

ห้องสมุดคณะเทคโนโลยีสารสนเทศ พระจอมเกล้าลาดกระบัง
การจัดกลุ่มผลการสืบค้นโดยใช้อัลกอริทึมซัพไฟกทรี

WEB SEARCH RESULTS CLUSTERING
USING SUFFIX TREE ALGORITHM



โดย

ชาลี จิตรีผ่อง

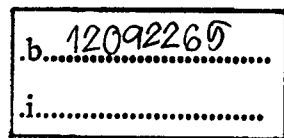
CHALEE JITTREEPHONG

อาจารย์ที่ปรึกษา

อพ.
๗52๗๓
2551

ผศ.ดร. พรฤดี เนติโสภาคกุล

เลขหมู่.....
เลขทะเบียน.....05421
วัน,เดือน,ปี.....11 ส.ย. 2552



รายงานนี้เป็นส่วนหนึ่งของวิชาโครงการพัฒนาระบบงาน
หลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ
คณะเทคโนโลยีสารสนเทศ
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ภาคเรียนที่ 1 ปีการศึกษา 2551

**WEB SEARCH RESULTS CLUSTERING
USING SUFFIX TREE ALGORITHM**

CHALEE JITTREEPHONG

**A SYSTEM DEVELOPMENT PROJECT
OF THE REQUIREMENT FOR THE DEGREE OF
MASTER OF SCIENCE PROGRAM IN INFORMATION TECHNOLOGY
FACULTY OF INFORMATION TECNOLOGY
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG**

1 / 2008

COPYRIGHT 2008

FACULTY OF INFORMATION TECHNOLOGY

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

หัวข้อ	การจัดกลุ่มผลการสืบค้นโดยใช้อัลกอริทึมซัพฟิกทรี
นักศึกษา	นายชาติ จิตรีพ้อง
รหัสนักศึกษา	49066705
ปริญญา	วิทยาศาสตรมหาบัณฑิต
สาขาวิชา	เทคโนโลยีสารสนเทศ
แขนงวิชา	วิทยาการสารสนเทศ
ปีการศึกษา	2551
อาจารย์ที่ปรึกษา	ผศ.ดร. พรฤดี เนติโสภาคกุล

บทคัดย่อ

ในปัจจุบันนี้อินเทอร์เน็ตถือเป็นเครื่องมือที่ช่วยให้ธุรกิจประสบความสำเร็จ ทำให้เกิดการสร้างเว็บเพจขึ้นเป็นจำนวนมาก และถูกจัดเก็บในเว็บโฮสติ้ง (Web hosting) ต่างๆ ทำให้เกิดความยุ่งยากและเสียเวลาในการสืบค้นข้อมูลที่ต้องการ การจัดกลุ่มผลการสืบค้นโดยใช้อัลกอริทึมซัพฟิกทรี (Web Search Results Clustering using Suffix Tree Clustering Algorithm) ทำการจัดกลุ่มข้อมูลตามความคล้ายคลึงกัน ผลของการศึกษาแสดงให้เห็นการจัดกลุ่มผลการสืบค้นโดยใช้อัลกอริทึมซัพฟิกทรี ทำให้ผู้ใช้สามารถเข้าถึงข้อมูลได้ง่ายและรวดเร็ว

Title	Web Search Results Clustering Using Suffix Tree Algorithm
Student	Mr. Chalee Jittreephong
Student ID.	49066705
Degree	Master of Science
Programme	Information Science
Academic Year	2008
Advisor	Asst.Prof. Dr. Ponrudee Netisopakul

ABSTRACT

Nowadays, Internet is a one of the emerging technologies. Many Businesses use internet for achieving their business objectives via e-business. For this reason, billions of web pages on the web provide the same idea. Therefore, it becomes critical for online businesses to show-up on search engines lists for search terms related to their products and services. However, users are forced to sift through to find relevant documents. Web Search Results Clustering using Suffix Tree Clustering Algorithm compiles results from search engine that have similar detail into same group. This paper shows that Web Search Results Clustering using Suffix Tree Clustering Algorithm to facilitate quickly access relevant information.

กิตติกรรมประกาศ

โครงการพัฒนาระบบงานฉบับนี้สำเร็จได้อย่างดี ด้วยคำแนะนำ และคำปรึกษาจาก ผศ.ดร. พรฤดี เนติโสภาคกุล ซึ่งเป็นอาจารย์ที่ปรึกษาโครงการพัฒนาระบบงาน ข้าพเจ้ารู้สึกซาบซึ้งในความอนุเคราะห์จากท่านอาจารย์ และขอขอบพระคุณเป็นอย่างสูง

ขอกราบขอบพระคุณคณาจารย์ คณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ทุก ๆ ท่านที่ได้ประสิทธิ์ประสาทวิชาให้กับข้าพเจ้า

ขอขอบคุณเพื่อนๆ พี่ๆ น้องๆ ใน คณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ทุกคนที่ให้คำแนะนำต่างๆ และคอยให้กำลังใจเสมอมา

ขอกราบขอบพระคุณ ผศ.วันเพ็ญ วสินารมณั ผศ. สุภาพร ทินประภา รศ. วสันต์ กันอ่ำ และคุณบงกชกร นิลชะชา พร้อมทั้งเพื่อนร่วมงานทุกท่านในคณะบริหารธุรกิจ มหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรี ที่ได้ให้คำปรึกษา ให้กำลังใจและให้การสนับสนุนในทุกเรื่อง

สุดท้ายนี้ข้าพเจ้าขอกราบขอบพระคุณ บิดา มารดา และครอบครัวของข้าพเจ้าที่เป็นกำลังใจ และให้การสนับสนุนในทุกเรื่อง ทำให้ข้าพเจ้าสามารถทำโครงการพัฒนาระบบงานฉบับนี้สำเร็จ ลุล่วงด้วยดี

คุณค่าและประโยชน์อันพึงมาจากโครงการพัฒนาระบบงานฉบับนี้ ข้าพเจ้าขอบแต่ผู้มีพระคุณทุกท่าน

ชาติ จิตรีผ่อง

สารบัญ

	หน้า
บทคัดย่อภาษาไทย.....	I
บทคัดย่อภาษาอังกฤษ.....	II
กิตติกรรมประกาศ.....	III
สารบัญ.....	IV
สารบัญตาราง.....	VI
สารบัญรูป.....	VII
บทที่ 1 บทนำ.....	1
1.1 ความเป็นมาและความสำคัญของปัญหา.....	1
1.2 ความมุ่งหมายและวัตถุประสงค์ของการศึกษา.....	2
1.3 ทฤษฎีหรือแนวความคิดที่ใช้ในการวิจัย.....	2
1.4 ขอบเขตการวิจัย.....	3
1.5 ขั้นตอนการศึกษา.....	3
1.6 เครื่องมือที่ใช้ในการพัฒนาระบบ.....	3
1.7 ประโยชน์และสิ่งที่คาดว่าจะได้รับ.....	3
1.8 นิยามคำศัพท์.....	3
บทที่ 2 ทฤษฎีพื้นฐานและงานวิจัยที่เกี่ยวข้อง.....	4
2.1 โครงสร้างสถาปัตยกรรมระบบจัดกลุ่มผลการสืบค้น.....	4
2.2 การจัดกลุ่มผลการสืบค้นโดยใช้อัลกอริทึมซัพฟิกทรี.....	5
2.2.1 การทำงานของอัลกอริทึมซัพฟิกทรี.....	6
2.3 การประเมินผลการจัดกลุ่ม.....	14
บทที่ 3 การออกแบบและพัฒนาระบบ.....	16
3.1 แนวคิดและวิธีการแก้ปัญหา.....	16
3.2 โครงสร้างการจัดกลุ่มผลการสืบค้น.....	16
3.3 การพัฒนาระบบ.....	19

สารบัญ (ต่อ)

	หน้า
บทที่ 4 ผลการวิจัย	24
บทที่ 5 สรุปผลการวิจัยและข้อเสนอแนะ	35
บรรณานุกรม	36
ภาคผนวก	37
ประวัติผู้เขียน	44

สารบัญตาราง

ตารางที่	หน้า
2.1 แสดงผลการกำหนดกลุ่มพื้นฐานและการคำนวณหาค่า Maximum phrase	10
2.2 แสดงการรวมกลุ่มพื้นฐานด้วยค่า α ที่แตกต่างกัน	12
2.3 แสดงผลการรวมกลุ่มพื้นฐานและคะแนน Final Score	13
4.1 แสดงข้อความหลังผ่านกระบวนการ Stemming.....	27
4.2 แสดงตัวอย่างการคำนวณอัตราการทำซ้ำซ้อนของเอกสาร (Overlap) และความครอบคลุมการจัดกลุ่ม (Coverage) คำว่า “Thailand”	33
4.3 แสดงตัวอย่างการคำนวณอัตราการทำซ้ำซ้อนของเอกสาร (Overlap) และความครอบคลุมการจัดกลุ่ม (Coverage) ของคำสืบค้นต่าง ๆ.....	34

สารบัญรูป

รูปที่	หน้า
2.1 โครงสร้างสถาปัตยกรรมระบบการจัดกลุ่มผลการสืบค้น	4
2.2 ตัวอย่างการสร้าง Suffix Tree ของเอกสารที่ 1	7
2.3 ตัวอย่างการสร้าง Suffix Tree ของเอกสาร D1 และ D2	8
2.4 ตัวอย่างการสร้าง Suffix Tree ของเอกสาร D1, D2 และ D3	8
2.5 ขั้นตอนการยุบรวม Internal node.....	9
2.6 กลุ่มพื้นฐาน	9
2.7 ตัวอย่างการรวมกลุ่มพื้นฐานด้วยค่า α ที่แตกต่างกัน	12
3.1 โครงสร้างระบบจัดกลุ่มผลการสืบค้น.....	18
3.2 อัลกอริทึมสำหรับการจัดกลุ่ม Suffix Tree Clustering.....	19
3.3 ขั้นตอนการนำเข้าข้อมูลจากการ Google Search Engine.....	20
3.4 ขั้นตอนกระบวนการทำ Pre-Processing.....	21
3.5 ขั้นตอนกระบวนการจัดกลุ่มด้วยอัลกอริทึมชฟฟิกรี	22
4.1 หน้าจอเริ่มต้นของระบบ	24
4.2 การกรอกคำค้น (keyword) ในช่อง textfield ที่เตรียมไว้	24
4.3 การเลือกจำนวนกลุ่มที่ต้องการ.....	25
4.4 การเลือกจำนวนผลลัพธ์ที่ต้องการจาก Google.....	25
4.5 การทำงานหลังการกดปุ่ม Search	25
4.6 ตัวอย่างการสร้าง Suffix Tree ของเอกสารจำนวน 11 เอกสาร	28
4.7 ผลลัพธ์ที่ได้จากการสืบค้นและทำการจัดกลุ่ม.....	30
4.8 ผลลัพธ์ที่ได้จากการสืบค้นด้วยคำที่มีลักษณะเป็นชื่อ (Entity Name).....	32

บทที่ 1

บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ด้วยพัฒนาการของเทคโนโลยีอินเทอร์เน็ตก้าวหน้าอย่างรวดเร็ว ทำให้เกิดการจัดเก็บและนำเสนอข้อมูลข่าวสารบนเว็บเป็นจำนวนมากมหาศาล กระจายอยู่ในระดับหลายพันล้านเว็บเพจและคาดว่าอัตราการเพิ่มขึ้นของข้อมูลในเครือข่ายอินเทอร์เน็ตมีมากกว่า 1.5 ล้านหน้าต่อวัน ซึ่งข้อมูลเหล่านั้นมีความสัมพันธ์กันและมีอิทธิพลต่อการดำเนินชีวิตของเรา แม้ว่าข้อมูลจำนวนมากมหาศาลจะเป็นสิ่งที่ดี ขณะเดียวกันก็ทำให้เกิดปัญหาสำหรับผู้ที่ต้องการสืบค้นข้อมูล ถึงแม้จะมีเครื่องมือที่ช่วยในการสืบค้น (Search Engine) ปัญหาของการสืบค้นข้อมูลจำนวนมากไม่ได้อยู่ที่สืบค้นข้อมูลไม่พบ แต่อยู่ที่ผลลัพธ์ที่ได้จากการสืบค้นมีมากเกินไป และข้อมูลจำนวนมากก็ไม่ใช่ว่าข้อมูลที่ผู้ใช้ต้องการทั้งหมด ผู้ใช้ต้องค้นหาเอกสารในลำดับรายการผลการสืบค้นเหล่านั้น ทำให้ผู้ใช้เสียเวลาในการสืบค้นข้อมูลที่ต้องการ โดยปกติผู้ใช้นักจะสืบค้นข้อมูลในลำดับต้น ๆ เท่านั้น ทำให้ผู้สืบค้นอาจไม่พบข้อมูลที่ต้องการ

สาเหตุของปัญหาที่เกิดขึ้นมีสองประการหลัก ๆ คือ ประการแรก เครื่องมือที่ใช้ในการสืบค้น (Search Engine) ในปัจจุบันส่วนใหญ่มีการพัฒนาขึ้นมาใช้งานเป็นเวลานานพอสมควร ในช่วงที่พัฒนาขึ้น เป็นการพัฒนาเพื่อใช้สืบค้นข้อมูลในระดับล้านเว็บเพจ แต่เมื่อใช้งานมาจนถึงปัจจุบันจำนวนข้อมูลได้เพิ่มขึ้นอย่างมหาศาล ทำให้เครื่องมือที่ใช้ในการสืบค้นในรูปแบบลำดับโดยเรียงลำดับจากผลลัพธ์ที่มีความสัมพันธ์กับคำที่ใช้ในการสืบค้นจากมากไปหาน้อย เมื่อมีข้อมูลเป็นจำนวนมาก ทำให้ผลลัพธ์ที่ได้จากสืบค้นมีจำนวนมากตามไปด้วย ซึ่งผลลัพธ์ของการสืบค้นข้อมูลนี้จะมีความสัมพันธ์กับคำที่ใช้ในการสืบค้นในหลากหลายเนื้อหาแตกต่างกันไป เมื่อผลลัพธ์ของการสืบค้นที่มีเนื้อหาแตกต่างกันถูกนำมาแสดงรวมกัน ในรูปแบบลำดับจึงทำให้เกิดความยุ่งยากแก่ผู้ใช้เพราะผู้ใช้งานจะต้องมองหาเฉพาะข้อมูลที่มีเนื้อหาที่ต้องการจากลำดับของผลลัพธ์ที่แสดงโดยเครื่องมือที่ใช้ในการสืบค้น ซึ่งเห็นได้ว่าเป็นการสืบค้นที่ซ้ำซ้อนและเป็นการเสียเวลา ประการที่สอง ผู้ใช้นักจะไม่ได้ดูไปถึงผลลัพธ์ที่อยู่ในลำดับหลัง ๆ จึงทำให้ผลลัพธ์ในลำดับหลัง ๆ ไม่ถูกนำไปใช้แม้จะเกี่ยวข้องกับคำที่ใช้สืบค้นและมีเนื้อหาที่ผู้ใช้ต้องการก็ตาม เหตุผลหนึ่งที่พบคือ ผู้สืบค้นมักใช้จำนวนคำน้อยอยู่ในช่วง 1 – 3 คำ ในการสืบค้นข้อมูลเพราะเนื่องจากผู้ใช้อาจยังไม่แน่ใจกับข้อมูลที่ต้องการจะสืบค้น หรือไม่สามารถกำหนดคำที่ใช้สืบค้นได้อย่างมีประสิทธิภาพ เช่น คำที่ใช้นักมีความหมายกว้าง หรือมีความหมายกำกวม

ถึงแม้ว่าเครื่องมือที่ใช้ในการสืบค้น (Search Engine) ต่าง ๆ จะพยายามสร้างฟังก์ชันช่วยเหลือเพิ่มเติมเพื่อให้ผู้ที่ต้องการสืบค้นข้อมูลได้ใช้สืบค้นข้อมูลให้ตรงตามความต้องการมากที่สุด โดยแสดงเฉพาะข้อมูลที่ต้องการจากการสืบค้นจริง ๆ เท่านั้น แต่ในความเป็นจริงมีคนจำนวนน้อยเท่านั้นที่รู้ถึงวิธีการใช้ฟังก์ชันเหล่านั้นอย่างถูกต้อง โดยคนส่วนใหญ่ใช้ไม่เป็นหรือบางคนอาจไม่รู้ด้วยซ้ำว่าเครื่องมือที่ใช้ในการสืบค้นต่าง ๆ มีฟังก์ชันที่มีประโยชน์เหล่านี้ให้ จึงมีความจำเป็นอย่างยิ่งที่จะต้องมีการพัฒนาวิธีการใหม่ ๆ เพื่อช่วยปรับปรุงการสืบค้นข้อมูล หรือการแสดงผลการสืบค้นข้อมูลให้มีประสิทธิภาพมากขึ้น

แนวทางหนึ่งที่สามารถนำมาแก้ปัญหา คือ การจัดผลลัพธ์ของการสืบค้นให้อยู่ในรูปของกลุ่มตามเนื้อหาที่เกี่ยวข้องกัน แล้วจึงแสดงลำดับของกลุ่มของผลลัพธ์ตามความสัมพันธ์กับคำที่ใช้ในการสืบค้น ซึ่งจะเป็นการช่วยให้การแสดงผลลัพธ์มีประสิทธิภาพมากขึ้น อันจะช่วยประหยัดเวลาของผู้ใช้ โดยผู้ใช้เลือกกลุ่มของเนื้อหาที่สนใจแทนที่จะเลือกดูผลการสืบค้นข้อมูลที่ละรายการ นอกจากนั้นยังเพิ่มโอกาสที่ผลการสืบค้นในลำดับหลัง ๆ จะถูกนำมาใช้ เพราะถูกนำมาแสดงในกลุ่มที่มีเนื้อหาเกี่ยวข้องกัน

จากปัญหาและแนวทางในการแก้ปัญหาที่กล่าวมาแล้วข้างต้น จึงทำให้เกิดความสนใจในการพัฒนาเครื่องมือสำหรับจัดกลุ่มผลลัพธ์จากการสืบค้นโดยนำอัลกอริทึมการจัดกลุ่ม ชื่อ Suffix Tree Clustering (STC) ซึ่งอัลกอริทึมดังกล่าวมีคุณสมบัติในด้านความเร็ว คือ เป็นเส้นตรง (linear time) ตามขนาดของข้อมูล และคำนึงถึงส่วนของการทับซ้อนกัน คือ เอกสารหนึ่งสามารถปรากฏได้มากกว่าหนึ่งกลุ่มเอกสาร

1.2 ความมุ่งหมายและวัตถุประสงค์ของการศึกษา

1. เพื่อศึกษาลักษณะ รูปแบบการทำงานของการทำงานของการจัดกลุ่ม (Clustering) ด้วยอัลกอริทึม Suffix Tree Clustering (STC) เนื่องจากได้วิเคราะห์แล้วว่า การจัดกลุ่มสามารถนำมาใช้แก้ปัญหาข้างต้นได้
2. พัฒนาโปรแกรมการจัดกลุ่มเอกสารให้มีประสิทธิภาพ โดยนำแนวทางที่ได้ศึกษามาพัฒนาเพื่อใช้ทดลองการจัดกลุ่มให้เห็นว่าผลการจัดกลุ่มที่ได้มีประสิทธิภาพ

1.3 ทฤษฎีและแนวคิดที่ใช้ในการวิจัย

งานวิจัยนี้ได้นำทฤษฎีและเทคนิคต่าง ๆ มาประยุกต์ใช้ ประกอบด้วย

1. เทคนิคการจัดกลุ่มผลการสืบค้น ชื่อ Suffix Tree Clustering (STC) ซึ่งมีคุณสมบัติที่ดีในด้านความเร็วและคำนึงถึงส่วนของการทับซ้อนกันของสมาชิก
2. เทคนิคการแตกประโยคเพื่อสร้างรายการลำดับของคำ n คำ (n -gram) ทำให้การสร้างโครงสร้างข้อมูลชื่อทำได้ง่ายขึ้น

1.4 ขอบเขตของการวิจัย

งานวิจัยนี้ นำเสนอเครื่องมือสำหรับการจัดกลุ่มผลการสืบค้นด้วยอัลกอริทึม Suffix Tree Clustering (STC) โดยที่อัลกอริทึมนี้จะรับข้อมูลอินพุตจากผลการสืบค้นของเครื่องมือที่ช่วยในการสืบค้นไปผ่านกระบวนการประมวลผลของ Suffix Tree Clustering (STC) และทำการรวมกลุ่มตามเงื่อนไขที่กำหนดของอัลกอริทึม โดยสามารถจัดกลุ่มหน้าเว็บภาษาไทยได้ด้วย

1.5 ขั้นตอนของการศึกษา

- 1. ศึกษาลักษณะการทำงานและการแสดงผลของเครื่องมือที่ช่วยในการสืบค้น (Search Engine)
- 2. ศึกษาทฤษฎีและแนวคิดจากหนังสือและบทความต่าง ๆ ที่เกี่ยวข้องกับการจัดกลุ่มผลการสืบค้นด้วยอัลกอริทึม Suffix Tree Clustering (STC)
- 3. วิเคราะห์และออกแบบระบบเครื่องมือสำหรับการจัดกลุ่มผลการสืบค้น
- 4. เขียนโปรแกรมตามทฤษฎีที่ได้ศึกษาโดยใช้ซอฟต์แวร์ JAVA และ JSP
- 5. ทดสอบโปรแกรมกับข้อมูลตัวอย่าง พร้อมทั้งแก้ไขข้อผิดพลาดของโปรแกรม
- 6. รวบรวมผลการทดสอบที่ได้จากการทำงานของโปรแกรม
- 7. วิเคราะห์ผลการทดสอบและสรุปผล
- 8. จัดทำเอกสารประกอบโครงการพัฒนาระบบ

1.6 เครื่องมือที่ใช้ในการพัฒนาระบบ

- | | |
|-------------------------------------|---|
| 1. Microsoft Windows XP | เป็นระบบปฏิบัติ |
| 2. Microsoft Office 2003 | ช่วยในการจัดทำเอกสาร |
| 3. Apache Tomcat 5.0 | เป็นเว็บเซิร์ฟเวอร์จำลองในการทดสอบโปรแกรม |
| 4. Macromedia Dreamweaver 8 | ใช้ในการออกแบบหน้าจอติดต่อกับผู้ใช้งาน |
| 5. Java SE Development Kit 1.5.0_16 | ใช้สำหรับแปล Source Code ภาษาจาวา |

1.7 ประโยชน์และสิ่งที่คาดว่าจะได้รับ

ผู้สืบค้นสามารถเข้าถึงผลลัพธ์ที่ต้องการ ได้อย่างรวดเร็ว อันจะช่วยประหยัดเวลาของผู้สืบค้นโดย ผู้ใช้เลือกกลุ่มของเนื้อหาที่สนใจแทนการค้นหาผลการสืบค้นข้อมูลที่ละรายการ

1.8 นิยามคำศัพท์

Common phrase	หมายถึง	วลีที่กลุ่มเอกสารใช้ร่วมกัน
Relevance	หมายถึง	ความเกี่ยวข้อง
Similarity	หมายถึง	ความคล้ายคลึงกัน
Snippets	หมายถึง	คำอธิบายสั้น ๆ ที่ Search Engine ให้กับแต่ละหน้าเว็บผลลัพธ์

บทที่ 2

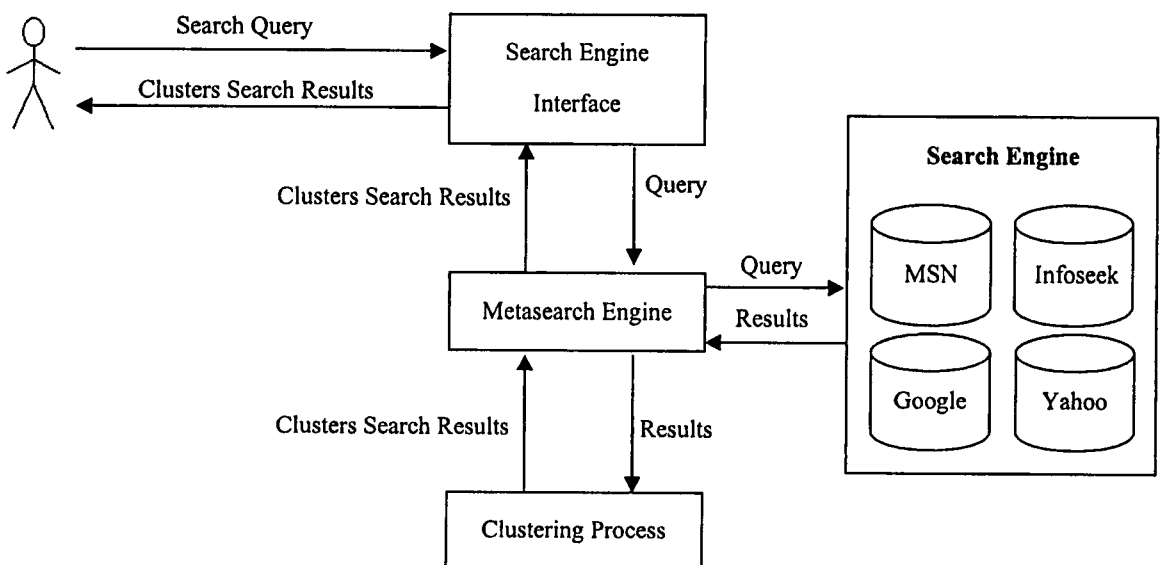
ทฤษฎีพื้นฐานและงานวิจัยที่เกี่ยวข้อง

การจัดกลุ่มผลการสืบค้นบนเว็บไซต์ (Web Search Results Clustering) คือ การนำผลการสืบค้นที่มีความเกี่ยวข้องกัน (Relevance) หรือมีความคล้ายคลึงกัน (Similarity) มารวมในกลุ่มเดียวกัน เพื่ออำนวยความสะดวกให้กับผู้สืบค้น เพื่อให้ผู้สืบค้นสามารถเข้าถึงข้อมูลที่ตรงกับความต้องการ และช่วยให้ผู้สืบค้นได้รับประโยชน์จากผลการสืบค้นมากขึ้น

2.1 โครงสร้างสถาปัตยกรรมระบบจัดกลุ่มผลการสืบค้น

โครงสร้างสถาปัตยกรรมระบบจัดกลุ่มผลการสืบค้น (Borch, 2006) ประกอบด้วย

1. Search Engine Interface ทำหน้าที่ในการรับคำสืบค้น (Search Query) จากผู้สืบค้น และส่งคืนผลการสืบค้นที่ถูกจัดกลุ่มให้กับผู้สืบค้น
2. Metasearch Engine เป็นตัวกลางที่ทำหน้าที่ควบคุมการเข้าถึงข้อมูลบนระบบเครือข่ายอินเทอร์เน็ต
3. Search Engine เป็นระบบสืบค้นที่ระบบจัดกลุ่มผลการสืบค้นต้องการนำผลการสืบค้นมาทำการจัดกลุ่ม เช่น Google, Yahoo, MSN, Infoseek เป็นต้น
4. Clustering Process เป็นกระบวนการทำงานของการจัดกลุ่มผลการสืบค้นด้วยเทคนิคต่าง ๆ เช่น Suffix Tree Clustering (STC), K-Mean เป็นต้น สำหรับโครงการพัฒนาระบบนี้ใช้อัลกอริทึม Suffix Tree Clustering (STC)



รูปที่ 2.1 โครงสร้างสถาปัตยกรรมระบบการจัดกลุ่มผลการสืบค้น

2.2 การจัดกลุ่มผลการสืบค้นโดยใช้อัลกอริทึมซัพฟิกรี (Web Search Results Clustering Using Suffix Tree Algorithm)

Oren Zamir and Oren Etzioni (1998) ได้ระบุข้อกำหนดที่จำเป็นในการจัดกลุ่มผลการสืบค้นก่อนการแสดงผลให้กับผู้สืบค้น ดังนี้

1. **ความเกี่ยวข้อง (Relevance)** วิธีการจัดกลุ่มเอกสารจะต้องสามารถจัดกลุ่มเอกสารที่มีความเกี่ยวข้องกันกับคำถามที่ผู้สืบค้นต้องการ
2. **Browsable Summaries** ผู้ใช้สามารถตัดสินใจได้ในทันทีว่าเอกสารนั้นอยู่ในความสนใจของผู้ใช้หรือไม่ ดังนั้นวิธีการนี้จำเป็นต้องให้คำจำกัดความของกลุ่มที่กะทัดรัดและแม่นยำ
3. **Overlap** เอกสารหนึ่งเอกสารอาจมีได้หลายหัวข้อ ดังนั้น เอกสารหนึ่งเอกสารจึงสามารถปรากฏได้มากกว่าหนึ่งกลุ่ม
4. **Snippet-tolerance** ต้องยอมให้ใช้ snippets แทนการใช้เอกสารเต็มรูปแบบในการจัดกลุ่มทำให้กระบวนการทำงานเร็วขึ้น ไม่ต้องเสียเวลาไปดึงข้อมูลที่เป็นเอกสารเต็มรูปแบบมาใช้งาน
5. **ความเร็ว (Speed)** คือ ระบบการจัดกลุ่มผลการสืบค้นจะต้องสามารถทำงานได้อย่างรวดเร็ว
6. **Incrementality** คือ เมื่อมีเอกสารเข้ามาใหม่ ระบบสามารถจัดกลุ่มได้เลยโดยไม่ต้องเริ่มตั้งแต่ต้นกระบวนการทำงาน จึงทำให้ประหยัดเวลา

จากข้อกำหนดที่จำเป็นในการจัดกลุ่มผลการสืบค้นข้างต้น ทำให้มีการพัฒนาอัลกอริทึมในการจัดกลุ่มผลการสืบค้นบนเครือข่ายอินเทอร์เน็ตหลายอัลกอริทึม และอัลกอริทึมหนึ่งที่ได้รับนิยมนำมาจัดกลุ่มผลการสืบค้น คือ อัลกอริทึม Suffix Tree Clustering (STC) อัลกอริทึมนี้ทำให้การจัดกลุ่มผลการสืบค้นมีความก้าวหน้า เพราะมีจุดเด่น ดังต่อไปนี้

1. มีลักษณะการทำงานแบบ Automatic Clustering and Labeling
2. ให้ความสำคัญกับวลีมากกว่าคำ
3. ลักษณะการจัดกลุ่มเป็นแบบ Incremental Clustering คือ สามารถจัดกลุ่มเอกสารที่เพิ่มเข้ามาใหม่เรื่อย ๆ โดยไม่ต้องเริ่มต้นกระบวนการจัดกลุ่มใหม่เมื่อมีเอกสารใหม่เข้ามา
4. เวลาในการทำงานเป็นเส้นตรงขึ้นอยู่กับจำนวนเอกสาร $O(n)$
5. สามารถทำงานกับคำอธิบายสั้น ๆ (Snippets) แทนเอกสารแบบเต็มรูปแบบ
6. มีลักษณะการทำงานที่ให้ผลการจัดกลุ่มแบบ Overlap Cluster คือ เอกสารหนึ่งเอกสารสามารถปรากฏได้หลาย ๆ กลุ่ม เพื่อเพิ่มพื้นที่ในการเข้าถึงเอกสารให้กับผู้สืบค้น

2.2.1 การทำงานของอัลกอริทึมชฟฟิทธิ ประกอบด้วยขั้นตอน ดังนี้

ขั้นตอนที่ 1 Document “Cleaning”

เป็นการเปลี่ยนแปลงตัวอักษรในแต่ละเอกสาร โดยการใช้กระบวนการทำงานของ Stemming Algorithm คือ ลบตัวหน้า (prefixes) และตัวหลัง (suffixes) ของคำ และลดรูปของคำให้อยู่ในรูปแบบของคำเอกพจน์ (singular) ซึ่งประกอบด้วยขั้นตอน ดังนี้

1.1 กำจัดสิ่งที่ไม่ใช่คำ (Tokenization) พิจารณาเป็นสายอักขระ (string) อักขระที่ไม่ใช่อักขระ เช่น ตัวเลข สัญลักษณ์ และเครื่องหมายวรรคตอนต่าง ๆ และส่วนที่เหลือจะเปลี่ยนให้เป็นตัวอักษรพิมพ์เล็ก (Lower Case) ทั้งหมด

1.2 กำจัดคำหยุด (Stop word Eliminating) เป็นการกรองเอาคำที่ปรากฏเป็นจำนวนมากในเอกสาร ตัวอย่างคำหยุด เช่น คำนำหน้า (Articles) เช่น a, an, the คำบุพบท (Prepositions) เช่น in, on, of คำสันธาน (Conjunctions) เช่น and, or, but, if คำสรรพนาม (Pronouns) เช่น I, you, we, they, he, she, it และคำต่าง ๆ ที่เป็นไปได้ออกจากเอกสาร

1.3 การเปลี่ยนรูปของคำให้อยู่ในรูปของรากศัพท์ของคำ (Stemming word) แทนคำที่เกิดจากรากศัพท์ (variant) ต่าง ๆ เช่น คำกริยาที่เติมท้ายด้วย “ing” คำกริยาที่เติมท้ายด้วย “ed” เป็นต้น ตัวอย่างเช่น คำว่า connect เป็นรากศัพท์ของคำต่อไปนี้ connects, connected, connecting, connection, connections เป็นต้น ซึ่งคำต่าง ๆ เหล่านี้มีความหมายที่คล้ายคลึงกัน ฟังก์ชันที่ใช้สำหรับเปลี่ยนรูปของคำให้กลับมาอยู่ในรูปของรากศัพท์ คือ อัลกอริทึมสำหรับการทำสเต็มมิงของพอร์เตอร์ (Porter’s Stemming Algorithm) (สามารถดูรายละเอียดเพิ่มเติมเกี่ยวกับการทำสเต็มมิงของพอร์เตอร์ ได้ที่ <http://tartarus.org/~martin/PorterStemmer/index.html>)

ขั้นตอนที่ 2 Identifying Base Clusters

เป็นการกำหนดกลุ่มพื้นฐาน (base cluster) โดยการใช้โครงสร้างข้อมูลที่เรียกว่า Suffix Tree มาช่วยในการสร้างและค้นหาวลีที่กลุ่มเอกสารใช้ร่วมกัน (common phrase) ประกอบด้วยขั้นตอนการทำงานดังนี้

2.1 การสร้าง suffix tree เริ่มจากการนำคำอธิบายสั้น ๆ (snippets) มาแตกประโยคออกเป็นคำเดี่ยวตามลำดับของคำ เช่น Cat ate cheeses แบ่งเป็น “cat” “ate” และ “cheeses” ตามลำดับ แล้วนำคำพร้อมลำดับและหมายเลขเอกสารมาสร้างเป็น tree

2.2 การสร้าง suffix tree ซึ่งมี root node เป็น directed tree โดยพิจารณาจาก 2 เงื่อนไข คือ ถ้าคำแรกของคำซ้ำกับคำใน node แรกของ sub tree ใดให้นำคำนั้นเข้าสร้างใน sub tree เดียวกันนั้น และตรวจสอบความซ้ำของคำในทุก node ที่ผ่านถ้าซ้ำให้ใช้ node เดิม ถ้าไม่ซ้ำให้สร้าง node ใหม่จนครบตามคำที่ถูกแบ่ง และถ้าคำแรกของคำที่แบ่งไม่ซ้ำกับคำใน node แรกของ sub tree ใด ให้เริ่มสร้าง suffix tree ใน sub tree ใหม่ ด้วยคำนั้นจนครบตามคำที่ถูกแบ่ง

2.3 เก็บตำแหน่งคำพร้อมหมายเลขเอกสารใน node สุดท้าย เพื่อให้ทราบจุดสิ้นสุดของแต่ละสายอักขระ

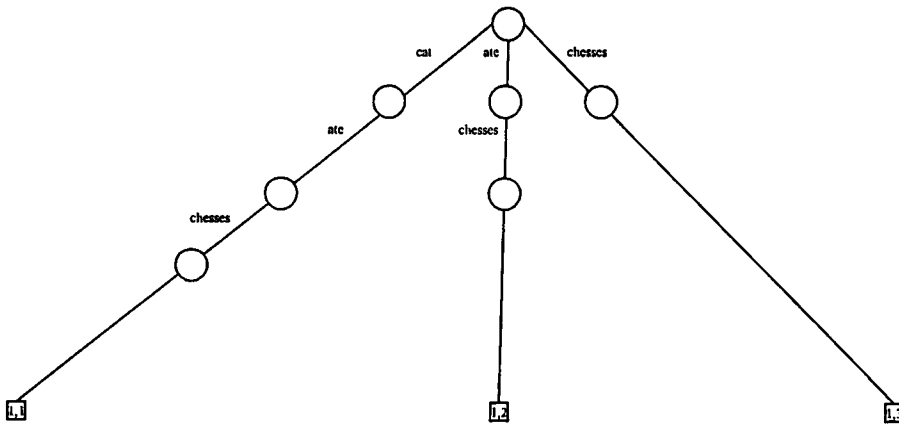
การสร้าง suffix tree จะกำหนดให้ node แทนกลุ่มเอกสาร และลิงค์ (link) แทนป้ายชื่อกลุ่มดังตัวอย่าง จากเอกสาร 3 เอกสาร ต่อไปนี้

เอกสารที่ 1 (D1) : cat ate cheeses

เอกสารที่ 2 (D2) : mouse ate chesses too

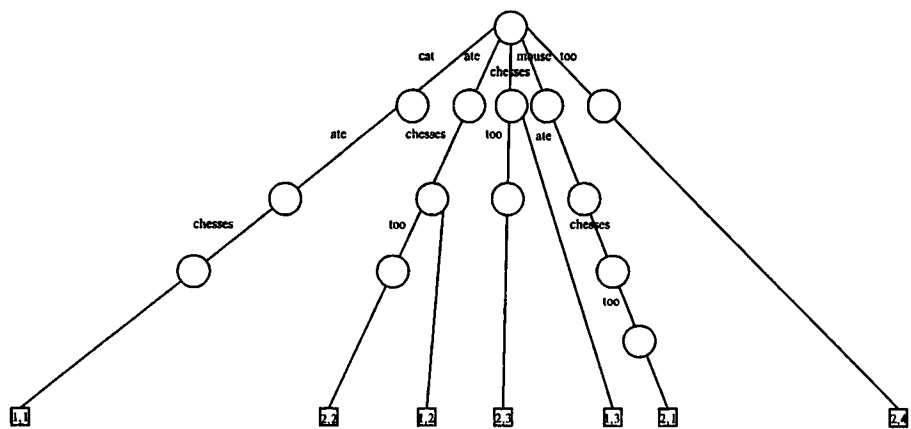
เอกสารที่ 3 (D3) : cat ate mouse too

การสร้าง Suffix Tree เริ่มต้นจากเอกสารที่ 1 ตำแหน่งที่ 1 คือ cat ate cheeses และส่วนของ edge จะเก็บหมายเลขเอกสารและตำแหน่งของเอกสาร นั่นคือ เอกสารที่ 1 ตำแหน่งที่ 1 หลังจากนั้นจะทำการเลื่อนตำแหน่งของคำในเอกสารเดิมครั้งละ 1 ตำแหน่ง ผลคือ ate cheeses แล้วนำเข้าสู่ขั้นตอนการสร้าง suffix tree ส่วนของ edge จะเก็บคำ เอกสารที่ 1 ตำแหน่งที่ 2 ทำซ้ำเหมือนกับขั้นตอนข้างต้น ซึ่งเมื่อเอกสารมีขนาดเท่ากับ 1 คำ ถือเป็นการทำงานจะเป็นรอบสุดท้าย

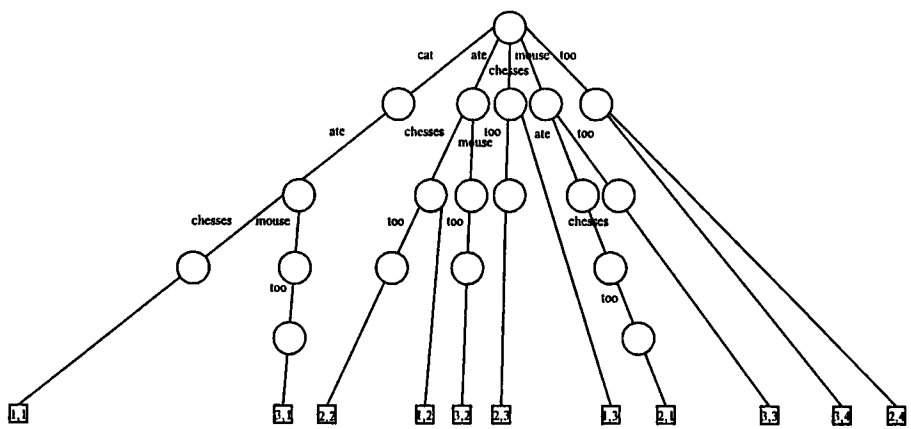


รูปที่ 2.2 ตัวอย่างการสร้าง Suffix Tree ของเอกสารที่ 1

เมื่อเริ่มเอกสารใหม่จะต้องพิจารณาจาก 2 เงื่อนไข คือ ถ้าคำแรกของคำซ้ำกับคำใน node แรกของ sub tree ไคให้นำคำนั้นเข้าสร้างใน sub tree เดียวกันนั้นและตรวจสอบความซ้ำของคำในทุก node ที่ผ่านถ้าซ้ำให้ใช้ node เดิม ถ้าไม่ซ้ำให้สร้าง node ใหม่จนครบตามคำที่ถูกแบ่ง และถ้าคำแรกของคำไม่ซ้ำกับคำใน node แรกของ sub tree ไค ให้สร้าง suffix tree ใน sub tree ใหม่ ด้วยคำนั้นจนครบตามคำที่ถูกแบ่ง ดังรูป 2.3 – 2.4



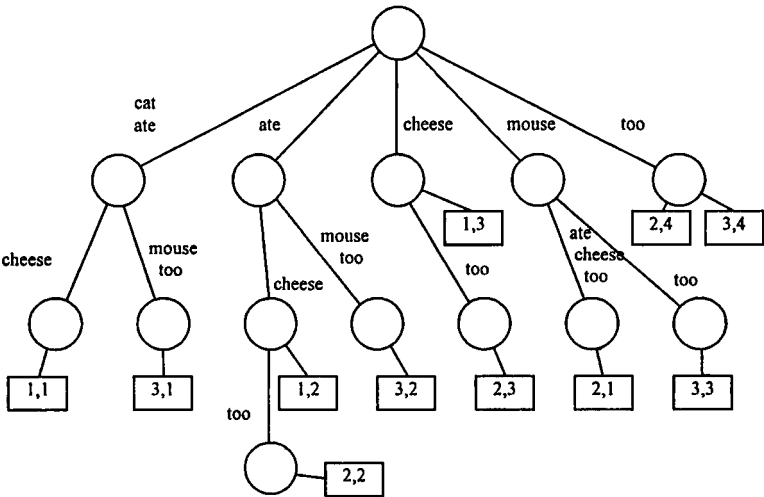
รูปที่ 2.3 ตัวอย่างการสร้าง Suffix Tree ของเอกสาร D1 และ D2



รูปที่ 2.4 ตัวอย่างการสร้าง Suffix Tree ของเอกสาร D1, D2 และ D3

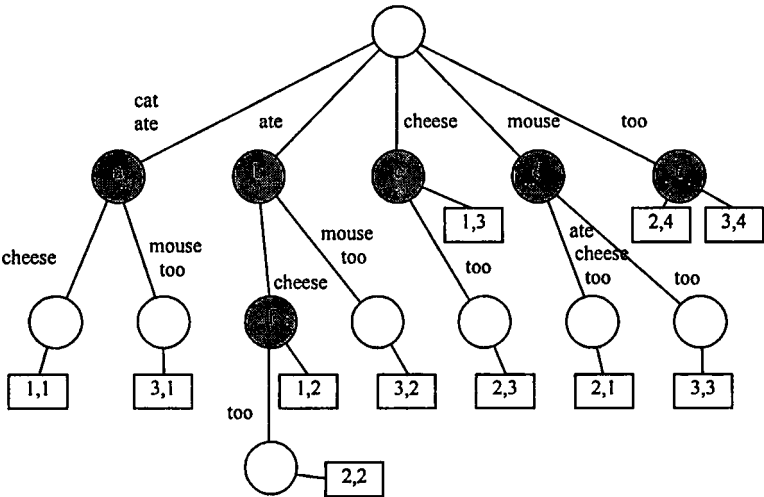
การสร้าง suffix tree จะทำงานวนซ้ำไปเรื่อย ๆ ตั้งแต่คำแรกของเอกสาร จนถึงคำสุดท้ายของเอกสารสุดท้าย และสามารถจัดกลุ่มเอกสารเพิ่มขึ้นได้เรื่อย ๆ โดยไม่ต้องเริ่มต้นการจัดกลุ่มใหม่เมื่อมีเอกสารใหม่เข้ามา เรียกวิธีการนี้ว่า Incremental Clustering ซึ่งการทำงานแต่ละรอบจะพบกลุ่มเอกสารที่ปรากฏในแต่ละ node และป้ายชื่อกลุ่ม (label) ในแต่ละลิงค์

2.4 การยุบรวม Internal node (Compact Internal Node) เพื่อลดจำนวน node และเวลาในการประมวลผลการสร้างและค้นหาวลีที่กลุ่มเอกสารใช้ร่วมกัน (common phrase) โดยจะทำการยุบรวม internal node ที่ไม่มีเอกสารประกอบอยู่ภายใน node และมีลิงค์ (link) หนึ่งลิงค์ออกจาก node หลักการยุบรวม internal node คือ การนำคำของแต่ละ node ที่ถูกยุบรวม เรียงจากบนลงล่างของ sub tree มาต่อกันให้เป็นวลี (phrase) ดังรูปที่ 2.5



รูปที่ 2.5 ขั้นตอนการยุบรวม Internal node

2.5 ค้นหากลุ่มพื้นฐาน (base cluster) โดยพิจารณาจากการใช้คำร่วมกัน (common phrase) ของกลุ่มเอกสาร นั่นคือ แต่ละ node ที่จะได้รับเลือกเป็นกลุ่มพื้นฐานจะต้องมีเอกสารหรือลิงค์ (link) เป็นส่วนประกอบอยู่มากกว่า 1 จึงจะกำหนดให้เป็นกลุ่มพื้นฐาน (base cluster) ดังรูป



รูปที่ 2.6 กลุ่มพื้นฐาน

2.6 คำนวณหาค่า Maximum phrase ของแต่ละ Phrase ตามสูตร

$$S(m) = |d| * f(|m_p|) * \sum (gfidf(w_i)) \tag{2.1}$$

- เมื่อ $S(m)$ คือ คะแนน Maximum phrase ของ cluster m กับ phrase m_p
- $|d|$ คือ จำนวนของเอกสารภายในกลุ่ม
- $|m_p|$ คือ จำนวนของคำที่ปรากฏในป้ายชื่อกลุ่ม
- f คือ ฟังก์ชันของการเกิดเหตุการณ์ phrase มีความยาวเป็นหนึ่ง
- w_i คือ คำที่ปรากฏใน phrase m_p
- $gfidf(w_i)$ คือ คะแนนความถี่ของคำปรากฏใน phrase m_p

สูตร การคำนวณค่า term frequency-inverse document frequency : $tfidf(w_i)$ คือ

$$tfidf(w_i,d) = (1+\log(tf(w_i,d))) * \log(1+N/df(w_i)) \tag{2.2}$$

เมื่อ $tf(w_i,d)$ คือ จำนวนความถี่ของคำที่ปรากฏในเอกสาร d

$df(w_i)$ คือ จำนวนเอกสารที่มีคำ w_i ปรากฏ

N คือ จำนวนเอกสารทั้งหมด

ตัวอย่างการคำนวณค่า $tfidf(w_i,d)$ ของกลุ่ม cat ate ในเอกสารที่ 1 และเอกสารที่ 3

❖ ในเอกสารที่ 1 ปรากฏคำว่า “cat ate” จำนวน 1 ครั้ง ค่า $tf(w_i,d) = 1$ และปรากฏคำว่า “cat ate” อยู่จำนวน 2 เอกสาร คือ เอกสารที่ 1 และเอกสารที่ 3 ค่า $df(w_i) = 2$ จากสมการข้างต้นสามารถแทนค่าได้ดังนี้

$$\begin{aligned} tfidf(w_i,d) &= (1 + \log(1) * \log (1 + (3 / 2))) \\ &= (1 + 0) * 1.322 \\ &= 1.322 \end{aligned}$$

❖ ในเอกสารที่ 3 ปรากฏคำว่า “cat ate” จำนวน 1 ครั้ง ค่า $tf(w_i,d) = 1$ และปรากฏคำว่า “cat ate” อยู่จำนวน 2 เอกสาร คือ เอกสารที่ 1 และเอกสารที่ 3 ค่า $df(w_i) = 2$ จากสมการข้างต้นสามารถแทนค่าได้ดังนี้

$$\begin{aligned} tfidf(w_i,d) &= (1 + \log(1) * \log (1 + (3 / 2))) \\ &= (1 + 0) * 1.322 \\ &= 1.322 \end{aligned}$$

ตารางที่ 2.1 แสดงผลการกำหนดกลุ่มพื้นฐานและการคำนวณหาค่า Maximum phrase

Node	Phrase	Documents	$tfidf(w_i,d)$	S(m)
a	cat ate	1,3	$1.322 + 1.322 = 2.644$	$2 * 2 * 2.644 = 10.576$
b	ate	1,2,3	$1 + 1 + 1 = 3$	$3 * 1 * 3 = 9$
c	cheese	1,2	$1.322 + 1.322 = 2.644$	$2 * 1 * 2.644 = 5.288$
d	mouse	2,3	$1.322 + 1.322 = 2.644$	$2 * 1 * 2.644 = 5.288$
e	too	2,3	$1.322 + 1.322 = 2.644$	$2 * 1 * 2.644 = 5.288$
f	ate cheese	1,2	$1.322 + 1.322 = 2.644$	$2 * 2 * 2.644 = 10.576$

ขั้นตอนที่ 3 Merging Clusters

คือ การนำกลุ่มพื้นฐานที่ได้จากขั้นตอนที่ 2 (node) มารวมกัน เพื่อลดจำนวนของกลุ่มก่อนการแสดงผลให้กับผู้สืบค้น โดยพิจารณาความคล้ายคลึงกัน (similarity) ตามสูตร

$$\text{similarity}(a, b) = 1 \text{ if } \left(\frac{|a \cap b|}{|a|} > \alpha \right) \text{ and } \left(\frac{|a \cap b|}{|b|} > \alpha \right) \quad (2.3)$$

$\text{similarity}(a, b) = 0$ otherwise

α แสดงความคล้ายคลึงกันของเอกสารในกลุ่ม โดยมีค่าระหว่าง 0 – 1 ถ้ากำหนดค่า α สูง แสดงว่าข้อมูลทั้งสองกลุ่มจะต้องมีความคล้ายคลึงกันมากจึงจะสามารถรวมกันได้ ในรายงานของ Oren Zamir and Oren Etzioni (1998) กำหนดให้ค่า $\alpha = 0.6$ ถ้าค่าความคล้ายคลึงกันที่คำนวณจากสูตรมากกว่าหรือเท่ากับ 0.6 ถือว่าค่าความคล้ายคลึงกัน (similarity) เท่ากับ 1 แสดงว่าเราสามารถรวมสองกลุ่มเข้าด้วยกันได้ แต่ถ้าค่าความคล้ายคลึงกันที่คำนวณจากสูตรน้อยกว่า 0.6 ถือว่าค่าความคล้ายคลึงกัน (similarity) เท่ากับ 0 แสดงว่าเราไม่สามารถรวมสองกลุ่มเข้าด้วยกันได้

ถ้ากำหนดค่า $\alpha = 0.6$ หาค่าความคล้ายคลึงกัน (similarity) ตามสูตรข้างต้น ดังนี้

$$\begin{aligned} &= \left(\frac{|\{1,3\} \cap \{1,2,3\}|}{|\{1,3\}|} > 0.6 \right) \text{ and } \left(\frac{|\{1,3\} \cap \{1,2,3\}|}{|\{1,2,3\}|} > 0.6 \right) \\ &= \left(\frac{|\{1,3\}|}{|\{1,3\}|} > 0.6 \right) \text{ and } \left(\frac{|\{1,3\}|}{|\{1,2,3\}|} > 0.6 \right) \\ &= \left(\frac{2}{2} = 1 > 0.6 \right) \text{ and } \left(\frac{2}{3} = 0.67 > 0.6 \right) \end{aligned}$$

similarity = 1

เนื่องจากเงื่อนไขทั้งสองเป็นจริง ทำให้ค่าความคล้ายคลึงกัน (similarity) มีค่าเท่ากับหนึ่ง จึงสามารถรวม node a เข้ากับ node b ได้

ถ้ากำหนดค่า $\alpha = 0.7$ หาค่าความคล้ายคลึงกัน (similarity) ตามสูตรข้างต้น ดังนี้

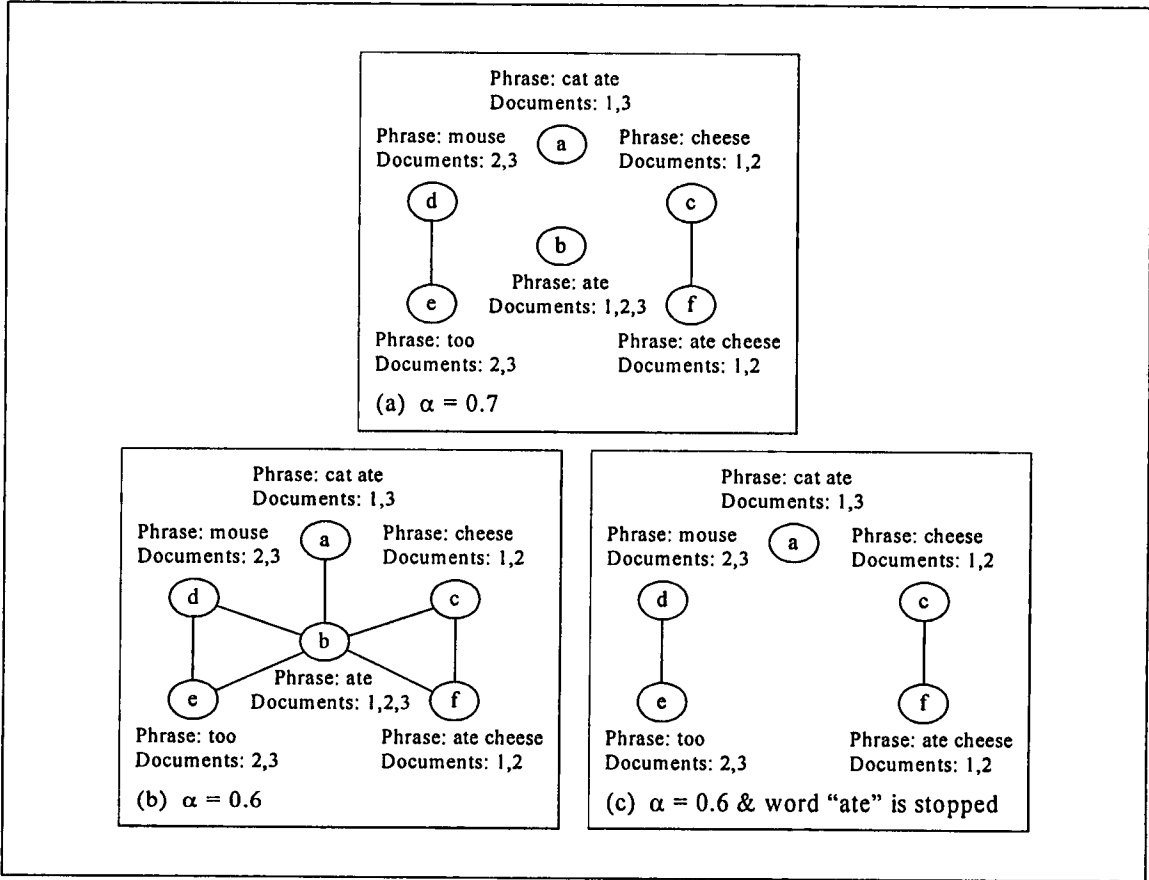
$$\begin{aligned} &= \left(\frac{|\{1,3\} \cap \{1,2,3\}|}{|\{1,3\}|} > 0.7 \right) \text{ and } \left(\frac{|\{1,3\} \cap \{1,2,3\}|}{|\{1,2,3\}|} > 0.7 \right) \\ &= \left(\frac{|\{1,3\}|}{|\{1,3\}|} > 0.7 \right) \text{ and } \left(\frac{|\{1,3\}|}{|\{1,2,3\}|} > 0.7 \right) \\ &= \left(\frac{2}{2} = 1 > 0.7 \right) \text{ and } \left(\frac{2}{3} = 0.67 < 0.7 \right) \end{aligned}$$

similarity = 0

เนื่องจากเงื่อนไขที่สองเป็นเท็จไม่สอดคล้องกับสูตรที่กำหนด ทำให้ค่าความคล้ายคลึงกัน (similarity) มีค่าเท่ากับศูนย์ จึงไม่สามารถรวม node a เข้ากับ node b ได้

ตารางที่ 2.2 แสดงการรวมกลุ่มพื้นฐานด้วยค่า α ที่แตกต่างกัน

Figure	Cluster Number	Phrase clusters	Documents
(a) $\alpha = 0.7$	1	a	1,3
	2	b	1,2,3
	3	d,e	2,3
	4	c,f	1,2
(b) $\alpha = 0.6$	1	a,b,c,d,e,f,g	1,2,3
(c) $\alpha = 0.6$ แต่ตัดกลุ่ม b ออก โดยกำหนดให้มันเป็นคำหยุด (stop word)	1	a	1,3
	2	d,e	2,3
	3	c,f	1,2



รูปที่ 2.7 ตัวอย่างการรวมกลุ่มพื้นฐานด้วยค่า α ที่แตกต่างกัน

จากรูปที่ 2.7 เป็นแผนภูมิการรวมกลุ่มพื้นฐานด้วยค่า α ที่แตกต่างกัน ทำให้ได้รับกลุ่มที่แตกต่างกัน เช่น รูป 2.7(a) คือ ผลการรวมกลุ่มพื้นฐานโดยกำหนดให้ค่า α มีค่าเท่ากับ 0.7 จะได้ผลลัพธ์การจัดกลุ่ม 4 กลุ่ม คือ กลุ่ม {a},{b},{d, e} และ {c, f} รูป 2.7(b) คือ ผลการรวมกลุ่มพื้นฐานโดยกำหนดให้ค่า α มีค่าเท่ากับ 0.6 จะได้ผลลัพธ์การจัดกลุ่ม 1 กลุ่ม คือ กลุ่ม {a, b, c, d, e, f} รูป 2.7(c) คือ ผลการรวมกลุ่มพื้นฐานโดยกำหนดให้ค่า α มีค่าเท่ากับ 0.6 และกำหนดให้ “ate” เป็นคำหยุด (stopped) จะได้ผลลัพธ์การจัดกลุ่ม 3 กลุ่ม คือ กลุ่ม {a}, {d, e} และ {c, f}

ขั้นที่ 4 ขั้นตอนการจัดลำดับกลุ่มเอกสาร (Ranking content based cluster)

หลังจากนำตัวแทนกลุ่มมารวมกัน ก็จะได้รายการของกลุ่ม ซึ่งในแต่ละกลุ่มจะประกอบไปด้วย label อย่างน้อย 1 label และจะมีรายการของเอกสารซึ่งอยู่ใน label นั้น ขั้นตอนต่อไป คือ การจัดลำดับผลการสืบค้นโดยใช้คะแนน final score ในการกำหนดคะแนนจะต้องคำนึงถึงสิ่งต่อไปนี้

- 1. กลุ่มที่มีจำนวนเอกสารมากควรมีคะแนนมากกว่ากลุ่มที่มีจำนวนเอกสารน้อย
- 2. คะแนนที่จะกำหนดให้แต่ละป้ายชื่อจะเป็นสัดส่วนกับค่าที่คาดหวัง เช่น ตัวอย่างป้ายชื่อที่พบบ่อย ๆ เช่น คำว่า “the” จะมีคะแนนน้อยกว่าป้ายชื่อคำว่า “cat”
- 3. ป้ายชื่อที่เป็นสมาชิกของกลุ่มหนึ่ง (sub set) มีโอกาสที่จะเป็นสมาชิกของอีกกลุ่มหนึ่ง ดังนั้น การนับควรระวังการนับซ้ำ เนื่องจากจะมีเอกสารที่เหมือนกัน

การแสดงผลการจัดกลุ่มนั้น จะแสดงผลโดยกลุ่มใดที่มีคะแนน final score สูงจะถูกแสดงอยู่ในลำดับต้น ๆ และถูกนำไปเป็นป้ายชื่อกลุ่ม

สามารถคำนวณคะแนน final score โดยใช้ข้อมูลจาก รูป 2.7(b) ซึ่งกำหนดค่า α เท่ากับ 0.6 เราสามารถคำนวณค่าโดยใช้สูตร ได้ดังนี้

$$S(c) = \sum_{b \in c} S(m)$$
 (2.4)

- เมื่อ $S(c)$ คือ คะแนน final score
- b คือ กลุ่มพื้นฐานที่ถูกรวมไว้ในกลุ่ม
- $S(m)$ คือ คะแนน $S(m)$ ของแต่ละกลุ่มพื้นฐานที่ถูกรวมไว้ในกลุ่ม

เมื่อรวมกลุ่มพื้นฐานตามเงื่อนไขการรวมกลุ่ม จะได้ผลการรวมกลุ่ม ดังตารางที่ 2.3

ตารางที่ 2.3 แสดงผลการรวมกลุ่มพื้นฐานและคะแนน Final Score

Figure	Cluster Number	Phrase clusters	Label	Final Score
(b) $\alpha = 0.6$	1	b, a b, d, e b, c, f	cat ate mouse ate too ate cheeses	$9+10.576 = 19.576$ $9+5.288+5.288 = 19.576$ $9 + 5.288 + 10.576 = 24.864$

จากคะแนน Final Score ที่คำนวณได้ เราสามารถแสดงผลการจัดกลุ่มได้ดังนี้

- ate
 - ate cheeses
 - mouse ate too
 - cat ate

2.3 การประเมินผลการจัดกลุ่ม

หลังจากการจัดกลุ่มแล้ว เราจะต้องทำการประเมินคุณภาพของแต่ละกลุ่ม ซึ่งการประเมินผลจะแตกต่างจากการประเมินผลการแบ่งกลุ่ม (Classification) เนื่องจากสามารถทำได้ง่ายโดยใช้ป้ายชื่อ (label) เพื่อใช้ในการทดสอบข้อมูล แต่สำหรับการจัดกลุ่ม (Clustering) เราจะไม่สามารถทราบได้เลยว่าอะไรถูก อะไรผิด ด้วยเหตุนี้การประเมินผลการจัดกลุ่มจึงเป็นเรื่องยาก ดังนั้น ในที่นี้จะขอนำเสนอเครื่องมือที่ใช้ในการวัดคุณภาพของการจัดกลุ่ม (Liu, 2007) ดังนี้

2.3.1 เอนโทรปี (Entropy) เป็นการวัดการซ้อนทับของกลุ่มเอกสาร ถ้าเอกสารในกลุ่มใดกลุ่มหนึ่งประกอบด้วยเอกสารเพียงประเภทเดียวแสดงว่าค่า Entropy เป็นศูนย์ คือ ไม่มีการซ้อนทับกันของกลุ่มข้อมูล แต่ถ้าแต่ละกลุ่มประกอบด้วยเอกสารหลายประเภท แสดงว่าเกิดการคละกันของเอกสารต่างประเภทเป็นจำนวนมาก ค่าของ Entropy จะมีค่ามาก สูตรในการคำนวณค่า Entropy มีดังนี้

$$\text{entropy}(D_i) = - \sum_{j=1}^k \text{Pr}_i(c_j) \log_2 \text{Pr}_i(c_j) \quad (2.5)$$

โดย $\text{Pr}_i(c_j)$ หาได้จากอัตราส่วนระหว่างความถี่ของคำที่ i เอกสารที่ j และเอนโทรปีเฉลี่ยของกลุ่มที่ถูกจัด คำนวณได้จากผลรวมของเอนโทรปีแต่ละกลุ่มซึ่งถ่วงน้ำหนักโดยขนาดของแต่ละกลุ่ม สามารถใช้สูตร ดังนี้

$$\text{entropy}_{\text{total}}(D) = \sum_{i=1}^k \frac{|D_i|}{|D|} \times \text{entropy}(D_i) \quad (2.6)$$

2.3.2 Purity เป็นการวัดการความบริสุทธิ์ของแต่ละกลุ่ม โดยในแต่ละกลุ่มจะต้องประกอบไปด้วยเอกสารหรือข้อมูลเพียงชนิดเดียวกัน โดยสามารถคำนวณความบริสุทธิ์ของแต่ละกลุ่มได้ตามสูตร

$$\text{purity}(D_i) = \max(\text{Pr}_i(c_j)) \quad (2.7)$$

และการวัดความบริสุทธิ์รวมทั้งหมดของแต่ละกลุ่ม สามารถคำนวณได้ ตามสูตร

$$\text{purity}_{\text{total}}(D) = \sum_{i=1}^k \frac{|D_i|}{|D|} \times \text{purity}(D_i) \quad (2.8)$$

2.3.3 การวัดอัตราการใช้ซ้อนของเอกสาร (Overlap) คือ การวัดจำนวนเอกสารที่ปรากฏมากกว่าหนึ่งกลุ่มเอกสาร ดังแสดงรายละเอียดในสมการที่ 2.9

$$\text{Overlap} = (c/s) - 1 \quad (2.9)$$

โดยที่ c คือ ผลรวมของจำนวนเอกสารในแต่ละกลุ่ม

s คือ จำนวนเอกสารทั้งหมด

2.3.4 การวัดประสิทธิภาพด้านความครอบคลุมการจัดกลุ่ม (Coverage) คือ การคำนวณจำนวนผลการสืบค้นที่สามารถจัดกลุ่มได้จากผลการสืบค้นทั้งหมด ดังแสดงรายละเอียดในสมการที่ 2.10

$$\text{Overage} = a/s \quad (2.10)$$

โดยที่ a คือ จำนวนเอกสารที่ถูกจัดกลุ่ม

s คือ จำนวนเอกสารทั้งหมด

จากการศึกษารายงานผลการวิจัยและทฤษฎีต่าง ๆ ดังที่ได้กล่าวข้างต้น เห็นได้ว่าการจัดกลุ่มผลลัพธ์ของการสืบค้นให้อยู่ในรูปของกลุ่มตามเนื้อหาที่เกี่ยวข้องกันด้วยอัลกอริทึมซัพฟิกทรี (Suffix Tree Clustering) มีลักษณะการจัดกลุ่มและการสร้างป้ายชื่อแบบอัตโนมัติ (Automatic Clustering and Labeling) โดยให้ผลการจัดกลุ่มแบบ Overlap Cluster คือ เอกสารหนึ่งเอกสารสามารถปรากฏได้หลาย ๆ กลุ่ม เพื่อเพิ่มพื้นที่ในการเข้าถึงเอกสารให้กับผู้สืบค้น ช่วยให้การแสดงผลลัพธ์ของการสืบค้นมีประสิทธิภาพมากขึ้น สำหรับรายละเอียดเกี่ยวกับพัฒนาโปรแกรมจะกล่าวในบทที่ 3

บทที่ 3

การออกแบบและพัฒนาระบบ

3.1 แนวคิดและวิธีการแก้ปัญหา

เครื่องมือที่ใช้ในการสืบค้นข้อมูล (Search engine) ในรูปแบบลำดับ เช่น Google Yahoo MSN เป็นต้น โดยมีลักษณะการทำงาน คือ เรียงลำดับผลลัพธ์ที่มีความสัมพันธ์กับคำที่ใช้ในการสืบค้นจากมากไปหาน้อย เมื่อมีข้อมูลเป็นปริมาณเพิ่มมากขึ้น ผลลัพธ์ที่ได้จากการสืบค้นก็จะมีจำนวนมากตามไปด้วย ซึ่งผลลัพธ์ที่ได้จากการสืบค้นจะมีความสัมพันธ์กับคำที่ใช้ในการสืบค้นในหลากหลายเนื้อหาแตกต่างกันไป เมื่อผลลัพธ์จากการสืบค้นที่มีเนื้อหาแตกต่างกันถูกนำมาแสดงรวมกัน ในรูปแบบลำดับจึงทำให้เกิดความยุ่งยากแก่ผู้สืบค้นเพราะผู้สืบค้นจะต้องมองหาเฉพาะข้อมูลที่มีเนื้อหาที่ต้องการจากลำดับของผลลัพธ์ที่แสดง เห็นได้ว่าเป็นการสืบค้นที่ซ้ำซ้อนและเสียเวลา ผู้สืบค้นมักจะไม่ได้ดูไปถึงผลลัพธ์ที่อยู่ในลำดับหลัง ๆ จึงทำให้ผลลัพธ์ในลำดับหลัง ๆ ไม่ถูกนำไปใช้แม้จะเกี่ยวข้องกับคำที่ใช้สืบค้นและมีเนื้อหาที่ผู้สืบค้นต้องการก็ตาม

แนวทางหนึ่งที่สามารถนำมาแก้ปัญหา คือ การจัดกลุ่มผลลัพธ์ของการสืบค้นให้อยู่ในรูปแบบของกลุ่มตามเนื้อหาที่เกี่ยวข้องกัน แล้วจึงแสดงลำดับของกลุ่มของผลลัพธ์ตามความสัมพันธ์กับคำที่ใช้ในการสืบค้น ซึ่งช่วยให้การแสดงผลลัพธ์ของการสืบค้นมีประสิทธิภาพมากขึ้น อันจะช่วยประหยัดเวลาของผู้สืบค้น โดยเลือกกลุ่มของเนื้อหาที่สนใจแทนที่จะเลือกดูผลการสืบค้นข้อมูลทีละรายการ และยังเพิ่มโอกาสที่ผลการสืบค้นในลำดับหลัง ๆ จะถูกนำมาใช้ เพราะถูกนำมาแสดงในกลุ่มที่มีเนื้อหาที่เกี่ยวข้องกัน

3.2 โครงสร้างการจัดกลุ่มผลการสืบค้น

โครงสร้างของการจัดกลุ่มผลการสืบค้น ประกอบด้วย 2 ส่วน คือ Metasearch Engine และ Clustering Engine

1. **Metasearch Engine** เป็นกลไกของกระบวนการสืบค้นที่ผู้สืบค้นต้องการผลลัพธ์จากการสืบค้นจาก Search engine โดยมีขั้นตอน ดังนี้

- 1.1 เริ่มต้นจากสร้าง URL แล้วส่ง URL ดังกล่าว ไปยังเครื่องมือที่ใช้ในการสืบค้นข้อมูล (Search Engine) เช่น Google Yahoo หรือ MSN เป็นต้น
- 1.2 เมื่อได้รับผลลัพธ์คืนจากเครื่องมือที่ใช้ในการสืบค้นข้อมูล (Search Engine) ระบบ จะทำการดึงส่วนประกอบที่สำคัญ 3 ส่วนออกมา คือ Title Snippet และ URL ของแต่ละ Title
- 1.3 ทำการลบ HTML Tags ทั้งหมด หลังจากนั้นสร้างรายการของเอกสาร (Result List) ซึ่งประกอบด้วย Titles Snippets URLs ก่อนนำเข้าสู่กระบวนการการจัดกลุ่ม

2. Clustering Engine เป็นกระบวนการการจัดกลุ่มผลลัพธ์ที่ได้จากกระบวนการ Metasearch Engine มีขั้นตอน ดังนี้

2.1 Pre-processing กระบวนการนี้ เริ่มต้นจากการกำจัดสิ่งที่ไม่ใช่คำ (Tokenization) และ คำหยุด (Stop word) เปลี่ยนคำให้เป็นตัวอักษรพิมพ์เล็ก (Lower Case) ทั้งหมด หลังจากนั้นเปลี่ยนรูปของคำให้กลับมาอยู่ในรูปของรากศัพท์ โดยใช้ Porter's Stemming Algorithm

2.2 การจัดกลุ่มผลการสืบค้นด้วย Suffix Tree Clustering มีขั้นตอนดังนี้

1. นำผลลัพธ์ที่ได้จากข้อ 2.1 (Result List) ซึ่งประกอบด้วย Titles, Snippets และ URLs มาแบ่งออกเป็นคำเดี่ยว ๆ ตามลำดับ แล้วนำคำพร้อมลำดับและหมายเลขเอกสารมาสร้างเป็น tree
2. สร้าง suffix tree ซึ่งมี root node เป็น directed tree โดยพิจารณาจาก 2 เงื่อนไข คือ ถ้าคำแรกของคำซ้ำกับคำใน node แรกของ sub tree ใดให้นำคำนั้นเข้าสร้างใน sub tree เดียวกันและตรวจสอบความซ้ำของคำในทุก node ที่ผ่านถ้าซ้ำให้ใช้ node เดิม ถ้าไม่ซ้ำให้สร้าง node ใหม่จนครบตามคำที่ถูกแบ่ง และถ้าคำแรกของคำที่แบ่งไม่ซ้ำกับคำใน node แรกของ sub tree ใด ให้เริ่มสร้าง suffix tree ใน sub tree ใหม่ ด้วยคำนั้นจนครบตามคำที่ถูกแบ่ง
3. การยุบรวม Internal node (Compact Internal Node) เพื่อลดจำนวน node และ เวลาในการประมวลผลการสร้างและค้นหาวลีที่กลุ่มเอกสารใช้ร่วมกัน (common phrase) จะทำการยุบรวม internal node ที่ไม่มีเอกสารประกอบอยู่ภายใน node และมีลิงค์ (link) หนึ่งลิงค์ออกจาก node หลักการยุบรวม internal node คือ การนำคำของแต่ละ node ที่ถูกยุบรวมเรียงจากบนลงล่างของ sub tree มาต่อกันให้เป็นวลี (phrase)
4. ค้นหากลุ่มพื้นฐาน (base cluster) โดยพิจารณาจากการใช้วลีร่วมกัน (common phrase) ของกลุ่มเอกสาร นั่นคือ แต่ละ node ที่จะได้รับเลือกเป็นกลุ่มพื้นฐานจะต้องมีเอกสารหรือลิงค์ (link) เป็นส่วนประกอบอยู่มากกว่าหนึ่งลิงค์จึงจะกำหนดให้เป็นกลุ่มพื้นฐาน (base cluster)
5. นำกลุ่มพื้นฐานที่ได้ (base cluster) มารวมกัน (Merging Clusters) เพื่อลดจำนวนของกลุ่มก่อนการแสดงผลให้กับผู้สืบค้น พร้อมกำหนดป้ายชื่อของกลุ่มที่นำมารวมกัน การรวมกลุ่มพิจารณาความคล้ายคลึงกัน (similarity) ของเอกสาร ดังสมการที่ 2.1 คือ

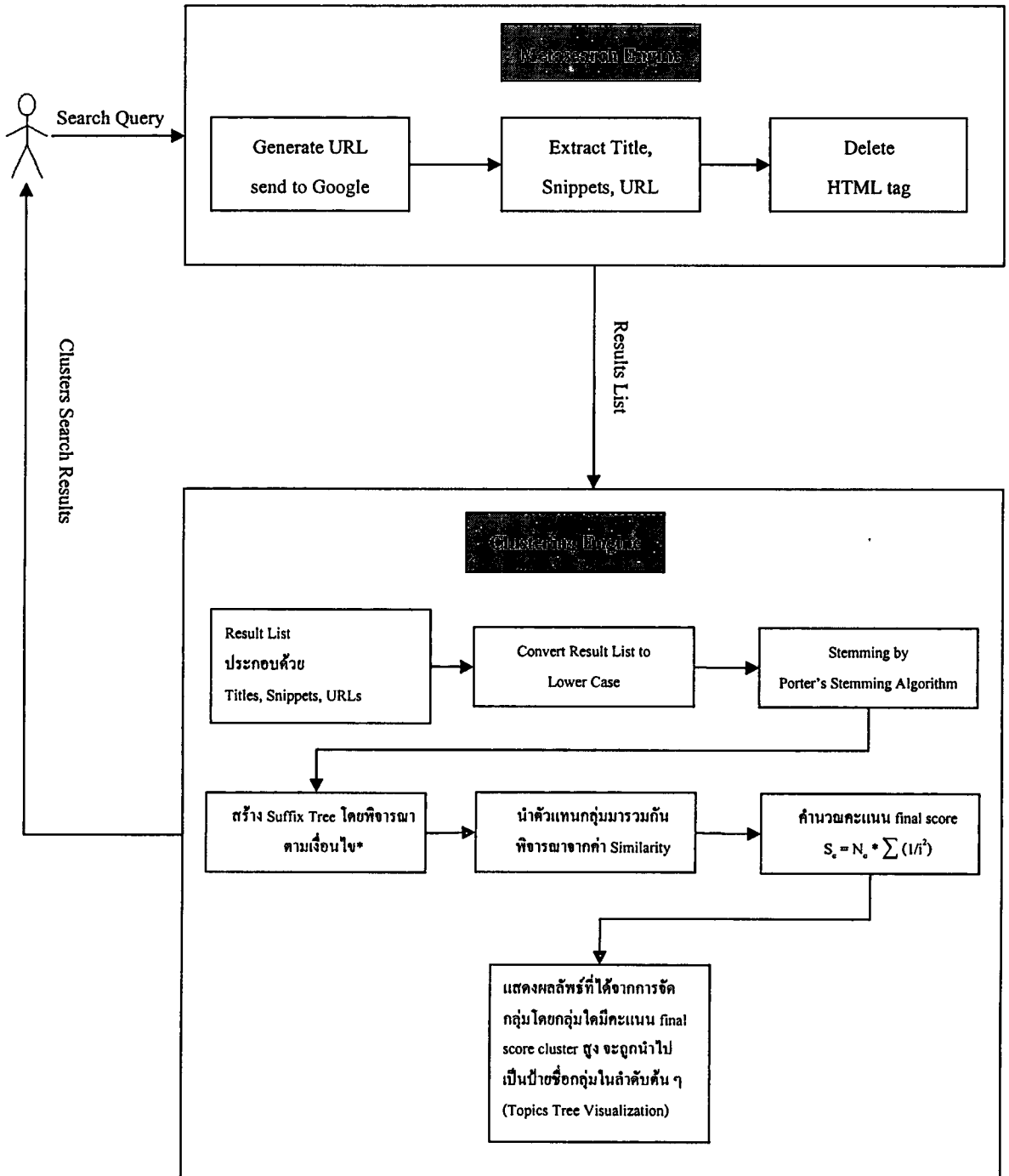
$$\text{similarity}(a, b) = 1 \text{ if } \left(\frac{|a \cap b|}{|a|} > \alpha \right) \text{ and } \left(\frac{|a \cap b|}{|b|} > \alpha \right)$$

$$\text{similarity}(a, b) = 0 \text{ otherwise}$$

6. จัดลำดับกลุ่มเอกสาร (Ranking content based cluster) โดยใช้สูตร ดังนี้

$$S_c = N_c * \sum (1/i^2) \quad (3.1)$$

โดยที่ N_c คือ จำนวนเอกสารในแต่ละกลุ่ม i ลำดับของเอกสารที่ได้จากการสืบค้นของ Google โดยเรียงจากกลุ่มเอกสารที่มีค่า S_c สูงไปหากลุ่มเอกสารที่มีค่า S_c น้อย



รูปที่ 3.1 โครงสร้างระบบจัดกลุ่มผลการสืบค้น (Borch, 2006)

3.3 การพัฒนาระบบ

```

/**
 *Suffix Tree Clustering Algorithm
 */
split text into sentences consisting of words;

/* Phase 1a. Creation of a Generalized Suffix Tree of all sentences */
for each document
{
    for each sentence
    {
        if (sentence length > 0)
        {
            insert sentence and all its substrings into generalized
            suffix tree and update internal nodes with the index to current
            document while rearranging the tree;
        }
    }
}

/* Phase 1b. Build a list of base clusters */
for each node in the tree
{
    if (number of documents in node's subtree > 2)
    {
        if (candidateBaseClusterScore > Minimal_Base_Cluster_Score)
        {
            add a base cluster to the list of base clusters;
        }
    }
}

/* Phase 2. Merge base clusters */
build a graph where nodes are base clusters and there is a link between node A and B if and
only if the number of common documents indexed by A and B is greater than the
Merge_Threshold;

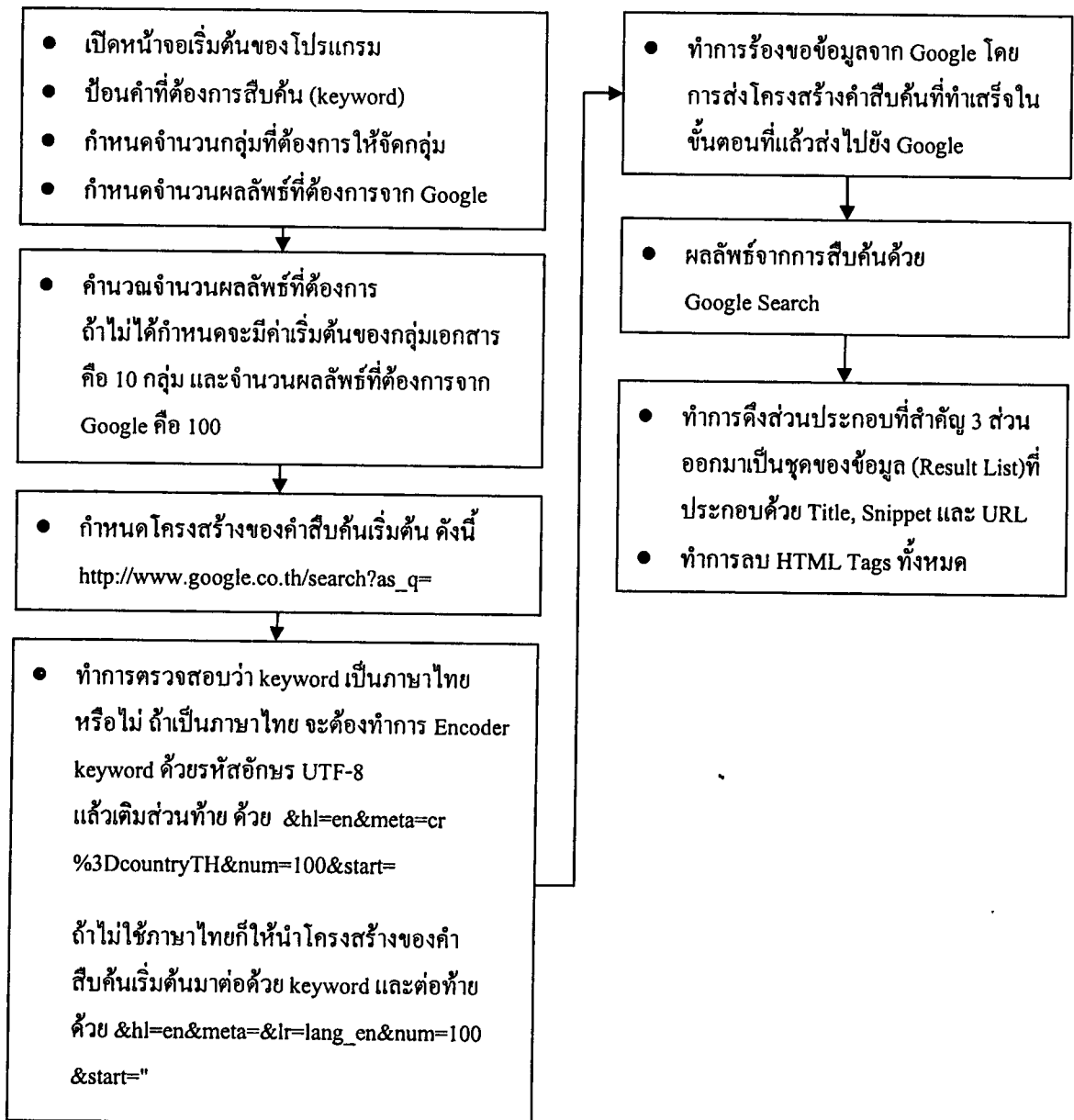
cluster are coherent subgraphs of that graph;

```

รูปที่ 3.2 อัลกอริทึมสำหรับการจัดกลุ่ม Suffix Tree Clustering (Zamir and Etzioni, 1998)

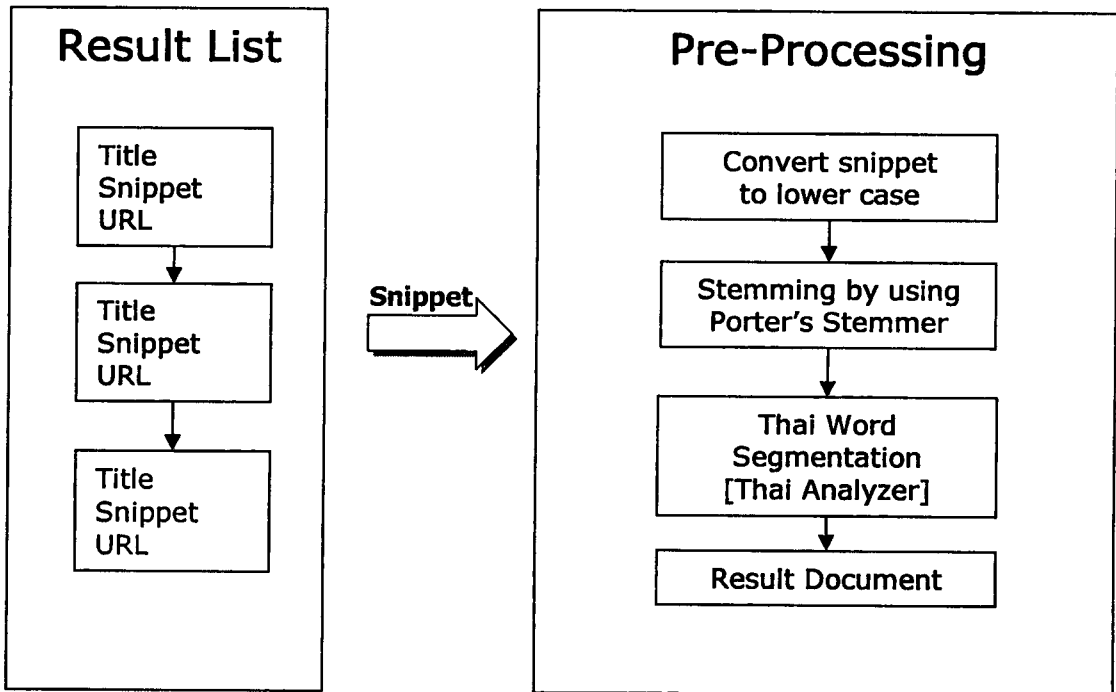
จากรูปที่ 3.1 แสดงโครงสร้างระบบจัดกลุ่มผลการสืบค้น และรูปที่ 3.2 แสดงอัลกอริทึมในการจัดกลุ่มข้างต้นสามารถเขียน System Flow Chart ได้ดังนี้

1. ผู้ใช้ป้อนคำที่ต้องการสืบค้นแล้วกดปุ่มค้นหา ถ้าคำสืบค้นเป็นภาษาไทยระบบจะทำการแปลงคำสืบค้นให้อยู่ในรูปของรหัส UTF-8
2. เมื่อได้รับผลลัพธ์คืนจากเครื่องมือที่ใช้ในการสืบค้นข้อมูล (Search Engine) ระบบจะทำการดึงส่วนประกอบที่สำคัญ 3 ส่วนออกมา คือ Title Snippet และ URL ของแต่ละ Title



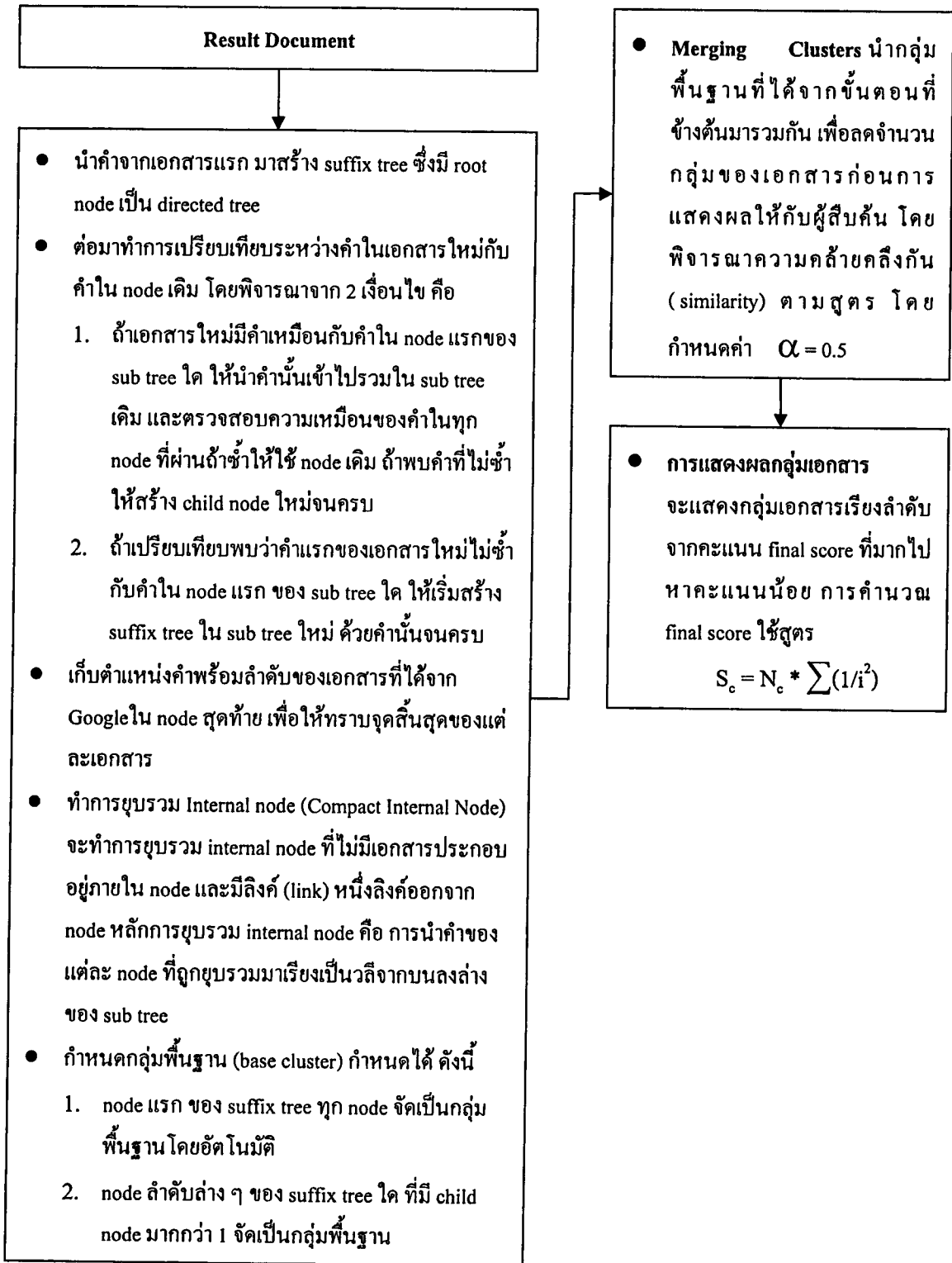
รูปที่ 3.3 ขั้นตอนการนำเข้าข้อมูลจากการ Google Search Engine

3. หลังจากนั้นสร้างรายการของเอกสาร (Result List) ซึ่งประกอบด้วย Titles Snippets URLs ก่อนนำเข้าสู่กระบวนการการจัดกลุ่ม
4. Pre-processing กระบวนการนี้ เริ่มต้นจากการกำจัดสิ่งที่ไม่ใช่คำ (Tokenization) และ คำหยุด (Stop word) เปลี่ยนคำให้เป็นตัวอักษรพิมพ์เล็ก (Lower Case) ทั้งหมด หลังจากนั้นเปลี่ยนรูปของคำให้กลับมาอยู่ในรูปของรากศัพท์ โดยใช้ Porter's Stemming Algorithm



รูปที่ 3.4 ขั้นตอนกระบวนการทำ Pre-Processing

5. สร้าง suffix tree ซึ่งมี root node เป็น directed tree
6. นำกลุ่มพื้นฐานที่ได้ (base cluster) มารวมกัน (Merging Clusters) เพื่อลดจำนวนของกลุ่มก่อนการแสดงผล
7. การจัดลำดับกลุ่มเอกสารโดยใช้สูตร $S_c = N_c * \sum (1/i^2)$

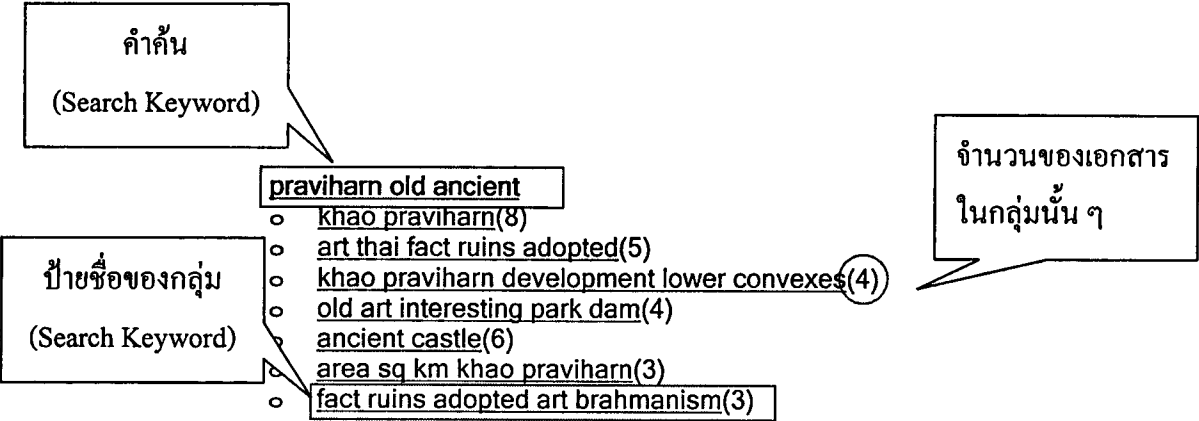


รูปที่ 3.5 ขั้นตอนกระบวนการจัดกลุ่มด้วยอัลกอริทึมฮัฟฟิงทรี

8. ป้ายชื่อกลุ่ม (Labeling Clusters)

การจัดกลุ่มผลการสืบค้นด้วยอัลกอริทึมซัพฟิฟิกทรี (suffix tree algorithm) มีประโยชน์ คือ ผู้ใช้สามารถเข้าถึงข้อมูลที่ต้องการจากโหนดในซัพฟิฟิกทรี ซึ่งภายในโหนดจะประกอบด้วยเอกสารที่เกี่ยวข้องหรือมีส่วนที่คล้ายคลึงกัน

ตัวอย่างป้ายชื่อกลุ่มจากเว็บไซต์

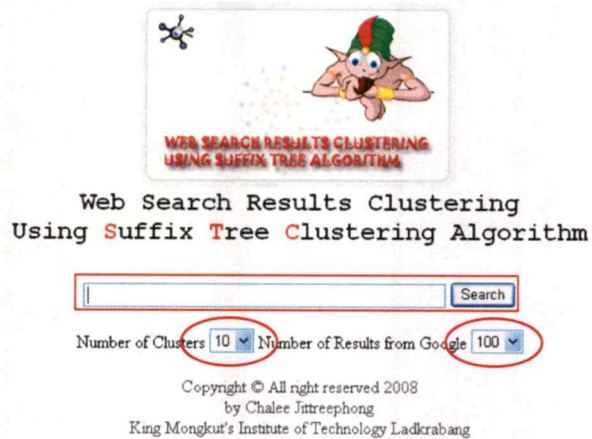


ในบทต่อไปจะกล่าวถึง การทดลองและผลการทดลอง โดยใช้ข้อมูลที่เป็นภาษาไทยและภาษาอังกฤษมาทดลองกับกระบวนการจัดกลุ่มผลการสืบค้นด้วยอัลกอริทึมซัพฟิฟิกทรี

บทที่ 4

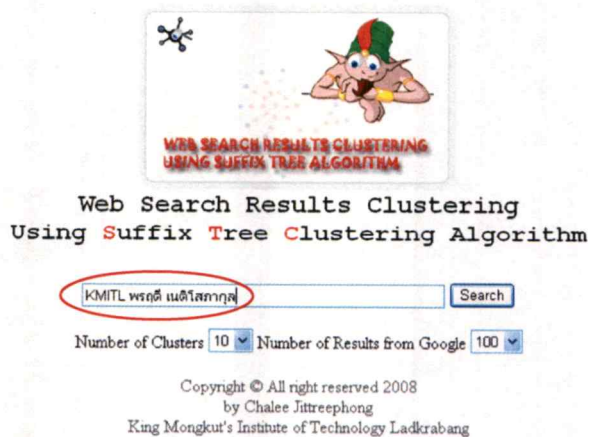
ผลการวิจัย

เมื่อเปิดโปรแกรมเว็บเบราว์เซอร์ (ซึ่งในที่นี้ใช้ Internet Explorer) และพิมพ์ <http://127.0.0.1:8080/clustering/start.jsp> เข้ามาที่หน้าหลักของระบบจัดกลุ่มผลการสืบค้น จะพบหน้าจอดังรูปที่ 4.1



รูปที่ 4.1 หน้าจอเริ่มต้นของระบบ

รูปที่ 4.1 แสดงส่วนประกอบของระบบซึ่งประกอบด้วย 3 ส่วนดังนี้ ส่วนที่ 1 คือ ส่วนผู้ใช้งานกรอกคำที่ต้องการสืบค้น ส่วนที่ 2 คือ จำนวนของกลุ่มที่ผู้ใช้งานต้องการให้ระบบจัดกลุ่มให้ ส่วนที่ 3 จำนวนผลลัพธ์หรือเอกสารที่ต้องการดึงเข้ามาจาก Google Search Engine ตัวอย่างหากผู้ใช้งานต้องการค้นหาชื่อ อาจารย์ที่ชื่อ พรฤดี เนติโสภาคกุล ที่สอนอยู่ที่สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง (KMITL) สามารถทำได้ดังนี้



รูปที่ 4.2 การกรอกคำค้น (keyword) ในช่อง textfield ที่เตรียมไว้



Web Search Results Clustering Using Suffix Tree Clustering Algorithm

KMITL พระภูมิ เนติโสภากุล

Number of Clusters Number of Results from Google

Copyright © All right reserved 2008
by Chalee Jittreephong
King Mongkut's Institute of Technology Ladkrabang

รูปที่ 4.3 การเลือกจำนวนกลุ่มที่ต้องการ



Web Search Results Clustering Using Suffix Tree Clustering Algorithm

KMITL พระภูมิ เนติโสภากุล

Number of Clusters Number of Results from Google

Copyright © All right reserved 2008
by Chalee Jittreephong
King Mongkut's Institute of Technology Ladkrabang

รูปที่ 4.4 การเลือกจำนวนผลลัพธ์ที่ต้องการจาก Google



Web Search Results Clustering Using Suffix Tree Clustering Algorithm



KMITL พระภูมิ เนติโสภากุล

Number of Clusters Number of Results from Google

Copyright © All right reserved 2008
by Chalee Jittreephong
King Mongkut's Institute of Technology Ladkrabang

รูปที่ 4.5 การทำงานหลังการกดปุ่ม Search

ผลลัพธ์การสืบค้นคำว่า “KMITL พรฤดี เนติโสภาคกุล” ซึ่งผลลัพธ์ที่ได้มีทั้งหมด

11 รายการ ดังนี้

- 1 intranet.it.kmitl.ac.th/km/upload/news_files/expert.doc
E-Mail : noppornit.kmitl. เบอร์ติดต่อ. 02-7372551-4 Ext. 531. ประวัติการศึกษา ... พรฤดี เนติโสภาคกุล.
Name - Surname : Asst.Prof. Dr.Ponrudee Netisopakul ...
http://intranet.it.kmitl.ac.th/km/upload/news_files/expert.doc
- 2 [คณะเทคโนโลยีสารสนเทศ สจล. - งานวิจัย - ประชมนานาชาติ](http://www.it.kmitl.ac.th/research.php?page=meetinginter)
พรฤดี เนติโสภาคกุล, The 7th IEEE International Symposium on National Performance
Measurement of Pilot Global Distance Training System at **KMITL**, a. ...
<http://www.it.kmitl.ac.th/research.php?page=meetinginter>
- 3 www.kmitl.ac.th/~klchartc
1. วิศวกรรมซอฟต์แวร์ แปลและเรียบเรียงโดย ผศ.ดร.พรฤดี เนติโสภาคกุล ... Email:
KLCHARTC@KMITL.ac.th. Since October 7, 2002. Last Updated: September 1, 2008 ...
<http://www.kmitl.ac.th/~klchartc/>
- 4 [บทความวิจัยของบุคลากร สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหาร ...](http://www.research.rector.kmitl.ac.th/Citation2551.html)
พรฤดี เนติโสภาคกุล, Proceedings of the 13th International Symposium on Software Reliability
Engineering Page 183 November 12-15 2003, อาจารย์ ...
<http://www.research.rector.kmitl.ac.th/Citation2551.html>
- 5 [คณะเทคโนโลยีสารสนเทศ สจล. - งานวิจัย](http://www.it.kmitl.ac.th/research.php?page=research)
พรฤดี เนติโสภาคกุล. การวัดความซับซ้อนของการพัฒนาระบบสารสนเทศเชิงวัตถุ, อ. ... Management
Model for the Faculty of Information Technology, (**KMITL**), ผศ.ดร. ...
<http://www.it.kmitl.ac.th/research.php?page=research>
- 6 [ขอแสดงความยินดีแด่](http://www.reccit.kmitl.ac.th/ENGLISH/MENU/newsletter/Newsletter4-4_April05.pdf)
พรฤดี เนติโสภาคกุล. หัวหน้าห้องปฏิบัติการวิจัย Information Science โทร.02739 2427-8 ภายใน
3310-11 ReCCITwebsite: <http://www.reccit.kmitl.ac.th> ...
http://www.reccit.kmitl.ac.th/ENGLISH/MENU/newsletter/Newsletter4-4_April05.pdf
- 7 [ข่าวสาร International Micro Robot Maze Contest 2004](http://www.reccit.kmitl.ac.th/ENGLISH/MENU/newsletter/Newsletter3-6_Dec04.pdf)
พรฤดี เนติโสภาคกุล ห้องปฏิบัติการวิจัยศาสตร์ข้อมูล ... ReCCITwebsite: <http://www.reccit.kmitl.ac.th>
Email: reccit@kmitl.ac.th ...
http://www.reccit.kmitl.ac.th/ENGLISH/MENU/newsletter/Newsletter3-6_Dec04.pdf
- 8 [Sheet1](http://www.agro.cmu.ac.th/department/fe/activity/NSTDA_50/List.xls)
133, 132, พรฤดี, เนติโสภาคกุล, -, สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง, 5, 02-737-
2551-4 ต่อ 530, 089-790-8120, ponrudee@it.kmitl.ac.th ...
http://www.agro.cmu.ac.th/department/fe/activity/NSTDA_50/List.xls
- 9 [Welcome to Central Library.KMITL.](http://www.lib.kmitl.ac.th/newbook/acquisition.asp?month=4&year=2007&lang=TH&libraryid=1)
Central Library,Library,**KMITL**,OPAC,INNOPAC,THAILAND,Bangko k. ... พรฤดี เนติโสภาคกุล.
T064377; รายงานโครงการวิจัยประจำปีงบประมาณ 2546 เรื่อง ...
<http://www.lib.kmitl.ac.th/newbook/acquisition.asp?month=4&year=2007&lang=TH&libraryid=1>
- 10 [Welcome to Central Library.KMITL.](http://www.lib.kmitl.ac.th/newbook/acquisition.asp?month=8&year=2007&lang=TH&libraryid=1)
Central Library,Library,**KMITL**,OPAC,INNOPAC,THAILAND,Bangko k. ... พรฤดี เนติโสภาคกุล.
T067398 ...
<http://www.lib.kmitl.ac.th/newbook/acquisition.asp?month=8&year=2007&lang=TH&libraryid=1>
- 11 [ชื่อ คณบดี ตำแหน่ง หลักสูตร สาขาวิชา หัวหน้าที่ สถานภาพ Amarottam ...](http://www.graduate.kmitl.ac.th/teacher/it.htm)
พรฤดี เนติโสภาคกุล, ป.เอก, ผศ. วท.ม./ปร.ด. เทคโนโลยีสารสนเทศ, สอน/สอน/ควบคุม, อ.ประจำ. พิเศษ
ม่วงนวล, ป.เอก, ผศ. วท.ม. เทคโนโลยีสารสนเทศ, สอน/สอน, อ.พิเศษ ...
<http://www.graduate.kmitl.ac.th/teacher/it.htm>

นำผลลัพธ์ดังกล่าวมาผ่านกระบวนการ Stemming ด้วย Porter's Stemming Algorithm จะได้ชุดของเอกสารที่แต่ละเอกสารประกอบด้วยคำต่าง ๆ ดังนี้

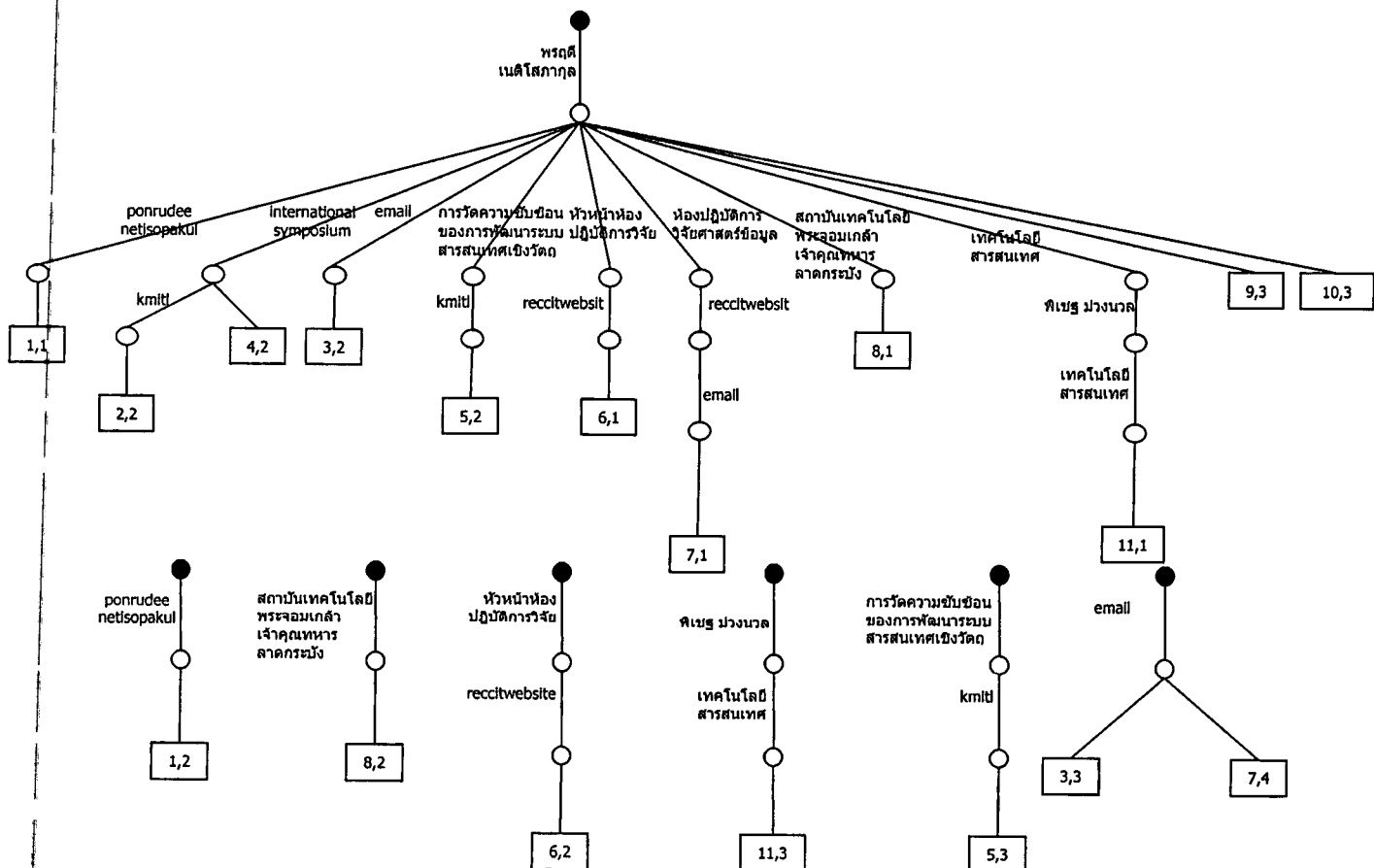
ตารางที่ 4.1 แสดงข้อความหลังผ่านกระบวนการ Stemming

	Original Word	Stemmed Word
1	intranet.it.kmitl.ac.th/km/upload/news_files/expert.doc E-Mail : noppornit.kmitl. เบอร์ติดต่อ. 02-7372551-4 Ext. 531. ประวัติการศึกษา ... พรฤดี เนติโสภาคกุล . Name - Surname : Asst.Prof. Dr.Ponrudee Netisopakul ... http://intranet.it.kmitl.ac.th/km/upload/news_files/expert.doc	พรฤดี เนติโสภาคกุล, ponrudee netisopakul
2	คณะเทคโนโลยีสารสนเทศ สจล. - งานวิจัย - ประชุมนานาชาติ พรฤดี เนติโสภาคกุล , The 7th IEEE International Symposium on National Performance Measurement of Pilot Global Distance Training System at KMITL , อ. ... http://www.it.kmitl.ac.th/research.php?page=meetinginter	คณะเทคโนโลยีสารสนเทศ สจล, พรฤดี เนติโสภาคกุล, international symposium, kmitl
3	www.kmitl.ac.th/~klchartc 1. วิศวกรรมซอฟต์แวร์ แปลและเรียบเรียงโดย ผศ.ดร. พรฤดี เนติโสภาคกุล ... Email: KLCHARTC@KMITL.ac.th. Since October 7, 2002. Last Updated: September 1, 2008 ... http://www.kmitl.ac.th/~klchartc/	วิศวกรรมซอฟต์แวร์, พรฤดี เนติโสภาคกุล, email
4	บทความวิจัยของบุคลากร สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหาร ... พรฤดี เนติโสภาคกุล , Proceedings of the 13th International Symposium on Software Reliability Engineering Page 183 November 12-15 2003, อาจารย์ ... http://www.research.rector.kmitl.ac.th/Citation2551.html	สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหาร, พรฤดี เนติโสภาคกุล, international symposium
5	คณะเทคโนโลยีสารสนเทศ สจล. - งานวิจัย พรฤดี เนติโสภาคกุล . การวัดความซับซ้อนของการพัฒนาระบบสารสนเทศเชิงวัตถุ, อ. ... Management Model for the Faculty of Information Technology, KMITL), ผศ.ดร. ... http://www.it.kmitl.ac.th/research.php?page=research	คณะเทคโนโลยีสารสนเทศ สจล, พรฤดี เนติโสภาคกุล, การวัดความซับซ้อนของการพัฒนาระบบสารสนเทศเชิงวัตถุ, kmitl
6	ขอแสดงความยินดีแก่ พรฤดี เนติโสภาคกุล . หัวหน้าห้องปฏิบัติการวิจัย Information Science โทร.02739 2427-8 ภายใน 3310-11 ReCCITwebsite: http://www.reccit.kmitl.ac.th ... http://www.reccit.kmitl.ac.th/ENGLISH/MENU/newsletter/Newsletter4-4_April05.pdf	พรฤดี เนติโสภาคกุล, หัวหน้าห้องปฏิบัติการวิจัย, reccitwebsite
7	ข่าวสาร International Micro Robot Maze Contest 2004 พรฤดี เนติโสภาคกุล ห้องปฏิบัติการวิจัยศาสตร์ข้อมูล ... ReCCITwebsite: http://www.reccit.kmitl.ac.th Email: reccit@kmitl.ac.th ... http://www.reccit.kmitl.ac.th/ENGLISH/MENU/newsletter/Newsletter3-6_Dec04.pdf	พรฤดี เนติโสภาคกุล, ห้องปฏิบัติการวิจัยศาสตร์ข้อมูล, reccitwebsite, email
8	Sheet1 133, 132, พรฤดี, เนติโสภาคกุล , -, สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง, 5, 02-737-2551-4 ต่อ 530, 089-790-8120, ponrudee@it.kmitl.ac.th ... http://www.agro.cmu.ac.th/department/fe/activiy/NSTDA_50/List.xls	พรฤดี เนติโสภาคกุล, สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

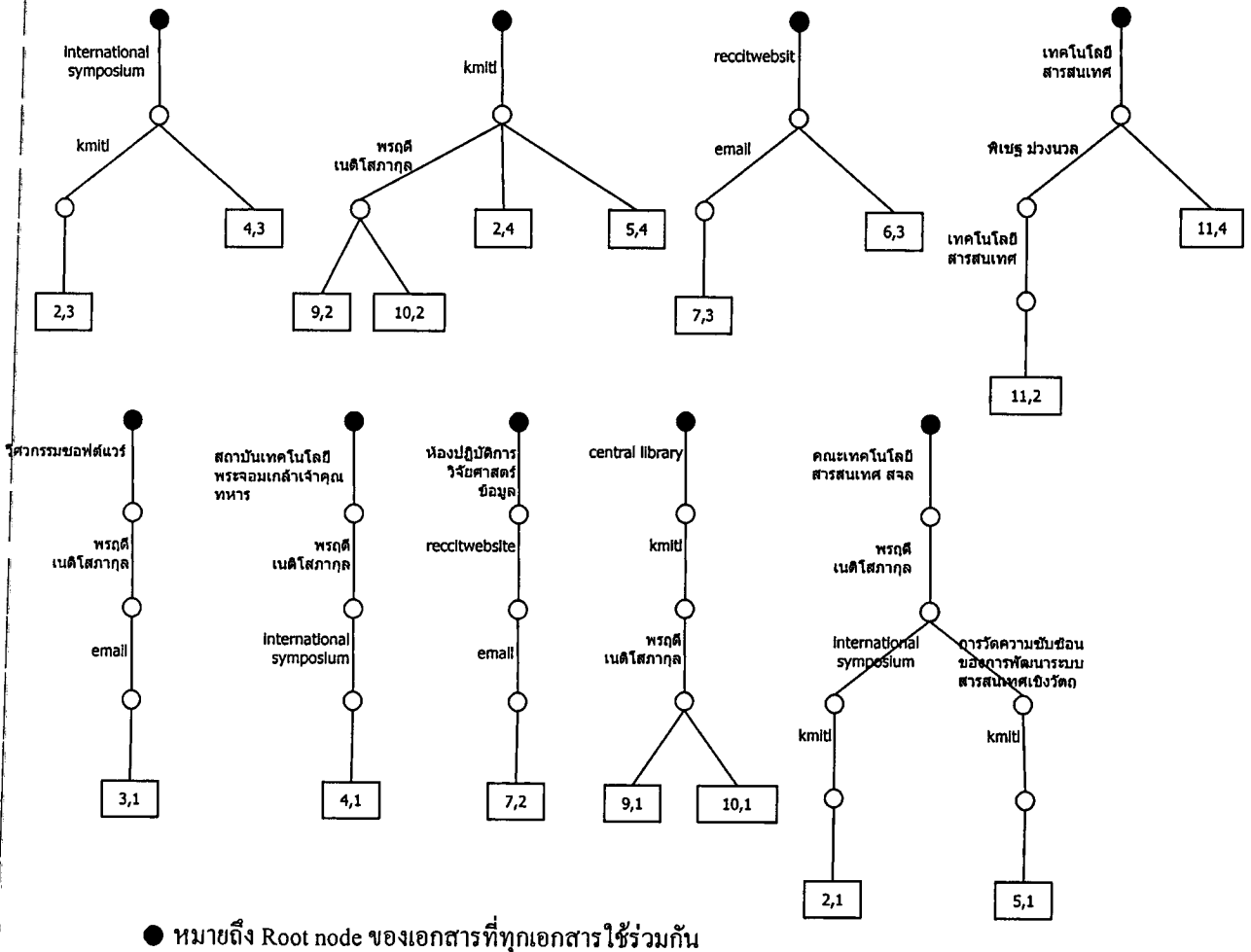
ตารางที่ 4.1 (ต่อ)

	Original Word	Stemmed Word
9	Welcome to Central Library,KMITL. Central Library,Library,KMITL,OPAC,INNOPAC,THAILAN D,Bangko k. ... พรฤดี เนติโสภากุล. T064377; รายงานโครงการวิจัยประจำปีงบประมาณ 2546 เรื่อง ... http://www.lib.kmitl.ac.th/newbook/acquisition.a sp?month=4&year=2007&lang=TH&libraryid=1	central library, kmitl, พรฤดี เนติโสภากุล
10	Welcome to Central Library,KMITL. Central Library,Library,KMITL,OPAC,INNOPAC,THAILAN D,Bangko k. ... พรฤดี เนติโสภากุล. T067398 ... http://www.lib.kmitl.ac.th/newbook/acquisition.a sp?month=8&year=2007&lang=TH&libraryid=1	central library, kmitl, พรฤดี เนติโสภากุล
11	ชื่อ คณวดี ตำแหน่ง หลักสูตร สาขาวิชา ทำหน้าที่ สถานภาพ Amarottam ... พรฤดี เนติโสภากุล, ป.เอก, ผศ. วท.ม./ปร.ด. เทคโนโลยีสารสนเทศ, สอน/สอน/ควบคุม, อ.ประจำ. พิเชษฐ ม่วงนวล, ป.เอก, ผศ. วท.ม. เทคโนโลยี สารสนเทศ, สอน/สอน, อ.พิเศษ ... http://www.graduate.kmitl.ac.th/teacher/it.htm	พรฤดี เนติโสภากุล, เทคโนโลยีสารสนเทศ, พิเชษฐ ม่วงนวล, เทคโนโลยีสารสนเทศ

นำผลลัพธ์ที่ผ่านกระบวนการ Stemming มาสร้างเป็น suffix tree ซึ่งมี root node เป็น directed tree ได้ดังนี้



รูปที่ 4.6 ตัวอย่างการสร้าง Suffix Tree ของเอกสารจำนวน 11 เอกสาร



รูปที่ 4.6 ตัวอย่างการสร้าง Suffix Tree ของเอกสารจำนวน 11 เอกสาร (ต่อ)

ทำการยุบรวม Internal node (Compact Internal Node) เพื่อลดจำนวน node กำหนดกลุ่มพื้นฐาน (base cluster) ของเอกสาร แล้วนำกลุ่มพื้นฐานที่ได้ (base cluster) มารวมกัน (Merging Clusters) เพื่อลดจำนวนของกลุ่มก่อนการแสดงผลให้กับผู้สืบค้น พร้อมกำหนดป้ายชื่อของกลุ่มที่นำมารวมกัน การรวมกลุ่มพิจารณาความคล้ายคลึงกัน (similarity) ของเอกสาร โดยกำหนดค่า $\alpha = 0.5$ การแสดงผลกลุ่มเอกสารจะแสดงกลุ่มเอกสารเรียงลำดับจากคะแนน final score ที่มากไปหาคะแนนน้อยรวมทั้งพิจารณาความยาวของป้ายชื่ออีกด้วย

Web Search Results Clustering Using Suffix Tree Clustering Algorithm

KMITL พรฤดี เนติโสภากุล

Number of Clusters: 10 Number of Results from Google: 100

Search Results of: **KMITL พรฤดี เนติโสภากุล**
(First 100 Results from Google)
Cluster of: **None**
Total Time 3.047 seconds

KMITL พรฤดี เนติโสภากุล (11)

- Central Library (2)
- คณะเทคโนโลยีสารสนเทศ สจล. (2)
- International Symposium (2)
- ReCCITwebsite (2)
- Email (2)
- (Other topics) (3)

1. intranet.it.kmitl.ac.th/km/upload/news_files/expert.doc
E-Mail : noppornit.kmitl. เนติโสภากุล. 02-7372551-4 Ext. 531. ประวัติการศึกษา ... พรฤดี เนติโสภากุล. Name - Surname : Asst.Prof. Dr.Ponrudee Netisopakul ...
http://intranet.it.kmitl.ac.th/km/upload/news_files/expert.doc
2. [คณะเทคโนโลยีสารสนเทศ สจล. - งานวิจัย - ประชมนานาชาติ](http://www.it.kmitl.ac.th/research.php?page=meetinginter)
พรฤดี เนติโสภากุล, The 7th IEEE International Symposium on National Performance Measurement of Pilot Global Distance Training System at **KMITL**, อ. ...
<http://www.it.kmitl.ac.th/research.php?page=meetinginter>
3. www.kmitl.ac.th/~klchartc
1. วิศวกรรมซอฟต์แวร์ แปลและเรียบเรียงโดย ผศ.ดร.พรฤดี เนติโสภากุล ... Email: KLCHARTC@KMITL.ac.th. Since October 7, 2002. Last Updated: September 1, 2008 ...
<http://www.kmitl.ac.th/~klchartc/>
4. [บทความวิจัยของคณาจารย์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหาร](http://www.research.ector.kmitl.ac.th/Citation2551.html) ...
พรฤดี เนติโสภากุล, Proceedings of the 13th International Symposium on Software Reliability Engineering Page 183 November 12-15 2003, อางกรณ์ ...
<http://www.research.ector.kmitl.ac.th/Citation2551.html>
5. [คณะเทคโนโลยีสารสนเทศ สจล. - งานวิจัย](http://www.it.kmitl.ac.th/research.php?page=research)
พรฤดี เนติโสภากุล. การวัดความซับซ้อนของการพัฒนาระบบสารสนเทศเชิงวัตถุ, อ. ... Management Model for the Faculty of Information Technology, **KMITL**), ผศ.ดร. ...
<http://www.it.kmitl.ac.th/research.php?page=research>

Copyright © All right reserved 2008 by Chalee Jitreephong, KMITL

รูปที่ 4.7 ผลลัพธ์ที่ได้จากการสืบค้นและทำการจัดกลุ่ม

ผลการจัดกลุ่มเอกสารข้างต้นสามารถจัดกลุ่มได้ทั้งหมด 6 กลุ่ม และถ้าผู้ใช้ต้องการให้แสดงผลลัพธ์ในกลุ่มใด ผู้ใช้เพียงคลิกบนชื่อของกลุ่มเอกสารนั้น โปรแกรมจะทำการประมวลผลและแสดงเอกสารในกลุ่มนั้น ๆ โดยเอกสารที่แสดงในแต่ละกลุ่มมีดังนี้

กลุ่มที่ 1 Central Library (2)

- 9 [Welcome to Central Library,KMITL.](http://www.lib.kmitl.ac.th/newbook/acquisition.asp?month=4&year=2007&lang=TH&libraryid=1)
Central Library,Library,KMITL,OPAC,INNOPAC,THAILAND,Bangko k. ... พรฤดี เนติโสภากุล.
T064377; รายงานโครงการวิจัยประจำปีงบประมาณ 2546 เรื่อง ...
<http://www.lib.kmitl.ac.th/newbook/acquisition.asp?month=4&year=2007&lang=TH&libraryid=1>
- 10 [Welcome to Central Library,KMITL.](http://www.lib.kmitl.ac.th/newbook/acquisition.asp?month=8&year=2007&lang=TH&libraryid=1)
Central Library,Library,KMITL,OPAC,INNOPAC,THAILAND,Bangko k. ... พรฤดี เนติโสภากุล.
T067398 ...
<http://www.lib.kmitl.ac.th/newbook/acquisition.asp?month=8&year=2007&lang=TH&libraryid=1>

กลุ่มที่ 2 คณะเทคโนโลยีสารสนเทศ สจล (2)

- 2 [คณะเทคโนโลยีสารสนเทศ สจล. - งานวิจัย - ประชมนานาชาติ](http://www.it.kmitl.ac.th/research.php?page=meetinginter)
พรฤดี เนติโสภากุล, The 7th IEEE International Symposium on National Performance Measurement of Pilot Global Distance Training System at **KMITL**, อ. ...
<http://www.it.kmitl.ac.th/research.php?page=meetinginter>
- 5 [คณะเทคโนโลยีสารสนเทศ สจล. - งานวิจัย](http://www.it.kmitl.ac.th/research.php?page=research)
พรฤดี เนติโสภากุล. การวัดความซับซ้อนของการพัฒนาระบบสารสนเทศเชิงวัตถุ, อ. ... Management Model for the Faculty of Information Technology, **KMITL**), ผศ.ดร. ...
<http://www.it.kmitl.ac.th/research.php?page=research>

กลุ่มที่ 3 International Symposium (2)

- 2 [คณะเทคโนโลยีสารสนเทศ สจล. - งานวิจัย - ประชมนานาชาติ](#)
พรฤดี เนติโสภากุล, The 7th IEEE **International Symposium** on National Performance Measurement of Pilot Global Distance Training System at **KMITL**, อ. ...
<http://www.it.kmitl.ac.th/research.php?page=meetinginter>
- 4 [บทความวิจัยของบุคลากร สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหาร ...](#)
พรฤดี เนติโสภากุล, Proceedings of the 13th **International Symposium** on Software Reliability Engineering Page 183 November 12-15 2003, อาจารย์ ...
<http://www.research.rector.kmitl.ac.th/Citation2551.html>

กลุ่มที่ 4 ReCCITwebsite (2)

- 6 [ขอแสดงความยินดีแด่](#)
พรฤดี เนติโสภากุล. หัวหน้าห้องปฏิบัติการวิจัย Information Science โทร.02739 2427-8 ภายใน 3310-11 **ReCCITwebsite**: <http://www.reccit.kmitl.ac.th> ...
http://www.reccit.kmitl.ac.th/ENGLISH/MENU/newsletter/Newsletter4-4_April05.pdf
- 7 [ข่าวสาร International Micro Robot Maze Contest 2004](#)
พรฤดี เนติโสภากุล ห้องปฏิบัติการวิจัยศาสตร์ข้อมูล ... **ReCCITwebsite**:
<http://www.reccit.kmitl.ac.th> Email: reccit@kmitl.ac.th ...
http://www.reccit.kmitl.ac.th/ENGLISH/MENU/newsletter/Newsletter3-6_Dec04.pdf

กลุ่มที่ 5 Email (2)

- 3 www.kmitl.ac.th/~klchartc
1. วิศวกรรมซอฟต์แวร์ แปลและเรียบเรียงโดย ผศ.ดร.พรฤดี เนติโสภากุล ... **Email**:
KLCHARTC@KMITL.ac.th. Since October 7, 2002. Last Updated: September 1, 2008 ...
<http://www.kmitl.ac.th/~klchartc/>
- 7 [ข่าวสาร International Micro Robot Maze Contest 2004](#)
พรฤดี เนติโสภากุล ห้องปฏิบัติการวิจัยศาสตร์ข้อมูล ... **ReCCITwebsite**: <http://www.reccit.kmitl.ac.th>
Email: reccit@kmitl.ac.th ...
http://www.reccit.kmitl.ac.th/ENGLISH/MENU/newsletter/Newsletter3-6_Dec04.pdf

กลุ่มที่ 6 (Other topics) (3)

- 1 intranet.it.kmitl.ac.th/km/upload/news_files/expert.doc
E-Mail : noppornit.kmitl. เบอร์ติดต่อ. 02-7372551-4 Ext. 531. ประวัติการศึกษา ... **พรฤดี เนติโสภากุล**.
Name - Surname : Asst.Prof. Dr.Ponrudee Netisopakul ...
http://intranet.it.kmitl.ac.th/km/upload/news_files/expert.doc
- 8 [Sheet1](#)
133, 132, **พรฤดี, เนติโสภากุล**, -, สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง, 5, 02-737-2551-4 ต่อ 530, 089-790-8120, ponrudee@it.kmitl.ac.th ...
http://www.agro.cmu.ac.th/departement/fe/activity/NSTDA_50/List.xls
- 11 [ชื่อ คณวดี ตำแหน่ง หลักสูตร สาขาวิชา ทำหน้าที่ สถานภาพ Amarottam ...](#)
พรฤดี เนติโสภากุล, ป.เอก, ผศ. วท.ม./ปร.ด. เทคโนโลยีสารสนเทศ, สอน/สอบ/ควบคุม, อ.ประจำ. พิเชฐ ม่วงนวล, ป.เอก, ผศ. วท.ม. เทคโนโลยีสารสนเทศ, สอน/สอบ, อ.พิเศษ ...
<http://www.graduate.kmitl.ac.th/teacher/it.htm>

ตัวอย่างการจัดกลุ่มโดยใช้คำสืบค้นที่มีลักษณะเป็นชื่อ (Entity Name) เช่น คำว่า “Thailand” จะได้กลุ่มเอกสารทั้งสิ้นจำนวน 16 กลุ่มเอกสาร ดังนี้ กลุ่ม Travel Guide จำนวน 8 เอกสาร กลุ่ม Travel จำนวน 16 เอกสาร กลุ่ม Information จำนวน 13 เอกสาร กลุ่ม Hotels จำนวน 13 เอกสาร กลุ่ม Phuket จำนวน 7 เอกสาร กลุ่ม Home จำนวน 9 เอกสาร กลุ่ม Travel Information จำนวน 5 เอกสาร กลุ่ม Provides จำนวน 8 เอกสาร กลุ่ม Lonely Planet จำนวน 4 เอกสาร กลุ่ม Thai จำนวน 7 เอกสาร กลุ่ม Map จำนวน 7 เอกสาร กลุ่ม Bangkok จำนวน 7 เอกสาร กลุ่ม Links จำนวน 6 เอกสาร กลุ่ม Asia จำนวน 6 เอกสาร กลุ่ม Country จำนวน 6 เอกสาร กลุ่ม (Other topics) จำนวน 39 เอกสาร ซึ่งในแต่ละกลุ่มก็มีเอกสารที่ซ้อนทับกัน คือ เอกสารหนึ่งเอกสารสามารถอยู่ได้หลายกลุ่ม

รูปที่ 4.8 ผลลัพธ์ที่ได้จากการสืบค้นด้วยคำที่มีลักษณะเป็นชื่อ (Entity Name)

เห็นได้ว่าการจัดกลุ่มผลลัพธ์ของการสืบค้นให้อยู่ในรูปของกลุ่มตามเนื้อหาที่เกี่ยวข้องกัน ช่วยให้การแสดงผลลัพธ์ของการสืบค้นมีประสิทธิภาพมากขึ้น ช่วยให้ผู้สืบค้นประหยัดเวลาโดยผู้ใช้เพียงเลือกกลุ่มของเนื้อหาที่สนใจ แทนที่จะเลือกดูผลการสืบค้นข้อมูลทีละรายการ นอกจากนั้นยังเพิ่มโอกาสที่ผลการสืบค้นในลำดับหลัง ๆ จะถูกนำมาใช้ เพราะถูกนำมาแสดงในกลุ่มที่มีเนื้อหาที่เกี่ยวข้องกัน

สำหรับการประเมินผลการจัดกลุ่มผลลัพธ์การสืบค้น โครงการพัฒนาระบบนี้ใช้เครื่องมือในการประสิทธิภาพ 2 ชนิดด้วยกัน คือ การวัดอัตราการทับซ้อนของเอกสาร (Overlap) และ การวัดประสิทธิภาพด้านความครอบคลุมการจัดกลุ่ม (Coverage)

การวัดอัตราการทับซ้อนของเอกสาร (Overlap) คือ การวัดจำนวนเอกสารที่ปรากฏมากกว่าหนึ่งกลุ่มเอกสาร ดังแสดงรายละเอียดในสมการที่ 2.9

การวัดประสิทธิภาพด้านความครอบคลุมการจัดกลุ่ม (Coverage) คือ การคำนวณจำนวนผลการสืบค้นที่สามารถจัดกลุ่มได้จากผลการสืบค้นทั้งหมด ดังแสดงรายละเอียดในสมการที่ 2.10

ตารางที่ 4.2 แสดงตัวอย่างการคำนวณอัตราการทับซ้อนของเอกสาร (Overlap) และความครอบคลุมการจัดกลุ่ม (Coverage) คำว่า “Thailand” ซึ่งระบบค้นคืนได้ทั้งหมด 100 เอกสาร และสามารถจัดกลุ่มได้ทั้งหมด 16 กลุ่ม ดังนี้

คำสืบค้น	จำนวนเอกสาร ทั้งหมด	ชื่อกลุ่ม	จำนวนเอกสาร ในกลุ่ม
Thailand	100	1. Travel Guide	8
		2. Travel	16
		3. Information	13
		4. Hotels	13
		5. Phuket	7
		6. Home	9
		7. Travel Information	5
		8. Provides	8
		9. Lonely Planet	4
		10. Thai	7
		11. Map	7
		12. Bangkok	7
		13. Links	6
		14. Asia	6
		15. Country	6
		16. (Other topics)	39
		รวม	161
Overlap = (161 ÷ 100) – 1 = 0.61			
Coverage = (100-39) ÷ 100 = 0.61			

เห็นได้ว่าในแต่ละกลุ่มจะมีเอกสารที่เกี่ยวข้องกันปรากฏ ด้วยค่าอัตราการทับซ้อนของเอกสาร (Overlap) 0.61 คือ เอกสารหนึ่งเอกสารสามารถอยู่ได้หลาย ๆ กลุ่ม ซึ่งมีเนื้อหาเกี่ยวข้องกัน นอกจากนั้นระบบยังสามารถนำผลลัพธ์จากการสืบค้นมาทำการจัดกลุ่มโดยคำนึงถึงความสัมพันธ์กับคำสืบค้น (key word) ด้วยอัตราความครอบคลุมของการจัดกลุ่ม (Coverage) 0.61ซึ่งหมายถึงระบบสามารถจัดกลุ่มเอกสารได้ประมาณ 61% จากเอกสารทั้งหมด 100 เอกสาร

ตารางที่ 4.3 แสดงตัวอย่างการคำนวณอัตราการทับซ้อนของเอกสาร (Overlap) และความครอบคลุม
กลุ่มการจัดกลุ่ม (Coverage) ของคำสืบค้นต่าง ๆ

คำสืบค้น	จำนวนเอกสารทั้งหมด	รวมจำนวนเอกสารที่สามารถจัดกลุ่มได้	จำนวนเอกสารที่ไม่สามารถจัดกลุ่มได้ (Other topics)	อัตรา การทับซ้อน (Overlap)	อัตราความ ครอบคลุม (Coverage)
ชนมไทย	94	115	28	0.22	0.70
เขาพระวิหาร	97	116	39	0.20	0.60
ค็อกเทล	100	110	24	0.10	0.72
จากรัวร์	69	92	18	0.33	0.74
ไทยแลนด์พรีเมียร์ลีก	59	83	7	0.41	0.88
นวดแผนไทย	93	121	30	0.30	0.68
บ้านเชียง	95	118	33	0.24	0.65
พรอติ เนดิโสภากุล	15	20	4	0.33	0.73
พรีเมียร์ลีก	94	141	19	0.50	0.80
พันธมิตรประชาชนเพื่อประชาธิปไตย	69	91	18	0.32	0.74
เมตริกซ์	77	89	19	0.15	0.75
โรคมะเร็ง	95	118	29	0.24	0.69
ลอนดอนเกมส์	34	43	5	0.26	0.85
อาหารเพื่อสุขภาพ	97	122	35	0.26	0.64
แอปเปิล	88	109	28	0.24	0.68
Bangalore	100	169	35	0.69	0.65
Beijing Olympics	100	197	23	0.97	0.77
Cinton	96	170	31	0.77	0.68
Data Warehouse	100	121	31	0.21	0.69
Disney	100	173	11	0.73	0.89
Hamburger Crisis	99	147	40	0.50	0.66
Kao Praviharn	28	31	4	0.11	0.86
Lehman Brothers	100	220	15	1.20	0.85
Long-term Mutual Fund	78	120	2	0.54	0.97
McCain	100	238	10	1.38	0.90
Obama	98	219	24	1.23	0.75
Paralympic Games	100	226	10	1.26	0.90
Retirement Mutual Funds	94	177	3	0.88	0.97
Taj Mahal	100	151	24	0.51	0.76
Tom Yam Kung Crisis	80	143	7	0.79	0.91

ตัวอย่างการคำนวณอัตราการทับซ้อนของเอกสาร (Overlap) และความครอบคลุมกลุ่มการจัด
กลุ่ม (Coverage) ของคำสืบค้นข้างต้น เห็นได้ว่าอัตราการทับซ้อนจะมีค่าเฉลี่ยเท่ากับ 0.53 สามารถ
สรุปได้ว่า ภายในกลุ่มที่ระบบได้ทำการจัดกลุ่มมีเอกสารที่มีเนื้อหาเกี่ยวข้องกัน คิดเป็นร้อยละ 53
และมีอัตราการครอบคลุม (Coverage) คิดเป็นร้อยละ 77 สรุปได้ว่า ระบบสามารถจัดกลุ่มเอกสารที่
ค้นคืนได้ในระดับที่น่าพอใจ สำหรับในบทถัดไปจะกล่าวถึงปัญหาและอุปสรรคและข้อเสนอแนะ
สำหรับการพัฒนาระบบงานในอนาคต

บทที่ 5

สรุปผลและข้อเสนอแนะ

5.1 สรุปผลการดำเนินงาน

การจัดกลุ่มผลการสืบค้นบนเว็บไซต์ (Web Search Results Clustering) คือ การนำผลการสืบค้นที่มีความเกี่ยวข้อง (Relevance) หรือมีความคล้ายคลึงกัน (Similarity) มารวมในกลุ่มเดียวกัน เพื่ออำนวยความสะดวกผู้สืบค้น โดยโครงการพัฒนาระบบนี้ได้อัลกอริทึมซัพฟิกทรีในการจัดกลุ่มผลการสืบค้นจาก Google Search Engine โดยผู้ใช้งานสามารถใช้คำสืบค้นทั้งภาษาไทยและภาษาอังกฤษในการสืบค้นและผู้ใช้สามารถที่จะกำหนดจำนวนกลุ่มที่ต้องการให้แสดงผล

5.2 ปัญหาและอุปสรรค

ระบบการจัดกลุ่มผลการสืบค้นโดยใช้อัลกอริทึมซัพฟิกทรีที่พัฒนาขึ้นนี้ เป็น meta-search ที่ยังต้องพึ่งข้อมูลจาก Google Search Engine อยู่ ซึ่งทำให้มีข้อจำกัด คือ

1. ระบบทำงานช้า ต้องรอผลลัพธ์จาก Google Search Engine ก่อนถึงจะทำการจัดกลุ่ม
2. ความสามารถในการกำจัดข้อมูลข่าวสารที่ไม่เกี่ยวข้องออกไปให้มากที่สุด (precision) และความสามารถในการสืบค้นหาข้อมูลข่าวสารสารสนเทศที่มีความเกี่ยวข้องกับสิ่งที่เราสนใจให้ได้มากที่สุด (recall) จะขึ้นอยู่กับผลลัพธ์ของ Google Search Engine ที่นำมาใช้
3. ถ้า Google Search Engine เปลี่ยนแปลงรูปแบบพารามิเตอร์ ตัวระบบจะต้องทำการคอมไพล์ใหม่ มิฉะนั้นจะไม่สามารถทำงานได้

5.3 แนวทางการพัฒนาในอนาคต

1. พัฒนาโปรแกรมให้มีตัวเลือก (Option) ให้ผู้ใช้งานกำหนดชื่อกลุ่มที่ต้องการล่วงหน้าได้
2. พัฒนาโปรแกรม ให้มีความฉลาดในการสร้างชื่อกลุ่มที่เหมาะสมตามเนื้อหาของเว็บในกลุ่มนั้น
3. พัฒนาโปรแกรม ให้เชื่อมต่อกับ APIs ของ Search Engine ต่าง ๆ และสามารถนำข้อมูล (Search Result) มาเก็บไว้ในเครื่องเพื่อการประมวลผลที่รวดเร็ว
4. เนื่องจาก Suffix Tree Clustering มีการจัดกลุ่มเป็นต้นไม้ที่มีลำดับชั้นอยู่แล้ว จึงควรพัฒนาโปรแกรม ให้มีการแสดงผลการจัดกลุ่มแบบมีลำดับชั้นด้วย

บรรณานุกรม

Borch, Hans Olaf. 2006. **On-Line Clustering of Web Search Results.**

[Online]. Available: www.diva-portal.org/diva/getDocument?urn_nbn_no_ntnu_diva-1041-1__fulltext.pdf

Branson, Steve. 2002. **Clustering Web Search Results Using Suffix Tree Methods.**

[Online]. Available: <http://www.stanford.edu/class/archive/cs/cs276a/cs276a.1032/projects/reports/arigreen-sbranson.pdf>

Chim, Hung. 2007. **A New Suffix Tree Similarity Measure for Document Clustering.**

[Online]. Available: <http://www2007.org/papers/paper091.pdf>

Jian-wu, Yang . 2004. **A Chinese Web Page Clustering Algorithm Based on the Suffix Tree.**

[Online]. Available: <http://www.springerlink.com/content/cv71p24q14l32433/fulltext.pdf>

Liu, Bing. 2007. **Web Data Mining Exploring Hyperlinks, Contents, and Usage Data.**

Berlin: Springer.

Zamir, Oren and Etzioni, Oren. 1998. **Web Document Clustering: A Feasibility Demonstration.**

[Online]. Available: <http://www.cs.washington.edu/homes/etzioni/papers/sigir98.pdf>

ภาคผนวก

ตัวอย่าง source code การสร้างคำสืบค้นลงใน URL ของ Google

```
/** Creates a new instance of GoogleCrawler */
public void simpleSearch(String keyword, ArrayList cache, int max, Language lang)
{
    int total = max/100 + ((max%100 > 0) ? 1 : 0);
    callback = new HttpCallback(total, cache);
    try {
        for(int page=0; page<total; page++)
        {
            StringBuffer url = new StringBuffer();
            url.append("http://www.google.co.th/search?as_q=");
            if(lang == Language.Thai) {
                url.append(URLEncoder.encode(keyword, "UTF-8"));
                url.append("&hl=en&meta=cr%3DcountryTH&num=100&start=");
            } else {
                url.append(keyword);
                url.append("&hl=en&meta=&lr=lang_en&num=100&start=");
            }
            url.append("&num=100&hl=th&as_qdr=all&start=");
            url.append(page*100);
            System.out.println(url);
            HttpURLConnection con = new HttpURLConnection(new URL(url.toString()));
            con.RequestHttpGetAsync(new Integer(page), callback);
        }
    }
}
```

ตัวอย่าง source code สำหรับการดึงส่วนประกอบที่สำคัญ 3 ส่วน

```
while ((found = r.HttpContent.indexOf("<h2 class=r><a href=\"", current)) != -1)
{
    current = found + 21;
    found = r.HttpContent.indexOf("\"", current);
    if (found == -1) break;
    //System.out.println(found + "1");

    //URL
    String URL = r.HttpContent.substring(current, found);
    //System.out.println(URL);

    found = r.HttpContent.indexOf("\">", current);
    if (found == -1) break;
    current = found + 5;
    found = r.HttpContent.indexOf("</a></h2>", current);
    if(found == -1) break;
    //System.out.println(found + "2");

    //TITLE
    String Title = r.HttpContent.substring(current, found);
    //System.out.println(Title);

    //found = r.HttpContent.indexOf("<font size=-1>", current);
    found = r.HttpContent.indexOf("<div class=std>", current);
    if (found == -1) break;
    current = found + 15;
    found = r.HttpContent.indexOf("<span class=a>", current);
    if (found == -1) break;
    //System.out.println("google na");
    //SNIPPET
    String Snippet = r.HttpContent.substring(current, found);
    try{
        search.metasearch.Document doc = new search.metasearch.Document(Title, URL, Snippet);
        String TitleNa = doc.getTitle() ;
        String SnippetNa = doc.getSnippet();
        String URLNa = doc.getUrl();
        //System.out.println(URLNa);
        ResultDoc Result = new ResultDoc(TitleNa, SnippetNa, URLNa);
        ResultList.add(Result);
    }
}
```

ตัวอย่าง source code สำหรับการลบแท็ก HTML

```
/** Creates a new instance of document */
public Document(String title, String snippet)
{
    Pattern p = Pattern.compile("<.+?>");
    Pattern r = Pattern.compile("\\&.+?;");
    Pattern q[] = new Pattern[6];
    char cc[] = new char[6];
    String a = "";
    if ( snippet.indexOf("amp") > -1 )
        a = snippet.substring( snippet.indexOf("amp") );
    q[0] = Pattern.compile("&lt;"); cc[0] = '<';
    q[1] = Pattern.compile("&gt;"); cc[1] = '>';
    q[2] = Pattern.compile("&quot;"); cc[2] = '"';
    q[3] = Pattern.compile("&"); cc[3] = '&';
    q[4] = Pattern.compile("&#0?39;"); cc[4] = '\'';
    q[5] = Pattern.compile("&#0?92;"); cc[5] = '\\';
    m_title = p.matcher(title).replaceAll("");
    m_snippet = p.matcher(snippet).replaceAll("");
}
```

ตัวอย่าง source code สำหรับการรายการของเอกสาร (Result Doc)

```
package preprocess;
public class ResultDoc
{
    public String Title;
    public String Snippet;
    public String URL;
    public ResultDoc(String title, String snippet, String url)
    {
        Title = title;
        Snippet = snippet;
        URL = url;
    }
}
```

ตัวอย่าง source code สำหรับการเพิ่มเอกสารลงไปในรายการของเอกสาร

```
ResultDoc Result = new ResultDoc(TitleNa, SnippetNa, URLNa);
ResultList.add(Result);
```

ตัวอย่าง source code สำหรับการเปลี่ยนรูปของคำให้กลับมาอยู่ในรูปของรากศัพท์

```
public String stem(String text, Boolean preserveSyntax) throws Throwable {
    if (stopPattern == null )
        createStopPattern();
    StringReader reader = new StringReader(text);
    TokenStream stream = m_analyzer.tokenStream("A", reader);
    PorterStemFilter filter = new PorterStemFilter(stream);
    StringBuffer buffer = new StringBuffer();
    int flag = 0;
    Token token;
    if(!preserveSyntax) {
        while((token = filter.next()) != null) {
            buffer.append(" ");
            buffer.append(token.termText());
        }
    } else {
        while((token = filter.next()) != null) {
            buffer.append(text.substring(flag, token.startOffset()));
            flag = token.endOffset();
            buffer.append(token.termText());
        }
    }

    if(flag < text.length()) buffer.append(text.substring(flag, text.length()-1));
    String res =stopPattern.matcher(" " + buffer ).replaceAll(" ").trim();
    while ( res.indexOf(" ")>-1 )
        res = res.replaceAll(" ","");
    return res;
}
```

ตัวอย่าง source code สำหรับการวิเคราะห์คำภาษาไทย

```
/** Builds an analyzer. */
public ThaiAnalyzer(String indexDir) throws IOException
{
    this(STOP_WORDS);
    ArrayList<String> stopThaiEng = new ArrayList<String>();
    File dir = new File(indexDir);
    File indexdir = dir.getParentFile();
}
/** Builds an analyzer with the given stop words. */
public ThaiAnalyzer(String[] stopWords) throws IOException
{
    stopSet = StopFilter.makeStopSet(stopWords);
    lexto= new LexTo();
}
/** Constructs a {@link ThaiTokenizer} filtered by a {@link
StandardFilter}, a {@link LowerCaseFilter} and a {@link StopFilter}. */
public TokenStream tokenStream(String fieldName, Reader reader) {
    TokenStream result = new ThaiTokenizer(reader, lexto);
    result = new ThaiFilter(result);
    result = new StandardFilter(result);
    result = new LowerCaseFilter(result);
    result = new StopFilter(result, stopSet);
    return result;
}
```

ตัวอย่าง source code สำหรับการจัดกลุ่มด้วยอัลกอริทึมซัพฟิกทรี

```
public class SuffixNode{
    ArrayList<String> Words=null;
    ArrayList<SuffixNode> children=null;
    SuffixNode parent=null;
    ArrayList<String> Documents=null;

    public SuffixNode(SuffixNode _parent, List<String> incomming_words){
        this();
        Words.addAll(incomming_words);
        parent=_parent;
    }
    public SuffixNode(){
        Words = new ArrayList<String>();
        children = new ArrayList<SuffixNode>();
        Documents = new ArrayList<String>();
    }
    public void addDocument(Document doc){
        addDocument(doc.getTermList(), doc.getDocumentID());
    }
    public void addDocument(List<String> suffixlist, String doc_name){
        for (int i = 0; i < suffixlist.size(); i++) {
            List<String> substr = new ArrayList<String>();
            substr.addAll(suffixlist.subList(i,suffixlist.size()));
            if (substr.size()> 0)
            {
                this.addSuffix(substr, doc_name);
            }
        }
    }
    public void addSuffix(List<String> _words, String doc_name){
        SuffixNode insert_node = this;
        insert_node = search(this, _words);
        insert(insert_node, _words, doc_name);
    }
    public SuffixNode search(SuffixNode start_node, List<String> text){
        //System.out.println("parent="+start_node.toString());
        for(int i=0; i<start_node.children.size();i++){
            SuffixNode child = start_node.children.get(i);
            //System.out.println(text.get(0));
            if(child.Words.get(0).equals(text.get(0))){
                //text.remove(0);
                return child;//search(child,text);
            }
        }
        return start_node;
    }
}
```

```

public void insert(SuffixNode current_node, List<String> text, String doc_name){
    int last_index_common = -1;
    int i;
    int min_common = Math.min(current_node.Words.size(), text.size());
    //> Find longest common prefix word
    for (i = 0; i < min_common; i++) {
        if(current_node.Words.get(i).equals(text.get(i))) {
            last_index_common++; //i++;
        }
        else break;
    }
    //last_index_common = i==min_common ? (i-1):i;
    // cut non-common word to new child node
    if((last_index_common)==current_node.Words.size()-1 &&
        current_node.Words.size()==text.size()){
        if(current_node.Documents.indexOf(doc_name)<0)
            {current_node.Documents.add(doc_name);}
        //System.out.println(text.get(0)+" "+doc_name);
        return; // text is equal to all words in this node
    }
    if(last_index_common<0){ // no common prefix matching
        //List<String> sub_text = text.subList(0, text.size());
        SuffixNode new_node = new SuffixNode(current_node, text);
        new_node.Documents.add(doc_name);
        current_node.children.add(new_node);
        return;
    }
    if(last_index_common<current_node.Words.size()-1){
        //create new node from tail of current_node
        //int start_index = last_index_common ;//<0 ? 0 : last_index_common;
        //System.out.println("start index =" +start_index);
        SuffixNode new_node = new SuffixNode(current_node,
            current_node.Words.subList(last_index_common+1, current_node.Words.size()));
        for (int j = 0; j < current_node.children.size(); j++) {
            current_node.children.get(j).parent=new_node;
            new_node.children.add(current_node.children.get(j));
        }
        if(current_node.Documents.indexOf(doc_name)<0)
            {current_node.Documents.add(doc_name);}
        //current_node.Documents.add(doc_name);
        new_node.Documents.addAll(current_node.Documents);
        current_node.Documents.clear();
        current_node.children.clear();
        current_node.children.add(new_node);
        // create new_words
        ArrayList<String> new_words = new ArrayList<String>();
        new_words.addAll(current_node.Words.subList(0, last_index_common+1));
        current_node.Words.clear();
        current_node.Words.addAll(new_words);
        new_words=null;
    }
    // create new node from tail of text
    if(last_index_common<text.size()-1){
        List<String> sub_text = text.subList(last_index_common+1, text.size());
        SuffixNode new_node = new SuffixNode(current_node, sub_text);
        new_node.Documents.add(doc_name);
        current_node.children.add(new_node);
    }
    else if(last_index_common==text.size()){
        System.out.println("<" +text.get(text.size()-1)+"> EQU ");
        if(current_node.Documents.indexOf(doc_name)<0)
            {current_node.Documents.add(doc_name);}
        //current_node.Documents.add(doc_name);
    }
}

public String toString(){
    String str="";
    for (int i = 0; i < Words.size(); i++) {
        str = str+" "+Words.get(i);
    }
    return str;
}

```

```

public void TraversAll(){
    SuffixNode _parent = this.parent;
    while(_parent!=null){
        //System.out.print("|");
        System.out.print(" ");
        _parent = _parent.parent;
    }
    System.out.println("--["+this.toString()+"] <"+this.listDocuments().size()+">");
    for (int i = 0; i<children.size();i++) {
        SuffixNode element = (SuffixNode) children.get(i);
        //System.out.println(">>");
        element.TraversAll();
    }
}

public ArrayList<String> listDocuments(){
    ArrayList<String> docs = new ArrayList<String>();
    //SuffixNode _parent = this.parent;
    for (int i = 0; i<children.size();i++) {
        SuffixNode element = (SuffixNode) children.get(i);
        //System.out.println(">>");
        //element.TraversAll();
        docs.addAll(element.listDocuments());
    }
    docs.addAll(Documents);
    return docs;
}

public SuffixNode getParent() {
    return parent;
}

public void setParent(SuffixNode parent) {
    this.parent = parent;
}

public ArrayList<SuffixNode> getChildren() {
    return children;
}

public void setChildren(ArrayList<SuffixNode> children) {
    this.children = children;
}
}

```

ตัวอย่าง source code สำหรับการรวมกลุ่มพื้นฐาน

```

public void MergingProcess(){
    for(int i=0;i<listBaseCluster.size()-1;i++){
        BaseCluster b1 = (BaseCluster) listBaseCluster.get(i);
        if(b1.isActive()){
            for(int j=i+1;j<listBaseCluster.size();j++){
                BaseCluster b2 = (BaseCluster) listBaseCluster.get(j);
                //System.out.println("compare b"+i+" with b"+j);
                if(b1.compareCluster(b2)>=0.5 && b2.compareCluster(b1)>=0.5){
                    this.combineCluster(b1, b2);
                    //System.out.println("....combining ..... "+b1.getLabelText());
                }
            } // end for j
        } // end if
    }
}

// Combine 2 clusters into a single BaseCluster
public void combineCluster(BaseCluster b1, BaseCluster b2){
    // TODO Combining the two BaseClusters
    // ([1] add label, [2] add document, [3] set deActive to merged cluster)
    // part [1] Merging the label
    String[] labels = b2.getLabel();
    for (int i = 0; i < labels.length; i++) {
        String text = labels[i];
        b1.addLabel(text);
    }
    // part [2]
    Iterator<String> iterator = b2.getDocument().iterator();
    while(iterator.hasNext()){
        String doc = iterator.next();
        b1.addDocument(doc);
    }
    // part [3]
    b2.setActive(false);
}
}

```

ตัวอย่าง source code สำหรับการจัดลำดับกลุ่มเอกสารโดยใช้สูตร $S_c = N_c * \sum (1/i^2)$

```
import java.util.ArrayList;
public class Scoring {
    private float cluster_score = 0;
    public Scoring(int number, ArrayList<String> doc){
        float rank_score = 0;
        for(int i=0;i <= number-1;i++){
            float rank = Integer.parseInt(doc.get(i));
            float doc_score = 1/(rank*rank);
            rank_score = rank_score+doc_score;
        }
        cluster_score = number * rank_score;
    }
    public float getCluster_score() {
        return cluster_score;
    }
    public void setCluster_score(float cluster_score) {
        this.cluster_score = cluster_score;
    }
}
```

ประวัติผู้เขียน

ชื่อ

นายชาลี จิตรีผ่อง

วัน เดือน ปี เกิด

10 กรกฎาคม พ.ศ. 2518

ที่อยู่

45/1 หมู่ 7 ตำบลสนามชัย

อำเภอบางไทร จังหวัดพระนครศรีอยุธยา .

E-mail Address

chalee_jittreephong@yahoo.com

ประวัติการศึกษา

ปริญญาตรี

บริหารธุรกิจบัณฑิต (ระบบสารสนเทศ)

คณะบริหารธุรกิจ สถาบันเทคโนโลยีราชมงคล ทุมธานี

ปีการศึกษา 2545

ประกาศนียบัตรวิชาชีพชั้นสูง

คอมพิวเตอร์ธุรกิจ

โรงเรียนไทยบริหารธุรกิจและพาณิชย์การ กรุงเทพมหานคร

ปีการศึกษา 2542