การสืบค้นฟังก์ชันการขึ้นต่อกัน โดยใช้การสมมูลและการปิดของแอททริบิวท์

MINING FUNCTIONAL DEPENDENCIES USING ATTRIBUTE EQUIVALENCES AND CLOSURES

นิภาพร ไฝขาว
NIPAPORN PHAIKHAO

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาการคอมพิวเตอร์
บัณฑิตวิทยาลัย
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
พ.ศ. 2551

KMITL-2008-SC-M-002-344

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

การสืบค้นฟังก์ชันการขึ้นต่อกัน
โดยใช้การสมมูลและการปิดของแอททริบิวท์

MINING FUNCTIONAL DEPENDENCIES
USING ATTRIBUTE EQUIVALENCES AND CLOSURES

นิภาพร ไฝขาว

NIPAPORN PHAIKHAO

เลขหมู่.....
เลขทะเบียน.....81396
วัน,เดือน,ปี.....11 ส.ค. 2551

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต

สาขาวิทยาการคอมพิวเตอร์

บัณฑิตวิทยาลัย

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ.2551

KMITL-2008-SC-M-002-344

MINING FUNCTIONAL DEPENDENCIES
USING ATTRIBUTE EQUIVALENCES AND CLOSURES

NIPAPORN PHAIKHAO

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

KMITL-2008-SC-M-002-344

COPYRIGHT 2008

SCHOOL OF GRADUATE STUDIES

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

หัวข้อวิทยานิพนธ์	การสืบค้นฟังก์ชันการขึ้นต่อกัน โดยใช้การสมมูลและการปิดของแอททริบิวท์
นักศึกษา	นางสาวนิภาพร ไฝขาว
รหัสประจำตัว	48067502
ปริญญา	วิทยาศาสตรมหาบัณฑิต
สาขาวิชา	วิทยาการคอมพิวเตอร์
พ.ศ.	2551
อาจารย์ที่ปรึกษาวิทยานิพนธ์	รศ.ดร.วีระ บุญจริง

บทคัดย่อ

งานวิจัยนี้มีวัตถุประสงค์เพื่อพัฒนาขั้นตอนวิธีใหม่สำหรับการสืบค้นฟังก์ชันการขึ้นต่อกันจากฐานข้อมูลขนาดใหญ่ โดยใช้การสมมูลของแอททริบิวท์ (Attribute Equivalence) และการปิดของแอททริบิวท์ (Attribute Closure) งานวิจัยนี้ได้พัฒนาขั้นตอนวิธีโดยใช้วิธีการค้นหาคำตอบแบบสองทิศทาง (Bidirectional Search) เพื่อสืบค้นฟังก์ชันการขึ้นต่อกันน้อยที่สุด (Minimal Functional Dependency) การทดสอบประสิทธิภาพของขั้นตอนวิธีใหม่นี้ทำโดยพัฒนาโปรแกรมเพื่อการทดสอบขั้นตอนวิธี แล้วทำการทดลองเพื่อนับจำนวนเซตแอททริบิวท์ที่ต้องตรวจสอบฟังก์ชันการขึ้นต่อกัน โดยทดลองกับชุดข้อมูลมาตรฐาน เพื่อให้ผลการทดลองสามารถเปรียบเทียบกับขั้นตอนวิธี FD_Mine ซึ่งใช้การค้นหาแบบกว้าง (Level-wise Search) ผลการทดลองพบว่า การค้นหาคำตอบแบบสองทิศทางด้วยวิธีการค้นหาจากด้านล่างสลับด้านบนสามารถลดจำนวนเซตแอททริบิวท์ที่ต้องตรวจสอบฟังก์ชันการขึ้นต่อกันได้มีประสิทธิภาพสูงสุด

Thesis Title	Mining Functional Dependencies Using Attribute Equivalences and Closures
Student	Miss.Nipaporn Phaikhao
Student ID.	48067502
Degree	Master of Science
Program	Computer Science
Year	2008
Thesis Advisor	Assoc.Prof.Dr.Veera Boonjing

ABSTRACT

This research proposes a new method for mining functional dependencies from large databases using the concepts of equivalence and closure of attribute sets. It develops an algorithm using Bidirectional search strategy for generating minimal set of functional dependencies. It is implemented and tested for counting number of candidate functional dependencies. It can reduce the size of the dataset and the number of candidates. The results of the experiments on 8 UCI datasets show that our approach can eliminate candidates more effective than the previous methods.

กิตติกรรมประกาศ

วิทยานิพนธ์นี้มิอาจสำเร็จลุล่วงได้ด้วยดี หากมิได้รับคำแนะนำ คำชี้แจง ความรู้ และแนวทางการแก้ปัญหาต่างๆ จาก รศ.ดร.วีระ บุญจริง ผู้เป็นอาจารย์ที่ปรึกษาวิทยานิพนธ์ ซึ่งท่านได้สละเวลาให้กับข้าพเจ้าอย่างเต็มที่ จึงใคร่ขอขอบพระคุณเป็นอย่างสูง

ขอขอบพระคุณ ผศ.ดร.จีรพร วีระพันธุ์ ผศ.ดร.ศรัณย์ อินทโกสุม และดร.เฉลิมศักดิ์ เลิศวงศ์เสถียร คณะกรรมการสอบวิทยานิพนธ์ ที่กรุณาให้คำแนะนำตลอดจนข้อชี้แนะที่เป็นประโยชน์ในการทำวิทยานิพนธ์อย่างดียิ่งมาโดยตลอด

ขอขอบพระคุณนายนิพัทธ์ อินทอง ที่กรุณาให้คำแนะนำ คำปรึกษา และให้โอกาสข้าพเจ้าได้ศึกษาต่อระดับปริญญาโท

ขอขอบคุณนายธนวัฒน์ ปัญญวรรณศิริ นายฮาหรูน บิลยะลา และนางสาววรลักษณ์ โล่ห์สุริยะ ที่ช่วยรับผิดชอบภาระงาน และให้ความช่วยเหลือด้านต่างๆ ตลอดมา

ขอขอบคุณนางสาวพนิดา ทรงรัมย์ นายสันทิฏฐ์ นรบิน และนายจรัสเดช ชัชวาลย์ สำหรับความช่วยเหลือ คำแนะนำ และคำปรึกษาในการทำวิทยานิพนธ์อย่างดีตลอดมา

ขอขอบคุณพี่ๆ เพื่อนๆ และน้องๆ ปริญญาโททุกคน สำหรับความช่วยเหลือ ช่วยอำนวยความสะดวกด้านต่างๆ และกำลังใจในการทำวิทยานิพนธ์อย่างดีตลอดมา

สุดท้ายนี้ ข้าพเจ้าขอกราบขอบพระคุณ บิดา มารดา ที่ให้การสนับสนุนการศึกษา และให้กำลังใจ เป็นแรงผลักดันให้ข้าพเจ้าทำวิทยานิพนธ์ให้สำเร็จอย่างดีตลอดมา

สำหรับคุณงามความดีและประโยชน์อันพึงมีจากวิทยานิพนธ์ฉบับนี้ ข้าพเจ้าขอมอบแด่ผู้มีพระคุณทุกท่าน บิดา มารดา ตลอดจนครูอาจารย์ที่เคารพทุกท่านที่ได้ประสิทธิ์ประสาทวิชาความรู้ และถ่ายทอดประสบการณ์ที่ดีให้กับข้าพเจ้า

นิภาพร ไฝขาว

สารบัญ

หน้า

บทคัดย่อภาษาไทย I

บทคัดย่อภาษาอังกฤษ II

กิตติกรรมประกาศ III

สารบัญ IV

สารบัญตาราง VI

สารบัญรูป VII

บทที่ 1 บทนำ 1

1.1 ความเป็นมาและความสำคัญของปัญหา 1

1.2 ความมุ่งหมายและวัตถุประสงค์ของการศึกษา 2

1.3 สมมุติฐานของการศึกษา 2

1.4 ขอบเขตการศึกษา 2

1.5 ส่วนประกอบของวิทยานิพนธ์ 3

บทที่ 2 ทฤษฎีและงานวิจัยที่เกี่ยวข้อง 4

2.1 ทฤษฎีฟังก์ชันการขึ้นต่อกัน 4

2.2 คุณสมบัติของฟังก์ชันการขึ้นต่อกัน 12

2.3 งานวิจัยที่เกี่ยวข้อง 13

บทที่ 3 ขั้นตอนวิธีการสืบค้นฟังก์ชันการขึ้นต่อกัน 15

3.1 ขั้นตอนวิธี 15

3.2 การวิเคราะห์ขั้นตอนวิธี 27

บทที่ 4 การทดลอง 28

4.1 การทดลอง 28

4.1.1 ข้อมูลที่ใช้ในการทดลอง 28

4.1.2 เครื่องมือที่ใช้ในการทดลอง 29

4.1.3 การออกแบบการทดลองและเหตุผล 29

สารบัญ (ต่อ)

หน้า

4.2 ผลการทดลอง 30
4.2.1 ผลการทดลอง 30
4.2.2 การเปรียบเทียบวิธีการค้นหาคำตอบ 30
4.2.3 การเปรียบเทียบกับขั้นตอนวิธี FD_Mine 34

บทที่ 5 สรุปและข้อเสนอแนะ 35
5.1 สรุป 35
5.2 ข้อเสนอแนะ 35

เอกสารอ้างอิง 36

ประวัติผู้เขียน 38

สารบัญตาราง

ตารางที่ หน้า

2.1 ตารางฐานข้อมูลเชิงสัมพันธ์ 5

2.2 ตารางฐานข้อมูลเชิงสัมพันธ์ 6

2.3 ตารางฐานข้อมูลเชิงสัมพันธ์ 9

2.4 ตารางฐานข้อมูลเชิงสัมพันธ์ 10

3.1 ตารางฐานข้อมูลเชิงสัมพันธ์ 21

4.1 ตารางชุดข้อมูลมาตรฐาน 28

4.2 ตารางแสดงจำนวนการตรวจสอบ FDs 30

4.3 ตารางแสดงร้อยละการตรวจสอบ FDs ที่ลดลงเมื่อเทียบกับ
จำนวนการตรวจสอบ FDs ทั้งหมด 31

4.4 ตารางแสดงค่าเฉลี่ยการตรวจสอบ FDs เมื่อเทียบกับจำนวน FDs 32

สารบัญรูป

รูปที่ หน้า

3.1 แสดงเซตแอททริบิวท์ระดับต่างๆ 18
3.2 แสดงผลการทำงานของขั้นตอนวิธี ณ ระดับที่ 1 23
3.3 สรุปผลการทำงานของขั้นตอนวิธี ณ ระดับที่ 1 23
3.4 แสดงผลการทำงานของขั้นตอนวิธี ณ ระดับที่ 4 24
3.5 สรุปผลการทำงานของขั้นตอนวิธี ณ ระดับที่ 4 24
3.6 แสดงผลการทำงานของขั้นตอนวิธี ณ ระดับที่ 2 25
3.7 สรุปผลการทำงานของขั้นตอนวิธี ณ ระดับที่ 2 25
3.8 แสดงผลการทำงานของขั้นตอนวิธี ณ ระดับที่ 3 26
3.9 สรุปผลการทำงานของขั้นตอนวิธี ณ ระดับที่ 3 26
3.10 สรุปผลการทำงานของขั้นตอนวิธี 27
4.1 กราฟแสดงร้อยละการตรวจสอบ FDs ที่ลดลง 31
4.2 กราฟแสดงค่าเฉลี่ยการตรวจสอบ FDs เมื่อเทียบกับจำนวน FDs 33

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ฟังก์ชันการขึ้นต่อกัน (Functional Dependency: FD) นำเสนอโดย E. F. Codd ในปี 1970 [4] เป็นความสัมพันธ์ระหว่างแอททริบิวท์ในตารางฐานข้อมูลเชิงสัมพันธ์ ซึ่งเป็นข้อบังคับ (Constraint) ที่สำคัญในการออกแบบฐานข้อมูล และทฤษฎีฐานข้อมูลเชิงสัมพันธ์ โดยเน้นการควบคุมบูรณภาพของข้อมูล (Integrity Constraint) และความถูกต้องของข้อมูล (Data Consistency)

การสืบค้นฟังก์ชันการขึ้นต่อกันเป็นงานวิจัยที่ได้รับความสนใจและการศึกษาจากนักวิจัยอย่างมาก งานวิจัยส่วนใหญ่มีวัตถุประสงค์เพื่อพัฒนาและปรับปรุงขั้นตอนวิธีที่มีประสิทธิภาพในการสืบค้นฟังก์ชันการขึ้นต่อกันจากฐานข้อมูลเชิงสัมพันธ์ เพื่อนำไปปรับโครงสร้างของฐานข้อมูลในขั้นตอนการออกแบบฐานข้อมูลให้มีการจัดเก็บข้อมูลที่เหมาะสม และเลือกใช้ข้อมูลได้อย่างถูกต้อง นอกจากนี้ยังนำไปประยุกต์ใช้สำหรับการสืบค้นกฎความสัมพันธ์ (Association Rule) บนฐานข้อมูลขนาดใหญ่ที่เรียกว่า เหมืองข้อมูล (Data Mining) ทำให้ได้มาซึ่งความรู้ที่มีความสำคัญและสามารถนำไปใช้ประโยชน์ได้

ขั้นตอนวิธี TANE [1, 2], FUN [7, 8] และ FD_Mine [10, 11] เป็นการสืบค้นโดยใช้วิธีการสร้างแล้วทดสอบ (Generate and Test) โดยจะสร้างคำตอบที่เป็นไปได้แล้วทำการทดสอบจนครบทุกคำตอบที่เป็นไปได้ในโครงข่ายคำตอบ ซึ่งจะลดขนาดเนื้อที่ (Search Space) ในการหาคำตอบโดยกฎการตัดที่มีประสิทธิภาพ และใช้แนวคิดการแบ่งพาร์ติชัน (Partition) โดยใช้วิธีการค้นหาแบบกว้าง (Level-wise Search) ซึ่งการทดสอบฟังก์ชันการขึ้นต่อกันจะมีประสิทธิภาพสูงเมื่อจำนวนของทูเพิล (Tuple) มาก ขั้นตอนวิธีทั้งสามแตกต่างกันในส่วนของกฎการตัดเซตแอททริบิวท์ที่ซ้ำซ้อน ซึ่งข้อเสียของการสืบค้นโดยใช้การสร้างแล้วทดสอบจะเสียเวลาจำนวนมากในการทดสอบให้ครบทุกคำตอบ

ขั้นตอนวิธี Dep-Miner [3] และ FastFDs [9] เป็นการสืบค้นโดยใช้วิธีการปกคลุมน้อยที่สุด (Minimal Cover) โดยพิจารณาคู่ของทูเพิล ซึ่งจะเริ่มจากการแบ่งพาร์ติชัน แล้วทำการสตริปพาร์ติชัน (Strip Partition) ที่มีขนาดตั้งแต่ 2 ขึ้นไป เพื่อหาเซตสอดคล้อง (Agree Set) และสร้างเซตมากที่สุด (Maximal Set) ในการสืบค้นฟังก์ชันการขึ้นต่อกัน ขั้นตอนวิธีทั้งสองแตกต่างกันตรงวิธีการค้นหา กล่าวคือ ขั้นตอนวิธี Dep-Miner ใช้วิธีการค้นหาแบบกว้าง ส่วนขั้นตอนวิธี FastFDs ใช้วิธีการค้นหาแนวลึก (Depth-first Search) ซึ่งข้อเสียของขั้นตอนวิธีทั้งสองจะเสียเวลาจำนวนมากในการคำนวณหาเซตสอดคล้อง

ดังนั้นงานวิจัยนี้จึงนำเสนอขั้นตอนวิธีการสืบค้นฟังก์ชันการขึ้นต่อกัน ซึ่งใช้ทฤษฎีฐานข้อมูลเชิงสัมพันธ์และทฤษฎีจากงานวิจัยเดิม โดยใช้การสมมูลของแอททริบิวท์ (Attribute Equivalence) และการปิดของแอททริบิวท์ (Attribute Closure) นอกจากนี้ยังใช้คุณสมบัติการขยายของฟังก์ชันการขึ้นต่อกัน (Augmentation Rule) ช่วยในการตัดเซตแอททริบิวท์ที่ซ้ำซ้อนด้วย และทำการค้นหาคำตอบจากโครงข่ายคำตอบสองทิศทาง (Bidirectional Search) ซึ่งผลจากขั้นตอนวิธีนี้จะลดจำนวนเซตแอททริบิวท์ที่ต้องตรวจสอบฟังก์ชันการขึ้นต่อกัน และได้ผลลัพธ์ถูกต้องครบถ้วน

1.2 ความมุ่งหมายและวัตถุประสงค์ของการศึกษา

งานวิจัยนี้มีวัตถุประสงค์เพื่อศึกษาทฤษฎีฟังก์ชันการขึ้นต่อกัน พร้อมทั้งศึกษาคุณสมบัติของฟังก์ชันการขึ้นต่อกัน และพัฒนาขั้นตอนวิธีใหม่สำหรับการสืบค้นฟังก์ชันการขึ้นต่อกันบนฐานข้อมูลขนาดใหญ่ โดยใช้การสมมูลของแอททริบิวท์ (Attribute Equivalence) และการปิดของแอททริบิวท์ (Attribute Closure) เพื่อลดจำนวนเซตแอททริบิวท์ที่ต้องตรวจสอบฟังก์ชันการขึ้นต่อกันจากโครงข่ายคำตอบ แล้วเปรียบเทียบจำนวนเซตแอททริบิวท์ที่ต้องตรวจสอบฟังก์ชันการขึ้นต่อกันกับขั้นตอนวิธีอื่นเพื่อหาข้อดีและข้อจำกัดของวิธีต่างๆ

1.3 สมมุติฐานของการศึกษา

การสืบค้นฟังก์ชันการขึ้นต่อกันโดยใช้การสมมูลของแอททริบิวท์ (Attribute Equivalence) และการปิดของแอททริบิวท์ (Attribute Closure) ซึ่งผลจากขั้นตอนวิธีนี้จะลดจำนวนเซตแอททริบิวท์ที่ต้องตรวจสอบฟังก์ชันการขึ้นต่อกันจากโครงข่ายคำตอบ

1.4 ขอบเขตการศึกษา

1.4.1 ศึกษาแนวทางในการสืบค้นฟังก์ชันการขึ้นต่อกันโดยใช้การสมมูลของแอททริบิวท์ และการปิดของแอททริบิวท์

1.4.2 การทดสอบประสิทธิภาพของขั้นตอนวิธีทำโดยพัฒนาโปรแกรมเพื่อทดสอบขั้นตอนวิธี

1.4.3 การทำการทดลองจะนับจำนวนเซตแอททริบิวท์ที่ต้องตรวจสอบฟังก์ชันการขึ้นต่อกัน

1.4.4 ชุดข้อมูลในการทดลองจะใช้ชุดข้อมูลมาตรฐาน

1.4.5 เปรียบเทียบจำนวนเซตแอททริบิวท์ที่ต้องตรวจสอบฟังก์ชันการขึ้นต่อกันกับขั้นตอนวิธีอื่น

1.5 ส่วนประกอบของวิทยานิพนธ์

ส่วนที่เหลือของวิทยานิพนธ์ฉบับนี้ ประกอบด้วยบทต่างๆ ดังนี้

บทที่ 2 กล่าวถึงทฤษฎีและงานวิจัยที่เกี่ยวข้อง โดยแบ่งเนื้อหาออกเป็น 3 ส่วน ดังนี้ ส่วนแรกกล่าวถึงทฤษฎีฟังก์ชันการขึ้นต่อกัน ส่วนที่สองกล่าวถึงคุณสมบัติของฟังก์ชันการขึ้นต่อกัน และส่วนสุดท้ายกล่าวถึงงานวิจัยที่เกี่ยวข้องกับการสืบค้นฟังก์ชันการขึ้นต่อกัน

บทที่ 3 กล่าวถึงขั้นตอนวิธีการสืบค้นฟังก์ชันการขึ้นต่อกัน โดยแบ่งเนื้อหาออกเป็น 2 ส่วน ดังนี้ ส่วนแรกกล่าวถึงขั้นตอนวิธีการสืบค้นฟังก์ชันการขึ้นต่อกัน ส่วนที่ 2 เป็นการวิเคราะห์ขั้นตอนวิธี

บทที่ 4 กล่าวถึงการทดลองและผลการทดลอง เพื่อเปรียบเทียบผลที่ได้จากการสืบค้นฟังก์ชันการขึ้นต่อกันโดยใช้การสมมูลและการปิดของแอททริบิวท์ ด้วยวิธีการค้นหาคำตอบจากโครงข่ายคำตอบสองทิศทาง และเปรียบเทียบผลกับขั้นตอนวิธี FD_Mine

บทที่ 5 สรุปผลการทดลอง และข้อเสนอแนะเกี่ยวกับการสืบค้นฟังก์ชันการขึ้นต่อกัน

บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

ในบทนี้จะกล่าวถึงนิยาม ทฤษฎีพื้นฐาน และคุณสมบัติของฟังก์ชันการขึ้นต่อกันพร้อมทั้งนำเสนองานวิจัยที่เกี่ยวข้อง โดยแบ่งเนื้อหาออกเป็น 3 ส่วน ดังนี้ ส่วนที่ 1 จะกล่าวถึงทฤษฎีฟังก์ชันการขึ้นต่อกัน ส่วนที่ 2 จะกล่าวถึงคุณสมบัติของฟังก์ชันการขึ้นต่อกัน และส่วนสุดท้ายจะกล่าวถึงงานวิจัยที่เกี่ยวข้องกับการสืบค้นฟังก์ชันการขึ้นต่อกัน

2.1 ทฤษฎีฟังก์ชันการขึ้นต่อกัน

ฟังก์ชันการขึ้นต่อกันเป็นความสัมพันธ์เชิงฟังก์ชันระหว่างแอททริบิวท์หนึ่ง หรืออาจมากกว่าหนึ่งแอททริบิวท์ (Attribute) ประกอบกัน แล้วสามารถระบุค่าของแอททริบิวท์อื่นๆ ในทูเพิล (Tuple) เดียวกันของรีเลชัน (Relation) นั้นได้

นิยามพื้นฐานของทฤษฎีฟังก์ชันการขึ้นต่อกัน [1, 7, 8, 10, 11, 12] มีรายละเอียด ดังนี้

นิยามที่ 2.1 ฟังก์ชันการขึ้นต่อกัน (Functional Dependency: FD)

กำหนดให้ X และ A เป็นเซตแอททริบิวท์ในตารางฐานข้อมูลเชิงสัมพันธ์ R และ r เป็นค่าข้อมูลของเซตแอททริบิวท์ในตารางฐานข้อมูลเชิงสัมพันธ์ จะกล่าวว่า แอททริบิวท์ X บ่งชี้แอททริบิวท์ A เขียนแทนด้วย $X \rightarrow A$ เป็นฟังก์ชันการขึ้นต่อกัน ก็ต่อเมื่อ สำหรับทุกๆ คู่แถวใดๆ ที่ t_i และ t_j ใน r ถ้าทุกๆ คู่แถวที่ t_i และ t_j ของเซตแอททริบิวท์ X มีค่าเท่ากัน แล้วทุกๆ คู่แถวที่ t_i และ t_j ของเซตแอททริบิวท์ A มีค่าเท่ากันด้วย นั่นคือ แต่ละค่าของเซตแอททริบิวท์ X จะมีค่าของเซตแอททริบิวท์ A ที่สอดคล้องกันเพียง 1 ค่าเสมอ หรือกล่าวว่า แอททริบิวท์ X บ่งชี้แอททริบิวท์ A เป็นฟังก์ชันการขึ้นต่อกัน เขียนแทนด้วย

$$\forall(t_i, t_j) \in r, t_i[X] = t_j[X] \Longrightarrow t_i[A] = t_j[A]$$

ตัวอย่างที่ 2.1 แอททริบิวท์ A บ่งชี้แอททริบิวท์ B เป็นฟังก์ชันการขึ้นต่อกัน ($A \rightarrow B$)

จากตารางที่ 2.1 แสดงให้เห็นว่าแต่ละค่าของแอททริบิวท์ A จะมีค่าแอททริบิวท์ B ที่สอดคล้องกันเพียง 1 ค่าเสมอ กล่าวคือ

แถวที่ t_1, t_7 มีค่าแอททริบิวท์ A เท่ากับ 0 เท่ากัน แล้วบ่งชี้แถวที่ t_1, t_7 ของแอททริบิวท์ B ซึ่งมีค่าเท่ากับ 1 เท่ากัน

แถวที่ t_2, t_3, t_5 มีค่าแอททริบิวท์ A เท่ากับ 1 เท่ากัน แล้วบ่งชี้แถวที่ t_2, t_3, t_5 ของแอททริบิวท์ B ซึ่งมีค่าเท่ากับ 0 เท่ากัน

แถวที่ t_4, t_6 มีค่าแอททริบิวท์ A เท่ากับ 3 เท่ากัน แล้วบ่งชี้แถวที่ t_4, t_6 ของแอททริบิวท์ B ซึ่งมีค่าเท่ากับ 2 เท่ากัน

ดังนั้น แอททริบิวท์ A บ่งชี้แอททริบิวท์ B เป็นฟังก์ชันการขึ้นต่อกัน

ตารางที่ 2.1 ตารางฐานข้อมูลเชิงสัมพันธ์

Tuple No.	A	B	C
t_1	0	1	2
t_2	1	0	2
t_3	1	0	2
t_4	3	2	2
t_5	1	0	1
t_6	3	2	1
t_7	0	1	2

ตัวอย่างที่ 2.2 แอททริบิวท์ A บ่งชี้แอททริบิวท์ C ไม่เป็นฟังก์ชันการขึ้นต่อกัน ($A \not\rightarrow C$)

จากตารางที่ 2.1 แสดงให้เห็นว่าแต่ละค่าของแอททริบิวท์ A ไม่สามารถบ่งชี้ค่าของแอททริบิวท์ C ที่สอดคล้องกันเพียง 1 ค่าเสมอ กล่าวคือ

แถวที่ t_1, t_2, t_3, t_4, t_7 มีค่าแอททริบิวท์ C เท่ากับ 2 เท่ากัน แต่แถวที่ t_1, t_2, t_3, t_4, t_7 ของแอททริบิวท์ A มีค่าไม่เท่ากัน เนื่องจาก แถวที่ t_1, t_7 ของแอททริบิวท์ A มีค่าเท่ากับ 0 แถวที่ t_2, t_3 ของแอททริบิวท์ A มีค่าเท่ากับ 1 และแถวที่ t_4 ของแอททริบิวท์ A มีค่าเท่ากับ 3

แถวที่ t_5, t_6 มีค่าแอททริบิวท์ C เท่ากับ 1 เท่ากัน แต่แถวที่ t_5, t_6 ของแอททริบิวท์ A มีค่าไม่เท่ากัน เนื่องจาก แถวที่ t_5 ของแอททริบิวท์ A มีค่าเท่ากับ 1 และแถวที่ t_6 ของแอททริบิวท์ A มีค่าเท่ากับ 3

ดังนั้น แอททริบิวท์ A บ่งชี้แอททริบิวท์ C ไม่เป็นฟังก์ชันการขึ้นต่อกัน

นิยามที่ 2.2 ฟังก์ชันการขึ้นต่อกันน้อยที่สุด (Minimal Functional Dependency)

กำหนดให้ X และ A เป็นเซตแอททริบิวท์ในตารางฐานข้อมูลเชิงสัมพันธ์ R จะกล่าวว่าแอททริบิวท์ X บ่งชี้แอททริบิวท์ A เป็นฟังก์ชันการขึ้นต่อกันน้อยที่สุด ก็ต่อเมื่อ เซตย่อยของเซตแอททริบิวท์ X บ่งชี้แอททริบิวท์ A ไม่เป็นฟังก์ชันการขึ้นต่อกัน เขียนแทนด้วย

$$\forall X' \subset X, X' \not\rightarrow A$$

ตารางที่ 2.2 ตารางฐานข้อมูลเชิงสัมพันธ์

Tuple No.	A	B	C
t_1	0	0	1
t_2	0	1	0
t_3	0	2	2
t_4	0	3	0
t_5	4	1	4
t_6	4	3	2
t_7	0	0	1

ตัวอย่างที่ 2.3 เซตแอททริบิวท์ AB บ่งชี้แอททริบิวท์ C เป็นฟังก์ชันการขึ้นต่อกันน้อยที่สุด

จากตารางที่ 2.2 แสดงให้เห็นว่าแต่ละค่าของเซตแอททริบิวท์ AB จะมีค่าของแอททริบิวท์ C ที่สอดคล้องกันเพียง 1 ค่าเสมอ กล่าวคือ

แถวที่ t_1, t_7 มีค่าเซตแอททริบิวท์ A และ B เท่ากับ 0 และ 0 ตามลำดับ แล้วบ่งชี้แถวที่ t_1, t_7 ของแอททริบิวท์ C ซึ่งมีค่าเท่ากับ 1 เท่ากัน

แถวที่ t_2 มีค่าเซตแอททริบิวท์ A และ B เท่ากับ 0 และ 1 ตามลำดับ แล้วบ่งชี้แถวที่ t_2 ของแอททริบิวท์ C ซึ่งมีค่าเท่ากับ 0 เพียงค่าเดียว

แถวที่ t_3 มีค่าเซตแอททริบิวท์ A และ B เท่ากับ 0 และ 2 ตามลำดับ แล้วบ่งชี้แถวที่ t_3 ของแอททริบิวท์ C ซึ่งมีค่าเท่ากับ 2 เพียงค่าเดียว

แถวที่ t_4 มีค่าเซตแอททริบิวท์ A และ B เท่ากับ 0 และ 3 ตามลำดับ แล้วบ่งชี้แถวที่ t_4 ของแอททริบิวท์ C ซึ่งมีค่าเท่ากับ 0 เพียงค่าเดียว

แถวที่ t_5 มีค่าเซตแอททริบิวท์ A และ B เท่ากับ 4 และ 1 ตามลำดับ แล้วบ่งชี้แถวที่ t_5 ของแอททริบิวท์ C ซึ่งมีค่าเท่ากับ 4 เพียงค่าเดียว

แถวที่ t_6 มีค่าเซตแอททริบิวท์ A และ B เท่ากับ 4 และ 3 ตามลำดับ แล้วบ่งชี้แถวที่ t_6 ของแอททริบิวท์ C ซึ่งมีค่าเท่ากับ 2 เพียงค่าเดียว

นอกจากนี้จากตารางที่ 2.2 แสดงให้เห็นว่าเซตย่อยของเซตแอททริบิวท์ AB ไม่สามารถบ่งชี้แอททริบิวท์ C ได้ กล่าวคือ

แอททริบิวท์ A ซึ่งเป็นเซตย่อยของเซตแอททริบิวท์ AB ไม่สามารถบ่งชี้แอททริบิวท์ C ได้ เนื่องจากแถวที่ t_1, t_2, t_3, t_4 และ t_7 ของแอททริบิวท์ A มีค่าเท่ากับ 0 เท่ากัน แต่แถวที่ t_1, t_7 ของแอททริบิวท์ C มีค่าเท่ากับ 1 แถวที่ t_2, t_4 ของแอททริบิวท์ C มีค่าเท่ากับ 0 และแถวที่ t_3 ของแอททริบิวท์ C มีค่าเท่ากับ 2 ดังนั้นแอททริบิวท์ A ไม่สามารถบ่งชี้แอททริบิวท์ C ได้ ($A \not\rightarrow C$)

แอททริบิวท์ B ซึ่งเป็นเซตย่อยของเซตแอททริบิวท์ AB ไม่สามารถบ่งชี้แอททริบิวท์ C ได้ เนื่องจากแถวที่ t_2, t_5 ของแอททริบิวท์ B มีค่าเท่ากับ 1 เท่ากัน แต่แถวที่ t_2 ของแอททริบิวท์ C มีค่าเท่ากับ 0 และแถวที่ t_5 ของแอททริบิวท์ C มีค่าเท่ากับ 4 ดังนั้นแอททริบิวท์ A ไม่สามารถบ่งชี้แอททริบิวท์ C ได้ ($B \not\rightarrow C$)

ดังนั้น สรุปได้ว่า เซตแอททริบิวท์ AB บ่งชี้แอททริบิวท์ C เป็นฟังก์ชันการขึ้นต่อกันน้อยที่สุด เนื่องจากไม่มีเซตย่อยของเซตแอททริบิวท์ AB บ่งชี้แอททริบิวท์ C ได้

นิยามที่ 2.3 แถวสมมูล (Equivalent Row)

กำหนดให้ $\prod_X$ เป็นเซตแถวสมมูลของแอททริบิวท์ X และ $|\prod_X|$ เป็นจำนวนกลุ่มแถวสมมูลในเซตแถวสมมูลของแอททริบิวท์ X จะกล่าวว่า ทุกๆ คู่แถวใดๆ ที่ t_i และ t_j เป็นแถวสมมูลของแอททริบิวท์ X ก็ต่อเมื่อทุกๆ คู่แถวที่ t_i และ t_j ของแอททริบิวท์ X มีค่าเท่ากัน เขียนแทนด้วย

$$[t_i]_X = \{t_j \in r \mid t_i[A] = t_j[A] \text{ for all } A \in X\}$$

นิยามที่ 2.4 พาร์ติชัน (Partition)

กำหนดให้ X เป็นเซตแอททริบิวท์ใน r พาร์ติชันของเซตแอททริบิวท์ X คือ เซตแถวสมมูลของแอททริบิวท์ X ซึ่งสมาชิกภายในเซตแถวสมมูลของแอททริบิวท์ X จะมีค่าข้อมูลเท่ากัน เขียนแทนด้วย

$$\prod_X = \{[t]_X \mid t \in r\}$$

ตัวอย่างที่ 2.4 พาร์ติชันของแอททริบิวท์ A

จากตารางที่ 2.2 แสดงให้เห็นว่า ค่าข้อมูลของแอททริบิวท์ A มีแถวสมมูล ดังนี้

แถวที่ t_1, t_2, t_3, t_4 และ t_7 มีค่าแอททริบิวท์ A เท่ากับ 0 โดยที่ $[t_1]_A = [t_2]_A = [t_3]_A = [t_4]_A = [t_7]_A = \{t_1, t_2, t_3, t_4, t_7\}$ เป็นแถวสมมูลกัน

แถวที่ t_5, t_6 มีค่าแอททริบิวท์ A เท่ากับ 4 โดยที่ $[t_5]_A = [t_6]_A = \{t_5, t_6\}$ เป็นแถวสมมูลกัน

ดังนั้นพาร์ติชันของแอททริบิวท์ A สรุปได้ว่า $\prod_A = \{\{t_1, t_2, t_3, t_4, t_7\}, \{t_5, t_6\}\}$, $|\prod_A| = 2$

ตัวอย่างที่ 2.5 พาร์ติชันของเซตแอททริบิวท์ AB

จากตารางที่ 2.2 แสดงให้เห็นว่า ค่าข้อมูลของแอททริบิวท์ AB มีแถวสมมูล ดังนี้

แถวที่ t_1, t_7 มีค่าเซตแอททริบิวท์ A และ B เท่ากับ 0 และ 0 ตามลำดับ โดยที่ $[t_1]_{AB} = [t_7]_{AB} = \{t_1, t_7\}$ เป็นแถวสมมูลกัน

แถวที่ t_2 มีค่าเซตแอททริบิวท์ A และ B เท่ากับ 0 และ 1 ตามลำดับ โดยที่ $[t_2]_{AB} = \{t_2\}$ เป็นแถวสมมูลกัน

แถวที่ t_3 มีค่าเซตแอททริบิวท์ A และ B เท่ากับ 0 และ 2 ตามลำดับ โดยที่ $[t_3]_{AB} = \{t_3\}$ เป็นแถวสมมูลกัน

แถวที่ t_4 มีค่าเซตแอททริบิวท์ A และ B เท่ากับ 0 และ 3 ตามลำดับ โดยที่ $[t_4]_{AB} = \{t_4\}$ เป็นแถวสมมูลกัน

แถวที่ t_5 มีค่าเซตแอททริบิวท์ A และ B เท่ากับ 4 และ 1 ตามลำดับ โดยที่ $[t_5]_{AB} = \{t_5\}$ เป็นแถวสมมูลกัน

แถวที่ t_6 มีค่าเซตแอททริบิวท์ A และ B เท่ากับ 4 และ 3 ตามลำดับ โดยที่ $[t_6]_{AB} = \{t_6\}$ เป็นแถวสมมูลกัน

ดังนั้นพาร์ติชันของเซตแอททริบิวท์ AB สรุปได้ว่า $\prod_{AB} = \{\{t_1, t_7\}, \{t_2\}, \{t_3\}, \{t_4\}, \{t_5\}, \{t_6\}\}$ และ $|\prod_{AB}| = 6$

นิยามที่ 2.5 กำหนดให้ X และ A เป็นเซตแอททริบิวท์ในตารางฐานข้อมูลเชิงสัมพันธ์ R จะกล่าวว่าแอททริบิวท์ X บ่งชี้แอททริบิวท์ A เป็นฟังก์ชันการขึ้นต่อกัน ก็ต่อเมื่อ จำนวนกลุ่มแถวสมมูลของเซตแอททริบิวท์ X มีค่าเท่ากับจำนวนกลุ่มแถวสมมูลของเซตแอททริบิวท์ $X \cup A$ แล้วแอททริบิวท์ X บ่งชี้แอททริบิวท์ A เป็นฟังก์ชันการขึ้นต่อกัน เขียนแทนด้วย

$$|\prod_X| = |\prod_{X \cup A}| \Rightarrow X \rightarrow A$$

ตัวอย่างที่ 2.6 พิจารณาตารางที่ 2.3

$|\prod_{CD}| = 5, |\prod_{C}| = 3, |\prod_{D}| = 2$

จากนิยามที่ 2.5 สรุปได้ว่า แอททริบิวท์ C บ่งชี้แอททริบิวท์ D ไม่เป็นฟังก์ชันการขึ้นต่อกัน และแอททริบิวท์ D บ่งชี้แอททริบิวท์ C ไม่เป็นฟังก์ชันการขึ้นต่อกัน

$|\prod_{AD}| = |\prod_{A}| = 4$

จากนิยามที่ 2.5 สรุปได้ว่า แอททริบิวท์ A บ่งชี้แอททริบิวท์ D เป็นฟังก์ชันการขึ้นต่อกัน

ตารางที่ 2.3 ตารางฐานข้อมูลเชิงสัมพันธ์

Tuple No.	A	B	C	D	E
t_1	1	100	1	2	50
t_2	4	101	1	2	50
t_3	1	102	2	2	70
t_4	1	200	1	2	50
t_5	2	101	3	3	100
t_6	2	200	1	3	70
t_7	3	100	3	2	50

นิยามที่ 2.6 การสมมูลของแอททริบิวท์ (Attribute Equivalent)

กำหนดให้ X และ Y เป็นเซตแอททริบิวท์ในตารางฐานข้อมูลเชิงสัมพันธ์ จะกล่าวว่า แอททริบิวท์ X และ Y เป็นแอททริบิวท์สมมูล เขียนแทนด้วย $X \leftrightarrow Y$ ก็ต่อเมื่อ แอททริบิวท์ X บ่งชี้แอททริบิวท์ Y เป็นฟังก์ชันการขึ้นต่อกัน และแอททริบิวท์ Y บ่งชี้แอททริบิวท์ X เป็นฟังก์ชันการขึ้นต่อกันด้วย

จากนิยามที่ 2.6 และคุณสมบัติตามสัจพจน์ของอาร์มสตรอง (Armstrong's Axiom) จะได้ข้อเสนอแทรก 2.1 และข้อเสนอแทรก 2.2 ดังนี้

ข้อเสนอแทรก 2.1 [10] กำหนดให้ X, Y, W และ Z เป็นเซตแอททริบิวท์ในตารางฐานข้อมูลเชิงสัมพันธ์ กล่าวได้ว่า ถ้า $X \leftrightarrow Y$ และ $XW \rightarrow Z$ แล้ว $YW \rightarrow Z$

ข้อเสนอแทรก 2.2 [10] กำหนดให้ X, Y, W และ Z เป็นเซตแอททริบิวท์ในตารางฐานข้อมูลเชิงสัมพันธ์ กล่าวได้ว่า ถ้า $X \leftrightarrow Y$ และ $WZ \rightarrow X$ แล้ว $WZ \rightarrow Y$

ตัวอย่างที่ 2.7 แอททริบิวท์ A และ C เป็นแอททริบิวท์สมมูล (A $\leftrightarrow$ C)

จากตารางที่ 2.4 แสดงให้เห็นว่าแต่ละค่าของแอททริบิวท์ A จะมีค่าของแอททริบิวท์ C ที่สอดคล้องกันเพียง 1 ค่าเสมอ และแต่ละค่าของแอททริบิวท์ C จะมีค่าของแอททริบิวท์ A ที่สอดคล้องกันเพียง 1 ค่าเสมอเช่นกัน กล่าวคือ

แถวที่ t_1, t_2, t_3, t_4 และ t_7 มีค่าแอททริบิวท์ A เท่ากับ 0 เท่ากัน แล้วบ่งชี้แถวที่ t_1, t_2, t_3, t_4 และ t_7 ของแอททริบิวท์ C ซึ่งมีค่าเท่ากับ 2 เท่ากัน

แถวที่ t_5, t_6 มีค่าแอททริบิวท์ A เท่ากับ 4 เท่ากัน แล้วบ่งชี้แถวที่ t_5, t_6 ของแอททริบิวท์ C ซึ่งมีค่าเท่ากับ 1 เท่ากัน

ดังนั้นแอททริบิวท์ A บ่งชี้แอททริบิวท์ C เป็นฟังก์ชันการขึ้นต่อกัน (A $\rightarrow$ C)

แถวที่ t_1, t_2, t_3, t_4 และ t_7 มีค่าแอททริบิวท์ C เท่ากับ 2 เท่ากัน แล้วบ่งชี้แถวที่ t_1, t_2, t_3, t_4 และ t_7 ของแอททริบิวท์ A ซึ่งมีค่าเท่ากับ 0 เท่ากัน

แถวที่ t_5, t_6 มีค่าแอททริบิวท์ C เท่ากับ 1 เท่ากัน แล้วบ่งชี้แถวที่ t_5, t_6 ของแอททริบิวท์ A ซึ่งมีค่าเท่ากับ 4 เท่ากัน

ดังนั้นแอททริบิวท์ C บ่งชี้แอททริบิวท์ A เป็นฟังก์ชันการขึ้นต่อกัน (C $\rightarrow$ A)

เนื่องจากแอททริบิวท์ A บ่งชี้แอททริบิวท์ C เป็นฟังก์ชันการขึ้นต่อกัน (A $\rightarrow$ C) และแอททริบิวท์ C บ่งชี้แอททริบิวท์ A เป็นฟังก์ชันการขึ้นต่อกัน (C $\rightarrow$ A) ด้วย ดังนั้น สรุปได้ว่าแอททริบิวท์ A และ C เป็นแอททริบิวท์สมมูล (A $\leftrightarrow$ C)

ตารางที่ 2.4 ตารางฐานข้อมูลเชิงสัมพันธ์

Tuple No.	A	B	C
t_1	0	0	2
t_2	0	1	2
t_3	0	2	2
t_4	0	3	2
t_5	4	1	1
t_6	4	3	1
t_7	0	0	2

นิยามที่ 2.7 เซตการปิดของแอททริบิวท์ภายใต้เซตของฟังก์ชันการขึ้นต่อกัน

กำหนดให้ F เป็นเซตฟังก์ชันการขึ้นต่อกัน และ U เป็นเซตแอททริบิวท์ เซตการปิดของแอททริบิวท์ X (X_F^+) ซึ่ง $X \subseteq U$ ประกอบด้วยแอททริบิวท์ X และเซตแอททริบิวท์ที่ถูกบ่งชี้โดยแอททริบิวท์ X เขียนแทนด้วย

$$X_F^+ = X \cup \{A \in U\text{-}X \mid F \models X \rightarrow A\}$$

นิยามที่ 2.8 เซตการปิดของแอททริบิวท์ภายใต้เซตของฟังก์ชันการขึ้นต่อกันน้อยที่สุด

กำหนดให้ F_m เป็นเซตฟังก์ชันการขึ้นต่อกันน้อยที่สุด และ U เป็นเซตแอททริบิวท์ เซตการปิดของแอททริบิวท์ X (X_{Fm}^+) ซึ่ง $X \subseteq U$ ประกอบด้วยแอททริบิวท์ X และเซตแอททริบิวท์ที่ถูกบ่งชี้โดยเซตย่อยของเซตแอททริบิวท์ X เขียนแทนด้วย

$$X_{Fm}^+ = X \cup U \{A \in U \mid X' \rightarrow A \in F_m, X' \subseteq X\}$$

นิยามที่ 2.9 การปิดแบบกึ่งของแอททริบิวท์ (Attribute Quasi-Closure)

การปิดแบบกึ่งของแอททริบิวท์ X ($X_r^{\Diamond}$) ประกอบด้วยเซตแอททริบิวท์ X และเซตการปิดของเซตแอททริบิวท์ X ซึ่งเป็นเซตที่ใหญ่ที่สุด และไม่เป็นเซตย่อยของเซตใดๆ เขียนแทนด้วย

$$X_r^{\Diamond} = X \cup U_{A \in X} (X - A)^+$$

ตัวอย่างที่ 2.8 การปิดแบบกึ่งของแอททริบิวท์ X ($X^{\Diamond}{}_r$) และการปิดของแอททริบิวท์ X ($X^+{}_r$) โดยกำหนดให้ X = (A, B)

จากตารางที่ 2.3 และจากนิยามที่ 2.5 สรุปได้ดังนี้

$(AB)^{\Diamond}{}_r$ = {A, B, D}

$(AB)^+{}_r$ = {A, B, D, C, E}

นิยามที่ 2.10 เซตการปิดแบบสมบูรณ์ภายใต้เซตของฟังก์ชันการขึ้นต่อกัน (Completely Nontrivial Closure)

กำหนดให้ F เป็นเซตของฟังก์ชันการขึ้นต่อกัน X และ Y เป็นเซตแอททริบิวท์ในตารางฐานข้อมูลเชิงสัมพันธ์ R จะกล่าวว่า เซตแอททริบิวท์ X และ Y เป็นเซตการปิดแบบสมบูรณ์ของ

แอททริบิวท์ X เขียนแทนด้วย X_r^* ก็ต่อเมื่อ เซตแอททริบิวท์ X และ Y ไม่มีสมาชิกซ้อนทับกันหรือกล่าวว่า $X \cap Y = \phi$ เขียนแทนด้วย

$$X^* = \{A \in R, r \models X \rightarrow A \mid X \cap A = \phi\}$$

ตัวอย่างที่ 2.9 การปิดแบบสมบูรณ์ของแอททริบิวท์ X (X_r^*) โดยกำหนดให้ X = (A, B)

จากตัวอย่างที่ 2.8 และจากข้อสังเกต สรุปได้ดังนี้

$(AB)_r^* = X_r^+ - X_r^{\Diamond} = \{C, E\}$

2.2 คุณสมบัติของฟังก์ชันการขึ้นต่อกัน

คุณสมบัติ 2.1 คุณสมบัติตามสัจพจน์ของอาร์มสตรอง (Armstrong's Axiom) มีดังนี้

1) กฎการสะท้อน (Reflexivity Rule)

   ถ้า $Y \subseteq X$ แล้ว $X \rightarrow Y$

2) กฎการขยาย (Augmentation Rule)

   ถ้า $X \rightarrow Y$ แล้ว $XZ \rightarrow YZ$

3) กฎการถ่ายทอด (Transitive Rule)

   ถ้า $X \rightarrow Y$ และ $Y \rightarrow Z$ แล้ว $X \rightarrow Z$

4) กฎการบ่งชี้ตัวเอง (Self Determination Rule)

   $A \rightarrow A$

5) กฎการยูเนียนเพิ่ม (Additivity Union Rule)

   ถ้า $X \rightarrow Y$ และ $X \rightarrow Z$ แล้ว $X \rightarrow YZ$

6) กฎการแตก (Projectivity Decomposition Rule)

   ถ้า $X \rightarrow YZ$ แล้ว $X \rightarrow Y$ และ $X \rightarrow Z$

7) กฎการถ่ายทอดแบบเทียม (Pseudo Transitive Rule)

   ถ้า $X \rightarrow Y$ และ $YW \rightarrow Z$ แล้ว $XW \rightarrow Z$

คุณสมบัติ 2.2 กำหนดให้ $\prod_X$ และ $\prod_Y$ เป็นพาร์ติชันของเซตแอททริบิวท์ X และ Y ในตารางฐานข้อมูลเชิงสัมพันธ์ จะกล่าวว่า แอททริบิวท์ X บ่งชี้แอททริบิวท์ Y เป็นฟังก์ชันการขึ้นต่อกัน ก็ต่อเมื่อ $|\prod_X| = |\prod_{XY}|$

คุณสมบัติ 2.3 ถ้า $|X|_r = |r|$ แล้วเซตแอททริบิวท์ X เป็นคีย์ของตารางฐานข้อมูลเชิงสัมพันธ์

คุณสมบัติ 2.4 ถ้า $X_r^* \cup X_r^{\Diamond} = R$ แล้วเซตแอททริบิวท์ X เป็นคีย์ของตารางฐานข้อมูลเชิงสัมพันธ์

คุณสมบัติ 2.5 กำหนดให้ X, Y เป็นเซตแอททริบิวท์ในตารางฐานข้อมูลเชิงสัมพันธ์ R และ S เป็นค่าข้อมูลของเซตแอททริบิวท์ในตารางฐานข้อมูลเชิงสัมพันธ์ โดยที่ $XY \subseteq S \subseteq R$ จะกล่าวว่า ถ้าแอททริบิวท์ X บ่งชี้แอททริบิวท์ Y เป็นฟังก์ชันการขึ้นต่อกันใน S แล้วแอททริบิวท์ X บ่งชี้แอททริบิวท์ Y เป็นฟังก์ชันการขึ้นต่อกันใน R ด้วย

2.3 งานวิจัยที่เกี่ยวข้อง

Huhtala และคณะ [1, 2] เสนอขั้นตอนวิธี TANE โดยแก้ไขปัญหาวิธีเดิมในการจัดเรียงแถวข้อมูล แล้วจึงนำค่าข้อมูลมาเปรียบเทียบกับแถวข้อมูลทั้งหมดซึ่งเป็นวิธีที่ไม่มีประสิทธิภาพ ดังนั้นขั้นตอนวิธี TANE จึงได้ปรับปรุงโดยใช้แถวสมมูล (Equivalence Class) แล้วแบ่งพาร์ติชันแถวสมมูลของเซตแอททริบิวท์ ทำให้การสืบค้นการขึ้นต่อกันมีประสิทธิภาพดีขึ้น และนิยามกฎที่ใช้ในการตัดเซตแอททริบิวท์ที่ซ้ำซ้อนโดยตัดเซตแอททริบิวท์ในการหาคำตอบก็ต่อเมื่อเซตแอททริบิวท์นั้นเป็นซุปเปอร์เซต (Super Set) ของเซตคำตอบที่อยู่ข้างซ้าย (Left Hand Side: LHS) ยกตัวอย่างเช่น ถ้า $AC \longrightarrow B$ เป็นฟังก์ชันการขึ้นต่อกัน แล้ว $ACD \longrightarrow B$ และ $ACDE \longrightarrow B$ เป็นฟังก์ชันการขึ้นต่อกันที่ได้จากการอนุมานจาก $AC \longrightarrow B$ ดังนั้น สรุปได้ว่าแอททริบิวท์ ACD และ ACDE เป็นแอททริบิวท์ที่ซ้ำซ้อน และสามารถตัดออกจากโครงข่ายคำตอบได้ นอกจากนี้ยังตัดเซตแอททริบิวท์ที่เป็นคีย์ของตารางฐานข้อมูลเชิงสัมพันธ์ ซึ่งขั้นตอนวิธี TANE จะทำการสืบค้นฟังก์ชันการขึ้นต่อกันด้วยการค้นหาแบบกว้าง (Level-wise Search)

Lopes และคณะ [3] เสนอขั้นตอนวิธี Dep-Miner ซึ่งใช้แนวคิดของเซตสอดคล้อง (Agree Set) และเซตมากที่สุด (Maximal Set) กล่าวคือ จะพิจารณาคู่ของทูเพิล ซึ่งจะเริ่มจากการแบ่งพาร์ติชันแถวสมมูล แล้วทำการสตริปพาร์ติชัน (Strip Partition) ที่มีขนาดตั้งแต่ 2 ขึ้นไป เพื่อหาเซตสอดคล้องและสร้างเซตมากที่สุดในการสืบค้นฟังก์ชันการขึ้นต่อกัน และใช้วิธีการค้นหาแบบกว้าง

Wyss และคณะ [9] เสนอขั้นตอนวิธี FastFDs ซึ่งใช้การค้นหาแนวลึก (Depth-First Search) โดยสร้างและท่องไปตามโครงสร้างต้นไม้ ขั้นตอนวิธีนี้จะเริ่มจากการหาเซตของทูเพิลที่แตกต่างกัน (Difference Set) เพื่อใช้หาการปกคลุมน้อยที่สุด (Minimal Cover) ผลการทดลองพบว่า การสืบค้นแนวลึกโดยใช้การปกคลุมน้อยที่สุดจะมีประสิทธิภาพสูงกว่าการสืบค้นแบบกว้าง โดยเปรียบเทียบกับขั้นตอนวิธี TANE และขั้นตอนวิธี Dep-Miner

Novelli และคณะ [7, 8] เสนอขั้นตอนวิธี FUN โดยนำเสนอแนวคิดใหม่ที่เรียกว่าเซตอิสระ (Free Set) มาใช้ในการสืบค้นฟังก์ชันการขึ้นต่อกัน กล่าวคือ เซตแอททริบิวท์ X จะเป็นเซตอิสระก็ต่อเมื่อ เซตย่อยของเซตแอททริบิวท์ X มีขนาดไม่เท่ากับขนาดของเซตแอททริบิวท์ X ยกตัวอย่างเช่น |CD| = 5, |C | = 3, |D| = 2 ดังนั้นเซตแอททริบิวท์ CD จึงเป็นเซตอิสระเพื่อนำไปสืบค้นฟังก์ชันการขึ้นต่อกัน ในทางกลับกันถ้า |AD| = |A| = 4 สรุปได้ว่า A $\longrightarrow$ D เป็นฟังก์ชันการขึ้นต่อกัน และเซตแอททริบิวท์ AD ไม่เป็นเซตอิสระ ดังนั้นเซตแอททริบิวท์ AD จึงถูกตัดออกจากเซตแอททริ-บิวท์ที่ต้องตรวจสอบฟังก์ชันการขึ้นต่อกันในระดับถัดไป นอกจากแนวคิดดังกล่าว ขั้นตอนวิธี FUN ได้นำการปิดของแอททริบิวท์ (Attribute Closure) และการปิดแบบกึ่งของแอททริบิวท์ (Attribute Quasi-Closure) มาใช้ในการสืบค้นฟังก์ชันการขึ้นต่อกันด้วยวิธีการค้นหาแบบกว้าง (Level-wise Search) การทดสอบประสิทธิภาพของขั้นตอนวิธีนี้ทำโดยพัฒนาโปรแกรมเพื่อการทดสอบขั้นตอนวิธี แล้วทำการทดลองกับชุดข้อมูลสังเคราะห์ เพื่อเปรียบเทียบเวลาการประมวลผล (Execution Time) และเปรียบเทียบประสิทธิภาพกับขั้นตอนวิธี TANE และขั้นตอนวิธี Dep-Miner ผลการทดลองพบว่า ขั้นตอนวิธี FUN มีประสิทธิภาพเชิงเวลาสูงกว่าขั้นตอนวิธี TANE และขั้นตอนวิธี Dep-Miner

Yao และคณะ [10, 11] เสนอขั้นตอนวิธี FD_Mine โดยนำเสนอนิยามการสมมูลของแอท-ทริบิวท์ เพื่อใช้ในการตัดเซตแอททริบิวท์ที่ซ้ำซ้อน และลดจำนวนครั้งในการตรวจสอบฟังก์ชันการขึ้นต่อกัน ยกตัวอย่างเช่น ถ้า A $\longleftrightarrow$ D เป็นแอททริบิวท์สมมูล แล้ว AB $\longrightarrow$ E เป็นฟังก์ชันการขึ้นต่อกัน ดังนั้น สรุปได้ว่า DB $\longrightarrow$ E เป็นฟังก์ชันการขึ้นต่อกันด้วย ขั้นตอนวิธีนี้ทำการสืบค้นโดยใช้การค้นหาแบบกว้าง การทดลองทำโดยพัฒนาโปรแกรม แล้วทดลองกับชุดข้อมูลมาตรฐาน ผลการทดลองพบว่า ขั้นตอนวิธี FD_Mine มีจำนวนครั้งในการตรวจสอบฟังก์ชันการขึ้นต่อกันน้อยกว่าขั้นตอนวิธี TANE

บทที่ 3
ขั้นตอนวิธีการสืบค้นฟังก์ชันการขึ้นต่อกัน

ในบทที่ 3 นี้เป็นการนำเสนอขั้นตอนวิธีการสืบค้นฟังก์ชันการขึ้นต่อกัน โดยแบ่งเนื้อหาออกเป็น 2 ส่วน ดังนี้ ส่วนแรกกล่าวถึงขั้นตอนวิธีการสืบค้นฟังก์ชันการขึ้นต่อกัน ส่วนที่ 2 เป็นการวิเคราะห์การทำงานของขั้นตอนวิธี

3.1 ขั้นตอนวิธี

ขั้นตอนวิธีการสืบค้นฟังก์ชันการขึ้นต่อกันโดยใช้การสมมูลและการปิดของแอททริบิวท์ แล้วทำการค้นหาคำตอบจากโครงข่ายคำตอบสองทิศทาง และนำคุณสมบัติการขยายของฟังก์ชันการขึ้นต่อกันช่วยในการตัดเซตแอททริบิวท์ที่ไม่เป็นฟังก์ชันการขึ้นต่อกัน ซึ่งจะลดจำนวนเซตแอททริบิวท์ที่ต้องตรวจสอบฟังก์ชันการขึ้นต่อกันออกจากโครงข่ายคำตอบจะได้ขั้นตอนวิธีที่มีประสิทธิภาพ โดยในการค้นหาคำตอบจากด้านบนของโครงข่ายคำตอบเป็นการค้นหาเพื่อหาเซตแอททริบิวท์ที่เป็นฟังก์ชันการขึ้นต่อกัน ส่วนการค้นหาคำตอบจากด้านล่างของโครงข่ายคำตอบเป็นการค้นหาเพื่อหาเซตแอททริบิวท์ที่ไม่เป็นฟังก์ชันการขึ้นต่อกัน ซึ่งขั้นตอนวิธีตามแนวคิดนี้แสดงได้ดังนี้

Algorithm: Bidirectional Search

Input: relation instance over R

Output: Minimal Functional Dependencies

```
1    L0 := ϕ, i := 1, n = number of candidate attributes, L1 := {{A} / A ∈ R}
2    while (i ≤ ⌊n/2⌋) do
3    {
4        Generate Candidate Set Level i and Prune
5        if Candidate Set = {}
6        {
7            go to step 14
8        } else
9        {
10           ComputeQuasiClosure,
11           ComputeCompleteNonTrivialClosure,
12           ObtainFDandNFD,
13           ComputeEQSet;
14           if i ≠ n – i
15           {
16               Generate Candidate Set Level n-i and Prune
17               if Candidate Set = {}
18               {
19                   go to step 28
20               } else
21               {
22                   ComputeQuasiClosure,
23                   ComputeCompleteNonTrivialClosure,
24                   ObtainFDandNFD,
25                   ComputeEQSet;
26               }
27           }
28       } i = i + 1
29   }
```

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

ขั้นตอนวิธีนี้ประกอบด้วยส่วนย่อย 6 ส่วน ดังนี้

1. การสร้างเซตแอททริบิวท์ระดับ i (Generate Candidate Set Level i)
2. การตัด (Prune)
3. การหาเซตการปิดแบบกึ่งของแอททริบิวท์ (ComputeQuasiClosure)
4. การหาเซตการปิดแบบสมบูรณ์ของแอททริบิวท์ (ComputeCompleteNonTrivial-Closure)
5. การเก็บผลลัพธ์จากการทดสอบฟังก์ชันการขึ้นต่อกัน (ObtainFDandNFD)
6. การหาแอททริบิวท์สมมูล (ComputeEQSet)

3.1.1 การสร้างเซตแอททริบิวท์ระดับ i

การสร้างเซตแอททริบิวท์ระดับ i เป็นการนำเซตแอททริบิวท์แต่ละตัวในตารางฐานข้อมูลเชิงสัมพันธ์มาประกอบรวมกันเป็นเซตแอททริบิวท์ที่มีจำนวนแอททริบิวท์เท่ากับ i ตัว

ตัวอย่างที่ 3.1 กำหนดให้ R = {A, B, C, D} สามารถสร้างเซตแอททริบิวท์แต่ละระดับได้ดังนี้

เซตแอททริบิวท์ระดับที่ 0: {}

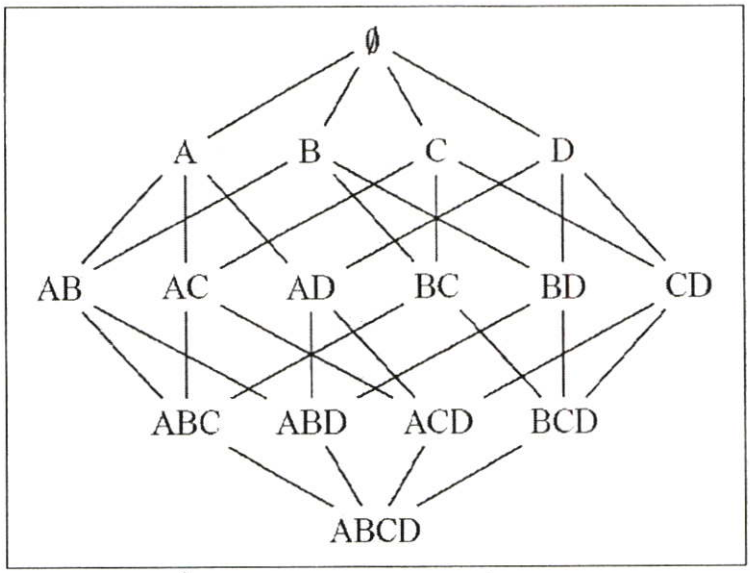
เซตแอททริบิวท์ระดับที่ 1: {A, B, C, D}

เซตแอททริบิวท์ระดับที่ 2: {AB, AC, AD, BC, BD, CD}

เซตแอททริบิวท์ระดับที่ 3: {ABC, ABD, ACD, BCD}

เซตแอททริบิวท์ระดับที่ 4: {ABCD}

แสดงดังรูปที่ 3.1



รูปที่ 3.1 แสดงเซตแอททริบิวท์ระดับต่างๆ

Procedure GenerateNextLevelCandidates(Candidate_Set)

{

$L = \phi$

for each $\{Y, Z\} \in$ Candidate_Set, $Y \neq Z$ do

$X = Y \cup Z$

}

3.1.2 การตัด

การตัดเซตแอททริบิวท์ที่ซ้ำซ้อนมีกฎดังนี้

1) ถ้าเซตแอททริบิวท์เป็นซุปเปอร์เซต (Super Set) ของเซตฟังก์ชันการขึ้นต่อกัน

2) ถ้าแอททริบิวท์ X และ Y เป็นแอททริบิวท์สมมูล กล่าวคือ ถ้า $X \rightarrow Y$ และ $Y \rightarrow X$ แล้ว $X \rightarrow Y$ จากคุณสมบัตินี้จึงตัดแอททริบิวท์ Y ออกจากโครงข่ายคำตอบ

3) ถ้าแอททริบิวท์ X เป็นคีย์ของตารางฐานข้อมูล จะตัดแอททริบิวท์ X ออกจากโครงข่ายคำตอบได้

Procedure Prune(Candidate_Set)

{

for each $X \in L_i$ do

{

if $\exists Y \rightarrow Z \in$ FD_Set then

Delete X from Candidate_Set

if $\exists X \in \{Y \mid X \leftrightarrow Y\} \in$ EQ_Set then

Delete Y from Candidate_Set

if $\exists X \in$ Key_Set then

Delete X from Candidate_Set

}

}

3.1.3 การหาเซตการปิดแบบกึ่งของแอททริบิวท์

การหาเซตการปิดแบบกึ่งของแอททริบิวท์ จะประกอบด้วยเซตแอททริบิวท์ X ณ ระดับที่กำลังทดสอบฟังก์ชันการขึ้นต่อกัน รวมกับเซตการปิดมากที่สุดของแอททริบิวท์ X ถ้า $X_i^* \cup X_i^\Diamond$ มีจำนวนแอททริบิวท์เท่ากับจำนวนแอททริบิวท์ในตารางฐานข้อมูลเชิงสัมพันธ์ R จะอนุมานได้ว่าแอททริบิวท์ X เป็นคีย์ของตารางฐานข้อมูลเชิงสัมพันธ์

Procedure ComputeQuasiClosure(Candidate_Set)

{

for each $X \in L_i$ do

$X_i^\Diamond = X_i^\Diamond \cup X_i$

for each $s \subset X_i$ and $s \in L_{i-1}$ do

$X_i^\Diamond = s' \cup s^\Diamond$

If ($|X| = |r|$) or $((X_i^* \cup X_i^\Diamond) = R)$ then

$X_i^* = R - X_i^\Diamond$

Add X_i to Key_Set

}

3.1.4 การหาเซตการปิดแบบสมบูรณ์ของแอททริบิวท์

การหาเซตการปิดแบบสมบูรณ์ของแอททริบิวท์ X กล่าวคือ กำหนดให้ X เป็นเซตแอททริบิวท์ และ Y เป็นเซตย่อยของแอททริบิวท์ $R\backslash X_i^{\Diamond}$ แอททริบิวท์ X บ่งชี้แอททริบิวท์ Y เป็นฟังก์ชันการขึ้นต่อกัน ($X \rightarrow Y$) ก็ต่อเมื่อ ขนาดพาร์ติชันของเซตแอททริบิวท์ X เท่ากับขนาดพาร์ติชันของเซตแอททริบิวท์ XY ($|\prod_X| = |\prod_{XY}|$) ดังนั้นแอททริบิวท์ Y จึงเป็นเซตการปิดแบบสมบูรณ์ของแอททริบิวท์ X เพื่อนำแอททริบิวท์ Y ทดสอบการเป็นฟังก์ชันการขึ้นต่อกัน

```
Procedure ComputeCompleteNonTrivialClosure(Candidate_Set)
{
        for each Y ⊂ R-X_i^◊ do
                if ∃Y ∉ {Y | s' ↛ Y} ∈ NFD_Set then
                    if X_i ∉ Key_Set then
                        if |∏_X| = |∏_XY| then
                            X_i^* = X_i^* ∪ Y
}
```

3.1.5 การเก็บผลลัพธ์จากการทดสอบฟังก์ชันการขึ้นต่อกัน

การเก็บผลลัพธ์จากการทดสอบฟังก์ชันการขึ้นต่อกัน เป็นการหาฟังก์ชันการขึ้นต่อกันของเซตแอททริบิวท์ X กล่าวคือ เซตแอททริบิวท์ X บ่งชี้แอททริบิวท์ Y เป็นฟังก์ชันการขึ้นต่อกัน ($X \rightarrow Y$) โดยแอททริบิวท์ Y เป็นเซตย่อยของเซตแอททริบิวท์ X^*

```
Procedure ObtainFDandNFD(Candidate_Set)
{
        If Candidate_Set is FD
                Add it to FD_Set
        Else
                Add it to NFD_Set
}
```

3.1.6 การหาแอททริบิวท์สมมูล

การหาเซตแอททริบิวท์สมมูลเป็นการนำเซตฟังก์ชันการขึ้นต่อกันมาทำการทดสอบ $X \subseteq Y^+$ และ $Y \subseteq X^+$ หรือไม่ ถ้าผลการทดสอบเป็นจริงจะเก็บผลลัพธ์ในเซตแอททริบิวท์สมมูล

```
Procedure ComputeEQSet(FD_Set)
{
        E = ϕ
        for each X ∈ L_i do
                for each X → Y ∈ FD_Set do
                    if (X ⊆ Y+ and Y ⊆ X+) then
                        EQ_Set = EQ_Set ∪ {X ↔ Y}
}
```

ตัวอย่างที่ 3.1 จากตารางที่ 2.5 แสดงการทำงานของขั้นตอนวิธีการสืบค้นฟังก์ชันการขึ้นต่อกันจากโครงข่ายคำตอบสองทิศทางด้วยวิธีการค้นหาจากด้านบนสลับด้านล่าง โดยกำหนดให้ตารางฐานข้อมูลเชิงสัมพันธ์ R = {A, B, C, D, E}

ตารางที่ 3.1 ตารางฐานข้อมูลเชิงสัมพันธ์

Tuple No.	A	B	C	D	E
1	0	0	0	2	0
2	0	1	0	2	0
3	0	2	0	2	2
4	0	3	1	2	0
5	4	1	1	1	4
6	4	3	1	1	2
7	0	0	1	2	0

การทำงานของขั้นตอนวิธีมีขั้นตอนดังนี้

1) ขั้นตอนที่ 1 กำหนดให้ i = 1 และ n = จำนวนแอททริบิวท์ในตารางฐานข้อมูลเชิงสัมพันธ์ R ซึ่งในที่นี้ n = 5
2) ขั้นตอนที่ 2 ตรวจสอบ $1 \leq 2$ เป็นจริง ดังนั้นจึงเข้าทำงานในลูป while
3) ขั้นตอนที่ 4 สร้างเซตแอททริบิวท์ในระดับที่ 1 คือ {A, B, C, D, E} และนำเซตแอททริบิวท์ที่สร้างขึ้นมาทำการตัด แต่เนื่องจาก FD_Set = {}, EQ_Set = {} และ Key_Set = {} ดังนั้นจึงไม่ทำการตัดเซตแอททริบิวท์
4) ขั้นตอนที่ 5 ตรวจสอบว่าเซตแอททริบิวท์เป็นเซตว่างหรือไม่ ปรากฏว่าไม่เป็นเซตว่าง ดังนั้นจะทำขั้นตอนที่ 10 ต่อไป
5) ขั้นตอนที่ 10 หาเซตการปิดแบบกึ่งของเซตแอททริบิวท์ ($X^{\Diamond}$) ได้ดังนี้ $A^{\Diamond}$ = {A}, $B^{\Diamond}$ = {B}, $C^{\Diamond}$ = {C}, $D^{\Diamond}$ = {D}, $E^{\Diamond}$ = {E} และตรวจสอบว่าเซตแอททริบิวท์เป็นคีย์หรือไม่ แต่เนื่องจากไม่มีเซตแอททริบิวท์ที่เป็นคีย์ ดังนั้น Key_Set = {}
6) ขั้นตอนที่ 11 หาเซตการปิดแบบสมบูรณ์ของเซตแอททริบิวท์ (X^{*}) ได้ดังนี้ A^{*} = {B, C, D, E}, B^{*} = {A, C, D, E}, C^{*} = {A, B, D, E}, D^{*} = {A, B, C, E}, E^{*} = {A, B, C, D}
7) ขั้นตอนที่ 12 นำเซตแอททริบิวท์ทำการทดสอบฟังก์ชันการขึ้นต่อกัน ปรากฏว่าเซตแอททริบิวท์ A บ่งชี้แอททริบิวท์ D เป็นฟังก์ชันการขึ้นต่อกัน และเซตแอททริบิวท์ D บ่งชี้แอททริบิวท์ A เป็นฟังก์ชันการขึ้นต่อกัน และเก็บผลลัพธ์จากการทดสอบ ดังนี้ FD_Set = {$A \longrightarrow D, D \longrightarrow A$}
8) ขั้นตอนที่ 13 นำเซตแอททริบิวท์ทำการหาเซตแอททริบิวท์สมมูล โดยทำการทดสอบเซตแอททริบิวท์ใน FD_Set ว่าเป็นแอททริบิวท์สมมูลหรือไม่ ปรากฏว่าเซตแอททริบิวท์ A กับเซตแอททริบิวท์ D เป็นแอททริบิวท์สมมูลกัน และเก็บผลลัพธ์จากการทดสอบ ดังนี้ EQ_Set = {A, D}

ระดับที่ 1				
X	$\lvert\prod_X\rvert$	$X^{\Diamond}$	X^{*}	FDs
A	2	A	BCDE	$A \rightarrow D$
B	4	B	ACDE	
C	2	C	ABDE	
D	2	D	ABCE	$D \rightarrow A$
E	3	E	ABCD	

รูปที่ 3.2 แสดงผลการทำงานของขั้นตอนวิธี ณ ระดับที่ 1

FD_Set = {A → D, D → A}

NFD_Set = {}

Key_Set = {}

EQ_Set = {A, D}

จำนวนการตรวจสอบ FDs ณ ระดับที่ 1 = 20 ครั้ง

รูปที่ 3.3 สรุปผลการทำงานของขั้นตอนวิธี ณ ระดับที่ 1

9) ขั้นตอนที่ 14 ทำการตรวจสอบว่า $1 \neq 4$ จริง ดังนั้นทำการสร้างเซตแอททริบิวท์ในขั้นตอนที่ 16 ต่อไปได้

10) ขั้นตอนที่ 16 สร้างเซตแอททริบิวท์ในระดับที่ 4 คือ {ABCD, ABCE, ABDE, ACDE, BCDE} และนำเซตแอททริบิวท์ที่สร้างขึ้นมาทำการตัด ปรากฏว่า FD_Set = {A→D, D→A} ดังนั้น ABCD, ABDE, ACDE, BCDE จะถูกตัด

11) ขั้นตอนที่ 17 ตรวจสอบว่าเซตแอททริบิวท์เป็นเซตว่างหรือไม่ ปรากฏว่าไม่เป็นเซตว่าง ดังนั้นจะทำขั้นตอนที่ 22

12) ขั้นตอนที่ 22 หาเซตการปิดแบบกึ่งของเซตแอททริบิวท์ ($X^{\Diamond}$) ได้ดังนี้ $ABCE^{\Diamond}$ = {ABCDE} แต่เนื่องจาก $ABCE^{\Diamond}$ = R ดังนั้นในการทดสอบฟังก์ชันการขึ้นต่อกันจึงไม่มีเซตแอททริบิวท์ในการทดสอบ

ระดับที่ 4				
X	$\prod X$	$X^{\Diamond}$	X^{*}	FDs
~~ABCD~~	~~7~~	~~ABCD~~		
ABCE	7	ABCDE		
~~ABDE~~	~~6~~	~~ABDE~~		
~~ACDE~~	~~5~~	~~ACDE~~		
~~BCDE~~	~~7~~	~~ABCDE~~		

รูปที่ 3.4 แสดงผลการทำงานของขั้นตอนวิธี ณ ระดับที่ 4

FD_Set = {}

NFD_Set = {}

Key_Set = {}

EQ_Set = {}

จำนวนการตรวจสอบ FDs ณ ระดับที่ 4 = 0 ครั้ง

รูปที่ 3.5 สรุปผลการทำงานของขั้นตอนวิธี ณ ระดับที่ 4

13) ขั้นตอนที่ 28 i = 2 วนกลับไปทำขั้นตอนที่ 2

14) ขั้นตอนที่ 2 ตรวจสอบ $2 \leq 2$ เป็นจริง ดังนั้นจึงเข้าทำงานในลูป while

15) ขั้นตอนที่ 4 สร้างเซตแอททริบิวท์ในระดับที่ 2 คือ {AB, AC, AD, AE, BC, BD, BE, CD, CE, DE} และนำเซตแอททริบิวท์ที่สร้างขึ้นมาทำการตัด ปรากฏว่า FD_Set = {A$\rightarrow$D, D$\rightarrow$A}, EQ_Set = {A, D}, Key_Set = {} ดังนั้น AD, BD, CD, DE จะถูกตัด

16) ขั้นตอนที่ 5 ตรวจสอบว่าเซตแอททริบิวท์เป็นเซตว่างหรือไม่ ปรากฏว่าไม่เป็นเซตว่าง ดังนั้นจะทำขั้นตอนที่ 10 ต่อไป

17) ขั้นตอนที่ 10 หาเซตการปิดแบบกึ่งของเซตแอททริบิวท์ ($X^{\Diamond}$) ได้ดังนี้ $AB^{\Diamond}$ = {ABD}, $AC^{\Diamond}$ = {ACD}, $AE^{\Diamond}$ = {ADE}, $BC^{\Diamond}$ = {BC}, $BE^{\Diamond}$ = {BE}, $CE^{\Diamond}$ = {CE} และตรวจสอบว่าเซตแอททริบิวท์เป็นคีย์หรือไม่ แต่เนื่องจากไม่มีเซตแอททริบิวท์ที่เป็นคีย์ ดังนั้น Key_Set = {}

18) ขั้นตอนที่ 11 หาเซตการปิดแบบสมบูรณ์ของเซตแอททริบิวท์ (X^{*}) ได้ดังนี้ AB^{*} = {CE}, AC^{*} = {BE}, AE^{*} = {BC}, BC^{*} = {AE}, BE^{*} = {AC}, CE^{*} = {AB}

19) ขั้นตอนที่ 12 นำเซตแอททริบิวท์ทำการทดสอบฟังก์ชันการขึ้นต่อกัน ปรากฏว่าเซตแอททริบิวท์ AB บ่งชี้แอททริบิวท์ E เป็นฟังก์ชันการขึ้นต่อกัน เซตแอททริบิวท์ BE บ่งชี้แอท-

ทริบิวท์ A เป็นฟังก์ชันการขึ้นต่อกัน และเซตแอททริบิวท์ CE บ่งชี้แอททริบิวท์ A เป็นฟังก์ชันการขึ้นต่อกัน และเก็บผลลัพธ์จากการทดสอบ ดังนี้ FD_Set = {AB $\rightarrow$ E, BE $\rightarrow$ A, BE $\rightarrow$ D, DB $\rightarrow$ E, CE $\rightarrow$ A, CE $\rightarrow$ D}

20) ขั้นตอนที่ 13 นำเซตแอททริบิวท์ทำการหาเซตแอททริบิวท์สมมูล โดยทำการทดสอบเซตแอททริบิวท์ใน FD_Set ว่าเป็นแอททริบิวท์สมมูลหรือไม่ ปรากฏว่าเซตแอททริบิวท์ AB กับเซตแอททริบิวท์ BE เป็นแอททริบิวท์สมมูลกัน และเก็บผลลัพธ์จากการทดสอบ ดังนี้ EQ_Set = {AB, BE}

ระดับที่ 2				
X	$\lvert\prod X\rvert$	$X^{\Diamond}$	X^{*}	FDs
AB	6	ABD	CE	AB $\rightarrow$ E
AC	3	ACD	BE	
~~AD~~	~~2~~	~~AD~~		
AE	4	ADE	BC	
BC	6	BC	AE	
~~BD~~	~~6~~	~~ABD~~		
BE	6	BE	AC	BE $\rightarrow$ A
~~CD~~	~~3~~	~~ACD~~		
CE	5	CE	AB	CE $\rightarrow$ A
~~DE~~	~~4~~	~~ADE~~		

รูปที่ 3.6 แสดงผลการทำงานของขั้นตอนวิธี ณ ระดับที่ 2

FD_Set = {AB $\rightarrow$ E, BE $\rightarrow$ A, BE $\rightarrow$ D, DB $\rightarrow$ E, CE $\rightarrow$ A, CE $\rightarrow$ D}

NFD_Set = {}

Key_Set = {}

EQ_Set = {AB, BE}

จำนวนการตรวจสอบ FDs ณ ระดับที่ 2 = 12 ครั้ง

รูปที่ 3.7 สรุปผลการทำงานของขั้นตอนวิธี ณ ระดับที่ 2

21) ขั้นตอนที่ 14 ทำการตรวจสอบว่า $2 \neq 3$ จริง ดังนั้นทำการสร้างเซตแอททริบิวท์ในขั้นตอนที่ 16 ต่อไปได้

22) ขั้นตอนที่ 16 สร้างเซตแอททริบิวท์ในระดับที่ 3 คือ {ABC, ABD, ABE, ACD, ACE, ADE, BCD, BCE, BDE, CDE} และนำเซตแอททริบิวท์ที่สร้างขึ้นมาทำการตัด ปรากฏว่า FD_Set = {A $\rightarrow$ D, D $\rightarrow$ A, AB $\rightarrow$ E, BE $\rightarrow$ A, BE $\rightarrow$ D, DB $\rightarrow$ E, CE $\rightarrow$ A, CE $\rightarrow$ D}, EQ_Set = {{A, D}, {AB, BE}}, Key_Set = {} ดังนั้น ABD, ABE, ACD, ACE, ADE, BCD, BDE, CDE จะถูกตัด

23) ขั้นตอนที่ 17 ตรวจสอบว่าเซตแอททริบิวท์เป็นเซตว่างหรือไม่ ปรากฏว่าไม่เป็นเซตว่าง ดังนั้นจะทำขั้นตอนที่ 22

24) ขั้นตอนที่ 22 หาเซตการปิดแบบกึ่งของเซตแอททริบิวท์ ($X^{\Diamond}$) ปรากฏว่าเซตแอททริบิวท์ ABC และ BCE เป็นคีย์ของตาราง และเก็บผลลัพธ์ ดังนี้ Key_Set = {ABC, BCE} และในการทดสอบฟังก์ชันการขึ้นต่อกันไม่มีเซตแอททริบิวท์ในการทดสอบ

ระดับที่ 3				
X	$\prod X$	$X^{\Diamond}$	X^{*}	FDs
ABC	7	ABCDE		
BCE	7	ABCDE		

รูปที่ 3.8 แสดงผลการทำงานของขั้นตอนวิธี ณ ระดับที่ 3

FD_Set = {}

NFD_Set = {}

Key_Set = {ABC, BCE}

EQ_Set = {}

จำนวนการตรวจสอบ FDs ณ ระดับที่ 3 = 0 ครั้ง

รูปที่ 3.9 สรุปผลการทำงานของขั้นตอนวิธี ณ ระดับที่ 3

25) ขั้นตอนที่ 28 i = 3 วนกลับไปทำขั้นตอนที่ 2

26) ขั้นตอนที่ 2 ตรวจสอบ $3 \leq 2$ ไม่เป็นจริง ดังนั้นจึงหยุดการทำงาน

จากการทำงานของขั้นตอนวิธีข้างต้น สรุปผลการทำงานของขั้นตอนวิธีได้ดังนี้

FD_Set = {A → D, D → A, AB → E, BE → A, BE → D, DB → E, CE → A, CE → D}

NFD_Set = {}

Key_Set = {ABC, BCE}

EQ_Set = {{A, D}, {AB, BE}}

จำนวนการตรวจสอบ FDs = 32 ครั้ง

รูปที่ 3.10 สรุปผลการทำงานของขั้นตอนวิธี

3.2 การวิเคราะห์ขั้นตอนวิธี

ความซับซ้อนของเวลา (Time Complexity) ที่ใช้ในการคำนวณของขั้นตอนวิธีขึ้นอยู่กับจำนวนของแอททริบิวท์ n, จำนวนของทูเพิล m และระดับของแอททริบิวท์ที่ถูกสร้างหาคำตอบ กรณีช้าที่สุด (Worst Case) คือ ไม่มีฟังก์ชันการขึ้นต่อกันในตารางฐานข้อมูลเชิงสัมพันธ์ จะมีการสร้างเซตแอททริบิวท์ในการตรวจสอบฟังก์ชันการขึ้นต่อกันทั้งหมด $2^m - 1$ ตัว ดังนั้นความซับซ้อนของเวลาเป็น $O(m2^n)$ สำหรับกรณีดีที่สุด (Best Case) จะมีการสร้างเซตแอททริบิวทในการตรวจสอบฟังก์ชันการขึ้นต่อกันทั้งหมด n ตัว ดังนั้นความซับซ้อนของเวลาเป็น O(n)

บทที่ 4

การทดลอง

จากขั้นตอนวิธีการสืบค้นฟังก์ชันการขึ้นต่อกันโดยใช้การสมมูลและการปิดของแอททริบิวท์ด้วยวิธีการค้นหาคำตอบจากโครงข่ายคำตอบ ซึ่งได้ทำการทดลองด้วยวิธีการค้นหาทั้งหมด 4 วิธี ดังนี้ 1) การค้นหาแบบกว้าง 2) การค้นหาจากด้านล่างด้านบน 3) การค้นหาจากด้านบนสลับด้านล่าง 4) การค้นหาจากด้านล่างสลับด้านบน โดยในบทนี้จะนำเสนอการทดลองและผลการทดลองเพื่อเปรียบเทียบจำนวนเซตแอททริบิวท์ที่ต้องตรวจสอบฟังก์ชันการขึ้นต่อกันจากโครงข่ายคำตอบ

4.1 การทดลอง

4.1.1 ข้อมูลที่ใช้ในการทดลอง

ข้อมูลที่ใช้ในการทดลองเป็นชุดข้อมูลมาตรฐานจาก UCI Machine Learning Repository (ที่มา: http://www1.ics.uci.edu/~mlearn/MLRepository.html) [5] ซึ่งประกอบด้วยชุดข้อมูลทั้งหมด 8 ชุดข้อมูล ดังตารางที่ 4.1

ตารางที่ 4.1 ตารางชุดข้อมูลมาตรฐาน

ชื่อชุดข้อมูล	จำนวนแอททริบิวท์	จำนวนทูเพิล	จำนวน FDs	จำนวน EQs	จำนวนคีย์
Abalone	9	4,177	40	4	1
Balance-scale	5	625	1	0	1
Bridge	13	108	35	2	1
Cancer-Wisconsin	11	699	26	3	2
Chess	7	28,056	1	0	1
Glass	11	214	67	4	1
Iris	5	150	4	0	1
Nursery	9	12,960	1	0	1

4.1.2 เครื่องมือที่ใช้ในการทดลอง

หน่วยประมวลผลกลาง (CPU)	: Intel Pentium 4 processor 1.6 GHz
หน่วยความจำ (RAM)	: 512 MB SD-RAM
หน่วยความจำสำรอง (Hard Disk)	: 20 GB
ระบบปฏิบัติการ (OS)	: Microsoft Windows XP Professional
โปรแกรมที่ใช้ในการพัฒนา	: Microsoft Visual Basic 6.0

4.1.3 การออกแบบการทดลองและเหตุผล

งานวิจัยนี้ได้ทำการทดลองเพื่อเปรียบเทียบจำนวนเซตแอททริบิวท์ที่ต้องตรวจสอบฟังก์ชันการขึ้นต่อกัน โดยทำการทดลองกับชุดข้อมูลมาตรฐานดังกล่าว และแบ่งวิธีการค้นหาคำตอบจากโครงข่ายคำตอบออกเป็น 4 วิธี ดังนี้

วิธีที่ 1 การค้นหาแบบกว้าง

วิธีที่ 2 การค้นหาจากด้านล่างด้านบน

วิธีที่ 3 การค้นหาจากด้านบนสลับด้านล่าง

วิธีที่ 4 การค้นหาจากด้านล่างสลับด้านบน

วิธีที่ 1 ทำการทดลองโดยได้พัฒนาโปรแกรมตามขั้นตอนวิธี FD_Mine แล้วทดลองกับชุดข้อมูลมาตรฐาน เพื่อให้ผลการทดลองสามารถเปรียบเทียบกับขั้นตอนวิธีการค้นหาคำตอบจากโครงข่ายสองทิศทาง

วิธีที่ 2, 3 และ 4 ทำการทดลองโดยได้พัฒนาโปรแกรมตามขั้นตอนวิธีการค้นหาคำตอบจากโครงข่ายคำตอบสองทิศทาง แล้วทดลองกับชุดข้อมูลมาตรฐาน เพื่อให้ผลการทดลองสามารถเปรียบเทียบกับขั้นตอนวิธี FD_Mine และพิจารณาว่าวิธีการค้นหาคำตอบจากโครงข่ายคำตอบสองทิศทางวิธีใด ทำให้ประสิทธิภาพของขั้นตอนวิธีการสืบค้นฟังก์ชันการขึ้นต่อกันโดยใช้การสมมูลและการปิดของแอททริบิวท์มีประสิทธิภาพสูงสุด

4.2 ผลการทดลอง

4.2.1 ผลการทดลอง

การทดลองเพื่อเปรียบเทียบจำนวนเซตแอททริบิวท์ที่ต้องตรวจสอบฟังก์ชันการขึ้นต่อกัน ได้ทำการทดลองด้วยวิธีการค้นหาคำตอบทั้ง 4 วิธีตามหัวข้อ 4.1.3 ซึ่งผลการทดลองแสดงในตารางที่ 4.2 ดังนี้

ตารางที่ 4.2 แสดงจำนวนการตรวจสอบ FDs

ชื่อชุดข้อมูล	**จำนวนการตรวจสอบ FDs**				
	จำนวนการตรวจสอบ FDs ทั้งหมด	การค้นหาแบบกว้าง	การค้นหาจากด้านล่างด้านบน	การค้นหาจากด้านบนสลับด้านล่าง	การค้นหาจากด้านล่างสลับด้านบน
Abalone	2,295	962	442	1,105	833
Balance-scale	75	70	10	35	13
Bridge	53,235	6,146	8,352	6,899	5,684
Cancer-Wisconsin	11,253	4,612	1,575	809	324
Chess	441	434	14	70	19
Glass	11,253	912	4,690	1,321	1,153
Iris	75	70	19	35	19
Nursery	2,295	2,286	18	117	25

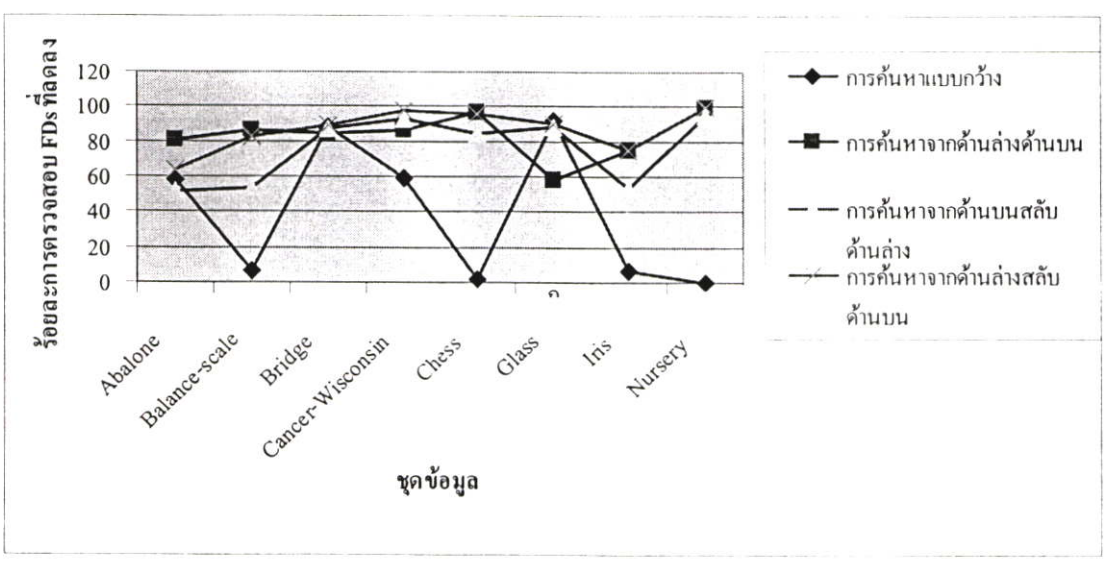
4.2.2 การเปรียบเทียบวิธีการค้นหาคำตอบ

จากตารางที่ 4.2 สามารถนำจำนวนการตรวจสอบฟังก์ชันการขึ้นต่อกันมาทำการคำนวณร้อยละการตรวจสอบฟังก์ชันการขึ้นต่อกันที่ลดลงเมื่อเทียบกับจำนวนการตรวจสอบฟังก์ชันการขึ้นต่อกันทั้งหมด เพื่อหาค่าเฉลี่ยการตรวจสอบฟังก์ชันการขึ้นต่อกันที่ลดลง ซึ่งแสดงได้ดังตารางที่ 4.3

ตารางที่ 4.3 แสดงร้อยละการตรวจสอบ FDs ที่ลดลงเมื่อเทียบกับจำนวนการตรวจสอบ FD ทั้งหมด

ชื่อชุดข้อมูล	ร้อยละการตรวจสอบ FDs ที่ลดลง			
	การค้นหาแบบกว้าง	การค้นหาจากด้านล่างด้านบน	การค้นหาจากด้านบนสลับด้านล่าง	การค้นหาจากด้านล่างสลับด้านบน
Abalone	58.08	80.74	51.85	63.70
Balance-scale	6.67	86.67	53.33	82.67
Bridge	88.45	84.31	87.04	89.32
Cancer-Wisconsin	59.02	86.00	92.81	97.12
Chess	1.59	96.83	84.13	95.69
Glass	91.90	58.32	88.26	89.75
Iris	6.67	74.67	53.33	74.67
Nursery	0.39	99.22	94.90	98.91
ค่าเฉลี่ยการตรวจสอบ FDs ที่ลดลง	39.10	83.35	75.71	86.48

จากตารางที่ 4.3 สามารถสร้างเป็นกราฟเพื่ออธิบายความสัมพันธ์ระหว่างการตรวจสอบฟังก์ชันการขึ้นต่อกันกับวิธีการค้นหาคำตอบจากโครงข่ายคำตอบทั้ง 4 วิธี เมื่อเทียบกับจำนวนการตรวจสอบฟังก์ชันการขึ้นต่อกันทั้งหมด ซึ่งแสดงได้ดังรูปที่ 4.1



รูปที่ 4.1 กราฟแสดงร้อยละการตรวจสอบ FDs ที่ลดลง

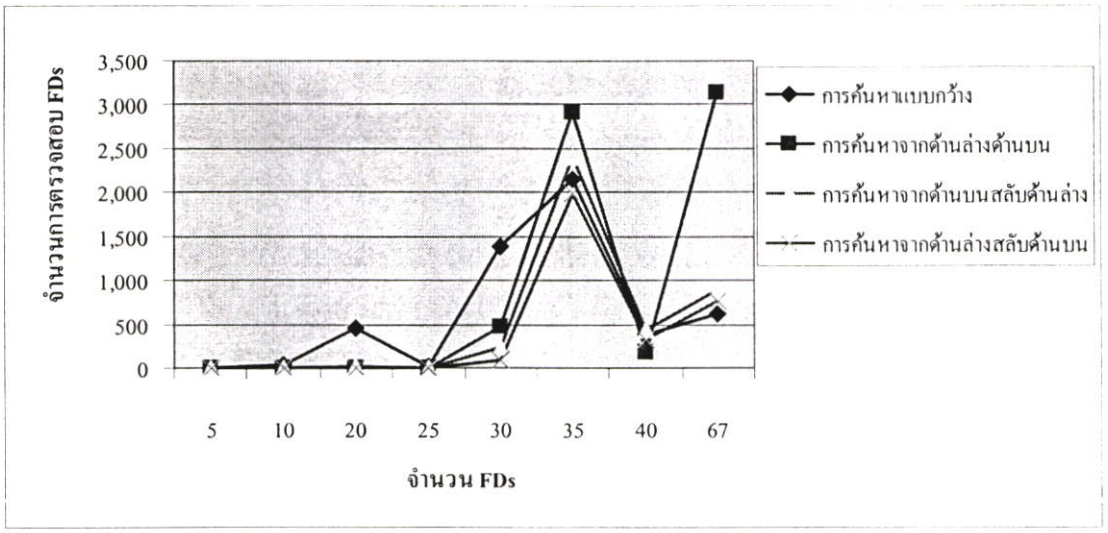
รูปที่ 4.1 แสดงการเปรียบเทียบร้อยละจำนวนการตรวจสอบฟังก์ชันการขึ้นต่อกันของแต่ละวิธีการค้นหาคำตอบ จากรูปดังกล่าวแสดงให้เห็นว่า การสืบค้นฟังก์ชันการขึ้นต่อกันด้วยวิธีการค้นหาคำตอบจากโครงข่ายคำตอบสองทิศทาง โดยเฉลี่ยสามารถลดการตรวจสอบฟังก์ชันการขึ้นต่อกันได้มากกว่าการค้นหาแบบกว้าง และจะเห็นได้ว่าการค้นหาจากด้านล่างสลับด้านบน โดยเฉลี่ยสามารถลดการตรวจสอบฟังก์ชันการขึ้นต่อกันได้มากกว่าการค้นหาจากด้านบนสลับด้านล่าง และการค้นหาจากด้านล่างด้านบน ตามลำดับ

จากตารางที่ 4.1 และ 4.2 สามารถนำจำนวนการตรวจสอบฟังก์ชันการขึ้นต่อกันมาหาค่าเฉลี่ยการตรวจสอบฟังก์ชันการขึ้นต่อกันเมื่อเทียบกับจำนวนฟังก์ชันการขึ้นต่อกันในตารางฐานข้อมูลเชิงสัมพันธ์ เพื่อหาค่าเฉลี่ยการตรวจสอบฟังก์ชันการขึ้นต่อกันสำหรับแต่ละวิธีการค้นหาคำตอบ ซึ่งแสดงได้ดังตารางที่ 4.4

ตารางที่ 4.4 แสดงค่าเฉลี่ยการตรวจสอบ FDs เมื่อเทียบกับจำนวน FDs

ชื่อชุดข้อมูล	**ค่าเฉลี่ยการตรวจสอบ FDs**			
	การค้นหาแบบกว้าง	การค้นหาจากด้านล่างด้านบน	การค้นหาจากด้านบนสลับด้านล่าง	การค้นหาจากด้านล่างสลับด้านบน
Abalone	384.8	176.8	442	333.2
Balance-scale	0.7	0.1	0.35	0.13
Bridge	2,151	2,923	2,415	1,989
Cancer-Wisconsin	1,199	410	210.34	84.24
Chess	4.34	0.14	0.7	0.19
Glass	611.04	3,142	885	773
Iris	2.8	0.76	1.4	0.76
Nursery	23	0.18	1.17	0.25
ค่าเฉลี่ยการตรวจสอบFDs	547.10	831.62	494.46	397.58

จากตารางที่ 4.4 สามารถสร้างเป็นกราฟเพื่ออธิบายความสัมพันธ์ระหว่างการตรวจสอบฟังก์ชันการขึ้นต่อกันกับวิธีการค้นหาคำตอบจากโครงข่ายคำตอบทั้ง 4 วิธี เมื่อเทียบกับจำนวนฟังก์ชันการขึ้นต่อกันในตารางฐานข้อมูลเชิงสัมพันธ์ ซึ่งแสดงได้ดังรูปที่ 4.2



รูปที่ 4.2 กราฟแสดงค่าเฉลี่ยการตรวจสอบ FDs เมื่อเทียบกับจำนวน FDs

รูปที่ 4.2 แสดงการเปรียบเทียบค่าเฉลี่ยการตรวจสอบฟังก์ชันการขึ้นต่อกันของแต่ละวิธีการค้นหาคำตอบ จากรูปดังกล่าวแสดงให้เห็นว่า การสืบค้นฟังก์ชันการขึ้นต่อกันด้วยวิธีการค้นหาจากด้านล่างสลับด้านบน โดยเฉลี่ยสามารถลดการตรวจสอบฟังก์ชันการขึ้นต่อกันได้มากกว่าการค้นหาจากด้านบนสลับด้านล่าง การค้นหาแบบกว้าง และการค้นหาจากด้านล่างด้านบน ตามลำดับ เมื่อเทียบกับจำนวนฟังก์ชันการขึ้นต่อกันในตารางฐานข้อมูลเชิงสัมพันธ์

จากรูปที่ 4.2 แสดงให้เห็นว่าประสิทธิภาพของขั้นตอนวิธีการสืบค้นฟังก์ชันการขึ้นต่อกันขึ้นอยู่กับจำนวนของฟังก์ชันการขึ้นต่อกันในตารางฐานข้อมูลเชิงสัมพันธ์ ซึ่งสามารถเปรียบเทียบและวิเคราะห์ผลการทดลอง ได้ดังนี้

1) ถ้าฟังก์ชันการขึ้นต่อกันมีลักษณะกระจายทั่วทั้งโครงข่ายคำตอบ และมีจำนวนมาก การค้นหาคำตอบแบบสองทิศทางด้วยวิธีการค้นหาจากด้านล่างสลับด้านบน และวิธีการค้นหาจากด้านบนสลับด้านล่าง จะมีประสิทธิภาพสูงกว่าการค้นหาวิธีอื่น โดยวิธีการค้นหาจากด้านล่างสลับด้านบน จะมีค่าเฉลี่ยการตรวจสอบฟังก์ชันการขึ้นต่อกันน้อยกว่าวิธีการค้นหาจากด้านบนสลับด้านล่าง เนื่องจากการค้นหาจากด้านล่างสามารถตัดเซตแอททริบิวท์ที่ไม่เป็นฟังก์ชันการขึ้นต่อกันได้ ซึ่งช่วยลดการตรวจสอบฟังก์ชันการขึ้นต่อกันในการค้นหาจากด้านบน
2) ถ้าฟังก์ชันการขึ้นต่อกันมีจำนวนน้อย การค้นหาคำตอบแบบสองทิศทางด้วยวิธีการค้นหาจากด้านล่างด้านบน จะมีประสิทธิภาพสูงกว่าการค้นหาวิธีอื่น เนื่องจากสามารถตัดเซตแอททริบิวท์ที่ไม่เป็นฟังก์ชันการขึ้นต่อกันได้มาก

3) ถ้าฟังก์ชันการขึ้นต่อกันมีการกระจายบริเวณด้านบนของโครงข่ายคำตอบ และมีจำนวนมาก การค้นหาแบบกว้างจะมีประสิทธิภาพสูงสุด เนื่องจากเซตแอททริบิวท์ที่เป็นคำตอบกระจายบริเวณด้านบนของโครงข่ายคำตอบ

จากตารางที่ 4.1 จะสังเกตเห็นว่า จำนวนของฟังก์ชันการขึ้นต่อกันจะขึ้นอยู่กับจำนวนของแอททริบิวท์ กล่าวคือ ถ้าตารางฐานข้อมูลเชิงสัมพันธ์มีจำนวนแอททริบิวท์มาก ทำให้ข้อมูลในตารางมีจำนวนฟังก์ชันการขึ้นต่อกันมาก ดังนั้นการสืบค้นฟังก์ชันการขึ้นต่อกันจากฐานข้อมูลขนาดใหญ่ จึงเหมาะกับการค้นหาคำตอบแบบสองทิศทางด้วยวิธีการค้นหาจากด้านล่างสลับด้านบน และการค้นหาจากด้านบนสลับด้านล่าง จะมีประสิทธิภาพสูงกว่าการค้นหาแบบกว้าง และการค้นหาจากด้านล่างด้านบน

4.2.3 การเปรียบเทียบกับขั้นตอนวิธี FD_Mine

การเปรียบเทียบประสิทธิภาพการสืบค้นฟังก์ชันการขึ้นต่อกันด้วยวิธีการค้นหาคำตอบแบบสองทิศทาง กับขั้นตอนวิธี FD_Mine ซึ่งใช้วิธีการค้นหาแบบกว้าง สามารถวิเคราะห์ประสิทธิภาพของขั้นตอนวิธีได้จากรูปที่ 4.2 กล่าวคือ ขั้นตอนวิธีการค้นหาคำตอบแบบสองทิศทางด้วยวิธีการค้นหาจากล่างสลับด้านบนและการค้นหาจากด้านบนสลับด้านล่างมีประสิทธิภาพสูงกว่าขั้นตอนวิธี FD_Mine เนื่องจากสามารถลดจำนวนเซตแอททริบิวท์ในการตรวจสอบฟังก์ชันการขึ้นต่อกันได้มากกว่าขั้นตอนวิธี FD_Mine ซึ่งถ้าขนาดฐานข้อมูลมีขนาดใหญ่ขึ้น ข้อมูลในตารางฐานข้อมูลเชิงสัมพันธ์มีจำนวนฟังก์ชันการขึ้นต่อกันมาก การค้นหาคำตอบแบบสองทิศทางจะมีประสิทธิภาพสูงกว่าการค้นหาแบบกว้าง

บทที่ 5
สรุปและข้อเสนอแนะ

5.1 สรุป

งานวิจัยนี้ได้นำเสนอการสืบค้นฟังก์ชันการขึ้นต่อกันโดยใช้การสมมูลและการปิดของแอททริบิวท์ ด้วยวิธีการค้นหาคำตอบจากโครงข่ายคำตอบสองทิศทาง ซึ่งวิธีการค้นหาคำตอบแบบสองทิศทาง มี 3 วิธี ดังนี้ 1) การค้นหาจากด้านล่างด้านบน 2) การค้นหาจากด้านบนสลับด้านล่าง 3) การค้นหาจากด้านล่างสลับด้านบน

จากผลการทดลองเพื่อเปรียบเทียบจำนวนการตรวจสอบฟังก์ชันการขึ้นต่อกัน พบว่า การค้นหาจากด้านล่างสลับด้านบน โดยเฉลี่ยสามารถลดจำนวนเซตแอททริบิวท์ที่ต้องตรวจสอบฟังก์ชันการขึ้นต่อกันได้มากกว่าการค้นหาอีก 2 วิธี และในการเปรียบเทียบกับขั้นตอนวิธี FD_Mine การค้นหาคำตอบแบบสองทิศทางสามารถลดจำนวนเซตแอททริบิวท์ที่ต้องตรวจสอบฟังก์ชันการขึ้นต่อกันได้มากกว่าขั้นตอนวิธี FD_Mine เมื่อฐานข้อมูลมีขนาดใหญ่ และมีจำนวนฟังก์ชันการขึ้นต่อกันมาก ยิ่งสนับสนุนการทำงานของขั้นตอนวิธีด้วยการค้นหาคำตอบแบบสองทิศทาง ซึ่งมีประสิทธิภาพสูงกว่าการค้นหาแบบกว้าง และถ้าฐานข้อมูลมีขนาดเล็ก มีจำนวนฟังก์ชันการขึ้นต่อกันน้อย การค้นหาแบบสองทิศทางด้วยวิธีการค้นหาจากด้านล่างด้านบนจะมีประสิทธิภาพสูงกว่าวิธีการค้นหาวิธีอื่น

5.2 ข้อเสนอแนะ

5.2.1 การสืบค้นฟังก์ชันการขึ้นต่อกันโดยใช้การสมมูลและการปิดของแอททริบิวท์ ด้วยวิธีการค้นหาคำตอบแบบสองทิศทาง ควรศึกษาและพัฒนาให้มีการเรียนรู้เชิงปรับตัว (Adaptive Learning) ซึ่งสามารถวิเคราะห์การสลับทิศทางการค้นหาคำตอบจากโครงข่ายคำตอบได้ ซึ่งจะส่งผลต่อประสิทธิภาพของขั้นตอนวิธีให้มีประสิทธิภาพสูงขึ้น

5.2.2 การสืบค้นฟังก์ชันการขึ้นต่อกันโดยใช้การสมมูลและการปิดของแอททริบิวท์สามารถนำไปประยุกต์ใช้ในการสืบค้นกฎความสัมพันธ์ (Association Rule) ของการทำเหมืองข้อมูลได้

5.2.3 การหาแอททริบิวท์สมมูล ควรศึกษาและพัฒนาให้มีจุดสมดุลในการหาแอททริบิวท์สมมูล เพื่อเพิ่มประสิทธิภาพของขั้นตอนวิธีให้มีประสิทธิภาพสูงขึ้น

เอกสารอ้างอิง

[1] Huhtala, Y., Kärkkäinen, J., Porkka, P. and Toivonen, H., **TANE: An Efficient Algorithm for Discovering Functional and Approximate Dependencies.** *The Computing Journal*, 42(2):100-111 (1999).

[2] Huhtala, Y., Kärkkäinen, J., Porkka, P. and Toivonen, H., **The TANE and TANE/MEM source code.** [Online]. Available: http://www.cs.helsinki.fi/research/fdk/datamining/tane/. 1999.

[3] Lopes, S., Petit, J. M. and Lakhal, L., **Efficient Discovery of Functional Dependencies and Armstrong Relations.** *International Conference on Extending Database Technology (EDBT 2000)*, Konstanz, Germany, 2000, 350-364.

[4] Mannila, H. and Raiha, K. –J., *The Design of Relational Databases*. Addison-Wesley, Menlo Park, California, 1992.

[5] Mannila, H., **Theoretical Frameworks for Data Mining.** *SIGKDD Explorations*, 1(2):30-32 (2000).

[6] Newman, D.J., Hettich, S., Blake, C.L. and Merz, C.J., **UCI Repository of machine learning databases.** [Online]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html. 1998.

[7] Novelli, N. and Cicchetti, R., **FUN: An Efficient Algorithm for Mining Functional and Embedded Dependencies.** *International Conference on Database Theory (ICDT 2001)*, London, UK, 2001, 189-203.

[8] Novelli, N. and Cicchetti, R., **Functional and Embedded Dependency Inference: A Data Mining Point of View.** *Information Systems*, 26(7):477-506 (2001).

[9] Wyss, C., Giannella, C. and Edward, E.L., **FastFDs, A Heuristic-Driven, Depth-First Algorithm for Mining Functional Dependencies from Relation Instances.** *Data Warehousing and Knowledge Discovery*, Third International Conference (DaWaK 2001), Munich, Germany, 2001, 101-110.

[10] Yao, H., Hamilton, H. J. and Butz, C. J., **FD_MINE: Discovering Functional Dependencies in a Database Using Equivalences**, University of Regina, Computer Science Department, Technical Report CS-02-04, August, 2002, ISBN 0-7731-0441-0.

[11] Yao, H. and Hamilton, H. J., **Mining Functional Dependencies from data**, Springer, September, 2007, ISBN 1384-5810.

[12] ดวงใจ วรเศรษฐเมธี, **นิยามใหม่ของรีดักโดยใช้ทฤษฎีฟังก์ชันการขึ้นต่อกัน**, วิทยานิพนธ์วิทยาศาสตรมหาบัณฑิต สาขาวิทยาการคอมพิวเตอร์ บัณฑิตวิทยาลัย, สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง, 2549.

ประวัติผู้เขียน

ชื่อ – สกุล	นางสาวนิภาพร ไฝขาว
วัน เดือน ปีเกิด	29 มิถุนายน พ.ศ. 2522
ที่อยู่	190/8 หมู่ที่ 7 ต.ละแม อ.ละแม จ.ชุมพร 86170
ประวัติการศึกษา	พ.ศ.2545 วิทยาศาสตรบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยทักษิณ
ประวัติการทำงาน	
ปัจจุบัน	นักวิชาการคอมพิวเตอร์ สำนักคอมพิวเตอร์ มหาวิทยาลัยทักษิณ
พ.ศ. 2545 – 2546	อาจารย์สาขาคอมพิวเตอร์ธุรกิจ โรงเรียนโปลีเทคนิคสุราษฎร์ธานี จ.สุราษฎร์ธานี