การจัดกลุ่มลำดับย่อยของข้อมูลอนุกรมเวลาโดยปราศจากพารามิเตอร์

นายนวิน มาดิการ

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2556

PARAMETER-FREE SUBSEQUENCE TIME SERIES CLUSTERING

Mr. Navin Madicar

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering Program in Computer Engineering
Department of Computer Engineering
Faculty of Engineering
Chulalongkorn University
Academic Year 2013

หัวข้อวิทยานิพนธ์	การจัดกลุ่มลำดับย่อยของข้อมูลอนุกรมเวลาโดยปราศจากพารามิเตอร์
โดย	นายนวิน มาดิการ
สาขาวิชา	วิศวกรรมคอมพิวเตอร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	ผู้ช่วยศาสตราจารย์ ดร.โชติรัตน์ รัตนามหัทธนะ

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

.....คณบดีคณะวิศวกรรมศาสตร์
(รองศาสตราจารย์ ดร.บุญสม เลิศหิรัญวงศ์)

คณะกรรมการสอบวิทยานิพนธ์

.....ประธานกรรมการ
(ศาสตราจารย์ ดร.บุญเสริม กิจศิริกุล)

.....อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก
(ผู้ช่วยศาสตราจารย์ ดร.โชติรัตน์ รัตนามหัทธนะ)

.....กรรมการ
(ดร.พีรพล เวทีกูล)

.....กรรมการภายนอกมหาวิทยาลัย
(รองศาสตราจารย์ ดร.กฤษณะ ไวยมัย)

นวิน มาดิการ : การจัดกลุ่มลำดับย่อยของข้อมูลอนุกรมเวลาโดยปราศจากพารามิเตอร์. (PARAMETER-FREE SUBSEQUENCE TIME SERIES CLUSTERING) อ.ที่ปรึกษาวิทยานิพนธ์หลัก : ผศ. ดร.โชติรัตน์ รัตนามหัทธนะ, 79 หน้า.

การจัดกลุ่มลำดับย่อยของข้อมูลอนุกรมเวลา เป็นการจัดกลุ่มรูปแบบหนึ่งในส่วนของงานวิจัยในด้านการทำเหมืองข้อมูลอนุกรมเวลา ซึ่งจะทำการพิจารณาบนข้อมูลอนุกรมเวลาหนึ่ง ๆ และจัดกลุ่มให้กับลำดับย่อยภายในข้อมูลอนุกรมเวลานั้น โดยวิเคราะห์จากความคล้ายคลึงกันของข้อมูล ลำดับย่อยที่มีความคล้ายคลึงกันของข้อมูลสูงจะถูกจัดอยู่ในกลุ่มเดียวกัน ในขณะที่ลำดับย่อยที่มีความคล้ายคลึงกันของข้อมูลต่ำจะถูกจัดอยู่ในกลุ่มที่ต่างออกไป โดยมีหลักเกณฑ์ที่สำคัญคือ ลำดับย่อยทุกลำดับไม่จำเป็นต้องถูกจัดกลุ่มทั้งหมด และ ลำดับย่อยที่ถูกจัดกลุ่มจะต้องไม่มีการซ้อนทับกัน ในงานวิจัยที่ผ่านมาทั้งหมดเกี่ยวกับการจัดกลุ่มลำดับย่อยของข้อมูลอนุกรมเวลา มีความจำเป็นที่จะต้องระบุพารามิเตอร์สำหรับกำหนดค่าความยาวของลำดับย่อยก่อนที่จะทำการจัดกลุ่ม ซึ่งก่อให้เกิดปัญหาสำคัญสองประการ คือ 1. เป็นการยากที่ผู้ใช้จะทราบค่าที่เหมาะสมในการจัดกลุ่มได้ บางครั้งต้องอาศัยความรู้จากผู้เชี่ยวชาญเฉพาะด้านของข้อมูลประเภทนั้น หรือแย่ไปกว่านั้นในกรณีที่ข้อมูลมีความซับซ้อนมาก ๆ แม้แต่ผู้เชี่ยวชาญเองก็ไม่สามารถระบุค่าที่เหมาะสมได้ และ 2. ความยาวของลำดับย่อยในการจัดกลุ่มจะถูกจำกัดโดยค่าพารามิเตอร์ที่กำหนดลงไปนี้ ทำให้ขาดอิสระในการจัดกลุ่มที่แท้จริง เพราะโดยทั่วไปแล้วในข้อมูลอนุกรมเวลาหนึ่ง ๆ ไม่จำเป็นที่ลำดับย่อยในแต่ละกลุ่มจะต้องมีความยาวเท่ากัน ทั้งสองปัญหานี้นำไปสู่ความไม่แม่นยำในการจัดกลุ่ม จึงเป็นที่มาของการนำเสนอการจัดกลุ่มลำดับย่อยของข้อมูลอนุกรมเวลาโดยปราศจากพารามิเตอร์ เพื่อให้ได้การจัดกลุ่มลำดับย่อยที่ง่ายต่อการใช้งานและอิสระ ครอบคลุมการจัดกลุ่มในทุกความยาว โดยการนำหลักของการค้นพบโมทีฟความยาวเหมาะสมสาหรับข้อมูลอนุกรมเวลามาประยุกต์ใช้สำหรับสร้างกลุ่มตั้งต้นที่มีความยาวเหมาะสม ซึ่งอาจประกอบด้วยความยาวเท่าใดก็ได้ ก่อนที่จะทำการคัดเลือกลำดับย่อยในความยาวที่เหมาะสมเหล่านี้มาทำการจัดกลุ่มต่อไป ทั้งนี้ได้ทำการทดลองเพื่อวัดประสิทธิภาพในแง่ของความแม่นยำเทียบกับวิธีการก่อนหน้าที่ต้องกำหนดพารามิเตอร์ โดยกำหนดพารามิเตอร์ที่ค่าจริงให้กับอัลกอริทึม เพื่อให้เห็นว่าวิธีการที่นำเสนอก็สามารถให้ผลลัพธ์ที่ดีใกล้เคียงกัน มากไปกว่านั้นยังได้ผลลัพธ์ที่เหนือกว่าอย่างเห็นได้ชัดในกรณีที่ข้อมูลอนุกรมเวลาประกอบด้วยกลุ่มของลำดับย่อยที่มีความยาวหลากหลาย

ภาควิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อนิสิต .....

สาขาวิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อ อ.ที่ปรึกษาวิทยานิพนธ์หลัก .....

ปีการศึกษา 2556

5570490121 : MAJOR COMPUTER ENGINEERING

KEYWORDS : STS Clustering / Parameter-Free / Time series / MDL

NAVIN MADICAR : PARAMETER-FREE SUBSEQUENCE TIME SERIES CLUSTERING. ADVISOR : ASST. PROF. CHOTIRAT RATANAMAHATANA, Ph.D., 79 pp.

Subsequence time series clustering, or STS Clustering, is one of the clustering methods in time series mining research. STS clustering considers a single time series and decomposes it to several subsequences. Then, it clusters similar subsequences together in a same group while the different subsequences are placed in distinct groups. The process runs with some constraints where not all subsequences in the time series need to be clustered (some subsequences are ignored) and the subsequences in any clusters must not overlap with each other. In prior research of STS clustering, all of them need at least one predefined parameter to define the width of the subsequences to be clustered that causes 2 major problems. First, it is a hard task for user to know the proper width of the subsequences to be clustered. Sometimes, they need some information from a domain expert, or to make things worse, even the domain expert cannot define what the proper width is if the time series is very complicated. Second, the width of the subsequences to be clustered is fixed by the predefined parameter. This limits the ability of the clustering to be inaccurate because the width of the subsequences should be allowed to be freely variant. Thus, the parameter-free STS Clustering algorithm is proposed in this thesis to solve the above problems. The proper length motif discovery algorithm is applied to find the initial clusters of the proper widths which can be any values, and then the rest of the subsequences are determined after (to be assigned into the initial groups or to be created as a new group). Absolutely, there are the experimental results in supporting this algorithm. The results show that the clustering's accuracy of this algorithm is comparable to the prior algorithm which requires predefined parameter. Even when the actual parameter is given, this algorithm can produce the comparable results. Moreover, this algorithm clearly outperforms the prior one in case of the time series containing subsequences of variable widths.

Department : Computer Engineering Student's Signature .....

Field of Study : Computer Engineering Advisor's Signature .....

Academic Year : 2013 .....

กิตติกรรมประกาศ

วิทยานิพนธ์เล่มนี้สำเร็จลุล่วงได้จากการสนับสนุนของผู้มีพระคุณหลาย ๆ ท่านไม่ว่าจะเป็นทั้งด้านการให้คำปรึกษา การให้ความรู้ รวมถึงการพูดคุยหรือการให้กำลังใจ ข้าพเจ้าเองได้บทเรียนและพัฒนาการสำหรับตนเองมากมายจากหลักสูตรการศึกษา การทำงานวิจัย รวมถึงการเขียนวิทยานิพนธ์เล่มนี้ ซึ่งถือเป็นประสบการณ์ที่ดีและมีค่าอย่างหนึ่งในชีวิต

ทั้งนี้ขอกล่าวขอบพระคุณอาจารย์ที่ปรึกษา ผู้ช่วยศาสตราจารย์ ดร.โชติรัตน์ รัตนามหัทธนะ เป็นท่านแรก ที่ให้ความดูแลเป็นอย่างดี ชี้แนะแนวทางในการทำงานวิจัยและให้คำปรึกษาต่าง ๆ มากมายที่ล้วนมีประโยชน์ อีกทั้งยังช่วยส่งเสริมและพัฒนาทักษะภาษาอังกฤษของข้าพเจ้า

ขอขอบพระคุณคณะกรรมการสอบวิทยานิพนธ์ ประกอบด้วย ศาสตราจารย์ ดร.บุญเสริม กิจศิริกุล ดร.พีรพล เวทีกูล และ รองศาสตราจารย์ ดร.กฤษณะ ไวยมัย ที่สละเวลามารับฟังการนำเสนอและช่วยตรวจสอบวิทยานิพนธ์ รวมถึงชี้แนะแนวทางในการทำและแก้ไขงานวิจัยให้ดีมีประสิทธิภาพ

ขอขอบคุณพี่ ๆ และ เพื่อน ๆ ในห้องปฏิบัติการทุกคน ที่คอยแสดงความคิดเห็น แสดงความเป็นห่วง ให้คำปรึกษาและให้ความช่วยเหลือต่าง ๆ ตลอดมา

สุดท้ายนี้ขอขอบพระคุณบิดามารดาและญาติทั้งหลาย ที่เป็นกำลังใจและให้การสนับสนุนเรื่อยมา

สารบัญ

หน้า

หน้า

หน้า

สารบัญตาราง

หน้า

หน้า

สารบัญภาพ

หน้า

หน้า

หน้า

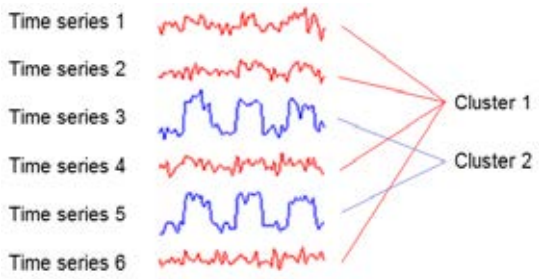
หน้า

หน้า

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

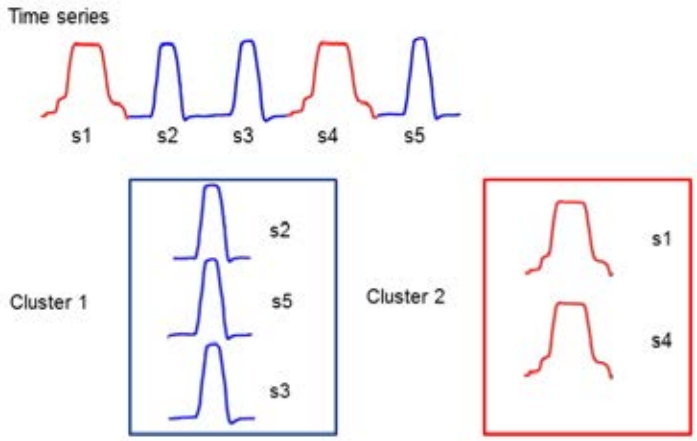
การจัดกลุ่มข้อมูลอนุกรมเวลา (Time Series Clustering) เป็นงานหนึ่งในหลาย ๆ งานที่ได้รับความสนใจมาก ในส่วนของการทำเหมืองข้อมูลอนุกรมเวลา (Time Series Mining) [8] ซึ่งเข้ามามีบทบาทอย่างมากในปัจจุบัน เนื่องจากมีการใช้งานข้อมูลอนุกรมเวลาอย่างแพร่หลายในหลากหลายสาขา เช่น ทางการแพทย์ใช้เก็บข้อมูลคลื่นหัวใจ, ชีพจร, คลื่นสมอง ทางด้านธุรกิจใช้เก็บข้อมูลทางการตลาด, ข้อมูลราคาหุ้น และอื่น ๆ อีกหลายสาขา โดยการจัดกลุ่มข้อมูลอนุกรมเวลาแบ่งออกเป็น 2 ประเภทหลัก ๆ คือ การจัดกลุ่มข้อมูลอนุกรมเวลาแบบทั้งอนุกรม (Whole Time Series Clustering) [11] และ การจัดกลุ่มลำดับย่อยของข้อมูลอนุกรมเวลา (Subsequence Time Series Clustering) [10][12][5][7] ทั้งสองแบบจะทำการพิจารณาความคล้ายคลึงกันของข้อมูลในแต่ละหน่วยย่อยและทำการจัดกลุ่มให้กับข้อมูลแต่ละหน่วยนั้น แตกต่างกันที่แบบแรกวิเคราะห์จากข้อมูลอนุกรมเวลาหลากหลายอนุกรม และทำการจัดกลุ่มให้กับข้อมูลแต่ละอนุกรมโดยพิจารณาทั้งอนุกรมเป็นหน่วยเดียว (ดังภาพที่ 1.1)



ภาพที่ 1.1 การจัดกลุ่มข้อมูลอนุกรมเวลาแบบทั้งอนุกรม
ข้อมูลอนุกรมเวลาทั้ง 6 ชุด ถูกจัดให้อยู่ใน 2 กลุ่ม ตามความคล้ายคลึงกันของข้อมูล

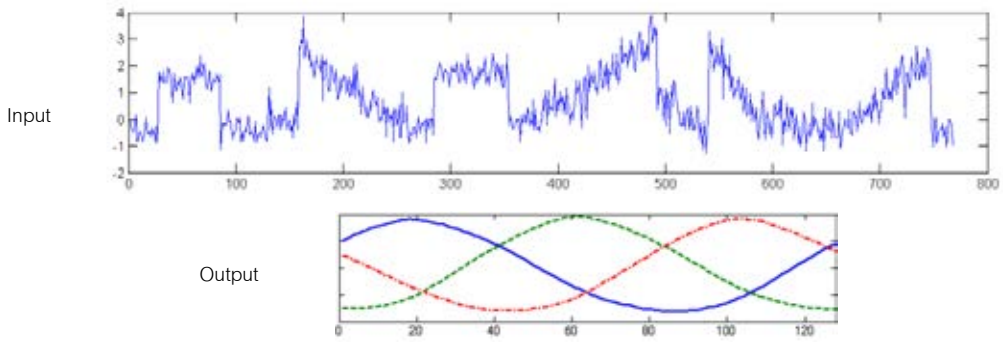
ในขณะที่แบบที่สองวิเคราะห์จากข้อมูลอนุกรมเวลาเพียงหนึ่งอนุกรม และพิจารณาจัดกลุ่มให้กับแต่ละลำดับย่อยภายในข้อมูลอนุกรมเวลานั้น (ดังภาพที่ 1.2) งานวิจัยส่วนมากให้ความสำคัญกับการจัดกลุ่มแบบแรก ทั้ง ๆ ที่การจัดกลุ่มแบบที่สองก็มีความสำคัญไม่แพ้กัน เพราะสามารถนำไปใช้ต่อในงาน

ส่วนอื่น ๆ ของการทำเหมืองข้อมูลอนุกรมเวลาได้ เช่น การค้นพบหลักเกณฑ์ (Rule Discovery) การจำแนกประเภท (Classification) การตรวจจับสิ่งผิดปรกติ (Anomaly Detection) เป็นต้น



ภาพที่ 1.2 การจัดกลุ่มลำดับย่อยของข้อมูลอนุกรมเวลา 1 ชุด
ลำดับย่อย 5 ชุด ถูกจัดให้อยู่ใน 2 กลุ่ม ตามความคล้ายคลึงกันของข้อมูล

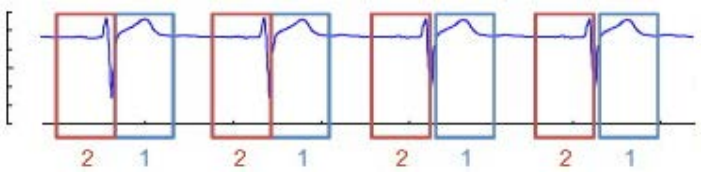
เหตุผลที่การจัดกลุ่มลำดับย่อยของข้อมูลอนุกรมเวลาไม่เป็นที่นิยมนั้นเนื่องจากมีงานวิจัย [1] ที่แสดงให้เห็นถึงความล้มเหลวในผลลัพธ์ของการจัดกลุ่มลำดับย่อยด้วยวิธีที่ถูกนำเสนอเป็นครั้งแรกใน [2] (ดังภาพที่ 1.3 อัลกอริทึมนี้ให้ผลลัพธ์เป็นรูปคลื่นไซน์ซึ่งต่างจากข้อมูลนำเข้าโดยสิ้นเชิง)



ภาพที่ 1.3 ผลลัพธ์รูปคลื่นไซน์จากการทำการจัดกลุ่มด้วยวิธี [2]
สีแต่ละสีแทนตัวแทนของแต่ละกลุ่ม ในที่นี้ผลลัพธ์มี 3 กลุ่ม

(ที่มา: Fujimaki, R., Hirose, S. and Nakata, T. Theorectical Analysis of Subsequence Time-Series Clustering from a Frequency-Analysis Viewpoint. In Proceedings of the SIAM International Conference on Data Mining (SDM), pp. 506-517, 2008.)

ทำให้งานวิจัยอื่น ๆ ที่ได้มีการนำเอาวิธีการนี้ไปใช้ถูกมองว่าล้มเหลวตามไปด้วย และถึงแม้จะมีงานวิจัยจำนวนหนึ่งที่พยายามจะแก้ไขปัญหาในจุดนี้ตามมา [10][12] แต่ไม่มีงานวิจัยใดที่แก้ปัญหานี้ได้ที่ดีเท่าที่ควร ความสนใจของปัญหาจึงค่อย ๆ ลดลงเนื่องจากความยากของปัญหา จนกระทั่งเมื่อไม่นานมานี้มีงานวิจัยที่ประสบความสำเร็จในการแก้ปัญหานี้โดยนำเสนอการจัดกลุ่มลำดับย่อยด้วยวิธีใหม่ [5][7] ที่มีหลักสำคัญ คือ ลำดับย่อยของข้อมูลอนุกรมเวลาไม่จำเป็นต้องถูกจัดกลุ่มทั้งหมด และต้องไม่มีการซ้อนทับกันของลำดับย่อยที่ถูกจัดกลุ่ม (ดังภาพที่ 1.4) เพราะข้อมูลบางส่วนอาจเป็นสัญญาณรบกวน หรือ ข้อมูลที่ไม่มีความหมายก็ได้ ดังนั้นการพยายามที่จะจัดกลุ่มให้กับข้อมูลเหล่านี้ทั้งหมด รวมทั้งการไม่คำนึงถึงส่วนที่ซ้อนทับกันของข้อมูล จึงนำไปสู่ผลลัพธ์ที่ล้มเหลว ดังที่ได้เห็นในงานวิจัยก่อน ๆ และจากผลการทดลองที่ถูกนำเสนอประกอบกับการทำการทดลองโดยตรงทำให้เห็นว่าวิธีการใหม่นี้สามารถให้ผลลัพธ์ที่มีความหมายกับการจัดกลุ่มลำดับย่อยได้



ภาพที่ 1.4 การจัดกลุ่มลำดับย่อยของข้อมูลอนุกรมเวลาจำเป็นต้องมีการละทิ้งข้อมูลบางส่วน
และต้องไม่มีการซ้อนทับระหว่างลำดับย่อยใด ๆ ที่ถูกจัดกลุ่ม
จากภาพ ข้อมูลถูกจัดเป็น 2 กลุ่ม ดังที่เห็นในกรอบ และส่วนที่เหลือคือส่วนที่ไม่ถูกจัดกลุ่ม

(ที่มา: Rodpongpun, S., Niennattrakul, V. and Ratanamahatana, C. A. Selective Subsequence Time Series clustering. Knowledge-Based Systems, vol. 35, pp. 361-368, 2012.)

แต่อย่างไรก็ตามวิธีการดังกล่าวเหล่านี้จำเป็นต้องมีการกำหนดพารามิเตอร์เพื่อระบุค่าความยาวของลำดับย่อยในการจัดกลุ่มก่อนดำเนินการ ซึ่งก่อให้เกิดปัญหาสำคัญสองประการ คือ 1. เป็นการยากที่ผู้ใช้จะทราบค่าที่เหมาะสมของลำดับย่อยในการจัดกลุ่มได้ บางครั้งต้องอาศัยความรู้จากผู้เชี่ยวชาญเฉพาะด้านของข้อมูลประเภทนั้น หรือแย่ไปกว่านั้นในกรณีที่ข้อมูลมีความซับซ้อนมากๆ แม้แต่ผู้เชี่ยวชาญเองก็ไม่สามารถระบุค่าที่เหมาะสมได้ และ 2. ความยาวของลำดับย่อยในการจัดกลุ่มจะถูกจำกัดโดยค่าพารามิเตอร์ที่กำหนดลงไปนี้ ทำให้ขาดอิสระในการจัดกลุ่มที่แท้จริง เพราะ

โดยทั่วไปแล้วในข้อมูลอนุกรมเวลาหนึ่ง ๆ ไม่จำเป็นที่ลำดับย่อยในแต่ละกลุ่มจะต้องมีความยาวเท่ากัน

ทั้งสองปัญหานี้เป็นตัวการนำไปสู่ความไม่แม่นยำในการจัดกลุ่ม งานวิจัยนี้จึงนำเสนอวิธีในการแก้ไขปัญหาโดยกำจัดส่วนของการกำหนดค่าพารามิเตอร์ที่สร้างข้อจำกัดของให้กับการจัดกลุ่มทิ้งไป โดยนำเสนออัลกอริทึมใหม่ เรียกว่าการจัดกลุ่มลำดับย่อยของข้อมูลอนุกรมเวลาโดยปราศจากพารามิเตอร์ (Parameter-Free Subsequence Time Series Clustering) ซึ่งง่ายต่อการใช้งานและมีอิสระในการจัดกลุ่มครอบคลุมทุกค่าความยาว วิธีการประกอบด้วยสองส่วน คือ ส่วนของการคัดเลือกความยาวที่เหมาะสมในการจัดกลุ่ม และ ส่วนของการจัดกลุ่ม โดยในส่วนแรกได้นำหลักของการค้นพบโมทีฟความยาวเหมาะสมสำหรับข้อมูลอนุกรมเวลา (Proper Length Motif Discovery for Time Series Data) [13] โดยพื้นฐานของความสามารถในการบีบอัดข้อมูล มาประยุกต์ใช้ในการหากลุ่มตั้งต้นที่มีความยาวเหมาะสม และนำผลลัพธ์ไปใช้ต่อในส่วนที่สอง คือ การจัดกลุ่ม ซึ่งจะอธิบายถึงวิธีการโดยละเอียดต่อไปในบทที่ 3

1.2 วัตถุประสงค์ของการวิจัย

นำเสนอวิธีการจัดกลุ่มลำดับย่อยของข้อมูลอนุกรมเวลาโดยไม่ต้องมีการกำหนดพารามิเตอร์เริ่มต้นใด ๆ

1.3 ขอบเขตของการวิจัย

1. งานวิจัยนี้ทดลองกับข้อมูลอนุกรมเวลาโดยใช้ชุดข้อมูลสำหรับการจำแนกประเภทและการจัดกลุ่มจาก UCR (University of California, Riverside) [9]
2. วัดผลด้วยการเปรียบเทียบความแม่นยำกับอัลกอริทึมก่อนที่ต้องการการกำหนดค่าพารามิเตอร์เริ่มต้นโดยกำหนดค่าความยาวจริงให้กับอัลกอริทึม และเปรียบเทียบออกมาในสองกรณี คือ กรณีที่ข้อมูลประกอบด้วยลำดับย่อยความยาวเท่ากัน และ กรณีที่ข้อมูลประกอบด้วยลำดับย่อยหลากหลายความยาว

1.4 ข้อจำกัดของการวิจัย

งานวิจัยนี้เป็นงานวิจัยแรกที่นำเสนอวิธีการจัดกลุ่มลำดับย่อยของข้อมูลอนุกรมเวลาโดยปราศจากพารามิเตอร์ ดังนั้นการเปรียบเทียบผลลัพธ์จึงทำได้เพียงเปรียบเทียบกับอัลกอริทึมก่อนหน้านี้ที่ต้องการการกำหนดพารามิเตอร์เริ่มต้น โดยกำหนดพารามิเตอร์ที่ให้ผลลัพธ์ดีที่สุดของอัลกอริทึมนั้น จึงอาจเกิดผลลัพธ์ที่ไม่ยุติธรรมขึ้นต่อทั้งสองฝ่าย

1.5 ประโยชน์ที่ได้รับ

สามารถจัดกลุ่มลำดับย่อยของข้อมูลอนุกรมเวลาได้อย่างอิสระและแม่นยำ โดยไม่ต้องมีการกำหนดค่าตั้งต้นใด ๆ

1.6 วิธีดำเนินการวิจัย

1. ศึกษาขั้นตอนและวิธีการของงานวิจัยที่เกี่ยวข้อง รวมทั้งวิเคราะห์ข้อและดีข้อเสีย
2. ออกแบบอัลกอริทึมสำหรับแก้ปัญหาในงานวิจัยนี้
3. พัฒนาโปรแกรมตามอัลกอริทึมที่ได้ออกแบบ
4. ทำการทดลองและวิเคราะห์ผลการทดลอง
5. ปรับปรุงอัลกอริทึมและโปรแกรมให้ได้ผลลัพธ์ที่ดีขึ้น
6. เปรียบเทียบผลการทดลองกับงานวิจัยก่อนหน้า
7. สรุปผลการทดลองและจัดทำวิทยานิพนธ์

1.7 ลำดับขั้นตอนในการเสนอผลการวิจัย

ส่วนหนึ่งของงานวิทยานิพนธ์นี้ได้รับการตีพิมพ์เป็นบทความทางวิชาการ จำนวน 1 เรื่อง ดังนี้

“Parameter-free subsequences time series clustering with various-width clusters” โดย นวิน มาดิการ, เหมวรรณ ศิวรักษ์, สุระ รอดพงษ์พันธ์ และ โชติรัตน์ รัตนามหัทธนะ “The fifth International Conference on Knowledge and Smart Technology (KST)” ซึ่งจัดขึ้น ณ มหาวิทยาลัยบูรพา จังหวัดชลบุรี ประเทศไทย ระหว่างวันที่ 31 มกราคม ถึง 1 กุมภาพันธ์ 2556

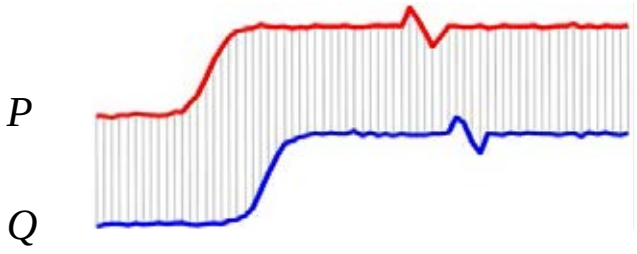
บทที่ 2
งานวิจัยที่เกี่ยวข้อง

ในส่วนนี้จะประกอบด้วยสองหัวข้อ คือ แนวคิดและทฤษฎี ซึ่งจะทำการสรุปและอธิบายทฤษฎีที่เกี่ยวข้องและได้นำมาใช้ในงานวิจัยนี้ และ งานวิจัยที่เกี่ยวข้อง ซึ่งจะอธิบายหลักการและวิธีการของแต่ละงานวิจัยก่อนหน้านี้ที่เกี่ยวข้องกับงานวิจัยนี้ ทั้งในส่วนของการจัดกลุ่มลำดับย่อยของข้อมูลอนุกรมเวลา และการค้นพบโมทีฟของข้อมูลอนุกรมเวลา

2.1 แนวคิดและทฤษฎี

Euclidean distance

ระยะทางยุคลิด เป็นมาตรวัดความแตกต่างระหว่างข้อมูลอนุกรมเวลาสองชุด โดยวัดระยะทางระหว่างจุดข้อมูลต่อจุดข้อมูลของข้อมูลอนุกรมเวลาทั้งสอง (ดังภาพที่ 2.1)



ภาพที่ 2.1 การคิดระยะทางยุคลิดระหว่างข้อมูลอนุกรมเวลา *P* และ *Q*

ซึ่งเป็นผลรวมระยะทางระหว่างจุดข้อมูลทุกจุดบนข้อมูลอนุกรมเวลา *P* และ *Q*

(ที่มา: Ratanamahatana, C. A. and Keogh, E. J. Making Time-series Classification More Accurate Using Learned Constraints. In Proceedings of the SIAM International Conference on Data Mining (SDM), pp. 11-22, 2004.)

นิยามของระยะทางยุคลิดระหว่างข้อมูลอนุกรมเวลา *P* และ *Q* คือ

$$EucDist(P,Q) = \sqrt{\sum_{k=1}^{l}(p_k - q_k)^2} \quad \text{---------- (2.1)}$$

เมื่อ $P = (p_1, p_2, p_3, \dots, p_l)$

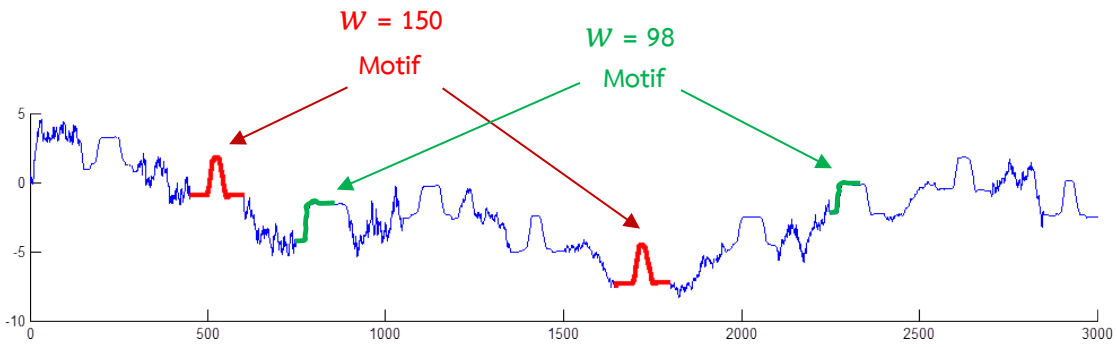
$Q = (q_1, q_2, q_3, \dots, q_l)$

k คือ ตำแหน่งของจุดข้อมูล โดยที่ k เป็นจำนวนเต็ม และ $1 \le k \le l$

และ l คือ ความยาวของข้อมูลอนุกรมเวลา P และ Q (ซึ่งต้องเท่ากัน)

Motif Discovery

โมทีฟ คือ คู่ของลำดับย่อยที่มีความคล้ายคลึงกันมากที่สุดในข้อมูลอนุกรมเวลาใด ๆ (เปรียบเทียบโดยระทางยุคลิดระหว่างลำดับย่อยทั้งสอง) โดยโมทีฟที่ได้จะแตกต่างกันออกไปตามความยาวของลำดับย่อย w ที่ถูกกำหนด (ดังภาพที่ 2.2)

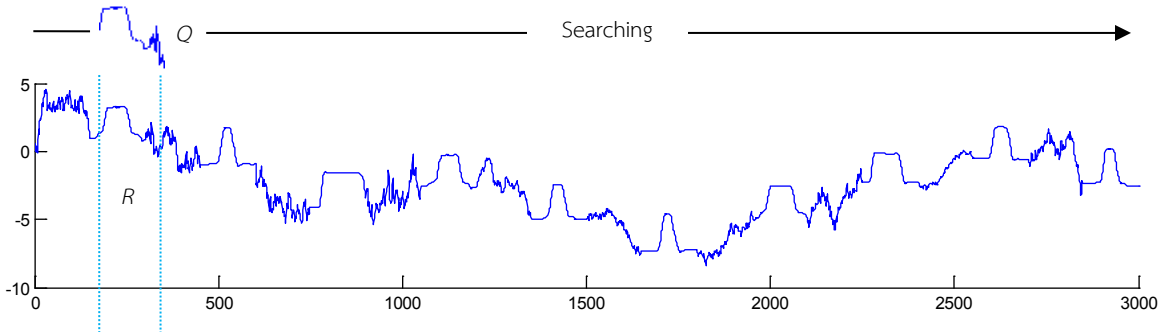


ภาพที่ 2.2 ตัวอย่างโมทีฟที่ถูกค้นพบในแต่ละความยาวที่ถูกกำหนด

วิธีการในการค้นพบโมทีฟแบบง่ายที่สุดคือวิธีบรูทฟอร์ซ (Brute force) โดยจะทำการเปรียบเทียบระยะทางยุคลิดระหว่างลำดับย่อยทั้งหมดในข้อมูลอนุกรมเวลาที่พิจารณา และให้ผลลัพธ์เป็นคู่ลำดับย่อยที่มีระยะทางยุคลิดน้อยที่สุด ซึ่งต้องการเวลาในการทำงานที่ค่อนข้างสูง แต่ ณ ปัจจุบันมีวิธีการค้นพบโมทีฟที่ได้รับการยอมรับว่าเร็วที่สุดโดยที่ให้ผลลัพธ์ถูกต้องตรงกับวิธีบรูทฟอร์ซ คือ การค้นพบโมทีฟในงานวิจัย [6] ซึ่งมีการนำคุณสมบัติความไม่เท่ากันของสามเหลี่ยม (Triangular Inequality) และเทคนิคอื่น ๆ มาใช้เพื่อลดเวลาในการทำงานให้น้อยลง อัลกอริทึมนี้มีชื่อว่า การค้นพบโมทีฟของเอ็มเค (MK Motif Discovery)

Subsequence Matching

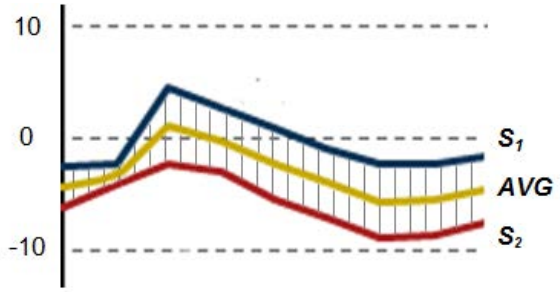
การจับคู่ลำดับย่อย คือ การค้นหาลำดับย่อยในข้อมูลอนุกรมเวลาใด ๆ ที่มีความคล้ายคลึงกับลำดับย่อยที่มีอยู่มากที่สุด (เปรียบเทียบโดยระทางยุคลิดระหว่างลำดับย่อยทั้งสอง) ซึ่งวิธีการนี้แตกต่างจากการค้นพบโมทีฟ เพราะลำดับย่อยที่จะนำไปค้นในข้อมูลอนุกรมเวลาถูกกำหนดไว้แล้วหรือ ทราบค่าอยู่แล้ว เรียกลำดับย่อยที่นำไปค้นนี้ว่า ลำดับย่อย *Q* และ ผลลัพธ์ที่ได้คือ ลำดับย่อย R (ดังภาพที่ 2.3) โดยทั่วไปแล้วอัลกอริทึมที่นิยมนำมาใช้ในการทำการจับคู่ลำดับย่อยมากที่สุด คือ One Nearest Neighbor (1NN)



ภาพที่ 2.3 ตัวอย่างการทำการจับคู่ลำดับย่อยด้วยลำดับย่อย *Q*

Amplitude Averaging

การเฉลี่ยแบบแอมพลิจูด คือ การหาค่าเฉลี่ยระหว่างจุดข้อมูลแต่ละจุดบนลำดับย่อยที่พิจารณาทั้งสอง (ดังภาพที่ 2.4)



ภาพที่ 2.4 ตัวอย่างการเฉลี่ยแบบแอมพลิจูด โดย *AVG* คือ
ลำดับย่อยซึ่งเป็นค่าเฉลี่ยระหว่างลำดับย่อย S_1 และ ลำดับย่อย S_2

โดยมีนิยามดังนี้

$$AVG_i = (S_{1_i} + S_{2_i}) / \ 2 \text{ ---------- (2.2)}$$

เมื่อ AVG คือ ลำดับย่อยผลลัพธ์

i คือ ตำแหน่งของจุดข้อมูล โดยที่ i เป็นจำนวนเต็ม และ $1 \le i \le l$

และ l คือ ความยาวของลำดับย่อย S_1 และ S_2 (ซึ่งต้องเท่ากัน)

Error of cluster

ค่าผิดพลาดของกลุ่ม คำนวณจากระยะทางยุคลิดระหว่างตัวแทนกลุ่ม (Cluster Center) คือ ลำดับย่อยที่เป็นค่าเฉลี่ยของลำดับย่อยทุกชุดภายในกลุ่มด้วยการเฉลี่ยแบบแอมพลิจูด กับลำดับย่อยทุกชุดภายในกลุ่ม

$$\tilde{E}(C_i) = \sum_{j=1}^{m} EucDist(S_j, \overline{C}_i) \text{ ---------- (2.3)}$$

เมื่อ C_i คือ กลุ่มที่ i โดยที่ i เป็นจำนวนเต็ม และ $1 \le i \le$ จำนวนกลุ่มทั้งหมด

$\overline{C}_i$ คือ ตัวแทนของกลุ่มที่ i

S_j คือ ลำดับย่อยที่ j ในกลุ่ม C_i โดยที่ j เป็นจำนวนเต็ม และ $1 \le j \le m$

และ m คือ จำนวนลำดับย่อยทั้งหมดในกลุ่ม C_i

Shannon Entropy

เอนโทรปีโดยแชนนอน ใช้สำหรับหาค่าประมาณของจำนวนบิตที่ใช้ในการแทนข้อมูลใด ๆ ในที่นี้นำมาใช้ในการแทนข้อมูลอนุกรมเวลา T โดยมีนิยามดังนี้

$$H(T) = -\sum_t P(T = t) \log_2 P(T = t) \text{ ---------- (2.4)}$$

เมื่อ T คือ ข้อมูลอนุกรมเวลาใดๆ

t คือ ค่าที่ไม่ซ้ำกันของแต่ละจุดข้อมูลในข้อมูลอนุกรมเวลา T

และ $P(T=t)$ คือ ความน่าจะเป็นที่จะเกิดค่า t ในข้อมูลอนุกรมเวลา T

โดยกำหนดให้ $P(T=t) \log_2 P(T=t)$ มีค่าเป็น 0 เมื่อ $P(T=t) = 0$

Minimum Description Length (MDL)

คือหลักการในการหาสมมติฐาน (Hypothesis) ที่มีความสามารถในการบีบอัดข้อมูลได้ดีที่สุด ตามนิยามที่ว่า ความยาวในการเก็บข้อมูล (Description Length) ที่น้อยที่สุดของข้อมูลใด ๆ บ่งบอกถึงความสามารถในการบีบอัดข้อมูลที่ดีที่สุดของสมมติฐานนั้น นิยามของความยาวในการเก็บข้อมูล *D* ด้วยสมมติฐาน *H* เป็นดังนี้

$$DL(D) = DL(H) + DL(D \mid H) \text{ ---------- (2.5)}$$

เมื่อ *D* คือ ข้อมูลใดๆ

H คือ สมมติฐานที่ใช้ในการบีบอัดข้อมูล *D*

D|H หรืออาจเขียนแทนด้วย *D'* คำนวณได้จาก *D-H*

และความยาวในการเก็บข้อมูลอนุกรมเวลาโดยทั่วไปมีนิยามดังนี้

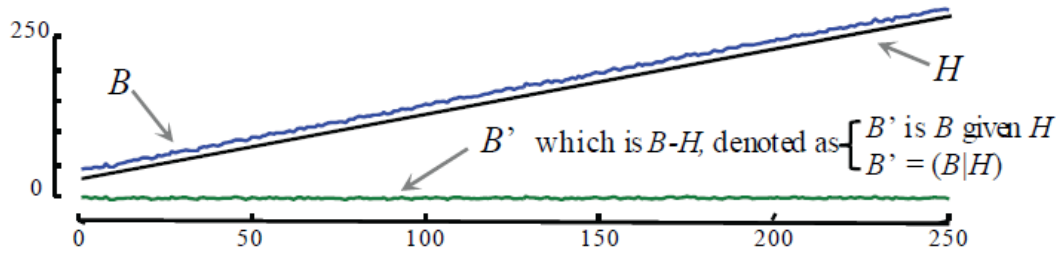
$$DL(T) = m * H(T) \text{ ---------- (2.6)}$$

เมื่อ *T* คือ ข้อมูลอนุกรมเวลาใด ๆ

m คือ ความยาวของข้อมูลอนุกรมเวลา *T*

และ *H*(*T*) คือ เอนโทรปีของข้อมูลอนุกรมเวลา *T*

ตัวอย่าง[1] กำหนดข้อมูลอนุกรมเวลา *B* ขนาด 250 จุดข้อมูล ประกอบด้วยค่าที่ไม่ซ้ำกันจำนวน 172 ค่า และ เอนโทรปีเท่ากับ 7.29 ดังนั้นจำนวนบิตที่ใช้ในการเก็บข้อมูลอนุกรมเวลา *B* คือ *DL*(*B*) = 250*7.29= 1,822 บิต



ภาพที่ 2.5 การบีบอัดข้อมูลอนุกรมเวลา *B* ด้วย *H* และ *B'*

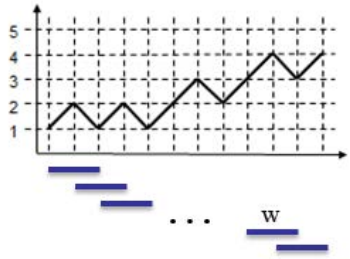
(ที่มา: Rakthanmanon, T., Keogh, E. J., Lonardi, S. and Evans, S. Time Series Epenthesis: Clustering Time Series Streams Requires Ignoring. In Proceedings of the 11th IEEE International Conference on Data Mining (ICDM), pp. 547-556, 2011.)

[1] ค่าตัวเลขทุกค่าในตัวอย่างนี้ยกมาจากค่าที่ถูกระบุไว้ในงานวิจัย [5]

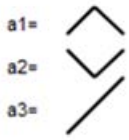
แต่ถ้ามีสมมติฐาน *H* เป็นเส้นตรงขนาด 250 จุดข้อมูลเท่ากัน เราสามารถสร้าง *B'* ขนาดเท่ากันได้โดย *B'= B-H* (ดังภาพที่ 2.5) ซึ่ง *B'* ประกอบด้วยค่าที่ไม่ซ้ำกันเพียง 10 ค่า และเอนโทรปีเท่ากับ 2.51 เราสามารถสร้าง *B* ได้ด้วย *B'* และ *H* โดยใช้จำนวนบิตเพียง *DL*(*B*) = *DL*(*H*) + *DL*(*B*|*H*) ซึ่งเท่ากับ (2*8) + (250*2.51) = 643 บิต ทั้งนี้เนื่องจาก *H* เป็นเส้นตรง จึงต้องการแค่ 2 ไบต์ (16 บิต) สำหรับเก็บค่าจุดเริ่มต้นและจุดสุดท้ายของ *H*

2.2 งานวิจัยที่เกี่ยวข้อง

การจัดกลุ่มลำดับย่อยของข้อมูลอนุกรมเวลาได้ถูกนำเสนอครั้งแรกในปี 1998 [2] โดยวิธีการคือเริ่มต้นจากการสกัดลำดับย่อยความยาว *w* ทุกชุดที่เป็นเซตย่อยของข้อมูลอนุกรมเวลานั้น ซึ่งจะทำให้ได้ลำดับย่อยความยาว *w* จำนวน *l*-*w*+1 ชุดโดยที่ *l* คือความยาวของข้อมูลอนุกรมเวลา (Time Series Length) (ดังภาพที่ 2.6) จากนั้นทำการจัดกลุ่มโดยใช้อัลกอริทึมการจัดกลุ่มแบบเคมีนส์ (*K*-means Clustering) ทำให้ได้ผลลัพธ์ดังภาพที่ 2.7



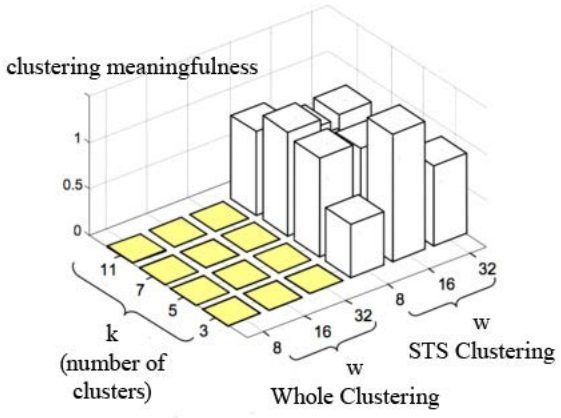
ภาพที่ 2.6 การสกัดลำดับย่อยความยาว *w* จากข้อมูลอนุกรมเวลาที่กำหนด



ภาพที่ 2.7 ผลลัพธ์จากการทำการจัดกลุ่มแบบเคมีนส์ ได้ลำดับย่อยซึ่งเป็นตัวแทนกลุ่ม 3 กลุ่ม

ซึ่งจากผลลัพธ์ที่นำเสนอ ดูเหมือนจะให้ผลลัพธ์ที่ถูกต้องและไม่มีปัญหา แต่ต่อมาเมื่อมีการศึกษาอย่างจริงจังจึงมีงานวิจัย [1] ที่ค้นพบปัญหาและแสดงผลการทดลองให้เห็นว่าการจัดกลุ่มลำดับย่อยด้วยวิธีนี้ไม่ได้ให้ผลลัพธ์ที่ถูกต้อง E. Keogh และคณะ ได้ทำการทดลองโดยเปรียบเทียบผลการทดลองระหว่างการจัดกลุ่มลำดับย่อยกับการจัดกลุ่มแบบทั้งอนุกรม โดยใช้ชุดข้อมูลเดียวกันและทำการทดลองจัดกลุ่มในแต่ละวิธี ข้อมูลที่ใช้ในการทดลองมาจากสองเซตซึ่งไม่มีความเกี่ยวข้องกัน เรียกว่าข้อมูลจากเซต *X* และ ข้อมูลจากเซต *Y* จากนั้นคำนวณค่าความมีความหมายของการจัดกลุ่ม

จากผลการทดลองที่ได้ ตามสูตร $\text{clustering meaningfulness}(X,Y)=\frac{\text{within\_set\_}X\text{\_distance}}{\text{between\_set\_}X\text{\_and\_}Y\text{\_distance}}$ (ค่านี้จะใกล้เคียง 0 ถ้าการจัดกลุ่มนั้นให้ผลลัพธ์ที่มีความหมาย ในทางกลับกันค่านี้จะใกล้เคียง 1 ถ้าการจัดกลุ่มนั้นให้ผลลัพธ์ที่ไม่มีความหมาย) ซึ่งจากผลการทดลอง (ดังภาพที่ 2.8) หมายความว่าการจัดกลุ่มลำดับย่อยของข้อมูลอนุกรมเวลาด้วยวิธีนี้ไม่ได้ให้ผลลัพธ์ที่มีความหมายโดยสิ้นเชิง ซึ่งทำให้เกิดผลกระทบต่อวงการอย่างมาก งานวิจัยหลายงานที่ได้นำอัลกอริทึมนี้ไปใช้ถูกตัดสินว่าให้ผลลัพธ์ที่ไม่ถูกต้องตามไปด้วย จึงได้มีความพยายามที่จะแก้ปัญหานี้ตามมาในหลายงานวิจัยแต่ก็ไม่มีงานใดที่ประสบความสำเร็จในการที่จะได้ผลลัพธ์ที่ดีเท่าที่ควร ปัญหานี้จึงถูกละเลยไปช่วงเวลาหนึ่ง และทำให้ความนิยมของหัวข้อวิจัยนี้ลดลงด้วย

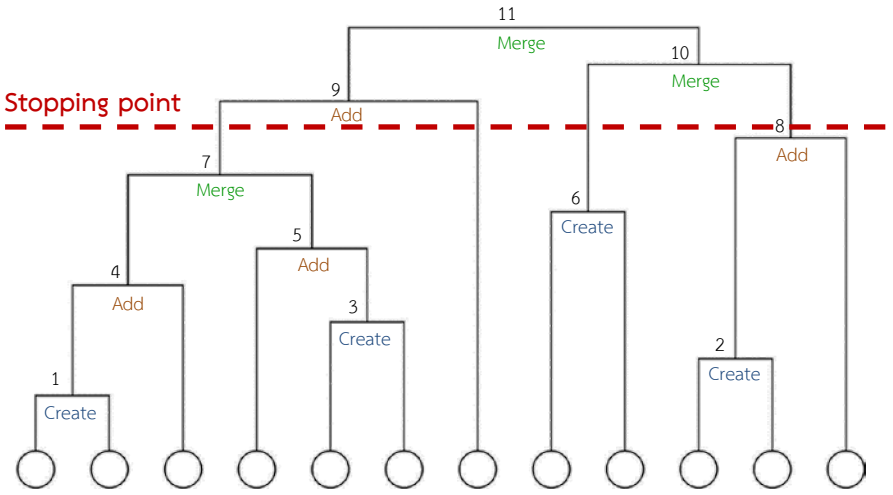


ภาพที่ 2.8 ผลลัพธ์จากการทำการทดลองจัดกลุ่มลำดับย่อยของข้อมูลอนุกรมเวลาเทียบกับการจัดกลุ่มข้อมูลอนุกรมเวลาแบบทั้งอนุกรม ในหลาย ๆ ค่า *k*

(ที่มา: Keogh, E. J., Lin, J. and Truppel, W. Clustering of Time Series Subsequences is Meaningless: Implications for Previous and Future Research. In Proceedings of the IEEE International Conference on Data Mining (ICDM), pp. 115–122, 2003.)

เมื่อไม่นานมานี้ได้มีงานวิจัยสองงาน [5][7] ที่ประสบความสำเร็จในการนำเสนอวิธีแก้ไขปัญหานี้ บนแนวคิดเดียวกันที่ว่า “ลำดับย่อยของข้อมูลอนุกรมเวลาไม่จำเป็นต้องถูกจัดกลุ่มทั้งหมด และ ต้องไม่มีการซ้อนทับกันของลำดับย่อยที่ถูกจัดกลุ่ม” ดังที่ได้กล่าวไว้แล้วก่อนหน้านี้ โดยใช้วิธีจัดกลุ่มแบบลำดับชั้น (Hierarchical Clustering) (ดังภาพที่ 2.9) ซึ่งประกอบด้วย 3 ขั้นตอนย่อย คือ สร้าง (Create) สร้างกลุ่มใหม่จากลำดับย่อยคู่หนึ่งที่คล้ายคลึงกันมากที่สุดจากกระบวนการค้นพบโมทีฟ, เพิ่ม (Add) เพิ่มลำดับย่อยที่คล้ายคลึงกับกลุ่มที่มีอยู่มากที่สุดจากกระบวนการจับคู่ลำดับย่อยเข้า

ไปในกลุ่ม และ รวม (Merge) ทำการรวมสองกลุ่มที่มีความคล้ายคลึงกันเข้าเป็นกลุ่มเดียวกัน โดยกระบวนการจัดกลุ่มจะเลือกขั้นตอนใดขั้นตอนหนึ่งในแต่ละลำดับชั้นของการจัดกลุ่มและจะดำเนินไปอย่างต่อเนื่อง จนกระทั่งถึงจุดสิ้นสุดของการจัดกลุ่ม (Stopping Point) อย่างไรก็ตามทั้งสองงานนี้มีจุดต่างกันในวิธีการเลือกขั้นตอนในการจัดกลุ่ม พารามิเตอร์ตั้งต้นที่ต้องกำหนด และ การนิยามจุดสิ้นสุดของการจัดกลุ่ม ดังนี้



ภาพที่ 2.9 ตัวอย่างการจัดกลุ่มลำดับย่อยแบบลำดับชั้น

Time Series Epenthesis: Clustering Time Series Streams Requires Ignoring [5]

ธนาวินท์ รักธรรมานนท์ และคณะ เลือกใช้หลักการของ MDL (Minimum Description Length) เป็นเกณฑ์ในการเลือกจัดกลุ่ม บนสมมติฐานที่ว่ากลุ่มที่ดีต้องมีตัวแทนของกลุ่มที่มีความสามารถในการบีบอัดลำดับย่อยที่เป็นสมาชิกในกลุ่มนั้นได้ดี ซึ่งความสามารถในการบีบอัดนี้จะขึ้นอยู่กับค่าการประหยัดบิต (Bitsave) ค่าการประหยัดบิตที่สูงหมายถึงความสามารถในการบีบอัดที่สูงตามไปด้วย (ใช้ความยาวในการเก็บข้อมูลน้อยที่สุดตามนิยามของ MDL) โดยกำหนดนิยามการคำนวณค่าการประหยัดบิตไว้ดังนี้

คำจำกัดความที่ 1 : ค่าการประหยัดบิต คือ จำนวนบิตที่ประหยัดไปจากการเก็บข้อมูลโดยใช้สมมติฐาน *H* ช่วย นิยามดังนี้

$$Bitsave = DL(\text{ก่อนหน้า}) - DL(\text{ภายหลัง}) \text{ ---------- (2.7)}$$

เมื่อ $DL(\text{ก่อนหน้า})$ คือ ความยาวในการเก็บข้อมูลตามปกติ และ $DL(\text{ภายหลัง})$ คือ ความยาวในการเก็บข้อมูลเมื่อนำสมมติฐาน *H* เข้ามาช่วย ซึ่งจะนิยามการคำนวณโดยละเอียดต่อไป

คำจำกัดความที่ 2 : ความยาวในการเก็บข้อมูลของกลุ่มใด ๆ (Description Length of Cluster) คือ ความยาวของการแทนข้อมูลทั้งกลุ่มใด ๆ ด้วยสมมติฐาน *H* มีนิยามดังนี้

$$DLC(C) = DL(H) + \sum_{A \in C} DL(A \mid H) - \max_{A \in C} DL(A \mid H) \text{ ---------- (2.8)}$$

เมื่อ *C* คือ กลุ่มใด ๆ *A* คือ ลำดับย่อยที่เป็นสมาชิกในกลุ่ม *C* และ *H* ในที่นี้คือ ตัวแทนกลุ่ม *C*

คำจำกัดความที่ 3 : ค่าการประหยัดบิตของการสร้าง คือ ค่าการประหยัดบิตที่ได้รับเมื่อทำการสร้างกลุ่มใหม่จากลำดับย่อยคู่หนึ่ง คำนวณได้ดังนี้

$$Bitsave_{creating} = DL(A) + DL(B) - DLC(C_{new}) \text{ ---------- (2.9)}$$

เมื่อ *A* และ *B* คือ ลำดับย่อยคู่ที่จะใช้สร้างกลุ่ม และ C_{new} คือ กลุ่มใหม่ที่เกิดขึ้น

คำจำกัดความที่ 4 : ค่าการประหยัดบิตของการเพิ่ม คือ ค่าการประหยัดบิตที่ได้รับเมื่อทำการเพิ่มลำดับย่อยใหม่เข้าในกลุ่มที่มีอยู่

$$Bitsave_{adding} = DL(A) + DLC(C) - DLC(C_{new}) \text{ ---------- (2.10)}$$

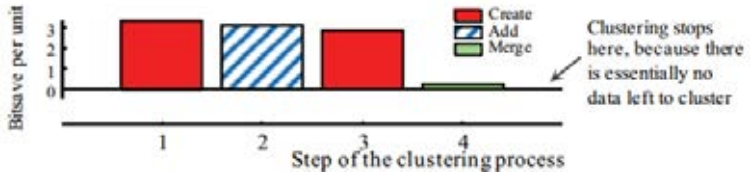
เมื่อ *A* คือ ลำดับย่อยที่จะเพิ่มเข้าในกลุ่ม *C* คือ กลุ่มเดิมก่อนที่จะทำการเพิ่ม และ C_{new} คือ กลุ่มใหม่ที่เกิดขึ้น

คำจำกัดความที่ 5 : ค่าการประหยัดบิตของการรวม คือ ค่าการประหยัดบิตที่ได้รับเมื่อทำการรวมกลุ่มสองกลุ่มที่มีอยู่เข้าด้วยกัน

$$Bitsave_{merging} = DLC(C_1) + DLC(C_2) - DLC(C_{new}) \text{ ---------- (2.11)}$$

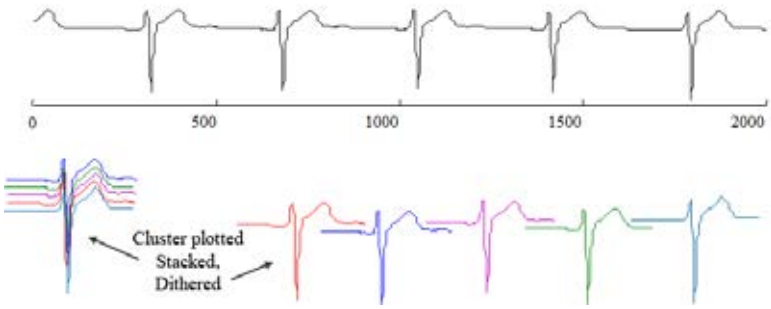
เมื่อ C_1 คือ และ C_2 คือกลุ่มสองกลุ่มที่จะถูกรวม และ C_{new} คือ กลุ่มใหม่ที่เกิดขึ้น

วิธีการคือเลือกขั้นตอนที่ให้ค่าการประหยัดบิตที่มากที่สุด และดำเนินการจัดกลุ่มเช่นนี้ต่อเนื่องไป จนกระทั่งขั้นตอนที่จะเลือกไม่สามารถช่วยประหยัดบิตได้อีก (ค่าการประหยัดบิตน้อยกว่า 0) หรือ ไม่เหลือขั้นตอนให้เลือกอีกต่อไป จึงหยุดการจัดกลุ่ม (ดังภาพที่ 2.10) ซึ่งทำให้ได้ผลลัพธ์ดังภาพที่ 2.11



ภาพที่ 2.10 แสดงแต่ละขั้นตอนของการจัดกลุ่มรวมทั้งจุดสิ้นสุดของการจัดกลุ่ม

(ที่มา: Rakthanmanon, T., Keogh, E. J., Lonardi, S. and Evans, S. Time Series Epenthesis: Clustering Time Series Streams Requires Ignoring. In Proceedings of the 11th IEEE International Conference on Data Mining (ICDM), pp. 547-556, 2011.)



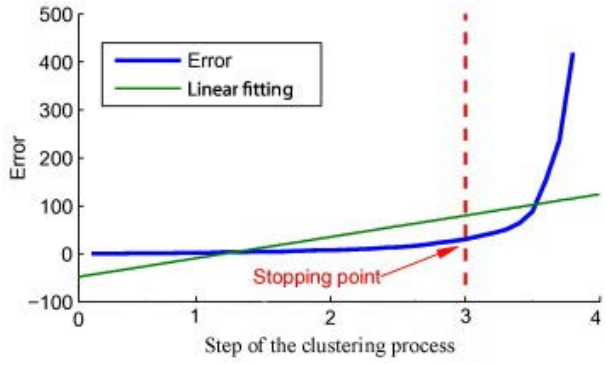
ภาพที่ 2.11 ผลทดลองของการจัดกลุ่มลำดับย่อยโดยหลักการ MDL
แสดงจำนวนกลุ่ม 1 กลุ่ม ซึ่งประกอบด้วยลำดับย่อยจำนวน 5 ชุด

(ที่มา: Rakthanmanon, T., Keogh, E. J., Lonardi, S. and Evans, S. Time Series Epenthesis: Clustering Time Series Streams Requires Ignoring. In Proceedings of the 11th IEEE International Conference on Data Mining (ICDM), pp. 547-556, 2011.)

เนื่องจากต้องป้องกันปัญหาเรื่องการถูกจำกัดความยาวของลำดับย่อย อัลกอริทึมนี้จึงมีการผ่อนผันให้ความยาวหละหลวมได้ช่วงหนึ่ง โดยพารามิเตอร์ที่ต้องการสำหรับอัลกอริทึมนี้คือ S ซึ่งเป็นความยาวของลำดับย่อยในการจัดกลุ่มโดยประมาณ ทำให้อัลกอริทึมนี้สามารถจัดการกับความยาวของลำดับย่อยได้ตั้งแต่ S ถึง $2S$

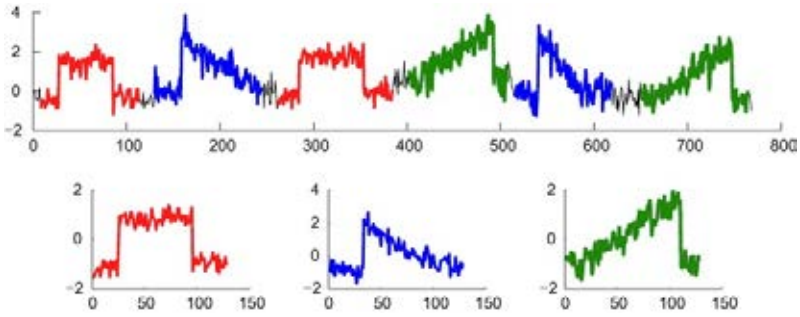
Selective Subsequence Time Series clustering [7]

สุระ รอดพงษ์พันธ์ และคณะ ใช้การคำนวณหาค่าผิดพลาดของกลุ่ม (Error of cluster) เป็นวิธีการในการเลือกขั้นตอนในการจัดกลุ่ม โดยเลือกขั้นตอนที่ให้ค่าผิดพลาดต่ำที่สุด เพราะกลุ่มที่ดีควรจะมีค่าผิดพลาดของกลุ่มต่ำ (ลำดับย่อยภายในกลุ่มมีความคล้ายคลึงกันสูง) และดำเนินการจัดกลุ่มเช่นนี้ต่อเนื่องไป จนกระทั่งเสร็จสิ้นซึ่งจะเหลือกลุ่มเพียงกลุ่มเดียว จากนั้นทำการลงจุดค่าผิดพลาดของทั้งกระบวนการลงบนกราฟแล้วหาสมการเส้นตรงของกราฟดังกล่าว จุดสิ้นสุดของการจัดกลุ่มคือจุดที่กราฟทั้งสองห่างกันมากที่สุดระหว่างจุดตัดของกราฟทั้งสอง (ดังภาพที่ 2.12) ผลลัพธ์สุดท้ายของอัลกอริทึมคือผลลัพธ์ ณ จุดสิ้นสุดการจัดกลุ่ม ดังแสดงในภาพที่ 2.13 และเช่นเดียวกับอัลกอริทึมที่ได้กล่าวถึงก่อนหน้า อัลกอริทึมนี้อนุญาตให้ความยาวของลำดับย่อยมีค่าได้ตั้งแต่ *w/f* ถึง *w*f* โดยที่ *w* คือ ความยาวของลำดับย่อย และ *f* คือ ค่าที่บ่งบอกถึงความสามารถในการยืดหยุ่นของลำดับย่อย หากไม่ต้องการให้มีความยืดหยุ่นใด ๆ ในการจัดกลุ่ม ให้กำหนดค่านี้เป็น 1 ฉะนั้นพารามิเตอร์ที่ต้องการสำหรับอัลกอริทึมนี้มี 2 ค่า คือ *w* และ *f*



ภาพที่ 2.12 กราฟแสดงค่าผิดพลาดของทั้งกระบวนการและสมการเส้นตรงของกราฟ รวมทั้งจุดสิ้นสุดของการจัดกลุ่ม

(ที่มา: Rodpongpun, S., Niennattrakul, V. and Ratanamahatana, C. A. Selective Subsequence Time Series clustering. Knowledge-Based Systems, vol. 35, pp. 361-368, 2012.)



ภาพที่ 2.13 ผลทดลองของการจัดกลุ่มลำดับย่อยโดยหลักความผิดพลาดของกลุ่ม แสดงจำนวนกลุ่ม 3 กลุ่ม ซึ่งแต่ละกลุ่มประกอบด้วยลำดับย่อยกลุ่มละ 2 ชุด และให้ผลลัพธ์ถูกต้องเมื่อเทียบกับผลลัพธ์ในภาพที่ 1.3

(ที่มา: Rodpongpun, S., Niennattrakul, V. and Ratanamahatana, C. A. Selective Subsequence Time Series clustering. Knowledge-Based Systems, vol. 35, pp. 361-368, 2012.)

ถึงแม้ผลการทดลองของทั้งสองงานได้แสดงให้เห็นว่าทั้งสองวิธีนี้สามารถให้ผลลัพธ์ที่มีความหมายสำหรับการจัดกลุ่ม แต่ปัญหาสำคัญของทั้งสองงานนี้คือ การกำหนดพารามิเตอร์เพื่อระบุความยาวของลำดับย่อยในการจัดกลุ่ม ซึ่งก่อให้เกิดความลำบากต่อการใช้งานดังที่ได้กล่าวไปแล้วก่อนหน้า และ ถึงแม้อัลกอริทึมทั้งสองได้ออกแบบวิธีจัดการกับปัญหาความยาวที่ถูกจำกัดของลำดับย่อย โดยยอมให้ค่าความยาวหละหลวมได้ช่วงหนึ่ง แต่ช่วงหนึ่งนี้เป็นเพียงช่วงเล็ก ๆ ขอบเขตของการจัดกลุ่มจึงยังถูกจำกัดอยู่ เรียกว่ายังไม่ใช่การจัดกลุ่มอย่างอิสระโดยแท้จริง

ดังนั้นแนวคิดของงานวิจัยนี้คือการกำจัดข้อจำกัดเหล่านี้ไปโดยทำให้อัลกอริทึมปราศจากการกำหนดค่าใด ๆ อาศัยข้อสังเกตที่ว่าการค้นพบโมทีฟ [6] ถูกใช้เป็นองค์ประกอบย่อยในทั้งสองงานข้างต้น จึงได้นำเสนอแนวคิดที่จะประยุกต์ใช้การค้นพบโมทีฟโดยปราศจากพารามิเตอร์ใน [4][13] กับการทำการจัดกลุ่มลำดับย่อยเพื่อให้ได้การจัดกลุ่มลำดับย่อยโดยปราศจากพารามิเตอร์ โดยจะกล่าวถึงรายละเอียดของวิธีการค้นพบโมทีฟโดยปราศจากพารามิเตอร์ในปัจจุบัน ซึ่งมีด้วยกัน 2 งานวิจัยดังนี้

Parameter-Free Motif Discovery for Time Series Data [4]

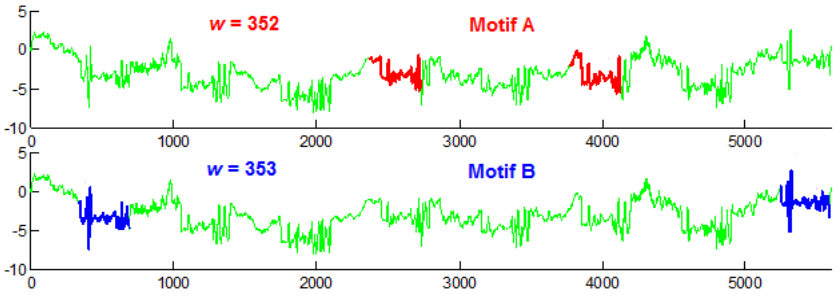
เป็นงานวิจัยงานแรกเกี่ยวกับการค้นพบโมทีฟโดยปราศจากพารามิเตอร์ โดยให้ความสำคัญกับบริเวณที่โมทีฟถูกค้นพบและความคล้ายคลึงระหว่างลำดับย่อยของโมทีฟเป็นหลัก (Location and Similarity Based) มีคำจำกัดความที่สำคัญในงานวิจัย ดังนี้

คำจำกัดความที่ 1 : โมทีฟอันดับที่ k (k^{th}-Motif) เนื่องด้วยโมทีฟตามนิยามพื้นฐานคือคู่ของลำดับย่อยที่มีความคล้ายคลึงกันมากที่สุดในข้อมูลอนุกรมเวลา โมทีฟอันดับที่ k คือ ลำดับย่อยคู่ที่มีความคล้ายคลึงกันมากที่สุดเป็นอันดับ k โดยที่ k เป็นจำนวนเต็ม มีค่าตั้งแต่ 1 ขึ้นไป และมีข้อจำกัดว่า แต่ละลำดับย่อยในโมทีฟอันดับใด ๆ จะต้องไม่มีการซ้อนทับกัน

คำจำกัดความที่ 2 : โมทีฟที่ดี (Best Motif) คือ โมทีฟที่อัลกอริทึมนี้ตัดสินว่าเป็นโมทีฟที่ดีที่สุดจากฟังก์ชันการให้คะแนนที่นำเสนอ โดยคำนึงถึงความบ่อยของการถูกค้นพบเมื่อความยาวเปลี่ยนแปลงเป็นหลัก

คำจำกัดความที่ 3 : โมทีฟที่ดีอันดับที่ k (k^{th}-Best Motif) เช่นเดียวกับโมทีฟอันดับที่ k โมทีฟที่ดีอันดับที่ k คือ โมทีฟที่อัลกอริทึมนี้ตัดสินว่าเป็นโมทีฟที่ดีเป็นอันดับที่ k จากฟังก์ชันการให้คะแนนดังกล่าว โดยที่ k เป็นจำนวนเต็ม มีค่าตั้งแต่ 2 ขึ้นไป

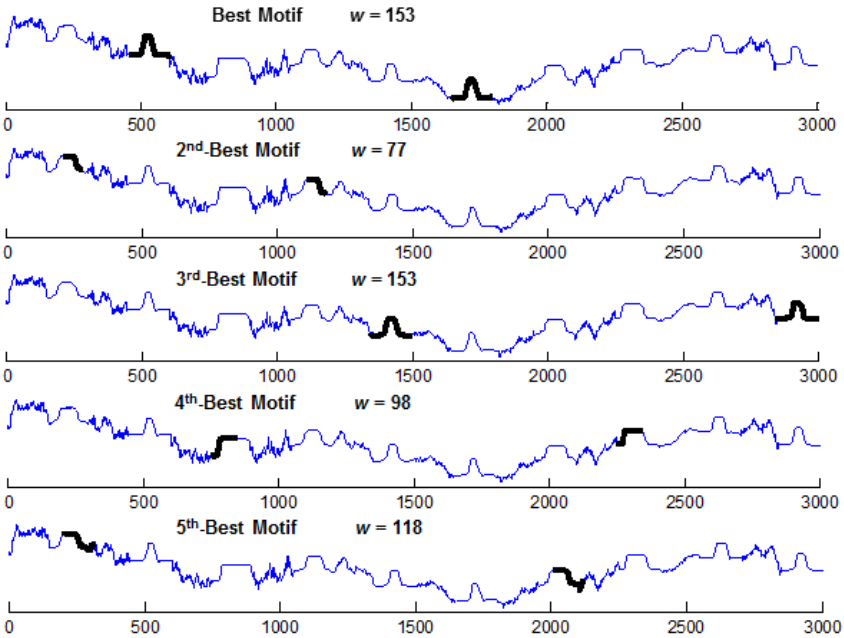
ปวัน นันทานิช และคณะ ได้นำเสนออัลกอริทึมในการค้นพบโมทีฟโดยปราศจากพารามิเตอร์เพื่อแก้ปัญหาเรื่องความอ่อนไหวของโมทีฟที่ถูกค้นพบในแต่ละค่าความยาวที่แตกต่างกัน โดยจากภาพที่ 2.14 จะเห็นว่าเพียงความยาวที่ต่างกันเพียงจุดข้อมูลเดียวสามารถทำให้ตำแหน่งของโมทีฟที่ถูกค้นพบเปลี่ยนไปเป็นอีกตำแหน่งหนึ่งได้



ภาพที่ 2.14 ความอ่อนไหวของโมทีฟที่ถูกค้นพบในความยาวที่แตกต่างกัน

(ที่มา: Nunthanid, P., Niennattrakul, V. and Ratanamahatana, C. A. Parameter-Free Motif Discovery for Time Series Data. In Proceedings of the 9th Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), pp. 1-4, 2012.

วิธีการค้นพบโมทีฟโดยปราศจากพารามิเตอร์ของงานวิจัยนี้ เริ่มต้นโดยนำเสนอการหาโมทีฟในทุกค่าความยาวที่เป็นไปได้ (ตั้งแต่ 2 ถึง ความยาวของข้อมูลอนุกรมเวลา/2) พร้อมทั้งวิธีการจัดกลุ่มและการให้คะแนนกับโมทีฟ ภายใต้นิยามที่กำหนดว่า “โมทีฟที่ดี คือ โมทีฟที่ถูกค้นพบซ้ำ ๆ ในบริเวณเดียวกันถึงแม้ความยาวเปลี่ยนไป” และให้ผลลัพธ์เป็นโมทีฟที่ถูกจัดอันดับตามเกณฑ์การให้คะแนนนี้ (ดังภาพที่ 2.15)



ภาพที่ 2.15 ผลลัพธ์ของการจัดอันดับโมทีฟในงานวิจัย [4] ประกอบด้วยโมทีฟทั้งหมด 5 อันดับ

(ที่มา: Pawan Nunthanid. Variable Length Motif Discovery for Time Series Data. Master’s Thesis, Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, 2011.)

แต่เนื่องจากงานวิจัยนี้เป็นงานวิจัยแรกที่นำเสนอวิธีการค้นพบโมทีฟโดยปราศจากพารามิเตอร์ จึงยังมีข้อบกพร่องอยู่ และปัญหาสำคัญที่เกิดขึ้น 2 ปัญหา คือ 1. ใช้เวลาตลอดทั้งกระบวนการสูงมาก เพราะ นอกจากต้องหาโมทีฟในทุกค่าความยาวที่เป็นไปได้แล้ว โมทีฟที่ต้องการในแต่ละค่าความยาวไม่ใช่เพียงคู่ของลำดับย่อยที่คล้ายคลึงกันที่สุด แต่เป็นคู่ของลำดับย่อยที่คล้ายคลึงกันในทุกอันดับ (ตามนิยามของโมทีฟอันดับที่ *k*) และ 2. มีการค้นพบว่าภายในกระบวนการมีการละทิ้งโมทีฟที่สำคัญไป ทำให้ส่งผลต่อความแม่นยำของผลลัพธ์

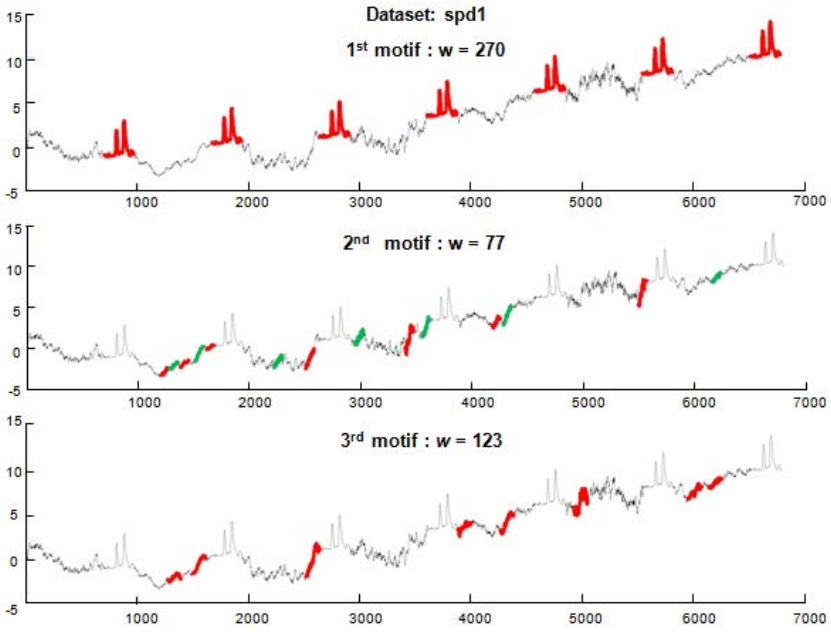
Proper Length Motif Discovery for Time Series Data using MDL Principle [13]

งานวิจัยอีกงานหนึ่งเกี่ยวกับการค้นพบโมทีฟโดยปราศจากพารามิเตอร์ โดยเปลี่ยนเทคนิคการค้นพบโมทีฟมาให้ความสำคัญกับความสามารถในการบีบอัดข้อมูล (Compression and Similarity Based) ด้วยหลักการ MDL แทน อีกทั้งยังนำเสนอโมทีฟด้วยนิยามที่แตกต่างไปจากเดิม คำจำกัดความที่สำคัญของงานวิจัย มีดังนี้

คำจำกัดความที่ 1 : โมทีฟ คือ กลุ่มของลำดับย่อยที่คล้ายคลึงกันและเกิดขึ้นบ่อยครั้งในข้อมูลอนุกรมเวลา (อย่างต่ำ 2 ครั้ง)

คำจำกัดความที่ 2 : โมทีฟอันดับที่ *k* (k^{th}-Motif) คือ โมทีฟที่มีความสามารถในการบีบอัดข้อมูลตามหลักของ MDL ได้ดีเป็นอันดับ *k* โดยที่ *k* ในที่นี้เป็นจำนวนเต็มมีค่าตั้งแต่ 1 เป็นต้นไป

เนื่องจากข้อบกพร่องของอัลกอริทึมแรกที่ได้กล่าวถึง สรชัย ยิ่งเจริญถาวรชัย ได้ศึกษาและพัฒนาอัลกอริทึมใหม่เกี่ยวกับการค้นพบโมทีฟโดยปราศจากพารามิเตอร์ ด้วยการให้นิยามของโมทีฟใหม่ที่ต่างออกไป คือ “โมทีฟเป็นกลุ่มของลำดับย่อยที่คล้ายคลึงกัน” (ไม่ใช่คู่ของลำดับย่อยดังที่ได้กล่าวถึงก่อนหน้า) (ดังภาพที่ 2.16)

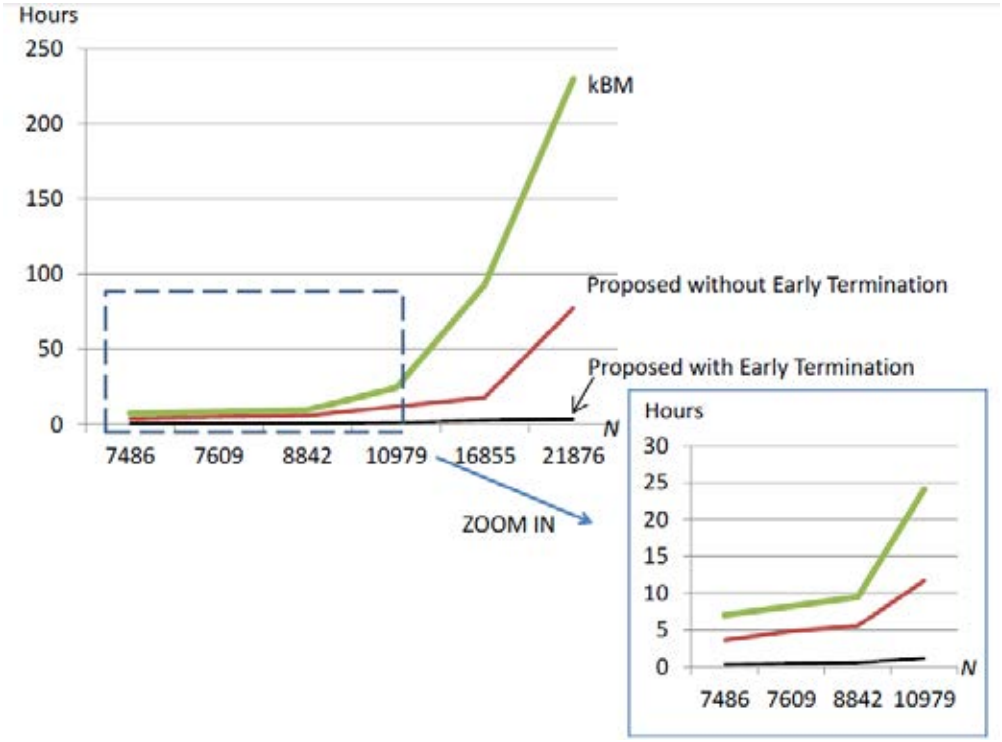


ภาพที่ 2.16 ผลลัพธ์ของการจัดอันดับโมทีฟในงานวิจัย [13]

จะเห็นว่าโมทีฟคือกลุ่มของลำดับย่อยซึ่งอาจมีได้มากกว่าหนึ่งคู่

(ที่มา: Sorrachai Yingchareonthawornchai. Proper Length Motif Discovery for Time Series Data using MDL Principle. Master's Thesis, Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, 2012.)

ผลลัพธ์ของอัลกอริทึมนี้เป็นการจัดอันดับโมทีฟเช่นเดียวกับอัลกอริทึมก่อนหน้า ต่างที่นิยามและวิธีการจัดอันดับโมทีฟ อัลกอริทึมนี้ได้นำหลักของ MDL มาใช้ในการจัดอันดับโมทีฟ คือ จัดอันดับตามความสามารถในการบีบอัดข้อมูล (ด้วยหลักการเดียวกับในงานวิจัย [5]) วิธีการนี้นำเสนอในประเด็นของความเร็วที่ใช้ในการประมวลผลซึ่งเร็วกว่าอัลกอริทึมก่อนหน้า และยังนำเสนอในส่วนของฟังก์ชัน Early Termination ซึ่งช่วยตัดเวลาของการประมวลผลข้อมูลโดยไม่จำเป็นทิ้งได้ (ดังภาพที่ 2.17) แต่ในประเด็นของความแม่นยำกลับไม่สามารถบ่งบอกได้ชัดเจนว่าแม่นยำกว่า เพราะมีการนิยามโมทีฟที่แตกต่างกัน



ภาพที่ 2.17 ผลการทดลองวัดประสิทธิภาพด้านความเร็วของอัลกอริทึม [13]
เมื่อเส้นบนคืออัลกอริทึมก่อนหน้า เส้นกลางคืออัลกอริทึมที่นำเสนอ
และเส้นล่างคืออัลกอริทึมที่นำเสนอพร้อมฟังก์ชันเสริม

(ที่มา: Sorrachai Yingchareonthawornchai. Proper Length Motif Discovery for Time Series Data using MDL Principle. Master's Thesis, Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, 2012.)

ถ้าพิจารณาในด้านของความครบถ้วนของโมทีฟผลลัพธ์และความเร็วในการประมวลผลแล้ว อัลกอริทึมที่กล่าวถึงล่าสุดนี้มีความเหมาะสมที่จะนำมาใช้ในงานวิจัยนี้มากกว่าอย่างเห็นได้ชัด แต่การจะนำผลลัพธ์ของอัลกอริทึมดังกล่าวมาใช้นั้น จำเป็นจะต้องมีการปรับเปลี่ยนนิยามของการค้นพบโมทีฟให้เหมาะสมสำหรับการนำมาใช้ในการจัดกลุ่ม นอกจากนี้ยังต้องมีการคัดเลือกเฉพาะบางโมทีฟจากโมทีฟผลลัพธ์ทั้งหมดของขั้นตอนการค้นพบโมทีฟนี้ เพื่อให้ได้เฉพาะโมทีฟที่เหมาะจะนำมาใช้เป็นกลุ่มตั้งต้นที่ดีสำหรับการจัดกลุ่มจริง ๆ ดังนั้นจึงมีความจำเป็นต้องนำเสนอวิธีการสำหรับคัดเลือกผลลัพธ์เหล่านี้ โดยจะกล่าวถึงรายละเอียดต่อไปในบทถัดไป

บทที่ 3

การจัดกลุ่มลำดับย่อยของข้อมูลอนุกรมเวลาโดยปราศจากพารามิเตอร์

ในวิทยานิพนธ์เล่มนี้นำเสนอวิธีการจัดกลุ่มลำดับย่อยข้อมูลอนุกรมเวลาโดยปราศจากพารามิเตอร์ ซึ่งเป็นงานวิจัยแรกของการจัดกลุ่มลำดับย่อยที่นำเสนอในแง่อัลกอริทึมที่ปราศจากพารามิเตอร์ โดยประยุกต์ใช้วิธีการของการค้นพบโมทีฟโดยปราศจากพารามิเตอร์ที่มีในปัจจุบันเพื่อหากลุ่มตั้งต้นที่มีความยาวเหมาะสมสำหรับการจัดกลุ่ม เพราะ อัลกอริทึมนี้ตั้งอยู่บนหลักความคิดที่ว่า การจัดกลุ่มที่ดีต้องเริ่มจากกลุ่มตั้งต้นที่ดี ซึ่งกลุ่มตั้งต้นในที่นี้ก็คือคู่ของลำดับย่อยที่มีความคล้ายคลึงกันหรือโมทีฟนั่นเอง เหนือสิ่งอื่นใดสิ่งที่จำเป็นต้องทราบเป็นอันดับแรกในบทนี้ คือ วิธีการนำเอาอัลกอริทึมในการค้นพบโมทีฟที่มีในปัจจุบันมาใช้ในงานวิจัยนี้นั้นไม่สามารถทำได้โดยตรง ทั้งนี้จำเป็นจะต้องมีการปรับปรุงและจัดการกับอัลกอริทึม โดยคำนึงถึงนิยามของโมทีฟที่จำเป็นจะต้องปรับให้ตรงกับความต้องการสำหรับการนำไปใช้งานในการจัดกลุ่ม และผลลัพธ์ของโมทีฟที่เหมาะสมจะนำไปใช้เป็นกลุ่มตั้งต้นในการจัดกลุ่ม ซึ่งจากผลลัพธ์ของอัลกอริทึมในการค้นพบโมทีฟในปัจจุบันทำเพียงจัดอันดับให้กับโมทีฟผลลัพธ์ตามนิยามของโมทีฟที่ดีในแต่ละอัลกอริทึมเท่านั้น แต่มีเพียงบางส่วนของโมทีฟผลลัพธ์เหล่านี้เท่านั้นที่มีความเหมาะสมจะนำมาใช้เป็นกลุ่มตั้งต้นสำหรับขั้นตอนการจัดกลุ่ม เพราะโมทีฟผลลัพธ์เหล่านี้อาจประกอบด้วยกลุ่มโมทีฟที่มีรูปร่างไม่สมบูรณ์ (Imperfect Motifs) ซึ่งจะก่อให้เกิดปัญหาภายหลังในการจัดกลุ่มหากไม่มีการจัดการในส่วนนี้ จึงได้นำเสนอวิธีการที่เรียกว่ากระบวนการขัดเกลาโมทีฟมาเพื่อจัดการกับปัญหานี้โดยเฉพาะ ซึ่งจะกล่าวถึงความสำคัญและรายละเอียดต่อไปในส่วนของอัลกอริทึม

ลำดับการนำเสนอในบทนี้จะแบ่งเป็น 2 หัวข้อหลัก คือ 3.1 คำจำกัดความที่ใช้ในงานวิจัย และ 3.2 อัลกอริทึมของการจัดกลุ่มลำดับย่อยของข้อมูลอนุกรมเวลาโดยปราศจากพารามิเตอร์

3.1 คำจำกัดความที่ใช้ในงานวิจัย

คำจำกัดความที่ 1 : ข้อมูลอนุกรมเวลา (Time Series) คือ เซตของจำนวนจริงที่ต่อเนื่องกันตามจุดข้อมูล ข้อมูลอนุกรมเวลาความยาว n นิยามดังนี้ $T = \{t_1, t_2, t_3, \ldots, t_n\}$

คำจำกัดความที่ 2 : ลำดับย่อย (Subsequences) ความยาว m ของข้อมูลอนุกรมเวลา T คือ เซตย่อยใด ๆ ของข้อมูลอนุกรมเวลา T ซึ่งมีความยาว m นิยามดังนี้ $S_i^m = \{t_i, t_{i+1}, t_{i+2}, \ldots, t_{i+m-1}\}$

เมื่อ i คือ ลำดับของลำดับย่อย หรือ ตำแหน่งเริ่มต้นของจุดข้อมูลในข้อมูลอนุกรมเวลา T โดยที่ $1 \le i \le n\text{-}m\text{+}1$ และ $1 < m < n$

คำจำกัดความที่ 3 : สไลดิ้งวินโดว์ (Sliding Window) ความยาว w คือ กระบวนการในการสกัดลำดับย่อยความยาว w ทั้งหมด ในข้อมูลอนุกรมเวลา T ความยาว n นิยามดังนี้

$$SlidingWindow(T,w) = \{S_1^w, S_2^w, S_3^w, \ldots, S_{n-w+1}^w\}$$

คำจำกัดความที่ 4 : ระยะทางยุคลิดโดยบรรทัดฐาน (Normalized Euclidean Distance) คือ ระยะทางยุคลิดที่ได้กล่าวไว้ในหัวข้อที่ 2.1 นำมาหารด้วย w ซึ่งเป็นความยาวของลำดับย่อยที่พิจารณาเพื่อให้เกิดบรรทัดฐานเดียวกันสำหรับระยะยุคลิดของคู่ลำดับย่อยที่พิจารณาในแต่ละความยาว นิยามดังนี้ $NormalizedEucDist(S_i^w, S_j^w) = \dfrac{EucDist(S_i^w, S_j^w)}{w}$

คำจำกัดความที่ 5 : การปรับมาตราแบบเอกรูป (Uniform Scaling) คือ การปรับให้ลำดับย่อยที่มีความยาวแตกต่างกัน มีความยาวเท่ากัน โดยขยายลำดับย่อยที่สั้นกว่าให้เท่ากับลำดับย่อยที่ยาวกว่า นิยามดังนี้ กำหนดให้ $S^w = \{s_1, s_2, s_3, \ldots, s_w\}$ คือลำดับย่อยใด ๆ ความยาว w และ $s_1, s_2, s_3, \ldots, s_w$ คือ ข้อมูลในแต่ละจุดข้อมูลของลำดับย่อยนั้น ต้องการขยายลำดับย่อยนี้ให้มีความยาว w' ซึ่งยาวกว่า จะได้ลำดับย่อยใหม่ $S^{w'} = \{s_1', s_2', s_3', \ldots, s_{w'}'\}$ โดย $s_k' = s_{\lceil k*w'/w \rceil}$ เมื่อ k คือ ตำแหน่งของจุดข้อมูลในลำดับย่อยนั้น

คำจำกัดความที่ 6 : โมทีฟ (Motif) ความยาว w คือ เซตของลำดับย่อยคู่หนึ่งที่มีความคล้ายคลึงกันมากที่สุดจากลำดับย่อยทั้งหมดที่ได้จากกระบวนการ Sliding Window ด้วยความยาว w นิยามดังนี้ $M_w = \{S_i^w, S_j^w\}$ เมื่อ $EucDist(S_i^w, S_j^w)$ มีค่าน้อยที่สุด โดยที่ไม่อนุญาตให้ S_i^w และ S_j^w มีการซ้อนทับกันใด ๆ ทั้งสิ้น

คำจำกัดความที่ 7 : ศูนย์กลางโมทีฟ (Motif Center) คือ ลำดับย่อยซึ่งเป็นค่าเฉลี่ยแบบแอมพลิจูดระหว่างลำดับย่อยทั้งสองของโมทีฟ

คำจำกัดความที่ 8 : การซ้อนทับกันทั้งหมด (Complete Overlap) สำหรับในกรณีของลำดับย่อย จะพิจารณาว่าลำดับย่อยคู่หนึ่งมีการซ้อนทับกันทั้งหมดก็ต่อเมื่อจุดข้อมูลทุกจุดในลำดับย่อยที่สั้นกว่าเป็นจุดข้อมูลบางส่วนของลำดับย่อยที่ยาวกว่า (ลำดับย่อยที่สั้นกว่าเป็นเซตย่อยของ

ลำดับย่อยที่ยาวกว่า) เช่น กำหนดให้ $w_1 \le w_2$ $S_i^{w_1}$ และ $S_j^{w_2}$ จะซ้อนทับกันทั้งหมดก็ต่อเมื่อ $S_i^{w_1} \subseteq S_j^{w_2}$

สำหรับในกรณีของโมทีฟ จะพิจารณาว่าโมทีฟคู่หนึ่งมีการซ้อนทับกันทั้งหมดก็ต่อเมื่อลำดับย่อยทั้งสองของโมทีฟหนึ่งมีการซ้อนทับกันทั้งหมดกับลำดับย่อยทั้งสองของอีกโมทีฟหนึ่ง เช่นโมทีฟ $M_{w_1} = \{S_{i_1}^{w_1}, S_{j_1}^{w_1}\}$ และ $M_{w_2} = \{S_{i_2}^{w_2}, S_{j_2}^{w_2}\}$ จะซ้อนทับกันทั้งหมดก็ต่อเมื่อ $S_{i_1}^{w_1} \subseteq S_{i_2}^{w_2}$ และ $S_{j_1}^{w_1} \subseteq S_{j_2}^{w_2}$

คำจำกัดความที่ 9 : การซ้อนทับกันบางส่วน (Partial Overlap) สำหรับในกรณีของลำดับย่อย จะพิจารณาว่าลำดับย่อยคู่หนึ่งมีการซ้อนทับกันบางส่วนก็ต่อเมื่อมีจุดข้อมูลอย่างน้อยหนึ่งจุดของลำดับย่อยคู่นี้ที่ซ้อนทับกัน โดยยกเว้นกรณีของการซ้อนทับกันทั้งหมด เช่น $S_1^3 = \{t_1, t_2, t_3\}$ และ $S_3^3 = \{t_3, t_4, t_5\}$ ซ้อนทับกันบางส่วนเพราะมีจุดข้อมูล t_3 ที่ซ้อนทับกันหนึ่งจุด

สำหรับในกรณีของโมทีฟ ให้พิจารณาลำดับย่อยทั้งสี่ชุด คือ $S_{i_1}^{w_1}, S_{j_1}^{w_1}, S_{i_2}^{w_2}, S_{j_2}^{w_2}$ เมื่อโมทีฟ $M_{w_1} = \{S_{i_1}^{w_1}, S_{j_1}^{w_1}\}$ และ $M_{w_2} = \{S_{i_2}^{w_2}, S_{j_2}^{w_2}\}$ ว่ามีจุดข้อมูลอย่างน้อยหนึ่งจุดที่ซ้อนทับกันหรือไม่

คำจำกัดความที่ 10 : ความยาวในการเก็บข้อมูลของกลุ่ม (Description Length of Cluster) ด้วยสมมติฐาน H อาจใช้ความยาวในการเก็บข้อมูลน้อยกว่าความยาวในการเก็บข้อมูลทั้งหมดตามปกติ โดยมีนิยามดังนี้ $DLC(C) = DL(H) + \sum_{A \in C} DL(A \mid H)$ เมื่อกลุ่มในที่นี้หมายถึงกลุ่มของลำดับย่อย โดย C คือ กลุ่มใด ๆ A คือ ลำดับย่อยที่เป็นสมาชิกในกลุ่ม C และ H คือสมมติฐานที่นำมาใช้ในการเก็บ C และการคำนวณ $DL(H)$ และ $DL(A \mid H)$ คำนวณจากสมการที่ 2.6 (ในหัวข้อที่ 2.1) เมื่อ $DL(A \mid H)$ คือ $DL(A - H)$

ยกตัวอย่างข้อมูลแบบไม่ต่อเนื่อง (Discrete Data) กลุ่มหนึ่งประกอบด้วยชื่อ David ที่เขียนแตกต่างกันในแต่ละภาษา $C = \{\text{'David'}, \text{'Davud'}, \text{'Dovid'}\}$ และสมมติให้แต่ละชื่อแทนลำดับย่อยแต่ละชุด กลุ่ม C นี้จะประกอบด้วย 3 ลำดับย่อย กำหนดให้สมมติฐาน H ของข้อมูลกลุ่มนี้คือลำดับย่อย David และสมมติให้การคำนวณค่าความยาวในการเก็บข้อมูลคิดเป็นรายตัวอักษร ตัวอักษรละ 8 บิต จะได้ $DL(H) = DL(\text{'David'}) = 8*5 = 40$ บิต และ $\sum_{A \in C} DL(A \mid H) = DL(\text{'David'} \mid \text{'David'}) + DL(\text{'David'} \mid \text{'Davud'}) + DL(\text{'David'} \mid \text{'Dovid'}) = DL(\text{'-'}) + DL(\text{'u'}) + DL(\text{'o'}) = 0+8+8 = 16$ บิต ดังนั้นความยาวในการเก็บข้อมูลของกลุ่ม C นี้ด้วยสมมติฐาน H= 'David' คือ 40+16 = 56 บิต

คำจำกัดความที่ 11 : ค่าการประหยัดบิต (Bitsave) เป็นมาตรวัดว่า การนำสมมติฐาน *H* มาใช้เก็บข้อมูลสามารถช่วยลดความยาวในการเก็บข้อมูลได้หรือไม่ และได้มากน้อยเพียงใด โดยมีนิยามดังนี้ *Bitsave* = *DL*(*ก่อนหน้า*) – *DL*(*ภายหลัง*) เมื่อ *DL*(*ก่อนหน้า*) คือ ความยาวในการเก็บข้อมูลทั้งหมดตามปกติ ($\sum_{A \in C} DL(A)$) และ *DL*(*ภายหลัง*) คือ ความยาวในการเก็บข้อมูลของกลุ่มด้วยสมมติฐาน *H* ตามคำจำกัดความที่ 10 และ จะตัดสินว่าช่วยลดความยาวในการเก็บข้อมูลได้ก็ต่อเมื่อค่าการประหยัดบิตมีค่ามากกว่าศูนย์ (*Bitsave* > 0)

จากตัวอย่างอย่างข้อมูลในคำจำกัดความที่แล้ว *DL*(*ก่อนหน้า*) คือ $\sum_{A \in C} DL(A) = DL(\text{'David'}) + DL(\text{'Davud'}) + DL(\text{'Dovid'})$ = 8*5+8*5+8*5 = 120 บิต ในขณะที่ *DL*(*ภายหลัง*) คำนวณให้เห็นไปแล้วว่าคือ 56 บิต ดังนั้นการนำสมมติฐาน *H*= 'David' มาใช้สำหรับข้อมูลชุดนี้สามารถประหยัดบิตได้มากกว่าจากเดิมถึง 120-56 = 64 บิต

ในอีกแง่หนึ่งสามารถใช้ค่าการประหยัดบิตเป็นมาตรวัดความคล้ายคลึงกันของลำดับย่อยได้เช่นกัน เพราะลำดับย่อยที่มีความคล้ายคลึงกันสูงจะส่งผลให้ค่าการประหยัดบิตสูงตามไปด้วย หากนำมาจัดอยู่ในกลุ่มเดียวกัน

คำจำกัดความที่ 12 : กลุ่มตั้งต้น (Initial Cluster) ในที่นี้ คือ โมทีฟทั้งหมดที่เหลือหลังจากผ่านกระบวนการขัดเกลาด้วยวิธีที่นำเสนอในงานวิจัยนี้ (ในหัวข้อที่ 3.2)

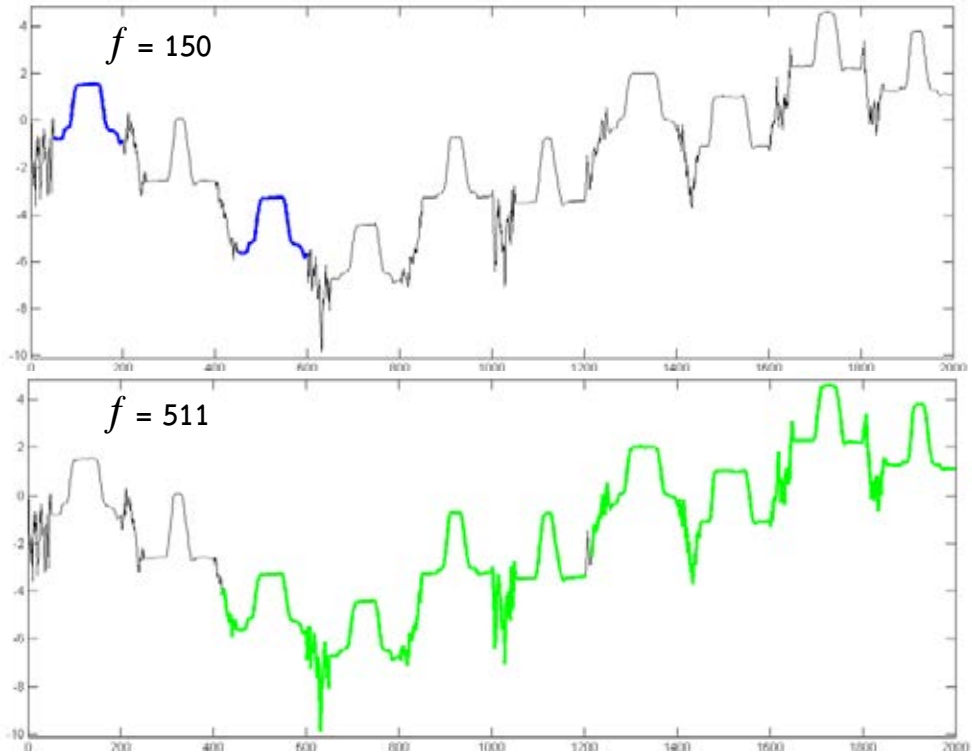
คำจำกัดความที่ 13 : กลุ่มตั้งต้นที่ดีที่สุด (Best Initial Cluster) คือ กลุ่มตั้งต้นที่ถูกจัดอันดับว่าดีที่สุดในกลุ่มตั้งต้นทั้งหมดที่มี ณ ขณะนั้น โดยจัดอันดับตามค่าการประหยัดบิตจากมากไปน้อย

3.2 อัลกอริทึมของการจัดกลุ่มลำดับย่อยของข้อมูลอนุกรมเวลาโดยปราศจากพารามิเตอร์

อัลกอริทึมแบ่งออกเป็นสองส่วนดังที่ได้กล่าวไปแล้วตอนต้น คือ ส่วนของการคัดเลือกกลุ่มตั้งต้นที่มีความยาวเหมาะสมโดยประยุกต์ใช้หลักของการค้นพบโมทีฟ และส่วนของการจัดกลุ่ม ซึ่งจะอธิบายรายละเอียดแยกจากกันในแต่ละส่วน ดังนี้

3.2.1 การคัดเลือกกลุ่มตั้งต้นที่มีความยาวเหมาะสม

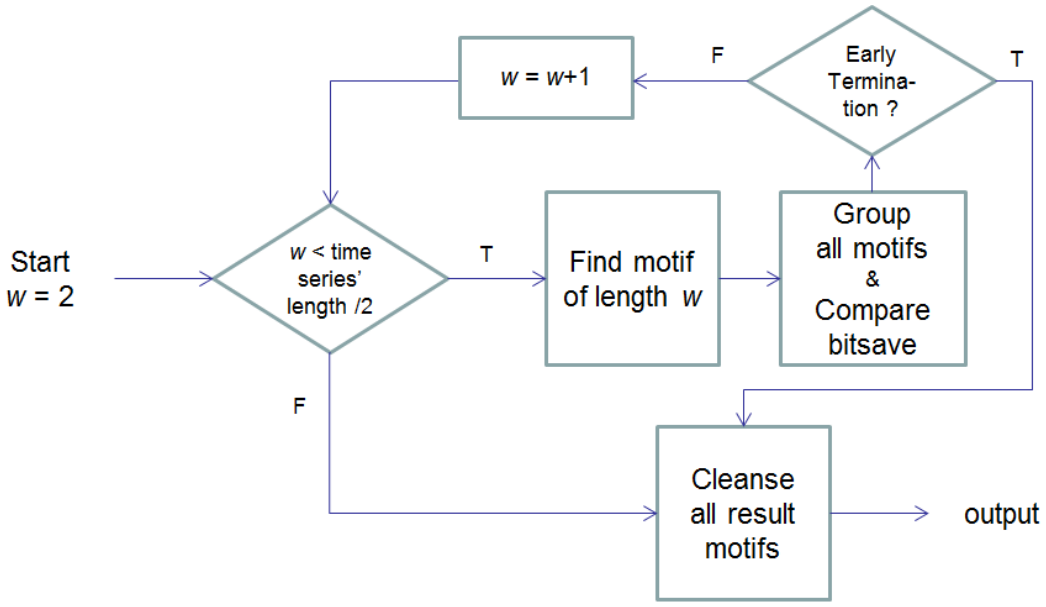
จุดประสงค์ของขั้นตอนนี้ คือ ค้นหาคู่ของลำดับย่อยที่มีความคล้ายคลึงกันและมีคุณสมบัติเหมาะสมพอที่จะทำหน้าที่เป็นกลุ่มตั้งต้นที่ดีในการจัดกลุ่ม การค้นพบโมทีฟจึงเป็นวิธีที่เหมาะสมที่สุดที่จะนำมาใช้ในกระบวนการ ทั้งนี้อัลกอริทึมในการค้นหาโมทีฟโดยปราศจากพารามิเตอร์มีเพียงสองอัลกอริทึมที่ได้กล่าวถึงในบทที่ 2 อัลกอริทึมที่นำมาใช้ในงานวิจัยนี้พยายามยึดนิยามตามงานวิจัย [4] คือ การค้นหาคู่ของลำดับย่อยที่มีความคล้ายคลึงกัน แต่เลือกใช้วิธีของงานวิจัย [13] เนื่องจากวิธีในงานวิจัย [4] ยังมีจุดบกพร่องอยู่มาก ทั้งที่ได้กล่าวถึงไปแล้วในประเด็นของเวลาในการประมวลผล และ โมทีฟบางโมทีฟที่ถูกละทิ้งไป และที่จะกล่าวถึงต่อไปนี้ คือ วิธีการจัดอันดับด้วยการนับความถี่ของโมทีฟที่ถูกค้นพบในบริเวณใกล้เคียงกันนั้นจะไม่มีความหมายเมื่อโมทีฟมีความยาวมาก ๆ (ดังภาพที่ 3.1)



ภาพที่ 3.1 ค่าความถี่จากผลการทดลองของโมทีฟผลลัพธ์ในความยาวที่แตกต่างกัน
(บน) โมทีฟความยาว 152 ที่ถูกค้นพบ พบว่ามีความถี่ของการถูกค้นพบ 150
(ล่าง) โมทีฟความยาว 785 ที่ถูกค้นพบ พบว่ามีความถี่ของการถูกค้นพบถึง 511

จากภาพแสดงผลที่ได้จากการทดลอง พบว่าเมื่อโมทีฟที่ถูกค้นพบมีความยาวมาก ๆ ค่าหนึ่งแล้ว ไม่ว่าจะขยายความยาวออกไปอีกเท่าใดโมทีฟที่ถูกค้นพบก็ยังคงเป็นที่โมทีฟในบริเวณเดิม ทำให้โมทีฟภาพล่างมีขนาดยาวกว่ามาก และความถี่ของโมทีฟก็มีค่าสูงมาก ๆ เช่นกัน เมื่อเทียบกับโมทีฟที่ควรจะเป็นจริง ๆ (ภาพบน) ดังนั้นจึงเปลี่ยนวิธีในการจัดอันดับโมทีฟมาเป็นการวัดค่าการประหยัดบิตของโมทีฟแต่ละโมทีฟแทน (ดังที่ได้กล่าวไว้ในหัวข้อที่แล้วว่าค่าการประหยัดบิตสามารถนำมาใช้เป็นมาตรวัดความคล้ายคลึงกันของคู่ลำดับย่อยได้) เพราะนอกจากจะตัดสินความคล้ายคลึงกันของโมทีฟได้ชัดเจนกว่าแล้ว เมื่อมีการขยายความยาวจนถึงค่ามาก ๆ ค่าหนึ่งแล้ว ค่าการประหยัดบิตของโมทีฟนั้นจะมีค่าเป็นลบ และค่อย ๆ เป็นลบมากขึ้นเรื่อย ๆ จึงเป็นที่มาของการนำเสนอฟังก์ชันในการหยุดประมวลผล (Early Termination) ในงานวิจัย [13] ซึ่งจะกล่าวถึงรายละเอียดต่อไป

ในส่วนของอัลกอริทึมที่นำเสนอในการหากลุ่มตั้งต้นนี้ประกอบด้วยสองขั้นตอนย่อย คือ การค้นพบโมทีฟ (Motif Discovery) ด้วยวิธีการใหม่ที่นำเสนอ และ กระบวนการขัดเกลาโมทีฟ (Cleansing Process) เพื่อให้ได้กลุ่มตั้งต้นที่ดีสำหรับนำไปใช้ต่อในอัลกอริทึมส่วนของการจัดกลุ่มต่อไป การทำงานของอัลกอริทึมโดยรวมในส่วนนี้ได้แสดงไว้เป็นผังงาน ดังภาพที่ 3.2



ภาพที่ 3.2 ผังงานการทำงานของอัลกอริทึมโดยรวมในส่วนของการคัดเลือกกลุ่มตั้งต้นที่มีความยาวเหมาะสม

- การค้นพบโมทีฟ (Motif Discovery)

เช่นเดียวกับงานวิจัยก่อน อัลกอริทึมเริ่มต้นโดยค้นหาโมทีฟในทุกค่าความยาวที่เป็นไปได้ คือ ตั้งแต่ 2 ถึง ความยาวของข้อมูลอนุกรมเวลา/2 ในแต่ละความยาวที่เปลี่ยนแปลงไปจะมีการคำนวณค่าการประหยัดบิตของโมทีฟ รวมทั้งการจัดกลุ่มโมทีฟที่ซ้อนทับกันทั้งหมดตามนิยามของการซ้อนทับกันทั้งหมด (Complete Overlap) เข้าด้วยกัน โมทีฟตั้งแต่สองโมทีฟขึ้นไปที่ซ้อนทับกันทั้งหมดจะถูกนำมาพิจารณาค่าการประหยัดบิต โมทีฟที่มีค่าการประหยัดบิตน้อยกว่าก็จะถูกละทิ้งไป แสดงการทำงานของอัลกิริทึมเป็นรหัสเทียมดังตารางที่ 3.1

ตารางที่ 3.1 อัลกอริทึมในการค้นพบโมทีฟที่นำเสนอใหม่

[*MGS*] MotifDiscovery(*T*)	
1.	*MGS* := Ø
2.	for *w* := 2 to length(*T*)/2
3.	*newmotif* := MKMotif(*T*, *w*)
4.	if CanEarlyTerminate(bitsave of *newmotif*)
5.	return *MGS*
6.	for each *motif* in *MGS*
7.	if CompleteOverlapped(*newmotif*,*motif*)
8.	if BetterBitsave(*newmotif*,*motif*)
9.	remove *motif* from *MGS*
10.	add *newmotif* into *MGS*
11.	else
12.	add *newmotif* into *MGS*
13.	merge the groups in *MGS* which contain same *motif*
14.	return *MGS*

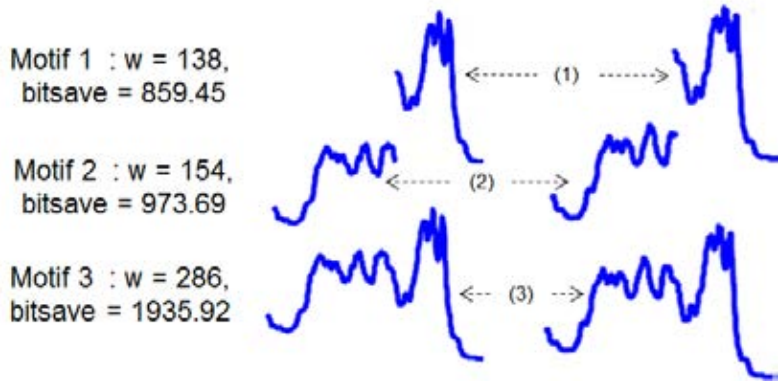
ข้อมูลนำเข้า คือ ข้อมูลอนุกรมเวลา *T* และ ข้อมูลส่งออก คือ กลุ่มของโมทีฟ *MGS* เริ่มต้นที่ค้นหาโมทีฟในทุกค่าความยาวที่เป็นไปได้ (บรรทัดที่ 2-3) โดยอัลกอริทึมที่นำมาใช้ในการค้นหาโมทีฟในที่นี้ คือ การค้นพบโมทีฟของเอ็มเค (MK Motif Discovery) [6] ซึ่งเป็นอัลกอริทึมที่ให้ผลลัพธ์ถูกต้องและเร็วที่สุดในปัจจุบันดังที่ได้กล่าวไว้ในบทที่ 2 จากนั้น (บรรทัดที่ 6-12) นำโมทีฟที่เพิ่งค้นพบมาตรวจสอบกับโมทีฟภายในกลุ่ม ณ ขณะนั้นว่ามีการซ้อนทับกันทั้งหมดหรือไม่ หากมีการซ้อนทับกันทั้งหมดให้ทำการเปรียบเทียบค่าการประหยัดบิต และ ละทิ้งโมทีฟที่มีค่าการประหยัดบิตน้อยกว่าไป

โดยวิธีการคำนวณค่าการประหยัดบิตของแต่ละโมทีฟทำได้ดังนี้ กำหนดให้ โมทีฟ $M_w = \{S_x^w, S_y^w\}$ ค่าการประหยัดบิตของโมทีฟนี้ คือ ค่าความยาวของกลุ่มโดยสมมติให้โมทีฟเปรียบเสมือนกลุ่มของลำดับย่อยซึ่งมีสมาชิกเป็นลำดับย่อยสองชุด สมมติฐาน H ซึ่งนำมาแทนคือค่าเฉลี่ยแบบแอมพลิจูดของลำดับย่อยทั้งสองนี้ เปรียบเทียบกับ ค่าความยาวในการเก็บลำดับย่อยทั้งสองชุดตามปกติตามสมการ

จาก $Bitsave = DL(\text{ก่อนหน้า}) - DL(\text{ภายหลัง})$

จะได้ $$Bitsave = \sum_{S \in M} DL(S) - (DL(H) + \sum_{S \in M} DL(S \mid H)) \text{ ---------- (3.1)}$$

และขั้นตอนสุดท้ายในแต่ละรอบ (บรรทัดที่ 13) เป็นวิธีการเพื่อจัดการกับโมทีฟที่ซ้อนทับกับโมทีฟภายในกลุ่มมากกว่าหนึ่งโมทีฟขึ้นไป (ดังภาพที่ 3.3) หากปราศจากขั้นตอนนี้ภายในกลุ่มจะประกอบด้วยโมทีฟเดียวกันซ้ำ ๆ หลายจำนวน ทั้งนี้การจะเลือกเก็บโมทีฟที่ใหญ่กว่าเพียงโมทีฟเดียวต้องมั่นใจว่าค่าการประหยัดบิตของโมทีฟนั้นมากกว่าโมทีฟที่เล็กกว่าที่ซ้อนทับกันทั้งหมดทุกโมทีฟ ไม่เช่นนั้นจะละทิ้งโมทีฟขนาดใหญ่กว่านี้ทิ้งไป



ภาพที่ 3.3 แสดงโมทีฟที่ซ้อนทับกันทั้งหมด (เส้นประเชื่อมระหว่างลำดับย่อยทั้งสองของโมทีฟ)

จะเห็นว่าโมทีฟ 1 และ โมทีฟ 2 ไม่ได้ซ้อนทับกันทั้งหมด (พิจารณาในแนวตั้ง) แต่เมื่อมีการค้นพบโมทีฟ 3 ซึ่งซ้อนทับกันทั้งหมดกับทั้งโมทีฟ 1 และ โมทีฟ 2 อีกทั้งยังมีค่าการประหยัดบิตที่ดีกว่าหากไม่มีการจัดการใด ๆ โมทีฟ 3 จะถูกบันทึกไว้สองครั้ง ทั้ง ๆ ที่เป็นโมทีฟเดียวกัน

ในส่วนของบรรทัดที่ 4 ที่อธิบายข้ามไป คือ ฟังก์ชันการในการหยุดประมวลผล (Early Termination) ดังที่ได้กล่าวไปแล้วว่าเมื่อขยายความยาวในการค้นหาโมทีฟจนถึงความยาวมาก ๆ ค่าหนึ่ง ค่าการ

ประหยัดบิตจะเป็นลบ หมายความว่า ลำดับย่อยทั้งสองนี้ไม่มีความคล้ายคลึงกันอีกต่อไป และยิ่งทำการขยายความยาวให้มากขึ้นไปอีก ค่าการประหยัดบิตนี้ก็จะยิ่งเป็นลบมากขึ้นเรื่อย ๆ จึงไม่มีประโยชน์ที่จะประมวลผลต่อไป ถ้าการทำงานของโปรแกรมเข้าสู่กรณีนี้จะทำการหยุดประมวลผลและนำผลลัพธ์ ณ ขณะนั้นไปใช้ต่อในขั้นถัดไปได้ทันที เพราะแม้จะประมวลผลต่อไปก็ไม่เกิดการเปลี่ยนแปลงกับผลลัพธ์นั้นอีกแต่อย่างใด

- การขัดเกลาโมทีฟ (Cleansing Process)

หลังจากที่ได้กลุ่มของโมทีฟที่เป็นผลลัพธ์ในขั้นตอนก่อนหน้า ขั้นตอนต่อไปคือการคัดเลือกโมทีฟที่จะทำหน้าที่เป็นกลุ่มตั้งต้นต่อไป โดยผ่านขั้นตอนการขัดเกลาโมทีฟที่จะนำเสนอต่อไปนี้เพราะโมทีฟที่เป็นผลลัพธ์เหล่านี้อาจประกอบด้วยกลุ่มโมทีฟมีรูปร่างไม่สมบูรณ์ (ดังภาพที่ 3.4) ที่ได้กล่าวถึงในตอนต้นของบทนี้ การขัดเกลาโมทีฟทำเพื่อละทิ้งโมทีฟที่ไม่สมบูรณ์เหล่านี้ไป เพราะโมทีฟเหล่านี้อาจส่งผลให้เกิดความคลาดเคลื่อนในการจัดกลุ่มต่อไปได้

ภาพที่ 3.4 ตัวอย่างโมทีฟที่ไม่สมบูรณ์ (บน) ข้อมูลอนุกรมเวลานำเข้า
(ล่าง) โมทีฟผลลัพธ์จากขั้นตอนการค้นพบโมทีฟที่นำเสนอ
ในที่นี้โมทีฟที่ถูกจัดอยู่อันดับที่ 2 คือ โมทีฟที่ไม่สมบูรณ์

จากภาพจะเห็นว่าโมทีฟทั้งสองอันดับไม่ได้ซ้อนทับกันทั้งหมด ดังนั้นผลลัพธ์จึงออกมาแยกกันทั้ง ๆ ที่เมื่อดูตามรูปร่างแล้ว เป็นโมทีฟรูปเดียวกัน และโมทีฟที่ถูกจัดอันดับอยู่ในอันดับสองในที่นี้เป็นโมทีฟที่ยังมีรูปร่างไม่สมบูรณ์ ซึ่งโดยทั่วไปแล้วโมทีฟที่มีความยาวสั้นกว่าจะมี ระยะทางยุคลิดโดยบรรทัดฐาน (Normalized Euclidean Distance) น้อยกว่าอยู่แล้ว หากละเลยการกำจัดโมทีฟที่ไม่สมบูรณ์นี้ไปจะส่งผลถึงขั้นตอนในการจัดกลุ่ม เพราะการจัดกลุ่มในงานวิจัยนี้เลือกใช้ระยะทางยุคลิดโดยบรรทัดฐานเป็นตัวกำหนดทางเลือกของขั้นตอนในการจัดกลุ่ม โดยเลือกขั้นตอนที่ให้ระยะทางยุคลิดโดยบรรทัดฐานน้อยที่สุด ดังนั้นหลังจากที่สร้างกลุ่มตั้งต้นแรกจากโมทีฟอันดับแรกในที่นี้ไป แทนที่ขั้นต่อไปจะนำลำดับย่อยที่รูปร่างเหมือนกับกลุ่มตั้งต้นแรกเพิ่มเข้าไปในกลุ่ม อาจเลือกที่จะสร้างกลุ่มตั้งต้นใหม่จากโมทีฟอันดับสองแทน เพราะมีค่าระยะทางยุคลิดโดยบรรทัดฐานที่ต่ำกว่า ทำให้การจัดกลุ่มสูญเสียรูปร่างที่ควรจะเป็น และส่งผลต่อความแม่นยำของการจัดกลุ่ม ซึ่งโดยทั่วไปแล้วผลลัพธ์จากการค้นพบโมทีฟจะประกอบด้วยผลลัพธ์ที่ไม่สมบูรณ์เช่นนี้เป็นจำนวนมาก เพราะฉะนั้นจึงต้องมีการกำจัดโมทีฟที่ไม่สมบูรณ์เหล่านี้ทิ้งไปเสียก่อน โดยวิธีการที่นำเสนอประกอบด้วยขั้นตอนดังนี้

1. จากโมทีฟที่เป็นผลลัพธ์ ให้นำศูนย์กลางของโมทีฟไปค้นหาลำดับย่อยที่มีความคล้ายคลึงกันในข้อมูลอนุกรมเวลาที่สนใจ โดยการคำนวณระยะทางยุคลิดกับทุกลำดับย่อย ยกเว้นลำดับย่อยที่เป็นองค์ประกอบของโมทีฟนี้

2. สร้างกลุ่มจำลองโดยการนำลำดับย่อยที่มีความคล้ายคลึงกันมาลองเพิ่มเข้าไปในกลุ่ม โดยเริ่มจากลำดับย่อยที่มีความคล้ายคลึงสูงก่อน และจะทำการเพิ่มเข้าไปในกลุ่มจำลองนี้เมื่อคำนวณค่าการประหยัดบิตแล้วได้ค่ามากกว่าศูนย์ โดยมีวิธีการคำนวณดังนี้

จาก $Bitsave = DL(\text{ก่อนหน้า}) - DL(\text{ภายหลัง})$ จะได้

$$Bitsave = [\sum_{S \in G} DL(S) - (DL(H) + \sum_{S \in G} DL(S \mid H))] - [\sum_{S \in G'} DL(S) - (DL(H) + \sum_{S \in G'} DL(S \mid H))] \quad (3.2)$$

ในที่นี้ H คือ ศูนย์กลางโมทีฟซึ่งคงที่ G คือ กลุ่มจำลองในขั้นตอนก่อนหน้า และ G' คือ กลุ่มจำลองภายหลังการเพิ่มลำดับย่อยอีกหนึ่งชุดเข้ามาในกลุ่ม ดังนั้น $\sum_{S \in G'} DL(S) = DL(S') + \sum_{S \in G} DL(S)$ และ

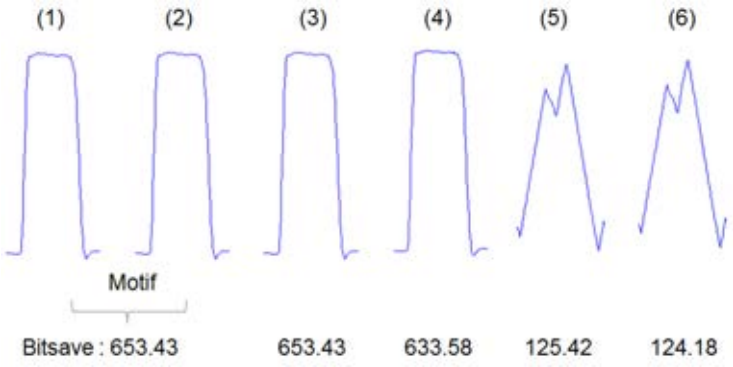
$\sum_{S \in G'} DL(S \mid H) = DL(S' \mid H) + \sum_{S \in G} DL(S \mid H)$ เมื่อ S' คือ ลำดับย่อยใหม่ที่จะนำมาเพิ่มเข้าในกลุ่ม

แทนในสมการ (3.2) $Bitsave = [\sum_{S \in G} DL(S) - (DL(H) + \sum_{S \in G} DL(S \mid H))] -$

$[DL(S') + \sum_{S \in G} DL(S) - (DL(H) + DL(S' \mid H) + \sum_{S \in G} DL(S \mid H))]$

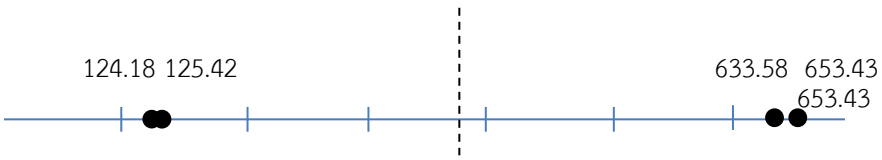
จะได้ $Bitsave = DL(S') - DL(S' \mid H)$ ---------- (3.3)

จากตัวอย่างในภาพที่ 3.4 เมื่อนำโมทีฟอันดับหนึ่งมาค้นหาลำดับย่อยที่คล้ายคลึงกันและคำนวณค่าการประหยัดบิตเสร็จสิ้นทุกขั้นตอนแล้วจะได้ผลลัพธ์ดังภาพที่ 3.5



ภาพที่ 3.5 ผลลัพธ์ของการสร้างกลุ่มจำลองด้วยโมทีฟอันดับแรกจากตัวอย่างในภาพที่ 3.4

3. จากกลุ่มจำลองในผลลัพธ์ของขั้นที่สอง จะเห็นว่ารูปร่างของลำดับย่อยเริ่มต่างกันที่ลำดับย่อยชุดที่ 5 เป็นต้น ซึ่งเมื่อสังเกตค่าการประหยัดบิตแล้วพบว่า มีความแตกต่างกันมากระหว่างลำดับย่อยชุดที่ 1 ถึง 4 และ ลำดับย่อยชุดที่ 5 ถึง 6 กระบวนการในขั้นตอนนี้จึงทำเพื่อหาเส้นแบ่งระหว่างข้อมูลสองกลุ่มนี้ โดยมีวิธีการดังนี้ เริ่มต้นที่ลงจุดค่าการประหยัดบิตลงบนเส้นตรงเส้นหนึ่ง (ดังภาพที่ 3.6)



ภาพที่ 3.6 เส้นตรงแสดงค่าการประหยัดบิตของแต่ละลำดับย่อยและโมทีฟ

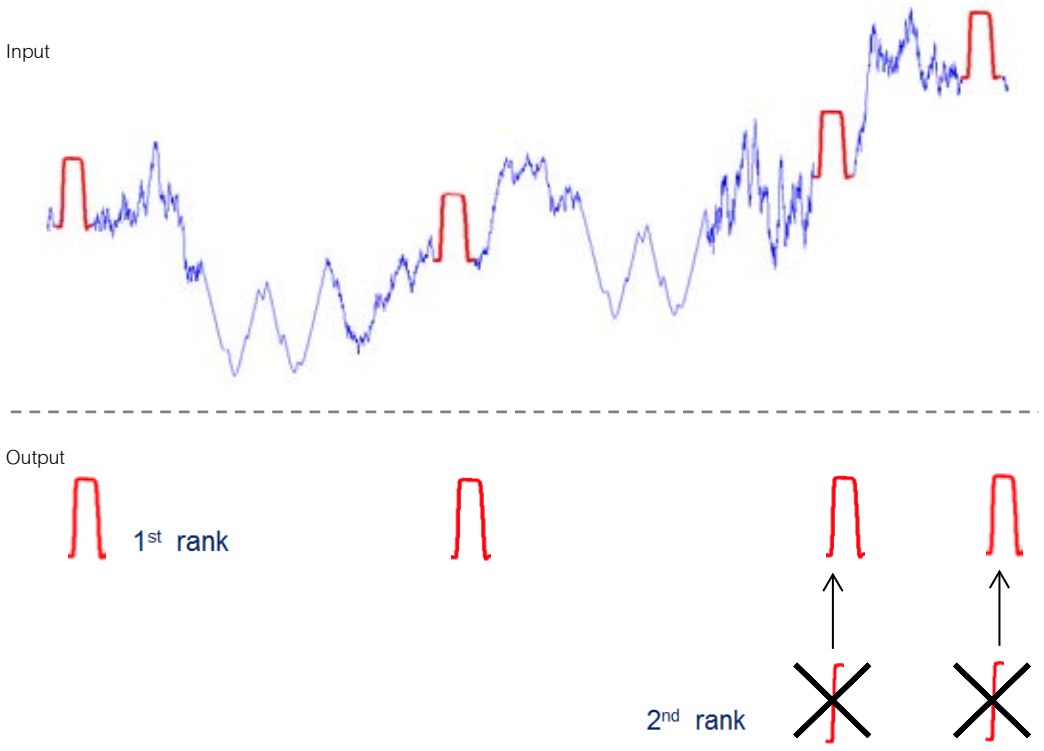
จากนั้นให้หาเส้นแบ่งที่สามารถแบ่งแยกข้อมูลออกเป็นสองกลุ่มได้ดีที่สุด โดยคำนวณช่องว่างระหว่างข้อมูลสองกลุ่ม (Gap) ดังนี้

$$\text{Gap} = \mu_R - \sigma_R - (\mu_L + \sigma_L) \text{---------- (3.4)}$$

เมื่อ μ_R และ σ_R แทนค่าเฉลี่ย และ ส่วนเบี่ยงเบนมาตรฐานของข้อมูลด้านขวา และ μ_L และ σ_L แทนค่าเฉลี่ย และ ส่วนเบี่ยงเบนมาตรฐานของข้อมูลด้านซ้าย ตามลำดับ

จากนั้นเมื่อสามารถแบ่งค่าข้อมูลเป็นสองฝั่งได้แล้วให้ตัดข้อมูลในฝั่งซ้ายทิ้งไป ผลลัพธ์ที่เหลือในกลุ่มจำลองจะเหลือเพียงข้อมูลที่รูปร่างคล้ายกันเท่านั้น (หมายเหตุ : ในการลงจุดค่าการประหยัดบิต เพื่อป้องกันเหตุการณ์ที่ในกลุ่มจำลองประกอบด้วยลำดับย่อยที่สมบูรณ์อยู่แล้ว ให้ลงจุดค่า 1 เพิ่มไปบนเส้นตรงเพื่อให้เกิดความแตกต่างระหว่างข้อมูล ซึ่งการใส่ค่า 1 เพิ่มไปนี้จะไม่ส่งผลต่อผลลัพธ์ในกรณีตัวอย่าง เพราะลำดับย่อยที่รูปร่างแตกต่างจะมีค่าการประหยัดบิตที่น้อยอยู่แล้ว)

4. หลังจากได้กลุ่มจำลองที่สมบูรณ์ในขั้นที่ 3 แล้ว ขั้นต่อไปนี้คือการกำจัดโมทีฟในอันดับต่ำกว่าซึ่งอาจเป็นองค์ประกอบของกลุ่มจำลองนี้ออก (ดังภาพที่ 3.7)

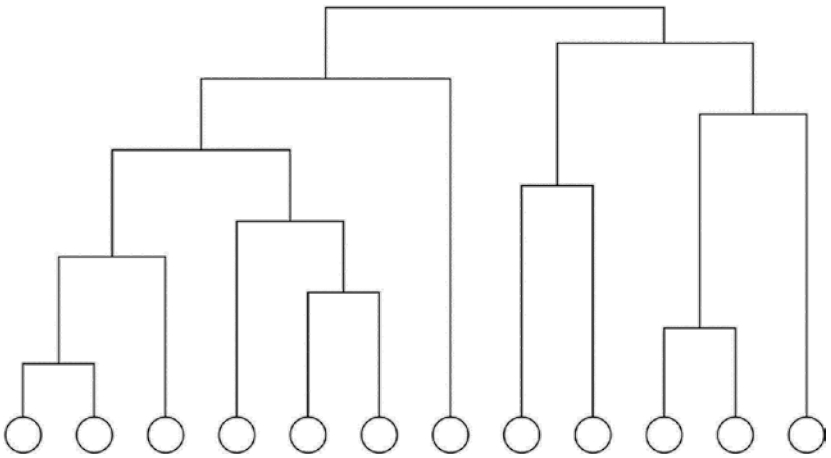


ภาพที่ 3.7 โมทีฟอันดับต่ำกว่าซึ่งเป็นองค์ประกอบของโมทีฟอันดับสูงกว่าจะถูกกำจัดไป

เมื่อกำหนดความสำคัญของโมทีฟที่มีค่าการประหยัดบิตสูงกว่าให้มากกว่าแล้ว โมทีฟที่มีความสำคัญน้อยกว่าซึ่งอาจมีขนาดเล็กกว่า หรือ ใหญ่กว่า หากพบว่ามีการซ้อนทับกันทั้งหมดของลำดับย่อยในกลุ่มจำลองที่สร้างขึ้นนี้ จะถือว่าเป็นโมทีฟที่มีรูปร่างเดียวกัน และโมทีฟที่มีความสำคัญน้อยกว่าเหล่านี้จะถูกกำจัดไป

3.2.2 การจัดกลุ่มลำดับย่อย

หลังจากที่ได้กลุ่มตั้งต้นซึ่งเป็นผลลัพธ์ของขั้นตอนที่แล้วเรียบร้อยแล้ว ขั้นตอนถัดไปคือการดำเนินการจัดกลุ่ม โดยใช้การจัดกลุ่มแบบลำดับชั้นเหมือนกับในงานวิจัยก่อน ๆ และเพื่อหลีกเลี่ยงปัญหาในการสร้างผลลัพธ์รูปคลื่นไซน์ จึงจำเป็นต้องมีข้อจำกัดในการจัดกลุ่ม ดังที่ได้กล่าวไปในตอนต้นคือ ลำดับย่อยทั้งหมดไม่จำเป็นต้องถูกจัดกลุ่ม และ ลำดับย่อยที่ถูกจัดกลุ่มนั้นล้วนต้องไม่มีส่วนซ้อนทับกันใด ๆ ทั้งสิ้น (ไม่ว่าซ้อนทับกันบางส่วน หรือ ซ้อนทับกันทั้งหมด) การจัดกลุ่มมีลักษณะเป็นแบบค่อยเป็นค่อยไปโดยคำนึงถึงค่าผิดพลาดกลุ่มเป็นหลัก โดยพยายามจัดกลุ่มให้ได้ค่าผิดพลาดของกลุ่มน้อยที่สุดในแต่ละขั้น และจะดำเนินการจัดกลุ่มอย่างนี้ต่อเนื่องไปจนถึงจุดสุดท้าย (ดังภาพที่ 3.8) ซึ่งมีลักษณะเป็นกรีดีอัลกอริทึม (Greedy Algorithm) และไม่ได้รับประกันว่าจะให้ค่าผิดพลาดรวมในจุดสุดท้ายน้อยที่สุด เมื่อเทียบกับบรูทฟอร์ซอัลกอริทึม (Brute Force Algorithm) แต่ให้ผลลัพธ์ที่ใกล้เคียงกันในเวลาการประมวลผลที่น้อยกว่ามาก จึงเลือกใช้กรีดีอัลกอริทึมสำหรับงานวิจัยนี้ จากนั้นเมื่อการจัดกลุ่มสิ้นสุดลงจึงมาทำการวิเคราะห์หาจุดสิ้นสุดการจัดกลุ่ม (Stopping Point) ที่เหมาะสมในภายหลัง ซึ่งในงานวิจัยนี้นำวิธีของงานวิจัย [7] มาใช้โดยตรง



ภาพที่ 3.8 การจัดกลุ่มลำดับย่อยแบบลำดับชั้น โดยแทนแต่ละลำดับย่อยด้วยรูปวงกลม

การนำเสนอในหัวข้อนี้จึงประกอบไปด้วยสองส่วน คือ ส่วนของอัลกอริทึมในการจัดกลุ่ม (Clustering Algorithm) และ เกณฑ์ในการหาจุดสิ้นสุดของการจัดกลุ่ม (Stopping Criterion)

- อัลกอริทึมในการจัดกลุ่ม (Clustering Algorithm)

อัลกอริทึมในการจัดกลุ่มลำดับย่อยที่นำเสนอนี้ประกอบด้วยการวนซ้ำของการเลือกแต่ละขั้นตอนในการจัดกลุ่ม โดยในแต่ละรอบของการวนซ้ำมีขั้นตอนให้เลือก 4 ขั้นตอน ประกอบด้วย

1. สร้างกลุ่มตั้งต้น (Create Initial Cluster) คือ การสร้างกลุ่มของลำดับย่อยจากกลุ่มตั้งต้นที่ดีที่สุด (Best Initial Cluster) ณ ขณะนั้น

2. สร้างกลุ่มถัดไป (Create Subsequent Cluster) คือ การสร้างกลุ่มของลำดับย่อยจากการค้นพบโมทีฟในความยาวเดียวกันกับกลุ่มที่ถูกสร้างไว้จากขั้นตอนแรก หรืออาจกล่าวได้ว่าเป็นการค้นพบโมทีฟอันดับถัดไปของโมทีฟในกลุ่มตั้งต้นแรก

3. เพิ่มลำดับย่อย (Add) คือ การเพิ่มลำดับย่อยที่มีความคล้ายคลึงกันมากที่สุดเข้าไปในกลุ่มที่ถูกสร้างไว้แล้ว โดยการเลือกลำดับย่อยมาเพิ่มในกลุ่มนั้นใช้เทคนิคการจับคู่ลำดับย่อยโดยจับคู่กับตัวแทนกลุ่ม

4. รวมกลุ่ม (Merge) คือ การรวมกลุ่มที่ถูกสร้างไว้สองกลุ่มเข้าด้วยกัน

โดยให้ทำการเลือกขั้นตอนที่ให้ค่าผิดพลาดของกลุ่มต่อความยาว (Error of Cluster per Width) ที่เปลี่ยนแปลงน้อยที่สุด ซึ่งมีวิธีการคำนวณดังนี้

กำหนดให้ ค่าผิดพลาดของกลุ่มต่อความยาวทั้งหมด

$$EPW(C_i) = \sum_{j=1}^{m} NormalizedEucDist(S_j, \overline{C}_i) \quad \text{---------- (3.5)}$$

เมื่อ C_i คือ กลุ่มที่ *i* โดยที่ *i* เป็นจำนวนเต็ม และ $1 \le i \le$ จำนวนกลุ่มทั้งหมด

$\overline{C}_i$ คือ ตัวแทนของกลุ่มที่ *i*

S_j คือ ลำดับย่อยที่ *j* ในกลุ่ม C_i โดยที่ *j* เป็นจำนวนเต็ม และ $1 \le j \le m$

และ *m* คือ จำนวนลำดับย่อยทั้งหมดในกลุ่ม C_i

ดังนั้น ค่าผิดพลาดของกลุ่มต่อความยาวที่เปลี่ยนแปลง คือ

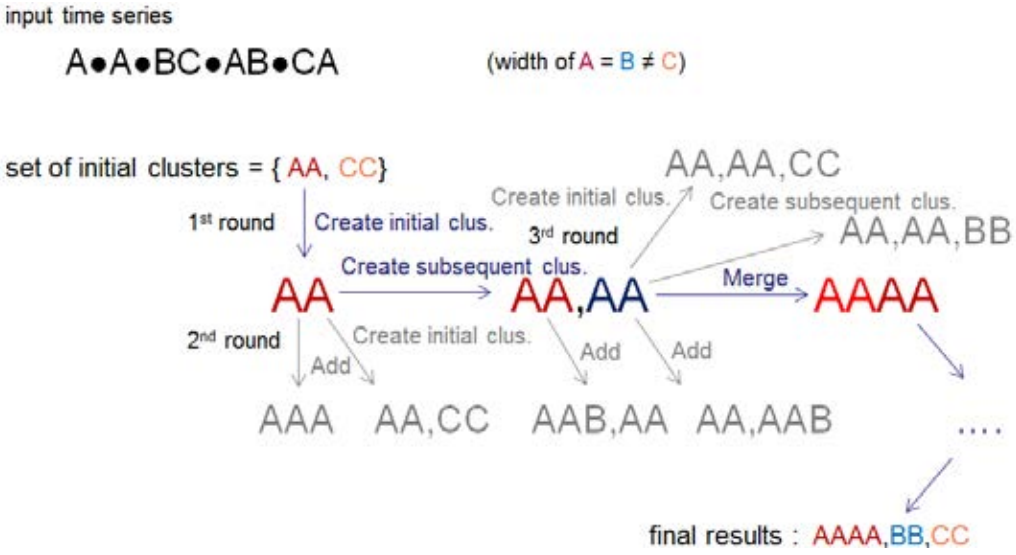
$$\Delta EPW = EPW(\textit{ภายหลัง}) - EPW(\textit{ก่อนหน้า}) \quad \text{---------- (3.6)}$$

อัลกอริทึมในการจัดกลุ่มแสดงเป็นรหัสเทียมดังตารางที่ 3.2 พร้อมทั้งยกตัวอย่างการจัดกลุ่มเพื่อให้เห็นภาพชัดเจนยิ่งขึ้นดังภาพที่ 3.9

ตารางที่ 3.2 อัลกอริทึมในการจัดกลุ่มลำดับย่อย

[*Clusters*] Clustering(*T,MGS*)	
1.	*Clusters* := ∅
2.	While there is an operation left
3.	*Clusters* := ChooseMinError(CreateInitialCluster(*MGS*, *T*, *Clusters*), CreateSubsequentCluster(*MGS, width*, *T*, *Clusters*), Add(*MGS, width*, *T*, *Clusters*), Merge(*Clusters*))
4.	return *Clusters*

กำหนดให้ข้อมูลอนุกรมเวลานำเข้าประกอบด้วยลำดับย่อยจากสามกลุ่มแทนด้วยสัญลักษณ์ดังนี้ A แทนลำดับย่อยในกลุ่ม A B แทนลำดับย่อยในกลุ่ม B และ C แทนลำดับย่อยในกลุ่ม C ซึ่งความยาวของลำดับย่อยในกลุ่ม A และ B มีค่าเท่ากัน แต่ไม่เท่ากับความยาวของลำดับย่อยในกลุ่ม C ส่วน ● แทนข้อมูลส่วนที่ไม่มีความหมายซึ่งจะไม่ถูกจัดกลุ่ม และกำหนดให้ผลลัพธ์จากการคัดเลือกกลุ่มตั้งต้นที่เหมาะประกอบด้วยกลุ่มตั้งต้น AA (ลำดับย่อยสองชุดในกลุ่ม A) และ กลุ่มตั้งต้น CC (ลำดับย่อยสองชุดในกลุ่ม C) เรียงลำดับตามค่าการประหยัดบิตของแต่ละกลุ่ม



ภาพที่ 3.9 แสดงตัวอย่างการจัดกลุ่มของลำดับย่อยด้วยวิธีที่นำเสนอโดยแทนข้อมูลนำเข้าด้วยสัญลักษณ์ (กำหนดให้สีอ่อนแทนขั้นตอนที่ไม่ถูกเลือกระหว่างกระบวนการจัดกลุ่ม)

จากตัวอย่าง ในรอบแรกของการจัดกลุ่มต้องเลือกสร้างกลุ่มจากกลุ่มตั้งต้นที่ดีที่สุดในขณะนั้นจากผลลัพธ์ของกลุ่มตั้งต้นทั้งหมดจากขั้นตอนที่แล้ว (AA และ CC) ซึ่งในที่นี้ คือ AA ต่อมาในรอบที่สองตัวเลือกที่สามารถทำได้ประกอบด้วยการเพิ่ม A ซึ่งเป็นลำดับย่อยที่มีความคล้ายคลึงกับกลุ่ม AA มากที่สุดเข้าไปในกลุ่ม สร้างกลุ่มใหม่จากกลุ่มตั้งต้นที่ดีที่สุดในขณะนี้ คือ CC หรือ สร้างกลุ่มถัดไปในความยาวเดียวกับกลุ่มตั้งต้น ซึ่งในที่นี้คือ AA อีกกลุ่มหนึ่ง สมมติให้การสร้างกลุ่ม AA อีกกลุ่มหนึ่งขึ้นมาให้ค่าผิดพลาดน้อยที่สุดจึงเลือกทำขั้นตอนนี้ ทำให้ผลลัพธ์ในรอบนี้ประกอบด้วยกลุ่มสองกลุ่ม คือ AA และ AA ในรอบที่สามตัวเลือกที่สามารถเลือกได้ในรอบนี้ประกอบด้วยการเพิ่ม B ซึ่งสมมติว่าเป็นลำดับย่อยที่มีความคล้ายคลึงกับกลุ่มที่มีอยู่ ณ ตอนนี้มากที่สุดซึ่งต้องเลือกว่าจะเพิ่มเข้าไปใน AA กลุ่มแรก หรือ AA กลุ่มที่สอง การสร้างกลุ่มตั้งต้นที่ดีที่สุดในขณะนี้ ซึ่งยังคงเป็น CC เช่นเดิมอยู่ การสร้างกลุ่มตั้งต้นถัดไปในความยาวเดียวกัน ในที่นี้คือ BB และเนื่องจากในรอบนี้มีกลุ่มที่ถูกสร้างไว้แล้วสองกลุ่ม ตัวเลือกที่เพิ่มขึ้นมาคือการรวมสองกลุ่มนี้เป็นกลุ่มเดียวกัน และสมมติให้ตัวเลือกที่ให้ค่าผิดพลาดน้อยที่สุดในรอบนี้คือการรวม ผลลัพธ์ในรอบนี้จึงเป็นการรวมกลุ่ม AA และ AA เข้าเป็นกลุ่มเดียวกันคือ AAAA จากนั้นในรอบถัด ๆ ไปก็ให้วนซ้ำทำการจัดกลุ่มในลักษณะนี้ต่อไป

โดยเลือกขั้นตอนที่ให้ค่าผิดพลาดน้อยที่สุด สุดท้ายแล้วจะได้ผลลัพธ์ในการจัดกลุ่มเป็นสามกลุ่มอย่างถูกต้องคือ AAAA, BB และ CC

รายละเอียดของแต่ละขั้นตอน แสดงเป็นรหัสเทียมดังตารางที่ 3.3, 3.4, 3.5 และ 3.6 สังเกตว่าในแต่ละขั้นตอนเมื่อทำการจัดกลุ่มแล้วจะต้องมีการลบลำดับย่อยที่ซ้อนทับกันออก (ทั้งในข้อมูลอนุกรมเวลา และ ในกลุ่มตั้งต้นที่เป็นผลลัพธ์ของขั้นตอนก่อนหน้า) เพื่อป้องกันปัญหาในการจัดกลุ่ม

ตารางที่ 3.3 การสร้างกลุ่มตั้งต้น

[*Clusters*, *epw*] CreateInitialCluster(*MGS*, *T*, *Clusters*)	
1.	*motif* := The first element in *MGS*
2.	*width* := width of *motif*
3.	*Clusters* := Update(*Clusters*, *motif*)
4.	*MGS* := RemoveOverlaps(*MGS*, *motif*) //remove from *MGS*
5.	*T* := RemoveOverlaps(*T*, *motif*) //remove from *T*
6.	*epw* := CalculateError(*Clusters*)
7.	return *Clusters*, *epw*

ตารางที่ 3.4 การสร้างกลุ่มถัดไป

[*Clusters*, *epw*] CreateSubsequentCluster(*MGS, width*, *T*, *Clusters*)	
1.	*motif* := MKMotif(*T*, *width*)
2.	*Clusters* :=Update(*Clusters*, *motif*)
3.	*MGS* := RemoveOverlaps(*MGS*, *motif*) //remove from *MGS*
4.	*T* := RemoveOverlaps(*T*, *motif*) //remove from *T*
5.	*epw* := CalculateError(*Clusters*)
6.	return *Clusters*, *epw*

ตารางที่ 3.5 การเพิ่มลำดับย่อย

[*Clusters*, *epw*] Add(*MGS*, *width*, *T*, *Clusters*)	
1.	*subsequences* := SlidingWindow(*T*, *width*)
2.	for each *c* in *Clusters*
3.	*s* := 1NN(cluster center of *c*, *subsequences*)
4.	*result* := add *s* in to *c*
5.	*Clusters* := Update(*Clusters*, ChooseMinError(*result*))
6.	*MGS* := RemoveOverlaps(*MGS*, *s*) //remove from *MGS*
7.	*T* := RemoveOverlaps(*T*, *s*) //remove from *T*
8.	*epw* := CalculateError(*Clusters*)
9.	retrun *Clusters*, *epw*

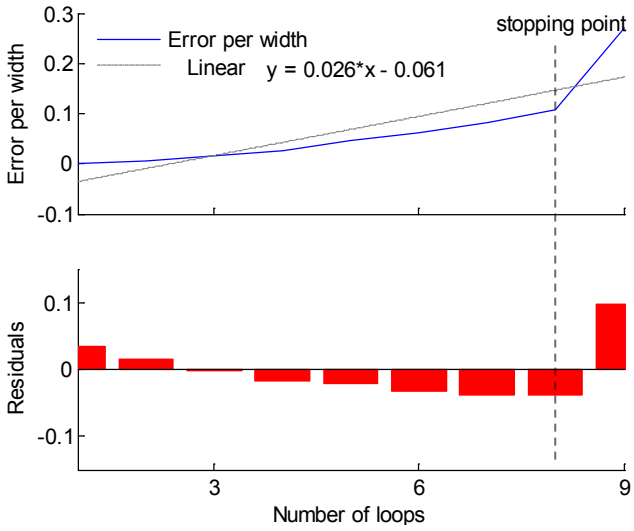
ตารางที่ 3.6 การรวมกลุ่ม

[*Clusters*, *epw*] Merge(*Clusters*)	
1.	for each *c1*, *c2* in *Clusters*
2.	*width* := max(width of *c1*, width of *c2*)
3.	if size of *c1* and *c2* are not equal
4.	min(width of *c1*, width of *c2*) := UniformScaling(min(width of *c1*, width of *c2*), *width*)
5.	*result* := merge *c1* and *c2*
5.	*Clusters* := Update(*Clusters*, ChooseMinError(*result*))
6.	*epw* := CalculateError(*Clusters*)
7.	return *Clusters*, *epw*

สังเกตบรรทัดที่ 3-4 ในตารางที่ 3.6 ในการรวมกลุ่มสองกลุ่มที่มีความยาวไม่เท่ากัน ให้ปรับความยาวของลำดับย่อยภายในกลุ่มที่สั้นกว่าก่อนโดยนำไปผ่านกระบวนการการปรับมาตราแบบเอกรูป (Uniform Scaling)

- เกณฑ์ในการหาจุดสิ้นสุดของการจัดกลุ่ม (Stopping Criterion)

ในการหาจุดสิ้นสุดของการจัดกลุ่มนี้ ทำได้โดยวิเคราะห์จากค่าผิดพลาดของกลุ่มต่อความยาวสะสมในแต่ละรอบของการวนซ้ำเลือกขั้นตอนในการจัดกลุ่ม ซึ่งคำนวณได้จากสมการที่ 3.5 เช่นเดียวกันกับที่คำนวณในขั้นตอนการจัดกลุ่ม จากนั้นให้ลงจุดค่าผิดพลาดของกลุ่มที่คำนวณได้ในแต่ละรอบของการวนซ้ำ และทำการหาสมการเชิงเส้นของกราฟที่ได้ โดยการหาสมการพหุนามดีกรีสอง ด้วยวิธีการประมาณแบบกำลังสองน้อยสุด (The Method of Least Square) และให้เปรียบเทียบกันระหว่างกราฟทั้งสอง จุดที่ห่างกันมากที่สุดระหว่างจุดตัดของกราฟทั้งสองคือจุดสิ้นสุดของการจัดกลุ่ม (ดังภาพที่ 3.10) หากมีจุดที่ห่างที่สุดห่างเท่ากัน ให้ยึดจุดทางขวาสุดเป็นหลัก และผลลัพธ์สุดท้ายของการจัดกลุ่ม คือ ผลลัพธ์ ณ จุดสิ้นสุดการจัดกลุ่มนี้

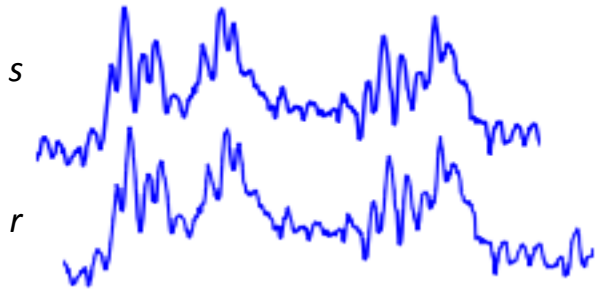


ภาพที่ 3.10 การหาจุดสิ้นสุดของการจัดกลุ่ม โดยจุดสิ้นสุดคือจุดที่กราฟค่าผิดพลาดของกลุ่ม (เส้นทึบ) และ สมการเชิงเส้น (เส้นประ) ห่างกันมากที่สุดในระหว่างจุดตัดของกราฟทั้งสอง

บทที่ 4

การทดลองและวิเคราะห์ผลการทดลอง

หลังจากที่ได้นำเสนอขั้นตอนและวิธีการในการจัดกลุ่มลำดับย่อยโดยปราศจากพารามิเตอร์ไปแล้ว สิ่งที่จะนำเสนอต่อไปในบทนี้ คือ ผลการทดลองเพื่อประเมินผลและสนับสนุนแนวคิดในการทำการจัดกลุ่มลำดับย่อยโดยปราศจากพารามิเตอร์ด้วยวิธีที่ได้นำเสนอไป สิ่งสำคัญในการนำเสนอที่จำเป็นต้องทราบเป็นอันดับแรกในบทนี้ คือ โดยทั่วไปแล้วจัดกลุ่มลำดับย่อยของข้อมูลอนุกรมเวลานั้น ผลลัพธ์ที่ได้จะไม่ออกมาตรงกับผลลัพธ์จริงในทุกจุดข้อมูล หรือ มีส่วนที่เหลื่อมกันเล็กน้อย (ดังภาพที่ 4.1) ดังนั้นการจะตัดสินว่าสามารถจัดกลุ่มได้ถูกต้องหรือไม่ จำเป็นจะต้องมีการกำหนดเกณฑ์ขั้นต่ำของการซ้อนทับกันระหว่างลำดับย่อยผลลัพธ์ที่ค้นพบด้วยอัลกอริทึมที่ทำการตรวจสอบและลำดับย่อยผลลัพธ์จริง ซึ่งในการทดลองนี้กำหนดให้เป็น 80% ของค่าความยาวของลำดับย่อยทั้งสองยูเนียนกัน หากต่ำกว่าจะถือว่าการจัดกลุ่มของลำดับย่อยที่พิจารณาอยู่นั้นไม่ถูกต้อง



ภาพที่ 4.1 ลำดับย่อยสองชุด *s* และ *r* ซึ่งมีส่วนที่เหลื่อมกันอยู่เล็กน้อย

การนำเสนอในบทนี้จะเริ่มกล่าวถึงเครื่องมือที่นำมาใช้ในการประเมินผลก่อน จากนั้นจึงจะแสดงผลการทดลองด้วยเครื่องมือดังกล่าวโดยเปรียบเทียบกับอัลกอริทึมที่ได้กล่าวถึงไปในบทที่ 2 ทั้งสองอัลกอริทึม [5][7] พร้อมทั้งวิเคราะห์ผลการทดลองควบคู่ไปกับการนำเสนอ

4.1 เครื่องมือวัดคุณภาพของการจัดกลุ่ม

Rand Index

หรือ RI [15] เป็นมาตรวัดความเห็นพ้องต้องกันระหว่างผลลัพธ์ในอัลกอริทึมที่จะทำการตรวจสอบและผลลัพธ์จริง ๆ ที่ควรจะเป็น (ผลลัพธ์ที่ถูกต้อง) ซึ่งนิยมมากในการนำมาใช้วัดประสิทธิภาพของการจัดกลุ่ม ในกรณีที่ทราบผลลัพธ์ที่ถูกต้องอยู่แล้ว โดย RI จะวัดค่าความสัมพันธ์ระหว่างคู่ของข้อมูลที่ถูกจัดกลุ่มโดยพิจารณาทุกคู่ที่เป็นไปได้ (สมมติข้อมูลที่พิจารณามีจำนวน n ข้อมูล คู่ของข้อมูลที่นำมาพิจารณา คือ $\binom{n}{2} = \frac{n!}{2!(n-2)!}$) กำหนดให้ผลลัพธ์ของการจัดกลุ่มด้วยอัลกอริทึมที่ตรวจสอบ คือ Cls1 และ ผลลัพธ์ที่ถูกต้องของการจัดกลุ่มนี้ คือ Cls2

ตัวอย่าง *Cls1* = (1,2,2,1,1) และ *Cls2* = (2,1,2,1,1) เมื่อเลขในวงเล็บแสดงหมายเลขของกลุ่มที่ถูกจัดให้กับข้อมูลตำแหน่งนั้น จากตัวอย่างนี้ข้อมูลที่พิจารณามีทั้งหมด 5 ข้อมูล หรือ n=5 จำนวนคู่ของข้อมูลที่นำมาพิจารณาทั้งหมดคือ $\binom{5}{2}$ ซึ่งเท่ากับ 10 ให้พิจารณาทั้งสิบคู่ของข้อมูลในทั้ง *Cls1* และ *Cls2* เปรียบเทียบกัน โดยนับจำนวนของ *a, b, c* และ *d*

เมื่อ *a* คือ จำนวนคู่ของข้อมูลที่อยู่ในกลุ่มเดียวกันทั้งใน *Cls1* และ *Cls2*

b คือ จำนวนคู่ของข้อมูลที่อยู่ในกลุ่มเดียวกันใน *Cls1* แต่ต่างกลุ่มกันใน *Cls2*

c คือ จำนวนคู่ของข้อมูลที่อยู่ต่างกลุ่มกันใน *Cls1* แต่อยู่กลุ่มเดียวกันใน *Cls2*

และ *d* คือ จำนวนคู่ของข้อมูลที่อยู่ต่างกลุ่มกันทั้งใน *Cls1* และ *Cls2*

(แสดงเป็นตารางความสัมพันธ์ ดังตารางที่ 4.0)

ตารางที่ 4.1 ความสัมพันธ์ของค่า *a*, *b*, *c* และ *d* ในการคำนวณค่า RI

Cls1 \ *Cls2*	คู่ที่เหมือนกัน	คู่ที่ต่างกัน
คู่ที่เหมือนกัน	*a*	*b*
คู่ที่ต่างกัน	*c*	*d*

จากตัวอย่าง ข้อมูลคู่แรกที่นำมาพิจารณาคือ (1,2) ใน *Cls1* และ (2,1) ใน *Cls* 2 (ดังตารางที่ 4.2) ซึ่งจะเห็นว่าอยู่กลุ่มเดียวกันใน *Cls1* แต่ต่างกลุ่มกันใน *Cls2* ดังนั้นค่า *d* จึงถูกนับเพิ่มเป็น 1

ตารางที่ 4.2 ตัวอย่างข้อมูลคู่แรกที่นำมาพิจารณาในการคำนวณค่า RI
(คู่ของข้อมูลในช่องที่แรเงา พิจารณาในแนวนอน)

Cls1	1	2	2	1	1
Cls2	2	1	2	1	1

คู่ถัดมาที่ทำการพิจารณาคือ (1,2) ใน *Cls1* และ (2,2) ใน *Cls* 2 (ดังตารางที่ 4.3) ในกรณีนี้ค่า *b* ถูกนับเพิ่มเป็น 1 และให้พิจารณาเช่นนี้ไปจนครบทั้งสิบคู่ จะได้ค่า *a*, *b*, *c* และ *d* ที่ครบถ้วน ซึ่งสำหรับตัวอย่างนี้ คือ $a = 1$, $b = 3$, $c = 3$ และ $d = 3$

ตารางที่ 4.3 ตัวอย่างข้อมูลคู่ที่สองที่นำมาพิจารณาในการคำนวณค่า RI

Cls1	1	2	2	1	1
Cls2	2	1	2	1	1

จากนั้นให้นำมาคำนวณค่า RI ดังนี้

$$RI = \frac{a+b}{a+b+c+d} \text{ ---------- (4.1)}$$

ค่า RI จะมีค่าอยู่ระหว่าง 0 ถึง 1 เรียงตามความถูกต้อง ซึ่งจากตัวอย่างนี้ค่า RI เท่ากับ 0.4 เพราะมีผลลัพธ์ที่ต่างกันถึง 2 ตำแหน่ง แต่การคิด RI นี้มีข้อจำกัดอยู่บางประการเมื่อนำมาใช้กับงานวิจัยนี้ คือ มาตรวัดนี้ไม่มีการนิยามความแตกต่างระหว่างข้อมูลที่ไม่ถูกจัดกลุ่มกับข้อมูลที่จัดกลุ่มผิด ดังนั้นวิธีเบื้องต้นในการคำนวณเมื่อมีข้อมูลที่ไม่ถูกจัดกลุ่มอยู่ คือ กำหนดกลุ่มให้อยู่ในกลุ่มอื่นที่นอกเหนือจากกลุ่มผลลัพธ์ทั้งหมดในการจัดกลุ่ม ซึ่งอาจทำให้เกิดความไม่เท่าเทียมในการประเมินผล จึงนำเสนอเครื่องมือวัดอื่น ๆ มาประเมินผลควบคู่ไปด้วย

***F*-Measure**

หรือ F_1 คือ มาตรวัดอีกชนิดหนึ่งที่สามารถนำมาใช้วัดความแม่นยำในการจัดกลุ่ม โดยพิจารณาทั้งค่า Precision (p) และ ค่า Recall (r) ของผลลัพธ์ในการจัดกลุ่มเทียบกับผลลัพธ์ที่ถูกต้องจริง โดยในที่นี้เลือกที่จะคำนวณค่า F_1 ของเซตของลำดับย่อยผลลัพธ์มากกว่าการคำนวณค่า F_1 ของแต่ละกลุ่มแล้วจึงหาค่า F_1 รวมของการจัดกลุ่ม โดยมีนิยามการคำนวณดังนี้

$$p = \frac{m}{n_C} \quad \text{---------- (4.2)} \qquad\qquad r = \frac{m}{n_R} \quad \text{---------- (4.3)}$$

เมื่อ m คือ จำนวนของลำดับย่อยที่จัดกลุ่มถูกต้องตรงกับผลลัพธ์จริง

n_C คือ จำนวนของลำดับย่อยทั้งหมดที่เป็นผลลัพธ์ของอัลกอริทึมที่ตรวจสอบ

และ n_R คือ จำนวนของลำดับย่อยทั้งหมดที่นำมาวิเคราะห์ (ลำดับย่อยผลลัพธ์)

$$F_1 = \frac{2 \cdot p \cdot r}{p + r} \quad \text{---------- (4.4)}$$

ค่า F_1 นี้ จะมีค่าตั้งแต่ 0 ถึง 1 เรียงตามความถูกต้อง เช่นเดียวกับ RI และสามารถแก้ปัญหาในส่วนของความแตกต่างระหว่างข้อมูลที่ไม่ถูกจัดกลุ่มและข้อมูลที่จัดกลุ่มผิดได้

Accuracy-on-Detection

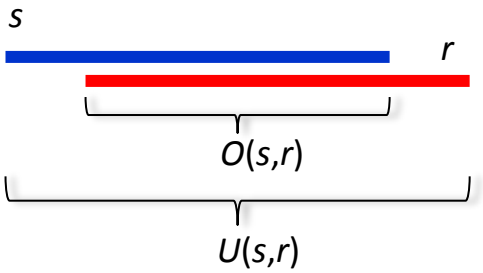
หรือ AoD [16] เป็นมาตรวัดที่ออกแบบขึ้นมาเพื่อวัดคุณภาพของอัลกอริทึมในงานวิจัยทางด้านข้อมูลอนุกรมเวลาโดยเฉพาะ จึงมีความเหมาะสมที่จะนำมาใช้ในงานวิจัยนี้ โดยใช้เปรียบเทียบความเหมือนระหว่างผลลัพธ์ของข้อมูลอนุกรมเวลาที่ได้จากอัลกอริทึมที่จะทำการตรวจสอบกับผลลัพธ์จริง เป็นเปอร์เซ็นต์ โดยคำนวณอัตราส่วนระหว่างส่วนที่ซ้อนทับกันและส่วนที่ยูเนียนของผลลัพธ์ทั้งสอง ซึ่งหลังจากทำการปรับปรุงให้เหมาะสมกับการนำมาใช้ในงานวิจัยนี้แล้ว มีนิยามการคำนวณดังนี้

กำหนดให้ผลลัพธ์จากการจัดกลุ่มลำดับย่อยด้วยอัลกอริทึมที่ตรวจสอบ คือ $C = \{ C_i \mid 0 < i \le k_C \}$ เมื่อ k_C คือ จำนวนกลุ่มทั้งหมดที่เป็นผลลัพธ์ของอัลกอริทึมนี้ และ แต่ละกลุ่มใด ๆ ให้ $C_i = \{s_{i_1}, s_{i_2}, s_{i_3}, \ldots, s_{n_i}\}$ เมื่อ s_{i_k} คือ แต่ละลำดับย่อยใด ๆ ในกลุ่ม C_i ซึ่งมีจำนวนเท่ากับ n_i ชุด

และ กำหนดให้ผลลัพธ์ที่ถูกต้องของการจัดกลุ่มลำดับย่อย คือ $R = \{ R_j \mid 0 < j \le k_R \}$ เมื่อ k_R คือ จำนวนกลุ่มทั้งหมดของผลลัพธ์ที่ถูกต้อง และ แต่ละกลุ่มใด ๆ ให้ $R_j = \{r_{j_1}, r_{j_2}, r_{j_3}, \ldots, r_{n_j}\}$ เมื่อ r_{j_k} คือ แต่ละลำดับย่อยใด ๆ ในกลุ่ม R_j ซึ่งมีจำนวนเท่ากับ n_j ชุด จะได้

$$AoD = \frac{\sum_{j=1}^{k_R} \sum_{s \in C_j, r \in R_j} O(s,r)}{\sum_{j=1}^{k_R} \sum_{s \in C_j, r \in R_j} U(s,r)} \times 100\% \quad \text{---------- (4.5)}$$

เมื่อ $O(s,r)$ คือ จำนวนจุดข้อมูลที่ซ้อนทับกันระหว่างลำดับย่อย s และ r และ $U(s,r)$ คือ จำนวนจุดข้อมูลที่ยูเนียนกันระหว่างลำดับย่อย s และ r (ดังภาพที่ 4.2) ทั้งนี้ให้คิดเฉพาะลำดับย่อยที่ผลลัพธ์ตรงกันระหว่างกลุ่ม C ใด ๆ และ กลุ่ม R ใด ๆ (หรือ คิดเฉพาะลำดับย่อยที่จัดกลุ่มถูกต้องเท่านั้น) ลำดับย่อยอื่น ๆ นอกเหนือจากนี้ รวมทั้งลำดับย่อยที่ไม่ถูกจัดกลุ่มใน C ทั้ง ๆ ที่ถูกจัดกลุ่มใน R ให้ค่า $O(s,r) = 0$ และ $U(s,r) = U(r,r)$



ภาพที่ 4.2 แสดงส่วนที่ซ้อนทับกัน $O(s,r)$ และ ส่วนที่ยูเนียนกัน $U(s,r)$ ระหว่างลำดับย่อย s และ r

4.2 ผลการทดลอง

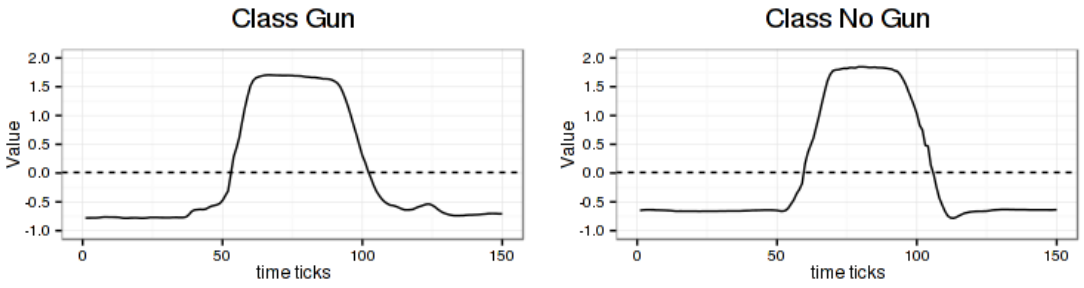
การทดลองในที่นี้ ใช้ข้อมูลทดลองจากข้อมูลการจัดกลุ่มและการจำแนกประเภทข้อมูลอนุกรมเวลาของมหาวิทยาลัยแคลิฟอร์เนีย ริเวอร์ไซด์ (The UCR Time Series Classification /Clustering Archive) ซึ่งเป็นฐานข้อมูลอนุกรมเวลาที่ใหญ่ที่สุดในโลก และได้รับการยอมรับอย่างเป็นสากล โดยนำข้อมูลจากแต่ละกลุ่มมาต่อกัน และคั่นกลางข้อมูลเหล่านี้ด้วยข้อมูลแบบสุ่ม (Randomwalk Data) เพื่อใช้เป็นข้อมูลอนุกรมเวลานำเข้าในการทดลอง ทั้งนี้ได้แบ่งการทดลอง

ออกเป็นสองกรณี คือ 1. กรณีที่ข้อมูลอนุกรมเวลาประกอบด้วยลำดับย่อยความยาวเดียวกัน และ 2. กรณีที่ข้อมูลอนุกรมเวลาประกอบด้วยความยาวแตกต่างกัน และแสดงผลการทดลองในแต่ละกรณีดังนี้

4.2.1 ข้อมูลประกอบด้วยด้วยลำดับย่อยความยาวเดียวกัน

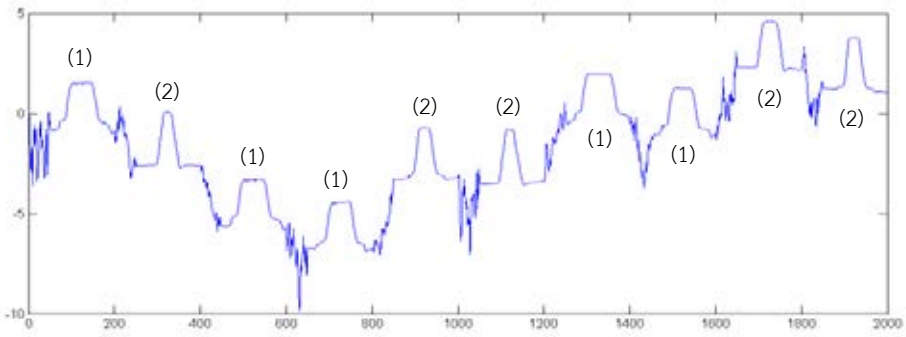
ในการทดลองนี้นำข้อมูลอนุกรมเวลาซึ่งประกอบด้วยลำดับย่อยแต่ละประเภทมาทำการวิเคราะห์และจัดกลุ่มตามความเหมาะสม โดยในการทดลองแต่ละครั้ง ข้อมูลอนุกรมเวลานำเข้าที่นำมาวิเคราะห์อาจประกอบด้วยกลุ่มของลำดับย่อยหลายกลุ่ม แต่ลำดับย่อยในทุก ๆ กลุ่มจะมีความยาวที่เท่ากัน และเปรียบเทียบผลลัพธ์ของการจัดกลุ่มที่ได้จากอัลกอริทึมที่นำเสนอ ซึ่งจะขอใช้ตัวย่อแทนด้วย PFSTS Clustering กับทั้งสองอัลกอริทึมในการจัดกลุ่มที่นำเสนอไปในบทที่ 2 ซึ่งจะขอใช้ตัวย่อแทนเช่นกัน ดังนี้ SSTS Clustering แทนอัลกอริทึมในงานวิจัย [7] และ MDL Clustering แทนอัลกอริทึมในงานวิจัย [5] โดยกำหนดพารามิเตอร์ให้เป็นค่าความยาวจริงของลำดับย่อยในข้อมูลทดลองซึ่งทราบค่าอยู่แล้ว ข้อมูลที่นำมาทำการทดลองและแสดงผลลัพธ์ให้เห็น มีดังนี้

1. ข้อมูล Gun-Point คือ ข้อมูลซึ่งบันทึกจากภาพถ่ายวีดีโอจำลองการชักปืนพกออกจากซองปืนขึ้นมาตั้งท่าเตรียมยิงและเก็บกลับเข้าไปในซองปืน โดยเปรียบเทียบความต่างระหว่างการชักปืนจริง ๆ กับ การชักปืนด้วยมือเปล่า ทำให้ได้ข้อมูลออกมา 2 กลุ่ม (ดังภาพที่ 4.3)



ภาพที่ 4.3 ความแตกต่างระหว่างข้อมูล Gun-Point ทั้งสองกลุ่ม
(ซ้าย) ข้อมูลกลุ่มที่ 1 การชักปืนจริง ๆ และ (ขวา) ข้อมูลกลุ่มที่ 2 การชักปืนด้วยมือเปล่า

ข้อมูลทั้งสองกลุ่มนี้มีความยาวเดียวกัน คือ 150 จุดข้อมูล นำข้อมูลจากทั้งสองกลุ่มมาต่อกัน โดยสุ่มเลือกตัวอย่างมาจากแต่ละกลุ่ม กลุ่มละ 5 ตัวอย่าง คั่นกลางด้วยข้อมูลแบบสุ่มความยาว 50 จุดข้อมูล ได้ผลลัพธ์เป็นข้อมูลอนุกรมนำเข้าหนึ่งชุด ความยาวรวม 2000 จุดข้อมูล (ดังภาพที่ 4.4) และแสดงรายละเอียดของข้อมูล ดังตารางที่ 4.4 โดยแสดงตำแหน่งและกลุ่มที่ถูกต้องของแต่ละข้อมูลตัวอย่างในรูปแบบ a(b) เมื่อ a แทนหมายเลขของกลุ่มที่ถูกต้อง และ b แทนตำแหน่งของข้อมูลที่ถูกต้องบนข้อมูลอนุกรมเวลา

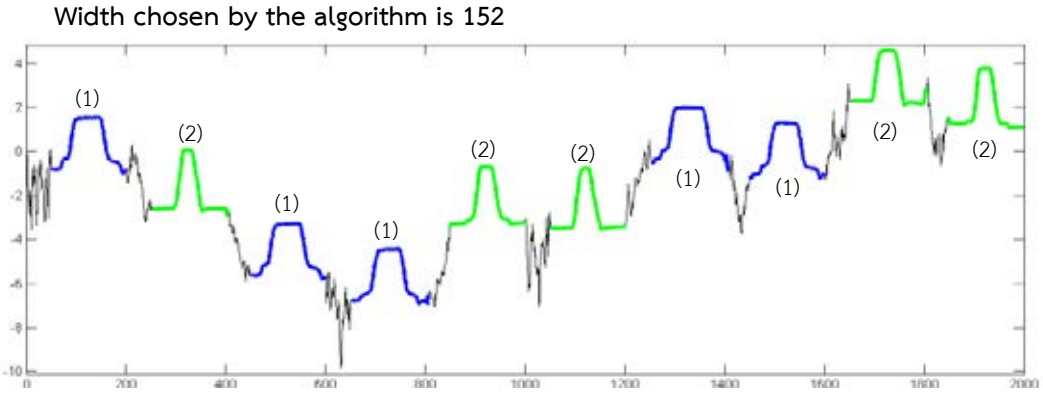


ภาพที่ 4.4 ข้อมูลอนุกรมเวลานำเข้าชุดที่ 1 จากข้อมูล Gun-Point คั่นกลางด้วยข้อมูลแบบสุ่ม

ตารางที่ 4.4 รายละเอียดของข้อมูลอนุกรมเวลานำเข้าชุดที่ 1

Example No.	1	2	3	4	5	6	7	8	9	10
Input Index	1(50)	2(250)	1(450)	1(650)	2(850)	2(1050)	1(1250)	1(1450)	2(1650)	2(1850)

ผลลัพธ์เมื่อทำการทดลองกับข้อมูลนำเข้าชุดนี้ด้วยอัลกอริทึมที่นำเสนอ แสดงดังภาพที่ 4.5 ด้วยค่าความยาวที่อัลกอริทึมเลือก คือ 152 และรายละเอียดของผลลัพธ์เปรียบเทียบกับอัลกอริทึมก่อนหน้าเมื่อกำหนดให้ค่าความยาวในการจัดกลุ่มเป็น 150 แสดงดังตารางที่ 4.5



ภาพที่ 4.5 ผลลัพธ์ของการจัดกลุ่มข้อมูลอนุกรมเวลานำเข้าชุดที่ 1
ด้วยอัลกอริทึม PFSTS Clustering

ตารางที่ 4.5 รายละเอียดของผลลัพธ์จากการทดลองสำหรับข้อมูลอนุกรมเวลานำเข้าชุดที่ 1

Example No.	1	2	3	4	5	6	7	8	9	10
PFSTS Clustering	1(49)	2(252)	1(449)	1(653)	2(848)	2(1049)	1(1253)	1(1449)	2(1653)	2(1848)
SSTS Clustering	1(51)	2(252)	1(451)	1(653)	2(850)	2(1049)	1(1255)	1(1449)	---------	2(1850)
MDL Clustering	1(51)	2(252)	1(451)	1(653)	3(850)	2(1049)	1(1255)	1(1449)	3(1653)	3(1850)

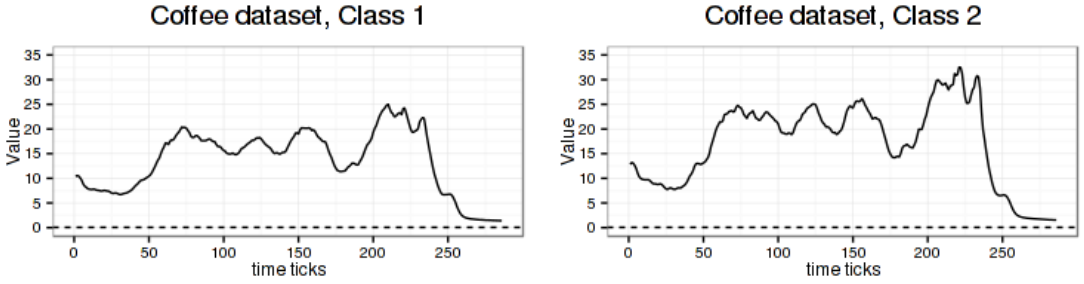
โดยทั่วไปแล้วค่าความยาวที่คลาดเคลื่อนในการจัดกลุ่มเพียงเล็กน้อยจะไม่ส่งผลต่อผลลัพธ์อย่างร้ายแรงเมื่อเทียบกับการจัดกลุ่มด้วยค่าความยาวจริง แต่จะส่งผลต่อค่า AoD เพียงเล็กน้อยเท่านั้น น้อยครั้งที่จะส่งผลต่อค่า RI และ F_1 (ทั้งนี้พิจารณาในอัลกอริทึมการจัดกลุ่มเดียวกัน) แต่ในกรณีของข้อมูลอนุกรมเวลานำเข้าชุดที่ 1 นี้ ค่าความยาวที่คลาดเคลื่อนกลับส่งผลดีต่อผลลัพธ์ เพราะทำให้จุดสิ้นสุดของการจัดกลุ่มเลื่อนออกไปหนึ่งขั้นตอน ทำให้ผลลัพธ์ของอัลกอริทึมที่นำเสนอถูกต้องและครบถ้วนที่สุด ดังที่เห็นว่า SSTS Clustering ไม่สามารถจัดกลุ่มให้กับลำดับย่อยชุดที่ 9 ได้ ส่วน MDL Clustering นั้นได้ผลลัพธ์เป็น 3 กลุ่ม ทั้ง ๆ ที่กลุ่มที่ 2 และ กลุ่มที่ 3 ควรจะเป็นกลุ่มเดียวกัน

ที่เป็นเช่นนี้ก็เพราะจุดสิ้นสุดของการจัดกลุ่มของทั้งสองอัลกอริทึมนั้นหยุดก่อนที่ควรจะเป็นไปหนึ่งขั้นตอน ต่างจากในอัลกอริทึมที่นำเสนอซึ่งสามารถหยุดจัดกลุ่มได้ในตำแหน่งที่ถูกต้อง จึงมีความแม่นยำสูงกว่าในทุกเครื่องมือวัด (ดังตารางที่ 4.6) ซึ่งถ้าสมมติให้ทั้งสองอัลกอริทึมที่เปรียบเทียบด้วยนี้ หยุดจัดกลุ่มช้ากว่าที่ควรจะเป็นหนึ่งขั้นตอน ก็จะให้ผลลัพธ์ที่เหมือนกัน (SSTS Clustering จะเพิ่มลำดับย่อยชุดที่ 9 เข้ามาในกลุ่มที่ 2 ในขณะที่ MDL Clustering จะรวมกลุ่มที่ 2 และ กลุ่มที่ 3 เป็นกลุ่มเดียวกัน) ทั้งค่า RI, AoD และ F_1 จะสูงขึ้นมาในระดับที่เท่าเทียมกัน

ตารางที่ 4.6 เปรียบเทียบค่า Rand Index, AoD และ F_1 ของแต่ละอัลกอริทึม
สำหรับข้อมูลอนุกรมเวลานำเข้าชุดที่ 1

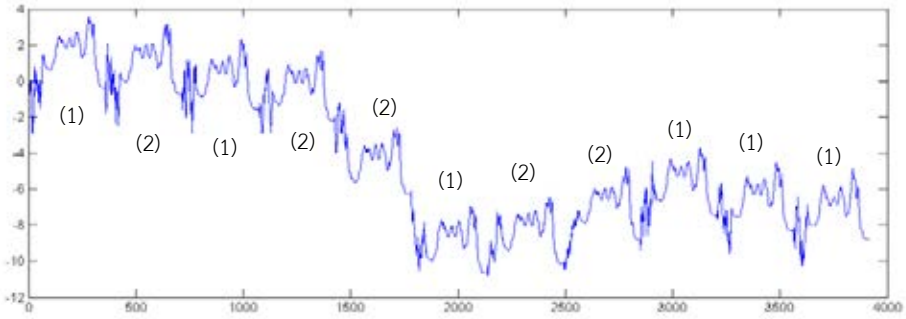
Measurement / Algorithm	RI	AoD	F_1
PFSTS Clustering	**1.00**	**97.26%**	**1.00**
SSTS Clustering	0.91	88.30%	0.95
MDL Clustering	0.87	68.43%	0.70

2. ข้อมูล Coffee คือ ข้อมูลสเปคโทรแกรมทางด้านอาหาร (Food Spectrogram) ซึ่งได้จากการวิเคราะห์องค์ประกอบทางเคมีของอาหาร ซึ่งในที่นี้วิเคราะห์ลักษณะของกาแฟสองสายพันธุ์ คือ อราบิกา (Arabica) และ โรบัสตา (Robusta) ทำให้ได้ข้อมูลเป็น 2 กลุ่ม (ดังภาพที่ 4.6)



ภาพที่ 4.6 ความแตกต่างระหว่างข้อมูล Coffee ทั้งสองกลุ่ม
(ซ้าย) ข้อมูลกลุ่มที่ 1 กาแฟอราบิกา และ (ขวา) ข้อมูลกลุ่มที่ 2 กาแฟโรบัสตา

ข้อมูลทั้งสองกลุ่มนี้มีความยาวเดียวกัน คือ 286 จุดข้อมูล นำข้อมูลจากทั้งสองกลุ่มมาต่อกัน โดยสุ่มเลือกตัวอย่างมา 6 ตัวอย่าง จากกลุ่มที่หนึ่ง และอีก 5 ตัวอย่าง จากกลุ่มที่สอง คั่นกลางด้วยข้อมูลแบบสุ่มความยาว 70 จุดข้อมูล ได้ผลลัพธ์เป็นข้อมูลอนุกรมนำเข้าหนึ่งชุด ความยาวรวม 3916 จุดข้อมูล (ดังภาพที่ 4.7) และแสดงรายละเอียดของข้อมูล ดังตารางที่ 4.7

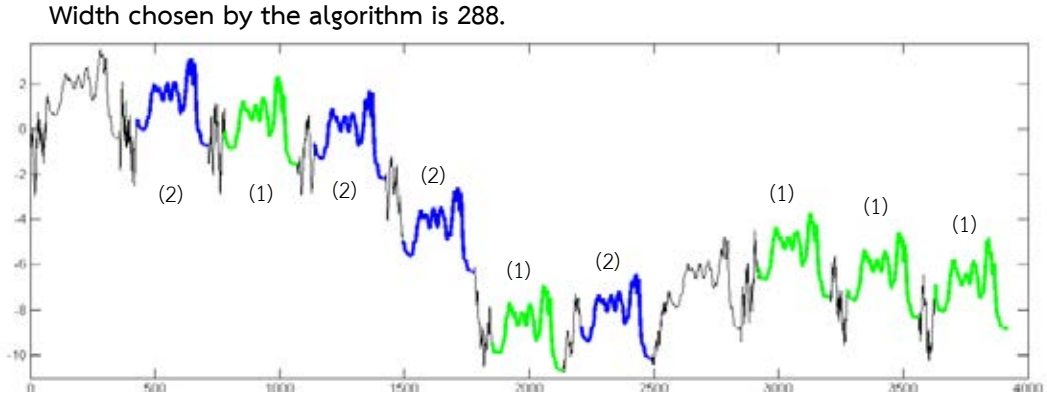


ภาพที่ 4.7 ข้อมูลอนุกรมเวลานำเข้าชุดที่ 2 จากข้อมูล Coffee คั่นกลางด้วยข้อมูลแบบสุ่ม

ตารางที่ 4.7 รายละเอียดของข้อมูลอนุกรมเวลานำเข้าชุดที่ 2

Example No.	1	2	3	4	5	6	7	8	9	10	11
Input Index	1(70)	2(426)	1(782)	2(1138)	2(1494)	1(1850)	2(2206)	2(2562)	1(2918)	1(3274)	1(3630)

เมื่อนำข้อมูลชุดนี้ไปประมวลผลด้วยอัลกอริทึมที่นำเสนอ ได้ผลลัพธ์ดังภาพที่ 4.8 ด้วยค่าความยาวที่อัลกอริทึมเลือก คือ 288 และแสดงรายละเอียดของผลลัพธ์เปรียบเทียบกับอัลกอริทึมก่อนหน้าโดยกำหนดความยาวของการจัดกลุ่มให้เป็น 286 ดังตารางที่ 4.8



ภาพที่ 4.8 ผลลัพธ์ของการจัดกลุ่มข้อมูลอนุกรมเวลานำเข้าชุดที่ 2
ด้วยอัลกอริทึม PFSTS Clustering

ตารางที่ 4.8 รายละเอียดของผลลัพธ์จากการทดลองสำหรับข้อมูลอนุกรมเวลานำเข้าชุดที่ 2

Example No.	1	2	3	4	5	6	7	8	9	10	11
PFSTS Clustering	-------	2(425)	1(780)	2(1137)	2(1493)	1(1849)	2(2205)	-------	1(2917)	1(3273)	1(3628)
SSTS Clustering	-------	2(427)	1(782)	2(1139)	2(1495)	1(1851)	2(2207)	------	1(2919)	1(3275)	1(3630)
MDL Clustering	2(71)	2(431)	1(782)	2(1143)	2(1495)	1(1851)	2(2207)	2(2563)	1(2919)	1(3275)	1(3630)

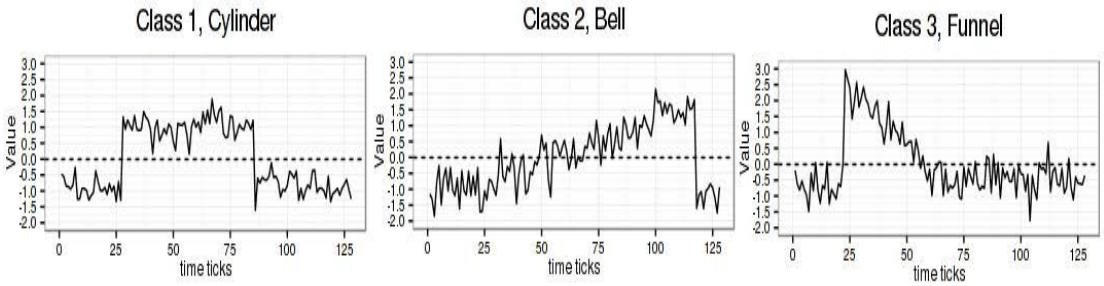
ในข้อมูลอนุกรมเวลานำเข้าชุดที่ 2 นี้ ความยาวในการเลือกจัดกลุ่มยังคงคลาดเคลื่อนจากที่ควรจะเป็นเล็กน้อย ผลลัพธ์ของการจัดกลุ่มที่ได้เป็นผลลัพธ์เดียวกับ SSTS Clustering แต่มีค่า AoD ที่ต่ำกว่าเล็กน้อยเนื่องจากความยาวที่คลาดเคลื่อนนี้ (ดังตารางที่ 4.9) จะเห็นว่าในกรณีนี้ MDL Clustering จัดกลุ่มให้กับลำดับย่อยชุดที่ 1 ผิดพลาดไป ซึ่งควรจะถูกจัดอยู่ในกลุ่มที่ 1 ในขณะที่ SSTS Clustering และอัลกอริทึมที่นำเสนอจัดกลุ่มให้กับลำดับย่อยขาดไปสองชุดเนื่องจากการจัดกลุ่มถึงจุดสิ้นสุดเร็วเกินไป (เร็วกว่าสองขั้นตอนเมื่อเทียบกับ MDL Clustering) ทำให้ผลลัพธ์โดยรวม

ในข้อมูลชุดนี้มีความแม่นยำน้อยกว่า MDL Clustering เล็กน้อย โดยมีค่า AoD ที่น้อยกว่าเพราะ MDL Clustering จัดกลุ่มให้กับลำดับย่อยถูกต้องมากกว่า 1 ชุด ส่วนค่า F_1 ไม่ต่างกันมาก เพราะเครื่องมือวัดนี้มีการให้ความสำคัญระหว่างการไม่จัดกลุ่มและการจัดกลุ่มผิด ถึงแม้ MDL Clustering จะจัดกลุ่มครบถ้วนกว่า แต่มีการจัดกลุ่มผิด จึงถูกหักคะแนนในส่วนนี้ไป ส่วนค่า RI ไม่สามารถบ่งบอกได้ชัดเจน เพราะเครื่องมือวัดนี้ไม่มีการคำนึงถึงลำดับย่อยที่ไม่ถูกจัดกลุ่ม

ตารางที่ 4.9 เปรียบเทียบค่า Rand Index, AoD และ F_1 ของแต่ละอัลกอริทึม สำหรับข้อมูลอนุกรมเวลานำเข้าชุดที่ 2

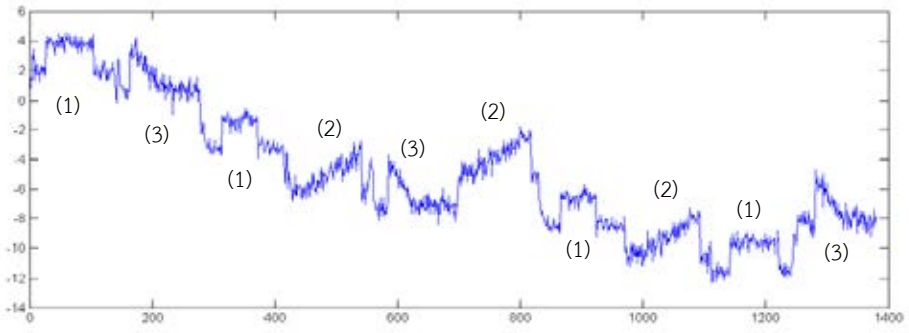
Measurement / Algorithm	RI	AoD	F_1
PFSTS Clustering	0.82	81.25%	0.90
SSTS Clustering	0.82	81.41%	0.90
MDL Clustering	0.82	**88.94%**	**0.91**

3. ข้อมูล CBF คือ ข้อมูลสังเคราะห์ซึ่งถูกออกแบบมาเพื่อวัดประสิทธิภาพของการจำแนกประเภท หรือ การจัดกลุ่ม โดยเฉพาะ ข้อมูลนี้ประกอบด้วย 3 กลุ่ม เรียกว่า Cylinder, Bell และ Funnel ตามลำดับ (ดังภาพที่ 4.9)



ภาพที่ 4.9 ความแตกต่างระหว่างข้อมูล CBF ทั้งสามกลุ่ม (ซ้าย) ข้อมูลกลุ่มที่ 1 Cylinder (กลาง) ข้อมูลกลุ่มที่ 2 Bell และ (ขวา) ข้อมูลกลุ่มที่ 3 Funnel

ข้อมูลทั้งสามกลุ่มนี้มีความยาวเดียวกัน คือ 128 จุดข้อมูล นำข้อมูลจากทั้งหมดมาต่อกันและคั่นกลางด้วยข้อมูลแบบสุ่มความยาว 10 โดยสุ่มเลือกตัวอย่างมาจากแต่ละกลุ่มดังนี้ กลุ่มที่หนึ่ง 4 ตัวอย่าง กลุ่มที่สอง 3 ตัวอย่าง และ กลุ่มที่สาม 3 ตัวอย่าง ได้ผลลัพธ์เป็นข้อมูลอนุกรมนำเข้าหนึ่งชุด ความยาวรวม 1380 จุดข้อมูล (ดังภาพที่ 4.10) และแสดงรายละเอียดของข้อมูล ดังตารางที่ 4.10

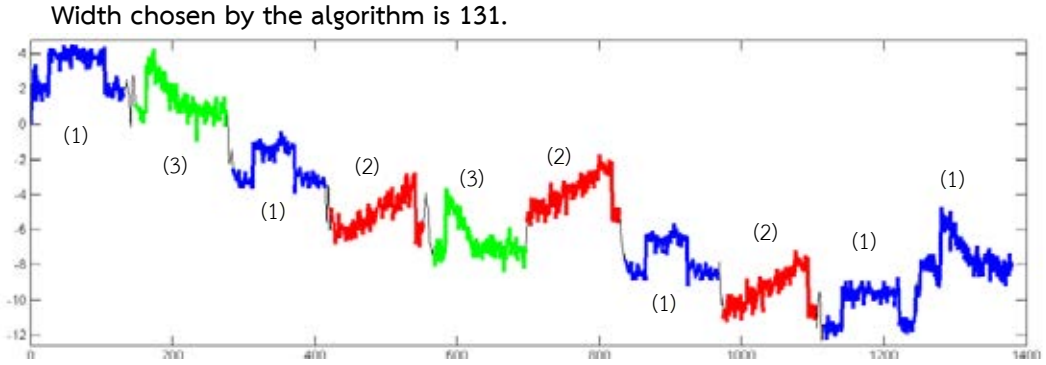


ภาพที่ 4.10 ข้อมูลอนุกรมเวลานำเข้าชุดที่ 3 จากข้อมูล CBF คั่นกลางด้วยข้อมูลแบบสุ่ม

ตารางที่ 4.10 รายละเอียดของข้อมูลอนุกรมเวลานำเข้าชุดที่ 3

Example No.	1	2	3	4	5	6	7	8	9	10
Input Index	1(10)	3(148)	1(286)	2(424)	3(562)	2(700)	1(838)	2(976)	1(1114)	3(1252)

ผลลัพธ์เมื่อทำการทดลองกับข้อมูลนำเข้าชุดนี้ด้วยอัลกอริทึมที่นำเสนอ แสดงดังภาพที่ 4.11 ด้วยค่าความยาวที่อัลกอริทึมเลือก คือ 131 และรายละเอียดของผลลัพธ์เปรียบเทียบกับอัลกอริทึมก่อนหน้าเมื่อกำหนดให้ค่าความยาวในการจัดกลุ่มเป็น 128 แสดงดังตารางที่ 4.11



ภาพที่ 4.11 ผลลัพธ์ของการจัดกลุ่มข้อมูลอนุกรมเวลานำเข้าชุดที่ 3
ด้วยอัลกอริทึม PFSTS Clustering

ตารางที่ 4.11 รายละเอียดของผลลัพธ์จากการทดลองสำหรับข้อมูลอนุกรมเวลานำเข้าชุดที่ 3

Example No.	1	2	3	4	5	6	7	8	9	10
PFSTS Clustering	1(0)	3(146)	1(284)	2(422)	3(566)	2(698)	1(836)	2(974)	1(1115)	1(1249)
SSTS Clustering	1(1)	3(149)	1(287)	2(425)	3(569)	2(701)	1(839)	2(977)	1(1116)	1(1250)
MDL Clustering	5(1)	3(149)	1(287)	4(441)	3(569)	2(701)	1(839)	2(977)	5(1116)	4(1244)

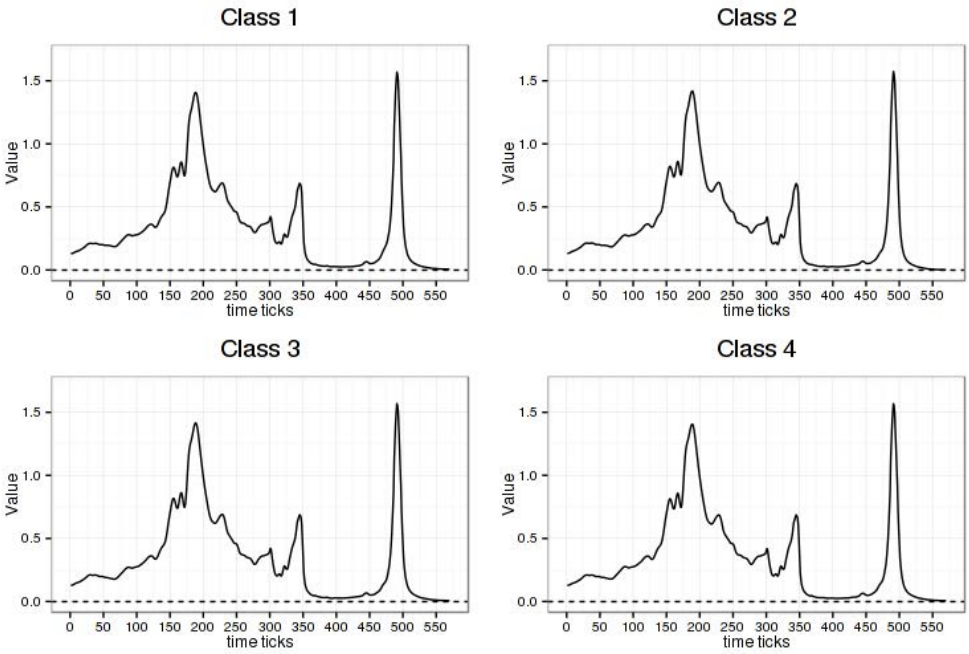
จากผลลัพธ์สำหรับข้อมูลอนุกรมเวลานำเข้าชุดที่ 3 นี้ อัลกอริทึมที่นำเสนอจัดกลุ่มให้กับลำดับย่อยชุดที่ 10 ซึ่งควรจะอยู่ในกลุ่มที่ 3 ผิดพลาดไป เช่นเดียวกับ SSTS Clustering จึงมีค่า F_1 และ RI ที่เท่ากัน (ดังตารางที่ 4.12) ในขณะที่ค่า AoD ยังคงต่ำกว่าเล็กน้อยเนื่องจากความยาวที่คลาดเคลื่อนเช่นเคย อย่างไรก็ตามผลลัพธ์สำหรับข้อมูลชุดนี้ดีกว่า MDL Clustering มาก ทั้งนี้เพราะจำนวนกลุ่มที่ MDL Clustering จัดให้กับลำดับย่อยทั้งหมดมีถึง 5 กลุ่ม ซึ่งเกินจากจำนวนกลุ่มจริงมาถึงสองกลุ่ม และในกรณีนี้ต่อให้ขยับจุดสิ้นสุดของการจัดกลุ่มของ MDL Clustering ออกไปจนเหลือจำนวนกลุ่ม 3 กลุ่มเท่ากัน ก็ยังได้ผลลัพธ์ที่มีความแม่นยำต่ำกว่า เพราะอัลกอริทึมจะทำการ

รวมกลุ่ม 4, 5 และ 2 เข้าด้วยกัน ปรากฏว่าจัดกลุ่มให้กับลำดับย่อยผิดไปถึง 3 ชุด คือ ชุดที่ 1, 9 และ 10 ปัญหาในกรณีนี้จึงมาจากตัวอัลกอริทึมเอง

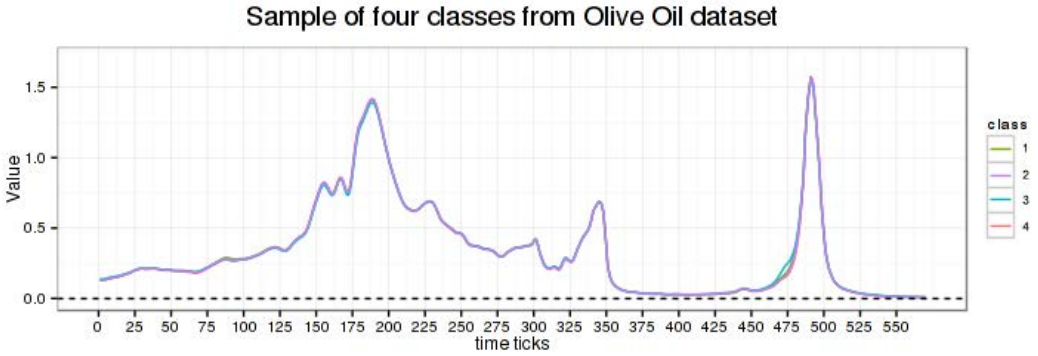
ตารางที่ 4.12 เปรียบเทียบค่า Rand Index, AoD และ F_1 ของแต่ละอัลกอริทึม สำหรับข้อมูลอนุกรมเวลานำเข้าชุดที่ 3

Measurement / Algorithm	RI	AoD	F_1
PFSTS Clustering	**0.87**	**86.43%**	**0.90**
SSTS Clustering	**0.87**	**86.50%**	**0.90**
MDL Clustering	0.80	58.51%	0.60

4. ข้อมูล Olive Oil คือ ข้อมูลสเปคโทรแกรมทางด้านอาหาร เช่นเดียวกับข้อมูล Coffee แต่ข้อมูลนี้วิเคราะห์ความแตกต่างระหว่างน้ำมันมะกอกบริสุทธิ์จากแต่ละประเทศ ในที่นี้ประกอบด้วย กรีซ อิตาลี โปรตุเกส และ สเปน ซึ่งล้วนแต่เป็นประเทศที่มีชื่อเสียงด้านการผลิตน้ำมันมะกอกบริสุทธิ์ที่มีคุณภาพสูง ทำให้ได้ข้อมูลออกมาสี่กลุ่ม ตามลำดับ (ดังภาพที่ 4.12) ซึ่งข้อมูลชุดนี้มีความแตกต่างของข้อมูลน้อยมาก ๆ แม้กระทั่งนำตัวอย่างจากแต่ละกลุ่มมาซ้อนทับกันให้เห็น (ดังภาพที่ 4.13) ยังยากที่จะเห็นความแตกต่าง

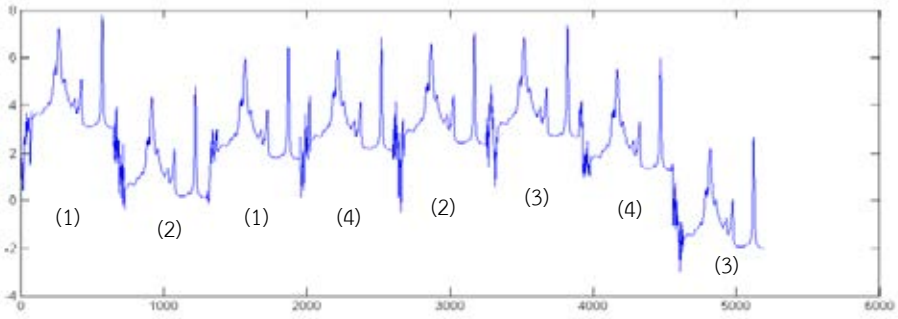


ภาพที่ 4.12 ความแตกต่างระหว่างข้อมูล Olive Oil ทั้งสี่กลุ่ม (บนซ้าย) ข้อมูลกลุ่มที่ 1 น้ำมันมะกอกบริสุทธิ์จากกรีซ (บนขวา) ข้อมูลกลุ่มที่ 2 น้ำมันมะกอกบริสุทธิ์จากอิตาลี (ล่างซ้าย) ข้อมูลกลุ่มที่ 3 น้ำมันมะกอกบริสุทธิ์จากโปรตุเกส และ (ล่างบน) ข้อมูลกลุ่มที่ 4 น้ำมันมะกอกบริสุทธิ์จากสเปน



ภาพที่ 4.13 ภาพเมื่อซ้อนทับระหว่างตัวอย่างข้อมูลในแต่ละกลุ่มของข้อมูล Olive Oil ทั้ง 4 กลุ่ม

ข้อมูลทุกกลุ่มในชุดนี้มีความยาว 570 จุดข้อมูล สุ่มเลือกตัวอย่างมาจากแต่ละกลุ่ม กลุ่มละ 2 ตัวอย่าง คั่นกลางด้วยข้อมูลแบบสุ่มความยาว 80 จุดข้อมูล ได้ผลลัพธ์เป็นข้อมูลอนุกรมนำเข้าหนึ่งชุด ความยาวรวม 5200 จุดข้อมูล (ดังภาพที่ 4.14) และแสดงรายละเอียดของข้อมูล ดังตารางที่ 4.13

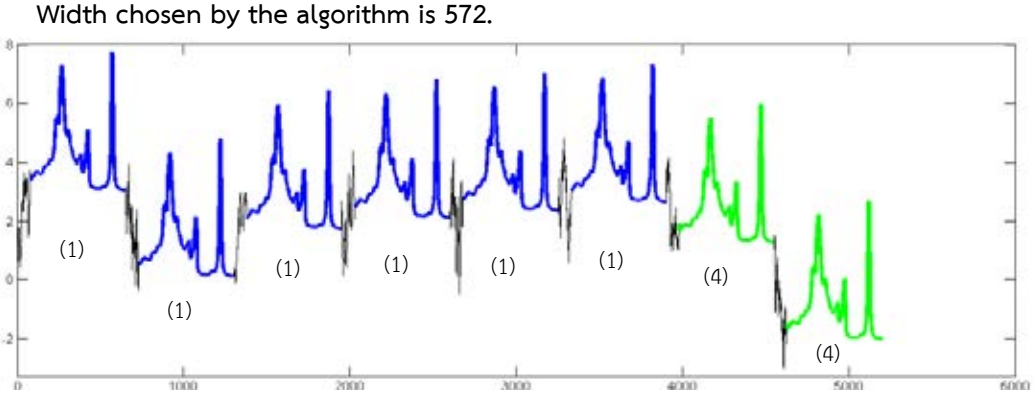


ภาพที่ 4.14 ข้อมูลอนุกรมเวลานำเข้าชุดที่ 4 จากข้อมูล Olive Oil คั่นกลางด้วยข้อมูลแบบสุ่ม

ตารางที่ 4.13 รายละเอียดของข้อมูลอนุกรมเวลานำเข้าชุดที่ 4

Example No.	1	2	3	4	5	6	7	8
Input Index	1(80)	2(730)	1(1380)	4(2030)	2(2680)	3(3330)	4(3980)	3(4630)

เมื่อทำการทดลองกับข้อมูลนำเข้าชุดนี้ด้วยอัลกอริทึมที่นำเสนอ ได้ผลลัพธ์ดังภาพที่ 4.15 ด้วยค่าความยาวที่อัลกอริทึมเลือก คือ 572 และรายละเอียดของผลลัพธ์เปรียบเทียบกับอัลกอริทึมก่อนหน้าเมื่อกำหนดให้ค่าความยาวในการจัดกลุ่มเป็น 570 แสดงดังตารางที่ 4.14



ภาพที่ 4.15 ผลลัพธ์ของการจัดกลุ่มข้อมูลอนุกรมเวลานำเข้าชุดที่ 4
ด้วยอัลกอริทึม PFSTS Clustering

ตารางที่ 4.14 รายละเอียดของผลลัพธ์จากการทดลองสำหรับข้อมูลอนุกรมเวลานำเข้าชุดที่ 4

Example No.	1	2	3	4	5	6	7	8
PFSTS Clustering	1(79)	1(729)	1(1379)	1(2029)	1(2679)	1(3329)	4(3978)	4(4628)
SSTS Clustering	1(80)	2(731)	1(1380)	1(2030)	2(2681)	1(3330)	1(3980)	1(4630)
MDL Clustering	1(81)	1(731)	1(1381)	1(2030)	1(2681)	1(3331)	1(3980)	1(4630)

ในข้อมูลอนุกรมเวลาชุดที่ 4 นี้ ถูกยกตัวอย่างขึ้นมาเป็นกรณพิเศษเพื่อให้เห็นถึงปัญหาในกรณีที่ค่าความยาวในการจัดกลุ่มส่งผลต่อผลลัพธ์ของการจัดกลุ่ม (มีผลต่อค่า ARI และ F_1 และส่งผลต่อค่า AoD ในปริมาณหนึ่ง) หากสังเกตจากค่าผลลัพธ์ (ดังตารางที่ 4.15) จะเห็นว่าความแม่นยำของทั้งสามอัลกอริทึมค่อนข้างแย่มาก และอัลกอริทึมที่นำเสนอยังมีความแม่นยำต่ำกว่า SSTS Clustering อีกด้วย ทั้งนี้ปัญหาหลักในกรณีนี้ไม่ได้ขึ้นอยู่กับจุดสิ้นสุดของการจัดกลุ่มเพียงอย่างเดียวเช่นที่ผ่านมา แต่ขึ้นอยู่กับมาตรวัดที่ทั้งสามอัลกอริทึมใช้ในการแยกแยะความแตกต่างของข้อมูล (ระยะทางยุคลิด) ซึ่งไม่สามารถแยกแยะความต่างของข้อมูลในแต่ละกลุ่มได้ (ในข้อมูลชุดนี้เองมีงานวิจัยที่ศึกษาโดยเฉพาะเกี่ยวกับการจำแนกประเภทของข้อมูลนี้โดยใช้เทคนิคการประมวลผลสัญญาณ (Signal Processing) และ สถิติศาสตร์ (Statistics) [14]) เนื่องจากข้อมูลทั้งสี่กลุ่มมีความคล้ายคลึงกันมาก ๆ และมีความอ่อนไหวต่อค่าผิดพลาดมาก ดังนั้นเพียงค่าความยาวที่คลาดเคลื่อนเพียงเล็กน้อยก็สามารถส่งผลต่อความสามารถในการจัดกลุ่มได้ จึงทำให้ผลลัพธ์ของอัลกอริทึมที่นำเสนอมีความแม่นยำน้อยกว่า SSTS Clustering ส่วน MDL Clustering นั้นมีปัญหาเรื่องจุดสิ้นสุดของการจัดกลุ่มมาเป็นปัจจัยด้วยจึงทำให้ผลลัพธ์ยิ่งแย่ลงไปอีก ดังที่เห็นว่า MDL Clustering จัดให้ลำดับย่อยทุกชุดอยู่ในกลุ่มเดียวกันทั้งหมด

ตารางที่ 4.15 เปรียบเทียบค่า Rand Index, AoD และ F_1 ของแต่ละอัลกอริทึม
สำหรับข้อมูลอนุกรมเวลานำเข้าชุดที่ 4

Measurement / Algorithm	RI	AoD	F_1
PFSTS Clustering	0.43	37.45%	0.38
SSTS Clustering	**0.57**	**49.93%**	**0.50**
MDL Clustering	0.14	24.95%	0.25

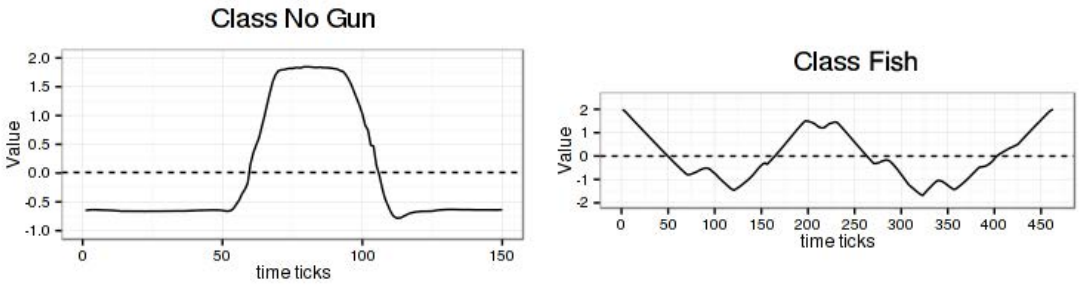
จากการทดลองทั้งหมดนี้ สิ่งที่เห็นเด่นชัดที่สุดคือความสามารถในการเลือกค่าความยาวที่เหมาะสมของอัลกอริทึมที่นำเสนอ ซึ่งสามารถเลือกได้ใกล้เคียงค่าความยาวจริงมาก มีความคลาดเคลื่อนเพียงเล็กน้อยเท่านั้น และโดยทั่วไปแล้วค่าที่คลาดเคลื่อนเพียงเล็กน้อยนี้จะไม่ส่งผลต่อความสามารถในการจัดกลุ่มอย่างร้ายแรง ยกเว้นในข้อมูลชุดสุดท้ายที่นำเสนอ ซึ่งเป็นกรณีตัวอย่างที่ยกขึ้นมาเป็นพิเศษ เพื่อให้เห็นถึงข้อมูลชุดที่มีความอ่อนไหวต่อค่าความผิดพลาดมาก ๆ ฉะนั้นผลลัพธ์ที่ได้โดยรวมจะใกล้เคียงกับ SSTS Clustering เพราะมีรากฐานในการจัดกลุ่มและจุดสิ้นสุดของการจัดกลุ่มเดียวกัน ดังนั้นหากในกรณีที่ไม่ทราบค่าความยาวที่ควรกำหนดให้ในการจัดกลุ่ม อัลกอริทึมที่นำเสนอนี้จะเป็นตัวเลือกที่สำคัญกว่าอัลกอริทึมทั้งหมดที่กล่าวถึง แต่สิ่งสำคัญอีกสิ่งหนึ่งที่สังเกตได้จากผลการทดลอง คือ ปัจจัยสำคัญที่ส่งผลอย่างมากต่อความแม่นยำในการจัดกลุ่ม คือ จุดสิ้นสุดของการจัดกลุ่ม ดังที่เห็นในผลลัพธ์ของแต่ละชุดข้อมูล ว่าเพียงจุดสิ้นสุดการจัดกลุ่มที่พลาดไปหนึ่งขั้นตอน ก็สามารถส่งผลต่อความแม่นยำในการจัดกลุ่มในปริมาณที่มากได้

4.2.2 ข้อมูลประกอบด้วยด้วยลำดับย่อยความยาวแตกต่างกัน

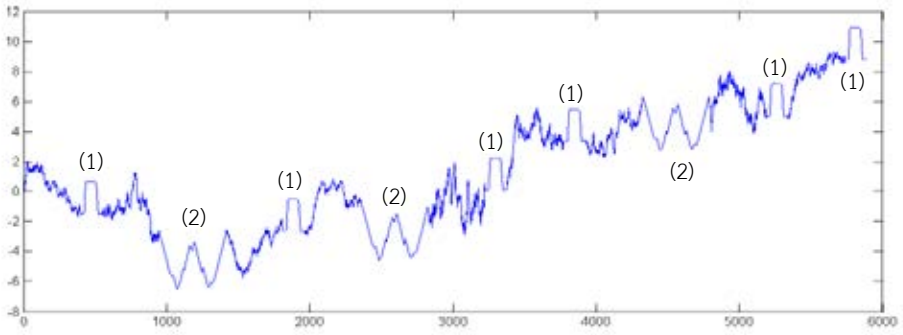
การทดลองนี้ข้อมูลอนุกรมเวลานำเข้าที่นำมาวิเคราะห์ประกอบด้วยลำดับย่อยหลากหลายกลุ่ม ซึ่งลำดับย่อยในแต่ละกลุ่มไม่จำเป็นต้องมีความยาวที่เท่ากัน และสามารถมีความยาวแตกต่างกันมากเพียงใดก็ได้ โดยไม่มีข้อจำกัดใด ๆ ทั้งสิ้น ดังนั้นอัลกอริทึมทั้งสองอัลกอริทึมที่นำมาเปรียบเทียบในหัวข้อก่อนหน้า ทั้ง SSTS Clustering และ MDL Clustering จึงไม่สามารถทำการจัดกลุ่มให้กับข้อมูลเหล่านี้ได้ ด้วยขีดจำกัดของพารามิเตอร์ที่กำหนด การเปรียบเทียบจึงนำเสนอโดยการ

ประมวลผลอัลกอริทึมดังกล่าวนี้ในแต่ละค่าความยาวที่แตกต่างกันของลำดับย่อยที่ปรากฏในข้อมูลอนุกรมเวลาที่พิจารณาทีละค่าความยาว โดยกำหนดพารามิเตอร์เป็นค่าความยาวจริงของลำดับย่อยนั้นและประมวลผลจนครบทุกค่าความยาว (การประเมินผลจากเครื่องมือวัดต่าง ๆ จะประเมินแยกกันในแต่ละค่าความยาวด้วยเช่นกัน) เปรียบเทียบกับอัลกอริทึมที่นำเสนอซึ่งไม่ต้องกำหนดพารามิเตอร์ใด ๆ และสามารถจัดกลุ่มให้กับลำดับย่อยทั้งหมดนี้ได้ภายในการประมวลผลเพียงครั้งเดียว ซึ่งเป็นข้อได้เปรียบหลักของอัลกอริทึมที่นำเสนอ และเนื่องจากผลลัพธ์ในหัวข้อที่แล้วแสดงให้เห็นว่า SSTS Clustering สามารถให้ค่าความแม่นยำที่ดีกว่าในชุดข้อมูลส่วนใหญ่ การทดลองในหัวข้อนี้จึงขอนำเฉพาะผลลัพธ์ของ SSTS Clustering มาเปรียบเทียบเท่านั้น เพื่อความสะดวกในการอ่านและการนำเสนอ โดยข้อมูลที่นำมาทำการทดลองมีดังนี้

1. ข้อมูลอนุกรมเวลานำเข้าชุดที่ 5 ความยาวรวม 5889 จุดข้อมูล (ดังภาพที่ 4.17) ประกอบด้วย ลำดับย่อยจาก 2 กลุ่ม (ดังภาพที่ 4.16) คือ ลำดับย่อยหนึ่งกลุ่มจากข้อมูล Gun-Point ซึ่งมีความยาว 150 จุดข้อมูล จำนวน 6 ตัวอย่าง และ ลำดับย่อยอีกหนึ่งกลุ่มจากข้อมูล Fish ซึ่งเป็นข้อมูลจากคอนทัวร์ (Contour) ของปลาแต่ละสายพันธุ์ ความยาว 463 จุดข้อมูล จำนวน 3 ตัวอย่าง คั่นกลางด้วยข้อมูลแบบสุ่มความยาว 400 จุดข้อมูล รายละเอียดของข้อมูลแสดงดังตารางที่ 4.16



ภาพที่ 4.16 ความแตกต่างระหว่างข้อมูลแต่ละกลุ่มในข้อมูลอนุกรมเวลานำเข้าชุดที่ 5 (ซ้าย) กลุ่มที่ 1 ข้อมูลจากข้อมูล Gun-Point และ (ขวา) กลุ่มที่ 2 ข้อมูลจากข้อมูล Fish

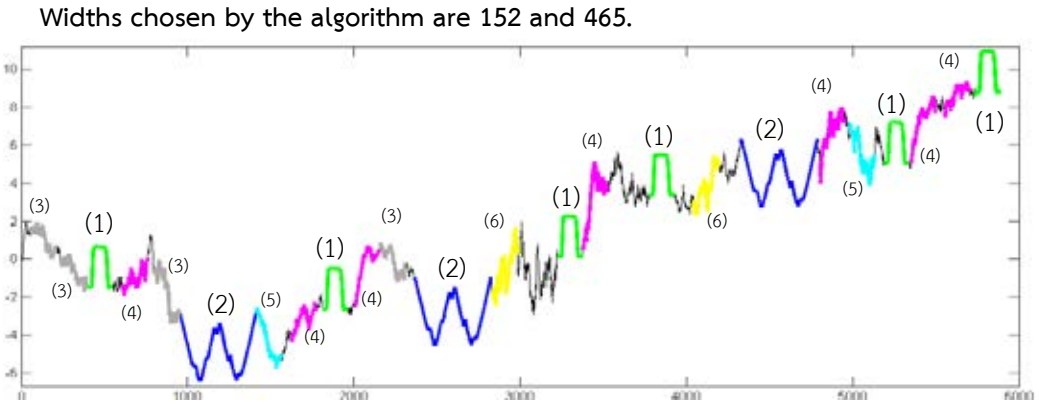


ภาพที่ 4.17 ข้อมูลอนุกรมเวลานำเข้าชุดที่ 5 จากข้อมูล Gun-Point 1 กลุ่ม และ Fish 1 กลุ่ม คั่นกลางด้วยข้อมูลแบบสุ่ม

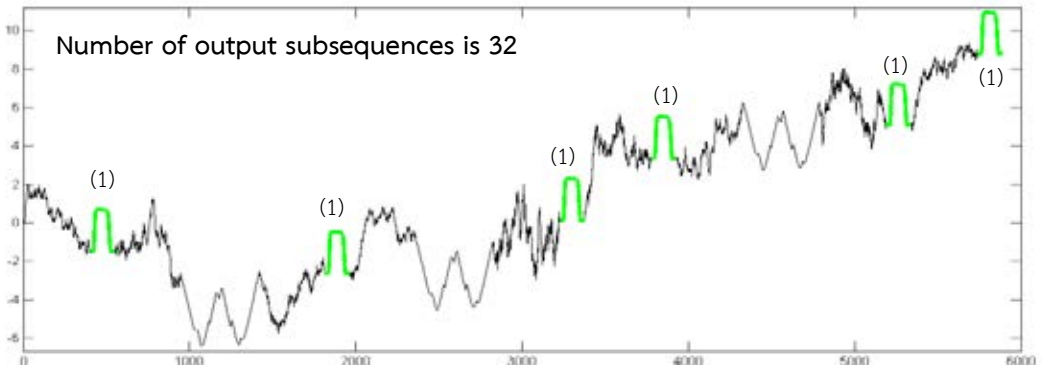
ตารางที่ 4.16 รายละเอียดของข้อมูลอนุกรมเวลานำเข้าชุดที่ 5

Example No.	1	2	3	4	5	6	7	8	9
Input Index	1(400)	2(950)	1(1813)	2(2363)	1(3226)	1(3776)	2(4326)	1(5189)	1(5739)

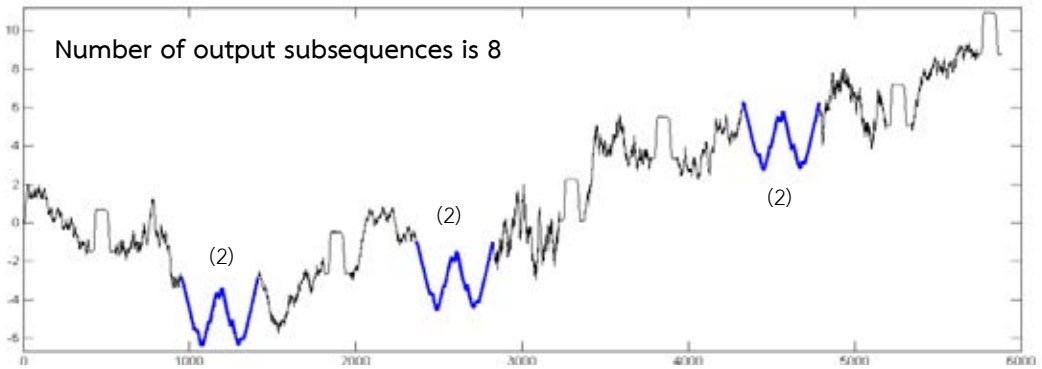
ผลลัพธ์ของอัลกอริทึมที่นำเสนอสำหรับข้อมูลนำเข้าชุดนี้ แสดงดังภาพที่ 4.18 ด้วยค่าความยาวที่อัลกอริทึมเลือก คือ 152 และ 465 ในขณะที่ผลลัพธ์ของ SSTS Clustering เมื่อกำหนดให้ค่าความยาวของการจัดกลุ่มเป็น 150 และ 463 แสดงดังภาพที่ 4.19 และ 4.20 ตามลำดับ และรายละเอียดของการจัดกลุ่มทั้งหมดแสดงดังตารางที่ 4.17



ภาพที่ 4.18 ผลลัพธ์ของการจัดกลุ่มข้อมูลอนุกรมเวลานำเข้าชุดที่ 5 ด้วยอัลกอริทึม PFSTS Clustering



ภาพที่ 4.19 ผลลัพธ์ของการจัดกลุ่มข้อมูลอนุกรมเวลานำเข้าชุดที่ 5
ด้วยอัลกอริทึม SSTS Clustering เมื่อกำหนดให้ค่าความยาวของการจัดกลุ่มเป็น 150



ภาพที่ 4.20 ผลลัพธ์ของการจัดกลุ่มข้อมูลอนุกรมเวลานำเข้าชุดที่ 5
ด้วยอัลกอริทึม SSTS Clustering เมื่อกำหนดให้ค่าความยาวของการจัดกลุ่มเป็น 463

ตารางที่ 4.17 รายละเอียดของผลลัพธ์จากการทดลองสำหรับข้อมูลอนุกรมเวลานำเข้าชุดที่ 5

Example No.	1	2	3	4	5	6	7	8	9
PFSTS Clustering	1(399)	2(949)	1(1812)	2(2362)	1(3225)	1(3775)	2(4325)	1(5188)	1(5737)
(*w*=150) SSTS Clustering	1(401)	------	1(1814)	------	1(3227)	1(3775)	------	1(5188)	1(5738)
(*w*=463) SSTS Clustering	------	2(949)	------	2(2362)	------	------	2(4325)	------	------

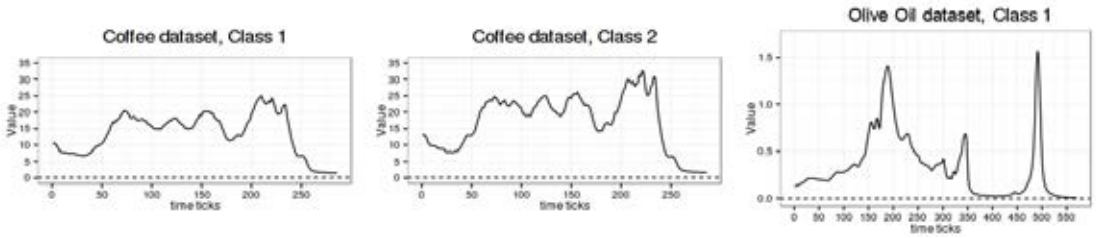
จากข้อมูลอนุกรมนำเข้าชุดที่ 5 นี้ จะเห็นว่าความยาวในการจัดกลุ่มที่อัลกอริทึมเลือกยังคงถูกต้องใกล้เคียงกับค่าความยาวจริง ส่งผลให้ผลลัพธ์ของอัลกอริทึมที่นำเสนอถูกต้องครบถ้วน (SSTS Clustering ไม่สามารถให้ผลลัพธ์เช่นนี้ได้ภายในการประมวลผลเพียงครั้งเดียว) เว้นแต่ว่ามีผลลัพธ์ที่ไม่ได้คาดหวังเกินมา จึงส่งผลให้ค่า F_1 ค่อนข้างต่ำ ในขณะที่ค่า AoD และ RI สูงมาก เพราะไม่มีการคำนึงถึงผลลัพธ์ที่เกินมาเหล่านี้ (ดังตารางที่ 4.18) โดยผลลัพธ์เหล่านี้เกิดจากความยาวของข้อมูลแบบสุ่มซึ่งมีขนาดยาวมากสำหรับในข้อมูลชุดนี้ (400 จุดข้อมูล จำนวน 9 ตำแหน่ง) อัลกอริทึมจึงพยายามจัดกลุ่มให้กับข้อมูลเหล่านี้ด้วยค่าความยาวที่สามารถจัดกลุ่มได้ คือ 152 โดยพยายามจัดให้คล้ายคลึงกันที่สุดเท่าที่จะทำได้ ปัญหานี้เกิดขึ้นเช่นเดียวกันใน SSTS Clustering ยิ่งกำหนดค่าความยาวของการจัดกลุ่มสั้นเพียงใดก็ยิ่งมีโอกาสเพิ่มจำนวนผลลัพธ์ที่เกินมามากขึ้น ทั้งนี้ผลลัพธ์ที่เกินมามีจำนวนมากเสียจนไม่อาจแสดงให้เห็นในภาพได้ เพื่อความสะดวกจึงแสดงเฉพาะลำดับย่อยในกลุ่มที่คาดหวังไว้เพียงเท่านั้น (เฉพาะในกรณีของ SSTS Clustering) โดยค่า F_1 จะเป็นสิ่งบ่งบอกว่าผลลัพธ์ที่เกินมานั้นมีมากน้อยเพียงใด (ยิ่งค่า F_1 ต่ำ จำนวนผลลัพธ์ที่เกินมายิ่งมาก)

ตารางที่ 4.18 เปรียบเทียบค่า Rand Index, AoD และ F_1 ของแต่ละอัลกอริทึม
สำหรับข้อมูลอนุกรมเวลานำเข้าชุดที่ 5

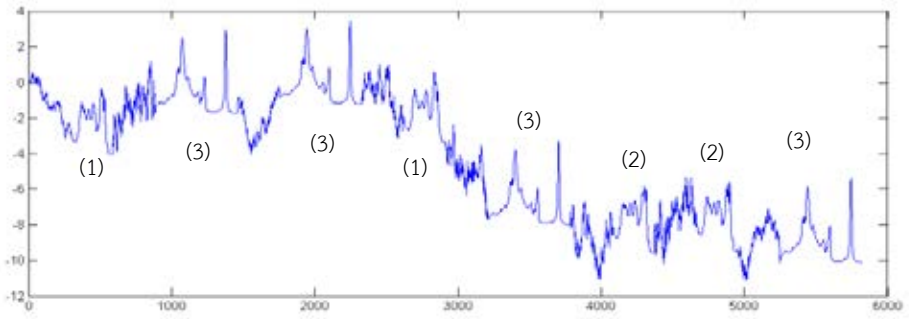
Measurement / Algorithm	RI	AoD	F_1
PFSTS Clustering	**1.00**	99.22%	**0.55**
(*w*=150) SSTS Clustering	**1.00**	98.68%	0.52
(*w*=463) SSTS Clustering	**1.00**	**99.57%**	**0.55**

2. ข้อมูลอนุกรมเวลานำเข้าชุดที่ 6 ความยาวรวม 5824 จุดข้อมูล (ดังภาพที่ 4.22) ประกอบด้วย ลำดับย่อยจาก 3 กลุ่ม (ดังภาพที่ 4.21) คือ ลำดับย่อยสองกลุ่มจากข้อมูล Coffee ความยาว 286 จุดข้อมูล จำนวนกลุ่มละ 2 ตัวอย่าง และ ลำดับย่อยอีกหนึ่งกลุ่มจากข้อมูล Olive Oil

ความยาว 570 จุดข้อมูล จำนวน 4 ตัวอย่าง คั่นกลางด้วยข้อมูลแบบสุ่มความยาว 300 จุดข้อมูล รายละเอียดของข้อมูลแสดงดังตารางที่ 4.19



ภาพที่ 4.21 ความแตกต่างระหว่างข้อมูลแต่ละกลุ่มในข้อมูลอนุกรมเวลานำเข้าชุดที่ 6 (ซ้าย, กลาง) ข้อมูลกลุ่มที่ 1 และ 2 จากข้อมูล Coffee ในกลุ่มที่ 1 และ 2 และ (ขวา) ข้อมูลกลุ่มที่ 3 จากข้อมูล Olive Oil ในกลุ่มที่ 1

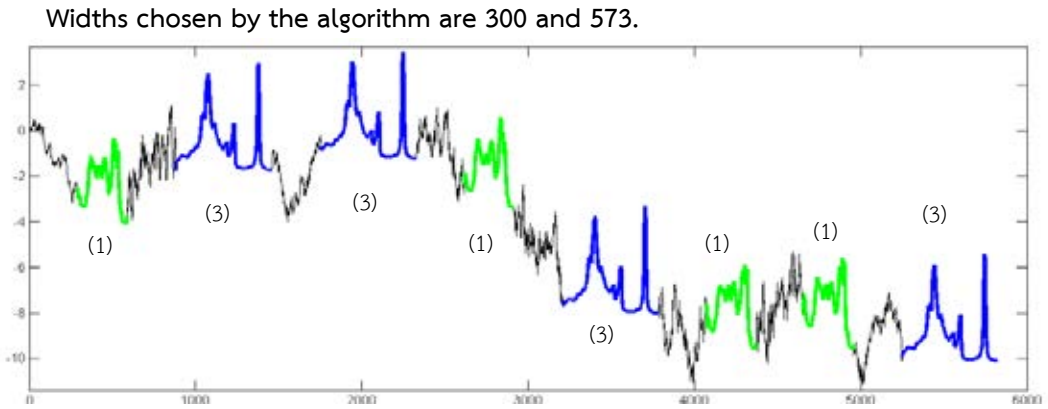


ภาพที่ 4.22 ข้อมูลอนุกรมเวลานำเข้าชุดที่ 6 จากข้อมูล Coffee 2 กลุ่ม และ Olive Oil 1 กลุ่ม คั่นกลางด้วยข้อมูลแบบสุ่ม

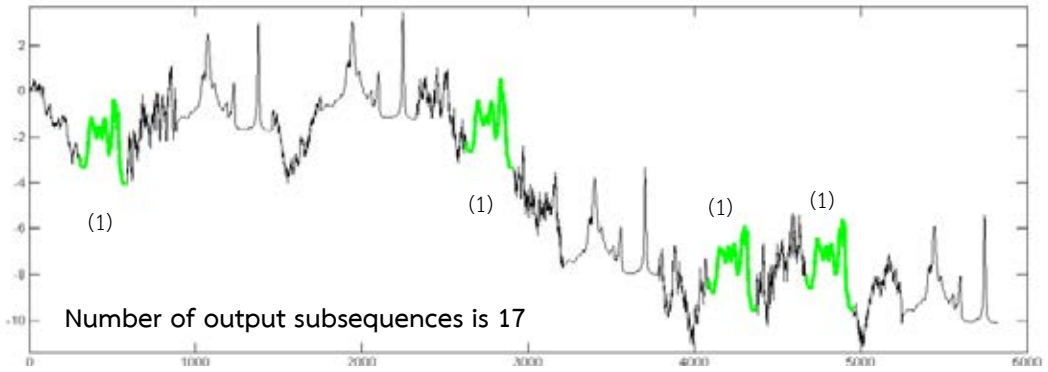
ตารางที่ 4.19 รายละเอียดของข้อมูลอนุกรมเวลานำเข้าชุดที่ 6

Example No.	1	2	3	4	5	6	7	8
Input Index	1(300)	3(886)	3(1756)	1(2626)	3(3212)	2(4082)	2(4668)	3(5254)

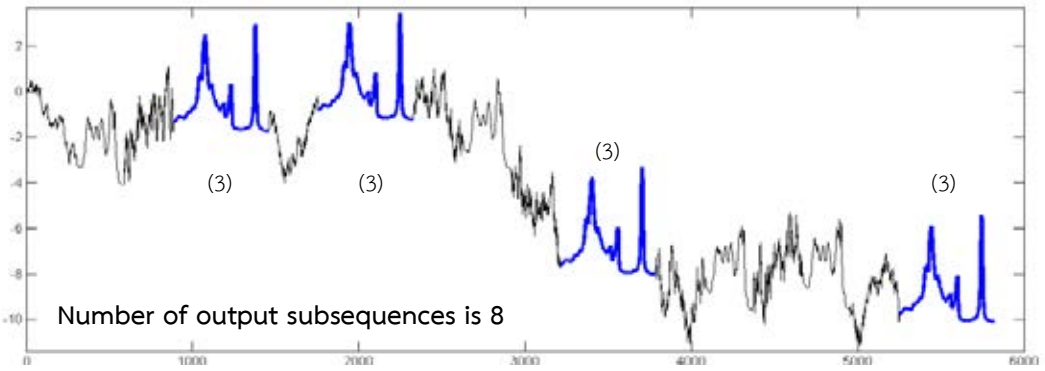
ผลลัพธ์ของอัลกอริทึมที่นำเสนอสำหรับข้อมูลนำเข้าชุดนี้ แสดงดังภาพที่ 4.23 ด้วยค่าความยาวที่อัลกอริทึมเลือก คือ 300 และ 573 ในขณะที่ผลลัพธ์ของ SSTS Clustering เมื่อกำหนดให้ค่าความยาวของการจัดกลุ่มเป็น 286 และ 570 แสดงดังภาพที่ 4.24 และ 4.25 ตามลำดับ และรายละเอียดของการจัดกลุ่มทั้งหมดแสดงดังตารางที่ 4.20



ภาพที่ 4.23 ผลลัพธ์ของการจัดกลุ่มข้อมูลอนุกรมเวลานำเข้าชุดที่ 6
ด้วยอัลกอริทึม PFSTS Clustering



ภาพที่ 4.24 ผลลัพธ์ของการจัดกลุ่มข้อมูลอนุกรมเวลานำเข้าชุดที่ 6
ด้วยอัลกอริทึม SSTS Clustering เมื่อกำหนดให้ค่าความยาวของการจัดกลุ่มเป็น 286



ภาพที่ 4.25 ผลลัพธ์ของการจัดกลุ่มข้อมูลอนุกรมเวลานำเข้าชุดที่ 6
ด้วยอัลกอริทึม SSTS Clustering เมื่อกำหนดให้ค่าความยาวของการจัดกลุ่มเป็น 570

ตารางที่ 4.20 รายละเอียดของผลลัพธ์จากการทดลองสำหรับข้อมูลอนุกรมเวลานำเข้าชุดที่ 6

Example No.	1	2	3	4	5	6	7	8
PFSTS Clustering	1(288)	3(883)	3(1754)	1(2614)	3(3210)	1(4070)	1(4656)	3(5251)
(*w*=286) SSTS Clustering	1(301)	--------	--------	1(2627)	--------	1(4084)	1(4670)	--------
(*w*=570) SSTS Clustering	--------	3(887)	3(1757)	--------	3(3213)	--------	--------	3(5254)

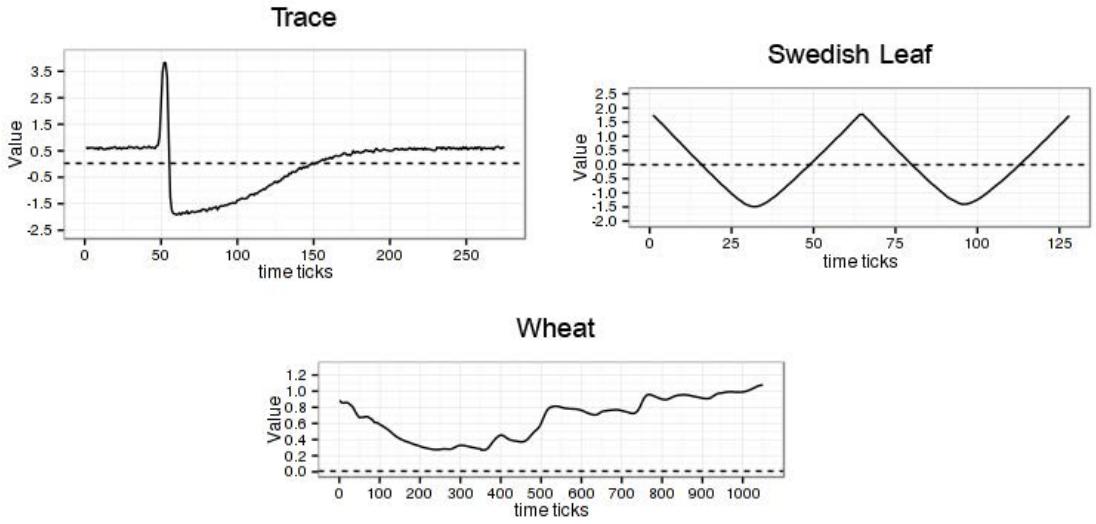
จากข้อมูลอนุกรมนำเข้าชุดที่ 6 นี้ ความยาวในการจัดกลุ่มที่อัลกอริทึมเลือกนี้มีความคลาดเคลื่อนมากขึ้นกว่าเดิม แต่ยังอยู่ในเกณฑ์ที่กำหนดและไม่ส่งผลต่อความสามารถในการจัดกลุ่ม ดังที่เห็นจากผลลัพธ์เมื่อเปรียบเทียบกับ SSTS Clustering ข้อมูลชุดนี้อัลกอริทึมที่นำเสนอสามารถจัดกลุ่มได้อย่างครบถ้วนและไม่มีข้อมูลที่เหนือความคาดหมายเกินมา เพราะความยาวของข้อมูลแบบสุ่มคือ 300 จุดข้อมูล เทียบกับความยาวที่สั้นที่สุดที่อัลกอริทึมเลือกคือ 286 จุดข้อมูล จึงเป็นไปได้ยากที่จะมีข้อมูลแบบสุ่มความยาว 286 จุดข้อมูล ที่คล้ายคลึงกัน และเหตุการณ์ดังกล่าวไม่เกิดขึ้นสำหรับข้อมูลชุดนี้ แต่ความผิดพลาดของอัลกอริทึมสำหรับข้อมูลชุดนี้คือไม่สามารถแยกข้อมูล Coffee สองกลุ่มออกจากกันได้จึงทำให้ค่าความแม่นยำจากเครื่องมือวัดต่าง ๆ ไม่สูงเท่าที่ควร (ดัง

ตารางที่ 4.21) ทั้งนี้สาเหตุมาจากการกำหนดจุดสิ้นสุดของการจัดกลุ่มที่ผิดพลาดไปจากจุดที่ควรจะเป็นจริง ๆ หนึ่งขั้นตอน เนื่องมาจากการวิเคราะห์จุดสิ้นสุดนั้นคำนึงถึงค่าผิดพลาดของทั้งระบบการจัดกลุ่มเป็นหลัก การรวมกลุ่มที่ 1 และ กลุ่มที่ 2 เข้าเป็นกลุ่มเดียวกันอาจให้ค่าผิดพลาดที่สูง แต่เมื่อเทียบกับการรวมกลุ่มที่เป็นผลลัพธ์จากการรวมกลุ่มที่ 1 และกลุ่มที่ 2 ในขั้นตอนที่แล้วกับกลุ่มที่ 3 ถือว่าเป็นค่าที่น้อยกว่ามาก ๆ เมื่อคำนึงถึงค่าผิดพลาดที่สูงขึ้นผิดปกติในขั้นตอนนี้ อัลกอริทึมจึงเลือกจุดนี้เป็นจุดสิ้นสุดของการจัดกลุ่มแทน ซึ่งปัญหานี้เกิดขึ้นเช่นกันใน SSTS Clustering นอกจากนี้ยังประสบปัญหาในเรื่องของผลลัพธ์ที่เกินมา จึงทำให้ค่า F_1 ของ SSTS Clustering ค่อนข้างต่ำ

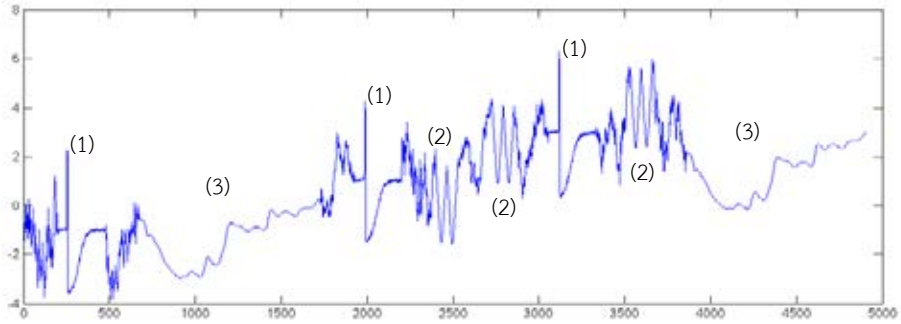
ตารางที่ 4.21 เปรียบเทียบค่า Rand Index, AoD และ F_1 ของแต่ละอัลกอริทึม
สำหรับข้อมูลอนุกรมเวลานำเข้าชุดที่ 6

Measurement / Algorithm	RI	AoD	F_1
PFSTS Clustering	0.86	82.33%	**0.75**
(*w*=286) SSTS Clustering	0.33	49.74%	0.35
(*w*=570) SSTS Clustering	**1.00**	**99.74%**	0.67

3. ข้อมูลอนุกรมเวลานำเข้าชุดที่ 7 ความยาวรวม 4909 จุดข้อมูล (ดังภาพที่ 4.27) ประกอบด้วย ลำดับย่อยจาก 3 กลุ่ม (ดังภาพที่ 4.26) คือ ลำดับย่อยหนึ่งกลุ่มจากข้อมูล Trace ซึ่งเป็นข้อมูลที่อ่านจากตัวรับรู้ (Sensor) ในโรงงานอุตสาหกรรม ความยาว 275 จุดข้อมูล จำนวน 3 ตัวอย่าง ลำดับย่อยอีกหนึ่งกลุ่มจากข้อมูล Swedish Leaf คือ ข้อมูลจากคอนทัวร์ของใบไม้สวีเดนในแต่ละสายพันธุ์ ความยาว 128 จุดข้อมูล จำนวน 3 ตัวอย่าง และ ลำดับย่อยกลุ่มสุดท้ายจากข้อมูล Wheat คือ ข้อมูลสเปคโทรแกรมของข้าวสาลีที่เจริญเติบโตในแคนาดาในแต่ละฤดู ความยาว 1050 จุดข้อมูล จำนวน 2 ตัวอย่าง คั่นกลางด้วยข้อมูลแบบสุ่มความยาว 200 จุดข้อมูล รายละเอียดของข้อมูลแสดงดังตารางที่ 4.22



ภาพที่ 4.26 ความแตกต่างระหว่างข้อมูลแต่ละกลุ่มในข้อมูลอนุกรมเวลานำเข้าชุดที่ 7 (บนซ้าย) ข้อมูลกลุ่มที่ 1 จากข้อมูล Trace (บนขวา) ข้อมูลกลุ่มที่ 2 จากข้อมูล Swedish Leaf และ (ล่าง) ข้อมูลกลุ่มที่ 3 จากข้อมูล Wheat

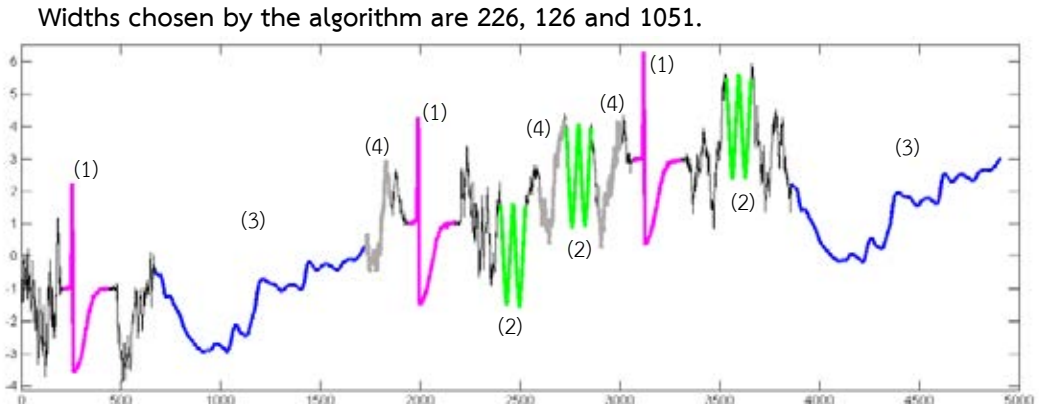


ภาพที่ 4.27 ข้อมูลอนุกรมเวลานำเข้าชุดที่ 7 จากข้อมูล Trace 1 กลุ่ม Wheat 1 กลุ่ม และ Swedish Leaf 1 กลุ่ม คั่นกลางด้วยข้อมูลแบบสุ่ม

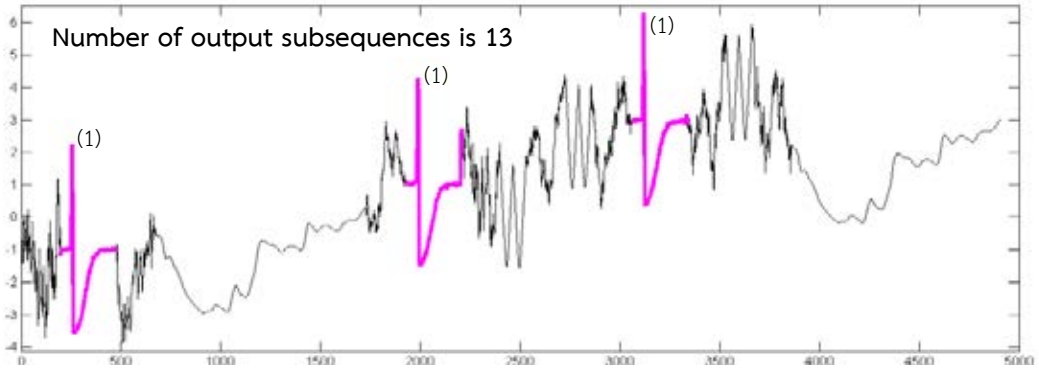
ตารางที่ 4.22 รายละเอียดของข้อมูลอนุกรมเวลานำเข้าชุดที่ 7

Example No.	1	2	3	4	5	6	7	8
Input Index	1(200)	3(675)	1(1925)	2(2400)	2(2728)	1(3056)	2(3531)	3(3859)

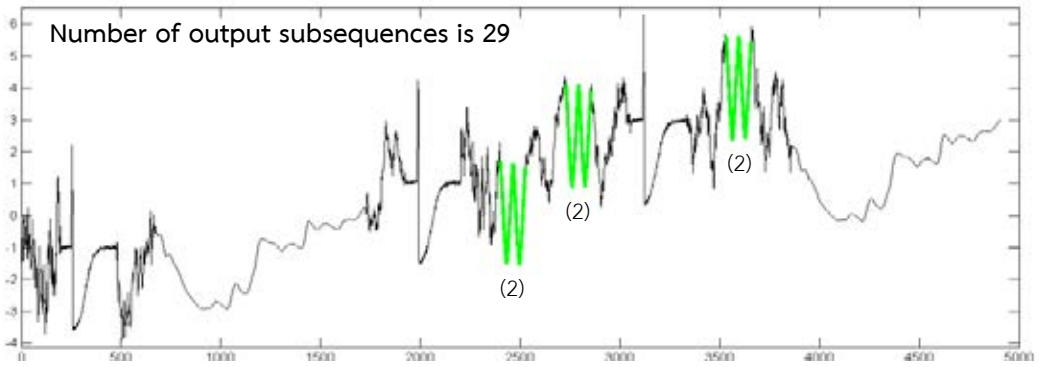
ผลลัพธ์ของอัลกอริทึมที่นำเสนอสำหรับข้อมูลนำเข้าชุดนี้ แสดงดังภาพที่ 4.28 ด้วยค่าความยาวที่อัลกอริทึมเลือก คือ 226, 126 และ 1051 ในขณะที่ผลลัพธ์ของ SSTS Clustering เมื่อกำหนดให้ค่าความยาวของการจัดกลุ่มเป็น 275, 128 และ 1050 แสดงดังภาพที่ 4.29, 4.30 และ 4.31 ตามลำดับ และรายละเอียดของการจัดกลุ่มทั้งหมดแสดงดังตารางที่ 4.23



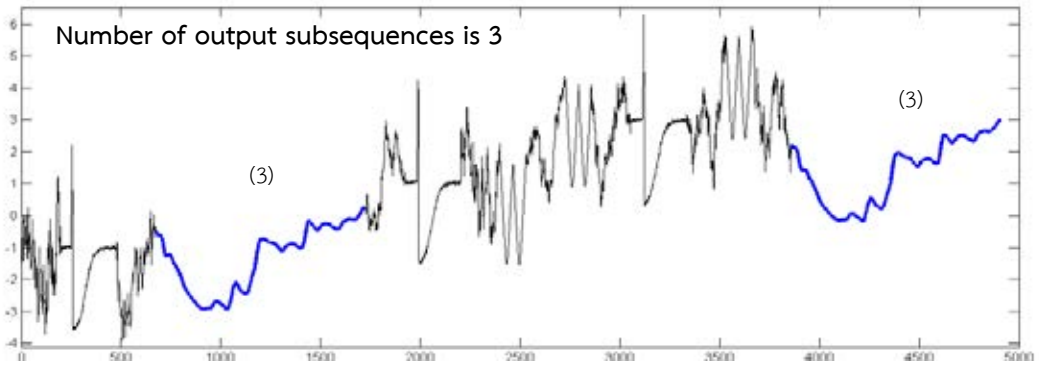
ภาพที่ 4.28 ผลลัพธ์ของการจัดกลุ่มข้อมูลอนุกรมเวลานำเข้าชุดที่ 7
ด้วยอัลกอริทึม PFSTS Clustering



ภาพที่ 4.29 ผลลัพธ์ของการจัดกลุ่มข้อมูลอนุกรมเวลานำเข้าชุดที่ 7
ด้วยอัลกอริทึม SSTS Clustering เมื่อกำหนดให้ค่าความยาวของการจัดกลุ่มเป็น 275



ภาพที่ 4.30 ผลลัพธ์ของการจัดกลุ่มข้อมูลอนุกรมเวลานำเข้าชุดที่ 7
ด้วยอัลกอริทึม SSTS Clustering เมื่อกำหนดให้ค่าความยาวของการจัดกลุ่มเป็น 128



ภาพที่ 4.31 ผลลัพธ์ของการจัดกลุ่มข้อมูลอนุกรมเวลานำเข้าชุดที่ 7
ด้วยอัลกอริทึม SSTS Clustering เมื่อกำหนดให้ค่าความยาวของการจัดกลุ่มเป็น 1050

ตารางที่ 4.23 รายละเอียดของผลลัพธ์จากการทดลองสำหรับข้อมูลอนุกรมเวลานำเข้าชุดที่ 7

Example No.	1	2	3	4	5	6	7	8
PFSTS Clustering	1(209)	3(673)	1(1943)	2(2401)	2(2729)	1(3073)	2(3532)	3(3858)
(*w*=275) SSTS Clustering	1(198)	--------	1(1932)	--------	--------	1(3062)	--------	--------
(*w*=128) SSTS Clustering	--------	--------	--------	2(2399)	2(2727)	--------	2(3530)	--------
(*w*=1050) SSTS Clustering	--------	3(674)	--------	--------	--------	--------	--------	3(3859)

ข้อมูลอนุกรมเวลานำเข้าชุดที่ 7 นี้ ประกอบด้วยความยาวของลำดับย่อยที่แตกต่างกัน 3 ค่า อีกทั้งความยาวระหว่างลำดับย่อยที่สั้นที่สุดกับลำดับย่อยที่ยาวที่สุดยังต่างกันถึงเกือบสิบเท่าตัว แต่อัลกอริทึมที่นำเสนอก็สามารถจัดกลุ่มให้กับลำดับย่อยทั้งหมดได้อย่างถูกต้องครบถ้วน ด้วยค่าความยาวที่ใกล้เคียงความยาวจริง ยกเว้นความยาวของข้อมูลกลุ่ม Trace ซึ่งคลาดเคลื่อนจากค่าจริงพอสมควรแต่ไม่เกินเกณฑ์ที่กำหนด จึงไม่มีผลต่อค่า RI แต่ส่งผลต่อค่า AoD และยังมีผลลัพธ์ที่เกินมาเล็กน้อย (ข้อมูลแบบสุ่มถูกจัดกลุ่ม 3 ชุด) จึงส่งผลให้ค่า F_1 ต่ำลง (ดังตารางที่ 4.24) แต่ยังคงมีค่าสูงที่สุดเมื่อเปรียบเทียบกับ SSTS Clustering

ตารางที่ 4.24 เปรียบเทียบค่า Rand Index, AoD และ F_1 ของแต่ละอัลกอริทึม
สำหรับข้อมูลอนุกรมเวลานำเข้าชุดที่ 7

Measurement / Algorithm	RI	AoD	F_1
PFSTS Clustering	**1.00**	95.26%	**0.84**
(*w*=275) SSTS Clustering	**1.00**	96.43%	0.64
(*w*=128) SSTS Clustering	**1.00**	98.45%	0.80
(*w*=1050) SSTS Clustering	**1.00**	**99.90%**	0.80

จากผลการทดลองทั้งหมดในหัวข้อนี้แสดงให้เห็นถึงข้อได้เปรียบอย่างชัดเจนของอัลกอริทึมที่นำเสนอเหนือทั้งสองอัลกอริทึมที่นำมาเปรียบเทียบ ทั้ง SSTS Clustering และ MDL Clustering ซึ่งไม่สามารถดำเนินการจัดกัดกลุ่มอย่างอิสระเช่นนี้ได้ ในขณะที่อัลกอริทึมที่นำเสนอสามารถเลือกความยาวที่เหมาะสม และดำเนินการจัดกลุ่มได้อย่างแม่นยำ ส่วนในประเด็นของผลลัพธ์ที่ไม่ได้คาดหวังนั้นโดยทั่วไปแล้วเกิดขึ้นได้ถ้าข้อมูลแบบสุ่มมีขนาดยาว เมื่อเทียบกับความยาวที่อัลกอริทึมเลือกในการจัดกลุ่ม หรืออีกแง่หนึ่งอาจมองว่าเป็นปัญหาเนื่องมาจากการกำหนดจุดสิ้นสุดของการจัดกลุ่มที่ไม่ถูกต้องก็เป็นได้ ทั้งนี้ปัญหาเรื่องจุดสิ้นสุดของการจัดกลุ่มเป็นปัญหาสำคัญของการจัดกลุ่มแบบลำดับชั้นอยู่แล้ว และส่งผลต่อผลลัพธ์ไม่ว่าข้อมูลจะประกอบด้วยลำดับย่อยความยาวเดียวกันหรือแตกต่าง

บทที่ 5

สรุปผลการวิจัย อภิปรายผลและข้อเสนอแนะ

งานวิจัยนี้เป็นงานวิจัยแรกที่มองเห็นปัญหาและนำเสนอออกมาในรูปแบบของการจัดกลุ่มลำดับย่อยโดยปราศจากพารามิเตอร์ เพื่อแก้ไขปัญหาของการจัดกลุ่มลำดับย่อยในปัจจุบันซึ่งต้องการพารามิเตอร์สำหรับกำหนดค่าความยาวในการจัดกลุ่มสองประการ คือ 1. ปัญหาเรื่องความยากในการกำหนดค่าความยาวที่เหมาะสมให้กับข้อมูลอนุกรมเวลาแต่ละชุด 2. ปัญหาเรื่องการถูกจำกัดของความยาวในการจัดกลุ่ม ซึ่งถูกจำกัดภายใต้พารามิเตอร์ที่กำหนดลงไป ทำให้ขาดอิสระในการจัดกลุ่มอย่างแท้จริง โดยวิธีการที่นำเสนอคือการประยุกต์ใช้อัลกอริทึมในการค้นพบโมทีฟโดยปราศจากพารามิเตอร์ที่มีอยู่ในปัจจุบันมาใช้เป็นเครื่องมือในการเลือกค่าความยาวที่เหมาะสมของการจัดกลุ่มในข้อมูลแต่ละชุด ซึ่งความยาวที่เลือกนี้อาจมีเพียงค่าเดียวหรือหลายค่าก็ได้ และแต่ละค่าสามารถแตกต่างกันมากเพียงใดก็ได้ โดยไร้ข้อจำกัด ทั้งนี้ได้ทำการทดลองเพื่อสนับสนุนแนวคิดดังกล่าว โดยทำการทดลองออกมาในสองกรณี คือ กรณีแรกให้ข้อมูลประกอบด้วยลำดับย่อยที่มีความยาวเท่ากัน และ กรณีที่สองให้ข้อมูลประกอบด้วยลำดับย่อยที่มีความยาวแตกต่างกัน ซึ่งผลการทดลองแสดงให้เห็นว่าสำหรับกรณีแรก อัลกอริทึมที่นำเสนอนี้สามารถให้ผลลัพธ์ของการจัดกลุ่มที่ใกล้เคียงกับการจัดกลุ่มด้วยอัลกอริทึมก่อนหน้านี้ที่ถึงแม้จะถูกกำหนดค่าความยาวในการจัดกลุ่มเป็นค่าความยาวจริงให้แล้วก็ตาม และสำหรับกรณีที่สอง อัลกอริทึมที่นำเสนอสามารถให้ผลลัพธ์ที่เหนือกว่าอัลกอริทึมที่นำมาเปรียบเทียบอย่างชัดเจน เพราะนอกจากจะไม่ต้องกำหนดค่าพารามิเตอร์ใด ๆ แล้ว ยังสามารถจัดกลุ่มให้กับลำดับย่อยในข้อมูลแต่ละชุดได้อย่างอิสระ (จัดกลุ่มได้ในหลากหลายความยาว โดยไม่มีข้อจำกัดระหว่างความยาวที่แตกต่างกัน) และแม่นยำ โดยที่อัลกอริทึมที่นำมาเปรียบเทียบนั้นไม่สามารถทำได้ กล่าวโดยสรุปคือ นอกจากอัลกอริทึมที่นำเสนอนี้จะสามารถทำการจัดกลุ่มให้กับลำดับย่อยได้โดยไม่ต้องมีการกำหนดค่าพารามิเตอร์ใด ๆ แล้ว ยังสามารถจัดกลุ่มให้กับลำดับย่อยที่มีความยาวแตกต่างกันได้อย่างแม่นยำอีกด้วย ซึ่งถือว่าประสบความสำเร็จและบรรลุตามจุดประสงค์ของงานวิจัยที่ได้กำหนดไว้ในตอนต้นทุกประการ โดยจะทำการสรุปและอภิปรายผลการวิจัย ก่อนที่จะกล่าวถึง ข้อจำกัดและข้อเสนอแนะสำหรับงานวิจัยนี้ต่อไปในหัวข้อที่ 5.1 และ 5.2 ต่อไปตามลำดับ

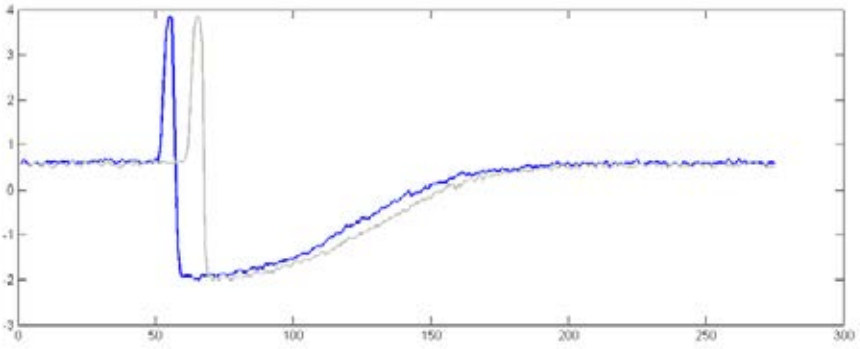
5.1 สรุปและอภิปรายผลการวิจัย

เป้าหมายหลักของงานวิจัยนี้คือการกำจัดขั้นตอนในการกำหนดพารามิเตอร์ของค่าความยาวในการจัดกลุ่มลำดับย่อยทิ้งไป บนสมมติฐานที่ว่าการกำจัดขั้นตอนนี้ไปจะนำไปสู่ผลลัพธ์ของการจัดกลุ่มที่อิสระมากขึ้นได้ โดยปล่อยให้ภาระในการเลือกความยาวที่เหมาะสมในการจัดกลุ่มเป็นของอัลกอริทึม ซึ่งสามารถเลือกค่าความยาวที่เหมาะสมสำหรับข้อมูลแต่ละชุดได้ แม้ข้อมูลจะประกอบด้วยลำดับย่อยที่มีความยาวเพียงค่าเดียวหรือหลากหลายค่าความยาวก็ตาม ทั้งนี้เทคนิคที่นำมาใช้ประกอบด้วย การค้นพบโมทีฟความยาวเหมาะสมสาหรับข้อมูลอนุกรมเวลา ในงานวิจัย [13] และกระบวนการขัดเกลาโมทีฟที่นำเสนอในงานวิจัยนี้ จากนั้นจึงทำการจัดกลุ่มด้วยอัลกอริทึมการจัดกลุ่มในงานวิจัยนี้ซึ่งปรับปรุงจากอัลกอริทึมที่มีในปัจจุบันให้สามารถจัดกลุ่มลำดับย่อยในหลากหลายค่าความยาวได้ โดยมีลักษณะเป็นการจัดกลุ่มแบบลำดับชั้น และพิจารณาจุดสิ้นสุดของการจัดกลุ่มจากการวิเคราะห์กราฟค่าความผิดพลาดของกลุ่ม และสมมติฐานนี้ก็ได้รับการสนันสนุนจากผลการทดลองว่าสามารถทำได้จริงด้วยผลลัพธ์ที่แม่นยำ แต่สิ่งที่สังเกตได้จากการทดลองทั้งหมดนี้คือสิ่งสำคัญซึ่งมีผลต่อความแม่นยำของการจัดกลุ่มลำดับย่อยมาก คือ การกำหนดจุดสิ้นสุดของการจัดกลุ่มของอัลกอริทึม ดังที่ได้กล่าวถึงในบทที่แล้วว่าเพียงการกำหนดจุดสิ้นสุดของการจัดกลุ่มที่ผิดพลาดไปหนึ่งขั้นตอน ก็สามารถส่งผลต่อความแม่นยำของผลลัพธ์ในปริมาณมาก และวิธีที่ใช้อยู่ในปัจจุบันยังไม่สามารถกำหนดจุดสิ้นสุดของการจัดกลุ่มที่ถูกต้องได้ดีเท่าที่ควร ทั้งนี้ปัญหาเรื่องการกำหนดจุดสิ้นสุดที่เหมาะสมเป็นปัญหาที่ยาก นับว่าเป็นปัญหาหลักและเป็นหัวใจสำคัญของการทำการจัดกลุ่มแบบลำดับชั้นมาตั้งแต่อดีตถึงปัจจุบัน และอยู่นอกเหนือขอบเขตของงานวิจัยนี้ จึงไม่ได้นำเสนอวิธีในการแก้ปัญหา

5.2 ข้อจำกัดและข้อเสนอแนะ

เนื่องจากงานวิจัยนี้เป็นงานวิจัยแรกที่นำเสนอวิธีการจัดกลุ่มลำดับย่อยของข้อมูลอนุกรมเวลาโดยปราศจากพารามิเตอร์ จึงเป็นการนำเสนอเพียงเฟรมเวิร์คของอัลกอริทึมในเชิงปราศจากพารามิเตอร์ ว่าสามารถทำได้จริง โดยพื้นฐานของการจัดกลุ่มถอดแบบมาจากอัลกอริทึมที่มีอยู่ในปัจจุบัน (Selective Subsequence Time Series Clustering [7]) ทั้งในเรื่องของมาตรวัดความคล้ายคลึงกันระหว่างลำดับย่อยซึ่งเป็นระยะทางยุคลิด การหาค่าเฉลี่ยแบบแอมพลิจูด รวมทั้งการกำหนดจุดสิ้นสุดของการจัดกลุ่มที่ได้กล่าวถึงในหัวข้อที่แล้ว ซึ่งเป็นที่ทราบดีสำหรับผู้วิจัยในด้านของ

การทำเหมืองข้อมูลอนุกรมเวลาว่า ทั้งระยะทางยุคลิดและการเฉลี่ยแบบแอมพลิจูดนี้ไม่สามารถให้ผลลัพธ์ที่ดีสำหรับข้อมูลที่มีการบิดเบือน (Warp) ในหน่วยเวลา (ดังภาพที่ 5.1)

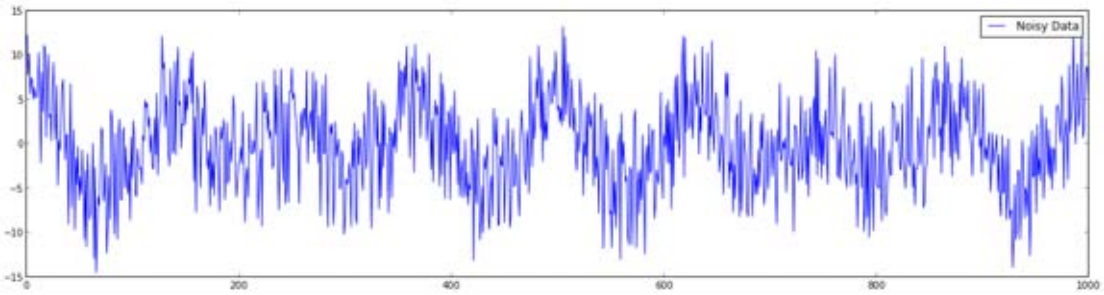


ภาพที่ 5.1 ตัวอย่างข้อมูลที่มีการบิดเบือนในหน่วยเวลา

โดยข้อมูลดังกล่าวนี้อาจมีการบิดเบือนในขนาดที่ไม่คงที่ แต่หากยิ่งบิดเบือนมากค่าความแตกต่างระหว่างระยะทางยุคลิดก็จะเพิ่มขึ้นตามไปด้วย ดังนั้นหากเปลี่ยนโครงสร้างของการจัดกลุ่มดังกล่าวโดยเปลี่ยนแปลงมาตรวัดมาใช้ไดนามิกไทม์วอร์ปปิง (Dynamic Time Warping) และเปลี่ยนแปลงการหาค่าเฉลี่ยเป็นการเฉลี่ยแบบรูปร่าง (Shape-based Averaging) ซึ่งให้ผลลัพธ์ดีกว่ากับข้อมูลประเภทดังกล่าว ก็อาจส่งผลให้ความแม่นยำของผลลัพธ์ในการจัดกลุ่มดียิ่งขึ้น แลกกับเวลาในการประมวลผลที่มากขึ้น ในที่นี้เวลาที่ใช้ในการประมวลของเฉพาะขั้นตอนการจัดกลุ่มน้อยกว่าเวลาในขั้นตอนการคัดเลือกความยาวที่เหมาะสมมาก ๆ คิดเป็น $O(mn^2)$ และ $O(M^2n^2)$ เมื่อ m คือ ความยาวของลำดับย่อยที่ยาวที่สุดในการจัดกลุ่ม M คือความยาวสุดท้ายของลำดับย่อยก่อนการหยุดประมวลผลในขั้นตอนการคัดเลือกความยาว และ n คือความยาวของข้อมูลอนุกรมเวลานำเข้า

สำหรับข้อจำกัดของการจัดกลุ่มด้วยอัลกอริทึมที่นำเสนอ เนื่องจากพื้นฐานในการเลือกค่าความยาวที่เหมาะสมในการจัดกลุ่มนั้นมาจากการคำนวณค่าการประหยัดบิตด้วยเทคนิค MDL ซึ่งขึ้นอยู่กับความคล้ายคลึงกันเริ่มต้นของลำดับย่อยภายในข้อมูลอนุกรมเวลาที่นำมาพิจารณา ดังนั้นสำหรับข้อมูลอนุกรมเวลาที่มีความแตกต่างกันระหว่างลำดับย่อยภายในอันเนื่องมาจากสัญญาณ

รบกวนหรือปัจจัยแวดล้อมอื่น ๆ (ดังภาพที่ 5.2) โดยทั่วไปแล้วอัลกอริทึมที่นำเสนอไม่สามารถทำการจัดกลุ่มให้กับข้อมูลประเภทนี้ได้ดี



ภาพที่ 5.2 ตัวอย่างข้อมูลอนุกรมเวลาที่มีสัญญาณรบกวนจำนวนมาก

แต่หากมีความต้องการที่จะทำการจัดกลุ่มข้อมูลประเภทนี้จริง ๆ ก็สามารถทำได้โดยนำข้อมูลนำเข้าไปผ่านขั้นตอนในการลดสัญญาณรบกวน (Noise Reduction) หรือ การทำให้ข้อมูลหยาบลง (Discretization) เพื่อเพิ่มความคล้ายคลึงกันระหว่างลำดับย่อยภายในให้มากขึ้นก่อนที่จะนำมาเป็นข้อมูลนำเข้าในอัลกอริทึมที่นำเสนอ

รายการอ้างอิง

[1] Keogh, E. J., Lin, J. and Truppel, W. Clustering of Time Series Subsequences is Meaningless: Implications for Previous and Future Research. In Proceedings of the IEEE International Conference on Data Mining (ICDM), pp. 115–122, 2003.

[2] Das, G., Lin, K., Mannila, H., Renganathan, G. and Smyth, P. Rule Discovery from Time Series. In Proceedings of the 3rd Knowledge Discovery and Data Mining (KDD), pp. 16–22, 1998.

[3] Ding, H., Trajcevski, G., Scheuermann, P., Wang, X. and Keogh, E. J. Querying and mining of time series data: experimental comparison of representations and distance measures. Proceedings of The Vldb Endowment 1, no. 2 (2008): 1542-1552.

[4] Nunthanid, P., Niennattrakul, V. and Ratanamahatana, C. A. Parameter-Free Motif Discovery for Time Series Data. In Proceedings of the 9th Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), pp. 1-4, 2012.

[5] Rakthanmanon, T., Keogh, E. J., Lonardi, S. and Evans, S. Time Series Epenthesis: Clustering Time Series Streams Requires Ignoring. In Proceedings of the 11th IEEE International Conference on Data Mining (ICDM), pp. 547-556, 2011.

[6] Mueen, A., Keogh, E. J., Zhu, Q., Cash, S. and Westover, M. B. Exact Discovery of Time Series Motifs. In Proceedings of the SIAM International Conference on Data Mining, pp. 473-484, 2009.

[7] Rodpongpun, S., Niennattrakul, V. and Ratanamahatana, C. A. Selective Subsequence Time Series clustering. Knowledge-Based Systems 35 (2012): 361-368.

[8] Fu, T. A review on time series data mining. Engineering Applications of Artificial Intelligence 24 (2011): 164-181.

[9] Keogh, E. J., Xi, X., Wei, L. and Ratanamahatana C. A., The UCR time series classification/clustering homepage. [Online]. 2008. Available from: www.cs.ucr.edu/~eamonn/time_series_dat/ [2013]

[10] Chen, J. R. Making Subsequence Time Series Clustering Meaningful. In Proceedings of the IEEE International Conference on Data Mining (ICDM), pp. 114-121, 2005.

[11] Wang, X., Smith, K. A., Hyndman, R. J. and Alahakoon, D. A Scalable Method for Time Series Clustering. Technical Report, Monarch University, Victoria, Australia, 2004.

[12] Dafas, P. A., and Gacez, A. S. D. Applied Temporal Rule Mining to Time Series. Technical Report, City University, London, UK, 2005.

[13] Sorrachai Yingchareonthawornchai, Proper Length Motif Discovery for Time Series Data using MDL Principle. Master's Thesis, Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, 2012.

[14] Tapp, H. S., Defernez, M. and Kemsley, E. K. FTIR Spectroscopy and Multivariate Analysis Can Distinguish the Geographic Origin of Extra Virgin Olive Oils. Journal of Agricultural and Food Chemistry 51, no. 21 (2003): 6110-6115.

[15] Rand, W. M. Objective Criteria for the Evaluation of Clustering Methods. Journal of the American Statistical Association (1971): 846-850.

[16] Niennattrakul, V., Wanichsan, D and Ratanamahatana, C. A. Accurate Subsequence Matching on Data Stream under Time Warping Distance. New Frontiers in Applied Data Mining (2010): 156-167.

ประวัติผู้เขียนวิทยานิพนธ์

นายนวิน มาดิการ เกิดเมื่อวันที่ 23 เมษายน พ.ศ. 2533 ประเทศไทย สำเร็จการศึกษาในระดับมัธยมศึกษาตอนต้นและตอนปลายจากโรงเรียนบดินทรเดชา (สิงห์ สิงหเสนี) จากนั้นเข้าศึกษาต่อในระดับอุดมศึกษาที่มหาวิทยาลัยเกษตรศาสตร์ คณะวิศวกรรมศาสตร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ ในปีการศึกษา 2551 จบการศึกษาในระดับปริญญาตรีในปีการศึกษา 2554 และเข้าศึกษาต่อทันทีในระดับปริญญาโทปีการศึกษา 2555 ที่จุฬาลงกรณ์มหาวิทยาลัย คณะวิศวกรรมศาสตร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ มีความสนใจส่วนตัวในเรื่องของการค้นพบความรู้ การทำเหมืองข้อมูล สถิติศาสตร์ และ เชาว์ปัญญาเชิงธุรกิจ