Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра математики факультету інформатики

СКІНЧЕННІ ІГРИ З НУЛЬОВОЮ СУМОЮ

Текстова частина до курсової роботи за спеціальністю „Системний аналіз” 124

Керівник курсової
роботи доцент,
кандидат фізико-
математичних наук
Чорней Р. К.

_____
(підпис)
“____” _____ 2023 р.

Виконав студент СА-1
Худенко К.В.
“____” _____ 2023 р.

Київ – 2023

Календарний план

№ п/п	Назва етапу дипломного проекту (роботи)	Термін виконання етапу	Примітки
1.	Отримання завдання на курсову роботу.	11.02.2023	
2.	Огляд технічної літератури за темою роботи.	21.03.2023	
3.	Аналіз методу повного перебору та створення програмного коду	03.04.2023	
3.	Аналіз методу домінуючих стратегій та створення програмного коду	15.04.2023	
4.	Аналіз методу максиміну і мінімакса та створення програмного коду	22.04.2023	
5.	Аналіз розв'язання задачі для змішаних стратегій та створення програмного коду	29.04.2023	
6.	Написання текстової частини роботи	10.05.2023	
7.	Створення презентації	12.05.2023	
8.	Корегування роботи згідно з результатами перевірки роботи науковим керівником.	14.05.2023	
8.	Остаточне оформлення пояснювальної роботи та слайдів.	17.05.2023	
10.	Подання роботи на кафедру для перевірки на плагіат	18.05.2023	
11.	Захист курсової роботи	18 – 29.05. 2023	

Студент _____

Керівник _____

“_____” _____

Зміст

1. ВСТУП

Розглядаючи такий величний науковий пласт як математика не можна оминути ту її частину, назва якої може зацікавити навіть тих, хто воліє не мати нічого спільного з математикою – теорію ігор.

У цій науковій роботі я досліджую клас задач, відомий як "скінченні ігри з нульовою сумою". Ця галузь математики вивчає взаємодію між двома гравцями в конфліктній ситуації, де кожна сторона має протилежні інтереси. Нашою основною метою є знайти розв'язок цих ігор, тобто стратегії, які максимізують виграш одного гравця при будь-якій відповіді іншого гравця.

Об'єкт мого дослідження є ігри з ненульовою сумою. У таких іграх сума виграшів двох гравців завжди дорівнює нулю, що означає, що виграш одного гравця є втратою іншого. Це робить ігри з нульовою сумою особливими, оскільки вони вимагають від гравців вдумливої стратегії та розуміння психології іншого гравця.

Наш дослідницький підхід базується на математичних методах та алгоритмах. Ми розробили власний код, який використовується для аналізу ігор з нульовою сумою та знаходження їх розв'язків. Цей код дозволяє нам проводити обчислення, моделювати різні сценарії та отримувати результати, які сприяють нашому розумінню таких ігор.

Новизною цієї роботи є наш підхід до моделювання скінченних ігор з нульовою сумою. Я розробив алгоритми, які дозволяють ефективно обробляти складність цих ігор і знаходити їх розв'язки. Наші результати відкривають нові можливості для дослідження таких ігор та застосування їх у різних сферах, таких як економіка, політика та біологія.

Оглядаючи попередні дослідження, я бачу, що скінченні ігри з нульовою сумою мають великий потенціал для застосування. Вивчення їх розв'язків може допомогти у прийнятті раціональних рішень та оптимізації виграшів в конфліктних ситуаціях. Наша стаття пропонує новий підхід та результати, які сприятимуть подальшому розвитку цієї галузі.

У наступних розділах ми детально розглянемо математичні моделі і алгоритми, які використовуються у нашому дослідженні, а також представимо результати наших обчислень. Наша робота є допомогою для розуміння складності та розв'язку ігор з нульовою сумою та може мати практичні застосування в різних галузях.

Висновки нашої роботи можуть сприяти розвитку стратегічного мислення та розумінню принципів взаємодії між гравцями в конфліктних ситуаціях.

2. ТЕОРЕТИЧНІ ВІДОМОСТІ

Нульова сума – це ситуація в теорії ігор, коли виграш однієї людини еквівалентний втраті іншої, тому чиста зміна багатства або вигоди дорівнює нулю. У грі з нульовою сумою може бути як два гравця, так і мільйони учасників. На фінансових ринках опціони та ф'ючерси є прикладами ігор з нульовою сумою, за винятком трансакційних витрат. Для кожної особи, яка виграє за контрактом, є контрагент, який програє.

Ігри з нульовою сумою зустрічаються в теорії ігор, але рідше, ніж ігри з ненульовою сумою. Покер та азартні ігри – популярні приклади ігор з нульовою сумою, оскільки сума виграних сумою одних гравців дорівнює загальним втратам інших. Такі ігри, як шахи та теніс, де є один переможець і переможений, також є іграми з нульовою сумою.

Зазначимо ключові моменти ігри з нульовою сумою:

- Гра з нульовою сумою – це ситуація, коли, якщо одна сторона програє, інша сторона виграє, а чиста зміна багатства дорівнює нулю.
- Ігри з нульовою сумою можуть включати лише двох гравців або мільйони учасників.

Наведемо приклад такої гри. Розглянемо гру в покер, в якій кожен гравець приходить на гру зі 100 доларами. Якщо гравців п'ять, то грошова сума для всіх п'яти гравців завжди становить 500 доларів. У будь-який момент гри певний гравець може мати більше 100 доларів, але тоді інший гравець повинен мати менше 100 доларів. Перемога одного гравця-це програш іншого гравця.

Для простоти викладу вивчатимемо гру двох гравців з нульовою сумою. Як уже зазначалося, гра — це сукупність правил, що описують сутність конфліктної ситуації та встановлюють:

1. вибір способу дій гравців на кожному етапі гри;

2. обсяг необхідної інформації, якою володіє кожен гравець у момент вибору свого способу дії;
3. плату кожного гравця після завершення будь-якого етапу гри.
4. Чистою стратегією гравця називається сукупність рекомендацій щодо ведення гри від початку до її завершення.
5. Гра називається скінченою, якщо в кожного гравця є скінчена кількість стратегій. У протилежному випадку гра є нескінченною.[1]

Нехай гра є скінченою, тоді результати рішень гравців можна виразити в грошовому еквіваленті або з допомогою інших цінностей, які збирається вигравати (придбавати) кожен гравець. Тобто для кожної комбінації вибраних гравцями чистих стратегій існує відповідна величина платежу.

У випадку парної гри ці платежі зручно подавати у вигляді платіжної матриці. Розв'язання гри полягає в знаходженні чистих стратегій для кожного гравця, які б максимізували виграш переможця та одночасно мінімізували б програш переможеного.

Однією із задач теорії гри є виявлення можливості певної рівноваги, що називається компромісом, яка найбільшою мірою задовольняє всіх учасників. Досить досконалою, в плані пошуку компромісного рішення, слід визнати теорію гри двох осіб з нульовою сумою, тобто такої парної гри, в якій виграш першого гравця завжди збігається з величиною програшу другого. В таких іграх, аналізуючи платіжну матрицю виграшів першого гравця, іноді можна знайти оптимальні чисті стратегії обох гравців.

Нехай відома матриця платежів парної гри з нульовою сумою, де елемент платіжної матриці A_{ij} — це виграш першого гравця, тобто сума, яку йому платить другий гравець (програш другого гравця) у випадку використання першим гравцем своєї чистої стратегії $S_i \rightarrow S$, а другим гравцем— своєї чистої стратегії $\Theta_j \rightarrow \Theta$.

Розв’язати гру — це означає знайти оптимальну стратегію для кожного гравця. Оптимальною стратегією гравця називається така стратегія, яка за багаторазового повторення гри забезпечує гравцеві максимально можливий середній виграш (або мінімально можливий середній програш).[2]

Рівністю за Нешем називають таку пару стратегій S_i та $Θ_j$, що одноособова зміна стратегії не покращує виграш / програш жодного з гравців.

Ключові моменти даного визначення такі:

- Рівновага Неша-це теорема про прийняття рішень в рамках теорії ігор, яка стверджує, що гравець може досягти бажаного результату, не відхиляючись від своєї початкової стратегії.
- У рівновазі Неша стратегія кожного гравця є оптимальною з урахуванням рішень інших гравців. Кожен гравець виграє, тому що кожен отримує бажаний результат.
- Дилема в'язня є поширеним прикладом теорії ігор, який адекватно демонструє ефект рівноваги Неша.
- Рівновага Неша часто обговорюється в поєднанні з домінуючою стратегією, яка говорить, що обрана стратегія суб'єкта призведе до кращих результатів з усіх можливих стратегій, які можуть бути використані, незалежно від стратегії, яку використовує противник.
- Рівновага Неша не завжди означає, що обрана найбільш оптимальна стратегія.[3]

Знаходити точки рівноваги Неша можна у дуже простий, але водночас довгий спосіб: потрібно для кожної пари стратегій S_i та $Θ_j$ перевіряти, що не існує пари (S_k; $Θ_j$) або пари (S_i; $Θ_k$), таких що їх результат у платіжній матриці був кращим для якогось з гравців.

Тобто, точка рівноваги має задовільняти таким умовам:

$$\forall k, S_k \neq S_i : A_{ij} \geq A_{kj}; (1)$$

$$\forall k, \theta_k \neq \theta_j : A_{ij} \leq A_{ik} (2)$$

При цьому для чистих стратегій не завжди можна знайти рівність за Нешем, хоча в той же час можна отримати й декілька таких точок.

Наприклад, якщо платіжна матриця виглядає $\begin{pmatrix} 2 & 2 \\ 1 & 1 \end{pmatrix}$, то маємо дві стратегії, що приводять до рівноваги: перший гравець обирає першу стратегію, так само як і другий гравець; або ж перший гравець обирає першу стратегію, а другий – другу.

Серед цікавих стратегій, що використовуються для розв'язання ігор, можна визначити домінуючу стратегію.

Стратегія X домінує в стратегії Y, якщо кожен запис для X більше або дорівнює відповідного запису для Y. У цьому випадку ми говоримо, що стратегія Y задомінована стратегією X.

Якщо стратегія X домінує над стратегією Y, ми можемо написати X≻Y.[4]

Говорячи більш математичною мовою можемо сказати, що:

$S_i \succ S_k \Leftrightarrow \forall j: A_{ij} \geq A_{kj}; \exists j: A_{ij} > A_{kj}$(3)

Відповідно для другого гравця домінуюча стратегія виглядає трішки інакше: $\theta_j \succ \theta_k \Leftrightarrow \forall i: A_{ij} \leq A_{ik}; \exists i: A_{ij} < A_{ik}$(4).

Так як задоміновані стратегії гране не вигідно у порівнянні зі стратегіями, що над ними домінують, то їх можна викреслити з платіжної матриці, після чого повторити пошук вже для новоутвореної системи.

Рівновагу Неша часто порівнюють з домінуючою стратегією, оскільки обидві вони є стратегіями теорії ігор. Рівновага Неша стверджує, що оптимальна стратегія для гравця полягає в тому, щоб дотримуватися своєї початкової стратегії, знаючи стратегію суперника, і що всі гравці дотримуються однієї стратегії.

Домінуюча стратегія стверджує, що обрана стратегія суб'єкта призведе до кращих результатів з усіх можливих стратегій, які можуть бути використані, незалежно від стратегії, яку використовує противник.

Важливо ще раз наголосити, що усі моделі теорії ігор працюють лише в тому випадку, якщо залучені гравці є "раціональними агентами", тобто це означає, що вони бажають конкретних результатів, діють у спробі вибрати найбільш оптимальний результат, враховують невизначеність у своїх рішеннях та реалістичні у своїх варіантах.

Обидва терміни схожі, але трохи відрізняються один від одного. Рівновага Неша говорить, що нічого не виграється, якщо хтось із гравців змінить свою стратегію, тоді як усі інші гравці зберігають свою стратегію. Домінуюча стратегія стверджує, що гравець вибере стратегію, яка призведе до найкращого результату, незалежно від стратегій, обраних іншими гравцями. Домінуюча стратегія може бути включена в рівновагу Неша, тоді як рівновага Неша може бути не найкращою стратегією в грі.

На жаль, обидві ці ідеї є не такими гарними як наступна. Ми можемо зрозуміти, що кожен гравець хоче збільшити свій виграш, або, як більш часто говорять про другого гравця у контексті ігор з нульовою сумою, зменшити свій програш. Тоді сідлова точка, так ще називають рівність за Нешем, так як жоден з гравців не бачить вигоди «злазити з сідла», тобто змінювати стратегію, є парою стратегій, що найбільш добре підходить до даного опису.

Для знаходження цієї пари стратегій використовують “принцип мінімакса”, сутністю якого є міркування, що супротивник зробить все для того, щоб перешкодити досягненню супротивником своєї цілі. Стратегію першого (другого) гравця називають оптимальною, якщо в разі її багаторазового застосування виграш (програш) першого (другого) гравця не зменшується (не збільшується), які б стратегії не застосовував супротивник.

Так як зазвичай платіжну матрицю будують від обличчя першого гравця, то прийнято розглядати, що платіжну матрицю виграшів він аналізує з позиції максимізації гарантованого виграшу(який насправді може бути й програшем), а

саме: для кожної своєї чистої стратегії S_i (i = 1,...,m) він визначає мінімальне значення виграшу $\alpha_i^+ = \min_{\theta_j \in \theta} A_{ij}$, та стратегії S_{i_0}, такої що

$$\alpha_{i_0}^+ = \alpha^+ = \max_{S_i \in S} \alpha_i^+ = \max_{S_i \in S} \min_{\theta_j \in \theta} A_{ij}.(5)$$

Число α^+ називається нижньою ціною гри, або макс-міном, а відповідна чиста стратегія S_{i_0} першого гравця називається макс-мінною.

Другий гравець, відповідно, ставить за мету мінімізацію свого гарантованого програшу (який в теорії може бути й виграшем). Для нього платіжна матриця рахується від'ємно, а тому для кожної чистої стратегії Θ_j він визначає величину максимального програшу $\beta_j^- = \max_{S_i \in S} A_{ij}$, та стратегії θ_{j_0}, такої що

$$\beta_{j_0}^- = \beta^- = \min_{\theta_j \in \theta} \beta_i^+ = \min_{\theta_j \in \theta} \max_{S_i \in S} A_{ij}.(6)$$

Число β^- називається верхньою ціною гри, або мінімаксом, а відповідна чиста стратегія θ_{j_0} другого гравця — мінімаксною.

Іноді дану сукупність називають єдиним загальним терміном "мінімаксні стратегії". Мінімаксні чисті стратегії S_{i_0} та θ_{j_0} є стійкими, тобто утворюють оптимальну пару стратегій, у тому випадку, коли нижня ціна гри дорівнює верхній. Тоді платіжна матриця F містить елемент $f_{i_0 j_0}$, що задовольняє умову:

$$A_{i_0 j_0} = \alpha^+ = \beta^-$$

Таким чином, якщо гра має сідлову точку, то чисті стратегії S_{i_0} та θ_{j_0} є оптимальними і тоді сукупність стратегії S_{i_0} та θ_{j_0} та ціна гри

$$V^* = A_{i_0 j_0} = \alpha^+ = \beta^-(7).$$

- утворюють розв'язок гри. Розв'язок гри має таку властивість: якщо один з гравців дотримується своєї оптимальної стратегії, то відхилятися від своєї оптимальної стратегії не вигідно для другого гравця.

Взагалі непогано було б помітити й довести, що якщо існує два розв'язки з чистими стратегіями, то має місце твердження: рівності за Нешем в одній грі мають однакову ціну гри.

І справді, розглянемо матрицю $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$.

1) Нехай за Нешем рівновага відбувається в парі стратегій (1,1) та (2,1) при тому, що $a \neq c$. Тоді за властивостями (1-2) маємо твердження:

$$\begin{cases} A_{11} \geq A_{21} \\ A_{21} \geq A_{11} \\ A_{11} \neq A_{21} \end{cases} \rightarrow \begin{cases} a \geq c \\ c \geq a \\ a \neq c \end{cases} \rightarrow \emptyset$$

Протиріччя.

Аналогічно доводиться $b = d$, $a = b$, $c = d$.

2) Нехай за Нешем рівновага відбувається в парі стратегій (1,1) та (2,2) при тому, що $a \neq d$. Тоді за властивостями(1-2) маємо твердження

$$\begin{cases} A_{11} \geq A_{21} \\ A_{21} \geq A_{22} \\ A_{22} \geq A_{12} \\ A_{12} \geq A_{11} \\ A_{11} \neq A_{21} \end{cases} \rightarrow \begin{cases} a \geq c \geq d \\ d \geq b \geq a \\ a \neq d \end{cases} \rightarrow \begin{cases} a \geq d \\ d \geq a \\ a \neq d \end{cases} \rightarrow \emptyset$$

Протиріччя. Отже, рівності за Нешем в одній грі мають однакову ціну гри.

Рівновага Неша важлива, оскільки вона допомагає гравцеві визначити найкращий виграш у ситуації, виходячи не лише з їхніх рішень, а й з інших залучених сторін. Баланс Неша може бути використаний у багатьох сферах життя, від бізнес-стратегій до продажу будинку, від війни до соціальних наук тощо.

Основним обмеженням рівноваги Неша є те, що він вимагає від гравця знання стратегії свого опонента. Рівновага Неша може настати лише в тому випадку, якщо гравець вирішить дотримуватися своєї поточної стратегії, якщо він знає стратегію свого суперника.

У більшості випадків, наприклад, на війні — будь-то військова війна або торги, - людина рідко знає стратегію противника або те, яким він хоче бачити

результат. На відміну від домінуючої стратегії, рівновага Неша не завжди призводить до найбільш оптимального результату. Це просто означає, що людина вибирає найкращу стратегію на основі інформації, яку вона має.

Більше того, у кількох іграх, зіграних з одними і тими ж супротивниками, рівновага Неша не враховує минулу поведінку, яка часто передбачає поведінку в майбутньому.

Рівновага Неша-це компонент теорії ігор, який стверджує, що гравець буде продовжувати дотримуватися обраної ним стратегії, знаючи стратегію свого супротивника, оскільки у нього немає стимулу змінювати курс. Рівновага Неша може бути застосована в різних ситуаціях реального життя, щоб визначити, яким буде найкращий виграш в тому чи іншому сценарії, грунтуючись на ваших рішеннях, а також знанні рішень вашого опонента.

На жаль, розглянуті нами методи працюють лише при наявності чистих стратегій та рівноваги Неша при них.

У загальному випадку значення ціни гри задовольняє умову: $\alpha^+ \leq V \leq \beta^-$, що має місце у випадках, коли гра не має сідлової точки, а мінімаксні чисті стратегії — не оптимальні. Це означає, що відшукання розв'язку гри у чистих стратегіях стає неможливим і кожна зі сторін може поліпшити свій стан шляхом багаторазового випадкового вибору певних своїх чистих стратегій з деяких підмножин (що належать множинам альтернативних чистих стратегій). Такі стратегії називаються змішаними.

Введемо таке означення: вектор, кожна з компонентів якого показує відносну частоту використання гравцем відповідної стратегії, називається змішаною стратегією даного гравця.

Нехай $P = (p_1;...p_m)$ — розподіл імовірності щодо вибору чистих стратегій першим гравцем у побудові своєї змішаної стратегії, яку надалі будемо

позначати через S_p. Тут p_i — імовірність вибору першим гравцем чистої стратегії S_i, і при цьому:

$$\sum_{i=1}^{m} p_i = 0; p_i \geq 0, i = 1, \ldots, m \ (8)$$

Аналогічно для другого гравця змішану стратегію позначимо через Q= (q_1;...q_n), а q_j — імовірність вибору другим гравцем чистої стратегії Θ_j за умови, що:

$$\sum_{j=1}^{n} q_i = 0; q_i \geq 0, j = 1, \ldots, n \ (9)$$

Серед змішаних стратегій першого та другого гравців відшукаємо оптимальні й позначимо їх через S_p та Θ_q відповідно. Тоді у загальному випадку оптимальним розв'язком гри буде сукупність (S_p; Θ_q).

Якщо перший гравець вибрав змішану стратегію S_p, а другий — змішану стратегію Θ_q то сподіваний виграш першого гравця (програш другого гравця) у ситуації багаторазового повторення гри становить величину V, таку що:

$$V = \sum_{i=1}^{m} \sum_{j=1}^{n} A_{ij} p_i q_i \ (10)$$

Методи пошуку оптимальних розв'язків гри базуються на таких положеннях, що були сформульовані Дж. фон Нейманом та О. Моргенштерном[5] як теореми:

1. кожна скінчена гра двох осіб з нульовою сумою має, принаймні, один (оптимальний) розв'язок, можливо у змішаних стратегіях;
2. якщо один з гравців застосовує свою оптимальну змішану стратегію, то його виграш дорівнює ціні гри незалежно від того, з якими ймовірностями (відносними частотами) другий гравець використовуватиме стратегії, що ввійшли в його оптимальну змішану стратегію.

Дж. фон Нейман та О. Моргенштерн в результаті довели так звану основну теорему теорії гри (теорема про мінімакс). Згідно з цією теоремою

$$V^* = \min_{Q \in \Delta_Q} \max_{P \in \Delta_P} \sum_{i=1}^{m} \sum_{j=1}^{n} A_{ij} p_i q_i \quad (11)$$

І при цьому зберігається нерівність:

$$\sum_{i=1}^{m} \sum_{j=1}^{n} A_{ij} p_i q_j^* \leq V^* \leq \sum_{i=1}^{m} \sum_{j=1}^{n} A_{ij} p_i^* q_j \quad (12)$$

Тобто ціна гри має верхню та нижню межі, а саме: $\alpha^+ \leq V^* \leq \beta^-$

Як бачимо з наведених результатів, значення ціни гри V* задає середньозважений виграш першого гравця (або середньозважений програш другого гравця) за багаторазового застосування оптимальних змішаних стратегій S_p^* та Θ_q^* гравців. Із співвідношення (11) випливає, що для пари стратегій S_p^* та Θ_q^* властиве таке: за багаторазового їх застосування виграш першого гравця не зменшується, а програш другого гравця не збільшується, які б свої стратегії не застосував супротивник.

Для розрахунку змішаної стратегії гравця використовують логіку, що можна розглянути з твердження (10). Якщо зафіксувати вибір другого гравця на стретегії j,то можна розглянути таку рівність $v = \sum_{i=1}^{m} A_{ij} p_i$

Взявши два таких набори стратегій для другого гравця як j та k маємо[6]:

$$\begin{cases} A_{1j} p_1 + A_{2j} p_2 + \cdots + A_{mj} p_m = v \\ A_{1k} p_1 + A_{2k} p_2 + \cdots + A_{mk} p_m = v \end{cases} \quad (13)$$

$$A_{1j} p_1 + A_{2j} p_2 + \cdots + A_{mj} p_m = A_{1k} p_1 + A_{2k} p_2 + \cdots + A_{mk} p_m$$

$$(A_{1j} - A_{1k}) p_1 + (A_{2j} - A_{2k}) p_2 + \cdots + (A_{mj} - A_{mk}) p_m = (14)$$

Отримали одне лінійне рівняння, а розглянувши всі можливі пари стратегій другого гравця отримаємо систему лінійних алгебраїчних рівнянь на m невідомих з C_n^2 рівняннями.

Для зручності розглянемо гру, де в кожного гравця по 2 стратегії, тобто платіжна матриця має вигляд $\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$

Тоді система рівнянь має вигляд: $\begin{cases} A_{11}p_1 + A_{21}p_2 = V \\ A_{12}p_1 + A_{22}p_2 = V \end{cases}(15)$

Так як у випадку з 2 стратегіями: $p_1 + p_2 = 1$, то можна виконати заміну $p_2 = 1 - p_1$, в результаті чого отримаємо:

$$A_{11}p_1 + A_{21}p_2 = A_{12}p_1 + A_{22}p_2$$

$$A_{11}p_1 + A_{21}(1 - p_1) = A_{12}p_1 + A_{22}(1 - p_1)$$

$$A_{11}p_1 - A_{21}p_1 - A_{12}p_1 + A_{22}p_1 = A_{22} - A_{21}$$

$$p_1 = \frac{A_{22} - A_{21}}{A_{11} - A_{12} - A_{21} + A_{22}} \quad (16)$$

Аналогічно можемо показати, що

$$q_1 = \frac{A_{22} - A_{12}}{A_{11} - A_{12} - A_{21} + A_{22}}, q_2 = 1 - q_1(17)$$

Підставивши у формулу отримуємо ціну гри:

$$V = A_{11}p_1 + A_{21}p_2 = A_{11}q_1 + A_{12}q_2(18)$$

Отже, результатом гри з матрицею 2х2 без сідлової точки будуть змішані стратегії $[p_1; 1 - p_1]$, $[q_1; 1 - q_1]$, та ціна гри V.3

Підіб'ємо підсумки:

1) розв'язати скінчену гру з нульовою сумою означає знайти оптимальні чисті стратегії гравців, якщо гра має сідлову точку, або ж знайти оптимальні змішані стратегії гравців S_p^* та Θ_q^*, а точніше — вектори чи сідлові точки, що відповідають теоремі про мінімакс, тобто умовам наведеним раніше, а також отримати ціну гри;

2) будь-яка парна матрична гра має розв'язок, якщо допускається використання змішаних стратегій;

3) гра без сідлової точки $\alpha^+ \leq V \leq \beta^-$) має розв'язок, можливо не єдиний, коли хоча б один з гравців використовує оптимальну змішану стратегію.

Отже, загальноприйняте поняття оптимальності розширюється за рахунок включення таких важливих елементів, як, наприклад, компромісне рішення, яке

задовольняє різні сторони конфлікту. Ця та інші особливості теорії гри дають змогу використовувати її методи для розв'язування різноманітних задач, що виникають в економічній науці та практиці. Дуже часто ці задачі допускають використання інших (неігрових) методів і моделей. Нижче буде наведено деякі приклади ігрового моделювання, а також показано розширені можливості зведення задач теорії гри до систем лінійних алгебраїчних рівнянь.

3. КОМП'ЮТЕРНА РЕАЛІЗАЦІЯ МЕТОДІВ ПОШУКУ РІВНОВАГИ НЕША З ЧИСТИМИ СТРАТЕГІЯМИ ДЛЯ ДОВІЛЬНОЇ СКІНЧЕННОЇ ГРИ З НУЛЬОВИМИ СУМАМИ ДЛЯ ДВОХ ГРАВЦІВ

Розглядаючи теоретичну інформацію було помічено, що ми маємо 3 основних способи знаходження рівноваги Неша для чистих стратегій двох гравців: це повний перебір, за домінуючими стратегіями та метод максиміна та мінімакса. Тому було вирішено розробити програмний код мовою Python, так як вона є сучасною мовою, що використовується зачасту у дата-майнингу, що в тих чи інших питаннях все таки стикається з теорією ігор та пошуком Нешевої рівноваги.

3.1 МЕТОД ПОВНОГО ПЕРЕБОРУ

У випадку повного перебору має місце простий цикл по рядкам *i* та стовпцям *j*, в якому значення вибраної клітинки (результат *i*-ої стратегії першого гравця та *j*- ої стратегії другого гравця) має бути не меншим за інші значення в стовпчику *j* та не більшим за інші значення у рядку *i*.

Приклад такої простої перевірки для однієї клітинки нашої платіжної матриці виглядає так:

```
pot_Eq = A[i][j]
        ch = True
        for i_J in range(n):
            if A[i_J][j] > pot_Eq:
                ch = False
```

```
            break
        if ch:
            for I_j in range(m):
                if A[i][I_j] < pot_Eq:
                    ch = False
                    break
        if ch:
            Dots.append((i,j)),
```

де pot_Eq – значення потенційної точки рівноваги Неша, а перевірка зупиняється, якщо хоч якась з перевірок є непідходящою за умовами. Якщо ж всі перевірки справджуються, то до результатів додається пара стратегій (i , j). Повний алгоритм даного методу знаходиться в додатку А.

3.2 МЕТОД ДОМІНУЮЧИХ СТРАТЕГІЙ

Для другого методу – за домінуючими стратегіями, маємо визначити такий алгоритм:

1. В матриці А будемо по черзі обирати ряд *i*.
2. Для обраної стратегії *i* перевіряємо домінантність над альтернативою k за таким критерієм (3)
3. Якщо стратегія *i* домінує над стратегією *k*, тоді видаляємо ряд *k* з матриці А утворюючи матрицю A^* та повертаємо матрицю A^* до кроку 1. Окрім того не забуваємо запам’ятати номер рядка *k*, так як при видаленні рядка нумерація стратегій зсувається.
4. Якщо жодна зі стратегій *i* не домінує, то схожим чином розглянемо стратегії другого гравця *j*.
5. Для обраної стратегії *j* перевіряємо домінантність над альтернативою k за таким критерієм (4)
6. Якщо стратегія *j* домінує над стратегією *k*, тоді видаляємо *k*-ий елемент кожного рядка матриці А утворюючи матрицю A^* та повертаємо

матрицю A^* до кроку 1. Окрім того не забуваємо запам'ятати номер стовпця *k*, так як при видаленні стовпця нумерація стратегій зсувається.

Наведемо частину реалізації даного алгоритму, в якій перевіряємо домінування стратегії *i* над стратегією *k* :

```
def check_domination_row(A:list, row:list):
    for i in range( len(A) ):
        for k in range ( len(A) ):
            if i != k:
                dom = True
                str_dom = False
                for j in range( len(A[i]) ):
                    if str_dom == False :
                        if A[i][j] > A[k][j]: str_dom = True
                    if A[i][j] < A[k][j]:
                        dom = False
                        break
                if dom and str_dom:
                    print ("Row ",row[k],'dominated by row ',row[i])
                    return (i,k)
    return False,
```

де dom – прапорець, що показує, що на даний момент ряд *i* домінує над рядом *k*, а str_dom = True вказує, що хоча б для однієї стратегії суперника результат при вибраній стратегії *i* переважає над результатом стратегії *k*.

```
   1  2  3   4
1  1  0  0  10
2 -1  0 -2   9
3  1  1  1   8
4 -2  0  0   7
Row  2 dominated by row  1
   1  2  3   4
1  1  0  0  10
3  1  1  1   8
4 -2  0  0   7
```

Рис. 1

На рисунку 1 зображено результат роботи шматка коду, в якому він перевірив домінацію ряда 1 над рядом 2, після чого його видалив, залишивши тільки 1-шу, 3-тю та 4-ту стратегії. Повний алгоритм даного методу знаходиться в додатку Б.

3.3 МЕТОД МАКСИМІНУ ТА МІНІМАКСУ

Для стратегії мінімаксу та максиміну треба застосувати алгоритм такого типу:

1. Створимо вектор min_R, в який зберемо найменше значення результату для кожної стратегії першого гравця.
2. Оберемо найбільший елемент maxmin_A серед min_R, для визначення найбільшого мінімального виграшу.
3. Створимо вектор max_C, в який зберемо найбільше значення результату для кожної стратегії другого гравця .
4. Оберемо найменший елемент minmax_A серед max_C, для визначення найменшого максимального програшу.
5. Якщо значення maxmin_A та minmax_A не збігаються, то немає рівноваги Неша при чистих стратегіях.
6. Якщо значення maxmin_A та minmax_A збігаються, то найкращими стратегіями для обох гравців будуть стратегії, що мають відповідне значення у векторах min_R та max_C.

В даному випадку розглянемо частину програмного коду, що знаходит наше значення minmax_A:

```
max_C = []
  for i in range(len(columns)):
    max_C.append(max(B[i]))
  minmax_A = min(max_C),
```

де B – транспонована матриця A. Повний алгоритм даного методу знаходиться в додатку В.

3.4 РЕЗУЛЬТАТИ ТА ПОРІВНЯННЯ РІЗНИХ МЕТОДІВ

Приклад 1.

Нехай дано матрицю A = $\begin{pmatrix} 3 & 1 \\ 2 & 0 \end{pmatrix}$, де перший гравець вибирає рядок, а другий – стовпчик.

Використавши наші алгоритми отримаємо такі результати:

1. Повним перебором – “*Player 1 need to play strategy 1 and Player 2 need play strategy 2*”
2. Домінуючі стратегії(рис.2) – “*Player 1 may play strategies: [1] and Player 2 may play strategies [2]*”
3. Максимін та мінімакс - “*Player 1 may play strategies: [1] and Player 2 may play strategies [2]*”

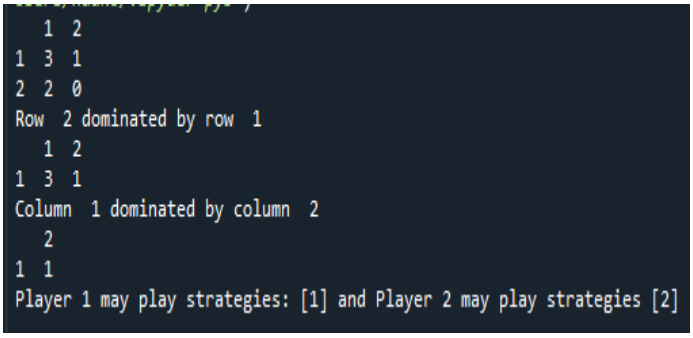

Рис.2

Приклад 2.

Нехай дано матрицю A = $\begin{pmatrix} -2 & 0 & 3 & 20 \\ 1 & -2 & -5 & -3 \\ 10 & -10 & -1 & 1 \\ 0 & 0 & 10 & 8 \end{pmatrix}$

Використавши наші алгоритми отримаємо такі результати:

1. Повним перебором – “*Player 1 need to play strategy 4 and Player 2 need play strategy 2*”
2. Домінуючі стратегії – “*Player 1 may play strategies: [1, 2, 3, 4] and Player 2 may play strategies [1, 2, 3]]*”
3. Максимін та мінімакс - “*Player 1 may play strategies: [4] and Player 2 may play strategies [2]*”

Тобто для даного прикладу ми вже бачимо, що при великій кількості стратегій методу домінуючих стратегій важко прибирати стратегії гравців, так як вірогідність того, що усі результати для якоїсь пари стратегій будуть в користь однієї, доволі малий.

Приклад 3.

Нехай дано матрицю A = $\begin{pmatrix} 1 & 0 & 0 & 10 \\ -1 & 0 & -2 & 9 \\ 1 & 1 & 1 & 8 \\ -2 & 0 & 0 & 7 \end{pmatrix}$

Використавши наші алгоритми отримаємо такі результати:

1. Повним перебором – “*Player 1 need to play strategy 3 and Player 2 need play strategy 1*

Player 1 need to play strategy 3 and Player 2 need play strategy 2

Player 1 need to play strategy 3 and Player 2 need play strategy 3"

2. Домінуючі стратегії – "*Player 1 may play strategies: [3] and Player 2 may play strategies [1, 2, 3]*"
3. Максимін та мінімакс - "*Player 1 need to play strategies [3] and Player 2 need play strategies [1, 2, 3]*"

Тобто для даного прикладу ми вже бачимо, що може виникнути ситуація при якій один з гравців має одну домінуючу стратегію, в той час як інший три. В даному випадку ми бачимо як рівні значення виграшу можуть утворювати більш ніж одну рівність Неша.

4. ПРИКЛАДИ РОЗВ'ЯЗАННЯ СКІНЧЕННОЇ ГРИ З НУЛЬОВИМИ СУМАМИ ДЛЯ МАТРИЦІ ГРИ 2Х2

Так як повний розв'язок не обмежується лише чистими стратегіями маємо розглянути й випадок зі змішаними стратегіями.

Зрозуміло, що у випадку, коли гра має сідлову точку, то обидва гравці використовуватимуть дану стратегію. Тому розглядаємо ситуацію, коли такої точки немає.

В даному випадку нам треба реалізувати алгоритм, що перетворює нашу матрицю А в СЛАР (14). Як вже було сказано такі великі системи мають і лінійно-залежні рядки, від яких нам треба позбуватись.

Задля спрощення розрахунків розглядатимемо лише ігри, де у кожного з гравців по 2 стратегії. Тоді, як ми вже зазначили, можемо знайти змішані стратегії для кожного з гравців за формулами (16-17).

Тоді загальний алгоритм розв'язку такої гри матиме вигляд:

1. Методом максиміну та мінімаксу перевіряємо наявність сідлових точок.
2. Якщо такі є, то отримаємо кращі стратегії для обох гравців, та ціну гри
3. Якщо немає, тоді обраховуємо вірогідність кожної стратегії для кожного гравця відповідно до згаданих вище формул.
4. Знайти ціну гри за формулою (18)

Повний алгоритм даного методу знаходиться в додатку Г.

4.1 РЕАЛІЗАЦІЯ РОЗВ'ЯЗАННЯ СКІНЧЕННОЇ ГРИ З НУЛЬОВИМИ СУМАМИ ДЛЯ МАТРИЦІ ГРИ 2X2

Приклад 1.

Нехай дано матрицю A = $\begin{pmatrix} 3 & 1 \\ 2 & 0 \end{pmatrix}$, де перший гравець вибирає рядок, а другий – стовпчик.

Як ми вже розглядали цю гру в пункті 3.4, вона має рівність Неша:

```
Player 1 need to play strategies [1] and Player 2 need play strategies [2]
Game price =  1.0
```

Приклад 2.

Нехай дано матрицю A = $\begin{pmatrix} 3 & -2 \\ 1 & 2 \end{pmatrix}$, де перший гравець вибирає рядок, а другий – стовпчик.

Тоді маємо такий розв'язок даної гри:

"*No equlibrium dots*

Player 1 need to play strategies with probability [0.16666666666666666, 0.8333333333333334] and Player 2 need play strategies with probability [0.6666666666666666 , 0.33333333333333337]

Game price = 1.3333333333333335"

```
No equlibrium dots
Player 1 need to play strategies with probability [  0.16666666666666666  ,  0.8333333333333334  ] and Player 2 need play strategies with probability [ 0.6666666666666666  ,  0.33333333333333337 ]
Game price =  1.3333333333333335
```

Тобто наш код може отримати розв'язок як для гри з чистими стратегіями, так і зі змішаними стратегіями гравців.

5. ВИСНОВКИ

У цій науковій статті ми досліджували скінченні ігри з нульовою сумою і розглядали питання знаходження рівноваги Неша для таких ігор. Наша основна мета полягала в отриманні розв'язків цих ігор, зокрема знаходженні чистих та змішаних стратегій, що є рівновагою Неша.

Для досягнення цієї мети ми розробили імплементацію алгоритмів, які дозволяють обчислювати рівновагу Неша у випадку, коли існують чисті стратегії, тобто методи повного перебору, домінуючих стратегій та мінімаксу, а також для випадку змішаних стратегій в матриці розміром 2x2. Наш розроблений код був успішно застосований до різних математичних прикладів, і ми отримали значні результати в знаходженні рівноваги Неша для цих конкретних випадків.

Відмічаємо, що алгоритм повного перебору більш ефективний та швидкісний для малої кількості стратегій гравців, проте є точним, на відміну від методу домінуючих стратегій, яка при збільшенні стратегій, може здебільшого лише відкидати зайві стратегії, що є задоміновані іншими. У той час використовувати стратегію мінімаксу та максиміну дуже ефективно та зручно, так як вона не збільшує час роботи в алгоритму так само сильно як і повний перебір.

Саме тому для повного розв’язання гри з нульовою сумою бажано поєднувати метод мінімаксу та перетворення платіжної матриці у систему лінійних рівнянь.

Отримані результати нашої роботи мають важливі наслідки і практичні застосування. Знаходження рівноваги Неша у скінченних іграх з нульовою сумою є ключовим поняттям в теорії ігор і може допомогти в розумінні рішень в конфліктних ситуаціях, економічних моделях, політичних стратегіях та інших сферах.

Наша робота внесла вагомий внесок у розвиток цієї галузі, зокрема шляхом розробки коду для знаходження рівноваги Неша у випадку наявності чистих стратегій та змішаних стратегій у матриці 2x2. Отримані результати можуть бути

використані як основа для подальшого дослідження складніших моделей та застосування в різних областях.

Загалом, наша робота розширює наше розуміння скінченних ігор з нульовою сумою та розвиток методів знаходження рівноваги Неша. Ми сподіваємося, що отримані результати стануть корисними для наукової спільноти та сприятимуть подальшому розвитку цієї важливої галузі.

6. СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Меньшиков И.С. Лекции по теории игр и экономическому моделированию. – М., МЗ Пресс, 2006.

2. Osborn, M.J. An introduction to game theory / M. J. Osborn. — N. Y. : Oxford University Press, 2004.

3. Bogomolnaia A., Le Breton M., Savvateev A., Weber S. Stability under unanimous consent, free mobility and core – International Journal of Game Theory. 2007

4. https://nordstromjf.github.io/IntroGameTheory/chapter-2.html

5. John von Neumann, Oskar Morgenstern - Theory of Games and Economic Behavior - Princeton University Press , 1944

6. Gilles D.B. Solutions to general non-zero-sum games. Contributions to the theory of games. IV (Kuhn H.W., Tucker A.W. eds.). Ann. Math. Studies, №40, − Princeton: Princeton Univ. Press, 1959

ДОДАТОК А

```
import pandas

def read_matrix ():
    A = []
    with open("matrix.txt") as f:
        lines = f.readlines()
    for line in lines:
        s = line.split()
        s1 = []
        for el in s:
            s1.append(float(el))
        A.append(s1)
    return A

def all_strategies (A:list):
    rows = []
    column = []
    for i in range( len(A) ):
        rows.append(i+1)
    for i in range ( len(A[0]) ):
        column.append(i+1)
    return rows, column

def guess_and_check (A:list, n:int , m:int):
    Dots = []
    for i in range(n):
        for j in range(m):
            pot_Eq = A[i][j]
            ch = True
            for i_J in range(n):
                if A[i_J][j] > pot_Eq:
                    ch = False
                    break
            if ch:
                for I_j in range(m):
                    if A[i][I_j] < pot_Eq:
                        ch = False
                        break
            if ch:
                Dots.append((i,j))
    return Dots

def Equlibrium_dots (A:list, rows:list, columns:list):
    Res = guess_and_check(A, len(rows), len(columns))
    if Res == []:
        print('No equlibrium dots')
    for el in Res:
        print('Player 1 need to play strategy',rows[el[0]],'and Player 2 need play
strategy',columns[el[1]])
        print('Game price = ',A[el[0]][el[1]])
```

```
Mas = read_matrix()
S1,S2 = all_strategies(Mas)
print(pandas.DataFrame(Mas,S1,S2))
Equlibrium_dots(Mas, S1, S2)
```

ДОДАТОК Б

```
import pandas

def read_matrix ():
    A = []
    with open("matrix.txt") as f:
        lines = f.readlines()
    for line in lines:
        s = line.split()
        s1 = []
        for el in s:
            s1.append(float(el))
        A.append(s1)
    return A

def all_strategies (A:list):
    rows = []
    column = []
    for i in range( len(A) ):
        rows.append(i+1)
    for i in range ( len(A[0]) ):
        column.append(i+1)
    return rows, column

def destroy_column(A:list, j:int):
    for i in range( len(A) ):
        A[i].pop(j)
    return A

def check_domination_row(A:list, row:list):
    for i in range( len(A) ):
        for k in range ( len(A) ):
            if i != k:
                dom = True
                str_dom = False
                for j in range( len(A[i]) ):
                    if str_dom == False :
                        if A[i][j] > A[k][j]:
                            str_dom = True
                    if A[i][j] < A[k][j]:
                        dom = False
                        break
                if dom == True and str_dom == True:
                    print ("Row ",row[k],'dominated by row ',row[i])
                    return (i,k)
    return False
```

```
def check_domination_column(A:list, column:list):
    m = len(column)
    for j in range( m ):
        for k in range ( m ):
            if j != k:
                dom = True
                str_dom = False
                for i in range( len(A) ):
                    if str_dom == False :
                        if A[i][j] < A[i][k]: str_dom = True
                    if A[i][j] > A[i][k]:
                        dom = False
                        break
                if dom and str_dom:
                    print ("Column ",column[k],'dominated by column ',column[j])
                    return (j,k)
    return False
def domination (A:list , rows:list, columns:list):
    row_dominating = check_domination_row(A, rows)
    if row_dominating == False:
        column_dominating = check_domination_column(A, columns)
        if column_dominating == False:
            return A, rows, columns
        else:
            B = destroy_column(A, column_dominating[1])
            columns.pop(column_dominating[1])
            print(pandas.DataFrame(B,rows,columns))
            B, rows, columns = domination(B, rows, columns)
            return B, rows, columns
    else:
        A.pop(row_dominating[1])
        rows.pop(row_dominating[1])
        print(pandas.DataFrame(A,rows,columns))
        B, rows, columns = domination(A, rows, columns)
        return B , rows, columns

def Domination_way(A:list, rows:list, columns:list):
    Res , BS1 , BS2 = domination(A,rows,columns)
    print('Player 1 may play strategies:',BS1,'and Player 2 may play strategies',BS2)

Mas = read_matrix()
S1,S2 = all_strategies(Mas)
print(pandas.DataFrame(Mas,S1,S2))
Domination_way(Mas, S1, S2)
```

ДОДАТОК В

```
import pandas

def read_matrix ():
A = []
    with open("matrix.txt") as f:
        lines = f.readlines()
    for line in lines:
        s = line.split()
        s1 = []
        for el in s:
            s1.append(float(el))
        A.append(s1)
    return A

def all_strategies (A:list):
    rows = []
    column = []
    for i in range( len(A) ):
        rows.append(i+1)
    for i in range ( len(A[0]) ):
        column.append(i+1)
    return rows, column

def maxmin(A:list, rows:list, columns:list):
    B=[]
    for j in range(len(columns)):
        C = []
        for i in range(len(rows)): C.append(A[i][j])
        B.append(C)
    min_R = []
    for i in range(len(rows)):
        min_R.append(min(A[i]))
    maxmin_A = max(min_R)
    max_C = []
    for i in range(len(columns)):
        max_C.append(max(B[i]))
    minmax_A = min(max_C)
    if minmax_A == maxmin_A:
        for i in reversed(range(len(rows))):
            if min_R[i] != minmax_A: rows.pop(i)
        for j in reversed(range(len(columns))):
            if max_C[j] != minmax_A: columns.pop(j)
        print('Player 1 need to play strategies',rows,'and Player 2 need play strategies',columns)
        print('Game price = ',minmax_A)
        return True
    else:
        print('No equlibrium dots')
        return False

Mas = read_matrix()
```

```
S1,S2 = all_strategies(Mas)
print(pandas.DataFrame(Mas,S1,S2))
maxmin(Mas, S1, S2)
```

ДОДАТОК Г

```
def read_matrix ():
A = []
    with open("matrix.txt") as f:
        lines = f.readlines()
    for line in lines:
        s = line.split()
        s1 = []
        for el in s:
            s1.append(float(el))
        A.append(s1)
    return A

def all_strategies (A:list):
    rows = []
    column = []
    for i in range( len(A) ):
        rows.append(i+1)
    for i in range ( len(A[0]) ):
        column.append(i+1)
    return rows, column

def maxmin(A:list, rows:list, columns:list):
    B=[]
    for j in range(len(columns)):
        C = []
        for i in range(len(rows)): C.append(A[i][j])
        B.append(C)
    min_R = []
    for i in range(len(rows)):
        min_R.append(min(A[i]))
    maxmin_A = max(min_R)
    max_C = []
    for i in range(len(columns)):
        max_C.append(max(B[i]))
    minmax_A = min(max_C)
    if minmax_A == maxmin_A:
        for i in reversed(range(len(rows))):
            if min_R[i] != minmax_A: rows.pop(i)
        for j in reversed(range(len(columns))):
            if max_C[j] != minmax_A: columns.pop(j)
        print('Player 1 need to play strategies',rows,'and Player 2 need play strategies',columns)
        print('Game price = ',minmax_A)
        return True
    else:
        print('No equlibrium dots')
        return False
```

```
def lineal_method(A:list, rows:list, columns:list):
    if len(rows) != 2 and len(columns) != 2:
        print("Method works only in 2x2 matrix")
        return None
    x = (A[1][1]-A[1][0])/(A[0][0]+A[1][1]-A[0][1]-A[1][0])
    y = (A[1][1]-A[0][1])/(A[0][0]+A[1][1]-A[0][1]-A[1][0])
    print('Player 1 need to play strategies with probability [ ',x,' , ',1-x,' ]  and Player 2 need play
strategies with probability [ ',y,' , ',1-y,' ]')
    res = A[0][0]*x + A[1][0]*(1-x)
    print ('Game price = ',res)

Mas = read_matrix()
S1,S2 = all_strategies(Mas)
print(pandas.DataFrame(Mas,S1,S2))
if maxmin(Mas,S1,S2) == False:
    lineal_method(Mas,S1,S2)
```