НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені Ігоря СІКОРСЬКОГО»
НАВЧАЛЬНО-НАУКОВИЙ ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра математичних методів захисту інформації

«До захисту допущено»
В.о. завідувача кафедри
_____ Сергій ЯКОВЛЄВ
«__» _____ 2022 р.

Магістерська дисертація

на здобуття ступеня магістра
за освітньо-науковою програмою
«Математичні методи криптографічного захисту інформації»

зі спеціальності: 113 Прикладна математика
на тему: **«Побудова загальної моделі STARK-доведень для підтвердження коректності отриманих результатів»**

Виконав: студент 2 курсу, групи ФІ-02мн
Бещук Андрій Андрійович

Керівник: д.т.н., проф., проф. каф. ММЗІ Ковальчук Л.В. _____

Консультант: _ _____

Рецензент: к.т.н., доц. каф. ММАД Орєхов О.А. _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент _____

Київ — 2022

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені Ігоря СІКОРСЬКОГО»
НАВЧАЛЬНО-НАУКОВИЙ ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра математичних методів захисту інформації

Рівень вищої освіти — другий (магістерський)
Спеціальність — 113 Прикладна математика,
ОНП «Математичні методи криптографічного захисту інформації»

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ Сергій ЯКОВЛЄВ
«__» _____ 2022 р.

ЗАВДАННЯ
на магістерську дисертацію

Студент: Бещук Андрій Андрійович

1. Тема роботи: «*Побудова загальної моделі STARK-доведень для підтвердження коректності отриманих результатів*»,

науковий керівник: д.т.н., проф., проф. каф. ММЗІ Ковальчук Л.В.,

затверджені наказом по університету №__ від «__» _____ 2022 р.

2. Термін подання студентом роботи: «__» _____ 2022 р.

3. Об'єкт дослідження: *процес часткового захисту персональних даних при виконанні автентифікації та авторизації*

4. Предмет дослідження: *STARK-протоколи та побудова загальної моделі та алгоритму інтерактивного та неінтерактивного STARK-протоколу*

5. Перелік завдань:

1) аналіз можливість застосування STARK-доведення при визначені коректності певних обчислень, зокрема, при доведенні того, що ці обчислення були виконані саме тою особою, яка є легітимною для виконання цих обчислень;

2) формулювання загального вигляду протоколу STARK-доведення, який може застосовуватись не тільки до тих моделей, які описані у

літературі у відповідних прикладах, а для будь-яких обчислень, які можна задати у вигляді виконуваного сліду.

6. Орієнтовний перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо): *«Презентація доповіді»*

7. Орієнтовний перелік публікацій: *планується доповідь на всеукраїнській конференції*

8. Дата видачі завдання: 10 вересня 2021 р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання	Примітка
1	Узгодження теми роботи із науковим керівником	01-15 вересня 2021 р.	Виконано
2	Огляд опублікованих джерел за тематикою доведень без розголошення і доведення цілісності обчислень	Вересень-жовтень 2021 р.	Виконано
3	Аналіз необхідних теоретичних відомостей і властивостей для побудови STARK-доведень	Жовтень-Березень 2022 р.	Виконано
4	Побудова загального алгоритму для інтерактивних і неінтерактивних STARK-доведень	Березень-Червень 2022 р.	Виконано

Студент _____ Бещук А.А.

Керівник _____ Ковальчук Л.В.

РЕФЕРАТ

Кваліфікаційна робота містить: 82 стор., 3 рисунки, 0 таблиць, 86 джерел.

Метою даною роботи є аналіз існуючих підходів до доведень без розголошення, зокрема, STARK-доведень, та побудова загальної моделі інтерактивного та неінтерактивного STARK-доведення. Об'єктом дослідження. Об'єктом дослідження є процес захисту персональних даних при виконанні автентифікації та авторизації. Предметом дослідження є STARK-протоколи та побудова загальної моделі та алгоритму STARK-протоколу.

Були розглянуті різні приклади застосування STARK-доведень, був побудований загальний вигляд STARK-протоколу, його алгоритм, проаналізовані умови, коли можна створити STARK-доведення, та була оцінений час його роботи та довжина доведення.

ДОВЕДЕННЯ БЕЗ РОЗГОЛОШЕННЯ, АНОНІМНІСТЬ, STARK, БЛОКЧЕЙН, АНОНІМНІСТЬ ТРАНЗАКЦІЙ

ABSTRACT

Qualification work consists of: 82 pages, 3 pictures, 0 tables, 86 sources.

The aim of this work is to analyze existing approaches to evidence without disclosure, in particular, STARK-evidence, and to build a general model of interactive and non-interactive STARK-proof. The object of study. The object of research is the process of personal data protection during authentication and authorization. The subject of the research is STARK-protocols and construction of the general model and algorithm of STARK-protocol.

Various examples of the use of STARK proofs were considered, a general view of the STARK protocol, its algorithm, the conditions under which STARK proofs can be created were analyzed, and the time of its operation and the length of proofing were estimated.

ZERO KNOWLEDGE PROOF, ANONYMITY, STARK, BLOCKCHAIN, ANONYMITY OF TRANSACTIONS

ЗМІСТ

ВСТУП

У цій роботі проводиться якісний і порівняльний аналіз різних методів забезпечення анонімності у найпоширеніших доведеннях без розголошення.

Будуть отримані результати у вигляді першого у світі формалізованого загального вигляду інтерактивних та неінтерактивних доведень STARK-протоколу, запропонованого алгоритму та оцінки його складності.

Актуальність дослідження визначається у бажанні, часто навіть необхідності, більшості людей та бізнесів зберігати анонімність своїх фінансів. Зокрема, виконанні нікому не підконтрольних та анонімних транзакцій в умовах повної недовіри.

Для цього є інструмент у вигляді доведень без розголошення, зокрема, STARK-доведення. Проте, на жаль, на даний час не сформульовано умов, за яких ці доведення будуть стійкими. Не зроблений аналіз таких умов, а також стосовно STARK-доведень не сформульовано якогось загального підходу до STARK-доведень. Не сформульовано, як такі доведення будуть виглядати у загальному випадку, та не існує його загального алгоритму, який можна напряму застосовувати для будь-якого виконуваного сліду. Причому, якщо загальну модель SNARK-доведень побудувати неможливо, бо вони всі базуються на різних математичних особливостях і підходах, то STARK на даний час доволі однотипні із точки зору математики й підходу. Завдяки цьому вдалося побудувати загальну модель STARK-доведень.

Тому у розгляді цих питань і є актуальність даного дослідження.

Метою дослідження є побудова загальної моделі та алгоритму інтерактивного та неінтерактивного STARK-доведення, та аналіз часу його роботи та аналіз довжини доведення. Для досягнення мети необхідно розв'язати такі часткові завдання:

1) розглянути та проаналізувати існуючі приклади використання STARK-доведень, та сформулювати умови, коли до деяких обчислень можна застосувати протоколи STARK-доведення їх коректності;

2) сформулювати послідовність дій та алгоритми для формування інтерактивних та неінтерактивних протоколів STARK-доведень;

3) побудувати оцінки часу роботи таких доведень та довжини цих доведень.

Об'єктом дослідження у даній роботі є процес часткового захисту персональних даних при виконанні автентифікації та авторизації.

Предметом дослідження є STARK-протоколи та побудова загальної моделі та алгоритму інтерактивного та неінтерактивного STARK-протоколу.

При розв'язанні поставлених завдань використовувались такі *методи дослідження*: абстрактна алгебра, теорія скінченних полів, теорія кодування.

Наукова новизна отриманих результатів: вперше визначені умови для побудови STARK-доведень, та побудовані загальні алгоритми для інтерактивних та неінтерактивних STARK-доведень, а також дана оцінка часу роботи й довжині доведення.

Практичне значення. Отримані результати дозволяють:

1) проаналізувати можливість застосування STARK-доведення при визначені коректності певних обчислень, зокрема, при доведенні того, що ці обчислення були виконані саме тою особою, яка є легітимною для виконання цих обчислень;

2) сформульовати загальний вигляд протоколу STARK-доведення, який може застосовуватись не тільки до тих моделей, які описані у літературі у відповідних прикладах, а для будь-яких обчислень, які можна задати у вигляді виконуваного сліду.

1 АНАЛІЗ СУЧАСНОГО СТАНУ У ГАЛУЗІ ПОБУДОВИ ДОВЕДЕНЬ БЕЗ РОЗГОЛОШЕННЯ ТА ЇХ МАСШТАБОВАНОСТІ

Масштабована верифікація цілісності обчислень з участю конфіденційних наборів даних. Задачу цієї роботи найкраще можна зобразити на наступному прикладі: нехай є державна база даних профілів ДНК D, і поліція P, яка стверджує, що в базі даних D відсутній профіль ДНК p деякого кандидата в президенти, який підозрюється у корупції. Чи можуть криптографічні протоколи переконати сумнівних громадян у чинності даного твердження, не розкриваючи повністю D або p, і не покладаючись на будь-яку третю довірену сторону (таких як Головний суддя) і з «достатніми» обчислювальними ресурсами?

1.1 Задача доведення цілісності обчислень

Приклад *відповідності профілю ДНК* (ВПД) є окремим випадком більш загальної задачі. Сторона P, що виконує обчислення C із використанням набору даних D, може мати намір повідомити неправильний результат $C(D)$. Це створює так звану задачу про цілісність обчислень (ЦО) – верифікація, що P дійсно повідомляє $C(D)$, а не більш сприятливий для P результат. Коли набір даних D є відкритим, то будь-яка зацікавлена в перевірці ЦО сторона V може повторно виконати обчислення $C(D)$ та порівняти отриманий результат із результатом, який повідомляє P. Тобто зробити так само як і клієнт ресторану може самостійно перерахувати чек, або як новий вузол Біткойна може перевірити свій блокчейн [1]. Проте, це інтуїтивно просте рішення не *масштабується*, оскільки час, який витрачає верифікатор T_V такий самий, як і час, необхідний для виконання програми T_C, і при

цьому V повинен ще також повністю прочитати набір даних D. Тут і надалі ми будемо використовувати термін *комітменти* (англ. – commitment) – це деякі числові значення, які залежать від певних величин, і за цими числовими значеннями не можна дізнатися ті величини, від яких вони залежать, і не можна підробити. Наприклад, ярким прикладом комітмента є хеш-функції. Якщо ми запишемо хеш-функцію від деякої величини, то ось цю величину (тобто, прообраз хеш-функції) і змінити не можна, і обчислити не можна. Схеми комітментів, які засновані на криптографічних хеш-функціях, зазвичай використовуються саме для обчислення короткого незмінного «відбитка пальця» cm_t в момент часу t для великого набору даних D_t [2]. Як правило cm_t має невелику довжину[1] у порівнянні з D_t, і може бути легко опублікований в блокчейні у якості відкритих даних. Таким чином, рішення для задачі по ЦО повинно мати масштабовану верифікацію, в якій час перевірки та час обміну даних масштабуються приблизно як $\log T_C$ та $|cm_t|$ (бітова довжина cm_t), а не як T_C та $|D_t|$. Як найменше час перевірки та час обміну даних мають бути строго менше ніж T_C та $|D_t|$ відповідно.

Коли набір даних D містить конфіденційні дані, просте перерахування $C(D)$ також не може бути застосовано, і сторона P, відповідальна за D, може приховувати порушення цілісності обчислень. Найбільш поширені методи розв'язання задачі ЦО із використанням конфіденційних даних зазвичай покладаються на деяку «довірену сторону», наприклад, аудитора або бухгалтера, щоб перевіряти чинність обчислень від імені громадськості. Це рішення все ще не пропонує масштабування, як і в тому випадку, коли дані є загальнодоступними. Погіршує проблему ще те, що ці методи вимагають від громадськості довіряти третій стороні. Це створює потенційну єдину точку збою в протоколі, оскільки ця третя сторона – наскільки це може бути обумовлено – може бути порушена, підкуплена або по примушена

[1]Зазвичай cm_t – це SHA2-хеш від Dt, який має довжину у 256 бітів для будь-якого набору даних.

зловмисними сторонами.

Системи доведення та аргументації з нульовим знанням (НЗ) — це автоматизовані протоколи, які замінюють людей-аудиторів для задач гарантування цілісності обчислень із використанням конфіденційних даних, та які використовуються для будь-яких швидких обчислень, усуваючи можливість впливу на результат та зниження обчислювальних витрат [3]. Система НЗ S для деякого обчислення C – це пара імовірнісних алгоритмів $S = (P,V)$. Тут і надалі сторону, яка доводить твердження, будемо позначати P. P – це алгоритм, що використовується для підтвердження цілісності обчислень, а V – це верифікатор, який перевіряє таке обчислення. Повнота та вагомість S означають, що P може ефективно довести всі будь-яке правильне твердження, але не зможе переконати V у хибному (з імовірністю, близькою до одиниці). Найперші теоретичні конструкції систем НЗ із можливістю масштабованих верифікацій для загальних обчислень обговорювалися на початку 1990-х років, і були засновані на імовірнісних доведеннях (ІД), тобто таких, чинність яких верифікується із деякою імовірністю. Відома теорема ІД [4, 5, 6, 7] пропонує компроміс між часом роботи, витраченим на побудову доведення T_P стороною P і часом роботи, який витрачає верифікатор V на перевірку (T_V). Цей компроміс означає, що час доведення збільшується поліноміально в порівнянні із часом прямого обчислення, тобто, $T_P = T_C^{O(1)}$; тоді як час перевірки зменшується експоненційно щодо того ж самого прямого обчислення, тобто $T_V = log^{O(1)}T_C$.

Система НЗ, заснована на теоремі ІД (НЗ-ІД) [8, 9, 10, 11, 12], має три додаткові переваги, які необхідні для постійної довіри суспільства до цілісності обчислень:

1) припущення, на яких ґрунтується безпека цих конструкцій:

а) існування стійких до колізій хеш-функцій [8] для інтерактивних рішень та загального доступу до випадкової функції («випадкова модель оракула» [13]) для неінтерактивних рішень [9];

б) невразливість до атак з боку великомасштабних квантових комп'ютерів; такі рішення називаються постквантовими безпечними (через очікуване збільшення кількості квантових комп'ютерів, а також через заклики про необхідність постквантових криптографічних протоколів від таких закладів, як, наприклад, USA NIST [14], особливо підкреслюється важливість постквантового безпечного рішення по системам НЗ);

2) НЗ-ІД є системою доведення знань (*СДЗ*) або, якщо вона реалізована, як описано вище, системою аргументу знання (*САЗ*) [2, 15]. Неофіційно, у контексті прикладу ВПД, НЗ-САЗ виступає доведенням, яке переконує громадськість, що поліція використала «справжній» набір даних D_t та профіль ДНК кандидата в президенти p, комітмент якого було оголошено раніше;

3) НЗ-ІД є прозорими (або «публічно випадковою») – це означає, що випадковість, яку використовує V, є загальнодоступною; зокрема, налаштування НЗ-ІД не вимагає зовнішнього довіреного етапу налаштування, на відміну від новіших рішень НЗ, включаючи такі, які використовуються криптовалютою Zcash; прозорість є важливою властивістю для постійної публічної довіри, оскільки вона серйозно обмежує здатність зловживати системою навіть наймогутніших зі сторони P, і, таким чином, прозорим системам громадськість може довіряти, до поки існує щось непередбачуване у спостережуваному всесвіті.

Підсумовуючи, НЗ-ІД є чудовим методом для забезпечення довіри суспільства до ЦО із залученням конфіденційних даних, який характеризується шістьма основними якостями:

1) прозорість;

2) універсальність – тобто, може бути застосовано до будь-якого ефективного обчислення C, навіть якщо воно вимагає допоміжного (і, можливо, конфіденційного) введення набору даних, як D_t;

3) конфіденційність – тобто, не ставить під загрозу допоміжні входи, такі як D_t;

4) пост-квантово безпечні;

5) доведення/аргументація наявності знання;

6) масштабована верифікація.

Хоч НЗ-ІД і відомі з середини 1990-х років, досі жодна з них не була реалізована в якійсь мові програмування, оскільки, за словами недавнього опитування [16], «доведення, що випливають із теореми ІД (попри асимптотичні вдосконалення), були настільки довгими та складними, що знадобилися б тисячі років, щоб їх створити та верифікувати, і знадобилося б більше біт пам'яті, ніж атомів у Всесвіті». Отже, нещодавні зусилля щодо реалізації систем НЗ для загальних обчислень були зосереджені на альтернативних методах, які не володіють всіма одразу зазначеними вище властивостями 1)-6), хоча і деякі з них надзвичайно ефективні на практиці для конкретних розмірів схем і для амортизованих обчислень.

Інтерактивні Доведення з Оракулом (ІДО) із масштабованістю доведень. Щоб покращити масштабованість доведень без шкоди для властивостей 1)–6), нещодавно було запропоновано нову модель [17, 18] – так звані інтерактивні доведення із оракулом (ІДО). Вони є узагальненням моделей інтерактивних доведень, ІД та інтерактивних ІД (ІІД) [19]. Як і в налаштуваннях системи ІД, V у системі ІДО не повинен читати всі повідомлення P, а скоріше може робити запити на їх отримання у випадкових місцях. Як і в налаштуваннях інтерактивних доведень, P і V взаємодіють протягом кількох раундів. Як і у випадку з НЗ-ІД, система НЗ-ІДО може бути перетворена в інтерактивну САЗ, припускаючи сімейство стійких до колізій хеш-функцій, і може бути перетворена в неінтерактивний аргумент знань у моделі оракула [17], який зазвичай реалізується за допомогою стандартної хеш-функції. Як і суворе узагальнення інтерактивних доведень/ІД/ІІД, модель ІДО має ряд переваг. Найбільш релевантним для цієї роботи є покращена можливість розширення ІДО. Ця перевага виконується як асимптотично – оскільки вхідний розмір

$n \to \infty$ ([20, 21]), – так і для конкретних вхідних довжин, які виникають на практиці. Виходячи з цієї ефективності, нещодавно було повідомлено про реалізацію доведення концепції щодо ІДО під кодовою назвою SCI [22]; однак SCI не має НЗ, а довжина конкретного доведення та час доведення все ще досить великі. Нещодавно були описані ІДО із (ідеальним) НЗ і масштабованими верифікаторами. Спочатку для задач типу NP, а потім для NEXP. В обох роботах час роботи P T_P обмежена $T_C log^{O(1)} T_C$. Це називається масштабованістю часу доведення (також відомим як квазілінійний час доведення).

Відтепер (універсальна) система НЗ (тільки пункт 6)) буде називатися повністю масштабовано або, просто масштабованою, якщо час роботи P та час роботи V є масштабованими. Це виправдано, тому що часи роботи для обох є майже оптимальними, аж до полілогарифмічних факторів. Система НЗ-ІДО, що задовольняє властивостям 1)-5) і властивості повної масштабованості 6), буде називатися Масштабованим та Прозорим ІДО знань (НЗ-МПІДО). Підсумовуючи, теоретичні конструкції систем НЗ-МПІДО були представлені нещодавно, але їх конкретна ефективність і застосовність до «практичних» обчислень досі не продемонстровані.

1.2 Основний внесок

У цьому розділі представлена нова конструкція (подвійно) масштабованої та прозорої системи НЗ в моделі ІДО (НЗ-МПІДО). Ця конструкція створюється як система НЗ із масштабованим та прозорорим доведенням (НЗ-МПД), до якої застосовується її «значуще» обчислення від доведення концепції, які мають дуже послідовний характер – представлений проблемою ВПД раніше. Ця реалізація досягає наступного:

1) час перевірки є суворо меншим, ніж наявний час роботи ($T_V < T_C$);

2) складність обміну даними є суворо меншою за розмір свідка.

Основною інновацією та основним джерелом покращення

продуктивності цієї системи є розширена опора на моделі ІДО, в т.ч. швидкого протоколу ІДО близькості (ІДОБ) Ріда-Соломона та нова процедура арифметизації. Підкреслюємо, що експоненціальне зменшення часу перевірки та розміру свідку, описані далі (і відображені на рисунку 1.1), застосовуються до будь-яких обчислень, визначених для довільно великого розміру свідка, хоча конкретна точка, в якій це прискорення матеріалізується, залежить від складності обчислення.

Обчислення відповідності профілю ДНК.

Як доведення концепції «змістовного» обчислення була створена НЗ-МПД для задачі відповідності профілю ДНК (ВПД), яка неформально описана далі. Це обчислення стосується наступного гіпотетичного сценарію: припустимо, що поліція (що діє як P) відповідає за державну базу даних криміналістичних профілів ДНК D, і в попередній момент t публікує (скажімо, у блокчейні) приховані комітменти cm_t до стану D_t бази даних на цей момент часу. Поліція тепер стверджує, що профіль ДНК p кандидата в президенти, який незабаром буде призначений і імовірно корумпований, не відображається в D_t, і таким чином вона хоче створити у масштабований спосіб доведення, яке переконає громадськість в тому, що обчислення ВПД було проведено правильно, а результати, про які повідомляє поліція також є правильними (із відповідними p і D_t).

Найпоширенішим стандартом для профілів ДНК, який використовується в більш ніж 50 країнах, є формат комбінованої системи індексу ДНК (КСІДНК). Згідно із цим стандартом індивідуум представлений кількістю коротких тандемних повторів (КТП) його/її ДНК, які були виміряні для набору з 20 «коротких локусів» [23] (кількість центральних локусів зросла з 13 до 20 з січня 2017 р.). Передбачається, що комітмент cm_t до стану D_t бази даних КСІДНК є загальнодоступною інформацією (скажімо, опублікованою в момент t в блокчейні), як і комітмент cm_p для профілю p кандидата в президенти. Припускається, що p було отримано незалежною лабораторією, яка

передала його (конфіденційно) поліції, одночасно опублікувавши cm_p. Припустимо, що поліція заявляє

«Значення α є результатом пошуку відповідності для профілю з комітментом cm_p у базі даних з комітментом cm_t»

Відповідь α буде однією з трьох можливих: «не збігається», «частково збігається» або «повністю збігається». Загальнодоступне (з відкритим вихідним кодом) обчислення C – це обчислення, яке було б виконано довіреною третьою стороною, яка верифікує наведене вище твердження. Для цього обчислення потрібні три загальнодоступні вхідні змінні: cm_t, cm_p, A; і дві конфіденціальні змінні: база даних профілю ДНК D' та індивідуальний профіль ДНК p'. Обчислення C завершується успішно тоді й тільки тоді, коли загальнодоступні вхідні дані (cm_t, cm_p, A) та конфіденційні (D', p') задовольняють наступним трьом умовам:

1) комітмент cm' конфіденційного входу D' дорівнює загальнодоступному входу cm_t;

2) комітмент $cm_p^{'}$ конфіденційних вхідних даних p' дорівнює загальнодоступному введеному cm_p;

3) результат пошуку відповідності для конфіденційного входу p' у конфіденційному наборі даних D' призводить до публічно оголошеного результату α.

Нехай $|D(n)|$ позначаємо довжину бітів набору даних $D(n)$, який містить n профілів (кожен профіль має довжину 40 байтів). Нехай $СЗ(n)$ позначає складність зв'язку НЗ-МПД для $D(n)$, тобто загальну кількість бітів, переданих між P і V. Аналогічно, нехай $T_C(n)$ позначає час, необхідний для простої перевірки обчислення C на D із n записами. І нехай $T_V(n)$ позначає час, необхідний V для його перевірки (обидва виміряні на фіксованому фізичному комп'ютері.)

Обчислення часу та стиснення розміру свідка Будемо вважати обчислення C, яке вимагає допоміжного конфіденційного входу D, який

змінюється за розміром, як приклад ВПД. Будь-яка НЗ-система $S = (P,V)$ для C включає відповідну пару показників для оцінки для часу та розміру свідка:

$$\rho_{time}(n) = \frac{T_V(n)}{T_C(n)};\ \rho_{size}(n) = \frac{СЗ(n)}{|D(n)|} \tag{1.1}$$

Показники швидкості (і пороги, визначені далі) залежать від C і системи S, тому позначення $\rho_{time}^{(S;C)}$ було б точніше, але перевага віддається простоті позначень, і припускається, що C і S відомі.

Показник швидкості менше за 1 вказує на стиснення, тобто перевірка в S є більш ефективною, ніж пряма верифікація (пряме повторне обчислення). У повністю розширених системах НЗ складність перевірки для V є полілогарифмічною в рамках складності P. Тому в кінцевому підсумку, при достатньо великих n, система досягає стиснення. Основне твердження тут полягає в тому, що вперше демонструється стиснення часу та розміру свідку для НЗ-МПД для великомасштабного послідовного обчислення. Визначення порогу стиснення для найменшого значення n_0 та для всіх $n \geq n_0$, щоб швидкість була меншою за 1 проводиться наступним чином:

$$\theta_{time} = min(n_0 | \forall(n) \geq n_0, p_{time}(n) < 1); \tag{1.2}$$

$$\theta_{size} = min(n_0 | \forall(n) \geq n_0, p_{size}(n) < 1) \tag{1.3}$$

На рисунку 1.1 продемонстровані показники для задачі ВПД у подвійній логарифмічній шкалі. Час стиснення знаходиться на рівні $\theta_{time} = 2.8 \times 10^5$ і поріг розміру свідка дорівнює $\theta_{size} = 9 \times 10^3$. Найбільша база даних, для якої було створено доведення під час тестів мала розмір $n_{max} = 2^{20} \approx 1.14 \times 10^6$ профілів ДНК. Більші бази даних вимагають більше дискового простору та оперативної пам'яті, ніж було доступно при перших обчисленнях. Кожен профіль займає 40 байтів, тож $|D_{n_{max}}| \approx 43$.

Показник часу для n_{max} буде $\rho_{time}(n_{max}) = 1/6$ і показник розміру свідку становить $\rho_{size}(n_{max}) = 1/100$. Рисунок також демонструє, що стиснення покращиться, якщо воно підтримується більш потужним технічним обладнанням, ніж те, на якому були виконані поточні тести.

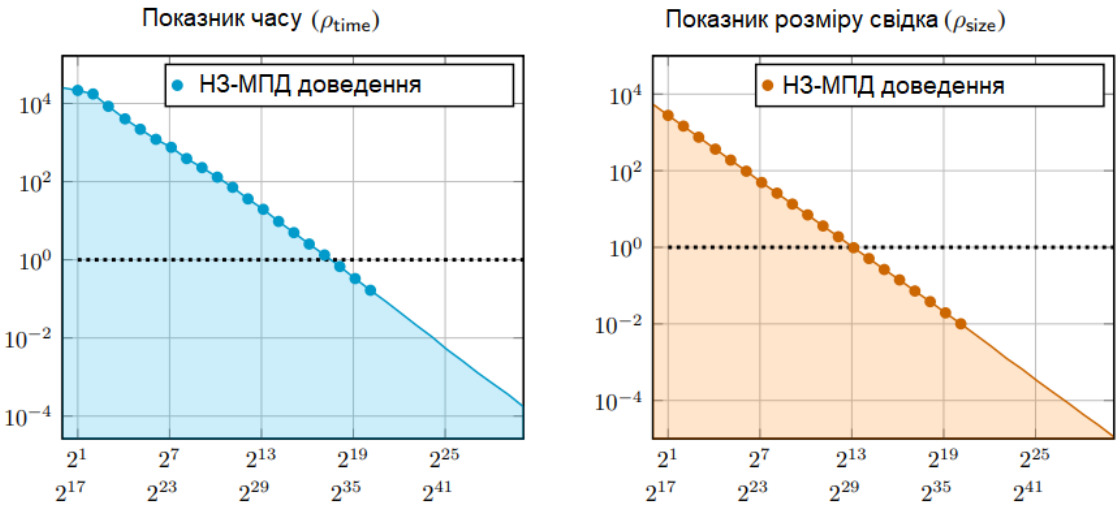

Рисунок 1.1 – Функції показників часу (ліворуч) і розміру свідка (праворуч) контрольного тесту ВПД НЗ-МПД як функція кількості записів (n) у базі даних (верхня горизонтальна вісь); та числа множення (нижня горизонтальна вісь). Розмір бази даних у байтах становить $40 \cdot n$, а обчислення одного профілю відповідає схемі із $\approx 2^{16}$ елементами множення. Складність обчислення для V не залежить від складності доведення, тому її вимірювали (на «стандартному» ноутбуці) навіть для значень n, більших за ті, для яких було створено доведення; значення, для яких було створено доведення, позначені повними колами.

1.3 Відкрита дискусія – застосування до децентралізованих суспільних функцій

Криптовалюти на чолі з біткойном підривають усталені фінансові системи, пропонуючи повністю децентралізовану грошову систему на заміну фіатній валюті. Гроші – це лише одна із соціальних функцій, яку можна децентралізувати, а також і юридичні контракти вже замінені автоматизованими смартконтрактами [24] в блокчейні Ethereum. Цей розділ буде завершено обговоренням двох очікуваних впливів систем НЗ-МПД на децентралізовані публічні бухгалтерські книги.

Масштабованість. Сьогодні в блокчейн відбувається бурхлива дискусія щодо правильного способу масштабування пропускної здатності транзакції без надмірного оподаткування часу та простору вузлів, які беруть участь у мережі. Як вперше вказав один зі співавторів [25] і це було підтримано кількома криптовалютними ініціативами [26, 27, 28], повністю масштабовані системи доведень (навіть без НЗ) можуть розв'язувати проблему масштабованості шляхом експоненціального зменшення часу перевірки. Мається на увазі, що один «вузол доведення» може генерувати за квазілінійний час доведення, який переконає всі інші вузли визнати дійсність поточного стану бухгалтерської книги, не вимагаючи від цих вузлів ані повторного виконання обчислень, ані зберігання всього стану блокчейну, який біг би знадобиться для такої перевірки.

Конфіденційність. Конфіденційність доведень НЗ вже використовується для покращення процесу ототожнення грошей і фінансової конфіденційності в криптовалютах. Протокол Zerocash [29] – який нещодавно реалізований у криптовалюті Zcash [30, 31] – використовує особливий тип доведень НЗ, які називаються стислими неінтерактивними доведеннями знань (НЗ-СНДЗ), заснованими на припущенні знання експоненти (ЗЕ) в термінах криптографії [32, 33].

Воно використовується для цілісного ведення децентралізованого реєстру, записи якого приховують комітмент невитрачених коштів. Ці НЗ-СНДЗ є непрозорими, оскільки вони вимагають «фази налаштування», яка використовує непублічну випадковість, яка в разі компрометації може бути використана для компрометації безпеки навіть всієї системи. З огляду на майбутнє, НЗ-МПД можуть замінити НЗ-СНДЗ і забезпечити взаємодію та конфіденційність Zcash, залишаючи властивість прозорості. Наразі НЗ-СНДЗ приблизно в 1000× разів коротші, ніж доведення НЗ-МПД, тому заміна НЗ-СНДЗ на МПД вимагає додаткових досліджень, щоб або скоротити довжину доведення, або об'єднати та стиснути декілька доведень НЗ-МПД за допомогою обчислень, які можна перевіряти частинами [34] (див. [69]).

1.4 Порівняння з іншими вже реалізованими універсальними системами НЗ

Останніми роками були помітні значні зусилля, щоб реалізувати в коді системи підтвердження НЗ із використанням різних теоретичних підходів, які відрізняються від підходів НЗ-МПД. Багато з цих систем перевершують НЗ-МПД для обчислень досить малого розміру, для паралельних обчислень малої глибини та/або для групових і амортизованих обчислень. Всі ці випадки надзвичайно корисні на практиці. З усім тим, для великомасштабних обчислень (особливо, для послідовних) покращення повної масштабованості запропонованого ІДО підходу стає більш помітною.

Далі буде коротко розглянуто різні підходи до реалізації, які є універсальними, тобто такими, які застосовуються до загальних обчислень і мов у множені NP. У роботах [16] та [22] описана додаткова інформація про рішення щодо забезпечення цілісності обчислень, включаючи ті, які не є універсальними та/або без НЗ.

Теоретичні дискусії. У великій кількості літератури щодо реалізацій систем НЗ у цій роботі потрібно обмежити сферу поточного обговорення і робити це трішки довільно, розглядаючи лише системи, які є НЗ, повними за Тьюрінгом і які були реалізовані в коді якоїсь мови програмування. Вони порівнюються для найбільш поширеного класу тверджень про цілісність обчислень і розглянуті з урахуванням наступних чотирьох властивостей:

1) асимптотична масштабованість P (квазілінійний час виконання);

2) асимптотична масштабованість V (полілогарифмічна верифікація);

3) прозорість;

4) постквантова захищеність.

Останню властивість можна замінити властивістю щодо покладання лише на стійкі до колізій хеш-функції[1].

– **Гомоморфна криптографія з відкритим ключем (ГКВК):** Цей підхід, започаткований Ishai та ін. [36] (для випадку «призначеного верифікатора») і Groth [37] (для випадку, що підлягає публічній верифікації), використовує ефективну теоретико-інформаційну модель під назвою «лінійний ІД», яка потім «компілюється» в криптографічну систему за допомогою ГКВК. Надзвичайно ефективний екземпляр, заснований на програмах квадратичного діапазону (ПКД), був представлений Gennaro та ін. [32] (див. [38, 39, 40, 41, 42, 43] для пов'язаних робіт та подальших удосконалень). Він служить, наприклад, системою доведення в Zerocash і Zcash. Перша реалізація системи на основі ПКД називається Pinocchio [44], а наступні реалізації включають libSNARK [33, 45], яка використовується в реалізаціях Zerocash і Zcash; додаткові реалізації з'являються в [46, 47, 48, 49, 50, 51, 52].

Теоретичні відмінності між ГКВК та НЗ-МПД полягають у прозорості та постквантовій безпеці – у ГКВК не вистачає обох властивостей. Час перевірки в ГКВК масштабується (тобто є полілогарифмічний у T_C) лише

[1] Це припущення охоплює лише інтерактивне налаштування

для обчислень, які повторюються багато разів, оскільки «фаза налаштування» ГКВК вимагає часу $\geq T_C$.

– **Задача дискретного логарифма (ЗДЛ):** Підхід, започаткований Groth [53] (пор. [54]) та реалізований у [72] покладається на міцність ЗДЛ для побудови прозорої системи. Алгоритм квантової факторизації Шора ефективно вирішує ЗДЛ, роблячи цей підхід квантово-вразливим. Крім того, складність V в підході ЗДЛ вимагає часу $\geq T_C$, отже, він не масштабується (згідно з запропонованим раніше визначенням терміна), хоча складність зв'язку в підході ЗДЛ є логарифмічною. Початкова реалізація цієї системи буде називатися BCCGP [72], а останню вдосконалену версію називають BulletProofs [56].

– **На основі інтерактивних доведень (ІД):** Протоколи ІД можна виконувати з НЗ [57], але тільки нещодавно протоколи ІД були ефективно «зменшені» до невеликих глибинних (непослідовних) обчислень за допомогою так званих «доведень для маглів» Goldwasser et al. [58, 18]. Це призвело до ряду реалізацій у коді. У ранніх роботах було відсутнє НЗ [59, 60, 61, 62], але в найсучасніших, як [63] та Hyrax [64], воно є.

Як і НЗ-МПД, ці нещодавні доведення на основі ІД є прозорими і мають масштабоване доведення, але є квантово-вразливими, а їх V не є масштабованим, оскільки він лінійно масштабується з часом обчислень для «стандартних» (тобто послідовних) обчислень (наприклад, він досить ефективний для пакетних і амортизованих обчислень і для малих схем).

– **Захищені багатосторонні обчислення (ЗБО):** Цей підхід, запропонований Ishai et al. [65] і реалізований спочатку в системі НЗВоо [66], а нещодавно в Ligero [67], «компілює» захищені протоколи ЗБО в системи НЗ-ІД, вимагаючи від P запису безпечного протоколу ЗБО, а розкрити точку зору однієї зі сторін.

Як і НЗ-МПД, доведення на основі ЗБО є прозорими, постквантовими безпечними та мають масштабований (квазілінійний) час доведення. Проте системи на основі ЗБО мають немасштабований V, який працює за

час $\geq T_C$, а складність зв'язку не масштабується, це $\sqrt{T_C}$ у сучасній системі [67]. Для конкретних схем і амортизованих обчислень він, однак, надзвичайно ефективний.

– **Поступова перевірка обчислення (ППО):** Цей підхід, запропонований Valiant [34] (пор. [68, 69]), зменшує використання простору для доведення, спираючись на припущення щодо вилучення знань. Цей підхід може бути застосований поверх інших систем доведення зі стислими (сублінійними) V, включаючи НЗ-МПД, але поки що був реалізований лише для однієї системи ГКВК [70].

У порівнянні з НЗ-МПД, системи, побудовані таким чином, успадковують більшість властивостей базової системи доведень. Зокрема, ППО на основі **ГКВК** є непрозорим і квантово-вразливим; однак V масштабується навіть для обчислень, що виконуються лише один раз, оскільки фаза налаштування виконується за полілогарифмічний час.

	масштабованість сторони, що доводить (квазілінійний час)	масштабованість верифікатора (полілогарифмічний час)	Прозорість (публічна випадковість)	Пост-квантова захищеність
ГКВК	Так	Тільки при повторюванних обчисленнях	Ні	Ні
ЗДЛ	Так	Ні	Так	Ні
ІД	Так	Ні	Так	Ні
ЗБО	Так	Ні	Так	Так
ППО + ГКВК	Так	Так	Ні	Ні
НЗ-СНДЗ	Так	Так	Так	Так

Рисунок 1.2 – Теоретичне порівняння універсальних (NP повних) реалізованих систем НЗ

1.5 Конкретні характеристики

Різні системи доведень НЗ засновані на різних криптографічних припущеннях і розроблені для різних обчислювальних задач. Їхні реалізації написані різними мовами програмування та перевірені на різноманітному обладнанні. Тому точні порівняння «яблука з яблуками» важко (якщо взагалі можливо) провести. Проте, у цьому розділі буде

спроба якісно порівняти реалізовані системи доведень, про які повідомлялося попередньо, з точки зору складності для V, P та зв'язку для обчислювальних задач, які за своєю природою подібні до ВПД.

Складність арифметичної схеми як стандартної вимірювальної бази. Усі реалізовані системи доведень, розглянуті тут (включаючи НЗ-МПД), використовують арифметизацію, щоб зменшити твердження про цілісність обчислень (ЦО) до тверджень про системи поліномів низького ступеня над скінченними полями. Усі інші досліджувані системи використовують прості поля $\mathbb{F}p$, хоча деякі (наприклад, на основі ЗБО та інтерактивних доведень) можуть працювати також над двійковими полями, як НЗ-МПД; підкреслюємо, що НЗ-МПД також може працювати над простими полями[1], але це не враховується в коді. Більшість систем (НЗ-МПД не мається на увазі) зводять оператори ЦО до арифметичних схем, тобто таких, які відповідають умовам у вигляді квадратичних поліномів. НЗ-МПД зводиться до систем поліномів вищого ступеня, наприклад, для еталона ВПД цей степінь дорівнює 8.

Складність арифметичної схеми є розумною метрикою для порівняння різних систем доведення. Основними параметрами, які впливають на складність системи доведення (і згадані в попередніх роботах), є глибина схеми, ширина схеми (кількість вентилів на кожному «рівні» схеми), недетермінований розмір свідка та складність множення, тобто кількість елементів множення (складність додавання також є актуальною, але більшість систем доведення менше впливає на неї.)

Обчислення ВПД відповідає арифметичній схемі з наступними параметрами, якщо його застосовано до бази даних з n записами:

– глибина контуру, де $depth_n = 62 \cdot n$;

– ширина контуру становить $w = 81$;

– складність свідка – це $wit_n = 40 \cdot n$ байтів;

– складність множення – це $mult_n = 1467 \cdot 62 \cdot n \approx 2^{16.4} \cdot n$.

[1]система FRI вимагає, наприклад, щоб p містила досить велику мультиплікаційну підгрупу порядку $2^{t+O(1)}$; таких простих полів багато, як випливає з теореми Лінніка [71].

Повна система НЗ-МПД ($P + V$) вимірювалися із 60 бітами безпеки для $n = 2^k$, $k = 1, \ldots, 20$ і так далі для арифметичних схем глибиною до $depth_n \approx 2^{25.9}$ і до $\approx 2^{36.4}$ множення виходу закінчується $\mathbb{F}_{2^{64}}$. Сам (неадаптивний) V був виміряний навіть для більшого входу, до $n = 2^{36}$.

Щоб спробувати порівняти одні системи методом «яблука до яблука» із іншими системами, було запущено кілька з них на одній машині (для вимірювання коду ВПД використовувалася інша машина) із використанням того ж контрольного обчислення на основі обчислення «достатньої підмножини», розглянутій в роботі [22]. Результати узагальнено на рисунку 1.3. Системи, які вимірювались:

– **libSNARK** (фіксація dc78fd, 7 вересня 2017 р.) із 80-бітною безпекою;

– **SCI** (ті самі вимірювання, що використовуються в [22]) 80-бітною безпекою;

– **BCCGP** із логарифмічною складністю зв'язку та 128-бітовою безпекою, однопотоковий (та сама система, що використовується в [72]);

– **Ligero** з 60 бітами безпеки (така ж система, що описана в [67]);

– **НЗ-МПД** з 60-бітною безпекою НЗ-МПД; оцінюється, що час роботи P для 80 біт буде довшим щонайбільше на 5%.

Тепер коротко обговоримо продуктивність цих (та інших попередніх робіт, про які повідомлялося), зосередившись на наступних параметрах складності: час доведення, час V та складність зв'язку.

Складність доведення. Майже всі системи, розглянуті раніше, мають складність доведення, яка лінійно або майже лінійно масштабується за розміром обчислень. Як показано на рисунку 1.3, P створює доведення у системі НЗ-МПД принаймні в 10 разів швидше, ніж інші виміряні системи в усьому діапазоні порівнюваних обчислень (усі системи були перевірені до максимального часу доведення 12 годин).

Складність V. Різні системи доведення відмінно підходять для різних топологій. Наприклад, Ligero досягає найкращої продуктивності для ланцюгів розміру s, які повторюються s разів (тобто, коли

$depth \approx w \approx \sqrt{mult}$), а Hyrax найкраще працює на ланцюгах малої глибини та масово паралельних ($depth = O(1)$ і w, $mult \gg depth$).

Для «глибоких» і «вузьких» ланцюгів, таких як ті, що виникають із ВПД, арифметична складність перевірки попередніх робіт масштабується принаймні як $\sqrt{mult}$ (і часто як $mult$), тоді як НЗ-МПД масштабується як $w + \log mult$. Отже, для середньо- та великомасштабних послідовних обчислень час перевірки НЗ-МПД кращий, ніж інші рішення, як показано середнім графіком на рисунку 1.3. Очікується, що порівняння з іншими роботами, такими як Hyrax і BulletProofs, буде поводитися аналогічно. Зокрема, швидкість P у Hyrax досягає $\approx$ 10 секунд для схеми з $\approx 2^{28}$ елементів (але виміряно на іншій машині); BCCGP і BulletProofs вимагають ще більшого часу роботи. Для порівняння, V НЗ-МПД для обчислення ВПД вимагає менше 50 мс (на іншій машині), навіть для величезних схем, з $n = 2^{36}$ записами (профілями), $wit_n \approx 2{,}5$ терабайтовими свідками та арифметичними схемами з $mult_n = 2^{52}$ множеннями і $depth \approx 2^{40}$.

Системи ГКВК, такі як Pinocchio і libSNARK, і системи ППО+ГКВК, такі як [70], відрізняються в цьому відношенні. Вони мають фазу попередньої обробки, яка виконується лише один раз на контур. Для Pinocchio і libSNARK час попередньої обробки зростає лінійно із розміром схеми. Наприклад, системі libSNARK потрібно $\approx$ 16 секунд для обчислення із 2^{20} елементами. Для системи ППО+ГКВК час попередньої обробки є постійним і не залежить від розміру схеми; однак ця константа досить велика в порівнянні з часом у НЗ-МПД V, вона становить $\approx$ 10 секунд для обчислень, подібних до ВПД.

Складність зв'язку (СЗ). Використання фази попередньої обробки в системах ГКВК і ППО+ГКВК призводить до надзвичайно малого СЗ постобробки. Система BCCGP також має надзвичайно коротку СЗ і, оскільки його попередня обробка є прозорою, його можна ефективно замінити коротким сідом на псевдовипадковий генератор. Таким чином, для всіх обчислень, виміряних на практиці, СЗ постобробки

Pinocchio, libSNARK і системи ППО+ГКВК становлять менше ніж 300 байтів, а BCCGP менш як 7 КБ [72] (див. також рисунок 1.3). Однак довжина ключа попередньої обробки лінійно масштабується з розміром схеми для ГКВК; система ППО+ГКВК відрізняється в цьому відношенні, вона має стислу довжину попередньої обробки навіть для великого розміру обчислень, але знову ж таки, ця довжина дуже велика – понад 40 МБ для обчислень, таких як ВПД.

Для Ligero складність зв'язку масштабується, як $70\sqrt{mult_n}$ елементів поля [67], а для Hyrax вона масштабується як $wit^{1/k} + 10 \cdot depth \cdot \log w$ елементів поля для довільного цілого k [64]. Збільшення k зменшує СЗ, але також збільшує час перевірки (який становить щонайменше $wit/(wit^{1/k})$). Використовуючи оцінку для Hyrax, швидкі розрахунки демонструють, що для схеми, яка виникає в результаті обчислення ВПД з, скажімо, $n = 2^{13}$ профілями, СЗ Hyrax досягне кількох мегабайтів, у порівнянні з НЗ-МПД СЗ, який становить менше ніж 1 мегабайта навіть для $n = 2^{36}$ профілів.

Висновки до розділу 1

Серед усіх систем НЗ, перевірених способом «яблука до яблука», описаним вище, НЗ-МПД створює доведення найшвидше для всіх розмірів ланцюгів, які були виміряні. Зокрема, вона в ≈ 10 разів швидше другої за швидкістю вимірюваної системи – libSNARK. Інші системи працюють краще (коротший зв'язок, швидша перевірка) на малих ланцюгах (НЗBoo, Ligero), ланцюгах малої глибини (Hyrax), а також на обчисленнях, повторюваних багато разів з тією ж фіксованою схемою (BulletProofs, Pinocchio, libSNARK). Однак для загальних великомасштабних обчислень НЗ-МПД має час верифікації та складність зв'язку, які перевершують усі інші прозорі системи, опубліковані на цей час для цього діапазону параметрів. Іншими словами, дана конкретна реалізація НЗ-МПД показує, що асимптотичні переваги повної масштабованості та прозорості проявляються вже для конкретних

обчислень, які є практично актуальними, як ВПД, і припускає, що наш тип системи потенційно корисний для побудови рішень масштабованості, наприклад, для децентралізованих криптовалют.

У більшості звичайних людей та бізнесів існує бажання, часто навіть необхідність, зберігати анонімність своїх фінансів. Тому з появою у 2009 році першої криптовалюти Bitcoin (BTC) [?] у багатьох людей та організацій виникли сподівання, що тепер вони отримають способи виконання швидких, зручних, нікому непідконтрольних та анонімних транзакцій.

Дійсно, концепція BTC містила багато революційних для того часу ідей та технологічних рішень, найсуттєвішим з яких є спосіб досягнення децентралізації в умовах повної недовіри. Але з моменту запуску BTC та до цього часу алгоритм функціонування мережі практично не змінювався. Зокрема, не було зроблено нічого для створення та втілення додаткових механізмів анонімності. Проте на базі платформи BTC було створено ряд інших криптовалют (аль-ткоїнів), які намагались тими чи іншими методами підвищити як анонімність користувачів, так і неможливість відстежувати обіг їх коштів.

Далі розглянуто основні способи забезпечення анонімності при користуванні різними криптовалютами та наведені їх переваги та недоліки.

Зазначимо, що анонімність отримувача коштів закінчується тоді, коли він виводить криптовалютні кошти на іменну банківську картку. Використання криптовалюти для оплати реальних товарів або послуг, як правило, вимагає її перетворення у валюту відповідної країни. Тому у цьому розділі в основному розглянуто способи збереження анонімності джерел, з яких криптовалюта надходить в електронний гаманець певної особи.

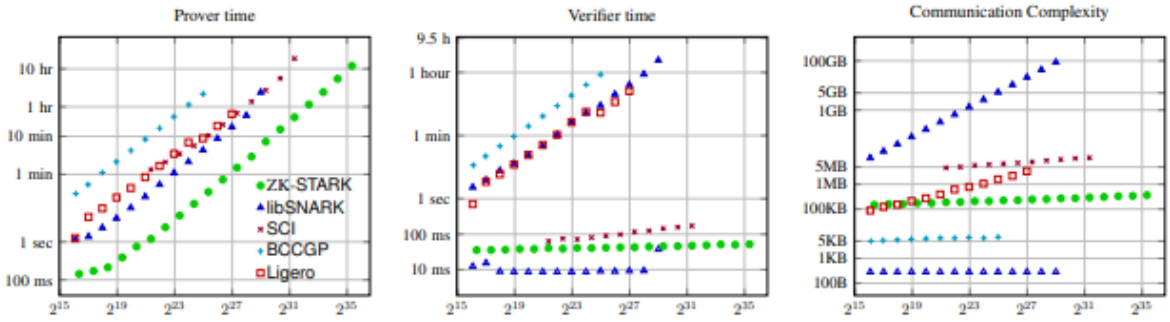

Рисунок 1.3 – Порівняння «яблука до яблука» різних реалізованих систем доведень як функція розміру обчислень, виміряного кількістю елементів множення. Усі системи були протестовані на одному сервері та виконано обчислення розміру та структури відповідно до програми «вичерпної суми підмножини» з [22]. Порівнюваними системами є SCI (фіолетові x позначки), в яких відсутні НЗ; libSNARK (сині трикутники); BCCGP (блакитний + позначки), виконані в однопотоковому режимі; Ligero (червоні квадратики) і НЗ-МПД (зелені кола). Зліва направо вимірюється час роботи P, час V та складність зв'язку. Для libSNARK порожні позначки на середній і правій діаграмах вимірюють лише час перевірки після обробки та складності зв'язку (СЗ) відповідно; повні оцінки вимірюють загальний час перевірки та СЗ, що включає (непрозору) фазу генерації ключа. Специфікація сервера: 32 ядра AMD з тактовою частотою 3,2 ГГц, з 512 ГБ оперативної пам'яті DDR3. (Кожна пара ядер має спільну пам'ять; це приблизно відповідає машині з 16 ядрами й гіперпотоком.)

2 ОСНОВНІ ОЗНАЧЕННЯ ТА ВЛАСТИВОСТІ РІЗНИХ ТИПІВ СИСТЕМ ДОВЕДЕНЬ БЕЗ РОЗГОЛОШЕННЯ

2.1 Від АПП до НЗ-МПД

Метою цього розділу є детальне визначення ІДО-конструкцій. Існування системи НЗ-МПІДО для мови NEXP вже було встановлено в [73, 20]; таким чином, увага буде зосереджена на конкретній ефективності та на детальному описі конструкції, реалізованої в коді.

Для цілей цієї роботи одиницею твердження про цілісність обчислень, позначений $\mathbb{x}$, визначається наступним:

1) відношенням переходу в просторі станів машини;

2) набором граничних обмежень (наприклад, входів і виходів).

Свідком $\mathbb{w}$ цілісності одиниці цілістності $\mathbb{x}$ є *правильний результат обчислення*, задане виконуваним слідом – послідовністю станів машини, яка дотримується як граничних, так і перехідних обмежень обчислення. Використання операторів ЦО для твердження, наприклад, *«значення α є результатом функції C за T кроків на публічному вході x»* у цьому форматі є простим застосуванням теореми Кука-Левіна.

Процес складається із наступних 4 частин:

1) Вихідною точкою є класичне алгебраїчне проміжне представлення (АПП) елементів $\mathbb{x}$ та $\mathbb{w}$, вони записуються як $\mathbb{x}_{АПП}$, $\mathbb{w}_{АПП}$ відповідно. Під «алгебраїчними» мається на увазі, що стани виконуваного сліду представлені послідовностями елементів у скінченному полі $\mathbb{F}$, а відношення переходу задається набором поліномів над змінними, що представляють «поточний» і «наступний» кроки виконуваного сліду.

2) Ми зводимо представлення АПП до іншого, в якому стани виконуваного сліду «розміщуються» на вузлах афінного графа так, що послідовні стани з'єднуються ребром у цьому графі. Неформально афінний граф – це «схема», яка має «алгебраїчну» топологію. Процес

«розміщення» машинних станів по вузлах схеми нагадує процес розміщення та маршрутизації, який зазвичай використовується в комп'ютерному та схемному проєктуванні, хоча поточний простір проєктування обмежений алгеброю, а не фізичною реальністю. Це перетворення буде називати алгебраїчним розміщенням та маршрутизацією (АРМ), а отримане уявлення є парою екземпляр/свідок АРМ ($\mathbb{x}_{APM}$, $\mathbb{w}_{APM}$). Це скорочення є детермінованим з боку V, тобто не передбачає випадковості зі сторони V та взаємодії. Також, він ще має ідеальну повноту і ідеальну вагомість (P, використовує випадковість, щоб отримати властивість НЗ; однак це використання не впливає на повноту чи вагомість.)

3) Представлення APM використовується для створення через за 1-раундового ІДО пари екземплярів задачі Ріда-Соломона по тестуванню наближеності (THP). Тепер екземпляри визначаються параметрами коду РС. Два коди, отримані в результаті скорочення, знаходяться в одному полі $\mathbb{F}$, але мають різні домени оцінки (L,L_{cmp}) і різні кодові швидкості (ρ, ρ_{cmp}). Свідком у цьому випадку є пара передбачуваних кодових слів ($f^{(0)}$), $g^{(0)}$). Випадковість V в 1-раундовому ІДО використовується (серед іншого) для «зв'язку» багатьох обмежень відношення переходу в одне випадкове. Таким чином, цей крок називається протоколом алгебраїчного зв'язування ІДО (АЗІ).

4) Для кожної з двох функцій (оракулів) $f^{(0)}$, $g^{(0)}$ викликається швидкий РС ІДО (FRI) протокол для наближеності [74], і це завершує скорочення.

2.2 Зауваження та позначення.

Щільність. Щільністю підмножини S' скінченної множини S називається $\mu(S'/S) \triangleq |S'|/|S|$.

Набори та функції. Позначимо через 0 постійну нульову функцію. Для множини S та функції f з областю визначення в S

позначимо $f(S) = \{f(x)|x \in S\}$, а для множини функцій $\mathcal{F}$ з областю визначення в S нехай позначається як $\mathbb{F}(S) = \bigcup_{f\in\mathcal{F}} f(S)$.

Двійкові поля. Скінченне поле позначається $\mathbb{F}$, а $\mathbb{F}_q$ – поле розміру q. Двійкове поле – скінченне поле характеристики 2. Всі поля, які розглядаються в цій роботі, є двійковими. Коли використовується термін афінного простору для підмножини $H \in \mathbb{F}$, яка є адитивним класом класу лінійного простору $\mathbb{F}_2$, то це означає $H = \alpha + V \triangleq \{\alpha + \upsilon|\upsilon \in V\}$ для деяких фіксованих $\alpha \in \mathbb{F}$ і $V \subseteq \mathbb{F}$ лінійного простору над двоелементним полем $\mathbb{F}_2$.

Зауваження. (Канонічне представлення двійкових полів). Припускається канонічне представлення для бінарних полів $\mathbb{F}$, заданого незвідним поліномом і примітивним елементом $g \in \mathbb{F}$ для нього (тобто g породжує $\mathbb{F}^*$). У цій роботі використовується стандартна основа $\{1, g, g^2, \ldots, g^{n-1}\}$, що представляє $\mathbb{F}_{2^n}$ над $\mathbb{F}_2$. Аналогічно, для фіксованого $\mathbb{F}_2$ підпростору розмірності k, (ненульовий) поліном степеня 2^k, який обертається в нуль на S, вважатиметься відомим V. Зазначимо, що цей поліном є лінеаризованим і має не більше k ненульових коефіцієнтів. Зокрема, при обговоренні арифметичної складності P та V ми припускаємо, що представлення всіх релевантних полів, примітивних елементів і поліномів підпростору відомі обом сторонам. Як стане очевидним, вони залежать лише від параметрів обчислення (наприклад, степінь і час виконання). Це припущення можна зробити без значної втрати загальності, оскільки V може попросити довідника представити всі такі уявлення разом із відповідним «доведенням примітивності» для g.

Коди. Розглядаються кодові слова в лінійному коді виправлення помилок C над $\mathbb{F}$ довжиною блоку n як функції з областю визначення S, $|S| = n$ і в діапазоні $\mathbb{F}$, тобто $C \subseteq \mathbb{F}^S$; для $\mathtt{x} \subseteq S$, входження $\mathtt{x}$ в кодове слово $\mathtt{w} \in C$ позначається як $\mathtt{w}(x)$. Для $\upsilon \in \mathbb{F}^S$ дозволяє $\mathtt{w}_\upsilon \in C$ бути кодовим словом, найближчим до υ за відносною відстанню Хеммінга, розривом зв'язує довільно (скажімо, довільно впорядковуючи елементи C і встановлюючи $\mathtt{w}_\upsilon$ найменшим $\mathtt{w} \in C$ з мінімальною відстанню до υ).

Умови. Для $S' \subset S$ позначаємо через $\mathbb{w}|_{S'}$, умова $\mathbb{w}$ на підмножині множини визначення S і аналогічно визначаємо $C|_{S'} = \{w|_{S'} | w \in C\}$, помічаючи $C|_{S'} \subseteq \mathbb{F}^{S'}$.

Інтерполянти, оцінки та розширення «малого» степеня. Інтерполянт з $\mathbb{w} \in \mathbb{F}^S$, позначається як $interpolant^w$, називається єдиний поліном P, $\deg(P) < |S|$, який задовольняє умову $\forall x \in S, P(x) = \mathbb{w}(x)$. Багатоточкове оцінювання $P(x)$ в області S є функцією $f : S \to \mathbb{F}$, яка задовольняє $f(x) = P(x)$ для всіх $\mathbb{x} \in S$. Коли $|S| > \deg(P)$, то інтерполянтом цієї оцінки є P.

Теорема 2.1. *(Арифметична складність розширення «малого» степеня (ОНС) [76]).*

Для $\mathbb{F}$ двійкового поля, $S \subseteq \mathbb{F}$, афінного простору $\mathbb{F}_2$ і $c_1, \ldots, c_n \in \mathbb{F}$ існує арифметична схема, яка обчислює необхідний рядок порад A, який вимагає $(n+1)polylog|S|$ арифметичних операцій в $\mathbb{F}$;

а також існує інша арифметична схема, яка використовує пораду a та обчислює ОНС функції $f: S \to \mathbb{F}$ до області визначення $\bigcup_{C_i} S + c_i$. Ця остання схема має арифметичну складність $3(n+1)(|S| \log |S|)$ над $\mathbb{F}$.

Для простоти ігнорується складність обчислення поради, яка в порівнянні з $3(n+1)(|S| \log |S|)$ мізерно мала.

Арифметична складність. Для арифметичної схеми C з вентильними елементами ≤ 2 над набором входів $\{+, \times, \div\}$, позначаємо через $T_{arith}(C)$ арифметичну складність C, яка визначається як загальна кількість входів в арифметичній схемі. Складність множення – це число входів $\times$. Складність додавання визначається аналогічним чином. Для функції $f : \mathbb{F}^n \to \mathbb{F}^m$, чия «канонічна схема» якої C_f неявно відома, позначимо як $T_{arith}(f) = T_{arith}(C_f)$.

Дано набір функцій $\mathcal{F} = \{f_1, f_2, \ldots, f_m\}$ відображення елементів $\mathbb{F}^n$ до $\mathbb{F}$ позначимо через $T_{arith}(\mathcal{F})$ (загальну) арифметичну складність $\mathcal{F}$ і визначимо її як арифметику схеми $C : \mathbb{F}^n \to \mathbb{F}^m$ таку, що для будь-якого $\vec{v} \in \mathbb{F}^n$ та для будь-якого $1 \leq i \leq m$, $C(\vec{v}_i) = f_i(\vec{v})$.

Функції, що оцінюються на наборах входів. Для функції $f : S \to \sum$ та $S' \subset S$ позначимо $f(S') = \{f(x)|x \in S'\}$. Коли $f(S')$ є одиночною множиною $\{\alpha\}$, то позначення буде простіше $\alpha = f(S')$.

2.3 Алгебраїчне проміжне представлення (АПП)

Алгебраїчний виконуваний слід програми, що виконується для T кроків, представлена масивом $\mathbb{w} \times T$, в якому кожен запис є елементом кінцевого поля. Один рядок описує стан обчислення в певний момент часу, а один стовпець відстежує «реєстр» та його вміст у часі. Легко скоротити пари (екземпляр, свідок) для відношення, яке визначає мову $L \in NTimeSpace(\hat{T}(n), \hat{w}(n))$ до екземплярів АПП з алгебраїчними виконуваними слідами розміру $T(n) = O(\hat{T}(n))$ і ширин b $w(n) = O(\hat{w}(n))$. Див., наприклад, [77] для прикладів таких скорочень.

Означення 2.1. (Алгебраїчне внутрішнє представлення (АВП)). Відношення $R_{АВП}$ є множиною пар $(\mathbb{x},\mathbb{w}) = (\mathbb{x}_{АПП}, \mathbb{w}_{АПП})$, яка задовольняє наступні умови:

1) **Формат екземпляра:** екземпляр $\mathbb{x}$ є кортежем $\mathbb{x} = (\mathbb{F},T,w,\mathcal{P},C,B)$, де:

– $\mathbb{F}$ є скінченним полем;

– T – ціле число, що представляє умови на час виконання;

– w – ціле число, що представляє ширину стану;

– $\mathcal{P} = \{P_1, \ldots, P_s\} \subset \mathbb{F}[X_1, \ldots, X_m, Y_1, \ldots, Y_m]$ – це набір обмежень. Степінь $\mathcal{P}$ дорівнює $\deg(\mathcal{P}) \triangleq max_{P \in \mathcal{P}} \deg(P)$

– C – монотонна булева схема над змінними $Z_1, \ldots, Z_s$ з кількома вхідними елементами логічних І або АБО. Розмір ланцюга – це кількість вентилів у ньому, а його степінь, позначений $\deg(C)$, визначається індуктивно таким чином:

– степінь вхідної змінної Z_i є $\deg(Z_i) = \deg(P_i)$, де $P_i \in \mathcal{P}$, що визначено вище;

– степінь логічного І елемента g_j з вхідними елементами $g_1, \ldots, g_t$ – це $\deg(g_j) = \max\{\deg(g_1), \ldots, \deg(g_t)\}$;

– степінь логічного АБО елемента g_j з вхідними елементами $g_1, \ldots, g_t$ – це $\deg(g_j) = \deg(g_1) + \cdots + \deg(g_t)$;

– $\deg(C)$ - це степінь його (єдиного) виходу.

– B – це набір граничних обмежень, де кожне граничні умови є кортежем (i,j,α) для $i \in [T]$, $j \in [w]$ та $\alpha \in \mathbb{F}$.

2) **Формат свідка:** Свідок $\mathbb{w}$ – це набір функцій $\mathbb{w}_1, \ldots, \mathbb{w}_w : [T] \to \mathbb{F}$. Свідок $\mathbb{w}$ задовольняє екземпляр тоді і тільки тоді, коли:

а) для всіх граничних обмежень (i, j,α) маємо $w_j(i) = \alpha$

б) для всіх $t \in [T-1]$ маємо $C(IsZero_s(\mathcal{P}(w[t],w[t+1]))) = TRUE$, де:

– $w[t] = (w_1(t), \ldots, w_w(t))$;

– $IsZero : \mathbb{F} \to \{TRUE,FALSE\}$ є відображенням, яке надсилає $0 \in \mathbb{F}$ до $TRUE$ і всі ненульові елементи до $FALSE$, і $IsZero_k : \mathbb{F} \to \{TRUE, FALSE\}^k$ є природним розширенням $IsZero$ для кількох входів і виходів, а саме $IsZero_k(\alpha_1, \ldots, \alpha_k) := (IsZero(\alpha_1), \ldots, IsZero(\alpha_k))$

Нарешті, $R_{АПП}$ – це множина всіх пар $(\mathbb{x},\mathbb{w})$ таких, що $\mathbb{w}$ задовольняють $\mathbb{x}$, і $L_{АПП} \triangleq \{\mathbb{x}|\exists\mathbb{w}, (\mathbb{x},\mathbb{w}) \in R_{АПП}\}$.

Зауваження. (Узагальнення на немонотонні схеми). Причина, по якій обмежується наведене вище визначення монотонними схемами, на відміну від загальних схем, які є більш показовими, полягає в тому, що додавання входів заперечення до булевої схеми може зруйнувати ідеальну повноту, знизити ефективність або вимагати більше раундів взаємодії (залежно від обраного шляху арифметизації). Часто достатньо використовувати закони Де Моргана, щоб ефективно підштовхнути елементи заперечення на входи C і включити їх у систему поліномів $\mathcal{P}$.

У скороченнях будуть використовуватися АПП, в яких $\mathbb{F}$ завжди є двійковим полем, і далі припускається, що розмір і степінь свідка є

«двійково дружніми» відповідно до наступного визначення.

Означення 2.2. (ДПАПП). Двійковий екземпляр АПП (ДПАПП) – це екземпляр АПП $\mathbb{x} = (\mathbb{F},T,w,\mathcal{P},C,\mathcal{B})$, який для деяких n, t, $d \in \mathbb{N}^+$ задовольняє наступні умови:

1) $|\mathbb{F}| = 2^n$;

2) $T = 2^t - 1$.

3) $\deg(\mathbf{C}) \leq 2^d$.

Такий екземпляр $\mathbb{x}$ називається (n,t,d)-ДПАПП екземпляром. Відношення $R_{ДПАПП}$ і мова $L_{ДПАПП}$ є природною умовою $R_{АПП}$ і $L_{АПП}$ для бінарних екземплярів АПП.

Наступна лема є «інформаційною теорією», тобто не спирається на відсутність недоведених криптографічних припущень і відповідає обчислювально необмеженими доведеннями. На відміну від попередніх протоколів НЗ-ІДО [17, 20], вона повністю визначає всі константи та не використовує жодних асимптотичних записів.

Лема 2.1. *(wi-ІДО знання для $R_{ДАПП}$). Для параметра помилки вагомості $\lambda : \mathbb{N}^+ \to \mathbb{N}^+$, ціле число $\mathcal{R} \geq 2$ і НЗ параметр $k \geq 1$, і деякої $\delta = (1 - 2^{\mathcal{R}}(1 + 2^{-d}))$. Нехай відношення $R_{АПП}^{(\lambda,\mathcal{R},k)}$ позначає умову $R_{АПП}$ щодо (n,t,d)-ДПАПП випадків, які задовольняють двом умовам:*

1) $n > \lambda + t + d + \mathcal{R} + k + 2$;,

2) $k + t \geq 6 + \log\lambda + \log\log\frac{1}{1-\frac{\delta}{10}}$.

Тоді $R_{АПП}^{\lambda,\mathcal{R}}$ має знання НЗ-ІДО з такими параметрами для екземплярів (n,t,d)-ДПАПП:

– похибка вагомості не більше $err = 2^{-\lambda}$;

– обмежена помилка в знаннях не більше $\epsilon' = 4 \cdot err$;

– арифметична складність P;

$$\mathbf{tp}^{\mathbb{F}} \leq (9w(t + d + \mathcal{R} + 3) + |\mathcal{P}| + w + T_{arith}(\mathcal{P}) + 12) \cdot 2^{t+d+\mathcal{R}+3};$$

– арифметична складність V

$$\mathbf{tv}^{\mathbb{F}} \leq 2\Bigg(|B|^2 + \lceil \frac{\lambda+2}{\log \frac{1}{1-\delta/10}} \rceil \cdot (8(w+|\Phi|+T_{arith}(\mathcal{P})+|B|)+21(k+\mathcal{R}+t+d+2) \Bigg);$$

– *складність округлення* $r = \frac{t+d}{2} + 2$*;*

– *складність запитів* $q \leq 8\lceil \frac{\lambda+2}{\log \frac{1}{1-\delta/10}} \rceil \cdot (2w + k + t + d + 2)$

Ми робимо припушення, що вагомість (двох окремих компонент) в зазначеному ІДО вище не є строгою. Відповідно, ми також припустимо що такая сама вагомість може бути отримана із меншною кількістю запитів, як зазначено у наступній лемі.

Лема 2.2. *(wi-ІДО для* $R^{\lambda}_{ДАПП}$ *із покращенним припущенням щодо вагомості). Нехай гіпотези 2.2 та 2.3 виконуються. Тоді для параметру помилки вагомості* $\lambda : \mathbb{N}^+ \to \mathbb{N}^+$*, цілого числа* $\mathcal{R} \geq 3$ *та НЗ параметру* $k \geq 1$*,* $R^{'(\lambda,\mathcal{R},k)}_{АПП}$ *позначає умови* $R_{ДАПП}$ *для* $(n,t,d) - ДАПП$ *екземпляру, який задовольяє умовам:*

1) $n > max58,\lambda,t + d + \mathcal{R} + k + 2$*;*

2) $k + t + \log \lceil \frac{\lambda+2}{\mathcal{R}} \rceil$*.*

Тоді $R^{'(\lambda,\mathcal{R},k)}_{АПП}$ *має НЗ-ІОД знання із наступними параметрами для нашого* $(n,t,d) - ДАПП$ *екземпляру:*

– *помилка вагомості і важкість округлення такі ж самі, як і у лемі 2.1;*

– *складність* V*:*

$$\mathbf{tv}^{\mathbb{F}} \leq 2|B|^2 + \lceil \frac{\lambda+2}{\mathcal{R}} \rceil \cdot (16(w + |\Phi| + T_{arith}(\mathcal{P}) + 42 \cdot (k + \mathcal{P} + t + d + 2));$$

– *складність запиту:*

$$q \leq 4 \cdot \Bigg(\lceil \frac{\lambda+2}{\mathcal{R}+d} \rceil \cdot (w + 3 + k + t + d) + \lceil \frac{\lambda+2}{\mathcal{R}} \rceil \cdot (3w + 3 + l + t + d) \Bigg);$$

– *складність* P*:*

$$\mathbf{tv}^{\mathbb{F}} \leq \Bigg(9w(t+d+\mathcal{R}+3) + |\mathcal{P}| + w + T_{arith}(\mathcal{P}) + \\ +12\lceil\frac{\lambda+2}{n-(10+2\log n)}\rceil\Bigg) \cdot 2^{t+d+\mathcal{R}+3}.$$

Означення 2.1 передбачає, що повний стан машини фіксується за допомогою $\mathbb{w}$ елементів поля. Деякі обчислення вимагають такої ж складності простору, як і часової складності ($w = \Omega(T)$), що призводить до АПП-свідків загального розміру $\Omega(w \cdot T) = \Omega(T^2)$. Щоб досягти свідків розміру $O(T \log T)$, незалежно від споживання пам'яті, потрібно наступне визначення. Воно дотримується підходу [78, 79] і використовує пару обмежень АПП – одне для верифікації правдивості послідовних кроків часу, а інше – для верифікації постійність пам'яті. Останній набір обмежень застосовується до перестановки кроків виконуваного сліду, і частина свідка є специфікацією цієї перестановки. Це відношення називається переставленим алгебраїчним проміжним представленням ($R_{ПАПП}$).

Означення 2.3. (Переставлене алгебраїчне проміжне представлення (ПАПП)). Відношення $R_{ПАПП}$ є множиною пар $(\mathbb{x},\mathbb{w}) = (\mathbb{x}_{ПАПП}, \mathbb{w}_{ПАПП})$, що задовольняють умовам:

1) **Формат екземпляра:** екземпляр $\mathbb{x}$ є кортежем $\mathbb{x} = (\mathbb{F},T,w,\mathcal{P}_T,\mathcal{P}_\pi,B)$ де

– $\mathbb{F}$ – скінченне поле;

– T – ціле число, що представляє умову на час виконання;

– w – ціле число, що представляє ширину стану;

– $\mathcal{P}_T, \mathcal{P}_\pi \subset \mathbb{F}[X_1,\ldots,X_w,Y_1,\ldots,Y_w]$ це два набори обмежень;

– B – це множина граничних обмежень, де кожне граничні умови є кортежем (i,j,α) для $i \in [T]$, $j \in [w]$, $\alpha \in \mathbb{F}$

2) **Формат свідка:** Свідок $\mathbb{w}$ – це пара $(\hat{w},\pi)$, де $\hat{w}$ – послідовність w функцій $w_1,\ldots,w_w : [T] \to \mathbb{F}$ і $\pi : [T] \to [T]$ є перестановкою. Будемо говорити, що $\mathbb{w}$ задовольняє екземпляр тоді та тільки тоді, коли:

а) для всіх граничних обмежень (i, j, α) маємо $w_j(i) = \alpha$;

б) для всіх $t \in [T-1]$ і для всіх $P \in \mathcal{P}_T$ маємо $P(w[t],w[t+1]) = 0$;

в) для всіх $t \in [T-1]$ і для всіх $P \in \mathcal{P}_\pi$ ми маємо $P(w[t],w[\pi(t)]) = 0$.

Відношення $R_{ПАПП}$ є множиною пар (𝕩,𝕨), де 𝕨 задовольняє 𝕩, і ПАПП $\triangleq \{\mathbb{x}|\exists \mathbb{w}, (\mathbb{x},\mathbb{w}) \in R_{АППwRAM}$

Визначення підвідношення двійкової ПАПП (ДПАПП) аналогічне визначенню бінарного АПП є у визначенні 2.2.

$L_{ПАПП}$ є NEXP-повним, як зазначено далі. Доведення наведено у [77].

Лема 2.3. *(wi-МПІДО для ДПАПП). Для кожної мови $L \in NTIME(T(n)), T(n) \geq n$ існує детерміноване скорочення за поліноміальний час від L до $L_{ПАПП}$, та відображаючий екземпляр 𝕩 з L до екземпляра $\mathbb{x} = (\mathbb{F},T,w,\mathcal{P}_T,\mathcal{P}_\pi,B)$ із $L_{ПАПП}$, що задовольняє умовам:*

– $T = O(T(n))$;

– $w = O(\log T(n))$;

– $\deg(\mathcal{P}_T \bigcup \mathcal{P}_\pi) = O(1)$, $|\mathcal{P}_T \bigcup \mathcal{P}_\pi| = O(\log T(n)$ *і* $T_{arith}(\mathcal{P}) = O(\log T(n)$;

– $|B| = O(n)$.

Крім того, недетермінований свідок 𝕨 для 𝕩 можна обчислити за час $O(T(n) \log T(n)) + n^{O(1)}$, якщо надано x і недетермінований свідок w при приналежності x до L.

2.3.1 Алгебраїчне розміщення та маршрутизація (АРМ).

Етап розробки схем, відомий як розміщення та маршрутизація, має справу з розташуванням станів обчислення для оптимізації певних фізичних обмежень. Наступна фаза робить щось подібне, розміщуючи та маршрутизуючи інформацію про стани обчислення таким чином, щоб оптимізувати алгебраїчні, а не фізичні умови. Тому називається бінарним відношенням алгебраїчного розміщення та маршрутизації (АРМ).

Нагадаємо, що $Aff_1(\mathbb{F})$ позначає 1-вимірну афінну групу над $\mathbb{F}$, її множина елементів ізоморфна до $\{aX + b | a \in \mathbb{F}^*, b \in \mathbb{F}\}$. Афінний граф, породжений множиною $\mathcal{N} = \{N_1, \ldots, N_s\} \subset Aff_1(\mathbb{F})$ – орієнтований граф із множиною вершин $\mathbb{F}$ та множиною ребер $\{(x, N(x)) | x \in \mathbb{F}, N \in \mathcal{N}\}$.

У наступному визначенні множина Φ охоплює як граничні, так і перехідні умови обчислення. Афінний граф додається до опису екземпляра через його утворювальну множину $\mathcal{N}$. Завдяки алгебраїчній топології кожен стовпець алгебраїчного виконуваного сліду (який відповідає реєстру і відстежується з часом) тепер є

кодовим словом Ріда Соломона зі швидкістю ρ і послідовністю показників (один показник на регістр/стовпець) також є частиною опису екземпляра. Поточна конструкція відрізняється від попередніх робіт [84, 79, 81, 22] за допомогою окремого кодового слова для кожного алгебраїчного регістра.

Означення 2.4. (Задача алгебраїчного розміщення та маршрутизації (APM)). Співвідношення R_{APM} містить усі пари (𝕩,𝕨), що задовольняють наступним умовам:

1) **Формат екземпляра:** екземпляр 𝕩 є кортежем $\mathbb{x} = (\mathbb{F}, \mathcal{T}, \mathcal{P}, \Phi, L, L_{cmp}, \vec{\rho}, \rho_{cmp})$ де:

– $\mathbb{F}$ – скінченне поле характеристики 2;

– $\mathcal{T}$ – це набір індексів, які називаються індексами алгебраїчного реєстру;

– $\mathcal{N} \subseteq \mathcal{T} \times Aff_1(\mathbb{F})$ – це набір пар, які називаються сусідами, кожна пара містить індекс алгебраїчного реєстру та член афінної групи над $\mathbb{F}$ (поліном степеня 1). Даний набір $S \subset \mathbb{F}$ позначаємо через $\mathcal{N}(S)$ набір $\mathcal{N}(S) := \{N(x) | x \in S, (\tau, N) \in \mathcal{N}$ для деяких $\tau \in \mathcal{T}$;

– $\Phi \subseteq (\mathbb{F} \times \mathbb{F}^{\mathcal{N}}) \to \mathbb{F}$ це набір відображень над змінною $\mathcal{V} := \{X_{|OC}\} \bigcup \{X_N\}_{N \in \mathcal{N}}$. Призначення для будь-якого $\phi \in \Phi$ виражається шляхом відображення $\alpha_{\vec{f}, \mathcal{N}} : \mathcal{V} \to \mathbb{F}$. Дано набір $S \subset \mathbb{F}$ і послідовність функцій, індексованих елементами $\mathcal{T}, \vec{f} \in (\mathbb{F}^{\mathcal{N}(S)})^{\mathcal{T}}$, позначимо

$\alpha_{\vec{f},\mathcal{N}} : S \to \mathbb{F}^{\mathcal{V}}$ визначене відображення $\left[\alpha_{\vec{f},\mathcal{N}}(x)\right]_{X_{|OC}} := x$, і для кожного $(\tau, N) \in \mathcal{N}\left[\alpha_{\vec{f},\mathcal{N}}(x)\right]_{(\tau,N)} := f_\tau(N(x))$. Для $\phi \in \Phi$ позначаємо через $\phi_{\mathcal{N}}\left[\vec{f}\right] \in \mathbb{F}^S$ вектор, визначенний як $\left[\phi_{\mathcal{N}}\left[\vec{f}\right]\right]_x := \phi\left(\alpha_{\vec{f},\mathcal{N}}(x)\right)$ для кожного $x \in S$;

– L та L_{cmp} – це два $\mathbb{F}_2$-афінні підпростори $\mathbb{F}$, які називаються простором оцінки-свідка та простором оцінки композиції відповідно;

– $\vec{p} \in (0,1)^{\mathcal{T}}$ являє собою послідовність показників, що називаються показниками свідків, а $\rho_{cmp} \in (0,1)$ називають показником композиції.

2) **Формат свідка та задовільність умови** Свідок $\hat{\mathbb{w}}$ – це послідовність функцій, індексованих елементами з $\mathcal{T}$; формально $\hat{\mathbb{w}} \in (L^{\mathbb{F}})^{\mathcal{T}}$. Кажуть, що $\hat{\mathbb{w}}$ задовольняє $\mathbb{x}$, якщо виконуються обидві наступні умови:

– **наявність в коді призначення:** $\forall\tau \in \mathcal{T} : \mathbb{w}_\tau \in RS[\mathbb{F}, L, \rho_\tau]$;

– **наявність в коді умов:** $\forall\phi \in \Phi, \phi_{\mathcal{N}}[\hat{\mathbb{w}}] \in RS[\mathbb{F}, L_{cmp}, \rho_{cmp}]$

Нехай R_{APM} – це бінарне відношення, яке містить усі пари $(\mathbb{x}, \hat{\mathbb{w}})$ такі, що $\mathbb{x}$ є екземпляром, як визначено вище, а $\hat{\mathbb{w}}$ задовольняє $\mathbb{x}$, і нехай APM – недетермінована мова, індукована R_{APM},

$$APM = \{\mathbb{x} | \exists\hat{\mathbb{w}}, (\mathbb{x}, \hat{\mathbb{w}}) \in R_{APM}.$$

Три властивості, визначені далі, використовуються пізніше, щоб довести НЗ та забезпечити кращу вагомість границь.

Означення 2.5. (властивості APM).

Кажуть, що екземпляр $\mathbb{x} := (\mathbb{F}, \mathcal{T}, \mathcal{N}, \Phi, L, L_{cmp}, \vec{\rho}, \rho_{cmp}) \in APM$ коли:

1) він є k-незалежним, якщо існує послідовність множин $\{V_\tau\}_{\tau\in\mathcal{T}}$ така, що кожна множина $V_\tau \subset \mathbb{F}^L$ є $k|L|$ незалежною, і для будь-якого $\hat{\mathbb{w}}$ такого, що $\mathbb{w}_\tau \in V_\tau$, справедливо, що $\hat{\mathbb{w}}$ задовольняє $\mathbb{x}$. Будемо $T_{Sampling}(\cdot)$ позначати арифметичну складність схеми, що на вході $\mathbb{w}_\tau \in V_\tau$ (і випадкових елементах поля), вибірки рівномірно випадкові $\mathbb{w} \in \{V_\tau\}_{\tau\in\mathcal{T}}$.

2) має δ-відстань, якщо:

– $\delta \leq 1 - \rho_{cmp}$;

– $\delta \leq 1 - \rho_\tau$ для будь-якого $\tau \in \mathcal{T}$;

– існує лінійний код $C \subset \mathbb{F}^{L_{cmp}}$ відносної відстані не менше за δ, що містить код $RS[\mathbb{F}, L_{cmp}, \rho_{cmp}]$, такий, що для будь-якої послідовності відображень $\vec{f} \in (\mathbb{F}^L)^{\mathcal{T}}$ де $f_\tau \in RS[\mathbb{F},L, \rho_\tau]$ для всіх $\tau \in \mathcal{T}$, виконується $\forall \phi \in \Phi, \phi_{\mathcal{N}}[\hat{w}] \in C$.

3) він є Θ-перекриванням, якщо для будь-якого $y \in L$ розмір множини $S_y := \{x \in L_{cmp} | y \in \mathcal{N}(\{x\})\}$ є максимумом Θ.

2.4 Зниження АРМ

Наступна пара теорем описує перший крок скорочення, на якому ДПАПП, або ДАПП, пара екземплярів-свідок, зводиться до пари екземпляр-свідок АРМ. Скорочення є детермінованими і не пов'язані з ними. Оскільки скорочення на стороні екземпляра виконується Vом, а редукція на стороні свідка стороною P, позначаються два скорочення теореми 2.2 через $V^{ДАПП \to АРМ}, P^{ПАПП \to АРМ}$ відповідно, а з теореми 2.3 $V^{ДПАПП \to АРМ}, P^{ДПАПП \to АРМ}$.

Ідеї, використані в обох скороченнях, засновані на ідеях з [82, 83, 84, 78, 81], де афінні графи використовуються для кодування та перевірки програмних слідів. Поточна конструкція найбільш подібна до тієї, що використовується в [22], де циклічний графік використовується для вбудовування відомого для перевірки порядку сліду, який зазвичай використовується для перевірки узгодженості двох послідовних конфігурацій, а взаємний графік De Bruijn – графік для перевірки перестановки, невідомої V, зазвичай використовується для перевірки узгодженості пам'яті. Новим в побудові в [22] у порівнянні з попередньою побудовою є наступні моменти:

1) використання циклічного графіка для послідовних перевірок узгодженості конфігурацій;

2) використання метелика де Брейна «спина до спини» для представлення перестановок замість «прямого» Граф Де Брюйна, який в $\times$2,5 більший;

3) вибіркова маршрутизація на графі Де Брюйна лише невеликої частини конфігурації.

Все це зменшує розмір свідка, ефективно зменшуючи ступені поліномів, які обробляє система доведення, і конкретну ефективність сторони, що доводить. Нова техніка в цій роботі використовує методи, описані в [22], і додатково розбиває свідка на кілька поліномів, кожен зі ступенем набагато меншим, ніж той, який був необхідний для представлення всього свідка, фактично ще більше знижуючи степінь поліномів, які обробляються системою доведень. Протокол системи підтвердження використовує інтерактивність, щоб усунути необхідність надання окремого доведення наближеності РС до кожного полінома-свідка. Скорочення використовують конструкцію НЗ-ІДО, запровадженого [73], готуючи екземпляр АРМ і свідка для протоколу з НЗ.

Теорема 2.2. *(Алгебраїчне розміщення та маршрутизація). Є два алгоритми*

$V^{ДАПП\to APM}, P^{ДАПП\to APM}$ *такі, що для будь-якого* $k, \mathcal{R}, t, d \in \mathbb{N}^+$ *і екземпляр ДАПП* $\mathbb{x} = (\mathbb{F}, T, w, \mathcal{P}, C, B)$ *з* $T = 2^t - 1 > 4, \deg \mathcal{P} \leq 2^d$ *і* $|\mathbb{F}| = 2^n$ *з* $n > 2 + k + \mathcal{R} + t + d$ *буде виконуватись наступне:*

1) Ідеальна повнота:
$(\mathbb{x}, \mathbb{w}) \in R_{АПП} \implies (V^{\text{ДАПП}\to APM}(x, k, \mathcal{R}), P^{ДАПП\to APM}(\mathbb{x}, \mathbb{w}, k, \mathcal{R}))$
$\in R_{APM}$

2) Ідеальна вагомість: $\mathbb{x} \in АПП \impliedby V^{\text{ДАПП}\to APM}(\mathbb{x}, k, \mathcal{R}) \in APM$

3) Вилучення знань: існує ефективний екстрактор знань $E^{\text{ДАПП}\to\text{АРМ}}$ *такий, що для кожного* $(V^{\text{ДАПП}\to\text{АРМ}}(\mathbb{x}, k, \mathcal{R}, \mathbb{w}_{APM}) \in R_{APM}$ *екстрактор виводить свідка* $\mathbb{w} \leftarrow E^{\text{ДАПП}\to\text{АРМ}}(\mathbb{x}, k, \mathcal{R}, \mathbb{w}_{APM})$ *такого, що* $(\mathbb{x}, \mathbb{w}) \in R_{APM}$

4) Нехай

$\mathbb{x}_{APM} = (\mathbb{F}, \mathcal{T}, \mathcal{P}, \Phi, L, L_{cmp}, \vec{\rho}, \rho_{cmp}) = V^{\text{ДПАПП}\to\text{АРМ}}(x,k,\mathcal{R},)$, *тоді:*

а) $|\mathcal{T}| = w$;

б) $|\mathcal{N}| = 3w$ *з такими самими 3-ма сусідніми поліномами для кожного індексу-свідка* $\tau \in \mathcal{T}$;

в) $|\Phi| =\leq 2|\mathcal{P}|$ *і* $T_{arith}(\Phi) \leq 2T_{arith}(\mathcal{P}) + 3t + O(w \cdot |\mathcal{P}| \cdot |B|)$;

г) $\log_2 |L| = 2 + k + \mathcal{R} + t + d$

д) $\log_2 |L_{cmp}| = k + \mathcal{R} + t + d$

е) максимальний розмір ρ_{max} *в* $\vec{\rho}$ *задовольняє* $\rho_{max} \leq 2^{-(2+\mathcal{R}+d)}$

ж) $\rho_{cmp} \leq 2^{-\mathcal{R}}$

з) $\mathbb{x}_{APM}$ *є* $(1 - 2^{-\mathcal{R}}(1 + 2^{-d}))$*-відстанню, 1-перекритим, і* $(2^{-(2+\mathcal{R}+d)}(1 - 2^{-k}))$*- незалежний з* $T_{Sampling}(\mathbb{x}_{APM}) \leq \tilde{O}(|L|)$

5) Арифметична складність над $\mathbb{F}$ *(див. зауваження 2.2:*

а) V*: Арифметична складність* $V^{\text{ДАПП}\to APM}(\mathbb{x},k,\mathcal{R})$ *це те, що потрібно для обчислення поліномів* $Z_{B,j}, \xi_{B,j} \in \mathbb{F}(x)$ *для всіх* $1 \leq j \leq w$*, і зокрема, це на* $2|B|^2$ *більше арифметичних дій*

[27]

б) P*: Арифметична складність* $P^{\text{ДАПП}\to APM}(\mathbb{x},\mathbb{w},k,\mathcal{R})$ *це найбільша накопичена арифметична складність наступного:*

i. арифметична складність $V^{\text{ДАПП}\to APM}(\mathbb{x},k,\mathcal{R})$;

ii. 2^t *множення та додавання за* $\mathbb{F}$;

iii. $3w$ *обчислень «малого» степеня (ОНС), кожне з яких над афінним простором розмірності не більше* $k + t$*, з оцінкою не більше ніж* $2^{\mathcal{R}+d+2}$ *переміщень; зокрема, кожна ОНС може бути обчислена з арифметичною складністю не більше за* $3 \cdot (k + t) \cdot 2^{k+t+\mathcal{R}+d+2}$ *(див. теорему 2.1):*

Наступна теорема є аналогом попередньої, викладеної більше для $R_{ДПАПП}$, ніж для $R_{ДАПП}$

Теорема 2.3. *(Алгебраїчне розміщення та маршрутизація – переставлена версія). Є два алгоритми, позначені як* $V^{\text{ДПАПП}\to APM}$,

[1] Для великих наборів граничних обмежень можна змінити конструкцію так, щоб $Z_{B,j}$ був лінеаризованим поліномом, і в цьому випадку арифметична складність зменшується до $O(|B| \log |B|)$; деталі пропущені, оскільки зазвичай $|B|$ є невеликим

$P^{\text{ДПАПП}\to APM}$ *таких, що для будь-якого* $k, \mathcal{R} \in N$, *де* $\mathcal{R} > 1$ *і ДПАПП екземпляр* $\mathbb{x} = (\mathbb{F}, T, w, \mathcal{P}_T, \mathcal{P}_\pi, B)$ *з* $T = 2^t - 1 > 4$, $max\{\deg \mathcal{P}_\mathcal{T}, \deg \mathcal{P}_\pi, 2\} \leq 2^d$, *і* $[\mathbb{F} : \mathbb{F}_2] > 2 + k + \mathcal{R} + t + \lceil \log(t+1) \rceil + d$ *виконується:*

1) Ідеальна повнота: $(\mathbb{x}, \mathbb{w}) \in R_{ДАПП} \implies (V^{\text{ДПАПП}\to APM}(\mathbb{x}, k, \mathcal{R}), P^{ДАПП\to APM}(\mathbb{x}, \mathbb{w}, k, \mathcal{R})) \in R_{APM}$;

2) Ідеальна вагомість: $\mathbb{x} \in ДПАПП \impliedby V^{\text{ДПАПП}\to APM}(\mathbb{x}, k, \mathcal{R}) \in APM$;

3) Вилучення знань: існує ефективний інструмент для вилучення знань $E^{\text{ДПАПП}\to APM}$ *такий, що для кожного* $(V^{\text{ДПАПП}\to APM}(\mathbb{x}, k, \mathcal{R}), \mathbb{w}_{APM}) \in R_{APM}$, *екстрактор виводить свідка* $\mathbb{w} \leftarrow E^{\text{ДПАПП}\to APM}(\mathbb{x}, k, \mathcal{R}, \mathbb{w}_{APM})$ *такий, що* $(\mathbb{x}, \mathbb{w}) \in R_{\text{APM}}$

4) Нехай $\mathbb{x}_{APM} = (\mathbb{F}, \mathcal{T}, \mathcal{N}, \Phi, L, L_{cmp}, \vec{\rho}, \rho_{cmp}) = V^{\text{ДПАПП}\to\text{APM}}(\mathbb{x}, k, \mathcal{R})$, *тоді:*

а) $|\mathcal{T}| = 2(w+1)$;

б) $\mathcal{N} \leq 11|\mathcal{T}|$ *з такими самими 11-ма сусідніми поліномами для кожного індексу-свідка* $\tau \in \mathcal{T}$;

в) $|\Phi| =\leq 2|\mathcal{P}_\mathcal{T}| + |\mathcal{P}_\pi| + 8w + 11$ *і* $T_{arith}(\Phi) \leq 2T_{arith}(\mathcal{P}_\mathcal{T}) + T_{arith}(\mathcal{P}_\pi) + O(w(|\mathcal{P}_\mathcal{T}| + |\mathcal{P}_\pi|) + t)$;

г) $\log_2 |L| = 2 + k + \mathcal{R} + t + \lceil \log(t+1) \rceil + d$;

д) $\log_2 |L_{cmp}| = k + \mathcal{R} + t + \lceil \log(t+1) \rceil + d$;

е) максимальне значення показника $\rho_{\max}$ *в* $\vec{\rho}$ *задовольняє* $\rho_{\max} \leq 2^{-(2+\mathcal{R}+d)}$;

ж) $\rho_{cmp} \leq 2^{-\mathcal{R}}$;

з) $\mathbb{x}_{APM}$ *є* $(1 - 2^{-\mathcal{R}}(1 + 2^{-d}))$*-відстань, 5-раундний і* $(2^{-(2+\mathcal{R}+d)}(1 - 2^{-k}))$*-незалежний із* $T_{Sampling}(\mathbb{x}_{APM}) \leq \tilde{O}(|L|)$

5) Арифметична складність над $\mathbb{F}$ *(див. зауваження 2.2):*

а) V*: Арифметична складність* $V^{\text{ДПАПП}\to APM}(\mathbb{x}, k, \mathcal{R})$ *це те, що потрібно для обчислення поліномів* $Z_{B,j}, \xi_{B,j} \in \mathbb{F}[x])$ *для всіх* $1 \leq j \leq w$, *і зокрема, це на* $2|B|^2$ *більше арифметичних дій;*

б) P: Арифметична складність $P^{\text{ДПАПП}\to APM}(\mathbb{x},\mathbb{w},k,\mathcal{R})$ *це найбільша накопичена арифметична складність наступного:*

i. арифметична складність $V^{\text{ДПАПП}\to APM}(\mathbb{x},k,\mathcal{R})$*;*

ii. складність маршрутизації взаємної мережі метеликів Де Брюйна розмірності t*;*

iii. $2^t + 2^{\lceil\log(t+1)\rceil}$ *множення та додавання в* $\mathbb{F}$*;*

iv. $5(w+1)$ *обчислень розширення «малого» степеня (ОНС), кожне з яких над афінним простором розмірності не більше* $k + t + \lceil\log(t+1)\rceil$*, з оцінкою не більше ніж* $2^{\mathcal{R}+d+2}$ *переміщень; зокрема, кожна ОНС може бути обчислена з арифметичною складністю не більше* $3 \cdot (k + t + \lceil\log(t+1)\rceil) \cdot 2^{k+t+\lceil\log(t+1)\rceil+\mathcal{R}+d+2}$ *(див. теорему 2.1)*

2.5 Алгебраїчне зв'язування ІДО (АЗІ)

Результатом скорочень, описаних у попередньому розділі, є пари у відношенні R_{APM}. Наступний етап у скороченні використовується 1-раунд ІДО, який називається *алгебраїчним зв'язуванням ІДО* (АЗІ), що дозволяє звести проблему до пари *задачі двійкового тестування наближеності РС* (ДТНР). Наступне визначення формально визначає відношення, що лежить в основі проблеми.

Означення 2.6. (Бінарне тестування наближеності РС (ТНР)). {Приклади проблеми тестування наближеності РС (ТНР) є трійками $\mathbb{x}_{RS} = (\mathbb{F},S,\rho)$, де $S \subseteq \mathbb{F}$ і $\rho \in [0,1]$. Свідок $\mathbb{w}_{RS}$ для $\mathbb{x}_{RS}$ є функцією $\mathbb{w}_{RS} : S \to \mathbb{F}$, і воно задовольняє $\mathbb{x}_{RS}$ тоді і тільки, якщо $\mathbb{w}_{RS} \in RS[\mathbb{F},S,\rho]$. Відношення R_{THP} -це множина пар $(\mathbb{x}_{RS}, \mathbb{w}_{RS})$ така, що $\mathbb{w}_{RS}$ задовольняє $\mathbb{x}_{RS}$.

Двійкове відношення ТНР ($R_{ДТНР}$) – це відношення, якому екземпляр задовольняє:

– $\mathbb{F}$ – це двійкове поле;

– S є афінним класом $\mathbb{F}_2$-лінійного підпростору $\mathbb{F}$, який визначається зсувом класу зміщення a_0 і базисом $(a_1, \ldots, a_k)$ таким, що $S = \left\{ a_0 + \sum_{i=1}^{k} \alpha_i a_i | \alpha_1, \ldots, \alpha_k \in \mathbb{F}_2 \right\}$;

– $\rho = 2^{-\mathcal{R}}$ для всіх $\mathcal{R} \in \mathbb{N}^+$.

Наступна теорема описує основні властивості протоколу АЗІ. Тут бере участь пара екземпляр-свідок R_{APM} і використовує один раунд взаємодії, щоб звести її до двох пар екземпляр-свідок відношення $R_{ДТНР}$.

Теорема 2.4. *(Властивості алгебраїчного зв'язування ІДО (АЗІ)). Нехай* $\mathbb{x} := (\mathbb{F}, \mathcal{T}, \mathcal{N}, \Phi, L, L_{cmp}, \vec{\rho}, \rho_{cmp})$ *бути екземпляром APM, що перекривається* Θ *і* δ*-відстанню, з* $\delta > 0$*. Протокол АЗІ, застосований до* $\mathbb{x}$*, задовольняє наступні властивості:*

1) ***Протокол розкладу***

а) P***:*** *На вході є* $(\mathbb{x}, \mathbb{w}')$*, на виході* $P^{АЗІ}(\mathbb{x}, \mathbb{w}')$*, одиничний оракул* $\mathbb{O}_{assignment}$ *складається з* $|\mathcal{T}| + 1$ *функції* $\mathbb{F}^L$ *та одиничної функції* $\mathbb{F}^{L_{cmp}}$*;*

б) V***:*** *Вхід* $\mathbb{x}$ *і вихід* $V^{АЗІ}(x)$ *є одним повідомленням* R*, що складається з* $2|\mathcal{T}| + |\Phi|$ *випадкових елементів поля пари* $\mathbb{x}_{RS} = (\mathbb{F}, L, \rho_{\max}), \mathbb{x}'_{RS} = (\mathbb{F}, L_{cmp}, \rho_{cmp})$ *стосовно ДТНР;*

в) ***Наведений вихід:*** *Оракул* $\mathbb{O}_{assignment}$ *і випадковість* V *визначають пару функцій* $f^{(0)} : L \to \mathbb{F}$ *і* $g^{(0)} : L_{cmp} \to \mathbb{F}$ *таких, що кожен запис* $f^{(0)}$ *залежить від* $|\mathcal{T}| + 1$ *записів в* $\mathbb{O}_{assignment}$ *і кожен запис* $g^{(0)}$ *залежить від* $|\mathcal{N}| + 1$ *записів в* $\mathbb{O}_{assignment}$*.*

2) ***Повнота*** *Якщо* $\mathbb{x}$ *задовольняється* $\mathbb{w}'$*, то для будь-якої випадковості* R *маємо* $f^{(0)} \in RS[\mathbb{F}, L, \rho_{max}]$ *і* $g^{(0)} \in RS[\mathbb{F}, L_{cmp}, \rho_{cmp}]$*.*

3) ***Вагомість*** *Нехай*

$$\zeta \triangleq \max \left\{ 4{,}1 + \Theta \frac{|L|}{|L_{cmp}|} \right\} \tag{2.1}$$

якщо $\mathbb{x} \notin APM$ *тоді для кожного* $\mathbb{O}_{assignment}$ *виконується принаймні одна з наступних умов,*

а) $Pr_R\left[\Delta_H(f^{(0)}, RS[\mathbb{F}, L, \rho_{max}]) < \frac{\delta}{2\zeta}\right] \leq \frac{1}{|\mathbb{F}|}$

б) $Pr_R\big[\Delta_H(g^{(0)}, RS[\mathbb{F},L_{cmp},\rho_{cmp}]) < \frac{\delta}{2\zeta}\big] \le \frac{1}{|\mathbb{F}|}$

4) ***Добування знань****: Припустимо* $\rho_{max} \le \frac{1}{4}$*. Тоді існує Лас-Вегаський (рандомізований) поліноміальний алгоритм часу* E*, для якого виконується наступне.*

Якщо $\mathbb{w}^*$ *не задовольняє обом умовам, зазначеним у пунктах 3а і 3б вище, то вихід* E *на вхід* $(\mathbb{x}, \mathbb{w}^*)$ *є свідком* $\mathbb{w}'$*, який задовольняє* $\mathbb{x}$

5) ***Ідеальне НЗ****: Якщо екземпляр* $\mathbb{x}$ *є* k*-незалежним, то протокол АЗІ має ідеальне НЗ щодо будь-якого* V*, який робить не більше* $k|L|$ *запитів до кожного* $\mathbb{w}_\tau$ *. Це стосується будь-якого вибору підпротоколів РС-ІДОБ.*

6) ***Арифметична складність****:*

а) ***Доведення:*** *Припустимо, що* P*, володіє* $\mathbb{w}'$*, яке задовольняє* $\mathbb{x}$*, тоді арифметична складність* P *не більше*

$$|\mathcal{T}| \cdot |L| \cdot (3\log|L| + 5) + 2|L_{cmp}| \cdot (T_{arith}(\Phi) + |\Phi| + 1) \qquad (2.2)$$

де $T_{arith}(\Phi)$ *позначає суму арифметичних складностей множини обмежених* Φ*.*

б) V***:*** V^{AZI} *не вимагає арифметичної складності і передбачає лише вибірку* $2|\mathcal{T}| + |\Phi|$ *рівномірно випадкові елементи поля з* $\mathbb{F}$*.*

2.5.1 Про помилку надійності протоколу АЗІ

Імовірність того, що «погана подія», для якої не виконується жоден із пунктів 3а та 3б, дорівнює $\frac{1}{|\mathbb{F}|}$ приведений до нижньої межі похибки надійності якнайменше $1/|\mathbb{F}|$. Повторюючи крок 1б кілька k разів, кожен з яких незалежним випадковим $R_1, \ldots, R_k$ зменшує похибку до $\frac{1}{|\mathbb{F}|^k}$ Підкреслимо, що для константи k збільшення арифметичної складності доведення є мізерно малою. У складності доведення переважає обчислення пункту 1а, яке виконується лише один раз і вимагає

$O(|L|\log|L|)$ операцій (перший доданок рівняння 2.2); наступні кроки лінійні в $|L|$ (другий доданок рівняння 2.2).

Ми припускаємо, що помилка надійності протоколу АЗІ нижча, ніж зазначено вище в пункті 3. Зокрема, вважається, що знаменник 2ζ з лівого боку пунктів 3а і 3б не є щільним. НЗ-МПД використовує протокол FRI як РС-ІДОБО в протоколі АЗІ, таким чином, можна висловити наступне припущення щодо надійності АЗІ таким чином, що воно буде піддаватися аналізу надійності протоколу FRI.

Відстань по блоку – позначення Поняття блокової міри відстані використовується в FRI; див. [74]. Зокрема, для афінного простору $L \subset \mathbb{F}$ підпростору $L_0 \subset L$ розміру $|L_0| = 2^\eta$ визначається L_0-відстань по блоку між f,$g : L \to \mathbb{F}$ – частка класів S від L_0 в L, на якій $f|s \neq g|s$, і нехай через $\mathcal{S}(L_0, L)$ позначимо множину класів L_0, які містяться в L.

Зважаючи на це позначення, будемо говорити, що L_0 і L_0' є задовільними для екземпляра $\mathtt{x} = (\mathbb{F}, \mathcal{T}, \mathcal{N}, \Phi, L, L_{cmp}, \vec{\rho}, \rho_{cmp})$ якщо:

1) $L_0 \subset L$ і $L_0' \subset L_{cmp}$;

2) для кожного суміжного $N \in \mathcal{N}$ і підмножини $\hat{\mathcal{S}} \subseteq \mathcal{S}(L_0', L_{cmp})$ виконується:

$$\left|\left\{S \in \mathcal{S}(L_0,L) | S \cap N\left(\bigcup_{\hat{S}\in\hat{\mathcal{S}}} \hat{\mathcal{S}}\right)\right\}\right| \geq \left|\bigcup_{\hat{S}\in\hat{\mathcal{S}}} \hat{\mathcal{S}}\right|$$

.

Неформально ця умова означає, що спроба змінити всі записи об'єднання класів в $\hat{\mathcal{S}}$ шляхом зміни записів $\mathtt{w}^*$ в наборі $N\left(\bigcup_{\hat{S}\in\hat{\mathcal{S}}} \hat{\mathcal{S}}\right)$ змінить принаймні стільки класів $\mathcal{S}(L_0,L)$ скільки є елементи в $\bigcup_{\hat{S}\in\hat{\mathcal{S}}} \hat{\mathcal{S}}$

Гіпотеза 2.1. *Припустимо* $\mathtt{x} = (\mathbb{F}, \mathcal{T}, \mathcal{N}, \Phi, L, L_{cmp}, \vec{\rho}, \rho_{cmp})$ *є незадовільним екземпляром АЗІ з* δ *відстанню і нехай* L_0, L_0' *є задовільними для* $\mathtt{x}$ *і задовольняють* $|L_0| = |L_0'| = 2^\eta$. *Нехай через* $\delta^{(0)}(f^{(0)})$ *позначається відстань* $f^{(0)}$ *від* $RS[\mathbb{F}, L_{cmp}, \rho_{max}]$, *використовуючи блокову міру відстані, викликаної* L_0'. *Аналогічно, нехай* $\delta^{(0)}(g^{(0)})$ *позначає відстань* $g^{(0)}$ *для* $RS[\mathbb{F}, L_{cmp}, \rho_{cmp}]$ *за допомогою*

блокової міри відстані, викликаної L_0'.

Зверніть увагу, $\delta^{(0)}(f^{(0)})$ і $\delta^{(0)}(g^{(0)})$ є випадковими величинами, які залежать від випадковості V R. Тоді:

$$Pr_R\left[\delta^{(0)}(f^{(0)}) < \min\left\{1-\rho_{\max}, \frac{2^{\eta}|L_{cmp}|}{|L|}\cdot\left(\delta-\delta^{(0)}(g^{(0)})\right)\right\}\right] \leq \frac{1}{|\mathbb{F}|} \quad (2.3)$$

Для практичних цілей безпеки може вистачити більш слабкої гіпотези, як наведена нижче. Надалі *псевдодоказник* P^* є поліноміально обмеженою рандомізованою машиною, яка діє як доведення у протоколі АЗІ, що вказано вище.

Гіпотеза 2.2. *Для кожного псевдодоказника* P^*
та досить великого екземпляра $\mathbb{x} = (\mathbb{F}, \mathcal{T}, \mathcal{N}, \Phi, L, L_{cmp}, \vec{\rho}, \rho_{cmp})$

$$Pr\left[\delta^{(0)}(f^{(0)}) < 1-\rho_{\max} \wedge \delta^{(0)}(g^{(0)}) < 1-\rho_{cmp}\right] \leq \frac{1}{|\mathbb{F}|} \quad (2.4)$$

Наведена вище ймовірність перевищує випадковість обох P^* і V.

2.6 Інтерактивне доведення близькості оракулом (ІДБО) та FRI-процедура

Згадаємо основні результати протоколу FRI – інтерактивного доведення близькості оракулом (ІДБО), як зазначено в [74].

Теорема 2.5. *(Властивості FRI). Сімейство кодів РС швидкості* $\rho = 2^{-\mathcal{R}}, \mathcal{R} \geq 2, \mathcal{R} \in \mathbb{N}$ *має ІДБО (FRI) з такими властивостями, де* $N = |L|$ *позначає довжину блоку (яка дорівнює введеній довжині вхідних даних від* P *для фіксованого коду* $RS[\mathbb{F},L,\rho]$:

– ***Розклад та раунди:*** *Протокол має дві фази: під час першого етапу (COMMIT)* V *надсилає публічну випадковість і* P*, посилає*

оракули; складність раунду є $\lceil \frac{(\log N)-\mathcal{R}}{2} \rceil$. *Під час другого етапу (QUERY)* V *запитує оракули і приймає рішення (*P *не бере участі в цій фазі);*

– P***:*** *складність* P *потребує менше ніж* $6N$ *арифметичних операцій у* $\mathbb{F}$*; Довжина доведення менша за* $N/3$ *елементів поля, а складність раунду – не більше за* $\frac{\log N}{2}$*;*

– V***:*** *для параметра повторення запиту* $\ell \in \mathbb{N}^+$ *складність запиту* $2\ell \log N$ *рішення* V *обчислюється з використанням щонайбільше* $\ell \cdot 21 \log N$ *арифметичних операцій над* $\mathbb{F}$*;*

– ***Вагомість:*** *Існує* $\delta_0 \geq \frac{1}{4}(1-3\rho) - \frac{1}{\sqrt{N}}$ *такий, що кожне* f*, яке є* δ*-віддаленим за відносною відстанню Хеммінга від коду, приймається з імовірністю щонайбільше*

$$\mathbf{err}_{FRI}(\delta) \triangleq \frac{3N}{|\mathbb{F}|} + (1 - \min\{\delta, \delta_0\})^{\ell}, \tag{2.5}$$

де $\ell \in \mathbb{N}^+$ *є параметром повторення, визначеним вище.*

– ***Паралелізація:*** *Кожне повідомлення від* P *може бути обчислене за час* $O(1)$ *на машині з довільним паралельним доступом (МДПД) із загальним читанням і виключним записом (ЗЧВЗ), припускаючи, що одна арифметична операція* $\mathbb{F}$ *займає одиницю часу.*

Ми припускаємо, що надійність майже дорівнює мірі відстані по блоку $\delta^{(0)}$*, для будь-якого значення відстані* $\delta^{(0)}$*, навіть тоді, коли* $\delta^{(0)} \approx 1 - \rho$ *Наступна гіпотеза формалізує це; див. [74] для обговорення цього.*

Гіпотеза 2.3. *Використовуючи позначення теореми 2.5, для будь-якого* $f : L \to \mathbb{F}$ *блокової відстані* $\delta^{(0)}$ *від* $RS[\mathbb{F},L,\rho], L = 2^k$ *і* $\upsilon \in [0, \delta^{(0)}]$ *принаймні з ймовірністю*

$$1 - \frac{(k)^2 \cdot 2^{\eta}}{\upsilon \cdot \eta^2 \cdot |\mathbb{F}|} \tag{2.6}$$

через випадковість V *під час фази COMMIT, ймовірність*

відхилення під час фази QUERY становить принаймні $1 - \delta^{(0)}(1 - \upsilon)$.

Висновки до розділу 2

Поточна реалізація коду для ВПД використовує параметри на стороні перевіряючого параметра з леми 2.2. ІДО перетворюється на прозору систему аргументів з обчислювальними нульовими знаннями за допомогою криптографічного компілятора Kilian/Micali. Поточний код для P реалізує обидва скорочення, описані в теоремах 3.4 і 3.5 [75]. Робота зосереджується на програмі ВПД, яка потребує невеликого простору, тому перша система в цьому випадку є більш ефективною.

Для хеш-функції, яка використовується для побудови дерев Меркла як комітментів для оракулів, використовується конструкція Девіса–Меєра, створену за допомогою AES128. Це дає оцінений параметр стійкості до колізій у 64 біти.

Використовуючи позначення в доведенні леми 2.2, було встановлено $\mathcal{R} = 3$ і $\lambda = 60$, що призводить до помилки безпеки не більше ніж $err \leq 2^{-60}$. Двійкове поле, яке використовується, має $|\mathbb{F}| = 2^{64}$ тобто $n = 64$. Ступінь системи обмежень дорівнює 8, отже $d = 3$. Закріплюємо параметр НЗ на k =1; це забезпечує нульове знання для $t \geq 8$, це отримується для розмірів наборів даних $n \geq 2^3$ тому що $t = [n \cdot 62]$. Для всіх менших наборів даних налаштування $k = 3$ буде достатнім для НЗ, але оскільки робота зосереджена на великих наборах даних, він не був реалізований.

Параметром повторення для числа FRI-COMMIT є $s = 2$ і для FRI-QUERY, ми маємо $\ell_{f^{(0)}} = 9$ і $\ell_{g^{(0)}} = 22$.

3 ПОБУДОВА ЗАГАЛЬНОЇ МАТЕМАТИЧНОЇ МОДЕЛІ STARK-ДОВЕДЕННЯ ТА ЇЇ АНАЛІЗ

Існує багато робіт на тему STARK доведень, які розглядають ти чи інші властивості цієї системи та застосування у сучасному світі. Тим не менш, у жодній із цих робіт ще не було запропоновано загальної математичної моделі побудови доведення типу STARK, яке було б універсальним для найбільш типових задач. Відповідно, у цьому розділі ми спробуємо в загальному вигляді описати математичну модель STARK-доведення, і побудувати її покроковий алгоритм. Але перед цим ми підсумуємо основні властивості STARK-доведення та розглянемо приклади. І вже після детального аналізу різних прикладів, ми підійдемо до дого, які основні моменти загального STARK-доведення, яка їх послідовність і специфіка вкористання для кожного прикладу.

Попередньо зазначимо, що STARK-доведення буде складатися із двох частин:

1) побудова комітментів згідно із заданими співвідношеннями;

2) доведення того, що степінь поліному обмежена зверху деяким числом, і під час цього доведення (що степінь обмежена) ми будемо використовувати FRI процедуру, яка детально описана у [85].

3.1 Загальні відомості про доведення типу STARK

У сучасному світі інколи стоїть задача перевірки цілісності обчислень. Одним із методів розв'язання цієї задачі є так звані STARK-доведення, яке розшифровується як «Scalable Transparent ARgument of Knowledge», яке можна перевести як «масштабований прозорий аргумент знань». Він багато чим схожий на доведення типу SNARK та Bulletproof, але у нього є декілька суттєвих відмінностей:

– постквантовість (доведення типу SNARK та Bulletproof не є постквантовими);

– створення доведення сторони P, яка доводить :

- у 10 разів швидше за схему типу SNARK;
- у 10 разів швидше за схему типу Bulletproof;

– перевірка доведення верифікатором V:

- у 10 разів швидше за схему типу SNARK;
- у 10 разів швидше за схему типу Bulletproof;

– довжина доведення:

- у 100 разів більше за доведення у схемі типу SNARK;
- у 20 разів більше за доведення у схемі типу Bulletproof;

– нема потреби у довіреній стороні;

– нема потреби у так званій множині значень SETUP, завдяки якій (наприклад, у схемі SNARK) створюються та перевіряються доведення.

Доведення типу STARK вимагають лише два наступних припущення:

– існує криптографічно стійка геш-функція (це припущення необхідне як для побудови інтерактивного STARK, так і неінтерактивного на основі інтерактивного) – для інтерактивної схеми iSTARK;

– для не інтерактивної схеми nSTARK вважаємо, що виконується «евристика Фіата-Шаміра»; це припущення полягає у тому, що деякий факт може бути публічно доведено без розкриття основної інформації.

Схема доведень STARK має дві основні властивості:

– *повнота*: якщо P дійсно володіє секретом, то завжди існує спосіб це довести V;

– *вагомість*: якщо P не володіє секретом, але намагається довести V, що володіє, то імовірність його переконати є нехтувано малою, тобто її можна як завгодно близько наблизити до нуля (наприклад, імовірність можна зробити меншою за $\frac{1}{2^{128}}$).

3.2 Математичне підґрунтя для схеми доведень типу STARK

Наведемо ті означення і твердження, які будемо використовувати долі при побудові загальної схеми STARK-доведення. Отже, нехай:

1) $\mathbb{F}_p$ будемо позначати просте поле характеристики , де p – просте число;

2) мультиплікативна група поля складається з усіх його ненульових елементів: $\mathbb{F}_p^* = \mathbb{F}_p \backslash \{0\}$;

3) $|\mathbb{F}_p^*| = p - 1$;

4) з теореми про властивості циклічної групи та теореми Лагранжа випливає, що $\mathbb{F}_p^*$ – циклічна група $\Rightarrow \forall q | (p-1)\ \exists \mathbf{G} \subset \mathbb{F}_p^* : |\mathbf{G}| = q$;

5) $\mathbf{G} \subset \mathbb{F}_p^* \Rightarrow \mathbf{G}$ – також циклічна, тобто $\exists g$ – генератор $\mathbf{G}$ такий, що $\mathbf{G} = \{g^0, g^1, \ldots, g^{q-1}\}$; $g^q = 1$;

6) із попереднього пункту маємо, що $\forall a \in \mathbf{G} : a^q = (g^k)^q = 1$;

7) $f(x) \in \mathbb{F}_p[X] \Rightarrow$ кількість коренів цього поліному $\mathbb{F}_p$, які належать полю, не перевищує $deg f$;

8) $f(x) = (x-1)(x-g)(x-g^2)\ldots(x-g^{q-1}) = (x^q - 1)$, тобто якщо степені однакові та корені поліномів однакові, і ці поліноми нормовані, тоді поліноми рівні;

9) теорема Безу: $f(a) = 0 \Leftrightarrow f(x) \vdots (x-a)$;

10) узагальнення теореми Безу: $a_1, \ldots, a_l$ – корені, $k_1, \ldots, k_l$ – кратності $\Rightarrow f(x) \vdots (x-a_1)_1^k \times \cdots \times (x-a_l)_l^k$;

11) інтерполяційна формула Лагранжа: $a_0, \ldots, a_n \in \mathbb{F}_p$ – різні, $b_0, \ldots, b_n \in \mathbb{F}_p$ – довільні $\Rightarrow \exists! f(x) \in \mathbb{F}_p$:

а) $f(a_i) = b_i, i = \overline{1,\ n}$;

б) $deg f \leq n$;

(формула, щоб побудувати такий поліном: $f(x) = \sum_{i=0}^{n} b_i \times \left\{ \prod_{j=0, j\neq i}^{n} \frac{x-a_j}{a_i - a_j} \right\}$).

3.3 Варіанти побудови STARK-доведень для різних типів задач

У цьому розділі ми розглянемо приклади побудови поліномів з певними властивостями, які характеризують істинність деяких тверджень або коректність виконання відповідних обчислень.

Варіант 3.1. Гіпотеза Коллатца. Нехай

$$a_0;\ a_i = \begin{cases} a_{i-1} : 2, \text{якщо } a_i \text{ – парне} \\ 3a_{i-1} + 1, \text{в іншому випадку} \end{cases}.$$

Гіпотеза Коллатца полягає у наступному:

$$\forall a_0\ \exists j \geq 1 : a_j = 1.$$

Сформулюємо приклад твердження на основі вищенаведеної гіпотези, яке можна довести, використовуючи схему STARK: «Послідовність Коллатца із $a_0 = 52$ набуває значення 1 після 11 ітерацій (тобто, $a_{11} = 1$).»

Тепер побудуємо доведення цього твердження. Отже, нехай

$a_0, a_1, \ldots, a_n$ – послідовність Коллатца;

A_i – бітове подання відповідного числа a_i (у форматі little endian – LSB зліва);

$A = \begin{pmatrix} A_0 \\ A_1 \\ \vdots \\ A_{n-1} \end{pmatrix}$ – бітова матриця розміру $n \times m$; так званий «слід» або «виконуваний слід».

A_{ij} – j-ий (зліва) біт в A_i, $\overline{j = 0,\ m-1}$;

$A_i = \underbrace{(A_{i,0}, A_{i,1}, \ldots, A_{i,m-1})}_{m \text{ бітів}}$ – m повинно бути достатньо великим, щоб можна було б записати $a_0, \ldots, a_n$.

Побудуємо умови у вигляді відповідних поліномів:

1) $\sum_{j=0}^{m-1} 2^i A_{0,j} = 0$ – початкова умова для a_0;

2) $\sum_{j=0}^{m-1} 2^i A_{n-1,j-1} = 0$ – тобто, останній ряд дорівнює 1;

3) $\forall 0 \le i \le n-1, \forall 0 \le j \le m-1 : A_{ij}^2 - A_{ij} = 0$ – тобто A_{ij} – біти;

4)

$$\forall 0 \le i \le n-2 : A_{i,0}(3 \cdot \sum_{j=0}^{m-1} 2^i A_{0,j} + 1 - \sum_{j=0}^{m-1} 2^i A_{i+1,j}) +$$

$$+(1 - A_{i,0}) \cdot (\sum_{j=0}^{m-1} 2^j A_{i,j} - 2 \cdot (\sum_{j=0}^{m-1} 2^j A_{i+1,j})) = 0.$$

Тепер продемонструємо арифметизацію у схемі STARK:

1) потрібно згенерувати виконуваний слід (ВС) та поліноміальні умови (ПУ) (P та V взаємопогоджують створені поліноміальні умови);

2) P у наступних ітераціях намагається довести V, що ПУ задовольняють означений ВС (зазначимо, що *P не показує сам ВС*).

Сформулюємо твердження, яке буде доводитись: «Обчислення виконано коректно, тобто отриманий результат є вірним.»

У цьому випадку ВС представлено таблицею, яка демонструє кроки, які є попереднім підґрунтям обчислень, тобто, один рядок є одним кроком.

Таким чином у прикладі (3.1) були наступні перетворення:

– твердження $\Rightarrow$ ВС;

– ВС $\Rightarrow$ ПУ.

У наступному прикладі (3.2) будуть наступні перетворення: ВС та ПУ $\Rightarrow$ поліном $f(a)$ та залежність, що $degf$ є малою $\Leftrightarrow$ ВС задовольняє ПУ.

Зауваження. Зазначимо, що $f(x),\ g(x) \in \mathbb{F}_p[X]$, $degf = degg = d$, $d \ll p$. А для випадкового $a \in \mathbb{F}_p[X]:\ P\{f(a) = g(a)\} \approx 0\ (\le \frac{d}{p})$.

Перед тим, як перейти до наступного прикладу, потрібно зазначити цілі, які мають буди досягнуті при розборі прикладу:

1) перевести ВС та ПУ у поліноми;

2) розширити поліном більшої області визначення (тобто, коли по поліном все побудований по вказаним точкам, а потім розширити область визначення);

3) перетворення створеного полінома в інший поліном, використовуючи ПУ;

4) продемонструвати залежність, що степінь створеного нового поліному є малою $\Leftrightarrow$ ВС є коректним.

Отже, коли цілі поставлені, перейдемо до наступного прикладу.

Варіант 3.2. Булевий ВС. Сформулюємо твердження, яке буде доводитись: «Р має послідовність із 512 чисел. Кожне із цих чисел належіть множині {0,1}.»

Тепер у V стоїть задача перевірити це твердження, знаючи значення обмеженої кількості цих чисел, яка набагато менше ніж 512.

$$\underline{\text{ВС}}: \left.\begin{matrix} \text{A}_0 \\ \text{A}_1 \\ \vdots \\ \text{A}_{511} \end{matrix}\right\} \text{по 1 біту у рядку (тобто, матриця розміру } 512 \times 1).$$

$\underline{\text{ПУ}}$: $A_i^2 - A_i = 0$.

Розпишемо алгоритм, за яким ВС та ПУ можна перевести у поліном:

1) Необхідно обрати скінченне поле $\mathbb{Z}_p$, $p = 96769$. Може скластися питання, чому обране саме таке значення p? Відповідь проста: $p - 1 \vdots 512 \Rightarrow \exists \mathbf{G} \subset \mathbb{Z}_p^* : |\mathbf{G}| = 512$.

2) Отже, обрали $\mathbf{G} \subset \mathbb{Z}_p^* : |\mathbf{G}| = 512$ та встановили, що g є генератором $\mathbf{G}$.

3) Тепер потрібно використати формулу Лагранжа для побудови

відповідного поліному $f(x)$, де $f(g^i) = A_i$, $i = \overline{0{,}511}$, $degf < 512$ (щоб розширити поліном $f(x)$ до більшої області визначення буде використовуватись коди Ріда-Соломона).

4) Далі потрібно використати $f(x)$, щоб створити інший поліном $p(x)$ такий, що буде виконуватись твердження: $deg\ p(x)$ буде «мала» $\Leftrightarrow$ ВС задовольняє ПУ.

Із узагальнення теореми Безу маємо: якщо $a_0, \ldots, a_{k-1}$ – корені полінома

$p(x) \Rightarrow \exists q(x) :\ p(x) = q(x) \cdot \prod_{i=0}^{k-1}(x - a_i),\ \forall x \neq a_i :\ q(x) = \frac{p(x)}{\prod_{i=0}^{k-1}(x-a_i)}$.

Тепер визначимо такий поліном $f(x)$, для якого буде виконуватись наступне: $f(g^i) = A_i$, $i = \overline{0{,}511}$, $degf \leq 512$ (використовуючи формулу Лагранжа): якщо $A_i \in \{0{,}1\} \Rightarrow \underbrace{f(g^i)^2}_{A_i^2} - \underbrace{f(g^i)}_{A_i} = 0 \Rightarrow g_i$, $i = \overline{0{,}511}$ – корені $f(x)^2 - f(x) \Rightarrow$ можна визначити такий $p(x) = \frac{f(g^i)^2 - f(g^i)}{\prod_{i=0}^{511}(x-g^i)}$, та $(\exists q(x) : f(x)^2 - f(x) = g(x) \cdot \prod_{i=0}^{511}(x - g^i)$, $q(x) = p(x)$, $x \neq g^i$, $i = \overline{0, 511}$, $deg\ q(x) \leq 2deg\ f - 512) \Leftrightarrow$ ВС – це список із 512 бітів.

Розглянемо наступний приклад із числами Фібоначчі.

Варіант 3.3. Числа Фібоначчі

Нехай маємо наступні параметри: $\mathbb{Z}_{\underbrace{96769}_{p}}$, $a_0 = a_1 = 1$, $a_{n+2} = (a_{n-1} + a_n) mod p$.

Нехай твердженням, яке потрібно буде довести, буде той факт, що $a_{511} = 62215$. Тоді ВС буде представлений наступною матрицею:

$$A = \begin{pmatrix} 1 \\ 1 \\ 2 \\ 3 \\ 5 \\ \dots \\ 26646 \\ 66169 \\ 92815 \\ 62215 \end{pmatrix},$$

де, відповідно, $g^0 = 1$, $g^1 = 1$, $g^2 = 2$, $\dots$, $g^{511} = 62215$.

А ПУ буде наступним:

1) $A_0 - 1 = 0$;

2) $A_1 - 1 = 0$;

3) $A_{i+2} - A_{i+1} - A_i = 0$, $i = \overline{2,509}$;

4) $A_{511} - 62215 = 0$.

Тепер потрібно перевести ці рівності у поліноміальні вирази так, що $f(x) : \forall i \in \{0,\dots,511\} : f(g^i) = A_i$. І потім виражаємо ПУ в термінах створеного полінома $f(x)$, тобто:

1) $f(1) - 1 = 0$;

2) $f(g) - 1 = 0$;

3) $\forall i \in \{0,\dots,511\} : f(g^{i+2}) - f(g^{i+1}) - f(g^i) = 0$;

4) $f(g^{511}) - 62215 = 0$.

Пункти 1), 2) та 4) є граничними умовами. Пункт 3) – це та ж сама

умова, що

$$\forall x \in \{ \underbrace{0,\, g,\, \ldots,\, g^{511}}_{\text{«номера» перших 509-и рядків ВС матриці A}} \} : f(xg^2) - f(xq) - f(x) = 0.$$

Проведемо наступні перетворення:

$$f(xg^2) - f(xq) - f(x) = 0,\, x \in \{0,\, g,\, \ldots,\, g^{511}\} \Leftrightarrow$$
$$\Leftrightarrow \{0,\, g,\, \ldots,\, g^{511}\} \text{ – корені } f(xg^2) - f(xq) - f(x) \Leftrightarrow$$
$$\Leftrightarrow f(xg^2) - f(xq) - f(x) = q(x) \prod_{i=0}^{509} (x - g^i) \Leftrightarrow$$
$$\Leftrightarrow g(x) = \frac{f(xg^2) - f(xq) - f(x)}{\prod_{i=0}^{509}(x - g^i)},$$

де $x \neq g^i$, $deg\ q(x) \leq 2$ (так як $deg f \leq 512$).

Зауваження. Зазначимо, що виконується рівносильність: $q(x)$ – має «малу» степінь $\Leftrightarrow$ твердження, яке доводиться, є дійсним.

Отже, було продемонстровано як можна із задачі перевірки, що ПУ задовольняє деякому ВС зробити задачу перевірки, що деякий поліном (відомий P) має «малу» степінь.

Перевірка доведення для верифікатора V полягає у наступних його діях:

1) він запитує так званий композитний поліном, який є лінійною комбінацією заданих поліномів з випадково вибраними коефіцієнтами;

2) перевіряє властивість «малого» степеню базуючись на результатах попереднього запиту.

Розглянемо кожний із цих кроків більш детально.

– V запитує ВС для якогось випадкового рядка w шляхом запиту $f(w)$, $f(gw)$, $f(g^2w)$ – три послідовні рядки (щоб сторона, яка доводить, P не могла «підсунути» інший поліном «малого» степіню).

– V обчислює $g(w) = \frac{f(g^2w) - f(gw) - f(w)}{\prod_{i=0}^{509}(x - g^i)} =$

$= \frac{(f(g^2w)-f(gw)-f(w))(w-g^{510})(w-g^{511})}{w^{512}-1}$, де $\prod_{i=0}^{509}(x-g^i) = \frac{w^{512}-1}{(w-g^{510})(w-g^{511})}$.

V може виконати ці обчислення за поліноміальний час. Також потрібно зазначити, що якщо у ВС є більш ніж одна колонка, то використовується кілька поліномів, для яких потім будується їх випадкова лінійна комбінація. Із цією випадковою лінійною комбінацією далі працюємо як з одним поліномом. А коефіцієнти до неї в інтерактивній версії STARK надсилає V, а в неінтерактивній вони будуються згідно з евристикою Фіата-Шаміра. Далі вже залишається перевірка на наявність «малого» степеню.

Розглянемо ще один приклад із використанням квадратичних чисел Фібоначчі.

Варіант 3.4. Квадратичні числа Фібоначчі.

Нехай задані деякі значення a_0, a_1, а всі інші елементи послідовності отримуються згідно з наступною формулою: $a_{i+2} = a_{i+1}^2 + a_i^2$. Ця послідовність буде дуже швидко зростати, тому кожний новий член послідовності буде обчислюватись за $mod\ p$, де $p = 3 \cdot 2^{80}$.

Нехай твердженням P, яке потрібно довести, виглядає так: $\exists x:\ a_0 = 1, a_1 = x\ \Rightarrow a_{1022} = 2338775057$.

Перша частина STARK доведення формується із двох кроків:

1) побудова LDE, тобто розширення «малого» степеня;

2) побудова комітментів.

Крок LDE у свою чергу містить наступні три кроки:

1) генерація вхідних даних;

2) інтерполяція;

3) вибір великого розширення (не менше, ніж у вісім разів більше).

Розглянемо кожний із цих кроків більш детально на прикладі запропонованого раніше твердження.

1) **генерація вхідних даних**:

$$\left.\begin{matrix} \mathrm{a}_0 \\ \mathrm{a}_1 \\ \vdots \\ \mathrm{a}_{1022} \end{matrix}\right\} \text{– виконуваний слід;}$$

2) **інтерполяція**: потрібно інтерполювати $f(x)$: $f(x_i) = a_i$, $i = \overline{0{,}1022}$; для вибору x_i на цьому кроці потрібно обрати $\mathbf{G} \subset \mathbb{F}_p^*$: $|\mathbf{G}| = q \geq 1023$; потрібно обрати g, яке є генератором $\mathbf{G}$; $x_i = g^i (mod) p$; $f(g^i) = a_i$, $i = \overline{0{,}1022}$ – поліноміальний слід;

3) **розширення**: розширюємо вихідну підгрупу $\mathbf{G}$ до більшої групи, яка є більшою як у мінімум 8 разів: $\mathbf{G} \subset \mathbf{H} = \{\acute{x}_j^i\}_{j=0}^{8191} = \{w, wh, wh^2, \ldots, wh^{8191}\}$, w,$h \in \mathbb{F}$, h – генератор $\mathbf{H}$; потрім потрібно обчислити кожний $f(\acute{x}_j)$, $j \geq 1$, використовуючи коди Ріда-Соломона.

Розглянемо тепер комітмент LDE (дерево Меркла). Отже, тільки що було обчислено $f(wh^0), \ldots, f(wh^{8191})$. Створимо дерево Меркла:

Повертаючись до нашого прикладу. Послідовність $\{a_i\}$ має задовольняти наступним рівностям:

$$\begin{cases} a_0 = 1; \\ a_{1022} = 2338775057; \\ a_{i+2} = a_{i+1}^2 + a_i^2. \end{cases} \tag{3.1}$$

Якщо (3.1) виконується для a_i, то твердження, сформульоване на початку прикладу є вірним.

Побудуємо такий поліном $f(x)$ та три раціональні функції $p_0(x)$, $p_1(x)$, $p_2(x)$, які відповідають умовам (3.1). Виконання умов (3.1) запишемо у еквівалентному вигляді із використанням $f(x)$, $p_0(x)$, $p_1(x)$, $p_2(x)$.

Отже, першим кроком буде переведення послідовності $\{a_n\} \Rightarrow f(x)$.

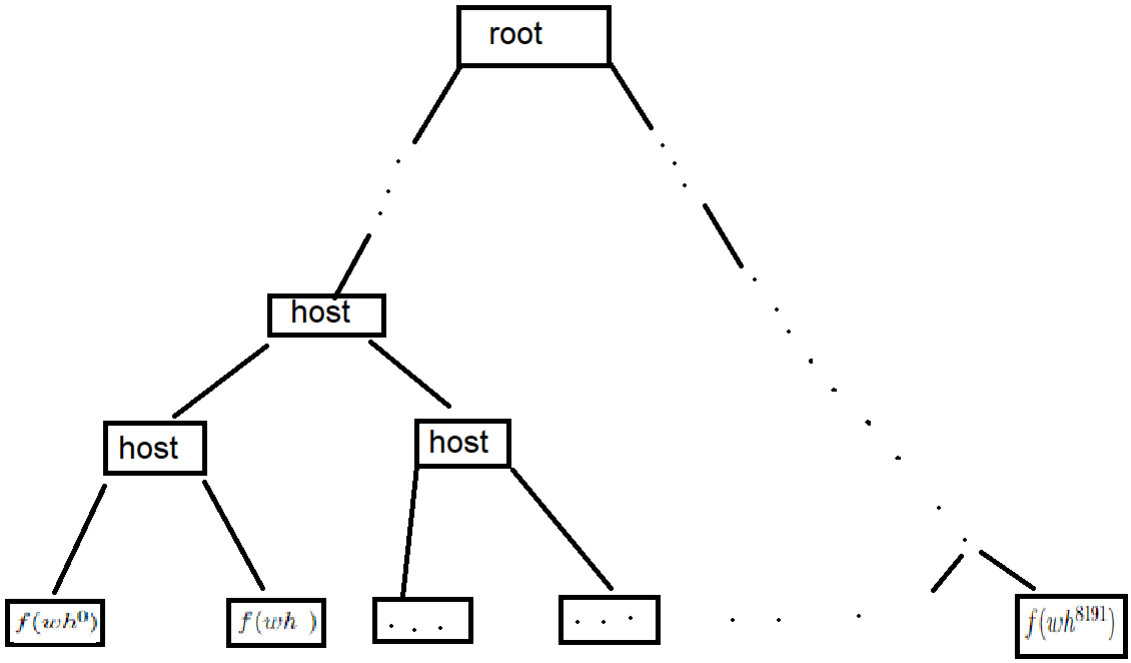

Рисунок 3.1 – Дерево Меркла

Побудуємо поліноміальні умови (для поліноміального сліду $f(g^i) = a_i$):

$$a_0 = 1 \Rightarrow f(g^0) = f(1) = 1;$$

$$a_{1022} = \cdots \Rightarrow f(g^{1022}) = \ldots;$$

$$a_{i+1} = a_{i+1}^2 + a_i^2 \Leftrightarrow f(g^2 g g^i) = f(g \cdot g^i)^2 + f(g^i)^2 \Leftrightarrow$$

$$\Leftrightarrow f(g^2 x) = f(gx)^2 + f(x)^2, x \in \{g^0, \ldots, g^{1020}\}$$

$\Rightarrow$ маємо три обмеження $f(x)$. Таким чином, якщо $f(x)$ задовольняє заданій умові $\Rightarrow$ твердження є вірним.

Наступним кроком буде використання коренів поліному для запису відповідних умов:

$f(1) = 1 \Rightarrow$ 1 є коренем $f(x) - 1$;

$$\ldots\ldots g^{1022} \ldots\ldots f(x) - \ldots;$$

$$\ldots\ldots \{g^i\}_{i=0}^{1020} \ldots\ldots f(g^2x) - f(gx)^2 + f(x)^2;$$

Яко всі ці елементи є коренями $\Rightarrow$ твердження є вірним.

Перейдемо до третього кроку, де продемонструємо перехід від коренів до раціональних функцій. Якщо зазначені раніше числа є коренями, то наступні вирази є поліномами:

$$p_0(x) = \frac{f(x) - 1}{x - 1};$$

$$p_1(x) = \frac{f(x) - \ldots}{x - g^{1022}};$$

$$p_2(x) = \frac{f(g^2x) - f(gx)^2 + f(x)^2}{\prod_{i=0}^{1020}(x - g^i)} =$$

$$= \frac{(f(g^2x) - f(gx)^2 + f(x)^2)(x - g^{1021})(x - g^{1022})(x - g^{1023})}{x^{1024} - 1},$$

де $\prod_{i=0}^{1023}(x - g^i) = x^{1024} - 1$.

Таким чином, якщо вирази p_0, p_1, p_2 є поліномами $\Rightarrow$ твердження є вірним.

Також, якщо вирази p_0, p_1, p_2 є поліномами $\Rightarrow \forall$ випадково обраних a_0, a_1, a_2: композиція поліномів (погоджена із деревом Меркла) $a_0p_0(x) + a_1p_1(x) + a_2p_2(x)$ буде поліномом.

В іншу сторону твердження теж буде виконуватись із високою імовірністю, але тільки якщо a_0, a_1, a_2 будуть обрані випадково.

3.4 Формалізована модель STARK-доведення

У цьому підрозділі буде представлений основний результат цієї роботи, у якому ми побудуємо загальну модель для типових інтерактивних доведень типу STARK. Як зазначалося раніше, у світі є багато робіт, призначених питанню STARK-доведень і їх використання на окремих прикладах при вирішуванні різних задач. Проте, досі немає жодної роботи, у якій був би хоча б один приклад формалізованого загального вигляду STARK-процедури (у тому числі загальний вигляд не був запропонований навіть автором STARK). Таким чином, у цьому підрозділі серед всіх робіт, присвячених STARK-протоколам, вперше розроблений і продемонстрований формалізований загальний вигляд STARK-процедури. Нижче ми продемонструємо, як виглядає цей алгоритм побудови STARK-протоколу.

Зауваження. Перш ніж ми наведемо алгоритм, зазначимо, що необхідною умовою для створення STARK-доведення є те, що повинна бути можливість запису виконуваного сліду (ВС). Це є необхідною і достатньою умовою. Тобто, STARK-доведення можна застосовувати для доведення коректності будь-яких обчислень, які можна формалізувати й представити у вигляді ВС.

Алгоритм 3.1. Загальний вигляд протоколу STARK-доведення.

1) Запишемо виконані обчислення, коректність яких буде доводитись, у вигляді ВС:

$$\underline{\text{ВС}}: \quad \left.\begin{array}{l} A_0 \\ A_1 \\ \vdots \\ A_l \end{array}\right\}$$

2) Задамо поліноми, які у фіксованих точках приймають значення, які відповідають ВС та побудуємо відповідне дерево Меркеля та комітмент, що відповідає кореню цього дерева для функції $f(x)$:

$$f(x): \ f(g_i) = A_i, \ |G| = q,$$
$$\mathbf{G} =< g > , \ G \subset \mathbb{F}_p^*, \ \mathbf{G} \subset \mathbf{H} \subset \mathbb{F}_p^*,$$

Після побудови полінома $f(x)$ відповідного дерева Меркеля, корінь цього дерева Меркеля у якості комітмента одразу ж надсилається V.

3) Сформулюємо умови, яким повинен задовольняти кожний A_i у термінах побудованих поліномів $f(g^i)$ відповідних раціональних функцій та їх коренів, з використанням теореми Безу:

$$p_0 = \frac{\dots}{(x - g^i)(x - \dots)};$$
$$\vdots$$
$$p_i(x) = \frac{\dots}{(x - g^j)(x - \dots)}.$$

4) V генерує випадково $\alpha_0, \ \dots, \ \alpha_i$, і надсилає їх запитом до P.

5) P будує випадкову лінійну комбінацію поліномів

$$C_0(x) = \alpha_0 p_0(x) + \dots + \alpha_i p_i(x).$$

Та будує дерево Меркеля для цього поліному на більшій групі $\mathbf{H}$ (яка якнайменше у 8 разів більше за групу $\mathbf{G}$), і будує всі вузли дерева Меркеля, обчислює його корінь, і цей корінь знову надсилає V у якості комітмента.

6) V випадково генерує β_1 і надсилає його як запит до P.

7) P знову будує дерево Меркеля на групі $\mathbf{H}$, яка описана вище, обчислює його корінь і у якості комітмента надсилає V.

8) На цьому кроці використовується FRI процедура [85]. На кожному кроці FRI процедури V надсилає відповідні запити запити β_{2_1}, β_{2_2}, $\beta_{2_3}, \dots$ до P. Кількість запитів приблизно дорівнює $\log_2 (deg \ C_1(x))$

9) На кожному кроці після отримання запиту від V, P будує

відповідний поліном $C_2(x)$, степінь якого вдвічі менший, ніж степінь попереднього поліному $C_1(x)$ завдяки FRI процедурі; а також $C_2(x)$ обчислює в кожному елементі групи $\mathbf{H}$, будує відповідне дерево, і P буде комітити тільки корінь цього дерева.

10) Процедури, описані на кроці 8) та 9) повторюються до того часу, доки P не отримає поліном нульового степіню, тобто $C_n \equiv const$. Кількість n-кроків приблизно дорівнює $\log_2 (deg\ C_1(x))$.

11) V випадково генерує запит v і надсилає його до P.

12) P будує $f(v)$, $f(vg)$, $f(vg^2)$ і надсилає всі ці значення V разом із відповідними шляхами, які ведуть до кореню дерева Меркеля. Цим кроком P підтверджує, що поліном $f(x)$ було побудовано коректно і він відповідає заявленому раніше комітменту, тобто кореню відповідного дерева Меркеля.

13) V за наданими значеннями і шляхами перевіряє, що при застосуванні наданих шляхів можна отримати той корінь дерева Меркеля, який був раніше закомічений P.

14) V обчислює $p_j(x)$, $j = \overline{0,i}$, $C_0(x)$ і відсилає запит P на отримання відповідних шляхів.

15) V обчислює значення $C_0(-x)$ і надсилає запит до P на шлях до нього. Всі отримані шляхи він перевіряє на відповідність кореню цього дерева Меркеля.

16) Процедура, описана у пункті 15) повторюється n разів, і закінчується перевіркою значення поліному нульового степіню.

17) Такі ж самі процедури виконуються для всіх тих поліномів, які були отримані за допомогою FRI процедури. На останньому етапі V доходить до константи $c_n^{'}$, яку він обчислив сам із попереднього поліному, і перевіряє, чи співпадає вона із константою, яку одразу закомітив P, тобто що $c_n^{'} = c_n$.

18) Якщо всі перевірки закінчились коректно, то V приймає це доведення, тобто, вважає його коректним. Якщо хоч якийсь етап не пройшов відповідну перевірку, то доведення вважається некоректним.

Описаний вище алгоритм відповідає інтерактивному протоколу

STARK-доведення. Для побудови неінтерактивного STARK-доведення використовується евристика Фіата-Шаміра [86]. Тобто, це означає, що всі запити V до P замінюються хеш-значеннями від всієї попередньої стенограми протоколу, і від всіх загальних параметрів протоколу. Це зроблено для того, щоб P не мав можливості підробити протокол доведення. Він не повинен мати можливість передбачити запит.

Висновки до розділу 3

У цьому розділі були розглянуті загальні відомості та приклади використання STARK-доведень для різних задач. Після опрацювання матеріалу, які були у попередніх розділах, ми побудували першу формалізовану загальну модель STARK-протоколу; показали, що є необхідною і достатньою умовою для побудови STARK-протоколу є можливість запису ВС. Хочемо ще раз підкреслити, що особливо зручно його використовувати, коли треба довести коректність однотипних обчислень. Розмір створеного STARK-доведення є приблизно $const \cdot \log_2 BC$, тобто перевірка суттєво швидша за виконання ВС. Таким чином, STARK-доведення можна використовувати у випадку, коли V замовляє у P якесь велике та довге обчислення, а потім може швидко верифікувати результат цього обчислення.

ВИСНОВКИ

Результати даної роботи можна окреслити наступними пунктами.

1) Був виконаний глибокий аналіз різних робіт по доведеннях без розголошення, різні типи, класичні задачі та різні підходи до вирішення тих чи інших задач.

2) Ґрунтуючись на проведеному аналізі, вперше був запропонований загальний алгоритм для побудови інтерактивного та неінтерактивного STARK-доведення, і показано, що STARK-доведення можуть бути застосовані для доведення будь-яких однотипних обчислень, які можна задати у вигляді виконуваного сліду (ВС). Ми показали, що розмір доведення буде дорівнювати $const \cdot \log_2 BC$, а швидкість доведення – $const \cdot \log_2 (deg\ C_1(x))$, де $C_1(x)$ – другий комітмент, створений на основі дерева Меркеля на більшій групі $\mathbf{H}$.

ПЕРЕЛІК ПОСИЛАНЬ

1. Satoshi Nakamoto. Bitcoin: a peer-to-peer electronic cash system [Електронний ресурс] – 2009. / Satoshi Nakamoto – Режим доступу: https://bitcoin.org/bitcoin.pdf.

2. Gilles Brassard, David Chaum, and Claude Crepeau. Minimum disclosure proofs of knowledge. Journal of Computer and System Sciences [Електронний ресурс] – 1988 – Режим доступу: https://www.sciencedirect.com/science/article/pii/0022000088900050.

3. Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. SIAM Journal on Computing [Електронний ресурс] – 1989 – Режим доступу: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.419.8132&rep=rep1&type=pdf.

4. Laszl o Babai, Lance Fortnow, and Carsten Lund. Nondeterministic exponential time has two-prover interactive protocols. In Proceedings of the 31st Annual Symposium on Foundations of Computer Science [Електронний ресурс] – Режим доступу: https://people.cs.uchicago.edu/~fortnow/papers/mip2.pdf.

5. Laszl o Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In Proceedings of the 23rd Annual ACM Symposium on Theory of Computing [Електронний ресурс] – 1991 – Режим доступу: https://www.researchgate.net/publication/221591854_Checking_Computations_in_Polylogarithmic_Time.

6. Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: a new characterization of NP [Електронний ресурс] – 1998 – Режим доступу: https://www.cs.umd.edu/~gasarch/TOPICS/pcp/AS.pdf.

7. Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems [Електронний ресурс] – 1998 – Режим доступу: https://madhu.seas.harvard.edu/papers/1992/almss-conf.pdf.

8. Joe Kilian. A note on efficient zero-knowledge proofs and arguments. [Електронний ресурс] – 1992 – Режим доступу: https://people.csail.mit.edu/vinodv/6892-Fall2013/efficientargs.pdf.

9. Silvio Micali. Computationally sound proofs. [Електронний ресурс] – 1992 – Режим доступу: https://epubs.siam.org/doi/abs/10.1137/S0097539795284959?journalCode=smjcat.

10. Cynthia Dwork, Uriel Feige, Joe Kilian, Moni Naor, and Shmuel Safra. Low communication 2-prover zero-knowledge proofs for NP. [Електронний ресурс] – 1992 – Режим доступу: https://www.iacr.org/cryptodb/data/paper.php?pubkey=1265.

11. Joe Kilian, Erez Petrank, and Gabor Tardos. Probabilistically checkable proofs with zero knowledge. [Електронний ресурс] – 1997 – Режим доступу: https://dl.acm.org/doi/10.1145/258533.258643.

12. Yuval Ishai, Mohammad Mahmoody, Amit Sahai, and David Xiao. On zero-knowledge PCPs: Limitations,simplifications, and applications. [Електронний ресурс] – 2015 – Режим доступу: http://www.cs.virginia.edu/~mohammad/files/papers/ZKPCPs-Full.pdf.

13. Amos Fiat and Adi Shamir. How to prove yourself: practical solutions to identification and signature problems. [Електронний ресурс] – 1986 – Режим доступу: https://link.springer.com/content/pdf/10.1007/3-540-47721-7_12.pdf.

14. Lily Chen, Lily Chen, Stephen Jordan, Yi-Kai Liu, Dustin Moody, Rene Peralta, Ray Perlner, and Daniel SmithTone. Report on post-quantum cryptography. [Електронний ресурс] – 2016 – Режим доступу: https://csrc.nist.gov/publications/detail/nistir/8105/final.

15. Mihir Bellare and Oded Goldreich. On defining proofs of knowledge. [Електронний ресурс] – 1993 – Режим доступу: https://www.wisdom.weizmann.ac.il/~oded/pok.html.

16. Michael Walfish and Andrew J. Blumberg. Verifying computations without reexecuting them. [Електронний ресурс] – 2015 – Режим доступу: https://dl.acm.org/doi/10.1145/2641562.

17. Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. Interactive Oracle Proofs. [Електронний ресурс] – 2016 – Режим доступу: https://www.iacr.org/archive/tcc2016b/99850156/99850156.pdf.

18. Omer Reingold, Guy N. Rothblum, and Ron D. Rothblum. Constant-round interactive proofs for delegatingcomputation. [Електронний ресурс] – 2016 – Режим доступу: https://epubs.siam.org/doi/10.1137/16M1096773.

19. Yael Kalai and Ran Raz. Interactive PCP [Електронний ресурс] – 2008 – Режим доступу: https://www.wisdom.weizmann.ac.il/~/ranraz/publications/Pipcp.pdf.

20. Eli Ben-Sasson, Alessandro Chiesa, Michael A. Forbes, Ariel Gabizon, Michael Riabzev, and Nicholas Spooner. On probabilistic checking in perfect zero knowledge. [Електронний ресурс] – 2016 – Режим доступу: https://arxiv.org/abs/1610.03798.

21. Eli Ben-Sasson, Alessandro Chiesa, Ariel Gabizon, Michael Riabzev, and Nicholas Spooner. Short interactive oracle proofs with constant query complexity, via composition and sumcheck. [Електронний ресурс] – 2016 – Режим доступу: https://eprint.iacr.org/2016/324.

22. Eli Ben-Sasson, Iddo Bentov, Alessandro Chiesa, Ariel Gabizon, Daniel Genkin, Matan Hamilis, Evgenya Pergament, Michael Riabzev, Mark Silberstein, Eran Tromer, and Madars Virza. Computational integrity with a public random string from quasi-linear PCPs. [Електронний ресурс] – 2016 – Режим доступу: https://eprint.iacr.org/2016/646.

23. Federal Bureau of Investigation. Combined dna index system (CODIS). [Електронний ресурс] – 2017 – Режим доступу: https://www.fbi.gov/services/laboratory/biometric-analysis/codis/codis-and-ndis-fact-sheet.

24. Nick Szabo. Formalizing and securing relationships on public networks. [Електронний ресурс] – 2017 – Режим доступу: http://myinstantid.com/szabo.pdf.

25. Eli Ben-Sasson. Universal and affordable computational integrity. [Електронний ресурс] – 2013 – Режим доступу: https://www.youtube.com/watch?v=Q4nWoEKUtgU&t=32s.

26. Michael Hearn. Corda and SGX: a privacy update. [Електронний ресурс] – 2017 – Режим доступу: https://www.corda.net/2017/06/corda-sgx-privacy-update/.

27. Arthur Breitman. Scaling tezos. [Електронний ресурс] – 2017 – Режим доступу: https://hackernoon.com/scaling-tezo-8de241dd91bd.

28. Protocol Labs. Filecoin: A decentralized storage network. [Електронний ресурс] – 2017 – Режим доступу: https://filecoin.io/filecoin.pdf.

29. Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from Bitcoin. [Електронний ресурс] – 2014 – Режим доступу: https://ieeexplore.ieee.org/document/6956581.

30. M. Peck. A blockchain currency that beat s bitcoin on privacy. [Електронний ресурс] – 2016 – Режим доступу: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7761864.

31. Daira Hopwood, Sean Bowe, Taylor Hornby, and Nathan Wilcox. [Електронний ресурс] – 2017 – Режим доступу: https://zips.z.cash/protocol/heartwood.pdf.

32. Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. Quadratic span programs and succinct NIZKs without PCPs. [Електронний ресурс] – 2013 – Режим доступу: https://link.springer.com/chapter/10.1007/978-3-642-38348-9_37.

33. Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, Eran Tromer, and Madars Virza. SNARKs for C: Verifying program executions succinctly and in zero knowledge. [Електронний ресурс] – 2013 – Режим доступу: https://www.iacr.org/archive/crypto2013/80420214/80420214.pdf.

34. Paul Valiant. Incrementally verifiable computation or proofs of knowledge imply time/space efficiency. [Електронний ресурс] – 2008 – Режим доступу: https://link.springer.com/chapter/10.1007/978-3-540-78524-8_1.

35. Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. Recursive composition and bootstrapping for SNARKs and proof-carrying data.

[Електронний ресурс] – 2013 – Режим доступу: https://eprint.iacr.org/2012/095.pdf.

36. Yuval Ishai, Eyal Kushilevitz, and Rafail Ostrovsky. Efficient arguments without short PCPs. [Електронний ресурс] – 2007 – Режим доступу: https://web.cs.ucla.edu/~rafail/PUBLIC/79.pdf.

37. Jens Groth. Short pairing-based non-interactive zero-knowledge arguments. [Електронний ресурс] – 2010 – Режим доступу: https://www.iacr.org/archive/asiacrypt2010/6477323/6477323.pdf.

38. Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. [Електронний ресурс] – 2008 – Режим доступу: https://link.springer.com/chapter/10.1007/978-3-540-78967-3_24.

39. Rosario Gennaro, Craig Gentry, and Bryan Parno. Non-interactive verifiable computing: Outsourcing computation to untrusted workers. [Електронний ресурс] – 2010 – Режим доступу: https://link.springer.com/chapter/10.1007/978-3-642-14623-7_25.

40. Helger Lipmaa. Progression-free sets and sublinear pairing-based non-interactive zero-knowledge arguments. [Електронний ресурс] – 2012 – Режим доступу: https://eprint.iacr.org/2011/009.pdf.

41. Nir Bitansky, Alessandro Chiesa, Yuval Ishai, Rafail Ostrovsky, and Omer Paneth. Succinct non-interactive arguments via linear interactive proofs. [Електронний ресурс] – 2013 – Режим доступу: https://eprint.iacr.org/2012/718.pdf.

42. Jens Groth. On the size of pairing-based non-interactive arguments. [Електронний ресурс] – 2016 – Режим доступу: https://eprint.iacr.org/2016/260.pdf.

43. Jens Groth and Mary Maller. Snarky signatures: Minimal signatures of knowledge from simulation-extractable snarks. [Електронний ресурс] – 2017 – Режим доступу: https://eprint.iacr.org/2017/540.pdf.

44. Brian Parno, Craig Gentry, Jon Howell, and Mariana Raykova. Pinocchio: Nearly practical verifiable computation. [Електронний ресурс] – 2013 – Режим доступу: https://eprint.iacr.org/2013/279.pdf.

45. SCIPR Lab. libsnark: a C++ library for zkSNARK proofs. [Електронний ресурс] – Режим доступу: https://github.com/scipr-lab/.

46. Srinath Setty, Andrew J. Blumberg, and Michael Walfish. Toward practical and unconditional verification of remote computations. [Електронний ресурс] – 2011 – Режим доступу: https://www.pepper-project.org/vercomp-hotos11.pdf.

47. Srinath Setty, Victor Vu, Nikhil Panpalia, Benjamin Braun, Andrew J. Blumberg, and Michael Walfish. Taking proof-based verified computation a few steps closer to practicality. [Електронний ресурс] – 2012 – Режим доступу: https://www.usenix.org/conference/usenixsecurity12.

48. Srinath Setty, Michael McPherson, Andrew J. Blumberg, and Michael Walfish. Making argument systems for outsourced computation practical (sometimes). [Електронний ресурс] – 2012 – Режим доступу: https://www.pepper-project.org/pepper-ndss12.pdf.

49. Srinath Setty, Benjamin Braun, Victor Vu, Andrew J. Blumberg, Bryan Parno, and Michael Walfish. Resolving the conflict between generality and plausibility in verified computation. [Електронний ресурс] – 2013 – Режим доступу: https://eprint.iacr.org/2012/622.pdf.

50. Eli Ben-Sasson, Alessandro Chiesa, Eran Tromer, and Madars Virza. Succinct non-interactive zero knowledge for a von Neumann architecture. [Електронний ресурс] – 2014 – Режим доступу: http://eprint.iacr.org/2013/879.

51. Riad S. Wahby, Srinath T. V. Setty, Zuocheng Ren, Andrew J. Blumberg, and Michael Walfish. Efficient RAM and control flow in verifiable outsourced computation. [Електронний ресурс] – 2015 – Режим доступу: https://eprint.iacr.org/2014/674.pdf.

52. George Danezis, Cedric Fournet, Jens Groth, and Markulf Kohlweiss. ´ Square Span Programs with Applications to Succinct NIZK Arguments. [Електронний ресурс] – 2014 – Режим доступу: https://eprint.iacr.org/2014/718.

53. Jens Groth. Efficient zero-knowledge arguments from two-tiered

homomorphic commitments. [Електронний ресурс] – 2011 – Режим доступу: https://www.iacr.org/archive/asiacrypt2011/70730421/70730421.pdf.

54. Jae Hong Seo. Round-efficient sub-linear zero-knowledge arguments for linear algebra. [Електронний ресурс] – 2011 – Режим доступу: https://link.springer.com/chapter/10.1007/978-3-642-19379-8_24.

55. Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Jens Groth, and Christophe Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. [Електронний ресурс] – 2016 – Режим доступу: https://eprint.iacr.org/2016/263.pdf.

56. Benedikt Bunz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. Bullet- ¨ proofs: Efficient range proofs for confidential transactions. [Електронний ресурс] – 2017 – Режим доступу: https://eprint.iacr.org/2017/1066.

57. Michael Ben-Or, Oded Goldreich, Shafi Goldwasser, Johan Hr astad, Joe Kilian, Silvio Micali, and Phillip Rogaway. Everything provable is provable in zero-knowledge. [Електронний ресурс] – 1988 – Режим доступу: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.70.4056&rep=rep1&type=pdf.

58. Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. Delegating computation: Interactive proofs for Muggles. [Електронний ресурс] – 2008 – Режим доступу: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/12/2008-DelegatingComputation.pdf.

59. Graham Cormode, Justin Thaler, and Ke Yi. Verifying computations with streaming interactive proofs. [Електронний ресурс] – 2011 – Режим доступу: https://arxiv.org/abs/1109.6882.

60. Graham Cormode, Michael Mitzenmacher, and Justin Thaler. Practical verified computation with streaming interactive proofs. [Електронний ресурс] – 2012 – Режим доступу: https://arxiv.org/abs/1105.2003.

61. Justin Thaler. Time-optimal interactive proofs for circuit evaluation. [Електронний ресурс] – 2012 – Режим доступу: https://eprint.iacr.org/2013/351.pdf.

62. Victor Vu, Srinath Setty, Andrew J. Blumberg, and Michael Walfish. A hybrid architecture for interactive verifiable computation. [Електронний ресурс] – 2013 – Режим доступу: https://ieeexplore.ieee.org/document/6547112.

63. Yupeng Zhang, Daniel Genkin, Jonathan Katz, Dimitrios Papadopoulos, and Charalampos Papamanthou. A zero-knowledge version of vsql. [Електронний ресурс] – 2017 – Режим доступу: https://eprint.iacr.org/2017/1146.

64. Riad S. Wahby, Ioanna Tzialla, abhi shelat, Justin Thaler, and Michael Walfish. Doubly-efficient zksnarks without trusted setup. [Електронний ресурс] – 2017 – Режим доступу: https://eprint.iacr.org/2017/1132.

65. Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge from secure multiparty computation. [Електронний ресурс] – 2007 – Режим доступу: https://web.cs.ucla.edu/~rafail/PUBLIC/77.pdf.

66. Irene Giacomelli, Jesper Madsen, and Claudio Orlandi. Zkboo: Faster zero-knowledge for boolean circuits. [Електронний ресурс] – 2016 – Режим доступу: https://eprint.iacr.org/2016/163.pdf.

67. Scott Ames, Carmit Hazay, Yuval Ishai, and Muthuramakrishnan Venkitasubramaniam. Ligero: Lightweight sublinear arguments without a trusted setup. [Електронний ресурс] – 2017 – Режим доступу: https://acmccs.github.io/papers/p2087-amesA.pdf.

68. Alessandro Chiesa and Eran Tromer. Proof-carrying data and hearsay arguments from signature cards. [Електронний ресурс] – 2010 – Режим доступу: https://people.eecs.berkeley.edu/~alexch/docs/CT10.pdf.

69. Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. Recursive composition and bootstrapping for SNARKs and proof-carrying data. [Електронний ресурс] – 2013 – Режим доступу: https://eprint.iacr.org/2012/095.pdf.

70. Eli Ben-Sasson, Alessandro Chiesa, Eran Tromer, and Madars Virza. Scalable zero knowledge via cycles of elliptic curves. [Електронний ресурс] –

2014 – Режим доступу: https://eprint.iacr.org/2014/595.pdf.

71. UV Linnik. On the least prime in an arithmetic progression. i. the basic theorem. [Електронний ресурс] – 1944 – Режим доступу: http://www.mathnet.ru/php/archive.phtml?wshow=paper&jrnid=sm&paperid=6196&option_lang=eng.

72. Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Jens Groth, and Christophe Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. [Електронний ресурс] – 2016 – Режим доступу: https://eprint.iacr.org/2016/263.pdf.

73. Eli Ben-Sasson, Alessandro Chiesa, Ariel Gabizon, and Madars Virza. Quasilinear-size zero knowledge from linear-algebraic PCPs. [Електронний ресурс] – 2016 – Режим доступу: https://eprint.iacr.org/2016/021.pdf.

74. Eli Ben-Sasson, Iddo Bentov, Ynon Horesh, and Michael Riabzev. Fast Reed-Solomon interactive oracle proofs of proximity (2nd revision). [Електронний ресурс] – 2017 – Режим доступу: https://drops.dagstuhl.de/opus/volltexte/2018/9018/pdf/LIPIcs-ICALP-2018-14.pdf.

75. Eli Ben-Sasson, Iddo Bentov, Ynon Horesh, and Michael Riabzev. Scalable, transparent, and post-quantum secure computational integrity. [Електронний ресурс] – 2018 – Режим доступу: https://eprint.iacr.org/2018/046.pdf.

76. Sian-Jheng Lin, Wei-Ho Chung, and Yunghsiang S. Han. Novel polynomial basis and its application to reedsolomon erasure codes. [Електронний ресурс] – 2014 – Режим доступу: https://arxiv.org/abs/1404.3458.

77. Evgenya Pergament. Algebraic RAM. Master's thesis, Technion. [Електронний ресурс] – 2017 – Режим доступу: http://www.cs.technion.ac.il/users/wwwb/cgi-bin/tr-get.cgi/2017/MSC/MSC-2017-04.pdf.

78. Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, and Eran Tromer. Fast reductions from RAMs to delegatable succinct constraint satisfaction problems. [Електронний ресурс] – 2013 – Режим доступу: https://eprint.iacr.org/2012/071.

79. Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, and Eran Tromer. On the concrete efficiency of probabilistically-checkable proofs. [Електронний ресурс] – 2013 – Режим доступу: https://www.researchgate.net/publication/262354053_On_the_concrete_efficiency_of_probabilistically-checkable_proofs.

80. Eli Ben-Sasson and Madhu Sudan. Short PCPs with polylog query complexity. [Електронний ресурс] – 2008 – Режим доступу: http://madhu.seas.harvard.edu/papers/2005/rspcpp-full.pdf.

81. Alessandro Chiesa and Zeyuan Allen Zhu. Shorter arithmetization of nondeterministic computations. [Електронний ресурс] – 2015 – Режим доступу: http://people.csail.mit.edu/zeyuan/paper/2015-arithmetization.pdf.

82. Alexander Polishchuk and Daniel A. Spielman. Nearly-linear size holographic proofs. [Електронний ресурс] – 1994 – Режим доступу: https://www.cs.yale.edu/homes/spielman/Research/holographic.html.

83. Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil Vadhan. Short PCPs verifiable in polylogarithmic time. [Електронний ресурс] – 2005 – Режим доступу: https://people.seas.harvard.edu/~salil/research/polylogPCP-abs.html.

84. Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil Vadhan. Short PCPs verifiable in polylogarithmic time. [Електронний ресурс] – 2005 – Режим доступу: https://people.seas.harvard.edu/~salil/research/polylogPCP-abs.html.

85. STARKWARE, Stark 101: Part 3 FRI Commitment. [Електронний ресурс] – Режим доступу: https://starkware.co/wp-content/uploads/2021/12/STARK101-Part3.pdf.

86. David Bernhard, Olivier Pereira, and Bogdan Warinschi, How not to Prove Yourself: Pitfalls of the Fiat-Shamir Heuristic and Applications to Helios. [Електронний ресурс] – Режим доступу: https://eprint.iacr.org/2016/771.pdf.