การเรียนรู้ของเครื่องจักร

MACHINE LEARNING

นางสาวจิราพร พุกสุข รหัส 44362549

นายตรัยเทพ ชาญธัญกร รหัส 44362580

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

ปีการศึกษา 2547

ใบรับรองโครงงานวิศวกรรม

หัวข้อโครงงาน	การเรียนรู้ของเครื่องจักร		
ผู้ดำเนินโครงงาน	นางสาวจิราพร	พุกสุข	รหัส 44362549
	นายตรัยเทพ	ชาญธัญกร	รหัส 44362580
อาจารย์ที่ปรึกษา	อาจารย์พงศ์พันธ์	กิจสนาโยธิน	
สาขาวิชา	วิศวกรรมคอมพิวเตอร์		
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์		
ปีการศึกษา	2547		

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้โครงงานฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิศวกรรมคอมพิวเตอร์

คณะกรรมการสอบโครงงานวิศวกรรม

.....ประธานกรรมการ

(อาจารย์พงศ์พันธ์ กิจสนาโยธิน)

.....กรรมการ

(อาจารย์ธนิต มาลากร)

.....กรรมการ

(อาจารย์พนมขวัญ ริยะมงคล)

หัวข้อโครงงาน	การเรียนรู้ของเครื่องจักร		
ผู้ดำเนินโครงงาน	นางสาวจิราพร	พุกสุข	รหัส 44362549
	นายตรัยเทพ	ชาญธัญกร	รหัส 44362580
อาจารย์ที่ปรึกษา	อาจารย์พงศ์พันธ์	กิจสนาโยธิน	
สาขาวิชา	วิศวกรรมคอมพิวเตอร์		
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์		
ปีการศึกษา	2547		

.....

บทคัดย่อ

โครงงานนี้เป็นการศึกษาและพัฒนาโปรแกรมเพื่อให้คอมพิวเตอร์มีการพัฒนาการเรียนรู้จากข้อมูลที่ได้รับจากผู้ใช้ ซึ่งเป็นส่วนหนึ่งของการพัฒนาด้านปัญญาประดิษฐ์ มีจุดมุ่งหมายเพื่อให้คอมพิวเตอร์มีการพัฒนาตนเองในการทำงานเดิมให้ดีขึ้นกว่าเดิม โครงงานนี้ได้พัฒนาโปรแกรมกับเกมโอเทโร่ ซึ่งเป็นเกมที่ไม่มีความซับซ้อนมากจนเกินไปและมีโครงสร้างเกมที่เหมาะสมกับหลักการที่นำมาใช้ โดยโปรแกรมสามารถเก็บข้อมูลการเล่นในแต่ละครั้งที่เล่นได้ครบตามต้องการเพื่อไว้ใช้เป็นข้อมูลสำหรับเลือกตำแหน่งการเล่นในครั้งต่อๆไป และทำให้เกมมีความเชี่ยวชาญมากขึ้นสามารถเล่นโต้ตอบกับมนุษย์ได้ดีขึ้น

ผลที่ได้รับจากโครงงานนี้คือ การพัฒนาให้เครื่องจักรซึ่งโดยปกติแล้วไม่สามารถเรียนรู้ได้ด้วยตนเอง ให้มีความสามารถในการเรียนรู้พัฒนาตนเองมากขึ้น และอาจเป็นแนวทางให้เกิดการพัฒนาศึกษาต่อในด้านการพัฒนาการเรียนรู้ของเครื่องจักร

Project Title	Machine Learning
Name	Miss Jiraporn Pooksook ID. 44362549
	Mr.Trithep Chantunyakorn ID. 44362580
Project Advisor	Mr. PhongPhun Kijsanayothin
Major	Computer Engineering
Department	Electrical and Computer Engineering
Academic Year	2004

.....

ABSTRACT

This project is study and development a program for computer to develop itself by learning from input data .The principle is a part of Artificial Intelligent that aim to develop a machine that can learn .This project is developed with Othello game because this game is not complicate and easy for playing . Futhermore the important thing in this game is basic structure that useful for this principle. This program can store all of data which takes from every time to have playing and use these data for next time. That made the game can learn and good play in next time.

The result of this project is the machine that can learn by itself and develop their skill.In the end, this project is the sample project for develop and study more in this principle.

กิตติกรรมประกาศ

โครงงานนี้สามารถสำเร็จลุล่วงไปได้ด้วยดี เนื่องจากความอนุเคราะห์ในหลายๆด้าน จากผู้มีพระคุณทุกท่าน ซึ่งก่อนอื่นใด ขอขอบพระคุณ พ่อแม่และครอบครัวที่ให้โอกาสในการศึกษา และเป็นกำลังใจที่ดีเยี่ยมตลอดมา อีกทั้งยังเป็นกำลังทุนทรัพย์ที่สำคัญยิ่ง ขอขอบคุณเพื่อนร่วมชั้นที่ให้คำปรึกษาในด้านต่างๆและให้คำพูดที่สร้างกำลังใจดีๆให้แก่กัน ที่สำคัญยิ่งและคงจะขาดเสียไม่ได้ ขอขอบพระคุณ อาจารย์พงศ์พันธ์ กิจสนาโยธิน อาจารย์ที่ปรึกษาโครงการ ที่เสียสละทั้งกำลังกายและกำลังใจในการทุ่มเทให้กับโครงการนี้ ทั้งให้คำปรึกษาและช่วยแก้ไขข้อผิดพลาดต่างๆทำให้โครงการนี้เสร็จออกมาสมบูรณ์ยิ่งนัก

ท้ายที่สุดขอขอบพระคุณทุกท่านที่สละเวลาอันมีค่าให้ความสนใจในโครงการนี้ซึ่งผู้พัฒนาโครงการหวังเป็นอย่างยิ่งว่า โครงการนี้คงสร้างความพึงพอใจ และให้ประโยชน์อย่างใดอย่างหนึ่งแก่ท่านไม่มากก็น้อย

นางสาวจิราพร พุกสุข

นายตรัยเทพ ชาญธัญกร

สารบัญ

สารบัญ(ต่อ)

สารบัญรูป

สารบัญตาราง

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

เนื่องจากปัจจุบันคอมพิวเตอร์ได้เข้ามามีบทบาทในชีวิตของมนุษย์เรามากขึ้น แต่ทว่าความสามารถของคอมพิวเตอร์นั้นมีขีดจำกัดเพียงเท่าที่มนุษย์กำหนดไว้ให้ ไม่รู้จักคิดพิจารณาได้เอง ได้เพียงแต่ทำงานซ้ำๆที่มนุษย์กำหนดให้ การทำงานของคอมพิวเตอร์จึงยังไม่เต็มประสิทธิภาพเพียงพอ

มนุษย์จึงมีความพยายามอย่างมากที่จะพัฒนาคอมพิวเตอร์ให้รู้จักคิดและพิจารณาได้เอง ไม่ต้องรับคำสั่งจากผู้ใช้งานเพียงอย่างเดียว ซึ่งก็คือการรู้จักพิจารณาเงื่อนไขในกรณีต่างๆให้มากขึ้นนั่นเอง โดยที่อาศัยหลักการของการรู้จักเก็บประสบการณ์ไว้เหมือนกับมนุษย์ทั่วๆไปที่ย่อมมีประสบการณ์ผ่านเข้ามาในแต่ละวันแตกต่างกันไป เพื่อให้เป็นข้อมูลใช้ในครั้งต่อๆไป เอาไว้ใช้เปรียบเทียบว่ากรณีเดิมนี้ที่เคยพบมาแล้ว ควรจะทำแบบเดิมหรือไม่

ในการพัฒนาขั้นต้นนี้ เพื่อความง่ายและสะดวกในขั้นทดลอง จึงเริ่มพัฒนาแนวคิดนี้กับเกมง่ายๆก่อน เพราะเกมมีกฎกติกาที่แน่นอน เข้าใจง่ายไม่ซับซ้อน และมีขอบเขตรู้ผลลัพธ์แน่นอน ซึ่งเหมาะกับการพัฒนาอัลกอริทึมในการเก็บประสบการณ์ในการเล่นเกมว่า ถ้าหากรอบต่อไปมีการเล่นเหมือนกับรอบเดิมที่เคยเล่น ก็จะรู้ว่าควรเล่นแบบเดิมหรือไม่

ซึ่งจะทำให้คอมพิวเตอร์เล่นเกมได้เก่งขึ้นโดยอาศัยประสบการณ์ที่มากขึ้นซึ่งเหมือนกับมนุษย์เรานั่นเองซึ่งในโครงงานนี้ได้เลือกใช้กับเกมโอเทโร่ (Othello) เพราะเป็นเกมที่มีการเล่นง่ายไม่ซับซ้อน กฎเกณฑ์การเล่นที่แน่นนอน ซึ่งทำให้สามารถสร้างเป็นเงื่อนไขต่างๆที่จำกัดอยู่ในขอบเขตที่ต้องการได้ อีกทั้งยังเป็นเกมที่อาศัยกระบวนการคิดที่ต้องมองถึงอนาคต และอาศัยประสบการณ์ของผู้เล่น ซึ่งเป็นเหมือนการจำลองให้คอมพิวเตอร์รู้จักเก็บและใช้ประสบการณ์ของตนเอง

จากโครงงานนี้ สามารถนำไปประยุกต์ใช้กับสิ่งอื่นๆเพื่อก่อให้เกิดประโยชน์ได้ ไม่ว่าจะเป็นการสร้างโครงงานใหม่ๆให้เกิดขึ้นอีกมากมาย และยังเป็นการพัฒนาประสิทธิภาพของคอมพิวเตอร์ให้ดียิ่งขึ้นอีกด้วย

1.2 วัตถุประสงค์

1. เพื่อศึกษาเรียนรู้และพัฒนาความรู้ทางด้านปัญญาประดิษฐ์สาขาการเรียนรู้ของเครื่องจักร
2. เพื่อให้การใช้คอมพิวเตอร์ให้เกิดประสิทธิภาพมากขึ้นไม่อยู่ในขอบเขตจำกัดตามคำสั่งของผู้ใช้
3. ช่วยทุนแรงงานมนุษย์ในการทำงานที่ซ้ำซากและมีเงื่อนไขจำกัดโดยให้คอมพิวเตอร์ทำแทน
4. เพื่อลดปัญหาความผิดพลาดที่เกิดจากมนุษย์ในการทำงานที่ซ้ำซากโดยมีเงื่อนไขเดิม

1.3 เป้าหมายและขอบเขตของโครงการ

1. เกมสามารถเล่นแล้วคอมพิวเตอร์ชนะมากกว่า 50% ของการเล่นทั้งหมด
2. เกมสามารถเก็บข้อมูลการเล่นได้ครบทั้งหมดทุกรอบของการเล่น
3. เกมสามารถเล่นตอบโต้กับคนได้เพียงหนึ่งคนเท่านั้น

1.4 แผนการดำเนินงานตลอดโครงการ

แผนงาน/เดือน	ก.พ	มี.ค	เม.ย	พ.ค	มิ.ย	ก.ค	ส.ค	ก.ย
ศึกษาข้อมูลและเก็บรายละเอียดของข้อมูล	←	—	→					
ทดลองสร้างฐานข้อมูลเพื่อเก็บข้อมูลจริง			←	—	→			
เขียนโปรแกรมเกมโอเทโร่เพื่อให้เข้ากันได้กับฐานข้อมูล					←	—	→	
ทดลองใช้งานจริงและหาข้อผิดพลาดของโปรแกรม							←	— →

1.5 ผลที่คาดว่าจะได้รับ

1. เพื่อศึกษาการเรียนรู้และพัฒนาด้านปัญญาประดิษฐ์สาขาการเรียนรู้ของเครื่องจักรให้ก้าวหน้าขึ้น
2. เพื่อให้การใช้คอมพิวเตอร์เกิดประสิทธิภาพมากขึ้นไม่อยู่ในขอบเขตจำกัดตามคำสั่งของผู้ใช้
3. ช่วยทุ่นแรงมนุษย์ในการทำงานที่ซ้ำซากและมีเงื่อนไขจำกัด

1.6 รายละเอียดงบประมาณของโครงการ นิสิต : คน : 1,000 บาท

1. ค่าเอกสาร	1,000 บาท
2. ค่ากระดาษ A4	200 บาท
3. ค่าหมึกพิมพ์	500 บาท
4. ค่าอุปกรณ์สำนักงาน	300 บาท
รวม	2,000 บาท

บทที่ 2
หลักการและทฤษฎีเบื้องต้น

โครงงานนี้พัฒนาขึ้นโดยอาศัยหลักการ การเรียนรู้ของเครื่องจักร (Machine Learning) ซึ่งเป็นสาขาหนึ่งของการค้นคว้าวิจัยด้านปัญญาประดิษฐ์ อันเป็นกระบวนการที่ทำให้เครื่องจักรสามารถเรียนรู้ได้ด้วยตนเอง โดยอาศัยจากข้อมูลเดิมที่ได้รับ สำหรับโครงงานนี้ ใช้หลักการเพียงส่วนหนึ่งของกระบวนการเรียนรู้ที่มีอยู่ทั้งหมดในการพัฒนาให้เกมสามารถเรียนรู้พัฒนาการเล่นได้เอง

2.1 การเรียนรู้ของเครื่องจักร (Machine Learning)

การเรียนรู้ของเครื่องจักรคือส่วนหนึ่งของการค้นคว้าวิจัยด้านปัญญาประดิษฐ์ที่เกี่ยวข้องกับทฤษฎี กระบวนการคำนวณที่พัฒนากระบวนการเรียนรู้และสร้างเครื่องจักรที่เรียนรู้ได้เอง

นิยามของการเรียนรู้คือ "การเปลี่ยนแปลงในระบบซึ่งทำให้เครื่องจักรทำงานได้ดีกว่าเดิมในการทำงานครั้งที่สองในการทำงานแบบเดียวกัน หรือในงานอื่นๆที่ใช้ชุดข้อมูลเดียวกัน "

หลักสำคัญของนิยามการเรียนรู้คือ ความชำนาญอย่างละเอียดและประณีต ในระบบที่อาศัยความชำนาญอย่างละเอียดและประณีตนี้ได้รับผลจากความรู้ที่ได้มา สิ่งมีชีวิตที่ฉลาดทั้งหมดต้องสามารถที่จะเรียนรู้ได้ในระดับการปรับตัวและระดับดั้งเดิม

ความสามารถในการเรียนรู้เป็นตัววัดความฉลาด และ เป็นความต้องการพื้นฐานของระบบปัญญาประดิษฐ์ ระบบปัญญาประดิษฐ์ต้องการเปลี่ยนธรรมชาติของงานหนักต่างๆของมนุษย์ เพื่อความสามารถที่จะเรียนรู้ได้เองและทำงานแทนมนุษย์ได้

การเรียนรู้ของเครื่องจักรเป็นส่วนหนึ่งของโปรแกรมที่อยู่ในการพัฒนาระบบการสอนอย่างฉลาด (Intelligent Tutoring Systems) และมีเทคนิคการเรียนรู้ของเครื่องจักรจำนวนมากที่ใช้ในระบบการสอนอย่างฉลาด

2.1.1 ประเภทของการเรียนรู้ของเครื่องจักร

การเรียนรู้ของเครื่องจักรมีหลายประเภท ซึ่งโครงงานนี้ได้นำหลักการกระบวนการเรียนรู้การจดจำ (Rote Learning Algorithm) ใช้พัฒนาโครงงาน

2.1.1.1 กระบวนการเรียนรู้ที่ชัดขึ้น (The Focusing Algorithm)

กระบวนการเรียนรู้ที่ชัดขึ้น มีการพิจารณาและค้นคว้าวิจัย โดยมีแนวคิดที่มีจุดมุ่งหมายเพื่อสร้างนิยามที่ถูกต้องให้กับข้อมูลที่ได้รับในทางด้านบวกแต่ไม่เกี่ยวกับข้อมูลที่ได้รับในทางด้านลบ

กระบวนการเรียนรู้ที่ชัดเจนใช้ version space (การแสดงออกของความรู้ที่ใช้จะเก็บส่วนของการแนะนำที่มีประโยชน์ตามลำดับของตัวอย่างการเรียนรู้โดยปราศจากการจำใดๆ ของตัวอย่าง) โดยค้นหาผ่าน concept space (การใช้ความแปรปรวนเพื่อหาเทคนิคที่ใกล้เคียง โดยอัตโนมัติ)

- Concept space ครอบคลุมแนวความคิดที่เป็นไปได้ทั้งหมด

- Version space ครอบคลุมเพียงแค่แนวความคิดซึ่งไม่มีการเปลี่ยนแปลงกับตัวอย่างการฝึกหัดใดๆ แนวคิดที่ชัดเจนมีความคล้ายคลึงกับแนวคิดการตัดออกของมิทเชลล์ (Mitchell 's candidate elimination Algorithm)

สมมุติเกี่ยวกับขนาดของพาหนะใช้เพื่อไปยังปลายทางที่กำหนด เหตุผลในตัวอย่างนี้คือชนิดของพาหนะ และขนาดสำหรับแต่ละเหตุผลของวิธีต้นไม้การตัดสินใจถูกใช้ครอบคลุมตัวอย่างทั้งหมดที่เป็นไปได้ที่เกิดขึ้น

กระบวนการเรียนรู้ที่ชัดขึ้นเหมือนแนวคิดการเรียนรู้ส่วนใหญ่ที่ไม่สามารถจัดการคำอธิบายซึ่งมีลักษณะที่แยกออกจากความคิด เช่น ในตัวอย่าง [ขนาด(เล็ก), พาหนะ] ความหมายคือ เล็ก และพาหนะ เป็นไปไม่ได้ที่จะแสดงขนาด เล็ก หรือขนาดกลาง และพาหนะได้

ปัญหาอื่นๆที่มีผลกับกระบวนการเรียนรู้ที่ชัดขึ้นคือ มีแต่ความรู้พื้นฐานเท่านั้นที่ถูกใช้แสดงภายในแนวความคิดต้นไม้การตัดสินใจ สิ่งนี้สามารถชี้แจงสำหรับการเรียนรู้ที่ผิดได้ แต่ไม่ใช่สำหรับมนุษย์โดยธรรมชาติที่กำลังเรียนรู้สถานการณ์ ในการเรียนรู้ของมนุษย์จะมีความรู้พื้นฐานกว้างกว่าจึงจะค้นพบส่วนประกอบทั่วไประหว่างตัวอย่างที่ด้านลบและที่ด้านบวกซึ่งสามารถช่วยให้ตั้งสมมุติฐานได้

2.1.1.2 การเรียนรู้ของความคิดและต้นไม้การตัดสินใจ (Concept Learning and Decision Trees)

ระบบการเรียนรู้ของแนวความคิดสร้างความรู้ในรูปของต้นไม้การตัดสินใจซึ่งสามารถที่จะช่วยแก้ ปัญหายากๆหมวดหมู่ ต้นไม้การตัดสินใจ คือ ความพยายามในการเข้าใกล้ในหมวดหมู่ซึ่งเริ่มจากจุดราก และเพิ่มที่จะสร้างต้นไม้ย่อยจนกระทั่งจุดใบไม้ ต้นไม้การตัดสินใจประกอบด้วย

- ใบไม้ หรือ จุดคำตอบ ซึ่งชี้บอกการแบ่งออกเป็นหมวดหมู่ที่อันใดอันหนึ่งเป็นบวกหรืออันใดอันหนึ่งเป็นลบ

- ไม่มีใบไม้ หรือจุดการตัดสิน ซึ่งบรรจุค่าชื่อ และสาขาไปยังต้นไม้การตัดสินใจอื่นๆ

2.1.1.3 กระบวนการเรียนรู้การแบ่งออกเป็นหมวดหมู่ (The classification algorithm)

กระบวนการเรียนรู้การแบ่งออกเป็นหมวดหมู่ ไม่มีเทคนิคการเรียนรู้แนวคิดที่ดีขึ้น แต่มีการสร้างสมมุติฐานในรูปแบบของต้นไม้การตัดสินใจ

2.1.1.4 กระบวนการเรียนรู้ ID 3 (Quinlan 's ID 3 algorithm)

แนวความคิด ID 3 ของควินลาน (Quinlan 's ID 3 Algorithm) เป็นการเพิ่มวิธีคิดการแบ่งออกเป็นหมวดหมู่ กระบวนการเรียนรู้ ID 3 เพิ่มสองความสามารถจากกระบวนการเรียนรู้การแบ่งออกเป็นหมวดหมู่ คือ กำลังเปิด (windowing) และกระตุ้นความสนใจในข้อมูล

Windowing สามารถถูกใช้งานได้ ถ้ามีตัวอย่างการฝึกหัดจำนวนมาก ส่วนย่อยของการฝึกหัดตั้งค่าจะถูกสุ่มเลือกเพื่อสร้างต้นไม้เริ่มต้น กรณีสิ่งที่นำเข้าที่ยังคงอยู่จะถูกใช้แยกประเภทในการสร้างต้นไม้ มีการแยกหมวดหมู่ที่ถูกต้องกับสิ่งที่นำเข้าเหล่านี้ สำหรับการฝึกหัดทั้งหมดและจนกระทั่งจบกระบวนการ

ถ้าสิ่งนี้มีการเลือกการแยกประเภทตัวอย่างไม่ถูกต้องจะถูกเพิ่มเติมที่การเปิด และกระบวนการดำเนินต่อไปจนกระทั่งมีการให้แบ่งออกเป็นหมวดหมู่ที่ถูกต้องสำหรับค่าที่ตั้งไว้ทั้งหมด จากหลักฐานซึ่งได้จากประสบการณ์จากการฝึกหัดทั้งหมดที่ตั้งค่าไว้

อย่างไรก็ตามจะมีการแนะนำโดยธอร์นทอน (Thornton) ข้อดีของการเปิดมีเล็กน้อย และโอดีฟี (O 'Keefe) บอกว่าวิธีการเปิดไม่ประกันตลอดเวลาว่าจะค้นพบต้นไม้การตัดสินใจที่ถูกต้องเว้นแต่การเปิดใช้การฝึกหัดทั้งหมดที่ตั้งค่าไว้

ข้อมูลที่กระตุ้นความสนใจ ถูกใช้สร้างต้นไม้ที่ตื้นกว่าโดยตัดสินใจคำสั่งในการเลือกค่าขั้นแรกใช้ข้อมูลที่กระตุ้นความสนใจจะคำนวณสัดส่วนของกรณีการฝึกหัดที่ทางด้านบวกและทางด้านลบที่สิ่งนี้ได้มีให้ที่จุดในเวลานั้น ในกรณีของจุดรากสิ่งนี้จะมีการฝึกหัดที่ตั้งค่า ค่ารู้จักกันดีว่าเป็นข้อมูลที่จำเป็นสำหรับจุดที่ถูกคำนวณ โดยใช้สูตรดังต่อไปนี้

เมื่อ p คือ สัดส่วนของกรณีทางด้านบวก และ q คือ สัดส่วนของกรณีทางด้านลบ

$$-p \log_2 p - q \log_2 q \quad \text{---------- (1)}$$

ค่าที่เป็นไปได้ทั้งหมดที่จุดอยู่ถัดไปที่ถูกพิจารณา ค่าทั้งหมดที่เป็นไปได้เหล่านี้ไม่เพียงถูกใช้จากจุดรากถึงจุดปัจจุบัน ที่รากค่าทั้งหมดที่เป็นไปไม่ได้

เป็นต้นว่า ถ้าพิจารณา " ความสูง " คือ พิจารณาที่จุดรากค่าจะถูกคำนวณสำหรับแต่ละค่า คือ " สั้น " และ " สูง "

ผลรวมของค่าเหล่านี้ต้องการปัจจัยรวมคือสัดส่วนของกรณีที่จุดกับค่าที่เจาะจงในกรณีของ " ความสูง " ที่รากจะมีผลรวมดังนี้

(สัดส่วนของความสูง) x (ข้อมูลที่จำเป็นสำหรับค่าความสูง) ---------- (2)

(สัดส่วนของความสั้น) x (ข้อมูลที่จำเป็นสำหรับค่าความสั้น) ---------- (2)

ผลรวมเป็นคะแนน และจะถูกดึงออกจากข้อมูลที่จำเป็นสำหรับจุดที่จะให้ความคาดหวังของข้อมูลที่ได้รับ

2.1.1.5 กระบวนการเรียนรู้ ID 4 (The ID 4 algorithm)

กระบวนการเรียนรู้ ID 4 ถูกพัฒนา โดย สคลิมเมอร์ และฟิชเชอร์ (Schlimmer & Fisher) แนวความคิดนี้สร้างต้นไม้การตัดสินใจได้ดีขึ้น งานที่เรียนรู้จำนวนมากที่ดีขึ้นเป็นตัวอย่างใหม่ หรือรายละเอียดที่มีอยู่ตลอดเวลา กระบวนการเรียนรู้ ID 4 ทำงานโดยสร้างต้นไม้ และปรับปรุงใหม่เป็นตัวอย่างใหม่ที่ใช้ได้

2.1.1.6 กระบวนการเรียนรู้ ID 5 และ ID5R (The ID5 and ID5R algorithm)

ID 5 และ ID5R มีการสร้างต้นไม้การตัดสินใจที่ดีขึ้นที่ดีกว่า ID 4 ความแตกต่างที่จำเป็นเพื่อแสดงเมื่อต้นไม้ต้องการโครงสร้างใหม่อีกครั้ง เพราะว่าค่าที่จุดไม่ต้องมีคะแนนต่ำที่สุด ต้นไม้ย่อย ใดๆ ไม่ถูกตัดทิ้ง ค่อนข้างจะคล้ายกับว่าค่านั้นถูกแทนที่ที่จุดถูกดึงสูงจนถึงจุด และ โครงสร้างต้นไม้ต่ำกว่าจุดที่ถูกกำหนด ไว้ ในกรณีของ ID 5 ต้นไม้ย่อยไม่ถูกปรับปรุงใหม่อีกครั้ง ขณะที่ใน ID5R ถูกปรับปรุงไม่ใช่เพราะว่าโครงสร้างต้นไม้ย่อยคำนวณได้มีประสิทธิภาพมากกว่า อย่างไรก็ตามผลลัพธ์ต้นไม้ย่อยไม่รับประกันว่าจะเหมือนกับการสร้างโดย ID 3 บนตัวอย่างการฝึกหัดเดียวกัน แต่ ID5R รับประกันในกรณีนี้

2.1.1.7 วิธีการเรียนรู้พื้นฐานการอธิบาย (Explanation based learning methods)

การเรียนรู้พื้นฐานการอธิบาย (EBL: Explanation based learning methods) สิ่งที่นำเข้าในวิธีการเรียนรู้พื้นฐานการอธิบายมีดังนี้

ตัวอย่างการฝึกหัด: ตัวอย่างที่ คือ กลุ่มของคำอธิบายข้อเท็จจริงตัวอย่างของความคิดหลัก

นิยามหลัก: ตัวอย่างที่ คือ กลุ่มของคำอธิบายข้อเท็จจริงตัวอย่างของความคิดหลัก

กฎเกณฑ์ที่ได้: เจาะจงรูปแบบในสิ่งที่นิยามความคิดที่การวางหลักอาจจะถูกแสดง

ความคิดหลัก: คำอธิบายระดับสูงของแนวคิดที่ถูกเรียนรู้

จากข้างต้น วิธีการเรียนรู้พื้นฐานการอธิบาย สามารถสร้างลักษณะทั่วไปของตัวอย่างที่ฝึกฝนเป็นสิ่งที่สามารถบรรยายความคิดหลัก และทำกฎเกณฑ์ได้

วิธีแรกของการสร้างการอธิบายโดยวิเคราะห์ตัวอย่างที่ฝึกฝนที่แสดงข้างต้น นิยามของความคิดการอธิบายนี้ต่อมาได้มีการใช้สูตรแสดงนิยามของความคิดทั่วๆ ไป

ความสามารถสำคัญของวิธีการเรียนรู้พื้นฐานการอธิบาย เพื่อสามารถที่จะพิสูจน์แนวความคิดที่ถูกต้องนี้ คือ ปัญหาที่การรู้จักของความคล้ายคลึงกันของระบบที่กำลังเรียนรู้

2.1.1.8 กระบวนการเรียนรู้แบบพวง (Clustering Algorithm)

กระบวนการเรียนรู้แบบพวง คือ แนวคิดการเรียนรู้อีกรูปแบบหนึ่ง กระบวนการเรียนรู้แบบพวงจะไม่รวมถึงการสอนภายนอก การเรียนรู้ของระบบชนิดนี้ไม่ได้แยกประเภทของวัตถุเข้าไปในลำดับชั้นของหมวดหมู่แต่จะวัดค่าความคล้ายคลึงกันระหว่างวัตถุ และมีการรวบรวมวัตถุอย่างมากที่สุดซึ่งคล้ายกันเข้าไปในกลุ่มเดียวกัน กระบวนการของกลุ่มที่ตัดสินใจสามารถทำอยู่ในอันใดอันหนึ่งจากล่างขึ้นบนหรือบนลงล่าง

กระบวนการเรียนรู้ของพวงผสมวัตถุเดี่ยวหรือกลุ่มของวัตถุเข้าไปในกลุ่มมากมายกว่า จนกระทั่งต้นกำเนิดส่วนหนึ่งของวัตถุผสมกันเข้าไปในประเภทเดี่ยวซึ่งที่ ที่ตั้งที่สูงสุดของลำดับชั้น

จากบนลงล่าง เป็นวิธีแบ่งต้นกำเนิดส่วนหนึ่งของวัตถุเข้าไปในหมวดหมู่ย่อย จนกระทั่งวัตถุแต่ละอันถูกกำหนดหมวดหมู่

2.1.1.9 กระบวนการเรียนรู้ความคล้ายคลึงกัน (Analogy Algorithm)

กระบวนการเรียนรู้ความคล้ายคลึงกันเป็นการใช้ความรู้ที่ได้จากสถานการณ์ใหม่ๆ โดยทั่วๆ ไปความคล้ายคลึงกันสร้างการปะติดปะต่อกันระหว่างกรณีเก่ากับกรณีใหม่ กล่าวอีกอย่างหนึ่ง คือ การสร้างรูปแบบของกรณีใหม่จากต้นกำเนิดความรู้เดิม

พื้นฐานของความคล้ายคลึงกันคือ ถ้ามีสองสถานการณ์ มีสิ่งหรือความสามารถมีส่วนร่วมกันดังนั้นสมควรให้มีสิ่งอื่นๆมีส่วนร่วมกัน เช่น เครื่องบินและเรือ ทั้งสองอย่างมีผู้โดยสารสิ่งอื่นๆ ที่มีส่วนร่วมกันคือ การเดินทางจากสถานที่หนึ่งไปยังที่อื่นๆ ทั้งสองอย่างมี

การกอบกู้ (Retrieval)	ขั้นตอนนี้เป็นการอุปมารวมถึงการเลือกความสามารถเป็นไปได้ของเป้าหมาย และต้นกำเนิดซึ่งทำให้เพิ่มการกอบกู้ของความคล้ายคลึงกันของประโยชน์ต้นกำเนิดและความรู้ที่อยู่บนพื้นฐานของความสามารถเหล่านี้
ความซับซ้อน (Elaboration)	ขั้นตอนนี้เป็นความสามารถเพิ่มเติมและความสัมพันธ์ของต้นกำเนิดบ่อยๆในบางกรณีที่พัฒนาการแก้ไขปัญหาอย่างจำเพาะเจาะจง สำหรับเป้าหมายหลักอาจจะหลีกเลี่ยงไม่ได้
การปะติดปะต่อและอนุมาน (Mapping and inferencing)	ขั้นตอนนี้รวมถึงการสร้างการปะติดปะต่อจากรูปลักษณ์ของต้นกำเนิดไปถึงเป้าหมายความคิดหลัก
การกลั่นกรอง (Justification)	ขั้นตอนนี้ตรวจดู ถ้าปะติดปะต่อระหว่างต้นกำเนิด และเป้าหมายสมบูรณ์บางครั้งต้องการแก้ไขการปะติดปะต่อนั้นใหม่
การเรียนรู้ (Learning)	ขั้นตอนนี้ คือการเก็บรวมรวมการเรียนรู้ไว้ใช้ในอนาคต

มีสองปัญหาอุปมาที่แก้ยุทธศาสตร์เหล่านี้ คือ ความคล้ายคลึงกันการเปลี่ยนสภาพ และความคล้ายคลึงกันการได้มา

ความคล้ายคลึงกันการเปลี่ยนสภาพ (Transformational analogy)

ความคล้ายคลึงกันการเปลี่ยนสภาพใช้การแก้ปัญหาในเรื่องปัญหาก่อนหน้าและเปลี่ยนรูปการแก้ปัญหาเพื่อว่าการแก้ปัญหาสามารถใช้แก้ปัญหาปัจจุบันในได้

เป็นต้นว่าวัตถุ X ถูกติดมาจากลอนดอนถึงอิสแทนบัล (Istanbul) และความต้องการวัตถุใหม่ที่จะเดินทางจากอิสแทนบัลถึงลอนดอน การใช้ความคล้ายคลึงกันการเปลี่ยนสภาพกับการแก้ปัญหาที่เป็นต้นฉบับ วัตถุใหม่อาจจะเป็นการส่งโดยเรือ เป็นพื้นฐานรูปแบบความคล้ายคลึงกันการเปลี่ยนสภาพสำหรับกรณีที่พื้นฐานเข้าใกล้กัน

ความคล้ายคลึงกันการได้มา (Derivational analogy)

ความคล้ายคลึงกันการได้มา เป็นการแก้ปัญหาต่างๆ อาจจะพิจารณาได้หลายทางเลือก โดยให้วัตถุการส่งของ X จากลอนดอนถึงอิสแทนบัล อย่างไรก็ตาม สำหรับเหตุผลต่างๆเหล่านี้ อาจจะไม่เหมาะสม เช่น วัตถุที่จะส่งมากเกินไปสำหรับการขนส่งอากาศ ความคล้ายคลึงกันการได้มาจะตรวจสอบดูส่วนประกอบต่างๆของวัตถุก่อนว่า การขนส่งอากาศพอที่จะรับได้หรือไม่ และอาจจะไม่แนะนำให้มีการส่งของสำหรับวัตถุใหม่ๆ

2.1.1.10 กระบวนการเรียนรู้การจดจำ (Rote Learning Algorithm)

กระบวนการเรียนรู้การจดจำ คือ การจดจำ (memorization) ระบบการเรียนรู้การจดจำไม่จำเป็นต้องทำการประมวลผลใดๆ เพื่อให้เข้าใจหรืออธิบายข้อมูล สิ่งเดียวที่ระบบต้องการทำคือการจดจำหรือเก็บข้อมูลข่าวสารที่เข้ามาสำหรับในครั้งต่อๆไป

ส่วนสำคัญของการศึกษาการเรียนรู้การจดจำถูกคิดค้นโดย ซามูเอล(Samuel) ได้มีการพัฒนาโปรแกรมเครื่องเล่นหมากรุก (checker player program) ซึ่งสามารถที่จะปรับปรุงประสิทธิภาพของโปรแกรมได้โดยการจดจำทุกๆ ตำแหน่งบอร์ดของหมากรุก

2.2 ความรู้เบื้องต้นเกี่ยวกับระบบฐานข้อมูล

2.2.1 ความหมายของระบบฐานข้อมูล

ระบบฐานข้อมูล (Database System) หมายถึง โครงสร้างสารสนเทศที่ประกอบด้วยรายละเอียดของข้อมูลที่เกี่ยวข้องกันที่จะนำมาใช้ในระบบต่าง ๆ ร่วมกัน

ระบบฐานข้อมูลจึงนับว่าเป็นการจัดเก็บข้อมูลอย่างเป็นระบบ ซึ่งผู้ใช้สามารถจัดการกับข้อมูลได้ในลักษณะต่าง ๆ ทั้งการเพิ่ม การแก้ไข การลบ ตลอดจนการเรียกดูข้อมูล ซึ่งส่วนใหญ่จะเป็นการประยุกต์นำเอาระบบคอมพิวเตอร์เข้ามาช่วยในการจัดการฐานข้อมูล

2.2.2 นิยามและคำศัพท์พื้นฐานเกี่ยวกับระบบฐานข้อมูล

บิต (Bit) หมายถึง หน่วยของข้อมูลที่มีขนาดเล็กที่สุด

ไบท์ (Byte) หมายถึง หน่วยของข้อมูลที่เกิดจากการนำบิตมารวมกันเป็นตัวอักขระ

เขตข้อมูล (Field) หมายถึง หน่วยของข้อมูลที่ประกอบขึ้นจากตัวอักขระตั้งแต่หนึ่งตัวขึ้นไปมารวมกันแล้วได้ความหมายของสิ่งใดสิ่งหนึ่ง เช่น ชื่อ ที่อยู่ เป็นต้น

ระเบียน (Record) หมายถึง หน่วยของข้อมูลที่เกิดจากการนเอาเขตข้อมูลหลาย ๆ เขตข้อมูลมารวมกัน เพื่อเกิดเป็นข้อมูลเรื่องใดเรื่องหนึ่ง

แฟ้มข้อมูล (File) หมายถึงหน่วยของข้อมูลที่เกิดจากการนำข้อมูลหลาย ๆ ระเบียนที่เป็นเรื่องเดียวกันมารวมกัน เช่น แฟ้มข้อมูลนักศึกษา แฟ้มข้อมูลลูกค้า แฟ้มข้อมูลพนักงาน

ส่วนในระบบฐานข้อมูล มีคำศัพท์ต่าง ๆ ที่เกี่ยวข้องดังนี้

เอนทิตี้ (Entity) หมายถึง ชื่อของสิ่งใดสิ่งหนึ่ง ได้แก่ คน สถานที่ สิ่งของ การกระทำ ซึ่งต้องการจัดเก็บข้อมูลไว้ เช่น เอนทิตี้ลูกค้า เอนทิตี้พนักงาน

- เอนทิตี้ชนิดอ่อนแอ (Weak Entity) เป็นเอนทิตี้ที่ไม่มีความหมาย หากขาดเอนทิตี้อื่นในฐานข้อมูล

แอททริบิวต์(Attribute) หมายถึง รายละเอียดข้อมูลที่แสดงลักษณะและคุณสมบัติของเอนทิตี้หนึ่ง ๆ เช่น เอนทิตี้นักศึกษา ประกอบด้วย

- แอทริบิวต์รหัสนักศึกษา
- แอททริบิวต์ชื่อนักศึกษา
- แอททริบิวต์ที่อยู่นักศึกษา

ความสัมพันธ์ (Relationships) หมายถึง ความสัมพันธ์ระหว่างเอนทิตี้ เช่น ความสัมพันธ์ระหว่างเอนทิตี้นักศึกษาและเอนทิตี้คณะวิชา เป็นลักษณะว่า นักศึกษาแต่ละคนเรียนอยู่คณะวิชาใดคณะวิชาหนึ่ง

ความสัมพันธ์ระหว่างเอนทิตี้ แบ่งออกเป็น 3 ประเภท คือ

- ความสัมพันธ์แบบหนึ่งต่อหนึ่ง (One-to-one Relationships) เป็นการแสดงความสัมพันธ์ของข้อมูลในเอนทิตี้หนึ่งที่มีความสัมพันธ์กับข้อมูลในอีกเอนทิตี้หนึ่ง ในลักษณะหนึ่งต่อหนึ่ง (1 : 1)

- ความสัมพันธ์แบบหนึ่งต่อกลุ่ม (One-to-many Relationships) เป็นการแสดงความสัมพันธ์ของข้อมูลในเอนทิตี้หนึ่ง ที่มีความสัมพันธ์กับข้อมูลหลาย ๆ ข้อมูลในอีกเอนทิตี้หนึ่ง ในลักษณะ (1:m)

- ความสัมพันธ์แบบกลุ่มต่อกลุ่ม (Many-to-many Relationships) เป็นการแสดงความสัมพันธ์ของข้อมูลสองเอนทิตี้ในลักษณะกลุ่มต่อกลุ่ม (m:n)

2.2.3 ความสำคัญของการประมวลผลแบบระบบฐานข้อมูล

จากการจัดเก็บข้อมูลรวมเป็นฐานข้อมูลจะก่อให้เกิดประโยชน์ดังนี้

1. สามารถลดความซ้ำซ้อนของข้อมูลได้

การเก็บข้อมูลชนิดเดียวกันไว้หลาย ๆ ที่ ทำให้เกิดความซ้ำซ้อน (Redundancy) ดังนั้นการนำข้อมูลมารวมเก็บไว้ในฐานข้อมูล จะช่วยลดปัญหาการเกิดความซ้ำซ้อนของข้อมูลได้ โดยระบบ

จัดการฐานข้อมูล (Database Management System : DBMS) จะช่วยควบคุมความซ้ำซ้อนได้ เนื่องจากระบบจัดการฐานข้อมูลจะทราบได้ตลอดเวลาว่ามีข้อมูลซ้ำซ้อนกันอยู่ที่ใดบ้าง

2. หลีกเลี่ยงความขัดแย้งของข้อมูลได้

หากมีการเก็บข้อมูลชนิดเดียวกันไว้หลาย ๆ ที่และมีการปรับปรุงข้อมูลเดียวกันนี้ แต่ปรับปรุงไม่ครบทุกที่ที่มีข้อมูลเก็บอยู่ก็จะทำให้เกิดปัญหาข้อมูลชนิดเดียวกัน อาจมีค่าไม่เหมือนกันในแต่ละที่ที่เก็บข้อมูลอยู่ จึงก่อให้เกิดความขัดแย้งของข้อมูลขึ้น (Inconsistency)

3. สามารถใช้ข้อมูลร่วมกันได้

ฐานข้อมูลจะเป็นการจัดเก็บข้อมูลรวมไว้ด้วยกัน ดังนั้นหากผู้ใช้ต้องการใช้ข้อมูลในฐานข้อมูลที่มาจากแฟ้มข้อมูลต่างๆ ก็จะทำได้โดยง่าย

4. สามารถรักษาความถูกต้องเชื่อถือได้ของข้อมูล

บางครั้งพบว่าการจัดเก็บข้อมูลในฐานข้อมูลอาจมีข้อผิดพลาดเกิดขึ้น เช่น จากการที่ผู้ป้อนข้อมูลป้อนข้อมูลผิดพลาดคือป้อนจากตัวเลขหนึ่งไปเป็นอีกตัวเลขหนึ่ง โดยเฉพาะกรณีมีผู้ใช้หลายคนต้องใช้ข้อมูลจากฐานข้อมูลร่วมกัน หากผู้ใช้คนใดคนหนึ่งแก้ไขข้อมูลผิดพลาดก็ทำให้ผู้อื่นได้รับผลกระทบตามไปด้วย ในระบบจัดการฐานข้อมูลจะสามารถใส่กฎเกณฑ์เพื่อควบคุมความผิดพลาดที่เกิดขึ้น

5. สามารถกำหนดความเป็นมาตรฐานเดียวกันของข้อมูลได้

การเก็บข้อมูลร่วมกันไว้ในฐานข้อมูลจะทำให้สามารถกำหนดมาตรฐานของข้อมูลได้รวมทั้งมาตรฐานต่าง ๆ ในการจัดเก็บข้อมูลให้เป็นไปในลักษณะเดียวกันได้ เช่นการกำหนดรูปแบบการเขียนวันที่ ในลักษณะ วัน/เดือน/ปี หรือ ปี/เดือน/วัน ทั้งนี้จะมีผู้ที่คอยบริหารฐานข้อมูลที่เราเรียกว่าผู้บริหารฐานข้อมูล (Database Administrator : DBA) เป็นผู้กำหนดมาตรฐานต่างๆ

6. สามารถกำหนดระบบความปลอดภัยของข้อมูลได้

ระบบความปลอดภัยในที่นี้ เป็นการป้องกันไม่ให้ผู้ใช้ที่ไม่มีสิทธิมาใช้ หรือมาเห็นข้อมูลบางอย่างในระบบ ผู้บริหารฐานข้อมูลจะสามารถกำหนดระดับการเรียกใช้ข้อมูลของผู้ใช้แต่ละคนได้ตามความเหมาะสม

7. เกิดความเป็นอิสระของข้อมูล

ในระบบฐานข้อมูลจะมีตัวจัดการฐานข้อมูลที่ทำหน้าที่เป็นตัวเชื่อมโยงกับฐานข้อมูล โปรแกรมต่าง ๆ อาจไม่จำเป็นต้องมีโครงสร้างข้อมูลทุกครั้ง ดังนั้นการแก้ไขข้อมูลบางครั้ง จึงอาจกระทำเฉพาะกับโปรแกรมที่เรียกใช้ข้อมูลที่เปลี่ยนแปลงเท่านั้น ส่วนโปรแกรมที่ไม่ได้เรียกใช้ข้อมูลดังกล่าว ก็จะเป็นอิสระจากการเปลี่ยนแปลง

2.2.4 รูปแบบของระบบฐานข้อมูล

รูปแบบของระบบฐานข้อมูล มีอยู่ด้วยกัน 3 ประเภท คือ

1. ฐานข้อมูลเชิงสัมพันธ์ (Relational Database)

เป็นการเก็บข้อมูลในรูปแบบที่เป็นตาราง (Table) หรือเรียกว่า รีเลชั่น (Relation) มีลักษณะเป็น 2 มิติ คือเป็นแถว (row) และเป็นคอลัมน์ (column) การเชื่อมโยงข้อมูลระหว่างตาราง จะเชื่อมโยงโดยใช้แอททริบิวต์ (attribute) หรือคอลัมน์ที่เหมือนกันทั้งสองตารางเป็นตัวเชื่อมโยงข้อมูล ฐานข้อมูลเชิงสัมพันธ์นี้จะเป็นรูปแบบของฐานข้อมูลที่นิยมใช้ในปัจจุบัน

2. ฐานข้อมูลแบบเครือข่าย (Network Database)

ฐานข้อมูลแบบเครือข่ายจะเป็นการรวมระเบียนต่าง ๆ และความสัมพันธ์ระหว่างระเบียนแต่จะต่างกับฐานข้อมูลเชิงสัมพันธ์ คือ ในฐานข้อมูลเชิงสัมพันธ์จะแฝงความสัมพันธ์เอาไว้ โดยระเบียนที่มีความสัมพันธ์กันจะต้องมีค่าของข้อมูลในแอททริบิวต์ใดแอททริบิวต์หนึ่งเหมือนกันแต่ฐานข้อมูลแบบเครือข่าย จะแสดงความสัมพันธ์อย่างชัดเจน

3. ฐานข้อมูลแบบลำดับชั้น (Hierarchical Database)

ฐานข้อมูลแบบลำดับชั้น เป็นโครงสร้างที่จัดเก็บข้อมูลในลักษณะความสัมพันธ์แบบพ่อ-ลูก (Parent-Child Relationship Type : PCR Type) หรือเป็นโครงสร้างรูปแบบต้นไม้ (Tree) ข้อมูลที่จัดเก็บในที่นี้ คือ ระเบียน (Record) ซึ่งประกอบด้วยค่าของเขตข้อมูล (Field) ของเอนทิตี้หนึ่ง ๆ ฐานข้อมูลแบบลำดับชั้นนี้คล้ายคลึงกับฐานข้อมูลแบบเครือข่าย แต่ต่างกันที่ฐานข้อมูลแบบลำดับชั้น มีกฎเพิ่มขึ้นมาอีกหนึ่งประการ คือ ในแต่ละกรอบจะมีลูกศรวิ่งเข้าหาได้ไม่เกิน 1 หัวลูกศร

2.2.5 โปรแกรมฐานข้อมูลที่นิยมใช้

โปรแกรมฐานข้อมูล เป็นโปรแกรมหรือซอฟแวร์ที่ช่วยจัดการข้อมูลหรือรายการต่าง ๆ ที่อยู่ในฐานข้อมูล ไม่ว่าจะเป็นการจัดเก็บ การเรียกใช้ การปรับปรุงข้อมูล โปรแกรมฐานข้อมูล จะช่วยให้ผู้ใช้สามารถค้นหาข้อมูลได้อย่างรวดเร็ว ซึ่งโปรแกรมฐานข้อมูลที่นิยมใช้มีอยู่ด้วยกันหลายตัว เช่น Access, FoxPro, Clipper, dBase, FoxBase, Oracle, SQL เป็นต้น โดยแต่ละโปรแกรมจะมีความสามารถต่างกัน บางโปรแกรมใช้ง่ายแต่จะจำกัดขอบเขตการใช้งาน บ่งโปรแกรมใช้งานยากกว่า แต่จะมีความสามารถในการทำงานมากกว่า

โปรแกรม Access นับเป็นโปรแกรมที่นิยมใช้กันมากในขณะนี้ โดยเฉพาะในระบบฐานข้อมูลขนาดใหญ่ สามารถสร้างแบบฟอร์มที่ต้องการจะเรียกดูข้อมูลในฐานข้อมูล หลังจากบันทึกข้อมูลในฐานข้อมูลเรียบร้อยแล้ว จะสามารถค้นหาหรือเรียกดูข้อมูลจากเขตข้อมูลใดก็ได้

นอกจากนี้ Access ยังมีระบบรักษาความปลอดภัยของข้อมูล โดยการกำหนดรหัสผ่านเพื่อป้องกันความปลอดภัยของข้อมูลในระบบได้ด้วย

โปรแกรม FoxPro เป็นโปรแกรมฐานข้อมูลที่มีผู้ใช้งานมากที่สุด เนื่องจากใช้ง่ายทั้งวิธีการเรียกจากเมนูของ FoxPro และประยุกต์โปรแกรมขึ้นใช้งาน โปรแกรมที่เขียนด้วย FoxPro จะสามารถใช้กลับ dBase คำสั่งและฟังก์ชั่นต่าง ๆ ใน dBase จะสามารถใช้งานบน FoxPro ได้

นอกจากนี้ใน FoxPro ยังมีเครื่องมือช่วยในการเขียนโปรแกรม เช่น การสร้างรายงาน โปรแกรม dBase เป็นโปรแกรมฐานข้อมูลชนิดหนึ่ง การใช้งานจะคล้ายกับโปรแกรม FoxPro ข้อมูลรายงานที่อยู่ในไฟล์บน dBase จะสามารถส่งไปประมวลผลในโปรแกรม Word Processor ได้ และแม้แต่ Excel ก็สามารถอ่านไฟล์ .DBF ที่สร้างขึ้นโดยโปรแกรม dBase ได้ด้วย

โปรแกรม SQL เป็นโปรแกรมฐานข้อมูลที่มีโครงสร้างของภาษาที่เข้าใจง่าย ไม่ซับซ้อน มีประสิทธิภาพการทำงานสูง สามารถทำงานที่ซับซ้อนได้โดยใช้คำสั่งเพียงไม่กี่คำสั่ง โปรแกรม SQL จึงเหมาะที่จะใช้กับระบบฐานข้อมูลเชิงสัมพันธ์ และเป็นภาษาหนึ่งที่มีผู้นิยมใช้กันมาก โดยทั่วไปโปรแกรมฐานข้อมูลของบริษัทต่าง ๆ ที่มีใช้อยู่ในปัจจุบัน เช่น Oracle, DB2 ก็มักจะมีคำสั่ง SQL ที่ต่างจากมาตรฐานไปบ้างเพื่อให้เป็นจุดเด่นของแต่ละโปรแกรมไป

2.3 ความรู้เบื้องต้นเกี่ยวกับภาษา SQL

SQL (Structured Query Language) เป็นภาษาที่ใช้ในการจัดการฐานข้อมูล และข้อมูลในฐานข้อชุดคำสั่ง SQL ได้รับความนิยมอย่างมากในระบบจัดการฐานข้อมูลแบบตารางสัมพันธ์ โดยมีการกำหนดมาตราฐานของชุดคำสั่ง SQL ขึ้น เรียกว่าเป็น ANSI-SQL เช่น ANSI-86, SQL-89 ปัจจุบันเป็น ANSI-92 ถึงจะมีมาตราฐานก็ตาม ผู้สร้างระบบจัดการฐานข้อมูลแต่ละรายจะเพิ่มเติมหรือดัดแปลงคำสั่งต่างๆ จากมาตราฐาน เพื่อให้ระบบจัดการฐานข้อมูลของตนเองมีประสิทธิภาพสูงขึ้น แต่โดยรวมจะมีคำสั่งพื้นฐานที่เหมือนกัน ดังนั้นหากเรียนรู้คำสั่ง SQL ของระบบจัดการฐานข้อมูลหนึ่งก็จะสามารถนำไปใช้ได้กับอีกระบบหนึ่งเช่นกัน

ชุดคำสั่ง SQL จะประกอบด้วยชุดคำสั่งที่สำคัญ 2 ชุดที่สำคัญ คือ

1. <u>ชุดคำสั่งสำหรับจัดการกับโครงสร้างของฐานข้อมูล</u> (Data Defination Language หรือ DDL) เป็นชุดคำสั่งสำหรับใช้ในการสร้าง เปลี่ยนแปลง และลบ ตาราง และดรรชนี ในฐานข้อมูล

2. <u>ชุดคำสั่งสำหรับจัดการกับข้อมูลในฐานข้อมูล</u> (Data Mainpulation Language หรือ DML) เป็นชุดคำสั่งสำหรับใช้ในการค้นหา แก้ไข เพิ่มเติม และลบ ข้อมูลในตารางต่างๆ ในฐานข้อมูล

2.3.1 ชุดคำสั่งสำหรับจัดการกับโครงสร้างของฐานข้อมูล (Data Defination Language)

คำสั่ง SQL สำหรับจัดการกับโครงสร้างฐานข้อมูลที่สำคัญจะมีอยู่เพียง 3 คำสั่งเท่านั้นคือ CREATE TABLE / INDEX ใช้สำหรับสร้างตาราง หรือดรรชนี และกำหนดความสัมพันธ์ของข้อมูลระหว่างตารางที่เกี่ยวข้องกัน

1. <u>การสร้างตารางใหม่ในฐานข้อมูล</u> สามารถสร้างได้ดังนี้

CREATE ชื่อตาราง (ชื่อฟิลด์ ประเภทของฟิลด์ [ขนาดของฟิลด์] [NOT NULL] [ดรรชนี],

ชื่อฟิลด์ ประเภทของฟิลด์ [ขนาดของฟิลด์] [NOT NULL] [ดรรชนี], ... ,

ชื่อฟิลด์ ประเภทของฟิลด์ [ขนาดของฟิลด์] [NOT NULL] [ดรรชนี])

สำหรับ NOT NULL หากต่อท้ายฟิลด์ใด ฟิลด์นั้นจะเป็นช่องว่างไม่ได้

2. การสร้างตารางใหม่พร้อมดรรชนี

ในการสร้างดรรชนีพร้อมกับการสร้างตารางนั้น ทำได้โดยการเพิ่มคำสั่ง CONSTRAINT ส่วนที่เป็น Index บนฟิลด์นั้น โดยมีรูปแบบดังนี้

CONSTRAINT name {PRIMARY KEY| UNIQUE | NOT NULL}

name ชื่อของดรรชนีที่ต้องการสร้าง

PRIMARY KEY กำหนดให้ดรรชนีที่สร้างเป็นดรรชนีหลัก

UNIQUE กำหนดค่าในดรรชนีนี้ข้อมูลซ้ำกันไม่ได้

NOT NULL ค่าในดรรชนีจะไม่มี ไม่ได้

3. การแก้ไขข้อมูลในตาราง

เช่นการเพิ่มฟิลด์ใหม่ให้กับตาราง ลบฟิลด์ออกจากตาราง การเพิ่มดรรชนีหรือความสัมพันธ์ รวมทั้งการลบความดรรชนีหรือความสัมพันธ์

4. การลบตารางออกจากฐานข้อมูล

คำสั่งที่ใช้ในการลบตารางออกจากฐานข้อมูล มีรูปแบบดังนี้

DROP TABLE ชื่อตาราง

5. การสร้างดรรชนี

การสร้างดรรชนีสามารถสร้างโดยคำสั่ง ALTER TABLE แล้วยังมีคำสั่ง CREATE INDEX สำหรับใช้ใน การสร้างดรรชนีบนตารางที่สร้างไว้แล้ว มีรูปแบบดังนี้

CREATE INDEX ชื่อดรรชนี ON ชื่อตาราง (ชื่อฟิลด์ [ASC|DESC], ชื่อฟิลด์ [ASC|DESC],..., [ASC|DESC]) [WITH {PRIMARY | DISALLOW NULL | GNORE NULL}]

โดยที่ ASC กำหนดให้จัดเรียงข้อมูลจากน้อยไปมาก

DESC กำหนดให้จัดเรียงข้อมูลจากมากไปหาน้อย

PRIMARY กำหนดให้เป็นดรรชนีหลักของตาราง

DISALLOW NULL กำหนดให้ดรรชนีที่สร้างนี้ทุกๆ ฟิลด์ ต้องไม่เป็น NULL

GNORE NULL กำหนดให้ดรรชนีที่สร้างนี้ ฟิลด์ที่ประกอบกันอาจมีค่าเป็น NULL ได้

6. การลบดรรชนี

การลบดรรชนีออกจากฐานข้อมูลทำได้โดยใช้คำสั่งDROP INDEX ชื่อดรรชนี ON ชื่อตารางที่ใช้สร้างดรรชนี

2.3.2 ชุดคำสั่งสำหรับการจัดการข้อมูล (Data Manipulation Language)

คำสั่ง SQL สำหรับจัดการข้อมูลในฐานข้อมูลที่สำคัญ มี 4 คำสั่งเท่านั้นคือ

คำสั่ง SELECT

เป็นคำสั่งสำหรับเลือกข้อมูล หรือค้นหาข้อมูลที่ต้องการจากฐานข้อมูล โดยสามารถกำหนดได้ว่าผลลัพธ์ที่ต้องการจะให้แสดงฟิลด์ใดบ้าง และกำหนดเงื่อนไขที่ต้องการข้อมูลใดบ้างขึ้นมาแสดง โดยมีรูปแบบ หลักๆ ดังนี้

SELECT [predicate] [field1, field2,..] [*]

FROM table-name [, …]

[WHERE search-criteria]

[GROUP BY group-list]

[ORDER BY sort-criteria}

predicate หมายถึงฟังก์ชั่นที่ใช้ในการหาค่าต่างๆ จากข้อมูลเช่น MAX, MIN เป็นต้น

field1, field2,… หมายถึงชื่อฟิลด์ที่ต้องการให้แสดง

table-name หมายถึงชื่อตารางที่ต้องการนำข้อมูลมาแสดง สามารถเรียกได้พร้อมกันหลายตาราง หรือตารางเดียวก็ได้

search-criteria หมายถึงเงื่อนไขสำหรับการค้นหาข้อมูล

group-list หมายถึงส่วนระบุสำหรับการจัดกลุ่มข้อมูลของผลลัพธ์ที่ได้จากคำสั่ง SELECT นั้น

sort-criteria หมายถึงส่วนระบุสำหรับใช้ในการจัดเรียงลำดับข้อมูลผลลัพธ์ที่ได้

ฟังก์ชั่นที่ใช้ร่วมในคำสั่ง SELECT จะเป็นฟังก์ชั่นทางสถิติเบื้องต้น ได้แก่

COUNT (field) เป็นคำสั่งสำหรับนับจำนวนข้อมูลทั้งหมดที่ได้มาจากคำสั่ง SELECT

SUM(field) เป็นคำสั่งสำหรับรวมค่าของข้อมูลใน field นั้นๆ ทุกเรกคอร์ด ที่ได้จากคำสั่ง SELECT

MAX(field) เป็นคำสั่งหาค่าสูงสุดของข้อมูลใน field นั้นทุกเรกคอร์ดที่ได้จากคำสั่ง SELECT

MIN(field) เป็นคำสั่งหาค่าที่น้อยที่สุดของข้อมูลใน field ทุกเรกคอร์ดที่ได้จากคำสั่ง SELECT

คำสั่ง DELETE

เป็นคำสั่งสำหรับใช้ในการลบข้อมูลออกจากตาราง โดยมีรูปแบบดังนี้

DELETE FROM table-name [WHERE search-criteria]

โดยที่

table-name คือชื่อของตารางที่ต้องการลบข้อมูลออก

Search-criteria คือเงื่อนไขในการลบข้อมูล หากไม่กำหนดจะถือว่าเป็นการลบข้อมูลทุก ๆ เรกคอร์ดจากตาราง

คำสั่ง UPDATE

เป็นคำสั่งสำหรับทำการเปลี่ยนแปลงข้อมูลในตาราง มีรูปแบบดังนี้

UPDATE table-nam SET field-name = value, …

WHERE search-criteria

โดยที่

table-name คือ ชื่อตาราง

field-name คือ ชื่อฟิลด์ที่ต้องการเปลี่ยนแปลงข้อมูล

value คือ ค่าที่ต้องการเปลี่ยนแปลง

Search-criteria คือเงื่อนไขในการเลือกเรกคอร์ดที่ต้องการเปลี่ยนแปลง

คำสั่ง INSERT

เป็นคำสั่งสำหรับเพิ่มเติมข้อมูลเข้าไปในตาราง รูปแบบของคำสั่งมีดังนี้

INSERT INTO table-name (field1, field2, …)

VALUES (value1, value2, …)

โดยที่

table-name คือ ชื่อตาราง

field1, field2,… คือ ชื่อฟิลด์ที่อยู่ในตารางที่ต้องการเพิ่มข้อมูลเข้าไป

value1, value2,…คือ ค่าของฟิลด์ที่เพิ่มเข้าไป

บทที่ 3
วิธีการออกแบบโปรแกรม

โครงงานนี้ใช้วิธีการออกแบบโปรแกรมโดยอาศัยหลักการ เก็บข้อมูลลงฐานข้อมูลในจำนวนที่มาก อันเป็นกระบวนการที่ทำให้เครื่องจักรสามารถเรียนรู้ได้ด้วยตนเอง โดยอาศัยจากข้อมูลเดิมที่ได้รับจากการทำงานแต่ละครั้ง สำหรับโครงงานนี้ ใช้หลักการเพียงส่วนหนึ่งของกระบวนการเรียนรู้ที่มีอยู่ทั้งหมดในการออกแบบโปรแกรมให้เกมสามารถเรียนรู้พัฒนาการเล่นได้เองโดยไม่ต้องมีโครงสร้างใดๆมากระทำ

3.1 ภาพรวมของการออกแบบโปรแกรม

การออกแบบโปรแกรมการเรียนรู้ของเครื่องจักรนี้ แบ่งออกเป็น 2 ส่วนคือการออกแบบโปรแกรมส่วนของ Server และการออกแบบโปรแกรมส่วนของ Client

3.1.1 การออกแบบโปรแกรมส่วนของ Server

3.1.1.1 ส่วนที่ทำหน้าที่รับการ Connection ของ Client

ส่วนนี้จะคอยควบคุมการติดต่อของ Client ในด้านการตรวจสอบชื่อไม่ให้ซ้ำกันการตรวจสอบการติดต่อ และการเลิกติดต่อ

3.1.1.2 ส่วนที่ทำหน้าที่รับ - ส่งข้อความระหว่าง Client และ Server

หากได้รับข้อความที่ขึ้นต้นด้วย MOVE หมายถึง ได้รับตำแหน่งที่เลือกเดิน Server ก็จะนำตำแหน่งที่ได้ไปหาผลลัพธ์และส่งผลลัพธ์ไปให้ผู้เล่นอีกฝ่าย

หากได้รับข้อความที่ขึ้นต้นด้วย END หมายถึง จบการส่งแล้ว Server ก็จะส่ง START ไปให้ยังผู้เล่นอีกฝ่าย หมายถึงให้เริ่มเล่นได้

3.1.1.3 RESULT OF MOVE

การออกแบบโปรแกรม ส่วน RESULT OF MOVE คือ การออกแบบโปรแกรมให้เปลี่ยนค่าตามตำแหน่งต่างๆที่ต้องเปลี่ยนไปตามกฎของเกมหลังจากที่ Client หนึ่งเลือกตำแหน่งเล่นแล้ว คือ ตรงกลางระหว่างตำแหน่งที่เลือกเล่นไปจนถึงตำแหน่งของอีก Client หากเป็น Client ฝ่ายตรงข้ามจะต้องเปลี่ยนเป็นของ Client ที่เลือกเล่นทั้งหมด

3.1.1.4 ENDGAME

การออกแบบโปรแกรม ส่วน ENDGAME คือ การออกแบบโปรแกรมให้ Server เป็นฝ่ายตรวจสอบว่าจบเกมหรือไม่ และ Client แต่ละฝ่ายมีคะแนนเท่าใด และส่งคะแนนกลับไปยัง Client ทั้งสอง Client

3.1.2 การออกแบบโปรแกรมส่วน Client

การออกแบบโปรแกรมส่วน Client นั้นจะมีการพัฒนา 2 แบบ คือ การพัฒนาส่วนที่ใช้ติดต่อกับ User ทั่วๆไป (User Interface) และการพัฒนาส่วนที่พัฒนาการเรียนรู้ของเกม (Machine Learning)

3.1.2.1 การออกแบบโปรแกรมส่วน USER INTERFACE

3.1.2.1.1 GRAPHIC INTERFACE

การออกแบบโปรแกรม ส่วน GRAPHIC INTERFACE คือ การออกแบบโปรแกรมส่วนแสดงผลของโปรแกรม เพื่อให้ง่ายต่อการใช้งานสำหรับผู้ใช้ ซึ่งโปรแกรม ประกอบไปด้วยส่วนแสดงผลภาพปกติ ใช้การวาดภาพเป็นตารางขนาด 8 คูณ 8 และใช้การวาดวงกลมแทนสัญลักษณ์ของผู้เล่นแต่ละฝ่ายซึ่งมีสีขาวกับสีดำ และส่วนแสดงผลปุ่มควบคุมต่างๆ ใช้การสร้างปุ่มควบคุมและเขียนโปรแกรมควบคุมตอบสนองการกดปุ่มควบคุมต่างๆ ซึ่งมี 2 ปุ่มคือ NEW GAME และ QUIT

3.1.2.1.2 EVENT LISTENER

การออกแบบโปรแกรม ส่วน EVENT LISTENER คือการออกแบบโปรแกรมควบคุมปฏิกิริยาต่างๆที่ได้รับจากผู้ใช้ซึ่งมี 2 ส่วนดังนี้

MOUSE LISTENER

ตอบสนองการคลิกบนส่วนแสดงผลของเกมเพื่อเก็บค่าตำแหน่งที่ผู้ใช้เลือก

BUTTON LISTENER

ตอบสนองการกดปุ่มต่างๆ ซึ่งมีปุ่ม NEW GAME และ QUIT

3.1.2.1.3 POSSIBLE MOVE

การออกแบบโปรแกรม ส่วน POSSIBLE MOVE คือ การออกแบบโปรแกรมหาตำแหน่งที่ผู้เล่นและคอมพิวเตอร์สามารถเล่นได้ ใช้การหาตำแหน่งของผู้เล่นก่อนและตำแหน่งที่ติดกันกับผู้เล่นต้องเป็นฝ่ายตรงข้ามไปตลอดจนกระทั่งพบตำแหน่งว่างอีกหนึ่งตำแหน่ง เพื่อเลือกเล่นในตำแหน่งว่าง

3.1.2.1.4 RESULT OF MOVE

การออกแบบโปรแกรม ส่วน RESULT OF MOVE คือ การออกแบบโปรแกรมให้เปลี่ยนค่าตามตำแหน่งต่างๆที่ต้องเปลี่ยนไปตามกฎของเกมหลังจากที่ผู้เล่นเลือกตำแหน่งเล่นแล้ว คือ ตรงกลาง ระหว่างตำแหน่งที่เลือกเล่นไปจนถึงตำแหน่งของฝ่ายผู้เลือกเล่นหากเป็นฝ่ายตรงข้ามจะต้องเปลี่ยนเป็นของฝ่ายผู้เลือกเล่นทั้งหมด ซึ่งจะใช้เปลี่ยนผลลัพธ์ค่าบอร์ดเฉพาะฝ่ายตนเองเท่านั้นในส่วนของฝ่ายตรงข้ามนั้นก็จะรับจาก Server แทน

3.1.2.2 การออกแบบโปรแกรมส่วน MACHINE LEARNING

3.1.2.2.1 QUERY DATABASE

การออกแบบโปรแกรม ส่วน QUERY DATABASE คือ การจัดการฐานข้อมูลโดยใช้คำสั่ง " SELECT " ซึ่งเป็นการเลือกข้อมูลจากฐานข้อมูล ตามเงื่อนไขที่กำหนด ในโปรแกรมนี้ส่วนการเลือกข้อมูลจากฐานข้อมูลคือ เมื่อหา Possiblemove ทั้งหมดได้แล้วคอมพิวเตอร์จะทำการเลือกเพียงตำแหน่งเดียวจากทั้งหมดที่คอมพิวเตอร์สามารถเดินได้ โดยจะทำการเลือกตำแหน่งที่มีค่า Fitness ดีที่สุด ซึ่งเกิดจากการนำจำนวนครั้งชนะหารด้วยจำนวนครั้งที่นำไปใช้ หากว่าค่า Fitness ที่ดีที่สุดนั้นมีค่าเท่ากัน จะทำการนำค่าตำแหน่งนั้นๆไปค้นหาค่า Fitness จากตาราง POSITION และเลือกค่าที่มี Fitness สูงสุดมาใช้

3.1.2.2.2 UPDATE DATABASE

การออกแบบโปรแกรม ส่วน UPDATE DATABASE คือ การจัดการฐานข้อมูลโดยใช้คำสั่ง " UPDATE " ซึ่งเป็นการปรับปรุงข้อมูลในฐานข้อมูล ด้วยค่าใหม่ที่กำหนดให้ ตามเงื่อนไขที่กำหนด ในโปรแกรมนี้ส่วนการปรับปรุงฐานข้อมูลคือ เมื่อจบเกมจะมีการเพิ่มจำนวนครั้งที่เลือกตำแหน่งเดินทั้งในตาราง DATA และ ตาราง POSITON โดยในโปรแกรมทุกๆครั้งที่ทำการเลือกเดินตำแหน่งใด ก็จะมีการเก็บตำแหน่งการเดินนั้นๆไว้เสมอ และเมื่อจบเกมจะนำตำแหน่งการเลือกเดินที่เก็บไว้มาเป็นรหัสชี้ในฐานข้อมูลเพื่อเพิ่มค่าจำนวนครั้งที่เลือกเดินไปอีก 1 และถ้าหากว่าคอมพิวเตอร์เป็นฝ่ายชนะ ก็จะมีการเพิ่มค่า Point (จำนวนครั้งที่ชนะ) อีกค่า หนึ่งด้วย แต่ถ้าหากว่าคอมพิวเตอร์เป็นฝ่ายแพ้จะไม่มีการเพิ่มค่า Point

3.1.2.3 INSERT DATABASE

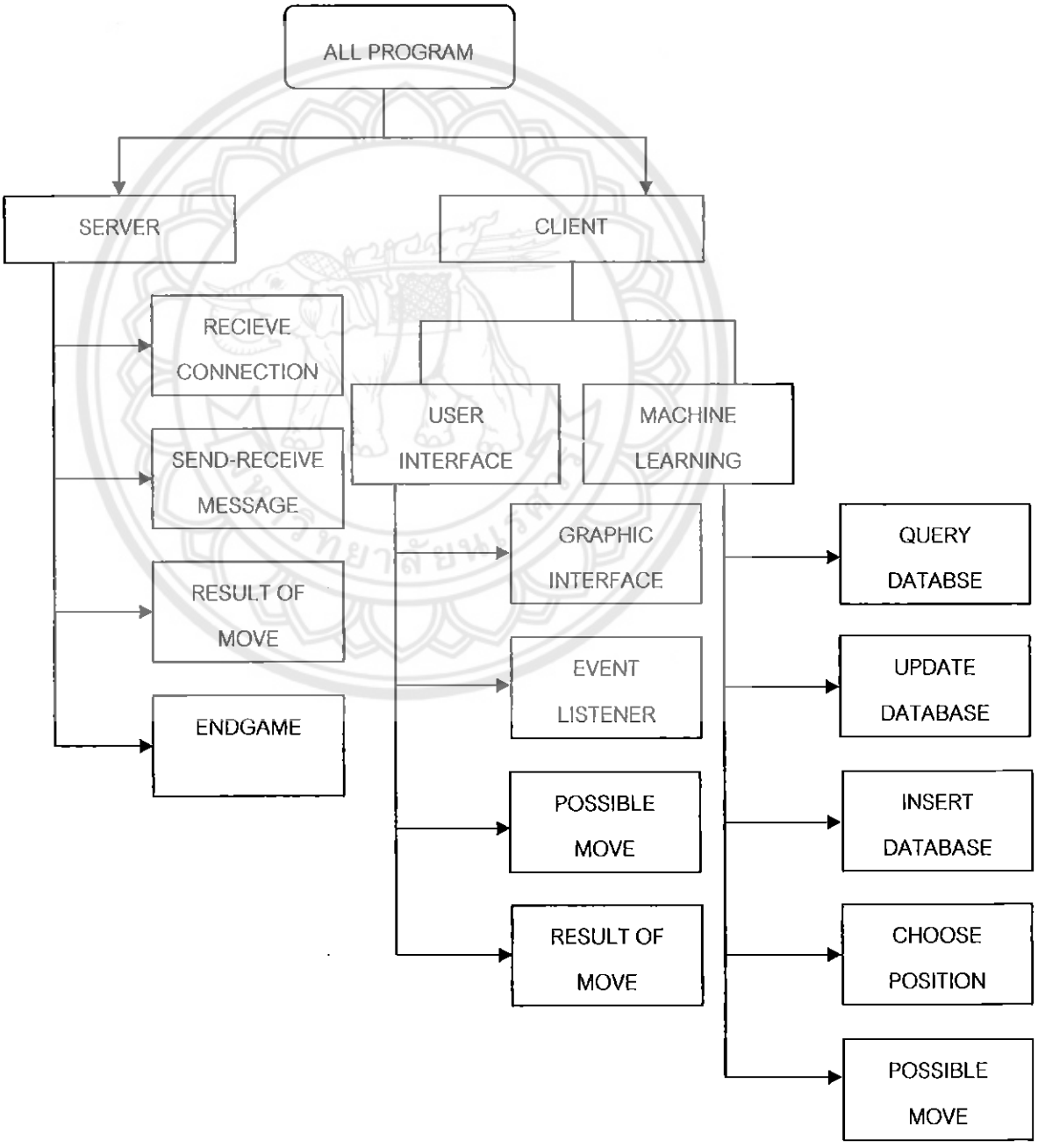
การออกแบบโปรแกรม ส่วน INSERT DATABASE คือ การจัดการฐานข้อมูลโดยใช้คำสั่ง " INSERT " ซึ่งเป็นการเพิ่มข้อมูลลงในฐานข้อมูล ด้วยค่าใหม่ที่กำหนดให้ ตามเงื่อนไขที่กำหนด ในโปรแกรมนี้ส่วนการเพิ่มข้อมูลในฐานข้อมูลคือ เมื่อคอมพิวเตอร์คำนวณหาตำแหน่งทั้งหมดที่สามารถเลือกเดินได้ ก็จะทำการนำข้อมูลตำแหน่งที่เดินได้ และกำหนด Time (จำนวนครั้งที่เดิน) เริ่มต้นเป็น 1 เสมอ และกำหนดค่า Point (จำนวนครั้งที่ชนะ)เริ่มต้นเป็น 1 เสมอ เช่นกัน เก็บลงในฐานข้อมูล เหตุที่ให้ค่า Time และ Point เริ่มต้นเป็น 1 เพราะว่าเมื่อนำค่า Point หารด้วย Time นั้นโอกาสที่จะถูกเลือกก็จะมีเท่าๆกันในทุกตำแหน่ง และโอกาสนี้ก็จะเปลี่ยนไปตามจำนวนครั้งที่ถูกนำไปใช้และครั้งที่ชนะ

3.1.2.4 CHOOSE POSITION

ใช้ในการเก็บข้อมูลการเลือกเดินของคอมพิวเตอร์เป็นวัตถุของคลาส OthelloStatus และเก็บวัตถุไว้ในเวกเตอร์อีกทีหนึ่งโดยเก็บค่าตัวแปรรหัสบอร์ด ตำแหน่งแถว และตำแหน่งคอลัมน์ เพื่อใช้ในการ เพิ่มค่าจำนวนครั้งในการเลือกไปใช้ตอนจบเกม ซึ่งจะทำทั้งตาราง DATA และตาราง POSITION

3.1.2.5 POSSIBLE MOVE

การออกแบบโปรแกรม ส่วน POSSIBLE MOVE คือ การออกแบบโปรแกรมหาตำแหน่งที่ผู้เล่นและคอมพิวเตอร์สามารถเล่นได้ ใช้การหาตำแหน่งของผู้เล่นก่อนและตำแหน่งที่ติดกันกับผู้เล่นต้องเป็นฝ่ายตรงข้ามไปตลอดจนกระทั่งพบตำแหน่งว่างอีกหนึ่งตำแหน่ง เพื่อเลือกเล่นในตำแหน่งว่าง



รูปที่ 3.1 แสดงภาพรวมการออกแบบโปรแกรม

3.2 การทำงานของโปรแกรม

การทำงานของโปรแกรมเริ่มต้นนั้น โปรแกรมทำการ Connect ไปยัง Server และให้ผู้เล่นเป็นฝ่ายเล่นคนแรก โดยให้หมากเดินของผู้เล่นเป็นหมากสีขาว และฝ่ายคอมพิวเตอร์เป็นหมากสีดำ โดยโปรแกรมจะรอรับค่าตำแหน่งเดินของผู้เล่นจากการคลิกเมาส์ และเมื่อมีการคลิกเกิดขึ้น จะเก็บตำแหน่งที่คลิกได้เป็นตัวเลข Pixel แล้วนำมาคำนวณว่าตำแหน่งที่คลิกอยู่ในแถวใดและคอลัมน์ใด จากนั้นโปรแกรมจะทำการส่งค่าตำแหน่งที่เลือกเดินไปยัง Server และ Server จะทำการคำนวณผลลัพธ์ที่เกิดขึ้นหลังจากการเลือกเดินตำแหน่งนั้นๆ แล้ว Server จะทำการส่งค่าผลลัพธ์ของบอร์ดไปยัง Client อีกฝ่ายซึ่งก็คือฝ่ายของการเรียนรู้ของโปรแกรม

เมื่อ Client ฝ่ายการเรียนรู้ได้รับค่าบอร์ดจาก Server โปรแกรมก็ทำการแสดงผล โปรแกรมก็จะนำผลลัพธ์ทั้งหมดบนบอร์ดมาเข้ารหัส เป็นรหัสตัวอักษร 32 ตัวเพื่อให้ง่ายต่อการอ้างอิงในฐานข้อมูล โดยหลักการการเข้ารหัสมีดังนี้

จากรูปจะเห็นว่า บอร์ดมีขนาด 8 คูณ 8 ซึ่งแต่ละช่องจะมี 3 สถานะ คือ

- ช่องว่าง กำหนดให้เป็นเลขไบนารี่ 00
- สีดำ กำหนดให้เป็นเลขไบนารี่ 01
- สีขาว กำหนดให้เป็นเลขไบนารี่ 10

และจาก 64 ช่องของบอร์ดจะทำให้ได้เลขไบนารี่ยาว 128 ตัวเลข เราจะทำการเลือกทีละ 4 ตัวเลข จาก ลำดับแรกไปจนสุดท้าย แล้วนำเลข 4 ตัวที่เลือกมาเทียบตามสถานะที่สามารถเป็นได้ 9 แบบ คือ

- ช่องว่าง,ช่องว่าง	0000	ได้อักษร A
- ช่องว่าง,สีดำ	0001	ได้อักษร B
- ช่องว่าง,สีขาว	0010	ได้อักษร C
- สีดำ,ช่องว่าง	0100	ได้อักษร D
- สีดำ,สีดำ	0101	ได้อักษร E
- สีดำ,สีขาว	0110	ได้อักษร F
- สีขาว,ช่องว่าง	1000	ได้อักษร G
- สีขาว,สีดำ	1001	ได้อักษร H
- สีขาว,สีขาว	1010	ได้อักษร I

เมื่อเทียบรหัสตัวเลขทีละ 4 ตัวเลขจนครบ 128 ตัวเลข ทำให้ได้ตัวอักษร 32 ตัวซึ่งทำให้ง่ายต่อการอ้างถึงในฐานข้อมูล และ การเก็บรหัสแบบนี้ทำให้เปลืองหน่วยความจำน้อยกว่าการเก็บเป็นเลขฐานสิบ ตัวอย่างเช่น จากรูปจะเห็นว่าเมื่อเริ่มเกมจะมีหมากสีขาวและสีดำอยู่ตรงกลางเสมอ เมื่อนำมาเข้ารหัสจะทำให้ได้รหัสตัวเลขคือ

แถวแรก	0000000000000000	แถวที่สอง	0000000000000000
แถวที่สาม	0000000000000000	แถวที่สี่	0000001001000000
แถวที่ห้า	0000000110000000	แถวที่หก	0000000000000000
แถวที่เจ็ด	0000000000000000	แถวที่แปด	0000000000000000

ทำให้ได้รหัสตัวอักษรคือ AAAA AAAA AAAA ACDA ABGA AAAA AAAA AAAA

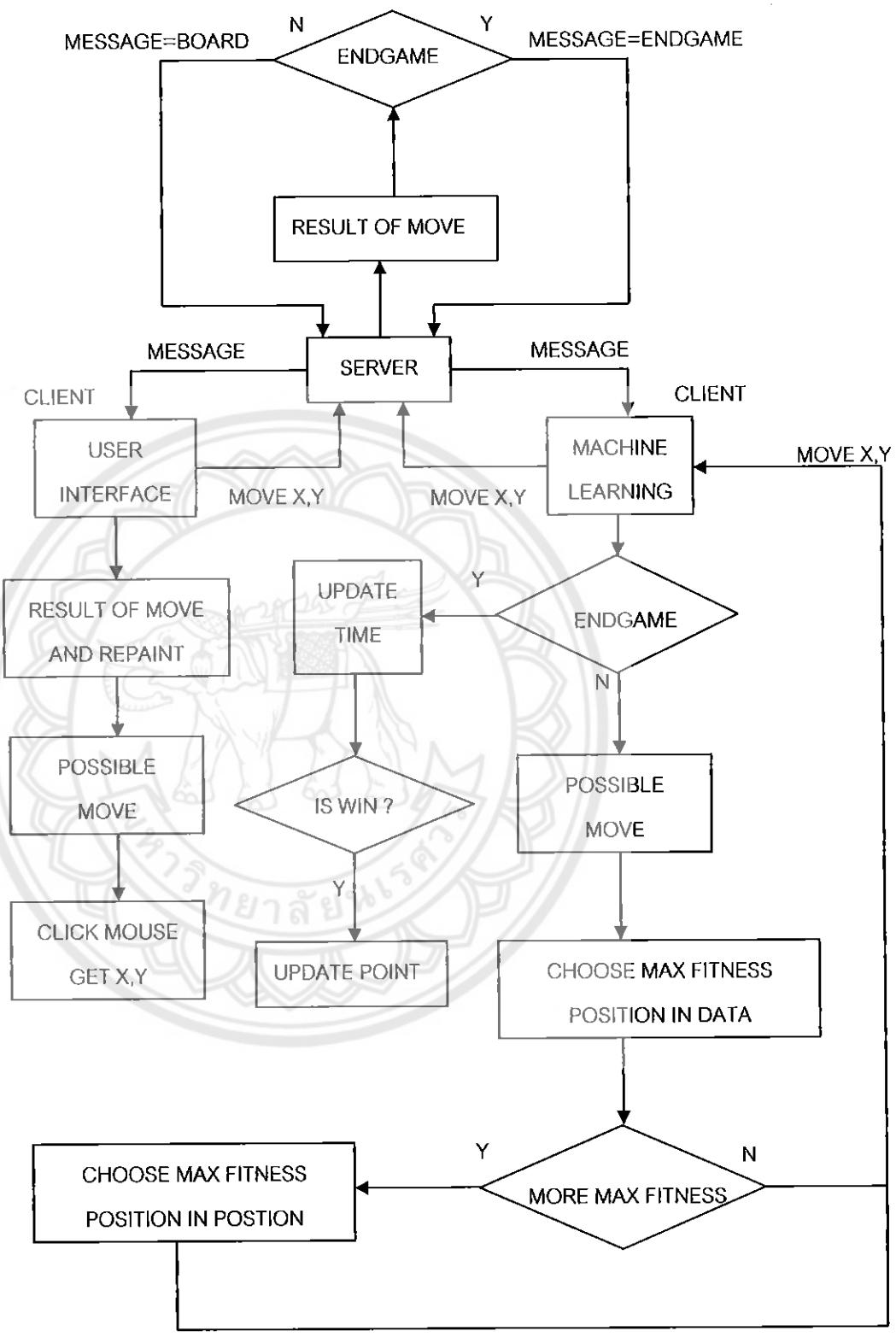
รูปที่ 3.2 แสดงตัวอย่างการเข้ารหัสบอร์ดเมื่อเริ่มเกม

เมื่อได้รหัสบอร์ดแล้วโปรแกรมจะทำการหาตำแหน่งทั้งหมดที่สามารถเดินได้ ซึ่งจากรูปข้างต้นจะเห็นตำแหน่งที่มีเครื่องหมายกากบาท ซึ่งก็คือตำแหน่งที่หมากสีขาว สามารถเดินได้นั่นเอง ซึ่งโปรแกรมของ Client จะเป็นฝ่ายคำนวณ เมื่อได้ตำแหน่งที่โปรแกรมสามารถเดินได้ทั้งหมดแล้วจะนำไปเก็บลงในฐานข้อมูล โดยจะมีการตรวจสอบก่อนว่ามีข้อมูลนี้ซ้ำหรือไม่ ถ้ามีอยู่ในฐานข้อมูลแล้วจะไม่ทำการเก็บซ้ำอีก

หลังจากนั้นโปรแกรมจะทำการเลือกตำแหน่งที่มีค่า Fitness มากที่สุด จากตาราง DATA ซึ่งคือค่าจำนวนครั้งที่ชนะต่อจำนวนครั้งที่เลือกใช้เป็นตำแหน่งที่เลือกเดิน และหากว่าค่า Fitness มากที่สุดมีเท่ากันหลายค่าจะทำการส่งค่าตำแหน่งนั้นๆที่มี Fitness มากที่สุด หาข้อมูลจากตาราง POSITION โดยใช้ค่าตำแหน่งเป็นรหัสดูว่า ที่ตำแหน่งนั้นมีค่า Fitness เท่าไรโดยดูทุกๆตำแหน่งและนำค่า Fitness มาเปรียบเทียบกัน และเลือกตำแหน่งที่มีค่า Fitness มากที่สุดซึ่งคือค่าจำนวนครั้งที่ชนะหารด้วยจำนวนครั้งที่เลือก

เมื่อได้ตำแหน่งที่จะเลือกเดินมาแล้ว โปรแกรมก็จะทำการเก็บตำแหน่งที่เลือกใช้ไว้เป็นวัตถุในคลาส OthelloStatus โดยเก็บรหัสบอร์ด ตำแหน่งแถว และตำแหน่งคอลัมน์ และเก็บไว้ในเวกเตอร์อีกทีก่อนจะส่งตำแหน่งที่เลือกเดินไปให้ Server

Server ก็ทำการหาผลลัพธ์หลังจากการเลือกตำแหน่ง และส่งค่าบอร์ดที่เปลี่ยนแปลงแล้วไปยัง Client อีกฝ่ายต่อไป โปรแกรมก็จะดำเนินไปเรื่อยๆจนกระทั่งจบเกม เมื่อจบเกมโปรแกรม Client ส่วนการเรียนรู้จะนำตำแหน่งที่เลือกใช้มาเพิ่มค่าจำนวนครั้งที่เลือกเดินด้วยการบวกอีก 1 โดยใช้รหัสบอร์ด ตำแหน่งแถว และตำแหน่งคอลัมน์ ในวัตถุเป็นเงื่อนไขอ้างอิงในตาราง DATA และเพิ่มค่าจำนวนครั้งที่เลือกเดินด้วยการบวกอีก 1 โดยใช้ตำแหน่งแถว และ ตำแหน่งคอลัมน์ในวัตถุเป็นเงื่อนไขอ้างอิงในตาราง POSITION และหากว่าเกมนั้นโปรแกรมเป็นฝ่ายชนะ ก็จะทำการเพิ่มจำนวนครั้งที่ชนะด้วยการบวกอีก 1 โดยใช้ตำแหน่งอ้างอิงอันเดียวกับการอ้างอิงเพื่อเพิ่มค่าจำนวนครั้งที่เลือกใช้ ทั้งในตาราง DATA และตาราง POSITION แต่หากว่าในเกมนั้นคอมพิวเตอร์เป็นฝ่ายแพ้หรือเสมอ จะไม่มีการเพิ่มค่าจำนวนครั้งที่ชนะ แต่ยังคงเพิ่มค่าจำนวนครั้งที่เลือกใช้เหมือนเดิม ซึ่งกระบวนการทั้งหมดแสดงดังรูป



รูปที่ 3.3 แสดงการทำงานของโปรแกรม

3.3 เครื่องมือที่ใช้พัฒนาโปรแกรม

เครื่องมือที่ใช้พัฒนาโปรแกรมประกอบด้วย

- compiler java version J2sdk 1.4.1_05
- My SQL
- My SQL –Front

3.4 การออกแบบฐานข้อมูล

การออกแบบฐานข้อมูลสำหรับโปรแกรมนี้ใช้ฐานข้อมูลเพียง 2 ตาราง คือ DATA กับ POSTION ซึ่งตาราง DATA ประกอบไปด้วย 6 คอลัมน์ด้วยกัน เป็นการเก็บข้อมูลการเดินที่เป็นไปได้ทั้งหมดจากแต่ละ Current_board และเก็บจำนวนครั้งที่นำตำแหน่งนั้นของ Current_board นั้นไปเลือกเดิน และเก็บจำนวนครั้งที่นำตำแหน่งนั้นของ Current_board นั้นไปเดินแล้วชนะ แสดงดังรูป

ตาราง DATA

Current_board	Row	Col	Time	Point

ตารางที่ 3.1 แสดงการออกแบบตาราง DATA

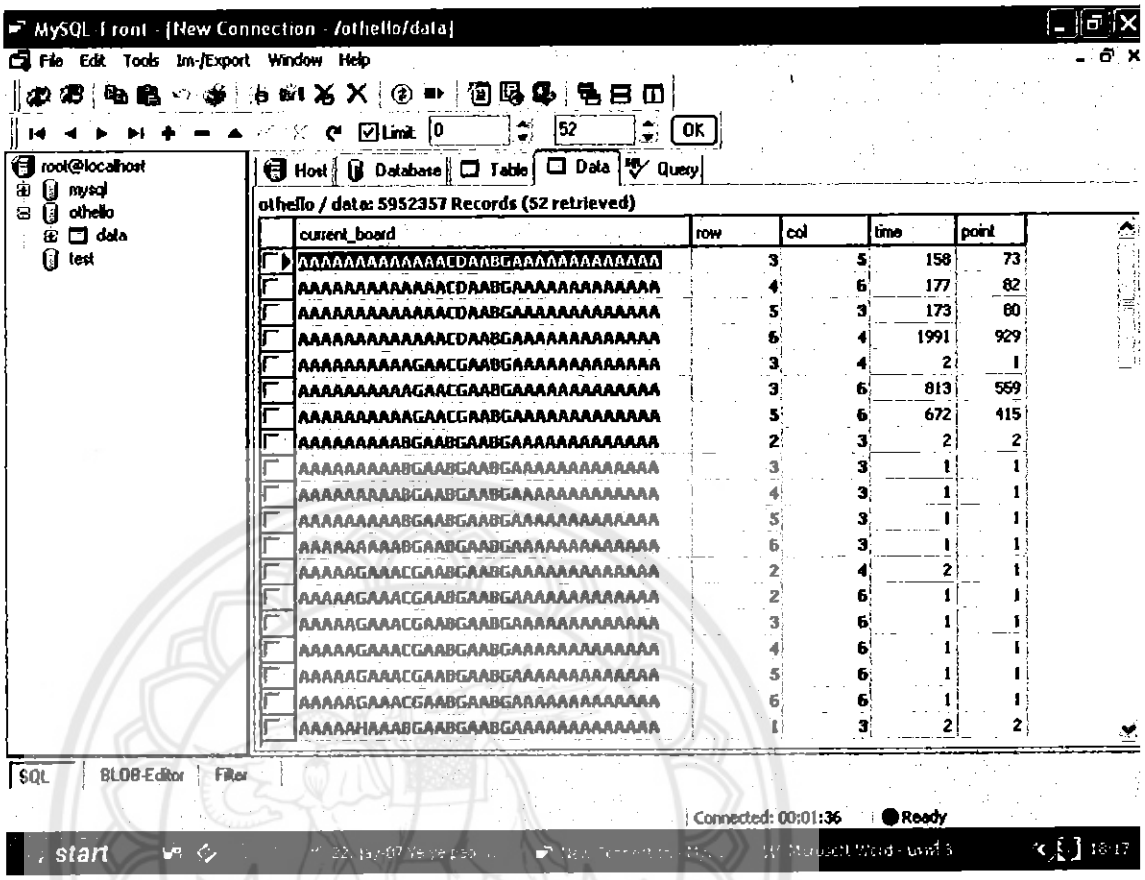
ชื่อฐานข้อมูล **OTHELLO**

ชื่อตาราง **DATA** ใช้เก็บข้อมูลการเล่นทั้งหมดของเกมในทุกๆครั้งที่มีการเล่น

รายละเอียดของแต่ละคอลัมน์มีดังนี้

Current_board	เก็บข้อมูลรหัสของตารางเกมในขณะนั้น ซึ่งเก็บเป็นตัวอักษร ขนาดไม่เกิน 32 ตัว
Row	เก็บข้อมูลตำแหน่งค่าแถว ซึ่งเก็บเป็นตัวเลข ขนาด 1 ตัว
Col	เก็บข้อมูลตำแหน่งค่าคอลัมน์ ซึ่งเก็บเป็นตัวเลข ขนาด 1 ตัว
Time	เก็บข้อมูลจำนวนครั้งที่เลือกข้อมูลแถวนั้นๆ (จำนวนครั้งที่เลือกตำแหน่งการเดิน) ซึ่งเก็บเป็นตัวเลข ขนาดสูงสุดไม่เกิน 10 หลัก
Point	เก็บข้อมูล ค่าจำนวนครั้งที่ชนะของตำแหน่งการเดิน ซึ่งเก็บเป็นตัวเลข จำนวนเต็มไม่เกิน 10หลัก

ซึ่งมีตัวอย่างการเก็บข้อมูลดังรูป



รูปที่ 3.4 แสดงตัวอย่างการเก็บข้อมูลในตาราง DATA

ส่วนตาราง POSITION นั้นจะเก็บความแตกต่างเพียง 64 แบบซึ่งก็คือจำนวน 64 ช่องในบอร์ด ซึ่งจะเก็บข้อมูลว่า ที่ตำแหน่งนั้นๆ ถูกนำไปใช้เลือกเดิน จำนวนกี่ครั้ง และตำแหน่งนั้นๆถูกนำไปเดินแล้วชนะกี่ครั้ง โดยที่ไม่คำนึงถึง Current_board ดังนั้นจะคือโอกาสในการนำไปเดินแล้วมีโอกาสชนะของตำแหน่งนั้นๆนั่นเอง ซึ่งโอกาสนั้นเกิดจาก การนำจำนวนครั้งที่ชนะหารด้วยจำนวนครั้งที่นำไปเดิน

แสดงดังรูป

ตาราง POSTION

row	col	time	point

ตารางที่ 3.2 แสดงการออกแบบตาราง POSTION

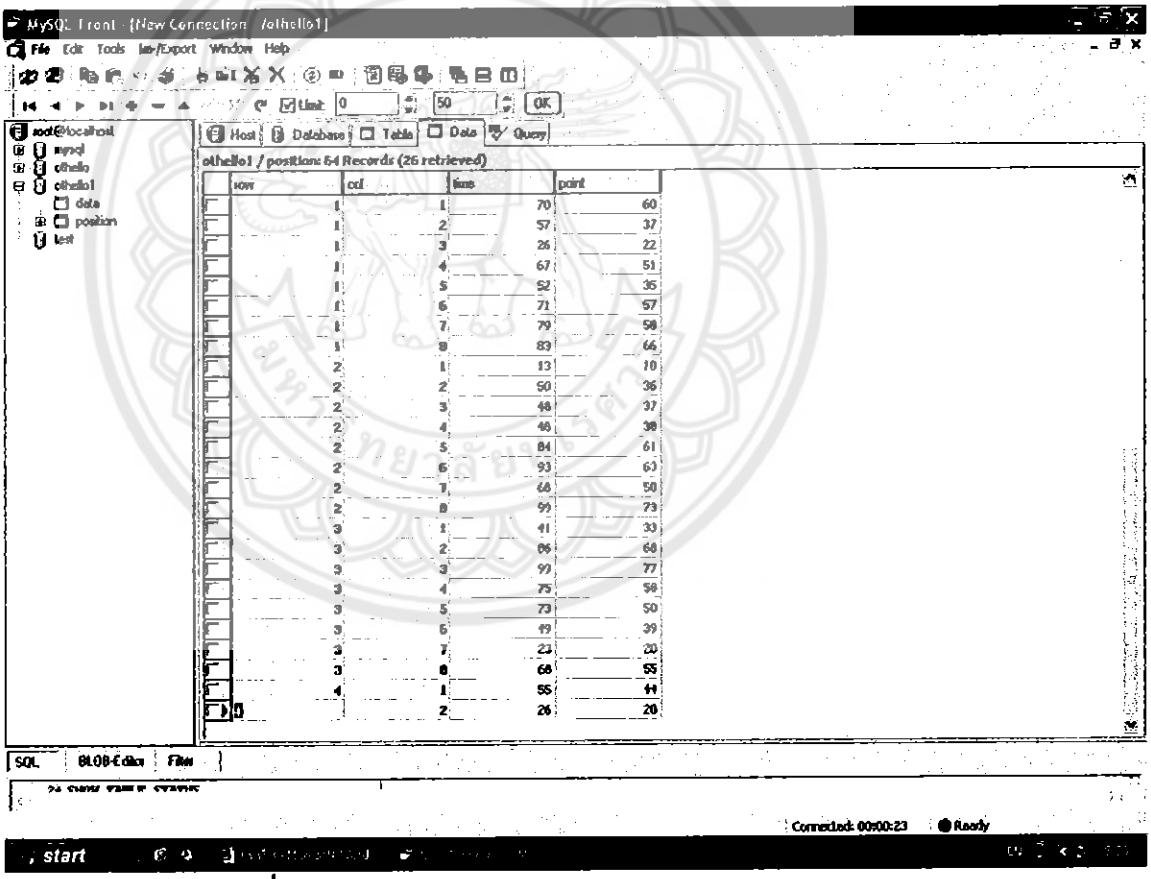
ชื่อฐานข้อมูล **OTHELLO**

ชื่อตาราง **POSITION** ใช้เก็บข้อมูลการเล่นทั้งหมดของเกมในทุกๆครั้งที่มีการเล่น

รายละเอียดของแต่ละคอลัมน์มีดังนี้

Row	เก็บข้อมูลตำแหน่งค่าแถว ซึ่งเก็บเป็นตัวเลข ขนาด 1 ตัว
Col	เก็บข้อมูลตำแหน่งค่าคอลัมน์ ซึ่งเก็บเป็นตัวเลข ขนาด 1 ตัว
Time	เก็บข้อมูลจำนวนครั้งที่เลือกข้อมูลแถวนั้นๆ (จำนวนครั้งที่เลือกตำแหน่งการเดิน) ซึ่งเก็บเป็นตัวเลข ขนาดสูงสุดไม่เกิน 10 หลัก
Point	เก็บข้อมูล ค่าจำนวนครั้งที่ชนะของตำแหน่งการเดิน ซึ่งเก็บเป็นตัวเลข จำนวนเต็มไม่เกิน 10หลัก

ซึ่งมีตัวอย่างการเก็บข้อมูลดังรูป



รูปที่ 3.5 แสดงตัวอย่างการเก็บข้อมูลในตาราง POSITION

ซึ่งค่า row,col ในฐานข้อมูลนั้นก่อนที่จะเก็บลงฐานข้อมูลจะต้องมีการบวกเพิ่มไปอีกหนึ่งค่าเสมอ เพราะว่าในบอร์ดเกมนั้นเก็บค่าเป็นอาเรย์ขนาด 8 คูณ 8 ซึ่งเริ่มที่ 0-7 แต่ว่าในฐานข้อมูลจะเริ่มที่ 1-9 เพื่อป้องกันปัญหาค่าเริ่มต้นในฐานข้อมูล

บทที่ 4
การทดสอบโปรแกรม

4.1 การออกแบบโปรแกรมเพื่อทดสอบเกม

เนื่องจากโปรแกรมต้องใช้การเก็บข้อมูลการเล่นในทุกๆรอบของการเล่น เพื่อใช้เป็นข้อมูลในการตัดสินใจในการเล่นครั้งต่อๆไป และเนื่องจากต้องมีการใช้ข้อมูลเป็นจำนวนมาก หากใช้การเล่นปกติก็คงทำให้การเก็บข้อมูลในการเล่นเป็นไปอย่างช้า ๆ จึงได้มีการพัฒนาโปรแกรมขึ้นมาอีกส่วนหนึ่งเพื่อไว้ใช้ในการทดสอบเกม โดยการพัฒนาโปรแกรมให้คอมพิวเตอร์เล่นเกมกับคอมพิวเตอร์กันเอง จึงมีการเขียนโปรแกรมในส่วน Client ขึ้นมาอีกแบบซึ่ง สมมุติให้โปรแกรมส่วนนี้คือคนที่มาเล่นกับเกมหรือโปรแกรมส่วนที่เกิดการเรียนรู้นั่นเอง

4.1.1 การออกแบบโปรแกรมส่วน Client

Client มีส่วนประกอบที่สำคัญ 2 ส่วนคือ

4.1.1.1 ส่วนการรับ-ส่งข้อความกับ Server

หากได้รับข้อความ START ก็จะเข้าใจความหมายว่าให้สามารถส่งตำแหน่งที่เลือกเดินไปได้ โดยส่งข้อความขึ้นต้นด้วย MOVE, แถว,หลัก,สีหมาก เช่น “MOVE, 4, 2, 2\0” พร้อมทั้งส่งข้อความEND ปิดท้ายด้วย

หากได้รับข้อความขึ้นต้นด้วย BOARD คือได้รับผลลัพธ์การเดินของฝ่ายตรงข้าม Client ก็จะทำการเก็บผลลัพธ์ไว้เพื่อใช้คำนวณหาการเดินของตนเองต่อไปและ หากได้รับ ENDGAME ก็เข้าใจความหมายว่าจบเกมแล้ว พร้อมทั้งรับคะแนนจาก Server เพื่อให้รู้ว่าฝ่ายใดเป็นฝ่ายชนะ

ส่วนอัลกอริทึมการเลือกตำแหน่งเดิน

โปรแกรม Client ในส่วนนี้จะมีการนำกฎมาใช้เลือกตำแหน่งของการเดิน ซึ่งกฎนี้จะมีโครงสร้างที่แน่นอน โดยมีแนวคิดคือ ใช้การพิจารณาจากตำแหน่งต่างๆที่อยู่บนบอร์ดโอเทโร่ ซึ่งจะพบว่าโอกาสที่เมื่อลงหมากเดินในตำแหน่งใดๆแล้วจะเกิดการชนะได้เท่าใด ซึ่งมีด้วยกัน กรณีดังนี้

- การลงที่ตำแหน่งมุมทั้ง 4 ด้านของบอร์ดทำให้เกิดโอกาสมากที่สุด เพราะหากลงมุมแล้วฝ่ายตรงข้ามจะไม่สามารถลงหมากครอบหมากของผู้เล่นได้ และมุมทำให้ยึดขอบได้ถึง 2 ด้านการลงที่ตำแหน่งชิดขอบทั้ง 4 ด้านของบอร์ด ทำให้เกิดโอกาสรองลงมา เพราะหากลงขอบแล้วฝ่ายตรงข้ามจะไม่สามารถลงหมากครอบหมากของผู้เล่นได้
- การลงที่ตำแหน่งกลาง เริ่มตั้งแต่แถวที่ 2 - 5 และ คอลัมน์ 2 - 5
- การลงที่ตำแหน่งก่อนถึงขอบ คือตั้งแต่แถวที่6 และคอลัมน์ที่ 6 เพราะ การลงตำแหน่งก่อนถึงขอบทำให้ฝ่ายตรงข้ามลงที่ตำแหน่งขอบและลงหมากครอบหมากของผู้เล่นได้

- การลงที่ตำแหน่งก่อนถึงมุม ทำให้เกิดโอกาสในการชนะน้อยที่สุด เพราะฝ่ายตรงข้ามสามารถลงที่ตำแหน่งมุมเพื่อครอบหมากของผู้เล่นได้

ซึ่งจากทุกๆกฎ ได้กำหนดค่าตัวเลขให้ตามลำดับความสำคัญของโอกาสที่จะชนะ ซึ่งค่าตัวเลขที่กำหนดนี้สามารถให้เป็นค่าเท่าใดก็ได้ แต่ต้องให้ค่าตัวเลขเพิ่มขึ้นตามโอกาสที่จะชนะ แสดงตัวอย่างดังรูป่

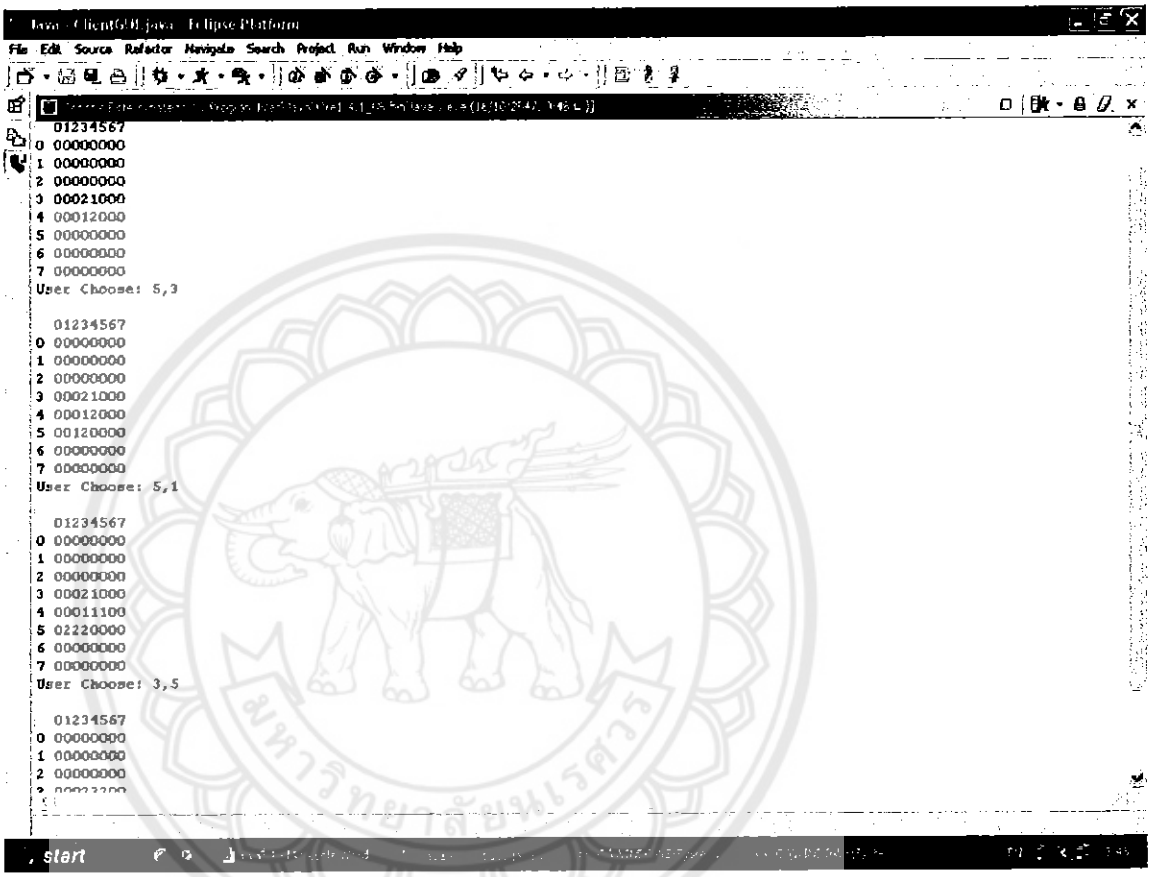
ตำแหน่ง	1	2	3	4	5	6	7	8
1	20	6	10	10	10	10	6	20
2	6	6	7	7	7	7	6	6
3	10	7	8	8	8	8	7	10
4	10	7	8	8	8	8	7	10
5	10	7	8	8	8	8	7	10
6	10	7	8	8	8	8	7	10
7	6	6	7	7	7	7	6	6
8	20	6	10	10	10	10	6	20

ตารางที่ 4.1 แสดงบอร์ดโอเทโร่และค่าประจำตำแหน่งของแต่ละตำแหน่งตามกฎ

4.2 การทดสอบการใช้งานเกม

4.2.1 ทดสอบการรับค่าจากการคลิกเมาส์

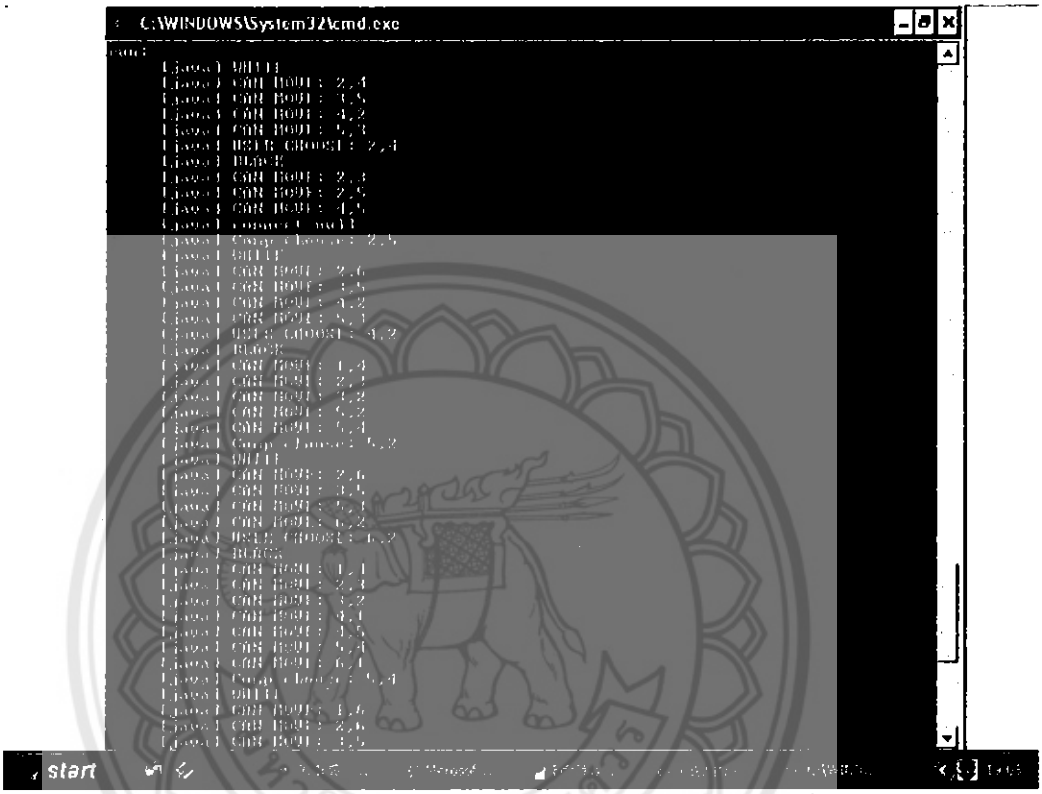
ทำการทดสอบว่าโปรแกรมสามารถรับค่าจากการคลิกเมาส์ในหน่วย Pixel แล้วสามารถเทียบในแถวและคอลัมน์ได้ถูกต้องหรือไม่ ดังรูป



รูปที่ 4.1 แสดงการรับค่าจากการคลิกเมาส์

4.2.2 ทดสอบการหาตำแหน่งที่สามารถเดินได้ทั้งหมด

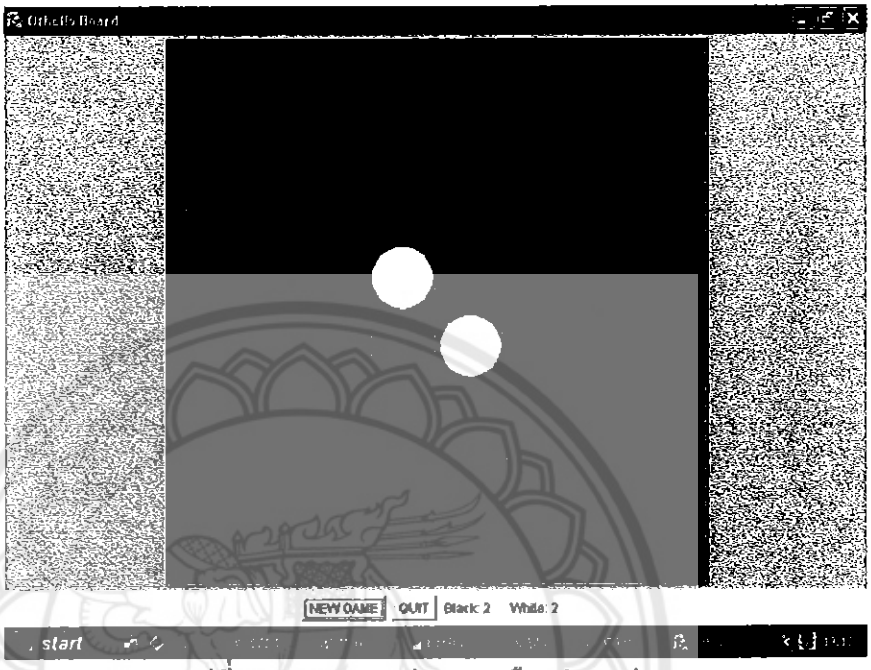
ทำการทดสอบว่าโปรแกรมสามารถคำนวณหาตำแหน่งที่สามารถเดินได้ทั้งฝ่ายผู้เล่นและฝ่ายคอมพิวเตอร์ได้ถูกต้องหรือไม่ ดังรูป



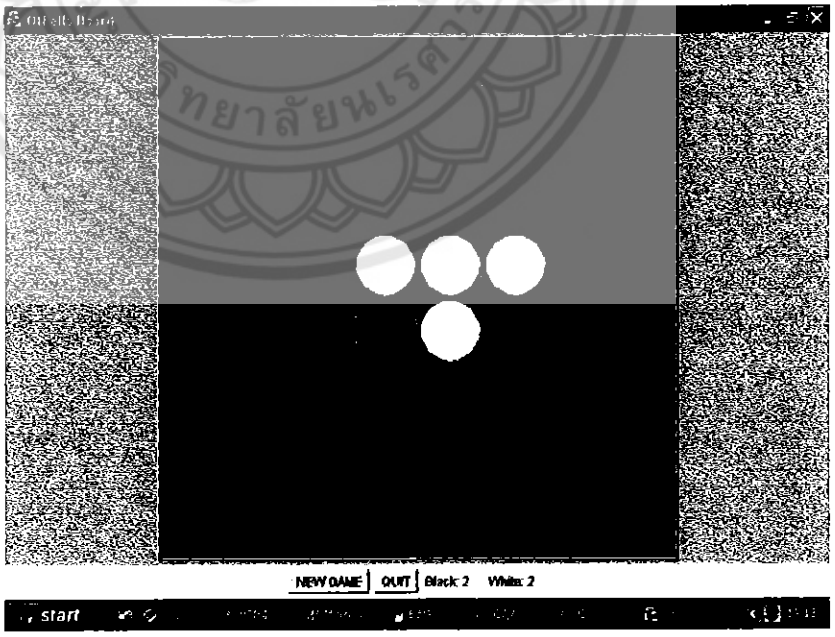
รูปที่ 4.2 แสดงการหาตำแหน่งที่สามารถเลือกเดินได้ทั้งหมด

4.2.3 ทดสอบการหาผลลัพธ์

ทำการทดสอบว่าโปรแกรมสามารถคำนวณหาผลลัพธ์ที่เกิดขึ้นหลังจากการเลือกตำแหน่งเดินแล้วได้ถูกต้องหรือไม่ ดังรูปตัวอย่างให้สมมติว่า จากบอร์ดนี้เลือกเดินที่ตำแหน่ง 3,5



รูปที่ 4.3 แสดงภาพก่อนการเลือกตำแหน่ง



รูปที่ 4.4 แสดงภาพหลังเลือกตำแหน่ง

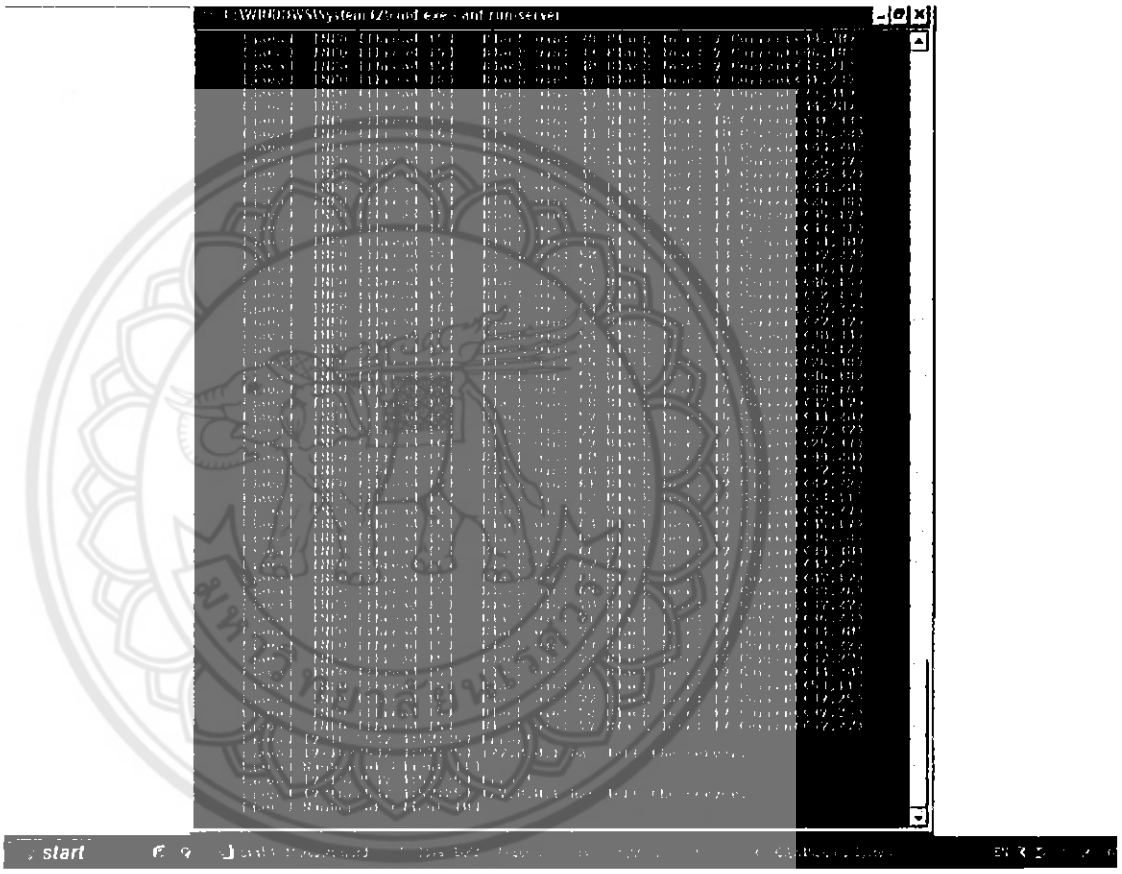
4.2.4 ทดสอบการเลือกตำแหน่งของคอมพิวเตอร์

ทำการทดสอบว่าโปรแกรมสามารถเลือกตำแหน่งได้ถูกต้องหรือไม่จากการคำนวณโดยให้เลือกตำแหน่งที่มีค่า Point / Time มากที่สุดจาก ตาราง DATA และหากเกิดกรณีมีค่า Fitness ที่มากที่สุดเท่าๆกันจะนำค่าแถวและคอลัมน์ไปหาค่า Fitness ที่มากที่สุดจากตาราง POSITION ซึ่งทดสอบว่าหากเกิดกรณีนี้ โปรแกรมสามารถเลือกตำแหน่งที่มีค่า Fitness มากที่สุดได้ถูกต้องหรือไม่ ดังรูป

รูปที่ 4.5 แสดงการเลือกตำแหน่งการเดินของคอมพิวเตอร์

4.2.5 ทดสอบการพัฒนาการเรียนรู้ของโปรแกรม

ทำการทดสอบว่าเมื่อให้โปรแกรมที่เกิดการเรียนรู้นั้นทำการเล่นเกมแข่งกับฝ่ายตรงข้ามที่มีรูปแบบการเดินแบบมีโครงสร้างเดิม แล้วมีการพัฒนาการเล่นได้ดีขึ้นจริง ซึ่งพบว่าเป็นไปตามหลักการที่ตั้งไว้ทุกประการ คือ ใน 100 รอบการแข่งแรกกับ 100 รอบการแข่งต่อมานั้นโปรแกรมที่เกิดการเรียนรู้นั้นมีการพัฒนาการเล่นให้ดีขึ้น มีจำนวนครั้งที่ชนะฝ่ายตรงข้ามเพิ่มขึ้น ดังรูป



รูปที่ 4.6 แสดงการพัฒนาการเล่นของโปรแกรมที่เกิดการเรียนรู้ใน 100 รอบแรก

รูปที่ 4.7 แสดงการพัฒนาการเล่นของโปรแกรมที่เกิดการเรียนรู้ใน 100 รอบต่อมา

บทที่ 5
บทสรุป

5.1 สรุปผลการทดลองที่ได้จากการพัฒนาโปรแกรม

วัตถุประสงค์ของโครงงานนี้ คือการพัฒนาโปรแกรมเกมโอเทโร่ให้มีการพัฒนาการเล่นให้เก่งขึ้นกว่าเดิมเมื่อพบกับรูปแบบการเล่นของฝ่ายตรงข้ามแบบเดิม โดยใช้ข้อมูลการเล่นจากรอบที่ผ่านมา ซึ่งพบว่าโปรแกรมสามารถรับตำแหน่งที่ผู้เล่นเลือกจากการคลิกเมาส์ได้ถูกต้อง,สามารถคำนวณหาตำแหน่งที่สามารถเลือกเดินได้ทั้งหมดทั้งฝ่ายคอมพิวเตอร์ฝ่ายผู้เล่นได้อย่างถูกต้องและแม่นยำ,สามารถคำนวณหาผลลัพธ์ที่เกิดขึ้นจากการเลือกตำแหน่งได้ถูกต้อง,สามารถเก็บข้อมูลลงฐานข้อมูลได้ครบถ้วนทุกๆเกมที่เล่น,สามารถเลือกข้อมูลจากฐานข้อมูลตามเงื่อนไขที่กำหนดได้ถูกต้อง,สามารถเลือกตำแหน่งที่จะเดินได้ถูกต้องตามกฎที่ให้เลือกตำแหน่งที่มีค่า Point / Time มากที่สุด,สามารถตรวจสอบการแพ้ชนะ และเพิ่มค่าจำนวนครั้งที่เลือกตำแหน่งมาใช้ และเพิ่มจำนวนครั้งที่ชนะได้อย่างถูกต้อง ซึ่งมีความสามารถครบตามวัตถุประสงค์พื้นฐานที่ผู้จัดทำต้องการทั้งสิ้น

5.2 การประเมินประสิทธิภาพ

เนื่องจากโปรแกรมการเรียนรู้นี้ ไม่มีการใช้ความรู้หรือกฎใดๆทั้งสิ้นในการพัฒนาการเรียนรู้ ซึ่งเพียงแต่อาศัยการฝึกฝนจากการแข่งเกมกับฝ่ายตรงข้าม ซึ่งจากการพิจารณาพบว่า ความแตกต่างของตารางบอร์ดที่เป็นไปได้นั้นมีจำนวนรูปแบบมากมาย ซึ่งถ้าคิดกรณีอย่างคร่าวๆ จะพบว่า แต่ละตำแหน่งของบอร์ดนั้นสามารถมีได้ 3 สถานะ ดังที่กล่าวข้างต้น และจำนวนของตำแหน่งบอร์ดทั้งหมดมี 64 ตำแหน่ง ดังนั้น จำนวนกรณีทั้งหมดที่เกิดขึ้นได้ คือ 3^{64} แบบ และยังมีกรณีที่ไม่สามารถเกิดขึ้นได้อีกหลายกรณี ดังนั้นจะพบว่าความแตกต่างของบอร์ดที่จะเกิดขึ้น และโปรแกรมการเรียนรู้นั้นต้องพบและได้แข่งขันนั้น มีจำนวนมากๆ ซึ่งในการทดสอบโปรแกรมนั้นไม่สามารถทำให้โปรแกรมการเรียนรู้ได้พบกับตารางบอร์ดที่แตกต่างกันได้จนครบทุกแบบในเวลาจำกัด อีกทั้งไม่เพียงแต่จะได้พบทุกแบบของความแตกต่างบอร์ดแล้วจะทำให้โปรแกรมสามารถพัฒนาการแข่งขันให้ดีขึ้นได้ การที่จะให้โปรแกรมพัฒนาการแข่งขันของตนให้ดีขึ้นได้ จำเป็นต้องพบกับรูปแบบเดิม 1 รูปแบบเป็นจำนวนหลายครั้ง สมมติ N ครั้ง ดังนั้น การพัฒนาการเรียนรู้ของโปรแกรมจำเป็นต้องแข่งขันเป็นจำนวน N คูณ กับความแตกต่างบอร์ดทั้งหมด อีกทั้งต้องใช้หน่วยความจำเป็นจำนวนมากในการเก็บฐานข้อมูลการเล่นทั้งหมดที่เกิดขึ้น

ผู้พัฒนาเล็งเห็นปัญหาในส่วนนี้และได้พิจารณาแล้วว่า เวลาและทรัพยากรที่ใช้พัฒนาโปรแกรมอาจไม่เพียงพอต่อการพัฒนาการแข่งขันของโปรแกรมให้เสร็จสิ้นทั้งหมด ดังนั้นจึงได้ทำการพิสูจน์การเรียนรู้ของโปรแกรมว่ามีการพัฒนาการเรียนรู้ขึ้นจริงหรือไม่ในการทำงานซ้ำเดิม

และพบว่า เมื่อให้โปรแกรมเล่นเกมกับฝ่ายตรงข้ามที่มีการเลือกตำแหน่งเดินแบบเดิม 100 ครั้ง ปรากฏว่า ใน 5 ครั้งแรก โปรแกรมเป็นฝ่ายแพ้ฝ่ายตรงข้าม และอีก 95 ครั้งต่อมาโปรแกรมเป็นฝ่ายชนะ ดังนั้นสรุปได้ว่า โปรแกรมมีการเรียนรู้ได้จริงและสามารถทำงานได้ดีกว่าเดิมในครั้งต่อๆไป ของงานเดิม ซึ่งจากโปรแกรมการเรียนรู้นี้สามารถพัฒนาได้ดีขึ้นจำนวน 95 เปอร์เซ็นต์ในงานเดิม

5.3 ปัญหาและการแก้ไข

5.3.1 ปัญหาการเปลี่ยนแปลงค่าจำนวนครั้งที่เลือกไปใช้และจำนวนครั้งที่ชนะ

จากการทำงานทั้งหมดที่ผ่านมาพบปัญหา คือ เมื่อให้ขณะที่คอมพิวเตอร์เลือกการเล่นได้แล้ว ให้ทำการเพิ่มจำนวนครั้งที่เลือกไปใช้ในขณะนั้นเลย แล้วถ้าหากว่าเกิดหยุดโปรแกรมกลางคัน หมายความว่าจะยังไม่รู้ผลแพ้ชนะ ดังนั้นจำนวนครั้งที่เลือกเพิ่มแล้ว แต่ไม่มีการเพิ่มจำนวนครั้งที่ชนะ จึงได้แก้ปัญหาด้วยการที่ เมื่อคอมพิวเตอร์เลือกการเล่นได้แล้วให้เก็บข้อมูลการเล่นที่เลือกมาจากฐานข้อมูลไว้ และเมื่อจบเกมจึงค่อยนำข้อมูลที่เก็บไว้ไปอ้างอิงในฐานข้อมูลเพื่อเพิ่มค่าจำนวนครั้งที่เลือกไปใช้

5.3.2 ปัญหาการฝึกฝนเกมให้เกิดการเรียนรู้กับข้อมูลจำนวนมาก

เนื่องจากรูปแบบ Current_board ของเกมนั้นมีจำนวนมาก มากจนขนาดที่ต้องใช้การเก็บข้อมูลเยอะมากและใช้เวลาในการฝึกฝนเกมเป็นเวลายาวนานมาก อีกทั้งยังต้องมีการฝึกฝนเกมให้พบเจอกับรูปแบบเกมที่หลากหลายมากๆ ซึ่งทำให้การฝึกฝนเกมใช้เวลาและทรัพยากรมาก จึงได้แก้ไขปัญหาด้วยการทำการฝึกฝนเกมด้วยโปรแกรมที่มีโครงสร้างรูปแบบการเลือกตำแหน่งเดินอย่างแน่นอน ทำให้โปรแกรมที่เกิดการเรียนรู้นั้นพบกับการเดินของฝ่ายตรงข้ามแบบเดิมๆ แล้วทำการเก็บจำนวนครั้งที่โปรแกรมที่เกิดการเรียนรู้ชนะ พบว่าในการเดินแบบเดิมของฝ่ายตรงข้ามนั้น เมื่อโปรแกรมที่เกิดการเรียนรู้พบกับการเดินแบบเดิมอีก ทำให้โปรแกรมนั้นสามารถพัฒนาและเลือกตำแหน่งที่ควรเดินได้ดีขึ้น ทำให้เกิดจำนวนครั้งที่มากขึ้น เช่น จากการทดสอบ ที่ 100 รอบการเล่นแรก โปรแกรมที่เกิดการเรียนรู้สามารถชนะได้ 74 ครั้ง และทำการทดสอบอีก 100 รอบ พบว่าโปรแกรมที่เกิดการเรียนรู้สามารถชนะได้ 81 ครั้งและเมื่อทำการทดสอบอีกเรื่อยๆ ก็พบว่าจำนวนครั้งที่ชนะนั้นมีจำนวนเพิ่มขึ้น ดังนั้นจึงทำให้พิสูจน์ได้ว่า เมื่อโปรแกรมที่เกิดการเรียนรู้ พบกับการเดินหมากแบบเดิม ทำให้ในครั้งต่อๆไปนั้นโปรแกรมที่เกิดการเรียนรู้มีการเล่นที่มีโอกาสชนะมากขึ้น

5.4 แนวทางในการพัฒนาครั้งต่อไป

ในการพัฒนาครั้งต่อไป ได้มีแนวคิดว่าอยากพัฒนาให้การเก็บข้อมูลมีการเก็บน้อยลง จะได้ไม่เปลืองทรัพยากรคอมพิวเตอร์ คือมีการพัฒนาในด้านการบีบอัดฐานข้อมูลให้ใช้ทรัพยากรในการเก็บน้อยลง แต่เก็บข้อมูลได้ครบถ้วนเหมือนเดิม และทำให้การเข้าถึงข้อมูลได้เร็วขึ้น และอยากพัฒนาให้มีการวิเคราะห์เพิ่มเติมจากข้อมูลที่ได้รับ เพราะอาจทำให้เกิดผลลัพธ์ใหม่ๆได้ และช่วยให้ไม่ต้องเก็บข้อมูลเป็นจำนวนมาก และยังเป็นการทำให้คอมพิวเตอร์รู้จักเรียนรู้มากขึ้น

เอกสารอ้างอิง

[1] กิตติ ภักดีวัฒนะกุล. **JAVA ฉบับโปรแกรมเมอร์.** กรุงเทพมหานคร: เคทีพี คอมพ์ แอนด์ คอนซัลท์. 2544

[2] ดร.วีระศักดิ์ ซึงถาวร. **JAVA PROGRAMMING.** กรุงเทพมหานคร: ซีเอ็ดยูเคชั่น. 2543

[3] ผศ.ดร.วรรณวิภา ติตถะสิริ. **คู่มือเรียน SQL ด้วยตนเอง.** กรุงเทพมหานคร: โปรวิชั่น.2545

[4] สงกรานต์ ทองสว่าง. **My SQL ระบบฐานข้อมูลสำหรับอินเตอร์เน็ต.** กรุงเทพมหานคร: ซีเอ็ดยูเคชั่น. 2545

ประวัติผู้เขียนโครงงาน

ชื่อ นางสาวจิราพร พุกสุข

ภูมิลำเนา 234/1 ม.7 ต.คลองขลุง อ.คลองขลุง จ.กำแพงเพชร

ประวัติการศึกษา

- จบระดับมัธยมศึกษาตอนปลายจากโรงเรียนสตรี-นครสวรรค์
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 4

สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

มหาวิทยาลัยนเรศวร

E-mail : guitardong@hotmail.com

ชื่อ นายตรัยเทพ ชาญธัญกร

ภูมิลำเนา 44 ถนนแสงอัสนี ต.ชุมแสง อ.ชุมแสง จ.นครสวรรค์

ประวัติการศึกษา

- จบระดับมัธยมศึกษาตอนปลายจากโรงเรียนชุมแสงชนูทิศ
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 4

สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

มหาวิทยาลัยนเรศวร

E-mail : engineeringjack@hotmail.com