Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра математики факультету інформатики

# 

**ОПТИМАЛЬНІ СТРАТЕГІЇ В БАГАТОКРОКОВІЙ ГРІ ЗІ СКІНЧЕННИМ ГОРИЗОНТОМ НА ПРИКЛАДІ ГРИ РЕВЕРСІ**

**Текстова частина до курсової роботи**

**за спеціальністю** «**Прикладна математика**»

#### Керівник курсової роботи

к.ф.-м.н. Чорней Р.К.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(підпис)*

“\_\_\_\_” \_\_\_\_\_\_\_\_\_\_ 2014 р.

Виконав студент Фітель Д.Р.

“\_\_\_\_” \_\_\_\_\_\_\_\_\_\_ 2014 р.

Київ 2014

**Зміст**

**Вступ**

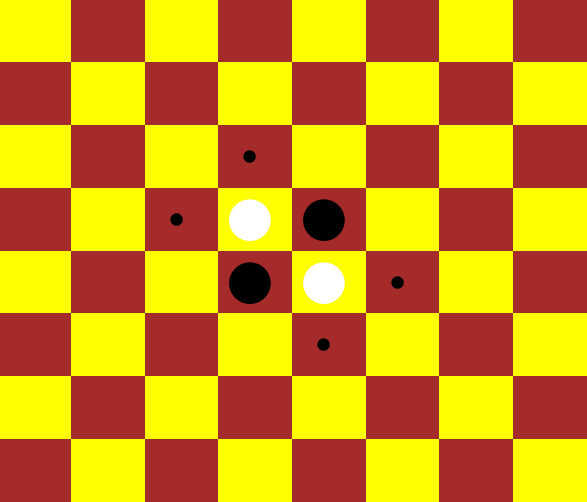
**Опис гри**

«Реверсі», або «Отелло», - настільна гра для двох гравців, в якій використовується дошка розміру 8x8 з нерозміченими клітинками та 64 ідентичних ігрових диски, білі з одного боку та чорні з іншого. Гравці по черзі розміщують по одному диску свого кольору так, щоб лінія дисків суперника опинилася між дисками гравця, після чого оточені диски перевертаються. Мета кожного гравця – завершити гру з більшою кількістю дисків, ніж у суперника.

Сучасна версія гри була винайдена 1883 року в Великобританії та до кінця 19 століття здобула популярність в Європі. Найімовірніше, ідея гри виникла під впливом давньої китайської гри «Го». Остання версія правил сформувалася в Японії в 70-х. Японська ігрова компанія Tsukuda Original зареєструвала гру під назвою «Отелло». Назва – пряма відсилка до п’єси «Отелло», де змагання фігур символізує драму між Отелло (чорні) та Дездемоною (білі), а традиційний зелений колір поля навіяний образом генерала Отелло на полі битви.

З метою запобігання неоднозначностей визначимо правила гри.

1. У грі беруть участь два гравці. Вважатимемо фігури першого гравця чорними (B), другого – білими (W).
2. Використовується стандартне ігрове поле розміру 8x8.
3. Перед початком гри чотири центральні клітинки зайняті фігурами гравців наступним чином:



1. Першими роблять хід чорні (можливі варіанти першого ходу на малюнку відмічені чорними точками).
2. Гравці по черзі ставлять на дошку фігури свого кольору так, щоб ряд фігур суперника опинився між його фігурами по горизонталі, вертикалі чи діагоналі, після чого оточені фігури суперника міняють колір.
3. Гравці на кожному кроці можуть обирати будь-який з доступних ходів. Якщо у гравця не залишилося можливих ходів, він пропускає поточний хід.
4. Гра завершується, коли у обох гравців не залишилося ходів. Найчастіше така ситуація виникає, коли всі клітинки поля заповнені, проте можливі ситуації, коли у обох гравців закінчилися можливі ходи до того, як вони зайняли всі клітинки.
5. Перемагає гравець, кількість фігур у якого в кінці матчу переважає кількість фігур суперника. Якщо кількість очок однакова у обох гравців, результатом вважається нічия.

Популярності набули декілька модифікацій гри.

1. Реверсі NxN. Окрім іншого розміру дошки, як правило, перед початком гри диски одного кольору розміщують не в шаховому порядку, а поряд.

При цьому дошки меншого, ніж 8x8, розміру не є цікавими, адже при цьому гра суттєво спрощується, і на сучасних комп’ютерах стає повністю детермінованою.

1. Антиреверсі. Перемагає той, у кого в кінці гри менше дисків.
2. Реверсі з чорною дірою. Одна з клітинок, випадковим чином обрана на початку гри, стає недоступною.

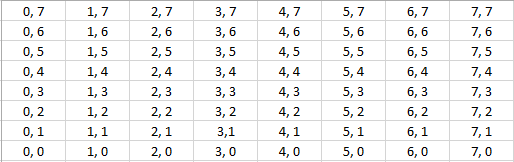
Дані модифікації цікаві для дослідження, адже їх дуже просто отримати з оригінальної програмної реалізації, проте вони суттєво змінюють ігровий процес. В даному випадку розглядатимемо стандартний набір правил.

**Основні поняття**

Перед тим, як перейти до розробки оптимальної стратегії, необхідно формалізувати об’єкти ігрового поля та правила гри.

***Означення 1***. Позначимо ci, j = [i, j] клітинку ігрового поля з координатами i та j в Декартовій системі координат на площині, вважаючи, що поле знаходиться в першій чверті.

Розмітка поля виглядає наступним чином:



***Означення 2***. F – множина всіх клітинок поля, .

|F| = 8\*8 = 64.

***Означення 3***. Bn – множина клітинок, зайнятих чорними фігурами на n-му кроці, Wn – білими, En – множина порожніх клітинок.

Множини Bn, Wn і En – розбиття множини всіх клітинок поля F, тобто має місце співвідношення

.

Твердження дуже легко доводиться: в будь-який момент часу кожна клітинка може бути або порожньою, або заповненою диском чорного кольору, або білого.

Тоді, згідно з правилами, на початку гри , , .

***Теорема 1***. Після кожного ходу одного з гравців (крім пасу):

1. Кількість порожніх клітинок зменшується на 1
2. Кількість фігур поточного гравця збільшується щонайменше на 2

, де B(W) – поточний гравець.

1. Кількість фігур суперника поточного гравця зменшується принаймні на 1

, де B(W) – суперник поточного гравця.

***Доведення***. Доведення кожного пункту тривіальне.

1. Згідно з правилами гри, на кожному кроці гравець розміщує на полі рівно 1 диск, при цьому жодна з попередніх фігур не змінює положення. Після кожного ходу

Вже відома рівність

Тоді має місце

Підставляючи першу рівність в третю, отримуємо

1. Здійснюючи хід, гравець розміщує на полі один диск свого кольору, причому як мінімум один диск суперника повинен змінити колір. При цьому жодна з його фігур до ходу не змінює стану. Отже, гравець збільшує свій рахунок щонайменше на 2 очка.
2. Згідно з правилами, гравець може здійснити хід лише тоді, коли одна або більше фігур суперника в ряд опиняються між його фігурами, після чого вони міняють колір. Оскільки жодної нової фігури кольору суперника за хід гравця не з’являється, то суперник втрачає як мінімум одну фігуру.

***Наслідок з теореми 1***. Ігрова партія складається щонайбільше з |E0| = 60 ходів, не враховуючи пропусків.

***Доведення***. На початку партії кількість порожніх клітинок |E0| = 60. За пунктом 1 теореми 1, на кожному кроці заповнюється рівно 1 порожня клітинка. Отже, після 60-го ходу все поле заповнене, що, згідно з правилами, означає кінець гри.

***Означення 4***. Зайнята клітинка в момент часу n називається стабільною, якщо вона не змінить колір за жодних наступних ходів гравців

Нестабільні клітинки дають мало інформації про поточний стан гри, адже кожна з них може змінити колір в майбутньому. Стабільні клітинки – важлива складова функції оцінки поточної позиції.

***Означення 5.*** Sn – множина стабільних клітинок на n-му кроці. Для зручності розглянемо розбиття Sn на дві підмножини SBn і SWn – множину стабільних позицій гравця чорними і білими відповідно, .

Очевидно, на початку гри , в кінці гри - , де f – останній крок.

***Теорема 2***. Заповнена клітинка є стабільною , якщо у кожній з чотирьох пар протилежних за напрямом сусідніх клітинок хоча б одна з клітинок або виходить за межі поля, або є стабільною клітинкою того ж кольору.

***Доведення***. Оскільки умови однакові для всіх ліній, достатньо показати, що твердження справедливе для будь-якої лінії окремо.

За правилами здійснення ходу, фігура міняє колір, якщо вона (можливо, з декількома іншими фігурами того ж кольору під ряд) під час ходу оточується фігурами протилежного кольору по одній з 4-х ліній. Отже, кожна з двох сусідніх фігур на цій лінії або іншого кольору, або того самого кольору, але після ходу разом з усіма оточеними фігурами міняє колір. Отже, якщо хоча б одна з сусідніх фігур ніколи не задовольнятиме цих умов, дана позиція є стабільною.

Розглянемо окремо випадки з умови теореми.

1. Сусідня клітинка виходить за межі дошки, тобто, сама фігура лежить на краю. Оскільки хоча б з одного боку не можна розмістити фігури кольору суперника, по даній лінії неможливо змінити колір фігури.
2. Сусідня фігура того ж кольору і стабільна. Завдяки стабільності сусідня фігура ніколи не змінить колір на протилежний, тобто перша умова не задовольняється. Також, оскільки сусідня фігура не може змінити кольору, ситуація, коли вона разом з сусідньою оточується фігурами суперника неможлива, бо тоді сусідня повинна змінити колір, що суперечить її стабільності.

Отже, виконання цих умов для усіх чотирьох ліній достатньо, щоб фігура була стабільною.

Кількість стабільних фігур, на відміну від звичайного рахунку, важлива при оцінці позиції, бо вона дає інформацію про гарантоване значення фінального рахунку:

***Теорема 3***. Алгоритм знаходження кількості стабільних фігур SBn і SWn.

1. Перевірити кожну фігуру за теоремою 2 і відмітити стабільні фігури.
2. Якщо на попередньому кроці жодної нової стабільної фігури не було знайдено, закінчити роботу, інакше повторити крок 1.

***Доведення***. Оскільки умови теореми 2 опираються на вже відомі стабільні фігури, при першій ітерації знаходяться стабільні фігури, що не залежать від стабільності сусідів (фігури на кутах), і на кожній наступній стабільні знаходяться серед сусідів стабільних фігур, знайдених на попередній ітерації. Якщо на якійсь ітерації жодна нова стабільна фігура не була знайдена, всі наступні ітерації почнуться з тієї ж позиції і, отже, не змінять результату.

Через часову складність алгоритм доцільно використовувати на практиці лише у випадку зберігання постійного кешу стабільних фігур і повторного запуску після кожного ходу.

***Теорема 4***. Окремі випадки стабільних позицій.

1. Заповнені кутові клітинки стабільні
2. Послідовність заповнених клітинок одного кольору вздовж краю поля, яка починається з кутової клітинки, є стабільною
3. Послідовність з 8-ми заповнених клітинок на одному краю поля стабільні незалежно від їхнього кольору.

***Доведення***.

1. Розглянемо заповнену клітинку [0, 0]. Сусідні клітинки [-1, 0] по горизонталі, [0, -1] по вертикалі і [-1, -1], [-1, 1] та [1, -1] по двох діагональних лініях виходять за межі поля, отже, за теоремою 2, заповнена клітинка [0, 0] стабільна.

Аналогічно доводиться стабільність решти кутових клітинок.

1. Не втрачаючи загальності, розглянемо випадок заповненої кутової клітинки [0, 0] і напрямок решти клітинок . За попереднім пунктом, клітинка [0, 0] стабільна. Нехай клітинка заповнена клітинка того ж кольору в напрямку d [n, 0] стабільна. Розглянемо сусідів клітинки [n + 1, 0]. [n, 0] по горизонталі стабільна , [n + 1, -1] по вертикалі, [n, -1], [n + 2, -1] по діагоналі виходять за межі поля. За теоремою 2, клітинка [n + 1, 0] стабільна. Отже, за індукцією, всі клітинки одного кольору в напрямку [1, 0] від початкової кутової клітинки [0, 0] стабільні.

Аналогічно доводиться стабільність послідовностей у напрямках по горизонталі і по вертикалі з початком у інших кутових позицій.

1. Розглянемо випадок нижнього горизонтального ряду (решта доводяться аналогічно). Нехай всі клітинки [n, 0], заповнені довільним чином. По горизонталі здійснити хід неможливо, адже для цього необхідне хоча б одне незаповнене поле. Також, для всіх клітинок [n, 0] сусідні клітинки по вертикалі і діагоналі [n – 1, -1], [n, -1], [n + 1, -1] виходять за межі поля, і, враховуючи четверту лінію – горизонтальну – за теоремою 2 кожна клітинка є стабільною.

Аналогічно твердження доводиться для решти крайніх рядів з клітинками виду [0, n], [7, n], [n, 7].

На практиці перевірка всіх клітинок на стабільність доволі ресурсоємна, і клітинки, що не лежать на краю поля, стають стабільними під кінець гри, коли глибина пошуку може досягати фінальних станів і, як наслідок, перевірка на стабільність втрачає сенс.

Доречно перевіряти на стабільність лише клітинки на краю поля, зекономлені обчислювальні ресурси виділивши на збільшення глибини пошуку.

**Пошук оптимальної стратегії**

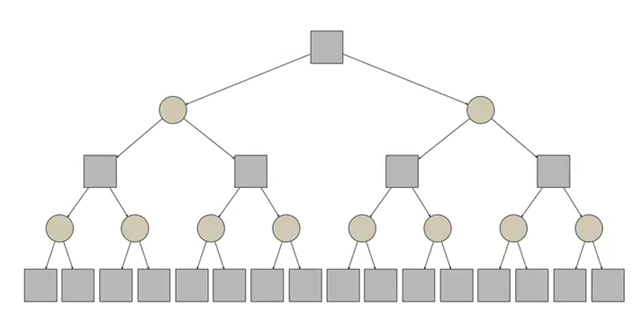
Реверсі, як і більшість популярних настільних ігор, добре піддається аналізу і програмуванню відповідного штучного інтелекту. Цьому сприяє ряд факторів:

* Реверсі – гра для двох гравців. Збільшення кількості учасників значно ускладнює аналіз гри.
* Реверсі – гра з нульовою сумою. Кожне очко одного гравця – це втрачене очко іншого. В більшості ігрових ситуацій в кінці гри всі клітинки заповнені, і сума очок гравців рівна 64.
* Реверсі – гра з повною інформацією. Обом гравцям відомі позиції всіх фігур в будь-який момент часу.
* Реверсі – повністю детерміністична гра. За наслідком з теореми 1, кількість кроків завжди обмежена, тому і ігрових комбінацій скінченна кількість. В будь-який момент часу можливо розглянути всі комбінації ходів гравців до завершення гри і, користуючись результатами, обрати оптимальний за певним критерієм хід.

Така класифікація дозволяє розглянути дерево всіх можливих станів і обирати наступний хід за певним критерієм. Наступний крок – обрати критерій оптимальності за описати алгоритм пошуку рішення.

**Ігрове дерево**

Розглянемо початок гри (момент часу t = 0). Згідно з правилами, перший хід робить гравець чорними (B). Зобразимо поточну позицію як корінь орієнтованого дерева. Оскільки черга чорних робити хід, нехай він буде чорного кольору. У гравця чорними в час t = 0 є 4 варіанти ходів, зобразимо можливі стани дошки після кожного з них як вершини дерева, в які з кореня виходять орієнтовані ребра. Отримаємо 4 нові вершини на першому рівні дерева. Оскільки черга гравця білими (W) робити хід, нехай вони будуть білого кольору. Аналогічно продовжимо розширювати дерево всіма можливими ходами з кожної вершини попереднього рівня до моменту, коли всі шляхи з кореня вестимуть у позицію, де у гравців не залишиться можливих варіантів зробити хід, тобто можливих закінчень гри. Зауважимо, що пропуск в даному випадку теж вважається ходом і відображається відповідним ребром у дереві.



Кожна вершина цього дерева відображає можливу позицію фігур в певний момент часу, а шляхи з кореня до кінцевих вершин – ігрові партії.

У обох гравців є можливість обходити всі шляхи з поточної вершини і за скінченний час знайти всі досяжні кінцеві вершини. Оскільки мета гравця – максимізувати власний рахунок в кінці партії, задача зводиться до вибору найкращого наступного ходу з інформацією про можливі закінчення, до яких призведе кожен з них.

***Означення 6***. – орієнтоване дерево з наступними властивостями:

1. Вершини – можливі ігрові позиції

Колір кожної вершини збігається з кольором гравця, чия черга здійснювати хід.

1. Корінь дерева – позиція на початку гри

Оскільки чорні ходять першими, корінь чорного кольору.

1. З кожної вершини виходять ребра до ігрових позицій, які можливі після того, як поточний гравець здійснить хід.

, так, що існує хід гравця кольору першої вершини, після якого позиція (Bn, Wn) переходить в (Bn + 1, Wn + 1).

Алгоритм побудови дерева станів T описаний вище. Його можна застосувати для довільної позиції в ненульовий момент часу, таким чином отримавши піддерево загального дерева, в яке входитимуть лише ті вершини, що відповідають досяжним позиціям.

***Означення 7***. Рівнем вершини v дерева T називається довжина шляху до неї від кореня.

***Теорема 5***. Дерево T дводольне, всі вершини на парних рівнях – чорного кольору, на непарних – білого.

***Доведення***. За означенням, корінь дерева (n = 0) чорного кольору, і кожна суміжна вершина, з’єднана з іншою ребром, протилежного кольору. Отже, всі вершини першого рівня (n = 1) білі. За індукцією, для кожного n маємо: якщо вершина 2n чорна, то 2n + 1 – біла, якщо 2n + 1 – біла, то 2n + 2 – чорна.

Розбиття множини вершин дерева за парністю рівнів відповідає також розбиттю на дві підмножини, так, що жодні дві вершини з однієї підмножини не є суміжними. Отже, граф T є дводольним: жодні дві вершини з парних рівнів не є суміжними і жодні дві вершини з непарних рівнів не є суміжними.

***Теорема 6***. Максимальний рівень вершини у дереві T менший, ніж 120.

***Доведення***. Оскільки рівень вершини – це кількість ребер у шляху від кореня до вершини, а кожне ребро відповідає за здійснений хід, то рівень рівний кількості здійснених ходів. За наслідком з теореми 1, ходів в партії може бути щонайбільше 60, не враховуючи пропусків. При цьому на шляху від кореня до вершини не може бути два пропуски під ряд, бо це означало б кінець гри. Оскільки перше і останнє ребро не можуть відповідати пропускам, пропусків може бути щонайбільше 59 (кожен другий хід – пропуск). Отже, в загальному кількість ребер на шляху до вершини не перевищує 120.

Така оцінка доволі груба, проте вона дає можливість встановити строге обмеження на глибину дерева.

Далі необхідно визначити дійснозначну функцію оцінки ігрових позицій, яка використовуватиметься для порівняння альтернатив і прийняття рішення про оптимальність ходу. В загальному випадку, для кожного гравця потрібна окрема функція, але, оскільки мета кожного гравця – збільшити свій відрив від суперника, і оскільки реверсі – гра з нульовою сумою, то можна, не втрачаючи загальності, визначити функцію корисності лише для гравця чорними (B), тоді відповідною функцією корисності для гравця білими (W) буде мінус функція корисності для чорних.

***Означення 8***. Функція - функція оцінки позиції з властивістю , де vn – кінцева позиція.

Про значення функції на станах, відмінних від кінцевих, поки що нічого не відомо, але припускаємо, що функція визначена для будь-якої позиції. Таким чином визначена функція дозволяє порівнювати кінцеві стани за відривом чорних від білих. Мета гравця чорними (B) – максимізувати її значення, гравця білими – мінімізувати.

Ігрову ситуацію завжди можна представити у вигляді піддерева загального ігрового дерева, розглядаючи лише шляхи, що виходять від вершини, яка відповідає поточній позиції. За умови повністю визначеної функції корисності задача вибору найкращого рішення зводиться до вибору ходу, який максимізує значення функції корисності на майбутніх позиціях за певним критерієм. Наступний крок – вибір і обґрунтування критерію.

**Мінімаксний пошук на дереві станів**

Оскільки процедура побудови дерева вже відома, наступний крок – вибір критерію оптимальності наступного ходу.

В загальному випадку вибір критерію залежить від поведінки суперника. Доцільно припустити, що суперник керується здоровим глуздом, і що його мета – перемогти. Але суперник може переслідувати інші цілі, він може помилятися, врешті решт, він може піддаватися або ходити довільним чином. За реальних обставин про суперника невідомо нічого. Щоб мати гарантію результату, припустимо, що суперник завжди ходить найгіршим для нас чином. Такий підхід називається песимістичним, а критерій, який відповідає йому при прийнятті рішення – мінімаксний критерій.

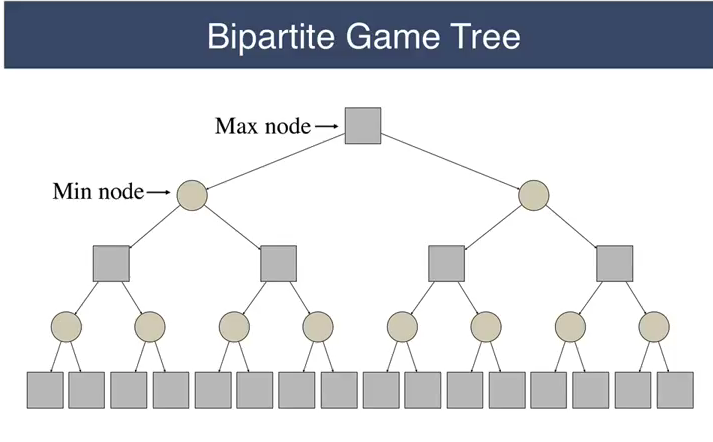
Згідно з попередньою домовленістю, функція корисності станів спільна для гравців, проте гравець чорними (B) їх максимізує, гравець білими (W) – мінімізує. Надалі під позначенням max гравця матимемо на увазі гравця чорними (B), min – білими (W).

Припустимо, що функція корисності визначена на вершинах, що відповідають станам гри після наступного ходу. Тоді в найпростішому випадку оптимальний хід для гравців:

, V’ – множина вершин наступного рівня, суміжних з поточною вершиною v – оптимальний для max гравця,

, V’ – множина вершин наступного рівня, суміжних з поточною вершиною v – оптимальний для min гравця.

За припущенням, функція корисності визначена на кінцевих вершинах. Опишемо алгоритм знаходження корисності всіх вершин за мінімаксним принципом. Розглянемо спрощену схему дерева. Не втрачаючи загальності, розглянемо лише випадок, коли початкова вершина відповідає черзі гравця чорними (B) здійснювати хід.



За теоремою 5, дерево дводольне. Після кожного гіпотетичного ходу гравця хід робить його суперник. Позначимо вершини дерева, в яких хід робить гравець, який максимізує цільову функцію (тобто гравець чорними), через max, решту вершин, коли функція мінімізується – min.

***Означення 9***. Функція корисності V для довільної вершини згідно з мінімаксним критерієм має вигляд

* для max вершин
* для min вершин

В результаті таке рекурентне визначення дозволяє знайти корисність кожної вершини, починаючи з кінцевих. На практиці воно реалізується рекурсивним обходом дерева в глибину.

***Теорема 7***. Значення max вершини – мінімальний гарантований результат max гравця за умови, що він користується принципом мінімаксу, значення min вершини – максимальний гарантований результат для min гравця за умови, що він користується принципом мінімаксу.

***Доведення***. Проведемо доведення за методом математичної індукції рівню вершини, починаючи з кінцевого. Почнемо з випадку, коли вершина є кінцевою. Тоді результат єдиний і рівний корисності вершини. Розглянемо ще випадок, коли max вершина є передостанньою, а кінцевими є min вершини. В такому разі значення вершини рівне найбільшому із значень min вершин, тобто найменшому гарантованому результату. Аналогічно доводиться випадок min вершини.

Припустимо, що для n останніх рівнів значення max вершин – мінімальне гарантоване для max гравця, min – максимальне гарантоване для min гравця. Розглянемо два наступні рівні. Нехай n + 1 – ий рівень – max вершини. Тоді значення кожної max вершини рівне найбільшому з найгірших для нього гарантованих результатів, тобто мінімальному гарантованому результату. Якщо ж n + 1 – ий рівень – min вершини, то таким самим спопобом можна показати, що max вершини n + 2 – го рівня задовольняють умови теореми. Аналогічно доводиться крок індукції для min вершин. Отже, теорема справедлива для всіх max і min вершин.

Таким чином, запропонований мінімаксний пошук на дереві станів дозволяє знайти гарантовану корисність всіх можливих ходів і, отже, обрати хід з найкращим для гравця гарантованим результатом.

Псевдокод для рекурсивної мінімаксної функції:

**function** minimax(node, depth, maximizingPlayer)

**if** depth = 0 **or** node is a terminal node

**return** the heuristic value of node

**if** maximizingPlayer

bestValue := -∞

**for each** child of node

val := minimax(child, depth - 1, FALSE)

bestValue := max(bestValue, val);

**return** bestValue

**else**

bestValue := +∞

**for each** child of node

val := minimax(child, depth - 1, TRUE)

bestValue := min(bestValue, val);

**return** bestValue

Для max гравця виклик функції виглядає наступним чином

minimax(origin, depth, TRUE)

**α-β відсікання**

Мінімаксний пошук на ігровому дереві дає гарантований результат, проте можна значно скоротити кількість відвідуваних вершин, при цьому результат роботи алгоритму не зміниться.

Важливо зауважити, що порядок відвідування вершин дерева відповідає рекурсивному обходу в глибину. Тобто, для набору вершин на деякому рівні алгоритм спочатку знайде вартість першої вершини, рекурсивно опустившись до кінцевих вершин, потім другої і так далі. Рекурсивний пошук починається з синів поточної вершини. При відвідуванні синів однієї з вершин може виявитися, що її значення буде в будь-якому випадку гіршим для гравця, ніж значення попередньо відвіданої вершини. В такому разі відвідування цілого піддерева, що починається з неї, можна припинити, тим самим зекономивши обчислювальні ресурси.

Нехай α значення вершини – деяке дійсне значення, що не перевищує корисності даної вершини. Початкове значення α рівне значенню вершини, якщо вона кінцева, інакше воно рівне , далі для max вершин воно рівне найбільшому значенню корисності відвіданих синів, а для min вершини – α значенню вершини–попередника.

Відповідно, значення β не менше, ніж корисність вершини. Початкове значення β рівне значенню вершини, якщо вона кінцева, інакше воно рівне , далі для min вершин воно рівне найменшому значенню корисності відвіданих синів, а для max вершини – β значенню вершини–попередника.

Гарантується, що

1. Значення α і β можуть змінюватися, але α ніколи не зменшиться, а β ніколи не збільшиться

Псевдокод для α-β відсікання

**function** alphabeta(node, depth, α, β, maximizingPlayer)

**if** depth = 0 **or** node is a terminal node

**return** the heuristic value of node

**if** maximizingPlayer

**for each** child of node

α := max(α, alphabeta(child, depth - 1, α, β, FALSE))

**if** β ≤ α

**break** *(\* β cut-off \*)*

**return** α

**else**

**for each** child of node

β := min(β, alphabeta(child, depth - 1, α, β, TRUE))

**if** β ≤ α

**break** *(\* α cut-off \*)*

**return** β

В середньому кількість відвіданих вершин при α-β пошуку складає , тобто за той самий час можна проглянути дерево з таким самим фактором розгалуження вдвічі глибше.

Оскільки відсікання на певному рівні можливе лише після відвідування першої вершини і можливе не для всіх вершин, виграш в кількості відвіданих вершин залежить від порядку вершин, в якому вони проглядаються.

**Функція оцінки позиції**

Описаний вище мінімаксний метод пошуку працює на довільному дереві за умови, що функція корисності стану визначена для всіх кінцевих вершин. За теоремою 6, глибина дерева може становити від 60 до 120 рівнів, що в поєднанні зі значним фактором розгалуження робить задачу повного обходу ігрового дерева неможливим за короткий проміжок часу. На практиці використовується обмежений мінімакс, з пошуком на максимум певну глибину. Виникає необхідність довизначити функцію корисності ігрового стану на довільну вершину.

Для вершин, що відповідають ігровим закінченням, функція корисності не повинна змінюватися, бо вона напряму відображає корисність цих позицій для гравців. Щодо решти нетермінальних станів, то для них значення функції корисності повинні враховувати позицію фігур гравців та потенційний фінал.

Насамперед, з досвіду гри в реверсі можна зробити декілька висновків.

1. Кількість очок на початку і в середині партії не можна використовувати для порівняння позицій. Більше того, в першій половині гри сильні суперники мінімізують кількість своїх фігур. Це прямий наслідок того, що більшість фігур можуть змінити свій колір в будь-який момент.
2. Позиції на чотирьох кутах поля надзвичайно важливі, адже вони ніколи не міняють кольору (завжди є стабільними).
3. Позиції на краях важливі, бо вони стають стабільними раніше, ніж фігури в центрі поля і дозволяють захоплювати більше фігур за останні ходи.
4. Потрібно уникати ситуацій, в яких фігури оточують фігури суперника, адже тоді не залишається можливих ходів.
5. Як наслідок з попереднього спостереження, потрібно віддавати перевагу ситуаціям, коли фігури суперника оточують фігури гравця.

Перелічені спостереження ведуть до кількох стратегічних принципів, які в комбінації здатні ефективно оцінити ігрову позицію.

1. ***Стабільність***. Значна вага присвоюється стабільним фігурам. Єдина гарантована характеристика позиції при обмеженому пошуку. Кількість стабільних позицій обох гравців гарантовано не менша, ніж фінальний рахунок. В пізніх етапах партії кількість стабільних позицій грає вирішальну роль, оскільки при щільному заповненні поля кожен хід значно впливає на рахунок.
2. ***Кутові позиції***. Додаткова вага присвоюється зайнятим кутовим клітинкам. Як правило, перші стабільні позиції – саме зайняті кути. Більше того, зайняті кути в першій половині гри дозволяють швидко збільшити кількість стабільних позицій за рахунок сусідніх клітинок. При цьому клітинки, сусідні до кутових, мають негативну вагу, адже зайняті одним гравцем, вони в більшості випадків дозволяють іншому зайняти відповідний кут.
3. ***Мобільність***. Базова вага присвоюється за кількість доступних ходів обох гравців незалежно від того, чия черга здійснювати хід. Призводить до агресивної стратегії затискування гравця. В поєднанні з урахуванням стабільних клітинок дозволяє обмежити доступні ходи суперника таким чином, щоб зайняти якнайбільше стабільних позицій. Відіграє ключову роль на самому початку партії, коли пошук не доходить до станів зі стабільними і зайнятими кутовими позиціями.
4. ***Кількість фігур (рахунок)***. Єдиний критерій для термінальних станів з вагою, що на порядок перевищує евристику всіх попередніх методів. Для решти станів може бути проігнорований, оскільки кількість фігур в середині партії не дає жодної корисної інформації, адже навіть за останній крок значна перевага може перейти до рук іншого гравця.

**Висновок**

**Список використаної літератури**

1. Stuart J. Russel, Peter Norvig. Artificial Intelligence. A Modern Approach – Prentice Hall, Englewood Cliffs, New Jersey, 1995 – 932 с.