УДК 004.023
Марченко О.О., аспірант
Марченко О.І., к.т.н., доцент,
Національний технічний університет України «Київський політехнічний інститут»

КРИТЕРІЙ «ГЛИБИНА-ШИРИНА» ДЛЯ КОНТРОЛЮ ФОРМИ ДЕРЕВА ПОШУКУ ПРИ ВИКОРИСТАННІ МЕТОДУ МОНТЕ-КАРЛО

Марченко О.О., Марченко О.І. Критерій «глибина-ширина» для контролю форми дерева пошуку при використанні методу Монте-Карло. У статті, на основі аналізу методу пошуку в дереві методом Монте-Карло (MCTS) та особливостей його поведінки при використанні в різних випадках, запропоновано оригінальний критерій «глибина-ширина» дерева для контролю форми дерева пошуку, а також для оцінювання і вибору потенційно кращих варіантів продовження такого пошуку. Передбачається, що цей критерій може бути використаний для розробки нових способів покращення методу MCTS та створення більш швидких реалізацій MCTS для конкретних застосувань.
Ключові слова: задачі штучного інтелекту, дерева ігор, пошук в дереві, метод Монте-Карло, MCTS, методи покращання MCTS.

Марченко А.А., Марченко А.И. Критерий «глубина-ширина» для контроля формы дерева поиска при использовании метода Монте-Карло. В статье, на основе анализа метода поиска в дереве методом Монте-Карло (MCTS) и особенностей его поведения при использовании в различных случаях, предложен оригинальный критерий «глубина-ширина» дерева для контроля формы дерева поиска, а также для оценивания и выбора потенциально лучших вариантов продолжения такого поиска. Предполагается, что этот критерий может быть использован для разработки новых способов улучшения метода MCTS и создания более быстрых реализаций MCTS для конкретных приложений.
Ключевые слова: задачи искусственного интеллекта, деревья игр, поиск в дереве, метод Монте-Карло, MCTS, методы улучшения MCTS.

Marchenko O.O., Marchenko O.I. "Depth-breadth" criterion for control of the search tree shape using Monte-Carlo method. Basing on analysis of the Monte-Carlo tree search (MCTS) method and specific features of its behavior for various cases of usage, a new tree "depth-width" criterion for controlling search tree shape along with for estimation and choose of potentially better options for the search continuation. It is supposed that this criterion could be used for development of new techniques of MCTS method improvement and for creation of faster MCTS implementations for concrete applications.
Keywords: artificial intelligence tasks, game trees, tree search, Monte-Carlo method, MCTS, MCTS improvement methods.
Рис. 3. Літ. 9.

Постановка наукової проблеми.

Розробка нових методів та алгоритмів, що можуть прискорити обчислення задач, час рішення яких зараз є занадто великим, була і залишається актуальною проблемою сьогодення. Особливо це важливо в галузі задач штучного інтелекту, яка відзначається надзвичайно великим обсягом даних і високими вимогами до швидкості їх обробки. Характерними і популярними задачами штучного інтелекту, на яких, як правило, виконується апробація нових методів, алгоритмів, структур даних є складні інтелектуальні ігри, наприклад, такі як ігри в шахи та Го.

Пошук по дереву з використанням методу Монте-Карло (Monte Carlo Tree Search – MCTS) [1, 2] є методом пошуку рішень у заданій області за допомогою випадкових значень із заданого простору значень і побудови дерева пошуку за отриманими результатами. Цей метод вже встиг суттєво вплинути на розв'язання задач штучного інтелекту, а особливо складних ігрових задач. За час існування методу MCTS було створено багато його варіантів та модифікацій з метою прискорення процесу пошуку та покращення рішень, що приймаються на його основі, але потужність концепції цього методу залишає ще дуже великий простір для нових досліджень для покращення пошуку в задачах штучного інтелекту.

Аналіз досліджень.

Методу MCTS присвячено вже багато досліджень і публікацій, на декілька з них автори посилаються в даній роботі. Найбільш фундаментальною узагальнюючою роботою з методу MCTS є

огляд 2012 року, підготовлений десятьма провідними вченими під керівництвом Камерона Броуне, що досліджують цей метод [2], але, незважаючи на її ґрунтовність, вона не є строгою класифікацією. Крім того, після 2012 року з'явились не тільки нові модифікації та покращення існуючих способів, але й принципово нові варіанти реалізації MCTS.

Структура та критерії класифікації різних способів реалізації методу MCTS, його покращення та паралелізації були розглянуті авторами в [1]. Автори вважають, що на основі систематизованої інформації про способи та підходи до покращення пошуку методом MCTS можна визначити нові критерії та місця в схемі роботи MCTS для покращення процесу пошуку.

Метою даної роботи є аналіз процесу пошуку в дереві методом Монте-Карло (MCTS) і особливостей його поведінки при використанні для різних задач та в різних випадках пошуку, а також визначення нового критерію для оцінювання і вибору потенційно кращих варіантів продовження такого пошуку.

Пошук в дереві з використанням методу Монте-Карло (MCTS)

Дерева ігор використовуються комп'ютерними гравцями для вибору ходу гри для таких ігор, як шахи чи Го. Використання повного дерева гри (того, яке містить всі можливі варіанти і послідовності гри) дозволяє реалізувати сильного гравця при застосуванні для пошуку у дереві методів Minimax та Negamax [3]. Однак ці методи потребують достатньо великого об'єму пам'яті та обчислювальної потужності для більшості ігор і тому прямі алгоритми Minimax та Negamax, як правило, не використовуються на практиці на повному дереві. Замість цього використовуються їх варіанти з Alpha-Beta-відсіканням, що дозволяє зменшити необхідні обчислювальні ресурси та зробити дерево більш підходящим для пошуку [3]. Однак для багатьох ігор, дерева яких мають велику степінь розгалуження (наприклад гра Го), використання методів з Alpha-Beta-відсіканням також є недостатнім для того, щоб створити сильного гравця [4].

Кожне піддерево дерева пошуку методом MCTS відповідає поточній позиції гри як коренева вершина, а кожна вершина в цьому піддереві зберігає інформацію про кількість виграшів і загальну кількість разів виконання гри через цю вершину, які позначаються у вершинах через дробову риску (кількість виграшів / кількість ігор) [2]. На рисунку 1 наведено приклад дерева MCTS, де в кожній вершині показано результат багатократного моделювання гри від цієї вершини. Всі термінальні (листові) вершини і нетермінальні вершини, до яких зберігається доцільність додавання нових листів-нащадків, формують межу (передній фронт) дерева MCTS.

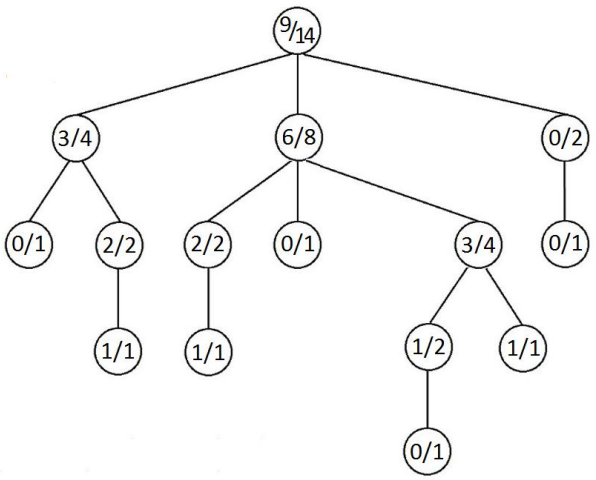

Рис.1. Приклад дерева MCTS

Кожне моделювання гри методом Монте-Карло в стандартному варіанті реалізації є виконанням процесу гри від деякої поточної позиції до кінця гри, вибираючи ігрові ходи випадковим чином [2]. Процес моделювання гри, починаючи від однієї й тієї самої вершини дерева (позиції гри), виконується багато разів, після чого відношення перемог до загальної кількості виконання гри формує сумарну оцінку цієї позиції. Після цього виконується переобчислення балів для кожної вершини дерева пошуку MCTS на шляху від поточної вершини до кореня дерева.

Загальний підхід методу MCTS [2] складається з наступних чотирьох послідовних етапів:

- етап вибору;
- етап розширення;
- етап моделювання;
- етап переобчислення.

Розглянемо більш детально етап вибору. На цьому етапі виконується процес спуску по дереву вниз з метою знаходження найкращого ходу для продовження гри. При цьому кожна наступна вершина, через яку йде спуск, вибирається згідно з певною політикою вибору, яка називається політикою дерева (Tree Policy).

Основна мета цієї політики полягає в тому, щоб зменшити час, витрачений на розгляд недоцільних ходів. Ця мета досягається за допомогою балансування між дослідженням нових, ще не змодельованих, ходів по відношенню до використання вже змодельованих наступних ходів. Використання – це зосередження на кращій в даний час вершині (ході), в той час як дослідження – це розглядання іншої, на даний момент не найкращої вершини (ходу), але яка, можливо, в кінцевому результаті, стане кращою.

Наприклад, однією з найбільш популярних політик дерева, яка використовується, в тому числі, для комп'ютерної гри Го, є політика з використанням методу верхніх довірчих границь – Upper Confidence Bounds (UCB) – або варіант на його основі. Згідно одному з таких варіантів, що називається UCB1 [2], відбирається вершина, для якої рівняння, показане на рисунку 2, дає максимальне значення.

$$\text{рейтинг (вершина)} = \frac{\text{виграші (вершина)}}{\text{кількість ігор (вершина)}} + C * \sqrt{\frac{\log(\text{кількість ігор (предок)})}{\text{кількість ігор (вершина)}}}$$

Рис.2. Рівняння UCB1

Це рівняння складається з двох доданків (умов): перший доданок – рейтинг виграшу вершини, в той час як другий доданок – апроксимація верхньої границі на довірчому інтервалі. Іншими словами, максимізація цього рівняння вибирає вершину, яка, можливо, в кінцевому результаті є кращою. Константа C підбирається таким чином, щоб оптимізувати баланс між процесами дослідження і використання (при більшому значенні C буде виконуватись більше досліджень). Спосіб реалізації MCTS, який для обчислення рейтингу вершин (ходів) використовує метод UCB, називається методом «верхніх довірчих границь, застосованих до дерев» – "Upper Confidence bounds applied to Trees" (UCT).

Коли процес вибору закінчився, і отримано конкретну вершину, виконується крок розширення – процес додавання нової вершини до цієї вершини дерева. Таким чином межа дерева постійно розширюється і стає в подальшому «більш глибокою». Процес розширення збільшує точність і відповідність дерева поточній ситуації гри, роблячи його більш реалістичним представленням можливих наслідків. По суті, розширюючи дерево ми робимо наступний крок у грі.

Після того, як ми додали нову вершину до дерева, тобто, зробили наступний можливий хід, починається крок моделювання – процес виконання моделювання гри від цієї вершини до закінчення гри. При моделюванні ми розглядаємо додану до дерева нову вершину як початковий стан гри, після

чого відбувається гра відповідно до її правил. Процес моделювання можна абстрагувати функцією, яка поверне одне з двох можливих значень: виграш чи програш (в деяких варіантах MCTS повертається також третє значення – нічия). Мета даного етапу – зібрати інформацію про те, скільки зіграних від цієї вершини партій було виграно, тобто, наскільки ефективним був цей крок, обраний на етапах вибору та розширення.

Після того, як моделювання закінчується і результат отриманий, починається крок переобчислення – процес розповсюдження цього результату проходом по дереву від початкової вершини моделювання до кореня. Як правило, новий результат об'єднується з попередніми результатами вважаючи, що всі попередні результати мають однакову вагу із отриманим, але можуть бути використані й інші функції переобчислення [5, 6].

Процес моделювання від вершини з подальшим переобчисленням результату подібний до взяття вибірки із певного ряду даних. Але оскільки при цьому змінюється дерево, то змінюється і ряд, з якого робиться вибірка, і тому цей ряд є нестаціонарним, тобто таким, який містить основну тенденцію розвитку подій.

Загальний підхід методу MCTS можна записати у вигляді алгоритму верхнього рівня наступним чином:

while ЗаданийОбчислювальнийБюджет ще не закінчився ***do***
Вершина ← ВибірВершиниДерева()
Листок ← РозширенняВідВершини(Вершина)
Результат ← МоделюванняВідЛистка(Листок)
ПереобчисленняВідЛистка(Листок,Результат)
end

У якості обчислювального бюджету використовується або певний період часу (час роздумів на один хід гри), або задана кількість ітерацій MCTS. Після того, як відбулася визначена кількість моделювань, чи пройшов визначений для моделювання час, пошук MCTS зупиняється, після чого відбирається найкращий хід відповідно до деяких критеріїв, наприклад, обирається вершина із найбільшою кількістю успішних моделювань. При цьому, піддерево, в якому обрана вершиною є коренем, зберігається як результат у поточному дереві MCTS, в той час як інші вершини відкидаються. Це дозволяє при наступних пошуках, що будуть виконуватись на цьому піддереві, брати його як початкове дерево, і збільшує загальну кількість результатів моделювань в цьому піддереві.

Критерій «глибина-ширина» для контролю форми дерева пошуку MCTS

Ігри та інші прикладення, де застосовується MCTS, можуть мати різну степінь розгалуження можливих дій (ходів) на кожній вершині дерева пошуку. Глибина дерева пошуку узгоджується з визначеннями глибини та ясності гри, які дав Томпсон у [7]. Якщо дерево пошуку для певної гри є глибоким і має низьку степінь розгалуження, то продумування на велику кількість ходів наперед стає нескладною задачею, оскільки дерево пошуку буде «вузьким». В такому випадку ясність гри буде високою і навіть недосвідчений гравець-людина зможе зрозуміти цінність подальших ходів і який з них є найкращим, в результаті чого вибір найкращого ходу з невеликої кількості варіантів може стати занадто простим рішенням. З іншої сторони, використання рівняння UCB1 для вибору ходів в грі за комп'ютер при малій степені розгалуження в дереві пошуку також буде вибирати найкращі ходи, оскільки вони будуть добре передбачувані, і гра стане повторюваною і нецікавою. З іншої сторони, якщо дерево пошуку MCTS для певної гри має високу степінь розгалуження, гравцю-людині буде дуже важно охопити і продумати багато варіантів продовження гри [8].

Накопичений досвід використання методу MCTS показав, що, в залежності від степені розгалуження можливих варіантів рішень на вершинах дерева, для одних прикладень (ігор) ефективність пошуку кращих ходів вище, якщо дерево зростає переважно в ширину, а для інших – в

глибину [2].

На процес більшого зростання дерева пошуку вглибину чи вширину впливають два фактори:

• значення константи С у рівнянні, показаному на рис.2, яка регулює баланс між кількістю досліджень (exploration) та кількістю використань (exploitation);

• відношення кількості виконання етапу вибору (етап 1) порівняно до кількості моделювань, виконаних на етапі 3.

Згідно першого фактору, було встановлено, чим більша константа С у рівнянні, показаному на рис.2, то тим більше буде виконуватись досліджень і тим більше дерево буде зростати вширину [2]. Згідно другого фактору, було встановлено, що чим більше виконується моделювань порівняно до кількості процесів вибору, то дерево більше зростає вглибину. Константа С фактично є параметром, який контролює форму дерева пошуку MCTS. При менших значеннях С створюється глибше і вужче дерево пошуку, в той час, як при більших значеннях С буде побудоване менш глибоке але більш широке дерево MCTS [9]. Рівняння UCB1 автоматично фокусує пошук на поточних найкращих варіантах подальших ходів одночасно з наданням можливості достатнього дослідження для того, щоб знайти потенційно найкращий хід, якого в дереві ще нема.

В роботі [8] для контролю за формою дерева пошуку, крім константи С, використовуються критерії глибини дерева та степені розгалуження кожної вершини (критерій ширини дерева).

Автори даної статті вважають, що критерій ширини дерева у вигляді степені розгалуження кожної вершини є недостатньо об'єктивним і змістовним для контролю за формою дерева пошуку, оскільки такий критерій показує тільки локальну ширину розгалуження для поточної вершини і потенційну (але не фактичну) ширину всього дерева, і пропонують замість степені розгалуження вершини використовувати відносний критерій «глибина-ширина», що обчислюється за певними правилами для кожної вершини дерева. Оскільки правила обчислення цього критерію є дуже простими, то його використання не вплине на час роботи пошуку.

Обчислення критерію «глибина-ширина» будемо виконувати за наступними правилами:

• у всіх термінальних вершин дерева (листків) і глибина, і ширина їх піддерев дорівнюють одиниці;

• глибина піддерева нетермінальної вершини обчислюється як максимальне значення зі значень глибин піддерев її нащадків плюс одиниця;

• ширина піддерева нетермінальної вершини обчислюється як сума значень ширин піддерев її нащадків.

На рисунку 3 показаний приклад розгалуженого дерева з обчисленими значеннями критерію «глибина\ширина».

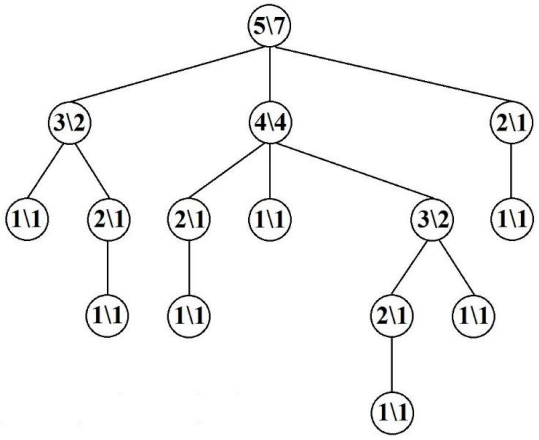

Рис.3. Розгалужене дерево з обчисленими значеннями критерію «глибина\ширина»

Перша цифра відповідає глибині піддерева, а друга цифра – ширині піддерева. Щоб відрізняти ситуацію, коли у вершині показаний критерій «глибина-ширина», а не критерій «кількість виграшів/кількість моделювань», на відміну від останнього, для розділу двох значень використовується символ «\» (back-slash), а не «/».

Якщо для певних задач відомо, яка форма дерева MCTS (більш глибока чи більш широка) є ефективнішою, то враховуючи таку наявну початкову інформацію, для цих задач з'являється можливість доцільного виконання динамічного контролю форми дерева за критерієм «глибина-ширина», направляючи подальшу побудову дерева пошуку у потрібному напрямку.

Розглянемо, які переваги при виконанні пошуку за методом MCTS можна отримати з допомогою запропонованого критерію «глибина-ширина». Цей критерій надає можливість без уповільнення роботи основної частини будь-якої реалізації методу MCTS, одночасно з роботою цієї основної частини, визначити нові перспективні вершини для продовження гри і виконати додаткове дослідження найбільш перспективних гілок дерева MCTS на вільних в цей час апаратних ресурсах.

Крім того, з'являється також можливість більш доцільно виконувати розпаралелення алгоритму на доступних в даний момент апаратних ресурсах. Якщо для виконання паралельної обробки доступна комп'ютерна система з невеликою кількістю комп'ютерів/процесорів/процесорних ядер, то можливості динамічної корекції критерію «глибина-ширина» дерева є обмеженими. З іншої сторони, зі зростанням доступної кількості та потужності вузлів комп'ютерної системи, з однієї сторони зростають можливості розпаралелення процесу пошуку, але, з іншої сторони, зростає також і складність організації ефективної паралелізації. Таким чином можна забезпечити коректне визначення потрібних апаратних ресурсів для ефективного розпаралелення на нижчих рівнях дерева гри, виходячи з інформації, що знаходиться у вузлах-предках без подальшого перерозподілу апаратних ресурсів (ядер, процесорів, потоків тощо) з можливим додатковим перерозподілом (перезаписуванням) інформації в різних типах та блоках пам'яті.

Висновки

Пошук у дереві з використанням методу Монте-Карло MCTS може показувати гарні результати у багатьох областях застосування, але не у всіх. Для одних задач MCTS виконує пошук краще, якщо форма дерева зростає переважно в ширину, а для інших – якщо в глибину. Паралелізація може бути виконана згідно декількох різних підходів. На основі аналізу методу MCTS та існуючих підходів контролю за процесом пошуку і формою дерева пошуку був запропонований оригінальний критерій «глибина-ширина» для такого контролю, який обчислюється за певними правилами. Відзначається, що підтримка такого критерію в процесі пошуку не призведе до суттєвих часових витрат. Подальшим напрямком досліджень можуть бути як деталізація застосування запропонованого критерію під час пошуку методом MCTS, так і розробка методів паралелізації на основі цього критерію.

1. Марченко О. І. Структура та критерії класифікації способів реалізації та покращення пошуку по дереву методом Монте-Карло / О. І. Марченко, О. О. Марченко, М. М. Орлова. – Комп'ютерно-інтегровані технології: освіта, наука, виробництво, 2015. – № 21. – С. 51–57.
2. Cameron Browne and others. A Survey of Monte Carlo Tree Search Methods // IEEE Trans. on Computational Intelligence and AI in Games. - Vol. 4. - No. 1. - March 2012, pp.1-48.
3. CS312 Recitation 21. Minimax search and Alpha-Beta Pruning [Електронний ресурс]. Режим доступу: http://www.cs.cornell.edu/courses/cs312/2002sp/lectures/rec21.htm
4. B. Bouzy and T. Cazenave, "Computer Go: an AI oriented survey" // Artificial Intelligence. Vol.132. – Issue 1, October 2001, pp.39-103.
5. Thomas Keller, Malte Helmert. Trial-based Heuristic Tree Search for Finite Horizon MDPs // [Електр. Ресурс]. Режим доступу: http://www.aaai.org/ocs/index.php/ICAPS/ICAPS13/paper/view/6026.
6. Zohar Feldman, Carmel Domshlak. On MABs and Separation of Concerns in Monte-Carlo Planning for MDPs // Proc. 24th Int. Conf. Automat. Plan. Sched., pp.120-127.
7. J. M. Thompson. Defining the abstract. The Games Journal, 2000. [Електронний ресурс]. Режим доступу: http://www.thegamesjournal.com/articles/DefiningtheAbstract.shtml
8. Gareth M. J. Williams. Determining Game Quality Through UCT Tree Shape Analysis. MSc. Thesis, Imperial College London, 2010.
9. Joel Veness, Kee Siong Ng, Marcus Hutter, William Uther, David Silver. A Monte-Carlo AIXI Approximation // Journal of Artificial Intelligence Research 40 (2011), pp.95-142.