МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кваліфікаційна наукова праця
на правах рукопису

Ольшевська Віта Анатоліївна

УДК 512, 519.1

Дисертація

«АЛГОРИТМИ І КОДИ НАД СИЛОВСЬКИМИ 2-ПІДГРУПАМИ СИМЕТРИЧНИХ ГРУП S_{2^n}»

Спеціальність 113 - Прикладна математика
Галузь знань 11 - Математика та статистика

Подається на здобуття наукового ступеня доктора філософії.
Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

_____ В. А. Ольшевська

Науковий керівник: **Олійник Богдана Віталіївна**, доктор фізико-математичних наук, професор

Електронний підпис
ОЛЬШЕВСЬКА
ВІТА
АНАТОЛІЇВНА
Ідентифікаційний
код
3424309881
Підписано у вчасно

Київ — 2023

Анотація

Ольшевська В. А. Алгоритми і коди над силовськими 2-підгрупами симетричних груп S_{2^n}. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії в галузі знань 11 "Математика та статистика" за спеціальністю 113 "Прикладна математика" — Національний університет «Києво-Могилянська академія», Київ, 2023.

Дисертаційна робота присвячена побудові і аналізу алгоритмів над силовськими 2-підгрупами симетричних груп та дослідженню властивостей кодів на цих групах.

Силовські p-підгрупи є класичними об'єктами в теорії груп, опис таких підгруп допомагає зрозуміти структуру самої групи. Зокрема, словські p-підгрупи симетричної групи досліджувалися професором Л.Калужніним та описані в його працях за допомогою вінцевих добутків циклічних груп. Він запропонував зображати елементи таких груп у вигляді спеціальних таблиць, а саме – впорядкованих наборів многочленів певного вигляду. Ю.Дмитрук у своїй праці описав алгебраїчні структури силовських 2-підгруп симетричних груп. Основою для створення означення та отриманих головних результатів послугували саме статті Л.Калужніна.

Елементи силовських підгруп симетричних груп можна розглядати як портери автоморфізмів кореневих дерев, які наведені в роботі В.Некрашевича. Мінімальна система твірних та графи Келлі на силовських p-підгрупах симетричних груп S_{p^n} описані А.Слупік та В.Сущанським.

Нехай маємо ціле та додатнє число $n > 1$. Силовську 2-підгрупу симетричної групи S_{2^n} позначимо $Syl_2(S_{2^n})$. Група $Syl_2(S_{2^n})$ ізоморфна вінцевому добутуку n-циклічних груп, що мають порядок 2, а саме:

$$Syl_2(S_{2^n}) \cong \underbrace{\mathbb{Z}_2 \wr \ldots \wr \mathbb{Z}_2}_{n \text{ разів}}.$$

Б.Павлик використав поліноміальне представлення елементів групи $Syl_2(S_{2^n})$.

Він описав дії над $Syl_2(S_{2^n})$ на множині мінімальних систем твірних цієї групи.

Нижній центральний ряд силовських підгруп симетричних груп S_n було досліджено А.Віером.

Оскільки такі структури досить зручні та ефективні в обчисленнях, природним продовженням є дослідження алгоритмічних властивостей та створення алгоритмів для обрахунків у силовських 2-підгрупах симетричних груп, елементи яких представлені саме деревами. Враховуючи характеристику порядку групи, особливої уваги заслуговує випадок, коли $p = 2$.

У дисертаційній роботі дослідження ґрунтується на властивостях бінарних кореневих n-рівневих дерев з мітками. У тому числі розглянуті алгоритми основних групових операцій для таких структур, алгоритми зв'язку між деревами та підстановками з силовських 2-підгруп симетричних груп. Окрім того, для кожного алгоритму доведено коректність та досліджено часову складність.

Використовуючи зв'язок підстановок з бінарними кореневими деревами з мітками, в дисертаційній роботі побудовано алгоритм пошуку кількості рухомих точок підстановок та відстані Хеммінга між елементами групи $Syl_2(S_{2^n})$. Логічним продовженням роботи стало дослідження кодів над підстановками із силовських 2-підгруп симетричних груп.

Починаючи із 1970-х років коди, побудовані на підстановках, та їх властивості широко досліджуються у різних сферах. Так І.Блейк у своїй роботі коротко розглянув та узагальнив відомі того часу результати про перестановочні масиви. Метою статті Дж.Денеса було показати, як використовувати перестановочне кодування для голосових операцій та паралельних обчислень. П.Камероном було отримано низку результатів про групи перестановок і перестановочні коди та зроблено огляд про відомі результати. Описано декілька відкритих проблем.

Під кодом на групі підстановок розуміють множину елементів із групи S_n, де довільна пара із множини має відстань не меншу від заданої. При цьому можуть ви-

користовувати як різні підгрупи симетричної групи, так і різні метрики, наприклад, Хеммінга, Улама, Левенштейна тощо.

Перестановочні коди використовуються як коди корекції помилок в каналах комунікацій з малою пропускною здатністю.

Одним з напрямків є дослідження кодів на алгебраїчних структурах. Р.Бейлі у своїй роботі наводить ефективні алгоритми декодування, коли перестановочні коди є підмножинами з S_n.

У дисертації розглядаються властивості кодів, які визначені на підстановках із силовської 2-підгрупи $Syl_2(S_{2^n})$ симетричної групи S_{2^n} з відстанню Хеммінга d_H над ними. Для їх дослідження також використано зв'язок групи $Syl_2(S_{2^n})$ із групою бінарних кореневих n-рівневих дерев з мітками. Досліджено метричні властивості кодів на підстановках. Також описано властивості відстані Хеммінга на підстановках із силовської 2-підгрупи $Syl_2(S_{2^n})$ симетричної групи S_{2^n} та побудовано алгоритм пошуку відстані Хеммінга для підстановок групи, що має складність $O(2^n)$. Окрім того, досліджено метричні властивості кодів на підстановках із $Syl_2(S_{2^n})$ та знайдено розміри і кількість кодів для максимальної та мінімальної ненульової відстані Хеммінга.

Таким чином, у дисертаційній роботі містяться наступні нові наукові результати:

1. Запропоновано алгоритми перетворення підстановки із силовської 2-підгрупи симетричної групи у кореневе бінарне n-рівневе дерево з мітками та обернений перехід від дерева з $LT_{2,n}$ до підстановки із $Syl_2(S_{2^n})$. Доведено їхню коректність і обчислено часову складність.

2. Запропоновано алгоритм множення підстановок силовської 2-підгрупи симетричної групи, що використовує зображення груп бінарними деревами з мітками, та обчислено його часову складність. Описано алгоритм знаходження оберненого елемента. Доведено його коректність та обчислено часову складність.

3. Описано алгоритм пошуку кількості рухомих точок підстановок силовських 2-підгруп симетричних груп з часовою складністю $O(2^n)$. Пораховано, що в середньому алгоримт виконує n кроків для групи $Syl_2(S_{2^n})$.

4. Запропоновано формулу, за якою можна визначити кількість підстановок із $Syl_2(S_{2^n})$, що мають задану, зокрема максимальну, кількість рухомих точок.

5. Наведено алгоритм пошуку відстані Хеммінга для підстановок із силовських 2-підгруп симетричних груп з часовою складністю $O(2^n)$. Оцінено, що в середньому алгоритм виконує n кроків для групи $Syl_2(S_{2^n})$.

6. Запропоновано рекурсивну формулу для обчислення кількості підстановочних кодів C_H над $Syl_2(S_{2^n})$ з максимальною відстанню Хеммінга та обчислено кількість таких кодів.

Практичне значення отриманих результатів. Результати дисертації мають теоретичний характер. Вони можуть використовуватися в теорії кодування, теорії груп підстановок.

Дисертація може бути використана для читання спеціалізованих курсів з теорії алгоритмів та теорії кодування для студентів математичних спеціальностей.

Ключові слова: група підстановок, силовські 2-підгрупи симетричних груп, алгоритми, часова складність, дерево, відстань Хеммінга, коди на групах підстановок.

Abstract

Olshevska V. A. Algorithms and Codes over Sylow 2-Subgroups of Symmetric Groups S_{2^n}. – Qualifying scientific work on the rights of the manuscript.

The thesis for obtaining Ph.D. degree in the field 11 “Mathematics and statistics” and the specialty 113 "Applied mathematics"— National University of Kyiv-Mohyla Academy, Kyiv, 2023.

The dissertation is dedicated to the construction and analysis of algorithms over Sylow 2-subgroups of symmetric groups and the investigation of properties of codes on these groups.

Sylow p-subgroups are classical objects in group theory. Describing such subgroups helps to understand the structure of the group itself. In particular, Sylow p-subgroups of the symmetric group have been studied by Professor L.Kaluzhnin and described in his works using wreath products of cyclic groups. He proposed representing the elements of such groups in the form of special tables, namely, ordered sets of polynomials of a specific form. In his work, Y.Dmitruk described the algebraic structures of the Sylow 2-subgroups of symmetric groups. The basis for creating the definition and obtaining the main results were indeed the articles of L.Kaluzhnin.

The elements of the Sylow subgroups of symmetric groups can be represented as portraits of automorphisms of rooted trees, which V. Nekrashevych introduced in his work. The minimal generating system and Cayley graphs on Sylow p-subgroups of symmetric groups S_{p^n} are described by A.Slupik and V.Sushchansky.

Let $n > 1$ be an integer positive number. We will denote the Sylow 2-subgroup of the symmetric group S_{2^n} as $Syl_2(S_{2^n})$. The group $Syl_2(S_{2^n})$ is isomorphic to the n times iterated wreath product of cyclic groups of order 2, i.e.

$$Syl_2(S_{2^n}) \cong \underbrace{\mathbb{Z}_2 \wr \ldots \wr \mathbb{Z}_2}_{n \text{ разів}}.$$

B.Pawlik used the polynomial representation of elements in the group $Syl_2(S_{2^n})$. He described the action of $Syl_2(S_{2^n})$ on a set of minimal generated sets of this group.

The lower central series of Sylow subgroups of symmetric groups S_n was studied by A.Weir.

Since tree-like data structures are convenient and efficient for calculations, this leads to the natural direction of developing algorithms for computations with Sylow 2- subgroups of symmetric groups using tree-representation. The basic case $p = 2$ deserves special attention due to the binary nature of the data involved.

The dissertation work is based on the properties of binary rooted labeled n-level trees. This includes the algorithms for basic group operations on such structures, algorithms for the connection between trees and permutations from Sylow 2-subgroups of symmetric groups. In addition, the correctness of each algorithm has been proven, and their time complexity has been investigated.

Using the connection between permutations and binary rooted labeled trees, the dissertation work presents an algorithm for finding the number of unfixed points of permutations and the Hamming distance between elements of the group $Syl_2(S_{2^n})$. A natural extension of this work is a studying of permutation codes over Sylow 2-subgroups of symmetric groups.

The permutation codes and their properties have been widely researched in various domains since the 1970s. I.Blake briefly reviewed and generalized known results about permutation arrays at that time. The goal of J.Denes's article was to demonstrate the application of permutation coding in voice operations and parallel computing. P.Cameron in his article provided an overview of results related to sets and groups of permutations and their connection with coding theory. Several open problems were also described in

the article.

A permutation code can be defined as follows: the set of elements from the group S_n with the minimum distance between every pair of them. Different subgroups of the symmetric group can be used, as well as different metrics. In general, there are studied codes with Hamming, Ulam, Levensteins, etc. distances.

Permutation codes are used as error-correction codes in communication channels with limited bandwidth.

The studying of codes on algebraic structures is one of the possible directions. R.Bailey presents efficient decoding algorithms when permutation codes are subsets from S_n in his work.

In the dissertation the properties of permutation codes defined on the Sylow 2-subgroup $Syl_2(S_{2^n})$ of the symmetric group S_{2^n} with Hamming distance d_H over them are considered. For this approach representation of permutations from $Syl_2(S_{2^n})$ by rooted labeled binary trees is used. Metric properties of permutation codes are studied.

The properties of the Hamming distance defined on permutations from the Sylow 2-subgroup $Syl_2(S_{2^n})$ of the symmetric group S_{2^n} are also described. An algorithm for finding the Hamming distance over elements from $Syl_2(S_{2^n})$ with complexity $O(2^n)$ is constructed. Additionally, metric properties of the codes that are defined on permutations from Sylow 2-subgroups $Syl_2(S_{2^n})$ of symmetric group S_{2^n} are studied. The capacity and number of codes for the maximum and the minimum nontrivial Hamming distance over codes are characterized.

Thus, the following new scientific results are included in the thesis.

1. The algorithm of transformation a permutation from the Sylow 2-subgroup of the symmetric group into a labeled rooted n-level binary tree and the reverse algorithm of transformation a tree from $LT_{2,n}$ into a permutation from $Syl_2(S_{2^n})$ are proposed. Their correctness are proved and time complexities are calculated.

2. The algorithm of multiplication of permutations from the Sylow 2-subgroup of the symmetric group is proposed. The time complexity is calculated. The representation of groups by labeled binary trees is using for this. The algorithm of finding the inverse element is described. Its correctness is proved and the time complexity is calculated.

3. The algorithm of counting the number of unfixed points of permutations from the Sylow 2-subgroups of symmetric groups is described with a time complexity $O(2^n)$. It has also been calculated that this algorithm performs n steps for the group $Syl_2(S_{2^n})$ in average.

4. A formula has been proposed by which we can determine the number of permutations in $Syl_2(S_{2^n})$ that have a specified, in particular maximum, number of unfixed points.

5. The algorithm for finding the Hamming distance over permutations from the Sylow 2-subgroups of symmetric groups with a time complexity $O(2^n)$ is provided. It has been proved that this algorithm performs n steps for the group $Syl_2(S_{2^n})$ in average.

6. A recursive formula has been proposed for calculating the number of permutation codes C_H over $Syl_2(S_{2^n})$ with the maximum Hamming distance. The number of such codes has been computed.

The practical significance of the results. The results of this dissertation are mainly theoretical. They can be applied in coding theory and the theory of permutation groups.

The dissertation can be used for lecturing special courses of algorithm theory and coding theory for students majoring in mathematics.

Keywords: permutation group, Sylow 2-subgroups of symmetric groups, algorithms, time complexity, tree, Hamming distance, permutation codes.

Список публікацій за темою дисертації

Публікації, які відображають основні наукові результати дисертації

1. В. А. Ольшевська. Алгоритм обчислень у силовських 2-підгрупах знакозмінних груп за допомогою системи комп'ютерної алгебри GAP // Могилянський математичний журнал, том 1, ISSN 2617-7080, С. 30-33, 2018.

2. Olshevska V. A. Algorithms for computations with Sylow 2-subgroups of symetric groups // Silesian Journal of Pure and Applied Mathematics, Vol. 10, e-ISSN 2719-7913, pp. 103-120, 2020.

3. В.А.Ольшевська. Алгоритм пошуку кількості рухомих точок підстановок із силовських 2-підгруп $Syl_2(s_{2^n})$ симетричних груп S_{2^n} //Могилянський математичний журнал, том 4, С. 34-40, 2022.

4. V.A.Olshevska. Permutation codes over Sylow 2-subgroups $Syl_2(s_{2^n})$ of symmetric groups S_{2^n} // Researches in Mathematics, Vol. 29, No. 2, ISSN 2664-4991, e-ISSN 2664-5009, pp. 28-43, 2021.

Публікації, які засвідчують апробацію матеріалів дисертації

5. В. А. Ольшевська. Застосування GAP для обчислень в силовських 2-підгрупах знакозмінних груп. Збірник тез XV Всеукраїнська науково-практична конференція студентів, аспірантів та молодих вчених «Теоретичні і прикладні проблеми фізики, математики та інформатики», т. 2, сс. 55-56, Київ, 25 -27 травня, 2017.

6. Olshevska V. A. Algorithms for computations in Sylow 2-subgroups of symmetric and alternating groups. Book of abstracts of 11th International Algebraic Conference in Ukraine dedicated to the 75th anniversary of V.V. Kirichenko, p. 94, Kyiv, July 3-7, 2017.

7. В. А. Ольшевська. Зображення підстановок кореневими деревами. Збірник наукових праць Міжнародна наукова конференція «Сучасні проблеми механіки та математики», том 3, с. 222, Львів, 22-25 травня, 2018р.

8. В.А.Ольшевська. Складність алгоритму зображення силовськаих 2-підгруп симетричних груп кореневими деревами. Збірник тез X Всеукраїнська наукова конференція молодих математиків, с. 91, Київ, 16-17 квітня, 2021.

9. Olshevska V. A. Fast multiplication algorithm for Sylow 2-subgroups of symmetric groups. Book of Abstracts of 13th International Algebraic Conference in Ukraine, p. 59, Kyiv, July 6-9, 2021.

10. Olshevska V. A. Permutation codes over Sylow 2-subgroups $Syl_2(S_{2^n})$ of symmetric groups S_{2^n} with Hamming distance. Book of Abstracts of International Algebraic Conference "At the End of the Year"2022, p. 38, Kyiv, December 27-28, 2022.

Зміст

Перелік умовних позначень

$!$ — факторіал

$(A, <)$ — множина A впорядкована за $<$

$A_H(2^n, d)$ максимальний допустимий розмір коду із множини $Syl_2(S_{2^n})$ довжини 2^n та відстанню Хеммінга не меншою за d

$A_O(n, d)$ — потужність підстановочного коду довжини n з найменшою відстанню d

$C_H(2^n, d)$ код, визначений на підстановках із $Syl_2(S_{2^n})$ та з відстанню Хеммінга d

$Coord$ — множина координат вершин дерева

D^{-1} — обернене дерево до дерева D

E — множина ребер графа

$L(v)$ множина всіх висячих вершин, що знаходяться під вершиною v

$LT_{2,n}$ — множина n- рівневих кореневих дерев з мітками

$OC(D)$ — множина координат вершин, що мають помітку 1 у дереві D

$OV(D)$ — множина вершин дерева D, координати яких належать до множини $OC(D)$

$Path_{v_0}(v)$ — шлях (множина вершин), що з'єднують корінь v_0 з певною вершиною v

$R(A)$ $:=\{w_t, w_{t-1}, \ldots, w_1\}$ — реверсивна множина до множини $A = \{w_1, w_2, \ldots, w_t\}$

$Syl_2(S_{2^n})$ — силовська 2-підгрупа симетричної групи S_n

T_n — n-рівневе кореневе дерево

$UV_D(v)$ впорядкована за $<$ множина вершин з мітками 1 дерева D, що знаходяться на шляху від кореня до v

V — множина вершин графа

Γ — граф

$\bigcup$ — об'єднання множин

$\mathbb{ACT}_w(v)$ переміщення вершини v відносно вершини w дерева $T_{2,n}$

$\mathbb{N}$ — множина натуральних чисел

π — підстановка

$\prod\limits_{1\leq i<\leq n}$ — операція добутку

ψ перетворення дерева з $LT_{2,n}$ у підстановку з $Syl_2(S_{2^n})$

$\setminus$ — різниця множин

$\sum\limits_{i=0}^{n} a_i$ — сума елементів

τ перетворення підстановки з $Syl_2(S_{2^n})$ у дерево з $LT_{2,n}$

$\triangle$ — симетрична різниця між двома множинами

$\varnothing$ — порожня множина

$c(v)$ — відображення, що визначає координати вершини v

$d_H(\pi_1, \pi_2)$ відстань Хемінга між підстановками π_1, π_2

$h(\pi)$ кількість рухомих точок підстановки π

i — номер вершини на рівні дерева

id — тривіальне відображення

j — номер рівня дерева

$v \succ w$ — вершина v знаходиться під вершиною w, а саме $w \in Path_{v_0}(v)$

v_0 — корінь дерева

$|A|$ — потужність множини A

Вступ

Актуальність теми

Дисертаційна робота присвячена побудові і аналізу алгоритмів над силовськими 2-підгрупами симетричних груп та дослідженню кодів на цих групах.

Силовські p-підгрупи є класичними об'єктами в теорії груп. Опис таких підгруп допомагає зрозуміти структуру самої групи. Зокрема, силовські p-підгрупи симетричної групи досліджувалися професором Л.Калужніним та описані в його працях за допомогою вінцевих добутків циклічних груп [24]. Він запропонував зображати елементи таких груп у вигляді спеціальних таблиць, а саме – впорядкованих наборів многочленів певного вигляду ([32], [41]). Ю.Дмитрук у своїй праці [15] описав алгебраїчні структури силовських 2-підгруп симетричних груп і продовжив роботу в цьому напрямку разом з Л.Калужніним у [16]. Основою для створення означення та отриманих головних результатів послугувала стаття Л.Калужніна [24].

Елементи силовських підгруп симетричних груп можна розглядати як портрети автоморфізмів кореневих дерев, які наведені в роботі В.Некрашевича [29]. Мінімальна система твірних та графи Келлі на силовських p-підгрупах симетричних груп S_{p^n} описані А.Слупік та В.Сущанським у [34].

Нехай маємо ціле та додатнє число $n > 1$. Силовську 2-підгрупу симетричної групи S_{2^n} позначимо $Syl_2(S_{2^n})$. Група $Syl_2(S_{2^n})$ ізоморфна вінцевому добутку n-циклічних груп, що мають порядок 2, а саме:

$$Syl_2(S_{2^n}) \cong \underbrace{\mathbb{Z}_2 \wr \ldots \wr \mathbb{Z}_2}_{n \text{ разів}}.$$

Б.Павлик використав поліноміальне представлення елементів групи $Syl_2(S_{2^n})$. Він описав дії над $Syl_2(S_{2^n})$ на множині мінімальних систем твірних цієї групи [30].

Нижній центральний ряд силовських підгруп симетричних груп S_n було дослі-

дженно А.Віером [38].

Робота у напрямку дослідження властивостей силовських p-підгруп симетричних груп є актуальною і донині, представлена у низці таких праць: [19], [23], [28], [31].

Оскільки такі структури досить зручні та ефективні в обчисленнях, природним продовженням є дослідження алгоритмічних властивостей та створення алгоритмів для обрахунків у силовських 2-підгрупах симетричних груп, елементи яких представлені саме деревами. Враховуючи характеристику порядку групи, особливої уваги заслуговує випадок, коли $p = 2$.

У десертаційній роботі дослідження ґрунтується на властивостях бінарних кореневих n-рівневих дерев з мітками. У тому числі розглянуті алгоритми основних групових операцій для таких структур, алгоритми зв'язку між деревами та підстановками з силовських 2-підгруп симетричних груп. Окрім того, для кожного алгоритму доведено корекність та досліджено часову складність.

Використовуючи зв'язок підстановок з бінариними кореневими деревами з мітками, в десертаційній роботі побудовано алгоритм пошуку кількості рухомих точок підстановок та відстані Хеммінга між елементами групи $Syl_2(S_{2^n})$. Логічним продовженням роботи стало дослідження кодів над підстановками із силовських 2-підгруп симетричних груп.

Починаючи із 1970-х років коди, побудовані на підстановках, та їх властивості широко досліджуються у різних сферах, наприклад, [5], [7], [8], [12]. Так І.Блейк у своїй роботі [5] коротко розглянув та узагальнив відомі того часу результати про перестановочні масиви. Метою статті Дж.Денеса було показати, як використовувати перестановочне кодування для голосових операцій та паралельних обчислень [12]. П.Камероном було отримано низку результатів про групи перестановок і перестановочні коди та зроблено огляд про відомі результати [7]. Описано кілька відкритих проблем.

Під кодом на групі підстановок розуміють множину елементів із групи S_n, де

довільна пара із множини має відстань не меншу від заданої [33]. При цьому можуть використовувати як різні підгрупи симетричної групи, так і різні метрики, наприклад, Хеммінга, Улама, Левенштейна тощо.

Перестановочні коди використовуються як коди корекції помилок в каналах комунікацій з малою пропускною здатністю [10], [22].

Одним з напрямків є дослідження властивостей кодів на алгебраїчних структурах. Р.Бейлі у [3] наводить ефективні алгоритми декодування, коли перестановочні коди є підмножинами з S_n.

Дослідження перестановочних кодів проводилися активно останнє дисятиліття і залишаються актуальною темою: [26], [9], [2], [4], [6], [35], [11],[21], [27].

У дисертації розглядаються властивості кодів, які визначені на підстановках із силовської 2-підгрупи $Syl_2(S_{2^n})$ симетричної групи S_{2^n} з відстанню Хеммінга d_H над ними. Для їх дослідження також використано зв'язок групи $Syl_2(S_{2^n})$ із групою бінарних кореневих n-рівневих дерев з мітками. Досліджено метричні властивості кодів на підстановках. Також описано властивості відстані Хеммінга на підстановках із силовської 2-підгрупи $Syl_2(S_{2^n})$ симетричної групи S_{2^n} та побудовано алгоритм пошуку відстані Хеммінга для підстановок групи, що має складність $O(2^n)$. Окрім того, досліджено метричні властивості кодів на підстановках із $Syl_2(S_{2^n})$ та знайдено розміри і кількість кодів для максимальної та мінімальної ненульової відстані Хеммінга.

Зв'язок роботи з науковими програмами, планами, темами

Дисертаційні дослідження проводилися на кафедрі математики Національного університету «Києво-Могилянська академія» як частина науково-дослідної теми «Дискретний аналіз і керування випадковими процесами» (номер державної реєстрації 0118U000648).

Мета і задачі дослідження

Метою дослідження є побудова нових алгоритмів над силовськими 2-підгрупами симетричних груп, що дозволяють множити підстановки і знаходити обернений елемент, використовуючи зображення підстановок деревами, та побудова алгоритму знаходження відстані Хеммінга між підстановками групи $Syl_2(S_{2^n})$ і опис властивостей кодів на силовській 2-підгрупі симетричної групи.

Об'єктом дослідження є алгоритмічні властивості силовських 2-підгруп симетричної групи та групових властивостей перестановочних кодів.

Предметом дослідження є алгоритми над силовськими 2-підгрупами симетричних груп та перестановочні коди.

Методи дослідження

Основними методами, що використані у роботі є методи комбінаторного аналізу, теорії груп, теорії алгоритмів, теорії кодування, методи складності обчислень.

Наукова новизна одержаних результатів

Результати дисертації, запропоновані до захисту, є новими і полягають у наступному:

1. Запропоновано алгоритми перетворення підстановки із силовської 2-підгрупи симетричної групи у кореневе бінарне n-рівневе дерево з мітками та обернений перехід від дерева з $LT_{2,n}$ до підстановки із $Syl_2(S_{2^n})$. Доведено їхню коректність і обчислено часову складність.

2. Запропоновано алгоритм множення підстановок силовської 2-підгрупи симетричної групи, що використовує зображення груп бінарними деревами з мітками, та обчислено його часову складність. Описано алгоритм знаходження оберненого елемента. Доведено його коректність та обчислено часову складність.

3. Описано алгоритм пошуку кількості рухомих точок підстановок силовських 2-підгруп симетричних груп з часовою складністю $O(2^n)$. Пораховано, що в середньому алгоримт виконує n кроків для групи $Syl_2(S_{2^n})$.

4. Запропоновано формулу, за якою можна визначити кількість підстановок із $Syl_2(S_{2^n})$, що мають задану, зокрема максимальну, кількість рухомих точок.

5. Наведено алгоритм пошуку відстані Хеммінга для підстановок із силовських 2-підгруп симетричних груп з часовою складністю $O(2^n)$. Оцінено, що в середньому алгоритм виконує n кроків для групи $Syl_2(S_{2^n})$.

6. Запропоновано рекурсивну формулу для обчислення кількості підстановочних кодів C_H над $Syl_2(S_{2^n})$ з максимальною відстанню Хеммінга та обчислено кількість таких кодів.

Практичне значення одержаних результатів

Результати дисертації мають теоретичний характер. Вони можуть бути використані в теорії кодування, теорії груп підстановок.

Дисетрація може використовуватися для читання спеціалізованих курсів з теорії алгоритмів та теорії кодування для студентів математичних спеціальностей.

Особистий внесок автора

Усі результати дисертаційної роботи отримані здобувачем самостійно. Перший розділ дисертації є допоміжним, у ньому викладені відомі результаті з посиланнями на відповідні джерела. всі роботи автора опубліковані самостійно. Результати, що виносяться на захист належать здобувачу.

Апробація результатів дисертації

Основні результати дисертації доповідалися на наукових конференціях, семінарах, міжнародному консорціуму для аспірантів.

Конференції:

1. XV Всеукраїнська науково-практична конфернція студентів, аспірантів та молодих вчених «Теоретичні і прикладні проблеми фізики, математики та інформатики», Київ, 25 -27 травня, 2017.

2. 11th International Algebraic Conference in Ukraine dedicated to the 75th anniversary of V.V. Kirichenko, Kyiv, July 3-7, 2017.

3. Міжнародна наукова конференція «Сучасні проблеми механіки та математики», Львів, 22-25 травня, 2018р.

4. VII щорічна PhD конференція Докторської школи ім. родини Юхименків НаУКМА "A LINEA Київ, 4-8 червня 2018.

5. X Всеукраїнська наукова конференція молодих математиків, Київ, 16-17 квітня, 2021.

6. 13th International Algebraic Conference in Ukraine, Kyiv, July 6-9, 2021.

7. International Algebraic Conference "At the End of the Year" 2022, Kyiv, December 27-28, 2022.

Семінари:

1. семінар аспірантів факультету інформатики НаУКМА (за участю гостьового професора з Вільнюського університету V.Dagiene), 12 квітня 2019.

Міжнародний консорціум для аспірантів:

1. 10th International Doctoral Consortium on Informatics and Informatics Engineering Education Research joint with Nordplus workshop Culturally Diverse Approaches to Learning Mathematics and Computational Thinking, Druskininkai, Lithuania, December 2–6, 2019,

Публікації

Результати дисертації опубліковано у 10 наукових працях: 4 статті та 6 тез. Серед них 1 стаття у журналі, що входить до міжнародної наукометричної бази Scopus [51], 1 стаття — у фаховому видання категорії Б з "Переліку затвердженого Міністерством освіти й науки України [50], 1 стаття – у періодичному науковому виданні держави, що входить до Європейського союзу [47], 2 тези – у матеріалах всеукраїнських наукових конференцій [43], [48], 4 тези – у матеріалах міжнародних наукових конференцій [44], [45], [49], [52].

Структура і обсяг дисертації

Дисертаційна робота складається з анотації, переліку умовних позначень, вступу, п'яти розділів, висновків, списку використаних джерел та додатків. Повний обсяг дисертації складає 138 сторінок друкованого тексту. Обсяг основного тексту — 112 сторінок. Список використаних джерел обсягом 5 сторінок містить 42 найменування. Додатки займають 4 сторінки, містять список опублікованих праць за темою дисертації та відомості про апробацію результатів дисертації.

Автор щиро вдячний своєму науковому керівнику Б. В. Олійник за постановку задач, увагу до роботи, сприяння розвитку, постійну підтримку та цінні поради.

Зміст роботи

Дисертаційна робота складається з п'яти розділів.

У **першому розділі** представленні необхідні означення та твердження, що розділенні на підрозділи відповідно до тем.

У першому підрозділі наведенні властивості та означення, що відносяться до теорії груп, а саме до груп підстановок. У тому числі згадані основні теоерми пов'язані з силовськими p-підгрупами, де p – просте число. Також наведений теоретичний матеріал про комутант та метрику в групах підстановок.

У другому підрозділі вводяться необхідні означення, пов'язані з теорією графів, а точніше – з деревами. Описано властивості дерев, розглянуто метрику на них, яку визначено як довжину найкоротшого шляху між двома вершинами.

Третій підрозділ відноситься до теорії складності алгоритмів. Безпосередньо у ньому наведене неформальне означення алгоритму, а також описані поняття середньої та часової складностей алгоритмів, в тому числі — асимптотичної оцінки O.

У четвертому підрозділі записані означення, пов'язані з кодами на групах підстановок, а саме: підстановочний код та потужність такого коду.

У **другому розділі** вводиться представлення вершин n-рівневого дерева за допомогою їх координат. Завдяки цьому фіксується заданий порядок вершин. Це дає можливість ввести операцію $switch$ для вершин дерева:

$switch(D, j, i)$ = "переставити між собою два піддерева дерева D, для яких вершина v з координатами (j, i) є коренем".

У випадку, коли координати вершини не були важливими, використовується позначення $switch(D, v)$ = "переставити між собою два піддерева дерева D, для яких вершина v є коренем".

Також у другому розділі описуються основні числові характеристики для дерев D із $LT_{2,n}$.

У першому підрозділі наведено алгоритм, що визначає операцію *switch* складності $O(2^{n-j})$ та представлено алгоритм множення двох кореневих дерев із $LT_{2,n}$ зі складністю $O(n \cdot 2^n)$.

У другому підрозділі введено означення відображення, яке представляє образ певної вершини при переміщенні через дію *switch*:

Означеннння 2.2. *Нехай w – деяка фіксована вершина дерева, $w \in V(T_n)$. Визначимо функцію $\mathbb{ACT}_w : V(T_n) \to V(T_n)$ наступним чином:*
$\mathbb{ACT}_w(v) = v'$ тоді і тільки тоді, коли v' є образом вершини v після $switch(T_n, w)$.

На основі цього означення наведено теорему, яка є аналогом алгоритму множення двох дерев:

Теорема 2.3. *Нехай $D_1, D_2 \in LT_{2,n}$ – довільні дерева. Тоді їхнім добутком буде дерево, у якому множина вершин з мітками 1 визначається як симетрична різниця двох множини вершин з мітками 1, де:*

- *перша – це результат відображення $\mathbb{ACT}$ вершин з мітками 1 дерева D_2 відносно впорядкованої за $<$ множини вершин з міткаим 1 дерева D_1,*
- *а друга – це множина з мітками 1 самого дерева D_1.*

Тобто:

$$OV(D_1 * D_2) = (\, \mathbb{ACT}_{(OV(D_1),<)}(OV(D_2))\,) \triangle OV(D_1)$$

Далі значну частину розділу присвячено дослідженню властивостей відображення вершин $\mathbb{ACT}$. Доведено ряд лем.

Лема 2.1. *Композиція відображень $\mathbb{ACT}$ для пари фіксованих вершин, що не знаходяться одна під одною, є комутативною. Тобто, для довільних вершин $v_1, v_2 \in V(T_n)$, таких що $v_1 \not> v_2$ та $v_2 \not> v_1$ маємо:*

$$\mathbb{ACT}_{v_1} \cdot \mathbb{ACT}_{v_2} = \mathbb{ACT}_{v_2} \cdot \mathbb{ACT}_{v_1}$$

Лема 2.2. *Нехай $v, v_1 \in V(T_n)$ та вершина v_1 знаходиться на меншому рівні ніж вершина v. Тоді відображення $\mathbb{ACT}$, що виконується по результату переміщення $\mathbb{ACT}$ вершини v відносно вершини v_1, буде ріносильне композиції відображень $\mathbb{ACT}$ відносно v_1, v, та знову v_1. Тобто для $v_1 < v$ маємо:*

$$\mathbb{ACT}_{\mathbb{ACT}_{v_1}(v)} = \mathbb{ACT}_{v_1} \cdot \mathbb{ACT}_v \cdot \mathbb{ACT}_{v_1} \tag{1}$$

Лема 2.3. *Нехай $A \subset V(T_n)$ – довільна впорядкована множина та $B, C \subset V(T_n)$ – довільні множини. Тоді переміщення результату симетричної різниці множин B та C відносно множини A рівносильне симетричній різниці образу переміщення множин B і C відносно A. Тобто:*

$$\mathbb{ACT}_A(B \triangle C) = \mathbb{ACT}_A(B) \triangle \mathbb{ACT}_A(C)$$

Лема 2.4. *Нехай $a, b \in V(T_n)$ – довільні вершини. Тоді композиція переміщення $\mathbb{ACT}$ відносно цих вершин рівносильна переміщенню відносно впорядкованої за $<$ множини, що є результатом симетричної різниці образу переміщення вершини a відносно b та самої вершини b. Тобто:*

$$\mathbb{ACT}_a \cdot \mathbb{ACT}_b = \mathbb{ACT}_{(\{\mathbb{ACT}_b(a)\} \triangle \{b\}, <)} \tag{2}$$

Лема 2.5. *Нехай $A \subset V(T_n)$ – довільна впорядкована за $<$ множина вершин дерева та $b \in V(T_n)$ – довільна вершина цього дерева. Тоді композиція відображень $\mathbb{ACT}$ відносно A та b рівносильна переміщенню відносно впорядкованої за $<$ множини, що є результатом симетричної різниці образу множини A відносно вершини b для відображення $\mathbb{ACT}$ та самої вершини b. Тобто:*

$$\mathbb{ACT}_A \cdot \mathbb{ACT}_b = \mathbb{ACT}_{(\mathbb{ACT}_b(A) \triangle \{b\}, <)}$$

Лема 2.6. *Нехай $A, B \subset V(T_n)$ – довільні впорядкованих за $<$ множини. Тоді композиція відображень $\mathbb{ACT}$ відносно множин A та B рівносильна переміщенню*

відносно впорядкованої за $<$ *множини, яка є результатом симетричної різниці образу* A *відносно* B *для відображення* $\mathbb{ACT}$ *та самої множини* B. *Тобто:*

$$\mathbb{ACT}_A \cdot \mathbb{ACT}_B = \mathbb{ACT}_{(\mathbb{ACT}_B(A) \triangle B, <)}$$

У третьому підрозділі описується вигляд бінарного дерева, яке є нейтральним елементом відносно введеної операції множення дерев.

Нехай E — це дерево з $LT_{2,n}$, в якому вершини не мають помітки 1: $OV(E) = \emptyset$.

Теорема 2.4. *Дерево* $E \in LT_{2,n}$ *є нейтральним елементом над* $LT_{2,n}$ *відносно операції множення дерев* $*$.

В четвертому підрозділі наведені поняття реверсивної множини та алгоритм взяття оберненого дерева складності $O(n \cdot 2^n)$. Були введенні властивості відображення $\mathbb{ACT}$ над реверсивними множинами. Це дало змогу довести коректність алгоритму взяття оберненого.

У п'ятому підрозділі наведені наступні два алгоритми: Перетворення дерева із $LT_{2,n}$ у підстановку з $Syl_2(S_{2^n})$ складності $O(n \cdot 2^n)$ та Перетворення підстановки з $Syl_2(S_{2^n})$ у дерево з $LT_{2,n}$ складності $O(2^n)$. Для обох алгоритмів доведено корекність.

На основі цих алгоритмів створено два відображення:

- $\psi : LT_{2,n} \to Syl_2(S_{2^n})$ задається алгоритмом 4;
- $\tau : Syl_2(S_{2^n}) \to LT_{2,n}$, задається алгоритмом 5

та отримано наступні результати:

Теорема 2.11. *Відображення* ψ *та* τ *є взаємооберненими, що дає нам бієкцію між* $LT_{2,n}$ *та* $Syl_2(S_{2^n})$.

Теорема 2.12. *Відображення* ψ *є ізоморфізмом між* $LT_{2,n}$ *та* $Syl_2(S_{2^n})$.

Наслідок 2.3. *Відображення τ є ізоморфізмом між $Syl_2(S_{2^n})$ та $LT_{2,n}$.*

З цієї теореми випливає ще один наслідок:

Наслідок 2.4. *Множина кореневих n-рівневих дерев з мітками $LT_{2,n}$ та заданою операцією множення $*$ є групою.*

Третій розділ присвячено дослідженню відстані Хеммінга для підстановок із групи $Syl_2(S_{2^n})$. Для цього також було використано представлення підстановок групи за допомогою бінарних n-рівневих кореневих дерев з мітками.

У першому підрозділі наводиться класичне означення кількості рухомих точок підстановки:

Означеннння 3.1. [17] *Кількість рухомих точок підстановки π – це кількість позицій, де елементи нижньої стрічки підстановки відрізняються від елементів верхньої стрічки.*

Також довденені деякі основні властивості кількості рухомих точок $h(\pi)$. Для цього було використоано представлення підстановок відповідними бінарними кореневими деревами з мітками.

Лема 3.1. *Нехай $\pi \in Syl_2(S_{2,n})$ – така підстановка, що відповідне їй дерево $D \in LT_{2,n}$ має єдину мітку 1 на вершині з координатами (j, i). Тоді кількість її рухомих точок буде рівна кількості висячих вершин піддірева з коренем на координатах (j, i). Тобто:*

$$h(\pi) = 2^{n-j}.$$

Лема 3.2. *Нехай підстановка π має відповідне їй дерево D, в якого мітки 1 знаходяться на вершинах $v_1, v_2, \ldots, v_r$ з координатами $(j_1, i_1), (j_2, i_2) \ldots, (j_r, i_r)$. Причому жодна вершина v_k, $k \in \{1, 2, \ldots, r\}$, не лежить на шляху, що сполучає будь-яку*

іншу вершину цієї множин з коренем. Тоді кількість рухомих точок такої підстановки буде рівна сумі висячих вершин всіх піддерев з коренями $v_1, v_2, \ldots, v_r$. *Тобто:*

$$h(\pi) = 2^{n-j_1} + 2^{n-j_2} + \ldots + 2^{n-j_r} = \sum_{k=1}^{r} 2^{n-j_k}.$$

Наступна лема описує кількість рухомих точок підстановки у випадку, коли у відповідному дереві вершина v_q знаходиться під вершиною v_k, тобто у піддереві, де v_k – корінь.

Лема 3.3. *Нехай підстановка* π *така, що у відповідному їй дереві* D *лише дві вершини* v_k *та* v_q *з координатами* (j_k, i_k) *та* (j_q, i_q) *мають мітки 1, причому* v_q *знаходиться під вершиною* v_k. *Тоді кількість рухомих точок такої підстановки буде рівна кількості висячих вершин піддерева з коренем* v_k. *Тобто для* $v_q \succ v_k$ *маємо:*

$$h(\pi) = 2^{n-j_k}.$$

Також введене означення головної вершини.

Означеннння 3.2. *Нехай вершина* v *дерева має мітку 1. Якщо на шляху між нею та коренем всі решта вершин мають мітки 0, то* v *будемо називати головною вершиною.*

Із лем було отримано наступний наслідок, на якому базується основна ідея для алгоритму пошуку кількості рухомих точок підстановки із $Syl_2(S_{2^n})$:

Наслідок 3.1 . *Кількість рухомих точок підстановки* π, *що задається деревом* D, *рівна сумі кількостей висячих вершин, що знаходяться під головними вершинами дерева.*

І відповідно далі представлено сам алгоритм знаходження кількості рухомих точок підстановки групи з часовою складністю $O(2^n)$, а також доведено його коректність.

Враховуючи будову дерев та можливі розміщення головних вершин на дереві для кожної підстановки, було отримано результат про середню кількість операцій порівняння даного алгоритму:

Теорема 3.1. *Для знаходження кількості рухомих точок підстановки* $\pi \in Syl_2(S_{2^n})$ *за алгоритмом 6 в середньому необхідно виконати* n *порівнянь.*

У другому підрозділі знайдено кількість підстановок, що мають мінімальну ненульову кількість рухомих точок, та кількість підстановок, що мають максимальну кількість рухомих точок. Останнє вдалося визначити за допомогою рекурсивної формули.

Теорема 3.2 . *Кількість підстановок* $\pi \in Syl_2(S_{2^n})$, *які мають* 2^n *рухомих точок дорівнює* $f(n)$, *що визначається рекурсивно наступним чином:*

$$f(n) = \begin{cases} 1, \text{ коли } n = 1 \\ 2^{2^n-2} + f(n-1) \cdot f(n-1) \end{cases}$$

Третій підрозділ даного розділу присвячений власне дослідженню відстані Хеммінга між підстановками із силовської 2-підгрупи $Syl_2(S_{2^n})$ симетричної групи S_{2^n}. Запропоновано теорему про пошук відстані Хеммінга для підстановок, що представлені кореневими деревами з мітками:

Теорема 3.3. *Нехай* π_1, π_2 *– підстановки із групи* $Syl_2(S_{2^n})$ *та* $D_1, D_2 \in LT_{2,n}$ *– відповідні їм дерева. Тоді відстань Хеммінга між цими підстановками рівносильна відстані Хеммінга між нейтральним елементом* e *групи* $Syl_2(S_{2^n})$ *та підстановкою, відповідне дерево якої є результатом симетричної різниці дерев* D_1 *та* D_2. *Тобто:*

$$d_H(\pi_1, \pi_2) = d_H\Big(e, \psi(D_1 \Delta D_2)\Big) \tag{3}$$

А також наведено твердження, що описує зв'язок відстані Хеммінга із кількістю рухомих точок підстановки:

Твердження 3.3. *Нехай $\pi \in Syl_2(S_{2^n})$, e — нейтральний елемент групи $Syl_2(S_{2^n})$. Тоді відстань Хеммінга між підстановкою та нейтральним елементом можна визначити як кількість рухомих точок цієї підстановки. Тобто:*

$$d_H(e, \pi) = h(\pi).$$

Дані результати стали основою для створення алгоритму пошуку відстані Хеммінга d_H для підстановок $\pi_1, \pi_2 \in Syl_2(S_{2^n})$, який наведений далі. Для даного алгоритму доведено коректність, знайдено часову складність, що становить $O(2^n)$ а також пораховано, що в середньому для всіх підстановок групи $Syl_2(S_{2^n})$ алгоритм виконуватиме n кроків.

У четвертому підрозділі наведені деякі властивості відстані Хеммінга для підстановок із $Syl_2(S_{2^n})$, які представлені кореневими деревами з мітками.

Лема 3.5. *Нехай m — це деяке парне число від 2^1 до 2^n. Тоді існують підстановки $\pi, \sigma \in Syl_2(S_{2^n})$ такі, що відстань Хеммінга між ними дорівнює числу m. Тобто:*

$$d_H(\pi, \sigma) = m.$$

Лема 3.8. *Нехай $\pi \in Syl_2(S_{2^n})$, d — деяке парне число від 0 до 2^n. Тоді кількість підстановок, що знаходяться на відстані Хеммінга d від підстановки π, дорівнює кількості дерев Q, відповідні підстановки яких рухають рівно d точок:*

$$|\{\sigma \in Syl_2(S_{2^n}) | d_H(\pi, \sigma) = d\}| = |\{Q \in LT_{2,n} | h(\psi(Q)) = d\}|.$$

У п'ятому підрозділі запропоновоно формулу, за якою можна визначити кількість підстановок із $Syl_2(S_{2^n})$, що мають задану кількість m рухомих точок:

Теорема 3.6. *Кількість підстановок із $Syl_2(S_{2^n})$, що мають m рухомих точок, визначається за формулою:*

$$u(n, m) = \sum_{b \in \mathbb{B}(m)} \prod_{k=1}^{n} \binom{2^{n-k} - g(n, k, b)}{b[k]} \cdot f^{b[k]}(k), \tag{4}$$

де:

1. $g(n,k,b) = \sum\limits_{k'=k+1}^{n} 2^{k'-k} \cdot b[k']$ — *кількість вершин на рівні* $j = n-k$, *які не можна зробити головними;*

2. $f(k) = 2^{2^k-1}$ – *кількість різних піддерев (а тому і їхніх відповідних підстановок), що можна отримати, для яких вершина на рівні* $j = n-k$ *є коренем.*

Четвертий розділ присвячено кодам на підстановках силовської 2-підгрупи симетричної групи, що задаються за допомогою відстані Хеммінга.

Наведено наступні означення.

- $C_H(2^n, d)$ – код, визначений на підстановках із $Syl_2(S_{2^n})$ та з відстанню Хеммінга d такий, що для будь-яких $\pi, \sigma \in Syl_2(S_{2^n})$ маємо:

$$\pi, \sigma \in C_H(2^n, d) \text{ тоді і тільки тоді, коли } d_H(\pi, \sigma) \geq d.$$

- $A_H(2^n, d)$ – максимальний допустимий розмір коду із 2-роздільних підстановок із множини $Syl_2(S_{2^n})$ довжини 2^n та відстанню Хеммінга не меншою за d.

Для зручності, запропоновано зображати довільний код $C_H(2^n, d)$ матрицею, рядки якої відповідають нижнім стрічкам підстановок із $C_H(2^n, d)$. Однією з властивостей коду є наступна:

Твердження 4.1. *Максимальний допустимий розмір коду підстановок із групи* $Syl_2(S_{2^n})$ *з відстанню Хеммінга не меншою ніж* 2^n *дорівнює* 2^n, *тобто*

$$A_H(2^n, 2^n) = 2^n.$$

У першому підрозділі знайдено кількість підстановочних кодів виду $C_H(2^n, 2^n)$ над елементами із $Syl_2(S_{2^n})$ та відстанню Хеммінга 2^n, тобто максимальною. Даний результат визначається рекурсивним способом.

Теорема 4.1. *Кількість підстановочних кодів $C_H(2^n, 2^n)$ з максимально відстанню Хеммінга $d_H = 2^n$ групи $Syl_2(S_{2^n})$ визначається рекурсивно та рівна:*

$$f(n) = \begin{cases} 4, \text{ якщо } n = 2; \\ f^4(n-1) \cdot (2^{n-1}!)^2, \text{ якщо } n > 2. \end{cases}$$

У другому підрозділі описаний інший крайовий випадок, тобто пораховано максимальний допустимий розмір коду з найменшою ненульовою відстанню Хеммінга:

Твердження 4.2. *Максимальний допустимий розмір коду підстановок із групи $Syl_2(S_{2^n})$ з відстанню Хеммінга не меншою ніж 2 становить 2^{2^n-1}, тобто:*

$$A_H(2^n, 2) = 2^{2^n-1}.$$

П'ятий розділ присвячено застосуванню інструментів комп'ютерної алгебри GAP задля отримання низки результатів для симетричної групи і її силовських 2-підгруп.

У першому підрозділі розглянуто групові операції, які можна реалізувати за допомогою GAP. На прикладі групи S_8 показано як на мові GAP:

1. породити групу S_8 множиною заданих підстановок;
2. знайти комутант для заданої групи S_8;
3. знайти порядок групи.

У другому підрозділі наведено алгоритм для перевірки мінімальності системи твірних силовської 2-підгрупи знакозмінної групи за допомогою фактор групи. Наведена реалізація алгоритму за допомогою GAP.

У третьому підрозділі наведені наступні результати для силовської 2-підгрупи знакозмінної групи A_8 та елементів $\alpha_0 = (1,5)(2,6)(3,7)(4,8)$, $\alpha_1 = (1,3)(2,4)$, $\alpha_2 = (1,2)(5,6)$, отримані із описаного алгоритму:

1. Набір елементів α_0, α_1 і α_2 є системою твірних $Syl_2(A_8)$.

2. Комутант групи $Syl_2(A_8)$ породжується підстановками $(1,3)(2,4)(5,7)(6,8)$, $(1,2)(3,4)$, $(1,3)(2,4)(5,8)(6,7)$ і є елементарною абелевою 2-групою порядку 8, тобто ізоморфний ${C_2}^3$;

3. Фактор по комутанту групи $Syl_2(A_8)$ також є елементарною абелевою 2-групою порядку 8.

4. Система твірних $\{\alpha_0, \alpha_1, \alpha_2\}$ є мінімальною.

Також описано властивості комутанта:

Теорема 5.1. *Кожен елемент комутанта групи $Syl_2(A_8)$ є комутатором.*

Теорема 5.2. *Множина комутаторів твірних $\alpha_0, \alpha_1, \alpha_2$ породжує в комутанті групи $Syl_2(A_8)$ власну підгрупу порядку 4.*

У четвертому підрозділі описані схожі результати для випадку силовської 2-підгрупи симетричної групи A_{16} та елементів $\alpha_0 = (1,9)(2,10)(3,11)(4,12)(5,13)(6,14)(7,$ $\alpha_1 = (1,5)(2,6)(3,7)(4,8)$, $\alpha_2 = (1,3)(2,4)$, $\alpha_3 = (1,2)(9,10)$.

1. Набір елементів $\alpha_0, \alpha_1, \alpha_2$ та α_3 є системою твірних $Syl_2(A_{16})$.

2. Комутант $Syl_2(A_{16})$ – не абелева група порядку 2^{10}, що породжується 5-ма твірними:

   $(1,4,2,3)(5,6)(9,12)(10,11)$, $(1,4)(2,3)(5,8)(6,7)$, $(1,2)(5,6)$,
   $(1,7,3,5)(2,8,4,6)(9,14,12,16)(10,13,11,15)$,
   $(1,7)(2,8)(3,6)(4,5)(9,16,10,15)(11,14,12,13)$.

3. Фактор-група $Syl_2(A_{16})$ по її комутанту – абелева група порядку 16, що породжується 4 елементами.

4. Система твірних $\{\alpha_0, \alpha_1, \alpha_2, \alpha_3\}$ групи $Syl_2(A_{16})$ є мінімальною.

1 Необхідні визначення та допоміжні твердження

1.1 Групи підстановок та метрики на них

1.1.1 Симетричні та знакозмінні групи, системи твірних

Нагадаємо означення бінарної операції.

Означення 1.1. *Бінарною дією* (або просто дією) $*$ на непорожній множині M називається довільне відображення $* : M \times M \longrightarrow M$. Результат застосування дії $*$ до пари (a, b) позначається $a * b$. Синонімом до терміну "дія"є слово "операція".

Означення 1.2. Нехай G – непорожня множина, $*$ – бінарна дія на G. Пару $(G, *)$ будемо називати *групою*, якщо виконуються наступні умови (аксіоми групи):

1. *асоціативність:* для довільних $a, b, c \in G : (a * b) * c = a * (b * c)$;
2. *існування нейтрального елемента:* існує такий елемент $e \in G$, що для довільного $a \in G : a * e = e * a = a$;
3. *оборотність:* для кожного $a \in G$ існує такий елемент $a^{-1} \in G$, що $a * a^{-1} = a^{-1} * a = e$.

Якщо дія комутативна, тобто для всіх $a, b \in G : a * b = b * a$, то група називається *комутативною* або *абелевою.*

Означення 1.3. Непорожня підмножина H множини G називається *підгрупою* групи $(G, *)$ якщо $(H, *)$ теж є групою.

Надалі для груп будемо опускати позначення групової дії $*$.

Означення 1.4. Нехай X, Y є множинами однакової потужності. Відображення $f : X \to Y$ називається *бієкцією між множинами X та Y*, якщо виконуються наступні умови:

1. для різних $x_1, x_2 \in X$ $f(x_1) \neq f(x_2)$

2. для кожного $y \in Y$ існує $x \in X$ такий, що $f(x) = y$

Означення 1.5. Нехай G та H є групами. Відображення $f : G \to H$ називається *гомоморфізмом*, якщо для кожного $g_1, g_2 \in G$ виконується:

$$f(g_1 * g_2) = f(g_1) * f(g_2)$$

Означення 1.6. Нехай G та H є групами. Відображення $f : G \to H$ називається *ізоморфізмом груп G та H*, якщо відображення f є бієкцією цих множин і гомеоморфізмом цих груп.

Нехай N – впорядкована множина n елементів. Нагадаємо, *підстановкою n-го степеня* називається взаємооднозначне відображення множини N. Пара $(i, j) \in N^2$ є *інверсією підстановки* π n-го степеня, якщо:

$$i < j \text{ та } \pi(i) > \pi(j).$$

Підстановка називається *парною*, якщо кількість її інверсій – парне число, тобто

$$\left|\{(i, j) \in \{1, 2, \ldots, n\}^2 | i < j, \pi(i) > \pi(j)\}\right| \not\vdots 2.$$

Симетричною групою S_n називається множина всіх підстановок n-го степеня відносно операції множення підстановок (суперпозиції відображень). Підмножина усіх парних підстановок утворює власну підгрупу S_n, яка називається *знакозмінною групою* A_n.

Іншим способом описати означення підстановки можна використовуючи підхід Д.Робінсона у [32].

Нехай X – не порожня множина. Бієкція $\pi : X \longrightarrow X$ називається *підстановкою* на X.

Множина всіх перестановок над X є групою відносно композиції і називається *симетричною групою* над X. Коли $X = \{1, 2, \dots n\}$, то використовують позначення S_n та саму групу називають *симетричною групою степеня* n. Така група має порядок $|S_n| = n!$.

Знаком підстановки $\pi \in S_n$ називається вираз

$$sign\,\pi = \prod_{1 \le i < j \le n} \frac{\pi(i) - \pi(j)}{i - j},$$

який дорівнює +1 або -1.

Підстановка π є *парною*, якщо $sign\,\pi = +1$ та *непарною*, якщо $sign\,\pi = -1$.

Множина всіх парних підстановок степеня n є підгрупою в S_n та називається *знакозмінною групою* A_n і має порядок $|A_n| = \frac{n!}{2}$.

Означення 1.7. [32] Множина S буде *системою твірних групи* G тоді й лише тоді, коли кожен елемент $g \in G$ має розклад над S, тобто

$$g = a_1 a_2 ... a_k,$$

де кожен множник a_i належить множині S або є оберненим до елемента з S.

Означення 1.8. Система твірних S групи G називається *незвідною*, якщо для кожного $a \in S$ множина $S \backslash \{a\}$ не є системою твірних G.

Означення 1.9. Система твірних S групи G називається *мінімальною*, якщо для довільної множини A, що є системою твірних G, виконується:

$$|A| \ \ge |S|.$$

1.1.2 Силовські підгрупи

Нехай p – просте число. Під *p-групою* ми маємо на увазі скінченну групу, порядок якої є степенем числа p (тобто дорівнює p^n для деякого цілого $n \ge 0$) [25].

Означення 1.10. Нехай G — скінченна група, p — просте число, яке є дільником порядку G. Підгрупи групи G порядку p^t називаються *p-підгрупами.*

Означення 1.11. Нехай $|G| = p^k s$, де s — деяке натуральне число, що не ділиться на p, p – просте. *Силовською p-підгрупою* групи G називається така p-підгрупа групи G, що має порядок p^k і позначається $Syl_p(G)$.

Теорема 1.1 (Силова про існування)**.** *Нехай G – скінченна група, $|G| = p^k s$, де p – просте і $p \nmid s$. Тоді для довільного r, $0 \leq r \leq k$ в групі G існує підгрупа подрядку p^r.*

1.1.3 Комутант симетричних та альтернативних груп

Відповідно до [39] нагадаємо основні означення.

Означення 1.12. *Комутатором елементів a* і b групи G називається елемент

$$[a, b] = a^{-1}b^{-1}ab.$$

Очевидно, що комутатор двох елементів буде дорівнювати нейтральному e тоді і лише тоді, коли ці елементи є переставними. Звичайною перевіркою можна довести наступні властивості комутатора [39].

Твердження 1.1. *Для довільних елементів a, b, c групи G виконуються рівності:*

- $[a, b]^{-1} = [b, a]$;
- $ab = ba \cdot [a, b]$;
- $[a, b]^x = [a^x, b^x]$.

Означення 1.13. *Комутантом групи G* називають підгрупу G', яка породжена всіма комутаторами, тобто

$$G' = \langle [a, b \mid a, b \in G \rangle.$$

Означення 1.14. Підгрупа H групи G називається *нормальною підгрупою групи* G, якщо для кожного $g \in G$ виконується:

$$g^{-1} \cdot H \cdot g = H$$

Означення 1.15. *Факторгрупою групи G по нормальній підгрупі H* є множина класів суміжності підгрупи H з множенням, що визначається наступним чином:

$$(aH) * (bH) = (ab)H.$$

Теорема 1.2. *Нехай G – група, G' – її комутант. Якщо фактор-групу G/G' не можна породити менше, ніж k елементами, то і саму групу G не можна породити менше, ніж k елементами.*

1.1.4 Метрики на групах підстановок

Нагадаємо класичне означення метрики, наведене у [13].

Нехай X – деяка множина. Функцію $d : X \times X \longrightarrow \mathbb{R}$ називають *метрикою на* X, якщо для довільних $x, y, z \in X$ виконується наступне:

1. $d(x, y) \geq 0$ (не від'ємність),
2. $d(x, y) = 0$ тоді і тільки тоді, коли $x = y$ (тотожність),
3. $d(x, y) = d(y, x)$ (симетричність),
4. $d(x, y) \leq d(x, z) + d(z, y)$ (нерівність трикутника).

Одним із прикладів метрики є *відстнь Хеммінга*. Вона позначається d_H та є метрикою в n-вимірному просторі $\mathbb{R}^n$. Обчислюється ця відстань як кількість координат, у яких вектори x та y мають різні значення, тобто:

$$\big|\{i : 1 \leq i \leq n, x_i \neq y_i\}\big|.$$

Таку метрику можна визначити і на групі підстановок S_n.

Означення 1.16. *Відстанню Хеммінга між двома підстановками* $\pi_1, \pi_2 \in S_n$ є кількість елементів, образи яких різні:

$$d_H(\pi_1, \pi_2) = \left| \{ x \in \{1, \ldots, k\} \mid \pi_1(k) \neq \pi_2(k) \} \right|. \tag{5}$$

А також нагадаємо, що *кількість рухомих точок* підстановки $\pi \in Syl_2(S_{2^n})$ – це кількість позицій k, де образ елемента підстановки відрізняється від його прообразу $\pi(k) \neq k$ [17]. Позначимо: $h(\pi)$.

1.2 Дерева та їх властивості

Нехай $n \in \mathbb{N}$, $V = \{v_1, v_2, \ldots, v_n\}$ – деяка множина, E – деяка підмножина пар із V^2.

Означення 1.17. *Граф* $\Gamma(V, E)$ – геометрична конфігурація, яка складається з множини вершин V і множини ребер E. Вершини v_i та v_j, $i < j$, графа називаються *суміжними* (або *сусідніми*), якщо ребро $(v_i, v_j) \in E$.

Зазначимо, що в графі може існувати кілька ребер між одними й тими ж двома вершинами $x, y \in V$. Такі ребра називаються *кратними ребрами* [14]. Якщо ж ребро з'єднує вершину саму з собою, то таке ребро називається *петлею*.

Надалі, розглядаємо *простий граф* – граф без кратних ребер та петель.

Означення 1.18. Послідовність вершин і ребер $v_{i_1}, (v_{i_1}, v_{i_2}), v_{i_2}, (v_{i_2}, v_{i_3}), \ldots, v_{i_n}$, у якій сусідні ребра суміжні одній і тій же вершині, називають *шляхом*. Кількість ребер в шляху називається *довжиною шляху*. Якщо $v_{i_1} = v_{i_n}$, то такий шлях називається *циклом*.

Означення 1.19. *Відстаню між вершинами* v_i *та* v_j називається довжина найкоротшого шляху між v_i та v_j.

Означення 1.20. Граф $\Gamma(V, E)$ називають *зв'язним*, якщо будь-які дві вершини $v_i, v_j \in V$ можна сполучити шляхом.

Означення 1.21. *Степенем вершини* $v \in V$ називається кількість суміжних вершин з v. Вершина степеня 1 називається *кінцевою* або *висячою*.

Означення 1.22. [40] Зв'язний граф без циклів називається *деревом*.

Означення 1.23. Зв'язний підграф дерева називається *піддеревом* заданого дерева.

Означення 1.24. Нехай $v_0 \in V$ - фіксована вершина дерева $\Gamma(V, E)$, яку будемо називати *коренем*. *Рівень дерева* Γ — множина вершин, які знаходяться на однаковій відстані під початкової вершини, кореня.

Означення 1.25. Вершина $v \in V$ *належить* рівню j дерева, якщо відстань між v та коренем $v_0 \in V$ дорівнює j.

Означення 1.26. n-рівнене кореневе дерево будемо називати *бінарним* і позначати T_n, якщо:

- корінь має степінь 2;
- степінь кожної вершини з 1-го по $(n-1)$-й рівень дорівнює 3;
- вершини n-го рівня є висячими.

Означення 1.27. *Вершиннопоміченим деревом* будемо називати таке бінарне n-рівневе кореневе дерево T_n, в якого на всіх вершинах із 0-го по $(n-1)$-ий рівні стоять мітки 0 або 1.

Позначимо множину таких n-рівневих дерев $LT_{2,n}$.

1.3 Елементи теорії складності алгоритмів

Нехай дано два скінчені алфавіти X (вхідний алфавіт) і Y (вихідний алфавіт).

Означення 1.28. *Алгоритм* — це процес послідовної побудови слів вихідного алфавіту за певними словами вхідного алфавіту, який розбивається на елементарні кроки, є направленим і масовим.

Нехай A – деякий алготм, що приймає на вхід значення фіксованої довжини вигляду $x \in \{0,1\}^*$.

Означення 1.29. [20] Функція $t_A : \{0,1\}^* \to \mathbb{N}$ будемо визначати, як кількість кроків $t_A(x)$ через які алгоритм A зупиняється для довільного $x \in \{0,1\}^*$.

Нас буде цікавити залежність кількості кроків алгоритму A від довжини вхідних даних, особливо у наступних двох випадках: максимум та середнє значення по всім вхідних даних однакової довжини n. Тому для t_A розглянемо наступні функції.

Означення 1.30. [20] Функцію $T_{A,max} : \mathbb{N} \to \mathbb{N}$ будемо називати *часовою складністю алгоритму* A та визначати як

$$T_{A,max}(n) = \max_{x \in \{0,1\}^n} \{t_A(x)\}$$

Означення 1.31. [36] Функцію $T_{A,avg} : \mathbb{N} \to \mathbb{N}$ будемо називати *середньою складністю алгоритму* A та визначати як

$$T_{A,avg}(n) = \frac{\sum\limits_{x \in \{0,1\}^n} t_A(x)}{\left|\{0,1\}^n\right|}$$

Введемо означення асимптотичної оцінки O, яку ще іноді називають O-нотацією або ж O-позначенням.

Означення 1.32. [20] Функція $f : \mathbb{N} \to \mathbb{N}$ рівна $O(g)$, де $g : \mathbb{N} \to \mathbb{N}$, якщо існує додатня константа c така, що для будь-якого достатньо великоно $n \in \mathbb{N}$ виконується:

$$f(n) \leq c \cdot g(n).$$

1.4 Коди на групах підстановок

Означення 1.33. [18] *Підстановочним кодом*, або просто *кодом* довжини n та з мінімальною відстанню d метрики $\mathbf{d}$ будемо називати таку множину підстановок $C \subset S_n$, що для кожної пари різних $\pi, \sigma \in C$:

$$\mathbf{d}(\pi, \sigma) \geq d.$$

$A_0(n, d)$ — *максимальний розмір (потужність) підстановочного коду* довжини n та з найменшою відстанню d [18].

1.5 Висновки до розділу

У першому розділі описані основні відомі означення та твердження, що використовуються у дисертації. Зокрема розглянуто основні твердження, що стосуються силовських p-підгруп симетричних груп, метрики Хеммінга на підстановках, дерев та графів. Наведені деякі визначення із теорії складності алгоритмів, в тому числі асимптотичної оцінки O. Також введенні означення кодів на групах підстановок.

2 Зображення підстановок із групи $Syl_2(S_{2^n})$ бінарними n-рівневими кореневими деревами з мітками

Введемо нумерацію вершини на дереві T_n наступним чином. Нехай v – вершина дерева T_n j-го рівня, i — номер вершини v на рівні j. *Координатами вершини v дерева D* будемо називати пару (j, i). При цьому координати кожної веришин дерева T_n знаходяться у межах $i \in \{1, \ldots, 2^j\}$, $j \in \{0, \ldots, n\}$.

Позначимо множину координат всіх вершин дерева T_n як $Coord(T_n)$.

Визначимо відображення $c : V(T_n) \to Coord(T_n)$ за правилом:

$$c(v) = (j, i), \text{ якщо } (j, i) \text{ – пара координат вершини } v \text{ дерева } T_n.$$

Зауважимо, що $(j, i) < (k, r)$, якщо $j < k$ або $j = k$ та $i < r$. Також будемо казати, що $v < w$ якщо $c(v) < c(w)$.

Нехай D – деяке дерево із $LT_{2,n}$.

Визначимо такі множини:

1. $OC(D) = \{(j, i) \in Coord(T_n) |$ (j, i) є координатами вершин, що мають міткою 1 у дереві D $\}$.

2. $OV(D) = \{v \in V(T_n) |$ пара координат (j, i) вершини v належать до множини $OC(D)\}$.

Вважаємо, що ці множини впорядкована за $<$:

$$OC(D) = (OC(D), <) \text{ та } OV(D) = (OV(D), <).$$

Приклад 2.1. *Розгленемо дерево $D \in LT_{2,4}$ (див. рис. 1):*

Тоді маємо таку множину координат з мітками 1 на вершинах:

$OC(D) = \{(0, 1), (2, 2), (2, 3), (2, 4), (3, 2), (3, 3), (3, 6)\}$.

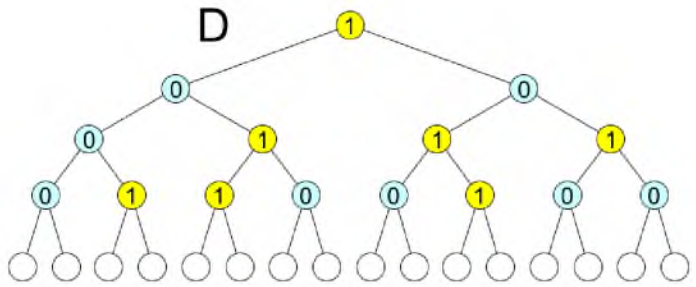


Рис. 1 Дерево D із множини $LT_{2,4}$

Введемо *операцію на множині всіх вершин* дерева D.

1. $switch(D, v)$ ="переставити між собою два піддерева дерева D, для яких вершина v є коренем";

2. $switch(D, j, i)$ ="переставити між собою два піддерева дерева D, для яких вершина v з координатами (j, i) є коренем".

Приклад 2.2. *Розглянемо операції switch для дерева $D \in LT_{2,3}$ та його вершини v:*

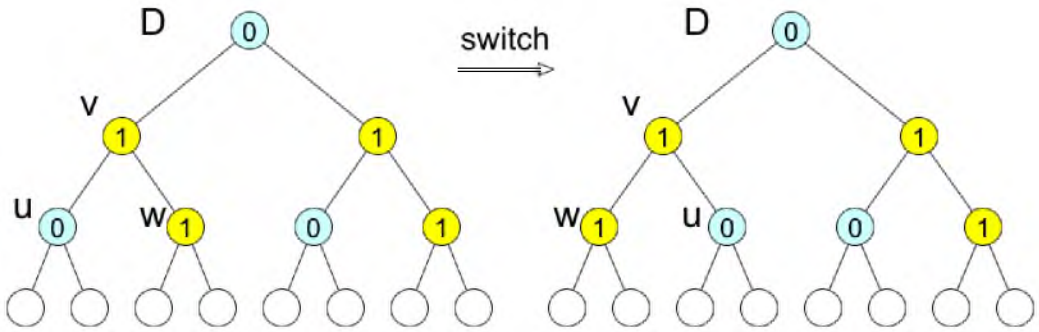


Рис. 2 Операція $switch$ для дерева $D \in LT_{2,3}$ відносно його вершини v

Для кожного дерева $D \in LT_{2,n}$ маємо такі числові характеристики:

1. На кожному j-му рівні дерева розташовано 2^j вершини $j \in \{0, 1, \cdots, n\}$.

2. Кількість висячих вершин n-рівневого дерева дорівнює 2^n.

3. Загальна кількість вершин, де можуть бути розміщені мітки (з 0-го по $(n-1)$-рівні), становить $2^0 + 2^1 + \cdots + 2^{n-1} = 2^n - 1$.

4. Кількість різних вершиннопомічених n-рівневих дерев 2^{2^n-1}. Тобто потужність множини $|LT_{2,n}| = 2^{2^n-1}$.

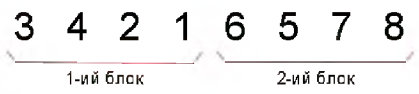
Нижню стрічку $a = (a_1, a_2, \ldots, a_{2^n})$ підстановки $\pi = \begin{pmatrix} 1 & 2 & \ldots & 2^n \\ a_1 & a_2 & \ldots & a_{2^n} \end{pmatrix}$ будемо називати *блоком* елементів. Підстановка π називається 2-*роздільною*, якщо для неї можливо виконати наступну перевірку.

1. На першому кроці розділяємо блок a на 2 підблоки однакової довжини: $u_1 = (a_1, \ldots, a_{2^{n-1}})$ та $u_2 = (a_{2^{n-1}+1}, \ldots, a_{2^n})$. Після чого перевіряємо чи кожен елемент з u_1 більший (менший) за кожен елемент з u_2.

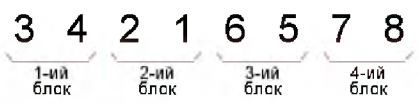
2. Якщо перший крок виконується, то далі повторюємо розділення блоків u_1 та u_2 на підблоки $u_{1,1}$, $u_{1,2}$ та $u_{2,1}$ $u_{2,2}$ і знову виконуємо перевірку величин елементів між відповідними підблоками I так далі, доки не отримаємо підблоки, що складатимуться з одного елементу.

Приклад 2.3. *Розглянемо підстановку* $\pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 3 & 4 & 2 & 1 & 6 & 5 & 7 & 8 \end{pmatrix} \in S_8$.

Тут $n = 3$. *Нижній рядок підстановки розіб'ємо на блоки довжин* $m = 4$, *а потім кожен з них на блоки довжини* $m = 2$:



$m = 4$. Кожен елемент 1-го блоку $u_1 = (3, 4, 2, 1)$ менший за кожен елемент 2-го блоку $u_2 = (6, 5, 7, 8)$.



$m = 2$. Кожен елемент 1-го блоку більший за кожен елемент 2-го та кожен елемент 3-го блоку менший за кожен елемент 4-го.

Отже, підстановка π є 2-роздільною.

За означенням вінцевого добутку [37] випливає, що будь-які 2-роздільні підстановки є елементами вінцевого добутку $\underbrace{S_2 \wr \ldots \wr S_2}_{n \text{ разів}}$. Отже, ці підстановки належать до силовської 2-підгрупи групи S_{2^n} відповідно до [32].

2.1 Операція множення бінарних дерев

Нехай вершина v має координати $c(v) = (r; k)$ у дереві D з мітками, тобто вершина v є k-тою вершиною рівня r дерева D. Відповідно до вершини v, маємо визначене k-те піддерево (рахуючи відповідні корені рівня зліва направо), корінь якого знаходиться на рівні r. Зауважимо, що всі рівні j цього піддерева будуть більшими за r.

Кількість вершин на рівні j будь-якого піддерева з коренем на рівні r буде становити 2^{j-r} [2]. А тому, враховуючи, що ми розглядаємо k-те піддерево, то всі вершини на рівні j нашого піддерева матимуть номери від $(k-1) \cdot 2^{j-r} + 1$ до $k \cdot 2^{j-r}$ включно.

Зазначимо, що для відповідного k-го піддерева дерева D існує своє ліве і праве піддерево.

Наведемо алгоритм операції $switch(D, v)$ – операції над деревом D відносно його вершини v, який буде використовуватися в алгоритмі множення дерев. У запропонованому алгоритмі здійснюється перестановка вершин, для яких вершина v є коренем. Образом кожної вершини w, що мала мітку 1 у лівому k-піддереві дерева D буде відповідна вершина w' у правому k-піддереві, друга координата якої збільшиться на половину кількості всіх вершин, тобто $c(w) = (j, i) \to c(w') = (j, i + \frac{2^{j-r}}{2})$. А образом кожної вершини w, що мала мітку 1 у правому k-піддереві дерева D буде вершина w' у лівому k-піддереві, друга координата якої зменшиться на цю ж величину: $c(w) = (j, i) \to c(w') = (j, i + \frac{2^{j-r}}{2})$. Решта вершин, що не знаходяться в підереві з коренем v залишаться незмінними в дереві D.

Algorithm 1: Алгоритм switch для вершини дерева

Input: Дерево D, задане множиною вершин з мітками $OC(D)$, та координати $(r; k)$ його вершини v.

Output: Змінене дерево D із визначеною за алгоритмом множиною $OC(D)$.

```
1 for j := r + 1 to n − 1 do
2     for i := (k − 1) ∗ 2^{j−r} + 1 to (k − 1) ∗ 2^{j−r} + 2^{j−r}/2 do
3         i_1 := i;
4         i_2 := i + 2^{j−r}/2;
5         if (j; i_1) ∈ OC(D) and (j; i_2) ∉ OC(D) then
6             OC(D) := (OC(D) \ (j; i_1)) ∪ {(j; i_2)}
7         if (j; i_1) ∉ OC(D) and (j; i_2) ∈ OC(D) then
8             OC(D) := (OC(D) \ (j; i_2)) ∪ {(j; i_1)}
```

Теорема 2.1. *Складність алгоритму switch для вершини r-го рівня дерева із $LT_{2,n}$ становить $O(2^{n-r})$.*

Доведення. В алгоритмі 1 маємо 2 цикли, в кожному з яких виконується до 11 елементарних операцій:

- до 3-х операцій присвоювання – стрічки 3, 4 та 6 або 8
- до 4-х операцій перевірки належності – стрічки 5 та 7
- до 2-х операцій кон'юнкції – стрічки 5 та 7
- до 2-х множинних операцій – стрічки 6 та 8

При цьому, кількість різних i для фіксованого j становить:

$$((k-1) * 2^{j-r} + \frac{2^{j-r}}{2}) - ((k-1) * 2^{j-r} + 1) + 1 = 2^{j-r}.$$

Таким чином, загальна кількість різних пар (j, i) дорівнює:

$$\sum_{r+1}^{n-1} 2^{j-r} = 2^1 + 2^2 + \ldots + 2^{n-r-1} = 2^{n-r} - 2,$$

що співдає з кількістю вершин піддерева дерева D у вершині r-го рівня, без кореня. Тому маємо наступну складність алгоритму:

$$O(11 \cdot (2^{n-r} - 2)) = O(2^{n-r})$$

□

Задамо бінарну дію $*$ на множині дерев $LT_{2,n}$ за допомогою наступного алгоритму множення.

Algorithm 2: Алгоритм множення дерев

Input: Два дерева: D_1 – лівий множник, D_2 – правий множник.

Output: Дерево D із визначеною за алгоритмом множиною $OC(D)$.

1 $OC(D) := OC(D_2)$;

2 **for** $(j, i) \in (OC(D_1), <)$ **do**

3 $\quad switch(D, j, i)$;

4 $\quad OC(D) := OC(D) \triangle \{(j, i)\}$;

На кроці 2 алгоритму розглядаємо (j, i) – координати вершини з міткою 1 дерева D_1. Оскільки, D_1 та D належать множині $LT_{2,n}$, то в дереві D теж буде існувати вершина з такими ж координатами. Далі для дерева D та вказаних координат (j, i):

- на кроці 3 виконуємо $switch(D, j, i)$;
- на кроці 4 змінюємо мітку вершини на протилежну.

Зауважимо, що в даному алгоритмі символом $\triangle$ ми позначили симетричну різницю між двома множинами.

Зазначимо, що в алгоритмі 2 прохід по множині $(OC(D_1), <)$ еквівалентен проходу по множині $(OV(D_1), <)$.

Приклад 2.4. *Розглянемо два дерева D_1, D_2 (див. рис. 3) та обчислимо їх добуток за алгоритмом множення.*

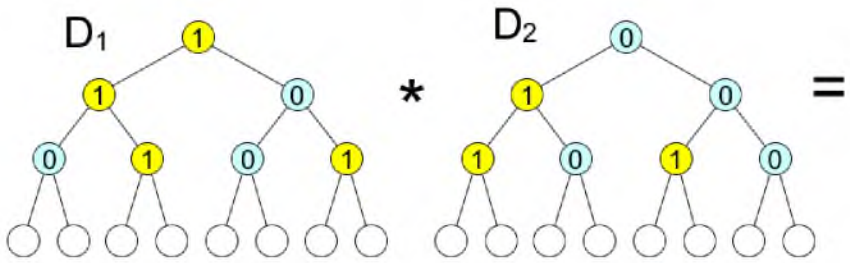
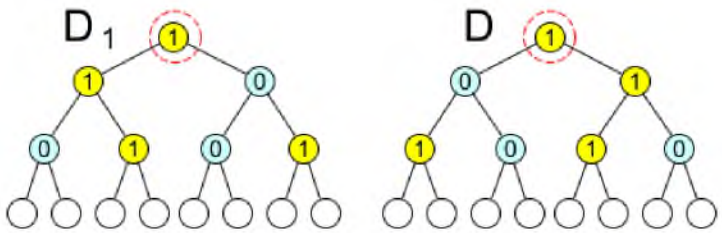


Рис. 3 Вхід для алгоритму 2

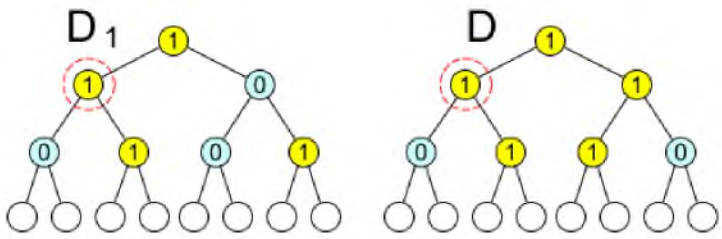
Для початку маємо: $OC(D_1) = \{(0,1),(1,1),(2,2),(2,4)\}$,
$OC(D_2) = \{(1,1),(2,1),(2,3)\}$. *Та ініціалізуємо множину* $OC(D) = OC(D_2)$.

Ітерація 1. $(j,i) = (0,1)$:



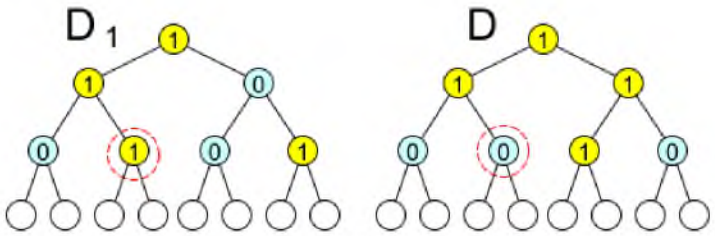
Після $switch(D,0,1)$ *отримали множину* $OC(D) = \{(1,2),(2,1),(2,3)\}$. *А далі* $OC(D) = OC(D)\Delta\{(0,1)\} = \{(0,1),(1,2),(2,1),(2,3)\}$

Ітерація 2. $(j,i) = (1,1)$:



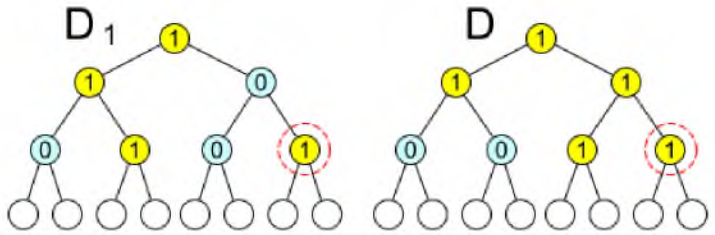
Після $switch(D, 1, 1)$ *отримали множину* $OC(D) = \{(0,1),(1,2),(2,2),(2,3)\}$. *А далі* $OC(D) = OC(D)\Delta\{(1,1)\} = \{(0,1),(1,1),(1,2),(2,2),(2,3)\}$

Ітерація 3. $(j,i) = (2,2)$:



Після $switch(D, 2, 2)$ *отримали множину* $OC(D) = \{(0,1),(1,1),(1,2),(2,2),(2,3)\}$. *А далі* $OC(D) = OC(D)\Delta\{(2,2)\} = \{(0,1),(1,1),(1,2),(2,3)\}$

Ітерація 4. $(j,i) = (2,4)$:



Після $switch(D, 2, 4)$ *отримали множину* $OC(D) = \{(0,1),(1,1),(1,2),(2,3)\}$. *А далі* $OC(D) = OC(D)\Delta\{(2,4)\} = \{(0,1),(1,1),(1,2),(2,3),(2,4)\}$

Результатом множення буде таке дерево (див. рис. 4):

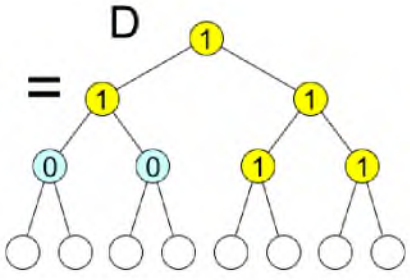


Рис. 4 Вихід для алгоритму 2

Теорема 2.2. *Складність алгоритму множення дерев із* $LT_{2,n}$ *становить* $O(n \cdot 2^n)$.

Доведення. В алгоритмі 2 множення дерев маємо цикл по (j,i), координатам з множини $OC(D_1)$. Тобто кількість ітерацій циклу дорівнює кількості елементів множини $OC(D_1)$. Найбільшу потужність ця множина матиме, якщо дерево D_1 на всіх вершинах з 0-го по $(n-1)$-ий рівні матиме мітки 1. Тобто коли:

$$|OC(D_1)| = |Coord(T_n)| = 2^n - 1.$$

У самому циклі для кожної пари координат (j,i) ми виконуємо по 2 операції - $switch(D,j,i)$ та симетричну різницю $OC(D) \triangle (j,i)$, де:

1. $switch(D,j,i)$ є операцією максимальної складності $O(2^{(n-j)})$;
2. $OC(D) \triangle (j,i)$ є композицією двух елементарних операцій, оскільки другий доданок симетричної різниці є одноелементною множиною.

Кількість різних координат (j,i) на j-му рівні становить 2^j.

Отже, всього буде наступна кількість операцій:

$$\sum_{j=0}^{n-1} 2^j \cdot (1 + 2^{(n-j)}) = (2^n - 1) + n \cdot 2^n = (n+1) \cdot 2^n - 1$$

Тому складність алгоритму дорівнює: $O((n+1) \cdot 2^n - 1) = O(n \cdot 2^n)$. □

2.2 Відображення вершин $\mathbb{ACT}$ та його властивості

Нехай $v_0, v, w \in V(T_n)$.

Позначимо $Path_{v_0}(v) := \{v_0, \ldots, v_{j-1}\}$ – шлях, що з'єднує корінь v_0 з деякою вершиною v j-ого рівня, $v_0, \ldots, v_{j-1} \in V(T_n)$, де множина ребер шляху опускається.

Означення 2.1. Говоритимемо, що *вершина* v *знаходиться під вершиною* w (вершина w знаходиться над вершиною v), якщо $w \in Path_{v_0}(v)$. Для цього використаємо позначення $v \succ w$.

Зауважимо, що якщо $v \succ w$, то $v > w$. В інший бік твердження не завжди виконується.

Означення 2.2. Нехай w – деяка фіксована вершина дерева, $w \in V(T_n)$. Визначимо функцію $\mathbb{ACT}_w : V(T_n) \to V(T_n)$ наступним чином:
$\mathbb{ACT}_w(v) = v'$ тоді і тільки тоді, коли v' є образом вершини v після $switch(T_n, w)$.

Зауважимо, що:

- якщо $v \succ w$ і $c(w) = (k, r)$, $c(v) = (j, i)$, то $c(v') = (j, i')$, де

$$i' = \begin{cases} i + 2^{j-k-1}, \text{ якщо } i \leq (r-1) \cdot 2^{j-k} + 2^{j-k-1} \\ \text{тобто, якщо } v \text{ знаходиться у лівій гілці піддерева з коренем } w \\ i - 2^{j-k-1}, \text{ якщо } i \geq (r-1) \cdot 2^{j-k} + 2^{j-k-1} + 1 \\ \text{тобто, якщо } v \text{ знаходиться у правій гілці піддерева з коренем } w \end{cases}$$

- якщо $v \not\succ w$, то $\mathbb{ACT}_w(v) = v$.

Зауваження 2.1. *Для довільної вершини $w \in V(T_n)$ маємо:*

$$\mathbb{ACT}_w^2 = id.$$

Нехай $A = \{w_1, \ldots, w_t\}$ – впорядкована множина, $B = \{v_1, \ldots, v_l\}$, $A, B \subset V(T_n)$ та v – довільна вершина з $V(T_n)$. Визначимо:

- $\mathbb{ACT}_v(B) := \{\mathbb{ACT}_v(v_1), \ldots, \mathbb{ACT}_v(v_t)\}$
- $\mathbb{ACT}_A(v) := (\mathbb{ACT}_{w_1} \cdot \ldots \cdot \mathbb{ACT}_{w_t})(v) = \mathbb{ACT}_{w_t}(\ldots(\mathbb{ACT}_{w_1}(v))\ldots)$

  Будемо вважати, що для порожньої множини $A = \emptyset$: $\mathbb{ACT}_A = id$.
- $\mathbb{ACT}_A(B) := \{\mathbb{ACT}_A(v_1), \ldots, \mathbb{ACT}_A(v_l)\}$

Нехай $D \in LT_{2,n}$, $w \in V(D)$.

Зауважимо, що результатом дії операції $switch(D, w)$ для дерева D по вершині w буде дерево, що визначається такою множиною вершин з мітками 1: $\mathbb{ACT}_w(OV(D))$.

Теорема 2.3. *Нехай $D_1, D_2 \in LT_{2,n}$ – довільні дерева. Тоді їхнім добутком буде дерево, у якому множина вершин з мітками 1 визначається як симетрична різниця двох множини вершин з мітками 1, де:*

- *перша – це результат відображення $\mathbb{ACT}$ вершин з мітками 1 дерева D_2 відносно впорядкованої за $<$ множини вершин з міткаим 1 дерева D_1,*
- *а друга – це множина з мітками 1 самого дерева D_1.*

Тобто:

$$OV(D_1 * D_2) = (\ \mathbb{ACT}_{(OV(D_1),<)}(OV(D_2))\ ) \triangle\ OV(D_1)$$

Доведення. Розглянемо кроки $2-4$ в алгоритмі 2 множення дерев. Цей цикл, можна замінити на два цикли наступним чином:

$$\begin{aligned}
&\textbf{For } v \in (OV(D_1), <) : \\
&\quad switch(D, v); \\
&\textbf{For } v \in (OV(D_1), <) : \\
&\quad OV(D) := OV(D) \triangle \{v\};
\end{aligned}$$

Тоді перший цикл можна переписати як: $OV(D) := \mathbb{ACT}_{(OV(D_1),<)}(OV(D))$.

Другий цикл можна переписати як: $OV(D) := OV(D) \triangle OV(D_1)$.

Враховуючи позначення $D = D_1 * D_2$, та крок 1 алгоритму 2, отримаємо такий результат дії алгоритму:

$$OV(D_1 * D_2) = \mathbb{ACT}_{(OV(D_1),<)}(OV(D_2))\ \triangle\ OV(D_1)$$

□

2.2.1 Властивості відображення вершин

Лема 2.1. *Композиція відображень* $\mathbb{ACT}$ *для пари фіксованих вершин, що не знаходяться одна під одною, є комутативною. Тобто, для довільних вершин* $v_1, v_2 \in V(T_n)$*, таких що* $v_1 \not\succ v_2$ *та* $v_2 \not\succ v_1$ *маємо:*

$$\mathbb{ACT}_{v_1} \cdot \mathbb{ACT}_{v_2} = \mathbb{ACT}_{v_2} \cdot \mathbb{ACT}_{v_1}$$

Доведення. Розглянемо довільну вершину $t \in V(T_n)$.

Випадок 1. Нехай $t \succ v_1$. Тоді для $\mathbb{ACT}_{v_1} t := t'$ отримаємо:

$$t' \succ v_1,\ t \not\succ v_2 \text{ та } t' \not\succ v_2.$$

Тоді:

$$\mathbb{ACT}_{v_2}(\mathbb{ACT}_{v_1} t) = \mathbb{ACT}_{v_2} t' = t'$$

$$\mathbb{ACT}_{v_1}(\mathbb{ACT}_{v_2} t) = \mathbb{ACT}_{v_1} t = t'$$

Випадок 2. Нехай $t \succ v_2$. Тоді міркування аналогічні до Випадку 1.

Випадок 3. Нехай $t \not\succ v_1$ та $t \not\succ v_2$. Тоді отримаємо:

$$\mathbb{ACT}_{v_2}(\mathbb{ACT}_{v_1} t) = \mathbb{ACT}_{v_2} t = t$$

$$\mathbb{ACT}_{v_1}(\mathbb{ACT}_{v_2} t) = \mathbb{ACT}_{v_1} t = t$$

Отже, $\mathbb{ACT}_{v_1} \cdot \mathbb{ACT}_{v_2} = \mathbb{ACT}_{v_2} \cdot \mathbb{ACT}_{v_1}$. □

Лема 2.2. *Нехай* $v, v_1 \in V(T_n)$ *та вершина* v_1 *знаходиться на меншому рівні ніж вершина* v*. Тоді відображення* $\mathbb{ACT}$*, що виконується по результату переміщення* $\mathbb{ACT}$ *вершини* v *відносно вершини* v_1*, буде ріносильне композиції відображень* $\mathbb{ACT}$ *відносно* v_1*,* v*, та знову* v_1*. Тобто для* $v_1 < v$ *маємо:*

$$\mathbb{ACT}_{\mathbb{ACT}_{v_1}(v)} = \mathbb{ACT}_{v_1} \cdot \mathbb{ACT}_{v} \cdot \mathbb{ACT}_{v_1} \tag{6}$$

Доведення. Розглянемо випадки, залежно від розміщення вершин.

Випадок 1. Нехай $v \not\succ v_1$. Тоді, за лемою 2.1 маємо:

$$\mathbb{ACT}_v \cdot \mathbb{ACT}_{v_1} = \mathbb{ACT}_{v_1} \cdot \mathbb{ACT}_v. \tag{7}$$

Також, оскільки $v \not\succ v_1$, то $\mathbb{ACT}_{v_1}(v) = v$, а тому:

$$\mathbb{ACT}_{\mathbb{ACT}_{v_1}(v)} = \mathbb{ACT}_v \tag{8}$$

Тоді, використовуючи (7) та (8), маємо:

$$\mathbb{ACT}_{v_1} \cdot \mathbb{ACT}_v \cdot \mathbb{ACT}_{v_1} = \mathbb{ACT}_{v_1} \cdot \mathbb{ACT}_{v_1} \cdot \mathbb{ACT}_v = id \cdot \mathbb{ACT}_v = \mathbb{ACT}_v = \mathbb{ACT}_{\mathbb{ACT}_{v_1}(v)}$$

Випадок 2. Нехай $v \succ v_1$. Позначимо $v' := \mathbb{ACT}_{v_1}(v)$.

Зафіксуємо довільну вершину $t \in V(T_n)$ та розглянемо різні варіанти її розміщення.

- Нехай $t \not\succ v_1$. Як наслідок, $t \not\succ v$ та $t \not\succ v'$, а тому:

  $$(\mathbb{ACT}_{v_1} \cdot \mathbb{ACT}_v \cdot \mathbb{ACT}_{v_1})(t) = t = \mathbb{ACT}_{v'}(t)$$

- Нехай $t \succ v_1$ та $t \not\succ v'$.

  Тоді з одного боку маємо: $\mathbb{ACT}_{v'}(t) = t$.

  Зауважимо, що оскільки $t \not\succ v' = \mathbb{ACT}_{v_1}(v)$, то при взятті образу відносно v_1 отримаємо:

  $$\mathbb{ACT}_{v_1}(t) \not\succ \mathbb{ACT}_{v_1}(\mathbb{ACT}_{v_1}(v)) = v,$$

  а тому $\mathbb{ACT}_v(\mathbb{ACT}_{v_1}(v)) = \mathbb{ACT}_{v_1}(v)$. Отже, з іншого боку:

  $$(\mathbb{ACT}_{v_1} \cdot \mathbb{ACT}_v \cdot \mathbb{ACT}_{v_1})(t) = \mathbb{ACT}_{v_1}(\mathbb{ACT}_v(\mathbb{ACT}_{v_1}(t))) =$$

  $$= \mathbb{ACT}_{v_1}(\mathbb{ACT}_{v_1}(t)) = t.$$

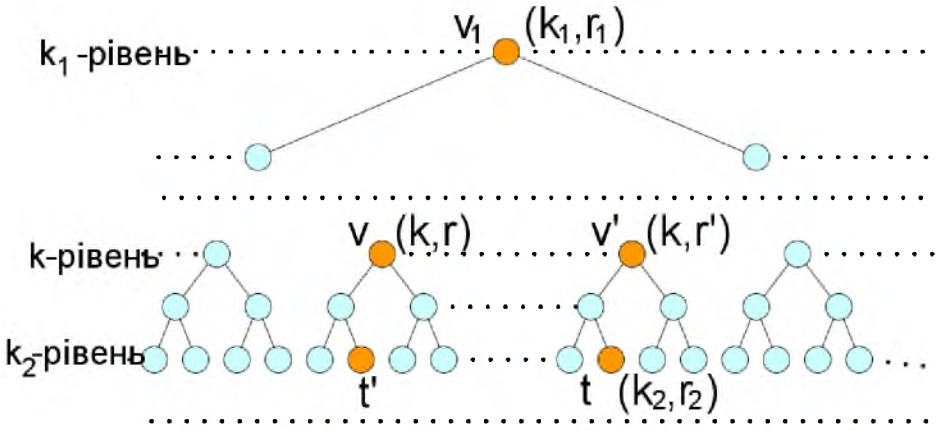


Рис. 5 Розміщення вершин на дереві T_n

- Нехай $t \succ v_1$ та $t \succ v'$. Нехай вершини v, v_1 та t мають наступні координати: $c(v) = (k, r)$, $c(v_1) = (k_1, r_1)$ та $c(t) = (k_2, r_2)$, де $k_1 < k < k_2$, $1 \leq r_1 \leq 2^{k_1}$ (див. рис. 5).

  Не обмежуючи загальності, будемо вважати що:

  (a) вершина v знаходиться лівіше від v'. Тому:

$$r' = r + 2^{k-k_1-1} \tag{9}$$

  (b) вершина t розміщена у лівій гілці піддерева з коренем v' дерева T_n. Тоді:

$$(r' - 1) \cdot 2^{k_2-k} + 1 \leq r_2 \leq (r' - 1) \cdot 2^{k_2-k} + 2^{k_2-k-1} \tag{10}$$

  Тоді за припущенням (b) та нерівності (10) випливає, що ліва частина рівності (6) з умови леми становить:

$$c(\mathbb{ACT}_{v'}(t)) = (k_2, r_2 + 2^{k_2-k-1}) \tag{11}$$

  Тепер знайдемо значення правої частини рівності (6).

  Із припущень (a) та (b) маємо, що t розміщена у правій гілці піддерева з коренем v_1 дерева T_n. Тоді образ вершини t буде у лівій гілці піддерева з коренем

v_1. Тому $\mathbb{ACT}_{v_1}(t) = t'$, де $c(t') = (k_2, r_2 - 2^{k_2-k_1-1})$. Отже,

$$(\mathbb{ACT}_{v_1} \cdot \mathbb{ACT}_v \cdot \mathbb{ACT}_{v_1})(t) = (\mathbb{ACT}_v \cdot \mathbb{ACT}_{v_1})(t') \tag{12}$$

Використаємо обмеження (10) на r_2 та підставимо значення r' із (9):

$$r_2 \leq (r' - 1) \cdot 2^{k_2-k} + 2^{k_2-k-1} = (r - 1 + 2^{k-k_1-1}) \cdot 2^{k_2-k} + 2^{k_2-k-1} =$$

$$= (r-1)2^{k_2-k} + 2^{k_2-k-1} + 2^{\cancel{k}-k_1-1+k_2-\cancel{k}}$$

Отримаємо:

$$\underbrace{r_2 - 2^{k_2-k_1-1}}_{\text{друга координата вершини t}'} \leq (r-1)2^{k_2-k} + 2^{k_2-k-1}$$

що означає, що t' знаходиться у лівій гілці піддерева з коренем v дерева T_n. А тому $\mathbb{ACT}_v(t') = t''$, де $c(t'') = (k_2, r_2 - 2^{k_2-k_1-1} + 2^{k_2-k-1})$. Отже, продовжуючи рівність (12), маємо:

$$(\mathbb{ACT}_v \cdot \mathbb{ACT}_{v_1})(t') = \mathbb{ACT}_{v_1}(t'') \tag{13}$$

Оскільки $t' \succ v$, то $t'' \succ v$. Окрім того, вершина v розміщена у лівій гілці піддерева з коренем v_1. Тому t'' також знаходиться у лівій гілці піддерева з коренем v_1. Тоді маємо такі координати образу вершини t'' відносно вершини v_1:

$$c(\mathbb{ACT}_{v_1}(t'')) = (k_2, r_2 - \cancel{2^{k_2-k_1-1}} + 2^{k_2-k-1} + \cancel{2^{k_2-k_1-1}}) = (k_2, r_2 + 2^{k_2-k-1}) \tag{14}$$

Із рівностей (11), (12), (13) та (14) маємо:

$$\mathbb{ACT}_{\mathbb{ACT}_{v_1}(v)}(t) = (\mathbb{ACT}_{v_1} \cdot \mathbb{ACT}_v \cdot \mathbb{ACT}_{v_1})(t).$$

□

Лема 2.3. *Нехай $A \subset V(T_n)$ – довільна впорядкована множина та $B, C \subset V(T_n)$ – довільні множини. Тоді переміщення результату симетричної різниці множин B та C відносно множини A рівносильне симетричній різниці образу переміщення множин B і C відносно A. Тобто:*

$$\mathbb{ACT}_A(B \triangle C) = \mathbb{ACT}_A(B) \triangle \mathbb{ACT}_A(C)$$

Доведення.

$$\mathbb{ACT}_A(B \triangle C) = \{\mathbb{ACT}_A(v) | v \in B \triangle C\} =$$

$$= \{\mathbb{ACT}_A(v) | v \in B\} \triangle \{\mathbb{ACT}_A(v) | v \in C\} = \mathbb{ACT}_A(B) \triangle \mathbb{ACT}_A(C)$$

□

Лема 2.4. *Нехай $a, b \in V(T_n)$ – довільні вершини. Тоді композиція переміщення $\mathbb{ACT}$ відносно цих вершин рівносильна переміщенню відносно впорядкованої за $<$ множини, що є результатом симетричної різниці образу переміщення вершини a відносно b та самої вершини b. Тобто:*

$$\mathbb{ACT}_a \cdot \mathbb{ACT}_b = \mathbb{ACT}_{(\{\mathbb{ACT}_b(a)\} \triangle \{b\}, <)} \tag{15}$$

Доведення. Випадок 1. Якщо $a < b$, то $\mathbb{ACT}_b(a) = a$ та $(\{a\} \triangle \{b\}, <) = \{a, b\}$. Тоді:

$$\mathbb{ACT}_{(\{\mathbb{ACT}_b(a)\} \triangle \{b\}, <)} = \mathbb{ACT}_{\{a,b\}} = \mathbb{ACT}_a \cdot \mathbb{ACT}_b$$

Випадок 2. Якщо $a = b$, то $\mathbb{ACT}_b(a) = \mathbb{ACT}_a(a) = a$ та $(\{a\} \triangle \{b\}, <) = \emptyset$. Тоді:

$$\mathbb{ACT}_{(\{\mathbb{ACT}_b(a)\} \triangle \{b\}, <)} = \mathbb{ACT}_\emptyset = id = \mathbb{ACT}_a \cdot \mathbb{ACT}_a$$

Випадок 3. Якщо $b < a$, то $b < \mathbb{ACT}_b(a)$. А тому $(\{\mathbb{ACT}_b(a)\} \triangle \{b\}, <) = \{b, \mathbb{ACT}_b(a)\}$. За лемою 2.2 маємо $\mathbb{ACT}_{\mathbb{ACT}_b(a)} = \mathbb{ACT}_b \cdot \mathbb{ACT}_a \cdot \mathbb{ACT}_b$. Тоді:

$$\mathbb{ACT}_{(\{\mathbb{ACT}_b(a)\} \triangle \{b\}, <)} = \mathbb{ACT}_{\{b, \mathbb{ACT}_b(a)\}} = \mathbb{ACT}_b \cdot \mathbb{ACT}_{\mathbb{ACT}_b(a)} = \mathbb{ACT}_a \cdot \mathbb{ACT}_b.$$

□

Лема 2.5. *Нехай $A \subset V(T_n)$ – довільна впорядкована за $<$ множина вершин дерева та $b \in V(T_n)$ – довільна вершина цього дерева. Тоді композиція відображень $\mathbb{ACT}$ відносно A та b рівносильна переміщенню відносно впорядкованої за $<$ множини, що є результатом симетричної різниці образу множини A відносно вершини b для відображення $\mathbb{ACT}$ та самої вершини b. Тобто:*

$$\mathbb{ACT}_A \cdot \mathbb{ACT}_b = \mathbb{ACT}_{(\mathbb{ACT}_b(A) \triangle \{b\}, <)}$$

Доведення. Нехай $A = \{a_1, \ldots, a_m\}$ and $1 \leq k \leq m$:

$$a_1 < \ldots < a_k \leq b < a_{k+1} < \ldots a_m \tag{16}$$

Тоді, використовуючи попередню лему 2.4 та враховуючи положення b відносно елементів множини A із (16), маємо:

$$\mathbb{ACT}_A \cdot \mathbb{ACT}_b = \mathbb{ACT}_{a_1} \cdot \ldots \cdot \mathbb{ACT}_{a_m} \cdot \mathbb{ACT}_b = \mathbb{ACT}_{a_1} \cdot \ldots \cdot \mathbb{ACT}_{(\{\mathbb{ACT}_b(a_m)\} \triangle \{b\}, <)} =$$

$$= \mathbb{ACT}_{a_1} \cdot \ldots \cdot \mathbb{ACT}_{\{b, \mathbb{ACT}_b(a_m)\}} = \mathbb{ACT}_{a_1} \cdot \ldots \cdot \mathbb{ACT}_b \cdot \mathbb{ACT}_{\mathbb{ACT}_b(a_m)} = \ldots$$

$$\ldots = \mathbb{ACT}_{a_1} \cdot \ldots \cdot \mathbb{ACT}_{a_k} \cdot \mathbb{ACT}_b \cdot \mathbb{ACT}_{\mathbb{ACT}_b(\{a_{k+1}, \ldots, a_m\})}$$

Оскільки для всіх $a \in \{a_1, \ldots, a_k\}$: $\mathbb{ACT}_b(a) = a$, то остання рівність дорівнює:

$$\mathbb{ACT}_{\mathbb{ACT}_b(\{a_1, \ldots, a_k\})} \cdot \mathbb{ACT}_b \cdot \mathbb{ACT}_{\mathbb{ACT}_b(\{a_{k+1}, \ldots, a_m\})} \tag{17}$$

Розглянемо випадки відносно a_k та b:

1. якщо $a_k \neq b$, то (17) дорівнює $\mathbb{ACT}_{(\mathbb{ACT}_b(A) \bigcup \{b\}, <)}$;
2. якщо $a_k = b$, то (17) дорівнює $\mathbb{ACT}_{(\mathbb{ACT}_b(A) \setminus \{b\}, <)}$.

Таким чином, в загальному випадку (17) дорівнює $\mathbb{ACT}_{(\mathbb{ACT}_b(A) \triangle \{b\}, <)}$ □

Лема 2.6. *Нехай $A, B \subset V(T_n)$ – довільні впорядкованих за $<$ множини. Тоді композиція відображень $\mathbb{ACT}$ відносно множин A та B рівносильна переміщенню відносно впорядкованої за $<$ множини, яка є результатом симетричної різниці образу A відносно B для відображення $\mathbb{ACT}$ та самої множини B. Тобто:*

$$\mathbb{ACT}_A \cdot \mathbb{ACT}_B = \mathbb{ACT}_{(\mathbb{ACT}_B(A) \triangle B, <)}$$

Доведення. Нехай $B = \{b_1, \ldots, b_l\}$. За лемою 2.5 для множини A та вершин b_1, b_2 маємо:

$$\mathbb{ACT}_A \cdot \mathbb{ACT}_{b_1} \cdot \mathbb{ACT}_{b_2} = \mathbb{ACT}_{(\mathbb{ACT}_{b_1}(A) \triangle \{b_1\}, <)} \cdot \mathbb{ACT}_{b_2} =$$

$$= \mathbb{ACT}_{(\mathbb{ACT}_{b_2}(\mathbb{ACT}_{b_1}(A) \triangle \{b_1\}, <) \triangle \{b_2\}, <)} = \mathbb{ACT}_{(\mathbb{ACT}_{b_2}(\mathbb{ACT}_{b_1}(A) \triangle \{b_1\}) \triangle \{b_2\}, <)} \tag{18}$$

Оскільки множина B впорядкована, то $b_1 < b_2$. А тому $\mathbb{ACT}_{b_2}(b_1) = b_1$. Із цієї рівності, леми 2.3 та (18) ми маємо:

$$\mathbb{ACT}_{(\mathbb{ACT}_{b_2}(\mathbb{ACT}_{b_1}(A) \triangle \{b_1\}) \triangle \{b_2\}, <)} = \mathbb{ACT}_{(\mathbb{ACT}_{\{b_1, b_2\}}(A) \triangle \{b_1, b_2\}, <)} \tag{19}$$

Тоді із (18) та (19) для всіх $b_3, \ldots, b_l$:

$$\mathbb{ACT}_A \cdot \mathbb{ACT}_B = \mathbb{ACT}_{(\mathbb{ACT}_{\{b_1, b_2\}}(A) \triangle \{b_1, b_2\}, <)} \cdot \prod_{k=3}^{l} \mathbb{ACT}_{b_k} = \ldots = \mathbb{ACT}_{\mathbb{ACT}_B(A) \triangle B}$$

□

Наслідок 2.1. *Нехай $D_1, D_2 \in LT_{2,n}$ – довільні дерева. Відображення $\mathbb{ACT}$ відносно впорядкованої за $<$ множини вершин з мітками 1 дерева, що є результатом добутку дерев D_1 та D_2, рівносильне композиції відображень $\mathbb{ACT}$ відносно впорядкованих множин вершин з міткаим 1 дерев D_2 та D_1 відповідно. Тобто:*

$$\mathbb{ACT}_{(OV(D_1 * D_2), <)} = \mathbb{ACT}_{(OV(D_2), <)} \cdot \mathbb{ACT}_{(OV(D_1), <)}$$

Доведення. Випливає з теореми 2.3 та леми 2.6 □

2.3 Нейтральний елемент бінарних дерев

Позначимо за E — таке дерево з $LT_{2,n}$, що $OV(E) = \emptyset$.

Теорема 2.4. *Дерево $E \in LT_{2,n}$ є нейтральним елементом над $LT_{2,n}$ відносно операції множення дерев $*$.*

Доведення. Для довільного $D \in LT_{2,n}$:

$$E * D = D,$$

бо $OV(E) = \emptyset$. Як наслідок, за алгоритмом множення 2 дерево D не зміниться.

$D * E = D$, бо дерево E має лише нульові мітки, а тому на кожній ітерації алгоритму множення 2 з'являться мітки 1 лише на вершинах $v \in OV(D)$. Як наслідок, $OV(D * E) = OV(D)$.

Отже, $D * E = E * D = E$. А тому, E — нейтральний елемент. □

2.4 Операція взяття оберненного дерева

Для впорядкованої множини $A = \{w_1, w_2, \ldots, w_t\}$ покладемо:

$$R(A) := \{w_t, w_{t-1}, \ldots, w_1\} \text{ – реверсивна множина до } A.$$

Розглянемо алгоритм знаходження оберненого дерева на $LT_{2,n}$.

Algorithm 3: Алгоритм знаходження оберненного дерева до заданого

Input: $OV(D)$ – множина всіх вершин з мітками 1 дерева D.

Output: $OV(D^{-1})$ – множина всіх вершин з мітками 1 дерева D^{-1}.

```
1 OV(D^{-1}) = ∅ ;
2 for v ∈ (OV(D), <) do
3     u = v ;
4     for w ∈ R(Path_{v_0}(v)) do
5         if w ∈ OV(D) then
6             u = 𝔸ℂ𝕋_w(u) ;
7     OV(D^{-1}) = OV(D^{-1}) ⋃ {u} ;
```

Кожну вершину v з міткою 1 дерева D будемо переміщати за допомогою відображення $\mathbb{ACT}$ відносно кожної вершини w, яка теж має мітку 1 на D а також знаходиться на шляху між v та коренем v_0. Зазначимо, що множина $Path_{v_0}(v)$ впорядкована за $<$. А тому цикл на кроці 4 виконується від найбільшого до найменшого рівнів.

Приклад 2.5. *Розглянемо дерево $D \in LT_{2,3}$ та за алгоритмом 3 знайдемо обернене до нього D^{-1}. Для початку маємо $OV(D^{-1}) = \emptyset$ (див. рис. 6).*

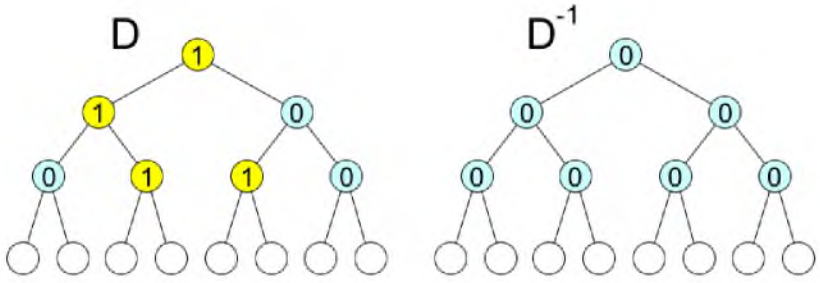
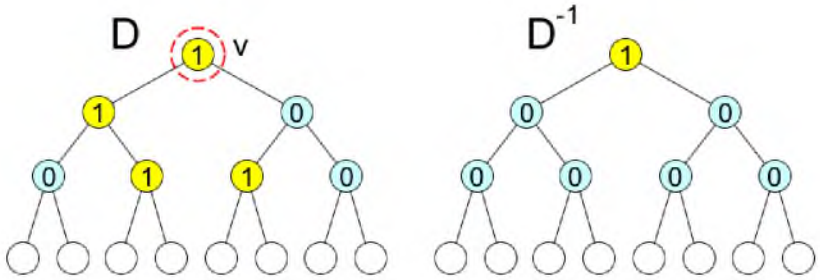
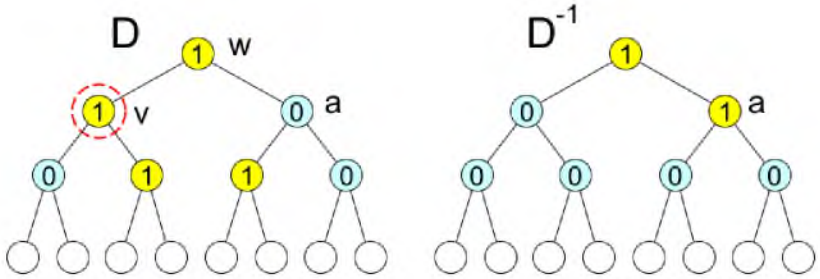


Рис. 6 Вхід для алгоритму 3 та ініціалізація D^{-1}

Ітерація 1:

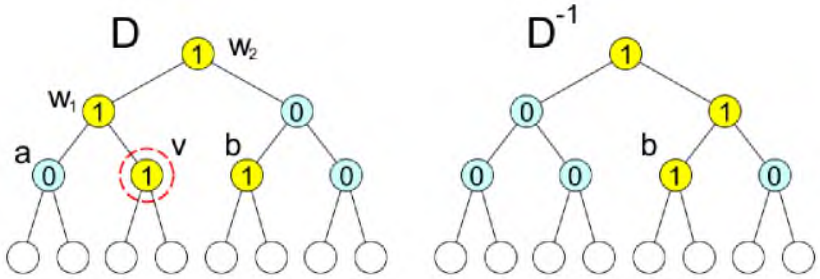


Ітерація 2:



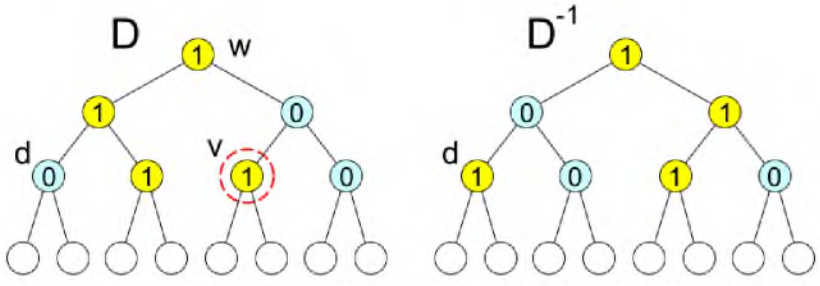
Маємо: $\mathbb{ACT}_w(v) = a$.

Ітерація 3:



Маємо: $\mathbb{ACT}_{w_1}(v) = a$ *та* $\mathbb{ACT}_{w_2}(a) = b$.

Ітерація 4:



Маємо: $\mathbb{ACT}_w(v) = d$.

Перевіримо властивість обернених елементів, використовуючи алгоритм множення 2 *(див. рис.* 7*):*

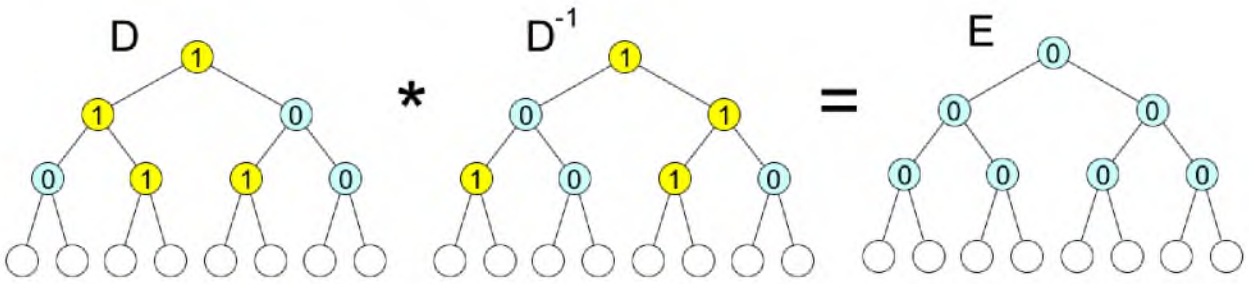


Рис. 7 Перевірка умови, що $D * D^{-1} = E$

Отже, дійсно отримали, що добуток обернених елементів дорівнює нейтральному.

Для доведення коректності алгоритму 3 наведемо наступні властивості операції $\mathbb{ACT}$.

2.4.1 Властивості відображення $\mathbb{ACT}$ над реверсивними множинами

Лема 2.7. *Нехай $A \subset V(T_n)$ – довільна впорядкована за $<$ множина вершин n-рівневого дерева. Тоді композиція відображень $\mathbb{ACT}$ відносно множини A та її реверсивної множини $R(A)$ є комутативною та рівна тривіальному відображенню. Тобто:*

$$\mathbb{ACT}_A \cdot \mathbb{ACT}_{R(A)} = \mathbb{ACT}_{R(A)} \cdot \mathbb{ACT}_A = id$$

Доведення. Нехай $A = \{w_1, w_2, \cdots, w_m\} \subset V(T_n)$. Тоді:

$$R(A) = \{w_m, \cdots, w_2, w_1\}.$$

Розглянемо довільну вершину $v \in V(T_n)$. Для неї:

$$(\mathbb{ACT}_A \cdot \mathbb{ACT}_{R(A)})(v) = (\mathbb{ACT}_{\{w_1,w_2,\ldots,w_m\}} \cdot \mathbb{ACT}_{\{w_m,\ldots,w_2,w_1\}})(v) =$$

$$= (\mathbb{ACT}_{w_1} \cdot \mathbb{ACT}_{w_2} \cdot \ldots \cdot \mathbb{ACT}_{w_m} \cdot \mathbb{ACT}_{w_m} \cdot \ldots \cdot \mathbb{ACT}_{w_2} \cdot \mathbb{ACT}_{w_1})(v)$$

Зазначимо, що $\mathbb{ACT}_w{}^2 = id$. Тому останнє дорівнює:

$$(\mathbb{ACT}_{w_1} \cdot \mathbb{ACT}_{w_2} \cdot \ldots \cdot \mathbb{ACT}_{w_{m-1}} \cdot id \cdot \mathbb{ACT}_{w_{m-1}} \cdot \ldots \cdot \mathbb{ACT}_{w_2} \cdot \mathbb{ACT}_{w_1})(v) = \ldots$$

$$\ldots = (\mathbb{ACT}_{w_1} \cdot \mathbb{ACT}_{w_1})(v) = id(v) = v$$

Отже, маємо:

$$(\mathbb{ACT}_A \cdot \mathbb{ACT}_{R(A)})(v) = v,$$

тобто $\mathbb{ACT}_A \cdot \mathbb{ACT}_{R(A)} = id$.

В інший бік доведення аналогічне.

□

Лема 2.8. *Нехай $v \in V(T_n)$ – довільна вершина дерева та $A \subset V(T_n)$ – впорядкована за $<$ множина вершин цього дерева, при чому всі вершини з A знаходяться на меншому рівні ніж вершина v. Тоді відображення $\mathbb{ACT}$ відносно переміщення вершини v над реверсивною множиною $R(A)$ рівносильне композиції відображень $\mathbb{ACT}$ відносно самої множини A, вершини v та множини $R(A)$.*

Тобто:

$$\mathbb{ACT}_{\mathbb{ACT}_{R(A)}(v)} = \mathbb{ACT}_A \cdot \mathbb{ACT}_v \cdot \mathbb{ACT}_{R(A)}$$

Доведення. Нехай $A = \{w_1, w_2, \ldots, w_m\} \subset V(T_n)$. Згідно позначень, $R(A) = \{w_m, \ldots, w_2, w_1\}$. Тоді:

$$\mathbb{ACT}_A \cdot \mathbb{ACT}_v \cdot \mathbb{ACT}_{R(A)} = \mathbb{ACT}_{w_1} \cdot \ldots \cdot \mathbb{ACT}_{w_{m-1}} \cdot \mathbb{ACT}_{w_m} \cdot$$

$$\cdot \mathbb{ACT}_v \cdot \mathbb{ACT}_{w_m} \cdot \mathbb{ACT}_{w_{m-1}} \cdot \ldots \cdot \mathbb{ACT}_{w_1} \tag{20}$$

За лемою 2.2 для v, w_m: $\mathbb{ACT}_{\mathbb{ACT}_{w_m}(v)} = \mathbb{ACT}_{w_m} \cdot \mathbb{ACT}_v \cdot \mathbb{ACT}_{w_m}$. Тому права частина рівності (20) дорівнює наступному:

$$\mathbb{ACT}_{w_1} \cdot \ldots \cdot \mathbb{ACT}_{w_{m-1}} \cdot \mathbb{ACT}_{\mathbb{ACT}_{w_m}(v)} \cdot \mathbb{ACT}_{w_{m-1}} \cdot \ldots \cdot \mathbb{ACT}_{w_1} \tag{21}$$

Знову за лемою 2.2 для $\mathbb{ACT}_{w_m}(v), w_{m-1}$ маємо:

$$\mathbb{ACT}_{\mathbb{ACT}_{w_{m-1}}(\mathbb{ACT}_{w_m}(v))} = \mathbb{ACT}_{w_{m-1}} \cdot \mathbb{ACT}_{\mathbb{ACT}_{w_m}(v)} \cdot \mathbb{ACT}_{w_{m-1}}.$$

Тому (21) дорівнює :

$$\mathbb{ACT}_{w_1} \cdot \ldots \cdot \mathbb{ACT}_{(\mathbb{ACT}_{w_m} \cdot \mathbb{ACT}_{w_{m-1}})(v)} \cdot \ldots \cdot \mathbb{ACT}_{w_1} = \ldots =$$

$$= \mathbb{ACT}_{w_1} \cdot \mathbb{ACT}_{(\mathbb{ACT}_{w_m} \cdot \ldots \cdot \mathbb{ACT}_{w_2})(v)} \cdot \mathbb{ACT}_{w_1} = \mathbb{ACT}_{(\mathbb{ACT}_{w_m} \cdot \mathbb{ACT}_{w_{m-1}} \cdot \ldots \cdot \mathbb{ACT}_{w_1})(v)} = \mathbb{ACT}_{\mathbb{ACT}_{R(A)}(v)}$$

□

Лема 2.9. *Нехай $A \subset V(T_n)$ – довільна впорядкована за $<$ множина вершин дерева. Відображення $\mathbb{ACT}$ відносно множини, що є образом переміщення множини A відносно $R(A)$, еквівалентне відображенню $\mathbb{ACT}$ відносно множини $R(A)$.*

Тобто:

$$\mathbb{ACT}_{\mathbb{ACT}_{R(A)}(A)} = \mathbb{ACT}_{R(A)}$$

Доведення. Для $A = \{w_1, w_2, \ldots, w_m\} \subset V(T_n)$ та відповідно $R(A) = \{w_m, \ldots, w_2, w_1\}$ маємо:

$$\mathbb{ACT}_{\mathbb{ACT}_{R(A)}(A)} = \mathbb{ACT}_{\mathbb{ACT}_{R(A)}(w_1)} \cdot \mathbb{ACT}_{\mathbb{ACT}_{R(A)}(w_2)} \cdot \ldots \cdot \mathbb{ACT}_{\mathbb{ACT}_{R(A)}(w_m)} \qquad (22)$$

Зауважимо, що для довільних $u, v \in V(T_n)$ таких, що $v \leq u$ виконується:

$$\mathbb{ACT}_u(v) = v$$

Тому права частина рівності (22) дорівнює наступному:

$$(\mathbb{ACT}_{w_1}) \cdot (\mathbb{ACT}_{\mathbb{ACT}_{w_1}(w_2)}) \cdot (\mathbb{ACT}_{\mathbb{ACT}_{\{w_2,w_1\}}(w_3)}) \cdot \ldots \cdot (\mathbb{ACT}_{\mathbb{ACT}_{\{w_{m-2},\ldots,w_2,w_1\}}(w_{m-1})}) \cdot$$

$$\cdot (\mathbb{ACT}_{\mathbb{ACT}_{\{w_{m-1},\ldots,w_2,w_1\}}(w_m)}) \qquad (23)$$

Тоді за лемою 2.8 для кожного $\mathbb{ACT}$ у дужках у (23) отримаємо:

$$(\mathbb{ACT}_{w_1}) \cdot (\mathbb{ACT}_{w_1} \cdot \mathbb{ACT}_{w_2} \cdot \mathbb{ACT}_{w_1}) \cdot (\mathbb{ACT}_{w_1} \cdot \mathbb{ACT}_{w_2} \cdot \mathbb{ACT}_{w_3} \cdot \mathbb{ACT}_{w_2} \cdot \mathbb{ACT}_{w_1}) \cdot \ldots$$

$$\ldots \cdot (\mathbb{ACT}_{w_1} \cdot \mathbb{ACT}_{w_2} \cdot \ldots \cdot \mathbb{ACT}_{w_{m-2}} \cdot \mathbb{ACT}_{w_{m-1}} \cdot \mathbb{ACT}_{w_{m-2}} \cdot \ldots \cdot \mathbb{ACT}_{w_2} \cdot \mathbb{ACT}_{w_1}) \cdot$$

$$\cdot (\mathbb{ACT}_{w_1} \cdot \mathbb{ACT}_{w_2} \cdot \ldots \cdot \mathbb{ACT}_{w_{m-1}} \cdot \mathbb{ACT}_{w_m} \cdot \mathbb{ACT}_{w_{m-1}} \cdot \ldots \cdot \mathbb{ACT}_{w_2} \cdot \mathbb{ACT}_{w_1}) =$$

$$= (\mathbb{ACT}_{w_1} \cdot \mathbb{ACT}_{w_1}) \cdot (\mathbb{ACT}_{w_2} \cdot \mathbb{ACT}_{w_1} \cdot \mathbb{ACT}_{w_1} \cdot \mathbb{ACT}_{w_2}) \cdot (\mathbb{ACT}_{w_3} \cdot \mathbb{ACT}_{w_2} \cdot \mathbb{ACT}_{w_1} \cdot \ldots$$

$$\ldots \cdot \mathbb{ACT}_{w_1} \cdot \mathbb{ACT}_{w_2} \cdot \ldots \cdot \mathbb{ACT}_{w_{m-2}}) \cdot (\mathbb{ACT}_{w_{m-1}} \cdot \mathbb{ACT}_{w_{m-2}} \cdot \ldots \cdot \mathbb{ACT}_{w_2} \cdot \mathbb{ACT}_{w_1} \cdot$$

$$\cdot \mathbb{ACT}_{w_1} \cdot \mathbb{ACT}_{w_2} \cdot \ldots \cdot \mathbb{ACT}_{w_{m-1}}) \cdot \mathbb{ACT}_{w_m} \cdot \mathbb{ACT}_{w_{m-1}} \cdot \ldots \cdot \mathbb{ACT}_{w_2} \cdot \mathbb{ACT}_{w_1} \qquad (24)$$

За лемою 2.7 вирази в дужках у (24) є тотожніми. Тому (24) дорівнює наступному:

$$\mathbb{ACT}_{w_m} \cdot \mathbb{ACT}_{w_{m-1}} \cdot \ldots \cdot \mathbb{ACT}_{w_2} \cdot \mathbb{ACT}_{w_1} = \mathbb{ACT}_{R(A)}$$

□

2.4.2 Коректність алгоритму взяття оберненного

Теорема 2.5. *Алгоритм 3 знаходження оберненого дерева визначений коректно.*

Доведення. Нехай дерева $D, D^{-1} \in LT_{2,n}$, де D^{-1} отримано з D за допомогою алгоритма взяття оберненого.

Для довільної вершини $v \in V(T_n)$ позначимо:

$$UV_D(v) = \{w \in Path_{v_0}(v) | w \in OV(D)\} -$$

впорядкована за $<$ множина вершин з мітками 1 дерева D, що знаходяться на шляху від кореня v_0 до v. Тоді, віповідно до алгоритму 3 взяття оберненного маємо:

$$OV(D^{-1}) = \{\mathbb{ACT}_{R(UV_D(v))}(v) | v \in OV(D)\}$$

Зауважимо, що для довільної вершини $v \in V(T_n)$:

$$\mathbb{ACT}_{R(UV_D(v))}(v) = \mathbb{ACT}_{R(OV(D))}(v),$$

оскільки $UV_D(v) \subset OV(D)$ та для довільної $w \in OV(D) \backslash R(UV_D(v))$: $v \not\succ w$. А тому:

$$OV(D^{-1}) = \mathbb{ACT}_{R(OV(D))}(OV(D)) \tag{25}$$

Необхідно довести, що таким чином визначене дерево D^{-1} є оберненим до D відносно операції множення $*$:

$$D * D^{-1} = D^{-1} * D = e.$$

Останнє є еквівалентним наступному:

$$OV(D * D^{-1}) = OV(D^{-1} * D) = \emptyset$$

Для добутку $D * D^{-1}$ за теоремою 2.3 маємо:

$$OV(D * D^{-1}) = \mathbb{ACT}_{OV(D)}(OV(D^{-1})) \triangle OV(D) \tag{26}$$

Враховуючи визначення множини $OV(D^{-1})$ із (25) отримаємо для рівності (26) наступне:

$$OV(D*D^{-1}) = \mathbb{ACT}_{OV(D)}(\mathbb{ACT}_{R(OV(D))}(OV(D))) \triangle OV(D) = OV(D) \triangle OV(D) = \emptyset$$

Останнє виконується за лемою 2.7, де $A := OV(D)$.

Для добутку $D^{-1}*D$ за теоремою 2.3 та враховуючи визначення множини $OV(D^{-1})$ із (25) маємо:

$$OV(D^{-1} * D) = \mathbb{ACT}_{OV(D^{-1})}(OV(D)) \triangle OV(D^{-1}) =$$

$$= \mathbb{ACT}_{\mathbb{ACT}_{R(OV(D))}(OV(D))}(OV(D)) \triangle \mathbb{ACT}_{R(OV(D))}(OV(D))$$

Тоді за лемою 2.9, де $A := OV(D)$, останнє буде дорівнювати:

$$\mathbb{ACT}_{R(OV(D))}(OV(D)) \triangle \mathbb{ACT}_{R(OV(D))}(OV(D)) = \emptyset$$

□

2.4.3 Складність алгоритму взяття оберненного

Теорема 2.6. *Складність алгоритму взяття оберненного дерева із* $LT_{2,n}$ *становить* $O(n \cdot 2^n)$.

Доведення. Нехай D є деревом з $LT_{2,n}$. В алгоритмі 3 знаходження оберненого дерева ми маємо головний цикл по усіх вершинах, що мають мітки 1 у дереві D. Відповідно, максимальна кількість ітерацій у циклі буде досягатись за умови, якщо $OV(D)$ буде мати максимальну потужність, а саме: 2^n.

Для кожної вершини з $OV(D)$ обраховується її образ відносно послідовних переміщень вершин, що мають мітку 1 від неї до кореня — $R(Path_{v_0}(v))$. Зазначимо, що за умов максимальності потужності $OV(D)$ маємо, що усі вершини на шляху від заданої до кореня будуть мати мітку 1, а тому і кількість операцій для заданої вершини буде максимальною.

А тому, кількість операцій у алгоритмі для заданої вершини рівна сумі рівню вершини та один, де додаткова операція є додаванням образу переміщень до фінальної множини вершини з мітками $OV(D^{-1}) - k + 1$. Зазначимо, що кількість вершин на рівні k дорівнює 2^k.

Тоді, загальна кількість операцій алгоритму становить:

$$\sum_{k=0}^{n-1} 2^k \cdot (k+1) = n \cdot 2^n - 2^n + 1$$

Відповідно, загальна складність алгоритму дорівнює $O(n \cdot 2^n - 2^n + 1) = O(n \cdot 2^n)$. □

2.5 Відповідність між бінарними кореневими деревами та підстановками із $Syl_2(S_{2^n})$

2.5.1 Перетворення дерева із $LT_{2,n}$ у підстановку з $Syl_2S_{2^n}$

Розглянемо алгоритм знаходження підстановки з підгрупи $Syl_2S_{2^n}$, що відповідає дереву з $LT_{2,n}$.

Algorithm 4: Алгоритм перетворення дерева у підстановку

Input: $OC(D)$ – множина координат всіх вершин з мітками 1 дерева D.

Output: $(a_{i_1}, a_{i_2}, \cdots a_{i_{2^n}})$ нижня стірчка підстановки π.

```
1 (a_1, a_2, ··· a_{2^n}) = (1, 2, ··· , 2^n) ;
2 for (j, i) ∈ (OC(D), <) do
3     m := 2^{n-j-1} ( кількість елементів в одному блоці) ;
4     for l := 1 to m do
5         b := a_{(2i-2)m+l};
6         a_{(2i-2)m+l} := a_{(2i-1)m+l};
7         a_{(2i-1)m+l} := b;
```

На вхід маємо $OC(D)$ – множину координат вершин з мітками 1 дерева D. Ініціалізуємо блок $(a_1, a_2, \cdots a_{2^n}) = (1, 2, \cdots, 2^n)$. Далі для кожної вершини v з координатами $(i, j) \in (OC(D), <)$ переставляємо місцями відповідні підблоки (елементи вектора a). На вихід отримуємо змінений вектор $a = (\pi(1), \pi(2), \ldots, \pi(2^n))$ – нижній рядок підстановки π.

Приклад 2.6. *Розглянемо дерево $D \in LT_{2,3}$. (див. рис. 8).*

Ініціалізуємо стрічку: $(a_1, a_2, \ldots, a_8) = (1, 2, \ldots, 8)$. Зауважимо, що $n = 3$.

Виконуємо цикл по множині $OC(D) = \{(1, 1), (2, 2), (2, 3)\}$.

Ітерація 1: $(j, i) = (1, 1)$. Розглядаємо блоки довжини $m = 2^{n-j-1} = 2$ та міняємо місцями відповідні елементи $2i - 1 = 1$-ого та $2i = 2$-го блоків:

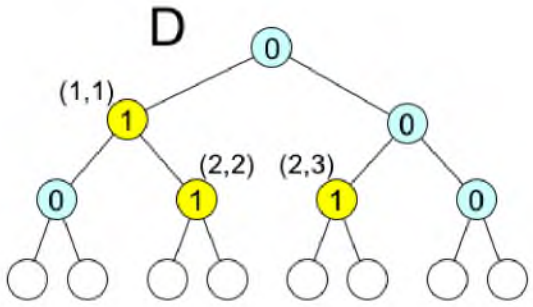


Рис. 8 3-рівневе дерево з мітками

$$a_{(2i-2)m+1} = a_1 = 1 \quad із \quad a_{(2i-1)m+1} = a_3 = 3$$
$$a_{(2i-2)m+2} = a_2 = 2 \quad із \quad a_{(2i-1)m+2} = a_4 = 4$$

Отримаємо $(a_1, a_2, \ldots, a_8) = (3, 4, 1, 2, 5, 6, 7, 8)$.

Ітерація 2: $(j, i) = (3, 2)$. *Розглядаємо блоки довжини* $m = 2^{n-j-1} = 1$ *та міняємо місцями відповідні елементи* $2i - 1 = 3$*-ого та* $2i = 4$*-го блоків:*

$$a_{(2i-2)m+1} = a_3 = 1 \quad із \quad a_{(2i-1)m+1} = a_4 = 2$$

Отримаємо $(a_1, a_2, \ldots, a_8) = (3, 4, 2, 1, 5, 6, 7, 8)$.

Ітерація 3: $(j, i) = (3, 3)$. *Розглядаємо блоки довжини* $m = 2^{n-j-1} = 1$ *та міняємо місцями відповідні елементи* $2i - 1 = 5$*-ого та* $2i = 6$*-го блоків:*

$$a_{(2i-2)m+1} = a_5 = 5 \quad із \quad a_{(2i-1)m+1} = a_6 = 6$$

(j, i)	m	Блоки	Номери блоків для обміну: $2i-1$ та $2i$		Результат ітерації (вигляд стрічки)
$(1, 1)$	2	| 1 2 | 3 4 | 5 6 | 7 8 |	1-ий	2-ій	$a = (3, 4, 1, 2, 5, 6, 7, 8)$
$(2, 2)$	1	| 3| 4| 1| 2| 5| 6| 7| 8|	3-й	4-ий	$a = (3, 4, 2, 1, 5, 6, 7, 8)$
$(2, 3)$	1	| 3| 4| 2| 1| 5| 6| 7| 8|	5-й	6-ий	$a = (3, 4, 2, 1, 6, 5, 7, 8)$

Табл. 1

Отримаємо $(a_1, a_2, \ldots, a_8) = (3, 4, 2, 1, 6, 5, 7, 8)$.

Звідки маємо підстановку: $\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 3 & 4 & 2 & 1 & 6 & 5 & 7 & 8 \end{pmatrix}$

Зауваження 2.2. *Нехай* $D \in LT_{2,n}$ *та* $\pi \in S_{2^n}$ *такі, що* π *є результатом дії алгоритму 4, в якому на вхід подається дерево* D. *Тоді для довільного* $1 \leq i \leq 2^n$ *в підстановці* $\pi(i)$ *виконується: і перейде в* $\pi(i)$ *тоді і ліше тоді, коли вершина з координатами* $(n, \pi(i))$ *під дією алгоритму 4 над* D *переходить у вершину з координатами* (n, i).

Теорема 2.7. *Алгоритм перетворення дерева у підстановку є коректним.*

Доведення. Зазначимо, що початкова стрічка $(1, 2, \ldots, 2^n)$ є 2-роздільною. Також зазначимо, що для довільної пари $(j, i) \in Coord(D)$ блоки $u_1 := (a_{(2i-2)m+1}, \ldots, a_{(2i-2)m+m})$ та $u_2 := (a_{(2i-1)m+1}, \ldots, a_{(2i-1)m+m})$ також є 2-роздільними. А також при перестановці їх значень місцями, блок $u = (a_{(2i-2)m+1}, \ldots, a_{(2i-2)m+m}, a_{(2i-1)m+1}, \ldots, a_{(2i-1)m+m})$ все ще залишається 2-роздільним.

Отже, кожне перетворення відносно пари координат $(j, i) \in OC(D)$ не змінює 2-роздільність стрічки a.

Таким чином отримана підстановка π, визначена стрічкою a, є 2-роздільною. $\square$

Теорема 2.8. *Складність алгоритму перевторення дерева з* $LT_{2,n}$ *у 2-роздільну підстановку з* S_{2^n} *становить* $O(n \cdot 2^n)$.

Доведення. Множина $OC(D)$ буде мати найбільшу потужність у випадку, коли кожна вершина v дерева з 0-го по $(n-1)$-ий рівні матиме мітку 1. Тоді для кожної v з координатами (j, i) буде потрібно виконати 2^{n-j-1} перестановок відповідних елементів вектора a.

Врахувавши, що на рівні j ми маємо 2^j вершин, причому $0 \leq j \leq (n-1)$,

отримаємо:

$$\sum_{j=0}^{n-1} 2^j \cdot 2^{n-j-1} = \sum_{j=0}^{n-1} 2^{n-1} = n \cdot 2^{n-1}$$

Отже, маємо складність алгоритму

$$O(n \cdot 2^{n-1}) = O(n \cdot 2^n).$$

□

2.5.2 Перетворення підстановки π із $Syl_2(S_{2^n})$ у дерево із $LT_{2,n}$

Розглянемо алгоритм, який для довільної підстановки $\pi \in Syl_2(S_{2^n})$ знаходить відповідне їй дерево $D \in LT_{2,n}$.

Algorithm 5: Алгоритм перетворення підстановки у дерево

Input: $(a_1, a_2, \ldots, a_{2^n})$ нижня стрічка 2-роздільної підстановки π,
$\pi \in Syl_2(S_{2^n})$.

Output: $OC(D)$.

```
1 OC(D) := ∅;
2 for j := 0 to n − 1 do
3     m := 2^{n−j−1} (довжина блоку);
4     for i := 1 to 2^j do
5         if a_{(2i−2)m+1} > a_{(2i−1)m+1} then
6             OC(D) := OC(D) ⋃ {(j, i)}
```

На вхід маємо нижню стрічку підстановки $a = (\pi(1), \pi(2), \ldots, \pi(2^n))$. В циклі алгоритму (кроки 2-4) ми попарно перевіряємо перші елементи (крок 5) блоків фіксованих довжин (крок 3), щоб визначити більші чи менші елементи знаходяться у відповідних блоках. Якщо умова кроку 5 виконується, то відповіднк пару координат додаємо в множину $OC(D)$ (крок 6). На вихід отримаємо множину координат $OC(D)$, що однозначно представляє дерево D.

Приклад 2.7. *Розглянемо 2-роздільну підстановку*

$$\pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 3 & 4 & 2 & 1 & 6 & 5 & 7 & 8 \end{pmatrix} \in S_8.$$

Тобто маємо вектор $(a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8) = (3, 4, 2, 1, 6, 5, 7, 8)$. *Ініціалізуємо множину* $OC(D) = \emptyset$. *Відповідно до алгоритму* 5 *маємо наступне (див. табл.* 2*):*

j	m	Блоки	i	$a_{(2i-2)m+1}$	$a_{(2i-1)m+1}$	Порівняння	$OC(D)$
0	4	3 4 2 1 | 6 5 7 8	1	$a_1 = 3$	$a_5 = 6$	$3 \not> 6$	$\emptyset$
1	2	3 4 | 2 1 | 6 5 | 7 8	1	$a_1 = 3$	$a_3 = 2$	$3 > 2$	$\{(1,1)\}$
			2	$a_5 = 6$	$a_7 = 7$	$5 \not> 7$	$\{(1,1)\}$
2	1	3| 4| 2| 1| 6| 5| 7| 8	1	$a_1 = 3$	$a_2 = 4$	$3 \not> 4$	$\{(1,1)\}$
			2	$a_3 = 2$	$a_4 = 1$	$2 > 1$	$\{(1,1),(2,2)\}$
			3	$a_5 = 6$	$a_6 = 5$	$6 > 5$	$\{(1,1),(2,2),(2,3)\}$
			4	$a_7 = 7$	$a_8 = 8$	$7 \not> 8$	$\{(1,1),(2,2),(2,3)\}$

Табл. 2

Отже, маємо наступний результат:

$OC(D) = \{(1,1),(2,2),(2,3)\}$.

Відповідно до цієї множини маємо дерево (див. рис. 9):

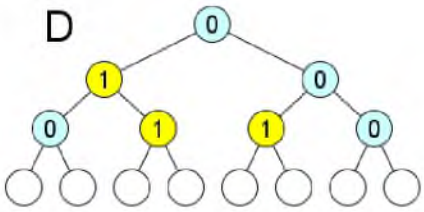


Рис. 9 Дерево D – результат алгоритму 5

Теорема 2.9. *Алгоритм* 5 *перетворення підстановки із* $Syl_2(S_{2^n})$ *у дерево з* $LT_{2,n}$ *є коректним.*

Доведення. Зазначимо, що:

1. координата j визначається циклом на кроці 2 з 0 до $n-1$, що відповідає рівням дерева з $LT_{2,n}$;

2. координата i визначається циклом на кроці 4 від 1 до 2^j, що відповідає номерам вершин рівня j для дерева з $LT_{2,n}$;

3. пара координат (j, i), що додається до множини $OC(D)$ є унікальною за рахунок проходження по циклам j та i відповідно до кроків 2 та 4.

Таким чином, множина $OC(D)$ є коректно визначеною множиною координат, що однозначно визначає дерево D з $LT_{2,n}$. □

Теорема 2.10. *Складність алгоритму 5 перетворення підстановки із $Syl_2(S_{2^n})$ у дерево з $LT_{2,n}$ дорівнює $O(2^n)$.*

Доведення. В алгоритмі маємо 2 вкладених цикли: перший по j (відповідає номеру рівня дерева) та другий по i (відповідає номеру вершини на рівні). Таким чином ми стільки разів звертаємося до вектора вектора a для порівняння величин його елементів, скільки вершин на дереві з 0-го по $(n-1)$-ий рівні. Отже, складність алгоритму дорівнює $O(2^n - 1) = O(2^n)$. □

Визначимо відображення:

- $\psi : LT_{2,n} \to Syl_2(S_{2^n})$ задається алгоритмом 4;
- $\tau : Syl_2(S_{2^n}) \to LT_{2,n}$, задається алгоритмом 5.

Теорема 2.11. *Відображення ψ та τ є взаємооберненими, що дає нам бієкцію між $LT_{2,n}$ та $Syl_2(S_{2^n})$.*

Доведення. Покажемо, що $\tau(\psi) = id$.

Нехай π – деяка 2-роздільна підстановка та $D = \tau(\pi)$ – відповідне їй дерево із $LT_{2,n}$, отримане за допомогою алгоритму 5. Тоді:

$$(j,i) \in OC(\tau(\pi)) \text{ тоді і тільки тоді, коли } a_{(2i-2)m+1} > a_{(2i-1)m+1} \text{ у підстановці } \pi. \tag{27}$$

З іншого боку, розглянемо довільне дерево $D \in LT_{2,n}$ та відповідну йому 2-роздільну підстановку $\pi = \psi(D)$, отриману за алгоритмом 4. Оскільки алгоритм 4 починає роботу із впорядкованої стрічки $(1, 2, \ldots, 2^n)$, то:

$$(j,i) \in OC(D) \text{ тоді і тільки тоді, коли } a_{(2i-2)m+1} > a_{(2i-1)m+1} \text{ в підстановці } \psi(D) \tag{28}$$

Тоді за (28) ми маємо:

$$(j,i) \in OC(D) \text{ тоді і тільки тоді, коли для } \psi(D): \ a_{(2i-2)m+1} > a_{(2i-1)m+1}$$

А останнє виконується, коли за (27) маємо: $(j,i) \in OC(\tau(\psi(D)))$.

Отже, $\tau(\psi) = id$.

Доведення для $\psi(\tau) = id$ проводиться аналогічно. □

Наслідок 2.2. *Нехай маємо дерево $D \in LT_{2,n}$, $\pi = \psi(D)$ — підстановка, яка обчислена за алгоритмом 4. Тоді для будь-яких $k, 1 \leq k \leq 2^n$, та вершини $v \in V(T_n)$ з координатами $c(v) = (n,s)$ маємо наступне:*

$$\pi(k) = s \text{ тоді і тільки тоді, коли } c(\mathbb{ACT}_{(OV(D),<)}(v)) = (n,k)$$

Доведення. Напряму випливає з алгоритму 4, визначення $\mathbb{ACT}$ та теореми 2.11. □

2.5.3 Теорема про ізоморфізм

Теорема 2.12. *Відображення ψ є ізоморфізмом між $LT_{2,n}$ та $Syl_2(S_{2^n})$.*

Доведення. Для початку зауважимо, що за теоремою 2.11 відображення ψ є бієкцією.

Потрібно показати, що відображення є гомеоморфізмом, тобто для довільних двох дерев $D_1, D_2 \in LT_{2,n}$ виконується:

$$\psi(D_1 * D_2) = \psi(D_1) \cdot \psi(D_2)$$

Тобто для підстановок $\pi_1 := \psi(D_1)$, $\pi_2 := \psi(D_2)$, $\pi := \psi(D_1 * D_2)$ та довільного числа $1 \leq i \leq 2^n$ покажемо, що:

$$\pi(i) = (\pi_1\pi_2)(i)$$

Виконання останньої рівності для кожного i буде еквівалентне наступній:

$$\pi^{-1}(i) = (\pi_1\pi_2)^{-1}(i) \tag{29}$$

Для довільної вершини w з координатами (n, s), $1 \leq s \leq 2^n$, та $1 \leq k \leq 2^n$ за наслідком 2.2 отримаємо:

$$\pi_1(k) = s \text{ тоді і тільки тоді, коли } c(\mathbb{ACT}_{(OV(D_1),<)}(w)) = (n, k) \tag{30}$$

$$\pi_2(k) = s \text{ тоді і тільки тоді, коли } c(\mathbb{ACT}_{(OV(D_2),<)}(w)) = (n, k) \tag{31}$$

$$\pi(k) = s \text{ тоді і тільки тоді, коли } c(\mathbb{ACT}_{(OV(D_1*D_2),<)}(w)) = (n, k) \tag{32}$$

Нехай вершина $v \in V(T_n)$ має координати $c(v) = (n, i)$. Тоді варто взяти:
$k := {\pi_1}^{-1}(i)$ в умові (30);
$k := {\pi_2}^{-1}(i)$ в умові (31);
$k := \pi^{-1}(i)$ в умові (32).

Отже, застосувавши (30)-(32), отримаємо, що рівність (29) буде еквівалентна наступному:

$$c(\mathbb{ACT}_{(OV(D_1*D_2),<)}(v)) = (n, \pi^{-1}(i)) = (n, (\pi_1\pi_2)^{-1}(i)) =$$

$$= (n, (\pi_2^{-1}\pi_1^{-1})(i)) = c((\mathbb{ACT}_{(OV(D_2),<)} \cdot \mathbb{ACT}_{(OV(D_1),<)})(v))$$

Таким чином, виконання рівності (29) для кожного такого значення i буде еквівалентне виконанню наступної рівності для кожної вершини n-го рівня $v \in V(T_n)$:

$$\mathbb{ACT}_{(OV(D_1 * D_2),<)}(v) = (\mathbb{ACT}_{(OV(D_2),<)} \cdot \mathbb{ACT}_{(OV(D_1),<)})(v), \tag{33}$$

Зауважимо, що остання рівність виконується для довільної вершини $v \in V(T_n)$ за наслідком 2.1. □

Наслідок 2.3. *Відображення τ є ізоморфізмом між $Syl_2(S_{2^n})$ та $LT_{2,n}$.*

Як випливає з алгоритмів 4 та 5, довільну підстановку $\pi \in Syl_2(S_{2^n})$ ми можемо зображати *відповідним* їй бінарним кореневим n-рівневим деревом $D = \tau(\pi) \in LT_{2,n}$ і навпаки.

Таким чином, з теореми про ізоморфізм випливає, що кожній підстановці відповідає деяке кореневе дерево і навпаки. Для довільної підстановки π відповідне кореневе дерево, отримано за допомгою алгоритму, будемо називати *відповідним* для підстановки π. А підстановку, отриману за алгоритмом будемо називати *відповідною підстановкою* для дерева.

Наслідок 2.4. *Множина кореневих n-рівневих дерев з мітками $LT_{2,n}$ та заданою операцією множення $*$ є групою.*

2.6 Висновки до розділу

У розділі 2 розглядається задача зображення підстановок із групи $Syl_2(S_{2^n})$ бінарними n-рівневими кореневими деревами з мітками з $LT_{2,n}$. Зокрема описано алгоритм множення двох таких дерев з часовою складністю $O(n \cdot 2^n)$ із використанням алгоритму switch, який має складність $O(2^{n-r})$. Введено алгоритм знаходження оберненого

дерева з часовою складністю $O(n \cdot 2^n)$. Також запропоновано два алгоритми: перетворення дерева із $LT_{2,n}$ у підстановку з $Syl_2(S_{2^n})$ з часовою складністю $O(n \cdot 2^n)$ та навпаки, перетворення підстановки із $Syl_2(S_{2^n})$ у бінарне n-рівневе дерево з $LT_{2,n}$ з часовою складністю $O(2^n)$. Коректність наведених алгоритмів ґрунтується на властивостях відображення $\mathbb{ACT}$, визначених у цьому розділу.

Доведено теорему про ізоморфізм між силовською 2-підгрупою $Syl_2(S_{2^n})$ симетричної групи S_{2^n} та множиною бінарних n-рівневих кореневих дерев з мітками $LT_{2,n}$, з якої випливає, що $LT_{2,n}$ з заданою операцією множення $*$ є групою.

Результати цього розділу було опубліковано у працях [44], [45], [47], [48], [49].

3 Відстань Хеммінга для підстановок із групи $Syl_2(S_{2^n})$

3.1 Кількість рухомих точок підстановки із $Syl_2(S_{2^n})$.

Означення 3.1. Кількість рухомих точок підстановки π – це кількість позицій, де елементи нижньої стрічки підстановки відрізняються від елементів верхньої стрічки [17]. Позначимо: $h(\pi)$.

Приклад 3.1. $\pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 2 & \boldsymbol{4} & \boldsymbol{3} & 5 & 6 & \boldsymbol{8} & \boldsymbol{7} \end{pmatrix}$

Кількість рухомих точок для цієї підстановки: $h(\pi) = 4$.

Кількість рухомих точок підстановки будемо рахувати, використовуючи ізоморфізм між силовською 2-підгрупою симетричної групи $Syl_2(S_{2^n})$ та групою бінарних кореневих дерев з мітками $LT_{2,n}$.

Нагадаємо, що 2^{n-j} – це кількість висячих вершин n-рівневого дерева, що знаходяться під вершиною з координатами (j, i).

Лема 3.1. *Нехай $\pi \in Syl_2(S_{2,n})$ – така підстановка, що відповідне їй дерево $D \in LT_{2,n}$ має єдину мітку 1 на вершині з координатами (j, i). Тоді кількість її рухомих точок буде рівна кількості висячих вершин піддерева з коренем на координатах (j, i). Тобто:*

$$h(\pi) = 2^{n-j}.$$

Доведення. Доведення леми 3.1 випливає безпосередньо із перевірки. □

Лема 3.2. *Нехай підстановка π має відповідне їй дерево D, в якого мітки 1 знаходяться на вершинах $v_1, v_2, \ldots, v_r$ з координатами $(j_1, i_1), (j_2, i_2) \ldots, (j_r, i_r)$. Причому жодна вершина v_k, $k \in \{1, 2, \ldots, r\}$, не лежить на шляху, що сполучає будь-яку іншу вершину цієї множин з коренем. Тоді кількість рухомих точок такої підста-*

новки буде рівна сумі висячих вершин всіх піддерев з коренями $v_1, v_2, \ldots, v_r$. *Тобто:*

$$h(\pi) = 2^{n-j_1} + 2^{n-j_2} + \ldots + 2^{n-j_r} = \sum_{k=1}^{r} 2^{n-j_k}.$$

Доведення. За умовою леми ми маємо, що під кожною вершиною v_k з міткою 1, $k \in \{1, \ldots, r\}$, знаходяться різні висячі вершини. А тому, за визначенням множення дерев за алгоритмом [?], дерево D розкладається у добуток:

$$D = D_1 \cdot D_2 \cdot \ldots \cdot D_r,$$

де D_k має єдину мітку 1 на вершині v_k з координатами (j_k, i_k), $k \in \{1, \ldots r\}$.

Тоді для кожного $k \in \{1, \ldots r\}$ за лемою 3.1:

$$h(\pi_k) = 2^{n-j_k},$$

де π_k відповідає дереву D_k. Оскільки підстановки $\pi_1, \ldots \pi_r$ діють на різні елементи, то:

$$h(\pi) = h(\pi_1) + h(\pi_2) + \ldots + h(\pi_r) =$$
$$= 2^{n-j_1} + 2^{n-j_2} + \ldots + 2^{n-j_r} = \sum_{k=1}^{r} 2^{n-j_k}$$

Лему доведено. □

Позначимо $L(v)$ – *множина всіх висячих вершин, що знаходяться під вершиною* v.

Лема 3.3. *Нехай підстановка* π *така, що у відповідному їй дереві* D *лише дві вершини* v_k *та* v_q *з координатами* (j_k, i_k) *та* (j_q, i_q) *мають мітки 1, причому* v_q *знаходиться під вершиною* v_k. *Тоді кількість рухомих точок такої підстановки буде рівна кількості висячих вершин піддерева з коренем* v_k. *Тобто для* $v_q \succ v_k$ *маємо:*

$$h(\pi) = 2^{n-j_k}.$$

Доведення. Оскільки $v_q \succ v_k$, то $j_q > j_k$ та мають місце такі співвідношення:

1. $|L(v_q)| = 2^{n-j_q}$ і це менше ніж $|L(v_k)| = 2^{n-j_k}$;

2. $L(v_q) \subset L(v_k)$.

Оскільки v_q знаходиться у лівій або правій гілці, для яких v_k є коренем, то і всі вершини під v_q будуть знаходитися у тій же самій гілці. А тому, підчас застосування алгоритму 4 перетворення дерева у підстановку, дія на координатах (j_q, i_q) не повертає висячі вершини у початкову гілку. Таким чином, за лемою 3.1, кількість рухомих точок підстановки π буде залежати від кількості висячих вершин з множини $L(v_k)$ та буде рівною її потужності:

$$h(\pi) = |L(v_k)| = 2^{n-j_k}.$$

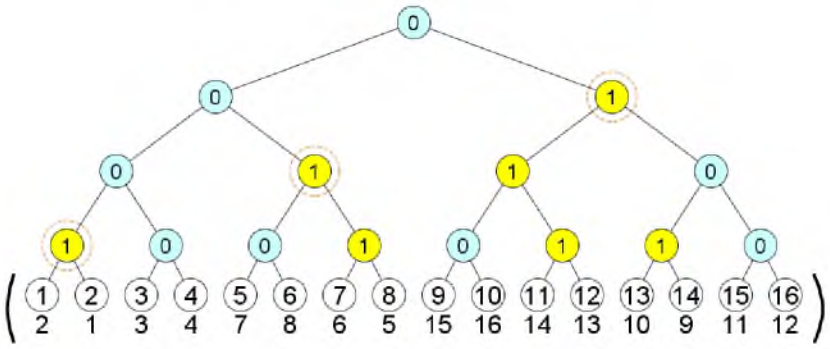
Лему доведено. □

Означення 3.2. Нехай вершина v дерева має мітку 1. Якщо на шляху між нею та коренем всі решта вершин мають мітки 0, то v будемо називати *головною вершиною.*

Наслідок 3.1. *Кількість рухомих точок підстановки π, що задається деревом D, рівна сумі кількостей висячих вершин, що знаходяться під головними вершинами дерева.*

Доведення. Доведення випливає із леми 3.2 та леми 3.3. □

Приклад 3.2. *Розглянемо підстановку $\pi \in Syl_2(S_{2^4})$ та відповідне їй дерево $D \in LT_{2,4}$:*



Тут головні вершини мають координати: $(1,2),(2,2),(3,1)$. *Відповідно до означення* 3.2 *та наслідку* 3.1, *маємо кількість рухомих точок:*

$$h(\pi) = 2^{4-1} + 2^{4-2} + 2^{4-3} = 8 + 4 + 2 = 14$$

Якщо виконати перевірку стандартного означення 3.1 *про кількість рухомих точок підстановки, то отримаємо таке ж значення.*

3.1.1 Алгоритм знаходження кількості рухомих точок підстановки із $Syl_2(S_{2^n})$

Введемо позначення:

- $a[k]$ – k-та координата стрічки a;
- $a[b,c]$ – стрічка, що має від b-ої до c-ої координат стрічки a, тобто є частиною стрічки a;
- $len(a)$ – функція, що визначає кількість координат стрічки a.

Наприклад, нехай маємо стрічку $a = (5,10,2,3,7,4)$, тоді:
$a[2] = 10$, $a[2,5] = (10,2,3,7)$, $len(a) = 6$.

Використовуючи алгоритм перетворення підстановки у дерево 5, задамо рекурсивний алгоритм для обчислення кількості рухомих точок підстановки $\pi \in Syl_2(S_{2^n})$.

Algorithm 6: Алгоритм знаходження кількості рухомих точок $h(\pi)$

Input: $a = (a[1], a[2], \cdots a[2^n])$ – нижня стрічка підстановки π.

Output: MovCount – кількість рухомих точок

1 *Створюємо рекурсивну підпрограму з аргументом a;*

2 **MovCount(a)**;

3 **if** $a[1] > a[\frac{len(a)}{2} + 1]$ **then**

4 $\quad$ **return** $len(a)$;

5 **if** $len(a) = 2$ **then**

6 $\quad$ **return** 0;

7 **return** $MovCount\Big(a\big[1\ , \frac{len(a)}{2}\big]\Big) + MovCount\Big(a\big[\frac{len(a)}{2} + 1\ , len(a)\big]\Big)$

Твердження 3.1. *Алгоритм 6 знаходження кількості рухомих точок підстановки* $\pi \in Syl_2(S_{2^n})$ *визначен коректно.*

Доведення. Умова на кроці 3 алгоритму буде досягатись лише у випадку, коли відповідна вершина ведповідного дерева D матиме мітку 1, відповідно до алгоритму 5 перетворення підстановки у дерево, крок 5. Кількість висячих вершин під головною вершиною буде співпадати довжині відповідної частини підстановки $len(a)$. Враховуючи вихід з рекурсивної підпрограми на кроці 4, знайдена вершина буде першою вершиною з міткою 1 у відповідному піддереві дерева D, а тому буде головною.

Якщо ж умова на кроці 3 не виконується, алгоритм переходить до лівої і правої частин 2-роздільної частини підстановки a $(a\big[1\ , \frac{len(a)}{2}\big],\ a\big[\frac{len(a)}{2} + 1\ , len(a)\big])$, що відповідає лівому і правому піддеревам відповідної вершини без мітки 1, на кроці 7.

Умова на кроці 5 виконується лише якщо наступні вершини відповідного дерева будуть висячими.

Таким чином алгоритм знаходження кількості рухомих точок 6 рекурсивним чином сумує кількість висячих вершин під головними вершинами відповідного дерева, що дорівнює кількості рухомих точок підстановки відповідно до наслідку 3.1. □

Твердження 3.2. *Часова складність алгоритму знаходження кількості рухомих точок підстановки* $\pi \in Syl_2(S_{2^n})$ *рівна* $O(2^n)$.

Доведення. Кожна пара елементів зі стрічки a на кроці порівняння відповідає певній вершині дерева із 0-го по $(n-1)$-ий рівні. У найгіршому випадку потрібно буде виконати стільки порівнянь, скільки таких вершин, а саме $2^n - 1$. Тому маємо оцінку

$$O(2^n - 1) = O(2^n).$$

Твердження доведено. □

Дана оцінка є оцінкою зверху та показує кількість кроків у найгіршому випадку. Таких випадків значно менше ніж решти, завдяки властивості 2-роздільності підстановок.

Створено програму, яка для алгоритму 6 рахує середнє значення кількості порівнянь для кожного n. Цю програму вдалося виконати та перевірити для невеликих $n = 2, 3, 4, 5$ за допомогою мови комп'ютерної алгебри Sage.

n	2^n довжина підстановки	Потужність $Syl_2(S_{2^n})$	Кількість порівнянь за означенням	Середня кількість порівнянь алгоритму
2	4	$\lvert Syl_2(S_4)\rvert = 2^3 = 8$	4	2
3	8	$\lvert Syl_2(S_8)\rvert = 2^7 = 128$	8	3
4	16	$\lvert Syl_2(S_{16})\rvert = 2^{15} = 32768$	16	4
5	32	$\lvert Syl_2(S_{32})\rvert = 2^{31}$	32	5

Можна побачити таку залежність: середнє значення кількості порівнянь дорівнює n для відповідного значення n. Тоді для загального випадку маємо наступну теорему.

Теорема 3.1. *Для знаходження кількості рухомих точок підстановки* $\pi \in Syl_2(S_{2^n})$ *за алгоритмом 6 в середньому необхідно виконати* n *порівнянь.*

Доведення. Доведення будемо виконувати за Методом математичниї індукції по n.

База індукції Результати роботи алгоритму, наведенні у таблиці вище для $n = 2, 3, 4, 5$.

Припущення індукції. Нехай для n умова теореми виконується. Тобто для множини $Syl_2(S_{2^n})$, що містить підстановки довжини 2^n (позначимо π) та кожна з яких задається відповідним n-рівневим деревом (позначимо D), в середньому для алгоритму 6 необхідно n порівнянь. Тобто:

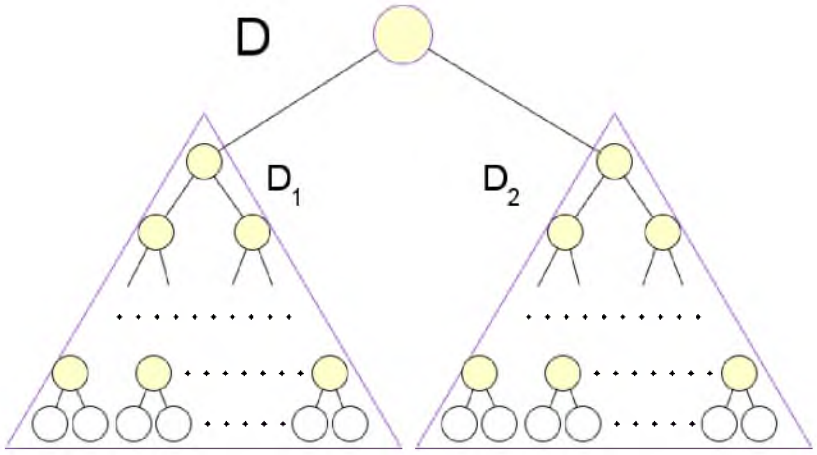
$$\frac{\Sigma}{2^{2^n-1}} = n, \text{ де } \Sigma = 1 \cdot N_1 + 2 \cdot N_2 + \ldots + (2^n - 1) \cdot N_{2^n-1},$$

N_k – це кількість підстановок для яких алгоритм 6 виконується за k кроків (порівнянь).

Тобто маємо:

$$n \cdot 2^{2^n-1} = 1 \cdot N_1 + 2 \cdot N_2 + \ldots + (2^n - 1) \cdot N_{2^n-1} \tag{34}$$

Індуктивний крок. Перевіримо для $n + 1$.



Зауважимо, що дерево D має $n + 1$ рівні та його корінь сполучає дві головні гілки, які є n-рівневими кореневими деревами D_1 та D_2 відповідно. Розглянемо випадки:

1. Нехай корінь дерева D має мітку 1. Тоді для знаходження кількості рухомих точок за алгоритмом 6 буде виконано лише 1 крок, бо на корені ми і зупинимося.

Таких дерев із множини $LT_{2,n+1}$, що мають мітку 1 на корені, всього $2^{2^{n+1}-2}$. Тобто для знаходження середньої оцінки будемо використовувати:

$$1 \cdot 2^{2^{n+1}-2} \tag{35}$$

2. Нехай корінь дерева D має мітку 0. Тоді алгоритм 6 розгляне корінь (1 дія) та перейде на рівень нижче, до лівого та правого піддерев. Тобто далі маємо таку таблицю:

Корінь дерева D (к-ть)	Кількість вершин лівого піддерева, D_1	Кількість вершин правого піддерева, D_2	Результат для обчислень
1	1	1	$(1+1+1)N_1N_1$
1	1	2	$(1+1+2)N_1N_2$
.....	.....	.....	.....
1	1	2^n-1	$(1+1+(2^n-1))N_1N_{2^n-1}$
1	2	1	$(1+2+1)N_2N_1$
1	2	2	$(1+2+2)N_2N_2$
.....	.....	.....	.....
1	2	2^n-1	$(1+2+(2^n-1))N_2N_{2^n-1}$
.....	.....	.....	.....
1	2^n-1	1	$(1+(2^n-1)+1)N_{2^n-1}N_1$
1	2^n-1	2	$(1+(2^n-1)+2)N_{2^n-1}N_2$
.....	.....	.....	.....
1	2^n-1	2^n-1	$(1+(2^n-1)+$ $+(2^n-1))N_{2^n-1}N_{2^n-1}$

Тому загальна сума для $n+1$ в цьому випадку буде:

$$(1+1+1)N_1N_1+(1+1+2)N_1N_2+\ldots+(1+1+(2^n-1))N_1N_{2^n-1}+$$

$$+(1+2+1)N_2N_1+(1+2+2)N_2N_2+\ldots+(1+2+(2^n-1))N_2N_{2^n-1}+\ldots$$

$$\ldots+(1+(2^n-1)+1)N_{2^n-1}N_1+(1+(2^n-1)+2)N_{2^n-1}N_2+\ldots+(1+(2^n-1)+$$

$$+(2^n-1))N_{2^n-1}N_{2^n-1}$$

З кожної стрічки винесемо за дужки N_1, N_2, … та N_{2^n-1} відповідно:

$$N_1\Big((1+1+1)N_1+(1+1+2)N_2+\ldots+(1+1+(2^n-1))N_{2^n-1}\Big)+$$

$$+N_2\Big((1+2+1)N_1+(1+2+2)N_2+\ldots+(1+2+(2^n-1))N_{2^n-1}\Big)+\ldots$$

$$+N_{2^n-1}\Big((1+(2^n-1)+1)N_1+(1+(2^n-1)+2)N_2+\ldots+(1+(2^n-1)+(2^n-1))N_{2^n-1}\Big)=$$

$$=N_1\Big(2(N_1+N_2+\ldots+N_{2^n-1})+(1N_1+2N_2+\ldots+(2^n-1)N_{2^n-1})\Big)+$$

$$+N_2\Big(3(N_1+N_2+\ldots+N_{2^n-1})+(1N_1+2N_2+\ldots+(2^n-1)N_{2^n-1})\Big)+\ldots$$

$$\ldots+N_{2^n-1}\Big(2^n(N_1+N_2+\ldots+N_{2^n-1})+(1N_1+2N_2+\ldots+(2^n-1)N_{2^n-1})\Big)$$

Зауважимо, що $N_1+N_2+\ldots+N_{2^n-1}$ – це вся кількість підстановок довжини 2^n з групи $Syl_2(S_{2^n})$, тобто $N_1+N_2+\ldots+N_{2^n-1}=|Syl_2(S_{2^n})|=2^{2^n-1}$.

Також застосуємо рівність (34):

$$N_1\Big(2\cdot 2^{2^n-1}+n\cdot 2^{2^n-1}\Big)+N_2\Big(3\cdot 2^{2^n-1}+n\cdot 2^{2^n-1}\Big)+\ldots+N_{2^n-1}\Big(2^n\cdot 2^{2^n-1}+n\cdot 2^{2^n-1}\Big)=$$

$$=2^{2^n-1}\Big(2N_1+nN_1+3N_2+nN_2+\ldots+2^nN_{2^n-1}+nN_{2^n-1}\Big)=$$

$$=2^{2^n-1}\Big(n(N_1+N_2+\ldots+N_{2^n-1})+(1N_1+2N_2+\ldots+(2^n-1)N_{2^n-1})+$$

$$+(N_1+N_2+\ldots+N_{2^n-1})\Big)=2^{2^n-1}\Big((n+1)\cdot 2^{2^n-1}+n\cdot 2^{2^n-1}\Big)=$$

$$=2^{2^{n+1}-2}\Big(2n+1\Big) \tag{36}$$

За результатами обох випадків (35) та (36) маємо середню оцінку для $n+1$:

$$\frac{1\cdot 2^{2^{n+1}-2}+2^{2^{n+1}-2}\cdot(2n+1)}{2^{2^{n+1}-1}}=\frac{2^{2^{n+1}-2}\cdot(2n+2)}{2^{2^{n+1}-1}}=\frac{2n+2}{2}=n+1.$$

Отже, теорему доведено.

□

3.2 Кількість підстановок із $Syl_2(S_{2^n})$ що мають мінімальну ненульову або максимальну кількість рухомих точок

3.2.1 Мінімальна ненульова кількість рухомих точок

Оскільки ми розглядаємо підстановки $\pi \in Syl_2(S_{2^n})$, то для них зберігається властивість 2-роздільності. А тому найменшою ненульовою кількість рухомих точок для таких підстановок є $h(\pi) = 2$.

Така рівність може виконуватися лише для підстановок, які задаються таким деревом $D \in LT_{2,n}$, що має лише одну вершину з міткою 1 і причому на $(n-1)$-му рівні (випливає із лем 3.1, 3.2 та 3.3). Тобто кількість таких підстановок буде рівна кількості таких дерев. А їх відповідно буде стільки, скільки різних вершин на $(n-1)$-рівні, тобто:

$$\left|\{\pi \mid h(\pi) = 2 \text{ та } \pi \in Syl_2(S_{2^n})\}\right| = 2^{n-1}$$

3.2.2 Максимальна кількість рухомих точок

Максимальною кількістю рухомих точок підстановки $\pi \in Syl_2(S_{2^n})$ є число, яке рівне довжині самої підстановки. Тобто $h(\pi) = 2^n$.

Теорема 3.2. *Кількість підстановок $\pi \in Syl_2(S_{2^n})$, які мають 2^n рухомих точок дорівнює $f(n)$, що визначається рекурсивно наступним чином:*

$$f(n) = \begin{cases} 1, \text{ коли } n = 1 \\ 2^{2^n-2} + f(n-1) \cdot f(n-1) \end{cases}$$

Доведення. Доведення будемо виконувати за Методом математичної індукції.

База індукції: $n = 1$. Для $Syl_2(S_{2^1})$ єдиною не тотожною підстановкою є транспозиція $(1, 2)$. А тому $f(1) = 1$ – визначено коректно.

Індуктивний крок. Нехай для індексів менших за n умова теореми виконується. Перевіримо для n. Нехай $D \in LT_{2,n}$ та розглянемо наступні випадки.

1. Нехай дерево D на корені має мітку 1. Тоді за лемою 3.3: $h(\pi) = 2^n$, де $\pi \in Syl_2(S_{2^n})$ відповідає дереву D.

   За означенням дерев із множини $LT_{2,n}$, кількість таких різних дерев становить 2^{2^n-2}.

2. Нехай дерево D на корені має мітку 0 (тобто не має мітку 1). Також нехай $D_1, D_2 \in LT_{2,n-1}$ – його ліве та праве піддерево відповідно. Тоді за наслідком 3.1:

   $$h(\pi) = 2^n \text{ тоді і тільки тоді, коли } h(\pi_1) = h(\pi_2) = 2^{n-1}.$$

   За припущенням індукції, кількість різних підстановок із $Syl_2(S_{2^{n-1}})$ (та відповідно і їх дерев із $LT_{2,n-1}$), що мають 2^{n-1} рухомих точок, становить $f(n-1)$.

   Тому кількість таких дерев $D \in LT_{2,n}$ рівна $f(n-1) \cdot f(n-1)$.

Оскільки випадки 1. та 2. неперетинні, то:

$$f(n) = 2^{2^n-2} + f(n-1) \cdot f(n-1).$$

Отже, теорему доведено. □

3.3 Відстань Хеммінга між елементами групи $Syl_2(S_{2^n})$

Нехай $D_1, D_2 \in LT_{2,n}$. Тоді:

- позначимо $D_1 \Delta D_2$ – таке дерево з $LT_{2,n}$, що визначається множиною вершин з мітками 1 $OV(D_1) \Delta OV(D_2)$;

- символом $\mathbb{ACT}_{D_1}(D_2)$ позначимо множину $\mathbb{ACT}_{(OV(D_1),<)}(OV(D_2))$ – результат відображення, що переміщає множину вершин з мітками 1 дерева D_2 відносно впорядкованої множини вершин з мітками 1 дерева D_1

Теорема 3.3. *Нехай π_1, π_2 – підстановки із групи $Syl_2(S_{2^n})$ та $D_1, D_2 \in LT_{2,n}$ – відповідні їм дерева. Тоді відстань Хеммінга між цими підстановками рівносильна відстані Хеммінга між нейтральним елементом e групи $Syl_2(S_{2^n})$ та підстановкою, відповідне дерево якої є результатом симетричної різниці дерев D_1 та D_2. Тобто:*

$$d_H(\pi_1, \pi_2) = d_H\Big(e, \psi(D_1 \Delta D_2)\Big), \tag{37}$$

Доведення. Застосуємо Метод математичної індукції по n.

База індукції. Для випадку $n = 1$ твердження теореми виконується для групи $Syl_2(S_2)$ прямим перебором елементів.

Припущення індукції. Припустимо, що для n умова теореми виконується.

Індуктивний крок. Перевіримо, чи виконується умова для $n + 1$.

Розглянемо випадки відносно корення v_0 для дерева $D_1 \Delta D_2$.

1. Нехай v_0 – корінь дерева $D_1 \Delta D_2$ та він має мітку 1, тобто

$$v_0 \in OV(D_1 \Delta D_2). \tag{38}$$

Непорушуючи загальності, вважаємо, що $v_0 \in OV(D_1)$ та $v_0 \notin OV(D_2)$.

За алгоритмом 5 перетворення підстановки у дерево маємо наступне:

$$\pi_1(1) > \pi_1(2^n + 1) \text{ та } \pi_2(1) < \pi_2(2^n + 1).$$

Оскільки $\pi_1, \pi_2 \in Syl_2(S_{2^{n+1}})$, то вони 2-роздільні, а тому:

$$\pi_1\Big(\{1, \ldots, 2^n\}\Big) = \{2^n + 1, \ldots 2^{n+1}\},\ \pi_1\Big(\{2^n + 1, \ldots, 2^{n+1}\}\Big) = \{1, \ldots 2^n\},$$

$$\pi_2\Big(\{1, \ldots, 2^n\}\Big) = \{1, \ldots, 2^n\},\ \pi_2\Big(\{2^n + 1, \ldots, 2^{n+1}\}\Big) = \{2^n + 1, \ldots, 2^{n+1}\},$$

Отже, ми маємо:

$$d_H(\pi_1, \pi_2) = 2^{n+1}. \tag{39}$$

З іншого боку, за умовою (38) та за алгоритмом 4 перетворення дерева у підстановку, маємо:

$$\pi(1) > \pi(2^n + 1) \text{ для } \pi = \psi(D_1 \Delta D_2).$$

Оскільки $\pi \in Syl_2(S_{2^{n+1}})$, то вона 2-роздільні, а тому:

$$\pi\Big(\{1, \dots, 2^n\}\Big) = \{2^n + 1, \dots 2^{n+1}\} \text{ та } \pi\Big(\{2^n + 1, \dots, 2^{n+1}\}\Big) = \{1, \dots 2^n\}.$$

Отже,

$$d_H(e, \pi) = 2^{n+1}. \tag{40}$$

Таким чином за рівностями (39) та (40) маємо (37).

2. Нехай v_0 – корінь дерева $D_1 \Delta D_2$ і він має мітку 0, тобто

$$v_0 \notin OV(D_1 \Delta D_2). \tag{41}$$

Тоді $v_0 \notin OV(D_1)$ і $v_0 \notin OV(D_2)$ або $v_0 \in OV(D_1)$ і $v_0 \in OV(D_2)$.

Розглянемо випадок $v_0 \notin OV(D_1)$ і $v_0 \notin OV(D_2)$. Нехай $\pi_1, \pi_2 \in Syl_2(S_{2^{n+1}})$ – відповідні підстановки дерев D_1, D_2.

Оскільки обидва дерева на корені мають мітку 0, то образом лівого піддререва буде ліве піддерево, а правого – праве. Тобто за алгоритмом 5 перетворення підстановки у дерево мають місце наступні нерівності:

$$\pi_1(1) < \pi_1(2^n + 1) \text{ та } \pi_2(1) < \pi_2(2^n + 1).$$

Оскільки $\pi_1, \pi_2 \in Syl_2(S_{2^{n+1}})$, то вони 2-роздільні, а тому:

$$\pi_1\Big(\{1, \dots, 2^n\}\Big) = \{1, \dots, 2^n\},\ \pi_1\Big(\{2^n + 1, \dots, 2^{n+1}\}\Big) = \{2^n + 1, \dots, 2^{n+1}\},$$

$$\pi_2\Big(\{1, \dots, 2^n\}\Big) = \{1, \dots, 2^n\},\ \pi_2\Big(\{2^n + 1, \dots, 2^{n+1}\}\Big) = \{2^n + 1, \dots, 2^{n+1}\}.$$

Зауважимо, що частина підстановки π, що відповідає лівому піддереву, визначена на елементах від 1 до 2^n. А частина підстановки π, що відповідає правому піддереву – на елементах від $2^n + 1$ до 2^{n+1}. Тому звуження такої підстановки $\pi \in Syl_2(S_{2^{n+1}})$ можна розбити на дві підстановки $\pi_1, \pi_2 \in Syl_2(S_{2^n})$ (див. Рис. 10):

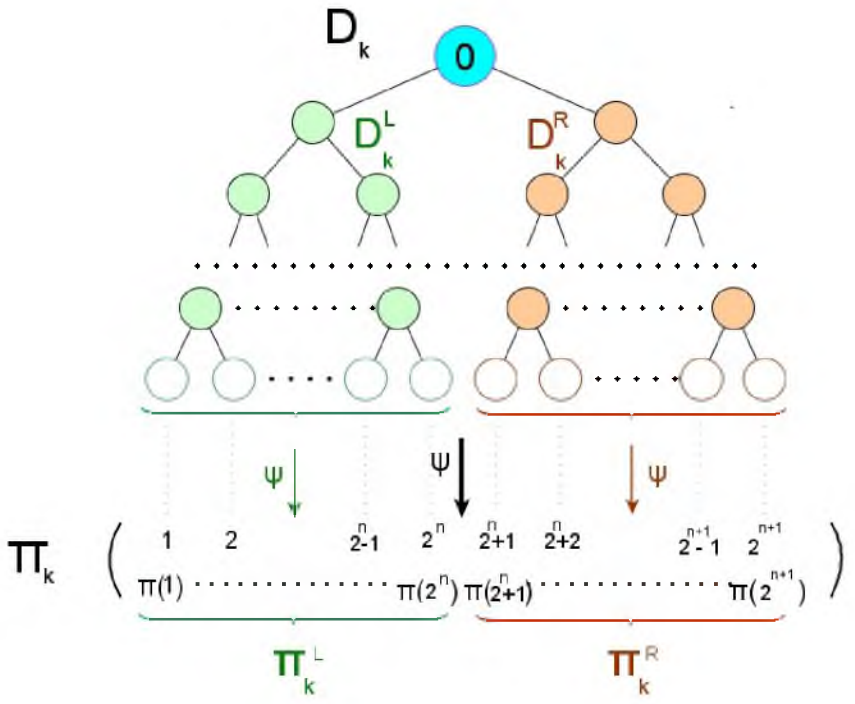


Рис. 10 Вигляд дерев та звуження підстановок.

Позначимо: $\pi_k^L := \pi_k\big|_{\{1,\dots,2^n\}}$, $\pi_k^R := \pi_k\big|_{\{2^n+1,\dots,2^{n+1}\}}$, $k = 1, 2$

Тоді:

$$d_H(\pi_1, \pi_2) = d_H(\pi_1^L, \pi_2^L) + d_H(\pi_1^R, \pi_2^R). \tag{42}$$

Дані підстановки π_k^L та $\pi_k^R R$, $k = 1, 2$, мають довжину 2^n та визначені n-рівневими деревами з мітками (лівим та правим піддеревами дерев D_1 та D_2 відповідно), тобто для них виконується припущення індукції. Тоді маємо:

$$d_H(\pi_1^L, \pi_2^L) = d_H(e^L, \psi(D_1^L \Delta D_2^L); \tag{43}$$

$$d_H(\pi_1^R, \pi_2^R) = d_H(e^R, \psi(D_1^R \Delta D_2^R)); \tag{44}$$

Із рівностей (42), (43) та (44) отримаємо:

$$d_H(\pi_1, \pi_2) = d_H(e^L, \psi(D_1^L \Delta D_2^L) + d_H(e^R, \psi(D_1^R \Delta D_2^R)). \tag{45}$$

Оскільки підстановки π_k^L та π_k^R визначенні на множинах, що не перетинаються, то:

$$d_H(\pi_1, \pi_2) = d_H(e^L, \psi(D_1^L \Delta D_2^L)) + d_H(e^R, \psi(D_1^R \Delta D_2^R)) = d_H(e, \psi(D_1 \Delta D_2)). \tag{46}$$

Зауважимо, що для випадку $v_0 \in OV(D_1)$ та $v_0 \in OV(D_2)$ міркування аналогічні.

Теорему доведено. □

Твердження 3.3. *Нехай $\pi \in Syl_2(S_{2^n})$, e — нейтральний елемент групи $Syl_2(S_{2^n})$. Тоді відстань Хеммінга між підстановкою та нейтральним елементом можна визначити як кількість рухомих точок цієї підстановки. Тобто:*

$$d_H(e, \pi) = h(\pi).$$

Доведення. Прямо випливає з визначень відстаней d_H та h. □

Враховуючи твердження 3.3, рівність (37) теореми 3.3 можна переписати наступним чином:

$$d_H(\pi_1, \pi_2) = h\big(\psi(D_1 \Delta D_2)\big). \tag{47}$$

Приклад 3.3. *Розглянемо дві підстановки $\pi_1, \pi_2 \in Syl_2(S_{2^4})$ та відповідні їм дерева $D_1, D_2 \in LT_{2,4}$ (див. Рис. 11):*

Після чого отримаємо дерево $D = D_1 \Delta D_2$ та відповідну йому підстановку $\pi = \psi(D_1 \Delta D_2)$ (див. Рис. 12):

Тоді за рівністю (47), $d_H(\pi_1, \pi_2) = d_H(e, \pi) = h(\pi) = 14$.

Окрім того, за формулою у визначенні відстані Хеммінга (5) також маємо: $d_H(\pi_1, \pi_2) = 14$.

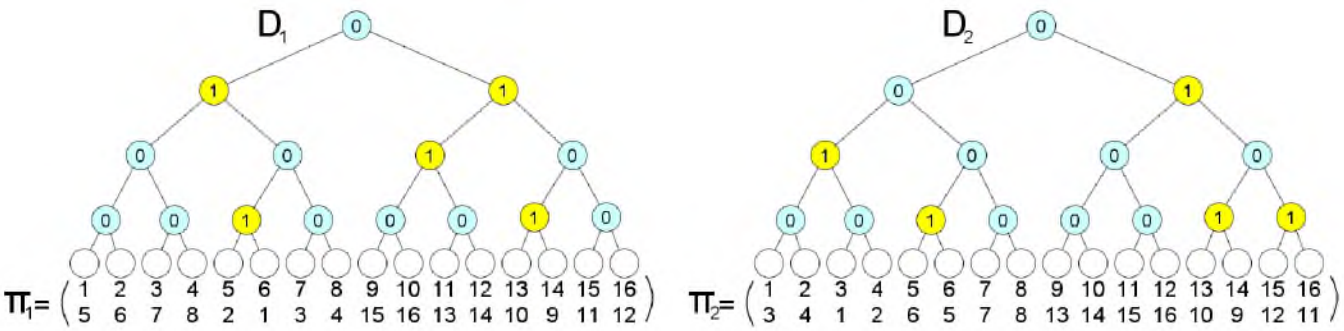


Рис. 11 Два дерева із $LT_{2,4}$

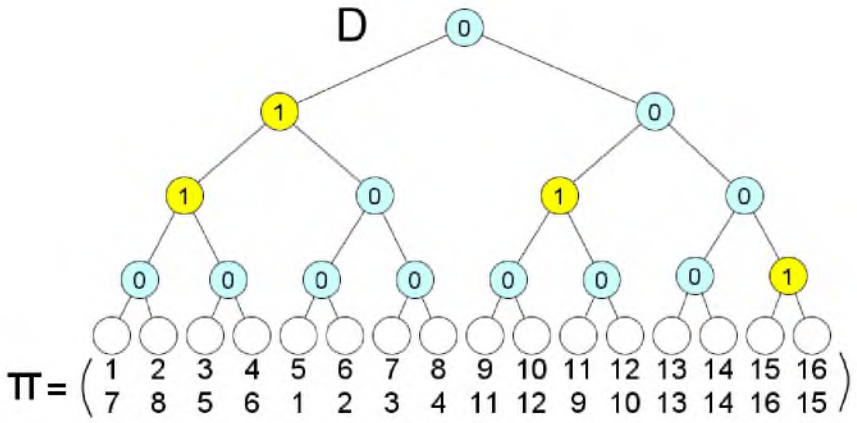


Рис. 12 Симетрична різниця дерев D_1 та D_2.

3.3.1 Алгоритм знаходження відстані Хеммінга

Нехай $\pi_1, \pi_2 \in Syl_2(S_{2^n})$ мають відповідні деревами $D_1, D_2 \in LT_{2,n}$.

Позначимо:

- $a[k]$ – k-та координата стрічки a;
- $a[b, c]$ – стрічка, що має від b-ої до c-ої координат стрічки a, тобто є частиною стрічки a;
- $len(a)$ – функція, що визначає кількість координат стрічки a.

Algorithm 7: Алгоритм знаходження відстані Хеммінга $d_H(\pi_1, \pi_2)$

Input: $a = (a[1], \cdots a[2^n]), b = (b[1], \cdots b[2^n])$ – нижні стрічки підстановок π_1 та π_2 відповідно.

Output: Відстань Хеммінга між підстановками π_1 та π_2

1 *Створюємо рекурсивну підпрограму з аргументами* a, b;

2 **Hem(a,b)**;

3 перевіряємо чи є різними мітки на вершинах з однаковими координатами для відповідних дерев:

4 **if** $\Big(a[1] > a[\frac{len(a)}{2} + 1]$ *xor* $b[1] > b[\frac{len(b)}{2} + 1]\Big)$ **then**

5 $\quad$ **return** $len(a)$;

6 **if** $len(a) = 2$ **then**

7 $\quad$ **return** 0;

8 **return** $Hem\Big(a[1, \frac{len(a)}{2}]; b[1, \frac{len(b)}{2}]\Big) + Hem\Big(a\big[\frac{len(a)}{2} + 1\ , len(a)\big]; b\big[\frac{len(b)}{2} + 1\ , len(b)\big]\Big)$

9 Hem(a,b)

Теорема 3.4. *Алгоритм знаходження відстані Хеммінга визначено коректно.*

Доведення. Зазначимо, що:

1. крок 4
$$\Big(a[1] > a[2^{n-1} + 1] \text{ xor } b[1] > b[2^{n-1} + 1]\Big) \tag{48}$$
буде виконуватися лише у тому випадку, коли відповідні вершини з однаковими координатами будуть мати різні мітки у деревах D_1, D_2. Тобто, вершина з тими ж координатами дерева $D_1 \Delta D_2$ буде мати мітку 1;

2. крок 6, $len(a) = 2$, буде досягатися лише у випадку, коли сусідні 2 образи підстановок будуть співпадати;

3. перехід рекурсії (крок 8) аналогічний переходу від вершини, до її нащадків у деревах D_1, D_2.

Таким чином, умова (48) буде виконуватися в алгоритмі лише на головних вершинах дерева $D_1 \Delta D_2$. Відповідно до визначення $h(\pi)$ у наслідку 3.1, кількість рухомих точок підстановки дорівнює сумі кількостей висячих вершин під головними. В свою чергу, кількість висячих вершин під головною визначається як довжина $len(a)$ відповідної стрічки набору a у рекурсивній підпрограмі.

Отже, алгоритм обраховує відстань $h(\psi(D_1 \Delta D_2))$. Тоді за рівністю (47) маємо:

$$h(\psi(D_1 \Delta D_2)) = d_H(\pi_1, \pi_2).$$

□

Твердження 3.4. *Складність алгоритму 7 знаходження відстані Хеммінга між підстановками $\pi_1, \pi_2 \in Syl_2(S_{2^n})$ рівна $O(2^n)$.*

Доведення. Враховуючи рекурсивність даного алгоритму, найбільша кількість операцій буде зроблена у випадку, коли буде виконана найбільша кількість звернень до цього ж алгоритму. А це буде досягатися у випадку, коли два дерева D_1 та D_2, що відповідають заданим підстановкам, матимуть однакові мітки на вершинах із однаковими координатами з 0-го по $(n-1)$-ий рівні. У такому випадку всі вершини дерева $D_1 \Delta D_2$ матимуть мітки 0. Для алгоритму це означає, що для кожної пари ми виконуємо по 2 порівняння на кроці 4. І таких пар буде 2^{n-1}. Отже, $O(2 * 2^{n-1}) = O(2^n)$. □

Теорема 3.5. *Середня складність знаходження відстані Хеммінга між підстановками $\pi_1, \pi_2 \in Syl_2(S_{2^n})$ за алгоритмом 7 становить $O(n)$.*

Доведення. Оскільки алгоритм 7 рекурсивний, то для кожної пари підстановок із $\pi_1, \pi_2 \in Syl_2(S_{2^n})$ він буде виконувати певну кількість викликів підпрограми Hem, яку ми позначимо за $N(\pi_1, \pi_2)$. Нехай $\Sigma(n) = \sum_{\pi_1, \pi_2 \in Syl_2(S_{2^n})} N(\pi_1, \pi_2)$ – загальна сума

цих кроків по всім можливим входам пар підстановок із $Syl_2(S_{2^n})$.

Позначимо $K(n)$ – кількість таких викликів в середньому для всіх пар підстановок з групи $Syl_2(S_{2^n})$. Тобто:

$$K(n) = \frac{\Sigma(n)}{\left|Syl_2(S_{2^n})\right|^2}, \text{ причому } \left|Syl_2(S_{2^n})\right| = 2^{2^n-1}.$$

Покажимо, що:

$$\Sigma(n) = n \cdot (2^{2^n-1})^2 \text{ та } K(n) = n \tag{49}$$

методом математичної індукції по n.

База індукції: $n = 1$. Група $Syl_2(S_{2^1})$ має потужність 2, а підстановки цієї групи визначаються міткою лише на корені дерев із $LT_{2,1}$. Всього різних пар підстановок, які можемо розглянути для алгоритму 7, буде 4, а тому $\Sigma(1) = 4$, $K(1) = 1$.

Припущення індукції. Нехай виконується рівність (49) для n.

Крок індукції. Перевіримо випадок для $n + 1$. Розгянемо два різні випадки залежно від розміщення міток на коренях $(n + 1)$-рівневих дерев, що відповідають підстановкам із $Syl_2(S_{2^{n+1}})$.

- *Випадок 1.* Нехай відповідні дерева мають різні мітки на коренях: 0 і 1 або 1 і 0. Таких варіантів буде 2. Тоді виконання алгоритму 7 зупиниться при першому ж входженні у підпрограму Het, тобто далі рекурсія не буде викликатися. Таких дерев із $LT_{2,n+1}$, в яких корінь має фіксовану мітку, всього $2^{2^{n+1}-2}$. Тому загальна кількість викликів Het в цьому випадку дорівнює:

$$1 \cdot 2 \cdot (2^{2^{n+1}-2})^2. \tag{50}$$

- *Випадок 2.* Нехай відповідні дерева мають однакові мітки на коренях: 0 і 0 або 1 і 1. Таких варіантів буде 2. Тоді алгоритм 7 один раз розгляне умову кроку 5, що відповідає розгляду кореня дерева, та піде далі рекурсивно до

піддерев, застосувуючи відповідні 2 виклики підпрограми Het. Зазначимо, що піддерева відповідних підстановок належать до $LT_{2,n}$. А тому за припущенням, в середньому вони вимагатимуть по $K(n) = 2n$ викликів підпрограми Het під час роботи алгоритму. Таким чином маємо наступну кількість застосувань Het для цього випадку:

$$(1 + 2K(n)) \cdot 2 \cdot (2^{2^{n+1}-2})^2. \tag{51}$$

Тоді за (50) та (51) маємо загальну кількість:

$$\Sigma(n+1) = 2 \cdot (2^{2^{n+1}-2})^2 + (1 + 2K(n)) \cdot 2 \cdot (2^{2^{n+1}-2})^2 =$$

$$= (2 + 2n) \cdot 2 \cdot \frac{1}{4} \cdot (2^{2^{n+1}-1})^2 = (n+1) \cdot (2^{2^{n+1}-1})^2,$$

а тому $K(n+1) = 2(n+1)$. Залишилось зауважити, що підпрограма Het виконує до 5 простих операцій у алгоритмі:

1. до 3х порівнянь у кроках 4, 6;
2. логічна операція xor у кроці 4;
3. додавання результатів виконань підпрограм Het у кроці 8

А тому середня складність алгоритму становить: $O(5 \cdot n) = O(n)$. □

3.4 Властивості відстані Хеммінга

Лема 3.4. *Відстань Хеммінга між довільними підстановками із $Syl_2(S_{2^n})$ є парним числом.*

Доведення. Нехай $\pi_1, \pi_2 \in Syl_2(S_{2^n})$ — 2-роздільні підстановки та $D_1, D_2 \in LT_{2,n}$ – відповідні дерева підстановок. Тоді за рівністю (47) маємо:

$$d_H(\pi_1, \pi_2) = h(\psi(D_1 \Delta D_2)).$$

Останнє значення є сумою кількістей висячих вершин, що знаходяться під головними вершинами дерева $D_1 \Delta D_2$ за наслідком 3.1. Залишилося зауважити, що кількість висячих вершин під деякою вершиною завжди є парним числом (степенем двійки). □

Лема 3.5. *Нехай m — це деяке парне число від 2^1 до 2^n. Тоді існують підстановки $\pi, \sigma \in Syl_2(S_{2^n})$ такі, що відстань Хеммінга між ними дорівнює числу m. Тобто:*

$$d_H(\pi, \sigma) = m.$$

Доведення. Оскільки за теоремою 3.3 $d_H(\pi_1, \pi_2) = d_H\Big(e, \psi(D_1 \Delta D_2)\Big)$, то для умови леми покладемо в якості підстановки π — нейтральний елемент $e \in Syl_2(S_{2^n})$. Зауважимо, що відповідне дерево $D_1 \in LT_{2,n}$ такої підстановки π має лише мітки 0. Розглянемо випадки $m = 2^n$ та $m \neq 2^n$.

1. Нехай $m = 2^n$. Тоді потрібно, щоб відповідне дерево $D_2 \in LT_{2,n}$ підстановки σ на корені мало мітку 1 і тоді отримаємо:

$$d_H(e, \sigma) = h(\psi(D_1 \Delta D_2)) = h(\psi(D_2)) = 2^n.$$

2. Нехай $m \neq 2^n$. Тоді дерево D_2 на корені має мітку 0.

   Також тоді парне число m можна єдиним чином представити у вигляді розкладу по степеням двійки (від 2^1 до 2^{n-1}):

$$m = m_1 \cdot 2^1 + \ldots + m_{n-1} \cdot 2^{n-1}, \text{ де } m_k = 0, 1, \ k \in \{1, n-1\}. \tag{52}$$

   Розставимо мітки 1 на дереві D_2, залежно від значень m_k, де k спадає від $n-1$ до 1 за наступними правилами:

   - рухаємося по дереву в напрямку від кореня до висячих вершин, тобто номери рівнів змінюються за правилом $j = n - k$;

- якщо $m_k = 0$, то пропускаємо рівень $j = n - k$ із відповідним значенням k ;
- якщо $m_k = 1$, то на рівні $j = n - k$ обираємо будь-яку одну вершину v таку, що жодна інша вершина із шляху від v до кореня v_0 не має мітки 1. На цій вершині v ставимо мітку 1. Тобто вона стане головною вершиною.

Останній пункт можливий, так як корінь має мітку 0. Крім того, при переході до наступного рівня кількість вершин на ньому збільшується в два рази, порівняно з попереднім, тобто у нас завжди буде для вибору 2 вершини, кожна з яких не має міток 1 на шляху між нею та коренем.

Отже, таке дерево D_2 буде мати щонайбільше по одній мітці 1 на кожному рівні j, $1 \leq j \leq 2^{n-1}$, окрім нульового. При чому, кожна вершина з міткою 1 буде головною вершиною дерева.

Оскільки кількість висячих вершин, що знаходяться під вершиною із рівня $j = n - k$, дорівнює $2^{n-j} = 2^k$, $k \in \{1, n-1\}$, то відповідно до (52) ми і отримаємо m – сума всіх висячих вершин під головними в дереві D_2. А тому:

$$d_H(e, \sigma) = h(\psi(D_1 \Delta D_2)) = h(\psi(D_2)) = m.$$

□

Результати наступних двох лем є відомими [8]. Оскільки будемо їх використовувати, то для повноти викладення матеріалу наведемо доведення.

Лема 3.6. *Відстань Хеммінга між підстановками зберігається при лівому множені. Тобто для довільних* $\pi_1, \pi_2, \sigma \in Syl_2(S_{2^n})$ *виконується:*

$$d_H(\sigma \cdot \pi_1, \sigma \cdot \pi_2) = d_H(\pi_1, \pi_2).$$

Доведення. Нехай $D_1, D_2 \in LT_{2,n}$ — відповідні дерева підстановок π_1, π_2. Тоді:

$$d_H(\pi_1, \pi_2) = h(\psi(D_1 \Delta D_2)). \tag{53}$$

Нехай $Q \in LT_{2,n}$ – відповідне дерево підстановки σ. Тоді:

$$d_H(\sigma \cdot \pi_1, \sigma \cdot \pi_2) = d_H(\psi(Q * D_1), \psi(Q * D_2)) = h\Big(\psi\big((Q * D_1)\Delta(Q * D_2)\big)\Big). \tag{54}$$

Зауважимо, що під симетричною різницею дерев A та B із $LT_{2,n}$ мається на увазі симетрична різниця відповідних множин вершин із мітками 1 $OV(A)\Delta OV(B)$. Останнє випливає з того, що кожне дерево з $LT_{2,n}$ однозначно визначається множиною вершин із мітками 1.

Розглянемо окремо операцію симемтричної різниці на деревах із рівності (54). За теоремою 2.3 отримаємо:

$$\begin{aligned}(Q * D_1)\Delta(Q * D_2) \ &= \Big(\mathbb{ACT}_Q(D_1)\Delta Q\Big)\Delta\Big(\mathbb{ACT}_Q(D_2)\Delta Q\Big) = \\ &= \mathbb{ACT}_Q(D_1)\Delta Q\Delta\mathbb{ACT}_Q(D_2)\Delta Q = \mathbb{ACT}_Q(D_1)\Delta\mathbb{ACT}_Q(D_2).\end{aligned}$$

Тоді за лемою 2.3, маємо:

$$\mathbb{ACT}_Q(D_1)\Delta\mathbb{ACT}_Q(D_2) = \mathbb{ACT}_Q\big(D_1\Delta D_2\big).$$

Отже, отримали, що

$$h\Big(\psi\Big((Q * D_1)\Delta(Q * D_2)\Big)\Big) = h\Big(\psi\Big(\mathbb{ACT}_Q\big(D_1\Delta D_2\big)\Big)\Big). \tag{55}$$

Оскільки відображення $\mathbb{ACT}$ не змінює кількість рухомих точок підстановки (кількість висячих вершин під головними вершинами відповідного дерева), то

$$h\Big(\psi\Big(\mathbb{ACT}_Q\big(D_1\Delta D_2\big)\Big)\Big) = h\Big(\psi\Big(D_1\Delta D_2\Big)\Big). \tag{56}$$

Отже, за рівностями (54), (55) та (56) маємо:

$$d_H(\sigma \cdot \pi_1, \sigma \cdot \pi_2) = h\Big(\psi\Big(D_1\Delta D_2\Big)\Big). \tag{57}$$

Тоді за рівностями (53) та (57) маємо:

$$d_H(\sigma \cdot \pi_1, \sigma \cdot \pi_2) = d_H(\pi_1, \pi_2).$$

□

Лема 3.7. *Відстань Хеммінга між підстановками зберігається при правому множені. Тобто для довільних* $\pi_1, \pi_2, \sigma \in Syl_2(S_{2^n})$ *виконується:*

$$d_H(\sigma \cdot \pi_1, \sigma \cdot \pi_2) = d_H(\pi_1, \pi_2).$$

Доведення. За лемою 3.6, множення зліва на підстановку $\sigma^{-1} \cdot \pi_1^{-1}$ не змінює відстань Хеммінга. А тому:

$$d_H(\pi_1 \cdot \sigma, \pi_2 \cdot \sigma) = d_H(\sigma^{-1} \cdot \pi_1^{-1} \cdot \pi_1 \cdot \sigma, \sigma^{-1} \cdot \pi_1^{-1} \cdot \pi_2 \cdot \sigma) =$$

$$d_H(e, \sigma^{-1} \cdot (\pi_1^{-1} \cdot \pi_2) \cdot \sigma) = h\Big(\tau\big(\sigma^{-1} \cdot (\pi_1^{-1} \cdot \pi_2) \cdot \sigma\big)\Big).$$

Кількість рухомих точок підстановки відповідає цикловому типу підстановки, як сумі довжин циклів. Операція спряження не змінює цикловий тип підстановки Тому з останньої рівності маємо:

$$h\Big(\tau\big(\sigma^{-1} \cdot (\pi_1^{-1} \cdot \pi_2) \cdot \sigma\big)\Big) = h\Big(\tau\big(\pi_1^{-1}\pi_2\big)\Big) = d_H(e, \pi_1^{-1}\pi_2).$$

За лемою 3.6 для множника π_1 маємо:

$$d_H(e, \pi_1^{-1}\pi_2) = d_H(\pi_1, \pi_1 \cdot \pi_1^{-1}\pi_2) = d_H(\pi_1, \pi_2).$$

Отже:

$$d_H(\pi_1 \cdot \sigma, \pi_2 \cdot \sigma) = d_H(\pi_1, \pi_2).$$

□

Лема 3.8. *Нехай $\pi \in Syl_2(S_{2^n})$, d — деяке парне число від 0 до 2^n. Тоді кількість підстановок, що знаходяться на відстані Хеммінга d від підстановки π, дорівнює кількості дерев Q, відповідні підстановки яких рухають рівно d точок:*

$$|\{\sigma \in Syl_2(S_{2^n}) | d_H(\pi, \sigma) = d\}| = |\{Q \in LT_{2,n} | h(\psi(Q)) = d\}|.$$

Доведення. Нехай $D \in LT_{2,n}$ — відповідне дерево підстановки $\pi \in Syl_2(S_{2^n})$ та $\sigma \in Syl_2(2_{2^n})$. Тоді підстановка σ буде знаходитися на відстані Хеммінга d від π за умови:

$$d = d_H(\pi, \sigma) = h\Big(\psi\big(D \Delta \tau(\sigma)\big)\Big). \tag{58}$$

Таким чином потужність множини $\{\sigma \in Syl_2(S_{2^n}) | d_H(\pi, \sigma) = d\}$ відповіває кількості підастановок σ, що задовільняє умові (58).

Оскільки $\tau : Syl_2(2_{2^n}) \to LT_{2,n}$ – це бієкція, то останнє буде залежати від кількості відповідних дерев $D' = \tau(\sigma) \in LT_{2,n}$ таких, що:

$$d = h\big(\psi(D \Delta D')\big).$$

Зазначимо, що останнє буде виконуватися для всіх дерев $D' \in \{D \Delta Q \big| Q \in LT_{2,n}, h(\psi(Q)) = d\}$, оскільки:

$$d = h\big(\psi(D \Delta D')\big) = h\big(\psi(D \Delta (D \Delta Q))\big) = h\big(\psi(D \Delta D \Delta Q)\big) = h\big(\psi(Q)\big).$$

При цьому:

$$|\{D \Delta Q \big| Q \in LT_{2,n}, h(\psi(Q)) = d\}| = |\{Q \in LT_{2,n} | h(\psi(Q)) = d\}|. \tag{59}$$

Із (59) випливає, що кількість підстановок σ, які знаходяться на відстані Хеммінга d від заданої підстановки π, рівна кількості таких дерев $Q \in LT_{2,n}$, для яких виконується:

$h(\psi(Q)) = d$ – кількість рухомих точок відповідних підстановок.

□

Отже, для групи $Syl_2(S_{2^n})$ задача пошуку кількості підстановок, що знаходяться на відстані Хеммінга d від певної фіксованої підстановки перетворюється у задачу пошуку підстановок, кількість рухомих точок яких рівна d.

3.5 Кількість підстановок, що мають задану кількість рухомих точок

Кожне натуральне число може бути представлено у вигляді суми степенів двійки. Наприклад число 10 може бути представлено у вигляді наступних сум:

$$10 = 0 * 2^0 + 1 * 2^1 + 0 * 2^2 + 1 * 2^3, \text{ або}$$

$$10 = 6 * 2^0 + 0 * 2^1 + 2 * 2^2 + 0 * 2^3.$$

Нехай m — деяке натуральне число.

Означення 3.3. Впорядкованим набором коефіцієнтів степенів двійки від 2^0 до 2^l розкладу числа m будемо називати такий набір $b = (b_0, b_1, \ldots, b_l)$, де:

$$m = b_0 \cdot 2^0 + b_1 \cdot 2^1 + \ldots + b_l \cdot 2^l,$$

та l є мінімальним таким натуральним числом, що $m \leq 2^l$.

Відповідно до визначення, для числа 10 маємо наступні набори $(0, 1, 0, 1)$ та $(6, 0, 2, 0)$.

Твердження 3.5. *Кількість різних наборів розкладу натурального числа m по степеням двійки обчислюється за формулою:*

$$f(m) = \begin{cases} f(m-1), \text{ якщо } m \text{ є непарним числом} \\ f(m-1) + f(\frac{m}{2}), \text{ якщо } m \text{ є парним числом} \end{cases}$$

Доведення. Розглянемо випадки відповідно до парності числа m.

1. Нехай m – непарне число. Зазначимо, що при цьому для кожного набору $b = (b_0, b_1, \ldots, b_l)$ розкладу числа m по степеням двійки число $b_0 \neq 0$.

   Тоді число m однозначно представляється у вигляді суми:

   $$m = 2 \cdot k + 1,$$

   для деякого k.

   Якщо $b = (b_0, b_1, \ldots, b_l)$ є набором розкладу числа m по степеням двійки:

   $$m = b_0 \cdot 2^0 + b_1 \cdot 2^1 + \ldots + b_l \cdot 2^l,$$

   то $b^{'} = (b_0 - 1, b_1, \ldots, b_l)$ є набором розкладу числа $2 \cdot k = m - 1$ по степеням двійки:

   $$2 \cdot k = m - 1 = (b_0 - 1) \cdot 2^0 + b_1 \cdot 2^1 + \ldots + b_l \cdot 2^l.$$

   І навпаки, якщо $b^{'} = (b_0, b_1, \ldots, b_l)$ є набором розкладу числа $2 \cdot k$ по степеням двійки у вигляді

   $$2k = b_0 \cdot 2^0 + b_1 \cdot 2^1 + \ldots + b_l \cdot 2^l,$$

   то набір $(b_0+1, b_1, \ldots, b_l)$ буде набором розкладу числа $2 \cdot k+1 = m$ по степеням двійки:

   $$m = 2 \cdot k + 1 = (b_0 + 1) \cdot 2^0 + b_1 \cdot 2^1 + \ldots + b_l \cdot 2^l$$

   Тому кількість різних наборів розкладу числа m по степеням двійки дорівнює кількості різних наборів розкладів числа $m - 1$ по степеням двійки, тобто:

   $$f(m) = f(m - 1).$$

2. Нехай число m є парним числом. Для кожного набору $b = (b_0, b_1, \ldots, b_l)$ розкладу числа m по степеням двійки або $b_0 = 0$, або $b_0 > 1$ — парне число. Розглянемо кожен з цих випадків окремо.

(а) Нехай $b_0 > 1$ — парне число для деякого набору b розкладу числа m по степеням двійки. Тоді набір b однозначно визначає набір $b' = (b_0 - 1, b_1, \ldots, b_l)$ розкладу числа $m-1$ по степеням двійки:

$$m - 1 = (b_0 - 1) \cdot 2^0 + b_1 \cdot 2^1 + \ldots + b_l \cdot 2^l$$

Відповідно, якщо $b' = (b_0, b_1, \ldots, b_l)$ є розкладом непарного числа $m-1$ по степеням двійки, то b_0 є непарним числом і відповідно $b = (b_0+1, b_1, \ldots, b_l)$ є розкладом числа m по степеням двійки.

Отже, кількість різних розкладів числа m по степеням двійки таких, що $b_0 > 1$ і є парним числом, дорівнює кількісті різних розкладів числа $m-1$ по степеням двійки, тобто $f(m-1)$.

(б) Нехай $b_0 = 0$. Тоді:

$$m = 0 \cdot 2^0 + b_1 \cdot 2^1 + \ldots + b_l \cdot 2^l = 2 \cdot (b_1 \cdot 2^0 + \ldots + b_l \cdot 2^{l-1})$$

Відповідно розклад b числа m однозначно визначає деякий розклад $b' = (b_1, \ldots, b_l)$ числа $\frac{m}{2}$ наступним чином:

$$\frac{m}{2} = b_1 \cdot 2^0 + \ldots + b_l \cdot 2^{l-1}$$

Отже, кожен розклад числа m по степеням двійки вигляду $(0, b_1, \ldots, b_l)$ однозначно визначає деякий рокзад b' числа $\frac{m}{2}$ по степеням двійки вигляду $(b_1, \ldots, b_l)$.

Вірно і навпаки: якщо $b' = (b_0, b_1, \ldots, b_{l-1})$ є розкладом числа $\frac{m}{2}$ по степеням двійки, то відповідно $b = (0, b_0, \ldots, b_{l-1})$ є розкладом числа m по степеням двійки.

Тому кількість різних розкладів числа m по степеням двійки таких, що $b_0 = 0$ дорівнює кількісті різних розкладів числа $\frac{m}{2}$, тобто $f(\frac{m}{2})$.

Зазначимо, що розглянуті випадки є неперетинними, а тому загальна кількість різних розкладів числа m по степеням двійки дорівнює:

$$f(m-1)+f(\frac{m}{2}).$$

□

Надалі, нас будуть цікавити розклади по степеням парних чисел, в яких $b_0 = 0$. Тому перший коефіцієнт b_0 розкладів b ми будемо опускати.

Відповідно до цього, перевизначимо через b – *впорядкований набір розкладу парного числа* m за зростанням степенів двійки від 2^1 до 2^l. Тоді для числа 10 маємо наступні приклади усіх таких розкладів b:

Варіант суми	b
$10 = 1*2^1 + 1*2^3$	$(1;0;1;0)$
$10 = 1*2^1 + 2*2^2$	$(1;2;0;0)$
$10 = 3*2^1 + 1*2^2$	$(3;1;0;0)$
$10 = 5*2^1$	$(5;0;0;0)$

Множину таких наборів для парного числа m будемо позначати як $\mathbb{B}(m)$.

Твердження 3.6. *Потужність множини різних наборів парного числа* $|\mathbb{B}(m)|$ *дорівнює* $f(\frac{m}{2})$, *де формулу* f *визначено в* 3.5

Доведення. Випливає безпосередньо із твердження 3.5 та частини його доведення п.2.а. □

Нагадаємо попередньо визначенні позначення:

- j – номер рівня дерева, $j \in \{0,\dots,n-1\}$;
- $b[r]$ – r-тий елемент з набору b;

- 2^{n-j} – кількість висячих вершин дерева, що знаходяться під вершиною із j-го рівня;

- k – показник степеня, тобто використовуємо для спрощення попереднього позначення і отримаємо залежність між рівнем та показником: $k = n - j$ або ж $j = n - k$.

Тоді маємо наступну теорему.

Теорема 3.6. *Кількість підстановок із $Syl_2(S_{2^n})$, що мають m рухомих точок, визначається за формулою:*

$$u(n,m) = \sum_{b \in \mathbb{B}(m)} \prod_{k=1}^{n} \binom{2^{n-k} - g(n,k,b)}{b[k]} \cdot f^{b[k]}(k), \tag{60}$$

де:

1. $g(n,k,b) = \sum\limits_{k'=k+1}^{n} 2^{k'-k} \cdot b[k']$ *— кількість вершин на рівні $j = n - k$, які не можна зробити головними;*

2. $f(k) = 2^{2^k - 1}$ *– кількість різних піддерев (а тому і їхніх відповідних підстановок), що можна отримати, для яких вершина на рівні $j = n - k$ є коренем.*

Доведення. Кількість рухомих точок залежить від множинм головних вершин відповідного дерева. Набір b задає кількість головних вершин кожного рівня з мітками (від 0-го до $n - 1$-го). Розміщення міток на вершинах під головними не впливає на кількість рухомих точок підстановки.

Нехай b — деякий набір кількостей головних вершин кожного рівня дерева. Тоді, для кожного показника k степеня:

- $\binom{2^{n-k} - g(n,k,b)}{b[k]}$ визначає кількість різних можливих варіантів розташування $b[k]$ штук головних вершин на рівні $j = n - k$.

- $f^{b[k]}(k)$ — це кількість різних розташувань міток (тобто кількість різних дерев) в $b[k]$ піддеревах з коренями у головних вершинах, що знаходяться на $j = n-k$ рівні.

Тоді, за правилом множення для набору b маємо, що добуток цих двох величин по всім $k \in \{1, \dots n\}$ — це кількість різних дерев, чий набір головних вершин визначений набором b та рухають рівно m висячих вершин.

Оскільки набір b однозначно визначається деревом, то дерева різних наборів не перетинаються. А тому за правилом суми маємо формулу 60.

□

3.6 Висновки до розділу

У розділі 3 розглядається відстань Хеммінга для підстановок із групи $Syl_2(S_{2^n})$. Для зображення підстановок групи використано відповідні кореневі бінарні n-рівневі дерева з мітками. Визначено кількість рухомих точок підстановок, а також для пошуку цієї кількості запропоновано алгоритм складності $O(2^n)$. Пораховано кількість підстановок, що мають мінімальну ненульову, максимальну або ж задану кількість рухомих точок. Запропоновано алгоритм складності $O(2^n)$ для обрахунку відстані Хеммінга між елементами групи $Syl_2(S_{2^n})$ та досліджено деякі основні властивості цієї метрики на силовській 2-підгрупі симетричної групи.

Результати цього розділу було опубліковано у працях [50], [51].

4 Коди на підстановках силовської 2-підгрупи симетричної групи

У теорії кодування розглядають коди, що визначені на симетричній групі підстановок S_n та її підгрупах. При цьому розглядаються різні метрики, такі як метрика Хеммінга, Улама, тощо.

Ми будемо досліджувати коди, визначені на групі $Syl_2(S_{2^n})$. Розглянемо, які властивості на цій групі буде мати відстань Хеммінга.

Відповідно до введених раніше означень, визначимо:

- $C_H(2^n, d)$ – код, визначений на підстановках із $Syl_2(S_{2^n})$ та з відстанню Хеммінга d такий, що для будь-яких $\pi, \sigma \in Syl_2(S_{2^n})$ маємо:

$$\pi, \sigma \in C_H(2^n, d) \text{ тоді і тільки тоді, коли } d_H(\pi, \sigma) \geq d.$$

- $A_H(2^n, d)$ – максимальний допустимий розмір коду із 2-роздільних підстановок із множини $Syl_2(S_{2^n})$ довжини 2^n та відстанню Хеммінга не меншою за d.

Для зручності, довільний код $C_H(2^n, d)$ будемо зображати матрицею, рядки якої відповідають нижнім стрічкам підстановок із $C_H(2^n, d)$.

Приклад 4.1. *Розглянемо групу*

$$Syl_2(S_{2^2}) = \left\{ \begin{pmatrix} 1&2&3&4 \\ 1&2&3&4 \end{pmatrix}, \begin{pmatrix} 1&2&3&4 \\ 1&2&4&3 \end{pmatrix}, \begin{pmatrix} 1&2&3&4 \\ 2&1&3&4 \end{pmatrix}, \begin{pmatrix} 1&2&3&4 \\ 2&1&4&3 \end{pmatrix}, \begin{pmatrix} 1&2&3&4 \\ 3&4&1&2 \end{pmatrix}, \begin{pmatrix} 1&2&3&4 \\ 3&4&2&1 \end{pmatrix}, \begin{pmatrix} 1&2&3&4 \\ 4&3&1&2 \end{pmatrix}, \begin{pmatrix} 1&2&3&4 \\ 4&3&2&1 \end{pmatrix} \right\}.$$

Одна із можливих підмножин, де відстань Хеммінга між будь-якими двома її підстановками буде не меншою за 4, наступна:

$$\left\{ \begin{pmatrix} 1&2&3&4 \\ 1&2&4&3 \end{pmatrix}, \begin{pmatrix} 1&2&3&4 \\ 2&1&3&4 \end{pmatrix}, \begin{pmatrix} 1&2&3&4 \\ 3&4&1&2 \end{pmatrix}, \begin{pmatrix} 1&2&3&4 \\ 4&3&2&1 \end{pmatrix} \right\} = C_H(2^2, 4).$$

І тоді маємо код у вигляді матриці:

$$M = \begin{pmatrix} 1 & 2 & 4 & 3 \\ 2 & 1 & 3 & 4 \\ 3 & 4 & 1 & 2 \\ 4 & 3 & 2 & 1 \end{pmatrix}.$$

Також тут $A_H(2^2, 4) = 4$.

Твердження 4.1. *Максимальний допустимий розмір коду підстановок із групи* $Syl_2(S_{2^n})$ *з відстанню Хеммінга не меншою ніж* 2^n *дорівнює* 2^n*, тобто*

$$A_H(2^n, 2^n) = 2^n.$$

Доведення. Припустимо, що код $C_H(2^n, 2^n)$ містить $2^n + k$ підстановок із $Syl_2(S_{2^n})$, $k \geq 1$. Тоді матриця цього коду буде мати $(2^n + k)$ рядків та 2^n стовбців. Оскільки елементи матриці беруться із множини $\{1, \ldots, 2^n\}$, то щонайменше буде дві стрічки, в яких на однакових місцях будуть розміщені однакові елементи. А тому відстань Хеммінга між підстановками, що відповідають цим стрічкам, буде меншою за 2^n, що суперечить умові твердження. Отже, кількість підстановок коду $C_H(2^n, 2^n)$ не може бути більшою за 2^n.

Покажемо тепер, що існує код розміру 2^n. Нехай $m = (m_0, \ldots, m_{n-1})$ – послідовність над множиною $\{0; 1\}$, тобто $m_j = 0, 1$, $j \in \{0, n-1\}$. Для кожного такого набору побудуємо дерево $D(m)$ наступним чином: якщо $m_j = 1$, то всі вершини дерева $D(m)$ j-го рівня будуть мати мітки 1, $j \in \{0, n-1\}$.

Зазначимо, що таких наборів рівно 2^n і кожен з них визначає різне дерево $D(m)$. Нехай m, t — дві різних послідовності. Тоді існує найменше таке j, що $m_j \neq t_j$. Тоді у відповідному дереві $D = D(m)\Delta D(t)$ j-й рядок буде містити мітки 1. Також, усі вершини з j-го рядка будуть головними, оскільки дане j є мінімальним. Тоді:

$$d_H(\psi(D(m)), \psi(D(t))) = h(\psi(D(m)\Delta D(t))) = 2^n.$$

Оскільки кількість різних послідовностей типу m дорівнює 2^n, то ми маємо 2^n різних підстановок коду $\left\{\psi(D(m))|m=(m_0,\ldots,m_{n-1}), m_i=0,1,\, i\in\{0,n-1\}\right\}=C_H(2^n,2^n)$. □

Приклад 4.2. *Нехай $Syl_2(S_{2^3})$. Також розглянемо максимальну відстань Хеммінга для цієї групи, 2^3. Тобто для довільних $\pi,\sigma\in Syl_2(S_{2^3})$:*

$$d_H(\pi,\sigma)=8.$$

Тоді можна навести такий приклад коду, для якого $A_H(2^3,2^3)=2^3$:

$$M=\begin{pmatrix}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8\\
2 & 1 & 4 & 3 & 6 & 5 & 8 & 7\\
3 & 4 & 1 & 2 & 7 & 8 & 5 & 6\\
4 & 3 & 2 & 1 & 8 & 7 & 6 & 5\\
5 & 6 & 7 & 8 & 1 & 2 & 3 & 4\\
6 & 5 & 8 & 7 & 2 & 1 & 4 & 3\\
7 & 8 & 5 & 6 & 3 & 4 & 1 & 2\\
8 & 7 & 6 & 5 & 4 & 3 & 2 & 1
\end{pmatrix}$$

4.1 Кількість підстановочних кодів виду $C_H(2^n,2^n)$

Зауваження 4.1. *З доведеного вище випливає, що кожен код $C_H(2^n,2^n)$ можна задати матрицею M, що має 2^n рядків і 2^n стовбців та для якої виконується:*

1. *всі елементи $a_{r,c}$ матриці M взяті із множини $\{1,\ldots,2^n\}$, $1\le r,c\le 2^n$;*
2. *в кожному стовбці всі елементи різні;*
3. *кожна стрічка такої матриці є нижнім рядком 2-роздільної підстановки.*

Зауважимо, що перестановка рядків матриці M не змінює коду, якому відповідає ця матриця. Тому, не порушуючи загальності, покладемо, що у матриці M на діагоналі будуть знаходитися елементи 1. Тобто

$$a_{r,r} = 1, \text{ де } 1 \leq r \leq 2^n.$$

Оскільки всі підстановки групи $Syl_2(S_{2^n})$ є 2-роздільними, то елементи 1 та 2 завжди будуть в одному блоці довжини 2. А також, оскільки в кожному стовбці може бути лише одна 2-ка, то отримаємо наступне розташування для неї:

$$2 = \begin{cases} a_{r,r+1}, & \text{якщо } r \text{ не парне;} \\ a_{r,r-1}, & \text{якщо } r \text{ парне.} \end{cases}$$

Таким чином, на головній діагоналі матриці будуть розташовані клітини вигляду $\begin{smallmatrix} 1 & 2 \\ 2 & 1 \end{smallmatrix}$ (див. Рис. 13):

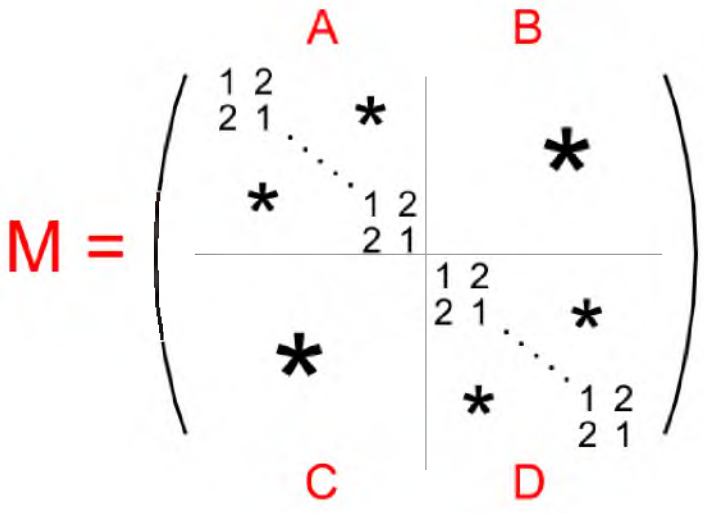


Рис. 13 Матриця M коду $C_H(2^n, 2^n)$.

Теорема 4.1. *Кількість підстановочних кодів $C_H(2^n, 2^n)$ з максимально відстан-*

ню Хеммінга $d_H = 2^n$ групи $Syl_2(S_{2^n})$ визначається рекурсивно та рівна:

$$f(n) = \begin{cases} 4, \ \textit{якщо } n = 2; \\ f^4(n-1) \cdot (2^{n-1}!)^2, \ \textit{якщо } n > 2. \end{cases}$$

Доведення. Відповідно до зауваження 4.1, кількість підстановочних кодів $C_H(2^n, 2^n)$ дорівнює кількості таких матриць M (див. Рис. 13).

Для доведення використаємо Метод матиматичної індукції.

База індукції. Нехай $n = 2$.

Розглянемо коди з підстановок із $Syl_2(S_{2^2})$ та відстанню Хеммінга $d_H(\pi, \sigma) = 4$ для довільних $\pi, \sigma \in Syl_2(S_{2^2})$:

$$\begin{pmatrix} 1 & 2 & 4 & 3 \\ 2 & 1 & 4 & 3 \\ 3 & 4 & 1 & 2 \\ 4 & 3 & 2 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \\ 4 & 3 & 1 & 2 \\ 3 & 4 & 2 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 4 & 3 \\ 2 & 1 & 3 & 4 \\ 3 & 4 & 1 & 2 \\ 4 & 3 & 2 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 4 & 3 \\ 2 & 1 & 3 & 4 \\ 3 & 3 & 1 & 2 \\ 3 & 4 & 2 & 1 \end{pmatrix}.$$

Легко перевірити, що інших кодів з такою ж відстанню немає.

Отже, $f(2) = 4$.

Індуктивний крок. Нехай умова теореми виконується для всіх попередніх значень n. Розглянемо тепер для випадку n.

Умовно, матрицю M можна розділити на 4 матриці: $M = \left(\begin{smallmatrix} A & B \\ C & D \end{smallmatrix}\right)$ (див. Рис. 13).

- Матриці A та D відповідають попередньому, $(n-1)$-му кроку. Тому для них, за комбінаторним правилом множення, маємо:

$$f^2(n-1). \tag{61}$$

- Враховуючи 2-роздільність кожної стрічки матриці M, маємо, що матриці B та C діють на елементах $\{2^{n-1}+1, 2^{n-1}+2, \ldots, 2^n\}$.

Зауважимо, що матрицю B (а також C) можна розглянути відповідним способом як матрицю A, наприклад розмістивши по головній діагоналі такіж клітини, тільки із елементів $2^{n-1}+1$ та $2^{n-1}+2$. А тому кількість різних впорядкувань з точністю до перестановок рядків співпадає з $f(n-1)$. Та кількість самих перестановок рядків рівна $2^{n-1}!$.

Отже, за правилом добутку маємо, що загальна кількість різних матриці B дорінює:

$$f(n-1)\cdot 2^{n-1}! \tag{62}$$

З аналогічних міркувань, для матриці C отримаємо таке ж значення:

$$f(n-1)\cdot 2^{n-1}! \tag{63}$$

Отже, в загальному випадку, кількість різних матриць M за (61)-(63) буде становити:

$$f(n)=f^2(n-1)\cdot f(n-1)\cdot 2^{n-1}!\cdot f(n-1)\cdot 2^{n-1}!=f^4(n-1)\cdot(2^{n-1}!)^2.$$

□

4.2 Найменший ненульовий випадок

Твердження 4.2. *Максимальний допустимий розмір коду підстановок із групи* $Syl_2(S_{2^n})$ *з відстанню Хеммінга не меншою ніж 2 становить* 2^{2^n-1}, *тобто:*

$$A_H(2^n,2)=2^{2^n-1}.$$

Доведення. Доведення випливає з того, що для довільних підстановок $\pi,\sigma\in Syl_2(S_{2^n})$ відстань Хеммінга така, що:

- $d_H(\pi,\sigma)$ – це завжди парне число за лемою 3.4;

- $d_H(\pi, \sigma) \geq 2$, де $\pi \neq \sigma$.

А тому матриця коду буде складатися зі всіх підстановок групи. Тобто

$$A_H(2^n, 2) = \left|Syl_2(S_{2^n})\right| = 2^{2^n - 1}.$$

□

Зауваження 4.2. *Існує єдиний код вигляду* $C_H(2^n, 2)$ *у групі* $Syl_2(S_{2^n})$.

4.3 Висновки до розділу

У розділі 4 розглядаються коди на підстановках силовської 2-підгрупи симетричної групи з відстанню Хеммінга. Для їх задання використано матричне представлення. Знайдено максимальні допустимі розміри коду підстановок із групи $Syl_2(S_{2^n})$ для найбільших та найменших відстаней Хеммінга. Пораховано кількість підстановочних кодів для максимальної відстані Хеммінга і наведено рекурсивну формулу.

Результати цього розділу було опубліковано у працях [51], [52].

5 Алгоритм обчислень у силовських 2-підгрупах знакозмінних груп за допомогою системи комп'ютерної алгебри GAP

Система комп'ютерної алгебри GAP (Groups, Algorithms and Programming [1], [42]) (м. Аахен, Німеччина 1986р.) спочатку була розроблена як інструмент комбінаторної теорії груп – розділу алгебри, що вивчає групи, представлені породжуючими елементами та певними співвідношеннями.

На даний момент система є міжнародним унікальним проектом, та об'єднує алгебру, теорію чисел, математичну логіку, інформатику та інші науки. Працює вона в різних версіях Unix/Linux, а також у Windows та MacOS. Проте деякі пакети, наприклад, графічний інтерфейс XGAP, функціонують лише в Unix/Linux.

GAP є системою з вільним доступом, яка постійно вдосконалюється та розширюється. Її ядро написане мовою C, а бібліотека функцій – на спеціальній мові, синтаксис якої є аналогом Pascal, та є об'єктно-орієнтовним програмуванням. Користувачі можуть створювати власні програми, а також оформляти власні розробки у вигляді пакета для системи GAP. Окрім того, затверджений пакет Радою GAP прирівнюється до наукової публікації.

5.1 Операції над групами

1. Нагадаємо, що група підстановок S_8 породжується наступною множиною підстановок: $(1,2)$ та $(1,2,3,4,5,6,7,8)$. На мові GAP:

$$gap > s8 := Group\ ((1,2),(1,2,3,4,5,6,7,8));$$
$$Group\ ([(1,2),(1,2,3,4,5,6,7,8)]),$$

де перша стрічка — запит на створення групи, що породжується множиною з

двох підстановок, а друга — автоматичний вивід на екран нашого об'єкту.

2. Група S_8 має підгрупу парних підстановок A_8, яка може бути представлена як підгрупа із парних підстановок або як її комутант:

$$gap > a8 := DerivedSubgroup(s8);$$
$$Group([(1,2,3),(2,3,4),(3,4,5),(2,6)(3,4),(3,7)(4,5),(3,5,6,7)(4,8)]),$$

де перша стрічка — функція, що для заданої групи S_8 знаходить її комутант, а друга — автоматичний вивід результату – групи A_8, що є підгрупою S_8.

3. Для зручності, можна присвоїти об'єкту ім'я.

$$gap > SetName(a8, "A8");$$

Після цього, кожного наступного разу, на екран буде показано не сам об'єкт, а його ім'я:

$$\begin{array}{ll} gap > a8; & \text{— вивести на екран об'єкт } a8 \\ A8 & \text{— результат на екрані.} \end{array}$$

4. Знайдемо порядок групи A_8:

$$\begin{array}{ll} gap > Size(a8); & \text{— запит на розмір групи } A_8 \\ 20160 & \text{— результат (розмірність).} \end{array}$$

Інші операції розглянемо далі разом із самим алгоритмом.

5.2 Алгоритм та його реалізація за допомогою GAP

Ідея алгоритму полягає в перевірці достатньої умови (теореми 1.2) для системи твірних S силовської 2-підгрупи групи A_{2^n}.

Вхід: S — множина підстановок.

Задача: за допомогою інструментів GAP шляхом порівняння відомих властивостей силовської групи з даною перевірити чи є S мінімальною системою твірних.

Позначення:

- G — група, яка утворюється множиною S та буде об'єктом дослідження;
- n — кількість елементів множини S;
- Com — комутант групи G;
- $Factor$ — фактор групи G по її комутанту;
- k, i — допоміжні змінні;
- St — система твірних групи A_{2^n};
- Alt — група A_{2^n};
- $Syl2$ — силовська 2-підгрупа групи A_{2^n};

Ініціалізуємо всі об'єкти, створюємо групу G за системою твірних S та знаходимо розмірність множини S:

```
1 CMG := function(S)
2 local G,Com,Faktor,k,i,St,Alt,Syl2;
3 G:=Group(S); n:=Size(S);
```

Відомо, що множина підстановок вигляду $\{(1,2,3),(1,2,4),...,(1,2,2^n)\}$ є системою твірних групи A_{2^n}. Задамо її мовою GAP за допомогою функції $AddSet$ та позначимо через St:

```
4 St:=[];
5 for i in [3,2^n] do
6     AddSet(St,(1,2,i));
7 od;
```

Створимо групу A_{2^n} по системі твірних St, яку позначимо Alt. Також, через $Syl2$ позначимо силовську 2-підгрупу групи A_{2^n}.

```
8  Alt:=Group(St);
9  Syl2:=SylowSubgroup(Alt,2);
```

Для того, щоб група G була силовською 2-підгрупою групи A_{2^n} необхідно перевірити наступні 2 умови:

1. G є підгрупою A_{2^n}:

```
10 if not IsSubgroup(Alt,G) then
11     Print(S,"is not in Alt(",2^n,")","\n");
12     return 0;
13 fi;
```

2. Потужність групи G співпадає з потужністю силовської 2-підгрупи групи A_{2^n}:

```
14 if Size(G)=Size(Syl2) then
15     Print(S,"is generators set of SylowSubgroup Alt(",2^n,")","\n");
16 else
17     Print(S,"is not generators set ","\n");
18     return 0;
19 fi;
```

Знайдемо комутант групи G та фактор групи G по комутанту:

```
20 Com:=CommutatorSubgroup(G,G);
21 Factor:=FactorGroup(G,Com);
```

Перевіримо на мінімальність систему твірних S відповідно до теореми:

```
22 k:=Minimum(Size(GeneratorsOfGroup(Factor)),Size(S)+1);
23 if k=Size(S) then
24     Print(S,"is min generators set of SylowSubgroup
25           of Alt(",2^n,")");
26 else
27     Print(S,"is not min generators set of SylowSubgroup
28           of Alt(",2^n,")");
```

```
29 fi;
```

Додаткові властивості:

- $30-34$ — абелевість комутанта групи G;
- 35 — потужність комутанта групи G;
- $36-37$ — кількість елементів мінімальної системи твірних комутанта групи G;

```
30 if IsAbelian(Com) then
31     Print("CommutatorSubgroup is abelian","\n");
32 else
33     Print("CommutatorSubgroup is not abelian","\n");
34 fi;
35 Print("Size of CommutatorSubgroup is",Size(Com),"\n");
36 Print("Count of min generatirs set of CommutatorSubgroup is",
37       Size(GeneratorsOfGroup(Com)),"\n");
```

- $38-42$ — абелевість фактора групи G по комутанту;
- 43 — потужність фактора групи G по комутанту;
- $44-45$ — кількість елементів мінімальної системи твірних фактора групи G по комутанту;

```
38 if IsAbelian(Factor) then
39     Print("FactorGroup is abelian","\n");
40 else
41     Print("FactorGroup is not abelian","\n");
42 fi;
43 Print("Size of FactorGroup is", Size(Factor),"\n");
44 Print("Count of min generatirs set of FactorGroup is",
45       Size(GeneratorsOfGroup(Factor)),"\n");
```

```
46
47 end;;
```

5.3 Результати GAP для A_8

Розглянемо наступні елементи групи $Syl_2(A_8)$:

$\alpha_0 = (1,5)(2,6)(3,7)(4,8)$, $\alpha_1 = (1,3)(2,4)$, $\alpha_2 = (1,2)(5,6)$.

За допомогою системи комп'ютерної алгебри GAP для випадку A_8 було отримано наступні результати:

1. набір елементів α_0, α_1 і α_2 є системою твірних $Syl_2(A_8)$.

2. Комутант групи $Syl_2(A_8)$ породжується підстановками $(1,3)(2,4)(5,7)(6,8)$, $(1,2)(3,4)$, $(1,3)(2,4)(5,8)(6,7)$ і є елементарною абелевою 2-групою порядку 8, тобто ізоморфний ${C_2}^3$;

3. Фактор по комутанту групи $Syl_2(A_8)$ також є елементарною абелевою 2-групою порядку 8.

4. Система твірних $\{\alpha_0, \alpha_1, \alpha_2\}$ є мінімальною.

5.3.1 Властивості комутанта групи

Спираючись на результати GAP для групи $Syl_2(A_8)$, були отримані наступні теореми:

Теорема 5.1. *Кожен елемент комутанта групи $Syl_2(A_8)$ є комутатором.*

Доведення. Комутант $Syl_2(A_8)$ складається з елементів $\{e, (13)(24)(57)(68), (12)(34), (14)(23)(57)(68), (56)(78), (13)(24)(58)(67), (12)(34)(56)(78), (14)(23)(58)(67)\}$.

Перевіркою, отримаємо наступне:

$$(13)(24)(57)(68) = [\alpha_1, \alpha_0]$$
$$(12)(34) = [\alpha_1, \alpha_2]$$
$$(14)(23)(57)(68) = [\alpha_1\alpha_2, \alpha_1\alpha_0]$$
$$(56)(78) = [\alpha_2, \alpha_0\alpha_1]$$
$$(13)(24)(58)(67) = [\alpha_1\alpha_2, \alpha_0\alpha_1]$$
$$(12)(34)(56)(78) = [\alpha_2, \alpha_1\alpha_0\alpha_1]$$
$$(14)(23)(58)(67) = [\alpha_2\alpha_1, \alpha_0]$$

□

Теорема 5.2. *Множина комутаторів твірних $\alpha_0, \alpha_1, \alpha_2$ породжує в комутанті групи $Syl_2(A_8)$ власну підгрупу порядку 4.*

Доведення. З доведення попередньої теореми:

$$[\alpha_1, \alpha_0] = (13)(24)(57)(68)$$
$$[\alpha_1, \alpha_2] = (12)(34)$$
$$[\alpha_0, \alpha_2] = e$$

Комутант має порядок 8, дільники його порядку: $1, 2, 4, 8$. Оскільки отримали лише 2 нетривіальні елементи, то весь комутант породити комутаторами твірних не можна. А, отже, вони породжують власну підгрупу порядку 4. □

5.4 Результати GAP для A_{16}

Розглянемо наступні елементи групи $Syl_2(A_{16})$:

$\alpha_0 = (1,9)(2,10)(3,11)(4,12)(5,13)(6,14)(7,15)(8,16)$, $\alpha_1 = (1,5)(2,6)(3,7)(4,8)$, $\alpha_2 = (1,3)(2,4)$, $\alpha_3 = (1,2)(9,10)$.

Для випадку $Syl_2(A_{16})$ за допомогою GAP були отримані наступні результати:

1. набір елементів $\alpha_0, \alpha_1, \alpha_2$ та α_3 є системою твірних $Syl_2(A_{16})$.

2. Комутант $Syl_2(A_{16})$ – не абелева група порядку 2^{10}, що породжується 5-ма твірними:

   $(1,4,2,3)(5,6)(9,12)(10,11)$, $(1,4)(2,3)(5,8)(6,7)$, $(1,2)(5,6)$,
   $(1,7,3,5)(2,8,4,6)(9,14,12,16)(10,13,11,15)$,
   $(1,7)(2,8)(3,6)(4,5)(9,16,10,15)(11,14,12,13)$.

3. Фактор-група $Syl_2(A_{16})$ по її комутанту – абелева група порядку 16, що породжується 4 елементами.

4. Система твірних $\{\alpha_0, \alpha_1, \alpha_2, \alpha_3\}$ групи $Syl_2(A_{16})$ є мінімальною.

5.5 Висновки до розділу

У розділі 5 наведений опис та приклади використання інструментів системи комп'ютерної алгебри GAP для обчислень у силовських 2-підгрупах знакозмінних груп A_{2^n}. Зокрема описані основні операції, які можемо виконувати за допомогою мови GAP та запропоновано алгоритм перевірки чи є обрана множина елементів системою твірних силовської 2-підгрупи знакозмінної групи A_{2^n}. Завдяки алгоритму було отримано висновки про комутант та його властивості для A_8 та A_{16}.

Результати цього розділу було опубліковано у працях [43], [46].

Висновки

У дисертаційній роботі досліджуються алгоритми та їхні властивості над силовськими 2- підгрупами $Syl_2(S_{2^n})$ симетричних груп S_{2^n}. Для цього було використано представлення елементів групи бінарними кореневими n-рівневими деревами з мітками із $LT_{2,n}$. Також розглядаються підстановочні коди над $Syl_2(S_{2^n})$.

Описано алгоритм множення двох бінарних n-рівневих кореневих дерев з мітками з часовою складністю $O(n \cdot 2^n)$ із використанням алгоритму switch, який має складність $O(2^{n-r})$. Введено алгоритм знаходження оберненого дерева з часовою складністю $O(n \cdot 2^n)$. Також запропоновано два алгоритми: перетворення дерева із $LT_{2,n}$ у підстановку з $Syl_2(S_{2^n})$ з часовою складністю $O(n \cdot 2^n)$ та навпаки, перетворення підстановки із $Syl_2(S_{2^n})$ у бінарне n-рівневе дерево з $LT_{2,n}$ з часовою складністю $O(2^n)$. Коректність наведених алгоритмів ґрунтується на властивостях відображення $\mathbb{ACT}$, визначених у цьому розділу.

Доведено теорему про ізоморфізм між силовською 2-підгрупою $Syl_2(S_{2^n})$ симетричної групи S_{2^n} та множиною бінарних n-рівневих кореневих дерев з мітками $LT_{2,n}$, з якої випливає, що $LT_{2,n}$ з заданою операцією множення $*$ є групою.

Визначено кількість рухомих точок підстановок із $Syl_2(S_{2^n})$, що задаються відповідними деревами із $LT_{2,n}$, а також для пошуку цієї кількості запропоновано алгоритм складності $O(2^n)$. Пораховано кількість підстановок, що мають мінімальну ненульову, максимальну або ж задану кількість рухомих точок. Запропоновано алгоритм складності $O(2^n)$ для обрахунку відстані Хеммінга між елементами групи $Syl_2(S_{2^n})$ та досліджено деякі основні властивості цієї метрики на силовській 2-підгрупі симетричної групи.

У дисертаційній роботі досліджено коди на підстановках силовської 2-підгрупи симетричної групи з відстанню Хеммінга. Для їх задання використано матричне представлення. Знайдено максимальні допустимі розміри коду підстановок із гру-

пи $Syl_2(S_{2^n})$ для найбільших та найменших відстаней Хеммінга. Також пораховано кількість підстановочних кодів для максимальної відстані Хеммінга і наведено рекурсивну формулу.

Також наведено опис та приклади використання інструментів системи комп'ютерної алгебри GAP для обчислень у силовських 2-підгрупах знакозмінних груп A_{2^n}. Зокрема описані основні операції, які можемо виконувати за допомогою мови GAP та запропоновано алгоритм перевірки чи є обрана множина елементів системою твірних силовської 2-підгрупи знакозмінної групи A_{2^n}. Завдяки алгоритму було отримано висновки про комутант та його властивості для A_8 та A_{16}.

Список використаних джерел

[1] GAP — Groups, Algorithms, and Programming - a System for Computational Discrete Algebra Version 4.9.3 (2018). URL https://www.gap-system.org.{"---}Titlefromthescreen

[2] Abdollahi, A., Bagherian, J., Jafari, F., Khatami, M., Parvaresh, F., Sobhani, R.: New upper bounds on the size of permutation codes under kendall τ-metric (2023)

[3] Bailey, R.F.: Error-correcting codes from permutation groups. Discrete Math. **309**(13), 4253–4265 (2009). DOI 10.1016/j.disc.2008.12.027

[4] Bailey, R.F., Nicholson, K.B.: Decoding twisted permutation codes (2023)

[5] Blake, I.F., Cohen, G., Deza, M.: Coding with permutations. Inf. Control **43**, 1–19 (1979). DOI 10.1016/S0019-9958(79)90076-7

[6] Bortolussi, L., Dinu, L.P., Sgarro, A.: Spearman permutation distances and Shannon's distinguishability. Fundam. Inform. **118**(3), 245–252 (2012). DOI 10.3233/FI-2012-712

[7] Cameron, P.J.: Permutation codes. Eur. J. Comb. **31**(2), 482–490 (2010). DOI 10.1016/j.ejc.2009.03.044

[8] Cameron, P.J., Gadouleau, M.: Remoteness of permutation codes. Eur. J. Comb. **33**(6), 1273–1285 (2012). DOI 10.1016/j.ejc.2012.03.027

[9] Chee, Y.M., Ling, S., Nguyen, T.T., Vu, V.K., Wei, H.: Permutation codes correcting a single burst deletion ii: Stable deletions. In: 2017 IEEE International Symposium on Information Theory (ISIT), pp. 2688–2692 (2017). DOI 10.1109/ISIT.2017.8007017

[10] Chee, Y.M., Purkayastha, P.: Efficient decoding of permutation codes obtained from distance preserving maps pp. 636–640 (2012). DOI 10.1109/ISIT.2012.6284273

[11] Chee, Y.M., Zhang, H.: Neural network decoders for permutation codes correcting different errors (2022)

[12] Dénes, J.: On some connections between permutations and coding. Discrete Math. **56**, 141–146 (1985). DOI 10.1016/0012-365X(85)90022-6

[13] Deza, M.M., Deza, E.: Encyclopedia of distances. Berlin: Springer (2009). DOI 10.1007/978-3-642-00234-2

[14] Diestel, R.: Graph theory, vol. 173. Berlin: Springer (2017). DOI 10.1007/978-3-662-53622-3

[15] Dmitruk, J.V.: Structure of Sylow two-subgroups of the symmetric group of degree 2^n. Ukr. Math. J. **30**, 117–124 (1978). DOI 10.1007/BF01085629

[16] Dmitruk, Y.V., Sushchanskij, V.I.: Structure of Sylow 2-subgroups of the alternating groups and normalizers of Sylow subgroups in the symmetric and alternating groups. Ukr. Math. J. **33**, 235–241 (1982). DOI 10.1007/BF01085560

[17] E. Irurozki B. Calvo, J.A.L.: Sampling and learning the Mallows and Weighted Mallows models under the Hamming distance (2014). URL http://hdl.handle.net/10810/11240

[18] Farnoud, F., Skachek, V., Milenkovic, O.: Error-correction in flash memories via codes in the Ulam metric. IEEE Trans. Inf. Theory **59**(5), 3003–3020 (2013). DOI 10.1109/TIT.2013.2239700

[19] Fry, A.S., Taylor, J.: On self-normalising sylow 2-subgroups in type a (2017)

[20] Goldreich, O.: P, NP, and NP-completeness. The basics of computational complexity. Cambridge: Cambridge University Press (2010). DOI 10.1017/CBO9780511761355

[21] Guenda, K., Gulliver, T.A.: On the permutation groups of cyclic codes. Journal of Algebraic Combinatorics **38** (2013). DOI 10.1007/s10801-012-0399-4

[22] Huczynska, S.: Powerline communication and the 36 officers problem. Philos. Trans. R. Soc. Lond., Ser. A, Math. Phys. Eng. Sci. **364**(1849), 3199–3214 (2006). DOI 10.1098/rsta.2006.1885

[23] HUNG, N.N., TIEP, P.H.: Irreducible characters of even degree and normal sylow 2-subgroups. Mathematical Proceedings of the Cambridge Philosophical Society **162**(2), 353–365 (2016). DOI 10.1017/s0305004116000669. URL https://doi.org/10.1017%2Fs0305004116000669

[24] Kaloujnine, L.: La structure des p-groupes de Sylow des groupes symétriques finis. Ann. Sci. Éc. Norm. Supér. (3) **65**, 239–276 (1948). DOI 10.24033/asens.961

[25] Lang, S.: Algebra., *Grad. Texts Math.*, vol. 211, 3rd revised ed. edn. New York, NY: Springer (2002)

[26] Montemanni, R., Barta, J., Smith, D.H.: The design of permutation codes via a specialized maximum clique algorithm. In: 2015 Second International Conference on Mathematics and Computers in Sciences and in Industry (MCSI), pp. 298–301 (2015). DOI 10.1109/MCSI.2015.54

[27] Moori, J., Radohery, G.F.R.: 2-designs and codes from simple groups $L_3(q)$ and higman-sims sporadic simple group HS. In: Coding theory and applications. 4th international castle meeting, ICMCTA, Palmela Castle, Portugal, September 15–18, 2014, pp. 281–289. Cham: Springer (2015). DOI 10.1007/978-3-319-17296-5_30

[28] Narayanan, S.: The representation theory of 2-sylow subgroups of the symmetric group (2020)

[29] Nekrashevych, V.: Self-similar groups., vol. 117. Providence, RI: American Mathematical Society (AMS) (2005)

[30] Pawlik, B.: The action of Sylow 2-subgroups of symmetric groups on the set of bases and the problem of isomorphism of their Cayley graphs. Algebra Discrete Math. **21**(2), 264–281 (2016)

[31] Pawlik, B.: The girth of cayley graphs of sylow 2-subgroups of symmetric groups s_{2^n} on diagonal bases (2018)

[32] Robinson, D.J.S.: A course in the theory of groups., *Grad. Texts Math.*, vol. 80, 2nd ed. edn. New York, NY: Springer-Verlag (1995)

[33] Slepian, D.: Permutation modulation. Proceedings of the IEEE **53**(3), 228–236 (1965). DOI 10.1109/PROC.1965.3680

[34] Slupik, A.J., Sushchansky, V.I.: Minimal generating sets and Cayley graphs of Sylow p-subgroups of finite symmetric groups. Algebra Discrete Math. **2009**(4), 167–184 (2009)

[35] Sobhani, R., Abdollahi, A., Bagherian, J., Khatami, M.: A note on good permutation codes from Reed-Solomon codes. Des. Codes Cryptography **87**(10), 2335–2340 (2019). DOI 10.1007/s10623-019-00621-0

[36] Szirmay-Kalos, L., Márton, G.: Worst-case versus average case complexity of ray-shooting. Computing **61**(2), 103–131 (1998). DOI 10.1007/BF02684409

[37] V. S. Sikora, V.I.S.: Операції на групах підстановок. Теорія та застосування. Видання друге. Чернівці: Технодрук (2017)

[38] Weir, A.J.: The sylow subgroups of the symmetric groups. Proceedings of the Ameri-

can Mathematical Society **6**(4), 534–541 (1955). URL http://www.jstor.org/stable/2033425

[39] Безущак О. О., Ганюшкін О. Г.: Теорія груп. Видавничо-поліграфічний центр "Київський університет"(2005)

[40] Бондарчук Ю. В., Олійник Б. В.: Основи дискретної математики. Києво-Могилянська академія (2009)

[41] Калужнин Л. А.: Избранные главы теории групп pp. 22–26 (1979)

[42] Корольський В. В., Крамаренко Т. Г., Семеріков С. О., Шокалюк С. В.: Інноваційні інформаційно-комунікаційні технології навчання математики. Кривий Ріг: Книжкове видавництво Киреєвського (2009)

Додатки

Список публікацій за темою дисертації

[43] В. А. Ольшевська. Застосування GAP для обчислень в силовських 2-підгрупах знакозмінних груп. Збірник тез XV Всеукраїнська науково-практична конференція студентів, аспірантів та молодих вчених «Теоретичні і прикладні проблеми фізики, математики та інформатики», т. 2, сс. 55-56, Київ, 25 -27 травня, 2017.

[44] Olshevska V. A. Algorithms for computations in Sylow 2-subgroups of symmetric and alternating groups. Book of abstracts of 11th International Algebraic Conference in Ukraine dedicated to the 75th anniversary of V.V. Kirichenko, p. 94, Kyiv, July 3-7, 2017.

[45] В. А. Ольшевська. Зображення підстановок кореневими деревами. Збірник наукових праць Міжнародна наукова конференція «Сучасні проблеми механіки та математики», том 3, с. 222, Львів, 22-25 травня, 2018р.

[46] В. А. Ольшевська. Алгоритм обчислень у силовських 2-підгрупах знакозмінних груп за допомогою системи комп'ютерної алгебри GAP // Могилянський математичний журнал, том 1, ISSN 2617-7080, С. 30-33, 2018.

[47] Olshevska V. A. Algorithms for computations with Sylow 2-subgroups of symetric groups // Silesian Journal of Pure and Applied Mathematics, Vol. 10, e-ISSN 2719-7913, pp. 103-120, 2020.

[48] В.А.Ольшевська. Складність алгоритму зображення силовськаих 2-підгруп симетричних груп кореневими деревами. Збірник тез X Всеукраїнська наукова конференція молодих математиків, с. 91, Київ, 16-17 квітня, 2021.

[49] Olshevska V. A. Fast multiplication algorithm for Sylow 2-subgroups of symmetric groups. Book of Abstracts of 13th International Algebraic Conference in Ukraine, p. 59, Kyiv, July 6-9, 2021.

[50] В.А.Ольшевська. Алгоритм пошуку кількості рухомих точок підстановок із силовських 2-підгруп $Syl_2(S_{2^n})$ симетричних груп S_{2^n} //Могилянський математичний журнал, том 4, С. 34-40, 2022

[51] V.A.Olshevska. Permutation codes over Sylow 2-subgroups $Syl_2(s_{2^n})$ of symmetric groups S_{2^n} // Researches in Mathematics, Vol. 29, No. 2, ISSN 2664-4991, e-ISSN 2664-5009, pp. 28-43, 2021.

[52] Olshevska V. A. Permutation codes over Sylow 2-subgroups $Syl_2(S_{2^n})$ of symmetric groups S_{2^n} with Hamming distance. Book of Abstracts of International Algebraic Conference "At the End of the Year"2022, p. 38, Kyiv, December 27-28, 2022.

Відомості про апробацію результатів дисертації

Конференції:

1. XV Всеукраїнська науково-практична конфернція студентів, аспірантів та молодих вчених «Теоретичні і прикладні проблеми фізики, математики та інформатики», Київ, 25 -27 травня, 2017.

2. 11th International Algebraic Conference in Ukraine dedicated to the 75th anniversary of V.V. Kirichenko, Kyiv, July 3-7, 2017.

3. Міжнародна наукова конференція «Сучасні проблеми механіки та математики», Львів, 22-25 травня, 2018р.

4. VII щорічна PhD конференція Докторської школи ім. родини Юхименків НаУКМА "A LINEA Київ, 4-8 червня 2018.

5. X Всеукраїнська наукова конференція молодих математиків, Київ, 16-17 квітня, 2021.

6. 13th International Algebraic Conference in Ukraine, Kyiv, July 6-9, 2021.

7. International Algebraic Conference "At the End of the Year" 2022, Kyiv, December 27-28, 2022.

Семінари:

1. семінар аспірантів факультету інформатики НаУКМА (за участю гостьового професора з Вільнюського університету V.Dagiene), 12 квітня 2019.

Міжнародний консорціум для аспірантів:

1. 10th International Doctoral Consortium on Informatics and Informatics Engineering Education Research joint with Nordplus workshop Culturally Diverse Approaches to Learning Mathematics and Computational Thinking, Druskininkai, Lithuania, December 2–6, 2019,

Документ підписано у сервісі Вчасно (продовження)
Ольшевська-Дисертація-з підписом.pdf

Документ відправлено: 16:43 04.12.2023

Власник документу

Електронний підпис
16:43 04.12.2023
Ідентифікаційний код: 3424309881
ОЛЬШЕВСЬКА ВІТА АНАТОЛІЇВНА
Власник ключа: ОЛЬШЕВСЬКА ВІТА АНАТОЛІЇВНА
Час перевірки КЕП/ЕЦП: 16:43 04.12.2023
Статус перевірки сертифікату: Сертифікат діє
Серійний номер: 5E984D526F82F38F04000000561ED2005EB39E04
Тип підпису: удосконалений