อัลกอริทึมการปรับปรุงความสัมพันธ์แบบมิติผสมสำหรับการเพิ่มข้อมูล

HYBRID-DIMENSION FAST UPDATE ALGORITHM

สุภาพร ฉัตรเศรษฐกุล
SUPAPORN CHATSETTAKUL

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาเทคโนโลยีสารสนเทศ
คณะเทคโนโลยีสารสนเทศ
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
พ.ศ. 2553
KMITL-2010-IT-M-001-004

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

อัลกอริทึมกฎความสัมพันธ์แบบมิติผสมสำหรับการเพิ่มข้อมูล

HYBRID-DIMENSION FAST UPDATE ALGORITHM

สุภาพร ฉัตรเศรษฐกูล

SUPAPORN CHATSETTAKUL

สขหมู่.....
เลขทะเบียน 110402
วัน,เดือน,ปี - 2 เม.ย. 2553

b.....
i.....

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต

สาขาวิชาเทคโนโลยีสารสนเทศ

คณะเทคโนโลยีสารสนเทศ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ.2553

KMITL-2010-IT-M-001-004

HYBRID-DIMENSION FAST UPDATE ALGORITHM

SUPAPORN CHATSETTAKUL

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

MASTER OF SCIENCE IN INFORMATION TECHNOLOGY

FACULTY OF INFORMATION TECHNOLOGY

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2010

KMITL-2010-IT-M-001-004

COPYRIGHT 2010

FACULTY OF INFORMATION TECHNOLOGY

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

คณะเทคโนโลยีสารสนเทศ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ใบรับรองวิทยานิพนธ์

หัวข้อวิทยานิพนธ์ อัลกอริทึมค้นหากฎความสัมพันธ์แบบมิติผสมสำหรับการเพิ่มข้อมูล

Hybrid Dimension Fast Update Algorithm

นักศึกษา นางสาวสุภาพร ฉัตรเศรษฐกูล

รหัสประจำตัว 48066425

ปริญญา วิทยาศาสตรมหาบัณฑิต

สาขาวิชา เทคโนโลยีสารสนเทศ

อาจารย์ที่ปรึกษาวิทยานิพนธ์ รองศาสตราจารย์ ดร.วรพจน์ กรีสุระเดช

คณะกรรมการสอบวิทยานิพนธ์	ลายมือชื่อ
รองศาสตราจารย์ ดร.พีระพนธ์ โสพัศสถิตย์ รองศาสตราจารย์ ดร.วรพจน์ กรีสุระเดช รองศาสตราจารย์ ดร.อาริต ธรรมโน รองศาสตราจารย์ ดร.บุญธีร์ เครือตราชู ดร. ปานวิทย์ ธุวะนุติ	

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

วัน / เดือน/ ปี ที่สอบ วันพุธที่ 2 มิถุนายน 2553 เวลา 09.30 น.

สถานที่สอบ ณ ห้อง M 22 ชั้น M คณะเทคโนโลยีสารสนเทศ

คณะเทคโนโลยีสารสนเทศรับรองแล้ว

(รองศาสตราจารย์ ดร.จันทร์บูรณ์ สถิตวิริยวงศ์)

คณบดีคณะเทคโนโลยีสารสนเทศ

วันที่ 7 **เดือน** สิงหาคม **พ.ศ.** 2553

สำนักทะเบียนและประมวลผล สจล.
วันที่ส่งเล่มวิทยานิพนธ์ฉบับสมบูรณ์
วันที่ 14 เดือน ธ.ค. พ.ศ. 53
ลงชื่อ.....

Thesis	Hybrid Dimension Fast Update Algorithm
Student	Miss. Supaporn Chatsettakul
Student ID.	48066425
Degree	Master of Science
Programme	Information Technology
Year	2010
Thesis Advisor	Assoc. Prof. Dr. Worapoj Kreesuradej

ABSTRACT

The problem of association rule mining has gained considerable prominence in data mining. Traditional association rule mining is limited to intra-transaction. However, most databases are usually collected in multi-dimension transaction and are dynamic in the sense that users may occasionally insert a new data into the database. Discovering multidimensional association rules from the databases is required to have a new algorithm. This thesis proposes Hybrid Dimension Fast Update algorithm (HDFUP) to deal with such databases. The propose algorithm is based on the concepts of FUP algorithm and a multidimensional join operator. The multidimensional join operator help to reduces the number of 2^{nd}- candidate itemsets. As a result, the propose algorithm has less execution time than that of FUP.

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จได้อย่างดีด้วยคำแนะนำ และคำปรึกษาจาก รศ.ดร.วรพจน์ กรีสุระเดช ซึ่งเป็นอาจารย์ผู้ควบคุมวิทยานิพนธ์ ข้าพเจ้ารู้สึกทราบซึ้งในความอนุเคราะห์จากท่านอาจารย์ และขอขอบพระคุณเป็นอย่างสูง

ขอกราบพระคุณคณาจารย์คณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ทุกท่านที่ได้ประสิทธิ์ประสาทวิชาให้กับข้าพเจ้า

ขอขอบคุณห้องวิจัยและปฏิบัติการ Data Mining and Data Exploration Lab (DME Lab) คณะเทคโนโลยีสารสนเทศ ในการทำวิจัย

ขอขอบคุณ พี่ๆ เพื่อนๆ น้องๆ ในสาขาเทคโนโลยีสารสนเทศ คณะเทคโนโลยีสารสนเทศทุกคนที่ให้คำแนะนำต่างๆ และคอยให้กำลังใจเสมอมา

ขอขอบคุณบัณฑิตศึกษาและบัณฑิตวิทยาลัย คณะเทคโนโลยีสารสนเทศที่ให้ความช่วยเหลือในการช่วยดำเนินเรื่องต่างๆ สำหรับการสอบและจัดทำรูปเล่มวิทยานิพนธ์จนแล้วเสร็จ

สุดท้ายนี้ข้าพเจ้าขอกราบขอบพระคุณ บิดา มารดา และครอบครัวของข้าพเจ้าที่เป็นกำลังใจและให้การสนับสนุนในทุกเรื่องๆ ทำให้ข้าพเจ้าสามารถทำวิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงด้วยดี

คุณค่าและประโยชน์อันพึงมาจากวิทยานิพนธ์ฉบับนี้ ข้าพเจ้าขอมอบแด่ผู้มีพระคุณทุกท่าน

สุภาพร ฉัตรเศรษฐกูล

สารบัญ

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญตาราง

สารบัญรูป

สารบัญรูป (ต่อ)

รูปที่ หน้า

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

การแข่งขันในโลกธุรกิจมีอัตราแนวโน้มไปในทิศทางที่สูงขึ้น ดังนั้นบริษัทต่างๆ จึงมองหาเครื่องมือหรือวิธีการที่จะเข้ามาช่วยให้ตนเป็นผู้นำในการแข่งขัน เทคโนโลยีสารสนเทศนับเป็นตัวเลือกอันดับต้นๆ ที่องค์กรจะนำเข้ามาประยุกต์ใช้ และมีบทบาทในการพัฒนาปรับปรุงระบบการทำงานที่ใช้สำหรับสนับสนุนการตัดสินใจ หรือช่วยวางแผนทางกลยุทธ์ในเชิงธุรกิจ

หัวใจหลักในระบบสารสนเทศก็คือข้อมูลดิบ ซึ่งจะนำไปประมวลผลหาสารสนเทศที่เป็นประโยชน์ หรือทำการสกัดหาความสัมพันธ์ข้อมูลที่ถูกจัดเก็บอยู่ในฐานข้อมูล ซึ่งอาจทำให้ค้นพบรูปแบบการเกี่ยวข้องกันระหว่างข้อมูลที่ไม่เคยทราบมาก่อน ภายในองค์กรธุรกิจอาจจะมีข้อมูลปริมาณมากถูกจัดเก็บเพิ่มเข้ามาใหม่ในฐานข้อมูล และอาจมีการขยายขนาดเพิ่มขึ้นเรื่อยๆ จึงจำเป็นต้องมีการค้นหากฎความสัมพันธ์ เพื่อให้ความสัมพันธ์ที่ได้นั้นเป็นปัจจุบันอยู่เสมอ

การค้นหาความรู้จากฐานข้อมูล (Knowledge Discovery in Database: KDD) เป็นกระบวนการสืบค้นความรู้ (Knowledge) จากกลุ่มข้อมูลดิบที่มีขนาดใหญ่ภายในฐานข้อมูล โดยขั้นตอนหนึ่งในการค้นหาความรู้จากฐานข้อมูล ก็คือขั้นตอนการทำดาต้าไมนิ่ง (Data Mining) และเป็นกระบวนการสำคัญสำหรับค้นหาความรู้จากฐานข้อมูล

ดาต้าไมนิ่งเป็นกระบวนการที่สำคัญในการสกัดแยกข้อมูล (Extract Data) ทำการสำรวจวิเคราะห์ ค้นหาลักษณะรูปแบบความสัมพันธ์ของข้อมูลที่น่าสนใจ หรือข้อมูลที่มีรูปแบบ (Pattern) ซึ่งมีความหมาย มีอยู่จริงในฐานข้อมูลและเชื่อถือได้ รวมถึงสามารถนำมาใช้ประโยชน์ได้ตรงตามความต้องการ เนื่องจากไม่มีเทคนิคใดของดาต้าไมนิ่งจะสามารถตอบคำถามได้ทุกปัญหา จึงทำให้มีเทคนิคสำหรับการทำดาต้าไมนิ่งหลากหลาย การคัดเลือกเทคนิคให้เหมาะสมกับปัญหาจึงเป็นเรื่องสำคัญอย่างยิ่ง

เทคนิคหนึ่งที่สำคัญในการทำดาต้าไมนิ่ง คือการค้นหากฎความสัมพันธ์ (Association Rule Discovery) หลักการทำงานโดยสรุปคือ ค้นหารูปแบบความสัมพันธ์ของข้อมูลจากข้อมูลจำนวนมากในฐานข้อมูลเพื่อนำไปใช้ในการวิเคราะห์ และทำนายคาดการณ์ (Prediction) จัดข้อมูลเหล่านั้นให้อยู่ในรูปแบบของกฎความสัมพันธ์

เทคนิคที่กล่าวมาเป็นที่แพร่หลายสำหรับองค์กรธุรกิจต่างๆ การค้นหากฎความสัมพันธ์ของข้อมูลที่มีอยู่ในฐานข้อมูลจำเป็นต้องทำการค้นหาค่า Large k-itemsets โดยค่า Large k-itemsets เป็นค่าของไอเทมเซต (itemsets) ที่ประกอบไปด้วยจำนวน k item โดยค่า k = 1, 2, 3,.., n อาจจะมี

จำนวนเท่ากับหรือมากกว่าค่าสนับสนุนขั้นต่ำ (Minimum Support) ที่กำหนดไว้ เมื่อได้ค่า Large k-itemsets แล้ว ก็จะนำมาสร้างเป็นกฎความสัมพันธ์ให้อยู่ในรูปแบบกฎ **IF** X **THEN** Y ซึ่งในแต่ละกฎประกอบด้วย 2 ส่วนคือ ส่วนด้านซ้ายของกฎ (Left-hand side) และส่วนด้านขวาของกฎ (Right-hand side) โดยด้านซ้ายมีเงื่อนไขที่เป็นจริง จะทำให้ส่วนด้านขวาของกฎเป็นจริง

ตัวอย่างของกฎความสัมพันธ์ **IF** computer **THEN** antivirus software ความหมายของกฎความสัมพันธ์นี้คือ ถ้าลูกค้าซื้อเครื่องคอมพิวเตอร์แล้ว จะซื้อซอฟต์แวร์แอนตี้ไวรัสไปด้วย ดังนั้น การเปลี่ยนแปลงที่เกิดขึ้นในฐานข้อมูลเมื่อมีทรานแซคชั่นหรือข้อมูลใหม่เพิ่มเข้าไปในฐานข้อมูลอาจทำให้กฎเดิมนี้เปลี่ยนแปลงไป

เมื่อมีข้อมูลเพิ่มจำเป็นต้องทำการค้นหากฎความสัมพันธ์ของฐานข้อมูลใหม่และจำเป็นต้องรักษาความคงอยู่ของกฎให้มีความถูกต้องเสมอ บางครั้งอาจทำให้ Large Itemsets เดิมเปลี่ยนแปลงไปเป็น Small Itemset หรือ Small Itemset เปลี่ยนแปลงไปเป็น Large Itemset จะเห็นได้ว่าการเพิ่มขยายของข้อมูลอาจมีผลต่อ Large k-itemsets เดิม (L_k) และอาจทำให้มีผลต่อการเปลี่ยนแปลงของกฎความสัมพันธ์ ทำให้ต้องดำเนินการค้นหา Large k-itemsets ใหม่ (L_k')

ปัญหาของการค้นหากฎความสัมพันธ์ใหม่ส่วนใหญ่เกิดได้จาก 2 สาเหตุคือ

1. เกิดจากการปรับปรุงข้อมูลในฐานข้อมูลทำให้ต้องมีการคัดเลือกค่า Large Itemsets ขึ้นมาใหม่ทั้งหมด เพื่อให้ผ่านค่าสนับสนุนขั้นต่ำที่กำหนดไว้ ค่าใดที่ผ่านค่าสนับสนุนขั้นต่ำแล้ว ก็จะเป็นค่า Large Itemsets ของฐานข้อมูลปรับปรุงใหม่ (Update Database) และนำค่าที่ได้มาหากฎความสัมพันธ์ใหม่อีกครั้ง

2. เกิดจากการค้นหาข้อมูลซ้ำในฐานข้อมูลเก่า (Original Database) และค้นหาในฐานข้อมูลใหม่ (Increment Database) เพื่อจะปรับปรุงค่า Large Itemsets ใหม่ทั้งหมด อาจทำให้จำนวนรอบการค้นหานั้นมากครั้งขึ้น เพราะไม่ได้มีการนำค่า Large Itemsets ที่ได้จากการค้นหาในฐานข้อมูลเดิมมาใช้ให้เกิดประโยชน์

การค้นหากฎความสัมพันธ์ของข้อมูลสามารถแบ่งได้เป็น 2 ประเภทคือ กฎความสัมพันธ์แบบหนึ่งมิติ (Single-Dimension Association Rules) คือการค้นหากฎความสัมพันธ์ในแอททริบิวต์ (attribute) เดียวกันที่มีอยู่ภายในทรานแซคชั่น และกฎความสัมพันธ์แบบหลายมิติ (Multidimension Association Rules) โดยกฎความสัมพันธ์แบบหลายมิติเป็นการค้นหากฎความสัมพันธ์ได้ทุกแอททริบิวต์ที่มีการจัดเก็บอยู่ในฐานข้อมูล

สำหรับส่วนของการค้นหากฎความสัมพันธ์แบบหลายมิติ ยังสามารถแบ่งได้อีก 2 ประเภท คือ การค้นหากฎความสัมพันธ์แบบระหว่างมิติ (Inter-Dimension Association Rule) ผลลัพธ์ของกฎความสัมพันธ์ที่ได้จะไม่มีการทำนายแอททริบิวต์ที่มีในทรานแซคชั่นซ้ำและ การค้นหากฎความสัมพันธ์แบบมิติผสม (Hybrid-Dimension Association Rules) ผลลัพธ์ของกฎความสัมพันธ์ที่ได้จะสามารถทำนายแอททริบิวต์หลักที่มีในทรานแซคชั่นซ้ำได้ ซึ่งความเป็นจริงการเก็บ

ทรานแซคชั่นข้อมูลในฐานข้อมูลมีการจัดเก็บทรานแซคชั่นข้อมูลแบบหลายมิติ การค้นหากฎความสัมพันธ์แบบมิติเดียวอาจไม่สอดคล้องกับรูปแบบการจัดเก็บในฐานข้อมูล หรือการค้นหาแบบกฎความสัมพันธ์แบบระหว่างมิติไม่สามารถตอบความสัมพันธ์ภายในแอททริบิวต์หลักได้

จึงได้นำการค้นหากฎความสัมพันธ์แบบมิติผสมมาใช้ในการค้นหากฎความสัมพันธ์ของข้อมูล เพื่อให้ได้กฎความสัมพันธ์ที่หลากหลายมากยิ่งขึ้น นำไปปรับใช้ร่วมกับการค้นหากฎความสัมพันธ์สำหรับการเพิ่มขึ้นของข้อมูลในฐานข้อมูล ซึ่งฐานข้อมูลที่องค์กรธุรกิจใช้กันอยู่ในปัจจุบันนั้นมักจะมีข้อมูลเพิ่มเข้าอยู่เสมอ เนื่องจากต้องปรับปรุงข้อมูลให้มีความทันสมัย จึงเล็งเห็นถึงการค้นหากฎความสัมพันธ์แบบมิติผสมในฐานข้อมูลที่มีการจัดเก็บทรานแซคชั่นแบบหลายมิติ เพื่อให้ได้กฎความสัมพันธ์ครบถ้วนและมีความถูกต้อง

1.2 ความมุ่งหมายและวัตถุประสงค์ของการศึกษา

1. ศึกษาการค้นหากฎความสัมพันธ์กรณีที่ฐานข้อมูลมีการเพิ่มข้อมูล
2. ศึกษาการค้นหากฎความสัมพันธ์แบบมิติผสม
3. ปรับปรุงอัลกอริทึมสำหรับการเพิ่มข้อมูลเข้าในฐานข้อมูล ให้มีความสามารถค้นหากฎความสัมพันธ์ของข้อมูล ด้วยวิธีการค้นหากฎความสัมพันธ์แบบมิติผสม

1.3 สมมติฐานของการศึกษา

ในงานวิจัยนี้ได้มุ่งเน้นไปที่การนำแนวคิดเรื่องการค้นหากฎความสัมพันธ์ข้อมูลแบบมิติผสมเพื่อค้นหากฎความสัมพันธ์ในกรณีที่มีการเพิ่มขึ้นของข้อมูลในฐานข้อมูลแบบทรานแซคชั่นหลายมิติ (Multi-Dimensional Transaction Database) โดยมีแนวคิดและทฤษฎีที่น่าสนใจดังนี้

1. Knowledge Discovery in Database
2. Association Rule
3. Hybrid-Dimension Association Rule

1.4 ขอบเขตการวิจัย

วิทยานิพนธ์ฉบับนี้ได้ศึกษาเกี่ยวกับวิธีการที่นำกฎความสัมพันธ์แบบมิติผสม มาใช้ร่วมกับการเพิ่มขึ้นของข้อมูลในฐานข้อมูลแบบทรานแซคชั่นหลายมิติ โดยอัลกอริทึม FUP (Fast Update) เป็นอัลกอริทึมสำหรับใช้ค้นหากฎความสัมพันธ์ของข้อมูลในกรณีเพิ่มข้อมูลเข้าสู่ฐานข้อมูล เพื่อให้สามารถค้นหากฎความสัมพันธ์แบบมิติผสมได้เมื่อมีการเพิ่มขึ้นของข้อมูล จึงได้ปรับปรุงอัลกอริทึม FUP ให้สามารถค้นหากฎความสัมพันธ์แบบมิติผสมได้

1.5 ขั้นตอนของการศึกษา

ขั้นตอนในการศึกษาวิธีการวิจัย จากเริ่มจนถึงสิ้นสุดการทำงานวิจัยดังนี้

1.5.1 ศึกษาทฤษฎีและงานวิจัยจากเอกสาร บทความต่างๆ ในส่วนที่เกี่ยวข้องกับการทำงานวิจัยในฉบับนี้

1.5.2 กำหนดหัวข้อ วัตถุประสงค์ ขอบเขตการทำงานวิจัย

1.5.3 วิเคราะห์และปรับปรุงอัลกอริทึม

1.5.4 เตรียมข้อมูลเพื่อใช้ทดลอง

1.5.5 พัฒนาโปรแกรม ทดสอบ และแก้ไขข้อผิดพลาด

1.5.6 รวบรวมผลการทดลองจากการทำงานของโปรแกรม

1.5.7 วิเคราะห์และสรุปผลการทดลอง

1.5.8 ดำเนินการจัดทำเอกสารงานวิจัย

วิทยานิพนธ์ฉบับนี้ได้แบ่งเนื้อหาทั้งหมดออกเป็น 5 บท คือ

บทที่ 1 ความเป็นมาของงานวิจัย ความมุ่งหมายและวัตถุประสงค์ สมมติฐาน ทฤษฎีที่ใช้ ขอบเขตของการวิจัย และขั้นตอนการศึกษา

บทที่ 2 ทฤษฎีพื้นฐานที่ใช้ในการวิจัย

บทที่ 3 การค้นหากฎความสัมพันธ์แบบมิติผสมสำหรับการเพิ่มข้อมูล

บทที่ 4 วิเคราะห์ผลการทดลอง

บทที่ 5 บทสรุปของงานวิจัย

บทที่ 2
ทฤษฎีพื้นฐานที่ใช้ในการวิจัย

2.1 การค้นหากฎความสัมพันธ์ของข้อมูล (Association Rule Discovery)

การค้นหากฎความสัมพันธ์ของข้อมูลเป็นวิธีหนึ่งสำหรับทำดาต้าไมนิ่ง ซึ่งเป็นวิธีที่ได้รับความสนใจอย่างแพร่หลาย ทั้งในการวิจัยเชิงการศึกษาและประยุกต์ใช้กับองค์กรธุรกิจเพื่อค้นหารูปแบบความสัมพันธ์ระหว่างข้อมูลในฐานข้อมูล จนเกิดเป็นสารสนเทศที่มีประโยชน์หรือค้นพบความสัมพันธ์ระหว่างข้อมูลที่ไม่เคยทราบมาก่อน และนำสารสนเทศที่ได้จากกฎความสัมพันธ์ไปใช้ในกระบวนการตัดสินใจ

การวิเคราะห์ตะกร้าสินค้า (market basket analysis) คือตัวอย่างการค้นหากฎความสัมพันธ์ของข้อมูลที่พบเห็นได้บ่อย เป็นกระบวนการวิเคราะห์พฤติกรรมการซื้อสินค้าของ ผู้ซื้อ โดยหาความสัมพันธ์ระหว่างสินค้าที่แตกต่างชนิดกันในตะกร้าสินค้า

การทำดาต้าไมนิ่งเพื่อค้นหากฎความสัมพันธ์ที่นำเสนอโดย R. Agrawal, T. Imielinski และ A. Swami ในปีค.ศ. 1993[1] โดยมีเป้าหมายเพื่อทำการค้นหาความน่าสนใจของไอเทมในกลุ่มข้อมูลของทรานแซคชั่นในฐานข้อมูลขนาดใหญ่

กฎความสัมพันธ์เป็นเทคนิคที่ใช้สำหรับค้นหากฎความสัมพันธ์ โดยกำหนดให้

$I = \{ I_1, I_2, \ldots, I_n \}$ คือเซตของไอเทมที่เรียงตามลำดับตัวอักษรก่อนหลังตามลำดับ (Lexicographic order)

T คือ เซตของไอเทมก็ต่อเมื่อ $T \subseteq I$

TID คือแต่ละทรานแซคชั่น T สัมพันธ์กันด้วยตัวระบุ (identifier) ที่เป็นหนึ่งเดียว (unique)

$D = \{T_1, T_2, \ldots, T_m\}$ คือ เซตของทรานแซคชั่นในฐานข้อมูล

X คือเซตของไอเทมในแต่ละทรานแซคชั่นก็ต่อเมื่อ $X \subseteq T$

กฎความสัมพันธ์แสดงในรูปของกฎ $X \rightarrow Y$ คือ IF X THEN Y ซึ่งค่า $X \subset I$ กับ $Y \subset I$ และ $X \cap Y = \phi$

การหากฎความสัมพันธ์มีเกณฑ์ในการวัด 2 ค่า

1. ค่าสนับสนุนขั้นต่ำ (Minimum Support) เป็นค่าใช้วัดไอเทมเซตที่ผ่านเกณฑ์ไปเป็น Large Itemsets มีค่าเป็นเปอร์เซ็นต์ระหว่าง 0 – 100 เปอร์เซ็นต์

2. ค่าความเชื่อมั่นขั้นต่ำ (Minimum Confidence) ภายหลังจากได้ค่า Large Itemsets นำมาสร้างเป็นกฎความสัมพันธ์ โดยนำค่าความเชื่อมั่นขั้นต่ำสำหรับพิจารณากฎความสัมพันธ์ของข้อมูลว่ากฎที่ได้มีความแข็งแกร่งหรือไม่ มีค่าเป็นเปอร์เซ็นต์ระหว่าง 0 – 100 เปอร์เซ็นต์

Transaction ID (TID)	Item
1	dry roasted peanuts, ice cream, milk
2	bread, jam, milk
3	bread, ice cream, jam, milk
4	bread, jam

รูปที่ 2.1 ข้อมูลการซื้อสินค้าของลูกค้า

จากตัวอย่างในรูปที่ 2.1 แสดงข้อมูลการซื้อสินค้าของลูกค้า กลุ่มของไอเทม I = {bread, dry roasted peanuts, ice cream, jam, milk} ดังนั้น {dry roasted peanuts, ice cream} จึงเป็นกลุ่มไอเทมเซตที่อยู่ในไอเทม I เช่นกันและในทรานแซคชั่นที่ 1 (TID 1) = {dry roasted peanuts, ice cream, milk} $\{\text{dry roasted peanuts, ice cream}\} \subseteq \{\text{dry roasted peanuts, ice cream, milk}\}$

โดยที่กฎ $X \rightarrow Y$ มีค่าสนับสนุน s ในกลุ่มทรานแซคชั่น D ถ้าค่า s% เป็นทรานแซคชั่นใน D ที่มี $X \cup Y$ เป็นเปอร์เซ็นต์ของค่าสนับสนุนที่มีข้อมูลของทรานแซคชั่นอยู่ใน D และ X กับ Y เกิดขึ้นพร้อมกัน (ค่าเปอร์เซ็นต์ของจำนวนรายการของข้อมูลที่สนใจหรือทรานแซคชั่นที่สนใจในฐานข้อมูล เทียบกับจำนวนรายการข้อมูลทั้งหมด)

ตัวอย่างในรูปที่ 2.1 แสดงให้เห็นว่ามีกลุ่มของไอเทม {bread, milk} อยู่ในทรานแซคชั่นที่ 2 และทรานแซคชั่นที่ 3 ดังนั้นค่าสนับสนุนสำหรับกลุ่มไอเทม {bread, milk} คือ 2/4*100 = 50% ค่าสนับสนุน 50 % ที่ได้มาจากกลุ่มไอเทม {bread, milk} จำนวน 2 ทรานแซคชั่นจากทั้งหมดจำนวน 4 ทรานแซคชั่นในฐานข้อมูล โดย กฎ {bread}→{milk} มีความหมายคือ ลูกค้าที่ซื้อนมและขนมปังไปด้วยกัน คิดเป็นร้อยละ 50 เมื่อเทียบกับรายการขายทั้งหมด

โดยกฎ $X \rightarrow Y$ มีค่าความเชื่อมั่น c ในกลุ่มทรานแซคชั่น D ถ้าค่า c% ในทรานแซคชั่น D เป็นเปอร์เซ็นต์ของค่าความเชื่อมั่นที่มีข้อมูลของทรานแซคชั่นอยู่ใน D ที่มี X แล้วจะมี Y ด้วย

ตัวอย่างในรูป 2.1 จากกฎ {bread}→{milk} จะทราบว่ามีทรานแซคชั่นที่ 2 และทรานแซคชั่น 3 ที่มี {bread, milk} บรรจุอยู่ และมีทรานแซคชั่นที่บรรจุ {bread} อยู่จำนวน 3 ทรานแซคชั่น คือทรานแซคชั่นที่ 2, 3 และ 4 ดังนั้นค่า confidence จากกฎ {bread}→{milk} คือ 2/3*100 = 66.7% หมายความว่ามีลูกค้าที่ซื้อขนมปังทั้งหมดร้อยละ 66.7 จะซื้อนมไปด้วย

ทฤษฎีสร้างเป็นสัญลักษณ์ความน่าจะเป็น ซึ่งมีค่าสนับสนุนและค่าความเชื่อถือสำหรับกฎความสัมพันธ์คือ

$$\text{support}(X \rightarrow Y) = P(XUY) = \frac{\text{support\_count}(X \cup Y)}{\text{amount\_transaction}} \quad (2.1)$$

$$\text{confidence}\,(X \rightarrow Y) = P(Y \mid X) = \frac{\text{support\_count}(X \cup Y)}{\text{support\_count}(X)} \quad (2.2)$$

การไมนิ่งเพื่อค้นหากฎความสัมพันธ์ซึ่งมีค่าสนับสนุนและค่าความเชื่อมั่นต้องผ่านเกณฑ์ค่าสนับสนุนขั้นต่ำและค่าความเชื่อมั่นขั้นต่ำ อย่างไรก็ตามการไมนิ่งกฎความสัมพันธ์จะมีขั้นตอนสำคัญ 2 ขั้นตอน

1. การค้นหา Large itemsets : Large itemsets เป็นไอเทมเซตที่ผ่านเกณฑ์ของค่าสนับสนุนขั้นต่ำ เขียนแทนด้วย L_k ในขั้นตอนนี้จะเป็นการค้นหา Large itemsets จนถึง Large k-itemsets $\{L_1, \ldots, L_k\}$

ตัวอย่างรูปที่ 2.1 ถ้ากำหนดให้ค่าสนับสนุนขั้นต่ำคือ 3 ทรานแซคชั่น สำหรับไอเทมเซต {bread} จะบรรจุอยู่ในทรานแซคชั่นที่ 2, 3 และ 4 ค่าสนับสนุนของ {bread} คือ 3 ดังนั้นค่าสนับสนุนของ {bread} ไม่น้อยกว่าค่าสนับสนุนขั้นต่ำคือ 3 เพราะฉะนั้นไอเทมเซต {bread} เป็นค่า Large itemset ส่วนไอเทมเซต {ice cream} มีอยู่ในทรานแซคชั่นที่ 1 และ ทรานแซคชั่นที่ 3 ค่าสนับสนุนของไอเทมเซต {ice cream} คือ 2 ดังนั้นค่าสนับสนุนของไอเทมเซตนี้น้อยกว่าค่าสนับสนุนขั้นต่ำ เพราะฉะนั้นจึงมีผลทำให้ไอเทมเซต {ice cream} เป็น Small itemset

2. การสร้างกฎความสัมพันธ์จาก Large itemsets โดยกฎที่ได้จะถูกต้องก็ต่อเมื่อมีค่าความเชื่อมั่นมากกว่าหรือเท่ากับค่าความเชื่อมั่นขั้นต่ำ

2.1.1 การค้นหากฎความสัมพันธ์ด้วยอัลกอริทึมแอพริออรี (Apriori Algorithm)

งานวิจัย Fast algorithm for mining association rules [2] ได้นำเสนออัลกอริทึมแอพริออรี ซึ่งเป็นอัลกอริทึมหนึ่งที่ใช้เพื่อหารูปแบบกฎความสัมพันธ์ และนำไปใช้ในการวิเคราะห์หรือทำนายปรากฏการณ์ เป็นอัลกอริทึมที่ได้รับความนิยมมีผู้ให้ความสนใจศึกษาแพร่หลายอย่างมากและนำไปประยุกต์ใช้ในงานวิจัยต่างๆ

นิยามและความหมายของคำในอัลกอริทึมแอพริออรี

1. k-itemsets คือ เซตของข้อมูลที่มีจำนวนสมาชิกไอเทม k ตัว
2. L_k คือ เซตของ Large k-itemset ซึ่งทุกเซตมีความถี่หรือค่าสนับสนุนมากกว่าหรือเท่ากับค่าสนับสนุนขั้นต่ำ

3. C_k คือ เซตของ Candidate k-itemset ที่สร้างมาจาก Large k-1 itemsets เป็นตัวเลือกที่อาจเป็น Large k itemsets

อัลกอริทึมเอพริออรีมีหลักการทำงานที่สำคัญ 2 ขั้นตอน

1. ขั้นตอนการเชื่อมความสัมพันธ์ (Join step) นำ L_{k-1} มาเชื่อมความสัมพันธ์กับ L_{k-1} ($L_{k-1} \bowtie L_{k-1}$) เพื่อสร้าง C_k โดยอัลกอริทึมเอพริออรีจำเป็นต้องให้ไอเทมเซตที่อยู่ในแต่ละทรานแซคชั่นต้องเรียงตามลำดับอักษร ซึ่งในกรณีที่เป็นการสร้าง C_1 (Candidate 1-itemsets) จะนำแต่ละไอเทมแต่ละตัวที่มีค่าสนับสนุนมากกว่าศูนย์ในทรานแซคชั่นของฐานข้อมูลมาสร้างได้เป็น C_1 โดยไม่ต้องทำการเชื่อมความสัมพันธ์

ตัวอย่าง ไอเทมเซต L_2 คือ {ab} และ {ac} ภายหลังเชื่อมความสัมพันธ์ จะได้ C_3 = {abc}

2. ขั้นตอนการตัดออก (Prune step) ในขั้นตอนนี้ทำการตัดไอเทมเซตออกจาก Candidate Itemsets (C_k) เมื่อใดก็ตามที่ (k-1)-subset ของ C_k ไม่ได้เป็นสมาชิกของ L_{k-1}

ตัวอย่าง C_3 คือ {abc} ดังนั้นซับเซตของ {abc} จะต้องมีไอเทมเซตอยู่ใน L_2 คือ {ab}, {ac} และ {bc} ครบทุกตัว

Algorithm: Apriori. Find frequent itemsets using and iterative level-wise approach based on candidate generation.

Input: Database, D, of transaction; minimum support threshold, min_sup.

Method:

L_1= find_frequent_1-itemsets(D);

for($k = 2; L_{k-1} \neq \phi; k++$){

C_k = apriori_gen(L_{k-1},min_sup);

for each transaction $t \in D$ { //สแกน D สำหรับนับแต่ละทรานแซคชั่น

C_t = subset(C_k,t);

for each candidate $c \in C_t$

c.count++;

$L_k = \{c \in C_k \mid c.count \geq min\_sup\}$ }

}

return $L = \bigcup_k L_k$;

รูปที่ 2.2 แสดง Apriori Algorithm

```
procedure apriori_gen (L_{k-1} : frequent (k-1)-itemset; min_sup : minimum support threshold)
    for each itemset l_1 ∈ L_{k-1}
    for each itemset l_2 ∈ L_{k-1}
    if (l_1[1] = l_2[1]) ∧ (l_1[2] = l_2[2]) ∧ ... ∧ (l_1[k-2] = l_2[k-2] ∧
    (l_1[k-1] < l_2[k-1]) then
    {
        c = l_1 ▷◁ l_2 ;// ขั้นตอนการ join : เป็นการสร้าง Candidate Itemsets
        if has _infrequent_subset(c, L_{k-1}) then
            delete c; //ขั้นตอนการ prune ออกจาก Candidate Itemsets
        else
            add c to C_k
        }
        return C_k;
```

รูปที่ 2.3 แสดง procedure apriori_gen

```
procedure has_infrequnt_subset (c:candidate k-itemset; L_{k-1} : frequent (k-1) – itemset;
    for each (k-1)-subset s of c
        if  s ∉ L_{k-1} then
        return True;
        return False;
```

รูปที่ 2.4 แสดง procedure has_infrequent_subset

โดยอัลกอริทึมเอพริออรีเป็นการทำการค้นหาค่า Large itemsets ในฐานข้อมูล ซึ่งจะทำการค้นหาโดยการวนรอบค้นหาแต่ละทรานแซคชั่นในฐานข้อมูล และจะทำการวนรอบค้นหาหลายครั้ง (Iteration) โดยนำเอาค่า Large k-1 itemsets ที่ได้จากการทำงานก่อนหน้านี้มาสร้าง C_k (Candidate k-item) แล้วไปผ่านกระบวนการคัดเลือกเพื่อให้ได้ค่า Large k-itemsets

อธิบายการทำงานของอัลกอริทึมเอพริออรี

1. ทำการหาค่า L_1 โดยได้จากการสร้าง C_1 (ได้จากแต่ละไอเทมเซตที่มีในทรานแซคชั่นที่อยู่ในฐานข้อมูล) นำแต่ละไอเทมเซตใน C สแกนหาในฐานข้อมูลว่ามีค่าสนับสนุนเท่าไร แล้วนำค่า

สนับสนุนแต่ละไอเทมเซตมาเปรียบเทียบกับค่าสนับสนุนขั้นต่ำว่ามากกว่าหรือเท่ากับค่าสนับสนุนขั้นต่ำหรือไม่ ไอเทมเซตที่ผ่านเกณฑ์จะปรากฏอยู่ใน L_1

2. ทำการหา C_2 โดยได้จากการเชื่อมความสัมพันธ์ L_1 กับ L_1 เข้าด้วยกันแล้วนำไปพิจารณาว่าแต่ละไอเทมเซตใน C_2 มีซับเซตอยู่ใน L_1 ทุกตัวหรือไม่ ถ้ามีไม่ครบทุกตัวก็จะทำการตัดออกจาก C_2 แล้วนำไอเทมเซตแต่ละตัวใน C_2 ไปสแกนหาค่าสนับสนุน แล้วนำค่าสนับสนุนของแต่ละไอเทมเซตมาเปรียบเทียบว่ามีค่ามากกว่าหรือเท่ากับค่าสนับสนุนขั้นต่ำหรือไม่ ไอเทมเซตที่ผ่านเกณฑ์จะปรากฏอยู่ใน L_2 และการค้นหาขั้นตอนถัดไปการทำงานเช่นเดียวกับขั้นตอนนี้

การทำงานของอัลกอริทึมเอพริออรีเมื่อกฎความสัมพันธ์ที่มีค่าสนับสนุนและมีค่าความเชื่อมั่นที่ได้ผ่านค่าสนับสนุนขั้นต่ำ และค่าความเชื่อมั่นขั้นต่ำตามกำหนด แต่ละกฎความสัมพันธ์ที่ผ่านเกณฑ์พิจารณาดังกล่าว สามารถได้เป็นกฎความสัมพันธ์ที่แข็งแกร่ง

อัลกอริทึมเอพริออรีนั้นเป็นอัลกอริทึมของการไมนิ่งในรูปแบบกฎความสัมพันธ์แบบหนึ่งมิติ ค้นหากฎความสัมพันธ์ที่อยู่ภายในแอททริบิวต์หลักเท่านั้น (เช่น แสดงความสัมพันธ์เฉพาะข้อมูลสินค้าในการซื้อของลูกค้า)

ข้อด้อยของอัลกอริทึมเอพริออรี

1. เมื่อฐานข้อมูลมีการเปลี่ยนแปลงต้องทำการค้นหากฎความสัมพันธ์ใหม่ทั้งหมด และในการค้นหากฎความสัมพันธ์ใหม่แต่ละครั้ง จะต้องทำการสแกนข้อมูลทั้งฐานข้อมูล เพราะไม่ได้นำ Large Itemsets ที่เป็นความรู้เดิมซึ่งเคยได้ค้นหาไว้มาใช้ให้เกิดประโยชน์

2. จากปัญหาข้างต้นทำให้ต้องสูญเสียเวลาในการค้นหากฎความสัมพันธ์ใหม่ทั้งหมดอาจจะใช้เวลามาก

3. เนื่องจากเป็นอัลกอริทึมสำหรับการค้นหากฎความสัมพันธ์ที่อยู่ในแอททริบิวต์หลักเดียวกันเท่านั้น เพราะฐานข้อมูลจริงอาจมีการเก็บข้อมูลที่หลายมิติ (แอททริบิวต์) เช่น อายุ ที่อยู่ และสินค้าของลูกค้า

2.1.2 อัลกอริทึมเอพริออรีสำหรับการหากฎความสัมพันธ์แบบมิติผสม

งานวิจัย Mining conditional Hybrid-dimension association rules on the basis of Multi-dimensional transaction database [8] ได้นำเสนอวิธีปรับการทำงานของอัลกอริทึมเอพริออรีที่มีความสามารถในการค้นหากฎความสัมพันธ์ของข้อมูลในฐานข้อมูลที่เก็บทรานแซคชั่นข้อมูลแบบมิติเดียว ทำให้อัลกอริทึมเอพริออรีสามารถค้นหากฎความสัมพันธ์ของข้อมูลเพียงมิติเดียว ซึ่งโดยทั่วไปการจัดเก็บข้อมูลในฐานข้อมูลปัจจุบันนี้มีการเก็บรายละเอียดของข้อมูลมากมายหลายมิติ ดังนั้นจำเป็นต้องปรับเปลี่ยนบางขั้นตอนของอัลกอริทึมเอพริออรี ให้สามารถนำมาใช้กับฐานข้อมูลที่มีการจัดเก็บทรานแซคชั่นข้อมูลแบบหลายมิติและเป็นการค้นกฎความสัมพันธ์แบบมิติผสมได้

ประเภทของมิติในฐานข้อมูลสามารถแบ่งได้เป็น 2 ประเภท

1. แอททริบิวต์รอง (Subordinate Attribute) จะแสดงรายละเอียดข้อมูลเกี่ยวกับแอททริบิวต์นั้นเพียงหนึ่งค่าเท่านั้นในแต่ละทรานแซคชั่น เช่นข้อมูลสินค้าที่มีในแต่ละทรานแซคชั่น

2. แอททริบิวต์หลัก (Main Attribute) จะแสดงรายละเอียดข้อมูลเกี่ยวกับแอททริบิวต์นั้นมากกว่าหนึ่งค่าในแต่ละทรานแซคชั่น เช่นข้อมูลพื้นที่ของผู้ซื้อที่มีในแต่ละทรานแซคชั่น

การค้นหากฎความสัมพันธ์สามารถแบ่งได้เป็น 2 ประเภท

1. **กฎความสัมพันธ์แบบมิติภายใน (Intra Dimension Association Rules)** เป็นการค้นหากฎความสัมพันธ์ที่แสดงความสัมพันธ์มิติเดียว

$$\text{buys (X, "notebook computer")} \rightarrow \text{buys (X, "antivirus software")} \quad (2.3)$$

จากกฎความสัมพันธ์ที่ 2.3 ความหมายของกฎความสัมพันธ์นี้คือ ถ้าลูกค้าซื้อเครื่องคอมพิวเตอร์โน้ตบุ๊คแล้ว จะซื้อซอฟต์แวร์แอนตี้ไวรัสไปด้วยกัน

กฎความสัมพันธ์ที่ได้มีเพียงมิติการซื้อมิติเดียวเท่านั้น

2. กฎความสัมพันธ์แบบหลายมิติ (Multidimension Association Rules) เป็นการค้นหากฎความสัมพันธ์ที่สามารถแสดงความสัมพันธ์ในแต่ละแอททริบิวต์ที่มีในทรานแซคชั่นได้หลายมิติ โดยสามารถแบ่งออกได้เป็น 2 ประเภทคือ

- กฎความสัมพันธ์แบบระหว่างมิติ (Inter Dimension Association Rules)

เป็นการค้นหากฎความสัมพันธ์แบบหลายมิติ กฎความสัมพันธ์ที่ได้จะไม่มีการทำนายแต่ละแอททริบิวต์ในทรานแซคชั่นข้อมูลซ้ำ

$$\text{age(X,"20…29")} \wedge \text{occupation (X,"student")} \rightarrow \text{buys (X,"notebook computer")} \quad (2.4)$$

จากกฎความสัมพันธ์ที่ 2.4 มีความหมายคือ ถ้าลูกค้าที่มีอายุระหว่าง 20-29 ปี และมีอาชีพเป็นนักเรียนจะซื้อเครื่องคอมพิวเตอร์โน้ตบุ๊ค

กฎความสัมพันธ์ที่ได้มามีทั้ง 3 มิติคือ มิติอายุ มิติอาชีพ และมิติการซื้อ ไม่มีมิติใดซ้ำกัน

- กฎความสัมพันธ์แบบมิติผสม (Hybrid Dimension Association Rules)

เป็นการค้นหากฎความสัมพันธ์แบบหลายมิติ กฎความสัมพันธ์ที่ได้สามารถมีการทำนายแอททริบิวต์หลักที่มีในทรานแซคชั่นข้อมูลซ้ำได้

$$\text{age(X,"20…29")} \wedge \text{buys (X,"notebook computer")} \rightarrow \text{buys (X,"antivirus software")} \quad (2.5)$$

จากกฎความสัมพันธ์ที่ 2.5 มีความหมายคือ ถ้าลูกค้าที่มีอายุระหว่าง 20-29 ปีที่ซื้อเครื่องคอมพิวเตอร์โน้ตบุ๊ค จะซื้อซอฟต์แวร์แอนตี้ไวรัสไปด้วยกัน

กฎความสัมพันธ์ที่ได้มามีสองมิติคือ มิติอายุ และมิติการซื้อ แสดงให้เห็นว่ามีการทำนายมิติการซื้อซึ่งเป็นแอททริบิวต์หลักซ้ำ

การสร้างความสัมพันธ์ในกฎความสัมพันธ์แบบมิติผสมสามารถแบ่งได้เป็น 2 รูปแบบ

1. การเชื่อมความสัมพันธ์แบบมิติภายใน (Intra-dimensional join)

กำหนดให้ l_1 และ l_2 เป็นไอเทมเซตที่เป็นสมาชิกอยู่ใน L_{k-1} โดยใน l_1 และ l_2 ตั้งแต่ไอเทมที่ 1 จนถึงไอเทมที่ k-1 ดังนี้

$$(l_1[1] = l_2[1]) \cap (l_1[2] = l_2[2]) \cap \ldots \cap (l_1[k-2] = l_2[k-2]) \cap (l_1[k-1] < l_2[k-1])$$

เมื่อไอเทมของ l_1 และ l_2 อยู่ในรูปแบบลำดับดังกล่าวข้างต้น

ผลลัพธ์คือ $l_1[1]\, l_1[2] \ldots l_1[k-1]\, l_2[k-1]$

ตัวอย่าง กำหนดให้ $L_3 = \{A, B, C\}, \{A, B, D\}$ โดย $l_1 = \{A, B, C\}$ และ $l_2 = \{A, B, D\}$

ผลลัพธ์หลังการเชื่อมความสัมพันธ์ l_1 และ l_2 คือ $\{A, B, C, D\}$

2. การเชื่อมความสัมพันธ์แบบระหว่างมิติ (Inter-dimensional join)

กำหนดให้ l_1 และ l_2 เป็นไอเทมเซตที่เป็นสมาชิกอยู่ใน L_{k-1} โดยใน l_1 ตั้งแต่ไอเทมที่ 2 จนถึงไอเทมที่ k-1 และใน l_2ตั้งแต่ไอเทมที่ 1 จนถึงไอเทมที่ k-2 โดย

$$(l_1[2] = l_2[1]) \cap (l_1[3] = l_2[2]) \cap \ldots \cap (l_1[k-1] = l_2[k-2]) \cap (l_1[1] < l_2[k-1])$$

เมื่อไอเทมของ l_1 และ l_2 อยู่ในรูปแบบลำดับดังกล่าวข้างต้น

ผลลัพธ์คือ $l_1[1]\, l_2[1] \ldots l_2[k-2]\, l_2[k-1]$

ตัวอย่างการ กำหนดให้ $L_3 = \{A, B, C\}, \{B, C, D\}$ โดย $l_1 = \{A, B, C\}$ และ $l_2 = \{B, C, D\}$

ผลลัพธ์หลังการเชื่อมความสัมพันธ์ l_1 และ l_2 คือ $\{A, B, C, D\}$

ดังนั้นเพื่อให้สามารถค้นหากฎความสัมพันธ์ได้อย่างครบถ้วนทุกแอททริบิวต์ของทรานแซคชั่นข้อมูลที่มีหลายมิติในฐานข้อมูล การค้นหากฎความสัมพันธ์แบบระหว่างมิติไม่สามารถทราบความสัมพันธ์ที่มีระหว่างภายในแอททริบิวต์หลักที่สนใจได้ จึงต้องใช้การค้นหากฎความสัมพันธ์แบบมิติผสม ซึ่งทำให้สามารถทราบความสัมพันธ์ที่มีระหว่างภายในแอททริบิวต์หลักที่สนใจ และทราบถึงความสัมพันธ์ของข้อมูลที่มีได้หลากหลายกว่าการค้นหากฎความสัมพันธ์แบบระหว่างมิติ ทำให้เห็นถึงจุดเด่นของการค้นหากฎความสัมพันธ์แบบมิติผสม จึงได้นำการค้นหากฎความสัมพันธ์แบบมิติผสมมาใช้

งานวิจัยนี้นำเสนอการค้นหากฎความสัมพันธ์แบบมิติผสมมาประยุกต์ใช้ในอัลกอริทึมเอพริออรี ผู้วิจัยได้ปรับปรุงอัลกอริทึมเอพริออรีในขั้นตอนการเชื่อมความสัมพันธ์ ให้สามารถใช้ได้กับทรานแซคชั่นข้อมูลแบบหลายมิติ และปรับปรุงอัลกอริทึมบางส่วนเพื่อให้สามารถค้นหาข้อมูลด้วยวิธีการค้นหากฎความสัมพันธ์แบบมิติผสม ซึ่งการค้นหากฎความสัมพันธ์แบบมิติผสม

ผลลัพธ์ของกฎความสัมพันธ์ที่ได้หลากหลายมากกว่าการค้นหากฎความสัมพันธ์แบบหนึ่งมิติและการค้นหากฎความสัมพันธ์แบบระหว่างมิติ

```
L1 = find_frequent_1_itemsets(D);
//compress the transaction database, according to the generated frequent 1-itemsets
D' = trans_compression(D);
//generate candidate 2-itemsets
C2 = apriori_gen1(L1);
//generate frequent 2-itemsets
L2 = find_frequent_2_itemsets(D');
//generate candidate k-itemsets Ck from frequent (k-1)
for (k=3; Lk-1 ≠ φ ; k++) do
Begin
// generate all the candidate k-itemsets Ck by joining
    Ck = apriori-gen(Lk-1);
//use the Apriori property to eliminate candidates having a subset that is not frequent
    for each transaction t∈ D' do
    begin (the t equal to each Record)
        Ct = subset(Ck,t);
        for each candidate c ∈ Ct do
            c.count++;
        end
        //all those candidate k-itemsets
        Lk = { c ∈ Ct | c.count >= minsup}
        // Ck satisfying minimum support form the set of frequent k-itemsets Lk
End
Answer = ⋃k Lk ;
//generate rules from all frequent itemsets
For each large itemsets Lk ∈ Answer, (k ≥ 2) do
genrules(Lk ,Lk)
```

รูปที่ 2.5 อัลกอริทึมเอพริออรีสำหรับการหากฎความสัมพันธ์แบบมิติผสม

```
procedure apriori_gen1 (L'k-1: Large itemsets)
{
    C[k] = null;
    for each l1 ∈ L'k-1
     for each l2 ∈ L'k-1
      if isInnerJoin (l1) or isInnerJoin (l2)
     // if l1 or l2 can make intradimension join
     // isInnerJoin (l1) is a bool funtion, l1 is parameter, its' function is to judge
        whether
     //an item l1 can make intradimension join, if the return value is 'true', it's allowed.
         then {
             c = l1 ⊳⊲ l2;
              InsertInto C[k]
              }
             for each c ∈ C[k]
    for each (k-1)-subset s of c
      if s ∉ L'k-1
       then delete c from C[k]
}
```

รูปที่ 2.6 แสดง procedure apriori_gen1

```
procedure apriori_gen (L'k-1: Large itemsets)
{
    C[k] = null;
    for each l1 ∈ L'k-1
     for each l2 ∈ L'k-1
      if not (isInnerJoin (l1)) and not (isInnerJoin (l2))
      then //make intradimension join {
        if (l1[1] = l2[1]) ∧ (l1[2] = l2[2]) ∧ ... ∧ (l1[k-2] = l2[k-2]) ∧ (l1[k-1] < l2[k-1])
         then {
            c = l1 ⊳⊲ l2;
            InsertInto C[k];
         }
        }
          else //make interdimension join {
            if (l1[2]= l2[1]) ∧ (l1[3]= l2[2]) ∧ ... ∧ (l1[k-1] = l2[k-2]) ∧ l1[1] < l2[k-1])
             then {
                c = l1 ⊳⊲ l2;
                InsertInto C[k];
             }
    for each c ∈ C[k]
      for each (k-1)-subset s of c
       if s ∉ L'k-1
        then delete c from C[k]
}
```

รูปที่ 2.7 แสดง procedure apriori_gen

อัลกอริทึมเอพริออรีสำหรับการค้นหากฎความสัมพันธ์แบบมิติผสมนี้ได้ปรับปรุงขั้นตอนการเชื่อมความสัมพันธ์ โดยเอาวิธีการสร้างกฎความสัมพันธ์ทั้งสองแบบคือการเชื่อมความสัมพันธ์แบบมิติภายใน และการเชื่อมความสัมพันธ์แบบระหว่างมิติ นำมาใช้ร่วมกันเพื่อให้อัลกอริทึมเอพริออรีสามารถค้นหากฎความสัมพันธ์แบบมิติผสม

ขั้นตอนการทำงานอัลกอริทึมเอพริออรีสำหรับการหากฎความสัมพันธ์แบบมิติผสม

1. เป็นขั้นตอนในการหาไอเทมเซตที่เป็น Large Itemsets จากฐานข้อมูลไปไว้ใน L_1 เมื่อได้ L_1 จะนำไปใช้สร้างค่า C_2

2. เป็นขั้นตอนหาค่า C_2 โดยมีวิธีสร้างจากการเชื่อมความสัมพันธ์ L_1 กับ L_1 ด้วย procedure apriori_gen1 โดยที่ l_1 กับ l_2 เป็นสมาชิกที่อยู่ใน L_1 แต่มีข้อจำกัดในการเชื่อมความสัมพันธ์ คือไม่สามารถเชื่อมความสัมพันธ์ภายในแอททริบิวต์รองที่เป็นแอททริบิวต์เดียวกันได้ เช่น แอททริบิวต์ของอายุซึ่งเป็นแอททริบิวต์รองไม่สามารถเชื่อมความสัมพันธ์กับแอททริบิวต์อายุที่เป็นแอททริบิวต์เดียวกันได้ เมื่อได้ผลลัพธ์ภายหลังทำการเชื่อมความสัมพันธ์คือ C_2 แล้วจะนำมาทำการตรวจสอบว่าแต่ละไอเทมเซตที่ได้มีซับเซตอยู่ใน L_1 อยู่ทุกซับเซตหรือไม่ ถ้าพบว่าไอเทมเซตที่พิจารณาไม่ผ่านเกณฑ์จะทำการตัดไอเทมเซตดังกล่าวออกจาก C_2

3. เป็นขั้นตอนในการสร้าง L_2 โดยนำ C_2 มาพิจารณา โดยไอเทมเซตใดใน C_2 มีค่าสนับสนุนมากกว่าหรือเท่ากับค่าสนับสนุนขั้นต่ำ เมื่อผ่านเกณฑ์ค่าสนับสนุนขั้นต่ำดังกล่าวก็จะไปปรากฏอยู่ใน L_2

4. เป็นการสร้าง C_k ตั้งแต่ k ที่มีค่าตั้งแต่ 3 เป็นต้นไป โดยเป็นการเชื่อมความสัมพันธ์ระหว่าง L_{k-1} กับ L_{k-1} ด้วยการนำ procedure apriori_gen มาใช้ ซึ่ง procedure apriori_gen นี้มีส่วนสำคัญในการทำงานที่มีการพิจารณารูปแบบของไอเทมเซตจาก 2 กรณี โดยมีเงื่อนไขการทำงานดังนี้คือ ถ้าเป็นการเชื่อมความสัมพันธ์ด้วยภายในแอททริบิวต์หลักเดียวกันเองการเชื่อมความสัมพันธ์ จะเป็นแบบมิติภายใน หากเป็นรูปแบบอื่นๆให้ทำการเชื่อมความสัมพันธ์ แบบระหว่างมิติ โดยในแต่ละรูปแบบจะมีการเช็คว่าไอเทมเซตดังกล่าวมีซับเซตอยู่ใน L_{k-1} ครบทุกซับเซตหรือไม่ ถ้าไม่ครบทุกตัวจะทำการตัดไอเทมเซตนั้นออกจาก C_k ได้ทันที เพราะว่าไอเทมเซตดังกล่าวไม่สามารถเป็น Large Itemsets ได้

5. ค่าสนับสนุนไอเทมเซตแต่ละตัวภายใน C_k ทุกตัวที่มีค่ามากกว่าหรือเท่ากับสนับสนุนขั้นต่ำจะไปปรากฏอยู่ใน L_k

6. ทำซ้ำขั้นตอนที่ 4 จนกระทั่งไม่สามารถสร้าง C_k ได้อีก

7. เป็นการนำ Large Itemsets ทั้งหมดที่หามาได้ นำมาสร้างกฎความสัมพันธ์โดยเริ่มจากตั้งแต่ 2 Large Itemset เป็นต้นไป

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

ข้อด้อยของกฎความสัมพันธ์แบบมิติผสมโดยใช้อัลกอริทึมเอพริออรี

1. อัลกอริทึมนี้ไม่ได้นำ Large Itemset ซึ่งเป็นความรู้เดิมที่เคยได้ค้นหาไว้มาใช้ให้เกิดประโยชน์

2. เมื่อฐานข้อมูลมีการเปลี่ยนแปลงต้องทำการค้นหากฎความสัมพันธ์ใหม่ทั้งหมดและในการค้นหากฎความสัมพันธ์ใหม่แต่ละครั้งจะต้องทำการสแกนข้อมูลในฐานข้อมูลใหม่ทั้งหมด

2.2 การค้นหากฎความสัมพันธ์ของการเพิ่มข้อมูล (Incremental Association Rule Discovery)

การค้นหากฎความสัมพันธ์ของการเพิ่มข้อมูล เป็นการค้นหากฎความสัมพันธ์เนื่องจากมีการเพิ่มขึ้นของข้อมูลใหม่เข้าสู่ฐานข้อมูลทำให้มีผลต่อการปรับปรุงค่าสนับสนุนไอเทมเซตของข้อมูลอาจมีผลทำให้กฎความสัมพันธ์เดิมที่เคยเป็นกฎความสัมพันธ์ที่แข็งแกร่งเปลี่ยนเป็นกฎความสัมพันธ์ที่อ่อนแอหรืออาจทำให้กฎความสัมพันธ์ที่อ่อนแอเปลี่ยนเป็นกฎความสัมพันธ์ที่แข็งแกร่ง ทำให้ต้องมีการค้นหากฎความสัมพันธ์ใหม่เพื่อให้กฎความสัมพันธ์ถูกต้องอยู่เสมอ และในการค้นหาค่าสนับสนุนของไอเทมเซตในฐานข้อมูลเก่า จะใช้เวลาในการค้นหามากเพราะโดยทั่วไปแล้วฐานข้อมูลเก่ามีขนาดใหญ่ และฐานข้อมูลที่เพิ่มใหม่จะมีขนาดเล็กกว่า

การเพิ่มข้อมูลเข้าสู่ฐานข้อมูลสามารถเป็นไปได้ทั้งหมด 4 กรณีคือ

กรณี	ฐานข้อมูลเดิม	ฐานข้อมูลเพิ่มใหม่	ฐานข้อมูลปรับปรุง
1	X.support >= minsupp*(DB)	X.support >= minsup*(db)	X.support >= minsupp*(DB+db)
2	X.support >= minsupp*(DB)	X.support < minsup*(db)	X.support >= หรือ < minsupp*(DB+db)
3	X.support < minsupp*(DB)	X.support >= minsup*(db)	X.support >= หรือ < minsupp*(DB+db)
4	X.support < minsupp*(DB)	X.support < minsup*(db)	X.support < minsupp*(DB+db)

รูปที่ 2.8 แสดงความเป็นไปได้ของไอเทมเซตภายหลังการปรับปรุงข้อมูล

นิยามและความหมายของคำในการค้นหากฎความสัมพันธ์ของการเพิ่มข้อมูล

1. DB (Original Database) คือ ฐานข้อมูลเดิม

110402

2. db (Incremental Database) คือ ฐานข้อมูลเพิ่มใหม่
3. DB' (Update Database, DB+db) คือ ฐานข้อมูลปรับปรุง เมื่อนำ db รวมเข้ากับ DB
4. L คือ ค่า Large Itemset ของฐานข้อมูลเดิม เป็นไอเทมเซตที่มีค่าสนับสนุนของไอเทมเซตนั้นผ่านค่าสนับสนุนขั้นต่ำของฐานข้อมูลเดิม
5. L' คือ ค่า Large Itemset ของฐานข้อมูลปรับปรุง เป็นไอเทมเซตที่มีค่าสนับสนุนของไอเทมเซตนั้นผ่านค่าสนับสนุนขั้นต่ำของฐานข้อมูลปรับปรุง
6. C คือ Candidate Itemset เป็นไอเทมเซตที่คาดว่าจะสามารถไปเป็น L'

2.2.1 การค้นหากฎความสัมพันธ์ด้วยอัลกอริทึม Fast Update (FUP)

งานวิจัย Maintenance of Discovered Association Rules in Large Databased: An incremental updating technique [3] ได้นำเสนออัลกอริทึม FUP เป็นอัลกอริทึมสำหรับการค้นหากฎความสัมพันธ์เมื่อมีการเพิ่มข้อมูลใหม่เข้าสู่ฐานข้อมูล และเป็นอัลกอริทึมแรกที่นำเสนอการเพิ่มข้อมูลเข้าสู่ฐานข้อมูล โดยอัลกอริทึมนี้มีจุดมุ่งหมายคือการลดการค้นหาในฐานข้อมูลเดิมที่มีขนาดใหญ่ ซึ่งจะให้ความสนใจในข้อมูลใหม่ที่เพิ่มเข้าสู่ฐานข้อมูลโดยนำความรู้ที่เคยได้จากการทำไมนิ่งค้นหาค่า Large Itemsets ก่อนหน้าการเพิ่มข้อมูลเข้าสู่ฐานข้อมูลมาใช้ประโยชน์ เพื่อลดจำนวนการค้นหาไอเทมเซตในทุกทรานแซคชั่นที่มีในฐานข้อมูลทั้งหมดเพื่อนับค่าสนับสนุนของแต่ละไอเทมเซตจะแตกต่างจากอัลกอริทึมเอพริออรีที่ต้องทำการค้นหาจำนวนของแต่ละไอเทมเซต โดยการเข้าไปค้นหาในฐานข้อมูลทั้งหมด แม้ว่าจะมีการเพิ่มข้อมูลเข้าสู่ฐานข้อมูลเข้าไปน้อยมากก็ตามโดยไม่สนใจว่าจะมีความรู้เดิมที่เคยได้จากการไมนิ่งค่า Large Itemsets จากฐานข้อมูลเดิม สามารถนำมาใช้ให้เกิดประโยชน์ต่อได้อีก เพราะในบางกรณีอัลกอริทึม FUP อาจไม่ต้องทำการสแกนหาในฐานข้อมูลเดิมซ้ำ

ในส่วนของสมาชิก Candidate Itemsets การทำงานของอัลกอริทึม FUP จะมีการเก็บเฉพาะไอเทมเซตของฐานข้อมูลเพิ่มใหม่ โดยจัดเก็บไอเทมเซตที่ไม่เป็นสมาชิกของ Large Itemsets ของฐานข้อมูลเดิมที่เคยได้ทำการไมนิ่งค้นหากฎความสัมพันธ์มาแล้ว ซึ่งในอัลกอริทึมเอพริออรีจะเป็นการสร้าง Candidate Itemsets จากฐานข้อมูลเดิมและข้อมูลเพิ่มใหม่ทำให้ได้ Candidate Itemsets เป็นจำนวนมาก เพราะจะรวมเอาทั้งไอเทมเซตที่มีอยู่แล้วและไอเทมเซตที่มีใหม่มารวมกันทำให้มีไอเทมเซตที่ต้องทำการสแกนหาหลายครั้ง ซึ่งในส่วนนี้อัลกอริทึม FUP สามารถลดการสแกนค้นหาในฐานข้อมูลได้ เพราะโดยส่วนใหญ่ฐานข้อมูลเพิ่มใหม่นั้นมีจำนวนข้อมูลที่น้อยกว่าฐานข้อมูลเดิม

อัลกอริทึม FUP มีการพิจารณาในส่วนของไอเทมเซตที่ไม่สามารถเป็น Large Itemsets โดยนำความรู้จาก Large Itemsets ของฐานข้อมูลเดิมนำมาพิจารณาร่วมกับ Large Itemsets ใหม่เพื่อเป็นการลดการค้นหาในฐานข้อมูลเพิ่มใหม่ในขั้นตอนถัดไป

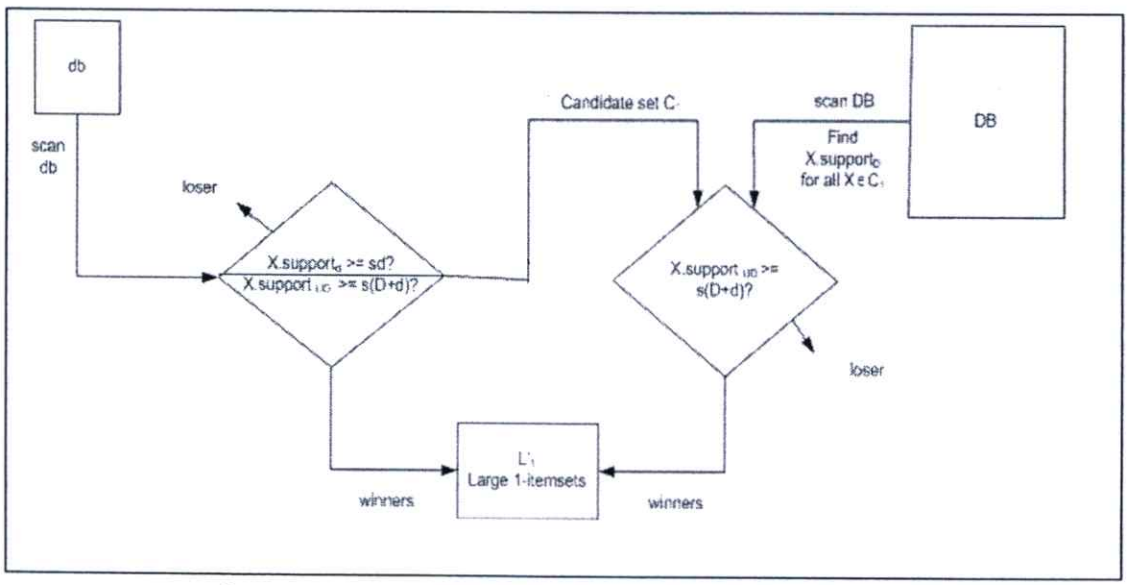
```
Input: DB: the original database (with its size, i.e., the total number of transaction,
          equal to D);
       L_k: the set of all large k- itemsets in DB, where k= 1, …, r;
       db: an increment database (with its size equal to d);
Output: L': The set of all large itemsets in DB ∪ db.
Method:The 1st iteration:/* find L'_1, the set of all large 1-itemsets in DB ∪ db */
       W = L_1; C = φ ; L_1'= φ ; P = φ ;/* W: winners, C: candidate sets,
       L'_1: initialized, P: for optimization */
       for all T ∈ db do /* scan db */
         for all 1-itemset X ⊆ T do {
           if X ∈ W then X.support_d++;
           else {
            if X ∉ C
             then { C = C ∪ {X}; X.support_d = 0;}
            X.supprt_d++;}/*init the support cont and add X into C */
           }
       for all X ∈ W do /* put winners into L'_1 */
         if X.support_UD ≥ s ×(D + d)
          then L_1' = L'_1 ∪{X};
       for all X ∈ C do /*prune candidate sets in C*/
         if X.support_d < s ×d
          then { C = C – {X}; P = P ∪{X};}/* P will be used for optimization */
       for all T ∈ DB do /* scan DB */
         for all 1-itemset X ⊆ T do {
           if X ∈ C then X.support_D++;
           if X ∈ P then removes X from T; /* Transaction T is reduced */
         };
       for all X ∈ C do /* put winners into L_1' */
         if X.support_UD ≥ s × (D + d) then L_1' = L_1' ∪{X};
       return L_1'. /* end of the 1st iteration */
```

รูปที่ 2.9 อัลกอริทึม FUP สำหรับหา Large 1-itemset

```
The k-th iteration:
/*for k = 2 or larger, repeat this program fragment to find L'k.
 the set of all large k-itemsets in the  updated database, until either L'k returned is
empty or db = ϕ */
W = Lk; Lk' = ϕ ; /*W: winners; Lk': initialized */
C = apriori_gen1(L'k-1) - Lk;
/* the size-k candidate sets */
for all k-itemset X∈ W do /* prune off losers in W */
  for all (k-1)-itemset Y ∈ Lk-1 – L'k-1 do
    if Y ⊆ X then { W = W – {X}; break;}
for all T ∈ db do {  /* scan db */
  for all X ∈ Subset(W,T) do X.supportd++;
 /* Subset(W,T) returns all the sets in W contained in T*/
for all X ∈ Subset(C,T) do X.supportd++; /* find support of all X ∈ C */
Reduce db (T);
/* Some items in transactions in db can be removed, discussed in next section */
}
for all X ∈ W do  /* put the winners from W into L'k */
  if  X.supportUD ≥ s × ( D+d )
   then L'k = L'k ∪{X};
for all X ∈ C do /* prune candidate sets in C*/
  if X.supportd < s × d then C = C – {X};
for all T ∈ DB do { /* scan DB */
  for all X ∈ Subset(C,T) do X.supportD++
Reduce_DB(T); }
/* Some items in transactions in DB can be removed, discussed in next section */
for all X ∈ C do
  if X.supportUD ≥ s× (D + d)
   then L'k = L'k ∪{X};
return Lk'. /* the end of the k-th iteration */
```

รูปที่ 2.10 อัลกอริทึม FUP สำหรับหาตั้งแต่ Large 2-itemsets

การทำงานของอัลกอริทึม FUP เพื่อการค้นหากฎความสัมพันธ์ยังอยู่บนพื้นฐานของอัลกอริทึมเอพริออรี โดยจะมีการวนรอบการทำงานซ้ำเพื่อหาความสัมพันธ์ของข้อมูลจาก 1-itemset ไปจนถึง k-itemset อัลกอริทึม FUP นี้จะใช้ค่าสนับสนุนขั้นต่ำเดียวกันทั้งหมด เหมือนกับอัลกอริทึมเอพริออรี และอัลกอริทึม FUP ยังได้นำเอาขั้นตอนการเชื่อมความสัมพันธ์ของอัลกอริทึมเอพริออรีมาใช้เพื่อสร้างความสัมพันธ์ของไอเทมเซต และได้นำการทำงานบางส่วนที่เป็นจุดด้อยของเอพริออรีมาปรับปรุง เพื่อทำให้การค้นหากฎความสัมพันธ์ของข้อมูลมีประสิทธิภาพมากกว่าอัลกอริทึมเอพริออรี



รูปที่ 2.11 วิธีการของอัลกอริทึม FUP สำหรับค้นหา Large 1-itemset

การค้นหา FUP ในส่วนแรกจะเป็นการค้นหา L_1' ดังนี้

1. เป็นขั้นตอนทำการสแกนค้นหาไอเทมเซตในฐานข้อมูลเพิ่มใหม่ สำหรับทุกไอเทมเซตที่เป็นสมาชิกอยู่ภายใน L_1 (Large 1-itemset) ของฐานข้อมูลเดิม เพื่อปรับปรุงค่าสนับสนุนในไอเทมเซตที่กำลังพิจารณา ถ้าหากว่าไอเทมเซตนั้นมีค่าสนับสนุนที่น้อยกว่าค่าสนับสนุนขั้นต่ำของฐานข้อมูลปรับปรุง ($X.support_{UD} < s(D+d)$) ก็ไม่สามารถผ่านเกณฑ์ไปเป็นไอเทมเซตใน L'_1 ของฐานข้อมูลปรับปรุง ดังนั้นจะเรียกไอเทมเซตนั้นว่า loser และถ้าไอเทมเซตที่พิจารณามีค่าสนับสนุนที่มากกว่าหรือเท่ากับค่าสนับสนุนขั้นต่ำของฐานข้อมูลปรับปรุง ($X.support_{UD} >= s(D+d)$) ดังนั้นจะเรียกไอเทมเซตที่ผ่านเกณฑ์ดังกล่าวว่า Winner

2. ดังนั้นจากการสแกนค้นหาในฐานข้อมูลเพิ่มใหม่จากขั้นตอนดังกล่าว จะทำการสร้าง C_1 (Candidate 1-itemset) เพื่อเก็บไอเทมเซตที่ไม่ได้เป็นสมาชิกของ L_1 ในฐานข้อมูลเดิม ($X \notin L_1$) โดยไอเทมเซตใดที่เป็นสมาชิกใน C_1 ($X \in C_1$) หากค่าสนับสนุนของไอเทมเซตใดมีค่าสนับสนุน

ต่ำกว่าค่าสนับสนุนขั้นต่ำของฐานข้อมูลเพิ่มใหม่ ($X.support_d < s \times d$) จะถูกตัดออกไปจาก C_1 เรียกไอเทมเซตเหล่านั้นว่า loser เพราะว่ามันไม่สามารถเป็นไอเทมเซตใน L'_1 ได้ จะทำการเก็บไอเทมเซตที่ถูกตัดออกจาก C_1 ไว้ใน P เพื่อนำมาใช้ในการพิจารณาเมื่อมีการสแกนในฐานข้อมูลเดิม จะทำการตัดไอเทมเซต X ที่เป็นสมาชิกของ P ($X \in P$) ออกจากทรานแซคชั่น T ที่เป็นสมาชิกในฐานข้อมูลเดิม ($T \in DB$) เพราะไอเทมเซตนี้ไม่สามารถเป็น Large Itemset ได้ในรอบต่อไป

3. เมื่อได้ไอเทมเซตทั้งหมดใน C_1 นำไอเทมเซตที่มีมาพิจารณาว่าแต่ละไอเทมเซตมีค่าสนับสนุนมากกว่าหรือเท่ากับค่าสนับสนุนขั้นต่ำของฐานข้อมูลเพิ่มใหม่ สำหรับไอเทมเซตที่ผ่านเกณฑ์จะนำไปทำการสแกนค้นหาในฐานข้อมูลเดิม ว่ามีไอเทมเซตใดที่เหมือนกันกับไอเทมเซตใน C_1 เพื่อนับจำนวนค่าสนับสนุนของแต่ละไอเทมเซตนั้นว่ามีค่าเป็นเท่าใด เมื่อได้ค่าสนับสนุนของไอเทมเซตที่ค้นหาในฐานข้อมูลเดิมมารวมกับค่าสนับสนุนของไอเทมเซตใน C_1 แล้วนำค่ามาพิจารณาว่า ถ้าค่าสนับสนุนไอเทมเซตใดมีค่ามากกว่าหรือเท่ากับค่าสนับสนุนขั้นต่ำของฐานข้อมูลปรับปรุง ก็จะปรากฏอยู่ใน L'_1 ถ้าปรากฏว่าค่าสนับสนุนของไอเทมเซตไม่ผ่านค่าสนับสนุนขั้นต่ำของฐานข้อมูลปรับปรุง จะเรียกไอเทมเซตนั้นว่า Loser

การค้นหา FUP ในการค้นหาตั้งแต่ L'_2 เป็นต้นไป ดังนี้

1. เมื่อทำในรอบแรกเสร็จแล้วจะได้ L'_1 ของฐานข้อมูลปรับปรุงและจะทำการหา C_2 (Candidate 2-itemsets) ด้วยการเชื่อมความสัมพันธ์ L'_1 กับ L'_1 เมื่อเสร็จการเชื่อมความสัมพันธ์ จะตัดไอเทมเซตที่มีไอเทมเซตที่เหมือนกับ L_2 ออกจาก C_2

2. จากนั้นจะมาพิจารณาขั้นตอนในกรณี $L_{k-1} - L'_{k-1}$

ตัวอย่าง $L_1 = \{I_1, I_2, I_3\}$ $L_2 = \{I_1I_2, I_2I_3\}$ และ $L'_1 = \{I_1, I_2, I_4\}$

ดังนั้นจะพบว่าไอเทม I_1 และ I_2 ของ L_1 ก็อยู่ใน L'_1 มีเพียง I_3 ใน L_1 เพียงไอเทมเดียวเท่านั้นที่ไม่เหมือนไอเทมใน L'_1 ดังนั้นจะตัดไอเทมเซตที่มี I_3 เป็นสมาชิกที่ไอเทมเซตนั้นอยู่ใน L_2 ออกไปคือ I_2I_3 เพราะไม่สามารถเป็น Large Itemset ได้ ทำให้เหลือเพียงไอเทมเซต $\{I_1I_2\}$

3. เป็นขั้นตอนที่ทำการสแกนค้นหาไอเทมเซตในฐานข้อมูลเพิ่มใหม่และนับค่าสนับสนุนของไอเทมเซต โดยไอเทมเซตใดที่ไม่เหมือนกับไอเทมเซตใน W จะถูกจัดเก็บใน C

4. เป็นขั้นตอนที่ลดการค้นหาในในฐานข้อมูลเพิ่มใหม่ เพื่อเป็นประโยชน์ในรอบถัดไป

5. สำหรับทุกไอเทมเซต X ที่เป็นสมาชิกอยู่ภายใน W จะถูกนำมาพิจารณา ถ้าไอเทมเซตใดที่มีค่าสนับสนุนมากกว่าหรือเท่ากับค่าสนับสนุนขั้นต่ำของฐานข้อมูลปรับปรุง ไอเทมเซตนั้นก็จะปรากฏอยู่ใน L'_k

6. สำหรับไอเทมเซต X ที่เป็นสมาชิกอยู่ภายใน C จะถูกนำมาพิจารณาว่าถ้าไอเทมเซตนั้นมีค่าสนับสนุนน้อยกว่าค่าสนับสนุนขั้นต่ำของฐานข้อมูลเพิ่มใหม่ ก็จะทำการตัดไอเทมเซตดังกล่าวออกจาก C

7. นำแต่ละไอเทมเซตของ C ไปสแกนหาในทุกทรานแซคชั่นของฐานข้อมูลเดิม เพื่อนับค่าสนับสนุนของไอเทมเซต

8. เป็นขั้นตอนที่ลดการค้นหาในฐานข้อมูลเดิม เพื่อเป็นประโยชน์ในรอบถัดไป

9. สำหรับทุกไอเทมเซต X ที่เป็นสมาชิกของ C ถ้าไอเทมเซตที่พิจารณาใดมีค่าสนับสนุนมากกว่าหรือเท่ากับค่าสนับสนุนขั้นต่ำของฐานข้อมูลปรับปรุง หากไอเทมเซตที่กำลังพิจารณานั้นผ่านเกณฑ์ค่าสนับสนุนขั้นต่ำดังกล่าวแล้วก็จะปรากฏอยู่ใน L'_k

ข้อเด่นของอัลกอริทึม FUP

1. มีการนำ Large Itemsets ที่เคยค้นหาจากฐานข้อมูลเดิมนำมาใช้ให้เกิดประโยชน์ เพื่อลดการค้นหาซ้ำในฐานข้อมูลเดิม

2. สามารถลดจำนวน Candidate Itemsets ที่ได้จากฐานข้อมูลเพิ่มใหม่ เพื่อนำไปใช้ค้นหาในฐานข้อมูลเดิมทำให้มีการค้นหาน้อยลง โดย Candidate Itemsets จะทำการค้นหาเฉพาะไอเทม-เซตที่ไม่เคยมีมาก่อนใน Large Itemsets ในฐานข้อมูลเดิม

ข้อด้อยของอัลกอริทึม FUP

เป็นอัลกอริทึมสำหรับการค้นหากฎความสัมพันธ์ที่แสดงเพียงมิติเดียวเท่านั้น เนื่องจากในปัจจุบันมีการเก็บทรานแซคชั่นในฐานข้อมูลแบบหลายมิติ

บทที่ 3
การค้นหากฎความสัมพันธ์แบบมิติผสมสำหรับการเพิ่มข้อมูล

3.1 การค้นหากฎความสัมพันธ์แบบมิติผสมสำหรับการเพิ่มข้อมูล

ปัจจุบันองค์กรส่วนใหญ่จะมีการจัดเก็บข้อมูลที่มีจำนวนมากลงในฐานข้อมูล โดยทั่วไปแล้วฐานข้อมูลมีรูปแบบการจัดเก็บทรานแซคชั่นข้อมูลแบบหลายมิติ เพื่อจัดเก็บรายละเอียดคุณลักษณะที่มีของข้อมูล ดังนั้นการค้นหากฎความสัมพันธ์ของทรานแซคชั่นข้อมูลเพียงหนึ่งมิติที่สนใจเฉพาะมิติเดียว ไม่สามารถตอบโจทย์การค้นหากฎความสัมพันธ์ในมิติอื่นที่มีได้ทั้งหมด จึงทำให้ต้องมีการปรับเปลี่ยนวิธีการค้นหา เพื่อให้สามารถนำไปใช้ค้นหากฎความสัมพันธ์ของข้อมูลที่มีในแต่ละมิติ ที่สามารถใช้กับทรานแซคชั่นข้อมูลแบบหลายมิติในฐานข้อมูลได้

ดังนั้นผู้วิจัยจึงเล็งเห็นความสำคัญของผลลัพธ์ที่อาจเป็นความรู้ที่เกิดขึ้นใหม่ จำเป็นจะต้องใช้วิธีการสำหรับการค้นหากฎความสัมพันธ์ข้อมูลแบบหลายมิติ ที่มีความสามารถแสดงความสัมพันธ์ที่มีในมิติต่างๆออกมาได้ว่าข้อมูลใดในมิติใดมีความสัมพันธ์กันอย่างไร

TID	Order ID
1	bread, jam, milk
2	bread, ice cream, jam
3	cola, ice cream
4	bread, cola, icecream
5	beer, peanuts

รูปที่ 3.1 แสดงฐานข้อมูลที่เก็บทรานแซคชั่นมิติเดียว

TID	Age	Area	Order ID
1	20…29	Bangkok	bread, jam, milk
2	20…29	Bangkok	bread, ice cream, jam
3	20…29	Chiang Mai	cola, ice cream
4	30…39	Chiang Mai	bread, cola, ice cream
5	30…39	Ayutthaya	beer, peanuts

รูปที่ 3.2 แสดงฐานข้อมูลที่เก็บทรานแซคชั่นหลายมิติ

ในรูปที่ 3.1 แสดงฐานข้อมูลที่จัดเก็บทรานแซคชั่นมิติเดียว มีรูปแบบการเก็บข้อมูลเพียงแอททริบิวต์หลัก (main attributes) คือ แอททริบิวต์ Order ID มีมิติการซื้อสินค้าเพียงมิติเดียว

ในฐานข้อมูลที่เก็บทรานแซคชั่นหลายมิติแสดงในรูปที่ 3.2 ประกอบด้วย มิติอายุ มิติพื้นที่ และมิติการซื้อสินค้า ซึ่งแอททริบิวต์รอง (subordinate attribute) คือแอททริบิวต์ Age และ Area และแอททริบิวต์หลัก (main attributes) คือแอททริบิวต์ Order ID

งานวิจัย Mining Conditional Hybird-Dimension Association Rules on The Basis of Multi-Dimensional Transaction Database[8] ได้ศึกษาการค้นหากฎความสัมพันธ์แบบมิติผสม ในฐานข้อมูลที่เก็บทรานแซคชั่นข้อมูลแบบหลายมิติ ซึ่งได้นำวิธีสร้างกฎความสัมพันธ์ มาใช้ในส่วนของการค้นหากฎความสัมพันธ์แบบมิติผสมไปใช้ในอัลกอริทึมเอพริออรี เพื่อให้สามารถค้นหากฎความสัมพันธ์ของข้อมูลแบบมิติผสมในฐานข้อมูลที่มีการจัดเก็บทรานแซคชั่นข้อมูลแบบหลายมิติได้ โดยที่เจ้าของงานวิจัยเล็งเห็นว่าผลของกฎความสัมพันธ์ที่ได้แบบหลายมิติ ย่อมดีกว่าได้จากกฎความสัมพันธ์มิติเดียว และการหากฎความสัมพันธ์แบบมิติผสมน่าจะมีผลลัพธ์หลากหลายกว่าการค้นหากฎความสัมพันธ์แบบระหว่างมิติ

โดยการค้นหากฎความสัมพันธ์ของข้อมูลแบบระหว่างมิตินั้น ยังมีข้อจำกัดอยู่บางส่วนด้วยวิธีการค้นหากฎความสัมพันธ์ข้อมูลระหว่างมิติ ไม่สามารถค้นหากฎความสัมพันธ์ระหว่างภายในแอททริบิวต์หลักของทรานซั่นในฐานข้อมูล ซึ่งทำให้ไม่พบความรู้ในส่วนนี้

งานวิจัย Maintenance of Discovered Association Rules in Large Databases: An Incremental Updating Technique[3] สามารถนำมาปรับใช้กับการค้นหากฎความสัมพันธ์แบบมิติผสมสำหรับการเพิ่มข้อมูล เพื่อให้สามารถค้นหากฎความสัมพันธ์ได้อย่างครบถ้วนทุกแอททริบิวต์ของทรานแซคชั่นข้อมูลแบบหลายมิติในฐานข้อมูล ที่การค้นหากฎความสัมพันธ์แบบระหว่างมิติไม่สามารถทราบความสัมพันธ์ที่มีระหว่างภายในแอททริบิวต์หลักที่สนใจได้ ทำให้เห็นถึงจุดเด่นของการค้นหากฎความสัมพันธ์แบบมิติผสม และนำการค้นหากฎความสัมพันธ์แบบมิติผสมมาใช้ เนื่องจากมีความสามารถเพิ่มขึ้นด้านการค้นหาภายในแอททริบิวต์หลักที่สนใจ ซึ่งการค้นหากฎความสัมพันธ์แบบมิติผสมทำให้ได้ผลลัพธ์ของกฎความสัมพันธ์มีความหลากหลาย

งานวิจัยนี้จึงนำเสนอการค้นหากฎความสัมพันธ์แบบมิติผสมภายใต้ชื่อว่า HDFUP โดยนำอัลกอริทึม FUP มาปรับปรุง เพื่อให้สามารถค้นหากฎความสัมพันธ์แบบมิติผสมในฐานข้อมูลที่มีทรานแซคชั่นข้อมูลแบบหลายมิติ โดยได้นำขั้นตอนการสร้างความสัมพันธ์ของข้อมูลแบบมิติผสมมาใช้ปรับในขั้นตอนการเชื่อมความสัมพันธ์ของอัลกอริทึม FUP โดยได้นำหลักการเชื่อมความสัมพันธ์จากหัวข้อ 2.1.2 การสร้างความสัมพันธ์ในกฎความสัมพันธ์แบบมิติผสม นำมาสร้างความสัมพันธ์ของข้อมูล ให้สามารถใช้กับทรานแซคชั่นข้อมูลแบบหลายมิติ และมีการปรับเปลี่ยนบางส่วนของอัลกอริทึม FUP เพื่อให้ค้นหากฎความสัมพันธ์แบบมิติผสมกับฐานข้อมูลที่มีการเก็บทรานแซคชั่นข้อมูลหลายมิติได้

Input: DB: the original database (with its size equal to D);

L_k: the set of all large k- itemsets in DB, k= 1, …, n;

db: an increment database (with its size equal to d);

Output: L': The set of all large itemsets in DB ∪ db.

Method:The 1st iteration:/* find L'_1, the set of all large 1-itemsets in DB ∪ db */

W = L_1; C = ϕ; L_1'= ϕ; /* W: winners, C: candidate sets, L'_1: initialized */

for all T ∈ db do /* สแกนหาใน db */

for all 1-itemset X ⊆ T do {

if X ∈ W then $X.support_d$++;

else {

if X ∉ C

then { C = C ∪ {X}; $X.support_d$ = 0;}

$X.support_d$++;

}/* จัดเก็บใน C และเพิ่มค่าสนับสนุน */

};

for all X ∈ W do /* นำค่า W ที่ผ่านเกณฑ์สนับสนุนขั้นต่ำเก็บใน L'_1 */

if $X.support_{UD} \geq s \times (D + d)$

then $L_1' = L'_1 \cup \{X\}$;

for all X ∈ C do /*ตัดค่า Candidate set ในC*/

if $X.support_d < s \times d$

then C = C – {X};

for all T ∈ DB do /* สแกนหาใน DB */

for all 1-itemset X ⊆ T do

if X ∈ C then $X.support_D$++;

for all X ∈ C do /* ค่าที่ผ่านเกณฑ์สนับสนุนขั้นต่ำนำไปเก็บไว้ใน L_1' */

if $X.support_{UD} \geq s \times (D + d)$

then $L_1' = L_1' \cup \{X\}$;

return L_1'. /* สิ้นสุดรอบที่1 */

รูปที่ 3.3 อัลกอริทึม HDFUP สำหรับหาตั้งแต่ Large 1-itemset

```
The k-th iteration: /*for k >= 2, until either L'_k returned is empty or db = φ */
    W = L_k; L_k' = φ;  /*W: winners; L_k': Large Itemset in Update Database */
    if k = 2   /* ขนาด k-candidate sets */
     then {C = apriori_gen1(L'_k-1) - L_k; }
      else  {C = apriori_gen2(L'_k-1) - L_k; };
    for all k-itemset X∈ W do /* ตัด losers ใน W */
       for all (k-1)-itemset Y ∈ L_k-1 – L'_k-1 do
          if Y ⊆ X then { W = W – {X}; break;}
    for all T ∈ db do {  /* สแกนใน db */
          for all X ∈Subset(W,T) do X.support_d++;
          for all X ∈ Subset(C,T) do X.support_d++;
    }
    for all X ∈ W do  /* นำค่า W ที่ผ่านเกณฑ์สนับสนุนขั้นต่ำเก็บใน L'_k */
       if  X.support_UD ≥ s × ( D+d )
         then L'_k = L'_k ∪ {X};
    for all X ∈ C do /* ตัดค่า candidate sets ใน C*/
       if X.support_d < s × d then C = C – {X};
    for all T ∈ DB do { /* สแกนหาใน DB */
       for all X ∈ Subset(C,T) do X.support_D++;
    for all X ∈ C do
       if  X.support_UD ≥ s × (D + d)
         then L'_k = L'_k ∪ {X};
    return L_k'. /* สิ้นสุดการทำงานที่ k-th iteration */
```

รูปที่ 3.4 อัลกอริทึม HDFUP สำหรับหาตั้งแต่ Large 2-itemsets

```
procedure apriori_gen1 (L'k-1: Large itemsets)
{
    C = null;
    for each l1 ∈ L'k-1
      for each l2 ∈ L'k-1
        if not( subatt (l1) and subatt (l2) )
        //กรณีไม่เป็น subordinate attribute เดียวกัน
         then {
              c = l1 ⊳⊲ l2 ; // เป็นการเชื่อมความสัมพันธ์แบบ intradimensional จาก
หัวข้อ 2.1.2
              InsertInto C ;
         }
}
```

รูปที่ 3.5 procedure apriori_gen1

```
procedure apriori_gen2 (L'k-1: Large itemsets)
{
    C = null;
    for each l1 ∈ L'k-1
      for each l2 ∈ L'k-1
        if (mainatt (l1) and mainatt (l2))
        // ในกรณีที่ main attribute
         then // intradimension join {
           if (l1[1] = l2[1]) ∧ (l1[2] = l2[2]) ∧ ... ∧ (l1[k-2] = l2[k-2]) ∧
             (l1[k-1] < l2[k-1])
            then {
              c = l1 ⊳⊲ l2 ;  //เป็นการเชื่อมความสัมพันธ์แบบ intradimensional
จากหัวข้อ 2.1.2
               InsertInto C;
            }
         }
          else // interdimension join {
            if (l1[2]= l2[1]) ∧ (l1[3]= l2[2]) ∧ ... ∧ (l1[k-1] = l2[k-2]) ∧
              (l1[1] < l2[k-1])
            then {
              c = l1 ⊳⊲ l2 ; //เป็นการเชื่อมความสัมพันธ์แบบ interdimensional
จากหัวข้อ 2.1.2
               InsertInto C;
            }
        for each c ∈ C
          for each (k-1)-subset s of c
            if s ∉ L'k-1   then delete c from C;
```

รูปที่ 3.6 procedure apriori_gen2

นิยามและความหมายของคำในอัลกอริทึม HDFUP

1. DB (Original Database) คือ ฐานข้อมูลเดิม
2. db (Incremental Database) คือ ฐานข้อมูลเพิ่มใหม่
3. DB+db (Update Database, DB') คือ ฐานข้อมูลปรับปรุง เมื่อนำ db รวมเข้ากับ DB
4. L คือ ค่า Large Itemsets ของฐานข้อมูลเดิม เป็นไอเทมเซตที่มีค่าสนับสนุนของไอเทมเซตนั้นผ่านค่าสนับสนุนขั้นต่ำของฐานข้อมูลเดิม
5. L' คือ ค่า Large Itemsets ของฐานข้อมูลปรับปรุง เป็นไอเทมเซตที่มีค่าสนับสนุนของไอเทมเซตนั้นผ่านค่าสนับสนุนขั้นต่ำของฐานข้อมูลปรับปรุง
6. C คือ Candidate Itemsets เป็นไอเทมเซตที่คาดว่าจะสามารถไปเป็น L'

การทำงานของอัลกอริทึม HDFUP มีดังนี้

1. สแกนค้นหาไอเทมเซตในฐานข้อมูลเพิ่มใหม่ สำหรับทุกไอเทมเซตที่เป็นสมาชิกใน L_1 ของฐานข้อมูลเดิม เพื่อทำการปรับปรุงค่าสนับสนุนของไอเทมเซตดังกล่าวและนำมาพิจารณา ถ้าหากว่าค่าสนับสนุนไอเทมเซตที่ได้น้อยกว่าค่าสนับสนุนขั้นต่ำของฐานข้อมูลปรับปรุง จะไม่สามารถผ่านไปเป็นไอเทมเซตใน Large 1-itemset ของฐานข้อมูลปรับปรุง (L'_1) และค่าสนับสนุนที่มีค่ามากกว่าหรือเท่ากับค่าสนับสนุนขั้นต่ำของฐานข้อมูลปรับปรุงสามารถผ่านไปเป็นไอเทมเซตใน Large 1-itemset ของฐานข้อมูลปรับปรุง

2. เมื่อเสร็จสิ้นขั้นตอนค้นหาในฐานข้อมูลเพิ่มใหม่แล้วก็ทำการสร้าง C เพื่อเก็บไอเทมเซตที่ไม่ได้เป็นสมาชิกของ L_1 โดยไอเทมเซตใดที่เป็นสมาชิกอยู่ภายใน C จะถูกนำมาพิจารณาค่าสนับสนุน ถ้าหากค่าสนับสนุนของไอเทมเซตใดใน C มีค่าสนับสนุนต่ำกว่าค่าสนับสนุนขั้นต่ำของฐานข้อมูลเพิ่มใหม่ ไอเทมเซตดังกล่าวจะถูกตัดออกไปจาก C เพราะว่าไอเทมเซตนั้นไม่มีโอกาสเป็นไอเทมเซตใน L'_1 ได้

3. เมื่อผ่านขั้นตอนข้างต้นจะได้ค่า C ที่มีไอเทมเซตซึ่งมีค่าสนับสนุนมากกว่าหรือเท่ากับค่าสนับสนุนขั้นต่ำของฐานข้อมูลเพิ่มใหม่ ทำการสแกนค้นหาในฐานข้อมูลเดิมว่ามีไอเทมเซตใดที่เหมือนกันกับไอเทมเซตใน C เพื่อทำการปรับปรุงค่าสนับสนุนของไอเทมเซตนั้นว่ามีค่าเท่าใด เมื่อได้ค่าสนับสนุนของไอเทมเซตดังกล่าวทั้งหมดแล้วจะนำมาเปรียบเทียบว่ามีค่ามากกว่าหรือเท่ากับค่าสนับสนุนขั้นต่ำของฐานข้อมูลปรับปรุงหรือไม่ ถ้าผ่านเกณฑ์ดังกล่าวก็จะเป็นไอเทมเซตใน L'_1

4. หลังจากทำรอบแรกเสร็จผลลัพธ์ที่ได้คือ L'_1 เป็น Large Itemsets ของฐานข้อมูลปรับปรุง ถัดมาจะเป็นขั้นตอนการหา C ด้วยการเชื่อมความสัมพันธ์ L'_1 กับ L'_1 ตาม procedure apriori_gen1 ซึ่งเป็นการสร้างความสัมพันธ์ระหว่างไอเทมเซตที่เกิดขึ้นทั้งหมดโดยมีข้อยกเว้นบางอย่างในการเชื่อมความสัมพันธ์ก็คือไม่สามารถเชื่อมความสัมพันธ์ กันด้วยแอททริบิวต์รองภายในแอททริบิวต์เดียวกันได้ เช่น แอททริบิวต์อายุ ซึ่งเป็นแอททริบิวต์รองไม่สามารถเชื่อม

ความสัมพันธ์กับแอททริบิวต์อายุที่อยู่ภายในแอททริบิวต์เดียวกันได้ และแอททริบิวต์พื้นที่ซึ่งเป็นแอททริบิวต์รองไม่สามารถเชื่อมความสัมพันธ์ กับแอททริบิวต์พื้นที่ภายในแอททริบิวต์เดียวกัน แต่แอททริบิวต์อายุที่เป็นแอททริบิวต์รองสามารถเชื่อมความสัมพันธ์ กับแอททริบิวต์พื้นที่ซึ่งเป็นแอททริบิวต์รองเช่นเดียวกันได้ เพราะเป็นแอททริบิวต์รองที่ต่างแอททริบิวต์กัน

เมื่อเสร็จสิ้นขั้นตอนเชื่อมความสัมพันธ์ทำให้ได้ไอเทมเซตของ C ที่มีขนาด 2-itemsets และจะนำมาพิจารณาเพื่อตัดไอเทมเซตใน C ที่มีเหมือนกับไอเทมเซตใน L_2 ออกไป เพื่อจะได้ไม่ต้องนำไอเทมเซตที่เคยค้นหาแล้วไปค้นหาซ้ำอีกภายในฐานข้อมูลเดิม สามารถนำเอา Large Itemset ที่เคยค้นหาไว้แล้วมาใช้ประโยชน์ได้อีก

5. พิจารณาขั้นตอน Y ว่าเป็นสมาชิกของ $L_{k-1} - L'_{k-1}$ หรือไม่ วิธีการพิจารณาเช่นเดียวกับหัวข้อ 2.2.1ในขั้นตอนการค้นหา FUP ที่ค้นหาตั้งแต่ L'_2 เป็นต้นไป จะได้ไม่ต้องทำการค้นหาในฐานข้อมูลเพิ่มใหม่อีก เพราะไอเทมเซตดังกล่าวนั้นไม่สามารถผ่านเกณฑ์ไปเป็น Large Itemsets ได้อย่างแน่นอน

6. สแกนในฐานข้อมูลเพิ่มใหม่เพื่อนับค่าสนับสนุนของไอเทมเซต X ที่หมือนกับไอเทมเซตใน W ในขั้นตอนนี้จะทำการค้นหาค่าสนับสนุนไอเทมเซต X ที่มีเหมือนกับไอเทมเซตที่อยู่ใน C ภายในฐานข้อมูลเพิ่มใหม่ด้วย

7. สำหรับไอเทมเซต X ที่เป็นสมาชิกอยู่ภายใน W ภายหลังการปรับปรุงค่าสนับสนุนแล้วจะถูกนำมาพิจารณาว่าค่าไอเทมเซตที่พิจารณามีค่าสนับสนุนมากกว่าหรือเท่ากับค่าสนับสนุนขั้นต่ำของฐานข้อมูลปรับปรุงหรือไม่ หากมากกว่าไอเทมเซตนั้นก็จะไปปรากฏอยู่ใน L'_2

8. สำหรับไอเทมเซต X ที่เป็นสมาชิกอยู่ภายใน C จะถูกนำมาพิจารณา หากไอเทมเซตใดมีค่าสนับสนุนน้อยกว่าค่าสนับสนุนขั้นต่ำของฐานข้อมูลเพิ่มใหม่ จะตัดไอเทมเซตดังกล่าวนั้นออกจาก C เพราะไม่สามารถไปเป็นค่า Large Itemsets ทำให้ไม่ต้องนำไปค้นหาในฐานข้อมูลเดิมอีก

9. นำแต่ละไอเทมเซตของ C ไปสแกนหาในทุกทรานแซคชั่นของฐานข้อมูลเดิม เพื่อนับค่าสนับสนุนของไอเทมเซต

10. สำหรับทุกไอเทมเซต X ที่เป็นสมาชิกอยู่ภายใน C นั้น ถ้าค่าสนับสนุนของไอเทมเซตที่พิจารณามากกว่าหรือเท่ากับค่าสนับสนุนขั้นต่ำของฐานข้อมูลปรับปรุง ไอเทมเซตนั้นจะไปปรากฏอยู่ภายใน L'_2

11. เมื่อเสร็จทุกขั้นตอนที่กล่าวมาแล้วจะได้ค่า L'_2 ในการทำงานของอัลกอริทึมในรอบถัดไปตั้งแต่ k ที่มีค่าตั้งแต่ 3 ขึ้นไป จะมีขั้นตอนที่แตกต่างจากรอบที่ 2 คือการสร้าง C เพราะว่าในรอบที่3 นั้นการทำงานต่างกันตรงขั้นตอนการเชื่อมความสัมพันธ์จะใช้ procedure apriori_gen2 แทน โดยการเชื่อมความสัมพันธ์นี้มีส่วนสำคัญในการทำงานที่มีการพิจารณารูปแบบของไอเทมเซตจาก 2 กรณี มีเงื่อนไขการทำงานคือ ถ้าเป็นการเชื่อมความสัมพันธ์ด้วยภายในแอททริบิวต์หลักเดียวกันเองใช้การเชื่อมความสัมพันธ์แบบมิติภายใน แต่หากเป็นรูปแบบอื่นๆให้ทำการเชื่อม

ความสัมพันธ์แบบระหว่างมิติ โดยในแต่ละรูปแบบจะมีการเช็คว่าไอเทมเซตดังกล่าวมีซับเซตอยู่ใน L_{k-1} ครบทุกซับเซตหรือไม่ ถ้ามีไม่ครบจะทำการตัดไอเทมเซตนั้นออกจาก C ภายหลังการเชื่อมความสัมพันธ์ จะทำการตรวจสอบว่าแต่ละไอเทมเซตใน C ที่ได้มีซับเซตแต่ละซับเซตทั้งหมดปรากฏอยู่ภายใน L_{k-1} อยู่ทุกซับเซตหรือไม่ ถ้าไอเทมเซตดังกล่าวไม่ปรากฏซับเซตอยู่ในทุกไอเทมเซตของ L_{k-1} จะทำการตัดออกจาก C ได้ทันที เพราะไอเทมเซตดังกล่าวไม่สามารถเป็น Large Itemsets

ตัวอย่างการทำงานของอัลกอริทึม HDFUP

กำหนดให้รายการขายสินค้ามีการเก็บรายละเอียดเกี่ยวกับอายุ พื้นที่อยู่ และสินค้า ซึ่งเป็นการเก็บข้อมูลในทรานแซคชั่นในฐานข้อมูลแบบหลายมิติ ค่าสนับสนุนขั้นต่ำ 20 เปอร์เซ็นต์

ID	Age	Area	Order ID
1	a	1	I_2, I_4
2	a	1	I_1, I_2, I_5
3	a	2	I_2, I_3
4	b	2	I_1, I_3
5	b	2	I_1, I_2, I_4
6	a	1	I_2, I_3
7	b	2	I_1, I_3

รูปที่ 3.7 แสดงทรานแซคชั่นในฐานข้อมูลเดิม

ID	Age	Area	Order ID
8	a	3	I_1, I_5, I_6
9	b	3	I_1, I_2, I_5, I_6, I_7
10	c	3	I_2, I_5
11	c	1	I_2, I_6
12	b	1	I_5, I_6
13	b	2	I_1, I_3, I_6

รูปที่ 3.8 แสดงทรานแซคชั่นในฐานข้อมูลเพิ่มใหม่

L_1	
Itemset	**Support**
{a}	4
{b}	3
{1}	3
{2}	4
$\{I_1\}$	4
$\{I_2\}$	5
$\{I_3\}$	4
$\{I_4\}$	2

L_2	
Itemset	**Support**
{a,1}	3
$\{a,I_2\}$	4
$\{a,I_3\}$	2
{b,2}	3
$\{b,I_1\}$	3
$\{b,I_3\}$	2
$\{1,I_2\}$	3
$\{2,I_1\}$	3
$\{2,I_2\}$	2
$\{2,I_3\}$	3
$\{I_1,I_2\}$	2
$\{I_1,I_3\}$	2
$\{I_2,I_3\}$	2
$\{I_2,I_4\}$	2

L_3	
Itemset	**Support**
$\{a,1,I_2\}$	3
$\{a,I_2,I_3\}$	2
$\{b,2,I_1\}$	3
$\{b,2,I_3\}$	2
$\{b,I_1,I_3\}$	2
$\{2,I_1,I_3\}$	2

L_4	
Itemset	**Support**
$\{b,2,I_1,I_3\}$	2

รูปที่ 3.9 แสดง Large Itemsets ของฐานข้อมูลเดิม

W		Scan db
Itemset	**Support**	**Support**
{a}	4	+1
{b}	3	+3
{1}	3	+2
{2}	4	+1
$\{I_1\}$	4	+3
$\{I_2\}$	5	+3
$\{I_3\}$	4	+1
$\{I_4\}$	2	+0

(ก)

W	
Itemset	**Support**
{a}	5
{b}	6
{1}	5
{2}	5
$\{I_1\}$	7
$\{I_2\}$	8
$\{I_3\}$	5
$\{I_4\}$	2

(ข)

รูปที่ 3.10 แสดงค่าไอเทมเซตใน W ภายหลังสแกนฐานข้อมูลเพิ่มใหม่

C	
Itemset	**Support**
{c}	2
{3}	3
$\{I_5\}$	4
$\{I_6\}$	5
$\{I_7\}$	1

(ก)

$X.support_d >= s \times d$

C	
Itemset	**Support**
{c}	2
{3}	3
$\{I_5\}$	4
$\{I_6\}$	5

(ข)

C		Scan DB
Itemset	**Support**	**Support**
{c}	2	+0
{3}	3	+0
$\{I_5\}$	4	+1
$\{I_6\}$	5	+0

(ค)

C	
Itemset	**Support**
{c}	2
{3}	3
$\{I_5\}$	5
$\{I_6\}$	5

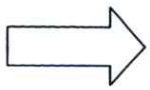
(ง)

รูปที่ 3.11 แสดงขั้นตอนหาค่าไอเทมเซตภายใน C และภายหลังสแกนฐานข้อมูลเดิม

W	
Itemset	**Support**
{a}	5
{b}	6
{1}	5
{2}	5
$\{I_1\}$	7
$\{I_2\}$	8
$\{I_3\}$	5
$\{I_4\}$	2

(ก)

$X.support_{UD} >= s \times (D+d)$



L'_1	
Itemset	**Support**
{a}	5
{b}	6
{1}	5
{2}	5
$\{I_1\}$	7
$\{I_2\}$	8
$\{I_3\}$	5

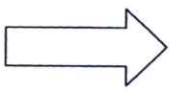
(ข)

รูปที่ 3.12 แสดงการหาค่าไอเทมเซตใน Large 1-itemsets จาก W

C	
Itemset	**Support**
{c}	2
{3}	3
$\{I_5\}$	5
$\{I_6\}$	5

(ก)

$X.support_{UD} >= s \times (D+d)$



L'_1	
Itemset	**Support**
{a}	5
{b}	6
{1}	5
{2}	5
{3}	3
$\{I_1\}$	7
$\{I_2\}$	8
$\{I_3\}$	5
$\{I_5\}$	5
$\{I_6\}$	5

(ข)

รูปที่ 3.13 แสดงการหา L'_1 ทั้งหมด

ขั้นตอนการทำงานเพื่อค้นหา L'_1

1. ในรูปที่ 3.7 แสดงรายการในแต่ละทรานแซคชั่นที่มีอยู่ภายในฐานข้อมูลเดิมก่อนทำการปรับปรุงซึ่งมีการเก็บทรานแซคชั่นแบบหลายมิติ รูปที่ 3.8 เป็นการแสดงรายการทรานแซคชั่นที่มีอยู่ภายในฐานข้อมูลปรับปรุงที่เพิ่มเข้ามาและมีมิติเหมือนกับฐานข้อมูลเดิม เพื่อนำมารวมเข้ากับฐานข้อมูลเดิม ทำให้เกิดเป็นฐานข้อมูลปรับปรุง และในรูปที่ 3.9 แสดงไอเทมเซตที่เป็นค่า Large Itemset ของฐานข้อมูลเดิมที่ได้จากการค้นหาก่อนหน้า

2. ในรูปที่ 3.10(ก) แสดงให้เห็นว่าค่าไอเทมเซต W ที่ได้มาจากค่า L_1 เมื่อมีการสแกนในฐานข้อมูลเพิ่มใหม่จะนำค่าไอเทมเซตที่มีใน W ไปเปรียบเทียบเพื่อพิจารณาว่าไอเทมเซตในฐานข้อมูลเพิ่มใหม่เหมือนกับค่าไอเทมเซตที่มีใน W หรือไม่ และทำการนับค่าสนับสนุน ส่วนในรูปที่ 3.10(ข) แสดงให้เห็นว่าในค่า W มีไอเทมเซตและค่าสนับสนุนขั้นต่ำที่ได้ภายหลังทำการสแกนหาภายในฐานข้อมูลเพิ่มใหม่

3.การหาค่าไอเทมเซต C ไอเทมเซตที่สแกนพบในฐานข้อมูลเพิ่มใหม่แต่ไม่ปรากฏอยู่ใน W จะทำการเพิ่มไอเทมเซตนั้นใน C แสดงในรูปที่ 3.11(ก) ค่าไอเทมเซตใน C จะนำมาพิจารณาว่าค่าสนับสนุนดังกล่าวมีค่าต่ำกว่าค่าสนับสนุนขั้นต่ำของฐานข้อมูลเพิ่มใหม่หรือไม่ หากไม่ผ่านเกณฑ์ดังกล่าวไอเทมเซตนั้นจะถูกตัดออกจาก C แสดงในรูปที่ 3.11(ข) เมื่อได้ค่า C จะนำไอเทมเซตทั้งหมดไปสแกนค้นหาภายในฐานข้อมูลเดิมเพื่อหาค่าสนับสนุนที่มีแต่ละไอเทมเซต แสดงในรูปที่ 3.11(ค) และในรูปที่ 3.11(ง) แสดงค่าไอเทมเซตทั้งหมดที่อยู่ใน C

4. นำค่าไอเทมเซตใน W จากรูปที่ 3.12 (ก) มาพิจารณา หากค่าสนับสนุนมีค่ามากกว่าหรือเท่ากับค่าสนับสนุนขั้นต่ำของฐานข้อมูลปรับปรุง นำไอเทมเซตดังกล่าวไปเป็นค่า Large itemsets ของฐานข้อมูลปรับปรุง แสดงในรูปที่ 3.12(ข)

5.นำค่าไอเทมเซตใน C จากรูปที่ 3.13(ก) มาพิจารณา หากค่าสนับสนุนมีค่ามากกว่าหรือเท่ากับค่าสนับสนุนขั้นต่ำของฐานข้อมูลปรับปรุง นำไอเทมเซตดังกล่าวไปเป็นค่า Large itemsets ของฐานข้อมูลปรับปรุง แสดงในรูปที่ 3.13(ข)

$L'_{k-1} \triangleright\triangleleft L'_{k-1}$	
Itemset	**Itemset**
{a,1}	$\{2,I_1\}$
{a,2}	$\{2,I_2\}$
{a,3}	$\{2,I_3\}$
$\{a,I_1\}$	$\{2,I_5\}$
$\{a,I_2\}$	$\{2,I_6\}$
$\{a,I_3\}$	$\{3,I_1\}$
$\{a,I_5\}$	$\{3,I_2\}$
$\{a,I_6\}$	$\{3,I_3\}$
{b,1}	$\{3,I_5\}$
{b,2}	$\{3,I_6\}$
{b,3}	$\{I_1,I_2\}$
$\{b,I_1\}$	$\{I_1,I_3\}$
$\{b,I_2\}$	$\{I_1,I_5\}$
$\{b,I_3\}$	$\{I_1,I_6\}$
$\{b,I_5\}$	$\{I_2,I_3\}$
$\{b,I_6\}$	$\{I_2,I_5\}$
$\{1,I_1\}$	$\{I_2,I_6\}$
$\{1,I_2\}$	$\{I_3,I_5\}$
$\{1,I_3\}$	$\{I_3,I_6\}$
$\{1,I_5\}$	$\{I_5,I_6\}$
$\{1,I_6\}$	

(ก)

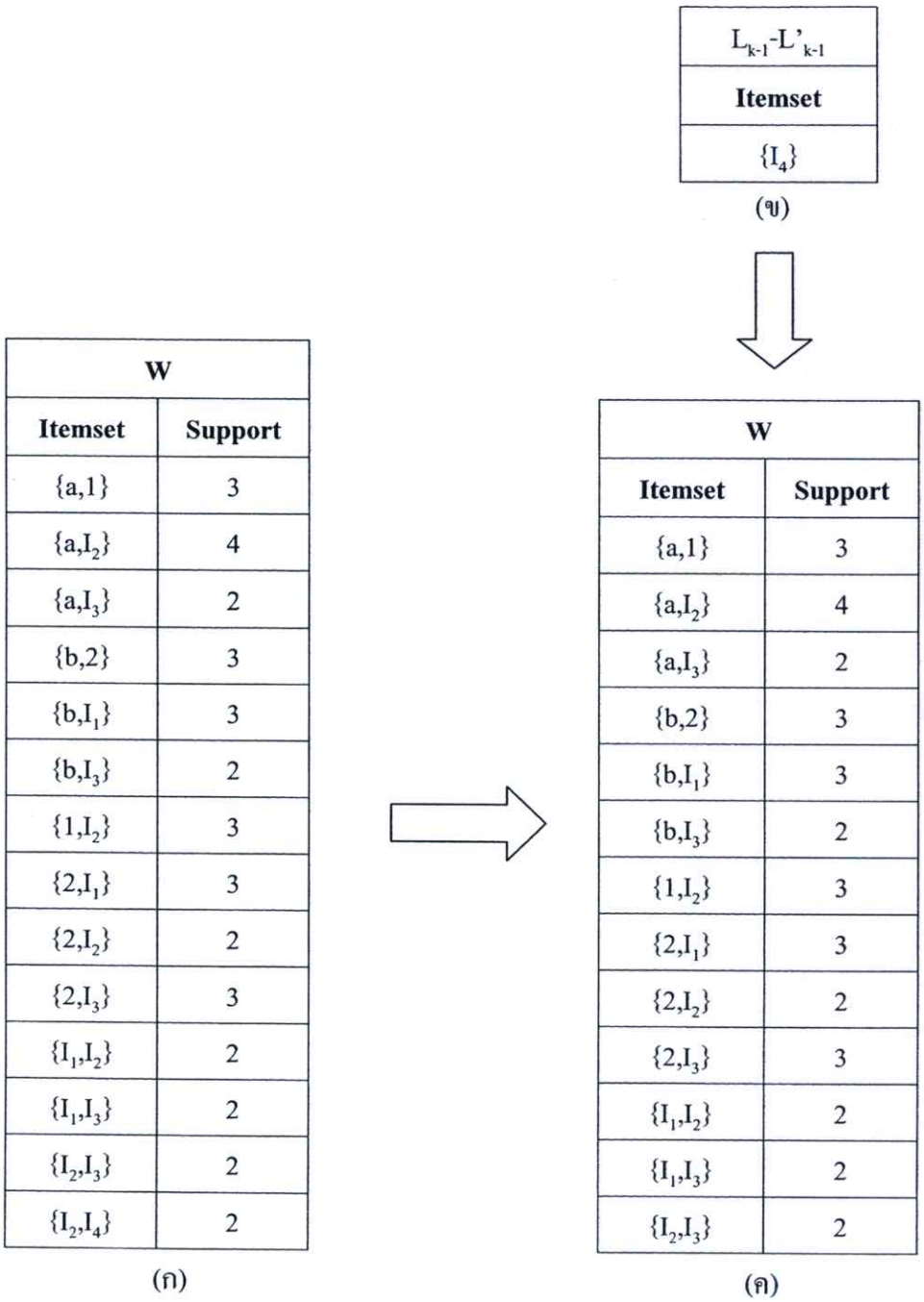
C
Itemset
{a,2}
{a,3}
$\{a,I_1\}$
$\{a,I_5\}$
$\{a,I_6\}$
{b,1}
{b,3}
$\{b,I_2\}$
$\{b,I_5\}$
$\{b,I_6\}$
$\{1,I_1\}$
$\{1,I_3\}$
$\{1,I_5\}$
$\{1,I_6\}$
$\{2,I_5\}$
$\{2,I_6\}$
$\{3,I_1\}$
$\{3,I_2\}$
$\{3,I_3\}$
$\{3,I_5\}$
$\{3,I_6\}$
$\{I_1,I_5\}$
$\{I_1,I_6\}$
$\{I_2,I_5\}$
$\{I_2,I_6\}$
$\{I_3,I_5\}$
$\{I_3,I_6\}$
$\{I_5,I_6\}$

(ข)

C	
Itemset	**Support**
{a,2}	0
{a,3}	1
$\{a,I_1\}$	1
$\{a,I_5\}$	1
$\{a,I_6\}$	1
{b,1}	1
{b,3}	1
$\{b,I_2\}$	1
$\{b,I_5\}$	2
$\{b,I_6\}$	3
$\{1,I_1\}$	0
$\{1,I_3\}$	0
$\{1,I_5\}$	1
$\{1,I_6\}$	2
$\{2,I_5\}$	0
$\{2,I_6\}$	1
$\{3,I_1\}$	2
$\{3,I_2\}$	2
$\{3,I_3\}$	0
$\{3,I_5\}$	3
$\{3,I_6\}$	2
$\{I_1,I_5\}$	2
$\{I_1,I_6\}$	3
$\{I_2,I_5\}$	2
$\{I_2,I_6\}$	2
$\{I_3,I_5\}$	0
$\{I_3,I_6\}$	1
$\{I_5,I_6\}$	3

(ค)

รูปที่ 3.14 แสดงผลค่าไอเทมเซต C ภายหลังการเชื่อมความสัมพันธ์รอบที่ 2



L_{k-1}-L'_{k-1}
Itemset
$\{I_4\}$

(ข)

W	
Itemset	**Support**
{a,1}	3
$\{a,I_2\}$	4
$\{a,I_3\}$	2
{b,2}	3
$\{b,I_1\}$	3
$\{b,I_3\}$	2
$\{1,I_2\}$	3
$\{2,I_1\}$	3
$\{2,I_2\}$	2
$\{2,I_3\}$	3
$\{I_1,I_2\}$	2
$\{I_1,I_3\}$	2
$\{I_2,I_3\}$	2
$\{I_2,I_4\}$	2

(ก)

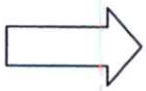
W	
Itemset	**Support**
{a,1}	3
$\{a,I_2\}$	4
$\{a,I_3\}$	2
{b,2}	3
$\{b,I_1\}$	3
$\{b,I_3\}$	2
$\{1,I_2\}$	3
$\{2,I_1\}$	3
$\{2,I_2\}$	2
$\{2,I_3\}$	3
$\{I_1,I_2\}$	2
$\{I_1,I_3\}$	2
$\{I_2,I_3\}$	2

(ค)

รูปที่ 3.15 แสดงการหา W ด้วยวิธีการตัดไอเทมที่ไม่สามารถเป็น L'_2

W		Scan db
Itemset	**Support**	**Support**
{a,1}	3	+0
{a,I_2}	4	+0
{a,I_3}	2	+0
{b,2}	3	+1
{b,I_1}	3	+2
{b,I_3}	2	+1
{1,I_2}	3	+1
{2,I_1}	3	+1
{2,I_2}	2	+0
{2,I_3}	3	+1
{I_1,I_2}	2	+1
{I_1,I_3}	2	+1
{I_2,I_3}	2	+0

(ก)



W	
Itemset	**Support**
{a,1}	3
{a,I_2}	4
{a,I_3}	2
{b,2}	4
{b,I_1}	5
{b,I_3}	3
{1,I_2}	4
{2,I_1}	4
{2,I_2}	2
{2,I_3}	4
{I_1,I_2}	3
{I_1,I_3}	3
{I_2,I_3}	2

(ข)

รูปที่ 3.16 แสดงผลค่าไอเทมเซตภายใน W ในรอบที่ 2

W	
Itemset	**Support**
{a,1}	3
{a,I_2}	4
{a,I_3}	2
{b,2}	4
{b,I_1}	5
{b,I_3}	3
{1,I_2}	4
{2,I_1}	4
{2,I_2}	2
{2,I_3}	4
{I_1,I_2}	3
{I_1,I_3}	3
{I_2,I_3}	2

(ก)

$X.support_{UD} >= s\times(D+d)$

L'_2	
Itemset	**Support**
{a,1}	3
{a,I_2}	4
{b,2}	4
{b,I_1}	5
{b,I_3}	3
{1,I_2}	4
{2,I_1}	4
{2,I_3}	4
{I_1,I_2}	3
{I_1,I_3}	3

(ข)

รูปที่ 3.17 แสดงการหาค่าไอเทมเซต L'_2 จากค่า W

C	
Itemset	**Support**
{a,2}	0
{a,3}	1
{a,I_1}	1
{a,I_5}	1
{a,I_6}	1
{b,1}	1
{b,3}	1
{b,I_2}	1
{b,I_5}	2
{ b,I_6}	3
{1,I_1}	0
{1,I_3}	0
{1,I_5}	1
{1,I_6}	2
{2,I_5}	0
{2,I_6}	1
{3,I_1}	2
{3,I_2}	2
{3,I_3}	0
{3,I_5}	3
{3,I_6}	2
{I_1,I_5}	2
{I_1,I_6}	3
{I_2,I_5}	2
{ I_2,I_6}	2
{I_3,I_5}	0
{I_3,I_6}	1
{I_5,I_6}	3

(ก)

$X.support_d >= s \times d$

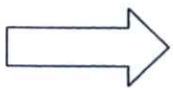
C	
Itemset	**Support**
{b,I_5}	2
{ b,I_6}	3
{1,I_6}	2
{3,I_1}	2
{3,I_2}	2
{3,I_5}	3
{3,I_6}	2
{I_1,I_5}	2
{I_1,I_6}	3
{I_2,I_5}	2
{ I_2,I_6}	2
{I_5,I_6}	3

(ข)

รูปที่ 3.18 แสดงการหาค่าไอเทมเซต C ผ่านค่าสนับสนุนขั้นต่ำในรอบที่ 2

C		Scan DB
Itemset	**Support**	**Support**
{b,I_5}	2	+0
{ b,I_6}	3	+0
{1,I_6}	2	+0
{3,I_1}	2	+0
{3,I_2}	2	+0
{3,I_5}	3	+0
{3,I_6}	2	+0
{I_1,I_5}	2	+1
{I_1,I_6}	3	+0
{I_2,I_5}	2	+1
{ I_2,I_6}	2	+0
{I_5,I_6}	3	+0

(ก)



C	
Itemset	**Support**
{b,I_5}	2
{ b,I_6}	3
{1,I_6}	2
{3,I_1}	2
{3,I_2}	2
{3,I_5}	3
{3,I_6}	2
{I_1,I_5}	3
{I_1,I_6}	3
{I_2,I_5}	3
{ I_2,I_6}	2
{I_5,I_6}	3

(ข)

รูปที่ 3.19 แสดงการหาค่าไอเทมเซต C ในรอบที่ 2 ภายหลังค้นหาในฐานข้อมูลเดิม

C	
Itemset	**Support**
$\{b,I_5\}$	2
$\{b,I_6\}$	3
$\{1,I_6\}$	2
$\{3,I_1\}$	2
$\{3,I_2\}$	2
$\{3,I_5\}$	3
$\{3,I_6\}$	2
$\{I_1,I_5\}$	3
$\{I_1,I_6\}$	3
$\{I_2,I_5\}$	3
$\{I_2,I_6\}$	2
$\{I_5,I_6\}$	3

(ก)

$X.support_{UD} >= s\times(D+d)$

L'_2	
Itemset	**Support**
{a,1}	3
$\{a,I_2\}$	4
{b,2}	4
$\{b,I_1\}$	5
$\{b,I_3\}$	3
$\{b,I_6\}$	3
$\{1,I_2\}$	4
$\{2,I_1\}$	4
$\{2,I_3\}$	4
$\{3,I_5\}$	3
$\{I_1,I_2\}$	3
$\{I_1,I_3\}$	3
$\{I_1,I_5\}$	3
$\{I_1,I_6\}$	3
$\{I_2,I_5\}$	3
$\{I_5,I_6\}$	3

(ข)

รูปที่ 3.20 แสดงการหาค่า L'_2 ทั้งหมด

ขั้นตอนการทำงานเพื่อค้นหา L'_2

1. ในรูปที่ 3.14 แสดงการหาค่า C ในขั้นตอนนี้เป็นการทำเชื่อมความสัมพันธ์ไอเทมเซตใน L'_1 กับ L'_1 ตาม procedure apriori_gen1 เมื่อได้ผลลัพธ์ไอเทมเซตจากการเชื่อมความสัมพันธ์แล้วดังในรูปที่ 3.14(ก) ให้ตัดไอเทมเซตที่มีเหมือนกับไอเทมเซตภายใน L_2 ออกทำให้ได้ค่า C ที่ไม่มีสมาชิก ไอเทมเซตใดเหมือนกับ L_2 แสดงในรูปที่ 3.14(ข)

2. สำหรับในการพิจารณาค่า W ที่ได้มาจาก L_2 รูปที่ 3.15(ก) ในขั้นตอนนี้พิจารณาเพิ่มในไอเทมเซตที่ L_1 มีต่างจากไอเทมเซตใน L'_1 ให้นำไอเทมเซตใน L_1 ที่มีสมาชิกไม่เหมือนกับ L'_1 ไปพิจารณาเพื่อตัดไอเทมเซตที่มีออกจากค่า W โดยในตัวอย่างนี้คือไอเทมเซต $\{I_4\}$ รูปที่ 3.15(ข) และค่านี้เป็นซับเซตของ $\{I_2,I_4\}$ ที่เป็นไอเทมเซตใน W เมื่อตัดออกจาก W เรียบร้อยแล้วจะได้ค่าไอเทมเซตที่เหลือทั้งหมดของ W แสดงในรูปที่ 3.15(ค)

3. ในรูปที่ 3.16(ก) แสดงให้เห็นค่าไอเทมเซตใน W เมื่อมีการสแกนในฐานข้อมูลเพิ่มใหม่ จะนำค่าไอเทมเซตที่มีอยู่ภายใน W ไปเปรียบเทียบเพื่อพิจารณาว่าไอเทมเซตในฐานข้อมูลเพิ่มใหม่ที่เหมือนกับค่าไอเทมเซตที่มีใน W และทำการนับค่าสนับสนุน และในรูปที่ 3.16(ข) แสดงค่าไอเทมเซตที่มีใน W ภายหลังทำการสแกนเพื่อค้นหาภายในฐานข้อมูลเพิ่มใหม่เรียบร้อยแล้ว

4. ในรูปที่ 3.17 แสดงการพิจารณาหาไอเทมเซตที่มีภายใน W ว่าไอเทมเซตใดมีค่าสนับสนุนมากกว่าหรือเท่ากับค่าสนับสนุนขั้นต่ำของฐานข้อมูลปรับปรุงก็จะปรากฏอยู่ใน L'_2

5. ในรูปที่ 3.14(ค) แสดงค่าสนับสนุนไอเทมเซตที่มีใน C ภายหลังทำการสแกนค้นหาภายในฐานข้อมูลเพิ่มใหม่

6. ในรูปที่ 3.18(ข) แสดงค่าไอเทมเซตที่ผ่านการคัดเลือกโดยพิจารณาจากค่าสนับสนุนขั้นต่ำของฐานข้อมูลเพิ่มใหม่โดยไอเทมเซตใดที่มีค่าสนับสนุนไม่ผ่านค่าสนับสนุนขั้นต่ำของฐานข้อมูลเพิ่มใหม่จะถูกตัดออกจากการเป็นสมาชิกไอเทมเซตภายใน C

7. ในรูปที่ 3.19(ก) เป็นแสดงค่าไอเทมเซตที่มีภายใน C แล้วนำค่าไอเทมเซตที่มีใน C นั้นไปค้นหาภายในฐานข้อมูลเดิมว่ามีค่าสนับสนุนเท่าใด ในรูปที่ 3.19(ข) แสดงค่าไอเทมเซตทั้งหมดที่มีอยู่ภาย C ภายหลังทำการค้นหาในฐานข้อมูลเดิมแล้ว

8.ในรูปที่ 3.20 แสดงการหา L'_2 ทั้งหมดภายหลังการนำค่าไอเทมเซตใน C มาพิจารณาจากค่าสนับสนุนขั้นต่ำของฐานข้อมูลปรับปรุง

$L'_{k-1} \triangleright\triangleleft L'_{k-1}$
Itemset
{a,1,I_2 }
{b,2,I_1}
{b,2,I_3}
{b, I_1,I_3}
{b,I_1,I_6}
{I_1,I_2,I_5}
{I_1,I_5,I_6}

(ก)

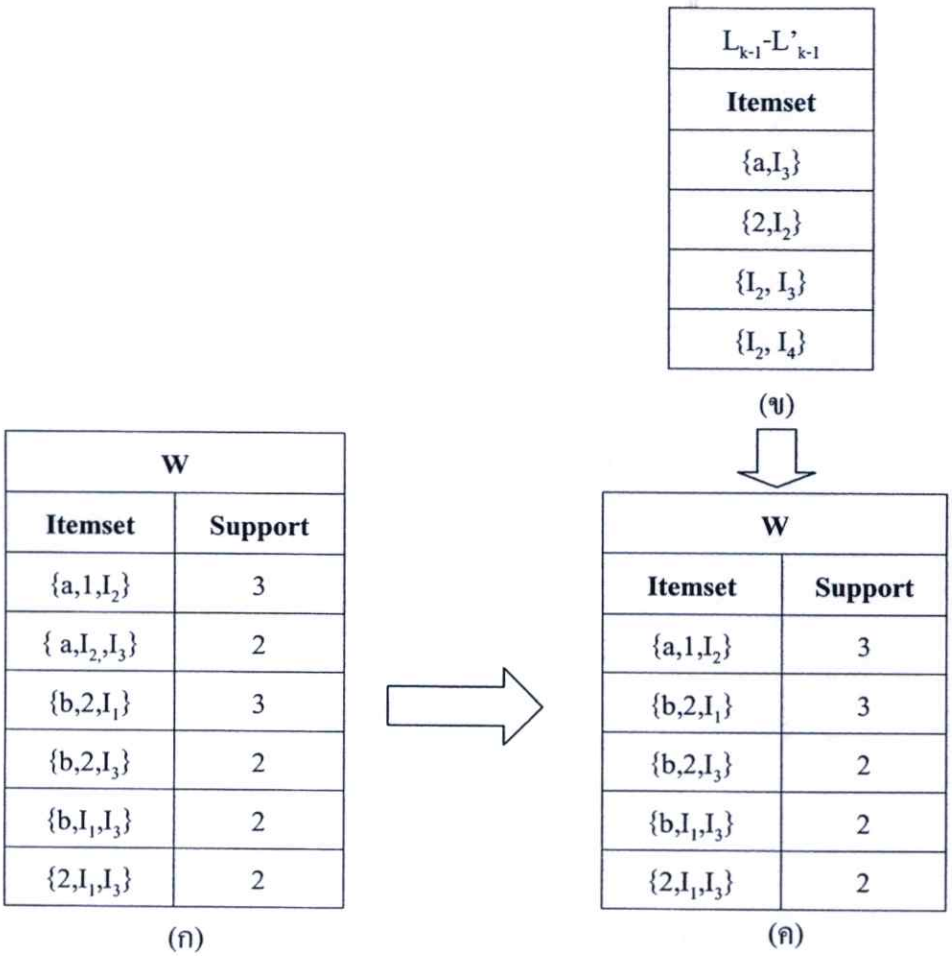
C
Itemset
{b, I_1,I_6}
{I_1,I_2,I_5}
{I_1,I_5,I_6}

(ข)

C	
Itemset	**Support**
{b, I_1,I_6}	2
{I_1,I_2,I_5}	1
{I_1,I_5,I_6}	2

(ค)

รูปที่ 3.21 แสดงผลค่าไอเทมเซต C ภายหลังการเชื่อมความสัมพันธ์รอบที่ 3



L_{k-1}-L'_{k-1}
Itemset
{a,I_3}
{2,I_2}
{I_2, I_3}
{I_2, I_4}

(ข)

W	
Itemset	**Support**
{a,1,I_2}	3
{ a,I_2,,I_3}	2
{b,2,I_1}	3
{b,2,I_3}	2
{b,I_1,I_3}	2
{2,I_1,I_3}	2

(ก)

W	
Itemset	**Support**
{a,1,I_2}	3
{b,2,I_1}	3
{b,2,I_3}	2
{b,I_1,I_3}	2
{2,I_1,I_3}	2

(ค)

รูปที่ 3.22 แสดงการหา W ด้วยวิธีการตัดไอเทมที่ไม่สามารถเป็น L_3

W		Scan db
Itemset	**Support**	**Support**
$\{a,1,I_2\}$	3	+0
$\{b,2,I_1\}$	3	+1
$\{b,2,I_3\}$	2	+1
$\{b,I_1,I_3\}$	2	+1
$\{2,I_1,I_3\}$	2	+1

(ก)

W	
Itemset	**Support**
$\{a,1,I_2\}$	3
$\{b,2,I_1\}$	4
$\{b,2,I_3\}$	3
$\{b,I_1,I_3\}$	3
$\{2,I_1,I_3\}$	3

(ข)

$X.support_{UD} >= s\times(D+d)$

L'_3	
Itemset	**Support**
$\{a,1,I_2\}$	3
$\{b,2,I_1\}$	4
$\{b,2,I_3\}$	3
$\{b,I_1,I_3\}$	3
$\{2,I_1,I_3\}$	3

(ค)

รูปที่ 3.23 แสดงการหาไอเทมเซต L'_3 จากค่า W

C	
Itemset	**Support**
$\{b, I_1,I_6\}$	2
$\{I_1,I_2,I_5\}$	1
$\{I_1,I_5,I_6\}$	2

(ก)

$X.support_d >= s\times d$

C	
Itemset	**Support**
$\{b, I_1,I_6\}$	2
$\{I_1,I_5,I_6\}$	2

(ข)

รูปที่ 3.24 แสดงการหาค่าไอเทมเซต C ผ่านค่าสนับสนุนขั้นต่ำในรอบที่ 3

C		Scan DB
Itemset	**Support**	**Support**
{b, I_1,I_6}	2	+0
{I_1,I_5,I_6}	2	+0

(ก)

C	
Itemset	**Support**
{b, I_1,I_6}	2
{I_1,I_5,I_6}	2

$X.support_{UD} >= s\times(D+d)$

L'_3	
Itemset	**Support**
{a,1,I_2}	3
{b,2,I_1}	4
{b,2,I_3}	3
{b,I_1,I_3}	3
{2,I_1,I_3}	3

(ค)

L'_4	
Itemset	**Support**
{b,2,I_1,I_3}	3

(ง)

รูปที่ 3.25 แสดงการหาค่า L'_3 ทั้งหมดและค่า L'_4

ขั้นตอนการทำงานเพื่อค้นหา L'_3

1. ในรูปที่ 3.21 แสดงการหาค่า C ในขั้นตอนนี้เป็นการทำเชื่อมความสัมพันธ์ L'_2 กับ L'_2 ตาม procedure apriori_gen2 จะได้ผลลัพธ์ปรากฏในรูปที่ 3.21(ก) ให้นำค่าไอเทมเซตที่มีเหมือนกับไอเทมเซตภายใน L_3 ออก ทำให้ได้ค่า C ที่ไม่มีสมาชิกไอเทมเซตใดเหมือนกับ L_3 แสดงในรูปที่ 3.21(ข) และในรูปที่ 3.21(ค) เป็นค่าไอเทมเซตภายใน C หลังการค้นหาภายในฐานข้อมูลเพิ่มใหม่

2. สำหรับในการพิจารณาค่า W ที่ได้มาจาก L_3 รูปที่ 3.22(ก) ในขั้นตอนนี้พิจารณาเพิ่มในไอเทมเซตที่ L_2 แตกต่างจากไอเทมเซตที่มีใน L'_2 ให้นำไอเทมเซตใน L_2 ที่มีสมาชิกไม่เหมือนกับ L'_2 นำไปพิจารณาเพื่อตัดไอเทมเซตที่มีออกจากค่า W โดยตัวอย่างนี้คือไอเทมเซต {a,I_3} {2,I_2} {I_2,I_3} {I_2,I_4} แสดงในรูปที่ 3.22(ข) และค่านี้เป็นซับเซตของ {a,I_2,I_3} ที่เป็นไอเทมเซตใน W เมื่อตัดออกจาก W เรียบร้อยแล้วจะได้ค่าไอเทมเซตที่เหลือทั้งหมดของ W แสดงในรูปที่ 3.22(ค)

3. ในรูปที่ 3.23 แสดงให้เห็นว่าค่าไอเทมเซต W เมื่อมีการสแกนในฐานข้อมูลเพิ่มใหม่จะนำค่าไอเทมเซตที่มีใน W ไปพิจารณาว่าไอเทมเซตในฐานข้อมูลเพิ่มใหม่ที่เหมือนกับค่าไอเทมเซต

ที่มีใน W และทำการนับค่าสนับสนุนแสดงในรูปที่ 3.21(ก) และในรูปที่ 3.21(ข) แสดงค่าไอเทมเซตที่มีใน W ภายหลังทำการสแกนเพื่อค้นหาภายในฐานข้อมูลเพิ่มใหม่เรียบร้อยแล้ว

4.ในรูปที่ 3.24(ก) แสดงค่าไอเทมเซตที่มีใน C ก่อนทำการสแกนค้นหาภายในฐานข้อมูลเพิ่มใหม่และค่าสนับสนุนที่นับได้ในการสแกนค้นหาภายในฐานข้อมูลเพิ่มใหม่ ในรูปที่ 3.24(ข) แสดงค่าไอเทมเซตที่มีอยู่ใน C ภายหลังทำการสแกนค้นหาภายในฐานข้อมูลเพิ่มใหม่

5. ในรูปที่ 3.23(ค) แสดงการพิจารณาหาไอเทมเซตที่มีภายใน W ว่า ไอเทมเซตใดสามารถผ่านค่าสนับสนุนขั้นต่ำของฐานข้อมูลปรับปรุงจะปรากฏใน L'_3

6. ในรูปที่ 3.24(ข) แสดงค่าไอเทมเซตที่ผ่านการคัดเลือกโดยพิจารณาจากค่าสนับสนุนขั้นต่ำของฐานข้อมูลเพิ่มใหม่ โดยไอเทมเซตใดที่มีค่าสนับสนุนไม่ผ่านค่าสนับสนุนขั้นต่ำของฐานข้อมูลเพิ่มใหม่โดยทำการตัดออกจากการเป็นสมาชิกไอเทมเซตภายใน C

7. ในรูปที่ 3.25(ก) แสดงค่าไอเทมเซตที่มีภายใน C แล้วนำค่าไอเทมเซตที่มีใน C นั้นไปค้นหาภายในฐานข้อมูลเดิมว่ามีค่าสนับสนุนเท่าใด ในรูปที่ 3.25(ข) แสดงค่าไอเทมเซตทั้งหมดที่มีอยู่ภาย C ภายหลังทำการค้นหาในฐานข้อมูลเดิมแล้ว ในรูปที่ 3.25(ค) แสดงการหา L'_3 ทั้งหมดภายหลังการนำค่าไอเทมเซตใน C มาพิจารณาจากค่าสนับสนุนขั้นต่ำของฐานข้อมูลปรับปรุง

8. ในรูปที่ 3.25(ง) เป็นรูปที่แสดงค่าไอเทมเซตที่มีอยู่ภายใน L'_4 ทั้งหมดโดยใช้วิธีการค้นหาเช่นเดียวกับการหาค่า L'_3 ทุกขั้นตอน

การสร้างกฎความสัมพันธ์

ตัวอย่าง ถ้ากำหนดให้ค่าความเชื่อมั่นขั้นต่ำคือ 70 เปอร์เซ็นต์ ไอเทมเซตที่มีอยู่ใน L'_3 คือ $\{a,1,I_2\}$ สามารถนำมาสร้างเป็นกฎความสัมพันธ์ได้คือ

1. $a \wedge 1$	$\rightarrow$	I_2	ค่าความเชื่อมั่นคือ	3/3	=	100 เปอร์เซ็นต์
2. $a \wedge I_2$	$\rightarrow$	1	ค่าความเชื่อมั่นคือ	3/4	=	75 เปอร์เซ็นต์
3. $1 \wedge I_2$	$\rightarrow$	a	ค่าความเชื่อมั่นคือ	3/4	=	75 เปอร์เซ็นต์
4. a	$\rightarrow$	$1 \wedge I_2$	ค่าความเชื่อมั่นคือ	3/5	=	60 เปอร์เซ็นต์
5. 1	$\rightarrow$	$a \wedge I_2$	ค่าความเชื่อมั่นคือ	3/5	=	60 เปอร์เซ็นต์
6. I_2	$\rightarrow$	$a \wedge 1$	ค่าความเชื่อมั่นคือ	3/8	=	37.5 เปอร์เซ็นต์

ดังนั้นสามารถสรุปได้ว่าไอเทมเซต $\{a,1,I_2\}$ มีกฎที่มีค่าความเชื่อมั่นที่ผ่านค่าความเชื่อมั่นขั้นต่ำเพียง 3 กฎเท่านั้นที่ผ่านค่าความเชื่อมั่นขั้นต่ำคือ กฎที่ 1, 2 และ 3

บทที่ 4

การทดลอง และผลการทดลอง

บทนี้จะกล่าวถึงชุดข้อมูลที่ใช้ในการทดลอง การทดลอง และผลการทดลอง ในงานวิจัยนี้ใช้โปรแกรม Microsoft Visual Studio 2008 และ SQL Server 2000 ในการทดลอง และการสร้างชุดข้อมูลสังเคราะห์ (Synthetic Dataset) ใช้โปรแกรม Matlab 7

4.1 ชุดข้อมูลที่ใช้ในการทดลอง

4.1.1 การสร้างชุดข้อมูลสังเคราะห์ (Synthetic Dataset)

ในงานวิจัย Fast algorithm for mining association rules [2] ได้มีการนำเสนอการสร้างชุดข้อมูลทรานแซคชั่นสังเคราะห์เพื่อใช้สำหรับประเมินประสิทธิภาพของอัลกอริทึมเพื่อเลียนแบบการซื้อและการขายสินค้าในร้านขายปลีก โดยใช้หลักการทางสถิติต่างๆ มาใช้ได้แก่ การสุ่มค่าความน่าจะเป็นเพื่อหาขนาดของไอเทมเซต, ขนาดของทรานแซคชั่น การสุ่มค่าไอเทมเพื่อนำไปใส่ในทรานแซคชั่นด้วยวิธีการแจกแจงแบบต่างๆ เช่น การแจกแจงปัวซอง, การแจกแจงแบบปกติ, การแจกแจงยูนิฟอร์ม, การแจกแจงแบบเอ็กซ์โพแนนเชียล เป็นต้น ชุดข้อมูลสังเคราะห์ที่แสดงแนวโน้มของสินค้าแต่ละไอเทมที่ถูกซื้อไปด้วยกันในแต่ละเซตของไอเทมสินค้า ดังนั้นขนาดของทรานแซคชั่นที่สร้างได้มาจากการจัดกลุ่มของไอเทมต่างๆ ด้วยค่าเฉลี่ยทำให้จำนวน Large Itemsets สูงสุดอาจประกอบด้วยขนาดของไอเทมที่ต่างกัน

ชุดข้อมูลสังเคราะห์จะมีพารามิเตอร์ต่างๆ ที่สำคัญดังนี้

1. N หมายถึงจำนวนของไอเทมของข้อมูลที่มีในฐานข้อมูล
2. | D | หมายถึงจำนวนของทรานแซคชั่นที่มีในฐานข้อมูล
3. | L | หมายถึงจำนวนชุดไอเทมเซตทั้งหมดที่สามารถเป็น Large Itemsets ได้
4. | I | หมายถึง จำนวนเฉลี่ยของไอเทมของชุดไอเทมเซตที่จะเป็น Large Itemsets

   เช่น | L | = 200, | I | = 4 หมายความว่าค่าเฉลี่ยของ | L | จำนวน 200 ชุด จะมีค่าเฉลี่ยของ Large Itemsets มีขนาดประมาณ 4 ไอเทม
5. | T | หมายถึงการกำหนดค่าเฉลี่ยของขนาดทรานแซคชั่นในชุดข้อมูลสังเคราะห์

ขั้นตอนแรกเป็นการสร้างชุดของจำนวนที่สามารถเป็น Large Itemsets ได้สูงสุดจำนวน | L | ชุดตามที่กำหนดไว้ โดยสร้างภายใต้ขนาดเฉลี่ยของขนาดสูงสุดที่สามารถเป็น Large Itemsets ได้ | I | ขั้นตอนต่อไปคือการเลือกไอเทมเข้าไปในทรานแซคชั่น โดยกำหนดค่าเฉลี่ยเท่ากับขนาดของทรานแซคชั่น | T |

4.1.2 การเตรียมชุดข้อมูลสำหรับการทดลองประกอบด้วยสองส่วนสำคัญคือ

1. ส่วนที่เป็นแอททริบิวต์หลักมีแอททริบิวต์เดียวเตรียมจากการสร้างชุดข้อมูลสังเคราะห์

2. ส่วนที่เป็นแอททริบิวต์รอง เมื่อได้แอททริบิวต์หลักและเตรียมการสร้างแอททริบิวต์รองจากการสร้างความสัมพันธ์ที่เป็นไปได้ของไอเทมที่มีในแอททริบิวต์รองระหว่างแอททริบิวต์ทุกแอททริบิวต์เป็นชุดแอททริบิวต์รอง จากนั้นในแต่ละชุดไอเทมเซตทั้งหมดที่สามารถเป็น Large Itemsets (L) จะทำการสุ่มชุดแอททริบิวต์รองสำหรับชุดไอเทมเซตทั้งหมดที่สามารถเป็น Large Itemsets ละ 1 ชุดแอททริบิวต์รอง จากนั้นในขั้นตอนคัดเลือกชุดแอททริบิวต์รองสำหรับแต่ละทรานแซคชั่นข้อมูลมีวิธีการคือ กำหนดให้ค่า Threshold = 0.7 ดังนั้นจะมีการสุ่มค่าตั้งแต่ 0 ถึง 1 ถ้าค่าที่ได้อยู่ตั้งแต่ 0 ถึง 0.7 จะนำชุดแอททริบิวต์รองมาใส่ในทรานแซคชั่นข้อมูลด้วยวิธีการเปรียบเทียบด้วยวิธีการนับจำนวนไอเทมของทรานแซคชั่นข้อมูล ใกล้เคียงกับชุดไอเทมเซตทั้งหมดที่สามารถเป็น Large Itemsets มากที่สุด โดยพิจารณาจากถ้าหากทรานแซคชั่นข้อมูลที่นำมาเปรียบเทียบมีจำนวนไอเทมที่เหมือนกับชุดไอเทมเซตทั้งหมดที่สามารถเป็น Large Itemsets มีหลายชุด จะพิจารณาว่าชุดไอเทมเซตทั้งหมดที่สามารถเป็น Large Itemsets นั้นมีชุดใดสั้นที่สุด ถ้ามีชุดที่สั้นเท่ากันจะนำชุดแอททริบิวต์รองชุดที่อยู่ลำดับท้ายสุดมาใช้เป็นแอททริบิวต์รองในทรานแซคชั่นข้อมูล หากค่าจากการสุ่มมากกว่าค่า 0.7 จะสุ่มชุดแอททริบิวต์รองที่เคยสร้างไว้มา 1 ชุดแอททริบิวต์รองเพื่อใช้เป็นแอททริบิวต์รองในทรานแซคชั่นข้อมูลนั้น

ในการทดลองได้แบ่งเป็น 3 ชุดการทดลอง

ชุดการทดลองที่ 1 และ 2 มีชุดข้อมูลการทดลองอย่างละ 3 ชุด

1. ทรานแซคชั่นจากชุดข้อมูลสังเคราะห์ T10I4 ทรานแซคชั่นของฐานข้อมูลเดิมจำนวน 20,000 ทรานแซคชั่น และฐานข้อมูลเพิ่มใหม่จำนวน 10,000 ทรานแซคชั่น และแอททริบิวต์รองจำนวน 2 แอททริบิวต์ (T10I4DB20Kdb10K)

2. ทรานแซคชั่นจากชุดข้อมูลสังเคราะห์ T10I4 ทรานแซคชั่นของฐานข้อมูลเดิมจำนวน 20,000 ทรานแซคชั่น และฐานข้อมูลเพิ่มใหม่จำนวน 6,000 ทรานแซคชั่น และจำนวนแอททริบิวต์รองจำนวน 2 แอททริบิวต์ (T10I4DB20Kdb6K)

3. ทรานแซคชั่นจากชุดข้อมูลสังเคราะห์ T10I4 ทรานแซคชั่นของฐานข้อมูลเดิมจำนวน 20,000 ทรานแซคชั่น และฐานข้อมูลเพิ่มใหม่จำนวน 2,000 ทรานแซคชั่นและจำนวนแอททริบิวต์รองจำนวน 2 แอททริบิวต์ (T10I4DB20Kdb2K)

ชุดการทดลองที่ 3 มีชุดข้อมูลจำนวน 3 ชุด

1. ทรานแซคชั่นจากชุดข้อมูลสังเคราะห์ T10I4 ทรานแซคชั่นของฐานข้อมูลเดิมจำนวน 20,000 ทรานแซคชั่น และฐานข้อมูลเพิ่มใหม่จำนวน 10,000 ทรานแซคชั่น และแอททริบิวต์รองจำนวน 10 แอททริบิวต์ (T10I4DB20Kdb10K)

2. ทรานแซคชั่นจากชุดข้อมูลสังเคราะห์ T10I4 ทรานแซคชั่นของฐานข้อมูลเดิมจำนวน 20,000 ทรานแซคชั่น และฐานข้อมูลเพิ่มใหม่จำนวน 6,000 ทรานแซคชั่น และแอททริบิวต์รองจำนวน 10 แอททริบิวต์ (T10I4DB20Kdb6K)

3. ทรานแซคชั่นจากชุดข้อมูลสังเคราะห์ T10I4 ทรานแซคชั่นของฐานข้อมูลเดิมจำนวน 20,000 ทรานแซคชั่น และฐานข้อมูลเพิ่มใหม่จำนวน 2,000 ทรานแซคชั่น และแอททริบิวต์รองจำนวน 10 แอททริบิวต์ (T10I4DB20Kdb2K)

ตารางที่ 4.1 แสดงข้อมูลบางส่วนของชุดการทดลองที่ 1

TID	SubAttribute1	SubAttribute2	OrderID
1	A045	E006	i005,i029,i067,i095
2	A035	E032	i001,i015,i028,i031,i032,i072,i076,i079,i080
3	A009	E021	i004,i010,i018,i042,i044,i054,i055,i065,i072
4	A016	E028	i003,i019,i029,i037,i059,i078,i091,i097,i098
5	A028	E001	i005,i026,i032,i041,i049,i056,i071,i079
6	A040	E033	i022,i023,i041,i047,i061,i070,i074,i077
7	A008	E044	i004,i009,i010,i016,i017,i053,i063,i087
8	A032	E040	i014,i019,i027,i032,i047,i051,i070,i078
9	A014	E029	i009,i025,i028,i034,i036,i044,i050

ตารางที่ 4.2 แสดงข้อมูลบางส่วนของชุดการทดลองที่ 2

TID	SubAttribute1	SubAttribute2	OrderID
1	A013	E009	i007,i014,i031,i037,i059,i078,i097
2	A033	E013	i005,i021,i026,i055,i063,i087
3	A016	E034	i009,i011,i025,i028,i031,i034,i036,i058,i078
4	A020	E030	i006,i008,i013,i019,i061,i071,i073,i078,i098
5	A019	E023	i004,i006,i019,i032,i053,i071,i073,i095,i098
6	A022	E030	i008,i030,i031,i033,i042,i048,i066,i074,i082
7	A018	E003	,i006,i008,i011,i031,i032,i033,i044,,i088,i097
8	A017	E023	i021,i023,i031,i032,i048,i056,i061,i075,i089
9	A002	E024	i002,i005,i022,i031,i048,i067,i078,i082, i085

ตารางที่ 4.3 แสดงข้อมูลบางส่วนของชุดการทดลองที่ 3

TID	Sub Attribute										OrderID
	1	2	3	4	5	6	7	8	9	10	
1	A002	B001	C004	D003	E003	F003	G003	H001	J001	K004	i026,i047,i070
2	A004	B003	C004	D001	E002	F003	G003	H002	J001	K004	i009,i014,i018,i021,i035,i053,i054,i058,i065
3	A004	B004	C004	D004	E004	F004	G004	H003	J001	K003	i002,i007,i010,i015,i022,i039,i045,i053,i082,i094
4	A003	B001	C002	D001	E001	F001	G003	H001	J004	K004	i005,i021,i026,i055,i063,i087
5	A002	B002	C002	D003	E004	F002	G004	H001	J002	K003	i002,i005,i007,i014,i023,i029,i034,i050,i075,i081
6	A002	B004	C001	D004	E001	F002	G004	H002	J002	K004	i009,i011,i014,i025,i028,i031,i034,i036,i058,i078
7	A001	B003	C001	D002	E004	F004	G002	H001	J003	K001	i002,i005,i008,i010,i016,i043,i052,i084,i088,i096
8	A001	B004	C002	D001	E002	F004	G003	H004	J003	K004	i014,i017,i024,i061,i062,i082
9	A004	B002	C003	D001	E002	F001	G003	H001	J004	K001	i003,i014,i027,i030,i036,i040,i042,i064,i078
10	A004	B002	C002	D004	E003	F003	G003	H004	J001	K002	i004,i010,i018,i042,i044,i054,i055,i065,i072

4.2 ผลการทดลอง

การทดลองทั้ง 3 ชุด เปรียบเทียบด้านประสิทธิภาพของอัลกอริทึม HDFUP ด้วยจำนวนไอเทมเซตจากการเชื่อมความสัมพันธ์ (join) สร้าง 2-Itemsets เปรียบเทียบกับอัลกอริทึม FUP และวัดประสิทธิภาพด้านเวลาในการประมวลผลของอัลกอริทึม HDFUP เปรียบเทียบกับอัลกอริทึม FUP และอัลกอริทึมอะพิโอริค้นหากฎความสัมพันธ์แบบมิติผสม (ในการทดลองจะเขียนแทนด้วย Apriori Hybrid) โดยวัดความถูกต้องของอัลกอริทึม HDFUP ด้วยค่า Large Itemsets ภายหลังการปรับปรุงฐานข้อมูล กับอัลกอริทึมอะพิโอริค้นหากฎความสัมพันธ์แบบมิติผสม และอัลกอริทึม FUP ต้องมีค่าเหมือนกัน

ผลการทดลองชุดที่ 1 และผลการทดลองชุดที่ 2 กำหนดค่าสนับสนุนขั้นต่ำที่ใช้ในการทดลองทั้งหมด 4 ค่าคือ 0.01, 0.02, 0.03, 0.04

ตารางที่ 4.4 ผลการทดลองชุดที่ 1 T10I4DB20Kdb10K

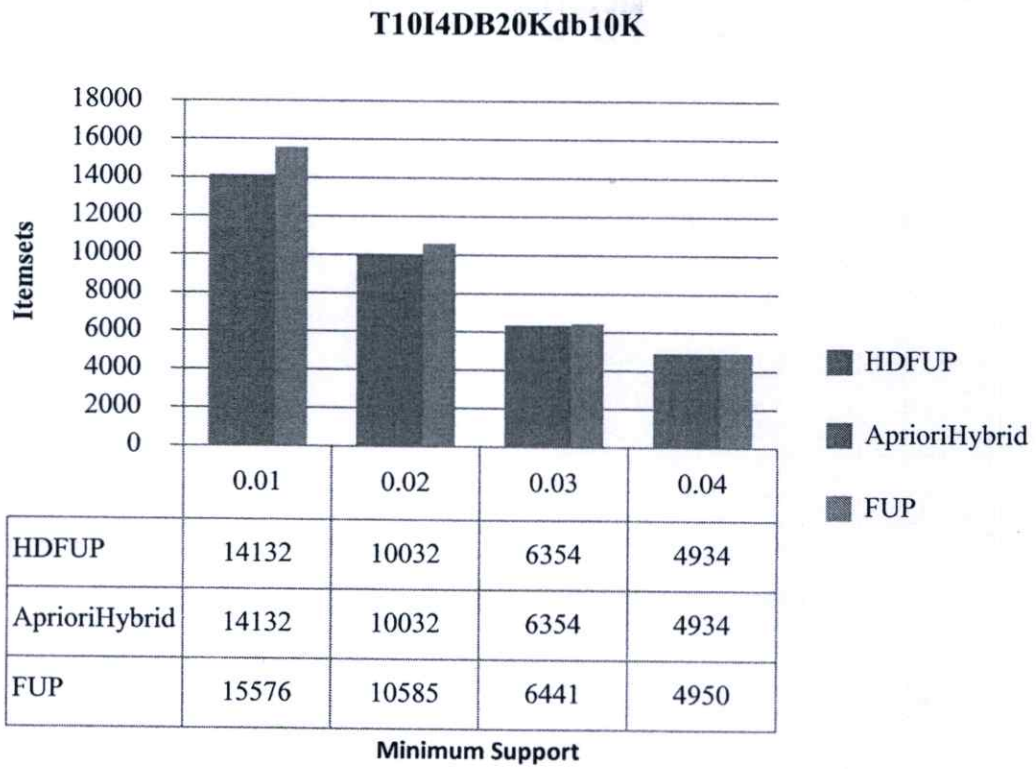
Min Support	Algorithm	Join Itemsets	Candidate Itemsets	Large Itemsets	Time(sec)
0.01	HDFUP	37,146	532	7,028	2668.337887
	AprioriHybrid	37,146	37,336	7,028	6693.392961
	FUP	38,590	532	7,028	2757.462059
0.02	HDFUP	11,735	90	1,194	570.5070021
	AprioriHybrid	11,735	11,925	1,194	1615.944438
	FUP	12,288	90	1,194	590.0242363
0.03	HDFUP	6,604	26	450	302.3441312
	AprioriHybrid	6,604	6,794	450	883.4295516
	FUP	6,691	26	450	305.5109365
0.04	HDFUP	4,986	11	204	227.3983995
	AprioriHybrid	4,986	5,176	204	685.8556116
	FUP	5,002	11	204	231.3544076

ตารางที่ 4.5 ผลการทดลองชุดที่ 1 T10I4DB20Kdb6K

Min Support	Algorithm	Join Itemsets	Candidate Itemsets	Large Itemsets	Time(sec)
0.01	HDFUP	37,490	523	7,079	1789.323209
	AprioriHybrid	37,490	37,680	7,079	5892.177149
	FUP	38,859	523	7,079	1849.788849
0.02	HDFUP	11,350	106	1,146	346.7730092
	AprioriHybrid	11,350	11,540	1,146	1359.370788
	FUP	11,836	106	1,146	357.9426279
0.03	HDFUP	6,508	47	456	184.6575243
	AprioriHybrid	6,508	6,698	456	753.9457
	FUP	6,591	47	456	187.5932
0.04	HDFUP	5,178	19	208	143.2118958
	AprioriHybrid	5,178	5,368	208	600.7726552
	FUP	5,203	19	208	143.5202521

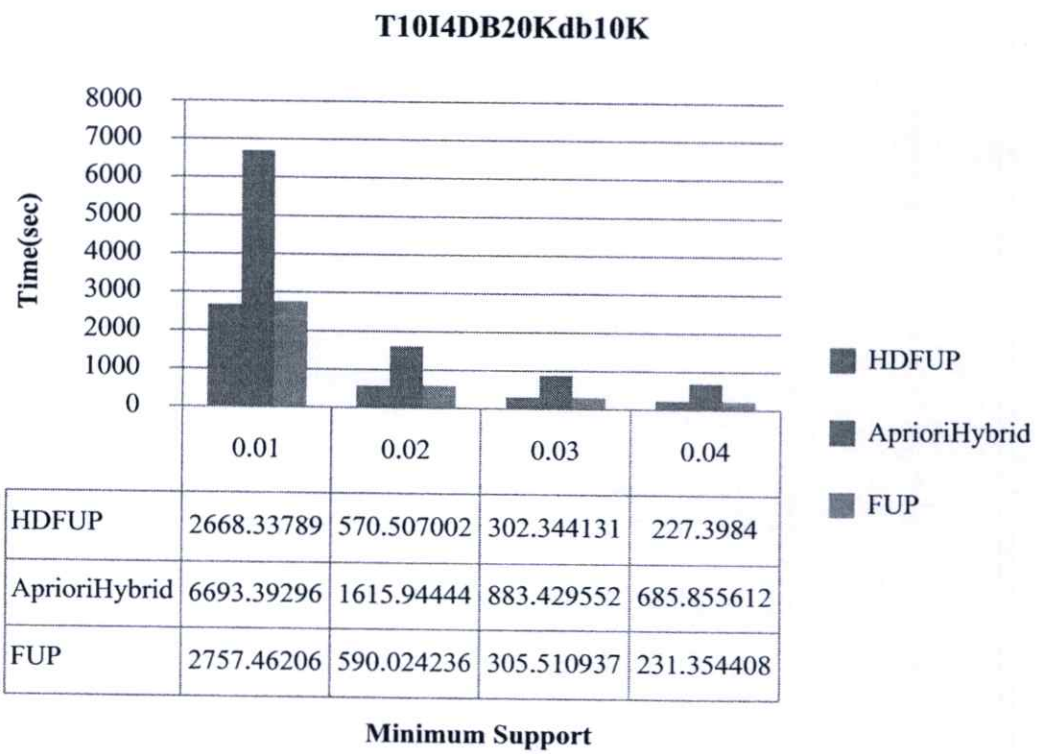
ตารางที่ 4.6 ผลการทดลองชุดที่ 1 T10I4DB20Kdb2K

Min Support	Algorithm	Join Itemsets	Candidate Itemsets	Large Itemsets	Time(sec)
0.01	HDFUP	37,528	547	6,865	862.0887141
	AprioriHybrid	37,528	37,718	6,865	4935.78387
	FUP	38,972	547	6,865	891.6007662
0.02	HDFUP	11,642	161	1,186	144.4406537
	AprioriHybrid	11,642	11,832	1,186	1179.705272
	FUP	12,149	161	1,186	148.7774611
0.03	HDFUP	6,709	55	452	70.1461243
	AprioriHybrid	6,709	6,899	452	655.5287513
	FUP	6,803	55	452	70.512124
0.04	HDFUP	5,275	18	210	50.6396903
	AprioriHybrid	5,275	5,465	210	515.6745057
	FUP	5,305	18	210	50.7624891



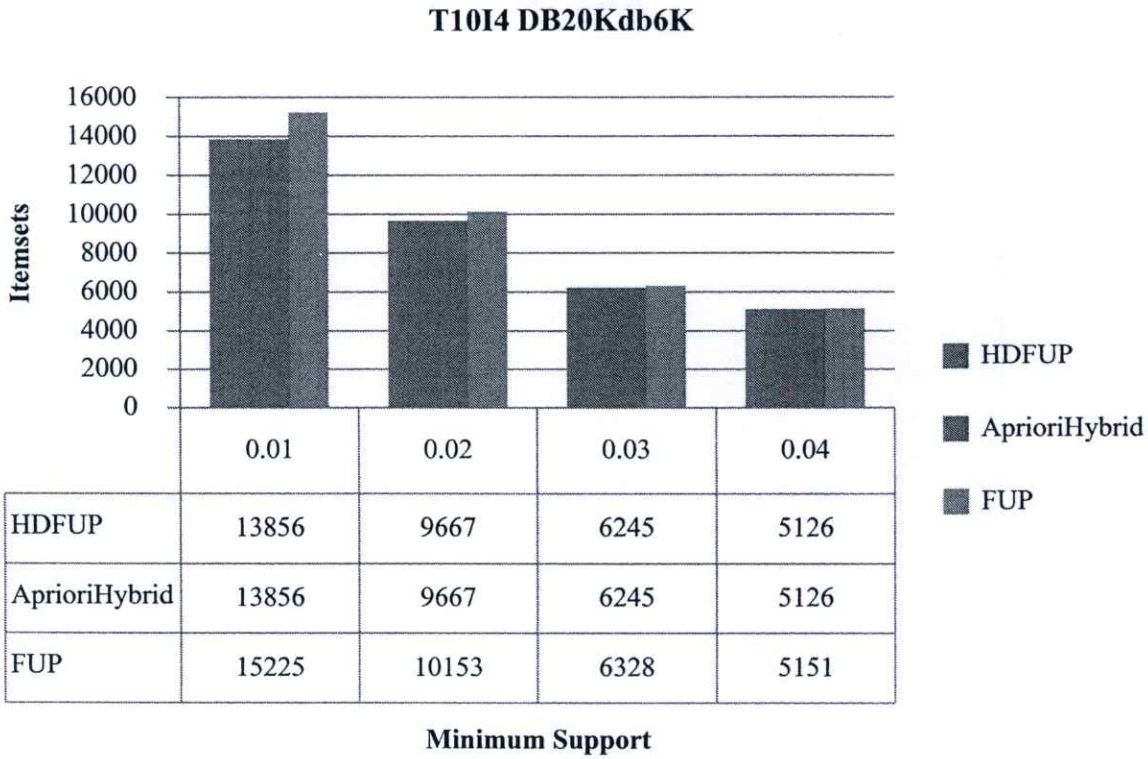
	0.01	0.02	0.03	0.04
HDFUP	14132	10032	6354	4934
AprioriHybrid	14132	10032	6354	4934
FUP	15576	10585	6441	4950

รูปที่ 4.1 ผลการทดลองการ join สร้าง 2-Itemsets ของข้อมูลชุดที่ 1 T10I4DB20db10K



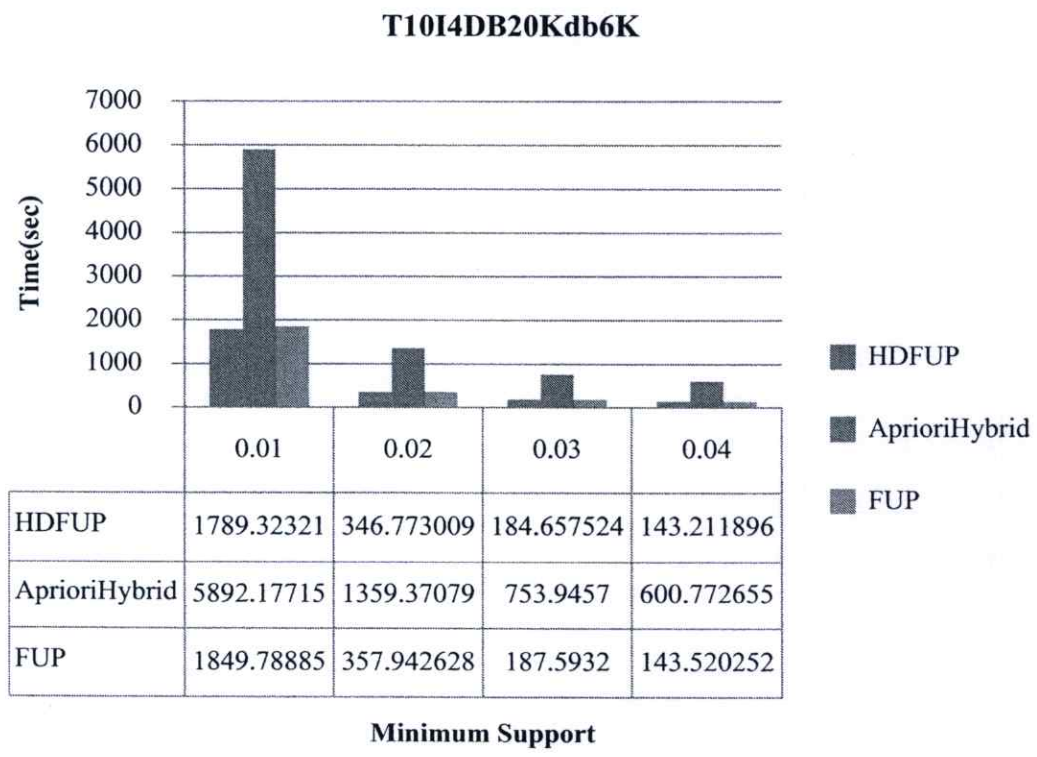
	0.01	0.02	0.03	0.04
HDFUP	2668.33789	570.507002	302.344131	227.3984
AprioriHybrid	6693.39296	1615.94444	883.429552	685.855612
FUP	2757.46206	590.024236	305.510937	231.354408

รูปที่ 4.2 ผลการทดลองแสดงเวลาของข้อมูลชุดที่ 1 T10I4DB20db10K



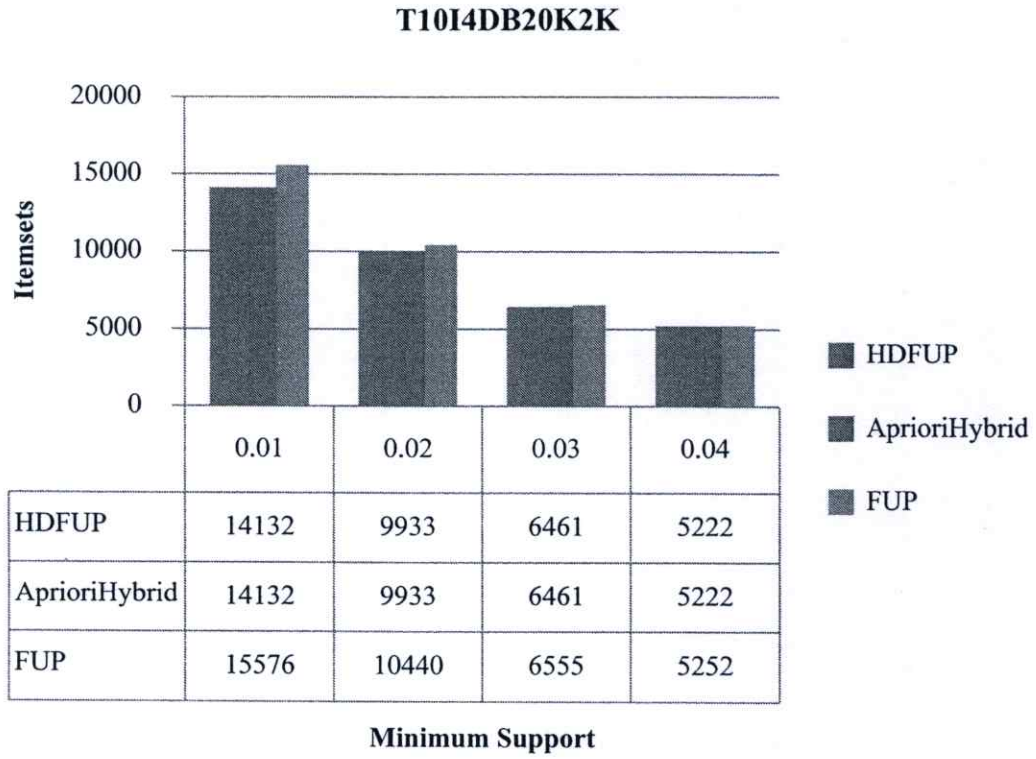
	0.01	0.02	0.03	0.04
HDFUP	13856	9667	6245	5126
AprioriHybrid	13856	9667	6245	5126
FUP	15225	10153	6328	5151

รูปที่ 4.3 ผลการทดลองการ join สร้าง 2-Itemsets ของข้อมูลชุดที่ 1 T10I4DB20db6K



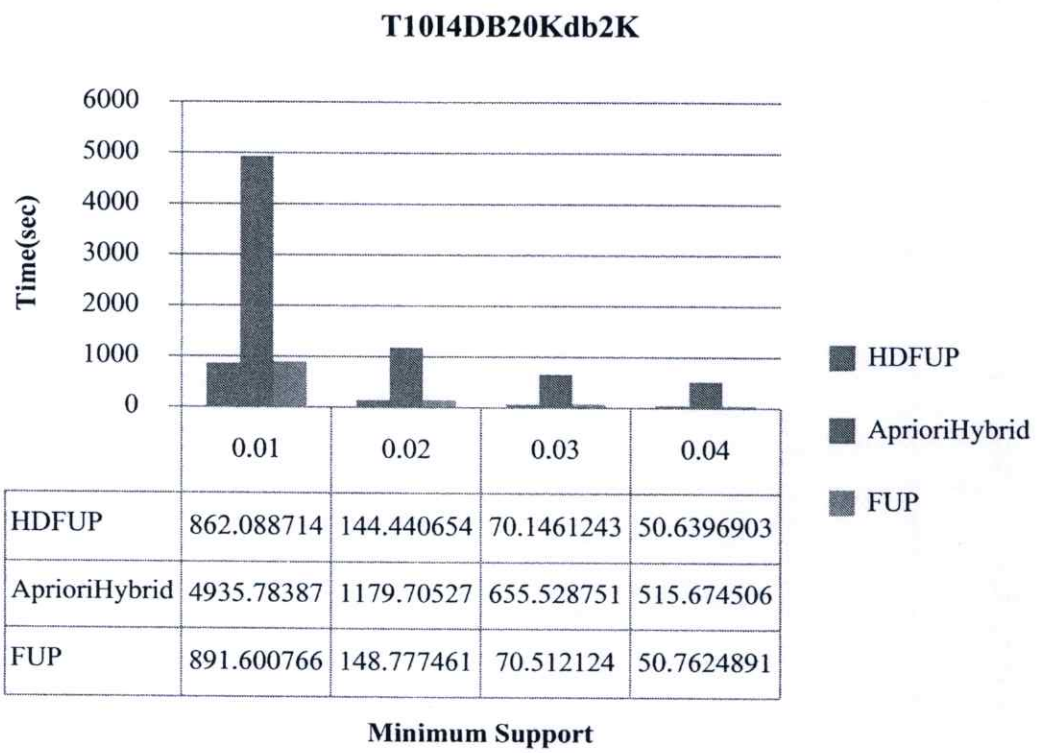
	0.01	0.02	0.03	0.04
HDFUP	1789.32321	346.773009	184.657524	143.211896
AprioriHybrid	5892.17715	1359.37079	753.9457	600.772655
FUP	1849.78885	357.942628	187.5932	143.520252

รูปที่ 4.4 ผลการทดลองแสดงเวลาของข้อมูลชุดที่ 1 T10I4DB20db6K



	0.01	0.02	0.03	0.04
HDFUP	14132	9933	6461	5222
AprioriHybrid	14132	9933	6461	5222
FUP	15576	10440	6555	5252

รูปที่ 4.5 ผลการทดลองการ join สร้าง 2-Itemsets ของข้อมูลชุดที่ 1 T10I4DB20db2K



	0.01	0.02	0.03	0.04
HDFUP	862.088714	144.440654	70.1461243	50.6396903
AprioriHybrid	4935.78387	1179.70527	655.528751	515.674506
FUP	891.600766	148.777461	70.512124	50.7624891

รูปที่ 4.6 ผลการทดลองแสดงเวลาของข้อมูลชุดที่ 1 T10I4DB20db10K

ตารางที่ 4.7 ผลการทดลองชุดที่ 2 T10I4DB20Kdb10K

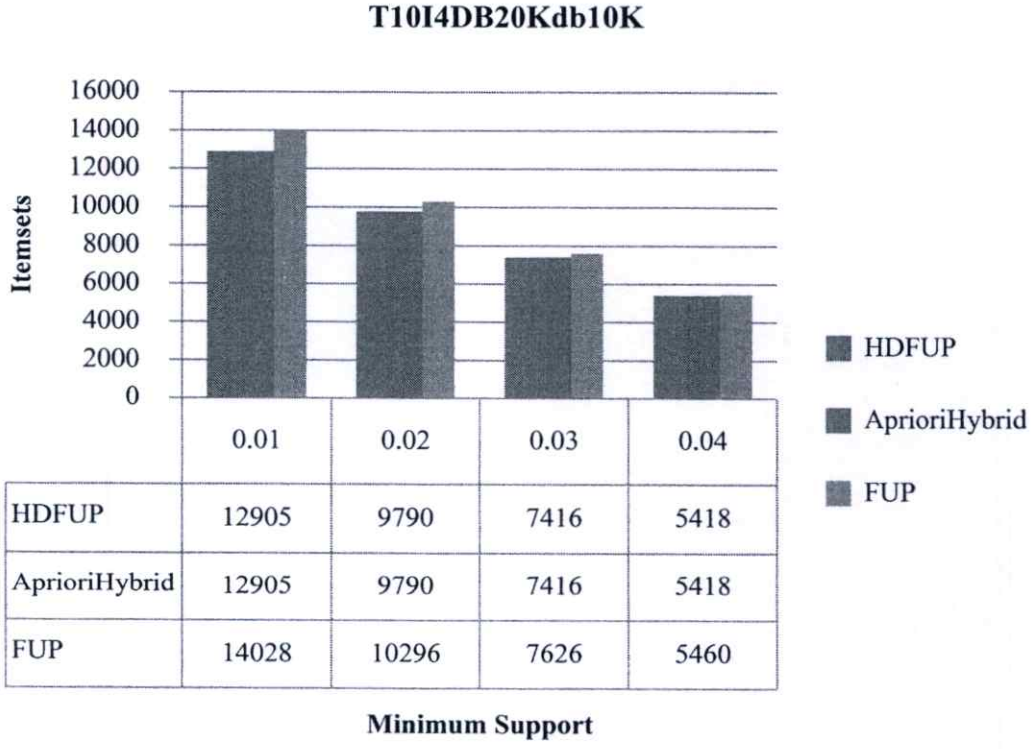
Min Support	Algorithm	Join Itemsets	Candidate Itemsets	Large Itemsets	Time(sec)
0.01	HDFUP	36,315	510	7,000	2591.069804
	AprioriHybrid	36,315	36,485	7,000	6609.560009
	FUP	37,438	510	7,000	2697.166737
0.02	HDFUP	11,585	101	1,219	570.5554022
	AprioriHybrid	11,585	11,755	1,219	1607.415225
	FUP	12,091	101	1,219	588.4018334
0.03	HDFUP	7,689	31	466	346.9010102
	AprioriHybrid	7,689	7,859	466	1010.164174
	FUP	7,899	31	466	354.2922223
0.04	HDFUP	5,476	15	217	247.8540357
	AprioriHybrid	5,476	5,646	217	728.8740807
	FUP	5,518	15	217	251.9060449

ตารางที่ 4.8ผลการทดลองชุดที่ 2 T10I4DB 20Kdb6K

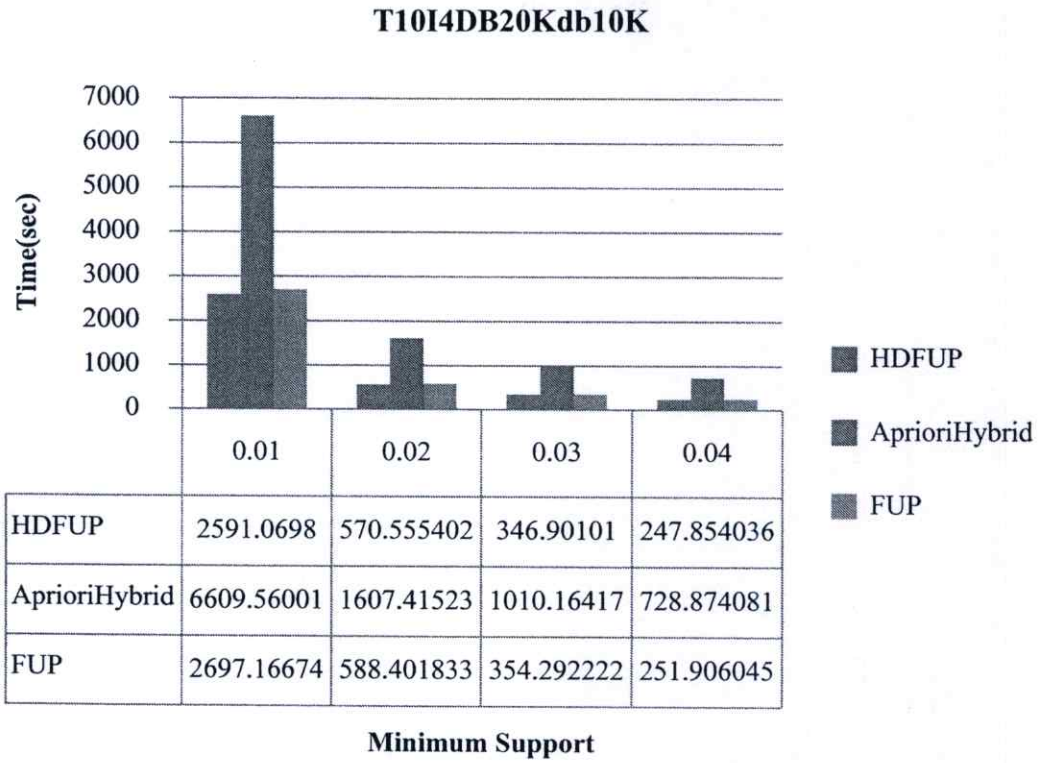
Min Support	Algorithm	Join Itemsets	Candidate Itemsets	Large Itemsets	Time(sec)
0.01	HDFUP	36,314	426	7,011	1716.814216
	AprioriHybrid	36,314	36,484	7,011	5755.460121
	FUP	37,437	426	7,011	1761.74988
0.02	HDFUP	11,597	128	1,244	361.8610356
	AprioriHybrid	11,597	11,767	1,244	1398.573657
	FUP	12,103	128	1,244	371.9514532
0.03	HDFUP	7,819	38	483	217.4175818
	AprioriHybrid	7,819	7,989	483	891.4299663
	FUP	8,044	38	483	222.1435909
0.04	HDFUP	5,668	21	217	156.000274
	AprioriHybrid	5,668	5,838	217	650.8867443
	FUP	5,724	21	217	157.022676

ตารางที่ 4.9 ผลการทดลองชุดที่ 2 T10I4DB20Kdb2K

Min Support	Algorithm	Join Itemsets	Candidate Itemsets	Large Itemsets	Time(sec)
0.01	HDFUP	36,451	675	7,010	860.6931117
	AprioriHybrid	36,451	36,621	7,010	4889.702191
	FUP	37,574	675	7,010	882.9771503
0.02	HDFUP	11,599	172	1,232	147.498259
	Apriori Hybrid	11,599	11,769	1,232	1181.858076
	FUP	12,106	172	1,232	151.4606661
0.03	HDFUP	8,029	64	476	82.6333451
	AprioriHybrid	8,029	8,199	476	770.501354
	FUP	8,285	64	476	84.7009488
0.04	HDFUP	5,573	25	219	53.9448947
	AprioriHybrid	5,573	5,743	219	542.8029534
	FUP	5,622	25	219	54.1632951

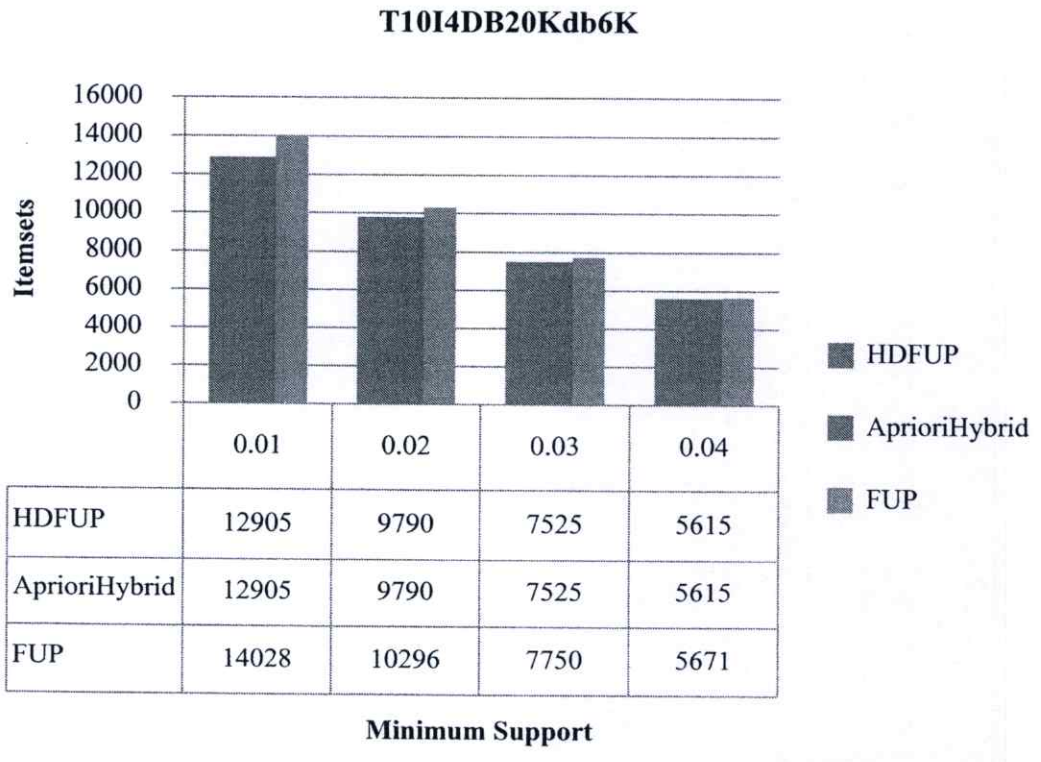


รูปที่ 4.7 ผลการทดลองแสดงผลการ join สร้าง 2-Itemsets ของข้อมูลชุดที่ 2 T10I4DB20db10K



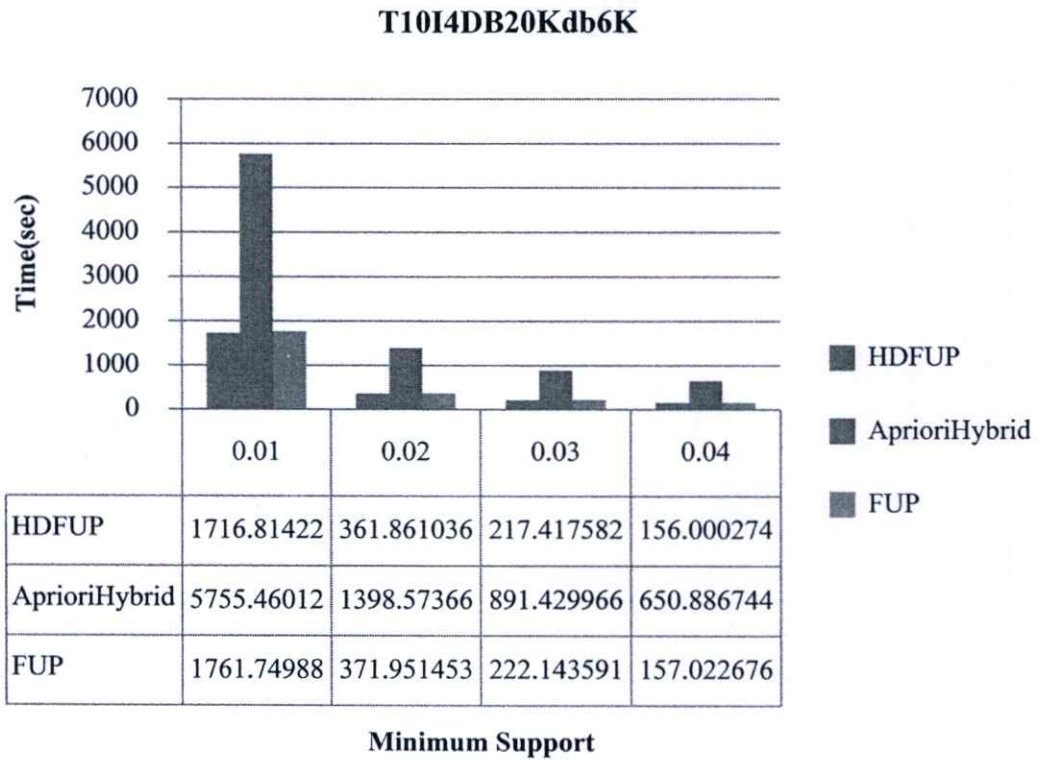
	0.01	0.02	0.03	0.04
HDFUP	2591.0698	570.555402	346.90101	247.854036
AprioriHybrid	6609.56001	1607.41523	1010.16417	728.874081
FUP	2697.16674	588.401833	354.292222	251.906045

รูปที่ 4.8 ผลการทดลองแสดงเวลาของข้อมูลชุดที่2 T10I4DB20db10K



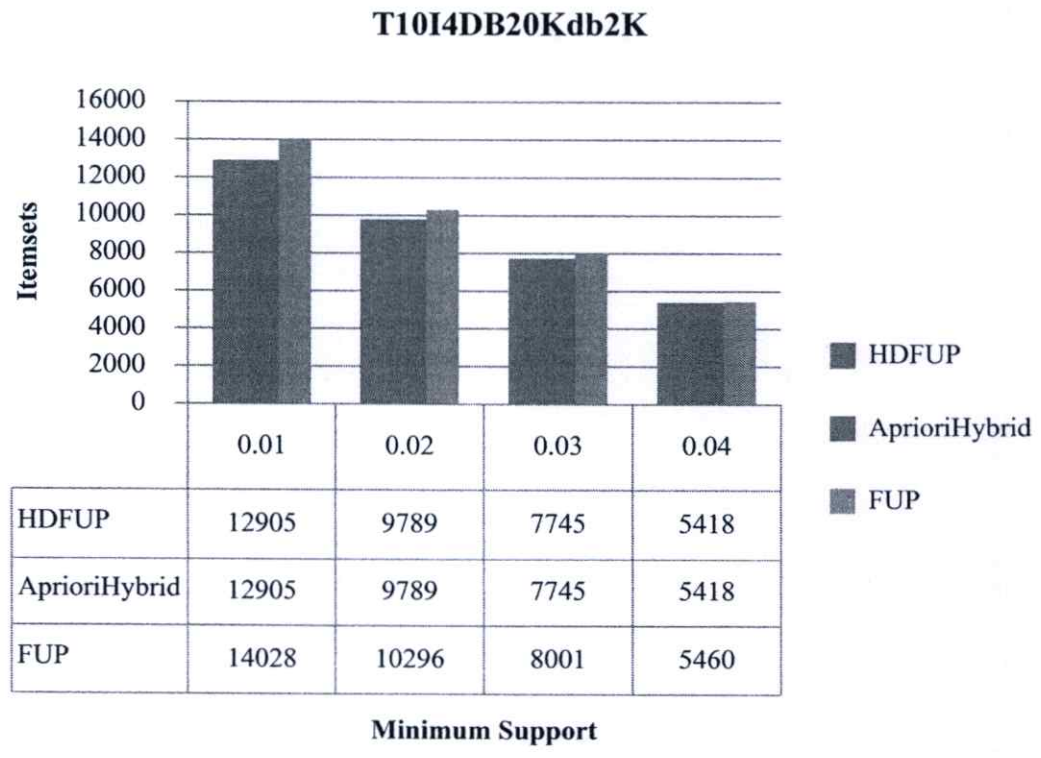
	0.01	0.02	0.03	0.04
HDFUP	12905	9790	7525	5615
AprioriHybrid	12905	9790	7525	5615
FUP	14028	10296	7750	5671

รูปที่ 4.9 ผลการทดลองแสดงผลการ join สร้าง 2-Itemsets ของข้อมูลชุดที่ 2 T10I4DB20db6K



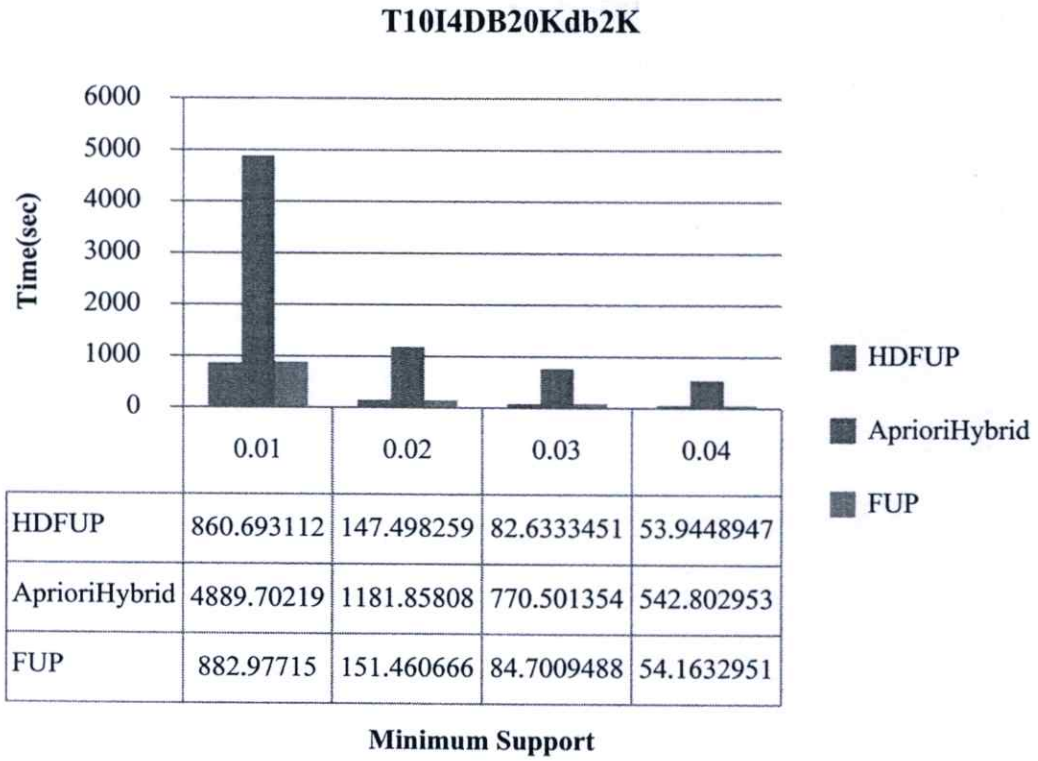
	0.01	0.02	0.03	0.04
HDFUP	1716.81422	361.861036	217.417582	156.000274
AprioriHybrid	5755.46012	1398.57366	891.429966	650.886744
FUP	1761.74988	371.951453	222.143591	157.022676

รูปที่ 4.10 ผลการทดลองแสดงเวลาของข้อมูลชุดที่2 T10I4DB20db6K



	0.01	0.02	0.03	0.04
HDFUP	12905	9789	7745	5418
AprioriHybrid	12905	9789	7745	5418
FUP	14028	10296	8001	5460

รูปที่ 4.11 ผลการทดลองแสดงผลการ join สร้าง 2-Itemsets ของข้อมูลชุดที่ 2 T10I4DB20db2K



	0.01	0.02	0.03	0.04
HDFUP	860.693112	147.498259	82.6333451	53.9448947
AprioriHybrid	4889.70219	1181.85808	770.501354	542.802953
FUP	882.97715	151.460666	84.7009488	54.1632951

รูปที่ 4.12 ผลการทดลองแสดงเวลาของข้อมูลชุดที่ 2 T10I4DB20db2K

ผลการทดลองชุดที่ 3 กำหนดค่าสนับสนุนขั้นต่ำที่ใช้ในการทดลองทั้งหมด 3 ค่าคือ 0.02, 0.03, 0.04

ตารางที่ 4.10 ผลการทดลองชุดที่ 3 T10I4DB20Kdb10K

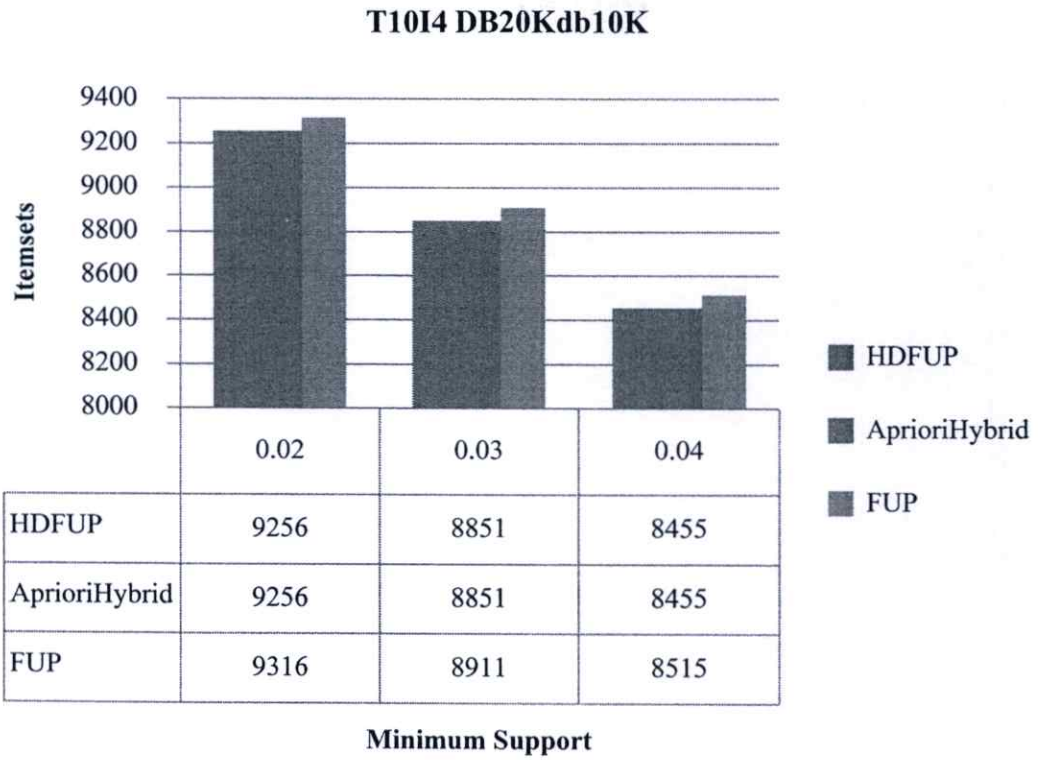
Min Support	Algorithm	Join Itemsets	Candidate Itemsets	Large Itemsets	Time(sec)
0.02	HDFUP	83048	1666	13692	8872.3904716
	Apriori Hybrid	83048	83188	13692	17077.740791
	FUP	83108	1666	13692	9284.0890194
0.03	HDFUP	34008	245	3672	2344.4720962
	Apriori Hybrid	34008	34148	3672	6253.418675
	FUP	34068	245	3672	2377.8960079
0.04	HDFUP	19352	111	1739	1179.3084525
	Apriori Hybrid	19352	19492	1739	3324.857171
	FUP	19412	111	1739	1181.2955662

ตารางที่ 4.11 ผลการทดลองชุดที่ 3 T10I4DB20Kdb6K

Min Support	Algorithm	Join	Candidate Itemsets	Large Itemsets	Time(sec)
0.02	HDFUP	83916	2788	14153	6628.6996381
	Apriori Hybrid	83916	84056	14153	14636.899943
	FUP	83976	2788	14153	7089.3375977
0.03	HDFUP	34168	434	3725	1482.9010061
	Apriori Hybrid	34168	34308	3725	5199.5631366
	FUP	34228	434	3725	1511.9392914
0.04	HDFUP	19732	142	1760	714.1224542
	Apriori Hybrid	19732	19872	1760	2798.3833167
	FUP	19792	142	1760	721.0084668

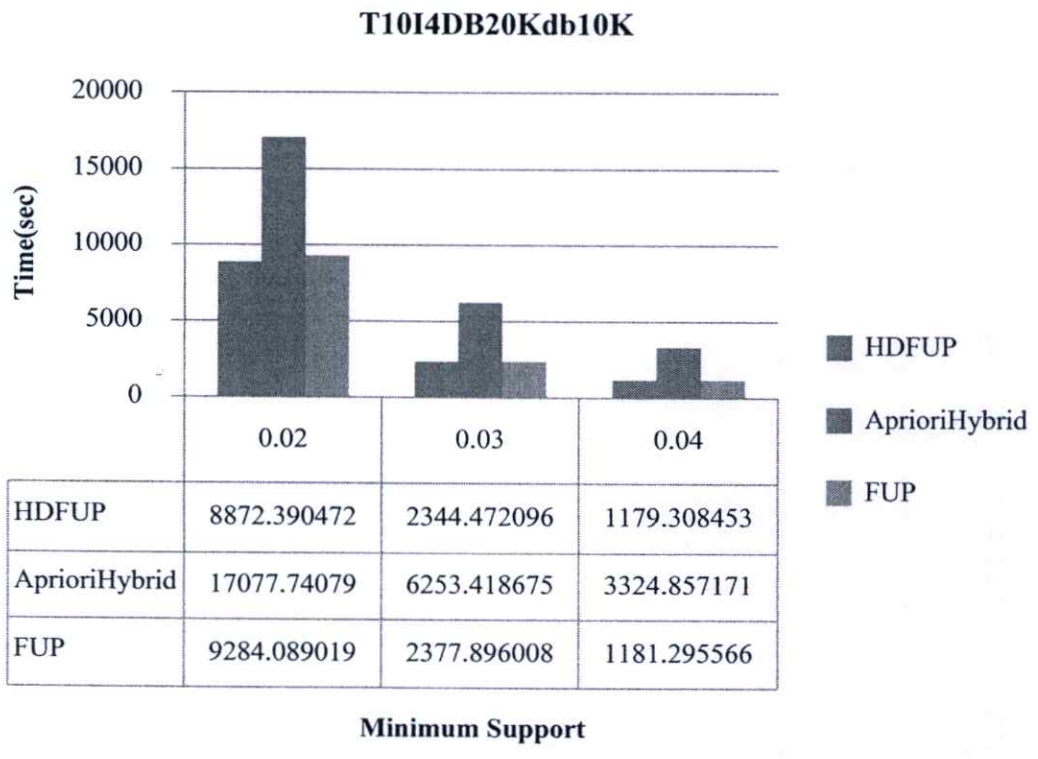
ตารางที่ 4.12 ผลการทดลองชุดที่ 3 T10I4DB20Kdb2K

Min Support	Algorithm	Join	Candidate Itemsets	Large Itemsets	Time(sec)
0.02	HDFUP	84099	6165	14264	5257.4027062
	Apriori Hybrid	84099	84239	14264	10445.6104541
	FUP	84159	6165	14264	6135.2409158
0.03	HDFUP	34000	859	3734	780.7126543
	Apriori Hybrid	34000	34140	3734	4609.6866588
	FUP	34060	859	3734	808.3692361
0.04	HDFUP	19518	302	1753	300.260128
	Apriori Hybrid	19518	19658	1753	2336.9057055
	FUP	19578	302	1753	304.4989354



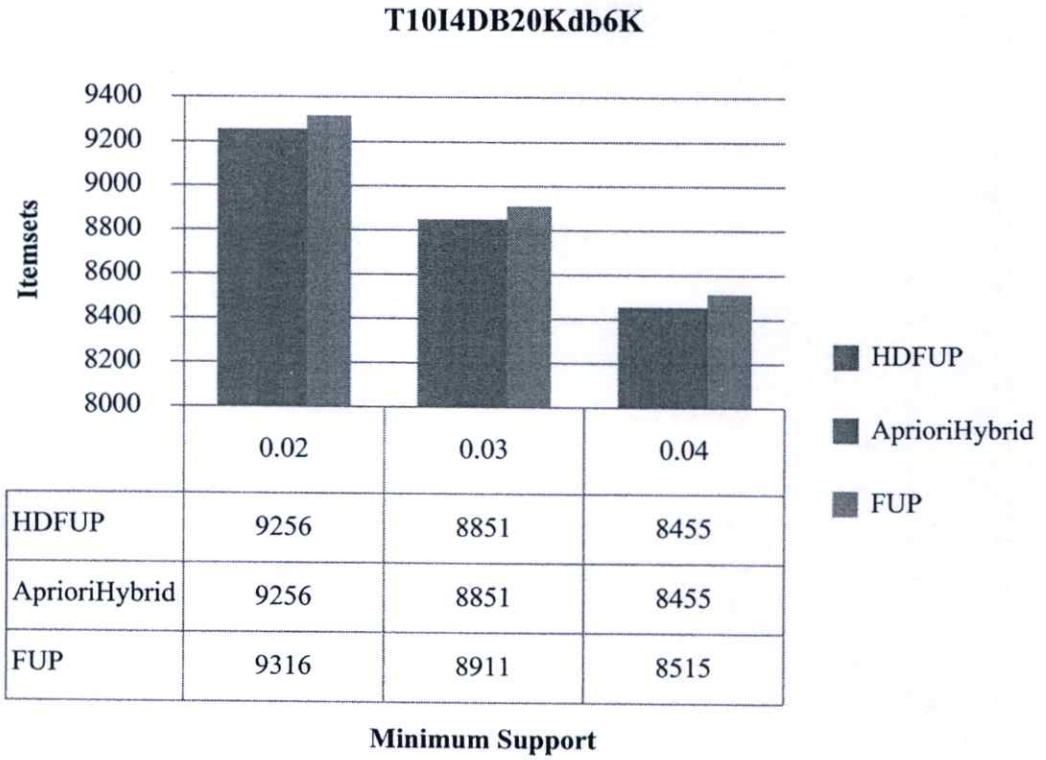
	0.02	0.03	0.04
HDFUP	9256	8851	8455
AprioriHybrid	9256	8851	8455
FUP	9316	8911	8515

รูปที่ 4.13 ผลการทดลองแสดงผลการ join สร้าง 2-Itemsets ของข้อมูลชุดที่ 3 T10I4DB20db10K



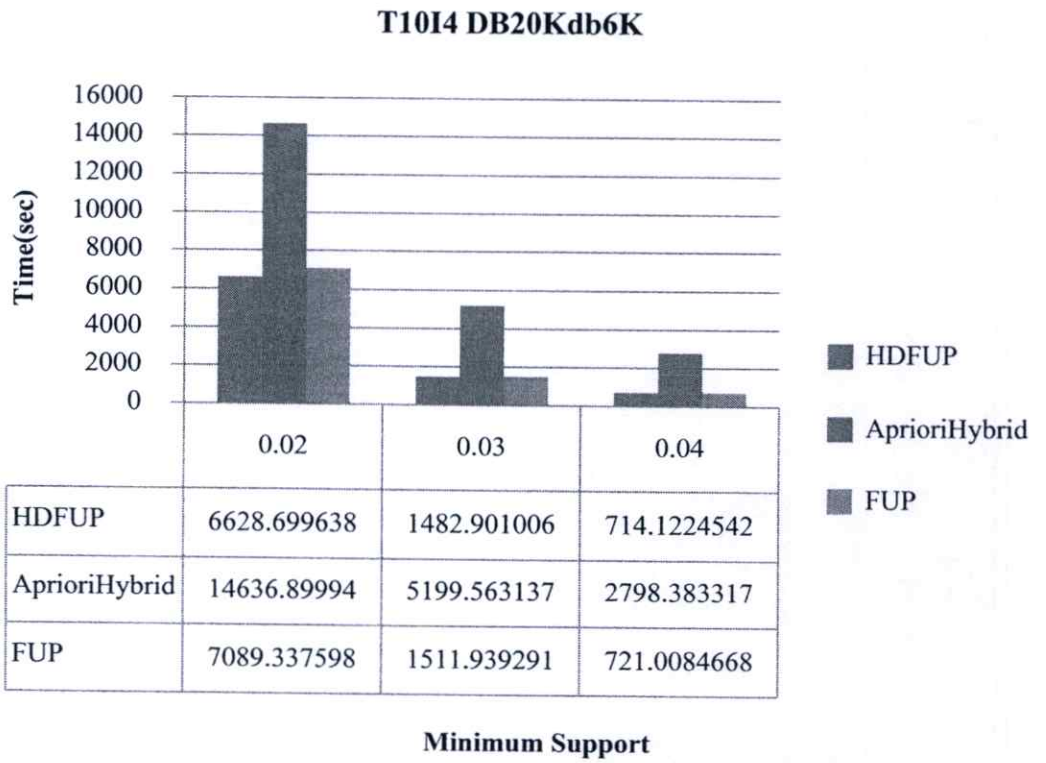
	0.02	0.03	0.04
HDFUP	8872.390472	2344.472096	1179.308453
AprioriHybrid	17077.74079	6253.418675	3324.857171
FUP	9284.089019	2377.896008	1181.295566

รูปที่ 4.14 ผลการทดลองแสดงเวลาของข้อมูลชุดที่3 T10I4DB20db10K



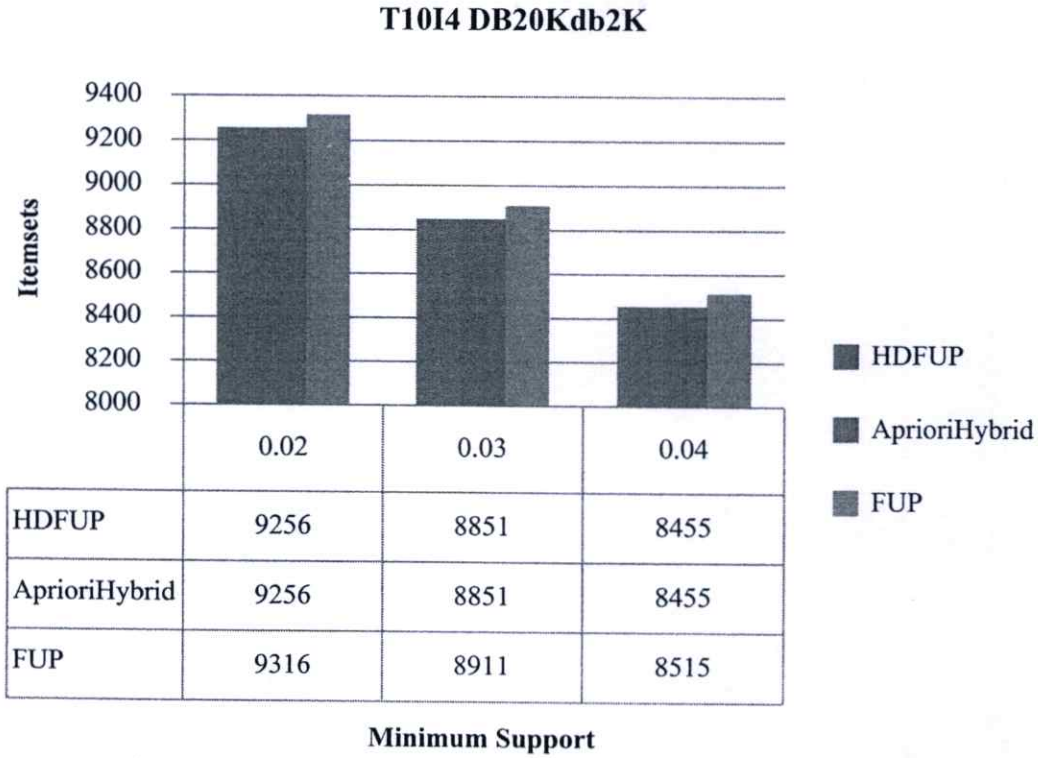
	0.02	0.03	0.04
HDFUP	9256	8851	8455
AprioriHybrid	9256	8851	8455
FUP	9316	8911	8515

รูปที่ 4.15 ผลการทดลองแสดงผลการ join สร้าง 2-Itemsets ของข้อมูลชุดที่ 3 T10I4DB20db6K



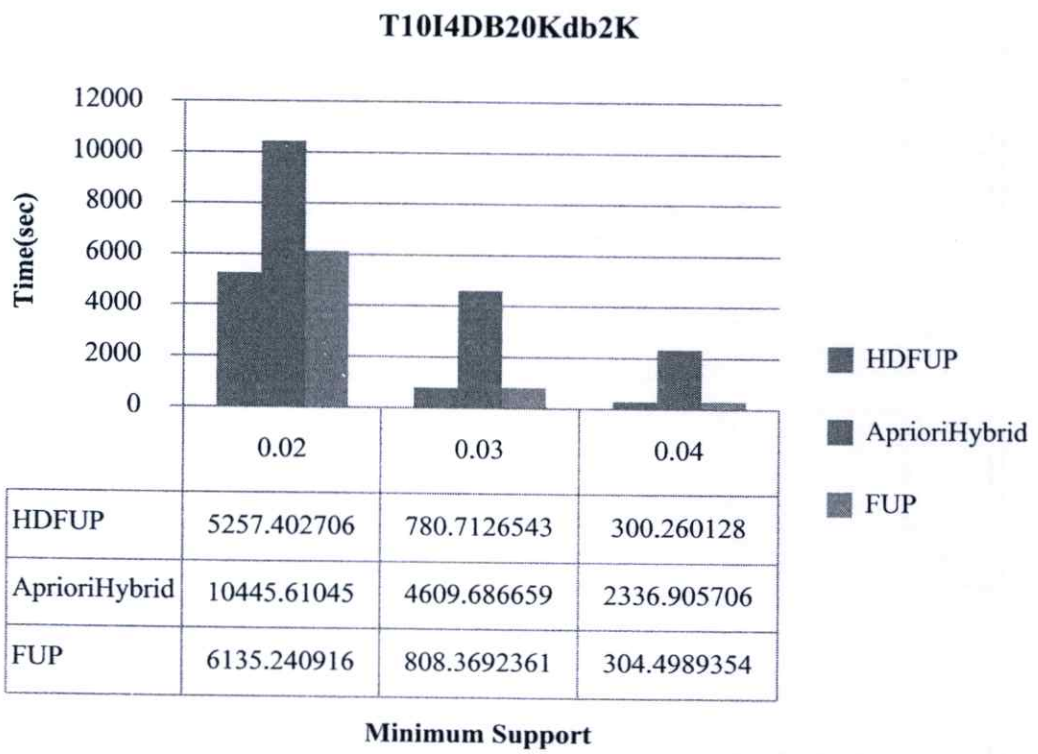
	0.02	0.03	0.04
HDFUP	6628.699638	1482.901006	714.1224542
AprioriHybrid	14636.89994	5199.563137	2798.383317
FUP	7089.337598	1511.939291	721.0084668

รูปที่ 4.16 ผลการทดลองแสดงเวลาของข้อมูลชุดที่3 T10I4DB20db6K



	0.02	0.03	0.04
HDFUP	9256	8851	8455
AprioriHybrid	9256	8851	8455
FUP	9316	8911	8515

รูปที่ 4.17 ผลการทดลองแสดงผลการ join สร้าง 2-Itemsets ของข้อมูลชุดที่ 3 T10I4DB20db2K



	0.02	0.03	0.04
HDFUP	5257.402706	780.7126543	300.260128
AprioriHybrid	10445.61045	4609.686659	2336.905706
FUP	6135.240916	808.3692361	304.4989354

รูปที่ 4.18 ผลการทดลองแสดงเวลาของข้อมูลชุดที่3 T10I4DB20db2K

ตารางที่ 4.13 ตารางเวลา 1-3 Itemsets ของชุดข้อมูลที่ 1 เพิ่มข้อมูลขนาด 10,000 ทรานแซคชั่น

Increment Database	Algorithm	Minimum Support	Time(sec)		
			1-Itemsets	2-Itemsets	3-Itemsets
10,000	HDFUP	0.01	7.7156137	665.4815688	1522.531474
		0.02	7.2680128	439.8131707	105.1425864
		0.03	6.5052114	278.4760892	14.9136262
		0.04	6.286811	217.2583817	3.8532068
	AprioriHybrid	0.01	11.8092208	1742.421862	3919.341288
		0.02	11.7312206	1257.050208	298.7873248
		0.03	11.7624206	821.1074422	43.6488767
		0.04	11.8404208	663.8003727	10.2148181
	FUP	0.01	7.7376136	725.354474	1541.688308
		0.02	7.2540127	458.4224052	105.9865861
		0.03	6.4740114	281.6428946	14.9760263
		0.04	6.271211	221.2183898	3.8648068

ตารางที่ 4.14 ตารางเวลา 1-3 Itemsets ของชุดข้อมูลที่ 1 เพิ่มข้อมูลขนาด 6,000 ทรานแซคชั่น

Increment Database	Algorithm	Minimum Support	Time(sec)		
			1-temsets	2-Itemsets	3-Itemsets
6,000	HDFUP	0.01	4.6488082	431.5903581	1038.556224
		0.02	4.4460079	264.5920648	67.1113179
		0.03	3.9312069	167.7002945	11.5440203
		0.04	3.8568074	136.9058841	2.4492043
	AprioriHybrid	0.01	10.2024179	431.5903581	3508.867363
		0.02	10.3740183	1053.844251	260.0524567
		0.03	10.1712179	697.598426	39.9516702
		0.04	10.3116181	581.6938217	8.7672154
	FUP	0.01	4.6644082	468.7028233	1050.817846
		0.02	4.4148078	275.1688833	67.7197199
		0.03	3.9312069	170.6207017	11.55926203
		0.04	3.8220067	137.2958411	2.4024043

ตารางที่ 4.15 ตารางเวลา 1-3 Itemsets ของชุดข้อมูลที่ 1 เพิ่มข้อมูลขนาด 2,000 ทรานแซคชั่น

Increment Database	Algorithm	Minimum Support	Time(sec)		
			1-temsets	2-Itemsets	3-Itemsets
2,000	HDFUP	0.01	1.7472031	218.2287833	509.7776954
		0.02	1.6224028	109.9333931	28.8132506
		0.03	1.4196025	63.4693126	4.6424083
		0.04	1.3260024	48.0812857	1.2324022
	AprioriHybrid	0.01	8.564415	1268.937429	2955.877592
		0.02	8.6268152	915.5344081	221.4891891
		0.03	8.5956151	610.1794717	31.4808553
		0.04	8.7516153	499.4036772	7.5196846
	FUP	0.01	1.7472031	236.1688148	516.0957065
		0.02	1.6224028	114.2234006	28.9068507
		0.03	1.4196025	63.8821122	4.6956083
		0.04	1.3104023	48.2196846	1.2324022

ตารางที่ 4.16 ตารางเวลา 1-3 Itemsets ของชุดข้อมูลที่ 2 เพิ่มข้อมูลขนาด 10,000 ทรานแซคชั่น

Increment Database	Algorithm	Minimum Support	Time(sec)		
			1-Itemsets	2-Itemsets	3-Itemsets
10,000	HDFUP	0.01	7.1604125	614.4850792	1525.603532
		0.02	6.9732123	431.3563577	110.8381946
		0.03	6.5668121	320.1009625	17.5344308
		0.04	6.0216106	236.9652165	4.6020081
	AprioriHybrid	0.01	10.7016188	1591.389995	4013.512649
		0.02	10.826419	1228.365759	311.2049466
		0.03	10.7484189	944.1604583	47.9232842
		0.04	10.7328189	706.800042	11.0916194
	FUP	0.01	7.1292126	659.9747592	1577.942772
		0.02	6.848412	449.1091888	110.994195
		0.03	6.4428113	327.5849754	17.5812309
		0.04	6.0060105	241.0640258	4.5708081

ตารางที่ 4.17 ตารางเวลา 1-3 Itemsets ของชุดข้อมูลที่ 2 เพิ่มข้อมูลขนาด 6,000 ทรานแซคชั่น

Increment Database	Algorithm	Minimum Support	Time(sec)		
			1-Itemsets	2-Itemsets	3-Itemsets
6,000	HDFUP	0.01	4.3368076	409.7035196	1030.912211
		0.02	4.1836.74	271.612077	69.4825221
		0.03	3.9000068	199.6959507	11.8560208
		0.04	3.6816065	149.5886627	2.7300048
	AprioriHybrid	0.01	9.3288164	409.7035196	1030.912211
		0.02	9.4068165	1066.090273	270.2080746
		0.03	9.2820163	830.6366596	44.8188787
		0.04	9.4372171	632.5263116	8.9232156
	FUP	0.01	4.3368077	439.4059718	1042.815032
		0.02	4.0872072	282.1888956	69.3421218
		0.03	3.8844068	204.4843599	11.8248208
		0.04	3.7212066	150.6026646	2.6988048

ตารางที่ 4.18 ตารางเวลา 1-3 Itemsets ของชุดข้อมูลที่ 2 เพิ่มข้อมูลขนาด 2,000 ทรานแซคชั่น

Increment Database	Algorithm	Minimum Support	Time(sec)		
			1-Itemsets	2-Itemsets	3-Itemsets
2,000	HDFUP	0.01	1.5216028	206.3727624	518.0301099
		0.02	1.4820026	109.5901925	31.1844548
		0.03	1.5132026	75.0517318	5.3352094
		0.04	1.3260023	51.7140908	0.9048016
	AprioriHybrid	0.01	8.0140142	1170.894059	2972.320021
		0.02	7.8468137	901.5879835	231.1456059
		0.03	8.0188149	720.4560654	36.5508641
		0.04	7.940414	526.7661252	8.0964142
	FUP	0.01	1.4820026	220.709187	524.1921207
		0.02	1.4820026	113.6305996	31.1532548
		0.03	1.5216027	77.0953354	5.3508094
		0.04	1.3260023	51.9324912	0.9048016

ตารางที่ 4.19 ตารางเวลา 1-3 Itemsets ของชุดข้อมูลที่ 3 เพิ่มข้อมูลขนาด 10,000 ทรานแซคชั่น

Increment Database	Algorithm	Minimum Support	Time(sec)		
			1-Itemsets	2-Itemsets	3-Itemsets
10,000	HDFUP	0.02	6.718384	503.8728199	6343.831846
		0.03	6.8223903	469.187836	1797.744825
		0.04	6.5193729	422.8521858	743.2045088
	AprioriHybrid	0.02	9.9155672	1267.113475	12102.1702
		0.03	10.577605	1253.989724	4807.967
		0.04	10.5416029	1175.530237	2121.042317
	FUP	0.02	6.7043834	508.8621052	6468.798994
		0.03	6.7433857	470.3319014	1823.961325
		0.04	6.719384	425.602343	742.0914452

ตารางที่ 4.20 ตารางเวลา 1-3 Itemsets ของชุดข้อมูลที่ 3 เพิ่มข้อมูลขนาด 6,000 ทรานแซคชั่น

Increment Database	Algorithm	Minimum Support	Time(sec)		
			1-Itemsets	2-Itemsets	3-Itemsets
6,000	HDFUP	0.02	3.944007	342.2385928	4541.80078
		0.03	3.9376074	293.2657156	1137.852399
		0.04	3.8376068	261.362859	444.1639801
	AprioriHybrid	0.02	8.6876154	1089.912942	10282.40327
		0.03	8.7204153	1024.614201	4022.799868
		0.04	8.564415	976.2037152	1800.121163
	FUP	0.02	3.960807	343.4365457	4728.284308
		0.03	3.9312069	294.746353	1161.797241
		0.04	3.8376067	262.3612608	450.082791

ตารางที่ 4.21 ตารางเวลา 1-3 Itemsets ของชุดข้อมูลที่ 3 เพิ่มข้อมูลขนาด 2,000 ทรานแซคชั่น

Increment Database	Algorithm	Minimum Support	Time(sec)		
			1-Itemsets	2-Itemsets	3-Itemsets
2,000	HDFUP	0.02	1.5300875	188.4737801	3236.572121
		0.03	1.39808	151.1766468	583.9994029
		0.04	1.3416024	116.0798039	179.047915
	AprioriHybrid	0.02	7.7064408	941.6118571	9037.032889
		0.03	7.5904341	924.3808715	3535.003191
		0.04	7.2384127	824.5146487	1492.189021
	FUP	0.02	1.5310876	188.8498016	3812.218046
		0.03	1.3660781	151.0406391	606.7907064
		0.04	1.3416024	116.5946048	182.8343216

ในการค้นหากฎความสัมพันธ์แบบมิติผสมสำหรับการค้นหากฎความสัมพันธ์เมื่อมีการเพิ่มข้อมูลด้วยอัลกอริทึม HDFUP จากการทดลองทั้ง 3 ชุด แสดงให้เห็นความแตกต่างของการเชื่อมความสัมพันธ์สร้าง 2-Itemsets ของอัลกอริทึม HDFUP ซึ่งอัลกอริทึมดังกล่าวมีจำนวนไอเทมเซตน้อยกว่าเมื่อเปรียบเทียบกับอัลกอริทึม FUP เนื่องจากการเชื่อมความสัมพันธ์ของอัลกอริทึม FUP เป็นการเชื่อมความสัมพันธ์ในทุกไอเทมเซต รวมถึงการเชื่อมความสัมพันธ์ภายในแอททริบิวต์ซึ่งทำให้ได้ไอเทมเซตที่ไม่ปรากฏในทรานแซคชั่นแบบหลายมิติอย่างแน่นอน แต่อัลกอริทึม HDFUP หลีกเลี่ยงการเชื่อมความสัมพันธ์ภายในแอททริบิวต์ร่วม และอัลกอริทึมเอพริออรีแบบการค้นหากฎความสัมพันธ์แบบมิติผสม ภายหลังการเชื่อมความสัมพันธ์ทำให้ได้ไอเทมเซตเช่นเดียวกับอัลกอริทึม HDFUP เนื่องจากใช้วิธีในการค้นหากฎความสัมพันธ์เช่นเดียวกับอัลกอริทึม HDFUP และประสิทธิภาพด้านเวลาของอัลกอริทึม HDFUP ใช้เวลาน้อยกว่าอัลกอริทึม FUP เนื่องจากอัลกอริทึม HDFUP มีการเชื่อมไอเทมเซตในขั้น 2-Itemsets มีจำนวนน้อยกว่าอัลกอริทึม FUP มีผลให้ไอเทมเซตใน Candidate Itemset มีจำนวนน้อยกว่าอัลกอริทึม FUP ทำให้การเข้าไปค้นหาค่าสนับสนุนในฐานข้อมูลใหม่ใช้เวลาน้อยกว่าอัลกอริทึม FUP ด้วย ผลการทดลองของชุดการทดลองที่ 1 และ 2 ในการค้นหาไอเทมเซตขั้น 3-Itemsets ขึ้นไปใช้เวลาใกล้เคียงกัน ในชุดการทดลองที่ 3 อัลกอริทึม HDFUP สามารถช่วยลดเวลาในการสร้างไอเทมเซตตั้งแต่ขั้น 3-Itemsets เนื่องผลการเชื่อมความสัมพันธ์ของอัลกอริทึม HDFUP หลีกเลี่ยงการเชื่อมสัมพันธ์ที่ไม่เกิดขึ้นจริงในฐานข้อมูลแบบทรานแซคชั่นหลายมิติ ทำให้ช่วยลดการตรวจสอบในการเช็คความเป็นซับเซตของไอเทมเซตที่สร้างขึ้นมา และอัลกอริทึมเอพริออรีแบบค้นหากฎความสัมพันธ์แบบมิติผสมใช้เวลา

ในการประมวลผลมากกว่าอัลกอริทึม HDFUP และอัลกอริทึม FUP เนื่องจากอัลกอริทึมเอพริออรีแบบค้นหากฎความสัมพันธ์แบบมิติผสมมีการนำไอเทมเซตทุกตัวที่เชื่อมได้ทั้งหมดไปเป็น Candidate Itemsets แต่อัลกอริทึม HDFUP และอัลกอริทึม FUP มีการคัดไอเทมเซตที่มีอยู่ใน L_k ออก ทำให้จำนวน Candidate Itemsets ของอัลกอริทึมเอพริออรีแบบค้นหากฎความสัมพันธ์แบบมิติผสมมีจำนวนมากกว่าอัลกอริทึม HDFUP และอัลกอริทึม FUP ทำให้การค้นหาในฐานข้อมูลมีจำนวนครั้งมากกว่าและมีผลทำให้ใช้เวลาในการค้นหามากขึ้นตามไปด้วย

จากผลการทดลอง Large Itemsets ของอัลกอริทึม HDFUP มีค่าเหมือนกับอัลกอริทึมแอพริออรีค้นหากฎความสัมพันธ์แบบมิติผสม และอัลกอริทึม FUP ทำให้ทราบความถูกต้องของผลลัพธ์ ในส่วนของ Candidate Itemsets สำหรับนำไปค้นหาในฐานข้อมูลเดิม อัลกอริทึม HDFUP และอัลกอริทึม FUP มีไอเทมเซตของ Candidate Itemsets เหมือนกัน แต่อัลกอริทึมแอพริออรีค้นหากฎความสัมพันธ์แบบมิติผสม มีไอเทมเซตของ Candidate Itemsets มากกว่าอัลกอริทึม HDFUP และอัลกอริทึม FUP เพราะอัลกอริทึมแอพริออรีค้นหากฎความสัมพันธ์แบบมิติผสม นำไอเทมเซตที่ได้จากการเชื่อมความสัมพันธ์ทั้งหมดเป็น Candidate Itemsets เพื่อใช้ค้นหาในฐานข้อมูลทั้งหมด และการกำหนดค่าสนับสนุนขั้นต่ำมีค่ามากอาจมีผลให้ไอเทมเซตของแอททริบิวต์รองไม่สามารถผ่านค่าสนับสนุนขั้นต่ำ ทำให้บางครั้งไอเทมในแอททริบิวต์รองไม่ปรากฏใน Large k-Itemset ใดเลย ทำให้ผลการเชื่อมความสัมพันธ์ของอัลกอริทึม HDFUP ที่ได้ไม่มีความแตกต่างจากอัลกอริทึม FUP

บทที่ 5
สรุปผลการวิจัย และข้อเสนอแนะ

5.1 สรุปผลการวิจัย

การค้นหากฎความสัมพันธ์ (Association Rule Discovery) เป็นเทคนิคหนึ่งในการทำดาต้าไมนิ่ง หลักการทำงานโดยสรุปคือ การค้นหารูปแบบความสัมพันธ์ของข้อมูลจากข้อมูลที่มีอยู่เป็นจำนวนมากในฐานข้อมูลเพื่อนำไปใช้ในการวิเคราะห์ และทำนายคาดการณ์ (Prediction) จัดข้อมูลเหล่านั้นให้อยู่ในรูปแบบของกฎความสัมพันธ์ การค้นหากฎความสัมพันธ์ที่มีอยู่ในฐานข้อมูลจำเป็นต้องค้นหาค่า Large k-itemsets ซึ่งค่า Large k-itemsets เป็นค่าของไอเทมเซต (itemsets) ที่ประกอบไปด้วยจำนวน k item โดยค่า k = 1, 2, 3, .., n อาจจะมีจำนวนเท่ากับหรือมากกว่าค่าสนับสนุนขั้นต่ำ (Minimum Support) ที่กำหนดไว้ เมื่อได้ค่า Large k-itemsets แล้วจะนำค่าดังกล่าวมาสร้างเป็นกฎความสัมพันธ์ให้อยู่ในรูปแบบกฎ **IF** X **THEN** Y ซึ่งในแต่ละกฎประกอบด้วย 2 ส่วนคือ ส่วนด้านซ้ายของกฎ (Left-hand side) และส่วนด้านขวาของกฎ (Right-hand side) ถ้าด้านซ้ายมีเงื่อนไขที่เป็นจริง จะทำให้ส่วนด้านขวาของกฎเป็นจริง

ปัจจุบันในฐานข้อมูลมีการจัดเก็บข้อมูลในรูปแบบทรานแซคชั่นหลายมิติ ซึ่งการค้นหากฎความสัมพันธ์มิติเดียวเป็นการค้นหากฎความสัมพันธ์ที่ไม่สอดคล้องกับรูปแบบการจัดเก็บในฐานข้อมูล ดังนั้นจึงได้นำการค้นหากฎความสัมพันธ์แบบมิติผสมมาใช้ในการค้นหากฎความสัมพันธ์ที่มีอยู่ภายในฐานข้อมูลแบบทรานแซคชั่นหลายมิติ เพื่อให้สอดคล้องกับการจัดเก็บข้อมูลและสามารถตอบความสัมพันธ์ได้อย่างครบถ้วนถูกต้องทุกความสัมพันธ์ ซึ่งการจัดเก็บข้อมูลโดยทั่วไปมีการปรับปรุงข้อมูลให้มีความทันสมัย ทำให้มีผลต่อความถูกต้องของกฎความสัมพันธ์

อัลกอริทึม FUP มีผู้นิยมนำมาศึกษาวิจัยพัฒนาในการค้นหากฎความสัมพันธ์ เนื่องจากอัลกอริทึมดังกล่าวทำการค้นหากฎความสัมพันธ์เมื่อมีการเพิ่มขึ้นของข้อมูล ซึ่งได้นำค่า Large Itemsets ที่เคยค้นหาไว้มาใช้ประโยชน์ เพื่อลดการสแกนในฐานข้อมูลเมื่อมีการเพิ่มขึ้นของข้อมูล

งานวิจัยนี้เป็นแนวความคิดที่นำการค้นหากฎความสัมพันธ์แบบมิติผสมใช้ร่วมกับอัลกอริทึม FUP ในการค้นหากฎความสัมพันธ์ของข้อมูลที่เพิ่มขึ้นในฐานข้อมูลที่มีการเก็บทรานแซคชั่นแบบหลายมิติ

ในการค้นหากฎความสัมพันธ์แบบมิติผสมสำหรับการค้นหากฎความสัมพันธ์เมื่อมีการเพิ่มข้อมูลด้วยอัลกอริทึม HDFUP จากการทดลองทั้ง 3 ชุด แสดงให้เห็นความแตกต่างของการเชื่อมความสัมพันธ์สร้าง 2-Itemsets ของอัลกอริทึม HDFUP ซึ่งอัลกอริทึมดังกล่าวมีจำนวนไอเทมเซตน้อยกว่าเมื่อเปรียบเทียบกับอัลกอริทึม FUP เนื่องจากการเชื่อมความสัมพันธ์ของอัลกอริทึม FUP

เป็นการเชื่อมความสัมพันธ์ในทุกไอเทมเซต รวมถึงการเชื่อมความสัมพันธ์ภายในแอททริบิวต์ซึ่งทำให้ได้ไอเทมเซตที่ไม่ปรากฏในทรานแซคชั่นแบบหลายมิติอย่างแน่นอน แต่อัลกอริทึม HDFUP หลีกเลี่ยงการเชื่อมความสัมพันธ์ภายในแอททริบิวต์รอง และอัลกอริทึมเอพริออรีแบบการค้นหากฎความสัมพันธ์แบบมิติผสม ภายหลังการเชื่อมความสัมพันธ์ทำให้ได้ไอเทมเซตเช่นเดียวกับอัลกอริทึม HDFUP เนื่องจากใช้วิธีในการค้นหากฎความสัมพันธ์เดียวกับอัลกอริทึม HDFUP และประสิทธิภาพด้านเวลาอัลกอริทึม HDFUP ใช้เวลาน้อยกว่าอัลกอริทึม FUP เนื่องจากอัลกอริทึม HDFUP มีการเชื่อมไอเทมเซตในขั้น 2-Itemsets มีจำนวนน้อยกว่าอัลกอริทึม FUP มีผลให้ไอเทมเซตใน Candidate Itemset มีจำนวนน้อยกว่าอัลกอริทึม FUP ทำให้การเข้าไปค้นหาค่าสนับสนุนในฐานข้อมูลใหม่ใช้เวลาน้อยกว่าอัลกอริทึม FUP ด้วย ผลการทดลองของชุดการทดลองที่ 1 และ 2 ในการค้นหาไอเทมเซตขั้น 3-Itemsets ขึ้นไปใช้เวลาใกล้เคียงกัน ในชุดการทดลองที่ 3 อัลกอริทึม HDFUP สามารถช่วยลดเวลาในการสร้างไอเทมเซตตั้งแต่ขั้น 3-Itemsets เนื่องจากผลการเชื่อมความสัมพันธ์ของอัลกอริทึม HDFUP หลีกเลี่ยงการเชื่อมสัมพันธ์ที่ไม่เกิดขึ้นจริงในฐานข้อมูลแบบทรานแซคชั่นหลายมิติทำให้ช่วยลดการตรวจสอบในการเช็คความเป็นซับเซตของไอเทมเซตที่สร้างขึ้นมา และอัลกอริทึมเอพริออรีแบบค้นหากฎความสัมพันธ์แบบมิติผสมใช้เวลาในการประมวลผลมากกว่าอัลกอริทึม HDFUP และอัลกอริทึม FUP เนื่องจากอัลกอริทึมเอพริออรีแบบค้นหากฎความสัมพันธ์แบบมิติผสมมีการนำไอเทมเซตทุกตัวที่เชื่อมได้ทั้งหมดไปเป็น Candidate Itemsets แต่อัลกอริทึม HDFUP และอัลกอริทึม FUP มีการคัดไอเทมเซตที่มีอยู่ใน L_k ออก ทำให้จำนวน Candidate Itemsets ของอัลกอริทึมเอพริออรีแบบค้นหากฎความสัมพันธ์แบบมิติผสมมีจำนวนมากกว่าอัลกอริทึม HDFUP และอัลกอริทึม FUP ทำให้การค้นหาในฐานข้อมูลมีจำนวนครั้งมากกว่าและมีผลทำให้ใช้เวลาในการค้นหามากตามด้วย

ผลการทดลองทั้ง 3 ชุด มีการเกิดไอเทมในแอททริบิวต์รองมีจำนวนน้อยในชุดข้อมูลที่ใช้ทำการทดลอง ทำให้ได้ผลความต่างด้านเวลาไม่มาก เนื่องจากการค้นหากฎความสัมพันธ์แบบมิติผสมมีจุดเด่นในเรื่องการลดจำนวนไอเทมเซตในขั้นการเชื่อมความสัมพันธ์สร้าง 2 –Itemsets เพื่อนำไปค้นหาในฐานข้อมูลเพิ่มใหม่ ซึ่งมีผลทำให้เวลาลดลง

5.2 ข้อเสนอแนะ

งานวิจัยนี้เป็นวิธีการค้นหากฎความสัมพันธ์แบบมิติผสมกรณีเพิ่มขยายฐานข้อมูลที่เก็บทรานแซคชั่นแบบหลายมิติเท่านั้น ดังนั้นการพัฒนาในภายหน้าให้สามารถค้นหากฎความสัมพันธ์แบบมิติผสมในกรณีการลดลงของข้อมูลในฐานข้อมูลได้ จะทำให้มีประสิทธิภาพครอบคลุมการปรับปรุงทั้งทางด้านการเพิ่มขึ้นและลดลงของข้อมูล งานวิจัยนี้ให้น้ำหนักความสำคัญสำหรับข้อมูล

ที่มีอยู่เดิมและข้อมูลที่เพิ่มใหม่ให้ค่าความสำคัญเท่ากัน ซึ่งอาจเพิ่มกรณีการให้น้ำหนักในด้านความสำคัญสำหรับข้อมูลใหม่มากกว่าข้อมูลที่มีอยู่เดิม

เอกสารอ้างอิง

[1] Agrawal, R., Imielinski, T., and Swami, A. 1993. "Mining association rules between sets of items in large databases." **Proceeding of the 1993 ACM SIGMOD Confonference on Washington DC**, USA.

[2] Agrawal, R. and Srikant, R. 1994. "Fast algorithms for mining association rules." **Proceedings of 20 th VLDB Conference Santiago**, Chile, pp. 487-499.

[3] Cheung, D.W., Han, J., Ng, V. T. and Wong,C.Y. 1996. "Maintenance of Discovered Association Rules in Large Databases: An incremental updating technique," **In 12th IEEE International Conference on Data Engineering**, pp.106-114.

[4] Thomas, S., Bodagala, S., Alsabti, K., and Ranka, S. 1997. "An efficient algorithm for the incremental updation of association rules in large databases." **In Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining**, New Port Beach, California.

[5] Chang, C., Li, Y. and Lee, J. 2005. "An efficient algorithm for incremental mining of association rules," **Proceedings of the 15th international workshop on research issues in data engineering: stream data mining and applications** , IEEE 2005.

[6] Zequn Zhou. 2000. "Maintaining Incremental Data Mining Association Rules." University of Windsor, Canada.

[7] Zhang, S., Zhang, C., Yan X. 2003. "Post-mining: maintenance of association rules by weighting." **Information Systems 28**" Elsevier Science, pp.691-707

[8] Yan Xin and Shi-Guang Ju. 2003. "Mining conditional Hybrid-dimension association rules on the basis of Multi-dimensional transaction database." **Proceeding of the Second International Conference on Machine Learning and Cybernetics,** Xi'an, pp.216-221.

[9] W.-G. Teng and M.-S. Chen. 2005. "Incremental mining on association rules." Springer-Verlag Berlin Heidelberg, pp.125-162.

[10] Jiawei Han and Micheline Kamper. 2006. **"Data Mining : concepts and techniques.**", 2nd . San Francisco : Morgan Kaufmann Pushers.

ภาคผนวก

ผลงานวิจัยที่ได้รับการตีพิมพ์เผยแพร่

1. S. Chatsettakul, **W.** Kreesuradej, "**Hybrid Dimension Fast Update Algorithm**," The 24th International Technical Conference on Circuits/Systems, Computers and (ITC-CSCC 2009), pp. 879-882, Jeju, Korea, July 5-8, 2009.


Sponsored by
Co-sponsored by
In cooperative with
ITC-CSCC 2009
The 24th International Technical Conference on
Circuits/Systems, Computers and Communications
Information
Table of Contents
Search This CD-ROM
CD-ROM Help
Exit
July 5~8, 2009
Jeju KAL Hotel, Jeju, Korea

Hybrid-Dimension Fast Update Algorithm

S.Chatsettakul, W.Kreesuradej*
Faculty of Information Technology, King Mongkut's Institute of Technology Ladkrabang, Bangkok, 10520, Thailand, Phone: (668)9173311, Fax: (662)7234910
*ople_chat@hotmail.com, worapoj@it.kmitl.ac.th**

Abstract

In this paper, an incremental updating technique is proposed for maintaining hybrid-dimension association rules. We propose a new incremental algorithm for discovering hybrid-dimension association rules. The new algorithm, called Hybrid-Dimension Fast Update Algorithm (HDFUP), is modified from Fast Update Algorithm.

Keywords: knowledge discovery, hybrid-dimension association rules, multi-dimension transaction database, incremental association rule, fast update algorithm

1. Introduction

Data mining is one of knowledge discovery in database (KDD) [1] which has recently received wide attention from many researchers. Data mining can be applied in many areas such as the retail industry and the finance sector.

Association rule mining is a data mining technique for discovering the interesting association relationships among huge amounts of business transaction records. An influential algorithm for association rule mining is Apriori algorithm. Apriori Algorithm [2] is algorithm for association rules mining. It uses prior knowledge of large itemsets and uses an iterative approach known as a level-wise search.

The association rules can be categorized into two kinds: single dimensional rules and multidimensional rules, according to the numbers of predication involved in the rules. The single dimensional rules mainly deal with the intra dimension associations. Regarding to multidimensional association rules, it can be divided into inter dimension association rules and hybrid-dimension association rules.

- Inter dimension association rules are the multidimensional association rules which have no repeated predicates. An example such a rule is:
  age(X, "20...29") ∧ occupation(X, "student") ⇒ buys(X, "laptop_computer"),
  Three predicates (age, occupation and buy) occurs only once in the rule.
- Hybrid dimension association rules are multidimensional association rules with repeated predications. An example of such a rule is:
  age(X, "20...29") ∧ buys(X, "laptop_computer") ⇒ buys(X, "laser_printer")

The rules discovered from a database only reflect the current state of the database. However, in a dynamic database where new transactions are inserted frequently, association rules discovered in the previous database possibly no longer valid and interesting rules in the updated database. As a result, new business information such as changing customer preference, or new seasonal trends may not be discovered.

For dynamic databases, several incremental updating techniques have been developed for mining association rules. One of the major work for incremental association rule mining is FUP algorithm [3] that was presented by Cheung. Fast Update Algorithm (FUP) maintains the association rules from newly inserted transactions. It uses the previous association rules mining large itemsets to generate new large itemsets association rules. The following properties are usually used in algorithm FUP.

- An original large itemsets X, i.e., $X \in L$, becomes loser in the update database $DB \cup db$ if and only if $X.support_{UD} < (s \times d)$.
- An original infrequent itemsets X, i.e., $X \notin L$, may become large itemsets in update database only if $X.support_d \geq s \times d$.
- If a k-itemset X whose (k-1)-subset becomes loser, i.e., the subset is in L_{k-1} but not in L'_{k-1}, X must be loser in update database. similarly

However, Fast Update Algorithm (FUP) can only be used for discovering single-dimensional association rules. Thus, we propose a new incremental algorithm for discovering hybrid dimension association rules.

The new algorithm, called Hybrid-Dimension Fast Update Algorithm (HDFUP), is based on FUP.

2. Hybrid-Dimension FUP Algorithm

Basically, the new algorithm is modified from FUP. To discover hybrid dimension association rules, the new algorithm need to have new join operation. The new join definition is defined as followings

Definition 1: The intra-dimension join

For the joining implement in $l_{k-1} \rhd\lhd l_{k-1}$, the first (k-2) items of l_1 and l_2 are the same as the following items: $(l_1[1] = l_2[1]) \cap (l_1[2] = l_2[2]) \cap \ldots \cap (l_1[k-2] = l_2[k-2]) \cap (l_1[k-1] < l_2[k-1]$. The result is $l_1[1]l_1[2]\ldots l_1[k-1]l_2[k-1]$.

Definition 2: The inter-dimension join

For the joining implement in $l_{k-1} \rhd\lhd l_{k-1}$, the items from the 2^{th} to the (k-1) in l_1 are the same as the items from the 1^{st} to the $(k-2)^{th}$ in l_2 and $l_1[1] < l_2[k-1]$ as follows: $(l_1[2] = l_2[1]) \cap (l_1[3] = l_2[2]) \cap \ldots \cap (l_1[k-1] = l_2[k-2]) \cap l_1[1] < l_2[k-1])$. The result is $l_1[1]l_2[1]\ldots l_2[k-2]l_2[k-1]$.

```
Input. DB: the original database (D is equal to total number
       of transactions);
       L1: the set of all large k- itemsets in DB,
       where k= 1, ..., s.
       db: an increment database (with its size equal to d).
Output. L': The set of all large itemsets in DB∪db.
Method:
       The 1st iteration:/* find L'1, the set of all
       large 1-itemsets in DB∪db */
       W = L1, C = φ, L1' = φ ; P = φ ,
       /* W: winners, C: candidate sets, L1': initialized,
       P: for optimization */
       for all T ∈ db do /* scan db */
        for all 1-itemset X ∈ T do {
          if X ∈ W then X.support_UD++,
          else {
            if X ∉ C
            then { C = C ∪ {X}; X.support_D = 0;}
            /*init the support cont and add X into C */
            X.support_UD++;}
        },
       for all X ∈ W do /* put winners into L1' */
        if X.support_UD ≥ s×(D + d)
          then L1' = L1' ∪ {X},
       for all X ∈ C do /*prune candidate sets in C*/
        if X.support_UD < s×(D + d)
          then { C = C − {X}, P = P ∪ {X},}
          /* P will be used for optimization */
       for all T ∈ DB do /* scan DB */
        for all 1-itemset X ⊆ T do {
          if X ∈ C then X.support_DB++,
          if X ∈ P then removes X from T,
          /* Transaction T is reduced */
        },
       for all X ∈ C do /* put winners into L1' */
        if X.support_UD ≥ s×(D + d)
          then L1' = L1' ∪ {X},
       return L1' /* end of the 1st iteration */
```

```
The k-th iteration: /*for k = 2 or larger, repeat this program
fragment to find L'k, the set of all large k-itemsets in the
updated database, until either L'k returned is empty or db =
W = Lk, Lk' = φ,
/*W: winners, Lk': initialized */
if k = 2
 then {C = apriori_gen1(L'k-1) - Lk }
 else {C = apriori_gen2(L'k-1) - Lk };
 /* the size-k candidate sets */
for all k-itemset X∈ W do
/* prune off losers in W */
 for all (k-1)-itemset Y ∈ Lk-1 − L'k-1 do
   if Y ⊆ X then { W = W − {X}; break;}
for all T ∈ db do { /* scan db */
 for all X ∈ (W,T) do X.support_UD++,
 /* Subset(W,T) returns all the sets in W contained in T*/
 for all X ∈ Subset(C,T) do X.support_D++,
 /* find support of all X ∈ C */
 Reduce db (T);
 /* Some items in transactions in db can be removed,
 discussed in next section */
}
for all X ∈ W do
/* put the winners from W into L'k */
if X.support_UD ≥ s×(D+d)
 then L'k = L'k ∪ {X};
for all X ∈ C do /* prune candidate sets in C*/
 if X.support_D < s×d then C = C − {X};
for all T ∈ DB do { /* scan DB */}
 for all X ∈ Subset(C,T) do X.support_D++
 Reduce_DB(T),
 /* Some items in transactions in DB can be removed,
 discussed in next section */
for all X ∈ C do
 if X.support_DB ≥ s×(D + d)
   then L'k = L'k ∪ {X};
return Lk'. /* the end of the k-th iteration */
```

Figure 1 HDFUP algorithm

```
procedure apriori_gen1 (L'k-1: Large itemsets)
{ C = null;
  for each l1 ∈ L'k-1
    for each l2 ∈ L'k-1
      if isInnerJoin (l1) or isInnerJoin (l2)
        then {
          c = l1 ⊳⊲ l2;
          InsertInto C
      for each c ∈ C
        for each (k-1)-subset s of c
          if s ∉ L'k-1
            then delete c from C
}
```

Figure 2 Apriori_gen1()

```
procedure apriori_gen2 (L'k-1: Large itemsets)
{ C = null;
  for each l1 ∈ L'k-1
    for each l2 ∈ L'k-1
      if isInnerJoin (l1) and isInnerJoin (l2)
        then //make intradimension join
        {if (l1[1] = l2[1]) ∧ (l1[2] = l2[2]) ∧ ... ∧
        (l1[k-2] = l2[k-2]) ∧ (l1[k-1] < l2[k-1])
          then {
            c = l1 ⋈ l2;
            InsertInto C;}
        }
        else //make interdimension join
          {if (l1[2] = l2[1]) ∧ (l1[3] = l2[2]) ∧ ... ∧
          (l1[k-1] = l2[k-2]) ∧ l1[1] < l2[k-1])
      then {
      c = l1 ⋈ l2;
      InsertInto C;
  }
for each c ∈ C
  for each (k-1)-subset s of c
    if s ∉ L'k-1
      then delete c from C
}
```

Figure 3 Apriori_gen2()

Based on the new join operation, the new algorithm, as shown in Figure1, 2 and 3, can be divided into 3 parts.

As for the first part, the 1 large-itemsets of updated database is determined. When new transactions are added, HDFUP scans them to generate the 1 candidate -itemsets (i.e. C_1) and then compares these 1 candidate-itemsets with the original 1 the large -itemsets. If a 1 candidate-itemset from the newly inserted transactions is also among the 1 large - itemsets from the original database, its new total count for the whole updated database can easily be calculated from its current count and previous count. If $X \in C_1$ and $X.support_{db} < s \times d$, these items are removed from candidate itemset and insert them to P for optimization. A scan of original database is performed when $X \in C_1$. In case of $X.support_{UD} \geq s \times (D+d)$, item X is inserted to L'_1. This part uses the same algorithm as FUP.

In the second part, the 2 large-itemsets of updated database is determined. C_2 is generated by applying apriori_gen1 procedure on L'_1 and removes the joined item which is similar to the original 2 large-itemsets. Any set $X \in L_2$, which has a subset Y such that $Y \in L_{k-1} - L'_{k-1}$, cannot be large and are filtered out from W. A scan of db having $X \in Subset(W,T)$ or $X \in Subset(C,T)$ is conducted in order to count the supporting value. If $X \in C_2$ and $X.support_{db} < s \times d$, these items are removed from candidate itemset. A scan of original database is performed when $X \in C_2$. In case of $X.support_{UD} \geq s \times (D+d)$, item X is inserted to L'_2. The algorithm of this part is shown in figure 2.

Regarding the last part, apriori_gen2 procedure is joined with L'_{k-1} when $k > 3$. This part is almost similar to that of the second part.

3. Experiment

As an example, a original database and Incremental database are shown in Figure 4. Both databases consist of multidimensional transactions of Product Orders, which includes two subordinate attributes: Age, Area and main attribute Order. Based on minimum support threshold equal to 0.2, figure 5 shows frequent itemsets obtained from the original database. Like FUP, hybrid-dimension association rules can be found from the frequent itemsets in figure 6.

In figure 6, $\{b,2,I_1,I_3\}$ is a frequent itemset. We can generate such a hybrid-dimension association rule:

$b \wedge 2 \wedge I_1 \Rightarrow I_3$ (support = 23%, confidence = 75%)

Age(X,b) ∧ Area(X,2) ∧ OrderID(X,I_1) ⇒ OrderID(X,I_3) (support = 23%, confidence = 75%)

This rule shows that the subordinate attributes Age and Area appear at most once, and the main attribute OrderID appears many times. Furthermore, All large-itemsets as shown in figure 6 indicate that the proposed algorithm can correctly discover all hybrid-dimension association rules.

ID	Age	Area	Order ID
1	a	1	I_2, I_4
2	a	1	I_1, I_2, I_5
3	a	2	I_2, I_3
4	b	2	I_1, I_3
5	b	2	I_1, I_2, I_4
6	a	1	I_2, I_3
7	b	2	I_1, I_3

(a)

ID	Age	Area	Order ID
8	a	3	I_1, I_5, I_6
9	b	3	I_1, I_2, I_5, I_6, I_7
10	c	3	I_2, I_5
11	c	1	I_2, I_6
12	b	1	I_5, I_6
13	b	2	I_1, I_3, I_6

(b)

Figure 4 (a) Original database
(b) Incremental database

L_1

Itemset	Support
{a}	4
{b}	3
{1}	3
{2}	4
$\{I_1\}$	4
$\{I_2\}$	5
$\{I_3\}$	4
$\{I_4\}$	2

L_2

Itemset	Support
{a,1}	3
$\{a,I_2\}$	4
$\{a,I_3\}$	2
{b,2}	3
$\{b,I_1\}$	3
$\{b,I_3\}$	2
$\{1,I_2\}$	3
$\{2,I_1\}$	3
$\{2,I_2\}$	2
$\{2,I_3\}$	3
$\{I_1,I_2\}$	2
$\{I_1,I_3\}$	2
$\{I_2,I_3\}$	2
$\{I_2,I_4\}$	2

L_3

Itemset	Support
$\{a,1,I_2\}$	3
$\{a,I_2,I_3\}$	2
$\{b,2,I_1\}$	3
$\{b,2,I_3\}$	2
$\{b,I_1,I_3\}$	2
$\{2,I_1,I_2\}$	2

L4

Itemset	Support
$\{b,2,I_1,I_2\}$	2

Figure 5 Large itemsets in original database

L_1

Itemset	Support
{a}	6
{b}	6
{1}	5
{2}	5
{3}	3
$\{I_1\}$	7
$\{I_2\}$	8
$\{I_3\}$	5
$\{I_5\}$	5
$\{I_6\}$	5

L_2

Itemset	Support
{a,1}	3
$\{a,I_2\}$	4
{b,2}	4
$\{b,I_1\}$	5
$\{b,I_3\}$	3
$\{b,I_6\}$	3
$\{1,I_2\}$	4
$\{2,I_1\}$	4
$\{2,I_3\}$	4
$\{3,I_6\}$	3
$\{I_1,I_2\}$	3
$\{I_1,I_5\}$	3
$\{I_1,I_6\}$	3
$\{I_2,I_6\}$	3
$\{I_5,I_6\}$	3

L_3

Itemset	Support
$\{a,1,I_2\}$	3
$\{b,2,I_1\}$	4
$\{b,2,I_3\}$	3
$\{b,I_1,I_3\}$	3
$\{2,I_1,I_2\}$	3

L_4

Itemset	Support
$\{b,2,I_1,I_2\}$	3

Figure 6 Large itemsets in updated database

4. Conclusion

In this paper, we propose HDFUP algorithm for discovering hybrid dimension association rules. The algorithm can guarantee to discover all hybrid-dimension association rules. In the future, further researches and experiments on the proposed algorithm will be presented.

References

[1] Han J., M. Kamber, "*Data mining: Concepts and Techniques*", Morgan Kaufmann Publishers, San Francisco, CA, pp. 227-256 ,2006

[2] R. Agrawal, R.srikant, "Fast Algorithms for Mining Association Rules", *Proceedings of the 20th International Conference on Very Large Data Bases*, pp. 478-499, September 1994.

[3] David W. Cheung, Jiawei Han, Vincent T. Ng, C.Y. Wong, "Maintenance of Discovered Association Rules in Large Databases: An Incremental Updating Technique", *Proceedings of the 12th International Conference on Data Engineering*, pp. 106-114, February 1996.

[4] Yan Xin, Shi-Guang Ju, "Mining Conditional Hybrid-Dimension Association Rules on The Basis of Multidimensional Transaction Database", Proceedings of the Second International Conference on Machine Learning and Cybernetics, pp.216-221, November 2003.

[5] W.-G. Teng, M.-S. Chen, "Incremental Mining on Association Rules", *Studies in Fuzziness and soft computing*, pp. 125-162, October 2005.

ประวัติผู้เขียน

น.ส.สุภาพร ฉัตรเศรษฐกูล เกิดเมื่อวันที่ 5 มีนาคม พ.ศ.2521 ที่จังหวัดปทุมธานี สำเร็จการศึกษาปริญญาตรีวิทยาศาสตรบัณฑิต สาขาวิทยาการคอมพิวเตอร์ คณะวิทยาศาตร์ มหาวิทยาลัยรามคำแหง ในปีการศึกษา 2546 และเข้าศึกษาต่อในระดับปริญญาโท หลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาเทคโนโลยีสารสนเทศ แขนงวิชาวิทยาการสารสนเทศ คณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ในปีการศึกษา 2548