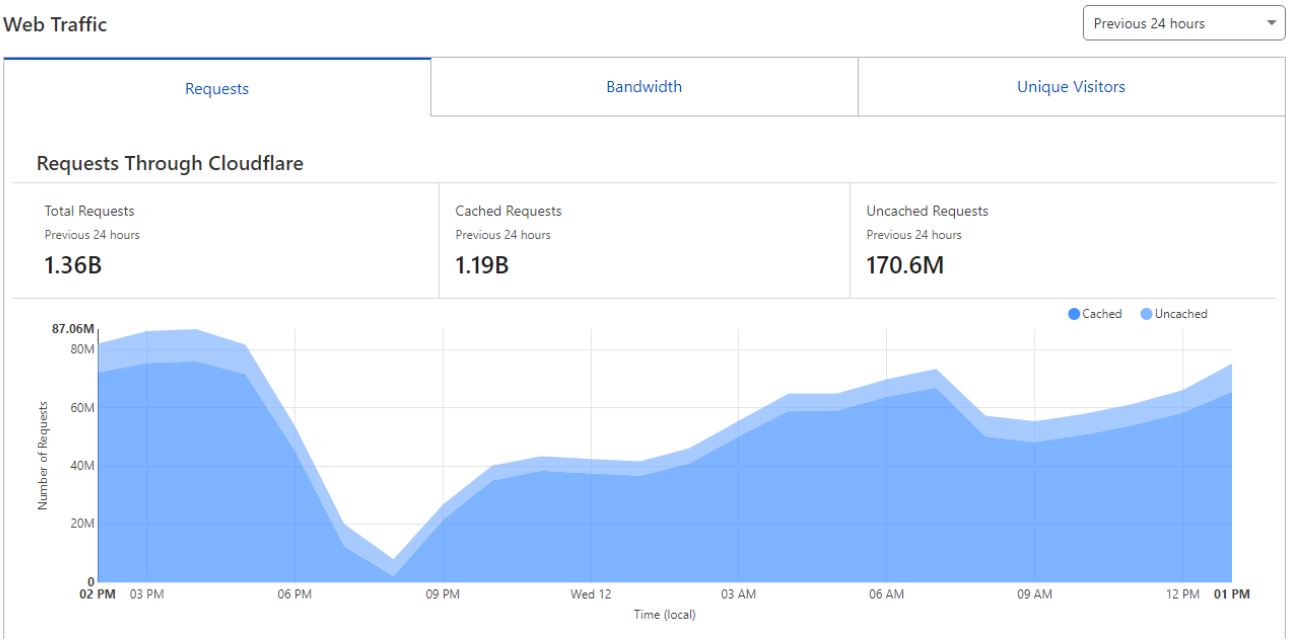


Рисунок 2.7 – Навантаження запитів на анонсер

Через таке високе навантаження MySQL не справляється з такою кількістю запитів, навіть на потужних серверах. Для цього використано розроблену модель даних, запис даних у Blockchain і їх вибірка через RPC-API з blockchain демона.

Кількість унікальних користувачів на аносері за 24 години (рис. 2.8).

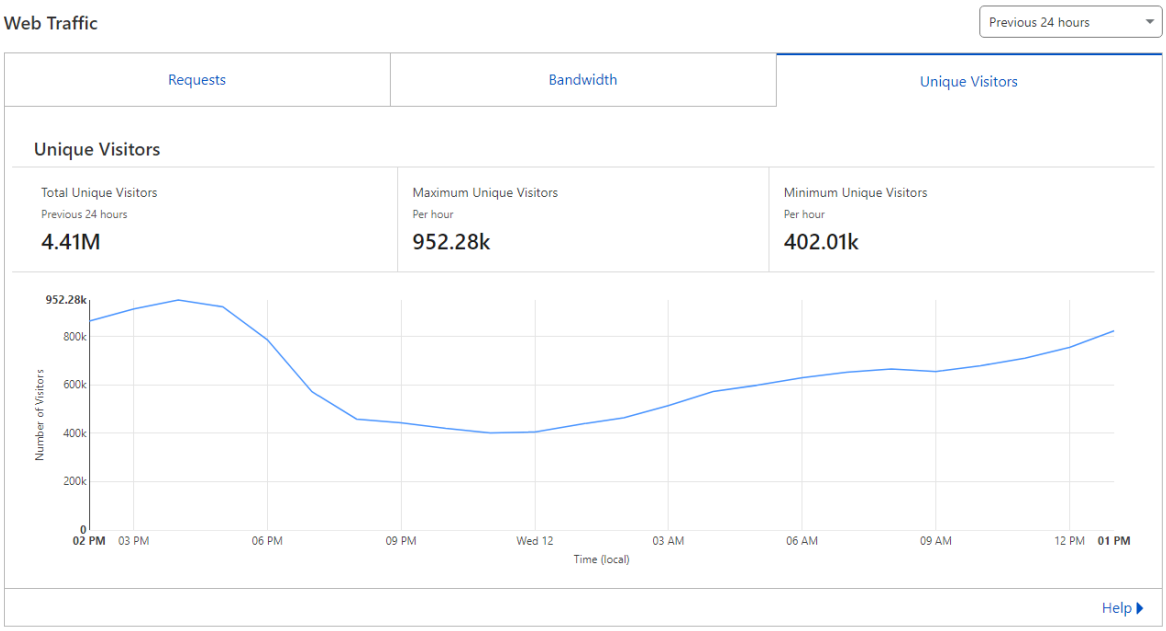


Рисунок 2.8 – Розширена статистика запитів

Зміст самих даних продемонстровано в таблиці 2.2 і на рисунку 2.9, де зображено їх обсяг за певний час з тимчасової бази даних.

Таблиця 2.2 – Вміст даних.

info_hash	F6B4AE4D3D3A775D1EF492F0F068341C6D8558A8
info_ip	1024644124
info_port	18181
info_peer	32443432343333303331333733333244464441454545453742 45…
info_md5	b2648dac14b071010c0e8ecf75ac0e49
info_sha1	f21d59ea795ef2ed705c30c3daf58b282e0286fa
info_upload	34488320
info_download	48611329
info_left	2439057652
info_update	2022-01-12 13:00:29
info_expire	1641988829
info_ipv46	61.18.212.28

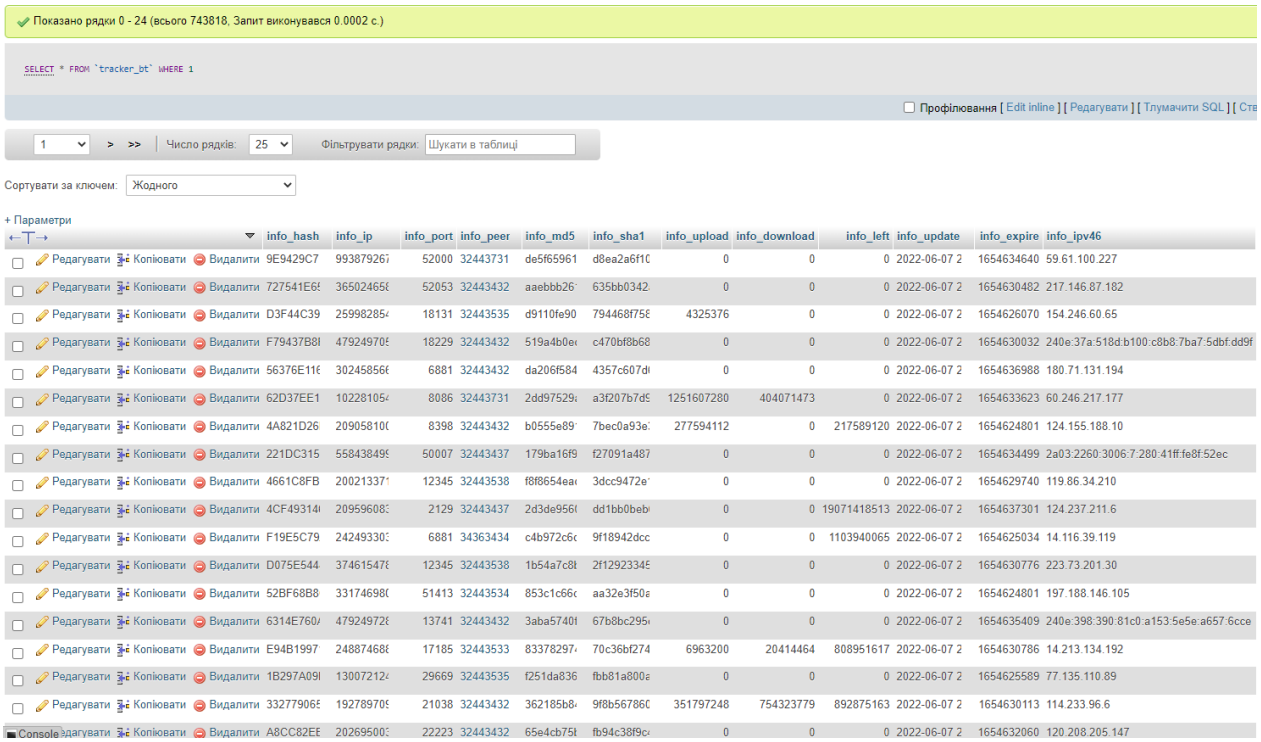


Рисунок 2.9 – Вибірка даних зі списку з тимчасової таблиці

Оскільки дані є статичними і рідко змінюються, міняючи лише при цьому параметри, можна використовувати блокчейн для їх запису і читання.

info_hash – унікальний хеш торрент роздачі. Складається зі вмісту даних, перетворених в словник, і взятий з нього хеш sha1.

У цей хеш входять:

- ITEM: розмір і шлях до файла.
- Ім’я: Ім’я для пошуку.
- Довжина pieces: довжина одного фрагмента даних.
- Фрагменти: SHA1 Hash з кожного фрагменту торрента.
- Приватний: прапор обмеженного доступу.

Для прикладу, якщо взяти випадковий торрент файл і декодувати його з Bencode формату, отримуємо такий зміст, додаток 3. На виході отримуємо продемонстрований зміст, додаток 4.

Потім зазначену стрічку кодуємо в sha1 і отримуємо на виході: 7EDA978ED7628595BB91C48B947F025BAE78CB77. Це info_hash даного торрент файла, тоді зазначений параметр буде основним для створення закритого ключа доступу до відкритого ключа на блокчейні.

info_ip – IP клієнта, у форматі ip2long, для прикладу: 127.0.0.1 буде 2130706433. Даний формат використовується в клієнтах і через те, що він в interger виді, відповідно по ньому дуже швидко здійснюється індексований пошук.

info_port – порт клієнта, з якого був відправлений пакет з даними на анонсер.

info_peer - інформація про піра в hex форматі для оптимізації місця.

info_md5 – контрольна сума info_hash та info_peer у форматі хешування md5.

info_sha1 – контрольна сума info_hash та info_peer у форматі shda1 для позбуття колізій, якщо будуть однакові суми в md5.

info_upload, info_download, info_left – статистика розданого, скачаного і скільки залишилось з клієнта.

info_update – останній раз пір був онлайн, контрольна часова мітка.

info_expire – коли закінчується контрольна часова мітка, для повторного оновлення статистики і даних про клієнта. У стандартному обновленні кожні 120 хвилин, для високої точності можна зменшити даний ліміт до 120 секунд, але навантаження на анонсер зросте в тисячі разів.

info_ipv46 – IP адреса клієнта в стандартному форматі IPv4, IPv6.

Для послідовного опрацювання і запису використовується анонсер, на який задають запити клієнти.

Формат запитів на анонсер з клієнтів:

41.210.4.45 - - [12/Jan/2022:14:32:41 +0100] "GET /announce?info_hash=%3dI%ac%f7%b0%12%24%80%24E%9b%c8%f35%d7S%f6%29%b1%d9&peer_id=-UT355W-%10%b4%fd%5d%3e%3al%14%2c%ebZz&port=10757&uploaded=0&downloaded=0&left=0&corrupt=0&key=0496B26B&event=stopped&numwant=0&compact=1&no_peer_id=1 HTTP/1.0" 200 329 "-" "uTorrent/355(111916048)(46096)" 118.250.161.197 - [12/Jan/2022:14:32:41 +0100] "GET /announce?info_hash=Y%25%26%1E%F6%8Bt%5C%90%CFw_%B9%CD%B0%EB%A9%09%D6%DB&peer_id=-BC0184-%B1%87%E4c%90%BD%0F%BC%3A%E5d%93&port=22223&natmapped=1&localip=192.168.124.24&port_type=lan&uploaded=0&downloaded=0&left=0&numwant=200&compact=1&no_peer_id=1&key=29642&event=started HTTP/1.0" 200 33 "-" "BitComet/1.84.11.29"

Декілька уточнень по параметрах:

- параметр event має два значення – started і stopped (запущена та зупинена роздача, причому для зупинених роздач використовується scrape анонсер).
- numwant – максимальна кількість пірів для зчитування.
- compact – режим показу віддаючих даних, компактний формат чи розширений, в залежності від клієнта. Різні клієнти по різному приймають дані.

Компактний формат виглядає ось так:

d8:intervali8481e5:peers300:>��8]TM�-��eP�|��gZ\[�[��L��^ﻥ_&:��f@��@f@���9uf�Q�~�f����"f����yiD·3ie�S ii����i�G

Розширений формат подано нижче:

d8:intervali8605e5:peersld2:ip13:62.210.203.564:porti23892eed2:ip13:77.141.30.1824:porti35941eed2:ip11:90.92.26.914:porti62348eed2:ip13:91.245.254.764:porti34018eed2:ip13:94.23.222.1634:porti55045eed2:ip12:95.138.38.584:porti47062eed2:ip14:102.64.166.2354:porti32576eed2:ip14:102.64.186.1844:porti14709eed2:ip14:102.129.81.2524:porti32498eed2:ip13:102.176.253.54:porti41250eed2:ip15:102.176.253.2304:porti63097eed2:ip14:105.68.189.2034:porti38451eed2:ip13:105.101.6.1804:porti21254eed2:ip15:105.105.214.2214:porti63925eed2:ip14:105.235.71.1424:porti52095eed2:ip15:109.128.238.1194:porti

34604eed2:ip12:154.0.27.1484:porti24038eed2:ip12:154.0.27.2034:porti15483eed2:ip12:154.0.27.2404 eeee

В основному Великі дані даного проекту подані за допомогою унікальних хешів, IP портів і ID пірів. За день отримуються декілька десятків мільйонів таких даних.

2.4. Розроблення методу запису Великих даних у блокчейн

Метод запису передбачає чотири етапи.

Етап 1. Структуризація даних, вхідні дані сортуються та автоматично очищуються від повторів.

Етап 2. Групування за змістом та довжиною даних. Далі здійснюється перетворення даних в hex вигляд і створення raw транзакції на основі hex даних.

Етап 3. Виконується перевірка на існування даних у блокчейн системі. Якщо такий запис вже існує, то він не записується, а записується мітка вказівник, де знаходяться зазначені дані.

Етап 4. Здійснюється запис у блокчейн за допомогою raw функцій перетворення даних в транзакцію.

У даній роботі, info_hash виступає як унікальний ідентифікатор роздачі. Вигляд даних у тимчасовій таблиці реляційної бази даних подано у Таблиці 2.3.

Таблиця 2.3 – Вигляд даних у тимчасовій таблиці реляційної бази даних

info_hash	45EABA5A4A93F11B211C7600B62EF4F228AA0A13
info_ip	3563463634
info_port	24487
info_peer	3244373134323334333433323330324436373446344336433434353536423530363335353435333800000000000000000000
info_md5	5fb6ec2c5e831587bafa224a9c94fa54
info_sha1	e729705820ca47c2399cb142b7c0d56f570f4719
info_upload	34488320
info_download	48611329
info_left	2439057652
info_update	2022-01-12 13:00:29
info_expire	1641988829
info_ipv46	61.18.212.28

Далі дані послідовно перетворюються в стрічку через розділовий знак | без ідентифікатора info_hash:

3563463634|24487|3244373134323334333433323330324436373446344336433434353536423530363335353435333800000000000000000000|5fb6ec2c5e831587bafa224a9c94fa54|e729705820ca47c2399cb142b7c0d56f570f4719|34488320|48611329|2439057652|1641992429|1641988829|61.18.212.28

Ідентифікатор виступає ключем, до якого прив’язані дані:

45EABA5A4A93F11B211C7600B62EF4F228AA0A13

На основі ідентифікатора створюється адреса вигляду: 2ExVocnskqgHXHmPSVZRBk5LWb8NsCXmf9. Тут міститимуться транзакції з даними про пірів на роздачі. Тобто Blockchain виступає в ролі

децентралізованої бази даних для швидкого запису і доступу до даних через вбудоване RPC API. Це дає змогу зменшити час доступу до даних, про що буде розглянуто в четвертому розділі.

Великі дані перекодовуються зі зрозумілого стрічкового формату в hex варіант з метою запису цих даних через raw транзакції:

3335363334363336333347c32343438377c333234343337333133343332333333343333333433333333333333303332343433363337333434363334343333363433333343334333533353336343233353330333363333333353335333433353333333830303030303030303030303030303030303030307c356662366563326335653833313538376261666132323461396339346661353347c65373239373035383230636134376332333939636231343262376330643536663537306634373139 7c33343438383332307c34383631313332397c323433393035373635327c3136343139393234323 97c313634313938383832397c36312e31382e3231322e32380d0a

Цей код є стрічкою даних одного запису в hex варіанті.

Для того, щоб записувати дані в блокчейн, необхідні монети.

Емісію даних монет створюватимуть майнери, вони в свою чергу будуть підтримувати мережу, формувати нові блоки з транзакціями та отримувати винагороду на seed адресу для наступних транзакцій.

Seed адреса зі свого боку виступає якорем для наступних транзакцій (рис. 2.10).

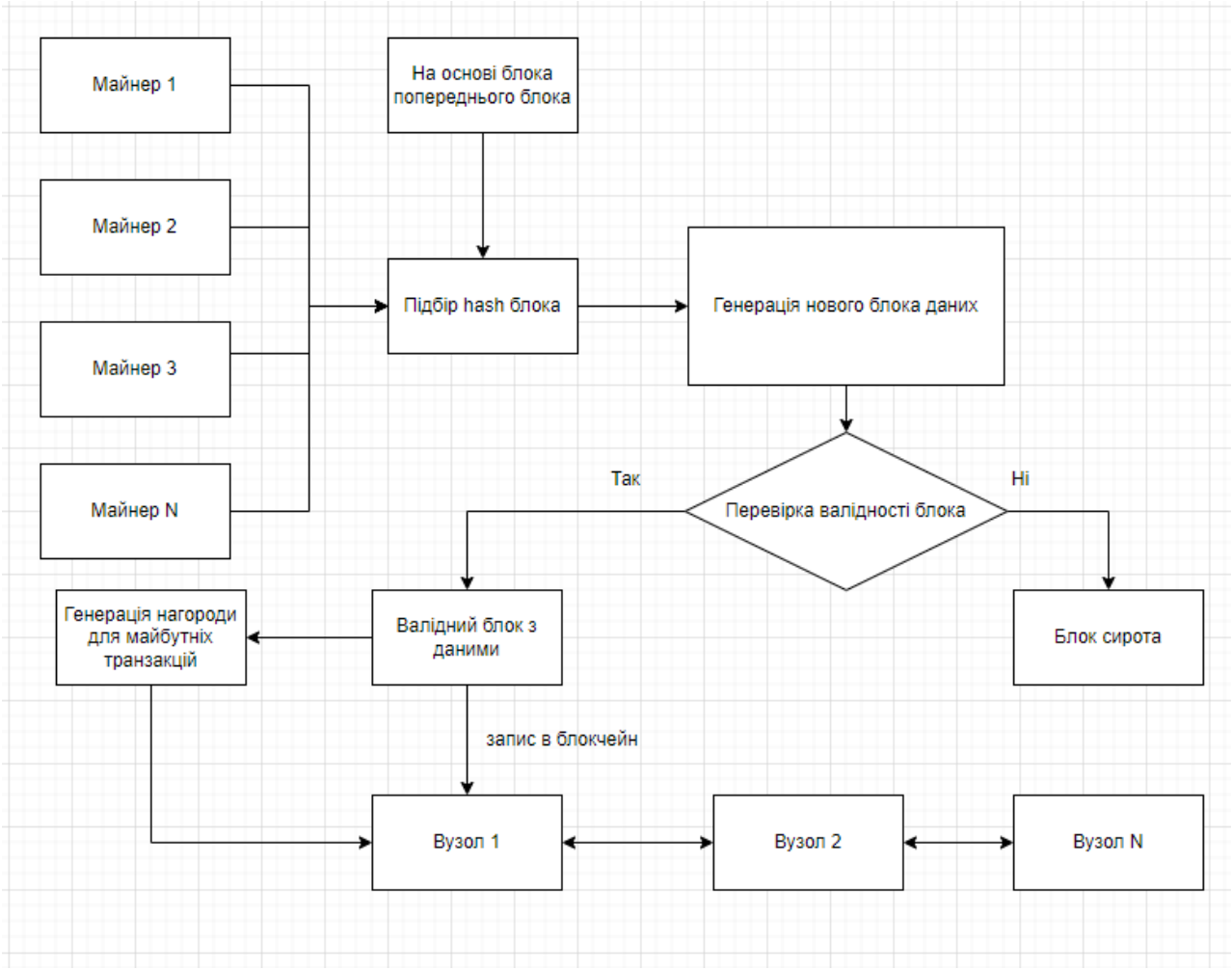


Рисунок 2.10 – Принцип записування в блокчейн

2.5. Розроблення методу перевірки якості внесених даних.

Метод передбачає два етапи. На першому етапі проводиться пошук унікальних info_md5 контрольних сум. На другому – здійснюється пошук info_sha1 контрольних сум. Якщо суми не мають повторів, тоді дані пишуться в блокчейн.

Отже, метод перевірки якості внесених даних для моделі Великих даних на основі блокчейна виглядає так:

Етап 1:

(1) Кожний вузол загальнодоступного блокчейну, позначений як *B*, зазвичай є вузлом зі своїми власними незалежними даними *Bdata*. Необхідно встановити відкритість і обмін даними *Bdata* між усіма вузлами і підтримувати непоправну модифікацію даних і відстеження часу.

(2) Центральний вузол загальнодоступного блокчейну записує дані *Data*, які потрібно зберегти на локальному носії даних, і рівномірно запаковує їх у *BlockHead* кожний період часу *P*.

Етап 2:

(3) Припустимо, що дані *Data*, записані за час *P*, подані як $BData_1$, $BData_2$, … $BData_n$. Метод рівномірного запаковування блоків у *BlockHead* полягає в наступному: блоки записуються як *Data*, а компоненти блоків включають $BData_1$, $BData_2$, … $BData_n$.; кожен *Data* містить метадані, модифіковані дані *Data* та адресу доступу до змінених даних; обчислити *DataHash*::=*Hash* ($BData_1$ || $BData_2$ || … || $BData_n$).

(4) Заголовок має назву *BHEAD* і включає *PreviousHash*, *BHASH*, *LinkOfDATA*, *Timestamp*, *Moment* і *Requirement*. *PreviousHash* — це хеш-значення попереднього блокхеда; *LinkofDATA* – адреса доступу поточного блоку *BDATA*; *Timestamp* — це мітка часу для створення головоломків; *Moment* є випадковим числом; *Requirement* — це вимога до хеш-значення *Hash*(*PreviousHash* || *BHASH* ||*LinkOfDATA* || *Timestamp* || *Moment* ||*ID*).

(5) Кожний центральний вузол додає дані до всього загальнодоступного блокчейну. Він повинен обчислити значення *Moment*, яке відповідає *Requirement*. Відповідно до природи хеш-функції обчислення *Moment* можуть покладатися лише на випадкові спроби. Далі центральний вузол транслює обчислений блок, і якщо всі проходять перевірку, він вважається правильним. Крім того, блок записується, а винагорода буде завершена в автономному режимі відповідно до статистики *ID*.

(6) Захищені паролем хеш-функції включають SHAH1, SHAH256, MD5.

2.6. Висновки

У даному розділі проведений детальний аналіз методів хешування даних у блокчейні зі всіх можливих, описано та проаналізовано алгоритми консенсуса, необхідних для роботи з блокчейном. Розроблено модель Великих даних для їх послідовного опрацювання, розроблений та продемонстрований, а також описаний метод перевірки якості внесених даних.

Показано, для яких типів даних доцільно застосовувати блокчейн та оцінено навантаженість (кількість запитів) на реляційну базу даних. Так, для однієї з досліджуваних баз даних здійснено 1.36 мільярдів запитів за 24 години. Через таке високе навантаження MySQL не справляється з такою кількістю запитів, навіть на потужних серверах. Це ще раз підтверджує, що використання децентралізованої бази даних все одно не дасть змогу зменшити час доступу до даних та обробити усі запити. Для цього використано розроблену модель даних, запис даних у Blockchain і їх вибірка через RPC-API з blockchain в процесі, що працює у фоновому режимі.

Результати розділу опубліковано у працях автора [3, 4, 5].

РОЗДІЛ 3. РОЗРОБЛЕННЯ АЛГОРИТМІВ ОПРАЦЮВАННЯ ВЕЛИКИХ ДАНИХ У БЛОКЧЕЙН

У розділі розроблено алгоритм визначення блоків-сиріт, алгоритм кодування-декодування даних в транзакції, алгоритм запису даних у блокчейн, а також метод перевірки якості внесених даних.

3.1. Розроблення алгоритму визначення блоків-сиріт

Однією з проблем, які виникають під час опрацювання даних у блокчейні, що значно знижує якість даних, є поява блоків-сиріт. Визначення якості даних у роботі наведено у підрозділі 3.4.

Блоки-сироти, часто застарілі блоки – це блоки, які не приймаються в мережу ланцюжка блоків через затримку в часі прийняття блоку, що розглядається у ланцюжку в порівнянні з іншим відповідним блоком. Блоки-сироти є дійсними та перевіреними блоками, але вони були відхилені ланцюжком. Їх також називають окремими блоками, оскільки вони існують ізольовано від ланцюжка блоків [60] (рис. 3.1).

Ключові моменти даного терміну та критерії таких блоків:

- Блок-сирота – це блок, який було вироблено в мережі ланцюжка блоків, але не прийнято через затримку в ній.
- Можуть бути два майнери, котрі шукають блок одночасно. Майнер, котрий знаходить блок швидше та затверджує його в мережі, отримує винагороду, а інший – блок-сироту.
- Немає винагороди за створення блоку, який у подальшому визначається як сирота.

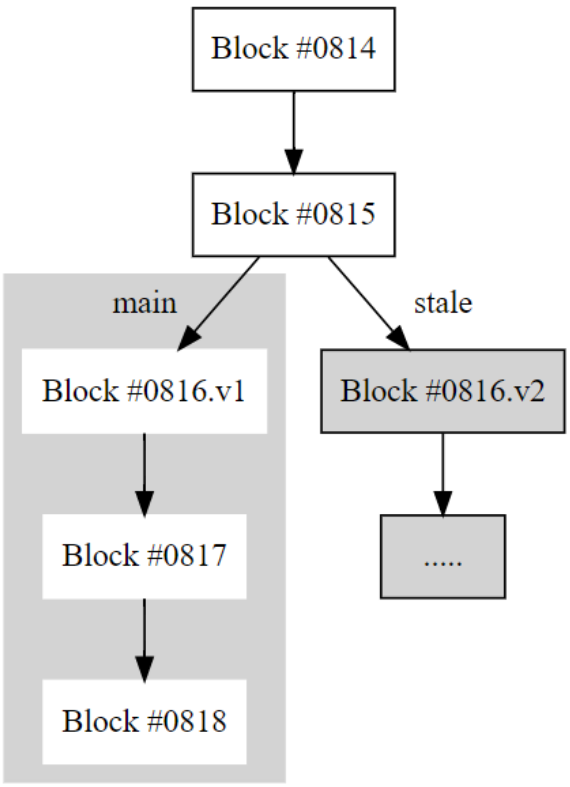


Рисунок 3.1 – Демонстація блоку-сироти

Блокчейн складається із ланцюга блоків, які діють як одиниці зберігання даних для фіксації деталей різних транзакцій, що відбуваються в мережі блокчейна. Під час стандартного процесу видобування видобувачі (майнери) намагаються генерувати нові блоки, вирішуючи складні математичні рівняння, необхідні для функціонування мережі блокчейна.

Перший видобувач, котрому вдається знайти новий блок, має право на винагороду за блок, і він записує першу транзакцію в новий знайдений блок. Щоб мережа ланцюжків блоків продовжувала функціонувати, знову виявлений блок додається як нова «одиниця» в ланцюжок блоків.

Однак можлива ситуація, коли два видобувачі виробляють блок в один і той же час. Вона виникає через те, що прийняття блоків блокчейн вузлами мережі не відбувається миттєво.

Через це затримка в прийнятті блоку може призвести до того, що інший видобувач знайде хеш того самого блоку швидше, що може призводити до тимчасової плутанини в мережі блокчейна, оскільки вузли намагаються вирішити, який блок із двох недавно ідентифікованих блоків вони хочуть прийняти.

У такій ситуації блок з більшою часткою підтвердження роботи (POW) приймається в блокчейн. Інший блок з меншим доказом роботи не додається в ланцюжок блоків і називається сирітським блоком. Такі блоки, по суті, є дійсними та перевіреними блоками, проте через робочий механізм мережі та час затримки, що в свою чергу призводить до затримки прийняття, один із блоків відхиляється, або стає сиротою.

Існують три фактори, що впливають на зниження ймовірності появи блоку-сироти:

Зменшення часу генерації блоку до 1 хвилини (зазвичай 10 хвилин).
Майнери для пошуку хешів нових блоків мають мати однаковий хешрейт.
Висока доступність вузлів для ретрансляції транзакцій в Mempool.

Тому, базуючись на цих правилах, у роботі розроблено метод визначення частоти втрати блоку.

Частота втрати блоків — це ймовірність того, що блок не буде частиною основного ланцюга. Система створює набір блоків довжиною |B| з підмножиною блоків як частиною головного ланцюга довжиною |M|. Таким чином, швидкість блокування сиріт можна визначити як:

$$O = \frac{|B| - |M|}{|B|}.$$

Для моделювання швидкості формування блоків сиріт використано марковський ланцюг. Фактором симуляції є час генерації блоку.

Позначаємо стан (0,0) початком симуляції. З іншого боку, якщо пул побудував k блоків на останньому «розгалуженому блоці», де він погодився зі спільнотою, а спільнота побудувала ℓ блоків за межами розгалуженого блоку, тоді ми позначаємо стан (k, ℓ). Враховуючи наявні механізми для вирішення невідповідностей, ми очікуємо, що стану (k, ℓ) для k і ℓ, що перевищує один чи два, матиме дуже низьку ймовірність появи.

Припускаємо, що пул відкриває нові блоки зі швидкістю λ1, тоді як решта спільноти робить це зі швидкістю λ2, з λ2>λ1. Тоді доцільно моделювати затримки зв'язку з експоненціальними випадковими змінними. Припустимо, що час, необхідний для повідомлення про відкриття блоку з пулу в спільноту і навпаки, експоненціально розподілений з параметром μ≫λ2.

Якщо система перебуває в стані (k, ℓ) з k≠ℓ, то вона повертається до стану (0,0) після встановлення зв’язку, оскільки тоді пул і спільнота домовляться про новий стан блокчейну. Однак, якщо k=ℓ≥1, тоді пул і спільнота мають різні, хоч однакові по довжині, версії блокчейну і продовжуватимуть майнінг на блокчейні, як вони бачать. Тому система залишається в стані (k,k) доти, доки не буде виявлено новий блок.

Модель Маркова має швидкості переходу:

$$q((k, \ell), (k+1, \ell))=\lambda 1,\ k\geq 0,\ \ell\geq 0 \qquad (3.1)$$

$$q((k, \ell), (k, \ell+1))=\lambda 2,\ k\geq 0,\ \ell\geq 0 \qquad (3.2)$$

$$q((k,\ell),(0,0))=\mu, k\neq \ell \qquad (3.3)$$

$$q((k, \ell), (k', l'))=0, \text{ інакше.} \qquad (3.4)$$

Перші два типи переходу, відображені в (3.1) і (3.2), відбуваються, коли пул (відповідно спільнота) майнить блок, тоді як третій, у (3.3), відбувається після встановлення зв’язку, коли ланцюжок знаходиться в стан (k, ℓ) з k≠ℓ. Ця остання швидкість є спрощенням того, що можна було б припустити: якщо

|k−ℓ|≥2, існує кілька завдань зв'язку, що виконуються, повідомляючи останні |k−ℓ| виявлення блоків у найдовшій гілці, і тільки коли надходить повідомлення про виявлення останнього блоку в найдовшій гілці, стан системи повертається до (0,0). Задля зручності в цій простій першій моделі це єдиний перехід, який ми врахували.

Як ми спостерігали вище, стани з |k−ℓ|≥2 мають дуже низьку ймовірність виникнення, і ми можемо очікувати, що ця модифікація не матиме великого впливу на стаціонарний розподіл.

3.2 Розроблення алгоритму кодування та декодування даних у транзакції

Для кодування інформації та декодування даних у транзакції буде використовуватись алгоритм перетворення будь-яких данних в hex значення за наступною функцією: strToHex(), код в додатку 5 (рис. 3.2).

Для декодування даних з транзакції буде використовуватись зворотній алгоритм для перетворення даних із hex у звичайний стрічковий варіант: hexToStr(), код у додатку 6 (рис. 3.3).

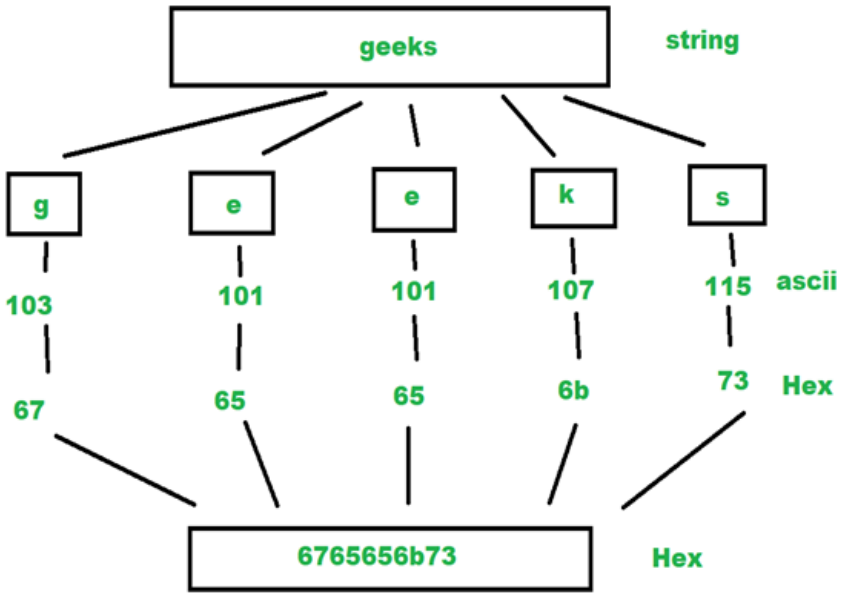


Рисунок 3.2 – Принцип переведення стрічки в шістнадцятковий формат

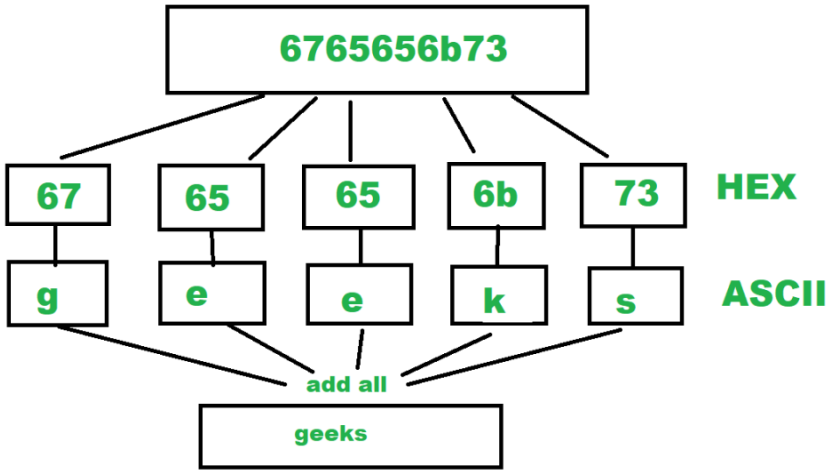


Рисунок 3.3 – Принцип шістнадцяткового формату в стрічку

За допомогою команди decoderawtransaction можна отримувати інформацію про вмістиме транзакції, яка була записана в блокчейн, наприклад:

decoderawtransaction
02000000016283218d44527c61383b45f96d80c592d9f871ea82fa1dd8933275e4f06745c501
0000006a47304402207954c7afbfb3da3f3792580483da407ca09c9a1f8f060e71cd6ae5b95f5
abde502202552b04f87b7f20bc01811eb49385d588838824ad06618e7a4cf302e12449a25012
1034e14fe1663a8ed2d068875a87aef4742155beccb4bd9279721e8494865c76b61ffffffff020
0e9a435000000001976a914154fda4a91130d498503e0493ada0170a0ad125388ac00000000
000000002a6a2834354541424135413441393346313142323131433736303042363245463446323238414130413133000000000

Отримуємо наступні дані про транзакцію з блокчейну, додаток 2.

Параметри:

- hash – хеш-транзакції в блокчейні.
- vin і vout – параметри для вхідних і вихідних даних транзакцій в ланцюзі блоків.

Дані в блокчейні знаходяться в розділі hex (метадані) в raw транзакції, якраз є, для прикладу в hex варіанті:

3335363334363336333347c32343438377c333234343337333133343332333333343333333433333333333333033323434333633373334343633343433333634333334334333353333533634323335333033363333333533353334333533333338303030303030303030303030303030303030307c356662366563326335653833313538376261666132323461396339346661353347c65373239373035383230636134376332333939636231343262376330643536663537306634373139 7c333434383833323 07c343836313133323 97c323433393035373635327c3136343139393234323 97c313634313938383832397c36312e31382e3231322e32380d0a

При звертанні на анонсер обробник на сервері задає запит до вузла блокчейну по заданому info_hash параметру і декодує всі транзакції, які були здійснені на адресу аналогічно info_hash в хеш варіанті. Таким чином, для роботи даної системи не потрібний один загальний координаційний центр для зберігання даних. Можна працювати з вузлами блокчейну, які самі між собою синхронізуються. Коли ми отримали дані за адресою, тоді буде згенеровано список транзакцій, у якому містяться дані про пірів поданого торрента, і їх можна швидко конвертувати в звичайний стрічковий варіант і матимемо:

3563463634|24487|32443731343233343334333233303244363734463443364334343535364235303633353534353338000000000000000000000|5fb6ec2c5e831587bafa224a9c94fa54|e729705820ca47c2399cb142b7c0d56f570f4719|34488320|48611329|2439057652|1641992429|1641988829|61.18.212.28

Отримавши ці дані, розбиваємо їх на масив через спеціальний символ | і видаємо як результат.

І якраз ці дані відправляються на scrape/announce по запиту від клієнта. Отже, у результаті отримуємо спрощення взаємозв’язку, тобто клієнт замість того, щоб задавати запити в базу даних, задає запити в блокчейн по своєму info_hash, з якого береться список пірів і відправляється йому. Таким чином,

вдалося кардинально *зменшити навантаження на базу даних* у кількість, пропорційну кількості серверів. При масштабуванні системи є *змога додавати вузли для блокчейну* і вузли для опрацювання даних.

3.3 Розроблення алгоритму запису даних у блокчейн

Метод запису даних у блокчейн побудовано наступний чином.

(1) Центральний вузол записує *BData*, що мають бути збережені, у локальне сховище, які необхідно зберегти на локальному носії даних, і рівномірно упаковує їх у заголовковий блок кожен період часу *P*.

(2) Припустимо, що *BData* записується протягом часу *P* як $BData_1$, $BData_2$, … $BData_n$. Спосіб об'єднання пакета в заголовковий блок полягає в наступному: блок записується як *BDATA*, а до складу блоку входять $BData_1$, $BData_2$, … $BData_n$.; кожен $BData_i$ включає метадані-*META*, модифіковані дані *BNEW* та адресу доступу до модифікованих даних *BLOCATION*, і метод запису заголовка може визначити, де *BData* було змінено, що він представляє собою після зміни.

(3) Розрахувати *DataHash=Hash* ($BData_1$ || $BData_2$ || … || $BData_n$); функція, що рахує хеш, містить SHAH1, SHAH256, MD5;

(4) Область записується як *BHEAD*, а композиція заголовка включає *PreviousHash*, *BHASH*, *LinkofDATA*, *Timestamp*, *BINDEX* і *Signature*. *PreviousHash* може гарантувати, що попередній блок не був змінений, *BHASH* може гарантувати, що *BDATA* не змінено, *Linkofdata* може знайти розташування *BDATA* та *Timestamp*, *Timestamp* — це мітка часу, яка встановлює заголовок блоку, який позначає часовий ряд блоку, *BINDEX* є глобальною індексною інформацією про блок, включаючи ключові слова, номер об’єкта та номер моделі, а *Signature* забезпечує повноваження блоку, який генерується центральним вузлом приватного блокчейну. *PreviousHash*

— це хеш-значення попереднього заголовку блоку, *LinkofDATA* — адреса доступу до блоку *BDATA, Timestamp* — це позначка часу для блоків, що будуються, а *Signature* — це підпис *PreviousHash*, *BHASH*, *LinkofDATA* та *Timestamp* центральним вузлом за допомогою нього закритий ключ, а саме *Sign*(*PreviousHash* || *BHASH* || *LinkOfDATA* || *Timestamp*).

На основі цього методу **розроблено алгоритм запису транзакцій**, що поданий нижче.

Алгоритм запису даних у блокчейн відбувається наступним чином.

1. Отримуємо адресу блокчейна, наприклад, 2Ng9J6jh95ZR8t9d6wqUE2sfWtwJo7TAcN. Створюється транзакція зі входом певної кількості монет.
2. За допомогою запиту до Blockchain RPC, отримуємо дані транзакції:


```
[
  {
    "txid": "e441450043c54f1d6e358f63cfbe618215f485ecb5feb851dfe2fbae902a4150",
    "vout": 1,
    "address": "2Ng9J6jh95ZR8t9d6wqUE2sfWtwJo7TAcN",
    "account": "",
    "scriptPubKey": "76a91469f91565dfabe1009d480611f2ea55cc1889ddde88ac",
    "amount": 8.78711729,
    "confirmations": 10,
    "spendable": true,
    "solvable": true
  }
]
```

3. Генеруємо спеціальну raw транзакцію, командою:

createrawtransaction
'[{"txid":"e441450043c54f1d6e358f63cfbe618215f485ecb5feb851dfe2fbae902a4150","vout":1}]'
'{"2DBLfPEDC88QxV4bnMzaVmXEd78YE5bSVX":1,"data":"44454641554c542e414243442e425a"}',

де вхідна транзакція має TX ID:

e441450043c54f1d6e358f63cfbe618215f485ecb5feb851dfe2fbae902a4150

4. Отримуємо адреси для наступної транзакції, оскільки блокчейн – це ланцюг з даними: 2DBLfPEDC88QxV4bnMzaVmXEd78YE5bSVX.
5. Кількість монет встановимо в 1.

Параметр vout має такі властивості:

- Tx складається з одного або декількох входів та одного або декількох виходів.
- Усі входи tx відносяться до невитраченого результату попередньої транзакції.
- Повна вартість введення завжди витрачається; tx не може витратити частину значення.
- Аналогічно, всі виходи витрачені і їх не можна частково витратити.
- Tx «проводить» виходи, на які посилаються у вхідній частині tx.
- Tx створює нові "невитрачені виходи", перераховані у вихідній частині tx.

HEX рядок: 44454641554c542e414243442e425a == DEFAULT.ABCD.BZ

Таким чином, маючи транзакцію, можна записати в неї будь-які дані, та записати назавжди в блокчейн.

6. Вихідні дані з команди createrawtransaction – ця стрічка є ідентифікатором для майбутнього цифрового підпису транзакції, яка буде відправлена в блокчейн:

010000000150412a90aefbe2df51b8feb5ec85f4158261becf638f356e1d4fc543004541e4010000000ffffffff0200e1f505000000001976a91401cd60975e1ca2d3a0583fc1c6d6b2a764ef408988ac00000000000000000116a0f44454641554c542e414243442e425a00000000

7. Робиться підпис транзакції командою через RPC API: signrawtransaction 010000000150412a90aefbe2df51b8feb5ec85f4158261becf638f356e1d4fc543004541e40100000000ffffffff0200e1f505000000001976a91401cd60975e1ca2d3a0583fc1c6d6b2a764ef408988ac0000000000000000116a0f44454641554c542e414243442e425a00000000
8. На виході отримуються такі дані, які вже є підписані:

{
"hex": "010000000150412a90aefbe2df51b8feb5ec85f4158261becf638f356e1d4fc543004541e4010000006a47304402203a889974c7f27c1a4faf1168509887a614e60327ed81623d5bb040bd079cb1ff022018e657b87a227bc27b2655750297f57222dac5683ff3a3cc0f1b451c93a20cea012102aef7fed90d530e790da2c7cf9638a0263aa7a1c72be5f786357bdee1c8e008a7ffffffff0200e1f505000000001976a91401cd60975e1ca2d3a0583fc1c6d6b2a764ef408988ac0000000000000000116a0f44454641554c542e414243442e425a00000000",
"complete": true
}

При помилці, транзакція просто не пройде, сам підпис не згенерується і буде мати наступний вигляд:

{
"hex": "0200000001fdfbd0831cdc9765606fcc7f7df4011396aac624e9a91776bc9f6e2a0b5f51d00100000000ffffffff0200e9a435000000001976a914154fda4a91130d498503e0493ada0170a0ad125388ac00000000000000002a6a2846373637454532344341334138343846323639454636353631453041323433313937344136394238000000000",
"complete": false,
"errors": [
{

"txid": "d0515f0b2a6e9fbc7617a9e924c6aa961301f47d7fcc6f606597dc1c83d0fbfd",
"vout": 1,
"witness": [
],
"scriptSig": "",
"sequence": 4294967295,
"error": "Input not found or already spent"
}
]
}

Input not found or already spent – означає помилка, коли виходи на транзакції вже були витрачені, тому дану транзакцію неможливо відправити в мережу.

9. Якщо транзакція успішно підписалась, відправляємо транзакцію в мережу:

sendrawtransaction

010000000150412a90aefbe2df51b8feb5ec85f4158261becf638f356e1d4fc543004541e4010000006a47304402203a889974c7f27c1a4faf1168509887a614e60327ed81623d5bb040bd079cb1ff022018e657b87a227bc27b2655750297f57222dac5683ff3a3cc0f1b451c93a20cea012102aef7fed90d530e790da2c7cf9638a0263aa7a1c72be5f786357bdee1c8e008a7ffffffff0200e1f505000000001976a91401cd60975e1ca2d3a0583fc1c6d6b2a764ef408988ac0000000000000000116a0f44454641554c542e414243442e425a00000000 true

Але можуть бути деякі нюанси, наприклад, обмеження по даних. Вони залежать від розміру блоку, таким чином, щоб входити в цей ліміт, дані треба ділити між транзакціями. Для поданої системи, було зроблено Blockchain, де розмір даних в 1 блоці сягає 10 MB.

3.4 Розроблення алгоритму перевірки якості внесених даних

Якість даних – узагальнене поняття, що відображає ступінь їх придатності для вирішення певного завдання. Відповідно до стандартів

основними критеріями якості є повнота, достовірність, точність, узгодженість, доступність та своєчасність.

Оцінка якості даних та дії щодо його підвищення є необхідним етапом будь-якого аналітичного проекту, оскільки аналітичні алгоритми або не зможуть працювати з неякісними даними, або будуть давати некоректні результати.

Приведення вихідних «сирих» даних у відповідність до необхідних критеріїв якості є найважливішим завданням Data Mining і утворює цілий напрямок, який називають передобробкою.

Основними проблемами, що викликають зниження якості даних, зазвичай вважають [61]:

- пропущені значення;
- дублікати;
- протиріччя;
- аномальні значення та викиди;
- шум;
- неповні даних;
- порушення цілісності даних;
- некоректні формати та подання даних;
- фіктивні значення;
- помилки введення даних;
- порушення структури.

У роботі проблемами, через які знижується якість даних, вважатимемо:

1. Дублікати.
2. Неповні дані.

3. Некоректні формати та подання даних (відрізняється від описаних в метаданих).
4. Фіктивні дані.
5. Суперечливі дані (козії хеш-функції).
6. Дані з порушеною структурою.

Профайлінг даних — один із найпоширеніших методів перевірки якості даних та виявлення проблем у Data Mining. Профайлінг виконується автоматично відповідно до деякого заздалегідь налаштованого сценарію на основі аналізу інформації про структуру даних [62].

У процесі профайлінгу перевіряються поля джерела даних на відповідність до заданих обмежень. Якщо параметри полів задовільняють обмеженням, дані вважаються відповідними необхідному рівню якості, в іншому випадку необхідно вживати заходів для приведення параметрів до відповідних обмежень.

Так як дані про пірів у даній дисертаційній роботі є динамічними з торрент-клієнтів, то перевірка їх якості полягає у перевірці співпадіння їх типів. Таким чином вирішується проблема:

- неспівпадіння форматів даних,
- неповних даних,
- порушення структури даних.

Якщо формат не зазначений у метаданих блоку або результат порожній, чи чогось не вистачає, наприклад, клієнт повернув через параметр port замість цифр випадкові знаки, то даний запис відхиляється і не додається в Blockchain [63].

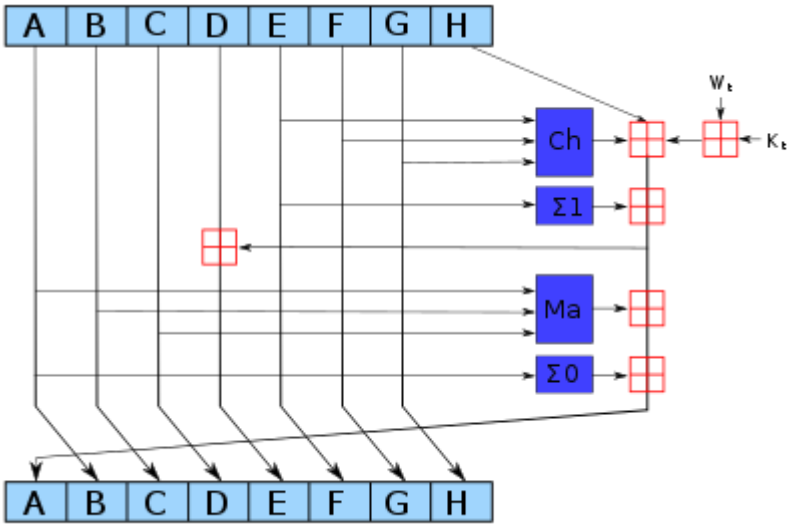
Таблиця 3.1 – Вигляд даних за їх типом

info_hash	sha1 hash
info_ip	integer
info_port	integer
info_peer	integer
info_md5	md5 hash (info_ip + info_hash + info_port + info_peer)
info_sha1	sha1 hash (info_ip + info_hash + info_port + info_peer)
info_upload	integer
info_download	integer
info_left	integer
info_update	tatestamp to timestamp
info_expire	integer
info_ipv46	string

Для цього типу даних важливі дані тільки info_hash, info_port, info_peer, а також info_ip. Для цілісності достатньо обчислити суму в рядок і взяти hash, загальний від них хеш у алгоритмі sha1 (рис. 3.4) і для контролю, щоб уникнути колізій, ще в алгоритмі md5 (рис. 3.5).

Таке подвійне хешування знижує ймовірність появи суперечливості даних (колізій хеш-функції).

Контрольна сума для перевірки цілістності та якості даних береться з параметрів: info_ip, info_port, info_peer, через те, що вони є статичними і відтворюють унікальність даних. Такі параметри, як-от info_upload, info_download, info_left, info_update, info_expire є динамічними.



Рисинок 3.4 – Ілюстрація sha256

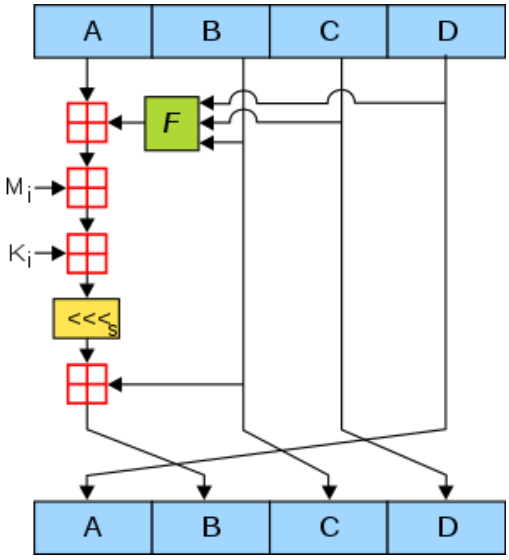


Рисунок 3.5 – Ілюстрація роботи md5

Результат sha256: f21d59ea795ef2ed705c30c3daf58b282e0286fa

Результат md5: b2648dac14b071010c0e8ecf75ac0e49

Вони створені для перевірки хешів тіла.

Обробник приймає ці дані, записує в тимчасову таблицю тимчасової бази даних SQL, знаходить їх хеш. Якщо хеш вже існує, тоді вони не записуються в блокчейн і просто викидаються для наступного запису. Також це стосується хеш даних порожній, тоді означає, що дані поступили браковані, тому їх не потрібно записувати. Це дає змогу вирішувати проблему дублікатів значень. Є особливо важливим для Великих даних, де одні і ті самі дані можуть надходити з різних джерел.

Отже, такі дані як info_upload, info_download, info_left, info_update, info_expire і info_ipv46 є чисто статистичні дані торрент клієнта, а основними виступають якраз IP, HASH торрента, порт клієнта і id піра з роздачі [65][66][67][68][69].

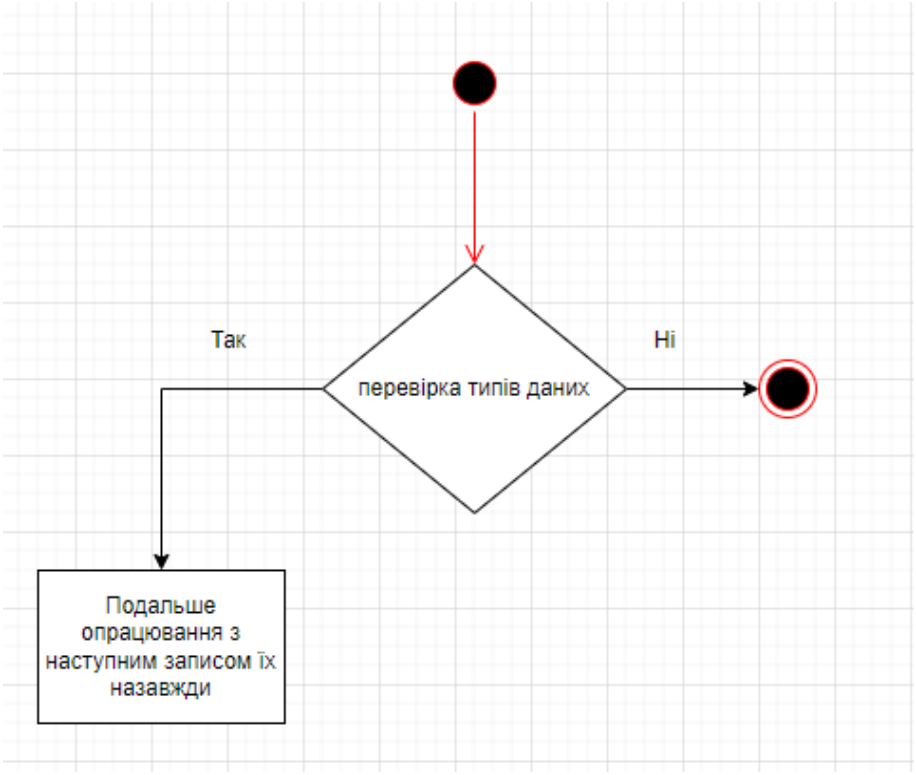


Рисунок 3.6 – Навантаження запитів на анонсер

Основна проблема таких систем – високонавантажуваність, і тому інколи дані є неповними [70][71][72]. Щоб такого уникнути треба також проводити перевірку на тип даних, він є статичним. Якщо тип не співпадає з

тим, що отримує обробник, то дані просто ігноруються і видаляються, навіть не записуються (рис. 3.6). Такий метод дозволяє уникнути проблем, коли навмисно хтось пробує записати неіснуючі дані чи сам клієнт відправляє неправдиві дані. Наприклад, коли скачане по факту одне і вираховується в integer форматі, а клієнт відправляє тип даних у форматі float з плаваючою крапкою.

3.5. Висновки

У даному розділі розробленено модель великих даних у блокчейн. Визначено основні операції над даними. Проаналізовано методи хешування. Розроблено метод визначення блоків-сиріт на основі аналізу часових міток та марковської моделі шляхом додавання таких блоків як нової одиниці у ланцюжку блоків та врахування часу затримки у прийнятті блоків вузлами. Це дає змогу забезпечити цілісність даних та усунути надмірність у них.

Розроблено алгоритм кодування та декодування даних у транзакції для наступного запису їх у блокчейн.

Розроблений алгоритм перевірки якості внесених даних, який базується на аналізі наявності дублікатів, суперечливих даних та даних з пропусками. Це дало змогу зменшити кількість запитів на анонсер майже удвічі. Також це дає змогу усунути проблему з keep-timeout та знизити можливість втрати даних.

Результати розділу опубліковано у працях автора [7, 8].

РОЗДІЛ 4. РОЗРОБЛЕННЯ АРХІТЕКТУРИ ТА АПРОБАЦІЯ РЕЗУЛЬТАТІВ

У розділі розроблено архітектуру системи та подано її у вигляді діаграми класів. Розроблено протокол обміну даними. Здійснено апробацію результатів та визначено кількісні параметри розробленої інформаційної системи. Розроблено архітектуру Великих даних у блокчейн для медицини, яка використана в інформаційній системі перевірки ліків, що впроваджена в лікарні швидкої допомоги м. Львова

4.1 Розроблення архітектури системи

Архітектура системи побудована з декількох модулів:

- Модуля запису в блокчейн, модуля читання з блокчейну,
- Модуля перевірки інформації та очистка даних від повторів,
- Модуля для тимчасових даних та їх опрацювання тощо.

Особливість такої архітектури полягає в тому, що вона ділиться на основні модулі для опрацювання даних у блокчейні. Тому, в принципі, дану систему можна використовувати для різних проектів з будь-якими даними.

Для спрощення побудови проекту архітектуру було поділено на 6 модулів:

- Announce – модуль для обробки даних, які поступають від клієнта.
- Scrape – модуль для віддачі даних клієнту.
- Tracker – модуль для опрацювання тимчасових вхідних даних.
- Sort and Clear – основний операційний модуль для опрацювання вхідних і вихідних даних, сам модуль містить у собі різноманітну кількість функцій для роботи системи.
- BigData – модуль для опрацювання великих даних, а саме запису та читання їх.

- Blockchain – модуль для реплікації по нодах, побудований на основі блокчейну з алгоритмом Scrypt, де ідентифікатором даних виступає адреса.

На рисунку 4.1 приведена діаграма класів, що відповідає загальному вигляду архітектури системи. Основні модулі обведені червоним.

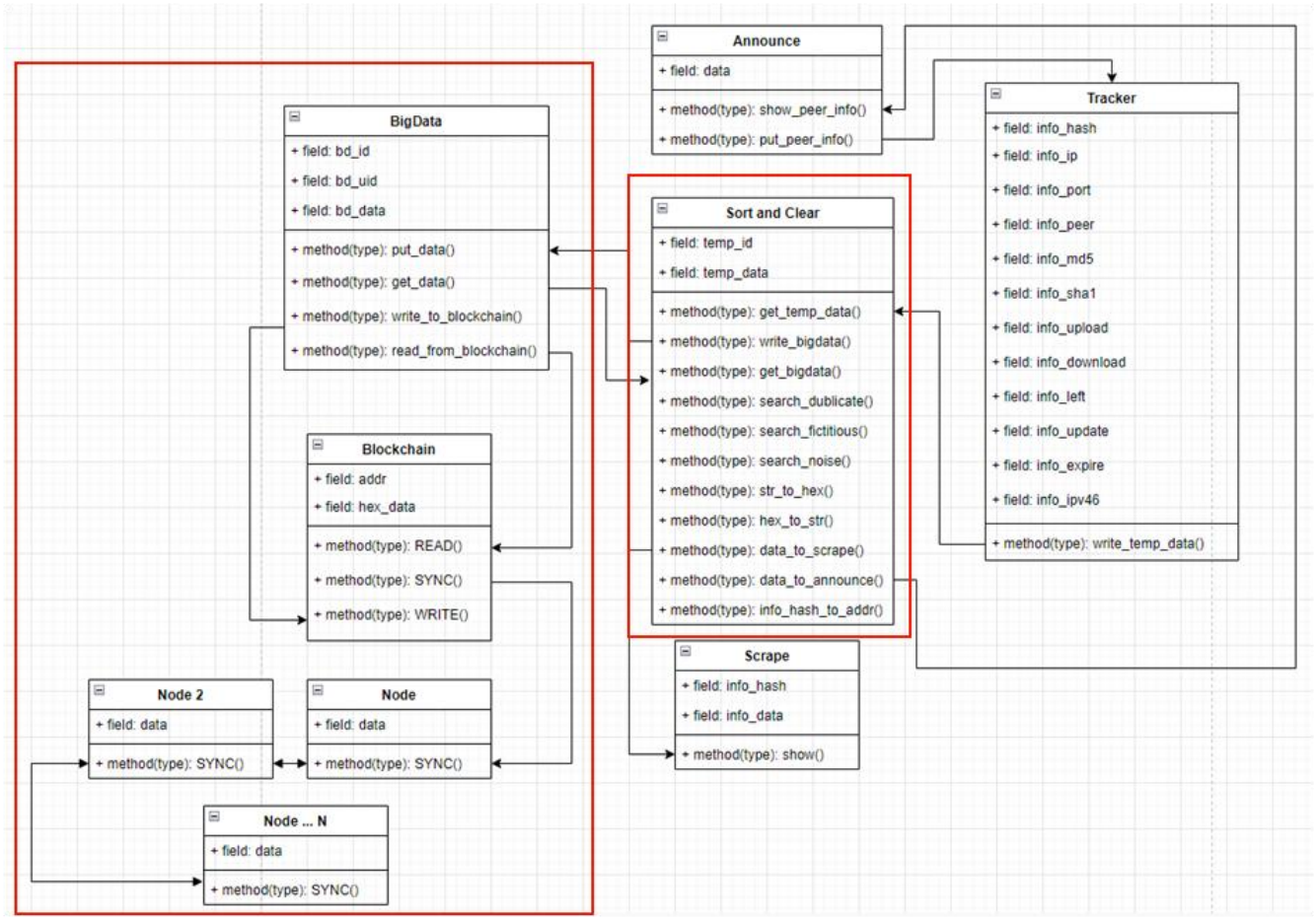


Рисунок 4.1 – Діаграма класів архітектури модулів

Таблиця 4.1 Атрибути класу «Announce»

Атрибут	**Опис**
data	Вхідні дані, які поступають на анонсер

Таблиця 4.2 Методи класу «Announce»

Метод	**Опис**
show_peer_info	Метод видобутку даних та демонстрація його клієнту
put_peer_info	Запис всіх вхідних даних в Tracker

Таблиця 4.3 Атрибути класу «Tracker»

Атрибут	**Опис**
info_hash	Інформація про хеш-ключ, є ідентифікатором
info_ip	Інформація про IP клієнта у числовому форматі
info_port	Інформація про порт клієнта
info_peer	Інформація про ID піра клієнта
info_md5	Хеш-сума md5 для контроля цілісності даних
info_sha1	Хеш-сума sha1 для контроля цілісності даних
info_upload	Інформація про роздане
info_download	Інформація про скачане
info_left	Інформація про те, скільки залишилось скачати
info_update	Часова мітка останнього оновлення піра
info_expire	Часова мітка неактуальності піра
info_ipv46	IP адреса клієнта у вигляді IPv4, IPv6

Таблиця 4.4 Методи класу «Tracker»

Метод	**Опис**
write_temp_data()	Запис даних у тимчасову базу даних з подальшим опрацюванням їх головним модулем системи

Особливість даної системи, що класи Tracker, Announce і Scrape є прикладом вхідних та вихідних даних. Основними модулями системи є *Sort and Clear, BigData, Blockchain.*

Таблиця 4.5 Атрибути класу «Sort and Clear»

Атрибут	**Опис**
temp_id	Ідентифікатор даних
temp_data	Безпосередньо дані, що можуть бути в різних форматах

Таблиця 4.6 Методи класу «Sort and Clear»

Метод	**Опис**
get_temp_data()	Видобуток тимчасових даних з бази даних
write_bigdata()	Запис даних у тимчасову таблицю великих даних
get_bigdata()	Читання даних із тимчасової таблиці великих даних
search_dublicate()	Пошук та очистка дублікатів даних
search_fictitious()	Пошук та очистка даних від фіктивних даних
search_noise()	Пошук та очистка даних від шуму
str_to_hex()	Перетворення стрічкових даних в hex вигляд
hex_to_str()	Перетворення даних з hex вигляду в стрічкові дані
data_to_scrape()	Перетворення даних в bencode дані про сідів
data_to_announce()	Перетворення даних в bencode дані про пірів
info_hash_to_addr()	Перетворення info_hash в адресу на блокчейн мережі

Таблиця 4.7 Атрибути класу «BigData»

Атрибут	**Опис**
bd_id	Ідентифікатор для тимчасової таблиці
bd_uid	Ідентифікатор uid, що дорівнює адресу в блокчейні
bd_data	Дані в hex форматі

Таблиця 4.8 Методи класу «BigData»

Метод	**Опис**
put_data()	Запис даних
get_data()	Видобування даних
write_to_blockchain()	Запис великих даних у блокчейн через RPC API
read_from_blockchain()	Читання великих даних із блокчейну через RPC API

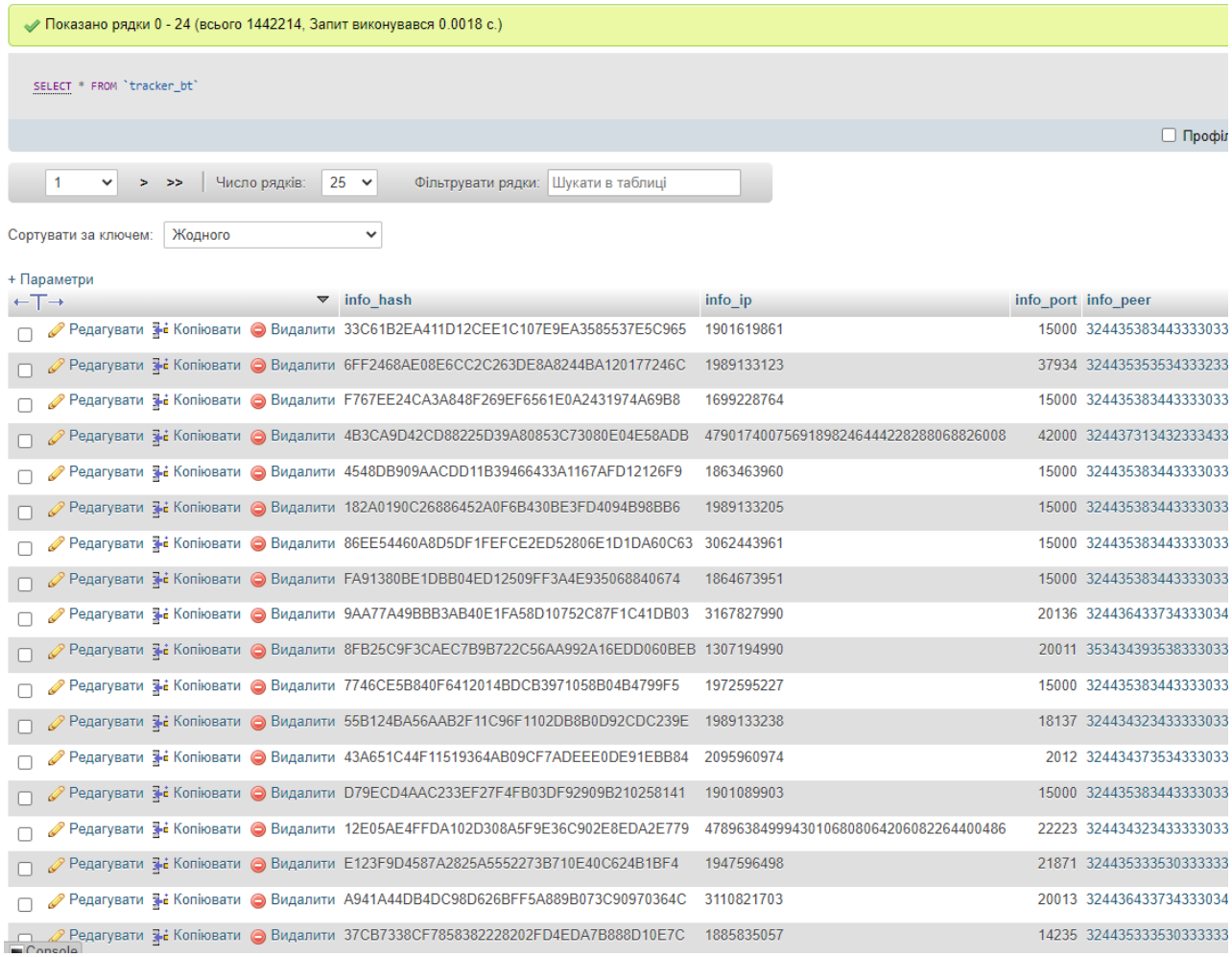
Таблиця 4.9 Атрибути класу «Blockchain»

Атрибут	**Опис**
addr	Ідентифікатор даних у блокчейні
hex_data	Великі дані у форматі hex

Таблиця 4.10 Методи класу «Blockchain»

Метод	**Опис**
READ()	Читання даних у блокчейн, де ідентифікатор адреса
WRITE()	Запис даних у блокчейн, де ідентифікатор адреса
SYNC()	Синхронізація даних з різними вузлами блокчейну

На рисунку 4.2 представлені дані, які чекають опрацювання та їх наступний запис у блокчейн.


Показано рядки 0 - 24 (всього 1442214, Запит виконувався 0.0018 с.)

SELECT * FROM `tracker_bt`

Профіл

1 > >> Число рядків: 25 Фільтрувати рядки: Шукати в таблиці

Сортувати за ключем: Жодного

+ Параметри

	info_hash	info_ip	info_port	info_peer
Редагувати Копіювати Видалити	33C61B2EA411D12CEE1C107E9EA3585537E5C965	1901619861	15000	324435383443333033
Редагувати Копіювати Видалити	6FF2468AE08E6CC2C263DE8A8244BA120177246C	1989133123	37934	324435353534333233
Редагувати Копіювати Видалити	F767EE24CA3A848F269EF6561E0A2431974A69B8	1699228764	15000	324435383443333033
Редагувати Копіювати Видалити	4B3CA9D42CD88225D39A80853C73080E04E58ADB	47901740075691898246444228288068826008	42000	324437313432333433
Редагувати Копіювати Видалити	4548DB909AACDD11B39466433A1167AFD12126F9	1863463960	15000	324435383443333033
Редагувати Копіювати Видалити	182A0190C26886452A0F6B430BE3FD4094B98BB6	1989133205	15000	324435383443333033
Редагувати Копіювати Видалити	86EE54460A8D5DF1FEFCE2ED52806E1D1DA60C63	3062443961	15000	324435383443333033
Редагувати Копіювати Видалити	FA91380BE1DBB04ED12509FF3A4E935068840674	1864673951	15000	324435383443333033
Редагувати Копіювати Видалити	9AA77A49BBB3AB40E1FA58D10752C87F1C41DB03	3167827990	20136	324436433734333034
Редагувати Копіювати Видалити	8FB25C9F3CAEC7B9B722C56AA992A16EDD060BEB	1307194990	20011	353434393538333033
Редагувати Копіювати Видалити	7746CE5B840F6412014BDCB3971058B04B4799F5	1972595227	15000	324435383443333033
Редагувати Копіювати Видалити	55B124BA56AAB2F11C96F1102DB8B0D92CDC239E	1989133238	18137	324434323433333033
Редагувати Копіювати Видалити	43A651C44F11519364AB09CF7ADEEE0DE91EBB84	2095960974	2012	324434373534333033
Редагувати Копіювати Видалити	D79ECD4AAC233EF27F4FB03DF92909B210258141	1901089903	15000	324435383443333033
Редагувати Копіювати Видалити	12E05AE4FFDA102D308A5F9E36C902E8EDA2E779	47896384999430106808064206082264400486	22223	324434323433333033
Редагувати Копіювати Видалити	E123F9D4587A2825A5552273B710E40C624B1BF4	1947596498	21871	324435333530333333
Редагувати Копіювати Видалити	A941A44DB4DC98D626BFF5A889B073C90970364C	3110821703	20013	324436433734333034
Редагувати Копіювати Видалити	37CB7338CF7858382228202FD4EDA7B888D10E7C	1885835057	14235	324435333530333333

Console

Рисунок 4.2 – Невідсортовані дані, отримані з анонсера за п’ять хвилин роботи системи, 1442214 записів.

Перетворення info_hash ідентифікатора в адресу на мережі блокчейн, де будуть відбуватись транзакції даних.

Таблиця 4.11 Генерація ідентифікатора в Blockchain мережі

info_hash ідентифікатор	33C61B2EA411D12CEE1C107E9EA3585537E5C965
Адреса на Blockchain	2ExVocnskqgHXHmPSVZRBk5LWb8NsCXmf9
Закритий ключ для доступа до адреси	5JyQFzBdYyg6LF9xdNkC59nC6FPJ8PJSQPRvVn8on946YASVr6j

На рис. 4.3 представлені дані, які вже є записані в Blockchain та синхронізовані між нодами з 2-ма підтвердженнями, де OP_RETURN якраз є хешовані великі дані.

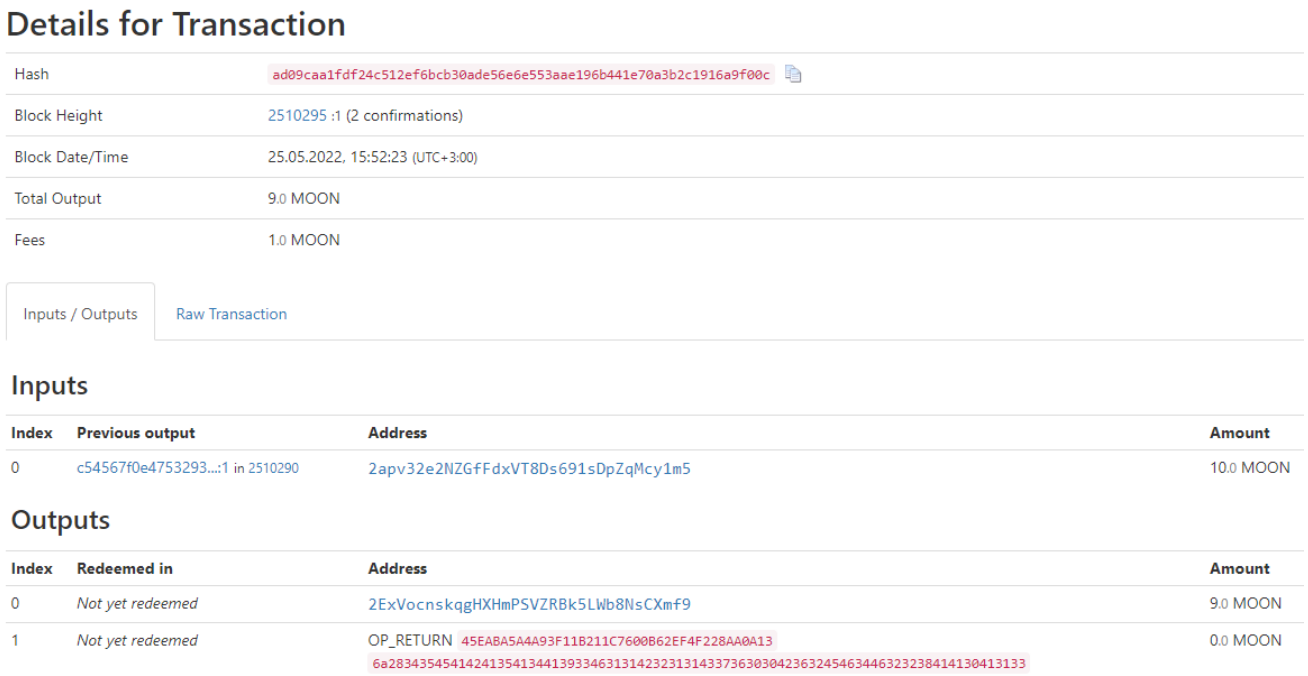


Рисунок 4.3 – Транзакція, яка вже назавжди записана в Blockchain.

На рисунку 4.4 представлено адресу 2ExVocnskqgHXHmPSVZRBk5LWb8NsCXmf9, яка зазначена рівною ідентифікатору: 33C61B2EA411D12CEE1C107E9EA3585537E5C965, де записана інформація про кількість пірів.

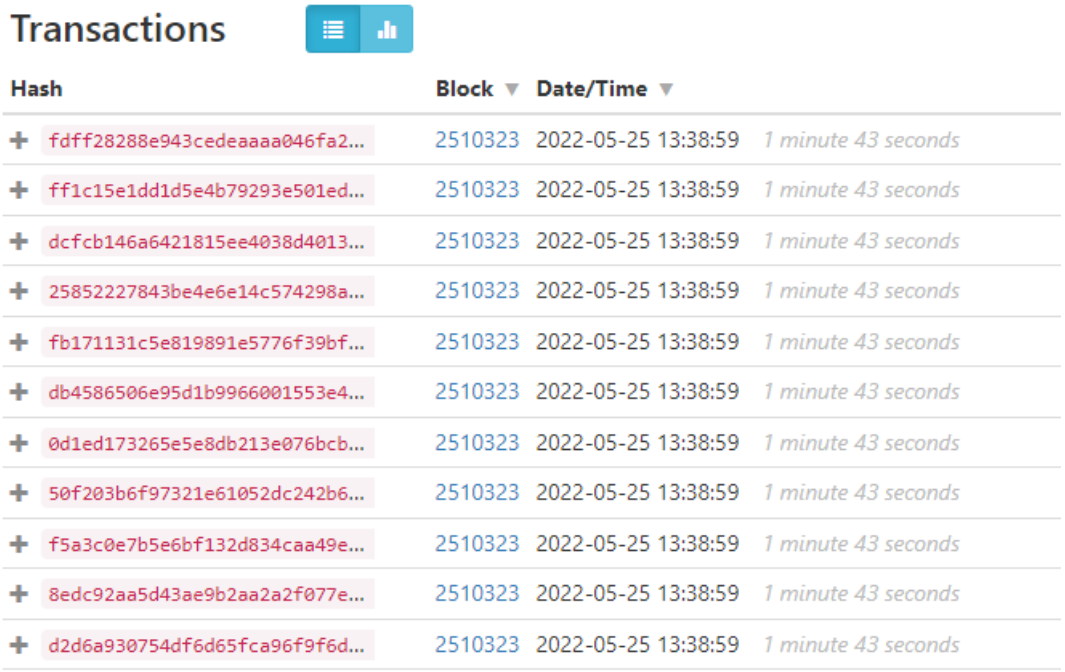


Рисунок 4.4 – Список транзакцій, які містять в собі інформацію про пірів.

Характеристика блокчейна наступна:

- розмір блоку 10 МБ;
- алгоритм блокчейну: scrypt;
- швидкість знаходження нових блоків 1 хвилина.

Блокчейн у даній системі побудований за таким принципом: є seed адреси, на яку за допомогою майнінгу приходять монети, приклад на рисунку 4.6. Для майнінга, використано три сервери конфігурації: AMD Ryzen 9 5950X [16c / 32t] (3.4 GHz) / 128 GB DDR4 ECC RAM / 2 x 3.84TB NVMe, так як в даному блокчейні низька складність. Дані монети марковані, як монети з 1 виходом, їх можно використовувати для запису транзакцій в блоках. Самі блоки встановлені рівні розміру 10 MB, що дозволяє записувати багато даних в один блок. Щоб отримати ці дані, треба мати закритий ключ адреси. Так як він наявний і рівний info_hash, який закодований однобічним хешуванням sha256, система генерує як відкритий, так і закритий ключ, за допомогою яких можна

одержувати дуже швидко дані з транзакцій, так як дана система направлена на швидкодію.

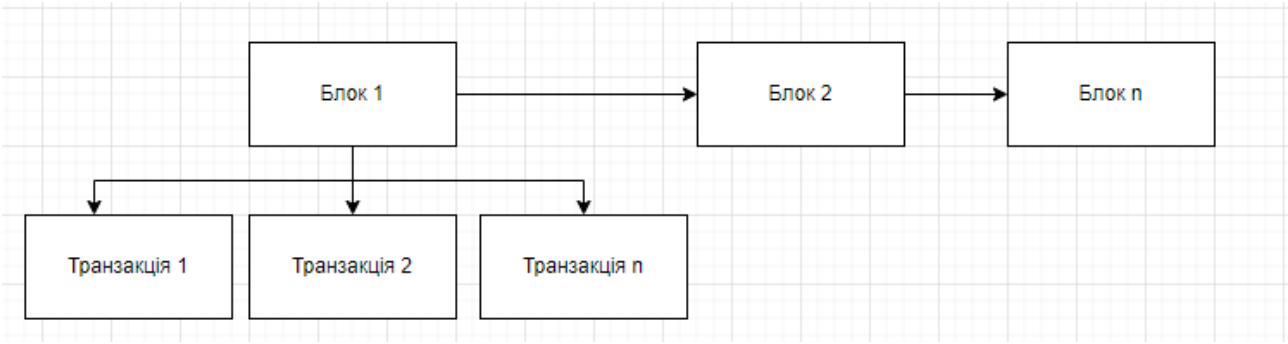


Рисунок 4.5 – Структура блокчейну в системі.

Timestamp	Block	Hash	Amount (YTN)	Balance (YTN)	TX Type
2022-05-20 08:29	1167164	321b08aa1dd6d6dd3115b02eafdf84b7380aad9b97d2eba4425403e9ed236d2c	25.01607251	201311.25769456	
2022-05-20 02:09	1166985	c72598af9b9ba5102418a848c9336bf2eaf0550c547087212f577204e8e9fb52	25.00000000	201286.24162205	
2022-05-20 01:29	1166962	354caaefddff0ddc42f5bb3a158f802b7cabe6322fe4845ff21094def4cdf4f5	25.00000000	201261.24162205	
2022-05-19 23:22	1166901	c7fae9b5b20cdc063ff173168e773667319aa8a2a8f1213dc357447f3f114448	25.00000000	201236.24162205	
2022-05-19 22:53	1166882	9e1ca330ea497af33ac3f2910a96211f50c79c34177a4734195b31d27d4e41af	25.00000000	201211.24162205	
2022-05-19 19:56	1166802	fac77ec015b5c0d8e9d41f0e86f51d9847193c7a3fde72022ac8fa163ae99d5f	25.00000000	201186.24162205	
2022-05-19 10:23	1166520	78760b2aa432992b4525fa6571e9d5b3510338695ecc3471a243e677e7e170b1	25.00000000	201161.24162205	
2022-05-19 09:23	1166493	4a7d657c1a8347147676927cd2d260cfa047c44c84653a83f1a1b1ea0366a4d4	25.10000000	201136.24162205	
2022-05-19 06:43	1166420	450a97705d97416b054b052dca534474985156f6b80f95a72e3b02189f1eee1a	25.00000000	201111.14162205	
2022-05-19 04:43	1166351	2c07f21f9ded7cb796f278094a9841134f3cb7c2284664812a29b3d731df4877	25.00000000	201086.14162205	
2022-05-19 03:59	1166331	04e9d5ed155f627ff75452cbc30cc179f5acac0cf2c9b5302ed37f557823f933	25.00000000	201061.14162205	
2022-05-19 03:35	1166319	a964a676957ff56c66915ebeaee431bf520385f7dd96c19c81496e99c74249ab	25.00000000	201036.14162205	
2022-05-19 01:31	1166260	67b20a4da6ec6c7b2ed675c89322e0e287415a7e7e15c7af95443b998e309fe4	25.10000000	201011.14162205	
2022-05-18 22:54	1166185	8854cab1cefb49f3a85f897eb66e5f1d46a3b14cdab2da722066b37b1faabd6b	25.00000000	200986.04162205	
2022-05-18 20:24	1166107	1816efc60ee8ee9e8eaa8c4c0e132008827010722e68e2e1952d4f1244bf6216	25.00000000	200961.04162205	
2022-05-18 19:34	1166082	0b635cf0c01ac744b6b8656bc253742096e8d5f9be8bf12c35efc8c537099ae6	25.14468830	200936.04162205	
2022-05-18 19:26	1166076	d644144a79bdb038789765abe5fc068afb30628003b6f8ea8da3c17350e05382	25.08469991	200910.89693375	
2022-05-18 18:35	1166051	ee37652916c5b634aabac93fd19638dca4ca85124f2212d5750ca456c80d8630	25.00000000	200885.81223384	

Рисунок 4.6 – Генерація монет за допомогою майнінгу для наступних транзакцій у блокчейн на seed адресу

4.2. Розроблення протоколів обміну даними

Для обміну даних базовий протокол буде використовувати базовий протокол прикладного рівня HTTPS [73][74][75]. Для отримання даних буде використовуватись REST API через метод GET з випадковим ключем для забезпечення конфіденційності передачі даних [76][77][78][79]. Структура URL адреса буде наступною http://1337.abcvg.info/announce – для прийому і віддачі даних з торрент клієнта, а також: http://1337.abcvg.info/scrape для отримання додаткових даних на рахунок пірів. Схема роботи такої системи, зображена на рисунку 4.7.

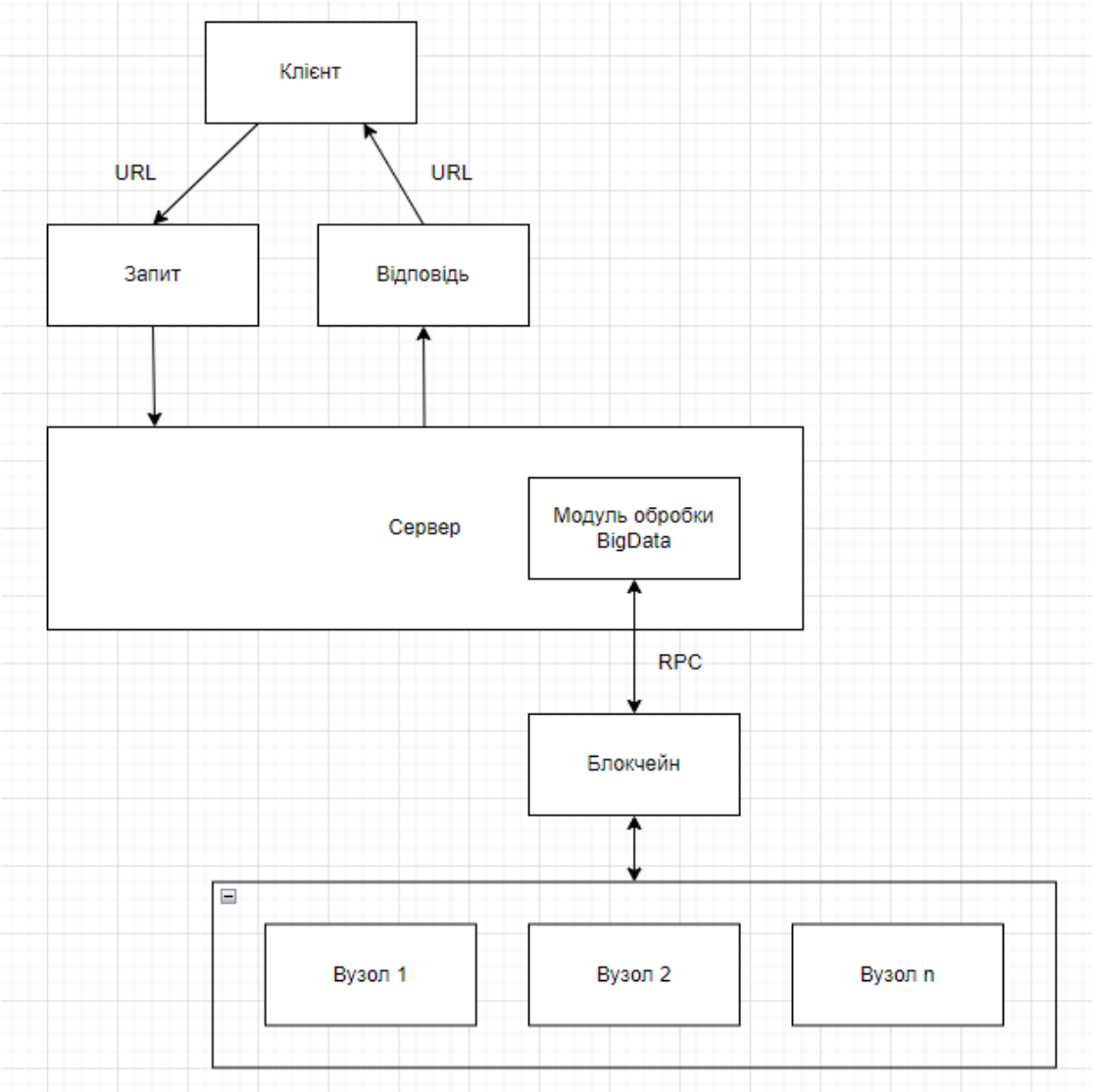


Рисунок 4.7 – Схема роботи протоколів обміну даних

Таблиця 4.12 – Схема роботи всього проекту

Клієнт	Торрент-клієнт, який передає дані на сервер
Запит	REST API запити
Відповідь	REST API відповіді
Сервер	Сервер для опрацювання всіх даних
Модуль обробки BigData	Модуль для опрацювання даних
Блокчейн	Блокчейн використовується як реплікаційна ДБ

Система розроблена на мові програмування PHP, C++ з використанням бази даних MySQL для запису тимчасових даних та Blockchain системою, на основі алгоритму scrypt для запису постійних незмінних даних, які дублюються між вузлами, серверами для майнінгу і створення емісії монет для майбутніх транзакцій.

4.3. Розроблення архітектури медичної системи з використання блокчейн-технології

Дану систему можна застосувати для будь-яких проектів з опрацюванням великих даних з блокчейном. Сам блокчейн і модулі розроблені для опрацювання будь-яких даних, так як їхні основні функції полягатимуть у підвищенні якості даних, а саме очищення від дублів та зменшення неповноти, а також, швидкий запис та швидке читання даних.

Для прикладу, демонстрація роботи даної системи з даними: пацієнт і його хвороби.

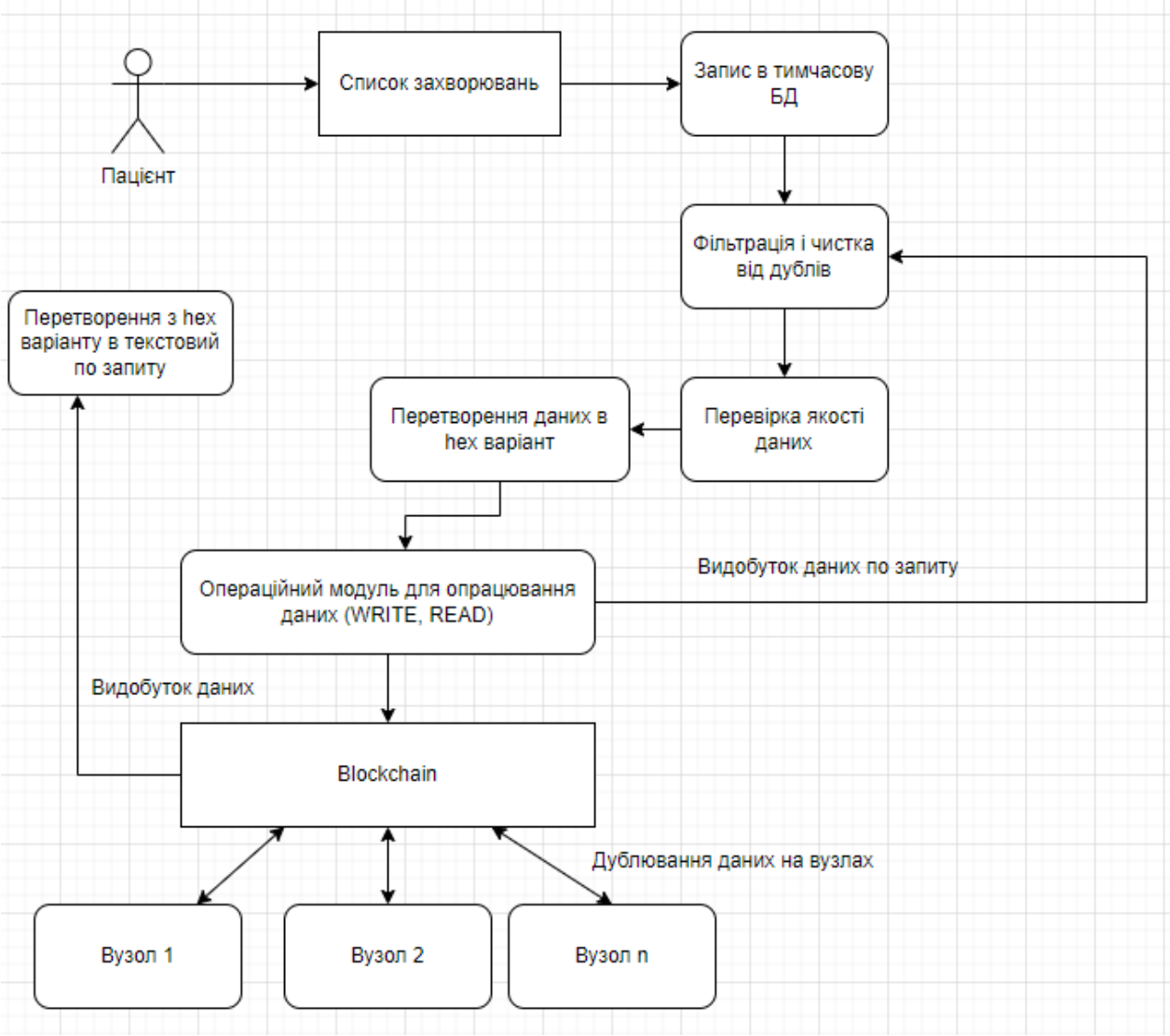


Рисунок 4.8 – Схема роботи системи з медичними даними

На рис. 4.8 представлена блок-схема роботи такої системи. Сама система працює, як заміна реляційній базі даних MySQL. Стандартні бази даних не справляються з великим обсягом незмінних даних для швидкої роботи. Представлена система ідеально підійде, наприклад, для ведення історії захворювань, щоб записати раз і назавжди, і не було можливості змінити чи видалити дані.

Приклад медичної системи зображено на ER діаграмі на рис. 4.9 з використанням Blockchain технологій для запису і збереження даних про

прийом ліків, які призначені лікарями для пацієнтів. Таким методом можна забезпечити надійність даних і використовувати блокчейн для інших даних, як запасний варінт на випадок збоїв головної бази даних.

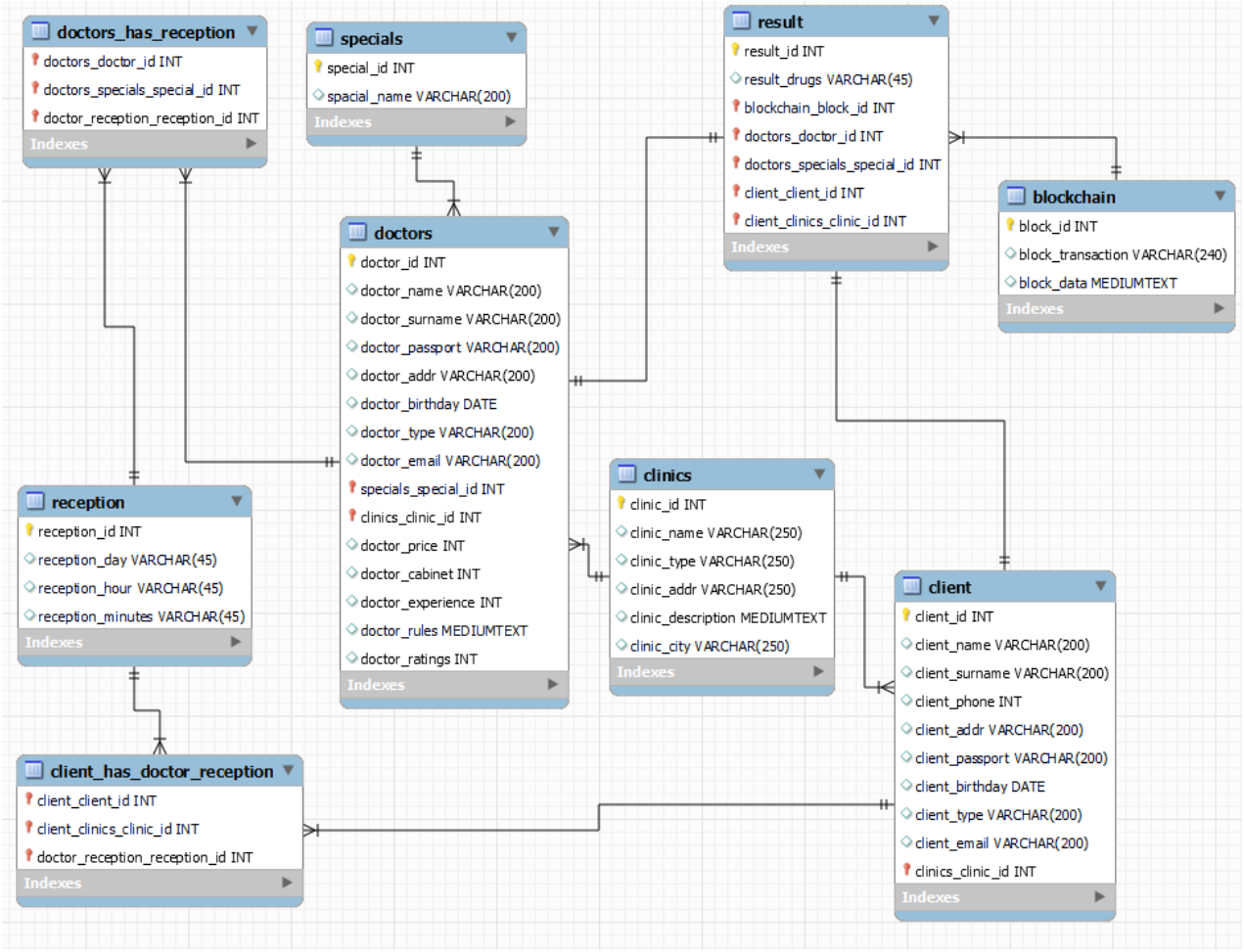


Рисунок 4.9 – ER-діаграма медичної системи

Структура таблиць бази даних медичної системи:

- Clinics – список клінік, які зареєстровані в системі.
- Doctors – список лікарів, які прив’язані до системи.
- Special – лікарські спеціальності.
- Reception – розклад прийому лікарів.
- Clients – відвідувачі клінік і лікарів.
- Result – інформація про висновок від лікарів про прийом ліків.

- Blockchain – спеціальна тимчасова таблиця даних для наступного запису через обробник даних у блокчейн.

Працює все аналогічно, як і для торрент-трекера, продемонстрованого в дисертаційній роботі, тільки замість даних про пірів маємо дані про прийом ліків.

4.4. Апробація результатів

Одною з кращих демонстрацій роботи даної системи є експлуатація у великонавантажувальних проектах з демонстрацією того, як в рази падає навантаження на сервер у випадку збереження постійних даних в Blockchain, а тимчасових даних в базах даних MySQL у порівнянні з використанням бази даних MySQL для всіх даних.

Основна суть роботи полягала у розробці системи, яка може опрацьовувати і бути практично цінною для великих даних і блокчейна.

Для прикладу обрано торрент анонсер, що містив мільйони записів, і для його оптимізації розроблено систему на основі блокчейну, щоб зменшувалося навантаження при величезній кількості запитів на сервер. При використанні класичної реляційної бази даних MySQL маємо проблему, а саме: оператори INSERT, SELECT при величезній кільності запитів в секунду не справлялися з обробкою даних, що створювало затримку і чергу у вибірці чи додаванні даних, що в свою чергу приводило до збільшення навантаження на сервер [80][81][82][83][84], див. рисунок 4.11. Для вирішення зазначеної проблеми та зменшення навантаження на сервер було спроектовано систему і обробник для опрацювання даних у блокчейні.

Схему роботи частини системи, коли при величезній кількості запитів, зменшується навантаження між серверами, зображено на рисунку 4.10, та безпосередньо результат зменшення навантаження продемонстрований на рисунку 4.12.

Такий підхід використання блокчейну дозволяє кратно зменшити навантаження для тих галузей, де є високе навантаження і стандартні бази даних не справляються, або справляються, але провокують високе навантаження на сервери і втрату даних. У такому підході немає єдиного центра, як у звичайних базах даних, кожен вузол – це є своя база даних [85][86][87][88]. Задаючи запит до вузла, не потрібно використовувати централізовані бази даних, так як за допомогою закритих ключів можна записувати та читати будь-які дані, що записані у блокчейн.

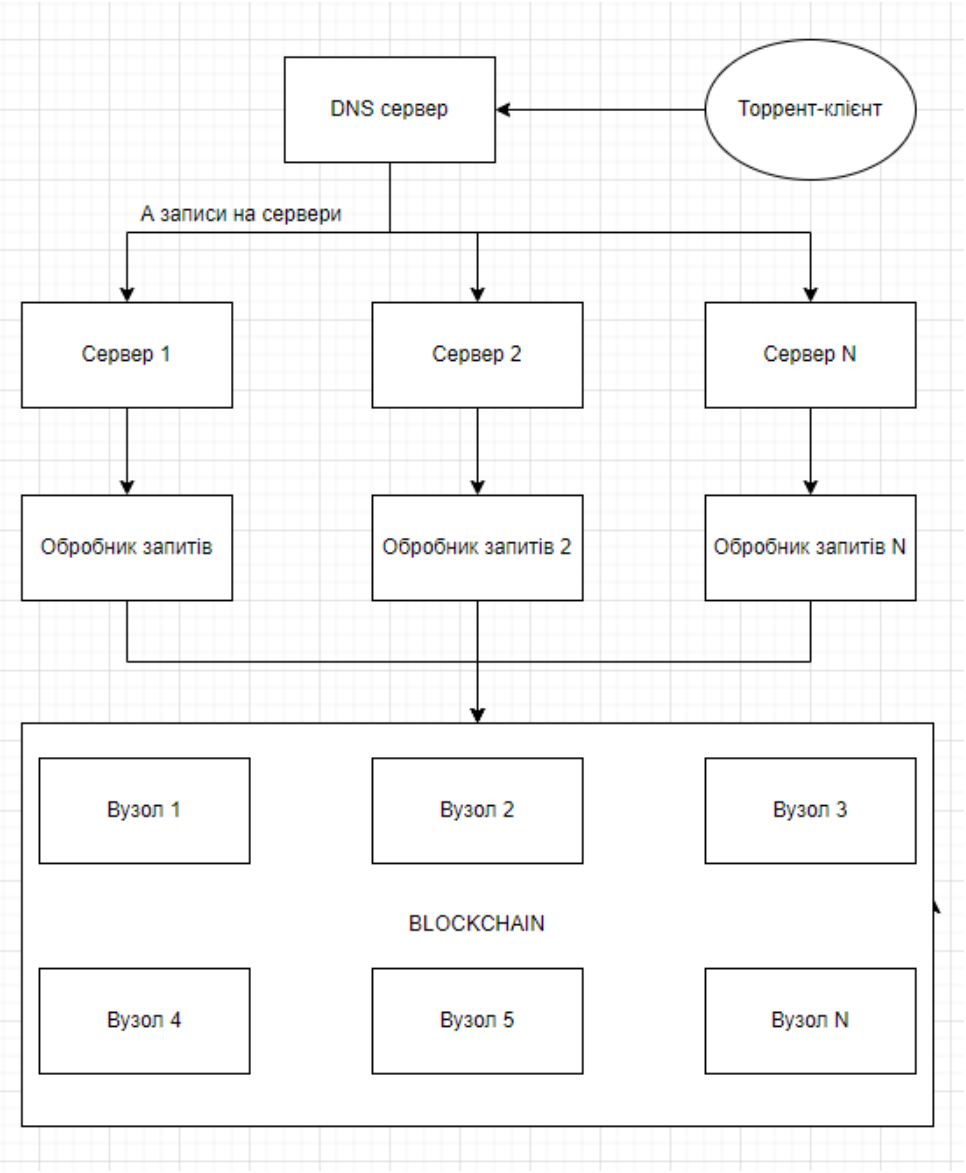


Рисунок 4.10 – Схема роботи системи

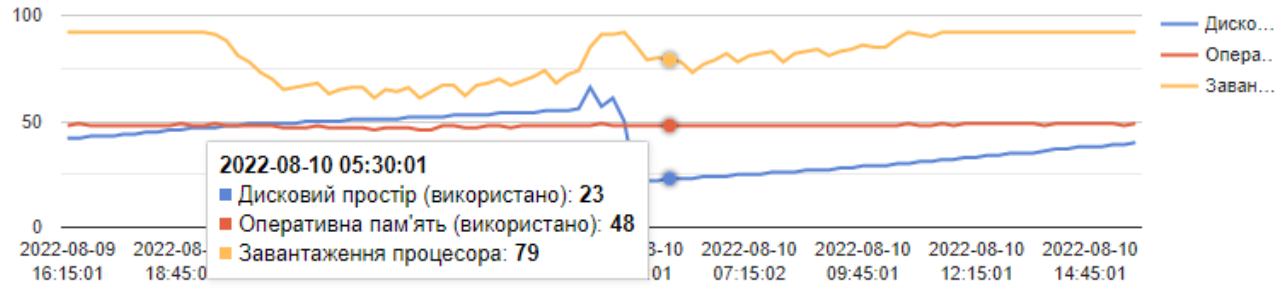


Рисунок 4.11 – Навантаження на сервер, до введення в експлуатацію

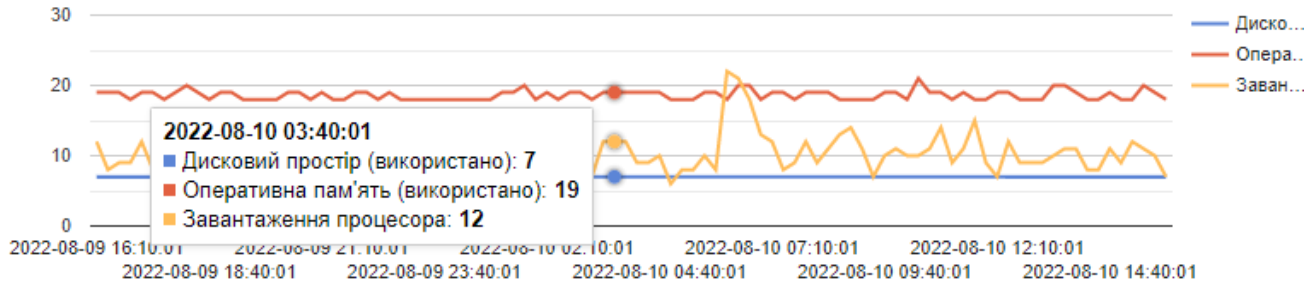


Рисунок 4.12 – Навантаження на сервер, після введення в експлуатацію

Демонстрація переключення анонсера по принципу, коли роль бази даних виконує MySQL (зображено на рисунку 4.11) та Blockchain (зображено на рисунку 4.12). Різниця очевидна, також видно, як кардинально спадає навантаженість на сервер (втричі).

Розроблену систему можна використовувати, як конструктор для оптимізації даних, які є незмінними, наприклад логи для статистики тощо. Як ідентифікатор може виступати адреса (відкритий ключ), а транзакції – хешовані мета-дані.

root@brown:~

```
top - 21:01:57 up 646 days, 14:39,  1 user,  load average: 9.66, 10.83, 10.59
Tasks: 219 total,   3 running, 214 sleeping,   0 stopped,   2 zombie
%Cpu(s): 46.0 us, 16.7 sy,  0.0 ni, 35.7 id,  0.0 wa,  0.0 hi,  1.6 si,  0.0 st
KiB Mem : 16141720 total,  1211456 free,  5948180 used,  8982084 buff/cache
KiB Swap:  8387580 total,        0 free,  8387580 used.  9044716 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
11416 root      20   0   10.8g   2.2g   1416 S 193.8 14.4  21438,41 qemu-kvm
11918 apache    20   0   53924  10488   2204 R 100.0  0.1  22116:08 nginx
14499 mysql     20   0 4831996 233768   4120 S  50.0  1.4 165981:14 mysqld
 1228 apache    20   0  457424  12436   3392 S   6.2  0.1   0:04.55 httpd
 3414 apache    20   0  457424  11868   3336 S   6.2  0.1   0:01.72 httpd
 3862 apache    20   0  457424  11844   3304 S   6.2  0.1   0:01.06 httpd
 4405 apache    20   0  457424  12120   3384 S   6.2  0.1   0:03.48 httpd
 5758 apache    20   0  457424  11904   3332 S   6.2  0.1   0:01.69 httpd
 6144 apache    20   0  457424  12136   3388 S   6.2  0.1   0:03.46 httpd
 9075 apache    20   0  457424  12132   3384 S   6.2  0.1   0:05.13 httpd
 9181 apache    20   0  457680  12500   3428 S   6.2  0.1   0:10.19 httpd
13347 apache    20   0  457424  11868   3332 S   6.2  0.1   0:02.12 httpd
14461 apache    20   0  457424  11852   3304 S   6.2  0.1   0:00.77 httpd
14462 apache    20   0  457424  11856   3304 S   6.2  0.1   0:00.77 httpd
16706 apache    20   0  457156  11784   3304 S   6.2  0.1   0:00.70 httpd
18737 apache    20   0  457424  12288   3400 S   6.2  0.1   0:05.89 httpd
18783 apache    20   0  457424  12524   3496 S   6.2  0.1   0:05.93 httpd
19742 apache    20   0  457424  12552   3500 S   6.2  0.1   0:08.48 httpd
22577 apache    20   0  457424  12212   3444 S   6.2  0.1   0:04.00 httpd
24543 apache    20   0  457424  12116   3384 S   6.2  0.1   0:03.28 httpd
27945 apache    20   0  457412  11876   3332 S   6.2  0.1   0:01.89 httpd
28057 apache    20   0  457424  11808   3296 S   6.2  0.1   0:00.43 httpd
31109 apache    20   0  457424  11800   3304 S   6.2  0.1   0:01.18 httpd
```

Рисунок 4.13 – Навантаження на сервер при використанні MySQL бази даних для збереження незмінних важливих даних

```
top - 21:02:48 up 91 days, 21:27,  1 user,  load average: 0.37, 0.37, 0.40
Tasks: 187 total,   1 running, 186 sleeping,   0 stopped,   0 zombie
%Cpu(s):  4.6 us,  1.1 sy,  0.0 ni, 92.8 id,  1.3 wa,  0.0 hi,  0.2 si,  0.0 st
KiB Mem:  16327680 total, 14539580 used,  1788100 free,   333324 buffers
KiB Swap:  8387580 total,        0 used,  8387580 free. 12586184 cached Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 1527 mysql     20   0 2429444 280280  11680 S   7.3  1.7  12331:45 mysqld
 2908 www-data  20   0   51980  25164   4524 S   4.3  0.2 435:04.19 nginx
32654 www-data  20   0  386360  12148   2536 S   0.3  0.1   0:17.98 apache2
    1 root      20   0   29684   5460   3040 S   0.0  0.0   0:57.97 systemd
```

Рисунок 4.14 – Навантаження на сервер при використанні Blockchain як бази даних для збереження незмінних важливих даних у блоках

Приведені дані з сервісу Cloudflare статистики запитів на анонсер, який покритий проксі сервером, до і після введення в експлуатацію:

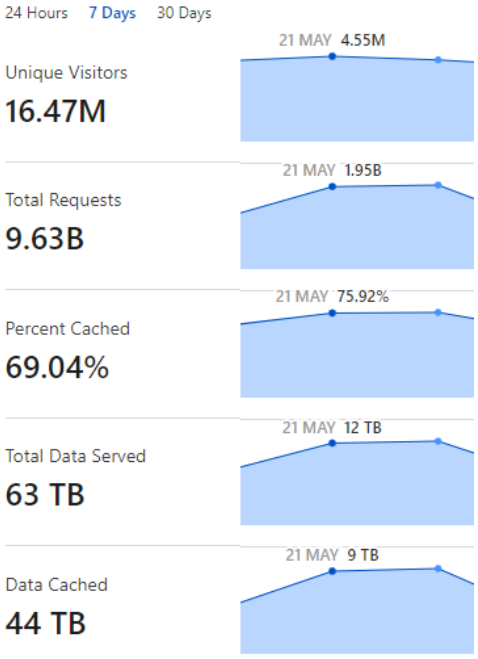


Рисунок 4.15 – Кількість запитів на анонсер до роботи системи

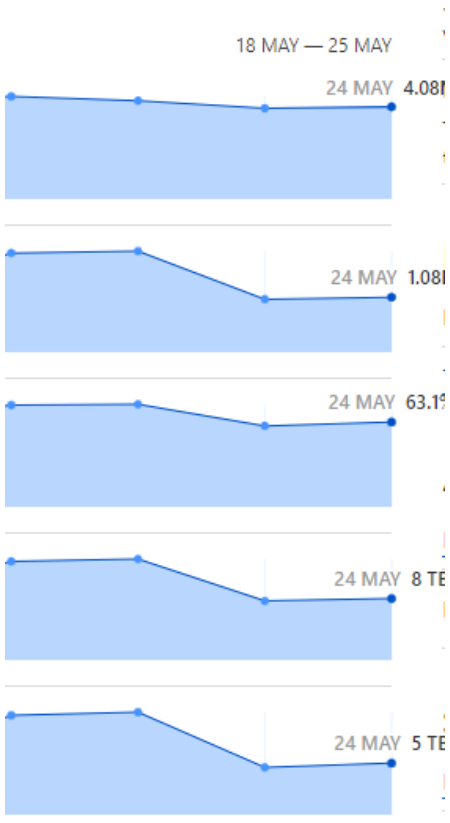


Рисунок 4.16 – Кількість запитів на анонсер після запуску роботи системи

Зниження запитів майже удвічі зумовлене тим, що падає навантаження на процесор сервера при такій схемі роботи і на сервері, який обробляє запити, не виникає проблеми з keep-timeout, де також знижується можливість втрати даних.

Розроблена модель Великих даних у блокчейн та її реалізація дає змогу зберігати дані такого розміру, що певні реляційні бази даних, наприклад MySQL, навіть на потужних серверах, не змогли б виконати операції CRUD (Create, Update, Delete) в час, зазначений у вимогах якостей Кодда (12 с) [89 – 101].

4.5. Висновки

У даному розділі розроблена архітектура системи, описаний та розроблений протокол для обміну даних, спроектована для майбутньої розробки архітектура медичної системи з використання блокчейн-технології для бази даних незмінних даних.

Розроблено архітектуру Великих даних у блокчейн для медицини, яка використана в інформаційній системі перевірки ліків, впровадженій в лікарні швидкої допомоги м. Львова. Розроблену систему можна використовувати, як конструктор для оптимізації даних, які є незмінними, наприклад, логи для статистики тощо.

Запис Великих даних у блокчейн на тестовій системі дозволив знизити кількість запитів майже удвічі. Це зумовлене тим, що падає навантаження на процесор сервера при такій схемі роботи і на сервері, який обробляє запити, не виникає проблеми з keep-timeout, де також знижується можливість втрати даних. Також експериментально доведено зменшення кількості запитів до системи (втричі) через підвищення якості даних, а саме через зменшення кількості дублікатів та блоків-сиріт.

Розроблена реалізація моделі Великих даних у блокчейн дає змогу зберігати дані такого розміру, що певні реляційні бази даних (тестування відбувалося на MySQL) навіть на потужних серверах, не змогли б виконати операції CRUD (Create, Update, Delete) в час, зазначений у вимогах якостей Кодда (12 с).

Проведена апробація результатів на практиці методом демонстрації навантаження до введення експлуатації і після введення в експлуатацію.

Результати розділу опубліковано в наукових працях автора [6, 9, 10].

ОСНОВНІ РЕЗУЛЬТАТИ ТА ВИСНОВКИ

У дисертаційній роботі вирішена актуальна науково-прикладна задача підвищення якості даних та скорочення часу доступу до даних на основі розроблення інформаційної технології блокчейн для Великих даних. При цьому отримано такі основні результати.

1. На підставі проведеного аналізу відомих рішень встановлено, що існуючі технології опрацювання даних, що використовують бази даних, не можуть бути використані у блокчейн через неможливість забезпечення прийнятного часу доступу до даних та їх опрацювання.
2. Розроблено модель Великих даних у блокчейн з поданням Великих даних як послідовності заголовків, метаданих та даних. Це дає змогу зберігати дані різної структури, зменшувати суперечність у даних, що надходять з різних джерел, а також фіксувати час надходження даних та створювати зв’язок з попереднім блоком, використовуючи додаткову хеш-функцію. Розроблена модель Великих даних у блокчейн та її реалізація дає змогу зберігати дані такого розміру, що певні реляційні бази даних (тестування відбувалося на MySQL) навіть на потужних серверах, не змогли б виконати операції CRUD (Create, Update, Delete) в час, зазначений у вимогах якостей Кодда (12 с).
3. Розроблено метод перевірки якості внесених даних, який полягає в пошуку унікальних контрольних сум та обчисленні хешів блоків, а також будується на основі інформаційної моделі Великих даних у блокчейн, що дає змогу забезпечити безутратне внесення даних та гарантує їх цілісність. Також це дало змогу втричі зменшити

кількість запитів до даних через зменшення кількості дублікатів та блоків-сиріт.

4. Розроблено метод визначення блоків-сиріт на основі марковських ланцюгів, шляхом додавання таких блоків як нової одиниці у ланцюжку блоків та врахування часу затримки у прийнятті блоків вузлами. Це дає змогу забезпечити цілісність даних та усунути надмірність у них.
5. Вдосконалено метод запису даних у блокчейн шляхом включення додаткових параметрів у заголовок блоку, підпису попереднього блоку та перевірки наявності блоку-предка. Це дає змогу гарантувати, що попередній блок не був змінений, а також скоротити час доступу до даних пропорційно до кількості серверів.
6. На основі методу запису в блокчейн розроблено алгоритм запису транзакцій у блокчейн. Таким чином, маючи транзакцію, можна записати в неї будь-які дані, та записати їх назавжди в блокчейн. Експериментально визначно, що навантаженість на сервер знижується втричі у випадку збереження даних у блокчейн порівняно з реляційною базою даних.
7. Розроблено алгоритм перевірки якості внесених даних, який базується на аналізі наявності дублікатів, суперечливих даних та даних з пропусками. Це дало змогу зменшити кількість запитів на анонсер майже удвічі. Також це дає змогу усунути проблему з keep-timeout та знизити можливість втрати даних.
8. Розроблено архітектуру Великих даних у блокчейн для медицини, яка використана в інформаційній системі перевірки ліків, впровадженій в лікарні швидкої допомоги м. Львова. Розроблену

систему можна використовувати, як конструктор для оптимізації даних, які є незмінними, наприклад, логи для статистики тощо.

Результати дисертаційної роботи також використано в навчальному процесі Національного університету «Львівська політехніка» при викладанні дисципліни «Хмарні технології» на кафедрі систем штучного інтелекту.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Zimmerman P. Blockchain structure and cryptocurrency prices // Staff working Paper / Bank of England. – 2020. – No 855, February 2020. – P.1-75. URL: https://www.bankofengland.co.uk/-/media/boe/files/working-paper/2020/blockchain-structure-and-cryptocurrency-prices.pdf

2. Biktimirov M. R. et al. Blockchain technology: Universal structure and requirements // Automatic Documentation and Mathematical Linguistics. – 2017. – Vol. 51 (6). – P. 235-238.
URL: https://link.springer.com/article/10.3103/S0005105517060036

3. Zheng Z. et al. Blockchain challenges and opportunities: A survey // International journal of web and grid services. – 2018. – Vol. 14, No 4. – P. 352-375. URL: https://allquantor.at/blockchainbib/pdf/zheng2018blockchain.pdf

4. Belotti M. et al. A vademecum on blockchain technologies: When, which, and how // IEEE Communications Surveys & Tutorials. – 2019. – Vol. 21 (4). – P. 3796-3838. URL: https://ieeexplore.ieee.org/document/8760539

5. Nofer M. et al. Blockchain // Business & Information Systems Engineering. – 2017. – Vol. 59, No 3. – P. 183-187.
URL: https://link.springer.com/article/10.1007/s12599-017-0467-3

6. Elrom E. Blockchain Nodes // The Blockchain Developer: Practical Guide for Designing, Implementing, Publishing, Testing, and Securing Distributed Blockchain-based Projects / Elad Elrom. – Apress Berkeley CA, 2019. – Ch. 2. – P. 31-72. URL: https://link.springer.com/article/10.1007/s12599-017-0467-3

7. Kushch S., Prieto-Castrillo F. Blockchain for dynamic nodes in a smart city // 2019 IEEE 5th World Forum on Internet of Things (WF-IoT). – IEEE, 2019. – P. 29-34. URL: https://wfiot.jkjmanagement.com/papers/1570523247.pdf

8. Florian M. et al. Erasing data from blockchain nodes // 2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW). – IEEE, 2019. – P. 367-376. URL: https://ieeexplore.ieee.org/document/8802472

9. Zhang R., Xue R., Liu L. Security and privacy on blockchain // ACM Computing Surveys (CSUR). – 2019. – Vol. 52 (3), art. No 51. – P. 30.
URL: https://dl.acm.org/doi/pdf/10.1145/3316481

10. Li X. et al. A survey on the security of blockchain systems // Future Generation Computer Systems. – 2020. – Vol. 107. – P. 841-853. URL: https://www.sciencedirect.com/science/article/abs/pii/S0167739X17318332

11. Stephen R., Alex A. A review on blockchain security // International Conference on Recent Advancements and Effectual Researches in Engineering Science and Technology (RAEREST) : Materials Science and Engineering. – IOP Publishing, 2018. – Vol. 396 (1), art. No 012030.
URL: https://iopscience.iop.org/article/10.1088/1757-899X/396/1/012030/pdf

12. Karame G., Capkun S. Blockchain security and privacy // IEEE Security & Privacy. – 2018. – Vol. 16 (04). – P. 11-12. URL: https://www.computer.org/csdl/magazine/sp/2018/04/msp2018040011/13rRUxBJhE9

13. Moubarak J., Filiol E., Chamoun M. On blockchain security and relevant attacks // IEEE Middle East and North Africa Communications Conference (MENACOMM), 18-20 April 2018. – IEEE, 2018. – P. 1-6.
URL: https://ieeexplore.ieee.org/document/8371010

14. Park J. H., Park J. H. Blockchain security in cloud computing: Use cases, challenges, and solutions //Symmetry. – 2017. – Vol. 9 (8), art No 164. URL: https://www.mdpi.com/2073-8994/9/8/164

15 Bach L. M., Mihaljevic B., Zagar M. Comparative analysis of blockchain consensus algorithms // 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). – IEEE, 2018. – P. 1545-1550. URP: https://ieeexplore.ieee.org/abstract/document/8400278

16. Lashkari B., Musilek P. A comprehensive review of blockchain consensus mechanisms // IEEE Access. – 2021. – Vol. 9. – P. 43620-43652. URL: https://ieeexplore.ieee.org/abstract/document/9376868

17. Mingxiao D. et al. A review on consensus algorithm of blockchain // 2017 IEEE international conference on systems, man, and cybernetics (SMC). – IEEE, 2017. – P. 2567-2572. URL: https://ieeexplore.ieee.org/abstract/document/8123011

18. Klinkmüller C. et al. Mining blockchain processes: extracting process mining data from blockchain applications // Management Business Process Management: Blockchain and Central and Eastern Europe. – Springer Cham, 2019. – Vol. 361. – P. 71-86. – URL: https://link.springer.com/chapter/10.1007/978-3-030-30429-4_6L

19. Islam N. et al. Is blockchain mining profitable in the long run? // IEEE Transactions on Engineering Management. – 2021. – P. 1-14. URL: https://ieeexplore.ieee.org/abstract/document/9325951

20. Yalcin H., Daim T. Mining research and invention activity for innovation trends: case of blockchain technology // Scientometrics. – 2021. – Vol. 126, No 5. – P. 3775-3806. URL: https://link.springer.com/article/10.1007/s11192-021-03876-4

21. Dorfleitner G., Muck F., Scheckenbach I. Blockchain applications for climate protection: A global empirical investigation // Renewable and Sustainable Energy Reviews. – 2021. – Vol. 149, art. No 111378. URL: https://www.sciencedirect.com/science/article/abs/pii/S1364032121006638

22. Qureshi A., Megías Jiménez D. Blockchain-based multimedia content protection: Review and open challenges // Applied Sciences. – 2020. – Vol. 11. – No 1, art. No 1. URL: https://www.mdpi.com/2076-3417/11/1/1

23. Wei P. C. et al. Blockchain data-based cloud data integrity protection mechanism //Future Generation Computer Systems. – 2020. – Vol. 102. – P. 902-911. URL: https://www.sciencedirect.com/science/article/abs/pii/S0167739X19313494

24. Boireau O. Securing the blockchain against hackers // Network Security. – 2018. – Vol. 2018, No 1. – P. 8-11.
URL: https://www.magonlinelibrary.com/toc/nese/2018/1

25. B. Rawat D., Chaudhary V., Doku R. Blockchain technology: Emerging applications and use cases for secure and trustworthy smart systems // Journal of Cybersecurity and Privacy. – 2020. – Vol. 1 (1). – P. 4-18. URL: https://www.mdpi.com/2624-800X/1/1/2

26. Li H. et al. A blockchain-based public auditing protocol with self-certified public keys for cloud data //Security and Communication Networks. – 2021. – Vol. 2021, art. No 3091104. – P. 10.
URL: https://downloads.hindawi.com/journals/scn/2021/3091104.pdf

27. Sagiroglu S., Sinanc D. Big data: A review //2013 international conference on collaboration technologies and systems (CTS). – IEEE, 2013. – P. 42-47. URL: https://ieeexplore.ieee.org/abstract/document/6567202

28. Shakhovska N. et al. Big Data analysis in development of personalized medical system // Procedia Computer Science. – 2019. – Vol. 160. – P. 229-234. URL: https://www.sciencedirect.com/science/article/pii/S187705091931676X

29. Yaqoob I. et al. Big data: From beginning to future // International Journal of Information Management. – 2016. – Vol. 36, Issue 6. – P. 1231-1247. URL: https://www.sciencedirect.com/science/article/abs/pii/S0268401216304753

30. Fan J., Han F., Liu H. Challenges of big data analysis // National science review. – 2014. – Vol. 1, Issue 2 (June 2014). – P. 293-314. URL: https://academic.oup.com/nsr/article/1/2/293/1397586

31. Vijayarani S., Sharmila S. Research in big data: an overview // Informatics Engineering, an International Journal (IEIJ). – 2016. – Vol. 4, No.3 (September 2016). – P. 1-20.
URL: https://www.researchgate.net/publication/339551786_RESEARCH_IN_BIG_DATA_-AN_OVERVIEW

32. Wu X. et al. Data mining with big data // IEEE transactions on knowledge and data engineering. – 2013. – Vol. 26, Issue 1. – P. 97-107.
URL: https://ieeexplore.ieee.org/abstract/document/6547630

33. Rodríguez-Mazahua L. et al. A general perspective of Big Data: applications, tools, challenges and trends // The Journal of Supercomputing. – 2016. – Vol. 72. – P. 3073-3113.
URL: https://link.springer.com/article/10.1007/s11227-015-1501-1

34. Addo-Tenkorang R., Helo P. T. Big data applications in operations/supply-chain management: A literature review // Computers & Industrial Engineering. – 2016. – Vol. 101. – P. 528-543.
URL: https://www.sciencedirect.com/science/article/abs/pii/S0360835216303631

35. Furht B., Villanustre F. Big data technologies and applications. – Springer International Publishing, Switzerland, 2016. – 405 p.
URL: https://link.springer.com/book/10.1007/978-3-319-44550-2

36. Yang W. et al. Big data real-time processing based on storm // 2013 12th IEEE international conference on trust, security and privacy in computing and communications, 16-18 July 2013. – IEEE, 2013. – P. 1784-1787.
URL: https://ieeexplore.ieee.org/document/6681052

37. Rathore M. M. U. et al. Real-time big data analytical architecture for remote sensing application // IEEE journal of selected topics in applied earth observations and remote sensing. – 2015. – Vol. 8, Issue 10. – P. 4610-4621.
URL: https://ieeexplore.ieee.org/abstract/document/7109130

38. Mohamed N., Al-Jaroodi J. Real-time big data analytics: Applications and challenges // 2014 international conference on high performance computing & simulation (HPCS), 21-25 July 2014. – IEEE, 2014. – P. 305-310. URL: https://ieeexplore.ieee.org/abstract/document/6903700

39. Choi T. M., Lambert J. H. Advances in risk analysis with big data // Risk Analysis. – 2017. – Vol. 37, Issue 8. – P. 1435-1442. URL: https://onlinelibrary.wiley.com/doi/10.1111/risa.12859

40. Shi Y. Advances in big data analytics: theory, algorithms and practices. – Springer Nature, 2022. – 728 p. URL: https://link.springer.com/book/10.1007/978-981-16-3607-3

41. Peng L. et al. The advances and challenges of deep learning application in biological big data processing // Current Bioinformatics. – 2018. – Vol. 13, No 4. – P. 352-359.
URL:https://www.ingentaconnect.com/content/ben/cbio/2018/00000013/00000004/art00007

42. Babiceanu R. F., Seker R. Big Data and virtualization for manufacturing cyber-physical systems: A survey of the current status and future outlook // Computers in industry. – 2016. – Vol. 81. – P. 128-137.
URL: https://www.sciencedirect.com/science/article/abs/pii/S0166361516300471

43. Mathivanan S., Jayagopal P. A big data virtualization role in agriculture: a comprehensive review // Walailak Journal of Science and Technology (WJST). – 2019. – Vol. 16, No 2. – P. 55-70.
URL: https://wjst.wu.ac.th/index.php/wjst/article/view/3620/957

44. Thuemmler C., Bai C. (ed.). Health 4.0: How virtualization and big data are revolutionizing healthcare. – Cham : Springer International Publishing, 2017. – 253 p. URL: https://link.springer.com/book/10.1007/978-3-319-47617-9

45. Rajaraman V. Big data analytics // Resonance. – 2016. – Vol. 21, No 8. – P. 695-716. URL: https://link.springer.com/article/10.1007/s12045-016-0376-7

46. Cardenas A. A., Manadhata P. K., Rajan S. P. Big data analytics for security // IEEE Security & Privacy. – 2013. – Vol. 11, Issue 6. – P. 74-76.
URL: https://ieeexplore.ieee.org/abstract/document/6682971

47. Fisher D. et al. Interactions with big data analytics // Interactions. – 2012. – Vol. 19, Issue 3. – P. 50-59.
URL: https://dl.acm.org/doi/fullHtml/10.1145/2168931.2168943

48. Dean J., Ghemawat S. MapReduce: simplified data processing on large clusters // Communications of the ACM. – 2008. – Vol. 51, No 1. – P. 107-113.
URL: https://www.cs.amherst.edu/~ccmcgeoch/cs34/papers/p107-dean.pdf

49. Han J. et al. Survey on NoSQL database // 2011 6th international conference on pervasive computing and applications. – IEEE, 2011. – P. 363-366.
URL: https://ieeexplore.ieee.org/abstract/document/6106531

50. Lam C. Hadoop in action. – Simon and Schuster, 2010. – 336 p. URL: https://www.manning.com/books/hadoop-in-action

51. Johnson D., Menezes A., Vanstone S. The elliptic curve digital signature algorithm (ECDSA) // International journal of information security. – 2001. – Vol. 1, No 1. – P. 36-63. URL: https://link.springer.com/article/10.1007/s102070100002

52. Vaudenay S. The Security of DSA and ECDSA // International workshop on public key cryptography. – Springer, Berlin, Heidelberg, 2003. – P. 309-323.
URL: https://link.springer.com/content/pdf/10.1007/3-540-36288-6_23.pdf

53. Doerner J. et al. Threshold ECDSA from ECDSA assumptions: the multiparty case // 2019 IEEE Symposium on Security and Privacy (SP). – IEEE, 2019. – P.1051-1066. URL: https://ieeexplore.ieee.org/abstract/document/8835354

54. Nyame G. et al. An ECDSA approach to access control in knowledge management systems using blockchain // Information. – 2020. – Vol. 11 (2), art. No 111. URL: https://www.mdpi.com/2078-2489/11/2/111

55. Sahoo M. et al. A blockchain based framework secured by ECDSA to curb drug counterfeiting // 2019 10th international conference on computing, communication and networking technologies (ICCCNT). – IEEE, 2019. – P. 1-6.
URL: https://ieeexplore.ieee.org/abstract/document/8944772

56. Xiong H. et al. On the design of blockchain-based ECDSA with fault-tolerant batch verification protocol for blockchain-enabled IoMT // IEEE journal of biomedical and health informatics. – 2021. – Vol. 26, Issue 5. – P. 1977-1986. URL: https://ieeexplore.ieee.org/abstract/document/9540329

57. Gennaro R., Goldfeder S., Narayanan A. Threshold-optimal DSA/ECDSA signatures and an application to bitcoin wallet security // International Conference on Applied Cryptography and Network Security. – Springer, Cham, 2016. – P. 156-174. URL: https://link.springer.com/content/pdf/10.1007/978-3-319-39555-5_9.pdf

58. Yi X., Lam K. Y. A new blind ECDSA scheme for bitcoin transaction anonymity // Proceedings of the 2019 ACM ASIA Conference on Computer and Communications Security. – 2019. – P. 613-620.
UPL: https://dl.acm.org/doi/abs/10.1145/3321705.3329816

59. Lindell Y., Nof A. Fast secure multiparty ECDSA with practical distributed key generation and applications to cryptocurrency custody // Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. – 2018. – P. 1837-1854. URL: https://dl.acm.org/doi/10.1145/3243734.3243788

60. Saad M. et al. Overview of attack surfaces in blockchain // Blockchain for distributed systems security. – 2019. – Part. 1, Ch. 3. – P. 51-66. URL: https://onlinelibrary.wiley.com/doi/10.1002/9781119519621.ch3

61. Chen M. S., Han J., Yu P. S. Data mining: an overview from a database perspective // IEEE Transactions on Knowledge and data Engineering. – 1996. – Vol. 8, Issue 6. – P. 866-883. URL: https://ieeexplore.ieee.org/abstract/document/553155

62. Hand D. J. Principles of data mining // Drug safety. – 2007. – Vol. 30, Issue 7. – P. 621-622. URL: https://link.springer.com/article/10.2165/00002018-200730070-00010

63. Legout A., Urvoy-Keller G., Michiardi P. Understanding bittorrent: An experimental perspective [Technical Report]. – 2005. – 16 p. URL: https://hal.inria.fr/inria-00000156v3/document

64. Qiu D., Srikant R. Modeling and performance analysis of BitTorrent-like peer-to-peer networks // ACM SIGCOMM computer communication review. – 2004. – Vol. 34, Issue 4. – P. 367-378.
URL: https://dl.acm.org/doi/pdf/10.1145/1015467.1015508

65. Johnsen J. A., Karlsen L. E., Birkeland S. S. Peer-to-peer networking with BitTorrent // Department of Telematics, NTNU. – 2005. – 19 p.
URL: http://web.cs.ucla.edu/classes/cs217/05BitTorrent.pdf

66. Arthur D., Panigrahy R. Analyzing BitTorrent and related peer-to-peer networks // Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm. – 2006. – P. 961-969. URL: http://web.cs.elte.hu/~kiraly/alkszem/Arthur-Panigrahy.Analyzing_BitTorent.pdf

67. Development of the system to integrate and generate content considering the cryptocurrent needs of users / 1. Lytvyn, V., Vysotska, V., Kuchkovskiy, V., Bobyk, I., Malanchuk, O., Ryshkovets, Y., ... & Panasyuk, V. // Eastern-European Journal of Enterprise Technologies. – 2019. – Vol. 1, No 2. – P. 18-39. (Q2)
URL: http://journals.uran.ua/eejet/article/view/154709/156604

68. Application of Online Marketing Methods and SEO Technologies for Web Resources Analysis within the Region / Kuchkovskiy, V., Andrunyk, V., Krylyshyn, M., Chyrun, L., Vysotskyi, A., Chyrun, S., ... & Brodovska, I. // In Computational Linguistics and Intelligent Systems (COLINS 2021 5th International Conference, Lviv, 22–23 April 2021, CEUR workshop proceedings. Aachen, CEUR-WS. org. – 2021. – Vol. 2870. – P. 1652-1693. URL: http://ceur-ws.org/Vol-2870/paper121.pdf

69. Кучковський В. В., Шаховська Н. Б. Блокчейн як база-даних, його використання, опис розумних контрактів і майбутній потенціал // Моделювання та інформаційні технології. – 2019. – Вип. 87. – С. 109–116.
URL: http://nbuv.gov.ua/UJRN/Mtit_2019_87_16.

70. Кучковський В. В. Алгоритми консенсуса блокчейн систем // Вісник Хмельницького національного університету. Серія: Технічні науки. – 2021. – № 3 (297). – С. 30–33.
URL: http://journals.khnu.km.ua/vestnik/wp-content/uploads/2021/08/7-2.pdf

71. Литвин В. В., Висоцька В. А., Кучковський В. В., Дуткевич С. Ю., Наум О. Метод інтеграції та управління контентом мережі інформаційних ресурсів туризму згідно з потребами користувача // Вісник Національного університету “Львівська політехніка”. Серія: Інформаційні системи та мережі. – 2018. – № 901. – С. 22–36. URL: https://science.lpnu.ua/sites/default/files/journal-paper/2019/feb/15577/181912maket-22-36.pdf

72. Литвин В. В., Висоцька В. А., Кучковський В. В., Оливко Р. М. Архітектура інформаційної системи інтеграції та формування контенту про криптовалюти на основі аналізу діяльності бірж // Вісник Національного університету “Львівська політехніка”. Серія: Інформаційні системи та мережі. – 2018. – № 901. – С. 43–60. URL: https://science.lpnu.ua/sites/default/files/journal-paper/2019/feb/15579/181912maket-43-60.pdf

73. Architecture of system for content integration and formation based on cryptographic consumer needs / Lytvyn, V., Kuchkovskiy, V., Vysotska, V., Markiv, O., & Pabyrivskyy, V. // IEEE 13th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT). – 2018. – Vol. 1. – P. 391-395. URL: https://ieeexplore.ieee.org/abstract/document/8526669

74. Kuchkovskiy, V., & Shakhovska, N. Information technology of Blockchain: Database, smart contracts, architecture // IEEE 14th International Conference on Computer Sciences and Information Technologies (CSIT). – 2019. – Vol. 2. – P. 55-59. URL: https://ieeexplore.ieee.org/abstract/document/8929885

75. Кучковський В. В. Алгоритми консенсусу // Trends in science and practice of today: abstracts of XXVIII International scientific and practical conference (Ankara, Turkey; June 01 – 04, 2021). – 2021. – С. 502–505. URL:

https://isg-konf.com/wp-content/uploads/2021/05/XXVIII-ConferenceJune-01-042021.pdf#page=502

76. Кучковський В. В. Змішані алгоритми консенсусу // Trends in science and practice of today: abstracts of XXVIII International scientific and practical conference (Ankara, Turkey; June 01 – 04, 2021). – 2021. – С. 506–507. URL: https://isg-konf.com/wp-content/uploads/2021/05/XXVIII-ConferenceJune-01-042021.pdf#page=502

77. Проненко Т. В. Практичне застосування блокчейн технології у медицині // In Технічні та математичні науки. Студентський науковий форум. – 2018. – С. 83–89.

78. Полівода, К. А. Блокчейн–один із головних технологічних проривів останнього часу // Матеріали II Всеукраїнської науково-практичної Інтернет-конференції студентів, аспірантів і молодих вчених "Розвиток послуг та інновацій в цифровій економіці". – КНЕУ, 2019. – С. 251–253. URL: https://sci.ldubgd.edu.ua/bitstream/123456789/9555/1/Zbirnyk_4_19.pdf#page=251

79. Осядлий, В., & Москаленко, А. Система керування медичними даними на основі блокчейн-технологій // Measuring and computing devices in technological processes. – 2022. – Vol. 2. – P. 29–38.
URL: https://vottp.khmnu.edu.ua/index.php/vottp/article/view/45/45

80. Khrystynets, N., Miskevych, O., & Mazurenko, V. Технології Blockchain для оптимізації процесів документообігу // Computer-integrated technologies: education, science, production. – 2020. – No 40. – P. 153-157.
URL: http://cit-journal.com.ua/index.php/cit/article/view/172

81. Rüth J. et al. Digging into browser-based crypto mining // Proceedings of the Internet Measurement Conference (IMC' 18). – 2018. – С. 70-76. URL: https://dl.acm.org/doi/10.1145/3278532.3278539

82. Uchibeke U. U. et al. Blockchain access control ecosystem for big data security // 2018 IEEE International Conference on Internet of Things (iThings) and

IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData). – IEEE, 2018. – С. 1373-1378. URL: https://ieeexplore.ieee.org/document/8726516

83. Tariq N. et al. The security of big data in fog-enabled IoT applications including blockchain: A survey // Sensors. – 2019. – Т. 19. – №. 8. – С. 1788. URL: https://www.mdpi.com/1424-8220/19/8/1788

84. Li J. et al. Blockchain-based public auditing for big data in cloud storage // Information Processing & Management. – 2020. – Т. 57. – №. 6. – С. 102382. URL: https://doi.org/10.1016/j.ipm.2020.102382

85. Hasan M. K. et al. Blockchain technology on smart grid, energy trading, and big data: security issues, challenges, and recommendations // Wireless Communications and Mobile Computing. – 2022. – Т. 2022. URL: https://www.hindawi.com/journals/wcmc/2022/9065768/

86. Shrier D., Wu W., Pentland A. Blockchain & infrastructure (identity, data security) / Massachusetts Institute of Technology-Connection Science. – 2016.– Part. 3, May 2016. – 19 p. URL: https://www.getsmarter.com/blog/wp-content/uploads/2017/07/mit_blockchain_and_infrastructure_report.pdf

87. Hassani H., Huang X., Silva E. Big-crypto: Big data, blockchain and cryptocurrency // Big Data and Cognitive Computing. – 2018. – Vol. 2, No. 4. – P. 34. URL: https://www.mdpi.com/2504-2289/2/4/34

88. Kumar N. et al. (ed.). Blockchain, Big Data and Machine Learning: Trends and Applications. – CRC Press, 2020. – 360 p.

89. Zheng S. et al. A blockchain-based trading platform for big data // IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). – IEEE, 2020. – P. 991-996. URL: https://ieeexplore.ieee.org/abstract/document/9162759

90. Wang L., Jones R. Big data, cybersecurity, and challenges in healthcare // 2019 SoutheastCon. – IEEE, 2019. – P. 1-6.

URL: https://ieeexplore.ieee.org/document/9020632

91. Onik M. M. H. et al. Blockchain in healthcare: Challenges and solutions // Big data analytics for intelligent healthcare management. – Academic Press, 2019. – Ch. 8. – P. 197-226. URL: https://doi.org/10.1016/B978-0-12-818146-1.00008-8

92. Demirbaga U., Aujla G. S. MapChain: A Blockchain-based Verifiable Healthcare Service Management in IoT-based Big Data Ecosystem // IEEE Transactions on Network and Service Management. – 2022. – P. 99.
URL: https://ieeexplore.ieee.org/document/9878261

93. Hanley M., Tewari H. Managing lifetime healthcare data on the blockchain // 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI). – IEEE, 2018. – P. 246-251. URL: https://ieeexplore.ieee.org/document/8560055

94. Jigna Hathaliya et al. Blockchain-based remote patient monitoring in healthcare 4.0 // 2019 IEEE 9th international conference on advanced computing (IACC). – IEEE, 2019. – P. 87-91.
URL: https://ieeexplore.ieee.org/abstract/document/8971593

95. Prokofieva M. et al. Blockchain in healthcare // Australasian Journal of Information Systems. – 2019. – Vol. 23. – P.1-22.
URL: https://journal.acs.org.au/index.php/ajis/article/view/2203/897

96. Zhihan Lv, Liang Qiao. Analysis of healthcare big data // Future Generation Computer Systems. – 2020. – Vol. 109. – P. 103-110. URL: https://www.sciencedirect.com/science/article/abs/pii/S0167739X20304829

97. Chattu V. K. et al. The emerging role of blockchain technology applications in routine disease surveillance systems to strengthen global health security // Big Data and Cognitive Computing. – 2019. – Vol. 3, No 2. – P. 25. URL: https://www.mdpi.com/2504-2289/3/2/25

98. Attaran M. Blockchain technology in healthcare: Challenges and opportunities // International Journal of Healthcare Management. – 2022. – Vol. 15, No 1. – P. 70-83.
URL: https://www.tandfonline.com/doi/full/10.1080/20479700.2020.1843887

99. Dhagarra D. et al. Big Data and blockchain supported conceptual model for enhanced healthcare coverage: The Indian context // Business Process Management Journal. – 2019. – Vol. 25, No. 7. – P. 1612-1632. URL: https://www.emerald.com/insight/content/doi/10.1108/BPMJ-06-2018-0164/full/html

100. Hassani H., Huang X., Silva E. S. Fusing Big Data, blockchain, and cryptocurrency // Fusing Big Data, Blockchain and Cryptocurrency. – Palgrave Pivot, Cham, 2019. – Ch. 5. – C. 99-117.
URL: https://link.springer.com/book/10.1007/978-3-030-31391-3

101. Khezr S. et al. Blockchain technology in healthcare: A comprehensive review and directions for future research // Applied sciences. – 2019. – Vol. 9. – №. 9. – P. 1736. URL: https://www.mdpi.com/2076-3417/9/9/1736

ДОДАТКИ

Додаток А. Таблиця порівняння алгоритму хешрейту до процесора

Алгоритм	Хешрейт
allium	2573.93 kH/s
argon2	93.85 kH/s
argon2d250	146.24 kH/s
argon2d500	46.67 kH/s
argon2d4096	4603.22 H/s
axiom	231.85 H/s
blake	112.44 MH/s
blake2b	148.75 MH/s
blake2s	411.84 MH/s
blakecoin	367.57 MH/s
bmw512	82.34 MH/s
c11	3847.93 kH/s
decred	57.48 MH/s
deep	10.72 MH/s
dmd-gr	6125.47 kH/s
groestl	6149.45 kH/s
hex	1057.27 kH/s
hmq1725	1178.19 kH/s
hodl	896.00 H/s
jha	3076.43 kH/s
keccak	66.64 MH/s
keccakc	64.65 MH/s
lbry	27.42 MH/s
lyra2h	238.97 kH/s

lyra2re	4014.17 kH/s
lyra2rev2	5281.15 kH/s
lyra2rev3	5735.30 kH/s
lyra2z	1805.08 kH/s
lyra2z330	920.15 H/s
m7m	556.20 kH/s
minotaur	2429.50 kH/s
myr-gr	12.84 MH/s
neoscrypt	91.04 kH/s
nist5	8802.26 kH/s
pentablake	23.49 MH/s
phi1612	3021.00 kH/s
phi2	2035.89 kH/s
power2b	1386.78 H/s
quark	4366.62 kH/s
qubit	8244.45 kH/s
scrypt	200.68 kH/s
sha256d	145.03 MH/s
sha256q	94.52 MH/s
sha256t	165.03 MH/s
sha3d	31.09 MH/s
skein	89.52 MH/s
skein2	67.25 MH/s
skunk	3419.39 kH/s
sonoa	327.96 kH/s
timetravel	4285.32 kH/s
timetravel10	4065.62 kH/s

tribus	13.98 MH/s
vanilla	367.80 MH/s
whirlpool	4797.36 kH/s
whirlpoolx	16.45 MH/s
x11	3899.26 kH/s
x11evo	3182.95 kH/s
x11gost	2130.83 kH/s
x12	2554.30 kH/s
x13	2252.58 kH/s
x13bcd	2432.99 kH/s
x13sm3	1915.04 kH/s
x14	2236.82 kH/s
x15	1982.19 kH/s
x16r	6909.53 kH/s
x16rv2	6894.27 kH/s
x16rt	1785.64 kH/s
x16rt-veil	1821.56 kH/s
x16s	1897.45 kH/s
x17	2039.65 kH/s
x21s	1340.98 kH/s
x22i	1163.86 kH/s
x25x	481.73 kH/s
xevan	673.17 kH/s
yescrypt	14.79 kH/s
yescryptr8	14.51 kH/s
yescryptr8g	14.77 kH/s
yescryptr16	1276.31 H/s

yescryptr32	615.74 H/s
yespower	1339.61 H/s
yespowerr16	1332.25 H/s
zr5	1518.10 kH/s

Додаток Б. Декодована транзакція з блокчейну


```
{
 "txid": "ad09caa1fdf24c512ef6bcb30ade56e6e553aae196b441e70a3b2c1916a9f00c",
 "hash": "ad09caa1fdf24c512ef6bcb30ade56e6e553aae196b441e70a3b2c1916a9f00c",
 "version": 2,
 "size": 242,
 "vsize": 242,
 "weight": 968,
 "locktime": 0,
 "vin": [
  {
   "txid": "c54567f0e4753293d81dfa82ea71f8d992c5806df9453b38617c52448d218362",
   "vout": 1,
   "scriptSig": {
    "asm":
"304402207954c7afbfb3da3f3792580483da407ca09c9a1f8f060e71cd6ae5b95f5abde502202552b04f87b7f20
bc01811eb49385d588838824ad06618e7a4cf302e12449a25[ALL]
034e14fe1663a8ed2d068875a87aef4742155beccb4bd9279721e8494865c76b61",
    "hex":
"47304402207954c7afbfb3da3f3792580483da407ca09c9a1f8f060e71cd6ae5b95f5abde502202552b04f87b7f
20bc01811eb49385d588838824ad06618e7a4cf302e12449a250121034e14fe1663a8ed2d068875a87aef47421
55beccb4bd9279721e8494865c76b61"
   },
   "sequence": 4294967295
  }
 ],
 "vout": [
  {
```

```
    "value": 9.00000000,
    "n": 0,
    "scriptPubKey": {
     "asm": "OP_DUP OP_HASH160 154fda4a91130d498503e0493ada0170a0ad1253
OP_EQUALVERIFY OP_CHECKSIG",
     "hex": "76a914154fda4a91130d498503e0493ada0170a0ad125388ac",
     "reqSigs": 1,
     "type": "pubkeyhash",
     "addresses": [
      "2ExVocnskqgHXHmPSVZRBk5LWb8NsCXmf9"
     ]
    }
   },
   {
    "value": 0.00000000,
    "n": 1,
    "scriptPubKey": {
     "asm": "OP_RETURN
34354541424135413441393346313142323131433736303042363245463446323238414130413133",
    "hex":
"333536333436333633347c32343438377c333234343337333133343332333333343333333343333333233333
330333234343336333733343436333434333336343333343334333533353336343233353330333633333335
333533343335333333383030303030303030303030303030303030303030307c356662366563326335653833
135383762616661323234613963393466613534 7c6537323937303538323063613437633233393963623134
326237633064353666353730663437313 97c33343438383332307c34383631313332397c323433393035373
635327c313634313939323432397c313634313938383832397c36312e31382e3231322e32380d0a
",
     "type": "nulldata"
    }
   }
  ]
}
```

Додаток В. Зміст торрент файлу

```
64                                          ;  - DICTIONARY (d)
|  35 3A 66 69 6C 65 73                     ;     - 5:files
|  6C                                       ;     - LIST (l)
|  |  64                                    ;        - DICTIONARY (d)
|  |  |  36 3A 6C 65 6E 67 74 68            ;          - 6:length
|  |  |  69 36 31 35 65                     ;          - i615e
|  |  |  34 3A 70 61 74 68                  ;          - 4:path
|  |  |  6C                                 ;          - LIST (l)
|  |  |  |  32 36 3A 66 72 65 65 20 61 75   ;          - 26:free audiobook verison.txt
|  |  |  |  64 69 6F 62 6F 6F 6B 20 76 65   ;
|  |  |  |  72 73 69 6F 6E 2E 74 78 74      ;
|  |  |  65                                 ;             - END (e)
|  |  65                                    ;           - END (e)
|  |  64                                    ;           - DICTIONARY (d)
|  |  |  36 3A 6C 65 6E 67 74 68            ;             - 6:length
|  |  |  69 33 39 33 34 31 37 65            ;             - i393417e
|  |  |  34 3A 70 61 74 68                  ;             - 4:path
|  |  |  6C                                 ;             - LIST (l)
|  |  |  |  36 31 3A 57 61 72 63 72 61 66   ;  - 61:Warcraft_ Official Movie
Novelization by Christie Golden.epub
|  |  |  |  74 5F 20 4F 66 66 69 63 69 61   ;
|  |  |  |  6C 20 4D 6F 76 69 65 20 4E 6F   ;
|  |  |  |  76 65 6C 69 7A 61 74 69 6F 6E   ;
|  |  |  |  20 62 79 20 43 68 72 69 73 74   ;
|  |  |  |  69 65 20 47 6F 6C 64 65 6E 2E   ;
|  |  |  |  65 70 75 62                     ;
|  |  |  65                                 ;             - END (e)
|  |  65                                    ;           - END (e)
|  65                                       ;        - END (e)
|  34 3A 6E 61 6D 65                        ;        - 4:name
|  36 31 3A 57 61 72 63 72 61 66            ;  - 61:Warcraft_ Official Movie
Novelization by Christie Golden EPUB
|  74 5F 20 4F 66 66 69 63 69 61            ;
|  6C 20 4D 6F 76 69 65 20 4E 6F            ;
|  76 65 6C 69 7A 61 74 69 6F 6E            ;
|  20 62 79 20 43 68 72 69 73 74            ;
|  69 65 20 47 6F 6C 64 65 6E 20            ;
|  45 50 55 42                              ;
|  31 32 3A 70 69 65 63 65 20 6C            ;        - 12:piece length
|  65 6E 67 74 68                           ;
|  69 31 30 34 38 35 37 36 65               ;        - i10485765e
|  36 3A 70 69 65 63 65 73                  ;        - 6:pieces
|  32 30 3A 43 92 4C 22 BB 42 9E            ;        - 20:43 92 4C 22 BB 42 9E EA BD FF
66 C6 79 4C 29 E4 F9 D0 F3 B9 65 65
|  EA BD FF 66 C6 79 4C 29 E4 F9            ;
|  D0 F3 B9                                 ;
65                                          ;     - END (e)
```

Додаток Г. Вихідний зміст торрент файлу

```
64 35 3A 66 69 6C 65 73 6C 64 36 3A 6C 65 6E 67 74 68 69 36 31 35 65 34
3A 70 61 74 68 6C 32 36 3A 66 72 65 65 20 61 75 64 69 6F 62 6F 6F 6B 20
76 65 72 73 69 6F 6E 2E 74 78 74 65 65 64 36 3A 6C 65 6E 67 74 68 69 33
39 33 34 31 37 65 34 3A 70 61 74 68 6C 36 31 3A 57 61 72 63 72 61 66 74
5F 20 4F 66 66 69 63 69 61 6C 20 4D 6F 76 69 65 20 4E 6F 76 65 6C 69 7A
61 74 69 6F 6E 20 62 79 20 43 68 72 69 73 74 69 65 20 47 6F 6C 64 65 6E
2E 65 70 75 62 65 65 65 34 3A 6E 61 6D 65 36 31 3A 57 61 72 63 72 61 66
74 5F 20 4F 66 66 69 63 69 61 6C 20 4D 6F 76 69 65 20 4E 6F 76 65 6C 69
7A 61 74 69 6F 6E 20 62 79 20 43 68 72 69 73 74 69 65 20 47 6F 6C 64 65
6E 20 45 50 55 42 31 32 3A 70 69 65 63 65 20 6C 65 6E 67 74 68 69 31 30
34 38 35 37 36 65 36 3A 70 69 65 63 65 73 32 30 3A 43 92 4C 22 BB 42 9E
EA BD FF 66 C6 79 4C 29 E4 F9 D0 F3 B9 65
```

Додаток Ґ. Перетворення стрічки в hex

```
function strToHex($string) {
    $hex='';
    for ($i=0; $i < strlen($string); $i++) {
        $hex .= dechex(ord($string[$i]));
    }
    return $hex;
}
```

Додаток Д. Перетворення hex в стрічку

```
function hexToStr($hex) {
    $string='';
    for ($i=0; $i < strlen($hex)-1; $i+=2) {
        $string .= chr(hexdec($hex[$i].$hex[$i+1]));
    }
    return $string;
}
```

Додаток Є. Код анонсера

```
function errorexit($reason) {
	exit(bencode(array('failure reason' => $reason)));
}

function resolve_ip($host) {
	$ip = ip2long($host);
	if (($ip === false) || ($ip == -1)) {
		$ip = ip2long(gethostbyname($host));
		if (($ip === false) || ($ip == -1)) {
			return false;
		}
	}
	return $ip;
}

function ipv6_numeric($ip) {
   $binNum = '';
   foreach (unpack('C*', inet_pton($ip)) as $byte) {
      $binNum .= str_pad(decbin($byte), 8, "0", STR_PAD_LEFT);
   }
   return base_convert(ltrim($binNum, '0'), 2, 10);
}

//header('Content-Type: text/plain');

if (empty($_GET['info_hash']) || empty($_GET['port']) ||
!is_numeric($_GET['port']) || empty($_GET['peer_id']) ||
```

```
!isset($_GET['uploaded']) || !is_numeric($_GET['uploaded']) || !isset($_GET['downloaded']) || !is_numeric($_GET['downloaded']) || !isset($_GET['left']) || !is_numeric($_GET['left']) || (!empty($_GET['event']) && ($_GET['event'] != 'started') && ($_GET['event'] != 'completed') && ($_GET['event'] != 'stopped'))) {
	errorexit('invalid request');
}

if ($require_announce_protocol == 'no_peer_id') {
	if (empty($_GET['compact']) && empty($_GET['no_peer_id'])) {
		errorexit('standard announces not allowed; use no_peer_id or compact option');
	}
}
else if ($require_announce_protocol == 'compact') {
	if (empty($_GET['compact'])) {
		errorexit('tracker requires use of compact option');
	}
}

//if (!isset($_GET['ip'])) $_GET['ip'] = '0.0.0.0';
//$_SERVER['REMOTE_ADDR'] = $_SERVER['HTTP_X_REAL_IP'];

if (!(strpos($_SERVER['REMOTE_ADDR'], ":") > -1)) {
  $ip = resolve_ip(empty($_GET['ip']) ? $_SERVER['REMOTE_ADDR'] : $_GET['ip']);
	if ($ip === false) {
		errorexit("unable to resolve host name $_GET[ip]");
```

```
        }
} else {
   $ip = ipv6_numeric(empty($_GET['ip']) ? $_SERVER['REMOTE_ADDR'] :
$_GET['ip']);
        if ($ip === false) {
                errorexit("unable to resolve host name $_GET[ip]");
        }
}

$announce_interval = rand(3600, 10800);

if (!empty($_GET['event']) && ($_GET['event'] == 'stopped')) {
        $expire_time = 0;
}
else {
        $expire_time = $announce_interval * $expire_factor;
}

$md5 = md5($_GET['info_hash'].$_GET['peer_id']);
$sha1 = sha1($_GET['info_hash']);
//echo $ip;

$res = dbquery("SELECT info_hash FROM ".$db_table_bt." WHERE
info_md5='".$md5."'");
if (!dbrows($res)) {
```

```
$columns = '`info_hash`, `info_ip`, `info_port`, `info_peer`, `info_md5`, `info_sha1`, `info_upload`, `info_download`, `info_left`, `info_expire`, `info_ipv46`';

$values = '\" . strToHex($_GET['info_hash'])  . '\', ' . $ip . ', ' . $_GET['port'] . ', \" . strToHex($_GET['peer_id']). '\', \" . $md5. '\', \" . $sha1. '\', ' . $_GET['uploaded'] . ', ' . $_GET['downloaded'] . ', ' . $_GET['left'] . ', ' . (time()+round($expire_time)). ', \" . $_SERVER['REMOTE_ADDR']. '\";
dbquery("INSERT IGNORE INTO `$db_table_bt` ($columns) VALUES ($values);") or errorexit('database error');

} else {
	if (!dbquery("UPDATE ".$db_table_bt." SET info_upload='".$_GET['uploaded']."', info_download='".$_GET['downloaded']."', info_left='".$_GET['left']."', info_expire = ".(time() + +round($expire_time))." WHERE info_md5='".$md5."'")) errorexit('database error');
}

$peers = array();
$numwant = empty($_GET['numwant']) ? 50 : intval($_GET['numwant']);
$query = dbquery("SELECT * FROM ".$db_table_bt." WHERE info_hash = '".strToHex($_GET['info_hash'])."' AND info_expire > '".time()."' LIMIT ".$numwant."") or errorexit('database error'); // ORDER BY RAND()

if (!empty($_REQUEST['compact'])) {
	$peers = '';
	while ($array = dbarray($query)) {
```

```
                $peers .= pack('Nn', $array['info_ip'], $array['info_port']);
        }
}
else if (!empty($_REQUEST['no_peer_id'])) {
        while ($array = dbarray($query)) {
                $peers[] = array('ip' => long2ip($array['info_ip']), 'port' =>
intval($array['info_port']));
        }
}
else {
        while ($array = dbarray($query)) {
                $peers[] = array('ip' => long2ip($array['info_ip']), 'port' =>
intval($array['info_port']), 'peer id' => hexToStr($array['info_peer']));
        }
}


$db_connect->close();

exit(bencode(array('interval' => intval($announce_interval), 'peers' =>
$peers)));
```

Додаток Ж. Код scare анонсера

```
function errorexit($reason) {
        exit(bencode(array('failure reason' => $reason)));
}

$scrape_interval = rand(3600, 10800);
```

```
if (empty($_GET['info_hash'])) {
	$hashes = array();
} else {
	parse_str(str_replace('info_hash=', 'info_hash[]=', $_SERVER['QUERY_STRING']), $array);
	$hashes = $array['info_hash'];
}

$files = array();
foreach ($hashes as $hash) {

	$hash_escaped = $hash;

	$complete = 0;
	$incomplete = 0;
	$downloaded = 0;

	$query = dbquery("SELECT info_ip, info_port, info_left, info_expire FROM ".$db_table_bt." WHERE info_hash = '".strToHex($hash_escaped)."'") or errorexit('problem query a2');
	while ($data = dbarray($query)) {
		if ($data['info_left'] == 0 && $data['info_expire'] > time()) $complete++;
		if ($data['info_left'] > 0 && $data['info_expire'] > time()) $incomplete++;
		if ($data['info_left'] == 0) $downloaded++;
	}
```

```
        $files[$hash] = array('complete' => $complete, 'incomplete' => $incomplete, 'downloaded' => $downloaded);
}

$db_connect->close();

exit(bencode(array('files' => $files, 'flags' => array('min_request_interval' => intval($scrape_interval)))));
```

Додаток З. Клас для опрацювання даних з БД в блокчейн

```
class Bitcoin
{
    private $username;
    private $password;
    private $proto;
    private $host;
    private $port;
    private $url;
    private $CACertificate;

    public $status;
    public $error;
    public $raw_response;
    public $response;

    private $id = 0;
```

```
    public function __construct($username, $password, $host = 'localhost', $port
= 8332, $url = null)
    {
        $this->username      = $username;
        $this->password      = $password;
        $this->host          = $host;
        $this->port          = $port;
        $this->url           = $url;

        $this->proto         = 'http';
        $this->CACertificate = null;
    }

    public function setSSL($certificate = null)
    {
        $this->proto         = 'https';
        $this->CACertificate = $certificate;
    }

    public function __call($method, $params)
    {
        $this->status       = null;
        $this->error        = null;
        $this->raw_response = null;
        $this->response     = null;

        $params = array_values($params);
```

```
        $this->id++;

        $request = json_encode(array(
            'method' => $method,
            'params' => $params,
            'id'     => $this->id
        ));

        $curl    = curl_init("{$this->proto}://{$this->host}:{$this->port}/{$this->url}");
        $options = array(
            CURLOPT_HTTPAUTH       => CURLAUTH_BASIC,
            CURLOPT_USERPWD        => $this->username . ':' . $this->password,
            CURLOPT_RETURNTRANSFER => true,
            CURLOPT_FOLLOWLOCATION => true,
            CURLOPT_MAXREDIRS      => 10,
            CURLOPT_HTTPHEADER     => array('Content-type: application/json'),
            CURLOPT_POST           => true,
            CURLOPT_POSTFIELDS     => $request
        );

        if (ini_get('open_basedir')) {
            unset($options[CURLOPT_FOLLOWLOCATION]);
        }

        if ($this->proto == 'https') {
```

```
            if (!empty($this->CACertificate)) {
                $options[CURLOPT_CAINFO] = $this->CACertificate;
                $options[CURLOPT_CAPATH] = DIRNAME($this->CACertificate);
            } else {
                $options[CURLOPT_SSL_VERIFYPEER] = false;
            }
        }

        curl_setopt_array($curl, $options);

        $this->raw_response = curl_exec($curl);
        $this->response     = json_decode($this->raw_response, true);

        $this->status = curl_getinfo($curl, CURLINFO_HTTP_CODE);

        $curl_error = curl_error($curl);

        curl_close($curl);

        if (!empty($curl_error)) {
            $this->error = $curl_error;
        }

        if ($this->response['error']) {
            $this->error = $this->response['error']['message'];
        } elseif ($this->status != 200) {
            switch ($this->status) {
```

```
                case 400:
                    $this->error = 'HTTP_BAD_REQUEST';
                    break;
                case 401:
                    $this->error = 'HTTP_UNAUTHORIZED';
                    break;
                case 403:
                    $this->error = 'HTTP_FORBIDDEN';
                    break;
                case 404:
                    $this->error = 'HTTP_NOT_FOUND';
                    break;
            }
        }

        if ($this->error) {
            return false;
        }

        return $this->response['result'];
    }
}
```

Додаток И. Акти про впровадження та дослідне випробовування результатів дисертації

"ЗАТВЕРДЖУЮ"
Проректор з науково-педагогічної роботи
Національного університету
"Львівська політехніка"
_____**О.Р.Давидчак**
"___" _____ 2022 р.

А К Т

про впровадження в навчальний процес результатів
дисертаційної роботи

Кучковського Володимира Володимировича

Цей акт складено про те, що результати дисертаційної роботи Кучковського Володимира Володимировича впроваджено у навчальний процес кафедри "Системи штучного інтелекту " Національного університету "Львівська політехніка".

Впровадження результатів дисертаційної роботи полягає в їхньому використанні при викладанні навчальних дисциплін як окремих розділів лекційних курсів, так і в циклах лабораторних робіт.

Зокрема для викладання дисципліни «Хмарні технології» для студентів освітньо-кваліфікаційного рівня «бакалавр», що навчаються за напрямом 122 "Комп'ютерні науки" використано такі результати:

- загальні принципи роботи хмарних технологій;
- використання блокчейн технологій в розробці хмарних технологій.

У лекційному курсі «Хмарні технології» для студентів кваліфікаційного рівня «бакалавр», що навчаються за напрямом 122 "Комп'ютерні науки", використано такі результати:

- метод запису і читання даних в блокчейні;
- демонстрація використання блокчейна у висконавантажуваних проектах.

Директор ІКНІ,
д.т.н., професор М.О. Медиковський

Завідувач кафедри СШІ,
д.т.н., професор Н.Б.Шаховська

Доцент кафедри СШІ,
к.е.н., доцент Н.І. Бойко

"ЗАТВЕРДЖУЮ"
Проректор з наукової роботи
Національного університету
«Львівська політехніка»
_____ І.В.Демидов
_____ 2022 р.

АКТ
використання наукових результатів
дисертаційної роботи Кучковського Володимира Володимировича,
представленої на здобуття наукового ступеня доктора філософії

Комісія в складі: голови комісії - начальника науково-дослідної частини д.т.н., с.н.с Небесного Р.В. та членів комісії - завідувача кафедри СШІ Шаховської Н.Б., професора кафедри СШІ Яковини В.С., доцента кафедри СШІ Хавалка В.М., доцента кафедри СШІ Кривенчука Ю.П. цим актом підтверджують, що результати дисертаційної роботи Кучковського В.В., зокрема

- метод запису Великих даних у блокчейн;
- метод перевірки якості внесених даних;
- інформаційна модель Великих даних та їх послідовного опрацювання

використано у науково-дослідних роботах фінансованих Міністерством освіти і науки України, що виконувались на кафедрі систем штучного інтелекту і включено до звіту: «Інформаційна технологія формування психофізичного портрету в умовах стресових ситуацій» (№ держ. реєстру 0119U002257).
Отримані автором результати використано:

- при розробленні систем зав'язаних з даними;
- при розробленні засобів для перевірки якості даних;
- при розробленні моделей даних і їх опрацювання.

Голова комісії:

начальник
науково-дослідної частини
д.т.н., с.н.с Р.В.Небесний

Члени комісії:

завідувача кафедри СШІ Н.Б.Шаховська
доцент кафедри СШІ М.І.Мельникова

доцент кафедри СШІ В.М.Хавалко

доцент кафедри СШІ Ю.П.Кривенчук

"ЗАТВЕРДЖУЮ"
Проректор з наукової роботи
Львівського Національного
Медичного університету
ім. Д.Галицького
д.мед.н., проф. Наконечний А.Й.
_____
_____ 2022 р.

АКТ

про впровадження результатів дисертаційної роботи

аспіранта кафедри «Системи штучного інтелекту»

Національного університету «Львівська політехніка»

Кучковського Володимира Володимировича

Ми, нижчепідписані члени комісії: завідувач кафедри хірургії та трансплантології ФПДО к.мед.н, доц. Щур О.О.

доц. Бігар В.Т.

доц. Маріна В.Н.

склали даний акт про те, що результати дисертаційної роботи Кучковського В.В. були впроваджені у навчальний та лікувальний процеси на кафедрі хірургії та трансплантології ФПДО, зокрема:

- метод опрацювання даних в блокчейні та пошуку цих даних на основі ключа для швидкого доступу до них. Розроблений унікальний алгоритм для хешування цих даних для більш кращого їх захисту. Це допомогло опрацьовувати дані по певним особливостям та категоріям беручи за основу ключі, в яких ключ виступає категорією, а в категоріях містяться дані.
- модель великих даних, які можна записувати в інформаційну технологію блокчейн для оптимізації баз даних та кількості запитів для висконавантажуваного проекту. Дана модель дозволяє класифікувати та кластеризувати дані, для їх швидкого опрацювання на функції вставки і вибірки. Це дало змогу зберігати вже посортовані та оброблені дані не в базі даних, а на вузлах блокчейну для кращого захисту і отримання цих даних виключно з вузлів, а не центральної бази даних.

Голова комісії доц. Щур О.О

Члени комісії