การบีบอัดข้อความภาษาไทยโดยใช้ PREDICTION BY PARTIAL MATCHING

สุรีวรรณ แจ้งจิตร

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
วิทยาศาสตรมหาบัณฑิต (วิทยาการคอมพิวเตอร์)
คณะสถิติประยุกต์
สถาบันบัณฑิตพัฒนบริหารศาสตร์
พ.ศ. 2547

การบีบอัดข้อความภาษาไทยโดยใช้ PREDICTION BY PARTIAL MATCHING

สุรีวรรณ แจ้งจิตร

คณะสถิติประยุกต์

คณะกรรมการสอบวิทยานิพนธ์ ได้พิจารณาแล้วเห็นสมควรอนุมัติให้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรมหาบัณฑิต (วิทยาการคอมพิวเตอร์)

อาจารย์ .....ประธานกรรมการ
(ดร. โอม ศรนิล)

ผู้ช่วยศาสตราจารย์.....กรรมการ
(ดร. สุรพงศ์ เอื้อวัฒนามงคล)

อาจารย์.....กรรมการ
(ดร. สุกัญญา สุรเนาวรัตน์)

อาจารย์..... คณบดี/ผู้อำนวยการ
(ดร. เลอสรรค์ โบสุวรรณ)
วันที่..19..เดือน..พฤษภาคม..พ.ศ. 2547

บทคัดย่อ

ชื่อวิทยานิพนธ์ : การบีบอัดข้อความภาษาไทยโดยใช้ Prediction by Partial Matching
ชื่อผู้เขียน : น.ส. สุรีวรรณ แจ้งจิตร
ชื่อปริญญา : วิทยาศาสตรมหาบัณฑิต (วิทยาการคอมพิวเตอร์)
ปีการศึกษา : 2546

ความก้าวหน้าทางเทคโนโลยีคอมพิวเตอร์ทำให้เอกสารอิเล็กทรอนิกมีปริมาณเพิ่มขึ้นอย่างรวดเร็ว เอกสารเหล่านี้ส่วนใหญ่เป็นภาษาอังกฤษ อย่างไรก็ตามเอกสารในภาษาอื่นก็มีเพิ่มมากขึ้นเป็นลำดับ ดังนั้นการบีบอัดข้อมูลเป็นวิธีหนึ่งในการเพิ่มประสิทธิภาพในการจัดเก็บข้อมูล

งานวิจัยนี้เสนอวิธีการบีบอัดข้อมูลที่มีประสิทธิภาพสูงสำหรับภาษาไทย โดยปรับปรุงวิธีการ Prediction by Partial Matching ซึ่งเป็นวิธีหนึ่งที่ให้ประสิทธิภาพการบีบอัดข้อมูลสูงโดยพิจารณาคุณลักษณะของภาษา

จากการทดลองเพื่อวัดประสิทธิภาพการบีบอัดข้อมูลของวิธีการที่นำเสนอกับข้อมูลลักษณะต่าง ๆ พบว่าวิธีการที่นำเสนอให้ประสิทธิภาพการบีบอัดข้อมูลได้ถึง 84 เปอร์เซ็นต์ ขึ้นอยู่กับลักษณะของข้อมูลทดสอบ และรูปแบบการประมวลเบื้องต้น ซึ่งให้ผลที่ดีกว่ารหัสฮัฟแมนและสารสนเทศเฉลี่ย แต่อย่างไรก็ดีวิธีการในกลุ่ม Prediction by Partial Matching นี้เป็นที่ยอมรับกันว่าใช้เวลาในการเข้ารหัส-ถอดรหัสมากวิธีหนึ่ง ดังนั้นวิธีการที่นำเสนอเหมาะสมสำหรับการนำไปใช้บีบอัดข้อมูล ที่เน้นประสิทธิภาพการบีบอัดข้อมูล

ABSTRACT

Title of Thesis	:	**Thai Text Compression Using Prediction by Partial Matching**
Author	:	**Ms. Sureewan Jangjit**
Degree	:	**Master of Science (Computer Science)**
Year	:	**2003**

Technological advances motivates the creation of a large amount of electronic documents. Most documents available world wide are written in English. Nevertheless, the number of documents in other languages is increasing quickly as well. Compression methodologies working well with English documents may not be appropriate for those in other languages. This research proposes as efficient data compression technique for Thai language which modifies the Prediction by Partial Matching method by taking into account the language characteristics. The experimental experiments show advantages over variations of Prediction by Partial Matching proposed previously, Huffman coding, and Entropy.

กิตติกรรมประกาศ

วิทยานิพนธ์เรื่องการบีบอัดข้อความภาษาไทย โดยใช้ Prediction by Partial Matching เอกสารสำเร็จได้ เนื่องมาจากบุคคลหลายท่านได้ให้ความกรุณาช่วยเหลือให้ข้อเสนอแนะ คำปรึกษา ความคิดเห็น และกำลังใจ

ผู้เขียนขอกราบขอบพระคุณ อ.ดร.โอม ศรนิล ที่ได้ให้คำแนะนำชี้แนะในทุกด้านและตรวจสอบวิทยานิพนธ์ทุกขั้นตอน และขอกราบขอบพระคุณ ผศ.ดร.สุรพงศ์ เอื้อวัฒนามงคล และ ดร.สุกัญญา สุรเนาวรัตน์ ที่ได้แนะนำแนวคิดในการบีบอัดข้อความภาษาไทยให้สำเร็จได้ตามวัตถุประสงค์

ขอขอบพระคุณอาจารย์ทุกท่านในคณะสถิติประยุกต์ที่ได้ถ่ายทอดความรู้ให้แก่ผู้เขียน

ขอกราบขอบพระคุณ คุณพ่อ คุณแม่ และพี่ชายที่ได้ให้ความรัก ความห่วงใย และส่งเสริมสนับสนุนในด้านการศึกษาตลอดมา

ท้ายสุดนี้ ขอขอบคุณเพื่อนนักศึกษาสาขาวิชาวิทยาการคอมพิวเตอร์ที่ให้ความช่วยเหลือ และกำลังใจตลอดการศึกษาในสถาบันบัณฑิตพัฒนบริหารศาสตร์

สุรีวรรณ แจ้งจิตร

กุมภาพันธ์ 2547

สารบัญ

สารบัญตาราง

สารบัญภาพ

สารบัญกราฟ

บทที่ 1

บทนำ

1.1 ความเป็นมาของปัญหา

การดำเนินชีวิตประจำวันของมนุษย์ในปัจจุบัน มีส่วนเกี่ยวข้องกับคอมพิวเตอร์ และเทคโนโลยีเสมอ ทำให้แนวโน้มของเทคโนโลยีสารสนเทศ และการใช้เทคโนโลยีสารสนเทศในด้านต่าง ๆ เพิ่มขึ้น เช่น ด้านการศึกษา ด้านการติดต่อสื่อสาร ด้านการทำงาน เป็นต้น ทำให้มีผลต่อความต้องการพื้นที่ในการจัดเก็บสารสนเทศ ค่าใช้จ่าย และเวลาในการติดต่อสื่อสารเพิ่มมากขึ้น ดังนั้นการแก้ปัญหาวิธีหนึ่ง คือ การลดขนาดข้อมูลที่จัดเก็บ ช่วยให้การติดต่อสื่อสารที่เร็วขึ้น หรือ สามารถส่งข้อมูลข้อมูลได้มากขึ้นในระยะเวลาเท่าเดิม และการลดขนาดข้อมูลที่จัดเก็บด้วยวิธีการบีบอัดข้อมูล ยังมีข้อดีในด้านการรักษาความปลอดภัยของข้อมูลระดับหนึ่งด้วย เพราะการบีบอัดข้อมูลต้องมีการเปลี่ยนรูปแบบของรหัสก่อนที่จะส่งผ่านเครือข่าย จากประโยชน์ของการบีบอัดข้อมูลทำให้ปัจจุบันมีการศึกษา-พัฒนาวิธีการบีบอัดข้อมูลเดิม และมีการออกแบบวิธีการบีบอัดข้อมูลใหม่ เพื่อให้เหมาะสมกับลักษณะงานในด้านต่าง ๆ

ดังนั้นในงานวิทยานิพนธ์นี้ มุ่งเน้นประสิทธิภาพในการบีบอัดข้อมูล จึงเลือกใช้วิธี Prediction by Partial Matching (PPM) และพัฒนาต่อเพื่อให้เหมาะกับลักษณะภาษาไทย

การลดขนาดข้อมูลมี 2 เทคนิค แสดงได้ดังภาพที่ 1.1 ได้แก่ เทคนิคแบบไม่ย้อนกลับ (Nonreversible techniques) หรือการอัดแน่นข้อมูล (Data Compaction) คือข้อมูลที่ถูกลดขนาด โดยตัดสิ่งที่ไม่จำเป็นทิ้ง เช่น ช่องว่าง, คำซ้ำซึ่งตัดให้เหลือเพียงหนึ่งตัว ดังนั้นข้อมูลที่ถูกลดขนาดลงจะอยู่ในรูปแบบใหม่ตลอด โดยไม่สามารถย้อนกลับคืนข้อมูลเดิมได้ และเทคนิคแบบย้อนกลับ (Reversible techniques) คือการเก็บข้อมูลเดิมไว้ทั้งหมด แต่เป็นการเก็บโดยลดขนาดลงด้วยการเข้ารหัส (Encoding) เรียกขั้นตอนนี้ว่า การบีบอัดข้อมูล (Data compression) และถ้าต้องการข้อมูลเดิมกลับมาต้องทำการถอดรหัส (Decoding) เรียกขั้นตอนนี้ว่า การขยายข้อมูล (Data decompression) ซึ่งการบีบอัดข้อมูล แบ่งได้ 2 เทคนิค คือ

1) การบีบอัดข้อมูลที่มีความสูญหายเล็กน้อย (Lossless) คือการบีบอัดข้อความ (Text compression) โดยที่ข้อความที่ถูกบีบอัดเมื่อขยายการบีบอัดจะกลับคืนสภาพเดิม โดยข้อมูลไม่ผิดพลาด หรือผิดพลาดเล็กน้อย แสดงได้ดังภาพที่ 1.2 ซึ่งวิธีการสร้างตัวแบบมี 2 วิธีดังนี้

(1.1) ตัวแบบทางสถิติ (Statistical model) หรือตัวแบบทางตัวอักขระ (Symbolwise model) ทำงานโดยใช้ค่าความน่าจะเป็นของตัวอักขระที่เกิดในข้อความ โดยตัว

อักขระมีความถี่สูงจะส่งผลให้กระแสข้อมูลบิต (Bit stream) มีขนาดลดลง เช่น รหัสฮัฟแมน (Huffman code), การเข้ารหัสเชิงคำนวณ (Arithmetic coding) เป็นต้น ซึ่งมีกระบวนการบีบอัดข้อมูล 2 ขั้นตอนดังนี้

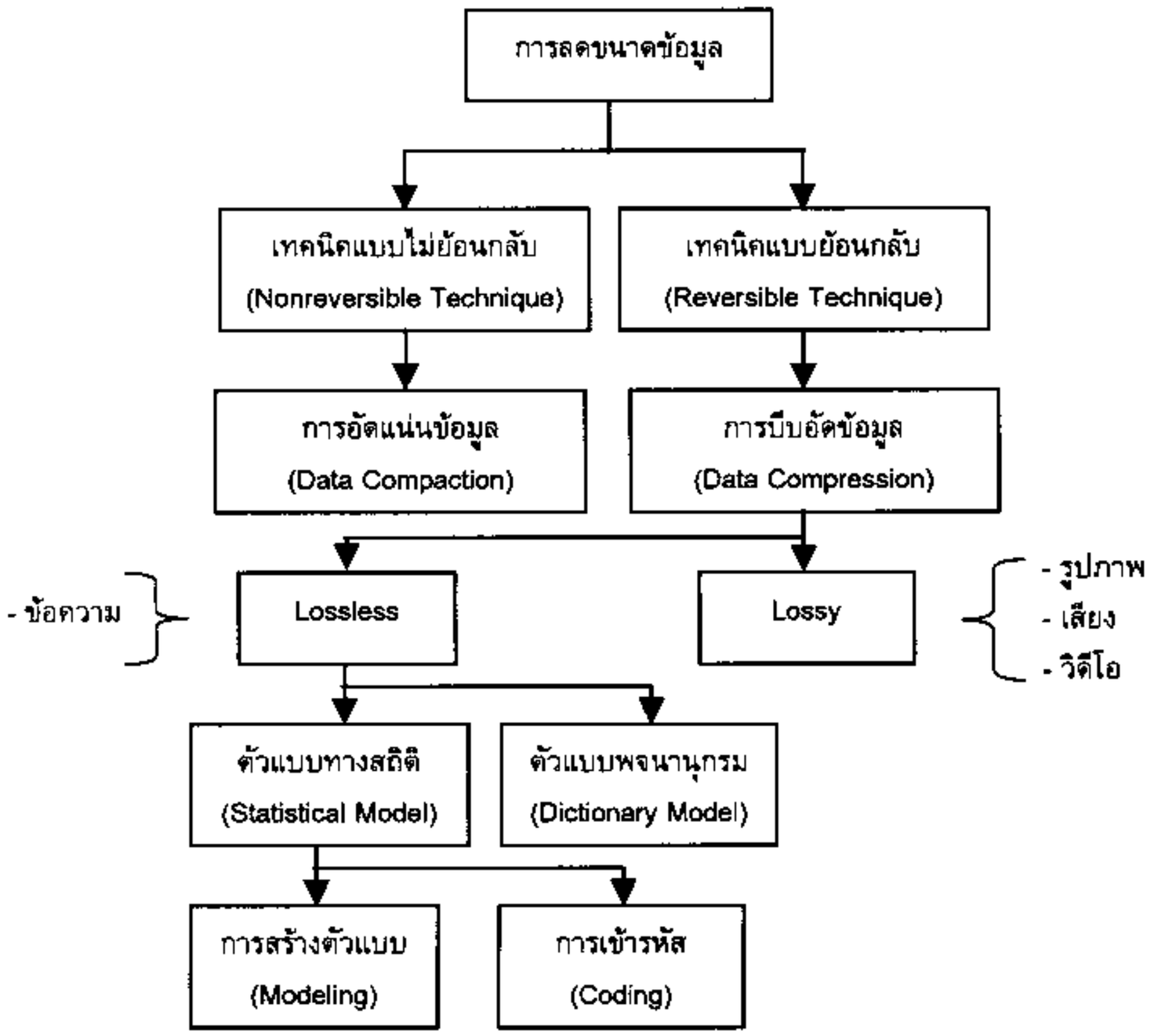
ขั้นตอนที่ 1 การสร้างตัวแบบ (Modeling) คือชุดข้อมูลสถิติ เพื่อนำไปใช้ประโยชน์ในการเข้ารหัส (Coding) เช่น โปรแกรมการบีบอัดก่อนจะทำการบีบอัดข้อมูลต้องนำข้อมูลที่ต้องการบีบอัดผ่านแบบจำลอง (Model) เพื่อหาความถี่ของตัวอักขระที่เกิดขึ้นในข้อความก่อนที่จะนำค่าที่ได้จากแบบจำลองมาใช้ในการเข้ารหัส ได้ผลลัพธ์เป็นกระแสข้อมูลบิตเพื่อส่งไปฝั่งผู้ถอดรหัสซึ่งกระแสข้อมูลบิตที่ส่งไปต้องสามารถนำไปถอดรหัส โดยใช้ข้อมูลจากแบบจำลองย้อนกลับคืนอยู่ในรูปข้อความเดิมได้ โดยแบบจำลองที่ดีจะให้อัตราการบีบอัดข้อมูลสูง

ขั้นตอนที่ 2 การเข้ารหัส (Coding) คือการนำค่าชุดข้อมูลสถิติจากแบบจำลองมาผ่านกระบวนการเข้ารหัสเพื่อการบีบอัด-ขยายข้อมูลข้อมูล

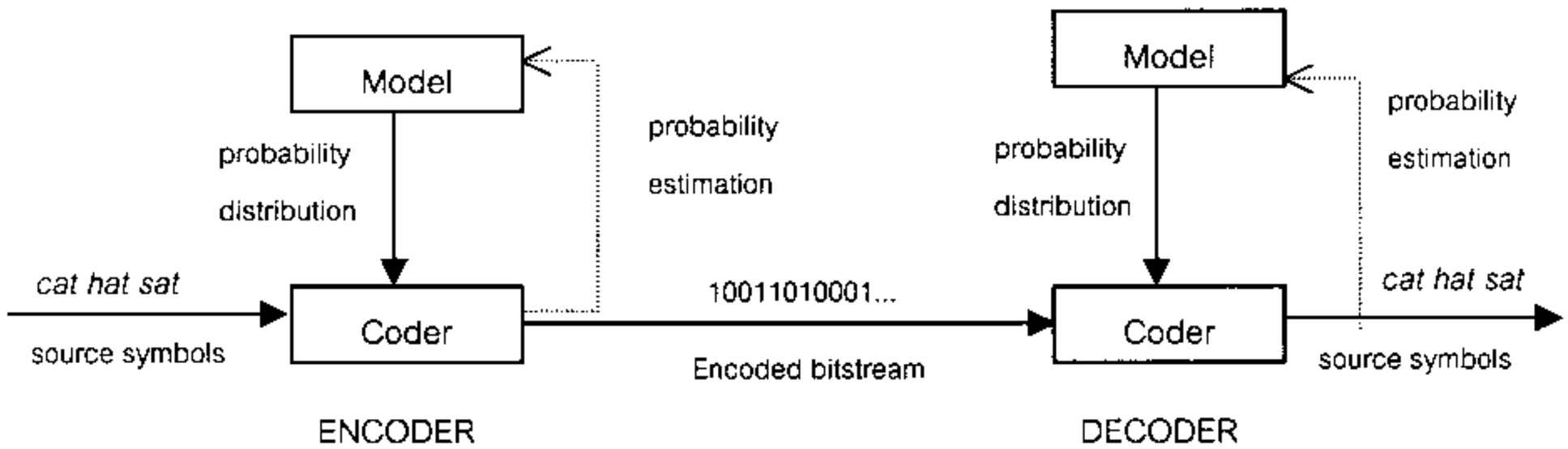
(1.2) ตัวแบบพจนานุกรม (Dictionary model) คือการนำคำที่เกิดซ้ำ ๆ เก็บเข้ารหัสไว้เหมือนพจนานุกรม โดยคาดหวังว่าคำที่เก็บในพจนานุกรมมักเป็นคำที่พบมากในข้อความ ดังนั้นการบีบอัดข้อมูล ถ้าพบคำที่ตรงกับพจนานุกรมมากส่งผลให้กระแสข้อมูลบิตมีขนาดลดลง เช่น อัลกอริทึมแอลแซดดับเบิลยู (LZW)

2) การบีบอัดข้อมูลที่มีความสูญเสียได้ (Lossy) เช่น การบีบอัดรูปภาพหรือ เสียง เป็นต้น ที่ยอมให้ผิดรูปจากต้นฉบับได้

ในวิทยานิพนธ์นี้ มุ่งเน้นในการนำเทคนิคการลดขนาดข้อมูลแบบย้อนกลับ โดยใช้วิธีการบีบอัดข้อมูลมีความสูญหายได้เล็กน้อย และใช้ตัวแบบทางสถิติ คือวิธี Prediction by Partial Matching ช่วยในการเข้ารหัส-ถอดรหัสข้อความภาษาไทย



ภาพที่ 1.1 แสดงเทคนิคการลดขนาดข้อมูล



ภาพที่ 1.2 แสดงกระบวนการบีบอัดข้อมูลของตัวแบบทางสถิติ

1.2 ขอบเขตของงานวิจัย

1.2.1 เสนอแนวคิดที่ปรับปรุงวิธี PPM จากการศึกษาคุณลักษณะของภาษาไทย พร้อมทั้งเสนอกลยุทธ์ในการประมวลผลข้อมูลภาษาไทยเบื้องต้น

1.2.2 ทำการทดลองเพื่อประเมินผลแนวคิดต่าง ๆ ที่นำเสนอ และเปรียบเทียบประสิทธิภาพในการบีบอัดข้อมูลภาษาไทยกับเทคนิคอื่น ๆ

1.2.3 ในการทำวิทยานิพนธ์นี้พัฒนาบนเครื่องไมโครคอมพิวเตอร์ ซึ่งมีรายละเอียดดังนี้

หน่วยประมวลผลกลาง:	Intel Celeron 433 Hz
ระบบปฏิบัติการ:	Microsoft Windows 98
หน่วยความจำ:	128 MB

1.3 วิธีดำเนินการวิจัย

ขั้นตอนการวิจัยแบ่งออกเป็น 5 ขั้นตอนดังนี้

1) ทำการศึกษาเอกสาร และงานวิจัยที่เกี่ยวข้อง ได้แก่
   (1) ศึกษางานวิจัยต่าง ๆ ที่เกี่ยวข้องกับการบีบอัดข้อมูลวิธีต่าง ๆ
   (2) ศึกษาการคุณลักษณะของภาษาไทย เพื่อหาแนวคิดในการประยุกต์ PPM
2) ทำการศึกษา และวิเคราะห์เพื่อหาแนวคิดต่าง ๆ ที่นำมาใช้ในการออกแบบวิธีการเข้ารหัส-ถอดรหัสข้อมูลด้วยวิธี PPM
3) ทำการออกแบบวิธีการเข้ารหัส-ถอดรหัสข้อความ พร้อมทั้งเสนอกลยุทธ์ในการประมวลผลข้อมูลภาษาไทยเบื้องต้น เพื่อให้ได้ประสิทธิภาพในการบีบอัดที่ดีขึ้น
4) ทำการทดลองเพื่อวัด และเปรียบเทียบประสิทธิภาพของวิธีการที่เสนอ
5) ทำการสรุปผลการวิจัย

บทที่ 2

เอกสารและงานวิจัยที่เกี่ยวข้อง

ในวิทยานิพนธ์นี้มีเอกสาร และงานวิจัยที่เกี่ยวข้องกับ สารสนเทศเฉลี่ย (Entropy rate) รหัสฮัฟแมน (Huffman code) การเข้ารหัสเชิงคำนวณ (Arithmetic coding) Prediction by Partial Matching (PPM) และงานวิจัยที่เกี่ยวข้องมีรายละเอียดดังนี้

2.1 สารสนเทศเฉลี่ย (Entropy rate)

สารสนเทศเฉลี่ย เรียกว่า Entropy (H) คือปริมาณที่วัดค่าเฉลี่ยของค่า binary symbols ที่ใช้ในการเข้ารหัส (Sayood, K., 2000: 15) ซึ่งเป็นค่าที่แสดงจำนวนหน่วยเฉลี่ยต่ำสุดของรหัสที่ใช้ในการเข้ารหัส (เรวดี ลิมปิโชติกุล, 2534: 13) โดยใช้ข้อมูลสถิติของข้อมูลต้นฉบับ เขียนสมการได้ $H(X) = -\sum_{i=0}^{N-1} p_i \log_2 p_i$ เมื่อ p_i คือค่าความน่าจะเป็นของตัวอักขระ โดยค่าเฉลี่ยจำนวนบิต/ตัวอักขระเท่ากับ $\lceil H(X) \rceil$ เช่น กำหนดให้มีชุดของตัวอักขระ {A,B,C,D,E} มีค่าความน่าจะเป็น 0.5, 0.2, 0.1, 0.1, 0.1 ตามลำดับ ค่าเฉลี่ยจำนวนบิต/ตัวอักขระที่ใช้ในการเข้ารหัส คือ

$$H(X) = \lceil -(0.5\log_2 0.5 + 0.2\log_2 0.2 + (0.1\log_2 0.1)*3) \rceil$$

$$H(X) = \lceil -(-0.5 + (-0.46438) + (-0.9965)) \rceil$$

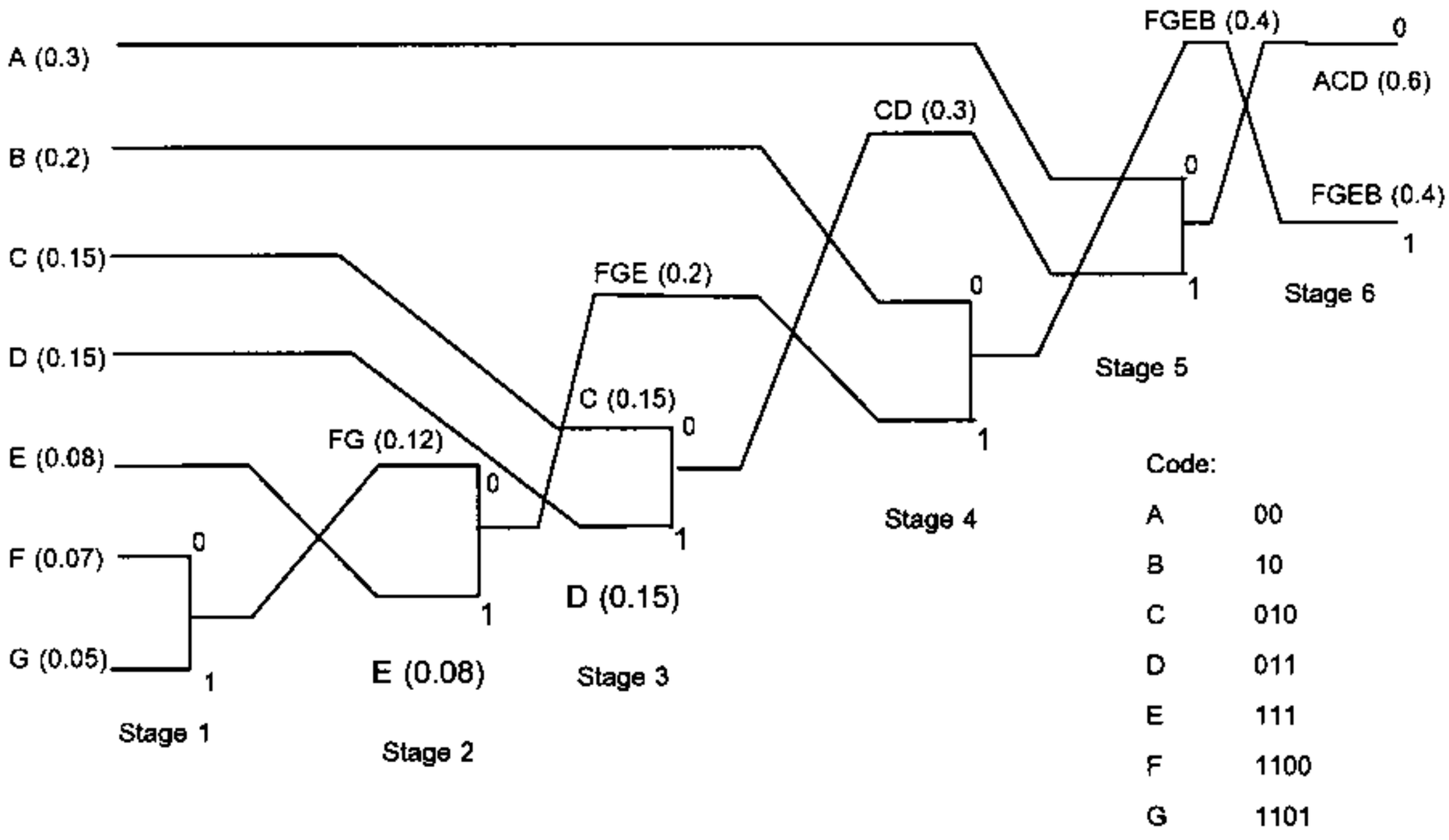
$$H(X) = \lceil -(-1.9) \rceil = 2$$

ผลลัพธ์คือ ค่าเฉลี่ย 2 บิต/ตัวอักขระ และสมมติถ้าข้อความยาว 10 ตัวอักขระ ได้แก่ AAAAABBCDE จะได้กระแสข้อมูลบิดยาว 20 บิต

2.2 รหัสฮัฟแมน (Huffman code)

การสร้างรหัสใหม่จากความน่าจะเป็นของรหัสต่าง ๆ กำหนดให้ $\{A_1, A_2, ..., A_n\}$ มีความน่าจะเป็น $\{P_1, P_2, ..., P_n\}$ โดยที่ $P_1 \leq P_2 \leq .... \leq P_n$ และนำค่าความน่าจะเป็นสร้างต้นไม้แบบทวิภาค (Binary tree) โดยสร้างต้นไม้เริ่มจากเลือกต้นไม้ 2 ต้นที่มีค่าความน่าจะเป็นน้อยสุด คือ P_1 และ P_2 มาผสาน (Merge) เป็นต้นไม้ 1 ต้น และให้ค่าความน่าจะเป็น (P_1+P_2) และตัด P_1, P_2 ออกจากกลุ่มค่าความน่าจะเป็น และแทรกค่า (P_1+P_2) เข้าไปในกลุ่มแทน ทำซ้ำ

จนกระทั่งได้ต้นไม้แสดงดังภาพที่ 2.1 ซึ่งวิธีนี้จะได้รหัสนำหน้าที่มีคุณสมบัติเป็นรหัสนำหน้า (Prefix code) ต่ำสุดที่ช่วยการเข้ารหัส-ถอดรหัสได้รหัสของตัวอักขระที่เป็นหนึ่งเดียว (Unique)



ภาพที่ 2.1 แสดงต้นไม้แบบทวิภาคที่สร้างจากรหัสฮัฟแมน

จากภาพที่ 2.1 แสดงการทำงานของรหัสฮัฟแมน พบว่า ตัวอักขระ "A" ให้กระแสข้อมูลบิดสั้นที่สุด เพราะมีค่าความน่าจะเป็นสูง แต่อักขระ "G" ให้กระแสข้อมูลบิดยาวที่สุด เพราะมีค่าความน่าจะเป็นต่ำ และเมื่อพิจารณาความน่าจะเป็นกับความยาวของรหัสมีความสัมพันธ์ ดังนี้ ความน่าจะเป็นของรหัสเรียงจากน้อยไปมาก คือ $P_1 \leq P_2 \leq P_3 \leq P_4 \leq ... \leq P_{n-1} \leq P_n$ เมื่อสร้างรหัสใหม่ตามรหัสฮัฟแมนได้ความยาวรหัสใหม่ดังนี้ $L(A_1) \geq L(A_2) \geq L(A_3) \geq L(A_4) \geq L(A_{n-1}) \geq L(A_n)$ คือรหัสที่มีความน่าจะเป็นสูงจะได้ความยาวของรหัสใหม่ที่สั้น และรหัสที่มีความน่าจะเป็นต่ำจะได้ความยาวของรหัสใหม่ที่ยาว

2.3 การเข้ารหัสเชิงคำนวณ (Arithmetic coding)

วิธีเข้ารหัสข้อมูลโดยการคำนวณ (Sayood, K., 2000: 75-104) กำหนดช่วงของเลขจำนวนจริง (Real number) ระหว่าง 0 ถึง 1 เขียนได้ [0, 1) คือ $0 \leq x < 1$ และมีวิธีการขยายช่วง เพื่อไม่ให้ค่าขอบล่าง (Lower) และขอบบน (Upper) เบนเข้าหากันจนบรรจบกัน และมีวิธีการขยายช่วง 3 กรณี เมื่อค่าขอบล่าง และขอบบนไปอยู่ในช่วงดังนี้ กรณีที่ 1 อยู่ในช่วงขอบล่าง เรียกเงื่อนไข E_1 คือ [0, 0.5) -> [0, 1); $E_1(x) = 2x$ จะส่งบิต "0" ออกไป กรณีที่ 2 อยู่

ในขอบบน เรียกเงื่อนไข E_2 คือ [0.5,1) -> [0, 1); $E_2(x) = 2(x– 0.5)$ จะส่งบิต "1" ออกไป กรณีที่ 3 จุดกึ่งกลาง (Mid point) เรียกเงื่อนไข E_3 คือ [0.25, 0.75) -> [0, 1); $E_3(x) = 2(x – 0.25)$ จะส่งกระแสข้อมูลบิตที่ตรงข้ามจากการเกิดเงื่อนไข E_1 หรือ E_2 เช่น ถ้าเงื่อนไข E_3 เกิด 3 ครั้งในขณะเข้ารหัส และเกิดเงื่อนไข E_2 จะมีการส่งบิต "1" ตามด้วยบิต "0" อีก 3 ตัว หรือถ้าเกิดเงื่อนไข E_1 หลังจากเงื่อนไข E_3 จะส่งบิต "0" ตามด้วย "1" อีก 3 ตัว

2.3.1 อัลกอริทึมสำหรับเข้ารหัสและถอดรหัส

กำหนดให้ $L^{(n)} = L^{(n-1)} + (U^{(n-1)} - L^{(n-1)}) Fx(x_n-1)$

$U^{(n)} = L^{(n-1)} + (U^{(n-1)} - L^{(n-1)}) Fx(x_n)$

เมื่อ x_n = ค่าของตัวแปรแบบสุ่มที่สอดคล้อง กับตำแหน่งที่ n ของตัวอักขระ

$L^{(n)}$ = ขีดจำกัดล่าง (Lower limit) ของช่วงระหว่างตำแหน่ง n ที่ทำงานซ้ำ

$U^{(n)}$ = ขีดจำกัดบน (Upper limit) ของช่วงระหว่างตำแหน่ง n ที่ทำงานซ้ำ

A = ชุดข้อมูลของตัวอักขระ

$P(x = i) = P(a_i)$ เมื่อ $x(a_i) = i$, $a_i \in A$

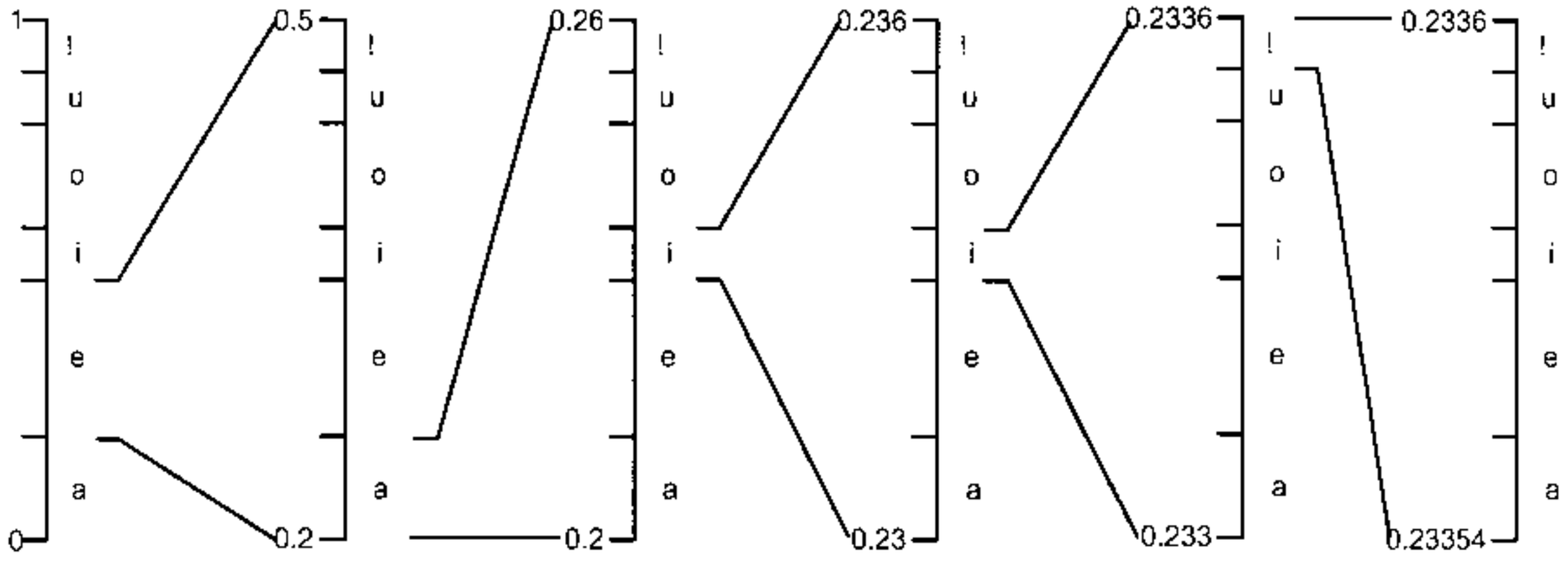
$$\text{ฟังก์ชันความน่าจะเป็นสะสม} = Fx(i) = \sum_{k=1}^{i} P(x = k)$$

เช่น $A = \{a_1, a_2, a_3\}$ และ $P(a_1) = 0.8$, $P(a_2) = 0.02$, $P(a_3) = 0.18$ ได้ $Fx(1) = 0.8$, $Fx(2) = 0.82$ และ $Fx(3) = 1$

วิธีนี้เริ่มต้นด้วยกำหนดค่า $L^{(0)} = 0$ และ $U^{(0)} = 1$ การบีบอัดข้อมูลจะใช้ค่าความน่าจะเป็นของข้อมูลตัวถัดไปเป็นค่าที่สร้างช่วงค่าใหม่ เพราะทุกครั้งที่มีการเข้ารหัสตัวอักขระแต่ละตัวจะมีการปรับช่วงขีดจำกัดล่าง และขีดจำกัดบนใหม่ให้แคบลง ดังนั้นเมื่อข้อความมีความยาวเพิ่มขึ้นช่วงค่าจะยิ่งแคบลง และจำนวนบิตในการแทนค่าช่วงจะเพิ่มขึ้น

ตัวอย่าง กำหนดให้มีตัวอักขระ คือ {a, e, i, o, u, !} และต้องการเข้ารหัสข้อความ "eaii!" โดยมีรายละเอียดของข้อมูลดังนี้

ตัวอักขระ	ความถี่	ค่าความน่าจะเป็น	ค่าความน่าจะเป็นสะสม	ช่วงข้อมูล
a	2	0.2	0.2	[0, 0.2)
e	3	0.3	0.5	[0.2 , 0.5)
i	1	0.1	0.6	[0.5 , 0.6)
o	2	0.2	0.8	[0.6 , 0.8)
u	1	0.1	0.9	[0.8 , 0.9)
!	1	0.1	1.0	[0.9 , 1.0)



ภาพที่ 2.2 แสดงช่วงข้อมูลที่ได้หลังเข้ารหัสข้อความ "eaii!"

ภาพที่ 2.2 แสดงช่วงข้อมูลหลังการเข้ารหัสข้อความ "eaii!" โดยมีขั้นตอนการเข้ารหัสดังนี้ กำหนดค่าเริ่มต้น L = 0, U = 1 เมื่อเข้ารหัสตัวอักขระ "e" ปรับช่วงข้อมูลได้ [0.2,0.5) และปรับค่า L = 0.2, U = 0.5 ทำซ้ำในการเข้ารหัสตัวอักขระตัวถัดไปพร้อมกับปรับช่วงข้อมูลใหม่ จนกระทั่งเข้ารหัสตัวอักขระครบจากภาพนี้จะได้ค่า L = 0.23354, U = 0.2336

การถอดรหัสต้องทราบข้อมูลดังนี้ ความน่าจะเป็นของตัวอักขระแต่ละตัว และค่า L, U สุดท้ายหลังการเข้ารหัส คือ [0.23354, 0.2336) โดยเริ่มต้นกำหนดค่า L = 0, U = 1 และแบ่งช่วงตามความน่าจะเป็นของตัวอักขระแล้วนำค่า L, U สุดท้าย ซึ่งจะได้ช่วงระหว่าง [0.2,0.5) คือ "e" หลังจากนั้นให้ลดช่วงค่าลง ทำซ้ำการแบ่งช่วงจาก L= 0.2, U = 0.5 ตามความน่าจะเป็นของตัวอักขระ และเปรียบช่วงที่ได้กับค่า L, U สุดท้ายอีกครั้งจนกระทั่งได้ข้อมูลที่เข้ารหัสครบทุกตัว

2.4 Prediction by Partial Matching (PPM)

PPM (Lynch, T.J.: 1985: 13-14) คือการบีบอัดข้อมูลที่มีพื้นฐานจากวิธีการเข้ารหัสเชิงคำนวณ โดยคำนวณความน่าจะเป็นของชุดของตัวอักขระก่อนหน้า (Context) ตามด้วยตัวอักขระปัจจุบันที่ไม่ซ้ำกันในแต่ละอันดับ (Order) เช่น PPM อันดับที่ 3 จะมีชุดตัวอักขระก่อนหน้า 3 ตัวในการทำนายตัวอักขระปัจจุบันว่าจะเป็นตัวอะไร โดยอาศัยคลังข้อมูลฝึกสอน (Training data) ซึ่งเป็นชุดสถิติใช้อ้างอิงในการเข้ารหัส-ถอดรหัส และอาจเกิดปัญหา Zero-frequency คือพบชุดตัวอักขระก่อนหน้าขนาดเท่ากับอันดับที่กำหนดตามด้วยตัวอักขระปัจจุบันที่ไม่เคยเกิดในอันดับนี้ หมายถึงตัวอักขระปัจจุบันมีค่าเท่ากับศูนย์ ดังนั้นต้องใช้ค่าความน่าจะเป็นของ Escape (ESC) ในการคำนวณในอัลกอริทึมการเข้ารหัส-ถอดรหัส ซึ่งมีหลายวิธี

ในการจัดการปัญหานี้ ได้แก่ วิธี A, วิธี B, วิธี C, วิธี D และเรียกวิธีที่ใช้แก้ปัญหาว่า วิธี PPMA, วิธี PPMB, วิธี PPMC และวิธี PPMD ตามลำดับ

2.4.1 แนวคิดในการพัฒนาวิธี PPMA, วิธี PPMB, วิธี PPMC และวิธี PPMD

2.4.1.1 แนวคิดในการพัฒนาวิธี PPMA

กำหนดให้ตัวอักขระที่นำมาทดสอบ คือ 0 และ 1 (Binary alphabet) โดยให้ C_0 คือ จำนวนครั้งที่เกิด 0 และ C_1 คือจำนวนครั้งที่เกิด 1 และความน่าจะเป็นของตัวอักขระ i คือ $P(i) = C_i/(C_0+C_1)$ โดยค่าของ $P(i) = 0$ ถ้า C_0 หรือ $C_1 = 0$ แต่การประมาณค่าความน่าจะเป็นต้องไม่เป็นศูนย์ (Non-zero) ดังนั้นแก้ปัญหาโดยกฎลาปลาซของการต่อเนื่อง (Laplace's law of succession) คือถ้าหากรู้สภาพตั้งต้นที่ดีพอ เราสามารถทำนายอนาคตของเอกภพทั้งเอกภพได้ โดยกฎลาปลาซของการต่อเนื่องทำให้เป็นลักษณะที่ไม่เป็นตัวอักขระแบบไบนารี (Non-binary alphabets) โดยเพิ่มค่าคงที่ เพื่อการเปลี่ยนแปลง สามารถคำนวณความน่าจะเป็นได้ดังนี้ $P(i) = (C_i+r)/(n+r)$ เมื่อ $n = C_0+C_1$ คือจำนวนตัวอักขระที่ไม่ซ้ำกันทั้งหมดในข้อความ, $i = 0$ หรือ 1 และ r เป็นค่าคงที่ที่มากกว่าศูนย์ โดยให้ $r = 1$ เป็นค่าคงที่ที่ดีที่สุด โดยมีข้อสมมติฐานว่าถ้าชุดของตัวอักขระก่อนหน้า P(i) มีการแจกแจงแบบมีรูปแบบแน่นอน (Uniform distribution) คือเหตุการณ์การเกิดตัวอักขระ 0 และ 1 มีโอกาสเกิดได้เท่า ๆ กัน และผลลัพธ์จากการทดลอง (Cleary, J.G. and Teahan, W.J.: 1995) ไม่เป็นตามข้อสมมติฐานข้างต้น แต่ก็เลือกวิธีนี้เป็นวิธีหนึ่งที่ใช้แก้ปัญหา Zero-frequency และนับตัวอักขระที่ไม่ซ้ำกันให้ ESC เป็น 1 เสมอ และนับค่าความถี่ให้ตัวอักขระตามที่เกิดจริงในข้อความ เพื่อใช้ความน่าจะเป็นตามที่เกิดจริงในข้อความ

การนับตัวอักขระที่ไม่ซ้ำกันให้ ESC เท่ากับ 1 เสมอให้ข้อดี คือการนับรวดเร็วและให้ค่าความน่าจะเป็นของ P(i) สูงขึ้น ดามดารางที่ 2.4 เปรียบเทียบวิธี PPMA มีการนับจำนวนตัวอักขระที่ไม่ซ้ำกันให้ P(ESC) เท่ากับ 1 และวิธี PPMC มีการนับจำนวนตัวอักขระที่ไม่ซ้ำกันให้ P(ESC) ตามที่เกิดจริงในข้อความ พบว่า ค่า P(i) ของวิธี PPMA สูงกว่า P(i) ของวิธี PPMC

2.4.1.2 แนวคิดในการพัฒนาวิธี PPMB

จากวิธี PPMA ได้พัฒนาวิธี PPMB ขึ้นเพื่อต้องการให้มีประสิทธิภาพการบีบอัดข้อมูลที่สูงขึ้น โดยมีแนวคิดจากการที่พบสัญลักษณ์ครั้งแรกอาจเกิดข้อผิดพลาดได้ (Error) เช่น ความผิดพลาดจากมนุษย์ในการพิมพ์ข้อมูลผิด (Human error) แต่เหตุการณ์ที่เกิดตั้งแต่ครั้งที่ 2 เป็นต้นไปมีโอกาสนับความถี่ที่เกิด ดังนั้นการนับค่าความถี่ให้ตัวอักขระจะเริ่มนับ

ตัวอักขระที่เจอครั้งที่ 2 เป็นค่าเริ่มต้น (ตัวอักขระที่เจอครั้งแรกไม่นับ) และนับจำนวนตัวอักขระที่ไม่ซ้ำกันให้ ESC ตามที่เกิดจริงในข้อความ เพื่อใช้ความน่าจะเป็นตามที่เกิดจริงในข้อความ

การนับตัวอักขระที่เจอครั้งที่ 2 เป็นค่าเริ่มต้นมีข้อเสีย คือการนับความถี่ตัวอักขระที่เจอครั้งแรกสูญเปล่า ส่งผลให้ค่าความน่าจะเป็นไม่ขึ้นกับความถี่ที่เกิดตามจริงในข้อความ ดังเช่นตัวอักขระที่มีความถี่เท่ากับ 1 จะไม่มีโอกาสเกิดทำให้ต้องมีเสียเวลาในการประมวลผลมากขึ้น จากตารางที่ 2.4 วิธี PPMB อันดับที่ 1 ตัวอักขระ "ก" ตามด้วย "ค" มีความถี่ที่เกิดตามจริงในข้อความเท่ากับ 1 แต่การนับตัวอักขระวิธีนี้ทำให้ตัวอักขระ "ก" ตามด้วย "ค" ไม่มีโอกาสเกิดในอันดับที่ 1 ดังนั้นต้องใช้ P(ESC) ในการประมวลผลการเข้ารหัส-ถอดรหัสแทน ซึ่งจะส่งผลต่อกระแสข้อมูลบิตที่ยาวยิ่งขึ้น

2.4.1.3 แนวคิดในการพัฒนาวิธี PPMC

จากวิธี PPMA และวิธี PPMB ได้พัฒนาวิธี PPMC เพื่อต้องการให้มีประสิทธิภาพการบีบอัดข้อมูลที่สูงขึ้น ได้เสนอแนวคิดจากการนับจำนวนตัวอักขระให้ P(i) และจำนวนตัวอักขระที่ไม่ซ้ำกันให้ P(ESC) ตามที่เกิดจริงในข้อความ เพราะต้องการใช้พื้นฐานคำนวณความน่าจะเป็นของตัวอักขระนั้นๆ เอง ดังนั้นเมื่อตัวอักขระเกิดในข้อความ 1 ครั้งจะบวกค่าให้ความถี่ตัวอักขระของ P(i) 1 ครั้ง และบวกค่าจำนวนตัวอักขระที่ไม่ซ้ำกันให้ P(ESC) 1 ครั้ง ซึ่งวิธี PPMC มีประสิทธิภาพการบีบอัดข้อมูลที่ดีกว่าวิธี PPMB ประมาณ 15 เปอร์เซ็นต์ และดีกว่าวิธี PPMA ประมาณ 25 เปอร์เซ็นต์ (Moffat, A.: 1990) และเรียกวิธี PPMC เป็นวิธีมาตรฐาน เพราะเป็นวิธีที่มีพื้นฐานจากความน่าจะเป็นตามที่เกิดจริงในข้อความ

2.4.1.4 แนวคิดในการพัฒนาวิธี PPMD

จากวิธี PPMC ได้พัฒนาเป็นวิธี PPMD เพื่อให้ได้ประสิทธิภาพการบีบอัดที่สูงขึ้น จึงเสนอแนวคิดคือ เมื่อตัวอักขระเกิดในข้อความ 1 ครั้ง จะบวก 1/2 ให้ค่าความถี่ตัวอักขระของ P(i) และบวก 1/2 ให้ตัวอักขระที่ไม่ซ้ำกันของ P(ESC) เพื่อปรับการคำนวณความน่าจะเป็นของ P(i) ของตัวอักขระที่มีความถี่สูงให้มีความน่าจะเป็นสูงยิ่งขึ้น และอักขระที่มีความถี่ต่ำให้มีความน่าจะเป็นที่ต่ำลง จากพื้นฐานที่ว่าตัวอักขระที่มีความถี่สูง หมายถึงตัวอักขระนั้นจะมีโอกาสเกิดในข้อความสูงด้วย ดังนั้นวิธี PPMD จึงให้ประสิทธิภาพที่ดีกว่าวิธี PPMC ประมาณ 1 เปอร์เซ็นต์ (Bell, T.C. and others, 1990) จากตารางที่ 2.4 เปรียบเทียบวิธี PPMC และวิธีPPMD อันดับที่ 0 ตัวอักขระ "ก" มีความถี่สูงที่สุด และตัวอักขระ "ค" มีความถี่ที่ต่ำ ดังนั้นวิธี PPMD ให้ค่าความน่าจะเป็นของตัวอักขระ "ก" สูงกว่าวิธี PPMC และให้ค่าความน่าจะเป็นของตัวอักขระ "ค" ต่ำกว่าวิธี PPMC

การคำนวณหาค่าความน่าจะเป็นตามวิธี PPMA, วิธี PPMB, วิธี PPMC และวิธี PPMD (Sayood, K., 2000: 137-149) สามารถสรุปตามตารางที่ 2.1

ตารางที่ 2.1 แสดงการหาค่าความน่าจะเป็นตามวิธี PPMA, วิธี PPMB, วิธี PPMC และวิธี PPMD

ค่าความน่าจะเป็น	วิธี PPMA	วิธี PPMB	วิธี PPMC	วิธี PPMD
P(i)	$\frac{C_i}{N+1}$	$\frac{C_i - 1}{N}$	$\frac{C_i}{N+d}$	$\frac{C_i - 1/2}{N}$
P(ESC)	$\frac{1}{N+1}$	$\frac{d}{N}$	$\frac{d}{N+d}$	$\frac{d/2}{N}$

ตารางที่ 2.1 P(i) คือค่าความน่าจะเป็นชุดของตัวอักขระก่อนหน้าตามด้วยตัวอักขระปัจจุบันตำแหน่งที่ i ในอันดับที่กำหนด และ P(ESC) คือค่าความน่าจะเป็นชุดของตัวอักขระก่อนหน้าตามด้วยตัวอักขระปัจจุบันที่ไม่พบในอันดับที่กำหนด โดย C แทนค่าความถี่ของตัวอักขระ, d แทนจำนวนตัวอักขระปัจจุบันที่ไม่ซ้ำกัน และ N แทนผลรวมความถี่ทั้งหมดของตัวอักขระ

ข้อแตกต่างของวิธี PPMA, PPMB, PPMC และ PPMD ในการคำนวณค่า P(i) คือ วิธี PPMA และ PPMC จะนับความถี่ให้ตัวอักขระตามที่เกิดจริงในข้อความ วิธี PPMD จะใช้ความถี่ให้ตัวอักขระตามที่เกิดจริงในข้อความลบด้วย 1/2 วิธี PPMB จะเริ่มต้นนับความถี่ให้ตัวอักขระที่เกิดครั้งที่ 2 เป็นต้นไป ส่วนการคำนวณค่า P(ESC) วิธี PPMA จะใช้จำนวนตัวอักขระที่ไม่ซ้ำกันเป็น 1 เสมอ วิธี PPMB และ PPMC จะใช้จำนวนตัวอักขระที่ไม่ซ้ำกันให้ตามที่เกิดจริงในข้อความ วิธี PPMD จะนับจำนวนตัวอักขระที่ไม่ซ้ำกันให้ P(ESC) ที่เกิดจริงลดลงครึ่งหนึ่ง ดังนั้นจากข้อแตกต่างของวิธี PPM ในแต่ละวิธี สามารถแสดงการนับความถี่ของตัวอักขระ และจำนวนตัวอักขระที่ไม่ซ้ำกันได้ โดยสมมติให้มีความถี่ของตัวอักขระดังตารางที่ 2.2

ตารางที่ 2.2 แสดงความถี่ของตัวอักขระ

อักขระ	ความถี่
ก	7
ข	4
ค	3
ม	14
ส	4
รวม	32

ตารางที่ 2.3 แสดงการแทนค่าตัวแปรในสมการวิธี PPMA, วิธี PPMB, วิธี PPMC และวิธี PPMD ในอันดับที่ 0

วิธี	ผลรวมความถี่ของตัวอักขระ	จำนวนตัวอักขระที่ไม่ซ้ำกัน	ผลรวมความถี่ทั้งหมด
PPMA	32	1	33
PPMB	27	5	32
PPMC	32	5	37
PPMD	29.5	2.5	32

ตารางที่ 2.2 แสดงความถี่ของตัวอักขระ สามารถแทนค่าตัวแปรของ PPM ในแต่ละวิธีได้ดังตารางที่ 2.3 และจากตารางที่ 2.1 แสดงการคำนวณความน่าจะเป็นของ P(i) และ P(ESC) ดังตารางที่ 2.4

ตารางที่ 2.4 แสดงการคำนวณความน่าจะเป็นของ P(i) และ P(ESC) วิธี PPMA, วิธี PPMB, วิธี PPMC และวิธี PPMD สร้างจากข้อความ "กกขกขกด" อันดับที่ 0-1

วิธี PPMA

อันดับที่ 0			
ชุดของตัวอักขระก่อนหน้า	ตัวอักขระปัจจุบัน	ความถี่	การคำนวณความน่าจะเป็น
-	ก	4	4/8 = 0.5
-	ข	2	2/8 = 0.25
-	ด	1	1/8 = 0.125
-	ESC	1	1/8 = 0.125
อันดับที่ 1			
ก	ก	1	1/5 = 0.2
ก	ข	2	2/5 = 0.4
ก	ด	1	1/5 = 0.2
ก	ESC	1	1/5 = 0.2
ข	ก	2	2/3 = 0.6667
ข	ESC	1	1/3 = 0.3333

วิธี PPMB

อันดับที่ 0			
ชุดของตัวอักขระก่อนหน้า	ตัวอักขระปัจจุบัน	ความถี่	การคำนวณความน่าจะเป็น
-	ก	3	3/7 = 0.4286
-	ข	1	1/7 = 0.1429
-	ESC	3	3/7 = 0.4286
อันดับที่ 1			
ก	ข	1	1/4 = 0.25
ก	ESC	3	3/4 = 0.75
ข	ก	1	1/3 = 0.3333
ข	ESC	2	2/3 = 0.6667

ตารางที่ 2.4 (ต่อ) แสดงการคำนวณความน่าจะเป็นของ P(i) และ P(ESC) วิธี PPMA, วิธี PPMB, วิธี PPMC และวิธี PPMD สร้างจากข้อความ "กกขกขกค" อันดับที่ 0-1

วิธี PPMC				วิธี PPMD			
อันดับที่ 0				อันดับที่ 0			
ชุดของตัวอักขระก่อนหน้า	ตัวอักขระปัจจุบัน	ความถี่	การคำนวณความน่าจะเป็น	ชุดของตัวอักขระก่อนหน้า	ตัวอักขระปัจจุบัน	ความถี่	การคำนวณความน่าจะเป็น
-	ก	4	4/10 = 0.4	-	ก	3.5	3.5/7 = 0.5
-	ข	2	2/10 = 0.2	-	ข	1.5	1.5/7 = 0.2143
-	ค	1	1/10 = 0.1	-	ค	0.5	0.5/7 = 0.0714
-	ESC	3	3/10 = 0.3	-	ESC	1.5	1.5/7 = 0.2143
อันดับที่ 1				อันดับที่ 1			
ก	ก	1	1/7 = 0.1429	ก	ก	0.5	0.5/4 = 0.125
ก	ข	2	2/7 = 0.2857	ก	ข	1.5	1.5/4 = 0.375
ก	ค	1	1/7 = 0.1429	ก	ค	0.5	0.5/4 = 0.125
ก	ESC	3	3/7 = 0.4286	ก	ESC	1.5	1.5/4 = 0.375
ข	ก	2	2/3 = 0.6667	ข	ก	1.5	1.5/2 = 0.75
ข	ESC	1	1/3 = 0.3333	ข	ESC	0.5	0.5/2 = 0.25

ตารางที่ 2.4 แสดงให้เห็นว่าการนับความถี่ตัวอักขระ และการนับความถี่ให้ ESC ที่แตกต่างกันมีผลต่อการคำนวณความน่าจะเป็นของ P(i) และ P(ESC) ด้วย ดังนั้นการเลือกข้อมูลที่นำมาสร้างคลังข้อมูลฝึกสอนที่ดี ควรมีความน่าจะเป็นของตัวอักขระที่พบบ่อยในข้อความสูง ทำให้ตัวอักขระที่มีโอกาสเกิดได้บ่อยในคลังข้อมูลทดสอบจะส่งผลต่อประสิทธิภาพในการบีบอัดข้อมูลที่ดีขึ้น

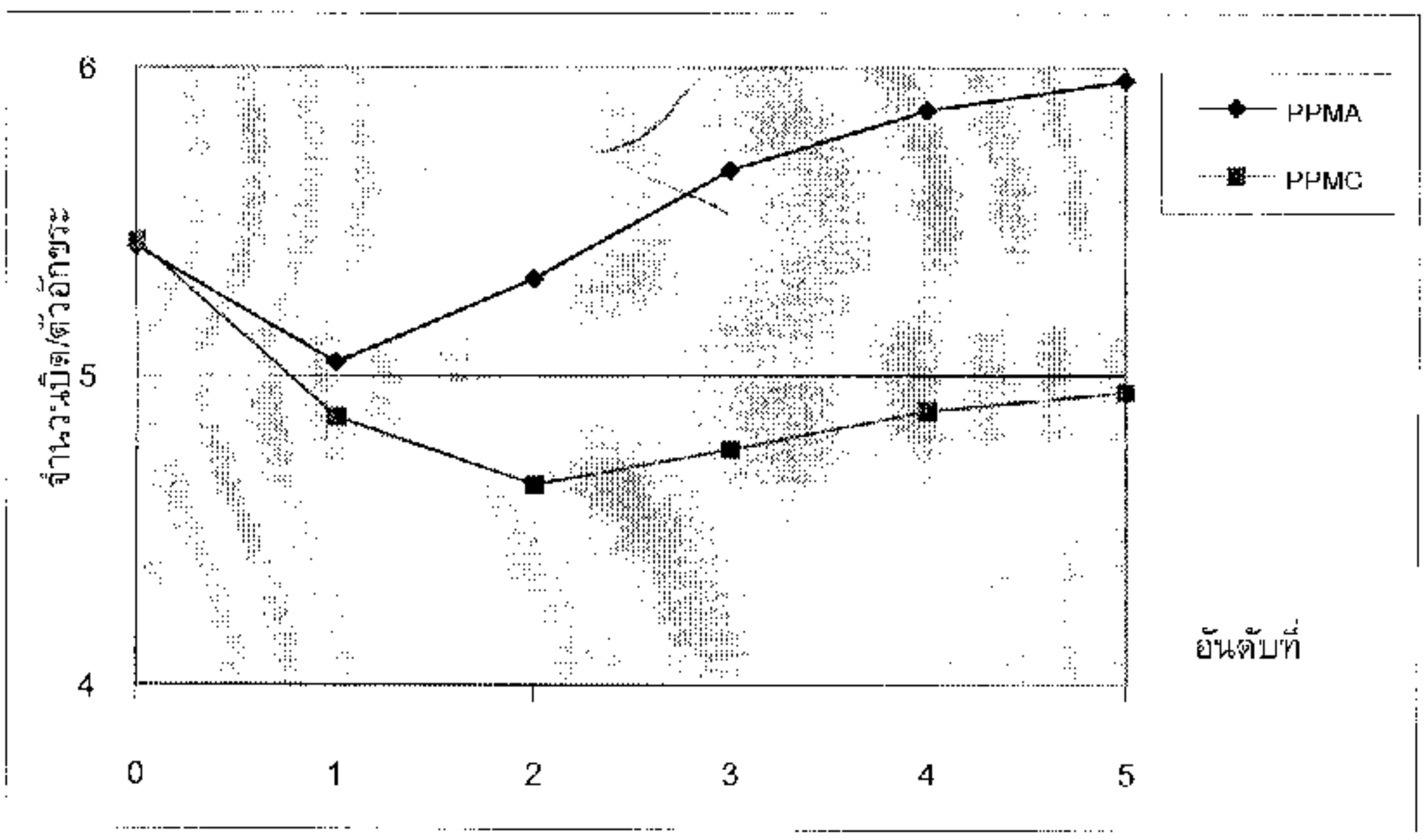
2.4.2 ปัจจัยที่มีผลต่อประสิทธิภาพในการบีบอัดข้อมูลของ PPM

จากตารางที่ 2.1-2.3 วิธี PPMA และวิธี PPMC มีการนับความถี่ของ ESC ที่ต่างกันมากที่สุด และนับตัวอักขระที่เกิดตามจริงในข้อความเหมือนกัน ดังนั้นจึงเลือก 2 วิธีนี้ เพื่อใช้เปรียบเทียบประสิทธิภาพในการบีบอัดข้อมูล โดยปัจจัยที่มีผลต่อประสิทธิภาพในการบีบอัดข้อมูล ได้แก่

2.4.2.1 การคำนวณความน่าจะเป็นของ P(ESC)

เมื่อพบปัญหา Zero-frequency คือชุดของตัวอักขระก่อนหน้าตามด้วย

ตัวอักขระปัจจุบันมีค่าความถี่เป็นศูนย์ ฝั่งผู้เข้ารหัสต้องคำนวณความน่าจะเป็นของ P(ESC) เพื่อส่งกระแสข้อมูลบิตบอกให้ฝั่งผู้ถอดรหัสทราบว่าชุดของตัวอักขระก่อนหน้าตามด้วยตัวอักขระปัจจุบันไม่เคยเกิดในอันดับที่กำหนด แสดงดังกราฟที่ 2.1 โดยใช้คลังข้อมูลฝึกสอน คือร่าย, กลอน, นิทานแก้ววิเศษ และบทความไอที และคลังข้อมูลทดสอบ คือโครงการคอมพิวเตอร์ไอซีที และประวัติศาสตร์ไทยราชวงศ์ไทย (แสดงรายละเอียดข้อมูลทดสอบในบทที่ 3)



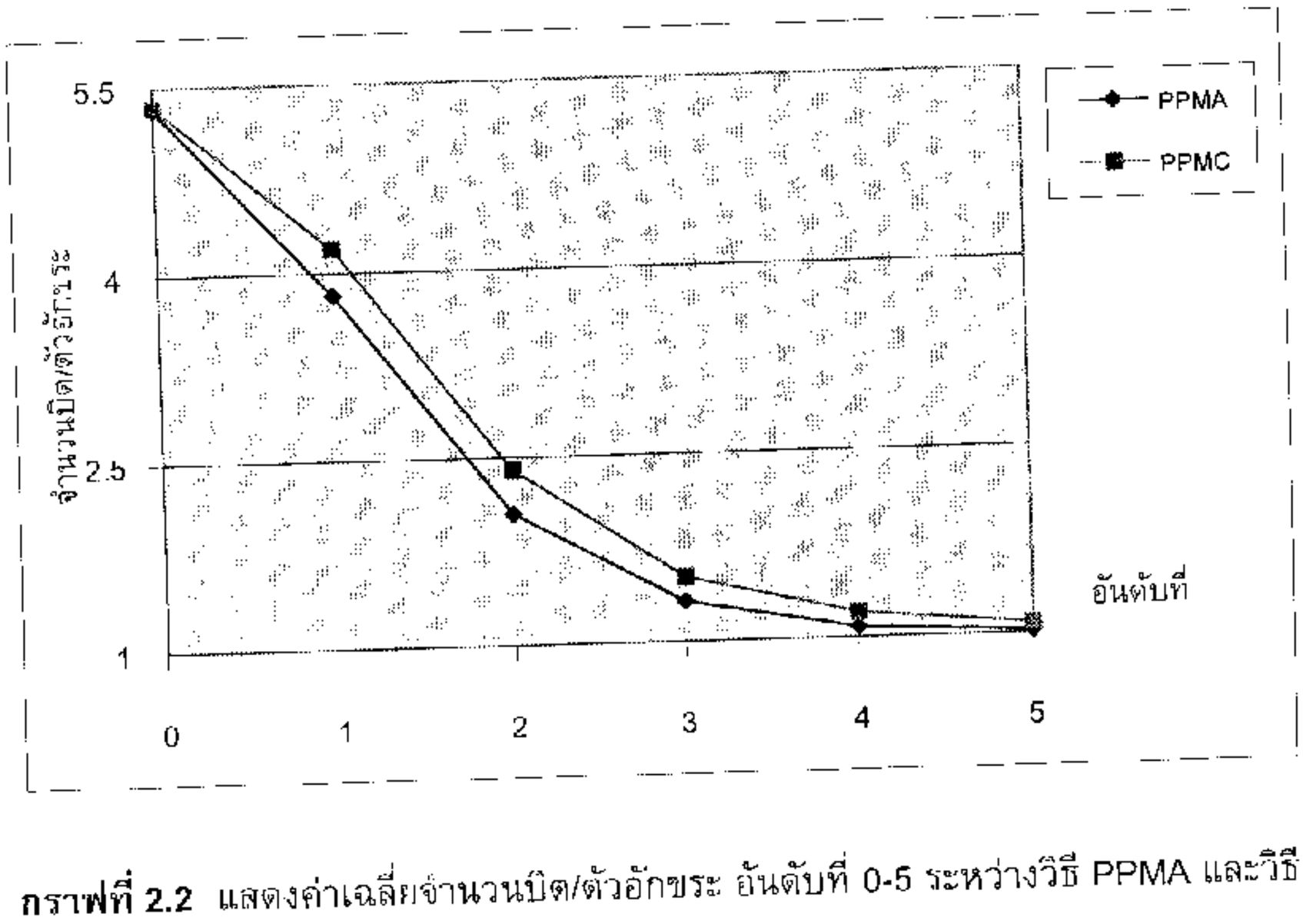
กราฟที่ 2.1 แสดงค่าเฉลี่ยจำนวนบิต/ตัวอักขระ อันดับที่ 0-5 ระหว่างวิธี PPMA และวิธี PPMC เมื่อพบปัญหา Zero-frequency

กราฟที่ 2.1 วิธี PPMC มีค่าความน่าจะเป็นของ P(ESC) ที่สูงกว่า ดังนั้นการเข้ารหัส-ถอดรหัสจะให้ค่าเฉลี่ยจำนวนบิต/ตัวอักขระที่ต่ำกว่าวิธี PPMA ที่อันดับ 0-5

2.4.2.2 การคำนวณความน่าจะเป็นของ P(i)

เมื่อไม่พบปัญหา Zero-frequency คือชุดของตัวอักขระตัวก่อนหน้าตามด้วยตัวอักขระปัจจุบันพบในคลังข้อมูลฝึกสอนในอันดับที่กำหนด ทำให้ตัวอักขระที่เข้ารหัส-ถอดรหัสมีการคำนวณความน่าจะเป็นของ P(i) โดยขึ้นกับความถี่ของตัวอักขระที่เกิดในข้อความในอันดับที่กำหนดในคลังข้อมูลฝึกสอน และจากการคำนวณความน่าจะเป็นของ P(ESC) จากหัวข้อ 2.4.2.1 และตารางที่ 2.4 แสดงได้ว่าการนับความถี่ของ ESC ที่ต่างกันส่งผลให้ผลรวมของความถี่ทั้งหมดต่างกัน ดังนั้นค่าความน่าจะเป็นของ P(i) จะแตกต่างกันด้วย แสดงดังกราฟ

ที่ 2.2 โดยใช้คลังข้อมูลฝึกสอน คือร่าย, กลอน, นิทานแก้ววิเศษ และบทความไอที มีคลังข้อมูลทดสอบ คือนิทานแก้ววิเศษ และบทความไอที (แสดงรายละเอียดข้อมูลทดสอบในบทที่ 3)



กราฟที่ 2.2 แสดงค่าเฉลี่ยจำนวนบิต/ตัวอักขระ อันดับที่ 0-5 ระหว่างวิธี PPMA และวิธี PPMC เมื่อไม่พบปัญหา Zero-frequency

กราฟที่ 2.4 และตารางที่ 2.4 พบว่า วิธี PPMA มีค่าความน่าจะเป็นของ P(i) ที่สูงกว่าวิธี PPMC ดังนั้นการเข้ารหัส-ถอดรหัสจะให้ค่าเฉลี่ยจำนวนบิต/ตัวอักขระที่ต่ำกว่าวิธี PPMC อันดับที่ 0-5 และกราฟที่ 2.1-2.2 แสดงว่าการนับความถี่ให้ตัวอักขระที่เกิดในข้อความมีผลต่อการคำนวณความน่าจะเป็นของ P(i) และการนับจำนวนตัวอักขระที่ไม่ซ้ำกันมีผลต่อการคำนวณความน่าจะเป็นของ P(ESC) ด้วย

2.4.3 หลักการของ PPM

1) สร้างตัวแบบ คือตาราง PPM ที่กำหนดอันดับเริ่มต้น สำหรับการเข้ารหัส-ถอดรหัส เช่น กำหนดอันดับเริ่มต้นที่ 2 ต้องสร้างตาราง PPM ตั้งแต่อันดับ 2, 1, 0 และ -1 แสดงดังภาพที่ 2.3

2) นำคลังข้อมูลทดสอบมาผ่านอัลกอริทึมการเข้ารหัส โดยอาศัยค่าของชุดสถิติจากตัวแบบในการคำนวณจะได้ผลลัพธ์ของการเข้ารหัส คือกระแสข้อมูลบิต

3) การถอดรหัสให้นำค่ากระแสข้อมูลบิตที่ได้จากการเข้ารหัส ผ่านอัลกอริทึมการถอดรหัส โดยอาศัยค่าสถิติจากตัวแบบในการคำนวณจะได้ผลลัพธ์เป็นคลังข้อมูลทดสอบดังเดิม

order 2

Prediction			c	p
be	→	o	1	1/2
	→	esc	1	1/2
eo	→	r	1	1/2
	→	esc	1	1/2
no	→	t	1	1/2
	→	esc	1	1/2
ob	→	e	2	2/3
	→	esc	1	1/3
or	→	n	1	1/2
	→	esc	1	1/2
ot	→	t	1	1/2
	→	esc	1	1/2
rn	→	o	1	1/2
	→	esc	1	1/2
to	→	b	2	2/3
	→	esc	1	1/3
tt	→	o	1	1/2
	→	esc	1	1/2

order 1

Prediction			c	p
b	→	e	2	2/3
	→	esc	1	1/3
e	→	o	1	1/2
	→	esc	1	1/2
n	→	o	1	1/2
	→	esc	1	1/2
o	→	b	2	2/7
	→	r	1	1/7
	→	t	1	1/7
	→	esc	3	3/7
r	→	n	1	1/2
	→	esc	1	1/2
t	→	o	2	2/5
	→	t	1	1/5
	→	esc	2	2/5

order 0

Prediction		c	p
→	b	2	2/19
→	e	2	2/19
→	n	1	1/19
→	o	4	4/19
→	r	1	1/19
→	t	3	3/19
→	esc	6	6/19

order -1

→	A	1	1 / A|

ภาพที่ 2.3 ตัวแบบอันดับเริ่มต้นที่ 2 หลังการประมวลผลข้อความ "tobeornottobe" ด้วยวิธี PPMC

2.4.3.1 อัลกอริทึมการเข้ารหัส

กำหนดให้ ตัวแปร: $U^{(n)}$ = ค่าขอบบนปัจจุบัน

$L^{(n)}$ = ค่าขอบล่างปัจจุบัน

$Cum\_Count(X_{n-1})$ = ค่าความถี่สะสมของตัวอักขระก่อนหน้า

$Cum\_Count(X_n)$ = ค่าความถี่สะสมของตัวอักขระปัจจุบัน

การกำหนดความยาวของคำ (Word length) ต้องกำหนดจำนวนบิตที่ใช้ได้เพียงพอในการแสดงถึงความถี่ที่เกิดของตัวอักขระ และกำหนดตัวนับเงื่อนไข E3 เริ่มต้นที่ศูนย์

$$L^{(n)} = L^{(n-1)} + \left\lfloor \frac{(U^{(n-1)} - L^{(n-1)} + 1) \times Cum\_Count(X_{n-1})}{TotalCount} \right\rfloor$$

$$U^{(n)} = L^{(n-1)} + \left\lfloor \frac{(U^{(n-1)} - L^{(n-1)} + 1) \times Cum\_Count(X_n)}{TotalCount} \right\rfloor - 1$$

การเลื่อนบิต (Shift bit) ของการเข้ารหัส

1. ถ้า MSB (Most Significant Bit) ของ L และ U เหมือนกันให้เลื่อนบิดไปทางซ้ายดังนี้
   1.1. เลื่อนบิต "0" เข้าทาง LSB (Least significant bit) ของ L เขียนสัญลักษณ์ได้ L ← 0
   1.2. เลื่อนบิต "1" เข้าทาง LSB ของ U เขียนสัญลักษณ์ได้ U ← 1
   1.3. ทำข้อ 1.1-1.2 จนกระทั่งได้บิต MSB ของ L และ U ต่างกัน เช่น L = 0000 และ U = 1111
2. ถ้าเกิดเงื่อนไข E3 หมายถึง บิต MSB ต่างกัน และบิดที่ 2 ของ U = 0 และ L = 1 ต้องแก้ปัญหาเงื่อนไข E3 โดยการขยายช่วง L และ U ดังนี้
   2.1. เลื่อนบิด "0" เข้าทาง LSB ของ L 1 บิต
   2.2. เลื่อนบิด "1" เข้าทาง LSB ของ U 1 บิต
   2.3. สลับบิต (Complement) MSB ของ L และ U ใหม่
3. กรณีที่ไม่พบตัวอักขระที่เข้ารหัสในอันดับปัจจุบัน ต้องใช้ค่าความน่าจะเป็นของ ESC นำไปคำนวณในอัลกอริทึมการเข้ารหัสแทน และลดอันดับลงโดยตัดชุดของตัวอักขระก่อนหน้าตัวที่ 1 ออก ดังเช่น จากภาพที่ 2.3 แสดงตัวอักขระก่อนหน้า 2 ตัวเพื่อทำนายตัวอักขระปัจจุบัน เช่น ให้ "ob" เป็นชุดของตัวอักขระก่อนหน้า และตัวปัจจุบันที่ต้องการเข้ารหัส คือ "e" เขียนสัญลักษณ์ได้ "ob" ->"e" การทำงานจะตรวจสอบคลังข้อมูลฝึกสอนจากอันดับที่ 2 เพื่อตรวจสอบว่า "ob" เคยเกิดในอันดับ 2 หรือไม่ ถ้าไม่จะลงไปทำที่อันดับ 1 เลย และตัดชุดของตัวอักขระก่อนหน้าตำแหน่ง MSB ออก เขียนได้ "b" -> "e" แต่ถ้าพบในอันดับ 2 แต่ไม่เคยเกิดชุดตัวอักขระก่อนหน้า "ob" ที่ตามด้วย "e" เลย จะใช้ค่าความน่าจะเป็นของ "ob" -> ESC เพื่อส่งไปทำการเข้ารหัส และไปทำอันดับ 1 แต่ถ้าพบในอันดับ 2 คือชุดของตัวอักขระก่อนหน้า "ob" ตามด้วย "e" จะใช้ความน่าจะเป็นของตัวอักขระปัจจุบัน คือ "e" เพื่อไปทำการเข้ารหัส

2.4.3.2 อัลกอริทึมการถอดรหัส

กำหนดให้ ตัวแปร: t = ขนาดของความยาวของคำโดยอ่านมาจากกระแสข้อมูลบิดที่ได้จากการเข้ารหัส

Scale3 = ตัวนับเงื่อนไข E3 เริ่มต้นที่ศูนย์

U = ค่าขอบบนปัจจุบัน

L = ค่าขอบล่างปัจจุบัน

Cum_Count(X –1) = ค่าความถี่สะสมของตัวอักขระก่อนหน้า

Cum_Count(X) = ค่าความถี่สะสมของตัวอักขระปัจจุบัน

$$\text{ความถี่สะสมของตัวอักขระ} = \left\lfloor \frac{(t - L + 1) \times \text{Total Count} - 1}{U - L + 1} \right\rfloor$$

$$L = L + \left\lfloor \frac{(U - L + 1) \times \text{Cum\_Count}(X-1)}{\text{TotalCount}} \right\rfloor$$

$$U = L + \left\lfloor \frac{(U - L + 1) \times \text{Cum\_Count}(X)}{\text{TotalCount}} \right\rfloor - 1$$

การเลื่อนบิตในการถอดรหัส

1. ถ้าบิต MSB ของ L และ U เหมือนกันให้เลื่อนบิตไปทางซ้ายโดย
   1.1. เลื่อนบิต “0” เข้าทาง LSB ของ L เขียนสัญลักษณ์ได้ L ← 0
   1.2. เลื่อนบิต “1” เข้าทาง LSB ของ U เขียนสัญลักษณ์ได้ U ← 1
   1.3. t เลื่อนบิต MSB ออก 1 บิต และเลื่อนบิตต่อไปจากกระแสข้อมูลบิต
2. ถ้า MSB ของ L และ U ต่างกัน (ค่า L และ U ไม่ต้องเลื่อนบิตไปทางซ้าย) และไม่เกิดกรณีเงื่อนไข E3 ให้ใช้ t ตัวเดิม
3. ถ้าเกิดกรณีเงื่อนไข E3 หมายถึงบิต MSB มีค่าต่างกัน และบิตที่ 2 ของ U = 0 และ L = 1 มีขั้นตอนดังนี้
   3.1. เลื่อนบิต “0” เข้าทาง LSB ของ L 1 บิต
   3.2. เลื่อนบิต “1” เข้าทาง LSB ของ U 1 บิต
   3.3. เลื่อนบิตต่อไปจากกระแสข้อมูลบิตเข้าทาง LSB
4. สลับบิต MSB ของ L, U และ t ใหม่
5. การเลื่อนบิตไปทางซ้ายทำจนกระทั่งบิด MSB L และ U ต่างกัน
6. จบการถอดรหัส เมื่อ ค่า t = L

2.5 งานวิจัยที่เกี่ยวข้อง

2.5.1 วิธีรหัสฮัฟแมนที่ปรับเปลี่ยน (THUFF)

วิธี THUFF ให้ประสิทธิภาพการบีบอัดข้อมูลภาษาไทยประมาณ 46 เปอร์เซ็นต์ (เรวดี ลิมปิโชติกุล, 2534: 31-37) โดยนำเสนอเทคนิคสำหรับการเข้ารหัส ซึ่งเป็นการแทนข้อมูลรหัสเดิมของข้อมูลด้วยรหัสใหม่เป็นไบนารี (Binary code) ตามรหัสฮัฟแมน ซึ่งเป็นการแทนรหัส 1 ครั้งต่อ 1 ตัวอักขระ แต่ในหลักการของ THUFF คือเมื่อแทนรหัสต้องตรวจสอบว่ากลุ่มอักขระที่เข้ารหัสตรงกับคำในตารางวิเคราะห์ข้อมูลคำไทยหรือไม่ ถ้าไม่ตรงกับคำในตารางวิเคราะห์ข้อมูลคำไทย จะแทนตามอัลกอริทึมปกติ แต่ถ้าตรงกับคำในตารางจะแทนด้วยอักขระพิเศษ ซึ่งเป็นอักขระที่ไม่มีในข้อมูลประเภทข้อความ และตามด้วยรหัสของคำคำนั้นที่ตรงกับในตารางวิเคราะห์ ข้อมูลคำไทย

การอาศัยตารางวิเคราะห์ข้อมูลคำไทย (ยืน ภู่วรวรรณ, 2527: 290-299) เป็นปัจจัยหลักที่ช่วยให้การบีบอัดมีประสิทธิภาพเพิ่มขึ้น ดังนั้นถ้าในการบีบอัดข้อความนั้นไม่มีคำตรงกับคำไทยในตารางที่ได้จัดเตรียมไว้จะทำให้ประสิทธิภาพการบีบอัดลดลง

วิธี THUFF ใช้ลักษณะความหมาย (Semantic) ภาษาไทยจะมีประสิทธิภาพที่ดีขึ้น ถ้านำไปใช้กับข้อมูลภาษาไทยล้วน และให้ประสิทธิภาพการบีบอัดข้อมูล และเวลาในการประมวลผลที่สูงกว่ารหัสฮัฟแมนมาก เพราะต้องตรวจสอบคำกับคำไทยในตารางในแต่ละรอบที่อ่านข้อมูล

ตัวอย่าง สมมติแฟ้มข้อมูลมีข้อความ "การรับงาน" กำหนดให้รหัสพิเศษ คือรหัส 255 ในตารางแอสกี และกำหนดให้ตารางคำไทยมีคำว่า "การ" และ"งาน" ในตำแหน่งที่ 20 และ 21 ตามลำดับ ดังภาพที่ 2.4

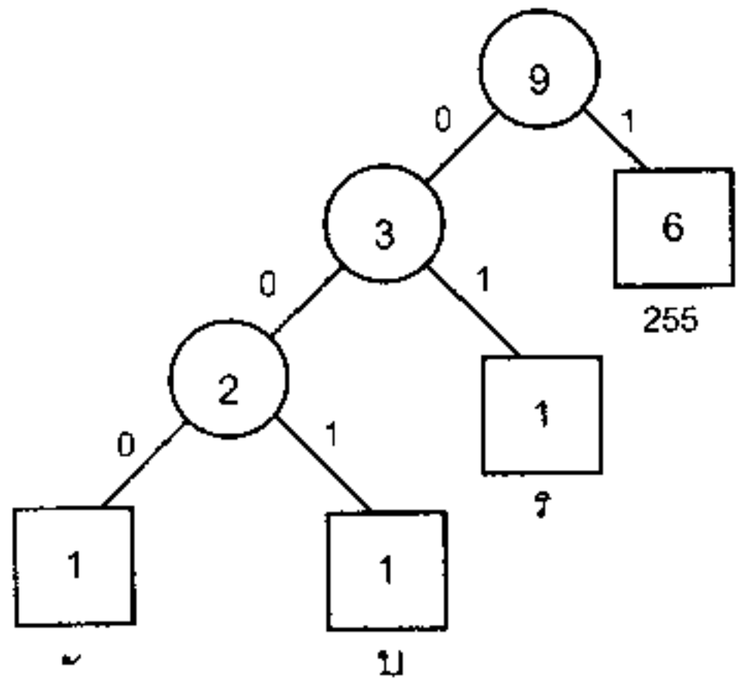
รหัส	ความถี่	รหัสใหม่	ความยาวรหัสใหม่
ร	1	01	2
บ	1	001	3
ั	1	000	3
.	.	.	.
255	6	1	1

ตารางรหัส

รหัสคำ	คำไทย
.	.
20	การ
21	งาน
.	.

ตารางคำไทย

ภาพที่ 2.4 แสดงตารางรหัสแบบฮัฟแมน และตารางคำไทย



ภาพที่ 2.5 ต้นไม้แบบฮัฟแมน

นำตารางรหัสที่มีความถี่ของตัวอักขระตามภาพที่ 2.4 มาสร้างต้นไม้แบบฮัฟแมน ดังภาพที่ 2.5 การเข้าแทนรหัสจะตรวจสอบกลุ่มอักขระที่เข้ารหัสตรงกับคำในตารางคำไทยหรือไม่ ถ้าไม่ตรงจะเป็นการแทนรหัส 1 ครั้งต่อ 1 ตัวอักขระ แต่ถ้ากลุ่มอักขระนั้นตรงกับคำในตารางไทย โดยแทนรหัส 2 ครั้ง คือการแทนรหัสพิเศษ คือรหัส 255 และแทนรหัสคำคำนั้นจากภาพที่ 2.4 ได้ข้อมูลเข้ารหัสตามภาพที่ 2.6 ดังนี้

1	00010100	01	000	001	1	00010101
255	(การ)	ร	ั	บ	255	(งาน)

ภาพที่ 2.6 ข้อมูลที่เข้ารหัส

2.5.2 วิธีรหัสแอลแซดดับเบิลยูที่ปรับเปลี่ยน (TLZW)

วิธี TLZW ให้ประสิทธิภาพการบีบอัดข้อมูลภาษาไทยประมาณ 55 เปอร์เซ็นต์ (เรวดี ลิมปิโชดิกุล, 2534: 37-43) โดยนำเสนอเทคนิคเพื่อพัฒนาจากวิธีอัลกอริทึมแอลแซดดับเบิลยู (LZW) ที่ไม่ต้องศึกษาความหมายในภาษาไทยแต่อาศัยการเรียนรู้ข้อมูล โดยการปรับตารางรหัส และกำหนดรหัสเริ่มต้น 256 รหัสตามค่าตารางแอสกี (ASCII table) เมื่อทำการบีบอัดข้อมูลจะเรียนรู้ข้อมูล โดยเปลี่ยนข้อมูลในลักษณะ <รหัสหน้า><รหัสท้าย> นำมาคำนวณผ่านตารางแฮช เพื่อให้ได้ตำแหน่งใหม่ที่เก็บเข้าตารางรหัส และจัดเตรียมแฟ้มรหัสคำไทย โดยกระจายกลุ่มอักขระของคำจากตัวแรก และตัวถัดไปจนถึงตัวสุดท้าย นำไปเปลี่ยนรูปกลุ่มอักขระให้เป็นรูปแบบของ <รหัสหน้า><รหัสท้าย> นำไปผ่านฟังก์ชันแฮชได้ตำแหน่ง และค่าในตารางแฮช เพื่อใช้ในการบีบอัด-ขยายข้อมูล วิธีนี้ให้ประสิทธิภาพในการบีบอัดข้อมูลไม่ดีถ้าใช้

ข้อความที่มีขนาดไม่ใหญ่พอในการสร้างตารางรหัสก่อนเข้ารหัสข้อมูล วิธีอัลกอริทึมของแอลแซดดับเบิลยูที่ปรับเปลี่ยนไม่ใช้ความหมายในภาษาไทย ดังนั้นจึงเหมาะกับข้อมูลทั่วไป เช่น ข้อความภาษาไทยปนกับข้อความภาษาอังกฤษ ซึ่งให้ประสิทธิภาพการบีบอัดข้อมูลที่ดีกว่าวิธีอัลกอริทึมของแอลแซดดับเบิลยู และกรณีข้อมูลมีขนาดใหญ่เวลาที่ใช้ในการบีบอัด-ขยายข้อมลจะใกล้เคียงกัน เพราะอัลกอริทึมได้มีจำนวนครั้งในการแทนรหัสที่น้อยลง ถึงแม้จะใช้เวลาส่วนหนึ่งในการสร้างรหัสคำไทยก็ตาม แต่กรณีข้อมูลมีขนาดไม่ใหญ่มากเวลาที่ใช้ในการบีบอัด-ขยายข้อมูลจะมากกว่าวิธีเดิม อย่างไรก็ตามประสิทธิภาพวิธีรหัสฮัฟแมนที่ปรับเปลี่ยนก็ยังดีกว่าใคร ตัวอย่างการเข้ารหัสในทั้งสองกรณีได้แก่ ถ้านำข้อความ "การงานการบ้านการงาม" มาเข้ารหัสด้วยวิธีอัลกอริทึมของแอลแซดดับเบิลยู จะได้ผลลัพธ์ตามภาพที่ 2.7 และ 2.8 ตามลำดับ

0 ... 255	รหัสแอสกี		
256	กา	->	161า
257	าร	->	210ร
258	รง	->	195ง
259	งา	->	167า
260	าน	->	210น
261	นก	->	185ก
262	การ	->	256ร
263	รบ	->	195บ
264	บ้	->	186 ้
265	้า	->	233า
266	านก	->	260ก
267	การง	->	262ง
268	งาม	->	259ม

ภาพที่ 2.7 แสดงรหัสที่สร้างจากอัลกอริทึมของแอลแซดดับเบิลยู

ก	า	ร	ง	า	น	กา	ร	บ	้	าน	การ	งา	ม

ภาพที่ 2.8 แสดงข้อมูลที่เข้ารหัสโดยอัลกอริทึมของแอลแซดดับเบิลยู

ถ้านำข้อความเดียวกันมาเข้ารหัสด้วยอัลกอริทึมของแอลแซดดับเบิลยูที่ปรับเปลี่ยน จะได้ผลลัพธ์ตามภาพที่ 2.9 และ 2.10 ตามลำดับ

0 ... 255	รหัสแอสกี		
256	...	->	...
257	...	->	...
...	...	->	...
274	กา	->	161า
275	การ	->	274ร
...	...	->	...
323	งา	->	167า
324	งาน	->	323น
...	...	->	...
780	...	->	...
781	การง	->	275ง
782	งานก	->	324ก
783	การบ	->	275บ
784	บ้านก	->	456ก
785	การงา	->	781า
786	าม	->	210ม

ภาพที่ 2.9 แสดงรหัสที่สร้างจากอัลกอริทึมของแอลแซดดับเบิลยูที่ปรับเปลี่ยน

การ	งาน	การ	บ้าน	การง	า	ม

ภาพที่ 2.10 แสดงข้อมูลที่เข้ารหัสโดยอัลกอริทึมของแอลแซดดับเบิลยูที่ปรับเปลี่ยน

การบีบอัดข้อมูลด้วยวิธีรหัสฮัฟแมน และวิธีแอลแซดดับเบิลยู จะให้ประสิทธิภาพการบีบอัดข้อมูลต่ำกว่าวิธี PPM แต่มีอัลกอริทึมที่ซับซ้อนน้อยกว่าวิธี PPM อย่างไรก็ตาม วิทยานิพนธ์นี้จะเน้นความมีประสิทธิภาพในการบีบอัดข้อมูลมากกว่าเวลาหรือ ความซับซ้อนของอัลกอริทึม ดังนั้นจึงเลือกวิธีการบีบอัดข้อมูลภาษาไทยด้วยวิธี PPM เป็นหลักในการพัฒนาวิธีที่จะใช้ประสิทธิภาพที่สูงขึ้น

บทที่ 3

การวิเคราะห์และการออกแบบ

3.1 การศึกษาและวิเคราะห์

3.1.1 การวิเคราะห์ลักษณะภาษาไทย

3.1.1.1 ลักษณะภาษาไทย

ตัวอักขระในภาษาไทยแบ่งออกเป็น 4 ระดับเรียกว่า พหุระดับ (Multi-level) แสดงรายละเอียดดังตารางที่ 3.1

ตารางที่ 3.1 แสดงระดับของอักขระภาษาไทย

ระดับ	เป็นระดับของอักขระ
1	วรรณยุกต์ ได้แก่ ไม้เอก, ไม้โท, ไม้ตรี, ไม้จัตวา และการันต์
2	อยู่ต่ำกว่าระดับที่ 1 ได้แก่ ไม้หันอากาศ, สระอิ, สระอี, สระอึ, สระอือ, ไม้ไต่คู้
3	เป็นระดับหลักของภาษาไทย อยู่ต่ำกว่าระดับที่ 2 ได้แก่ พยัญชนะ 44 ตัว รวมทั้ง ฤ ฦ ะ า ๅ ำ เ แ โ ใ ไ ๆ และ ฯ
4	ระดับต่ำสุด เป็นสระ ได้แก่ อุ, อู และอิ

ภาษาไทยเป็นภาษาที่มีลักษณะการเขียนติดกัน ไม่มีเครื่องหมายบอกขอบเขต ยกเว้นกรณีจบประโยค การป้อนข้อความภาษาไทยเข้าคอมพิวเตอร์จะทำในลักษณะที่เป็นเชิงเส้น คือ ทำให้เกิดการผสมระหว่างพยัญชนะ, สระ และสัญลักษณ์ในสายอักขระ

3.1.1.2 ลักษณะพยัญชนะประสม

พยัญชนะประสม (บรรเทา กิตติศักดิ์ และกัมพุชนาฏ เปรมกมล, 2530: 60-61) คือ พยัญชนะ 2 ตัวประสมกัน โดยพิจารณาในแง่การออกเสียงมี 2 ประเภท ได้แก่ อักขระควบแท้ คือ พยัญชนะประสมที่ออกเสียงพยัญชนะทั้ง 2 ตัว พร้อมกันสนิทจนเป็นเสียงเดียวกัน มี 12 เสียง 16 รูป ตามตารางที่ 3.2 และอักขระควบไม่แท้ คือพยัญชนะที่ควบกล้ำ กับ

พยัญชนะ “ร” แต่ออกเสียงเหมือนพยัญชนะเดี่ยว ไม่ออกเสียงเป็นพยัญชนะควบกล้ำแต่รูปเป็นพยัญชนะประสม ออกเสียงเฉพาะพยัญชนะตัวแรกเท่านั้น ดังตารางที่ 3.3

ตารางที่ 3.2 แสดงพยัญชนะประสมประเภทอักขระควบแท้

อักขระควบแท้	ตัวอย่าง
/กร/	กราบ กระ กราน เกรง กร้าน กริ่ง
/กล/	กลับกลาย เกลี้ยงเกลา เกลื่อน กลาด เกลียด กลัว กลอก
/กว/	กว้าง กวัดแกว่ง กวาด ไกว แกว่ง แกว่น
/คร/, /ขร/	ขรุขระ ขรัว ตรึกโครม
/คล/, /ขล/	คละ คลุ้ง คล้าย โคลง คล้าม คล่อง ขลุ่ย ขลุก ขลิก ขลุม ขลุ่ย
/คว/, /ขว/	ควาย ความ คว้า ควัก ขวนขวาย แขวน ขวาน ไขว่ ขวับ
/ตร/	ตริตรอง ตรวจตรา ตรู่ ตราบ ตรอก ตรม ตรอม
/ทร/	นิทรา จันทรา
/ปร/	ปราบปราม ปรับปรุง ปรับ แปร๊ด แประ ปรุง เปรียว
/ปล/	เปลี่ยนแปลง ปลอม ปล้น เปล้า ปลิง
/พร/	พร้อม แพรวพราว พราย พร่ำ พรั่น พรึง
/พล/, /ผล/	พลับพลึง พลับพลา พลิก พล่าม พลอย พลั้ง ผลุนผลัน เผลอไผล

มีอักขระควบแท้ที่เกิดขึ้นใหม่ เช่น เสียง /ดร/ /ฟร/ /ฟล/ /บร/ /บล/ ได้รับอิทธิพลจากภาษาต่างประเทศ โดยเฉพาะภาษาอังกฤษ เช่น ดรัมเมเยอร์ ฟรี ฟลอร์ บรอดเวย์ บลูส์ เป็นต้น

ตารางที่ 3.3 แสดงพยัญชนะประสมประเภทอักขระควบไม่แท้

อักขระควบไม่แท้	ออกเสียงพยัญชนะ	ตัวอย่าง
/จร/	/จ/	จริง
/ซร/, /ศร/, /สร/	/ซ/, /ศ/, /ส/	ไซร้ เศร้า สร้าง
/ทร/	/ซ/	ทราบ ทรัพย์ ทรง

3.1.2 คุณลักษณะของข้อมูลภาษาไทยที่ใช้ในการศึกษา

หัวข้อนี้ได้กล่าวถึงการศึกษาคุณลักษณะของข้อมูลที่ใช้ในการศึกษา

3.1.2.1 เอกสารที่ใช้การศึกษา

งานวิจัยนี้ศึกษาข้อมูลภาษาไทยที่ใช้ทั่วไปในชีวิตประจำวัน ดังนั้นคลังข้อมูลฝึกสอน และคลังข้อมูลทดสอบได้รวบรวมจากเอกสารต่าง ๆ ดังตารางที่ 3.4

ตารางที่ 3.4 แสดงรายละเอียดของเอกสารที่ใช้ในการศึกษา

เนื้อหา	ขนาด (ไบต์)	จำนวนพยัญชนะ (%)	จำนวนสระ, สัญลักษณ์ (%)	พยัญชนะประสม (%)
ร่าย	584	57.1918	42.8082	7.2289
กลอน	887	53.664	46.336	7.8059
นิทานแก้ววิเศษ	2,074	52.8447	47.1553	7.6782
บทความไอที	3,552	58.7684	41.2316	11.3581
โครงการคอมพิวเตอร์ไอซีที	4,466	57.6646	42.3354	9.8592
ประวัติศาสตร์ไทยราชวงศ์ไทย	8,278	59.4624	40.5376	9.4928
สามก๊ก	9,314	57.0026	42.9974	5.5826
รวมนิทานพื้นบ้าน	9,997	54.8655	45.1345	6.4742
มงคลชีวิต	18,244	53.215	46.785	7.1719
ไม้ประดับ	23,688	55.0194	44.9806	6.3618

● ร่าย (กำชัย ทองหล่อ, 2537: 456) เป็นคำประพันธ์ที่เรียงคำให้คล้องจองกัน มีเนื้อหาเกี่ยวกับคำสอนของแม่สอนลูก, กลอน (กำชัย ทองหล่อ, 2537: 464-466) เป็นคำประพันธ์ที่เรียงมีสัมผัสกัน แต่ไม่บังคับเอกโท และครุลหุ ประกอบด้วยกลอนหก เนื้อหาเกี่ยวกับการทำความดีแล้วได้ดี กลอนเจ็ดและแปด เนื้อหาเกี่ยวกับการพูด กลอนเก้า เนื้อหาเกี่ยวกับการรักประเทศ ซื่อสัตย์ สามัคคี ขยัน และยุติธรรม, นิทานแก้ววิเศษ (วิชิต นานาน, 2546) เป็นนิทานพื้นบ้านมีเนื้อหาเกี่ยวกับแก้ววิเศษ, บทความไอที (ศูนย์เทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ, 2546) เป็นเรื่องของการใช้บัตรอิเล็กทรอนิกส์เพื่อยืนยันตัวตนของผู้ถือบัตรในการใช้บริการด้านต่างๆ, โครงการคอมพิวเตอร์ไอซีที (ลลิดา รัตนศรีทัย, 2546) เกี่ยวกับการดำเนินงานโครงงานด้านคอมพิวเตอร์ที่ไม่ประสบผลสำเร็จ, ประวัติศาสตร์ไทยราชวงศ์ไทย (การท่องเที่ยวแห่งประเทศไทย, 2544) เป็นหลักฐานทางโบราณคดีของชนชาติที่ครอบครองดินแดนไทยในยุคต่าง ๆ ตั้งแต่พ่อขุนศรีอินทราทิตย์กษัตริย์พระองค์แรกจนถึงพระบาทสมเด็จพระเจ้าอยู่หัวภูมิพลอดุลยเดช รัชกาลปัจจุบัน, สามก๊ก (กิติพัท มหาพันธ์, 2541) เป็นเนื้อหาเรื่องสามก๊ก ตอนคำสาบานในสวนดอกท้อ, ตอนการณ์แปรปรวนในราชสำนัก, ตอนทรราชย์ตั๋งโต๊ะ, ตอนรวบรวมหัวเมืองสู้ตั๋งโต๊ะ, ตอนโยกย้าย

เมืองหลวง, ตอนซุนเกี๋ยนถึงแก่กรรม, ตอนกลหญิงงาม, ตอนลิฉุย กุยกี เข้าราชสำนัก และตอนโจโฉกรีฑาทัพเหยียบชีจิ๋ว, รวมนิทานพื้นบ้าน (วิชิต นานาน, 2546) เป็นนิทานพื้นบ้านที่มีเนื้อหาดังนี้ เรื่องความโลภ มีเนื้อหาเกี่ยวกับชายอาชีพทำไร่ที่มีความโลภกับฝูงลิง, เรื่องน้ำผึ้งหยดเดียว มีเนื้อหาเกี่ยวกับการทะเลาะกันโดยไม่มีเหตุผล, เรื่องกบขี้เกียจ มีเนื้อหาเกี่ยวกับความขี้เกียจของกบกับปูนาที่มีความขยัน, เรื่องแก้ววิเศษ มีเนื้อหาเกี่ยวกับแก้ววิเศษ, เรื่องความพยายามของเด็กน้อย, เรื่องช้างของพระราชา และกบกาเจ็ดปาก, มงคลชีวิต (ชมรมพุทธศาสตร์สากลในอุปถัมภ์สมเด็จพระมหารัชมังคลาจารย์, 2547) เป็นเนื้อหาเกี่ยวกับข้อปฏิบัติในการสร้างชีวิต และเป็นการฝึกใจโดยตรง, ไม้ประดับ (สำนักบริการคอมพิวเตอร์ มหาวิทยาลัยเกษตรศาสตร์, 2540) เป็นเนื้อหาเกี่ยวกับโป๊ยเซียน

3.1.2.2 พยัญชนะ, สระ และสัญลักษณ์ที่นำมาทดลอง

พยัญชนะมี 46 ตัว สระ-สัญลักษณ์มี 47 ตัว รวมสิ้น 93 ตัว แสดงในตารางที่ 3.5

ตารางที่ 3.5 แสดงพยัญชนะ, สระ และสัญลักษณ์ที่นำมาใช้ในการทดลอง

พยัญชนะ	พยัญชนะ	พยัญชนะ	สระ-สัญลักษณ์	สระ-สัญลักษณ์	สระ-สัญลักษณ์
ก	ฒ	ร	สระอะ	ไม้โท	เลขสาม
ข	ณ	ฤ	สระอา	ไม้ตรี	เลขสี่
ฃ	ด	ล	สระอิ	ไม้จัตวา	เลขห้า
ค	ต	ฦ	สระอี	ไม้ไต่คู้	เลขหก
ฅ	ถ	ว	สระอึ	การันต์	เลขเจ็ด
ฆ	ท	ศ	สระเอ	ไม้ยมก	เลขแปด
ง	ธ	ษ	สระแอ	ไปยาลน้อย	เลขเก้า
จ	น	ส	สระอุ	นขลิขิตปิด	ปรัศนี
ฉ	บ	ห	สระอู	นขลิขิตเปิด	เซมิโคลอน (Semi-colon)
ช	ป	ฬ	สระเอือ	มหัพภาค	โคลอน (Colon)
ซ	ผ	อ	ไม้โอ	ช่องว่าง	ยติภังค์
ฌ	ฝ	ฮ	ไม้มลาย	\r (Carriage return)	จุลภาค
ญ	พ		ไม้ม้วน	\n (New line)	อัศเจรีย์
ฎ	ฟ		สระอำ	\a (Bell)	
ฏ	ภ		ไม้หันอากาศ	เลขศูนย์	
ฐ	ม		ฟันหนู	เลขหนึ่ง	
ฑ	ย		ไม้เอก	เลขสอง	

3.2 แนวคิดในการออกแบบ

วิธีการของ PPM เริ่มต้นจากการบีบอัดข้อมูลภาษาอังกฤษ (Yingying, 2001: 58-63; Moffat, A., 1990) แต่ภาษาไทยมีคุณลักษณะแตกต่างไปจากภาษาอื่น คือตัวอักขระในภาษาไทยแบ่งออกเป็น 4 ระดับ และลักษณะพยัญชนะประสม ดังนั้นจึงมีแนวคิดในการปรับปรุงวิธี PPM จากการศึกษาคุณลักษณะของภาษาไทยโดยแบ่งเป็นประเด็นหลัก ๆ ดังนี้

3.2.1 แนวคิดในการพัฒนาวิธีที่ปรับปรุง

จากวิธี PPMA, วิธี PPMB, วิธี PPMC และวิธี PPMD ที่พัฒนาเพื่อขึ้นใช้กับภาษาอังกฤษ เมื่อนำมาทดสอบกับภาษาไทยส่งผลให้จำนวนความถี่ของตัวอักขระที่เกิดตามชุดของตัวอักขระก่อนหน้าส่วนใหญ่มีความถี่ต่ำ เพราะภาษาไทยมีรูปแบบของการผสมระหว่างพยัญชนะ, สระ และสัญลักษณ์ ดังตารางที่ 3.6

ตารางที่ 3.6 แสดงคลังข้อมูลฝึกสอน คือร่าย, กลอน, นิทานแก้ววิเศษ และบทความไอที เฉพาะชุดของตัวอักขระก่อนหน้า "า" และ "น" อันดับที่ 1

ชุดของตัวอักขระก่อนหน้า	ตัวอักขระปัจจุบัน	ความถี่	ชุดของตัวอักขระก่อนหน้า	ตัวอักขระปัจจุบัน	ความถี่
า	แ	5	น	พ	2
	ว	13		ส	13
	ม	48		ไ	6
	ง	45		น	7
	ล	6		บ	11
	อ	1		ร	11
	ญ	2		ท	11
	ะ	9		แ	13
	ร	87		เ	23
	ท	8		ห	7
	น	40		ธ	1
	ษ	6		ำ	9
	ต	11		ผ	2
	ค	7		า	18
	ก	33		จ	3
	ศ	2		ย	8
	ช	13		ต	10
	ถ	2		อ	10
	บ	7		ม	4
	พ	8		โ	6

ตารางที่ 3.6 (ต่อ) แสดงคลังข้อมูลฝึกสอน คือร่าย, กลอน, นิทานแก้ววิเศษ และบทความไอทีเฉพาะชุดของตัวอักขระก่อนหน้า "า" และ "น" อันดับที่ 1

ชุดของตัวอักขระก่อนหน้า	ตัวอักขระปัจจุบัน	ความถี่	ชุดของตัวอักขระก่อนหน้า	ตัวอักขระปัจจุบัน	ความถี่
า	ด	4	น	ว	6
	ป	7		ก	23
	ไ	6		ค	2
	โ	2		ๆ	2
	บ	4		ะ	1
	ใ	8		)	1
	ฝ	1		ฟ	1
	ESC	27		ESC	27

ตารางที่ 3.6 พบว่า จำนวนความถี่ของตัวอักขระที่เกิดตามชุดของตัวอักขระก่อนหน้า "า" และ "น" ส่วนใหญ่มีความถี่ต่ำ เนื่องจากลักษณะภาษาไทยมีการผสมพยัญชนะ, สระ และสัญลักษณ์ ทำให้ชุดของตัวอักขระก่อนหน้าตามด้วยตัวอักขระปัจจุบันมีความหลากหลาย ดังนั้นจึงเสนอแนวทางในการปรับปรุงวิธี PPM ให้เหมาะสมกับภาษาไทย เพื่อให้มีประสิทธิภาพการบีบอัดข้อมูลที่สูงขึ้น โดยลดการนับตัวอักขระปัจจุบันที่ไม่ซ้ำกัน ทำให้ความน่าจะเป็นของ P(ESC) ลดลง และส่งผลให้ความน่าจะเป็นของ P(i) สูงขึ้นด้วย ดังเช่นในวิธี PPMD มีการนับจำนวนอักขระตัวปัจจุบันที่ไม่ซ้ำกันลดลงจากวิธี PPMC ครึ่งหนึ่ง ส่งผลให้วิธี PPMD มีประสิทธิภาพการบีบอัดข้อมูลที่สูงกว่าวิธี PPMC เมื่อไม่พบปัญหา Zero-frequency เนื่องจาก P(i) มีค่าสูงกว่า ในทางกลับกันวิธี PPMC ให้ประสิทธิภาพการบีบอัดข้อมูลที่สูงกว่าวิธี PPMD เมื่อพบปัญหา Zero-frequency เนื่องจาก P(ESC) มีค่าสูงกว่า

จากแนวคิดข้างต้นเราจึงลดจำนวนตัวอักขระตัวปัจจุบันที่ไม่ซ้ำกันตามที่เกิดจริงในข้อความลงครึ่งหนึ่ง ถ้าจำนวนตัวอักขระตัวปัจจุบันที่ไม่ซ้ำกันเป็นจำนวนจริงให้ปัดเศษขึ้นเป็นจำนวนเต็ม เพื่อให้จำนวนอักขระตัวปัจจุบันที่ไม่ซ้ำกันอยู่ระหว่างวิธี PPMC และวิธี PPMD ทำให้ได้ประสิทธิภาพในการบีบอัดข้อมูลที่ดีทั้งสองกรณี คือเมื่อพบและไม่พบปัญหา Zero-frequency ส่วนการนับความถี่ของตัวอักขระใช้พื้นฐานความน่าจะเป็นของตัวอักขระที่เกิดตามจริงในข้อความ เพื่อให้ตัวอักขระมีโอกาสเกิดตามจริง โดยตัวอักขระที่มีความถี่สูงจะได้กระแสข้อมูลบิตสั้น และตัวอักขระที่มีความถี่ต่ำจะให้กระแสข้อมูลบิตที่ยาว โดยการคำนวณความน่าจะเป็นของ P(i) และ P(ESC) แสดงดังตารางที่ 3.7

ตารางที่ 3.7 แสดงสมการวิธีที่ปรับปรุง

ค่าความน่าจะเป็น	วิธีที่ปรับปรุง
P(i)	$\frac{Ci}{N + \lceil d/2 \rceil}$
P(ESC)	$\frac{\lceil d/2 \rceil}{N + \lceil d/2 \rceil}$

ตารางที่ 3.7 เรียกวิธีนี้ว่า Prediction by Partial Matching method Hybrid (PPMH) โดย P(i) คือค่าความน่าจะเป็นชุดของตัวอักขระก่อนหน้าตามด้วยตัวอักขระปัจจุบันตำแหน่งที่ i ในอันดับที่กำหนด และ P(ESC) คือค่าความน่าจะเป็นชุดของตัวอักขระก่อนหน้าตามด้วยตัวอักขระปัจจุบันที่ไม่พบในอันดับที่กำหนด โดย C แทนค่าความถี่ของตัวอักขระ, d แทนจำนวนตัวอักขระปัจจุบันที่ไม่ซ้ำกัน และ N แทนผลรวมความถี่ทั้งหมดของตัวอักขระ

ตารางที่ 3.8 แสดงการคำนวณความน่าจะเป็น P(i) และ P(ESC) ของวิธี PPMH อันดับที่ 0-1 จากข้อความ "กกขกขกด"

วิธี PPMH

อันดับที่ 0				อันดับที่ 1			
ชุดของตัวอักขระก่อนหน้า	ตัวอักขระปัจจุบัน	ความถี่	การคำนวณความน่าจะเป็น	ชุดของตัวอักขระก่อนหน้า	ตัวอักขระปัจจุบัน	ความถี่	การคำนวณความน่าจะเป็น
-	ก	4	4/9 = 0.4444	ก	ก	1	1/6 = 0.1667
-	ข	2	2/9 = 0.2222	ก	ข	2	2/6 = 0.3333
-	ค	1	1/9 = 0.1111	ก	ค	1	1/6 = 0.1667
-	ESC	2	2/9 = 0.2222	ก	ESC	2	2/6 = 0.3333
				ข	ก	2	2/3 = 0.6667
				ข	ESC	1	1/3 = 0.3333

3.2.2 การปรับปรุงเพิ่มเติมโดยใช้การแยกพยัญชนะ, สระ และสัญลักษณ์

เนื่องจากลักษณะภาษาไทยมีการผสมกันระหว่างพยัญชนะ, สระ และสัญลักษณ์ มีความหลากหลายของชุดของตัวอักขระก่อนหน้าทำให้ตัวอักขระปัจจุบันมีความถี่ต่ำ ดังแสดงในตารางที่ 3.6 ในขณะเดียวกันภาษาไทยมีลักษณะของพยัญชนะประสม ดังนั้นพยัญชนะน่าจะ

ติดกัน จึงได้เสนอการประมวลผลข้อมูลภาษาไทยก่อนการเข้ารหัส โดยมีการทดลองกับรูปแบบการประมวลผลข้อมูลเบื้องต้น ดังนี้

1) แบบภาษาไทยปกติ (แทนด้วย NORMAL) เช่น "ประวัติศาสตร์"
2) แบบแยกพยัญชนะออกจากสระและสัญลักษณ์ โดยแทนที่พยัญชนะด้วย \a (Escape sequence) เพื่อประโยชน์ในการย้อนกลับเป็นภาษาไทยปกติ (แทนด้วย DIVIDE) เช่น "ปรวดศสดร00ะ0ิ0า000" ในตัวอย่างนี้ขอใช้สัญลักษณ์ 0 แทน \a เพื่อให้เห็นตัวอย่างที่ชัดเจนขึ้น
3) แบบแยกพยัญชนะออกจากสระและสัญลักษณ์ โดยเปลี่ยนรูปสระและสัญลักษณ์เป็นรหัสฮัฟแมน โดยพิจารณาอักขระ 1 และ 0 เป็นตัวอักขระ (แสดงในตารางที่ 3.9) (แทนด้วย DIVIDE_HUF_CHR) เช่น "ปรวดศสดร01101110111010100 10111111101011101101110111110010" เพื่อให้มีความถี่ของอักขระ 1 หรือ 0 ที่สูงขึ้น และมีตัวอักขระปัจจุบันที่ไม่ซ้ำกันลดจำนวนลง
4) แบบแยกพยัญชนะออกจากสระและสัญลักษณ์ โดยเปลี่ยนรูปสระและสัญลักษณ์เป็นรหัสฮัฟแมน และพิจารณา 0 และ 1 เป็นบิต (แทนด้วย DIVIDE_HUF_BIT) ทำให้ได้สายอักขระของสระและสัญลักษณ์ที่หลากหลาย โดยมีตัวอักขระปัจจุบันที่ไม่ซ้ำกันเพิ่มขึ้น

ตารางที่ 3.9 แสดงรหัสฮัฟแมน

สระ-สัญลักษณ์	รหัสฮัฟแมน	สระ-สัญลักษณ์	รหัสฮัฟแมน	สระ-สัญลักษณ์	รหัสฮัฟแมน
\a	101	สระอู	000110	เลขห้า	111101001010
สระอา	011	สระอำ	000011	เลขหก	111101000001
ช่องว่าง	1100	ไม้ไต่คู้	000010	เลขแปด	111101000000
ไม้เอก	0101	ไม้โอ	1111011	เลขสาม	1111010010111
ไม้หันอากาศ	0100	สระอี	1110011	เลขเจ็ด	1111010001000
ไม้โท	0011	การันต์	1110010	ไม้จัตวา	11110100101101
สระเอ	0010	มหัพภาค	111101011	ไม้ตรี	11110100101100
สระอิ	11111	ไม้ยมก	1111010011	ปรัศนี	11110100010111
สระอี	11101	ฟันหนู	11110101001	เซมิโคลอน	11110100010110
สระอะ	11011	เลขเก้า	11110101000	โคลอน	11110100010101
สระแอ	00010	นขลิขิตปิด	11110100100	ยติภังค์	11110100010100
สระอือ	00000	นขลิขิตเปิด	11110100011	จุลภาค	11110100010011
ไม้ม้วน	111100	เลขหนึ่ง	11110100001	อัศเจรีย์	11110100010010
สระอุ	111000	เลขสี่	111101010111		
ไม้มลาย	110101	เลขสอง	111101010110		
\r	110100	เลขศูนย์	111101010101		
\n	000111	ไปยาลน้อย	111101010100		

ตารางที่ 3.10 แสดงรูปแบบ NORMAL, รูปแบบ DIVIDE และรูปแบบ DIVIDE_HUF_CHR จากข้อความ "ประวัติศาสตร์ไทย"

รูปแบบ NORMAL	รูปแบบ DIVIDE	รูปแบบ DIVIDE_HUF_CHR
-	ปรวตศสตรทย	ปรวตศสตรทย
ป	\a	101
ร	\a	101
สระอะ	สระอะ	11011
ว	\a	101
ไม้หันอากาศ	ไม้หันอากาศ	0100
ต	\a	101
สระอิ	สระอิ	11111
ศ	\a	101
สระอา	สระอา	011
ส	\a	101
ต	\a	101
ร	\a	101
การันต์	การันต์	1110010
ไ	ไ	110101
ท	\a	101
ย	\a	101

ตารางที่ 3.11 แสดงสายอักขระของการประมวลผลข้อมูลเบื้องต้นในรูปแบบ DIVIDE_HUF_CHR และรูปแบบ DIVIDE_HUF_BIT ข้อความ "ปรวตศสตรทย 1011011101110101001011111101011101101111001011010110110 1" ในตารางที่ 3.10

	รูปแบบ DIVIDE_HUF_CHR	รูปแบบ DIVIDE_HUF_BIT
พยัญชนะ	ปรวตศสตรทย	ปรวตศสตรทย
สระและสัญลักษณ์	10110111	ท
	01110101	ย
	00101111	/
	11101011	-
	10110110	ถ
	11110010	๒
	11010110	สระอี
	1101	\r

ตารางที่ 3.11 นำรูปแบบ DIVIDE_HUF_CHR และรูปแบบ DIVIDE_HUF_BIT ไปสร้างคลังข้อมูลฝึกสอนได้ดังตารางที่ 3.12 และ 3.13 ตามลำดับ

ตารางที่ 3.12 แสดงคลังข้อมูลฝึกสอน อันดับที่ 0-2 จากรูปแบบ DIVIDE_HUF_CHR ในตารางที่ 3.11

อันดับที่ 0		อันดับที่ 1			อันดับที่ 2		
ตัวอักขระปัจจุบัน	ความถี่	ชุดของตัวอักขระก่อนหน้า	ตัวอักขระปัจจุบัน	ความถี่	ชุดของตัวอักขระก่อนหน้า	ตัวอักขระปัจจุบัน	ความถี่
ป	1	ป	ร	1	ปร	ว	1
ร	2		ESC	1		ESC	1
ว	1	ร	ว	1	รว	ต	1
ต	2		ท	1		ESC	1
ค	1		ESC	2	วต	ค	1
ส	1	ว	ต	1		ESC	1
ท	1		ESC	1	ตค	ส	1
ย	1	ต	ค	1		ESC	1
1	40		ร	1	คส	ค	1
0	20		ESC	2		ESC	1
ESC	10	ค	ส	1	สต	ร	1
			ESC	1		ESC	1
		ส	ต	1	ตร	ท	1
			ESC	1		ESC	1
		ท	ย	1	รท	ย	1
			ESC	1		ESC	1
		ย	1	1	ทย	1	1
			ESC	1		ESC	1
		1	0	18	ย1	0	1
			1	21		ESC	1
			ESC	2	10	1	16
		0	1	18		0	2
			0	2		ESC	2
			ESC	2	01	1	11
						0	6
						ESC	2
					11	0	11
						1	10
						ESC	2
					00	1	2
						ESC	1

ตารางที่ 3.13 แสดงคลังข้อมูลฝึกสอน อันดับที่ 0-2 จากรูปแบบ DIVIDE_HUF_BIT ในตารางที่ 3.11

อันดับที่ 0		อันดับที่ 1			อันดับที่ 2		
ตัวอักขระปัจจุบัน	ความถี่	ชุดของตัวอักขระก่อนหน้า	ตัวอักขระปัจจุบัน	ความถี่	ชุดของตัวอักขระก่อนหน้า	ตัวอักขระปัจจุบัน	ความถี่
ป	1	ป	ร	1	ปร	ว	1
ร	2		ESC	1		ESC	1
ว	1	ร	ว	1	รว	ต	1
ต	2		ท	1		ESC	1
ค	1		ESC	2	วต	ค	1
ส	1	ว	ต	1		ESC	1
ท	2		ESC	1	ตค	ส	1
ย	1	ต	ค	1		ESC	1
น	1		ร	1	คส	ต	1
/	1		ESC	2		ESC	1
่	1	ค	ส	1	สต	ร	1
ถ	1		ESC	1		ESC	1
๒	1	ส	ต	1	ตร	ท	1
สระอี	1		ESC	1		ESC	1
\r	1	ท	ย	1	รท	ย	1
ESC	15		น	1		ESC	1
			ESC	2	ทย	ท	1
		ย	ท	1		ESC	1
			ESC	1	ยท	น	1
		น	/	1		ESC	1
			ESC	1	ทน	/	1
		/	่	1		ESC	1
			ESC	1	น/	่	1
		่	ถ	1		ESC	1
			ESC	1	/่	ถ	1
		๒	สระอี	1		ESC	1
			ESC	1	่ถ	๒	1
		สระอี	\r	1		ESC	1
			ESC	1	ถ๒	สระอี	1
						ESC	1
					๒สระอี	\r	1
						ESC	1

ตารางที่ 3.10-3.13 พบว่า รูปแบบ DIVIDE_HUF_CHR มีความถี่ของตัวอักขระ 0, 1 สูง ซึ่งส่งผลต่อประสิทธิภาพในการบีบอัดข้อมูลที่ดีขึ้น และรูปแบบ DIVIDE_HUF_BIT มีตัวอักขระในแต่ละอันดับมีความถี่ต่ำ และมีจำนวนตัวอักขระปัจจุบันที่ไม่ซ้ำกันในข้อความสูง โดยสายอักขระของสระและสัญลักษณ์ คือ "ทน/ถ๊สระอึ\๙" ซึ่งเป็นสายอักขระที่ไม่พบบ่อยในข้อความภาษาไทย ทำให้ชุดของตัวอักขระก่อนหน้าตามด้วยตัวอักขระปัจจุบันมีความถี่ต่ำ จึงส่งผลต่อประสิทธิภาพในการบีบอัดข้อมูลให้ต่ำลงด้วย

ดังนั้นการศึกษาขั้นตอนต่อไปจะทำการทดลองการประมวลผลข้อมูลเบื้องต้น 3 รูปแบบ คือ รูปแบบที่ 1 NORMAL, รูปแบบที่ 2 DIVIDE และรูปแบบที่ 3 DIVIDE_HUF_CHR ตามลำดับ

3.3 การออกแบบโครงสร้างและการทำงาน

หัวข้อนี้ กล่าวถึงรายละเอียดในการออกแบบโครงสร้างและการทำงานของระบบ โดยมีรายละเอียดดังนี้

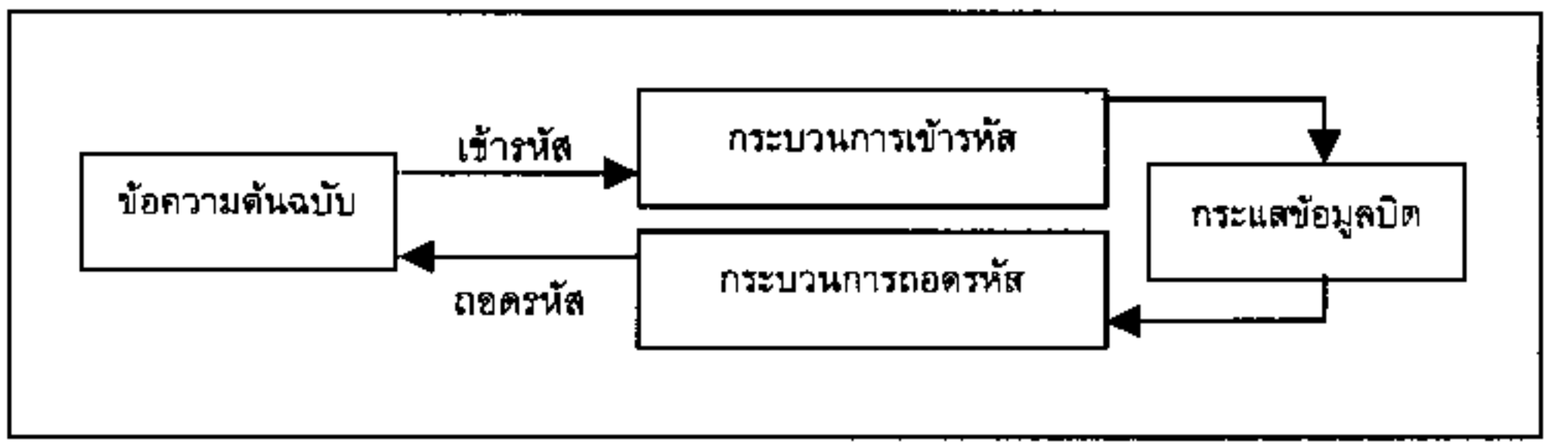
3.3.1 โครงสร้างของระบบ

ประกอบด้วยกระบวนการที่สำคัญในการเข้ารหัส-ถอดรหัส ดังนี้

- **การสร้างตัวแบบ** คือการวิเคราะห์ข้อมูลจากคลังข้อมูลฝึกสอน เพื่อสร้างตารางชุดข้อมูลสถิติ เพื่อนำไปใช้อ้างอิงในการเข้ารหัส-ถอดรหัส
- **กระบวนการเข้ารหัส** คือการนำคลังข้อมูลทดสอบไปประมวลผลตามอัลกอริทึมการเข้ารหัส โดยอาศัยตัวแบบที่สร้างขึ้น เพื่อให้ได้ผลลัพธ์เป็นกระแสข้อมูลบิต
- **กระบวนการถอดรหัส** คือการนำกระแสข้อมูลบิตที่ได้จากกระบวนการเข้ารหัสมาผ่านอัลกอริทึมการถอดรหัส โดยอาศัยตัวแบบที่สร้างขึ้นเพื่อให้ได้ผลลัพธ์ คือ คลังข้อมูลที่ถูกเข้ารหัส

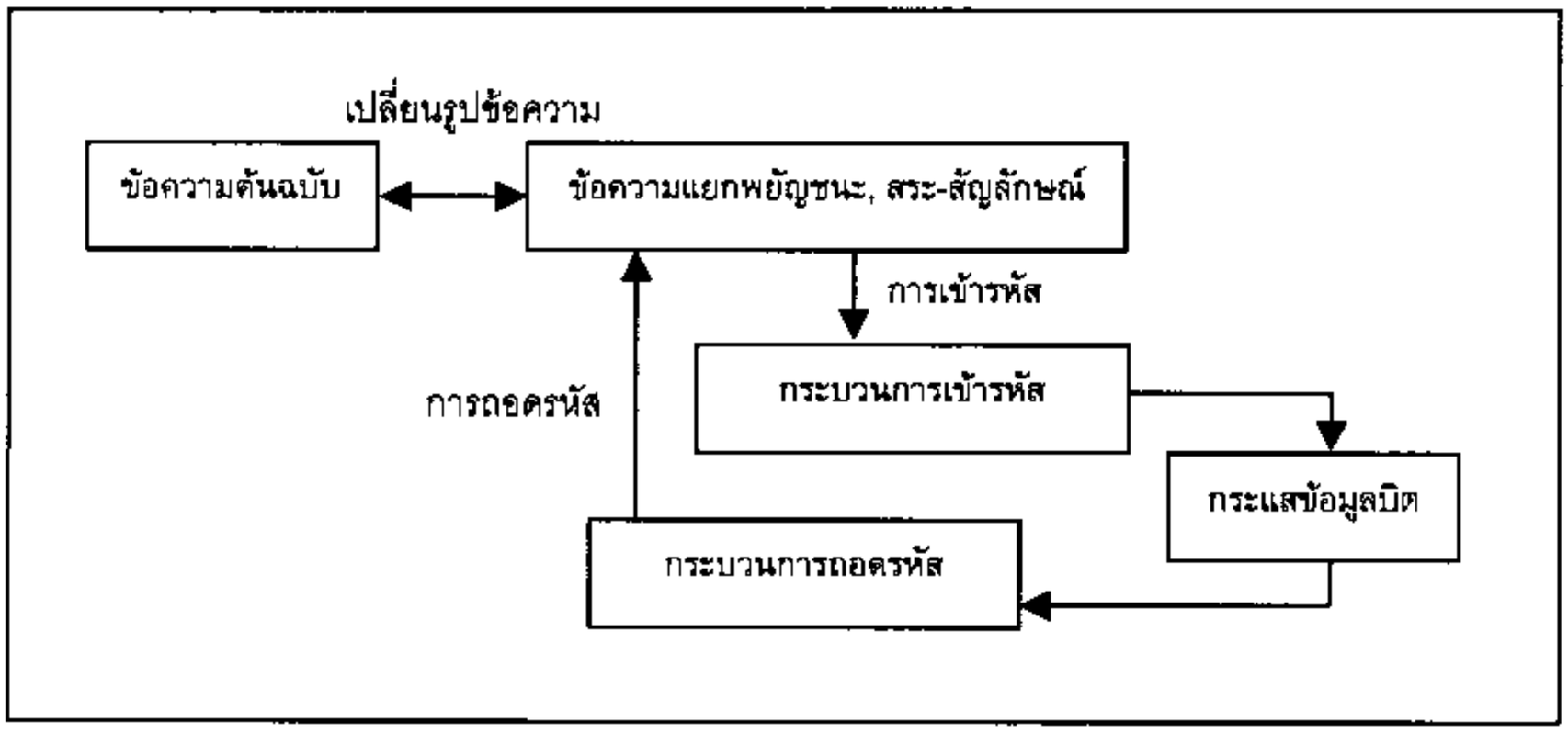
3.3.2 การทำงานของระบบ

จากตาราง 3.1 ตัวอักขระภาษาไทยแบ่งออกเป็น 4 ระดับ ดังนั้นเมื่อนำเข้าคอมพิวเตอร์ต้องประมวลผลให้อยู่ในรูป 1 ระดับ ทำให้เกิดการผสมระหว่างพยัญชนะ, สระและสัญลักษณ์ ดังนั้นหากนำข้อความเปลี่ยนรูปใหม่ โดยแยกพยัญชนะ และสระ-สัญลักษณ์ ทำให้เกิดความถี่สูงขึ้น และสอดคล้องกับหลักการของ PPM ซึ่งความถี่ตัวอักขระสูงจะส่งผลให้จำนวนกระแสข้อมูลบิตต่ำ โครงสร้างการทำงานของระบบขึ้นอยู่กับลักษณะการประมวลผลสายอักขระเบื้องต้น ได้แก่ รูปแบบ NORMAL, รูปแบบ DIVIDE และรูปแบบ DIVIDE_HUF_CHR แสดงดังภาพที่ 3.1, 3.2 และ 3.3 ตามลำดับ

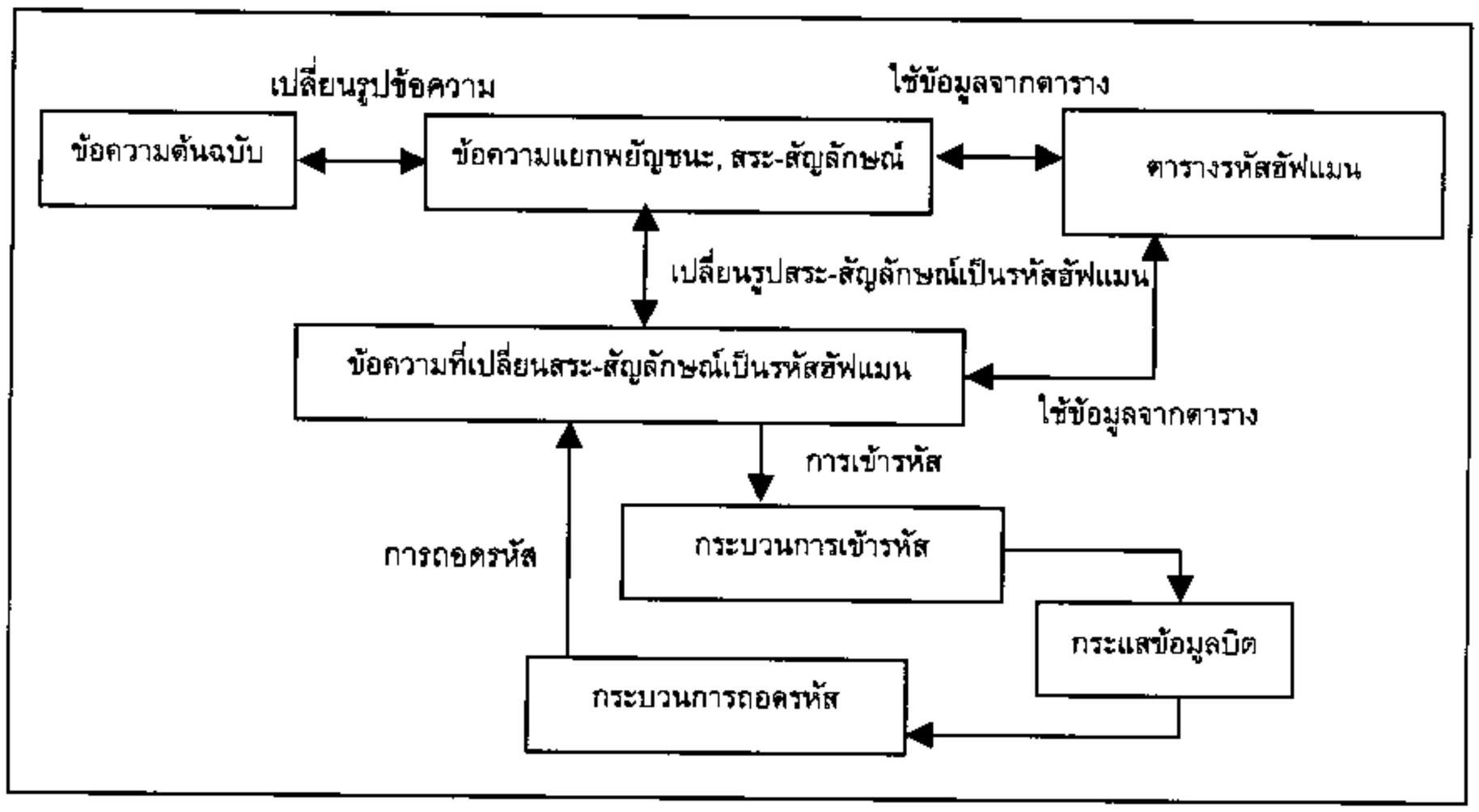


ภาพที่ 3.1 แสดงการทำงานของการเข้ารหัส-ถอดรหัส ของการประมวลผลข้อมูลเบื้องต้นใน รูปแบบ NORMAL

หมายเหตุ: ข้อความต้นฉบับ คือข้อความที่ต้องการเข้ารหัส



ภาพที่ 3.2 แสดงการทำงานของการเข้ารหัส-ถอดรหัส ของการประมวลผลข้อมลเบื้องต้นใน รูปแบบ DIVIDE



ภาพที่ 3.3 แสดงการทำงานของการเข้ารหัส-ถอดรหัส ของการประมวลผลข้อมูลเบื้องต้นในรูปแบบ DIVIDE_HUF_CHR

บทที่ 4

ผลการทดลอง

4.1 รายละเอียดการทดลอง

ในบทนี้จะทำการทดลองเพื่อเปรียบเทียบประสิทธิภาพในการบีบอัดข้อมูล และโดยใช้ข้อมูลทดสอบข้อมูลดังรายละเอียดในบทที่ 3 และกำหนดตัวแปรคงที่ (Fixed variables) ที่ใช้ในการทดลอง คือ จำนวนบิตที่ใช้เข้ารหัส-ถอดรหัสเท่ากับ 16 บิต

การวัดประสิทธิภาพประกอบด้วย การวัดอัตราการบีบอัดข้อมูล และเวลาที่ใช้ในการเข้ารหัส-ถอดรหัสของ PPM โดยเวลาในการเข้ารหัส ประกอบด้วย เวลาประมวลผลข้อมูลภาษาไทยเบื้องต้น ดังรายละเอียดในบทที่ 3 และเวลาที่เข้ารหัสได้ผลลัพธ์เป็นกระแสข้อมูลบิต ส่วนเวลาในการถอดรหัส ประกอบด้วย เวลาที่เปลี่ยนรูปจากกระแสข้อมูลบิตเป็นรูปแบบภาษาไทยที่ผ่านการประมวลผลเบื้องต้น และเวลาในการถอดรหัสข้อมูลภาษาไทยเบื้องต้นเป็นรูปแบบภาษาไทยปกติ โดยอัตราการบีบอัดข้อมูลคำนวณได้จาก

$$\text{อัตราการบีบอัดข้อมูล (\%)} = \left(1 - \left(\frac{\text{ขนาดข้อมูลหลังการบีบอัด (ไบต์)}}{\text{ขนาดข้อมูลก่อนการบีบอัด (ไบต์)}}\right)\right) \times 100$$

4.1.1 ขั้นตอนการทดลอง

1) การทดลองที่ 1 คือการวัดประสิทธิภาพการบีบอัดข้อมูล ประกอบด้วย การวัดอัตราการบีบอัดข้อมูล และเวลาที่ใช้ในการเข้ารหัส-ถอดรหัส โดยเปรียบเทียบวิธี PPMA, วิธี PPMB, วิธี PPMC, วิธี PPMD และวิธี PPMH อันดับที่ 0-5 โดยทดลองกับการประมวลผลข้อมูลเบื้องต้นในทั้ง 3 รูปแบบ (NORMAL, DIVIDE และ DIVIDE_HUF_CHR) รายละเอียดดังแสดงในบทที่ 3 โดยแบ่งกรณีทดลองเป็น 2 กรณี คือ กรณีที่หนึ่งเมื่อคลังข้อมูลทดสอบเป็นสับเซตของคลังข้อมูลฝึกสอน และกรณีที่สองเมื่อคลังข้อมูลทดสอบไม่เป็นสับเซตของคลังข้อมูลฝึกสอน

2) การทดลองที่ 2 คือการวัดประสิทธิภาพการบีบอัดข้อมูล และเวลาที่ใช้ในการเข้ารหัส-ถอดรหัส เปรียบเทียบวิธี PPMH ในทั้ง 3 รูปแบบ, วิธีรหัสฮัฟแมน และสารสนเทศเฉลี่ย

4.2 การทดลองและผลการทดลอง

ตารางที่ 4.1 แสดงรายละเอียดขนาดข้อมูลของข้อความภาษาไทย 3 รูปแบบ

ตารางที่ 4.1 แสดงขนาดของข้อมลที่ถกประมวลผลเบื้องต้นใน 3 รปแบบ

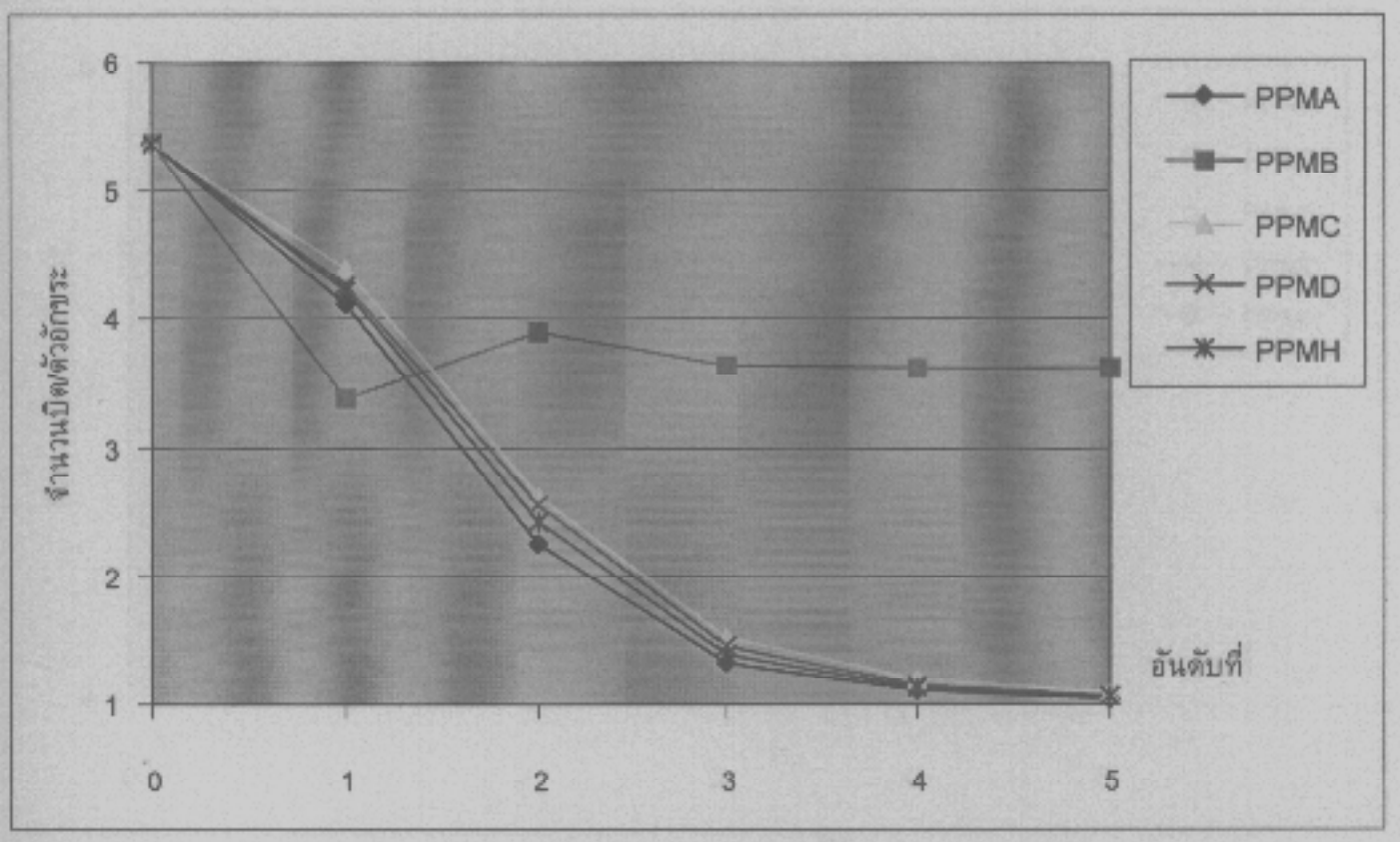
ข้อมูลทดสอบ ลำดับที่	เนื้อหา	ขนาดข้อมูล (ไบต์)		
		NORMAL	DIVIDE	DIVIDE_HUF_CHR
1	ร่าย	584	918	2,468
2	กลอน	887	1,363	3,782
3	นิทานแก้ววิเศษ	2,072	3,168	8,657
4	บทความไอที	3,562	5,633	15,520
5	โครงการคอมพิวเตอร์ไอซีที	4,466	7,024	19,151
6	ประวัติศาสตร์ไทยราชวงศ์ไทย	8,278	13,189	35,134
7	สามก๊ก	9,314	14,708	40,871
8	รวมนิทานพื้นบ้าน	9,997	15,467	42,632
9	มงคลชีวิต	18,244	28,308	78,649
10	ไม้ประดับ	23,688	36,721	102,341
	ค่าเฉลี่ย	8,109	12,650	34,921

จากตารางที่ 4.1 การประมวลผลข้อมูลภาษาไทยเบื้องต้นในรูปแบบ NORMAL, DIVIDE และ DIVIDE_HUF_CHR มีผลต่อขนาดของแฟ้มข้อมูล และประสิทธิภาพในการบีบอัดข้อมูล โดยการทดลองที่ 1 และ 2 ใช้คลังข้อมูลฝึกสอน คือ ร่าย, กลอน, นิทานแก้ววิเศษ และบทความไอที, คลังข้อมูลทดสอบ คือ ร่าย, กลอน, นิทานแก้ววิเศษ, บทความไอที, โครงการคอมพิวเตอร์ไอซีที, ประวัติศาสตร์ไทยราชวงศ์ไทย, สามก๊ก, รวมนิทานพื้นบ้าน, มงคลชีวิต และไม้ประดับ

4.2.1 การทดลองที่ 1 ค่าเฉลี่ยจำนวนบิต/ตัวอักขระ และเวลาที่ใช้เข้ารหัส-ถอดรหัสของ วิธี PPMA, วิธี PPMB, วิธี PPMC, วิธี PPMD และวิธี PPMH

1) การทดลองการประมวลผลข้อมูลภาษาไทยเบื้องต้นในรูปแบบ NORMAL

(1.1) เมื่อคลังข้อมูลทดสอบเป็นสับเซตของคลังข้อมูลฝึกสอน แสดงดังกราฟที่ 4.1

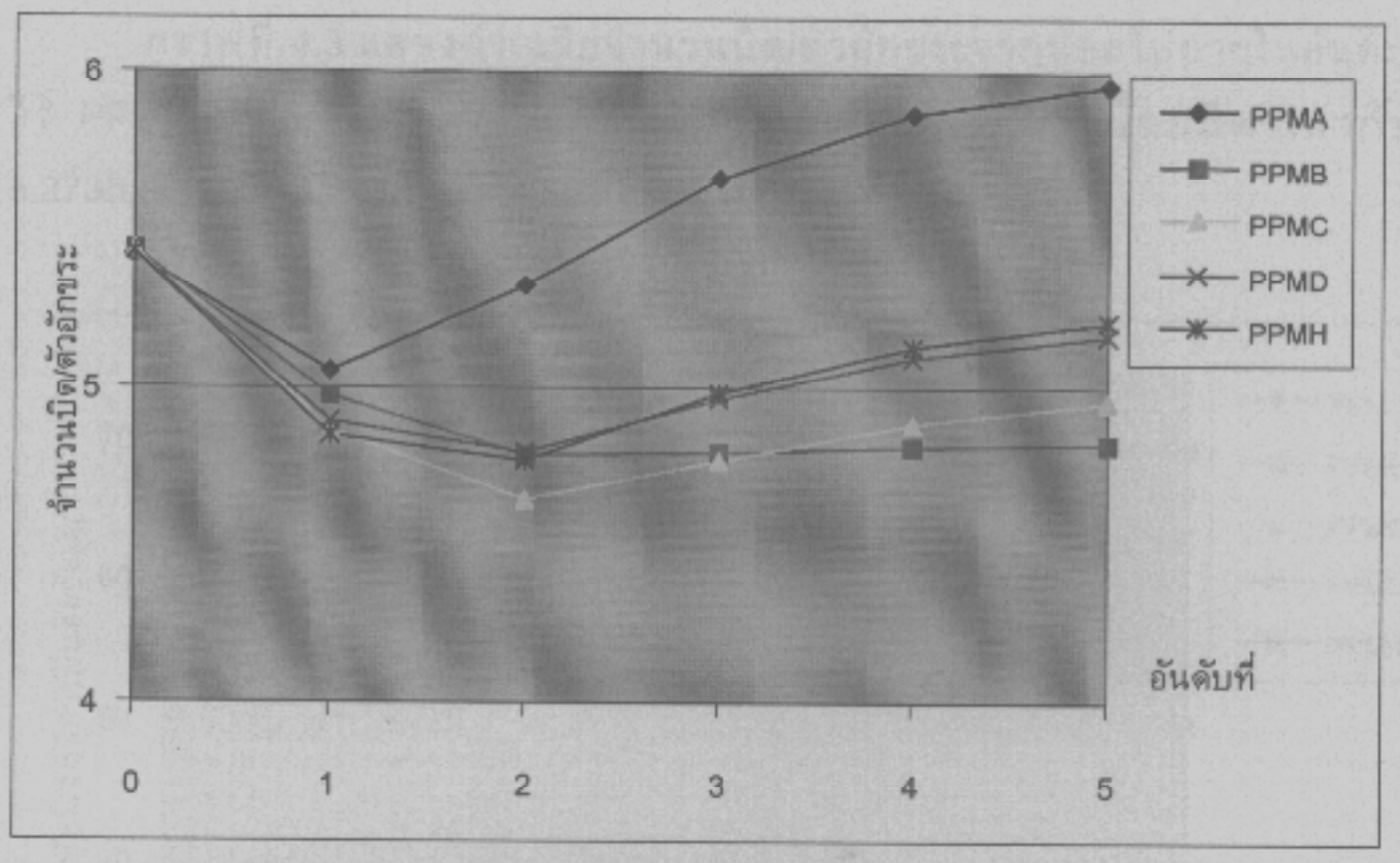


กราฟที่ 4.1 แสดงค่าเฉลี่ยจำนวนบิต/ตัวอักขระของข้อมูลทดสอบลำดับที่ 1-4 อันดับที่ 0-5 รูปแบบ NORMAL

กราฟที่ 4.1 แสดงค่าเฉลี่ยจำนวนบิต/ตัวอักขระจากน้อยไปมากในอันดับที่ 0-5 ดังนี้ วิธี PPMA, วิธี PPMH, วิธี PPMD และวิธี PPMC และเรียงลำดับค่าเฉลี่ยจำนวนบิต/ตัวอักขระจากน้อยไปมากในอันดับที่ 2 เพราะเป็นอันดับที่ให้ค่าเฉลี่ยจำนวนบิต/ตัวอักขระต่ำกว่าอันดับที่ 1 และเวลาที่ใช้การเข้ารหัส-ถอดรหัสไม่สูงเกินไป (ต่ำกว่าตั้งแต่อันดับ 3 เป็นต้นไป) นอกจากนั้นแนวโน้มของประสิทธิภาพจะไม่ต่างไปจากอันดับที่สูงขึ้น ดังนี้ วิธี PPMA, วิธี PPMH, วิธี PPMD, วิธี PPMC และวิธี PPMB โดยมีค่าเท่ากับ 2.2511, 2.4209, 2.5549, 2.6231 และ 3.9062 บิต ตามลำดับ

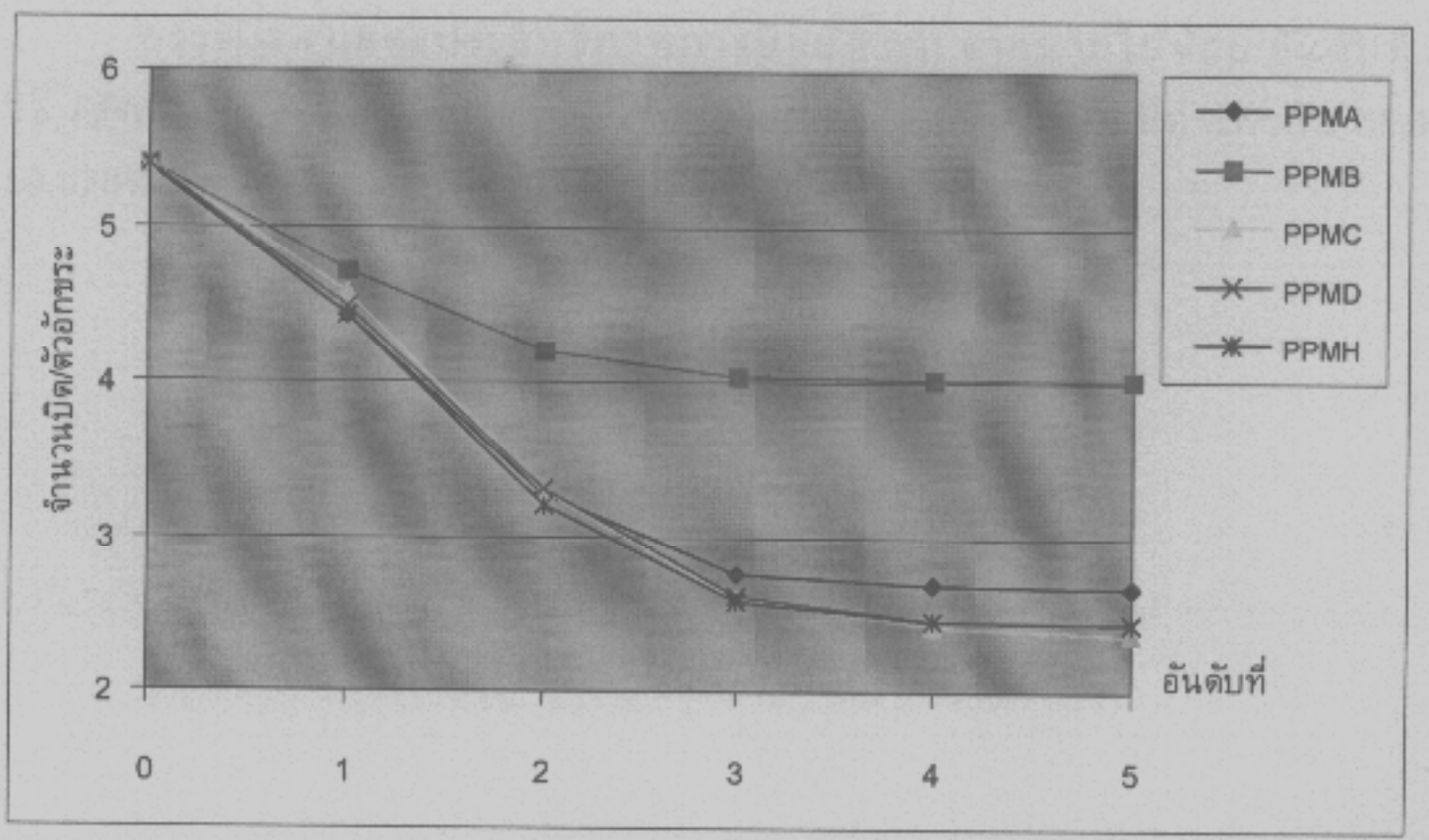
วิธี PPMB ให้ค่าเฉลี่ยจำนวนบิตน้อยที่สุด อันดับที่ 0-1 เท่านั้น โดยเมื่ออันดับมากกว่า 2 ขึ้นไป จะให้ค่าเฉลี่ยจำนวนบิตที่สูงกว่าวิธีอื่น ๆ เพราะวิธี PPMB นับค่าความถี่ให้ตัวอักขระจะเริ่มนับตัวอักขระที่เจอครั้งที่ 2 ดังนั้นตัวอักขระที่มีความถี่เท่ากับ 1 จะไม่มีโอกาสนับความถี่ที่เกิด ทำให้พบปัญหา Zero-frequency โดยลดไปทำอันดับที่ต่ำลงไป ส่งผลให้ได้กระแสข้อมูลบิตยาว ดังเช่น อันดับที่ 2 เป็นต้นไป ตัวอักขระไม่ซ้ำกันมีจำนวนมาก ส่งผลให้ตัวอักขระมีความถี่ต่ำ และถ้าความถี่เท่ากับ 1 ก็จะไม่มีโอกาสเกิดด้วย

(1.2) เมื่อคลังข้อมูลทดสอบไม่เป็นสับเซตของคลังข้อมูลฝึกสอน แสดงดังกราฟที่ 4.2



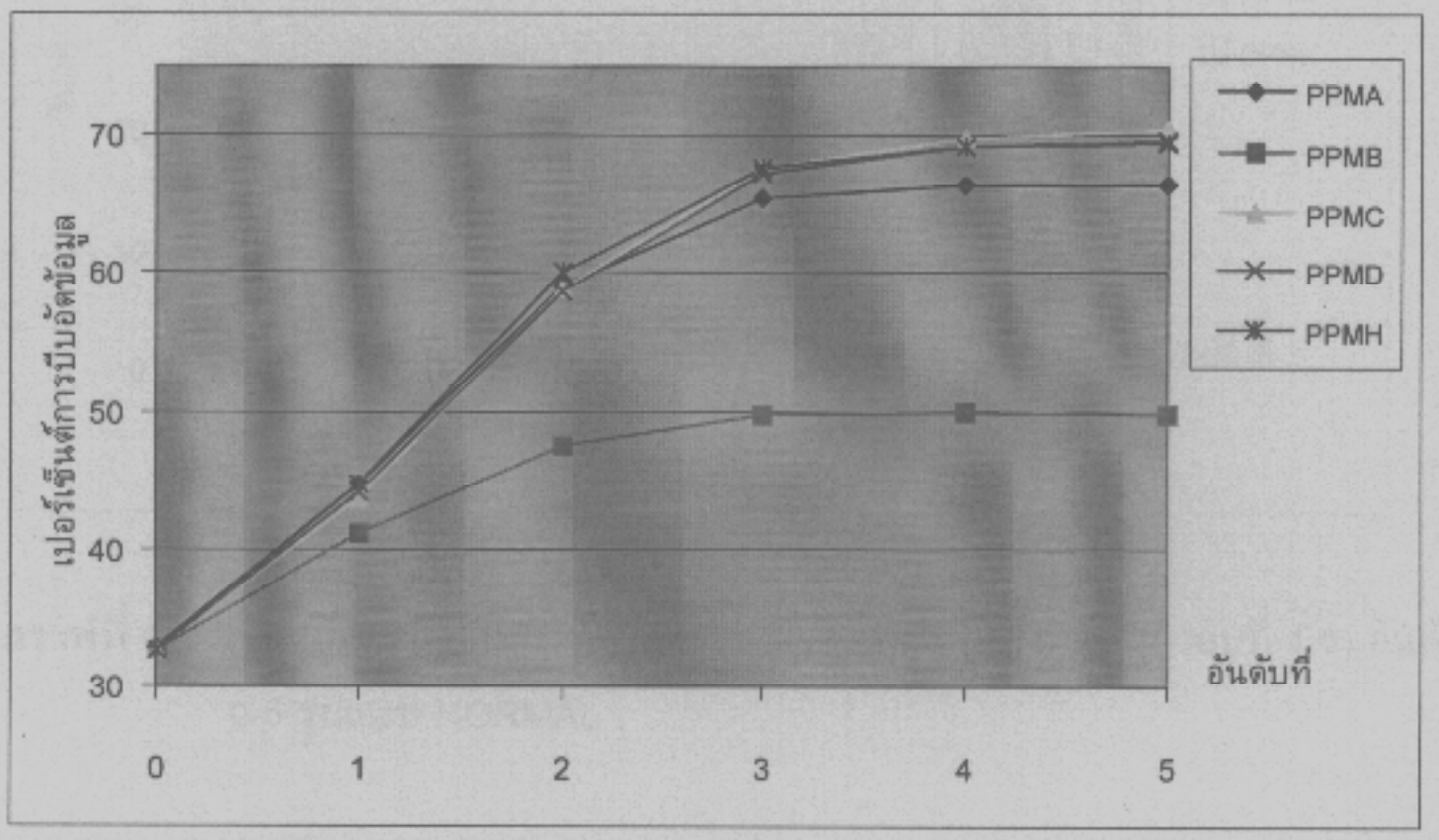
กราฟที่ 4.2 แสดงค่าเฉลี่ยจำนวนบิต/ตัวอักขระของข้อมูลทดสอบลำดับที่ 5-6 อันดับที่ 0-5 รูปแบบ NORMAL

กราฟที่ 4.2 แสดงค่าเฉลี่ยจำนวนบิต/ตัวอักขระจากน้อยไปมากในอันดับที่ 2 ดังนี้ วิธี PPMC, วิธี PPMH, วิธี PPMB, วิธี PPMD และวิธี PPMA โดยมีค่าเท่ากับ 4.6457, 4.7632, 4.7814, 4.7957 และ 5.3182 บิต ตามลำดับ



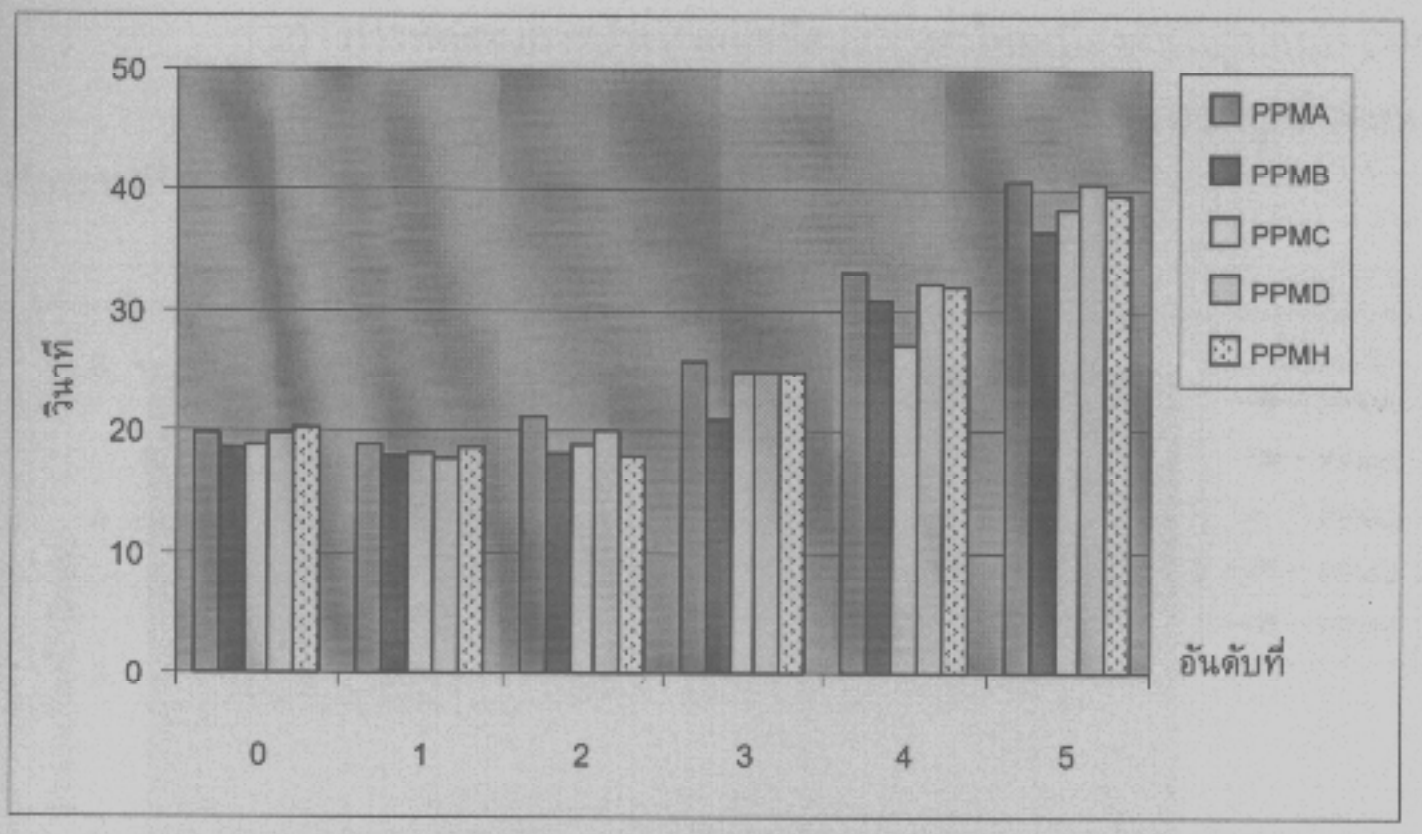
กราฟที่ 4.3 แสดงค่าเฉลี่ยจำนวนบิต/ตัวอักขระของข้อมูลทดสอบทั้งหมด (ลำดับที่ 1-6) อันดับที่ 0-5 รูปแบบ NORMAL

กราฟที่ 4.3 แสดงค่าเฉลี่ยจำนวนบิต/ตัวอักขระจากน้อยไปมากในอันดับที่ 2 ดังนี้ วิธี PPMH, วิธี PPMA, วิธี PPMC, วิธี PPMD และวิธี PPMB โดยมีค่าเท่ากับ 3.2017, 3.2735, 3.2973, 3.3018 และ 4.198 บิต ตามลำดับ

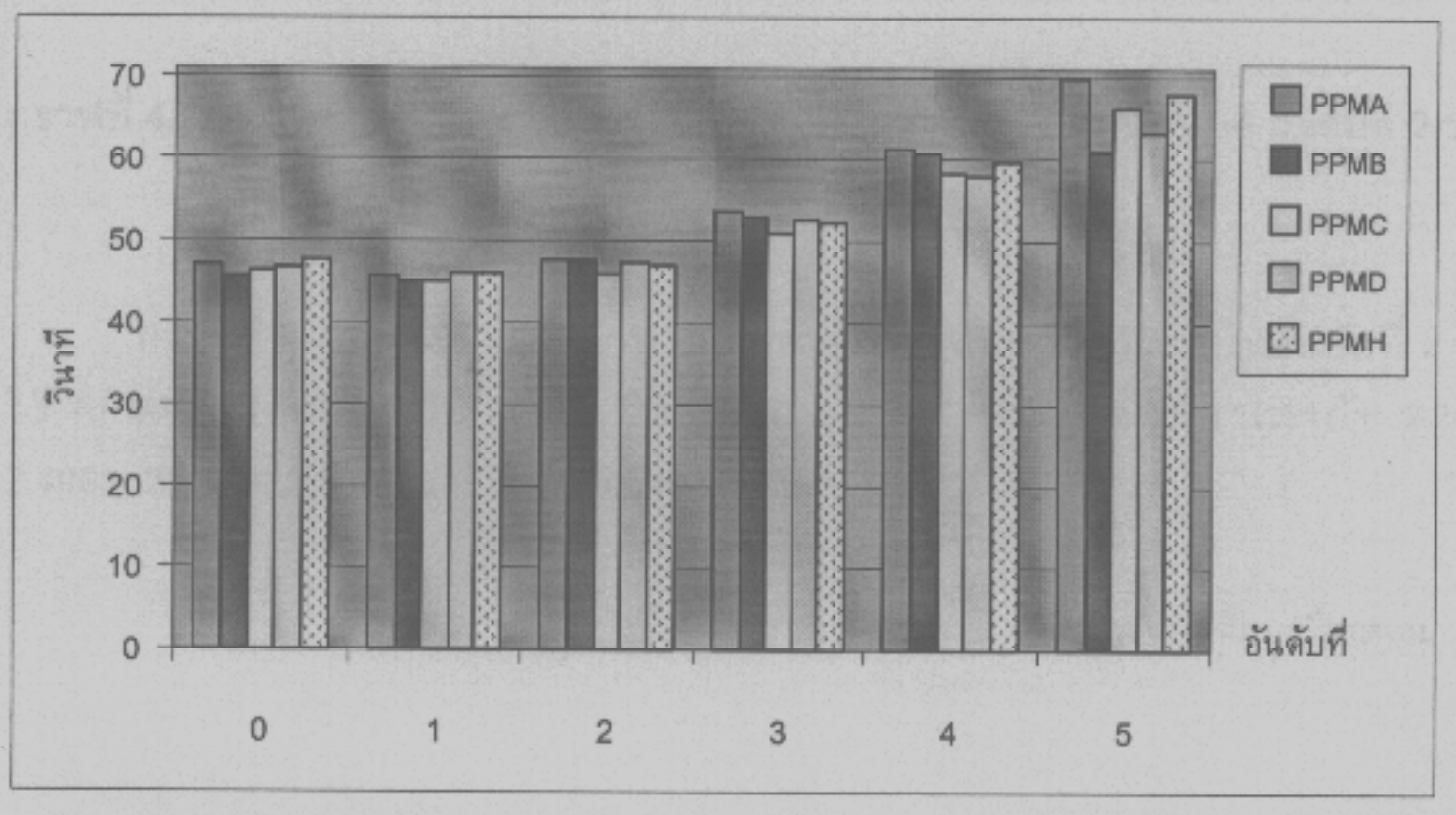


กราฟที่ 4.4 แสดงเปอร์เซ็นต์ การบีบอัดของข้อมูลทดสอบทั้งหมด (ลำดับที่ 1-6) อันดับที่ 0-5 รูปแบบ NORMAL

กราฟ 4.4 แสดงประสิทธิภาพการบีบอัด (%) จากมากไปน้อย อันดับที่ 2 ดังนี้ วิธี PPMH, วิธี PPMA, วิธี PPMC, วิธี PPMD และวิธี PPMB โดยมีค่าเท่ากับ 59.9791%, 59.078%, 58.7842%, 58.7273% และ 47.5257% ตามลำดับ



กราฟที่ 4.5 แสดงเวลาเฉลี่ยในการเข้ารหัสของข้อมูลทดสอบทั้งหมด (ลำดับที่ 1-6) อันดับที่ 0-5 รูปแบบ NORMAL

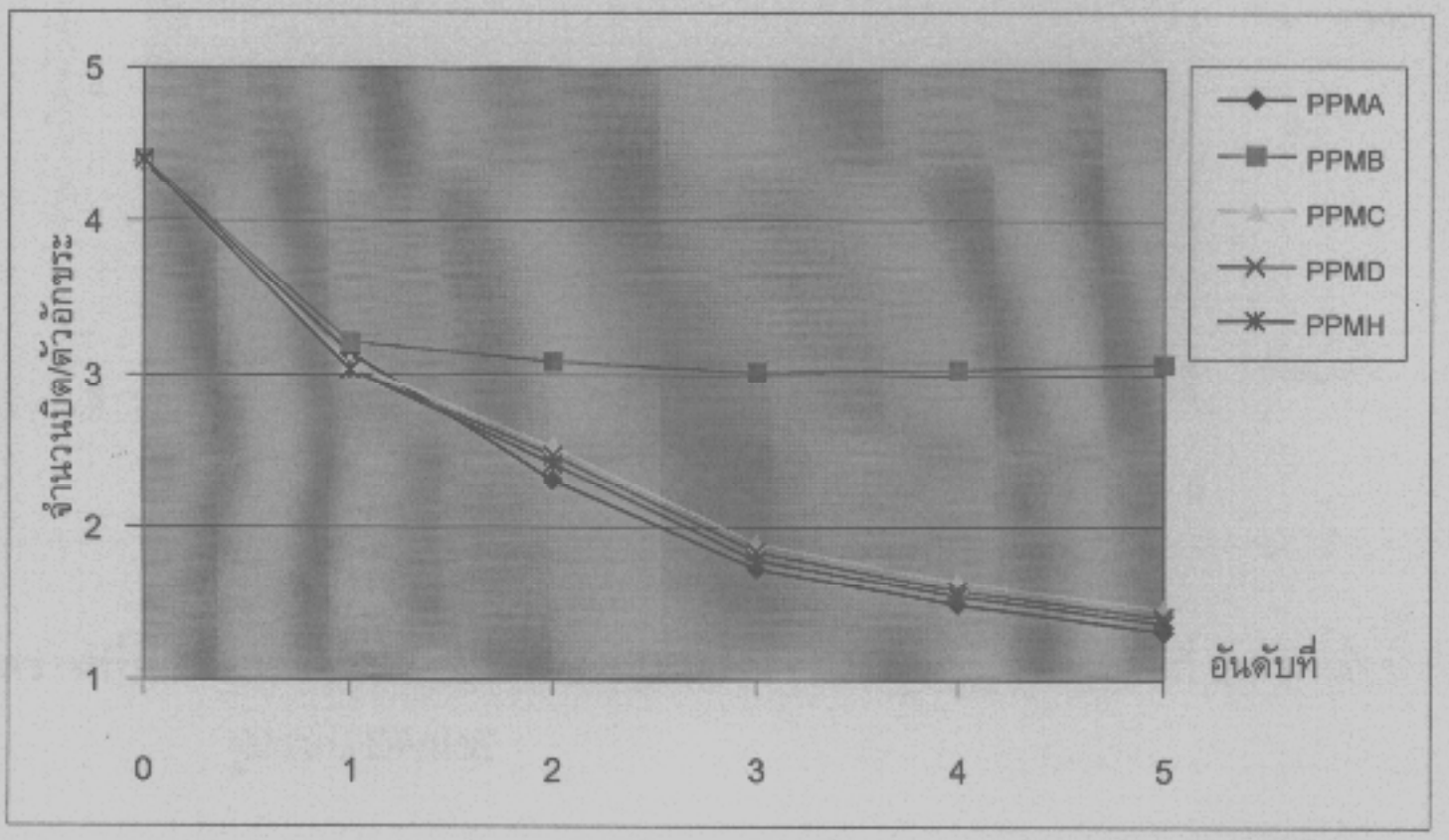


กราฟที่ 4.6 แสดงเวลาเฉลี่ยของการถอดรหัสของข้อมูลทดสอบทั้งหมด (ลำดับที่ 1-6) อันดับที่ 0-5 รูปแบบ NORMAL

จากกราฟที่ 4.5-4.6 จะเห็นได้ว่าเวลาเฉลี่ยของการถอดรหัสมีค่าประมาณ 2 เท่าของเวลาที่ใช้ในการเข้ารหัส

2) การทดลองการประมวลผลข้อมูลภาษาไทยเบื้องต้นในรูปแบบ DIVIDE

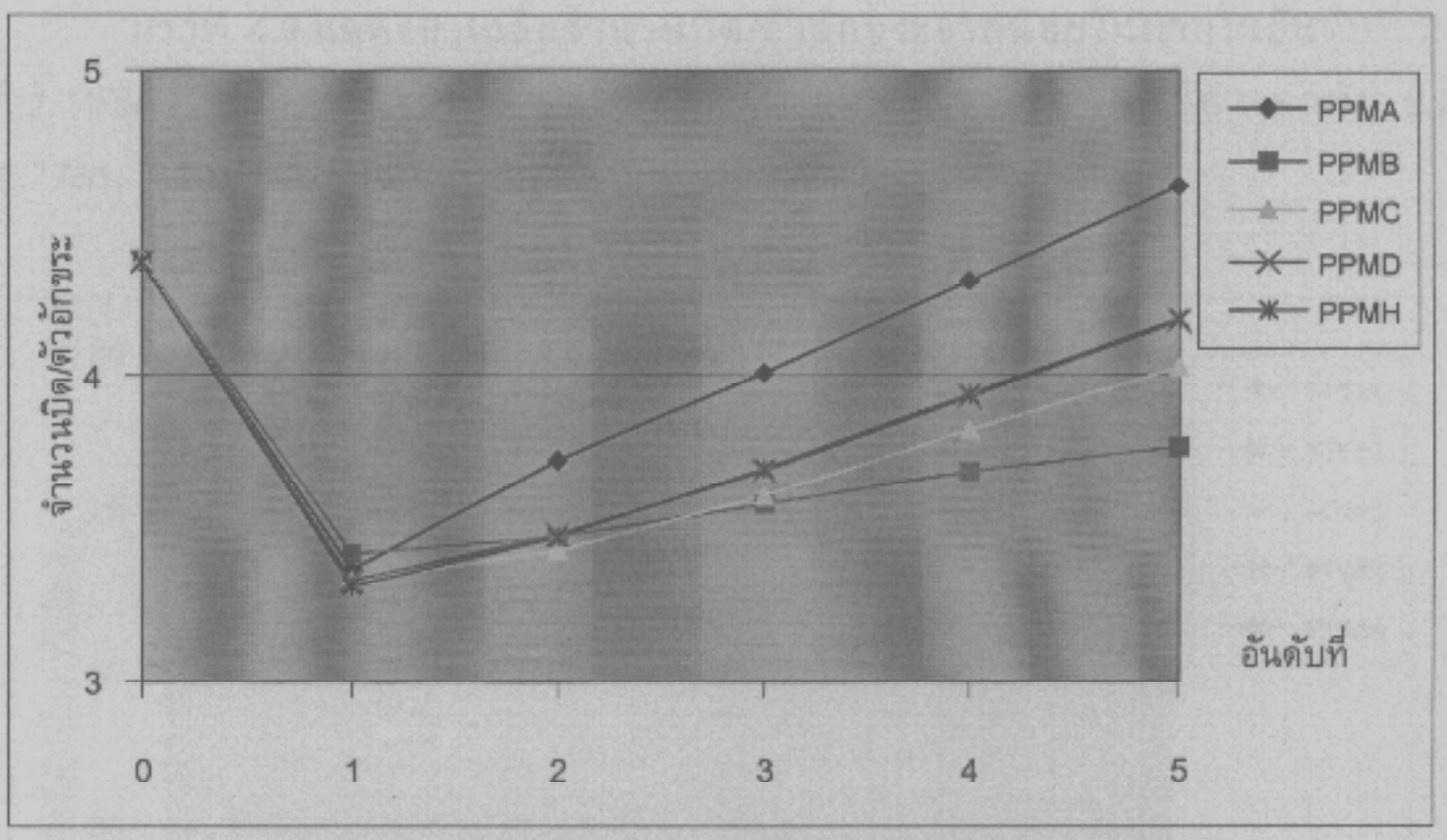
(2.1) เมื่อคลังข้อมูลทดสอบเป็นสับเซตของคลังข้อมูลฝึกสอน แสดงดังกราฟที่ 4.7



กราฟที่ 4.7 แสดงค่าเฉลี่ยจำนวนบิต/ตัวอักขระของข้อมูลทดสอบลำดับที่ 1-4 อันดับที่ 0-5 รูปแบบ DIVIDE

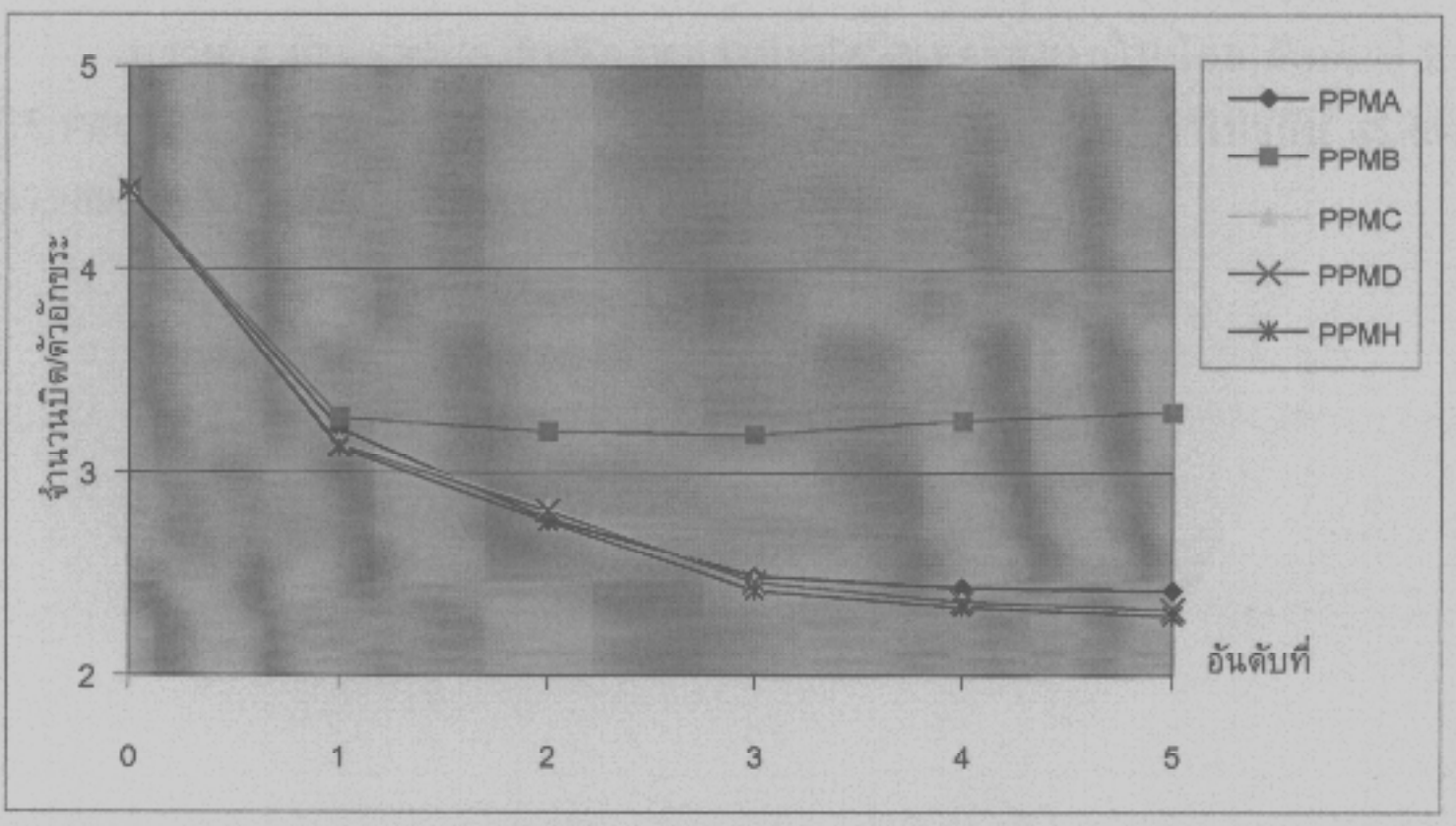
กราฟที่ 4.7 แสดงค่าเฉลี่ยจำนวนบิต/ตัวอักขระจากน้อยไปมากในอันดับที่ 2 ดังนี้ วิธี PPMA, วิธี PPMH, วิธี PPMD, วิธี PPMC และวิธี PPMB โดยมีค่าเท่ากับ 2.3037, 2.4063, 2.4698, 2.5184 และ 3.0759 บิต ตามลำดับ

(2.2) เมื่อคลังข้อมูลทดสอบไม่เป็นสับเซตของคลังข้อมูลฝึกสอน



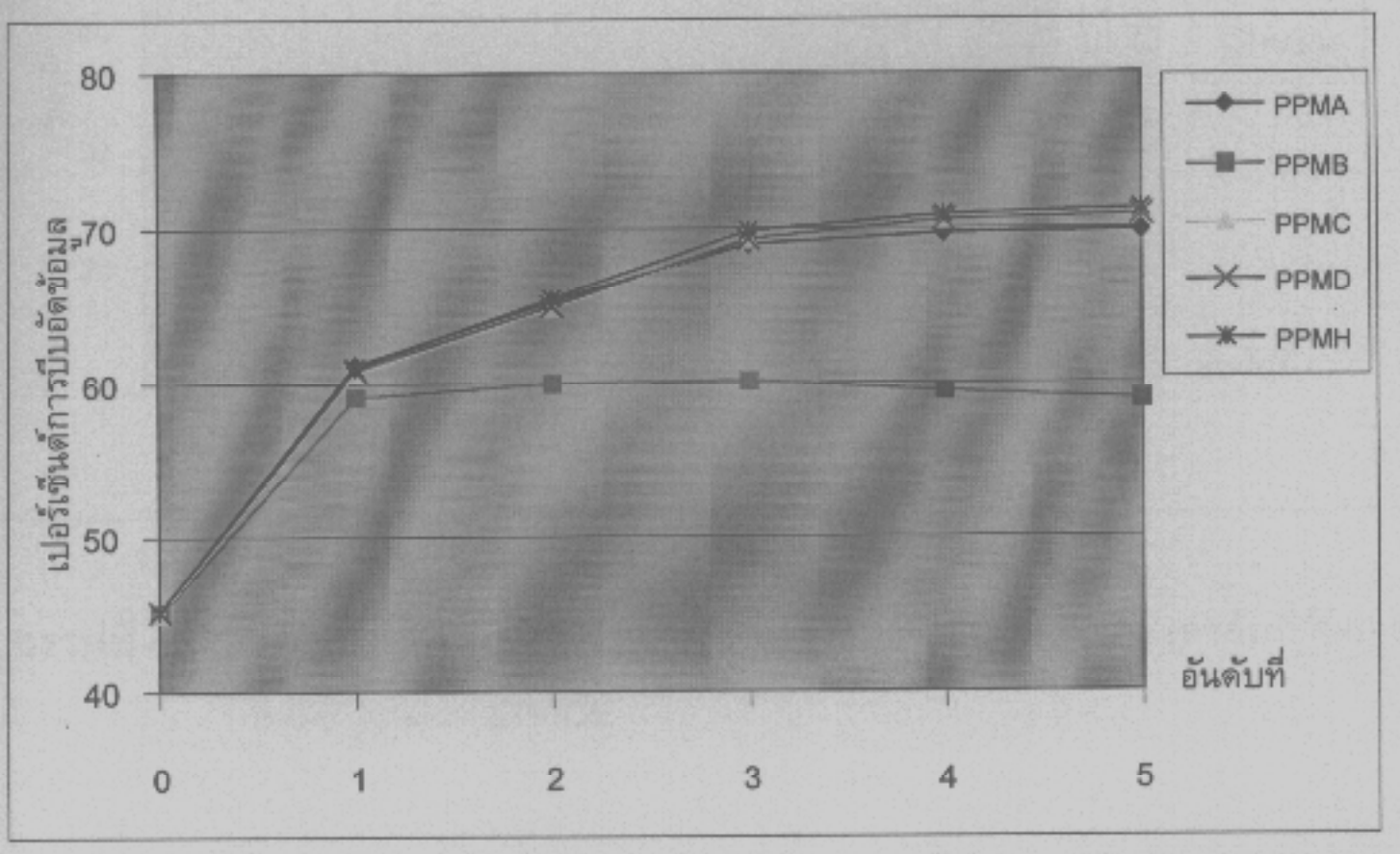
กราฟที่ 4.8 แสดงค่าเฉลี่ยจำนวนบิต/ตัวอักขระของข้อมูลทดสอบลำดับที่ 5-6 อันดับที่ 0-5 รูปแบบ DIVIDE

กราฟ 4.8 แสดงค่าเฉลี่ยจำนวนบิต/ตัวอักขระจากน้อยไปมากในอันดับที่ 2 ดังนี้ วิธี PPMC, วิธี PPMH, วิธี PPMB, วิธี PPMD และวิธี PPMA โดยมีค่าเท่ากับ 3.4273, 3.4761, 3.4763, 3.4791 และ 3.7208 บิต ตามลำดับ



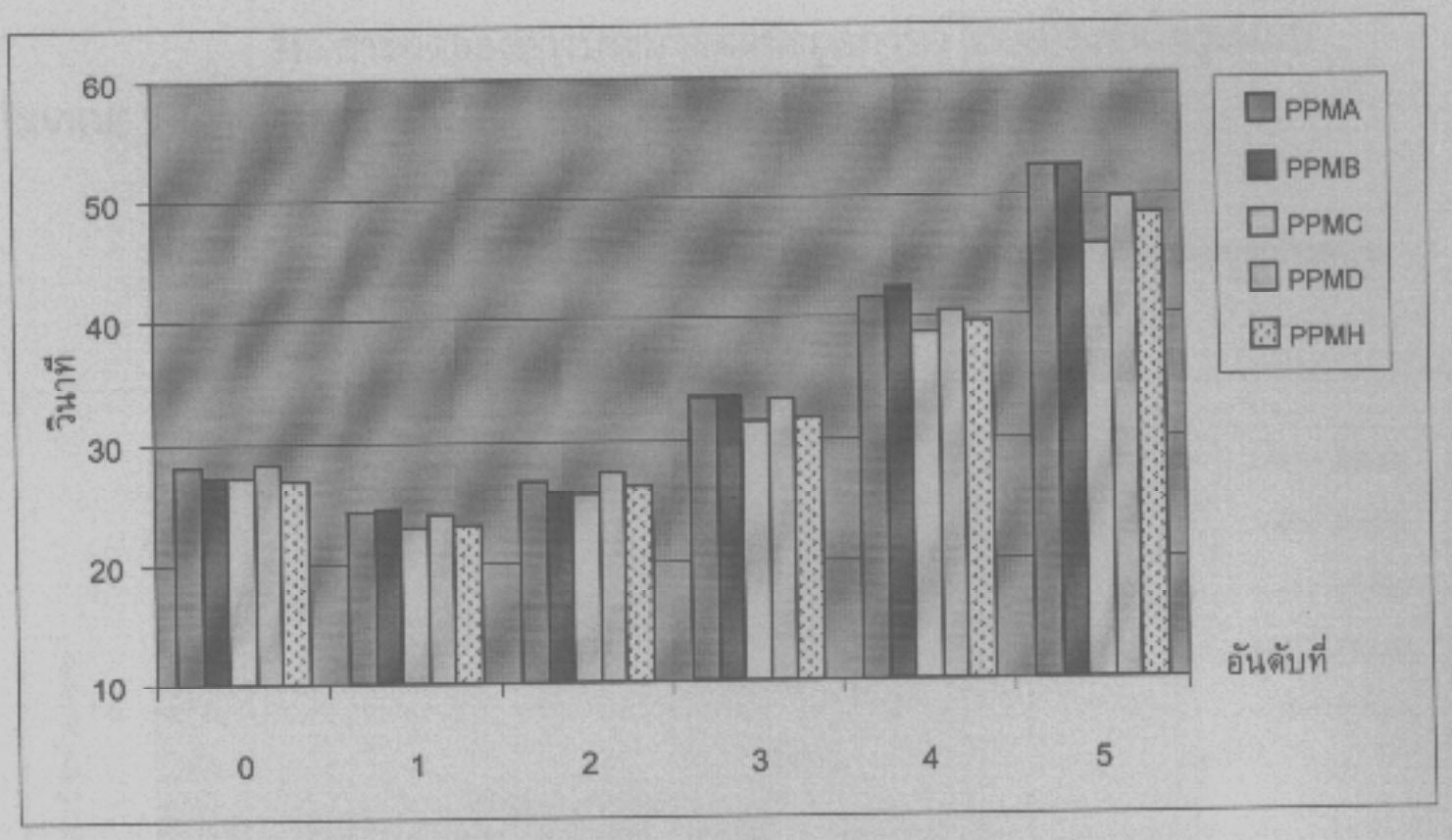
กราฟที่ 4.9 แสดงค่าเฉลี่ยจำนวนบิต/ตัวอักขระของข้อมูลทดสอบทั้งหมด (ลำดับที่ 1-6) อันดับที่ 0-5 รูปแบบ DIVIDE

กราฟ 4.9 แสดงค่าเฉลี่ยจำนวนบิต/ตัวอักขระจากน้อยไปมากในอันดับที่ 2 ดังนี้ วิธี PPMH, วิธี PPMA, วิธี PPMD, วิธี PPMC และวิธี PPMB โดยมีค่าเท่ากับ 2.7629, 2.7761, 2.8062, 2.8214 และ 3.2094 บิต ตามลำดับ

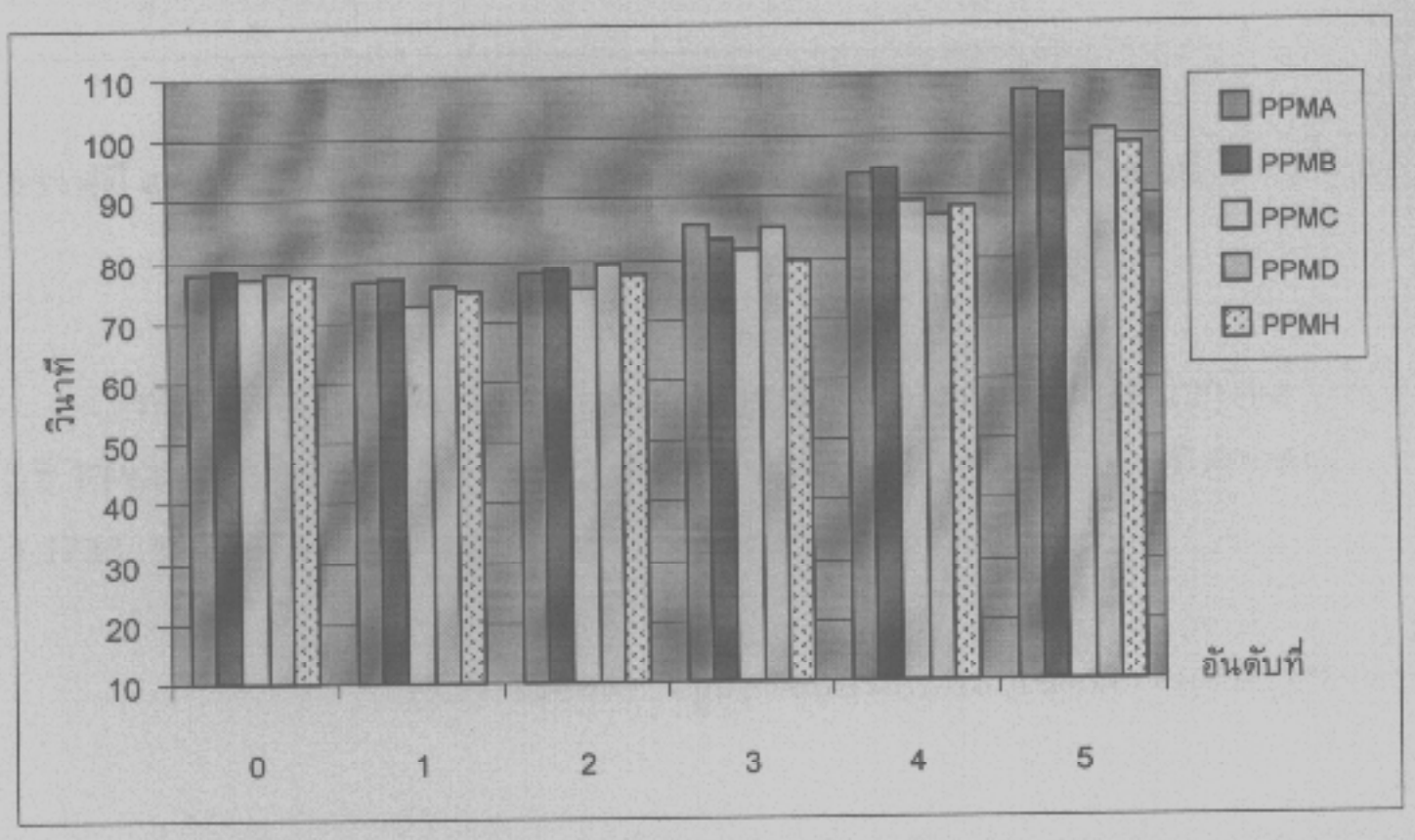


กราฟที่ 4.10 แสดงเปอร์เซ็นต์การบีบอัดของข้อมูลทดสอบทั้งหมด (ลำดับที่ 1-6) อันดับที่ 0-5 รูปแบบ DIVIDE

กราฟ 4.10 แสดงประสิทธิภาพการบีบอัด (%) จากมากไปน้อย อันดับที่ 2 ดังนี้ วิธี PPMH, วิธี PPMA, วิธี PPMD, วิธี PPMC และวิธี PPMB โดยมีค่าเท่ากับ 65.4643%, 65.2996%, 64.9225%, 64.7331% และ 59.8833% ตามลำดับ



กราฟที่ 4.11 แสดงเวลาเฉลี่ยของการเข้ารหัสของข้อมูลทดสอบทั้งหมด (ลำดับที่ 1-6) อันดับที่ 0-5 รูปแบบ DIVIDE

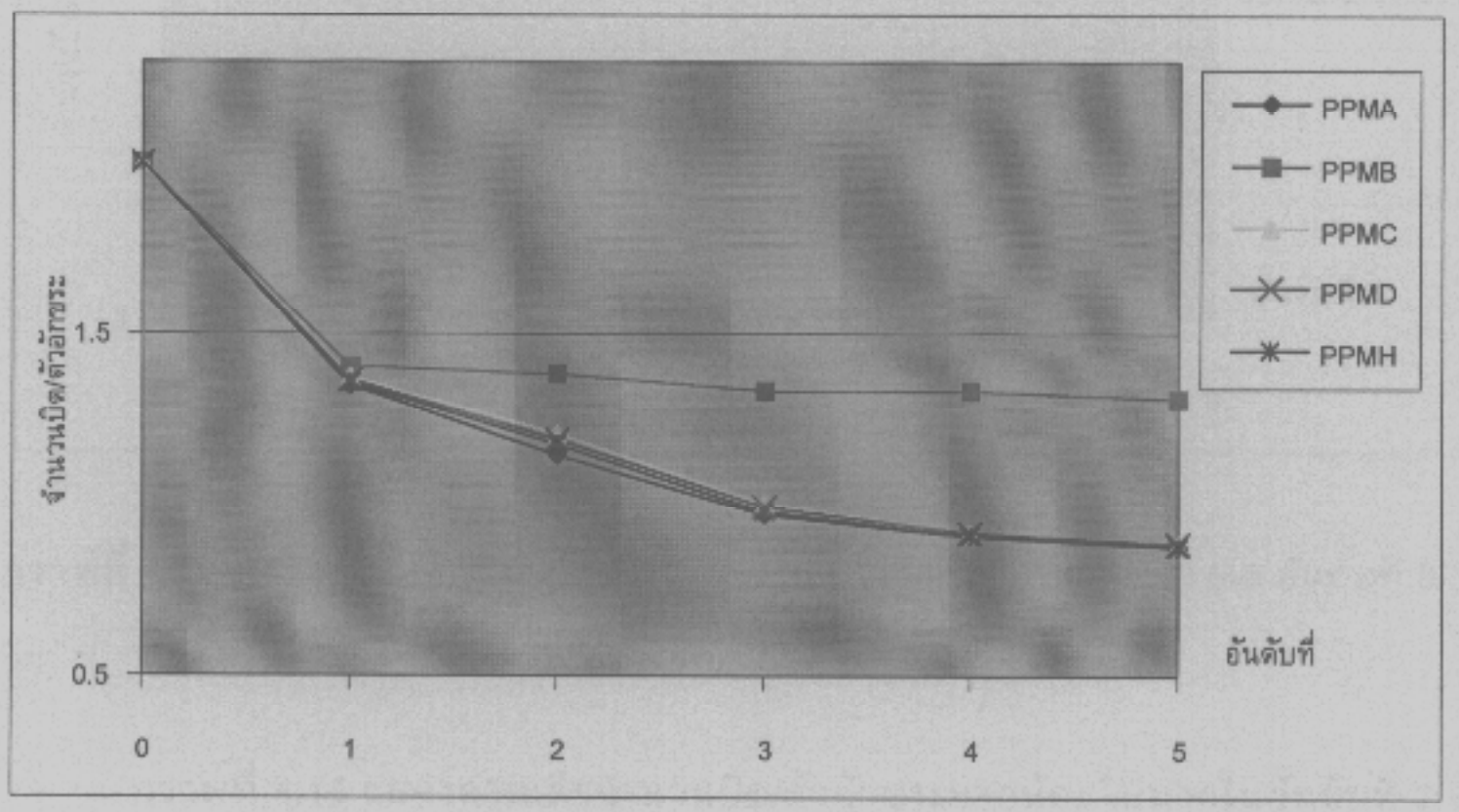


กราฟที่ 4.12 แสดงเวลาเฉลี่ยของการถอดรหัสของข้อมูลทดสอบทั้งหมด (ลำดับที่ 1-6) อันดับที่ 0-5 รูปแบบ DIVIDE

จากกราฟที่ 4.11-4.12 จะเห็นได้ว่าเวลาเฉลี่ยของการถอดรหัสมีค่าประมาณ 3 เท่าของเวลาที่ใช้ในการเข้ารหัส

3) การทดลองการประมวลผลข้อมูลภาษาไทยเบื้องต้นในรูปแบบ DIVIDE_HUF_CHR

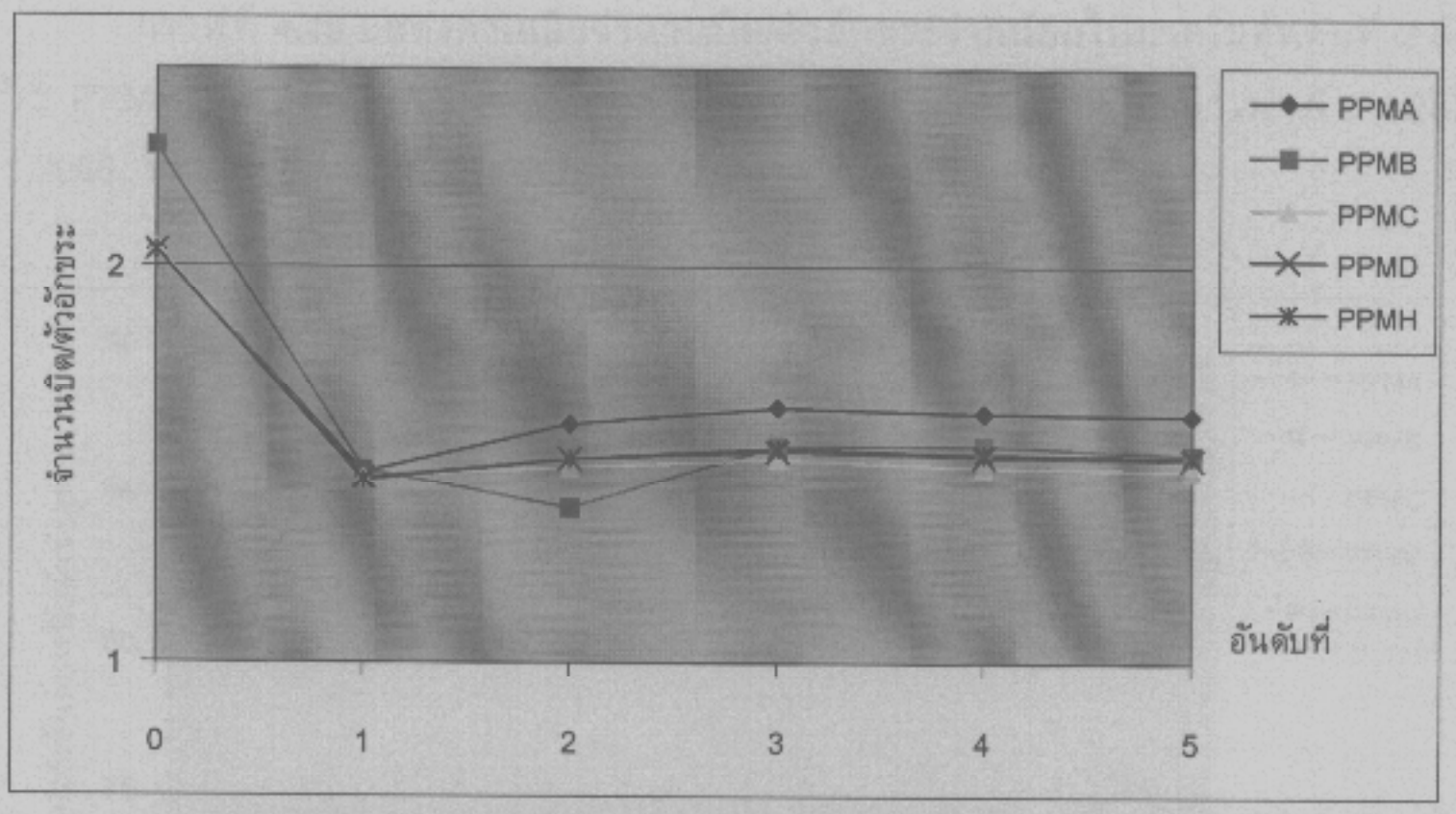
(3.1) เมื่อคลังข้อมูลทดสอบเป็นสับเซตของคลังข้อมูลฝึกสอน



กราฟที่ 4.13 แสดงค่าเฉลี่ยจำนวนบิต/ตัวอักขระของข้อมูลทดสอบลำดับที่ 1-4 อันดับที่ 0-5 รูปแบบ DIVIDE_HUF_CHR

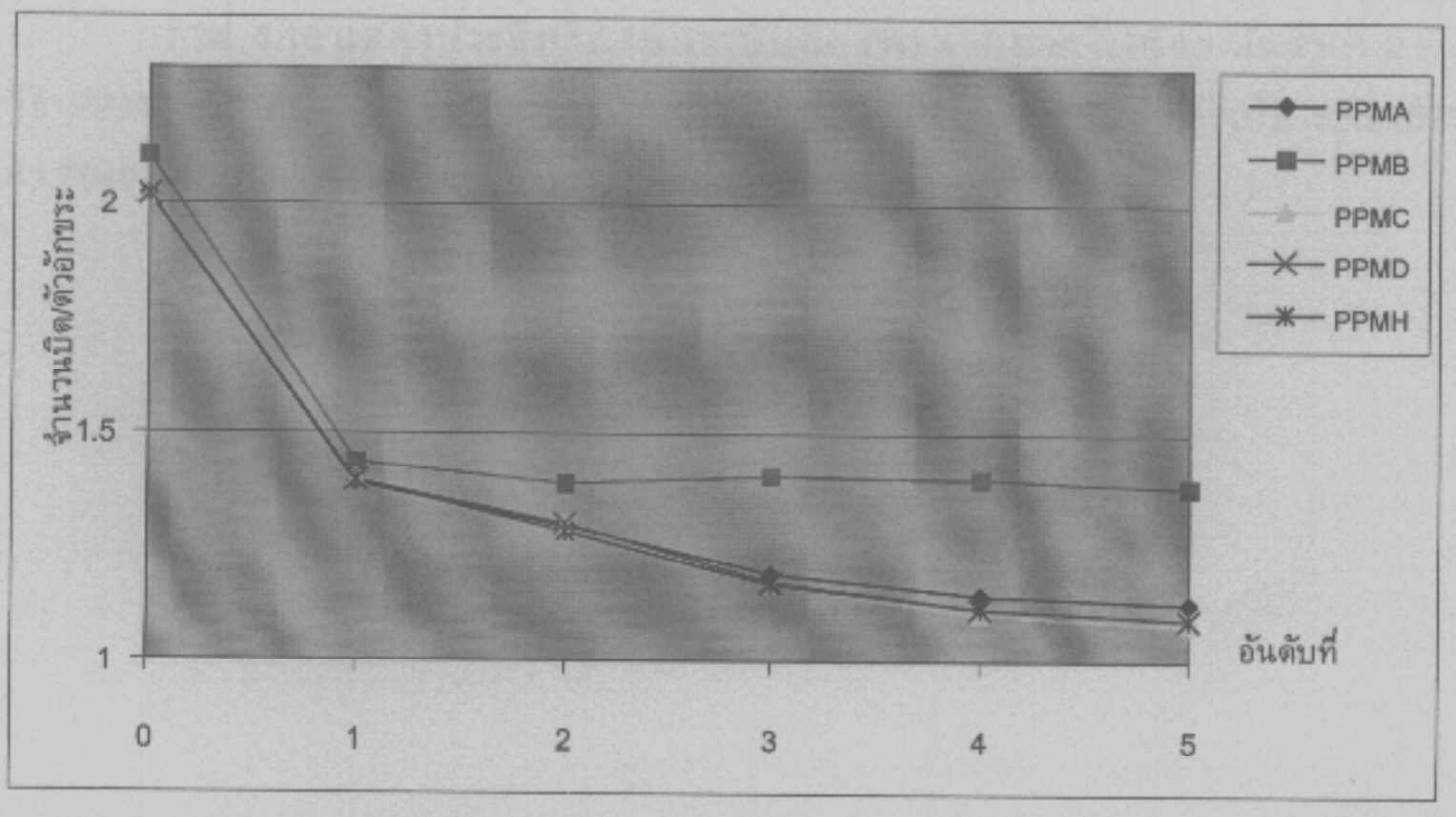
กราฟที่ 4.13 แสดงค่าเฉลี่ยจำนวนบิต/ตัวอักขระจากน้อยไปมากในอันดับที่ 2 ดังนี้ วิธี PPMA, วิธี PPMH, วิธี PPMD, วิธี PPMC และวิธี PPMB โดยมีค่าเท่ากับ 1.1438, 1.1733, 1.1958, 1.2049, และ 1.3861 บิต ตามลำดับ

(3.2) เมื่อคลังข้อมูลทดสอบไม่เป็นสับเซตของคลังข้อมูลฝึกสอน



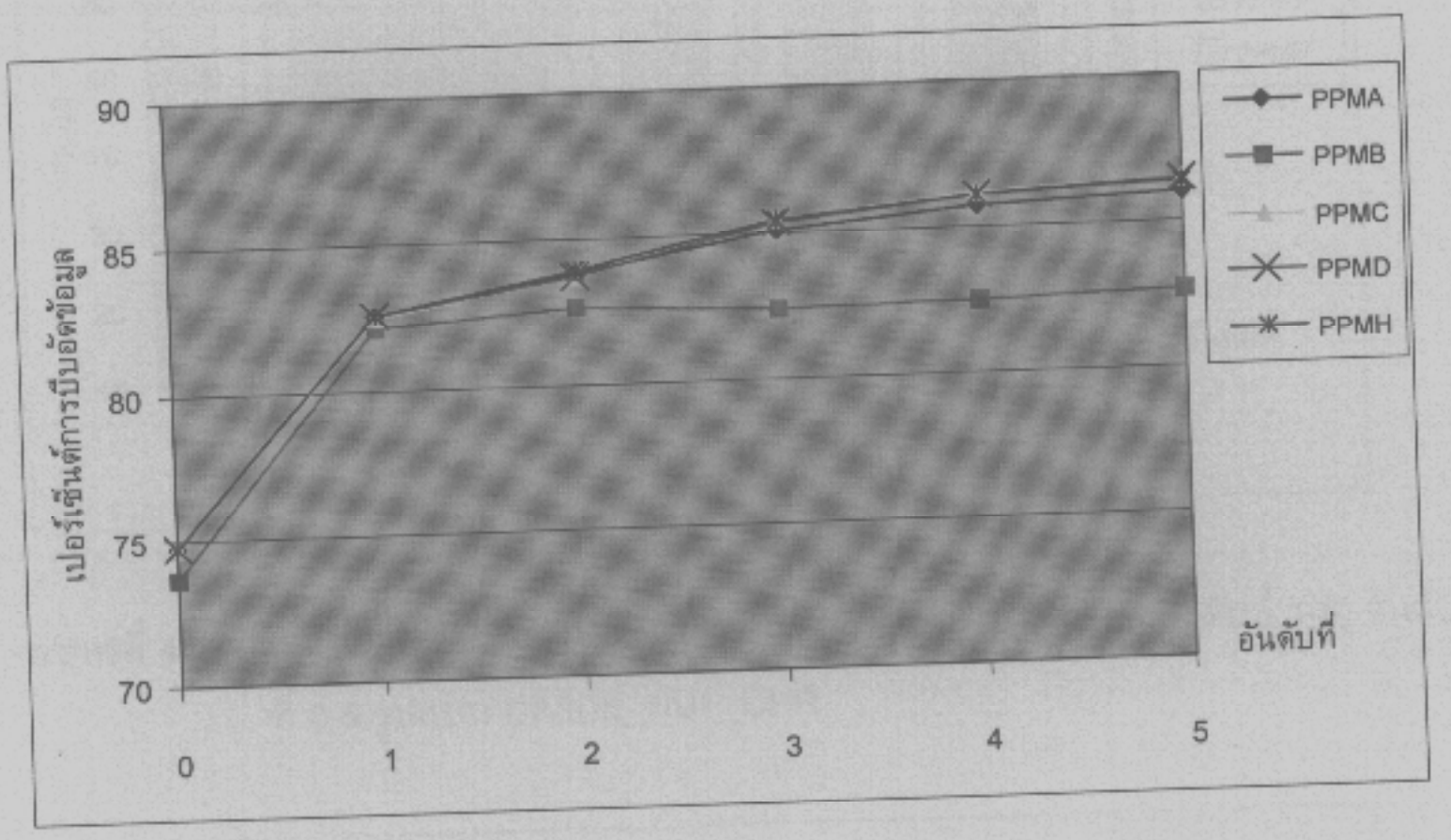
กราฟที่ 4.14 แสดงค่าเฉลี่ยจำนวนบิต/ตัวอักขระของข้อมูลทดสอบลำดับที่ 5-6 อันดับที่ 0-5 รูปแบบ DIVIDE_HUF_CHR

กราฟที่ 4.14 แสดงค่าเฉลี่ยจำนวนบิต/ตัวอักขระจากน้อยไปมากในอันดับที่ 2 ดังนี้ วิธี PPMB, วิธี PPMC, วิธี PPMH, วิธี PPMD และวิธี PPMA โดยมีค่าเท่ากับ 1.3924, 1.496, 1.5162, 1.5166 และ 1.6019 บิต ตามลำดับ



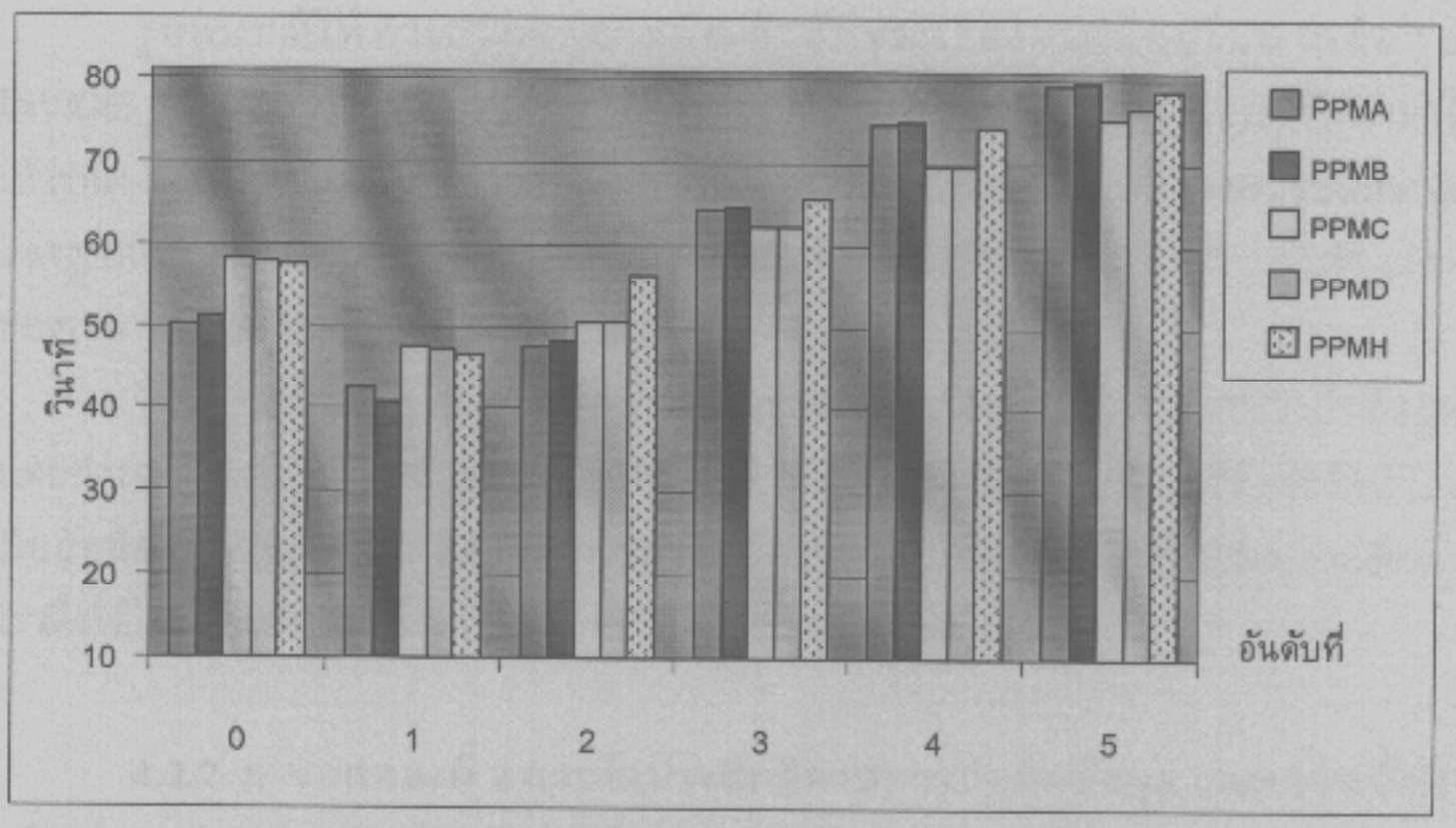
กราฟที่ 4.15 แสดงค่าเฉลี่ยจำนวนบิต/ตัวอักขระของข้อมูลทดสอบทั้งหมด (ลำดับที่ 1-6) อันดับที่ 0-5 รูปแบบ DIVIDE_HUF_CHR

กราฟที่ 4.15 แสดงค่าเฉลี่ยจำนวนบิต/ตัวอักขระจากน้อยไปมากในอันดับที่ 2 ดังนี้ วิธี PPMH, วิธี PPMA, วิธี PPMC, วิธี PPMD และวิธี PPMB โดยมีค่าเท่ากับ 1.2876, 1.2965, 1.3019, 1.3027 และ 1.3882 บิต ตามลำดับ

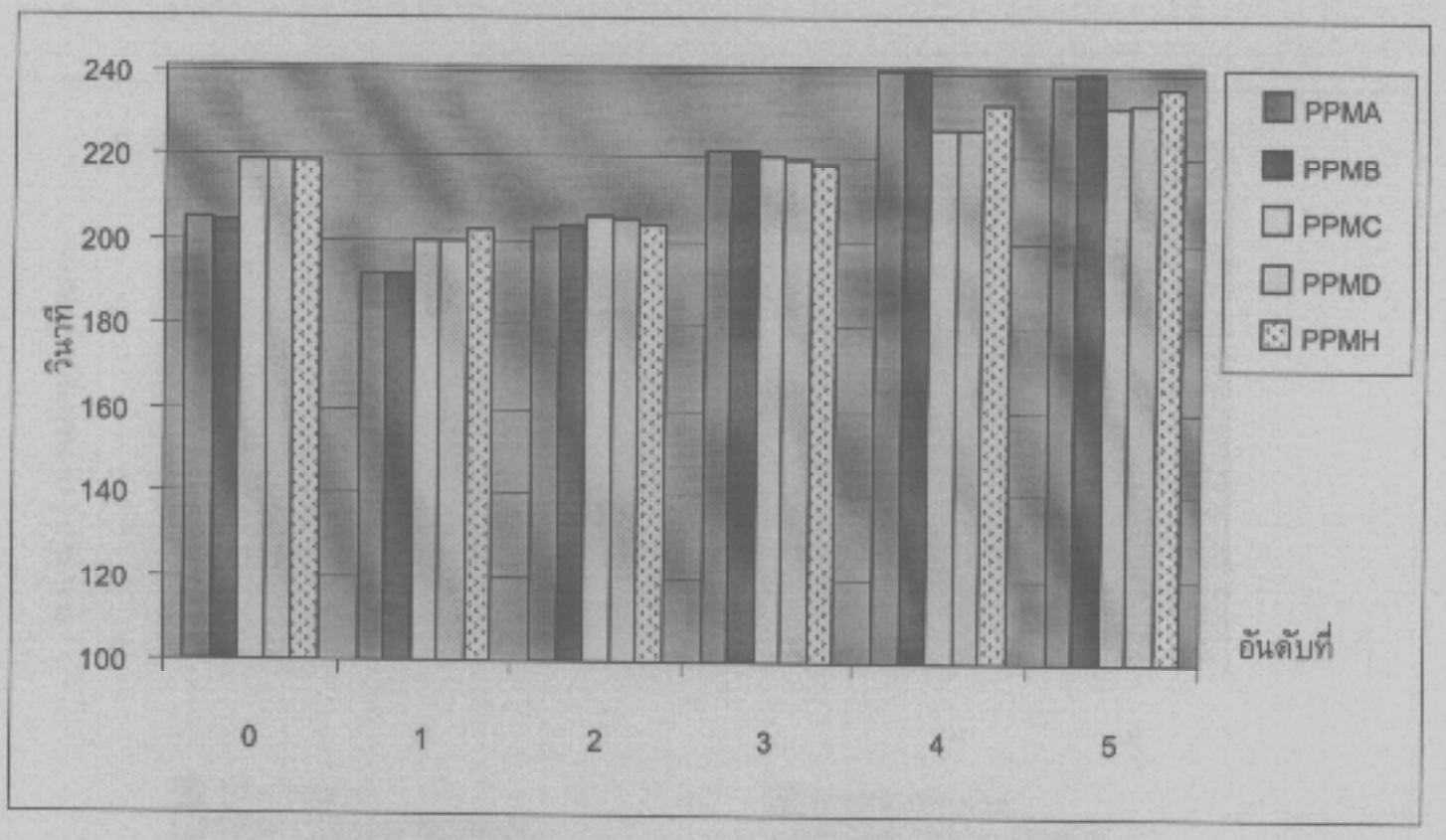


กราฟที่ 4.16 แสดงเปอร์เซ็นต์ การบีบอัดของข้อมูลทดสอบทั้งหมด (ลำดับที่ 1-6) อันดับที่ 0-5 รูปแบบ DIVIDE_HUF_CHR

กราฟ 4.16 แสดงประสิทธิภาพการบีบอัด (%) จากมากไปน้อย อันดับที่ 2 ดังนี้ วิธี PPMH, วิธี PPMA, วิธี PPMC, วิธี PPMD และวิธี PPMB โดยมีค่าเท่ากับ 83.9052%, 83.794%, 83.7262%, 83.7166% และ 82.6478% ตามลำดับ



กราฟที่ 4.17 แสดงเวลาเฉลี่ยของการเข้ารหัสของข้อมูลทดสอบทั้งหมด (ลำดับที่ 1-6) อันดับที่ 0-5 รูปแบบ DIVIDE_HUF_CHR



กราฟที่ 4.18 แสดงเวลาเฉลี่ยของการถอดรหัสของข้อมูลทดสอบทั้งหมด (ลำดับที่ 1-6) อันดับที่ 0-5 รูปแบบ DIVIDE_HUF_CHR

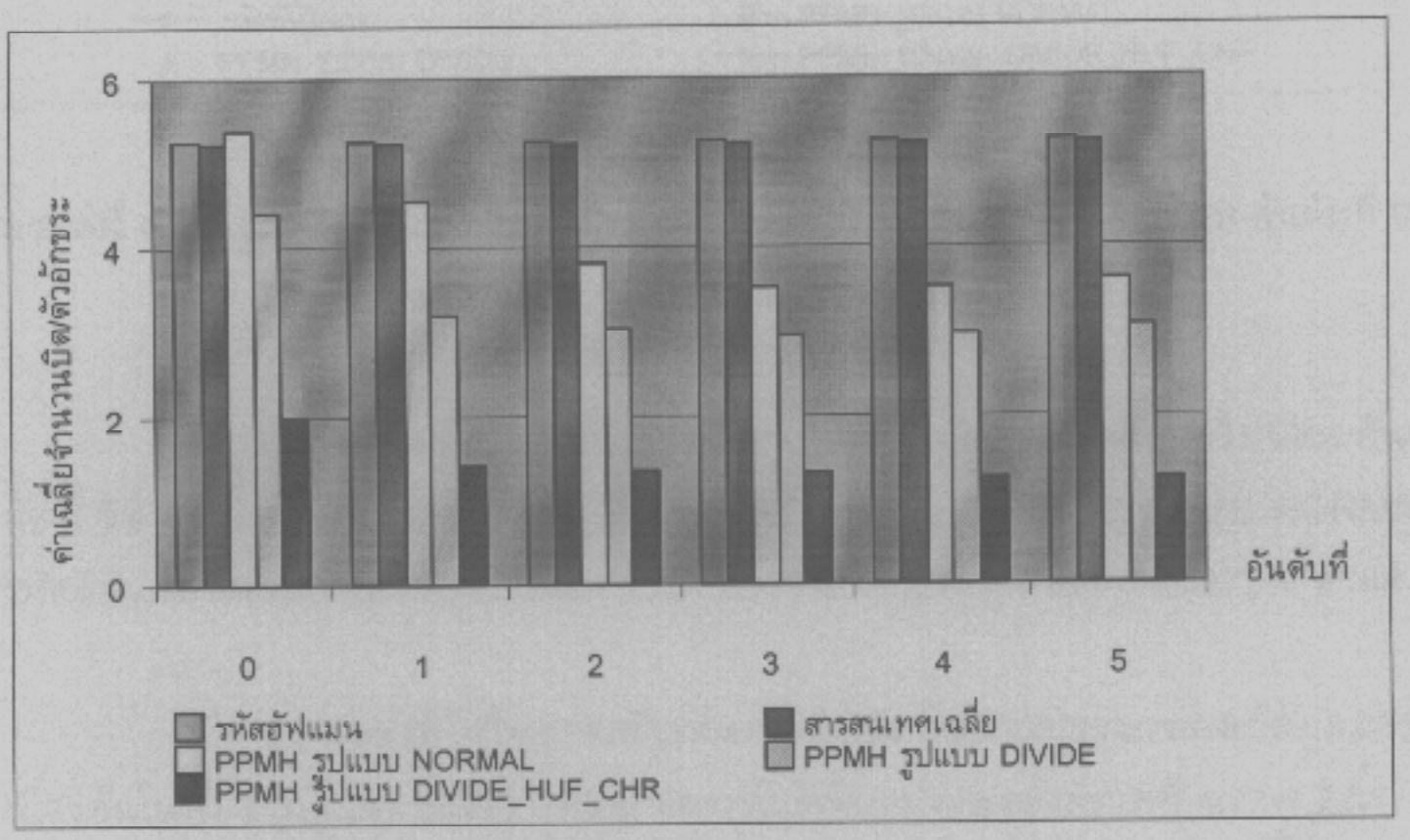
จากกราฟที่ 4.17-4.18 จะเห็นได้ว่าเวลาเฉลี่ยของการถอดรหัสมีค่าประมาณ 4 เท่าของเวลาที่ใช้ในการเข้ารหัส

รูปแบบที่ให้ค่าเฉลี่ยจำนวนบิต/ตัวอักขระเรียงจากน้อยไปมากคือ รูปแบบ DIVIDE_HUF_CHR, รูปแบบ DIVIDE และ รูปแบบ NORMAL ส่วนรูปแบบที่ให้เวลาในการเข้ารหัส-ถอดรหัสเรียงลำดับจากมากไปน้อยคือ รูปแบบ DIVIDE_HUF_CHR, รูปแบบ DIVIDE และรูปแบบ NORMAL โดยให้เวลาที่ใช้ในการถอดรหัสมีค่าประมาณ 4 เท่า, 3 เท่า และ 2 เท่าของเวลาที่ใช้ในการเข้ารหัส ตามลำดับ

วิธี PPMH มีแนวโน้มที่จะให้ประสิทธิภาพในการบีบอัดข้อมูลในการบีบอัดข้อมูลสูงสุดและค่าเฉลี่ยจำนวนบิต/ตัวอักขระต่ำที่สุด (ลำดับที่ 1-6) วิธี PPMH จะให้ประสิทธิภาพการบีบอัดข้อมูลดีประมาณ 2 บิต/ตัวอักขระ และอัตราการบีบอัดข้อมูลได้ถึงประมาณ 84% ดังนั้นเราจึงใช้ให้ PPMH เป็นตัวแบบของ PPM ในการทดลองต่อไป

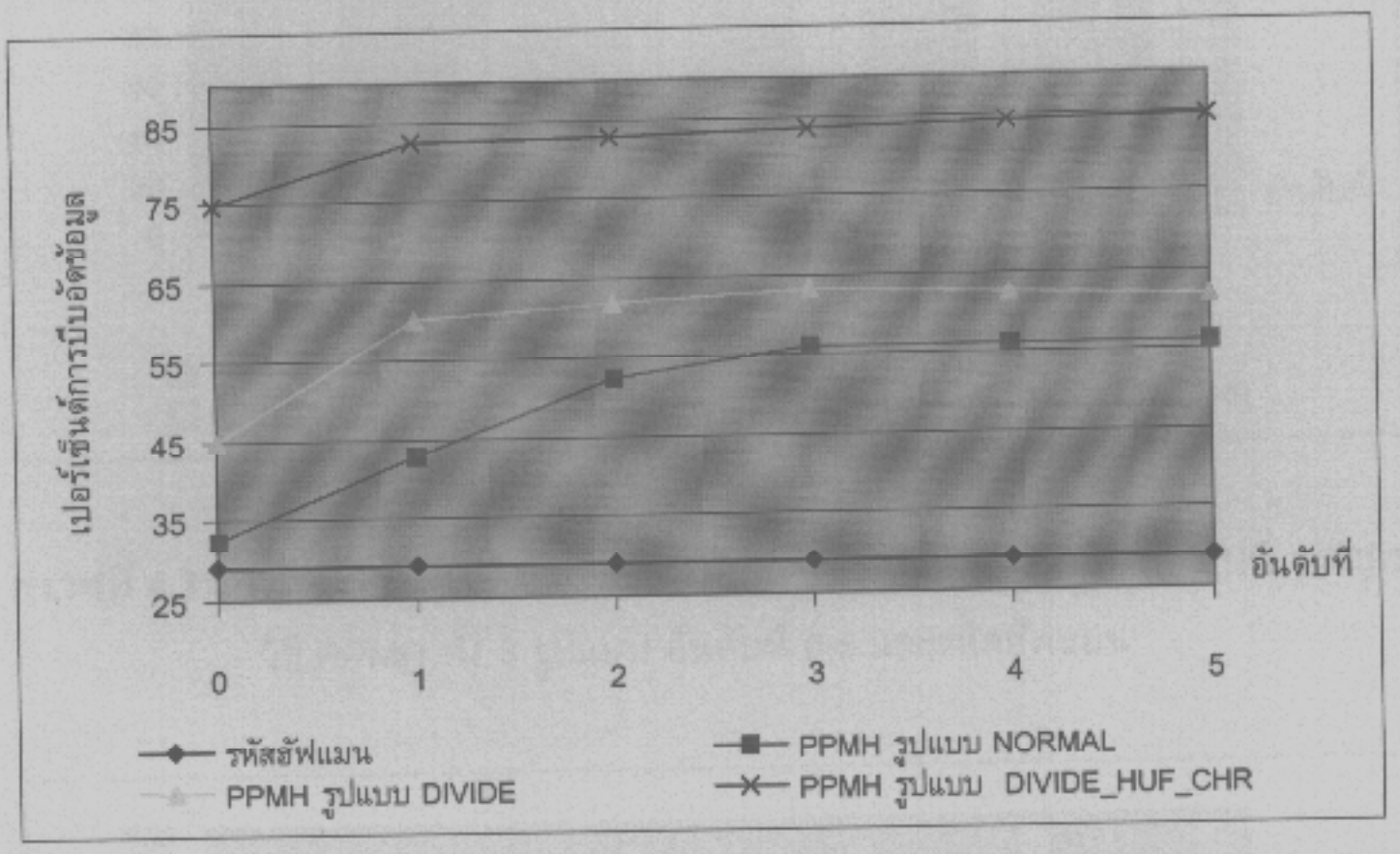
4.2.2 การทดลองที่ 2 การวัดประสิทธิภาพการบีบอัดข้อมูล และเวลาที่ใช้ในการเข้ารหัส-ถอดรหัสวิธี PPMH, รหัสฮัฟแมน และสารสนเทศเฉลี่ย

1) ค่าเฉลี่ยจำนวนบิต/ตัวอักขระในการประมวลผลข้อมูลภาษาไทยเบื้องต้นใน 3 รูปแบบของวิธี PPMH, รหัสฮัฟแมน และสารสนเทศเฉลี่ย แสดงดังกราฟที่ 4.19



กราฟที่ 4.19 แสดงผลค่าเฉลี่ยจำนวนบิต/ตัวอักขระของวิธี PPMH ทั้ง 3 รูปแบบ ในอันดับที่ 0-5, รหัสฮัฟแมน และสารสนเทศเฉลี่ย ของข้อมูลทดสอบทั้งหมด (ลำดับที่ 1-10)

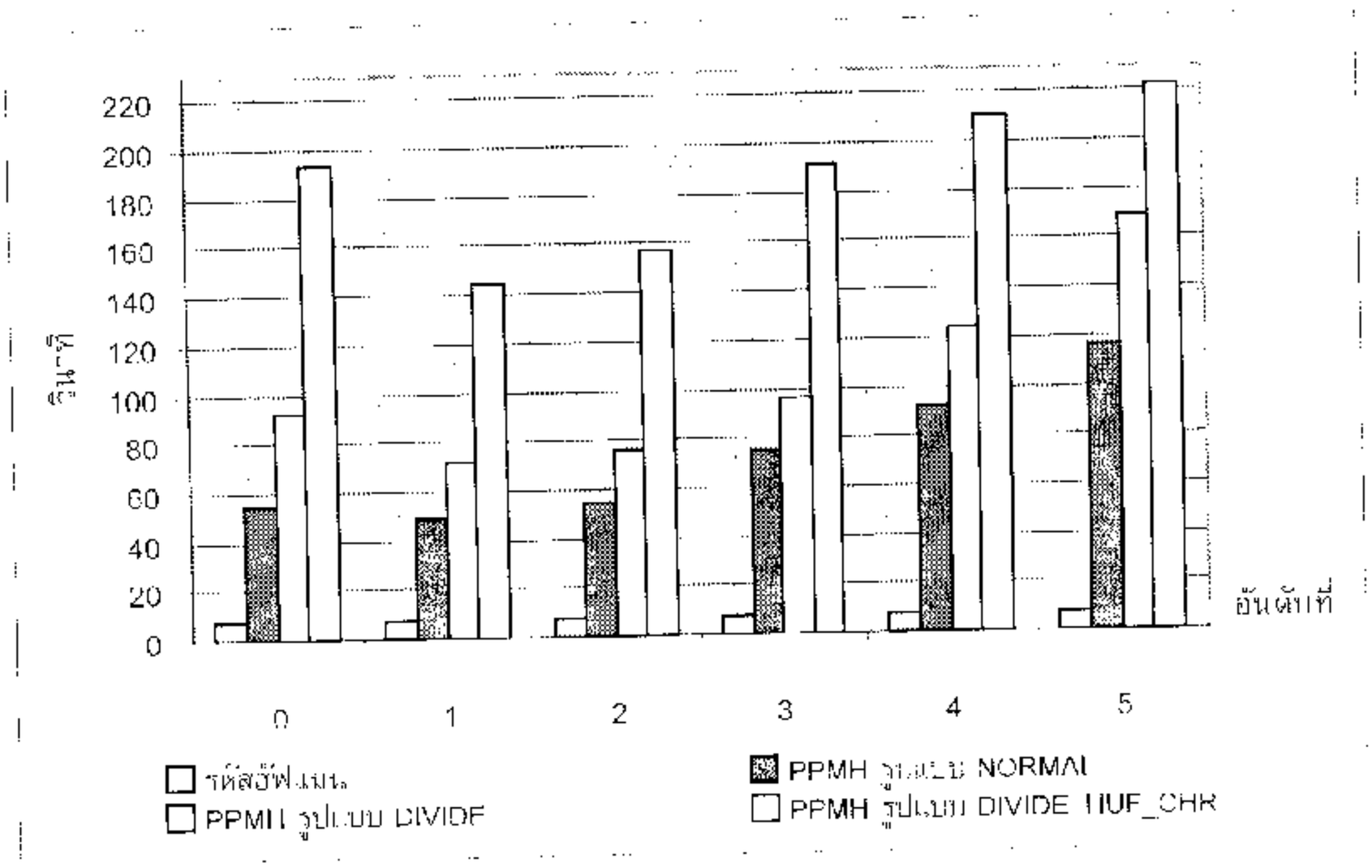
กราฟที่ 4.19 แสดงค่าเฉลี่ยจำนวนบิต/ตัวอักขระจากน้อยไปมากได้ดังนี้ วิธี PPMH อันดับที่ 2 รูปแบบ DIVIDE_HUF_CHR, รูปแบบ DIVIDE, รูปแบบ NORMAL, สารสนเทศเฉลี่ย และรหัสฮัฟแมนโดยมีค่าเท่ากับ 1.36832, 3.04705, 3.7944, 5.23624 และ 5.25321 บิต ตามลำดับ โดยอันดับที่จะไม่มีผลต่อประสิทธิภาพการบีบอัดข้อมูลด้วยวิธีรหัสฮัฟแมน และสารสนเทศเฉลี่ย



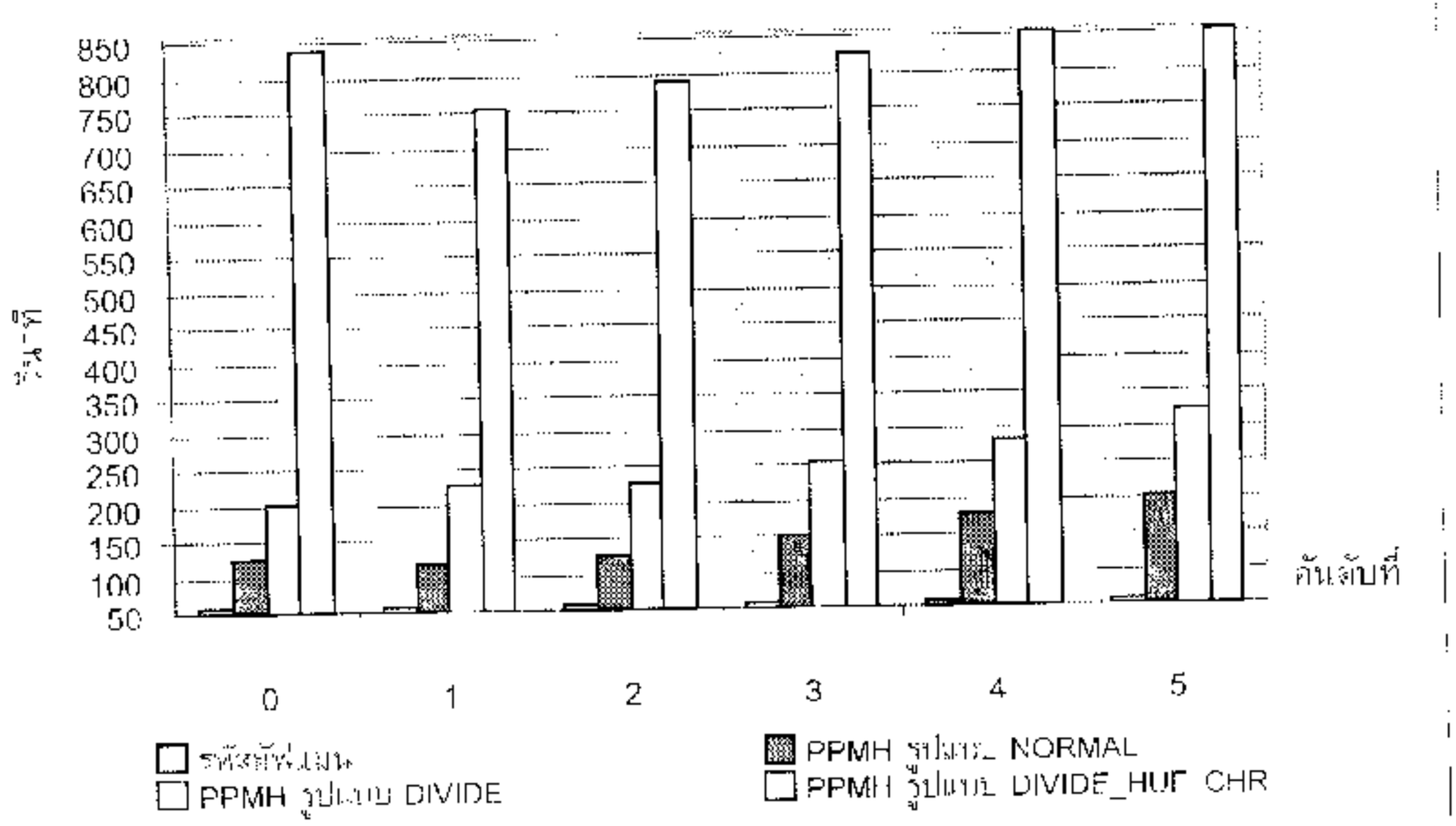
กราฟที่ 4.20 แสดงเปอร์เซ็นต์ การบีบอัดข้อมูลของวิธี PPMH ทั้ง 3 รูปแบบ อันดับที่ 0-5 และรหัสฮัฟแมน ของข้อมูลทดสอบทั้งหมด (ลำดับที่ 1-10)

กราฟที่ 4.20 แสดงประสิทธิภาพการบีบอัด (%) จากมากไปน้อย อันดับที่ 2 ดังนี้ วิธี PPMH รูปแบบ DIVIDE_HUF_CHR, รูปแบบ DIVIDE, รูปแบบ NORMAL และรหัสฮัฟแมน โดยมีค่าเท่ากับ 82.8961%, 61.9119%, 52.5701% และ 29.0272% ตามลำดับ

2) เวลาที่ใช้ในการเข้ารหัส-ถอดรหัส ในการประมวลผลข้อมูลภาษาไทยเบื้องต้นในทั้ง 3 รูปแบบ ของวิธี PPMH และรหัสฮัฟแมน แสดงดังกราฟที่ 4.21-4.22



กราฟที่ 4.21 แสดงเวลาเฉลี่ยของการเข้ารหัสในข้อมูลทดสอบทั้งหมด (ลำดับที่ 1-10) ของวิธี PPMH ทั้ง 3 รูปแบบ อันดับที่ 0-5 และรหัสฮัฟแมน



กราฟที่ 4.22 แสดงเวลาเฉลี่ยของการถอดรหัสในข้อมูลทดสอบทั้งหมด (ลำดับที่ 1-10) ของวิธี PPMH ทั้ง 3 รูปแบบ อันดับที่ 0-5 และรหัสฮัฟแมน

การทดลองที่ 2 พบว่า วิธี PPMH รูปแบบ DIVIDE_HUF_CHR, รูปแบบ DIVIDE และรูปแบบ NORMAL อันดับที่ 0-5 จะให้ประสิทธิภาพการบีบอัดข้อมูลที่สูงกว่ารหัสฮัฟแมนและสารสนเทศ ในทางกลับกันเวลาที่ใช้ในการเข้ารหัส-ถอดรหัสของรหัสฮัฟแมนจะน้อยกว่าวิธีPPMH รูปแบบ NORMAL, วิธี PPMH รูปแบบ DIVIDE และวิธี PPMH รูปแบบ DIVIDE_HUF_CHR ตามลำดับ นอกจากนี้เวลาที่ใช้ในการถอดรหัสรูปแบบ DIVIDE_HUF_CHR จะมากกว่าเวลาในการเข้ารหัส เพราะในการถอดรหัสข้อมูลต้องตรวจสอบกระแสข้อมูลบิตกับรหัสฮัฟแมนในตารางรหัสฮัฟแมนในแต่ละรอบที่อ่านข้อมูล

วิธี PPMH มีแนวโน้มที่จะให้ประสิทธิภาพในการบีบอัดข้อมูลสูงสุด ทั้งกรณีที่คลังข้อมูลทดสอบเป็นและไม่เป็นสับเซตของคลังข้อมูลฝึกสอน วิธี PPMH ในรูปแบบ DIVIDE_HUF_CHR จะให้ค่าเฉลี่ยต่ำสุดประมาณ 2 บิต/ตัวอักขระ และอัตราการบีบอัดข้อมูลได้ถึงประมาณ 84%

บทที่ 5

สรุปผลการวิจัยและข้อเสนอแนะ

การบีบอัดข้อมูลเป็นวิธีหนึ่งที่ช่วยเพิ่มประสิทธิภาพในการจัดเก็บข้อมูล งานวิจัยนี้เสนอแนวทางในการบีบอัดข้อความภาษาไทย โดยปรับปรุงวิธีการ Prediction by Partial Matching ด้วยการใช้ลักษณะของภาษาไทย เรียกว่า PPM-Hybrid (PPMH) และการประมวลผลข้อมูลเบื้องต้น 2 รูปแบบ คือ รูปแบบ DIVIDE และรูปแบบ DIVIDE_HUF_CHR

จากการทดลองเพื่อวัดประสิทธิภาพ พบว่า วิธี PPMH ให้ประสิทธิภาพการบีบอัดข้อมูลที่ดีที่สุด ทั้งกรณีที่คลังข้อมูลทดสอบที่เป็นและไม่เป็นสับเซตของคลังข้อมูลฝึกสอน เมื่อเปรียบเทียบกับ PPM อื่น ๆ รหัสฮัฟแมน และสารสนเทศเฉลี่ย

การประมวลข้อมูลเบื้องต้นทั้ง 2 รูปแบบ ให้ประสิทธิภาพการบีบอัดข้อมูลที่ PPM อันดับที่ 2 ได้ถึง 65.4643% และ 83.9052% และค่าจำนวนบิต/ตัวอักขระเฉลี่ยเท่ากับ 2.7629 และ 1.2876 บิต ตามลำดับ ในทางกลับกันเวลาที่ใช้ในการเข้ารหัส-ถอดรหัสจะสูงขึ้น โดยเรียงลำดับจากมากไปน้อยได้ดังนี้ รูปแบบ DIVIDE_HUF_CHR, รูปแบบ DIVIDE, รูปแบบที่ไม่ทำการประมวลผลเบื้องต้น และรหัสฮัฟแมน ตามลำดับ

วิธี PPMH รูปแบบ DIVIDE เป็นวิธีที่ให้ประสิทธิภาพการบีบอัดข้อมูลที่สูงกว่าวิธี PPMH รูปแบบที่ไม่ทำการประมวลผลเบื้องต้น และวิธีนี้ยังใช้เวลาในการเข้ารหัส-ถอดรหัสน้อยกว่าวิธี PPMH รูปแบบ DIVIDE_HUF_CHR สำหรับประสิทธิภาพการบีบอัดข้อมูลด้วยวิธี PPMH ในรูปแบบ DIVIDE อันดับที่ 2 เรียงจากน้อยไปมากได้แก่ วิธี PPMH, วิธี PPMA, วิธี PPMD, วิธี PPMC และ วิธี PPMB โดยมีค่าเท่ากับ 65.4643%, 65.2996%, 64.9225%, 64.7331% และ 59.8833% ตามลำดับ

ข้อเสนอแนะในการเลือกคลังข้อมูลฝึกสอนสำหรับสร้างตัวแบบ ซึ่งจะส่งผลต่อประสิทธิภาพในการบีบอัดข้อมูล ดังนั้นควรเลือกคลังข้อมูลฝึกสอนให้สอดคล้องกับลักษณะงานที่นำไปใช้

บรรณานุกรม

การท่องเที่ยวแห่งประเทศไทย. 2544. (7 ตุลาคม). ประวัติศาสตร์ไทยราชวงศ์ไทย. (Online). Available URL: http://www.discountthailand.com/thai/about_thailand/overview/

กำชัย ทองหล่อ. 2537. **หลักภาษาไทย**. พิมพ์ครั้งที่ 9. กรุงเทพมหานคร: สำนักพิมพ์รวมสาส์น.

กิติพัท มหาพันธ์. 2541. (7 ตุลาคม). สามก๊ก. (Online). Available URL: http://www.geocities.com/Paris/Cafe/1235/

ชมรมพุทธศาสตร์สากลในอุปถัมภ์สมเด็จพระมหารัชมังคลาจารย์. 2547. (10 มกราคม). มงคลชีวิต. (Online). Available URL: http://www.ibscenter.net/NewsSystem/

บรรเทา กิตติศักดิ์ และกัมพุชนาฏ เปรมกมล. 2530. **ลักษณะควรสังเกตบางประการของภาษาไทย**. พิมพ์ครั้งที่ 2. กรุงเทพมหานคร: โรงพิมพ์ไทยวัฒนา

ยืน ภู่วรวรรณ. 2527. **การวิเคราะห์ข้อมูลคำไทย**. ห้องปฏิบัติการวิจัยไมโครคอมพิวเตอร์ มหาวิทยาลัยเกษตรศาสตร์: 290-299.

ราชบัณฑิตยสถาน. 2525. **พจนานุกรมฉบับราชบัณฑิตยสถาน พ.ศ. 2525**. กรุงเทพมหานคร: อักษรเจริญทัศน์.

เรวดี ลิมปิโชติกุล. 2534. การปรับเปลี่ยนการอัดข้อความภาษาไทย. วิทยานิพนธ์วิทยาศาสตรมหาบัณฑิต จุฬาลงกรณ์มหาวิทยาลัย.

ลลิดา รัตนศรีทัย. 2546. (7 ตุลาคม). โครงการคอมพิวเตอร์ไอซีที. (Online). Available URL: http://www.thairath.co.th/thairath1/2546/itdigest/itdigest/nov/05/itdigest.php

วิชิต นานาน. 2546. (7 ตุลาคม). นิทานแก้วิเศษ, ความโลภ, น้ำผึ้งหยดเดียว, กบขี้เกียจ, ช้างของพระราชา, กบกาเจ็ดปาก. (Online). Available URL: http://www.nitanbox.com/tale_menu1.htm

ศูนย์เทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ. 2546. (7 ตุลาคม). บทความไอที. (Online). Available URL: http://www.nectec.or.th/courseware/computer/articles/0007.html

สำนักบริการคอมพิวเตอร์ มหาวิทยาลัยเกษตรศาสตร์. 2540. (10 มกราคม). ไม้ประดับ. (Online). Available URL: http://www.doae.go.th/library/

Bell, T. C., Cleary, J. G. and Witten, I. H. 1990. **Text Compression**. Englewood Cliffs, NJ: Prentice Hall.

Cleary, J.G. and Teahan, W.J. 1995. **Experiments on the zero frequency problem**. IEEE Transactions on Communication: 1-10.

Cleary, J.G. and Witten, I.H. 1984. **Data compression using adaptive coding and partial string matching**. IEEE Transactions on Communications: 396-402.

Sayood, K. 2000. **Introduction to Data Compression** 2nd ed. San Fran-cisco, California: Academic Press.

Lynch, T.J. 1985. **Data Compression:Techniques and Application**. Van Nostrand Reinhold Company: 3-14.

Mark Allen Weiss. 1999. **Data Structures & Algorithm analysis in C++: Algorithm Design Techniques** 2nd ed. Florida: Addison Wesley Longman.

Moffat, A. 1990. **Implementin the PPM data compression scheme**. IEEE Transactions on Communication: 1917-1921.

Witten, I.H., Moffat, A. and Bell, T.C. 1991. **The Zero-Frequency Problem: Estimating the Probabilities of Novel Events in Adaptive Text Compression**. IEEE Transactions on Information Theory: 1085 -1094.

_____. 1999. **Managing Gigabytes-Compressing and Indexing Documents and Images**. 2nd ed. San Fran-cisco, California: Morgan Kaufmann Press.

ประวัติผู้ทำวิทยานิพนธ์

ชื่อ-นามสกุล	น.ส. สุรีวรรณ แจ้งจิตร
ประวัติการศึกษา	ปริญญาตรี หลักสูตรวิทยาศาสตรบัณฑิต สาขาวิทยาการคอมพิวเตอร์ คณะคอมพิวเตอร์และเทคโนโลยีชั้นสูง สถาบันราชภัฏเพชรบุรี ปีการศึกษา 2539
ตำแหน่งและสถานที่ทำงานปัจจุบัน	อาจารย์ สาขาวิชาคอมพิวเตอร์ คณะวิทยาศาสตร์และเทคโนโลยี สถาบันราชภัฏกาญจนบุรี