ВІННИЦЬКИЙ ДЕРЖАВНИЙ ПЕДАГОГІЧНИЙ УНІВЕРСИТЕТ ІМЕНІ МИХАЙЛА КОЦЮБИНСЬКОГО

Факультет математики, фізики, комп'ютерних наук і технологій
Кафедра математики та інформатики

ДИПЛОМНА РОБОТА

на тему: **«БАГАТОКРИТЕРІАЛЬНА ОПТИМІЗАЦІЯ МЕХАНІЗМУ ПРИЙНЯТТЯ РІШЕНЬ НА ПРИКЛАДІ РЕАЛІЗАЦІЇ ШАХОВОЇ ПРОГРАМИ»**

Студентки 2 курсу МСОІ групи
Освітньої програми Середня освіта.
Інформатика, математика
Спеціальності 014 Середня освіта
(Інформатика)
Галузі знань 01 Освіта/Педагогіка
Ступеня вищої освіти магістра
Руденка Сергія Миколайовича

Науковий керівник:
кандидат технічних наук, доцент
Бак Сергій Миколайович

Національна шкала _____
Кількість балів: _____ Оцінка: ECTS _____

Голова комісії	_____ (підпис)	_____ (ініціали, прізвище)
Члени комісії	_____ (підпис)	_____ (ініціали, прізвище)
	_____ (підпис)	_____ (ініціали, прізвище)
	_____ (підпис)	_____ (ініціали, прізвище)

м. Вінниця – 2019 рік

АНОТАЦІЯ

У роботі розглянуто особливості прийняття рішення в режимі реального часу шляхом додавання алгоритму оцінки шахової позиції. Проаналізовані різні підходи оцінки позиції, доступні зараз, історичний розвиток та результати протистояння штучного інтелекту та людини. Покращено алгоритм оцінки, наявний зараз, розроблено застосунок. Також проаналізовано вибір методів, форм і засобів розробки, тренування та тестування. Розглянуто та запропоновано подальші кроки розвитку алгоритму.

Експериментально перевірено ефективність розробленої методики.

ANNOTATION

In the master's thesis are being considered different types of real-time making decision algorithms in example of chess position evaluation. There were different modern techniques of position evaluation were investigated, overviewed historical development and Computer vs Human game results were shown. Position evaluation algorithm was improved. It also analyzes the choice of methods, forms and means of training and the organization of the development and testing outcomes.

The effectiveness of the developed methodology was experimentally verified.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

AI – Artificial Intelligence

Alpha* - сімейство програм від Google/DeepMind, до якого входять AlphaGo, AlphaZero та інші похідні.

ШІ – штучний інтелект

ІС – інтелектуальна система

ММ – MiniMax, мінімакс алгоритм

НМ – негамакс алгоритм

ВСТУП

Актуальність теми дослідження. У наш час надзвичайно велика увага приділяється дослідженню та розробці механізму штучного інтелекту (ШІ). Така увага до нього обумовлена тим, що задачі, які перед ним ставлять, різнопланові і можуть не мати якогось чіткого, наперед відомого, розв'язку, але є тими завданнями, які вирішує людина впродовж свого життя. Основна задача ШІ – розв'язати поставлену задачу, базуючись на попередньому досвіді (своєму чи чужому) і, найголовніше, навчатися. В будь-якому випадку, на сьогоднішній день ШІ є одним з основних елементів будь-якої інтелектуальної системи (ІС). З точки зору розробки ШІ, розрізняють 2 основних підходи: низхідний (для моделювання психічних процесів, таких як мова, емоції, мислення) та висхідний (вивчення штучних нейронних мереж) [1, 2].

Основна ідея застосування ШІ – спростити життя людині: пришвидшити виконання рутинної роботи, підвищити якість виконання поставлених завдань тощо. Особливий інтерес викликають сьогодні дослідження в області ігрових технологій при використанні ШІ: шахи і го, покер, комп'ютерні стратегії в режимі реального часу. В будь-якій з цих областей важливу роль відіграє те, наскільки учасник ігрового процесу зможе швидко і якісно пристосуватися до противника, його дій, емоцій. Можливість навчатися на помилках, запам'ятовувати виграшні стратегії, робити переоцінку ситуації з урахуванням можливих продовжень, швидкість реакції, де це потрібно – ось основні критерії оцінки дієздатності алгоритму [24].

Вирішення зазначених вище проблем вимагає нових нетрадиційних підходів, розробки та використання новітніх методів, технологій та інструментів – штучних нейронних мереж, машинного навчання, статистики і прогнозування

тощо. На сьогоднішній день саме такі методи і технології є найбільш популярними і перспективними в розробці означеної теми.

Об'єкт і предмет дослідження.

Об'єктом дослідження є процес аналізу шахової позиції.

Предметом дослідження є методи пошуку і прийняття рішення.

Метою роботи є розробка алгоритму прийняття рішення про наступний хід, шляхом оцінювання позиції з використанням «таблиць ходів» та відсіканням вперед.

Для досягнення визначеної мети необхідно виконати наступні **завдання:**

- дослідити етапи розвитку, застосування та оцінки якості штучного інтелекту;
- дослідити існуючі методи пошуку рішення на предмет їх ефективності з метою вибору оптимального методу;
- проаналізувати та порівняти способи оцінки позиції людиною та комп'ютером;
- розробити алгоритм вибору наступного ходу шляхом використання таблиць ходів замість таблиць позицій;
- розробити логічну структуру бази даних ходів;
- оцінити можливості та перспективи подальшого розвитку і застосування розробленого алгоритму та можливість його практичного застосування.

Методи дослідження: багатокритеріальні алгоритми пошуку та оцінки позиції, комп'ютерне моделювання, методи штучного інтелекту, моделювання штучних нейромереж, методи самонавчання штучних нейромереж.

Практичне значення результатів дослідження. Результати дипломної роботи можуть бути використаними:

- при розробці навчальних програм для гри в шахи;

- на практичних та лабораторних уроках з інформатики та програмування;
- при підготовці до олімпіад з програмування.

Публікації та апробація результатів дипломної роботи. Результати дипломної роботи опубліковані у науково-популярному альманасі «Математика та інформатика навколо нас» ([19]) і доповідались на Всеукраїнські Інтернет-конференції «Математика та інформатика у вищій школі: виклики сучасності» ([24]), а також на науковому семінарі під час проходження науково-дослідної практики.

Структура роботи. Дипломна робота складається зі вступу, двох розділів (з висновками до них), загальних висновків та списку використаних джерел з 26 найменувань (з них англійською – 14). Робота викладена на 42 сторінках друкованого тексту і містить 8 рисунків.

РОЗДІЛ 1. АНАЛІЗ СУЧАСНОГО СТАНУ РОЗВИТКУ ШІ ВЦІЛОМУ ТА КОМП'ЮТЕРНИХ ШАХІВ ЯК ОБЛАСТЬ ЗАСТОСУВАННЯ

1.1. Способи оцінки якості штучного інтелекту в протистояннях з людьми

Досі не існує чіткого способу визначити, чи машинний інтелект «розумний» і наскільки. Загальноприйнятим способом вважається тест Тьюрінга, коли суддя С спілкується з людиною А і комп'ютером В. Задача судді: однозначно визначити хто із співбесідників людина, а хто – робот. Задача програми – ввести суддю в оману. Режим тестування відбувається всліпу в режимі «лише текст», щоб була можливість перевірити саме інтелект, а не уміння розпізнавати емоції, жести і мову. Виходячи з того, що комп'ютери зараз працюють і реагують швидше людини (хоча перші комп'ютери – повільніше) одним із основних правил є те, що суддя повинен витримувати певну паузу, перед відправкою наступного повідомлення. Якщо суддя, а як результат і більша частина жюрі, не може визначитися, або помилився – вважається, що машина пройшла тест. Цій ідеї понад 50 років. Запропонована вона була в 1950р. Аланом Тьюрінгом у статті філософського журналу Mind, метою якої було визначити, чи може комп'ютер мислити як людина[3]. Алан вирішив розглянути гру в імітацію, коли чоловіка та жінку розводять у різні кімнати і задають їм написані запитання і отримують надруковані відповіді, при чому жінка намагається відповісти як чоловік і навпаки. А аудиторія, що задавала

запитання, повинна визначити наприкінці, хто ж насправді де сидів. Базуючись на цій грі, А. Тьюрінг у своїй статті підняв запитання: а що буде, якщо одного з прихованих учасників замінити на машину, чи зможе вона відповідати, як людина, чи здатна вона мислити. Ця стаття стала передумовою офіційної організації проведення тесту Тьюрінга, яку започаткували в 1990р. Основною офіційною платформою для проведення тесту вважається премія Хью Льобнера – щорічний конкурс «AI Loebner», який вперше був проведений 1991р. Найбільш «людяній» програмі гарантується приз в $2k за час проведення одного конкурсу. Передбачено також дві медалі – срібна (за проходження стандартного текстового тесту, приз $25k) та золота (проходження одночасно текстового, звукового та візуального тестів, приз $100k). У разі вручення золотої медалі конкурс офіційно вважатиметься закритим. Але до сьогодні ще жодна програма не отримала навіть срібної медалі.

Незважаючи на те, що з моменту написання статті пройшло все-таки доволі багато часу та що ще жодного разу премія за успішне проходження тесту Тьюрінга так і не вручалася, люди навчилися використовувати машини з їх «інтелектом» у різних галузях, наприклад у настільних іграх (рис. 1 [4]).

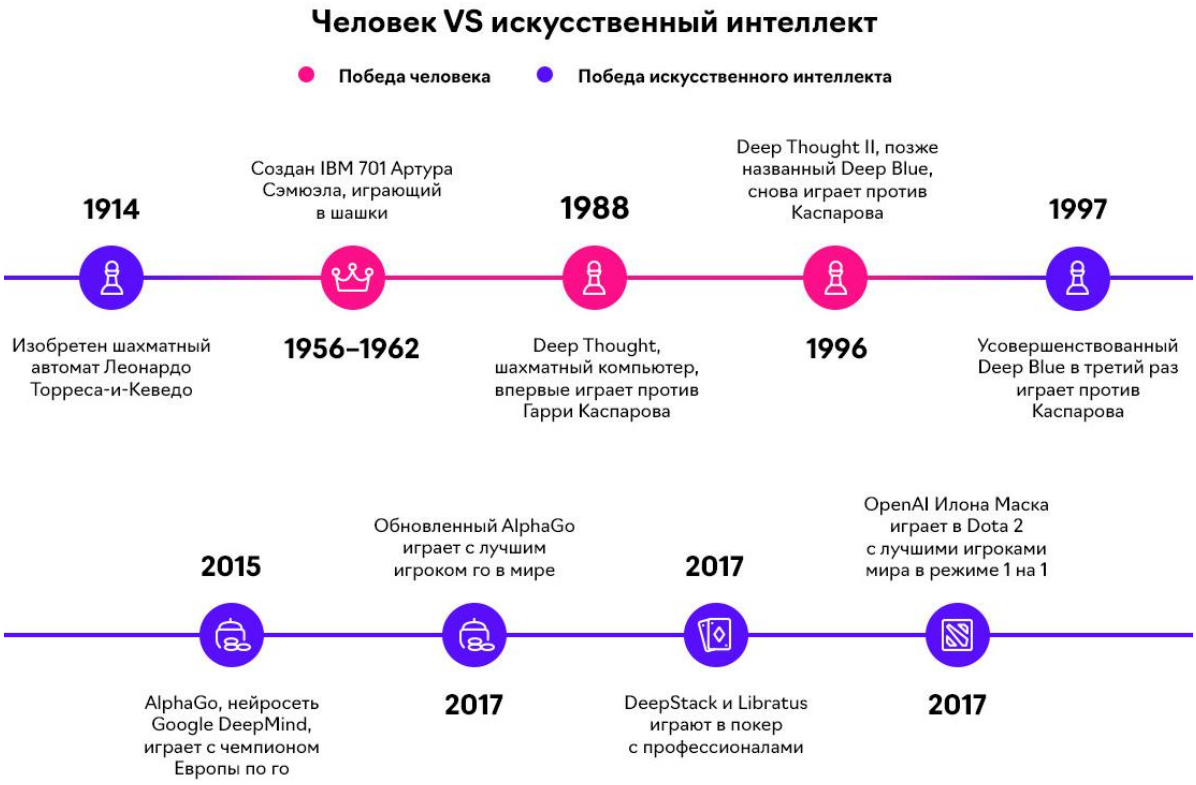


Рис. 1. Історія протистоянь людей та ШІ в настільних іграх за останні 100 років

Можна також зазначити, що багато експертів, таких як Елон Маск, Біл Гейтс та покійний нині Стівен Хокінг висловлювали певні острахи щодо розвитку ШІ та, щоб запобігти всякого роду неочікуваностей, був підписаний список з 23 пунктів розвитку ШІ (*Asilomar AI Principles*) [15]. Також відкритим залишається питання кібербезпеки ШІ.

1.2. Розвиток комп'ютерних шахів

Ще в 1951р., А. Тьюрінг написав алгоритм, за допомогою якого машина могла б грати в шахи, тільки в ролі машини виступав сам винахідник. Цей нонсенс навіть отримав назву – «паперова машина Тьюрінга». Алгоритм був досить умовний, і зберігся навіть запис партії, де «паперова машина» Тьюрінга програла одному з його колег. Через відсутність доступу до комп'ютера, програма жодного разу не перевірялась в роботі. В тому ж таки 1951 році, математик Клод Шеннон написав свою першу статтю про шахове програмування [18]. Він писав: *«Хоча, можливо, це й не має ніякого практичного значення, саме питання є, теоретично, цікавим, і сподіватимемося, що вирішення цієї задачі послужить поштовхом для вирішення інших задач аналогічної природи й більшого значення»*. К. Шеннон

також відзначає теоретичне існування найкращого ходу в шахах і практичну неможливість його знайти [5].

На даний момент основним рушієм в галузі розвитку ШІ є Google-команда DeepMind. Вони створили нейромережу Alpha Go, яка здатна за декілька годин гри, без жодних зовнішніх даних і впливів (знаючи тільки правила конкретної гри) та в режимі самонавчання вийти на рівень гри найсильніших людей в галузі (рис. 2) [6, 11]. Після кожного чергового матчу були доопрацювання і донавчання мережі, а відповідно виходили і нові версії програми. Їх сімейство називатимемо Alpha* [19] (Alpha Go → Alpha Lee → Alpha Master → Alpha Go Zero → AlphaZero → AlphaStar). Але що цікаво, якщо для шахів, го і шогі (японські шахи) Alpha* просто обирала швидше більш «людські» (інтуїтивні, нематематичні) ходи за рахунок алгоритму навчання з підкріпленням і для вибору з повного дерева рішень кількість можливих продовжень різко зменшувалася (в шахах замість $80*10^6$ позицій в секунду Stockfish'а розглядалися $70*10^3$, шогі – $40*10^3$ замість $35*10^6$ Elm'и, а в го – $16*10^3$, при тому що через складність гри і кількість комбінацій аналогів, як таких, не було взагалі [10]), то схоже що для версії гри в покер ШІ Libratus навчився блефувати, а DeepStack використовує інтуіцію! [7, 8, 9]

Суть методу навчання з підкріпленням полягає у взаємодії тестової системи (агента) із деяким середовищем окремими, дискретними в часі, кроками. Якісним нововведенням від DeepMind можна вважати відсутність подачі зовнішніх знань та використанні в якості середовища аналогічний до агента об'єкт. Таким чином, створили матч продукту самого з собою. Даний метод являється частковим випадком механізму навчання нейромережі з учителем, але в ролі вчителя являється не третя сторона, а саме середовище, що і є перевагою, оскільки алгоритм вчить сам себе. На початку тестування ШІ знає

лише правила гри, але з часом він поєднує свою щойно напрацьовану нейромережу із потужним алгоритмом пошуку.

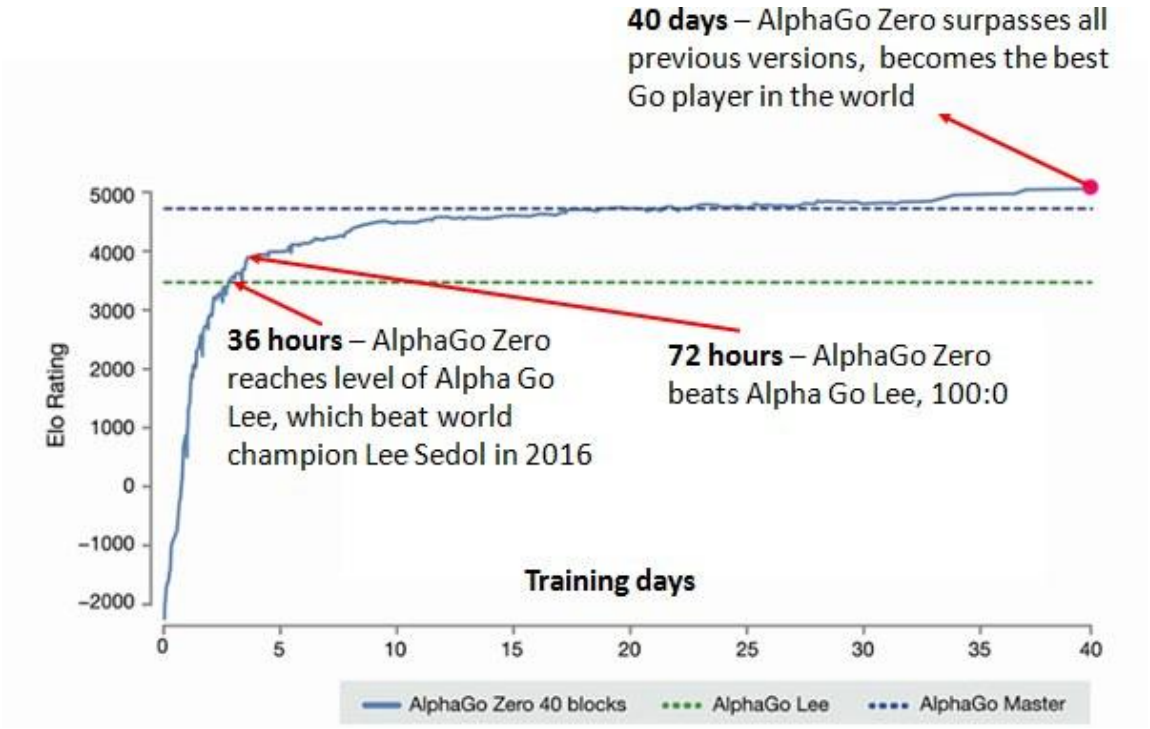


Рис. 2. Розвиток гри AlphaGo під час навчання

Далі, чим довше система думає, тим точніше налаштовується нейромережа і, відповідно, якісніше відбувається прогнозування ходів та вірогідного результату партії, на відміну від традиційних програм, які просто роблять перебір всіх можливих варіантів *(оскільки вони доступні, але їх занадто багато. Наприклад для шахів з дошкою 8x8, разом із неможливими за правилами позиціями, може виникнути* $\frac{64!}{32!*8!^2*2!^6} \approx 10^{43}$ *різних позицій. Це число називають числом Шеннона [18]. Для го з полем 19x19 таких позицій в рази більше)*, оцінюванням позиції на конкретній глибині та вибором ходу з кращою оцінкою (рис. 3 [11]). Навчання відбувається ітеративно,

продуктивність підвищується, що спричиняє суттєве посилення мережі та продукту вцілому.

Оцінювання відбувається наступним чином [12]. На кожному кроці t агент отримує стан s_t та із множини A дозволених дій обирає таку дію a_t, яка не заперечує правилам гри π. У відповідь агент отримує наступну позицію s_{t+1} та скалярне значення оцінки потенційного ходу – нагороду r_t Процес продовжується до тих пір, поки не буде отримана остаточна позиція, після чого процес перезапускається. Загальна оцінка «гілки» розраховуватиметься за формулою (1):

$$R_t = \sum_{k=0}^{\infty} \gamma^k * r_{t+k}\,, \tag{1}$$

де $\gamma \in (0;1]$ – імовірність вибору даної події на кроці.

Ціль агента – максимізувати результат оцінки ходу для кожної позиції.

Відсутність вхідних початкових знань, впливу вчителя-людини є перевагами, оскільки виключає вплив «недосконалого» людського досвіду на прийняття машинних рішень. Опубліковані партії з го вже стали певного роду канонами та їх починають вивчати та застосовувати на практиці на різних рівнях. Гросмейстер з шахів С. Шипов вже назвав AlphaZero новим шаховим Богом [13] і весь шаховий світ з нетерпінням чекає публікації усіх 100 зіграних партій із чемпіоном світу серед програм Stockfish (на даний момент доступно лише 10) [10].

Значення отриманих результатів алгоритму із самонавчанням важко переоцінити. Перш за все, до розв’язання стандартних задач можна знайти більш оптимальні і нетрадиційні підходи.

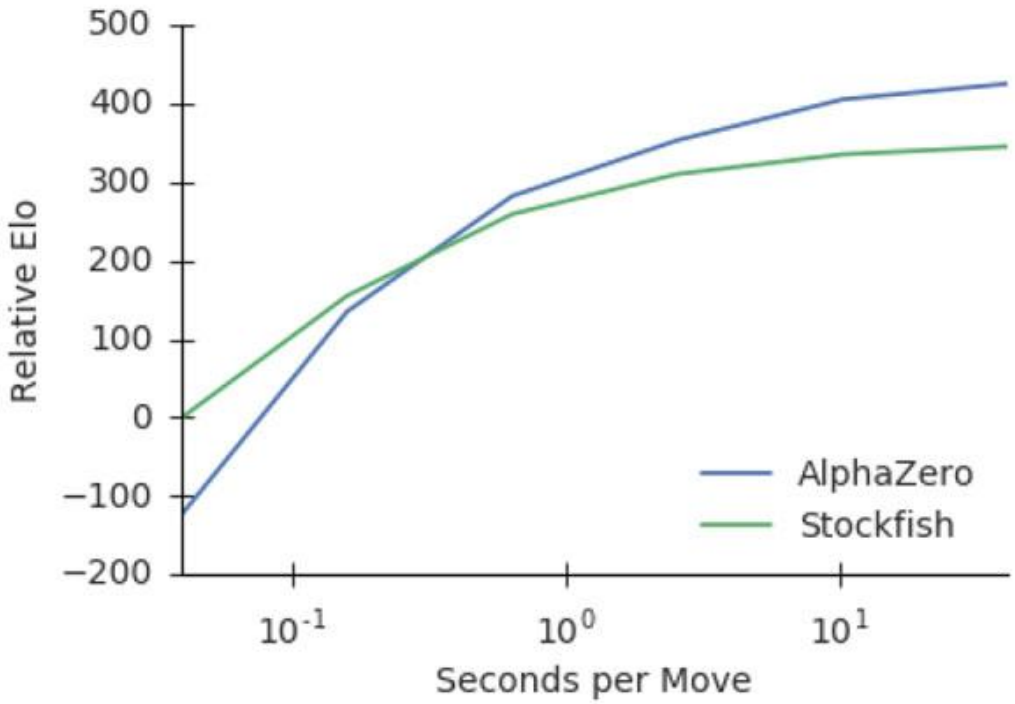


Рис. 3. Якість зробленого ходу в залежності від часу, затраченого на роздуми (AlphaZero в порівнянні зі Stockfish)

Самонавчання з підкріпленням, яке використовували Alpha*, дає можливість покращувати навіть ті підходи, які були запропоновані як оптимальні і нові на попередніх ітераціях. Імовірно, це дасть можливість людству осягнути новий рівень пізнання світу та стати більш винахідливими та інноваційними. Застосування таких механізмів у техніці дасть можливість спрогнозувати можливі катастрофи або їх наслідки, знайти способи зниження енерговитрат чи навіть знайти революційно нові джерела енергії, що теж є немало важливим.

Проте у даного підходу Google є й інша сторона медалі. Нейромережа для Alpha* повинна працювати (принаймні у період навчання) на спеціальному забезпеченні, спроектованому все тим же Google'ом: tensor processing units (TPU). AlphaZero використовує 5,000 TPU першого покоління (1 TPU Gen1

здатний виконати до 700 млн. операцій в секунду) для генерації ігор самої з собою, щоб навчити мережу або 64 TPU другого покоління для безпосереднього тренування. А це вимагає колосальних затрат енергії [14].

1.3. Людський алгоритм оцінки позиції та пошуку ходу

Все вище описане має відношення до навчання нейромережі та її роботи. Нейромережа оперує гілками розвитку після того чи іншого ходу, але для того, щоб вибрати ту чи іншу гілку, навіть живій людині, необхідно спочатку правильно оцінити позицію, спочатку поточну, а потім – яка станеться після якогось ходу.

Людський алгоритм оцінки позиції виглядає наступним чином [21]:

- матеріальне співвідношення сил
- мобільність фігур, структура їх розташування, взаємозв'язки
- спробувати з'ясувати ціль останнього ходу противника
- вибрати декілька ходів-кандидатів і для кожного з них:
    - знайти найсильнішу відповідь суперника
    - рекурсивно повторити всі попередні кроки в уяві для новоутвореної позиції і дійти до певної глибини
    - прийняти рішення залишити чи відсіяти той чи інший хід-кандидат на певному етапі
- обрати суб'єктивно найкращий хід серед ходів-кандидатів, які залишилися після відсіювання на попередньому кроці

Суб'єктивною ця оцінка є тому, що кожен гравець по-своєму може трактувати одну і ту ж позицію в різних умовах (бліц партія чи класична, психологічний стан, турнірне положення і т.д.). Саме через це, а також тому, що людські ресурси обмежені, та й тривалість партії не є нескінченною, гравці дуже часто помиляються при виборі того чи іншого плану гри.

Висновки до розділу 1

Отже, як видно з огляду відомих досліджень, рівень розвитку ШІ є досить високим, щоб перевершити людину, але це і заважає йому пройти тест Тьюрінга. Виходячи з цього міркування є цілком доцільним спробувати виключити фактор людської помилки і емоцій, застосувавши при цьому людський алгоритм оцінки шахової позиції і вибору подальшого плану гри.

РОЗДІЛ 2. РОЗРОБКА АЛГОРИТМУ ОЦІНКИ ПОЗИЦІЇ ТА ПРИЙНЯТТЯ РІШЕННЯ ЛЮДСЬКИМ ПІДХОДОМ

2.1. Порівняння існуючих алгоритмів оцінки позиції та вибору ходу.

На даний момент «кремнієві монстри» вміють оцінювати позицію математично. Для цього використовуються переважно алгоритми MiniMax (далі – мінімакс, ММ) та NegaMax (негамакс, НМ). Їх принципова різниця полягає в тому, що ММ асоціює білих як гравця на максимум, а чорних на мінімум, при чому всі розрахунки завжди виконуються на пошук максимуму (тобто за білих);

в свою чергу НМ виконує симетричну оцінку в залежності від того, чий зараз хід [20]. Формула оцінки змінюється в залежності від того, наскільки важливим є той чи інший фактор позиції для розробника, але базовою формулою оцінки позиції вважається формула, запропонована К. Шенноном [18]:

$$f\ p\ = 200 * \ K - K'\ + 9 * \ Q - Q'\ + 5 * \ R - R'\ + 3$$
$$* \quad B - B'\ + \ N - N'\ \ + 1 * \ P - P'\ - 0.5$$
$$* \quad D - D'\ + \ S - S'\ + \ I - I'\ \ + 0.1 * \ M - M'\ + \cdots$$

де, KQRBNP = кількість королів, ферзів, тур, слонів, коней та пішаків

D,S,I = здвоєні, блоковані та ізольовані пішаки

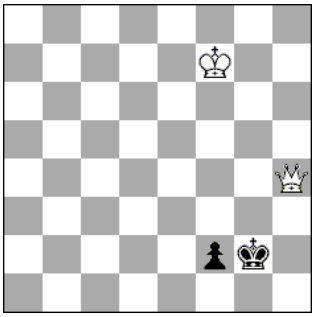
M = мобільність (кількість доступних легальних ходів)

Як видно, оцінка позиції це не що інше, як результат віднімання оцінки матеріальної складової сторони, яка повинна зробити хід, від аналогічної оцінки її опонента (позначено відповідно як K' Q', R'...) та мобільностей фігур. [20].

Але у такому випадку можуть виникнути ситуації, коли на дошці присутня матеріальна перевага однієї із сторін, але виграти неможливо, наприклад (Рис. 4):

a) крайній пішак і слон не того кольору, що і поле перетворення пішака
b) два коня проти «голого» короля
c) ферзь проти слонового пішака, що знаходиться за одне поле до перетворення.





a) b) c)

Рис. 4 – Теоретично нічийні позиції з матеріальною нерівновагою

Що це означає для комп’ютера? Згідно з оцінкою, по формулі є перевага, цілком імовірно що значна, тому варто грати на перемогу, але за правильної гри «слабшої» сторони виграти просто неможливо. Що робити в такому випадку? За правилами шахів якщо впродовж 50 ходів з дошки не зникла жодна фігура, або не було зроблено жодного ходу пішаком, або позиція повторилася тричі – присуджується нічия. Тому, нейромережа, зігравши певну кількість таких партій на перемогу, «розчарується» і заповнить такими позиціями свою власну базу знань як нічийними. Для звичайних шахових програм, які не вміють навчатися, все ще більш прозаїчніше: вже є готові бази знань: таблиці закінчень з сумарною кількістю фігур на дошці до 7, нічийні таблиці, дебютні книги та інші. З цього можна зробити висновок, що шахові програми є в більшості своїй статистичними, зазвичай вони не містять власної інформації про конкретну позицію, а дерево розрахунків час від часу може змінюватися. Як наслідок, робота програми ускладнюється за рахунок оцінки нерелевантних позицій, особливо на великій глибині, коли можливих розгалужень стає занадто багато (*середня глибина* розрахунків кожного конкретного варіанту комп’ютером по замовчуванню сягає близько 27-35 ходів, людина ж *завершує* свої розрахунки десь в межах 15-20). Тому, незважаючи на все нові й нові підходи ШІ в покращенні якості оцінювання, рівень саме розуміння комп’ютером шахів є досить слабким порівняно з людським рівнем. Оскільки знання вцілому є обмеженими, логічно припустити, що і методика отримання та розвитку цих знань теж буде обмеженою. Тому спробуємо запропонувати метод, за допомогою якого буде можливо перевірити чи попередні результати пошуку та

оцінок досі залишаються актуальними. Розглядатимемо його на невеликих глибинах розрахунків.

2.2. Базове обґрунтування нового підходу. Використання ланцюгів

Перш за все такий підхід буде більш «людським» оскільки в пам’яті зберігатиметься не позиція, а послідовність ходів. Саме таким чином прийнято вести запис партій (відома стартова позиція та послідовність зроблених ходів) та відбувається обдумування під час реальної гри.

Після того, як позиція була оцінена, а послідовність ходів збережена можна робити припущення, що в будь-якій іншій схожій позиції цю

послідовність і оцінку можна застосувати з мінімальною похибкою. З точки зору розрахунку варіантів це дасть нам значне покращення в швидкодії, оскільки при відтворенні вже відомої позиції нам не потрібно знову розраховувати оцінку та обирати план. Збережені послідовності ходів називатимемо ланцюгами, які в свою чергу породжують деревоподібні структури, але неглибокі.

Аналогічно відбувається і в реальній грі людиною. Людська пам’ять поділяється на дві: довгострокову та короткострокову. Коли з’являється відома структура, людина майже не задумуючись дістає потрібний ланцюг ходів із довгострокової пам’яті, яка постійно поповнюється. Але такий підхід не працюватиме для тактичних ходів, особливою категорією яких є приховані ходи, оскільки без спеціального пошуку побачити їх дуже важко. Для таких ситуацій пропонується створити репрезентаційну дошку (дошку відтворення) і на ній буде можливість розпізнати такі ходи без реалізації спецпошуку, адже навіть за умови врахування всіх тактичних загроз звичайний пошук може недооцінити їх силу і пропустити деякі ходи. Якщо ж таке все-таки сталося, то необхідно повторно перевірити деякі ходи, які націлені на ключові поля. Даний підхід можна буде вважати успішним якщо реалізований алгоритм не буде допускати явних промахів та зможе конкурувати із іншими сильними програмами.

Ланцюг динамічних ходів це коротка послідовність ходів від самого початку пошуку. Він є важливим оскільки виникає при виключенні з пошуку. Виключення з’являється в той момент, коли поточна оцінка спростовує часткову позицію і означає, що подальший аналіз цієї позиції непотрібний. В момент, коли було визначено значимий результат, результат пошуку може бути збережений із відміткою про можливість використання в подібних позиціях замість виконання нового пошуку. Будь-яка зміна в позиції може спричинити

зміну оцінки, тому неможливо 100%-во відтворити ту ж саму послідовність ходів, тому робити переоцінку необхідно весь час, але збережена оцінка дає можливість вибрати ходи (послідовності)-кандидати, для яких власне і потрібно робити розрахунки. Якщо вникає ситуація, коли на черговий хід не можна покластися (оцінка дуже відрізняється від збереженої або знаходяться нові ресурси і дана гілка стає невалідною) необхідно оновити оцінку і зробити додатковий пошук. Таким чином необхідно враховувати ризики та зробити вибір між невеликою величиною обрізки з потребою пізніше зробити екстрапошук та більш легким повним αβ-пошуком [22], але постійно. Даний алгоритм вимагає також реалізацію таблиць подібних ходів.

Ланцюги ходів є більш тимчасовими та перехідними, проміжними. Вони створюються та оновлюються впродовж гри, але не зберігаються. В зазначених таблицях зберігається більш узагальнена інформація гри і може бути збережена до файлу. В будь-якій наступній грі ці таблиці можуть бути вичитані, оновлені, знову збережені. В такому випадку вони репрезентують розвиток шахової програми, її досягнення [25].

2.3. База знань

Отже, було визначено необхідність використання таблиць динаміки, або таблиці пам’яті, та поверхневого аналізу позиції в процесі пошуку. Це дасть можливість використовувати партії гросмейстерів та розіграшу цих партій для поповнення своєї бази оцінок. Таблиці ходів в подальшому використовуються короткостроковою та довгостроковою пам’яттю, можуть бути додатково згенеровані програмою.

Таблиці ходів можна порівнювати із дебютними книгами. Переваги над останніми очевидні: економія місця, швидший пошук, можливість помістити в масив, ширший діапазон.

Таблиці покращуються шляхом прив’язки полів між ними, таким чином щоб можна було відслідкувати коректну послідовність ходів. Наприклад, хід Kf3 може належати більш ніж одному ланцюгу. На рисунку можна побачити, що собою представляють таблиці ходів. Великий білий квадрат представляє собою дошку, тому для класичних шахів являє собою масив на 64 елементи. Маленькі чорні квадрати відповідають зробленому ходу, які щойно додані і, відповідно, збільшують лише значимості оцінки цього ходу. Формат запису ходу: (W/B) – білі /чорні, (P/N/B/R/Q/K) – пішак/кінь/інші фігури, і останнім іде зроблений хід: звідки і куди.

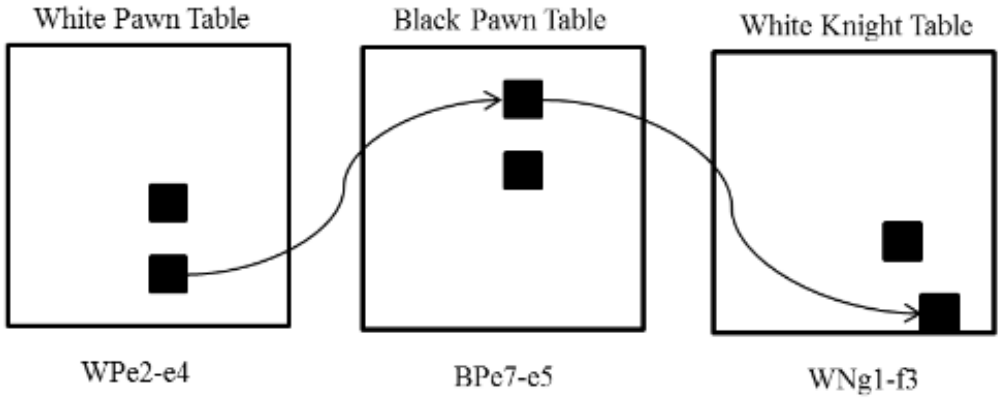


Рис. 5 Приклад таблиць ходів та зв’язків між ними

Коротко- та довгострокова пам’ять формується на основі таблиць ходів, але використовуються по-різному. Послідовність ходів зберігається в клітинці, на якій знаходиться перша фігура в ланцюгу, як показано на Рисунку. Короткострокова пам’ять може використовувати послідовність як 1 шар, так і більше. Це може бути окремою структурою для довгострокової пам’яті і будь яке входження може бути додане чи видалено негайно. Як результат, ми

отримуємо один із видів «вбивчої евристики» - можливість самостійно знаходити чи вивчати найкращі ходи по всій дошці.

Довгострокова пам’ять використовує більш постійну версію таблицю, коли 3-шаровий хід (хід, з трьома можливими продовженнями) перетворюються в структуру з довгострокової пам’яті. В цьому випадку, коли є співпадіння по позиції, ми знаходимо відповідний базовий хід і відтворюємо потрібну гілку. Значимість та оцінка відповідної гілки може змінитися в подальшому. Перезбереження таблиць до бази даних можливе і це надає можливість здобувати нові знання під час будь якої нової гри. Тим не менш, це не призводить до значного зростання бази в розмірах, оскільки по суті відбувається лише зміна важливості гілки, ніж її підкріплення і це призводить до того, що більш звичні ходи будуть використовуватися в типових позиціях. Після того, як послідовність додана, більш звичною практикою являється відтворення і оновлення саме цієї гілки, ніж додавання нової.

2.4. Сильні і ключові поля

Ці таблиці ходів потім сприяють появі шаблону найкращих ходів для кожного типу фігур протягом гри або етапу пошуку. Наприклад, на рисунках 6 і 7 показано деякі статистичні дані аналізу пошуку, які можуть бути корисними для обґрунтування, заснованого на знаннях. Кожну фігуру можна розглядати окремо, але, як правило, результати можуть бути об'єднані. Якщо ви намагаєтеся отримати певну форму знань з таблиць ходів, то використання «найсильнішого» поля насправді може бути не найвигіднішим ходом.

Наприклад, на рисунку 6 показані таблиці переміщення ферзя і слона для вказаної позиції, при цьому відображаються лише найкращі додатні зважені коефіцієнти. Легко побачити, що кращі ходи білих націлені на використання слабкостей королівського флангу чорних. В той же час зважені поля вказують на 2 конкретних ходи, які дещо розмивають отримані знання. Наприклад, якщо можна зробити декілька ходів з одного поля, і якщо один з них можна виділити як найкращий, то інші будуть зменшувати свої значення ваги і можуть стати дуже малими, або вони можуть взаємно зменшувати вагу один одного, і тому ходи, які часто розглядаються, можуть отримати негативні значення.

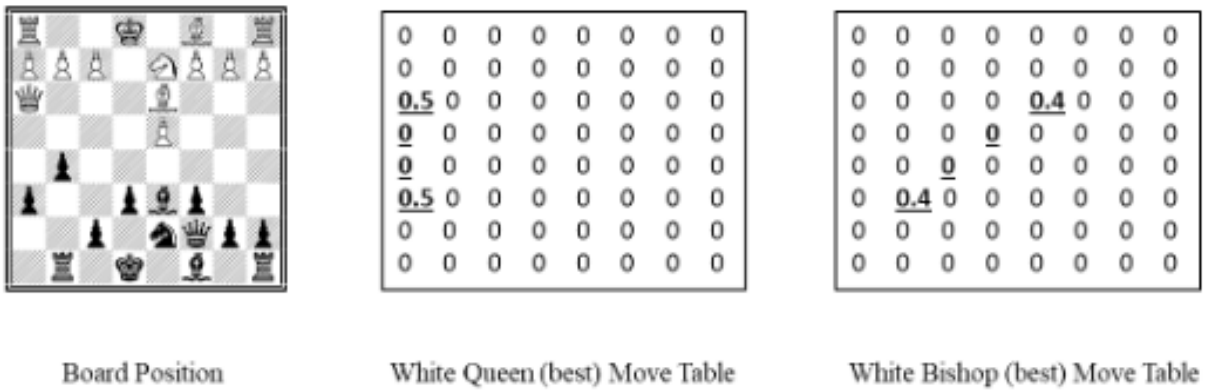
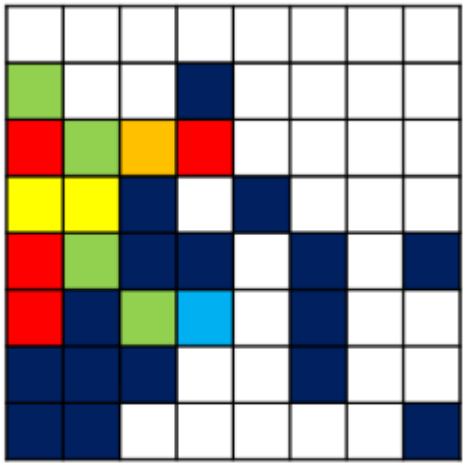


Рис. 6. Таблиці ходів ферзя та слона в заданій позиції

Як альтернатива, на рисунку 7 показано зібрані значення ваги усіх можливих збережених ходів білого ферзя. Варто звернути увагу, що значення поля в даній таблиці є від’ємними, окрім значення 0, які означають, що для даного поля не збережено жодного доступного ходу і воно не використовується в пошуку. Таким чином легко зрозуміти, що найкращим ходом буде поле з вагою, яка є від’ємною, але найбільшою за модулем.

0	0	0	0	0	0	0	0
-246	0	0	-65	0	0	0	0
-802	-213	-459	-521	0	0	0	0
-335	-317	-98	0	-62	0	0	0
-739	-202	-82	-47	0	-17	0	-64
-975	-61	-259	-195	0	-17	0	0
-36	-55	-93	0	0	-63	0	0
-60	-81	0	0	0	0	0	-16

White Queen (accum) Move Table



Relative Importance Map View

Рис. 7. Зібрані сили ходів білого ферзя та відповідна їм карта важливості/значимості

Таким чином є багато міркувань щодо найкращого значення і одна з ідей це отримання абсолютного значення сумарної ваги для демонстрації «важливості» поля для фігури та найкращої ваги для відображення найсильнішого поля. В подальшому сильні і слабкі поля беруться до уваги при виборі плану гри, наприклад при захопленні відкритих ліній. Для візуалізації цієї оцінки на рисунку 7 є карта важливості полів, яка використовує лише їх зваженості: найбільш важливими є червоні поля, далі по пріоритету оранжеві, жовті, зелені, блакитні та темносині як найслабші. Білі поля не враховувалися взагалі.

2.5. Ряди оціночної функції

Повертаючи оцінку для шахового ходу з таблиці можна зіткнутися з проблемою того, що повертати, оскільки це значення може відрізнятися в залежності від позиції. Для вирішення цієї задачі необхідно зберігати N останніх оцінок в порядку від найновішої до найстарішої. Такий формат надасть

найбільшої значимості найновішій оцінці. Якщо часовий фактор не береться до уваги, тоді необхідно використовувати щось накшталт:

$$E_1 + \frac{E_2}{2} + \frac{E_3}{3} + \cdots,$$

де E_1 найновіша оцінка, E_2 – попередня до E_1 і т.д.

Але оскільки оцінки можуть швидко змінюватися, то значимість E_1 дуже швидко розсіється. Тому, більш придатною альтернативою визначимо наступний ряд:

$$\frac{(E_1 + \frac{E_1 + E_2}{2} + \frac{E_1 + E_2 + E_3}{3} + \cdots + \frac{E_1 + E_2 + \cdots + E_n}{n})}{n}$$

Використовуючи такий ряд ми зробимо спадання значимості E_1 при послідовності E_1, E_2 і т.д. більш плавним. Таким чином дана функція є підходящою для згладжування важливості відповідних оцінок. З точки зору інтелекту такий ряд цікаво розглянути як послідовність нейронів мозку, коли перший «запалений» нейрон є найбільш важливим (в нашому випадку - найновіший) і останній в черзі – найменш важливий.

2.6. Стратегія та аналіз характеристик

Аналіз характеристик надає можливість порадити ключові поля, які потрібно використати при виборі ходу. В той час, як перевага у просторі вказує на область для вторгнення, ключові поля є більш конкретними цілями. В це поняття можна включати як поля, що часто використовувалися раніше, взяті з уже наявної бази знань, так і ключові поля, отримані з поточного пошуку. На відміну від популярних ходів, взятих з бази знань, побудованої на результатах

ігор, ключові поля можна отримати лише з характеристичного аналізу, зробленого під час конкретної поточної гри.

2.6.1. Характеристики з бази даних

Цілком нормальною практикою можна вважати побудову базу даних типових позицій, найчастіших ходів у них та отриманих результатів. База індексується по зробленому ходу, а потім зберігається статистика про частоту появи кожної фігури на конкретному полі. Наприклад, береться 3 шари послідовностей ходів із довготривалої пам'яті. Це вимагає певної постійності у грі, проте все ще може бути зіграно в партії. Також, та чи інша послідовність повинна бути додана при відтворенні щонайменше тричі, що означатиме можливість відтворення певної підструктури в різних позиціях. Маючи $2\text{-}3*10^6$ ходів, можна згенерувати таку базу розміром близько 20 Мб, що доволі небагато.

2.6.2. Популярні та ключові поля

Ходи на популярні поля, що є частиною αβ-пошуку, можна також визначити із бази даних частоти ходів, яка щойно була описана. Частотна дошка фігури (дошка, де відмічена статистика/імовірність появи фігури на конкретному полі в той чи інший момент гри) може бути відтворена разом з найпопулярнішими полями. Якщо якісь ходи-кандидати співпадають з положенням якихось фігур тоді ми додаємо їх в чергу пошуку. Вторгнення в такому випадку відбувається 2ма шляхами: безпосереднім потраплянням на поле, або нападом на конкретне поле після зробленого ходу. Якщо при пошуку завжди обираються тактичні ходи, тоді популярні поля можуть «витіснити» інші «спокійні» ходи або ходи без взяття. Тому при реалізації нейромережі варто певним чином сортувати такі ходи за категоріями або розставляти додаткові пріоритети.

В той час як популярні ходи це звичайне статистичне число, кількість/частота, то інший тип аналізу повертає характеристики як результат порівняння двох критичних позицій: першої – основної, кореневої - та другої, яка є позицією із підпослідовності по дереву пошуку, коли змінюється значення оцінки. Різниця в позиціях може бути визначена як додатна, підсилююча, так і від'ємна, послаблююча за будь-яку сторону. Перш за все різницею можна назвати зміну наявності якоїсь фігури між двома позиціями. Цей процес показує той момент, коли послаблюється значення оцінки проте все ще є можливість зробити якийсь безпечний хід, але робити це недоцільно. Після визначення такого роду полів, якщо знаходиться хід, який впливає на їх значимість, то такий хід передодається до черги кандидатів.

2.6.3. Приховані тактичні ходи

Ще один тип ходів, які можуть бути не розпізнані та не враховані в пошуку – зв'язки (рисунок 8). Це ходи тактичні, але дещо приховані, але цілком природньо очікувати від механізму пошуку, що він повинен їх бачити.

Рис. 8. Непряма атака чорного слона e7 на ферзя білих a3

На рисунку 8 можна побачити позицію, яка була отримана після «спокійного» ходу Фа4-а3. Хід виглядає безпечним, але ним білі стали під прихованого Се7. Між ними знаходяться дві фігури чорних, але форсованими ходами (d5, ed, Ne4, Re4) діагональ звільняється і слон забере ферзя на а3 з виграною позицією. Але недоліком алгоритму відсікання вперед є те, що ці

форсовані ходи є «небезпечними», тому що ведуть до тимчасової втрати матеріалу, а отже можуть не включитися до пошуку. Проте є шанс розпізнати такий сценарій звичайними методами не використовуючи пошук. Все що необхідно для цього знати: відносна вартість фігур та як ці фігури ходять. Коли згенеровано послідовність ходів, то нова дошка характеристик буде також створена задля збереження та оцінки будь-якої далекобійної фігури та лінії/діагоналі, по яким ця фігура атакує. Для цієї фігури також будуть відмічені всі фігури, що стоять на її шляху. В даному випадку для чорного слона e7 буде створена така таблиця, оскільки слон – далекобійний по діагоналям. Таким чином, дві чорні фігури та білий ферзь на a3 будуть знайдені. В найгіршому випадку обидва противники упустять можливість ударити фігуру, але взяття буде легальним. Порахувавши сумарну вартість фігур по діагоналі, то стане зрозумілим, що навіть втративши 2 своїх фігури та забравши ферзя білих, чорні матимуть матеріальну перевагу. Тому, рішенням може бути вибір усіх ходів (безпечних і не тільки) цих фігур, то такий тип тактики повинен включитися в пошук. Звичайно, це зайві рухи, яких потрібно уникати, але це рідкісні випадки.

Якщо в будь-якому випадку позиція має бути проаналізована, то алгоритм відрізання вперед являється неефективним. Отже, необхідно, щоб алгоритм вмів розпізнати таку структуру за певним шаблоном.

2.7. Новий алгоритм пошуку

Описавши вище всі ключові моменти, тепер можна запропонувати новий алгоритм пошуку для комп’ютерних шахів, але максимально наближений до людських роздумів.

Перш за все необхідно розставити типи і пріоритети для ходів-кандидатів:

1. Безпечні ходи-взяття. Безпечні форсуючі та форсовані ходи.
2. Інші безпечні ходи

3. Ненадійні та небезпечні взяття (призводять до матеріальної нерівності), ненадійні форсуючі та форсовані ходи
4. Ненадійні інші ходи.

Перш за все необхідно відділити безпечні та ненадійні ходи. Ненадійні найчастіше характеризуються зміною матеріальної рівноваги, небезпечні – відкритими загрозами королю чи ферзю. Форсуючі ходи це ті, які *змушують* противника ходити певною фігурою чи на певне поле через створення певних загроз. Форсовані – власне ті ходи, які є відповіддю на форсуючі ходи. Взяття – хід, коли відбувається захоплення чужої фігури та зняття її з дошки, а інші ходи – це ходи на пусте поле без негайної втрати матеріалу чи позиційних характеристик.

2.7.1. Тактичні ходи

Тактичні ходи необхідно враховувати завжди і власне безпечні тактичні ходи є результатом повного αβ-пошуку. Тим не менш першими ходами в послідовностях для αβ-пошуку є безпечні взяття, форсуючі та форсовані ходи. Завжди корисно додати до списку пошуку ненадійні взяття, оскільки послідовності пошуку можуть включати фіксовані або прив’язані фігури, що не є одразу очевидним.

2.7.2. Позиційний аналіз

Через використання відсікання вперед можуть з’являтися певні позиції, які потрібно в подальшому проаналізувати на наявність важливих і критичних характеристик в позиції. При даному трактуванні нового алгоритму, це є недоліком в порівнянні з легким процесом повного перебору, але це приймається як напрям подальшого розвитку даного алгоритму.

2.8. Прийняття рішення. Тестування алгоритму

Після поверхневих оглядів усіх ходів-кандидатів, зазвичай в конкретній позиції залишається зробити вибір між 2 основними – 1 отриманий з повного перебору і 1 з пошуку в довгостроковій пам'яті. Якщо оцінка останнього краща, то новий пошук виконується лише для першого – це називається фінальною або остаточною перевіркою. Тут проходить пошук з повним перебором, але лише для одного ходу. Якщо результат досі слабкий, то необхідно виконати додавання ходів з використанням ключових полів. Після даної перевірки робиться вибір щодо кращого ходу в даній позиції.

На даний момент комп'ютерна програма знаходиться на етапі розробки та покращень. Було проведено 2 типи ранніх тестувань. Результати наведені нижче. 3 тест заключатиме в собі тестування з іншими програмами.

Перш за все проведено перевірку використовуючи дебютні бази для того, щоб показати що використання таблиці ходів з динамічними ланцюгами мають не тимчасовий чи випадковий ефект в режимі гри в реальному часі. Можна обирати випадковий дебют кожної гри і щоразу отримувати покращення з мінімальними затратами. Даний алгоритм показав результат 7-3 в 10 5-хвилинних іграх.

Тести на варіативність довготривалої пам'яті. Щоб побачити, як таблиці ходів впливають на розміри пошуку, протестуємо додавання 1, 2 та 4 ходів для початку процесу пошуку ланцюгів ходів. В статті [26] показано як підхід відсікання вперед використовуючи динамічні ланцюги ходів може покращити пошуку на 90% порівняно зі стандартним підходом, а тому це чудова область для досліджень та розширення можливостей, додаючи використання додаткових знань у вигляді таблиць ходів. В таблиці 1 показано результати відповідних об'ємів області пошуку для динамічних ланцюгів ходів з та без використання додаткових таблиць довготривалої пам'яті. Ці дані були отримані шляхом

прогону повної партії з 98 окремих позицій (49 ходів) до глибини пошуку 5 півходів. Результати отримані лише за використання алгоритму негамаксу альфа-бета пошуку і показують наскільки розміри таблиці пошуку менші в порівнянні зі стандартними підходами. Ланцюги ходів використовували пошук вершини в 23 рази менше ніж традиційні пошуки в ширину. Варто зазначити, що даний алгоритм не бере до уваги якість і силу ходу, лише зменшує область пошуку.

Таблиця 1 – Результати застосування ланцюгів ходів та таблиць ходів

Тип пошуку	**Ланцюги ходів**	**Ланцюги ходів + таблиці ходів (1 хід)**	**Ланцюги ходів + таблиці ходів (2 хід)**	**Ланцюги ходів + таблиці ходів (4 хід)**
Пройдено вершин	545	13367	12117	12550

Цікаво те, що додавання лише одного ходу до таблиці значно підвищує розміри області пошуку (98%), але додавання більш як одного ходу не має аналогічного ефекту. Таким чином, незалежне використання короткострокових ланцюгів ходів шукає з середнім коефіцієнтом розгалуження 3-4. Додавання принаймні 1 ходу збільшує цей коефіцієнт до 6-7, що є стандартним розміром. Подальше збільшення не впливає з таким ефектом на область пошуку. Отже, при продовженні пошуку в глибину, в момент коли всі можливі підходи фільтрації та відсікання вже застосовані, необхідно ще додавати функціонал для визначення сили ходу, пріоритизацію гілок. Інший можливий варіант використання алгоритму – застосовувати його лише в 30% ситуацій, використовувати як частину альфа-бета алгоритму. Але це те, що стосується створення нової таблиці для конкретно взятої окремої позиції. Якщо ж таблиця

вже була побудована на попередніх кроках, то на даному кроці вона може бути більш значимою.

Висновки до розділу 2

Отже, як видно з аналізу та результатів тестування, людський механізм оцінки є цілком прийнятним, щоб застосовувати та розвивати його в поєднанні з чисто математичними методами оцінки позиції. Це допоможе розвивати використання аналогічного підходу і в інших галузях застосування ШІ. Людський фактор помилки виключається за рахунок відсутності емоцій. Таблиці ходів є досить економними в режимі реального часу.

ЗАГАЛЬНІ ВИСНОВКИ

В даній роботі представлені етапи становлення комп'ютерних шахів як певного розділу штучного інтелекту, проведено порівняння та аналіз методів оцінки позиції та вибору кращого ходу штучним інтелектом та людиною, запропоновано концепції нового алгоритму пошуку для комп'ютера з використанням ключових моментів людської оцінки позиції.

Відповідно до поставлених завдань в роботі зроблено:

– досліджено етапи розвитку, застосування та оцінки якості штучного інтелекту;
– проаналізовані та порівняні способи оцінки позиції людиною та комп'ютером;
– розроблено алгоритм вибору наступного ходу шляхом використання таблиць ходів замість таблиць позицій;
– розроблено логічну структуру бази даних ходів;

Перевірити якість даного алгоритму на практиці можна буде після безпосередньої його реалізації. Перевірка планується проводитися наступними діями:

- гра з реальними людьми;
- гра з іншими комп'ютерними програмами, які є у вільному доступі;
- гра з нейромережею lc0 (LeelaChessZero) – неофіційний аналог AlphaZero з відкритим кодом та платформою для тестування;
- використання при розв'язанні шахових задач.

В ході роботи основними недоліками вказано «недалекоглядність», але він компенсується швидкістю пошуку, та можливі неврахування прихованих загроз. Усунення цих недоліків будуть братися як цілі для подальших досліджень.

При аналізі існуючих підходів використано результати, опубліковані в статті «Тенденції розвитку штучного інтелекту розглянутого на базі нейромереж із самонавчанням на прикладі програм для настільних ігор»[19].

Підводячи підсумки, також хотілося б відзначити найяскравіші та найважливіші досягнення ШІ за останні декілька років [16, 17, 23]:

- Sketch2Code - Microsoft навчила ШІ програмувати сайти по зображенню «від руки»
- AlphaGo переміг чемпіона світу по грі в Го. AlphaZero переміг чемпіона світу з комп'ютерних шахів Stockfish8 (Людина не може виграти в «традиційних» шахових програм з початку 2000-х).
- Libratus та DeepStack перемогли кращих гравців в покер (таке тривалий час вважалося нездійсненним, оскільки крім математичної складової, важливу роль відіграє психологія гравців і уміння блефувати).
- Програма AlphaStar, змогла обіграти двох професійних гравців в стратегію реального часу StarCraft II. Кожного з гравців-людей нейромережа перемогла в п'яти матчах.
- Самокерована машина Tesla доставила в лікарню людину з серцевим приступом
- Штучний інтелект спрогнозував результати Кентуккійського Дербі
- Microsoft AI на сьогоднішній день розуміє людську мову краще самих людей та навіть зумів поступити до університету
- ШІ спричинив революцію в діагностиці раку, передбачає приступи епілепсії, підвищив результати операцій з трансплантології
- Завдяки самонавчанню, дрони зі ШІ DroNet літають без gps і карт

Як видно із досліджень, тестувань та поточних впроваджень в повсякденне життя, ШІ має дуже широкі сфери застосування. Але утримання в своїх будинках сервери, здатні виконувати роботу, аналогічну до кількатисячної

системи TPU не виглядає можливим, принаймні найближчим часом. Тому найбільш актуальними напрямками для покращення ситуації є оптимізація алгоритмів вибору ходу, пониження навантажень на систему, покращення критеріїв вибору дії на більш ранній стадії. Тим не менш, чекатимемо також і того, чи буде Alpha* приймати участь в проходженні тесту Тьюрінга найближчими роками.

Результати дипломної роботи опубліковано в працях [19] і [24].

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Штучний інтелект [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Штучний_інтелект.

2. Artificial intelligence [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Artificial_intelligence

3. Тест Тьюрінга [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Тест_Тьюрінга

4. История соревнований ИИ и человека: кто кого [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://vc.ru/flood/39184-istoriya-sorevnovaniy-ii-i-cheloveka-kto-kogo.

5. Комп'ютерні шахи [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Комп'ютерні_шахи

6. Python: ИИ для “Четыре в ряд” с алгоритмом AlphaZero [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://proglib.io/p/connect4-alphazero/.

7. Николенко С. И. что же делают alphago и alphazero? глубокое обучение с подкреплением [Електронний ресурс] / Сергей Игоревич Николенко // Data Science UA Conference 2018. – 2018. – Режим доступу до ресурсу: https://logic.pdmi.ras.ru/~sergey/slides/N18_DataScienceUAReinforcement.pdf.

8. Yakovenko N. CMU’s Libratus Bluffs its way to Victory in #BrainsVsAI Poker Match [Електронний ресурс] / Nikolai Yakovenko. – 2017. – Режим доступу до ресурсу: https://medium.com/@Moscow25/cmus-libratus-bluffs-its-way-to-victory-in-brainsvsai-poker-match-99abd31b9cd4.

9. Expert-Level Artificial Intelligence in Heads-Up No-Limit Poker [Електронний ресурс]. – 2016. – Режим доступу до ресурсу: https://www.deepstack.ai/.

10. Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm [Електронний ресурс] / D.Silver, T. Hubert, J. Schrittwieser, DeepMind. – 2017. – Режим доступу до ресурсу: https://arxiv.org/pdf/1712.01815.pdf.

11. Silver A. The future is here – AlphaZero learns chess [Електронний ресурс] / Albert Silver. – 2017. – Режим доступу до ресурсу: https://en.chessbase.com/post/the-future-is-here-alphazero-learns-chess.

12. Asynchronous Methods for Deep Reinforcement Learning [Електронний ресурс] / [V. Mnih, A. Badia, M. Mirza та ін.]. – 2016. – Режим доступу до ресурсу: https://arxiv.org/pdf/1602.01783.pdf.

13. Шипов С. AlphaZero - новый бог шахмат! [Електронний ресурс] / Сергей Шипов // Crestbook Шахматы (Відео-канал). – 2017. – Режим доступу до ресурсу: https://www.youtube.com/watch?v=ba4_M7UINfo.

14. What's Inside AlphaZero's Chess Brain? [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://www.chess.com/article/view/whats-inside-alphazeros-brain.

15. Asilomar AI Principles [Електронний ресурс]. – 2017. – Режим доступу до ресурсу: https://futureoflife.org/ai-principles/.

16. Романов В. 6 наиболее важных достижений искусственного интеллекта в 2016 году [Електронний ресурс] / Владислав Романов. – 2016. – Режим доступу до ресурсу: https://www.techcult.ru/technology/3816-dostizheniya-ai .

17. Степанов Д. Microsoft научила ИИ программировать сайты по картинке от руки [Електронний ресурс] / Дмитрий Степанов. – 2018. – Режим доступу до ресурсу:http://www.cnews.ru/news/top/2018-08-28_microsoft_nauchila_ii_generirovat_sajty_po_eskizu.

18. Shannon C. Programming a Computer for Playing Chess [Електронний ресурс] / Claude Shannon // Philosophical Magazine. Т. 7/41. – №314. – С. 256–275. – 1950. – Режим доступу до ресурсу: http://archive.computerhistory.org/projects/chess/related_materials/text/2-0 and 2-1.Programming_a_computer_for_playing_chess.shannon/2-0 and 2-1.Programming_a_computer_for_playing_chess.shannon.062303002.pdf.

19. Руденко С. Тенденції розвитку штучного інтелекту розглянутого на базі нейромереж із самонавчанням на прикладі програм для настільних ігор [Електронний ресурс] / Сергій Руденко. // Науково-популярний альманах "Математика та інформатика навколо нас". – 2018. – №2. – С. 219 – 229. – Режим доступу до ресурсу: https://fmft.vspu.edu.ua/wp-content/uploads/2019/01/Альманах_випуск-2_2018_.pdf

20. Evaluation [Електронний ресурс] – Режим доступу до ресурсу: https://www.chessprogramming.org/Evaluation

21. Чеботарев О. В. Уроки шахматной стратегии / Олег Владимирович Чеботарев. – Москва: Воениздат, 1981. – 126 с.

22. Alpha-beta pruning [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Alpha-beta_pruning

23. AlphaStar: Mastering the Real-Time Strategy Game StarCraft II [Електронний ресурс] – Режим доступу до ресурсу: https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/

24. Руденко С. Модифікація технології оцінки шахової позиції комп’ютерною програмою / С. Руденко, С. Бак // Матеріали ІІ Всеукраїнської науково-практичної Інтернет-конференції «Математика та інформатика у вищій школі: виклики сучасності» (Вінниця, 15-16 травня 2019 р.) [Електронне

наукове видання]: збірник матеріалів. – Вінниця, 2019. – С. 32-36. – Режим доступу до ресурсу https://conference.vspu.edu.ua/public/conferences/2/schedConfs/2/MIinHS2019_materials.pdf

25. A More Human Way to Play Computer Chess [Електронний ресурс] / Kieran Greer. Distributed Computing Systems, Belfast, UK – 2019. – Режим доступу до ресурсу: https://arxiv.org/ftp/arxiv/papers/1503/1503.04333.pdf .

26. Greer, K., (2000). Computer Chess Move Ordering Schemes Using Move Influence, Artificial Intelligence Journal, Vol. 120, No. 2, July, pp. 235-250.