การระบุต้นตอของการคัดลอกรหัสต้นฉบับ

Original Source Code Identification

ชวลิต เสาร์แบน

มหาวิทยาลัยบูรพา

2562

การระบุต้นตอของการคัดลอกรหัสต้นฉบับ

ชวลิต เสาร์แบน

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาการสารสนเทศ
คณะวิทยาการสารสนเทศ มหาวิทยาลัยบูรพา
2562

ORIGINAL SOURCE CODE IDENTIFICATION

CHAWALIT SAOBAN

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR MASTER OF SCIENCE

IN INFORMATICS

FACULTY OF INFORMATICS

BURAPHA UNIVERSITY

2019

คณะกรรมการควบคุมวิทยานิพนธ์และคณะกรรมการสอบวิทยานิพนธ์ได้พิจารณาวิทยานิพนธ์ของ ชวลิต เสาร์แบน ฉบับนี้แล้ว เห็นสมควรรับเป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการสารสนเทศ ของมหาวิทยาลัยบูรพาได้

คณะกรรมการควบคุมวิทยานิพนธ์

..... อาจารย์ที่ปรึกษาหลัก
(ผู้ช่วยศาสตราจารย์ ดร. สุนิสา ริมเจริญ)

คณะกรรมการสอบวิทยานิพนธ์

..... ประธาน
(ผู้ช่วยศาสตราจารย์ ดร. ภารุจ รัตนวรพันธุ์)

..... กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร. ณัฐนนท์ ลีลาตระกูล)

..... กรรมการ
(ผู้ช่วยศาสตราจารย์ ดร. สุนิสา ริมเจริญ)

คณะวิทยาการสารสนเทศอนุมัติให้รับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการสารสนเทศ ของมหาวิทยาลัยบูรพา

..... คณบดีคณะวิทยาการสารสนเทศ
(ผู้ช่วยศาสตราจารย์ ดร. กฤษณะ ชินสาร)
วันที่.....เดือน.....พ.ศ.....

60910062: สาขาวิชา: วิทยาการสารสนเทศ; วท.ม. (วิทยาการสารสนเทศ)
คำสำคัญ: การบ้านวิชาการเขียนโปรแกรม, การลอกเลียนแบบ, รหัสต้นฉบับ, การจำแนกข้อมูล

ชวลิต เสาร์แบน : การระบุต้นตอของการคัดลอกรหัสต้นฉบับ. (Original Source Code Identification) คณะกรรมการควบคุมวิทยานิพนธ์: สุนิสา ริมเจริญ ปี พ.ศ. 2562.

การลอกเลียนรหัสต้นฉบับการบ้านวิชาการเขียนโปรแกรมถือเป็นปัญหาสำคัญ เนื่องด้วยวิชาการเขียนโปรแกรมถือได้ว่าเป็นพื้นฐานสำคัญของการเรียนด้านเทคโนโลยีสารสนเทศ อาจทำให้ส่งผลกระทบต่อการเรียน ตลอดจนอาชีพการงานในอนาคต ซึ่งพฤติกรรมนี้ควรปรับปรุงแก้ไขเพื่อไม่ให้เกิดปัญหาต่อสังคม ดังนั้น ถ้าผู้สอนสามารถระบุบุคคลที่เป็นต้นตอของการคัดลอกและบุคคลคัดลอกรหัสต้นฉบับได้ จะทำให้ทราบถึงกลไกของการลอกเลียนแบบและนำไปสู่แนวทางการแก้ไขปัญหาตลอดจนการบ่งชี้ถึงพฤติกรรมที่ไม่ถูกต้อง งานวิจัยฉบับนี้จึงนำเสนอวิธีการระบุต้นตอของการคัดลอกรหัสต้นฉบับ โดยนำเสนอวิธีการ 2 แนวทาง ประกอบด้วย แนวทางที่ 1 การใช้วิธีการจำแนกเพื่อระบุต้นตอของการคัดลอกรหัสต้นฉบับจำนวน 6 เทคนิคได้แก่ Decision Tree, REP Tree, Random Forest, Neural Network, K-Nearest Neighbor, และ Naïve Bayes สำหรับข้อมูลที่นำมาใช้เป็นข้อมูลที่สร้างขึ้นเองโดยใช้รูปแบบการลอกเลียนแบบในวิชาการเขียนโปรแกรม โดยผลลัพธ์ของการทดลองพบว่า เทคนิคของ Decision Tree และ REP Tree ให้ค่าความถูกต้องมากที่สุดด้วยอัตราร้อยละ 82.62 แนวทางที่ 2 การระบุต้นตอการคัดลอกรหัสต้นฉบับจากการอนุมานข้อมูลในอดีต ซึ่งได้ใช้ข้อมูลจากการบ้านวิชาการเขียนโปรแกรมที่ทราบว่าใครเป็นต้นฉบับจากการสารภาพของนิสิต จำนวน 965 โปรแกรม และนำข้อมูลเวลาของการส่ง และเกรดเฉลี่ยสะสมนำมาพิจารณาประกอบด้วย โดยผลการทดลองแบ่งออกเป็น 2 กรณี ได้แก่ 1) กรณีที่ใช้ค่าขีดแบ่งจากค่าความคล้ายคลึงกันจากโปรแกรม JPlag ในการจัดกลุ่ม ผลการระบุต้นฉบับมีความถูกต้องร้อยละ 69.39 2) กรณีที่สมมติว่าทราบกลุ่มนักเรียนที่รหัสต้นฉบับ สามารถระบุต้นฉบับได้ถูกต้องได้ถูกต้องร้อยละ 91.27 อย่างไรก็ตาม แนวทางที่ 2 ที่ผู้วิจัยได้นำเสนอกล่าวคือ การใช้ข้อมูลในอดีตมาอนุมานเพื่อระบุต้นฉบับ อาจใช้ไม่ได้ในสถานการณ์จริง เนื่องจากยากที่จะได้ข้อมูลการลอกกันมา และพฤติกรรมการลอกของนักเรียนก็อาจจะเปลี่ยนไปหลังจากสารภาพแล้ว

60910062: MAJOR: INFORMATICS; M.Sc. (INFORMATICS)

KEYWORDS: programming assignment, plagiarism, source codes, classification

CHAWALIT SAOBAN : ORIGINAL SOURCE CODE IDENTIFICATION . ADVISORY COMMITTEE: SUNISA RIMCHAROEN, 2019.

Source-code plagiarism in programming assignments is a serious issue. It can lead to bad consequences of students in personal and professional life. Students who copy someone else's source-code did not learn anything. It would be useful if teachers know who is the owner of the original copy and someone is a copier. They may employ strategies to prevent the plagiarism and give advice to students who did wrong. This thesis, therefore, proposes two methodologies to identify the original copy of the source codes among the plagiarized programs. First methodology, we use six classification techniques to classify the suspect programs i.e. Decision Tree, REP Tree, Random Forest, Neural Network, K-nearest Neighbor, and Naïve Bayes. The experimental results show that the decision tree algorithm performs best. It yields the accuracy of 82.61% in testing. Second methodology, the owner of an original source code is identified by inferring from past behavior. This methodology also incorporates submission time and GPA to decide the original one. The results are reported in two cases. Case 1, the source codes that have higher similarity score than the threshold are grouped together. We then identify the original one in each group using the proposed rule. It yields accuracy of 69.39%. Case 2, we assume that we already known the groups of suspected students. In this case, the accuracy in identifying original is 91.27%. However, the second methodology may not be convenient to adopt in real situation due to the difficulty of acquiring historical data of student confession. Students' behavior may also change after they confess.

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปด้วยดี ด้วยได้รับความกรุณาจากบุคคลหลายๆ ท่าน ซึ่งคอยให้คำแนะนำชี้แนะและสนับสนุนช่วยเหลือ ตลอดระยะเวลาของการศึกษางานวิจัย จึงขอใคร่ขอกราบขอบพระคุณบุคคลที่เกี่ยวข้องดังต่อไปนี้

ลำดับแรกขอขอบพระคุณ ผู้ช่วยศาสตราจารย์ ดร. สุนิสา ริมเจริญ อาจารย์ที่ปรึกษาหลัก ผู้คอยให้คำชี้แนะและช่วยเหลือในการศึกษาค้นคว้างานวิจัยต่างๆที่เกี่ยวข้อง ตลอดจนคอยเอาใจใส่ ให้กำลังใจ ทำให้การทำวิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้ด้วยดี

ลำดับต่อมาขอขอบพระคุณ ผู้บังคับบัญชากองบังคับอำนวยการตำรวจภูธรภาค 2 ที่คอยสนับสนุนและส่งเสริมในการศึกษาเล่าเรียน ตลอดจนให้กำลังใจในการทำวิทยานิพันธ์

ลำดับต่อมาขอขอบพระคุณ ครู อาจารย์ เพื่อน และพี่ ที่คอยให้กำลังใจและเป็นที่ปรึกษาในการทำงานวิจัยนี้

และท้ายที่สุดที่สำคัญยิ่ง ขอขอบพระคุณพ่อแม่ พี่ชาย และพี่สาว ผู้ที่คอยให้กำลังใจ คอยช่วยเหลือ สนับสนุน ตลอดสร้างแรงบันดาลใจทำให้มีความมุ่งมั่นตั้งใจในการทำวิทยานิพันธ์ให้สำเร็จลุล่วง ซึ่งทั้งหมดของการศึกษาค้นคว้าและทำงานวิจัยในครั้งนี้ ผู้วิจัยหวังเป็นอย่างยิ่งว่าความรู้และประสบการณ์ในการทำวิทยานิพนธ์ฉบับนี้จะสามารถนำไปใช้ให้เกิดประโยชน์ต่อสังคมและประเทศชาติต่อไป

ชวลิต เสาร์แบน

สารบัญ

หน้า

สารบัญตาราง

หน้า

สารบัญรูปภาพ

หน้า

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญ

ในวงการศึกษา การลอกเลียนแบบงาน เป็นปัญหาที่สำคัญอีกประการหนึ่งที่จะส่งผลกระทบต่อการพัฒนาองค์ความรู้ของผู้เรียน ทำให้ผู้ที่ลอกเลียนผลงานของคนอื่นขาดความรู้ความเข้าใจต่อชิ้นงานดังกล่าว และหากทำอย่างต่อเนื่องอาจส่งผลกระทบต่อความสำเร็จด้านการศึกษาได้ โดยเฉพาะอย่างยิ่งในระดับอุดมศึกษาซึ่งเป็นสถาบันที่ผลิตบุคลากรที่มีความรู้ความสามารถในการพัฒนาสังคมและประเทศชาติ จึงได้มีความพยายามในการค้นหาวิธีการตรวจจับการลอกเลียนแบบต่างๆ อาทิ การลอกเลียนการบ้าน วารสาร วรรณกรรม ตลอดจนการลอกเลียนผลงานวิจัย ซึ่งหนึ่งในนั้นมีงานวิจัยที่มีการค้นหาวิธีการตรวจจับการลอกเลียนการเขียนโปรแกรมหรือรหัสต้นฉบับ (Source Code) เช่น (Prechelt, Malpohl, & Philippsen, 2002) การค้นหาการลอกเลียนแบบระหว่างโปรแกรมด้วยเครื่องมือที่ชื่อว่า “JPlag” ด้วยการดึงโทเค็น (Token) ของชุดคำสั่งระหว่างสองโปรแกรมนำมาเปรียบเทียบกันโดยอาศัยอัลกอริทึม “Greedy String Tiling” แล้วนำมาคำนวณหาค่าความคล้ายคลึงกัน (Similarity) ระหว่างสองโปรแกรม (Bandara & Wijayarathna, 2011) เสนอถึงการใช้เครื่องมือเพื่อตรวจสอบการลอกเลียนแบบรหัสต้นฉบับ (Source Code) ด้วยวิธีการเรียนรู้ (Machine learning) เพื่อระบุความเป็นเจ้าของโปรแกรมหรือรหัสต้นฉบับซึ่งใช้ 3 อัลกอริทึมสำหรับการเรียนรู้ ได้แก่ Navie Bayes, ขั้นตอนวิธีการเพื่อนบ้านใกล้ที่สุด (K-Nearest Neighbor Algorithm: kNN) และขั้นตอนวิธีการเรียนรู้ด้วยตนเองของเอดาบูส (AdaBoost Meta-learning Algorithm) (Ohno & Murao, 2011) นำเสนอการตรวจสอบการลอกเลียนแบบรหัสต้นฉบับ 2 ขั้นตอน กล่าวคือ การวัดค่าความคล้ายคลึงกันโดยใช้เครื่องมือที่เรียกว่า “SIM” และ ตรวจสอบลักษะรูปแบบการเขียนโปรแกรมของแต่ละคนด้วยขั้นตอนวิธี CM (CM Algorithm) (Mann & Frew, 2006) เสนอการวิเคราะห์ความสัมพันธ์ระหว่างรหัสต้นฉบับหรือโปรแกรมที่มีการลอกเลียนแบบและโปรแกรมทั่วไปในการบ้านของผู้เรียน โดยมีปัจจัยที่เป็นตัววัดอยู่ 5 ชนิด ได้แก่ การหาค่าความคล้ายคลึงกันใช้เครื่องมือผ่านเว็บไซต์ Turnitin.com, การนับจำนวนบรรทัดของโปรแกรม (Line of code) , การหาค่าความซับซ้อนโครงสร้างโปรแกรม (Cyclometric complexity), การนับจำนวนฟังก์ชัน (Function count) และการนับจำนวนบรรทัดคำอธิบาย (Comment line) และ

(Ji, Park, Woo, & Cho, 2008) ได้นำเสนอการสร้างวิวัฒนาการลอกเลียนแบบด้วยโครงสร้างต้นไม้ ซึ่งนำเทคนิคแนวคิด Local Alignment มาช่วยในการหาค่าความแตกต่างและความคล้ายคลึงกัน (Identity and Similarity) ของโปรแกรมทั้งสอง ซึ่งจะได้ค่าความคล้ายคลึงกันของโปรแกรมทั้งสอง โดยกรณีหาค่าความคล้ายคลึงกันวัดจากโปรแกรมที่ 1 เทียบกับโปรแกรมที่ 2 ได้ค่ามา มากกว่า ค่าความคล้ายคลึงกันจากโปรแกรมที่ 2 เทียบกับโปรแกรมที่ 1 จะสรุปว่า โปรแกรมที่ 1 เป็นโปรแกรมที่ลอกเลียนแบบ ในทางตรงกันข้าม ถ้าน้อยกว่า โปรแกรมที่ 2 จะเป็นโปรแกรมที่ลอกเลียนแบบ หลังจากนั้น เมื่อได้ค่าความคล้ายคลึงกันระหว่างโปรแกรมทุกโปรแกรมทั้งหมด ก็จะนำมาสร้างเป็นกราฟแล้วจึงแปลงกราฟมาเป็นต้นไม้โดยใช้ขั้นตอนวิธี (Algorithm) ที่เรียกว่าต้นไม้แบบแผ่ที่เล็กที่สุด (Minimum Spanning Tree)

ผู้วิจัยเห็นว่า เนื่องจากการเขียนโปรแกรมเป็นความรู้พื้นฐานของการเรียนในสาขาด้านเทคโนโลยีสารสนเทศและคอมพิวเตอร์ หากขาดทักษะด้านดังกล่าวอาจทำให้ประสบปัญหาต่อการศึกษาในวิชาอื่นๆ ซึ่งโดยส่วนใหญ่จำเป็นต้องอาศัยทักษะด้านการเขียนโปรแกรม นอกจากนี้อาจส่งผลกระทบต่ออาชีพการงานในอนาคตอีกด้วย โดยในปัจจุบันอาชีพด้านเทคโนโลยีมีแนวโน้มเป็นที่ต้องการของตลาด ทั้งนี้อาจดูได้จากผลสำรวจของสมาคมวิทยาลัยแห่งชาติและแรงงาน (The National Association of Colleges and Employers: NACE) ที่ได้ทำการสำรวจและจัดอันดับความต้องการวิชาเอกหรือสาขาปริญญาตรีของนายจ้างในสหรัฐอเมริกาสูงสุด 10 อันดับ ในปี 2019 มีสาขาวิชาที่ติดอันดับในกลุ่มของเทคโนโลยี ได้แก่ คอมพิวเตอร์และวิทยาการสารสนเทศ (Computer & Information Sciences) (Koc, Kahn, Koncz, Salvadge, & Longenberger, 2018)

การหาค่าความคล้ายคลึงกันของรหัสต้นฉบับที่มีการลอกเลียนแบบ เป็นสิ่งที่ไม่ได้เป็นปัญหาของการแก้ไขปัญหาในลอกเลียนแบบรหัสต้นฉบับ ผู้วิจัยเห็นว่า การแก้ปัญหาจำเป็นต้องทราบถึงผู้ที่ลอกเลียนแบบ โดยหลักทั่วไปผู้ที่ลอกเลียนแบบจะมีจำนวนมากกว่าผู้ที่เป็นต้นตอของการคัดลอกรหัสต้นฉบับ ดังนั้น ผู้วิจัยจะศึกษาวิธีการเพื่อหาบุคคลที่เป็นเจ้าของรหัสต้นฉบับในการเขียนโปรแกรมการบ้านสำหรับวิชาการเขียนโปรแกรม ในกลุ่มผู้เริ่มเรียน กล่าวคือ นิสิตชั้นปี ๑ เนื่องจากผู้วิจัยเล็งเห็นถึงความสำคัญในขั้นพื้นฐานในการเรียน วิชาเขียนโปรแกรม หากแก้ปัญหาตั้งแต่เริ่มต้นจะช่วยทำให้ผู้เรียนสามารถเข้าใจและมีโอกาสพัฒนาทักษะความสามารถให้ก้าวหน้าพร้อมสู่การเรียนและการทำงานในอนาคตมากยิ่งขึ้น โดยผู้วิจัยได้วิเคราะห์แนวทางจากงานวิจัยที่ผ่านมา ดังนี้

1.1.1 การใช้เครื่องมือตรวจสอบการลอกเลียนรหัสต้นฉบับด้วยวิธีการเรียนรู้ (Machine learning) สำหรับงานวิจัยชิ้นนี้ ผู้วิจัยเห็นว่า การสร้างการเรียนรู้ (Learning) กับรหัสต้นฉบับของผู้เรียนแต่ละคนจำเป็นต้องอาศัยจำนวนโปรแกรมจำนวนมากและต้องใช้เวลาถึงจะได้เครื่องมือที่ทำนายผลได้อย่างแม่นยำ แต่เนื่องจากชุดข้อมูลที่นำมาใช้ในการทดลองนี้ได้รับมาจากผู้ที่เริ่มต้นการเขียนโปรแกรมทำให้อาจจะยังไม่สามารถเขียนโปรแกรมด้วยตัวเองเป็นจำนวนมากได้ ประกอบกับในการเรียนการสอนมีเวลาอย่างจำกัด อาจจะไม่เพียงพอในการนำมาใช้สำหรับการหาผู้ที่ลอกเลียนแบบรหัสต้นฉบับ นอกจากนี้ การเรียนการสอนเริ่มต้นยังไม่มีโครงสร้างคำสั่งที่มีความซับซ้อนทำให้รูปแบบอาจจะมีลักษณะคล้ายคลึงกันมาก ซึ่งมีโอกาสทำให้การทำนายผลคลาดเคลื่อนได้มาก

1.1.2 การตรวจสอบการลอกเลียนแบบรหัสต้นฉบับโดย 2 ขั้นตอน วิธีการนี้อธิบายด้วยเหตุผลตามข้อ 1.1.1 เนื่องจากต้องดูรูปแบบ (Style) การเขียนโปรแกรมของแต่ละบุคคลซึ่งอาจจะไม่สามารถสร้างเป็นโมเดลได้

1.1.3 การสร้างวิวัฒนาการลอกเลียนแบบด้วยโครงสร้างต้นไม้ สำหรับงานวิจัยชิ้นนี้ ผู้วิจัยมีข้อสังเกตุในประเด็นต่างๆ ดังนี้

1.1.3.1 ตามทฤษฎีของ Local alignment ค่าที่ได้ไม่ว่าจะนำข้อความที่ 1 เปรียบเทียบกับข้อความที่ 2 หรือ ข้อความที่ 2 เปรียบเทียบกับข้อความที่ 1 ผลลัพธ์จะได้เหมือนกัน

1.1.3.2 ข้ออ้างของงานวิจัยชิ้นนี้ยกถึงกฎของธรรมชาติหรือหลักทั่วไปว่า หากเปรียบเทียบค่าตามข้อที่ 1.1.3.1 แล้วนำมาใช้ในสูตรในการคำนวณตามหลักการความน่าจะเป็น ค่าที่ได้หากเมื่อค่าที่โปรแกรมที่ 1 เปรียบเทียบกับโปรแกรมที่ 2 มากกว่าค่าโปรแกรมที่ 2 เปรียบเทียบกับโปรแกรมที่ 1 จะสรุปว่า โปรแกรมที่ 1 เป็นโปแกรมที่ลอกเลียนแบบ ตรงข้ามกันหากน้อยกว่า โปรแกรมที่ 2 จะเป็นโปรแกรมที่ลอกเลียนแบบ ซึ่งผู้วิจัยเห็นว่าการยกข้ออ้างมาไม่ได้รับการพิสูจน์จึงยังขาดความน่าเชื่อถือ

จากเหตุผลที่ได้กล่าวมาทั้งหมดนี้ ในงานวิจัยชิ้นนี้จะนำวิธีการต่างๆ ที่ได้กล่าวมาประยุกต์ให้เหมาะสมกับข้อมูล และสภาพแวดล้อม เพื่อหาบุคคลที่เป็นเจ้าของรหัสต้นฉบับ โดยการสร้างขั้นตอนวิธีด้วยการวิเคราะห์คุณลักษณะข้อมูล (Attribute) ที่ได้จากเครื่องมือ “JPlag” ยกตัวอย่าง ค่าความคล้ายคลึงกันระหว่างโปรแกรมสองโปรแกรม จำนวนโปรแกรมที่มีค่าความคล้ายคลึงกันมาก เป็นต้น แล้วนำข้อมูลที่ได้มาจำแนกเป็นกลุ่มข้อมูล (Classification)

1.2 วัตถุประสงค์ของงานวิจัย

1.2.1 เพื่อศึกษาและสร้างขั้นตอนวิธีในการระบุต้นตอของการคัดลอกรหัสต้นฉบับ จากการลอกเลียนแบบการบ้านวิชาการเขียนโปรแกรม

1.2.2 เพื่อสามารถนำผลลัพธ์ที่ได้จากการสร้างรูปแบบการลอกเลียนแบบรหัสต้นฉบับ มาวิเคราะห์และระบุต้นตอของการลอกได้

1.3 ขอบเขตของงานวิจัย

1.3.1 ขั้นตอนวิธีการสำหรับการระบุต้นตอของการคัดลอกรหัสต้นฉบับจะเกิดขึ้นจากการนำปัจจัยและข้อมูลจากงานวิจัยที่ได้ศึกษามาประยุกต์ให้เหมาะสมกับลักษณะข้อมูล และสภาพแวดล้อม

1.3.2 การศึกษาข้อมูลสำหรับงานวิจัยนี้ได้แบ่งออกเป็น 2 ส่วน ได้แก่ (1) ข้อมูลที่ผู้วิจัยได้สร้างขึ้นเอง และ (2) ชุดข้อมูลจากการบ้านในวิชาการเขียนโปรแกรมของนิสิตชั้นปีที่ 1 ของมหาวิทยาลัยแห่งหนึ่ง

1.4 ขั้นตอนและวิธีการดำเนินการวิจัย

1.4.1 ศึกษาข้อมูลงานวิจัยที่มีความเกี่ยวข้องกับการลอกลอกเลียนแบบการเขียนโปรแกรมหรือรหัสต้นฉบับ

1.4.2 ศึกษาและลองใช้งานเครื่องมือที่สามารถวิเคราะห์ความเหมือนกันของรหัสต้นฉบับกับข้อมูลที่จำลองขึ้น

1.4.3 สร้างข้อมูลการลอกเลียนแบบรหัสต้นฉบับ โดยผู้วิจัยได้ดำเนินการสร้างขึ้นเอง และรวบรวมชุดข้อมูลรหัสต้นฉบับจากวิชาการเขียนโปรแกรมต้นฉบับ เพื่อมาวิเคราะห์รูปแบบการลอกเลียนแบบ

1.4.4 วิเคราะห์ปัจจัยที่ได้ศึกษานำมาทดสอบกับข้อมูลทั้ง 2 ส่วนตามข้อ 1.4.3

1.4.5 สร้างขั้นตอนวิธีเพื่อระบุต้นตอของการคัดลอกรหัสต้นฉบับ หรือโปรแกรม และสร้างกราฟการลอกเลียนแบบเพื่อใช้ตรวจสอบว่า แต่ละรหัสต้นฉบับ มีการลอกเลียนแบบมาจากรหัสต้นฉบับใดบ้าง ซึ่งจะนำมาแปลงรูปแบบกราฟมาในลักษณะต้นไม้วิวัฒนาการลอกเลียน

1.4.6 ทดสอบและตรวจสอบความถูกต้องของขั้นตอนวิธีการระบุต้นตอของการคัดลอกรหัสต้นฉบับ

1.4.7 ปรับปรุงขั้นตอนวิธีการระบุต้นตอของการคัดลอกรหัสต้นฉบับ

1.4.8 สรุปผลการวิจัยและตีพิมพ์ผลการทำวิจัย

1.4.9 เรียบเรียงและจัดทำเล่มวิทยานิพนธ์

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1.5.1 ได้ขั้นตอนวิธีการระบุต้นตอของการคัดลอกรหัสต้นฉบับ จากการลอกเลียนแบบการบ้านวิชาการเขียนโปรแกรม

1.5.2 สามารถสร้างรูปแบบวิวัฒนาการของโครงสร้างต้นไม้ในการลอกเลียนแบบของกลุ่มผู้เรียนในวิชาการเขียนโปรแกรม

1.5.3 สามารถช่วยแยกกลุ่มผู้เรียนที่ไม่ได้เขียนโปรแกรมด้วยตนเอง เพื่อเป็นแนวทางแก่ผู้สอนในการนำไปใช้ในการแก้ปัญหาของการลอกเลียนการบ้านวิชาการเขียนโปรแกรม

บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

ในงานวิจัยนี้มีจุดมุ่งหมายเพื่อต้องการศึกษาและหาวิธีการแก้ไขปัญหาของการที่มีผู้เรียนวิชาการเขียนโปรแกรมมีการนำรหัสต้นฉบับมาลอกเลียนแบบซึ่งได้กล่าวไปในบทที่ 1 แล้วนั้น ในบทที่ 2 นี้จะนำเสนอทฤษฎีและงานวิจัยที่มีความเกี่ยวข้อง ดังนี้

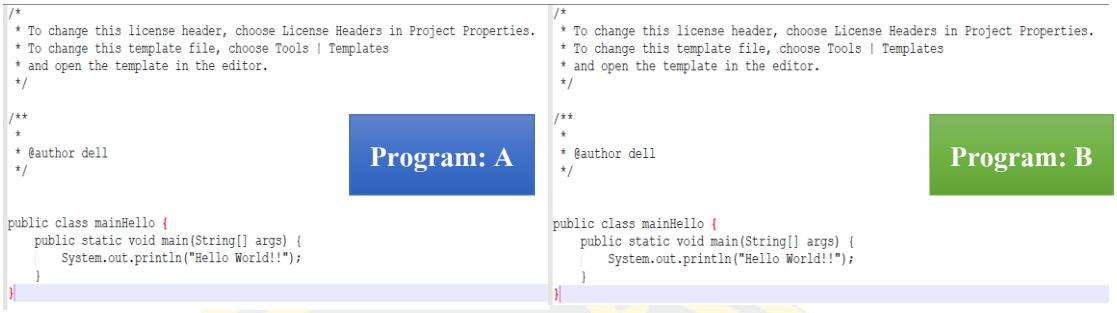
2.1 ทฤษฎีที่เกี่ยวข้องกับงานวิจัย

2.1.1 นิยามและประเภทของการลอกเลียน (Definition and Type of Plagiarism)

ในบทที่ 1 และบทที่ 2 ช่วงต้น ได้กล่าวถึงคำว่า **“การลอกเลียนแบบ (Plagiarism)”** ซึ่งเป็นคำที่มีความเกี่ยวข้องกับหัวข้อที่ผู้วิจัยได้ศึกษาและจะนำเสนอการระบุต้นตอของการคัดลอกรหัสต้นฉบับ เพื่อให้ผู้อ่านมีความเข้าใจตรงกันกับผู้วิจัย จึงขอยกความหมายคำดังกล่าวจากหนังสือเล่มเล็ก (Booklet) ฉบับออนไลน์ของมหาวิทยาลัยเคอร์ติน (Curtin University) (Price & Yorke, 2015) กล่าวคือ “**การนำเสนอผลงานหรือทรัพย์สินของผู้อื่นมาเป็นของตนเองโดยปราศจากการรับรองจากเจ้าของผลงานที่แท้จริง**” ทั้งนี้ งานวิจัยนี้จะให้คำนิยามของคำว่า **ผลงานหรือทรัพย์สิน** คือ รหัสต้นฉบับ

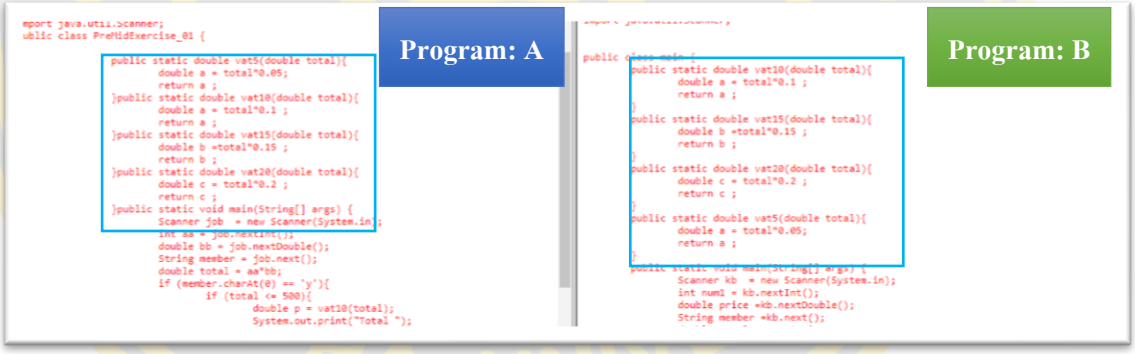
เพื่อให้สามารถเข้าใจลักษณะการลอกเลียนแบบการเขียนโปรแกรมได้มากขึ้น ผู้วิจัยได้ศึกษางานวิจัย (Ji, Woo, & Cho, 2007) ได้กล่าวถึงการนิยามการแบ่งระดับกระบวนการลอกเลียนแบบของ Faidhi และ Robinson ไว้ 6 ระดับ ดังนี้

ระดับ 0 ไม่มีการเปลี่ยนแปลง (Original Program) คือ การนำรหัสต้นฉบับทั้งหมดนำมาใช้โดยที่ไม่ได้เปลี่ยนแปลงสิ่งใด ดังตัวอย่างภาพที่ 1 จะเห็นได้ว่า โปรแกรม A เมื่อเทียบกับโปรแกรมโปรแกรม B ไม่มีสิ่งใดที่มีความแตกต่างกันเลย



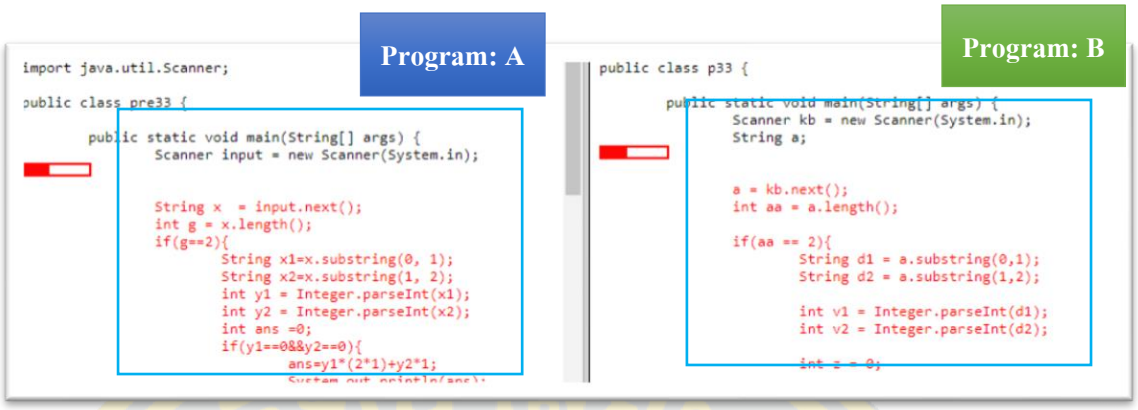
ภาพที่ 1 ตัวอย่างการลอกเลียนแบบชนิดระดับ 0: Original Program

- ระดับ 1 การเปลี่ยนแปลงคอมเมนต์และการเยื้องของบรรทัดคำสั่ง (Comment and Indentation) ดังตัวอย่างภาพที่ 2 สังเกตจากโปรแกรม A และโปรแกรม B ในกรอบสี่เหลี่ยมจะเห็นว่าในแต่ละเมธอด (Method) ของ โปรแกรม A จะอยู่หลังวงเล็บปีกกา ในขณะที่เมธอดของโปรแกรม B จะอยู่ข้างล่างอีกบรรทัด ซึ่งเมธอดทั้ง 2 โปรแกรมเหมือนกันแต่อยู่ตำแหน่งต่างกัน



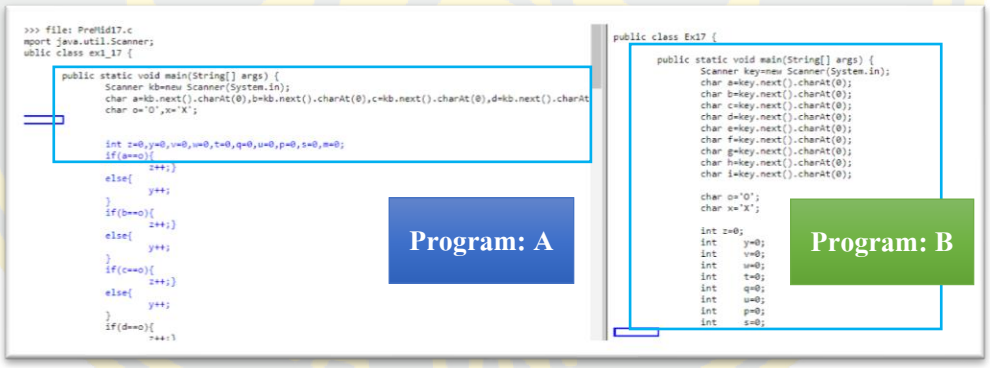
ภาพที่ 2 ตัวอย่างการลอกเลียนแบบชนิดระดับ 1: Comment and Indentation

- ระดับ 2 การเปลี่ยนแปลงชื่อตัวแปรหรือชื่อฟังก์ชัน (Identifier name) การลอกเลียนแบบในลักษณะนี้ จะมีกระบวนนำตัวแปรไปใช้หรือมีโครงสร้างฟังก์ชันที่เหมือนกัน เพียงแค่เปลี่ยนชื่อของตัวแปรหรือฟังก์ชันเท่านั้น ดังตัวอย่างภาพที่ 3 เป็นการเปลี่ยนเพียงชื่อตัวแปร แต่การใช้ตัวแปรทั้งโปรแกรม A และ โปรแกรม B เหมือนกันกล่าวคือ มีตัวแปรชนิดเดียวกัน ลักษณะการใช้งานหรือมีโครงสร้างของการทำงานเหมือนกัน



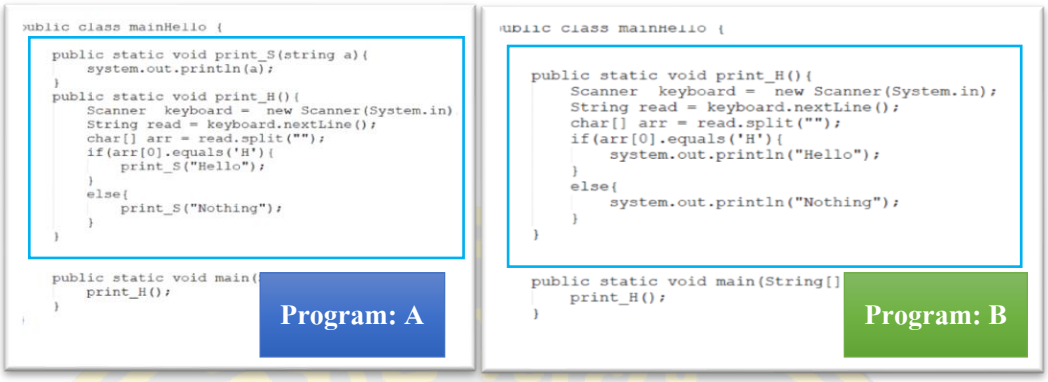
ภาพที่ 3 ตัวอย่างการลอกเลียนแบบชนิดระดับ 2: Identifier name

- ระดับ 3 การเปลี่ยนแปลงตำแหน่งตัวแปร (Variable position) ตัวอย่างภาพที่ 4 จะเห็นได้ว่าลักษณะการใช้งานตัวแปรมีจำนวนเท่ากัน เพียงแค่เปลี่ยนชื่อ และเปลี่ยนตำแหน่ง กล่าวคือ ตัวแปรโปรแกรม A จะมีประกาศเรียงต่อกันในบรรทัดเดียว แต่ขณะโปรแกรม B จะประกาศแยกทีละบรรทัด แต่มีลักษณะชนิดเดียวกัน จำนวนตัวแปรเท่ากัน



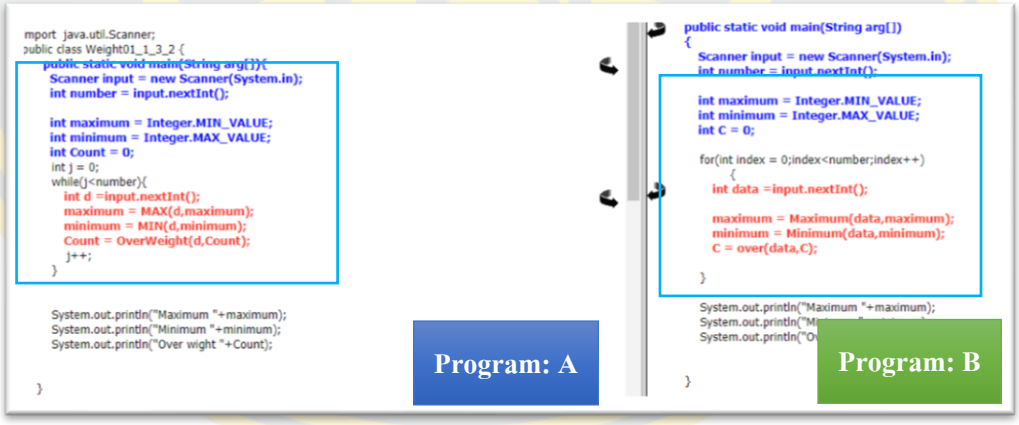
ภาพที่ 4 ตัวอย่างการลอกเลียนแบบชนิดระดับ 3: Variable position

- ระดับ 4 การรวมฟังก์ชัน (Function combination) ดังตัวอย่างภาพที่ 5 จะได้เห็นได้ว่า โปรแกรม B นำเมธอดทั้ง 2 ในโปรแกรม A มารวมไว้ในเมธอดเดียว



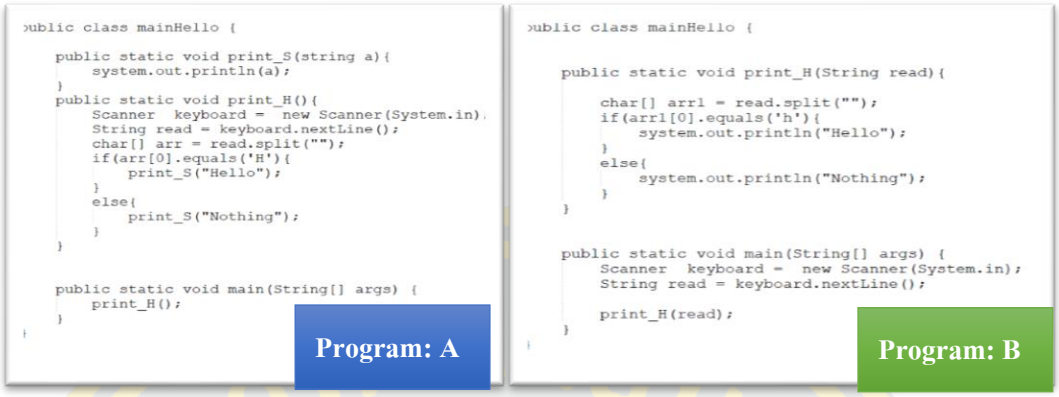
ภาพที่ 5 ตัวอย่างการลอกเลียนแบบชนิดระดับ 4: Function combination

- ระดับ 5 การเปลี่ยนแปลงโครงสร้างทางเลือก (Program statement) กล่าวคือ คำสั่งทางเลือกต่างๆ ของการเขียนโปรแกรม เช่น คำสั่ง Switch…case, if…else, for, while เป็นต้น ดังตัวอย่างภาพที่ 6 จะเห็นได้ว่า โปรแกรม A จะใช้โครงสร้างทางเลือกแบบวนซ้ำแบบ While ในขณะที่โปรแกรม B จะใช้คำสั่ง for ซึ่งในส่วนอื่นๆจะมีลักษณะคล้ายๆกัน



ภาพที่ 6 ตัวอย่างการลอกเลียนแบบชนิดระดับ 5: Program statement

- ระดับ 6 การเปลี่ยนแปลงตัวควบคุมตรรกะ (Program control logic) ในระดับนี้ถือเป็นการลอกเลียนขั้นสูงผู้ลอกเลียนต้องมีทักษะการเขียนโปรแกรม ดังนั้น ระดับนี้จึงมีความยากต่อการตรวจสอบ ดังอย่างภาพที่ 7 มีการแยกและรวมเมธอด วางรูปแบบการเขียนโปรแกรมที่ค่อนข้างต่างกัน แต่เนื่องจากตัวอย่างอาจจะยังไม่ได้ปรับส่วนประกอบอื่นมาก จึงอาจจะยังสามารถดูได้ง่ายว่าโปรแกรมมีคล้ายคลึง แต่หากมีการปรับเปลี่ยน เช่น ตัวแปร ฟังก์ชัน หรือ ชนิดตัวแปรต่างๆ และนำวิธีการในระดับก่อนหน้านี้มาใช้ซึ่งจะทำให้การตรวจสอบยากขึ้น



ภาพที่ 7 ตัวอย่างการลอกเลียนแบบชนิดระดับ 6: Program control logic

2.1.2 ขั้นตอนวิธีแบบต้นไม้ตัดสินใจ (Decision Tree)

เทคนิคการเรียนรู้ด้วยต้นไม้ตัดสินใจเป็นการทำเหมืองข้อมูล(Data mining) ด้วยการแบ่งประเภทหรือจำแนกหมวดหมู่ข้อมูลโดยการนำข้อมูลมาเรียนรู้เพื่อสร้างแบบจำลองการทำนายในรูปแบบโครงสร้างต้นไม้ โดยเลือกจากคุณลักษณะ(Attribute) ของข้อมูลมาช่วยในการตัดสินใจ

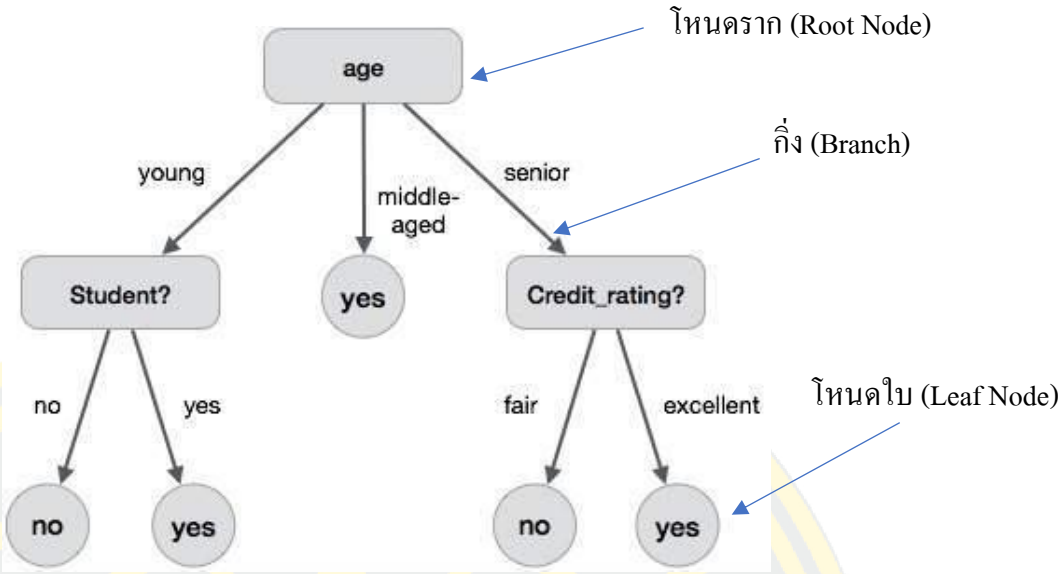
ส่วนประกอบของต้นไม้ตัดสินใจ ประกอบด้วย (ศรีเปารยะ & สินสมบูรณ์ทอง, 2560)

1) โหนด (Node) คือ คุณลักษณะข้อมูลที่เป็นตัวกำหนดว่าข้อมูลจะไปในทิศทางใด โดยโหนดที่อยู่สูงที่สุดจะถูกเรียกว่า โหนดราก (Root node)

2) กิ่ง (Branch) คือ ค่าคุณลักษณะของโหนดที่แตกกิ่งออก โดยจำนวนกิ่งจะมีจำนวนเท่ากับคุณลักษณะข้อมูลของโหนด

3) โหนดใบ (Leaf Node) คือ กลุ่มโหนดที่อยู่ล่างสุดของต้นไม้ โดยไม่มีการแตกกิ่งใดอีก

โดยสามารถยกตัวอย่างเพื่อให้เกิดความเข้าใจได้ ดังภาพที่ 8

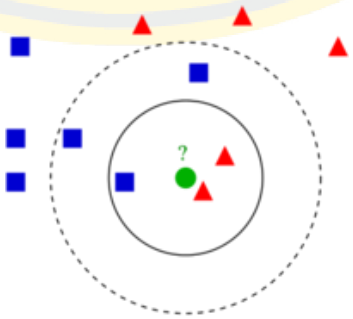


ภาพที่ 8 แสดงตัวอย่างส่วนประกอบต้นไม้ตัดสินใจ

ที่มา: https://www.tutorialspoint.com/data_mining/dm_dti.htm

2.1.3 ขั้นตอนวิธีความใกล้เคียงกันมากที่สุด (K-Nearest Neighbor)

เทคนิควิธีความใกล้เคียงกันมากที่สุด เป็นการทำเหมืองข้อมูลด้วยการจำแนกประเภทข้อมูล ในลักษณะไม่มีการสร้างโมเดลสำหรับทำนายล่วงหน้า เมื่อมีข้อมูลใหม่เข้ามาจะใช้วิธีการตรวจสอบว่าคุณลักษณะของข้อมูลมีความใกล้เคียงกับคุณลักษณะของข้อมูลเก่าใดมากที่สุด ทั้งนี้การใช้คุณลักษณะข้อมูลใดนำมาใช้ในการวัดระยะของความใกล้เคียงจะต้องมีการเรียนรู้ของข้อมูลเก่าก่อน นอกจากนี้ สิ่งสำคัญคือ จำนวนของข้อมูลที่ใกล้เคียงจะต้องมีการเปรียบเทียบจำนวนเท่าใดก็เป็นปัจจัยสำคัญในการทำให้ขั้นตอนวิธีมีประสิทธิภาพ เพื่อให้มีความเข้าใจมากยิ่งขึ้นดูตัวอย่างภาพที่ 9



ภาพที่ 9 การจัดกลุ่มข้อมูลของขั้นตอนวิธีการความใกล้เคียงมากที่สุด (K-Nearest Neighbor)

ที่มา: https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

จะเห็นได้ว่า เมื่อมีข้อมูลใหม่ (วงกลมสีเขียว) เมื่อกำหนดค่าใกล้เคียงเป็น K=3 ผลลัพธ์จะได้จำนวนสามเหลี่ยมเท่ากับ 2 และสี่เหลี่ยมเท่ากับ 1 วงกลมจึงจะถูกทำนายว่ามีคุณลักษณะเป็นสามเหลี่ยม แต่หากเลือก K=5 ผลลัพธ์จะได้จำนวนสี่เหลี่ยมเท่ากับ 3 และสามเหลี่ยมได้เท่ากับ 2 เพราะฉะนั้นวงกลมจะถูกทำนายว่ามีคุณลักษณะเป็นสี่เหลี่ยมทันที

2.1.4 ขั้นตอนวิธีแบบนาอีฟเบย์ (Naïve Bayes)

เทคนิควิธีแบบนาอีฟเบย์อาศัยหลักการความน่าจะเป็น (Probability) ตามทฤษฎีของเบย์ (Bayes Theorem) ดังสมการที่ 1 เพื่อหาสมมติฐานที่ถูกต้องที่สุด โดยใช้ความรู้ก่อนหน้า

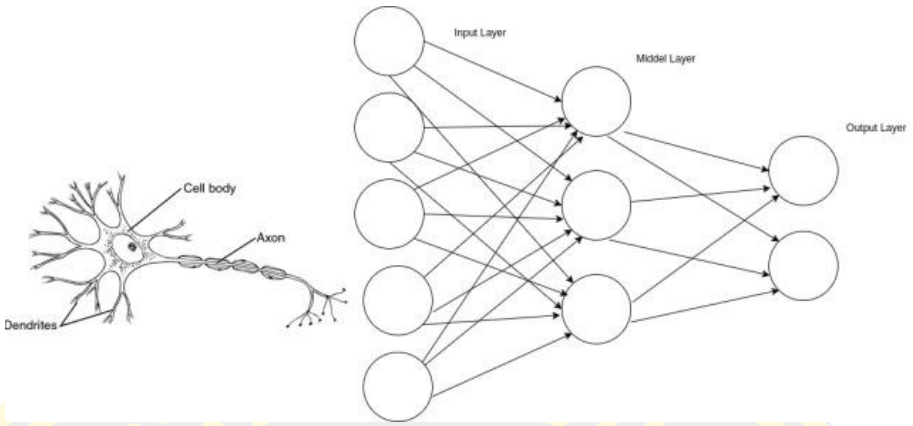
$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \quad (1)$$

- B คือ ข้อมูลที่นำมาใช้การคำนวณแจกแจงความน่าจะเป็นของสมมติฐาน A
- P(A) คือ ความน่าจะเป็นก่อนหน้าของสมมติฐาน A
- P(B) คือ ความน่าจะเป็นก่อนหน้าของชุดข้อมูลตัวอย่าง B
- P(B|A) คือ ความน่าจะเป็นของ B เมื่อรู้ A
- P(A|B) คือ ความน่าจะเป็นของ A เมื่อรู้ B

โดยนำหลักการดังกล่าวมาสร้างโมเดลสำหรับจำแนกกลุ่มของข้อมูลแต่ละคุณลักษณะของข้อมูลที่เรียนรู้ ที่ให้เหตุการณ์ที่เกิดขึ้นเป็นอิสระต่อกัน

2.1.5 ขั้นตอนวิธีโครงข่ายประสาทเทียม (Artificial Neural Network)

เทคนิควิธีโครงข่ายประสาทเทียม เป็นการพยายามสร้างการเรียนรู้ให้มีลักษณะคล้ายกับสมองมนุษย์ โดยให้เรียนรู้โดยการป้อนข้อมูลเข้ามาเพื่อสร้างแบบจำลองในการทำนายข้อมูลใหม่ที่เข้ามา เมื่อมีข้อมูลเข้ามาจะมีส่วนประมวลผลที่เป็นการจำลองเซลล์ประสาทของมนุษย์ (Neuron) ซึ่งส่วนดังกล่าวเป็นการเรียงต่อกันเป็นโครงสร้างขึ้นมา ดังตัวอย่างภาพที่ 10



ภาพที่ 10 ลักษณะรูปแบบของวิธีโครงสร้างข่ายประสาทเทียม (Artificial Neural Network)
ที่มา: https://dzone.com/articles/an-introduction-to-the-artificial-neural-network

2.2 งานวิจัยที่เกี่ยวข้อง

ในบทที่ 1 ผู้วิจัยได้อธิบายถึงข้อมูลงานวิจัยที่ได้ศึกษามารวมถึงการให้เหตุผลถึงงานวิจัยที่ใดที่มีเกี่ยวข้องกับงานของผู้วิจัย ในบทนี้จะได้นำเสนอรายละเอียดเพิ่มเติม จากที่ได้กล่าวไปแล้วว่าการวัดความเหมือนหรือความคล้ายคลึงของโปรแกรมเป็นเรื่องที่ไม่ได้ยากมากนัก เห็นได้จากที่มีงานวิจัยตลอดจนการสร้างเครื่องมือในการหาค่าความคล้ายคลึงระหว่างโปรแกรมจำนวนมาก เครื่องมือชนิดหนึ่งที่ได้รับความนิยมคือ “JPlag” (Prechelt et al., 2002) มีกระบวนการทำงานโดยเริ่มต้นจะนำโปรแกรม 2 โปรแกรมมาแปลงจากคำสั่งต่างๆ ของภาษาที่ใช้ในการเขียนโปรแกรมให้กลายเป็นโทเค็น ลักษณะดังภาพที่ 11

	Java source code	Generated tokens
1	`public class Count {`	BEGIN_CLASS
2	`  public static void main(String[] args)`	VAR_DEF, BEGIN_METHOD
3	`            throws java.io.IOException {`	
4	`    int count = 0;`	VAR_DEF, ASSIGN
5		
6	`    while (System.in.read() != -1)`	APPLY, BEGIN_WHILE
7	`      count++;`	ASSIGN, END_WHILE
8	`    System.out.println(count+" chars.");`	APPLY
9	`  }`	END_METHOD
10	`}`	END_CLASS

ภาพที่ 11 การแปลงคำสั่งในภาษาจาวา (Java) เป็นโทเค็น(Prechelt et al., 2002)

หลังจากนั้นนำโทเค็นที่ได้จากทั้งสองโปรแกรมมาเปรียบเทียบกันโดยใช้ขั้นตอนวิธี Greedy String Tiling โดยขั้นตอนวิธีดังกล่าวมีกระบวนการตามภาพที่ 12

```
0   Greedy-String-Tiling(String A, String B) {
1       tiles = {};
2       do {
3           maxmatch = M;
4           matches = {};
5           Forall unmarked tokens A_a in A {
6               Forall unmarked tokens B_b in B {
7                   j = 0;
8                   while (A_{a+j} == B_{b+j} &&
9                           unmarked(A_{a+j}) && unmarked(B_{b+j}))
10                      j++;
11                  if (j == maxmatch)
12                      matches = matches ⊕ match(a, b, j);
13                  else if (j > maxmatch) {
14                      matches = {match(a, b, j)};
15                      maxmatch = j;
16                  }
17              }
18          }
19          Forall match(a, b, maxmatch) ∈ matches {
20              For j = 0 ... (maxmatch − 1) {
21                  mark(A_{a+j});
22                  mark(B_{b+j});
23              }
24              tiles = tiles ∪ match(a, b, maxmatch);
25          }
26      } while (maxmatch > M);
27      return tiles;
28  }
```

ภาพที่ 12 ขั้นตอนวิธี Greedy String Tilling (Prechelt et al., 2002)

เมื่อดำเนินการเปรียบเทียบเสร็จแล้ว นำมาใช้ในสูตรตามสมการที่ 2 เพื่อหาค่าความคล้ายคลึงกันของโปรแกรม

$$sim(A,B) = \frac{2 \cdot coverage(tiles)}{(|A|+|B|)} \quad (2)$$

เมื่อ $coverage(tiles) = \sum_{match(a,b,length)} length$

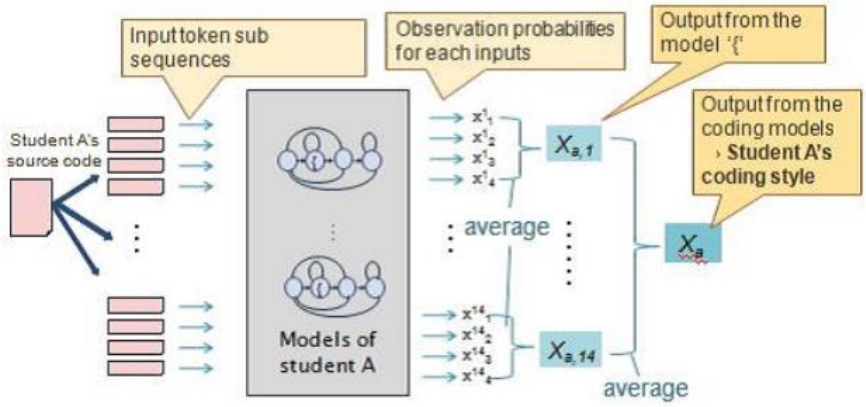
นอกจากนี้ยังมีเครื่องมืออื่นที่มีการค้นหาความคล้ายคลึงกัน เช่น MOSS SIM YAP3 Plague COPS เป็นต้น แต่ทั้งนี้ ผู้วิจัยได้เลือกใช้ JPag เป็นเครื่องมือสำหรับงานวิจัยนี้ เนื่องจากเห็นว่า เป็นเครื่องมือที่ใช้งานง่าย สะดวกต่อการดึงข้อมูลคุณลักษณะต่างๆ ของโปรแกรมที่นำมาเปรียบเทียบ ตลอดจนสามารถใช้ได้ทั้งบนเครื่อง (Stand Alone) และเว็บไซต์(Web Site) เนื่องจากระบบเป็นลักษณะเว็บเซอร์วิส (Web Service) ดังนั้น หากนำไปต่อยอดจะมีความสะดวกและยืดหยุ่นในการใช้งาน

แต่สำหรับประเด็นที่ผู้วิจัยต้องการศึกษาเพื่อการแก้ไขปัญหาการลอกเลียนแบบรหัสต้นฉบับ คือ การค้นหาเพื่อระบุต้นตอของผู้เป็นเจ้าของรหัสต้นฉบับ ในการศึกษาค้นคว้าข้อมูลงานวิจัย (Bandara & Wijayarathna, 2011) ได้นำการเรียนรู้ด้วยวิธีด้วยการสร้างรูปแบบการเขียนโปรแกรมของบุคคล ซึ่งได้ใช้เทคนิคเรียนรู้แบบนาอีฟเบย์ และเทคนิคการหาค่าใกล้เคียงที่สุด ซึ่งได้กำหนดโทเค็นหรือลักษณะการใช้รูปแบบคำสั่งไว้ตามตารางที่ 2 โดยนำคุณลักษณะดังกล่าวมาเรียนรู้ตามเทคนิคทั้งสองแบบจะได้รูปแบบของการเขียนโปรแกรมของแต่ละบุคคล

สำหรับงานวิจัยต่อไปเป็น (Ohno & Murao, 2011) การสร้างโมเดลโดยการเรียนรู้ด้วยวิธีขั้นตอน CM เพื่อสร้างรูปแบบของการเขียนโปรแกรมแต่ละบุคคล โดยมีการวัดประสิทธิภาพของโมเดลด้วย SIM แสดงได้ดังภาพที่ 13

ตารางที่ 1 คุณลักษณะสำหรับการเรียนรู้ (Bandara & Wijayarathna, 2011)

Metric Name	Description
LineLengthCalculator (LLC)	This metric measures the number of characters in one source code line.
LineWordsCalculator (LWC)	This metric measure the number of words in one source code line.
AccessCalculator (ACL)	Java uses the four level access controls: public, protected, default and private. This metrics calculates the relative frequency of these access levels used by programmers.
CommentsFrequencyCalculator (CFC)	Java uses three types of comments. This metrics calculate the relative frequency of those comment types used by the programmers.
IndentifiersLengthCalculator (ILC)	This metric calculates the length of each identifier of Java programs.
InLineSpaceInlineTabCalculator (INT)	This metric calculates the whitespaces that occurs on the interior areas of non-whitespace lines.
TrailTabSpaceCalculator (TTS)	This metric measures the whitespaces and tabs occurring at the end of each non-whitespace line.
UnderscoresCalculator (USC)	This metric measures the number of underscore characters used in identifiers.
IndentSpaceTabCalculator (IST)	This metric calculates the indentation of whitespaces used at the beginning of each non-whitespace lines



ภาพที่ 13 รูปแบบการเรียนรู้ด้วยขั้นวิธีแบบ CM (CM Algorithm) (Ohno & Murao, 2011)

งานวิจัยสุดท้าย เป็นงานวิจัยที่สร้างรูปแบบการลอกเลียนแบบรหัสต้นฉบับด้วยวิธีการสร้างวิวัฒนาการแบบต้นไม้ โดยอาศัยขั้นตอนวิธี Local Alignment ที่มีการปรับค่าวัดความคล้ายคลึงกัน ดังภาพที่ 14 ซึ่งผู้วิจัยอธิบายว่า ผลลัพธ์จากการหาค่าจากมาตราวัดที่ปรับดังกล่าว ค่าความคล้ายคลึงกันของโปรแกรม A(P_a) และโปรแกรม B(P_b) หาก P_a เทียบ P_b ไม่เท่ากับ P_b เทียบ P_a

$$W_D[x,y] = \begin{cases} -\alpha \cdot \log_2(f_x \cdot f_y) & \text{if } x = y \\ \beta \cdot \log_2(f_x \cdot f_y) & \text{if } x \neq y \\ \gamma \cdot \log_2 f_x & \text{if } y \text{ is a gap symbol} \\ \delta \cdot \log_2 f_y & \text{if } x \text{ is a gap symbol.} \end{cases}$$

ภาพที่ 14 มาตราวัดความคล้ายคลึงกันของโปรแกรมด้วย Local Alignment (Ji et al., 2008)

เมื่อได้ค่าความคล้ายคลึงกันด้วยค่ามากที่สุดของ W_D ในขั้นตอนดังกล่าวแล้ว ซึ่งแทนด้วย AsymScore(P_a, P_b) นำผลลัพธ์ที่ได้มาคำนวณอยู่ในรูปแบบมาตรฐาน (Normalization) ด้วยสมการที่ 3 แล้วจึงนำค่าที่อยู่ในรูปแบบมาตรฐานมาหาเส้นทางด้วยสมการที่ 4 ซึ่งเป็นค่าที่ใช้วัดว่ามีการลอกเลียนแบบหรือไม่ โดยกำหนดให้ ถ้า EvolDist(P_a, P_b) < EvolDist (P_b, P_a) จะสรุปได้ว่า P_b ลอกเลียนแบบ P_a ตรงข้ามถ้าไม่ใช่สรุปได้ว่า P_a ลอกเลียนแบบ P_b ดังนั้น เมื่อนำโปรแกรมทุกโปรแกรมของชุดข้อมูลมาเปรียบเทียบกันด้วยวิธีการนี้จะได้การระบุเส้นของการลอกเลียนแบบใน

ทุกคู่ทั้งหมดเป็นลักษณะกราฟ แล้วจึงแปลงกราฟให้กลายเป็นต้นไม้ ด้วยวิธีการต้นไม้แบบแผ่ที่เล็กที่สุด (Minimum Spanning Tree)

$$Asym(P_a, P_b) = \frac{2 \cdot AsymScore(P_a,P_b)}{AsymScore(P_a,P_a)+AsymScore(P_b,P_b)} \quad (3)$$

$$EvolDist(P_a, P_b) = 1 - Asym(P_a, P_b) \quad (4)$$

บทที่ 3
การระบุต้นตอการคัดลอกรหัสต้นฉบับด้วยวิธีการจำแนกข้อมูล

ในบทนี้จะนำเสนอขั้นตอนการทดลอง และผลการทดลอง เพื่อระบุต้นตอของการคัดลอกรหัสต้นฉบับด้วยวิธีการจำแนกข้อมูล ซึ่งเป็นวิธีการแรกที่จะนำเสนอ สำหรับในวิธีการที่ 2 เป็นวิธีการการระบุต้นตอการคัดลอกรหัสต้นฉบับจากการอนุมานด้วยข้อมูลในอดีต จะได้กล่าวต่อไปในบทที่ 4

3.1 ขั้นตอนการทดลอง

3.1.1 การศึกษาเบื้องต้น

จากการศึกษานิยามการลอกเลียนแบบโปรแกรมของ Faidhi และ Robinson ได้กำหนดลักษณะดังกล่าวไว้ 6 ระดับ ตามตารางที่ 2 ผู้วิจัยจึงได้นำโปรแกรมจากฐานข้อมูลเก่าของผู้เรียนปริญญาตรีชั้นปีที่ 1 มาศึกษาเพื่อดูรูปแบบและลักษณะของโปรแกรมว่ามีรูปแบบการลอกเลียนในระดับใด ตลอดจนศึกษาเครื่องมือสำหรับการสร้างคุณลักษณะของข้อมูลเพื่อนำมาใช้ในการวิจัย ซึ่งได้เลือกเครื่องมือ JPlag โดยการศึกษาข้อมูลในเบื้องต้นมีการดำเนินการ ดังนี้

3.1.1.1 ใช้ชุดข้อมูลตัวอย่างจากโปรแกรมของนิสิตที่เรียนอยู่ในระดับปริญญาตรีปีที่ 1 ในปีการศึกษา 2559 จำนวน 140 โปรแกรม

3.1.1.2 นำข้อมูลดังกล่าวใส่ในโปรแกรม JPlag เพื่อตรวจสอบค่าความคล้ายคลึงกันตั้งแต่ระดับ 60% ขึ้นไป

3.1.1.3 หลังจากนั้นผู้วิจัยได้ดูลักษณะข้อมูลที่ได้จากโปรแกรม JPlag

3.1.1.4 ศึกษาข้อมูลกลุ่มโปรแกรมที่มีค่าความคล้ายคลึงกัน เพื่อตรวจสอบดูว่ารูปแบบของโปรแกรมที่มีค่าความคล้ายคลึงกันนั้น มีลักษณะใดตามคำนิยามของ Faidhi และ Robinson ซึ่งจากการดูกลุ่มข้อมูลเหล่านั้น พบว่า มีระดับการลอกเลียนแบบในระดับไม่เกิน 4

3.1.1.5 นำค่าความคล้ายคลึงกัน มาหารูปแบบในการลอกเลียนแบบ ผู้วิจัยพบว่าการใช้ข้อมูลเพียงแค่ค่าความคล้ายคลึงกัน ยังไม่สามารถระบุต้นฉบับได้

3.1.1.6 นำคุณลักษณะของจำนวนครั้งในการส่งในระบบ เวลา และเกรด เข้ามาช่วยในการตัดสินใจในการระบุเจ้ารหัสต้นฉบับ ซึ่งจากการตรวจสอบและทดลองเบื้องต้นก็มีผลลัพธ์ที่อาจจะช่วยในการตัดสินใจได้

ตารางที่ 2 ระดับการลอกเลียนแบบ

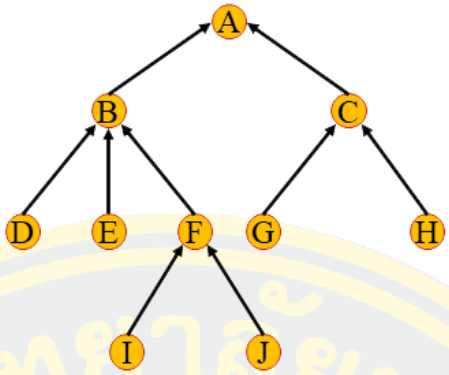
Level	**Plagiarism Method**
0	Original Program
1	Comment and Indentation
2	Identifier Name
3	Variable Position
4	Function Combination
5	Program Statements
6	Program Control Logic

3.1.2 การเตรียมข้อมูล

3.1.2.1 การสร้างข้อมูลสำหรับการวิจัย

หลังจากได้ศึกษาจากข้อมูลจริงในเบื้องต้นพบว่า หากนำข้อมูลดังกล่าวมาใช้ในการทดลองจะไม่สามารถยืนยันขั้นตอนวิธี (Algorithm) ว่าถูกต้องหรือแม่นยำเพียงใด เพราะเนื่องจากเราทราบเพียงแค่ว่ารหัสต้นฉบับมีความคล้ายคลึงกัน แต่ไม่มีเฉลยว่าใครเป็นต้นฉบับ หรือใครเป็นคนลอก ผู้วิจัยจึงได้ดำเนินการสร้างข้อมูลใหม่ โดยกำหนดรูปแบบโครงสร้างการลอกเลียนแบบไว้สำหรับการตรวจสอบ อธิบายได้ดังนี้

ผู้วิจัยได้ดำเนินการสร้างโปรแกรมที่มีรูปแบบโครงสร้างในการลอกเลียนแบบ โดยรูปแบบการลอกเลียนต้องมีระดับไม่เกิน 4 และเพื่อให้เข้าใจรูปแบบโครงสร้างของการลอกเลียนแบบจึงแสดงตัวอย่างดังภาพที่ 15 โหนดราก คือ Node A จะเป็น ต้นตอของการคัดลอกรหัสต้นฉบับ ในการทดลองมีการสร้างต้นตอของการคัดลอกและรหัสต้นฉบับที่มีการลอกเลียนแบบจำนวนทั้งสิ้น 78 โปรแกรม โดยมีรายละเอียดดังตารางที่ 3



ภาพที่ 15 ตัวอย่างรูปแบบการลอกเลียนแบบที่สร้างโดยตนเอง

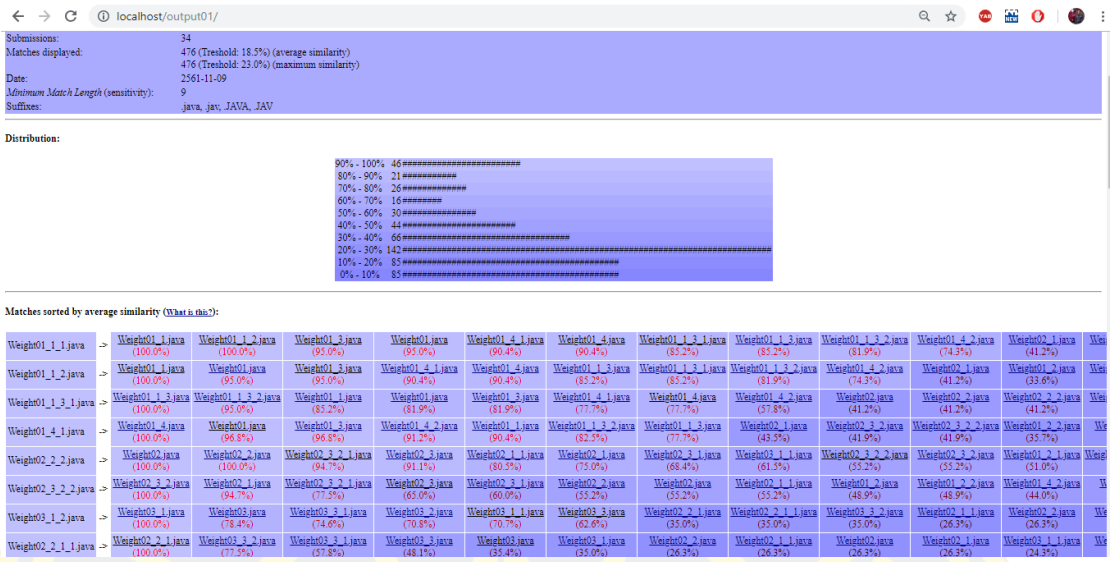
ตารางที่ 3 ชุดข้อมูลสำหรับการทดลอง

ปัญหา / จำนวน	ปัญหาที่ 1	ปัญหาที่ 2	ปัญหาที่ 3
รหัสต้นฉบับทั้งหมด	34	30	14
ต้นตอของการตัดลอกรหัสต้นฉบับ	4	4	2
ลำดับชั้นของต้นไม้	3	3	3

3.1.2.2 การสร้างคุณลักษณะของข้อมูล

เมื่อดำเนินการสร้างข้อมูลหรือโปรแกรมสำหรับการทดลองในการลอกเลียนแบบแล้ว ลำดับต่อมาคือการนำโปรแกรมดังกล่าวมาทดลองกับเครื่องมือ JPlag เพื่อนำมาสร้างคุณลักษณะให้กับข้อมูล มีกระบวนการ ดังนี้

1) สร้างโฟลเดอร์ใส่ไฟล์โปรแกรมทั้งหมดที่ได้สร้างขึ้น และดำเนินการใช้คำสั่งเพื่อหาค่าความคล้ายคลึงกันด้วยโปรแกรม JPlag ผ่านคอมมานไลน์ โปรแกรมจะดำเนินการเปรียบเทียบทุกคู่ในชุดข้อมูลของโปรแกรม โดยผลลัพธ์ที่ได้เป็นไฟล์ข้อมูลเพื่อแสดงผลบนเว็บเซอร์วิส ตัวอย่างดังภาพที่ 16



ภาพที่ 16 ตัวอย่างการแสดงผลลัพธ์จากโปรแกรม JPlag

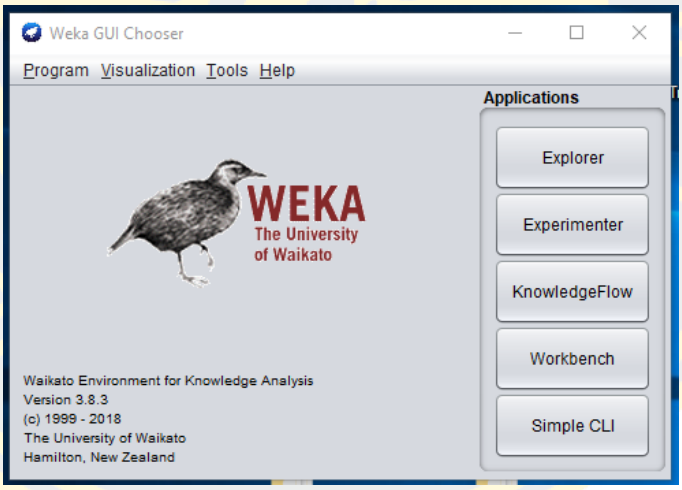
2) เขียนโปรแกรมเพื่อดึงข้อมูลจากไฟล์ที่ได้จากโปรแกรม JPlag นำมาใช้สำหรับการสร้างคุณลักษณะ โดยในแต่ละเรคคอร์ดจะจัดเก็บเป็นคุณลักษณะของแต่ละคู่โปรแกรมที่คล้ายคลึงกัน ตัวอย่างดังภาพที่ 17

3) คุณลักษณะที่นำมาใช้ในงานวิจัยนี้ โดยดึงข้อมูลจากการประมวลผลของโปรแกรม JPlag แสดงดังตารางที่ 4

4) การกำหนดค่าพารามิเตอร์ (Parameter Settings) เนื่องจากการทดลองวิธีการนี้จะนำข้อมูลที่ได้มาเรียนด้วยวิธีการจำแนก โดยใช้โปรแกรม “Weka 3.8.3” เป็นเครื่องมือสำหรับการทดลอง ตัวอย่างโปรแกรมปรากฏตามภาพที่ 18 และจากการทดลองข้อมูลในเบื้องต้นค่าพารามิเตอร์ที่มีความเหมาะสมกับเทคนิคต่างๆ แสดงดังตารางที่ 5

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	P_A	P_B	Treshold_	P_A_T	P_A_n	P_A_m	P_A_m	P_A_A	P_A_S	P_B_T	P_B_n	P_B_m	P_B_
2	Weight01_1_1.java	Weight01_1.java	100	100	11	41.2	100	85.3273	15.8107	100	11	41.2	10
3	Weight01_1_1.java	Weight01_1_2.java	100	100	11	41.2	100	85.3273	15.8107	100	11	41.2	10
4	Weight01_1_1.java	Weight01_3.java	95	95.082	11	41.2	100	85.3273	15.8107	95.082	14	43.5	10
5	Weight01_1_1.java	Weight01.java	95	95.082	11	41.2	100	85.3273	15.8107	95.082	14	43.5	10
6	Weight01_1_1.java	Weight01_4.java	90.4	93.4426	11	41.2	100	85.3273	15.8107	93.4426	12	41.9	10
7	Weight01_1_1.java	Weight01_4_1.java	90.4	93.4426	11	41.2	100	85.3273	15.8107	93.4426	12	41.9	10
8	Weight01_1_1.java	Weight01_1_3.java	85.2	85.2459	11	41.2	100	85.3273	15.8107	85.2459	13	41.2	10
9	Weight01_1_1.java	Weight01_1_3_1.java	85.2	85.2459	11	41.2	100	85.3273	15.8107	85.2459	13	41.2	10
10	Weight01_1_1.java	Weight01_1_3_2.java	81.9	81.9672	11	41.2	100	85.3273	15.8107	81.9672	15	42.3	9
11	Weight01_1_1.java	Weight01_4_2.java	74.3	73.7705	11	41.2	100	85.3273	15.8107	73.7705	12	44	91.
12	Weight01_1_1.java	Weight02_1.java	41.2	32.7869	11	41.2	100	85.3273	15.8107	32.7869	16	41.2	94.
13	Weight01_1_3_1.java	Weight01_1_3.java	100	100	13	41.2	100	73.1692	19.8582	100	13	41.2	10
14	Weight01_1_3_1.java	Weight01_1_3_2.java	95	95.082	13	41.2	100	73.1692	19.8582	95.082	15	42.3	9
15	Weight01_1_3_1.java	Weight01_1_2.java	85.2	85.2459	13	41.2	100	73.1692	19.8582	85.2459	11	41.2	10
16	Weight01_1_3_1.java	Weight01_1.java	85.2	85.2459	13	41.2	100	73.1692	19.8582	85.2459	11	41.2	10
17	Weight01_1_3_1.java	Weight01_3.java	81.9	81.9672	13	41.2	100	73.1692	19.8582	81.9672	14	43.5	10
18	Weight01_1_3_1.java	Weight01.java	81.9	81.9672	13	41.2	100	73.1692	19.8582	81.9672	14	43.5	10
19	Weight01_1_3_1.java	Weight01_4_1.java	77.7	80.3279	13	41.2	100	73.1692	19.8582	80.3279	12	41.9	10
20	Weight01_1_3_1.java	Weight01_4.java	77.7	80.3279	13	41.2	100	73.1692	19.8582	80.3279	12	41.9	10
21	Weight01_1_3_1.java	Weight01_4_2.java	57.8	57.3771	13	41.2	100	73.1692	19.8582	57.3771	12	44	91.
22	Weight01_1_3_1.java	Weight02_2_2.java	41.2	32.7869	13	41.2	100	73.1692	19.8582	32.7869	11	41.2	10
23	Weight01_1_3_1.java	Weight02.java	41.2	32.7869	13	41.2	100	73.1692	19.8582	32.7869	11	41.2	10
24	Weight01_1_3_1.java	Weight02_2.java	41.2	32.7869	13	41.2	100	73.1692	19.8582	32.7869	11	41.2	10
25	Weight01_1_2.java	Weight01_1.java	100	100	11	41.2	100	85.3273	15.8107	100	11	41.2	10
26	Weight01_1_2.java	Weight01.java	95	95.082	11	41.2	100	85.3273	15.8107	95.082	14	43.5	10
27	Weight01_1_2.java	Weight01_3.java	95	95.082	11	41.2	100	85.3273	15.8107	95.082	14	43.5	10
28	Weight01_1_2.java	Weight01_4_1.java	90.4	93.4426	11	41.2	100	85.3273	15.8107	93.4426	12	41.9	10
29	Weight01_1_2.java	Weight01_4.java	90.4	93.4426	11	41.2	100	85.3273	15.8107	93.4426	12	41.9	10
30	Weight01_1_2.java	Weight01_1_3.java	85.2	85.2459	11	41.2	100	85.3273	15.8107	85.2459	13	41.2	10
31	Weight01_1_2.java	Weight01_1_3_2.java	81.9	81.9672	11	41.2	100	85.3273	15.8107	81.9672	15	42.3	9
32	Weight01_1_2.java	Weight01_4_2.java	74.3	73.7705	11	41.2	100	85.3273	15.8107	73.7705	12	44	91.

ภาพที่ 17 ตัวอย่างการจัดเก็บคุณลักษณะของโปรแกรม



ภาพที่ 18 โปรแกรม “Weka 3.8.3”

ตารางที่ 4 กำหนดคุณลักษณะสำหรับการระบุต้นตอเจ้าของรหัสต้นฉบับ (Source Code)

คุณลักษณะ	ตัวย่อ	คำอธิบาย
file1 name	P_A	ชื่อโปรแกรมที่ 1 ที่นำมาเปรียบเทียบ (กำหนดให้ชื่อโปรแกรม A)
file2 name	P_B	ชื่อโปรแกรมที่ 2 ที่นำมาเปรียบเทียบ (กำหนดให้ชื่อโปรแกรม B)
Similarity Score	Thr_SIM	ค่าความคล้ายคลึงเฉลี่ยจากการนำโปรแกรม A ไปเปรียบเทียบโปรแกรม B และนำโปรแกรม B ไปเปรียบเทียบโปรแกรม A (ค่าเฉลี่ยของ Thr_PA กับ Thr_PB)
Similarity between A and B	Thr_PA	ค่าความคล้ายคลึงกันของโปรแกรม A เทียบกับโปรแกรม B
Similarity between B and A	Thr_PB	ค่าความคล้ายคลึงกันของโปรแกรม B เทียบกับโปรแกรม A
The Number of Programs that are similar to Program A	Num_SPA	จำนวนโปรแกรมทั้งหมดที่มีความคล้ายคลึงกับโปรแกรม A
The Number of Programs that are similar to Program B	Num_SPB	จำนวนโปรแกรมทั้งหมดที่มีความคล้ายคลึงกับโปรแกรม B
Minimum Similarity score of the programs that compare to A	Min_PA	ค่าความคล้ายคลึงกันที่น้อยสุดเมื่อนำโปรแกรมต่างๆมาเปรียบเทียบกับโปรแกรม A
Minimum Similarity score of the programs that compare to B	Min_PB	ค่าความคล้ายคลึงกันที่น้อยสุดเมื่อนำโปรแกรมต่างๆมาเปรียบเทียบกับโปรแกรม B
Maximum Similarity score of the programs that compare to A	Max_PA	ค่าความคล้ายคลึงกันที่มากสุดเมื่อนำโปรแกรมต่างๆมาเปรียบเทียบกับโปรแกรม A
Maximum Similarity score of the programs that compare to B	Max_PB	ค่าความคล้ายคลึงกันที่น้อยสุดเมื่อนำโปรแกรมต่างๆมาเปรียบเทียบกับโปรแกรม B
Average Similarity the program that compare to A	AVG_PA	ค่าเฉลี่ยความคล้ายคลึงกันเมื่อนำโปรแกรมต่างมาเปรียบเทียบกับโปรแกรม A
Average Similarity the program that compare to B	AVG_PB	ค่าเฉลี่ยความคล้ายคลึงกันเมื่อนำโปรแกรมต่างมาเปรียบเทียบกับโปรแกรม B
Standard deviation similarity scores compared to A	SD_PA	ค่าส่วนเบี่ยงเบนมาตรฐานความคล้ายคลึงกันเมื่อนำโปรแกรมต่างๆมาเปรียบเทียบกับโปรแกรม A
Standard deviation similarity scores compared to B	SD_PB	ค่าส่วนเบี่ยงเบนมาตรฐานความคล้ายคลึงกันเมื่อนำโปรแกรมต่างๆ มาเปรียบเทียบกับโปรแกรม B
First line number of program A that similar to B	NSA	ตัวเลขบรรทัดแรกที่โปรแกรม A มีค่าความคล้ายคลึงกับ B
First line number of program A that similar to A	NSB	ตัวเลขบรรทัดแรกที่โปรแกรม B มีค่าความคล้ายคลึงกับ A
The Number of Tokens	NST	จำนวนโทเค็นทั้งหมดที่มีค่าคล้ายคลึงกันในแต่ละบรรทัดของคู่โปรแกรมที่นำมาเปรียบเทียบกัน
Total number of similar lines of program A and B	DLP	ผลรวมของผลต่างทั้งหมดของจำนวนบรรทัดที่คล้ายคลึงกันระหว่างโปรแกรม A และโปรแกรม B

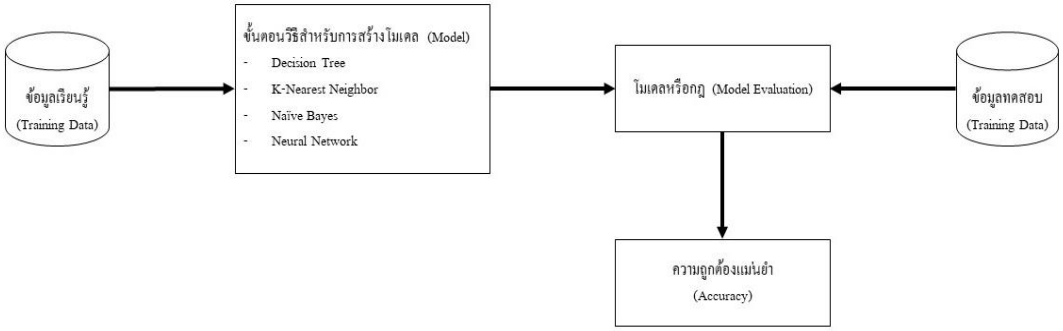
ตารางที่ 5 การกำหนดค่าสำหรับพารามิเตอร์ในเทคนิคต่างๆ

เทคนิค	พารามิเตอร์	ค่าที่เหมาะสมสำหรับการทดลอง
Decision Tree (J48)	no pruning	yes
REP Tree	no pruning	yes
Random forest	min variance prop	0.003
	numFolds	2
	seed	2
Neuron Network (MLP)	hidden layer	5
	learning rate	0.1
	momentum	0.8
	training time	2000
K-Nearest Neighbor (Ibk)	k	2
Naïve Bayes	-	-

3.1.3 วิธีการทดลอง

การทดลองสร้างโมเดลในการระบุต้นตอของการคัดลอกรหัสต้นฉบับ ลำดับแรก ผู้วิจัยได้แบ่งข้อมูลที่ได้จากข้อ 3.1.2 ออกเป็น 2 กลุ่ม ได้แก่ ข้อมูลสำหรับการเรียนรู้ (Training Data) และข้อมูลสำหรับทดสอบ (Testing Data) หลังจากนั้น นำข้อมูลส่วนที่ 1 มาสร้างโมเดลหรือกฎสำหรับการระบุต้นตอของการคัดลอกรหัสต้นฉบับ และนำส่วนที่ 2 มาทดสอบ สำหรับการสร้างโมเดลนี้ ผู้วิจัยได้ใช้วิธีการจำแนกประเภท 6 เทคนิค ประกอบด้วย (1) วิธีต้นไม้ตัดสินใจใช้ขั้นตอนวิธีชนิด J48 (2) วิธีแบบ REP Tree (3) วิธีแบบ Random forest (4) วิธีโครงข่ายประสามเทียมใช้ขั้นตอนวิธี multilayer perceptron (5) วิธีความใกล้เคียงมากที่สุดใช้ขั้นตอนวิธี IBk เนื่องจากสามารถกำหนดระยะห่างและทางเลือก เพื่อกำหนดค่า K โดยใช้ Cross-validation (6) วิธีนาอีฟเบย์เป็นวิธีที่มีประสิทธิภาพ และมีขั้นตอนวิธีไม่ซับซ้อน

หลังจากได้กฎหรือขั้นตอนวิธี สำหรับการระบุต้นตอการคัดลอกรหัสต้นฉบับ จึงนำกฎหรือขั้นตอนวิธีดังกล่าวมาทดสอบกับชุดข้อมูลสำหรับทดสอบ (Testing Data) เพื่อตรวจสอบความแม่นยำของกฎ เพื่อให้เข้าใจมากยิ่งขึ้นสามารถอธิบายได้ตามแผนภาพกระบวนสร้างกฎการระบุต้นตอการคัดลอกรหัสต้นฉบับ ดังภาพที่ 19



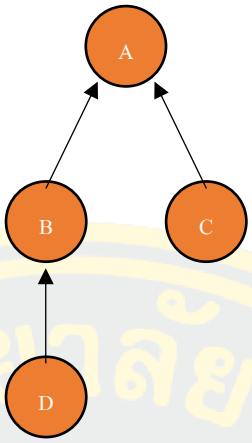
ภาพที่ 19 กระบวนการสร้างขั้นตอนวิธีเพื่อการระบุต้นตอการคัดลอกรหัสต้นฉบับ

3.1.4 การวัดประสิทธิภาพ

การทดสอบประสิทธิภาพของกฎการระบุต้นตอการคัดลอกรหัสต้นฉบับด้วยการจำแนกข้อมูลด้วยวิธีการ 6 เทคนิค เพื่อให้สามารถทราบถึงความสามารถและประสิทธิภาพในการทำนายผลของแต่ละเทคนิคในการระบุต้นตอของการคัดลอกรหัสต้นฉบับ ผู้วิจัยจึงอาศัยค่าร้อยละความถูกต้อง (Accuracy) เป็นมาตรฐานการวัดผล ตัวอย่างการวัดความถูกต้อง สมมติว่า เรามีโครงสร้างการลอกเลียนแบบดังภาพที่ 20 โดยกำหนดให้แต่ละโหนดแทนรหัสต้นฉบับของนิสิต เราจะพิจารณาความถูกต้องของกฎดังนี้

ถ้ากฎสามารถระบุเส้นเชื่อมและหัวลูกศรระหว่างโหนดได้เป็นตามลักษณะเดียวกับโครงสร้างที่กำหนดตามภาพได้ ให้ถือว่า กฎสามารถจำแนกได้อย่างถูกต้อง ซึ่งกำหนดให้การจำแนกได้อย่างถูกต้องระหว่างคู่โปรแกรมมีค่าเท่ากับ 1 เช่น ถ้า กฎสามารถจำแนกเส้นระหว่าง A และ B ได้เป็น หัวลูกศรชี้เข้ายัง A จึงมีค่านับ 1 ดังนั้น การวัดร้อยละความถูกต้องจึงสามารถอธิบายได้ดังนี้

$$\text{ร้อยละความถูกต้อง} = \frac{\text{จำนวนเส้นเชื่อมและหัวลูกศรระหว่างคู่รหัสต้นฉบับที่ระบุถูกต้อง}}{\text{จำนวนเส้นเชื่อมระหว่างโหนดทั้งหมด}} \times 100$$



ภาพที่ 20 ตัวอย่างโครงสร้างการลอกเลียนแบบ

3.2 ผลการทดลอง

ในการทดลองนี้ ผู้วิจัยใช้รหัสต้นฉบับในการเรียนรู้เพื่อสร้างกฎสำหรับระบุต้นตอการคัดลอกรหัสต้นฉบับด้วยวิธีการจำแนก 6 เทคนิค จำนวน 60 ไฟล์ และใช้โปรแกรมสำหรับการวัดประสิทธิภาพของโมเดลดังกล่าว จำนวน 17 ไฟล์ โดยผลลัพธ์ปรากฏตามตารางที่ 6

ตารางที่ 6 ผลการทดลองของการระบุต้นตอการคัดลอกด้วยวิธีการจำแนก

ขั้นตอนวิธี	ค่าความถูกต้อง	
	จากการเรียนรู้ (%)	จากการทดสอบ (%)
Decision Tree (J48)	94.74	82.61
REP Tree	93.23	82.61
Random Forest	100.00	73.91
Neural Network (MLP)	89.47	69.57
K-Nearest Neighbor (Ibk)	100.00	65.22
Navie Bayes	66.17	60.87

จากตารางที่ 6 จะเห็นได้ว่า การจำแนกด้วย 6 เทคนิค ผลลัพธ์ของการทดสอบมี 2 เทคนิคที่ให้ค่าความถูกต้องมากที่สุด ได้แก่ Decision Tree, และ REP Tree แต่อย่างไรก็ตาม REP Tree ให้ค่าความถูกต้องในชุดข้อมูลสำหรับการเรียนรู้น้อยกว่า Decision Tree เหตุผลอาจเกิดจากที่ REP Tree มาจากการตัดกิ่งของต้นไม้ ทำให้อาจมีข้อผิดพลาดการจำแนกมากกว่า Decision Tree เล็กน้อย

ผู้วิจัยจึงจะใช้กฎที่ได้รับจากเทคนิค Decision Tree มาวิเคราะห์ผลการทดลอง ซึ่งกฎที่ได้รับแสดงดังภาพที่ 21

```
if  Thr_SIM <= 97.5
|  if DLP <= 9
|  |  if Num_SPB <= 5
|  |  |  if Num_SPA <= 7
|  |  |  |  if NSB <= 27 classify as C
|  |  |  |  else classify as B
|  |  |  else
|  |  |  |  if Num_SPA <= 10 classify as A
|  |  |  |  else classify as C
|  |  else
|  |  |  if NST <= 26 classify as C
|  |  |  else
|  |  |  |  if AVG_PB <= 71.190909
|  |  |  |  |  if Max_PA <= 97.5
|  |  |  |  |  |  if NSA <= 26
|  |  |  |  |  |  |  if NST <= 27 classify as B
|  |  |  |  |  |  |  else
|  |  |  |  |  |  |  |  if Min_PA <= 44 classify as C
|  |  |  |  |  |  |  |  else classify as A
|  |  |  |  |  |  else classify as B
|  |  |  |  |  else
|  |  |  |  |  |  if SD_PA <= 16.398742 classify as C
|  |  |  |  |  |  else
|  |  |  |  |  |  |  if DLP <= 1 classify as A
|  |  |  |  |  |  |  else
|  |  |  |  |  |  |  |  if SD_PA <= 17.621168 classify as A
|  |  |  |  |  |  |  |  else classify as C
|  |  |  |  else
|  |  |  |  |  if AVG_PB <= 74.233333
|  |  |  |  |  |  if Max_PB <= 97.5 classify as C
|  |  |  |  |  |  else
|  |  |  |  |  |  |  if Num_SPB <= 12
|  |  |  |  |  |  |  |  if SD_PA <= 16.972676 classify as C
|  |  |  |  |  |  |  |  else classify as B
|  |  |  |  |  |  |  else classify as C
|  |  |  |  |  else classify as C
|  else
|  |  if Num_SPB <= 11classify as A
|  |  else
|  |  |  if Min_PB <= 42.3
|  |  |  |  if DLP <= 11classify as C
|  |  |  |  else classify as A
|  |  |  else classify as C
else classify as B
```

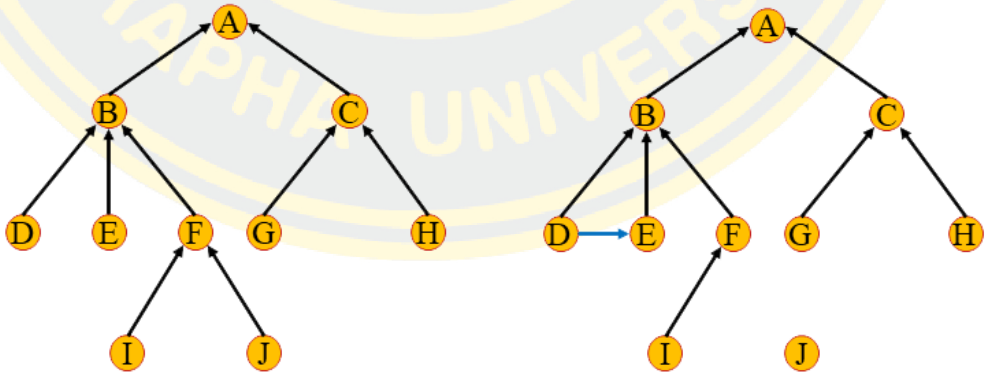
3.3

ภาพที่ 21 กฎที่ได้รับจากเทคนิค Decision Tree

สำหรับเทคนิคอื่นๆ ได้แก่ (1) เทคนิค Random Forest เป็นเทคนิคที่ใช้วิธีการสร้างต้นไม้ตัดสินใจ (Decision Tree) จำนวนหลายต้น ซึ่งแต่ละต้นจะใช้คุณสมบัติและข้อมูลที่แตกต่างกัน ทำให้ได้ผลลัพธ์ของค่าความถูกต้องในการเรียนรู้เป็นร้อยละ 100 แต่ในทางกลับกันในการทดสอบให้ผลลัพธ์เพียงร้อยละ 73.91 จึงอาจอธิบายได้ว่ากฎสำหรับการเรียนรู้เฉพาะเจาะจงกับข้อมูลสอนมากเกินไป (2) เทคนิค Neural Network ชนิด Multi-layer perceptron เป็นเทคนิคที่ใช้ขั้นตอนการส่งค่าย้อนกลับ (Backpropagation) เพื่อเรียนรู้จากค่าความผิดพลาดของผลลัพธ์เปรียบเทียบกับผลลัพธ์ที่

เป็นค่าเฉลย ซึ่งผลลัพธ์ที่ผิดพลาดจะถูกส่งกลับเพื่อไปปรับค่าน้ำหนัก (Weight) แต่เนื่องจากข้อมูลของการทดลองนี้มีค่าความใกล้เคียงกันมาก ทำให้ค่าผิดพลาดมีความใกล้เคียงกัน การปรับค่าน้ำหนักจึงไม่แม่นยำ ทำให้เกิดข้อผิดพลาดในเทคนิคนี้ (3) เทคนิค K-Nearest Neighbor ซึ่งในการทดลองใช้ค่า K=2 จากการทดลองจะเห็นได้ว่า ผลลัพธ์ที่ได้รับจากข้อมูลที่เรียนรู้ได้ค่าความถูกต้องถึงร้อยละ 100 ทำให้การจำแนกในข้อมูลสำหรับการทดสอบได้ค่าน้อย เนื่องจากโมเดลมีความเฉพาะเจาะจงกับข้อมูลในการเรียนรู้มากเกินไปเช่นเดียวกับเทคนิค Random Forest และ (4) เทคนิค Naïve Bayes ซึ่งเป็นเทคนิคที่ให้ค่าความถูกต้องน้อยที่สุดของทั้งหมด จากที่ได้กล่าวในเหตุผลของ Neural Network ว่าข้อมูลมีความใกล้เคียงกันมากทำให้การแบ่งหรือจำแนกทำได้ยาก โดยเฉพาะเทคนิคนี้ เป็นเทคนิคที่โดยปกติจะมีความเหมาะสมกับข้อมูลที่สามารถแบ่งออกเป็นกลุ่มๆ (Nominal Data) เทคนิคนี้จึงไม่เหมาะสมกับการทดลองดังกล่าวนี้ ส่งผลทำให้ได้ผลลัพธ์ร้อยละความถูกต้องน้อยที่สุด

จากที่ได้กล่าวข้างต้น ผู้วิจัยจะนำผลลัพธ์จากเทคนิค Decision Tree มาวิเคราะห์ผลการทดลอง เพื่อให้เกิดความเข้าใจในการอธิบายในกระบวนการวิเคราะห์การระบุต้นตอการคัดลอกรหัสต้นฉบับ จึงสร้างโครงสร้างต้นไม้ลอกเลียนแบบ จำนวน 10 โปรแกรม ตามภาพที่ 22 กำหนดให้แต่ละโปรแกรม แทนเป็น โหนดของต้นไม้ ให้หัวลูกศร (⟵) เป็นตัวบ่งชี้เส้นทางการลอกเลียนแบบหรือคัดลอก ยกตัวอย่าง A ⟵ B หมายถึง โปรแกรม B คัดลอกโปรแกรม A



(1) โครงสร้างการลอกเลียนที่เป็นเฉลย (2) โครงสร้างการลอกเลียนแบบที่ได้จากการทดลอง

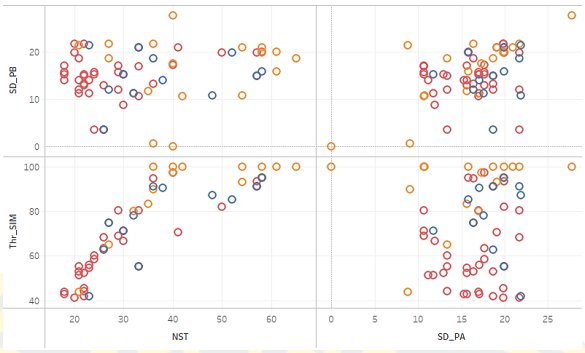
ภาพที่ 22 ตัวอย่างของรูปแบบโครงสร้างการลอกเลียน

ในภาพที่ 22 (2) เป็นการสร้างโครงสร้างการลอกเลียนรหัสต้นฉบับจากการทดสอบด้วยกฎตามภาพที่ 21 หากนำมาเทียบกับโครงสร้างการลอกเลียนที่เป็นเฉลยตามภาพที่ 22 (1) พบว่า ยังมีข้อผิดพลาดในการจำแนกเส้นทางการลอกเลียนแบบ จำนวน 2 ส่วน ได้แก่ ส่วนที่ 1 กฎไม่สามารถระบุการลอกเลียนแบบระหว่างโหนด F กับ โหนด J ได้ และ ส่วนที่ 2 กฎมีการจำแนกเส้นทางการลอกเลียนแบบผิด ในเส้นเชื่อมระหว่าง โหนด D กับ โหนด E ซึ่งเมื่อตรวจสอบจากโครงสร้างการลอกเลียนแบบที่เป็นเฉลย จะไม่ปรากฏการลอกเลียนแบบในเส้นเชื่อมดังกล่าวนี้ ซึ่งผู้วิจัยนำผลข้อผิดพลาดดังกล่าวมาวิเคราะห์ ตารางที่ 7 แสดงข้อมูลที่จะถูกนำมาพิจารณาในการจำแนกเพื่อระบุต้นตอการคัดลอกรหัสต้นฉบับ ซึ่ง 2 แถวสุดท้ายเป็นค่าของผลลัพธ์ที่ต้องการ (Actual Class) และค่าผลลัพธ์ที่ได้จากการจำแนกด้วยกฎในภาพที่ 22 (Predicted Class) ในการพิจารณาข้อผิดพลาดทั้ง 2 กรณี ได้แก่ โหนด J-F และโหนด D-E ในลำดับแรกจะวิเคราะห์ข้อผิดพลาดระหว่าง โหนด J-F ซึ่งตามผลลัพธ์ที่ต้องการเป็นคลาส B แต่กฎได้จำแนกเป็นคลาส C จากการสังเกตโดยการนำค่าคุณลักษณะโหนดของ J-B ซึ่งกฎได้ทำนายเป็นคลาส C ซึ่งได้ผลลัพธ์ที่ต้องการ มาเปรียบเทียบกับค่าคุณลักษณะของโหนด J-F ในการจำแนกของกฎที่บรรทัด 29 เป็น “if AVG_PB <= 74.233333” ซึ่งค่าคุณลักษณะ AVG_PB ของโหนด J-F คือ 73.17 และโหนด J-B คือ 85.33 จึงทำให้โหนด J-B จำแนกเป็นคลาส C ในขณะโหนด J-F ต้องถูกพิจารณาต่อในบรรทัดที่ 32 “if Num_SPB <= 12” ทำให้ J-F จำแนกเป็นคลาส C (ซึ่งผลลัพธ์ที่ต้องการเป็นคลาส B) แต่เนื่องจากค่าคุณลักษณะ Num_SPB ของโหนด J-F มีค่ามากกว่า 12 สำหรับในกรณีของโหนด D-E เกิดจากการที่กฎในบรรทัดที่ 1 “if Thr_SIM <= 97.5” ทำให้โหนด D-E จำแนกเป็นคลาส B เนื่องจากค่าของ Thr_SIM ของ D-E คือ 100 อย่างไรก็ตาม ข้อผิดพลาดดังกล่าวนี้เป็นเพียงส่วนหนึ่งซึ่งผู้วิจัยได้ยกมาวิเคราะห์ซึ่งการจำแนกให้เกิดข้อผิดพลาดลดลงอาจจำเป็นต้องเรียนรู้และตรวจสอบเพื่อเพิ่มประสิทธิภาพของโมเดลให้มากขึ้น ซึ่งจะนำประเด็นดังกล่าวนี้ไปพิจารณาเพื่อพัฒนาต่อยอดงานในอนาคตต่อไป

นอกจากนี้ ผู้วิจัยได้ตรวจสอบจากแผนภาพการกระจายของข้อมูล (Scatter plots) ของคุณลักษณะจำนวนหนึ่ง ได้แก่ Thr_SIM, NST, SD_PA, และ SD_PB พบว่า ค่าส่วนใหญ่เกาะกลุ่มไม่กระจาย บางส่วนมีการทับซ้อน ผู้วิจัยจึงเห็นว่าลักษณะของข้อมูลนี้อาจจะเป็นผลทำให้การจำแนกมีความยากต่อการทำนายผลให้ถูกต้องแม่นยำ แผนภาพแสดงการกระจายของข้อมูลแสดงได้ดังภาพที่ 23

ตารางที่ 7 ตัวอย่างค่าของคุณลักษณะจากชุดข้อมูลสำหรับทดสอบ

โหนด / คุณลักษณะ	J กับ F	J กับ B	D กับ E	D กับ B
Thr_SIM	95.00	81.90	100.00	100.00
Thr_PA	95.08	81.97	100.00	100.00
Num_SPA	15	15	11	11
Min_PA	42.30	42.30	41.20	41.20
Max_PA	95.00	95.00	100.00	100.00
AVG_PA	72.15	72.15	85.33	85.33
SD_PA	19.84	19.84	15.81	15.81
Thr_PB	95.08	81.96	100.00	100.00
Num_SPB	13	11	11	11
Min_PB	41.20	41.20	41.20	41.2
Max_PB	100.00	100.00	100.00	100.00
AVG_PB	73.17	85.33	85.33	85.33
SD_PB	19.86	15.81	15.81	15.81
NSA	48	35	65	65
NSB	48	44	58	67
NST	58	50	61	61
DLP	0	9	7	2
Actual Class	B	C	C	B
Predicted Class	C	C	B	B



ภาพที่ 23 แผนภาพแสดงการกระจายของค่าคุณลักษณะของชุดข้อมูลสำหรับการทดสอบ

บทที่ 4
การระบุต้นตอการคัดลอกรหัสต้นฉบับจากการอนุมานด้วยข้อมูลในอดีต

จากที่ได้กล่าวไว้แล้วในบทที่ 3 ว่า ผู้วิจัยจะได้นำเสนอวิธีการระบุต้นตอการคัดลอกรหัสต้นฉบับโดยการศึกษาพฤติกรรมการคัดลอกของนักเรียน แล้วนำมาอนุมานเพื่อระบุต้นฉบับ ผู้วิจัยได้รับข้อเสนอแนะจากคณะกรรมการสอบเค้าโครงงานวิจัยที่ได้ให้ข้อคิดเห็นถึงประเด็นของการระบุว่าบุคคลใดจะเป็นผู้ลอกหรือผู้ให้ลอกอาจไม่สามารถใช้เพียงการบ้านเดียวเพื่อยืนยันคำตอบได้ อาจจะต้องมีการตรวจสอบลักษณะความสัมพันธ์จากการบ้านในอดีตมาเพื่อช่วยในการตัดสินใจหรือพิจารณาการเป็นต้นฉบับได้ดียิ่งขึ้น นอกจากนี้ จากการวิเคราะห์ผลของการทดลองในวิธีการที่ 1 เนื่องจากการเรียนรู้ด้วยวิธีการจำแนก ยังมีข้อจำกัดหลายอย่าง ประการแรก เนื่องจากการเรียนรู้ด้วยข้อมูลในลักษณะใดก็จะสามารถจำแนกในข้อมูลลักษณะเดียวกับข้อมูลเรียนรู้เท่านั้น เมื่อมีข้อมูลใหม่เข้ามาโมเดลที่ถูกสร้างขึ้นจะไม่สามารถจำแนกได้อย่างถูกต้อง จึงจำเป็นต้องมีการเรียนรู้ใหม่ทุกๆครั้งหากข้อมูลมีการเปลี่ยนแปลง ประการต่อมา การจำแนกในลักษณะระบุเส้นทางของการเป็นต้นฉบับแบบลำดับชั้นหรือลำดับการสืบทอดข้อมูล จนถึงต้นตอการคัดลอกเป็นเรื่องที่ยากที่จะหาต้นฉบับของการคัดลอก ด้วยเหตุผลที่คณะกรรมการได้แนะนำและจากการนำไปศึกษาจึงได้คิดวิธีการใหม่ขึ้น ซึ่งจะกล่าวเป็น 2 ส่วน ได้แก่ ขั้นตอนการทดลอง และผลการทดลอง

4.1 ขั้นตอนการทดลอง

4.1.1 การเตรียมข้อมูล

ข้อมูลที่นำมาใช้ในวิธีการนี้เป็นไปตามข้อเสนอแนะของคณะกรรมการ กล่าวคือ เป็นข้อมูลที่เกิดจากการส่งการบ้านวิชาการเขียนโปรแกรมของนิสิตชั้นปีที่ 1 จำนวน 193 คน ซึ่งมีขั้นตอนในการเตรียมข้อมูล ดังนี้

4.1.1.1 ผู้วิจัยเลือกการบ้านที่ผู้เขียนโปรแกรมสามารถเขียนได้หลากหลายรูปแบบ ซึ่งจะช่วยให้สามารถระบุการคัดลอกได้มากขึ้น โดยการบ้านที่นำมาใช้ในการทดลองในวิธีการดังกล่าว จำนวน 4 การบ้าน ประกอบด้วย (1) การบ้านที่ 1 จำนวน 8 ข้อ (2) การบ้านที่ 2 จำนวน 7 ข้อ (3) การบ้านที่ 3 จำนวน 10 ข้อ และ (4) การบ้านที่ 4 จำนวน 22 ข้อ

4.1.1.2 นำการบ้านทั้ง 4 ชุด มาหาค่าความคล้ายคลึงกัน ด้วยโปรแกรม JPlag เพื่อเป็นการจำแนกกลุ่มโปรแกรมที่มีความคล้ายคลึงกัน

4.1.1.3 จัดเก็บรวบรวมชุดคำตอบทั้ง 4 ชุด บันทึกรูปแบบการลอกเลียนเพื่อนำไปใช้ในกฎที่สร้างขึ้น และทดสอบกฎในการหาวิธีการระบุต้นตอของการคัดลอกรหัสต้นฉบับ

4.1.1.4 นอกจากชุดข้อมูลของการลอกเลียนที่ได้อธิบายมาตอนต้น ผู้วิจัยได้นำข้อมูลที่มาช่วยสนับสนุนในการวิเคราะห์และหาวิธีการระบุต้นตอเจ้าของรหัสต้นฉบับ ได้แก่ (1) เวลาในการส่งการบ้านของนิสิตแต่ละคน (Submission time) (2) เกรดเฉลี่ยสะสม (GPA)

4.1.2 วิธีการทดลอง

4.1.2.1 เมื่อได้ชุดข้อมูลที่มีโครงสร้างของการลอกเลียนแบบ จำนวน 4 ชุด ผู้วิจัยดำเนินการแบ่งข้อมูลออกเป็น 2 ชุด ได้แก่ การสร้างชุดข้อมูลสำหรับการนำไปใช้กับกฎที่สร้างขึ้น จำนวน 3 การบ้าน และ ข้อมูลสำหรับนำไปทดสอบกฎที่นำเสนอ จำนวน 1 การบ้าน

4.1.2.2 สำหรับในขั้นตอนการสร้างข้อมูลสำหรับการนำไปใช้กับกฎที่สร้างขึ้น ผู้วิจัยนำชุดข้อมูลการลอกเลียนแบบมาบันทึกในลักษณะรูปแบบเมทริกซ์การลอกเลียนแบบของแต่ละการบ้าน โดยวิธีการสร้างเมทริกซ์การเรียนรู้ แสดงได้ดังภาพที่ 24 สามารถอธิบายได้ดังนี้

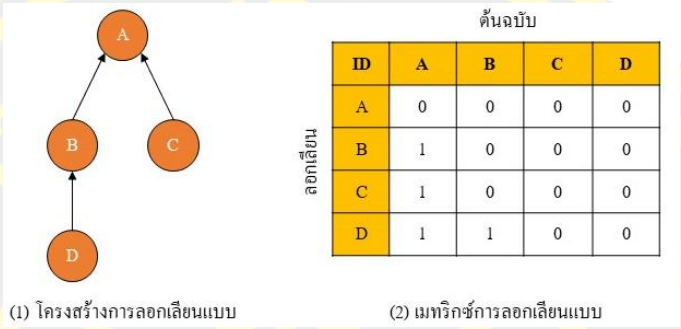
1) กำหนดให้ ภาพที่ 24 (1) เป็นโครงสร้างการลอกเลียนแบบ กำหนดให้แต่ละโหนดแทนรหัสต้นฉบับของแต่ละบุคคล และให้เส้นลูกศรเป็นตัวบ่งชี้ถึงการลอกเลียน ยกตัวอย่าง A←B หมายถึง B ลอกเลียนมาจาก A

2) กำหนดให้เมทริกซ์แนวนอนแทนรหัสต้นฉบับที่มีการลอกเลียนแบบ และเมทริกซ์แนวตั้งแทนรหัสต้นฉบับที่เป็นต้นตอของการคัดลอก ซึ่งเมื่อนำโครงสร้างลอกเลียนแบบในภาพที่ 24 (1) มาสร้างเมทริกซ์ จะได้เป็นเมทริกซ์ 4×4 ตามภาพที่ 24 (2) ซึ่งวิธีการบันทึกลงในเมทริกซ์มีวิธีการดังต่อไปนี้

- โหนด A ไม่มีเส้นลูกศรชี้ไปที่โหนด จึงมีค่าเท่ากับ 0
- โหนด B มีเส้นลูกศรไปยังโหนด A จึงบันทึกรหัสต้นฉบับที่มีการลอกเลียนเลียนแบบโหนด A เท่ากับ 1
- โหนด C มีเส้นลูกศรไปยังโหนด A จึงบันทึกรหัสต้นฉบับที่มีการลอกเลียนแบบโหนด A เท่ากับ 1 เช่นเดียวกับโหนด B

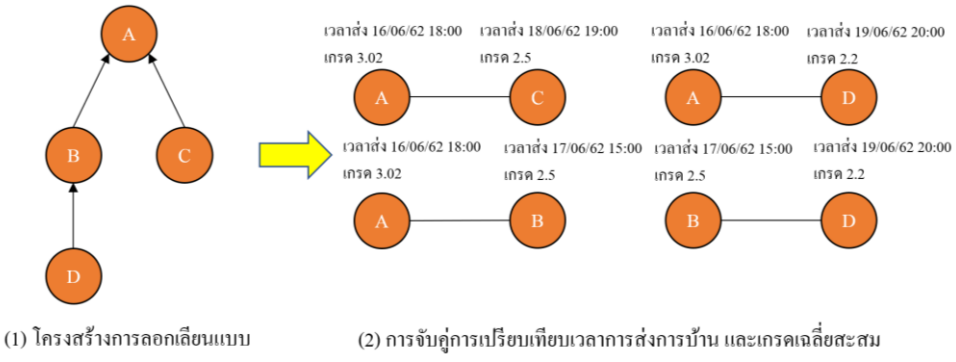
- โหนด D มีเส้นลูกศรไปยังโหนด B จึงบันทึกรหัสต้นฉบับที่มีการลอกเลียนแบบโหนด B เท่ากับ 1 แต่เนื่องจากโหนด B มีการลอกเลียนโหนด A โหนด D จึงต้องมีการลอกเลียน A เช่นเดียวกันกับโหนด B ทำให้ค่าในช่องโหนด A เท่ากับ 1

3) จากภาพที่ 24 เป็น 1 โครงสร้างใน 1 ข้อของการบ้าน หากมีการบ้านหลายข้อเมทริกซ์จะมีการอัปเดตค่าในตารางตามจำนวนที่มีการลอกเลียนแบบ



ภาพที่ 24 การสร้างเมทริกซ์การลอกเลียนแบบ

4.1.2.3 จากที่ได้อธิบายในขั้นตอนการเตรียมข้อมูลว่า วิธีการนี้จะนำข้อมูลอื่นมาสนับสนุนการวิเคราะห์และระบุต้นตอของการคัดลอกรหัสต้นฉบับ ซึ่งจะนำข้อมูลของเวลาในการส่งการบ้าน โดยมีสมมติฐานว่า รหัสต้นฉบับที่คัดลอกจะส่งช้ากว่ารหัสต้นฉบับที่เป็นต้นตอของการคัดลอก เกรดเฉลี่ยสะสมของบุคคลที่คัดลอกจะน้อยกว่าบุคคลที่เป็นต้นตอของการคัดลอก นอกจากนี้ ผู้วิจัยต้องการตรวจสอบว่า หากบุคคลที่มีการลอกเลียนแบบอยู่ต่อเนื่องในการเรียนจะมีปัญหาหรือเรียนไม่ผ่าน ด้วยตรวจสอบกับเกรดที่ได้ในวิชาการเขียนโปรแกรมอีกประการหนึ่ง โดยการจับคู่เพื่อเปรียบเทียบในแต่ละชั้นของการลอกเลียนแบบเพื่อตรวจสอบทั้งเวลาและเกรดเฉลี่ย ซึ่งแสดงตัวอย่างได้ดังภาพที่ 25



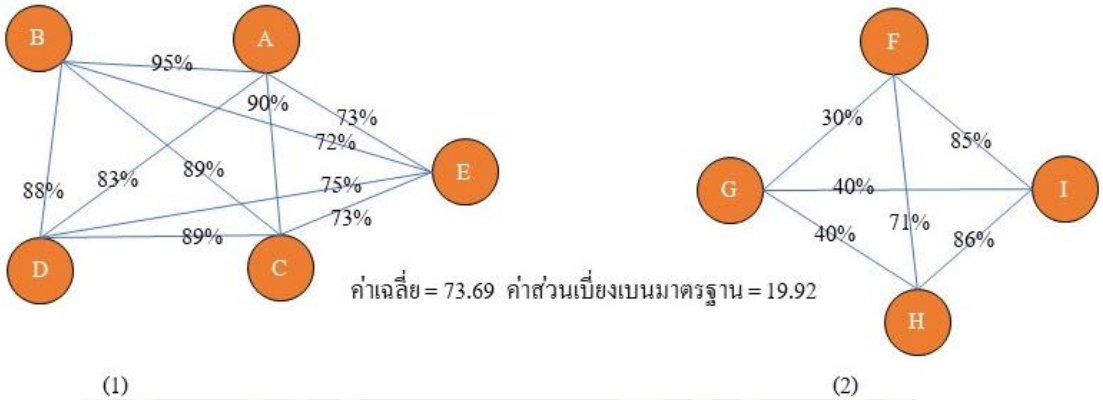
ภาพที่ 25 วิธีการเปรียบเทียบเวลาของการส่งการบ้าน และเกรดเฉลี่ยสะสม

4.1.2.4 สำหรับการทดสอบกฎที่นำเสนอ โดยใช้ชุดข้อมูลที่ได้จากการบ้าน 4 เพื่อระบุต้นฉบับ ซึ่งจะทดสอบแยกเป็น 2 การทดลอง ได้แก่

การทดลองที่ 1 ทดสอบกับกลุ่มของรหัสต้นฉบับที่ใช้วิธีการจัดกลุ่มด้วยค่าขีดแบ่งความคล้ายคลึงกัน

วิธีการทดสอบ เริ่มต้นจากการนำรหัสต้นฉบับมาหาค่าความคล้ายคลึงกันด้วยโปรแกรม JPlag หลังจากนั้น ให้คำนวณค่าเฉลี่ยและค่าส่วนเบี่ยงเบนมาตรฐานของค่าความคล้ายคลึงกันของรหัสต้นฉบับที่ได้จากการเปรียบเทียบแต่ละคู่ของรหัสต้นฉบับด้วยโปรแกรม JPlag แล้วนำค่าทั้งสองมาบวกกันเพื่อใช้เป็นค่าขีดแบ่งสำหรับการจัดกลุ่ม ขั้นตอนต่อไป เลือกกลุ่มรหัสต้นฉบับที่มีค่าความคล้ายคลึงกันมากกว่าหรือเท่ากับค่าขีดแบ่ง หลังจากนั้น จะได้กลุ่มของรหัสต้นฉบับที่มีค่าความคล้ายคลึงที่ใกล้เคียงกัน ซึ่งผู้วิจัยจะสันนิษฐานว่าเป็นกลุ่มที่มีการลอกเลียนกัน โดยแสดงตัวอย่างกรณีนี้ได้ดังภาพที่ 26 สมมติเป็นการบ้านที่นิสิตได้ทำขึ้น จำนวน 1 ข้อ และกำหนดให้แต่ละโหนดแทนเป็นรหัสต้นฉบับหนึ่งๆของนิสิตแต่ละคน หลังจากเมื่อนำโปรแกรม JPlag มาหาค่าความคล้ายคลึงกันของแต่ละคู่ของรหัสต้นฉบับดังกล่าวจะได้ดังภาพ (ในกรณีที่ไม่มีเส้นโยงหมายถึงมีค่าความคลึงกันเป็นศูนย์) และในการคำนวณหาค่าเฉลี่ยและค่าส่วนเบี่ยงเบนมาตรฐานของค่าคล้ายคลึงกันทั้งหมดจะได้ 73.69 และ 19.92 ตามลำดับ ดังนั้น ค่าความคล้ายคลึงกันของรหัสต้นฉบับแต่ละคู่ที่จะถูกเลือกให้อยู่ในกลุ่มต้องมีค่ามากกว่าหรือเท่ากับค่าขีดแบ่งซึ่งเท่ากับ 93.61 จึงเหลือเพียงคู่ของโหนด A และโหนด B ซึ่งมีค่าความคล้ายคลึงกันมากกว่าค่าขีดแบ่งดังภาพที่ 26(1) ผลลัพธ์ในการจัดกลุ่มด้วยค่าขีดแบ่งนี้จึงมีเพียงกลุ่มเดียว ซึ่งมีรหัสต้นฉบับของ A และ B ภายในกลุ่มนั้น

การทดลองที่ 2 ทดสอบกับกลุ่มของรหัสต้นฉบับที่เราทราบอยู่แล้วว่าทุกคนที่อยู่ในกลุ่มนี้มีการลอกกันมาจริงๆ (หมายเหตุ การทดลองนี้ทำขึ้นเพื่อทดสอบว่ากฎที่นำเสนอสามารถระบุต้นตอของการคัดลอกได้แม่นยำเพียงใด โดยมีสมมุติฐานว่าเราทราบกลุ่มคนที่ลอกกัน แต่ในแง่ของการนำไปใช้งานจริงไม่สามารถทำแบบนี้ได้)



ภาพที่ 26 ตัวอย่างกลุ่มของรหัสต้นฉบับที่มีค่าความคล้ายกัน

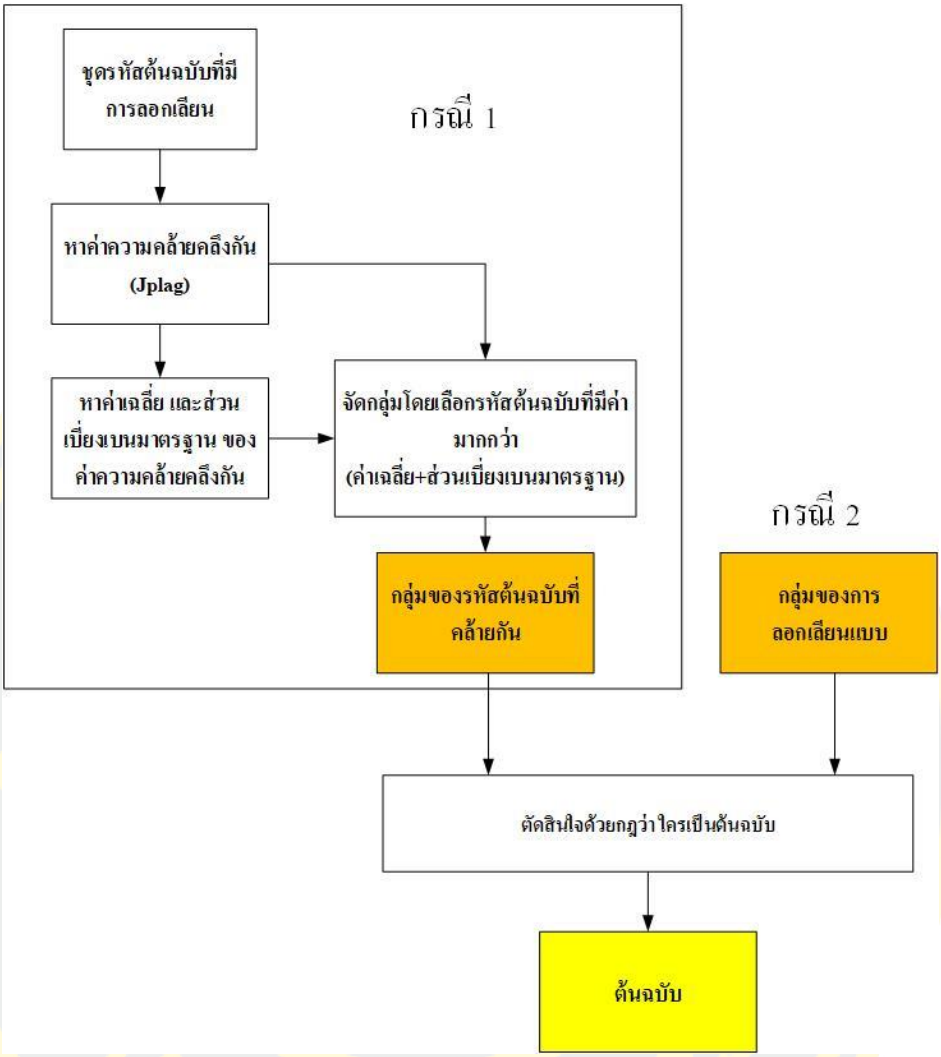
4.1.2.5 สำหรับกฎที่ผู้วิจัยได้นำเสนอ จากที่ได้กล่าวมาข้างต้นแล้วว่า เราจะอาศัยค่าเมทริกซ์การลอกเลียนแบบ (ซึ่งในงานวิจัยนี้จะมี 3 เมทริกซ์ที่เก็บข้อมูลประวัติการลอกการบ้านจาก 3 การบ้านก่อนหน้า) เวลาการส่งการบ้าน และเกรดเฉลี่ยสะสมมาใช้ในการระบุต้นตอการคัดลอกรหัสต้นฉบับ จากข้อมูลดังกล่าวผู้วิจัยจึงได้นำมาสร้างกฎ โดยให้พิจารณาดังนี้

1) ถ้าจำนวนครั้งที่นิสิตคนนั้นเคยเป็นต้นฉบับมาก่อนจากการบ้านที่ผ่านมาทั้งหมด (ซึ่งคำนวณได้จากผลรวมการเป็นต้นฉบับในเมทริกซ์การลอกเลียนแบบทั้งสามเมทริกซ์กล่าวคือ ค่าผลรวมเมทริกซ์ในแนวตั้งของรหัสนิสิตที่ต้องการแต่ละเมทริกซ์) มีค่าเท่ากับศูนย์ และเวลาของการส่งการบ้านของรหัสนิสิตใดที่มีค่าน้อยกว่า ให้สรุปว่านิสิตคนที่มีส่งการบ้านมาก่อนเป็นต้นฉบับ

2) ถ้ารหัสนิสิตคนใดส่งการบ้านก่อน และรหัสนิสิตคนที่ส่งการบ้านก่อนมีจำนวนครั้งการเป็นต้นฉบับมาก่อนจากการบ้านที่ผ่านมาทั้งหมดมากที่สุดในกลุ่มของการลอกเลียนแบบ ให้สรุปว่ารหัสนิสิตที่ส่งการบ้านก่อนและผลรวมของการเป็นต้นฉบับมากที่สุดเป็นต้นฉบับ

3) ถ้าเกรดเฉลี่ยสะสมของคนใดมากที่สุด และคนที่เกรดเฉลี่ยสะสมมากที่สุดส่งการบ้านก่อนบุคคลในกลุ่มลอกเลียนแบบ ให้สรุปว่ารหัสนิสิตที่เกรดเฉลี่ยสะสมมากที่สุดและส่งการบ้านก่อนเป็นต้นฉบับ

เพื่อให้เข้าใจในขั้นตอนการทดสอบเพื่อระบุต้นตอของการคัดลอกรหัสต้นฉบับ จึงขอแสดงกระบวนการตามภาพที่ 27



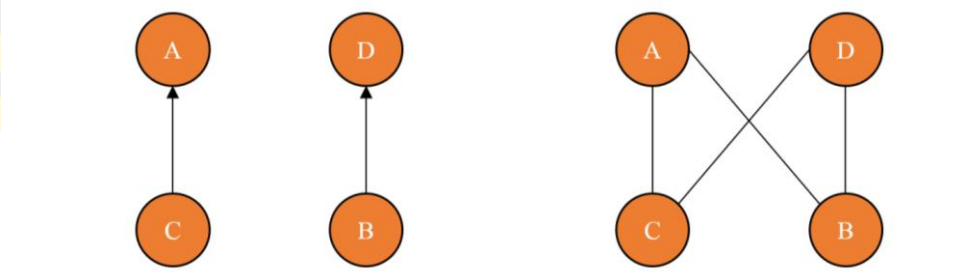
ภาพที่ 27 กระบวนทดสอบการระบุต้นตอเจ้าของรหัสต้นฉบับ

4.1.3 การวัดประสิทธิภาพ

ผู้วิจัยใช้ร้อยละความถูกต้อง ในการวัดประสิทธิภาพของกฎที่ผู้วิจัยได้นำเสนอ ซึ่งมีวิธีการวัดความถูกต้องเป็น 2 ลักษณะ ดังนี้

การทดลองที่ 1 เนื่องจากวิธีการนี้เป็นการจัดกลุ่มโดยใช้ค่าขีดแบ่งความคล้ายคลึงกันทำให้กลุ่มที่ถูกจัดขึ้นนั้น มีความแตกต่างจากโครงสร้างการลอกเลียนแบบที่ผู้วิจัยได้รับมาจากการสารภาพของนิสิต เนื่องจากชุดข้อมูลที่นำมาทดสอบมีนิสิตไม่ได้ให้คำตอบการลอกเลียนแบบ และนิสิตบางรายกรอกข้อมูลผิดพลาด เช่น ระบุรหัสนิสิตที่นำมาใช้ในการคัดลอกหรือระบุบุคคลที่เป็นต้นตอของการคัดลอกในข้อเดียวกันแต่กรอกไม่เหมือนเดิม เมื่อมีการเปรียบเทียบกับรหัสต้นฉบับ

ในคู่อื่นๆ ที่ค่าความคล้ายคลึงกัน ทำให้การนำข้อมูลมาใช้ ผู้วิจัยจึงจำเป็นต้องเลือกข้อมูลที่ระบุรหัสนิสิตที่ได้นำมาใช้ในการคัดลอกอย่างถูกต้อง ทำให้ข้อมูลที่ได้มาในแต่ละการบ้านจึงไม่สามารถระบุกลุ่มการลอกเลียนแบบได้ถูกต้องสมบูรณ์ได้ จึงทำให้การจัดกลุ่มด้วยค่าขีดแบ่งความคล้ายคลึงกันแตกต่างออกไป ตัวอย่างภาพที่ 28 (1) เป็นชุดข้อมูลได้รับมาจากคำสารภาพของนิสิต แต่เมื่อนำโปรแกรม JPlag มาหาค่าความคล้ายกันแล้ว ใช้ค่าขีดแบ่งเพื่อจัดกลุ่มดังภาพที่ 28 (2) มีความแตกต่างออกไปจากภาพที่ 28 (1) ดังนั้นในการทดลองที่ 1 นี้ ผู้วิจัยจึงกำหนดให้การวัดความถูกต้อง โดยพิจารณาว่า ถ้ากฎสามารถระบุต้นตอการคัดลอกรหัสต้นฉบับเป็นโหนดที่มีหัวลูกศรชี้เข้าหาได้ (ซึ่งตามความจริงแล้วอาจจะไม่ได้เป็นโหนดราก) ก็ให้สรุปว่า ถูกต้อง เช่น ภาพที่ 28 (2) หากกฎระบุได้ว่า A หรือ D เป็นต้นฉบับ ก็จะนับค่าความถูกต้องดังกล่าวเป็น 1 ทั้งนี้ ในการวัดร้อยละความถูกต้องจึงไม่อาจสามารถใช้จำนวนกลุ่มที่ได้รับจากคำสารภาพของนิสิตได้ เนื่องจากค่าขีดแบ่งความคล้ายคลึงกันจัดกลุ่มแตกต่างออกไป ดังนั้น ผู้วิจัยจึงวัดร้อยละความถูกต้อง ด้วยจำนวนกลุ่มที่ค่าขีดแบ่งให้มาทั้งหมด เช่น จำนวนกลุ่มทั้งหมดของทุกข้อในการบ้านซึ่งได้รับมาจากการจัดกลุ่มโดยค่าขีดแบ่งความคล้ายคลึงกัน จำนวน 70 กลุ่ม แล้ว กฎสามารถระบุได้ถูกต้อง จำนวน 56 กลุ่ม ร้อยละความถูกต้องจึงสามารถคำนวณด้วยการนำ 56 หาร 70 แล้วจึงนำมาคูณด้วย 100 มีค่าความถูกต้องร้อยละ 80



(1) ชุดข้อมูลที่ได้รับมาจากคำสารภาพของนิสิต (2) การจัดกลุ่มโดยค่าขีดแบ่งความคล้ายคลึงกัน

ภาพที่ 28 ตัวอย่างชุดข้อมูลที่ถูกสร้างขึ้นจากค่าขีดแบ่งความคล้ายคลึงกัน

การทดลองที่ 2 เนื่องจากเราทราบอยู่แล้วว่ากลุ่มนี้ลอกกันจริงๆ การวัดความถูกต้องจึงพิจารณาดังนี้ ถ้ากฎสามารถระบุต้นตอการคัดลอกรหัสต้นฉบับ ซึ่งเป็นโหนดรากของกลุ่มได้ ให้สรุปว่า การระบุต้นตอของการคัดลอกรหัสต้นฉบับได้ถูกต้อง เพราะฉะนั้นวิธีการนี้จึงมีความง่ายไม่ซับซ้อนในการวัดประสิทธิภาพด้วยร้อยละความถูกต้อง โดยการนำผลลัพธ์ที่กฎระบุต้นตอการคัดลอกรหัสต้นฉบับถูกต้องทั้งหมดหารด้วยจำนวนกลุ่มที่ได้รับจากนิสิตให้การรับสารภาพ แล้วจึง

นำมาคูณด้วยร้อยเพื่อเปลี่ยนค่าเป็นร้อยละ เช่น ถ้ากฎระบุต้นตอการคัดลอกรหัสต้นฉบับได้ถูกต้องทั้งหมดในทุกกลุ่มของทุกข้อในการบ้าน จำนวน 68 กลุ่ม ซึ่งกลุ่มที่ได้รับมาจากการให้การสารภาพของนิสิต มีจำนวนทั้งหมด 70 ดังนั้น ค่าความถูกต้องจึงเป็นร้อยละ 97.14

4.2 ผลการทดลอง

4.2.1 การเตรียมข้อมูลสำหรับการทดลอง

จากที่ได้กล่าวไว้แล้วในข้อที่ 4.1.1 เกี่ยวกับการเตรียมข้อมูลของวิธีการนี้ ผู้วิจัยได้นำข้อมูลจากการส่งการบ้านของนิสิตชั้นปีที่ 1 ซึ่งทราบว่าใครเป็นต้นตอของการคัดลอกรหัสต้นฉบับแล้ว แต่ด้วยชุดข้อมูลที่ได้นำมาใช้มีนิสิตบางรายไม่ได้กรอกคำเฉลยหรือบางรายกรอกผิดพลาดตามที่ได้กล่าวมาแล้ว ดังนั้น ผู้วิจัยจึงได้เลือก และตรวจสอบข้อมูลที่มีความถูกต้องมากที่สุด โดยจำนวนของรหัสต้นฉบับที่นำไปใช้ในกฎที่ผู้วิจัยนำเสนอ และทดสอบกฎที่นำเสนอ เพื่อระบุต้นตอของการคัดลอกรหัสต้นฉบับ ปรากฏตามตารางที่ 8

ตารางที่ 8 ชุดข้อมูลสำหรับการทดลอง

ชุดข้อมูล	จำนวนข้อย่อย	จำนวนรหัสต้นฉบับ
การบ้านที่ 1	8	167
การบ้านที่ 2	7	158
การบ้านที่ 3	10	228
การบ้านที่ 4	22	412

4.2.2 ชุดข้อมูลที่นำไปใช้ในกฎที่สร้างขึ้น

จากที่ได้กล่าวไปข้างต้น ชุดข้อมูลซึ่งจะนำไปใช้ในกฎที่สร้างขึ้นในการระบุต้นตอของการคัดลอกรหัสต้นฉบับ แบ่งออกเป็น 3 ส่วน ประกอบ (1) เมทริกซ์ความสัมพันธ์ (2) เวลาในการส่งการบ้าน (3) เกรดเฉลี่ยสะสม

4.2.2.1 การสร้างเมทริกซ์ความสัมพันธ์การเป็นต้นฉบับ – การคัดลอก ผู้วิจัยเลือกใช้รหัสต้นฉบับในการบ้าน 1 – 3 ซึ่งจะได้เมทริกซ์ขนาด 193×193 จำนวน 3 เมทริกซ์ ซึ่งแยกออกเป็นแต่ละการบ้าน

4.2.2.2 สำหรับข้อมูลที่นำมาใช้ในกฎที่ถูกสร้างขึ้นในส่วนที่ 2 และส่วนที่ 3 ผู้วิจัยได้ทดสอบความสัมพันธ์ของ เวลาและเกรดว่าสามารถนำมาช่วยระบุต้นฉบับได้หรือไม่ โดยทดลองกับการบ้าน 1 – 3 ผลการเปรียบเทียบเวลาส่งการบ้านก่อน พบว่าสามารถระบุการเป็นต้นตอของการคัดลอกรหัสต้นฉบับ โดยมีค่าความถูกต้องเฉลี่ยร้อยละ 91.22 จะเห็นว่าเวลาการส่งการบ้านก่อนมีผลในระดับสูงที่บ่งชี้การเป็นต้นตอของการคัดลอกรหัสต้นฉบับ และสำหรับเกรดเฉลี่ย เนื่องจากกลุ่มที่ทดลองเป็นนิสิตชั้นปี 1 ที่ศึกษาอยู่เทอม 2 จึงนำเกรดเฉลี่ยสะสมในเทอม 1 มาพิจารณาร่วมด้วย ทั้งนี้ ในวิชาส่วนใหญ่ยังเป็นการเรียนเกี่ยวกับสาขาวิชาทั่วไปยังไม่ใช่การเรียนในสาขาที่เกี่ยวข้องกับเทคโนโลยี ทำให้คุณสมบัติของเกรดเฉลี่ยยังมีผลการจำแนกในระดับไม่สูงมากเพียงร้อยละ 64.13 อย่างไรก็ตาม ก็สามารถนำทั้งสองส่วนนี้มาพิจารณาในการระบุต้นฉบับได้ นอกจากนี้ ผู้วิจัยได้ตรวจสอบกลุ่มบุคคลที่มีพฤติกรรมการลอกเลียนในการบ้านที่ 1-3 โดยเลือกบุคคลที่มีจำนวนการลอกเลียนแบบจำนวนมากกว่า ผลบวกของค่าเฉลี่ยกับค่าส่วนเบี่ยงเบนมาตรฐาน พบว่า เกรดโดยเฉลี่ยของบุคคลดังกล่าวมีผลการเรียนในวิชาการเขียนโปรแกรมคือเกรด D

4.2.3 ผลการทดสอบกฎที่นำเสนอเพื่อระบุต้นตอของการคัดลอกรหัสต้นฉบับ

ในการทดสอบประสิทธิภาพของขั้นตอนวิธีเพื่อค้นหาต้นตอของการคัดลอกรหัสต้นฉบับซึ่งได้กำหนดไว้ในข้างต้น เมื่อนำข้อมูลที่ใช้สำหรับการทดสอบกฎที่นำเสนอ ซึ่งจะอธิบายเป็น 2 กรณี ประกอบด้วย กลุ่มที่ได้จากค่าขีดแบ่งความคล้ายคลึงกันด้วยโปรแกรม JPlag กับข้อมูลกลุ่มที่มีการลอกเลียนแบบแล้ว และผลลัพธ์ของการจำแนกการเป็นต้นตอของการคัดลอกรหัสต้นฉบับ แสดงดังตารางที่ 9 และ 10 ตามลำดับ

ตารางที่ 9 ผลการทดลองของวิธีการทดลองที่ 1

ข้อที่	จำนวนกลุ่มที่ได้จากการใช้ค่าขีดแบ่งความคล้ายคลึงกัน	จำนวนกลุ่มที่กฎสามารถระบุต้นตอของการคัดลอกได้ถูกต้อง
1	6	4
2	3	3
3	8	7
4	7	6
5	8	5
6	6	3
7	4	1
8	4	3
9	3	1
10	6	5
11	4	2
12	3	1
13	2	1
14	3	1
15	5	3
16	3	2
17	3	3
18	3	2
19	3	2
20	3	2
21	7	7
22	4	4
รวม	98	68

ตารางที่ 10 ผลการทดลองของวิธีการทดลองที่ 2

ข้อที่	จำนวนกลุ่มที่ได้รับมาจากการสารภาพของนิสิต	จำนวนกลุ่มที่กฎสามารถระบุต้นตอของการคัดลอกได้ถูกต้อง
1	10	9
2	6	6
3	9	8
4	12	11
5	11	9
6	9	8
7	4	3
8	5	5
9	2	1
10	6	5
11	4	4
12	3	3
13	2	2
14	3	2
15	5	5
16	6	6
17	4	4
18	3	3
19	4	4
20	3	3
21	8	7
22	7	7
รวม	126	115

ตารางที่ 9 เป็นผลการทดลองที่ได้จากการทดลองที่ 1 เราสามารถนำผลที่ได้มาคำนวณหาประสิทธิภาพของกฎที่ใช้ในการทดลองนี้ โดยนำจำนวนกลุ่มทั้งหมดที่กฎสามารถระบุต้นตอการคัดลอกรหัสต้นฉบับได้อย่างถูกต้อง ซึ่งได้เท่ากับ 68 จากจำนวนกลุ่มทั้งหมดที่ได้รับจากค่าขีดแบ่งในแต่ละข้อของการบ้าน คือ 98 ทำให้ค่าความถูกต้องมีค่าเท่ากับร้อยละ 69.39 และจากตารางที่ 10 เป็นผลการทดลองที่ได้จากการทดลองที่ 2 เราจึงหาค่าความถูกต้องเพื่อบอกประสิทธิภาพของกฎในวิธีการนี้ ด้วยนำจำนวนกลุ่มทั้งหมดที่กฎสามารถระบุต้นตอของการคัดลอกรหัสต้นฉบับได้อย่างถูกต้อง มีค่าเท่ากับ 115 และจำนวนกลุ่มทั้งหมดในแต่ละข้อของการบ้านซึ่งได้รับจากให้การสารภาพของนิสิต มีจำนวน 126 ดังนั้น ค่าความถูกต้องจึงมีค่าเท่ากับร้อยละ 91.27

4.2.4 การวิเคราะห์ผลการทดลอง

จากผลการทดลองจะเห็นได้ว่า กรณีที่สมมติว่าทราบกลุ่มการลอกเลียนแบบ (การทดลองที่ 2) ร้อยละความถูกต้องสูงกว่า ในกรณีที่ใช้ค่าขีดแบ่งความคล้ายคลึงกัน (การทดลองที่ 1) แต่ทั้งสองวิธีไม่สามารถเปรียบเทียบผลลัพธ์กันได้ เนื่องจากการคำนวณใช้ข้อมูลที่ต่างกัน ซึ่งเหตุผลที่แตกต่างกันได้อธิบายแล้วว่า ข้อมูลชุดทดสอบ ตลอดจนข้อมูลในการเรียนรู้ ยังมีข้อผิดพลาดหรือยังเป็นข้อมูลที่ไม่สมบูรณ์จึงอาจจะยังไม่สามารถนำมาวิเคราะห์ได้อย่างถูกต้องแม่นยำ ซึ่งจากการสังเกตและตรวจสอบข้อมูลดังกล่าว ยังมีผู้ที่ไม่ได้ให้คำตอบตามความเป็นจริงหรือยังมีบางคนยังไม่ได้เข้ามากรอกข้อมูลของการคัดลอกรหัสต้นฉบับ ด้วยเหตุผลนี้เอง ในการจัดกลุ่มหากไม่สามารถได้ข้อมูลที่ครบถ้วนสมบูรณ์ เมื่อนำมาจัดกลุ่มโดยใช้ค่าความคล้ายคลึงกัน ทำให้กลุ่มของข้อมูลบางส่วนรวมหรือแยกออกจากกัน ทำให้การทดลองในการทดลองที่ 1 ผู้วิจัยจึงได้กำหนดการวัดประสิทธิภาพของการตรวจสอบความถูกต้องไว้ ต่างจากการทดลองที่ 2 ทั้งนี้ เพื่อใช้เป็นกรณีศึกษานำไปต่อยอดและพัฒนาให้เกิดความสมบูรณ์และถูกต้องแม่นยำในงานวิจัยในอนาคต แต่ถึงอย่างไรหากนำผลของการทดลองที่ 2 มาพิจารณาจะเห็นว่ากฎที่ได้กำหนดไว้ให้ค่าความถูกต้องที่แม่นยำ ถ้าหากมีกลุ่มการลอกเลียนแบบที่แน่นอน ดังนั้น ปัจจัยของจำนวนของการเป็นต้นฉบับของการบ้านที่ผ่านมา เวลาของการส่งงานก่อนหลัง ตลอดจนเกรดเฉลี่ยสะสม สามารถนำมาพิจารณาจำแนกเป็นต้นตอการคัดลอกรหัสต้นฉบับได้ จากผลการวิเคราะห์ข้อมูลจึงสามารถอธิบายความสัมพันธ์ของข้อมูลได้ว่า ผู้ที่มีพฤติกรรมการลอกเลียนแบบส่วนใหญ่จะคัดลอกรหัสต้นฉบับในกลุ่มเดิมซ้ำๆ เวลาการส่งก่อนหรือส่งหลัง สามารถช่วยบ่งชี้ว่าใครเป็นต้นฉบับได้

นอกจากนี้ มีข้อสังเกตจากคณะกรรมที่ว่าหากเอากฎที่นำเสนอไปใช้จริง การที่เราจะทราบว่าใครลอกใครบ้างนั้น อาจไม่สามารถนำมาใช้ได้ เพราะจากการทดลองจะเห็นได้ว่า นิสิตบางรายก็ไม่ได้ให้คำตอบหรือสารภาพ อีกทั้งการที่จะสอบสวนเพื่อให้นิสิตสารภาพการบ้านในอดีต เพื่อนำไปใช้ระบุต้นฉบับในอนาคตมันอาจไม่สามารถนำไปใช้ได้ เนื่องจากผู้ลอกเลียนแบบหรือผู้เป็นต้นฉบับอาจไม่ลอกเลียนในกลุ่มเดิมๆ จึงไม่ยุติธรรมต่อนิสิตในการชี้ว่าบุคคลดังกล่าวลอกเลียนบุคคลเดิมๆ หรือบุคคลเดิมๆ จะเป็นต้นฉบับอีก ผู้วิจัย จึงได้นำไปวิเคราะห์เพิ่มเติมและเห็นว่า ด้วยเหตุผลของข้อสังเกตเหล่านี้มีมูลเหตุที่เชื่อได้ว่า ข้อมูลเมทริกซ์การลอกเลียนแบบอาจจะไม่เหมาะสมในการนำมาพิจารณา แต่อย่างไรก็ตาม วิธีการนี้เวลาการส่งและเกรดสะสมยังอาจจะสามารถช่วยในการระบุต้นตอการคัดลอกได้

บทที่ 5
อภิปรายและสรุปผล

5.1 สรุปผลการวิจัย

ในงานวิจัยนี้ได้นำเสนอวิธีการระบุต้นตอของการคัดลอกรหัสต้นฉบับจากการบ้านวิชาการเขียนโปรแกรม โดยจะนำเสนอแบ่งออกเป็น 2 วิธีการ วิธีการที่ 1 ระบุต้นตอการคัดลอกรหัสต้นฉบับด้วยวิธีการจำแนก 6 เทคนิค ประกอบด้วย Decision Tree, REP Tree, Random Forest, Neural Network, K-Nearest Neighbor, และ Naïve Bayes โดยผู้วิจัยได้ศึกษารูปแบบการลอกเลียนแบบการเขียนโปรแกรมของการบ้านนิสิตชั้นปีที่ 1 จำนวน 140 โปรแกรม พบว่า เมื่อนำรหัสต้นฉบับ (Source codes) ดังกล่าวไปหาค่าความคล้ายคลึงกันด้วยโปรแกรม JPlag และเลือกเฉพาะค่าที่สูงกว่าร้อยละ 60 ทำให้เห็นระดับการลอกเลียนการเขียนโปรแกรมโดยส่วนใหญ่ไม่เกินระดับ 4 แต่เนื่องจากโปรแกรมที่ได้นำมาศึกษานี้ ผู้วิจัยไม่ทราบว่าบุคคลใดเป็นต้นฉบับ จึงจำเป็นต้องสร้างโปรแกรมโดยให้มีรูปแบบการลอกเลียนแบบในลักษณะในระดับการลอกเลียนแบบไม่เกินระดับ 4 ตามลักษณะโปรแกรมที่ได้นำมาศึกษา ซึ่งชุดโปรแกรมที่ได้สร้างขึ้นมีจำนวน 78 โปรแกรม ทั้งนี้ โปรแกรมที่เป็นตันฉบับมีจำนวน 10 โปรแกรม และโปรแกรมที่มีการคัดลอกมีจำนวน 68 โปรแกรม สำหรับการทดลองด้วยวิธีการจำแนกนี้จะใช้คุณลักษณะในการเรียนรู้ทั้งหมด 17 ชนิด เช่น ค่าความคล้ายกันระหว่างโปรแกรม, จำนวนบรรทัดที่คล้ายคลึงกัน ซึ่งข้อมูลจะถูกแบ่งนำมาเรียนรู้ และนำไปใช้ในการทดสอบ หลังจากเมื่อนำชุดข้อมูลไปทดลองกับวิธีการจำแนกด้วย 6 เทคนิค ผลลัพธ์ที่ได้ขั้นตอนวิธี Decision Tree และ REP Tree ให้ค่าร้อยละความถูกต้องมากที่สุดกล่าวคือ ร้อยละ 82.61 แต่อย่างไรก็ตามด้วยข้อจำกัดของวิธีการนี้ซึ่งเป็นการเปรียบเทียบระหว่างโปรแกรม เช่น โปรแกรม A เทียบโปรแกรม B จะบอกได้ว่า A คัดลอก B, B คัดลอก A, หรือ โปรแกรมทั้งสองไม่ได้คัดลอกกัน ซึ่งไม่สามารถบอกได้ว่าได้ลอกบุคคลอื่นหรือไม่ และ ด้วยข้อจำกัดของวิธีการจำแนก โมเดลที่ได้มีความเหมาะสมกับข้อมูลที่มีลักษณะแบบเดิม หากมีข้อมูลที่เปลี่ยนแปลงไปจำเป็นต้องมีการเรียนรู้ใหม่ นอกจากนี้ ในการระบุแบบลำดับชั้นของวิธีการนี้ด้วยข้อมูลที่ได้ศึกษายังมีไม่เพียงพอจึงอาจทำให้เกิดข้อผิดพลาดได้ง่าย ทำให้อาจไม่สามารถระบุต้นตอการคัดลอกได้ ผู้วิจัยจึงได้นำเสนอวิธีการอีกหนึ่งวิธีการ ซึ่งเป็นการระบุต้นตอของรหัสต้นฉบับของกลุ่มที่มีการคัดลอกโดยใช้ข้อมูลการลอกกันในอดีตมาอนุมาน กำหนดวิธีการทดลองใน 2 กรณี ได้แก่ การทดลองจากชุดข้อมูลซึ่งได้จัดกลุ่มโดยค่าความคล้ายคลึงกัน

และการทดลองจากชุดข้อมูลที่ทราบว่าใครลอกใครจากที่นิสิตสารภาพ ข้อมูลรหัสต้นฉบับที่ใช้ในการสร้างกฎได้มาจากการบ้านวิชาการเขียนโปรแกรม จำนวน 553 โปรแกรม และข้อมูลสำหรับการทดสอบกฎที่นำเสนอ จำนวน 412 โปรแกรม ซึ่งชุดข้อมูลเหล่านี้มีเฉลยว่าใครเป็นต้นฉบับแล้วเพื่อสามารถใช้ยืนยันคำตอบได้อย่างถูกต้อง นอกจากนี้ยังใช้ข้อมูลเวลาของการส่งการบ้าน และเกรดเฉลี่ยสะสมของแต่ละบุคคลมาช่วยพิจารณาเพื่อหาวิธีการระบุต้นฉบับจากกฎที่ผู้วิจัยได้นำเสนออีกด้วย และผลการทดสอบกฎพบว่า ผลลัพธ์ของกรณีที่ 1 การจัดกลุ่มด้วยค่าขีดแบ่งความคล้ายคลึงกันด้วยโปรแกรม JPlag ให้ค่าความถูกต้องร้อยละ 69.39 และสำหรับกรณีที่ 2 เป็นกรณีที่สมมติว่าทราบกลุ่มการลอกเลียนแบบแล้ว ให้ผลลัพธ์ค่าความถูกต้องร้อยละ 91.27 จะเห็นได้ว่าหากข้อมูลถูกจัดกลุ่มได้อย่างถูกต้อง ขั้นตอนวิธีในการระบุต้นตอของการคัดลอกรหัสต้นฉบับมีค่าความถูกต้องที่สูงปรากฏตามผลการทดลองรูปแบบที่ 2 แต่อย่างไรก็ตามในความเป็นจริงการรู้กลุ่มการลอกเลียนแบบไม่ใช่เรื่องง่าย จึงต้องมีการจัดกลุ่มโดยใช้ข้อมูลต่างๆ มาประกอบซึ่งค่าความคล้ายคลึงก็เป็นปัจจัยหนึ่งที่ช่วยให้สามารถดำเนินการดังกล่าวได้ดี อย่างไรก็ตาม ข้อจำกัดของวิธีการที่ 2 นี้ คือ ข้อมูลเมทริกซ์การลอกเลียนแบบที่ถูกนำมาใช้ในกฎที่นำเสนอในความเป็นจริงเป็นเรื่องยากที่จะได้มา และเมื่อมีการสารภาพจากนิสิตในการบ้านก่อน ก็อาจทำให้พฤติกรรมของนิสิตในการลอกเลียนแบบในครั้งถัดไปก็อาจจะเปลี่ยนไปซึ่งไม่ได้คัดลอกจากคนเดิม หรือบุคคลที่เป็นต้นฉบับอาจไม่ให้ลอกเลียนซึ่งข้อมูลในอดีตจึงไม่สามารถตัดสินได้

5.2 ข้อเสนอแนะและแนวทางการวิจัยในอนาคต

5.2.1 จากการทดลองทั้ง 2 วิธี เนื่องด้วยผู้วิจัยสนใจเพียงร้อยละค่าความถูกต้อง จึงไม่ได้วัดประสิทธิภาพในค่าอื่น ได้แก่ Precision, และ Recall ดังนั้น เพื่อให้การวัดประสิทธิภาพได้อย่างถูกต้อง ในการวิจัยในอนาคตจึงควรนำค่าเหล่านี้มาพิจารณาและวัดประสิทธิภาพของโมเดลหรือกฎต่างๆด้วย รวมทั้งการทำ confusion matrix ก็อาจช่วยให้สามารถวิเคราะห์ข้อผิดพลาดของโมเดลได้ดียิ่งขึ้น

5.2.2 จากข้อสังเกตของคณะกรรมการอีกข้อ คือผู้วิจัยไม่ได้นำข้อมูลจริงที่ได้จากการสารภาพของผู้เรียนไปทดลองการด้วยวิธีการที่ 1 (วิธีการจำแนกข้อมูลเพื่อระบุต้นตอการคัดลอก) ดังนั้น ในอนาคต การทดลองนี้จึงน่าจะเป็นประโยชน์ เพื่อที่จะได้ทราบและเปรียบเทียบขั้นตอนวิธีการจำแนกว่า สามารถนำไปใช้กับข้อมูลจริงได้มากน้อยเพียงใด

5.2.3 จากการศึกษาและได้รับข้อสังเกตจากคณะกรรมการสอบ ในวิธีการที่ 2 ซึ่งได้ใช้การอนุมานด้วยข้อมูลในอดีตยังมีข้อจำกัดและปัญหาที่ไม่อาจสามารถนำกฎที่นำเสนอไปใช้จริงได้ ผู้วิจัยจึงนำข้อจำกัดและปัญหาที่พบมาวิเคราะห์เพื่อเสนอแนวทางของการศึกษาวิจัยในอนาคต

ประการแรก ผู้วิจัยเห็นว่า กฎที่นำเสนอโดยใช้ชุดข้อมูลที่ได้จากการยอมรับสารภาพการคัดลอกหรือเป็นต้นตอของการคัดลอกรหัสต้นฉบับ (เมทริกซ์การลอกเลียนแบบ) สำหรับในการใช้งานจริงแล้วนั้น การได้มาซึ่งชุดข้อมูลนี้เป็นเรื่องยาก เนื่องจากบุคคลที่เป็นผู้ที่คัดลอกหรือเป็นต้นฉบับอาจไม่ยอมรับการกระทำดังกล่าว เห็นได้จากชุดข้อมูลที่นำมาวิจัยที่มีนิสิตบางรายไม่ได้ให้ข้อมูลตามความเป็นจริง หรือการสารภาพจากบุคคลโดยทั่วไปแล้วก็ไม่อาจเชื่อถือได้มาก การที่นำข้อมูลนี้มาใช้ระบุการเป็นต้นตอการคัดลอกรหัสต้นฉบับ จึงอาจไม่ยุติธรรมต่อบุคคลที่ถูกระบุว่าเป็นต้นตอของการคัดลอกรหัสต้นฉบับ

ประการที่ 2 วิธีการระบุต้นตอของการคัดลอกรหัสต้นฉบับ โดยการอนุมานด้วยข้อมูลในอดีต ผู้วิจัยเห็นว่า การนำข้อมูลในอดีตมาระบุการกระทำในปัจจุบันเลยนั้น เป็นข้อสันนิษฐานที่ไม่ค่อยสมเหตุสมผล เช่น อดีตเคยเป็นผู้ที่ก่อเหตุข่มขืน ปัจจุบันมีเหตุในลักษณะเดียวกันจึงระบุเลยว่าคนดังกล่าวเป็นผู้กระทำ ผู้วิจัยจึงเห็นว่า ข้อมูลในอดีตอาจจะนำมาใช้เพื่อช่วยสร้างความเชื่อมั่นในการตัดสินใจเพียงเท่านั้นได้ ไม่อาจสามารถนำไปชี้วัดโดยตรงได้ นอกจากนี้ การชี้วัดบุคคลมีพฤติกรรมที่ไม่ถูกต้องเพียงแค่หลักฐานอย่างเดียว ขาดการใช้ตรรกะหรือการพิจารณา มันเป็นการที่ไม่ยุติธรรมต่อผู้ที่ได้ผลกระทบจากการระบุพฤติกรรมดังกล่าว

ดังนั้น ด้วยเหตุผลทั้งสองประการ ผู้วิจัยจึงขอนำเสนอวิธีการใหม่ที่เป็นการหาค่าความเชื่อมั่นการเป็นต้นตอของการคัดลอกรหัสต้นฉบับ เพื่อใช้เป็นทางเลือกให้ผู้สอนนำไปพิจารณาตัดสินใจว่า บุคคลใดเป็นต้นฉบับในแต่ละกลุ่มที่ลอกเลียนแบบ โดยมีวิธีการ ดังนี้

1) วิธีการหาค่าความเชื่อมั่นของการเป็นต้นตอการคัดลอกรหัสต้นฉบับ โดยใช้ข้อมูลในอดีตและปัจจุบัน ซึ่งชุดข้อมูลที่นำมาใช้ประกอบด้วย ค่าความเชื่อมั่นของการจำแนกต้นตอการคัดลอกรหัสต้นฉบับด้วยกฎ Decision Tree (กฎที่ได้รับจากวิธีการที่ 1 ของงานวิจัยนี้), ค่าความเชื่อมั่นของการระบุต้นตอการคัดลอกรหัสต้นฉบับด้วยเวลาส่งการบ้าน, ค่าความเชื่อมั่นของการระบุต้นตอของการคัดลอกรหัสต้นฉบับด้วยเกรดเฉลี่ยสะสม

2) ก่อนที่จะนำเสนอกฎที่จะนำมาใช้ในวิธีการใหม่นี้ ตามที่ได้กล่าวในตอนต้นว่าวิธีการใหม่นี้จะเสนอการหาค่าความเชื่อมั่นในกลุ่มที่มีการลอกเลียนแบบ เพราะฉะนั้นผู้จัยจะขอ

อธิบายวิธีเพื่อจัดกลุ่มด้วยค่าขีดแบ่งความคล้ายคลึงกัน โดยใช้วิธีการจัดกลุ่มเช่นเดียวกับกับการทดลองที่ 1 ของวิธีการที่ 2 การระบุต้นตอของการคัดลอกรหัสต้นฉบับจากการอนุมานด้วยข้อมูลในอดีต

3) เมื่อได้กลุ่มต่างๆ ออกมาแล้ว นำมาคำนวณหาค่าความเชื่อมั่นการเป็นต้นตอของการคัดลอกรหัสต้นฉบับจากกฎที่จะนำเสนอต่อไปนี้

ค่าความเชื่อมั่นการเป็นต้นตอของการคัดลอกรหัสต้นฉบับของบุคคลรายหนึ่ง	=	(ค่าความเชื่อมั่นของการเป็นต้นตอของการคัดลอกรหัสต้นในปัจจุบันของบุคคลรายหนึ่ง+ค่าเฉลี่ยของค่าความเชื่อมันของการเป็นต้นตอของการคัดลอกรหัสต้นฉบับในอดีตของบุคคลรายหนึ่ง)/2
ค่าความเชื่อมั่นของการเป็นต้นตอของการคัดลอกรหัสต้นในปัจจุบันของบุคคลรายหนึ่ง	=	(ค่าความเชื่อมั่นของการจำแนกต้นตอการคัดลอกรหัสต้นฉบับด้วยกฎ Decision Tree +ค่าความเชื่อมั่นของการระบุต้นตอการคัดลอกรหัสต้นฉบับด้วยเวลาส่งการบ้าน+ค่าความเชื่อมั่นของการระบุต้นตอของการคัดลอกรหัสต้นฉบับด้วยเกรดเฉลี่ยสะสม)/3
ค่าเฉลี่ยของค่าความเชื่อมันของการเป็นต้นตอของการคัดลอกรหัสต้นฉบับในอดีตของบุคคลรายหนึ่ง	=	(ค่าเฉลี่ยของค่าความเชื่อมันของการเป็นต้นตอของการคัดลอกรหัสต้นฉบับในอดีตของบุคคลรายหนึ่งสำหรับการบ้านที่ 1+ค่าเฉลี่ยของค่าความเชื่อมันของการเป็นต้นตอของการคัดลอกรหัสต้นฉบับในอดีตของบุคคลรายหนึ่งสำหรับการบ้านที่ 2+...+ค่าเฉลี่ยของค่าความเชื่อมันของการเป็นต้นตอของการคัดลอกรหัสต้นฉบับในอดีตของบุคคลรายหนึ่งสำหรับการบ้านที่ n)/n

4) สำหรับการคำนวณหาค่าความเชื่อมั่นของการเป็นต้นตอของการคัดลอกรหัสต้นฉบับของทั้ง 3 ข้อมูลที่นำมาใช้ในกฎที่ได้นำเสนอ มีวิธีคำนวณดังนี้

4.1) ค่าความเชื่อมั่นของการจำแนกต้นตอการคัดลอกรหัสต้นฉบับด้วยกฎ Decision Tree

4.1.1) นำกลุ่มที่ได้จากข้อ 2) มาจำแนกด้วยกฎที่ได้รับจากวิธีการที่ 1 มาสร้างโครงสร้างต้นไม้การลอกเลียนแบบรหัสต้นฉบับ

4.1.2) เมื่อได้โครงสร้างต้นไม้แล้ว ลำดับถัดไปกำหนดค่าความเชื่อมั่นการเป็นต้นตอของการคัดลอกรหัสต้นฉบับโดยให้โหนดรากซึ่งคือต้นฉบับมีค่าความเชื่อมั่นการปันต้นตอของการคัดลอกเท่ากับ 0.8261 โดยค่าดังกล่าวกำหนดตามค่าความถูกต้องซึ่งได้จากงานวิจัยในวิธีการที่ 1

4.1.3) นอกจากรหัสต้นฉบับที่เป็นโหนดราก กำหนดค่าความเชื่อมั่นการเป็นต้นตอของการคัดลอกรหัสต้นฉบับกับโหนดอื่นมีค่าเท่ากับ 0

4.2) ค่าความเชื่อมั่นของการระบุต้นตอการคัดลอกรหัสต้นฉบับด้วยเวลาส่งการบ้าน

4.2.1) นำกลุ่มที่ได้จากข้อ 2) มาหาต้นตอของการคัดลอกรหัสต้นฉบับโดยให้พิจารณา ถ้ารหัสต้นฉบับใดส่งการบ้านก่อน ให้ถือว่ารหัสต้นฉบับนั้นเป็นต้นตอของการคัดลอกรหัสต้นฉบับ

4.2.2) เมื่อได้ต้นตอของการคัดลอกรหัสต้นฉบับในกลุ่ม กำหนดให้ค่าความเชื่อมั่นการเป็นต้นตอของการคัดลอกรหัสต้นฉบับมีค่า 0.9122 โดยค่าดังกล่าวกำหนดตามค่าความถูกต้องซึ่งได้จากงานวิจัยในวิธีการที่ 2

4.2.3) นอกจากรหัสต้นฉบับที่เป็นต้นตอของการคัดลอกแล้ว โหนดที่เหลือภายในกลุ่มให้กำหนดค่าความเชื่อมั่นการเป็นต้นตอของการคัดลอกรหัวต้นฉบับนั้นมีค่าเท่ากับ 0

4.3) ค่าความเชื่อมั่นของการระบุต้นตอของการคัดลอกรหัสต้นฉบับด้วยเกรดเฉลี่ยสะสม

4.3.1) นำกลุ่มที่ได้จากข้อ 2) มาหาต้นตอของการคัดลอกรหัสต้นฉบับโดยให้พิจารณา ถ้ารหัสต้นฉบับใดมีเกรดเฉลี่ยสะสมมากที่สุดในกลุ่ม ให้ถือว่าเป็นต้นตอของการคัดลอกรหัสต้นฉบับ

4.3.2) เมื่อได้ต้นตอของการคัดลอกรหัสต้นฉบับแล้ว กำหนดให้ค่าความเชื่อมั่นการเป็นต้นตอของการคัดลอกรหัสต้นฉบับมีค่า 0.6413 เช่นเดียวกันเรากำหนดค่าดังกล่าวจากค่าความถูกต้องที่ได้จากงานวิจัย

4.3.3) นอกจากรหัสต้นฉบับที่เป็นต้นตอของการคัดลอกแล้ว โหนดที่เหลือภายในกลุ่มให้กำหนดค่าความเชื่อมั่นการเป็นต้นตอของการคัดลอกรหัวต้นฉบับนั้นมีค่าเท่ากับ 0

5) เมื่อทำทุกขั้นตอนเราจะได้เมทริกซ์ความเชื่อมั่นการเป็นต้นฉบับจำนวนเท่ากับการบ้านของผู้เรียน ผู้สอนก็จะสามารถดูประวัติความเชื่อมั่นการเป็นต้นตอของการคัดลอกรหัสต้นฉบับของนิสิตแต่ละคน ซึ่งสามารถนำไปช่วยพิจารณาในการหามาตราการป้องกันการคัดลอกต่อไปได้

ภาคผนวก

ที่ ๕๖/๒๕๖๒

เอกสารรับรองผลการพิจารณาจริยธรรมการวิจัยในมนุษย์
มหาวิทยาลัยบูรพา

คณะกรรมการพิจารณาจริยธรรมการวิจัยในมนุษย์ มหาวิทยาลัยบูรพา ได้พิจารณาโครงการวิจัย

รหัสโครงการวิจัย Sci 039/2562
โครงการวิจัยเรื่อง การระบุต้นตอของการคัดลอกรหัสต้นฉบับ
หัวหน้าโครงการวิจัย นายชวลิต เสาร์แบน
หน่วยงานที่สังกัด นิสิตระดับบัณฑิตศึกษา คณะวิทยาการสารสนเทศ

คณะกรรมการพิจารณาจริยธรรมการวิจัยในมนุษย์ มหาวิทยาลัยบูรพา ได้พิจารณาแล้วเห็นว่าโครงการวิจัยดังกล่าวเป็นไปตามหลักการของจริยธรรมการวิจัยในมนุษย์ โดยที่ผู้วิจัยเคารพสิทธิและศักดิ์ศรีในความเป็นมนุษย์ ไม่มีการล่วงละเมิดสิทธิ สวัสดิภาพ และไม่ก่อให้เกิดภยันตรายแก่ตัวอย่างการวิจัยและผู้เข้าร่วมโครงการวิจัย

จึงเห็นสมควรให้ดำเนินการวิจัยในขอบข่ายของโครงการวิจัยที่เสนอได้ (ดูตามเอกสารตรวจสอบ)

๑. เอกสารโครงการวิจัยฉบับภาษาไทย ฉบับที่ ๑ วันที่ ๑๙ เดือน เมษายน พ.ศ. ๒๕๖๒
๒. เอกสารชี้แจงผู้เข้าร่วมโครงการวิจัย ฉบับที่ - วันที่ - เดือน - พ.ศ. -
๓. เอกสารแบบแสดงความยินยอมของผู้เข้าร่วมโครงการวิจัย
ฉบับที่ - วันที่ - เดือน - พ.ศ. –
๔. เอกสารแสดงรายละเอียดเครื่องมือที่ใช้ในการวิจัยซึ่งผ่านการพิจารณาจากผู้ทรงคุณวุฒิแล้ว หรือชุดที่ใช้เก็บข้อมูลจริงจากผู้เข้าร่วมโครงการวิจัย ฉบับที่ - วันที่ - เดือน - พ.ศ. -

การรับรองผลการพิจารณาจริยธรรมการวิจัยในมนุษย์ฉบับนี้ มีผลถึงวันที่ ๑๘ เดือน เมษายน พ.ศ. ๒๕๖๓

ออกให้ ณ วันที่ ๑๙ เดือน เมษายน พ.ศ. ๒๕๖๒

ลงนาม
(ผู้ช่วยศาสตราจารย์ ดร.วิทวัส แจ้งเอี่ยม)
ประธานคณะกรรมการพิจารณาจริยธรรมการวิจัยในมนุษย์
มหาวิทยาลัยบูรพา

Identifying an original copy of the source codes in programming assignments

Chawalit Saoban
Faculty of Informatics
Burapha University
Chon Buri, Thailand 20131
60910062@go.buu.ac.th

Sunisa Rimcharoen
Faculty of Informatics
Burapha University
Chon Buri, Thailand 20131
rsunisa@buu.ac.th

***Abstract*—Source-code plagiarism in programming assignments is a serious issue. It can lead to bad consequences of students in personal and professional life. Students who copy someone else's source-code did not learn anything. This improper behavior is unacceptable. It would be useful if teachers know who the owner of the original copy and someone is a copier. They may employ strategies to prevent the plagiarism and give advice to students who did wrong. This paper, therefore, proposes the methodology to identify the original copy of the source codes among the plagiarized programs. The data are collected from an introduction to programming course and some data are generated to test the algorithms. We use six classification techniques to classify the suspect programs i.e. Decision Tree, REP Tree, Random Forest, Neural Network, K-nearest Neighbor, and Naïve Bayes. The experimental results show that the decision tree algorithm performs best. It yields the accuracy of 82.61% in testing.**

Keywords—plagiarism, programming assignments, source code, classification

I. Introduction

Plagiarism is "the practice of taking someone else's work or ideas and passing them off as one's own." [1]. It occurs often to academic e.g. assignment, publication, research, etc. This problem must be considered as an important issue because it may affect academic integrity, society, industries, etc. If we are not well aware of this issue, new knowledges and technologies will not be developed because there is no new creative work. In a computer programming course, we also have a problem. Students copy source-codes from their classmates. Someone copies the whole source-codes or some parts without any changes. Someone changes the codes a little bit e.g. a variable name. This problem is serious and must be addressed carefully. Misbehavior and negative attitudes may cause students getting poor academic achievement and affecting their careers in the future.

According to [2], modification of source-codes can be characterized as six levels. Level 0 is keeping the source code without any changes. Level 1 is changing comments and indentations. Level 2 is changing the name of variables or functions. Level 3 is changing the variable, constant or function declaration. Level 4 is changing the program modules. Level 5 is changing the program statements and control structures. Level 6 is changing the program logic. There are two main approaches of the source code plagiarism detection [3] i.e. feature comparison and structure comparison. The feature comparison approach calculates similarity between two programs by comparing the features extracted from the source codes e.g. the number of comment lines, the number of indented lines. The structure comparison approach estimates the similarity of two programs by analyzing their structures e.g. parsing the programs, comparing their tokens. Thus, this methodology is difficult to detect plagiarized codes when the plagiarism level is 4 or higher [4].

There have been many pieces of research that propose the methods to detect source code plagiarism. Prechelt et al. [5] and Duracik et al. [6] proposed the evaluation of JPlag system in finding the similarity of plagiarized programs. Funseca [7] presented the approach for dealing with the large scale of source codes. Bandara et al. [8] introduced using students'programming styles to detect the plagiarism. They employ three machine learning techniques (i.e. Naïve Bayes, K-Nearest Neighbor, and AdaBoost Meta-Learning Algorithm) to learn structure-based properties of plagiarized programs. Ohno et al. [9] also uses a student's coding style incorporated with Hidden Markov model to verify the source-code owner. Schneider et al. [10] proposed the new approach to consider interaction logs collected during the creation process e.g. string insertion, cursor movement, copy and paste etc. All these researches attempt to detect a pair of source codes that seems to be similar, but do not aim to identify the original source code. Ji et al. [11] proposed the generating of source code plagiarism evolution tree, which can show the inheritance of source codes. They calculate an asymmetric similarity score of two programs based on a local alignment technique. They defined EvolDist(A, B) as an estimated distance from transforming program A into program B. If the EvolDist(A, B) is less than the EvolDist(B, A), then they concluded that program B is evolved from program A. Otherwise, program A is evolved from program B. After identifying the evolution direction, all pairs of programs associated with EvolDist are used to generate the evolution tree by finding a minimum spanning tree.

Most of the previous works did not focus on identifying the original copy of the source codes. However, we think that if we know the original, we can manage the plagiarism problem at the root cause. This is because if we keep an eye on the student who own the original source code and do not let him to share it to his friends, there is no source code to copy. Other students must do the assignment by themselves.

There was [11] that can identify the original one. They used the data sets obtained from the East-Asia ICPC (International Collegiate Programming Contest). The source codes are collected from competitors who have high programming skills. Therefore, structures of the plagiarized programs are complex and lie in a high level of plagiarism. In addition, these data sets are not public for researchers to do the experiments. In this paper, thus, we use the data sets obtained from the first-year-students' programming-assignments. There are 140 programs that have an average

978-1-7281-0719-6/19/$31.00 ©2019 IEEE

similarity computed by JPlag [5] of 61%, which is quite high. We observed the pairs of source codes that have similarity scores more than 60% and found that the plagiarism levels are not higher than level 4. Unfortunately, from these data, we do not know which a source code is original. We, then, create a data set by imitating students' plagiarism-styles. We have 10 original programs and the other 68 programs are derived from them. This data set is divided into training and testing data. We compare each pair of source codes using JPlag and the similarity scores are gathered. We also extract useful attributes from the source codes. The six classification techniques i.e. Decision Tree, REP Tree, Random Forest, Neural Network, K-nearest Neighbor, and Naïve Bayes are applied to classify the pair of source codes (program A and program B) into three classes: 1) program A is the original 2) program B is the original and 3) A and B are not plagiarized programs.

This paper is organized as follows. Section 2 presents a background about the tools for detecting source code plagiarism and classification techniques. Section 3 describes details of the proposed method. Section 4 shows the experiment results and analysis. Section 5 is a conclusion.

II. Background

A. Plagiarism detection tools

Most of the tools for detecting plagiarized programs work based on comparing the similarities of the program structures. This paper also uses similarity scores and other features to identify the original copy of the source codes. Recently, there are many tools for comparing the pair of programs e.g. JPlag [5], MOSS [14], SIM [15], etc. These tools have different properties. Hage et al. [12] and Ahadi et al. [13] published the papers that present the comparison of features of plagiarism detection tools in terms of current in used, supported language, open source, etc. In this paper, we select the tool that can run on websites (due to the reason that our students' programming-assignments are submitted via the website and it would be convenient to manage them online). From the survey [12-13], MOSS and JPlag are what we want. However, Moss run on the Moss server at Stanford University, it is inconvenient for us. Thus, we select JPlag to use in the experiments.

B. Classification algorithms

Classification is a method to identify a group or a category to the things. It is one of the supervised learning techniques for data mining. Each record in the training data contains the attribute known as the target value, which plays an important role in guiding the correct answer to a classification algorithm during the learning process. The algorithm learns from this data and construct a model to classify the data into a proper class. The following subsection will introduce the classification algorithms used in this paper.

- Decision Tree

Decision tree consists of root node, intermediate node, and leaf node. Root node and intermediate node contain the features that are tested for decision-making. Leaf node contains the classes that are the result of classification. J48 is one of the popular decision tree algorithms. It is an extension of ID3.

This algorithm finds the best feature that will be used as a decision node by calculating information gain, which is derived from entropy. This process continues until all data is classified.

- REP Tree

Reduced Error Pruning Tree (REP Tree) is a decision tree that uses the reduced-error pruning technique. The algorithm checks each internal node and try to replace the node with the most common class if it does not reduce the accuracy.

- Random Forest

Random forest constructs multiple models of decision trees. Each decision tree may has different data and features. Each tree is trained with different training data sampling with replacement from the data. Afterwards the class will be decided from the votes of all trees. This technique is called bagging ensemble or bootstrap aggregating.

- Neural Network

This paper uses Multi-Layer Perceptron algorithm (MLP) which is a feedforward neural network that consists of three main layers, i.e. input layer, hidden layer, and output layer. It employs a supervised learning technique called backpropagation for training. It learns by comparing the output of the network with the target output, and the error is propagated back to the previous layer. The weights are then adjusted.

- K-Nearest Neighbor

K-nearest neighbor is a classification technique, which does not create a model for prediction. It uses predefined K-value to consider K objects that have the nearest distances. Generally, the distance between X and Y is calculated using Euclidean Distance as follows:

$$D(x,y) = \sqrt{\sum_{i=1}^{n} (X_i - Y_i)^2} \tag{1}$$

- Naïve Bayes

This classification technique uses the Bayesian theorem. For creating the model, there are n attributes defined as: A1, A2, …, An. Let E is a vector < a1, a2, …, an >, where ai is the value of Ai. Let C is a class variable and c(E) is denoted as a class of E. The Bayesian classification is defined as (2).

$$c(E) = \underset{c \in C}{\operatorname{argmax}} P(c) P(a_1, a_2, \ldots, a_3 | c) \tag{2}$$

Assuming that all attributes are independent given the class label, thus, the Naïve Bayes is defined as:

$$c(E) = \underset{c \in C}{\operatorname{argmax}} P(c) \prod_{i=1}^{n} P(a_i | c) \tag{3}$$

III. METHODOLOGY

In this section, we present the method to identify the original copy of source codes. It consists of four steps:

1) Data preparation 2) Attribute selection 3) Algorithm and parameter settings 4) Performance evaluation.

B. Data Preparation

As mentioned above, that data sets used in the experiments are created by imitating student' plagiarism-styles. Because of the 140 programs from the students are not labeled. From the data we do not know which one is the original. Therefore, we use the styles of these source codes as the patterns to create the new ones. The following steps are how we create them.

1) Design three programming problems

2) Write source codes to solve those problems (the original ones), all the original codes must have different structures. All these source codes have similarity scores close to zero when use the JPlag tool to check the plagiarism. These programs can be viewed as root nodes.

3) Create the plagiarized programs which are derived from the programs in step 2), and then create the plagiarized programs that derived from the plagiarized ones (Level 2 and 3 of the tree) by changing some parts of the source codes according to level 1-4 [2].

We have 78 programs that will be used as the data sets in the experiments. The detail is shown in Table I. The problem 1 and 2 are used to train the model. The problem 3 is a test problem.

TABLE I. DESCRIPTION OF DATA SETS USED IN THE EXPERIMENTS

Problem / Description	Problem 1	Problem 2	Problem 3
The number of all source codes	34	30	14
The number of original source codes	4	4	2
Level of the tree	3	3	3

C. Attribute Selection

In this step, we extract information from the data by running the JPlag to compare each pair of source codes. The data obtained from JPlag are 1) the average similarities in each pair of programs 2) similarities of program A compared with B and program B compared with program A 3) groups of similar lines 4) the number of tokens in each group that have similar lines. An output obtained from JPlag is shown in Fig. 1. All the attributes used in this research show in Table II.

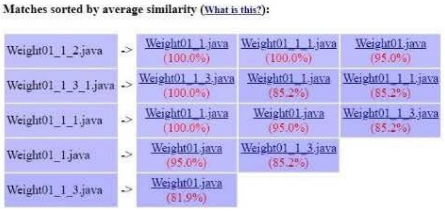


Fig. 1. A result of JPlag

D. Algorithm and Parameter Settings

We use six classification techniques for the experiments. All techniques run with Weka 3.83. The parameters are set as Table III.

TABLE III. PARAMETER SETTINGS

Techniques	Parameters	Values
Decision Tree (J48)	no pruning	yes
REP Tree	no pruning	yes
Random forest	min variance prop	0.003
	numFolds	2
	seed	2
Neuron Network (MLP)	hidden layer	5
	learning rate	0.1
	momentum	0.8
	training time	2000
K-Nearest Neighbor (Ibk)	k	2
Naïve Bayes	-	-

TABLE II. DESCRIPTION OF DATA SETS USED IN THE EXPERIMENTS

Attribute	Abbreviation	Description
Similarity Score	Thr_SIM	A similarity threshold in the comparison of Program A and Program B
Threshold of Program A	Thr_PA	A similarity threshold of Program A compared to Program B
Threshold of Program B	Thr_PB	A similarity threshold of Program B compared to Program A
The Number of Programs that are similar to Program A	Num_SPA	The number of all programs that are similar to Program A
The Number of Programs that are similar to Program B	Num_SPB	The number of all programs that are similar to Program B
Minimum Similarity of Program A	Min_PA	A minimum similarity threshold of Program A
Minimum Similarity of Program B	Min_PB	A minimum similarity threshold of Program B
Maximum Similarity of Program A	Max_PA	A maximum similarity threshold of Program A
Maximum Similarity of Program B	Max_PB	A maximum similarity threshold of Program B
Average Similarity of Program A	AVG_PA	An average similarity threshold of Program A
Average Similarity of Program B	AVG_PB	An average similarity threshold of Program B
Standard deviation of Program A	SD_PA	A standard deviation of similarity threshold of Program A
Standard deviation of Program B	SD_PB	A standard deviation of similarity threshold of Program B
The Number of Similar Lines of Program A	NSA	The number of all lines that are similar to program A when compared with program B
The Number of Similar Lines Program B	NSB	The number of all lines that are similar to program B when compared with program A
The Number of Similar Tokens	NST	The number of tokens in similar lines which is compared between program A and program B
Different Number of Lines	DLP	The number of differences in all similar lines between program A and program B
Classes	C	The result of classification 1) Program A is the original 2) Program B is the original and 3) They are not the plagiarize programs

E. Performance Evaluation

In the experiments, we need to know the performance of the classification techniques in identifying original source codes. We measure the performance of six techniques by comparing accuracies from classifying the testing set.

IV. Experiment Results

In the experiments, we use 60 programs to train the algorithms to create models for identifying the original source codes and uses 17 programs as the testing data set for evaluating the performance of the models. The experiment results are shown in Table IV.

TABLE IV. THE ACCURACIES OF THE ALGORITHMS IN IDENTIFYING THE ORIGINAL SOURCE CODES

Algorithm	Accuracies	
	Training (%)	*Testing (%)*
Decision Tree (J48)	94.74	82.61
REP Tree	93.23	82.61
Random Forest	100.00	73.91
Neural Network (MLP)	89.47	69.57
K-Nearest Neighbor (Ibk)	100.00	65.22
Navie Bayes	66.17	60.87

Table IV shows that the decision tree (J48) and the REP Tree yielded the best accuracy for the testing data. However, in the training data set the decision tree (J48) outperformed the REP Tree. Due to the fact that the REP tree is a pruned tree, it is obvious that the decision tree without pruning is fit to data and yields more accuracy than the REP tree. Therefore, in this paper, we will analyze the results from the decision tree. The rule obtained from the decision tree is shown in Fig. 3.

Regarding the random forest, which is created from multi-decision trees with different features and data, it yielded an accuracy of 100% for the training data, but it performed worse when it was applied to the testing data. This model is over fit. The neural network model with multi-layer perceptron, which used the backpropagation algorithm to learn from the error of the output compared with the target output, obtained less accuracy than the decision tree models. We observed that why the MLP performed worse than the decision tree models. The reason may come from the values in each attribute is quite similar. The weights of the MLP may not be adjusted to differentiate the classes. The KNN (k=2) also yielded the accuracy of 100% for the training data but the accuracy greatly reduces when applied to the testing data. This model is over fit to the training data. From the experiments, the Naïve Bayes earned the smallest accuracy. This is because this algorithm is suitable to nominal data. In our experiments, the attribute values have small ranges so that it may hard for Naïve Bayes to distinguish the classes.

To illustrate the classification's results, we represented the results as a phylogenetic tree in Fig. 2. There are 10 programs that are denoted as the nodes in the tree. Let an arrow (←) indicates the direction of imitation such as A←B denotes that program B is derived from program A.

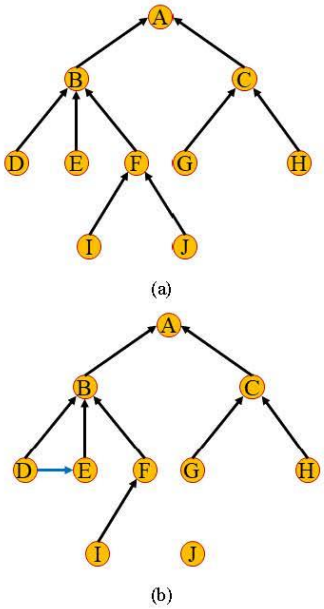


Fig. 2. An example of plagiarized programs that is represented as graphs: (a) the desired output and (b) the output obtained from our experiment

The graph in Fig. 2 (b) is the one that is constructed from using the rule presented in Fig. 3. Most of the original source codes are correctly identified using our proposed rule. It has only 2 incorrect pieces 1) it cannot correctly identify the plagiarism between node F and node J, 2) it decides that node D is derived from Node E, but in fact they are not plagiarized programs. The analysis of these misclassified cases will present as follows.

TABLE V. AN EXAMPLE OF ATTRIBUTE VALUES FROM TESTING SET

Node / Attribute	J and F	J and B	D and E	D and B
Thr_SIM	95.00	81.90	100.00	100.00
Thr_PA	95.08	81.97	100.00	100.00
Num_SPA	15	15	11	11
Min_PA	42.30	42.30	41.20	41.20
Max_PA	95.00	95.00	100.00	100.00
AVG_PA	72.15	72.15	85.33	85.33
SD_PA	19.84	19.84	15.81	15.81
Thr_PB	95.08	81.96	100.00	100.00
Num_SPB	13	11	11	11
Min_PB	41.20	41.20	41.20	41.2
Max_PB	100.00	100.00	100.00	100.00
AVG_PB	73.17	85.33	85.33	85.33
SD_PB	19.86	15.81	15.81	15.81
NSA	48	35	65	65
NSB	48	44	58	67
NST	58	50	61	61
DLP	0	9	7	2
Actual Class	B	C	C	B
Predicted Class	C	C	B	B

Table V shows the data that are used to decide the class. The last two rows show the target classes and the predicted classes. Our proposed rules yielded two cases of misclassification: node J-F and node D-E. Firstly, we will analyze node J-F that why the rule returned the predicted result as the class C instead of B. We observed the data of node J-F compared with node J-B as shown Table V. The rule can predict the class of node J-B correctly. From the proposed rule (Figure 3.) at the line number 29, "if AVG_PB <= 74.233333", the attribute AVG_PB of the node J-F is 73.17 while that of the node J-B is 85.33. Therefore, the node J-B is classified as the class C. In this case, the node J-F has the value of AVG_PB <= 74.233333, then it must be checked by the rule at the line number 32 "if Num_SPB <= 12". The value of the attribute Num_SPB of the node J-F is 13, which is grater than 12, therefore the class is predicted as the class C (indeed it should be a class B). For the later case, the misclassification of the node D-E occured from the rule at the line number 1, "if Thr_SIM <= 97.5". The value of the attribute Thr_SIM of the node D-E is 100, which is grater than 97.5, then it is classified as the class B. However, these are only two cases that the proposed rule cannot predict the class correctly. This may need more training data and more validation data to tune the model. We will take these issues into consideration in the future work.

```
1  if Thr_SIM <= 97.5
2  | if DLP <= 9
3  | | if Num_SPB <= 5
4  | | | if Num_SPA <= 7
5  | | | | if NSB <= 27 classify as C
6  | | | | else classify as B
7  | | | else
8  | | | | if Num_SPA <= 10 classify as A
9  | | | | else classify as C
10 | | else
11 | | | if NST <= 26 classify as C
12 | | | else
13 | | | | if AVG_PB <= 71.190909
14 | | | | | if Max_PA <= 97.5
15 | | | | | | if NSA <= 26
16 | | | | | | | if NST <= 27 classify as B
17 | | | | | | | else
18 | | | | | | | | if Min_PA <= 44 classify as C
19 | | | | | | | | else classify as A
20 | | | | | | else classify as B
21 | | | | | else
22 | | | | | | if SD_PA <= 16.398742 classify as C
23 | | | | | | else
24 | | | | | | | if DLP <= 1 classify as A
25 | | | | | | | else
26 | | | | | | | | if SD_PA <= 17.621168 classify as A
27 | | | | | | | | else classify as C
28 | | | | else
29 | | | | | if AVG_PB <= 74.233333
30 | | | | | | if Max_PB <= 97.5 classify as C
31 | | | | | | else
32 | | | | | | | if Num_SPB <= 12
33 | | | | | | | | if SD_PA <= 16.972676 classify as C
34 | | | | | | | | else classify as B
35 | | | | | | | else classify as C
36 | | | | | else classify as C
37 | else
38 | | if Num_SPB <= 11classify as A
39 | | else
40 | | | if Min_PB <= 42.3
41 | | | | if DLP <= 11classify as C
42 | | | | else classify as A
43 | | | else classify as C
44 else classify as B
```

Fig. 3. The rule for identifying an original source code

In addition, we show the scatter plots of the attributes' values i.e. Thr_SIM, NST, SD_PA and SD_PB. Fig. 4 illustrates that some ranges of these values are overlap. The classes cannot be separated using these values. So, it is difficult to perfectly classify them.

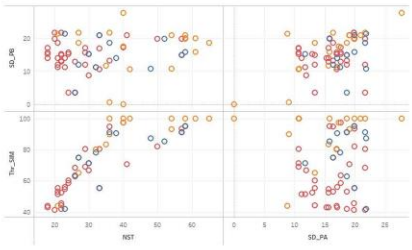


Fig. 4. Scatter plot of the attributes' values.

V. Conclusion

This paper proposes the rule for identifying the original source codes from a set of plagiarized programs. The aim of this research is to find the original copy of the source codes submitted in programming assignments. We have studied the plagiarism level of 140 students' programs collected from the first-year-students' programming-assignments. We select the programs that have a similarity score more than 60% (the similarity score obtained by JPlag). We found that the plagiarism levels are not higher than level 4. We created 78 programs by imitating student' plagiarism-styles which have 10 original programs and 68 plagiarized programs, and extracted 17 attributes from those source codes e.g. similarity score, line of codes. These data are divided into training and testing data. Six classification techniques are used in the experiments i.e. decision tree, REP tree, random forest, neural network, K-nearest neighbor, and Naïve Bayes. The decision tree using J48 and the REP tree yield the best accuracy of 82.61% in the testing data.

References

[1] https://en.oxforddictionaries.com [accessed: Mar, 15,2019]

[2] J. Faidhi, and S.Robinson, "An empirical approach for detecting program similarity and plagiarism within a university programing environment," Computers & Education, vol. 11, 1987, pp.11-19.

[3] C. Arwin, and S.M.M. Tahaghoghi, "Plagiarism Detection across Programming Languages," Australasin Conputer Science Conferenc, Vol. 48, pp.277-286,2006.

[4] U. Bandata, and G. Wijayarathna, "A Machine Learning Based Tool for Source Code Plagiarism Detection," International Journal of Machine Learning and Computing, Vol.1, No. 4, October 2011.

[5] L. Prechelt, G. Malpohl, and M. Philippsen, "Finding plagiarisms among a set of programs with JPlag," J. UCS, vol. 8, no. 11, pp. 1016-1038, 2002.

[6] M. Ďuračík, E. Kršák, and P. Hrkút, "Scalable Source Code Plagiarism Detection Using Source Code Vectors Clustering," in 2018 IEEE 9th International Conference on Software Engineering and Service Science)ICSESS(, 2018, pp. 499-502: IEEE.

[7] N. G. Fonseca, L. Macedo, and A. J. Mendes, "Using early plagiarism detection in programming classes to address the student's difficulties," in 2018 International Symposium on Computers in Education)SIIE(, 2018, pp. 1-6: IEEE.

[8] U. Bandara and G. Wijayarathna, "A machine learning based tool for source code plagiarism detection," International Journal of Machine Learning and Computing, vol. 1, no. 4, p. 337, 2011.

[9] A. Ohno and H. Murao, "A two-step in-class source code plagiarism detection method utilizing improved CM algorithm and SIM," International Journal of Innovative Computing, Information, and Control, vol. 7, no. 8, pp. 4729-4739, 2011.

[10] J. Schneider, A. Bernstein, J. Vom Brocke, K. Damevski, and D. C. Shepherd, "Detecting plagiarism based on the creation process," IEEE Transactions on Learning Technologies, vol. 11, no. 3, pp. 348-361, 2018.

[11] J.-H. Ji, S.-H. Park, G. Woo, and H.-G. Cho, "Generating pylogenetic tree of homogeneous source code in a plagiarism detection system," International Journal of Control, Automation, and Systems, vol. 6, no. 6, pp. 809-817, 2008.

[12] J. Hage, P. Rademaker, and N. van Vugt, "A comparison of plagiarism detection tools," Utrecht University. Utrecht, The Netherlands, vol. 28, p. 1, 2010.

[13] A. Ahadi and L. Mathieson, "A Comparison of Three Popular Source code Similarity Tools for Detecting Student Plagiarism," in Proceedings of the Twenty-First Australasian Computing Education Conference, 2019, pp. 112-117: ACM.

[14] S. Schleimer, D. S. Wilkerson, and A. Aiken, "Winnowing: local algorithms for document fingerprinting," Proceedings of the 2003 ACM SIGMOD international conference on Management of data, 2003, pp. 76-85: ACM.

[15] D. Grune and M. Huntjens, "Het detecteren van kopieën bij informatica-practica," Informatie, vol. 31, no. 11, pp. 864-867, 1989.

CERTIFICATE OF CONTRIBUTIONS

CHAWALIT SAOBAN AND SUNISA RIMCHAROEN

ENTITLED

IDENTIFYING AN ORIGINAL COPY OF THE SOURCE CODES IN PROGRAMMING ASSIGNMENTS

HAS CONTRIBUTED TO

2019 - 16TH INTERNATIONAL JOINT CONFERENCE ON COMPUTER SCIENCE AND SOFTWARE ENGINEERING (JCSSE)

JULY 10 - 12, 2019
AMARI PATTAYA HOTELS, PATTAYA, THAILAND

K. Neammanee.

KRITSANA NEAMMANEE, PH.D.
DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE, FACULTY OF SCIENCE, CHULALONGKORN UNIVERSITY
HONORARY COMMITTEE

C. Lursinsap

CHIDCHANOK LURSINSAP, PH.D.
DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE, FACULTY OF SCIENCE, CHULALONGKORN UNIVERSITY
GENERAL CHAIR

K. Chinnasarn

KRISANA CHINNASARN, PH.D.
FACULTY OF INFORMATICS, BURAPHA UNIVERSITY
TECHNICAL PROGRAM CHAIR

IEEE IEEE THAILAND SECTION Technical Co-Sponsor ECTI Association

Ref.48/2562

JCSSE2019 Your paper 1570542979 ('Identifying an original copy of the source codes in programming assignments') กล่องจดหมาย ×

jcsse=inform...@edas.info อ. 14 พ.ค. 19:10

ถึง ฉัน, Sunisa, Krisana, Watcharaphong, Saichon, Chakchai, Suwanna, Suphakant, Janya, Monnat, Praisan

อังกฤษ > ไทย แปลข้อความ ปิดสำหรับ: อังกฤษ ×

Dear Mr. Chawalit Saoban:

Congratulations – We are happy to inform you that your paper 1570542979 ('Identifying an original copy of the source codes in programming assignments') for JCSSE2019 - 2019 16th International Joint Conference on Computer Science and Software Engineering has been accepted for oral presentation, but requires Major Revisions. At least one author has to make a registration and make an oral presentation during the conference sessions (July 10th, 2019 – 12th, 2019). Registration fee is non-refundable. The registration instructions are available at https://jcsse.informatics.buu.ac.th/2019/registration-policy.php.

For the paper to be qualified for the selection process for inclusion in the IEEEXplore® Digital Library (IEEE Catalog Number: #46885) and indexed by the relevant databases, you are required to do the followings:

1. Take the review comments into account when producing the camera ready version of your final manuscript. According to the IEEE conference policy, the similarity score must be lower than 25% when you submit your camera ready. If your paper fails to comply with this policy, we reserve the right not to include your paper in the selection process for publishing in the IEEEXplore®.

2. Follow the camera-ready formatting instructions at https://jcsse.informatics.buu.ac.th/2019/cameraready-submission.php. Note that the camera-ready submission is starting from May 16, 2019 until May 31, 2019.

3. As the authors of high quality papers, you have an opportunity to choose the following options:

3.1 After a week from the end of conference day, you may receive an invitation letter and choose to submit your work to a special issue of Journal of Internet Technology (JIT), which is indexed by Scopus and ISI (http://jit.ndhu.edu.tw). However, if you decide to do so, your paper will be excluded from IEEEXplore® Digital Library.

3.2 Alternatively, you may choose to extend your work up to 70% and submit to ECTI-CIT which is indexed by Scopus (https://www.tci-thaijo.org/index.php/ecticit). Review process will be conducted by ECTI-CIT. For this option, your paper, as submitted to JCSSE will still be included in IEEEXplore® Digital Library.

On behalf of Faculty of Science, Chulalongkorn University and Faculty of Informatics, Burapha University, We would like to thank you for you contribution to JCSSE and looking forward to seeing you during the conference event at Amari Pattaya Hotels, Chonburi, Thailand.

****

The reviewers comments are below or can be found at https://edas.info/showPaper.php?m=1570542979.

======= JCSSE'19 Review 1 =======

บรรณานุกรม

Bandara, U., & Wijayarathna, G. (2011). A machine learning based tool for source code plagiarism detection. *International Journal of Machine Learning and Computing, 1*(4), 337.

Ji, J.-H., Park, S.-H., Woo, G., & Cho, H.-G. (2008). Generating pylogenetic tree of homogeneous source code in a plagiarism detection system. *International Journal of Control, Automation, and Systems, 6*(6), 809-817.

Ji, J.-H., Woo, G., & Cho, H.-G. (2007). *A source code linearization technique for detecting plagiarized programs.* Paper presented at the ACM SIGCSE Bulletin.

Koc, E. W., Kahn, J., Koncz, A. J., Salvadge, A., & Longenberger, A. (2018). Top Degrees in Demand. *Job Outlook 2019*, 15.

Mann, S., & Frew, Z. (2006). *Similarity and originality in code: plagiarism and normal variation in student assignments.* Paper presented at the Proceedings of the 8th Australasian Conference on Computing Education-Volume 52.

Ohno, A., & Murao, H. (2011). A two-step in-class source code plagiarism detection method utilizing improved CM algorithm and SIM. *International Journal of Innovative Computing, Information, and Control, 7*(8), 4729-4739.

Prechelt, L., Malpohl, G., & Philippsen, M. (2002). Finding plagiarisms among a set of programs with JPlag. *J. UCS, 8*(11), 1016-1038.

Price, C., & Yorke, J. (2015). STUDENT GUIDELINES FOR AVOIDING PLAGIARISM. Academic integrity at Curtin, 4. Retrieved January 16,2019,from https://academicintegrity.curtin.edu.au/local/docs/StudentPlagiarismGuide.pdf

ศรีเปารยะ, ส., & สินสมบูรณ์ทอง, ส. (2560). การเปรียบเทียบประสิทธิภาพวิธีการจำแนกกลุ่มการเป็นโรคไตเรื้อรัง: กรณีศึกษาโรงพยาบาล. *Thai Science and Technology Journal, 25*(5), 839-853.

มหาวิทยาลัยบูรพา
BURAPHA UNIVERSITY

ประวัติย่อของผู้วิจัย

ชื่อ-สกุล	ชวลิต เสาร์แบน
วัน เดือน ปี เกิด	16 January 1988
สถานที่เกิด	สุโขทัย
สถานที่อยู่ปัจจุบัน	110/3 หมู่ 8 ตำบลบ้านสวน อำเภอเมืองชลบุรี จังหวัดชลบุรี 20000
ตำแหน่งและประวัติการทำงาน	รองสารวัตร ฝ่ายอำนวยการ 6 กองบังคับการอำนวยการ ตำรวจภูธรภาค 2
ประวัติการศึกษา	วศ.บ. สาขาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร
รางวัลหรือทุนการศึกษา	ตำรวจภูธรภาค 2