УДК: 004.8 https://doi.org/10.15407/jai2025.04.043

О. В. Ізвалов[1], **С. Д. Паращук**[2], **О. П. Бондар**[3]
[1,2,3]Економіко-технологічний інститут імені Роберта Ельворті, Україна
вул. Євгена Чикаленка, 3, м. Кропивницький, Кіровоградська обл., 25006
[1]alexey@globalgamejam.org
[2]sparashchuk@gmail.com
[3]bondarkla@ukr.net
[1]https://orcid.org/0000-0002-4935-7153
[2]https://orcid.org/0000-0002-8609-3206
[3]https://orcid.org/0000-0001-5877-5667

ВИКОРИСТАННЯ ГРИ БАШЕ ЯК МАЙДАНЧИКА ДЛЯ НАВЧАННЯ ОСНОВАМ МЕТОДІВ ТА СИСТЕМ ШТУЧНОГО ІНТЕЛЕКТУ

O. Izvalov[1], **S. Parashchuk**[2], **O. Bondar**[3]
[1,2,3]Robert Elvorti Economics and Technology Institute, Ukraine
Yevhena Chykalenka st., 3, Kropyvnytskyi, Kirovohradska region, 25006
[1]alexey@globalgamejam.org
[2]sparashchuk@gmail.com
[3]bondarkla@ukr.net
[1]https://orcid.org/0000-0002-4935-7153
[2]https://orcid.org/0000-0002-8609-3206
[3]https://orcid.org/0000-0001-5877-5667

USING BACHET'S GAME AS A PLAYGROUND FOR TEACHING THE FUNDAMENTALS OF ARTIFICIAL INTELLIGENCE METHODS AND SYSTEMS

Анотація. Узагальнено досвід викладання дисципліни «Методи та системи штучного інтелекту» для студентів другого курсу спеціальності «Комп'ютерні науки». Аналіз навчальних програм цієї дисципліни показав, що приклади, які використовуються для введення важливих понять ШІ, є різноманітними та розпорошеними. Ми представили варіант гри Баше на віднімання як задачу, яку студенти мають розв'язати, розробляючи ШІ-агентів різних типів. Розроблено симуляційне середовище, яке слугує платформою для інтелектуальних агентів, здатних грати в різні варіанти гри Баше. Технології, що використовувалися для створення інтелектуальних агентів для гри в цю гру, включали: пошук оптимального рішення вручну та попереднє програмування агента; запуск пошуку на графі ігрових позицій; Q-навчання; генетичне програмування; нейронні мережі. Під час вивчення кожної технології студенти розробляли агентів, які повинні були вміти грати в гру Баше, змагатися з агентами, створеними іншими студентами, та адаптуватися до зміни правил гри. Таким чином, опанування цих технологій на спільному прикладі забезпечило глибоке розуміння концепцій, моделей, підходів та систем ШІ, а також їхньої внутрішньої роботи, можливостей та обмежень. Питання, які виникають у класі під час обговорення результатів, спонукають до подальших досліджень і підвищують активність та залученість студентів до вивчення моделей і систем штучного інтелекту. Гра Баше є гарним прикладом задачі, яка перекидає місток між різними галузями ШІ завдяки своїй простоті для розуміння, можливості побудови ментальної моделі та позиційного графу за допомогою ручки та паперу, змагальності та гнучкості. Змагання ШІ-агентів, створених студентами, та аналіз реакції агентів на зміну правил гри забезпечили сильний інтерес студентів до наступних, більш просунутих концепцій ШІ та подальших елементів навчальної програми.

Ключові слова: генетичні алгоритми, нейронні мережі, навчання бещ вчителя, інтелектуальні агенти, освіта.

Abstract. The paper summarizes the experience of teaching the "Methods and Systems of Artificial Intelligence" course to second-year Computer Science students. An analysis of the curricula for this discipline showed that the examples used to introduce important AI concepts are diverse and fragmented. We introduced a variant of Bachet's subtraction game as a problem for students to solve by developing various types of AI agents. A simulation environment was developed to serve as a platform for intelligent agents capable of playing different variants of Bachet's game. The techniques used to create intelligent agents for this game included: manual search for the optimal solution and hard-coding the agent; state-space search (graph search); Q-learning; genetic programming; and neural networks. While studying each technique, students developed agents required to play Bachet's game, compete against agents created by other students, and adapt to changes in the game rules. By applying these distinct technologies to a single, consistent task, students gain

a unique opportunity to benchmark performance and understand the specific advantages and limitations of each method—comparing the exactness of exhaustive search against the generalization capabilities of neural networks. Questions arising during classroom discussions of the results stimulate further research and increase student engagement and activity in studying artificial intelligence models and systems. Bachet's game serves as an excellent example of a problem that bridges different areas of AI due to its ease of understanding, the ability to construct a mental model and a game state graph using pen and paper, its competitive nature, and its flexibility. The competitive element, combined with the analytical challenge of adapting to changing game mechanics, significantly boosts student engagement and stimulates lively classroom discussions, bridging the gap between theoretical models and practical system implementation, which gives a basis for understanding the more advanced concepts.

Keywords: genetic algorithms, neural networks, unsupervised learning, intelligent agents, education.

Вступ

Зміст навчальної програми зі спеціальності 122 «Комп'ютерні науки» в Україні стандартизовано Міністерством освіти і науки [1]. Цей стандарт містить програмні результати навчання бакалаврів з комп'ютерних наук та їхні компетентності. Одним з програмних результатів (ПР4 у Стандарті) є «*Використовувати методи обчислювального інтелекту, машинного навчання, нейромережевої та нечіткої обробки даних, генетичного та еволюційного програмування для розв'язання задач розпізнавання, прогнозування, класифікації, ідентифікації об'єктів керування тощо*». Цей програмний результат навчання пов'язаний зі спеціальною компетентністю (СК2 у Стандарті): «*Здатність до виявлення статистичних закономірностей недетермінованих явищ, застосування методів обчислювального інтелекту, зокрема статистичної, нейромережевої та нечіткої обробки даних, методів машинного навчання та генетичного програмування тощо*». Для реалізації вищезазначених компетентностей та результату навчання типовий навчальний план спеціальності «Комп'ютерні науки» включає принаймні одну навчальну дисципліну зі штучного інтелекту (ШІ). Найчастіше вона має назву «Методи та системи штучного інтелекту». Ця дисципліна має такі передумови, як «Дискретна математика», «Програмування», «Теорія ймовірностей».

Забезпечуючи однакові очікувані результати навчання та формуючи однакові компетентності, дисципліни українських вищих навчальних закладів, пов'язані зі ШІ, дуже різноманітні за своєю структурою. Отже, виникає питання, які концепції ШІ та які завдання слід пропонувати студентам на початку курсу, щоб забезпечити сильну зацікавленість і подальше глибоке розуміння методів і систем ШІ.

Постановка проблеми

Враховуючи часові обмеження: 36 лекційних годин та 18 лабораторних годин, ми повинні були розробити теми навчальної дисципліни «Методи та системи ШІ», які б відповідали стандарту Вищої освіти України бакалаврського рівня за спеціальністю «Комп'ютерні науки», забезпечували високу активність студентів, плавне введення в сферу ШІ та створювали підґрунтя для глибокого розуміння та вивчення подальших, більш просунутих концепцій ШІ в рамках курсу.

Аналіз сучасних досліджень і публікацій

У роботі [2] було проведено комплексний аналіз освітніх програм зі ШІ провідних університетів світу, визначено поточний стан навчальних планів зі ШІ та напрями їхнього вдосконалення. Згідно з результатами дослідження, дизайн курсу повинен мати акцент на практичних знаннях і навичках, починаючи з базових навичок програмування, щоб розуміти і застосовувати новітні технології ШІ для вирішення реальних задач.

Процес розробки та обґрунтування вибору компонентів для навчальної програмі з ШІ описано в [3]. Область знань зі ШІ, на думку авторів, повинна включати неінформований та інформований пошук по графах для вирішення задач, а також включати ігри як практичні приклади.

Викладання ШІ учням середніх шкіл та розробка відповідних навчальних програм і планів також викликає інтерес [4]. Метою впровадження ШІ було надання учням можливості стати вдумливими, творчими та відповідальними щодо вирішення проблем,

які можуть використовувати ШІ як інструмент для сталого розвитку.

Під час підготовки змісту дисципліни «Методи та системи штучного інтелекту» в Економіко-технологічному інституті імені Роберта Ельворті ми також проаналізували понад 20 навчальних програм українських університетів і виявили, що приклади, які використовуються в якості завдань у курсі ШІ, суттєво відрізняються. Наприклад: аналіз даних з використанням Python та NumPy [5], кластерний аналіз з використанням GNU Octave [6], нечітка логіка [7], нейронні мережі для апроксимації функцій з використанням NeurophStudio for Windows [8], системи штучного інтелекту на основі формальної логіки [9], розв'язування задач з використанням логічної мови програмування Пролог [10], інтелектуальні агенти [11] та пошук в графах [12]. Певним чином варіюється і назва відповідної дисципліни: від «Методи та системи штучного інтелекту» (найпоширеніша назва) до «Системи штучного інтелекту» або «Методи та засоби штучного інтелекту» чи «Методи та технології обчислювального інтелекту». Незважаючи на різноманітність, всі ці підходи відповідають вимогам Державного стандарту [1].

Ми провели аналіз відповідних курсів інших країн і в них задачі пошуку на графі є найпоширенішою відправною точкою. Наприклад: у курсах «Вступ до ШІ» в університеті Берклі [13], «Штучний інтелект» в Массачусетському технологічному інституті [14] або «Штучний інтелект: Принципи і методи» у Стенфордському університеті [15]. Для практичних прикладів застосування задач пошуку на графах використовуються різноманітні ігри.

Найбільш відомі підручники та лабораторні практикуми, що використовуються в дисципліні, починаються або з задач пошуку графів [16, 17], або з генетичних алгоритмів (ГА) [18, 19]. ГА в цьому випадку використовуються для оптимізації структури графів або пошуку маршрутів.

Мета дослідження

У цій статті узагальнено наш досвід та висновки щодо розробки початкової частини курсу «Методи та системи штучного інтелекту» для студентів бакалаврату комп'ютерних наук, що сприятиме ґрунтовному, всебічному та наскрізному ознайомленню з методами та системами штучного інтелекту. Ми обрали задачу, яка може бути розв'язана різними підходами до ШІ і, таким чином, продемонструвати студентам внутрішню роботу цих підходів. Ми пропонуємо розпочати курс «Методи та системи штучного інтелекту» з розробки інтелектуальних агентів для гри Баше. Для цього можна залучити просте програмування з використанням умовних операторів, а також пошуку по графах, Q-навчання, генетичні алгоритми та нейронні мережі. Оскільки проблема, що вирішується, фактично однакова для всіх цих методів, і різні агенти можуть змагатися один з одним, наш підхід забезпечує міцний фундамент для розуміння та самостійності навчальної діяльності студентів.

Впровадження гри Баше у курс «Методи та системи ШІ»

Дисципліна «Методи та системи штучного інтелекту» вивчається в Економіко-технологічному інституті імені Роберта Ельворті студентами другого курсу спеціальності «Комп'ютерні науки». Метою навчальної дисципліни є вивчення теоретичних основ ШІ та отримання початкового досвіду розробки систем, що реалізують технології ШІ. Дисципліна відповідає компетентностям та забезпечує досягнення результатів навчання, визначених Державним освітнім стандартом. Навчальний план включає 36 лекційних годин та 18 годин лабораторних занять.

Розробляючи структуру курсу в нашому інституті, ми прагнули знайти зразок, який буде зрозумілим для студентів, легко впроваджуваним, із наочним внутрішнім принципом роботи, і такий, що може стати майданчиком для студентських експериментів з різними методами ШІ. Тому в нашому інституті в перший день курсу

студенти знайомляться з грою Баше. Ця гра добре спрацювала як майданчик для проєктування ШІ-агентів, що реалізують різні технології, і сприяла розумінню студентами їхніх внутрішніх принципів.

Гра Баше - це гра в числа, яку описав Клод Гаспар Баше де Мезір'як на початку 17 століття [20]. Витоки цієї гри можна простежити до Леонардо Фібоначчі (13 століття). Гра розрахована на двох гравців, які роблять свої ходи послідовно. Її оригінальне формулювання таке: "*Два гравці, починаючи з 0, додають до нього цілі числа, які не перевищують 10. Перший гравець, який досягне 100, виграє*".

Гра стала відомим поняттям розважальної математики і часто формулюється у вигляді гри на віднімання [21]. Початкова позиція гри описується цілим невід'ємним числом N, яке представляє кількість камінців у купі. Гравці можуть брати довільну кількість камінців з купки, від 1 до k (включно). Останній гравець, який зробить хід, виграє. По суті, це та сама гра, що описана Баше, тільки додавання замінено на віднімання, задля спрощення арифметичного аналізу ендшпілів гри. Ми також запропонували формулювання гри на віднімання для наших студентів.

Гра Баше є покроковою неупередженою грою з досконалою інформацією. Це дозволяє побудувати граф гри для аналізу. Враховуючи, що кожна позиція описується лише одним числом, граф гри набагато простіший, ніж граф гри в хрестики-нулики або шахи - ігор, які часто використовуються як ігрові майданчики для навчання методам ШІ.

Переваги гри Баше як навчального прикладу полягають у тому, що її можна легко оцінити подумки, на дошці або просто за допомогою ручки та паперу, а потім ідеї, які народжуються під час аналізу проблеми, можуть бути реалізовані в розробленому програмному забезпеченні. Крім того, навіть незначні зміни правил у порівнянні з варіантом, описаним Клодом Гаспаром Баше, призводять до цікавих і несподіваних стратегій.

На першому занятті з дисципліни «Методи та системи штучного інтелекту» студентам пояснюють правила віднімання варіанту гри Баше з k=3 і пропонують зіграти кілька партій в парах, починаючи з N=15. Це відносно невелике число забезпечує швидке завершення гри.

Спочатку студенти помічають, що гравець, який залишає супернику 4 камінці, виграє. Адже на будь-який можливий хід суперника (1, 2 або 3 камінці з купки) можна відповісти виграшним ходом гравця (взяти 3, 2 або 1 камінчик відповідно).

Пізніше помічається наступна особлива позиція, яка має N=8. Якщо залишити таку позицію для суперника, то після будь-якого його ходу можна залишити супернику вже описану позицію N=4. І, нарешті, студенти приходять до думки, що виграшна стратегія гри може бути описана наступним чином: «*Спробуйте залишити наступному гравцеві 12, 8, 4 або 0 камінців після свого ходу*».

Сформулювавши стратегію, студентам пропонується реалізувати її у вигляді інтелектуального агента, здатного грати в цю гру.

Симуляційне середовище для експериментів з інтелектуальними агентами для гри у гру Баше та її варіації

Для тестування агентів студентам надається тестове середовище, розроблене для імітації турнірів між парами або групами різних агентів.

Можливості середовища включають в себе:

– створення необхідної кількості агентів заданих типів;

– визначення правила гри;

– запуск m-кратного кругового турніру між агентами;

– симуляція турнірної партії (рис.1);

– візуалізація послідовності ходів у певній грі;

– ведення статистики результатів агентів у кожній грі та в турнірі в цілому;

– представлення результатів агентів у вигляді часової діаграми або таблиці;

– видалення деяких агентів з турніру або додавання нових.

Інтерфейс взаємодії між середовищем та агентом дозволяє обмінюватися даними (рис. 2). На початку кожної гри агент

отримує інформацію про правила гри.

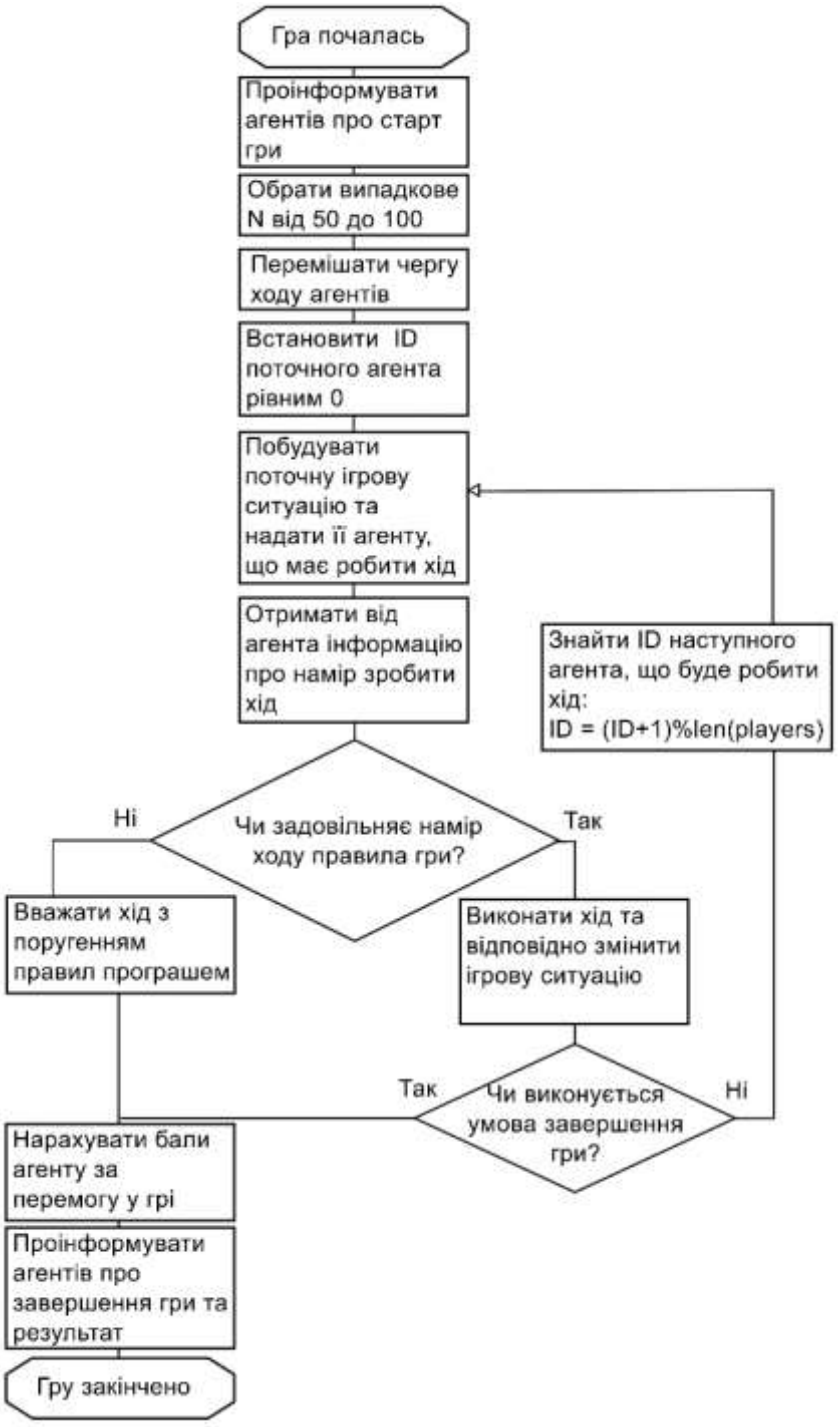

Рис. 1. Блок-схема гри між агентами

Правила не є жорстко закодованими в рушії симулятора, вони можуть бути модифіковані для більшої гнучкості. Правила визначають наступні аспекти гри:

– кількість гравців, які роблять ходи у грі (оригінальні правила розраховані на 2-х гравців, але симулятор здатен імітувати ігри між довільною кількістю гравців у одній грі);

– дозволені ходи (це може бути не просто інтервал цілих чисел [1 ... k], а будь-який набір цілих чисел, і зміна цього параметра може призвести до різних стратегій перемоги);

– умова перемоги (в оригінальних правилах перемагає гравець, який зробив хід останнім, проте існує варіант мізерної гри, де гравець, який зробив хід останнім,

програє); у випадку, якщо в одній партії грає більше 2-х гравців, переможцями вважаються всі гравці, які не програли, і вони отримують переможні бали;

– обмеження на повторення ходів (початкові правила не містили жодних обмежень, проте гра кардинально змінюється і стає набагато складнішою, коли забороняється повторювати один або два безпосередньо попередніх ходи).

Рис. 2. Інтерфейс обміну даними між симулятором та агентами

Симулятор інформує агента, який має чергу ходу, про поточну ігрову ситуацію, надсилаючи йому відповідний об'єкт з даними. Цей об'єкт містить число N, яке описує доступну на даний момент кількість камінців, а також може містити список заборонених ходів (якщо варіант гри передбачає обмеження на повторення ходів).

Агент отримує об'єкт даних від симулятора, аналізує його та формує власний об'єкт даних, який описує його власний намір ходу. Такий об'єкт містить число n камінців, які він має намір взяти з купки. Оскільки система <Симулятор-Агенти> є відкритою, а дизайн агентів доступний для третіх осіб (наприклад, студентів), симулятор виконує валідацію намірів ходу, отриманих від агента, перш ніж виконати хід. Валідний об'єкт наміру ходу повинен містити ціле число, це число повинно бути серед дозволених ходів і не перевищувати поточну кількість камінців у купі, N. Якщо агент подає некоректний хід, це автоматично вважається поразкою і поточна гра завершується.

Додатковим заходом для забезпечення коректного перебігу симуляції навіть у випадку помилок у програмуванні агентів є відстеження часу. Якщо агент реагує на свій хід довше, ніж заданий час (100 мс), агенту автоматично зараховується поразка, а поточна гра завершується.

Маючи у своєму розпорядженні симулятор, студентам пропонується запрограмувати власних агентів. Середовище також містить прості агенти, які реалізують стратегії «*Завжди брати один камінь*» (One-taker) та «*Завжди брати випадкову дозволену кількість камінців*» (Random agent). Початкові числа, N, для турнірів вибираються випадковим чином у діапазоні 50-100, і перший агент, що робить хід, також вибирається випадковим чином для кожної гри.

ШІ як набір умов

Студенти змогли екстраполювати свої початкові уявлення про алгоритм виграшу для невеликих чисел та реалізувати універсальний алгоритм виграшу: «*Зробіть хід, який залишає наступному гравцеві таке N, що ділиться на 4. Якщо такий хід неможливий, зробіть випадковий хід*» (рис. 3).

Рис. 3. Оптимальна стратегія для початкових правил гри Баше

Коректно запрограмовані агенти студентів здатні перемагати простих тестових агентів, а між собою грають у нічию. Деякі студенти на цьому етапі згадують популярний інтернет-мем (з посиланням на героїв мультфільму «Скубі-Ду») про те, що ШІ - це просто набір операторів if-then-else (рис. 4).

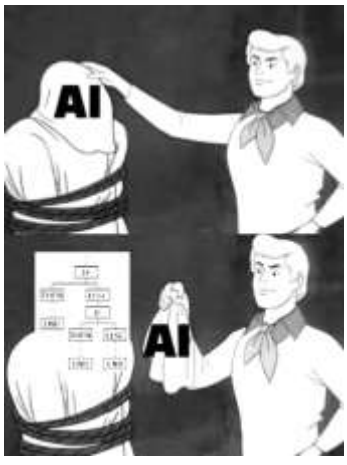

Рис. 4. Гумористичне зображення штучного інтелекту як набору умовних операторів

Після успішного завершення програмування свого першого ШІ-агента студенти цікавляться найбільш концептуальним питанням ШІ: «*Чи можемо ми розробити агента, в якого з самого початку не буде закладена виграшна стратегія, а який буде здатний самостійно її розробляти?*». Це питання зумовлює подальшу роботу над грою Баше в курсі.

Пошук по графу для розв'язання гри

Після розв'язання гри Баше за класичними правилами та реалізації ШІ-агентів, студенти можуть узагальнити виграшну стратегію на варіант правил, в якому гравці можуть брати будь-яку кількість камінців від 1 до k (включно). У цьому випадку виграшна стратегія полягає у тому, щоб залишити супернику таке число N, що N%(k+1)=0. Також вони можуть визначити виграшну стратегію для варіанту гри «Мізер». У цьому випадку супернику слід залишати такі числа N, що N%(k+1)=1.

Введення довільної множини дозволених ходів (наприклад {1, 3, 4}) призведе до побудови узагальненого алгоритму. На орієнтованому графі ігрових ситуацій (рис. 5) кожна позиція описується числом N та гравцем, чия черга ходити.

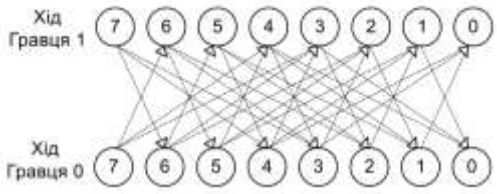

Рис. 5. Граф позицій гри Баше, починаючи з N=7, де множина дозволених ходів {1, 3, 4}

Позиція {N=0, ID гравця =1} є виграшною для гравця 0. Отже, всі позиції, які мають хоча б один із зв'язків, що веде до неї, є виграшними для гравця 0. Це позиції {N=1, ID гравця=0}, {N=3, ID гравця=0}, {N=4, ID гравця=0}. Симетрично, позиції {N=0, ID гравця =0}, {N=1, ID гравця =1}, {N=3, ID гравця =1}, {N=4, ID гравця =1} є виграшними для гравця 1, тому вони є програшними для гравця 0.

На наступному кроці ми визначимо програшні позиції. Це такі позиції, з яких всі вихідні ребра ведуть до виграшних позицій для суперника. Також виграшні позиції для поточного гравця можна визначити як позиції, з яких є хоча б одне вихідне ребро, що веде до програшної позиції для суперника. Алгоритм розмітки графа представлено на рис. 6. Розмічений граф з вершинами для N<=7 показано на рис. 7.

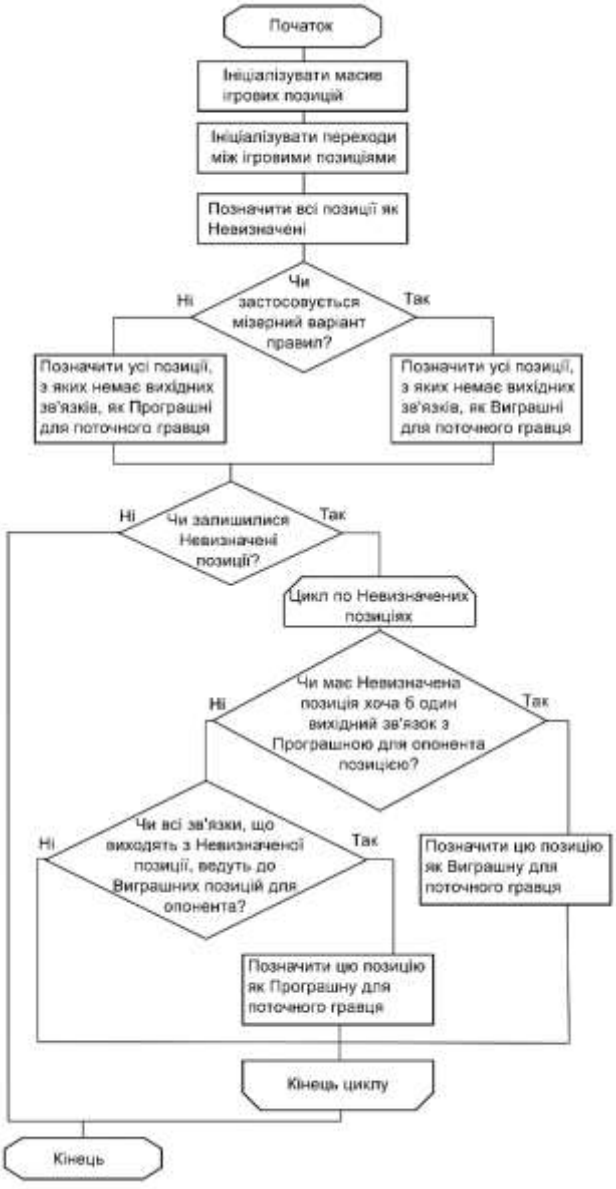

Рис. 6. Алгоритм визначення виграшної та програшної ігрових позицій для гри з 2-ма гравцями

Алгоритми розмітки графу та пошуку на ньому дозволяють створити універсального агента, який зможе розв'язувати та грати у гру Баше для 2-х гравців, як в режимі нормальної гри, так і в режимі «Мізер».

Агенти на базі Q-навчання

Наступне питання, яке цікавить студентів: «*Чи може ШІ-агент розробити виграшну стратегію методом спроб і помилок?*», тобто, спочатку взагалі не розуміти правил гри, але після кількох ігор почати грати все краще і краще?

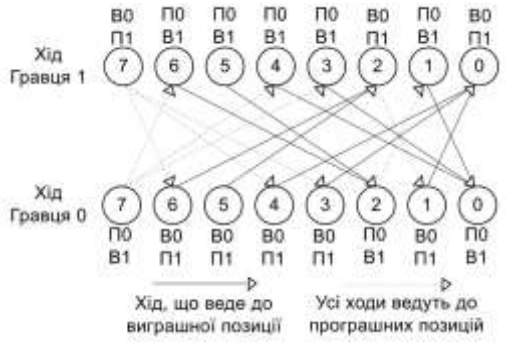

Рис. 7. Результати аналізу графа гри для допустимого набору ходів {1, 3, 4}, нормальний варіант гри

Q-навчання - найяскравіша ілюстрація агента, що самонавчається. Для кращого ефекту на лекції можна продемонструвати студентам суто механічну модель, попередньо оголосивши твердження: «*Масив сірникових коробок з кольоровими намистинами може навчитися вигравати в гру Баше*».

Суто механічний пристрій, який реалізує метод Q-навчання ШІ, був описаний в [22] для гри в покрокову гру «Нехарawn». Ідея полягає в тому, щоб зробити коробку для кожної можливої ігрової позиції і покласти туди кольорові намистини, які відповідають усім можливим ходам, які можна зробити з цієї позиції. Коли настає черга машини робити хід, з коробки береться випадкова намистина і виконується відповідний хід.

Всі намистини, вийняті з коробок, залишаються ззовні до кінця гри. Якщо гра закінчується перемогою машини, намистини повертаються до коробок, а до них додаються додаткові намистини того ж кольору. Якщо гра закінчується поразкою машини, намистини не повертаються до коробок. З другого правила є вийняток: якщо взята намистинка була останньою свого кольору, а загальна кількість намистинок менше 10, то вона повертається в коробку, але разом з нею в коробку додаються намистинки всіх інших кольорів. Це правило гарантує, що на початкових етапах навчання машина не «забуде» повністю потенційно виграшну стратегію.

Гру з Q-навчальною машиною можна проводити без механічного пристрою, просто виписуючи числа на дошці. Початковий стан машини, яка «вчиться» грати з позиції N=10 і дозволеними ходами 1, 2 або 3, показано на рис. 8.

Для отримання найшвидших результатів можна змоделювати гру між двома агентами Q-навчання, які мають спільну пам'ять. Розглянемо приклад гри.

Початково ймовірності зробити будь-який хід з будь-якої позиції дорівнюють 1/3. З позиції N=10 гравець 0 робить хід n=1. З позиції N=9 гравець 1 робить хід n=3. З позиції N=6 Гравець 0 робить хід n=2. З позиції N=4 Гравець 1 робить хід n=2. З позиції N=2 Гравець 0 робить хід n=3. Гравець 0 програє, оскільки цей хід є невалідним (неможливо взяти 3 камінці з купки, у якій їх 2). Стан пам'яті відразу після гри показано на рис. 9.

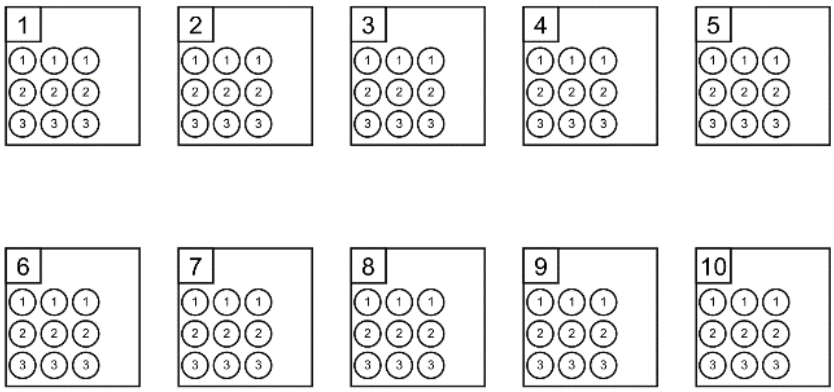

Рис. 8. Початковий стан Q-навчальної машини

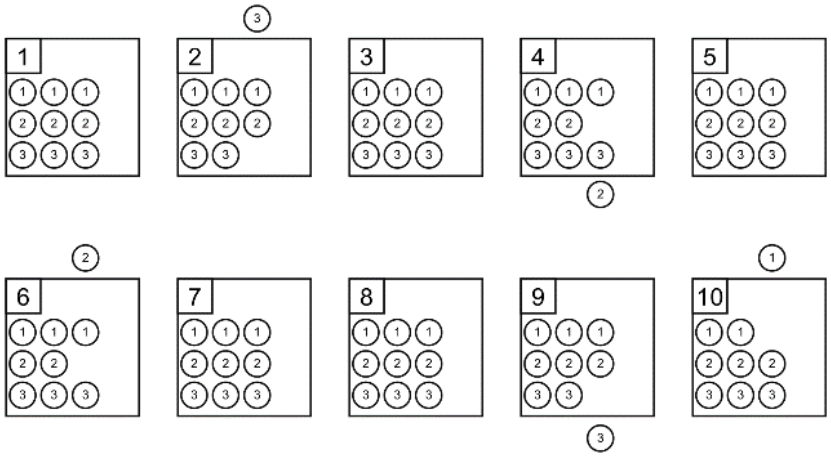

Рис. 9. Стан пам'яті одразу після завершення першої гри
(ходи гравця 0 показані над комірками, ходи гравця 1 - під комірками)

Оскільки гравець 0 програв, а гравець 1 став переможцем, вміст комірок відповідно змінюється (рис. 10).

Як бачимо, іноді ймовірність зробити виграшний хід може зменшуватись (хід n=2 з N=6 є виграшним, але тепер його ймовірність складає лише 0.25, оскільки гравець 0, який його зробив, зрештою програв гру). А іноді ймовірність виграшного ходу може збільшитися (з N=10 виграшним є хід n=2, ймовірність якого до гри була 1/3, а після гри стала 3/8).

Однак найважливішим фактом є те, що ймовірність останнього програшного ходу завжди зменшується. Гравець 0 програв партію через спробу зробити хід n=3 з позиції N=2, що заборонено правилами. Тому після гри ймовірність зробити цей хід становить не 1/3, а лише 1/4.

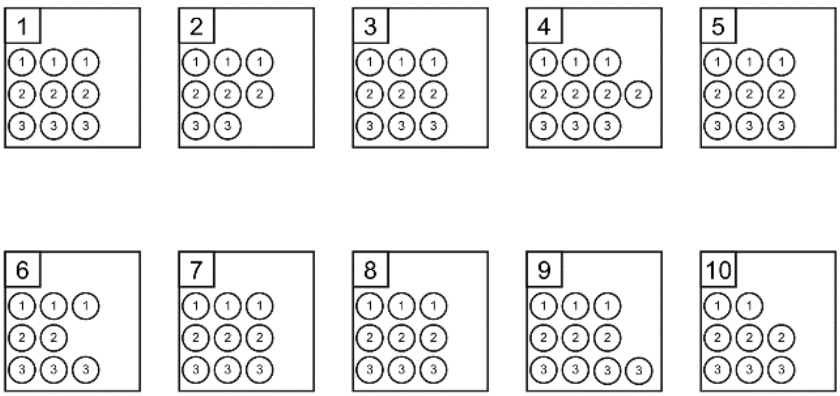

Рис. 10. Стан пам'яті після внесення змін після гри

Після наступних ігор тенденція до того, що машина робить правильні ходи, стає все більш помітною. Спочатку ймовірності зробити неможливі ходи (де n>N) наближаються до нуля. Потім ймовірність зробити хід n=3 з позиції N=3 наближається до 1. Після цього ймовірність зробити хід n=N%4 зростає для N в межах 5...7. Приклади змін у пам'яті показані на рис.11. Таблиця 1 показує, як змінюються ймовірності зробити виграшний хід з позиції в процесі навчання. Абсолютно програшні позиції (N=4 і N=8) не включені в таблицю.

Зрозумівши принципи Q-навчання на механічній або паперовій моделі, студенти можуть написати агент, який їх реалізує.

Проводячи серію турнірів, можна побачити, як зростає частота перемог агента і як змінюється вміст виграшних ходів у його пам'яті. Обговорюються ідеї подальшого покращення швидкості навчання: мати декілька Q-агентів зі спільною пам'яттю, скоригувати формулу, яка регулює зміну ймовірностей здійснення різних ходів тощо.

Агенти Q-навчання здатні адаптуватися до зміни правил гри. Щоб продемонструвати це, ми можемо провести серію турнірів, де Q-агенти, що навчаються, змагатимуться з агентами, що роблять ходи випадковим чином, та з жорстко закодованими «IF-агентами», які реалізують виграшну стратегію для звичайної гри Баше для набору дозволених ходів {1, 2, 3} (Рис. 12). Жорстко закодовані “IF-агенти” будуть лідирувати, повністю перемагаючи випадкових агентів. А Q-навчальні агенти з часом, турнір за турніром, покращуватимуть свої результати і наближатимуться до ідеально запрограмованих IF-агентів.

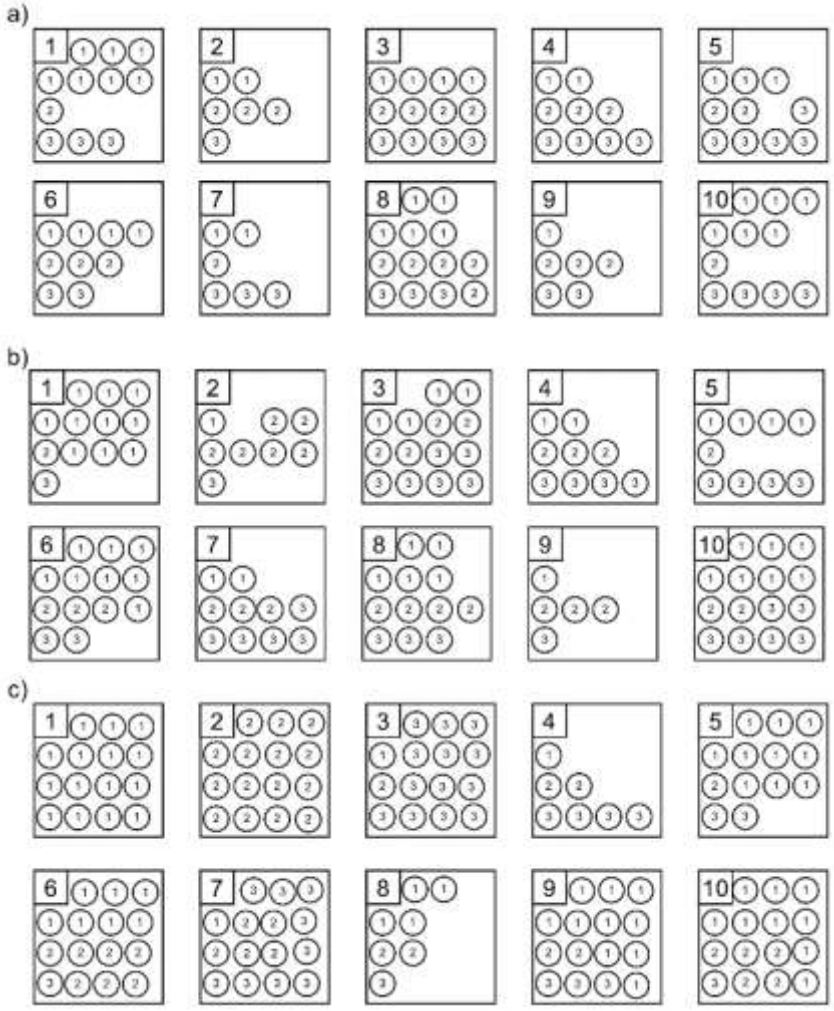

Рис. 11. Зміни в пам'яті Q-навчальної машини після а) 10 ігор б) 20 ігор в) 50 ігор

Таблиця 1. Ймовірності зробити виграшний хід з позицій після декількох ігор

N	10 ігор	20 ігор	50 ігор
1	64%	83%	100%
2	50%	75%	100%
3	33%	43%	87%
5	30%	44%	77%
6	33%	23%	47%
7	50%	40%	60%
9	17%	20%	67%
10	9%	13%	33%

Але як тільки ми введемо зміну в правила, наприклад, введемо варіант, де останній гравець програє, а не виграє, ситуація кардинально зміниться. IF-агенти, які запрограмовані на те, щоб не залишати жодного каменя для суперника, почнуть свідомо робити програшні ходи. Q-агенти, які побудували виграшну стратегію для нормального варіанта гри, теж погіршать свої результати. А агенти, які роблять випадкові ходи, вийдуть у лідери. Але це не триватиме довго. Q-агенти, що навчаються, за результатами програшів підлаштують свою пам'ять і навчаться робити нові ходи, які забезпечать їм перемогу в новому наборі правил. Деталізована зміна результатів агентів до і після зміни правил показана на рис.13.

Рис. 12. Результати Q-навченого, випадкового агента та IF-агента в серії турнірів, де було введено зміну правил гри

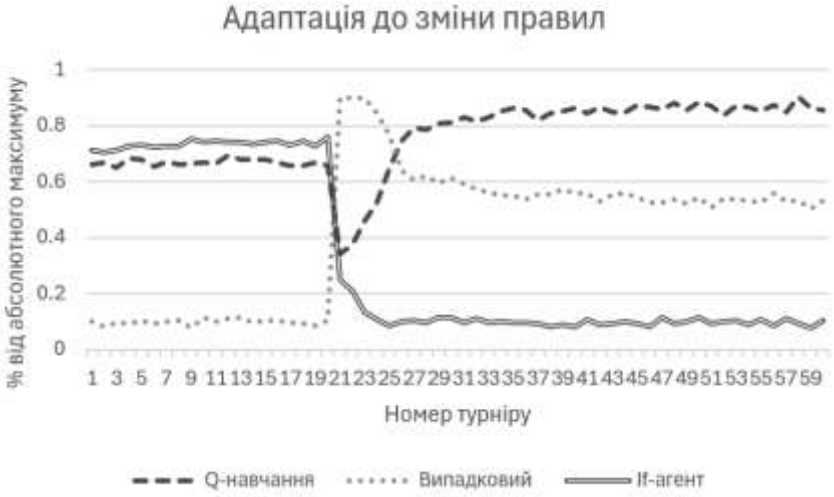

Рис. 13. Деталізоване зображення моменту зміни правил у серії турнірів

Після проведення цього експерименту студенти отримують досвід адаптації ШІ до мінливих умов, що є однією з властивостей інтелекту.

Аналіз графіка результатів турнірів підводить до ще одного цікавого питання, яке можна обговорити зі студентами. Можна помітити, що Q-навчальні агенти покращують свої результати на рис. 12, але ніколи не досягають результатів жорстко закодованих IF-агентів. Отже, «*Чи може Q-навчальний агент показати точно таку ж продуктивність, як і жорстко закодований агент на початкових правилах гри?*».

Після вивчення вмісту пам'яті Q-агента, що навчається, студенти можуть запропонувати зміни в алгоритмах і формулах налаштування пам'яті. Згодом обговорення призведе до концепції, яку неможливо було реалізувати в механічній машині з бісером, але легко в її комп'ютерній моделі. Зміни пам'яті слід помножити на коефіцієнт q^n, де $0<q<1$, а n - кількість ігрових ходів до останнього ігрового ходу. Таким чином, ми отримаємо більш плавне навчання, і можливі правильні ходи на початку гри (для більших N) не будуть стерті з пам'яті однією помилкою під час ендшпілю.

Студентам пропонується реалізувати цю концепцію у своїх запрограмованих агентах, порівняти поведінку Q-агентів для різних значень q і визначити оптимальне значення коефіцієнта.

Генетичні алгоритми

Наступний підхід, який вивчається на прикладі гри Баше, - генетичне програмування [23]. Кожен агент, розроблений у генетичній парадигмі, може розглядатися як «істота», чия «ДНК» контролює його стратегію.

Найпростіший спосіб закодувати ДНК агента - це використати масив, в якому його N-й елемент - це хід, який слід зробити з позиції N камінців. Розмір цього масиву буде динамічно збільшуватися, якщо агент стикається з ігровими позиціями, які ще не були закодовані в його ДНК (рис.14).

Рис. 14. Алгоритм поведінки простого генетичного агента

Згідно з парадигмою генетичного програмування, повинна існувати можливість створення ДНК нових агентів шляхом кросинговеру або мутації ДНК попереднього покоління.

Кросинговери реалізуються шляхом з'єднання підмасивів від 2 батьків таким чином, що другий підмасив починається з індексу, який знаходиться відразу після індексу, на якому закінчується перший підмасив.

Мутації ДНК реалізуються як випадковий вибір елементу масиву і заміна його на нове випадкове значення (рис.15).

Для демонстрації здатності генетичних агентів до еволюції та навчання необхідно

створити початкову популяцію розміром 5-10 агентів. Потім запускається турнір, і рахунок агента у турнірні використовується як його функція пристосованості. Після цього частина агентів, які показали найгірші результати, видаляється повністю. І створюються додаткові генетичні агенти, використовуючи найрезультативніших агентів кращих у якості батьківських.

Рис. 15. Приклади ДНК, кросинговеру та мутацій

Перші генетичні агенти будуть робити випадкові ходи, і часто ці ходи можуть суперечити правилам гри. Наприклад, ймовірність отримати число 1 у комірці масиву ДНК [1] у щойно створеного агента (для гри з доступними ходами 1, 2 або 3) становить лише 1/3. Але після кількох ітерацій студенти помітять, що всі агенти, які вижили, зроблять хід n=1 з ігрової позиції N=1.

Згодом для все більшої кількості елементів ДНК, таких, що N%4!=0, твердження DNA[N]=N%4 стане істинним. Для порівняння ефективності навчання генетичних агентів у турнірі можна додати 2 додаткових агента: випадковий агент та жорстко закодований IF-агент. Таким чином, співвідношення найкращого та середнього результатів турніру генетичних агентів та результатів цих 2 агентів покаже, як відбувається цифрова еволюція (рис. 16).

Під час типової симуляції еволюції можна помітити, що як тільки з'являється корисна мутація, вона поширюється по всій популяції, і відбувається прорив, коли генетичні агенти різко покращують свій результат у порівнянні з агентом, який виконує випадкові ходи. Також можуть відбуватися відкати, коли результати погіршуються, але загальний напрям спрямований у бік покращення. Однак, генетичним агентам потрібно набагато більше поколінь, щоб перевершити агента, який має оптимальну стратегію, жорстко закодовану в ньому.

Причиною цього є те, що чим більше елементів ДНК зрівнюються з виграшними ходами, тим менша ймовірність того, що наступна мутація буде корисною. Навпаки, мутації з більшою ймовірністю перетворять правильний хід на неправильний, погіршуючи загальну пристосованість. І через 250 поколінь ми, швидше за все, побачимо популяцію агентів, які, будучи здатними перевершити випадкову стратегію, все ще мають «помилки в своїй ДНК» (таблиця 2).

Лише через кілька тисяч поколінь генетичний агент здатен виграти одну-дві партії в турнірі у жорстко закодованого IF-агента, а для стабільних результатів у масштабах популяції потрібні десятки тисяч поколінь.

Студенти для досягнення бажаного результату еволюції швидше, можуть спробувати змінити такі параметри моделювання, як:

– кількість генетичних агентів, які залишаються в популяції після турніру;

– кількість генетичних агентів, які додаються до популяції перед кожним турніром;

– кількість негенетичних агентів, які використовуються як конкуренти;

– ймовірність мутацій.

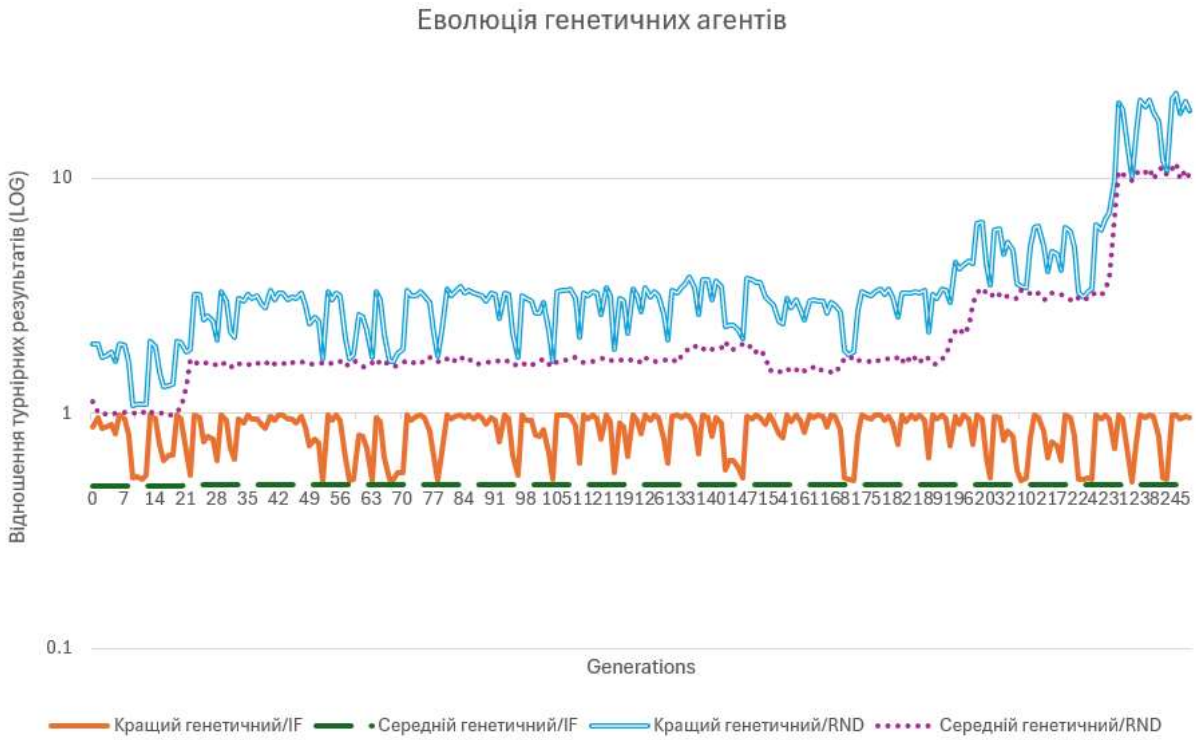

Рис. 16. Відношення (LOG-шкала) найкращого та середнього турнірних результатів генетичних агентів до результатів жорстко закодованого оптимального агента та агента з випадковими ходами

Таблиця 2. Відсоток генетичних агентів, для яких ДНК[N]==N%4

N	%	N	%	N	%	N	%
1	100	9	100	17	100	25	100
2	100	10	100	18	0	26	0
3	100	11	100	19	2	27	0
5	98	13	100	21	88	29	100
6	0	14	0	22	0	30	100
7	100	15	91	23	0	31	0

Також студенти можуть запропонувати ідеї альтернативного кросинговеру та мутації. Наприклад, використовувати в кросинговері ДНК більш ніж 2 агентів або змінювати більше одного елемента ДНК при мутації.

Експерименти з різними підходами до генетичного навчання привели до більш глибокого розуміння цієї концепції. Наступне питання, яке виникає: «*Чи можуть генетичні агенти бути більш адаптивними?*».

На наступному занятті можна познайомити учнів з новою моделлю генетичного агента. Цей тип агента має іншу структуру ДНК. Оскільки попередній аналіз різних варіантів гри Баше показав, що виграшний хід пов'язаний з остачею від ділення кількості камінців, N, то більш гнучкий генетичний агент матиме свою ДНК у вигляді масиву трійок чисел {a, b, c}. Кожна трійка інтерпретується як правило «*Якщо N%a==b, то взяти c камінців*».

Агент перебирає всі правила у своїй ДНК і робить хід згідно з тим, який може бути застосований до поточної ігрової ситуації (або робить випадковий хід, якщо відповідного правила не знайдено (рис.17)).

Довжина ДНК для вдосконаленого генетичного агента може бути довільною. Кросинговер працює як об'єднання двох підмасивів від батьківських агентів. Довжина ДНК після кросинговеру може змінюватися в обох напрямках.

Завдяки структурі ДНК вдосконаленого генетичного агента, його мутації

можуть здійснюватися шістьма різними способами (рис.18). Застосований тип мутації обирається випадковим чином серед можливих варіантів:

а) змінити значення a у випадковій трійці {a, b, c};

б) змінити значення b у випадковій трійці {a, b, c};

в) змінити значення c у випадковій трійці {a, b, c};

г) видалити випадкову трійку {a, b, c} з ДНК;

е) додати до ДНК нову трійку {a, b, c} з випадковими значеннями;

f) поміняти місцями дві випадково обрані трійки {a1, b1, c1} та {a2, b2, c2} у ДНК.

Рис.17. Алгоритм поведінки вдосконаленого генетичного агента

Алгоритм еволюційного навчання залишається фактично тим самим:

1) посів початкової популяції генетичних агентів;

2) додавання декількох негенетичних агентів для бенчмаркінгу;

3) проведення турніру між агентами;

4) видалення генетичних агентів з найнижчим показником пристосованості;

5) створення нових генетичних агентів за допомогою мутації та кросинговеру генетичних агентів з найкращим показником пристосованості;

6) повторення кроків 3-5 до тих пір, поки не буде досягнута необхідна результатисність у турнірі або не пройде задана кількість поколінь.

Детальне вивчення ДНК вдосконалених генетичних агентів призводить до кількох цікавих спостережень.

«Геномне сміття», явище нефункціональних ділянок ДНК, описане в роботі [24], має свою схожість у ДНК генетичних агентів. Наприклад, якщо трійка {3, 2, 3} передує трійці {9, 2, 1}, то остання ніколи не вплине на поведінку агента. Дійсно, якщо для деякого N виконується умова N%9==2, то умова N%3==2 також буде істинною, і в цій ситуації буде взято n=3 камінців.

І, як свідчать останні відкриття в генетиці, ділянки, які раніше вважалися «геномним сміттям», мають важливі функції [25], так і невикористані трійки можуть почати працювати і впливати на

поведінку агента після певної мутації.

Явище повторюваних послідовностей ДНК, яке має місце в генетиці людини [26], має свою схожість у ДНК штучних агентів. Після низки поколінь у масивах ДНК мають тенденцію з'являтися повторювані трійки. Для ШІ це має пояснення. Корисна трійки (наприклад, {4, 2, 2}, яка інтерпретується як правило «*Якщо N%4==2, взяти 2 камені*») може з'явитися в будь-якій частині ДНК агента. І після схрещування 2 агентів нащадок може отримати 2 копії цього «гена». Згодом нащадок з 2 копіями корисної трійки є більш стійким до негативних мутацій, які можуть змінити одне з цих правил.

Агенти досягають своєї оптимальної поведінки через кілька етапів, і можна простежити, як нова корисна адаптація поширюється в популяції. ДНК агента, який реалізує оптимальну гру, часто далека від мінімально можливої ДНК (рис. 19).

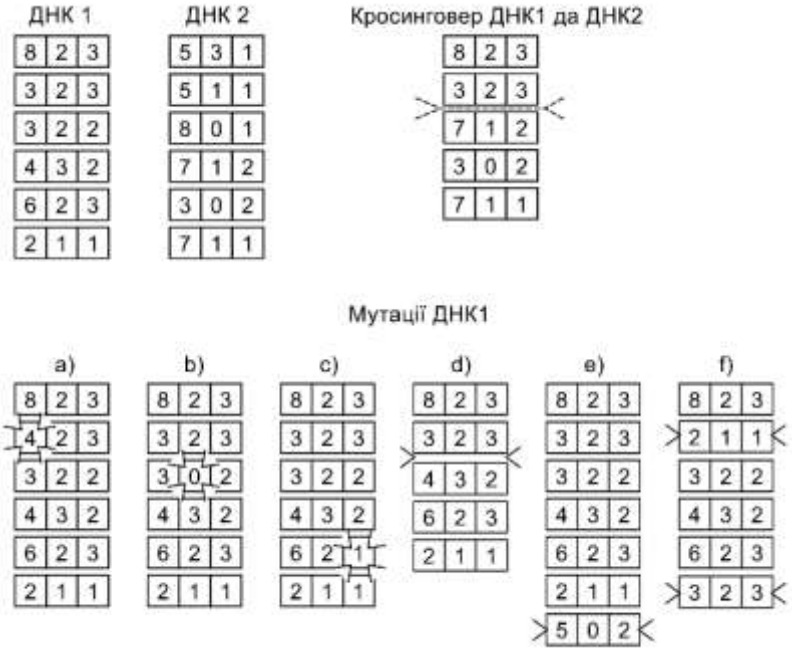

Рис. 18. Кросинговер та різні типи мутацій ДНК вдосконаленого генетичного агента

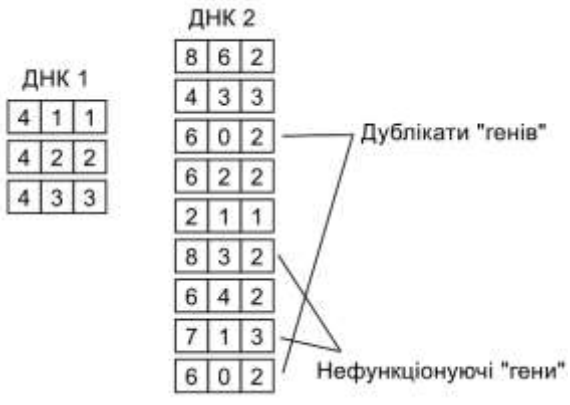

Рис. 19. Мінімально можливий (ДНК1) та фактично еволюціонувавший (ДНК2) масив трійок, які обидва реалізують оптимальну стратегію гри Баше

Серед правил з мінімальної ДНК лише правило {4, 3, 3} присутнє у ДНК, отриманій шляхом еволюції. Правило {4, 1, 1} реалізується більш загальним правилом {2, 1, 1}. Це правило диктує генетичному агенту зробити хід n=1 з усіх позицій, де N непарне. Однак, оскільки правило {2, 1, 1} знаходиться в масиві пізніше, ніж правило {4, 3, 3}, то на

позиції виду N=4m+3 воно не вплине, і агент забере по 3 камінці з позицій N=4m+3 і по 1 камінцю з позицій N=4m+3.

Правило {4, 2, 2} відсутнє в ДНК агента, що еволюціонував. Але закодована ним поведінка (взяти 2 камінці з позицій виду N=4m+2) реалізується набором правил: {8, 6, 2}, {6, 0, 2}, {6, 2, 2}, {6, 4, 2}. По суті, оскільки будь-яке парне число можна записати в одній з 3 форм: 6m, 6m+2 або 6m+4, агент буде брати 2 камінці з будь-якої позиції з N камінців, де N парне. Позиції парного N можуть бути N=4m+2 (у яких хід 2 камінців є виграшним) або N=4m (які є програшними позиціями, і будь-який хід не вплине на результат агента у турнірі проти іншого агента, що грає оптимально). Отже, для будь-якого парного N розроблений набір правил змусить агента забрати 2 камінці, що забезпечить йому перемогу.

У наведеному прикладі еволюційної ДНК також є приклади повторюваних «генів» (триплети {6, 0, 2}), а також трійки, які ніяк не впливають на поведінку агента. Триплет {8, 3, 2} «затіняється» триплетом {4, 3, 3}, який зустрічається в ДНК раніше. Отже, з позиції N=8m+3=4(2m)+3 агент зробить хід 3 камінцями, а не 2. А триплет {7,1,3} не вплине на шанси виграшу, оскільки для кожної виграшної позиції у вигляді N=7m+1 існує більш раннє правило, яке вже існує в ДНК (таблиця 3).

Таблиця 3. Доведення того, що трійка {7, 1, 3} є нефункціональним «геном»

N	Правило	Хід
1	{2, 1, 1}	1
15	{4, 3, 3}	3
22	{6, 4, 2}	2
29	{2, 1, 1}	1
36	{6, 0, 2}	2
43	{4, 3, 3}	3
50	{8, 6, 2}	2
57	{2, 1, 1}	1
64	{6, 4, 2}	2
71	{4, 3, 3}	3
78	{8, 6, 2}	2
85	{2, 1, 1}	1
92	{6, 2, 2}	2
99	{4, 3, 3}	3

Існування нефункціональних «генів» може бути корисною особливістю для адаптації генетичних агентів до змін правил гри. Як приклад, на рис. 20 показано симуляцію на 300 поколінь.

Популяція покращених генетичних агентів грає в турнірах гри Баше за стандартними правилами разом з трьома раніше описаними типами агентів: агент з жорстко закодованою оптимальною стратегією, агент, який робить випадкові ходи та агент, який реалізує підхід Q-навчання.

Спочатку лідирують агенти з жорстко закодованою стратегією, а агенти з Q-навчанням посідають друге місце. Агенти Q-навчання покращують свої результати протягом перших 8 турнірів, а потім виходять на плато. Жорстко закодованим агентам вдається не програти жодної гри жодному іншому типу агентів до 64-го покоління.

Найкращий результат генетичних агентів осцилює, основним фактором набору очок для них є перемоги над випадковими агентами, результати яких коливаються у протифазі.

Вм'ятина на графіку результату жорстко закодованих агентів, яка починається з 64 покоління, відображає момент, коли корисна мутація дозволила деяким генетичним агентам перемогти жорстко закодованих, але після п’яти поколінь ця мутація зникла без подальшого розповсюдження.

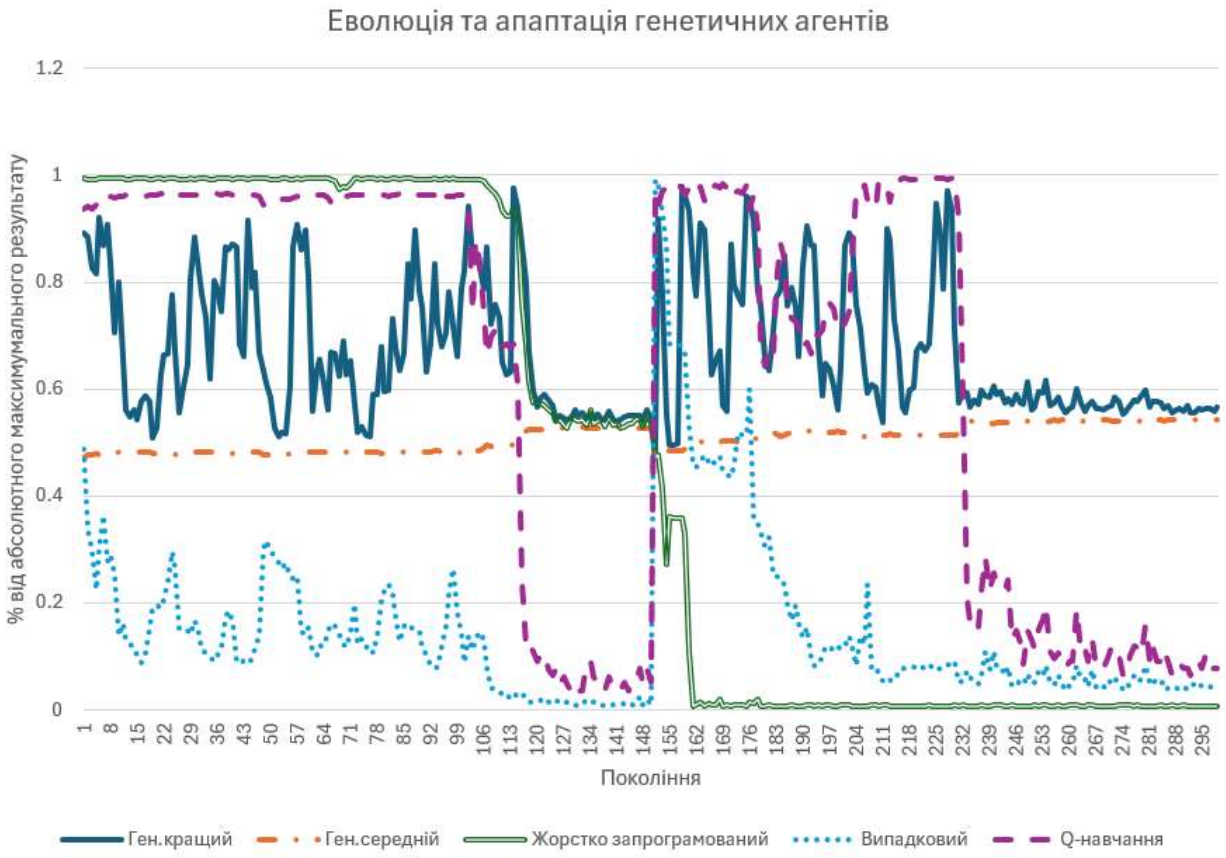

Рис. 20. Еволюція та адаптація генетичних агентів (зміна правил відбувається після 150 покоління)

У процесі моделювання на 99-му поколінні з'явилася наступна корисна мутація генетичних агентів, яка дозволила їм перемогти Q-навчених і посісти 2-е місце. І, нарешті, на 113-му поколінні з'явилася остання мутація, яка уможливила оптимальну стратегію гри, і поширилася по всій популяції генетичних агентів. Після цього результати жорстко закодованих агентів також впали до 50%, кількість перемог будь-якого типу агентів стала приблизно однаковою, і обидва типи агентів могли набрати лише 50% від абсолютно можливого максимуму очок у турнірі.

Різке падіння результату Q-агентів пояснюється особливостями їхнього алгоритму навчання. Досягнення перемоги над агентом, який робить ходи випадковим чином, може не вимагати оптимальних ходів. Але тоді ці ходи будуть зафіксовані в пам'яті Q-навченого агента як виграшні. І після цього Q-навчальний агент буде намагатися робити ці ходи в іграх з жорстко закодованими або еволюційними генетичними агентами, що призводитиме до програшу. Ця ж особливість (однакова винагорода за перемогу як над сильним, так і над слабким супротивником) не дозволила агентам, що Q-навчаються, розділити 1-е місце з жорстко закодованими агентами на першому етапі симуляції.

Після того, як генетичні агенти адаптувалися до гри, на 150-му поколінні було змінено правила. Звичайна гра Баше була замінена на мізерний варіант, де останній хід веде до програшу. Жорстко запрограмовані агенти почали програвати майже всі ігри, оскільки вони були спеціально запрограмовані брати останню фішку з купи, не даючи цього зробити іншому гравцеві.

Оскільки на момент зміни правил і агенти Q-навчання, і генетичні агенти також використовували стратегію забирати останній камінь, що залишився, лідерство перейшло до агентів, що роблять випадкові ходи. Так тривало не більше 3-х турнірів, поки агенти Q-навчання не адаптувалися до нових правил. Ще через 10 поколінь генетичні агенти виробили стратегію, яка

дозволила їм вигравати у випадкових агентів і, набагато рідше, у Q-агентів.

До покоління 176 (26 поколінь після зміни правил) генетичним агентам вдалося виграти значну частину ігор у Q-агентів, але мутація, яка уможливила це, схоже, зникла до покоління 211 (61 покоління після зміни правил). Нарешті, до покоління 232 популяція генетичних агентів виробила стійку адаптацію до нових правил гри, зайняла перше місце в турнірі і утримувала його до кінця симуляції. Амплітуда коливань зменшилася, оскільки генетичні агенти мали тенденцію отримувати приблизно однакову кількість перемог і поразок в іграх між собою, і вони вигравали у випадкових, жорстко закодованих і Q-навчальних агентів.

Найкращі генетичні агенти досягли близько 57% від абсолютно можливої кількості перемог у турнірних іграх, а в середньому результат генетичних агентів становив 54%.

Експерименти з генетичними агентами сприяли глибокому розумінню студентами принципів, методів та функціонування ШІ-агентів і, крім того, надали цінний досвід у проведенні наукових досліджень. Серед питань, які стали відправною точкою для подальших експериментів студентів, були такі:

В якому середовищі генетичне навчання буде найшвидшим: серед суто випадкових агентів чи серед агентів, які мають жорстко закодовану ідеальну стратегію?

Які типи мутацій генетичних агентів матимуть найбільший вплив на їхню еволюцію та швидкість адаптації?

Як довжина початкової ДНК впливає на швидкість еволюції та адаптації?

Як створювати генетичних агентів для більш складних правил гри (із забороненими ходами тощо)?

Також експериментально перевіряються ймовірність мутацій, відсоток виживання та інші гіперпараметри. Результати експериментів, виконаних студентами, представляються на заняттях з методів та систем штучного інтелекту та на засіданнях студентського наукового гуртка.

Нейронні мережі

Маючи досвід у програмування наперед заданої стратегії, пошуку у графі ігрових позицій, Q-навчанні та генетичному програмуванні, студенти замислюються: «*А що, якби агенти могли розробляти власні правила, дивлячись на ігрову ситуацію та результати?*». Це природно призводить до наступної теми - нейронні мережі. Дисципліна «Методи та системи штучного інтелекту» охоплює широкий спектр типів нейронних мереж та їх застосування. І на одному з занять ми разом зі студентами повертаємося до задачі про гру Баше, щоб побудувати нейромережевого агента, який зможе грати в цю гру і вигравати.

Щоб побудувати нейромережу для гри в Баше, необхідно задати форму вхідних даних для ігрової позиції. Зручно задавати кількість камінців в ігровій позиції, N, у вигляді масиву бітів. Тоді, якщо гру планується грати для N<=100, знадобиться мережа з 7 входами.

Мережа повинна мати стільки ж вихідних нейронів, скільки є варіантів ходів, тому для того, щоб агент міг грати в гру в базовому варіанті правил гри, потрібно три вихідних нейрона. Структура прихованих шарів може бути обрана довільно і є предметом дослідження студентів. Для початку використаємо один прихований шар з 20 нейронів (рис. 21).

Здатність нейронної мережі до узагальнення можна добре проілюструвати на прикладі класичних правил гри Баше. Можна запустити кероване навчання нейронної мережі, навчивши її знаходити правильні ходи для позицій N<20. Після цього навчена мережа здатна визначати виграшні ходи для всіх позицій N<100. Однак, цей випадок є скоріше щасливим збігом обставин. Через спосіб кодування ігрових позицій номер найбільш активованого нейрона фактично залежить лише від 2 з 7 вхідних даних, а обмеженої кількості вибірок достатньо для того, щоб побудувати правильні ваги та зміщення за допомогою зворотного розповсюдження.

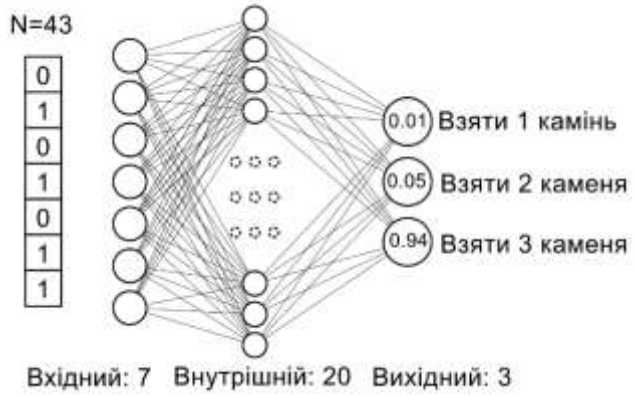

Рис. 21. Топологія та приклад роботи нейронної мережі, здатної грати в гру Баше

У загальному випадку поділ за модулем є дуже складною задачею для нейронної мережі. Тому для побудови найкращого нейронного агента для гри Баше потрібна комбінація декількох підходів до навчання без вчителя.

Нейронний агент, як і агент Q-навчання, буде відслідковувати свої ходи в межах однієї гри. Якщо гра закінчується перемогою, він пробіжить по списку позицій, з якими зіткнувся, і виконає алгоритм оберненого розповсюдження у мережі, щоб підсилити раніше зроблений вибір. У випадку поразки зворотне поширення також використовується для посилення ходів, які не були зроблені в грі.

Також реалізовано концепцію еволюційного навчання нейронних агентів. Після кожного турніру агенти, що показали найгірші результати, виключаються, а натомість створюються нові. Їх нейронні мережі створюються з нейронних мереж найкращих нейронних агентів за допомогою кросинговеру та мутацій.

Оскільки топологія нейронної мережі у всіх нейронних агентів однакова, кросинговер нейронної мережі працює як випадковий вибір ваг і зміщення кожного нейрона від одного з батьківських агентів. Мутація нейронної мережі - це зміна 10% її (випадково вибраних) нейронів. Зсув і вхідні ваги модифікованого нейрона множаться на випадковий коефіцієнт від -2 до 2.

Ця комбінація методів ШІ дозволила нам побудувати нейромережевий агент, який здатний навчатися та адаптуватися (Рис. 22).

У цьому експерименті випадкові агенти, агенти з жорстко закодованою оптимальною стратегією, агенти Q-навчання, генетичні агенти та нейронні агенти грали в серії турнірів за класичними правилами гри Баше. До 4-го покоління найкращим нейромережевим агентам вдалося зрівнятися за результатами з жорстко закодованими агентами, а найкращим генетичним агентам вдалося це зробити до 10-го покоління. Середній результат генетичних агентів виявився більш ніж у 2 рази вищим за середній результат нейромережевих агентів.

Після зміни правил першим адаптувався нейронний агент, за ним слідував агент Q-навчання. Найкращі генетичні агенти змогли адаптуватися до нових правил гри до 267 покоління (рис. 23), тоді як в середньому результати генетичних агентів перевищили результати нейронних агентів до 161 покоління.

Студенти охоче брали участь у порівнянні різних ШІ-агентів, розробці та вдосконаленні підходів до їхнього навчання, а також експериментували та представляли результати своїх експериментів. Ідеї для експериментів включали

– спробувати різні способи кодування ігрової ситуації;

– використовувати нейронні мережі з більшою кількістю прихованих шарів або без них;

– динамічно змінювати топологію нейронної мережі;

– випробовувати різні функції активації (в тому числі некласичні, такі як sin або cos);

– розширювати вхідні дані нейронної мережі, щоб врахувати додаткові правила гри, наприклад, дозволити/заборонити повтори.

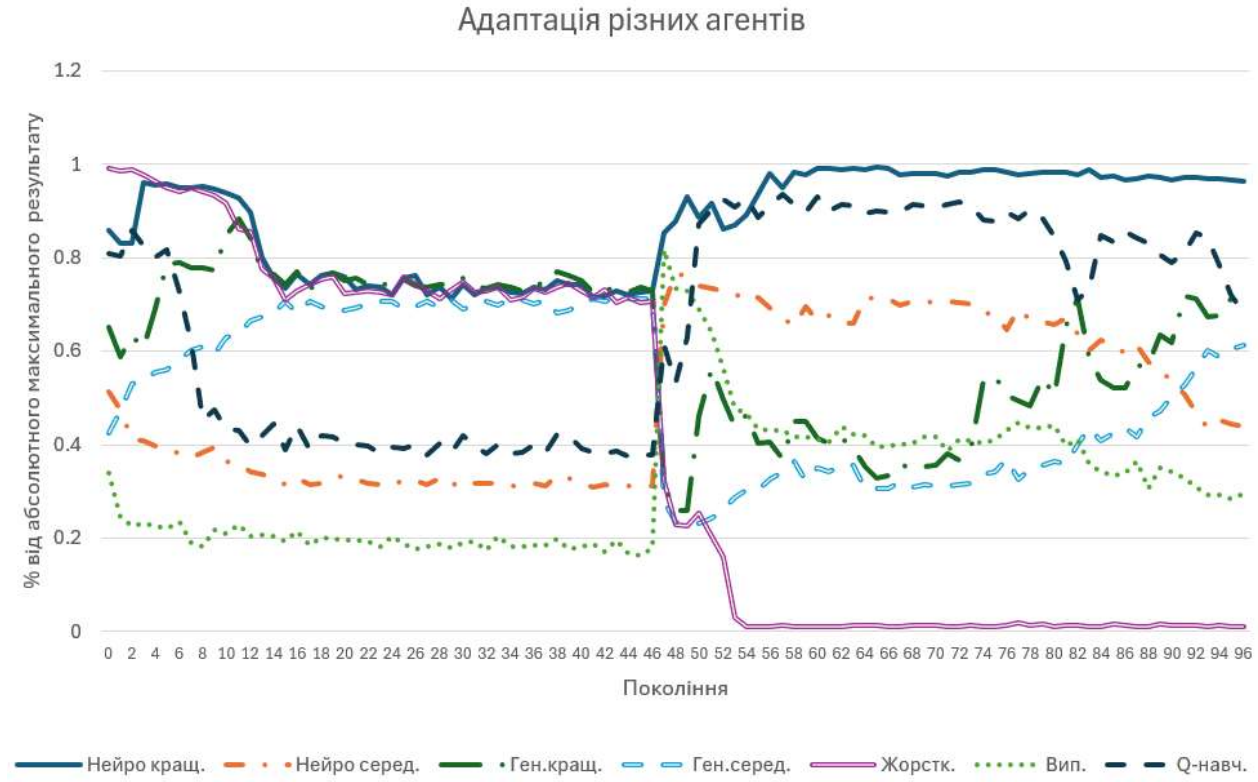

Рис. 22. Результати турніру агентів на основі нейронних мереж в умовах зміни правил гри

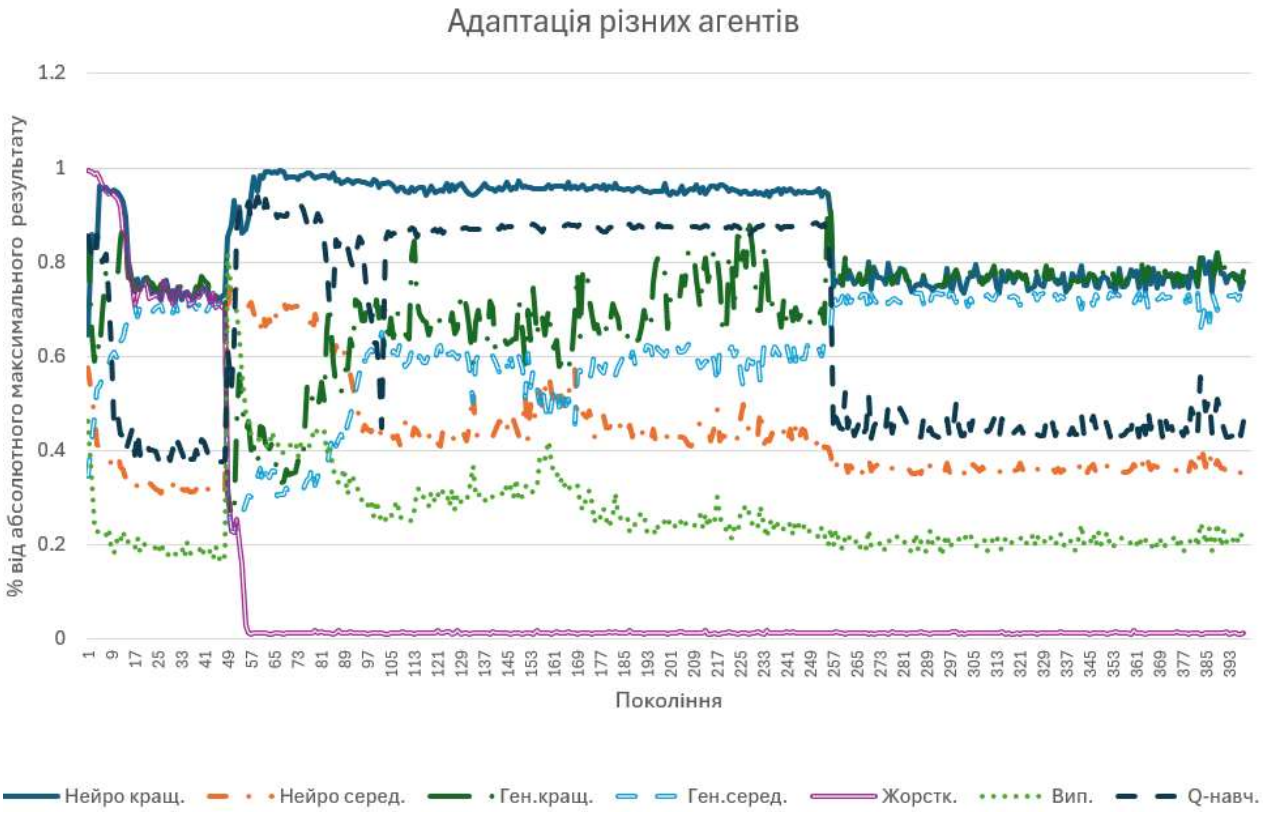

Рис. 23. Продовжений до покоління 400 графік поведінки агентів після зміни правил

Також деякі студенти створили власну експериментальну установку з власним користувацьким інтерфейсом для швидкого тестування своїх ідей. Деякі з результатів були представлені на наукових конференціях [27, 28].

Висновки

Наше дослідження показує, що гра Баше надає можливості для вивчення широкого спектру технологій, методів і систем ШІ. Серед них – пошук у графах, Q-навчання, генетичні алгоритми та нейронні мережі. Ця гра виявилася дуже ефективним експериментальним майданчиком для порівняння різних підходів і сприяння їх глибшому розумінню. Серед студентів виник сильний інтерес як до самого предмету, так і до наукових досліджень. Ми можемо назвати кілька факторів, які працювали на користь цього результату.

1. Гра проста для розуміння, і студенти можуть порівняти свої уявлення про виграшні стратегії зі стратегіями, які з'являються після навчання агентів.

2. Кількість станів гри набагато менша, ніж у шахів чи навіть хрестиків-нуликів, тому можна повністю побудувати граф гри та проаналізувати гру за допомогою ручки та паперу або на дошці.

3. Змагальний характер гри надихає студентів на пошук найефективніших стратегій навчання, щоб їхні власні агенти могли перемогти агентів, створених іншими студентами, у регулярних турнірах.

4. Правила гри легко змінюються, і зміни в правилах часто спонукають до пошуку зовсім іншої стратегії перемоги. Це допомагає проілюструвати здатність ШІ-агентів адаптуватися до мінливого середовища.

Методи та технології штучного інтелекту, які ілюструються на прикладі гри Баше, займають близько чверті всього курсу і стають основою для глибшого розуміння наступних його складових, таких як розпізнавання зображень, генеративний ШІ, прогнозування часових рядів та інших концепцій.

Матеріали курсу доступні на GitHub за ліцензією MIT [29] з дозволом на використання іншими викладачами у сфері ІТ. Результати нашої роботи пройшли апробацію та обговорення на наукових конференціях, і одним з результатів стало коригування структури навчального плану та збільшення аудиторних годин, відведених на цю дисципліну на наступний навчальний рік. Це забезпечить можливість організації більшої кількості студентських заходів, подібних до описаних у цій статті, у майбутньому.

Література

1. Наказ Міністерства освіти і науки України (2019) Про затвердження стандарту вищої освіти за спеціальністю 122 «Комп'ютерні науки» для першого (бакалаврського) рівня вищої освіти, Отримано з https://mon.gov.ua/npa/pro-zatverdzhennya-standartu-vishoyi-osviti-za-specialnistyu-122-kompyuterni-nauki-dlya-pershogo-bakalavrskogo-rivnya-vishoyi-osviti (дата перегляду 5 червня 2024 р.)
2. Lee, J., & Cho , J. (2024). Artificial Intelligence Curriculum Development for Intelligent System Experts in University. International Journal on Advanced Science, Engineering and Information Technology, 14(2), 409–419. https://doi.org/10.18517/ijaseit.14.2.18860
3. Eaton, E., & Epstein, S. L. (2024). Artificial Intelligence in the CS2023 Undergraduate Computer Science Curriculum: Rationale and Challenges. Proceedings of the AAAI Conference on Artificial Intelligence, 38(21), 23078-23083. https://doi.org/10.1609/aaai.v38i21.30352
4. Cai, J., Kwan, M., Hou, C., Liu, D., & Yam, Y. (2023). Curriculum Design of Artificial Intelligence and Sustainability in Secondary School. DOI: 10.5703/1288284317666
5. Федорін, І. (2022) Методи та технології обчислювального інтелекту: практикум. КПІ ім. Ігоря Сікорського. 317 с.
6. Орсірій, О. (2022) Methods and systems of artificial intelligence. Методи та системи штучного інтелекту. Національний університет «Одеська політехніка». Отримано з https://op.edu.ua/education/programs/components/14149 (дата перегляду 5 червня 2024 р.)
7. Лубко, Д., Шаров, С. (2019) Методи та системи штучного інтелекту. Таврійський державний агротехнологічний університет. Мелітополь: ФОП Однорог Т.В.. 265 с. ISBN 978-617-7566-68-6
8. Звенигородський, О. (2024) Силабус курсу "Методи та засоби штучного інтелекту" Отримано з https://duikt.edu.ua/uploads/p_163_35672080.pdf (дата перегляду 5 червня 2024 р.)
9. Прокончук, Ю. (2022) Методи та системи штучного інтелекту. Силабус. Придніпровська державна академія будівництва та архітектури. Отримано з https://pgasa.dp.ua/wp-content/uploads/2023/01/Metody-ta-systemy-shtuchnogo-intelektu.pdf (дата перегляду 2 червня 2024 р.)
10. Шаповалова, С. (2023). Методи та системи штучного інтелекту. Силабус. КПІ імені Ігоря

Сікорського. Отримано з https://dte.kpi.ua/wp-content/uploads/2024/02/po-15_metod_ta_syst_shtuch_intelekt.pdf (дата перегляду 6 червня 2024 р.)

11. Антіпова, К. (2023) Силабус «Системи штучного інтелекту». Чорноморський національний університет імені Петра Могили. Отримано з https://chmnu.edu.ua/wp-content/uploads/Syllabus_Artificial-Intelligence-Systems.pdf (дата перегляду 6 червня 2024 р.)

12. Грабовський, В. (2020) Методи та системи штучного інтелекту. Львівський національний університет імені Івана Франка Отримано з https://electronics.lnu.edu.ua/wp-content/uploads/Navch.-Prohp_METODY-TA-SYSTEMY-SHTUCHNOHO-INTELEKTU-2020.pdf (дата перегляду 6 червня 2024 р.)

13. Klein, D. & Abreel, P. (2014) UC Berkeley CS188 Intro to AI -- Course Materials Отримано з http://ai.berkeley.edu/home.html (дата перегляду 6 червня 2024 р.)

14. Winston, P. (2015) Artificial Intelligence. OpenCourseWare. Отримано з https://ocw.mit.edu/courses/6-034-artificial-intelligence-fall-2010/pages/tutorials/ (дата перегляду 6 червня 2024 р.)

15. Finn, C. & Anari, N (2020) CS221: Artificial Intelligence: Principles and Techniques. Stanford. Отримано з https://stanford-cs221.github.io/spring2020/ (дата перегляду 6 червня 2024 р.)

16. Савченко, С., Синельніков, О. (2017) Методи та системи штучного інтелекту: Навчальний посібник для студентів напряму підготовки 6.050101 «Комп'ютерні науки». К. : НАУ, 2017. 190 с.

17. Russel, S. & Norwig, P. Artificial Intelligence: A Modern Approach, 4th US ed. Отримано з https://aima.cs.berkeley.edu/ (дата перегляду 6 червня 2024 р.)

18. Троцько, В. (2020) Методи штучного інтелекту: навчально-методичний і практичний посібник. ВНЗ «Університет економіки та права «КРОК» 86 с.

19. Іщенко,Г., Рогоза, В.,Сергеєв-Горчинський, О., Харченко, К. (2019) Методи та системи штучного інтелекту: Лабораторний практикум; Київ : КПІ ім. Ігоря Сікорського. 78 с.

20. C. Bachet & A. Labosne (1874) Probl`emes plaisants et d´electables qui se font par les nombres. Paris: Gauthier-Villars, 242 p.

21. Golomb, S. (1966) A mathematical investigation of games of "take-away". Journal of Combinatorial Theory Volume 1, Issue 4, December 1966, Pages 443-458

22. Gardner, M. (1972) The Unexpected Hanging and Other Mathematical Diversions. New York: NY Simon and Schuster, 262p.

23. Russell, S. & Norvig, P. (2010) Artificial Intelligence A Modern Approach Third Edition. Upper Saddle River: Pearson 1151p.

24. Comings D. (1972). "The structure and function of chromatin". Advances in human genetics. Springer. pp. 237–431.

25. Jagannathan, M. & Yamashita, Y. (2021) Defective Satellite DNA Clustering into Chromocenters Underlies Hybrid Incompatibility in Drosophila. Molecular Biology and Evolution, Volume 38, Issue 11, November 2021, Pages 4977–4986, https://doi.org/10.1093/molbev/msab221

26. de Koning, A., Gu, W., Castoe, T., Batzer, M. & Pollock D. (2011) Repetitive elements may comprise over two-thirds of the human genome. PLOS Genetics. 7 (12): e1002384. doi:10.1371/journal.pgen.1002384. PMC 3228813. PMID 22144907.

27. Міжнародна наукова конференція "Штучний інтелект: досягнення, виклики, ризики" (2024) Отримано з https://www.ipai.net.ua/uk/stattya/mizhnarodna-naukova-konferenciya-shtuchnyy-intelekt-dosyagnennya-vyklyky-ryzyky (дата перегляду 6 червня 2024 р.)

28. IV Всеукраїнська науково-практична конференція "Освітня робототехніка та штучний інтелект" (2024) Отримано з https://imzo.gov.ua/events/iv-vseukrains-ka-naukovo-praktychna-konferentsiia-osvitnia-robototekhnika-ta-shtuchnyy-intelekt/ (дата перегляду 6 червня 2024 р.)

29. Ізвалов, О. (2024) Репозиторій для вивчення дисципліни «Методи та системи штучного інтелекту» Отримано з https://github.com/GeneralVimes/AI (дата перегляду 6 червня 2024 р.)

References

1. Ministry of Education and Science of Ukraine. (2019). Order on approval of the standard of higher education for the specialty 122 "Computer Science" for the first (bachelor's) level of higher education. Retrieved from https://mon.gov.ua/npa/pro-zatverdzhennya-standartu-vishoyi-osviti-za-specialnistyu-122-kompyuterni-nauki-dlya-pershogo-bakalavrskogo-rivnya-vishoyi-osviti (accessed on June 5, 2024). [in Ukrainian]

2. Lee, J., & Cho, J. (2024). Artificial Intelligence Curriculum Development for Intelligent System Experts in University. International Journal on Advanced Science, Engineering and Information Technology, 14(2), 409–419. https://doi.org/10.18517/ijaseit.14.2.18860

3. Eaton, E., & Epstein, S. L. (2024). Artificial Intelligence in the CS2023 Undergraduate Computer Science Curriculum: Rationale and Challenges. Proceedings of the AAAI Conference on Artificial Intelligence, 38(21), 23078-23083. https://doi.org/10.1609/aaai.v38i21.30352

4. Cai, J., Kwan, M., Hou, C., Liu, D., & Yam, Y. (2023). Curriculum Design of Artificial Intelligence and Sustainability in Secondary School.
DOI: 10.5703/1288284317666

5. Fedorin, I. (2022). Methods and technologies of computational intelligence: A practicum. Igor Sikorsky Kyiv Polytechnic Institute. 317 p. [in Ukrainian]

6. Orsiriy, O. (2022). Methods and systems of artificial intelligence. Odesa Polytechnic National University. Retrieved from https://op.edu.ua/education/programs/components/14149 (accessed on June 5, 2024). [in Ukrainian]

7. Lubko, D., & Sharov, S. (2019). Methods and systems of artificial intelligence. Dmytro Motornyi Tavria State Agrotechnological University. Melitopol: FOP Odnoroh T.V. 265 p. ISBN 978-617-7566-68-6. [in Ukrainian]

8. Zvenyhorodskyi, O. (2024). Syllabus of the course "Methods and Tools of Artificial Intelligence". Retrieved from https://duikt.edu.ua/uploads/p_163_35672080.pdf (accessed on June 5, 2024). [in Ukrainian]

9. Prokonchuk, Yu. (2022). Methods and systems of artificial intelligence. Syllabus. Prydniprovska State

Academy of Civil Engineering and Architecture. Retrieved from https://pgasa.dp.ua/wp-content/uploads/2023/01/Metody-ta-systemy-shtuchnogo-intelektu.pdf (accessed on June 2, 2024). [in Ukrainian]

10. Shapovalova, S. (2023). Methods and systems of artificial intelligence. Syllabus. Igor Sikorsky Kyiv Polytechnic Institute. Retrieved from https://dte.kpi.ua/wp-content/uploads/2024/02/po15_metod_ta_syst_shtuch_intelekt.pdf (accessed on June 6, 2024). [in Ukrainian]

11. Antipova, K. (2023). Syllabus "Artificial Intelligence Systems". Petro Mohyla Black Sea National University. Retrieved from https://chmnu.edu.ua/wp-content/uploads/Syllabus_Artificial-Intelligence-Systems.pdf (accessed on June 6, 2024). [in Ukrainian]

12. Hrabovskyi, V. (2020). Methods and systems of artificial intelligence. Ivan Franko National University of Lviv. Retrieved from https://electronics.lnu.edu.ua/wp-content/uploads/Navch.-Prohp_METODY-TA-SYSTEMY-SHTUCHNOHO-INTELEKTU-2020.pdf (accessed on June 6, 2024). [in Ukrainian]

13. Klein, D., & Abbeel, P. (2014). UC Berkeley CS188 Intro to AI -- Course Materials. Retrieved from http://ai.berkeley.edu/home.html (accessed on June 6, 2024).

14. Winston, P. (2015). Artificial Intelligence. OpenCourseWare. Retrieved from https://ocw.mit.edu/courses/6-034-artificial-intelligence-fall-2010/pages/tutorials/ (accessed on June 6, 2024).

15. Finn, C., & Anari, N. (2020). CS221: Artificial Intelligence: Principles and Techniques. Stanford. Retrieved from https://stanford-cs221.github.io/spring2020/ (accessed on June 6, 2024).

16. Savchenko, S., & Synelnikov, O. (2017). Methods and systems of artificial intelligence: Study guide for students of specialty 6.050101 "Computer Science". Kyiv: NAU. 190 p. [in Ukrainian]

17. Russell, S., & Norvig, P. Artificial Intelligence: A Modern Approach, 4th US ed. Retrieved from https://aima.cs.berkeley.edu/ (accessed on June 6, 2024).

18. Trotsko, V. (2020). Methods of artificial intelligence: Educational, methodological and practical guide. "KROK" University. 86 p. [in Ukrainian]

19. Ishchenko, H., Rohoza, V., Serheiev-Horchynskyi, O., & Kharchenko, K. (2019). Methods and systems of artificial intelligence: Laboratory practicum. Kyiv: Igor Sikorsky Kyiv Polytechnic Institute. 78 p. [in Ukrainian]

20. Bachet, C., & Labosne, A. (1874). Problèmes plaisants et délectables qui se font par les nombres [Pleasant and Delectable Problems to be Done by Numbers]. Paris: Gauthier-Villars, 242 p.

21. Golomb, S. (1966). A mathematical investigation of games of “take-away”. Journal of Combinatorial Theory, 1(4), 443-458.

22. Gardner, M. (1972). The Unexpected Hanging and Other Mathematical Diversions. New York, NY: Simon and Schuster, 262 p.

23. Russell, S., & Norvig, P. (2010). Artificial Intelligence: A Modern Approach (3rd ed.). Upper Saddle River: Pearson. 1151 p.

24. Comings, D. (1972). The structure and function of chromatin. Advances in human genetics. Springer. pp. 237–431.

25. Jagannathan, M., & Yamashita, Y. (2021). Defective Satellite DNA Clustering into Chromocenters Underlies Hybrid Incompatibility in Drosophila. Molecular Biology and Evolution, 38(11), 4977–4986. https://doi.org/10.1093/molbev/msab221

26. de Koning, A., Gu, W., Castoe, T., Batzer, M., & Pollock, D. (2011). Repetitive elements may comprise over two-thirds of the human genome. PLOS Genetics, 7(12): e1002384. doi:10.1371/journal.pgen.1002384.

27. International Scientific Conference "Artificial Intelligence: Achievements, Challenges, Risks". (2024). Retrieved from https://www.ipai.net.ua/uk/stattya/mizhnarodna-naukova-konferenciya-shtuchnyy-intelekt-dosyagnennya-vyklyky-ryzyky (accessed on June 6, 2024). [in Ukrainian]

28. IV All-Ukrainian Scientific and Practical Conference "Educational Robotics and Artificial Intelligence". (2024). Retrieved from https://imzo.gov.ua/events/iv-vseukrains-ka-naukovo-praktychna-konferentsiia-osvitnia-robototekhnika-ta-shtuchnyy-intelekt/ (accessed on June 6, 2024). [in Ukrainian]

29. Izvalov, O. (2024). Repository for studying the discipline "Methods and Systems of Artificial Intelligence". Retrieved from https://github.com/GeneralVimes/AI (accessed on June 6, 2024). [in Ukrainian]

The article has been sent to the editors 09.12.25.
After processing 17.12.25.
Submitted for printing 30.12.25.