การลดขนาดพื้นที่การค้นหาเส้นทางที่ดีที่สุดจากเว็บเซอร์วิสกราฟ

SEARCH SPACE REDUCTION OF OPTIMAL PATH DISCOVERY FROM WEB SERVICE GRAPH

ณัทภต นันตาลิต
NUTTAPOD NUNTALID

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชา วิทยาการคอมพิวเตอร์
บัณฑิตวิทยาลัย
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
พ.ศ. 2551

KMITL-2008-SC-M-002-056

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

การลดขนาดพื้นที่การค้นหาเส้นทางที่ดีที่สุดจากเว็บเซอร์วิสกราฟ

SEARCH SPACE REDUCTION OF OPTIMAL PATH DISCOVERY FROM WEB SERVICE GRAPH

ณัทภต นันตาลิต

NUTTAPOD NUNTALID

TN.
S26 3827
2551

เลขหมู่ 87113
เลขทะเบียน
วัน,เดือน,ปี 30 ส.ค. 2552

b 12038969
i

เอกสารฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรมหาบัณฑิต

สาขาวิชา วิทยาการคอมพิวเตอร์

บัณฑิตวิทยาลัย

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ. 2551

ISBN KMITL-2008-SC-M-002-056

SEARCH SPACE REDUCTION OF OPTIMAL PATH DISCOVERY FROM WEB SERVICE GRAPH

NUTTAPOD NUNTALID

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

MASTER OF SCIENCE IN COMPUTER SCIENCE

SCHOOL OF GRADUATE STUDIES

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2008

ISBN KMITL-2008-SC-M-002-056

COPYRIGHT 2008

SCHOOL OF GRADUATE STUDIES

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

หัวข้อวิทยานิพนธ์	การลดขนาดพื้นที่การค้นหาเส้นทางที่ดีที่สุดจากเว็บเซอร์วิสกราฟ
นักศึกษา	นายณัทภต นันตาลิต
รหัสประจำตัว	49067552
ปริญญา	วิทยาศาตรมหาบัณฑิต
สาขา	วิทยาการคอมพิวเตอร์
พ.ศ.	2551
อาจารย์ที่ปรึกษา	ผศ.ดร.ศรัณย์ อินทโกสุม

บทคัดย่อ

งานวิจัยนี้เสนอแนวคิดการลดพื้นที่การค้นหาเส้นทางของขั้นตอนวิธีค้นหาเส้นทางในเว็บเซอร์วิสกราฟ โดยขั้นตอนวิธีการค้นหาแบบ A* ถูกนำมาใช้แทนการค้นหาแบบแนวกว้างก่อนในการค้นหาเส้นทางที่ดีที่สุด ผลการวิจัยพบว่าถ้าใช้ขั้นตอนวิธีการค้นหาแบบ A* สามารถลดพื้นที่ในการค้นหาเส้นทางในเว็บเซอร์วิสกราฟลงได้ 1 ระดับเมื่อใช้ฟังก์ชันฮิวริสติกแบบที่ 1 โดยถ้าเว็บเซอร์วิสที่เป็นคำตอบอยู่ที่ระดับชั้น n การค้นหาในระดับชั้นที่ n – 1 จะค้นเพียงเว็บเซอร์วิสเดียว แล้วสามารถนำไปสู่เว็บเซอร์วิสที่เป็นคำตอบได้ทันที ในขณะที่การค้นหาแบบแนวกว้างก่อนจะต้องค้นทุกเว็บเซอร์วิสในระดับที่ n – 1 และถ้าใช้ขั้นตอนวิธีการค้นหาแบบ A* โดยใช้ฟังก์ชันฮิวริสติกแบบที่ 2 สามารถลดพื้นที่ในการค้นหาลงได้เพิ่มขึ้น

Thesis	Search Space Reduction of Optimal Path Discovery from Web Service Graph
Student	Mr. Nuttapod Nuntalid
Student ID	49067552
Degree	Master of Science
Program	Computer Science
Year	2551
Thesis Advisor	Assist Professor Dr.Sarun Intakosum

ABSTRACT

This research proposes an idea to reduce search space in path discovery algorithm for web service graph. The part of the algorithms namely, path discovery, is the topic of the improvement. The A* search algorithm is used to replace the breadth-first search in order to find the optimal path. The result of this research indicates that the A* search algorithm, which is used with heuristic type 1,can reduce one search level in that if the target service is at level n of the web service graph, only one web service at level n – 1 is needed to be visited before the target web service is reached. On the other hand, all web services at level n – 1 are needed to be visited using breadth-first search approach. In addition, the A* search algorithm which is used with heuristic type 2 can reduce space more then type 1.

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จได้อย่างดีด้วยคำแนะนำจาก ผศ.ดร.ศรัณย์ อินทโกสุม ซึ่งเป็นอาจารย์ผู้ควบคุมวิทยานิพนธ์ ข้าพเจ้ารู้สึกทราบซึ้งในความอนุเคราะห์ และขอขอบพระคุณเป็นอย่างสูง

ขอกราบพระคุณคณาจารย์สาขาวิทยาการคอมพิวเตอร์ และเจ้าหน้าที่ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ทุก ๆ ท่านที่ได้ประสิทธิ์ประสาทวิชา และอำนวยความสะดวกให้กับข้าพเจ้า

ขอขอบคุณเพื่อนๆ พี่ๆ น้องๆ ในสาขาสาขาวิทยาการคอมพิวเตอร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ทุกคนที่ให้คำแนะนำต่างๆ และคอยให้กำลังใจเสมอมา

ขอขอบคุณบัณฑิตศึกษาและบัณฑิตวิทยาลัย คณะวิทยาศาสตร์ที่ให้ความช่วยเหลือในเรื่องต่างๆ

สุดท้ายนี้ข้าพเจ้าขอกราบขอบพระคุณ บิดา มารดา และครอบครัวของข้าพเจ้าที่เป็นกำลังใจ และให้การสนับสนุนในทุกเรื่องๆ ทำให้ข้าพเจ้าสามารถทำวิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงด้วยดี

คุณค่าและประโยชน์อันพึงมาจากวิทยานิพนธ์ฉบับนี้ ข้าพเจ้าขอมอบแด่ผู้มีพระคุณทุกท่าน

ณัทภต นันตาลิต

สารบัญ

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญตาราง

สารบัญรูป

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของปัญหา

เว็บเซอร์วิสเป็นเทคโนโลยีที่ทำให้แอปพลิเคชันต่างๆ แลกเปลี่ยนข้อมูลกันได้ แม้ว่าจะสร้างบนสถาปัตยกรรม ภาษา หรือฐานข้อมูลที่แตกต่างกัน และเพื่อให้การสืบค้นเป็นไปอย่างมีประสิทธิภาพจึงต้องมีระบบทะเบียนและสืบค้นเว็บเซอร์วิสที่เรียกว่ายูดีดีไอ (UDDI) แต่เมื่อเว็บเซอร์วิสมีจำนวนเพิ่มมากขึ้น ทำให้ประสิทธิภาพการค้นหาเว็บเซอร์วิสในยูดีดีไอลดลง จึงเป็นงานวิจัยที่ได้รับความสนใจเป็นอย่างมากตั้งแต่อดีตจนมาถึงปัจจุบัน ซึ่งงานวิจัยส่วนใหญ่จะมุ่งเน้นในด้านการปรับปรุงขั้นตอนวิธีต่างๆที่จะนำมาใช้ให้สามารถเพิ่มประสิทธิภาพในการค้นหาและจัดกลุ่มเว็บเซอร์วิสให้ดีขึ้น เช่นในงานวิจัย [7] [8] และ [9] ได้นำเสนอการใช้ขั้นตอนวิธีแบบจับคู่ความคล้ายคลึงทางด้านความหมายของเว็บเซอร์วิส และ จับคู่ความคล้ายคลึงกันทางโครงสร้างข้อมูลของพารามิเตอร์ของเว็บเซอร์วิส เพื่อให้สามารถค้นหาเว็บเซอร์วิสได้รวดเร็วขึ้น

เมื่อยูดีดีไอได้พัฒนาถึงเวอร์ชัน 2 จึงมีงานวิจัยกลุ่มหนึ่งเกิดขึ้น ซึ่งนำเสนอวิธีการที่แตกต่างออกไป กล่าวคือพยายามที่จัดเว็บเซอร์วิสที่มีอยู่ในยูดีดีไอให้อยู่ในรูปแบบของกราฟ ซึ่งเรียกว่าเว็บเซอร์วิสกราฟ เพื่อจะได้นำขั้นตอนวิธีที่ใช้กับทฤษฎีกราฟมาใช้ได้ โดยแนวคิดพื้นฐานของเว็บเซอร์วิสกราฟ เริ่มมาจากยุคที่เว็บมีความนิยมแรกๆ ได้มีงานวิจัยกลุ่มหนึ่งที่พยามทำการจัดกลุ่มเว็บให้อยู่ในรูปของกราฟ เรียกว่าเว็บกราฟ ดังเช่นงานวิจัย [1] และ [2] ซึ่งในแต่ละโหนด (node) ที่อยู่ในเว็บกราฟคือเว็บเพจ และเส้นเชื่อม (edge) คือไฮเปอร์ลิงค์ (hyperlink) เมื่อมีเว็บเซอร์วิสเกิดขึ้น งานวิจัย [3] และ [4] จึงได้นำเสนอโครงสร้างข้อมูลแบบเว็บเซอร์วิสกราฟขึ้น โดยแต่ละโหนดในเว็บเซอร์วิสกราฟ คือเว็บเซอร์วิส และเส้นเชื่อมในเว็บเซอร์วิสกราฟจะแสดงถึงความสัมพันธ์ระหว่างเว็บเซอร์วิสหนึ่งไปยังอีกเว็บเซอร์วิสหนึ่ง

แต่ปัญหาอย่างหนึ่งของเว็บเซอร์วิสกราฟคือ มีความซับซ้อนทางด้านพื้นที่ (space complexity) ในการค้นหาเส้นทางสูง ดังนั้นในงานวิจัยฉบับนี้จึงได้นำเสนอวิธีลดความซับซ้อนทางด้านพื้นที่ในการค้นหาลง โดยนำขั้นตอนวิธีการค้นหาแบบ A* มาใช้ในขั้นตอนการค้นหาเส้นทางในเว็บเซอร์วิสกราฟ

1.2 ความมุ่งหมายและวัตถุประสงค์ของการศึกษา

งานวิจัยนี้มีจุดประสงค์เพื่อนำเสนอวิธีการที่จะลดขอบเขตหรือพื้นที่ในการค้นหาเส้นทางที่ดีที่สุดในเว็บเซอร์วิสกราฟ

1.3 สมมุติฐานของการศึกษา

การปรับปรุงโครงสร้างข้อมูลแบบเว็บเซอร์วิสกราฟ โดยนำขั้นตอนวิธีการค้นหาแบบ A* มาใช้ในขั้นตอนการค้นหาเส้นทางในเว็บเซอร์วิสกราฟ จะสามารถลดพื้นที่ในการค้นหาเส้นทางที่ดีที่สุดในเว็บเซอร์วิสกราฟได้ และเมื่อปรับปรุงฟังก์ชันฮิวริสติกโดยคำนวณจากการพิจารณาโครงสร้างของการเก็บเว็บเซอร์วิสกราฟและใช้ความสัมพันธ์ของพารามิเตอร์มาคำนวณร่วมกัน สามารถลดพื้นที่ในการค้นหาได้มากกว่าการใช้ฟังก์ชันฮิวริสติกโดยคำนวณจากการพิจารณาโครงสร้างของการเก็บเว็บเซอร์วิสกราฟเพียงอย่างเดียว

1.4 ทฤษฎีหรือแนวคิดในการวิจัย

โครงสร้างข้อมูลแบบเว็บเซอร์วิสกราฟเป็นกราฟแบบไม่มีวงระบุทิศทาง (directed acyclic graph) โดยในขั้นตอนการค้นหาเส้นทางในเว็บเซอร์วิสกราฟนั้น ใช้ขั้นตอนวิธีการค้นหาแบบแนวกว้างก่อน ซึ่งได้คำตอบเป็นเส้นทางที่ดีที่สุด แต่ทำการค้นหาทีละระดับชั้นจนเกือบทั่วทั้งเว็บเซอร์วิสกราฟ ผู้วิจัยจึงมีความเห็นว่าควรนำขั้นตอนวิธีการค้นหาแบบ A* มาใช้แทนขั้นตอนวิธีการค้นหาแบบแนวกว้างก่อน เนื่องจากสามารถได้คำตอบที่ดีที่สุดเช่นกัน แต่ขั้นตอนวิธีการค้นหาแบบ A* มีฟังก์ชันฮิวริสติกที่ใช้สำหรับคำนวณต้นทุน ซึ่งจะส่งผลให้พื้นที่ในการค้นหาเส้นทางในเว็บเซอร์วิสกราฟลดลง

ในงานวิจัยนี้จึงนำเสนอการปรับปรุงโครงสร้างข้อมูลแบบเว็บเซอร์วิสกราฟ โดยจะทำการปรับปรุงขั้นตอนการค้นหาเส้นทางในเว็บเซอร์วิสกราฟโดยใช้ขั้นตอนวิธีการค้นหาแบบ A* ซึ่งจะนำเสนอฟังก์ชันฮิวริสติก 2 แบบ โดยแบบที่ 1 จะคำนวณจากการพิจารณาโครงสร้างของเว็บเซอร์วิสกราฟเป็นหลัก ส่วนแบบที่ 2 จะพิจารณาจากโครงสร้างของเว็บเซอร์วิสกราฟและพิจารณาจากความสัมพันธ์ของพารามิเตอร์ส่งออกของเว็บเซอร์วิสตำแหน่งที่กำลังสำรวจเทียบกับพารามิเตอร์ที่ต้องการทั้งหมดของการร้องขอ

ในส่วนที่ 2 จะนำเสนอ ถึงวิธีการทดลองและผลการทดลอง เพื่อที่จะเปรียบเทียบระหว่างโครงสร้างข้อมูลแบบเว็บเซอร์วิสกราฟที่ผู้วิจัยทำการปรับปรุงโดยเปรียบเทียบระหว่างขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชันฮิวริสติกแบบที่ 1 และขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชันฮิวริสติกแบบที่ 2 โดยจะทำการเปรียบเทียบจำนวนเว็บเซอร์วิสที่เข้าไปสำรวจในเว็บ

เซอร์วิสกราฟในการค้นหาเส้นทาง ซึ่งจะแสดงถึงพื้นที่หรือขอบเขตในการหาเส้นทางในเว็บเซอร์วิสกราฟ

1.5 ขั้นตอนของการศึกษา

ด้านทฤษฎี จะทำการศึกษาโครงสร้างข้อมูลของเว็บเซอร์วิสกราฟ และศึกษาขั้นตอนวิธีการค้นหาแบบ A* เพื่อเป็นแนวทางในการปรับปรุงโครงสร้างข้อมูลแบบเว็บเซอร์วิสกราฟ

ด้านการประยุกต์ใช้งาน จะนำทฤษฎีที่ได้ศึกษาในข้างต้นมาปรับปรุงโครงสร้างข้อมูลแบบเว็บเซอร์วิสกราฟ ขั้นตอนการค้นหาเส้นทางในเว็บเซอร์วิสกราฟ แล้วทำการเปรียบเทียบอัตราการเติบโตของฟังก์ชัน ในด้านของความซับซ้อนทางด้านเวลา (time complexity) และความซับซ้อนทางด้านพื้นที่ (space complexity) ระหว่างการใช้ขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชันฮิวริสติกแบบที่ 1 และขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชันฮิวริสติกแบบที่ 2 และสุดท้ายจะทำการทดลองเพื่อเปรียบเทียบให้เห็นความแตกต่างในการใช้งานจริง ระหว่างพื้นที่ในการค้นหาเส้นทางในเว็บเซอร์วิสกราฟ

1.6 ส่วนประกอบของวิทยานิพนธ์

วิทยานิพนธ์ฉบับนี้ประกอบไปด้วยส่วนต่างๆ ดังนี้คือ บทที่ 2 จะกล่าวถึงทฤษฎีและงานวิจัยที่เกี่ยวข้อง บทที่ 3 กล่าวถึงวิธีในการลดพื้นที่การค้นหาสำหรับเว็บเซอร์วิสกราฟ ในบทที่ 4 กล่าวถึงการทดลองและผลการทดลอง และบทที่ 5 เป็นบทสรุป

บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

ในบทนี้จะอธิบายถึงทฤษฎีและงานวิจัยที่เกี่ยวข้องกับเรื่องเว็บเซอร์วิสกราฟ โดยแบ่งเนื้อหาออกเป็นส่วนๆดังนี้คือ ส่วนที่หนึ่งกล่าวถึงความรู้พื้นฐานของเว็บเซอร์วิสและยูดีดีไอ ส่วนที่สองจะกล่าวถึงงานวิจัยที่เกี่ยวข้อง ส่วนที่สามกล่าวถึงโครงสร้างข้อมูลแบบเว็บเซอร์วิสกราฟ และสุดท้ายกล่าวถึงขั้นตอนวิธีการค้นหาแบบ A*

2.1 เว็บเซอร์วิสและยูดีดีไอ

2.1.1 เว็บเซอร์วิส

ในอดีตที่ผ่านมาองค์กรต่าง ๆ มักจะประสบปัญหาจากการทำงานร่วมกันโดยใช้แอปพลิเคชันต่างๆที่ถูกพัฒนามาจากหลากหลายแพลตฟอร์มหลากหลายระบบปฏิบัติการ หลากหลายภาษา และถึงแม้ว่าองค์กรต่างๆสามารถที่จะเชื่อมต่อแอปพลิเคชันต่างๆ เข้าด้วยกันได้ แต่การทำเช่นนั้นทำให้มีค่าใช้จ่ายที่สูงและมีความสลับซับซ้อนมาก ด้วยเหตุนี้เองจึงมีความต้องการมาตรฐานกลางเพื่อทำให้การติดต่อสื่อสารและทำงานร่วมกันขององค์กรให้สะดวกและรวดเร็วมากขึ้นซึ่งมาตรฐานกลางนั้นก็คือเว็บเซอร์วิส

เว็บเซอร์วิสเป็นระบบซอฟต์แวร์ที่ออกแบบมาเพื่อสนับสนุนการทำงานระหว่างคอมพิวเตอร์กับคอมพิวเตอร์ผ่านระบบเครือข่าย โดยใช้ภาษาเอ็กซ์เอ็มแอล (XML) เป็นภาษาที่ใช้ในการอธิบาย ซึ่งแสดงได้ดังดังรูปที่ 2.1 ซึ่งในการเรียกใช้ซอฟต์แวร์และการติดต่อสื่อสารระหว่างโปรแกรมโดยผู้ให้บริการเว็บเซอร์วิสหนึ่งอาจจะเป็นผู้ขอบริการเว็บเซอร์วิสอื่น ตัวอย่างเช่น เว็บเซอร์วิสที่ให้บริการข้อมูลก่อนการซื้อขายหุ้นอาจจะเป็นผู้ขอใช้บริการของเว็บเซอร์วิสที่ให้บริการการให้ข่าวความสามารถที่ทำให้โปรแกรมเชื่อมต่อกับโปรแกรมอื่นได้นั้นเป็นจุดเด่นของเว็บเซอร์วิสที่เชื่อมบริการหลายๆบริการเข้าด้วยกันมาตรฐานหลายมาตรฐานได้ถูกนำเสนอขึ้นเพื่อสนับสนุนการทำให้เว็บเซอร์วิสติดต่อกันได้อย่างมีประสิทธิภาพเช่น การใช้เอกสารภาษา WSDL (Web Services Description Language) ซึ่งเป็นภาษาเอ็กซ์เอ็มแอลประเภทหนึ่งโดย WSDL เป็นภาษาที่ใช้ในการอธิบายการเรียกใช้งานเว็บเซอร์วิสซึ่งเปรียบเสมือนการอ่านคู่มือการใช้งานโปรแกรมนั่นเองแต่จะมีข้อแตกต่างกันตรงที่ไม่เฉพาะมนุษย์เท่านั้นที่สามารถเข้าใจ โปรแกรมที่สามารถอ่านเอกสารภาษาเอ็กซ์เอ็มแอลเข้าใจก็สามารถที่จะเข้าใจเอกสาร WSDL ได้เช่นกัน ซึ่งจากคุณสมบัตินี้ทำให้การเรียกใช้เว็บเซอร์วิสเป็นไปได้อย่างอัตโนมัติ นอกจากภาษาเอ็กซ์เอ็มแอลจะ

ถูกใช้เป็นภาษาในการอธิบายการเรียกใช้เว็บเซอร์วิสแล้วยังเป็นภาษาที่ใช้ในการบันทึกข้อมูลระหว่างผู้ให้บริการและผู้ขอใช้การบริการเว็บเซอร์วิส

```
< UNIVERSITY xmlns : xsi=<"http://locolhost/..... ">
    <Staff>
        <StaffID>10001</StaffID>
        <salary> 99999</salary>
        <Works_in>
            <Name> CS </Name>
            <head></head>
        </works_in>

        <SSN> 1234567891234 </SSN>
        <name> John Doe </name>
        <age> 55 </age>
        <sex> M </sex>
        <spouse></spouse>

        <nation>
            <name>Thailand </name>
            <area> 123456789</area>
            <population> 66000000 </population>
        </nation>
    </Staff>
< /UNIVERSITY>
```

รูปที่ 2.1 ภาษาเอ็กซ์เอ็มแอล

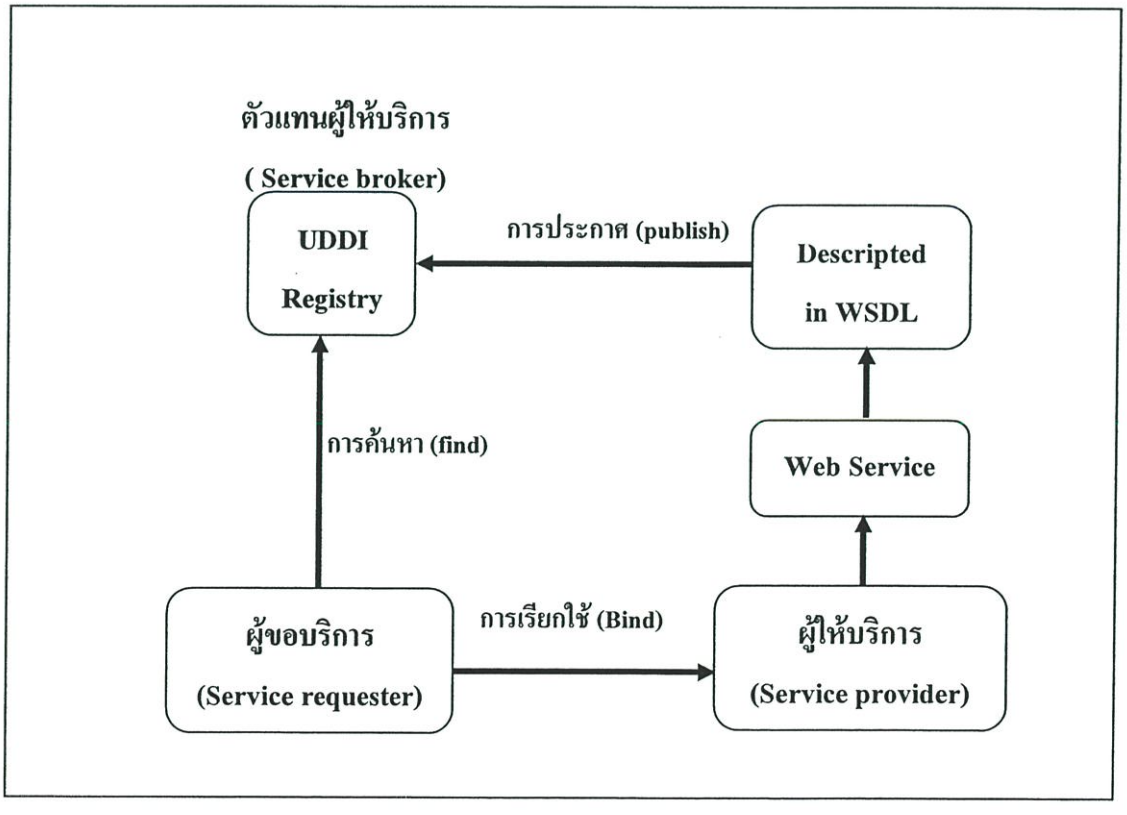
รูปแบบของข้อมูลเอ็กซ์เอ็มแอลที่ใช้ในการติดต่อนี้เรียกว่า SOAP (Simple Object Access Protocol) เนื่องจากข้อมูลที่ติดต่ออยู่ในรูปแบบภาษาเอ็กซ์เอ็มแอลทำให้โปรแกรมต่างๆ สามารถติดต่อกันได้ถึงแม้ว่าอาจจะถูกพัฒนาและเรียกใช้บนแพลตฟอร์มที่แตกต่างกันหรือใช้ภาษาแตกต่างกันในการพัฒนา ทั้งนี้เนื่องจากเอ็กซ์เอ็มแอล เป็นภาษาข้อความ (text) ซึ่งระบบปฏิบัติการทุกระบบสามารถที่จะเข้าใจได้ นอกจากนี้การที่เอ็กซ์เอ็มแอลมีแท็ก (tag) และรูปแบบโครงสร้างที่อธิบายข้อมูลด้วยตัวมันเองทำให้การเข้าใจ และการจัดการนั้นสามารถทำได้โดยโปรแกรมและช่วยทำให้การติดต่อระหว่างผู้ให้บริการและผู้ใช้เว็บเซอร์วิสเป็นไปได้อย่างอัตโนมัติ

2.1.2 ยูดีดีไอ

ยูดีดีไอเกิดจากการที่ผู้พัฒนาเว็บเซอร์วิสต้องการที่จะประกาศให้คนอื่นทราบและเรียกใช้งานเว็บเซอร์วิสของตนเองได้ จึงต้องประกาศหรือแจ้งให้ผู้อื่นทราบ ส่วนผู้ที่ต้องการใช้เว็บเซอร์วิสก็ย่อมต้องการที่จะทราบว่ามีเว็บเซอร์วิสใดบ้างที่เปิดให้บริการ ดังนั้นจึงต้องมีระบบจัดการเพื่อลงทะเบียนค้นหาและขอรายละเอียดเกี่ยวกับเว็บเซอร์วิสต่างๆระบบนี้เรียกว่ายูดีดีไอ (UDDI Registry) ซึ่งโดยสรุปแล้วยูดีดีไอจะทำหน้าที่ให้กับบุคคลสองกลุ่มคือ ผู้ที่ต้องการนำเว็บเซอร์วิสของตนมาเผยแพร่และเปิดให้บริการ (Publisher) ยูดีดีไอต้องอนุญาตให้ผู้ให้บริการ (Publisher หรือ

Service provider) นั้นทำการจัดการเกี่ยวกับเว็บเซอร์วิสที่ตนจะเผยแพร่ไม่ว่าจะเพิ่มเข้าบัญชีรายชื่อ แก้ไข หรือลบออก นอกจากนี้ ยูดีดีไอ จะต้องให้บริการในส่วนของผู้ที่จะมาติดต่อเพื่อค้นหาและขอข้อมูลเกี่ยวกับเว็บเซอร์วิส หรือเรียกว่าผู้ขอบริการ (Consumer หรือ Service requester) ยูดีดีไอต้องให้บริการในการสืบค้นข้อมูลและให้รายละเอียดของเว็บเซอร์วิสต่างๆ ที่ผู้ให้บริการได้นำมาเผยแพร่ไว้ จากนั้นก็จะเป็นหน้าที่ของผู้ขอบริการเองที่ตัดสินใจเลือกใช้เซอร์วิสและทำการติดต่อไปยังผู้ให้บริการตามข้อมูลที่ผู้ให้บริการได้เผยแพร่

ยูดีดีไอถูกนำเสนอโดยกลุ่ม http://www.uddi.org และกำหนดให้ยูดีดีไอเป็นมาตรฐานสำหรับการทำงานข้ามแพลตฟอร์มเมื่อมีผู้ที่ต้องการลงทะเบียนเว็บเซอร์วิสในครั้งแรกจะต้องลงทะเบียน รายละเอียดต่างๆเพื่ออธิบายรายละเอียดของเว็บเซอร์วิสนั้น ซึ่งลงทะเบียนโดยระบุ URL ของ WSDL และคำอธิบายรายละเอียดเว็บเซอร์วิส ดังนั้นเมื่อผู้ให้บริการต้องการเผยแพร่เว็บเซอร์วิส หรือผู้ขอบริการต้องการค้นหาว่ามีเว็บเซอร์วิสใดบ้างที่เปิดให้บริการ ซึ่งหากไม่มีบริการลงทะเบียนเว็บเซอร์วิสหรือบริการค้นหาเว็บเซอร์วิสเป็นไปได้ว่าการพัฒนาที่สมบูรณ์จะไม่เกิดขึ้น ในรูปที่ 2.2 แสดงถึงการทำงานของยูดีดีไอและเว็บเซอร์วิส



รูปที่ 2.2 การทำงานของยูดีดีไอและเว็บเซอร์วิส

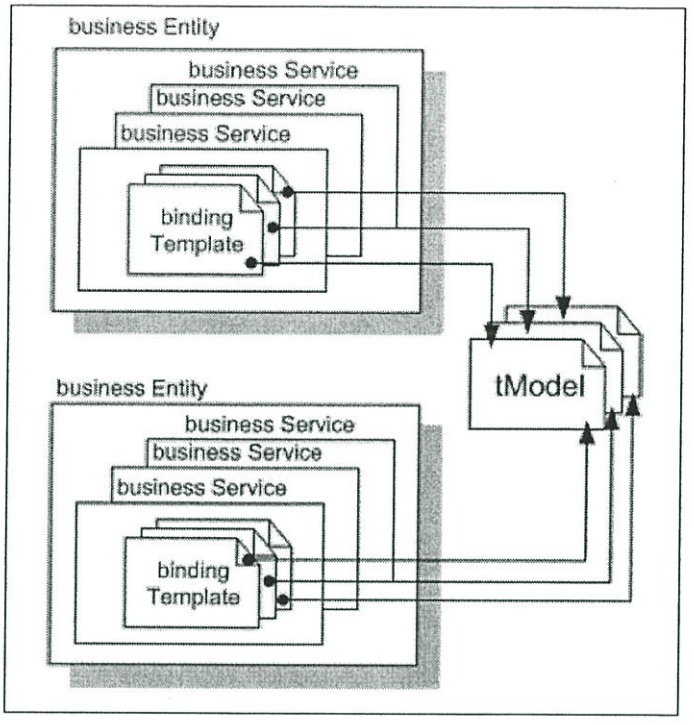
จากรูปที่ 2.2 มีตัวกลางให้บริการในการลงทะเบียน และค้นหาเว็บเซอร์วิสขึ้นหรือเรียกว่าตัวแทนผู้ให้บริการ (Service Broker) ซึ่งพัฒนาขึ้นได้จากการอ้างอิงมาตรฐานยูดีดีไอซึ่งเป็นมาตรฐานที่มีไว้ใช้สำหรับค้นหาเว็บเซอร์วิส เมื่อผู้ให้บริการต้องการให้บริการเว็บเซอร์วิส หรือต้องการประกาศให้ผู้อื่นทราบเกี่ยวกับเว็บเซอร์วิสของตนต้องทำการลงทะเบียนเว็บเซอร์วิสไว้ที่ยูดีดีไอโดยผ่านตัวแทนผู้บริการ (Service Broker) เพื่อให้ผู้ขอบริการที่ต้องการใช้บริการเว็บเซอร์วิสทำการค้นหาเว็บเซอร์วิสที่ต้องการ หลังจากผู้ขอบริการทราบว่าใครคือผู้ให้บริการเว็บเซอร์วิสนั้นขั้นตอนสุดท้ายคือผู้พัฒนาทั้งสองฝ่ายทำข้อตกลงเพื่อที่จะใช้เว็บเซอร์วิสนั้นเพื่อการแลกเปลี่ยนข้อมูลซึ่งกันและกัน

โครงสร้างข้อมูลของยูดีดีไอประกอบด้วยส่วนหลักอยู่ 2 ส่วน คือ

1) ส่วนของรายละเอียดข้อมูลของผู้ให้บริการหรือผู้ลงทะเบียนเว็บเซอร์วิสซึ่งจะอยู่ในรูปแบบของบิซิเนสเอนทิตี (Business Entity) ที่อธิบายถึงบริการต่างๆ ที่มีอยู่ในบิซิเนสเซอร์วิส (Business Service) โดยจะมีไบน์ดิงเทมเพลต (Binding Template) ทำหน้าที่เชื่อมโยงไปยังส่วนที่ 2

2) tModel คือส่วนในการค้นหาซึ่ง ถือเป็นหัวใจสำคัญของการค้นหาบนยูดีดีไอ เพราะ tModel จะมีหน้าที่เป็นดัชนี (index) ของยูดีดีไอ โดยจะมีนิยามประเภท กลุ่ม หรือหมวดหมู่ของเซอร์วิสและอ้างอิงไปยังเอกสาร WSDL

รูปที่ 2.3 อธิบายถึงโครงสร้างข้อมูลของยูดีดีไอ โดยบิซิเนสเอนทิตีเป็นตัวแทนของผู้ให้บริการที่ต้องการประกาศเผยแพร่เซอร์วิสของตน ซึ่งบิซิเนสเอนทิตีจะเป็นหน่วยใหญ่สุดของเว็บเซอร์วิสในยูดีดีไอ ซึ่งมีข้อมูลเกี่ยวกับบิซิเนสเซอร์วิสที่ตนให้บริการ กล่าวคือบิซิเนสเซอร์วิสเป็นส่วนหนึ่งของบิซิเนสเอนทิตี ซึ่งหนึ่งบิซิเนสเอนทิตีสามารถมีได้หลายบิซิเนสเซอร์วิส โดยบิซิเนสเซอร์วิสจะเก็บรายละเอียดเกี่ยวกับตัวเซอร์วิสและหมวดหมู่ของตัวเซอร์วิสรวมถึงไบน์ดิงเทมเพลตซึ่งเป็นตัว แทนของข้อมูลที่อยู่ภายในบิซิเนสเซอร์วิส มีหน้าที่เชื่อมโยงไปยัง tModel ซึ่งช่วยให้การค้นหาข้อมูลของเว็บเซอร์วิสง่ายขึ้น โดยในปัจจุบันยูดีดีไอตั้งแต่เวอร์ชัน 2 สามารถประกาศความสัมพันธ์ของบิซิเนสเซอร์วิสที่จะใช้งานร่วมกัน เรียกว่า Publisher Assertion ซึ่งจะอธิบายรายละเอียดของการประกาศ Publisher Assertion ในหัวข้อที่ 2.2



รูปที่ 2.3 โครงสร้างข้อมูลของยูดีดีไอ (http://uddi.xml.org)

2.1.2.1 การใช้งานยูดีดีไอ

ในการจัดการเว็บเซอร์วิสในยูดีดีไอนั้น ยูดีดีไอได้จัดเตรียมฟังก์ชันสำหรับการใช้งาน โดยมีฟังก์ชันอยู่ 2 กลุ่ม (ในที่นี่จะนำเสนอในส่วนของยูดีดีไอ เวอร์ชัน 2) ดังนี้คือ

กลุ่มที่ 1 ฟังก์ชันสำหรับการสืบค้นและดึงข้อมูลจากยูดีดีไอ

ชื่อฟังก์ชัน	หน้าที่
find_business	ค้นหาบิซิเนส
find_service	ค้นหาเว็บเซอร์วิส
find_binding	ค้นหาจุดให้บริการ
find_tModel	ค้นหา tModel (กลุ่ม หมวดหมู่ของเว็บเซอร์วิส)
find_relatedBusinesses	ค้นหาบิซิเนสที่มีความสัมพันธ์กัน
get_businessDetail	ดึงรายละเอียดของบิซิเนส
get_businessDetailExt	ดึงรายละเอียดของบิซิเนสเพิ่มเติม
get_serviceDetail	ดึงรายละเอียดของเว็บเซอร์วิส
get_bindingDetail	ดึงรายละเอียดของจุดให้บริการ
get_tModelDetail	ดึงรายละเอียดของ tModel

กลุ่มที่ 2 ฟังก์ชันสำหรับแก้ไขและบันทึกข้อมูลในยูดีดีไอ

ชื่อฟังก์ชัน	หน้าที่
get_authToken	ขอใช้สิทธิในการแก้ไขข้อมูลในยูดีดีไอ
discard_authToken	ยกเลิกการขอใช้สิทธิในการแก้ไขข้อมูลในยูดีดีไอ
save_business	บันทึกข้อมูลของบิซิเนส
save_service	บันทึกข้อมูลของเว็บเซอร์วิส
save_binding	บันทึกข้อมูลจุดให้บริการ
save_tModel	บันทึกข้อมูล tModel
delete_business	ลบข้อมูลของบิซิเนส
delete_service	ลบข้อมูลเว็บเซอร์วิส
delete_binding	ลบข้อมูลจุดให้บริการ
delete_tModel	ลบข้อมูล tModel

ต่อไปนี้เป็นตัวอย่างรูปแบบการเรียกใช้งานของฟังก์ชันต่างๆของยูดีดีไอบางส่วน ดังรูปที่ 2.4 ถึงรูปที่ 2.7

ในรูปที่ 2.4 เป็นวิธีการเรียกฟังก์ชัน get_authToken ซึ่งฟังก์ชันนี้จะต้องเรียกใช่ก่อนทุกครั้งก่อนที่จะทำการเรียกฟังก์ชันในกลุ่มที่ 2 ทั้งหมด เพราะการที่จะกระทำการใดๆกับยูดีดีไอได้จะต้องขอใช้สิทธิในการแก้ไขข้อมูลในยูดีดีไอก่อน

```
<get_authToken generic="2.0" xmlns="urn:uddi-org:api_v2" userID="***" cred="***"/>
```

รูปที่ 2.4 การขอใช้สิทธิในการแก้ไขข้อมูลในยูดีดีไอ

ในรูปที่ 2.5 เป็นวิธีการเรียกใช้ฟังก์ชัน save_business โดยต้องระบุรายละเอียดต่างๆเกี่ยวกับบิซิเนสเซอร์วิสเพื่อบันทึกในยูดีดีไอ โดย businessKey จะได้หลังจากการบันทึกข้อมูลบิซิเนสเซอร์วิสในยูดีดีไอแล้ว โดยคีย์นี้จะถูกกำหนดโดยยูดีดีไอ

```
<save_business  generic="2.0" xmlns="urn:uddi-org:api_v2">
        <authInfo>***</authInfo>
        <businessEntity businessKey="">
                <name>***</name>
                <description>***</description>
                <contacts>
                        <contact useType="***">
                                <personName>***</personName>
                                <phone>***</phone>
                                <email>***</email>
                        </contact>
                </contacts>
        </businessEntity>
</save_business>
```

รูปที่ 2.5 การบันทึกบิซิเนสเซอร์วิส

รูปที่ 2.6 เป็นวิธีการเรียกใช้ฟังก์ชัน delete_business โดยข้อมูลที่สำคัญที่ใช้ในการลบบิซิเนสเซอร์วิส คือ businessKey เท่านั้น ถ้าไม่ทราบก็ไม่สามารถลบได้

```
<delete_business  generic="2.0"  xmlns="urn:uddi-org:api_v2">
        <authInfo>*****</authInfo>
        <businessKey>*****</businessKey>
</delete_business>
```

รูปที่ 2.6 การลบบิซิเนสเซอร์วิส

รูปที่ 2.7 เป็นวิธีการเรียกใช้ฟังก์ชัน find_business โดยสามารถระบุรูปแบบต้องการค้นหาได้ดังนี้คือ ค้นหาตามคุณสมบัติ ค้นหาโดยชื่อของบิซิเนสเซอร์วิส ค้นหาโดย URL ค้นหาโดย businessKey ค้นหาตามหมวดหมู่ และค้นหาตาม tModel

```
<find_business  maxRows="100" generic="2.0“ xmlns="urn:uddi-org:api_v2">
        <findQualifier>***</findQualifier>
        <name>***</name>
        <discoveryURLs>***</discoveryURLs>
        <identifierBag>***</identifierBag>
        <categoryBag>***</categoryBag>
        <tModelBag>***</tModelBag>
</find_business>
```

รูปที่ 2.7 การค้นหาบิซิเนสเซอร์วิส

จากรูปที่ยกตัวอย่างไปข้างต้น เป็นตัวอย่างการเรียกใช้ฟังก์ชันของยูดีดีไอบางส่วน ซึ่งจะพบว่าอยู่ในรูปของภาษาเอ็กซ์เอ็มแอล เพื่อจะส่งการร้องขอผ่านทาง SOAP ได้ แต่ภาษาที่ใช้ในการพัฒนาระบบในปัจจุบันที่รองรับการพัฒนาเว็บเซอร์วิส ได้จัดเตรียมฟังก์ชันต่างๆสำหรับการเรียกใช้ฟังก์ชันของยูดีดีไอ โดยไม่จำเป็นต้องเขียนในรูปเอ็กซ์เอ็มแอลเองซึ่งมีความรวดเร็วและสะดวกในการพัฒนามากขึ้น

จากทั้งหมดที่กล่าวมาข้างต้นในหัวข้อที่ 2.1.2 สรุปได้ว่ายูดีดีไอมีความสามารถในการจัดเก็บข้อมูลเกี่ยวกับเว็บเซอร์วิสที่ผู้ให้บริการเว็บเซอร์วิสนำมาลงทะเบียนไว้และผู้ใช้งานยูดีดีไอสามารถที่จะสืบค้นเพื่อหาเว็บเซอร์วิสที่เหมาะสมตรงกับความต้องการซึ่งกระบวนการในการสืบค้นเว็บเซอร์วิสของยูดีดีไอนั้นอาศัยพื้นฐานการทำงานกับภาษาเอ็กซ์เอ็มแอลและ WSDL ซึ่งในปัจจุบันอาจไม่เพียงพอเนื่องจากยังคงเป็นการสืบค้นโดยการใช้คำสำคัญ (Keywords) ในการค้นหาซึ่งเป็นการสืบค้นตามชื่อของบิสซิเนส ชื่อของเซอร์วิส หรือตามประเภทของเซอร์วิสตามที่ระบุใน tModel เป็นต้น ทำให้มีโอกาสที่ผลลัพธ์จากการสืบค้นเว็บเซอร์วิสไม่ตรง หรือไม่เพียงพอกับความต้องการจริงของผู้ขอใช้บริการ และเว็บเซอร์วิสที่ลงทะเบียนในยูดีดีไอเพิ่มมากขึ้นเรื่อยๆทำให้การค้นหาเว็บเซอร์วิสในยูดีดีไอมีประสิทธิภาพลดลง ปัจจุบันจึงมีงานวิจัยที่นำเสนอแนวคิดและวิธีการในการเพิ่มประสิทธิภาพการค้นหาเว็บเซอร์วิสในยูดีดีไอออกมาอย่างต่อเนื่อง

2.2 เว็บเซอร์วิสกราฟ

ความเป็นมาของเว็บเซอร์วิสกราฟนั้นเริ่มจากการพัฒนางานวิจัยที่เกี่ยวข้องกับการค้นหาเว็บเซอร์วิสในยูดีดีไอ ซึ่งในงานวิจัยส่วนใหญ่จะใช้วิธีการค้นหาเว็บเซอร์วิสโดยใช้พารามิเตอร์เป็นเครื่องตรวจสอบความสัมพันธ์ของเว็บเซอร์วิสนั้นๆ ซึ่งจะต้องค้นหาเกือบทั้งยูดีดีไอ จึงได้มี

การพยายามจัดรูปแบบเว็บเซอร์วิสเพื่อให้อยู่ในรูปแบบของกราฟขึ้นและลดขอบเขตการค้นหาในยูดีดีไอ เพื่อให้สามารถทำทฤษฎีโครงสร้างข้อมูลและคุณสมบัติของกราฟมาประยุกต์ใช้กับเว็บเซอร์วิสได้ ซึ่งทำให้สามารถจัดการกับเว็บเซอร์วิสเป็นเรื่องง่ายขึ้น ดังนั้นการค้นหาและจัดกลุ่มของเว็บเซอร์วิสโดยใช้โครงสร้างข้อมูลเว็บเซอร์วิสกราฟจึงถูกนำเสนอโดยใช้วิธีการที่ต่างออกไปจากวิธีการค้นหาและจัดกลุ่มเว็บเซอร์วิส แบบเดิม

ในปี ค.ศ.2006 Jianxun Liu และ Lian Choa [3] ได้นำเสนอวิธีการค้นหาและจัดกลุ่มเว็บเซอร์วิสโดยใช้โครงสร้างข้อมูลแบบเว็บเซอร์วิสกราฟโดยมี 2 ขั้นตอนหลักด้วยกันคือขั้นตอนแรกทำการสร้างเว็บเซอร์วิสกราฟ เพื่อจัดกลุ่มและลดพื้นที่ในการค้นหาเว็บเซอร์วิสในยูดีดีไอโดยใช้รูปแบบการทำงานของยูดีดีไอที่มีในเวอร์ชัน 2 ขึ้นไปที่เรียกว่า Publisher Assertion เพื่อค้นหาความสัมพันธ์เว็บเซอร์วิสนั้นกับเว็บเซอร์วิสอื่นที่เกี่ยวข้องกันที่ผู้ให้บริการให้ลงทะเบียนไว้โดยจะเก็บเว็บเซอร์วิสไว้ในรูปของเมตริกซ์ และขั้นตอนที่สองจะทำการค้นหาเส้นทางที่ดีที่สุดในเว็บเซอร์วิสกราฟโดยใช้ขั้นตอนวิธีการค้นหาแบบแนวกว้างก่อนเพื่อให้ได้กลุ่มและลำดับของเว็บเซอร์วิสที่จะใช้งานตามการร้องขอ

ในปี ค.ศ.2007 ผู้วิจัยกลุ่มเดียวกันได้นำเสนองานวิจัย [4] ซึ่งเป็นงานวิจัยที่ต่อเนื่องจากงานวิจัย[3] โดยเสนอวิธีการใช้งานยูดีดีไอให้สามารถทำงานร่วมกับโครงสร้างข้อมูลแบบเว็บเซอร์วิสกราฟได้โดยจากงานวิจัย [3] ได้นำเสนอการเก็บเว็บเซอร์วิสกราฟในรูปของเมตริกซ์ แต่ใน [4] ได้ปรับปรุงโดยใช้วิธีการเก็บซึ่งคล้ายกับดัชนีของฐานข้อมูล

โดยสรุป เว็บเซอร์วิสกราฟคือการจัดกลุ่มเว็บเซอร์วิสที่มีความสัมพันธ์กันให้อยู่ในรูปโครงสร้างข้อมูลแบบกราฟ และกราฟที่ได้นั้นจะเป็นกราฟแบบไม่มีวงแบบระบุทิศทางซึ่งจากงานวิจัย [3] และ [4] นำเสนอการจัดการเว็บเซอร์วิสที่อยู่ในยูดีดีไอให้อยู่ในรูปโครงสร้างข้อมูลแบบเว็บเซอร์วิสกราฟ โดยนำเสนออยู่ 3 ส่วนคือ การสร้างเว็บเซอร์วิสกราฟ การเก็บเว็บเซอร์วิสกราฟ และการค้นหาเส้นทางในเว็บเซอร์วิสกราฟเพื่อนำไปใช้งาน

2.2.1 การสร้างเว็บเซอร์วิสกราฟ

ในการสร้างเว็บเซอร์วิสกราฟจะใช้คุณสมบัติใหม่ของยูดีดีไอซึ่งมีอยู่ในเวอร์ชัน 2 ขึ้นไปที่เรียกว่า Publisher Assertion ด้วยคุณสมบัตินี้จะอนุญาตให้กำหนดความสัมพันธ์ของบิซิเนสเอนทิ-ตี และทำหน้าที่เป็นคีย์อ้างอิง ซึ่งมีหน้าที่บอกความสัมพันธ์ระหว่าง 2 บิซิเนสเอนทิตี เมื่อยูดีดีไอตรวจสอบความถูกต้องเรียบร้อยการสอบถามข้อมูลของความสัมพันธ์ระหว่างเว็บเซอร์วิสก็จะสามารถทราบได้โดยเรียกใช้ได้ผ่านทางยูดีดีไอ ผ่านทางส่วนการประกาศที่เรียกว่า Publisher Assertion โดยประกาศในลักษณะของภาษาเอ็กซ์เอ็มแอล ซึ่ง Publisher Assertion ทำหน้าที่เป็นคีย์อ้างอิงที่มีหน้าที่บอกความสัมพันธ์ระหว่าง 2 บิซิเนสเอนทิตี (จะมีเฉพาะ ยูดีดีไอในเวอร์ชัน 2 ขึ้นไป โดยสามารถทำได้ดังตัวอย่างของการประกาศ Publisher Assertion ดังรูปที่ 2.8

```
<add_publisherAssertsions generic=" 2.0" Xmlns="urn: uddi-org:api_v2">
 <AuthInfo>[authinfo]</AuthInfo>
 <PublisherAssertion>
 <!-BK1 represent one department businesskey,BK2 represent another department businesskey->
   <fromKey>BK1</from>
   <toKey>BK2</toKey>
   <keyedReference tmodelKey="uuid:tbd" keyName="Holding Company"  keyValue=
"peer-peer" />
 </PublisherAssertion>
</add_publisherAssertsions>
```

รูปที่ 2.8 การประกาศ publisher Assertion

จากรูปที่ 2.8 จะเป็นการประกาศความสัมพันธ์ของ 2 บิซิเนสเอนทิตีของสองแผก 2 แผนก กล่าวคือ BK1 และ BK2 เป็นคีย์ของบิซิเนสเอนทิตีของแผนกที่ 1 และ บิซิเนสเอนทิตีของแผนกที่ 2 ตามลำดับ

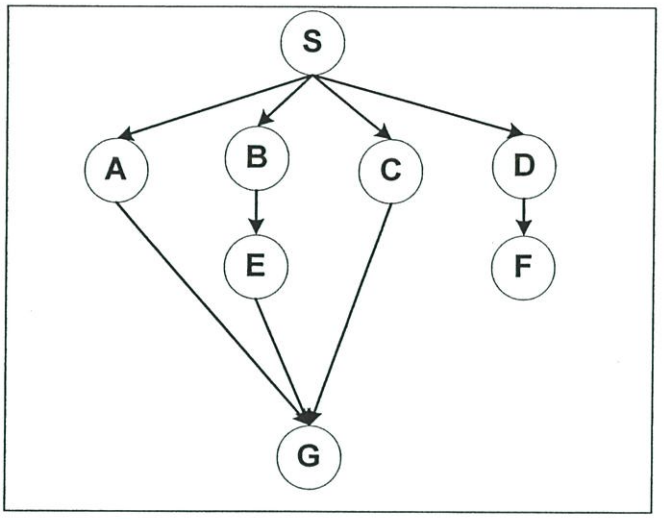
ขั้นตอนวิธีในการสร้างเว็บเซอร์วิสกราฟทั้งหมดสามารถแสดงได้ดังรูปที่ 2.9 โดยจะทำการ ค้นหา Publisher Assertion ผ่านทางฟังก์ชันของยูดีดีไอชื่อ find_relatedBusinesses เมื่อสร้างเว็บเซอร์วิสกราฟแล้วจะเก็บไว้ในรูปแบบของตารางดัชนีซึ่งจะแสดงรายละเอียดของการเก็บในหัวข้อที่ 2.2.2 โดยก่อนที่จะเก็บจะทำการตรวจสอบก่อนเพื่อไม่ให้กราฟมีการวนกลับโดยเริ่มจากการค้นหาความสัมพันธ์ระหว่างบิซิเนสเอนทิตีถ้ามีความสัมพันธ์แบบมีทิศทาง (directed link) ระหว่างเว็บเซอร์วิสในบิซิเนสเอนทิตีที่มีความสัมพันธ์กัน โดยพิจารณาจากพารามิเตอร์ตามวิธีการหาความสัมพันธ์ของเว็บเซอร์วิส กล่าวคือถ้ากำหนดให้ W_1 และ W_2 คือเว็บเซอร์วิสสองเว็บเซอร์วิสโดย W_1 จะมีความสัมพันธ์แบบมีทิศทางกับ W_2 ก็ต่อเมื่อ พารามิเตอร์ส่งออก (output parameter) ของ W_1 เป็นสับเซตของ พารามิเตอร์รับเข้า (input parameter) ของ W_2 หรือ W_1 .output $\supseteq$ W_2 .input ซึ่งผลที่ได้จากขั้นตอนวิธีการสร้างเว็บเซอร์วิสกราฟในรูปที่ 2.9 จะได้กราฟแบบไม่มีวงแบบระบุทิศทางดังรูปที่ 2.10

```
Function WSGConstruction(){
Input: The UDDI repository
Output: Web service graph(semantic overlay)
WSG=new graph()
While(UDDI repository not emthy){
        FromBE=Select one unvisited business entity from UDDI repository;
        ToBESet=Find the related businessEntities of FromBE by invoking find_relatedBusinesses
API;
        WS_formBE = Get all the Web services published by fromBE business entity
        for(each related business,ToBE,in ToBESet) {
            WS_toBE=Get all the web services published by toBE business entity
            for(each Web service, WS_from, in WS_fromBE)
               for(each Web service, WS_to, in WS_toBE)
                  if((W_from .out ⊇ W_to .in)
                     WSG.addEdge(WS_from, WS_to) ;// add relation index
        }
        if(All business entities are visited)
          return WSG;
 }
return empty;
}
```

รูปที่ 2.9 ขั้นตอนวิธีการสร้างเว็บเซอร์วิสกราฟ [3]

รูปที่ 2.10 เป็นตัวอย่างของเว็บเซอร์วิสกราฟโดยที่ S, A, B, C, D, E, F,G คือเว็บเซอร์วิสที่อยู่ในเว็บเซอร์วิสกราฟ S เป็นเว็บเซอร์วิสที่เก็บในเว็บเซอร์วิสกราฟเป็นเว็บเซอร์วิสแรกซึ่งจะถูกเรียกใช้ก่อนและเว็บเซอร์วิสสุดท้ายที่ต้องเรียกใช้งานคือเว็บเซอร์วิส G จากรูปที่ 2.6 เว็บเซอร์วิสกราฟนี้มีเส้นทางทั้งหมด 4 เส้นทางคือ S->A->G, S->B->E->G, S->C->G และ S->D->F โดยเส้นทางแต่ละเส้นทางในกราฟคือลำดับของการเรียกใช้งานเว็บเซอร์วิส แต่เว็บเซอร์วิสกราฟนี้มี 3 เส้นทางที่ไปสู่คำตอบได้ หรือไปหาเว็บเซอร์วิสGได้ คือ S->A->G, S->B->E->G, S->C->G ส่วนอีกเส้นทางคือ S->D->F เป็นเส้นทางที่ไปไม่ถึงคำตอบหรือ ซึ่งเส้นทางที่ไปไม่ถึงคำตอบนี้เป็นผล

จากขั้นตอนการสร้างเว็บเซอร์วิสกราฟในส่วนของการตรวจสอบเพื่อป้องกันกราฟวนกลับไปหาเว็บเซอร์วิสที่มีอยู่ในเว็บเซอร์วิสกราฟแล้ว(กราฟแบบไม่มีวงแบบระบุทิศทาง)



รูปที่ 2.10 เว็บเซอร์วิสกราฟ

จากขั้นตอนวิธีการสร้างเว็บเซอร์วิสกราฟในรูปที่ 2.9 เมื่อผู้วิจัยได้พิจารณาในเรื่องของอัตราการเติบโตของฟังก์ชันแล้ว เมื่อกำหนดเว็บเซอร์วิสในยูดีดีไอมีจำนวน n เว็บเซอร์วิส และให้โอกาสที่เว็บเซอร์วิสหนึ่งๆ จะมีความสัมพันธ์ไปยังเว็บเซอร์วิสอื่นคือ p ดังนั้นจะทำให้ทั้งเว็บเซอร์วิสกราฟ มีเส้นความสัมพันธ์ทั้งหมดจำนวน pC_2^n ความสัมพันธ์ โดยที่ C_2^n คือการเลือกโดยไม่สนใจลำดับ (combinations) โดยในที่นี้เลือกจากสิ่งของ 2 สิ่ง(คือมีความสัมพันธ์ และไม่มีความสัมพันธ์) จากเว็บเซอร์วิสทั้งหมดในอยู่ดีดีไอคือ n อัตราการเติบโตในขั้นตอนนี้จะประกอบไปด้วยในส่วนของการสร้างเว็บเซอร์วิสกราฟทั้งหมดคือ pnC_2^n และในการตรวจสอบเพื่อทำให้เว็บเซอร์วิสกราฟเป็นกราฟแบบไม่มีวง (acyclic graph) pC_2^n (เพื่อตัดความสัมพันธ์ที่วนกลับออกไป) ดังนั้นอัตราการเติบโตของฟังก์ชันในขั้นตอนนี้คือ

$$
\begin{aligned}
\mathrm{T} &= pnC_2^n + pC_2^n \\
&= \left\{ pn\left(\frac{n!}{(n-2)!2!} \right) \right\} + \left\{ p\left(\frac{n!}{(n-2)!2!} \right) \right\} \\
&= \left\{ pn\left(\frac{n(n-1)(n-2)!}{2(n-2)!} \right) \right\} + \left\{ p\left(\frac{n(n-1)(n-2)!}{2(n-2)!} \right) \right\} \\
&= \left\{ pn\left(\frac{n(n-1)}{2} \right) \right\} + \left\{ p\left(\frac{n(n-1)}{2} \right) \right\}
\end{aligned}
$$

$$= \frac{p}{2}[\{n(n(n-1))\}+\{n(n-1)\}]$$

$$= \frac{p}{2}\{(n^3-n^2)+(n^2-n)\}$$

แต่ในที่นี้จะเห็นว่า $\frac{p}{2}$ เป็นค่าคงที่ดังนั้นจึงไม่พิจารณา

$$O(\max((n^3-n^2),(n^2-n)))$$

$$= O(n^3-n^2)$$

$$= O(\max(n^3,n^2))$$

$$= O(n^3)$$

ดังนั้นอัตราการเติบโตของฟังก์ชันในขั้นตอนนี้คือ $O(n^3)$ ซึ่งในการจะปรับปรุงในขั้นตอนนี้จะต้องคำนึงถึงจุดนี้ด้วยคือควรจะมีอัตราการเติบโตไม่เกิน $O(n^3)$ หลังการปรับปรุง

2.2.2 การเก็บเว็บเซอร์วิสกราฟ

การเก็บเว็บเซอร์วิสกราฟซึ่งเดิมจากงานวิจัย [3] นั้นจะเก็บในรูปของเมทริกซ์ แต่ปัจจุบันในงานวิจัย [4] ได้นำเสนอวิธีเก็บอยู่ในรูปแบบของตารางดัชนี (Index table) คล้ายกับการเก็บข้อมูลในฐานข้อมูลซึ่งทำให้การเข้าถึงข้อมูลจะทำได้รวดเร็วมากกว่า โดยรูปแบบการเก็บมีรายละเอียดดังตารางที่ 2.1 โดยจะมีอยู่ 4 สดมภ์คือ BUSSINESS_KEY SERVICE_KEY LINK และ NAME โดย BUSSINESS_KEY และ SERVICE_KEY คือตัวเลขเดียวกับที่ได้จากการผู้ให้บริการทำการลงทะเบียนเว็บเซอร์วิสในยูดีดีไอ ซึ่งจะใช้เป็นคีย์หลัก (primary key) ส่วนสดมภ์ LINK จะเก็บ SERVICE_KEY ของเว็บเซอร์วิสที่มีความสัมพันธ์กับเว็บเซอร์วิสอื่น ถ้ามีความสัมพันธ์กับหลายเว็บเซอร์วิสก็จะคั่นด้วย เครื่องหมาย @ ระหว่าง SERVICE_KEY และฟังก์ชันที่ต้องจัดเตรียมไว้สำหรับการจัดการกับเว็บเซอร์วิสกราฟ [4] จะมีอยู่ 3 ฟังก์ชันด้วยกันคือ สร้างเว็บเซอร์วิสกราฟและเก็บไว้ในตารางดัชนี คือ WSG_build() แก้ไขเว็บเซอร์วิสกราฟในตารางดัชนี คือ WSG_update() และลบเว็บเซอร์วิสกราฟออกจากตารางดัชนี คือ WSG_delete()

ตารางที่ 2.1 ตารางดัชนี (Index table) [4]

Filed name	Field type	Is primary key?	Is Null?	Field length
BUSSINESS_KEY	VARCHAR	Yes	No	255
SERVICE_KEY	VARCHAR	Yes	No	255
LINK	TEXT	No	Yes	
NAME	VARCHAR	No	No	255

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

ตารางที่ 2.2 การเก็บเว็บเซอร์วิสกราฟในตารางดัชนี

BUSSINESS_KEY	SERVICE_KEY	LINK	NAME
210	201	212@211@310@314	S
111	212	101	A
111	211	124	B
310	310	101	C
401	314	501	D
401	124	101	E
089	501		F
202	101		G

ในตารางที่ 2.2 แสดงถึงตัวอย่างการเก็บเว็บเซอร์วิสกราฟจากรูปที่ 2.10 โดยในที่นี้ตัวเลขของ สดมภ์ BUSSINESS_KEY และ SERVICE_KEY เป็นตัวเลขที่สมมติขึ้นมาแทนตัวเลขที่ได้จากการลงทะเบียนเว็บเซอร์วิสในยูดีดีไอ เช่นในแถวแรก จะเป็นข้อมูลเว็บเซอร์วิส S โดยในสดมภ์ LINK จะเก็บ SERVICE_KEY ของเว็บเซอร์วิสที่มีความสัมพันธ์ กับเว็บเซอร์วิส S ในที่นี้มีอยู่ 4 SERVICE_KEY คือ เว็บเซอร์วิส A คือ 212 เว็บเซอร์วิส B คือ 211 เว็บเซอร์วิส C คือ 310 เว็บเซอร์วิส D คือ 211 โดยถูกขั้นไว้โดยเครื่องหมาย @ แต่ในส่วนของแถวที่เป็นข้อมูลของเว็บเซอร์วิส F และ G จะเห็นว่า สดมภ์ LINK ว่างเปล่าแสดงว่าทั้งสองเว็บเซอร์วิสนี้เป็นจุดสิ้นสุดของเส้นทางนั้นๆ หรือไม่มีความสัมพันธ์ต่อไปยังเว็บเซอร์วิสอื่น

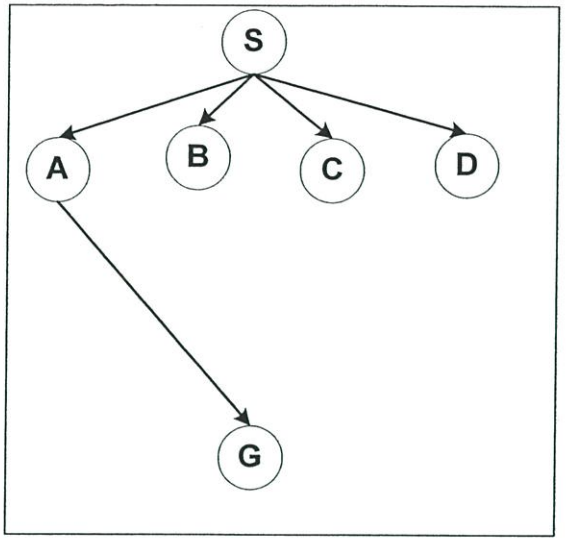
2.2.3 การค้นหาเส้นทางในเว็บเซอร์วิสกราฟ

ในการหาเส้นทางจะค้นหาเส้นทางที่ดีที่สุดในเว็บเซอร์วิสกราฟออกมา ซึ่งในขั้นตอนนี้จะทำได้ไม่ยากเนื่องจากในขั้นตอนนี้เว็บเซอร์วิสได้จัดให้อยู่ในรูปของกราฟ โดยทำการค้าหาเส้นทางที่ดีที่สุดเว็บเซอร์วิสกราฟโดยใช้ขั้นตอนวิธีค้นหาแบบแนวกว้างก่อน ซึ่งจะได้เส้นทางที่ดีที่สุดที่แสดงถึงลำดับและขั้นตอนในการเรียกใช้งานเว็บเซอร์วิสโดยแสดงขั้นตอนวิธีการค้นหาเส้นทางในเว็บเซอร์วิสกราฟ ได้ดังรูปที่ 2.11

87113

```
Function WSGMatch () {
Input : service request ;
Output : service composition flow
WSG=new Graph(); // Initialize service request Web service graph object
SCFset =null; // Initialize service composition flow set
EdgeSet=Over search the WSG, according to BFS (All edge and node that were visited)
Maxmatch=0;// Initalize matching degree
While (EdgeSet is not the last) {
        toEdgeSet[]=Find in semantic overlay //Find in index table
        for (each ToEdgeSet) {
           While (each edge in EdgeSet can match edge in ToEdgeSet) {
                  If (edge in ToEdgeSet) {
                      Match=match+1;
                      Edge=next edge in EdgeSet;
                      SCFSet=SCFset + edge in ToEdgeset
                  }Else
                      Edge=next edge in EdgeSet
            }
             if(maxmatch=match)
                return SCFset;
          }
       if (EdgeSet is at the last)
          return SCFset;
}
return empty ;
```

รูปที่ 2.11 ขั้นตอนวิธีค้นหาเส้นทางในเว็บเซอร์วิสกราฟ [3]



รูปที่ 2.12 เว็บเซอร์วิสทั้งหมดที่ถูกสำรวจในการค้นหาเส้นทาง

รูปที่ 2.12 แสดงถึงลำดับของเว็บเซอร์วิสทั้งหมดที่ทำการสำรวจในเว็บเซอร์วิสกราฟในการค้นหาเส้นทางที่ดีที่สุดโดยใช้ขั้นตอนวิธีการค้นหาแบบแนวกว้างก่อนโดยทำการค้นจากรูปที่ 2.10ซึ่งเส้นทางที่ดีที่สุดที่ค้นได้คือ S->A->G โดยเริ่มสำรวจทีละระดับชั้น เริ่มจากซ้ายไปขวา เริ่มต้นจากเว็บเซอร์วิส S เป็นระดับแรก ระดับที่ 2 คือ A, B, C, D และระดับสุดท้ายคือ G ซึ่งถ้าพิจารณาจากรูปที่ 2.10 จะเห็นว่าเส้นทาง S->C->G เป็นเส้นทางที่ดีที่สุดเช่นเดียวกับเส้นทาง S->A->G แต่ในที่นี้ทำการค้นจากซ้ายไปขวาเมื่อเจอคำตอบซึ่งในที่นี้คือเว็บเซอร์วิส G ก็ทำการหยุดค้นจึงได้คำตอบคือ S->A->G

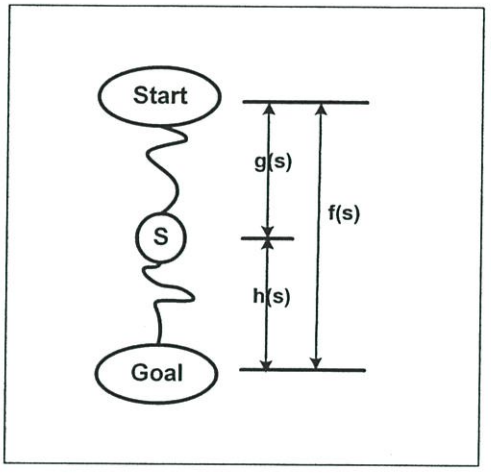
2.3 ขั้นตอนวิธีการค้นหาแบบ A*

ขั้นตอนวิธีการค้นหาแบบ A* เป็นการเพิ่มความสามารถของขั้นตอนวิธีการค้นหาแบบดีที่สุดก่อน (best first search) กล่าวคือการค้นหาแบบดีที่สุดก่อน จะเก็บทุกโหนด (node) โดยไม่มีการตัดทิ้งไป การไม่ตัดทิ้งนี้จึงทำให้ขั้นตอนวิธีการค้นหาแบบดีที่สุดก่อนไม่พลาดเส้นทางที่นำไปสู่เส้นทางที่นำไปสู่คำตอบเลยโดยแต่ละขั้นตอนจะเลือกโหนดที่มีค่าฮิวริสติกดีที่สุด โดยพิจารณาจากทุกโหนดที่ยังไม่ถูกกระจาย คือสถานะที่ยังไม่ได้สร้างโหนดลูก (child node) แต่ขั้นตอนวิธีการค้นหาแบบ A* จะพิจารณาเพิ่มถึงต้นทุนจากโหนดเริ่มต้นมายังโหนดปัจจุบันเพื่อใช้คำนวณหาค่าฮิวริสติกด้วย ในกรณีของขั้นตอนวิธีการค้นหาแบบ A* ฟังก์ชันที่ใช้หาต้นทุนที่น้อยที่สุดคือ f(s) ของโหนด s ใดๆ ซึ่งหาได้จาก $f(s) = g(s) + h(s)$ โดยที่

g(s) คือ ฟังก์ชันฮิวริสติกที่ใช้สำหรับการคำนวณต้นทุนจากโหนดเริ่มต้นมายังโหนดปัจจุบัน

h(s) คือ ฟังก์ชันฮิวริสติกที่ใช้สำหรับการคำนวณการประมาณต้นทุนจากโหนดปัจจุบันไปยังคำตอบ

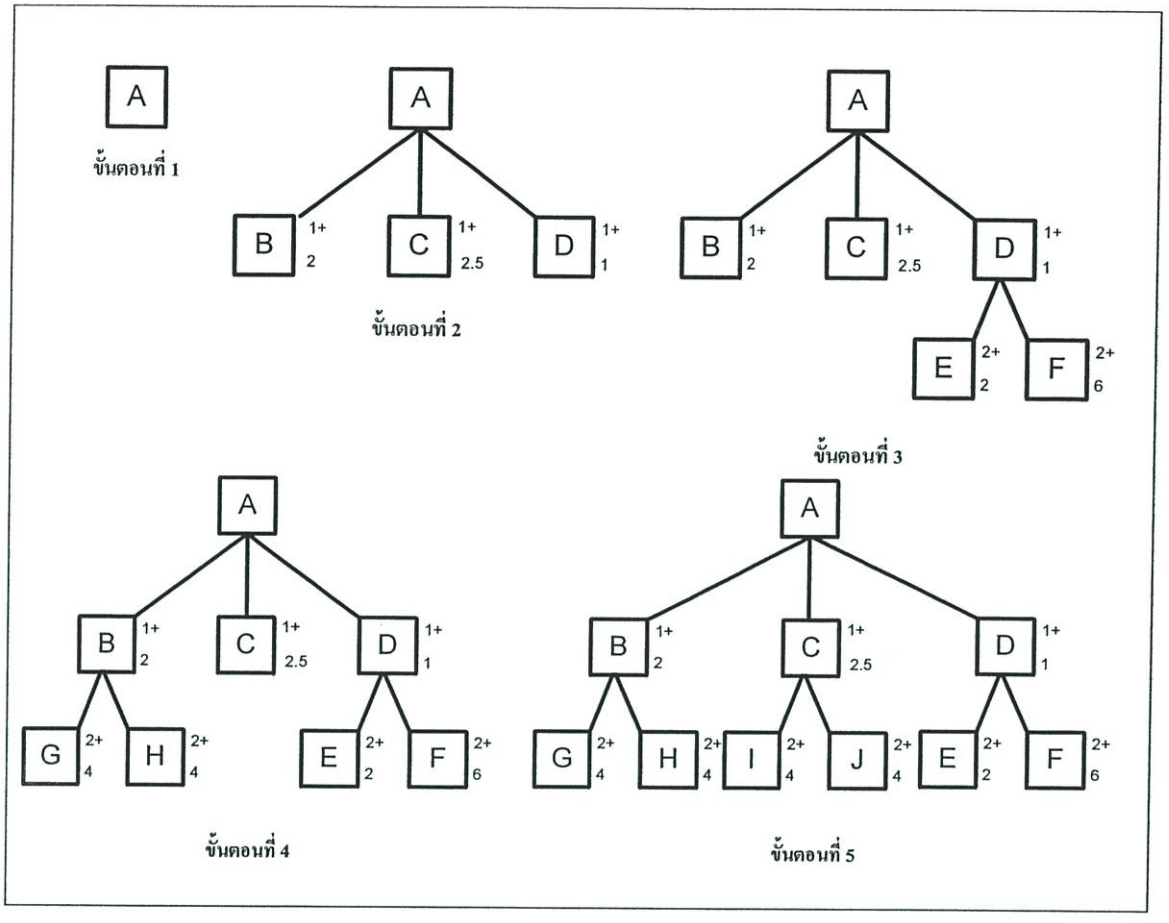
ดังนั้นจึงสรุปได้ว่า f(s) เป็นฟังชั่นที่สำหรับการคำนวณการประมาณต้นทุนจากสถานะเริ่มต้นไปยังสถานะคำตอบ (ยิ่งมีค่าน้อยเท่าไหร่ยิ่งดี) แสดงได้ดังรูปที่ 2.13



รูปที่ 2.13 ขั้นตอนวิธีการค้นหาแบบ A*

โดยทั่วไปฟังก์ชัน h(s) คือฟังก์ชันฮิวริสติกที่ใช้สำหรับการค้นหาแบบอื่น เช่น ขั้นตอนวิธีการค้นหาแบบปีนเขา (Hill climbing) และขั้นตอนวิธีการค้นหาแบบดีที่สุดก่อน เป็นต้น แต่สำหรับฟังก์ชัน h(s) ของขั้นตอนวิธีการค้นหาแบบ A* นั้นจะเป็นการประมาณการมากกว่า กล่าวคือทำการประมาณต้นทุนจากโหนดปัจจุบันไปโหนดคำตอบเพราะจะรู้ต้นทุนจริงก็ต่อเมื่อได้ทำการค้นหาจริงจากโหนดปัจจุบันไปยังคำตอบแล้วเท่านั้น ส่วนฟังก์ชัน g(s) ไม่ใช่ฟังก์ชันฮิวริสติกที่ประมาณต้นทุน เช่นเดียวกับฟังก์ชัน h(s) เพราะสามารถหาต้นทุนจริงได้เนื่องจากได้ทำการค้นหาจากจุดเริ่มต้นจนมาถึงตำแหน่งปัจจุบันแล้ว ดังนั้นฟังก์ชัน f(s) จึงจัดได้ว่าเป็นฟังก์ชันฮิวริสติกที่ประมาณต้นทุนจากจุดเริ่มต้นไปจนถึงคำตอบ แต่ไม่ใช่การคำนวณต้นทุนจริง

ขั้นตอนวิธีการค้นหาแบบ A* นั้นใช้ฟังก์ชัน f(s) จึงต้องให้ความสำคัญกับอยู่สองประการ คือ (1) โหนดที่ดีเหมาะสมแก่การเลือกต้องมีค่าของฟังก์ชัน h(s) ที่คำนวณได้น้อยๆ กล่าวคือต้นทุนที่จะนำไปสู่คำตอบหลังจากนี้จะต้องมีค่าน้อยกว่าหรือเท่ากับค่าของฟังก์ชัน h(s) ของโหนดก่อนหน้าเท่านั้นจะมากกว่าไม่ได้และ (2) ต้นทุนที่จ่ายไปแล้วกระทั่งถึงโหนดปัจจุบัน หรือค่าของฟังก์ชัน g(s) ที่คำนวณได้นั้นต้องน้อยด้วย เราจึงได้ว่า ขั้นตอนวิธีการค้นหาแบบ A* จะค้นหาเส้นทางที่มีต้นทุนโดยรวมน้อยที่สุด



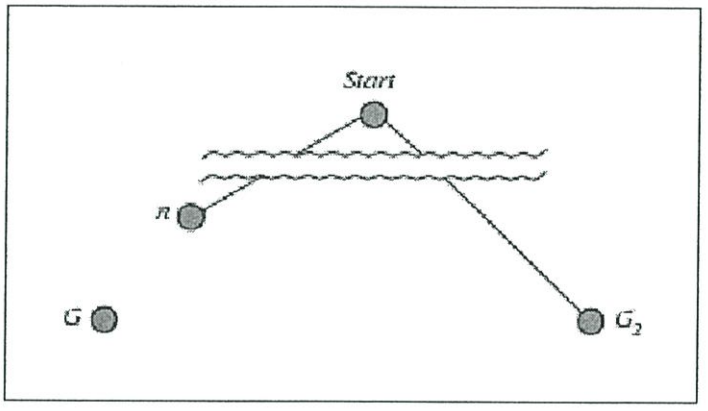
รูปที่ 2.14 การค้นหาโดยใช้ขั้นตอนวิธีการค้นหาแบบ A*

จากรูปที่ 2.14 แสดงตัวอย่างของการค้นหาด้วยขึ้นตอนวิธีการค้นหาแบบ A* โดยสมมติให้ว่า ต้นทุน หรือ ระยะห่างจากโหนดพ่อแม่ (parent node) ไปยังโหนดลูก โดยเส้นเชื่อมแต่ละเส้นเชื่อม มีต้นทุนเท่ากับ 1 หน่วย เช่น ค่าต้นทุนจริงจาก A ไปยัง B หรือ C ไปยัง D มีค่าเท่ากับ 1 หน่วย ซึ่งในที่นี้ก็คือฟังก์ชัน g(s) นั่นเอง จากรูปที่ 2.10 ในขั้นตอนที่ 4 โหนด C จะถูกเลือกมากระจายโดยขั้นตอนวิธีการค้นหาแบบ A* เนื่องจากมีค่าของฟังชั่น f(s) ที่คำนวณได้น้อยที่สุด ซึ่งมีค่าเท่ากับ 3.5 ซึ่งมีค่าน้อยกว่าโหนด E ที่มีค่าเท่ากับ 4 แม้ว่าค่าฟังก์ชัน h(s)ที่คำนวณได้ ซึ่งเท่ากับ 2 จะมีค่าน้อยกว่าโหนด C ที่มีต้นทุนทีคำนวณได้จากฟังก์ชัน h(s) เท่ากับ 2.5 ซึ่งมากกว่าก็ตาม

จากคุณสมบัติของฟังก์ชันฮิวริสติกของขั้นตอนวิธีการค้นหาแบบ A* ดังที่กล่าวจากข้างต้นและจากตัวอย่างในรูปที่ 2.14 ขั้นตอนวิธีการค้นหาแบบ A* จะได้คำตอบที่ดีที่สุด (optimal) เช่นเดียวกับขั้นตอนวิธีการค้นหาแนวกว้างก่อน แต่โดยปรกติแล้วการที่ขั้นตอนวิธีการค้นหาแบบ A* จะได้คำตอบที่ดีที่สุดนั้นจะต้องประกอบด้วยคุณสมบัติ 2 ประการคือ 1) Admissible Heuristic และ 2) Monotonic (Consistent Heuristics)

2.3.1 Admissible Heuristic

คุณสมบัติ Admissible Heuristic มีความหมายว่าฟังก์ชัน h(s) จะต้องมีค่าต้นทุนที่คำนวณได้ไม่เกิน h*(s) โดยที่ h*(s) คือต้นทุนจริงจากโหนดปัจจุบันไปยังคำตอบที่อยู่บนเส้นทางที่ดีที่สุด (optimal path) กล่าวคือ $0 \leq h(s) \leq h^*(s)$



รูปที่ 2.15 คุณสมบัติ Admissible Heuristic

จากรูปที่ 2.15 ถ้าสมมติว่ามีเป้าหมายอยู่ 2 โหนดคือ G และ G_2 โดยที่เส้นทางไปสู่ G เป็นเส้นทางที่ดีที่สุด กำหนดให้ C* เป็นต้นทุนจากจุดเริ่มต้น (start) ไปหาโหนด G (หรือ $C^* = f(G)$) และ n คือโหนดบนเส้นทางที่ดีที่สุดก็จะได้ความสัมพันธ์ดังนี้คือ

$f(G_2) = g(G_2)$ เพราะ $h(G_2) = 0$

$f(G) = g(G) = C^*$ เพราะ $h(G) = 0$

$g(G_2) \rangle g(G)$ เพราะ เส้นทางจากสถานะเริ่มต้นไปถึง G เป็นเส้นทางที่ดีที่สุด

$f(G_2) \rangle f(G)$ เพราะ เส้นทางจากสถานะเริ่มต้นไปถึง G เป็นเส้นทางที่ดีที่สุด

ถ้าให้ n เป็นโหนดบนเส้นทางที่ดีที่สุดที่ไปสู่ G จะได้ว่า

$h(n) \leq h^*(n)$ เมื่อฟังก์ชัน h(n) มีคุณสมบัติ Admissible Heuristic

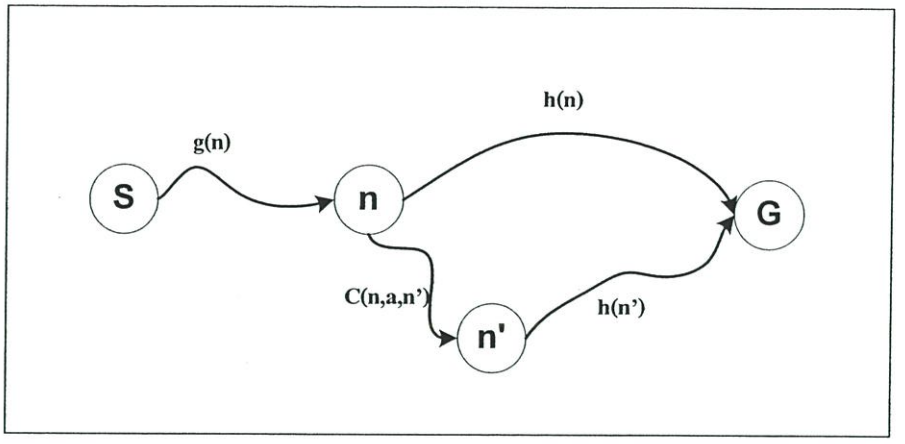
$g(n) + h(n) \leq g(n) + h^*(n)$ เมื่อฟังก์ชัน h(n) มีคุณสมบัติ Admissible Heuristic

$f(n) \leq f(G)$

ดังนั้นขั้นวิธีการค้นหาแบบ A* จะไม่เลือกไปเส้นทางที่ไปสู่สถานะ G_2 ซึ่งก็คือได้เส้นทางที่ดีที่สุด สรุปแล้วสิ่งที่จำเป็นต้องพิจารณาว่ามีคุณสมบัตินี้หรือไม่คือ $0 \leq h(s) \leq h^*(s)$

2.3.2 Monotonic

คุณสมบัตินี้มีความหมายว่าค่าที่คำนวณได้จากฟังก์ชัน f(s) ตลอดเส้นทางในแต่ละเส้นทางจะต้องไม่ลดลง



รูปที่ 2.16 คุณสมบัติ Monotonic (Consistent Heuristics)

เมื่อพิจารณาจากรูปที่ 2.16 ประกอบ โดยกำหนดให้ S คือโหนดเริ่มต้น n คือโหนดที่สนใจ G คือคำตอบ และกำหนดให้ h(n) มีคุณสมบัติ Admissible Heuristic C(n,a,n') คือต้นทุนจากโหนด n ไปยังโหนด n' โดย a คือการกระทำใดๆที่ทำให้ จากโหนด n เลือกไปสู่โหนด n' ดังนั้น f(n) จะมีคุณสมบัติ Monotonic เมื่อ

$f(n) \leq f(n')$ หรือ $h(n) - h(n') \leq C(n,a,n')$ และ

$f(n') = g(n') + h(n')$

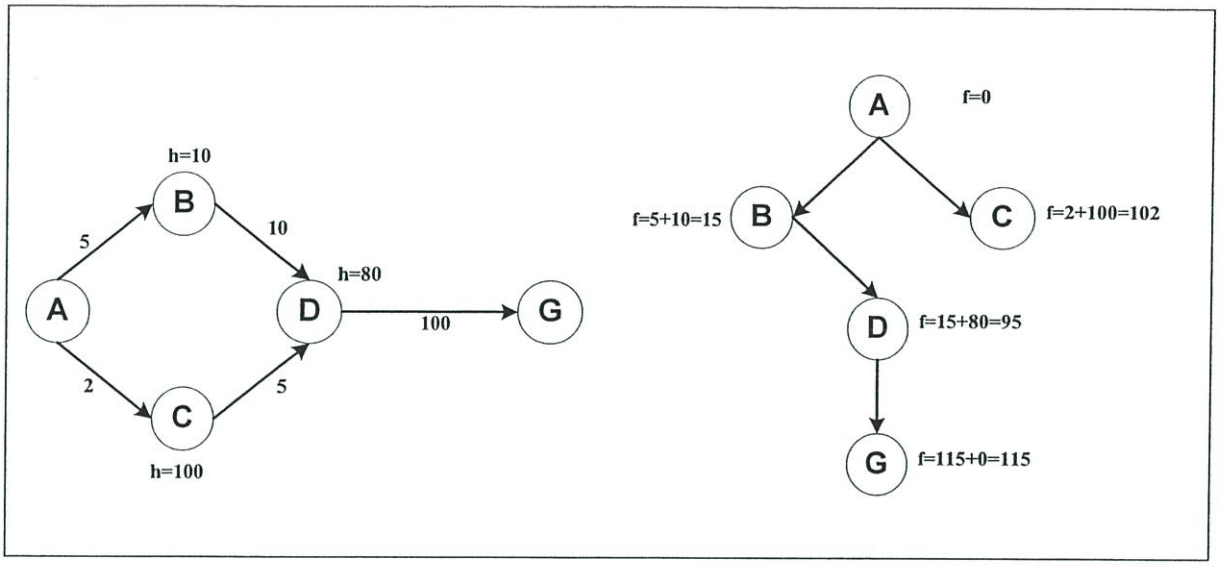
$f(n') = g(n) + c(n,a,n') + h(n')$

$f(n') \geq g(n) + h(n)$

$f(n') = f(n)$

ซึ่งอาจจะพิจารณาโดยวิธีข้อขัดแย้ง (contradiction) ว่าขั้นตอนวิธีการค้นหาแบบ A* จะให้คำตอบที่ดีที่สุด จะมีคุณสมบัติ Admissible Heuristic และ Monotonic โดยพิจารณารูปที่ 2.13 ประกอบ จะได้ว่า $f(G) = C^*$ และ $f(G_2) > C^*$ ถ้า ขั้นตอนวิธีการค้นหาแบบ A* นี้ ให้ผลลัพธ์คือโหนด G_2 เมื่อ n คือสถานะบนเส้นทางที่ดีที่สุดดังนั้น $f(G_2) \leq f(n)$ จากคุณสมบัติ Monotonic เราจะต้องได้ว่า $C^* \geq f(n)$ หรือ $f(n') \geq f(n)$ ดังนั้นต้องทำให้ $C^* \leq f(G_2)$ ซึ่งเกิดข้อขัดแย้งเกิดขึ้น หรือสามารถพิจารณาง่ายๆได้ว่า f(n) มีคุณสมบัติ Monotonic หรือไม่นั้น ค่าของ f(n) ตลอดเส้นทางจะต้องไม่ลดลง

รูปที่ 2.17 แสดงถึงตัวอย่างของขั้นตอนวิธีการค้นหาแบบ A* ที่มีคุณสมบัติ Admissible Heuristic คุณสมบัติเดียวแต่ไม่มีคุณสมบัติ Monotonic จึงไม่สามารถได้คำตอบที่ดีที่สุดได้ ซึ่งจะเห็นว่าในที่นี้คำตอบที่ดีที่สุดคือเส้นทาง จาก A ไป B ไป D และสุดท้ายคือ G แต่จากรูปที่ 2.17 ได้เส้นทาง A ไป B ไป D และไป G ซึ่งไม่ใช่คำตอบที่ดีที่สุด



รูปที่ 2.17 A* ที่มีคุณสมบัติ Admissible คุณสมบัติเดียว

เมื่อพิจารณาที่คุณสมบัติ Admissible จาก $0 \leq h(s) \leq h^*(s)$ เมื่อพิจารณาค่า h(s) และ h*(s) ทีละโหนดในรูปที่ 2.17 จะมีค่าดังนี้คือ

h(A) = 0 h*(A) = 107

h(B) = 10 h*(B) = 110

h(C) = 100 h*(C) = 105

h(D) = 80 h*(D) = 110

h(G) = 0 h*(G) = 0

จากการพิจารณาในทุกโหนดพบว่าอยู่ในเงื่อนไข $0 \leq h(s) \leq h^*(s)$ แสดงว่ามีคุณสมบัติ Admissible

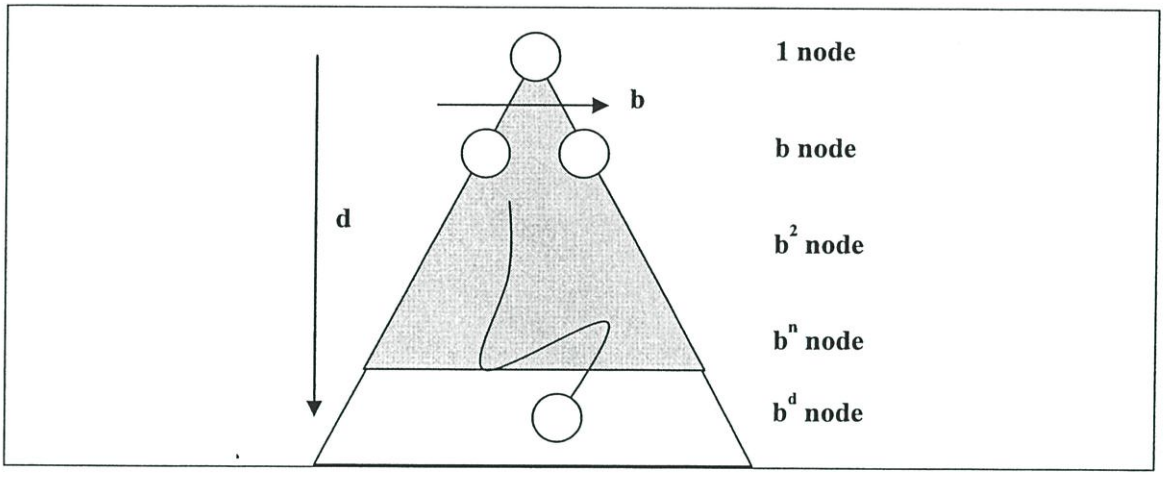
ในส่วนของคุณสมบัติ Monotonic จะพิจารณาค่า f(n) ว่าในตลอดเส้นทางจะต้องไม่ลดลง เมื่อพิจารณาที่โหนด C ซึ่งอยู่บนเส้นทางที่ดีที่สุดพบว่าที่โหนด C นั้นค่า f(C) คือ 102 ซึ่งจะเห็นว่าค่าของ f(s) ของโหหนด D มีค่า 95 ซึ่งมีค่าน้อยกว่า 102 และจาก $f(n) \leq f(n')$ ในกรณีนี้ n คือโหนด C และ n' คือโหนด B ซึ่งจะมีคุณสมบัติ Monotonic ก็ต่อเมื่อ $f(C) \leq f(B)$ ซึ่งในที่นี้เป็นเท็จ ดังนั้นตัวอย่างในรูปที่ 2.17 จึงมีคุณสมบัติ Admissible เพียงคุณสมบัติเดียว

เพราะฉะนั้นขั้นตอนวิธีการค้นหาแบบ A* จะได้คำตอบที่ดีที่สุดได้ จะต้องมีคุณสมบัติ 2 ข้อคือ Admissible Heuristic และMonotonic

2.3.3 อัตราการเติบโตของฟังก์ชัน

อัตราการเติบโตของฟังก์ชันของขั้นตอนวิธีการค้นหาแบบ A* จะขึ้นอยู่กับฟังก์ชันฮิวริสติกที่ใช้ โดยพิจารณาจาก 2 กรณีคือ

1) ทุกๆกรณีของ h(s) ทำให้ $|h(s) - h^*(s)| \leq O(\log(h^*(s)))$ หรือไม่ เมื่อ $h^*(s)$ คือต้นทุนของเส้นทางที่ดีที่สุดจากโหนดปัจจุบันไปยังคำตอบ อัตราเติบโตของฟังก์ชันจะอยู่ในรูปของพหุนาม (polynomial) ซึ่งสามารถพิจารณาจากขั้นตอนวิธีที่ใช้ตามปรกติ



รูปที่ 2.18 การหาอัตราการเติบโตของฟังก์ชัน

2) เมื่อไม่เป็นไปตามเงื่อนไขที่ 1 จะเป็นกรณีที่แย่ที่สุด (worst case) คือกรณีที่เข้าไปค้นหาในทุกเส้นเชื่อมและทุกโหนด อัตราการเติบโตของฟังก์ชันเป็นแบบเอ็กซ์โพเนนเชียล (exponential) ในการพิจารณากระทำได้คล้ายกับขั้นตอนวิธีการค้นหาแบบแนวกว้างก่อน [5] [6] โดยพิจารณารูปที่ 2.18 ประกอบ โดยกำหนดให้ b คือจำนวนโหนดลูกที่เกิดขึ้นได้ของโหนดนั้นๆ (branching factor) หรือเส้นทางที่จะไปได้จากโหนดปัจจุบัน ส่วน d คือความลึกของกราฟ ดังนั้นในระดับความลึก d จะได้ $T= (b^d) \times T_{1\ NODE}$ โดยที่ $T_{1\ NODE}$ คือค่าประมาณเวลาในการค้นหา 1 โหนด ซึ่งมีค่าเท่ากับ 1 หน่วยเวลา ดังนั้นจะได้ $T = b^d$ เพราะฉะนั้นอัตราการเติบโตของฟังก์ชันที่ได้คือ $O(b^d)$ หรือประมาณได้เท่ากับ หรือ $O(|E| + |V|)$ นั้นเอง โดยที่ $|E|$ คือ จำนวนเส้นเชื่อมทั้งหมดที่ทำการสำรวจ และ $|V|$ คือ จำนวนโหนดทั้งหมดที่ทำการสำรวจ

บทที่ 3
การลดพื้นที่การค้นหาสำหรับเว็บเซอร์วิสกราฟ

ในบทนี้จะนำเสนอถึงวิธีการลดพื้นที่ในการค้นหาสำหรับโครงสร้างข้อมูลแบบเว็บเซอร์วิสกราฟ ซึ่งได้กล่าวถึงในบทที่ 2 ในบทนี้จะนำเสนอการปรับปรุงในส่วนของขั้นตอนการค้นหาเส้นทางในเว็บเซอร์วิสกราฟ โดยใช้ขั้นตอนวิธีการค้นหาแบบ A* ซึ่งจะนำเสนอฟังก์ชันฮิวริสติก 2 แบบ

3.1 การปรับปรุงการค้นหาเส้นทางในเว็บเซอร์วิสกราฟ

จากบทที่ 2 ขัวข้อที่ 2.2.3 ในขั้นตอนการค้นหาเส้นทางในเว็บเซอร์วิสกราฟ ได้ใช้ขั้นตอนวิธีค้นหาแบบแนวกว้างก่อน ซึ่งคุณสมบัติหนึ่งของขั้นตอนวิธีค้นหาแบบแนวกว้างก่อนคือสามารถหาเส้นทางที่ดีที่สุดได้ แต่ในการค้นหาจะกระจายการค้นหาที่ละระดับชั้น ซึ่งผลในการค้นหาเส้นทางในเว็บเซอร์วิสกราฟคือจะต้องเข้าไปค้นเกือบทั่วทั้งเว็บเซอร์วิสกราฟโดยไม่จำเป็น ดังนั้นผู้วิจัยจึงมีความเห็นว่าควรนำขั้นตอนวิธีในการค้นหาแบบ A* มาใช้แทนขั้นตอนวิธีค้นหาแบบแนวกว้างก่อนเพราะสามารถหาเส้นทางที่ดีที่สุดได้เช่นเดียวกัน เนื่องจากขั้นตอนวิธีการค้นหาแบบ A* มีการคำนวณต้นทุนจากฟังก์ชันฮิวริสติกถึง 2 ค่าด้วยกันซึ่งได้กล่าวไปแล้วในบทที่ 2 โดยพิจารณาจากต้นทุนที่น้อยที่สุดก่อน หรือกล่าวคือพิจารณาค่า f(s) ที่น้อยที่สุดเป็นหลักโดยค่า f(s) คำนวณได้จาก f(s) = g(s) + h(s)โดยในที่นี้จะกำหนดให้ g(s) คือฟังก์ชันฮิวริส- ติกที่คำนวณต้นทุนจากเว็บเซอร์วิสเริ่มต้นมายังเว็บเซอร์วิสปัจจุบัน และ h(s) เป็นฟังก์ชันที่คำนวณต้นทุนจากเว็บเซอร์วิสปัจจุบันไปหาเว็บเซอร์วิสคำตอบ

เมื่อพิจารณาถึงรูปแบบของเว็บเซอร์วิสกราฟ เป็นกราฟแบบไม่มีวงแบบระบุทิศทาง และจากการศึกษาลักษณะการเก็บของเว็บเซอร์วิสกราฟ ณ เว็บเซอร์วิสใดๆจะสามารถทราบได้ว่ามีความสัมพันธ์กับเว็บเซอร์วิสใดต่อไปบ้างโดยดูได้จาก สดมภ์ชื่อ LINK ของแถวที่เก็บเว็บเซอร์วิสนั้นๆ ในตางรางดัชนี ซึ่งได้กล่าวถึงในบทที่ 2 ตารางที่ 2.1 และตัวอย่างในตารางที่ 2.2 จากคุณสมบัติข้อนี้ของเว็บเซอร์วิสกราฟทำให้สามารถนำมาใช้ประกอบการคำนวณของฟังก์ชันฮิวริส-ติกเพื่อใช้สำหรับการประมาณต้นทุนจากเว็บเซอร์วิสปัจจุบันไปยังเว็บเซอร์วิสคำตอบได้ โดยนำ SERVICE_KEY ที่มีอยู่ในสดมภ์ LINK ไปเทียบกับ SERVICE_KEY ของเว็บเซอร์วิสคำตอบหรือเว็บเซอร์วิสสุดท้ายที่ต้องเรียกใช้งาน ซึ่งถ้าตรงกันแสดงว่าจากเว็บเซอร์วิสปัจจุบันมีเส้นทางไปสู่คำตอบได้ ทำให้ไม่ต้องเข้าไปสำรวจเว็บเซอร์วิสอื่นๆอีก และเมื่อพิจารณาจากรูปแบบความสัมพันธ์ของพามิเตอร์ส่งออกของเว็บเซอร์วิส กล่าวคือพิจารณาจากความสัมพันธ์ของจำนวน

พารามิเตอร์ที่เหมือนกันของพารามิเตอร์ส่งออกของเว็บเซอร์วิสตำแหน่งที่กำลังสำรวจเทียบกับพารามิเตอร์ที่ต้องการทั้งหมดของการร้องขอก็สามารถใช้สำหรับการประมาณต้นทุนจากเว็บเซอร์วิสปัจจุบันไปยังเว็บเซอร์วิสคำตอบได้เช่นกัน

การใช้ขั้นตอนวิธีการค้นหาแบบ A* เพื่อค้นหาเส้นทางในเว็บเซอร์วิสกราฟทำได้ดังต่อไปนี้คือ เมื่อ f(s) = g(s) + h(s) โดยในที่นี้จะนำเสนอ ฟังก์ชันฮิวริสติกซึ่งในที่นี้หมายถึงฟังก์ชัน h โดยจะมี 2 แบบด้วยกันคือ แบบที่ 1 แนวคิดมาจากการพิจารณาจากโครงสร้างของเว็บเซอร์วิสกราฟเป็นหลัก ส่วนแบบที่ 2 จะพิจารณาจากโครงสร้างของเว็บเซอร์วิสกราฟและพิจารณาจากความสัมพันธ์ของพารามิเตอร์ส่งออกของเว็บเซอร์วิสตำแหน่งที่กำลังสำรวจเทียบกับพารามิเตอร์ที่ต้องการทั้งหมดของการร้องขอ โดยกำหนดให้

g(s) คือค่าต้นทุนของระยะทางจากเว็บเซอร์วิสเริ่มต้นมายังเว็บเซอร์วิสปัจจุบันโดยกำหนดให้ต้นทุนของความสัมพันธ์ระหว่างเว็บเซอร์วิสหนึ่งกับอีกเว็บเซอร์วิสหนึ่งมีค่าเท่ากับ 1 หน่วย เพื่อในการคำนวณต้นทุนตัวเลขที่ได้จะแสดงถึงระยะห่างจากเว็บเซอร์วิสเริ่มต้น

3.1.1 ขั้นตอนวิธีการค้นหาแบบ A* โดยใช้ฟังก์ชัน h แบบที่ 1

กำหนดให้ $h(s) = \frac{1}{1+|S_{link} \cap G|}$ เมื่อ

s คือเว็บเซอร์วิสปัจจุบันที่ทำการสำรวจ

S_{link} คือเซ็ตเว็บเซอร์วิสที่มีความสัมพันธ์กับเซอร์วิสปัจจุบัน หรือเส้นทางที่จะไปต่อได้จากเว็บเซอร์วิสปัจจุบัน ซึ่งอยู่ในสคมภ์ LINK ในตารางดัชนี

G คือเว็บเซอร์วิสสุดท้ายที่ต้องเรียกใช้งานในเว็บเซอร์วิสกราฟ

ถ้าสมาชิกในเซ็ต S_{link} มีสมาชิกเหมือนกับสมาชิกในเซ็ต G มากค่าของ $|S_{link} \cap G|$ ก็จะมากขึ้นตามซึ่งแสดงว่าเส้นทางนี้มีโอกาสไปถึงคำตอบมากขึ้นด้วย ซึ่งเมื่อคำนวณจาก $\frac{1}{1+|S_{link} \cap G|}$ ค่าที่คำนวณได้ก็จะเป็นค่าน้อยๆซึ่งแสดงถึงต้นทุนที่ต่ำแสดงว่าเป็นค่าที่ดีเหมาะต่อการเลือก ตามหลักการของขั้นตอนวิธีการค้นหาแบบ A*

เมื่อพิจารณาฟังก์ชัน h ที่ $|S_{link} \cap G|$ | จะทำให้รู้ล่วงหน้าได้ก่อน ถ้าพบว่าเว็บเซอร์วิสที่มี link ไปถึงคำตอบได้ ก็ไปเส้นทางนั้นเลยโดยไม่ต้องพิจารณาเส้นทางอื่น หรือสำรวจเว็บเซอร์วิสอื่นต่อไป ซึ่งจะทำให้ขอบเขตหรือพื้นที่ในการค้นหาในเว็บเซอร์วิสกราฟลดลง และที่ต้องนำ 1 ไปบวกเนื่องจากมีกรณีที่ผลการคำนวณจาก $|S_{link} \cap G|$ มีค่าเป็น 0 ทำให้เกิดการหารด้วยศูนย์เกิดขึ้นแล้วก็จะทำให้ค่า h เป็น infinity ในส่วนที่ต้องเป็น $\frac{1}{1+|S_{link} \cap G|}$ เนื่องจากต้องการให้ค่าที่คำนวณได้จากฟังก์ชัน h ที่มีโอกาสไปถึงคำตอบมีค่าน้อยๆ กล่าวคือถ้า S_{link} ไปถึงคำตอบได้นั้นค่า $|S_{link} \cap G|$ จะมากกว่า 1 ถ้ากรณีที่ไม่มีความสัมพันธ์กันก็จะเป็น 0 ดังนั้นค่า h ที่คำนวณได้ใน

กรณี ที่ S_{link} ไปถึงคำตอบได้นั้นค่าจะไม่เกิน 0.5 ส่วนในกรณีที่ไม่มีความสัมพันธ์กันค่า h ก็จะเป็น 1 จึงทำให้เลือกไปเส้นทางที่ไปถึงสถานะคำตอบได้ก่อนนั่นเอง

3.1.2 ขั้นตอนวิธีการค้นหาแบบ A* โดยใช้ฟังก์ชัน h แบบที่ 2

ฟังก์ชัน h แบบที่ 2 จะพิจารณาจากโครงสร้างของเว็บเซอร์วิสกราฟและพิจารณาจากความสัมพันธ์ของพารามิเตอร์ส่งออกของเว็บเซอร์วิสตำแหน่งที่กำลังสำรวจเทียบกับพารามิเตอร์ที่ต้องการทั้งหมดของการร้องขอ โดยกำหนดให้

$$h(s) = \frac{1}{1+|S_{link} \cap G| + |(R-T) \cap S_{out}|}$$ โดย

S_{link} คือเซ็ตเว็บเซอร์วิสที่มีความสัมพันธ์กับเซอร์วิสปัจจุบัน หรือเส้นทางที่จะไปต่อได้จากเว็บเซอร์วิสปัจจุบัน ซึ่งอยู่ในสดมภ์ LINK ในตารางดัชนี

G คือ เซ็ตเว็บเซอร์วิสสุดท้ายที่ต้องเรียกใช้งานในเว็บเซอร์วิสกราฟ หรือเว็บเซอร์วิสคำตอบ

R คือ เซ็ตของพารามิเตอร์ทั้งหมดที่ต้องการจากการร้องขอนั้นๆ

T คือ เซ็ตของพารามิเตอร์ส่งออกของเว็บเซอร์วิสที่ได้ทำการสำรวจไปแล้วก่อนเว็บเซอร์วิส s ซึ่งคือเว็บเซอร์วิสปัจจุบันที่ทำการสำรวจ

S_{out} คือ เซ็ตของพารามิเตอร์ส่งออกของเว็บเซอร์วิส s ซึ่งคือเว็บเซอร์วิสปัจจุบันที่ทำการสำรวจ

ในส่วนของการคำนวณค่า $|S_{link} \cap G|$ ได้กล่าวไปแล้วในหัวข้อ 3.1.1

$R-T$ หาเพื่อต้องการนำพารามิเตอร์ที่ยังไม่พบมาพิจารณาเท่านั้น ส่วนพารามิเตอร์ที่พบแล้วจะไม่นำมาพิจารณาอีก เมื่อคำนวณค่า $|(R-T) \cap S_{out}|$ คือการนำพารามิเตอร์ส่งออกของเว็บเซอร์วิสปัจจุบันมาพิจารณากับพารามิเตอร์ของการร้องขอนั้นที่ยังหาไม่พบเท่านั้น ซึ่งถ้ามีค่ามากแสดงว่าเส้นทางนั้นๆมีโอกาสได้คำตอบมาก โดยในที่นี้จะพิจารณาจากชื่อของพารามิเตอร์เป็นหลัก เช่น พารามิเตอร์ที่ต้องการทั้งหมดคือ R={A, B, C, D} พารามิเตอร์ส่งออกของเว็บเซอร์วิสที่ได้ทำการสำรวจไปแล้วคือ T={X, Y, Z, D} และ พารามิเตอร์ส่งออกของเว็บเซอร์วิสปัจจุบันคือ S_{out} ={A, E} เมื่อคำนวณในส่วน $R-T$ จะได้ผลลัพธ์คือ {A, B, C} ซึ่งก็คือพารามิเตอร์ที่ยังหาไม่พบ เมื่อนำผลลัพธ์ที่ได้มาคำนวณ $(R-T) \cap S_{out}$ จะเห็นว่าได้ผลลัพธ์คือ {A} ดังนั้นสมาชิกของ T ในตอนนี้คือ {X, Y, Z, D, A} เป็นต้น

จากที่กล่าวไว้ในบทที่ 2 ในหัวข้อที่ 2.4.3 ว่าการที่ขั้นตอนวิธีการค้นหาแบบ A* นั้นจะสามารถให้คำตอบที่ดีที่สุดได้ต้องมีคุณสมบัติสองอย่างด้วยกันคือ Admissible Heuristic และ Monotonic (Consistent Heuristics) ดังนั้นจึงต้องทำการตรวจสอบก่อน

3.1.3 คุณสมบัติ Admissible Heuristic

ขั้นตอนวิธีการค้นหาแบบ A* นี้มีคุณสมบัติ Admissible Heuristic หรือไม่ จากที่ได้กล่าวไปแล้วในหัวข้อที่ 2.4.1 สิ่งที่สำคัญในการพิจารณาของคุณสมบัตินี้คือ $0 \leq h(s) \leq h^*(s)$ โดยที่ $h^*(s)$ คือระยะทางจากเว็บเซอร์วิสปัจจุบันถึงเว็บเซอร์วิสคำตอบ ถ้าพิจารณาในตำแหน่งที่เว็บเซอร์วิสนั้นเป็นเว็บเซอร์วิสที่มี link ไปสู่คำตอบจะมีค่า h ในแบบที่ 1 คือ

$$h(s) = \frac{1}{1+|s_{link} \cap G|}$$

$$h(s) = \frac{1}{1+1}$$

$$h(s) = 0.5$$

จากข้างต้นถ้า $|s_{link} \cap G|$ แล้วมีจำนวนมากกว่า 1 ค่า h ที่คำนวณก็จะมีค่าน้อยกว่า 0.5 และถ้าในกรณีที่ไม่มีจำนวนสมาชิกที่เหมือนกันเลยซึ่งจะทำให้ h จะมีค่าดังนี้คือ

$$h(s) = \frac{1}{1+|s_{link} \cap G|}$$

$$h(s) = \frac{1}{1+0}$$

$$h(s) = 1$$

และ $h^*(s)$ จะมีค่ามากกว่าหรือเท่ากับ 1 เนื่องจากระยะทางที่สั้นที่สุดที่ไปหาคำตอบคือ 1 ดังนั้น $0 \leq h(s) \leq h^*(s)$ ในทุกๆตำแหน่งของเว็บเซอร์วิสในเว็บเซอร์วิสกราฟ แสดงว่า h แบบที่ 1 Admissible

ในส่วนของ h แบบที่ 2 $|(R-T) \cap s_{out}|$ มีค่าเป็น 0 ค่าก็จะเป็นไปได้ตาม h แบบที่ 1 ตามที่ได้กล่าวมาข้างต้น แต่ถ้าไม่เป็น 0 จะทำให้ค่า h ที่คำนวณได้จะน้อยกว่าแบบที่ 1 แต่จะมีค่ามากกว่า 0 และไม่เกิน 1 เสมอ และจากที่ได้กล่าวแล้วว่า $h^*(s)$ จะมีค่ามากกว่าหรือเท่ากับ 1 เนื่องจากระยะทางที่สั้นที่สุดที่ไปหาคำตอบก็คือ 1 ดังนั้น $0 \leq h(s) \leq h^*(s)$ ในทุกๆตำแหน่งของเว็บเซอร์วิสในเว็บเซอร์วิสกราฟ แสดงว่า h แบบที่ 2 Admissible

เพราะฉะนั้นขั้นตอนวิธีการค้นหาแบบA* ที่ผู้วิจัยได้นำเสนอโดยใช้ฟังก์ชัน h ทั้ง 2 แบบมีคุณสมบัติ Admissible Heuristic

3.1.4 คุณสมบัติ Monotonic

คุณสมบัติ Monotonic จากที่ได้กล่าวไปในหัวข้อที่ 2.4.2 การพิจารณาคุณสมบัตินี้จะทำการพิจารณาว่า ค่าที่คำนวณได้จากฟังก์ชัน f(s) ตลอดเส้นทางในแต่ละเส้นทางจะต้องไม่ลดลง

จาก $f(s) = g(s) + h(s)$ ซึ่งถ้าพิจารณาฟังก์ชัน h ทั้ง 2 แบบที่ผู้วิจัยได้นำเสนอจะมีค่าระหว่าง $0 < h(s) \leq 1$ และจากที่ฟังก์ชัน h ทั้ง 2 แบบคุณสมบัติ Admissible Heuristic ทำให้การสำรวจกราฟจะไม่หลงไปเส้นทางอื่นที่ไม่ใช่เส้นทางที่ดีที่สุด ส่วนค่าของฟังก์ชัน gในแต่ละ

เส้นทางจะถูกเพิ่มขึ้นทีละ 1 หน่วยตลอดเส้นทาง ดังนั้น ค่า f ของเว็บเซอร์วิสในตำแหน่งปัจจุบันจะไม่สามารถมีค่าน้อยกว่าเว็บเซอร์วิส ในลำดับก่อนหน้าของเส้นทางนั้นๆ ได้เลย ดังนั้นเว็บเซอร์วิสปัจจุบัน f(s) จะมีค่าน้อยกว่าสถานะถัดไปเสมอ เพราะฉะนั้นขั้นตอนวิธีการค้นหาแบบ A* ที่ผู้วิจัยนำเสนอมีคุณสมบัติ Monotonic

จากหัวข้อที่ 3.1.3 และ 3.1.4 สรุปได้ว่าขั้นตอนวิธีการค้นหาแบบ A* ที่ผู้วิจัยนำเสนอ มีคุณสมบัติ Admissible Heuristic และ Monotonic เพราะฉะนั้นแสดงว่า ขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h ทั้ง 2 แบบให้คำตอบที่ดีที่สุดได้เช่นเดียวกับขั้นตอนวิธีค้นหาแบบแนวกว้างก่อน

3.1.5 การนำขั้นตอนวิธีการค้นหาแบบ A* ไปใช้งาน

ในการนำไปใช้งานกับเว็บเซอร์วิสกราฟนั้น ได้นำเสนอตามรูปที่ 3.1 และ 3.3 โดยในรูปที่ 3.1 จะแสดงขั้นตอนวิธีสำหรับขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 1 และรูปที่ 3.3 จะแสดงขั้นตอนวิธีสำหรับขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 2

```
1: Function : AstarWSG1()
2: Input : WSG
3:Output :visited web services (R)
4: w ← first web service in WSG, T ← T ∪ w
5: While( T not empty) {
6:       S ← T
7:       for ( each w in S){
8:             R ← R ∪ w
9:             If (h(s) <1) // similar to replace by If (f(s) not integer)
10:                  R ← R ∪ G
11:                  return R and Exit all loop
12:      }
13:    T ← T ∪ S_link
14:    T ← T –R
15: }
```

รูปที่ 3.1 ขั้นตอนวิธีค้นหาแบบ A* โดยใช้ h แบบที่ 1

ตัวแปรต่างๆในรูปที่ 3.1 อธิบายได้ดังนี้

R คือเซตของเว็บเซอร์วิสที่ผ่านการสำรวจในการค้นหาเส้นทางที่ดีที่สุดจากเว็บเซอร์วิสกราฟนั้นๆ

W คือเว็บเซอร์วิสปัจจุบันที่ทำการสำรวจ

WSG คือเว็บเซอร์วิสกราฟ

T คือเซ็ตของเว็บเซอร์วิสที่เตรียมสำหรับการสำรวจต้นทุนในวงวน (loop) ถัดไป

S คือเซ็ตเว็บเซอร์วิสสำหรับการสำรวจต้นทุนในวงวนปัจจุบัน

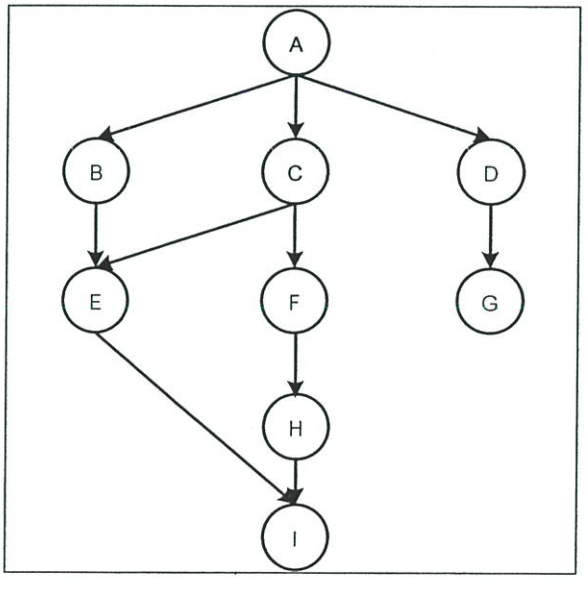
S_{link} คือเซ็ตของเว็บเซอร์วิสที่มีความสัมพันธ์กับสาชิกในเซ็ต S ซึ่งอยู่ในสดมภ์ LINK

G คือเว็บเซอร์วิสสุดท้ายที่ต้องเรียกใช้งานในเว็บเซอร์วิสกราฟ หรือเว็บเซอร์วิสคำตอบ

← คือเครื่องหมายกำหนดค่า

โดยในที่นี้ค่าเริ่มต้นจะกำหนดในบรรทัดที่ 4 โดยกำหนดให้ W คือเว็บเซอร์วิสแรกในตารางดัชนีหรือเว็บเซอร์วิสแรกในกราฟนั้นเอง และ T เริ่มแรกให้มีสมาชิกตัวเดียวคือเว็บเซอร์วิสแรกในตารางดัชนีของเว็บเซอร์วิสกราฟนั้นๆ

จากรูปที่ 3.1 เมื่อพิจารณาที่ในบรรทัดที่ 9 จะเห็นว่าไม่ได้ทำการหาค่าต้นทุนน้อยสุดตามหลักการวิธีการของขั้นตอนวิธีการค้นหาแบบ A* เนื่องจากในที่นี้จะได้ผลลัพธ์เท่ากับการหาค่าน้อยสุดแต่สะดวก เพราะว่าในที่นี่ค่า h แบบที่ 1 จะส่งผลให้การตรวจสอบเว็บเซอร์วิสในเว็บเซอร์วิสกราฟเป็นไปทีละระดับ ซึ่งสามารถพิจารณาเพียงค่า h ก็เพียงพอ



รูปที่ 3.2 เว็บเซอร์วิสกราฟ

จากขั้นตอนวิธีในรูปที่ 3.1 ถ้าทำการค้นหาเส้นทางที่ดีที่สุดจากเว็บเซอร์วิสกราฟจากรูปที่ 3.2 ซึ่งในที่นี้คือ input โดยแต่ละโหนดคือเว็บเซอร์วิส จะได้เส้นทางที่ดีที่สุดคือ A-> B -> E-> I และ output คือเซ็ตของเว็บเซอร์วิสที่ต้องเข้าไปสำรวจทั้งหมดในการสำรวจครั้งนี้คือ R= {A, B, C,

D, E, I} ซึ่งถ้าใช้ขั้นตอนวิธีการค้นหาแบบแนวกว้างก่อนจะได้เส้นทางที่ดีที่สุดเส้นทางที่ดีที่สุดเช่นกัน คือ A-> B -> E-> I แต่เว็บเซอร์วิสที่เข้าไปสำรวจคือ {A, B, C, D, E, F, G, I } ซึ่งจะเห็นว่ามีจำนวนมากกว่า เนื่องจากขั้นตอนวิธีการค้นหาแบบ A* โดยใช้ฟังก์ชัน h แบบที่ 1 ในระดับชั้นก่อนหน้าคำตอบ 1 ระดับชั้น ค่า h ที่คำนวณได้จะมีค่าไม่เกิน 0.5 จึงทำให้เข้าไปทำการสำรวจเพียงแค่ 1 เว็บเซอร์วิสแล้วไปยังคำตอบได้เลย ส่งผลทำพื้นที่หรือสำนวนเว็บเซอร์วิสที่สำรวจมีจำนวนน้อยกว่าขั้นตอนวิธีการค้นหาแบบแนวกว้างก่อนประมาณ 1 ระดับชั้นในการค้นหา เนื่องจากเนื่องจากถ้าเว็บเซอร์วิสที่เป็นคำตอบอยู่ในระดับชั้นที่ n การค้นหาในระดับชั้นที่ n-1 โดยใช้ขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 1 จะค้นเพียงเว็บเซอร์วิสเดียวแล้วสามารถนำไปสู่เว็บเซอร์วิสที่เป็นคำตอบได้ทันที

รูปที่ 3.3 แสดงขั้นตอนวิธีสำหรับขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 2 โดยตัวแปรต่างๆมีดังนี้คือ

R คือเซตของเว็บเซอร์วิสที่ผ่านการสำรวจในการค้นหาเส้นทางที่ดีที่สุดจากเว็บเซอร์วิสกราฟนั้นๆ

W คือเว็บเซอร์วิสปัจจุบันที่ทำการสำรวจ

WSG คือเว็บเซอร์วิสกราฟ

T คือเซ็ตของเว็บเซอร์วิสที่เตรียมสำหรับการสำรวจต้นทุนในวงวน (loop) ถัดไป

S คือเซ็ตเว็บเซอร์วิสสำหรับการสำรวจต้นทุนในวงวนปัจจุบัน

S_{link} คือเซ็ตของเว็บเซอร์วิสที่มีความสัมพันธ์กับสาชิกในเซ็ต S ซึ่งอยู่ในสดมภ์ LINK

G คือเว็บเซอร์วิสสุดท้ายที่ต้องเรียกใช้งานในเว็บเซอร์วิสกราฟ

min คือเว็บเซอร์วิสที่เมื่อใช้เส้นทางจาเว็บเซอร์วิสปัจจุบันมาเว็บเซอร์วิสนี้จะมีค่าของฟังก์ชัน f น้อยสุดในขณะนั้น

min_{link} คือเซ็ตของเว็บเซอร์วิสที่มีความสัมพันธ์กับเว็บเซอร์วิส min ซึ่งอยู่ในสดมภ์ LINK ในตารางดัชนี

CHK คือตัวแปรสำหรับตรวจสอบว่าในระดับชั้นนั้นมีเว็บเซอร์วิสที่มีต้นทุนน้อยที่สุดหรือไม่

← คือเครื่องหมายกำหนดค่า

โดยในที่นี้ค่าเริ่มต้นจะกำหนดในบรรทัดที่ 4 โดยกำหนดให้ W คือเว็บเซอร์วิสแรกในตารางดัชนีหรือเว็บเซอร์วิสแรกในกราฟนั้นเอง ให้ T เริ่มแรกให้มีสมาชิกตัวเดียวคือเว็บเซอร์วิสแรกในตารางดัชนีของเว็บเซอร์วิสกราฟนั้นๆ กำหนดให้ min มีค่าเริ่มต้นคือเว็บเซอร์วิสแรกในเว็บเซอร์วิสกราฟ และ R ให้มีสมาชิก 1 ตัวคือเว็บเซอร์วิสแรกในกราฟ และกำหนดให้มีค่า CHK เพื่อตรวจสอบว่าในระดับชั้นนั้นมีเว็บเซอร์วิสที่มีต้นทุนน้อยที่สุดหรือไม่ ถ้าไม่มีก็จะกระจายการค้นลงมาทั้งระดับชั้น แต่ถ้ามีเว็บเซอร์วิสที่มีต้นทุนน้อยสุดจะกระจายการค้นลงมาเฉพาะเว็บเซอร์วิสที่มีต้นทุนน้อยที่สุดเท่านั้น

ในขั้นตอนวิธีในรูปที่ 3.3 ถ้าทำการค้นหาเส้นทางที่ดีที่สุดจากเว็บเซอร์วิสกราฟในรูปที่ 3.2 ซึ่งในที่นี้คือ Input โดยแต่ละโหนดคือเว็บเซอร์วิส จะได้เส้นทางที่ดีที่สุดคือ A-> B -> E-> I เช่นเดียวกับขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ h แบบที่ 1 แต่เว็บเซอร์วิสที่เข้าไปสำรวจมีโอกาสที่จะมีจำนวนน้อยกว่า

```
1: Function : AstarWSG2()
2: Input : WSG
3:Output :visited web services (R)
4: w⟵ first web service in WSG, T ⟵ T⋃ w, min⟵w, R ⟵ R ⋃w , CHK ⟵False
5: While( Σ not empty){
6:          S ⟵ T
7:           for ( each w in S){
8:                 If(f(min)>f(w)) { min⟵w }
10:                  Else { CHK ⟵ True , R ⟵ R ⋃ w}
11:          }
12:         If(CHK) { T ⟵ T ⋃ S_link }
13:          Else { T ⟵ T ⋃ min_link , R ⟵ R⋃ min }
13: T ⟵ T –R
14: CHK ⟵False
15: }
```

รูปที่ 3.3 ขั้นตอนวิธีค้นหาแบบ A* โดยใช้ h แบบที่ 2

ขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ h แบบที่ 2 ในรูปที่ 3.3 จะมีเว็บเซอร์วิสที่เข้าไปสำรวจมีโอกาสที่จะมีจำนวนน้อยกว่าขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ h แบบที่ 1 เนื่องจากมีการนำพารามิเตอร์มาพิจารณาร่วมด้วย แต่จะลดพื้นที่ในการค้นหาเพิ่มขึ้นได้มากน้อยเท่าไหร่นั้น ขึ้นอยู่กับความสัมพันธ์ของพารามิเตอร์ส่งออกของเว็บเซอร์วิสนั้นกับพารามิเตอร์ที่ต้องการทั้งหมดของการร้องขอซึ่งกรณีที่แย่ที่สุดก็คือไม่มีความสัมพันธ์ระหว่างพารามิเตอร์เลยจนถึงระดับที่ชั้นก่อนถึงคำตอบซึ่งก็คือในเว็บเซอร์วิสก่อนหน้านี้คำนวณค่า $|(R-T)\cap s_{out}|$ ได้เป็น 0 ตลอด

ดังนั้นกรณีที่แย่ที่สุดของขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ h แบบที่ 2 นี้จะสามารถลดพื้นที่ในการค้นหาลงได้เท่ากับ h แบบ 1 คือลดลงได้ 1 ระดับชั้น แต่ในความเป็นจริงจะเป็นไปได้น้อยมากที่พารามิเตอร์จะไม่มีความสัมพันธ์กันเลยจนถึงระดับชั้นที่ n-1 เมื่อเว็บเซอร์วิสที่เป็น

คำตอบอยู่ในระดับชั้นที่ n ดังนั้นในการทำงานจริงจึงมีโอกาสที่จะสามารถลดพื้นการค้นหาลงได้มากกว่า 1 ระดับชั้น

3.1.6 อัตราการเติบโตของฟังก์ชัน

ในส่วนของอัตราเติบโตของฟังก์ชัน จากที่ได้กล่าวไปในหัวข้อที่ 2.3 ว่าอัตราการเติบโตของฟังก์ชันของขั้นตอนวิธีการค้นหาแบบ A* ในส่วนของความซับซ้อนทางด้านเวลา มีโอกาสเป็นได้ใน 2 กรณีคือ 1) ค่าประมาณเท่ากับขั้นตอนวิธีการค้นหาแบบแนวกว้างก่อน ซึ่งอยู่ในรูปของเอ็กซ์โพเนนเชียล คือมีค่าเท่ากับ $O(b^d)$ หรือประมาณได้เท่ากับ หรือ $O(|E|+|V|)$ และ 2) ในกรณีที่ $|h(s) - h^*(s)| \leq O(\log(h^*(s)))$ เมื่อ $h^*(s)$ คือค่าต้นทุนของเส้นทางที่ดีที่สุดจากเว็บเซอร์วิสปัจจุบันไปถึงเว็บเซอร์วิสสุดท้ายหรือเว็บเซอร์วิสคำตอบ อัตราการเติบโตของฟังก์ชันจะอยู่ในรูปของอยู่ในรูปพหุนาม

ซึ่งในขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h ทั้ง 2 แบบ ที่ได้นำเสนอจะอยู่ในรูปแบบของเอ็กซ์โพเนนเชียล คือมีค่าเท่ากับ $O(b^d)$ โดยสามารถพิสูจน์แบบข้อขัดแย้งได้ดังนี้

พิสูจน์ขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 1

สมมติให้ขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 1 มีอัตราการเติบโตของฟังก์ชันอยู่ในรูปของอยู่ในรูปพหุนาม ซึ่งเป็นไปตามเงื่อนไข $|h(s) - h^*(s)| \leq O(\log(h^*(s)))$ สำหรับทุกๆเว็บเซอร์วิสในเว็บเซอร์วิสกราฟ

โดยกำหนดให้ n คือจำนวนเว็บเซอร์วิสในเว็บเซอร์วิสกราฟ s คือเว็บเซอร์วิสตำแหน่งปัจจุบัน และ $h^*(s)$ คือค่าต้นทุนของเส้นทางที่ดีที่สุดจากเว็บเซอร์วิสปัจจุบันไปถึงเว็บเซอร์วิสสุดท้าย

เมื่อพิจารณาในกรณีที่เว็บเซอร์วิสกราฟมีเส้นทางทุกเส้นทางมีระยะทางเท่ากันโดยแต่ละเส้น- ทางจะไม่แตกกิ่งเลยตั้งแต่เว็บเซอร์วิสแรกจนถึงเว็บเซอร์วิสเส้นทางเช่นในกรณีที่เว็บเซอร์วิสกราฟมีเส้นทางอยู่ 2 เส้นทางที่ไปยังคำตอบ โดยแต่ละเส้นทางมีความลึกเท่ากันและมีความลึกอย่างน้อย 3 หน่วย ถ้าพิจารณาที่ตำแหน่งเริ่มต้น $h^*(s)$ จะมีค่า $\frac{n}{2}$ และค่า h(s) จะมีค่าเป็น 1 ดังนั้นจาก $|h(s) - h^*(s)| \leq O(\log(h^*(s)))$ จะได้

$$|1 - \frac{n}{2}| \leq O(\log(\frac{n}{2}))$$

$$|1 - \frac{n}{2}| \leq O(\log(n)+\log(1/2))$$

$$O(|1|)-O(|\frac{n}{2}|) \leq O(\max(\log(n)+\log(1/2)))$$

$$O(1)-O(\frac{n}{2}) \leq O(\log(n))$$

$$\max (O(1)-O(\frac{n}{2})) \leq O(\log(n))$$

$$-O(\frac{n}{2}) \leq O(\log(n))$$

$$-\frac{1}{2}O(n) \leq O(\log(n))$$

$$O(n) \leq O(\log(n))$$

จากข้างต้นพบว่าเกิดข้อขัดแย้งขึ้นในกรณีนี้ เนื่องจาก $O(n) \leq O(\log(n))$ เป็นเท็จ เพราะฉะนั้นขั้นตอนวิธีการค้นหาแบบ A* โดยการใช้ฟังก์ชัน h แบบที่ 1 ความซับซ้อนทางด้านเวลาจะอยู่ในรูปของเอ็กซ์โพเนนเชียล คือ $O(b^d)$

พิสูจน์ขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 2

สมมติให้ขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 2 มีอัตราการเติบโตของฟังก์ชันอยู่ในรูปของอยู่ในรูปพหุนาม ซึ่งเป็นไปตามเงื่อนไข $|h(s) - h^*(s)| \leq O(\log(h^*(s)))$ สำหรับทุกๆเว็บเซอร์วิสในเว็บเซอร์วิสกราฟ

โดยกำหนดให้ n คือจำนวนเว็บเซอร์วิสในเว็บเซอร์วิสกราฟ s คือเว็บเซอร์วิสตำแหน่งปัจจุบัน และ $h^*(s)$ คือค่าต้นทุนของเส้นทางที่ดีที่สุดจากเว็บเซอร์วิสปัจจุบันไปถึงเว็บเซอร์วิสสุดท้าย

ในที่นี้จะพิจารณากรณีเดียวกับการพิสูจน์ขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 1 คือ ในกรณีที่เมื่อพิจารณาในกรณีที่เว็บเซอร์วิสกราฟมีเส้นทางทุกเส้นทางมีระยะทางเท่ากันโดยแต่ละเส้นทางจะไม่แตกกิ่งเลยตั้งแต่เว็บเซอร์วิสแรกจนถึงเว็บเซอร์วิสเส้นทาง เช่นในกรณีที่เว็บเซอร์วิสกราฟมีเส้นทางอยู่ 2 เส้นทางที่ไปยังคำตอบ โดยแต่ละเส้นทางมีความลึกเท่ากันและมีความลึกอย่างน้อย 3 หน่วย ถ้าพิจารณาที่ตำแหน่งเริ่มต้น $h^*(s)$ จะมีค่า $\frac{n}{2}$ ส่วนค่าที่คำนวนได้ของฟังก์ชัน h แบบที่ 2 จะมีค่า $0 < h(s) \leq 1$ แต่ในที่นี้สมมติว่าเป็นกรณีที่ $|(R-T) \cap S_{out}|$ มีค่าเป็น 0 ซึ่งจะทำให้ค่าของฟังก์ชัน h แบบที่ 2 มีค่าเป็น 1 ดังนั้นจาก $|h(s) - h^*(s)| \leq O(\log(h^*(s)))$ จะได้ผลลัพธ์เช่นเดียวกับการพิสูจน์ขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 1 คือได้ $O(n) \leq O(\log(n))$ ซึ่งเป็นเท็จ

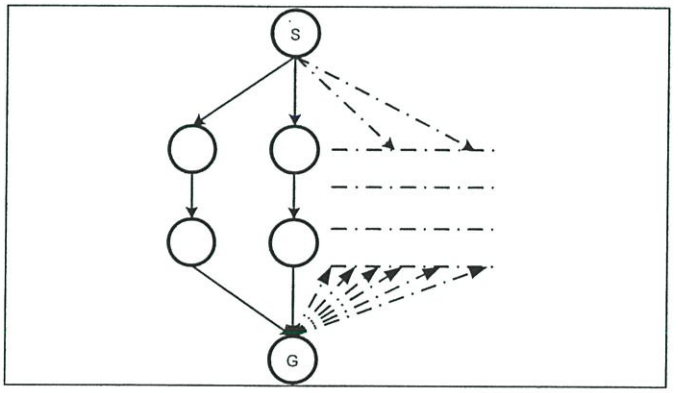
จากข้างต้นเกิดข้อขัดแย้งขึ้นเนื่องจาก $O(n) \leq O(\log(n))$ เป็นเท็จ เพราะฉะนั้นขั้นตอนวิธีการค้นหาแบบ A* โดยใช้ฟังก์ชัน h แบบที่ 2 ความซับซ้อนทางด้านเวลาจะอยู่ในรูปของเอ็กซ์โพเนนเชียล คือ $O(b^d)$ เช่นเดียวกับ ขั้นตอนวิธีการค้นหาแบบ A* โดยใช้ฟังก์ชัน h แบบที่ 1

ดังนั้นตามทฤษฎีสรุปได้ว่าอัตราการเติบโตของฟังก์ชันในส่วนของความซับซ้อนทางด้านเวลาจะอยู่ในรูปของเอ็กซ์โพเนนเชียลคือ $O(b^d)$ เท่ากันทั้งขั้นตอนวิธีการค้นหาแบบ A* ที่การใช้ฟังก์ชัน h แบบที่ 1 และที่ใช้ฟังก์ชัน h แบบที่ 2 แต่ในการทำงานจริง การคำนวณฟังก์ชัน h แบบที่

2 จะต้องเข้าไปตรวจสอบพารามิเตอร์ของแต่ละเว็บเซอร์วิสซึ่งจะอยู่ในยูดีดีไอ ซึ่งจะต้องใช้เวลาในการเข้าไปค้นข้อมูลในระบบเครือข่ายด้วย

ในส่วนของความซับซ้อนทางด้านพื้นที่มีอัตราการเติบโตของฟังก์ชันของขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 1 คือ $O(b^{d-1})$ เนื่องจากถ้าเว็บเซอร์วิสที่เป็นคำตอบอยู่ในระดับชั้นที่ n การค้นหาในระดับชั้นที่ n-1 โดยใช้ขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 1 จะค้นเพียงเว็บเซอร์วิสเดียวแล้วสามารถนำไปสู่เว็บเซอร์วิสที่เป็นคำตอบได้ทันที ทำให้สามารถลดพื้นที่ในการค้นหาเส้นทางในเว็บเซอร์วิสกราฟลงได้ประมาณ 1 ระดับชั้น

ถ้าใช้ขั้นตอนวิธีการค้นหาแบบ A* ใช้ฟังก์ชัน h แบบที่ 2 ถึงแม้ว่าอัตราการเติบโตของฟังก์ชันในกรณีที่แย่ที่สุดคือ คือ $O(b^{d-1})$ เท่ากับการฟังก์ชัน h แบบที่ 1 ในกรณีที่การหาความสัมพันธ์ของพารามิเตอร์จาก $|(R-T)\cap S_{out}|$ มีค่าเป็น 0 ตลอดตั้งแต่จุดเริ่มต้นจนถึงระดับชั้นที่ n-1 เมื่อเว็บเซอร์วิสที่เป็นคำตอบอยู่ในระดับชั้นที่ n แต่ในกรณีส่วนใหญ่จะสามารถลดพื้นที่ในการค้นหาลงได้เพิ่มขึ้นมากกว่า 1 ระดับชั้น ทั้งนี้จะขึ้นอยู่กับความสัมพันธ์ของพารามิเตอร์ส่งออกของเว็บเซอร์วิสในเว็บเซอร์วิสกราฟ ซึ่งจะเป็นไปได้น้อยมากที่จะไม่พบพารามิเตอร์ที่ต้องการเลยในระดับชั้นก่อนหน้าระดับชั้น n-1



รูปที่ 3.4 เว็บเซอร์วิสกราฟที่จำนวนเว็บเซอร์วิสมากและมีความกว้างมาก

3.2 สรุปการปรับปรุงโครงสร้างข้อมูลแบบเว็บเซอร์วิสกราฟ

ในบทนี้ได้แสดงถึงวิธีการปรับปรุงโครงสร้างข้อมูลแบบเว็บเซอร์วิสกราฟในขั้นตอนการค้นหาเส้นทางในเว็บเซอร์วิสกราฟโดยนำขั้นตอนวิธีการค้นหาแบบ A* มาใช้ โดยนำเสนอฟังก์ชัน h อยู่ 2 แบบ ซึ่งแบบที่ 1 แนวคิดมาจากการพิจารณาจากโครงสร้างของเว็บเซอร์วิสกราฟเป็นหลัก ส่วนแบบที่ 2 จะพิจารณาจากโครงสร้างของเว็บเซอร์วิสกราฟและพิจารณาจากความสัมพันธ์ของพารามิเตอร์ส่งออกของเว็บเซอร์วิสตำแหน่งที่กำลังสำรวจเทียบกับพารามิเตอร์ที่ต้องการทั้งหมด

ของการร้องขอ ซึ่งพบว่าอัตราการเติบโตของฟังก์ชัน ในส่วนของความซับซ้อนทางด้านเวลาของขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 1 และแบบที่ 2 มีค่า $O(b^d)$ เท่ากันในทางทฤษฎี แต่ในทางปฏิบัติจะการใช้ฟังก์ชัน h แบบที่ 2 จะใช้เวลามากกว่าเนื่องจากการเข้าไปค้นข้อมูลในระบบเครือข่าย และในส่วนของความซับซ้อนทางด้านพื้นที่ของขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 1 และฟังก์ชัน h แบบที่ 2 คือ $O(b^{d-1})$ ซึ่งสามารถลดพื้นที่ในการค้นหาเส้นทางในเว็บเซอร์วิสกราฟลงได้ประมาณ 1 ระดับชั้นเนื่องจากถ้าเว็บเซอร์วิสที่เป็นคำตอบอยู่ในระดับชั้นที่ n การค้นหาในระดับชั้นที่ n-1 จะค้นเพียงเว็บเซอร์วิสเดียวแล้วสามารถนำไปสู่เว็บเซอร์วิสที่เป็นคำตอบได้ทันที และเมื่อเว็บเซอร์วิสในเว็บเซอร์วิสกราฟในระดับชั้นที่ n-1 มีจำนวนมากก็จะเห็นความแตกต่างมากยิ่งขึ้น แต่สำหรับขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 2 จะสามารถลดพื้นในการค้นหาได้เพิ่มมากขึ้น ทั้งนี้จะขึ้นอยู่กับรูปแบบความสัมพันธ์ของพารามิเตอร์ของเว็บเซอร์วิสในเว็บเซอร์วิสกราฟ

ในบทที่ 4 จะนำเสนอถึงการทดลองและผลการทดลองเพื่อแสดงให้เห็นถึงความแตกต่างระหว่างพื้นที่ในการค้นหาเส้นทางในเว็บเซอร์วิสกราฟของขั้นตอนวิธีการค้นหาแบบ A* โดยใช้ฟังก์ชัน h ในแบบที่ 1 และขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h ในแบบที่ 2 ในรูปแบบที่ให้เห็นถึงจำนวนที่ลดได้จริง และจากข้างต้นการใช้ฟังก์ชัน h ในแบบที่ 2 สามารถที่จะลดพื้นที่การหาได้เพิ่มขึ้น แต่เวลาที่สูญเสียไปจากการเข้าไปค้นข้อมูลของเว็บเซอร์วิสที่อยู่ในยูดีดีไอในระบบเครือข่าย จึงได้นำเสนอการทดลองเปรียบเทียบในเรื่องของเวลาที่ใช้จริงระหว่างขั้นตอนวิธีการค้นหาแบบ A* โดยใช้ ฟังก์ชัน h ในแบบที่ 1 และขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h ในแบบที่ 2

บทที่ 4

การทดลองและผลการทดลอง

ในบทที่ 3ได้แสดงถึงวิธีการปรับปรุงโครงสร้างข้อมูลแบบเว็บเซอร์วิสกราฟในขั้นตอนการค้นหาเส้นทางในเว็บเซอร์วิสกราฟโดยนำขั้นตอนวิธีการค้นหาแบบ A* มาใช้ โดยนำเสนอฟังก์ชัน h อยู่ 2 แบบ ซึ่งแบบที่ 1 แนวคิดมาจากการพิจารณาจากโครงสร้างของเว็บเซอร์วิสกราฟเป็นหลัก ส่วนแบบที่ 2 จะพิจารณาจากโครงสร้างของเว็บเซอร์วิสกราฟและพิจารณาจากความสัมพันธ์ของพารามิเตอร์ส่งออกของเว็บเซอร์วิสตำแหน่งที่กำลังสำรวจเทียบกับพารามิเตอร์ที่ต้องการทั้งหมดของการร้องขอ ซึ่งพบว่าอัตราการเติบโตของฟังก์ชัน ในส่วนของความซับซ้อนทางด้านเวลาของขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 1 และแบบที่ 2 มีค่า $O(b^{d})$ เท่ากันในทางทฤษฎี แต่ในทางปฏิบัติจะการใช้ฟังก์ชัน h แบบที่ 2 จะใช้เวลามากกว่าเนื่องจากการเข้าไปค้นข้อมูลในระบบเครือข่าย และในส่วนของความซับซ้อนทางด้านพื้นที่ของขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 1 และฟังก์ชัน h แบบที่ 2 คือ $O(b^{d-1})$ ซึ่งสามารถลดพื้นที่ในการค้นหาเส้นทางในเว็บเซอร์วิสกราฟลงได้ประมาณ 1 ระดับชั้น และถ้าเว็บเซอร์วิสที่เป็นคำตอบอยู่ในระดับชั้นที่ n เมื่อเว็บเซอร์วิสในเว็บเซอร์วิสกราฟในระดับชั้นที่ n-1 มีจำนวนมากก็จะเห็นความแตกต่างมากยิ่งขึ้น และเมื่อเว็บเซอร์วิสในเว็บเซอร์วิสกราฟในระดับชั้นก่อนถึงคำตอบมีจำนวนมากก็จะเห็นความแตกต่างมากยิ่งขึ้น แต่สำหรับขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 2 มีโอกาสที่จะลดพื้นที่ในการค้นหาลงได้มากกว่า 1 ระดับชั้นซึ่งจะขึ้นอยู่กับรูปแบบความสัมพันธ์ของพารามิเตอร์ของเว็บเซอร์วิสในเว็บเซอร์วิสกราฟ

ดังนั้นในบทนี้จะนำเสนอขั้นตอนการทดลอง และผลการทดลอง เพื่อทดลองให้เห็นถึงความแต่งต่างในการทำงานจริงว่าพื้นที่หรือขอบเขตในการค้นหาเส้นทางที่ดีที่สุดในเว็บเซอร์วิสกราฟมีจำนวนเว็บเซอร์วิสที่เข้าไปสำรวจลดลงระหว่างขั้นตอนวิธีการค้นหาแบบ A* โดยใช้ฟังก์ชันh แบบที่ 1 และ ใช้ฟังก์ชัน h แบบที่ 1 โดยทำการเปรียบเทียบพื้นที่ในการค้นหาเส้นทางในเว็บเซอร์วิสกราฟซึ่งหมายถึงจำนวนเว็บเซอร์วิสที่เข้าไปสำรวจในเว็บเซอร์วิสกราฟในการค้นหาเส้นทางที่ดีที่สุดระหว่างการใช้ขั้นตอนวิธีการค้นหาแบบแนวกว้างก่อนในการค้นหาเส้นทางที่ดีที่สุดในเว็บเซอร์วิสกราฟซึ่งเป็นวิธีการเดิม และใช้ขั้นตอนวิธีการค้นหาแบบ A* ตามที่ได้นำเสนอและทดลองให้เห็นถึงความแตกต่างของเวลาในการทำงานจริงว่าระหว่างขั้นตอนวิธีการค้นหาแบบ A* โดยใช้ ฟังก์ชัน h แบบที่ 2 ที่ต้องใช้เวลาในการค้นหาข้อมูลในระบบเครือข่ายนั้นเพิ่มขึ้น กับขั้นตอนวิธีการค้นหาแบบ A* โดยใช้ ฟังก์ชัน h แบบที่ 1

4.1 เครื่องมือที่ใช้ในการทดลอง

1) ส่วนประกอบทางด้านฮาร์ดแวร์

ฮาร์ดแวร์ที่ใช้ทำการทดลองเป็นคอมพิวเตอร์ส่วนบุคคลมีองค์ประกอบดังนี้

ก) หน่วยประมวลผลกลาง (CPU) Intel Dual Core processor 1.8 กิกะเฮิร์ต

ข) หน่วยคำจำหลัก (RAM) 1.0 กิกะไบต์

ค) หน่วยความจำสำรอง (Disk Storage) ขนาด 100 กิกะไบต์

2) ส่วนประกอบทางด้านซอร์ฟแวร์

ก) ระบบปฏิบัติการ Windows XP version7 Professional

ข) Eclipse

ค) Microsoft c++ .NET 2005

ง) GTL plugin

จ) โปรแกรม Graph Magics

ฉ) โปรแกรมตารางการคำนวณ Microsoft Excel 2003

ช) โปรแกรมฐานข้อมูล MySql (เก็บกราฟในรูปตารางดัชนีและจำลองยูดีดีไอ)

4.2 วิธีการทดลอง

ในการทดลองจะทำการเตรียมข้อมูลโดยการสร้างกราฟขึ้นมาให้มีลักษณะเดียวกับเว็บเซอร์วิสกราฟ คือเป็นกราฟแบบไม่มีวงแบบระบุทิศทางโดยใช้ GTL plugin ร่วมกับตัวแปลภาษาซีในการสร้างกราฟโดย และใช้โปแกรม Graph Magics ในการตรวจสอบลักษณะของกราฟอีกครั้ง ซึ่งผลที่ได้จะได้ออกมาในรูปแบบของเมตริกซ์อยู่ในแฟ้มข้อมูลแบบข้อความ (text file) ดังรูปที่ 4.1 โดยในบรรทัดแรกแสดงถึงจำนวนโหนดที่มีอยู่ในกราฟ ในที่มีจำนวน 8 โหนด ในแต่ละบรรทัดจะมีจำนวนกลุ่มของตัวเลขตามจำนวนโหนดในที่นี้จะมีอยู่ 8 ตัว ตัวเลขแต่ละตัวคือโหนดตำแหน่งแรกคือโหนดที่ 1 ถัดไปคือโหนดที่ 2 ตามลำดับ และจะมีอยู่ 8 บรรทัดเท่ากับจำนวนโหนด ซึ่งตัวเลขที่แสดงนั้นจะมีตัวเลขเป็น 0 และ 1 เมื่อ 0 หมายถึงไม่มีเส้นเชื่อม 1 หมายถึงมีเส้นเชื่อม เช่นในบรรทัดที่ 1 หมายความว่า โหนดที่ 1 มีเส้นเชื่อมไปหาโหนดที่ 2 โหนดที่ 3 และโหนดที่ 8 บรรทัดที่ 2 หมายความว่า โหนดที่ 2 มีเส้นเชื่อมไปหาโหนดที่ 5 เป็นต้น แต่เมื่อสังเกตในบรรทัด 7 จะพบว่าตัวเลขทุกตำแหน่งเป็น 0 ซึ่งความว่าโหนดที่ 7 ไม่มีเส้นเชื่อมไปยังโหนดใดเลย ซึ่งแสดงว่าโหนดที่ 7 คือโหนดสุดท้ายในกราฟ

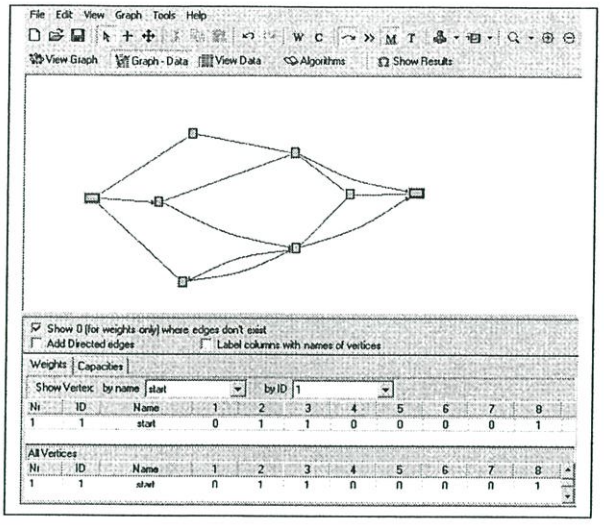
```
8
0 1 1 0 0 0 0 1
0 0 0 0 1 0 0 0
0 0 0 1 0 0 0 0
0 0 1 0 0 0 1 1
0 0 0 0 0 0 1 1
0 0 0 0 0 0 1 0
0 0 0 0 0 0 0 0
0 0 0 1 1 0 0 0 0
```

รูปที่ 4.1 กราฟที่อยู่ในรูปแฟ้มข้อมูลแบบข้อความ

ในการทดลองจะสร้างกราฟแบบเครือข่ายขึ้นมาเนื่องจากสามารถที่จะกำหนดพารามิเตอร์ให้กับโหนดนั้นๆได้ ซึ่งใช้จำลองพารามิเตอร์ของเว็บเซอร์วิส โดยจะอยู่ในรูปแฟ้มข้อมูลแบบข้อความโดยพารามิเตอร์ที่ได้ จะเป็นตัวเลขซึ่งแทนชื่อของพารามิเตอร์ ในแฟ้มข้อมูลจะมีจำนวนบรรทัดเท่ากับจำนวนโหนด ตัวอย่างของข้อมูลแฟ้มข้อมูล เช่นในบรรทัดที่ 3 มีตัวเลข 3 2 5 หมายถึงโหนดที่ 3 มีพารามิเตอร์ส่งออก 3 พารามิเตอร์ซึ่งก็คือพารามิเตอร์ชื่อ 3 พารามิเตอร์ชื่อ 2 และพารามิเตอร์ชื่อ 5 เป็นต้น โดยแฟ้มข้อมูลของพารามิเตอร์และแฟ้มข้อมูลของเมตริกซ์กราฟจะเก็บแยกแฟ้มข้อมูล

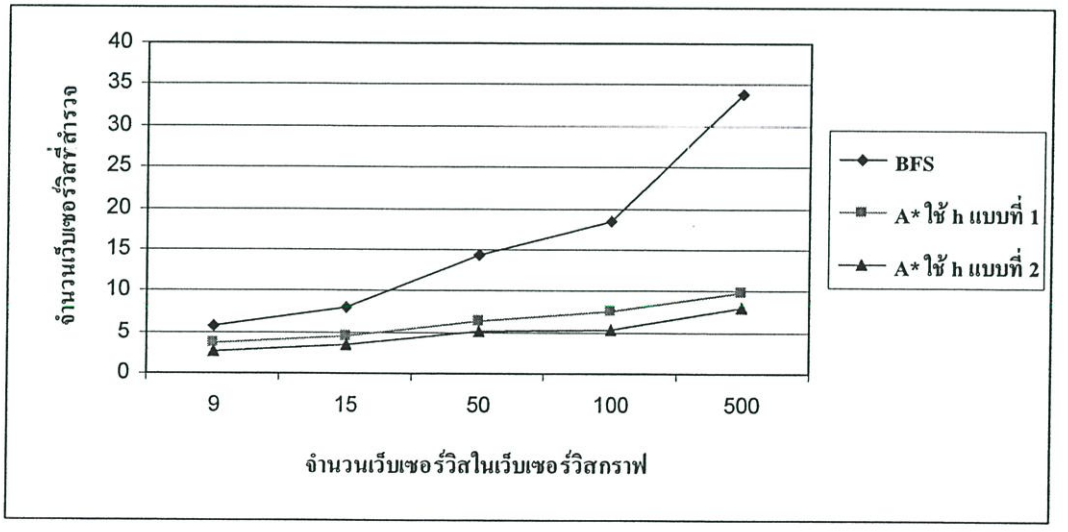
เมื่อได้กราฟที่อยู่ในรูปเมตริกซ์แล้ว จะใช้โปแกรม Graph Magics ในการตรวจสอบลักษณะของกราฟอีกครั้งเพื่อตรวจสอบความถูกต้องของกราฟอีกครั้ง ซึ่งเมื่อนำเมตริกซ์กราฟจากรูปที่ 4.1 จะได้กราฟลักษณะดังรูปที่ 4.2 แต่ในโปรแกรมนี้จะไม่สามารถแสดงพารามิเตอร์ได้จึงใช้ตรวจสอบลักษณะกราฟเท่านั้น

ในการทดลองกำหนดให้มีจำนวนโหนดในกราฟที่แตกต่างกันไปซึ่งแทนจำนวนเว็บเซอร์วิสในเว็บเซอร์วิสกราฟ โดยมีจำนวนดังนี้คือ 9 เว็บเซอร์วิส, 15 เว็บเซอร์วิส, 50 เว็บเซอร์วิส, 100 เว็บเซอร์วิสและ 500 เว็บเซอร์วิส โดยกราฟแต่ละขนาดจะทำการทดลอง 30 ครั้งโดยไม่บังคับว่ากราฟจะต้องมีรูปแบบอย่างไรนอกจากว่าต้องเป็นกราฟแบบไม่มีวงระบุทิศทางเท่านั้น ซึ่งเหตุที่จำลองขึ้นมาเพื่อให้สะดวกต่อการทดลองกับเว็บเซอร์วิสจำนวนมากได้ เนื่องจากถ้าใช้เว็บเซอร์วิสจริงจำนวนมากจะต้องติดต่อเพื่อขออนุญาตเข้าใช้เว็บเซอร์วิสในหลายบิสซิเนสเอนทิตีซึ่งส่วนใหญ่จะเป็นองค์กรทางธุรกิจที่เป็นผู้ให้บริการ ซึ่งโอกาสเป็นไปได้น้อยที่จะได้เว็บเซอร์วิสตามจำนวนที่ต้องการเพื่อทำการทดลอง



รูปที่ 4.2 กราฟแบบเครือข่าย

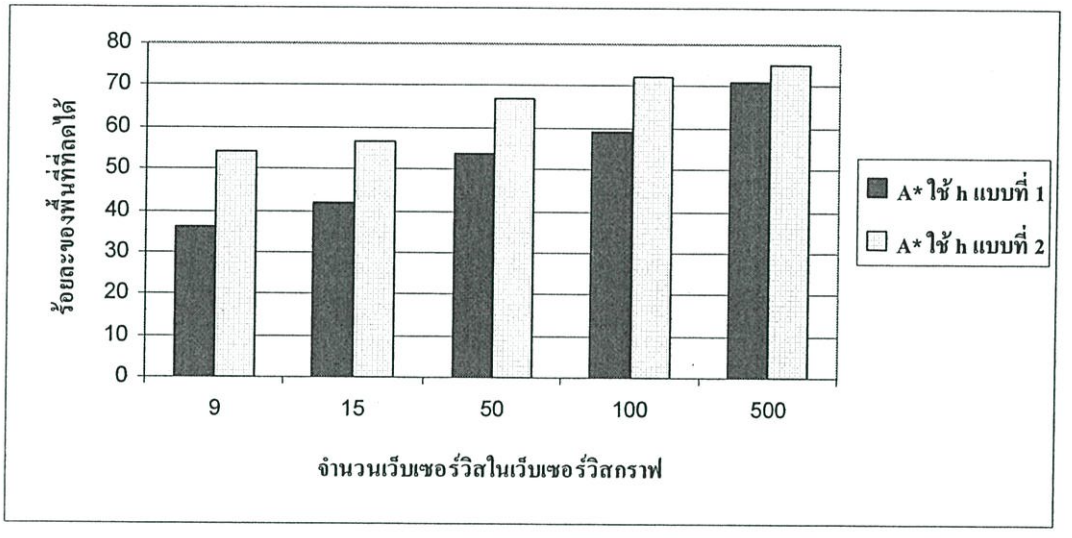
รูปที่ 4.3 เป็นผลการทดลองของการเปรียบเทียบพื้นที่ในการค้นหาเส้นทางที่ดีที่สุดในเว็บเซอร์วิสกราฟระหว่างการใช้ขั้นตอนวิธีค้นหาแบบแนวกว้างก่อนซึ่งเป็นวิธีการเดิม กับขั้นตอนวิธีการค้นหาแบบ A* โดยใช้ฟังก์ชัน h แบบที่ 1 และใช้ฟังก์ชัน h ในแบบที่ 2



รูปที่ 4.3 ผลการทดลอง

จากผลการทดลองในรูปที่ 4.3 เมื่อนำมาพิจารณาความแตกต่างของพื้นที่ที่ใช้ในการค้นหา พบว่าในการใช้ขั้นตอนวิธีการค้นหาแบบ A* เมื่อใช้ฟังก์ชัน h แบบที่ 1และแบบที่ 2 ที่สามารถลดพื้นที่ในการค้นหาได้จากวิธีการเดิมซึ่งใช้ขั้นตอนวิธีการค้นหาแบบแนวกว้างก่อน โดยจะแสดงเป็นร้อยละของพื้นที่ในการค้นหาที่สามารถลดได้แสดงได้ดังรูปที่ 4.4 ซึ่งพบว่าจำนวนพื้นที่ในการ

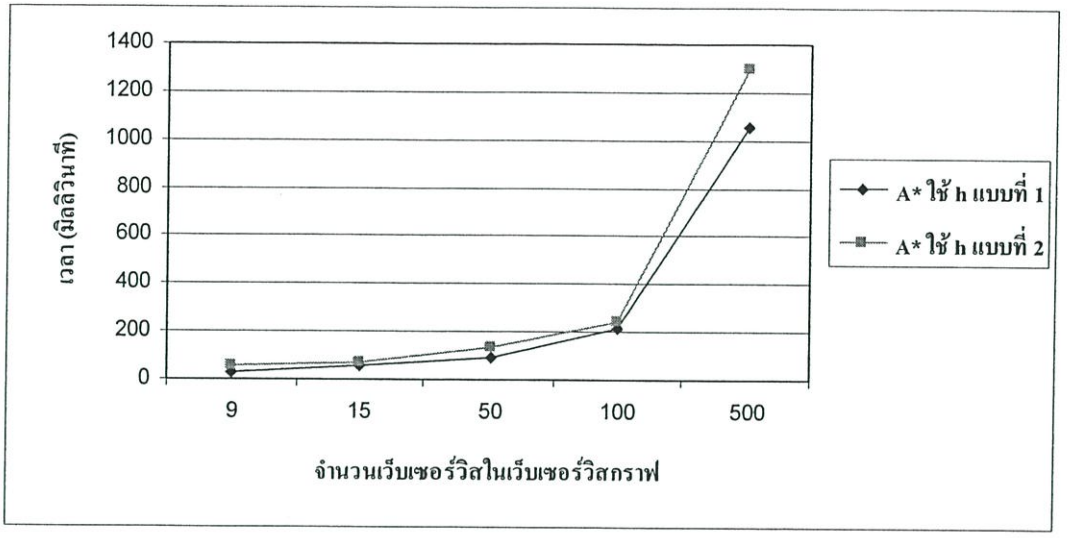
ค้นหาลดลงโดยเฉลี่ยได้มากถึง 52.34 % เมื่อใช้ขั้นตอนวิธีการค้นหาแบบ A* โดยใช้ฟังก์ชัน h แบบที่ 1 ซึ่งสามารถลดพื้นที่ในการค้นหาได้มากขึ้นเมื่อใช้ขั้นตอนวิธีการค้นหาแบบ A* โดยใช้ฟังก์ชัน h แบบที่ 2 ซึ่งจำนวนพื้นที่ในการค้นหาลดลงโดยเฉลี่ยได้เพิ่มขึ้นจากการใช้ h แบบที่ 1 อีก 12.6 % ซึ่งทั้งนี้จะขึ้นอยู่กับความสัมพันธ์ของพารามิเตอร์ส่งออกของเว็บเซอร์วิสกับพารามิเตอร์ที่ต้องการทั้งหมด



รูปที่ 4.4 ร้อยละของพื้นที่ที่ลดได้

รูปที่ 4.5 เมื่อทำการเปรียบเทียบในส่วนของเวลาที่ใช้จริง ระหว่างขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 1 และขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 2

เมื่อทำการเปรียบเทียบในส่วนของเวลาที่ใช้ซึ่งแสดงได้ดังรูปที่ 4.5 พบว่าใช้เวลาในการค้นหามากกว่าโดยเฉลี่ยแล้วประมาณ 36.62 มิลลิวินาที ซึ่งก็ถือว่าไม่มากนักเนื่องจากความแตกต่างอยู่ในหน่วยของมิลลิวินาที โดยเวลาที่มากกว่านั้นเกิดขึ้นเนื่องจากต้องเข้าไปตรวจสอบพารามิเตอร์ของเว็บเซอร์วิสที่อยู่ในยูดีดีไอซึ่งข้อมูลอยู่ในระบบเครือข่าย และทั้งนี้จะขึ้นอยู่กับคุณภาพของเครือข่ายซึ่งขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 1 นั้นไม่ต้องเข้าไปตรวจสอบพารามิเตอร์ในยูดีดีไออีกเนื่องจากได้ทำการเก็บเว็บเซอร์วิสกราฟอยู่ในรูปตารางดัชนีไว้แล้ว จึงทำให้ทำให้เวลาที่ใช้น้อยกว่า



รูปที่ 4.5 เวลาที่ใช้ในการค้นหา

4.3 สรุปและอภิปรายผลการทดลอง

จากรูปที่ 4.3 ได้แสดงผลการเปรียบเทียบพื้นที่ในการค้นหาเส้นทางในเว็บเซอร์วิสกราฟ ซึ่งได้ทำการเปรียบเทียบจำนวนเว็บเซอร์วิสที่เข้าไปค้นในเว็บเซอร์วิสกราฟระหว่างขั้นตอนวิธีการค้นหาแบบ A* โดยใช้ฟังก์ชัน h แบบที่ 1 และขั้นตอนวิธีการค้นหาแบบ A* โดยใช้ฟังก์ชัน h แบบที่ 2 ซึ่งพบว่าขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 1 สามารถลดจำนวนพื้นที่ในการค้นหาหรือจำนวนของเว็บเซอร์วิสที่เข้าไปค้นในเว็บเซอร์วิสกราฟ ได้ 1 ระดับชั้น ซึ่งถ้าเว็บเซอร์วิสในเว็บเซอร์วิสกราฟในระดับชั้นที่ n-1 มีจำนวนมาก เมื่อเว็บเซอร์วิสที่เป็นคำตอบอยู่ในระดับชั้นที่ n ก็จะเห็นความแตกต่างมากยิ่งขึ้น และเมื่อใช้ขั้นตอนวิธีการค้นหาแบบ A* โดยใช้ฟังก์ชัน h แบบที่ 2 สามารถลดพื้นที่การค้นลงได้เพิ่มขึ้น ซึ่งทั้งนี้จะขึ้นอยู่กับความสัมพันธ์ของพารามิเตอร์ส่งออกของเว็บเซอร์วิสกับพารามิเตอร์ที่ต้องการทั้งหมด

ในรูปที่ 4.5 แสดงให้เห็นว่าขั้นตอนวิธีการค้นหาแบบ A* โดยใช้ฟังก์ชัน h แบบที่ 2 จะใช้เวลาในการค้นหามากกว่าขั้นตอนวิธีการค้นหาแบบ A* โดยใช้ฟังก์ชัน h แบบที่ 1 เนื่องจากต้องเข้าไปค้นข้อมูลในระบบเครือข่าย แต่ถือว่าไม่มากนักเนื่องจากความแตกต่างอยู่ในหน่วยของมิลลิวินาที

ดังนั้นสรุปได้ว่าการปรับปรุงขั้นตอนการค้นหาเส้นทางในเว็บเซอร์วิสกราฟโดยใช้ขั้นตอนวิธีการค้นหาแบบ A* โดยใช้ฟังก์ชัน h แบบที่ 2 ซึ่งใช้โครงสร้างการเก็บของเว็บเซอร์วิสกราฟในการคำนวณและใช้ความสัมพันธ์ของพารามิเตอร์มาประกอบการคำนวณ สามารถที่จะลดพื้นการค้นหาได้มากกว่าขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 1 ซึ่งพิจารณาจากโครงสร้างการเก็บของเว็บเซอร์วิสกราฟในการคำนวณเป็นหลัก และขั้นตอนวิธีการค้นหาแบบ A*

โดยใช้ฟังก์ชัน h แบบที่ 2 จะใช้เวลาในการค้นหาเพิ่มมากขึ้นแต่จากผลการทดลองแสดงให้เห็นว่าเวลาที่เพิ่มขึ้นนั้นไม่มากนัก ทั้งนี้ขึ้นอยู่กับคุณภาพสัญญาณของระบบเครือข่าย

บทที่ 5
สรุปผลและแนวทางในการวิจัย

5.1 สรุปผลและวิเคราะห์ผลการทดลอง

งานวิจัยนี้ได้นำเสนอวิธีการลดพื้นที่ในการค้นหาเส้นทางที่ดีที่สุดในเว็บเซอร์วิสกราฟโดยทำการปรุงในขั้นตอนการค้นหาเส้นทางในเว็บเซอร์วิสกราฟซึ่งเดิมจากงานวิจัย [3][4] ได้ใช้ขั้นตอนวิธีการค้นหาแบบแนวกว้างก่อนซึ่งผู้วิจัยพบว่าพื้นที่ในการค้นหาเส้นทางหรือจำนวนเว็บเซอร์วิสที่ต้องเข้าไปสำรวจนั้นสามารถทำให้ลดลงได้อีก เนื่องจากขั้นตอนวิธีการค้นหาแบบแนวกว้างก่อนจะทำการค้นหาไปทีละระดับชั้นจนครบทั้งระดับก่อนจะไปค้นในระดับชั้นถัดไปทำให้เว็บเซอร์วิสเกือบทั่วทั้งเว็บเซอร์วิสกราฟจะถูกเข้าไปสำรวจเมื่อใช้วิธีการนี้

จากข้างต้นผู้วิจัยจึงได้ทำการปรับปรุงขั้นตอนการค้นหาเส้นทางในเว็บเซอร์วิสกราฟโดยใช้ขั้นตอนวิธีการค้นหาแบบ A* เนื่องจากขั้นตอนวิธีการค้นหาแบบ A* สามารถได้คำตอบที่ดีที่สุดเช่นเดียวกันแต่มีฟังก์ชันฮิวริสติกสำหรับการคำนวณหาต้นทุนที่น้อยที่สุดเพื่อให้จำนวนโหนดที่ต้องเข้าไปสำรวจมีจำนวนลดลง โดยในงานวิจัยนี้ได้นำเสนอ ขั้นตอนวิธีการค้นหาแบบ A* โดยมีฟังก์ชันฮิวริสติก 2 แบบโดยในที่นี้คือฟังก์ชัน h แบบที่ 1 แนวคิดมาจากการพิจารณาจากโครงสร้างของเว็บเซอร์วิสกราฟเป็นหลัก ส่วนแบบที่ 2 จะปรับปรุงจากแบบที่ 1 ซึ่งจะคำนวณเพิ่มเติมโดยการพิจารณาจากโครงสร้างของเว็บเซอร์วิสกราฟและพิจารณาจากความสัมพันธ์ของพารามิเตอร์ส่งออกของเว็บเซอร์วิสตำแหน่งที่กำลังสำรวจเทียบกับพารามิเตอร์ที่ต้องการทั้งหมดของการร้องขอ

จากทฤษฎีพบว่าขั้ตอนวิธีการค้นหาแบบ A* โดยใช้ ฟังก์ชัน h แบบที่ 1 จะสามารถลดพื้นที่ในการค้นหาได้ 1 ระดับชั้น ซึ่งถ้าเว็บเซอร์วิสในเว็บเซอร์วิสกราฟในระดับชั้นที่ n-1 มีจำนวนมาก เมื่อเว็บเซอร์วิสที่เป็นคำตอบอยู่ในระดับชั้นที่ n ก็จะเห็นความแตกต่างมากยิ่งขึ้น จากผลการทดลองพบว่าสามารถลดลงได้โดยเฉลี่ย 52.34 % และเมื่อใช้ขั้นตอนวิธีการค้นหาแบบ A* โดยใช้ฟังก์ชัน h แบบที่ 2 ในการค้นหาเส้นทางจะสามารถลดพื้นที่การค้นลงได้เพิ่มขึ้นจากการใช้ฟังก์ชัน h แบบที่ 1 อีก12.6 % ทั้งนี้จำนวนของเว็บเซอร์วิสที่สามารถลดได้จะขึ้นอยู่กับรูปแบบความสัมพันธ์ของพารามิเตอร์ส่งออกของเว็บเซอร์วิสตำแหน่งที่กำลังสำรวจเทียบกับพารามิเตอร์ที่ต้องการทั้งหมดของการร้องขอ ซึ่งกรณีที่แย่ที่สุดของการใช้ฟังก์ชัน h แบบที่ 2 จะสามารถลดได้เท่ากับการใช้ฟังก์ชัน h แบบที่ 1

จากผลการทดลองในบทที่ 4 พบว่าการใช้ขั้นตอนวิธีการค้นหาแบบ A* โดยการใช้ฟังก์ชัน h แบบที่ 2 ในขั้นตอนการค้นหาเส้นทางในเว็บเซอร์วิสกราฟสามารถลดจำนวนพื้นที่ในการค้นหา

ได้มากกว่าการใช้ขั้นตอนวิธีการค้นหาแบบ A* ที่ใช้ฟังก์ชัน h แบบที่ 1 แต่เวลาที่ใช้ในการค้นหาของการใช้ฟังก์ชัน h แบบที่ 2 จะมากกว่าใช้ฟังก์ชัน h แบบที่ 1 โดยเฉลี่ย 36.62 มิลลิวินาที ซึ่งถือว่าไม่มากนัก ทั้งนี้เวลาที่ใช้ของขั้นตอนวิธีการค้นหาแบบ A* โดยใช้ฟังก์ชัน h แบบที่ 2 จะขึ้นอยู่กับความเร็วในการให้บริการของระบบเครือข่าย

5.2 แนวทางในการพัฒนาการวิจัย

งานวิจัยนี้ได้ปรับปรุงขั้นตอนการค้นหาเส้นทางในเว็บเซอร์วิสกราฟโดยใช้ขั้นตอนวิธีการค้นหาแบบ A* ซึ่งพบว่าสามารถลดพื้นที่ในการค้นหาได้

เมื่อพิจารณาโครงสร้างข้อมูลของเว็บเซอร์วิสกราฟพบว่าในขั้นตอนการสร้างเว็บเซอร์วิสกราฟ พบว่ากราฟที่สร้างได้จากขั้นตอนนี้ เกิดเส้นทางที่ไปไม่ถึงคำตอบเกิดขึ้น ซึ่งเป็นสาเหตุทำให้ต้องเข้าไปสำรวจเว็บเซอร์วิสในเส้นทางที่ไปไม่ถึงคำตอบ ผู้วิจัยจึงมีความเห็นว่าถ้าปรับปรุงในขั้นตอนการสร้างเว็บเซอร์วิสกราฟให้มีเพียงเส้นทางที่ไปถึงคำตอบได้เท่านั้น จะสามารถลดพื้นในการค้นหาเส้นทางในเว็บเซอร์วิสกราฟลงได้เพิ่มขึ้น

ในการคำนวณฟังก์ชัน h แบบที่ 2 จะพิจารณาความสัมพันธ์จากชื่อพารามิเตอร์ของเว็บเซอร์วิสเป็นหลัก ซึ่งถ้าปรับปรุงให้เพิ่มความสามารถในการคำนวณโดยการพิจารณาไปถึงโครงสร้างของข้อมูลพารามิเตอร์ร่วมกับการพิจารณาจากชื่อของพารามิเตอร์ ตามแนวทางของงานวิจัย [9] จะสามารถลดพื้นในการค้นหาได้เพิ่มขึ้น

บรรณานุกรม

[1] A. Broder, R. Kumar, F. Maghoul, and et al. 2000. "Graph structure in the Web." **Computer Networks.** Vol.33(1-6) : 309-320.

[2] G. Meghabghab. 2002. "Discovering authorities and hubs in different topological Web graph Structures." **Information Processing and Management.** Vol.38(1) : 111-140.

[3] J.X. Liu and L.Chao. 2006. "Web Service as a Graph and Its Application for Service Discovery." **Proc. Of the IEEE International Conference on Grid and Cooperative Computing(GCC).**

[4] J.X. Lui, J.Liu and L.Chao. 2007. "Design and Implementation of an Extended UDDI Registration Center for Web Service Graph." **IEEE International Conference on Web Services(ICWS).**

[5] Mark Allen Weiss. 2006. **Data Structures & Analysis in JAVA**. Addison Wesley Longman,Inc.

[6] Michael Negnevitsky. 2005. **Artificial Intelligence: A Guide to intelligent Systems.** 2nd ed. Addison Wesley Longman,Inc.

[7] S.Grimm. 2007. "Intersection-Based Matchmaking for Semantic Web Service Discovery." **IEEE International Conference on Internet and Web Applications and Services(ICIW).**

[8] S.C. Oh, B.W. On, E.J. Larson and D.W. Lee. 2005 "BF*:Web Service Discovery and Composition as Graph Search Ploblem." **Proc. Of the IEEE International Conference on e-Technology e-Commerce and e-Service(EEE).**

[9] S.C. Oh, J.W. Yoo, H.Y. Kil, D.W. Lee and S.R. T. Kumara. 2007. "Semantic Web-Service Discovery and Composition Using Flexible Parameter Matching." **IEEE International Conference on E-Commerce Technology and IEEE International Conference on Enterprise Computing, E-Commerce and E-Services(CEC-EEE).**

ประวัติผู้ทำวิจัย

นายณัทภต นันตาลิต เกิดที่จังหวัดจังหวัดเชียงใหม่ สำเร็จการศึกษาปริญญาตรีวิทยาศาสตรบัณฑิต สาขาวิทยาการคอมพิวเตอร์ จากภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยแม่โจ้ ในปีการศึกษา 2547 ได้เข้านำงานในบริษัทนิกส์ซอฟแวร์ประเทศไทย ตำแหน่ง วิศวกรซอฟแวร์ ในปี 2548 และปีเดียวกันได้เข้ารับราชการครูในตำแหน่งครูผู้ช่วย โรงเรียนศึกษานารีอนุสรณ์ 1 พื้นที่การศึกษาเชียงใหม่เขต 3 ต่อมาได้เข้าทำงานในบริษัทอีดีไอสยาม ในตำแหน่ง โปรแกรมเมอร์ในปี 2549 และเข้าศึกษาต่อในระดับปริญญาโท หลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิทยาการคอมพิวเตอร์ ภาคคณิตศาสตร์และวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ในปีการศึกษา 2549