รายงานวิจัยฉบับสมบูรณ์

การค้นหากฎความสัมพันธ์แบบเพิ่มขยายในกรณีข้อมูลถูกเพิ่มและลบ

Incremental Association Rule Discovery when Data is Inserted and Deleted

วรพจน์ กรีสุระเดช

ได้รับทุนสนับสนุนงานวิจัยจากงบประมาณเงินรายได้ ประจำปีงบประมาณ 2557
คณะเทคโนโลยีสารสนเทศ
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ชื่อโครงการ การค้นหากฎความสัมพันธ์แบบเพิ่มขยายในกรณีข้อมูลถูกเพิ่มและลบ
แหล่งเงิน แหล่งเงินรายได้
ประจำปีงบประมาณ 2557 **จำนวนเงินที่ได้รับการสนับสนุน** 50,000 บาท
ระยะเวลาทำการวิจัย 1 **ตั้งแต่** ตุลาคม 2556 **ถึง** กันยายน 2557
ชื่อ-สกุล หัวหน้าโครงการ รองศาสตราจารย์ ดร.วรพจน์ กรีสุระเดช
คณะเทคโนโลยีสารสนเทศ
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

บทคัดย่อ

งานวิจัยนี้ได้นำเสนออัลกอริทึมสำหรับการค้นหากฎความสัมพันธ์แบบเพิ่มขยายเมื่อมีการเพิ่มข้อมูลชุดใหม่เข้ามาและมีข้อมูลเก่าถูกลบออกจากฐานข้อมูล เนื่องจากเมื่อฐานข้อมูลมีการเปลี่ยนแปลงจะมีผลต่อกฎความสัมพันธ์ที่หาไว้แล้วในฐานข้อมูลเดิม โดยฟรีเควนท์ไอเทมเซตที่ได้ทำการค้นหาจากฐานข้อมูลเดิมอาจไม่เป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลอัพเดท ในขณะเดียวกันอินฟรีเควนท์ไอเทมเซตในฐานข้อมูลเดิมก็อาจกลายเป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลอัพเดทได้ แนวคิดหลักของอัลกอริทึมที่นำเสนอในงานวิจัยนี้คือการเก็บทั้งฟรีเควนท์ไอเทมเซตและไอเทมเซตที่มีแนวโน้มเป็นฟรีเควนท์ไอเทมเซต (Promising frequent itemsets: PF) เมื่อมีการเปลี่ยนแปลงของรายการในฐานข้อมูล โดยการนำหลักทางสถิติของเบอร์นูลลี (Bernoulli trials) มาใช้ในการคาดคะเนไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ไอเทมเซต เพื่อลดจำนวนไอเทมเซตที่จะนำไปสแกนในฐานข้อมูลเดิม จากผลการทดลองพบว่าอัลกอริทึมที่นำเสนอในงานวิจัยนี้สามารถทำงานได้อย่างถูกต้องและมีประสิทธิภาพ

คำสำคัญ : การค้นหากฎความสัมพันธ์แบบเพิ่มขยาย การค้นหากฎความสัมพันธ์ การทำเหมืองข้อมูล

Research Title: Incremental Association Rule Discovery when Data is Inserted and Deleted
Researcher: Associate Professor Dr.Worapoj Kreesuradej
Faculty: Faculty of Information Technology

ABSTRACT

The maintenance of association rules for dynamic database is an important problem because the updates may not only invalidate some existing rules but also make other rules relevant. This paper proposes an Incremental Association Rule Discovery Algorithm which can efficiently handle in case of insertion as well as deletion simultaneously. Basically, the proposed algorithm maintains the support counts of frequent itemsets and promising frequent itemsets, i.e., infrequent itemsets that promise to be frequent in the future, in an original database. Promising frequent itemsets, which are obtained by using the principle of Bernoulli trials, can help to reduce a number of times to rescan the original database. The support counts of new candidate itemsets are approximated by using the principle of maximum possible value. The experimental results show that the execution time of the proposed algorithm is faster than that of Apriori, FUP2, and Pre-large algorithm.

Keywords: Association Rule Discovery, Data Mining, Incremental Association Rule Discovery, Rules Maintenance

กิตติกรรมประกาศ

งานวิจัยเรื่องการค้นหากฎความสัมพันธ์แบบเพิ่มขยายในกรณีข้อมูลถูกเพิ่มและลบ ผู้วิจัยได้ทำการศึกษาและพัฒนาอัลกอริทึมเพื่อช่วยในการเพิ่มประสิทธิภาพในการค้นหากฎความสัมพันธ์ในกรณีที่มีข้อมูลใหม่เพิ่มเข้ามาและมีข้อมูลเก่าถูกลบออกจากฐานข้อมูลเดิม งานวิจัยฉบับนี้สำเร็จได้ด้วยดี โดยการวิจัยครั้งนี้ได้รับทุนสนับสนุนการวิจัยจากสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง จากแหล่งทุนประเภทรายได้ ประจำปีงบประมาณ พ.ศ. 2557 ผู้วิจัยต้องขอขอบพระคุณเป็นอย่างสูงมา ณ โอกาสนี้

รองศาสตราจารย์ ดร.วรพจน์ กรีสุระเดช

สารบัญ

สารบัญ (ต่อ)

สารบัญตาราง

สารบัญรูป

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของปัญหา

เทคโนโลยีสารสนเทศมีความสำคัญต่อการดำเนินงานขององค์กรทุกองค์กร โดยเฉพาะอย่างยิ่งปัจจุบันโลกมีการเปลี่ยนแปลงอยู่ตลอดเวลา ไม่ว่าจะเป็นการเปลี่ยนแปลงทางเศรษฐกิจ สังคม ประชากร กฎหมาย เทคโนโลยี และอื่นๆ ทำให้มีการแข่งขันระหว่างองค์กรสูงขึ้น ดังนั้นองค์กรใดที่สามารถบริหารงานได้อย่างมีประสิทธิภาพ สามารถเข้าถึงหรือใช้งานข้อมูลได้อย่างรวดเร็วย่อมจะทำให้องค์กรดังกล่าวมีความได้เปรียบในการแข่งขันสูง

ข้อมูล จัดเป็นทรัพยากรสำคัญและเป็นองค์ประกอบของระบบสารสนเทศที่ทุกองค์กรต้องมี และมีความจำเป็นต้องใช้ประโยชน์จากข้อมูลให้ได้ประสิทธิภาพสูงสุด เพื่อช่วยให้การตัดสินใจดำเนินงานขององค์กรเป็นไปอย่างมีประสิทธิภาพ แต่ในปัจจุบันจะเห็นได้ว่า แม้เทคโนโลยีสารสนเทศโดยเฉพาะอย่างยิ่งเทคโนโลยีการบันทึกข้อมูลมีความเจริญก้าวหน้ามากมายเพียงใด แต่อัตราการใช้ประโยชน์จากข้อมูลนั้นยังคงมีน้อยมาก ยังคงมีความรู้ (Knowledge) อีกมากมายที่ถูกซ่อนอยู่ในข้อมูลดังกล่าว แต่เรายังไม่ได้นำออกมาใช้

การทำเหมืองข้อมูล (data mining) หรือบางครั้งเรียกว่า การค้นหาความรู้ในฐานข้อมูล (Knowledge Discovery in Database: KDD) เป็นกระบวนการที่ใช้ในการค้นหารูปแบบ หรือความสัมพันธ์ที่ซ่อนอยู่ในข้อมูลจำนวนมหาศาลโดยอัตโนมัติ ซึ่งในปัจจุบันมีองค์กรหลายๆ องค์กรต่างพยายามนำเอาการทำเหมืองข้อมูลเข้าไปช่วยในการสร้างความได้เปรียบให้กับองค์กร เช่น การนำเอาเทคนิคของการทำเหมืองข้อมูลไปช่วยในการจัดกลุ่มลูกค้าเงินกู้ชั้นดีของธนาคาร หรือการจำแนกกลุ่มลูกค้าที่มีความสามารถในการซื้อสินค้าราคาสูงจำพวกบ้านรถและรถยนต์ เป็นต้น

การค้นหากฎความสัมพันธ์ (Association rules) เป็นเทคนิคหนึ่งในการทำเหมืองข้อมูลที่ใช้ในการวิเคราะห์ความสัมพันธ์ของข้อมูลระหว่างรายการ โดยจะทำการค้นหาและดึงรูปแบบของข้อมูลระหว่างรายการต่างๆ ที่ถูกจัดเก็บไว้ในฐานข้อมูล และสร้างให้อยู่ในรูปแบบของกฎความสัมพันธ์ เช่น การหากฎความสัมพันธ์จากการซื้อสินค้าของลูกค้า แล้วนำกฎที่ได้ไปช่วยในการหากลยุทธ์ส่งเสริมการขายเพื่อเพิ่มยอดขายให้กับองค์กร

อัลกอริทึมสำหรับค้นหากฎความสัมพันธ์ที่ได้รับความนิยมคืออัลกอริทึมอะพริโอริ [1] ซึ่งทำการค้นหากฎความสัมพันธ์ของฐานข้อมูลโดยเริ่มจากการหาเซตของไอเทมที่เกิดขึ้นร่วมกัน โดยจำนวนรายการที่เกิดขึ้นร่วมกันนั้นต้องมีมากกว่าหรือเท่ากับค่าที่ใช้ในการวัดความสัมพันธ์ที่ได้กำหนดไว้ เรียกค่าที่ใช้วัดความสัมพันธ์นี้ว่าค่าสนับสนุนน้อยที่สุด (Minimum support: s%) และเรียกเซตของไอเทมที่ผ่านเกณฑ์ค่าสนับสนุนน้อยที่สุดเหล่านี้ว่าฟรีเควนท์ k-ไอเทมเซต (Frequent k-itemset) เมื่อ k คือจำนวนไอเทมที่ประกอบขึ้นเป็นไอเทมเซต (k=1,2,...,n) จากนั้นฟรีเควนท์ k-ไอ

เทมเซต (k≥2) ที่ได้ จะถูกนำมาสร้างเป็นกฎความสัมพันธ์ในรูปของ IF...THEN rules เช่น IF diaper THEN beer โดยมีค่าความเชื่อมั่นน้อยที่สุด (minimum confidence: c%) เป็นค่าที่ใช้ตรวจสอบว่ากฎใดควรจัดอยู่ในกฎที่น่าสนใจ

ในการหากฎความสัมพันธ์โดยทั่วไปมักจะดำเนินการโดยตั้งสมมติฐานให้ฐานข้อมูลไม่มีการเปลี่ยนแปลง (Static database) อย่างไรก็ตาม รายการที่จัดเก็บอยู่ในฐานข้อมูลนั้นมักมีการปรับปรุงเพื่อให้ทันสมัยอยู่ตลอดเวลา (Dynamic database) ซึ่งการปรับปรุงนี้มีทั้งการเพิ่ม ลบ และแก้ไขรายการข้อมูล ส่งผลให้เกิดการเปลี่ยนแปลงกฎความสัมพันธ์ที่ได้ค้นหาไว้แล้ว โดยอาจทำให้กฎที่มีอยู่ไม่มีความถูกต้อง ซึ่งในการค้นหาความสัมพันธ์จากฐานข้อมูลที่ได้รับการปรับปรุงใหม่ (Updated database) โดยไม่นำความรู้ที่ได้จากการค้นหากฎความสัมพันธ์ของฐานข้อมูลเดิม (Original database) มาใช้นั้น จะสิ้นเปลืองเวลาและทรัพยากรในการประมวลผลเป็นอย่างมาก เนื่องจากต้องทำการสแกนรายการข้อมูลทั้งหมดที่อยู่ฐานข้อมูลปรับปรุง ทั้งๆ ที่ข้อมูลส่วนใหญ่คือรายการในฐานข้อมูลเดิมที่เคยถูกสแกนไปแล้ว ดังนั้น จึงมีการนำกระบวนการค้นหากฎความสัมพันธ์แบบเพิ่มขยาย (Incremental association rules mining) มาใช้ในการปรับปรุงกฎความสัมพันธ์เมื่อมีการเปลี่ยนแปลงของรายการในฐานข้อมูล

จากปัญหาดังกล่าวจึงเป็นที่มาของการวิจัยทางด้านการค้นหากฎความสัมพันธ์แบบเพิ่มขยาย (Incremental Mining on Association Rules) ซึ่งงานวิจัยนี้จะทำการศึกษาและพัฒนาเกี่ยวกับการเพิ่มขยายการค้นหากฎความสัมพันธ์เมื่อข้อมูลถูกเพิ่มและลบ เพื่อลดระยะเวลาในการประมวลผลจากฐานข้อมูลเดิม

1.2 ความมุ่งหมายและวัตถุประสงค์ของการศึกษา

1. ศึกษาค้นคว้าอัลกอริทึมที่เกี่ยวข้องกับการค้นหากฎความสัมพันธ์แบบเพิ่มขยายในกรณีข้อมูลถูกเพิ่มและลบ

2. ออกแบบอัลกอริทึมการค้นหากฎความสัมพันธ์แบบเพิ่มขยายในกรณีที่ข้อมูลถูกเพิ่มและลบ

1.3 ทฤษฎีและแนวคิดที่ใช้ในการวิจัย

ทฤษฎีและแนวคิดต่างๆ ที่นำมาประยุกต์ใช้ในการวิจัยเกี่ยวกับการหากฎความสัมพันธ์แบบเพิ่มขยายเมื่อข้อมูลถูกเพิ่มและลบ ประกอบด้วย

1. การค้นหากฎความสัมพันธ์ (Association rule mining) เป็นทฤษฎีและแนวคิดเกี่ยวกับการหากฎความสัมพันธ์จากฐานข้อมูลขนาดใหญ่ โดยข้อมูลที่จะถูกนำมาสร้างกฎความสัมพันธ์จะต้องต้องผ่านเกณฑ์ที่ใช้ในการวัดความสัมพันธ์ที่ได้กำหนดไว้

2. การค้นหากฎความสัมพันธ์แบบเพิ่มขยาย (Incremental association rule mining) เป็นทฤษฎีและแนวคิดที่นำเสนออัลกอริทึมในการหากฎความสัมพันธ์เมื่อมีการเปลี่ยนแปลงของ

รายการในฐานข้อมูล โดยนำความรู้ที่ได้จากการค้นหาความสัมพันธ์ของฐานข้อมูลเดิมมาใช้ เพื่อให้ได้อัลกอริทึมที่ทำงานได้อย่างมีประสิทธิภาพมากยิ่งขึ้น

3. ทฤษฎีหลักการความน่าจะเป็นของเบอร์นูลลี (Bernoulli trials) เป็นทฤษฎีในการคำนวณหาค่าความน่าจะเป็นจากการทดลองทางสถิติ

1.4 ขอบเขตของการวิจัย

1. ศึกษาอัลกอริทึมการค้นหากฎความสัมพันธ์แบบเพิ่มขยายในกรณีข้อมูลถูกเพิ่มและลบ
2. ออกแบบและทดลองอัลกอริทึมการค้นหากฎความสัมพันธ์แบบเพิ่มขยายในกรณีข้อมูลถูกเพิ่มและลบ ภายใต้เงื่อนไขที่ค่าสนับสนุนขั้นต่ำมีค่าคงที่

1.5 ขั้นตอนของการศึกษา

งานวิจัยเรื่องการกฎความสัมพันธ์แบบเพิ่มขยายเมื่อข้อมูลถูกเพิ่มและลบ มีขั้นตอนของการศึกษาดังนี้

1.5.1 ศึกษาแนวคิด ทฤษฎีและงานวิจัยที่เกี่ยวข้องกับงานวิจัย จากตำรา และเอกสารบทความต่างๆ

1.5.2 กำหนดหัวข้อ วัตถุประสงค์ และขอบเขตการทำงานของงานวิจัย

1.5.3 ออกแบบอัลกอริทึมใหม่ที่พัฒนาประสิทธิภาพการทำงานการเพิ่มขยายกฎความสัมพันธ์

1.5.4 พัฒนาโปรแกรมการทำงานของอัลกอริทึม ด้วยซอฟต์แวร์แมทแลป (MATLAB) รวมถึงการทดสอบการทำงานและแก้ไขข้อผิดพลาดของโปรแกรม

1.5.5 ทดลองการทำงานของอัลกอริทึมด้วยชุดข้อมูลสังเคราะห์ที่สร้างขึ้น เพื่อวัดประสิทธิภาพการทำงานของอัลกอริทึม

1.5.6 รวบรวมผลการทดลองจากการทำงานของอัลกอริทึม วิเคราะห์และสรุปผลการทดลอง

1.5.7 เรียบเรียงและจัดทำเล่มรายงาน

บทที่ 2
ทฤษฎีพื้นฐานที่ใช้ในการวิจัยและงานวิจัยที่เกี่ยวข้อง

2.1 การค้นหากฎความสัมพันธ์

การค้นหากฎความสัมพันธ์ (Association rules) เป็นเทคนิคที่ใช้ในการวิเคราะห์ความสัมพันธ์ของข้อมูลระหว่างรายการที่ปรากฏอยู่ในฐานข้อมูลขนาดใหญ่ โดยจะทำการค้นหาและดึงรูปแบบของข้อมูลระหว่างรายการต่างๆ ทั้งหมดที่ได้จัดเก็บไว้ในฐานข้อมูล และสร้างให้อยู่ในรูปแบบของกฎความสัมพันธ์ ซึ่งพื้นฐานการทำงานของการค้นหากฎความสัมพันธ์คือการนับความถี่หรือจำนวนครั้งที่ไอเทมเกิดขึ้นร่วมกัน ซึ่งข้อมูลของแต่ละไอเทมจะมีค่าเป็นแบบไบนารี (Binary) เช่น ลูกค้า “ซื้อ/ไม่ซื้อ” สินค้า โดยจะไม่นำจำนวนสินค้าที่ซื้อในแต่ละรายการมาพิจารณา กำหนดให้ D เป็นเซตของรายการในฐานข้อมูล, T เป็นรายการ (Transaction) ซึ่งประกอบด้วยเซตของไอเทม I โดยรายการ T แต่ละรายการจะสัมพันธ์กับตัวระบุรายการ (Transaction Identifier: TID) และ I เป็นเซตของไอเทม I โดย $I = I_1, I_2, ..., I_m$

ไอเทมที่ปรากฏอยู่ในแต่ละรายการของฐานข้อมูลจะถูกนำมาหาความสัมพันธ์และสร้างเป็นกฎความสัมพันธ์โดยจะแสดงอยู่ในรูปของ IF...THEN rule ไอเทมเซตที่จะนำมาสร้างเป็นกฎความสัมพันธ์ได้จะต้องมีจำนวนของข้อมูลที่เกิดขึ้นมากกว่าหรือเท่ากับตัววัด 2 ตัว คือ ค่าสนับสนุนน้อยที่สุด (Minimum support) และค่าความเชื่อมั่นน้อยที่สุด (Minimum confidence)

สมมติกำหนดให้ A และ B เป็นไอเทมที่เกิดขึ้นในรายการ ($A \subseteq T$ และ $B \subseteq T$) จะนำ A และ B มาสร้างเป็นกฎความสัมพันธ์ได้ก็ต่อเมื่อมีจำนวนรายการที่มี A และ B เกิดขึ้นร่วมกันหรือค่าสนับสนุนของไอเทมเซต {AB} มากกว่าหรือเท่ากับค่าสนับสนุนน้อยที่สุด เปอร์เซ็นต์ของรายการในฐานข้อมูลที่มี A และ B เกิดขึ้นร่วมกันก็คือค่าความน่าจะเป็นที่จะเกิดไอเทม A และ B พร้อมกัน ($P(A \cup B)$) นั่นเอง ซึ่งกฎความสัมพันธ์ที่ได้จะแสดงในรูปแบบของ $A \Rightarrow B$ โดย $A \subset I$, $B \subset I$ และ $A \cap B = \varnothing$ กฎ $A \Rightarrow B$ จะมีค่าความเชื่อมั่น c เมื่อ c เป็นเปอร์เซ็นต์ของรายการในฐานข้อมูลที่มี A แล้วจะมี B ด้วย ซึ่งเป็นความน่าจะเป็นแบบมีเงื่อนไข ($P(B|A)$) แนวคิดเกี่ยวกับค่าสนับสนุน (Support) และค่าความเชื่อมั่น (Confidence) สำหรับการหากฎความสัมพันธ์สามารถแสดงให้อยู่ในรูปของความน่าจะเป็นได้ดังนี้

$$\text{Support } (A \Rightarrow B) = P(A \cup B)$$

$$\text{Confidence } (A \Rightarrow B) = P(B|A)$$

นิยามและความหมายของคำต่างๆ ที่ใช้ในการค้นหากฎความสัมพันธ์ ได้แก่

1) ไอเทม (Item) คือข้อมูลแต่ละตัวที่ใช้ในการค้นหากฎความสัมพันธ์ เช่น bread, milk, beer, diaper เป็นต้น

2) ไอเทมเซต (Itemset) คือ ความสัมพันธ์ของข้อมูลที่หาได้ ไอเทมเซตจะประกอบด้วยไอเทมที่มีความยาวแตกต่างกัน โดยทั่วไปจะใช้ k แทนขนาดความยาวไอเทมเซต (k = 1, 2, 3,..., n) เรียกว่า k-ไอเทมเซต ตัวอย่างเช่น 2-ไอเทมเซต หมายถึง ไอเทมเซตที่ประกอบด้วยสมาชิกของไอเทม 2 ตัว เช่น {bread, milk}, {bread, diaper} เป็นต้น

3) แคนดิเดตไอเทมเซตหรือไอเทมเซตตัวเลือก (Candidate itemset: C) คือ ชุดของไอเทมเซตที่ได้จากการเชื่อมกัน (Join) ของฟรีเควนท์ไอเทมเซตในระดับก่อนหน้านี้ เช่น $C_3 = L_2 \times L_2$

4) ค่าสนับสนุน (Support value) เป็นค่าแสดงความสัมพันธ์ระหว่างจำนวนของไอเทมเซตที่ปรากฏในรายการทั้งหมดในฐานข้อมูล

$$\text{Support} = \frac{n}{N} \tag{2.1}$$

เมื่อ n คือ จำนวนครั้งที่ไอเทมเซตนั้นๆ ปรากฏในรายการของฐานข้อมูล และ N คือ จำนวนรายการทั้งหมดในฐานข้อมูล

5) ค่าสนับสนุนน้อยที่สุด (Minimum support) คือ ค่าสนับสนุนที่น้อยที่สุดหรือค่าสนับสนุนขั้นต่ำที่ทำให้ความสัมพันธ์ที่ได้มานั้นยังมีความน่าสนใจ ซึ่งเป็นค่าที่ถูกกำหนดโดยผู้ใช้

6) ค่าความเชื่อมั่น (Confidence) เป็นค่าแสดงความเข้มแข็งของกฎความสัมพันธ์ที่เกิดขึ้น

$$\text{Confidence } (A \Rightarrow B) = P(B|A) \tag{2.2}$$

$$P(A|B) = \frac{P(A \cap B)}{P(A)}$$

เมื่อ P(B|A) คือความน่าจะเป็นที่ B จะเกิดขึ้นเมื่อ A เกิดขึ้นแล้ว และ P(A) คือความน่าจะเป็นของไอเทม A

7) ค่าความเชื่อมั่นน้อยที่สุด (Minimum confidence) เป็นค่าที่ใช้ทดสอบว่ากฎความสัมพันธ์ใดจะเป็นกฎที่มีความน่าสนใจหรือเป็นกฎที่เข้มแข็ง (Strong rule)

8) ฟรีเควนท์ไอเทมเซต (Frequent itemset: F) หรือ ลาร์จไอเทมเซต (Large itemset: L) คือชุดของไอเทมเซตที่มีความน่าสนใจที่จะถูกนำไปสร้างกฎความสัมพันธ์ ซึ่งหาได้จากไอเทมเซตที่มีค่าสนับสนุนของไอเทมเซตที่เกิดร่วมกันมากกว่าหรือเท่ากับค่าสนับสนุนน้อยที่สุดที่กำหนดไว้

9) อินฟรีเควนท์ไอเทมเซต (Infrequent itemset) หรือสมอลล์ไอเทมเซต (Small itemset: S) คือชุดของไอเทมเซตที่มีค่าสนับสนุนของไอเทมเซตที่เกิดร่วมกันน้อยกว่าค่าสนับสนุนน้อยที่สุดที่กำหนดไว้

2.1.1 ปัญหาของการค้นหากฎความสัมพันธ์ ประกอบด้วย 2 ปัญหาหลักๆ คือ

2.1.1.1 การหาฟรีเควนท์ไอเทมเซตทั้งหมดที่ปรากฏในฐานข้อมูล โดยค่าสนับสนุนของฟรีเควนท์ไอเทมเซตเหล่านี้จะต้องมีค่ามากกว่าหรือเท่ากับค่าสนับสนุนน้อยที่สุดที่ผู้ใช้กำหนด ขั้นตอนนี้จัดเป็นขั้นตอนที่สำคัญมากในการค้นหากฎความสัมพันธ์ ถ้าข้อมูลที่นำมาวิเคราะห์มีจำนวนไอเทมทั้งหมด n ไอเทม จำนวนไอเทมเซตที่มีโอกาสที่จะถูกสร้างขึ้นมาจะมีมากถึง 2^n-1 ไอเทมเซต การพัฒนาอัลกอริทึมเพื่อหาฟรีเควนท์ไอเทมเซตที่มีประสิทธิภาพจัดเป็นงานที่ท้าทาย ดังนั้นงานวิจัยส่วนใหญ่จึงมุ่งเน้นการสร้างอัลกอริทึมเพื่อพัฒนากระบวนการทำงานในส่วนนี้ โดย [2] ได้แบ่งเทคนิคในการหาฟรีเควนท์ไอเทมเซตออกเป็น 2 กลุ่มคือ 1) การหาฟรีเควนท์ไอเทมเซตด้วยการ

สร้างแคนดิเดต 2) การหาฟรีเควนท์ไอเทมเซตโดยไม่สร้างแคนดิเดต ซึ่งจะกล่าวรายละเอียดในหัวข้อถัดไป

2.1.1.2 การสร้างกฎความสัมพันธ์ โดยการนำฟรีเควนท์ไอเทมเซตตั้งแต่ 2 ไอเทมเซตขึ้นไปที่ได้จากข้อ 2.1.1.1 มาสร้างเป็นกฎความสัมพันธ์ ในส่วนนี้กฎความสัมพันธ์ที่ได้จะต้องมีค่าความเชื่อมั่นมากกว่าค่าความเชื่อมั่นน้อยที่สุดที่ผู้ใช้กำหนด

2.2 การหาฟรีเควนท์ไอเทมเซต

2.2.1 การหาฟรีเควนท์ไอเทมเซตด้วยการสร้างแคนดิเดต

อัลกอริทึมในกลุ่มนี้จะเริ่มจากสร้างลิสต์ (List) ของแคนดิเดตไอเทมเซต และนำค่าสนับสนุนของแคนดิเดตไอเทมเซตเหล่านี้ไปตรวจสอบกับค่าสนับสนุนน้อยที่สุดที่ผู้ใช้กำหนดไว้ แคนดิเดตไอเทมเซตที่มีค่าสนับสนุนมากกว่าหรือเท่ากับค่าสนับสนุนน้อยที่สุดจะจัดเป็นฟรีเควนท์ไอเทมเซต ซึ่งขั้นตอนทั้งหมดที่กล่าวมานี้จำเป็นต้องอาศัยการเข้าถึงข้อมูลหลายครั้งและใช้เวลาในการประมวลผลนาน จึงมีงานวิจัยหลายชิ้นพยายามหาวิธีในการลดจำนวนไอเทมเซตที่จะต้องสแกนในแต่ละรอบ ซึ่งอัลกอริทึมในกลุ่มนี้ที่มีชื่อเสียงมากที่สุดก็คืออัลกอริทึมอะพริโอรินั่นเอง

2.2.1.1 อัลกอริทึมอะพริโอริ (Apriori algorithms)

อัลกอริทึมอะพริโอริ [1] เป็นอัลกอริทึมแรกที่นำเสนอวิธีการในการลดจำนวนไอเทมเซตที่จะถูกสแกนในรอบถัดไป ด้วยแนวคิดที่ว่า "ถ้าสับเซต (k-1) ใดๆ ของแคนดิเดตไอเทมเซต ไม่ได้เป็นสมาชิก L_k-1 แคนดิเดตไอเทมเซตนั้นๆ จะไม่สามารถเป็นฟรีเควนท์ไอเทมเซตในระดับต่อไปได้ ดังนั้นจะลบแคนดิเดตไอเทมเซตนั้นๆ ออกไป" นั่นคือ ไอเทมเซตที่มีค่าสนับสนุนต่ำกว่าค่าสนับสนุนน้อยที่สุดจะถูกกำจัดออกไป รวมถึงซูเปอร์เซต (Superset) ของไอเทมเซตนั้นๆ ด้วย ทำให้จำนวนแคนดิเดตไอเทมเซตลดขนาดลง คุณสมบัตินี้มีชื่อเรียกว่า Anti-monotone

อัลกอริทึมอะพริโอริจะทำการค้นหาข้อมูลที่เป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลด้วยการสแกนข้อมูลแต่ละรายการในฐานข้อมูลผ่านการวนรอบซ้ำ (Iterations) และในการวนรอบซ้ำแต่ละครั้งจะค้นหาจำนวนสมาชิกของฟรีเควนท์ไอเทมเซตเพิ่มขึ้นทีละ 1 ระดับ โดยอาศัยความรู้ที่ได้จากขั้นตอนก่อนหน้า เช่น หา C_2 จาก L_1 เรียกการค้นหาฟรีเควนท์ไอเทมเซตที่มีการเพิ่มสมาชิกไปในแต่ละระดับแบบนี้ว่าการค้นหาแบบทีละระดับ (Levelwise search) ซึ่งประกอบด้วย 2 ขั้นตอน คือ

1) ขั้นตอนการเชื่อม (Join step)

เริ่มจากการสร้างและนับแคนดิเดต 1-ไอเทมเซต (C_1) โดยต้องมีการเรียงลำดับตัวอักษรของข้อมูลในรายการจากน้อยไปมาก (Lexicographic order) จากนั้นนำค่าสนับสนุนของ C_1 มาพิจารณาเพื่อหาฟรีเควนท์ 1-ไอเทมเซต (L_1) และนำ L_1 มาเชื่อมกันเองเพื่อสร้าง C_2 ทำซ้ำเช่นนี้ไปเรื่อยๆ จนไม่สามารถสร้างแคนดิเดตไอเทมเซตได้อีก ในกรณีที่ไม่ใช่ 1-ไอเทมเซต L_{k-1} ที่นำมาเชื่อมกันนั้นจะต้องมีสมาชิกทุกตัวก่อนตัวสุดท้ายเหมือนกัน เช่น L_3 = {ABC, ABD, AEF} จะได้ C_4 = {ABCD}

2) ขั้นตอนการตัด (Prune step)

เป็นขั้นตอนที่ช่วยลดจำนวนไอเทมเซตที่ต้องนำไปสแกนในฐานข้อมูลแต่ละรอบโดยกำหนดเงื่อนไขไว้ว่า "ถ้าสับเซต (k-1) ใดๆ ของแคนดิเดตไอเทมเซต (C_k) ไม่ได้เป็นสมาชิกของฟรีเควนท์ L_{k-1} แคนดิเดตไอเทมเซตนั้นๆ จะไม่สามารถเป็นฟรีเควนท์ไอเทมเซตในระดับต่อไปได้" ดังนั้น ในขั้นตอนนี้ C_k จะถูกแตกเป็นสับเซตย่อยที่ประกอบด้วย สมาชิก k-1 ตัว แล้วพิจารณาว่าสับเซตย่อยที่แตกมานั้นว่าทุกตัวต้องเป็น L_{k-1} ถ้าพบว่าตัวใดตัวหนึ่งของสับเซตย่อยไม่เป็น L_{k-1} จะทำการลบ C_k นั้น

ออกไปเนื่องจากไอเทมเซตนั้นๆ จะไม่สามารถกลายมาเป็นฟรีเควนท์ไอเทมเซตได้ เช่น L_2 = {AB AD, AE, BE} จะได้ C_3 = {ABD, ABE} ในกรณีนี้ ABD ซึ่งมีสับเซตเป็น {AB, AD, BD} จะถูกตัดออก เนื่องจาก BD ไม่ได้เป็น L_2 ขั้นตอนและตัวอย่างของการหาฟรีเคนท์ไอเทมเซตด้วยอัลกอริทึมอะพริโอริแสดงได้ดังรูปที่ 2.1 และ 2.2

Algorithm 1: Apriori

Input: D: database over the set of items $\jmath$,
Output: F: the set of frequent itemsets

1: $k = 1$; $C_k = \jmath$
2: **while** $C_k \neq 0$ **do**
3: support_count(D, C_k)
4: **for all** candidates $c \in C_k$ **do**
5: **if** $c.support \geq minsup$ **then**
6: $F_k = c$
7: **end if**
8: **end for**
9: $C_{k+1} = candidate\_generation(F_k)$
10: $k = k + 1$
11: **end while**
12: $F = \cup_{j=1}^{k} F_j$

รูปที่ 2.1 ขั้นตอนในการหาฟรีเควนท์ไอเทมเซตของอัลกอริทึมอะพริโอริ

D

Transaction
ACD
BCE
ABCE
BE

1-pass →

C_1

1- itemset	Count
A	2
B	3
C	3
D	1
E	3

→

L_1

Large 1-itemset	Count
A	2
B	3
C	3
E	3

C_2

2- itemset
AB
AC
AE
BC
BE
CE

2-pass →

C_2

2- itemset	Count
AB	1
AC	2
AE	1
BC	2
BE	3
CE	2

→

L_2

Large 2-itemset	Count
AC	2
BC	2
BE	3
CE	2

C_3

3- itemset
BCE

3-pass →

C_3

3- itemset	Count
BCE	2

→

L_3

Large 3-itemset	Count
BCE	2

Result →

Large itemsets	Count
BCE	2
AC	2

รูปที่ 2.2 ตัวอย่างในการหาฟรีเควนท์ไอเทมเซตของอัลกอริทึมอะพริโอริ

ทั้งนี้ มีงานวิจัยหลายชิ้นที่นำเสนออัลกอริทึมที่มีพื้นฐานการทำงานจากอะพริโอริ เช่น อัลกอริทึมพาร์ทิชัน (Partition algorithm) ที่ใช้หลักการเช่นเดียวกันกับอัลกอริทึมอะพริโอริ แต่จะมีการแบ่งฐานข้อมูลออกเป็นส่วนย่อยๆ เรียกว่าพาร์ทิชัน (Partition) จากนั้นจะทำการสร้างแคนดิเดต k-ไอเทมเซต และค้นหาฟรีเควนท์ k-ไอเทมเซตจากแต่ละพาร์ทิชัน สำหรับไอเทมเซตที่จะกลายมาเป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลได้นั้น จะต้องเป็นฟรีเควนท์ไอเทมเซตอย่างน้อยในหนึ่งพาร์ทิชัน

2.2.2 การหาฟรีเควนท์ไอเทมเซตโดยไม่สร้างแคนดิเดต [3]

โดยทั่วไปแล้วในกรณีที่ข้อมูลมีขนาดใหญ่ การสร้างแคนดิเดตไอเทมเซตจำนวนมากก็จำเป็นต้องใช้หน่วยความจำขนาดใหญ่มากเช่นกัน นอกจากนี้ สัดส่วนระหว่างฟรีเควนท์ไอเทมเซตและแคนดิเดตไอเทมเซตนั้นมีค่าน้อยมาก ซึ่งบางครั้งอาจจะน้อยถึง 1:500 อัลกอริทึมแพทเทรินโกร์ท (Pattern Growth algorithm) จึงถูกนำเสนอขึ้นมาเพื่อแก้ไขข้อจำกัดเหล่านี้

2.2.2.1 อัลกอริทึมแพทเทรินโกร์ท (Pattern Growth algorithm)

แนวคิดหลักคือการหลีกเลี่ยงการสร้างแคนดิเดตไอเทมเซตโดยการนำโครงสร้างรูปต้นไม้ (Tree) มาใช้เก็บข้อมูลของฟรีเควนท์ไอเทมเซต ขั้นตอนในการค้นหาฟรีเควนท์ไอเทมเซต มีดังนี้

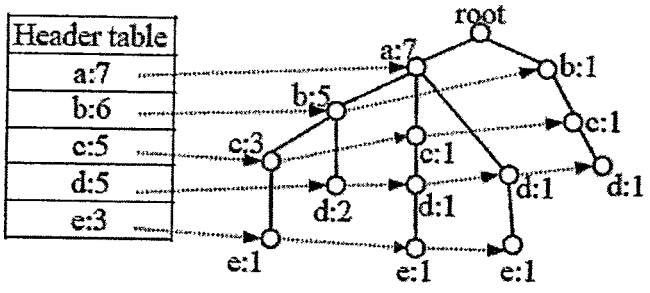
1) สแกนฐานข้อมูลทั้งหมด 1 รอบ เพื่อหาฟรีเควนท์ 1-ไอเทมเซต

2) เรียงลำดับฟรีเควนท์ 1-ไอเทมเซต จากมากไปน้อยตามค่าสนับสนุนของไอเทมเซตนั้นๆ ตารางที่ 2.1 แสดงตัวอย่างของชุดข้อมูล (คอลัมน์ 1-2) และฟรีเควนท์ 1-ไอเทมเซตที่เรียงลำดับแล้วของแต่ละรายการ (คอลัมน์ 3) เมื่อกำหนดให้ค่าสนับสนุนน้อยที่สุดมีค่าเท่ากับ 3

3) สแกนฐานข้อมูลอีกครั้งเพื่อทำการสร้าง FP-Tree โดยเริ่มสร้างกิ่งเดี่ยวจากรายการที่ 1 จากนั้นไล่รายการในลำดับต่อๆ ไป เพื่อทำการขยายกิ่งและเพิ่มค่าสนับสนุนให้กับโหนดของไอเทมที่ปรากฏอยู่แล้ว โดยไอเทมที่ไม่เป็นฟรีเควนท์ 1 ไอเทมเซต จะไม่ถูกนำมาสร้างเป็นกิ่ง (รูปที่ 2.3)

ตารางที่ 2.1 แสดงตัวอย่างชุดข้อมูลและฟรีเควนท์ 1-ไอเทมเซต ที่เรียงลำดับจากมากไปน้อยของแต่ละรายการ

TID	Items	Sorted frequent items
1	b,d,a	a,b,d
2	c,b,d	b,c,d
3	c,d,a,e	a,c,d,e
4	d,a,e	a,d,e
5	c,b,a	a,b,c
6	c,b,a	a,b,c
7	f,g	
8	b,d,a	a,b,d
9	c,b,a,e,f,g	a,b,c,e



รูปที่ 2.3 FP-Tree ที่สร้างจากข้อมูลในตาราง 2.1

2.3 การค้นหากฎความสัมพันธ์แบบเพิ่มขยาย

ข้อมูลที่ถูกจัดเก็บในฐานข้อมูลสามารถเกิดการเปลี่ยนแปลงได้ตลอดเวลา ซึ่งอาจเกิดจากการเพิ่มรายการเข้าไปในฐานข้อมูล (Insert) หรือลบรายการที่มีอยู่ในฐานข้อมูล (Delete) ในที่นี้จะเรียกส่วนของฐานข้อมูลก่อนทำการเปลี่ยนแปลงว่าฐานข้อมูลเดิม (Original database: DB) เรียกส่วนของข้อมูลใหม่ว่าฐานข้อมูลส่วนเพิ่ม (Incremental database: db) ซึ่งครอบคลุมได้ทั้งกรณีเพิ่มและลบข้อมูล และเรียกฐานข้อมูลที่ผ่านการเปลี่ยนแปลงแล้วว่าฐานข้อมูลอัพเดท (Updated database: UD)

เมื่อฐานข้อมูลมีการเปลี่ยนแปลงจะมีผลต่อกฎความสัมพันธ์ที่หาไว้แล้วในฐานข้อมูลเดิม เนื่องจากฟรีเควนท์ไอเทมเซตที่ได้ทำการค้นหาจากฐานข้อมูลเดิม อาจจะไม่เป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลอัพเดท ในขณะที่อินฟรีเควนท์ไอเทมเซตในฐานข้อมูลเดิมก็อาจจะกลายมาเป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลอัพเดทได้ ทั้งนี้ สามารถจัดกลุ่มไอเทมเซตที่ปรากฏเมื่อฐานข้อมูลมีการเปลี่ยนแปลงออกได้เป็น 4 กรณี ดังแสดงในตารางที่ 2.2

สำหรับกรณีที่ 1 และ 2 เนื่องจากเป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลเดิมทำให้ทราบค่าสนับสนุนของไอเทมเซตและสามารถนำค่าสนับสนุนนี้มาใช้ทำการค้นหาค่าสนับสนุนอัพเดทได้ ในกรณีที่ 3 และ 4 เนื่องจากไอเทมเซตนั้นไม่เป็นฟรีเควนท์ไอเทมเซตทั้งในฐานข้อมูลเดิม จึงไม่มีการเก็บค่าสนับสนุนของไอเทมเซตนั้นไว้ ดังนั้น หากต้องการอัพเดทค่าสนับสนุนของไอเทมเซต จะต้องทำการสแกนฐานข้อมูลเดิมเพื่อหาค่าสนับสนุนที่เกิดขึ้นจริงในฐานข้อมูลเดิมของไอเทมเซตนั้นๆ จากนั้นจึงสามารถหาฟรีเควนท์ไอเทมเซตของฐานข้อมูลอัพเดทได้

ตารางที่ 2.2 แสดงกรณีของไอเทมเซตที่ปรากฏเมื่อฐานข้อมูลมีการเปลี่ยนแปลง

DB \ UD	Large	Small
Large	กรณีที่ 1	กรณีที่ 2
Small	กรณีที่ 3	กรณีที่ 4

โดยทั่วไปแล้วฐานข้อมูลเดิมมักจะมีขนาดใหญ่กว่าส่วนของข้อมูลที่เพิ่มเข้ามา ดังนั้น งานวิจัยส่วนใหญ่จะนำเสนอวิธีเพื่อลดการสแกนฐานข้อมูลเดิมและสามารถค้นหากฎความสัมพันธ์

ใหม่ที่เกิดขึ้นได้อย่างมีประสิทธิภาพ แนวคิดของงานวิจัยที่นำเสนอในด้านการเพิ่มขยายการค้นหากฎความสัมพันธ์สามารถแบ่งได้เป็น 2 รูปแบบ คือ

2.3.1 การค้นหากฎความสัมพันธ์แบบเพิ่มขยายที่ให้ความสำคัญกับข้อมูลใหม่ที่เพิ่มเข้ามา

เนื่องจากโดยปกติแล้วจำนวนข้อมูลใหม่ที่เพิ่มเข้ามามีขนาดน้อยกว่าฐานข้อมูลเดิม ทำให้ฟรีเควนท์ไอเทมเซตที่เกิดขึ้นในฐานข้อมูลส่วนเพิ่มนี้อาจไม่เป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลเดิม แนวคิดนี้จะใช้วิธีการต่างๆ เพื่อประมาณค่าที่คาดว่าจะเกิดขึ้นในฐานข้อมูลเดิมให้กับฟรีเควนท์ไอเทมเซตใหม่ที่เกิดขึ้นเพื่อหลีกเลี่ยงการสแกนฐานข้อมูลเดิม ตัวอย่างของงานวิจัยในกลุ่มนี้ได้แก่ Weighting technique และ Time Influence Function

2.3.2 การค้นหากฎความสัมพันธ์แบบเพิ่มขยายที่ให้ความสำคัญกับข้อมูลเก่าและข้อมูลใหม่เท่ากัน

งานวิจัยในกลุ่มนี้ จะให้ความสำคัญกับข้อมูลเก่า (old dataset) และข้อมูลใหม่ (new dataset) เท่ากัน ซึ่งผลลัพธ์ของกฎที่ได้จะเหมือนกับการใช้อัลกอริทึมอะพริโอริประมวลผลข้อมูลทั้งเก่าและใหม่รวมกันทั้งหมด ทั้งนี้ เทคนิคที่ใช้ในงานวิจัยกลุ่มนี้ถูกจำแนกออกเป็นกลุ่มย่อยได้ 3 กลุ่ม ดังนี้

2.3.2.1 อัลกอริทึมที่มีพื้นฐานการทำงานของอะพริโอริ (Apriori-based)

งานวิจัยกลุ่มนี้เป็นกลุ่มที่ใช้หลักการของอัลกอริทึมอะพริโอริที่ได้รับความนิยมสูงในการหากฎความสัมพันธ์ โดยงานวิจัยแรกได้แก่อัลกอริทึม FUP ซึ่งนำเสนอโดย Cheung และคณะ [4] อัลกอริทึมนี้จะทำงานด้วยหลักการพื้นฐานของอะพริโอริ และเพิ่มการใช้ประโยชน์จากผลการไมนิงในรอบก่อนหน้า นั่นคือการเก็บค่าฟรีเควนท์ไอเทมเซตที่หาได้ในรอบก่อนหน้านี้ไว้มาใช้ประโยชน์ร่วมกับข้อมูลใหม่ที่ถูกเพิ่มเข้ามา เพื่อลดจำนวนแคนดิเดตไอเทมเซตที่จะต้องสแกนในฐานข้อมูลเดิม แต่อัลกอริทึม FUP สามารถกระทำได้เฉพาะกรณีที่มีการเพิ่มข้อมูลใหม่เข้าไปเท่านั้น หลังจากนั้นได้มีการพัฒนาอัลกอริทึม FUP2[5] เพื่อปรับปรุงประสิทธิภาพของ FUP โดยสามารถทำการหากฎความสัมพันธ์ได้ทั้งกรณีที่มีการเพิ่ม ลบ และการแก้ไข

ต่อมา Thomas S และคณะ [6] ได้เสนออัลกอริทึม Negative border ซึ่งใช้หลักการเช่นเดียวกับอะพริโอริ โดยมีการเก็บฟรีเควนท์ไอเทมเซตเพื่อใช้ประโยชน์ในการไมนิงในรอบต่อไปเช่นเดียวกับอัลกอริทึม FUP และ FUP2 นอกจากจะเก็บฟรีเควนท์ไอเทมเซตแล้วอัลกอริทึม Negative border ยังทำการเก็บค่าแคนดิเดตไอเทมเซตที่เป็นอินฟรีเควนท์เอาไว้ด้วย เพื่อเพิ่มประสิทธิภาพในการหากฎความสัมพันธ์ ซึ่งข้อเสียของอัลกอริทึมนี้คือต้องใช้หน่วยความจำในการเก็บทั้งฟรีเควนท์และอินฟรีเควนท์ไอเทมเซต

เพื่อแก้ปัญหาพื้นที่หน่วยความจำ Hong และ คณะ [7][8] ได้นำเสนอแนวคิดของ Pre-Large ได้นำเสนอแนวคิดในการเก็บไอเทมเซต 2 ประเภท คือ ฟรีเควนท์ไอเทมเซตและอินฟรีเควนท์ไอเทมเซตที่มีโอกาสเป็นฟรีเควนท์ในอนาคต เพื่อช่วยเพิ่มประสิทธิภาพในการหาฟรีเควนท์ไอเทมเซต

ในฐานข้อมูลปรับปรุง ต่อมา Amornchewin และ Kreesuradej [9] ได้นำเสนอ Probability-based Incremental Associaition Rule Discovery Algorithm ซึ่งเป็นอัลกอริทึมที่นำเสนอการเพิ่มขยายการค้นหากฎความสัมพันธ์โดยอาศัยหลักการความน่าจะเป็นโดยอาศัยทฤษฎีของเบอร์นูลลีมาใช้ในการทำนายและจัดเก็บไอเท็มที่คาดว่าจะเป็นฟรีเควนท์เมื่อมีการเพิ่มข้อมูลใหม่เข้ามา ทำให้ลดการสแกนฐานข้อมูลเดิมและสามารถทำการค้นหาฟรีเควนท์ไอเทมเซตในฐานข้อมูลปรับปรุงได้อย่างครบถ้วน

2.3.2.2 อัลกอริทึมที่มีพื้นฐานการทำงานแบบพาร์ทิชัน (Partition-based)

งานวิจัยกลุ่มนี้ เป็นกลุ่มที่นำเสนอเทคนิคการค้นหากฎความสัมพันธ์แบบเพิ่มขยายโดยการแบ่งข้อมูลที่มีอยู่ในฐานข้อมูลออกเป็นส่วนๆ ที่เรียกว่าพาร์ทิชัน เนื่องจากงานวิจัยจากกลุ่มที่มีพื้นฐานการทำงานของอะพริโอริมีข้อจำกัดหลักอยู่ 2 ประเด็นคือ 1) ปัญหาการสร้างแคนดิเดตไอเทมเซตจำนวนมาก และ 2) ปัญหาการสแกนฐานข้อมูลหลายๆ รอบ Chang-Hung Lee และคณะ [10] จึงเสนออัลกอริทึม Sliding-Window Filtering : SWF พยายามที่จะแก้ไขปัญหาดังกล่าว อัลกอริทึม SWF เป็นอัลกอริทึมที่ได้รับความนิยมในงานวิจัยแบบพาร์ทิชัน ทำงานโดยการแบ่งฐานข้อมูลออกเป็นพาร์ทิชัน แล้วทำการประมวลผลในแต่ละส่วนเพื่อเก็บค่าแคนดิเดตไอเทมเซตไว้ เพื่อคำนวณหาฟรีเควนท์ไอเทมเซตเป็นลำดับต่อไป จุดเด่นของ SWF คือจะเริ่มคำนวณหาแคนดิเดตไอเทมเซตที่ $k \geq 2$

2.3.2.3 อัลกอริทึมที่มีพื้นฐานการทำงานแบบแพทเทิร์นโกร์ท (Pattern-Growth based)

แนวคิดหลักคือการนำโครงสร้างรูปต้นไม้ (FP-tree) มาใช้เก็บสารสนเทศของฟรีเควนท์ไอเทมเซต เพื่อหลีกเลี่ยงการสร้างแคนดิเดตไอเทมเซต ทำให้สามารถลดการสแกนฐานข้อมูลได้ แต่เนื่องจาก FP-tree ไม่สามารถประยุกต์ใช้งานได้โดยตรงกับปัญหาทางด้านการหากฎความสัมพันธ์แบบเพิ่มขยาย จึงมีผู้นำเสนองานวิจัยที่สนับสนุนปัญหาดังกล่าว ได้แก่ DB-tree และPotFP-tree [11] ซึ่งพัฒนาหลักการทำงานมาจากอัลกอริทึม FP-Growth ต่างกันที่ DB-tree เก็บไอเทมทั้งหมดใน FP-tree ในขณะที่ FP-Growth จะเก็บเฉพาะฟรีเควนท์ 1-ไอเทมเซต ส่วน PotFp-tree จะจัดเก็บไอเทมที่มีโอกาสพัฒนาเป็นฟรีเควนท์ 1-ไอเทมเซตได้

2.4 งานวิจัยสำหรับการค้นหากฎความสัมพันธ์แบบเพิ่มขยาย

2.4.1 Fast Update Algorithm: FUP และ FUP2 [4][5]

อัลกอริทึม FUP [4] เป็นงานวิจัยแรกที่นำเสนอเทคนิคการบำรุงรักษากฎความสัมพันธ์เมื่อมีการเพิ่มข้อมูลใหม่เข้ามาในฐานข้อมูล โดยมุ่งเน้นที่จะลดจำนวนไอเทมเซตของข้อมูลที่เพิ่มเข้ามาเพื่อนำไปปรับปรุงค่าสนับสนุนในฐานข้อมูลเดิม ซึ่งจะถูกประมวลผลเก็บไว้ก่อนหน้านี้ในกรณีที่ไอเทมเซตนั้นมีค่าสนับสนุนผ่านค่าสนับสนุนน้อยที่สุดของฐานข้อมูลเดิม แต่ในกรณีที่ค่าสนับสนุนของไอเทมเซตนั้นไม่ผ่านค่าสนับสนุนน้อยที่สุด จะต้องทำการสแกนฐานข้อมูลเดิมใหม่เพื่อนำค่าสนับสนุนมาอัพเดท

การทำงานของ FUP อาศัยหลักการเดียวกับอะพริโอริ โดยแบ่งการทำงานเป็น 2 ส่วน คือ 1) การวนรอบซ้ำสำหรับ 1-ไอเทมเซต 2) การวนรอบซ้ำสำหรับ k-ไอเทมเซต เมื่อ $k \geq 2$ ซึ่งแต่ละส่วนมีรายละเอียดขั้นตอนการทำงาน ดังนี้

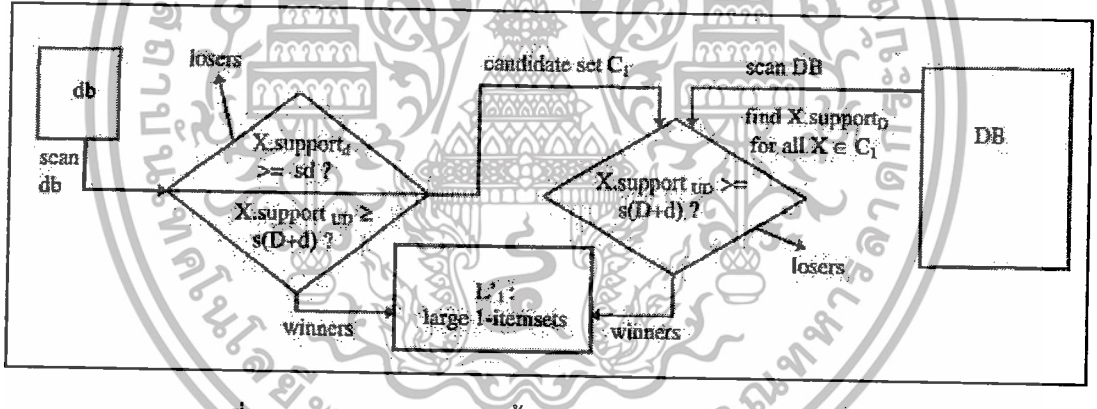
1) การวนรอบซ้ำสำหรับ 1-ไอเทมเซต

การทำงานในส่วนนี้ จะเป็นการสแกนฐานข้อมูลส่วนเพิ่มเพื่อหาลาร์จ 1-ไอเทมเซตในฐานข้อมูลอัพเดท ส่วนไอเทมที่ไม่มีโอกาสเป็นลาร์จ 1-ไอเทมเซตในฐานข้อมูลอัพเดทจะถูกตัดทิ้ง โดยมีหลักการพิจารณาดังนี้

1.1) ในกรณี $X \in L_1$ จะสามารถอัพเดทค่าสนับสนุนได้โดยการนำค่าสนับสนุนของไอเทม X จากฐานข้อมูลเดิม (DB) และจากฐานข้อมูลส่วนเพิ่ม (db) มารวมกัน ซึ่งถ้าค่าสนับสนุนที่ได้มีค่ามากกว่าหรือเท่ากับค่าสนับสนุนน้อยที่สุด แสดงว่าไอเทม X เป็นลาร์จ 1-ไอเทมเซตและให้ $X \in L'$

1.2) ในกรณี $X \notin L_1$ จะสามารถอัพเดทค่าสนับสนุนได้ก็ต่อเมื่อทำการสแกนฐานข้อมูลเดิม ทั้งนี้ สามารถลดจำนวนไอเทมที่จะสแกนในฐานข้อมูลเดิมได้โดยการพิจารณาเฉพาะไอเทมที่เป็นลาร์จในฐานข้อมูลส่วนเพิ่มเท่านั้น หลังจากนั้นจึงนำค่าสนับสนุนที่อัพเดทแล้วมาตรวจสอบว่าเป็นลาร์จ 1-ไอเทมเซตหรือไม่ด้วยวิธีการเช่นเดียวกับข้อ 1.1 ส่วนไอเทม X ที่ไม่เป็นลาร์จในฐานข้อมูลส่วนเพิ่มจะถูกตัดออกและไม่นำมาสแกนในฐานข้อมูลเดิม

เมื่อเสร็จสิ้นกระบวนการทำงานในรอบนี้ จะได้ L'_1 ของฐานข้อมูลที่อัพเดทแล้ว ทั้งนี้กระบวนการทำงานในรอบแรกเพื่อหาลาร์จ 1-ไอเทมเซตของอัลกอริทึม FUP แสดงได้ดังรูปที่ 2.4



รูปที่ 2.4 กระบวนการวนรอบซ้ำสำหรับ 1-ไอเทมเซตของ FUP

2) การวนรอบซ้ำสำหรับ k-ไอเทมเซต เมื่อ $k \geq 2$

การทำงานในส่วนนี้จะเป็นการหาลาร์จ k-ไอเทมเซต เมื่อ $k \geq 2$ ซึ่งจะมีบางขั้นตอนที่มีหลักการทำงานคล้ายกับการทำงานในรอบแรก นั่นคือการหาไอเทมที่มีโอกาสเป็น L'_k เพื่อลดการสแกนข้อมูล โดยอาศัยแนวคิดที่ว่า "ถ้าไอเทม X ใดๆ ที่เป็น loser ในการประมวลผลรอบที่ k-1 (เมื่อ $k \geq 2$) แล้วไอเทมเซตใดๆ ของ L_k ในฐานข้อมูลเดิมที่มีไอเทม X ดังกล่าวเป็นสับเซตอยู่จะไม่สามารถเป็น winner ในรอบที่ k ได้". ขั้นตอนการทำงานภายใต้แนวคิดดังกล่าวมีดังนี้

2.1) ไอเทมเซตที่เคยเป็นลาร์จ 1-ไอเทมเซตในฐานข้อมูลเดิมแต่ไม่เป็นลาร์จ 1-ไอเทมเซตในฐานข้อมูลอัพเดท ซูเปอร์เซตของไอเทมเซตเหล่านี้จะไม่สามารถเป็นลาร์จ 2-ไอเทมเซตได้

และจะถูกตัดออกไปจาก L_2 ส่วนไอเทมเซตที่เหลือจะถูกนำไปสแกนใน db เพื่ออัพเดทค่าสนับสนุน และพิจารณาว่าจะถูกเก็บใน L'_2 หรือไม่

2.2) หาลาร์จ 2-ไอเทมเซตที่เกิดขึ้นใหม่ โดยในขั้นตอนนี้จะเริ่มสร้างแคนดิเดต 2-ไอเทมเซต (C_2) ด้วยการเชื่อมระหว่าง $L'_1 * L'_1$ เพื่อนำไอเทมเซตที่ได้ไปสแกนใน db และหา L'_2 โดยไม่ต้องนำไอเทมเซตที่เป็น L_2 มาสร้าง C_2 เนื่องจากสามารถอัพเดทค่าสนับสนุนของไอเทม X ได้จากขั้นตอนที่ 2.1 แล้ว ส่วนในกรณี $X \in C_2$ และ $X \notin L_2$ ให้นำไอเทม X นั้นไปสแกนใน db เพื่อค่าสนับสนุน สำหรับไอเทม X ที่เป็นลาร์จใน db ที่จะถูกนำไปสแกนใน DB จากนั้นอัพเดทค่าสนับสนุนและพิจารณาว่าควรเพิ่มไอเทม X นั้นเป็นสมาชิกของ L'_2 หรือไม่

2.3) ทำซ้ำข้อ 2.1-2.2 เพื่อหาฟรีเควนท์ k-ไอเทมเซต ($k \geq 3$) ในรอบต่อๆ ไป

เนื่องจากอัลกอริทึม FUP สามารถใช้งานได้ในกรณีของการเพิ่มรายการใหม่เข้าไปในฐานข้อมูลเดิมเท่านั้น ยังไม่สามารถใช้ได้กับกรณีการลบรายการออกจากฐานข้อมูลได้ จึงได้มีการพัฒนา FUP2 [5] ซึ่งรองรับทั้งการเพิ่มและลบรายการ อัลกอริทึม FUP2 จะแบ่งการทำงานออกเป็น 2 กรณี คือ 1) กรณีที่รายการถูกลบอย่างเดียว 2) กรณีที่รายการถูกลบและเพิ่ม ตารางที่ 2.3 แสดงความหมายของสัญลักษณ์ต่างๆ ที่ใช้ในอัลกอริทึม FUP2 ซึ่งมีรายละเอียดขั้นตอนการทำงาน ดังนี้

ตารางที่ 2.3 แสดงความหมายของสัญลักษณ์ต่างๆ ที่ใช้ในอัลกอริทึม FUP2

Database	Support count of item X	Large k-itemsets
Δ^+	δ_x^+	-
D^-	-	-
Δ^-	δ_x^-	-
$D = \Delta^- \cup D^-$	σ_x	L_k
$D' = D^- \cup \Delta^+$	σ'_x	L'_k

1) กรณีที่รายการถูกลบอย่างเดียว

สำหรับการลบรายการนั้น ไอเทมเซตที่เป็นสมอลล์ในฐานข้อมูลเดิม จะมีโอกาสเป็นลาร์จในฐานข้อมูลอัพเดทได้ก็ต่อเมื่อไอเทมเซตนั้นเป็นสมอลล์ในรายการที่ถูกลบ รูปที่ 2.5 แสดงตัวอย่างของฐานข้อมูลเดิมและรายการที่ถูกลบ เมื่อกำหนด min_sup = 0.25 ซึ่งอัลกอริทึม FUP2 มีขั้นตอนการทำงาน ดังนี้

1.1) การวนรอบซ้ำสำหรับ 1-ไอเทมเซต

1.1.1) สร้างแคนดิเดตไอเทมเซต $C_k = I$ ซึ่งได้มาจากการไมนิงก่อนหน้านี้ จากนั้นแบ่ง C_k ออกเป็น 2 ส่วน คือ P_k และ Q_k โดย P_k หมายถึงไอเทมเซตที่เป็นลาร์จในฐานข้อมูลเดิม และ Q_k หมายถึงไอเทมเซตที่เป็นสมอลล์ในฐานข้อมูลเดิม

1.1.2) สแกนรายการที่ถูกลบ (Δ^-) เพื่อหาค่าสนับสนุนของแต่ละไอเทมเซต ในกรณีที่ไอเทมเซตนั้นอยู่ใน P_k สามารถนำค่าสนับสนุนจากฐานข้อมูลเดิมมาอัพเดทค่าสนับสนุนใหม่ได้จาก $\sigma_x' = \sigma_x - \delta_x^-$ ส่วนในกรณีที่ไอเทมเซตนั้นอยู่ใน Q_k และ $\delta_x^- \geq |\Delta^-| \times s\%$ ไอเทมเซตนั้น

จะถูกตัดทิ้ง เนื่องจากไอเทมเซตที่เป็นสมอลล์ในฐานข้อมูลเดิมและเป็นลาร์จในรายการที่ถูกลบออก จะไม่สามารถเป็นลาร์จในฐานข้อมูลอัพเดทได้ ทั้งนี้ อัลกอริทึมจะเก็บข้อมูลไอเทมเซตที่เป็นสมอลล์ของรายการที่ถูกลบไว้ใช้ในรอบต่อไป

1.1.3) สแกน D^- เพื่อหาค่าสนับสนุนของแต่ละไอเทมเซตใน Q_k ที่เหลืออยู่ จากนั้นอัพเดทค่าสนับสนุนของแต่ละไอเทมเซต สำหรับไอเทมเซตใน $P_k \cup Q_k$ ที่ $\sigma_x' \geq |D'| \times s\%$ จะเป็นไอเทมเซตที่เป็นลาร์จในฐานข้อมูลอัพเดทและถูกเก็บไว้ใน L'_k

1.2) การวนรอบซ้ำสำหรับ k-ไอเทมเซต เมื่อ $k \geq 2$

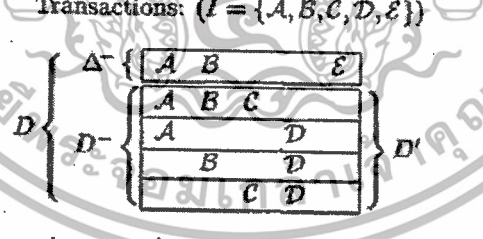
1.2.1) สร้างแคนดิเดตไอเทมเซต $C_k = L'_{k-1} \cup L'_{k-1}$ แบ่ง C_k ออกเป็น 2 ส่วน คือ P_k และ Q_k

1.2.2) สามารถตัดไอเทมเซตใน Q_k ที่ไม่มีโอกาสเป็นลาร์จในฐานข้อมูลอัพเดทได้ โดยยังไม่จำเป็นต้องสแกนรายการที่ถูกลบเพื่อหาหาค่าสนับสนุน ภายใต้แนวคิดว่า "ซูเปอร์เซตทุกตัวของสมอลล์ 1-ไอเทมเซตจะต้องเป็นสมอลล์ไอเทมเซตด้วย" ดังนั้น ถ้าไอเทมเซตใดๆ ใน Q_k มีสับเซตเป็นสมอลล์ 1-ไอเทมเซตในรายการที่ถูกลบ (ซึ่งหาได้จากข้อ 1.1.2) แสดงว่าไอเทมเซตนั้นมีโอกาสเป็นลาร์จในฐานข้อมูลอัพเดท แคนดิเดตไอเทมเซตกลุ่มนี้จะถูกย้ายไปเก็บไว้ใน R_k ส่งผลให้สามารถลดจำนวนแคนดิเดตไอเทมเซตใน Q_k ที่ต้องสแกนในขั้นตอน 1.2.3 ได้

1.2.3) สแกน Δ^- เพื่อหาค่าสนับสนุนของแต่ละไอเทมเซตใน $P_k \cup Q_k$ สำหรับไอเทมเซตที่อยู่ใน P_k สามารถอัพเดทค่าสนับสนุนใหม่ได้จาก $\sigma_x' = \sigma_x - \delta_x^-$ ส่วนในกรณีที่ไอเทมเซตนั้นอยู่ใน Q_k และ $\delta_x^- \geq |\Delta^-| \times s\%$ ไอเทมเซตนั้นจะถูกตัดทิ้ง

1.2.4) ย้ายแคนดิเดตไอเทมเซตใน R_k มารวมกับ Q_k ที่เหลืออยู่ จากนั้นสแกน D^- เพื่อหาค่าสนับสนุนของแต่ละไอเทมเซตใน Q_k สำหรับไอเทมเซตใน $P_k \cup Q_k$ ที่ $\sigma_x' \geq |D'| \times s\%$ จะเป็นไอเทมเซตที่เป็นลาร์จในฐานข้อมูลอัพเดทและถูกเก็บไว้ใน L'_k

1.2.5) ทำซ้ำขั้นตอน 1.2.1 – 1.2.4 หยุดการทำงานเมื่อ $|L'_k| < k+1$



Large itemsets (support threshold $s = 25\%$) in $D = \Delta^- \cup D^-$:

Itemsets(X)	A	B	C	D	AB
σ_X	3	3	2	3	2

รูปที่ 2.5 ตัวอย่างของฐานข้อมูลเดิม, รายการที่ถูกลบ, และลาร์จไอเทมเซต เมื่อกำหนด min_sup = 0.25

2) กรณีที่รายการถูกลบและเพิ่ม

ขั้นตอนการทำงานหลักๆ คือ สแกน Δ^- ก่อน จากนั้นจึง Δ^+ เพื่ออัพเดทค่าสนับสนุนของไอเทมเซตที่อยู่ใน P_k อย่างไรก็ตาม สำหรับการลดจำนวนแคนดิเดตไอเทมเซตที่ต้องสแกนใน Q_k นั้น ในกรณีนี้จะไม่สามารถนำแนวคิดที่ว่า "ถ้าไอเทมเซตใดๆ ใน Q_k มีสับเซตเป็นสมอลล์ 1-ไอเทมเซตใน Δ^- แสดงว่าไอเทมเซตนั้นมีโอกาสเป็นลาร์จในฐานข้อมูลอัพเดท" มาใช้ได้เหมือนกรณีลบข้อมูลอย่างเดียว เนื่องจากค่าสนับสนุนของแต่ละไอเทมเซตและจำนวนรายการทั้งหมดที่เพิ่มเข้ามาใน Δ^+ อาจทำให้ไอเทมเซตนั้นๆ มีโอกาสเป็นได้ทั้งลาร์จและสมอลล์

FUP2 ได้นำแนวคิดของการหาขอบเขต (bound) ของค่าสนับสนุน δ_X^+ และ δ_X^- มาใช้ในการลดจำนวนแคนดิเดตไอเทมเซตที่ต้องสแกนใน Δ^- และ Δ^+ โดยกำหนดให้ขอบเขตบน (upper bound) b_Y^- มีค่าเท่ากับค่าต่ำสุดของ δ_X^- และ b_Y^+ มีค่าเท่ากับค่าต่ำสุดของ δ_X^+ เมื่อ $X \subset Y$ และ $|X| = |Y|-1$ ทั้งนี้สำหรับไอเทมเซต X และ Y เมื่อ $X \subseteq Y$ แต่ละรายการของไอเทม Y ที่เกิดขึ้นใน Δ^- และ Δ^+ จะต้องมีไอเทม X อยู่ในนั้นด้วย ดังนั้น $\delta_X^- \geq \delta_Y^-$ และ $\delta_X^+ \geq \delta_Y^+$

จากแนวคิดดังกล่าว ทำให้สามารถลดจำนวนแคนดิเดตไอเทมเซตที่ต้องสแกนใน Δ^- ได้ดังนี้ สำหรับแต่ละแคนดิเดตไอเทมเซต X ที่อยู่ใน Q_k ถ้า $b_X^+ \leq (|\Delta^+|-|\Delta^-|) \times s\%$ ไอเทมเซตนั้นๆ จะไม่มีโอกาสเป็น L' จึงตัดไอเทมเซตนั้นออกได้ สำหรับแต่ละแคนดิเดตไอเทมเซต X ที่อยู่ใน P_k จะสามารถตัดไอเทมเซตนั้นออกได้ถ้า $\sigma_X + b_X^+ < |D'| \times s\%$ หลังจากนั้นจึงสแกน Δ^- เพื่อหาค่า δ_X^-

ด้วยหลักการเดียวกันนี้เราสามารถลดจำนวนแคนดิเดตไอเทมเซตที่ต้องสแกนใน Δ^+ ได้ โดยไอเทมเซตที่อยู่ใน Q_k จะถูกตัดออกถ้า $b_X^+ - \delta_X^- \leq (|\Delta^+|-|\Delta^-|) \times s\%$ และไอเทมเซตที่อยู่ใน P_k จะถูกตัดออกถ้า $\sigma_X + b_X^+ - \delta_X^- \leq |D'| \times s\%$ หลังจากนั้นจึงสแกน Δ^+ เพื่อหา δ_X^+ ของไอเทมเซตที่เหลืออยู่

2.1) ในกรณีที่ $|\Delta^-| > |\Delta^+|$ เราสามารถลดจำนวนแคนดิเดตไอเทมเซตที่ต้องสแกนใน Δ^- ได้เพิ่มอีกสำหรับไอเทมเซตใน Q_k โดยพิจารณาว่าถ้า $b_X^+ - b_X^- \geq (|\Delta^+|-|\Delta^-|) \times s\%$ แสดงว่าไอเทมเซตนั้นมีโอกาสเป็น L' ไอเทมเซตเหล่านี้จะถูกย้ายไปอยู่ใน R_k ทำให้จำนวนไอเทมเซตที่ต้องสแกนใน Δ^- ลดลง แต่จะไม่ใช้วิธีนี้ในกรณีที่ $|\Delta^-| < |\Delta^+|$ เนื่องจากจะส่งผลให้จำนวนไอเทมเซตที่ต้องสแกนใน D^- เพิ่มขึ้น โดยจะกำหนดให้ $b_X^- = \delta_X^-$ สำหรับทุกไอเทมเซตที่อยู่ใน R_k เพื่อใช้ในการคำนวณรอบต่อๆ ไป จากนั้นสามารถลดจำนวนแคนดิเดตไอเทมเซตที่ต้องสแกน Δ^+ ได้โดยตัดไอเทมเซตที่มีค่า $\delta_X^+ \leq (|\Delta^+| - |\Delta^-|) \times s\%$ ออกไปจาก R_k

ด้วยหลักการที่กล่าวมาทั้งหมด จะได้ขั้นตอนการทำงานของ FUP2 ในกรณีที่มีทั้งการเพิ่มและลบรายการ ดังนี้

2.1) สร้างแคนดิเดตไอเทมเซต โดยในการทำงานรอบแรก k=1 จะกำหนดให้ $C_k = I$ สำหรับการทำงานรอบต่อๆ ไป $k \geq 2$ กำหนดให้ $C_k = L'_{k-1} \cup L'_{k-1}$

2.2) คำนวณ b_X^+ สำหรับแต่ละไอเทมเซต $X \in C_k$ โดยกำหนดให้ $b_X^+ = |\Delta^+|$ เมื่อ k=1 และ $b_X^+ = \delta_X^+$ ที่มีค่าน้อยที่สุด เมื่อ $k \geq 2$

2.3) แบ่ง C_k ออกเป็น 2 ส่วน คือ P_k และ Q_k

2.4) สำหรับแต่ละไอเทมเซต X ใน P_k ที่มีค่า $\sigma_x + b_X^+ < |D'| \times s\%$ จะถูกตัดทิ้ง

2.5) สำหรับแต่ละไอเทมเซต X ใน Q_k ที่มีค่า $b_X^+ \leq (|\Delta^+| - |\Delta^-|) \times s\%$ จะถูกตัดทิ้ง

2.6) ถ้า $|\Delta^-| \le |\Delta^+|$ ให้ $R_k = \varnothing$ แต่ถ้า $|\Delta^-| > |\Delta^+|$ ให้คำนวณ b_X^- สำหรับแต่ละไอเทมเซต X ใน Q_k และถ้า $b_X^+ - b_X^- \ge (|\Delta^+|-|\Delta^-|) \times s\%$ ให้ย้ายไอเทมเซตนั้นไปไว้ใน R_k และกำหนดให้ $\delta_X^- = b_X^-$

2.7) สแกนรายการที่ถูกลบ (Δ^-) เพื่อหาค่าสนับสนุน δ_X^- ของแต่ละไอเทมเซต

2.8) ไอเทมเซตที่อยู่ใน P_k และมีค่า $\sigma_x + b_X^+ - \delta_X^- < |D'| \times s\%$ จะถูกตัดทิ้ง

2.9) ไอเทมเซตที่อยู่ใน Q_k และมีค่า $b_X^+ - \delta_X^- \le (|\Delta^+| - |\Delta^-|) \times s\%$ จะถูกตัดทิ้ง

2.10) สแกนรายการที่เพิ่มเข้ามา (Δ^+) เพื่อหาค่าสนับสนุน δ_X^+ ของแต่ละไอเทมเซต X ที่อยู่ใน $P_k \cup Q_k \cup R_k$

2.11) สำหรับแต่ละไอเทมเซตที่อยู่ใน P_k สามารถนำค่าสนับสนุนจากฐานข้อมูลเดิมมาอัพเดทค่าสนับสนุนใหม่ได้ด้วยการนำค่าสนับสนุนมาบวกกัน $\sigma_x' = \sigma_x + \delta_X^+$

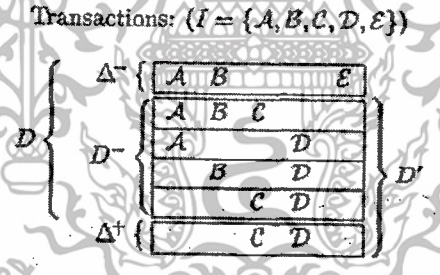
2.12) สำหรับแต่ละไอเทมเซตที่อยู่ใน Q_k ถ้า $\delta_X^+ - \delta_X^- \le (|\Delta^+| - |\Delta^-|) \times s\%$ ให้ลบไอเทมเซตนั้นทิ้ง

2.13) สำหรับแต่ละไอเทมเซตที่อยู่ใน R_k ถ้า $\delta_X^+ \le (|\Delta^+| - |\Delta^-|) \times s\%$ ให้ลบไอเทมเซตนั้นทิ้ง

2.14) สแกนฐานข้อมูลที่ลบรายการออกไปแล้ว (D^-)และหาค่าสนับสนุนสำหรับแต่ละไอเทมเซต X ที่อยู่ใน $Q_k \cup R_k$ จากนั้นนำค่าที่ได้ไปบวกกับ δ_x^+ เพื่ออัพเดทค่าสนับสนุน σ_x'

2.15) สำหรับไอเทมเซตใน $P_k \cup Q_k \cup R_k$ ที่ $\sigma_x' \ge |D'| \times s\%$ จะเป็นลาร์จไอเทมเซตในฐานข้อมูลอัพเดทและถูกเก็บไว้ใน L'_k

2.16) หยุดการทำงานเมื่อ $|L'_k| < k+1$



Large itemsets (support threshold $s = 25\%$) in $D' = D^- \cup \Delta^+$:

Itemsets(X)	A	B	C	D	CD
σ'_X	2	2	3	4	2

รูปที่ 2.6 ตัวอย่างของฐานข้อมูลเดิม รายการที่ถูกลบ รายการที่เพิ่ม และลาร์จไอเทมเซตที่อัพเดทแล้วเมื่อกำหนด min_sup = 0.25

ข้อดีของ FUP2

1) มีการเก็บผลลัพธ์จากการค้นหากฎความสัมพันธ์ในฐานข้อมูลเดิมมาใช้ร่วมกับฐานข้อมูลใหม่ที่เพิ่มเข้ามา

2) สามารถลดจำนวนแคนดิเดตไอเทมเซตที่ต้องสแกนในฐานข้อมูลเดิม

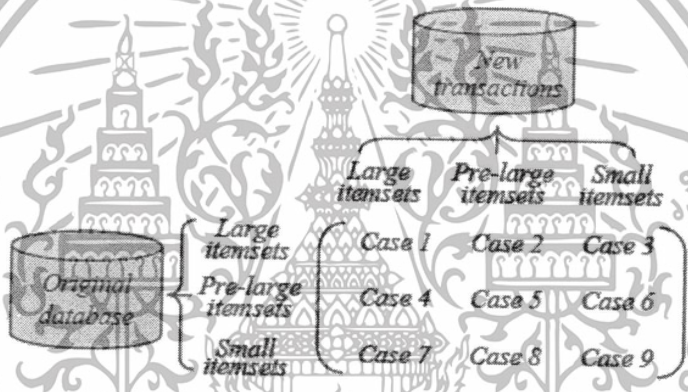
3) รองรับการทำงานทั้งในกรณีเพิ่มและลบรายการข้อมูล

ข้อจำกัดของ FUP2

1) ต้องสแกนฐานข้อมูลเดิมทุกรอบ k ในกรณีที่ต้องการอัพเดทค่าสนับสนุนของไอเทมเซตที่เป็นสมอลล์ในฐานข้อมูลเดิม

2.4.2 Pre-large itemsets for Insertion [7]

แนวคิดของไอเทมเซตแบบพรีลาร์จคือการเก็บไอเทมเซตที่ไม่ได้เป็นลาร์จไอเทมเซต แต่มีแนวโน้มที่จะเป็นลาร์จในอนาคตเมื่อมีรายการใหม่เพิ่มเข้ามา โดยทำการค้นหากฎความสัมพันธ์ด้วยหลักการของ FUP แนวคิดนี้จะใช้ขีดแบ่ง 2 ค่า คือ 1) Lower support threshold ซึ่งใช้เป็นค่าสนับสนุนน้อยที่สุดของไอเทมเซตที่จะถูกจัดเป็นพรีลาร์จ และ 2) Upper support threshold ซึ่งใช้เป็นค่าสนับสนุนน้อยที่สุดของไอเทมเซตที่จะถูกจัดเป็นลาร์จ ทั้งนี้ไอเทมเซตที่มีค่าสนับสนุนต่ำกว่า Lower support threshold จะถูกจัดเป็นไอเทมเซตแบบสมอลล์ ซึ่งเมื่อพิจารณาจากฐานข้อมูลเดิมและรายการใหม่ที่ถูกเพิ่มเข้ามา พบว่าสามารถแบ่งสถานการณ์ได้เป็น 9 กรณี ดังรูปที่ 2.7 และตารางที่ 2.4



รูปที่ 2.7 กรณีของไอเทมเซตที่ปรากฏเมื่อมีการเพิ่มรายการข้อมูลในอัลกอริทึมพรีลาร์จ

กรณีที่ 1, 5, 6, 8 และ 9 ไม่ส่งผลให้เกิดการเปลี่ยนแปลงต่อกฎความสัมพันธ์
กรณีที่ 2 และ 3 อาจส่งผลให้กฎความสัมพันธ์บางข้อถูกลบออกไป
กรณีที่ 4 และ 7 อาจส่งผลให้เกิดกฎความสัมพันธ์ใหม่ อย่างไรก็ตาม ในกรณีที่จำนวนรายการที่ถูกเพิ่มเข้ามาใหม่มีจำนวนน้อยมากเมื่อเทียบกับจำนวนรายการทั้งหมดที่อยู่ในฐานข้อมูลเดิม ไอเทมเซตที่เป็นสมอลล์ในฐานข้อมูลเดิมจะไม่สามารถเปลี่ยนเป็นลาร์จในฐานข้อมูลที่อัพเดทได้ ถึงแม้ว่าไอเทมเซตนั้นจะเป็นลาร์จในรายการที่เพิ่มเข้ามาใหม่ก็ตาม

ตารางที่ 2.4 แสดงผลของกฎความสัมพันธ์ที่อาจเกิดการเปลี่ยนแปลงจากการเพิ่มรายการข้อมูล

Cases: Original – New	Results
Case 1: Large – Large	Always large
Case 2: Large – Pre-large	Large or pre-large, determined from existing information
Case 3: Large – Small	Large or pre-large or small, determined from existing information
Case 4: Pre-large – Large	Pre-large or large, determined from existing information
Case 5: Pre-large – Pre-large	Always pre-large
Case 6: Pre-large – Small	Pre-large or small, determined from existing information
Case 7: Small – Large	Pre-large or small when the number of transactions is small
Case 8: Small – Pre-large	Small or pre-large
Case 9: Small – Small	Always small

ความหมายของสัญลักษณ์ต่างๆ ที่ใช้ในอัลกอริทึมพรีลาร์จ

D	หมายถึง	ฐานข้อมูลเดิม
T	หมายถึง	เซตของรายการที่ถูกเพิ่มเข้ามา
U	หมายถึง	ฐานข้อมูลที่อัพเดทแล้ว
d	หมายถึง	จำนวนรายการที่มีอยู่ใน D
t	หมายถึง	จำนวนรายการที่มีอยู่ใน T
S_l	หมายถึง	Lower support threshold สำหรับพรีลาร์จไอเทมเซต
S_u	หมายถึง	Upper support threshold สำหรับลาร์จไอเทมเซต, $S_u > S_l$
L_k^D	หมายถึง	เซตของลาร์จ k-ไอเทมเซตที่อยู่ใน D
L_k^T	หมายถึง	เซตของลาร์จ k-ไอเทมเซตที่อยู่ใน T
L_k^U	หมายถึง	เซตของลาร์จ k-ไอเทมเซตที่อยู่ใน U
P_k^D	หมายถึง	เซตของพรีลาร์จ k-ไอเทมเซตที่อยู่ใน D
P_k^T	หมายถึง	เซตของพรีลาร์จ k-ไอเทมเซตที่อยู่ใน T
P_k^U	หมายถึง	เซตของพรีลาร์จ k-ไอเทมเซตที่อยู่ใน U
C_k	หมายถึง	เซตของแคนดิเดต k-ไอเทมเซตทุกตัวที่อยู่ใน T
I	หมายถึง	ไอเทมเซต
$S^D(I)$	หมายถึง	ค่าสนับสนุนของ I ที่เกิดขึ้นใน D
$S^T(I)$	หมายถึง	ค่าสนับสนุนของ I ที่เกิดขึ้นใน T
$S^U(I)$	หมายถึง	ค่าสนับสนุนของ I ที่เกิดขึ้นใน U

ดังที่กล่าวไว้ข้างต้น ในกรณีที่จำนวนรายการที่ถูกเพิ่มเข้ามาใหม่มีจำนวนน้อยมากเมื่อเทียบกับจำนวนรายการทั้งหมดที่อยู่ในฐานข้อมูลเดิม ไอเทมเซตที่เป็นสมอลล์ในฐานข้อมูลเดิมจะไม่สามารถเปลี่ยนเป็นลาร์จในฐานข้อมูลอัพเดทได้ ถึงแม้ว่าไอเทมเซตนั้นจะเป็นลาร์จในรายการที่เพิ่มเข้ามาใหม่ก็ตาม ดังนั้น อัลกอริทึมพรีลาร์จจะพิจารณาขนาดของฐานข้อมูลเดิมและจำนวนรายการที่เพิ่มเข้ามา นำมาคำนวณเปรียบเทียบกับ Lower support threshold และ Upper support threshold ตามสมการ (2.3) โดยถ้าจำนวนรายการที่เพิ่มเข้ามามีค่าน้อยกว่าหรือเท่ากับค่าที่คำนวณได้ จะไม่สแกนฐานข้อมูลเดิม เนื่องจากไอเทมเซตที่เป็นสมอลล์ในฐานข้อมูลเดิมแต่เป็นลาร์จในรายการที่เพิ่มเข้ามาใหม่ยังมีจำนวนไม่เพียงพอที่จะเปลี่ยนเป็นลาร์จในฐานข้อมูลอัพเดทได้ แต่ถ้าจำนวนรายการที่เพิ่มเข้ามามีค่ามากกว่าค่าที่คำนวณได้อัลกอริทึมพรีลาร์จจะทำงานด้วยหลักการของอัลกอริทึม FUP

$$t \leq \frac{(S_u - S_l)d}{1 - S_u} \qquad (2.3)$$

อัลกอริทึมพรีลาร์จจะแบ่งการทำงานออกเป็น 2 ส่วนหลัก คือ 1) การทำงานกับฐานข้อมูลเดิม และ 2) การทำงานกับรายการใหม่ที่ถูกเพิ่มเข้ามา มีรายละเอียด ดังนี้

1) การทำงานกับฐานข้อมูลเดิม

อัลกอริทึมนี้จะทำการประมวลผลในส่วนของฐานข้อมูลเดิมโดยใช้ S_u และ S_l เพื่อหาลาร์จ 1-ไอเทมเซตและพรีลาร์จ 1-ไอเทมเซตตามลำดับ หลังจากนั้นจะนำทั้งลาร์จ 1-ไอเทมเซตและพรี

ลาร์จ 1-ไอเทมเซตไปทำการดำเนินการเชื่อมเพื่อสร้างแคนดิเดต C_2 โดยใช้หลักการพื้นฐานของอะพริโอริ ทำเช่นนี้ซ้ำไปเรื่อยๆ จนกว่าจะไม่สามารถสร้างแคนดิเดตของระดับถัดไปได้ ตัวอย่างของผลลัพธ์ที่ได้จากการประมวลผล แสดงดังตารางที่ 2.5-2.6

ตารางที่ 2.5 แสดงลาร์จไอเทมเซตที่ได้จากการไมนิงฐานข้อมูลเดิม

1 item	Count	2 items	Count	3 items	Count
A	5	BC	4	BCE	4
B	6	BE	6		
C	6	CE	4		
E	6				

ตารางที่ 2.6 แสดงพรีลาร์จไอเทมเซตที่ได้จากการไมนิงฐานข้อมูลเดิม

1 item	Count	2 items	Count	3 items	Count
D	4	AC	3	ABE	3
		AE	3		
		CD	3		

2) การทำงานกับรายการใหม่ที่ถูกเพิ่มเข้ามา

เมื่อมีรายการใหม่ถูกเพิ่มเข้ามา อัลกอริทึมจะสแกนเฉพาะส่วนของรายการใหม่นี้เพื่อหา 1-ไอเทมเซตและนำไปเปรียบเทียบกับผลลัพธ์ที่ได้จากฐานข้อมูลเดิม หากไอเทมเซตนี้อยู่ในกลุ่มลาร์จหรือพรีลาร์จของฐานข้อมูลเดิม ก็จะสามารถอัพเดทค่าสนับสนุนได้จากข้อมูลเดิมที่มีอยู่แล้ว แต่ถ้าไอเทมเซตนี้ไม่ได้อยู่ในลาร์จหรือพรีลาร์จของฐานข้อมูลเดิม และจำนวนรายการใหม่ที่ถูกเพิ่มเข้ามามีค่ามากกว่า Safety threshold ที่คำนวณได้ ฐานข้อมูลเดิมจะถูกสแกนใหม่อีกครั้งเพื่อหาและอัพเดทลาร์จไอเทมเซตที่พบใหม่นี้ โดยมีการเก็บจำนวนรายการใหม่ที่ถูกเพิ่มเข้ามานับจากฐานข้อมูลเดิมถูกสแกนใหม่ครั้งสุดท้ายไว้ในตัวแปร c รายละเอียดขั้นตอนการทำงานของอัลกอริทึม มีดังต่อไปนี้

อินพุต: S_u, S_l, ลาร์จไอเทมเซตและพรีลาร์จไอเทมเซตจากฐานข้อมูลเดิม, เซตของรายการใหม่ที่ถูกเพิ่มเข้ามา

เอาต์พุต: เซตของกฎความสัมพันธ์จากฐานข้อมูลอัพเดท

ขั้นตอนการทำงาน:

2.1) คำนวณหา Safety number จาก

$$f = \frac{(S_u - S_l)d}{1 - S_u}$$

2.2) กำหนดให้ k=1 เมื่อ k คือความยาวของไอเทมเซตที่กำลังอยู่ในการทำงานรอบปัจจุบัน

2.3) หา C_k และค่าสนับสนุนของแต่ละไอเทมเซตจากรายการที่ถูกเพิ่มเข้ามา

2.4) ใช้ข้อมูลอ้างอิงจากฐานข้อมูลเดิม แบ่ง C_k ออกเป็น 3 กลุ่ม ได้แก่ ลาร์จไอเทมเซต, พรีลาร์จไอเทมเซต, และสมอลล์ไอเทมเซต

2.5) สำหรับแต่ละไอเทมเซต I ที่อยู่ในลาร์จ k-ไอเทมเซตของฐานข้อมูลเดิม $\mathbf{L}_k^D$ ให้ดำเนินการต่อไปนี้

2.5.1 กำหนดให้ค่าความถี่อัพเดท $S^U(I) = S^T(I) + S^D(I)$

2.5.2 ถ้า $S^U(I) / (d+t+c) \geq S_u$ กำหนดให้ I เป็นลาร์จไอเทมเซตและให้ $S^D(I) = S^U(I)$ พร้อมทั้งเก็บ I ไว้ใน $S^D(I)$

ถ้า $S^U(I) / (d+t+c) \geq S_l$ กำหนดให้ I เป็นพรีลาร์จไอเทมเซตและให้ $S^D(I) = S^U(I)$ พร้อมเก็บ I ไว้ใน $S^D(I)$

ถ้าไม่ใช่ทั้งสองกรณีข้างต้น ไม่ต้องดำเนินการใดๆ กับไอเทม I

2.6) สำหรับแต่ละไอเทมเซต I ที่อยู่ในพรีลาร์จ k-ไอเทมเซตของฐานข้อมูลเดิม $\mathbf{P}_k^D$ ให้ดำเนินการต่อไปนี้

2.6.1 กำหนดให้ค่าความถี่ใหม่ $S^U(I) = S^T(I) + S^D(I)$

2.6.2 ถ้า $S^U(I) / (d+t+c) \geq S_u$ กำหนดให้ I เป็นลาร์จไอเทมเซตและให้ $S^D(I) = S^U(I)$ พร้อมทั้งเก็บ I ไว้ใน $S^D(I)$

ถ้า $S^U(I) / (d+t+c) \geq S_l$ กำหนดให้ I เป็นพรีลาร์จไอเทมเซตและให้ $S^D(I) = S^U(I)$ พร้อมเก็บ I ไว้ใน $S^D(I)$

ถ้าไม่ใช่ทั้งสองกรณีข้างต้น ไม่ต้องดำเนินการใดๆ กับไอเทม I

2.7) สำหรับแต่ละไอเทมเซต I จากแคนดิเดตไอเทมเซต ที่ไม่อยู่ทั้งใน $\mathbf{L}_k^D$ และ $\mathbf{P}_k^D$ ของฐานข้อมูลเดิม ให้ดำเนินการต่อไปนี้

2.7.1 ถ้า I เป็นลาร์จไอเทมเซต $\mathbf{L}_k^T$ หรือพรีลาร์จไอเทมเซต $\mathbf{P}_k^T$ ของรายการใหม่ที่เพิ่มเข้ามา ให้นำ I ไปเก็บใน rescan-set R ซึ่งจะถูกนำไปใช้ในขั้นตอนที่ 2.8

2.7.2 ถ้า I เป็นสมอลล์ของรายการใหม่ที่เพิ่มเข้ามา ไม่ต้องดำเนินการใดๆ

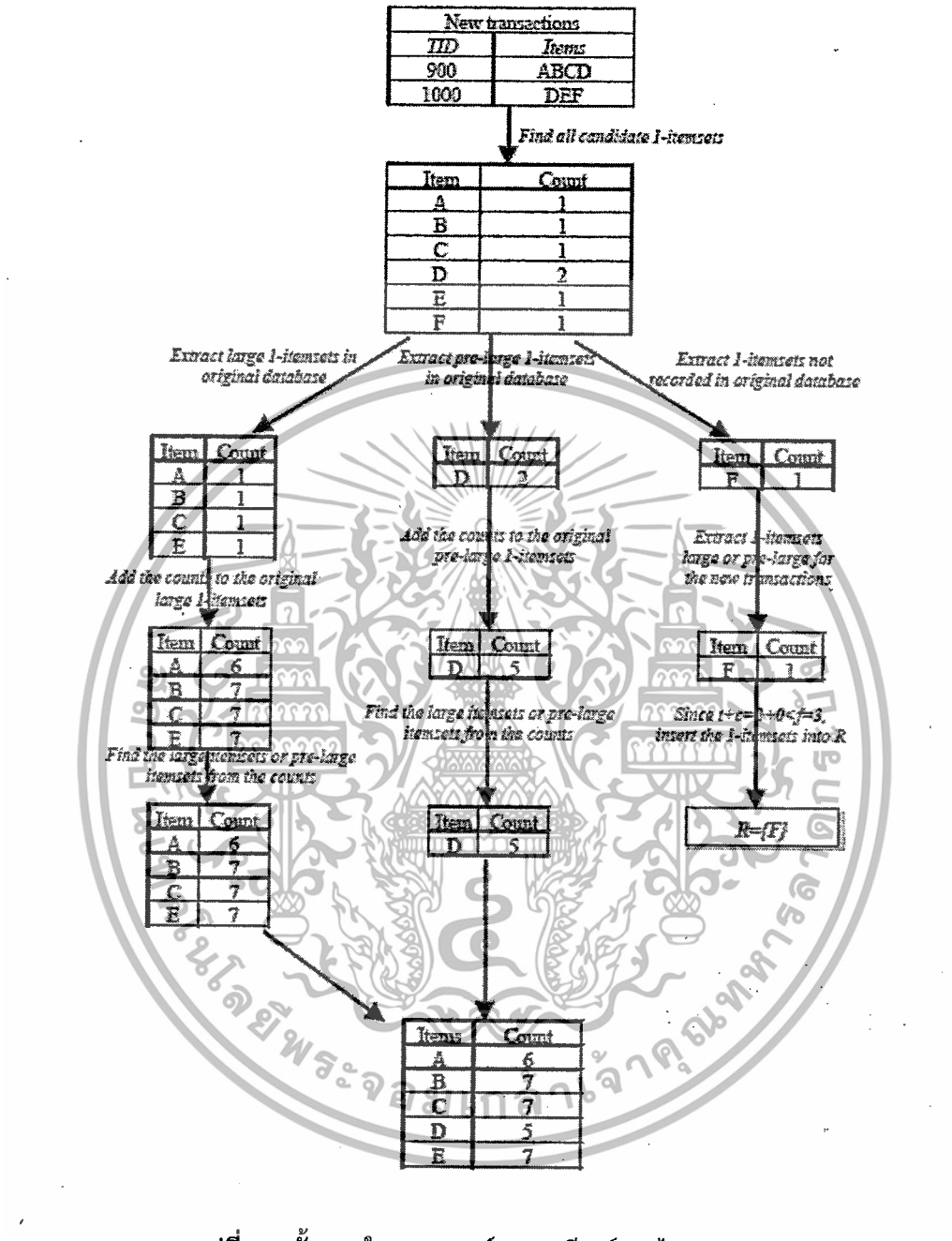
2.8) ถ้า $t + c \leq f$ หรือ R เป็นเซตว่าง ไม่ต้องดำเนินการใดๆ แต่ถ้า Rไม่ใช่เซตว่าง ให้สแกนฐานข้อมูลเดิมใหม่เพื่อหาว่าไอเทมเซตที่อยู่ใน rescan-set R เป็นลาร์จหรือพรีลาร์จ

2.9) สร้าง C_{k+1} จากลาร์จไอเทมเซตและพรีลาร์จไอเทมเซตที่อัพเดทแล้ว $(\mathbf{L}_k^U \cup \mathbf{P}_k^U)$ ด้วยหลักการเชื่อมของอะพริโอริ และหาค่าสนับสนุนของไอเทมเซตเหล่านี้ที่ปรากฏอยู่ในรายการใหม่ที่เพิ่มเข้ามา กำหนดให้ $k = k+1$

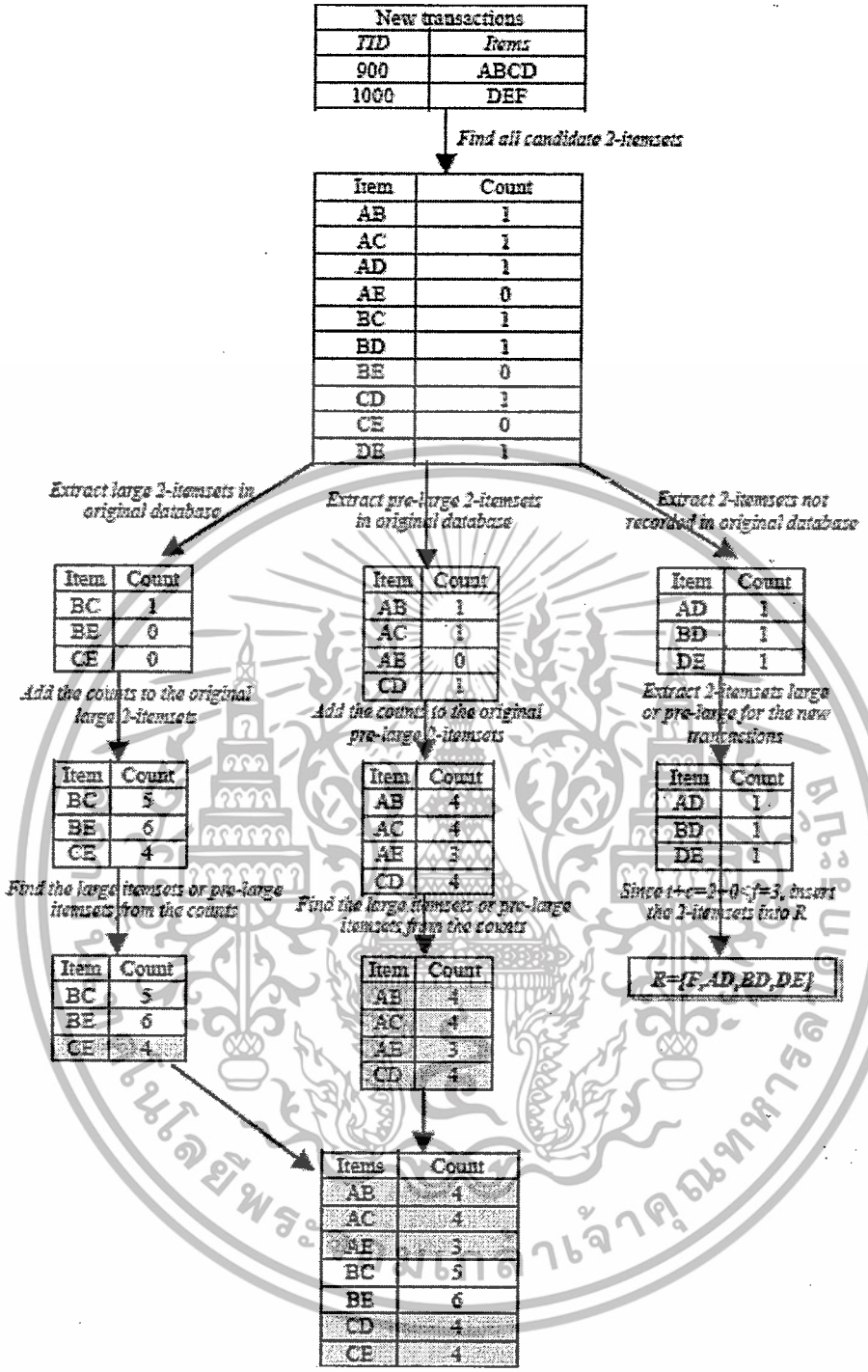
2.10) ทำขั้นตอนที่ 2.4-2.9 ซ้ำ จนกระทั่งไม่พบลาร์จไอเทมเซตหรือพรีลาร์จไอเทมเซตใหม่ จากนั้นปรับปรุงกฎความสัมพันธ์ตามลาร์จไอเทมเซตที่ได้รับการอัพเดท

2.11) ถ้า $t + c > f$ กำหนดให้ $d = d+t+c$ และกำหนดค่า $c = 0$ แต่ถ้าไม่ใช่ ให้กำหนดค่า $c = t+c$

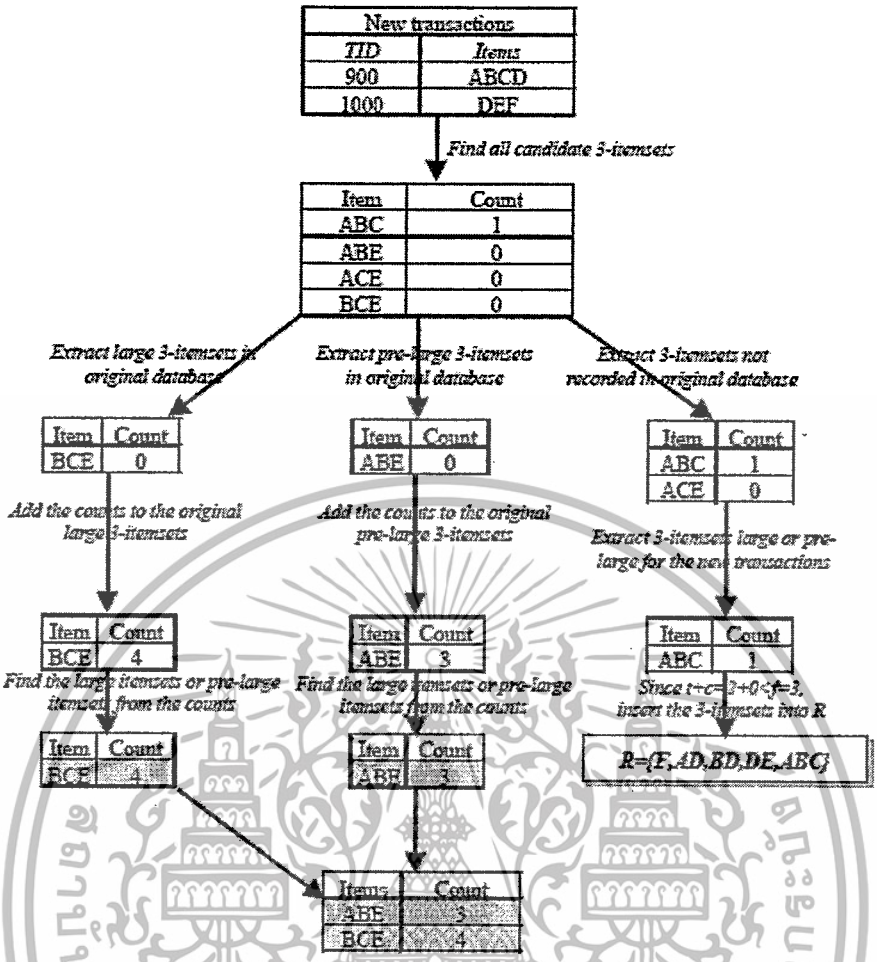
หลังจากเสร็จสิ้นขั้นตอนที่ 2.11 จะได้กฎความสัมพันธ์สุดท้ายจากฐานข้อมูลที่ได้รับการปรับปรุงแล้ว รูปที่ 2.8-2.10 แสดงตัวอย่างผลลัพธ์ที่ได้การทำงานของอัลกอริทึมพรีลาร์จไอเทมเซต เมื่อกำหนดให้ $S_l = 30\%$, $S_u = 50\%$ และใช้ข้อมูลลาร์จและพรีลาร์จไอเทมเซตของฐานข้อมูลเดิมจากตาราง 2.5-2.6 มาใช้ในการอัพเดทฟรีเควนท์ไอเทมเซตเมื่อมีรายการข้อมูลเพิ่มเข้ามาใหม่ 2 รายการ



รูปที่ 2.8 ขั้นตอนในการหาลาร์จและพรีลาร์จ 1-ไอเทมเซต



รูปที่ 2.9 ขั้นตอนในการหาลาร์จและพรีลาร์จ 2-ไอเทมเซต

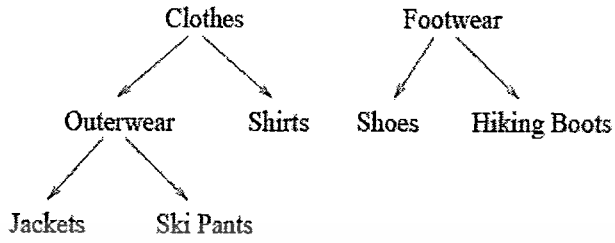


รูปที่ 2.10 ขั้นตอนในการหาลาร์จและพรีลาร์จ 3-ไอเทมเซต

2.4.3 Pre-large Itemsets for Deletion [8]

งานวิจัยนี้นำหลักการของการค้นหากฎความสัมพันธ์ในหลายระดับ (Generalized association rules on multiple levels) [12] มาใช้ร่วมกับแนวคิดพรีลาร์จไอเทมเซตเพื่อบำรุงรักษากฎความสัมพันธ์ในกรณีที่มีรายการข้อมูลถูกลบออกจากฐานข้อมูลเดิม การค้นหากฎความสัมพันธ์ในหลายระดับเป็นการค้นหากฎความสัมพันธ์ของรายการที่มีความเกี่ยวข้องกันซึ่งอาจแสดงอยู่ในรูปแบบของรูปต้นไม้ลำดับชั้น (Hierarchy tree) เพื่อแสดงโครงสร้างการจัดหมวดหมู่ที่กำหนดไว้ล่วงหน้าดังตัวอย่างในรูปที่ 2.11 โดยโหนดปลายทางในรูปต้นไม้หมายถึงไอเทมที่ปรากฏในรายการ โหนดภายในหมายถึงคลาส (Classes) หรือคอนเซปต์ (Concepts) ที่เกิดขึ้นจากโหนดในระดับล่าง ทั้งนี้ กฎความสัมพันธ์ที่เกิดขึ้นอาจมาจากไอเทมในระดับใดของโครงสร้างการจัดหมวดหมู่ข้อมูลก็ได้ เช่น เราอาจอนุมานได้ว่าลูกค้าที่ซื้อ Outerwear มีแนวโน้มจะซื้อ Hiking Boots จากข้อเท็จจริงที่เกิดขึ้นในรายการที่ว่าลูกค้าซื้อ Jackets พร้อม Hiking Boots และซื้อ Ski Pants พร้อม Hiking Bootes อย่างไรก็ตาม ค่าความเชื่อมั่นของกฎ Outerwear ==> Hiking Boots ไม่

จำเป็นต้องเท่ากับผลรวมของ Jackets==> Hiking Boots กับ Ski Pants==> Hiking Boots เนื่องจากอาจมีรายการข้อมูลที่เกิด Jackets, Ski Pants, และ Hiking Boots อยู่ในรายการเดียวกัน



รูปที่ 2.11 ตัวอย่างโครงสร้างการจัดหมวดหมู่ข้อมูล

ขั้นตอนการค้นหากฎความสัมพันธ์ในหลายระดับแบ่งออกเป็น 4 ขั้นตอน คือ

1) Ancestor ของแต่ละไอเทมที่เกิดขึ้นในแต่ละรายการจะถูกเพิ่มเข้ามาตามโครงสร้างข้อมูลที่ได้กำหนดไว้ล่วงหน้าแล้ว เช่น ถ้าปรากฏ Jackets อยู่ในรายการข้อมูล จะเพิ่ม Outerwear และ Clothes เข้าไปในรายการนั้นด้วย

2) สร้างแคนดิเดตไอเทมเซตและสแกนรายการข้อมูลเพื่อหาค่าสนับสนุนของแต่ละไอเทมเซต ทั้งนี้ ไอเทมเซตประกอบขึ้นจากไอเทมที่เป็นได้ทั้งโหนดปลายทางและโหนดภายใน เช่น {Jackets, Footwear}, {Shirts, Shoes} คือตัวอย่างของไอเมเซตที่ได้จากรูป 2.11

3) นำลาร์จไอเทมเซตที่ได้มาสร้างเป็นกฎความสัมพันธ์ กฎที่มีค่าความเชื่อมั่นมากกว่าค่าความเชื่อมั่นน้อยที่สุดจะถูกเก็บไว้

4) กฎที่ไม่น่าสนใจจะถูกตัดทิ้งไป โดยกฎที่เหลืออยู่จะต้องสอดคล้องกับเงื่อนไขต่อไปนี้

4.1) นำ Ancestor ของไอเทมที่อยู่ในกฎมาแทนที่ไอเทมนั้นๆ กฎที่ไม่มี Ancestor rule จะถูกตัดทิ้ง

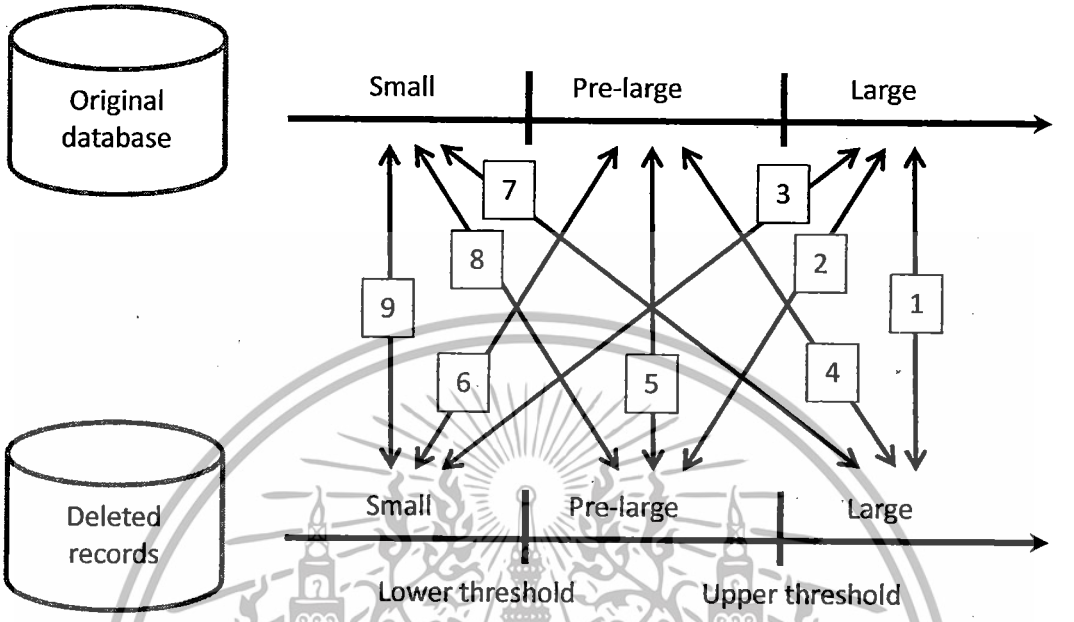
4.2) ค่าสนับสนุนของกฎจะต้องมีค่าเป็น R เท่าของค่าสนับสนุนที่คาดหวังไว้ของ Ancestor rule ของกฎนั้นๆ เมื่อ R คือค่าที่ผู้ใช้กำหนด

4.3) ค่าความเชื่อมั่นของกฎจะต้องมีค่าเป็น R เท่าของค่าความเชื่อมั่นที่คาดหวังไว้ของ Ancestor rule ของกฎนั้นๆ เมื่อ R คือค่าที่ผู้ใช้กำหนด

เพื่อลดความซ้ำซ้อนของกฎที่ได้มา ข้อมูล Taxonomy จะถูกนำมาใช้ในการตัดกฎที่ซ้ำซ้อนออกไป ภายใต้แนวคิดว่า "ถ้าค่าสนับสนุนและค่าความเชื่อมั่นของกฎใดๆ มีค่าใกล้เคียงกับค่าคาดหวังที่เกิดจาก Ancestor ของกฎนั้น จะถือว่ากฎนั้นมีความซ้ำซ้อน" สมมติว่ามีกฎ X => Y และ Z = X U Y ค่าสนับสนุนของ Z จะมีค่าเท่ากับค่าสนับสนุนของ X => Y กำหนดให้ $E_{\hat{Z}}$ [Pr(Z)] คือค่าคาดหวังของความน่าจะเป็นที่จะเกิด Pr(Z) เมื่อเกิด Pr($\hat{Z}$) โดย $\hat{Z}$ คือ Ancestor ของ Z และกำหนดให้ Z = {z_1, z_2,..., z_n}, $\hat{Z}$ = {$\hat{Z}_1$,..., $\hat{Z}_j$, Z_{j+1},..., Z_n} และ $1 \le j \le n$ เมื่อ $\hat{Z}_i$ คือ Ancestor ของ z_i จะสามารถหาค่า $E_{\hat{Z}}$ [Pr(Z)] ได้จากสมการที่ 2.4

$$E_{\hat{Z}}[\Pr(Z)] = \frac{\Pr(z_1)}{\Pr(\hat{z}_1)} \times \cdots \times \frac{\Pr(z_j)}{\Pr(\hat{z}_j)} \times \Pr(\hat{Z}). \qquad (2.4)$$

เมื่อมีการลบรายการข้อมูลออกจากฐานข้อมูลเดิม อาจส่งผลให้เกิดการเปลี่ยนแปลงต่อกฎความสัมพันธ์ที่ได้เคยค้นหาไว้แล้ว ตามแนวคิดของพรีลาร์จกรณีของไอเทมเซตที่ปรากฏเมื่อมีการลบรายการข้อมูลเกิดขึ้นได้ 9 กรณี ดังแสดงในรูป 2.12



รูปที่ 2.12 กรณีของไอเทมเซตที่ปรากฏเมื่อมีการลบรายการข้อมูลออกจากฐานข้อมูลเดิมในอัลกอริทึมพรีลาร์จ

กรณีที่ 2, 3, 4, 7 และ 8 ไม่ส่งผลให้เกิดการเปลี่ยนแปลงต่อกฎความสัมพันธ์
กรณีที่ 1 อาจส่งผลให้กฎความสัมพันธ์บางข้อถูกลบออกไป
กรณีที่ 5, 6 และ 9 อาจส่งผลให้เกิดกฎความสัมพันธ์ใหม่ ซึ่งกรณีที่ 5 และ 6 นั้นสามารถนำความรู้ที่ได้จากการไมนิงไว้แล้วมาอัพเดทค่าสนับสนุนของไอเทมเซตได้

ทั้งนี้โดยปกติแล้วจำนวนรายการที่ถูกลบออกไปจะมีจำนวนน้อยมากเมื่อเทียบกับจำนวนรายการทั้งหมดที่อยู่ในฐานข้อมูลเดิม สัญลักษณ์เพิ่มเติมที่ใช้ในอัลกอริทึมพรีลาร์จนอกเหนือจากที่เคยกล่าวไว้แล้วคือ L_k^T หมายถึง เซตของ k-ไอเทมเซตทั้งหมดใน T ที่เกิดขึ้นใน ($L_k^D \cup P_k^D$) เมื่อ T คือเซตของรายการที่ถูกลบ

เมื่อมีรายการถูกลบออก อัลกอริทึมจะสแกนเฉพาะส่วนของรายการที่ถูกลบออกนี้เพื่อหา 1-ไอเทมเซตและนำไปเปรียบเทียบกับผลลัพธ์ที่ได้จากฐานข้อมูลเดิม หากไอเทมเซตนี้อยู่ในกลุ่มลาร์จหรือพรีลาร์จของฐานข้อมูลเดิม ก็จะสามารถอัพเดทค่าสนับสนุนได้จากข้อมูลเดิมที่มีอยู่แล้ว แต่ถ้าไอเทมเซตนี้ไม่ได้อยู่ในลาร์จหรือพรีลาร์จของฐานข้อมูลเดิม และจำนวนรายการที่ถูกลบออกมีค่ามากกว่า safety threshold ที่คำนวณได้ ฐานข้อมูลเดิมจะถูกสแกนใหม่อีกครั้งเพื่อหาและอัพเดทพรีลาร์จไอเทมเซตที่พบใหม่นี้ โดยมีการเก็บจำนวนรายการที่ถูกลบออกนับจากฐานข้อมูลเดิมถูกสแกนใหม่ครั้งสุดท้ายไว้ในตัวแปร c รายละเอียดขั้นตอนการทำงานของอัลกอริทึม มีดังต่อไปนี้

อินพุต: S_u, S_l, ลาร์จไอเทมเซตและพรีลาร์จไอเทมเซตจากฐานข้อมูลเดิม, เซตของรายการที่ถูกลบออก, โครงสร้างข้อมูลที่กำหนดไว้ล่วงหน้า, ค่าความเชื่อมั่นที่กำหนดไว้ล่วงหน้า λ, ค่าขีดแบ่งความสนใจที่กำหนดไว้ล่วงหน้า α

เอาต์พุต: เซตของกฎความสัมพันธ์จากฐานข้อมูลอัพเดท

ขั้นตอนการทำงาน:

2.1) คำนวณหา Safety number จาก

$$f = \frac{(S_u - S_l)d}{1 - S_u}$$

2.2) เพิ่ม Ancestor ของไอเทมเข้าไปในรายการที่ถูกลบดังตัวอย่างในตารางที่ 2.7 กำหนดให้ k=1 เมื่อ k คือความยาวของไอเทมเซตในการทำงานรอบปัจจุบัน

2.3) หา C_k และค่าสนับสนุนของแต่ละไอเทมเซตจากรายการที่ถูกลบออก ใช้ข้อมูลการไมนิงจากฐานข้อมูลเดิมในการแบ่ง C_k ออกเป็น 3 กลุ่ม ได้แก่ ลาร์จไอเทมเซต, พรีลาร์จไอเทมเซต และสมอลล์ไอเทมเซต

2.4) สำหรับแต่ละไอเทมเซต I ที่อยู่ในลาร์จ k-ไอเทมเซตของฐานข้อมูลเดิม $\mathbf{L}_k^D$ ให้ดำเนินการต่อไปนี้

2.4.1 กำหนดให้ค่าความถี่ใหม่ $S^U(I) = S^D(I) - S^T(I)$

2.4.2 ถ้า $S^U(I) / (d\text{-}t\text{-}c) \geq S_u$ กำหนดให้ I เป็นลาร์จไอเทมเซตและให้ $S^D(I) = S^U(I)$ พร้อมทั้งเก็บ I ไว้ใน $S^D(I)$

ถ้า $S^U(I) / (d\text{-}t\text{-}c) \geq S_l$ กำหนดให้ I เป็นพรีลาร์จไอเทมเซตและให้ $S^D(I) = S^U(I)$ พร้อมเก็บ I ไว้ใน $S^D(I)$

ถ้าไม่ใช่ทั้งสองกรณีข้างต้น ไม่ต้องดำเนินการใดๆ กับ I

2.5) สำหรับแต่ละไอเทมเซต I ที่อยู่ในพรีลาร์จ k-ไอเทมเซตของฐานข้อมูลเดิม $\mathbf{P}_k^D$ ให้ดำเนินการต่อไปนี้

2.5.1 กำหนดให้ค่าความถี่ใหม่ $S^U(I) = S^D(I) - S^T(I)$

2.5.2 ถ้า $S^U(I) / (d\text{-}t\text{-}c) \geq S_u$ กำหนดให้ I เป็นลาร์จไอเทมเซตและให้ $S^D(I) = S^U(I)$ พร้อมทั้งเก็บ I ไว้ใน $S^D(I)$

ถ้า $S^U(I) / (d\text{-}t\text{-}c) \geq S_l$ กำหนดให้ I เป็นพรีลาร์จไอเทมเซตและให้ $S^D(I) = S^U(I)$ พร้อมเก็บ I ไว้ใน $S^D(I)$

ถ้าไม่ใช่ทั้งสองกรณีข้างต้น ไม่ต้องดำเนินการใดๆ กับ I

2.6) สำหรับแต่ละไอเทมเซต I จากแคนดิเดตไอเทมเซต ที่ไม่อยู่ทั้งใน $\mathbf{L}_k^D$ และ $\mathbf{P}_k^D$ ของฐานข้อมูลเดิม ให้ดำเนินการต่อไปนี้

2.6.1 ถ้า I เป็นลาร์จไอเทมเซต $\mathbf{L}_k^T$ หรือพรีลาร์จไอเทมเซต $\mathbf{P}_k^T$ ของรายการที่ถูกลบออก ไม่ต้องดำเนินการใดๆ

2.6.2 ถ้า I เป็นสมอลล์ของรายการใหม่ที่เพิ่มเข้ามา ให้นำ I ไปเก็บใน rescan-set R ซึ่งจะถูกนำไปใช้ในขั้นตอนที่ 2.7

2.7) ถ้า $t + c \leq f$ หรือ R เป็นเซตว่าง ไม่ต้องดำเนินการใดๆ แต่ถ้า R ไม่ใช่เซตว่าง ให้สแกนฐานข้อมูลเดิมใหม่เพื่อหาว่าไอเทมเซตที่อยู่ใน rescan-set R เป็นลาร์จหรือพรีลาร์จ

2.8) สร้าง C_{k+1} จากลาร์จไอเทมเซตและพรีลาร์จไอเทมเซตที่อัพเดทแล้ว ($L_k^U \cup P_k^U$) ด้วยหลักการเชื่อมของอะพริโอริ โดย C_2 ที่เกิดจากไอเทมเซตและ Ancestor ของตัวมันเองจะถูกตัดทิ้ง จากนั้นกำหนดให้ k = k+1

2.9) ทำขั้นตอนที่ 2.3-2.8 ซ้ำ จนกระทั่งไม่พบลาร์จไอเทมเซตหรือพรีลาร์จไอเทมเซตใหม่ จากนั้นปรับปรุงกฎความสัมพันธ์ตามลาร์จไอเทมเซตที่ได้รับการอัพเดทโดยกฎที่ได้ต้องมีค่าความเชื่อมั่นมากกว่าหรือเท่ากับค่าความเชื่อมั่นที่กำหนดไว้ ผลลัพธ์ที่ได้คือกฎความสัมพันธ์ที่ไม่มี Ancestor rule

2.10) หากฎ Close ancestor y สำหรับแต่ละกฎ x และคำนวณค่าสนับสนุนความน่าสนใจและค่าความเชื่อมั่นความน่าสนใจของกฎจาก

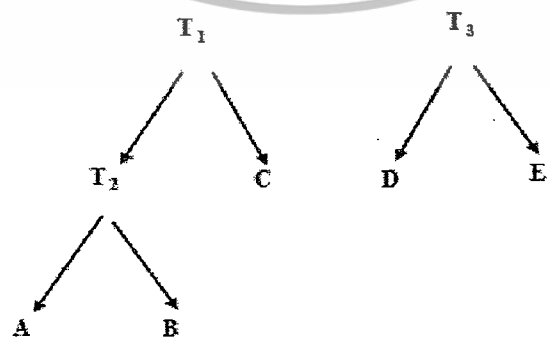
$$I_{support}(x) = \frac{count_x}{\frac{\prod_{k=1}^{r+1} count_{x_k}}{\prod_{k=1}^{r+1} count_{y_k}} \times count_y} \quad (2.5)$$

$$I_{confidence}(x) = \frac{confidence_x}{\frac{count_{x_{r+1}}}{count_{y_{r+1}}} \times confidence_y} \quad (2.6)$$

2.11) กฎที่มีค่าสนับสนุนความน่าสนใจและค่าความเชื่อมั่นความน่าสนใจมากกว่าหรือเท่ากับค่าขีดแบ่งความน่าสนใจ α ที่กำหนดไว้ จะถือเป็นกฎที่น่าสนใจ

2.12) ถ้า $t + c \geq f$ กำหนดให้ d = d-t-c และกำหนดค่า c = 0 แต่ถ้าไม่ใช่ ให้กำหนดค่า c = t+c

หลังจากเสร็จสิ้นขั้นตอนที่ 2.12 จะได้กฎความสัมพันธ์สุดท้ายจากฐานข้อมูลอัพเดท ตารางที่ 2.7 – 2.9 และรูปที่ 2.18 แสดงตัวอย่างของฐานข้อมูลเดิม, โครงสร้างการจัดหมวดหมู่ข้อมูลที่กำหนดไว้ล่วงหน้า, การเพิ่ม Ancestor เข้าไปในรายการข้อมูลที่ถูกลบออก, และลาร์จไอเทมเซตของฐานข้อมูลอัพเดท



รูปที่ 2.13 โครงสร้างการจัดหมวดหมู่ข้อมูลที่กำหนดไว้ล่วงหน้า

ตารางที่ 2.7 แสดงตัวอย่างของฐานข้อมูลเดิม

TID	ITEMS
100	A
200	A, E
300	B, E
400	A, B, D
500	D
600	A, B
700	A, C, E
800	B, D
900	C, D, E
1000	A, D, E

ตารางที่ 2.8 แสดงตัวอย่างการเพิ่ม ancestor เข้าไปในรายการข้อมูลที่ถูกลบออก

TID	Items
900	C, D, E, T_1, T_3
1000	A, D, E, T_3, T_2, T_1

ตารางที่ 2.9 แสดงลาร์จไอเทมเซตของฐานข้อมูลอัพเดท

1-itemset	2-itemset	3-itemset
{T_1}	{T_1, T_3}	None
{T_2}	{T_2, T_3}	
{T_3}		
{A}		

ข้อดีของ Pre-large itemsets

1) มีการเก็บผลลัพธ์จากการค้นหากฎความสัมพันธ์ในฐานข้อมูลเดิมมาใช้ร่วมกับฐานข้อมูลใหม่ที่เพิ่มเข้ามา

2) สามารถใช้งานได้ทั้งในกรณีของการเพิ่มและการลบรายการข้อมูล

3) มีการตรวจสอบจำนวนรายการที่เพิ่มเข้ามาหรือจำนวนรายการที่ถูกลบออกว่าเพียงพอที่จะส่งผลให้เกิดการเปลี่ยนแปลงของสมอลล์ไอเทมเซตในฐานข้อมูลเดิมหรือไม่ ทำให้ไม่ต้องสแกนฐานข้อมูลเดิมทุกครั้งที่มีรายการใหม่เกิดขึ้น

ข้อจำกัดของ Pre-large itemsets

1) ต้องสแกนฐานข้อมูลเดิมทุกรอบ k ในกรณีที่ต้องการอัพเดทค่าสนับสนุนของไอเทมเซตที่เป็นสมอลล์ในฐานข้อมูลเดิม

2) ไม่สามารถเพิ่มและลบรายการข้อมูลพร้อมกันได้

2.4.4 Probability-based Incremental Association Rule Discovery Algorithm [9]

อัลกอริทึมนี้มีแนวคิดพื้นฐานคล้ายกับพรีลาร์จไอเทมเซต คือ มีการเก็บทั้งฟรีเควนท์ไอเทมเซตและไอเทมเซตที่มีแนวโน้มจะเป็นฟรีเควนท์ไอเทมเซต (expected frequent itemsets: EF) เมื่อมีรายการใหม่เพิ่มเข้ามา โดยการนำหลักทางสถิติของเบอร์นูลลี (Bernoulli trials) มาใช้ในการคาดคะเนไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ไอเทมเซต ซึ่งในการทดลองจะประกอบด้วยผลการทดลอง 2 เหตุการณ์คือ เหตุการณ์ที่ประสบความสำเร็จหรือความไม่สำเร็จ ในที่นี้ถ้าพบว่าไอเทมเซตใดเป็นฟรีเควนท์ไอเทมเซตหมายถึงเหตุการณ์ที่เกิดความสำเร็จ และเหตุการณ์ที่ไอเทมเซตใดเป็นอินฟรีเควนท์ไอเทมเซตถือเป็นเหตุการณ์ที่ไม่สำเร็จ

ด้วยแนวคิดของกฎว่าด้วยจำนวนมาก (Law of large number) ของเบอร์นูลลีที่กล่าวไว้ว่า "ค่าเฉลี่ยของตัวแปรสุ่มของตัวอย่างประชากรจำนวนมากจะมีค่าเข้าใกล้ค่าเฉลี่ยของประชากรทั้งหมด" อัลกอริทึมนี้จะทำการหาฟรีเควนท์ไอเทมเซตจากฐานข้อมูลเดิมซึ่งมีจำนวน n รายการ และนำหลักของเบอร์นูลลีมาใช้ในการคาดคะเนหาไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ โดยการนำค่าสนับสนุนของไอเทมเซตในฐานข้อมูลเดิมมาคำนวณหาค่าความน่าจะเป็นที่ไอเทมเซตใดๆ จะมีค่าสนับสนุนไม่น้อยกว่า k เมื่อ มีรายการใหม่เพิ่มเข้ามา m รายการ ซึ่งคำนวณได้จากสมการดังต่อไปนี้

$$P(x \geq k)_{itemset} = 1 - P(x < k)_{itemset} \quad (2.7)$$

เมื่อ k = ค่าสนับสนุนน้อยที่สุดของ (n+m) รายการ และ $P(x < k)_{itemset}$ คือค่าความน่าจะเป็นที่จะเกิดไอเทมเซต x ในฐานข้อมูลอัพเดทอย่างน้อย k-ค่า ซึ่งคำนวณได้จาก

$$P(x < k)_{itemset} = \sum_{x=0}^{k-1} \binom{n+m}{x} \cdot P_{itemset}^{x} \cdot (1-P_{itemset})^{n+m-x} \quad (2.8)$$

เมื่อ

$$P(x)_{itemset} = \binom{n+m}{x} \cdot P_{itemset}^{x} \cdot (1-P_{itemset})^{n+m-x} \quad (2.9)$$

จากนั้นจะนำค่าที่คำนวณได้ไปเปรียบเทียบกับค่าความน่าจะเป็นน้อยที่สุดที่คาดว่าไอเทมเซตจะกลายเป็นฟรีเควนท์ ($Prob_{pt}$) ซึ่งเป็นค่าขีดแบ่ง (Threshold) ที่ผู้ใช้กำหนดไว้ $Prob_{pt}$ จะเป็นค่าที่แสดงถึงระดับความเชื่อมั่นที่ EF จะมีโอกาสเป็นฟรีเควนท์ไอเทมเซตเมื่อมีรายการใหม่เพิ่มเข้ามา ยิ่งกำหนดค่า $Prob_{pt}$ ไว้สูง จำนวน EF ที่เก็บไว้ยิ่งน้อยลง รูปที่ 2.14 แสดงการทำนายค่าคาดหวังของไอเทมเซตที่คาดว่าจะกลายเป็นฟรีเควนท์ไอเทมเซตด้วยการทดลองแบบเบอร์นูลลี

รูปที่ 2.15 แสดงตัวอย่างการคำนวณเมื่อกำหนดให้ฐานข้อมูลเดิมมีจำนวน 10 รายการ มีรายการใหม่ถูกเพิ่มเข้ามา 5 รายการ ค่าสนับสนุนน้อยที่สุด = 0.4 และ $Prob_{pt}$ = 0.1 ผลลัพธ์ที่ได้คือ {A, B, C} เป็นฟรีเควนท์ 1-ไอเทมเซต, {E} เป็นไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ 1-ไอเทมเซต เนื่องจากค่าความน่าจะเป็นที่คำนวณได้มีค่ามากกว่า $Prob_{pt}$

ฟรีเควนท์ k-ไอเทมเซตและไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ที่หามาได้จะถูกนำไปสร้างแคนดิเดต (k+1)-ไอเทมเซตด้วยหลักการเชื่อมของอะพริโอริ จากนั้นก็ทำการหาฟรีเควนท์ (k+1)-ไอเทมเซตและไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ตามวิธีการที่กล่าวมาแล้วข้างต้น ทำเช่นนี้ไปเรื่อยๆ

จนกว่าจะไม่สามารถสร้าง C_k ในระดับต่อไปได้แล้ว รูปที่ 2.16 แสดงตัวอย่างของการสร้าง C_k ในฐานข้อมูลเดิม

เมื่อมีรายการใหม่ถูกเพิ่มเข้ามาในฐานข้อมูลเดิม อาจส่งผลให้กฎความสัมพันธ์ที่ค้นพบก่อนหน้านี้มีการเปลี่ยนแปลง ดังนั้น การอัพเดทค่าฟรีเควนท์ไอเทมเซตและไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์จึงเป็นขั้นตอนที่สำคัญมาก ความหมายของสัญลักษณ์ที่ใช้ในอัลกอริทึมมีดังนี้

DB หมายถึง ฐานข้อมูลเดิม

db หมายถึง ฐานข้อมูลส่วนเพิ่มหรือรายการที่เพิ่มเข้ามา

UP หมายถึง ฐานข้อมูลอัพเดท

k หมายถึง จำนวนไอเทมเซต

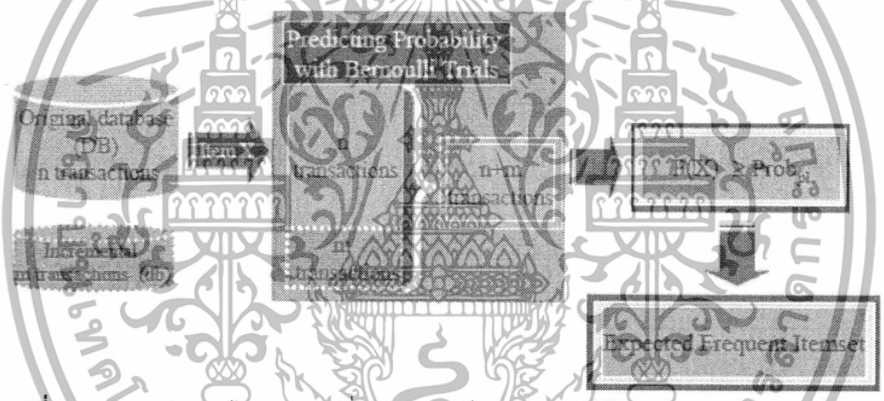
σ หมายถึง ค่าสนับสนุนน้อยที่สุด

ρ หมายถึง ค่าคาดหวังน้อยที่สุดที่ไอเทมจะกลายเป็นฟรีเควนท์

C_k หมายถึง แคนดิเดต k-ไอเทมเซต

F_k หมายถึง ฟรีเควนท์ k-ไอเทมเซต

EF_k หมายถึง ไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์



รูปที่ 2.14 การทำนายไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ด้วยหลักการของเบอร์นูลลี

TID	List of item
1	A, B, E
2	B, D
3	B,C
4	A, B,D
5	A, C
6	B, C
7	A, C
8	A, B, C, E
9	A, B, E
10	A,C

$$P(x \geq 6)_A = 1 - \sum_{x=0}^{5} \binom{15}{x} \frac{7}{10}^{x} \frac{3}{10}^{15-x} = 1$$

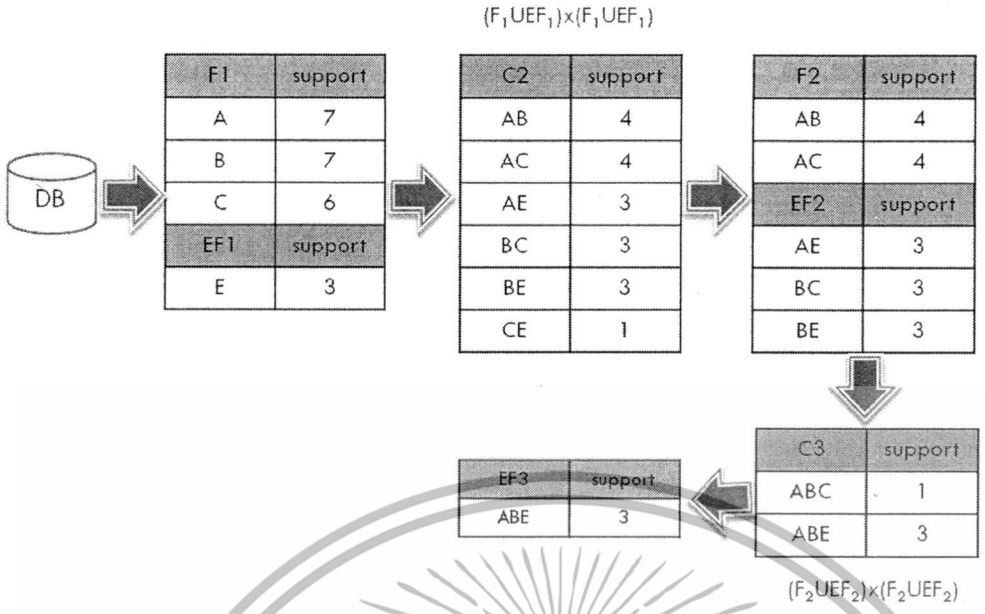
$$P(x \geq 6)_B = 1 - \sum_{x=0}^{5} \binom{15}{x} \frac{7}{10}^{x} \frac{3}{10}^{15-x} = 1$$

$$P(x \geq 6)_C = 1 - \sum_{x=0}^{5} \binom{15}{x} \frac{6}{10}^{x} \frac{4}{10}^{15-x} = 1$$

$$P(x \geq 6)_D = 1 - \sum_{x=0}^{5} \binom{15}{x} \frac{2}{10}^{x} \frac{8}{10}^{15-x} = 0.06$$

$$P(x \geq 6)_E = 1 - \sum_{x=0}^{5} \binom{15}{x} \frac{3}{10}^{x} \frac{7}{10}^{15-x} = 0.28$$

รูปที่ 2.15 ตัวอย่างรายการที่เกิดขึ้นในฐานข้อมูลเดิมและตัวอย่างการคำนวณหาค่าความน่าจะเป็นของแคนดิเดต 1-ไอเทมเซต



รูปที่ 2.16 ตัวอย่างการสร้างแคนดิเดต k-ไอเทมเซตของฐานข้อมูลเดิมในอัลกอริทึมการเพิ่มขยายการค้นหากฎความสัมพันธ์โดยอาศัยหลักความน่าจะเป็น

Algorithm1 : Main Algorithm

Input : $DB, db, k, \delta^{UP}, \rho^{UP}, \rho^{DB}, C_1^{DB}, F_1^{DB}, EF_1^{DB}$ *and their count*

Output : F_k^{UP}, EF_k^{UP}

1. $k=1$
2. *if* $k=1$
3. *Update* 1−*itemset*
4. $k=k+1$
5. *else*
6. *for* $(k=2; F_k^{UP} \neq \phi; k++)$ *do*
7. *Generate Candidate Itemset*
8. *Update* $k-$ *itemset*(*return m, Temp _ scanDB*)
9. *// m is the maximum itemset of Temp _ scanDB*
10. $k=k+1$
11. *end do*
12. *end if*
13. $k=2$
14. *while* $(Temp\_scanDB_k \neq \phi$ *and* $(k \leq m)$ *do*
15. *Scan Original Database*$(Temp\_scanDB_k)$
16. $k=k+1$
17. *end do*
18. *clear Temp _ scanDB*

รูปที่ 2.17 อัลกอริทึมหลักในการเพิ่มขยายการค้นหากฎความสัมพันธ์โดยอาศัยหลักความน่าจะเป็น

อัลกอริทึมหลักในการอัพเดท k-ไอเทมเซต (รูปที่ 2.17) จะแบ่งการทำงานออกเป็น 3 ส่วน คือ 1) การอัพเดทฟรีเควนท์ 1-ไอเทมเซตและไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ 2) การอัพเดทฟรีเควนท์ k-ไอเทมเซตและไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ (k ≥ 2) โดยสแกนเฉพาะรายการข้อมูลส่วนที่ถูกเพิ่มเข้ามาใหม่ 3) การสแกนฐานข้อมูลเดิม ซึ่งแต่ละส่วนมีรายละเอียดการทำงานดังนี้

1) การอัพเดทฟรีเควนท์ 1-ไอเทมเซตและ 1-ไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ (รูปที่ 2.18)

1.1) ทำการสแกนรายการที่เพิ่มเข้ามา (db) เพื่อหา C_1^{db} และค่าความสนับสนุน c(X,db)

1.2) อัพเดทค่าสนับสนุน c(X,UP) โดยนำค่าสนับสนุนที่ได้จากข้อ 1.1 ไปรวมกับค่าสนับสนุนของไอเทมเซตในฐานข้อมูลเดิม (DB)

1.3) ทำการพิจารณาเพื่อหาไอเทมเซตที่เป็นฟรีเควนท์ 1-ไอเทมเซตและ EF_1 โดยกรณีที่ $c(X,UP) \geq \sigma^{UP}$ ให้ไอเทมเซตนั้นๆ เป็นฟรีเควนท์ 1-ไอเทมเซตและเพิ่มไอเทมนั้นเข้าไปใน F_1^{UP} ส่วนกรณีที่ $\rho^{UP} \leq c(X,UP) < \sigma^{UP}$ ให้ไอเทมเซตนั้นๆ เป็น EF_1^{UP}

Algorithm 2 : Updating 1-itemsets

Input : $DB, db, \sigma^{UP}, \rho^{UP}, C_1^{DB}, F_1^{DB}, EF_1^{DB}, C_1^{db}$ *and their count*

Output : $F_1^{UP}, EF_1^{UP}, C_1^{UP}$ *and their count*

1. *Scan db and find count* $c(X,db)$ *for all* $X \in C_1^{DB} \cup C_1^{db}$
2. *for all* $X \in C_1^{DB} \cup C_1^{db}$ *do*
3. $c(X,UP) = c(X,DB) + c(X,db)$
4. *end do*
5. $F_1^{UP} = \left\{ X \in C_1^{UP} \middle| c(X,UP) \geq \sigma^{UP} \right\}$
6. $EF_1^{UP} = \left\{ X \in C_1^{UP} \middle| \rho^{UP} \leq c(X,UP) < \sigma^{UP} \right\}$

รูปที่ 2.18 การอัพเดท 1-ไอเทมเซตในอัลกอริทึมการเพิ่มขยายการค้นหากฎความสัมพันธ์โดยอาศัยหลักความน่าจะเป็น

2) การอัพเดทฟรีเควนท์ k-ไอเทมเซตและ k-ไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ สำหรับ $k \geq 2$ โดยสแกนเฉพาะข้อมูลส่วนที่ถูกเพิ่มเข้ามาใหม่

2.1) การสร้างแคนดิเดต k-ไอเทมเซต ขั้นตอนนี้จะทำการสร้างแคนดิเดตไอเทมเซตและหาไอเทมเซตใหม่ (C_2^{new}) ที่เพิ่งปรากฏขึ้นมาในฐานข้อมูลส่วนเพิ่ม (รูปที่ 2.19) โดยแบ่งเป็น 2 กรณี คือ

- กรณี k = 2, หาแคนดิเดตไอเทมเซต C_2^{db} ได้จากการนำ F_1^{UP} มาเชื่อมกัน $C_2^{db} = F_1^{UP} * F_1^{UP}$ จากนั้นนำแคนดิเดตไอเทมเซตที่ได้มาทำการตรวจสอบกับฟรีเควนท์ 2-ไอเทมเซตและ 2-ไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ในฐานข้อมูลเดิม ถ้าไอเทมเซตนั้นๆ ไม่ได้อยู่ในฟรีเควนท์ 2-ไอเทมเซตและ 2-ไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ในฐานข้อมูลเดิม แสดงว่าไอเทมเซตนั้นเป็นไอเทมเซตใหม่ (C_2^{new}) ที่เพิ่งปรากฏขึ้นมาในฐานข้อมูลส่วนเพิ่ม

- กรณี k > 2, หาแคนดิเดตไอเทมเซตจาก $C_k^{db} = F_{k-1}^{db} * F_{k-1}^{db}$ เนื่องจากแคนดิเดต k-ไอเทมเซตจากฐานข้อมูลส่วนเพิ่มจะเป็นอัพเดทฟรีเควนท์ไอเทมเซต F_k^{UP} ได้ก็ต่อเมื่อสับเซตของไอเทมเซตนั้นๆ เป็น (k-1)-อัพเดทฟรีเควนท์ไอเทมเซต ดังนั้นไอเทมเซตใหม่ C_k^{new} จะสามารถหาได้โดยการตรวจสอบว่าไอเทมเซตนั้นไม่อยู่ใน $F_k^{DB} \cup EF_k^{DB}$ แต่อยู่ใน F_{k-1}^{UP} ซึ่งจะเป็นการตัดไอเทมเซตที่ไม่สามารถเป็น k-อัพเดทฟรีเควนท์ไอเทมเซตออกไปจากฐานข้อมูลส่วนเพิ่มได้

Algorithm 3: Generating Candidate k- itemsets

Input : $F_1^{UP}, F_{k-1}^{UP}, F_{k-1}^{db}, k$

Output : C_k^{new}

1. *if* $k = 2$ *then*
2. *if* $(length(F_1^{UP}) \geq 2)$ *then*
3. $C_2^{db} = F_1^{UP} * F_1^{UP}$
4. *for all* $X \in C_2^{db}$ *do*
4. $C_2^{new} = \{X \in C_2^{db} \mid X \in (F_2^{DB} \cup EF_2^{DB})\}$
5. *end do*
6. *end if*
7. *else if* $k > 2$ *then*
8. $C_k^{db} = F_{k-1}^{db} * F_{k-1}^{db}$
9. *for all* $X \in C_k^{db}$ *do*
10. $C_k^{new} = \{X \in C_k^{db} \mid X \in F_{k-1}^{UP} \text{ and } X \in (F_k^{DB} \cup EF_k^{DB})\}$
11. *end do*
12. *end if*

รูปที่ 2.19 การสร้าง k-ไอเทมเซตในอัลกอริทึมการเพิ่มขยายการค้นหากฎความสัมพันธ์โดยอาศัยหลักความน่าจะเป็น

2.2) อัพเดทค่าสนับสนุนของฟรีเควนท์ k-ไอเทมเซตและ k-ไอเทมเซตที่คาดว่าจะเป็นฟรีเคนท์ เมื่อ k ≥ 2 (รูป 2.20)

สแกนฐานข้อมูลส่วนเพิ่มเพื่อหาค่าสนับสนุน c(X,db) และ c(Y,db) เมื่อ X คือไอเทมเซตที่อยู่ใน $F_k^{DB} \cup EF_k^{DB}$ และ Y คือไอเทมเซตใหม่ C_k^{new} จากนั้นอัพเดทค่าสนับสนุนของอัพเดทแคนดิเดต k-ไอเทมเซต C_k^{UP} ซึ่งสามารถแบ่งออกได้เป็น 3 กรณี คือ

กรณีที่ 1 ไอเทมเซต X อยู่ใน $F_k^{DB} \cup EF_k^{DB}$, c(X,UP) = c(X,DB) + c(X,db)

กรณีที่ 2 ไอเทมเซต X อยู่ใน $F_k^{DB} \cup EF_k^{DB}$ และ X ไม่ได้อยู่ใน C_k^{new}, c(X,UP) = c(X,DB)

กรณีที่ 3 ไอเทมเซต X อยู่ C_k^{new} แต่ไม่ได้อยู่ใน $F_k^{DB} \cup EF_k^{DB}$ ในกรณีนี้จะไม่สามารถอัพเดทค่าสนับสนุนของไอเทมเซตได้โดยตรง เนื่องจากไม่ทราบค่าสนับสนุนของไอเทมเซตนั้นๆ ในฐานข้อมูลเดิม แต่สามารถอนุมานได้ว่าค่าสนับสนุนสูงสุดที่เป็นไปได้ของไอเทมเซตนั้นในฐานข้อมูลเดิมคือ ρ^{DB}-1 ดังนั้นค่าสนับสนุนอัพเดทสูงสุดที่เป็นไปได้ของไอเทมเซตใดๆ คือ c(X,db) + (ρ^{DB}-1) ซึ่งถ้ามีค่าน้อยกว่าค่าสนับสนุนน้อยที่สุดก็แสดงว่าไอเทมเซตนั้นไม่เป็นอัพเดทฟรีเควนท์ไอเทมเซต แต่

ถ้าค่าสนับสนุนมีค่ามากกว่าหรือเท่ากับค่าสนับสนุนน้อยที่สุดก็แสดงว่าไอเทมเซตนั้นๆ มีโอกาสที่จะเป็นฟรีเควนท์ไอเทมเซต ไอเทมเซตนั้นจะถูกเก็บไว้ที่ Temp_scanDB เพื่อนำไปสแกนหาค่าสนับสนุนที่แท้จริงในฐานข้อมูลเดิมต่อไป

หลังจากอัพเดทค่าสนับสนุนของไอเทมเซตแล้ว ก็ทำการพิจารณาเพื่อหาไอเทมเซตที่เป็นฟรีเควนท์ k-ไอเทมเซตและ k-ไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ ดังนี้

- กรณีที่ $c(X,UP) \geq \sigma^{UP}$ ให้ไอเทมเซตนั้นเป็นฟรีเควนท์ k-ไอเทมเซต และเพิ่มไอเทมเซตนั้นเข้าไปใน F_k^{UP}
- กรณีที่ $\rho^{UP} \leq c(X,UP) < \sigma^{UP}$ ให้ไอเทมเซตนั้นเป็น k-ไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ และเพิ่มไอเทมเซตนั้นเข้าไปใน EF_k^{UP}

3) การสแกนฐานข้อมูลเดิม (รูปที่ 2.21)

3.1) ทำการสแกนฐานข้อมูลเดิมเพื่อหาสนับสนุนที่แท้จริงของไอเทมเซตที่อยู่ใน Temp_scanDB

3.2) อัพเดทค่าสนับสนุน c(X,UP) = c(X,DB) + c(X,db)

3.3) นำค่าสนับสนุนที่ได้มาพิจารณาว่าเป็นฟรีเควนท์ k-ไอเทมเซตหรือ k-ไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์หรือไม่ ดังนี้ กรณีที่ $c(X,UP) \geq \sigma^{UP}$ ให้ไอเทมเซตนั้นๆ ฟรีเควนท์ k-ไอเทมเซตและเพิ่มไอเทมเซตนั้นเข้าไปใน F_k^{new} ส่วนกรณีที่ $\rho^{UP} \leq c(X,UP) < \sigma^{UP}$ ให้ไอเทมเซตนั้นๆ เป็น k-ไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์และเพิ่มไอเทมนั้นเซตเข้าไปใน EF_k^{new}

3.4) อัพเดทฟรีเควนท์ k-ไอเทมเซตและ k-ไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ โดยให้ $F_k^{UP} = F_k^{UP} \cup F_k^{new}$ และ $EF_k^{UP} = EF_k^{UP} \cup EF_k^{new}$

Algorithm 4 : Update ($k \geq 2$) itemset

Input: $DB, db, \sigma^{UP}, \rho^{UP}, \rho^{DB}, F_k^{DB}, EF_k^{DB}$ *and their count*

Output: F_k^{UP} *and* EF_k^{UP}, F_k^{db}, *Temp_scanDB and their count, m*

1. *Scan db and find count c(X,db) and c(Y,db)*
2. $\forall X \in \left(F_k^{DB} \cup EF_k^{DB}\right)$ *and* $Y \in C_k^{new}$
3. *for all* $X \in \left(F_k^{DB} \cup EF_k^{DB} \cup C_k^{new}\right)$ *do*
4. *if* $X \in \left(F_k^{DB} \cup EF_k^{DB}\right)$ *and* $X \in C_k^{new}$ *then*
5. $c(X,UP) = c(X,DB) + c(X,db)$
6. *elseif* $X \in \left(F_k^{DB} \cup EF_k^{DB}\right)$ *and* $X \notin C_k^{new}$ *then*
7. $c(X,UP) = c(X,DB)$
8. *elseif* $X \notin \left(F_k^{DB} \cup EF_k^{DB}\right)$ *and* $X \in C_k^{new}$ *then*
9. $Temp\_scanDB_k = \{X \mid (c(X,db) + (\rho^{DB} - 1)) \geq \sigma^{UP}\}$
10. *end if*
11. *end do*
12. $F_k^{UP} = \{X \mid c(X,UP) \geq \sigma^{UP}\}$
13. $EF_k^{UP} = \{X \mid \rho^{UP} \leq c(X,UP) < \sigma^{UP}\}$

รูปที่ 2.20 การอัพเดท (k≥2) ไอเทมเซตในอัลกอริทึมการการค้นหากฎความสัมพันธ์แบบเพิ่มขยายโดยอาศัยหลักความน่าจะเป็น

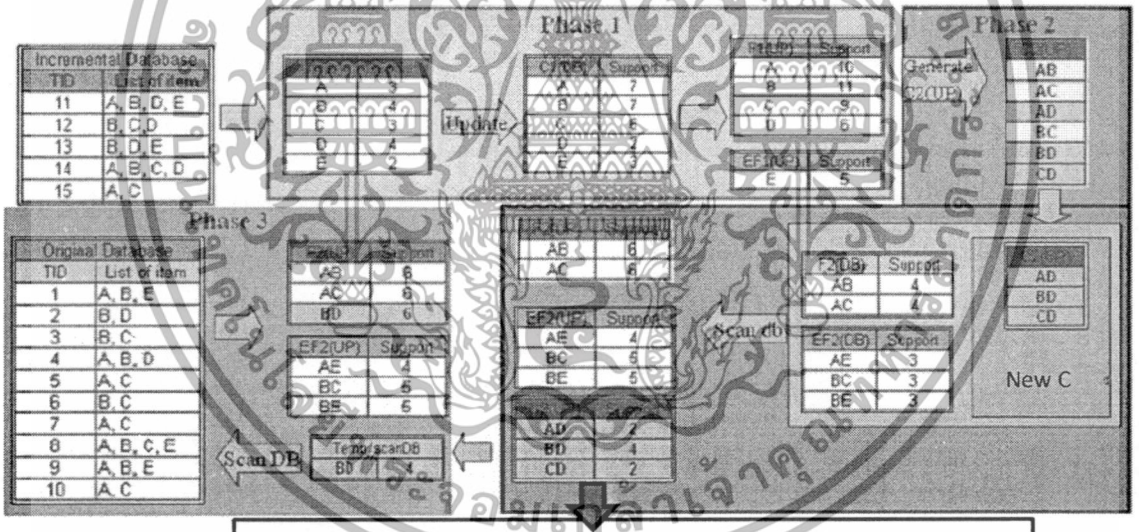
Algorithm 5 : Scanning an original database

Input : $Temp\_scanDB_k, \sigma^{UP}, \rho^{UP}, F_k^{UP}, EF_k^{UP}$ *and their count*

Output : F_k^{UP}, EF_k^{UP} *and their count*

1. *Scan DB and obtain count* $c(X, DB)$ *for all* $Temp\_scanDB_k$
2. *for all* $X \in Temp\_scanDB_k$ *do*
3. $\quad c(X, UP) = c(X, DB) + c(X, db)$
4. *end do*
5. $F_k^{new} = \{X \mid X \in Temp\_scanDB_k \text{ and } c(X, UP) \geq \sigma^{UP}\}$
6. $EF_k^{new} = \{X \mid X \in Temp\_scanDB_k \text{ and } \rho^{UP} \leq c(X, UP) < \sigma^{UP}\}$
7. $F_k^{UP} = F_k^{UP} \cup F_k^{new}$
8. $EF_k^{UP} = EF_k^{UP} \cup EF_k^{new}$

รูปที่ 2.21 การสแกนฐานข้อมูลเดิมในอัลกอริทึมการเพิ่มขยายการค้นหากฎความสัมพันธ์โดยอาศัยหลักความน่าจะเป็น



$Min\_sup^{UP} = 6$

$Min\_sup^{db} = 2$

$Min\_sup_{Prob\_pl} = 3$

{AD,BD,CD} are new C_2 in the updated database

If support + (support$_{Prob_pl}$ - 1) >= min_sup^{UP} => move to Temp_scanDB

AD = 2+2 = 4 => Prune

BD = 4+2 = 6 => move to Temp_scanDB

CD = 2+2 = 4 => Prune

รูปที่ 2.22 ตัวอย่างการอัพเดทฟรีเควนท์ k-ไอเทมเซตและk-ไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ในการเพิ่มขยายการค้นหากฎความสัมพันธ์โดยอาศัยหลักความน่าจะเป็น

รูปที่ 2.22 แสดงตัวอย่างการอัพเดทฟรีเควนท์ k-ไอเทมเซตและ k-ไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ เมื่อฐานข้อมูลเดิมมีจำนวนรายการ 10 รายการ, ความรู้เกี่ยวกับฟรีเควนท์ k-ไอเทมเซตและ k-ไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ของฐานข้อมูลเดิมจากรูปที่ 2.16 สมมติให้มีจำนวนรายการข้อมูลเพิ่มเข้ามาใหม่ 5 รายการ และกำหนดค่าสนับสนุนน้อยที่สุด = 0.4

ข้อดีของ Probagility-based algorithm

1) มีการเก็บผลลัพธ์จากการค้นหากฎความสัมพันธ์ในฐานข้อมูลเดิมมาใช้ร่วมกับฐานข้อมูลใหม่ที่เพิ่มเข้ามา

2) ไม่จำเป็นต้องสแกนฐานข้อมูลเดิมทุกรอบ k ในกรณีที่ต้องการอัพเดทค่าสนับสนุนของไอเทมเซตที่เป็นสมอลล์ในฐานข้อมูลเดิม เนื่องจากมีการเก็บ k-ไอเทมเซตที่มีโอกาสจะเป็นฟรีเควนท์ไว้ใน Temp_Scan จากนั้นจึงทำการสแกนฐานข้อมูลเดิมเพียงครั้งเดียว

ข้อจำกัดของ Probagility-based algorithm

1) ใช้ได้เฉพาะในกรณีที่มีการเพิ่มข้อมูลเท่านั้น

2) การทำนายไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ด้วยหลักการของเบอร์นูลลีต้องกำหนดจำนวนรายการข้อมูลที่จะเพิ่มเข้ามาไว้ล่วงหน้า (Fixed size)

2.5 การประมาณค่าความน่าจะเป็นด้วยหลักทางสถิติของเบอร์นูลลี

การทดลองเบอร์นูลลี (Bernoulli trials) เป็นการทดลองที่สนใจผลลัพธ์เพียง 2 เหตุการณ์ คือ เหตุการณ์ที่ประสบความสำเร็จ (Success) และเหตุการณ์ที่ไม่ประสบความสำเร็จ (Failure) โดยความน่าจะเป็นที่จะเกิดความสำเร็จในแต่ละครั้งมีค่าเท่ากับ p และความน่าจะเป็นที่จะเกิดความไม่สำเร็จมีค่า q=(1-p) เมื่อทำการทดลองซ้ำๆ จำนวน n ครั้ง ที่เป็นอิสระต่อกัน ค่าความน่าจะเป็นที่จะเกิดเหตุการณ์ประสบความสำเร็จ x ครั้ง สามารถคำนวณได้จาก

$$P(x) = \binom{n}{x} \cdot p^{x} \cdot (1-p)^{n-x} \qquad (2.10)$$

บทที่ 3
การค้นหากฎความสัมพันธ์แบบเพิ่มขยาย ในกรณีข้อมูลถูกเพิ่มและลบ

ข้อมูลที่ถูกจัดเก็บในฐานข้อมูลสามารถเกิดการเปลี่ยนแปลงได้ตลอดเวลา ซึ่งอาจเกิดจากการเพิ่มรายการเข้าไปในฐานข้อมูล (Insert) หรือลบรายการที่มีอยู่ในฐานข้อมูล (Delete) ในที่นี้จะเรียกส่วนของฐานข้อมูลก่อนทำการเปลี่ยนแปลงว่าฐานข้อมูลเดิม (Original database: DB) เรียกส่วนของข้อมูลใหม่ว่าฐานข้อมูลส่วนเพิ่ม (Incremental database: db) ซึ่งครอบคลุมได้ทั้งกรณีเพิ่มและลบข้อมูล และเรียกฐานข้อมูลที่ผ่าน การเปลี่ยนแปลงแล้วว่าฐานข้อมูลอัพเดท (Updated database: UD) เมื่อฐานข้อมูลมีการเปลี่ยนแปลงจะมีผลต่อกฎความสัมพันธ์ที่หาไว้แล้วในฐานข้อมูลเดิม เนื่องจากฟรีเควนท์ไอเทมเซตที่ได้ทำการค้นหาจากฐานข้อมูลเดิม อาจไม่เป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลอัพเดท ในขณะที่อินฟรีเควนท์ไอเทมเซตในฐานข้อมูลเดิมก็อาจกลายเป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลอัพเดทได้

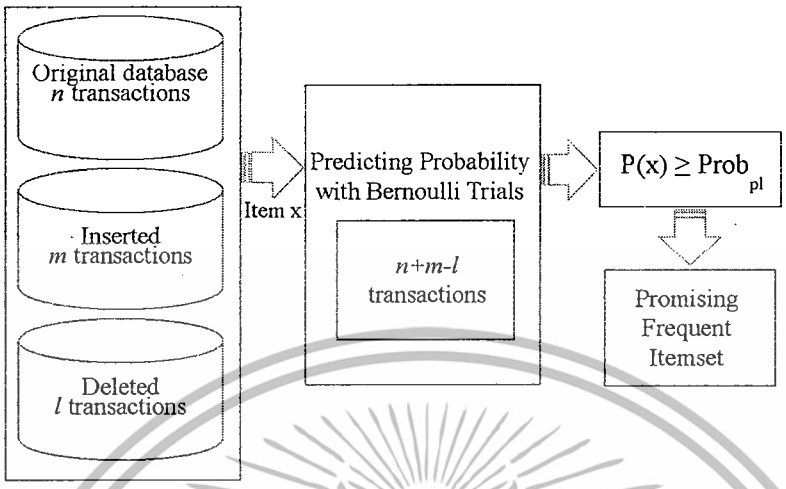
การค้นหากฎความสัมพันธ์แบบเพิ่มขยายเป็นทฤษฎีและแนวคิดที่นำเสนออัลกอริทึมในการหากฎความสัมพันธ์เมื่อมีการเปลี่ยนแปลงของรายการในฐานข้อมูล โดยนำความรู้ที่ได้จากการค้นหาความสัมพันธ์ของฐานข้อมูลเดิมมาใช้ เพื่อให้ได้อัลกอริทึมที่ทำงานได้อย่างมีประสิทธิภาพมากยิ่งขึ้น ทั้งนี้ ในกรณีที่ไอเทมเซตที่ต้องการอัพเดทค่าสนับสนุนเป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลเดิม ค่าสนับสนุนของไอเทมเซตจะถูกเก็บไว้เพื่อนำมาใช้ในอนาคต จึงสามารถอัพเดทค่าสนับสนุนเมื่อฐานข้อมูลมีการเปลี่ยนแปลงได้อย่างง่ายดาย ส่วนในกรณีที่ไอเทมเซตนั้นๆ ไม่เป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลเดิม จึงไม่มีการเก็บค่าสนับสนุนของไอเทมเซตเหล่านี้ไว้ ดังนั้น จึงต้องทำการสแกนฐานข้อมูลเดิมเพื่อหาค่าสนับสนุนที่เกิดขึ้นจริงก่อนจึงจะสามารถอัพเดทค่าสนับสนุนของไอเทมเซตได้ หลังจากนั้นถึงจะพิจารณาได้ว่าไอเทมเซตนั้นๆ จะเป็นฟรีเควนท์ไอเทมเซตหรือไม่

สำหรับงานวิจัยนี้ จะนำเสนอวิธีแก้ปัญหาการปรับปรุงกฎความสัมพันธ์เมื่อมีการเพิ่มข้อมูลชุดใหม่เข้ามาและมีข้อมูลเก่าถูกลบออกจากฐานข้อมูล แนวคิดหลักของอัลกอริทึมคือการเก็บทั้งฟรีเควนท์ไอเทมเซตและไอเทมเซตที่มีแนวโน้มจะเป็นฟรีเควนท์ไอเทมเซต (Promising frequent itemsets: PF) เมื่อมีการเปลี่ยนแปลงของรายการในฐานข้อมูล โดยการนำหลักทางสถิติของเบอร์นูลลี (Bernoulli trials) มาใช้ในการคาดคะเนไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ไอเทมเซต เพื่อลดจำนวนไอเทมเซตที่จะนำไปสแกนในฐานข้อมูลเดิม โดยในส่วนขั้นตอนการทำงานของอัลกอริทึมที่นำเสนอ รายละเอียดดังนี้

3.1 การคาดคะเนไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ไอเทมเซต

จากแนวคิดของกฎที่ว่าด้วยจำนวนมาก (Law of large number) ของเบอร์นูลลีที่กล่าวไว้ว่า “ค่าเฉลี่ยของตัวแปรสุ่มของตัวอย่างประชากรจำนวนมาก จะมีค่าเข้าใกล้ค่าเฉลี่ยของประชากรทั้งหมด” ดังนั้น งานวิจัยนี้จะนำหลักทางสถิติของเบอร์นูลลี (Bernoulli trials) มาใช้ในการคาดคะเนไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ไอเทมเซตโดยนำค่าสนับสนุนของไอเทมเซตในฐานข้อมูลเดิมมาคำนวณหาค่าความน่าจะเป็นที่ไอเทมเซตใดๆ จะมีโอกาสเป็นฟรีเควนท์ไอเทมเซตในอนาคต เมื่อมี

ข้อมูลใหม่ถูกเพิ่มเข้ามา m รายการ มีข้อมูลเก่าถูกลบออก l รายการ และฐานข้อมูลเดิมมีจำนวน n รายการ



รูปที่ 3.1 การคาดคะเนไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์

จากหลักการความน่าจะเป็นแบบเบอร์นูลลีที่ประกอบด้วยการทดลองจำนวน n ครั้งที่มีความเป็นอิสระต่อกัน และการเกิดเหตุการณ์ในการทดลองแต่ละครั้งจะประกอบด้วยผลสำเร็จของการทดลอง และผลความล้มเหลวของการทดลอง เมื่อนำหลักการของเบอร์นูลลีมาประยุกต์ใช้กับการค้นหากฎความสัมพันธ์ สามารถนำมาหลักการดังกล่าวมาพิจารณาการเกิดของไอเทมเซตในฐานข้อมูลได้คือ ให้การทดลอง n ครั้ง หมายถึง จำนวนรายการในฐานข้อมูลจำนวน n รายการ และผลการทดลองได้แก่ การเกิดหรือการปรากฏขึ้นของไอเทมเซตนั้นในแต่ละรายการของฐานข้อมูล ซึ่งถ้าไอเทมเซตที่พิจารณาปรากฏอยู่ในรายการนั้นๆ จะหมายถึงผลการทดลองที่สำเร็จ แต่ถ้าไอเทมเซตที่พิจารณาไม่ปรากฏอยู่ในรายการนั้นๆ จะหมายถึงผลการทดลองที่ล้มเหลว ค่าความน่าจะเป็นที่จะไอเทมใดๆ จะปรากฏขึ้นจำนวน x ครั้ง เมื่อมีข้อมูลใหม่ถูกเพิ่มเข้ามา m รายการ และมีข้อมูลเก่าถูกลบออก l รายการ สามารถหาได้จากสมการ (3.1)

$$P(x)_{itemset}=\binom{n+m-l}{x}\cdot p_{itemset}^{x}\cdot(1-p_{itemset})^{n+m-l-x} \tag{3.1}$$

ในที่นี้ $P(x)_{itemset}$ จะถูกประมาณค่าจากค่าสนับสนุนของไอเทมเซตในฐานข้อมูลเดิมภายใต้สมมติฐานว่าไอเทมเซตที่เกิดในฐานข้อมูลเดิมมีความแตกต่างกันน้อยมากกับการเกิดขึ้นของไอเทมเซตในส่วนของข้อมูลอัพเดท สมมติว่า k คือค่าสนับสนุนขั้นต่ำของฐานข้อมูลอัพเดท ไอเทมเซตใดๆ จะเป็นฟรีเควนท์ในฐานข้อมูลอัพเดทได้ก็ต่อเมื่อไอเทมนั้นมีค่าสนับสนุนอัพเดทมากกว่าหรือเท่ากับ k ซึ่งสามารถคำนวณหาค่าความน่าจะเป็นที่ไอเทมเซตใดๆ จะมีค่าสนับสนุนมากกว่าหรือเท่ากับ k ได้ดังสมการ (3.2)

$$P(x\geq k)_{itemset}=1-P(x<k)_{itemset} \tag{3.2}$$

โดย $P(x<k)_{itemset}$ จากสมการ 3.2 หาได้จาก

$$P(x<k)_{itemset}=\sum_{x=0}^{k-1}\binom{n+m-l}{x}\cdot p_{itemset}^{x}\cdot(1-p_{itemset})^{n+m-l-x} \qquad (3.3)$$

ทั้งนี้ ค่าความน่าจะเป็นของการเกิดไอเทมเซตที่พิจารณาในฐานข้อมูล หรือค่า p สามารถคำนวณได้ดังนี้

$$p=\frac{\sigma_x}{|DB|} \qquad (3.4)$$

เมื่อ σ_X แทนจำนวนรายการที่ไอเทมเซตนั้นๆ ปรากฏ และ $|DB|$ แทน จำนวนรายการทั้งหมดของฐานข้อมูลเดิม (Original database)

เมื่อคำนวณค่า $P(x\geq k)$ ของแต่ละไอเทมเซตได้แล้ว อินฟรีเควนท์ไอเทมเซตใดๆ ที่มีค่า $P(x\geq k)$ มากกว่าหรือเท่ากับ $Prob_{pl}$ จะถูกจัดเป็นไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ (Promising frequent itemset) โดย $Prob_{pl}$ คือค่าที่ผู้ใช้กำหนดขึ้นซึ่งเป็นตัวกำหนดความน่าจะเป็นขั้นต่ำที่อินฟรีเควนท์ไอเทมเซตใดๆ จะกลายเป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลอัพเดทได้ ถ้าตั้งค่า $Prob_{pl}$ ไว้สูงจำนวนไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ก็จะน้อยลง

3.2 อัลกอริทึมการค้นหากฎความสัมพันธ์แบบเพิ่มขยายในกรณีข้อมูลถูกเพิ่มและลบ

การค้นหากฎความสัมพันธ์แบบเพิ่มขยายในงานวิจัยฉบับนี้ จะใช้หลักการหากฎความสัมพันธ์โดยใช้อัลกอริทึมอะพริโอริเป็นฐาน ซึ่งเป็นการค้นหาข้อมูลตามลำดับจำนวนสมาชิกของไอเทมเซตจากน้อยไปหามาก (k = 1, 2, 3,... n) โดยนำเอาไอเทมเซตที่เป็นฟรีเควนท์ k-1 ไอเทมเซตมาใช้ในการสร้าง แคนดิเดต k ไอเทมเซต ด้วยขั้นตอนการเชื่อม (join) และการตัด (prune) ไอเทมที่ไม่มีโอกาสเป็นฟรีเควนท์ไอเทมเซตออกไป นอกจากนี้ยังได้นำหลักการความน่าจะเป็นที่ใช้ในงานวิจัยการเพิ่มขยายการค้นหากฎความสัมพันธ์โดยใช้หลักความน่าจะเป็นเข้ามาประยุกต์ร่วมด้วย ในการหาความน่าจะเป็นของไอเท็มเซตที่คาดว่าจะเป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลปรับปรุง เพื่อจะนำไปใช้ในการค้นหาฟรีเควนท์ไอเทมเซตในรอบถัดไป และลดจำนวนไอเทมเซตที่จะถูกนำไปสแกนในฐานข้อมูลเดิม เพื่อช่วยลดระยะเวลาในการประมวลผล ตารางที่ 3.1 แสดงสัญลักษณ์ที่ใช้ในอัลกอริทึมที่นำเสนอในงานวิจัยนี้

อัลกอริทึมการค้นหากฎความสัมพันธ์แบบเพิ่มขยายในกรณีที่ข้อมูลถูกเพิ่มและลบ จะแบ่งการทำงานออกเป็น 2 ขั้นตอนหลัก ได้แก่ 1) การหาฟรีเควนท์ไอเทมเซตและไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ในฐานข้อมูลเดิม 2) การอัพเดทฟรีเควนท์ไอเทมเซตและไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ในฐานข้อมูลอัพเดท โดยมีรายละเอียดการทำงานของแต่ละขั้นตอน ดังนี้

3.2.1 การค้นหาฟรีเควนท์ไอเทมเซตและไอเทมที่คาดว่าจะเป็นฟรีเควนท์ในฐานข้อมูลเดิม

การพิจารณาว่าไอเทมเซตใดจะเป็นฟรีเควนท์ไอเทมเซตได้นั้นจะดูได้จากค่าสนับสนุนของไอเทมเซตนั้นๆ ที่ปรากฏในฐานข้อมูลว่ามีค่ามากกว่าหรือเท่ากับค่าสนับสนุนขั้นต่ำที่ผู้ใช้กำหนด ส่วนไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์นั้นหาได้อินฟรีเควนท์ไอเทมเซตที่มีค่า $P(x\geq k)$ มากกว่าหรือเท่ากับ $Prob_{pl}$ เมื่อ k คือค่าสนับสนุนขั้นต่ำของฐานข้อมูลอัพเดท และ $Prob_{pl}$ เป็นค่าความน่าจะ

เป็นขั้นต่ำที่ผู้ใช้กำหนดไว้ ตัวอย่างเช่น สมมติว่าฐานข้อมูลเดิมมีจำนวน 6 รายการ ดังตาราง 3.2 มีรายการข้อมูลใหม่เพิ่มเข้ามา 3 รายการ และมีข้อมูลเก่าถูกลบออก 1 รายการ ดังตารางที่ 3.3 และ 3.4 ตามลำดับ ถ้ากำหนดค่าสนับสนุนขั้นต่ำที่ 40% และ $Prob_{pl}$ = 0.1 ไอเทมเซตที่มีค่าสนับสนุนมากกว่าหรือเท่ากับ 3 (6*40%=2.4) จะเป็นฟรีเควนท์ไอเทมเซต เมื่อมีรายการใหม่เพิ่มเข้ามา 3 รายการ และรายการเก่าถูกลบออก 1 รายการ ฐานข้อมูลอัพเดทจะมีจำนวน 8 รายการ โดยมีค่าสนับสนุนขั้นต่ำ k=4 (8*40%=3.2) ซึ่งเมื่อนำค่านี้ไปคำนวณตามสมการที่ 3.2 จะได้ผลลัพธ์ดังตารางที่ 3.6

ตารางที่ 3.1 สัญลักษณ์ที่ใช้ในอัลกอริทึมการค้นหากฎความสัมพันธ์แบบเพิ่มขยาย

DB	ฐานข้อมูลเดิม
D'	ฐานข้อมูลอัพเดท
σ_X	ค่าสนับสนุนของไอเทม x ในฐานข้อมูลเดิม
σ'_X	ค่าสนับสนุนของไอเทม x ในฐานข้อมูลอัพเดท
Δ^+	รายการที่ถูกเพิ่มเข้ามา
Δ^-	รายการที่ถูกลบออกมา
δ_x^+	ค่าสนับสนุนของไอเทม x ใน Δ^+
δ_x^-	ค่าสนับสนุนของไอเทม x ใน Δ^-
s	ค่าสนับสนุนขั้นต่ำ
ρ	ค่าสนับสนุนขั้นต่ำของไอเทมที่คาดว่าจะเป็นฟรีเควนท์ใน D
ρ'	ค่าสนับสนุนขั้นต่ำของไอเทมที่คาดว่าจะเป็นฟรีเควนท์ใน D'
C_k	แคนดิเดต k-ไอเทมเซต
F_k	ฟรีเควนท์ k-ไอเทมเซตเดิม
F'_k	ฟรีเควนท์ k-ไอเทมเซตอัพเดท
PF_k	k-ไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์เดิม
PF'_k	k-ไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์อัพเดท
$Prob_{PI}$	ค่าความน่าจะเป็นที่น้อยที่สุดที่คาดว่าไอเทมเซตจะกลายเป็นฟรีเควนท์

ตารางที่ 3.2 ตัวอย่างรายการในฐานข้อมูลเดิม

TID	Items
100	ABC
200	ADE
300	DE
400	AE
500	ABDE
600	AE

ตารางที่ 3.3 ตัวอย่างรายการที่ถูกเพิ่มเข้ามา

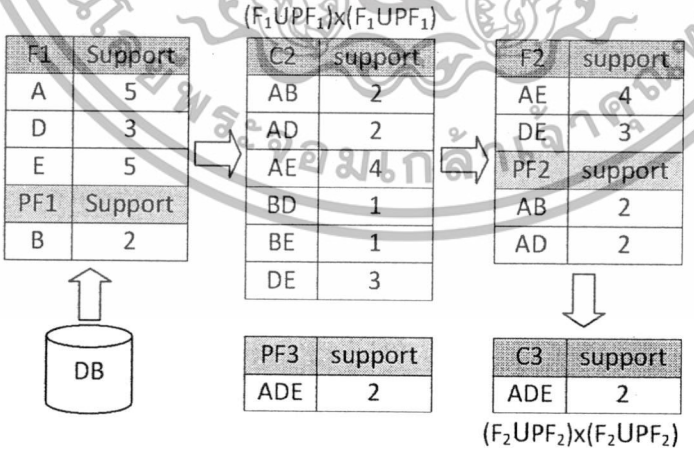
TID	Items
700	BCE
800	BCE
900	ABCE

ตารางที่ 3.4 ตัวอย่างรายการที่ถูกลบออก

TID	Items
100	ABC

ตารางที่ 3.5 แคนดิเดต 1-ไอเทมเซตและ $P(x \geq 4)$

Item	Count	$P(x \geq 4)$
A	5	$1 - \sum_{x=0}^{3} \binom{8}{x} \frac{5}{6}^{x} \frac{1}{6}^{8-x} = 0.9954$
B	2	$1 - \sum_{x=0}^{3} \binom{8}{x} \frac{2}{6}^{x} \frac{4}{6}^{8-x} = 0.2589$
C	1	$1 - \sum_{x=0}^{3} \binom{8}{x} \frac{1}{6}^{x} \frac{5}{6}^{8-x} = 0.0307$
D	3	$1 - \sum_{x=0}^{3} \binom{8}{x} \frac{3}{6}^{x} \frac{3}{6}^{8-x} = 0.6367$
E	5	$1 - \sum_{x=0}^{3} \binom{8}{x} \frac{5}{6}^{x} \frac{1}{6}^{8-x} = 0.9954$



รูปที่ 3.2 ตัวอย่างการหาฟรีเควนท์ไอเทมเซตและไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ในฐานข้อมูลเดิม

จากตารางที่ 3.5 จะได้ {A, D, E} เป็นฟรีเควนท์ 1-ไอเทมเซต และ {B} เป็นไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ เนื่องจากค่าความน่าจะเป็นที่คำนวณได้มีค่ามากกว่า $Prob_{pl}$ จากนั้นใช้หลักการเชื่อม (Join) เช่นเดียวกับอัลกอริทึมอะพริโอริ โดยนำฟรีเควนท์ k-1 ไอเทมเซตและ k-1 ไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์มาใช้ในการสร้างแคนดิเดต k-ไอเทมเซต ซึ่งจะได้ผลลัพธ์ดังรูป 3.2

3.2.2 การอัพเดทฟรีเควนท์ไอเทมเซตและไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ในฐานข้อมูลอัพเดท

อัลกอริทึมในการอัพเดทฟรีเควนท์ไอเทมเซตและไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ประกอบด้วยกระบวนการย่อย 3 กระบวนการ ได้แก่ 1) การอัพเดท 1-ไอเทมเซต 2) การอัพเดท k-ไอเทมเซต เมื่อ $k \geq 2$ และ 3) การสแกนฐานข้อมูลเดิม ดังแสดงขั้นตอนในรูป 3.3

Algorithm 1

Input: DB, Δ^+, Δ^-, C_1^{DB}, F_k^{DB}, PF_k^{DB}, s

Output: C_1', F_k', PF_k'

Subprocedure I: Updating 1-itemsets // k=1

1. scan $\Delta^+$and $\Delta^-$to find out $\delta_x^+$and δ_x^-
2. for each itemset $X \in C_1^{DB} \cup C_1^{\Delta+} \cup C_1^{\Delta-}$
3. $\sigma_x' = \sigma_x + \delta_x^+ - \delta_x^-$
4. $C_1' = C_1' \cup \{X\}$
5. end
6. $F_1' = \{X \in C_1' \mid \sigma_x' \geq |D'| \times s$
7. $PF_1' = \{X \in C_1' \mid \rho' \leq \sigma_x' < |D'| \times s$

Subprocedure II: Updating k-itemsets // k≥2

8. Temp_scanDB = ∅
9. while $F_{k-1}' \cup PF_{k-1}' \neq \emptyset$
10. $C_k = (F_{k-1}' \cup PF_{k-1}') \times (F_{k-1}' \cup PF_{k-1}')$
11. scan Δ^+ to find out δ_x^+ for each X in C_k
12. for each itemset $X \in C_k$
13. if $X \notin F_k^{DB} \cup PF_k^{DB}$
14. move X to C_k^{new}
15. for each itemset $X \in C_k$ and $X \in F_k^{DB} \cup PF_k^{DB}$
16. $\sigma_x' = \sigma_x + \delta_x^+$
17. if $\sigma_x' < |D'| \times s$
18. remove X from C_k
19. for each itemset $X \in C_k^{new}$
20. if $\delta_x^+ + (\rho - 1) < |D'| \times s$
21. remove X from C_k^{new}
22. scan Δ^- to find out $\delta_x^-$for each X in $C_k \cup C_k^{new}$
23. for each itemset $X \in C_k$ and $X \in F_k^{DB} \cup PF_k^{DB}$
24. $\sigma_x' = \sigma_x' - \delta_x^-$

25. for each itemset $X \in C_k^{new}$
26. if $\delta_X^+ - \delta_X^- + (\rho - 1) \geq |D'| \times s$
27. move X to Temp_scanDB
28. $F_k' = \{X \in C_k \mid \sigma_X' \geq |D'| \times s$
29. $PF_k' = \{X \in C_k \mid \rho' \leq \sigma_X' < |D'| \times s$
30.end
31. for each itemset $X \in C_k^{new}$
32. if $\delta_X^+ - \delta_X^- + (\rho - 1) \geq |D'| \times s$
33. move X to Temp_scanDB
34. $F_k' = \{X \in C_k \mid \sigma_X' \geq |D'| \times s$
35. $PF_k' = \{X \in C_k \mid \rho' \leq \sigma_X' < |D'| \times s$
36.end

Subprocedure III: Scanning an original database

37.scan DB to find out σ_x for each X in Temp_scanDB
38.for each itemset $X \in Temp\_scanDB_k$
39. $\sigma_X' = \sigma_x + \delta_X^+ - \delta_X^-$
40. $F_k^{new} = \{X \in Temp\_scanDB_k$ and $\sigma_X' \geq |D'| \times s$
41. $PF_k^{new} = \{X \in Temp\_scanDB_k$ and $\rho' \leq \sigma_X' < |D'| \times s$
42. $F_k' = F_k' \cup F_k^{new}$
43. $PF_k' = PF_k' \cup PF_k^{new}$

รูปที่ 3.3 กระบวนการอัพเดทฟรีเควนท์ไอเทมเซตและไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์

ในกระบวนการย่อยที่ 1 นั้น รายการข้อมูลใหม่ที่ถูกเพิ่มเข้ามาและรายการข้อมูลเก่าที่ถูกลบออกจากฐานข้อมูลเดิมจะถูกสแกน เพื่อหาค่าสนับสนุนของแต่ละ 1-ไอเทมเซต ที่เกิดขึ้นใน Δ^+ และ Δ^- ค่าสนับสนุนที่ได้จะถูกนำไปคำนวณร่วมกับค่าสนับสนุนของ 1-ไอเทมเซตจากฐานข้อมูลเดิมที่ถูกเก็บไว้จากกระบวนการก่อนหน้านี้ ทำให้สามารถอัพเดทค่าสนับสนุนของ 1-ไอเทมเซตได้อย่างง่ายดาย จากนั้นก็สามารถหา 1-ฟรีเควนท์ไอเทมเซตและ 1-ไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ในฐานข้อมูลอัพเดทได้

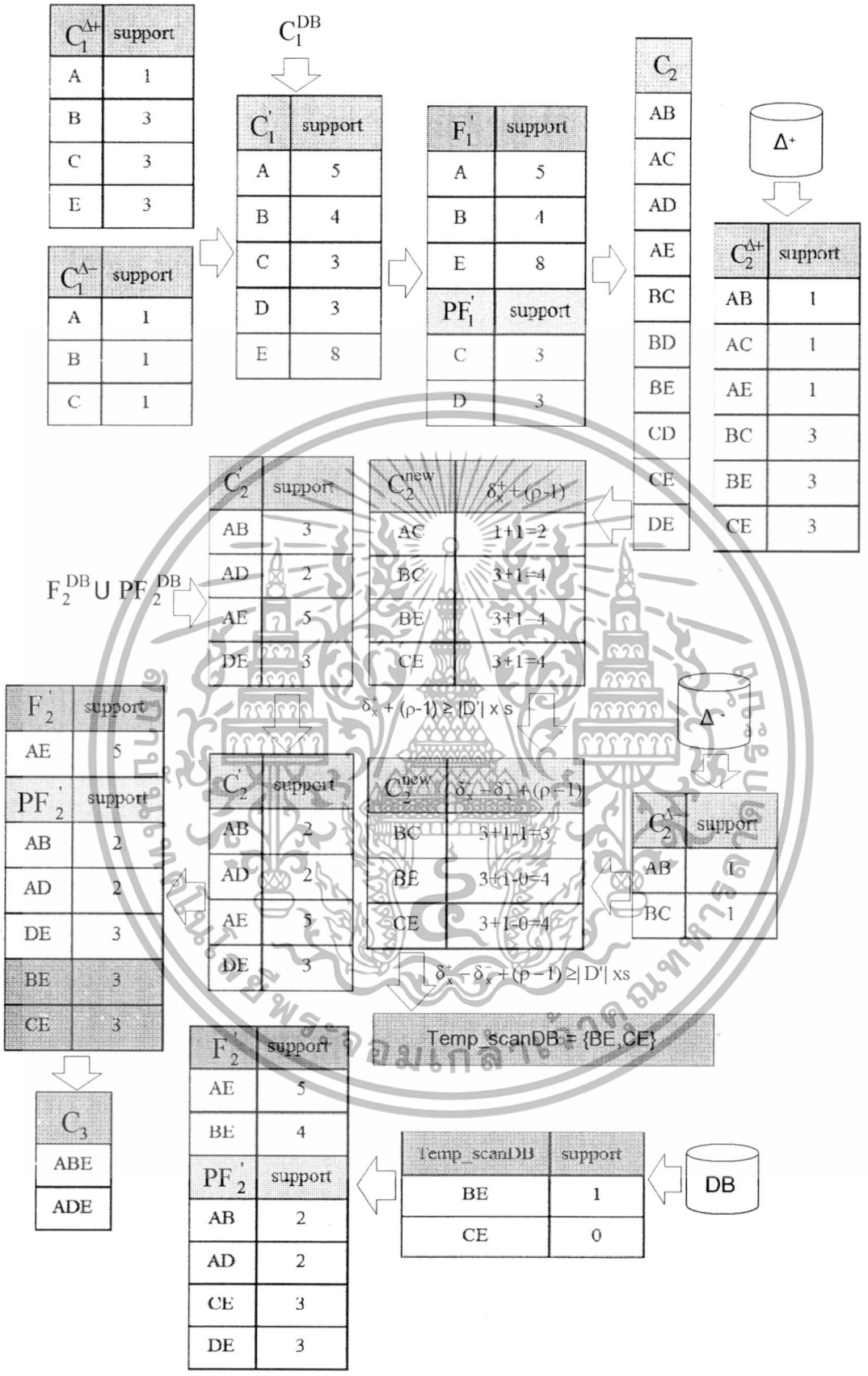
ในกระบวนการย่อยที่ 2 ฟรีเควนท์ k-ไอเทมเซตและ k-ไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์จะถูกอัพเดทสำหรับ $k \geq 2$ โดยจะทำกระบวนการย่อยนี้ซ้ำไปเรื่อยๆ จนกว่าจะไม่สามารถสร้างแคนดิเดต k-ไอเทมเซตได้อีกต่อไป ขั้นตอนนี้เริ่มจากการนำ (k-1)-ฟรีเควนท์ไอเทมเซตและ (k-1)-ไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ที่อัพเดทแล้วมาเชื่อม (Join) ตามหลักการของ Apriori เพื่อสร้างแคนดิเดต k-ไอเทมเซต (C_k) จากนั้นจึงดำเนินการสแกน Δ^+ เพื่อหาค่าสนับสนุนของไอเทมเซตที่เกิดขึ้นใน Δ^+ สำหรับแคนดิเดตไอเทมเซตที่เป็นฟรีเควนท์ไอเทมเซตหรือเป็นไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ในฐานข้อมูลเดิมนั้น เราสามารถอัพเดทค่าสนับสนุนได้โดยการนำค่าสนับสนุนจากฐานข้อมูลเดิมมาคำนวณ ดังแสดงในบรรทัดที่ 16 ซึ่งไอเทมเซตที่ค่าสนับสนุนอัพเดทมีค่าน้อยกว่า $|D'| \times s$ จะถูกตัดทิ้ง (Prune) จาก C_k เนื่องจากไม่มีโอกาสเป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลอัพเดท ส่วนแคนดิเดตไอเทมเซตที่ไม่ได้เป็นฟรีเควนท์ไอเทมเซตหรือเป็นไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ใน

ฐานข้อมูลเดิมจะถูกย้ายไปเก็บไว้ใน C_k^{new} เนื่องจากเราไม่รู้ค่าสนับสนุนในฐานข้อมูลเดิมของไอเทมเซตเหล่านี้จึงไม่สามารถอัพเดทค่าสนับสนุนของไอเทมเซตในฐานข้อมูลอัพเดทได้ งานวิจัยนี้จะทำการประมาณค่าสนับสนุนของ C_k^{new} โดยใช้หลักการของค่าที่เป็นไปได้สูงสุด (principle of maximum possible value) เนื่องจากแคนดิเดตไอเทมเซตเหล่านี้ไม่ได้อยู่ใน $F_k^{DB} \cup PF_k^{DB}$ แสดงว่าค่าสนับสนุนมากสุดที่เป็นไปได้ในฐานข้อมูลเดิมของไอเทมเซตเหล่านี้คือ ρ-1 ดังแสดงในบรรทัดที่ 20 ไอเทมเซตที่มีค่าประมาณของค่าสนับสนุนน้อยกว่า $|D'| \times s$ จะถูกตัดทิ้ง เนื่องจากไม่มีโอกาสเป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลอัพเดท

ในขั้นตอนต่อไป Δ^- จะถูกสแกนเพื่อหาค่าสนับสนุนของไอเทมเซตที่เหลืออยู่ใน $C_k \cup C_k^{new}$ ด้วยหลักการเช่นเดียวกันกับที่กล่าวข้างต้น แคนดิเดตไอเทมเซตที่เป็นฟรีเควนท์ไอเทมเซตหรือเป็นไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ในฐานข้อมูลเดิมนั้น เราสามารถอัพเดทค่าสนับสนุนได้โดยการนำค่าสนับสนุนจากฐานข้อมูลเดิมมาคำนวณ ดังแสดงในบรรทัดที่ 24 ไอเทมเซตที่มีค่าสนับสนุนผ่านเกณฑ์ขั้นต่ำจะเป็นฟรีเควนท์ไอเทมเซตและไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ที่ผ่านการอัพเดทแล้ว ดังแสดงในบรรทัดที่ 28-29 ส่วนแคนดิเดตไอเทมเซตที่เหลืออยู่ซึ่งไม่ได้เป็นฟรีเควนท์ไอเทมเซตหรือเป็นไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ในฐานข้อมูลเดิม จะถูกประมาณค่าสนับสนุนโดยใช้หลักการของค่าที่เป็นไปได้สูงสุด ดังแสดงในบรรทัดที่ 26 แคนดิเดตไอเทมเซตที่มีค่าประมาณของค่าสนับสนุนมากกว่าหรือเท่ากับ $|D'| \times s$ จะถูกย้ายไปเก็บไว้ใน Temp_scanDB เนื่องจากไอเทมเซตเหล่านี้มีโอกาสที่จะเป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลอัพเดทได้ ดังนั้นจึงถูกเก็บไว้เพื่อนำไปสแกนหาค่าสนับสนุนที่แท้จริงในฐานข้อมูลเดิมในกระบวนการย่อยที่ 3 ซึ่งขั้นตอนทั้งหมดในกระบวนการย่อยที่ 2 นี้ จะถูกทำซ้ำไปเรื่อยๆ จนกว่าจะไม่สามารถสร้างแคนดิเดต k-ไอเทมเซตได้อีกต่อไป

ในกระบวนการย่อยที่ 3 แคนดิเดตไอเทมเซตทั้งหมดที่เก็บไว้ใน Temp_scanDB จะถูกนำไปสแกนหาค่าสนับสนุนที่แท้จริงฐานข้อมูลเดิม ซึ่งเมื่อหาค่าสนับสนุนในฐานของมูลเดิมได้แล้วก็จะสามารถอัพเดทค่าสนับสนุนได้ดังแสดงในบรรทัดที่ 39 จากนั้นจึงทำการอัพเดท k-ฟรีเควนท์ไอเทมเซตและ k-ไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ ดังแสดงในบรรทัดที่ 40-43

รูปที่ 3.4 แสดงตัวอย่างการอัพเดท k-ฟรีเควนท์ไอเทมเซตและ k-ไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ เมื่อฐานข้อมูลเดิมมีจำนวน 6 รายการ ดังตาราง 3.3 มีรายการข้อมูลใหม่เพิ่มเข้ามา 3 รายการ และมีข้อมูลเก่าถูกลบออก 1 รายการ ดังตารางที่ 3.4 และ 3.5 ตามลำดับ



รูปที่ 3.4 ตัวอย่างการอัพเดท k-ฟรีเควนท์ไอเทมเซตและ k-ไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์

บทที่ 4
การทดลองและวิเคราะห์ผลการทดลอง

รายการที่จัดเก็บอยู่ในฐานข้อมูลนั้นมักมีการปรับปรุงเพื่อให้ทันสมัยอยู่ตลอดเวลา (Dynamic database) ซึ่งการปรับปรุงนี้มีทั้งการเพิ่ม ลบ และแก้ไขรายการข้อมูล ส่งผลให้เกิดการเปลี่ยนแปลงกฎความสัมพันธ์ที่ได้ค้นหาไว้แล้ว โดยอาจทำให้กฎที่มีอยู่ไม่มีความถูกต้องเนื่องจากไอเทมเซตที่พบว่าเป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลเดิมอาจไม่เป็นฟรีเควนท์ไอเทมเซตอีกต่อไปเมื่อมีข้อมูลใหม่ถูกเพิ่มเข้ามาหรือมีข้อมูลเก่าถูกลบออก

งานวิจัยนี้ได้นำเสนอวิธีการในการปรับปรุงฟรีเควนท์ไอเทมเซตโดยนำค่าความน่าจะเป็นของไอเทมเซตที่เกิดในฐานข้อมูลเดิมมาใช้ในการทำนายไอเทมที่คาดว่าจะเป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลปรับปรุง ซึ่งนอกจากจะสามารถช่วยให้ค้นหาฟรีเควนท์ไอเทมเซตได้อย่างถูกต้องและมีประสิทธิภาพแล้ว ยังสามารถลดจำนวนไอเทมเซตที่ต้องนำไปสแกนในฐานข้อมูลเดิมได้อีกด้วย

ในบทนี้จะกล่าวถึงวัตถุประสงค์ของการทดลอง วิธีการทดลอง และผลการทดลอง ของอัลกอริทึมการค้นหากฎความสัมพันธ์แบบเพิ่มขยายโดยใช้หลักความน่าจะเป็นเมื่อข้อมูลถูกเพิ่มและลบ

4.1 วัตถุประสงค์ของการทดลอง

เพื่อแสดงให้เห็นถึงประสิทธิภาพการทำงานของอัลกอริทึมการค้นหากฎความสัมพันธ์เพิ่มโดยใช้หลักความน่าจะเป็นเมื่อข้อมูลถูกเพิ่มและลบ โดยการเปรียบเทียบประสิทธิภาพการทำงานกับอัลกอริทึมที่ใช้ในการค้นหากฎความสัมพันธ์แบบเพิ่มขยาย ได้แก่ Apriori, FUP2 และ Pre-Large ในหัวข้อนี้จะกล่าวถึงวัตถุประสงค์ของการทดลอง ซึ่งประกอบด้วย 2 วัตถุประสงค์หลัก ดังนี้

1. เพื่อทดสอบความถูกต้องของผลลัพธ์ที่ได้จากการเพิ่มข้อมูลใหม่และลบข้อมูลเก่าจากฐานข้อมูลเดิม เนื่องจากอัลกอริทึมการค้นหากฎความสัมพันธ์เพิ่มขยายทั้งหมดที่ใช้ในการทดลอง ได้แก่ FUP2, Pre-Large และอัลกอริทึมที่นำเสนอในงานวิจัยนี้ ล้วนมีพื้นฐานการทำงานมาจากอัลกอริทึม Apriori ดังนั้น ในการทดสอบความถูกต้องของผลลัพธ์จะทำการเปรียบเทียบผลที่ได้จากแต่ละอัลกอริทึมกับผลที่ได้จากอัลกอริทึม Apriori เป็นหลัก

2. เพื่อทดสอบประสิทธิภาพในการทำงานของอัลกอริทึมในกรณีที่ข้อมูลถูกเพิ่มและลบออกจากฐานข้อมูลเดิม การทดสอบประสิทธิภาพของอัลกอริทึมในงานวิจัยนี้ จะเป็นการทดสอบเพื่อวัดประสิทธิภาพการค้นหากฎความสัมพันธ์แบบเพิ่มขยาย โดยเปรียบเทียบจากเวลาที่ใช้ในการประมวลผล (Execution Time) ของแต่ละอัลกอริทึม ในกรณีที่ 1) มีการเพิ่มและลบข้อมูลด้วยค่าสนับสนุนขั้นต่ำ (Minimum support) ที่แตกต่างกัน และ 2) มีการเพิ่มและลบข้อมูลด้วยขนาดที่แตกต่างกัน

4.2 วิธีการทดลอง

ในการทดลองสำหรับงานวิจัยนี้ จะเป็นการทดลองโดยการ 1) เพิ่มข้อมูลใหม่จำนวน 20,000 รายการ และลบข้อมูลเก่าออก 10,000 รายการ โดยกำหนดค่าสนับสนุนขั้นต่ำที่แตกต่างกัน

ในช่วงระหว่าง 3%-7% และ 2) เพิ่มข้อมูลใหม่และลบข้อมูลเก่าออกจากฐานข้อมูลเดิมด้วยขนาดที่แตกต่างกัน โดยการกำหนดค่าสนับสนุนขั้นต่ำที่ 5%

สำหรับชุดข้อมูลที่นำมาทดลอง เป็นชุดข้อมูลสังเคราะห์ (Synthesis Dataset) ซึ่งเป็นชุดข้อมูลที่นำเสนอโดย Agrawal และคณะ [1] ซึ่งได้เสนอวิธีการสร้างชุดข้อมูลสังเคราะห์เพื่อใช้ในการประเมินประสิทธิภาพของอัลกอริทึม โดยอาศัยหลักการทางสถิติมาใช้ในการสร้างชุดข้อมูล สำหรับการทดลองในงานวิจัยนี้ ชุดข้อมูลที่ใช้คือชุดข้อมูล I4T10D100K ซึ่งมีรายละเอียดดังนี้

- ค่าเฉลี่ยความยาวสูงสุดของฟรีเควนท์ไอเทมเซต (I) คือ 4
- ค่าเฉลี่ยของไอเทมที่เกิดขึ้นในแต่ละรายการ(T) คือ 10
- จำนวนรายการในฐานข้อมูลเดิม (D) คือ 100,000

4.3 ผลการทดลอง

เมื่อนำข้อมูลทั้ง 2 ชุดดังกล่าวข้างต้นมาทำการทดลองด้วยโปรแกรม MATLAB 7.6 ผลการทดลองเป็นดังนี้

4.3.1 **ผลการทดลองการเพิ่มข้อมูลใหม่จำนวน 20,000 รายการ และลบข้อมูลเก่าออก 10,000 รายการ โดยกำหนดค่าสนับสนุนขั้นต่ำที่แตกต่างกันในช่วงระหว่าง 3%-7%** ในที่นี้ได้ทำการทดสอบประสิทธิภาพของอัลกอริทึมโดยเปรียบเทียบกับอัลกอริทึม Apriori, FUP2 และ Pre-Large ผลการทดลองแสดงตามตารางที่ 4.1

ตารางที่ 4.1 ผลการเปรียบเทียบเวลาที่ใช้ในการประมวลผลเมื่อมีการเพิ่มข้อมูล 20,000 รายการ และลบข้อมูล 10,000 รายการ

อัลกอริทึม	ค่าสนับสนุนขั้นต่ำ (%)				
	3	4	5	6	7
Aprioiri	448.8226	352.4753	276.6451	177.0909	134.2924
FUP2	143.6619	97.6027	78.7477	59.6372	41.7659
Pre-large	284.1642	218.3193	165.5858	107.9419	70.5397
Proposed-algorithm	123.9547	88.3476	57.9848	37.5175	29.4387

จากตารางที่ 4.1 จะเห็นได้ว่าอัลกอริทึมที่นำเสนอในงานวิจัยนี้ใช้เวลาในการประมวลผลน้อยกว่าอัลกอริทึม Apriori, FUP2 และ Pre-Large ทั้งนี้เนื่องจากการเก็บค่าไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์นั้น สามารถช่วยลดเวลาที่ใช้ในการปรับปรุงค่าสนับสนุนของไอเทมเซตที่ไม่ได้เป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลเดิมบางตัวได้โดยไม่จำเป็นต้องสแกนฐานข้อมูลเดิมทุกครั้ง ในขณะที่อัลกอริทึม Apriori และ FUP2 นั้น จำเป็นต้องสแกนฐานข้อมูลเดิมทั้งหมดทุกรอบ k เพื่อปรับปรุงค่าสนับสนุนของไอเทมเซตที่ไม่ได้เป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลเดิม เมื่อพิจารณาถึงความถูกต้องของความถูกต้องในการประมวลผลโดยเปรียบเทียบกับอัลกอริทึม Apriori ผลการทดสอบพบว่าอัลกอริทึมที่นำเสนอในงานวิจัยนี้ให้ผลลัพธ์ที่ถูกต้อง คือมีจำนวนฟรีเควนท์ไอเทมเซตเท่ากับจำนวนฟรีเควนท์ไอเทมเซตที่หาได้จากอัลกอริทึม Apriori ดังแสดงในตารางที่ 4.2

ตารางที่ 4.2 จำนวนฟรีเควนท์ไอเทมเซตและไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ในฐานข้อมูลเดิม

k-itemset (k)	Apriori / FUP2 algorithms		Pre-large algorithm		Proposed-algorithm	
	F	PF	F	PF	F	PF
1	144	0	144	13	144	4
2	243	0	243	30	243	8
3	305	0	305	28	305	14
4	266	0	266	21	266	21
5	173	0	173	7	173	0
6	73	0	73	0	73	0
7	18	0	18	0	18	0
8	2	0	2	0	2	0
Total	1,224	0	1,224	99	1,224	47

ตารางที่ 4.2 แสดงจำนวนฟรีเควนท์ไอเทมเซตและไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ในฐานข้อมูลเดิม เมื่อกำหนดค่าสนับสนุนขั้นต่ำที่ 3% โดยคอลัมน์ F คือจำนวนฟรีเควนท์ไอเทมเซต และคอลัมน์ PF คือจำนวนไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ ทั้งนี้จะเห็นได้ว่าทุกอัลกอริทึมที่นำมาเปรียบเทียบในการทดลองนี้มีฟรีเควนท์ไอเทมเซตเท่ากับฟรีเควนท์ไอเทมเซตที่หาได้จากอัลกอริทึม Apriori โดยอัลกอริทึม Pre-Large และอัลกอริทึมที่นำเสนอในงานวิจัยนี้จะมีการเก็บไอเทมเซตบางตัวที่ไม่ได้เป็นฟรีเควนท์ไอเทมเซตเอาไว้ด้วย ในขณะที่อัลกอริทึม Apriori และ FUP2 จะเก็บเฉพาะฟรีเควนท์ไอเทมเซตเท่านั้น

สาเหตุที่สำคัญอีกประการหนึ่งที่ทำให้อัลกอริทึมที่นำเสนอในงานวิจัยนี้ใช้เวลาในการประมวลผลน้อยกว่าอัลกอริทึม Apriori, FUP2 และ Pre-Large ก็คืออัลกอริทึมที่นำเสนอในงานวิจัยนี้สามารถตัดไอเทมเซตที่ไม่มีโอกาสเป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลปรับปรุงได้ โดยใช้แนวคิดของค่าสนับสนุนที่เป็นไปได้สูงสุด (Principle of maximum possible number) ทำให้ลดจำนวนไอเทมเซตที่จำเป็นต้องนำไปสแกนในฐานข้อมูลเดิมได้อย่างมีประสิทธิภาพ ดังแสดงในตารางที่ 4.3

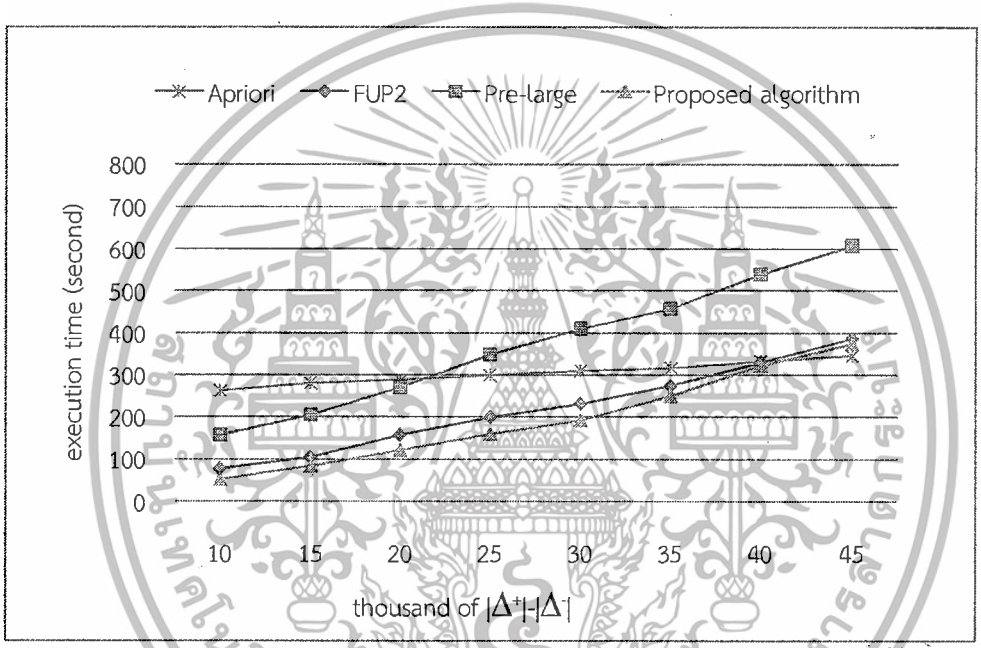
ตารางที่ 4.3 จำนวนไอเทมเซตที่ถูกสแกนในฐานข้อมูลเดิม

k-itemset (k)	Apriori	FUP2	Pre-large	Proposed-algorithm
1	235	1	107	0
2	6441	25	2718	2
3	445	17	177	2
4	282	1	15	0
5	173	0	0	0
6	73	0	1	0
7	18	0	0	0
8	2	0	0	0
Total	7,669	44	3,018	4

จากตารางที่ 4.3 จะเห็นว่าจำนวนไอเทมเซตที่ถูกสแกนในฐานข้อมูลเดิมในอัลกอริทึมที่นำสเนอในงานวิจัยนี้ มีจำนวนน้อยมากเมื่อเปรียบเทียบกับอัลกอริทึม Apriori, FUP2 และ Pre-Large

ทำให้ใช้เวลาในการประมวลผลน้อยกว่า นอกจากนั้นแล้ว อัลกอริทึมที่นำเสนอในงานวิจัยนี้จะสแกนฐานข้อมูลเดิมเพียงครั้งเดียวสำหรับ k-ไอเทมเซต ทั้งหมด ในขณะที่อัลกอริทึม Apriori, FUP2 และ Pre-Large จะสแกนฐานข้อมูลเดิมตามจำนวน k ที่เกิดขึ้น

4.3.2 ผลการทดลองการเพิ่มข้อมูลใหม่และลบข้อมูลเก่าออกจากฐานข้อมูลเดิมด้วยขนาดที่แตกต่างกัน ในที่นี้ ได้ทำการทดสอบประสิทธิภาพของอัลกอริทึมโดยเปรียบเทียบกับอัลกอริทึม Apriori, FUP2 และ Pre-Large โดยการกำหนดค่าสนับสนุนขั้นต่ำเท่ากับ 5% โดยแต่ละรอบจะกำหนดจำนวนรายการของข้อมูลที่เพิ่มเข้ามาให้มีขนาดเป็น 2 เท่า ของจำนวนรายการที่ถูกลบออกจากฐานข้อมูลเดิม เพื่อให้เกิดความแตกต่างระหว่างขนาดของฐานข้อมูลเดิมและขนาดของฐานข้อมูลปรับปรุง ผลการทดลองแสดงตามรูปที่ 4.1



รูปที่ 4.1 เวลาที่ใช้ในการประมวลผลเมื่อเพิ่มและลบข้อมูลด้วยขนาดที่แตกต่างกัน

จากรูปที่ 4.1 อัลกอริทึมที่นำเสนอในงานวิจัยนี้ใช้เวลาในการประมวลผลน้อยกว่าอัลกอริทึม Apriori, FUP2 และ Pre-Large โดยอัลกอริทึม Apriori จะเพิ่มขึ้นเล็กน้อยตามขนาดของฐานข้อมูลปรับปรุง ในขณะที่อัลกอริทึม FUP2, Pre-Large, และอัลกอริทึมที่นำเสนอในงานวิจัยนี้จะเพิ่มขึ้นแบบชัดเจนตามจำนวนรายการที่ถูกเพิ่มเข้ามาและลบออกจากฐานข้อมูลเดิม นอกจากนี้ จะเห็นได้ว่าเวลาที่ใช้ในการประมวลผลของอัลกอริทึม FUP2 และอัลกอริทึมที่นำเสนอในงานวิจัยนี้จะน้อยกว่าเวลาที่ใช้ในการประมวลของอัลกอริทึม Apriori ก็ต่อเมื่อส่วนต่างของจำนวนรายการที่เพิ่มเข้ามาและถูกลบออกจากฐานข้อมูลเดิม ($|\Delta^+|-|\Delta^-|$) มีค่าไม่เกิน 40,000 รายการ หรือคิดเป็น 200% ของขนาดฐานข้อมูลเดิม

4.4 สรุปผลการทดลอง

จากการทดลองโดยการ 1) เพิ่มข้อมูลใหม่จำนวน 20,000 รายการ และลบข้อมูลเก่าออก 10,000 รายการ โดยกำหนดค่าสนับสนุนขั้นต่ำที่แตกต่างกันในช่วงระหว่าง 3%-7% และ 2) เพิ่มข้อมูลใหม่และลบข้อมูลเก่าออกจากฐานข้อมูลเดิมด้วยขนาดที่แตกต่างกัน โดยการกำหนดค่าสนับสนุนขั้นต่ำที่ 5% โดยใช้ชุดข้อมูลสังเคราะห์ I4T10D100K ในการทดลองจะแบ่งวัตถุประสงค์ในการวัดผลการทดลองออกเป็น 2 ประเด็น ได้แก่ 1) เพื่อทดสอบความถูกต้องของผลลัพธ์ที่ได้จากการเพิ่มข้อมูลใหม่และลบข้อมูลเก่าจากฐานข้อมูลเดิม และ 2) เพื่อทดสอบประสิทธิภาพในการทำงานของอัลกอริทึมในกรณีที่ข้อมูลถูกเพิ่มและลบออกจากฐานข้อมูลเดิม สามารถสรุปผลการทดลองทั้ง 2 ประเด็นได้ดังนี้

1. **ความถูกต้องของผลลัพธ์ที่ได้จากการเพิ่มข้อมูลใหม่และลบข้อมูลเก่าจากฐานข้อมูลเดิม** เนื่องจากอัลกอริทึมการค้นหากฎความสัมพันธ์เพิ่มขยายทั้งหมดที่ใช้ในการทดลอง ได้แก่ FUP2, Pre-Large และอัลกอริทึมที่นำเสนอในงานวิจัยนี้ ล้วนมีพื้นฐานการทำงานมาจากอัลกอริทึม Apriori ดังนั้น ในการทดสอบความถูกต้องของผลลัพธ์จะทำการเปรียบเทียบผลที่ได้จากแต่ละอัลกอริทึมกับผลที่ได้จากอัลกอริทึม Apriori จากการทดลองพบว่าอัลกอริทึมที่นำเสนอในงานวิจัยนี้ให้ผลลัพธ์ในการประมวลผลที่ถูกต้อง นั่นคือ มีจำนวนฟรีเควนท์ไอเทมเซตเท่ากับฟรีเควนท์ไอเทมเซตที่ได้จากอัลกอริทึม Apriori

2. **ประสิทธิภาพในการทำงานของอัลกอริทึมในกรณีที่ข้อมูลถูกเพิ่มและลบออกจากฐานข้อมูลเดิม** ทำการทดสอบโดยเปรียบเทียบจากเวลาที่ใช้ในการประมวลผล (Execution Time) ของแต่ละอัลกอริทึม ในกรณีที่ 1) มีการเพิ่มและลบข้อมูลด้วยค่าสนับสนุนขั้นต่ำ (Minimum support) ที่แตกต่างกัน และ 2) มีการเพิ่มและลบข้อมูลด้วยขนาดที่แตกต่างกัน ผลการทดลองพบว่าอัลกอริทึมที่นำเสนอในการทดลองนี้ใช้เวลาในการประมวลผลน้อยกว่าอัลกอริทึม Apriori, FUP2, และ Pre-Large ในทั้ง 2 กรณี

บทที่ 5
สรุปและข้อเสนอแนะ

5.1 สรุปผลการวิจัย

การค้นหากฎความสัมพันธ์ (Association rules) เป็นเทคนิคหนึ่งในการทำเหมืองข้อมูลที่ใช้ในการวิเคราะห์ความสัมพันธ์ของข้อมูลระหว่างรายการ โดยจะทำการค้นหาและดึงรูปแบบของข้อมูลระหว่างรายการต่างๆ ที่ถูกจัดเก็บไว้ในฐานข้อมูล และสร้างให้อยู่ในรูปแบบของกฎความสัมพันธ์ ซึ่งพื้นฐานของการค้นหากฎความสัมพันธ์คือการนับความถี่หรือจำนวนครั้งที่ไอเทมเกิดขึ้นร่วมกัน ไอเทมที่ปรากฏอยู่ในปต่ละรายการของฐานข้อมูลจะถูกนำมาหาความสัมพันธ์และสร้างเป็นกฎความสัมพันธ์ โดยจะแสดงอยู่ในรูปของ IF...THEN rule ไอเทมเซตที่จะนำมาสร้างเป็นกฎความสัมพันธ์ได้จะต้องมีจำนวนของข้อมูลที่เกิดขึ้นมากกว่าหรือเท่ากับตัววัด 2 ตัว คือ ค่าสนับสนุนน้อยที่สุด (minimum support threshold) และค่าความเชื่อมั่นน้อยที่สุด (Minimum confidence threshold)

ในการหากฎความสัมพันธ์โดยทั่วไปมักจะดำเนินการโดยตั้งสมมติฐานให้ฐานข้อมูลไม่มีการเปลี่ยนแปลง (Static database) แต่ในธรรมชาติของข้อมูลที่เกิดขึ้นในโลกของความเป็นจริง ฐานข้อมูลมักจะมีการเพิ่มขึ้นหรือลดลงอยู่ตลอดเวลา (Dynamic Database) ส่งผลให้เกิดการเปลี่ยนแปลงกฎความสัมพันธ์ที่ได้ค้นหาไว้แล้ว โดยอาจทำให้กฎที่มีอยู่ไม่มีความถูกต้อง ดังนั้น เพื่อปรับปรุงกฎความสัมพันธ์ให้มีความทันสมัยอยู่ตลอดเวลา จะต้องมีการประมวลผลทั้งข้อมูลเก่าแล้วข้อมูลใหม่ร่วมกันเพื่อปรับปรุงกฎความสัมพันธ์อย่างสม่ำเสมอ อย่างไรก็ดี การประมวลผลทั้งข้อมูลเก่าและใหม่ร่วมกันทำให้สิ้นเปลืองระยะเวลาในการประมวลผลเป็นอย่างมาก

จากปัญหาดังกล่าวจึงเป็นที่มาของการวิจัยทางด้านการเพิ่มขยายการค้นหากฎความสัมพันธ์ (Incremental Mining on Association Rules) ซึ่งงานวิจัยนี้จะทำการศึกษาและพัฒนาเกี่ยวกับการเพิ่มขยายการค้นหากฎความสัมพันธ์เมื่อข้อมูลข้อมูลถูกเพิ่มและลบ เพื่อลดระยะเวลาในการประมวลผลจากฐานข้อมูลเดิม

งานวิจัยหลายชิ้นได้นำพื้นฐานการทำงานจากอัลกอริทึม Apriori มาใช้ในการค้นหากฎความสัมพันธ์แบบเพิ่มขยายในกรณีข้อมูลถูกเพิ่มและลบออกจากฐานข้อมูลเดิม เช่น FUP2 และ Pre-Large อัลกอริทึมเหล่านี้มีการนำความรู้ที่ได้จากการไมนิงฐานข้อมูลเดิมมาใช้เพื่อช่วยเพิ่มประสิทธิภาพในการค้นหากฎความสัมพันธ์ นั่นคือ ฟรีเควนท์ไอเทมเซตที่ได้จากฐานฐานข้อมูลเดิมจะถูกเก็บไว้เพื่อนำมาใช้ในการคำนวณหาค่าสนับสนุนของไอเทมเซตในฐานข้อมูลปรับปรุง ทำให้สามารถลดจำนวนแคนดิเดตไอเทมเซตที่จะต้องถูกนำไปสแกนในฐานข้อมูลเดิมได้ ด้วยวิธีการประมวลผลดังกล่าวจึงทำให้อัลกอริทึม FUP2 และ Pre-Large จึงใช้เวลาในการประมวลผลน้อยกว่า Apriori อย่างไรก็ตาม ในแต่ละรอบ k เมื่อมีการค้นพบแคนดิเดตไอเทมเซตที่ไม่ได้เป็นสมาชิกของฟรีเควนท์ไอเทมเซตของฐานข้อมูลเดิม แคนดิเดตไอเทมเซตนั้นๆ จะถูกนำไปสแกนในฐานข้อมูลเดิมทุกรอบที่ k

เพื่อลดจำนวนการสแกนฐานข้อมูลเดิมให้เหลือจำนวนรอบการสแกนที่น้อยที่สุดในงานวิจัยการค้นหากฎความสัมพันธ์แบบเพิ่มขยายโดยอาศัยหลักความน่าจะเป็น [9] หรืออัลกอริทึม Probability-based ได้นำเสนอเทคนิคการประมาณค่าความน่าจะเป็นของไอเทมเซตที่คาดว่าจะเป็น

ฟรีเควนท์ไอเทมเซตสำหรับเก็บไว้เพื่อนำไปสแกนฐานข้อมูลเดิมเพียงครั้งเดียวในรอบสุดท้าย โดยพื้นฐานการหาความน่าจะเป็นของอัลกอริทึมดังกล่าวใช้หลักการหาความน่าจะเป็นเบอร์นูลลี แต่เนื่องจากอัลกอริทึม Probability-based ถูกออกแบบมารองรับเฉพาะกรณีมีข้อมูลเพิ่มเข้ามาใหม่ งานวิจัยนี้จึงใช้หลักการหาความน่าจะเป็นเบอร์นูลลีเพื่อค้นหากฎความสัมพันธ์แบบเพิ่มขยายในกรณีข้อมูลถูกเพิ่มและลบ

งานวิจัยนี้ได้นำเสนอวิธีการในการปรับปรุงฟรีเควนท์ไอเทมเซตโดยนำค่าความน่าจะเป็นของไอเทมเซตที่เกิดในฐานข้อมูลเดิมมาใช้ในการทำนายไอเทมที่คาดว่าจะเป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลปรับปรุง ซึ่งนอกจากจะสามารถช่วยให้ค้นหาฟรีเควนท์ไอเทมเซตได้อย่างถูกต้องและมีประสิทธิภาพแล้ว ยังสามารถลดจำนวนไอเทมเซตที่ต้องนำไปสแกนในฐานข้อมูลเดิมได้อีกด้วย

สำหรับการทดลองในงานวิจัยนี้ ผู้วิจัยใช้ชุดข้อมูลสังเคราห์ I4T10D100K โดยแบ่งวัตถุประสงค์ในการวัดผลการทดลองออกเป็น 2 ประเด็น ได้แก่ 1) เพื่อทดสอบความถูกต้องของผลลัพธ์ที่ได้จากการเพิ่มข้อมูลใหม่และลบข้อมูลเก่าจากฐานข้อมูลเดิม และ 2) เพื่อทดสอบประสิทธิภาพในการทำงานของอัลกอริทึมในกรณีที่ข้อมูลถูกเพิ่มและลบออกจากฐานข้อมูลเดิม ทั้งนี้ได้ทำการทดลองโดยการ 1) เพิ่มข้อมูลใหม่จำนวน 20,000 รายการ และลบข้อมูลเก่าออก 10,000 รายการ โดยกำหนดค่าสนับสนุนขั้นต่ำที่แตกต่างกันในช่วงระหว่าง 3%-7% และ 2) เพิ่มข้อมูลใหม่และลบข้อมูลเก่าออกจากฐานข้อมูลเดิมด้วยขนาดที่แตกต่างกัน โดยการกำหนดค่าสนับสนุนขั้นต่ำที่ 5% จำนวนฟรีเควนท์ไอเทมเซตที่เกิดขึ้นและเวลาที่ใช้การการประมวลผลจะถูกนำไปเปรียบเทียบกับอัลกอริทึม Apriori, FUP2, และ Pre-Large จากผลการทดลองพบว่าอัลกอริทึมที่นำเสนอในงานวิจัยนี้สามารถทำงานได้อย่างถูกต้องและมีประสิทธิภาพ

5.2 ข้อเสนอแนะ

ในงานวิจัยนี้ ในการคำนวณค่าความน่าจะเป็นเพื่อหาไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลปรับปรุงนั้น จำเป็นต้องระบุจำนวนรายการข้อมูลที่เพิ่มเข้ามาและจำนวนรายการที่ถูกลบออกจากฐานข้อมูลเดิมไว้ล่วงหน้า จึงจะสามารถคำนวณหาไอเทมเซตที่คาดว่าจะเป็นฟรีเควนท์ไอเทมเซตในฐานข้อมูลปรับปรุงได้ ซึ่งในความเป็นจริงแล้วเราไม่สามารถทราบได้ว่าจะมีรายการที่ถูกเพิ่มเข้ามาหรือถูกลบออกจากฐานข้อมูลเดิมกี่รายการ ดังนั้นในงานวิจัยครั้งต่อไป จึงควรจะค้นหาวิธีการคำนวณความน่าจะเป็นโดยไม่จำเป็นต้องทราบจำนวนรายการข้อมูลที่เพิ่มเข้ามาและจำนวนรายการที่ถูกลบออกจากฐานข้อมูลเดิม

บรรณานุกรม

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

[2] B. Nath, D. K. Bhattacharyya and A. Ghosh. **"Incremental association rule mining: a survey"**, WIREs Data Mining Knowledge Discovery, pp. 1-13, 2013.

[3] Jiawei Han, Jien Pei, and Yiwen Yin, **"Mining Frequent Patterns without Candidate Generation"**, Dallas, TX USA., 2000.

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

[5] Cheung, D.W., Lee, S.D. and Kao, B., "A general Incremental Technique for Maintaining Discovered Association Rules", In Proceedings of the fifth International Conference on Database Systems for Advanced Applications, Melbourne, Australia, April 1997.

[6] Thomas S, Bodagala S, Alsabti S, and Ranka S., **"An efficient algorithm for the incremental updation of association rules in large databases"**, In Proceeding of the 3rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1997

[7] Tzung-Pei Hong, Ching-Yao Wang and Yu-Hui Tao, "A new incremental data mining algorithm using pre-large itemsets", Intelligent Data Analysis, Vol. 5, pp. 111–129, 2001.

[8] TZUNG-PEI HONG, TZU-JUNG HUANG, "Maintenance of Generalized Association Rules for Record Deletion Based on the Pre-Large Concept", Proceedings of the 6th WSEAS Int. Conf. on Artificial Intelligence, Knowledge Engineering and Data Bases, Greece, 2007.

[9] Ratchadaporn Amornchewin and Worapoj Kreesuradej, "Mining Dynamic Databases using Probability-Based Incremental Association Rule Discovery Algorithm", Journal of Universal Computer Science, vol. 15, no. 12 (2009), 2409-2428

[10] Chang-Hung Lee, Cheng-Ru Lin, Ming-Syan Chen, **"Sliding window filtering: an efficient method for incremental mining on a time-variant database"**, Information Systems, Vol. 30, pp. 227–244, 2005.

[11] C.I. Ezeife and Y. Su. **"Mining Incremental Association Rules with Generalized FP-Tree",** In Proceedings of the 15th Conference of the Canadian Society for Computational Studies of Intelligence on Advances in Artificial Intelligence, pp.147-169, 2002.

[12] R. Srikant and R. Arawal, **"Mining generalized association rules,"** The 21st International Conference on Very Large Databases, pp. 407-419, Zurich, Switzerland, 1995.