# การพัฒนาระบบปัญญาประดิษฐ์ สำหรับเกมโอเทลโล่

**Developing an Artificial Intelligent system for Othello game**

นายเข็มทิพย์ เข็มทอง รหัส 46360210



ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

ปีการศึกษา 2551


# ใบรับรองโครงงานวิศวกรรม

| | |
|---|---|
| **หัวข้อโครงงาน** | **การพัฒนาระบบปัญญาประดิษฐ์ สำหรับ เกมโอเทลโล่** |
| **ผู้ดำเนินโครงงาน** | **นายเข็มทิพย์ เข็มทอง รหัส 46360210** |
| **อาจารย์ที่ปรึกษา** | อาจารย์จิราพร พุกสุข |
| **สาขาวิชา** | วิศวกรรมคอมพิวเตอร์ |
| **ภาควิชา** | วิศวกรรมไฟฟ้าและคอมพิวเตอร์ |
| **ปีการศึกษา** | 2551 |

.....

**คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้โครงงานฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์**

**คณะกรรมการสอบโครงงานวิศวกรรม**

.....**ประธานกรรมการ**

**(อาจารย์จิราพร พุกสุข)**

.....**กรรมการ**

**(อาจารย์ภาณุพงศ์ สอนคม)**

.....**กรรมการ**

**(ดร.สุรเดช จิตประไพกุลศาล)**

| | |
|---|---|
| **หัวข้อโครงงาน** | **การพัฒนาระบบปัญญาประดิษฐ์ สำหรับ เกมโอเทลโล่** |
| **ผู้ดำเนินโครงงาน** | **นายเข็มทิพย์ เข็มทอง รหัส 46360210** |
| **อาจารย์ที่ปรึกษา** | **อาจารย์จิราพร พุกสุข** |
| **สาขาวิชา** | **วิศวกรรมคอมพิวเตอร์** |
| **ภาควิชา** | **วิศวกรรมไฟฟ้าและคอมพิวเตอร์** |
| **ปีการศึกษา** | **2551** |

.....


## บทคัดย่อ

โครงงานนี้ได้พัฒนาโปรแกรมปัญญาประดิษฐ์สำหรับเกมโอเทลโล่ **เพื่อเป็นการพัฒนาระบบคอมพิวเตอร์ให้มีความสามารถในการตัดสินใจโดยอาศัยการเล่นเกมตอบโต้กับคู่แข่ง** โปรแกรมนี้จะใช้หลักการตรวจสอบความเหมาะสมของแต่ละตำแหน่ง **และจากการคำนวณคะแนนเพื่อจะนำไปใช้ในการตัดสินใจเลือกตำแหน่งที่เหมาะสมที่สุดในการเล่นเกมโอเทลโล่รอบนั้นๆ จา**กผลการทดลองพบว่าโปรแกรมสามารถเล่นตอบโต้กับคู่แข่งได้อย่าง**ถูกต้อง และรวดเร็ว และยังสามารถคิดคาดเดาผลลัพธ์ล่วงหน้าได้หนึ่งระดับเพื่อใช้ในการตัดสินใจวางตัวหมากในตำแหน่ง**ต่างๆ ได้อย่างมีประสิทธิภาพ

| | | | |
|---|---|---|---|
| **Project title** | Development Artificial intelligent system for Othello game | | |
| **Name** | Mr.Khemthip | Khemthong | ID. 46360210 |
| **Project advisor** | Ms.Jiraporn | Pooksook | |
| **Major** | Computer Engineering. | | |
| **Department** | Electrical and Computer Engineering. | | |
| **Academic year** | 2008 | | |

.....


## Abstract

This project develops an artificial intelligent program for Othello Game. Computer system can make a decision to complete with opponent. This program is used to verify a suitable position and calculate score it can make a response efficiently and calculate forward in each turn. Experiments show that the programs can predict forward 1 turn in game.

# กิตติกรรมประกาศ

โครงงานวิศวกรรมคอมพิวเตอร์สำเร็จได้ด้วยดี ก็เนื่องจากความอนุเคราะห์จากท่านอาจารย์ที่ปรึกษา อาจารย์จิราพร พุกสุข รวมถึงคณะกรรมการโครงงาน อาจารย์ภาณุพงศ์ สอนคม และ ดร.สุรเดช จิตประไพกุลศาล ที่กรุณาให้คำปรึกษา แนะนำวิธีการในการทำงาน ตลอดถึงการตรวจสอบการทำงานพร้อมทั้งเสนอแนวทางการแก้ไขตลอดระยะเวลาการทำโครงงาน สุดท้ายต้องขอขอบพระคุณอาจารย์ทุกท่านเพื่อนกลุ่ม MD ทีม และเพื่อนๆ ทุกคนที่ยังไม่ได้เอ่ยนามที่คอยสนับสนุนในการทำโครงงานครั้งนี้

เข็มทิพย์ เข็มทอง

ผู้จัดทำโครงงาน

# สารบัญ

หน้า

# สารบัญ(ต่อ)

หน้า

# สารบัญตาราง

# สารบัญรูป

รูปที่ หน้า

# สารบัญรูป(ต่อ)

รูปที่ หน้า

# สารบัญรูป(ต่อ)

รูปที่ หน้า

# บทที่ 1
# บทนำ

## 1.1 ที่มาและความสำคัญของโครงงาน

เกมหมากกระดานนั้น มีการเล่นกันมานาน การเล่นหมากกระดานนั้นมีประโยชน์ เช่น ฝึกให้ผู้เล่นคิดแบบเป็นระบบ มีแบบแผน รู้จักวางกลยุทธ์ในแต่ละสถานการณ์ ทำให้มีการพัฒนาทั้งทางด้าน IQ (intelligence quotient หมายถึง ความฉลาดทางสติปัญญา, เชาว์, ไหวพริบ) และ EQ (emotion quotient หมายถึง ความฉลาดทางอารมณ์ สามารถแยกแยะอารมณ์ เข้ากับผู้อื่นได้) ในโครงงานนี้จะกล่าวถึง เกมหมากกระดานที่ชื่อว่า โอเทลโล่ ซึ่งเป็นเกมหมากกระดานที่ได้รับความนิยมเล่นจากผู้เล่นมากพอสมควร เพราะเป็นหมากกระดานที่เล่นง่าย สามารถเล่นได้ทุกเพศทุกวัย

การชนะเกมเมื่อเล่นจบเกมคือ การที่ผู้เล่นทั้งสองฝ่ายที่มีตัวหมากคนละสีกัน ลงหมากจนครบทุกช่องในกระดานแล้ว หรือผู้เล่นทั้งสองไม่สามารถลงหมากได้ โดยฝ่ายที่มีหมากสีของตนมากกว่าจะเป็นฝ่ายชนะ

จากวิธีการเล่นดังกล่าว การที่ฝ่ายใดจะเป็นผู้ชนะนั้น จึงต้องเดินหมากในตำแหน่งที่กินหมากคู่แข่งได้ และยึดครองตำแหน่งที่ได้เปรียบบนกระดานไปด้วย ตั้งแต่ต้นจนจบเกม

จากลักษณะดังกล่าวของเกมโอเทลโล โครงงานนี้มีจุดมุ่งหมายสำคัญที่จะพัฒนาระบบปัญญาประดิษฐ์(Artificial Intelligence) ขึ้นเพื่อให้คอมพิวเตอร์สามารถพัฒนาตนเองให้มีการเรียนรู้การตัดสินใจในการเลือกตำแหน่งวางหมากสำหรับเล่นเกม ในตำแหน่งที่ได้เปรียบฝ่ายตรงข้ามและสามารถเอาชนะคู่แข่งได้ โดยมีกระบวนการคิดตัดสินใจที่รวดเร็ว และถูกต้องแม่นยำ

## 1.2 วัตถุประสงค์โครงงาน

1. ศึกษาและเข้าใจกฎกติกาเทคนิคและวิธีการ ในการเล่นเกมหมากกระดานโอเทลโล่
2. สามารถสร้างและพัฒนา เกมโอเทลโล โดยใช้ ภาษา C#ได้
3. สามารถสร้างและพัฒนาในส่วนระบบปัญญาประดิษฐ์ ของเกมโอเทลโล โดยใช้ ภาษา C# ได้
4. สร้างและพัฒนากระบวนการการตัดสินใจของคอมพิวเตอร์ ในการเล่นเกมโอเทลโล่ โดยอาศัยหลักการด้านปัญญาประดิษฐ์

## 1.3 ขอบข่ายการทำงาน

1. ศึกษากฎกติกาเทคนิคและวิธีการในการเล่นเกมหมากกระดานโอเทลโล่
2. สร้างและพัฒนาตัวเกมโอเทลโล่และปัญญาประดิษฐ์ ด้วยภาษา C#
3. พัฒนาการเรียนรู้ของ ปัญญาประดิษฐ์ ด้วยหลักการ การตรวจสอบความเหมาะสมของตำแหน่ง ที่คิดและประยุกต์ขึ้น
4. ปัญญาประดิษฐ์สามารถเลือกตำแหน่งที่ดีที่สุดในระดับตาถัดไปได้ และตอบสนองได้ภายในระยะเวลา ไม่เกิน 1 วินาที

## 1.4 ขั้นตอนดำเนินงาน

| กิจกรรม | 2551 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | ก.พ. | | | | มี.ค. | | | | เม.ย. | | | | พ.ค. | | | | มิ.ย. | | | | ก.ค. | | | | ส.ค. | | | | ก.ย. | | | |
| | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| 1. ศึกษากฎกฎิการการเล่นหมากกระดานโอเทลโล่ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2. ศึกษาเทคนิควิธีการเล่นหมากกระดานโอเทลโล่ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3. ศึกษาเครื่องมือ และภาษาคอมพิวเตอร์ ที่ใช้ในการพัฒนาตัวเกมและปัญญาประดิษฐ์ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4. ประมวลผลข้อมูลทั้งหมดที่ศึกษามา | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5. สร้างและพัฒนาระบบ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6. ทดสอบประสิทธิภาพของระบบ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7. ทำการแก้ไขข้อผิดพลาด | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 8. สรุปข้อดีข้อเสียของระบบ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 9. จัดทำเอกสารประกอบโครงงาน | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

## 1.5 ผลที่คาดว่าจะได้รับ

1. เข้าใจกฎกฎิกา เทคนิค และ วิธีการเล่นเกมโอเทลโล่ อย่างถูกต้อง แม่นยำ
2. เข้าใจการเขียนภาษา C# และสามารถสร้างตัวเกมละปัญญาประดิษฐ์ของเกมโอเทลโล่ ตามหลักการ การตรวจสอบความเหมาะสมของตำแหน่ง ที่ต้องการได้
3. ปัญญาประดิษฐ์ของเกมสามารถใช้หลักการ การตรวจสอบความเหมาะสมของตำแหน่ง มาใช้ในการคิดล่วงหน้าได้
4. ปัญญาประดิษฐ์สามารถตอบสนองภายในระยะเวลา 1 วินาที

## 1.6 งบประมาณ

| | | | | |
|---|---|---|---|---|
| 1. | ค่าวัสดุสำนักงาน | เป็นเงิน | 400 | บาท |
| 2. | ค่าจัดพิมพ์เอกสาร | เป็นเงิน | 300 | บาท |
| 3. | ค่าถ่ายเอกสาร | เป็นเงิน | 300 | บาท |
| | | รวมเป็นเงินทั้งสิ้น | 1,000 | บาท (หนึ่งพันบาทถ้วน) |

# บทที่ 2
# หลักการและทฤษฎี

บทนี้จะกล่าวถึงหลักการและทฤษฎีในการคิดพัฒนาปัญญาประดิษฐ์ เพื่อใช้ในการคำนวณการเดินหมากของโอเทลโล่ โดยทำการพิจารณาแนวทางจากกฎกติกาในการเล่นเกมเพื่อใช้พัฒนาในส่วนของการเดินหมากที่ถูกต้อง และการสังเกตการเดินของฝ่ายตรงข้ามเพื่อหาแนวทางที่จะวางหมากแต่ละตัวให้มีความได้เปรียบมากที่สุด โดยอธิบายในหัวข้อดังต่อไปนี้

## 2.1 กฎกติกาในการเล่นเกม โอเทลโล่

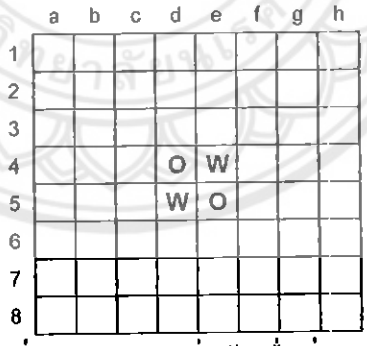
2.1.1 เกมโอเทลโล่ คือหมากกระดานที่ประกอบไปด้วย

- กระดานจัตุรัสที่ถูกแบ่งเป็นตาราง (โดยทั่วไปจะใช้ขนาด 8x8 ช่องเป็นมาตรฐาน)
- เม็ดหมากจำนวน 64 ตัว โดยตัวหมากจะมีลักษณะด้านหนึ่งเป็นสีขาว อีกด้านเป็นสีดำ

2.1.2 การเริ่มเล่นโอเทลโล่ ผู้เล่นทั้งสองต้องวางหมากในตำแหน่งเริ่มต้น ดังรูป และเริ่มเกมโดยหมากสีดำจะเป็นฝ่ายเดินก่อนเสมอ

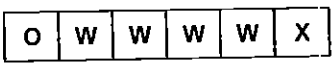
หมายเหตุ W แทนหมากสีขาว

O แทนหมากสีดำ




**รูปที่ 2.1** ตำแหน่งเริ่มต้นเมื่อเริ่มเกม

2.1.3 การพลิกหมากในเกมโอเทลโล่ จะเป็นการนำหมากสีของตน ไปวางบนช่องที่ปิดท้ายหมากของฝ่ายตรงข้าม (ในลักษณะหนีบหมาก) ทำให้หมากฝ่ายตรงข้ามถูกพลิกเป็นสีของตน




**รูปที่ 2.2** หมากดำจะเดินในตำแหน่งกากบาท

| | | | | | |
|---|---|---|---|---|---|
| o | o | o | o | o | o |

**รูปที่ 2.3** หลังหมากดำจะเดินในตำแหน่งกากบาทแล้ว
หมากขาวที่อยู่ระหว่างหมากดำจะถูกพลิกกลายเป็นสีดำ

**2.1.4 การเล่นโอเทลโลในแต่ละตาเดิน** ผู้เล่นจะต้องพลิกหมากคู่แข่งทุกตาจึงจะสามารถเดินได้ หากไม่มีตำแหน่งให้พลิก ผู้เล่นฝ่ายนั้นจะต้องผ่าน เพื่อให้อีกฝ่ายเป็นฝ่ายเดิน

**2.1.5 การพลิกหมาก** สามารถทำได้ทั้งแนวตั้ง แนวนอน และแนวทแยง และสามารถพลิกได้โดยไม่จำกัดจำนวน

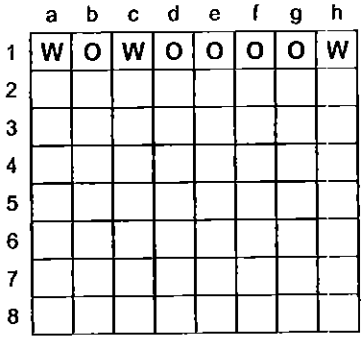
| | a | b | c | d | e | f | g | h |
|---|---|---|---|---|---|---|---|---|
| 1 | o | o | o | o | o | o | o | o |
| 2 | o | o | w | w | w | o | o | o |
| 3 | w | w | w | w | o | o | w | o |
| 4 | w | w | w | o | o | w | w | o |
| 5 | | w | w | o | o | w | w | o |
| 6 | o | o | w | w | w | o | w | o |
| 7 | | w | o | o | o | w | o | o |
| 8 | o | o | o | o | o | o | o | o |

รูปที่ 2.4 เป็นตาหมากดำเดินโดย หมากดำจะเดินในตำแหน่ง A5

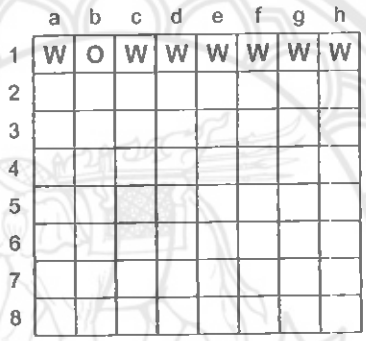
| | a | b | c | d | e | f | g | h |
|---|---|---|---|---|---|---|---|---|
| 1 | o | o | o | o | o | o | o | o |
| 2 | o | o | w | o | w | o | o | o |
| 3 | o | w | o | w | o | o | w | o |
| 4 | o | o | w | o | o | w | w | o |
| 5 | o | o | o | o | o | w | w | o |
| 6 | o | o | w | w | w | o | w | o |
| 7 | | w | o | o | o | w | o | o |
| 8 | o | o | o | o | o | o | o | o |

**รูปที่ 2.5** หลังจากหมากดำลงที่ตำแหน่งA5 หมากขาวจะถูกพลิกกลายเป็นสีดำ
ซึ่งในรูปนี้แสดงให้เห็นการพลิกหมากทั้งสามแนวคือแนวนอน (B5, C5)
แนวตั้ง (A3, A4) แนวทะแยง (B4, C3, D2)

**2.1.6 ไม่สามารถพลิกหมากสีของฝ่ายตนเองได้**

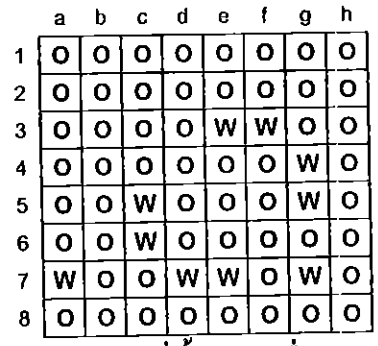



**รูปที่ 2.6** หมากขาวลงที่ตำแหน่ง H1 อาจมองดูเหมือนว่าพลิกได้ทั้งแถวในช่อง B1 – G1




**รูปที่ 2.7** แต่หมากขาวจะพลิกหมากดำได้แค่ D1 – G1 เท่านั้น
โดยหมากขาวที่ C1 จะไม่ถูกพลิกไปด้วย

2.1.7 วัตถุประสงค์ของเกม คือ การทำให้หมากมีจำนวนหมากกว่าหมากของฝ่ายคู่แข่งเมื่อเกมสิ้นสุดลง โดยเกมจะจบลงได้เมื่อทั้งสองฝ่ายลงจนครบทุกช่อง หรือ เมื่อทั้งสองฝ่ายไม่มีตำแหน่งให้เดินได้




**รูปที่ 2.8** แสดงตัวอย่างเกมที่สิ้นสุดลงเมื่อเดินจนครบทุกช่อง
โดยเกมนี้หมากดำเป็นฝ่ายชนะ 50 – 10

| | a | b | c | d | e | f | g | h |
|---|---|---|---|---|---|---|---|---|
| 1 | | W | W | W | W | W | W | W |
| 2 | W | O | O | O | W | W | W | W |
| 3 | W | O | O | W | O | W | W | W |
| 4 | W | O | W | O | W | W | W | W |
| 5 | W | O | W | W | O | W | O | W |
| 6 | W | W | W | O | W | O | O | W |
| 7 | W | W | O | O | W | W | O | W |
| 8 | W | W | W | W | W | W | W | |

**รูปที่ 2.9 แสดงตัวอย่างเกมที่สิ้นสุดลงเมื่อทั้งสองฝ่ายไม่มีตาเดินต่อ**
**โดยเกมนี้หมากขาวเป็นฝ่ายชนะ 45 - 17**

## 2.2 หลักการของปัญญาประดิษฐ์

ปัญญาประดิษฐ์ เป็นสาขาหนึ่งของศาสตร์ทางด้านคอมพิวเตอร์ที่มีส่วนเกี่ยวข้องกับการทำให้คอมพิวเตอร์สามารถทำงานได้คล้ายมนุษย์ ที่สามารถประมวลผลในลักษณะของการหาเหตุผล การตัดสินใจ การแก้ปัญหา การที่จะทำให้คอมพิวเตอร์ทำงานเหล่านี้ได้ จำเป็นที่จะต้องมีเครื่องคอมพิวเตอร์ที่มีศักยภาพในการประมวลผลที่มีประสิทธิภาพสูง

ในการพัฒนาระบบปัญญาประดิษฐ์โดยพื้นฐานแล้วผู้พัฒนาจะต้องเข้าใจและสามารถใช้เทคนิคในการสร้างโปรแกรม ที่จะทำให้คอมพิวเตอร์สามารถประมวลผลตามที่ต้องการได้ โดยระบบปัญญาประดิษฐ์นั้นจะต้องมีความรู้อยู่ด้วย เมื่อนำมาประมวลผลด้วยคอมพิวเตอร์แล้ว ระบบก็จะสามารถแก้ปัญหาบางอย่างได้เช่นเดียวกับมนุษย์

### 2.2.1 การแก้ปัญหาของปัญญาประดิษฐ์

เทคนิคของปัญญาประดิษฐ์เบื้องต้นประกอบไปด้วย วิธีการแก้ปัญหา หลักการในการกำหนดหลักเกณฑ์ของปัญหา ซึ่งถือเป็นกระบวนการพื้นฐานในการแก้ปัญหา (problem solving) ในเชิงคอมพิวเตอร์ การแก้ปัญหาโดยวิธีการของห้วงสถานะ (state and space) มีวิธีการคือ กำหนดจุดเริ่มต้นของปัญหา และเป้าหมายของการแก้ปัญหาว่าคืออะไร

### 2.2.2 ข้อสังเกตในการแก้ปัญหา

ระบบการผลิต เป็นกระบวนการในการวางโครงสร้างโปรแกรมแบบ AI ให้มีลักษณะง่ายต่อการอธิบายกระบวนการ หรือวิธีการแก้ปัญหา อันประกอบไปด้วย

1. กฎ ซึ่งแต่ละข้อจะต้องประกอบไปด้วย 2 ส่วน คือส่วนที่อธิบายถึงเงื่อนไขของกฎ และส่วนที่อธิบายถึงผลของกฎ
2. ฐานข้อมูลที่มีข้อมูลที่ต้องการ บางส่วนของฐานข้อมูลจะเป็นแบบถาวร และบางส่วนจะเป็นสิ่งที่เกี่ยวข้องกับการแก้ปัญหาในช่วงนั้น
3. กลไกในการควบคุม เป็นส่วนที่กำหนดลำดับของกฎที่จะนำมาใช้หรือเปรียบเทียบกับฐานข้อมูลเพื่อบอกลำดับของการแก้ปัญหา กลไกสำคัญในการ

ควบคุม ได้แก่ การกำหนดทิศทางสำหรับการค้นหา กระบวนการในการเลือกกฎ และการค้นหาที่เป็นแบบฮิวริสติก

ในการแก้ปัญหาของปัญญาประดิษฐ์นั้น มีเรื่องใหญ่ที่จะต้องพิจารณาถึงดังนี้

**2.2.3 การกำหนดทิศทางสำหรับการค้นหา**

โดยปกติแล้วการกำหนดทิศทางของการค้นหา มี 2 อย่างคือ

1. การหาเหตุผลแบบเดินหน้า (forward reasoning)
2. การหาเหตุผลแบบย้อนหลัง (backward reasoning)

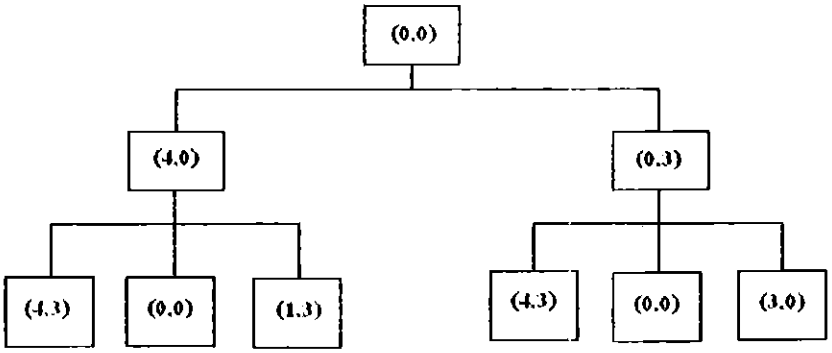
สิ่งเหล่านี้เป็นกระบวนการของการกำหนดทิศทางสำหรับการค้นหาและเลือกกฎที่เหมาะสม ในการเริ่มต้นค้นหาคำตอบจากส่วนใดของห้วงปัญหา และจัดเตรียมกฎให้เหมาะสม โดยปกติจะอาศัยโครงสร้างข้อมูล แบบต้นไม้ (tree structure) และ\หรือ แบบกราฟ (graph structure)

การหาเหตุผลแบบเดินหน้า จะนำสถานะเริ่มต้นมาเป็น root (root หมายถึง node แรกของโครงสร้างต้นไม้) ของโครงสร้างต้นไม้ และสร้าง node ลูก (successor) ( node หมายถึง จุดข้อมูลของโครงสร้างต้นไม้ เชื่อมต่อกัน เป็นระดับต่างๆ กัน โดยจะเริ่มต้นจาก root ) ในระดับถัดไป โดยการหากฎทุกข้อ ซึ่งทางด้านเงื่อนไขของกฎที่ตรงกับ root node และใช้ทางด้านผลของกฎสร้างเป็น node ลูก ของ node นั้น และสร้างระดับต่อไปด้วยการนำ node ลูก มากระทำด้วยวิธีเดียวกันจนครบทุก node ในระดับนั้น และทำจนครบกฎทุกข้อ

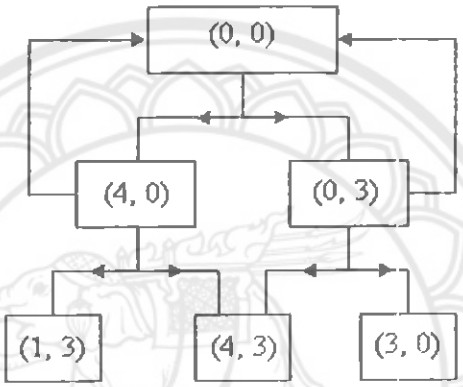
การหาเหตผลแบบย้อนหลัง จะนำสถานะเป้าหมายมาเป็น root ของโครงสร้างต้นไม้ และสร้าง node ลูก (successor) ในระดับถัดไป โดยการหากฎทุกข้อ ซึ่งทางด้านเงื่อนไขของกฎที่ตรงกับ root node และใช้ทางด้านเงื่อนไขของกฎสร้างเป็น node ลูก ของ node นั้น และสร้างระดับต่อไปด้วยการนำ node ลูก มากระทำด้วยวิธีเดียวกันจนครบกทุก node ในระดับนั้น และทำจนครบกฎทุกข้อ

**2.2.4 รูปแบบของโครงสร้างข้อมูลที่ใช้สำหรับการค้นหา**

โดยส่วนใหญ่แล้วโครงสร้างข้อมูล จะถูกกำหนดในรูปของ โครงสร้างต้นไม้ หรือ โครงสร้างกราฟ ซึ่งโครงสร้างทั้งสองนี้สามารถเปลี่ยนแปลงไปมาได้




**รูปที่ 2.10** โครงสร้างข้อมูลแบบต้นไม้




**รูปที่ 2.11** โครงสร้างข้อมูลแบบกราฟ

การสร้างรูปแบบโครงสร้างข้อมูลแบบกราฟ โดยทั่วไปแล้วจะดีกว่าแบบต้นไม้ เนื่องจากจะประหยัดพื้นที่หน่วยความจำ และค้นหาข้อมูลได้รวดเร็ว แต่มีข้อเสียคือ การเชื่อมต่อข้อมูลทำได้ยากใช้เวลาเชื่อมต่อนาน ดังนั้นการออกแบบโครงสร้างจึงนิยมสร้างเป็นแบบต้นไม้ก่อนแล้วจึงค่อยเปลี่ยนเป็นกราฟ

การเปลี่ยนแปลงกระบวนการค้นหาในโครงสร้างต้นไม้ (tree search procedure) เป็นกระบวนการค้นหาในโครงสร้างกราฟ (graph search procedure) สามารถทำได้ดังนี้

1. ตรวจสอบเซตของ node ที่ได้สร้างมาแล้วว่า node ที่จะสร้างใหม่มีอยู่หรือไม่
2. ถ้ายังไม่มี สร้าง node ใหม่ในระดับถัดไปให้กับโครงสร้างต้นไม้
3. ถ้ามี ให้ทำดังนี้

- ตั้งให้ node ที่จะสร้างขึ้นมาใหม่ชี้ไปยัง node ที่มีอยู่แล้ว และตัด node ที่จะเกิดทิ้งไป

- ในการตรวจสอบเส้นทางที่สั้นที่สุด ให้ทำการตรวจสอบทุกระดับ ที่มีการสร้างเส้นทางใหม่แล้ว เลือกเส้นทางที่สั้นที่สุด

**2.2.5 การแสดงความรู้**

เป็นกระบวนการในการแสดงความหมายที่ปรากฏอยู่ในแต่ละ node ว่าจะมีวิธีการเช่นไรในการแสดงความรู้ โดยปกติแล้ว การแสดงความรู้เป็นแบบออบเจ็คต์ที่เชื่อมต่อกันด้วยตัวเลขแสดงความสัมพันธ์เป็นวิธีการที่ง่าย แต่มีข้อจะต้องระวังคือ

1. ความรู้ทั้งหมดจะสามารถรวมเป็นความรู้เดียวกันได้อย่างไร เช่น หากว่าเรากำลังอธิบายว่า 'table is under the window' แล้วเมื่อเปลี่ยนฐานความรู้ว่า CENTER (table, room) ระบบจะรู้ได้อย่างไรว่า UNDER (table, window) ใช้งานไม่ได้แล้ว

2. การจัดลำดับอย่างไรให้การค้นหาทำได้ง่าย เช่น ถ้าจะเติม ABOVE (ceiling, floor) เข้าไปในฐานความรู้ จะใส่ตรงไหนเพื่อไม่ให้มีการต้องบอกทุกครั้งเมื่อมีการกล่าวอ้างถึงเรื่องของห้อง

**2.2.6 กระบวนการในการเลือกกฎ**

โดยปกติการเลือกกฎจะกระทำโดยการเปรียบเทียบ (maching) ซึ่งทำได้โดยการนำ สถานะปัจจุบัน (current state) ไปเปรียบเทียบกับส่วนเงื่อนไขของกฎ กระบวนการเปรียบเทียบนี้นิยมใช้อยู่ 2 วิธีคือ

1. การทำดัชนี (index) หลักการของการแก้ปัญหาแบบนี้คือ ให้หากฎทุกข้อที่มีเงื่อนไขของกฎตรงกับเงื่อนไขที่กำหนดในสถานะปัจจุบัน แล้วดึงเอากฎทุกข้อที่ได้ มาทำการเปรียบเทียบหาคำตอบ เมื่อได้คำตอบการแก้ปัญหาจะสิ้นสุดลง ถ้าหากไม่พบคำตอบ ให้ทำการเปรียบเทียบใหม่ จนพบคำตอบ

2. การเปรียบเทียบด้วยตัวแปร (matching with variable) การแก้ปัญหาแบบนี้เป็นการอาศัยการเปรียบเทียบ ตัวแปรของกฎและความจริง ดังตัวอย่างต่อไปนี้

Fact : แดงใหญ่ เป็นบิดา ของแดง

Fact : แดง เป็นบิดา ของแดงเล็ก

จากความเป็นจริงดังกล่าว คอมพิวเตอร์จะไม่สามารถบอกความสัมพันธ์ระหว่างแดงใหญ่กับแดงเล็กได้ จึงต้องมีกฎบางอย่างที่กล่าวถึงเรื่องดังกล่าว

Rule : ถ้า X เป็นบิดา ของ Y และ

Y เป็นบิดา ของ Z

ดังนั้น X เป็นปู่ ของ Z

เมื่อมีความจริงดังกฎที่ได้แสดงแล้ว การแทนค่าความจริงด้วยตัวแปลเหล่านี้แล้ว จะได้ความสัมพันธ์ว่าแดงใหญ่เป็นปู่ของแดงเล็ก เราเรียกลักษณะแบบนี้ว่าการอนุมาน การอนุมานนี้จะเลือกกฎตามความสัมพันธ์ (relation) ที่เหมือนกันกับความจริงเท่านั้น

**2.2.7 การค้นหาแบบฮิวริสติก**

ทางด้านปัญญาประดิษฐ์ การค้นหาคำตอบอาศัยวิธีการทางฮิวริสติก (heuristic search) ความแตกต่างของการค้นหาข้อมูลแบบธรรมดา และแบบฮิวริสติกนั้นอยู่ที่ การค้นหาข้อมูลธรรมดา ผู้ที่ทำการค้นข้อมูลนั้นจะต้องตรวจสอบข้อมูลที่ละตัวจนครบทุกตัวจนครบ แต่ฮิวริสติกจะไม่ลงไปดูข้อมูลทุกตัว วิธีการนี้จะเลือกได้คำตอบที่เหมาะสมให้กับการค้นหา ซึ่งมีข้อดีคือสามารถทำการค้นหาคำตอบจากข้อมูลที่มีขนาดใหญ่มากๆ ได้ แต่มีข้อเสียคือคำตอบที่ได้เป็นคำตอบที่ดีเท่านั้น ไม่แน่ว่าจะดีที่สุด แต่เนื่องจากว่า ปัญหาในบางลักษณะนั้นใหญ่มาก และเป็นไปไม่ได้ที่จะทำการค้นหาด้วยวิธีธรรมดา กระบวนการของฮิวริสติกจึงเป็นสิ่งจำเป็น

นอกจากการค้นหาแบบฮิวริสติกแล้ว ยังมีอีกสิ่งหนึ่งที่สำคัญคือ ฮิวริสติกฟังก์ชัน (heuristic function) ซึ่งหมายถึง ฟังก์ชันที่ทำหน้าที่ในการวัดขนาดของความเป็นไปได้ ในการแก้ปัญหาซึ่งจะแสดงด้วยตัวเลข

วิธีการดังกล่าวจะกระทำได้โดยการพิจารณาถึงวิธีการ (aspects) ต่างๆ ที่ใช้ในการแก้ปัญหา ณ สถานะหนึ่งว่าสามารถแก้ปัญหาได้ตามที่ต้องการหรือไม่ โดยกำหนดเป็นน้ำหนักที่ให้กับการแก้ปัญหาของแต่ละวิธี น้ำหนักเหล่านี้จะถูกแสดงด้วยตัวเลขที่กำกับไว้กับ node ต่างๆ ในกระบวนการค้นหาและค่าเหล่านี้จะเป็นตัวที่ใช้ประมาณความเป็นไปได้ว่า เส้นทางที่ผ่าน node นั้นจะมีความเป็นไปได้ในการนำไปสู่หนทางการแก้ปัญหาได้มากน้อยแค่ไหน

จุดประสงค์ที่แท้จริงของฮิวริสติก ฟังก์ชันก็คือ การกำกับทิศทางของกระบวนการค้นหาเพื่อให้อยู่ในทิศทางที่ได้ประโยชน์มากที่สุด โดยการบอกว่าเราควรเลือกเดินทางไหนก่อน ในกรณีที่มีเส้นทางมากกว่าหนึ่งเส้นทางต้องเลือก

**ตัวอย่างปัญหา**

ปัญหา เซลล์แมนต้องเดินทางไปยังเมืองจำนวน N เมือง โดยเริ่มจากเมืองหนึ่งและเดินทางไปยังแต่ละเมือง (เมืองละ 1 ครั้ง) แล้วกลับมาสิ้นสุด ณ เมืองที่เริ่มต้น จึงพยายามที่จะหาเส้นทางที่ดีที่สุด (ในด้านค่าใช้จ่าย หรือระยะทางที่สั้นที่สุด)

ความยุ่งยาก จำนวนของเส้นทาง(นับเพียงเส้นทางทิศทางเดียว) เมื่อ N คือจำนวนเมือง คือ $R = 0.5(N-1)!$ สำหรับเมือง 10 เมืองจะมีเส้นทางถึง 181,440 เส้นทาง ถ้ามี 11 เมืองจะมีถึงสองล้านเส้นทาง และถ้ามี 20 เมืองจะมีถึง $6.1 \times 1016$ เส้นทาง จะเห็นว่าการเพิ่มจำนวนเมืองเพียงเล็กน้อยจะทำให้เกิดทางเลือกในการแก้ปัญหาเพิ่มขึ้นมากมาย

วิธีแก้ปัญหา การใช้วิธีการค้นหาโดยการอ้างอย่างสมบูรณ์ (complete enumeration) และการใช้อัลกอริธึมไม่มีประสิทธิภาพและไม่มีประสิทธิผลเพียงพอเนื่องจากมีทางเลือกค่อนข้างมาก การใช้วิธีฮิวริสติกสามารถแก้ปัญหาลักษณะนี้ได้ดี และอาจใช้เวลาเพียงไม่นาน วิธีแก้ปัญหาโดยใช้ฮิวริสติก คือ "เริ่มที่เมืองใดๆ และเดินทางต่อไปยังเมืองที่ใกล้ที่สุด ทำแบบนี้เรื่อยๆ จนกระทั่งเดินทางถึงเมืองสุดท้าย และกลับมาที่เมืองเริ่มต้น" วิธีแก้ปัญหาอีกวิธีหนึ่ง คือ ใช้การลองผิดลองถูก

คือ "เริ่มที่เมืองใดๆ แล้วเดินทางไปยังเมืองที่อยู่ภายนอกก่อน โดยไม่มีการเดินตัดเส้นทางที่เดินไปแล้วและไม่มีการย้อนกลับ แล้วจึงกลับมาสิ้นสุดที่เมืองเริ่มต้น"

**ตัวอย่างการค้นหาแบบฮิวริสติก**

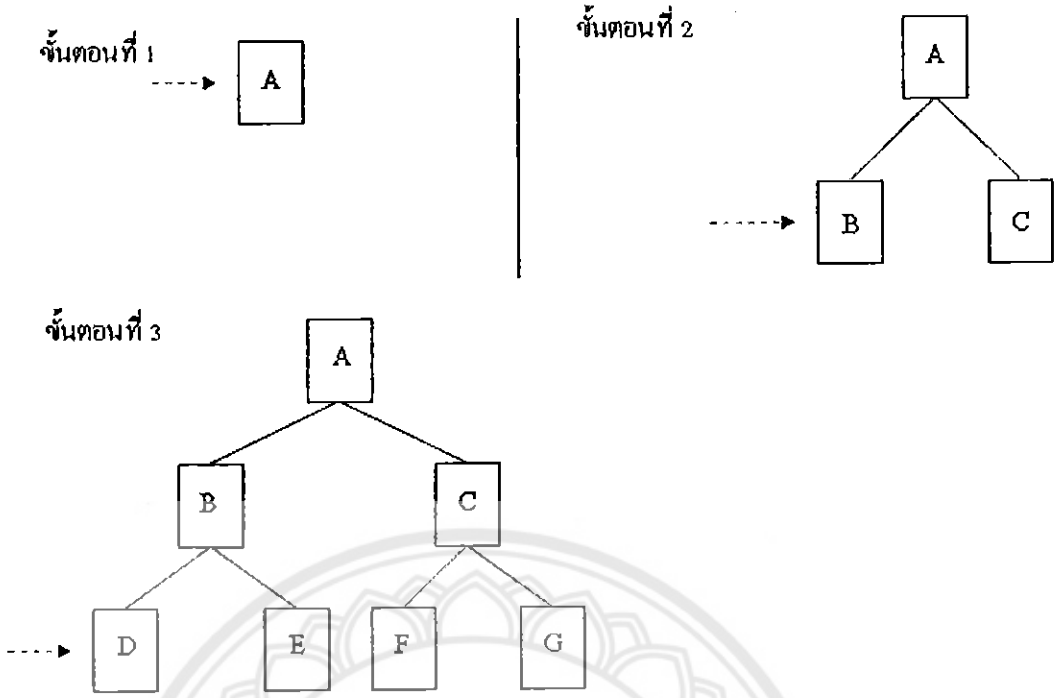
**Breadth-first search**

ทุก node ที่อยู่ในระดับเดียวกันของต้นไม้จะถูกตรวจสอบก่อน node ที่อยู่ในระดับถัดไป วิธีการของการค้นหาแบบนี้ทำโดยเริ่มจากการสร้าง node ลูกให้กับสถานะเริ่มต้นก่อน แล้วทำการตรวจสอบว่ามี node ใดที่เป็นเป้าหมายได้หรือไม่ ถ้าหากว่ามีก็เป็นอันว่าการค้นหาสิ้นสุด ถ้าไม่มีจาก node ที่เป็น node ลูกทุกตัวดังกล่าวข้างต้น ให้สร้าง node ลูกกับ node เหล่านั้น แล้วทำการตรวจสอบ node ลูกทุกตัวของ node เหล่านั้น ถ้าพบ เป้าหมายการค้นหาก็สิ้นสุด ถ้าไม่พบก็ให้ทำการสร้าง node ลูกให้กับ node เหล่านั้นต่อไปอีก ทำเช่นนี้ไปเรื่อยๆ จนกว่าจะพบเป้าหมาย หรือจนไม่สามารถสร้าง node ลูกใหม่ได้อีกแล้ว

จากรูปที่ ขั้นตอนที่ 1 ตรวจสอบรูท node A จากนั้นขั้นตอนที่ 2 สร้าง node ลูกให้กับ A (คือ B และ C) แล้วตรวจสอบ B และ C ในขั้นตอนที่ 3 สร้าง node ลูกให้กับ B และ C (คือ D, E, F และ G) ตามลำดับ

การค้นหาแบบ Breadth-first search จะมีข้อเสียคือ

- ใช้หน่วยความจำมาก เพราะจำนวนของ node ในแต่ละระดับจะเพิ่มขึ้นแบบทวีคูณ (exponential)
- การทำงานจะมีขั้นตอนมาก
- การทำงานที่ไม่จำเป็นและการทำงานที่ซ้ำซ้อนจะมีมาก




รูปที่ 2.12 Breadth-first search

## 2.3 ภาษาที่ใช้ในการพัฒนา

.Net Framework เป็นแพลตฟอร์มใหม่ ที่พัฒนาขึ้นโดย บริษัท Microsoft เพื่อใช้สำหรับการพัฒนา Application .Net Framework โดยถูกออกแบบมาเพื่อให้สามารถ ถูกใช้ได้จากหลายภาษา เช่น C#, C++, Visual Basic, Jscrupt, Delphi และอื่นๆ เพื่อให้การทำงานเหล่านี้เป็นไปได้ จึงเกิดภาษาเหล่านี้ขึ้นมาในรูปของ Version เฉพาะ สำหรับ .Net ซึ่งได้แก่ Managed C++, Visual Basic.Net, Jscript.Net, Borland C#, Delphi8 เป็นต้น นอกจากนี้ .Net Framework ยังสามารถสื่อสารกับภาษาอื่นได้ด้วย

.Net Framework มีพื้นฐานประกอบขึ้นด้วย Library ของ Source Code ขนาดใหญ่ ซึ่งถูกเรียกใช้จากภาษา Client ที่มีอยู่ เช่น C#, C++.Net โดยการใช้เทคนิคเชิงวัตถุ (OOP) Library ดังกล่าว ถูกแบ่งกลุ่มออกเป็น Modul ต่างๆ ดังนั้นจึงใช้ตามผลลัพธ์ที่ต้องการได้ เช่น Windows Application เป็นต้น โดยจุดมุ่งหมายคือ ระบบปฏิบัติการ ที่แตกต่างกัน อาจจะสนับสนุน Modul เหล่านี้ บาง Modul หรือทั้งหมด ขึ้นอยู่กับคุณลักษณะของระบบปฏิบัติการ เช่น PDA จะรวมเอาการสนับสนุน Function หน้าที่ที่เป็นแก่นของ .Net ทั้งหมดเป็นต้น

ส่วน Library.Net Framework ได้กำหนดชนิดข้อมูลพื้นฐานบางอย่างเอาไว้ ชนิดข้อมูลเป็นตัวแทนของข้อมูล และการแบ่งกฎเกณฑ์เหล่านี้ จะส่งเสริมความสามารถในการสัมพันธ์ระหว่างภาษา โดยใช้ .Net Framework สิ่งนี้ถูกเรียกว่า Common Type System (CTS) เช่นเดียวกับการจัดให้

มี Library .Net Common Language Runtime (CLR) ซึ่งรับผิดชอบในการจัดการ กับระบบปฏิบัติการของ Application ทั้งหมดที่ถูกพัฒนาขึ้นมาด้วย Library .Net Framework

ไวยากรณ์ ของ C# ถือเป็นการผสมผสานกัน ของ code ของ C#, C++ และ Java จึงทำให้ code อ่านได้ง่าย C# นั้นไม่มีการแจ้งเตือน เกี่ยวกับช่องว่างที่อยู่ใน code ดังนั้นไม่ว่าโปรแกรม จะมี white space มากเท่าไรก็ได้ ทำให้สามารถที่จะจัดรูปแบบ source code ได้ตามความต้องการ แต่ก็จะต้องทำตามกฎที่มีอยู่ code C# นั้นสร้างขึ้นจาก statement ชุดหนึ่งและแต่ละ statement จะจบด้วยเครื่องหมาย ; (semi colon) เนื่องจาก white space นั้นจะถูกมองข้ามไป จึงสามารถสร้างหลายๆ statement ในบรรทัดเดียวกันได้ หรือจะพิจารณาใช้ carriage return เข้าไปด้านหลังเครื่องหมาย ; ซึ่งเป็นสิ่งที่ยอมรับได้ C# เป็นภาษาที่มีโครงสร้าง เหมือนกับ C++ ทุกประการ ดังนั้นหากมีพื้นฐานทางด้านของ C++ การศึกษา C# ก็จะทำได้ง่าย

ข้อดีของ C# นั้นถือว่าเป็นภาษาหลักของ .Net และ C# นั้นสามารถเขียนเพื่อใช้งานในด้านโปรแกรมที่ซับซ้อน เช่น โปรแกรมควบคุมฮาร์ดแวร์ และโปรแกรมสามมิติ โปรแกรมประยุกต์ขนาดใหญ่ ได้ง่าย จุดเด่นหลักๆ ของภาษา C# มีอยู่ 4 ประการดังนี้

1. เป็นภาษาที่เน้นชิ้นส่วน (Component oriented) แนวคิดเรื่องซอฟแวร์ชิ้นส่วน ภาษา C# เอื้ออำนวยการใช้งานในลักษณะนี้ โดยให้สร้างพร็อปเพอร์ตี้ม, เมธอด และอีเวนต์ได้ง่ายๆ เป็นเอกสารอธิบายการทำงานไปในตัว การดึงโค้ดที่เคยเขียนไว้มาใช้งานสามารถทำได้ง่าย ไม่ต้องทำเป็นไฟล์ header ที่ใช้งานไม่ค่อยสะดวก
2. สิ่งต่างๆ เป็นออบเจ็กต์ทั้งหมด ในภาษา C++ และ JavaScrip ไทป์พื้นฐานต่างๆ (เช่น int, double) ถือเป็น primitive type คือเป็นไทป์ที่ฝังอยู่ในภาษาจึงดูเหมือนไม่มีที่มา เพราะสามารถเรียกใช้ได้อย่างลอยๆ ส่วนภาษา Smalltalk และภาษา Lips ไทป์พื้นฐานล้วนเป็นออบเจ็กต์ ซึ่งเป็น reference type ทั้งหมดทำให้การทำงานมีประสิทธิภาพลดลง ส่วนภาษา C# แม้ทุกอย่างเป็นออบเจ็กหมด แต่ไทป์พื้นฐานจะมีภาวะเป็น value type จึงไม่ลดทอนประสิทธิภาพ ยกตัวอย่างเช่น

```
Int foo;
```

คือการนิยามตัวแปรชื่อ foo มีไทป์เป็น int หรือจำนวนเต็ม เนื่องจากไทป์ int แม้จะดูว่าเป็นไทป์พื้นฐาน แต่ความจริงคือคลาส Sysyem.int32 ดังนี้ foo จึงมีภาวะเป็นออบเจ็กต์ แต่จะพิเศษกว่าตรงที่เป็น value type

```
Foo + 1;
```

จากโค้ด แม้เลข 1 จะเป็นตัวเลขที่พิมพ์เข้าไป (literal) แต่ก็ยังกลายเป็นออบเจ็กต์ที่มีไทป์เป็น System.int32 ไปด้วย เนื่องจากทุกไทป์ล้วน inherit จาก System.Object เลข 1 จึงสามารถเรียกใช้เมธอดต่างๆ ที่นิยามไว้ใน System.Object ได้ ยกตัวอย่างโค้ดดังนี้

String Bar = 1.ToString();

จะเห็นว่าเลข 1 มีเมธอด ToString ได้ด้วย เพราะ C# มองว่าเป็นออบเจ็กตัวหนึ่ง

3. เป็นภาษาที่ทนทาน คำว่าทนทาน (robust) ในที่นี้หมายถึงทนต่อความผิดพลาด ไม่ทำให้ระบบแฮงก์หรือระบบช้า ทั้งนี้เพราะข้อดีของ C# ดังนี้

**Garbage collection** เป็นกลไกบริหารหน่วยความจำเพื่อทำลายออบเจ็กที่หมดเงื่อนไขในการดำรงอยู่ ป้องกันการเกิดอาการ "หน่วยความจำรั่ว" (memory leaks)

**Exceptions** เป็นตัวดักจับและรายงานความผิดพลาดแบบ run-time error

**Type-safety** ป้องกันการเรียกใช้ตัวแปรที่ยังไม่ได้ถูกกำหนดค่าเริ่มต้น และป้องกัน casts ที่ไม่ปลอดภัย

Versioning เพื่อรักษาความปลอดภัยในการใช้คลาสและการสืบทอด

สิ่งต่างๆ เหล่านี้ทำให้ภาษา C# มีคุณสมบัติ industrial strength คือสามารถนำไปใช้เขียนโปรแกรมประยุกต์ในลักษณะที่มีความเค้นสูง หรืออ่อนไหวต่อความผิดพลาดสูงได้ดี

4. เขียนโค้ดแล้วต่อเนื่องได้ ภาษา C# จัดเตรียมกลไกไว้หลายอย่างที่ช่วยให้นำโค้ด ที่เขียนไว้ในโปรเจ็คหนึ่งไปใช้กับอีกโปรเจ็คหนึ่งได้ง่าย และภาษา C# ยังสามารถเรียกใช้คลาสหลายพันคลาสที่อยู่ภายใน .Net Framework ได้โดยตรง ทำให้ลดเวลาการพัฒนาซอฟต์แวร์ได้มาก

# บทที่ 3
# วิธีการดำเนินการ

ในบทนี้จะกล่าวถึงวิธีการดำเนินการของปัญญาประดิษฐ์ในการค้นหาตำแหน่งที่เหมาะสมในการเดินหมาก โดยจะเริ่มต้นการการที่ปัญญาประดิษฐ์ทำการเก็บคะแนนรวมจากการคำนวณ ในแต่ละรอบเพื่อให้ได้คะแนนรวมที่ถูกต้องเหมาะสมที่สุดแล้วจึงพิจารณาหาตำแหน่ง ที่มีคะแนนมากและเหมาะสมที่สุดเพื่อตัดสินใจเดินในตำแหน่งที่ดีที่สุดสำหรับปัญญาประดิษฐ์ ซึ่งรายละเอียดของแต่ละขั้นตอนในการดำเนินการจะแสดงดังนี้

## 3.1 แนวคิดการทำงานของปัญญาประดิษฐ์

| | a | b | c | d | e | f | g | h |
|---|---|---|---|---|---|---|---|---|
| 1 | G | C | S | | | S | C | G |
| 2 | C | C | | | | | C | C |
| 3 | S | | | | | | | S |
| 4 | | | | O | W | | | |
| 5 | | | | W | O | | | |
| 6 | S | | | | | | | S |
| 7 | C | C | | | | | C | C |
| 8 | G | C | S | | | S | C | G |

รูปที่ 3.1 แสดงตำแหน่งต่างๆ ที่ได้เปรียบเสียเปรียบบนกระดานโอเทลลโล่

หมายเหตุ

O หมายถึง หมากสีดำ

W หมายถึง หมากสีขาว

C หมายถึง ตำแหน่งที่เสียเปรียบของกระดาน ได้แก่ ตำแหน่งที่ติดมุมทั้งแนวนอน แนวทแยง และ แนวตั้ง A2, A7, B1, B2, B7, B8, G1, G2, G7, G8, H2 และ H7

S หมายถึง ตำแหน่งที่ได้เปรียบของขอบระดานทั้ง 4 ด้านของกระดาน คือ A3, A6, C1, C8, F1, F8, H3 และ H6

G หมายถึง ตำแหน่งที่ได้เปรียบของกระดานทั้ง 4 มุมของกระดาน คือ A1, A8, H1 และ H8

**3.1.1 แนวการคิดของปัญญาประดิษฐ์มีดังนี้**

แนวคิดของ AI คือ คิดแบบพยายามกันไม่ให้คู่แข่งเดินในตำแหน่งที่ได้เปรียบได้ก่อนก่อนที่จะลงเพื่อให้ AI ได้ครอบครองตำแหน่งได้เปรียบของเกม หรือ ชนะเกม จะมีการมองเกมล่วงหน้าว่าจะเกิดอะไรขึ้นถ้าลงในแต่ละตำแหน่ง เช่นหากเดินหมากในตำแหน่งนี้แล้ว จะทำให้ตาเดินของฝ่ายตรงข้ามลดลง จะบีบให้คู่แข่งต้องลงในจุดที่เสียเปรียบ จะทำให้ AI ชนะเกมเลยหรือไม่ โดยในการคิดว่าจะเดินตำแหน่งใดนั้น AI จะใช้วิธีเก็บคะแนนความเหมาะสมของแต่ละตำแหน่งที่สามารถลงหมากได้ และ หากตำแหน่งใดได้คะแนนมากที่สุดก็ควรจะเดินในตำแหน่งนั้น

**อธิบายเพิ่มเติม**

1. **เดินหมากแล้วทำให้คู่แข่งต้องเดินในตำแหน่งเสียเปรียบ มีอยู่ 2 ส่วน คือ**

ตำแหน่งที่เสียเปรียบของกระดาน หมายถึง ตำแหน่ง C ดังที่แสดงไว้ในรูป เพราะหากฝ่ายใดมีตัวหมากอยู่ในตำแหน่งนี้ จะมีโอกาสอย่างสูงที่คู่แข่งจะครอบครองตำแหน่งมุม ภายในอย่างน้อยที่สุด 2 รอบการเดิน

ตำแหน่งที่เสียเปรียบของเกม หมายถึง ตำแหน่งที่คู่แข่งเดินแล้ว AI จะสามารถพลิกหมากของคู่แข่งได้หมดทั้งแถว หรือ ได้ครอบครองตำแหน่งได้เปรียบอื่นๆ ของกระดาน

2. เมื่อเดินหมากแล้วทำให้คู่แข่งเหลือตำแหน่งที่สามารถเดินได้ในรอบถัดไปน้อยที่สุด ยกตัวอย่าง เช่น หากตรวจสอบแล้วพบว่าAI เดินได้ 4 ตำแหน่ง แต่ละตำแหน่งเมื่อเดินแล้ว ทำให้ตาเดินของคู่แข่งเป็น 10, 7, 5 และ 2 ตำแหน่ง ตามลำดับ ให้เลือกเดินในตำแหน่งที่คู่แข่งเหลือตาเดินเพียง 2 ตำแหน่ง
3. เดินหมากในตำแหน่งที่ได้เปรียบของกระดาน ซึ่งมีอยู่สองลักษณะ คือ

ตำแหน่งมุมทั้งสี่มุมของกระดาน (G) เพราะว่า เมื่อ AI ครอบครองแล้วหมากจะไม่ไม่สามารถถูกพลิกได้อีก

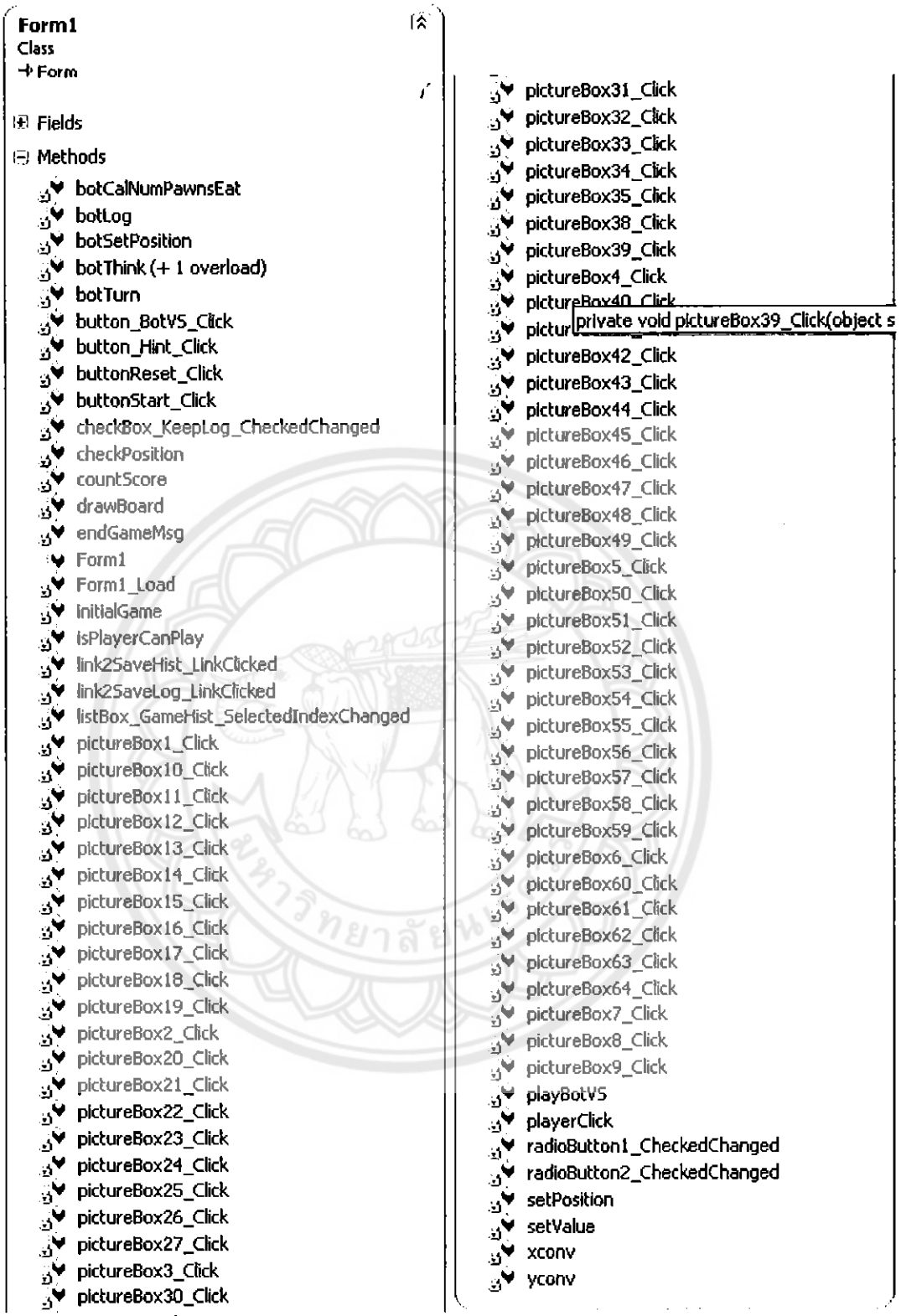
ตำแหน่งขอบกระดานทั้ง 4 ด้านของกระดาน (S) เพราะว่า เมื่อ AI ครอบครองตำแหน่งดังกล่าวแล้ว หากมีหมากของคู่แข่งลงในตำแหน่งที่ติดกับตำแหน่งนี้แล้ว AI จะสามารถพลิกหมากของคู่แข่งได้ทันที

4. เดินหมากแล้วทำให้หมากทั้งหมดบนกระดานเป็นหมากสีเดียวกับAI หมายถึงการชนะเกมในทันทีที่เดินหมากในตำแหน่งนี้
5. พิจารณาต่อไปว่าจะเกิดอะไรขึ้นเมื่อเดินในตำแหน่งดังกล่าว
   ไม่เดินหมากในตำแหน่งที่เมื่อเดินแล้วทำให้คู่แข่งสามารถชนะเกมได้ทันที
   ไม่เดินหมากในตำแหน่งที่เมื่อเดินแล้วทำให้คู่แข่งครอบครองตำแหน่งมุมได้
   ไม่เดินหมากในตำแหน่งที่เมื่อเดินแล้วทำให้คู่แข่งครอบครองตำแหน่งที่ได้เปรียบ

## 3.2 Class Diagram ของโปรแกรม

โปรแกรมเกมโอเทลโล่เกม มีองค์ประกอบการทำงานดังนี้ คือ

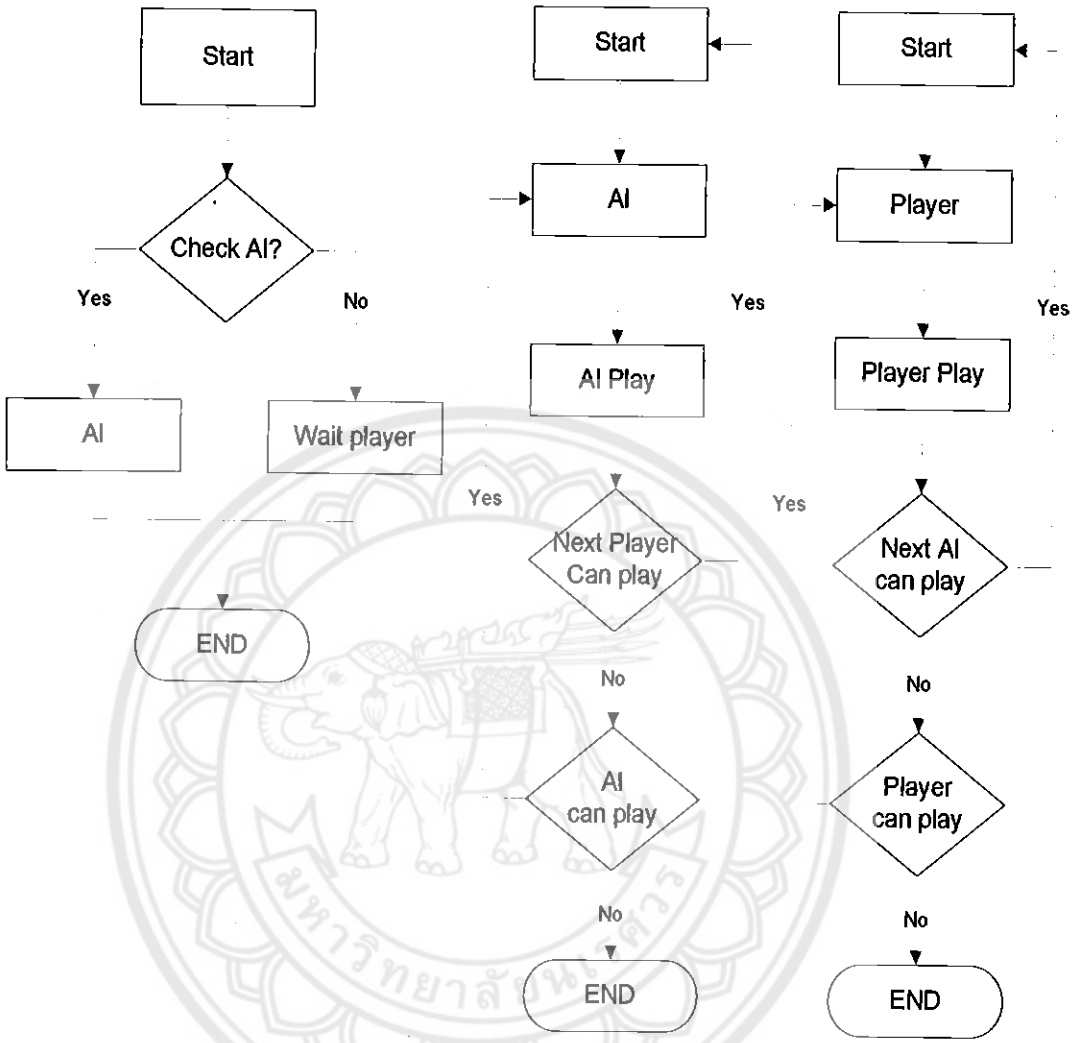

**รูปที่ 3.2** ภาพแสดง Class diagram ของโปรแกรมโอเทลโล่เกม




**รูปที่ 3.3** ภาพแสดง Class diagram ของโปรแกรมโอเทลโล่เกม (ต่อ)

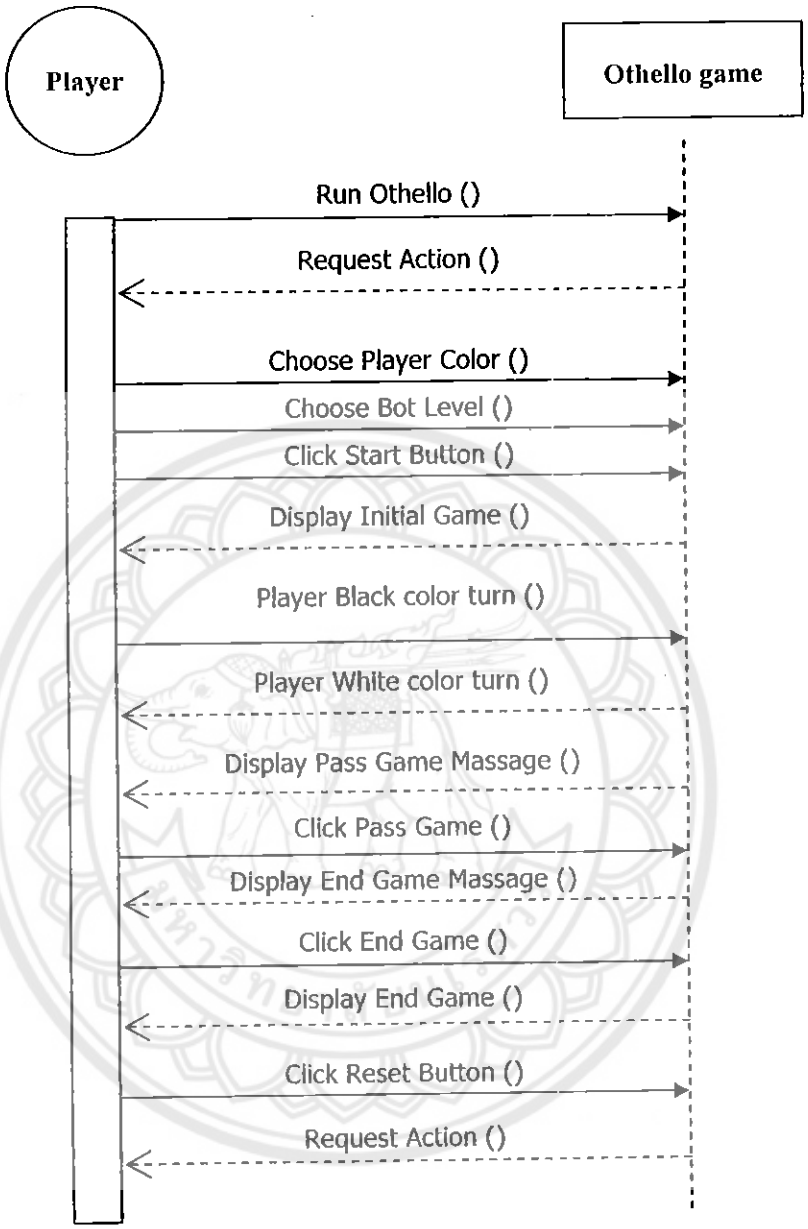
Class Form1 นี้จะเป็น class เดียวของโปรแกรม โดยจะมีฟังชันก์การทำงานต่างๆ อยู่ภายใน Form1 ทั้งหมด

## 3.3 โครงสร้างการทำงานของโปรแกรม




รูปที่ 3.4 ภาพ Diagram แสดงการทำงานของโปรแกรม

### 3.4 Sequence Diagram ของโปรแกรม



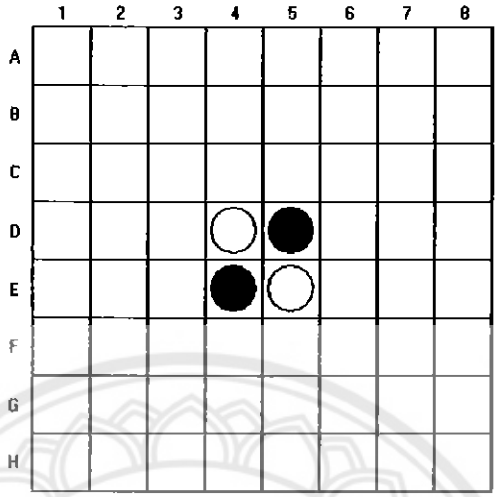

**รูปที่ 3.6** ภาพแสดง Sequence Diagram การเล่นโปรแกรม โอเทลโล่เกม

### 3.5 วิธีการคิดของปัญญาประดิษฐ์

ในการเดินหมากของปัญญาประดิษฐ์นั้น การที่ปัญญาประดิษฐ์จะตัดสินใจเดินในตำแหน่งใดก็ตาม ระบบจะตัดสินใจจากการคิดคะแนนของปัญญาประดิษฐ์ โดยหลักการคิดคะแนนเหล่านี้สามารถอธิบายได้ เป็นขั้นตอนดังนี้

### 3.5.1 การเริ่มต้นเกม

เมื่อเริ่มเกมขึ้นมา จะมีหมากสีละสองหมาก วางไขว้กันอยู่ตรงกลางกระดานดังรูป

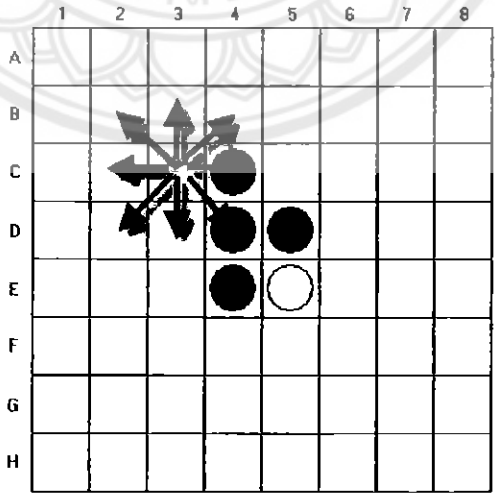



รูปที่ 3.6 ภาพแสดงตำแหน่งของหมากบนกระดานเมื่อเล่นเริ่มเล่นเกม

### 3.5.2 การเช็คตำแหน่งที่สามารถเดินได้ของปัญญาประดิษฐ์

กำหนดให้หมากสีขาว คือปัญญาประดิษฐ์ คู่แข่งคือหมากสีดำ

วิธีการค้นหาตำแหน่งที่สามารถเดินได้ของปัญญาประดิษฐ์ จะมีวิธีทำได้ดังนี้คือ ปัญญาประดิษฐ์จะทำการเช็คในตำแหน่งที่ว่างอยู่ โดยตำแหน่งว่างดังกล่าวนั้นจะต้องติดกับหมากของคู่แข่งด้วย




รูปที่ 3.7 ภาพแสดงการเช็คตำแหน่ง ที่สามารถเดินได้ของปัญญาประดิษฐ์

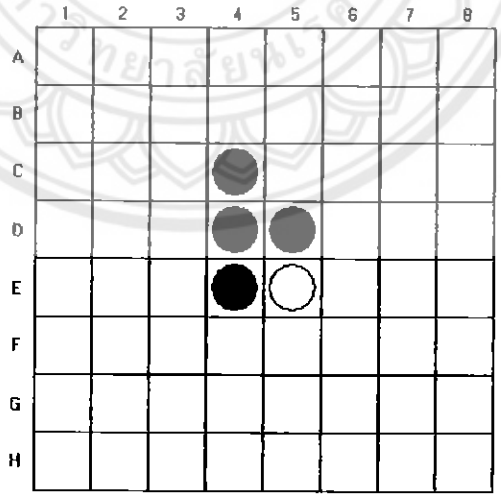
จากภาพตำแหน่งที่ปัญญาประดิษฐ์จะเช็คก็คือ ตำแหน่งว่างที่ติดกับหมากของคู่แข่ง ซึ่งได้แก่ B3, B4, B5, C3, C5, C6, D3, D6, E3, E6, F3, F4 และ F5 ตำแหน่งที่ทำไฮไลท์เป็นสีฟ้านั้นคือตำแหน่งที่ปัญญาประดิษฐ์สามารถเดินได้

โดยที่ตำแหน่ง C3 จะเช็คตำแหน่งอื่นๆ รอบตัวทั้งแปดทิศ ดังนี้

| | | | |
|---|---|---|---|
| C3 ไปด้าน C2 | พบตำแหน่งว่าง | เดินไม่ได้ | |
| C3 ไปด้าน B2 | พบตำแหน่งว่าง | เดินไม่ได้ | |
| C3 ไปด้าน B3 | พบตำแหน่งว่าง | เดินไม่ได้ | |
| C3 ไปด้าน B4 | พบตำแหน่งว่าง | เดินไม่ได้ | |
| C3 ไปด้าน C4 | พบหมากคู่แข่ง | พบตำแหน่งว่าง | เดินไม่ได้ |
| C3 ไปด้าน D4 | พบหมากคู่แข่ง | พบหมากตัวเอง | เดินได้ |
| C3 ไปด้าน D3 | พบตำแหน่งว่าง | เดินไม่ได้ | |
| C3 ไปด้าน D2 | พบตำแหน่งว่าง | เดินไม่ได้ | |

จะพบว่าตำแหน่ง C3 นี้สามารถเดินได้ จึงเป็นตำแหน่งที่ทำเครื่องหมายสีทึบเอาไว้นั่นเอง และในตำแหน่งอื่นๆ นั้นก็ใช้วิธีการเดียวกันนี้ ในการเช็คว่าตำแหน่งเหล่านั้นสามารถเดินได้หรือไม่

### 3.5.3 การคิดของปัญญาประดิษฐ์ (Level 1)




**รูปที่ 3.8** ภาพแสดงหมากเมื่อเริ่มต้นเล่นเกมต่อจากผู้เล่นเดินในตำแหน่ง C4

หลังจากที่ผู้เล่นเดินหมากไปแล้วในตำแหน่ง C4 แล้ว ปัญญาประดิษฐ์จะสามารถเดินหมากได้ในตำแหน่ง สีฟ้า ที่แสดงให้เห็นจากรูป ซึ่งได้แก่ ตำแหน่ง C3 C5 และ E3

ในการที่ ปัญญาประดิษฐ์จะตัดสินใจลงหมากที่ตำแหน่งใดนั้น จะตัดสินใจจากการรวมคะแนนที่คำนวนได้ โดยการรวมคะแนนมีวิธิการคิดดังนี้ คือ

1. ระบบกำหนดคะแนนในการกินหมากคู่แข่งไว้เป็น 50 คะแนน ต่อ หมาก 1 ตัว
2. ระบบกำหนดคะแนนในแต่ละตำแหน่งบนกระดาน หลังจากกินหมากคู่แข่งแล้วเดินลงตำแหน่งนั้น มีคะแนนดังนี้

**ที่มาของตารางคะแนนของปัญญาประดิษฐ์**

ผู้พัฒนาทำการกำหนดคะแนน โดยเริ่มต้นจากตำแหน่งที่ได้เปรียบและเสียเปรียบของกระดาน ในตำแหน่งต่างๆ เช่นตำแหน่งมุมกระดาน ที่มีความได้เปรียบสูงสุด จึงมีการกำหนดคะแนนให้มากที่สุดของกระดาน และตำแหน่งติดมุมกระดาน ในแนวทแยงนั้น จะถือว่าเป็นตำแหน่งที่เมื่อเดินแล้ว จะทำให้มีโอกาสมากที่สุด ที่จะเสียการครอบครองมุมให้กับฝ่ายตรงข้ามได้ จึงมีคะแนนติดลบ และตำแหน่งอื่นๆ จะมีคะแนนมากน้อยลดหลั่นไปตามความได้เปรียบเสียเปรียบ โดยอ้างอิงคะแนนจากตำแหน่งได้เปรียบเสียเปรียบของกระดาน จากรูปที่ 3.1 การกำหนดคะแนนตามหลักการดังกล่าวนี้ กระทำโดยการกำหนดในตารางคะแนนที่ 1 ขึ้นมาจากหลักการข้างต้น จะใช้เป็นตารางคะแนน ของ AI 1 และ AI 2 ส่วนตารางคะแนนที่ 2 และ ตารางคะแนนที่ 3 ของ AI 2 นั้น จะกำหนดขึ้นมาโดยอ้างอิงการลดหลั่นคะแนน จากตารางที่ 1 เป็นหลัก และใน AI 2 จะใช้หลักการ Minimization เข้ามาช่วย หมายถึง ใช้การกินหมากในแต่ละรอบ ในจำนวนน้อยที่สุด เท่าที่จะเป็นไปได้ เข้ามามีส่วนร่วมในการคิดคะแนนด้วย และหลักการ Minimization นั้นจะเดินในตำแหน่งขอบกระดาน ก็ต่อเมื่อตำแหน่งอื่นๆ ในลำดับที่กลางกระดานเข้ามานั้นเดินไม่ได้ และจะบีบ ให้ฝ่ายตรงกันข้ามต้องเดินหมากในตำแหน่ง ติดกับตำแหน่งมุม ทั้ง 4 มุมของกระดาน เพื่อให้ฝ่ายตน ได้ครอบครองมุม เพื่อครองความได้เปรียบของเกม

โดยตารางคะแนนที่นำมาใช้นั้น ได้ผ่านการทดลอง โดยการนำค่าคะแนนในแบบต่างๆ 10 รูปแบบ นำมาทดลอง เพื่อหาคะแนนที่ดี ในการตัดสินใจเดินหมาก โดยคะแนนที่นำมาใช้นี้AI 2 สามารถเอาชนะ AI 1 ได้มากกว่า ร้อยละ60 ผลการทดลองคะแนนในแบบต่างๆ มีดังนี้

**แบบที่ 1**

AI 1 คะแนนกินหมากเป็น 50 คะแนน

AI 2 คะแนนกินหมากครั้งที่ 1 เป็น -10 คะแนน

ตารางคะแนนที่ 1เป็น

{ 1000, -1000, -20, -20, -20, -20, -1000, 1000},
{-1000, -2000, 2, 2, 2, 2, -2000, -1000},
{ -20, 2, 2, 2, 2, 2, 2, -20},
{ -20, 2, 2, 0, 0, 2, 2, -20},
{ -20, 2, 2, 0, 0, 2, 2, -20},
{ -20, 2, 2, 2, 2, 2, 2, -20},
{-1000, -2000, 2, 2, 2, 2, -2000, -1000},
{ 1000, -1000, -20, -20, -20, -20, -1000, 1000}

AI 2 คะแนนกินหมากครั้งที่ 2 เป็น 1 คะแนน

ตารางคะแนนที่ 2 เป็น 0 ทุกตำแหน่ง

AI 2 คะแนนกินหมากครั้งที่ 3 เป็น -1 คะแนน

ตารางคะแนนที่ 3 เป็น 0 ทุกตำแหน่ง

ผลการแข่ง AI 1 vs. AI 2

(20 เกม) [06-00-14] [12-01-07] [11-01-08] [10-00-10] [10-01-09] [09-00-11] [08-00-12]
[11-00-09] [07-00-13] [11-00-09] [11-00-09] [10-00-10] [14-00-06] [12-00-08]
[04-00-16] [10-00-10] [11-00-09] [06-01-13] [11-01-08] [10-00-10] {สรุป 10-4-6}

(50 เกม) [27-00-23] [30-00-20] [26-00-24] [28-00-22] [22-00-28] [21-01-28] [25-00-25]
[23-00-27] [20-00-30] [26-00-24] [29-00-21] [21-00-29] [25-00-25] [22-01-27]
[27-00-23] [22-01-27] [23-00-27] [25-00-25] [25-01-24] [21-00-29] {สรุป 8-3-9}

(100 เกม) [51-00-49] [57-00-43] [52-01-47] [51-00-49] [45-00-55] [42-01-57] [50-00-50]
[42-00-58] [43-00-57] [57-00-43] [52-01-47] [51-00-49] [45-00-55] [42-01-57]
[57-00-43] [52-01-47] [51-00-49] [45-00-55] [42-01-57] [50-00-50] {สรุป 10-2-8}

การอ่านผลการแข่งขันอ่านได้ดังนี้คือ [06-00-14] เป็นผลการแข่งขัน ระหว่างปัญญาประดิษฐ์ระดับที่ 1 และระดับที่ 2 โดยตัวเลขดังกล่าวหมายความว่า จากการแข่งขัน 20 เกม 06 หมายถึง AI 1 ชนะ 6 เกม 00 หมายถึง มีการเสมอกัน 0 เกม และ 14 หมายถึง AI 2 ชนะ 14 เกม

**แบบที่ 2**

AI 1 คะแนนกินหมากเป็น 50 คะแนน

AI 2 คะแนนกินหมากครั้งที่ 1 เป็น 10 คะแนน

ตารางคะแนนที่ 1 เป็น

```
{1000000, -100,  200,  50,  50,  200,  -100,  1000000},
{-100,    -200,  -50,  -10, -10,  -50,  -200,    -100},
{ 200,     -50,  100,   25,  25,  100,   -50,     200},
{  50,     -10,   25,    0,   0,   25,   -10,      50},
{  50,     -10,   25,    0,   0,   25,   -10,      50},
{ 200,     -50,  100,   25,  25,  100,   -50,     200},
{-100,    -200,  -50,  -10, -10,  -50,  -200,    -100},
{1000000, -100,  200,  50,  50,  200,  -100,  1000000}
```

AI 2 คะแนนกินหมากครั้งที่ 2 เป็น -10 คะแนน

ตารางคะแนนที่ 2 เป็น

```
{-100000, 1000, -2000, -500, -500, -2000, 1000, -100000},
{  1000, 2000,   500,  100,  100,  500,  2000,   1000},
{ -2000,  500,  -500, -250, -250, -500,   500,  -2000},
{  -500,  100,  -250,    0,    0, -250,   100,   -500},
{  -500,  100,  -250,    0,    0, -250,   100,   -500},
{ -2000,  500,  -500, -250, -250, -500,   500,  -2000},
{  1000, 2000,   500,  100,  100,  500,  2000,   1000},
{-100000, 1000, -2000, -500, -500, -2000, 1000, -100000}
```

AI 2 คะแนนกินหมากครั้งที่ 3 เป็น 10 คะแนน

ตารางคะแนนที่ 3 เป็น

```
{10000, -100, 200, 50, 50, 200, -100, 10000},
{ -100, -200, -50, -10, -10, -50, -200,  -100},
{  200,  -50,  50,  25,  25,  50,  -50,   200},
{   50,  -10,  25,   0,   0,  25,  -10,    50},
{   50,  -10,  25,   0,   0,  25,  -10,    50},
{  200,  -50,  50,  25,  25,  50,  -50,   200},
{ -100, -200, -50, -10, -10, -50, -200,  -100},
{10000, -100, 200, 50,  50, 200, -100, 10000}
```

ผลการแข่งขัน AI 1 vs AI 2

(20 เกม) [09-00-11] [09-03-08] [04-03-13] [10-01-09] [09-01-10] [08-02-10] [09-05-06]
[10-01-09] [11-02-07] [09-02-09] [12-02-06] [11-03-06] [09-03-08] [06-04-10]
[05-02-13] [12-03-05] [11-01-08] [10-02-08] [12-00-08] [14-00-06] {สรุป 13-1-6}

(50 เกม) [27-00-23] [25-06-19] [26-01-23] [26-00-24] [25-00-25] [24-00-26] [28-00-22]
[25-00-25] [24-00-26] [22-00-28] [25-06-19] [26-01-23] [26-00-24] [25-00-25]
[24-01-25] [26-01-23] [26-00-24] [25-00-25] [24-00-26] [24-00-26] {สรุป 10-4-6}

(100 เกม) [50-01-49] [52-00-48] [49-00-51] [48-00-52] [53-00-47] [50-00-50] [55-00-45]
[48-00-52] [53-00-47] [52-00-48] [49-00-51] [48-00-52] [53-00-47] [50-00-50]
[49-00-51] [48-00-52] [48-00-52] [53-00-47] [50-00-50] [55-00-45] {สรุป 9-3-8}

**แบบที่ 3**

AI 1 คะแนนกินหมากเป็น 50 คะแนน

AI 2 คะแนนกินหมากครั้งที่ 1 เป็น -10 คะแนน

ตารางคะแนนที่ 1 เป็น

{ 200, -600, -400, -400, -400, -400, -600, 200},
{-600, -800, -200, -200, -200, -200, -800, -600},
{-400, -200, 10, 10, 10, 10, -200, -400},
{-400, -200, 10, 0, 0, 10, -200, -400},
{-400, -200, 10, 0, 0, 10, -200, -400},
{-400, -200, 10, 10, 10, 10, -200, -400},
{-600, -800, -200, -200, -200, -200, -800, -600},
{ 200, -600, -400, -400, -400, -400, -600, 200}

AI 2 คะแนนกินหมากครั้งที่ 2 เป็น 1 คะแนน

ตารางคะแนนที่ 2 เป็น

{100, -30, 50, 30, 30, 50, -30, 100},
{-30, -50, -10, -10, -10, -10, -50, -30},
{ 50, -10, 10, 2, 2, 10, -10, 50},
{ 30, -10, 2, 0, 0, 2, -10, 30},
{ 30, -10, 2, 0, 0, 2, -10, 30},
{ 50, -10, 10, 2, 2, 10, -10, 50},
{-30, -50, -10, -10, -10, -10, -50, -30},
{100, -30, 50, 30, 30, 50, -30, 100}

AI 2 คะแนนกินหมากครั้งที่ 3 เป็น -1 คะแนน

ตารางคะแนนที่ 3 เป็น

{ 20, -60, -40, -40, -40, -40, -60, 20},
{-60, -80, -20, -20, -20, -20, -80, -60},
{-40, -20, -1, -1, -1, -1, -20, -40},
{-40, -20, -1, 0, 0, -1, -20, -40},
{-40, -20, -1, 0, 0, -1, -20, -40},
{-40, -20, -1, -1, -1, -1, -20, -40},
{-60, -80, -20, -20, -20, -20, -80, -60},
{ 20, -60, -40, -40, -40, -40, -60, 20}

ผลการแข่งขัน AI 1 vs AI 2

(20 เกม) [09-00-11] [13-00-07] [03-00-17] [09-00-11] [07-00-13] [10-00-10] [10-00-10]
[12-00-08] [08-00-12] [08-00-12] [12-00-08] [09-00-11] [10-00-10] [10-00-10]
[14-00-06] [09-00-11] [10-00-10] [10-00-10] [07-00-13] [09-00-11] {สรุป 4-6-10}

(50 เกม) [22-00-28] [27-03-20] [26-00-24] [22-01-27] [28-02-20] [25-00-25] [26-00-24]
[27-00-23] [19-00-31] [24-02-24] [23-06-21] [26-00-24] [22-01-27] [26-00-24]
[22-01-27] [28-02-20] [25-00-25] [26-00-24] [19-00-31] [24-02-24] {สรุป 10-4-6}

(100 เกม) [55-00-45] [43-01-56] [41-00-59] [53-00-47] [54-00-46] [41-00-59] [44-01-55]
[50-00-50] [48-00-52] [54-00-46] [55-00-45] [48-00-52] [53-00-47] [54-00-46]
[55-00-45] [48-00-52] [53-00-47 ] [54-00-46] [41-00-59] [44-01-55] {สรุป 10-1-9}

**แบบที่ 4**

AI 1 คะแนนกินหมากเป็น 50 คะแนน

AI 2 คะแนนกินหมากครั้งที่ 1 เป็น -10 คะแนน

ตารางคะแนนที่ 1 เป็น

{ 200, -600, -400, -400, -400, -400, -600, 200},
{-600, -800, -200, -200, -200, -200, -800, -600},
{-400, -200, 10, 10, 10, 10, -200, -400},
{-400, -200, 10, 0, 0, 10, -200, -400},
{-400, -200, 10, 0, 0, 10, -200, -400},
{-400, -200, 10, 10, 10, 10, -200, -400},
{-600, -800, -200, -200, -200, -200, -800, -600},
{ 200, -600, -400, -400, -400, -400, -600, 200}

AI 2 คะแนนกินหมากครั้งที่ 2 เป็น 1 คะแนน

ตารางคะแนนที่ 2 เป็น

{-100, 30, 30, 30, 30, 30, 30, -100},
{ 30, 50, -10, -10, -10, -10, 50, 30},
{ 30, -10, 2, 2, 2, 2, -10, 30},
{ 30, -10, 2, 0, 0, 2, -10, 30},
{ 30, -10, 2, 0, 0, 2, -10, 30},
{ 30, -10, 2, 2, 2, 2, -10, 30},
{ 30, 50, -10, -10, -10, -10, 50, 30},
{-100, 30, 30, 30, 30, 30, 30, -100}

AI 2 คะแนนกินหมากครั้งที่ 3 เป็น -1 คะแนน

ตารางคะแนนที่ 3 เป็น

{ 20, -60, -40, -40, -40, -40, -60, 20},
{-60, -80, -20, -20, -20, -20, -80, -60},
{-40, -20, -1, -1, -1, -1, -20, -40},
{-40, -20, -1, 0, 0, -1, -20, -40},
{-40, -20, -1, 0, 0, -1, -20, -40},
{-40, -20, -1, -1, -1, -1, -20, -40},
{-60, -80, -20, -20, -20, -20, -80, -60},
{ 20, -60, -40, -40, -40, -40, -60, 20}

ผลการแข่งขัน AI 1 vs AI 2

(20 เกม) [13-07-00] [10-01-09] [11-00-09] [06-00-14] [11-00-09] [13-00-07] [08-00-12]
[10-00-10] [09-00-11] [11-00-09] [09-00-11] [10-01-09] [11-00-09] [09-00-11]
[11-00-9] [08-00-12] [10-00-10] [13-00-7] [06-00-14] [07-01-12] {สรุป 10-2-8}

(50 เกม) [27-04-19] [27-02-21] [23-00-27] [24-00-26] [20-00-30] [26-03-21] [23-04-23]
[27-02-21] [23-00-27] [24-00-26] [27-00-23] [26-03-21] [26-00-24] [22-00-28]
[25-00-25] [22-00-28] [24-01-25] [27-03-20] [21-00-29] [25-05-20] {สรุป 9-2-9}

(100 เกม) [50-00-50] [48-00-52] [54-00-46] [55-00-45] [48-00-52] [53-00-47] [54-00-46]
[42-00-58] [43-00-57] [57-00-43] [52-01-47] [51-00-49] [45-00-55] [42-01-57]
[44-01-55] [44-00-56] [48-00-52] [46-00-54] [66-00-34] [55-00-45] {สรุป 9-1-10}

**แบบที่ 5**

AI 1 คะแนนกินหมากเป็น 50 คะแนน

AI 2 คะแนนกินหมากครั้งที่ 1 เป็น -10 คะแนน

ตารางคะแนนที่ 1 เป็น

{ 200, -600, -400, -400, -400, -400, -600, 200},
{-600, -800, -200, -200, -200, -200, -800, -600},
{-400, -200, 10, 10, 10, 10, -200, -400},
{-400, -200, 10, 0, 0, 10, -200, -400},
{-400, -200, 10, 0, 0, 10, -200, -400},
{-400, -200, 10, 10, 10, 10, -200, -400},
{-600, -800, -200, -200, -200, -200, -800, -600},
{ 200, -600, -400, -400, -400, -400, -600, 200}

AI 2 คะแนนกินหมากครั้งที่ 2 เป็น 1 คะแนน

ตารางคะแนนที่ 2 เป็น

{-100, 30, 30, 30, 30, 30, 30, -100},
{ 30, 90, -30, -30, -30, -30, 90, 30},
{ 30, -30, 2, 2, 2, 2, -30, 30},
{ 30, -30, 2, 0, 0, 2, -30, 30},
{ 30, -30, 2, 0, 0, 2, -30, 30},
{ 30, -30, 2, 2, 2, 2, -30, 30},
{ 30, 90, -30, -30, -30, -30, 90, 30},
{-100, 30, 30, 30, 30, 30, 30, -100}

AI 2 คะแนนกินหมากครั้งที่ 3 เป็น -1 คะแนน

ตารางคะแนนที่ 3 เป็น

{ 20, -60, -40, -40, -40, -40, -60, 20},
{-60, -80, -20, -20, -20, -20, -80, -60},
{-40, -20, -1, -1, -1, -1, -20, -40},
{-40, -20, -1, 0, 0, -1, -20, -40},
{-40, -20, -1, 0, 0, -1, -20, -40},
{-40, -20, -1, -1, -1, -1, -20, -40},
{-60, -80, -20, -20, -20, -20, -80, -60},
{ 20, -60, -40, -40, -40, -40, -60, 20}

ผลการแข่งขัน AI 1 vs AI 2

(20 เกม) [06-00-14] [12-00-08] [09-00-11] [08-00-12] [07-00-13] [07-00-13] [10-00-10]
[10-00-10] [11-00-09] [14-00-06] [14-00-06] [08-00-12] [09-00-11] [10-00-10]
[08-00-12] [11-00-09] [14-00-06] [10-00-10] [07-00-13] [08-00-12] {สรุป 6-4-10}

(50 เกม) [18-00-32] [27-00-23] [22-00-28] [24-00-26] [25-00-25] [32-00-18] [23-00-27]
[23-00-27] [28-00-22] [25-00-25] [25-00-25] [25-00-25] [22-00-28] [25-00-25]
[24-00-26] [20-00-30] [22-00-28] [24-01-25] [27-00-23] [19-00-31] {สรุป 4-5-11}

(100 เกม) [39-00-61] [50-00-50] [57-00-53] [51-01-48] [48-01-51] [47-00-53] [53-00-47]
[41-00-59] [42-00-58] [55-00-45] [54-00-46] [50-00-50] [52-00-48] [45-00-55]
[44-01-55] [44-00-56] [48-00-52] [46-00-54] [66-00-34] [55-00-45] {สรุป 6-3-11}

**แบบที่ 6**

AI 1 คะแนนกินหมากเป็น 50 คะแนน

AI 2 คะแนนกินหมากครั้งที่ 1 เป็น -10 คะแนน

ตารางคะแนนที่ 1 เป็น

{ 200, -600, -400, -400, -400, -400, -600, 200},
{-600, -800, -200, -200, -200, -200, -800, -600},
{-400, -200, 10, 10, 10, 10, -200, -400},
{-400, -200, 10, 0, 0, 10, -200, -400},
{-400, -200, 10, 0, 0, 10, -200, -400},
{-400, -200, 10, 10, 10, 10, -200, -400},
{-600, -800, -200, -200, -200, -200, -800, -600},
{ 200, -600, -400, -400, -400, -400, -600, 200}

AI 2 คะแนนกินหมากครั้งที่ 2 เป็น 1 คะแนน

ตารางคะแนนที่ 2 เป็น

{-200, 90, 30, 30, 30, 30, 90, -200},
{ 90, 200, -30, -30, -30, -30, 200, 90},
{ 30, -30, 2, 2, 2, 2, -30, 30},
{ 30, -30, 2, 0, 0, 2, -30, 30},
{ 30, -30, 2, 0, 0, 2, -30, 30},
{ 30, -30, 2, 2, 2, 2, -30, 30},
{ 90, 200, -30, -30, -30, -30, 200, 90},
{-200, 90, 30, 30, 30, 30, 90, -200}

AI 2 คะแนนกินหมากครั้งที่ 3 เป็น -1 คะแนน

ตารางคะแนนที่ 3 เป็น

{ 20, -60, -40, -40, -40, -40, -60, 20},
{-60, -80, -20, -20, -20, -20, -80, -60},
{-40, -20, -1, -1, -1, -1, -20, -40},
{-40, -20, -1, 0, 0, -1, -20, -40},
{-40, -20, -1, 0, 0, -1, -20, -40},
{-40, -20, -1, -1, -1, -1, -20, -40},
{-60, -80, -20, -20, -20, -20, -80, -60},
{ 20, -60, -40, -40, -40, -40, -60, 20}

ผลการแข่งขัน AI 1 vs AI 2

(20 เกม) [08-00-12] [09-00-11] [08-00-12] [06-00-14] [11-00-09] [10-00-10] [08-00-12]
[06-00-14] [12-00-08] [09-00-11] [11-00-09] [11-01-08] [12-00-08] [08-00-12]
[06-00-14] [11-00-09] [12-00-08] [12-00-08] [10-00-10] [08-00-12] {สรุป 8-2-10}

(50 เกม) [27-00-23] [21-00-29] [23-00-27] [25-00-25] [22-01-27] [30-00-20] [23-00-27]
[24-00-26] [20-00-30] [26-00-24] [25-01-24] [27-00-23] [29-00-21] [27-00-23]
[26-00-24] [24-00-26] [20-00-30] [25-01-24] [27-00-23] [29-00-21] {สรุป 11-1-8}

(100 เกม) [51-00-49] [57-00-43] [51-00-49] [52-01-47] [42-01-57] [47-01-52] [45-00-55]
[56-00-44] [44-00-56] [51-00-49] [57-00-43] [51-00-49] [52-01-47] [42-01-57]
[52-01-47] [42-01-57] [47-01-52] [45-00-55] [56-00-44] [44-00-56] {สรุป 11-0-9}

**แบบที่ 7**

AI 1 คะแนนกินหมากเป็น 50 คะแนน

AI 2 คะแนนกินหมากครั้งที่ 1 เป็น -10 คะแนน

ตารางคะแนนที่ 1 เป็น

{ 200, -600, -400, -400, -400, -400, -600, 200},
{-600, -800, -200, -200, -200, -200, -800, -600},
{-400, -200, 10, 10, 10, 10, -200, -400},
{-400, -200, 10, 0, 0, 10, -200, -400},
{-400, -200, 10, 0, 0, 10, -200, -400},
{-400, -200, 10, 10, 10, 10, -200, -400},
{-600, -800, -200, -200, -200, -200, -800, -600},
{ 200, -600, -400, -400, -400, -400, -600, 200}

AI 2 คะแนนกินหมากครั้งที่ 2 เป็น 1 คะแนน

ตารางคะแนนที่ 2 เป็น

{-200, 70, -70, 50, 50, -70, 70, -200},
{ 70, 100, 50, 30, 30, 50, 100, 70},
{ -70, 50, -30, -2, -2, -30, 50, -70},
{ 50, 30, -2, 0, 0, -2, 30, 50},
{ 50, 30, -2, 0, 0, -2, 30, 50},
{ -70, 50, -30, -2, -2, -30, 50, -70},
{ 70, 100, 50, 30, 30, 50, 100, 70},
{-200, 70, -70, 50, 50, -70, 70, -200}

AI 2 คะแนนกินหมากครั้งที่ 3 เป็น -1 คะแนน

ตารางคะแนนที่ 3 เป็น

{ 20, -60, -40, -40, -40, -40, -60, 20},
{-60, -80, -20, -20, -20, -20, -80, -60},
{-40, -20, -1, -1, -1, -1, -20, -40},
{-40, -20, -1, 0, 0, -1, -20, -40},
{-40, -20, -1, 0, 0, -1, -20, -40},
{-40, -20, -1, -1, -1, -1, -20, -40},
{-60, -80, -20, -20, -20, -20, -80, -60},
{ 20, -60, -40, -40, -40, -40, -60, 20}

ผลการแข่งขัน AI 1 vs AI 2

(20 เกม) [06-03-11] [07-03-10] [10-03-07] [07-03-10] [12-00-08] [11-00-09] [07-01-12]
[08-01-11] [12-00-08] [09-00-11] [10-03-07] [08-01-11] [12-01-07] [10-00-10]
[07-03-10] [10-03-07] [07-03-10] [12-00-08] [11-00-09] [07-01-12] {สรุป 9-1-10}

(50 เกม) [19-03-28] [28-03-19] [20-00-30] [26-03-21] [24-00-26] [28-00-22] [27-03-20]
[20-04-26] [24-02-24] [25-05-20] [23-06-21] [27-01-22] [28-00-22] [19-03-28]
[26-03-21] [27-00-23] [26-00-24] [21-00-29] [28-02-20] [27-00-23] {สรุป 13-1-6}

(100 เกม) [42-08-52] [56-06-38] [52-05-43] [38-10-52] [50-05-45] [49-02-49] [51-03-46]
[50-03-47] [49-04-47] [49-05-46] [46-00-54] [49-00-51] [42-08-52] [39-09-52]
[38-10-52] [50-05-45] [49-02-49] [51-03-46] [38-10-52] [42-08-52] {สรุป 9-2-9}

**แบบที่ 8**

AI 1 คะแนนกินหมากเป็น 50 คะแนน

AI 2 คะแนนกินหมากครั้งที่ 1 เป็น -10 คะแนน

ตารางคะแนนที่ 1 เป็น

{ 200, -600, -400, -400, -400, -400, -600, 200},
{-600, -800, -200, -200, -200, -200, -800, -600},
{-400, -200, 10, 10, 10, 10, -200, -400},
{-400, -200, 10, 0, 0, 10, -200, -400},
{-400, -200, 10, 0, 0, 10, -200, -400},
{-400, -200, 10, 10, 10, 10, -200, -400},
{-600, -800, -200, -200, -200, -200, -800, -600},
{ 200, -600, -400, -400, -400, -400, -600, 200}

AI 2 คะแนนกินหมากครั้งที่ 2 เป็น 1 คะแนน

ตารางคะแนนที่ 3 เป็น

{ 200, 50, 100, 50, 50, 100, 50, 200},
{ 50, -150, -50, -20, -20, -50, -150, 50},
{ 100, -50, 10, 1, 1, 10, -50, 100},
{ 50, -20, 1, 0, 0, 1, -20, 50},
{ 50, -20, 1, 0, 0, 1, -20, 50},
{ 100, -50, 10, 1, 1, 10, -50, 100},
{ 50, -150, -50, -20, -20, -50, -150, 50},
{ 200, 50, 100, 50, 50, 100, 50, 200}

AI 2 คะแนนกินหมากครั้งที่ 3 เป็น -1 คะแนน

ตารางคะแนนที่ 3 เป็น

{ 20, -60, -40, -40, -40, -40, -60, 20},
{-60, -80, -20, -20, -20, -20, -80, -60},
{-40, -20, -1, -1, -1, -1, -20, -40},
{-40, -20, -1, 0, 0, -1, -20, -40},
{-40, -20, -1, 0, 0, -1, -20, -40},
{-40, -20, -1, -1, -1, -1, -20, -40},
{-60, -80, -20, -20, -20, -20, -80, -60},
{ 20, -60, -40, -40, -40, -40, -60, 20}

ผลการแข่งขัน AI 1 vs AI 2

(20 เกม) [08-00-12] [10-00-10] [11-00-09] [09-01-10] [13-00-07] [10-00-10] [11-00-09] [10-00-10] [9-00-11] [10-00-10] [10-00-10] [09-01-10] [13-00-07] [10-00-10] [10-00-10] [11-00-09] [09-01-10] [13-00-07] [10-00-10] [11-00-09] {สรุป 7-8-5}

(50 เกม) [30-00-20] [21-00-29] [27-00-23] [26-00-24] [24-00-26] [22-01-27] [14-00-36] [30-01-19] [21-00-29] [13-00-37] [30-00-20] [21-00-29] [27-00-23] [26-00-24] [24-00-26] [22-01-27] [14-00-36] [30-01-19] [21-00-29] [13-00-37] {สรุป 8-0-12}

(100 เกม) [55-00-45] [48-00-52] [43-01-56] [52-00-48] [41-00-59] [44-01-55] [53-00-47] [53-01-46] [53-00-47] [53-01-46] [53-00-47] [54-00-46] [41-00-59] [41-00-59] [43-01-56] [53-01-46] [53-00-47] [53-01-46] [53-00-47] [54-00-46] {สรุป 13-0-7}

**แบบที่ 9**

AI 1 คะแนนกินหมากเป็น 50 คะแนน

AI 2 คะแนนกินหมากครั้งที่ 1 เป็น -10 คะแนน

ตารางคะแนนที่ 1 เป็น

{ 200, -600, -400, -400, -400, -400, -600, 200},
{-600, -800, -200, -200, -200, -200, -800, -600},
{-400, -200, 10, 10, 10, 10, -200, -400},
{-400, -200, 10, 0, 0, 10, -200, -400},
{-400, -200, 10, 0, 0, 10, -200, -400},
{-400, -200, 10, 10, 10, 10, -200, -400},
{-600, -800, -200, -200, -200, -200, -800, -600},
{ 200, -600, -400, -400, -400, -400, -600, 200}

AI 2 คะแนนกินหมากครั้งที่ 2 เป็น 1 คะแนน

ตารางคะแนนที่ 2 เป็น

{-1000, 100, -200, -50, -50, -200, 100, -1000},
{ 100, 200, 50, 10, 10, 50, 200, 100},
{ -200, 50, -100, -25, -25, -100, 50, -200},
{ -50, 10, -25, 0, 0, -25, 10, -50},
{ -50, 10, -25, 0, 0, -25, 10, -50},
{ -200, 50, -100, -25, -25, -100, 50, -200},
{ 100, 200, 50, 10, 10, 50, 200, 100},
{-1000, 100, -200, -50, -50, -200, 100, -1000}

AI 2 คะแนนกินหมากครั้งที่ 3 เป็น -1 คะแนน

ตารางคะแนนที่ 3 เป็น

{ 20, -60, -40, -40, -40, -40, -60, 20},
{-60, -80, -20, -20, -20, -20, -80, -60},
{-40, -20, -1, -1, -1, -1, -20, -40},
{-40, -20, -1, 0, 0, -1, -20, -40},
{-40, -20, -1, 0, 0, -1, -20, -40},
{-40, -20, -1, -1, -1, -1, -20, -40},
{-60, -80, -20, -20, -20, -20, -80, -60},
{ 20, -60, -40, -40, -40, -40, -60, 20}

ผลการแข่งขัน AI 1 vs AI 2

(20 เกม) [09-03-08] [09-02-09] [11-02-07] [09-01-10] [10-01-09] [11-01-08] [08-02-10]
[05-04-11] [08-03-09] [08-03-09] [09-03-08] [09-02-09] [11-02-07] [09-01-10]
[11-01-08] 0[8-02-10] [05-04-11] [08-03-09] [08-03-09] [10-01-09] {สรุป 8-2-10}

(50 เกม) [27-04-19] [27-02-21] [23-04-23] [23-04-23] [13-02-35] [19-06-25] [26-03-21]
[24-02-24] [26-04-20] [13-07-30] [27-04-19] [27-02-21] [23-04-23] [23-04-23]
[13-02-35] [19-06-25] [26-03-21] [24-02-24] [26-04-20] [13-07-30] {สรุป 8-6-6}

(100 เกม) [56-07-37] [46-08-46] [52-07-41] [41-09-50] [45-11-44] [50-06-44] [37-09-54]
[52-08-40] [41-06-53] [42-08-50] [56-07-37] [46-08-46] [52-07-41] [41-09-50]
[45-11-44] [50-06-44] [37-09-54] [52-08-40] [41-06-53] [42-08-50] {สรุป 10-2-8}

**แบบที่ 10**

AI 1 คะแนนกินหมากเป็น 50 คะแนน

AI 2 คะแนนกินหมากครั้งที่ 1 เป็น -10 คะแนน

ตารางคะแนนที่ 1

{ 200, -600, -400, -400, -400, -400, -600, 200},
{-600, -800, -200, -200, -200, -200, -800, -600},
{-400, -200, 10, 10, 10, 10, -200, -400},
{-400, -200, 10, 0, 0, 10, -200, -400},
{-400, -200, 10, 0, 0, 10, -200, -400},
{-400, -200, 10, 10, 10, 10, -200, -400},
{-600, -800, -200, -200, -200, -200, -800, -600},
{ 200, -600, -400, -400, -400, -400, -600, 200}

AI 2 คะแนนกินหมากครั้งที่ 2 เป็น -50 คะแนน

ตารางคะแนนที่ 2

{-1000, 100, -200, -50, -50, -200, 100, -1000},
{ 100, 200, 50, 10, 10, 50, 200, 100},
{ -200, 50, -100, -25, -25, -100, 50, -200},
{ -50, 10, -25, 0, 0, -25, 10, -50},
{ -50, 10, -25, 0, 0, -25, 10, -50},
{ -200, 50, -100, -25, -25, -100, 50, -200},
{ 100, 200, 50, 10, 10, 50, 200, 100},
{-1000, 100, -200, -50, -50, -200, 100, -1000}

AI 3 คะแนนกินหมากครั้งที่ 3 เป็น -1 คะแนน

ตารางคะแนนที่ 3

{ 20, -60, -40, -40, -40, -40, -60, 20},
{-60, -80, -20, -20, -20, -20, -80, -60},
{-40, -20, -1, -1, -1, -1, -20, -40},
{-40, -20, -1, 0, 0, -1, -20, -40},
{-40, -20, -1, 0, 0, -1, -20, -40},
{-40, -20, -1, -1, -1, -1, -20, -40},
{-60, -80, -20, -20, -20, -20, -80, -60},
{ 20, -60, -40, -40, -40, -40, -60, 20}

ผลการแข่งขัน AI 1 vs AI 2

(20 เกม) [11-00-09] [10-00-10] [09-00-11] [09-00-11] [14-00-06] [12-00-08] [09-01-10]
[10-00-10] [11-02-07] [06-00-14] [11-00-09] [10-00-10] [09-00-11] [09-00-11]
[14-00-06] [12-00-08] [09-01-10] [10-00-10] [11-02-07] [06-00-14] {สรุป 8-4-8}

(50 เกม) [26-01-23] [26-00-24] [25-00-25] [26-01-23] [24-00-26] [26-01-23] [29-01-20]
[23-02-25] [27-00-23] [21-03-26] [26-01-23] [26-00-24] [25-00-25] [26-01-23]
[24-00-26] [26-01-23] [29-01-20] [23-02-25] [27-00-23] [21-03-26] {สรุป 12-2-6}

(100 เกม) [47-00-53] [44-04-52] [48-00-52] [53-00-47] [52-00-48] [52-04-44] [55-00-45]
[52-00-48] [47-00-53] [54-00-46] [44-01-55] [53-00-47] [41-00-59] [47-00-53]
[44-04-52] [48-00-52] [53-00-47] [52-00-48] [52-04-44] [55-00-45] {สรุป 11-0-9}

ผลการทดลองคิดเป็นร้อยละ ดังนี้

แบบที่ 1 AI 1 ชนะร้อยละ 47, เสมอกันร้อยละ 15 และ AI 2 ชนะร้อยละ 38

แบบที่ 2 AI 1 ชนะร้อยละ 53, เสมอกันร้อยละ 13 และ AI 2 ชนะร้อยละ 34

แบบที่ 3 AI 1 ชนะร้อยละ 40, เสมอกันร้อยละ 18 และ AI 2 ชนะร้อยละ 42

แบบที่ 4 AI 1 ชนะร้อยละ 47, เสมอกันร้อยละ 08 และ AI 2 ชนะร้อยละ 45

แบบที่ 5 AI 1 ชนะร้อยละ 27, เสมอกันร้อยละ 20 และ AI 2 ชนะร้อยละ 53

แบบที่ 6 AI 1 ชนะร้อยละ 50, เสมอกันร้อยละ 05 และ AI 2 ชนะร้อยละ 45

แบบที่ 7 AI 1 ชนะร้อยละ 51, เสมอกันร้อยละ 07 และ AI 2 ชนะร้อยละ 42

แบบที่ 8 AI 1 ชนะร้อยละ 47, เสมอกันร้อยละ 13 และ AI 2 ชนะร้อยละ 40

แบบที่ 9 AI 1 ชนะร้อยละ 43, เสมอกันร้อยละ 17 และ AI 2 ชนะร้อยละ 40

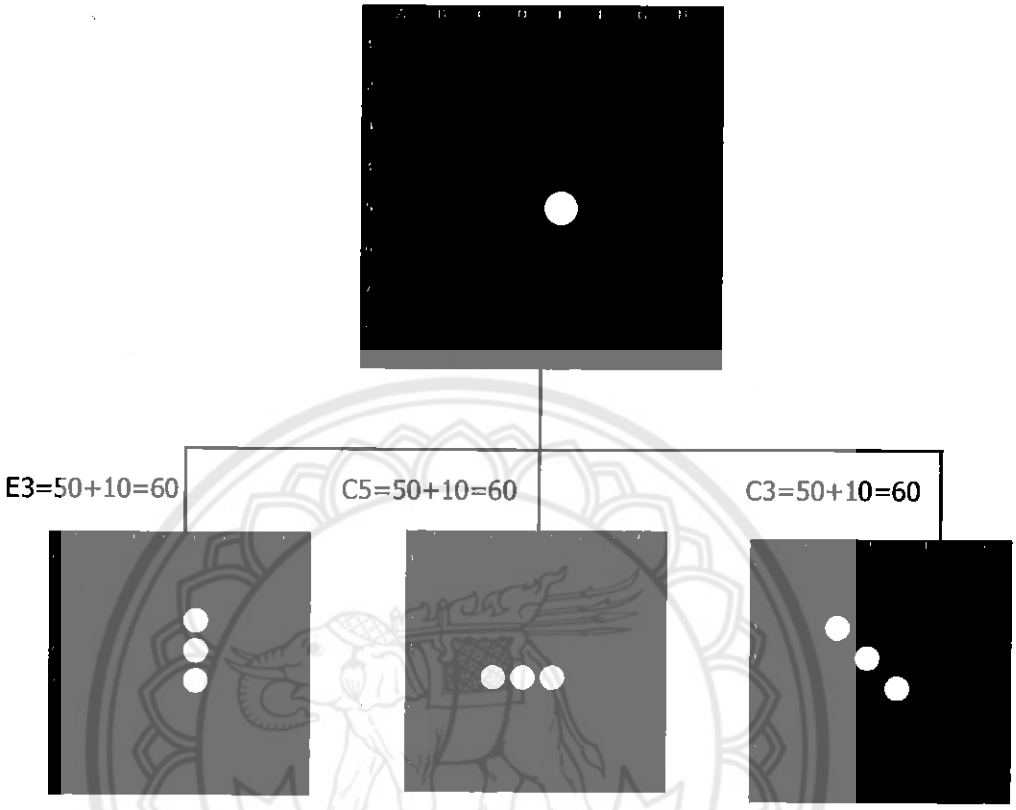
แบบที่ 10 AI 1 ชนะร้อยละ 51, เสมอกันร้อยละ 10 และ AI 2 ชนะร้อยละ 39

จากผลการทดลอง จะเห็นได้ว่า การทดลองในรูปแบบที่ 5 นั้นได้ผลการทดลอง ที่ปัญญาประดิษฐ์ระดับที่ 2 สามารถชนะได้ร้อยละ 53 ปัญญาประดิษฐ์ระดับที่ 1 สามารถชนะได้ ร้อยละ 27 ดังนั้นจึงถือว่าการกำหนดตารางคะแนน ในแบบที่ 5 เป็นคะแนนที่ดีในการทดลองนี้ จึงได้นำหลักการในรูปแบบที่ 5 มาเป็น Algorithm หลักในการกำหนดตารางการคิดคะแนนของปัญญาประดิษฐ์

**ตารางที่** 3.1 ตารางแสดงคะแนนในแต่ละตำแหน่งบนกระดานที่วางหมากหลังจากกินหมากคู่แข่ง

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 200 | -600 | -400 | -400 | -400 | -400 | -100 | 200 |
| -600 | -800 | -200 | -200 | -200 | -200 | -800 | -100 |
| -400 | -200 | 10 | 10 | 10 | 10 | -200 | -400 |
| -400 | -200 | 10 | 0 | 0 | 10 | -200 | -400 |
| -400 | -200 | 10 | 0 | 0 | 10 | -200 | -400 |
| -400 | -200 | 10 | 10 | 10 | 10 | -200 | -400 |
| -600 | -800 | -200 | -200 | -200 | -200 | -800 | -600 |
| 200 | -600 | -400 | -400 | -400 | -400 | -600 | 200 |

ดังนั้น แต่ละตำแหน่งคิดคะแนนได้ดังนี้ คือ



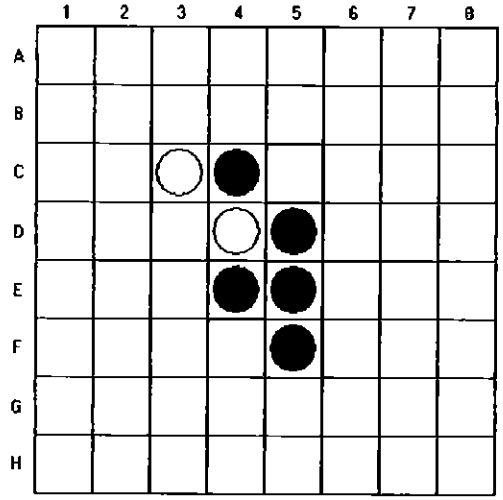

รูปที่ 3.9 ภาพแสดง แผนภาพ Tree การคิดคะแนนของปัญญาประดิษฐ์ระดับที่ 1

C3 ได้จากการกินหมากคู่แข่ง 50 P. ตำแหน่งที่ลง 10 P. รวมเป็น 60 คะแนน

C5 ได้จากการกินหมากคู่แข่ง 50 P. ตำแหน่งที่ลง 10 P. รวมเป็น 60 คะแนน

E3 ได้จากการกินหมากคู่แข่ง 50 P. ตำแหน่งที่ลง 10 P. รวมเป็น 60 คะแนน

เมื่อในแต่ละตำแหน่งที่สามารถเดินหมากได้ มีคะแนนรวมครบหมดแล้ว ปัญญาประดิษฐ์จะเลือกเดินในตำแหน่งที่ได้คะแนนรวมมากที่สุดนั่นเอง และในกรณีที่ตำแหน่งที่ได้คะแนนรวมมากที่สุดมีมากกว่าหนึ่งตำแหน่ง ปัญญาประดิษฐ์จะทำการสุ่มเลือกเดินในตำแหน่ง C3, C5 และ E3




**รูปที่ 3.10** ภาพแสดงเมื่อปัญญาประดิษฐ์เดินในตำแหน่ง C3 และคู่แข่งเดินในตำแหน่ง F5

หลักจากที่ผู้เล่นเดินในตำแหน่ง F5 แล้ว ปัญญาประดิษฐ์จะเดินได้ในตำแหน่งเหล่านี้ คือ B4, C5, D6, F4 และ F6

คิดคะแนนได้ดังนี้ คือ

B4 ได้จากการกินหมากคู่แข่ง 50 P. ตำแหน่งที่ลง -200 P. รวมเป็น -150 คะแนน

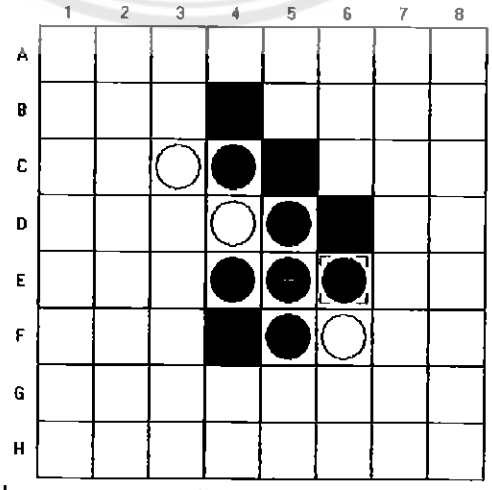
C5 ได้จากการกินหมากคู่แข่ง 50 P. ตำแหน่งที่ลง 10 P. รวมเป็น 60 คะแนน

D6 ได้จากการกินหมากคู่แข่ง 50 P. ตำแหน่งที่ลง 10 P. รวมเป็น 60 คะแนน

F4 ได้จากการกินหมากคู่แข่ง 50 P. ตำแหน่งที่ลง 10 P. รวมเป็น 60 คะแนน

F6 ได้จากการกินหมากคู่แข่ง 50 P. ตำแหน่งที่ลง 10 P. รวมเป็น 60 คะแนน

คะแนนรวมมากที่สุดคือ ตำแหน่ง C5, D6, F4 และ F6 ดังนั้นปัญญาประดิษฐ์จะสุ่มเลือกเดินในตำแหน่งดังกล่าว




**รูปที่ 3.11** ภาพแสดงเมื่อปัญญาประดิษฐ์เดินในตำแหน่ง F6 และคู่แข่งเดินในตำแหน่ง E6

หลักจากที่ผู้เล่นเดินในตำแหน่ง E6 แล้ว ปัญญาประดิษฐ์จะเดินได้ในตำแหน่งเหล่านี้ คือ B4, C5, D6 และ F4

คิดคะแนนได้ดังนี้ คือ

B4 ได้จากการกินหมากคู่แข่ง 50 P. ตำแหน่งที่ลง -200 P. รวมเป็น -150 คะแนน

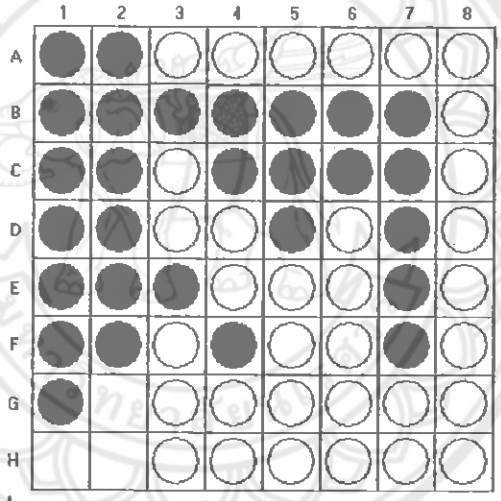
C5 ได้จากการกินหมากคู่แข่ง 50 P. ตำแหน่งที่ลง 10 P. รวมเป็น 60 คะแนน

D6 ได้จากการกินหมากคู่แข่ง 100 P. ตำแหน่งที่ลง 10 P. รวมเป็น 110 คะแนน

F4 ได้จากการกินหมากคู่แข่ง 100 P. ตำแหน่งที่ลง 10 P. รวมเป็น 110 คะแนน

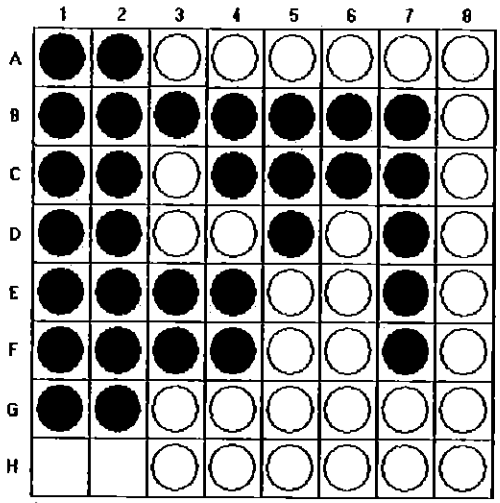
คะแนนรวมมากที่สุดคือ ตำแหน่ง D6 และ F4 ดังนั้นปัญญาประดิษฐ์ สุ่มเลือกเดินในตำแหน่งดังกล่าว

กระบวนการคิดดังที่กล่าวมา จะเป็นกระบวนการหลักที่ปัญญาประดิษฐ์ ใช้คิดคำนวณตัดสินใจเดินหมาก ในตำแหน่งต่างๆ ไปจนกระทั่งเกมจบ




**รูปที่ 3.12** ภาพแสดงเมื่อปัญญาประดิษฐ์เดินในตำแหน่ง B2 และคู่แข่งเดินในตำแหน่ง A2

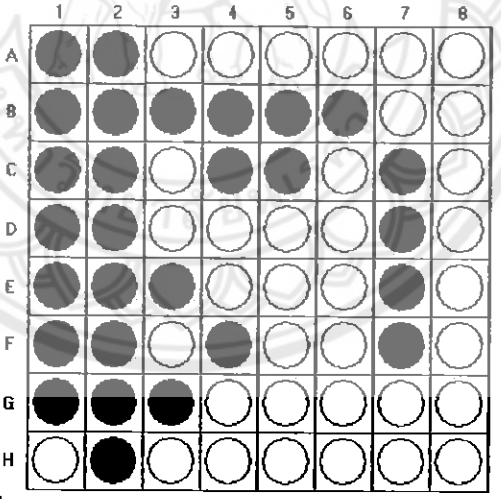
หลักจากที่ผู้เล่นเดินในตำแหน่ง A2 แล้ว ปัญญาประดิษฐ์ไม่มีตำแหน่งที่สามารถเดินได้ จึงต้องผ่านหนึ่งรอบเพื่อให้คู่แข่งเดินต่อ หนึ่งรอบ




รูปที่ 3.13 ภาพแสดงเมื่อปัญญาประดิษฐ์ผ่านหนึ่งรอบ และคู่แข่งเดินในตำแหน่ง G2

หลักจากที่ผู้เล่นเดินในตำแหน่ง G2 แล้ว ปัญญาประดิษฐ์จะเดินได้ในตำแหน่งเหล่านี้ คือ H1 คิดคะแนนได้ดังนี้ คือ

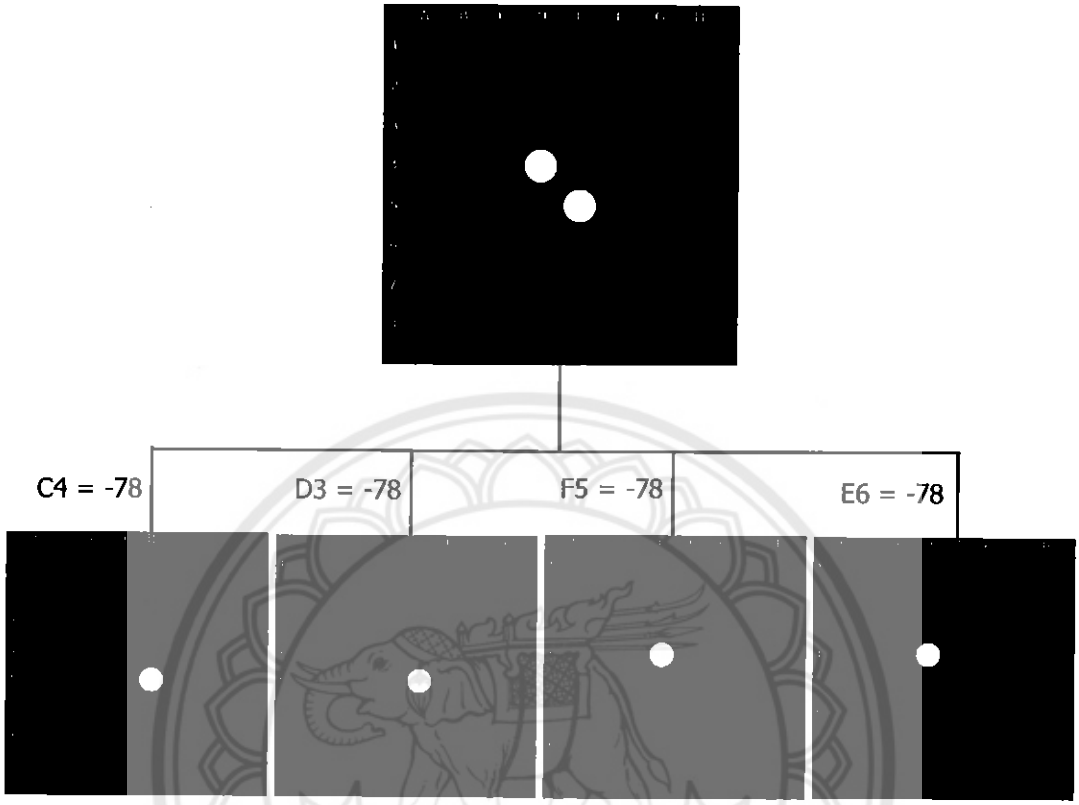
H1 ได้จากการกินหมากคู่แข่ง 300 P. ตำแหน่งที่ลง 200 P. รวมเป็น 500 คะแนน คะแนนรวมมากที่สุดคือ ตำแหน่ง H1 ดังนั้น ปัญญาประดิษฐ์จะเลือกเดินในตำแหน่งดังกล่าว



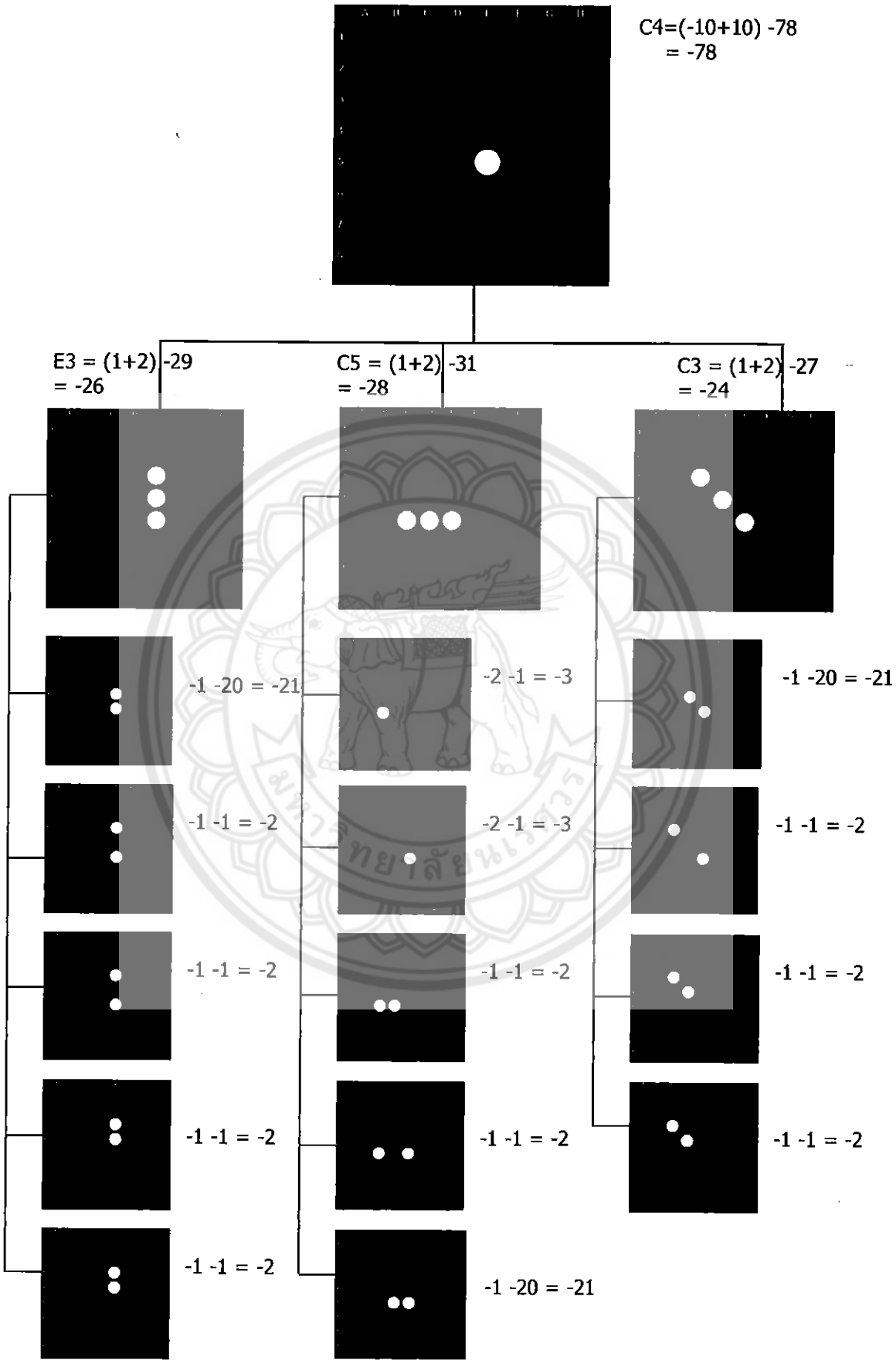

**รูปที่ 3.14** ภาพแสดงเมื่อปัญญาประดิษฐ์เดินในตำแหน่ง H1 และคู่แข่งเดินในตำแหน่ง H2

หลักจากที่ผู้เล่นเดินในตำแหน่ง H2 แล้ว ไม่มีตำแหน่งว่างเหลือบนกระดานแล้ว ถือว่าจบเกมโดย เมื่อนับจำนวนหมากทั้งสองฝ่ายแล้ว ปัญญาประดิษฐ์เป็นฝ่ายชนะไปด้วยจำนวนหมาก 36 - 28

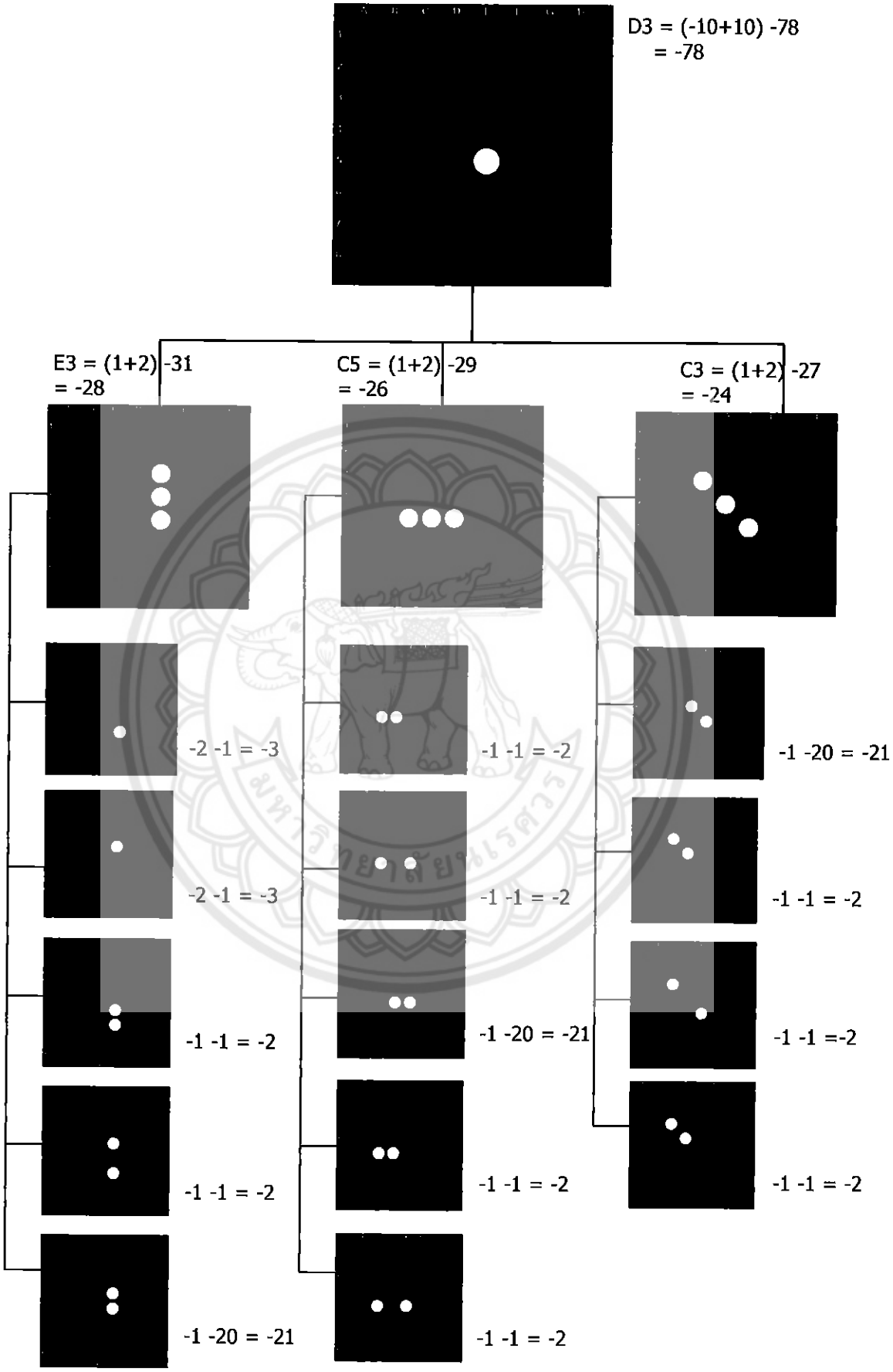
### 3.5.4 การคิดของปัญญาประดิษฐ์ (Level 2)



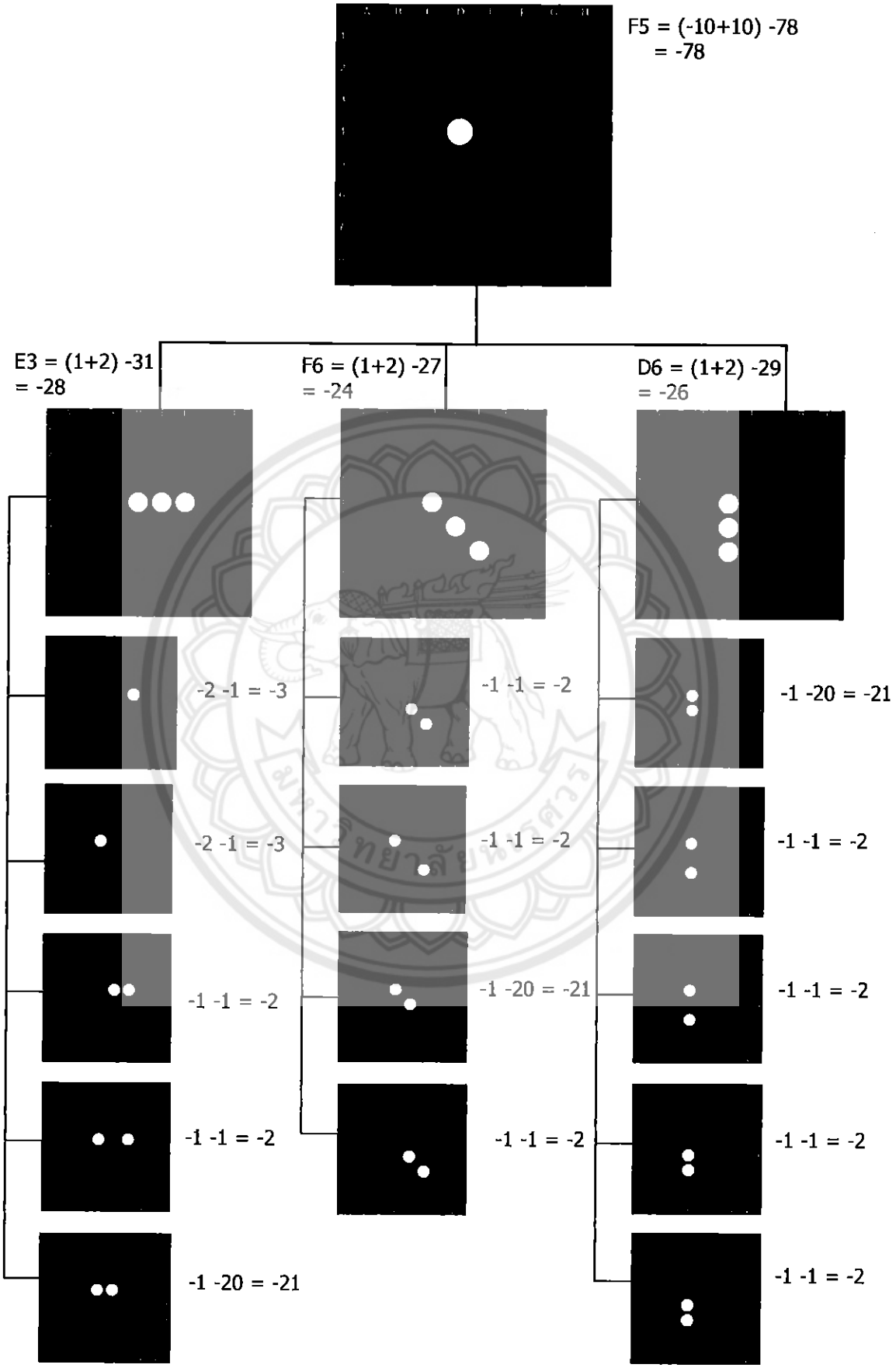

รูปที่ 3.15 ภาพแสดง แผนภาพ Tree การคิดคะแนนในตำแหน่งเริ่มต้น



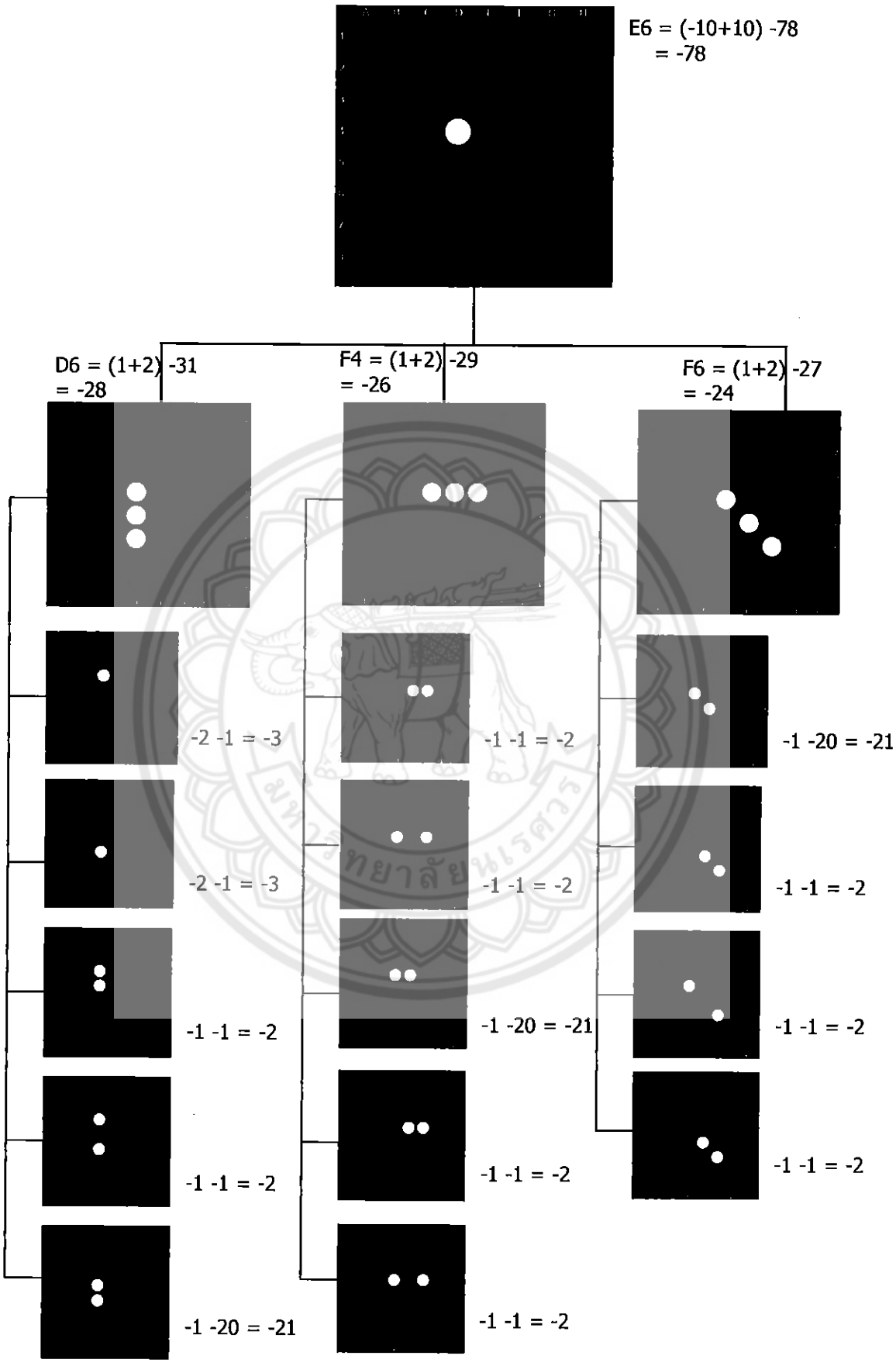

**รูปที่ 3.16** ภาพแสดง แผนภาพ Tree การคิดคะแนนในตำแหน่ง C4




**รูปที่ 3.17** ภาพแสดง แผนภาพ Tree การคิดคะแนนในตำแหน่ง D3

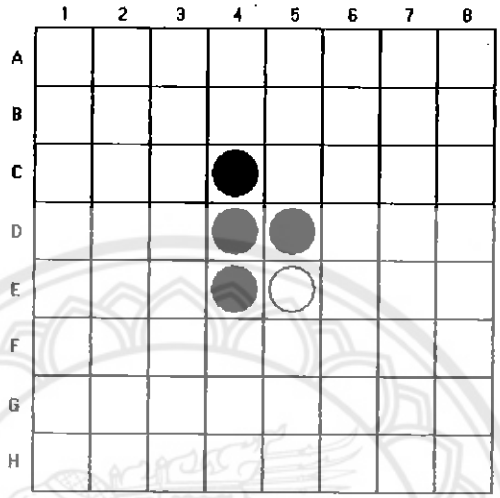



**รูปที่ 3.18** ภาพแสดง แผนภาพ Tree การคิดคะแนนในตำแหน่ง F5




**รูปที่ 3.19** ภาพแสดง แผนภาพ Tree การคิดคะแนนในตำแหน่ง E6

แผนภาพ Tree ดังกล่าวนี้เป็นสิ่งที่แสดงให้เห็นถึง โครงสร้างการคำนวณคะแนนของแต่ละตำแหน่ง เพื่อให้ได้คะแนนรวมออกมาก่อนที่ ปัญญาประดิษฐ์จะตัดสินใจเดินมากในตำแหน่ง ที่ได้คะแนนคำนวณออกมามากที่สุด และในการแสดงตัวอย่างการเล่นโอเทลโล่เกม กับปัญญาประดิษฐ์ระดับที่ 2 นั้น ปัญญาประดิษฐ์ก็จะใช้ วิธีการดังที่กล่าวมา ในการคำนวณหาตำแหน่งเดินหมาก




รูปที่ 3.20 ภาพแสดงหมากเมื่อเริ่มต้นเล่นเกมต่อจากผู้เล่นเดินในตำแหน่ง C4

หลังจากที่ผู้เล่นเดินหมากไปแล้วในตำแหน่ง C4 แล้ว ปัญญาประดิษฐ์จะสามารถเดินหมากได้ในตำแหน่ง สีฟ้า ที่แสดงให้เห็นจากรูป ซึ่งได้แก่ ตำแหน่ง C3 C5 และ E3

ในการที่ ปัญญาประดิษฐ์จะตัดสินใจลงหมากที่ตำแหน่งใดนั้น จะตัดสินใจจากการรวมคะแนนที่คำนวนได้ โดยการรวมคะแนนใน Level 2 นี้มีวิธิการคิดดังนี้ คือ

1. ปัญญาประดิษฐ์ เช็คตำแหน่งที่สามารถเดินได้
2. ปัญญาประดิษฐ์ จำลองการเดินในตำแหน่งจาก ข้อ 1 โดยใช้คะแนนสำหรับปัญญาประดิษฐ์ (คะแนนที่ 1) คะแนนกินหมากฝ่ายตรงข้าม หมากละ -10 คะแนน
3. เช็คตำแหน่งที่คู่แข่งสามารถเดินได้ ต่อจากตาที่ผ่านมา ปัญญาประดิษฐ์ จำลองการเดินของคู่แข่งในตำแหน่งจากตาที่ผ่านมา ใช้คะแนนรวมสำหรับคู่แข่งเดิน (คะแนนที่ 2) คะแนนในการกินหมากฝ่ายตรงข้าม หมากละ 1 คะแนน
4. เช็คตำแหน่งที่สามารถเดินได้ ต่อจากตาที่ผ่านมา ปัญญาประดิษฐ์ จำลองการเดิน ให้ครบทุกตำแหน่งจาก ข้อ 3 โดยใช้คะแนนสำหรับปัญญาประดิษฐ์ (คะแนนที่ 3) คะแนนในการกินหมากฝ่ายตรงข้าม หมากละ -1 คะแนน
5. นำ คะแนนที่ 1, คะแนนที่ 2 และ คะแนนที่ 3 มารวมกัน จะได้คะแนนรวมของตำแหน่ง ที่เลือกเดินจาก ข้อ 1 ได้

6. ในการหาคะแนนรวมของตำแหน่งอื่นๆ ใน ข้อ 1 นั้น จะคำนวณตาม ข้อ 2 – 4 ตามลำดับ จนครบทุกตำแหน่ง

**ตารางที่ 3.2** ตารางคะแนนสำหรับปัญญาประดิษฐ์ (คะแนนที่ 1)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 200 | -600 | -400 | -400 | -400 | -400 | -100 | 200 |
| -600 | -800 | -200 | -200 | -200 | -200 | -800 | -100 |
| -400 | -200 | 10 | 10 | 10 | 10 | -200 | -400 |
| -400 | -200 | 10 | 0 | 0 | 10 | -200 | -400 |
| -400 | -200 | 10 | 0 | 0 | 10 | -200 | -400 |
| -400 | -200 | 10 | 10 | 10 | 10 | -200 | -400 |
| -600 | -800 | -200 | -200 | -200 | -200 | -800 | -600 |
| 200 | -600 | -400 | -400 | -400 | -400 | -600 | 200 |

ตารางที่ 3.3 ตารางคะแนนสำหรับคู่แข่งเดิน (คะแนนที่ 2)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| -100 | 30 | 30 | 30 | 30 | 30 | 1000 | -100 |
| 30 | 90 | -30 | -30 | -30 | -30 | 90 | 30 |
| 30 | -30 | 2 | 2 | 2 | 2 | -30 | -30 |
| 30 | -30 | 2 | 0 | 0 | 2 | -30 | 30 |
| 30 | -30 | 2 | 0 | 0 | 2 | -30 | 30 |
| 30 | -30 | 2 | 2 | 2 | 2 | -30 | 30 |
| 30 | 90 | -30 | -30 | -30 | -30 | 90 | 30 |
| -100 | 30 | 30 | 30 | 30 | 30 | 30 | -100 |

**ตารางที่ 3.4 ตารางคะแนนสำหรับปัญญาประดิษฐ์ (คะแนนที่ 3)**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 20 | -60 | -40 | -40 | -40 | -40 | -10 | 20 |
| -60 | -80 | -20 | -20 | -20 | -20 | -80 | -10 |
| -40 | -20 | 1 | 1 | 1 | 1 | -20 | -40 |
| -40 | -20 | 1 | 0 | 0 | 1 | -20 | -40 |
| -40 | -20 | 1 | 0 | 0 | 1 | -20 | -40 |
| -40 | -20 | 1 | 1 | 1 | 1 | -20 | -40 |
| -60 | -80 | -20 | -20 | -20 | -20 | -80 | -60 |
| 20 | -60 | -40 | -40 | -40 | -40 | -60 | 20 |

ดังนั้น แต่ละตำแหน่งคิดคะแนนได้ดังนี้ คือ

C3 กินหมากคู่แข่ง -10 P. ตำแหน่งที่ลง 10 P. รวมเป็น 0 คะแนน

**คู่แข่ง** C2 กินหมาก 1 P. ตำแหน่งที่ลง -30 P. รวมเป็น -29 P. (คะแนนรวม -29)

**คู่แข่ง** -> AI B2 กินหมาก -1P. ตำแหน่งที่ลง -80 P. รวมเป็น -81 P. (คะแนนรวม -81)

**คู่แข่ง** -> AI B4 กินหมาก -2P. ตำแหน่งที่ลง -20 P. รวมเป็น -22 P. (คะแนนรวม -103)

**คู่แข่ง** -> AI C5 กินหมาก -1P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -105)

**คู่แข่ง** -> AI D6 กินหมาก -1P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -107)

**คู่แข่ง** -> AI E3 กินหมาก -1P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -109)

**คู่แข่ง** -> AI F4 กินหมาก -2P. ตำแหน่งที่ลง -1 P. รวมเป็น -3 P. (คะแนนรวม -112)

**คู่แข่ง** -> AI คะแนนรวม -112 P.

**คู่แข่ง** D3 กินหมาก 1 ตำแหน่งที่ลง 2 P. รวมเป็น 3 P. (คะแนนรวม -26)

**คู่แข่ง** -> AI C5 กินหมาก -2 P. ตำแหน่งที่ลง -1 P. รวมเป็น -3 P. (คะแนนรวม -115)

**คู่แข่ง** -> AI E3 กินหมาก -2 P. ตำแหน่งที่ลง -1 P. รวมเป็น -3 P. (คะแนนรวม -118)

**คู่แข่ง -> AI คะแนนรวม -118 P.**

**คู่แข่ง E6 กินหมาก 1 ตำแหน่งที่ลง 2 P. รวมเป็น 3 P. (คะแนนรวม -23)**

**คู่แข่ง -> AI B4 กินหมาก -2 P. ตำแหน่งที่ลง -20 P. รวมเป็น -22 P. (คะแนนรวม -140)**

**คู่แข่ง -> AI C5 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -142)**

**คู่แข่ง -> AI D6 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -144)**

**คู่แข่ง -> AI F4 กินหมาก -2 P. ตำแหน่งที่ลง -1 P. รวมเป็น -3 P. (คะแนนรวม -147)**

**คู่แข่ง -> AI F6 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -149)**

**คู่แข่ง -> AI คะแนนรวม -149 P.**

**คู่แข่ง F5 กินหมาก 1 P. ตำแหน่งที่ลง 2 P. รวมเป็น 3 P. (คะแนนรวม -20)**

คู่แข่ง -> AI B4 กินหมาก -2 P. ตำแหน่งที่ลง -20 P. รวมเป็น -22 P. (คะแนนรวม -171)

คู่แข่ง -> AI C5 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -173)

คู่แข่ง -> AI D6 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -175)

คู่แข่ง -> AI F4 กินหมาก -2 P. ตำแหน่งที่ลง -1 P. รวมเป็น -3 P. (คะแนนรวม -178)

คู่แข่ง -> AI F6 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -180)

คู่แข่ง -> AI คะแนนรวม -180 P.

คู่แข่ง คะแนนรวม -20 P.

C3 คะแนนรวม 0 -20 -180 = -200 P.

C5 กินหมากคู่แข่ง -10 P. ตำแหน่งที่ลง 10 P. รวมเป็น 0 คะแนน

คู่แข่ง B6 กินหมาก 1 P. ตำแหน่งที่ลง -30 P. รวมเป็น -29 P. (คะแนนรวม -29)

คู่แข่ง -> AI B3 กินหมาก -1 P. ตำแหน่งที่ลง -20 P. รวมเป็น -21 P. (คะแนนรวม -21)

คู่แข่ง -> AI B5 กินหมาก -1 P. ตำแหน่งที่ลง -20 P. รวมเป็น 21 P. (คะแนนรวม -42)

คู่แข่ง -> AI C3 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -44)

คู่แข่ง -> AI D3 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -46)

คู่แข่ง -> AI E3 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -48)

คู่แข่ง -> AI F3 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -50)

คู่แข่ง -> AI คะแนนรวม -50 P.

คู่แข่ง C6 กินหมาก 2 P. ตำแหน่งที่ลง 2 P. รวมเป็น 4 P. (คะแนนรวม -25)

คู่แข่ง -> AI B5 กินหมาก -2 P. ตำแหน่งที่ลง -20 P. รวมเป็น -22 P. (คะแนนรวม -72)

คู่แข่ง -> AI C3 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -74)

คู่แข่ง -> AI E3 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -76)

คู่แข่ง -> AI คะแนนรวม -76 P.

คู่แข่ง D6 กินหมาก 1 P. ตำแหน่งที่ลง 2 P. รวมเป็น 3 (คะแนนรวม -22)

คู่แข่ง -> AI C3 กินหมาก -2 P. ตำแหน่งที่ลง -1 P. รวมเป็น -3 P. (คะแนนรวม -79)

คู่แข่ง -> AI C7 กินหมาก -1 P. ตำแหน่งที่ลง -20 P. รวมเป็น -21 P. (คะแนนรวม -100)

คู่แข่ง -> AI E3 กินหมาก -2 P. ตำแหน่งที่ลง -1 P. รวมเป็น -3 P. (คะแนนรวม -103)

คู่แข่ง -> AI E7 กินหมาก -1 P. ตำแหน่งที่ลง -20 P. รวมเป็น -21 P. (คะแนนรวม -124)

คู่แข่ง -> AI คะแนนรวม -124 P.

คู่แข่ง E6 กินหมาก 2 P. ตำแหน่งที่ลง 2 P. รวมเป็น 4 P. (คะแนนรวม -18)

คู่แข่ง -> AI C3 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -126)

คู่แข่ง -> AI E3 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -128)

คู่แข่ง -> AI F5 กินหมาก -2 P. ตำแหน่งที่ลง -1 P. รวมเป็น -3 P. (คะแนนรวม -131)

คู่แข่ง -> AI คะแนนรวม -131 P.

คู่แข่ง F6 กินหมาก 1 P. ตำแหน่งที่ลง 2 P. รวมเป็น 3 P. (คะแนนรวม -15)

คู่แข่ง -> AI B3 กินหมาก -1 P. ตำแหน่งที่ลง -20 P. รวมเป็น -21 P. (คะแนนรวม -152)

คู่แข่ง -> AI C3 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -154)

คู่แข่ง -> AI D3 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -156)

คู่แข่ง -> AI E3 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -158)

คู่แข่ง -> AI F3 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -160)

คู่แข่ง -> AI F5 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -162)

คู่แข่ง -> AI คะแนนรวม -162 P.

คู่แข่ง คะแนนรวม -15 P.

C5 คะแนนรวม 0 -15 -162 = -177 P.

E3 กินหมากคู่แข่ง -10 P. ตำแหน่งที่ลง 10 P. รวมเป็น 0 P.

คู่แข่ง F2 กินหมาก 1 P. ตำแหน่งที่ลง -30 P. รวมเป็น -29 P. (คะแนนรวม -29)

คู่แข่ง -> AI B4 กินหมาก -2 P. ตำแหน่งที่ลง -20 P. รวมเป็น -22 P. (คะแนนรวม -22)

คู่แข่ง -> AI C3 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -24)

คู่แข่ง -> AI C5 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -26)

คู่แข่ง -> AI C6 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -28)

คู่แข่ง -> AI E2 กินหมาก -1 P. ตำแหน่งที่ลง -20 P. รวมเป็น -21 P. (คะแนนรวม -49)

คู่แข่ง -> AI คะแนนรวม -49 P.

คู่แข่ง F3 กินหมาก 1 P. ตำแหน่งที่ลง 2 P. รวมเป็น -3 P. (คะแนนรวม -26)

คู่แข่ง -> AI C3 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -51)

คู่แข่ง -> AI C5 กินหมาก -2 P. ตำแหน่งที่ลง -1 P. รวมเป็น -3 P. (คะแนนรวม -54)

คู่แข่ง -> AI G3 กินหมาก -1 P. ตำแหน่งที่ลง -20 P. รวมเป็น -21 P. (คะแนนรวม -75)

คู่แข่ง -> AI คะแนนรวม -75 P.

คู่แข่ง F4 กินหมาก 1 P. ตำแหน่งที่ลง 2 P. รวมเป็น 3 P. (คะแนนรวม -23)

คู่แข่ง -> AI C3 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -77)

คู่แข่ง -> AI C5 กินหมาก -2 P. ตำแหน่งที่ลง -1 P. รวมเป็น -3 P. (คะแนนรวม -80)

คู่แข่ง -> AI G3 กินหมาก -1 P. ตำแหน่งที่ลง -20 P. รวมเป็น -21 P. (คะแนนรวม -101)

คู่แข่ง -> AI G5 กินหมาก -1 P. ตำแหน่งที่ลง -20 P. รวมเป็น -21 P. (คะแนนรวม -122)

คู่แข่ง -> AI คะแนนรวม -122 P.

คู่แข่ง F5 กินหมาก 1 P. ตำแหน่งที่ลง 2 P. รวมเป็น 3 P. (คะแนนรวม -20)

คู่แข่ง -> AI B4 กินหมาก -2 P. ตำแหน่งที่ลง -20 P. รวมเป็น -22 P. (คะแนนรวม -144)

คู่แข่ง -> AI C5 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -146)

คู่แข่ง -> AI C6 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -148)

คู่แข่ง -> AI E6 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -150)

คู่แข่ง -> AI G6 กินหมาก -1 P. ตำแหน่งที่ลง -20 P. รวมเป็น -21 P. (คะแนนรวม -171)

คู่แข่ง -> AI คะแนนรวม -171 P.

คู่แข่ง F6 กินหมาก 1 P. ตำแหน่งที่ลง 2 P. รวมเป็น 3 P. (คะแนนรวม -17)

คู่แข่ง -> AI B4 กินหมาก -2 P. ตำแหน่งที่ลง -20 P. รวมเป็น -22 P. (คะแนนรวม -193)

คู่แข่ง -> AI C5 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -195)

คู่แข่ง -> AI C6 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -197)

คู่แข่ง -> AI E6 กินหมาก -1 P. ตำแหน่งที่ลง -1 P. รวมเป็น -2 P. (คะแนนรวม -199)

คู่แข่ง -> AI คะแนนรวม -199 P.

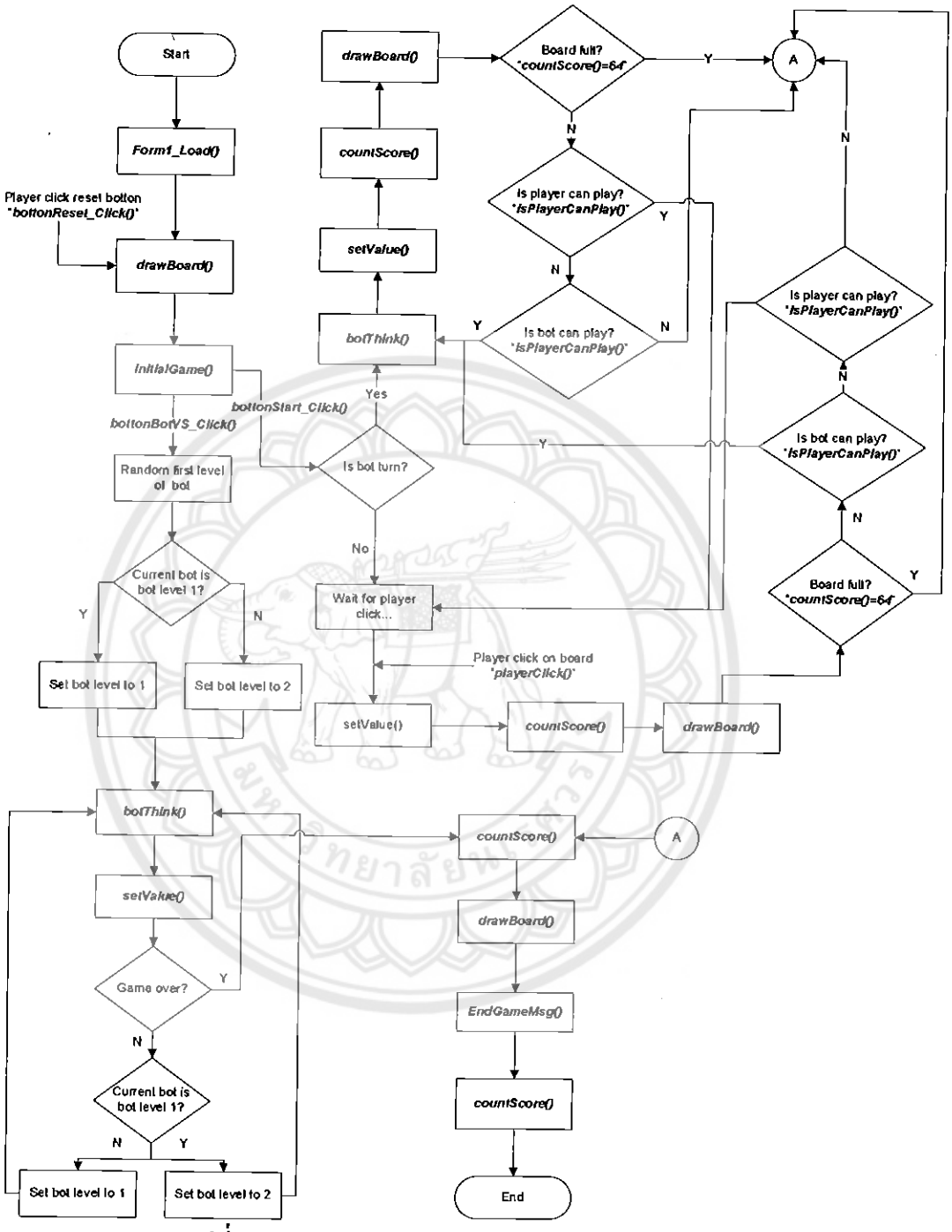
คู่แข่ง คะแนนรวม -17 P.

E3 คะแนนรวม 0 -17 -199 = -216 P.

เมื่อในแต่ละตำแหน่งที่สามารถเดินหมากได้ มีคะแนนรวมครบหมดแล้ว ปัญญาประดิษฐ์จะเลือกเดินในตำแหน่งที่ได้คะแนนรวมมากที่สุดนั่นเอง และในกรณีนี้ที่ตำแหน่งที่ได้คะแนนรวมมากที่สุดคือ ตำแหน่ง C5 ปัญญาประดิษฐ์จะเลือกเดินในตำแหน่งนี้

และในรอบถัดไป ปัญญาประดิษฐ์จะใช้วิธีการ ดังที่กล่าวมา ในการคำนวณหาคะแนนที่มากที่สุด ของแต่ละตำแหน่ง เพื่อเลือกเดินในตำแหน่งที่ดีที่สุด จนเกมจบลง

## 3.6 โครงสร้างการทำงานของ Code โปรแกรม

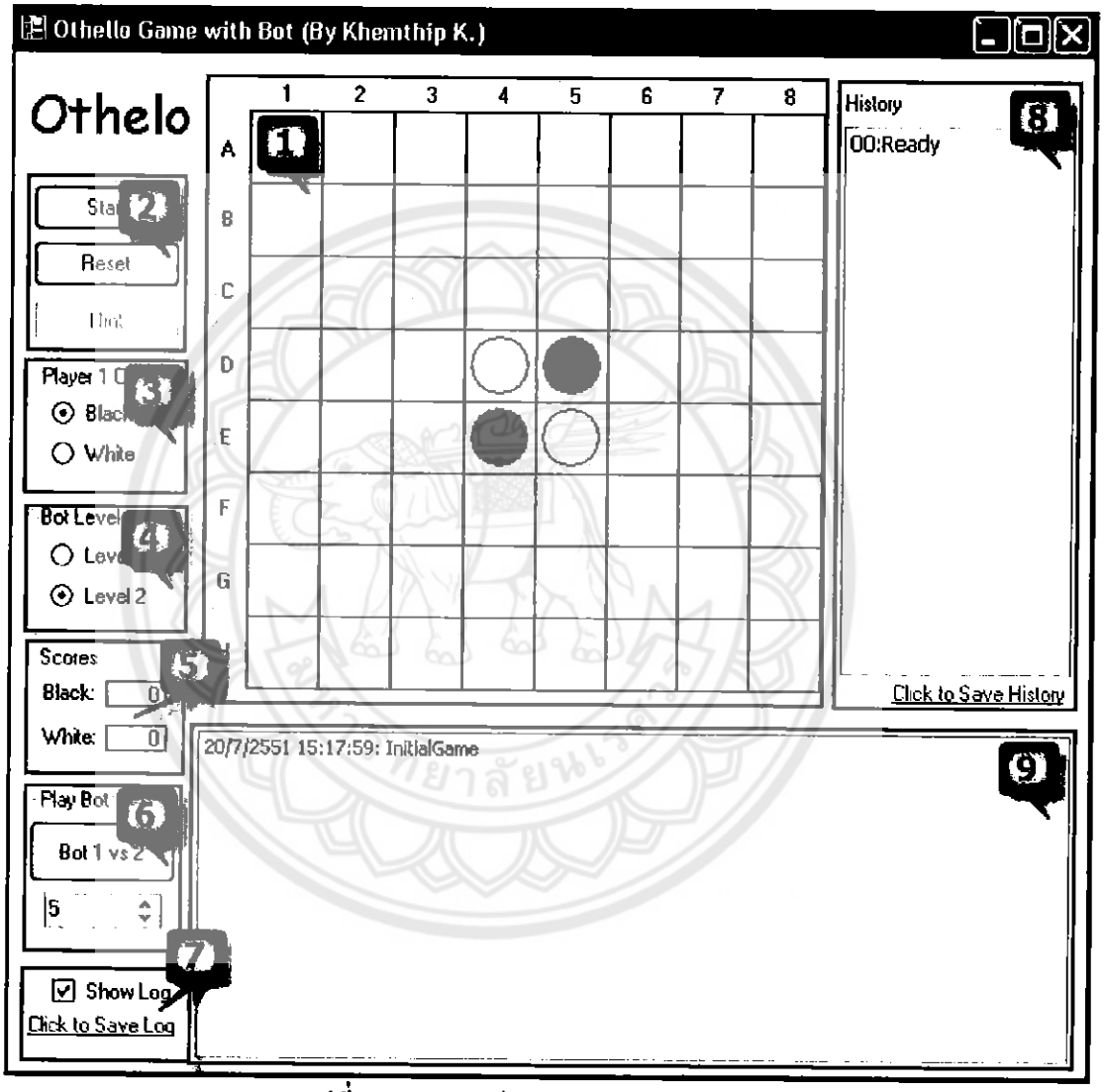



**รูปที่ 3.21** แสดง Diagram การทำงานของ code โปรแกรม

# บทที่ 4

# ผลการทดลอง

## 4.1 ผลการทดลองการเริ่มทำงานของโปรแกรม Othello Game




**รูปที่ 4.1** แสดงหน้าต่างตัวเกม Othello Game

เมื่อเริ่มต้นโปรแกรมขึ้นมา ระบบจะแสดงหน้าต่างเกมขึ้นมา และมีส่วนประกอบต่างๆ ของหน้าต่างเกม อธิบายได้ดังนี้ คือ

1. กระดานของเกมโอเทลโล่
2. ปุ่มกดต่างๆ เพื่อควบคุมเกมซึ่งได้แก่ Start, Reset และ Hint
3. ช่องสำหรับเลือกสีของเบี้ย สำหรับผู้เล่น

4. ช่องสำหรับเลือก ระดับของปัญญาประดิษฐ์
5. ช่องแสดงคะแนน ของแต่ละสี
6. ส่วนสำหรับการกำหนดการแข่งกันของปัญญาประดิษฐ์ ทั้ง 2 ระดับ
7. ช่องสำหรับเลือกเพื่อให้ระบบแสดงและบันทึก Log ของเกม
8. ช่องสำหรับแสดงและบันทึก History ของเกม
9. ช่องสำหรับแสดง Log ของเกม

## 4.2 ผลการทดลองการทำงานของส่วนต่างๆ ของโปรแกรม Ohtello Game

การทำงานของแต่ละส่วนของโปรแกรม สามารถอธิบายได้ดังนี้

**ก่อนการเริ่มเล่นเกม**

เมื่อเปิดเกมขึ้นมาที่หน้าต่างของเกมแล้ว ก่อนที่จะมีการเริ่มเกมนั้น **ผู้เล่นจะต้องมีการควบคุม**ส่วนต่างๆ ก่อนการเริ่มเกมดังนี้ คือ

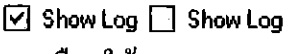
- Color เลือกสีของเบี้ยได้ ระหว่างสีดำ และสีขาว


**รูปที่ 4.2** ภาพแสดงการเลือกสีระหว่าง สีดำ หรือ สีขาว

- Bot Level เลือกระดับของปัญญาประดิษฐ์ ระหว่างระดับที่ 1 หรือ ระดับที่ 2


**รูปที่ 4.3** ภาพแสดงการเลือระดับของปัญญาประดิษฐ์

- Show Log คลิกเลือกเพื่อให้ระบบ แสดงหรือไม่แสดง Log ของเกม




**รูปที่ 4.4** ภาพแสดงการเลือกให้ระบบแสดง Log ของเกมหรือไม่

- ช่องแสดง Log ของเกม


รูปที่ 4.5 ภาพแสดงบริเวณที่ ใช้แสดง Log ของเกม

เมื่อเริ่มต้นโปรแกรมขึ้นมา ระบบจะแสดง วัน/เดือน/ปี เวลา และข้อความ InitialGame

- Click to Save Log เป็น Link ที่กดเพื่อทำการบันทึก Log ของเกมไว้

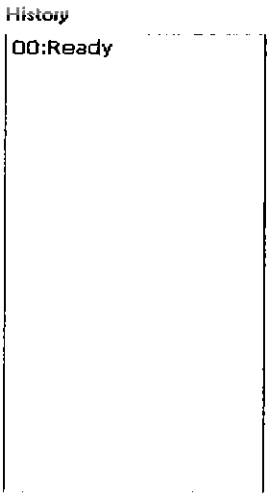
Click to Save Log

รูปที่ 4.6 ภาพแสดง Link ที่กดเพื่อให้ระบบบันทึก Log ของเกม

เมื่อกด Link Click to Save Log ระบบจะทำการบันทึก Log ไว้ใน Path :

D:\ProjectOthello\othello\Othello\Othello\bin\Debug

- ช่องแสดง History ของเกม แสดง History ของเกมในการเดินแต่ละครั้ง




รูปที่ 4.7 ภาพแสดง บริเวณที่ระบบใช้แสดง History ของเกม

เมื่อเริ่มต้นโปรแกรม ส่วนนี้จะแสดงข้อความเป็น 00: Ready

- Click to Save History เป็น Link ที่กดเพื่อทำการบันทึก History ของเกมไว้

Click to Save History

**รูปที่ 4.8** ภาพแสดง Link ที่กดเพื่อให้ระบบบันทึก History ของเกม

เมื่อกด Link Click to Save History ระบบจะทำการบันทึก History ไว้ใน Path : D:\ProjectOthello\othello\Othello\Othello\bin\Debug

- Start ปุ่มกดเพื่อเริ่มเล่นเกม

Start

**รูปที่ 4.9** ภาพแสดง ปุ่ม Start กดเพื่อเริ่มเกม

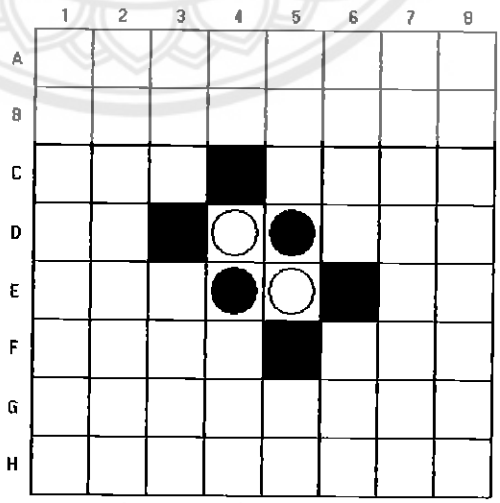
ขณะกำลังเล่นเกม

เมื่อเริ่มเล่นเกมแล้ว ขณะที่กำลังเล่นเกมอยู่นั้น สามารถควบคุมกระดานเกมในส่วนต่างๆ ได้ดังนี้ คือ

- Hint ปุ่มกดเพื่อให้ระบบ แสดงสียังตำแหน่งที่สามารถเดินได้ ให้กับผู้เล่น

Hint

**รูปที่ 4.10** ภาพแสดง ปุ่ม Hint กดเพื่อแสดงตำแหน่งที่สามารถเดินได้

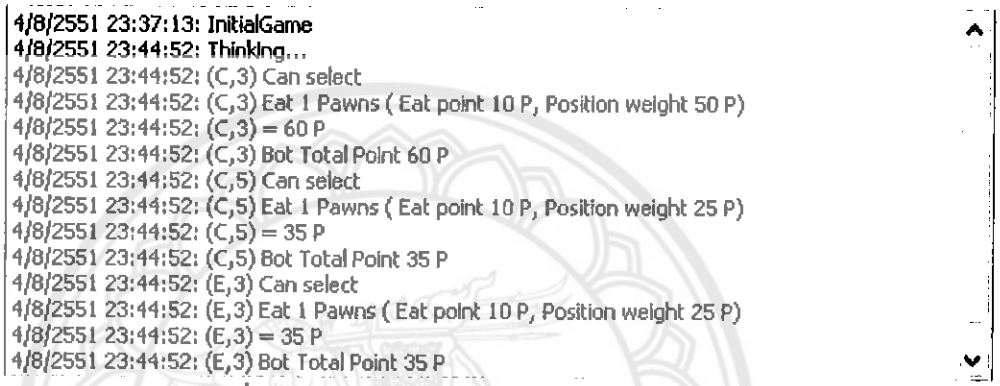



**รูปที่ 4.11** ภาพแสดง สีที่ตำแหน่งที่สามารถเดินได้ ภายหลังกดปุ่ม Hint

- Show Log คลิกเลือกเพื่อให้ระบบ แสดงหรือไม่แสดง Log ของเกม

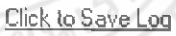

**รูปที่ 4.12** ภาพแสดงตัวเลือก การแสดง Log ของเกม

- ช่องแสดง Log ของเกม




**รูปที่ 4.13** ภาพแสดง Log ของเกมเมื่อเลือก Show Log

- Click to Save Log เป็น Link ที่กดเพื่อทำการบันทึก Log ของเกมไว้

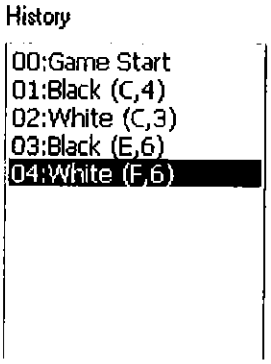



**รูปที่ 4.14** ภาพแสดง Link Click to Save Log

เมื่อกด Link Click to Save Log ระบบจะทำการบันทึก Log ไว้ใน Path :

D:\ProjectOthello\othello\Othello\Othello\bin\Debug

- ช่องแสดง History ของเกม




**รูปที่ 4.15** ภาพแสดง History ของเกม

- Click to Save History เป็น Link ที่กดเพื่อทำการบันทึก History ของเกมไว้


**รูปที่ 4.16** ภาพแสดง Link Click to Save History

เมื่อกด Link Click to Save History ระบบจะทำการบันทึก History ไว้ใน Path : D:\ProjectOthello\othello\Othello\Othello\bin\Debug

- Reset ปุ่มกดเพื่อ clear ทุกๆ ส่วนของเกม ให้กลับไปอยู่ในลักษณะ ก่อนเริ่มเกม


รูปที่ 4.17 ภาพแสดง ปุ่ม Reset

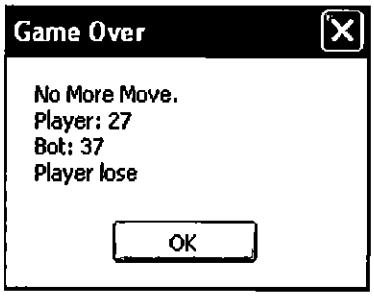
- Pop Up การผ่านเกม เมื่อฝ่ายใดฝ่ายหนึ่งไม่มีตาเดินในรอบนั้น ก็จะผ่านให้ฝ่ายตรงข้ามเดินต่อหนึ่งรอบ หรือจนกว่าจะมีตาเดิน หรือจนกว่าเกมจะจบลง


รูปที่ 4.18 ภาพแสดง Pop Up การผ่านเกม

การจบเกม

หลังจากที่เริ่มเล่นเกมมาจนกระทั่ง จบเกม การจบเกมนั้นมีหลายกรณี เช่น กระดานถูกเดินจนเต็ม หรือ มีตำแหน่งว่างอยู่แต่ทั้งสองฝ่ายไม่สามารถเดินยังตำแหน่งนั้นได้ หรือ หมากทั้งกระดานที่มีอยู่ เป็นสีใดสีหนึ่งทั้งหมด หรือ เมื่อทั้งสองฝ่ายไม่สามารถเดินต่อได้และมีจำนวนเบี้ยของทั้งสองฝ่ายเท่ากัน โดยลักษณะต่างๆ เหล่านี้ จะมี Pop Up ขึ้นมาเพื่อแจ้งสถานะการจบเกม ดังนี้คือ

- Pop Up การจบเกมแบบเดินจนเต็มกระดาน




**รูปที่ 4.19** ภาพแสดง Pop Up การจบเกมแบบที่ 1

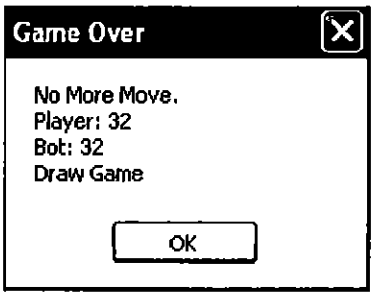
- Pop Up การจบเกมแบบมีตำแหน่งว่าง แต่ทั้งสองฝั่งไม่สามารถเดินต่อได้


**รูปที่ 4.20** ภาพแสดง Pop Up การจบเกมแบบที่ 2

- Pop Up การจบเกมแบบหมากทั้งกระดานเป็นสีใดสีหนึ่ง


**รูปที่ 4.21** ภาพแสดง Pop Up การจบเกมแบบที่ 3

- Pop Up การจบเกมแบบเสมอกัน




**รูปที่ 4.22** ภาพแสดง Pop Up การจบเกมแบบที่ 4

- Reset ปุ่มกดเพื่อ clear ทุกๆ ส่วนของเกม ให้กลับไปอยู่ในลักษณะ ก่อนเริ่มเกม


รูปที่ 4.23 ภาพแสดง ปุ่ม Reset

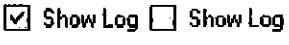
การแข่งขันกันของปัญญาประดิษฐ์

เมื่อต้องการดูผลการแข่งขันกัน ระหว่างปัญญาประดิษฐ์ ในส่วนนี้สามารถควบคุมการแข่งขันกันของปัญญาประดิษฐ์ได้ดังนี้ คือ

- เลือกจำนวนรอบ ที่จะให้ปัญญาประดิษฐ์แข่งขันกัน

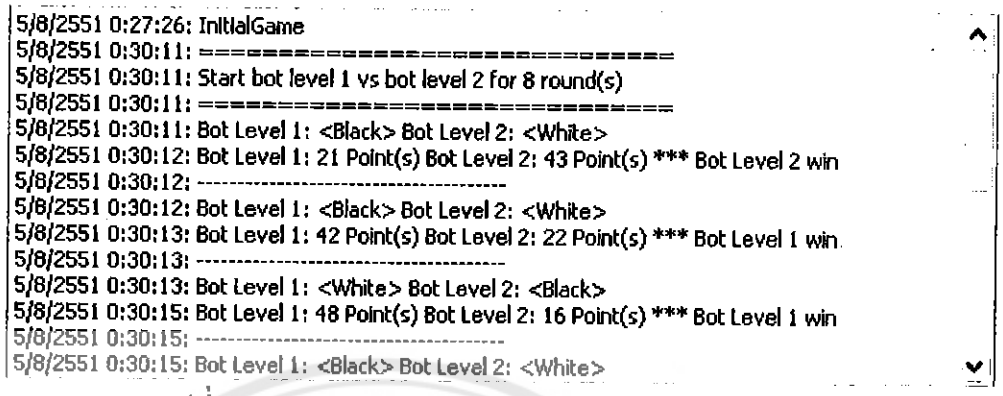

รูปที่ 4.24 ภาพแสดง การเลือกจำนวนรอบ

- Show Log คลิกเลือกเพื่อให้ระบบ แสดงหรือไม่แสดง Log ของเกม



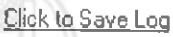

**รูปที่ 4.25** ภาพแสดง การเลือกการ Show Log

- ช่องแสดง Log ของเกม




รูปที่ 4.26 ภาพแสดง Log การแข่งขันกันของ ปัญญาประดิษฐ์

- Click to Save Log เป็น Link ที่กดเพื่อทำการบันทึก Log ของเกมไว้




รูปที่ 4.27 ภาพแสดง Link Click to Save Log

- Bot 1 vs 2 ปุ่มกดเพื่อให้ปัญญาประดิษฐ์เริ่มแข่งขันกัน

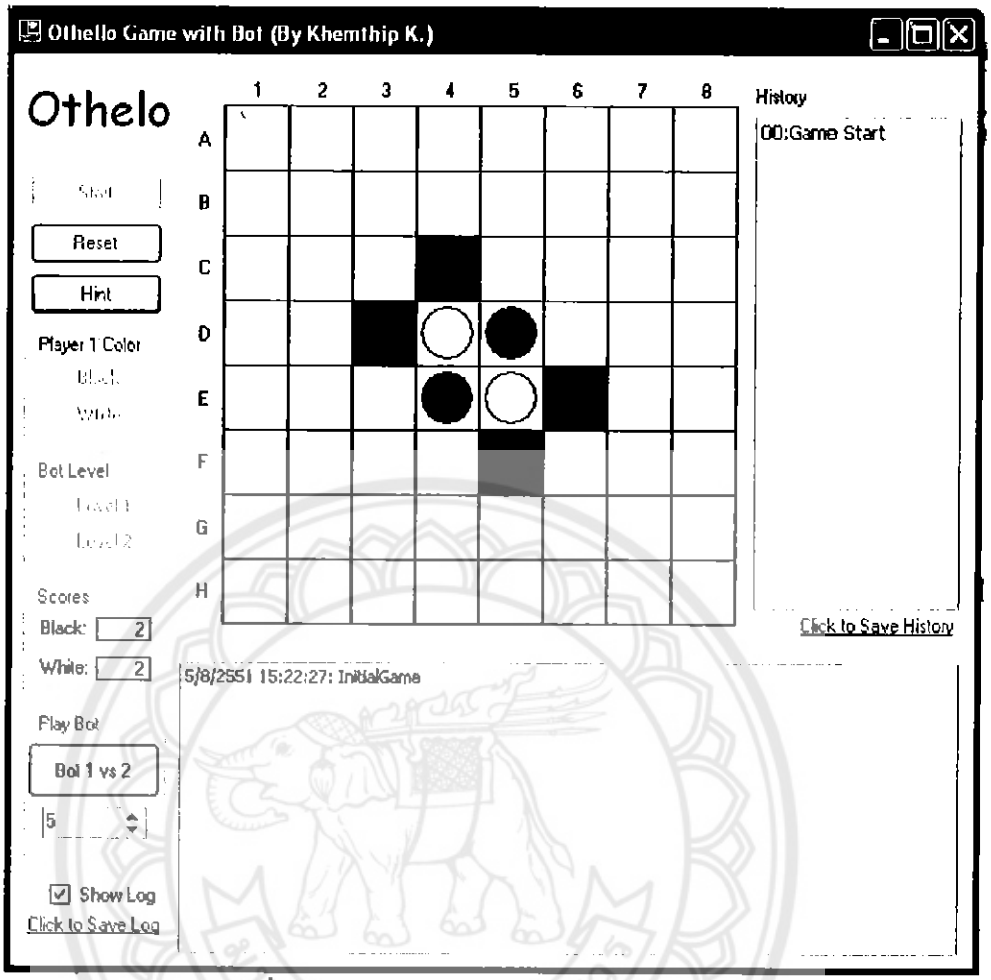

รูปที่ 4.28 ภาพแสดง ปุ่มกด Bot 1 vs 2

## 4.3 ผลการทดลอง การทำงานของปัญญาประดิษฐ์ใน Othello game

### 4.3.1 การทำงานของปัญญาประดิษฐ์ ระดับที่ 1

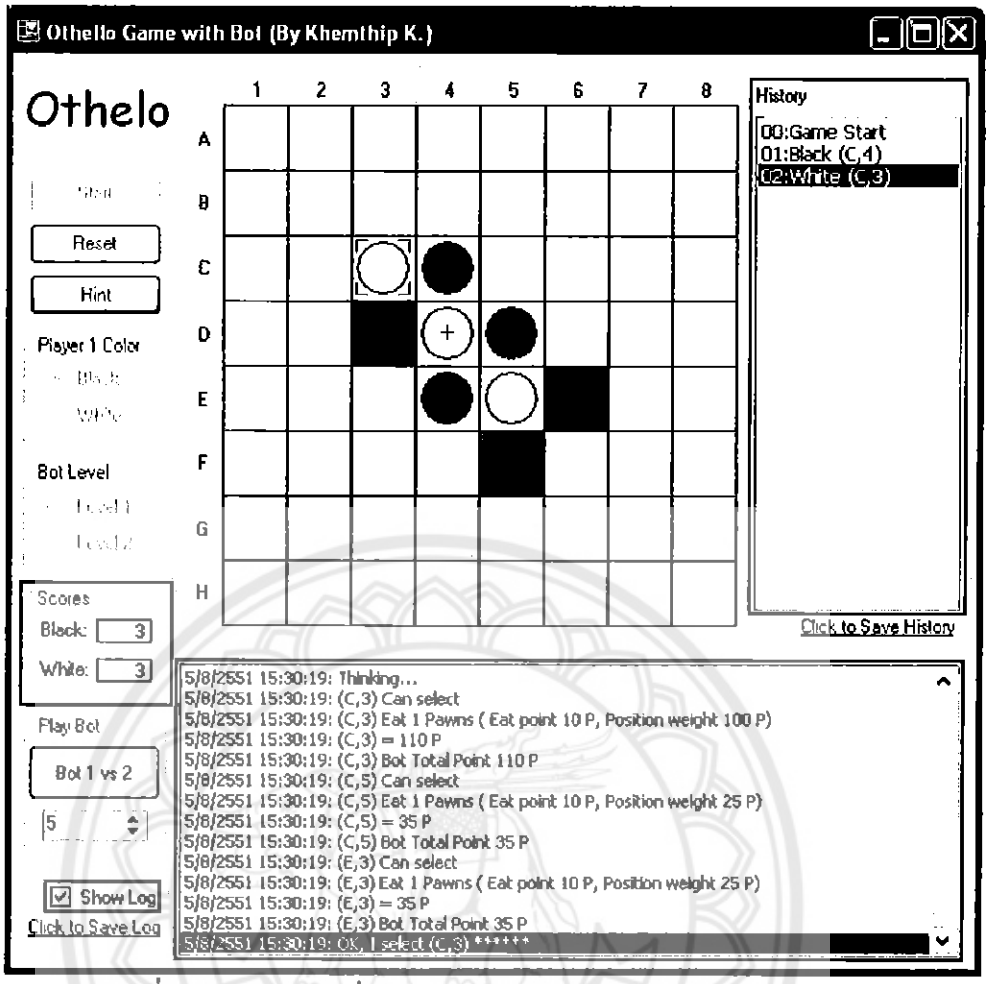
ผลการทดลอง การเล่นหมากกระดาน Othello กับปัญญาประดิษฐ์ ระดับที่ 1 จะแสดงเป็นลำดับ จนกระทั่งจบเกม ดังต่อไปนี้

เมื่อเริ่มเปิดโปรแกรมขึ้นมา ก่อนเล่นจะทำการเลือกสีเบี้ยของผู้เล่น ในที่นี้เลือกเป็นสีดำ และแข่งกับปัญญาประดิษฐ์ ระดับที่ 1 และเลือกให้ระบบ Show Log ของเกมด้วย แล้วจึงกดปุ่ม Start เกมก็จะเริ่มขึ้น และเมื่อเริ่มเกมแล้ว หากผู้เล่นที่ยังไม่ชำนาญมากนัก ต้องการทราบว่าตำแหน่งใดสามารถเดินได้บ้าง เมื่อกดที่ปุ่ม Hint ระบบจะแสดง ตำแหน่งที่สามารถเดินได้ขึ้นมา เป็นตำแหน่งที่ทำเครื่องหมายเป็นสี ดังภาพ



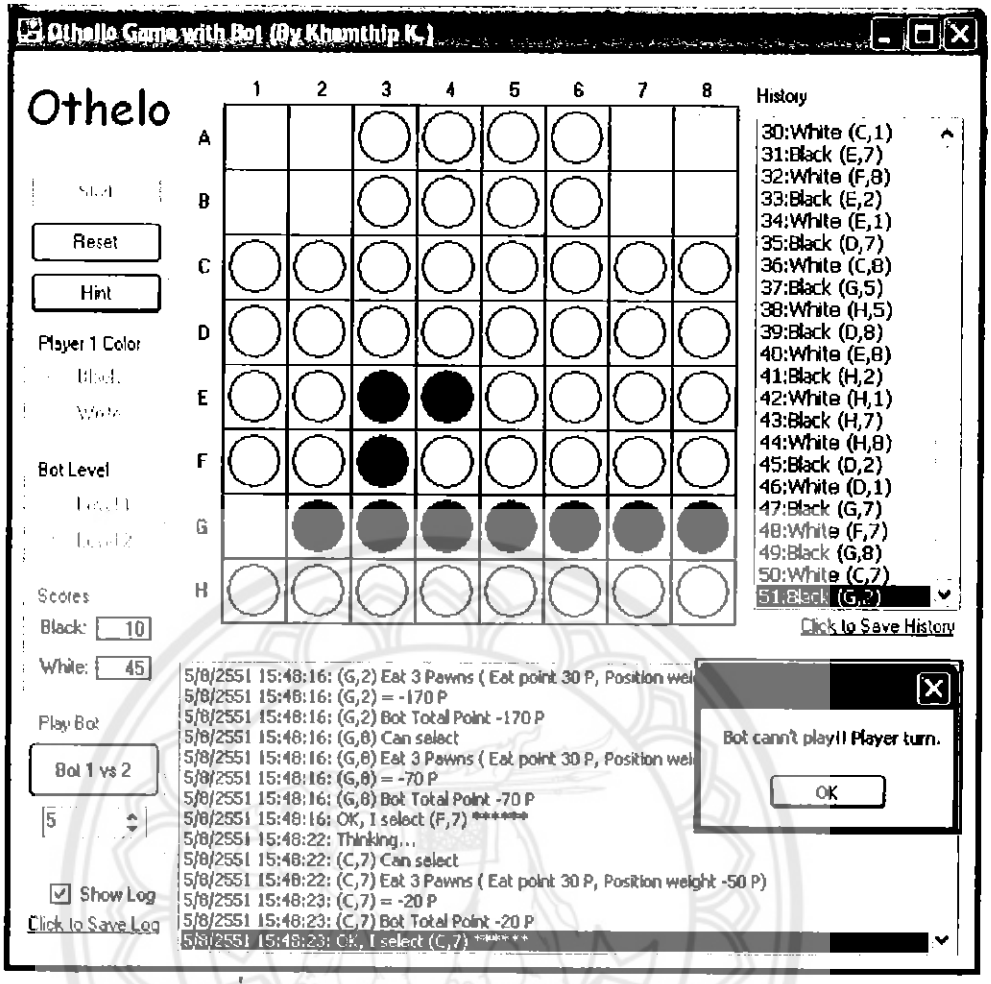

รูปที่ 4.29 ภาพแสดงกระดานของเกม เมื่อเริ่มเกม

เมื่อฝ่ายเบี้ยสีดำเดินเรียบร้อยแล้ว โดยในช่อง History จะแจ้งให้ทราบด้วยว่า เบี้ยสีใดเดินตำแหน่งใดไปบ้าง จากนั้นเบี้ยสีขาวซึ่งในที่นี้คือ ปัญญาประดิษฐ์ ก็จะเริ่มกระบวนการคิดของตัวเอง ซึ่งกระบวนการคิดเหล่านี้ได้อธิบายไว้ก่อนหน้านี้แล้ว และขั้นตอนการคิดนี้ จะแสดงให้ผู้เล่น ได้เห็นตลอดเวลาที่มีการคลิกเลือกให้ระบบ Show Log ซึ่งจะแสดงออกมาในช่อง Show Log ด้านล่างของกระดาน และตรงช่อง Score ก็จะแสดงคะแนนของแต่ฝ่ายให้ตลอด ดังภาพต่อไปนี้



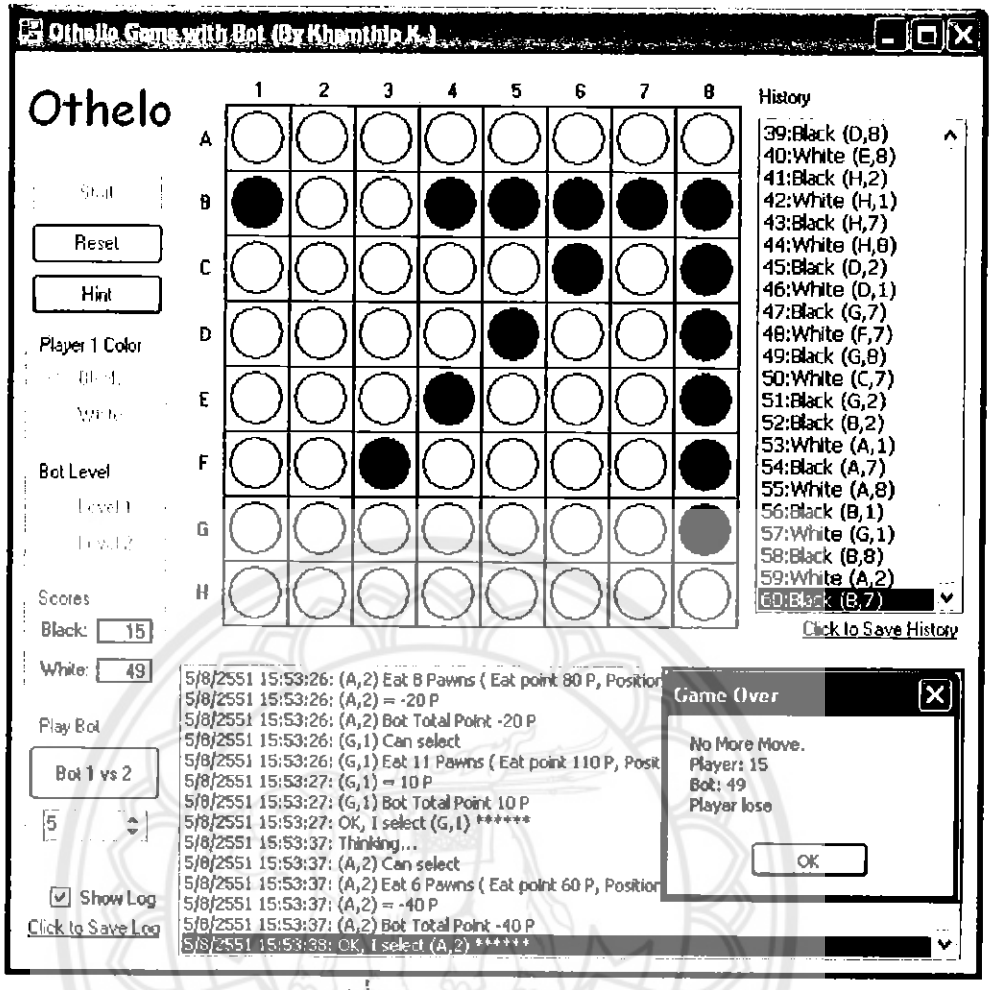

รูปที่ 4.30 ภาพแสดงเมื่อมีการเดินหมาก และมีการแสดง Log ของเกม

เกมดำเนินมาเรื่อยๆ จนกระทั่ง มีรอบที่ ปัญญาประดิษฐ์ไม่สามารถเดินได้ จึงต้องผ่าน 1 รอบ เพื่อให้ผู้เล่นได้เดินต่อ ดังภาพต่อไปนี้




รูปที่ 4.31 ภาพแสดงการผ่านรอบ ของปัญญาประดิษฐ์

เกมดำเนินต่อ จนกระทั่งจบเกมลง โดยผลจบลงที่ ปัญญาประดิษฐ์เป็นฝ่ายชนะไปด้วย คะแนน 49 : 15




รูปที่ 4.32 ภาพแสดงการจบเกม

การแข่งขันดังกล่าวนี้ สามารถตรวจสอบวิธีการคิด และ History ของปัญญาประดิษฐ์ได้ดังต่อไปนี้

ตัวอย่าง History ของเกม

---- Saved on 5/8/2551 16:02:46 ----

| | | | |
|---|---|---|---|
| 00:Game Start | 01:Black (C,4) | 02:White (C,3) | 03:Black (F,5) |
| 04:White (F,6) | 05:Black (E,6) | 06:White (F,4) | 07:Black (G,4) |
| 08:White (F,3) | 09:Black (G,6) | 10:White (C,6) | 11:Black (F,2) |
| 12:White (H,4) | 13:Black (E,3) | 14:White (F,1) | 15:Black (G,3) |
| 16:White (C,5) | 17:Black (B,5) | 18:White (H,3) | 19:Black (B,4) |
| 20:White (A,4) | 21:Black (A,5) | 22:White (A,6) | 23:Black (D,3) |
| 24:White (D,6) | 25:Black (B,6) | 26:White (H,6) | 27:Black (B,3) |
| 28:White (A,3) | 29:Black (C,2) | 30:White (C,1) | 31:Black (E,7) |
| 32:White (F,8) | 33:Black (E,2) | 34:White (E,1) | 35:Black (D,7) |

36:White (C,8) 37:Black (G,5) 38:White (H,5) 39:Black (D,8)

40:White (E,8) 41:Black (H,2) 42:White (H,1) 43:Black (H,7)

44:White (H,8) 45:Black (D,2) 46:White (D,1) 47:Black (G,7)

48:White (F,7) 49:Black (G,8) 50:White (C,7) 51:Black (G,2)

52:Black (B,2) 53:White (A,1) 54:Black (A,7) 55:White (A,8)

56:Black (B,1) 57:White (G,1) 58:Black (B,8) 59:White (A,2)

60:Black (B,7)

Log ของเกม มีดังนี้

ตัวอย่าง Log ของเกม แสดงในช่วงต้นเกม และจบเกม

---- Saved on 5/8/2551 16:02:42 ----

5/8/2551 15:22:27: InitialGame

5/8/2551 15:30:19: Thinking...

5/8/2551 15:30:19: (C,3) Can select

5/8/2551 15:30:19: (C,3) Eat 1 Pawns ( Eat point 10 P, Position weight 100 P)

5/8/2551 15:30:19: (C,3) = 110 P

5/8/2551 15:30:19: (C,3) Bot Total Point 110 P

5/8/2551 15:30:19: (C,5) Can select

5/8/2551 15:30:19: (C,5) Eat 1 Pawns ( Eat point 10 P, Position weight 25 P)

5/8/2551 15:30:19: (C,5) = 35 P

5/8/2551 15:30:19: (C,5) Bot Total Point 35 P

5/8/2551 15:30:19: (E,3) Can select

5/8/2551 15:30:19: (E,3) Eat 1 Pawns ( Eat point 10 P, Position weight 25 P)

5/8/2551 15:30:19: (E,3) = 35 P

5/8/2551 15:30:19: (E,3) Bot Total Point 35 P

5/8/2551 15:30:19: OK, I select (C,3) ******

...

5/8/2551 15:53:37: Thinking...

5/8/2551 15:53:37: (A,2) Can select

5/8/2551 15:53:37: (A,2) Eat 6 Pawns ( Eat point 60 P, Position weight -100 P)

5/8/2551 15:53:37: (A,2) = -40 P

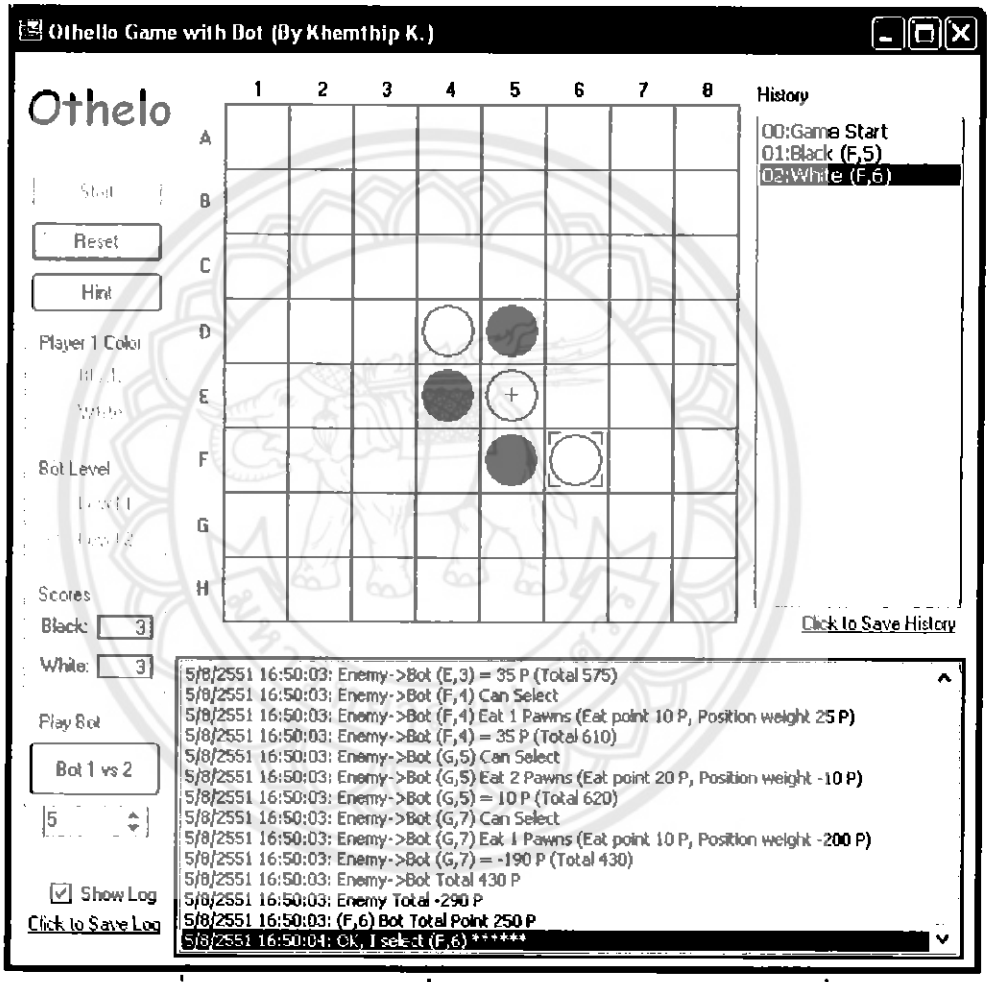
5/8/2551 15:53:37: (A,2) Bot Total Point -40 P

5/8/2551 15:53:38: OK, I select (A,2) ******

### 4.3.2 การทำงานของปัญญาประดิษฐ์ ระดับที่ 2

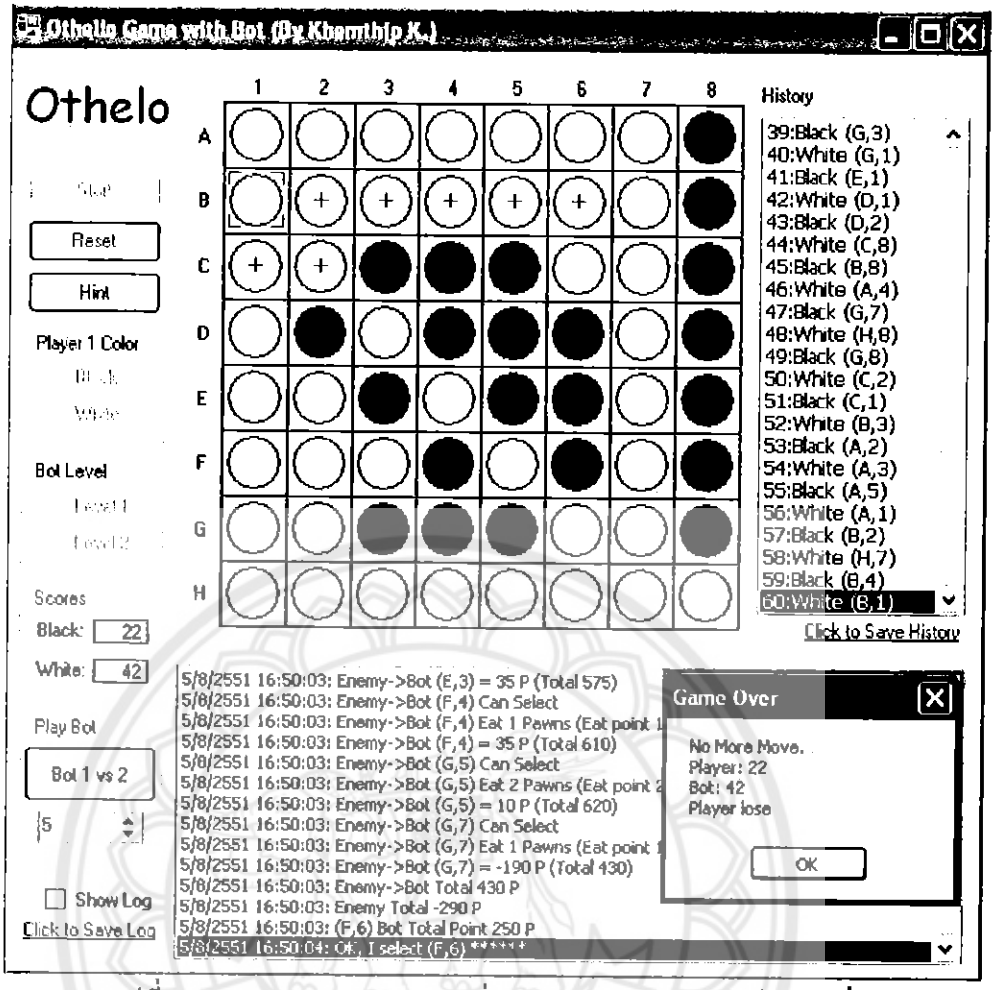
ผลการทดลอง การเล่นหมากกระดาน Othello กับปัญญาประดิษฐ์ ระดับที่ 2 จะแสดงเป็นลำดับ จนกระทั่งจบเกม ดังต่อไปนี้

โดยวิธีการต่างๆ ในการเริ่มเกม จะกระทำเช่นเดียวกับ ที่ทำในการเล่นเกม กับปัญญาประดิษฐ์ ระดับที่ 1 ส่วนที่แตกต่างกันก็คือ กระบวนการคิดของปัญญาประดิษฐ์ ในระดับที่ 2 นี้ จะมีการคิดล่วงหน้า 1 รอบ ดังภาพต่อไปนี้




**รูปที่ 4.33** ภาพแสดง การเริ่มเล่นเกมกับปัญญาประดิษฐ์ระดับที่ 2

การแข่งขันดำเนินต่อ จนกระทั้งเกมจบลง ผลการแข่งขัน ปัญญาประดิษฐ์เป็นฝ่ายชนะ ด้วยผลคะแนน 42 : 22




รูปที่ 4.34 ภาพแสดง การจบเกมเมื่อแข่งกับปัญญาประดิษฐ์ ระดับที่ 2

การแข่งขันดังกล่าวนี้ สามารถตรวจสอบวิธีการคิด และ History ของปัญญาประดิษฐ์ได้ ดังต่อไปนี้

ตัวอย่าง History ของเกม

---- Saved on 5/8/2551 16:58:37 ----

| | | | |
|---|---|---|---|
| 00:Game Start | 01:Black (F,5) | 02:White (F,6) | 03:Black (E,6) |
| 04:White (D,6) | 05:Black (C,6) | 06:White (B,6) | 07:Black (F,7) |
| 08:White (G,6) | 09:Black (E,7) | 10:White (F,8) | 11:Black (H,6) |
| 12:White (G,5) | 13:Black (A,6) | 14:White (H,5) | 15:Black (E,8) |
| 16:White (D,8) | 17:Black (H,4) | 18:White (F,4) | 19:Black (E,3) |
| 20:White (F,3) | 21:Black (G,4) | 22:White (B,7) | 23:Black (C,4) |
| 24:White (C,3) | 25:Black (D,3) | 26:White (C,7) | 27:Black (A,8) |
| 28:White (C,5) | 29:Black (B,5) | 30:White (E,2) | 31:Black (G,2) |
| 32:White (H,1) | 33:Black (F,2) | 34:White (H,3) | 35:Black (H,2) |
| 36:White (A,7) | 37:Black (D,7) | 38:White (F,1) | 39:Black (G,3) |

40:White (G,1) 41:Black (E,1) 42:White (D,1) 43:Black (D,2)

44:White (C,8) 45:Black (B,8) 46:White (A,4) 47:Black (G,7)

48:White (H,8) 49:Black (G,8) 50:White (C,2) 51:Black (C,1)

52:White (B,3) 53:Black (A,2) 54:White (A,3) 55:Black (A,5)

56:White (A,1) 57:Black (B,2) 58:White (H,7) 59:Black (B,4)

60:White (B,1)

Log ของเกม มีดังนี้

ตัวอย่าง Log แสดงในช่วงต้นเกม และช่วงจบเกม

---- Saved on 10/14/2008 4:35:52 PM ----

10/14/2008 4:30:53 PM: InitialGame

10/14/2008 4:31:02 PM: Thinking...

10/14/2008 4:31:02 PM: (C,3) Can select

10/14/2008 4:31:02 PM: (C,3) Eat 1 Pawns ( Eat point -10 P, Position weight 10 P)

10/14/2008 4:31:02 PM: (C,3) = 0 P

10/14/2008 4:31:02 PM: Enemy (C,2) Can Select

10/14/2008 4:31:02 PM: Enemy (C,2) Eat 1 Pawns (Eat point 1 P, Position weight -30 P)

10/14/2008 4:31:02 PM: Enemy (C,2) = -29 P (Total -29)

10/14/2008 4:31:02 PM: Enemy->Bot (B,2) Can Select

10/14/2008 4:31:02 PM: Enemy->Bot (B,2) Eat 1 Pawns (Eat point -1 P, Position weight -80 P)

10/14/2008 4:31:02 PM: Enemy->Bot (B,2) = -81 P (Total -81)

10/14/2008 4:31:02 PM: Enemy->Bot (B,4) Can Select

10/14/2008 4:31:02 PM: Enemy->Bot (B,4) Eat 2 Pawns (Eat point -2 P, Position weight -20 P)

10/14/2008 4:31:02 PM: Enemy->Bot (B,4) = -22 P (Total -103)

10/14/2008 4:31:02 PM: Enemy->Bot (C,5) Can Select

10/14/2008 4:31:02 PM: Enemy->Bot (C,5) Eat 1 Pawns (Eat point -1 P, Position weight -1 P)

10/14/2008 4:31:02 PM: Enemy->Bot (C,5) = -2 P (Total -105)

10/14/2008 4:31:02 PM: Enemy->Bot (D,6) Can Select

10/14/2008 4:31:02 PM: Enemy->Bot (D,6) Eat 1 Pawns (Eat point -1 P, Position weight -1 P)

10/14/2008 4:31:02 PM: Enemy->Bot (D,6) = -2 P (Total -107)

10/14/2008 4:31:02 PM: Enemy->Bot (E,3) Can Select

10/14/2008 4:31:02 PM: Enemy->Bot (E,3) Eat 1 Pawns (Eat point -1 P, Position weight -1 P)

10/14/2008 4:31:02 PM: Enemy->Bot (E,3) = -2 P (Total -109)

10/14/2008 4:31:02 PM: Enemy->Bot (F,4) Can Select

10/14/2008 4:31:02 PM: Enemy->Bot (F,4) Eat 2 Pawns (Eat point -2 P, Position weight -1 P)

10/14/2008 4:31:02 PM: Enemy->Bot (F,4) = -3 P (Total -112)

10/14/2008 4:31:02 PM: Enemy->Bot Total -112 P

10/14/2008 4:31:02 PM: Enemy (D,3) Can Select

10/14/2008 4:31:02 PM: Enemy (D,3) Eat 1 Pawns (Eat point 1 P, Position weight 2 P)

10/14/2008 4:31:02 PM: Enemy (D,3) = 3 P (Total -26)

10/14/2008 4:31:02 PM: Enemy->Bot (C,5) Can Select

10/14/2008 4:31:02 PM: Enemy->Bot (C,5) Eat 2 Pawns (Eat point -2 P, Position weight -1 P)

10/14/2008 4:31:02 PM: Enemy->Bot (C,5) = -3 P (Total -115)

10/14/2008 4:31:02 PM: Enemy->Bot (E,3) Can Select

10/14/2008 4:31:02 PM: Enemy->Bot (E,3) Eat 2 Pawns (Eat point -2 P, Position weight -1 P)

10/14/2008 4:31:02 PM: Enemy->Bot (E,3) = -3 P (Total -118)

10/14/2008 4:31:02 PM: Enemy->Bot Total -118 P

10/14/2008 4:31:02 PM: Enemy (E,6) Can Select

10/14/2008 4:31:02 PM: Enemy (E,6) Eat 1 Pawns (Eat point 1 P, Position weight 2 P)

10/14/2008 4:31:02 PM: Enemy (E,6) = 3 P (Total -23)

10/14/2008 4:31:02 PM: Enemy->Bot (B,4) Can Select

10/14/2008 4:31:02 PM: Enemy->Bot (B,4) Eat 2 Pawns (Eat point -2 P, Position weight -20 P)

10/14/2008 4:31:02 PM: Enemy->Bot (B,4) = -22 P (Total -140)

10/14/2008 4:31:02 PM: Enemy->Bot (C,5) Can Select

10/14/2008 4:31:02 PM: Enemy->Bot (C,5) Eat 1 Pawns (Eat point -1 P, Position weight -1 P)

10/14/2008 4:31:02 PM: Enemy->Bot (C,5) = -2 P (Total -142)

10/14/2008 4:31:02 PM: Enemy->Bot (D,6) Can Select

10/14/2008 4:31:02 PM: Enemy->Bot (D,6) Eat 1 Pawns (Eat point -1 P, Position weight -1 P)

10/14/2008 4:31:02 PM: Enemy->Bot (D,6) = -2 P (Total -144)

10/14/2008 4:31:02 PM: Enemy->Bot (F,4) Can Select

10/14/2008 4:31:02 PM: Enemy->Bot (F,4) Eat 2 Pawns (Eat point -2 P, Position weight -1 P)

10/14/2008 4:31:02 PM: Enemy->Bot (F,4) = -3 P (Total -147)

10/14/2008 4:31:02 PM: Enemy->Bot (F,6) Can Select

10/14/2008 4:31:02 PM: Enemy->Bot (F,6) Eat 1 Pawns (Eat point -1 P, Position weight -1 P)

10/14/2008 4:31:02 PM: Enemy->Bot (F,6) = -2 P (Total -149)

10/14/2008 4:31:02 PM: Enemy->Bot Total -149 P

10/14/2008 4:31:02 PM: Enemy (F,5) Can Select

10/14/2008 4:31:02 PM: Enemy (F,5) Eat 1 Pawns (Eat point 1 P, Position weight 2 P)

10/14/2008 4:31:02 PM: Enemy (F,5) = 3 P (Total -20)

10/14/2008 4:31:02 PM: Enemy->Bot (B,4) Can Select

10/14/2008 4:31:02 PM: Enemy->Bot (B,4) Eat 2 Pawns (Eat point -2 P, Position weight -20 P)

10/14/2008 4:31:02 PM: Enemy->Bot (B,4) = -22 P (Total -171)

10/14/2008 4:31:02 PM: Enemy->Bot (C,5) Can Select

10/14/2008 4:31:02 PM: Enemy->Bot (C,5) Eat 1 Pawns (Eat point -1 P, Position weight -1 P)

10/14/2008 4:31:02 PM: Enemy->Bot (C,5) = -2 P (Total -173)

10/14/2008 4:31:02 PM: Enemy->Bot (D,6) Can Select

10/14/2008 4:31:02 PM: Enemy->Bot (D,6) Eat 1 Pawns (Eat point -1 P, Position weight -1 P)

10/14/2008 4:31:02 PM: Enemy->Bot (D,6) = -2 P (Total -175)

10/14/2008 4:31:02 PM: Enemy->Bot (F,4) Can Select

10/14/2008 4:31:02 PM: Enemy->Bot (F,4) Eat 2 Pawns (Eat point -2 P, Position weight -1 P)

10/14/2008 4:31:02 PM: Enemy->Bot (F,4) = -3 P (Total -178)

10/14/2008 4:31:02 PM: Enemy->Bot (F,6) Can Select

10/14/2008 4:31:02 PM: Enemy->Bot (F,6) Eat 1 Pawns (Eat point -1 P, Position weight -1 P)

10/14/2008 4:31:02 PM: Enemy->Bot (F,6) = -2 P (Total -180)

10/14/2008 4:31:02 PM: Enemy->Bot Total -180 P

10/14/2008 4:31:02 PM: Enemy Total -20 P

10/14/2008 4:31:02 PM: (C,3) Bot Total Point -200 P

...

10/14/2008 4:31:58 PM: Thinking...

10/14/2008 4:31:58 PM: (H,1) Can select

10/14/2008 4:31:58 PM: (H,1) Eat 3 Pawns ( Eat point -30 P, Position weight 200 P)

10/14/2008 4:31:58 PM: (H,1) = 170 P

10/14/2008 4:31:58 PM: Enemy (H,2) Can Select

10/14/2008 4:31:58 PM: Enemy (H,2) Eat 11 Pawns (Eat point 11 P, Position weight 30 P)

10/14/2008 4:31:58 PM: Enemy (H,2) = 41 P (Total 41)

10/14/2008 4:31:58 PM: Enemy->Bot Total 0 P

10/14/2008 4:31:58 PM: Enemy Total 41 P

10/14/2008 4:31:58 PM: (H,1) Bot Total Point 211 P

10/14/2008 4:31:58 PM: (H,2) Can select

10/14/2008 4:31:58 PM: (H,2) Eat 6 Pawns ( Eat point -60 P, Position weight -600 P)

10/14/2008 4:31:58 PM: (H,2) = -660 P

10/14/2008 4:31:58 PM: Enemy (H,1) Can Select

10/14/2008 4:31:58 PM: Enemy (H,1) Eat 11 Pawns (Eat point 11 P, Position weight -100 P)

10/14/2008 4:31:58 PM: Enemy (H,1) = -89 P (Total -89)

10/14/2008 4:31:58 PM: Enemy->Bot Total 0 P

10/14/2008 4:31:58 PM: Enemy Total -89 P

10/14/2008 4:31:58 PM: (H,2) Bot Total Point -749 P

10/14/2008 4:31:58 PM: OK, I select (H,1) ******

## 4.4 ผลการทดลอง การแข่งขันกันของปัญญาประดิษฐ์

### 4.4.1 ผลการทดลองการแข่งขันของปัญญาประดิษฐ์

จากการทดลอง ให้ปัญญาประดิษฐ์ ระดับที่ 1 แข่งกับ ระดับที่ 2 ได้ผลการแข่งขันดังนี้

**ตารางที่ 4.1** แสดงผลการแข่งขันจนจบเกม 30 รอบ รอบละ 10 เกม, 50 เกม, 100 เกม และ 250 เกม ของการแข่งขันระหว่างปัญญาประดิษฐ์ระดับที่ 1 กับ ระดับที่ 2

| รอบที่ | ผลการแข่งขัน AI ระดับที่ 1 ชนะ : เสมอ : AI ระดับที่ 2 ชนะ | | | |
|---|---|---|---|---|
| | 10 เกม | 50 เกม | 100 เกม | 250 เกม |
| 1 | 03-D0-07 | 24-D0-26 | 42-D0-58 | 125-D0-125 |
| 2 | 07-D0-03 | 28-D0-22 | 53-D0-47 | 128-D0-122 |
| 3 | 04-D0-06 | 32-D0-18 | 61-D0-39 | 124-D0-126 |
| 4 | 01-D0-09 | 31-D0-19 | 48-D0-52 | 118-D0-132 |
| 5 | 02-D0-08 | 22-D0-28 | 46-D0-54 | 137-D0-113 |
| 6 | 03-D0-07 | 26-D0-24 | 50-D0-50 | 119-D0-131 |
| 7 | 06-D1-03 | 24-D0-26 | 60-D0-40 | 116-D0-134 |
| 8 | 05-D0-05 | 15-D0-35 | 46-D1-53 | 113-D0-137 |
| 9 | 08-D0-02 | 23-D1-26 | 45-D0-55 | 140-D0-110 |
| 10 | 04-D0-06 | 26-D0-24 | 48-D0-52 | 106-D0-144 |
| 11 | 08-D0-02 | 22-D0-28 | 50-D1-49 | 122-D1-127 |
| 12 | 06-D0-04 | 25-D0-25 | 45-D0-55 | 148-D0-102 |
| 13 | 03-D0-07 | 31-D0-19 | 48-D0-52 | 117-D0-133 |
| 14 | 05-D0-05 | 25-D0-25 | 51-D0-49 | 131-D0-119 |
| 15 | 05-D0-05 | 24-D1-25 | 55-D0-45 | 118-D0-132 |
| 16 | 04-D0-06 | 24-D0-26 | 45-D0-55 | 126-D0-124 |
| 17 | 03-D0-07 | 23-D0-27 | 49-D0-51 | 124-D0-126 |
| 18 | 06-D0-04 | 21-D0-29 | 50-D0-50 | 128-D0-122 |
| 19 | 02-D0-08 | 28-D0-22 | 37-D0-63 | 123-D0-127 |
| 20 | 06-D0-04 | 28-D0-22 | 55-D0-45 | 127-D0-123 |
| 21 | 04-D0-06 | 24-D0-26 | 48-D0-52 | 140-D0-110 |
| 22 | 04-D0-06 | 29-D0-21 | 48-D0-52 | 124-D0-126 |
| 23 | 08-D0-02 | 19-D0-31 | 50-D0-50 | 125-D0-132 |
| 24 | 06-D0-04 | 25-D0-25 | 53-D0-47 | 125-D0-125 |
| 25 | 03-D0-07 | 25-D0-25 | 44-D0-56 | 143-D0-107 |
| 26 | 05-D0-05 | 29-D0-21 | 54-D0-46 | 128-D0-122 |
| 27 | 05-D0-05 | 21-D0-29 | 49-D0-51 | 125-D0-132 |
| 28 | 04-D0-06 | 19-D0-31 | 55-D0-45 | 128-D0-122 |
| 29 | 07-D0-03 | 19-D0-31 | 56-D0-44 | 123-D0-127 |
| 30 | 03-D0-07 | 24-D0-26 | 49-D0-51 | 121-D0-129 |

จากตารางที่แสดงผลการแข่งขัน ระหว่างปัญญาประดิษฐ์ ระดับที่ 1 กับ ระดับที่ 2 การทดลองได้ทำการจัดให้ปัญญาประดิษฐ์แข่งกัน 30 รอบ โดยแต่ละรอบ แข่งกันรอบละ 10 เกม, 50 เกม, 100 เกม และ 250 เกม ผลลัพธ์ในตาราง เช่น 03-D0-07 หมายถึง AI ระดับที่ 1 ชนะ 3 เกม, เสมอกัน 0 เกม และ AI ระดับที่ 2 ชนะ 7 เกม

ตารางที่ 4.2 แสดงการสรุปผล 12,300 เกม ของการแข่งขันระหว่างปัญญาประดิษฐ์

| ผลการแข่งขัน 12,300 เกม | | | |
|---|---|---|---|
| รายการ | AI 1 win | Draw | AI 2 win |
| ผลการแข่งขัน | 5744 | 6 | 6550 |
| คิดเป็นร้อยละ | 46.7 | 0.1 | 53.2 |

จากตารางจะพบว่า การแข่งขันระหว่างปัญญาประดิษฐ์ทั้งสองระดับนั้น คิดเป็นอัตราร้อยละได้ ดังนี้

รวมแข่งขันทั้งหมด 12,300 เกม AI ระดับที่ 1 ชนะร้อยละ 46.7 AI ระดับที่ 2 ชนะร้อยละ 53.2

จากผลแข่งขันกัน ปัญญาประดิษฐ์ระดับที่ 1 ชนะร้อยละ 46.7 และปัญญาประดิษฐ์ระดับที่ 2 ชนะร้อยละ 53.2 จากผลการทดลองนี้ จึงสรุปได้ว่าปัญญาประดิษฐ์ระดับที่ 2 เก่งกว่า

4.4.2 ผลการทดลองการแข่งขันของปัญญาประดิษฐ์ ประเด็นฝ่ายเริ่มเกมก่อน

ประเด็นหนึ่งในการแข่งขัน ที่เกี่ยวกับการสลับกันเล่นนั้น คือ การที่ผู้เริ่มเล่นเกมก่อน จะมีความได้เปรียบฝ่ายตรงข้ามหรือไม่ จึงเกิดการทดลองในส่วนนี้ขึ้น เพื่อพิสูจน์ว่าฝ่ายเริ่มเกมก่อนมีความได้เปรียบหรือไม่ โดยการทดลองนี้ ได้กำหนดให้ มีการแข่งขัน 20 รอบๆ ละ 20 เกม และ 50 เกมขึ้น โดยกำหนดให้มีการแข่งขันกันดังนี้คือ AI 1 vs. AI 1, AI 2 vs. AI 2, AI 1 vs AI 2 (โดย AI 1 เริ่มเกมก่อน) และ AI 1 vs AI 2 (โดย AI 2 เริ่มเกมก่อน) และการทดลองดังกล่าวได้ผลการทดลองออกมา ตามตารางดังนี้

**ตารางที่ 4.3** แสดงผลการทดลองการแข่งขันระหว่าง AI 1 vs. AI 1

| รอบที่ | ผลการแข่งขัน AI 1 : Draw : AI 1 | |
|---|---|---|
| | 20 เกม | 50 เกม |
| 1 | 10-D1-09 | 26-D0-24 |
| 2 | 11-D1-08 | 17-D2-31 |
| 3 | 09-D1-10 | 24-D3-23 |
| 4 | 07-D1-12 | 22-D2-26 |
| 5 | 13-D0-07 | 26-D5-19 |
| 6 | 09-D0-11 | 23-D1-26 |
| 7 | 11-D1-08 | 28-D3-19 |
| 8 | 10-D1-09 | 19-D6-25 |
| 9 | 10-D1-09 | 28-D1-21 |
| 10 | 10-D0-10 | 24-D2-24 |
| 11 | 09-D0-11 | 23-D0-27 |
| 12 | 09-D0-11 | 24-D4-22 |
| 13 | 10-D2-08 | 21-D1-28 |
| 14 | 08-D1-11 | 30-D2-18 |
| 15 | 11-D0-09 | 20-D2-28 |
| 16 | 07-D2-11 | 24-D0-26 |
| 17 | 08-D0-12 | 25-D3-22 |
| 18 | 09-D1-10 | 26-D3-21 |
| 19 | 15-D0-05 | 25-D0-25 |
| 20 | 11-D1-08 | 25-D5-20 |
| สรุป | 10 - D0 - 9 | 10 - D2 - 8 |

**ตารางที่ 4.4** แสดงผลการทดลองการแข่งขันระหว่าง AI 2 vs. AI 2

| รอบที่ | ผลการแข่งขัน AI 2 : Draw : AI 2 | |
|---|---|---|
| | 20 เกม | 50 เกม |
| 1 | 20-D0-00 | 50-D0-00 |
| 2 | 20-D0-00 | 50-D0-00 |
| 3 | 20-D0-00 | 50-D0-00 |
| 4 | 20-D0-00 | 50-D0-00 |
| 5 | 20-D0-00 | 50-D0-00 |
| 6 | 20-D0-00 | 50-D0-00 |
| 7 | 20-D0-00 | 50-D0-00 |
| 8 | 20-D0-00 | 50-D0-00 |
| 9 | 20-D0-00 | 50-D0-00 |
| 10 | 20-D0-00 | 50-D0-00 |
| 11 | 20-D0-00 | 50-D0-00 |
| 12 | 20-D0-00 | 50-D0-00 |
| 13 | 20-D0-00 | 50-D0-00 |
| 14 | 20-D0-00 | 50-D0-00 |
| 15 | 20-D0-00 | 50-D0-00 |
| 16 | 20-D0-00 | 50-D0-00 |
| 17 | 20-D0-00 | 50-D0-00 |
| 18 | 20-D0-00 | 50-D0-00 |
| 19 | 20-D0-00 | 50-D0-00 |
| 20 | 20-D0-00 | 50-D0-00 |
| สรุป | 20 - D0 - 0 | 20 - D0 - 0 |

ตารางที่ 4.5 แสดงผลการทดลองการแข่งขันระหว่าง AI 1 vs. AI 2 (AI 1 เริ่มเกมก่อน)

| รอบที่ | ผลการแข่งขัน AI 1 : Draw : AI 2 | |
|---|---|---|
| | 20 เกม | 50 เกม |
| 1 | 13-D0-07 | 39-D0-11 |
| 2 | 15-D0-05 | 34-D0-16 |
| 3 | 14-D0-06 | 37-D0-13 |
| 4 | 15-D0-05 | 33-D0-17 |
| 5 | 14-D0-06 | 33-D0-17 |
| 6 | 16-D0-04 | 35-D0-15 |
| 7 | 14-D0-06 | 37-D0-13 |
| 8 | 15-D0-05 | 34-D0-16 |
| 9 | 16-D0-04 | 34-D0-16 |
| 10 | 14-D0-06 | 37-D0-13 |
| 11 | 14-D0-06 | 29-D0-21 |
| 12 | 13-D0-07 | 37-D0-13 |
| 13 | 13-D0-07 | 31-D0-19 |
| 14 | 15-D0-05 | 36-D0-14 |
| 15 | 14-D0-06 | 33-D1-16 |
| 16 | 15-D0-05 | 36-D0-14 |
| 17 | 13-D0-07 | 37-D0-13 |
| 18 | 15-D0-05 | 29-D0-21 |
| 19 | 16-D0-04 | 37-D0-13 |
| 20 | 14-D0-06 | 31-D0-19 |
| สรุป | 20 – D0 - 0 | 20 – D0 - 0 |

ตารางที่ 4.6 แสดงผลการทดลองการแข่งขันระหว่าง AI 1 vs. AI 2 (AI 2 เริ่มเกมก่อน)

| รอบที่ | ผลการแข่งขัน AI 1 : Draw : AI 2 | |
|---|---|---|
| | 20 เกม | 50 เกม |
| 1 | 07-D0-13 | 16-D0-34 |
| 2 | 05-D0-15 | 18-D0-32 |
| 3 | 06-D0-14 | 14-D0-36 |
| 4 | 04-D1-15 | 12-D0-38 |
| 5 | 08-D0-12 | 14-D0-36 |
| 6 | 09-D0-11 | 12-D0-38 |
| 7 | 04-D0-16 | 18-D0-32 |
| 8 | 07-D0-13 | 10-D0-40 |
| 9 | 03-D0-17 | 14-D0-36 |
| 10 | 05-D0-15 | 18-D0-32 |
| 11 | 07-D0-13 | 16-D0-34 |
| 12 | 05-D0-15 | 18-D0-32 |
| 13 | 06-D0-14 | 14-D0-36 |
| 14 | 04-D1-15 | 12-D0-38 |
| 15 | 08-D0-12 | 14-D0-36 |
| 16 | 09-D0-11 | 12-D0-38 |
| 17 | 04-D0-16 | 18-D0-32 |
| 18 | 07-D0-13 | 10-D0-40 |
| 19 | 03-D0-17 | 14-D0-36 |
| 20 | 05-D0-15 | 18-D0-32 |
| สรุป | 0 - D0 – 20 | 0 – D0 - 20 |

จากตารางผลการทดลองดังกล่าวนั้น สามารถสรุปผลได้ดังนี้

กรณี AI 1 vs. AI 1 นั้นผลการแข่งขันสำหรับฝ่ายเริ่มเกมก่อนชนะร้อยละ 50

กรณี AI 2 vs. AI 2 นั้นผลการแข่งขันสำหรับฝ่ายเริ่มเกมก่อนชนะร้อยละ 100

กรณี AI 1 vs. AI 2 (AI 1 เริ่มก่อน) ฝ่ายเริ่มเกมก่อนชนะร้อยละ 100

กรณี AI 1 vs. AI 2 (AI 2 เริ่มก่อน) ฝ่ายเริ่มเกมก่อนชนะร้อยละ 100

จากผลการทดลองดังกล่าวนี้ จึงสามารถสรุปได้ว่า โปรแกรมโอเทลโล่เกมที่พัฒนานี้ เมื่อทำการทดลองแข่งกันเองของปัญญาประดิษฐ์ ทั้ง 2 ระดับ ฝ่ายที่เริ่มเล่นก่อนจะเป็นฝ่ายที่มีโอกาสชนะเกม จึงถือว่าฝ่ายเริ่มก่อนได้เปรียบ

## 4.5 ผลการทดลอง การแข่งขันระหว่างผู้เล่นกับปัญญาประดิษฐ์

### 4.5.1 ผลการทดลอง การแข่งขันระหว่างผู้เล่นกับปัญญาประดิษฐ์ ระดับที่ 1

โดยการทดลองนี้ใช้ผู้เล่น 15 คนที่ไม่ใช่ผู้พัฒนาโปรแกรม และมีระดับความสามารถในการเล่นเกมโอเทลโล 3 ระดับ ระดับละ 5 คน คือ

ระดับ a (หมายถึงคนที่เล่นเกมโอเทลโล่ แบบเล็กน้อยเคยเล่นไม่เกิน 10 ครั้ง)

ระดับ b (หมายถึงคนที่เล่นเกมโอเทลโล่ แบบธรรมดา เคยเล่นระหว่าง 10 – 50 ครั้ง)

ระดับ c (หมายถึงคนที่เล่นเกมโอเทลโล่ แบบจริงจังเพื่อการแข่งขัน เคยเล่น 50 ครั้งขึ้นไป)

ทำการทดลองเล่นเกมกับปัญญาประดิษฐ์ ระดับที่ 1 จำนวนรอบ คนละ 20 รอบ ได้ผลการทดลองออกมาดังตาราง

**ตารางที่ 4.7** แสดงผลการแข่งขัน 20 รอบ ระหว่างผู้เล่นกับปัญญาประดิษฐ์ ระดับที่ 1

| ผู้เล่น | การแข่งขัน 20 รอบ ระหว่างผู้เล่น กับปัญญาประดิษฐ์ระดับที่ 1 | | |
|---|---|---|---|
| | ผู้เล่นชนะ | เสมอ | ผู้เล่นแพ้ |
| ผู้เล่นระดับ a | | | |
| คนที่ 1 | 2 | 0 | 18 |
| คนที่ 2 | 3 | 0 | 17 |
| คนที่ 3 | 1 | 0 | 19 |
| คนที่ 4 | 4 | 0 | 16 |
| คนที่ 5 | 1 | 0 | 19 |
| ผู้เล่นระดับ b | | | |
| คนที่ 1 | 11 | 0 | 9 |
| คนที่ 2 | 9 | 0 | 11 |
| คนที่ 3 | 11 | 0 | 9 |
| คนที่ 4 | 9 | 0 | 11 |
| คนที่ 5 | 12 | 0 | 8 |
| ผู้เล่นระดับ c | | | |
| คนที่ 1 | 18 | 0 | 2 |
| คนที่ 2 | 19 | 0 | 1 |
| คนที่ 3 | 20 | 0 | 0 |
| คนที่ 4 | 18 | 0 | 2 |
| คนที่ 5 | 20 | 0 | 0 |

จากตารางเป็นการแสดงผลการทดลอง ของผู้เล่นระดับ a, ระดับ b และ ระดับ c

**ตารางที่** 4.8 แสดงการสรุปผล 20 รอบ ของการแข่งขันระหว่างผู้เล่นกับปัญญาประดิษฐ์ ระดับที่ 1

| ผู้เล่น | ผลแพ้ชนะคิดเป็นร้อยละ | |
|---|---|---|
| | ผู้เล่นชนะ | ผู้เล่นแพ้ |
| ผู้เล่นระดับ a | ร้อยละ 11 | ร้อยละ 89 |
| ผู้เล่นระดับ b | ร้อยละ 52 | ร้อยละ 48 |
| ผู้เล่นระดับ c | ร้อยละ 95 | ร้อยละ 5 |

จากตารางจะพบว่า ในผู้เล่นระดับ a ปัญญาประดิษฐ์ระดับที่ 1 สามารถชนะได้มากกว่า ถึงร้อยละ 89 ในผู้เล่นระดับ b อัตราการชนะระหว่างปัญญาประดิษฐ์ระดับที่ 1 กับผู้เล่น ใกล้เคียงกัน และในผู้เล่นระดับ c ปัญญาประดิษฐ์ระดับที่ 1 สามารถชนะได้เพียง ร้อยละ 5 เท่านั้น จากผลการทดลองทำให้สรุปได้ว่า ปัญญาประดิษฐ์นี้ มีความสามารถอยู่ในระดับปานกลาง

**4.5.2 ผลการทดลองการแข่งขันระหว่างผู้เล่นกับปัญญาประดิษฐ์ ระดับที่ 2**

โดยการทดลองนี้ใช้ผู้เล่น 6 คนที่ไม่ใช่ผู้พัฒนาโปรแกรม และมีระดับความสามารถในการเล่นเกมโอเทลโล 3 ระดับ ระดับละ 2 คน คือ

ระดับ a (หมายถึงคนที่เล่นเกมโอเทลโล่ แบบเล็กน้อยเคยเล่นไม่เกิน 10 ครั้ง)

ระดับ b (หมายถึงคนที่เล่นเกมโอเทลโล่ แบบธรรมดา เคยเล่นระหว่าง 10 – 50 ครั้ง)

ระดับ c (หมายถึงคนที่เล่นเกมโอเทลโล่ แบบจริงจังเพื่อการแข่งขัน เคยเล่น 50 ครั้งขึ้นไป)

ทำการทดลองเล่นเกมกับปัญญาประดิษฐ์ ระดับที่ 2 จำนวนรอบ คนละ 20 รอบ ได้ผลการทดลองออกมาดังตาราง

**ตารางที่** 4.9 แสดงผลการแข่งขันจนจบเกม 20 รอบ ระหว่างผู้เล่นกับปัญญาประดิษฐ์ ระดับที่ 2

| ผู้เล่น | การแข่งขัน 20 รอบ ระหว่างผู้เล่น กับปัญญาประดิษฐ์ระดับที่ 2 | | |
|---|---|---|---|
| | ผู้เล่นชนะ | เสมอ | ผู้เล่นแพ้ |
| ผู้เล่นระดับ a | | | |
| คนที่ 1 | 3 | 0 | 17 |
| คนที่ 2 | 2 | 0 | 18 |
| คนที่ 3 | 1 | 0 | 19 |
| คนที่ 4 | 2 | 0 | 18 |
| คนที่ 5 | 1 | 0 | 19 |
| ผู้เล่นระดับ b | | | |
| คนที่ 1 | 9 | 0 | 11 |
| คนที่ 2 | 10 | 0 | 10 |
| คนที่ 3 | 11 | 0 | 9 |
| คนที่ 4 | 10 | 0 | 10 |
| คนที่ 5 | 8 | 0 | 12 |
| ผู้เล่นระดับ c | | | |
| คนที่ 1 | 19 | 0 | 1 |
| คนที่ 2 | 19 | 0 | 1 |
| คนที่ 3 | 17 | 0 | 3 |
| คนที่ 4 | 19 | 0 | 1 |
| คนที่ 5 | 19 | 0 | 1 |

จากตารางเป็นการแสดงผลการทดลอง ของผู้เล่นระดับ a, ระดับ b และ ระดับ c

**ตารางที่ 4.10** แสดงการสรุปผล 20 รอบ ของการแข่งขันระหว่างผู้เล่นกับปัญญาประดิษฐ์ ระดับที่ 2

| ผู้เล่น | ผลแพ้ชนะคิดเป็นร้อยละ | |
|---|---|---|
| | ผู้เล่นชนะ | ผู้เล่นแพ้ |
| ผู้เล่นระดับ a | ร้อยละ 9 | ร้อยละ 91 |
| ผู้เล่นระดับ b | ร้อยละ 48 | ร้อยละ 52 |
| ผู้เล่นระดับ c | ร้อยละ 93 | ร้อยละ 7 |

จากตารางจะพบว่า ในผู้เล่นระดับ a ปัญญาประดิษฐ์ระดับที่ 2 สามารถชนะได้มากกว่า ถึงร้อยละ 91 ในผู้เล่นระดับ b อัตราการชนะระหว่างปัญญาประดิษฐ์ระดับที่ 2 กับผู้เล่น ใกล้เคียงกัน และในผู้เล่นระดับ c ปัญญาประดิษฐ์ระดับที่ 2 สามารถชนะได้ ร้อยละ 7 จากผลการทดลองทำให้สรุปได้ว่า ปัญญาประดิษฐ์นี้ มีความสามารถอยู่ในระดับปานกลาง

เมื่อเปรียบเทียบผลการแข่งขัน ระหว่างผู้เล่นกับปัญญาประดิษฐ์ทั้ง 2 ระดับ จะได้ผลดังนี้

| ผู้เล่น | ผลแพ้ชนะคิดเป็นร้อยละ | |
|---|---|---|
| | ผู้เล่นชนะ | ผู้เล่นแพ้ |
| ผู้เล่นระดับ a | ร้อยละ 11 | ร้อยละ 89 |
| ผู้เล่นระดับ b | ร้อยละ 52 | ร้อยละ 48 |
| ผู้เล่นระดับ c | ร้อยละ 95 | ร้อยละ 5 |

| ผู้เล่น | ผลแพ้ชนะคิดเป็นร้อยละ | |
|---|---|---|
| | ผู้เล่นชนะ | ผู้เล่นแพ้ |
| ผู้เล่นระดับ a | ร้อยละ 9 | ร้อยละ 91 |
| ผู้เล่นระดับ b | ร้อยละ 48 | ร้อยละ 52 |
| ผู้เล่นระดับ c | ร้อยละ 93 | ร้อยละ 7 |

จากการเปรียบเทียบผลการแข่งขัน ของทั้ง 2 ระดับจะพบว่า ปัญญาประดิษฐ์ในระดับที่ 2 นั้นมีอัตราการเล่นชนะผู้เล่น ได้มากกว่า ในผู้เล่นทั้ง 3 ระดับ จากผลการทดลองจึงสรุปได้ว่า AI 2 นั้นเก่งกว่า AI 1

# บทที่ 5
# บทสรุป

จากการทำการทดลองเล่นเกม Othello Game โดยใช้ปัญญาประดิษฐ์ที่พัฒนาขึ้นโปรแกรมได้ทำการตัดสินใจเลือกเดินในตำแหน่ง ต่างๆ ที่ได้เปรียบต่อปัญญาประดิษฐ์ ได้อย่างถูกต้อง โดยอาศัยหลักการและทฤษฎีที่ประยุกต์คิดค้นขึ้น เพื่อเป็นแนวทางการตัดสินใจของปัญญาประดิษฐ์ เช่น หลักการให้คะแนนจากการกินเบี้ยคู่แข่งแต่ละตัว หลักการให้คะแนนแต่ละตำแหน่ง ที่ต้องเดินเบี้ยในตำแหน่งนั้น เป็นต้น โดยทฤษฎีดังกล่าวนำมาพัฒนากระบวนการขั้นตอนการทำงานเพื่อให้การตัดสินใจได้อย่างถูกต้อง เช่น การตัดสินใจเดินในตำแหน่งที่ได้เปรียบ หรือการไม่เดินในตำแหน่งที่ทำให้เสียเปรียบ และการกำหนดการคิดล่วงหน้า 1 รอบตาเดินเพื่อตรวจสอบความได้เปรียบ ของแต่ละตำแหน่ง เป็นต้น ซึ่งผลการทำงานของปัญญาประดิษฐ์ที่ทำการพัฒนาขึ้นสามารถทำงานตามเป้าหมายที่วางไว้ โดยสามารถสรุปการทำงานได้ดังต่อไปนี้

## 5.1 สรุปผลการทดลอง

### 5.1.1 ผลการทดลอง ผู้เล่นแข่งขันกับปัญญาประดิษฐ์ ระดับที่ 1

จากการทดลอง แข่งขันเกม Othello Game กับปัญญาประดิษฐ์ ระดับที่ 1 ซึ่ง ในที่นี้หมายถึงการคิดของปัญญาประดิษฐ์ในระดับที่ 1 นี้ จะทำการคิดเพียงระดับดับเดียว คือ เมื่อถึงรอบเดินก็จะคำนวณแค่ว่า เดินได้ตำแหน่งใดบ้าง และแต่ละตำแหน่งกินหมากคู่แข่งได้กี่ตัว และเมื่อกินแล้ว ตัวหมากของตนจะไปลงตรงตำแหน่งใด และตำแหน่งนั้นมีคะแนนเท่าไร หลังจากนั้นจะทำการรวมคะแนนเหล่านี้ทั้งหมด แล้วตัดสินใจเดินในตำแหน่งที่ได้คะแนนรวมมากที่สุด โดยจากการทดลองเล่น พบว่า ปัญญาประดิษฐ์สามารถตัดสินใจเดินในตำแหน่งต่างๆ ได้อย่างถูกต้อง จนจบการแข่งขัน

### 5.1.2 ผลการทดลอง ผู้เล่นแข่งขันกับปัญญาประดิษฐ์ ระดับที่ 2

จากการทดลอง แข่งขันเกม Othello Game กับปัญญาประดิษฐ์ ระดับที่ 2 ซึ่ง ในที่นี้หมายถึงการคิดของปัญญาประดิษฐ์ในระดับที่ 2 นี้ จะทำการคิดล่วงหน้า 1 รอบ คือ เมื่อถึงรอบเดินก็จะคำนวณว่า เดินตำแหน่งใดได้บ้าง แล้วเอาแต่ละตำแหน่งเหล่านี้ มาทำการจำลองการเดินเพื่อคิดคะแนน โดยจำลองว่าเมื่อตนเดินแล้ว (คะแนนที่ 1) คู่แข่งเดินตำแหน่งใดได้บ้าง (คะแนนที่ 2) แล้วหลังจากนั้น ตนเดินตำแหน่งใดได้บ้าง (คะแนนที่ 3 ) เมื่อได้คะแนนครบทั้ง 3 ส่วนแล้ว จึงจะนำคะแนนมารวมกัน แล้วตรวจสอบว่าตำแหน่งใด ได้คะแนนรวมมากที่สุด ก็จะตัดสินใจเดินในตำแหน่งนั้น โดยจากการทดลองเล่น พบว่า ปัญญาประดิษฐ์สามารถตัดสินใจเดินในตำแหน่งต่างๆ ได้อย่างถูกต้อง จนจบการแข่งขัน

**5.1.3 ผลทดลอง การแข่งขันกันระหว่างปัญญาประดิษฐ์ ระดับที่ 1 กับ ระดับที่ 2**

จากการทดลอง การแข่งขันเกม Othello Game โดยการกำหนดให้ ปัญญาประดิษฐ์ ระดับที่ 1 แข่งกับ ระดับที่ 2 โดยผลที่ได้ออกมานั้นปัญญาประดิษฐ์ สามารถแข่งขันกันจนจบเกมได้ อย่างถูกต้อง ไม่มีปัญหา และผลการแข่งขันนั้น ผลัดกันแพ้ ผลัดกันชนะ โดยภาพรวมปัญญาประดิษฐ์ในระดับที่ 2 สามารถเอาชนะ ระดับที่ 1 ได้มากกว่า

## 5.2 ปัญหาและอุปสรรค

1. ปัญหาการตรวจสอบตำแหน่งที่สามารถเดินหมากได้ การแก้ปัญหาคือ จะต้องเช็คก่อนว่าตำแหน่งเป้าหมายเป็นตำแหน่งว่าง และตำแหน่งอื่นๆ รอบตำแหน่งเป้าหมายว่าจะต้องเป็นหมากของฝ่ายตรงกันข้ามหมากเดียวหรือหลายหมากแล้วจะต้องไปเจอหมากของตนด้วย ตำแหน่งนั้นจึงจะสามารถเดินหมากได้ และทำการเช็คแบบเดียวกันนี้ ให้ครบทั้ง 8 ทิศทางรอบตำแหน่งเป้าหมาย
2. การตัดสินใจเดินของปัญญาประดิษฐ์ทั้ง 2 ระดับ ในบางครั้งอาจจะตัดสินใจเดินในตำแหน่งที่ไม่ใช่ตำแหน่งที่ได้เปรียบที่สุด สาเหตุมาจากผลคะแนนรวม หมายความว่าหากมีตำแหน่งบริเวณขอบกระดานซึ่งเป็นตำแหน่งที่ได้เปรียบ คือเมื่อเดินแล้วสามารถครอบครองตำแหน่งบริเวณขอบได้ทั้งแถว แต่ปัญญาประดิษฐ์กลับตัดสินใจเดินหมากในตำแหน่งกลางกระดาน ที่ไม่ทำให้เกมได้เปรียบอันเนื่องมาจาก ตำแหน่งที่เดินนี้มีผลคะแนนรวม มากกว่าตำแหน่งได้เปรียบที่ขอบกระดาน ดังที่กล่าวมา ต้องมีการปรับน้ำหนักคะแนนให้เหมาะสม
3. น้ำหนักคะแนนของการกินหมาก และแต่ละตำแหน่งบนกระดาน ยังไม่ใช่คะแนนที่ดีที่สุด และไม่ใช่วิธีการที่ดีที่สุดในการทำโครงงานนี้ เพราะจะเห็นได้ว่า ความสามารถของปัญญาประดิษฐ์นั้น ยังไม่สามารถเอาชนะผู้เล่นที่มีประสบการณ์ได้
4. โปรแกรมไม่สามารถการันตีได้ว่า จะสามารถชนะคู่แข่งได้ทุกครั้ง
5. การตอบสนองของโปรแกรม จะยิ่งช้ามากขึ้น ถ้ามีในขณะที่เล่นจะต้องมีการ Show Log ขนาดใหญ่ วิธีการแก้ปัญหาคือทำการปิดการ Show Log ไปก่อนจนจบการแข่งขัน แล้วหากต้องการ ตรวจสอบ Log ต่างๆ นั้น ให้ทำการกด save เมื่อจบเกมได้

## 5.3 ข้อเสนอแนะ

จากการพัฒนาปัญญาประดิษฐ์ที่ใช้ในการตัดสินใจเดินหมาก ในตำแหน่งที่ดีและได้เปรียบที่สุด เพื่อให้ปัญญาประดิษฐ์ชนะการแข่งขันนั้น โปรแกรมปัญญาประดิษฐ์นี้ยังสามารถนำไปพัฒนาต่อได้ ดังนี้

1. สามารถปรับเปลี่ยน คะแนนการกินหมากคู่แข่ง และคะแนนตำแหน่งที่เดิน เพื่อให้การติดสินใจของปัญญา ประดิษฐ์ มีความได้เปรียบ และมีประสิทธิภาพมากขึ้น

2. สามารถเพิ่มเติม ฟังก์ชันพิเศษเพื่อการตัดสินใจเดิน ในตำแหน่งที่จำเป็นต้องเดิน (หมายถึงตำแหน่งที่แม้จะไม่ใช่ตำแหน่งที่มีผลคะแนนรวมมากที่สุด แต่เมื่อเดินแล้วสามารถทำให้ปัญญาประดิษฐ์สามารถครอบครองตำแหน่งขอบกระดานที่ได้เปรียบ ก็จะต้องเดิน) ถึงแม้ว่าจะขัดกับหลักการใน ข้อที่ 1 ก็ตาม (เพราะหลักการข้อที่ 1 นั้นจะให้ความสำคัญในการตัดสินใจเดินหมากกับคะแนนรวมมากที่สุด) เพื่อครองความได้เปรียบของการแข่งขัน

3. สามารถนำ Source Code ของโปรแกรมไปดัดแปลง เพื่อทำเป็นเกมหมากกระดานอื่นๆ ที่มีลักษณะการเล่น คล้ายๆ Othello Game ได้

## 5.4 สรุป

โครงงานนี้ออกแบบเพื่อ ทำการพัฒนาตัวเกมระบบปัญญาประดิษฐ์สำหรับเกมโอเทลโล่ โดยใช้ภาษา C# ในการพัฒนา ตลอดระยะเวลาการพัฒนา ได้ทำการศึกษาทฤษฎีและหลักการเกี่ยวกับปัญญาประดิษฐ์ ประเภทต่างๆ ว่ามีการคิด การทำงานในลักษณะใด โดยได้รวบรวมความรู้จากเอกสารและสื่อต่างๆ ทำให้มีความรู้ ความเข้าใจเกี่ยวกับปัญญาประดิษฐ์เพิ่มขึ้น ในระดับหนึ่ง และได้ทำการศึกษาเรียนรู้ทฤษฎีและหลักการ เขียนโปรแกรมด้วยภาษา C# ทำให้มีความรู้ความเข้าใจเพิ่มขึ้นในระดับที่สามารถพัฒนาโปรแกรมในโครงงานได้

ผลลัพธ์ที่ได้จากการพัฒนาโปรแกรมเกมและปัญญาประดิษฐ์ของเกมโอเทลโล่นั้น โปรแกรมสามารถเล่นตอบสนองผู้เล่น และสามารถตัดสินใจเดินหมาก ในตำแหน่งที่ได้เปรียบจากการคำนวณอยู่เสมอ เพื่อเอาชนะการแข่งขันให้ได้ โดยกระบวนการคิดของปัญญาประดิษฐ์นี้ ทำออกมา 2 แบบ คือ ปัญญาประดิษฐ์ระดับที่ 1 ที่สามารถตอบสนองการเล่นแบบตรงไปตรงมา และปัญญาประดิษฐ์ระดับที่ 2 ที่สามารถคาดเดาการเดินล่วงหน้า ได้ 1 ระดับเพื่อคำนวณคะแนน ซึ่งปัญญาประดิษฐ์ทั้ง 2 ระดับสามารถตอบสนองการแข่งขัน กับผู้เล่นได้อย่างถูกต้องตาม algorithm ที่กำหนดไว้ การทดสอบเพื่อหาน้ำหนักคะแนนที่เหมาะสมนั้นทำให้ได้ความรู้ในด้าน การคิดล่วงหน้านั้นทำให้ได้เปรียบ และการเป็นฝ่ายเดินก่อนก็มีส่วนได้เปรียบในโครงงานนี้

# บรรณานุกรม

[1] ดร.ก่อเกียรติ เก่งสกุล. บุญเจริญ ศิริเนาวกุล. **ทฤษฎีและการประยุกต์ใช้งาน ปัญญาประดิษฐ์ และ ระบบผู้เชี่ยวชาญ.** กรุงเทพฯ : ซีเอ็ดยูเคชั่น. 2534

[2] Stuart J. Russell and Peter Norvig. **Artificial Intelligence A Modern Approach Second Edition.** PRENTICE HALL, USR, NJ 07458

[3] นิรันด์ ประวิทย์ธนา. **เก่ง C# ให้ครบสูตร.** กรุงเทพฯ : วิตตี้ กรุ๊ป. 2545.

[4] ณัช ภู่วรวรรณ. **เริ่มต้นเรียนรู้ C#.** กรุงเทพฯ : ซีเอ็ดยูเคชั่น. 2545

[5] ศุภชัย สมพานิช **คู่มือการเขียนโปรแกรม Visual C#.Net ฉบับโปรแกรมเมอร์.** นนทบุรี : อินโฟเพรส. 2546

# ประวัติผู้เขียนโครงงาน

ชื่อ นายเข็มทิพย์ เข็มทอง

ภูมิลำเนา 2/1 หมู่ 2 ตำบลตะกุดไร อำเภอชนแดน จังหวัดเพชรบูรณ์ 67190

ประวัติการศึกษา

- จบชั้นมัธยมศึกษาจากโรงเรียนดงขุยวิทยาคม จังหวัดเพชรบูรณ์
- ปัจจุบันกำลังศึกษาอยู่ระดับปริญญาตรีชั้นปีที่ 6
  สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์
  มหาวิทยาลัยนเรศวร

E-mail : khemisu@hotmail.com