Міністерство освіти та науки України
Львівський національний університет імені Івана Франка
Факультет електроніки та комп’ютерних технологій
Кафедра Оптоелектроніки та комп’ютерних технологій

**Курсова робота**

на тему ” Скінченні динамічні ігри ”

Виконав:
студент групи ФЕІм-13
Назарко Я.Р.
Науковий керівник:
Катеринчук І.М.

Львів 2023

## АНОТАЦІЯ

Дана курсова робота присвячена вивченню динамічних скінченних стратегій і розв’язання пов’язаних завдань з розділу Теорії ігор. У роботі проведено дослідження і опис пов’язаних тем та аналіз методів розв’язання завдань Теорії ігор, а саме їх переваги і недоліки. Досліджено пов’язану літературу і розв’язано певні завдання теорії ігор. Для виконання роботи було використано інструментарій мови програмування Pythone.

## ABSTRACT

This research paper is dedicated to the study of dynamic finite strategies and solving related problems in the field of Game Theory. The paper explores and describes relevant topics and analyzes methods for solving Game Theory problems, including their advantages and disadvantages. It investigates relevant literature and solves specific Game Theory tasks. Python programming language was utilized as the primary tool for conducting the research.

# ЗМІСТ

КУРСОВА РОБОТА ..... 1

**ВСТУП** ..... 4

**1. ТЕОРІЯ ІГОР** ..... 6

1.1. ВИДИ ГРУП ІГОР ..... 6

1.1.1. СКІНЧЕННІ ДИНАМІЧНІ ІГРИ ..... 7

1.2. РІВНОВАГА НЕША ..... 9

1.3. МЕТОДИ РОЗВ’ЯЗАННЯ СКІНЧЕННИХ ДИНАМІЧНИХ ІГОР ..... 11

1.3.1 МЕТОД ЗВОРОТНОГО ІНДУКЦІЇ ..... 12

1.3.2. МЕТОД СТРАТЕГІЇ НА ОСНОВІ РІВНОВАГИ ..... 14

1.3.3. МЕТОД ДЕКОМПОЗИЦІЇ ..... 15

1.3.4. ЕВОЛЮЦІЙНІ МЕТОДИ ..... 16

**2. ІНСТРУМЕНТАРІЙ РОЗВ’ЯЗКУ** ..... 19

2.1. МОВА ПРОГРАМУВАННЯ PYTHON ..... 19

2.2. ФУНКЦІЯ FSOLVE ..... 20

**3. РОЗВ’ЯЗАННЯ ЗАВДАНЬ ТЕОРІЇ ІГОР** ..... 22

3.1. ПАРНЕ ЗМАГАННЯ З ВИГРАШАМИ ..... 22

3.2. РІВНЯННЯ ДИНАМІКИ РЕПЛІКАТОРІВ ..... 24

**ВИСНОВКИ** ..... 29

**СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ** ..... 30

ДОДАТОК А ..... 31

ДОДАТОК Б ..... 32

# ВСТУП

Теорія ігор є галуззю математики та економіки, яка вивчає раціональні стратегії та прийняття рішень в умовах конфлікту та взаємодії між різними учасниками, відомими як гравці. Ця теорія досліджує різні типи ігор, їхні правила, стратегії та можливі результати.

Одним із основних понять в теорії ігор є поняття гри. Гра визначається як ситуація, в якій кілька гравців взаємодіють між собою та вибирають свої дії залежно від дій інших гравців. Кожний гравець має свої цілі та переваги, і його результат залежить від взаємодії з іншими гравцями.

Одним із ключових понять в теорії ігор є стратегія. Стратегія визначається як план дій гравця, який він обирає для досягнення своїх цілей. Кожний гравець має свій набір стратегій, і вибір конкретної стратегії впливає на результат гри.

Теорія ігор вивчає різні типи ігор, включаючи ділові та економічні ситуації, політичні конфлікти, взаємодію між країнами та інші сфери життя. Вона досліджує такі поняття, як рівновага Неша, домінантна стратегія, корисливість, колективна діяльність та інші.

Однією з цікавих аплікацій теорії ігор є аналіз стратегій у вирішенні конфліктів. Вона дозволяє досліджувати оптимальні стратегії в умовах конфлікту та знаходити рішення, які приймаються з урахуванням інтересів різних сторін.

Також теорія ігор має важливі застосування в економіці, особливо в області прийняття рішень та вивченні взаємодії між підприємствами. Вона допомагає визначати оптимальні ціни, стратегії маркетингу та розробляти моделі, що враховують конкуренцію та співробітництво між фірмами.

Узагалі, теорія ігор є потужним інструментом для вивчення взаємодії, стратегічного мислення та прийняття рішень. Вона дозволяє аналізувати різні сценарії, прогнозувати результати та знаходити оптимальні рішення в умовах складних ігрових ситуацій. Ця галузь знаходить застосування в багатьох сферах

життя і надає цінний інструментарій для аналізу та вдосконалення взаємодії між різними учасниками.

Увага цієї курсової роботи хоч і покриває теорію ігор в цілому, але більш сконцентрована на скінченних динамічних іграх. Скінченні динамічні ігри є однією з важливих груп ігор в теорії ігор. Вони включають в себе ситуації, де гравці приймають рішення у послідовності в часі, враховуючи вплив своїх дій на майбутні результати гри.

# 1. ТЕОРІЯ ІГОР

## 1.1. Види груп ігор

У теорії ігор існують різні групи ігор, кожна з яких вивчає певні аспекти взаємодії між гравцями та рішення, які вони приймають. Основні групи ігор включають:

Статичні ігри: В цих іграх гравці приймають рішення одночасно або безпосередньо, без послідовності дій. Кожен гравець вибирає свою стратегію, і результат залежить від комбінації стратегій всіх гравців. Прикладом статичних ігор є класична гра "Посідання" або гра в камінь-ножиці-папір.

Динамічні ігри: У динамічних іграх гравці приймають рішення послідовно, залежно від попередніх дій і результатів гри. Ці ігри можуть мати деревоподібну структуру, де кожна гілка представляє послідовність рішень та можливих станів гри. Прикладом динамічної гри є "Шахи", де гравці приймають рішення на основі поточного стану дошки та можливих ходів опонента.

Кооперативні ігри: У цих іграх гравці співпрацюють між собою, формуючи коаліції та приймаючи спільні рішення. Вони можуть ділити прибутки або ресурси відповідно до угоди. Кооперативні ігри досліджують, як гравці співпрацюють та взаємодіють, щоб досягти спільних цілей. Прикладом такої гри є "Збирання врожаю", де гравці працюють разом, щоб зібрати якомога більше продукції.

Некооперативні ігри: В цих іграх гравці приймають рішення індивідуально, без спільного планування або співпраці. Кожен гравець діє власноруч, максимізуючи свою вигоду. Некооперативні ігри зосереджені на вивченні стратегій та рівноваги, де гравці досягають оптимальних результатів, зважаючи на дії опонентів. Прикладом такої гри є "Продаж на аукціоні", де гравці ставлять пропозиції та конкурують за предмети, намагаючись отримати їх за найкращою ціною.

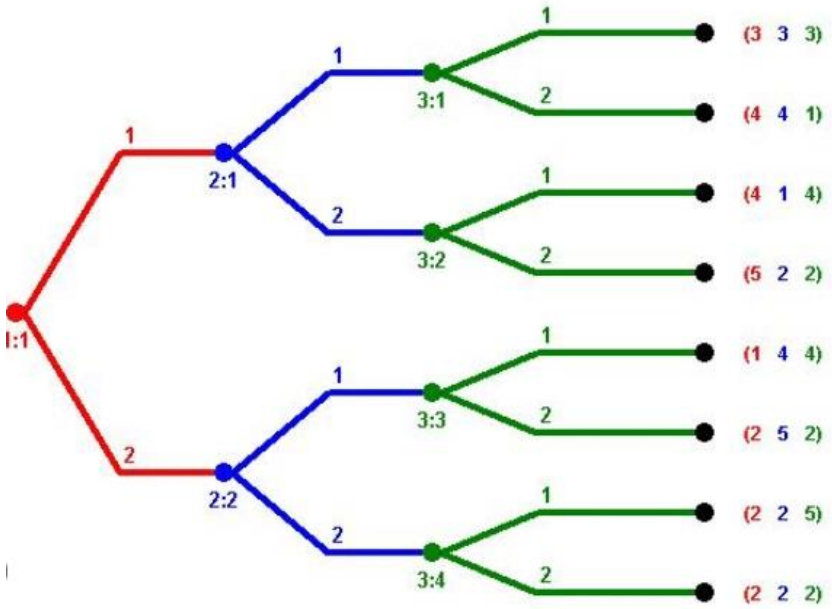
Кожна група ігор в теорії ігор вивчає різні аспекти стратегічного взаємодії між гравцями. Дослідження цих груп ігор допомагає розуміти рішення, які приймають гравці, та прогнозувати можливі результати гри.

### 1.1.1. Скінченні динамічні ігри

Скінченні динамічні ігри є розширенням поняття скінченних стратегічних ігор, де гравці приймають рішення впродовж декількох послідовних ходів. У таких іграх час відіграє важливу роль, оскільки рішення, прийняті гравцями на одному етапі гри, можуть впливати на наступні етапи.

Ключовими складовими скінченних динамічних ігор є:

1. Гравці: Гра включає двох або більше гравців, які впливають на хід гри своїми рішеннями.
2. Стратегії: Кожен гравець має вибір зі стратегій, які визначають їх дії на кожному етапі гри.
3. Послідовність ходів: Гра складається з послідовності ходів, де кожен гравець приймає рішення на кожному етапі.
4. Виграші: Існує функція виграшу, яка призначає кожній комбінації стратегій виграш або втрату для кожного гравця.




Розв'язання скінченних динамічних ігор включає в себе аналіз рівноваги, визначення оптимальних стратегій та прогнозування розвитку гри з

використанням різних методів, таких як метод зворотного індукції, метод стратегій на основі рівноваги та метод декомпозиції.

Ці інструменти дозволяють вивчити рішення, які гравці приймають в складних ситуаціях з довготривалими наслідками. Скінченні динамічні ігри застосовуються в різних областях, таких як економіка, біологія, політика та інші, для аналізу конфліктних ситуацій та прийняття раціональних рішень.

У таких іграх гравці вибирають свої дії в кожному кроці гри залежно від стану гри, який змінюється внаслідок їхніх дій та рішень. Кожний гравець має свою власну інформацію про стан гри, і його рішення в кожному кроці може змінити подальший розвиток подій.

У скінченних динамічних іграх важливим поняттям є поняття стратегії із сподіваним корисним наслідком (Strategy with Expected Utility, SEU). Це стратегія, де гравець приймає рішення на основі своїх очікувань щодо майбутніх наслідків та корисності кожної можливої дії.

В скінченних динамічних іграх використовуються різні підходи для аналізу і розв'язання. Один з них - це дерево гри (game tree), яке візуалізує послідовність рішень та можливі стани гри. Кожна гілка дерева представляє можливий варіант розвитку гри в залежності від рішень гравців.

Для розв'язання скінченних динамічних ігор використовуються різні концепції, такі як рівновага Неша (Nash equilibrium), яка визначає набір стратегій, де ні один гравець не може односторонньо поліпшити свій результат, враховуючи рішення інших гравців. Існування рівноваги Неша є важливим поняттям в аналізі скінченних динамічних ігор і дозволяє прогнозувати поведінку гравців та можливі результати гри.

Узагалі, скінченні динамічні ігри відкривають широкі можливості для дослідження та аналізу різних ситуацій, де гравці приймають рішення у послідовності в часі. Це дозволяє моделювати складні сценарії в різних галузях, таких як економіка, управління, політика та інші, та знаходити оптимальні стратегії для досягнення поставлених цілей.

## 1.2. Рівновага Неша

Рівновага Неша є поняттям з теорії ігор, яке описує стан, у якому жоден гравець не має мотивації змінити свою стратегію, за умови, що всі інші гравці діють також відповідно до своїх стратегій. Іншими словами, рівновага Неша є концепцією стійкості, де жодна сторона не бажає змінювати свої дії, знаючи дії своїх суперників.

У рівновазі Неша, кожен гравець вибирає свою стратегію оптимально, враховуючи стратегії інших гравців. Жоден гравець не має інцентиву відхилитися від своєї поточної стратегії, оскільки будь-яка зміна призведе до гіршого результату.

В рівновазі Неша, кожен гравець максимізує свій виграш, враховуючи дії інших гравців. Це призводить до стану, в якому неможливо отримати більше виграшу, змінивши свою стратегію при фіксованих стратегіях інших гравців.

Рівновага Неша може мати різні форми, такі як одностороння рівновага, де тільки один гравець має оптимальну стратегію, або багатостороння рівновага, де кілька гравців мають оптимальні стратегії. Рівновага Неша може бути домінуючою (коли немає альтернативних рішень, які забезпечують кращий виграш), або недомінуючою (коли існують альтернативні рішення зі збереженням гарантованого виграшу).

Важливо відзначити, що рівновага Неша може бути неєдиним рішенням у деяких ігрових ситуаціях, і можуть існувати інші типи рівноваг, такі як сумішні рівноваги, де гравці використовують ймовірнісні стратегії.

Для знаходження рівноваги Неша в скінчених динамічних іграх можуть використовуватись різні методи, такі як метод зворотної індукції, метод стратегій на основі рівноваги, метод декомпозиції та інші. Кожен з цих методів має свої переваги та недоліки і може бути ефективним в різних ситуаціях.

Розуміння рівноваги Неша дозволяє аналізувати стратегії гравців, прогнозувати їхні дії та розробляти оптимальні рішення в ігрових ситуаціях. Використання концепції рівноваги Неша може бути корисним у багатьох

галузях, таких як економіка, політика, соціологія, біологія та інші, де аналіз взаємодій між різними сторонами є важливим аспектом.

Застосування рівноваги Неша в еволюційних методах розв'язання скінченних динамічних ігор пов'язане з використанням принципів біологічної еволюції для визначення оптимальних стратегій.

Один з найпоширеніших методів, який базується на рівновазі Неша, це "Еволюційний алгоритм". Цей алгоритм моделює еволюційний процес, де різні стратегії представлені в популяції гравців, і вони піддаються взаємодії та еволюції протягом часу.

У цьому методі кожна стратегія отримує виграш на основі взаємодії з іншими стратегіями в популяції. Згідно з принципом рівноваги Неша, стратегії, які мають більший виграш, отримують перевагу та збільшують свою частку в популяції, тоді як менш успішні стратегії поступово виходять з обігу. Цей процес повторюється протягом декількох поколінь, що призводить до знаходження рівноважної стратегії або групи стратегій, які втілюють рівновагу Неша.

Еволюційні методи, такі як генетичні алгоритми, еволюційні стратегії або генетичне програмування, можуть використовувати рівновагу Неша як основу для оцінки та вдосконалення стратегій в процесі еволюції. Вони можуть здатні ефективно знаходити рівноважні стратегії в складних ігрових ситуаціях, де існує багато можливих стратегій та взаємодій гравців.

Переваги еволюційних методів полягають у їхній здатності до адаптації до змінюючих умов та великої кількості можливих стратегій. Вони також можуть бути застосовані до задач з неповною інформацією про інших гравців або невизначеністю в ігрових параметрах.

Однак, еволюційні методи також мають свої обмеження. Вони можуть бути обчислювально витратними, особливо при розгляді складних ігрових ситуацій або великих популяцій гравців. Крім того, вони можуть не завжди знаходити глобальну оптимальну рівновагу, а лише локальні рівноваги. Також,

результати еволюційних методів можуть бути чутливими до початкових умов та параметрів алгоритму.

Загалом, застосування рівноваги Неша в еволюційних методах дозволяє моделювати процеси адаптації та еволюції в складних ігрових середовищах. Вони можуть бути корисними для аналізу стратегій гравців та пошуку оптимальних рішень в різних областях, таких як економіка, біологія, соціологія та інші, де взаємодії та конфлікти між різними сторонами є важливими аспектами.

### 1.3. Методи розв’язання скінченних динамічних ігор

У теорії ігор існує кілька методів розв'язання скінченних динамічних ігор, які дозволяють знайти оптимальні рішення для гравців. Основні методи включають:

1. Метод зворотного індукції: Цей метод базується на принципі зворотного рахунку від кінця гри до початку. Гра розбивається на послідовні кроки, і для кожного кроку розв'язується підгра, яка складається зі станів гри, доступних на цьому кроці. Гравці приймають оптимальні рішення на кожному кроці, з урахуванням очікуваних виграшів у майбутніх станах гри.
2. Метод стратегій на основі рівноваги: Цей метод використовує поняття рівноваги в смислі Неша для знаходження оптимальних стратегій гравців. Він базується на припущенні, що гравці вибирають стратегії, які недоміновані, тобто вони не можуть досягти кращих результатів, змінивши свою стратегію, при умові, що інші гравці залишають свої стратегії незмінними. Цей метод вимагає розв'язання системи рівнянь, щоб знайти стратегії, які утворюють рівновагу.
3. Метод декомпозиції: Цей метод розбиває складну динамічну гру на менші підгри або модулі, що можуть бути розв'язані окремо. Кожна підгра розглядається окремо, і розв'язок підгри може бути

знайдений за допомогою інших методів, таких як метод зворотного індукції або метод стратегій на основі рівноваги. Результати підгр об'єднуються для отримання розв'язку всієї гри.

4. Інші методи: Крім основних методів, існують інші підходи до розв'язання скінченних динамічних ігор, такі як метод мультиагентних систем, метод апроксимаційного розв'язання і т.д. Ці методи зазвичай базуються на числових алгоритмах або еволюційних підходах для пошуку оптимальних рішень.

Кожен метод має свої переваги та обмеження, і вибір методу залежить від конкретної гри та поставленої задачі.

### 1.3.1 Метод зворотного індукції

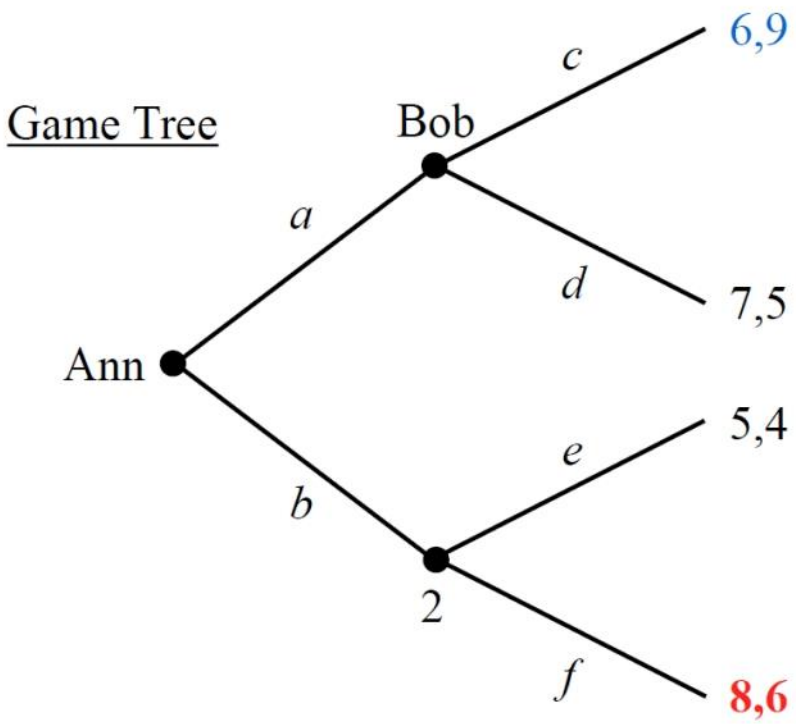
Метод зворотного індукції є одним з ключових методів розв'язання скінченних динамічних ігор і має свої переваги і недоліки. Основні з них наступні:

Переваги методу зворотного індукції:

1. Визначення оптимальних стратегій: Метод дозволяє визначити оптимальні стратегії для кожного гравця на кожному кроці гри. Це дозволяє гравцям приймати раціональні рішення, зважаючи на очікувані виграші в майбутніх станах гри.
2. Розкриття рівноваги: Метод зворотного індукції допомагає розкрити рівновагу в грі, тобто набір стратегій, при якому ні один гравець не має стимулу змінити свою стратегію відповідно до дій опонентів. Це дозволяє прогнозувати, як гравці поводитимуться в грі.
3. Прикладний аспект: Метод зворотного індукції є практично застосовним і знайшов широке застосування в багатьох галузях, таких як економіка, менеджмент, політика тощо. Він дозволяє моделювати різні ситуації взаємодії між гравцями та розраховувати оптимальні стратегії.

Недоліки методу зворотного індукції:

1. Обчислювальна складність: Метод зворотного індукції може бути обчислювально важким, особливо для складних ігор з великою кількістю гравців і станів. Розв'язок може вимагати значних обчислювальних ресурсів і часу.
2. Залежність від моделі: Метод зворотного індукції базується на моделі гри, яка може бути спрощеною або ідеалізованою версією реальної ситуації. Це може призвести до недостатньо точного опису дійсних стратегій гравців.
3. Відсутність динаміки: Метод зворотного індукції розглядає гру як послідовність ходів і не враховує можливості динамічної зміни стратегій гравців протягом гри. В деяких ситуаціях це може бути недостатнім для повного розуміння та моделювання гри.




Враховуючи ці переваги і недоліки, метод зворотного індукції є потужним інструментом для розв'язання скінченних динамічних ігор, але його застосування вимагає уважного аналізу та оцінки його обмежень у конкретних ситуаціях.

### 1.3.2. Метод стратегії на основі рівноваги

Метод стратегій на основі рівноваги є іншим популярним методом розв'язання скінченних динамічних ігор. Він також має свої переваги і недоліки:

Переваги методу стратегій на основі рівноваги:

1. Реалістичність: Метод стратегій на основі рівноваги дозволяє моделювати реалістичні взаємодії між гравцями. Він враховує той факт, що гравці приймають рішення, враховуючи дії опонентів і намагаються максимізувати свій виграш.
2. Універсальність: Метод стратегій на основі рівноваги може застосовуватися до різних типів ігор, включаючи кооперативні і некооперативні ігри. Він може використовуватися для аналізу як ситуацій співпраці, так і конфлікту між гравцями.
3. Ітераційний підхід: Метод стратегій на основі рівноваги може бути реалізований через ітераційний процес, де гравці поступово вдосконалюють свої стратегії, засновані на результатах попередніх ігор. Це дозволяє досягти стабільної рівноваги і покращити виграш для гравців.

Недоліки методу стратегій на основі рівноваги:

1. Залежність від початкових стратегій: Результати методу стратегій на основі рівноваги можуть значно залежати від початкових стратегій, які вибирають гравці. Невдалі початкові стратегії можуть призвести до отримання під-оптимальних рішень.
2. Обчислювальна складність: Розв'язок методом стратегій на основі рівноваги може бути обчислювально складним для складних ігор з багатьма гравцями і великою кількістю стратегій. Вимагається значна обчислювальна потужність для знаходження рівноваги Неша.
3. Неврахування динамічних змін: Метод стратегій на основі рівноваги не враховує можливості динамічних змін стратегій

гравців протягом гри. В деяких ситуаціях це може бути недостатнім для адекватного моделювання гри.

Необхідно уважно розглядати ці переваги і недоліки при застосуванні методу стратегій на основі рівноваги і враховувати особливості конкретної гри для досягнення найкращих результатів.

### 1.3.3. Метод декомпозиції

Метод декомпозиції є одним із підходів до розв'язання скінченних динамічних ігор, який включає розбиття гри на менші підзадачі. Ось деякі переваги і недоліки цього методу:

Переваги методу декомпозиції:

1. Скорочення обчислювальної складності: Декомпозиція гри дозволяє знизити обчислювальну складність, розбиваючи складну гру на менші підзадачі. Це полегшує аналіз і розв'язання гри, особливо у випадках, коли гра має велику кількість гравців і стратегій.
2. Фокус на локальних рішеннях: Метод декомпозиції дозволяє кожному гравцеві концентруватися на власних рішеннях, без потреби аналізувати всі можливі комбінації ходів інших гравців. Це спрощує процес прийняття рішень і дозволяє гравцям зосередитися на своїх власних інтересах.
3. Зручність для паралельних обчислень: Метод декомпозиції часто може бути легко розпаралелений, що дозволяє використовувати багатоядерні або розподілені обчислювальні системи для розв'язання гри. Це забезпечує ефективне використання ресурсів і скорочення часу обчислення.

Недоліки методу декомпозиції:

1. Втрата загальної інформації: При декомпозиції гри може виникнути втрата загальної інформації, оскільки гравці фокусуються на своїх локальних рішеннях. Це може призвести до під-оптимальних

стратегій, оскільки гравці не повністю усвідомлюють взаємозв'язок своїх рішень зі стратегіями інших гравців.

2. Проблема зближення до глобальної рівноваги: При декомпозиції гри може виникнути проблема зближення до глобальної рівноваги, оскільки гравці діють незалежно один від одного і можуть не здійснювати оптимальних ходів з точки зору гри в цілому. Це може призвести до ситуацій, коли гра не досягає глобальної оптимальності.
3. Складність координації: Коли гра розбита на менші підзадачі, може виникнути складність координації рішень між гравцями. Необхідно забезпечити механізми комунікації і співпраці між гравцями, щоб досягти колективної ефективності.

Важливо зрозуміти, що вибір методу розв'язання скінченних динамічних ігор залежить від конкретного контексту і особливостей гри. Кожен метод має свої переваги і недоліки, і їх використання повинно бути обгрунтовано і затверджено відповідними методами аналізу і розрахунку.

### 1.3.4. Еволюційні методи

Методи еволюційного підходу до розв'язання скінченних динамічних ігор базуються на принципах еволюції та вибору стратегій. Основна ідея полягає в тому, що гравці розглядаються як популяція, де стратегії вважаються генотипами, а виграші від ігор - фітнесом.

Ось кілька методів еволюційного підходу до розв'язання скінченних динамічних ігор:

Еволюційний алгоритм: Використовується модель еволюції для емуляції процесу природного відбору. Гравці розглядаються як індивіди, а стратегії - як їх генотипи. Процес еволюції включає генерацію нових стратегій, їх оцінку на основі виграшів та вибір стратегій для наступного покоління.

Генетичні алгоритми: Застосовуються ідеї генетики для розв'язання скінченних динамічних ігор. Генетичні алгоритми включають оператори схрещування (комбінування генотипів), мутації (зміни генотипів) та відбору

(вибір найкращих стратегій). Ці оператори використовуються для створення нових стратегій та покращення рівня стратегічної гри.

Еволюційна стратегія: В цьому методі гравці пристосовуються до оточуючого середовища шляхом зміни своїх стратегій. Кожен гравець підтримує певну кількість стратегій, які змінюються відповідно до виграшів. Частіше виграючі стратегії мають більше шансів бути використаними в наступній грі.

Переваги еволюційного підходу включають його здатність до самоорганізації, здатність до розв'язання некооперативних ігор та можливість розробки стратегій без потреби заздалегідь визначених правил гри. Однак, недоліками є високий обчислювальний обсяг та велика кількість експериментів, які потрібно провести для отримання стабільних результатів.

Ці методи еволюційного підходу до розв'язання скінченних динамічних ігор є ефективними інструментами для дослідження стратегічних ігор та знаходження оптимальних стратегій для гравців. Вибір конкретного методу залежить від особливостей гри та поставлених цілей дослідження.

#### 1.3.4.1 Еволюційна стійка стратегія

ESS (Evolutionarily Stable Strategy) - це концепція в теорії ігор, яка використовується для опису стратегій, які відповідають стійким еволюційним рівновагам в популяції.

У теорії ігор, популяція складається з різних стратегій, які конкурують між собою за ресурси або вигоди. ESS є такою стратегією, яка, якщо вона стане домінуючою в популяції, не може бути успішно замінена будь-якою іншою стратегією природним добором.

Формально, стратегія називається ESS, якщо вона задовольняє двом умовам:

1. Інваріантність: Якщо більшість популяції складається з особин, що використовують ESS, то жодна інша альтернативна стратегія не може успішно інвазувати цю популяцію.

2. Інвазійна успішність: Якщо популяція складається з певної кількості особин, що використовують ESS, то жодна інша альтернативна стратегія не може успішно інвадувати цю популяцію, навіть якщо її початкова кількість менша.

ESS відображає стійкі рівноваги, де жодна окрема стратегія не має можливості інвазії і перемогти в популяції, що використовує ESS. Це допомагає пояснити довгострокову стійкість певних стратегій в природніх популяціях.

ESS використовується для аналізу еволюційних процесів в теорії ігор, де добір відбувається через конкуренцію та розмноження. Визначення та ідентифікація ESS допомагає розуміти, які стратегії можуть вижити та стати стійкими в еволюційному процесі.

# 2. ІНСТРУМЕНТАРІЙ РОЗВ'ЯЗКУ

## 2.1. Мова програмування Python

Python - це високорівнева, інтерпретована, загального призначення мова програмування, яка була розроблена у 1991 році Гвідо ван Россумом. Назва мови походить від улюбленого комедійного шоу "Літня циркова шоу Монти Пайтона".

Основні характеристики мови Python:

1. Простота: Python прославляється своєю лаконічністю та зрозумілістю синтаксису, що робить його легким для вивчення та використання навіть початківцями.
2. Читабельність: Синтаксис Python спрямований на зрозумілість та легкість читання, що робить код більш зрозумілим і піддається супроводу та рефакторингу.
3. Універсальність: Python підтримує широкий спектр програмних парадигм, таких як процедурне програмування, об'єктно-орієнтоване програмування та функціональне програмування.
4. Розширюваність: Python дозволяє легко використовувати код, написаний на інших мовах програмування, і має багатий екосистему модулів та бібліотек, що розширюють його функціональність.
5. Крос-платформенність: Python підтримується на багатьох операційних системах, включаючи Windows, macOS та різні дистрибутиви Linux.

Переваги мови Python:

1. Простота вивчення: Python має простий і зрозумілий синтаксис, що допомагає новачкам швидко освоїти основи програмування.
2. Велика спільнота: Python має активну та розгалужену спільноту розробників, що забезпечує підтримку, документацію та розробку великої кількості сторонніх бібліотек та модулів.

3. Багатий екосистема: Python має широкий вибір бібліотек для різних задач, таких як наукові обчислення, веб-розробка, машинне навчання, обробка даних та багато інших.
4. Висока продуктивність: Хоча Python є інтерпретованою мовою, вона має ефективний збирач сміття та оптимізовані бібліотеки, що дозволяють досягти високої продуктивності у багатьох сценаріях використання.

Недоліки мови Python:

1. Швидкість виконання: У порівнянні з низькорівневими мовами, такими як C++ або Rust, Python може бути повільнішим у виконанні, особливо для великих обчислень або вимогливих за ресурсами додатків.
2. Обмежена підтримка мобільних платформ: Python не має такої широкої підтримки для мобільної розробки, як деякі інші мови програмування, такі як Java або Kotlin.
3. Обмежена мультипоточність: У зв'язку з глобальною блокуванням потоків, Python може мати обмежену підтримку паралельного виконання на багатопроцесорних системах.

Незважаючи на деякі недоліки, Python є однією з найпопулярніших мов програмування в світі, завдяки своїй простоті, екосистемі та широкому спектру застосувань. Вона широко використовується в таких галузях, як веб-розробка, наукові обчислення, штучний інтелект та аналіз даних.

## 2.2. Функція fsolve

fsolve - це функція з бібліотеки SciPy у мові програмування Python, яка використовується для чисельного розв'язання нелінійних рівнянь. Вона використовує ітераційний метод для знаходження числового розв'язку системи рівнянь.

fsolve приймає два обов'язкових аргументи: функцію, яка описує систему рівнянь, і початкове наближення для розв'язку. Функція повинна приймати вектор значень як вхідний параметр і повертати вектор значень рівнянь.

fsolve використовує числові методи, такі як метод Ньютона або метод Бройдена, для ітеративного покрокового наближення до розв'язку системи рівнянь. Вона продовжує ітерацію до тих пір, поки не досягне заданої точності або не досягне максимальної кількості ітерацій.

За допомогою fsolve можна чисельно розв'язувати різні задачі, такі як знаходження коренів рівнянь, знаходження рівноваги в системі рівнянь, апроксимація функцій і т.д.

Одним з основних переваг fsolve є його зручний та простий інтерфейс використання, який дозволяє швидко та ефективно розв'язувати нелінійні рівняння. Однак, варто враховувати, що успішність методу залежить від правильного початкового наближення та властивостей самої задачі.

Недоліком fsolve може бути його час виконання, особливо для складних систем рівнянь, а також можливість зациклення або недостатня точність при неправильному виборі початкового наближення.

Узагальнюючи, fsolve є потужним інструментом для чисельного розв'язання нелінійних рівнянь, але потребує розсудливого підходу та належного вибору початкових значень для досягнення точного та надійного розв'язку.

# 3. РОЗВ'ЯЗАННЯ ЗАВДАНЬ ТЕОРІЇ ІГОР

## 3.1. Парне змагання з виграшами

Розглянемо парне змагання з виграшами, наведеними в таблиці нижче (де a<b).

| | $A$ | $B$ |
|---|---|---|
| $A$ | $a-b,a-b$ | $2a,0$ |
| $B$ | $0,2a$ | $a,a$ |

Виведіть рівняння динаміки реплікатора для цієї гри та знайдіть усі фіксовані точки.

Щоб отримати рівняння реплікаторних динамік для даної гри, нам потрібно обчислити середні виграші для кожної стратегії та визначити пристосованість кожної стратегії в популяції.

Позначимо частоту стратегії A як x і частоту стратегії B як y.

Середній виграш для стратегії A дорівнює:

Виграш_A = x^2 * (a - b) + x * y * 2a

= x^2 * (a - b) + 2axy

Середній виграш для стратегії B дорівнює:

Виграш_B = y^2 * a + x * y * a

= y^2 * a + axy

Щоб обчислити пристосованість кожної стратегії, нормалізуємо середні виграші, поділивши їх на загальний середній виграш популяції:

Пристосованість_A = Виграш_A / (Виграш_A + Виграш_B)

= (x^2 * (a - b) + 2axy) / ((x^2 * (a - b) + 2axy) + (y^2 * a + axy))

Пристосованість_B = Виграш_B / (Виграш_A + Виграш_B)

= (y^2 * a + axy) / ((x^2 * (a - b) + 2axy) + (y^2 * a + axy))

Тепер ми можемо записати рівняння реплікаторних динамік:

dx/dt = x * (Пристосованість_A - середня_пристосованість)

= x * (Пристосованість_A - (x * Пристосованість_A + y * Пристосованість_B))

dy/dt = y * (Пристосованість_B - середня_пристосованість)

= y * (Пристосованість_B - (x * Пристосованість_A + y * Пристосованість_B))

де середня_пристосованість - це середня пристосованість популяції, яка обчислюється за формулою:

середня_пристосованість = x * Пристосованість_A + y * Пристосованість_B

Щоб знайти точки рівноваги реплікаторних динамік, встановимо dx/dt = 0 та dy/dt = 0:

dx/dt = 0:

x * (Пристосованість_A - (x * Пристосованість_A + y * Пристосованість_B)) = 0

dy/dt = 0:

y * (Пристосованість_B - (x * Пристосованість_A + y * Пристосованість_B)) = 0

Вирішення цих рівнянь дозволить знайти точки рівноваги реплікаторних динамік, які представляють стійкі стани популяції, де частоти стратегій більше не змінюються з часом.

```
Результати реплікаторних динамік:
Частота стратегії A: 0.0
Частота стратегії B: 1.0
```

```
def replicator_dynamics(x, y, a, b):
    payoff_A = x**2 * (a - b) + 2 * a * x * y
    payoff_B = y**2 * a + a * x * y
    total_payoff = payoff_A + payoff_B
    fitness_A = payoff_A / total_payoff
    fitness_B = payoff_B / total_payoff
    dx_dt = x * (fitness_A - (x * fitness_A + y * fitness_B))
    dy_dt = y * (fitness_B - (x * fitness_A + y * fitness_B))
    return dx_dt, dy_dt
# Початкові значення частот стратегій
x = 0.5
```

```
y = 0.5
# Параметри гри
a = 1
b = 2
# Кількість кроків реплікаторних динамік
num_steps = 100
# Ітерація кроків реплікаторних динамік
for step in range(num_steps):
    dx_dt, dy_dt = replicator_dynamics(x, y, a, b)
    x += dx_dt
    y += dy_dt
# Виведення результатів
print("Результати реплікаторних динамік:")
print(f"Частота стратегії A: {x}")
print(f"Частота стратегії B: {y}")
```

## 3.2. Рівняння динаміки реплікаторів

Популяція птахів розподіляється так, що на будь-якій даній території є тільки дві самки і два дерева, придатні для гніздування (T1 і T2). Якщо дві самки вибирають те саме місце для гніздування, потім кожна з них виховує по 2 потомства. Якщо вони вибрали різні місця, тоді вони більш вразливі для хижаків і здатні виростити лише 1 потомство кожна. Цю ситуацію можна змоделювати як парну конкурсну гру. Виведіть рівняння динаміки реплікатора та покажіть лише фіксовані точки, які відповідають ESS і є еволюційним кінцем балів.

Щоб отримати рівняння динаміки реплікаторів для даної гри, спочатку потрібно визначити виграші для кожної стратегії.

Позначимо стратегії наступним чином:

A1: Перша самка обирає дерево T1, а друга самка обирає дерево T1.

A2: Перша самка обирає дерево T1, а друга самка обирає дерево T2.

B1: Перша самка обирає дерево T2, а друга самка обирає дерево T1.

B2: Перша самка обирає дерево T2, а друга самка обирає дерево T2.

Виграші для кожної стратегії такі:

Виграш(A1) = 2 нащадки (якщо обидві самки обирають T1)

Виграш(A2) = 1 нащадок (якщо перша самка обирає T1, а друга самка обирає T2)

Виграш(B1) = 1 нащадок (якщо перша самка обирає T2, а друга самка обирає T1)

Виграш(B2) = 2 нащадки (якщо обидві самки обирають T2)

Тепер обчислимо середній виграш для кожної стратегії. Оскільки в популяції є лише дві самки, середній виграш для кожної стратегії дорівнює виграшу цієї стратегії.

Середній виграш(A1) = Виграш(A1) = 2 нащадки

Середній виграш(A2) = Виграш(A2) = 1 нащадок

Середній виграш(B1) = Виграш(B1) = 1 нащадок

Середній виграш(B2) = Виграш(B2) = 2 нащадки

Наступною кроком є обчислення придатності кожної стратегії, яка представляє собою нормалізований виграш, поділений на загальний середній виграш популяції:

Придатність(A1) = Виграш(A1) / Середній виграш = 2 нащадки / 2 нащадки = 1

Придатність(A2) = Виграш(A2) / Середній виграш = 1 нащадок / 2 нащадків = 0.5

Придатність(B1) = Виграш(B1) / Середній виграш = 1 нащадок / 2 нащадків = 0.5

Придатність(B2) = Виграш(B2) / Середній виграш = 2 нащадки / 2 нащадки = 1

Тепер ми можемо сформулювати рівняння динаміки реплікаторів для цієї гри:

dx/dt = x * (Fitness(A1) - Середня придатність)

dy/dt = y * (Fitness(A2) - Середня придатність)

де x - частота стратегії A1 в популяції, y - частота стратегії A2 в популяції,

Fitness(A1) - придатність стратегії A1, Fitness(A2) - придатність стратегії A2,

Середня придатність - середнє значення придатності в популяції, яке обчислюється як

Середня придатність = x * Fitness(A1) + y * Fitness(A2).

Тепер ми можемо знайти фіксовані точки цих рівнянь, встановивши dx/dt = 0 та dy/dt = 0:

dx/dt = 0:

x * (Fitness(A1) - Середня придатність) = 0

dy/dt = 0:

y * (Fitness(A2) - Середня придатність) = 0

Вирішення цих рівнянь дозволить нам знайти фіксовані точки рівнянь динаміки реплікаторів. Тільки ті фіксовані точки, які відповідають Еволюційно Стійким Стратегіям (ECC), є еволюційними кінцевими точками.

```
# Початкові значення для частот стратегій
x0 = 0.5
y0 = 0.5

# Виграші для кожної стратегії
A1 = 2
A2 = 1

# Розв'язання рівнянь динаміки реплікаторів
result = fsolve(dynamics, (x0, y0), args=(A1, A2))

# Отримання фіксованих точок
fixed_points = [(x, y) for x, y in zip(result[0], result[1])]

# Визначення еволюційно стійких стратегій
ESS = [point for point in fixed_points if point[0] > 0 and point[1] > 0]

# Виведення результатів
print("Фіксовані точки:")
for point in fixed_points:
    print(f"({point[0]:.2f}, {point[1]:.2f})")

print("\nЕволюційно стійкі стратегії (ECC):")
for ess in ESS:
    print(f"({ess[0]:.2f}, {ess[1]:.2f})")
```

import numpy as np

```
from scipy.optimize import fsolve
def fitness(A1, A2):
    total_fitness = A1 + A2
    return A1 / total_fitness, A2 / total_fitness

def dynamics(variables, *args):
    x, y = variables
    A1, A2 = args

    fitness_A1, fitness_A2 = fitness(A1, A2)
    average_fitness = x * fitness_A1 + y * fitness_A2

    dx_dt = x * (fitness_A1 - average_fitness)
    dy_dt = y * (fitness_A2 - average_fitness)

    return dx_dt, dy_dt
# Початкові значення для частот стратегій
x0 = 0.5
y0 = 0.5
# Виграші для кожної стратегії
A1 = 2
A2 = 1
# Розв'язання рівнянь динаміки реплікаторів
result = fsolve(dynamics, (x0, y0), args=(A1, A2))
# Отримання фіксованих точок
fixed_points = [(x, y) for x, y in zip(result[0], result[1])]
# Визначення еволюційно стійких стратегій
ESS = [point for point in fixed_points if point[0] > 0 and point[1] > 0]
# Виведення результатів
print("Фіксовані точки:")
for point in fixed_points:
```

```
    print(f"({point[0]:.2f}, {point[1]:.2f})")


print("\nЕволюційно стійкі стратегії (ЕСС):")
for ess in ESS:
    print(f"({ess[0]:.2f}, {ess[1]:.2f})")
```

Цей код використовує функцію fsolve з бібліотеки scipy.optimize для розв'язання рівнянь динаміки реплікаторів.

Програма виводить фіксовані точки і Еволюційно Стійкі Стратегії (ЕСС), які знайдені за допомогою розв'язку рівнянь динаміки реплікаторів.

# ВИСНОВКИ

В даній курсовій роботі проведено дослідження теорії ігор, а саме розділу скінченних динамічних ігор. Наведено методи розв’язання скінченних динамічних ігор такі як метод індукції, метод рівноваги та акцентовано увагу на еволюційних методах. Опрацьовано рівновагу Неша і описано еволюцію стійких стратегій. Створено код на мові програмування python який розв’язує перед тим проаналізовані завдання теорії ігор. Загалом робота відносно коротка, але це процес дослідження питання який має найбільшу цінність.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

[1] Game Theory: An Introduction by Steven Tadelis

[2] Evolutionary Games and Population Dynamics by Josef Hofbauer and Karl Sigmund

[3] Python Crash Course: A Hands-On, Project-Based Introduction to Programming by Eric Matthes

[4] Python for Data Analysis" by Wes McKinney

[5]Games and Economic Behavior

[6] Journal of Artificial Intelligence Research

[7] Games Theory Decision, interaction and evolution by James N. Webb

# ДОДАТОК А

```
def replicator_dynamics(x, y, a, b):
    payoff_A = x**2 * (a - b) + 2 * a * x * y
    payoff_B = y**2 * a + a * x * y
    total_payoff = payoff_A + payoff_B
    fitness_A = payoff_A / total_payoff
    fitness_B = payoff_B / total_payoff
    dx_dt = x * (fitness_A - (x * fitness_A + y * fitness_B))
    dy_dt = y * (fitness_B - (x * fitness_A + y * fitness_B))
    return dx_dt, dy_dt
# Початкові значення частот стратегій
x = 0.5
y = 0.5
# Параметри гри
a = 1
b = 2
# Кількість кроків реплікаторних динамік
num_steps = 100
# Ітерація кроків реплікаторних динамік
for step in range(num_steps):
    dx_dt, dy_dt = replicator_dynamics(x, y, a, b)
    x += dx_dt
    y += dy_dt
# Виведення результатів
print("Результати реплікаторних динамік:")
print(f"Частота стратегії A: {x}")
print(f"Частота стратегії B: {y}")
```

# ДОДАТОК Б

```
import numpy as np
from scipy.optimize import fsolve
def fitness(A1, A2):
    total_fitness = A1 + A2
    return A1 / total_fitness, A2 / total_fitness

def dynamics(variables, *args):
    x, y = variables
    A1, A2 = args

    fitness_A1, fitness_A2 = fitness(A1, A2)
    average_fitness = x * fitness_A1 + y * fitness_A2

    dx_dt = x * (fitness_A1 - average_fitness)
    dy_dt = y * (fitness_A2 - average_fitness)

    return dx_dt, dy_dt
# Початкові значення для частот стратегій
x0 = 0.5
y0 = 0.5
# Виграші для кожної стратегії
A1 = 2
A2 = 1
# Розв'язання рівнянь динаміки реплікаторів
result = fsolve(dynamics, (x0, y0), args=(A1, A2))
# Отримання фіксованих точок
fixed_points = [(x, y) for x, y in zip(result[0], result[1])]
# Визначення еволюційно стійких стратегій
ESS = [point for point in fixed_points if point[0] > 0 and point[1] > 0]
# Виведення результатів
```

```
print("Фіксовані точки:")
for point in fixed_points:
    print(f"({point[0]:.2f}, {point[1]:.2f})")

print("\nЕволюційно стійкі стратегії (ЕСС):")
for ess in ESS:
    print(f"({ess[0]:.2f}, {ess[1]:.2f})")
```