การค้นหากฎความสัมพันธ์ด้วยการเขียนโปรแกรมเชิงตรรกะด้วยเงื่อนไขบังคับ

THE DISCOVERY OF ASSOCIATION RULES USING CONSTRAINT LOGIC PROGRAMMING

ไพชยนต์ คงไชย, นิตยา เกิดประสพ และ กิตติศักดิ์ เกิดประสพ
สาขาวิชาวิศวกรรมคอมพิวเตอร์ สำนักวิชาวิศวกรรมศาสตร์
มหาวิทยาลัยเทคโนโลยีสุรนารี อ.เมือง จ.นครราชสีมา 30000

บทคัดย่อ

การหากฎความสัมพันธ์นั้นเป็นการหาความสัมพันธ์จากข้อมูลขนาดใหญ่ เพื่อให้ได้รูปแบบความสัมพันธ์ของข้อมูลและนำไปใช้ในการวางแผนการตลาด แต่ในการหากฎความสัมพันธ์ด้วยวิธีการในปัจจุบันมีการใช้ระยะเวลาในการประมวลผลข้อมูลค่อนข้างมาก เพราะกฎที่ได้จากการวิเคราะห์ความสัมพันธ์นั้นมีจำนวนมากและบางกฎที่ได้มานั้นไม่มีประโยชน์ ในงานวิจัยนี้จึงได้เสนอการค้นหากฎความสัมพันธ์ด้วยการเขียนโปรแกรมเชิงตรรกะด้วยเงื่อนไขบังคับ ซึ่งเป็นวิธีการที่เพิ่มเงื่อนไขข้อบังคับเข้าไปในการวิเคราะห์หากฎความสัมพันธ์ เพื่อให้ผู้ใช้สามารถระบุกฎที่ปรากฏเฉพาะไอเท็มที่ต้องการและยังสามารถกำหนดความยาวของกฎได้ จากการทดลองพบว่าถ้าผู้ใช้มีการกำหนดเงื่อนไขที่เฉพาะเจาะจง เช่น เงื่อนไขและ จะทำให้จำนวนกฎและเวลาที่ใช้ในการประมวลผลลดลงถึง 96% เมื่อเทียบกับแบบไม่กำหนดเงื่อนไข

คำสำคัญ: การทำเหมืองข้อมูล, กฎความสัมพันธ์, โปรแกรมเชิงตรรกะด้วยเงื่อนไขบังคับ

ABSTRACT

The discovery of association rules is a search for relationships from large data to obtain patterns or models that are useful for marketing planning. But current techniques in association mining take much time. Moreover, the discovered rules are normally abundant, redundant, and some of them are useless. Therefore, this research proposes a constraint-based technique to scope the search space and also to reduce the search time by allowing user to specify the items of interest and length of the association rules. From experimental results we found that with specific constraint such as the condition “and”, the running time

and number of discovered rules can be reduced up to 96% when compared to mining with no constraint.

KEYWORDS: Data mining, Association Rule Mining, Constraint Logic Programming

1. บทนำ

ปัจจุบันการค้นหารูปแบบของข้อมูลเพื่อช่วยในการสนับสนุนการตัดสินใจ หรือการค้นหาความสัมพันธ์ในข้อมูลเพื่อปรับมูลค่าในคลังสินค้ามีผู้นิยมเป็นจำนวนมาก แต่การค้นหาความรู้จากข้อมูลนั้นค่อนข้างยากและซับซ้อนเนื่องจากข้อมูลที่นำมาหารูปแบบนั้นมักจะมีจำนวนมาก [1] จึงต้องใช้เวลาเตรียมข้อมูล และใช้เวลาในการค้นหาความรู้เป็นเวลานานโดยในการค้นหาความสัมพันธ์ของข้อมูลนั้นจะทำให้ได้รับความรู้หรือกฎความสัมพันธ์เป็นจำนวนมาก ซึ่งจะยุ่งยากต่อการนำไปใช้เพราะจะต้องมาคัดแยกกฎที่เราต้องการนำไปใช้ออกจากกฎทั้งหมดซึ่งในขั้นตอนคัดแยกอาจจะเกิดข้อผิดพลาดทำให้เสียกฎที่ต้องการไป

ในงานวิจัยนี้จึงมุ่งเน้นพัฒนากระบวนการในการออกแบบอัลกอริทึมและพัฒนาโปรแกรม เพื่อหากฎความสัมพันธ์จากข้อมูลตามที่ผู้ใช้กำหนด ซึ่งจะทำให้ผู้ใช้ได้รับกฎที่มีความจำเป็นต่อผู้ใช้และลดจำนวนของกฎที่ไม่จำเป็น ผู้ใช้สามารถกำหนดเงื่อนต่างๆ เพื่อให้ได้กฎตามที่ผู้ใช้ต้องการ เช่น การกำหนดสมาชิกที่ผู้ใช้ต้องการ การกำหนดสมาชิกของกฎที่ผู้ใช้ไม่ต้องการ การกำหนดขนาดของกฎ การกำหนดเป้าหมายของกฎ เป็นต้น ซึ่งปัจจัยที่ทำให้เวลาในการค้นพบและจำนวนกฎที่ลดลงก็คือ การใช้เงื่อนไขเพื่อลดขอบเขตการค้นหากฎที่จะเป็นคำตอบที่ผู้ใช้ต้องการ กระบวนการที่ใช้ค้นหากฎความสัมพันธ์ในงานวิจัยนี้ได้ใช้เทคนิคการทำเหมืองข้อมูลประเภทค้นหาความสัมพันธ์ของข้อมูล โดยใช้อัลกอริทึมที่มีความนิยมสูงคืออัลกอริทึม Apriori [2] โดยนำมาผสมผสานกับการใช้เงื่อนไขและในขั้นตอนการพัฒนาโปรแกรมได้ใช้วิธีการเขียนเชิงตรรกะด้วยเงื่อนไขบังคับ เนื่องจากเป็นเทคนิคการเขียนโปรแกรมที่สามารถระบุเงื่อนไขบังคับ เพื่อลดขนาดของขอบเขตการค้นหาคำตอบ และได้พัฒนา 3 โปรแกรมเพื่อทดสอบประสิทธิภาพของแต่ละโปรแกรมคือ โปรแกรม Apriori โปรแกรม ACIF (Association Rule Discovery With Constraints In Frequent Itemset Mining) และ โปรแกรม ACAF (Association Rule Discovery With Constraints After Itemset Mining) ซึ่งโปรแกรม ACIF เป็นโปรแกรมที่มีการทำงานเหมือนอัลกอริทึม Apriori แต่จะมีการใช้เงื่อนไขในระหว่างการค้นหาไอเท็มเซตที่ปรากฏบ่อย ส่วนโปรแกรม ACAF เป็นโปรแกรมที่มีการทำงานเหมือนอัลกอริทึม Apriori แต่จะมีการใช้เงื่อนไขบังคับหลังจากการค้นหาไอเท็มเซตปรากฏบ่อยทั้งหมด

2. ที่มาและทฤษฎีที่เกี่ยวข้อง

ในการหากฎความสัมพันธ์จากข้อมูลนั้น มีงานวิจัยหลายงานที่มุ่งเน้นการวิจัยเพื่อเพิ่มประสิทธิภาพในการค้นหากฎความสัมพันธ์ เช่น การลดเวลาในการประมวลผล การลดจำนวนกฎที่ซับซ้อน การลดทรัพยากรในการประมวลผล เป็นต้น จากการศึกษาได้สรุปงานวิจัยที่เกี่ยวข้องได้ดังต่อไปนี้

Srikant, R., Vu, Q. และ Agrawal, R. [3] ได้นำเสนอแนวคิดการทำเหมืองข้อมูลเพื่อค้นหากฎความสัมพันธ์ของข้อมูลโดยใช้เงื่อนไขบังคับ โดยใช้หลักการของการทำเหมืองข้อมูลเพื่อค้นหากฎความสัมพันธ์ (Association Rule) และได้ใช้เทคนิคเงื่อนไขบังคับ (Constraints) เพื่อให้ผู้ใช้ (User) สามารถระบุได้ว่ากฎที่มีความสัมพันธ์หรือรูปแบบของข้อมูล (Rule) ที่ได้มานั้นจะต้องประกอบด้วยไอเท็ม (Item) ที่ผู้ใช้ระบุด้วย เพื่อให้ได้กฎตามที่ผู้ใช้ต้องการ ทั้งนี้ในการใช้เงื่อนไขบังคับเข้ามาในการทำเหมืองข้อมูลแบบค้นหากฎความสัมพันธ์ สามารถช่วยลดโครงสร้างแลตทิซ (Lattice Structure) การที่โครงสร้างลดลงเนื่องมาจากเทคนิคการพรุน (Prunning) และเมื่อโครงสร้างลดลง ผลที่ตามมาก็คือทรัพยากร (Memory) ที่ใช้ในการประมวลผลและระยะเวลาในการทำงานของการทำเหมืองข้อมูลแบบค้นหากฎความสัมพันธ์ก็จะลดลงเช่นกัน แต่งานวิจัยนี้ยังเป็นเพียงแค่แนวคิด โดยยังไม่มีการดำเนินการจริง ต่อมา Gallo, A., Esposito, R., Meo, R. และ Botta, M. [4] ได้นำแนวคิดของ Srikant, R., Vu, Q. และ Agrawal, R. [3] ไปประยุกต์ใช้และได้เสนอเกี่ยวกับการเพิ่มประสิทธิภาพของกฎความสัมพันธ์โดยใช้เงื่อนไขบังคับ ในงานวิจัยนี้เขาได้พบปัญหาว่าการค้นหากฎความสัมพันธ์จากฐานข้อมูลจะทำให้พบไอเท็มที่ปรากฏบ่อยที่มีลักษณะคล้ายกันอยู่มาก จึงทำให้ใช้เวลาในการประมวลผลและใช้ทรัพยากรมาก ทีมงานนี้ได้เสนออัลกอริทึมซึ่งช่วยในการสอบถามข้อมูล (Query) [5, 6] ในฐานข้อมูลเพื่อให้ได้กฎความสัมพันธ์ตามที่ผู้ใช้ต้องการและช่วยลดเวลาในการค้นหาต่อจากนั้น Trifonov, T. และ Georgieva, T. [7] ได้นำแนวคิดการเพิ่มประสิทธิภาพของกฎความสัมพันธ์โดยใช้เงื่อนไขบังคับไปประยุกต์ใช้จริงกับข้อมูลเสียงของระฆัง โดยเขาได้นำเทคนิคการทำเหมืองข้อมูลเพื่อค้นหากฎความสัมพันธ์ด้วยเงื่อนไขบังคับมาสร้างโปรแกรมด้วยภาษาจาวาและภาษาสอบถามข้อมูล SQL เพื่อง่าย สะดวกและรวดเร็วต่อคนที่ต้องการค้นหากฎความสัมพันธ์และผู้ใช้ไม่จำเป็นต้องมีความรู้ภาษาสอบถามข้อมูล แต่การหาความกฎสัมพันธ์ของงานวิจัยนี้ได้เฉพาะข้อมูลของเสียงระฆัง

จากการศึกษางานวิจัยที่เกี่ยวข้องพบว่าในงานวิจัยที่ผ่านมา เป็นเพียงการนำแนวคิดการเพิ่มประสิทธิภาพของกฎความสัมพันธ์โดยใช้เงื่อนไขบังคับ มาพัฒนาด้วยภาษาสอบถามข้อมูล SQL ทำให้ใช้เวลาและทรัพยากรมากกว่าการพัฒนาโปรแกรมด้วยวิธีการเขียนเชิงตรรกะด้วยเงื่อนไขบังคับ [8] นอกจากนี้งานวิจัยที่ผ่านมายังไม่สามารถกำหนดขนาดของกฎความสัมพันธ์ที่ค้นพบได้

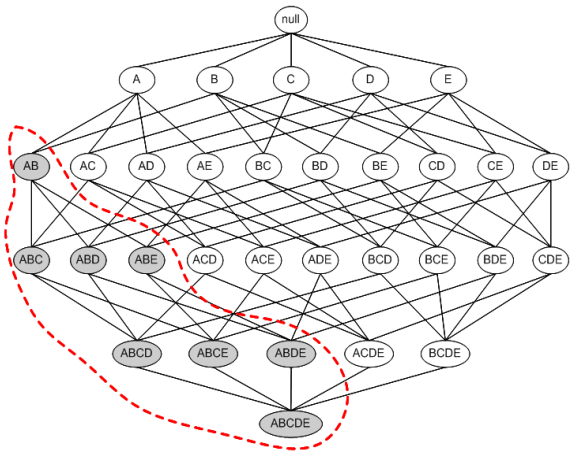
ดังนั้นงานวิจัยนี้จึงได้พัฒนาโปรแกรมหากฎความสัมพันธ์โดยใช้ภาษาโปรล็อกและมีรูปแบบการเขียนแบบใช้กฎเงื่อนไขข้อบังคับ เพื่อให้ผู้ใช้สามารถกำหนดขนาดของกฎ สมาชิกของกฎและเป้าหมายของกฎได้

3. วิธีดำเนินการวิจัย

งานวิจัยนี้มีวัตถุประสงค์ที่จะพัฒนาการทำเหมืองข้อมูลเพื่อลดปัญหา การใช้เวลาในการประมวลผลนานและกฎความสัมพันธ์ที่ได้รับจากการประมวลผลมีจำนวนมากได้ โดยโปรแกรมที่พัฒนาอนุญาตให้ผู้ใช้สามารถระบุส่วนประกอบและความยาวของกฎได้ และพัฒนาโปรแกรมด้วยภาษาโปรล็อกและได้ใช้รูปแบบการเขียนโปรแกรมเชิงตรรกะด้วยเงื่อนไขบังคับ นอกจากนี้ได้นำอัลกอริทึม Apriori มาช่วยในการหารูปแบบของกฎความสัมพันธ์ ซึ่งในหัวข้อนี้จะนำเสนอถึง การทำเหมืองข้อมูลเพื่อค้นหาความสัมพันธ์ อัลกอริทึม Apriori โปรแกรมเชิงตรรกะด้วยเงื่อนไขบังคับ การออกแบบอัลกอริทึม ACIF และ ACAF และการสอบถามเพื่อหากฎความสัมพันธ์ด้วยเงื่อนไขบังคับ

3.1 อัลกอริทึมเอไพรออริ (Apriori Algorithm)

ในปี ค.ศ.1994 Rakesh Agrawal และ Ramakrishnan Srikant [2] ได้ปรับปรุงอัลกอริทึม AIS (Agrawal Imielinski Swami) ให้ทำงานได้เร็วขึ้นและได้เรียกอัลกอริทึมที่พัฒนาขึ้นใหม่นี้ว่า อัลกอริทึม Apriori โดยใช้เทคนิค Support-based Pruning เพื่อช่วยตัดหรือลดจำนวน Candidate Itemsets แทนที่จะแจกแจงทั้งหมด ดังรูปที่ 1



รูปที่ 1 โครงสร้างแลตทิซที่มีการใช้เทคนิคการพรุน

จากรูปจะเห็นได้ว่ามีการทำเส้นประตั้งแต่ไอเท็มเซต {A, B} ที่มีสองไอเท็ม จนมาถึงไอเท็มเซต {A, B, C, D, E} ที่มีห้าไอเท็มซึ่งมีความหมายว่า ไม่สนใจไอเท็มที่มี {A, B} เป็นซับเซต เนื่องจากค่า Support ของ A และ B ไม่ถึงเกณฑ์ Minimum Support เทคนิคนี้จึงได้ตัดเซตที่มี A และ B เป็นสมาชิกออกเพื่อช่วยลดเวลาในการสร้าง Candidate Itemsets และการทำงานของอัลกอริทึม Apriori มีการทำงานเป็นแบบลูป

3.2 โปรแกรมเชิงตรรกะด้วยเงื่อนไขบังคับ (Constraint logic programming: CLP)

เป็นการเขียนโปรแกรมเหมือนโปรแกรมเชิงตรรกะปกติทั่วไปแต่ได้เพิ่มเทคนิคโดยการใส่เงื่อนไขบังคับเข้าไป โปรแกรมเชิงตรรกะปกติทั่วไปจะใช้วิธีแบบวนซ้ำด้วย Recursion [8] แต่ใน CLP จะมี Special Predicate ซึ่งเป็น Built-in อยู่ภายในซึ่งจะทำให้การเขียนโปรแกรมสะดวกยิ่งขึ้น เช่น เพรดิเคต foreach เป็นเพรดิเคตที่ทำหน้าที่วนลูปเพื่อดึงค่าที่อยู่ในลิสต์ออกมาทีละตัว และจะหยุดทำงานเมื่อลิสต์นั้นเป็นลิสต์ว่าง เพรดิเคต count เป็นเพรดิเคตที่ทำหน้าที่วนลูปเพื่อนับค่า และเพรดิเคต fromto เป็นเพรดิเคตที่ทำหน้าที่วนลูปเพื่อดึงค่าออกจากลิสต์หรือเพิ่มค่าเข้าไปในลิสต์ โดยเพรดิเคตนี้ถ้าใช้ร่วมกับ foreach จะทำหน้าที่เป็นการเพิ่มค่าเข้าไปในลิสต์ เป็นต้น

นอกจากนี้การเขียนโปรแกรมเชิงตรรกะด้วยเงื่อนไขบังคับยังเหมาะกับการแก้ปัญหาสมการทางคณิตศาสตร์ [9] สมมติว่าต้องการหาค่าของ A และ B จากสมการ 2AB = 20 โดยค่าของ A และ B มีค่า 1 ถึง 5 จากนั้นทำการประมวลผล ซึ่งจะได้ผลลัพธ์เป็นสองคำตอบคือ A = 2 และ B = 5 และคำตอบที่สองจะได้ A = 5 และ B = 2 โดยการหาคำตอบในลักษณะนี้ ได้ใช้เทคนิคการค้นหาคำตอบที่เรียกว่าเทคนิคการย้อนกลับ (Backtrack)

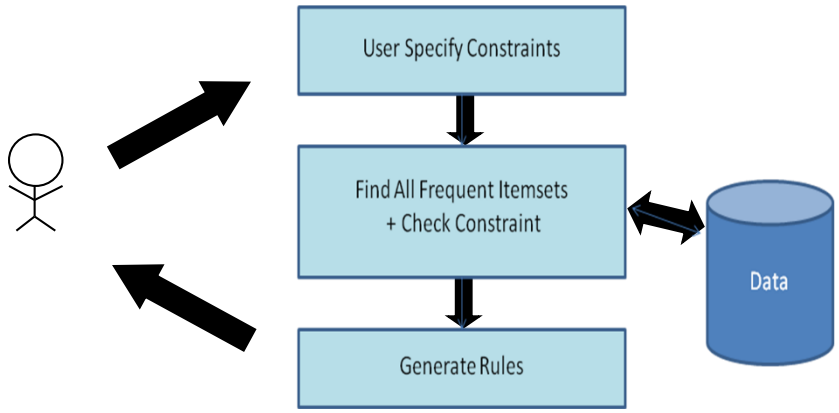
3.3 การออกแบบอัลกอริทึม ACIF และ ACAF

แนวคิดหลักของงานวิจัยนี้คือ การปรับปรุงกระบวนการค้นหากฎความสัมพันธ์ให้สามารถผนวกเงื่อนไขเงื่อนบังคับที่ผู้ใช้ระบุเข้าเป็นส่วนหนึ่งของกระบวนการเพื่อลดขอบเขตการค้นหาและเพื่อให้ผลลัพธ์ที่ตรงกับความต้องการของผู้ใช้ การผนวกเงื่อนไขบังคับนี้ผู้วิจัยได้ออกแบบไว้ 2 แนวทาง คือ ใช้เงื่อนไขบังคับขณะมีการสร้างไอเท็มเซตปรากฏบ่อย และใช้เงื่อนไขบังคับภายหลังการสร้างไอเท็มเซตปรากฏบ่อย ซึ่งมีรายละเอียดดังนี้

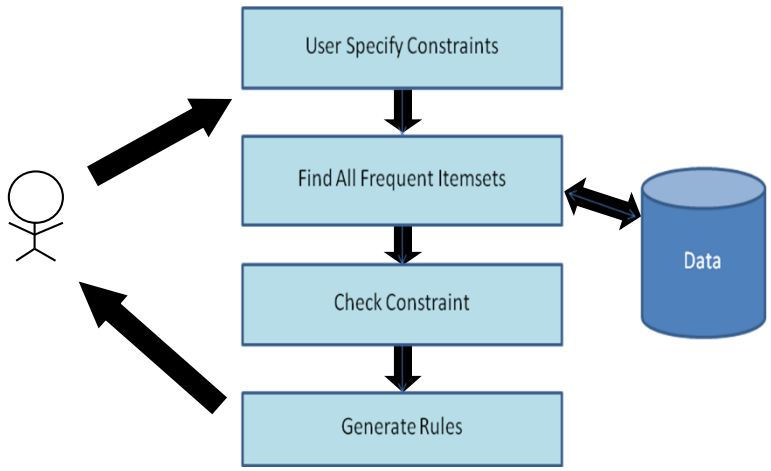
3.3.1 ใช้เงื่อนไขบังคับขณะมีการสร้างไอเท็มเซตปรากฏบ่อย ACIF (Association Rule Discovery With Constraints In Frequent Itemset Mining)

เป็นรูปแบบการหากฎความสัมพันธ์ตามที่ผู้ใช้งานระบุไอเท็มที่ต้องการหรือไม่ต้องการและขนาดของกฎได้ โดยจะมีการตรวจสอบเงื่อนไขบังคับขณะมีการสร้างไอเท็มเซตปรากฏบ่อย และมีกรอบแนวคิดการหากฎความสัมพันธ์ ดังรูปที่ 2 ซึ่งจะประกอบไปด้วย 3 ส่วน คือ

1) ผู้ใช้ (User) ซึ่งผู้ใช้มีหน้าที่ระบุค่าสนับสนุนขั้นต่ำ อย่างเช่นถ้าผู้ใช้ต้องการหากฎความสัมพันธ์ที่มีความถี่เท่ากับสอง หมายถึงกฎที่พบในทรานแซคชันสองทรานแซคชันจากทรานแซคชันทั้งหมด ผู้ใช้ก็จะต้องระบุค่าสนับสนุนขั้นต่ำเท่ากับสอง และผู้ใช้จะต้องระบุค่าความเชื่อมั่นขั้นต่ำและถ้าหากผู้ใช้ต้องการเพียงแค่บางกฎความสัมพันธ์ที่ประกอบไปด้วยไอเท็มที่สนใจผู้ใช้ก็จะต้องระบุเงื่อนไขด้วย



รูปที่ 2 การใช้เงื่อนไขบังคับขณะมีการสร้างไอเท็มเซตปรากฏบ่อย (ACIF)



รูปที่ 3 การใช้เงื่อนไขบังคับภายหลังการสร้างไอเท็มเซตปรากฏบ่อย (ACAF)

2) หาไอเท็มเซตปรากฏบ่อยทั้งหมด (Find All Frequent Itemset) ทั้งหมด ได้แก่ไอเท็มเซตที่มีค่ามากกว่าหรือเท่ากับค่าสนับสนุนขั้นต่ำ และใช้เงื่อนไขบังคับตามที่ผู้ใช้ได้ระบุมา (Check Constraints) เพื่อให้ได้ไอเท็มเซตตามที่ผู้ใช้ได้ระบุ

3) เป็นการสร้างกฎความสัมพันธ์จากไอเท็มเซตปรากฏบ่อยจากการใช้เงื่อนไขบังคับ (Generate Rules) ซึ่งการสร้างกฎความสัมพันธ์นั้น กฎจะต้องมีค่าความเชื่อมั่นมากกว่าหรือเท่ากับค่าความเชื่อมั่นขั้นต่ำ

3.3.2 ใช้เงื่อนไขบังคับภายหลังการสร้างไอเท็มเซตปรากฏบ่อย ACAF (Association Rule Discovery with Constraints after Itemset Mining)

รูปแบบนี้ได้ใช้แนวคิดของอัลกอริทึม Apriori เป็นหลักและได้เพิ่มส่วนของเงื่อนไขบังคับเข้าไปในอัลกอริทึม โดยมีกรอบแนวคิดการหากฎความสัมพันธ์ ดังรูปที่ 3 จากกรอบแนวคิดการหากฎความสัมพันธ์จะประกอบไปด้วย 4 ส่วน คือ

1) ผู้ใช้ (User) ซึ่งผู้ใช้มีหน้าที่ระบุค่าสนับสนุนขั้นต่ำ (Minimum Support) ค่าความเชื่อมั่นขั้นต่ำ (Minimum Confidence) และถ้าหากผู้ใช้ต้องการเพียงแค่บางกฎความสัมพันธ์ที่ประกอบไปด้วยไอเท็มที่สนใจผู้ใช้ก็จะต้องระบุเงื่อนไขด้วย

2) หาไอเท็มเซตปรากฏบ่อยทั้งหมด (Find All Frequent Itemset) ทั้งหมด ได้แก่ไอเท็มเซตที่มีค่ามากกว่าหรือเท่ากับค่าสนับสนุนขั้นต่ำ

3) ใช้เงื่อนไขบังคับตามที่ผู้ใช้ได้ระบุมา (Check Constraints) เพื่อให้ได้ไอเท็มเซตตามที่ผู้ใช้ได้ระบุ เช่น ผู้ใช้ต้องการไอเท็มเซตที่มีสมาชิกคือ {A} จากทรานแซคชัน {A, B}, {B ,D}, {E, C}, {A, B, D} คำตอบที่ได้จากการใช้เงื่อนไขบังคับจะทำให้ได้ไอเท็มเซตคือ {A, B}, {A, B, D} ซึ่งจะทำให้ได้ผลลัพธ์เป็นจำนวนไอเท็มเซตที่ลดลงและตรงกับความต้องการของผู้ใช้

4) จากขั้นตอนที่ 3 จะได้ไอเท็มเซตปรากฏบ่อยจากการใช้เงื่อนไขบังคับ และในขั้นตอนที่ 4 นี้จะเป็นการสร้างกฎความสัมพันธ์จากไอเท็มเซตปรากฏบ่อย (Generate Rules) ซึ่งการสร้างกฎความสัมพันธ์นั้น กฎจะต้องมีค่าความเชื่อมั่นมากกว่าหรือเท่ากับค่าความเชื่อมั่นขั้นต่ำ

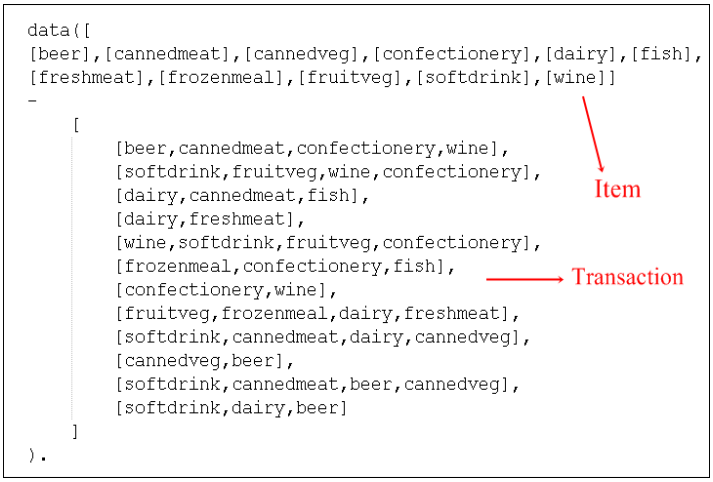
3.4 การสอบถามเพื่อหากฎความสัมพันธ์ด้วยเงื่อนไขบังคับ

การสอบถามเพื่อหากฎความสัมพันธ์จะต้องเตรียมข้อมูลให้อยู่ในรูปแบบของภาษาโปรล็อก ดังรูปที่ 4 และในการสอบถามเพื่อหาคำตอบของโปรแกรมจะอยู่ในรูปแบบการสอบถามเชิงตรรกะ โดยจะมีการสอบถามในรูปแบบต่างๆ ดังนี้

- การสอบถามแบบไม่มีเงื่อนไขพิเศษ คือ มีการทำงานเหมือนอัลกอริทึม Apriori ที่มีการกำหนดเฉพาะค่าสนับสนุนต่ำสุดและค่าเชื่อมั่นต่ำสุด

- การสอบถามแบบใช้เงื่อนไขกำหนดขนาดของกฎ คือ การสอบถามเพื่อให้ได้ขนาดสมาชิกของกฎความสัมพันธ์ตามผู้ใช้ต้องการ เช่น ถ้าผู้ใช้ต้องการกฎที่มีสมาชิกมากกว่า 2 ไอเท็ม กฎที่ได้รับก็จะมีขนาดมากกว่า 2 ไอเท็ม

- การสอบถามแบบกำหนดสมาชิกที่ต้องการ เป็นการสอบถามเพื่อให้ได้สมาชิกตามที่ผู้ใช้กำหนดโดยการใช้เงื่อนไขรูปแบบนี้จะใช้ในช่วงที่โปรแกรมขึ้นกล่องข้อความมาสามครั้งและจะใช้เงื่อนไขได้ในครั้งแรก



รูปที่ 4 รูปแบบของข้อมูลที่จะนำมาหากฎความสัมพันธ์

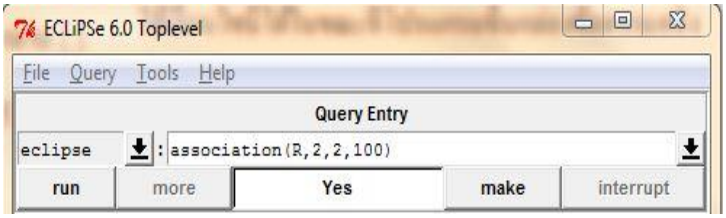
- การสอบถามแบบกำหนดสมาชิกที่ไม่ต้องการ เป็นการสอบถามเพื่อให้ได้กฎที่ไม่มีสมาชิกที่เราไม่ต้องการ โดยการใช้เงื่อนไขรูปแบบนี้จะใช้ในช่วงที่โปรแกรมขึ้นกล่องข้อความมาสามครั้งและจะใช้เงื่อนไขได้ในกล่องข้อความครั้งที่สอง

- การสอบถามแบบใช้เงื่อนไขหรือ (OR) คือ การสอบถามเพื่อให้ได้กฎความสัมพันธ์ที่ประกอบด้วยสมาชิกบางตัวตามที่ผู้ใช้ต้องการ โดยการใช้เงื่อนไขรูปแบบนี้จะใช้ในช่วงที่โปรแกรมขึ้นกล่องข้อความมาสามครั้ง

- การสอบถามแบบใช้เงื่อนไขและ (AND) คือ การสอบถามเพื่อให้ได้กฎความสัมพันธ์ที่ประกอบด้วยสมาชิกทุกตัวตามที่ผู้ใช้ต้องการ โดยการใช้เงื่อนไขรูปแบบนี้จะคล้ายกับการสอบถามแบบใช้เงื่อนไขหรือ (OR) ใช้ในช่วงที่โปรแกรมขึ้นกล่องข้อความมาสามครั้ง

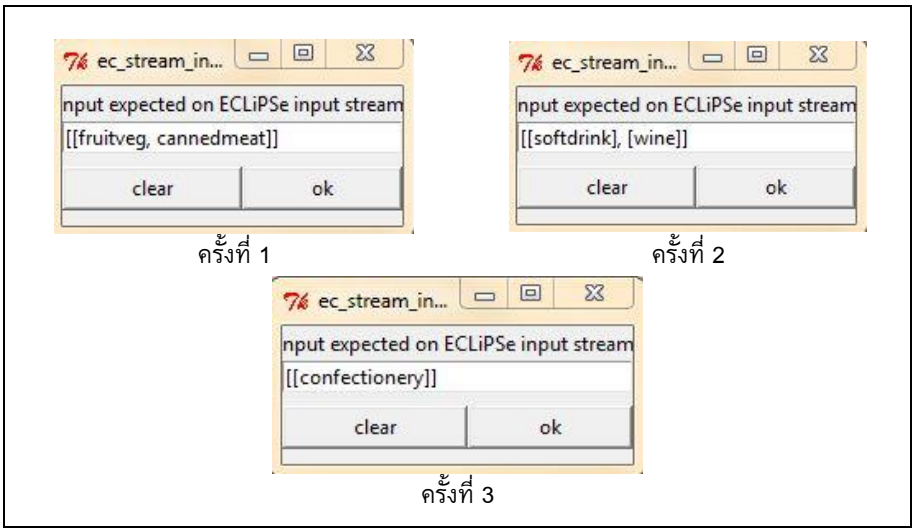
- การสอบถามแบบใช้เงื่อนไขกำหนดเป้าหมาย คือ การสอบถามโดยมีการกำหนดสมาชิกหลัง then เพื่อให้ได้กฎความสัมพันธ์ตามผู้ใช้ต้องการ และสามารถใช้เงื่อนไขนี้ได้ในขณะที่โปรแกรมขึ้นกล่องข้อความครั้งที่สาม

ตัวอย่างการสอบถามเพื่อหากฎความสัมพันธ์ที่มีขนาดมากกว่า 2 ไอเท็ม ประกอบด้วย สมาชิกกฎ คือ softdrink และ wine และไม่มี fruitveg หรือ cannedmeat เป็นสมาชิกของกฎ โดยมีเป้าหมายของกฎ คือ confectionery โดยจะมีการสอบถามดังรูปที่ 5



รูปที่ 5 ตัวอย่างการสอบถามเพื่อหากฎความสัมพันธ์

จากรูปที่ 5 เพรดิเคต association คือ เพรดิเคตสอบถามเพื่อหากฎความสัมพันธ์ หมายเลข 2 ตัวแรก หมายถึงกำหนดขนาดของกฎความสัมพันธ์มีขนาดมากกว่า 2 ไอเท็ม หมายเลข 2 ตัวถัดมาเป็นการกำหนดค่าสนับสนุนต่ำสุด และหมายเลข 100 คือค่าความเชื่อมั่นต่ำสุด หลังจากการสอบถามโปรแกรมจะขึ้นกล่องข้อความมา 3 ครั้ง ครั้งแรกกำหนดสมาชิกที่ต้องการใส่ [[softdrink], [wine]] ครั้งที่สองกำหนดสมาชิกที่ไม่ต้องการใส่ [[fruitveg, cannedmeat]] และครั้งที่สามกำหนดเป้าหมายใส่ [[confectionery]] (ดังรูปที่ 6)



รูปที่ 6 ตัวอย่างการใส่เงื่อนไขในกล่องข้อความสามครั้ง

หลังจากใส่เงื่อนไขครบทั้ง 3 กล่องข้อความและใช้ข้อมูลในการสอบถามจากรูปที่ 4 จะได้กฎความสัมพันธ์หนึ่งกฎ คือ

If [softdrink, wine] 2 then [confectionery] 2

กฎความสัมพันธ์จะอยู่ในรูปแบบ If … then … หมายเลข 2 หลัง wine หมายความว่า พบไอเท็ม softdrink และ wine 2 ทรานแซคชันจากทรานแซคชันทั้งหมด และหมายเลข 2 หลัง confectionery หมายความว่า พบไอเท็ม softdrink, wine และ confectionery 2 ทรานแซคชันจากทรานแซคชันทั้งหมด

4. เปรียบเทียบผลการวิจัยและอภิปรายผล

4.1 เปรียบเทียบผลการวิจัย

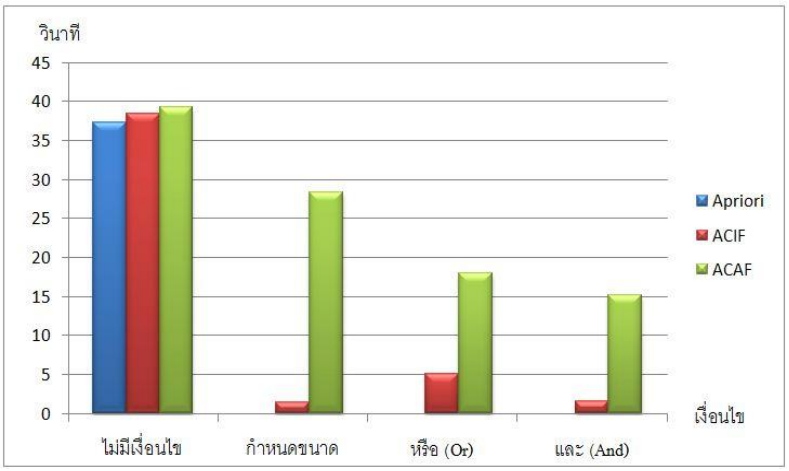
การทดสอบระบบใช้เครื่องคอมพิวเตอร์มีหน่วยประมวลผลกลาง Intel Core i7 หน่วยความจำหลัก 4 GB และข้อมูลที่ในการทดสอบใช้ข้อมูลมาตรฐาน ซึ่งเป็นข้อมูลเกี่ยวกับหมากรุก (Chess) [10] โดยมีข้อมูลทั้งหมด 3,196 แถว ประกอบด้วยแอททริบิวท์จำนวน 37 แอททริบิวท์ และทดสอบกับสามโปรแกรม คือ โปรแกรม Apriori โปรแกรม ACIF และโปรแกรม ACAF การเปรียบเทียบจะใช้เงื่อนไขในรูปแบบต่าง ๆ เพื่อหากฎความสัมพันธ์และจะใช้ตัวชี้วัด คือ เวลาในการประมวลผล (หน่วยเป็นวินาที) และจำนวนกฎความสัมพันธ์ โดยจะแสดงการเปรียบเทียบเวลาในตารางที่ 1 (รูปที่ 7) และแสดงการเปรียบเทียบจำนวนกฎความสัมพันธ์ในตารางที่ 2 (รูปที่ 8)

ตารางที่ 1 เวลาในการประมวลผลของโปรแกรม Apriori, ACIF และ ACAF

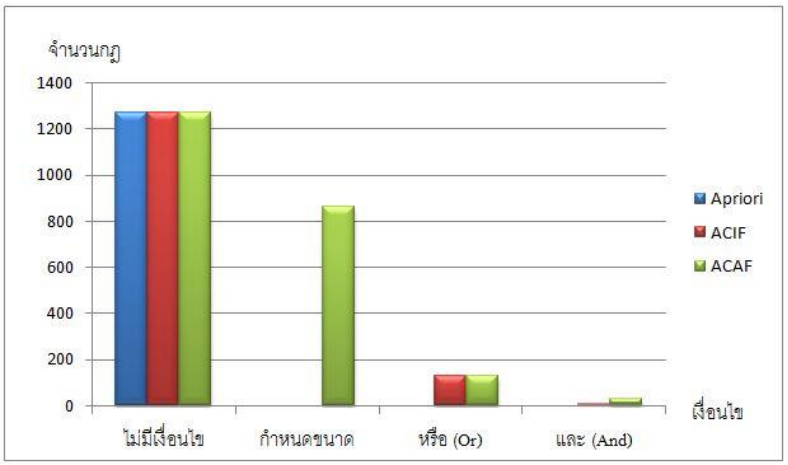
โปรแกรม	การใช้เงื่อนไขรูปแบบต่าง ๆ เพื่อหากฎความสัมพันธ์						
	ไม่มีเงื่อนไข (วินาที)	กำหนดขนาดของกฎ		ระบุไอเท็มด้วยเงื่อนไขหรือ (Or)		ระบุไอเท็มด้วยเงื่อนไขและ (And)	
	เวลาที่ใช้	เวลาที่ใช้	เวลาที่ลดลง	เวลาที่ใช้	เวลาที่ลดลง	เวลาที่ใช้	เวลาที่ลดลง
Apriori	37.29 (s)	-	-	-	-	-	-
ACIF	38.48 (s)	1.45	96%	5.09	87%	1.56	96%
ACAF	39.35 (s)	28.34	28%	18.05	55%	15.22	61%

ตารางที่ 2 เวลาในการประมวลผลของโปรแกรม Apriori, ACIF และ ACAF

โปรแกรม	การใช้เงื่อนไขรูปแบบต่าง ๆ เพื่อหากฎความสัมพันธ์			
	ไม่มีเงื่อนไข	กำหนดขนาดของกฎ	ระบุไอเท็มด้วยเงื่อนไขหรือ (Or)	ระบุไอเท็มด้วยเงื่อนไขและ (And)
Apriori	1268 กฎ	ไม่สามารถทำได้	ไม่สามารถทำได้	ไม่สามารถทำได้
ACIF	1268 กฎ	0 กฎ	130 กฎ	2 กฎ
ACAF	1268 กฎ	862 กฎ	130 กฎ	28 กฎ



รูปที่ 7 แผนภูมิการเปรียบเทียบเวลาในการประมวลผลของโปรแกรม Apriori, ACIF และ ACAF



รูปที่ 8 การเปรียบเทียบจำนวนกฎความสัมพันธ์ของโปรแกรม Apriori ACIF และ ACAF

จากตารางที่ 1 และ 2 ในการใช้เงื่อนไขในรูปแบบและสัญลักษณ์ต่าง ๆ มีคำอธิบายดังนี้

1) ไม่มีเงื่อนไข เป็นการกำหนดให้โปรแกรมไม่ใช้เงื่อนไขใดๆ นอกเหนือจากเงื่อนไขของอัลกอริทึม Apriori ที่ให้ผู้ใช้ระบุค่า Minimum Support และ Minimum Confidence

2) กำหนดขนาดของกฎ ในการทดสอบครั้งนี้ได้กำหนดให้ กฎที่ได้รับนั้นจะต้องมีขนาดสมาชิกมากกว่า 3 ไอเท็ม

3) ระบุไอเท็มด้วยเงื่อนไขหรือ (Or) ในการทดสอบได้กำหนดเงื่อนไขให้สมาชิกของกฎจะต้องมี 7 หรือ 8 และไม่มี 60 หรือ 61 เป็นสมาชิกในกฎ

4) ระบุไอเท็มด้วยเงื่อนไขและ (And) ในการทดสอบได้กำหนดเงื่อนไขให้สมาชิกของกฎจะต้องมี 7 และ 29 และไม่มี 58 และ 40 เป็นสมาชิกในกฎ

5) สัญลักษณ์ – หมายถึง ไม่สามารถทำได้ ส่วนเวลาที่ใช้ หมายถึง เวลาที่ใช้ในการประมวลผล และเวลาที่ลดลง หมายถึง เวลาที่ลดลงเมื่อเทียบกับการใช้เงื่อนไขในรูปแบบไม่มีเงื่อนไข โดยแสดงเป็นร้อยละ

4.2 อภิปรายผล

จากผลการทดสอบโปรแกรม Apriori ACIF และ ACAF ด้วยข้อมูล Chess จำนวน 3,196 รายการ และใช้รูปแบบการระบุเงื่อนไขหลายลักษณะสรุปผลการทดสอบเปรียบเทียบได้ดังนี้

1) รูปแบบไม่มีเงื่อนไข หมายถึง การค้นหากฎความสัมพันธ์ใช้เฉพาะเงื่อนไข Minimum Support และ Minimum Confidence เท่านั้นรูปแบบนี้โปรแกรมทั้งสามจะใช้เวลาที่ไม่ค่อยจะต่างกันมากทั้งนี้เพราะทั้งสามโปรแกรมมีพื้นฐานการทำงานที่เหมือนกันคือทำงานเหมือนอัลกอริทึม Apriori อยู่แล้ว ซึ่งถ้าไม่ใช้เงื่อนไขในการหาความสัมพันธ์ของข้อมูลในการทำงานของทั้งสามโปรแกรมนี้ก็แทบจะไม่แตกต่างกันเลยและจำนวนกฎที่ได้รับก็มีจำนวนที่เท่ากันด้วย

2) รูปแบบกำหนดขนาดของกฎ การประมวลผลในรูปแบบนี้พบว่าโปรแกรมทั้งสามใช้เวลาที่แตกต่างกันมาก โปรแกรม Apriori ไม่มีฟังชันก์การกำหนดขนาดของกฎจึงไม่สามารถหากฎที่มีเฉพาะกฎที่กำหนดขนาดได้ ส่วนโปรแกรม ACIF นั้นได้ใช้เวลาเพียง 1.45 วินาทีหรือ 4% ของรูปแบบไม่มีเงื่อนไข แต่ไม่พบกฎที่มีขนาดมากกว่า 3 ไอเท็มเพราะว่าโปรแกรมไม่สามารถหากฎความสัมพันธ์ที่มีขนาดมากกว่า 3 ได้เนื่องจากในการใช้เงื่อนไขระหว่างการหาไอเท็มเซตปรากฏบ่อยนั้น ได้มีการเช็คขนาดแคนดิเดตไอเท็มเซตว่าเซตที่จะเป็นไอเท็มเซตปรากฏบ่อยนั้นจะต้องมีสมาชิกมากกว่า 3 ไอเท็ม ตัวอย่างการเช็คขนาดมีดังนี้ ในการหาไอเท็มเซตรอบแรกจะไม่มีการเช็คขนาดแต่พอถึงรอบสองจะมีการเช็คขนาด ซึ่งรอบสองนั้นจะประกอบด้วยแคนดิเดตไอเท็มเซตที่มีสมาชิก 2 ตัวและจะถูกนำไปเช็คกับเงื่อนไขว่าสมาชิกของแคนดิเดตจะต้องมากกว่า 3 ไอเท็ม ซึ่งในรอบนี้ไม่มีเซตไหนผ่านเงื่อนไข จึงไม่มีการสร้างไอเท็มเซตและแคนดิเดตไอเท็มเซตขนาด 3 ไอเท็มขึ้นไปเกิดขึ้นจึงทำให้จำนวนกฎที่ได้รับนั้นเป็นศูนย์ ในส่วนโปรแกรม ACAF นี้จะใช้เวลามากที่สุด

เพราะการใช้เงื่อนไขกำหนดขนาดจะใช้หลังจากหาไอเท็มปรากฏบ่อยบ่อยทั้งหมดแล้วจะได้ไอเท็มเซตทุกขนาด จึงทำให้สามารถเช็คเงื่อนไขได้และจะใช้เวลามากกว่าโปรแกรมอื่นแต่จะได้กฎที่ต้องการครบถ้วน

3) รูปแบบระบุไอเท็มด้วยเงื่อนไขหรือ (Or) โปรแกรม Apriori ไม่สามารถรูปแบบนี้ได้ เพราะว่าในโปรแกรมไม่สามารถใช้เงื่อนไขนี้ได้ จึงไม่สามารถหากฎที่มีเฉพาะกฎที่กำหนดเงื่อนไขไว้ได้ ส่วนโปรแกรม ACIF จะใช้เวลาค่อนข้างน้อยกว่า โปรแกรม ACAF เพราะโปรแกรม ACIF มีการใช้เงื่อนไขในระหว่างการหาไอเท็มเซตปรากฏบ่อยจึงทำให้การประมวลผลใช้เวลาน้อยกว่าการประมวลผลหลังจากการหาไอเท็มปรากฏบ่อยด้วยโปรแกรม ACAF แต่กฎที่ได้ก็มีจำนวนที่เท่ากันและเหมือนกัน

4) รูปแบบระบุไอเท็มด้วยเงื่อนไขและ (And) โปรแกรม Apriori ไม่สามารถใช้เงื่อนไขนี้ได้ ส่วนโปรแกรม ACIF จะใช้เวลาน้อยที่สุดจากทั้ง 3 โปรแกรมและจะได้ผลลัพธ์เป็นกฎความสัมพันธ์เพียง 2 กฎ ซึ่งกฎที่ได้รับนั้นจะได้รับกฎไม่ครบตามเงื่อนไขและกฎที่ได้รับมีขนาด 2 ไอเท็ม เป็นเพราะว่าในการใช้รูปแบบและจะเป็นการเลือกไอเท็มเซตที่มีสมาชิกประกอบด้วยเงื่อนไขที่ผู้ใช้กำหนดเช่นถ้ามีแคนดิเดตไอเท็มเซตขนาด 2 ไอเท็มอยู่ 5 ชุด คือ {a, b}, {a, c}, {a, d}, {b, e}, {a, f} สมมติผู้ใช้กำหนดเงื่อนไขที่ต้องการเป็น a และ b ไอเท็มเซตที่ผ่านเงื่อนไขก็คือ {a, b} เพียงชุดเดียว ซึ่งในการสร้างแคนดิเดตจำนวน 3 ไอเท็มไม่สามารถสร้างได้เพราะมีไอเท็มเซตที่ผ่านเงื่อนไขมาเพียงชุดเดียวจึงเป็นสาเหตุที่ทำให้ได้รับกฎที่น้อยกว่าความเป็นจริง ซึ่งจะต่างจากโปรแกรม ACAF ที่จะใช้เวลาค่อนข้างมากแต่จะให้กฎที่ครบถ้วนตามเงื่อนไข

5. สรุปผลการวิจัย

การทำเหมืองข้อมูลเพื่อหากฎความสัมพันธ์ในปัจจุบัน จะใช้เวลาค่อนข้างมากและกฎความสัมพันธ์ที่ได้รับ อาจจะมีมากเกินความต้องการของผู้ใช้งาน ทำให้ผู้ใช้งานนำไปประยุกต์ใช้ได้ยาก ดังนั้นงานวิจัยนี้จึงได้นำเสนอการค้นหากฎความสัมพันธ์ด้วยการเขียนโปรแกรมเชิงตรรกะด้วยเงื่อนไขบังคับ ซึ่งเป็นวิธีการเพิ่มเงื่อนไขข้อบังคับตามที่ผู้ใช้ระบุ โดยสามารถระบุไอเท็มและขนาดของกฎได้ เพื่อให้ได้รับกฎความสัมพันธ์ตามที่ผู้ใช้ต้องการ ในการทดลองได้ทดสอบประสิทธิภาพด้วย 3 โปรแกรม คือ โปรแกรม Apriori โปรแกรม ACAF และโปรแกรม ACIF เพื่อเปรียบเทียบจำนวนกฎความสัมพันธ์ที่ได้รับและเวลาที่ใช้ในการประมวลผล ซึ่งผลการทดลองแสดงให้เห็นว่า ถ้าไม่ใช้เงื่อนไขเพิ่มเติมในการค้นหาความสัมพันธ์ทั้ง 3 โปรแกรมจะให้ผลลัพธ์เป็นกฎความสัมพันธ์ที่มีจำนวนกฎที่เท่ากันและเวลาในการประมวลผลไม่แตกต่างกันมาก แต่ถ้าหากใช้เงื่อนไขกำหนดขนาดของสมาชิกโปรแกรม Apriori จะไม่สามารถกำหนดขนาดของกฎได้และโปรแกรม ACIF จะสามารถกำหนดขนาดได้ที่เงื่อนไขสมาชิกที่ไม่เกิน 2 ไอเท็ม แต่ถ้ากำหนดขนาดมากกว่านั้นโปรแกรมไม่สามารถค้นหากฎได้ ส่วนโปรแกรม ACAF จะให้กฎที่ครบถ้วนตามเงื่อนไขที่กำหนดขนาด แต่ถ้าใช้

เงื่อนไขแบบ “Or” โปรแกรม ACIF จะใช้เวลาในการค้นหากฎน้อยมากและให้กฎที่ครบถ้วน ส่วนโปรแกรม ACAF จะใช้เวลาค่อนข้างมากและจะได้รับกฎที่ครบถ้วนเช่นกัน ส่วนการใช้เงื่อนไขสุดท้ายคือ “And” โปรแกรม ACIF จะใช้เวลาน้อยแต่ให้กฎที่ถูกต้องไม่ครบถ้วน ส่วนโปรแกรม ACAF จะใช้เวลามากกว่าแต่จะให้กฎที่ครบถ้วน จึงสรุปได้ว่าโปรแกรม ACAF ถึงจะใช้เวลามากกว่าโปรแกรม ACIF แต่จะให้กฎความสัมพันธ์ที่ครบถ้วนและถูกต้องมากกว่า

กิตติกรรมประกาศ

งานวิจัยนี้ได้รับทุนสนับสนุนจากมหาวิทยาลัยเทคโนโลยีสุรนารี

เอกสารอ้างอิง

[1] Agrawal, R., Imielinski, T., Swami, A. (1993). “Mining association rules between sets of items in large databases” **Proceedings of the ACM SIGMOD International Conference on Management of Data.** pp. 207 - 216.

[2] Agrawal, R., Srikant, R. (1994). “Fast algorithms for mining association rules”. **Proceedings of the International Conference on very Large Databases**. pp. 487 - 499.

[3] Srikant, R., Vu, Q., Agrawal, R. (1997). “Mining association rules with item constraints”. **Proceedings of 1997 ACM KDD**. pp. 67 - 73.

[4] Gallo, A., Esposito, R., Meo, R., Botta, M. (2005). “Optimization of Association Rules Extraction Through Exploitation of Context Dependent Constraints”, **AI*IA 2005**. pp. 258 - 269.

[5] Jeudy, B., Boulicaut, J. (2002). “Costraint-Based discovery and inductive query: application to association rule mining”. **Pattern Detection and Discovery**. pp. 110 - 124.

[6] Ng, R.T., Lakshmanan, L.V.S., Han, J., Pang, A. (1998). “Exploratory mining and pruning optimizations of constrained association rules”. **Proceedings of 1998 ACM SIGMOD Int. Conf. Management of Data**. pp. 13 - 24.

[7] Trifonov, T., Georgieva, T. (2009). “Application for Discovering the Constraint-Based Association Rules in an Archive for Unique Bulgarian Bells”. **European Journal of Scientific**. Vol.31. (No.3): pp. 366 - 371.

[8] Kerdprasop, K., Kerdprasop, N. (2006). “Frequent pattern discovery with declarative programming”. **Proceedings of the 1st Rajamangala University of Technology Conference. Trang, Thailand**.

[9] Simonis, H. (2008). **Constraint logic programming**. COSYTEC SA.

[10] Goethals, B. (2014). **Frequent Itemset Mining Dataset Repository. [Online].** Available**:** http://fimi.ua.ac.be/data/

ประวัติผู้เขียนบทความ

ไพชยนต์ คงไชย ปัจจุบันเป็นนักศึกษาปริญญาเอกที่มหาวิทยาลัยเทคโนโลยีสุรนารี สาขาวิศวกรรมคอมพิวเตอร์ สำเร็จการศึกษาในระดับปริญญาตรีและปริญญาโทในสาขาวิศวกรรมคอมพิวเตอร์ มหาวิทยาลัยเทคโนโลยีสุรนารี เมื่อปี พ.ศ. 2553 และ 2555 ตามลำดับ งานวิจัยที่มีความสนใจ คือ การทำเหมืองข้อมูลโดยการใช้เงื่อนไขรวมถึงบังคับ การหากฎความสัมพันธ์ การเขียนโปรแกรมเชิงฟังก์ชันและการเขียนโปรแกรมเชิงตรรกะ สถิติและการเรียนรู้ของเครื่อง อีเมล์: Zaguraba_ii@hotmail.com

รองศาสตราจารย์ ดร. นิตยา เกิดประสพ เป็นอาจารย์ประจำสาขาวิศวกรรมคอมพิวเตอร์ มหาวิทยาลัยเทคโนโลยีสุรนารี สำเร็จการศึกษาในระดับปริญญาตรีสาขารังสีเทคนิคจากมหาวิทยาลัยมหิดลในปี พ.ศ. 2528 และปริญญาโทสาขาวิทยาการคอมพิวเตอร์จากมหาวิทยาลัยสงขลานครินทร์ในปี พ.ศ. 2534 และปริญญาเอกในสาขาวิทยาการคอมพิวเตอร์จากมหาวิทยาลัย Nova Southeastern ที่สหรัฐอเมริกาในปี พ.ศ. 2542 เป็นสมาชิกสมาคมวิชาชีพ ACM และ IEEE Computer Society งานวิจัยที่มีความสนใจ คือ การค้นหาความรู้ในฐานข้อมูล ปัญญาประดิษฐ์ การเขียนโปรแกรมเชิงตรรกะ และฐานข้อมูลอัจฉริยะ อีเมล์: nittaya@sut.ac.th

รองศาสตราจารย์ ดร. กิตติศักดิ์ เกิดประสพ เป็นหัวหน้าสาขาวิศวกรรมคอมพิวเตอร์ มหาวิทยาลัยเทคโนโลยีสุรนารี สำเร็จการศึกษาในระดับปริญญาตรีภาควิชาคณิตศาสตร์จากมหาวิทยาลัยศรีนครินทรวิโรฒในปี พ.ศ. 2529 สำเร็จการศึกษาระดับปริญญาโทสาขาวิทยาการคอมพิวเตอร์จากมหาวิทยาลัยสงขลานครินทร์ในปี พ.ศ. 2534 และปริญญาเอกในสาขาวิทยาการคอมพิวเตอร์จากมหาวิทยาลัย Nova Southeastern ที่สหรัฐอเมริกาในปี พ.ศ. 2542 งานวิจัยที่มีความสนใจ คือ เหมืองข้อมูล ปัญญาประดิษฐ์ การเขียนโปรแกรมเชิงฟังก์ชันและการเขียนโปรแกรมเชิงตรรกะ สถิติและการคำนวณ อีเมล์: kerdpras@sut.ac.th