วิธีการใหม่สำหรับงานสร้างกฎความสัมพันธ์ของข้อมูลการซื้อขาย

NEW APPROACH OF ASSOCIATION RULES FOR A MARKET BASKET ANALYSIS

เชาวนี ศรีวิศาล

CHOUVANEE SRIVISAL

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตร์มหาบัณฑิต

สาขาวิชาเทคโนโลยีสารสนเทศ

บัณฑิตวิทยาลัย

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ. 2545

ISBN 974-648-807-4

NEW APPROACH OF ASSOCIATION RULES FOR A MARKET BASKET ANALYSIS

CHOUVANEE SRIVISAL

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

MASTER OF SCIENCE IN INFORMATION TECHNOLOGY

SCHOOL OF GRADUATE STUDIES

KING MONGKUT'S INSITUTE OF TECHNOLOGY LADKRABANG

2002

ISBN 974–648-807-4

หัวข้อวิทยานิพนธ์	วิธีการใหม่สำหรับงานสร้างกฎความสัมพันธ์ของข้อมูลการซื้อขาย
นักศึกษา	นางสาว เชาวนี ศรีวิศาล
รหัสประจำตัว	42067010
ปริญญา	วิทยาศาสตรมหาบัณฑิต
สาขาวิชา	เทคโนโลยีสารสนเทศ
พ.ศ.	2545
อาจารย์ผู้ควบคุมวิทยานิพนธ์	รศ. ดร. วิเชียร เปรมชัยสวัสดิ์

บทคัดย่อ

งานวิจัยนี้ได้นำเสนอวิธีการใหม่สำหรับการหากฎความสัมพันธ์ของงานทางการตลาด โดยกระบวนการใหม่นั้นจะมีการอ่านฐานข้อมูลเพียง 1 ครั้ง โดยมีการทำงานหลักๆอยู่ 2 ขั้นตอนดังนี้ ขั้นตอนที่ 1 เป็นกระบวนการเตรียมข้อมูลก่อนที่จะนำเข้าไปหารูปแบบทั้งหมด โดยในขั้นตอนนี้จะเรียงลำดับทรานแซกชั่นใหม่โดยเรียงตามจำนวนไอเท็มในทรานแซกชั่นนั้นๆ เหตุที่ทำเช่นนี้เพื่อที่จะให้ช่วยลดจำนวนรอบในอ่านฐานข้อมูลเพื่อสร้างรูปแบบไอเท็มเซตทั้งหมด ส่วนขั้นตอนที่ 2 เป็นการหารูปแบบไอเท็มเซตที่น่าสนใจทั้งหมดโดยเทียบกับค่าสนับสนุนต่ำสุด เพื่อแสดงให้เห็นประสิทธิภาพของวิธีการที่นำเสนอนี้ได้ทำการทดลองวิธีการที่นำเสนอนี้เปรียบเทียบกับวิธีเอไพออริ(Apriori) และดีไอซี (DIC) โดยพิจารณาจากเวลาที่ใช้ในการหารูปแบบไอเท็มเซต พบว่าวิธีการที่นำเสนอนี้ทำงานได้เร็วกว่าวิธีการเอไพโอริ (Apriori) และดีไอซี (DIC) ในขณะที่ให้ผลลัพธ์อย่างเดียวกับ โดยในการทดลองจะมีการเปลี่ยนแปลงค่า ค่าสนับสนุนต่ำสุด และจำนวนทรานแซกชั่นในฐานข้อมูล

Thesis Title	New Approach of Association rules for A Market Basket Analysis
Student	Miss Chouvanee Srivisal
Student ID.	42067010
Degree	Master of Science
Programme	Information Technology
Year	2002
Thesis Advisor	Assoc.Prof.Dr Wichian Premchaiswadi

ABSTRACT

In this paper proposes a new scheme of association rule algorithm for market basket. The algorithm, which uses one passes over the data, consists of two steps namely pre-processing and patterns classification. In the first step all transactions are reorded transaction by the number of items in transaction. Then the second step is uesd to determining all groups of items, which frequently appear together in transaction. The algorithm is implemented and tested using variable minimum support and variable size of synthetic data. The experimental results are compared with the Apriori and DIC Algorithm. The experimental results show that the execution time of the propose algorithm is less than that of the Apriori and DIC algorithm significantly.

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงได้อย่างดี ด้วยคำแนะนำและคำปรึกษาเกี่ยวกับแนวทางในการทำวิจัยเพื่อที่จะหาวิธีการใหม่ในสำหรับการสร้างกฎความสัมพันธ์ของข้อมูลการซื้อขาย รวมทั้งได้ทำการทดสอบและเปรียบเทียบจาก รศ.ดร. วิเชียร เปรมชัยสวัสดิ์ ซึ่งเป็นอาจารย์ผู้ควบคุมวิทยานิพนธ์ ทั้งนี้ผู้วิจัยรู้สึกซาบซึ้งในความอนุเคราะห์จากท่านและขอกราบขอบพระคุณเป็นอย่างสูง

ขอขอบพระคุณ คณาจารย์ในคณะเทคโนโลยีสารสนเทศทุกท่านที่ได้ประสิทธิ์ประสาทความรู้ต่างๆ รวมทั้งให้คำปรึกษาเมื่อมีข้อสงสัย เพื่อใช้เป็นพื้นฐานในการทำงานวิจัยครั้งและแก้ปัญหาที่พบในการทำงานวิจัยครั้ง นอกจากนั้นก็ต้องขอขอบคุณเจ้าหน้าที่คณะเทคโนโลยีสารสนเทศทุกท่านที่ช่วยในการติดต่อประสานงานในการทำกิจกรรมทุกอย่าง

ขอขอบคุณหัวหน้าและเพื่อนอาจารย์ทุกท่านของภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยสงขลานครินทร์ และมหาวิทยาลัยสงขลานครินทร์ซึ่งเป็นสถาบันที่สนับสนุนทุนให้มาศึกษาต่อในระดับมหาบัณฑิตในครั้ง

ขอขอบคุณ คุณแม่ ที่คอยเป็นกำลังใจและเป็นทุกๆ อย่างในการทำงานวิจัยครั้งนี้เป็นอย่างมาก นอกจากนั้นก็ต้องขอขอบคุณเพื่อนๆ นักศึกษาทุกๆ คนที่คอยให้ความช่วยเหลือให้คำแนะนำต่างๆ พร้อมทั้งช่วยตรวจเทียบและแก้ไขข้อบกพร่องหรือความผิดพลาดต่างๆ ที่เกิดขึ้น จนสำเร็จสมบูรณ์ยิ่งขึ้น และยังเป็นกำลังใจให้ผู้วิจัยอย่างใกล้ชิดตลอดมา

สุดท้ายขอขอบคุณบัณฑิตวิทยาลัยที่ได้ให้ทุนสนับสนุนการไปนำเสนอผลงานวิจัยนี้และด้วยคุณค่าและประโยชน์อันพึงมีจากวิทยานิพนธ์ฉบับนี้ ผู้วิจัยขอมอบแด่ผู้มีพระคุณทุกท่าน

เชาวนี ศรีวิศาล

สารบัญ

หน้า

บทคัดย่อภาษาไทย I

บทคัดย่อภาษาอังกฤษ II

กิตติกรรมประกาศ III

สารบัญ IV

สารบัญตาราง VII

สารบัญภาพ VIII

บทที่ 1 บทนำ 1

1.1 ความเป็นมาและความสำคัญของปัญหา 1

1.2 วัตถุประสงค์ของงานวิจัย 2

1.3 ทฤษฎีหรือแนวคิดที่ใช้ในงานวิจัย 3

1.4 แผนการดำเนินงานวิจัย 3

1.4.1 ขั้นตอนการดำเนินงานวิจัย 3

1.4.2 ระยะเวลาที่ใช้ในการแต่ละขั้นตอน 3

1.5 ขอบเขตงานวิจัย 4

1.6 ประโยชน์ที่คาดว่าจะได้รับ 4

บทที่ 2 ทฤษฎีและหลักการที่เกี่ยวข้อง 5

2.1 หลักการทำงานทั่วไป 5

2.2 ข้อกำหนดของปัญหา 5

2.2.1 ข้อกำหนดทั่วไปของ Association Rule 5

2.3 รูปแบบโครงสร้างข้อมูล 6

2.3.1 รูปแบบข้อมูลแนวนอน 7

2.3.2 รูปแบบข้อมูลแนวตั้ง 8

2.4 รูปแบบการเข้าถึงรูปแบบไอเท็มเซตในทรี 8

2.4.1 การค้นหาในแนวกว้าง 8

2.4.2 การค้นหาในแนวลึก 9

2.5 กระบวนการทำงานของวิธีการ Apriori 9

สารบัญ(ต่อ)

หน้า

2.6 กระบวนการ DIC 11
2.6.1 การนับจำนวนรูปแบบไอเท็มเซตสูงสุด 13
2.6.2 ขั้นตอนการทำงานของกระบวนการ DIC 14
2.6.3 โครงสร้างของการเก็บข้อมูลรูปแบบไอเท็มเซต 15
2.7 สรุปหลักการทำงานที่เกี่ยวข้อง 15

บทที่ 3 ทฤษฎีและหลักการทำงานของวิธีการใหม่ 16
3.1 กระบวนการทำงานแบบใหม่ 16
3.2 ขั้นตอนการทำงานของกระบวนการใหม่ 17
3.2.1 การจัดเตรียมข้อมูลก่อนการประมวลผล 17
3.2.2 การประมวลผล 17
3.3 ตัวอย่างขั้นตอนการทำงานของวิธีการใหม่ 18
3.4 รูปแบบโครงสร้างข้อมูล 20
3.5 โครงสร้างของการเก็บข้อมูลรูปแบบไอเท็มเซต 21

บทที่ 4 การทดสอบและวัดประสิทธิภาพ 24
4.1 การเตรียมเครื่องมือสำหรับการทดสอบ 24
4.2 การสร้างชุดข้อมูลสมมุติ 25
4.3 รูปแบบการทำงานของโปรแกรม 26
4.3.1 ไฟล์ข้อมูล 26
4.3.2 ไฟล์คุณสมบัติของข้อมูล 26
4.4 การวัดประสิทธิภาพ 28

บทที่ 5 สรุปผลการวิจัยและข้อเสนอแนะ 36
5.1 ข้อสรุป 36
5.2 ผลที่ได้รับจากการทำวิทยานิพนธ์ 37
5.3 ปัญหา 37
5.4 ข้อเสนอแนะ 38

สารบัญ(ต่อ)

หน้า

เอกสารอ้างอิง 39

ภาคผนวก ผลงานที่ได้รับการตีพิมพ์ 40

ประวัติผู้เขียน 54

สารบัญตาราง

ตารางที่ หน้า

2.1 ตัวอย่างการจัดเก็บข้อมูลแนวนอนแบบแทนด้วยรหัส 7
2.2 ตัวอย่างการจัดเก็บข้อมูลแนวนอน 7
2.3 ตัวอย่างการเก็บข้อมูลแบบแนวตั้ง 8
4.1 ตัวแปรที่ใช้ในการสร้างฐานข้อมูลสมมุติ 25
4.2 ตัวอย่างการกำหนดตัวแปรในฐานข้อมูลสมมุติ 25
4.3 แสดงค่าสนับสนุนต่ำสุดที่ใช้ในการการทดสอบสำหรับชุดข้อมูล 29
4.4 แสดงเวลาเฉลี่ยในการหารูปแบบไอเท็มเซตที่น่าสนใจ 33

สารบัญภาพ

ภาพที่ หน้า

2.1 ตัวอย่างแฮชทรี 9
2.2 ขั้นตอนการทำงานของวิธี Apriori 10
2.3 ตัวอย่างขั้นตอนการทำงานของวิธี Apriori 11
2.4 รอบในการอ่านข้อมูลของวิธีการ DIC 12
2.5 โครงสร้างทรีของรูปแบบไอเท็มสำหรับวิธีการ DIC เมื่อจบการทำงาน 13
3.1 จำนวนรอบในการอ่านข้อมูลของวิธีการ Apriori , DIC และ New Approach 18
3.2 ลำดับการเข้าไปเพิ่มค่าสนับสนุนให้กับรูปแบบไอเท็มต่างๆ 19
3.3 ขั้นตอนการทำงานของวิธีการใหม่เมื่อกำหนดค่าสนับสนุนต่ำสุดเท่ากับ 50% 19
3.4 ตัวอย่างข้อมูลที่จัดเก็บแบบข้อมูลแนวนอน 21
3.5 ตัวอย่างอธิบายรูปแบบไอเท็มที่จัดเก็บในโครงสร้างแฮชทรีที่จำลองขึ้น 22
4.1 ผลลัพธ์จากการทำงานของโปรแกรมกับข้อมูล Transa8 ที่ค่าสนับสนุนต่ำสุด = 0.25% 27
4.2 กราฟแท่งแสดงเวลาในการหารูปแบบไอเท็มที่น่าสนใจของฐานข้อมูล Transa8 29
4.3 กราฟแท่งแสดงเวลาในการหารูปแบบไอเท็มที่น่าสนใจของฐานข้อมูล T10I2D100K 30
4.4 กราฟแท่งแสดงเวลาในการหารูปแบบไอเท็มที่น่าสนใจของฐานข้อมูล T10I4D100K 30
4.5 กราฟแท่งแสดงเวลาในการหารูปแบบไอเท็มที่น่าสนใจของฐานข้อมูล T10I4D200K 31
4.6 กราฟแท่งแสดงเวลาในการหารูปแบบไอเท็มที่น่าสนใจของฐานข้อมูล T10I6D200K 31
4.7 กราฟแท่งแสดงเวลาในการหารูปแบบไอเท็มที่น่าสนใจของฐานข้อมูล T20I6D100K 32
4.8 กราฟแท่งแสดงเวลาเฉลี่ยในการหารูปแบบไอเท็มที่น่าสนใจแต่ละตัวของวิธีการทั้ง 3 วิธี 34

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในปัจจุบันองค์กรทางธุรกิจกำลังเติบโตและมีการเก็บข้อมูลต่างๆไว้มากมาย ซึ่งข้อมูลเหล่านี้สามารถที่จะนำไปใช้ให้เกิดประโยชน์ต่อการวางนโยบายขององค์กร จึงทำให้มีความต้องการในการที่จะดึงข้อมูลทางธุรกิจที่ได้เก็บรวบรวมไว้จำนวนมากออกมาเพื่อใช้ประโยชน์ เช่นในกระบวนการวินิจฉัยเพื่อวางนโยบายทางการตลาดของผู้บริหารในระดับสูงอาจได้มาจากข้อมูลรูปแบบการซื้อสินค้าของลูกค้า นอกจากที่จะนำไปใช้เพื่อการวางนโยบายทางด้านการตลาดแล้วอาจนำข้อมูลเหล่านี้ไปใช้สำหรับการตัดสินใจในเรื่องทิศทางการจัดเก็บสินค้าให้เหมาะสมกับช่วงเวลาอีกด้วย กระบวนการวินิจฉัยข้อมูลโดยอัตโนมัตินี้ทำขึ้นเพื่อที่จะทำการค้นหาความรู้ที่แอบซ่อนอยู่ในข้อมูลจำนวนมากซึ่งกระบวนการเหล่านี้เรียกว่า Knowledge discovery [1, 2, 5, 6] โดยวิธีการแบบนี้เป็นลักษณะการทำงานแบบหนึ่งของงานทางด้านดาต้าไมนิ่ง (Data Mining) และวิธีการที่นิยมนำมาใช้เพื่อการค้นหาความรู้เกี่ยวกับรูปแบบการซื้อสินค้าของลูกค้านั้นคือวิธีการแบบ แอสโซซิเอชั่นรูล์ (Association Rule) โดยผลลัพธ์ที่ได้จากกระบวนการนี้จะอยู่ในรูปแบบของการระบุความน่าจะเป็นเมื่อมีการซื้อสินค้าชนิดหนึ่ง แล้วจะทำให้มีการซื้อสินค้าอีกชนิดหนึ่งในการซื้อสินค้าครั้งเดียวกันกี่เปอร์เซ็นต์ ตัวโปรแกรมต้นแบบนี้ใช้ในการวิเคราะห์ข้อมูลด้านการขายหรือบาสเก็ตดาต้า (Basket data) โดยข้อมูลการขายที่ใช้สำหรับการพิจารณานี้ประกอบด้วยรายการสินค้าที่ลูกค้าซื้อ (Items) และรายการแสดงทรานแซกชั่น (Transaction Identifier)

งานค้นหาความรู้ของวิธีการแอสโซซิเอชั่นรูลย์สำหรับข้อมูลด้านการขาย เมื่อเรากำหนดให้ฐานข้อมูลที่ประกอบไปด้วยทรานแซกชั่นจำนวนมาก โดยที่แต่ละทรานแซกชั่นนั้นแสดงสินค้าหรือไอเท็ม (Item) ที่ซื้อโดยลูกค้าในครั้งหนึ่งๆ งานของวิธีการแอสโซซิเอชั่นรูลย์นั้นมีอยู่ 2 ขั้นตอนหลัก ขั้นตอนที่ 1 คือ การหาชุดของไอเท็มที่มีความน่าสนใจ โดยความน่าสนใจนี้พิจารณาจากค่าสนับสนุนของแต่ละรูปแบบเปรียบเทียบกับค่าสนับสนุนที่กำหนดให้ ถ้าหากว่าค่าสนับสนุนของรูปแบบใดมากกว่าค่าสนับสนุนที่กำหนดก็จะถือว่ารูปแบบไอเท็มนั้นๆ น่าสนใจ และนำรูปแบบดังกล่าวไปพิจารณาต่อในขั้นตอนที่ 2 ต่อไป ขั้นตอนที่ 2 ของวิธีการแอสโซซิเอชั่นรูลย์คือการสร้างกฎจากรูปแบบที่ได้มาจากขั้นตอนที่ 1 ขั้นตอนที่ยากขั้นตอนหนึ่งในวิธีการแอสโซซิเอชั่นรูลย์นั้นคือขั้นตอนของการหารูปแบบที่มีค่ามากกว่าค่าสนับสนุน ดังนั้นในงานวิจัยนี้เราจึงเน้นทำเฉพาะในส่วนที่ขั้นตอนที่ 1 เท่านั้น

สำหรับงานวิจัยนี้ได้ใช้แนวคิดจากวิธีการแอสโซซิเอชั่นแบบเอไพออริ (Apriori Algorithm) ซึ่งเป็นวิธีที่ได้รับความนิยมวิธีการหนึ่ง และแบบดีไอซี (DIC) ซึ่งเป็นกระบวนการที่มีการพัฒนาขึ้นมาภายหลังและมีความรวดเร็วกว่าวิธีการแบบเอไพออริ (Apriori) เพราะมีการอ่านฐานข้อมูลน้อยครั้งกว่าวิธีการเอไพออริ (Apriori) และในวิธีการแบบใหม่นี้เราก็ได้นำเอาแนวคิดจากวิธีการดีไอซี (DIC) มาทำการปรับปรุงเพิ่มเติมเพื่อให้มีการทำงานในขั้นตอนของการหากลุ่มไอเท็มที่มีค่ามากกว่าค่าสนับสนุนที่กำหนดซึ่งปรากฏในทรานแซกชั่นพร้อมกัน โดยเราพบว่าปัญหาในการหา กลุ่มไอเท็มดังกล่าวนั้นสำหรับในวิธีการแบบเอไพออริ (Apriori) เราพบว่าวิธีการนี้จะต้องอ่านฐานข้อมูลหลายครั้งเท่ากับจำนวนระดับกลุ่มไอเท็มจนกว่าจะไม่มีรูปแบบใดที่มีค่ามากกว่าค่าสนับสนุนที่กำหนดไว้ (minimum support) แล้วก็จะหยุด ดังนั้นอาจทำให้เสียเวลามากในการที่จะต้องอ่านฐานข้อมูลหลายรอบ ซึ่งในการอ่านแต่ละครั้งนั้นย่อมต้องเสียเวลาเป็นอย่างมาก และนอกจากนั้นที่แต่ละระดับยังจะต้องมีการเอารูปแบบจากระดับก่อนหน้ามารวม (Join) กันเพื่อสร้างเป็นรูปแบบที่อาจเกิดขึ้นในระดับถัดไปอีก ดังนั้นจะเห็นได้ว่าเวลาที่เสียไปกับการรอคอยเพื่อให้การทำงานที่ระดับหนึ่งเสร็จสิ้นก่อนแล้วจึงเริ่มการทำงานที่ระดับถัดมามีมากพอควร ส่วนวิธีการแบบดีไอซี (DIC) นั้นการเริ่มหารูปแบบในระดับถัดไปนั้นจะเร็วกว่าแบบวิธีการเอไพออริ (Apriori) แต่ก็ยังต้องอ่านข้อมูลหลายรอบถึงจึงจะได้รูปแบบทั้งหมด จากปัญหาดังกล่าวเราจึงพยายามหาวิธีการแบบใหม่เพื่อที่จะลดเวลาในส่วนนี้ลงไป โดยผลลัพธ์ที่ได้รับยังคงมีความถูกต้องซึ่งเราต้องการให้วิธีการใหม่นี้มีการอ่านฐานข้อมูลเพียง 1 ครั้งเท่านั้น แต่ก็ได้รูปแบบทั้งหมดที่มีค่ามากกว่าค่าสนับสนุนที่เรากำหนด สำหรับโครงสร้างที่จะใช้ในการเก็บรูปแบบนั้นเราพบว่าทั้งวิธีการเอไพโอริ (Apriori) และวิธีการดีไอซี (DIC) นั้นใช้แฮชทรี (Hash Tree) ดังนั้นในวิธีการแบบใหม่เราจึงยังคงใช้โครงสร้างการเก็บรูปแบบเป็นแฮชทรีด้วยเช่นกัน แต่ในการเก็บข้อมูลในส่วนของโหนดใบใดๆ นั้นเราจะเก็บเฉพาะไอเท็มสุดท้ายของรูปแบบไอเท็มเซตเท่านั้น

1.2 วัตถุประสงค์ของงานวิจัย

ในการจัดทำงานงานวิจัยนี้ก็เพื่อวัตถุประสงค์ดังต่อไปนี้

1) เพื่อพัฒนาวิธีการหารูปแบบกลุ่มไอเท็มที่มีการซื้อร่วมกันบ่อยๆ ที่รวดเร็วมากกว่าวิธีการแบบเอไพโอริ (Apriori) และดีไอซี (DIC)

2) เพื่อพัฒนาวิธีการหารูปแบบความสัมพันธ์ของข้อมูลซ่อนอยู่ในข้อมูลจำนวนมากเพื่อนำมาใช้ประโยชน์

3) เพื่อหาวิธีการที่สามารถหารูปแบบได้รวดเร็วโดยการลดจำนวนรอบในการอ่านฐานข้อมูลให้น้อยที่สุด

1.3 ทฤษฎีหรือแนวคิดที่ใช้ในการวิจัย

ในงานวิจัยนี้จำเป็นที่จะต้องนำแนวคิดและทฤษฎีทางด้านต่างดังต่อไปนี้

1) ทฤษฎีทางการทำงานของกระบวนการแอสโซซิเอชั่นรูลย์ (Association Rule)

2) ทฤษฎีและวิธีการทำงานของกระบวนการเอไพโอริ (Apriori)

3) ทฤษฎีและวิธีการทำงานของกระบวนการดีไอซี (DIC: Dynamic Itemset Counting)

1.4 แผนการดำเนินงานวิจัย

1.4.1 ขั้นตอนการดำเนินงานวิจัย

ศึกษาผลงานวิจัยและเอกสารทางวิชาการที่เกี่ยวข้องกับการวิจัยซึ่งได้มีผู้วิจัยไว้แล้ว

1) กำหนดหัวข้อ เป้าหมาย วัตถุประสงค์ และขอบเขตการทำวิทยานิพนธ์

2) ศึกษาทฤษฎีและหลักการที่เกี่ยวข้อง

3) วิเคราะห์และออกแบบวิธีการที่ใหม่

4) พัฒนาโปรแกรมและทดสอบความถูกต้องของวิธีการต่างๆ

5) วิเคราะห์ผล เปรียบเทียบ และสรุปผล

6) จัดทำเอกสารประกอบวิทยานิพนธ์

1.4.2 ระยะเวลาที่ใช้ในแต่ละขั้นตอน

ขั้นตอนการทำงาน	ระยะเวลาที่ใช้ (เดือนที่)											
	1	2	3	4	5	6	7	8	9	10	11	12
ศึกษาผลงานวิจัยและเอกสารทางวิชาการ	■	■	■	■	■							
กำหนดหัวข้อ เป้าหมาย วัตถุประสงค์ และขอบเขตการทำวิทยานิพนธ์				■								
ศึกษาทฤษฎีและหลักการที่เกี่ยวข้อง					■	■						
วิเคราะห์และออกแบบวิธีการที่ใหม่						■	■	■				
พัฒนาโปรแกรมและทดสอบความถูกต้องของวิธีการต่างๆ							■	■	■	■	■	

ขั้นตอนการทำงาน	ระยะเวลาที่ใช้ (เดือนที่)											
	1	2	3	4	5	6	7	8	9	10	11	12
วิเคราะห์ผล เปรียบเทียบ และสรุปผล										■	■	■
จัดทำเอกสารประกอบวิทยานิพนธ์												■

1.5 ขอบเขตของงานวิจัย

สำหรับงานของวิธีการแอสโซซิเอชั่นรูลย์นั้นประกอบด้วยงานหลักๆ อยู่ 2 อย่าง คือการหารูปแบบของกลุ่มไอเท็มที่มักเกิดด้วยกันบ่อย และการนำรูปแบบเหล่านั้นไปสร้างเป็นกฎเพื่อนำไปใช้งานต่อไป แต่สำหรับในงานวิจัยส่วนนี้เราจะทำเฉพาะในส่วนของขั้นตอนการหารูปแบบของกลุ่มไอเท็มเท่านั้น และเครื่องมือที่จะใช้ในการพัฒนางานวิจัยนี้มีดังนี้

1) ภาษาที่ใช้เขียนโปรแกรมต้นแบบทั้ง 3 วิธีการนั้นใช้ภาษา JAVA

2) เครื่องคอมพิวเตอร์ที่ใช้ในการทดสอบผลการทำงานเป็นเครื่อง Pentium III ความเร็ว 750 MHz หน่วยความจำ 128 M

3) ข้อมูลที่ใช้ในการทดสอบเป็นข้อมูลที่ได้จากการสร้างขึ้นโดยโปรแกรมของทาง IBM ซึ่งค้นหาข้อมูลได้จากเว็บไซต์ http://www.almaden.ibm.com/cs/quest/syndata.html โดยตัวสร้างข้อมูลตัวนี้ก็ได้มีการอ้างถึงในทุกงานวิจัย

1.6 ประโยชน์ที่คาดว่าจะได้รับ

เพื่อให้ได้วิธีการหารูปแบบไอเท็มที่มีค่ามากกว่าค่าสนับสนุนแบบใหม่ สำหรับวิธีการแอสโซซิเอชั่นเพื่อการแก้ปัญหา และจุดบกพร่องในการทำงานของวิธีการพื้นฐานที่มีใช้อยู่ก่อนหน้านี้ ซึ่งเราพบว่าจากวิธีการพื้นฐานเหล่านั้นนั้นยังมีข้อบกพร่องอยู่หลายประการด้วยกัน ตัวอย่างเช่น ในเรื่องของจำนวนรอบในการอ่านข้อมูล และจุดเริ่มต้นของการสร้างรูปแบบในระดับถัดไป และด้วยข้อบกพร่องดังกล่าว เราจึงได้พัฒนาวิธีการใหม่ขึ้นเพื่อให้ได้วิธีการในการหารูปแบบความสัมพันธ์ของข้อมูลที่ทำงานได้เร็วมากกว่าวิธีการเดิม โดยที่ผลลัพธ์ที่ได้จากวิธีการนี้ยังคงมีความถูกต้องเช่นเดิม ซึ่งประโยชน์ก็คือ เวลาที่ใช้ในการทำงานที่น้อยลง แต่ก็ยังได้ข้อมูลที่ถูกต้องเช่นเดิม ดังนั้นประโยชน์คือเวลาที่ใช้ในการหารูปแบบไอเท็มเซตที่น่าสนใจจะน้อยลงกว่าวิธีการพื้นฐานแบบเดิม

บทที่ 2
ทฤษฎีและหลักการที่เกี่ยวข้อง

2.1 หลักการทำงานทั่วไป

ในส่วนนี้เราจะอธิบายกระบวนการทำงานของวิธีการแอสโซซิเอชั่นรูลย์ (Association Rule) เกี่ยวกับข้อมูลการซื้อ-ขายสินค้า โดยงานของวิธีการนี้คือเป็นการอธิบายกลุ่มของสินค้าที่มีการซื้อบ่อยๆ เทียบกับกลุ่มสินค้าอื่น ตัวอย่างเช่น เราอาจบอกได้ว่าจากกฎ "80% ของลูกค้าที่ซื้อขนมปังและนมจะซื้อไข่ด้วยในครั้งเดียวกัน" โดยค่าที่ได้จากการทำงานของกระบวนการแอสโซซิเอชั่น (Association) ที่นำไปพิจารณาบ่อยครั้งนั้นมีอยู่สองค่าที่สำคัญคือ support และ confident โดยที่ค่า support ของไอเท็มเซตนั้นคือเปอร์เซ็นต์ของทรานแซกชั่นในฐานข้อมูลที่มีไอเท็มเซตนั้นแสดงดังสมการที่ 2.1 โดยที่เงื่อนไขของกฎความสัมพันธ์ระหว่างไอเท็มเซตคือ เมื่อเกิด A แล้วจะเกิด B เมื่อกำหนดให้ A และ B เป็นไอเท็มเซตที่ไม่ซ้ำกัน ซึ่งสามารถแทนเป็นสัญลักษณ์ได้ดังสมการที่ 2.1 จากสมการที่ 2.1 ค่า confident [1, 3, 4, 9] ของรูปแบบนี้คืออัตราส่วนของค่าสนับสนุนของรูปแบบกับค่าสนับสนุนของรูปแบบส่วนหน้าซึ่งเขียนได้ ดังสมการที่ 2.3

$$A \Rightarrow B \text{ , where itemset } A, B \subset I \text{ and } A \cap B = \phi \qquad (2.1)$$

2.2 ข้อกำหนดของปัญหา

เรากำหนดให้ $I = \{i_1, i_2, \ldots, i_m\}$ เป็นชุดของเอ็ดทริบิว(Attribute) ที่แตกต่างกัน N ตัวบางครั้งเรียกว่าไอเท็ม (Items) โดยที่แต่ละทรานแซกชั่น (Transaction) T ในฐานข้อมูล D มีตัวที่อ้างถึงเพียงตัวเดียวเท่านั้น และภายในทรานแซกชั่น (Transaction) นั้นจะมีชุดของรายการสินค้า (Items) ซึ่งจะเรียกว่า ไอเท็มเซต (Itemset) และไอเท็มเซตที่มี k ตัวจะเรียกว่า เค-ไอเท็มเซต (k-itemset)

Transaction (T) มีการอธิบายเพียงแบบเดียวและประกอบไปด้วยชุดของไอเท็ม (Itemset): $T \subseteq I$ ยกตัวอย่างเช่น I = {A, B, C, D, E} แต่ T = {B, D, E} เป็นต้น ฐานข้อมูล (Databases) D ประกอบด้วยหลายทรานแซกชั่น

2.2.1 ข้อกำหนดทั่วไปของAssociation Rule

ไอเท็มเซต (Itemset) คือชุดของไอเท็มที่ปรากฏขึ้นพร้อมกัน โดยเรากำหนดให้รูปแบบของไอเท็มเซตนั้นแบ่งออกเป็น 2 ส่วน คือส่วนหัว h(g) และส่วนหาง t(g) ตัวอย่างเช่นรูปแบบไอเท็มเซตเป็น {ABC, ABD, ABE} ส่วนหัวของรูปแบบนี้ก็คือ h(g) = {AB} และส่วนคือ t(g) = {C, D, E} ดังนั้นอาจเขียนได้ว่ารูปแบบไอเท็มเซตนั้นมีรูปแบบดังสมการ $h(g) \cup t(g)$ โดยที่ $i \in t(g)$

และ i ในที่นี้หมายถึงไอเท็มหรือสินค้าแต่ละตัวนั้นเองซึ่งจากตัวอย่างที่ยกให้เห็นข้างต้น i ก็คือสินค้าชนิด C, D และ E นั่นเอง

ค่าสนับสนุน(Support) แทนด้วยสัญลักษณ์ $supp(\alpha)$ ของไอเท็มเซต α หรือบางครั้งเรียกว่าค่าความถี่ของรูปแบบ (Frequent) แทนด้วย $fr(\alpha,D)$ คืออัตราส่วนของทรานแซกชั่นที่มีไอเท็มเซตนั้นอยู่กับจำนวนทรานแซกชั่นทั้งหมด ซึ่งแสดงดังสมการที่ 2.2

ค่าความเชื่อมั่นของกฎ (Confident) เป็นค่าที่แสดงว่ากฎที่สร้างขึ้นนั้นมีความเชื่อมั่นว่าจะเกิดขึ้นมากน้อยเพียงใด ซึ่งค่าความเชื่อมั่นนี้สามารถหาได้ดังสมการที่ 2.3

$$fr(\alpha, D) \; or \; supp(\alpha) = \frac{|\{T \in D \,|\, T \; contains \; \alpha\}|}{|D|} \tag{2.2}$$

$$\text{Confident} = supp\,(A \cup B) \,/\, supp\,(A) \tag{2.3}$$

โดยที่ $fr(\alpha,D)$ หมายถึง ค่าสนับสนุนของรูปแบบไอเท็มเซต α ในฐานข้อมูล D ที่มีค่ามากกว่าค่าสนับสนุนต่ำสุด

$supp(\alpha)$ หมายถึง ค่าสนับสนุนของรูปแบบไอเท็มเซต α

$|T|$ หมายถึงจำนวนทรานแซกชั่นในฐานข้อมูลที่มีรูปแบบไอเท็มเซต α

$|D|$ หมายถึงจำนวนทรานแซกชั่นทั้งหมดในฐานข้อมูล

โดยจากสูตรเราจำเป็นที่จะต้องหาว่ารูปแบบ(α) นั้นเกิดขึ้นภายในรายการซื้อสินค้าในฐานข้อมูลที่มีอยู่จำนวนกี่ครั้งแล้วนำค่าที่ได้ไปหารด้วยจำนวนทรานแซกชั่นทั้งหมดในฐานข้อมูล โดยจะทำการพิจารณารูปแบบ (α) ตั้งแต่ระดับ 1-ไอเท็มเซต ไปจนถึง เคไอเท็มเซต (k-itemset)

2.3 รูปแบบโครงสร้างข้อมูล

สำหรับขั้นตอนในการหาค้นหาความรู้นั้นมีอยู่ด้วยกันหลายขั้นตอนแต่ขั้นตอนที่สำคัญขั้นตอนหนึ่งนั้นก็คือการเตรียมข้อมูลให้เหมาะสมกับงานที่เราจะทำ เช่นการปรับเปลี่ยนรูปแบบข้อมูลให้อยู่ในลักษณะที่เหมาะสมและง่ายต่อการนำไปใช้งาน ในการทำงานของวิธีการแอสโซซิเอชั่นนั้นรูปแบบของฐานข้อมูลทรานแซกชั่นที่เหมาะสมมีอยู่ 2 รูปแบบด้วยกันคือ รูปแบบข้อมูลแนวนอน (Horizontal data) และรูปแบบข้อมูลแนวตั้ง (Vertical data)

2.3.1 รูปแบบข้อมูลแนวนอน (Horizontal data)

รูปแบบข้อมูลที่เก็บอยู่ในลักษณะข้อมูลแนวนอน (Horizontal data) นั้นเป็นรูปแบบพื้นฐานทั่วๆไป โดยชุดข้อมูลนั้นประกอบไปด้วยรายการของทรานแซกชั่น โดยที่แต่ละทรานแซกชั่นจะมีค่ารหัสทรานแซกชั่น(Transaction IDentifier) เป็นตัวระบุถึงข้อมูลรายการซื้อสินค้า (Items) การเก็บข้อมูลซึ่งรูปแบบการจัดเก็บข้อมูลแนวนอนนี้จะได้แสดงดังในตารางที่ 2.1 และตารางที่ 2.2 ซึ่งจากทั้ง 2 ตารางนั้นจะมีข้อมูลการซื้อขายสินค้า 4 ครั้ง โดยมีสินค้าที่ขายอยู่ 3 ชนิดคือสินค้า A, สินค้า B และสินค้า C และการแทนค่าเป็น 0 และ 1 ในตารางที่ 2.1 นั้นหมายถึงว่ามีการซื้อสินค้านั้นๆ ในทรานแซกชั่นนั้นหรือไม่ โดยถ้าหากว่าค่าเท่ากับ 0 หมายถึงว่าไม่ได้ซื้อสินค้า แต่ถ้าหากว่าค่าเท่ากับ แสดงว่ามีการซื้อสินค้าชนิดนั้น แต่ไม่สามารถบอกได้ว่าซื้อสินค้าไปกี่ชิ้น

ตารางที่ 2.1 ตัวอย่างการจัดเก็บข้อมูลแนวนอนแบบแทนด้วยรหัส

TID	A	B	C
T1	1	0	1
T2	0	1	0
T3	1	1	1
T4	1	1	0

ตารางที่ 2.2 ตัวอย่างการจัดเก็บข้อมูลแนวนอน

TID	ITEM
T1	AC
T2	B
T3	ABC
T4	AB

2.3.2 รูปแบบข้อมูลแนวตั้ง (Vertical data)

รูปแบบข้อมูลที่เก็บอยู่ในลักษณะข้อมูลแนวตั้ง (Vertical data) บางครั้งเรียกว่าโครงสร้างแบบกลับกัน(Inverted layout) [1] ชุดของข้อมูลสำหรับโครงสร้างแบบนี้ประกอบด้วยรายการของไอเท็ม สินค้า โดยที่แต่ละรายการจะแสดงรายการรหัสทรานแซกชั่นดังที่ได้แสดงไว้ในตารางที่ 2.3 โดยหากมองในลักษณะของข้อมูลแนวนอนก็หมายความว่าแต่ละรายการจะบอกว่าไอเท็มแต่ละตัวนั้นมีปรากฏอยู่ในทรานแซกชั่นใดบ้างนั้นเอง ซึ่งการจัดรูปแบบข้อมูลในลักษณะนี้ใช้ในวิธีการ Eclat [1] ส่วนค่า 0 และ 1 ที่ปรากฏในตารางที่ 2.3 นั้นก็มีความหมายเช่นเดียวกับที่ได้อธิบายไว้แล้วก่อนหน้านี้

ตารางที่ 2.3 ตัวอย่างการเก็บข้อมูลแบบแนวตั้ง

ITEM	T1	T2	T3	T4
A	1	0	1	1
B	0	1	1	0
C	1	0	1	1

2.4 รูปแบบการเข้าถึงรูปแบบไอเท็มเซตในทรี

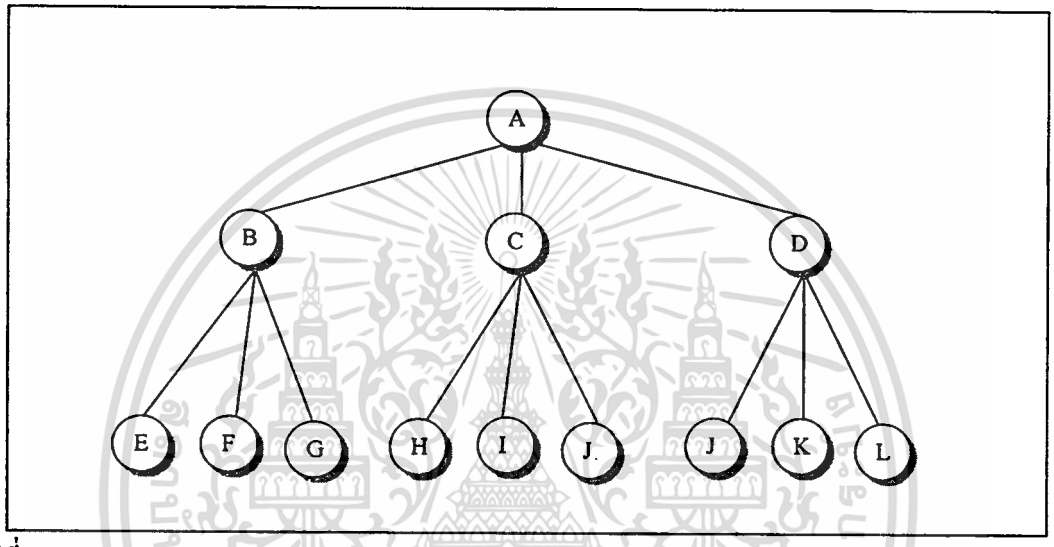
เนื่องจากการจัดเก็บรูปแบบไอเท็มเซตของวิธีการแอสโซซิเอชั่นรูลย์นั้นส่วนมากจะจัดเก็บลงในแฮชทรีดังนั้นเราจึงต้องมีกระบวนการที่จะเข้าถึงรูปแบบแต่ละตัวที่จัดเก็บไว้ โดยรูปแบบในการเข้าถึงนั้นก็มีอยู่ 2 รูปแบบที่นิยมใช้กันคือ การค้นหาในแนวกว้าง (Breadth-First Search) และการค้นหาในแนวลึก (Depth-First Search)

2.4.1 การค้นหาในแนวกว้าง (Breadth-First Search : BFS)

การค้นหาแบบนี้เหมาะสำหรับการเข้าถึงที่ต้องการเข้าถึงทุกๆโหนดในทรี โดยเริ่มต้นที่โหนดรากและเข้าถึงโหนดต่างๆ ไปที่ละระดับ จากตัวอย่างทรีในรูปที่ 2.1 จะได้ลำดับการเข้าถึงโหนดต่างๆดังนี้ {A, B, C, D, E, F, G, H, I, J, K, L, M} และจากการทำงานของวิธีการBFS นี้ได้นำมาใช้งานในการเข้าถึงรูปแบบของวิธีการ Apriori

2.4.2 การค้นหาในแนวลึก (Depth-First Search : DFS)

การค้นหาแบบนี้เราเริ่มต้นค้นหาตั้งแต่ที่โหนดราก วิธีการค้นหาแบบนี้ก็จะค้นหาลงไปในระดับล่างของโหนดนั้นแทนที่จะไปค้นหาโหนดถัดไปในระดับเดียวกับโหนดปัจจุบัน ซึ่งจากตัวอย่างทรีในรูปที่ 2.1 เมื่อทำตามวิธีการ DFS แล้วจะได้ลำดับของโหนดที่เข้าถึงดังนี้ {A, B, E, F, G, C, H, I , J, D, K, L, M}



รูปที่ 2.1 ตัวอย่างแฮชทรี

2.5 กระบวนการทำงานของวิธีการApriori

วิธีการ Apriori นั้นใช้ลักษณะการนับความสัมพันธ์ของชุดไอเท็ม ที่เกิดขึ้นในข้อมูลจากบนลงล่าง โดยในแต่ละรอบของการทำงานก็จะนำเอาชุดของไอเท็ม ที่พบในระดับก่อนหน้ามาจอยกันเพื่อสร้างเป็นรูปแบบที่อาจเกิดขึ้นได้ในระดับถัดไปแทนด้วยสัญลักษณ์ C_k โดยวิธี Apriori จะมีการทดสอบความถี่ของรูปแบบในระดับ k-1 ไอเท็มโดยการเลือกเฉพาะรูปแบบที่มีค่ามากกว่าค่าสนับสนุนต่ำสุดที่กำหนดไว้ มาจอยกันเพื่อสร้างเป็นรูปในระดับ k-ไอเท็ม ซึ่งขั้นตอนนี้จะช่วยลดรูปแบบตัวแทนในระดับถัดไปที่ไม่มีความจำเป็นออกไปจำนวนมาก จากนั้นก็จะนำเอารูปแบบที่ได้ไปจัดเก็บอยู่ใน hash tree เพื่อความรวดเร็วในการนับค่าสนับสนุน (support) โดยที่ชุดของไอเท็มทั้งหมดก็จะมีการจัดเก็บที่โหนดล่างสุดของต้นไม้ (Leaf node) ส่วนกระบวนการเพิ่ม (insertion) จะเริ่มที่โหนดราก (Root node) และลงไปสู่ไอเท็มที่ต้องการ เมื่อพบแล้วก็ใส่ค่ารูปแบบนั้นที่โหนดปลายหรือโหนดใบ (Leaf node) [1, 2, 4, 6]

วิธีการนี้จะเห็นได้ว่าเรานั้นจำเป็นที่จำต้องเข้าไปอ่านข้อมูลหลายรอบเพื่อนับค่าสนับสนุนของรูปแบบความสัมพันธ์ของไอเท็มที่ระดับต่างๆ ดังนั้นจึงเห็นได้ว่าเราต้องเสียเวลามากในการอ่านข้อมูลทรานแซกชั่น โดยมีจำนวนรอบของการอ่านเท่ากับค่ามากที่สุดในตัวแทนของไอเท็มเซต

สมมุติให้รูปแบบที่เกิดขึ้นเป็นแบบ C_1 ={A, B, C, D, E, F} เมื่อ k = 1 ไอเท็มและจะเลือกไปใช้สำหรับการพิจารณาในขั้นตอนถัดไปโดยดูจากค่าสนับสนุนต่ำสุด (minimum support) ที่เราได้กำหนดไว้ สมมุติในกรณีนี้เราได้ L_1 ={A, B, D, F} เราก็จะใช้ค่าที่ได้จากส่วนนี้ไปใช้เพื่อนำไปใช้ในการหารูปแบบที่จะใช้พิจารณาระดับที่สูงขึ้นไป โดยรูปแบบที่เกิดขึ้นนั้นจะได้มาจากการรวม (Join) ภายในของรูปแบบในระดับก่อนหน้า ซึ่งในที่นี้ที่ระดับ k = 2 items ก็จะได้มาจากการรวม (join) กันของไอเท็ม(item)ในระดับ $L_1 \times L_1$ ซึ่งก็จะได้เป็น C_2={AB, AC, AD, AF, BD, BF, DF } และเมื่อได้รูปแบบเหล่านี้มาแล้วก็จะเข้าไปนับค่าสนับสนุน (support) ของรูปแบบต่างๆ ที่ปรากฏขึ้นใน C_2 และเลือกรูปแบบที่มีค่าสนับสนุน(support) สูงกว่าค่าสนับสนุนต่ำสุด (minimum support) ที่ตั้งไว้เหมือนกับในขั้นตอนของการหา L1 และในขั้นตอนต่อๆไปก็มีกระบวนการทำงานเช่นนี้เช่นกัน หรืออาจเขียนในลักษณะสัญลักษณ์แทนได้โดยเป็นไปตามขั้นตอนดังรูปที่ 2.2 ซึ่งเป็นวิธีการแบบ Apriori และตัวอย่างขั้นตอนการทำงานของวิธีการ Apriori จากชุดข้อมูลที่กำหนดขึ้นซึ่งแสดงไว้ในรูปที่ 2.3

Pseudo code Apriori

L_1 = {frequent 1-Itemset}

For ($k = 2$, $L_{k-1} \neq \phi$: k++)

C_k = Set of New Candidates

For all transaction t $\in D$

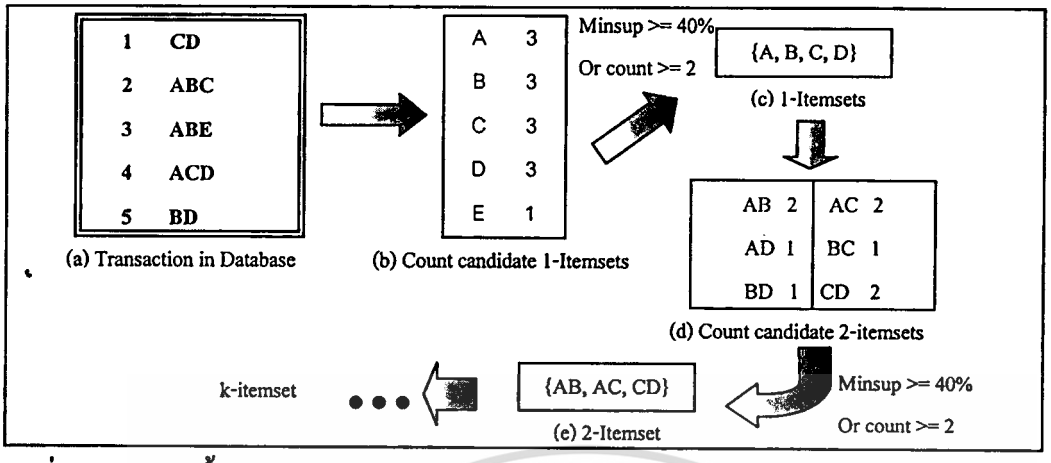
For all k-subset s of t

If ($s \in C_k$) s.count ++

$Lk = \{c \in C_k \mid c.\ count \geq minimum\ support\}$

Set of all frequent item set = $U_k L_k$

รูปที่ 2.2 ขั้นตอนการทำงานของวิธี Apriori



รูปที่ 2.3 ตัวอย่างขั้นตอนการทำงานของวิธี Apriori

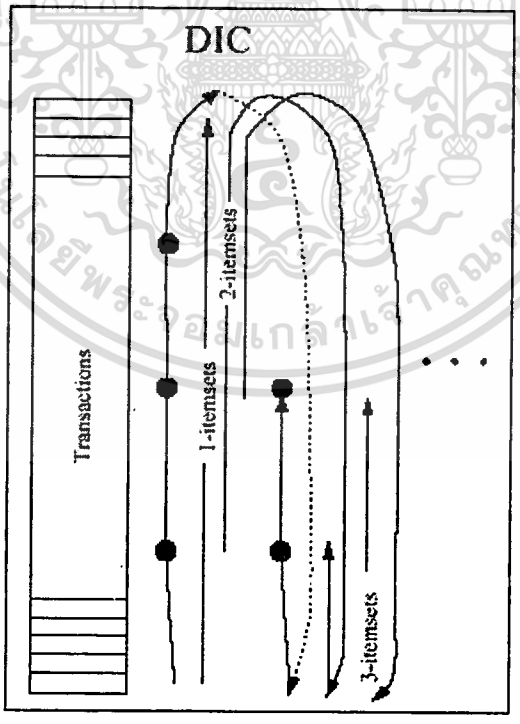
2.6 กระบวนการ DIC (Dynamic Itemset Counting)

จากกระบวนการหารูปแบบไอเท็มเซตสูงสุดนั้นเราพบว่าปัญหาของการวิเคราะห์ข้อมูลการซื้อขายนั้นมีอยู่หลายอย่างในวิธีการ Apriori ซึ่งปัญหาอย่างแรกคือ การแสดงวิธีการค้นหารูปแบบไอเท็มเซตสูงสุด ซึ่งใช้การเข้าไปอ่านข้อมูลจากฐานข้อมูลหลายครั้ง ดังนั้นจึงได้มีการพัฒนาวิธีการ DIC เพื่อที่จะได้เข้าไปอ่านฐานข้อมูลเพียงแค่ 1 – 2 ครั้งเท่านั้นก็ได้รูปแบบไอเท็มเซตทั้งหมด โดยที่เราก็จะใช้ตัวอย่าง itemset เพียง 2 –3 ตัว เพื่อเป็นตัวอย่างการทำงานของกระบวนการ DIC นี้ โดยที่กระบวนการ DIC พยายามหาวิธีการของการจัดวางไอเท็มใหม่ซึ่งสามารถที่จะพัฒนาประสิทธิภาพในระดับล่างของกระบวนการได้

กระบวนการได้มาของรูปแบบไอเท็มเซตสูงสุดหรือรูปแบบไอเท็มเซตที่มีจำนวนไอเท็มมากที่สุด วิธีการหนึ่งที่เป็นที่รู้จักมากคือวิธีการ Apriori ซึ่งเหมือนกับวิธีการอื่นๆในการหารูปแบบไอเท็มเซตสูงสุด ขั้นตอนแรกของมันคือหาค่าที่ระดับ 1 ไอเท็มเซต และเลือกเอาแต่ค่าที่มากกว่าค่าที่กำหนดมาเป็น 1–รูปแบบไอเท็มเซตสูงสุด จากนั้นก็รวมค่าเหล่านั้นเพื่อสร้างเป็นรูปแบบไอเท็มเซตในระดับที่ 2 จากนั้นก็นับค่ารูปแบบเหล่านั้นแล้วเลือกเอาเฉพาะค่าที่มีค่ามากกว่าค่าที่กำหนด แล้วเอาไปสร้างเป็นรูปแบบที่สนใจในระดับที่ 3 ต่อไป และเป็นอย่างนี้ไปเรื่อยจนกว่าจะไม่มีรูปแบบใหม่เกิดขึ้นจึงหยุดดังนั้นวิธีการเช่นนี้ทำให้ต้องมีการอ่านข้อมูลหลายรอบเท่ากับจำนวนของค่ามากที่สุดในตัวแทนของไอเท็มเซต เพื่อที่จะหาค่าสนับสนุนของแต่ละรูปแบบในรอบที่ k ซึ่งขั้นตอนการทำงานของวิธีการนี้แสดงอยู่ในรูปที่ 2.2 และปัญหาที่เป็นไปได้ของวิธีการ Apriori มี 2 อย่างคือประสิทธิภาพจากจำนวนรอบที่ต้องอ่านข้อมูล และประสิทธิภาพในการอ่านข้อมูลแต่ละรอบ

วิธีการ Dynamic Itemset Counting (DIC) เป็นวิธีการที่ได้พัฒนาขึ้นและจัดได้ว่าเป็นวิธีการนี้จะลดจำนวนรอบในการทำงานกับข้อมูลระหว่างที่มีการเก็บค่าจำนวนไอเท็มเซต ซึ่งนับในรอบใดๆของความสัมพันธ์ที่มีค่าเท่ากันเมื่อเปรียบเทียบกระบวนการที่มีการสุ่มสร้างรูปแบบไอเท็มเซต โดยความรู้สึกเหมือนกับว่าด้านหลัง DIC มีการทำงานรถไฟที่กำลังวิ่งไปบนข้อมูลด้วยการหยุดที่ช่วงทรานแซกชั่น M ตัว (หมายเหตุ M เป็นค่าตัวแปรในการทดลองของเราโดยเราพยายามหาค่าตั้งแต่ช่วง 100 ถึง 10,000) และเมื่อไปถึงส่วนสุดท้ายของไฟล์มันก็จะมีการทำงานหนึ่งรอบกับข้อมูลและมันก็จะเริ่มต้นที่ส่วนเริ่มต้นของรอบถัดไป โดยเราก็สมมุติให้ผู้โดยสารบนรถไฟนั้นก็คือ ไอเท็มเซต เมื่อไอเท็มเซต อยู่ในรถไฟแล้ว เราก็นับค่าเพิ่มเมื่อมันเกิดขึ้นในทรานแซกชั่นที่อ่าน

ถ้าเราพิจารณา Apriori ในเชิงอุปมาอุปไมยทุกไอเท็มเซต ต้องได้มาในตอนเริ่มต้นของการเข้าไปอ่านข้อมูลและเหลือออกมาในตอนจบ โดย 1ไอเท็มเซต นั้นได้มาจากตอนรอบแรกและ 2 ไอเท็มเซต ได้ออกมาจากการอ่านในรอบที่ 2 ส่วนในวิธีการ DIC เราได้เพิ่มความยืดหยุ่นสำหรับการยอมให้ไอเท็มเซต ยังอยู่เมื่อมีการหยุดเช่นเดียวกับการได้ผลลัทธ์ออกมาที่ที่หยุดในที่เดียวกันเมื่อเวลาถัดมาที่รถไฟผ่านมาอีกครั้ง ดังนั้นทุกทรานแซกชั่นในไฟล์ได้เห็นไอเท็มเซตนั้น ซึ่งหมายความว่าเราสามารถที่จะนับค่าของไอเท็มเซตนั้นได้ตลอดโดยไม่ต้องรอให้จบรอบหนึ่งก่อน



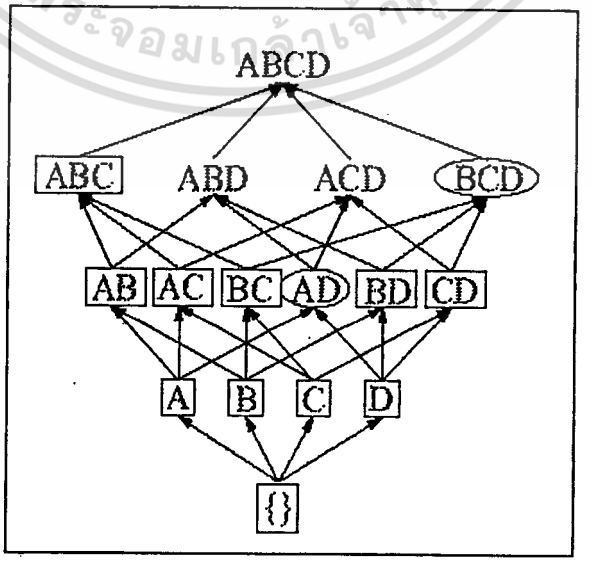
รูปที่ 2.4 รอบในการอ่านข้อมูลของวิธีการ DIC

ยกตัวอย่างจากรูปที่ 2.4 เมื่อเรากำหนดให้ทำการค้นหาข้อมูลจาก 40,000 ทรานแซกชั่น และใน M=10,000 เราจะนับทุก 1 ไอเท็มเซต ในตอนแรก 40,000 ทรานแซกชั่น เราจะอ่าน อย่างไรก็ตามเราจะเริ่มต้นนับ 2 ไอเท็มเซต หลังจาก 10,000 ทรานแซกชั่นแรกได้อ่านไปแล้ว และเราจะอ่าน 3 ไอเท็มเซต หลังจากอ่านไปแล้ว 20,000 ทรานแซกชั่น สำหรับในขณะนี้เราสมมุติว่าไม่มี 4 ไอเท็มเซต ที่เราต้องการที่จะนับ เมื่อเราไปถึงจุดสิ้นสุดของไฟล์เราก็จะหยุดการนับ 1ไอเท็มเซต และกลับไปเริ่มต้นใหม่อ่านไฟล์ใหม่เพื่อนับ 2 และ 3 ไอเท็มเซตและหลังจากจบ 10000 ทรานแซกชั่นแรกเราก็หยุดการนับค่า 2 ไอเท็มเซต และหลังจากจบการนับ 20,000 ทรานแซกชั่นเราก็จบการนับ 3ไอเท็มเซต เมื่อรวมแล้วเราจะได้ว่าเราอ่านข้อมูลเพียงแค่ 1.5 รอบแทนที่จะอ่านข้อมูลถึง 3 รอบ

2.6.1 การนับจำนวนรูปแบบไอเท็มเซตสูงสุด

ไอเท็มเซตจากโครงข่ายขนาดใหญ่ซึ่งมีค่าเป็นเซตว่างที่ส่วนล่างและเป็นเซตของทุกๆ ไอเท็มเซตที่ด้านบน สำหรับบางรูปแบบที่มีค่ามากค่าสนับสนุนที่กำหนดไว้(ระบุโดยการแสดงด้วยรูปกรอบสี่เหลี่ยม) และที่เหลือจะเป็นรูปแบบที่มีค่าน้อยกว่า โดยจากตัวอย่างประกอบด้วย เซตว่าง, A, B, C, D , AB, AC, BC, BD, CD, ABC ต่างก็มีค่ามากกว่าค่าที่กำหนดไว้

เพื่อแสดงไอเท็มเซตซึ่งมีขนาดใหญ่เราสามารถที่จะนับพวกมันได้ เมื่อเราต้องการที่จะรู้จำนวน อย่างไรก็ตามมันเป็นไปไม่ได้ที่จะนับทุกๆไอเท็มเซต ที่มีค่าน้อยกว่า แต่การนับค่าที่น้อยกว่าค่าที่กำหนดจำนวนหนึ่ง(ไอเท็มเซต ซึ่งไม่ได้รวมอยู่ใน ไอเท็มเซตที่มีค่าน้อยตัวอื่นๆ) และถ้าไอเท็มเซตนั้นมีค่าน้อยแล้วเซตที่สร้างขึ้นมาจากตัวมันก็ย่อมที่จะมีค่าน้อยไปด้วยตัวไอเท็มเซตที่มีค่าน้อยนั้นแสดงด้วยตัววงกลม



รูปที่ 2.5 โครงสร้างทรีของรูปแบบไอเท็มสำหรับวิธีการ DIC เมื่อจบการทำงาน

กระบวนการซึ่งนับจำนวนค่าไอเท็มเซตที่มากกว่าต้องค้นพบและนับทุกไอเท็มเซตที่มีค่ามากกว่าค่าที่กำหนดรวมทั้งนับค่าที่น้อยกว่าด้วย วิธีการ DIC นั้นจะมีการอธิบายแต่ละไอเท็มเซตโดยการใช้ตัวระบุที่แตกต่างกันดังนี้

1) Solid Box แทนด้วย SS หรือสัญลักษณ์  เป็นค่าที่ระบุอย่างชัดเจนว่าเป็นรูปแบบที่มากกว่าค่าที่กำหนดโดยแสดงไว้เมื่อจบการนับที่ระดับต่างๆ

2) Solid Circle แทนด้วย SC สัญลักษณ์  เป็นค่าระบุอย่างชัดเจนว่าเป็นรูปแบบที่มีค่าน้อยกว่าค่าที่กำหนดเมื่อจบการนับที่ระดับนั้นๆ

3) Dashed Box แทนด้วย DS สัญลักษณ์  เป็นการคาดว่ารูปแบบนั้นน่าจะมีค่ามากกว่าค่าที่กำหนดแสดงว่าในขณะนั้นกำลังมีการนับที่ระดับนั้นๆ อยู่

4) Dashed Circle แทนด้วย DC สัญลักษณ์  เป็นค่าเป็นการคาดว่ารูปแบบนั้นน่าจะมีค่ามากกว่าค่าที่กำหนดแสดงว่าในขณะนั้นกำลังมีการนับที่ระดับนั้นๆ อยู่

2.6.2 ขั้นตอนการทำงานของกระบวนการ DIC

ขั้นตอนการทำงานของ DIC

1) ที่เซตว่างให้ระบุด้วยกรอบสี่เหลี่ยมและ 1 ไอเท็มเซต ให้ระบุด้วยรูปวงกลมเส้นประส่วนตัวอื่นๆ ก็ไม่มีการระบุใด

2) อ่านทรานแซกชั่นมาทั้งหมด M ตัวโดยจากการทดลองเราจะอยู่ในช่วงค่า 100 ถึง 10,000 และสำหรับแต่ละทรานแซกชั่น

3) ถ้าหากว่าสถานะของรูปแบบไอเท็มเซตเป็นเส้นประวงกลมและเมื่อนับไปจนมากกว่าค่าที่กำหนดก็เปลี่ยนเป็นรูปแบบเส้นประสี่เหลี่ยม และถ้ารูปแบบที่ใช้สร้างเป็นรูปแบบไอเท็มเซตในระดับถัดไป (Superset) นั้นมีรูปแบบเป็นรูปสี่เหลี่ยมทึบหรือเส้นประก็ให้เพิ่มการนับสำหรับรูปแบบนั้นและหาระบุรูปแบบนั้นเป็นรูปวงกลมเส้นประ

4) ถ้ารูปแบบที่เป็นเส้นประนั้นมีการนับจนครบแล้วก็ให้เปลี่ยนเป็นเส้นทึบและหยุดนับค่าสำหรับรูปแบบนั้น

5) ถ้าเราอยู่ที่ตำแหน่งสุดท้ายของทรานแซกชั่นก็ให้กลับไปเริ่มต้นใหม่

6) ถ้ายังมีรูปแบบที่เป็นเส้นประก็กลับไปเริ่มต้นใหม่

ในรูปแบบนี้ DIC เริ่มต้นนับโดย 1ไอเท็มเซตและมีการเพิ่มการนับเป็น 2, 3, 4, ..., k ไอเท็มเซต หลังจากที่อ่านผ่านไป 2 –3 รอบ โดยตามแนวคิดนี่เราต้องการให้ค่า M มีค่าน้อยที่สุดเท่าที่จะเป็นไปได้ดังนั้นเราสามารถเริ่มนับไอเท็มเซต ได้เร็วที่สุดในขั้นตอนที่ 3 อย่างไรก็ตาม ขั้นตอนที่ 3 และ 4 นั้นก่อให้เกิดข้อมูลส่วนหัวที่ใช้บอกรายละเอียดของไอเท็มเซตแต่ละตัวจำนวนมาก ตัวอย่างข้อมูลที่ต้องเก็บเช่น ตำแหน่งทรานแซกชั่นเริ่มต้นนับเพิ่มค่าของไอเท็มเซต แต่ละตัว เป็นต้น ดังนั้นเราจึงไม่สามารถที่จะลด M ให้น้อยกว่า 10,000 ได้

2.6.3 โครงสร้างของการเก็บข้อมูลรูปแบบไอเท็มเซต (The Data Structure of itemset)

โครงสร้างข้อมูลที่ใช้ในวิธีการนี้แท้จริงแล้วคล้ายคลึงกับแฮชทรีที่ใช้ในเทคนิค Apriori แต่ละโหนดจะเก็บไอเท็มสุดท้ายของรูปแบบไอเท็ม และแสดงค่าตัวนับ,ตัวระบุสถานะ ค่าเริ่มต้นนับรูปแบบนี้ที่ใด สถานะของมันและกิ่งถ้ามันมีความสัมพันธ์กับโหนดอื่นๆตัวเซตว่างนั้นจะอยู่ที่ส่วนรากของทรี รูปแบบที่ระดับ 1ไอเท็มเซต ทุกตัวจะก็ติดต่อกับโหนดราก และกิ่งของพวกมันการจะมีการระบุโดยค่าไอเท็มที่ปรากฏรูปแบบไอเท็มตัวอื่นๆต่างมีความสัมพันธ์กับตัวไอเท็มที่อยู่ก่อนหน้าทุกตัวยกเว้นตัวสุดท้าย โดยพวกมันจะระบุโดยใช้ค่าไอเท็มตัวสุดท้าย เมื่อเกิดทรานแซกชั่น ABC ดังนั้น A, AB, ABC, AC, B , BC และ C จะเพิ่มค่าขึ้น ตามลำดับ ซึ่งแสดงไว้ดังในรูปที่ 2.5

2.7 สรุปหลักการทำงานที่เกี่ยวข้อง

จากหลักการทำงานที่เกี่ยวข้องดังที่ได้กล่าวมาแล้วจะเห็นได้ว่าวิธีการหากฎของแอสโซซิเอชั่นรูลย์ก็ยังคงมีข้อบกพร่องและได้มีการพัฒนาวิธีการเพื่อแก้ไขข้อบกพร่องเพื่อให้การทำงานดีขึ้น โดยข้อบกพร่องของวิธีการก่อนหน้าอย่างในวิธีการ Apriori และ DIC คือการที่ต้องเข้าไปอ่านข้อมูลหลายๆ รอบเพื่อให้ได้รูปแบบไอเท็มที่มีค่ามากกว่าค่าสนับสนุนที่ผู้วิเคราะห์กำหนด สำหรับในวิธีการใหม่ที่พัฒนาขึ้นนี้จะอ่านข้อมูลทรานแซกชั่นทั้งหมดเพียงแค่ 1 รอบเท่านั้น โดยเราพบว่าถ้าสามารถที่จะเริ่มต้นสร้างรูปแบบในระดับถัดไปได้เร็วเท่าใดจำนวนรอบการอ่านก็จะน้อยลงเวลาที่จะใช้ในการหารูปแบบก็ย่อมที่จะน้อยลงด้วย ซึ่งวิธีการ Apriori นั้นการเริ่มสร้างรูปแบบในระดับถัดไปนั้นจะสร้างหลังจากอ่านจบแต่ละรอบ และวิธีการ DIC จึงได้ปรับปรุงการสร้างรูปแบบให้มีการสร้างรูปแบบใหม่ทุกๆ การอ่านทรานแซกชั่นจำนวนหนึ่ง และสำหรับวิธีการใหม่ที่พัฒนานั้นจะมีการสร้างรูปแบบในระดับถัดไปเมื่อจำนวนไอเท็มในทรานแซกชั่นเพิ่มขึ้น ซึ่งก็จะทำให้มีการสร้างรูปแบบในระดับถัดไปได้เร็วขึ้น ส่วนกระบวนการทำงานของวิธีการใหม่นั้นมีการทำงานอย่างไร จะได้อธิบายไว้ในบทที่ 3

บทที่ 3
ทฤษฎีและหลักการทำงานของวิธีการใหม่

เนื่องจากวิธีการทำงานแบบพื้นฐานของวิธีการแอสโซซิเอชั่นรูลย์นั้นยังมีข้อบกพร่องอยู่หลายประการที่อาจทำให้การทำงานในส่วนของการหารูปแบบไอเท็มที่มีค่ามากกว่าค่าสนับสนุนนั้นยังช้าอยู่ดังนั้นเราจึงพยายามหารูปแบบวิธีการทำงานแบบใหม่เพื่อที่จะทำให้การทำงานในส่วนดังกล่าวนั้นทำงานได้ดีและรวดเร็วขึ้น โดยจุดบกพร่องที่เรามองเห็นของวิธีพื้นฐานที่มีมาก่อนหน้านี้ก็เช่นในเรื่องของจำนวนรอบที่ต้องเข้าไปอ่านข้อมูลเพื่อให้ได้รูปแบบทั้งหมดที่มีค่ามากกว่าค่าสนับสนุนที่กำหนดให้ และจุดที่จะเริ่มสร้างรูปแบบในระดับถัดไป

3.1 กระบวนการทำงานแบบใหม่

วิธีการใหม่นี้พัฒนาขึ้นเพื่อลดความบกพร่องในการทำงานของวิธีการพื้นฐานในการทำงานของแอสโซซิเอชั่นรูลย์ ซึ่งแอสโซซิเอชั่นรูลย์นั้นนิยมใช้เพื่อหารูปแบบความสัมพันธ์ของงานในด้านต่างๆ เช่นงานทางด้านการตลาด โดยเราต้องการที่จะได้วิธีการใหม่ที่มีความรวดเร็วมากขึ้นกว่าวิธีการพื้นฐานเดิม และเนื่องจากเรามองเห็นจุดบกพร่องของวิธีการแบบ Apriori ที่ต้องมีการเข้าไปอ่านข้อมูลจากฐานข้อมูลหลายครั้ง โดยแต่ละครั้งที่อ่านนั้นก็ย่อมต้องเสียเวลามาก และสำหรับวิธีการ DIC นั้นเป็นวิธีการที่ช่วยลดจำนวนครั้งในการอ่านข้อมูล แต่ก็มีข้อเสียในเรื่องของการทำงานกับข้อมูลที่มีความหลากหลายมาก ดังนั้นเราจึงพยายามหาวิธีการในการที่จะลดการกระจายของข้อมูลลง แต่ว่ายังคงมีจำนวนรอบการอ่านข้อมูลที่น้อยอยู่เหมือนกับวิธีการ DIC

วิธีการแบบใหม่นั้นจะมีการอ่านฐานข้อมูลครั้งแรกเพื่อเป็นการเตรียมข้อมูลโดยการจัดเรียงทรานแซกชั่นใหม่โดยที่จะเรียงทรานแซกชั่นตามลำดับของจำนวนไอเท็มในทรานแซกชั่นนั้นแล้วจึงจะไปสู่ขั้นตอนที่ 2 คือการหารูปแบบของไอเท็มที่ระดับถัดไป โดยโครงสร้างของตัวที่เป็นตัวเก็บรูปแบบไอเท็มเซตที่ใช้นั้นจะมีรูปแบบเป็นแฮชทรี และกำหนดให้มีการเข้าถึงรูปแบบไอเท็มต่างในทรีเป็นแบบดีเอฟเอส (DFS)

ในกระบวนการแบบใหม่นี้เราได้พยายามที่จะทำให้มีการเริ่มสร้างรูปแบบไอเท็มที่จะนำมานับค่าสนับสนุนให้เร็วขึ้นกว่าวิธีการพื้นฐานแบบเดิม โดยเราพิจารณาจากความจริงที่ว่าทรานแซกชั่นที่มีจำนวนไอเท็ม n ตัวนั้นย่อมที่จะเกิดรูปแบบไอเท็มเซตที่ระดับ 1 ไอเท็มเซต ถึง n ไอเท็มเซตเท่านั้น และไม่สามารถที่จะเกิดรูปแบบในระดับ n+1 ไอเท็มได้เลย ดังนั้นเมื่อเราทำการเรียงทรานแซกชั่นตามจำนวนไอเท็มในทรานแซกชั่นจากจำนวนไอเท็มน้อยไปจำนวนไอเท็มมากแล้ว ดังนั้นเราจึงสามารถที่จะอ่านข้อมูลในทรานแซกชั่นทั้งหมดเพียงครั้งเดียวก็จะได้รูปแบบของไอเท็มเซตทั้งหมดที่มีค่ามากกว่าค่าสนับสนุนที่ผู้วิเคราะห์กำหนดไว้

3.2 ขั้นตอนการทำงานของกระบวนการใหม่

กระบวนการทำงานแบบใหม่นี้พัฒนาขึ้นมาเพื่อการแก้ปัญหาต่างดังที่ได้กล่าวมาแล้วข้างต้นโดยวิธีการทำงานของกระบวนการใหม่นี้จะประกอบด้วย 2 ส่วนหลักคือ

1) การจัดเตรียมข้อมูลก่อนการประมวลผล

2) การประมวลผล

3.2.1 การจัดเตรียมข้อมูลก่อนการประมวลผล

ในขั้นตอนของการจัดเตรียมข้อมูลก่อนการประมวลผลนั้นเราก็จะนำเอาข้อมูลทรานแซกชั่นทั้งหมดมาทำการจัดเรียงลำดับทรานแซกชั่นเหล่านั้นใหม่โดยที่เราจะทำการจัดเรียงทรานแซกชั่นทั้งหมดตามจำนวนไอเท็มในทรานแซกชั่นนั้น และการเรียงลำดับก็จะจัดเรียงจากจำนวนไอเท็มน้อยไปสู่จำนวนไอเท็มมาก เมื่อได้แฟ้มข้อมูลที่จัดเรียงเรียบร้อยแล้วก็จะไปทำงานในขั้นตอนของการประมวลผลต่อไป

3.2.2 การประมวลผล

ในขั้นตอนนี้ถือว่าเป็นขั้นตอนที่สำคัญมากเพราะจะเป็นขั้นตอนที่เราจะหารูปแบบไอเท็มเซตที่มีค่ามากกว่าค่าสนับสนุนต่ำสุดที่เรากำหนด โดยในขั้นตอนของการประมวลผลนั้นก็จะมีลำดับการทำงานดังนี้

1) สร้างทรีของรูปแบบไอเท็มเซต โดยกำหนดให้โหนดรากเป็นเซตว่าง และระดับที่ถัดมาเป็นรูปแบบในระดับ 1 ไอเท็มเซตซึ่งก็คือค่าของสินค้าหรือบริการทุกตัวที่เรามีนั้นเอง

2) กำหนดค่าเริ่มต้นของระดับไอเท็มเซตที่จะนับค่าสนับสนุนเท่ากับ 1 (k =1)

3) อ่านทรานแซกชั่นทุกตัวจนครบ

3.1) ถ้าค่า k น้อยกว่าจำนวนไอเท็มในทรานแซกชั่น

3.1.1) เพิ่มค่า k ขึ้นอีก 1 (k = k+1)

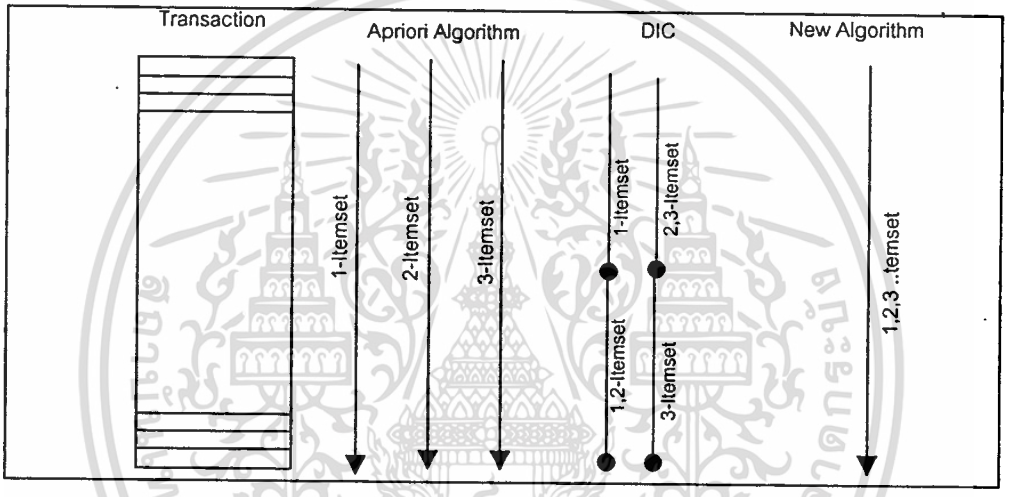
3.1.2) สร้างรูปแบบไอเท็มเซตที่ระดับถัดไป โดยพิจารณาสร้างรูปแบบจากไอเท็มเซตในระดับก่อนหน้า

3.2) นับเพิ่มค่ารูปแบบทั้งหมดที่สามารถเกิดขึ้นในทรานแซกชั่นนั้นเช่นทรานแซกชั่นคือ{ABC} รูปแบบที่จะเกิดขึ้นได้คือ {A, AB, ABC, AC, B, BC, C} โดยวิธีการเข้าไปนับเพิ่มค่าของรูปแบบนั้นแสดงดังรูปที่ 3.2 และหากมีรูปแบบใดที่ไม่พบก็จะสร้างรูปแบบไอเท็มนั้นเพิ่มเติมลงในทรี

4) หารูปแบบทั้งหมดที่มีค่ามากกว่าค่าสนับสนุนที่กำหนดไว้ และแสดงรูปแบบไอเท็มที่ได้มาเหล่านั้นออกมาแยกตามระดับไอเท็มเซต

5) จบการทำงาน

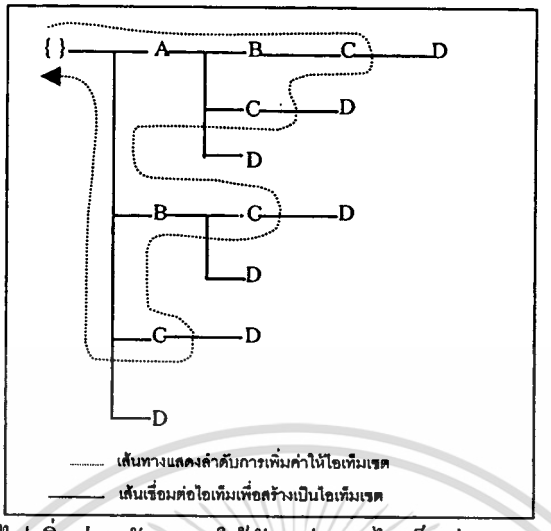
จากขั้นตอนของกระบวนการแบบใหม่นี้เราจะเห็นได้ว่าเราสามารถที่จะเริ่มต้นให้มีการนับรูปแบบที่ระดับต่างได้เร็วขึ้นกว่ากระบวนการ Apriori และ DIC เพราะในวิธีการ Apriori นั้นกว่าจะสร้างรูปแบบในระดับถัดไปได้ก็ต้องอ่านข้อมูลจนจบฐานข้อมูลก่อน ส่วนวิธีการ DIC นั้นกว่าจะสร้างรูปแบบในระดับถัดไปได้ก็ต้องอ่านข้อมูลไปเป็นช่วงหนึ่งตามที่ผู้ใช้กำหนดก่อน แต่สำหรับวิธีการใหม่นี้ก็จะมีการอ่านข้อมูลมาก่อนจำนวนหนึ่งเพื่อสร้างเป็นทรีของรูปแบบจากนั้นก็จะนับรูปแบบไอเท็มเซตทุกระดับที่ปรากฏในทรานแซกชั่นซึ่งทำให้การนับรูปแบบรวดเร็วขึ้นกว่าเดิม เพราะหากเราเริ่มการนับรูปแบบที่ระดับสูงขึ้นได้เร็วเท่าไหร่เวลาในการทำงานก็น่าจะน้อยลงเท่านั้นซึ่งพอที่จะแสดงจำนวนรอบการเข้าถึงของแต่ละวิธีการได้ดังรูปที่ 3.1



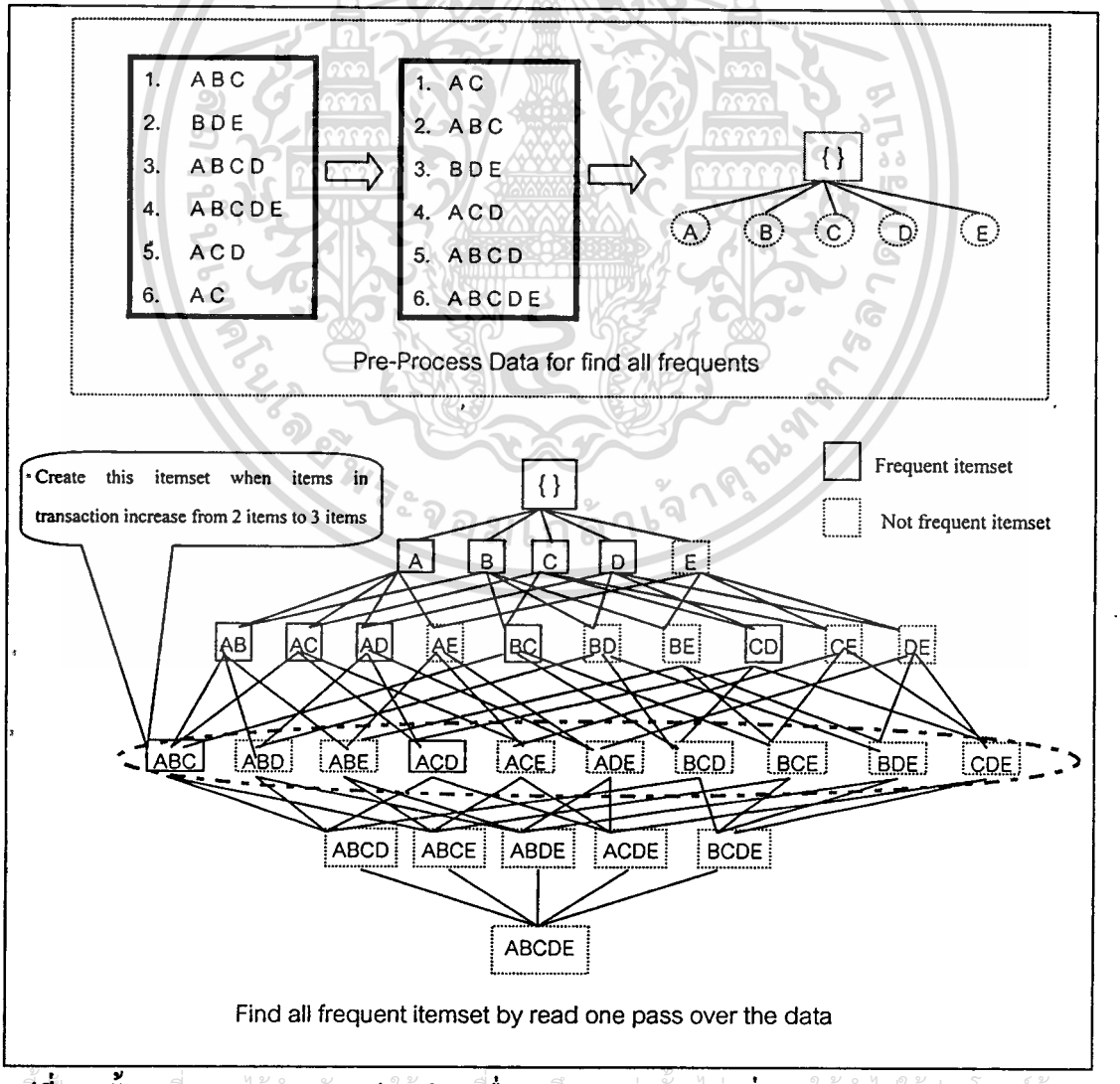
รูปที่ 3.1 จำนวนรอบในการอ่านข้อมูลของวิธีการ Apriori, DIC และ New Approach

3.3 ตัวอย่างขั้นตอนการทำงานของวิธีการใหม่

ในรูปที่ 3.2 ก็จะแสดงให้เห็นถึงวิธีการเข้าไปนับรูปแบบเพิ่มสำหรับวิธีการใหม่นี้ซึ่งใช้การเข้าในแบบ DFS โดยสมมุติให้ทรานแซกชั่นเข้าเป็น A B C ดังนั้นรูปแบบที่จะได้ทั้งหมดคือ {A, AB, ABC, AC, B, BC, C} ซึ่งจะเห็นได้ว่าลำดับของการเข้าไปเพิ่มค่าสนับสนุนให้กับแต่ละรูปแบบไอเท็มเป็นดังรูปที่ 3.2 ส่วนการเข้าถึงในขณะที่ทำการสร้างรูปแบบในระดับถัดไปนั้นจะใช้การเข้าถึงแบบ BFS และนอกจากนั้นจากรูปที่ 3.3 ซึ่งแสดงให้เห็นถึงวิธีการทำงานของกระบวนการใหม่แบบย่อๆ ก็จะเห็นได้ว่าวิธีการทำงานในรูปแบบใหม่นี้สามารถที่จะหารูปแบบทั้งหมดได้ครบถ้วนและยังใช้จำนวนรอบในการอ่านข้อมูลน้อยกว่าทุกวิธี ซึ่งเหตุผลดังที่กล่าวมาเหล่านี้จึงทำให้น่าเชื่อว่าวิธีการใหม่นี้มีประสิทธิภาพในการทำงานดีกว่าวิธีการพื้นฐานก่อนหน้าที่มีมา



รูปที่ 3.2 ลำดับการเข้าไปเพิ่มค่าสนับสนุนให้กับรูปแบบไอเท็มต่างๆ



รูปที่ 3.3 ขั้นตอนการทำงานของวิธีการใหม่เมื่อกำหนดค่าสนับสนุนต่ำสุดเท่ากับ 50%

จากรูปแบบขั้นตอนการทำงานข้างต้นก็จะพบว่าวิธีการใหม่นี้สามารถทำงานได้เร็วกว่าวิธีการ Apriori และ DIC มากและถึงแม้ว่าวิธีการนี้จะมีการเตรียมข้อมูลบ้างแต่ก็พบว่าเวลาที่ใช้ในการเตรียมข้อมูลนั้นน้อยมากจนไม่มีผลกระทบต่อการทำงานเท่าไหร่นั้น เหตุผลที่กล่าวได้ว่าวิธีการใหม่นี้ทำงานได้เร็วกว่าวิธีการ Apriori และ DIC นั้นก็เพราะว่า จากผลลัพธ์ที่ได้จะเห็นว่าถ้าเป็นการทำงานในกระบวนการ Apriori นั้นจะต้องอ่านข้อมูลถึง 3 ครั้งจึงจะหยุด และหากว่าเป็นวิธีการ DIC ก็จะต้องอ่านถึง 2 รอบกว่าถ้าหากกำหนดให้ช่วงของการอ่านข้อมูลเป็น 3 ทรานแซกชั่น ในขณะที่วิธีการใหม่นั้นมีการอ่านเพื่อหารูปแบบใหม่เพียงรอบเดียวเท่านั้นถ้าหากไม่นับจากช่วงการเตรียมข้อมูล ดังนั้นจึงทำให้การทำงานของวิธีการใหม่นี้สามารถทำงานได้รวดเร็วขึ้นกว่าวิธีการอื่นที่ได้นำมาเปรียบเทียบเพราะเรามีการลดเวลาในส่วนของ I/O ลงไป และนอกจากที่จะช่วยในเรื่องของการลด I/O ของระบบลงแล้วเรายังได้นำคุณสมบัติในการเกิดรูปแบบไอเท็มมาใช้พิจารณาร่วมด้วย นั้นคือเราพบว่าถ้าหากในทรานแซกชั่นใดมีจำนวนไอเท็มเท่ากับ k ไอเท็ม แล้วรูปแบบไอเท็มเซตของทรานแซกชั่นนั้นจะเกิดได้ในตั้งแต่ 1 ไอเท็มเซตจนถึง kไอเท็มเซตเท่านั้น และย่อมที่จะไม่มีโอกาสเกิดรูปแบบไอเท็มเซตที่มากกว่า kไอเท็มเซตอย่างแน่นอน ดังนั้นเมื่อเราได้ทำการจัดเรียงทรานแซกชั่นใหม่โดยเรียงจากจำนวนไอเท็มน้อยไปสู่จำนวนไอเท็มมาก จึงทำให้เราสามารถที่จะอ่านทรานแซกชั่นทั้งหมดเพียงรอบเดียวเท่านั้น ก็ทำให้ได้ได้ค่าสนับสนุนของรูปแบบทั้งหมด

3.4 รูปแบบโครงสร้างข้อมูล

ขั้นตอนที่สำคัญขั้นตอนหนึ่งนั้นก็คือการเตรียมข้อมูลให้เหมาะสมกับงานที่เราจะทำ เช่น การปรับเปลี่ยนรูปแบบข้อมูลให้อยู่ในลักษณะที่เหมาะสมและง่ายต่อการนำไปใช้งาน ซึ่งในการทำงานของวิธีการแอสโซซิเอชั่นนั้นรูปแบบของฐานข้อมูลทรานแซกชั่นที่เหมาะสมนั่นมีอยู่ 2 รูปแบบคือ รูปแบบข้อมูลแนวนอน (Horizontal data) และรูปแบบข้อมูลแนวตั้ง (Vertical data) ดังที่ได้กล่าวมาแล้วในบทที่ 2 ซึ่งรูปแบบโครงสร้างข้อมูลที่เราใช้ในงานวิจัยนี้เป็นรูปแบบข้อมูลแนวนอนเพราะมีความเหมาะสมกับวิธีการการทำงานของเราซึ่งพิจารณาการเพิ่มรูปแบบไอเท็มระดับสูงขึ้นเมื่อจำนวนไอเท็มในทรานแซกชั่นเพิ่มขึ้น

รูปแบบข้อมูลที่เก็บอยู่ในลักษณะข้อมูลแนวนอน (Horizontal data) นั้นเป็นรูปแบบพื้นฐานทั่วๆไป โดยชุดข้อมูลนั้นประกอบไปด้วยรายการของทรานแซกชั่น โดยที่แต่ละทรานแซกชั่นจะมีค่ารหัสทรานแซกชั่น(Transaction IDentifier) เป็นตัวระบุถึงข้อมูลรายการซื้อสินค้า (Items) และสิ่งจำเป็นต้องเพิ่มเข้ามาสำหรับวิธีการนี้คือต้องมีการบอกจำนวนสินค้าในรายการแต่ละรายการด้วยทั้งนี้เพื่อให้ค่านี้สำหรับการเรียงข้อมูลรายการการซื้อขายสินค้า ซึ่งรูปแบบข้อมูลที่จัดเก็บเพื่อที่นำไปใช้งานของวิธีการใหม่นี้แสดงไว้ดังรูปที่ 3.4

Tid	Item	num_item
1	1,5,10,12,54,70,120,151,334,343,418,510,528,736,814,855,862,953	18
2	70,261,276,687,722,803,868,923,968	9
3	217,277,374,530,782,882,960	7
4	126,142,308,440,480,865,938,948,958,989	10
5	28,46,48,349,368,402,623,731,742,890,946	11
6	72,127,268,618,711,740,844,918	8
7	93,111,205,212,635,826,906,982	8
8	2,107,160,161,165,172,283,331,348,351,362,390,438,521,704,845,883,890,91	19
9	24,97,214,216,316,319,586,617,707,710,722,736,766,777,787,800,806,895	18
10	54,71,86,304,361,533,578,797,825,845,871,909,918	13
11	314,467,483,496,714,751	6
12	217,438,460,469,545,642,676,710,806,867,946	11
13	258,259,268,354,440,550,589,726,738,793,913,921	12
14	53,400,694,775,883,890,925,940	8
15	110,168,181,217,279,368,405,427,484,757,985	11
16	72,318,487,510,694,733,845,938	8
17	0,41,207,294,597,768,852,878	8
18	276,339,529,558,593,651	6
19	123,272,348,402,421,470,604,705,711,811,820,984	12
20	124,424,523,529,766,893	6

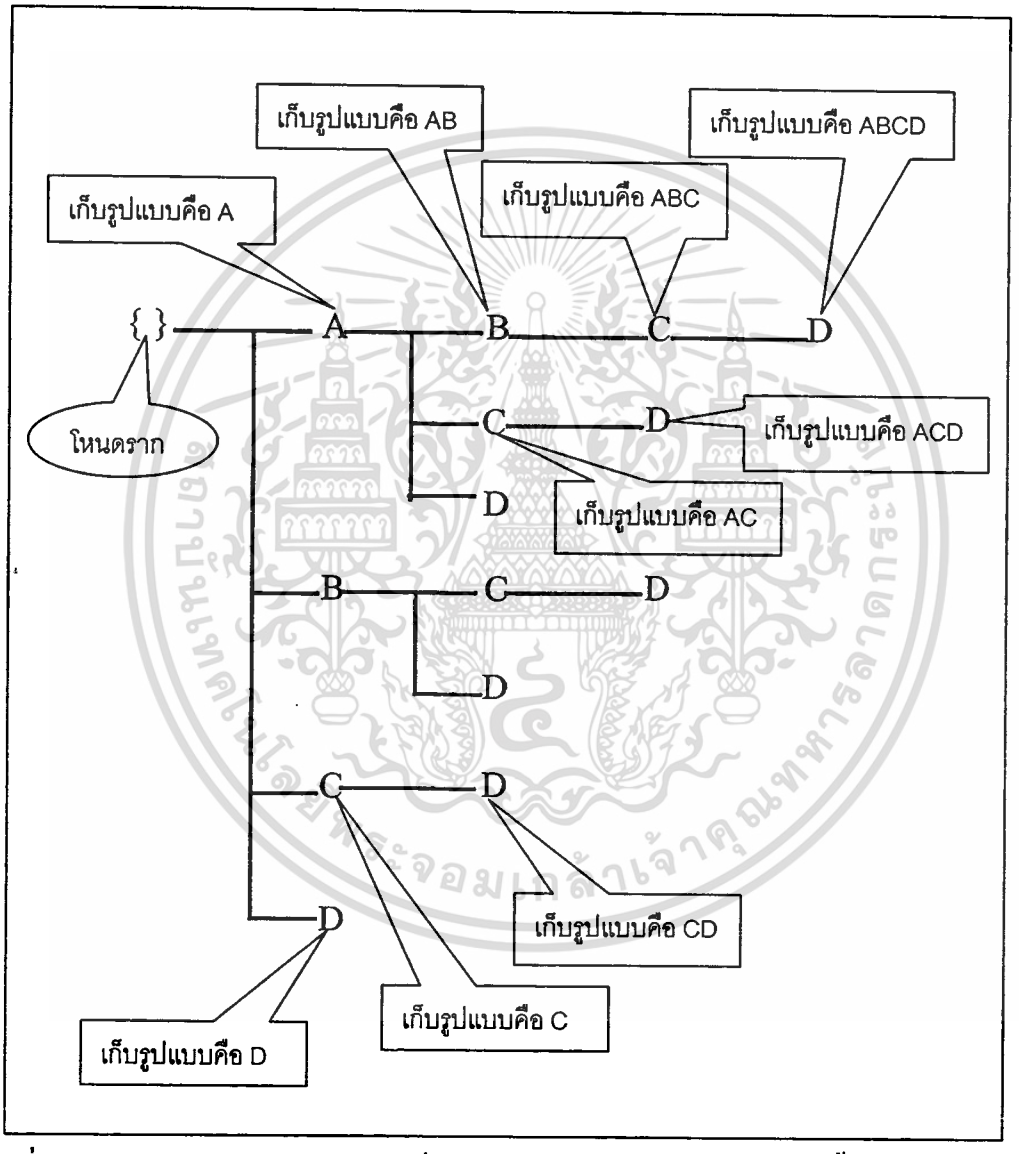
Record: 3 of 99888

รูปที่ 3.4 ตัวอย่างข้อมูลที่จัดเก็บแบบข้อมูลแนวนอน

3.5 โครงสร้างของการเก็บข้อมูลรูปแบบไอเท็มเซต

โครงสร้างข้อมูลสำหรับจัดเก็บไอเท็มนั้นนับได้ว่าเป็นส่วนสำคัญมากเลยส่วนหนึ่ง เพราะว่าข้อมูลได้จากส่วนนี้จะเป็นข้อมูลที่ถูกนำไปใช้เพื่อพิจารณาเลือกว่ารูปแบบใดบ้างที่ถือว่าเป็นรูปแบบที่น่าสนใจน่าจะมีประโยชน์ในการที่จะนำไปใช้งานต่อไป และถ้าหากว่าเรามีโครงสร้างในการจัดเก็บและวิธีการเข้าถึงข้อมูลส่วนนี้ช้าแล้ว ก็อาจจะมีผลทำให้การทำงานของวิธีการใหม่ที่ได้พัฒนาขึ้นนี้ช้าไปด้วย ดังนั้นโครงสร้างข้อมูลของรูปแบบไอเท็มเซตจึงเป็นโครงสร้างตัวหนึ่งที่มีความสำคัญต้องคำนึงถึงเป็นอย่างมาก โดยที่โครงสร้างข้อมูลที่ใช้ในวิธีการใหม่นี้คล้ายคลึงกับเทคนิคที่ใช้ในวิธีการ Apriori และ วิธีการ DIC ซึ่งก็คือการเก็บข้อมูลโดยใช้แฮชทรีโดยที่ แต่ละโหนดจะเก็บไอเท็มสุดท้ายของรูปแบบไอเท็ม นอกจากนั้นก็จะเก็บค่าแสดงค่าตัวนับและตัวระบุสถานะของไอเท็มนั้นไว้ด้วย สถานะของมันและกิ่งถ้ามันมีความสัมพันธ์กับโหนดอื่นๆตัวเซตว่างนั้นจะอยู่ที่ส่วนรากของทรี รูปแบบที่ระดับ 1 ไอเท็มเซต ทุกตัวจะก็ติดต่อกับโหนดราก และกิ่งของ

พวกมันการจะมีการระบุโดยค่าไอเท็มที่ปรากฏรูปแบบไอเท็มตัวอื่นๆต่างมีความสัมพันธ์กับตัวไอเท็มที่อยู่ก่อนหน้าทุกตัวยกเว้นตัวสุดท้าย โดยพวกมันจะระบุโดยใช้ค่าไอเท็มตัวสุดท้าย ซึ่งรูปแบบการเข้าถึงข้อมูลในโครงสร้างจำลองการเก็บข้อมูลไอเท็มนั้นได้แสดงไว้ดังในรูปที่ 3.2 แล้วส่วนในรูปที่ 3.5 นี้จะยกตัวอย่างอธิบายถึงค่ารูปแบบที่จัดเก็บอยู่ในโครงสร้างข้อมูลแบบแฮชทรีที่เราจำลองขึ้นนี้



รูปที่ 3.5 ตัวอย่างอธิบายรูปแบบไอเท็มที่จัดเก็บในโครงสร้างแฮชทรีที่จำลองขึ้น

จากรูปแบบของโครงสร้างข้อมูลเข้าและโครงสร้างข้อมูลของการจัดเก็บรูปแบบไอเท็มต่างที่ใช้ในการทำงานกับวิธีการแบบใหม่นี้ รวมเข้ากับวิธีการทำงานของวิธีการใหม่ที่ได้พัฒนาขึ้นมา เมื่อเราได้ทำการทดลองกับข้อมูลการซื้อขายจริงที่ได้จำลองขึ้นมา โดยเราได้เขียนโปรแกรมการทำงานของทั้ง 3 วิธีการคือ วิธีการ Apriori ,วิธีการ DIC และวิธีการใหม่ที่พัฒนาขึ้น โดยใช้ ภาษา JAVA และเหตุผลของการใช้ภาษานี้ก็เพราะว่ามีเครื่องมือต่างๆที่สนับสนุนการทำงานอยู่มากมาย มีแหล่งข้อมูลที่จะช่วยเมื่อเรามีปัญหาในการพัฒนาโปรแกรมอยู่มากกมาย และภาษา JAVA ยังสามารถที่จะทำงานได้กับทุกๆระบบปฏิบัติการ ดังนั้นเมื่อต้องการนำวิธีการเหล่านี้ไปใช้งานก็จะสามารถใช้งานได้ง่าย และสำหรับเรื่องการทดสอบการทำงานของวิธีการทั้ง 3 วิธีการนั้นจะได้แสดงไว้ในบทที่ 4

บทที่ 4
การทดสอบและวัดประสิทธิภาพ

4.1 การเตรียมเครื่องมือสำหรับการทดสอบ

ในการทดสอบประสิทธิการทำงานของกระบวนการใหม่ที่ได้พัฒนาขึ้น เมื่อเทียบกับกระบวนการ Apriori และกระบวนการ DIC นั้นเราจำเป็นที่จะต้องใช้ข้อมูลชุดเดียวกันในการทดสอบ ทั้งนี้ก็เพื่อเป็นการตรวจสอบความถูกต้องของผลลัพธ์ที่ได้จากการทำงานของแต่ละวิธีการ โดยเราสร้างชุดข้อมูลที่จะใช้ในการทดสอบวัดประสิทธิภาพการทำงานของแต่ละวิธี จากโปรแกรมของบริษัท IBM ซึ่งเป็นเครื่องมือสำหรับสร้างชุดข้อมูลรายการซื้อขายสินค้าที่ใช้มากในหลายๆ งานวิจัยที่ได้นำมาเป็นตัวอย่างอ้างอิง อย่างเช่นในงานวิจัยของ Mohammed Javeed Zaki และคณะ เรื่อง "A New Algorithm for Fast discovery of Association rule" เป็นต้น โดยที่ข้อมูลและรายละเอียดต่างๆของโปรแกรมตัวนี้สำหรับผู้ที่สนใจนั้นท่านสามารถที่จะศึกษาได้จากที่โฮมเพจตามที่อยู่ต่อไปนี้ http://www.almaden.ibm.com/cs/quest/syndata.html

ในส่วนนี้เป็นการทดสอบประสิทธิภาพของวิธีการใหม่กับวิธีการแบบ Apriori และ DIC ครั้งนี้เราได้ทำการทดสอบเครื่องที่เป็นเครื่องคอมพิวเตอร์แบบพีซี และคุณสมบัติของเครื่องที่ใช้เป็นตัวทดสอบมีดังนี้

- เครื่องที่มีความเร็วในการประมวลผลคำสั่ง 1 GHz MIPS
- หน่วยความจำ 256 MB
- ฮาร์ดดิสก์ขนาด 20 GByte

และข้อมูลที่ใช้สำหรับทดสอบนั้นเป็นชุดข้อมูลที่แสดงรายการซื้อขายสินค้า โดยเก็บไว้ในรูปของฐานข้อมูล และจากการสร้างขึ้นโดยใช้โปรแกรมของ IBM ดังที่ได้กล่าวมาแล้วในตอนต้น และในการสร้างฐานข้อมูลขึ้นมานั้นเราจะกำหนดความแตกต่างให้กับตัวแปรต่างเพื่อสร้างชุดข้อมูลที่มีความหลากหลาย. ทั้งนี้เพื่อที่จะนำมาใช้ในการทดสอบวิธีการต่างๆ ต่อไป โดยตัวแปรและความหมายของตัวแปรแต่ละตัว ที่เราจะต้องกำหนดนั้นดังแสดงในตารางที่ 4.1 และในการเปรียบเทียบประสิทธิภาพของการทำงานของแต่ละวิธีการนั้นจะพิจารณาในเรื่องต่างๆ ดังต่อไปนี้

- ขนาดข้อมูลที่ต่างกันหรือจำนวนรายการการซื้อ-ขายสินค้าโดยกำหนดให้ขนาดข้อมูลตั้งแต่ 100,000 ทรานแซกชั่นขึ้น
- จำนวนสินค้า(Item) ที่ขายอยู่ในร้าน ซึ่งค่าส่วนใหญ่ที่ใช้คือ 1000 ชนิดขึ้นไป (เป็นค่าที่หลายๆ งานวิจัยใช้ในการทดสอบประสิทธิการทำงานของวิธีการที่ปรับปรุง)
- ค่าสนับสนุนต่ำสุด (Minimum Support) ที่จะใช้สำหรับการเลือกข้อมูลที่น่าสนใจ ซึ่งเราก็จะกำหนดค่าให้อยู่ในช่วง 3.0 % - 0.01 %

และในการวัดนั้นเราจะเริ่มจับเวลาในการทำงานของแต่ละกระบวนการตั้งแต่เริ่มต้นสร้างรูปแบบและได้รูปแบบทั้งหมดออกมา โดยโปรแกรมต้นแบบของวิธีการทั้ง 3 แบบนั้นเขียนโดยใช้ภาษา JAVA

4.2 การสร้างชุดข้อมูลสมมุติ

เพื่อการคำนวณหาค่าประสิทธิภาพการทำงานของกระบวนการวิธีต่างๆ กับข้อมูลที่มีความหลายหลากข้อมูลมากๆ ดังนั้นเราจึงสร้างข้อมูลจำลองของรายการการซื้อขายสินค้าของลูกค้าที่จะสามารถแทนเหตุการณ์จริงที่เกิดขึ้น โดยถ้าหากพิจารณาจากเหตุการณ์จริงที่เกิดขึ้นนั้นเราอาจจะเห็นได้ว่าลูกค้าแต่ละคนอาจจะมีการซื้อหลายครั้ง แต่ในกระบวนการนี้เราจะพิจารณาโดยไม่สนใจว่าเป็นรายการซื้อที่เกิดมาจากลูกค้าคนเดียวกันหรือไม่ และในโปรแกรมการสร้างข้อมูลนั้นเราจำเป็นที่จะต้องมีการกำหนดตัวแปรต่างให้กับระบบโดยมีตัวแปรต่างๆที่ต้องกำหนดดังในตารางที่ 4.1

ตารางที่ 4.1 ตัวแปรที่ใช้ในการสร้างฐานข้อมูลสมมุติ

สัญลักษณ์	ความหมาย
|D|	จำนวนทรานแซกชั่น (ขนาดของฐานข้อมูล)
|T|	ค่าเฉลี่ยจำนวนไอเท็มต่อทรานแซกชั่น
|I|	ค่าเฉลี่ยขนาดสูงสุดของชุดความสัมพันธ์ของไอเท็ม (จำนวนไอเท็มในรูปแบบ)
L	จำนวนรูปแบบความสัมพันธ์ทั้งหมด
N	จำนวนไอเท็ม

ตารางที่ 4.2 ตัวอย่างการกำหนดตัวแปรในฐานข้อมูลสมมุติ

DATASET	D	T	I	L	N
Transa8	10,000	4	4	200	8
T10I2D100K	100,000	10	2	1,000	1,000
T10I4D100K	100,000	10	4	1,000	1,000
T10I4D200K	200,000	10	4	1,000	1,000
T10I6D200K	200,000	10	6	1,000	1,000
T20I6D100K	100,000	20	6	1,000	1,000

จากตารางที่ 4.2 ซึ่งแสดงการกำหนดค่าต่างของฐานข้อมูลสมมุติที่ใช้สำหรับการทดสอบประสิทธิภาพการทำงานของแต่ละวิธีการ ซึ่งการกำหนดค่าตัวแปรต่างของฐานข้อมูลจำลองในลักษณะได้ใช้ในหลายๆ งานวิจัย [1, 5, 6] ดังนั้นผู้วิจัยจึงได้ยึดเอารูปแบบเหล่านี้มาใช้ในสร้างฐานข้อมูล เพื่อใช้ในการทำการทดสอบประสิทธิภาพของวิธีการใหม่นี้เทียบกับวิธีการ Apriori และวิธีการ DIC ด้วยเช่นกัน

4.3 รูปแบบการทำงานของโปรแกรม

ในการนำเอาวิธีการใหม่ที่ได้พัฒนาขึ้นมาใช้งานจริงนั้นผู้วิจัยได้เลือกใช้ภาษา JAVA เพื่อแสดงการทำงาน ส่วนวิธีการ Apriori และวิธีการ DIC ก็ใช้ภาษาเดียวกันเพื่อแสดงการทำงานของวิธีการเหล่านั้น ทั้งนี้เพื่อให้สิ่งแวดล้อมในการทำงานของทั้ง 3 วิธีการนี้เหมือนกันให้มากที่สุด และผลลัพธ์ที่ได้จากการทำงานของทั้ง 3 โปรแกรมก็จะได้แสดงไว้ดังรูปที่ 4.1 ซึ่งเป็นรูปผลลัพธ์ในการรูปแบบไอเท็มเซตที่น่าสนใจของการทำงานกับชุดข้อมูล Transa8 เมื่อเรากำหนดค่าสนับสนุนต่ำสุดคือ 0.25 โดยในการใช้งานโปรแกรมเหล่านี้นั้นเราต้องใส่ชื่อของไฟล์ข้อมูลที่เกี่ยวข้อง 2 ตัว คือ

- ชื่อไฟล์ข้อมูล
- ชื่อไฟล์คุณสมบัติของข้อมูล

4.3.1 ไฟล์ข้อมูล

ในไฟล์ข้อมูลนั้นเราจะมีข้อมูลเฉพาะรายการไอเท็มที่ลูกค้าซื้อในแต่ละทรานแซกชั่นเท่านั้น โดยไอเท็มเหล่านี้จะต้องมีจัดเรียงลำดับ ตัวอย่างเช่นสมมุติให้ I= {A, B, C, D, E} รูปแบบไอเท็มในทรานแซกชั่นนั้นจะต้องเรียงลำดับจากน้อยไปมาก ดังนั้นค่าในทรานแซกชั่นที่สามารถเกิดขึ้นได้ก็จะเป็น T={B, D, E} แต่จะไม่อยู่ในรูปแบบ T= {E, B, D} โดยเด็ดขาด โดยไฟล์ข้อมูลนี้จะจัดเก็บในรูปแบบนามสกุลเป็น TXT

4.3.2 ไฟล์คุณสมบัติของข้อมูล

ในไฟล์คุณสมบัติข้อมูลนั้นคือเป็นไฟล์ที่เก็บคุณสมบัติต่างๆ ของชุดข้อมูลที่เราใช้ในการทดสอบ เช่น การบอกจำนวนไอเท็มในฐานข้อมูลชุดนั้น จำนวนทรานแซกชั่นหรือรายการซื้อขายสินค้าทั้งหมดในฐานข้อมูลชุดนั้น และนอกจากนั้นก็ยังใช้สำหรับการระบุค่าสนับสนุนต่ำสุดที่ต้องการที่จะทดสอบด้วย

```
Algoritha apriori starting now.....

 Please enter transaction filename: transa8.txt
 Please enter configuration filename: config8.txt

Input configuration: 8 items, 10000 transactions, minsup = 0.25%

Frequent 1-itemsets:
[1, 2, 3, 4, 5, 6, 7, 8]
Frequent 2-itemsets:
[1 2, 1 3, 1 4, 1 5, 1 6, 1 7, 1 8, 2 3, 2 4, 2 5, 2 6, 2 7, 2 8, 3 4, 3 5, 3 6, 3 7, 3 8, 4 5, 4 6, 4 7, 4 8, 5 6,
Frequent 3-itemsets:
[1 2 3, 1 2 4, 1 2 5, 1 2 6, 1 2 7, 1 2 8, 1 3 4, 1 3 5, 1 3 6, 1 3 7, 1 4 5, 1 4 6, 1 4 7, 1 4 8, 1 5 6, 1 5 7, 1 5
Frequent 4-itemsets:
[1 2 3 4, 1 2 3 5, 1 2 3 6, 1 2 3 7, 1 2 4 5, 1 2 4 6, 1 2 4 7, 1 2 5 6, 1 2 5 7, 1 2 6 7, 1 3 4 5, 1 3 4 6, 1 3 4 7
Frequent 5-itemsets:
[1 2 3 4 5, 1 2 3 4 6, 1 2 3 4 7, 1 2 3 5 6, 1 2 3 5 7, 1 2 3 6 7, 1 2 4 5 6, 1 2 4 5 7, 1 2 4 6 7, 1 2 5 6 7, 1 3 4
Frequent 6-itemsets:
[1 2 3 4 5 6, 1 2 3 4 5 7, 1 2 3 4 6 7, 1 2 3 5 6 7, 1 2 4 5 6 7, 1 3 4 5 6 7, 1 4 5 6 7 8, 2 3 4 5 6 7]
Frequent 7-itemsets:
[1 2 3 4 5 6 7]

Execution time is: 17.0 seconds.
Total Itemset that maximum than 0.25% has :172 Itemset
Process Exit...
```

a) วิธีการ Apriori

```
Enter new transaction filename. transa8.txt
Enter new configuration filename: config8.txt

Input has  8 items, 10000 transactions, minsup = 0.25%
Processing Strating: 1.2.3.4.5 6.7.8.9 10.11.12.



Frequent 1-itemsets:
[1.2.3.4.5.6.7.8]
Frequent 2-itemsets:
[1 2.1 3.1 4.1 5.1 6.1 7.1 8.2 3.2 4.2 5.2 6.2 7.2 8.3 4.3 5.3 6.3 7.3 8.4 5.4 6.4 7.4 8.5 6.5 7.5 8.6 7.6 8.7 8]
Frequent 3-itemsets:
[1 2 3.1 2 4.1 2 5.1 2 6.1 2 7.1 2 8.1 3 4.1 3 5.1 3 6.1 3 7.1 4 5.1 4 6.1 4 7.1 4 8.1 5 6.1 5 7.1 5 8.1 6 7.1 6 8.1
Frequent 4-itemsets:
[1 2 3 4.1 2 3 5.1 2 3 6.1 2 3 7.1 2 4 5.1 2 4 6.1 2 4 7.1 2 5 6.1 2 5 7.1 2 6 7.1 3 4 5.1 3 4 6.1 3 4 7.1 3 5 6.1 3
Frequent 5-itemsets:
[1 2 3 4 5.1 2 3 4 6.1 2 3 4 7.1 2 3 5 6.1 2 3 5 7.1 2 3 6 7.1 2 4 5 6.1 2 4 5 7.1 2 4 6 7.1 2 5 6 7.1 3 4 5 6.1 3 4
Frequent 6-itemsets:
[1 2 3 4 5 6.1 2 3 4 5 7.1 2 3 4 6 7.1 2 3 5 6 7.1 2 4 5 6 7.1 3 4 5 6 7.1 4 5 6 7 8.2 3 4 5 6 7]
Frequent 7-itemsets:
[1 2 3 4 5 6 7]

Execution time is: 7.25 seconds.
Total Itemset that maximum than 0.25% has :172 Itemset
Process Exit...
```

b) วิธีการ DIC

```
 Please enter transaction filename: otran8.txt
 Please enter configuration filename: config8.txt

Input has : 8 items, 10000 transactions, minsup = 0.25% (counter >= 25.0 )
Processing Strating: 1.2.3.4.5.6



Frequent 1-itemsets:
[1.2.3.4.5.6.7.8]
Frequent 2-itemsets:
[1 2.1 3.1 4.1 5.1 6.1 7.1 8.2 3.2 4.2 5.2 6.2 7.2 8.3 4.3 5.3 6.3 7.3 8.4 5.4 6.4 7.4 8.5 6.5 7.5 8.6 7.6 8.7 8]
Frequent 3-itemsets:
[1 2 3.1 2 4.1 2 5.1 2 6.1 2 7.1 2 8.1 3 4.1 3 5.1 3 6.1 3 7.1 4 5.1 4 6.1 4 7.1 4 8.1 5 6.1 5 7.1 5 8.1 6 7.1 6 8.1 7
Frequent 4-itemsets:
[1 2 3 4.1 2 3 5.1 2 3 6.1 2 3 7.1 2 4 5.1 2 4 6.1 2 4 7.1 2 5 6.1 2 5 7.1 2 6 7.1 3 4 5.1 3 4 6.1 3 4 7.1 3 5 6.1 3 5
Frequent 5-itemsets:
[1 2 3 4 5.1 2 3 4 6.1 2 3 4 7.1 2 3 5 6.1 2 3 5 7.1 2 3 6 7.1 2 4 5 6.1 2 4 5 7.1 2 4 6 7.1 2 5 6 7.1 3 4 5 6.1 3 4 5
Frequent 6-itemsets:
[1 2 3 4 5 6.1 2 3 4 5 7.1 2 3 4 6 7.1 2 3 5 6 7.1 2 4 5 6 7.1 3 4 5 6 7.1 4 5 6 7 8.2 3 4 5 6 7]
Frequent 7-itemsets:
[1 2 3 4 5 6 7]

Execution time is: 3.08 seconds.
Total Itemset that maximum than 0.25% has :172 Itemset
Process Exit
```

c) วิธีการใหม่ (New Approach)

รูปที่ 4.1 ผลลัพธ์จากการทำงานของโปรแกรมกับข้อมูล Transa8 ที่ค่าสนับสนุนต่ำสุด= 0.25%

4.4 การวัดประสิทธิภาพ

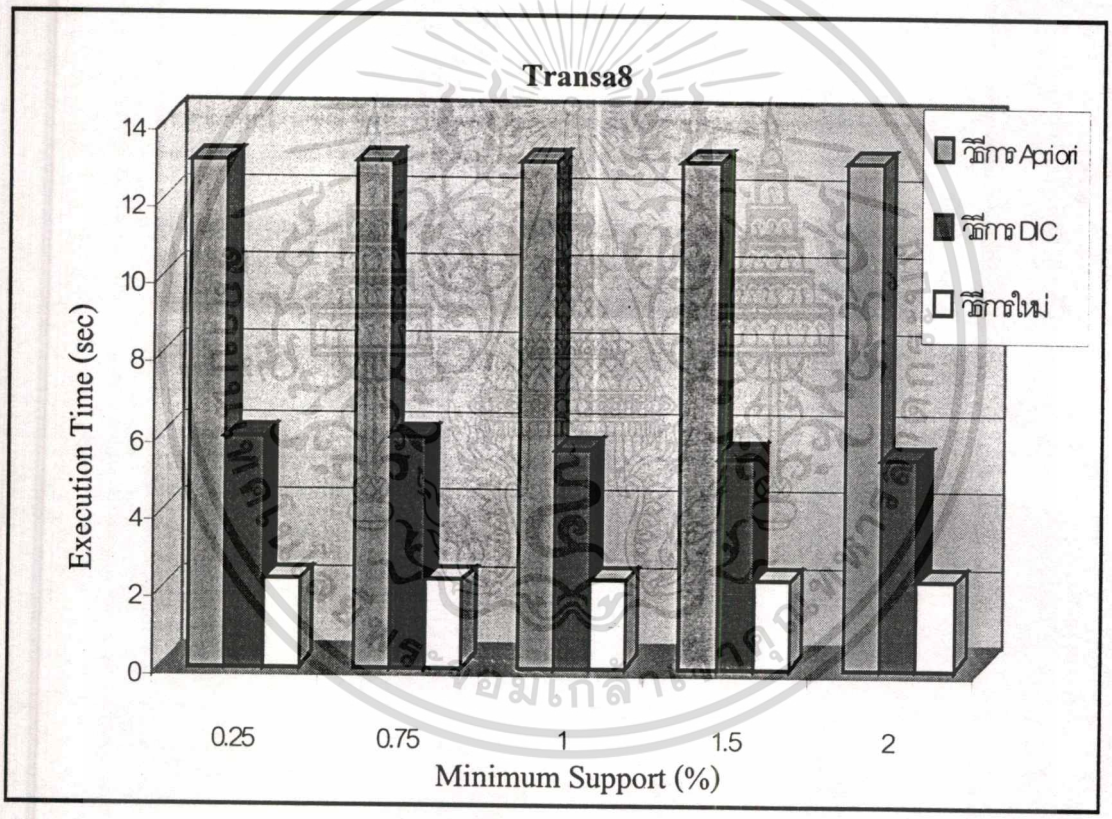
การเปรียบเทียบประสิทธิภาพการทำงานในเชิงเวลาการประมวลผล โดยพิจารณาเปรียบเทียบเวลาในการประมวลผลระหว่างวิธีการใหม่ที่พัฒนาขึ้นกับวิธีการแบบ Apriori และแบบ DIC ซึ่งเราจะสังเกตเห็นได้จากการเปลี่ยนแปลงค่าต่างๆ ต่อไปนี้

- ค่าสนับสนุนต่ำสุดลงเรื่อยๆ ซึ่งค่าค่าสนับสนุนต่ำสุดนี้จะเป็นค่าที่ใช้สำหรับการเลือกรูปแบบไอเท็มที่น่าสนใจมาใช้สร้างเป็นกฎต่อไป โดยที่จำนวนของรูปแบบที่ได้ก็จะมีจำนวนเพิ่มมากขึ้นเมื่อกำหนดให้ค่าสนับสนุนต่ำสุดที่ต่ำๆ จะใช้เวลาในการประมวลผลนานมากขึ้น ส่วนเมื่อค่าสนับสนุนสูงๆ เวลาในการประมวลผลก็จะน้อยลงเพราะจำนวนรูปแบบที่น่าสนใจจะลดลง
- ค่าจำนวนรายการทรานแซกชั่นในฐานข้อมูล ซึ่งเมื่อจำนวนรายการมากขึ้นก็ทำให้เวลาในการทำงานเพื่อรหารูปแบบที่น่าสนใจนั้นใช้เวลามากขึ้น
- ค่าจำนวนรูปแบบความสัมพันธ์ของข้อมูล โดยเมื่อเรากำหนดให้ค่านี้สูงๆ ซึ่งนั้นก็หมายความว่ารูปแบบทรีที่ใช้เก็บรูปแบบไอเท็มก็จะมีขนาดใหญ่และต้องมีการสร้างรูปแบบในระดับถัดไปมากขึ้น
- ค่าจำนวนค่าเฉลี่ยไอเท็มต่อทรานแซกชั่น ซึ่งในที่นี้ส่วนใหญ่เราจะไม่ค่อยทำการเปลี่ยนแปลงค่านี้สักเท่าไหร่ โดยค่าส่วนใหญ่ก็จะกำหนดไว้ที่ 10 ไอเท็มต่อทรานแซกชั่น

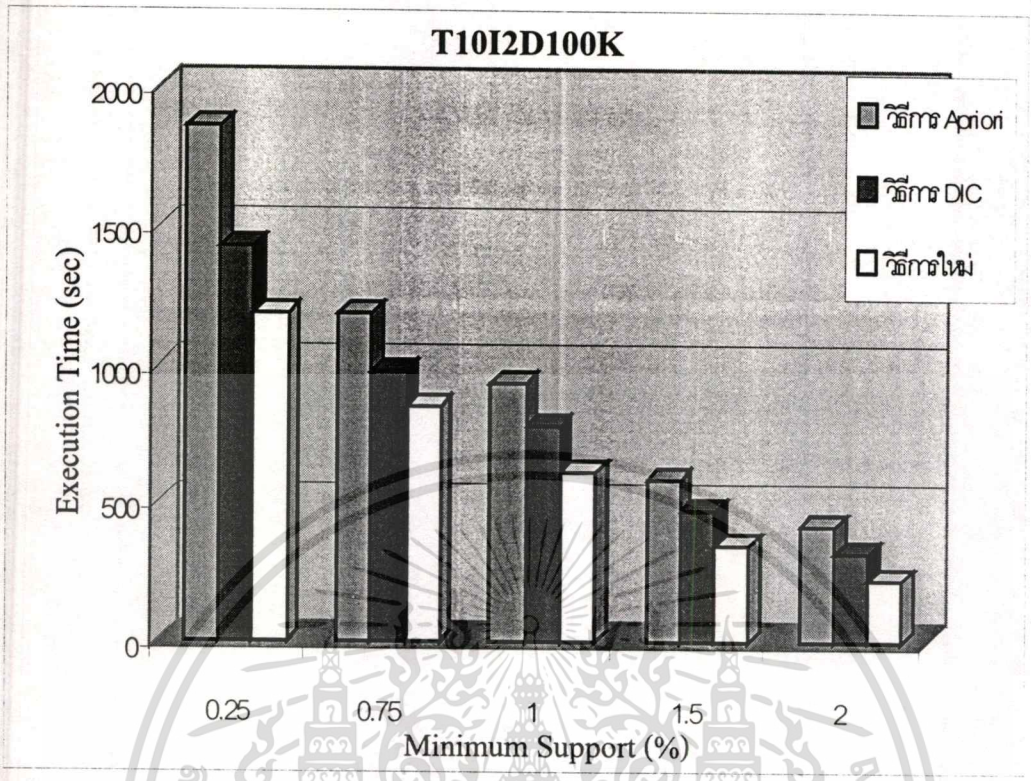
เมื่อเราเตรียมข้อมูลสำหรับใช้ในการทดสอบโปรแกรมเรียบร้อยแล้วเราก็จะเริ่มทำการทดสอบประสิทธิภาพการทำงานของแต่ละวิธีการต่อไป โดยที่ในการทดสอบแต่ละครั้งเราก็จะทำการเปลี่ยนแปลงค่าสนับสนุนต่ำสุด ไปเรื่อยๆ และในการทดลองกับแต่ละวิธีการของข้อมูลแต่ละตัวนั้นเราก็จะทำการทดสอบกับค่าสนับสนุนต่ำสุดที่แตกต่างกัน 5 ค่า ซึ่งแสดงไว้ดังในตารางที่ 5.3 วัดค่าเวลาในการทำงานของวิธีการแต่ละวิธี ซึ่งค่าเวลาที่ใช้ในการทำงานของวิธีการ Apriori, วิธีการ DIC และวิธีการใหม่นั้นดังที่ได้แสดงไว้ในรูปที่ 4.1 โดยในแต่ละรูปนั้นจะเป็นการแสดงกราฟแท่งที่บอกเวลาการทำงานของแต่ละวิธีการ เมื่อเรากำหนดค่าสนับสนุนต่ำสุดดังในตารางที่ 4.3 ของชุดข้อมูลทั้ง 7 ตัว ดังที่ได้แสดงชื่อชุดข้อมูลในตารางที่ 4.2

ตารางที่ 4.3 แสดงค่าสนับสนุนต่ำสุดที่ใช้ในการทดสอบสำหรับชุดข้อมูล

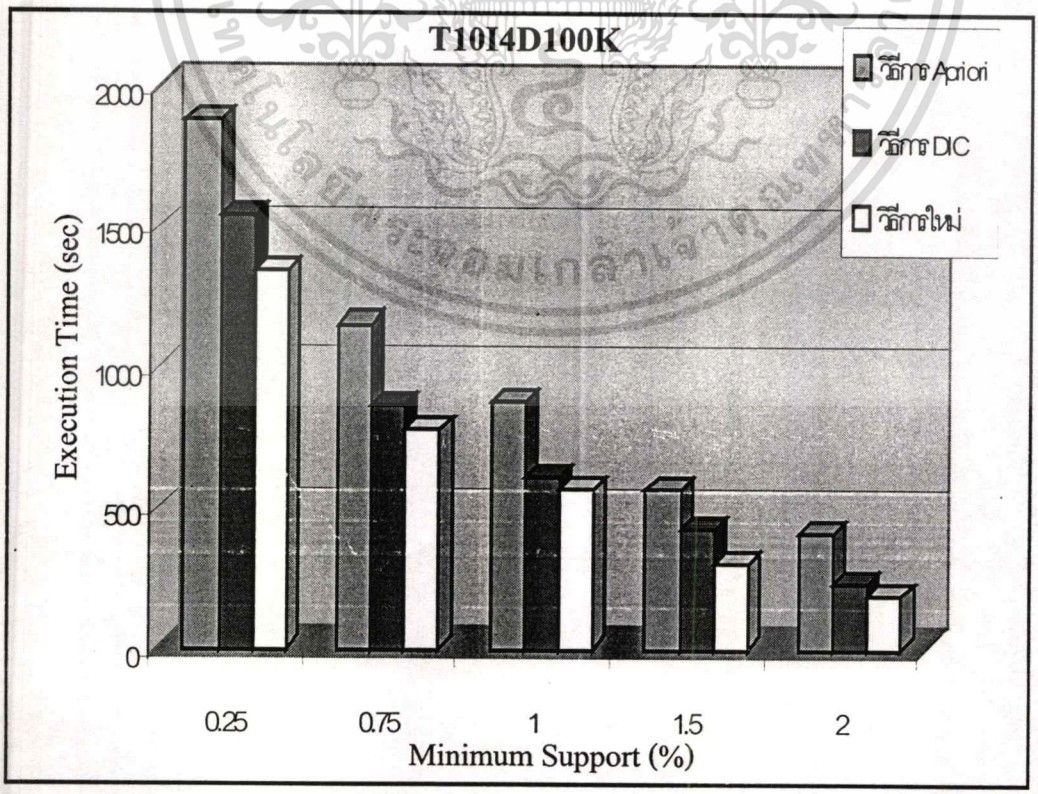
ฐานข้อมูล	ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3	ครั้งที่ 4	ครั้งที่ 5
Transa8	0.25 %	0.75 %	1.00 %	1.50 %	2.00 %
T10I2D100K	0.25 %	0.75 %	1.00 %	1.50 %	2.00 %
T10I4D100K	0.25 %	0.75 %	1.00 %	1.50 %	2.00 %
T10I4D200K	0.25 %	0.75 %	1.00 %	1.50 %	2.00 %
T10I6D200K	0.25 %	0.75 %	1.00 %	1.50 %	2.00 %
T20I6D100K	0.25 %	0.75 %	1.00 %	1.50 %	2.00 %



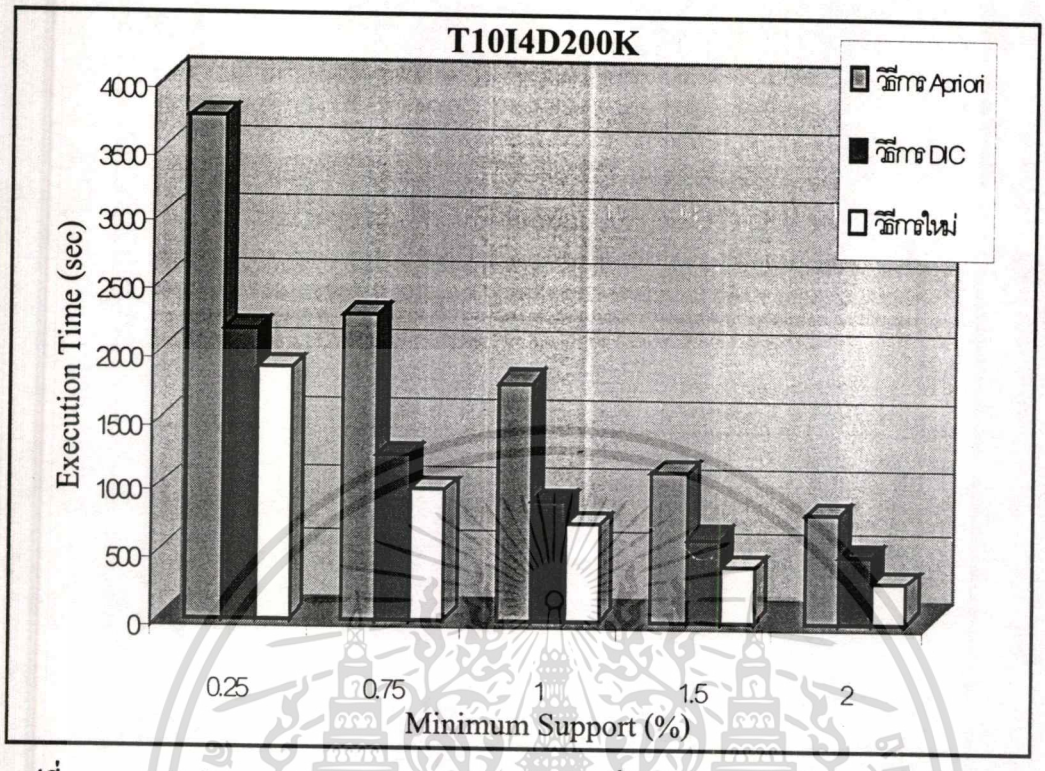
รูปที่ 4.2 กราฟแท่งแสดงเวลาในการหารูปแบบไอเท็มที่น่าสนใจของฐานข้อมูล Transa8



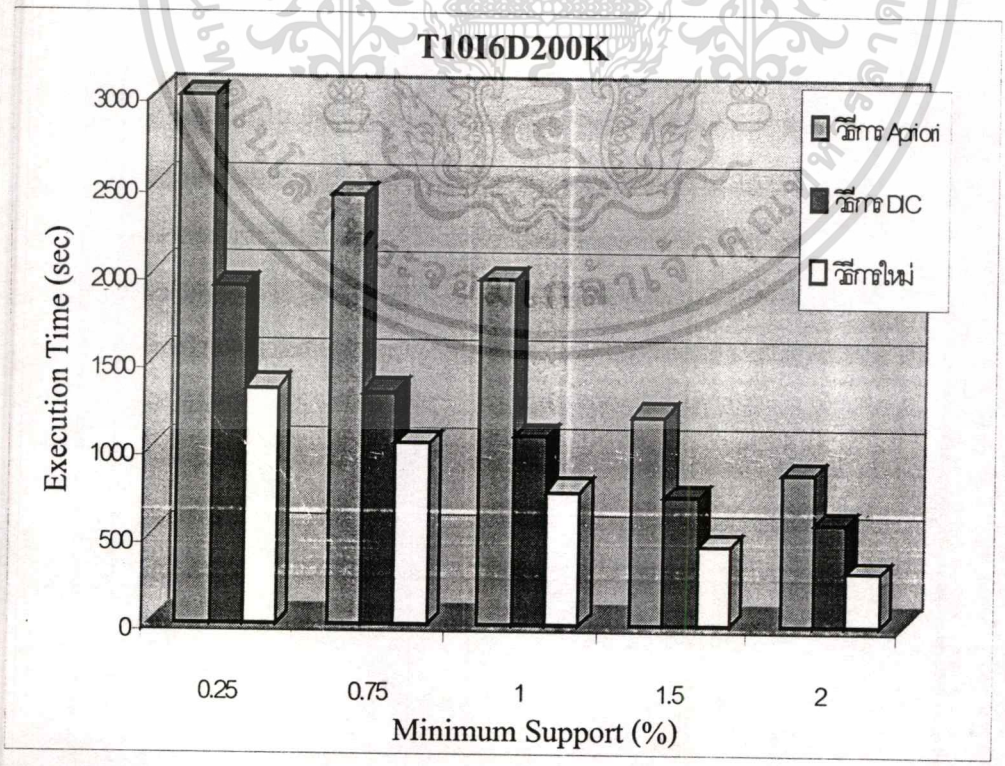
รูปที่ 4.3 กราฟแท่งแสดงเวลาในการหารูปแบบไอเท็มที่น่าสนใจของฐานข้อมูล T10I2D100K



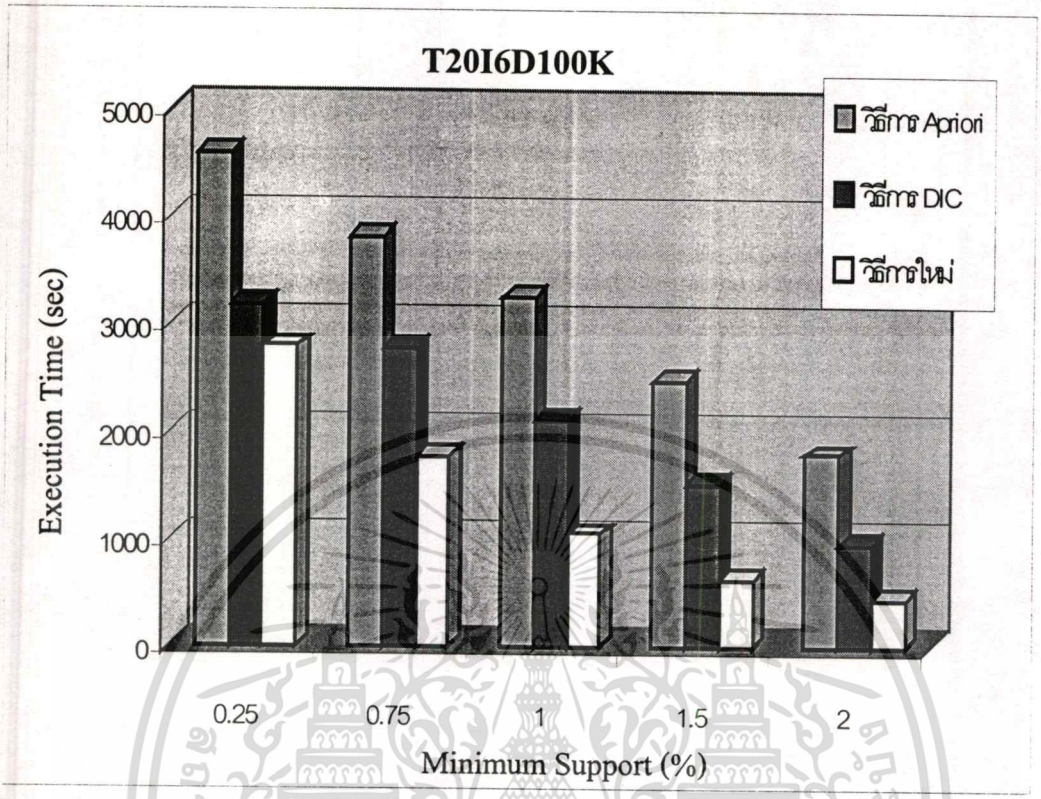
รูปที่ 4.4 กราฟแท่งแสดงเวลาในการหารูปแบบไอเท็มที่น่าสนใจของฐานข้อมูล T10I4D100K



รูปที่ 4.5 กราฟแท่งแสดงเวลาในการหารูปแบบไอเท็มที่น่าสนใจของฐานข้อมูล T10I4D200K



รูปที่ 4.6 กราฟแท่งแสดงเวลาในการหารูปแบบไอเท็มที่น่าสนใจของฐานข้อมูล T10I6D200K



รูปที่ 4.7 กราฟแท่งแสดงเวลาในการหารูปแบบไอเท็มที่น่าสนใจของฐานข้อมูล T20I6D100K

จากรูปที่ 4.2 ถึง รูปที่ 4.7 ซึ่งแสดงมาก่อนหน้านี้เป็นรูปที่แสดงให้เห็นถึงเวลาที่ใช้ในการทำงานเพื่อหารูปแบบไอเท็มเซตที่น่าสนใจของวิธีการ Apriori, วิธีการ DIC และวิธีการใหม่ที่ได้พัฒนาขึ้นกับฐานข้อมูลชุดต่างๆ ซึ่งเราสังเกตุได้ว่าแท่งกราฟที่แทนเวลาการทำงานของวิธีการใหม่นั้นเตี้ยที่สุดซึ่งนั้นก็หมายความว่าเวลาที่ใช้ในการทำการหารูปแบบที่น่าสนใจของข้อมูลชุดนี้เมื่อใช้วิธีการใหม่นั้นน้อยกว่าวิธีการ Apriori และวิธีการ DIC โดยที่จำนวนรูปแบบไอเท็มที่มีค่าสนับสนุนมากว่าค่าสนับสนุนต่ำสุดที่ได้จากวิธีการทั้ง 3 นั้น มีจำนวนเท่ากัน ดังนั้นเราจึงอาจจะสามารถสรุปได้ว่าวิธีการหารูปแบบไอเท็มที่น่าสนใจโดยวิธีการใหม่นี้ใช้เวลาในการหารูปแบบน้อยกว่าวิธีการ Apriori และวิธีการ DIC ทั้งนี้เหตุผลที่เป็นไปได้อันหนึ่งก็คือเรื่องจำนวนรอบในการอ่านข้อมูลเพื่อหารูปแบบไอเท็มเซตที่น่าสนใจ เนื่องจากเมื่อเราใช้วิธีการใหม่ในการหารูปแบบไอเท็มเซตซึ่งใช้จำนวนรอบในการอ่านข้อมูลแค่เพียงครั้งเดียวก็มีส่วนช่วยให้เวลาในการหารูปแบบไอเท็มเซตที่น่าสนใจนั้นลดลงไปด้วย เพราะว่าทั้งวิธีการ Apriori และวิธีการ DIC นั้นมีจำนวนรอบในการอ่านข้อมูลที่มากกว่าวิธีการใหม่

นอกจากนี้ผู้วิจัยยังแสดงข้อมูลเวลาเฉลี่ยในการหารูปแบบไอเท็มเซตที่น่าสนใจของแต่ละวิธีการ โดยค่าที่ได้มานี้เป็นค่าเฉลี่ยของวิธีการนั้นๆ เมื่อมีการเปลี่ยนแปลงค่าสนับสนุนต่ำสุดไปเรื่อยๆ ตามค่าที่ได้แสดงในตารางที่ 4.4 โดยที่เวลาเฉลี่ยแต่ละค่านั้นคำนวณได้ตามสมการที่ 4.1 และผลลัพธ์ของเวลาเฉลี่ยนั้น

$$Itime = \frac{Total\ Time}{Number\ of\ \ frequent\ Itemset} \tag{4.1}$$

โดยที่ *Itime* หมายถึงเวลาเฉลี่ยในการหารูปแบบไอเท็มเซตที่น่าสนใจแต่ละตัว

Total Time หมายถึงเวลาทั้งหมดที่ใช้ในการหารูปแบบไอเท็มเซต

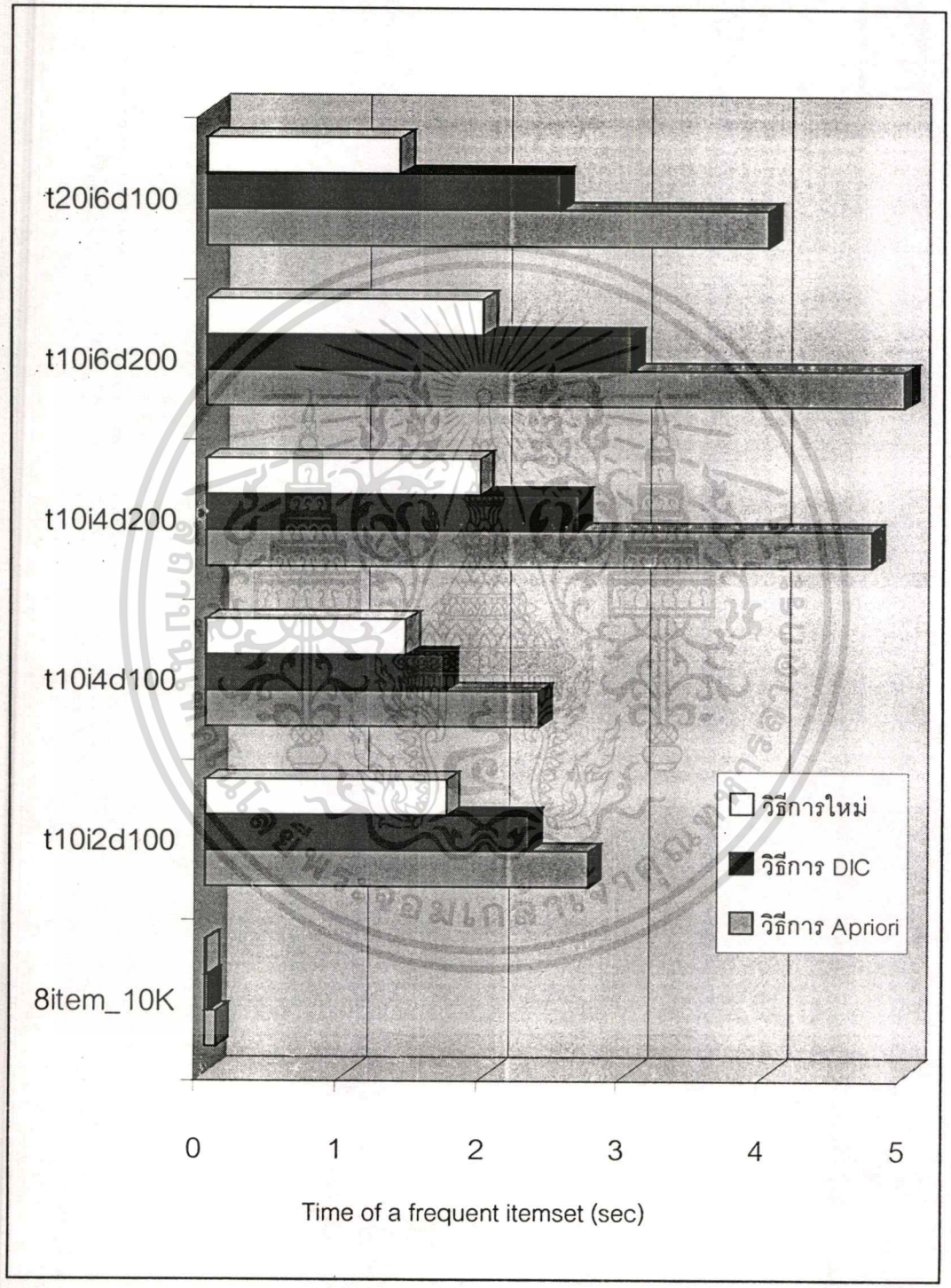
Number of Frequent Itemset หมายถึงจำนวนรูปแบบไอเท็มเซตทั้งหมดที่น่าสนใจ

ตารางที่ 4.4 แสดงเวลาเฉลี่ยในการหารูปแบบไอเท็มเซตที่น่าสนใจ

Data	Apriori (sec)	DIC (sec)	New Approach (sec)
Transa8	0.075	0.028	0.014
T10I2D100K	2.72	2.30	1.71
T10I4D100K	2.37	1.69	1.42
T10I4D200K	4.74	2.65	1.94
T10I6D200K	4.98	3.02	1.96
T20I6D100K	4.00	2.51	1.37

เรานั้นนำเสนอข้อมูลเหล่านี้ทั้งในรูปแบบของกราฟและตารางทั้งนี้เพื่อให้ง่ายในการพิจารณาเปรียบเทียบเวลาจากผลลัพธ์ของเวลาเฉลี่ยที่ใช้ในการหารูปแบบไอเท็มเซตที่น่าสนใจ 1 ตัว โดยที่แสดงไว้ในตารางที่ 4.4 และในรูปที่ 4.8 นั้นซึ่งจะพบว่าเวลาเฉลี่ยที่ใช้ในการหารูปแบบไอเท็มที่น่าสนใจแต่ละตัวโดยวิธีการใหม่ที่ได้พัฒนานั้นต่ำกว่าวิธีการ Apriori และวิธีการ DIC เมื่อจำนวนรูปแบบไอเท็มเซตที่น่าสนใจค่าต่างๆ ซึ่งเมื่อค่าเฉลี่ยของเวลาน้อยทำให้เวลาในการทำงานทั้งหมดของวิธีการใหม่นั้นมีค่าน้อยลงด้วยตามลำดับ และจากข้อมูลในตารางจะพบว่ายิ่งเมื่อจำนวนรายการซื้อ-ขายของข้อมูลมากขึ้นเวลาของวิธีการ Apriori และ DIC จะเพิ่มขึ้นอย่างมาก ดังสังเกตได้จากค่าเฉลี่ยในตัวอย่างข้อมูล T10I4D100K และ T10I4D200K ซึ่งจะพบว่าค่าเวลาเฉลี่ยของวิธีการ Apriori และ DIC นั้นเพิ่มขึ้นเกือบ 2 เท่าในขณะที่วิธีการใหม่นั้นเพิ่มขึ้นเพียงเล็กน้อยเท่านั้น

และแต่ถ้าหากว่าเป็นการเพิ่มค่าเฉลี่ยจำนวนไอเท็มสูงสุดในรูปแบบไอเท็มที่น่าสนใจหล่ะก็จะไม่ค่อยเห็นผลความแตกต่างของเวลาเฉลี่ยที่ใช้ในการหารูปแบบไอเท็มเซตแต่ละตัว ซึ่งสังเกตได้จากค่าของเวลาเฉลี่ยในชุดข้อมูล T10I2D100K กับ T10I4D100K



รูปที่ 4.8 กราฟแท่งแสดงเวลาเฉลี่ยในการหารูปแบบไอเท็มที่น่าสนใจแต่ละตัวของวิธีการทั้ง 3 วิธี

จากข้อมูลการทดลองทั้งหมดจึงทำให้สรุปได้ว่าวิธีการทำงานในการหารูปแบบไอเท็มเซต วิธีการใหม่นั้นใช้เวลาในการหารูปแบบไอเท็มเซตที่น่าสนใจน้อยกว่าวิธีการ Apriori และวิธีการ DIC ดังนั้นจึงอาจจะกล่าวได้ว่าจำนวนรอบในการอ่านข้อมูลนั้นมีผลต่อเวลาที่ใช้ในการหารูปแบบไอเท็มเซต เพราะจำนวนรอบในการอ่านข้อมูลของวิธีการใหม่นี้ใช้เพียง 1 รอบเท่านั้นส่วนวิธีการอื่นๆ นั้นใช้จำนวนรอบที่มากกว่าจึงมีผลทำให้เวลาในการทำงานมากขึ้นตามไปด้วยซึ่งผลสรุปทั้งหมดนั้นจะด้กล่าวไว้ในบทที่ 5 ต่อไป

บทที่ 5
สรุปผลการวิจัยและข้อเสนอแนะ

ในบทนี้จะกล่าวถึงผลสรุปที่ได้จากการทำวิทยานิพนธ์ และกล่าวถึงปัญหาที่เกิดขึ้นระหว่างการทำวิทยานิพนธ์ ครั้งนี้ รวมทั้งข้อเสนอแนะสำหรับผู้ที่สนใจที่จะทำวิทยานิพนธ์เรื่องนี้ต่อไป เพื่อจะได้ใช้เป็นแนวทางในการพัฒนาหรือศึกษาต่อไป

5.1 ข้อสรุป

ในการพัฒนาวิธีการเพื่อหารูปแบบไอเท็มเซตที่น่าสนใจโดยวิธีการใหม่ที่ได้พัฒนาขึ้นมานี้สามารถทำการหารูปแบบไอเท็มเซตที่น่าสนใจได้รวดเร็วกว่าวิธีการ Apriori และวิธีการ DIC เพราะเมื่อพิจารณาจากผลการทดลองที่ได้แสดงไว้ในบทที่ 4 โดยจากผลลัพธ์ที่ได้จึงทำให้พอสรุปได้ว่าจำนวนรอบในการอ่านข้อมูลเพื่อนับค่าสนับสนุนสำหรับรูปแบบไอเท็มเซตนั้นมีผลต่อเวลาในการทำการหารูปแบบไอเท็มเซตที่น่าสนใจ เพราะว่าวิธีการใหม่ที่พัฒนาขึ้นนี้เราสามารถที่จะอ่านข้อมูลในฐานข้อมูลเพียงครั้งเดียวก็ได้ค่าสนับสนุนของรูปแบบไอเท็มเซตทั้งหมดแล้ว ในขณะที่ จากวิธีการแบบ Apriori นั้นจำนวนรอบในการอ่านข้อมูลก็จะขึ้นอยู่กับความยาวสูงสุดของรูปแบบไอเท็มเซต ถ้าหากว่าความยาวสูงสุดของรูปแบบไอเท็มเซตนั้นเท่ากับ 4 นั้นก็หมายความว่าจำนวนรอบในการอ่านฐานข้อมูลก็จะเท่ากับ 4 โดยที่ในแต่ละรอบนั้นก็ต้องอ่านตั้งแต่รายการการซื้อขายตั้งแต่รายการแรกจนถึงรายการการซื้อขายตัวสุดท้ายทุกครั้งไป ดังนั้นจึงทำให้วิธีการนี้ต้องใช้เวลาในการอ่านข้อมูลจากฐานข้อมูลพอสมควร ส่วนจากวิธีการแบบ DIC นั้นถึงแม้ว่าวิธีการนี้จะพยายามที่จะลดจำนวนรอบในการอ่านข้อมูลลงแล้วก็ตามแต่ว่าวิธีการนี้ก็ยังจำเป็นที่จะต้องอ่านข้อมูลอย่างน้อย 2 รอบ เหตุผลอีกประการก็คือการกำหนดค่า M ซึ่งเป็นค่าที่ใช้สำหรับการแบ่งช่วงข้อมูล และเพื่อเป็นตัวระบุการสร้างรูปแบบไอเท็มเซตในระดับถัดไปนั้นค่อนข้างที่จะหาค่าที่เหมาะสมกับข้อมูลแต่ละตัวได้ยาก ดังนั้นจึงอาจจะเป็นสาเหตุที่ทำให้เวลาในการทำงานโดยใช้วิธีการนี้ยังคงช้าอยู่ ซึ่งปัญหาต่างเหล่านี้ที่ได้กล่าวมาวิธีการใหม่ที่ได้พัฒนาขึ้นมานี้ก็ได้ทำการแก้ไขโดยให้วิธีการใหม่นี้อ่านฐานข้อมูลเพียงครั้งเดียว และมีการสร้างรูปแบบไอเท็มเซตในระดับถัดไปเมื่อจำนวนไอเท็มในรายการการซื้อ-ขายสินค้าเพิ่มขึ้น ซึ่งการทำอย่างนี้ก็จะช่วยลดเหตุการณ์ที่จะต้องอ่านข้อมูลซ้ำ ซึ่งก็ช่วยทำให้การหารูปแบบไอเท็มเซตที่น่าสนใจนั้นทำได้รวดเร็วขึ้น

5.2 ผลที่ได้รับจากการทำวิทยานิพนธ์

ในการทำวิทยานิพนธ์ครั้งนี้เราได้รับประโยชน์มากมายโดยมีทั้งที่จะเกิดขึ้นกับตัวเองหรือกับสังคม หลายประการดังต่อไปนี้

1. ได้ฝึกฝนพัฒนาทักษะในเรื่องของการทำวิจัย
2. การฝึกการคิดและแก้ปัญหาอย่างเป็นระบบ
3. ฝึกฝนการค้นหาข้อมูลเพื่อใช้ประกอบการทำงานวิจัย
4. เป็นการเพิ่มพูนความรู้ความเข้าใจในเรื่องเหล่านี้ให้กับผู้ทำการวิจัยด้วย
5. ได้นำเสนอผลงานวิจัยนี้ออกสู่เวทีสัมมนาต่างๆ
6. รวมทั้งแนวทางที่ผู้วิจัยได้พัฒนาขึ้นนี้อาจจะสามารถนำไปใช้ประโยชน์เพื่อหารูปแบบความสัมพันธ์ของข้อมูลในด้านอื่นๆ ได้
7. รวมทั้งยังเป็นการพัฒนาวิธีการที่จะใช้ช่วยในการหารูปแบบไอเท็มเซตที่น่าสนใจได้รวดเร็วมากขึ้นกว่า
8. เป็นการพัฒนางานวิจัยของประเทศ

5.3 ปัญหา

ปัญหาที่พบในการวิจัยคือ การที่จะนำข้อมูลจริงมาทดสอบกับระบบนั้นทำได้ยากเนื่องจากว่าข้อมูลเหล่านี้ส่วนใหญ่แล้วจะมีการเก็บก็ในบริษัทที่มีขนาดใหญ่ และบริษัทส่วนใหญ่ก็ถือว่าข้อมูลเหล่านี้นั้นเป็นความลับของทางบริษัท ดังนั้นเมื่อผู้วิจัยเข้าไปดำเนินขอข้อมูลรายการซื้อขายเหล่านี้จึงมักได้รับคำปฏิเสธ ที่ว่าไม่สามารถให้ข้อมูลส่วนนี้ได้ ดังนั้นจึงจำเป็นที่จะต้องใช้ข้อมูลที่จำลองขึ้นมาแทน แต่ว่าข้อมูลที่จำลองขึ้นมานี้ก็ถือได้ว่าเป็นข้อมูลที่มีความน่าเชื่อถือพอสมควร เพราะในงานวิจัยทางด้านนี้ส่วนใหญ่ก็จะใช้ข้อมูลเหล่านี้ในการทดสอบการทำงานของโปรแกรมเช่นกัน

ส่วนปัญหาอีกตัวหนึ่งคือเรื่องของหน่วยความจำหลักที่ใช้สำหรับเก็บข้อมูลรูปแบบไอเท็มทั้งหมดนั้น ซึ่งในบางครั้งหากว่าข้อมูลมีความสัมพันธ์กันมาก ทำให้รูปแบบที่เกิดขึ้นมากดังนั้น ก็อาจจะทำให้ไม่สามารถเก็บรูปแบบทั้งหมดได้ ก็จะทำให้โปรแกรมไม่สามารถที่จะทำงานต่อได้ แต่ปัญหาเหล่านี้พบบ่อยในการทำวิจัยช่วงแรก แต่ภายหลังก็สามารถแก้ไขปัญหาเหล่านี้ได้แล้ว โดยการใช้คุณสมบัติของภาษาโปรแกรม นั้นคือในช่วงแรกของการทำวิจัยก็ได้ใช้ภาษา Visual Basic ในการเขียนโปรแกรม แต่การเข้าไปจัดการเรื่องหน่วยความจำนั้นทำได้ยาก ดังนั้นผู้วิจัยจึงเปลี่ยนมาใช้ภาษา Java ในการพัฒนาโปรแกรมแทน เพราะเป็นภาษาโปรแกรมที่มีเครื่องมือช่วยในการทำงานมาก

5.4 ข้อเสนอแนะ

ในการทำงานวิจัยเรื่องนี้ต่อไปในอนาคตผู้วิจัยก็ขอเสนอความคิดเห็นต่างเพื่อที่จะได้เป็นประโยชน์ต่อการทำงานวิจัยด้านนี้ต่อไปในอนาคต

1. น่าจะได้มีการนำเอาข้อมูลการซื้อ-ขายสินค้าที่เกิดขึ้นจริงเข้ามาเป็นข้อมูลที่ใช้ในการทดสอบโปรแกรม เพื่อว่าจะได้เห็นประสิทธิภาพการทำงานจริงๆ เมื่อใช้กับข้อมูลจริง
2. เมื่อนำไปใช้งานกับข้อมูลจริงต้องมีการเรียงลำดับไอเท็มในทรานแซกชั่นจากน้อยไปหามากด้วยเพราะหากไม่มีการเรียงลำดับไอเท็มในทรานแซกชั่นก็อาจทำให้การทำงานของโปรแกรมผิดพลาดได้
3. งานวิจัยนี้ได้ทำการทดสอบแต่เฉพาะกับวิธีการที่ได้มีการอ้างอิงและเป็นที่รู้จักมากในปัจุบัน แต่ยังไม่ได้ทำการทดลองเปรียบเทียบเวลากับงานวิจัยใหม่ๆ ที่พัฒนาออกในช่วงนี้ จึงคิดว่าต่อไปในอนาคตน่าจะได้ทำการทดสอบ วิธีการใหม่ที่ได้พัฒนาขึ้นกับงานวิจัยอื่นๆ
4. เนื่องจากในงานวิจัยนี้ไม่ได้รวมเวลาในส่วนของการเตรียมข้อมูล ดังนั้นในงานวิจัยที่จะพัฒนาเพิ่มต่อไปข้างหน้าก็น่าจะมีการรวมเอาเวลาในการเตรียมข้อมูลเหล่านี้เข้าไปรวมพิจารณาด้วย
5. ในการทำการทดลองใช้งานโปรแกรมควรใช้เครื่องที่มีหน่วยความจำขนาด 256 Mbytes ขึ้นไป เพราะจากการทดลองในเอกสารอ้างอิงส่วนใหญ่แล้วจะใช้หน่วยความจำขนาดนี้

เอกสารอ้างอิง

[1] Mohammed Javeed Zaki et al. "A New Algorithm for Fast discovery of Association rule." Technical Report 651, University of Rochester. 1997.

[2] Mohammed Javeed Zaki et al. "A Localized Algorithm for Parallel Association Mining." DepartMent of Computer Science, University of Rochester. 1998.

[3] N. Pasquier. et al. "Discovering frequent closed itemsets for association rule." Proc ICDT Conf. 1999. pp. 398-416.

[4] R. Agrawal et al. "Mining Association rule between set of item in large Database." In ACM SIGMOD Intl. Conf. Management of Data. 1993.

[5] Rakesh Agrawal and Ramakrishnan Srikant. "Mining Sequential Patterns." In Proc IEEE. 1995. pp. 3-14.

[6] Ulrich Guntzer and Jochen Hipp. "Algorithms for Association Rule Mining – A General survey and comparison." ACM SIGKDD volume 2., 2000. pp. 58-64.

[7] Ilias Petrounias. et al. "Discovering Temporal Association Rule: In Temporal Database." Proc. Of IADT'98. 1998 . pp. 312-319

[8] Christian Hidber. "Online Association Rule Mining." Technical Report UCB//CSD-98-1004. 1998.

[9] Sergey Brin. et al. "Dynamic Itemset Counting and Implication Rule for Market Basket Data." In ACM SIGMOD intl. Conf. Management of Data. 1997.

[10] "Algorithm for discovering frequent set" [online]. Available: http://www.lsi.upc.es/~gcasas/ algorism_ cjt_angles.htm.1998.

ภาคผนวก

ผลงานวิจัยที่ได้รับการตีพิมพ์

1. เชาวนี ศรีวิศาล และ รศ.ดร.วิเชียร เปรมชัยสวัสดิ์. "วิธีการแอสโซวิเอชั่นรูลย์สำหรับงานทางการตลาด" .ในงานประชุมสัมมนาทางวิชาการ NCSEC 2001 จัดโดยมหาวิทยาลัยเชียงใหม่. เดือนพฤศจิกายน 2544. หน้า 223-231.
2. Chouvanee Srivisal and Wichian Premchaiswadi. "A New Approach of Association Rules Algorithm for A Market Basket". ISCIT'2001 vol.1., 2001. pp.538-541.

วิธีการแอสโซซิเอชั่นรูลย์สำหรับงานทางการตลาด
(Association Rules Algorithm for Market Basket)

เชาวนี ศรีวิศาล

คณะเทคโนโลยีสารสนเทศ
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ถนนฉลองกรุง เขตลาดกระบัง กรุงเทพ 10520
Email :s2067010@kmitl.ac.th

รศ. ดร.วิเชียร เปรมชัยสวัสดิ์

คณะเทคโนโลยีสารสนเทศ
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ถนนฉลองกรุง เขตลาดกระบัง กรุงเทพ 10520
Email : wichian@it.kmitl.ac.th

บทคัดย่อ

งานวิจัยนี้นำเสนอวิธีการใหม่สำหรับการหากฎความสัมพันธ์ของงานทางการตลาด โดยกระบวนการที่นำเสนอนี้จะมีการอ่านฐานข้อมูลเพียง 2 ครั้ง โดยมีการทำงานหลักๆอยู่ 2 ขั้นตอน ดังนี้ ขั้นตอนที่1เป็นกระบวนการเตรียมข้อมูลก่อนที่จะนำเข้าไปหารูปแบบทั้งหมด โดยในขั้นตอนนี้จะได้รูปแบบในระดับ 1 ไอเท็มเซตและเรียงลำดับทรานเซ็กชั่นใหม่ตามจำนวนไอเท็มในทรานเซ็กชั่น การทำเช่นนี้จะช่วยลดจำนวนรอบในอ่านฐานข้อมูลเพื่อสร้างรูปแบบไอเท็มเซตทั้งหมด ส่วนขั้นตอนที่ 2 เป็นการหารูปแบบที่น่าสนใจทั้งหมดโดยเทียบกับค่าสนับสนุนต่ำสุด เพื่อแสดงให้เห็นประสิทธิภาพของวิธีการที่นำเสนอนี้ได้ทำการทดลองวิธีการที่นำเสนอนี้เปรียบเทียบกับวิธี Apriori และ DIC โดยพิจารณาจากเวลาที่ใช้ในการหารูปแบบไอเท็มเซต พบว่าวิธีการที่นำเสนอนี้ทำงานได้เร็วกว่าวิธีการ Apriori และ DIC ในขณะที่ให้ผลลัพธ์อย่างเดียวกับ โดยในการทดลองจะที่มีการเปลี่ยนแปลงค่า ค่าสนับสนุนต่ำสุด และจำนวนทรานเซ็กชั่นในฐานข้อมูล

Abstract

The paper proposes a new scheme of association rule algorithm for market basket. The algorithm, which uses fewer passes over the data, consists of two steps pre-processing and pattern classification. Preprocessing process is used to find 1-itemsets, which their support greater than minimum support, and to reorder transaction by the number of items in transaction. Pattern classification process is to employ all groups of items, which frequently appear together in transaction. The algorithm is implemented and tested with many value of minimum support and size of synthetic data. The experimental results are compared with that of the Apriori and DIC Algorithm. The experimental result shows that the scheme the execution time of the algorithm is less than that of the Apriori and DIC algorithm significantly.

1. บทนำ

ในปัจจุบันองค์กรทางธุรกิจกำลังเติบโตและมีการเก็บข้อมูลต่างๆไว้มากมาย ข้อมูลเหล่านี้สามารถที่จะนำไปใช้ให้เกิดประโยชน์ต่อการวางนโยบายขององค์กร จึงทำให้เกิดความต้องการในการที่จะดึงข้อมูลทางธุรกิจที่ได้เก็บรวบรวมไว้เหล่านี้ออกมาเพื่อใช้ประโยชน์ เช่นในการวินิจฉัยเพื่อวางนโยบายทางการตลาดของผู้บริหารในระดับสูงอาจได้มาจากข้อมูลรูปแบบการซื้อสินค้าของลูกค้า นอกจากที่จะนำไปใช้เพื่อการวางนโยบายทางด้านการตลาดแล้วอาจนำข้อมูลเหล่านี้ไปใช้สำหรับการตัดสินใจในเรื่องทิศทางการจัดเก็บสินค้าให้เหมาะสมกับช่วงเวลาอีกด้วย ในหลายกรณีที่เราไม่ทราบความสัมพันธ์ระหว่างข้อมูลมาก่อน ดังนั้นจึงมีความต้องการกระบวนการวินิจฉัยข้อมูลโดยอัตโนมัติเพื่อที่จะทำการค้นหาความรู้ที่แอบ

ซ่อนอยู่ในข้อมูลจำนวนมากซึ่งกระบวนการเหล่านี้เรียกว่า Knowledge discovery [1,2,5,6] โดยวิธีการแบบนี้เป็นลักษณะการทำงานแบบหนึ่งของงานทางด้านดาต้าไมนิ่ง (Data Mining) และวิธีการที่นิยมนำมาใช้เพื่อการค้นหาความรู้เกี่ยวกับรูปแบบการซื้อสินค้าของลูกค้านั้นคือวิธีการแบบ แอสโซซิเอชั่นรูล์ (Association Rule) โดยผลลัพธ์ที่ได้จากกระบวนการนี้จะอยู่ในรูปแบบของการระบุความน่าจะเป็นเมื่อมีการซื้อสินค้าชนิดหนึ่งแล้วจะทำให้มีการซื้อสินค้าอีกชนิดหนึ่งในการซื้อสินค้าครั้งเดียวกันกี่เปอร์เซ็นต์ ตัวโปรแกรมต้นแบบนี้ใช้ในการวิเคราะห์ข้อมูลด้านการขายหรือบาสเก็ตดาต้า (Basket data) โดยข้อมูลที่ใช้สำหรับการพิจารณานี้ประกอบด้วยรายการสินค้าที่ลูกค้าซื้อ (Itemset) และรายการแสดงทรานเซ็กชั่น (Transaction Identifier)

สำหรับงานวิจัยนี้ได้ใช้แนวคิดจากวิธีการ Association แบบ Apriori Algorithm [1,5,6,7] ซึ่งเป็นวิธีที่ได้รับความนิยมเป็นอย่างมากวิธีการหนึ่ง ในวิธีการแบบ Apriori นั้นจะต้องอ่านข้อมูลจากฐานข้อมูลหลายครั้งจนกว่าจะไม่มีรูปแบบใดที่มีค่ามากกว่าค่าสนับสนุนที่กำหนด (minimum support) ไว้แล้วจึงหยุด จึงทำให้เสียเวลามากในการที่จะต้องอ่านฐานข้อมูลหลายครั้ง นอกจากนั้นที่แต่ละระดับไอเท็มเซตยังจะต้องมีการเอารูปแบบจากระดับก่อนหน้ามาจอย (Join) กันเพื่อสร้างเป็นรูปแบบที่อาจเกิดขึ้นในระดับถัดไปอีก จะเห็นได้ว่าเวลาที่เสียไปกับการรอคอยเพื่อให้การทำงานที่ระดับหนึ่งเสร็จสิ้นก่อนแล้วจึงเริ่มการทำงานที่ระดับถัดมามีมากพอควร จากปัญหาดังกล่าวงานวิจัยนี้จึงพยายามหาวิธีการแบบใหม่เพื่อที่จะลดเวลาในส่วนนี้ลงไป โดยผลลัพธ์ที่ได้รับยังคงมีความถูกต้อง ซึ่งวิธีการใหม่นี้จะมีการอ่านฐานข้อมูลน้อยครั้งกว่าแต่ก็ได้รูปแบบทั้งหมดที่มีค่ามากกว่าค่าสนับสนุนที่เรากำหนด รวมทั้งมีการกำหนดโครงสร้างที่จะใช้ในการเก็บรูปแบบที่ง่ายในการนับเพิ่มค่า ทั้งหมดนี้จึงเป็นเหตุผลที่น่าจะทำให้การทำงานรวดเร็วมากยิ่งขึ้น

ในรายงานฉบับนี้จะได้กล่าวถึงเรื่องกระบวนการทำงานโดยทั่วไปของกระบวนการ Association Ruleในหัวข้อที่ 2 และในหัวข้อที่ 3 นั้นเป็นการยกตัวอย่างวิธีการที่นิยมใช้กันมากคือวิธีการ Apriori และวิธีการ DIC หัวข้อที่ 4 จะกล่าวถึงเรื่องวิธีการหารูปแบบการซื้อสินค้าโดยวิธีการใหม่ หัวข้อที่ 5 กล่าวถึงเรื่องของการทดสอบประสิทธิภาพการทำงานและความถูกต้องของผลลัพธ์ระหว่างวิธีการ Apriori และวิธีการ DIC เทียบกับวิธีการแบบใหม่และหัวข้อที่ 6 เป็นการสรุปผลการทำงานและข้อดีข้อเสียของวิธีการแบบใหม่

2. กระบวนการ Association Rule

ในส่วนนี้เราจะอธิบายกระบวนการทำงานของวิธีการแอสโซซิเอชั่น (Association Rule) เกี่ยวกับข้อมูลการซื้อ-ขายสินค้า โดยงานของวิธีการนี้คือเป็นการอธิบายกลุ่มของสินค้าที่มีการซื้อบ่อยเทียบกับกลุ่มสินค้าอื่น ตัวอย่างเช่น จากกฎเราอาจบอกได้ว่า " 80% ของลูกค้าที่ซื้อขนมปังและนมจะซื้อไข่ด้วยในครั้งเดียวกัน " โดยค่าที่ได้จากการทำงานของกระบวนการแอสโซซิเอชั่น (Association) [3,4] ที่นำไปพิจารณาบ่อยครั้งนั้นมีอยู่สองค่าที่สำคัญคือ ค่าสนับสนุน(support) และค่าความเชื่อมั่น (confident) โดยที่เงื่อนไขของกฎความสัมพันธ์ระหว่างไอเท็มเซตคือ เมื่อเกิด A และจะเกิด B เมื่อกำหนดให้ A และ B เป็นไอเท็มเซตที่ไม่ซ้ำกัน ซึ่งสามารถแทนเป็นสัญลักษณ์ได้ดังสมการที่ 1 ส่วนค่าความเชื่อมั่นของรูปแบบนี้คืออัตราส่วนของค่าสนับสนุนของรูปแบบกับค่าสนับสนุนของรูปแบบส่วนหน้าซึ่งเขียนได้ดังนี้สมการที่ 3

$$A \Rightarrow B, \text{ where itemset } A, B \subset I \text{ and } A \cap B = \phi \qquad (1)$$

2.1 ข้อกำหนดของปัญหา

กำหนดให้ $I = \{i_1, i_2, \ldots, i_n\}$ ชุดของเอ็ดทริบิว(Attribute) ที่แตกต่างกัน N ตัวบางครั้งเรียกว่าไอเท็ม (Items) โดยที่แต่ละทรานเซ็กชั่น (Transaction) T ในฐานข้อมูล D มีตัวที่อ้างถึงเพียงตัวเดียวเท่านั้น และภายในทรานเซ็คชั่น (Transactiion) นั้นจะมีชุดของรายการสินค้า (Items) ซึ่งจะเรียกว่า ไอเท็มเซต (Itemset) และไอเท็มเซตที่มี k ตัวจะเรียกว่า k-itemset

Transaction (T) มีการอธิบายเพียงแบบเดียวและประกอบไปด้วยชุดของไอเท็ม (Itemset): $T \subseteq I$ ยกตัวอย่างเช่น I = {A, B, C, D ,E} แต่ T = {B, D, E} เป็นต้น ฐานข้อมูล (Databases) D ประกอบด้วยหลายทรานเซ็กชั่น โดยที่ค่าสนับสนุน(Support) คือเศษส่วนของจำนวนทรานเซ็กชั่นที่มีรูปแบบนั้นอยู่กับจำนวนทรานเซ็กชั่นทั้งหมดในฐานข้อมูล ส่วนค่าความถี่ (*frequency*) คือค่าสนับสนุนของรูปแบบไอเท็มเซตที่มีค่ามากกว่าค่าสนับสนุนต่ำสุดที่กำหนดมา เมื่อกำหนดให้ α คือรูปแบบไอเท็มเซตของการซื้อสินค้าที่จะเกิดขึ้นได้ จะได้สูตรดังแสดงในสมการที่ 2 [1]

$$fr(\alpha, D) \quad or\, supp\,(\alpha, D) = \frac{|\{T \in D | T\ contains\ \alpha\}|}{|D|} \qquad (2)$$

โดยที่ $supp(\alpha, D)$ คือ ค่าสนับสนุนของรูปแบบ α

$fr(\alpha, D)$ คือ ค่าสนับสนุนของรูปแบบ α ที่มีค่ามากกว่าค่าสนับสนุนต่ำสุดที่กำหนด

$$confident = supp(A \cup B) / supp(A) \qquad (3)$$

โดยที่ *confident* คือ ค่าความเชื่อมั่นสำหรับกฎ

A คือรูปแบบไอเท็มเซตในส่วนหน้า

B คือ รูปแบบไอเท็มในส่วนหลัง

2.2 รูปแบบโครงสร้างข้อมูล

การแบ่งข้อมูลนั้นเป็นการแบ่งข้อมูลตามลักษณะการอ้างถึงข้อมูลเหล่านั้นโดยแบ่งได้เป็น 2 ลักษณะคือ ข้อมูลแนวนอน (Horizontal data) รูปแบบที่มีการอ้างถึงข้อมูลโดยใช้รหัสแทนรายการ (Transaction Identifier) เป็นตัวระบุถึงข้อมูล (Items) ซึ่งเป็นรูปแบบที่ใช้กันทั่วไป ส่วนรูปแบบที่ 2 คือรูปแบบโครงสร้างข้อมูลแบบแนวตั้ง (Vertical data) เป็นรูปแบบการอ้างถึงข้อมูลโดยใช้ค่ารหัสสินค้าในการอ้างถึงเพื่อบอกว่าสินค้าตัวนั้นเกิดขึ้นทรานเซ็กชั่นใดบ้าง ในวิธีการใหม่ของเรานั้นจะใช้โครงสร้างข้อมูลแบบแนวนอน ซึ่งเหมือนกันกับในวิธีการของ Apriori และ DIC

3. วิธีการต่างๆของ Association Rule

กระบวนการทำงานสำหรับงานทางด้าน Association Rule นั้นมีอยู่ด้วยกันหลายวิธีการ แต่ในที่นี้จะขอเสนอเพียง 2 วิธีการ นั้นคือวิธีการ Apriori และวิธีการ DIC

3.1 วิธีการ Apriori

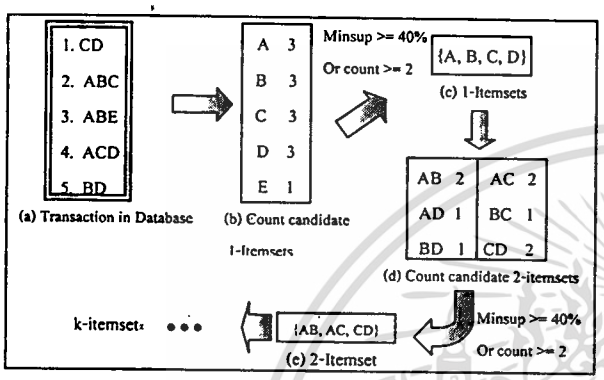
วิธีการ Apriori นั้นมีขั้นตอนการทำงานดังแสดงในรูปที่ 1 โดยมีการนับความสัมพันธ์ของชุด Item ที่เกิดขึ้นในข้อมูลจากบนลงล่าง โดยในแต่ละรอบของการทำงานก็จะนำเอาชุดของ item ที่พบในระดับก่อนหน้ามาจอยกันเพื่อสร้างเป็นรูปแบบที่อาจเกิดขึ้นได้ในระดับถัดไปแทนด้วยสัญลักษณ์ C_k โดยวิธี Apriori จะมีการทดสอบความถี่ของรูปแบบในระดับ k-1 ไอเท็มโดยการเลือกเฉพาะรูปแบบที่มีค่ามากกว่าค่าสนับสนุนต่ำมาจอยกันเพื่อสร้างเป็นรูปในระดับ k-ไอเท็ม [1, 2, 4, 6]

```
Pseudo code Apriori
  L1 = {frequent 1-Itemset}
  For (k = 2 , Lk-1 ≠ ϕ: k++)
    Ck = Set of New Candidates
    For all transaction t ∈D
      For all k-subset s of t
       If (s ∈ Ck) s.count ++
      Lk = {c ∈ Ck | c. count ≥ minimum support}
  Set of all frequent item set = UkLk
```

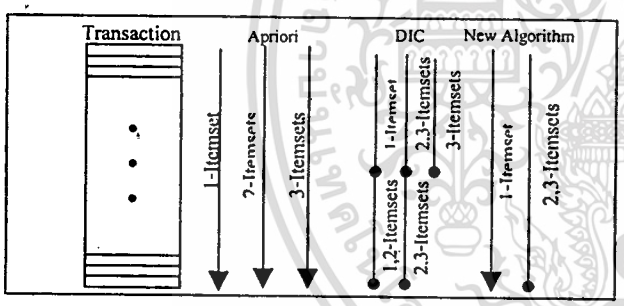
รูปที่ 1. กระบวนการทำงานของวิธี Apriori

จากตัวอย่างข้อมูลดังรูปที่ 2(a) ซึ่งมีไอเท็มทั้งหมด 5 ไอเท็ม คือ {A, B, C, D, E} โดยในที่นี้หากเรากำหนดให้ค่าสนับสนุนเท่ากับ 40% (จากตัวอย่างนี้รูปแบบที่จะนำพิจารณาในระดับถัดไปนั้นจะต้องมีอยู่ในทรานเซ็กชั่นอย่างน้อย 2 ทรานเซ็กชั่น) และเมื่อนับค่าสนับสนุนของรูปแบบที่อาจเกิดขึ้นในระดับ 1-ไอเท็มเซตจะได้ดังรูป 2(b) ซึ่งมีไอเท็มที่ได้รับเลือกไปเพื่อสร้างรูปแบบในระดับ 2-ไอเท็มเซตดังในรูปที่ 2(c) ส่วนรูปแบบของ 2-ไอเท็มเซตและค่าสนับสนุนของแต่ละรูปแบบในระดับ 2-ไอเท็มเซต แสดงในรูปที่ 2(d) จากนั้นก็เลือกเฉพาะรูปแบบที่มาก

กว่าค่าสนับสนุนเพื่อสร้างเป็นรูปแบบในระดับถัดไปโดยกระบวนการทำงานก็จะทำวนซ้ำเช่นนี้จนกว่าจะไม่มีรูปแบบใดที่ระดับ k-ไอเท็มเซตมีค่ามากกว่าค่าสนับสนุนก็จะหยุดการทำงาน ดังนั้นจะเห็นได้ว่าวิธีการนี้จำต้องเข้าไปอ่านข้อมูลหลายรอบเพื่อนับค่าสนับสนุนของรูปแบบของไอเท็มที่ระดับต่าง ดังรูปที่ 3



รูปที่ 2. ตัวอย่างการทำงานของกระบวนการ Apriori



รูปที่ 3. จำนวนรอบการอ่านข้อมูลของวิธี Apriori, DIC และวิธีใหม่ที่ระดับไอเท็มเซตสูงสุดเท่ากับ 3

3.2 กระบวนการ DIC (Dynamic Itemset Counting)

กระบวนการ DIC [6] เป็นวิธีการหนึ่งของแอสโซซิเอชั่นรูล์ (Association Rule) ซึ่งจะมีการนับค่าสนับสนุนของรูปแบบโดยไม่ต้องมีการอ่านฐานข้อมูลหลายรอบเท่ากับวิธีการ Apriori โดยที่วิธีการ DIC นี้จะลดจำนวนรอบในการทำงานกับข้อมูลระหว่างที่มีการเก็บค่าจำนวน itemset ซึ่งนับในรอบใดๆของความสัมพันธ์ซึ่งต่ำเท่ากับเมื่อเปรียบเทียบกระบวนการที่มีการสุ่ม โดยความรู้สึกเหมือนกับว่าด้านหลัง DIC มีการทำงานรถไฟที่กำลังวิ่งไปบนข้อมูลด้วยการหยุดที่ช่วง transaction M ตัว (M เป็นค่าตัวแปรในการทดลองของเราโดยเราพยายามหาค่าตั้งแต่ช่วง 100 ถึง 10,000) และเมื่อไปถึงส่วนสุดท้ายของไฟล์มันก็จะมีการทำงานหนึ่งรอบกับข้อมูลและมันก็จะเริ่มต้นที่ส่วนเริ่มต้นของรอบถัดไป โดยเราก็สมมุติให้ผู้โดยสารบนรถไฟนั้นก็คือ Itemset เมื่อ Itemset อยู่ในรถไฟแล้ว เราก็นับค่าเพิ่มเมื่อมันเกิดขึ้นใน transaction ที่อ่าน

ถ้าเราพิจารณา Apriori ในเชิงอุปมาอุปไมยทุก Itemset ที่เป็นไปได้ต้องมีอยู่ในตอนเริ่มต้นของการเข้าไปอ่านข้อมูลและเหลือออกมาเฉพาะ Itemset ที่ต้องการในตอนจบ โดย 1-itemset ที่ได้มาจากตอนรอบแรก ส่วน 2–itemset ได้ออกมาจากการอ่านในรอบที่ 2 ส่วนในวิธีการ DIC เราได้เพิ่มความยืดหยุ่นสำหรับการยอมให้มีการสร้าง Itemset ในระดับถัดไปในขณะที่การนับรูปแบบของระดับก่อนหน้ายังไม่จบ แต่ Itemset ที่แต่ละระดับก็จะมีจุดเริ่มต้นและจุดที่จะหยุดนับค่าสนับสนุนต่างกันไป ดังนั้นทุกทรานเซ็กชั่น ในไฟล์ได้เห็น Itemset เหล่านั้น และเมื่อวนมาถึงจุดเริ่มต้นของไอเท็มนั้นอีกรอบก็หยุดนับค่าสนับสนุนของรูปแบบ Itemset นั้นเพราะถือว่าได้อ่านครบทุกทรานเซ็กชั่นแล้ว ดังนั้นข้อดีของวิธีการนี้คือสามารถนับรูปแบบ itemset แต่ละระดับได้โดยไม่ต้องรอให้นับค่าสนับสนุนรูปแบบไอเท็มของระดับก่อนหน้าจบรอบก่อน

วิธีการ DIC นั้นจะมีการอธิบายแต่ละ itemset โดยการใช้ตัวระบุที่แตกต่างกันดังนี้

1) Solid Box (SS) ระบุอย่างชัดเจนว่าเป็นรูปแบบที่มากกว่าค่าที่กำหนดโดยแสดงไว้เมื่อจบการนับที่ระดับต่างๆ

2) Solid Circle (SC) ระบุอย่างชัดเจนว่าเป็นรูปแบบที่มีค่าน้อยกว่าค่าที่กำหนดเมื่อจบการนับที่ระดับนั้นๆ

3) Dashed Box (DS) เป็นการคาดว่ารูปแบบนั้นน่าจะมีค่ามากกว่าค่าที่กำหนดและกำลังมีการนับที่ระดับนั้นๆ อยู่

4) Dashed Circle (DC) เป็นการคาดว่ารูปแบบนั้นน่าจะมีค่ามากกว่าค่าที่กำหนดและกำลังมีการนับที่ระดับนั้นๆ อยู่

ขั้นตอนการทำงานของ DIC มีดังต่อไปนี้

1) ที่เซตว่างให้ระบุด้วย SS และ 1-itemset ให้ระบุด้วยรูป DC ส่วนตัวอื่นๆ ก็ไม่มีการระบุใด

2) อ่านทรานเซ็กชั่นมาทั้งหมด M

3) ถ้าหากว่าเป็น DC และเมื่อนับไปจนมีค่ามากกว่าค่าที่กำหนดก็เปลี่ยนเป็น DS และถ้ารูปแบบที่ใช้สร้างเป็น Superset นั้นมีรูปแบบเป็นรูป SS หรือ DS ก็ให้เพิ่มการนับสำหรับรูปแบบนั้นและระบุรูปแบบนั้นเป็น DC

4) ถ้ารูปแบบที่เป็น DS หรือ DC มีการนับจนครบแล้วก็ให้เปลี่ยนเป็น SS หรือ SC ตามลำดับแล้วหยุดนับค่ารูปแบบนั้น

5) ถ้าเราอยู่ที่ตำแหน่งสุดท้ายของทรานเซ็กซ์ชั่นก็ให้กลับไปเริ่มต้นใหม่

6) ถ้ายังมีรูปแบบที่เป็น DC หรือ DS ก็กลับไปเริ่มต้นใหม่

ในรูปแบบนี้ DIC เริ่มต้นนับโดย 1-itemset และมีการเพิ่มการนับเป็น 2, 3, 4, ..., k–itemsets หลังจากที่อ่านผ่านไป 2 –3 รอบ โดยตามแนวคิดนี่เราต้องการให้ค่า M มีค่าน้อยเท่าที่จะเป็นไปได้ดังนั้นเราสามารถเริ่มนับ itemset ได้เร็วที่สุดในขั้นตอนที่ 3 อย่างไรก็ตาม ขั้นตอนที่ 3 และ 4 นั้นก่อให้เกิด overhead จำนวนมากดังนั้นเราจึงไม่สามารถที่จะลด M ให้น้อยมากๆได้

4. กระบวนการทำงานแบบใหม่

กระบวนการใหม่นี้พัฒนาขึ้นเพื่อพัฒนาขั้นตอนการหาความรู้เกี่ยวกับงานทางด้านการตลาดให้มีความรวดเร็วมากขึ้นเนื่องจากเรามองเห็นว่าวิธีการแบบ Apriori นั้นต้องมีการเข้าไปอ่านข้อมูลจากฐานข้อมูลหลายครั้ง โดยแต่ละครั้งนั้นก็ย่อมต้องเสียเวลามาก ดังนั้นเราจึงได้ใช้แนวคิดจากวิธีการ DIC ซึ่งเป็นวิธีการที่ช่วยลดจำนวนครั้งในการอ่านข้อมูล แต่ก็มีข้อเสียในเรื่องของการทำงานกับข้อมูลที่มีความหลากหลายมาก เราจึงพยายามหาวิธีการในการที่จะลดการกระจายของข้อมูลลงและให้มีการเริ่มสร้างรูปแบบไอเท็มเซตในระดับถัดไปเร็วขึ้น โดยการจัดเตรียมข้อมูลเข้าเพื่อลดความหลากหลายของข้อมูลลงและเปลี่ยนรูปแบบการกำหนดช่วงการอ่านข้อมูลเพื่อนับเพิ่มรูปแบบไอเท็มในระดับต่างๆ

วิธีการแบบใหม่นั้นจะมีการอ่านฐานข้อมูลครั้งแรกเพื่อเป็นการเตรียมข้อมูลและลดจำนวนข้อมูลที่จะต้องใช้สำหรับสร้างเป็นทรีและมีการจัดเรียงทรานเซ็กซ์ชั่นใหม่โดยเรียงตามจำนวนไอเท็มในทรานเซ็กซ์ชั่น จากนั้นเราจะอ่านค่าไอเท็มที่เกิดขึ้นในแต่ละทรานเซ็กซ์ชั่นเข้ามาแล้วนำไปนับเพิ่มค่ารูปแบบไอเท็มเซตที่เกิดขึ้น ในทรานเซ็กซ์ชั่นโดยโครงสร้างของตัวที่เป็นตัวเก็บรูปแบบไอเท็มเซตที่ใช้นั้นจะมีรูปแบบเป็นทรี

4.1 กระบวนการทำงานของวิธีการใหม่

กระบวนการทำงานแบบใหม่นี้จะมีการลดจำนวนรูปแบบไอเท็มที่จะนำมาสร้างเป็นทรีในระดับที่หนึ่งลงเพราะจะมีการหาไอเท็มเซตที่ระดับ 1-ไอเท็มเซตที่มีค่ามากกว่าค่าสนับสนุนมาก่อน รวมทั้งมีการลดข้อมูลในทรานเซ็กซ์ชั่นให้มีเฉพาะไอเท็มที่เลือกมา

ขั้นตอนการทำงานของวิธีการหาความสัมพันธ์ของการซื้อสินค้าแบบใหม่ที่ได้พัฒนาขึ้นเป็นไปตามลำดับต่อไปนี้

1. อ่านทรานเซ็กซ์ชั่นทั้งหมดหนึ่งรอบเพื่อหารูปแบบในระดับ 1 ไอเท็มเซตและปรับปรุงทรานเซ็กซ์ชั่นใหม่อีกครั้งโดยการตัดไอเท็มที่ไม่อยู่ในไอเท็มเซตระดับที่ 1 ออกจากทรานเซ็กซ์ชั่น และเรียงทรานเซ็กซ์ชั่นเหล่านั้นใหม่โดยเรียงตามจำนวนไอเท็มในทรานเซ็กซ์ชั่นจากจำนวนไอเท็มขายน้อยไปมาก

2. สร้างทรีของรูปแบบไอเท็มเซต โดยกำหนดให้โหนดรากเป็นเซตว่าง และระดับที่ถัดมาเป็นรูปแบบในระดับ 1 ไอเท็มเซต จากนั้นสร้างรูปแบบไอเท็มเซตในระดับที่ 2 โดยรูปแบบที่ได้มาจากการจอย (Join) กันของไอเท็มในระดับที่ 1

3. กำหนดค่าเริ่มต้นของระดับไอเท็มเซตที่จะนับค่าสนับสนุนเท่ากับ 2 (k = 2)

4. อ่านทรานเซ็กซ์ชั่นทุกตัวจนครบ

   4.1 ถ้าค่า k น้อยกว่าจำนวนไอเท็มในทรานเซ็กซ์ชั่น

      4.1.1 เพิ่มค่า k ขึ้นอีก 1

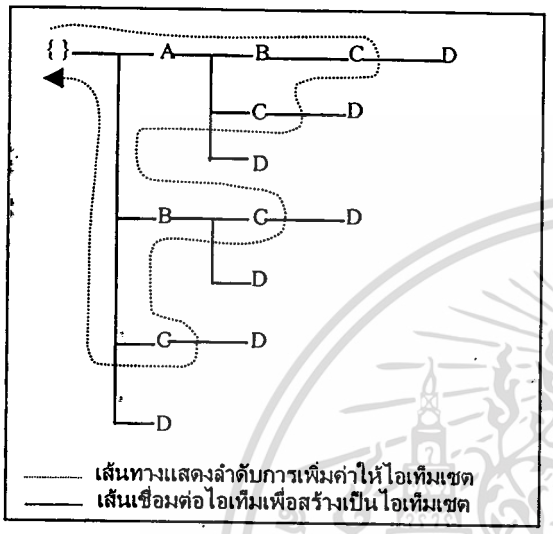
      4.1.2 สร้างรูปแบบไอเท็มเซตที่ระดับถัดไป โดยพิจารณาสร้างรูปแบบจากไอเท็มเซตในระดับก่อนหน้า

   4.2 นับเพิ่มค่ารูปแบบทั้งหมดที่สามารถเกิดขึ้นในทรานเซ็กซ์ชั่นนั้นเช่นทรานเซ็กซ์ชั่นคือ {ABC} รูปแบบที่จะ

เกิดขึ้นได้คือ {A, AB, ABC, AC, B, BC, C} โดยวิธีการเข้าไปนับเพิ่มค่าของรูปแบบนั้นแสดงดังรูปที่ 4

5. หารูปแบบทั้งหมดที่มีค่ามากกว่าค่าสนับสนุนที่กำหนดไว้และแสดงรูปแบบเหล่านั้นออกมาแยกตามระดับไอเท็มเซต

6. จบการทำงาน



รูปที่ 4. แสดงเส้นทางการนับเพิ่มค่าสนับสนุนให้กับรูปแบบไอเท็มเซตในทรี

จากขั้นตอนของการทำงานของวิธีการที่นำเสนอนี้เราจะเห็นได้ว่าวิธีการที่นำเสนอสามารถที่จะเริ่มต้นให้มีการนับรูปแบบที่ระดับต่างๆ ได้เร็วขึ้นกว่าเดิม และยังไม่ต้องอ่านวนหลายรอบเหมือนกระบวนการ Apriori และ DIC เพราะในวิธีการ Apriori นั้นกว่าจะสร้างรูปแบบในระดับถัดไปได้ก็ต้องอ่านข้อมูลจนจบฐานข้อมูลก่อน ส่วนวิธีการ DIC นั้นกว่าจะสร้างรูปแบบในระดับถัดไปได้ก็ต้องอ่านข้อมูลไปเป็นช่วงหนึ่งตามที่ผู้ใช้กำหนดก่อน แต่สำหรับวิธีการที่นำเสนอนี้มีอ่านข้อมูลเพียง 2 รอบก็ได้รูปแบบทั้งหมดแล้ว เนื่องจากเริ่มทำการสร้างรูปแบบในระดับถัดไปเมื่อจำนวนไอเท็มในทรานเซ็กซ์ชั่นเพิ่มขึ้น โดยเหตุผลที่ทำเช่นนี้เพราะพบว่าถ้าในทรานเซ็กซ์ชั่นนั้นมี 2 ไอเท็มย่อมจะไม่มีโอกาสเกิดรูปแบบในระดับ 3 ไอเท็มเซต ดังนั้นเมื่อเราเรียงข้อมูลเข้าตามลำดับของจำนวนไอเท็มในทรานเซ็กซ์ชั่นแล้วพบว่าค่าจำนวนไอเท็มเพิ่มขึ้นเราก็เพียงแต่สร้างรูปแบบในระดับนั้นขึ้นมาเพิ่มเติม และนับเพิ่มค่าสำหรับทุกรูปแบบที่เกิดขึ้นดังตัวอย่างในขั้นตอนที่ 4.2 และเมื่ออ่านครบทุกทรานเซ็กซ์ชั่นแล้วเราก็จะได้ค่าสนับสนุนของทุกรูปแบบทุกระดับไอเท็มเซตครบถ้วนโดยไม่ต้องมีการอ่านย้อนกลับไปอีกครั้งเหมือนกับวิธีการ Apriori และ DIC ดังนั้นจะเห็นได้ว่าจำนวนรอบในการอ่านข้อมูลของวิธีการที่นำเสนอนั้นจะน้อยกว่าวิธีการ Apriori และ DIC ส่วนในรูปที่ 4 นั้นแสดงวิธีการเข้าถึงรูปแบบของไอเท็มเซตโดยกำหนดให้เส้นประแทนเส้นทางลำดับการเพิ่มค่าให้กับไอเท็มเซตซึ่งจากภาพลำดับของการเพิ่มค่าสนับสนุนให้กับไอเท็มเซตก็คือ A, AB, ABC, AC, B, BC, C ส่วนเส้นทึบนั้นแสดงการเชื่อมต่อไอเท็มเข้าด้วยกันเพื่อให้เกิดเป็นไอเท็มเซตต่างขึ้นมา เช่น A---B รูปแบบไอเท็มเซตที่ A คือ A และรูปแบบไอเท็มเซตที่ B คือ AB

5. การทดสอบประสิทธิภาพและความถูกต้องในการทำงาน

5.1 การสร้างชุดข้อมูลสมมุติ

เพื่อเป็นการทดสอบประสิทธิภาพของวิธีการใหม่ที่ได้นำเสนอ ในส่วนนี้จึงได้ทำการทดสอบโดยการเปรียบเทียบกับวิธีการ Apriori และ DIC โดยเครื่องที่นำมาทำการทดสอบเป็นเครื่องที่มีความเร็วในการประมวลผลคำสั่ง 750 MHz MIPS และหน่วยความจำ 128 MB โดยฐานข้อมูลที่นำมาทดสอบนั้นได้มาจากการสร้างขึ้นโดยใช้โปรแกรมของ IBM[1] โดยตัวโปรแกรมจะสร้างรายการการซื้อสินค้าในร้าน ฐานข้อมูลที่สร้างขึ้นมานั้นเราจะกำหนดความแตกต่างในเรื่องต่างๆ ดังในตารางที่ 1

ตารางที่ 1 ตัวแปรที่ใช้ในการสร้างฐานข้อมูลสมมุติ

$\|D\|$	จำนวนทรานเซ็กซ์ชั่น (ขนาดของฐานข้อมูล)
$\|T\|$	ค่าเฉลี่ยจำนวนไอเท็มต่อทรานเซ็กซ์ชั่น
$\|I\|$	ค่าเฉลี่ยขนาดสูงสุดของชุดความสัมพันธ์ของไอเท็ม (จำนวนไอเท็มในรูปแบบ)
L	จำนวนรูปแบบความสัมพันธ์ทั้งหมด
N	จำนวนไอเท็ม

[1] http://www.almaden.ibm.com/cs/quest/syndata.html

ในการเปรียบเทียบประสิทธิภาพนั้นจะพิจารณาในเรื่องของขนาดข้อมูลที่ต่างกันโดยกำหนดให้ขนาดข้อมูลตั้งแต่ 100,000 ทรานเซ็กชั่นขึ้นไป และกำหนดให้ช่วงของค่าสนับสนุนต่ำสุดอยู่ที่ 3.0 % - 0.01 % และในการวัดนั้นเราจะเริ่มจับเวลาในการทำงานของแต่ละกระบวนการตั้งแต่เริ่มต้นสร้างรูปแบบและได้รูปแบบทั้งหมดออกมา โดยโปรแกรมต้นแบบของวิธีการทั้ง 3 แบบนั้นเขียนโดยใช้ภาษา Java

โดยหากพิจารณาจากเหตุการณ์จริงที่เกิดขึ้นนั้นเราอาจจะเห็นได้ว่าลูกค้าแต่ละคนอาจจะมีการซื้อหลายครั้งแต่ในกระบวนการนี้เราจะพิจารณาโดยไม่สนใจว่าเป็นรายการซื้อที่เกิดมาจากลูกค้าคนเดียวกันหรือไม่

5.2 ความถูกต้องของผลลัพธ์

ในขั้นตอนการทำงานขั้นตอนหนึ่งของการทำงานวิธีใหม่นี้เราจะมีการตัดไอเท็มบางไอเท็มออกไปเพื่อเป็นการลดจำนวนไอเท็มที่จะนำไปสร้างเป็นทรีของรูปแบบไอเท็มเซตนั้นไม่มีผลกระทบต่อผลลัพธ์สุดท้าย เพราะว่าหากเราพิจารณาจากวิธีการ Apriori เราก็จะพบว่าไอเท็มเซตที่ระดับ 1 ไอเท็มเซต (มีไอเท็มเพียงตัวเดียว) ตัวใดที่มีค่าสนับสนุนน้อยกว่าค่าสนับสนุนที่กำหนดไว้นั้นย่อมที่จะไม่มีโอกาสเกิดรูปแบบไอเท็มเซตในระดับ 2-ไอเท็มเซต ขึ้นไปได้เลยเพราะว่ารูปแบบไอเท็มเซตในระดับ 2 ไอเท็มเซตนั้นเกิดมาจากการจอย (Join) ไอเท็มเซตในระดับ 1-ไอเท็มเซต ดังนั้นการตัดไอเท็มที่มีค่าสนับสนุนน้อยกว่าค่าที่กำหนดไว้จึงไม่มีผลกระทบกับผลลัพธ์

5.3 การวัดประสิทธิภาพ

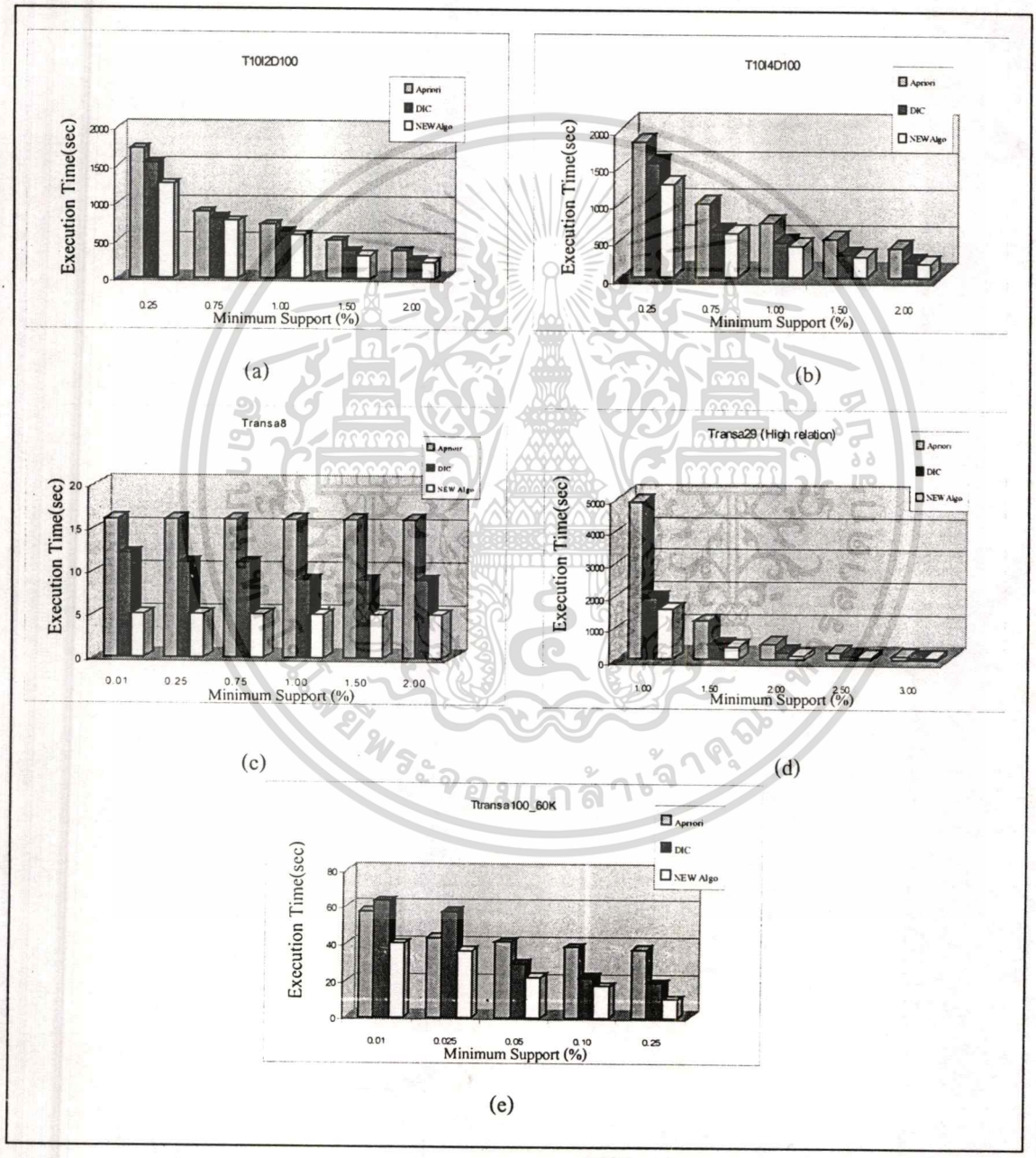
ในการวัดประสิทธิภาพของวิธีการที่ได้พัฒนาขึ้นนั้นพิจาณาได้จากรูปที่ 5 ซึ่งแสดงการเปรียบเทียบประสิทธิภาพการทำงานในเชิงเวลาการประมวลผล โดยพิจารณาเปรียบเทียบระหว่างวิธีการใหม่ที่พัฒนาขึ้นกับวิธีการแบบ Apriori และแบบ DIC ซึ่งเราจะสังเกตเห็นได้ว่าเมื่อมีการลดค่าสนับสนุนต่ำสุดลงเรื่อยๆ จำนวนของรูปแบบที่ได้ก็จะมีจำนวนเพิ่มมากขึ้นไปด้วยดังนั้นเวลาที่ใช้ในการประมวลผลสำหรับค่าสนับสนุนต่ำสุดที่ต่ำๆ จะใช้เวลาในการประมวลนานมากกว่า เมื่อค่าสนับสนุนสูงๆ และเมื่อเรากำหนดค่าสนับสนุนต่ำๆก็จะทำให้วิธีการแบบ Apriori นั้นต้องเข้าไปอ่านฐานข้อมูลหลายรอบมากขึ้นจึงทำให้เวลาในการทำงานเพิ่มมากขึ้นเพราะมีระดับไอเท็มเซตมากขึ้น ส่วน DIC ก็อ่านเพิ่มขึ้นเช่นกันแต่ไม่เท่ากับแบบวิธี Apriori ในขณะที่วิธีการใหม่ก็ยังมีการอ่านข้อมูลเท่าเดิม เมื่อกำหนดค่าสนับสนุนต่ำๆรูปแบบที่เกิดขึ้นก็จะมีมาก ซึ่งทำให้ต้องเข้าไปเพิ่มรูปแบบใหม่ในทรีบ่อยครั้งขึ้นทำให้เวลาการทำงานเพิ่มมากขึ้นด้วย

โดยจากกราฟผลการทดลองที่กำหนดให้มีจำนวนไอเท็มเฉลี่ยต่อทรานเซ็กชั่นเท่ากับ 10 และกำหนดให้จำนวนความสัมพันธ์ของไอเท็มสูงสุดเป็น 2 และ 4 ตามลำดับเมื่อขนาดของฐานข้อมูลเท่ากับ 100,000 ทรานเซ็กชั่นซึ่งผลที่ได้จะเห็นว่าการทำงานของวิธีการแบบใหม่นั้นทำงานได้รวดเร็วกว่าดังในรูปที่ 5(a) และ 5(b) และนอกจากนั้นเรายังได้ลองทดสอบกับรูปแบบข้อมูลที่มีรูปแบบไอเท็มเซตสูงโดยในที่นี้กำหนดให้มีรูปแบบสูงสุดถึงระดับ 7 ไอเท็มเซตและไม่มีการลดจำนวนไอเท็มลงเลยเมื่อผ่านการประมวลผลก่อนเข้าแล้วในแฟ้มข้อมูล Transa8 ดังในรูปที่ 5(c) และไอเท็มที่รูปแบบความสัมพันธ์ของแต่ละไอเท็มสูงในแฟ้มข้อมูล Transa29 ดังในรูปที่ 5(d) ส่วนในรูปที่ 5(e) เป็นผลการทดลองสำหรับรูปแบบข้อมูลที่มีความสัมพันธ์กันน้อยมากก็จะเห็นได้ว่าที่ค่าสนับสนุน 0.01 และ 0.025 นั้นวิธีการของ Apriori นั้นทำงานได้ดีกว่า DIC ซึ่งจากผลการทดลองกับข้อมูลชุดนี้ก็แสดงให้เห็นถึงข้อเสียของวิธีการ DIC แต่เมื่อใช้วิธีการใหม่แล้วจะพบว่าการทำงานดีขึ้นเพราะมีการตัดไอเท็มที่ไม่น่าสนใจออกไปในขณะที่ทำการประมวลผลข้อมูลเข้าก่อนจึงช่วยลดเวลาในการหากฎความสัมพันธ์ลง

6. สรุปผลการทดลอง

จากผลการทดลองจะเห็นว่าวิธีการแบบใหม่ที่ได้นำเสนอนี้มีความเร็วในการทำงานดีกว่าวิธีการแบบ Apriori และวิธีการ DIC เนื่องจากเวลาที่ใช้ในการหารูปแบบน้อยกว่าวิธีการอื่นๆ

ทั้งนี้เพราะวิธีการใหม่ลดการทำงานในส่วนที่ต้องเข้าไปอ่านข้อมูลจากฐานข้อมูลหลายครั้งเพื่อให้ได้รูปแบบทั้งหมดที่เป็นไปได้ แล้วยังมีการลดขนาดของทรีที่มีการสร้างขึ้นเพราะเรามีการจัดการข้อมูลเข้าก่อน จำนวนรอบในการอ่านข้อมูลก็ลดลงเพราะมีการเรียงทรานเซ็กซ์ชั่นใหม่ตามจำนวนไอเท็มในทรานเซ็กซ์ชั่น โดยใช้คุณสมบัติที่ว่าจำนวนไอเท็ม k ไอเท็มในทรานเซ็กซ์ชั่นจะไม่เกิดรูปแบบไอเท็มเซตที่ระดับ k+1 ไอเท็มเซตแต่มีโอกาสเกิดรูปแบบในระดับ 1 ถึง k ไอเท็มเซตเท่านั้น จากคุณสมบัตินี้เราจึงทำการสร้างรูปแบบในระดับถัดไปก็เมื่อจำนวนไอเท็มในทรานเซ็กซ์ชั่นเพิ่มขึ้น ดังนั้นเราจึงอ่านข้อมูลเพียง 1 รอบก็ได้รูปแบบทั้งหมด ถึงแม้ว่าวิธีการใหม่ต้องเสียเวลาในการเพิ่มรูปแบบใหม่ลงในทรีบาง แต่ประสิทธิภาพการทำงานของวิธีการใหม่นี้ก็ยังทำงานได้ดีกว่าวิธีการอื่นๆ



รูปที่ 5. ผลการทดลองเปรียบเทียบเวลาของกระบวนการใหม่กับกระบวนการ Apriori และ DIC

7. เอกสารอ้างอิง

[1] Mohammed Javeed Zaki, Srinivasan Parthasarathy, and Wei Li. "A New Algorithm for Fast discovery of Association rule", Technical Report 651, University of Rochester, July 1997.

[2] Mohammed Javeed Zaki, Srinivasan Parthasarathy, and Wei Li. "A Localized Algorithm for Parallel Association Mining", Department of Computer Science, University of Rochester, 1998.

[3] N. Pasquier, Y. Bastide, R. Taouil and L. Lakhal, "Discovering frequent closed itemsets for association rule", Proc ICDT Conf., January 1999., pp 398-416.

[4] R. Agrawal, T. Imielinski, and A. Swami, "Mining Association rule between set of item in large Database", In ACM SIGMOD Int. Conf. Management of Data , May 1993.

[5] S.Brin, R.Motwani, J.Ullman i S.Tsur. "Dynamic Itemsets Counting and Implication Rule for Market Basket Data", Int. Conf. Management of Data, ACM Press, 1997.

[6] Ulrich Guntzer,Jochen Hipp, "Algorithms for Association Rule Mining – A General survey and comparison" , ACM SIGKDD volumn 2., July 2000 ,pp 58-64.

[7] Xiaodong Chen, Ilias Petrounias, and H. Heathfield, "Discovering Temporal Association Rule: in Temporal Database", Proc. Of IADT'98., pp.312-319.

ISCIT 2001 FMA2-02

A NEW APPROACH OF ASSOCIATION RULE ALGORITHM FOR A MARKET BASKET

Chouvanee Srivisal and Wichian Premchaiswadi

Faculty of Information Technology
King Mongkut's Institute of Technology
Ladkrabang, Bangkok, 10520, Thailand
Tel : (02) 737-2551-4 ext 401 Fax :(02) 737-2500
Email : s2067010@kmitl.ac.th

Faculty of Information Technology
King Mongkut's Institute of Technology
Ladkrabang, Bangkok, 10520, Thailand
Tel : (02) 737-2551-4 ext 401 Fax :(02) 737-2500
Email : wichian@it.kmitl.ac.th

ABSTRACT

In this paper, we propose a new scheme of association rule algorithm for a market basket. The algorithm, which uses one passes over the data, consists of two steps, pre-processing and pattern classification. First step, preprocess all transaction, which reorder transaction by the number of items in transaction. Second step, determining all groups of items, which frequently appear together in transaction. The algorithm is implemented and tested using variable minimum support and variable size of synthetic data. The experimental results are compared with the Apriori and DIC Algorithm. The experimental result shows that proposed the execution time of the algorithm is less than that of the Apriori and DIC algorithm significantly.

1. INTRODUCTION

Data mining or knowledge discovery in databases (KDD) [1,2,3,4,6,7] is the nontrivial extraction of implicit, previously unknown and potential useful information from the data store in large databases. Association rules can be applied to many problem domains including business, marketing management, financial management, engineering, manufacturing and medicine. For example, in market analysis, this kind of rule can be used to analyze customer-buying habits. If customers are buying milk, how likely are they to buy bread and sugar on the same trip to the supermarket? Once discovered, rules can be used for focusing marketing efforts such as sale promotion and product placement. For instance, placing milk, bread and sugar within close proximity may further encourage the sale of these items together within single visits to the store.

Discovering of association rules is one of the central tasks in data mining, whose goal is to identify the frequent itemsets, and then infer association rules among them. Frequent itemsets (also called large itemsets) are itemsets that are frequently bought together in the sane transactions. An association rule is an implication between itemset A and B of the form $X \Rightarrow Y$ where $X \cap Y = \phi$. Two measures for association rules used: support and confident, which indicate the usefulness and the certainly of discovered rules, respectively. For example, the rule "{bread, milk}⇒{eggs}(support = 40 and confident = 70%) " means the fact that there are 40 transactions in the database under consideration where bread, milk and eggs are bought together, and 70% of customers who purchased bread and milk and also bought eggs

In this paper, we propose a new approach of association rule algorithm for a market basket data. The algorithm that uses fewer passes over the data. The first pass for pre-process data, that consist of find 1-itemsets, creates new transactions that have only item in 1-itemsets and reordered transactions by number of items in transaction, and the second pass for finding all frequent itemsets. Our algorithm operates in a bottom-up search direction to count itemsets's support. We present efficiently of new scheme for very small minimum support and variable size of synthetic data.

2. PROBLEM DEFINITION AND COMMON APPROACH OF FINDING ASSOCIATION RULES

The task of mining association rules over basket data was first in [4], which can be formally stated as follows. Let $I = \{i_1, i_2, ..., i_m\}$ be a set of database items. Let D be a database of transactions, where each transaction T, in database, D, has a unique identifier, and contains a set of items, called an itemset such that $T \subseteq I$. Let $X \subseteq I$ be an itemset, A transaction $T \in D$ is said to contain X if and only if. $X \subseteq T$. An itemset that contain k items is called "***k-itemsets***". For an itemset $X \subset I$, the support of X is number of transactions that contain X, which denote it by supp(X). An itemset that has the support greater or equal to a specified minimum support is called a **frequent itemset**. The set of frequent k-itemsets is commonly denoted by L_k

An association rule is a condition implication among itemsets, $A \Rightarrow B$, where itemsets A, $B \subset I$ and $A \cap B = \phi$. The rule $A \Rightarrow B$ hold in database D with ***confidence*** c, which if c% of transaction in D that contain A also contain B, which can be written as the ratio supp($A \cup B$) / supp(A). The rule $A \Rightarrow B$

has ***support*** s if there are s transactions in D contains the itemset A ∪ B, which denote it by supp (A∪B). Then we describe the common approach of finding frequent itemsets.

2.1 Base Algorithm (Apriori Algorithm)

Various algorithms have been proposed to discover the frequent itemset [1,4,5,6]. The Apriori algorithm is one of the most popular algorithms in the mining of association rules in a centralized database

Apriori[1,2,3,4,5,6] uses the downward closure property of itemset support that any subset of a frequent itemset, L_k, must also be frequent. During each iteration of the algorithm only the itemset found to be frequent in previous iterations are used to generate a new candidate set, C_k, to be counted during the current iteration. The candidates are stored in a hash tree to facilitate fast support counting. For counting C_k for each transaction in database, all k-subset of transactions are generated in lexicographical order. Each subset is searched in the hash tree, and the count of the candidate incremented if it matches the subset. When finish algorithm used k pass over the data. The general structure of the algorithm is given in Figure 2.1.

L_1 = {frequent 1-Itemset}
For ($k = 2$, $L_{k-1} \neq \phi$: k++)
  C_k = Set of New Candidates
  For all transactions $t \in D$
   For all k-subset s of t
    If ($s \in C_k$) s.count ++
  $L_k = \{c \in C_k \mid c.\ count \geq minimum\ support\}$
  Set of all frequent item set = $U_k L_k$

Fig 2.1 The Apriori Algorithm

2.2 Dynamic Itemset Counting Algorithm

Dynamic Itemset Counting (DIC), [3,5,6] an algorithm which reduces the number of passes made over the database while keeping the number of itemsets which are counted in any pass. DIC works like a train running over the data with stop at intervals M transaction apart. When the train reaches the end of database, it has made one pass over the data and start at the beginning of the next pass. The "passengers" on the train are itemsets. When an itemset is on the train, then count its occurrence in the transaction that are read. DIC have added flexibility for allowing itemset to get on at any stop as long as they get off at the same stop the next time the train goes around. Therefore, the itemset has seen all transactions in the database. So DIC can start counting an itemset as soon as we suspect it may be necessary to count it instead of waiting until the end of the previous pass.

For example, if we are mining 20,000 transactions and M = 10,000 and assume high itemset = 3. DIC have made about 2.5 passes over the data instead of the 3 passes of Apriori algorithm would make. The number of pass over the data of this show in Figure 3.2 .

3. NEW APPROACH

This new approach uses one pass over the data and consists of two steps, pre-process and fined all frequent itemset. First step, pre-processing, is to reorder transaction by the number item in transactions. The second step read all transaction to find all frequent itemsets. A new approach have added flexibility of allowing itemset to get on when the number of items in transactions increasing and stop counting when end database or read all transaction. This new approach starts counting just 1-itemsets and then quickly add counter 2,3,4,...,k-itemsets. After just number of items in transaction increase. So we can start counting an itemset as soon as the number of item in transaction increase instead of waiting until the end of previous pass. Because of this new approach has lower number of pass over the data was reduce I/O cost so the execution time of the algorithm is less than that of the Apriori and DIC algorithm.

New approach algorithm:

1. Arrange transactions in order of the number of items in transaction.
2. Stipulate root node to be a empty set.
3. Create all items for 1-itemset in the next level of tree.
4. Set the itemset level, *l*, for counting support to be 1
5. While not End-Of-File:
   5.1 if *l* is less than the number of items in transactions
      5.1.1 Increment *l*
      5.1.2 Create l-itemset in the next level of a tree.
   5.2 For each transaction increment the respective counter for all subitemsets that appear in the transaction. The technique of an incremental itemset's counter is shown in Figure 3.1
6. Show all frequent itemsets
7. End.

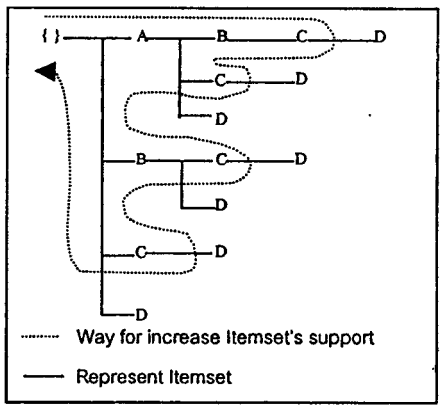


Fig 3.1 Hash tree Data Structure and Technique for increasing counter of a itemset

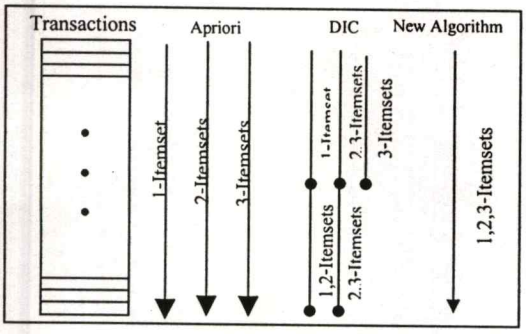


Fig 3.2 The number of passes over the data of each algorithm

3.1 Implemented Data Structure

The implement of new approach requires a data structure, which can keep track of many itemsets. In particular, it must support the following operations:

1. Add a new itemset
2. Maintain a counter for every itemset. When transaction are read, increment the counter of those active itemset, which occur in the transaction.
3. When a itemset does become large, determine what new itemsets should be added.

The data structure used for this is exactly like the hash tree used in Apriori and DIC. Every itemset we are counting has a node associated with it, as do all of its prefixes. The empty itemset is the root node. All the 1-itemsets are attached to the root, and their branches are labeled by the item they represent. All other itemsets are attached to their prefixes containing all but their last item. Figure 3.1 shows a sample tire of this form. The dotted path represents the traversal, which is made through the tries when the transactions ABC are encountered so A, AB, ABC, AC, B, BC and C must be increment and they are in that order.

4. EXPERIMENTAL RESULT

We compare a new approach against Apriori and DIC for decreasing value of minimum support on the different database. Our experiment used a 750 MHz MIPS processor with main memory 128 MB. We used different synthetic database, which mimic transaction in a retailing environment. These databases have been used as benchmark database of many association rule algorithms. We implemented new approach, Apriori and DIC in Java.

4.1 Test data

Varying parameters generated the synthetic data [1,5]. The different database parameters varied in our experiment are shown in Table 4.1. The numbers of transactions increase from 10,000 to 100,000. Our experiments were run on minimum support levels between 0.01% and 3.0%. In the experiment, we observed the execution time of these algorithms.

Table 4.1 The parameters for generate different database

$\|D\|$	Number of transactions
$\|T\|$	Average size of transaction
$\|I\|$	Average size of a maximal potentially frequent itemset
L	The number of maximal potentially frequent itemset
N	Number of items

4.2 Performance

We compare a new approach against Apriori and DIC for decreasing value of minimum support on different databases. As the support decreases, the size and the number of frequent itemsets increase. Apriori and DIC thus have to mark multiple passes over the data and perform poorly.

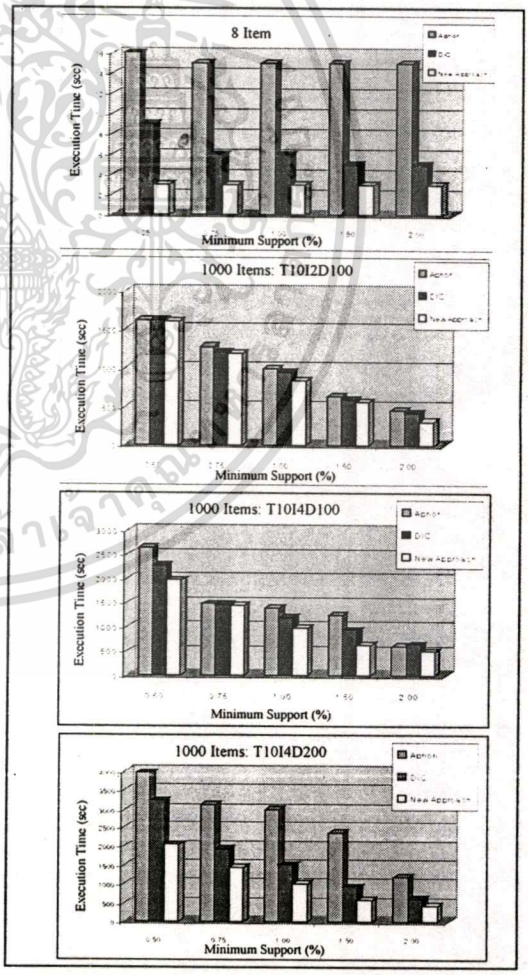


Figure 4.1 Execution Time of any Database

In Figure 4.1 shows the execution time for any database .The execution time of a new approach is less than other algorithms. The number of items in

our experiment is 8 to 1000 items and when the item increase the time is increased too but when minimum support increases the time is decreased. The average time for finding frequent itemsets of each algorithm is shown in Figure 4.2, which the average time of new approach is less than other algorithms, so the execution time will be lower than Apriori and DIC. So we concludes that, the new approach can do, that find all frequent itemset, better than Apriori and DIC because it use lower execution time for finding all frequent itemsets.

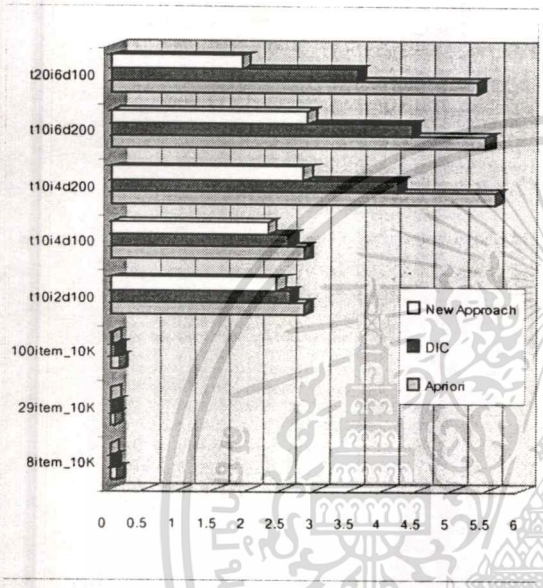


Figure 4.2 Average time for finding one itemset of these algorithms

5. CONCLUSION

In this paper, we proposed a new approach for association mining that can find all frequent itemsets by useing minimum execution time than Apriori and DIC, and evaluate this effectiveness. The proposed algorithms preprocess database, greatly reducing I/O costs. Two main techniques are employed in this algorithm. First reorder all transactions in database by number of items in transaction. Second find frequent itemsets from database by generate next itemset when number of items in transaction increase. Experimental results indicate more than an order of magnitude improvement over previous algorithms, Apriori and DIC.

REFERENCES

[1] Mohammed Javeed Zaki, Srinivasan Parthasarathy, and Wei Li: "A New Algorithm for Fast discovery of Association rule", Technical Report 651, University of Rochester, July 1997.

[2] Mohammed Javeed Zaki, Srinivasan Parthasarathy, and Wei Li: "A Localized Algorithm for Parallel Association Mining", Department of Computer Science, University of Rochester, 1998.

[3] N. Pasquier, Y. Bastide, R. Taouil and L. Lakhal: "Discovering frequent closed itemsets for association rule", Proc ICDT Conf., pp 398-416, January 1999.

[4] R. Agrawal, T. Imielinski, and A. Swami: "Mining Association rule between set of item in large Database", ACM SIGMOD Int. Conf. Management of Data, May 1993.

[5] S.Brin, R.Motwani, J.Ullman i S.Tsur: "Dynamic Itemsets Counting and Implication Rule for Market Basket Data", ACM SIGMOD Int. Conf. Management of Data, 1997.

[6] Ulrich Guntzer,Jochen Hipp: "Algorithms for Association Rule Mining – A General survey and comparison" , ACM SIGKDD, Vol.2, pp 58-64, July 2000.

[7] Xiaodong Chen, Ilias Petrounias, and H. Heathfield, "Discovering Temporal Association Rule: in Temporal Database", Proc. Of IADT'98., pp.312-319.

ประวัติผู้เขียน

นางสาวเชาวนี ศรีวิศาล เกิดเมื่อวันที่ 17 กุมภาพันธ์ 2519 ที่จังหวัดสุราษฎร์ธานี สำเร็จการศึกษา วิทยาศาสตรบัณฑิต (วิทยาการคอมพิวเตอร์) จากมหาวิทยาลัยสงขลานครินทร์ ปีการศึกษา 2540

ปีพุทธศักราช 2541 เข้ารับราชการตำแหน่งอาจารย์ สังกัดภาควิชาคณิตศาสตร์ คณะวิทยาศาสตร์ มหาวิทยาลัยสงขลานครินทร์ วิทยาเขตหาดใหญ่

ปัจจุบันรับทุนโครงการพัฒนาอาจารย์ในระดับปริญญาโทของมหาวิทยาลัยสงขลานครินทร์