การจัดกลุ่มเอกสารข้อความภาษาไทยด้วยขั้นตอนวิธี Spherical K-Means แบบขนานบนพิรุณลีนุกซ์คลัสเตอร์

Thai Text Document Clustering using Parallel Spherical K-Means Algorithm on PIRUN Linux Cluster

ฒนาศัย กรึงไกร[1] และ ชุลีรัตน์ จรัสกุลชัย[2]
Canasai Kruengkrai[1] and Chuleerat Jaruskulchai[2]
Intelligent Information Retrieval and Database Laboratory
Department of Computer Science, Faculty of Science
Kasetsart University, Bangkok, 10900, Thailand
E-mail: g4364115[1], fscichj[2]@ku.ac.th

บทคัดย่อ

การจัดกลุ่มเอกสาร คือการแยกเอกสารออกเป็นกลุ่มตามความคล้ายคลึงและความสัมพันธ์กัน ขั้นตอนวิธีแบบลำดับหลายวิธีได้ถูกพัฒนาเพื่อจัดการกับปัญหาการจัดกลุ่มเอกสาร อย่างไรก็ตาม ขั้นตอนวิธีเหล่านี้ให้ผลลัพธ์ที่ช้าเกินไป เมื่อนำมาใช้กับชุดเอกสารขนาดใหญ่ ในงานวิจัยฉบับนี้ เรานำเสนอวิธีการแบบขนานเพื่อจัดกลุ่มเอกสาร โดยมีพื้นฐานอยู่บนขั้นตอนวิธี spherical k-means เราทำการทดลองบนพิรุณคลัสเตอร์ซึ่งเป็นคอมพิวเตอร์แบบขนาน โดยใช้เทคโนโลยีการประมวลผลระบบคลัสเตอร์และใช้ลีนุกซ์เป็นระบบปฏิบัติการ ชุดข้อมูลประกอบด้วยเอกสารข้อความภาษาไทยซึ่งนำมาจากข่าวหนังสือพิมพ์จำนวน 4,800 ข่าว ผลการทดลองแสดงว่า การใช้ขั้นตอนวิธีแบบขนานสามารถเพิ่มประสิทธิภาพการจัดกลุ่มได้อย่างมาก และเรายังพบว่าขั้นตอนวิธีของเรายังมีประสิทธิภาพดีเมื่อขนาดของปัญหาเพิ่มขึ้นอีกด้วย

Abstract

Document clustering is the process of grouping similar or related documents into classes. Many sequential algorithms have been developed to deal with document clustering problems. However, these algorithms are too slow when are applied to large document collections. In this paper, we propose a parallel algorithm for clustering text documents based on spherical k-means. We implemented our algorithm on the PIRUN Linux Cluster, which is a parallel computer using cluster computing technology. The data set consists of 4,800 articles taken from a Thai newspaper. Experimental results show that the use of parallel algorithm can significantly improve clustering performance. Furthermore, we find that our algorithm is also effective when the problem size is scaled up.

1. บทนำ

ปัจจุบันการจัดเก็บเอกสารในรูปแบบของสื่ออิเล็กทรอนิกส์มีแนวโน้มที่เพิ่มมากขึ้นเรื่อยๆ ในเว็บเซิร์ฟเวอร์หนึ่งอาจบรรจุหน้าเอกสารจำนวนหลายพันหน้า หรืออาจมากถึงหลายล้านหน้าเช่น เว็บเสิร์จเอ็นจิน การแยกเอกสารเหล่านี้ออกเป็นประเภทต่างๆด้วยมนุษย์ไม่ทันต่อปริมาณของเอกสารใหม่ๆที่เกิดขึ้น ในศาสตร์การค้นคืนสารสนเทศ (Information Retrieval) ได้วิจัยปัญหานี้ในหัวข้อของการจัดกลุ่มเอกสาร (Document Clustering) ซึ่งพยายามคิดค้นวิธีที่จะจัดกลุ่มเอกสารปริมาณมากๆแบบอัตโนมัติ

การจัดกลุ่มเอกสารมีจุดประสงค์คือ แยกเอกสารออกเป็นกลุ่มตามความคล้ายคลึงและความสัมพันธ์กันซึ่งขึ้นอยู่กับข้อ

ความที่ปรากฏในเอกสารแต่ละฉบับ การดำเนินการของการจัดกลุ่มเอกสาร แบ่งได้สองประเภทคือ โกลบอล (global) และโลคอล (local) ในการจัดกลุ่มแบบโกลบอล เอกสารถูกจัดกลุ่มตามที่ปรากฏอยู่ในชุดเอกสารทั้งหมด ส่วนในการจัดกลุ่มแบบโลคอล เอกสารถูกจัดกลุ่มตามเซตของเอกสารที่ค้นคืนมาได้จากคำขอ (query) ปัจจุบัน [1] ในงานวิจัยนี้จะเน้นเฉพาะการจัดกลุ่มแบบโกลบอล

งานวิจัยในต่างประเทศได้พัฒนาขั้นตอนวิธีการจัดกลุ่มเอกสารมาอย่างต่อเนื่อง ถึงแม้ว่าจะมีขั้นตอนวิธีที่ให้ผลลัพธ์ได้ในเวลาที่รวดเร็ว (เวลาเกือบจะเป็น linear) แต่เมื่อนำมาปฏิบัติจริงกับชุดเอกสารข้อความเต็ม (full text) จำนวนมากๆ ก็ยังไม่สามารถให้ผลลัพธ์ได้ในเวลาที่เหมาะสม

ในงานวิจัยนี้เราจึงนำเสนอขั้นตอนวิธีการจัดกลุ่มแบบขนาน โดยมีพื้นฐานอยู่บนขั้นตอนวิธี spherical k-means [4] ซึ่งมีจุดมุ่งหมายเพื่อลดเวลาการประมวลผล แนวคิดคือ ให้เอกสารทั้งหมดถูกแบ่งออกไปประมวลผลเป็นชุดย่อยๆ และสามารถถูกจัดกลุ่มในเวลาเดียวกันได้อย่างอิสระ เปรียบได้กับให้นักวิจัยคนหนึ่งอ่านรายงานวิจัยทีละฉบับ หลังจากนั้นแยกรายงานออกเป็นกลุ่มตามเนื้อหาที่คล้ายคลึงกัน กับให้นักวิจัยหลายคนซึ่งมีความรู้เท่ากันช่วยกันอ่านและช่วยกันแยกรายงานออกเป็นกลุ่มย่อมทำงานได้เร็วกว่าและได้ปริมาณงานที่มากกว่า

งานวิจัยฉบับนี้จะนำเสนอความรู้เบื้องต้นและงานวิจัยที่เกี่ยวข้องของการจัดกลุ่มเอกสารในหัวข้อที่สอง หัวข้อที่สามนำเสนอรายละเอียดของขั้นตอนวิธี spherical k-means แบบลำดับและแบบขนาน หัวข้อที่สี่นำเสนอวิธีการทดลอง ผลการทดลองและการวิเคราะห์ ในหัวข้อที่ห้าสรุปเนื้อหาของงานวิจัย

2. ความรู้เบื้องต้นและงานวิจัยที่เกี่ยวข้อง

2.1. การแทนเอกสารด้วย Vector Space Model (VSM)

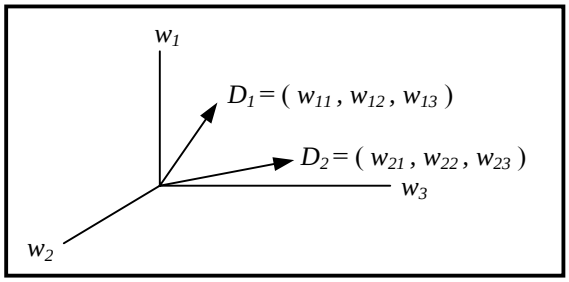
ในส่วนนี้จะอธิบายถึงวิธีการแทนเอกสารที่ไม่มีโครงสร้าง (unstructured text document) ด้วย VSM [10] ใน VSM เอกสารแต่ละฉบับ เปรียบเสมือนกับเวคเตอร์ของคำ โดยที่ขนาดของเวคเตอร์ขึ้นอยู่กับจำนวนของคำที่ปรากฏอยู่ในเอกสารฉบับนั้น กำหนดให้ w_{ik} คือค่าน้ำหนักของคำ k ที่ปรากฏในเอกสาร i เวคเตอร์สำหรับเอกสาร D_i สามารถเขียนแทนด้วย $D_i = (w_{i1}, w_{i2},\ldots, w_{it})$ ซึ่ง t คือจำนวนของคำที่ไม่ซ้ำกันในชุดเอกสารทั้งหมด ดังนั้นใน space ของชุดเอกสารหนึ่งจะมีมิติเท่ากับ t-มิติ เช่นในรูปที่ 1 แสดงเวคเตอร์เอกสารใน 3-มิติ

โดยคุณสมบัติของเวคเตอร์ทำให้สามารถคำนวณค่าความคล้ายคลึง (similarity) ของเอกสารคู่หนึ่งๆได้จากค่าสัมประสิทธิ์ cosine ของมุมระหว่างคู่ของเวคเตอร์ซึ่งจะมีค่าอยู่ระหว่าง 0 ถึง 1 มีสูตรดังนี้

$$\mathrm{sim}(D_i , D_j) = \frac{D_i \bullet D_j}{\|D_i\| \times \|D_j\|} \qquad (1)$$

2.2. การให้น้ำหนักคำ (Term Weighting)

คำที่มีความสำคัญหรือใช้เป็นตัวแทนของเอกสารที่ดีควรจะปรากฏอยู่เป็นจำนวนมากในเนื้อหาของเอกสารเฉพาะฉบับนั้น และปรากฏอยู่น้อยมากในชุดเอกสารที่เหลือทั้งหมด แต่ถ้าคำนั้นปรากฏเป็นจำนวนมากในทุกๆเอกสาร แสดงว่าคำดังกล่าวไม่สามารถเป็นตัวแทนของเอกสารใดๆได้ ซึ่งคำเหล่านี้เรียกว่า คำหยุด (stopwords) เช่นคำว่า “ที่” “และ” “ซึ่ง” เป็นต้น ดังนั้นการให้น้ำหนักคำๆหนึ่งในเอกสารฉบับหนึ่งจะพิจารณาจากความถี่ของคำ (term frequency) ที่ปรากฏในเอกสารนั้นและจำนวนของเอกสารทั้งหมดที่มีคำๆนั้นปรากฏอยู่



รูปที่ 1. เวคเตอร์เอกสารใน 3-มิติ

วิธีการให้น้ำของคำที่ใช้กันอย่างมากในการสืบค้นข้อมูล คือ $tf \times idf$ (term frequency × inverse document frequency) โดยที่ค่า idf คำนวณจากค่า $\log(N / df)$ ซึ่ง N คือจำนวนเอกสารในชุดเอกสารทั้งหมดและ df คือจำนวนเอกสารที่มีคำๆ นั้นปรากฏอยู่ วิธีการให้น้ำหนักของคำใน [11] มีการ normalization ทำให้เวคเตอร์เอกสารมีขนาด 1 หน่วยมีสูตรดังนี้

$$w_{ik} = \frac{tf_{ik} \times \log(N / df_k)}{\sqrt{\sum_{j=1}^{t} (tf_{ij})^2 \times (\log(N / df_j))^2}} \qquad (2)$$

โดยที่ tf_{ik} คือความถี่ของคำ k ในเอกสาร i, N คือจำนวนของเอกสารในชุดเอกสาร, df_k คือจำนวนของเอกสารในชุดเอกสารซึ่งบรรจุคำ k จากสูตร (1) เราจึงหาค่าความคล้ายคลึงของคู่เอกสารได้จาก inner product เนื่องจาก $\|D_i\| = \|D_j\| = 1$ ดังนั้น

$$\mathrm{sim}(D_i, D_j) = D_i \bullet D_j = \sum_{k=1}^{t} (w_{i,k} \times w_{j,k}) \qquad (3)$$

ตัวอย่างเช่น ในชุดเอกสารหนึ่งประกอบด้วยเอกสาร D_1, D_2, D_3 เอกสารแต่ละฉบับหลังผ่านการตัดคำ (word segmentation) และดึงคำหยุดออกไปแล้วมีลักษณะตามรูปที่ 2 จากนั้นหาความถี่ของคำที่ไม่ซ้ำกันในเอกสารแต่ละฉบับ ตามตารางที่ 1 แล้วจึงทำการหาคำที่ไม่ซ้ำกันทั้งหมดในชุดเอกสารและกำหนดค่า df และ idf ให้แต่ละคำตามตารางที่ 2

เวคเตอร์เอกสารทั้งหมดแทนด้วยเมทริกซ์หนึ่งตามรูปที่ 3 โดยที่แถวของเมทริกซ์คือ เอกสารทั้งหมด และสดมภ์คือคำที่ไม่ซ้ำกันทั้งหมดในชุดเอกสาร ถ้าคำในสดมภ์ปรากฏอยู่ในเวคเตอร์เอกสารฉบับหนึ่งๆให้ค่าน้ำหนักตามสูตร (2) แต่ถ้าไม่ปรากฏคำนั้นในเอกสารที่กำลังพิจารณาอยู่ ก็ให้คำนั้นมีค่าน้ำหนักเป็น 0 ตอนนี้จากเอกสารที่ไม่มีโครงสร้างสามารถถูกแทนได้อย่างเป็นระบบด้วย VSM ซึ่งอยู่ในรูปของเมทริกซ์เอกสาร-คำ ซึ่งจะใช้เป็น input ในขั้นตอนการจัดกลุ่มเอกสารต่อไป และเราสามารถหาค่าความคล้ายคลึงของเอกสารคู่หนึ่งๆได้จากสูตร (3) เช่น

$$\begin{aligned}\mathrm{sim}(D_1, D_3) &= (0.57)(0) + (0)(0) + (0.29)(0.35) \\ &\quad + (0)(0.94) + (0.77)(0) \\ &= 0.1015\end{aligned}$$

D_1: คอมพิวเตอร์ สารสนเทศ คอมพิวเตอร์ คอมพิวเตอร์
D_2: อินเตอร์เน็ต คอมพิวเตอร์ อินเตอร์เน็ต ข้อมูล
D_3: ระบบ อินเตอร์เน็ต

รูปที่ 2. ชุดเอกสาร

ตารางที่ 1. ความถี่ของคำในชุดเอกสาร

เอกสาร	คำ	*tf*
D_1	คอมพิวเตอร์	3
D_1	สารสนเทศ	1
D_2	อินเตอร์เน็ต	2
D_2	คอมพิวเตอร์	1
D_2	ข้อมูล	1
D_3	ระบบ	1
D_3	อินเตอร์เน็ต	1

ตารางที่ 2. ค่า *idf* ของคำในชุดเอกสาร

คำ	*df*	*idf*
T_1: คอมพิวเตอร์	2	0.18
T_2: สารสนเทศ	1	0.48
T_3: อินเตอร์เน็ต	2	0.18
T_4: ระบบ	1	0.48
T_5: ข้อมูล	1	0.48

$$\begin{array}{c} \\ D_1 \\ D_2 \\ D_3 \end{array} \begin{array}{c} \begin{array}{ccccc} T_1 & T_2 & T_3 & T_4 & T_5 \end{array} \\ \begin{bmatrix} 0.74 & 0.67 & 0 & 0 & 0 \\ 0.57 & 0 & 0.29 & 0 & 0.77 \\ 0 & 0 & 0.35 & 0.94 & 0 \end{bmatrix} \end{array}$$

รูปที่ 3. เมทริกซ์เอกสาร-คำ

2.3. งานวิจัยที่เกี่ยวข้อง

ขั้นตอนวิธีการจัดกลุ่มสามารถแบ่งได้เป็น 2 ขั้นตอนวิธีใหญ่ๆคือ hierarchical และ nonhierarchical (หรือ partitioning) รายละเอียดสามารถดูได้ใน [8] ขั้นตอนวิธี Agglomerative Hierarchical Clustering (AHC) เป็นวิธีแบบ hierarchical ที่ใช้กันมากซึ่งสามารถแบ่งเป็นวิธีย่อยได้อีก 3 วิธีคือ single-link, group average-link และ complete-link ขึ้นอยู่กับการนิยามระยะทางระหว่างกลุ่ม ขั้นตอนวิธี AHC มีค่าความซับซ้อนของเวลา (time complexity) เป็น quadratic เมื่อนำมาใช้กับเอกสารปริมาณมากๆจะใช้เวลาประมวลผลสูง ข้อได้เปรียบของวิธี nonhierarchical คือการประมวลผลมีค่าความซับซ้อนของเวลาต่ำกว่าคือเกือบจะเป็น linear วิธีที่ใช้ทั่วไปได้แก่ k-means, single-pass, buckshot และ fractionation [2]

ใน [4] ได้นำเสนอขั้นตอนวิธีการจัดกลุ่มเอกสารที่เรียกว่า spherical k-means โดยประยุกต์จากขั้นตอนวิธี k-means ซึ่งปกติวัดระยะทางระหว่างคู่ของเวคเตอร์ใดๆโดยใช้ระยะทาง Euclidean แต่ใช้ค่าสัมประสิทธิ์ cosine แทน แนวคิดในการจัดกลุ่มเอกสารคือ หาเวคเตอร์ที่เป็นตัวแทนของกลุ่มเวคเตอร์เอกสารแต่ละกลุ่ม ให้เวคเตอร์เอกสารทั้งหมดเปรียบเทียบกับเวคเตอร์เหล่านี้เพื่อหากลุ่มที่อยู่ใกล้ที่สุดจนกว่าจะพบเงื่อนไขการหยุด รายละเอียดของขั้นตอนวิธีนี้จะกล่าวในส่วนที่ 3.1

งานวิจัยการจัดกลุ่มเอกสารโดยใช้การประมวลผลแบบขนาน ได้มุ่งเน้นการลดเวลาการประมวลผลเพื่อให้สามารถจัดกลุ่มเอกสารจำนวนมากให้ได้ผลลัพธ์ในเวลาที่ไม่มากเกินไป ใน [9] ได้นำขั้นตอนวิธี single-pass และ single-link มาประมวลผลแบบขนาน ในการทดลองใช้โมเดล multiple instruction multiple data (MIMD) และประมวลผลบนเครื่อง Intel Paragon

ขั้นตอนวิธี k-means ได้มีนักวิจัยนำเสนอวิธีการประมวลผลแบบขนานไว้เช่นเดียวกัน โดยใน [6] ได้เสนอการประมวลผลแบบขนานโดยใช้โมเดล single program multiple data (SPMD) ในการทดลองทำบนเครือข่ายของ workstations (network of workstations: NOWs) ลักษณะขั้นตอนวิธีเป็นแบบ master-slave กล่าวคือ โพรเซสเซอร์หลักหรือ master โพรเซสเซอร์แบ่งชุดข้อมูลออกเป็นสับเซตย่อยเท่ากับจำนวนกลุ่มที่กำหนด แล้วส่งแต่ละสับเซตให้กับ slave โพรเซสเซอร์ จากนั้น slave โพรเซสเซอร์ประมวลผลตามขั้นตอนจนกว่าการเคลื่อนย้ายของข้อมูลในแต่ละสับเซตจะคงที่ ข้อจำกัดของวิธีการนี้คือจำนวน slave โพรเซสเซอร์จำเป็นต้องเท่ากับจำนวนกลุ่มที่ต้องการ ใน [3] ได้เสนอการประมวลผลแบบขนานของขั้นตอนวิธี k-means ใช้โมเดล SPMD ทำการทดลองบนเครื่อง IBM SP2 ลักษณะขั้นตอนวิธีแตกต่างจากงานวิจัยที่ได้กล่าวมาแล้ว โดยแบ่งข้อมูลออกเป็นชุดย่อยๆตามจำนวนโพรเซสซึ่งจำนวนของโพรเซสไม่ขึ้นอยู่กับจำนวนกลุ่มที่ต้องการ ทำให้สามารถเพิ่มจำนวนโพรเซสเซอร์ได้อย่างอิสระ อย่างไรก็ตามขั้นตอนวิธีจัดกลุ่มแบบขนานทั้งสองไม่ได้ทำการทดลองกับข้อมูลที่เป็นเอกสาร

งานวิจัยนี้นำเสนอขั้นตอนวิธีการจัดกลุ่มเอกสารแบบขนานซึ่งมีพื้นฐานอยู่บนขั้นตอนวิธี spherical k-means โดยใช้โมเดล SPMD เขียนโปรแกรมด้วยภาษา C และใช้ MPI (Message Passing Interface) ซึ่งเป็นข้อกำหนดมาตรฐานสำหรับไลบรารีของฟังก์ชัน message-passing ในการทดลองเราใช้ข้อมูลที่เป็นเอกสารข้อความภาษาไทยซึ่งแทนด้วย VSM และประมวลผลบนพิรุณคลัสเตอร์ (PIRUN: Pile of Inexpensive and Redundant Universal Nodes) ของมหาวิทยาลัยเกษตรศาสตร์ [14] ซึ่งเป็นเครื่องคอมพิวเตอร์แบบขนาน (parallel computer) ใช้เทคโนโลยีการประมวลผลระบบคลัสเตอร์ (cluster computing) โดยนำเครื่องพีซี (personal computer: PC) มาเชื่อมต่อกันด้วยเครือข่ายความเร็วสูง มีจำนวนเครื่องทั้งหมด 72 โหนดและสามารถให้ประสิทธิภาพในการประมวลผลเทียบเท่าเครื่องซูปเปอร์คอมพิวเตอร์เครื่องหนึ่ง

3. ขั้นตอนวิธีการจัดกลุ่มเอกสาร

3.1. ขั้นตอนวิธี Spherical k-means แบบลำดับ

นิยามสัญลักษณ์ที่ใช้ในขั้นตอนวิธีดังนี้ กำหนดให้ x_i แทนเวคเตอร์เอกสาร π_j แทนส่วนของเวคเตอร์เอกสารโดยที่

$$\bigcup_{j=1}^{k} \pi_j = \{ x_1, x_2, \ldots, x_n \} \text{ และ } \pi_j \cap \pi_l = \phi \text{ ถ้า } j \neq l \quad (4)$$

ทำการวัดคุณภาพของ π_j , $1 \leq j \leq k$ โดยใช้ฟังก์ชัน objective ตามสมการข้างล่างนี้

$$Q(\pi_j) = \sum_{j=1} \sum_{x \in \pi_j} x^T c_j \quad (5)$$

โดยที่ $x^T c_j$ แทน inner product ระหว่างเวคเตอร์เอกสารและเวคเตอร์ concept $c_j = \mathrm{m}_j / \|\mathrm{m}_j\|$, m_j คือเวคเตอร์เฉลี่ย (centroid) ของเวคเตอร์เอกสารที่บรรจุอยู่ใน π_j ขั้นตอนวิธี spherical k-means แสดงตามตารางที่ 3

ตารางที่ 3. ขั้นตอนวิธี Spherical k-means แบบลำดับ

1. เลือกจำนวนกลุ่ม k แล้วแบ่งเวคเตอร์เอกสารอย่างสุ่มออกเป็น $\pi_j^{(0)}$, $1 \leq j \leq k$ ให้ $c_j^{(0)}$ แทนเวคเตอร์ concept ที่สอดคล้องกับส่วนของเวคเตอร์ที่กำหนด และกำหนดค่าของ iteration $t = 0$
2. สำหรับแต่ละเวคเตอร์เอกสาร x_i หาเวคเตอร์ concept ที่อยู่ใกล้ x_i มากที่สุด และคำนวณ $\pi_j^{(t+1)}$ ใหม่ซึ่งถูกกำหนดโดย $c_j^{(t)}$ โดยที่
$$\pi_j^{(t+1)} = \{x \in x_i, 1 \leq i \leq n: x^T c_j^{(t)} \geq x^T c_l^{(t)}, 1 \leq l \leq n \}, 1 \leq j \leq k \quad (6)$$
3. คำนวณเวคเตอร์ concept $c_j^{(t+1)}$ ซึ่งสอดคล้องกับ $\pi_j^{(t+1)}$ ที่ได้จากขั้นตอนที่ 2
4. ถ้าฟังก์ชัน objective เปลี่ยนแปลงน้อยกว่าค่ากำหนด (threshold) แล้วจบการทำงาน ไม่เช่นนั้นเพิ่มค่า t ด้วย 1 และย้อนกลับไปขั้นตอนที่ 2

3.2. ตอนวิธี Spherical k-means แบบขนาน

ในส่วนนี้จะกล่าวถึงการนำขั้นตอนวิธี spherical k-means แบบลำดับมาทำการประมวลผลแบบขนาน โดยแบ่งเวคเตอร์เอกสารออกเป็นชุดย่อยๆให้แต่ละโพรเซสเซอร์จำนวนเท่าๆกัน

กำหนดให้ P แทนจำนวนโพรเซสเซอร์ทั้งหมดและ n แทนจำนวนเวคเตอร์เอกสารทั้งหมด แต่ละโพรเซสเซอร์ p_l , $0 \leq l \leq$ P-1 รับผิดชอบเวคเตอร์เอกสารจำนวน n/P โดยที่โพรเซสเซอร์ p_l จะมีเวคเตอร์เอกสาร x_i , $(l)(n/\mathrm{P}) + 1 \leq i \leq (l + 1)(n/\mathrm{P})$ ในการประมวลผลแต่ละโพรเซสเซอร์ p_l สามารถอ่านเวคเตอร์เอกสารจากฮาร์ดดิสก์เพียงครั้งเดียว เนื่องจากขนาดของข้อมูลลดลงจนเหมาะกับความจุของหน่วยความจำของแต่ละโพรเซสเซอร์

สำหรับการประมวลผลแบบลำดับในส่วนที่ 3.1 ขั้นตอนที่ใช้เวลาประมวลผลมากที่สุดคือ การหาเวคเตอร์ concept c_j , $1 \leq j \leq k$ ที่อยู่ใกล้เวคเตอร์เอกสาร x_i , $1 \leq i \leq n$ มากที่สุดในขั้นตอนที่ 2 เพราะต้องคำนวณ inner product ระหว่างเวคเตอร์เอกสารและเวคเตอร์ concept ทั้งหมดเป็นจำนวน kn ครั้ง แต่เนื่องจากการคำนวณแต่ละครั้งกระทำกับข้อมูลที่ต่างกันและเป็นอิสระต่อกัน ถ้าให้แต่ละโพรเซสเซอร์ p_l มีเวคเตอร์ concept c_j , $1 \leq j \leq k$ เหมือนกัน จะสามารถหาเวคเตอร์ concept c_j ที่อยู่ใกล้ที่สุดในเวลาเดียวกันได้ ดังนั้นแต่ละโพรเซสเซอร์จะทำการคำนวณลดลงเหลือ $k(n/\mathrm{P})$ ครั้ง

รูปที่ 4 (ก) แสดงการหาเวคเตอร์ concept ที่อยู่ใกล้ที่สุดแบบลำดับสำหรับ 15 เวคเตอร์เอกสารและกลุ่ม 3 กลุ่ม รูปที่ 4 (ข) แสดงการแบ่งเวคเตอร์เอกสารให้ 3 โพรเซสเซอร์ช่วยกันทำงาน แต่ละโพรเซสเซอร์จะทำงานน้อยลงและสามารถทำไปพร้อมๆกันได้

ในระหว่างการประมวลผล เวคเตอร์เอกสาร x_i จะไม่มีการเคลื่อนย้ายระหว่างโพรเซสเซอร์ โดยแต่ละเวคเตอร์เอกสาร x_i จะมีตัวแปรหนึ่งใช้เก็บตำแหน่งของกลุ่มที่อยู่ เมื่อจบการทำงานจึงส่งตัวแปรนี้กลับไปยังโพรเซสเซอร์ p_0

กำหนดให้ฟังก์ชัน CCV (compute concept vector) ทำหน้าที่ในการรวมค่าโลคอล m_j, $1 \leq j \leq k$ ในแต่ละโพรเซสเซอร์ให้เป็นโกลบอล m_j โดยเรียกใช้ฟังก์ชัน MPI_Allreduce [12] ผลลัพธ์ที่ได้จะส่งคืนกลับไปยังทุกโพรเซสเซอร์ จากนั้นแต่ละโพรเซสเซอร์ทำการ normalization ค่าโกลบอล m_j ก็จะได้ค่าเวคเตอร์ concept c_j เหมือนกัน ในรูปที่ 5 แสดงการทำงานของ

ฟังก์ชัน CCV ขั้นตอนวิธี spherical k-means แบบขนานแสดงตามตารางที่ 4

ตารางที่ 4. ขั้นตอนวิธี Spherical k-means แบบขนาน

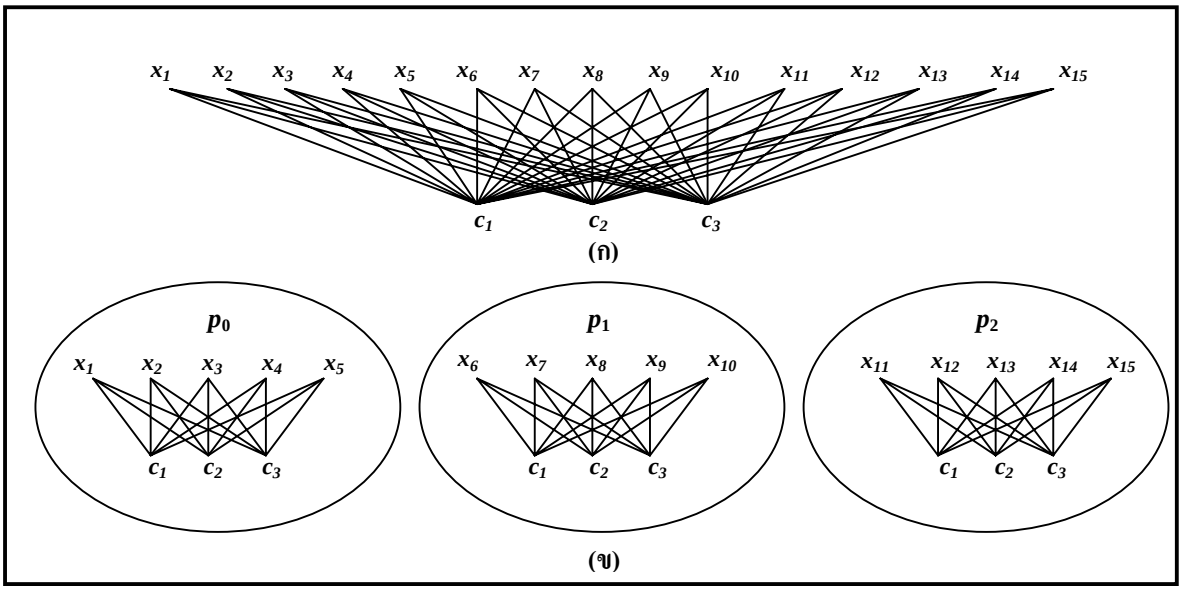
สำหรับทุกโพรเซสเซอร์ p_l, $0 \le l \le$ P-1

1. โพรเซสเซอร์ p_l อ่านเวคเตอร์เอกสาร x_i , $(l)(n/\mathrm{P}) + 1 \le i \le (l+1)(n/\mathrm{P})$ จากฮาร์ดดิสก์
2. แต่ละโพรเซสเซอร์ p_l แบ่งเวคเตอร์เอกสาร x_i ออกเป็น k กลุ่มอย่างสุ่มจากนั้นคำนวณโลคอล m_j แล้วเรียกฟังก์ชัน CCV จะได้เวคเตอร์ concept c_j
3. โพรเซสเซอร์ p_l หาเวคเตอร์ concept c_j ที่อยู่ใกล้เวคเตอร์เอกสาร x_i มากที่สุดและคำนวณโลคอล m_j
4. แต่ละโพรเซสเซอร์ p_l คำนวณเวคเตอร์ concept c_j ใหม่โดยเรียกฟังก์ชัน CCV
5. ถ้าฟังก์ชัน objective เปลี่ยนแปลงน้อยกว่าค่ากำหนดให้ส่งตำแหน่งของกลุ่มที่เวคเตอร์เอกสาร x_i อยู่ไปยังโพรเซสเซอร์ p_0 แล้วจบการทำงาน ไม่เช่นนั้นให้ย้อนกลับไปขั้นตอนที่ 3

4. การทดลองและผลการทดลอง

4.1. การเตรียมข้อมูลและการกำหนดสภาพแวดล้อมการทดลอง

เอกสารที่ใช้ในการทดลองนำมาจากหนังสือพิมพ์เดลินิวส์ ตั้งแต่ 1 ก.ย.- 22 พ.ย. 2538 จำนวน 4,800 ข่าวโดยมีข่าว 5 ชนิด คือ เศรษฐกิจ (1,146 ข่าว) ต่างประเทศ (1,653 ข่าว) การเมือง (828 ข่าว) พระราชสำนัก (47 ข่าว) สังคม (1,126 ข่าว) ซึ่งเรากำหนดให้ 1 ข่าวแทน 1 เอกสาร ในขั้นตอนแรกของการเตรียมข้อมูลจะต้องมีการแยกเอกสารออกเป็นคำๆ ถ้าเป็นเอกสารภาษาอังกฤษจะทำได้สะดวกเพราะมีการเว้นวรรคอยู่แล้ว แต่ภาษาไทยมีการเขียนในลักษณะที่ติดต่อกันโดยไม่มีการใช้เครื่องหมายวรรคตอนใดๆ มีการเว้นวรรคซึ่งก็ไม่ได้มีกฎเกณฑ์แน่นอน การตัดคำในภาษาไทยจึงเป็นเรื่องที่สำคัญมากเรื่องหนึ่ง ดังนั้นเราใช้วิธีการตัดคำ 2 วิธีคือ วิธีกฎทางภาษาศาสตร์ [5] และวิธี Longest Matching (LM) [13] ซึ่งใช้รายการคำไทยจำนวน 32,675 คำ เราจึงได้ชุดเอกสารทดสอบ 2 ชุด



รูปที่ 4. (ก) แสดงการหาเวคเตอร์ concept ที่อยู่ใกล้ที่สุดแบบลำดับ, (ข) แสดงการแบ่งเวคเตอร์เอกสารให้ 3 โพรเซสเซอร์

เพื่อไม่ให้ VSM มีมิติมากเกินไปเราจึงดึงคำหยุดออกซึ่ง [5] ได้รวบรวมไว้และดึงคำที่ปรากฏในเอกสารน้อยกว่า 0.2% ของเอกสารทั้งหมดออก สุดท้ายในชุดเอกสารที่ 1 (กฎทางภาษาศาสตร์) มีคำที่ไม่ซ้ำกัน 3,627 คำ และชุดเอกสารที่ 2 (LM) มีคำที่ไม่ซ้ำกัน 4,204 คำ ดังนั้น VSM จะถูกแทนด้วยเมทริกซ์เอกสาร-คำขนาด 4,800×3,627 และ 4,800×4,204 สำหรับชุดเอกสารที่ 1 และ 2 ตามลำดับ

งานวิจัยนี้เราทำการประมวลผลบนพิรุณคลัสเตอร์ซึ่งมี 72 โหนด แต่ละโหนดประกอบด้วย โพรเซสเซอร์ Pentium III 500 MHz หน่วยความจำ SDRAM PC-100 128 MB ไม่มีฮาร์ดดิสก์ในตัว NIC SMC 1211TX 100Base-TX Ethernet ใช้ Fast Ethernet Switch 3COM SuperStack II 3300 เป็น interconnection network และใช้ลีนุกซ์เป็นซอฟต์แวร์ระบบปฏิบัติการ รายละเอียดสถาปัตยกรรมของพิรุณ คลัสเตอร์ดูได้ใน [14]

4.2. ผลการทดลอง

- **ประสิทธิผลของการจัดกลุ่มเอกสาร**

เราวัดผลของการจัดกลุ่มเอกสารด้วย F-measure [7] ซึ่งเป็นค่าที่รวมเอาค่าความแม่นยำ (precision: P) และค่าความระลึก (recall: R) ไว้ในค่าเดียว เรากำหนดให้ชนิดของข่าวที่มีจำนวนมากที่สุดในกลุ่มใดๆเป็นหัวข้อเรื่อง (topic) มีสูตรดังนี้

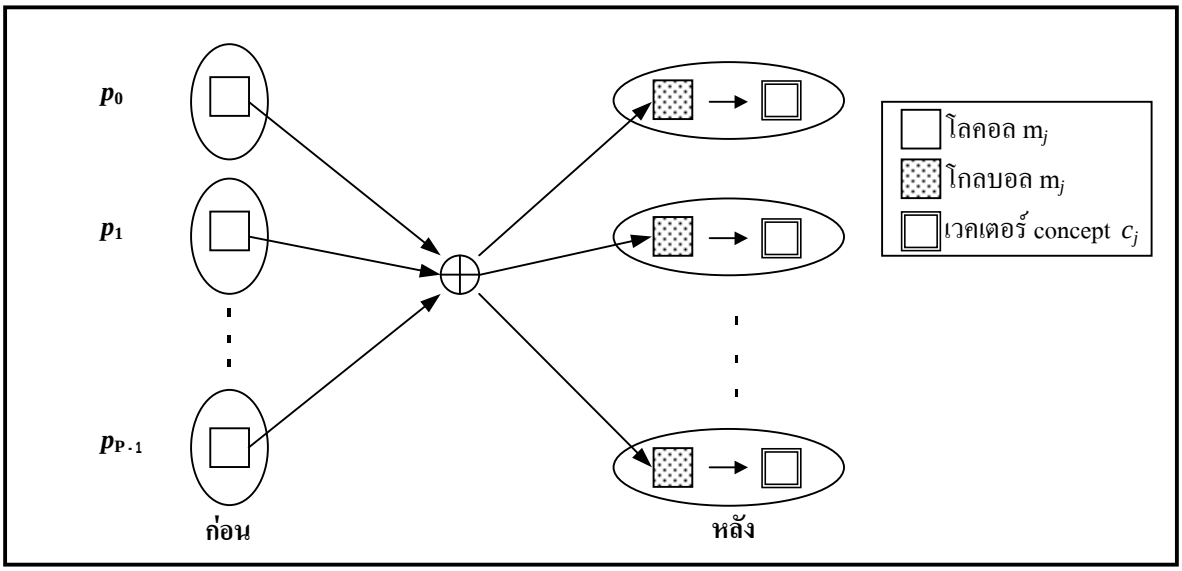
$$P_{i,t} = \frac{\text{จำนวนเอกสารที่เป็นหัวข้อเรื่อง } t \text{ ในกลุ่ม } i}{\text{จำนวนเอกสารในกลุ่ม } i} \quad (7)$$

$$R_{i,t} = \frac{\text{จำนวนเอกสารที่เป็นหัวข้อเรื่อง } t \text{ ในกลุ่ม } i}{\text{จำนวนเอกสารหัวข้อเรื่อง } t \text{ ในชุดเอกสาร}} \quad (8)$$

$$F_{i,t} = \frac{2(P_{i,t} R_{i,t})}{P_{i,t} + R_{i,t}} \quad (9)$$

$$F_{total} = \frac{\sum_{t \in T} (|t| \, max_i F_{i,t})}{\sum_{t \in T} |t|} \quad (10)$$

โดยที่ T คือเซตของหัวข้อเรื่อง, $|t|$ คือจำนวนเอกสารหัวข้อเรื่อง t ในการทดลองเรากำหนด $K = 5$ เพื่อให้เท่ากับจำนวนหัวข้อข่าวที่มี และกำหนดว่าถ้าฟังก์ชัน objective เปลี่ยนแปลงน้อยกว่า 0.01 ให้จบการทำงาน เราทำการทดสอบซ้ำเป็นจำนวน 5 ครั้งแล้วนำค่า F_{total} มาหาค่าเฉลี่ย จากการทดลองเราได้ค่า F_{total} เท่ากับ 0.65 สำหรับชุดเอกสารที่ 1 และ 0.73 สำหรับชุดเอกสารที่ 2

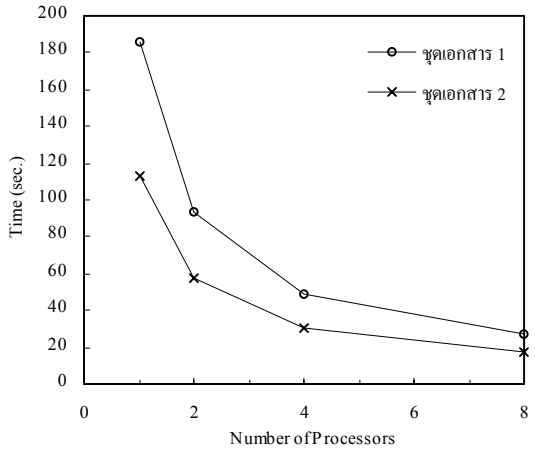


รูปที่ 5. แสดงการทำงานของฟังก์ชัน CCV โดยเรียกฟังก์ชัน MPI_Allreduce และใช้ operation MPI_SUM

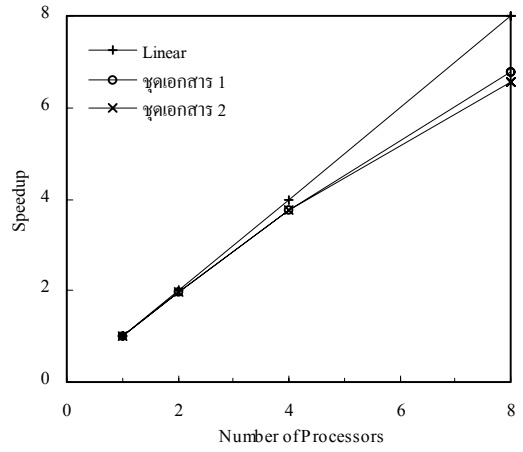
- **ประสิทธิภาพของการประมวลผลแบบขนาน**

ในการวัดเวลาเราทำการวัด 5 ครั้งจากนั้นนำค่าที่ได้มาหาค่าเฉลี่ย (ในการวัดเวลาเรารวมเวลาอ่าน/เขียนข้อมูลจากฮาร์ดดิสก์ด้วย) และคำนวณค่า speedup จากเวลาการประมวลใน 1 โพรเซสเซอร์หารด้วยเวลาการประมวลใน P โพรเซสเซอร์ โดยเราเพิ่มจำนวนโพรเซสเซอร์จนมากที่สุดที่ 8 โพรเซสเซอร์ กำหนดจำนวนกลุ่ม $K = 5$ ผลของการวัดเวลาการประมวลผลของชุดเอกสาร 1 และ 2 จำนวน 4,800 เอกสาร แสดงตามรูปที่ 6 และค่า speedup แสดงตามรูปที่ 7

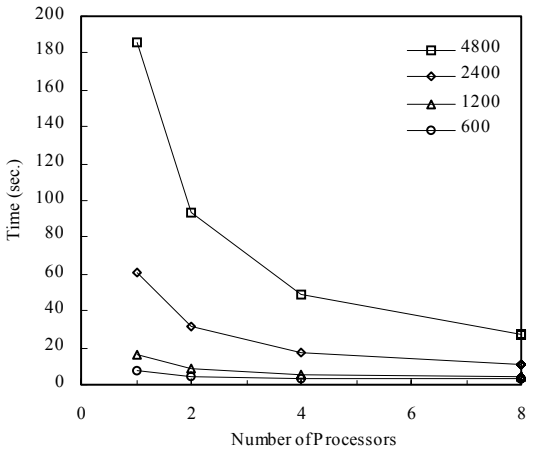
เพื่อทดสอบว่าขั้นตอนวิธี spherical k-means แบบขนานที่เราได้ออกแบบตอบสนองกับขนาดของปัญหาที่แตกต่างกันอย่างไรเราจึงนำเอกสารชุดที่ 1 มาทดสอบ โดยกำหนดมิติของเวคเตอร์เอกสารคงที่เท่ากับ 3,627 มิติและจำนวนกลุ่ม $K = 5$ ทดสอบที่จำนวน 600 1,200 2,400 และ 4800 เอกสาร ได้ผลของเวลาการประมวลผลแสดงตามรูปที่ 8 และค่า speedup ที่สัมพันธ์กันแสดงตามรูปที่ 9



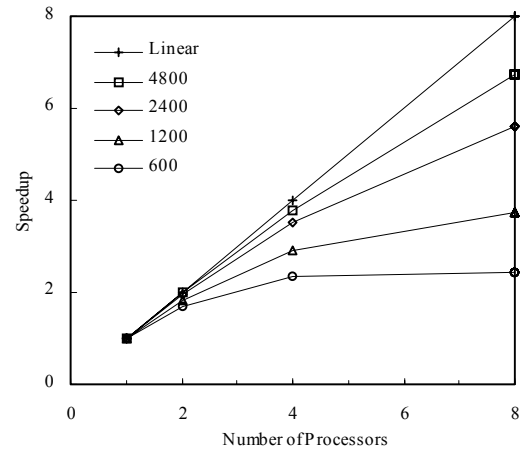
รูปที่ 6. เวลาการประมวลผลของชุดเอกสาร 1 และ 2



รูปที่ 7. ค่า Speedup สำหรับชุดเอกสาร 1 และ 2



รูปที่ 8. เวลาการประมวลผลของชุดเอกสาร 1 ในจำนวนที่แตกต่างกัน



รูปที่ 9. ค่า Speedup ของชุดเอกสาร 1 ในจำนวนที่แตกต่างกัน

4.3. วิเคราะห์

จากผลการทดลองสังเกตได้ว่า การทดสอบขั้นตอนวิธี spherical k-means แบบขนานกับชุดเอกสารที่ใช้วิธีการตัดคำที่ต่างกัน ส่งผลให้ประสิทธิผลของการจัดกลุ่มแตกต่างกัน โดยที่ชุดเอกสาร 2 ซึ่งตัดคำด้วยวิธี LM จะให้ผลที่ดีกว่าชุดเอกสาร 1 ซึ่งตัดคำด้วยวิธีกฎทางภาษาศาสตร์ เพราะการตัดคำด้วยวิธี LM มักจะให้ผลที่ถูกต้องมากกว่าวิธีกฎทางภาษาศาสตร์ และเรายังสังเกตได้ว่าผลลัพธ์ของกลุ่มที่ได้จากชุดเอกสาร 2 สามารถจัดเอกสารออกเป็นกลุ่มหลักๆตามชนิดของข่าวได้ทุกครั้ง ยกเว้นข่าวในพระราชสำนักเนื่องจากมีเป็นจำนวนน้อยเกินไป ส่วนผลลัพธ์ของกลุ่มที่ได้จากชุดเอกสาร 1 ในบางครั้งบางหัวข้อข่าวจะหายไป

สำหรับประสิทธิภาพของการประมวลผลแบบขนาน ตามรูปที่ 7 สำหรับเอกสารจำนวน 4,800 เอกสาร เมื่อเพิ่มจำนวนโพรเซสเซอร์ตั้งแต่ 2 จนถึง 4 โพรเซสเซอร์ ค่า speedup ที่ได้เกือบจะเป็นเส้นตรง (linear) และได้ค่า speedup สูงสุดเป็น 6.76 สำหรับชุดเอกสารที่ 1 และ 6.56 สำหรับชุดเอกสารที่ 2 เมื่อจำนวนโพรเซสเซอร์เท่ากับ 8 โพรเซสเซอร์

โดยทั่วไปในการประมวลผลแบบขนานสำหรับขนาดปัญหาคงที่ (fixed problem size) เมื่อเราเพิ่มจำนวนโพรเซสเซอร์เรื่อยๆ ปัญหาที่ประมวลผลต่อโพรเซสเซอร์ลดลง ทำให้อัตราส่วนของเวลาในการประมวลผลต่อเวลาในการติดต่อสื่อสารเพิ่มขึ้น ดังนั้นเมื่อถึงจุดหนึ่ง ค่า speedup ที่ได้จึงลดลง สังเกตจากในรูปที่ 9 สำหรับเอกสารจำนวน 600 เอกสาร เมื่อเพิ่มจำนวนโพรเซสเซอร์จาก 4 จนถึง 8 ค่า speedup เกือบจะไม่เปลี่ยนแปลง วิธีแก้คือ ให้ขยายขนาดของปัญหาพร้อมกับจำนวนโพรเซสเซอร์จะทำให้ค่า speedup ไม่ถูกจำกัด ในรูปที่ 9 เมื่อเราเพิ่มจำนวนเอกสารมากขึ้น ค่า speedup จะเพิ่มขึ้นเรื่อยๆ โดยสังเกตที่จำนวนโพรเซสเซอร์เท่ากับ 8 โพรเซสเซอร์ เอกสารตั้งแต่ 600 ถึง 4,800 เอกสาร มีค่า speedup เพิ่มขึ้นจาก 2.42 ถึง 6.76

5. สรุป

งานวิจัยนี้ได้นำเสนอขั้นตอนวิธีการจัดกลุ่มเอกสารแบบขนานซึ่งมีพื้นฐานอยู่บนขั้นตอนวิธี spherical k-means และทำการประมวลผลบนพิรุณคลัสเตอร์ จากผลการทดลองกับเอกสารข้อความภาษาไทยซึ่งนำมาจากข่าวหนังสือพิมพ์จำนวน 4,800 ข่าว พบว่าประสิทธิผลที่ได้จากการจัดกลุ่มดีในระดับหนึ่ง และขั้นตอนวิธีแบบขนานที่ได้นำเสนอนอกจากจะช่วยเพิ่มประสิทธิภาพในการจัดกลุ่มแล้วยังสามารถขยายจำนวนเอกสารที่ต้องการจัดกลุ่มได้อีก โดยทั่วไปเมื่อเพิ่มจำนวนเอกสารมากขึ้นเรื่อยๆ การประมวลผลบนเครื่องคอมพิวเตอร์เครื่องหนึ่งถูกจำกัดด้วยความจุของหน่วยความจำ ทำให้ต้องทำการประมวลผลสลับกับการอ่านเขียนข้อมูลจากฮาร์ดดิสก์จำนวนหลายครั้ง ซึ่งเวลาในการอ่านเขียนข้อมูลจากฮาร์ดดิสก์เป็นต้นทุนที่สูง แต่เมื่อนำมาประมวลผลแบบขนานเอกสารทั้งหมดจะถูกแบ่งออกไปประมวลผลเป็นชุดย่อยๆ ซึ่งขนาดของชุดเอกสารย่อยนี้จะเหมาะสมกับความจุของหน่วยความจำในแต่ละโพรเซสเซอร์ ด้วยเหตุนี้เราจึงสามารถจัดกลุ่มเอกสารจำนวนมากได้และอ่านข้อมูลจากฮาร์ดดิสก์เพียงครั้งเดียว

6. กิตติกรรมประกาศ

งานวิจัยนี้สำเร็จได้ เราขอขอบคุณ ผศ.ดร. ภุชงค์ อุทโยภาศ ซึ่งให้ความรู้ในการโปรแกรม MPI และแนวคิดในการวิเคราะห์ผลการทดลองแบบขนาน สำนักบริการคอมพิวเตอร์ที่ให้เราได้ทำการทดลองบนพิรุณคลัสเตอร์ และงานวิจัยนี้ได้รับการสนับสนุนจากสถาบันวิจัยและพัฒนาแห่งมหาวิทยาลัยเกษตรศาสตร์

7. เอกสารอ้างอิง

[1] Baeza-Yates, R. and B. Ribeiro-Neto. Modern Information Retrieval. The ACM Press, 1999.

[2] Cutting, D. R., D. R. Karger, J. O. Pedersen and J.W. Tukey. Scatter/Gather: A Cluster-based Approach to Browsing Large Document Collections. in the Proceedings of SIGIR'92, 1992.

[3] Dhillon, I. S. and D. S. Modha. A Data-Clustering Algorithm on Distributed Memory Multiprocessors. Large-Scale Parallel Data Mining 1999, pp. 245-260.

[4] _____. Concept Decompositions for Large Sparse Text Data using Clustering. Machine Learning, 42:1, pp. 143-175, January, 2001.

[5] Jaruskulchai, C. An Automatic Indexing for Thai Text Retrieval, Ph.D. Thesis, George Washington University, USA, Aug 31, 1998.

[6] kantabutra, S. and A. L. Couch. Parallel k-means Clustering Algorithm on NOWs. NECTEC Technical Journal, Vol. 1, No. 5, December 1999.

[7] Larsen, B. and C. Anoe. Fast and Effective Text Mining Using Linear-time Document Clustering, KDD-99, San Diego, California, 1999.

[8] Rasmussen, E. Clustering Algorithms, pp. 419-442. In W. B. Frakes and R. Baeza-Yates. Information Retrieval: Data Structures and Algorithms. Prentice Hall, 1992.

[9] Ruocco, A. S., O. Frieder. Clustering and Classification of Large Document Bases in a Parallel Environment. JASIS 48(10), pp. 932-943, 1997.

[10] Salton, G., A. Wong, C. S. Yang. A Vector Space Model for Automatic Indexing. CACM 18(11), pp. 613-620, 1975.

[11] Salton, G. and J. Allan. Selective Text Utilization and Text Traversal. In Proceedings of Hypertext '93, pp. 131-144.

[12] Snir, M., S. Otto, S. Huss-Lederman, D. Walker and J. Dongarra. MPI: The Complete Reference. The MPI Press, 1996.

[13] Sornlertlamvanich, V. Word Segmentation for Thai in Machine Translation System. Machine Translation, National Electronics and Computer Technology Center, Bangkok, pp. 50-56, 1993.

[14] http://pirun.ku.ac.th.