รหัสโครงการ SUT7-705-47-12-49

รายงานการวิจัย

การพัฒนาโปรแกรมเพื่อค้นหาข้อผิดพลาดทางตรรกะของภาษาซี

(Development of Detectable Logic Error Program in C Language)

หัวหน้าโครงการ
ผู้ช่วยศาสตราจารย์ ดร. พิชโยทัย มหัทธนาภิวัฒน์
สาขาวิชาวิศวกรรมคอมพิวเตอร์
สำนักวิชาวิศวกรรมศาสตร์
มหาวิทยาลัยเทคโนโลยีสุรนารี

ได้รับทุนอุดหนุนการวิจัยจากมหาวิทยาลัยเทคโนโลยีสุรนารี ปีงบประมาณ พ.ศ. 2547
ผลงานวิจัยเป็นความรับผิดชอบของหัวหน้าโครงการวิจัยแต่เพียงผู้เดียว

กุมภาพันธ์ 2548

กิตติกรรมประกาศ

ขอขอบคุณ หน่วยงานที่กำหนดทุนสนับสนุนการทำวิจัยจากมหาวิทยาลัยเทคโนโลยีสุรนารี ประจำปีงบประมาณ 2547 และ คุณสิทรา ชัยมงคล ที่ได้ช่วยเขียนภาษา php ทำให้โปรแกรมการค้นหาข้อผิดพลาดเชิงตรรกะของภาษาซีทำงานได้ตรงจุดมุ่งหมาย

ขอขอบคุณ อาจารย์สมพันธุ์ ชาญศิลป์ และ อาจารย์ ดร. ดะชา ชาญศิลป์ ที่ได้ช่วยให้ข้อเสนอแนะ และตรวจสอบผลการทำงานของโปรแกรมที่ได้พัฒนาขึ้น สุดท้ายนี้ขอขอบคุณนักศึกษาที่เรียนวิชา การโปรแกรมคอมพิวเตอร์ ซึ่งได้ทดลองใช้โปรแกรมในการทำปฏิบัติการ ทำให้เห็นผลการทำงานจริงของโปรแกรมดังกล่าว

ผศ.ดร.พิชโยทัย มหัทธนาภิวัฒน์
หัวหน้าโครงการวิจัย
กุมภาพันธ์ 2548

บทคัดย่อ

ภาษาซี เป็นภาษาแบบโครงสร้างและเป็นที่นิยมใช้กันทั่วไปในวงการวิทยาศาสตร์และเทคโนโลยี โดยภาษาซีเป็นภาษาที่ได้ออกแบบให้มีความยืดหยุ่นสูงในการเขียนโปรแกรม แต่ความยืดหยุ่นนี้ทำให้การตรวจจับข้อผิดพลาดบางประการทำไม่ได้ในขณะแปลเป็นภาษาเครื่อง (compilation) ซึ่งผู้เขียนโปรแกรมจะต้องทราบเอง ดังนั้นโปรแกรมที่พัฒนาเพื่อค้นหาข้อผิดพลาดทางตรรกะของภาษาซี จะมุ่งเน้นเพื่อค้นหาข้อผิดพลาดที่ตัวแปลภาษาไม่สามารถตรวจจับได้ โดยที่ถ้าไม่ตรวจจับอาจทำให้ผลการทำงานของโปรแกรมที่เขียนขึ้นมาผิดพลาดไปจากความต้องการได้ โปรแกรมที่พัฒนาดังกล่าวสามารถทำการตรวจจับข้อผิดพลาดได้ดี และยังชี้ให้เห็นถึงลวามผิดพลาดนั้นซึ่งจะช่วยให้ผู้เขียนโปรแกรม โดยเฉพาะอย่างยิ่งผู้เริ่มเขียน สามารถเห็นถึงข้อผิดพลาดและเรียนรู้การเขียนโปรแกรมภาษาซีได้เร็วขึ้น

ABSTRACT

C is a structure programming language that is widely used in science and technology. It is designed to have flexibility in writing code but its flexibility made it hard to detect some errors during compilation. Therefore, the programmer must examine the code by himself if the compiler cannot detect some errors.

The developed program to detect some logic errors will help the programmer to detect some errors, especially the location of the statements in C, so that the source program will run as required. The developed program works well and helps the programmer, especially the novice, to locate and learn about errors with suggestions.

สารบัญ

สารบัญภาพ

หน้า

บทที่ 1
บทนำ

1.1 ความสำคัญและที่มาของปัญหาการวิจัย

เทคโนโลยีสารสนเทศเป็นหนึ่งในปรัชญาความเป็นเลิศในภารกิจการเรียนสอนของมหาวิทยาลัยเทคโนโลยีสุรนารี ดังนั้น วิชาเทคโนโลยีสารสนเทศ 1 จึงถูกบรรจุในทุกหลักสูตรของการเรียนการสอน และสำหรับนักศึกษาวิชาวิศวกรรมศาสตร์นั้น สำนักวิชาวิศวกรรมศาสตร์ได้ตระหนักถึงความสามารถของบัณฑิตที่จะต้องมีวิธีการคิดและสามารถเขียนโปรแกรมทางคอมพิวเตอร์ได้ จึงบรรจุวิชาการโปรแกรมคอมพิวเตอร์ไว้ในทุกหลักสูตรของสาขาวิชาของสำนักวิชา

เมื่อพิจารณาเนื้อหาของวิชา เทคโนโลยีสารสนเทศ 1 ในส่วนของการเขียนโปรแกรม และวิชา การโปรแกรมคอมพิวเตอร์แล้ว จะเห็นว่า มีการใช้ภาษาซีเป็นเครื่องมือเรียนรู้ในการเขียนโปรแกรม ภาษาซีมีข้อดีในแง่ของการเป็นภาษาโครงสร้าง ทำให้ผู้เขียนโปรแกรมต้องมีหลักการคิดที่เป็นเหตุเป็นผล และภาษาซียังเป็นภาษาที่มีการใช้งานกันอยู่ทั่วไป ทั้งในด้าน วิทยาศาสตร์ คณิตศาสตร์ และ วิศวกรรมศาสตร์ ประกอบกับการเรียนภาษาซี จะเป็นพื้นฐานที่ดีในการศึกษาภาษาคอมพิวเตอร์อื่นๆได้

นักศึกษาที่เรียนภาษาซี จะต้องเข้าใจในรูปแบบไวยากรณ์ของภาษา เพื่อนำมาเขียนเป็นโปรแกรมให้ทำงานได้ตามแนวคิดที่ตนเองต้องการ โดยจะต้องเขียนโปรแกรมให้ถูกต้องตามหลักไวยากรณ์ซึ่งตรวจสอบได้โดยการแปลเป็นภาษาเครื่องก่อนจะสั่งให้โปรแกรมทำงาน

ในบางกรณี เราสามารถเขียนโปรแกรมภาษาซี และสามารถตรวจสอบหลักไวยากรณ์ได้ถูกต้อง แต่โปรแกรมยังทำงานไม่เป็นไปตามที่เราต้องการจะให้เป็น ทั้งนี้เป็นเพราะข้อผิดพลาดทางตรรกะ ซึ่งมี 2 กรณีคือ

1. ข้อผิดพลาดทางตรรกะที่เกิดจากการวางรูปคำสั่ง
2. ข้อผิดพลาดทางตรรกะที่เกิดจากการใส่สูตรผิด หรือการเรียงคำสั่งผิด

การวิจัยนี้มุ่งเน้นพัฒนาโปรแกรมเพื่อเป็นเครื่องมือช่วยค้นหาข้อผิดพลาดทางตรรกะประเภทที่ 1 อันจะเป็นประโยชน์แก่ผู้เรียนภาษาซี หรือแม้แต่ผู้ใช้ภาษาซีโดยทั่วไป โดยโปรแกรมที่พัฒนาแล้วจะชี้ให้ผู้เขียนโปรแกรมภาษาซีเห็นจุดต่างๆที่น่าจะเกิดข้อผิดพลาดขึ้น เพื่อให้ผู้เขียนโปรแกรมแก้ไขให้ถูกต้อง

1.2 วัตถุประสงค์ของการวิจัย

เพื่อพัฒนาโปรแกรมเพื่อค้นหาข้อผิดพลาดทางตรรกะด้านการวางรูปคำสั่งของภาษาซี โดยโปรแกรมที่พัฒนาแล้ว จะมีคุณสมบัติ ดังนี้

1. ทำการนำ source code ของโปรแกรมที่ผ่านการตรวจสอบไวยากรณ์ถูกต้องแล้ว มารันโปรแกรมที่พัฒนานี้ เพื่อหาข้อผิดพลาดทางตรรกะด้านการวางรูปคำสั่ง ที่อาจจะเกิดขึ้น
2. โปรแกรมจะแสดงตำแหน่งต่างๆ ที่น่าจะเกิดข้อผิดพลาดขึ้น พร้อมคำแนะนำผู้เขียนโปรแกรมภาษาซีสามารถทำตามข้อแนะนำของโปรแกรมนี้

1.3 ประโยชน์ที่ได้รับจากการวิจัย

1. เป็นการสร้างและพัฒนาองค์ความรู้แก่ นักวิจัย อาจารย์ และ นักศึกษา
2. เป็นเครื่องมือช่วยตรวจสอบหลักการเขียนโปรแกรมด้วยภาษาซีให้ถูกต้องยิ่งขึ้น
3. ห้องปฏิบัติการคอมพิวเตอร์ที่มีการสอนการเขียนโปรแกรมด้วยภาษาซี สามารถนำโปรแกรมที่พัฒนานี้ไปใช้ประโยชน์ในเชิงปฏิบัติได้

บทที่ 2
แนวทางการวิจัยและขอบเขตการวิจัย

2.1 แนวทางการวิจัย

ภาษาซี จัดเป็นภาษาคอมพิวเตอร์ระดับสูงภาษาหนึ่งซึ่งเกิดขึ้นมาร่วม 30 ปีมาแล้วโดย Dennis Ritchie ณ ห้องปฏิบัติการ Bell (Dietel, 1994) เพื่อเป็นภาษาในการพัฒนาระบบปฏิบัติการ UNIX การที่ภาษานี้ได้รับความนิยมและยังคงใช้กันอยู่ในหมู่นักเขียนโปรแกรมทั่วโลก เป็นเพราะว่า

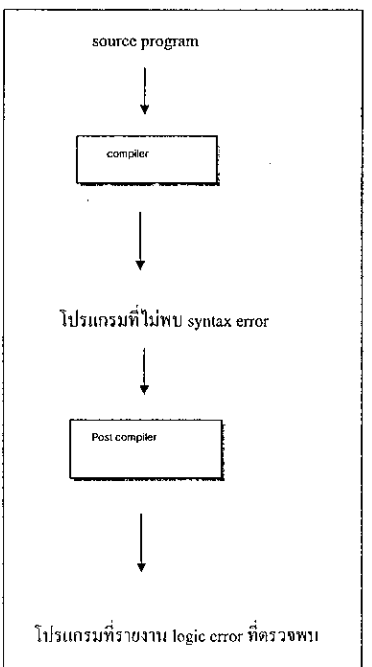
1. ภาษาซีไม่ขึ้นกับฮาร์ดแวร์ นั่นคือโปรแกรมที่เขียนกับเครื่องคอมพิวเตอร์เครื่องหนึ่ง สามารถนำไปใช้งานกับคอมพิวเตอร์อีกเครื่องหนึ่งได้ ไม่ว่าเครื่องคอมพิวเตอร์เครื่องนั้นจะเป็นระดับไมโครคอมพิวเตอร์ มินิคอมพิวเตอร์ หรือ เมนเฟรมคอมพิวเตอร์
2. โปรแกรมที่เขียนด้วยภาษาซี สามารถทำงานได้เร็วกว่าโปรแกรมที่พัฒนาด้วยภาษาอื่น
3. ภาษาซี มีฟังก์ชันมาตรฐานจำนวนมาก ซึ่งสนับสนุนการพัฒนาโปรแกรมประยุกต์ใช้งานทางด้านวิทยาศาสตร์ และ วิศวกรรมศาสตร์
4. ภาษาซี สนับสนุนลักษณะการเขียนโปรแกรมเป็นฟังก์ชัน ทำให้พัฒนาโปรแกรมเป็นส่วนๆได้ง่าย และยังสามารถ นำฟังก์ชันของผู้พัฒนาอื่นๆเข้ามารวมในโปรแกรมได้
5. ภาษาซี สามารถเขียนได้ในรูปแบบโปรแกรมแบบมีโครงสร้าง อันประกอบด้วยคุณลักษณะการทำงานแบบเรียงลำดับ การทำงานแบบมีเงื่อนไข และ การทำงานแบบวนซ้ำ (Kruse, Tondo, Leung, 1997).

สถาบันการศึกษาจำนวนมากที่มีการเรียนการสอนด้านวิทยาศาสตร์และเทคโนโลยีได้ตระหนักถึงความสำคัญ ในการเรียนรู้หลักการโปรแกรมคอมพิวเตอร์ และภาษาซีเป็นภาษาคอมพิวเตอร์ภาษาหนึ่งที่ถูกนำมาใช้ในการศึกษา

หลักการเขียนโปรแกรมภาษาซีต้องใช้โปรแกรมพวกเอดิเตอร์มาบันทึกชุดคำสั่งลงแฟ้ม และต้องระบุนามสกุลให้แฟ้มเป็น .C ในที่นี้จะเรียกว่าโปรแกรมดั้งเดิม (source program) หลังจากเขียนโปรแกรมภาษาซีเสร็จ ก่อนจะให้คอมพิวเตอร์ทำงานตามโปรแกรม ต้องทำการแปลงโปรแกรมภาษาซีให้เป็นภาษาเครื่องเรียกว่าการคอมไพล์ (compilation) หากมีข้อผิดพลาดเกิดขึ้นในช่วงคอมไพล์นี้ จะเรียกว่าข้อผิดพลาดนั้นว่าข้อผิดพลาดทางไวยากรณ์ของภาษา (syntax error) ซึ่งจะต้องได้รับการแก้ไขก่อน โดยเข้าไปแก้ไขชุดคำสั่งในแฟ้มโปรแกรมดั้งเดิมภาษาซี (source program) แล้วทำการคอมไพล์ใหม่ เมื่อโปรแกรมปราศจากข้อผิดพลาดทางไวยากรณ์แล้วจึงจะได้โปรแกรมภาษาเครื่อง ที่สามารถสั่งให้คอมพิวเตอร์ทำงานตามโปรแกรมได้ (execute program)

แต่ผลลัพธ์จากการทำงานของโปรแกรมอาจได้ไม่ตรงตามความต้องการ ข้อผิดพลาดที่เกิดขึ้นในช่วงนี้เรียกว่าข้อผิดพลาดเชิงตรรกะ (logic error) ซึ่งข้อผิดพลาดบางอย่างยากที่จะถูกตรวจพบได้ เช่น การกำหนดสูตรในการคำนวณผิด หรือ การใส่ลำดับคำสั่งผิด เป็นต้น และมีข้อผิดพลาดเชิงตรรกะบางอย่างที่สามารถจะตรวจพบได้ เช่นอาจเกิดจากการใส่หรือวางตำแหน่งสัญลักษณ์ผิด แต่ผ่านการคอมไพล์

ดังนั้นจะขอยกตัวอย่างข้อผิดพลาดเชิงตรรกะที่สามารถตรวจพบได้บางส่วน ที่น่าจะเป็นประโยชน์แก่ผู้เริ่มศึกษาภาษาซี หรือแม้แต่ผู้มีประสบการณ์แล้วได้ตระหนักถึง เพื่อการเขียนโปรแกรมภาษาซีได้อย่างถูกต้อง ซึ่งข้อผิดพลาดดังกล่าวจะได้นำมาพัฒนาเครื่องมือช่วยหลังการคอมไพล์ (Sommerville, 2001) เพื่อให้ได้โปรแกรมถูกต้องยิ่งขึ้น ดังรูปที่ 2.1



รูปที่ 2.1 แสดงการได้ซึ่งโปรแกรมที่ถูกต้องจากการกรองข้อผิดพลาดเชิงไวยากรณ์ และข้อผิดพลาดเชิงตรรกะ

ข้อผิดพลาดเชิงตรรกะที่โปรแกรม C post compiler ควรตรวจพบ (Mahatthanapiwat , 2003) มีดังนี้

การใส่เครื่องหมายแสดงการจบคำสั่ง

คำสั่งทุกคำสั่งในภาษาซีสามารถปิดท้ายด้วยเครื่องหมาย ; ได้ แต่ในบางกรณี การใส่เครื่องหมาย ; ปิดท้ายคำสั่งก็ให้ผลลัพธ์ผิดพลาดได้ โดยที่ตัวแปลภาษาซีจะไม่พบข้อผิดพลาดทางไวยากรณ์ เช่น

```
if(x == 3);
    printf("Three\n");
```

เมื่อคอมพิวเตอร์ทำงานตามโปรแกรม จะพิมพ์ข้อความ Three ออกมาทางจอภาพเสมอไม่ว่า x จะมีค่าเป็นอะไรก็ตาม ทั้งนี้เพราะเกิดความเข้าใจว่าถ้าเงื่อนไขหลังคำสั่ง if เป็นจริงให้ทำคำสั่งว่าง (null statement) 1 คำสั่ง ซึ่งผิดวัตถุประสงค์ที่ต้องการให้พิมพ์ข้อความ Three เมื่อ x มีค่าเท่ากับ 3 เท่านั้น ดังนั้นโปรแกรมที่ถูกต้องควรแก้ไขเป็น

```
if(x == 3)
   printf("Three\n");
```

นอกจากนี้การนำเครื่องหมาย ; ไปวางไว้ข้างหลังคำสั่งเกี่ยวกับการวนซ้ำ เช่น คำสั่ง for และ while ก็จะทำให้คอมพิวเตอร์ทำงานให้ผลผิดพลาดเช่นกัน เช่น

```
for( n = 1; n <= 10; n++);
   printf("Test\n");
```

การสั่งงานลักษณะนี้จะทำให้เกิดการวนซ้ำ 10 รอบโดยแต่ละรอบจะทำคำสั่งว่าง หลังจากจบการวนซ้ำแล้วจึงทำคำสั่งถัดไปคือแสดงข้อความ Test ครั้งเดียว ซึ่งผิดวัตถุประสงค์ที่เราต้องการให้พิมพ์ข้อความ Test 10 ครั้ง ดังนั้นควรแก้ไขโปรแกรมให้ถูกต้องโดยตัดเครื่องหมาย ; หลังวงเล็บปิดของคำสั่ง for ออก

```
for( n = 1; n <= 10; n++)
   printf("Test\n");
```

การกำหนดขอบเขตของชุดคำสั่งด้วย { }

ถ้าต้องการให้คอมพิวเตอร์ทำงานเป็นชุดคำสั่ง (มีคำสั่งมากกว่า 1 คำสั่ง) ในกรณีที่เงื่อนไขเป็นจริงหรือเท็จ เช่นกรณีเงื่อนไขของคำสั่ง if-else หรือ คำสั่งเกี่ยวกับการวนซ้ำ เช่น for, while, do-while จำเป็นต้องกำหนดเครื่องหมายวงเล็บปีกกาครอบคลุมชุดคำสั่งที่ต้องการนั้น ถ้าไม่ใส่วงเล็บปีกกาคร่อมจะทำให้คอมพิวเตอร์ทำงานผิดพลาด เช่น

```
if(grade < 50)
  printf("you failed\n");
  printf("you have to work hard\n");
```

เมื่อคอมไพล์ จะไม่แสดงข้อผิดพลาดทางไวยากรณ์ และเมื่อโปรแกรมทำงาน จะเข้าใจว่าผู้เขียนโปรแกรมต้องการให้ คำสั่ง printf แรกเพียงคำสั่งเดียวทำงานเมื่อเงื่อนไขเป็นจริง (ค่าของ grade น้อยกว่า 50 จริง) ซึ่งจริงๆแล้วผู้เขียนต้องการให้คำสั่ง printf ทั้งสองทำงานเมื่อเงื่อนไขของ if เป็นจริงเท่านั้น ดังนั้นการไม่ใช้เครื่องหมาย { } กำหนดชุดคำสั่งจึงทำให้คำสั่ง printf คำสั่งที่สอง ทำงานเสมอไม่ว่าค่าของ grade จะเป็นเท่าไรก็ตาม ทางแก้โปรแกรมที่ถูกต้องคือ

```
if(grade < 50) {
  printf("you failed\n");
  printf("you have to work hard\n");
}
```

การหารเลขจำนวนเต็ม

ในเรื่องของการคำนวณทางคณิตศาสตร์ ภาษาซีมีหลักการว่า เลขจำนวนเต็ม กระทำการทางคณิตศาสตร์ กับ เลขจำนวนเต็ม จะได้ผลลัพธ์เป็นจำนวนเต็ม ดังนั้นผู้เขียนโปรแกรมต้องระมัดระวังเมื่อมีการคำนวณ โดยเฉพาะอย่างยิ่ง การหารจะทำให้เกิดการปัดเศษทิ้งหมด เช่น จากสูตรการหาพื้นที่ของสามเหลี่ยม ถ้า base และ height เป็นตัวแปรที่เก็บค่าความยาวของฐานและส่วนสูงของสามเหลี่ยมตามลำดับ โดย area เป็นตัวแปรที่เก็บผลลัพธ์จากการคำนวณพื้นที่สามเหลี่ยม ดังนั้น

```
area = 1 / 2 * base * height;
```

เมื่อคอมพิวเตอร์ทำงานตามคำสั่งบรรทัดนี้ จะให้ผลลัพธ์ของพื้นที่มีเท่ากับ 0 ทั้งนี้เพราะ 1 / 2 ซึ่งเป็นเลขจำนวนเต็มหารกันให้ผลลัพธ์ 0.5 แต่ถูกปัดเศษกลายเป็น 0 ดังนั้นจึงทำให้ผลลัพธ์ของนิพจน์ทั้งหมดมีค่าเป็น 0

การแก้ปัญหานี้สามารถทำได้ 2 วิธี คือใช้วิธีกำหนด เป็นเลขจำนวนจริงหารกัน และ อีกวิธีหนึ่งคือใช้หลักการบังคับประเภทข้อมูล (type cast) ให้เป็นข้อมูลเลขทศนิยม ดังตัวอย่าง

วิธีที่ 1 `area = 1.0 / 2.0 * base * height;`

วิธีที่ 2 `area = (float) 1 / 2 * base * height;`

การใช้เครื่องหมาย ==

ภาษาซีมีรูปแบบให้ใช้เครื่องหมาย == เพื่อเป็นการเปรียบเทียบค่านิพจน์ทางซ้ายและทางขวาว่าเท่ากันหรือไม่ แทนที่จะใช้เครื่องหมาย = เพียงตัวเดียวเหมือนในสมการทางคณิตศาสตร์ ผลลัพธ์ของการเปรียบเทียบนิพจน์ด้วยเครื่องหมาย == จะมี 2 กรณีคือจริง หรือ เท็จ ซึ่งค่าจริงคือตัวเลขที่ไม่ใช่ 0 และค่าเท็จคือตัวเลขที่เป็น 0 เท่านั้น

ปกติ คำสั่ง if, while, do-while เป็นคำสั่งที่ทำงานโดยขึ้นอยู่กับค่าความจริงของเงื่อนไข ดังนั้นการใช้เครื่องหมายในเงื่อนไขผิด อาจทำให้เกิดข้อผิดพลาดได้ เช่น

```
x = 3;
if(x = 5)
    printf("It is five\n");
else
    printf("It is not five\n");
```

โปรแกรมดังกล่าวจะไม่มีข้อผิดพลาดทางไวยากรณ์ แต่เมื่อคอมพิวเตอร์ทำงานตามโปรแกรม จะพิมพ์ข้อความว่า It is five ซึ่งจริงๆแล้วโปรแกรมควรจะพิมพ์ข้อความว่า It is not five เพราะค่าของ 3 ไม่เท่ากับ 5 การใส่เครื่องหมายเปรียบเทียบผิดดังกล่าวทำให้เงื่อนไขของ if ยังคงเป็นจริง (ผลลัพธ์ในวงเล็บของ if จะมีค่าเป็น 5 ซึ่งไม่ใช่ 0) การแก้ไขทำได้โดย

```
x = 3;
```

```
if(x == 5)
    printf("It is five\n");
else
    printf("It is not five\n");
```

การทำให้โปรแกรมหลุดจากการวนซ้ำ

เมื่อภาษาซีทำงานคำสั่ง while ซึ่งเป็นคำสั่งเกี่ยวกับการวนซ้ำ จะมีตรวจสอบเงื่อนไขว่าเป็นจริงหรือไม่ ถ้าเงื่อนไขเป็นจริง ชุดคำสั่งที่ถูกคร่อมอยู่ในวงเล็บปีกกา จะกระทำทีละคำสั่ง จนถึงคำสั่งสุดท้าย แล้วจึงกลับไปตรวจสอบเงื่อนไขใหม่ว่าเป็นจริงหรือเท็จ ถ้าเงื่อนไขยังคงเป็นจริง จะกระทำชุดคำสั่งซ้ำอีก แล้วกลับไปตรวจสอบจนกว่าเงื่อนไขจะเป็นเท็จ การทำงานจึงจะหลุดจากการวนซ้ำ การมีคำสั่งที่จะทำให้หลุดจากการวนซ้ำได้จึงเป็นสิ่งสำคัญอย่างยิ่ง ตัวอย่างเช่น

```
x = 1;
while(x < 10) {
    printf("x = %d\n", x);
    x = x - 1;
}
```

ส่วนของโปรแกรมนี้จะทำให้คอมพิวเตอร์พิมพ์ค่าของ x ออกมาไม่สิ้นสุด เพราะค่าของ x สวนทางที่จะทำให้เงื่อนไขของคำสั่ง while กลายเป็นเท็จได้ ดังนั้นต้องแก้ใหม่เป็น

```
x = 1;
while(x < 10) {
    printf("x = %d\n", x);
    x = x + 1;
}
```

ไม่ใช้เลขจำนวนจริงในการเปรียบเทียบความเท่ากัน

ทั้งนี้เพราะเลขจำนวนจริง เป็นเลขที่มีความไม่ถูกต้อง 100% เพราะมีส่วนที่ทศนิยมอยู่ ดังนั้นการเขียนโปรแกรมดังต่อไปนี้จึงก่อให้เกิดการทำงานผิดพลาดได้

```
float i;

i = 1.0;
while( i != 10.0) {
   printf("Hello\n");
   i = i + 1.0;
}
```

โปรแกรมนี้จะพิมพ์ข้อความ Hello ออกมาไม่รู้จบ เพราะค่าของ i จะไม่มีทางเท่ากับค่า 10 ได้อย่างแท้จริง ดังนั้นต้องประกาศตัวแปร i ให้เก็บเลขจำนวนเต็มดังนี้

```
int i;

i = 1;
while( i != 10) {
   printf("Hello\n");
   i = i + 1;
}
```

การใช้คำสั่ง break หลังคำสั่งท้ายสุดของแต่ละ case ในคำสั่ง switch-case

การไม่ใส่คำสั่ง break หลังคำสั่งสุดท้ายของ แต่ละคำสั่ง case จะทำให้คอมพิวเตอร์ไปทำงานคำสั่งของ case ถัดไป หลังจากได้ทำชุดคำสั่งใน case ที่ตรงเงื่อนไขเสร็จแล้ว เช่น

```
x = 2;
switch (x) {
        case 1 : prin f("one");
        case 2 : prin f("two");
        case 3 : prin.f("three");
}
```

เมื่อโปรแกรมทำงานจะพิมพ์ข้อความทั้ง two และ three ดังนั้นให้แก้ไขเป็น

```
x = 2;
switch (x)  {
        case 1 : printf("one");
                 break;
        case 2 : printf("two");
                 break;
        case 3 : printf("three");
                 break;
}
```

2.2 ขอบเขตของการวิจัย

จะพัฒนาโปรแกรมตรวจสอบข้อผิดพลาดของภาษาซี โดยพัฒนาด้วยภาษา PHP ซึ่งมีหลักไวยากรณ์คล้ายคลึงภาษาซีมาก ซึ่งภาษา PHP จะทำให้สามารถเผยแพร่การใช้งานได้ง่าย โดยผู้ที่จะทำการตรวจสอบ สามารถใช้ web browser เพื่อเปิดหน้า (page) ของโปรแกรมตรวจสอบ และทำการ upload ชื่อแฟ้มข้อมูลต้นฉบับภาษาซี (ด้วยมาตรฐาน ANSI C) ของตนเอง มาที่หน้า (page) นั้น แล้วสามารถเลือกปุ่มเพื่อทำการตรวจสอบต่อไป

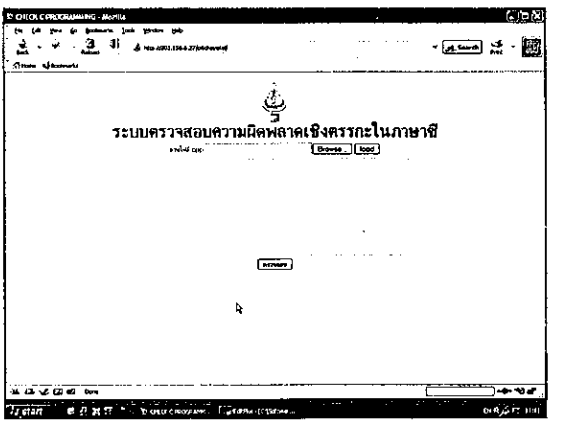
2.3 ข้อตกลงเบื้องต้น

1. โปรแกรมต้นฉบับ (source code) ควรผ่านการคอมไพล์ด้วยคอมไพเลอร์ภาษาซีมาแล้วโดยไม่มีข้อผิดพลาดด้านไวยากรณ์
2. ควรเป็นโปรแกรมสั้น ๆ ความยาวไม่เกิน 100 บรรทัด
3. อาจใช้ทดสอบคำสั่งโดยไม่ต้องพิมพ์ในฟังก์ชันเมนก็ได้
4. ใช้สำหรับทดสอบคำสั่งที่ใช้บ่อย ๆ เช่น scanf, printf, for, while, do-while, switch-case เป็นต้น
5. ใช้ทดสอบกับตัวแปรพื้นฐาน เช่น int, float, char เท่านั้น
6. ไม่ควรพิมพ์คำสั่งมากกว่า 1 คำสั่งในบรรทัดเดียว

บทที่ 3
การทำงานของโปรแกรมเพื่อค้นหาข้อผิดพลาดเชิงตรรกะของภาษาซี

3.1 การทำงานของโปรแกรม

เมื่อผู้ใช้ๆ โปรแกรมเว็บบราว์เซอร์ (Internet Explorer หรือ Mozilla) ต่อเข้ามาที่ URL ที่ติดตั้งโปรแกรมเพื่อค้นหาข้อผิดพลาดเชิงตรรกะของภาษาซี จะปรากฏหน้าต่างแรก ดังรูปที่ 3.1



รูปที่ 3.1 แสดงหน้าต่างแรกโปรแกรมตรวจสอบข้อผิดพลาดเชิงตรรกะของภาษาซี

อธิบายหน้าต่างแรกได้ดังนี้

1.แสดงโลโก้ของมหาวิทยาลัยเทคโนโลยีสุรนารี

2.แสดงชื่อของระบบตรวจสอบ

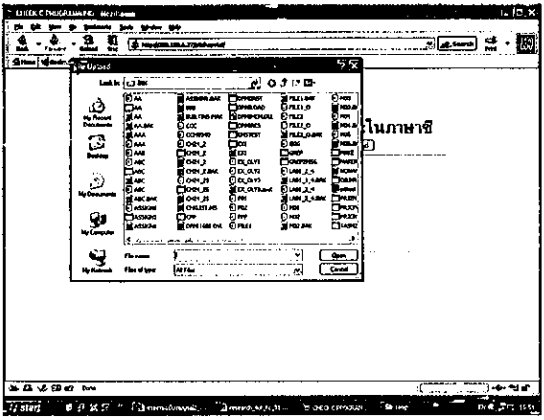
3.อ่านไฟล์ cpp เป็นการให้ใส่ชื่อแฟ้มภาษาซีที่ต้องการตรวจสอบ

4.ปุ่ม browse เป็นปุ่มที่ช่วยให้สามารถดึงแฟ้มภาษาซีที่ต้องการจากตัวขับและเส้นทางที่ผู้ใช้สามารถกำหนดได้

5.ปุ่ม load เป็นการดึง source code ภาษาซีจากที่ระบุในข้อ 3 หรือ 4 มาไว้ในพื้นที่ช่องใหญ่

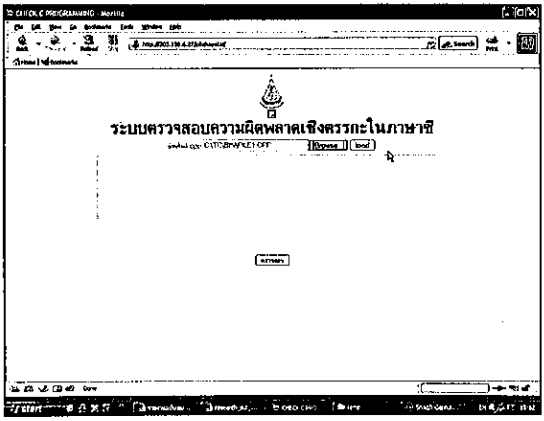
6.พื้นที่ช่องใหญ่เป็นรายการของ source code ที่ได้จากการ load

7.ปุ่มตรวจสอบ เป็นปุ่มที่ใช้ตรวจสอบความถูกต้องของ source code



รูปที่ 3.2 แสดงผลการทำงานเมื่อกดปุ่ม browse

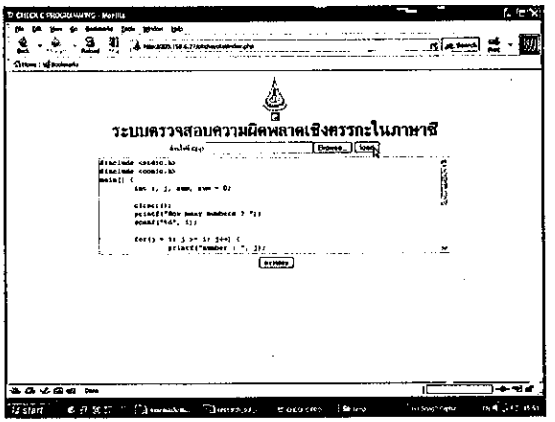
เมื่อกดปุ่ม browse ผู้ใช้จะสามารถระบุเส้นทางที่ต้องการค้นหาแฟ้มต้นฉบับภาษาซีได้ดัง รูปที่ 3.2 และสามารถเลือกแฟ้มที่ต้องการ ดังตัวอย่างในรูปที่ 3.3



รูปที่ 3.3 แสดงการเลือกแฟ้มที่จะทำการตรวจสอบ

3.2 การดึง source code มาเพื่อตรวจสอบ

เมื่อระบุแฟ้มต้นฉบับภาษาซีที่ต้องการแล้ว ให้กดปุ่ม load เพื่อดึงแฟ้มต้นฉบับมาไว้ในช่องพื้นที่เพื่อรอการตรวจสอบต่อไป ดังแสดงในรูปที่ 3.4



รูปที่ 3.4 แสดงการ load ข้อมูลจากแฟ้มที่ต้องการ

พื้นที่ในช่องพื้นที่เป็นพื้นที่ๆผู้ใช้งานสามารถทำการแก้ไข source code ได้ (ควรทำหลังจากทราบผลการตรวจสอบแล้ว) และสามารถเก็บผลการแก้ไขไว้เป็นแฟ้มซึ่งจะได้กล่าวถึงต่อไป

3.3 แฟ้มข้อมูลต้นฉบับภาษาซีที่ใช้ตรวจสอบ

ในที่นี้ได้สร้างแฟ้มภาษาซีชื่อ file1.cppขึ้น โดยทำการแก้ไขโปรแกรมบางส่วน และนำไปแปลเป็นภาษาเครื่อง (compile) ซึ่งจะไม่ปรากฏข้อผิดพลาดขึ้นเลย แต่เมื่อรันผลการทำงานของโปรแกรมที่ได้ทดลองจะพบการทำงานที่ผิดพลาดไปจากความต้องการ แฟ้มที่ใช้ทดลองแสดงในรูปที่ 3.5

```
#include <stdio.h>
#include <conio.h>
main() {
        int i, j, num, sum = 0;

        clrscr();
        printf("How many numbers ? ");
        scanf("%d", i);

        for(j = 1; j >= i; j++) {
                printf("number : ", j);
                scanf("%d", &num);
                if(num % 2 == 0);
                sum += num;
        }
        printf("sum of even numbers = %d\n", sum);
        getch();
        return 0;
}
```

รูปที่ 3.5 แสดงแฟ้มต้นฉบับที่ต้องการตรวจสอบ

แฟ้มต้นฉบับภาษาซีนี้ จะถูกใช้เพื่อทดสอบกรณีต่างๆ ดังนี้

1. การละเว้นเครื่องหมาย & หน้าตัวแปรสำหรับคำสั่งการอ่านข้อมูล scanf
2. การทดสอบการวนรอบที่ไม่ทำให้การวนรอบทำงาน
3. การใส่เครื่องหมายปิดคำสั่ง (;) หลังคำสั่ง if
4. การพิมพ์ค่าตัวแปรและรูปแบบที่ไม่สัมพันธ์กัน

3.4 ผลการตรวจสอบโดยใช้โปรแกรมค้นหาข้อผิดพลาดเชิงตรรกะของภาษาซี

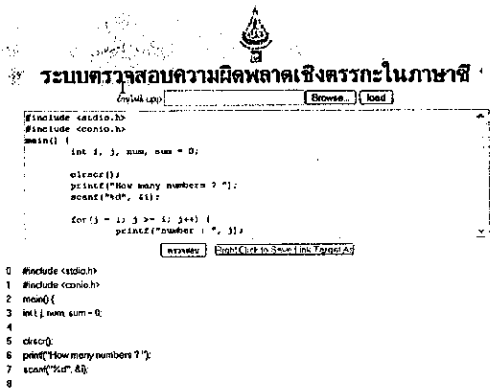
เมื่อทำการ load แฟ้มข้อมูลภาษาซี แล้ว ให้กดปุ่ม ตรวจสอบ เพื่อทำการตรวจสอบข้อผิดพลาดที่ไม่สามารถตรวจได้โดย compiler

จากการใช้แฟ้มทดสอบ file1.cpp จะได้ผลการตรวจสอบดังรูปที่ 3.6

```
0  #include <stdio.h>
1  #include <conio.h>
2  main() {
3  int i, j, num, sum = 0;
4
5  clrscr();
6  printf("How many numbers ? ");
7  scanf("%d", i); ..& ต้องอยู่หน้าตัวแปร i
8
9  for(j = 1; j >= i; j++) {
   //NO LOOP ตรวจสอบเงื่อนไข
10 printf("number : ", j);
11 scanf("%f", &num); "%ไม่สัมพันธ์กับ &num"
12 if(num % 2 == 0);
   // บรรทัด 12 คำสั่ง if(num % 2 == 0); มีเครื่องหมาย ; อยู่ท้ายวงเล็บ กรุณาตรวจสอบอีกครั้ง
13 sum += num;
14 }
15
16 printf("sum of even numbers = %f\n", sum); "%ไม่สัมพันธ์กับ sum"
17 getch();
18 return 0;
19 }
```

รูปที่ 3.6 แสดงผลการตรวจสอบโดยใช้โปรแกรมค้นหาข้อผิดพลาดเชิงตรรกะของภาษาซี

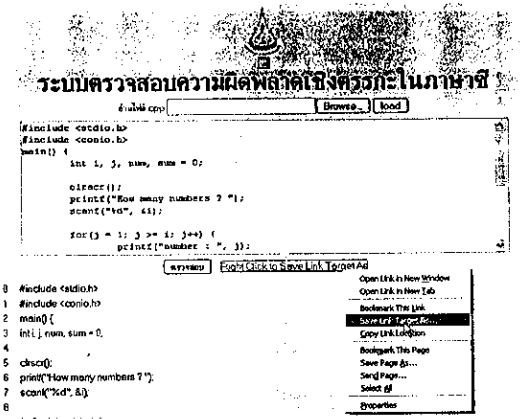
ในที่นี้จะทำการตรวจสอบทีละบรรทัด ถ้าบรรทัดใดมีข้อผิดพลาด จะแสดงสาเหตุที่เกิดข้อผิดพลาดขึ้นด้วยตัวอักษรสีแดง ผู้ใช้โปรแกรมสามารถแก้ไข source code ใน ช่องพื้นที่ตามข้อแนะนำ และสามารถกดปุ่มตรวจสอบได้อีก จนกระทั่งไม่พบว่ามีบรรทัดใดเกิดข้อผิดพลาด ดังแสดงในรูปที่ 3.7



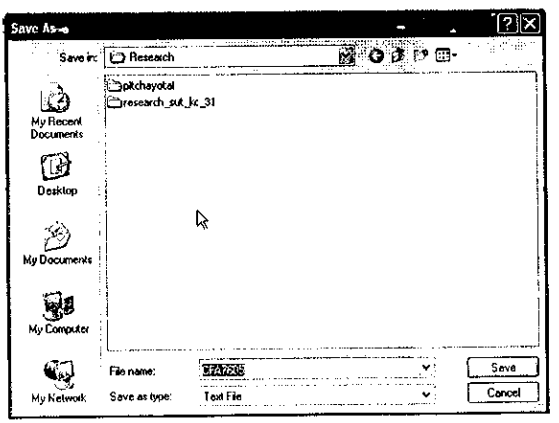
รูปที่ 3.7 แสดง source code ที่ผ่านการตรวจสอบโดยสมบูรณ์

3.5 การเก็บแฟ้มที่ได้แก้ไขแล้ว

เราสามารถเก็บแฟ้มต้นฉบับภาษาซีที่ได้ทำการแก้ไขเรียบร้อยแล้วเพื่อนำไปรันในภายหลัง โดยสามารถทำได้ดังนี้ ให้คลิกเม้าส์ปุ่มขวาตรงข้อความ Right Click to Save Link Target as ซึ่งจะปรากฏรายการหน้าต่างขึ้น ให้เลือกคำสั่ง Save Link Target As... ดังแสดงในรูปที่ 3.8 หลังจากนั้นจะมีชื่อตั้งให้โดยอัตโนมัติดังรูปที่ 3.9 ผู้ใช้งานสามารถเปลี่ยนชื่อได้ และสามารถเก็บบันทึกแฟ้มที่แก้ไขแล้วในตัวขับและเส้นทางตามต้องการ



รูปที่ 3.8 แสดงคำสั่งในการเก็บแฟ้มที่ได้ทำการแก้ไขแล้ว



รูปที่ 3.9 แสดงการเก็บแฟ้มตามที่ผู้ใช้สามารถกำหนดเองได้

บทที่ 4
บทสรุปและข้อเสนอแนะ

การพัฒนาโปรแกรมเพื่อค้นหาข้อผิดพลาดเชิงตรรกะของภาษาซี ได้ประสบความสำเร็จดังวัตถุประสงค์ ซึ่งผู้วิจัยจะสรุปความสามารถของโปรแกรมได้ ดังต่อไปนี้

1.1 ระบบโปรแกรมเพื่อการค้นหาข้อผิดพลาดเชิงตรรกะของภาษาซี

ระบบโปรแกรมเพื่อการค้นหาข้อผิดพลาดเชิงตรรกะของภาษาซี มีลักษณะและคุณสมบัติดังต่อไปนี้

- โปรแกรมสามารถเรียกใช้ได้จาก web page ที่ได้มีการลงโปรแกรมไว้ ดังนั้นผู้ใช้งานทั่วไปที่ต่อ internet และมี browser จึงสามารถเรียกใช้งานโปรแกรมได้
- สามารถทดสอบกับตัวแปรพื้นฐานในภาษาซีได้ เช่น ตัวแปรเลขจำนวนเต็ม ตัวแปรเลขทศนิยม ตัวแปรอักขระ เป็นต้น
- โปรแกรมสามารถตรวจจับการวางเครื่องหมายปิดคำสั่งผิดตำแหน่งได้
- ความสามารถในการตรวจสอบการทำงานกับคำสั่งวนรอบ เช่น จะสามารถทำคำสั่งวนรอบได้หรือไม่ ความสามารถในการออกจากการวนรอบได้ เป็นต้น
- ตรวจสอบการใช้ & กับตัวแปรที่ต้องมีการรับข้อมูลจากแป้นพิมพ์ด้วยคำสั่ง scanf
- การพิมพ์และการรับข้อมูลควรมีตัวแปรตรงประเภท และจำนวนตามรูปแบบที่กำหนด
- ตรวจสอบการใช้เครื่องหมาย = กับคำสั่งที่มีการเปรียบเทียบ
- ตรวจสอบ case ต่างๆในคำสั่ง switch ซึ่งควรมีการ break ทุกกรณี
- สามารถเก็บผลลัพธ์ที่ได้แก้ไขจากการตรวจสอบเพื่อนำไปรันในภายหลัง

1.2 การทดสอบโปรแกรมเพื่อค้นหาข้อผิดพลาดเชิงตรรกะของภาษาซี

ผู้วิจัยได้ทดสอบการใช้งานโปรแกรมนี้กับนักศึกษาที่เรียนวิชาการโปรแกรมคอมพิวเตอร์ด้วยภาษาซี โ ยได้สร้างแฟ้มต้นฉบับภาษาซีดังกล่าว และให้นักศึกษาได้ทำการแก้ไขโดยไม่ต้องใช้โปรแกรมช่ ย เทียบกับการใช้โปรแกรมช่วย พบว่านักศึกษาสามารถทำการแก้ไขได้เร็วขึ้นเมื่อใช้โปรแกรม และตระหนักถึงข้อผิดพลาดที่ไม่สามารถพบได้ด้วย compiler ของภาษาซี การทำงานของโปรแกรมดังกล่าวเป็นที่น่าพอใจ

1.3 ข้อเสนอแนะ

โปรแกรมการค้นหาข้อผิดพลาดเชิงตรรกะของภาษาซี สามารถทำการตรวจสอบได้ตามการเขียนโปรแกรมพื้นฐานแล้ว แต่ยังไม่คลอบคลุมทุกคำสั่งในภาษาซี ดังนั้นการพัฒนาต่อไปจึงควรมุ่งประเด็นพัฒนาดังนี้

- [ ] ความสามารถในการตรวจสอบกับฟังก์ชันที่ผู้เขียนโปรแกรมเขียนขึ้นเอง
- [ ] ความสามารถในการตรวจสอบกับชนิดข้อมูลที่ซับซ้อนขึ้นเช่น อาร์เรย์ พอยน์เตอร์ และข้อมูลแบบโครงสร้าง
- [ ] การทำงานกับขบวนการเรียกตนเอง (recursion)

บรรณานุกรม

Mahatthanapiwat, P., (2003) Detectable Logic Error in C Language, SUT Journal, Vol. 10, No. 3

Deitel, H., Deitel, P. (1994). C:How to program. Prentice Hall International, Inc.

Sommerville, Ian. (2001). Software Engineering (6th ed.). Addison Wesley.

Kruse R., Tondo C., Leung, B. (1997) Data Structure & Program Design in C (2nd ed.) Prentice Hall International Inc.

ประวัติผู้วิจัย

ชื่อ นายพิชโยทัย มหัทธนาภิวัฒน์ (Mr.Pichayotai Mahatthanapiwat)

วันเดือนปีเกิด 10 พฤศจิกายน 2507

ตำแหน่งปัจจุบัน ผู้ช่วยศาสตราจารย์ประจำสาขาวิชาวิศวกรรมคอมพิวเตอร์

หน่วยงานที่ติดต่อได้พร้อมทั้งโทรศัพท์และโทรสาร

สาขาวิชาวิศวกรรมคอมพิวเตอร์ สำนักวิชาวิศวกรรมศาสตร์

มหาวิทยาลัยเทคโนโลยีสุรนารี

111 ถนนมหาวิทยาลัย ตำบลสุรนารี อำเภอเมือง

จังหวัดนครราชสีมา 30000

โทรศัพท์ 044-224461, 044-224422

โทรสาร 044-224220

ประวัติการศึกษา

Ph. D (Computer Engineering) Chulalongkorn University Bangkok 2002.

M.Sc (Computer Science) Chulalongkorn University Bangkok 1991.

B. Eng (Mechanical Engineering) KMIT (Thonburi) Bangkok 1987.

ผลงานทางวิชาการ

[1] Mahatthanapiwat, P., and Rivepiboon, W. Direct Access to Terminal Virtual Path in OODB. Proceedings of National Computer Science and Engineering, 1999.

[2] Mahatthanapiwat, P., and Rivepiboon, W., Virtual Path Signature: An Approach for Flexible Searching in OODB. Proceedings of International Conference on Intelligent Technology, pp. 335-340, 2000.

[3] Mahatthanapiwat, P., and Rivepiboon, W. Bra ich Index: An Approach for Query Processing in OODB. International Journal of Inf ormation Technology, Vol. 7. No. 2, 2001.

[4] Mahatthanapiwat, P., Access Method of Aggregation Hierarchy as a Tree in OODB, Ph. D Thesis, Chulalongkorn University, 2002.

[5] Mahatthanapiwat, P., Detectable Logic Error in C Language, SUT Journal, Vol. 10, No. 3, 2003.

[6] Mahatthanapiwat, P., Join Signature, Proceeding of the 7th National Computer Science and Engineering Conference, 2003.

[7] Mahatthanapiwat, P, and Rivepiboon, W. , Virtual Path Signature: An Approach for Flexible Searching in OODB., International Journal of Intelligent Systems, Vol. 19, No.1/2, 2004.