ระบบค้นคืนสารสนเทศ โดยใช้ ฟัซซี่ ลอจิก

INFORMATION RETRIEVAL SYSTEM USING FUZZY LOGIC

นาย พิศักดิ์ คุรุเสถียร

นาย พุฒตรี ธาราธิคุณเดช

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2544

ระบบค้นคืนสารสนเทศ โดยใช้ ฟัซซี่ ลอจิก

INFORMATION RETRIEVAL SYSTEM USING FUZZY LOGIC

โดย

นาย พิศักดิ์ คุรุเสถียร

นาย พุฒตรี ธาราธิคุณเดช

อาจารย์ที่ปรึกษา

รศ.ดร. เอื้อน ปิ่นเงิน

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2544

ปริญญานิพนธ์ปีการศึกษา 2544

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง ระบบค้นคืนสารสนเทศ โดยใช้ ฟัซซี่ ลอจิก

INFORMATION RETRIEVAL SYSTEM USING FUZZY LOGIC

ผู้จัดทำ

1. นาย พิศักดิ์ คุรุเสถียร รหัสประจำตัว 41014316

2. นาย พุฒตรี ธาราธิคุณเดช รหัสประจำตัว 41014317

_____ อาจารย์ที่ปรึกษา

(รศ.ดร. เอื้อน ปิ่นเงิน)

ระบบค้นคืนสารสนเทศ โดยใช้ ฟัซซี่ ลอจิก

นายพิศักดิ์ คุรุเสถียร 41014316
นายพุฒตรี ธาราธิคุณเดช 41014317
รศ.ดร. เอื้อน ปิ่นเงิน อาจารย์ที่ปรึกษา
ปีการศึกษา 2544

บทคัดย่อ

ระบบค้นคืนสารสนเทศสามารถแบ่งได้ ออกได้ 2 กระบวนการคือ การบวนการจัดเก็บ เพื่อใช้เป็นข้อมูลในการค้นคืน และ กระบวนการค้นคืนข้อมูลที่เกิดจากผู้ใช้ โดยระบบค้นคืนสารสนเทศแบบดั้งเดิมนั้น จะมีปัญหาทั้ง 2 ประการ คือ ในกระบวนการจัดเก็บจำเป็นต้องทำการสร้าง index term ด้วยการพิมพ์เข้าไปในแต่ละเอกสาร ซึ่งต้องเสียเวลามากสำหรับระบบที่มีจำนวนเอกสารมาก ดังนั้นจึงจำเป็นต้องทำ index term อัตโนมัติ ในการจัดเก็บเพื่อความสะดวก และ ปัญหาในกระบวนการค้นคืนข้อมูลคือ ผู้ใช้จำเป็นต้องรู้ keyword ที่ตรงกับข้อมูลที่ใช้ในการค้นคืน ซึ่งเป็นไปได้ยากที่ผู้ใช้จะรู้ ดังนั้น ในระบบค้นคืนสารสนเทศแบบใหม่ ผู้ใช้สามารถกำหนด keyword ที่ไม่จำเป็นต้องตรงกับข้อมูลที่ใช้ในการค้นคืน โดยที่ระบบจะสามารถค้นคืนข้อมูลได้โดยใช้ความสัมพันธ์ในการตัดสินใจ ทฤษฎี ฟัซซี่ ลอจิก ถูกนำมาใช้เพื่ออธิบายความสัมพันธ์ของข้อมูลที่คลุมเครือ การนำทฤษฎี ฟัซซี่ ลอจิก มาใช้จะช่วยให้ระบบค้นคืนสารสนเทศมีความยืดหยุ่น และ ใช้ประโยชน์ได้มากขึ้น

INFORMATION RETRIEVAL SYSTEM USING FUZZY LOGIC

Pisat Kurustian 41014316

Puttree Taratikundij 41014317

Advisor Assoc.Prof.Dr.Ouen Pinngem

ABSTRACT

Information retrieval system (IR System) mainly involves the process of storing and retrieving information. Traditional IR systems suffer from many problems in both processes. The storing process in those system requires the user to define the index terms for each document, which spends a lot of time if there are lots of documents in the system. Automatic indexing technique can facilitate the user by automatically provide index terms. A problem in the retrieval process is that users need to state exact keywords to retrieve information. This is difficult if users don't know such exact keywords. In modern IR system users can retrieve information using inexact queries. The system will retrieve the information that is most relevant. Fuzzy logic theory can be used in the system that deals with vague information. Applying fuzzy logic to the retrieved process enables the user to provide inexact or incomplete query for the retrieval, thus improves the flexibility and usability of the system.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้คงไม่อาจสำเร็จได้ด้วยดี หากไม่ได้รับความช่วยเหลือ และร่วมมือจากหลาย ๆ ฝ่ายด้วยกัน บุคคลแรกที่จะขาดเสียไม่ได้ในความสำเร็จดังกล่าวนี้ คือ อาจารย์ เอื้อน ปิ่นเงิน อาจารย์ที่ปรึกษาปริญญานิพนธ์ ที่ไม่เพียงแต่สละเวลาเท่านั้น แต่ยังสละแรงกายและแรงใจในการเอาใจใส่ดูแล แนะนำ และช่วยเหลือเสมอมาจนปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงด้วยดี อีกทั้งคณาจารย์ทุกท่านที่ถ่ายทอดวิชาความรู้มาให้ ซึ่งในความกรุณาของอาจารย์ที่มี ต้องขอขอบพระคุณเป็นอย่างสูง

ประการถัดมาที่มีส่วนสำคัญในความสำเร็จไม่ยิ่งหย่อนกันเลยนั่นคือ เพื่อน ๆ ในภาควิชาวิศวกรรมคอมพิวเตอร์ ที่มีส่วนสนับสนุน ให้กำลังใจ และ เป็นที่ปรึกษา อีกทั้งยังร่วมทุกข์ร่วมสุขกันมาเป็นเวลานานในช่วงที่ทำการค้นคว้าหาข้อมูล ความสำเร็จที่มีในวันนี้ มีเพื่อน ๆ ทุกคนร่วมเดินไปจนถึงเส้นชัยด้วยกัน

และบุคคลที่สำคัญที่สุดที่ทำให้มีวันนี้ได้ ก็คือ บิดา มารดา อันเป็นที่เคารพอย่างสูง ในพระคุณอันมิอาจเปรียบได้กับสิ่งอื่นใด ความรักความเอาใจใส่ และ โอกาสการศึกษาที่มีให้ อีกทั้งกำลังใจที่มีไม่เคยขาดหายไป พระคุณอันสูงสุดที่มิอาจตอบแทนได้หมด ข้าพเจ้าขอระลึกในพระคุณ และกราบขอบพระคุณเป็นอย่างสูงมา ณ ที่นี้

นาย พิศักดิ์ คุรุเสถียร

นาย พุฒตรี ธาราธิคุณเดช

สารบัญ

หน้าที่

หน้าที่

หน้าที่

สารบัญรูปภาพ

หน้าที่

สารบัญตาราง

หน้าที่

บทที่ 1
บทนำ

1.1 ความเป็นมาของโครงงาน

ระบบค้นคืนสารสนเทศโดยใช้ฟัซซี่ลอจิก เป็นระบบที่พัฒนามาจากระบบค้นคืนแบบเก่า (ระบบค้นคืนสารสนเทศแบบ บูลีน) ที่ความสามารถในการค้นคืนมีข้อจำกัด เพราะ ข้อมูลที่ใช้ในการค้นคืนจำเป็นต้องตรงกับข้อมูลที่จัดเก็บ ทำให้เป็นที่ยุ่งยากในการหาข้อมูล ดังนั้นจึงมีแนวความคิดที่จะพัฒนาระบบค้นคืนข้อมูลที่สามารถตอบสนองกับความต้องการของผู้ใช้ที่แต่ละบุคคนนั้น อาจใช้คำเพื่อเป็นสื่อในการค้นคืนข้อมูลที่แตกต่างกันออกไป และทฤษฎี ฟัซซี่ เป็น ทฤษฎีที่ว่าด้วยความไม่ชัดเจนประกอบกับ ทฤษฎี ฟัซซี่ นี้ยังไม่ค่อยมีผู้สนใจนำไปใช้กับระบบค้นคืนมากนัก จึงเป็นหัวข้อที่น่าสนใจในการศึกษา

1.2 วัตถุประสงค์ของโครงงาน

- เพื่อสามารถสร้างระบบค้นคืนสารสนเทศได้
- เพื่อศึกษา ทฤษฎี ที่สามารถนำไปประยุกต์ใช้กับระบบค้นคืนสารสนเทศได้
- เพื่อสามารถพัฒนาระบบค้นคืนสารสนเทศให้ตอบสนองความต้องการของผู้ใช้ได้

1.3 ขอบเขตของโครงงาน

- ออกแบบการจัดเก็บข้อมูลของเอกสาร
- ออกแบบกระบวนการทำ automatic indexing
- ออกแบบระบบค้นคืนสารสนเทศโดยใช้ ทฤษฎี ฟัซซี่ ลอจิกเป็น โครงสร้าง
- ออกแบบหน้าจอติดต่อกับผู้ใช้

1.4 วิธีการดำเนินงาน

- ศึกษาทฤษฎี ฟัซซี่ลอจิก และ นำเอา ทฤษฎีฟัซซี่ลอจิก มาประยุกต์ใช้กับการค้นคืนข้อมูล
- ศึกษาการเขียนโปรแกรมบนฝั่ง เซิร์ฟเวอร์ โดยภาษาที่ใช้ ได้แก่ ภาษา JSP และการเขียนโปรแกรมติดต่อ Data base โดย DBMS ที่ใช้คือ DB2
- ศึกษาการจัดการข้อมูลโดยมี DB2 เป็น DBMS
- ศึกษา Neural Network เพื่อใช้ทำ index อัตโนมัติ
- ศึกษาพฤติกรรมในการค้นหาข้อมูลของผู้ใช้ เพื่อนำมาปรับปรุงวิธีการค้นคืน

1.5 ผลที่คาดหวังว่าจะได้รับ

- ระบบค้นคืนสารสนเทศที่สามารถใช้คำค้นหรือคีย์เวิร์ดที่คลุมเครือได้

1.6 เนื้อหาโดยรวม(Outline)

ในบทที่ 2 จะอธิบายภาพรวมของการค้นคืนเอกสารว่าระบบที่สร้างขึ้นนั้นต้องแบ่งออกเป็นส่วนประกอบใดบ้างเพื่อให้ผู้อ่านได้เข้าใจ ความหมายของคำว่า data retrieval และ คำว่า information retrieval ว่ามีความเหมือนหรือแตกต่างกันอย่างไร ในบทนี้จะอธิบาย คำจำกัดความคำว่า Automatic indexing มีผลอย่างไรต่อการค้นหาเอกสาร และท้ายบทจะเป็น การออกแบบเบื้องต้นของระบบค้นคืนข้อมูลที่ยังไม่ได้นำเอา ทฤษฎี ฟัซซี่ ลอจิกมาใช้

ในบทที่ 3 เริ่มต้นจะอธิบายทฤษฎีของฟัซซี่เบื้องต้น เพื่อทำความเข้าใจต่อผู้อ่าน และให้นิยามคำจำกัดความของ ฟัซซี่ เซต ในส่วนที่ 2 ของบทนี้ จะพูดถึง การนำเอา ทฤษฎี ฟัซซี่ มาใช้ในเรื่องการค้นคืนข้อมูล ว่ามีรูปแบบการคำนวณเป็นอย่างไร

ในบทที่ 4 จะเกี่ยวกับถึงการทำ text operation ซึ่งประกอบไปด้วยขั้นตอนต่างๆ ได้แก่ การตัดคำ การกำจัด Stopwords การทำ Stremming (Suffix Removal) การเลือก indexเทอม และการทำ Thesaurus

ในบทที่ 5 จะกล่าวถึงการเลือก index term แบบอัตโนมัติ (Automatic Indexing)

ในบทที่ 6 จะเป็นเรื่องเกี่ยวกับรูปแบบในการ Query และ Ranking แบบต่างๆ

ในบทที่ 7 การออกแบบและ โครงงานที่ทำ

ในบทที่ 8 เป็นการวัดประสิทธิภาพ โดยใช้เครื่องมือวัดคือ Precision และ Call back

ในบทที่ 9 เป็นบทสรุปของเนื้อหาปริญญานิพนธ์ ทั้งหมด และการพัฒนา

บทที่ 2
การค้นคืนข้อมูล

2.1 ภาพรวมของระบบค้นคืนข้อมูล

ระบบค้นคืนข้อมูล เกี่ยวข้องกับการ นำเสนอข้อมูล การจัดเก็บ การจัดการ และ การเข้าถึงข้อมูล การนำเสนอและการจัดการกับข้อมูล จำเป็นต้องให้ผู้ใช้เข้าถึงข้อมูลที่สนใจได้ง่าย แต่สิ่งนี้ความเป็นจริงแล้วไม่ใช้สิ่งง่าย เพราะ พฤติกรรมความต้องการของผู้ใช้นั้นย่อมมีความแตกต่างกันไป เป็นเพราะข้อมูลที่ผู้ใช้ในการค้นหาข้อมูลนั้นส่วนใหญ่จะเป็นภาษาไม่มีระเบียบทางด้านโครงสร้างของภาษา ดังนั้นต้องมีระบบค้นคืนที่สามารถแปลงความต้องการของผู้ใช้ที่อยู่ในรูปภาษาที่ไม่มีระเบียบทางด้านโครงสร้างให้เป็นภาษาที่มีโครงสร้างเพื่อใช้ในการค้นคืนข้อมูล

2.2 ความแตกต่างระหว่าง Information Retrieval กับ Data Retrieval

ในการค้นคืนข้อมูลแบบ Data Retrieval นั้นผู้ใช้จำเป็นต้องระบุ คำสำคัญ(key word) เพื่อใช้ในการค้นหา แต่ การค้นคืนข้อมูลแบบ information retrieval นั้นสามารถใช้ข้อมูลที่ไม่เป็นโครงสร้างและไม่จำเป็นต้องเป็นคำสำคัญ ดังนั้นในการค้นคืนข้อมูลแบบ Data Retrieval นั้นผู้ใช้จำเป็นต้องเข้าใจวิธีการสร้างคำสำคัญเพราะไม่เช่นนั้นแล้วการค้นคืนข้อมูลจะไม่สามารถกระทำได้เพราะ ข้อมูลที่ใช้ในการค้นหานั้นต้องมีความสัมพันธ์แบบตรงกันทุกประการ ยกตัวอย่างเช่น ในระบบการค้นคืนข้อมูลจำเพาะ ถ้าผู้ใช้ต้องการหาข้อมูลเกี่ยวกับ Internet ผู้ใช้อาจใช้คำว่า 'Internet' เป็นข้อมูลในการค้นหา แต่ผู้ใช้ไม่อาจรู้ได้เลยว่าคำว่า 'Internet' เป็นคำสำคัญหรือไม่ ถ้าคำว่า 'Internet' ไม่ใช่คำสำคัญ ก็ไม่สามารถใช้คำว่า 'Internet' ในการค้นหาได้

ถึงแม้ว่าการค้นคืนข้อมูลแบบ Data Retrieval จะใช้ได้ดีในระบบฐานข้อมูลแต่ก็มีจุดเสียที่ไม่สามารถค้นหาโดยใช้หัวเรื่อง(subject) หรือหัวข้อ(title) ได้ ดังนั้นในการแก้ไขปัญหานี้ จึงเกิดระบบค้นคืนข้อมูลที่สามารถสรุปความหมายของเอกสารนั้น ๆ เพื่อระบุหัวเรื่อง หรือหัวข้อของเอกสารนั้น ๆ ได้ โดยจะนำเอาข้อมูลที่ได้จากการตีความหมายมาเป็นคิดความสัมพันธ์กับข้อมูลการค้นหาของผู้ใช้

โดยสรุปแล้วระบบการค้นคืนข้อมูลแบบ Information Retrieval นั้นจะเน้นความสัมพันธ์กับผู้ใช้เป็นหลัก โดยที่ระบบต้องสามารถค้นคืนเอกสารที่มีความสัมพันธ์กับข้อมูลของผู้ใช้ได้ทั้งหมด และต้องไม่นำเอกสารที่ไม่มีความสัมพันธ์กับข้อมูลของผู้ใช้มาแสดง หรือ มาแสดงน้อยที่สุด

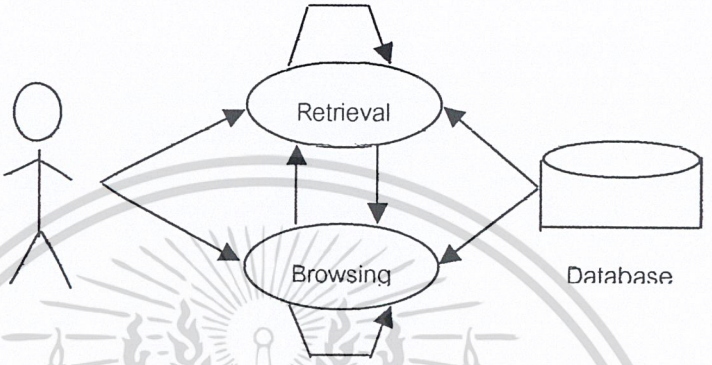
2.3 แนวคิดพื้นฐานของระบบค้นคืนข้อมูล

สิ่งที่มีผลต่อการค้นหาข้อมูลที่สัมพันธ์กันจะขึ้นอยู่กับปัจจัย 2 สิ่งด้วยกันนั่นคือ งานที่เกิดจากผู้ใช้ และ มุมมองในการค้นคืนเอกสาร(logical view of document)

2.3.1 งานที่เกิดจากผู้ใช้

จะแบ่งงานของผู้ใช้ออกเป็นได้ 2 แบบใหญ่ ๆ ด้วยกันคือ

1. การค้นคืนข้อมูล
2. การ browse ข้อมูล



รูปที่ 2-1 งานของผู้ใช้ที่มีต่อระบบ

จากรูป 2-1 จะเห็นได้ว่างานของผู้ใช้นั้นแบ่งเป็น 2 ประเภทด้วยกัน คือการค้นคืนข้อมูล และการ browse ข้อมูล ซึ่ง

การค้นคืนข้อมูล(retrieval) งานของผู้ใช้ที่ต้องแปลความต้องการของผู้ใช้ให้เป็นภาษาที่สามารถใช้ในการค้นคืนข้อมูลได้ทั้งในรูปแบบการค้นคืนข้อมูลแบบ Information Retrieval และการค้นคืนข้อมูลแบบ Data Retrieval เราจะเรียกสิ่งนี้ว่าเป็นงานของการค้นคืน

การ browse ข้อมูล คือการค้นคืนข้อมูลที่มีโต้ตอบระหว่างผู้ใช้กับระบบ ซึ่งปัจจุบันเป็นส่วนจำเป็นมากขึ้นซึ่งอาจเนื่องมาจากการค้นคืนข้อมูลแบบปัจจุบัน จะเป็นการค้นคืนโดยให้ข้อมูลแบบกว้าง ๆ ดังการค้นหาข้อมูล url ของ web site ดังนั้นการโต้ตอบกับผู้ใช้จะสามารถอำนวยความสะดวกกับผู้ใช้ได้มากขึ้น

2.3.2 มุมมองในการค้นคืนเอกสาร

มุมมองในการค้นคืนเอกสารนั้นมีด้วยกันหลายวิธี อย่างแรกเป็นวิธีการสร้าง index term โดยการนำคำที่อยู่ในเอกสารที่มีค่าความถี่สูง ออกมาเป็น index term หรือ อาจเป็นการระบุโดยคน ซึ่งผลที่ได้ยังเป็นผลที่มีประสิทธิภาพในการค้นคืนที่ต่ำอยู่ ต่อมา ประสิทธิภาพของเครื่องคอมพิวเตอร์มีมากขึ้น ดังนั้นมุมมองในการค้นคืนเอกสาร จะนำเอาข้อความทั้งหมดที่มีอยู่ในเอกสารนั้นมาเป็นมุมมองในการค้นคืน แต่ข้อเสียคือ เนื้อที่ในการจัดเก็บ เวลาในการค้นหา และที่สำคัญคือ คำบางคำจะมีทุก ๆ เอกสารเช่น article 'the'

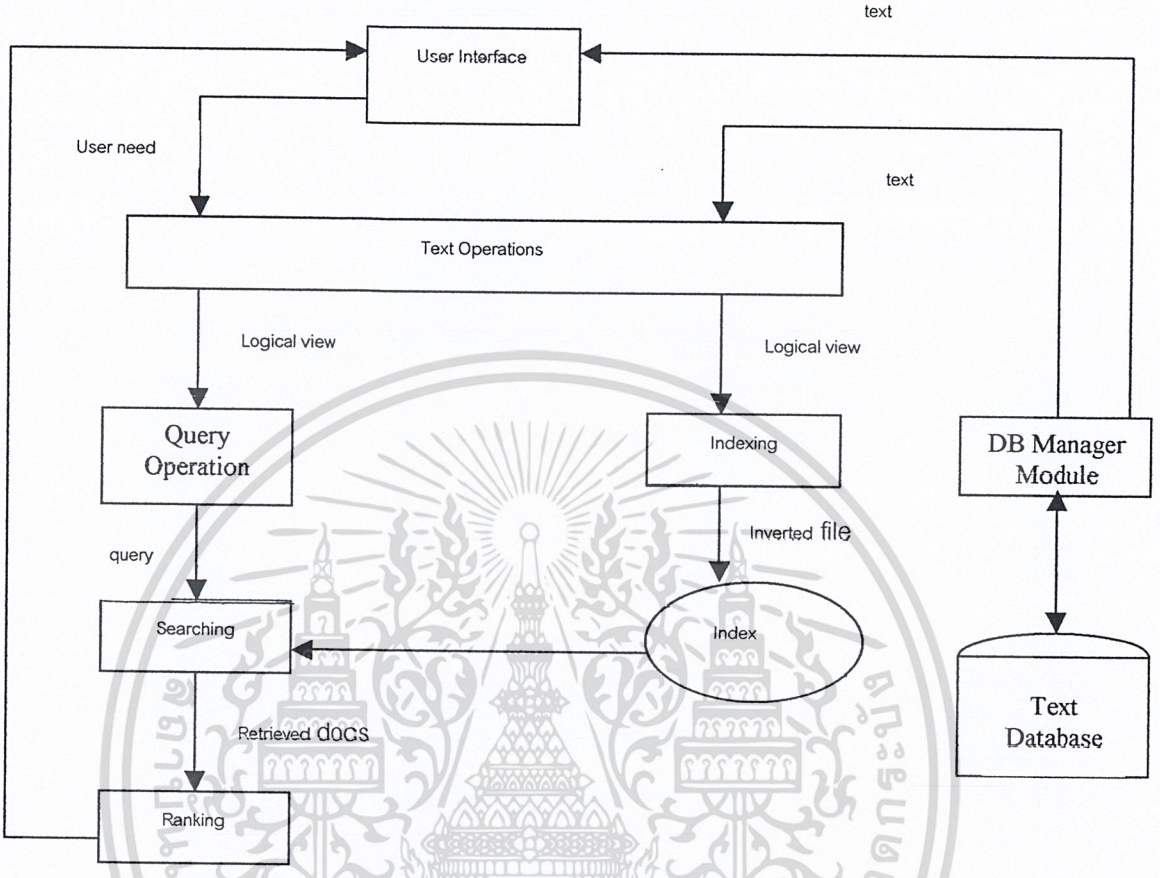
ดังนั้นในมุมมองสมัยใหม่ จะพยายามลดจำนวน keyword โดยใช้วิธีการกำจัดคำ(stop word) ในที่นี้หมายถึงการกำจัด คำจำพวก article หรือ คำเชื่อมประโยค with และอื่น ๆ วิธี stemming คือการลดขนาดของคำเช่น normally ไปเป็น normal เป็นต้น วิธีการการทำหนดกลุ่มของคำนาม โดยการกำจัดคำขยายกริยา กำจัดคำขยายคำนาม เป็นต้น นอกจากนี้แล้วจำเป็นต้องแบ่งแยกโครงสร้างของเอกสาร ได้แก่ บท หัวข้อ หัวข้อย่อย ในการทำกระบวนการเช่นนี้ไม่สามารถทำได้โดยคนทั้งหมด จึงได้เกิดรูปแบบที่เรียกว่า การทำ Automatic indexing

Automatic Indexing

การจัดทำ automatic indexing คือการโปรแกรมคอมพิวเตอร์โดยให้โปรแกรมนั้นสามารถที่จะรับอินพุทของระบบเป็นเท็กซ์ไฟล์ (text file) และสามารถที่จะให้ out put ของระบบออกมาเป็น index term ได้ ซึ่ง index term นั้นจะเป็นตัวแทนของเอกสารที่จะบอกได้ว่าเอกสารนั้นกล่าวถึงหรือเกี่ยวข้องสัมพันธ์กับเรื่องใด

โดยส่วนมากแล้วการจัดทำอินเด็กซ์อัตโนมัตินั้นจะจัดทำเพื่อที่จะนำไปใช้เกี่ยวกับการจัดเก็บข้อมูลและค้นคืนข้อมูล (information retrieval) ให้ได้เร็วที่สุด และเป็นการจัดเก็บข้อมูลให้เป็นหมวดหมู่ที่มีประสิทธิภาพ เช่นในระบบของห้องสมุดสมัยใหม่ ก็จะทำการจัดเก็บข้อมูลพร้อมทั้ง index term ของหนังสือหรือเอกสารนั้น ๆ เมื่อมีผู้ใช้ต้องการเข้ามาค้นหาข้อมูลต่าง ๆ ก็ทำการใส่ข้อมูลในการค้นหา และระบบจะทำการเปรียบเทียบ ข้อมูลที่ผู้ใช้ใส่เข้ามากับตัว index term ที่ได้มีการจัดเก็บไว้แล้ว เพื่อที่จะดึงข้อมูลที่ตรงกับสิ่งที่ผู้ใช้ต้องการได้

2.4 กระบวนการค้นคืนข้อมูลเบื้องต้น



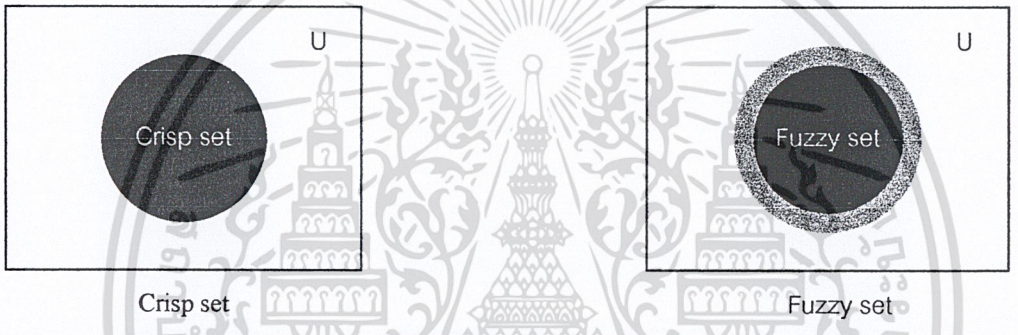
รูปที่ 2-2 กระบวนการค้นคืนเบื้องต้น

จากรูป 2-2 ได้อธิบายระบบค้นคืนสารสนเทศเบื้องต้นที่เป็นเพียงระบบที่ยังไม่ได้มีโครงสร้างอื่นมาขยายประสิทธิภาพในการค้นคืนข้อมูลโดยรูปข้างต้นนี้จะเป็นพื้นฐานในการนำเอาไปพัฒนาระบบอื่นๆ ที่มีประสิทธิภาพได้ โดยวิทยานิพนธ์ฉบับนี้จะนำเอาโครงสร้างนำไปพัฒนาระบบค้นคืนสารสนเทศที่ใช้ทฤษฎี ฟัซซี่ มาขยายประสิทธิภาพในการค้นคืนซึ่งจะได้อธิบายไว้ในบทที่ 7

บทที่ 3
ทฤษฎี ฟัซซี่เซตเบื้องต้น

3.1 ทฤษฎี ฟัซซี่เซต

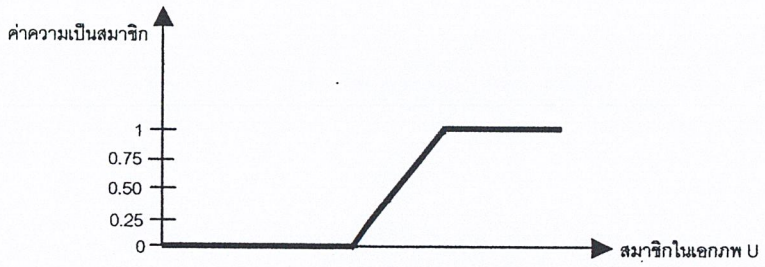
ทฤษฎีฟัซซี่เซต ถูกนำเสนอในรูปแบบของค่าขอบเขตที่คลุมเครือ แนวความคิดหลักคือการกำหนดฟังก์ชั่นความเป็นสมาชิกให้แก่สมาชิกภายในเซต ฟังก์ชั่นนี้จะมีค่าอยู่ระหว่าง 0 ถึง 1 [0,1] โดยที่ค่า 0 หมายถึง ไม่มีความเป็นสมาชิกเลย ค่า 1 หมายถึงความเป็นสมาชิกโดยสมบูรณ์ และค่าที่อยู่ระหว่าง 0 ถึง 1 หมายถึง ความเป็นสมาชิกแค่บางส่วน ดังนั้นความความเป็นสมาชิกของ ฟัซซี่เซตจะมีความต่อเนื่อง



รูปที่ 3-1 แสดงความแตกต่างระหว่าง Crisp set และ Fuzzy set

จากรูปที่ 3.1 จะเห็นว่า Crisp set จะประกอบไปด้วยส่วนที่เป็นสีขาวและสีดำเท่านั้น หมายถึง ค่าความเป็นสมาชิกจะมีค่าเป็น 0 หรือ 1 เท่านั้น {0,1} ส่วน Fuzzy set จะประกอบไปด้วย 3 ด้วยด้วยกันคือส่วนสีขาว สีดำ และ สีเทา ซึ่งหมายถึง ค่าความเป็นสมาชิกใน Fuzzy set นี้ จะประกอบไปด้วย ค่าความเป็นสมาชิก ที่อยู่ระหว่าง 0 ถึง 1 [0,1] ซึ่งสามารถอธิบายได้ดังนี้

นิยาม ฟัซซี่สับเซต A ของเอกภพ U ถูกกำหนดโดยฟังก์ชั่นความเป็นสมาชิก $\mu A : U \longrightarrow [0,1]$ ซึ่งกำหนดสมาชิกแต่ละตัว u ของเอกภพ ให้มีค่า $\mu A(u)$ มีค่าอยู่ในช่วง [0,1] ซึ่งสามารถแสดงได้ในรูป 3-2 ซึ่งจะยกตัวอย่างในการอธิบายได้ เช่นกำหนดให้ระยะทาง 20 ก.ม. ให้ถือว่าเป็นระยะทางที่ถึงว่าไกลดังนั้นระยะทางที่มากกว่า 20 ก.ม.จะมีค่าความเป็นสมาชิกกับคำว่าไกลเป็น 1 และ ค่าที่น้อยกว่า 13 ก.ม. ให้ถือว่ายังเป็นระยะทางที่ใกล้อยู่ ดังนั้นค่าที่อยู่ระหว่าง 13.001 ก.ม. ถึง 14.9999 จะมีค่าความเป็นสมาชิกกับคำว่าไกลเป็นเท่าไร ทฤษฎี ฟัซซี่ จะมาใช้ในการอธิบายข้อมูลประเภทนี้



รูปที่ 3-2 กราฟแสดงความเป็นสมาชิก

เช่นเดียวกับ เซ็ต ทั่ว ๆ ไปที่จำเป็นต้องมีตัวดำเนินการในการทำงานซึ่งประกอบไปด้วย คอมพรีเมนต์ ,ยูเนี่ยน และ อินเตอร์เซ็ก ฟัซซี่ เซ็ต ก็เช่นเดียวกันที่จำเป็นต้องมีตัวดำเนินการในการประมวลข้อมูลแต่ มีข้อแตกต่างกันในบางส่วนซึ่งสามารถอธิบายได้ดังนี้

ตัวดำเนินการหลัก ๆ 3 ประเภทของ ฟัซซี่เซต

1. คอมพรีเมนต์ของฟัซซี่เซต
2. ยูเนี่ยนของฟัซซี่เซต
3. อินเตอร์เซ็กฟัซซี่เซต

นิยาม ให้ U คือเอกภพ A และ B เป็นฟัซซี่สับเซต ของ เอกภพ U

$$\mu_{\bar{A}}(u) = 1 - \mu_A(u)$$
$$\mu_{A \cup B}(u) = \max(\mu_A(u), \mu_B(u))$$
$$\mu_{A \cap B}(u) = \min(\mu_A(u), \mu_B(u))$$

ซึ่งนิยามนี้สามารถเปลี่ยนแปลงได้ ซึ่งเกิดจากขั้นตอนการวัดประสิทธิภาพในการประมวลผล โดยนัก ฟัซซี่แต่ละคนอาจให้นิยามของ ตัวดำเนินการที่แตกต่างกันไป ขึ้นอยู่กับว่านำเอาหลักการของ ฟัซซี่ไปใช้อธิบายเรื่องใด โดยนิยามที่ยกเป็นตัวอย่างนี้เป็น นิยามของการทำ ยูเนี่ยนของฟัซซี่เซต และ อินเตอร์เซ็กของฟัซซี่ที่อยู่ในรูปแบบที่ง่ายที่สุด โดยนิยามนี้อาจใช้ได้ไม่ดีกับงานบางอย่าง ยกตัวอย่างเช่น นำเอาไปใช้งานในการทำ ระบบค้นคืนข้อมูล ซึ่งมีวิธีแก้ไขนิยามนี้ซึ่งจะอธิบายในหัวข้อ 5.1 ทฤษฎีฟัซซี่ที่นำมาใช้ในการค้นคืน

3.2 ทฤษฎีฟัซซี่ที่นำมาใช้ในการค้นคืน

การนำเอาทฤษฎีฟัซซี่มาใช้ในการค้นคืนข้อมูล เราจะนำเอาทฤษฎีฟัซซี่มาจับความสัมพันธ์ของอินเด็กซ์เทอม และความสัมพันธ์ของเอกสารที่มีอินเด็กซ์เทอมร่วมกันมาอยู่ในรูปของ เมตริก โดยที่ความสัมพันธ์นี้จะเขียนในรูปแบบแถวและคอลัมน์ในลักษณะของบัญชีคำ โดยความสัมพันธ์ดังกล่าวคิดได้จากสูตรดังนี้

$$c_{i,j} = \frac{n_{i,j}}{n_i + n_j - n_{i,j}}$$

โดยจากสูตรสามารถอธิบายได้ดังนี้

$c_{i,j}$ คือ ความสัมพันธ์คู่อินเด็กซ์เทอม iกับ j

$n_{i,j}$ คือ จำนวนเอกสารที่มีทั้ง อินเด็กซ์เทอม iและj เป็นอินเด็กซ์เทอม

n_i คือ จำนวนเอกสารที่มี อินเด็กซ์เทอม iเป็นอินเด็กซ์เทอม

n_j คือ จำนวนเอกสารที่มี อินเด็กซ์เทอม j เป็นอินเด็กซ์เทอม

โดยเราสามารถยกตัวอย่างความสัมพันธ์ของอินเด็กซ์เทอมในรูปเมตริกได้ดังนี้

	Index term1	Index term2	Index term3	Index term4
Index term1	1	0.47	0.34	0
Index term2	0.47	1	0.03	0
Index term3	0.34	0.03	1	0
Index term4	0	0	0	1

ตาราง 3-1 ตารางความสัมพันธ์ของ index term

จากตาราง 3-1 จะอธิบายได้ดังนี้

- $c_{i,i}$ จะมีค่าเท่ากับ 1 เสมอเพราะ $n_{i,i} = n_i$
- $c_{1,2}$ มีค่า 0.47 แสดงให้เห็นถึงความสัมพันธ์ของ index term1 ที่มีต่อ index term2 ซึ่งความสัมพันธ์นี้แสดงให้เห็นว่า มีเอกสารหลายเอกสารที่มีทั้ง index term1 และ index term2 เป็น index term ในเอกสารฉบับนั้นเมื่อเทียบกับ เอกสารที่มี index term1 หรือ index term2 เป็น index term และ ค่า $c_{1,2}$ จะเท่ากับค่า $c_{2,1}$
- $c_{1,4}$ มีค่า 0 หมายถึง ไม่มีเอกสารใดเลยที่มีทั้ง index term1 และ index term4 เป็น index term ที่อยู่ในเอกสารเดียวกัน ดังนั้นเราจะกล่าวได้ว่าความสัมพันธ์ของ index term1 และ index term4 ไม่มีความสัมพันธ์กันเลย

ตามหลักการความสัมพันธ์นี้ ถ้า query ที่ผู้ใช้ป้อนเข้ามาตรงกับ index term ที่มีเอกสารเป็นสมาชิก $k_i \in \{d_1, d_6, d_{20}, ..., d_n\}$ ค่าความเป็นสมาชิกของเอกสารแต่ละฉบับ ที่มี k_i เป็น index term จะมีค่า

เท่ากับ 1 และถ้าเรานำความสัมพันธ์ของ index term เช่นจากตาราง จะเป็นว่า index term1 มีความสัมพันธ์กับ index term 2 เท่ากับ 0.47 ดังนั้นเอกสารที่เป็นสมาชิกของ $k_2 \in \{d_6, d_8, d_{20}, ..., d_n\}$ แต่ค่าความเป็นสมาชิกของเอกสารเซตนี้เมื่อใช้ index term1 เป็นหลักค่าของความเป็นสมาชิกจะไม่เท่ากับ 1 แต่จะเท่ากับ 0.47 ตามสูตรต่อไปนี้

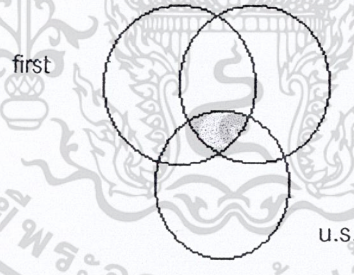
$$\mu_{i,j} = 1 - \prod_{k_l \in d_j} (1 - c_{i,l})$$

จากสูตรนี้เราจะได้ว่า

ค่าความเป็นสมาชิกของเอกสาร d_j ที่มีต่อ index term k_i เมื่อค่า index term k_l มีความสัมพันธ์กับ index term k_i มาก ค่า$c_{i,j}$ ~1 จะทำให้ค่า $\mu_{i,j}$ เข้าใกล้ค่า 1 ด้วย หมายถึงค่า index term k_i เป็นค่าฟัซซี่ index ที่ดีต่อเอกสาร d_j แต่ในทางตรงกันข้ามถ้า เมื่อค่า index term k_l มีความสัมพันธ์กับ index term k_i น้อย ค่า$c_{i,j}$ ~0 จะทำให้ค่า $\mu_{i,j}$ เข้าใกล้ค่า 0 ด้วย หมายถึงค่า index term k_i เป็นค่าฟัซซี่ index ที่ไม่ดีต่อเอกสาร d_j

<u>การทำ อินเตอร์เซ็ก</u>

แต่ในการใช้งานจริงแล้วนั้น query ที่ผู้ใส่เข้ามาส่วนมากจะเป็นกลุ่มของคำดังนั้นจึงมีเทอมในการค้นหาหลายเทอมเช่นผู้ใช้ใส่ query เข้ามาอย่างเช่น 'who is the first president of U.S.A?' ซึ่งกลุ่มก้อนของ query ชุดนี้เมื่อจัดให้อยู่ในเทอมจะได้ออกมา 3 เทอมคือคำว่า 'first' , 'president' , 'u.s.a'



รูปที่ 3-3 รูปความสัมพันธ์ของ index term กับการค้นหาของผู้ใช้แบบ อินเตอร์เซ็ก

ดังที่แสดงในรูป 3-3 จะเห็นว่าในการค้นหาโดยใช้ทั้ง 3 เทอมนั้นเราต้องนำค่าความเป็นในแต่ละเทอมมาandกันดังนั้นเราจะได้ว่า

$$\mu_{first \cap president \cap u.s.a}(x) = \min(\mu_{first}(x), \mu_{president}(x), \mu_{u.s.a}(x)) \qquad \forall x \in U$$

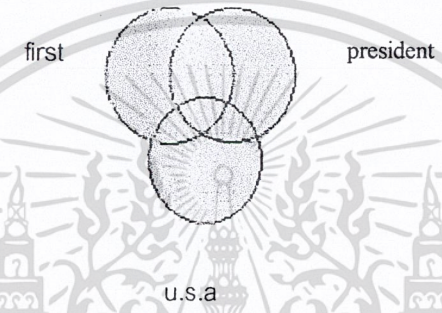
ซึ่งสูตรนี้ได้จากนิยามของ fuzzy แต่มีข้อเสียคือจะไม่มีความต่อเนื่อง เพราะความเป็นสมาชิกจะขึ้นอยู่กับเทอมที่มีค่าความเป็นสมาชิกน้อยที่สุดเพียงเทอมเดียวเท่านั้น ดังนั้นจึงต้องคิดใหม่ให้แต่แต่ละเทอมมีผลต่อความเป็นสมาชิกจึงได้สูตรใหม่ดังนี้

$$\mu_{first \cap president \cap u.s.a}(x) = \mu_{first}(x).\mu_{president}(x).\mu_{u.s.a}(x) \qquad \forall x \in U$$

ซึ่งสูตรนี้จะทำให้ผลลัพธ์มีความต่อเนื่องเพิ่มขึ้น

<u>การทำ ยูเนี่ยน</u>

เช่นเดียวกันกับ ถ้าหากผู้ใช้ต้องการค้นหาเอกสารที่ประกอบด้วย “first” or “president” or “u.s.a” จะได้เทอมออกมา 3 เทอมดังนี้



รูปที่ 3-4 รูปความสัมพันธ์ของ index term กับการค้นหาของผู้ใช้แบบ หรือ

ดังที่แสดงในรูป 3-4 จะเห็นว่าในการค้นหาโดยใช้ทั้ง 3 เทอมนั้นเราต้องนำค่าความเป็นในแต่ละเทอมมา or กันดังนั้นเราจะได้ว่า

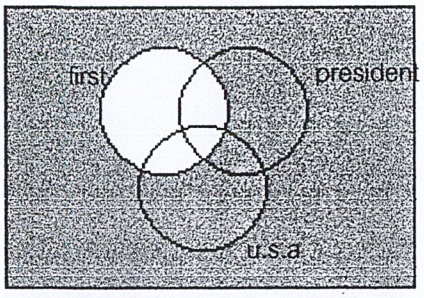
$$\mu_{first \cup president \cup u.s.a}(x) = \max(\mu_{first}(x), \mu_{president}(x), \mu_{u.s.a}(x)) \qquad \forall x \in U$$

ซึ่งสูตรนี้ได้จากนิยามของ fuzzy แต่มีข้อเสียคือจะไม่มีความต่อเนื่อง เพราะความเป็นสมาชิกจะขึ้นอยู่กับเทอมที่มีค่าความเป็นสมาชิกมากที่สุดเพียงเทอมเดียวเท่านั้น ดังนั้นจึงต้องคิดใหม่ให้แต่แต่ละเทอมมีผลต่อความเป็นสมาชิกจึงได้สูตรใหม่ดังนี้

$$\mu_{first \cup president \cup u.s.a}(x) = (1-(1-\mu_{first}(x)).(1-\mu_{president}(x)).(1-\mu_{u.s.a}(x))) \qquad \forall x \in U$$

<u>รูปแบบการทำคอมพรีเมนต์</u>

หากผู้ใช้มีความต้องค้นหาเอกสารที่ไม่มีทำว่า “first” เป็น index term จะได้ดังรูป 3-5



รูปที่ 3-5 รูปความสัมพันธ์ของ index term กับการค้นหาผู้ใช้แบบ not

สูตรในการคำนวณแบบ คอมพรีเมนต์คือ

$$\bar{\mu}_{first}(x) = (1 - \mu_{first}(x)) \qquad \forall x \in U$$

โดยจะนำสูตรต่าง ๆ ที่ได้อธิบายไว้ในบทนี้นำไปใช้ในการสร้างระบบค้นคืนสารสนเทศแบบฟัซซี่ซึ่งได้อธิบายไว้ในบทที่ 7 ซึ่งจะอธิบายรายละเอียดของระบบ ในส่วนของการสร้างการค้นคืน

บทที่ 4
การจัดการข้อความ

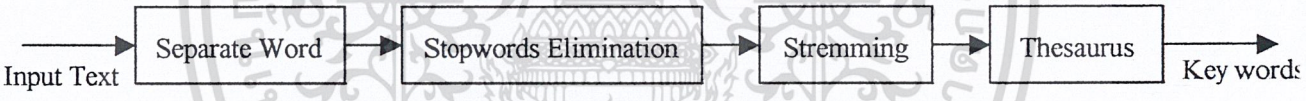
การจัดการข้อความมีความสำคัญอย่างมากในการสร้างระบบค้นหาข้อมูล เนื่องจากเป็นการกำจัดข้อมูลที่ไม่มีความสำคัญ ลดขนาดข้อมูลที่ใช้ แบ่งข้อมูลออกเป็นส่วนย่อย และวิเคราะห์ข้อมูลเหล่านั้น โดยจะมีการใช้การจัดการข้อความใน 2 ส่วนด้วยกัน คือ ส่วนของการทำindex อัตโนมัติ (Automatic Indexing) และส่วนการจัดการกับข้อความที่ผู้ใช้ป้อนเข้ามาเพื่อสืบค้น (Query)

4.1 ขั้นตอนในการจัดการข้อความ

ในส่วนของการจัดการกับข้อความที่ผู้ใช้ป้อนเข้ามาเพื่อสืบค้น มีขั้นตอนตามรูปที่ 4-1 ดังนี้

1. แบ่งคำ (Separate Word)
2. Stopwords Elimination
3. Stremming
4. Thesaurus

จากนั้นจึงนำคำที่ได้ไปใช้ในการค้นหา(Search) เพื่อเปรียบเทียบกับindexที่เก็บไว้

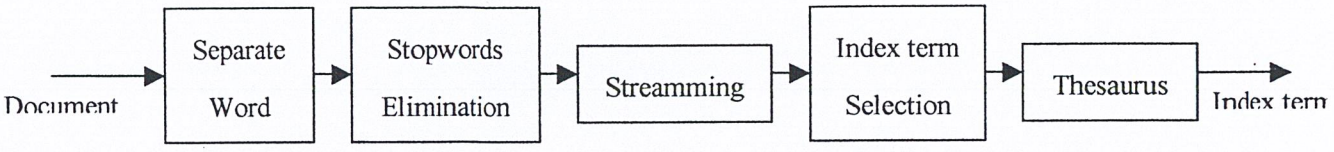


รูปที่ 4-1 แสดงขั้นตอนการจัดการข้อความที่ผู้ใช้ป้อนเข้ามาเพื่อสืบค้น

ในส่วนของการทำ index อัตโนมัติ มีขั้นตอน ดังนี้

1. แบ่งคำ (Separate Word)
2. Stopwords Elimination
3. Stremming
4. Index Term Selection
5. Thesaurus

จากนั้นจึงนำคำที่ได้ไปเก็บไว้ใช้เป็น Index Terms ของเอกสาร เพื่อใช้ในการสืบค้นข้อมูลตามรูป 4-2 ได้ดังนี้



รูปที่ 4-2 แสดงข้อความการจัดการข้อความภายในเอกสารเพื่อสร้าง index

4.1.1 การแบ่งคำ

การแบ่งคำเป็นกระบวนการในการเปลี่ยนกลุ่มของตัวอักษรให้กลายเป็นกลุ่มของคำ หรือก็คือการแบ่งข้อความที่ต่อเนื่องกันออกเป็นคำๆนั่นเอง

ในภาษาอังกฤษ ระหว่างคำจะถูกแบ่งด้วยช่องว่าง (Space) หรือเครื่องหมายต่างๆ เช่น . , - ‘’ : เป็นต้น โดยเครื่องหมายเหล่านี้จะใช้ในการแบ่งคำออกจากกัน ซึ่งในบางกรณีนั้น เครื่องหมายเหล่านี้ก็มีความสำคัญต่อความหมายของคำ เมื่อตัดทิ้งไปจะไม่สามารถเข้าใจความหมายได้ เช่น Unix-like OS แต่ในบางกรณีก็ไม่มีผลมากนัก เช่น State-of-the-art ซึ่งเมื่อแยกออกเป็นคำๆแล้ว ก็ยังคงความหมายเดิมอยู่นอกจากนี้คำบางคำที่เมื่ออยู่ด้วยกันจะมีความหมายพิเศษเพิ่มขึ้น เช่น Object Orient ก็จะทำให้เกิดความยุ่งยากในการแบ่งออกเป็นคำๆ

นอกจากนี้ตัวเลขก็ยังเป็นปัญหาที่สำคัญ เพราะตัวเลขอาจมีความสำคัญต่อความหมายของเอกสารหรือไม่มีก็ได้ นอกจากนี้ยังไม่มีความหมายในตัวเองหากขาดกลุ่มคำที่เกี่ยวข้อง ไม่เหมาะกับการเลือกเป็น Index term เพราะการค้นหาให้ได้ข้อมูลตรงกับความต้องการเป็นไปได้ยาก

เพื่อแก้ปัญหาต่างๆเหล่านี้ จึงมีการกำหนดรูปแบบในการแบ่งคำ เพื่อให้ได้ข้อมูลที่ตรงกับความต้องการมากที่สุด ดังนี้

1. แบ่งคำ โดยใช้เครื่องหมายต่างๆเป็นตัวแบ่ง ยกเว้น – และ . ซึ่งต้องไปพิจารณาเงื่อนไขข้อ 2
2. กรณี – จะคิดเป็นคำเดียวกันต่อเมื่อ
   - มีบางกลุ่มในกลุ่มที่เชื่อมกันด้วย – เป็น ตัวอักษรพิมพ์ใหญ่ทั้งหมด
     เช่น N-Layer, N-P-N
   - มีบางกลุ่มในกลุ่มที่เชื่อมกันด้วย – เป็นตัวเลขทั้งหมด
     เช่น B-79, M-16

   กรณี . จะคิดเป็นคำเดียวกันก็ต่อเมื่อ เป็นตัวอักษรพิมพ์ใหญ่ 1 ตัว สลับกับ . 1 จุด
   เช่น W.T.O. , W.H.O.
3. สำหรับกลุ่มของตัวเลขที่ถูกแบ่งจากข้อ 1 โดยไม่มีตัวอักษรอยู่ในกลุ่มเลยจะถูกกำจัด ส่วนที่มีตัวอักษรอยู่ในกลุ่มจะคงไว้ เช่น Direct3D, RFC2345
4. ตัวอักษรทั้งหมดจะถูกทำให้เป็นพิมพ์เล็ก (Lower Case) เพราะโดยปกติแล้ว Case ของตัวอักษรจะไม่มีความสำคัญในการสืบค้นข้อมูล

ในกรณีที่กลุ่มคำ 2 กลุ่ม มีความสัมพันธ์กัน และมีความหมายพิเศษเมื่ออยู่รวมกัน หรือ ไม่ตรงตามกรณีที่กำหนดแต่มีความสัมพันธ์กันนั้น

หากคำทั้งคู่นั้นมีจำนวนคำที่ใกล้เคียงกัน อยู่ในตำแหน่งใกล้เคียงกัน คือรวมกันอยู่เป็นคู่ๆ แล้วหากคำใดคำหนึ่งได้รับเลือกให้เป็น Index Term อีกคำหนึ่งก็ควรได้รับเลือกด้วยเช่นเดียวกัน และมีโอกาสน้อยมากที่คำทั้งคู่นี้ที่อยู่ในเอกสารอื่น ซึ่งไม่มีความสัมพันธ์กันระหว่างคำทั้งสอง จะได้รับเลือกให้เป็น Index Term พร้อมกัน

ในกรณีการสืบค้นก็เช่นเดียวกัน หากคำทั้งคู่ถูกใช้ในการสืบค้นพร้อมกัน เอกสารที่มี Index term ทั้งคู่อยู่ย่อมจะอยู่ในตำแหน่งที่ดีกว่าเอกสารที่มีเพียงคำใดคำหนึ่งเป็น Index term ดังนั้นทำให้เราสามารถละเว้นการพิจารณาความสัมพันธ์ระหว่างคำเหล่านี้ไปได้

ตัวอย่างการแบ่งคำ เช่น

จะเห็นได้ว่า 3 และ . จะหายไป และ I เปลี่ยนเป็น i

หรือ How are you ? จะแบ่งได้เป็น how are you

? หายไป และ H เปลี่ยนเป็น h

X = 8*(9+3/2) จะแบ่งได้เป็น x

ตัวเลข (8932) และสัญลักษณ์ (*+/) หายไปทั้งหมด

4.1.2 Stopwords Elimination

Stopwords คือ คำที่ไม่มีความหมาย ไม่มีความสำคัญ ทำหน้าที่เชื่อมระหว่างคำต่างๆ พบได้ทั่วไปในเอกสารหรือข้อความ ไม่เกี่ยวข้องและไม่สามารถสื่อถึงสาระสำคัญของเอกสารได้ จึงไม่เหมาะที่จะใช้เป็นตัวแทนของเอกสารและไม่เหมาะที่จะนำมาทำการวิเคราะห์ สืบค้น ตัวอย่างเช่น

- Articles - a, an, the
- Pronouns - he, she, it, they, I, you
- Verbs - is, am, are, can, should
- Conjunctions - and, but, or
- Adjectives - all, still
- Adverbs - already, about, also, along
- Noun - example

ส่วนใหญ่นั้น Stopwords มักเป็นคำที่ถูกใช้บ่อยๆและพบทั่วไปในประโยคภาษาอังกฤษ ซึ่งโดยเฉลี่ยแล้วจะพบมากถึง 40 - 50 % ของคำทั้งหมดในเอกสารเลยทีเดียว ดังนั้นการตัด Stopwords ทิ้งไปจะช่วยลดระยะเวลาในการทำงาน และช่วยเพิ่มความถูกต้องด้วย

ลักษณะที่สำคัญของ Stopwords คือ มักจะมีความถี่ในการพบสูงมากๆ ซึ่งสูงกว่าความถี่ในการพบคำสำคัญทั่วไป ส่วนใหญ่แล้วจะเป็นพวก Articles, Conjunctions และ Prepositions แต่ที่เป็นคำประเภทอื่นๆก็มีเช่นเดียวกัน

ส่วนคำที่มีลักษณะตรงข้ามกับ Stopwords คือ เป็นคำสำคัญของเอกสาร ส่วนใหญ่มักเป็นคำนาม (Noun) และคำขยายนาม (Adjective) และมักมีความถี่ในการพบไม่มากหรือน้อยจนเกินไป อาจดูได้จากส่วนของเอกสารที่พบคำนั้นๆด้วย

ในการกำจัด Stopwords นั้น การใช้ Stopwords List เป็นวิธีที่ง่ายและค่อนข้างได้ผล เพราะส่วนใหญ่คำเหล่านี้มักเหมือนกันในทุกเอกสาร และเป็นกลุ่มของคำที่มีจำนวนไม่มากนัก โดยเราจะตรวจสอบคำทุกคำที่ได้จากกระบวนการตัดคำกับกลุ่มของ Stopwords ที่รวบรวมไว้ในฐานข้อมูล และกำจัดคำที่เป็น Stopwords ทิ้งไป ทั้งนี้อาจต้องระวังในการเลือกคำที่นำมาใส่ใน Stopwords Listให้เป็นคำที่ไม่มีความสำคัญต่อเอกสารจริงๆ เพราะหากผิดพลาดแล้ว อาจทำให้คำสำคัญของเอกสารนั้นๆหายไป ซึ่งส่งผลกระทบรุนแรงทั้งต่อ Index terms ที่ได้และการสืบค้นข้อมูล

ตัวอย่างในการกำจัด Stopwords เช่น

เนื่องจาก i และ have อยู่ใน Stopwords List จึงกำจัดคำเหล่านี้ทิ้ง

หรือ this, is, a, test, documents this, is, a จะถูกกำจัดออกไป

เหลือเพียง test, documents

there, are, the, ant, here there, are, the, here จะถูกกำจัดออกไป

เหลือเพียง ant

4.1.3 Stremming

คำว่า Strem หมายถึง ส่วนของคำที่เหลืออยู่หลังจากนำ Affixes (Prefixes และ Suffixes) ออกไป

ดังนั้น การทำ Stremming ก็คือ การทำให้คำศัพท์ต่างๆที่ถูกแปลงรูปไป เพื่อเปลี่ยนหน้าที่ของคำและนำไปใช้ในประโยค แต่ยังคงความหมายเดิม กลับมาอยู่ในรูปของรากศัพท์ (Root Word) เดิม ทั้งนี้เพื่อให้การสืบค้นและวิเคราะห์ข้อมูลมีประสิทธิภาพ และลดปริมาณการจัดเก็บข้อมูลที่ซ้ำซ้อนอยู่ในรูปของคำที่มีความหมายเดียวกัน แต่มีหลายรูปเหล่านี้

ตัวอย่างของคำเหล่านี้ เช่น

connect, connected, connecting, connection, connections, connector

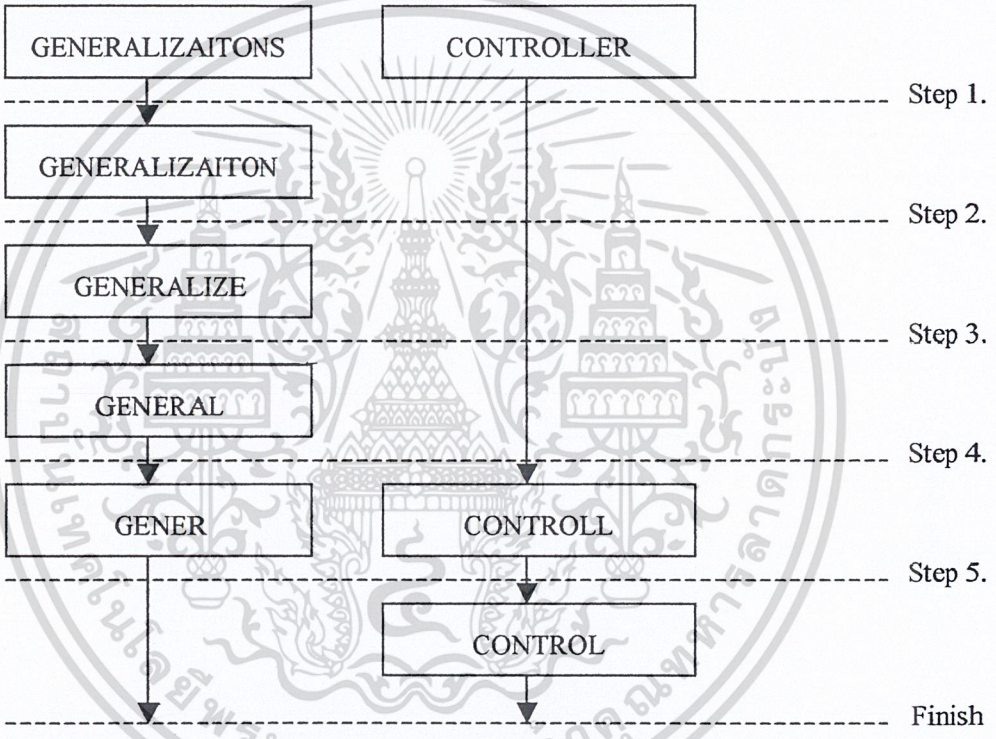
protect, protected, protecting, protection, protections, protector

เป็นต้น

ในการเปลี่ยนคำเหล่านี้กลับสู่รากศัพท์นั้น จำเป็นที่จะต้องใช้ความรู้ด้านไวยากรณ์ในภาษาอังกฤษ ทั้งในด้านการเปลี่ยนรูปของคำและโครงสร้างของประโยค ทั้งนี้กระบวนการพื้นฐานอยู่ที่การเปลี่ยนหรือลบ Prefix และ Suffix ต่างๆออกจากคำ แต่ส่วนที่สำคัญจริงๆแล้วอยู่ที่ Suffix เพราะส่วนใหญ่แล้วการเพิ่ม Prefix ให้กับคำ มักทำให้คำเปลี่ยนความหมายไป

ปัจจุบันอัลกอริทึม (Algorithm) ในการกำจัด Suffix ที่เป็นที่ยอมรับ คือ Porter's algorithm ซึ่งอาศัยการหา Suffix ที่ยาวที่สุดที่ตรงกับที่ได้กำหนดไว้ โดยใช้หลักไวยากรณ์ภาษาอังกฤษ

Porter's Algorithm แบ่งออกเป็น 5 ขั้นตอน ซึ่งคำแต่ละคำต้องผ่านขั้นตอนเหล่านี้ทั้งหมด ซึ่งคำจะเปลี่ยนแปลงไปเรื่อยๆ เมื่อผ่านแต่ละขั้นตอน เช่นดังรูปที่ 4-3 ซึ่งเป็นการยกตัวอย่างได้ดังนี้



รูปที่ 4-3 แสดงการเปลี่ยนแปลงของคำในขั้นตอนต่างๆของ Porter's Algorithm

จะเห็นได้ว่า คำเปลี่ยนรูปไปเรื่อยๆตามขั้นตอน จากผลการทดลองกับคำ 10000 คำ ได้ผลลัพธ์ ดังตาราง

Step	Number of reduced words
1	3597
2	766
3	327
4	2424
5	1373
Total	6350

ตาราง 4-1 แสดงจำนวนของคำที่เปลี่ยนแปลงในขั้นตอนต่างๆของ Porter's Algorithm (ต่อ 10000 คำ)

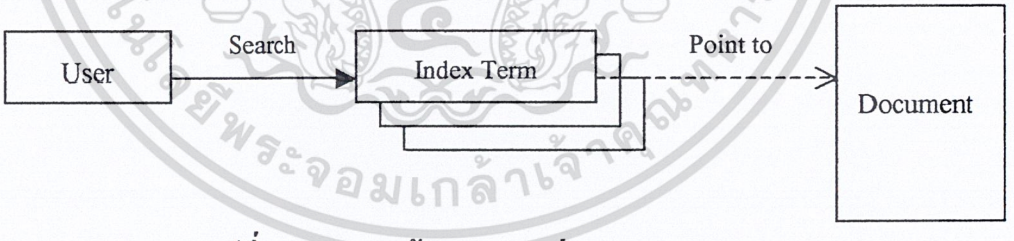
เนื่องจากคำที่ได้จากอัลกอริทึมนี้ อาจไม่ถูกต้องตรงตามรากศัพท์เดิม เพราะอัลกอริทึมมีหน้าที่เพียงทำให้คำที่มาจากรากศัพท์เดียวกันมีรูปเหมือนกัน โดยไม่รับประกันว่าคำที่ได้จะมีความหมายหรือไม่ ดังนั้นผลลัพธ์จึงอาจเป็นคำที่ไม่มีความหมาย เช่น happy -> happi, airline -> airlin, revive -> reviv เป็นต้น ดังนั้นจึงมีความจำเป็นจะต้องเก็บคำอ้างอิงไว้ เพื่อจะสามารถแสดงผลให้ผู้ใช้ได้ในภายหลัง โดยเลือกเก็บคำใดคำหนึ่งจากกลุ่มคำที่มาจากรากศัพท์เดียวกัน

นอกจากการใช้อัลกอริทึมในการตัด Suffix ทิ้งแล้ว ในบางกรณีที่คำมีการเปลี่ยนไปโดยสิ้นเชิง หรือเป็นกรณีที่เป็นข้อยกเว้น จำเป็นจะต้องมีฐานข้อมูลเกี่ยวกับคำเหล่านี้และรากศัพท์เดิมเอาไว้ เพื่อใช้ในการแปลงกลับมาเป็นรากศัพท์เดิมด้วย

4.1.4 การเลือก index (Index Terms Selection)

การเลือก Index terms เป็นขั้นตอนที่สำคัญมากในการจัดการเอกสาร เพราะ Index terms ถือเป็นตัวแทนของเอกสาร ในการสืบค้น เราจะไม่สืบค้นที่ตัวเอกสารโดยตรง แต่จะสืบค้นผ่าน Index terms ดังนั้นผลลัพธ์ของการสืบค้นจะถูกต้องหรือไม่ ส่วนใหญ่ขึ้นอยู่กับ Index terms ที่เลือกใช้

คำหรือกลุ่มคำที่ถูกเลือกมาเป็น Index terms จะต้องเป็นคำที่มีความสัมพันธ์กับเนื้อหาภายในเอกสาร เป็นส่วนที่เกี่ยวข้องกับใจความของเอกสาร และมีลักษณะเฉพาะเอกสาร ไม่ใช่คำที่พบได้ทั่วไปในเอกสารต่างๆ และเมื่อพิจารณา Index terms ทั้งหมดแล้วต้องเป็นตัวแทนของเอกสารทั้งฉบับได้โดยสมบูรณ์ดังแสดงไว้รูปที่ 4-4



รูปที่ 4-4 แสดงการค้นหาเอกสารผ่าน Index terms

จำนวนของ Index terms ก็มีส่วนสำคัญ หากเลือก Index terms น้อยเกินไปก็จะไม่สามารถเป็นตัวแทนของเอกสารทั้งฉบับได้ หากเลือกมากเกินไปก็จะทำให้การค้นหาขาดความแม่นยำ ทั้งนี้จำนวนของ Index term ในเอกสารแต่ละฉบับอาจมีไม่เท่ากันได้ ทั้งนี้เนื่องจากความยาวและเนื้อหาภายในเอกสารที่แตกต่างกัน

การเลือก Index terms นั้น แบ่งออกเป็น 2 วิธี คือ

1. <u>การเลือกโดยมนุษย์</u> วิธีนี้ใช้คนเป็นผู้อ่านเอกสาร แล้วเลือกคำสำคัญออกมาเป็น Index terms เป็นวิธีที่มีความถูกต้องแม่นยำมาก หากผู้เลือกมีความชำนาญ แต่เป็นวิธีที่ต้องใช้เวลานาน และต้องการ

บุคลากรที่มีความสามารถ และมีความรู้เกี่ยวกับเนื้อหาภายในเอกสารเป็นอย่างดี ซึ่งปกติจะเป็นหน้าที่ของเจ้าของเอกสารหรือบรรณารักษ์

เนื่องจากเป็นกระบวนการที่ใช้คน เอกสารเดียวกันจึงอาจมี Index terms แตกต่างกันได้ ขึ้นกับผู้เลือก นอกจากนี้ยังอาจมีโอกาสเกิด Index terms ไม่ครบถ้วนได้ง่าย

2. <u>การเลือกโดยอัตโนมัติ</u> วิธีนี้ใช้โปรแกรมเป็นผู้พิจารณาเลือก Index terms โดยส่วนใหญ่พิจารณาจากความถี่และตำแหน่งของคำที่พบในเอกสาร ซึ่งต้องมีขั้นตอนในการปรับค่า และการคำนวณต่างๆ เพื่อให้มีความถูกต้องเพิ่มขึ้น

วิธีการนี้มีโอกาสผิดพลาดได้มากกว่าการเลือกโดยมนุษย์ ซึ่งส่วนใหญ่มักเกิดจากการเลือก Index terms ผิด จึงมักใช้คนตรวจสอบผลลัพธ์ที่ได้อีกครั้ง

สำหรับรายละเอียดของวิธีการเลือก Index terms อัตโนมัติจะกล่าวถึงในบทต่อไป

4.1.5 Thesauri

Thesaurus คือ กลุ่มคำที่มีความหมายใกล้เคียงกันหรือเหมือนกัน กล่าวคือ สิ่งหนึ่งๆอาจถูกเรียกได้ในหลายรูปแบบขึ้นกับผู้เรียก เช่น เราอาจเรียกจักรยานว่า bicycle, cycle หรือ bike ก็มีความหมายถึงจักรยาน 2 ล้อเช่นเดียวกัน

Thesaurus อาจเป็นได้ทั้งคำเหมือน (Synonyms) กลุ่มคำ (Phrases) หรือคำที่เกี่ยวข้องในหัวข้อนั้นๆ จุดประสงค์ในการสร้าง Thesauri List ก็คือ

1. การพยายามหาคำที่เป็นตัวแทนของกลุ่ม เพื่ออ้างอิงทั้งในการสร้าง Index term และการค้นหาข้อมูล
2. ช่วยให้ผู้ใช้สามารถค้นหาข้อมูลได้ตรงกับความต้องการมากขึ้น
3. อาจช่วยผู้ใช้ในการแบ่งระดับความสัมพันธ์ (Hierarchical relationship) ของคำที่ค้นหา ว่าต้องการให้ค้นหาในความหมายที่กว้างมากแค่ไหน

ตัวอย่างในการใช้ Thesaurus เพื่อสร้างคำอ้างอิง เพื่อเป็นตัวแทนของกลุ่ม เช่น

coward	USE FOR	craven
craven	USE	coward

จะเห็นได้ว่า เราใช้คำว่า coward ในการอ้างอิงแทนคำว่า craven ซึ่งในการสร้าง Index term เราจะใช้ coward แทน craven และเมื่อผู้ใช้ค้นหาคำว่า craven เราจะเปลี่ยนเป็นค้นหาคำว่า coward แทน โดยที่ความสัมพันธ์ระหว่างคำในกรณีนี้ต้องเป็นความสัมพันธ์ที่สามารถใช้ทดแทนกันได้ โดยสมบูรณ์

นอกจากนี้ในกรณีที่คำสามารถใช้แทนกันได้โดยสมบูรณ์นี้ เราอาจนำ Thesaurus มาใช้เพียงด้านใดด้านหนึ่ง เช่น ใช้เฉพาะส่วน Query หรือ ใช้ เฉพาะส่วนของการทำ Index terms ก็ได้ ซึ่งจะให้ผลลัพธ์เหมือนกัน แต่กรณีที่ใช้ส่วน Query จะทำให้การสืบค้นช้าลง ส่วนกรณีที่ใช้ส่วนการทำ Index terms ก็จะทำให้ใช้เนื้อที่ในการเก็บข้อมูลมากขึ้น

นอกจาก Synonyms แล้ว เรายังสามารถใช้คำที่เป็น Near-synonyms มาใช้เป็น Thesaurus ได้ โดยการแบ่งระดับความสัมพันธ์ออกเป็นลำดับชั้น (Hierarchical relationship) ซึ่งส่วนใหญ่จะแบ่งออกเป็น ความหมายเชิงกว้าง (Broader Term : BT) และความหมายเชิงแคบ (Narrower Term : NT) ซึ่งผู้ใช้จะเป็นผู้กำหนดว่าต้องการให้สืบค้นแบบใด โดยในกรณีที่ไม่มีการแบ่งระดับความสัมพันธ์ เราจะเรียกรวมว่าเป็น Related Term (RT)

$$RT = NT \cup BT$$

การแบ่งความสัมพันธ์ออกเป็นระดับชั้นนั้น อาจทำให้ผู้ใช้เกิดความสับสนได้ง่าย ต้องใช้เนื้อที่จัดเก็บข้อมูลมาก ทำให้เกิดความยุ่งยากและเพิ่มเวลาที่ใช้ในการสืบค้นด้วย ดังนั้น จึงมักนิยมใช้ Thesaurus แบบความหมายเหมือนกันมากกว่า

ข้อควรระวังในการสร้าง Thesauri ก็คือ ความสัมพันธ์ระหว่างคำ ในบางกรณีจะมีลักษณะเป็นความสัมพันธ์แบบทางเดียว ($A \in B$ แต่ $B \notin A$) เช่น tunnel = tube แต่ tube $\neq$ tunnel เป็นต้น และนอกจากนี้ คำศัพท์หนึ่งคำอาจมีได้หลายความหมาย ซึ่งในแต่ละความหมายอาจมีกลุ่มของคำเหมือนที่แตกต่างกันไป

บทที่ 5
การเลือก index อัตโนมัติ

เนื่องจากการเลือก Index terms เป็นงานที่ต้องใช้เวลา ความชำนาญ และความรู้เกี่ยวกับเนื้อหาภายในเอกสารประกอบกัน รวมทั้งอาจเกิดข้อผิดพลาดได้ง่าย จึงมีแนวความคิดที่จะใช้คอมพิวเตอร์มาทำหน้าที่แทนในส่วนนี้ โดยพิจารณาจากข้อมูลเชิงสถิติของการพบคำนั้นๆในเอกสาร และวิเคราะห์ด้วยวิธีการต่างๆ เพื่อให้ได้ข้อสรุปว่าคำนั้นๆ สมควรถูกเลือกเป็น Index term หรือไม่

วิธีการวิเคราะห์เพื่อหา Index terms แบบอัตโนมัตินั้นมีอยู่หลายวิธี แต่วิธิที่นิยมใช้ ได้แก่

5.1 การนับความถี่ของคำ (Frequency Count)

วิธีการนี้จะใช้ความถี่ของคำที่พบในเอกสารมาคำนวณตามที่กำหนด ซึ่งส่วนใหญ่แล้วจะเลือกคำที่มีความถี่มาก แต่ไม่มากจนเกินไป เป็น Index term ทั้งนี้เพราะคำที่พบมากๆในเอกสารนั้น ส่วนใหญ่มักเป็น Stopwords หรือคำที่ไม่มีความหมายต่อเอกสาร แต่คำสำคัญที่เกี่ยวข้องกับเอกสารนั้น มักพบมากกว่าคำปกติ แต่ไม่มากจนเกินไปนัก

นอกจากนี้เรายังสามารถเพิ่มความถูกต้องให้กับการเลือก โดยการใช้ตำแหน่งที่พบคำในเอกสารมาประกอบการคำนวณ โดยกำหนดค่าน้ำหนักต่อการพบคำในแต่ละส่วนให้ต่างกัน เช่น คำที่พบในส่วน Abstract ให้ค่าน้ำหนักมากกว่าคำที่พบในส่วนเนื้อเรื่อง เป็นต้น ซึ่งค่าน้ำหนักนี้จะถูกนำไปใช้เป็นข้อมูลในการเลือก Index terms ต่อไป

วิธีนี้เป็นวิธีพื้นฐานที่ทำได้ง่าย ไม่ซับซ้อน และมีประสิทธิภาพดีพอสมควร หากมีการกำจัด Stopwords ก่อนทำการเลือกได้อย่างมีประสิทธิภาพ แต่ข้อเสียของวิธีนี้ คือ กฎเกณฑ์ที่กำหนดไว้คงที่ ไม่มีการเปลี่ยนแปลง ทำให้ขาดความยืดหยุ่น

5.2 การใช้ Neural Network

วิธีการนี้ ใช้แนวความคิดเหมือนกับวิธีการนับความถี่ของคำ ทั้งในส่วนของข้อมูลที่จำเป็นต้องใช้ในการวิเคราะห์ และขั้นตอนการวิเคราะห์ แต่ต่างกันที่วิธีการคำนวณไม่คงที่ สามารถเปลี่ยนแปลงได้ โดยอาศัยการเรียนรู้ผ่าน Neural Network

วิธีนี้จะมีการทำงานที่ยุ่งยากกว่าวิธีการนับความถี่คำ ใช้เวลาในการทำงาน และการเรียนรู้มาก มีการออกแบบและสร้างที่ยุ่งยากกว่า แต่จะสามารถเรียนรู้จากข้อมูลใหม่ๆได้ ทำให้มีการปรับตัวให้ทำงานได้อย่างถูกต้องมากขึ้น

5.2.1 ทฤษฎี Neural Network

เป็นการจำลองการทำงานของสมองมนุษย์มาเป็นรูปแบบการประมวลผลข้อมูล ปกติแล้วในสมองของมนุษย์จะประกอบด้วยเซลล์ประสาทจำนวนมาก ซึ่งทำงานอย่างอิสระ แต่ละเซลล์จะมีการเชื่อมต่อระหว่างกันเรียกว่า Synapse มีส่วนที่เป็น Input ของเซลล์ เรียกว่า Dendride และส่วนที่เป็น Output เรียกว่า Axon ซึ่งการติดต่อทำได้โดยการสร้างความต่างศักย์ (Voltage) ซึ่งถือว่าเซลล์ประสาทเหล่านี้เป็นหน่วยย่อยที่สุดในการทำงานของสมอง

ซึ่งการทำงานจะเป็นไปในลักษณะที่สัญญาณประสาทถูกส่งต่อกันไปเป็นทอดๆ จากเซลล์หนึ่งไปยังอีกเซลล์หนึ่ง โดยแต่ละเซลล์จะทำหน้าที่ในการคำนวณในส่วนย่อย และมีการเชื่อมต่อถึงกันที่ซับซ้อน เพื่อให้ได้ผลการคำนวณที่ต้องการ

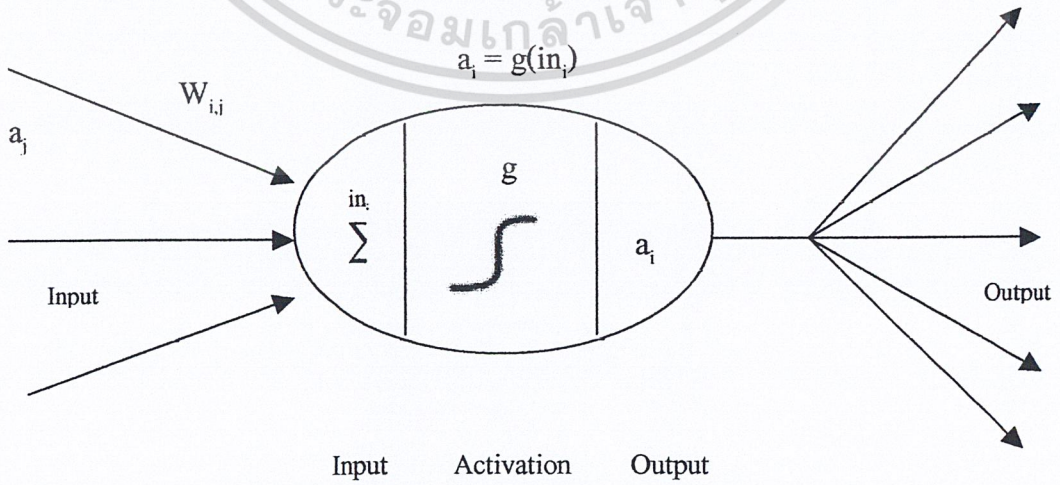
ข้อดีของการทำงานแบบนี้ คือ เราไม่จำเป็นต้องทราบถึงรูปแบบการคำนวณที่แน่นอน แต่ปล่อยให้เป็นหน้าที่ของกระบวนการเรียนรู้ ที่จะทำหน้าที่สร้างรูปแบบการคำนวณที่ใกล้เคียงกับการคำนวณค่าที่ถูกต้องขึ้นมา โดยอาศัยการปรับค่าที่อยู่ภายในหน่วยย่อย หรือ ค่าที่บริเวณจุดเชื่อมต่อระหว่างหน่วยย่อยเหล่านี้ ว่าจะให้ค่านำเข้า (Input) ใดมีความสำคัญเท่าไร

เมื่อนำมาใช้ในทางคอมพิวเตอร์ เราจึงสร้างหน่วยประมวลผลซึ่งมีลักษณะคล้ายเซลล์ประสาท โดยให้แต่ละหน่วยประกอบด้วยส่วนรับข้อมูล (Input) ที่มีค่าน้ำหนัก (Weight) อยู่เพื่อระบุค่าความสำคัญของ Input นั้นๆ ภายในจะมีการประมวลผลโดยส่วนแรกจะนำค่า Input ทั้งหมดมารวมกัน จากนั้นจึงนำไปแทนค่าใน Activation Function แล้วได้ออกมาเป็น Output ดังแสดงไว้ที่รูปที่ 5-1 ซึ่ง Output จากหน่วยหนึ่งจะมีค่าเพียงค่าเดียวเท่านั้น

เขียนเป็นสมการได้ ดังนี้

$$in_i = \sum_j W_{j,i} a_j$$

$$a_i = g(in_i)$$



รูปที่ 5-1 แสดงหน่วยประมวลผลใน Neural Network

เราสามารถเลือกใช้ Activation Function ได้หลายรูปแบบ แต่ที่นิยมใช้ที่สุดคือ Sigmoid Function เพราะสะดวกในการหา Derivative ซึ่งมักจำเป็นต้องใช้ในกระบวนการเรียนรู้ต่อไป

$$sigmoid(x) = 1 / (1+e^{-x})$$

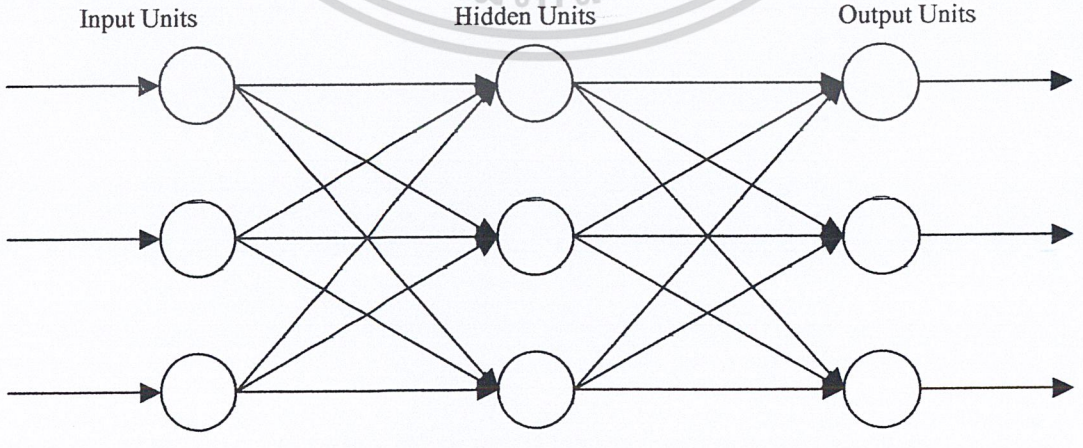
จากส่วนพื้นฐานนี้ สามารถนำมาต่อเป็น Network ได้หลายรูปแบบ ซึ่งแต่ละรูปแบบจะมีความเหมาะสมในการใช้งานแตกต่างกันไป

Algorithm	Type	Function
Hopfield	Recursive	Optimization
Multi-layered perceptron	Feedforward	Classification
Kohonen	Self-organizing	Data coding
Temporal difference	Predictive	Forecasting

ตาราง 5-1 แสดงรูปแบบการเชื่อมต่อพื้นฐานชนิดต่างๆ

โดยในแต่ละรูปแบบยังสามารถแบ่งเป็นรูปแบบย่อยๆที่แตกต่างกันไป หรือนำเอารูปแบบต่างๆมาผสมกัน เพื่อให้ได้รูปแบบใหม่ ได้อีก ขึ้นกับการออกแบบ และความเหมาะสมในการนำไปใช้งานประเภทต่างๆ

ในที่นี้ เราจะให้ความสนใจกับ Multi-layered perceptron Network เท่านั้นดังแสดงรูปที่ 5-2 เนื่องจากเป็นชนิดที่นิยมใช้กันอย่างกว้างขวาง และเหมาะสมกับงานการจำแนกและแบ่งกลุ่มข้อมูล ตามรูปแบบ (Pattern) ซึ่ง Network แบบนี้จะมีลักษณะแบ่งออกเป็นชั้น(Layer) ซึ่งประกอบไปด้วย Input units, Output units และ Hidden units ซึ่งจำนวนชั้นของ Hidden units จะแตกต่างกันไปตามความซับซ้อนของการคำนวณที่ใช้ กล่าวคือ ยิ่งการคำนวณซับซ้อนมาก ก็ต้องมี Layers ของ Hidden Unit มากขึ้น



รูปที่ 5-2 ตัวอย่างของ Multi-layered Perceptron

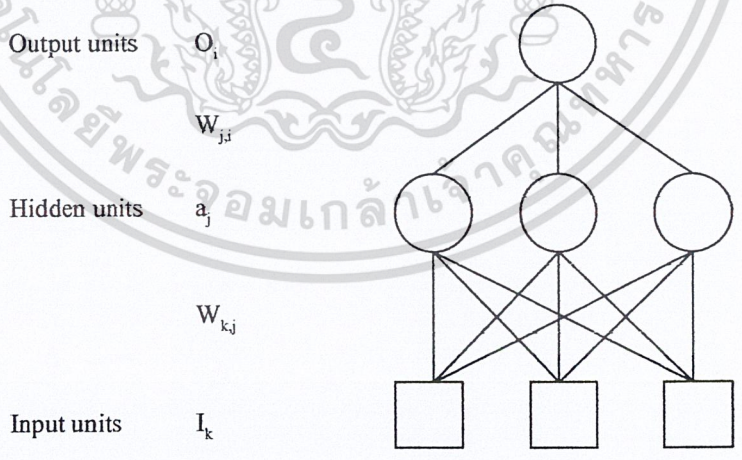
ข้อมูลที่รับเข้ามาทาง Input Unit นั้นจะถูกผ่านไปให้แต่ละหน่วยประมวลผลที่อยู่ใน Layer เดียวกัน เพื่อทำการประมวลผล และสัญญาณเหล่านี้จะถูกส่งผ่านต่อไปเรื่อยๆ สู่ Layer ถัดไป ซึ่งจะมีการให้ค่าความสำคัญ(Weight) ของ Input ที่รับเข้ามาจากหน่วยต่างๆแตกต่างกันไป และสัญญาณจะถูกส่งต่อระหว่าง Layer ไปเรื่อยๆ จนกระทั่งไปถึง Output unit ซึ่งจะส่งผลลัพธ์ที่ได้ออกมา

โดยค่าน้ำหนักของ Input ต่างๆนั้น จะถูกปรับโดยวิธีการเรียนรู้(Learning) โดยวิธีการเรียนรู้ ก็คือ การนำข้อมูลที่ทราบผลลัพธ์แล้ว มาใช้เป็น Input และเปรียบเทียบค่า Output ที่ได้กับผลลัพธ์หาค่าความผิดพลาด จากนั้นจึงนำค่าความผิดพลาดที่ได้มาทำการปรับน้ำหนัก (Weight) ภายใน Neural Network ซึ่งวิธีการปรับค่านั้นขึ้นอยู่กับวิธีการเรียนรู้ที่เลือกใช้

ซึ่งวิธีการเรียนรู้นั้น มีหลายรูปแบบ ขึ้นกับการออกแบบ โครงสร้างภายในของ Neural Network และประเภทของงานที่นำไปใช้

5.2.2 Back-propagation

เป็นวิธีการเรียนรู้รูปแบบหนึ่ง ซึ่งเป็นที่นิยมใช้ โดยอาศัยหลักการกระจายความรับผิดชอบต่อความผิดพลาดออกไปยังหน่วยย่อยทุกหน่วยใน Network เท่าๆกัน โดยจะกระจายความผิดพลาดลงไปในแต่ละ Layer ซึ่งจะสังเกตได้ว่าใน Multi-layered Perceptron Network นั้น ค่าที่ได้จากแต่ละ Layer จะส่งผลกระทบถึง Layer ถัดไป ในลักษณะของการยกกำลัง ดังนั้นการกระจายความผิดพลาดจาก Layer หลังไปสู่ Layer ต้นๆ นั้น จำเป็นต้องใช้หลักการอินทิเกรต เพื่อให้ได้อัตราความผิดพลาดที่เกิดขึ้นใน Layer นั้น



รูปที่ 5-3 แสดง Neural Network และค่าต่างๆ ที่ใช้ในการคำนวณ Back Propagation

จากรูปที่ 5-3 สามารถอธิบายได้ดังนี้ หากเรามี Network ที่ประกอบด้วย Input unit k,Hidden unit j และ Output unit i จะเขียนเป็นสมการได้ดังนี้

$$W_{j,i} \leftarrow W_{j,i} + (\alpha * a_j * Err_i * g'(in_i))$$

เมื่อ α แทน ค่าการเรียนรู้ และ Err_i แทน ผลต่างระหว่าง Output ของ i กับผลที่ถูกต้อง

ให้ $\Delta_i = Err_i\, g'(in_i)$, $\quad W_{j,i} \leftarrow W_{j,i} + (\alpha * a_j * \Delta_i)$

ซึ่งเมื่อพิจารณาค่า Δ สำหรับ Layer ถัดไปจะได้ว่า

$$\Delta_j = g'(in_j)\sum_i W_{j,i}\Delta_j$$

$$W_{k,j} \leftarrow W_{k,j} + \alpha * I_k * \Delta_j$$

เมื่อ I_k แทน ค่าที่ได้จาก Input unit k

อัตราความเร็วของการเรียนรู้ขึ้นอยู่กับตัวแปร α ซึ่งหากมีค่ามากจะทำให้ค่าภายใน Neural Network เปลี่ยนแปลงไปอย่างรวดเร็วระหว่างการเรียนรู้ แต่ก็มีข้อเสียคือ จะทำให้ค่าที่ได้ไม่ละเอียด ซึ่งทำให้ผลการทำงานของ Neural Network ผิดพลาด ไม่เที่ยงตรงได้

ในกรณีที่เราใช้ค่า α ไม่มากนัก อาจทำให้ในการทำ Back Propagation เพียงรอบเดียว ไม่สามารถปรับค่าให้ถูกต้องได้ ดังนั้นส่วนใหญ่ในการเรียนรู้แต่ละครั้งนั้น จะเป็นการทำ Back Propagation หลายๆรอบกับข้อมูลกลุ่มเดิม โดยยิ่งค่า α ต่ำ จำนวนรอบในการทำงานก็ยิ่งต้องเพิ่มมากขึ้น

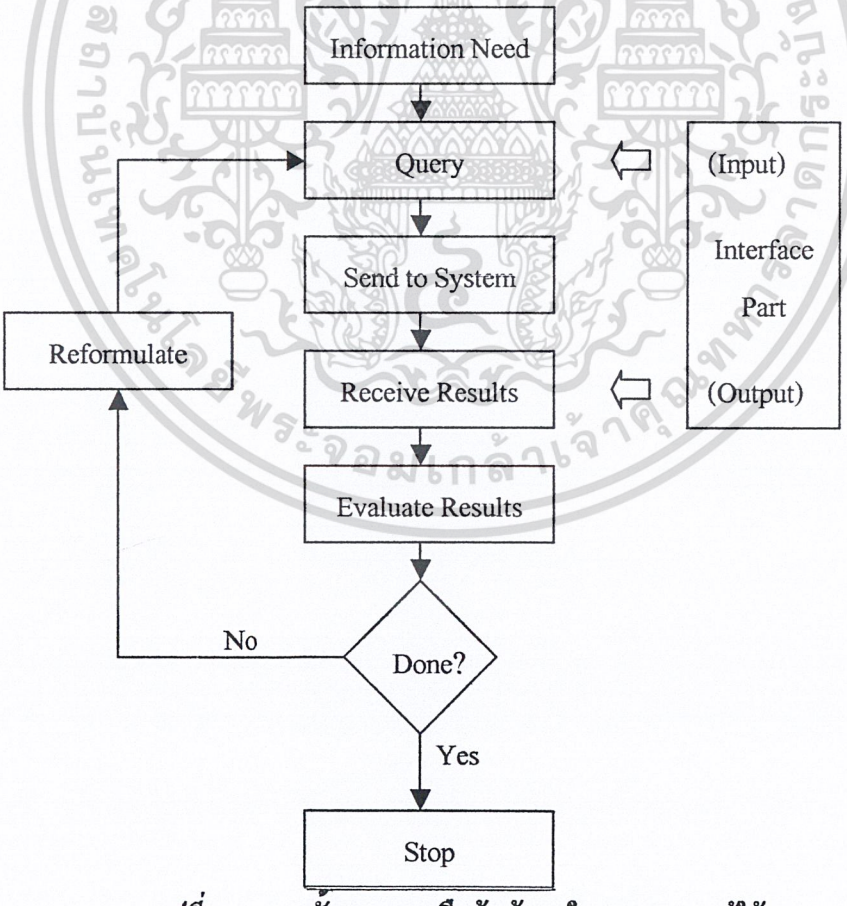
นอกจากนี้ปริมาณและการกระจายตัวของข้อมูลตัวอย่างที่ใช้ในการเรียนรู้ (Training Set) นั้นก็เป็นสิ่งสำคัญที่มีผลต่อการเรียนรู้มาก ข้อมูลที่นำมาใช้เป็นตัวอย่างในการเรียนรู้นั้นต้องเป็นข้อมูลที่เป็นมาตรฐานตามปกติ ไม่ใช่ข้อมูลที่มีรูปแบบ โครงสร้าง ที่ผิดไปจากข้อมูลปกติ นอกจากนี้ต้องมีจำนวนมากพอที่จะทำให้มีรูปแบบที่หลากหลายพอที่จะครอบคลุมถึงลักษณะปกติของข้อมูลที่จะทำการวิเคราะห์ได้หมด

บทที่ 6
ส่วนติดต่อผู้ใช้

ผลลัพธ์ของการสืบค้นข้อมูลจะถูกต้องมากน้อยเพียงใด ส่วนที่สำคัญส่วนหนึ่งก็คือความสามารถในการสื่อสารกันระหว่างผู้ใช้กับระบบ กล่าวคือ ผู้ใช้ต้องสามารถระบุความต้องการในการสืบค้นให้ระบบรับรู้ได้อย่างชัดเจน และระบบก็ต้องสามารถแสดงผลลัพธ์ของการสืบค้นให้ผู้ใช้สามารถเข้าใจและเลือกใช้ได้อย่างมีประสิทธิภาพ ซึ่งนับเป็นส่วนสำคัญที่ต้องคำนึงถึงในการออกแบบระบบ ทั้งในด้านรูปแบบและวิธีการ ซึ่งจะต้องสัมพันธ์กับกลไกการสืบค้นที่เลือกใช้อีกด้วย

6.1 ขั้นตอนในการสืบค้น

หากพิจารณาขั้นตอนในการสืบค้นข้อมูลทั่วไปแล้ว สามารถเขียนแผนภาพแสดงขั้นตอนในมุมมองของผู้ใช้ได้ ดังแสดงไว้ในรูปที่ 6-1 ดังนี้



รูปที่ 6-1 แสดงขั้นตอนการสืบค้นข้อมูลในมุมมองของผู้ใช้

จากรูปที่ 6-1 จะเห็นได้ว่า ส่วนที่ผู้ใช้ติดต่อกับระบบ มีอยู่ 2 ส่วน คือ

1. ส่วน Query (Input สู่ระบบ)
2. ส่วน Receive Results (Output ออกจากระบบ)

6.2 การ Query

เป็นส่วนที่ผู้ใช้อธิบายความต้องการในการสืบค้นให้ระบบรับทราบ ดังนั้นส่วนนี้จึงเป็นส่วนที่ต้องให้ความสำคัญในการออกแบบมาก เพราะต้องใช้วิธีการที่ผู้ใช้สามารถเข้าใจได้ง่าย และต้องมีความยืดหยุ่นพอที่จะตอบสนองความต้องการในรูปแบบต่างๆในการสืบค้นของผู้ใช้ได้ครบถ้วน

รูปแบบในการ Query แบ่งออกเป็นรูปแบบใหญ่ ๆ ได้ดังนี้

6.2.1การ Query โดยใช้ Keyword (Keyword-Based Queries) แบ่งออกเป็น 4 ประเภท ได้แก่

Single-Word Queries

เป็นรูปแบบพื้นฐานของการสืบค้นข้อมูล กล่าวคือ เมื่อผู้ใช้ใส่คำหรือกลุ่มคำ เข้าสู่ระบบ ระบบจะทำการแบ่งข้อมูลที่ได้รับออกเป็นคำ จากนั้นจึงนำคำที่ได้เหล่านี้ไปเปรียบเทียบกับคำที่เป็น Index terms ของเอกสารต่างๆ เพื่อเปรียบเทียบต่อไป

ส่วนการจัดลำดับของผลลัพธ์ที่ได้ (Ranking) นั้น อาจจัดตามจำนวนคำที่ตรงกับกลุ่มคำที่ใช้สืบค้น หรือความถี่ของคำที่พบในเอกสาร

Context Queries

นอกจากจะทำการค้นหาคำที่ผู้ใช้ระบุแล้ว วิธีการนี้ ผู้ใช้ยังสามารถระบุ Context หรือ คำที่อยู่ใกล้เคียงกับคำนั้นได้อีกด้วย ซึ่งระบบจะทำการค้นหาคำและตรวจสอบคำที่อยู่ใกล้เคียงกับคำนั้น เพื่อทำการพิจารณาเปรียบเทียบกับ Context ที่ได้รับมาด้วย ซึ่งแบ่งออกเป็น 2 กรณีย่อย คือ

- Phrase วิธีนี้จะตรวจสอบเฉพาะบริเวณใกล้ๆ หรือคำที่อยู่ติดกันเท่านั้น
- Proximity วิธีนี้จะตรวจสอบในบริเวณข้างเคียงตามที่ได้กำหนดไว้ ซึ่งอาจกำหนดได้ตั้งแต่ในประโยคเดียวกันไปจนถึงในย่อหน้าเดียวกัน

ตัวอย่างของการค้นหาวิธีนี้ เช่น

ค้นหาคำว่า 'Retrieval' ที่มี Context เป็น 'Enhance'

Phrase, '… enhance the retrieval …'

Proximity, '… enhance the power of retrieval …'

การจัดลำดับของผลลัพธ์นั้น อาจจัดแบบเดียวกับ Single-Word Queries หรือ นำระยะห่างระหว่างคำและ Context มาเป็นตัวแปรในการคำนวณด้วยก็ได้

Boolean Queries

วิธีนี้เป็นการเชื่อมต่อคำที่ต้องการสืบค้นแต่ละคำหรือแต่ละกลุ่มด้วยเครื่องหมายทางตรรกศาสตร์ ซึ่งเครื่องหมายเหล่านี้ปกติที่ใช้กันมีอยู่ 3 ประเภท ได้แก่

OR เช่น (e_1 OR e_2) ค้นข้อมูลที่ตรงกับ e_1 หรือ e_2

AND เช่น (e_1 AND e_2) ค้นข้อมูลที่ตรงกับทั้ง e_1 และ e_2

BUT เช่น (e_1 BUT e_2) ค้นข้อมูลทั้งหมดที่ตรงกับ e_1 แต่ไม่ตรงกับ e_2 (=NOT)

ซึ่งในรูปแบบนี้จะไม่มีการเรียงลำดับของผลลัพธ์ เนื่องจากข้อมูลมีโอกาสเพียงเป็นสมาชิกหรือไม่เป็นสมาชิกของเซตของผลลัพธ์เท่านั้น

เนื่องจากผู้ใช้ส่วนใหญ่ประสบปัญหาในการใช้งาน Query แบบนี้ เพราะค่อนข้างยุ่งยาก ต้องมีความรู้พื้นฐานเกี่ยวกับตรรกศาสตร์ ต่อมาจึงมีการผ่อนคลายกฎเกณฑ์ระหว่างเครื่องหมายลงบ้าง โดยใช้การเรียงลำดับผลลัพธ์เข้าช่วย เพื่อให้ผู้ใช้สามารถใช้งานได้สะดวกขึ้น

Natural Language

ในกรณีของ Fuzzy Boolean แล้ว ระหว่าง AND กับ OR นั้นจะไม่มีความแตกต่างกันอีกต่อไป เนื่องจากหากข้อมูลส่วนใดตรงกับกลุ่มคำที่ใช้สืบค้นมาก ก็จะมีค่าความเป็นสมาชิกในเซตของผลลัพธ์สูง ข้อมูลส่วนใดตรงกับกลุ่มคำที่ใช้สืบค้นน้อย จะมีค่าความเป็นสมาชิกในเซตของผลลัพธ์ต่ำ

ส่วนในกรณีของ NOT นั้น หากอนุญาตให้ผู้ใช้สามารถใส่กลุ่มคำที่ไม่ต้องการในการสืบค้นเข้ามาด้วย โดยหากข้อมูลใดมีคำในกลุ่มคำนี้อยู่จะทำให้ค่าความเป็นสมาชิกในเซตของผลลัพธ์ลดลง

ซึ่งจากเหตุผลทั้งหมดนี้ เมื่อเราใช้การเรียงลำดับผลลัพธ์ตามค่าความเป็นสมาชิกแล้ว ก็จะทำให้เราไม่จำเป็นต้องใช้เครื่องหมายทางตรรกศาสตร์เข้าช่วยในการ Query อีกต่อไป และหากเพิ่มส่วนจัดการกับข้อความเพื่อช่วยกำจัดคำที่ไม่สำคัญต่อการสืบค้นออกไป ผู้ใช้จะสามารถ Query ได้ด้วยภาษาธรรมชาติ ซึ่งวิธีนี้จะทำให้ผู้ใช้เข้าใจง่าย และสะดวกในการใช้งาน

6.2.2การ Query โดยใช้ส่วนของคำ (Pattern Matching)

วิธีนี้เป็นการตรวจสอบว่าส่วนของคำที่ใช้ในการสืบค้นตรงกับส่วนของคำใน Index term หรือไม่ ตัวอย่างการค้นหาในวิธีนี้ เช่น

สืบค้นคำว่า 'Compt'

ผลลัพธ์ที่ได้จะเป็นเอกสารที่มี Index term เป็น '<u>compute</u>r', '<u>computa</u>tion' หรือ '<u>computi</u>ng'

นอกจากนี้ วิธีนี้ยังสามารถใช้สัญลักษณ์ต่างๆช่วยสร้างรูปแบบเพื่อระบุลักษณะของกลุ่มคำผู้ใช้ต้องการได้ เช่น ใช้ %c แทน ตัวอักษร 1 ตัว, ใช้ * แทนการซ้ำ 1 ถึง N ครั้ง เป็นต้น ซึ่งส่วนนี้จะแตกต่างกันไปตามการออกแบบของแต่ละระบบ

แต่ข้อเสียของวิธีการนี้ คือ ผู้ใช้ต้องทำความเข้าใจกับระบบพอสมควร จึงจะใช้งานได้ตรงตามความต้องการ นอกจากนี้ยังมีโอกาสได้รับข้อมูลที่ไม่ต้องการในการสืบค้นได้ง่าย และต้องใช้เวลาในการสืบค้นนานกว่าปกติ เนื่องจากต้องเปรียบเทียบกับ Index term ทีละคำ ซึ่งในแต่ละคำก็ต้องพยายามแบ่งเป็นส่วนย่อย เพื่อค้นหาส่วนที่ตรงกัน

6.2.3การ Query โดยใช้โครงสร้างข้อมูล (Structural Queries)

ลักษณะการรับข้อมูลแบบนี้ ออกแบบมาเพื่อใช้กับข้อมูลที่มีการแบ่งเป็นส่วนๆอย่างชัดเจน เช่น หนังสือที่สามารถแบ่งได้เป็นบทๆ เป็นต้น ซึ่งสามารถใช้ร่วมกับการสืบค้นแบบอื่นๆ เพื่อเพิ่มประสิทธิภาพในการสืบค้นได้ แบ่งออกเป็น 3 ประเภท ได้แก่

Fixed Structure

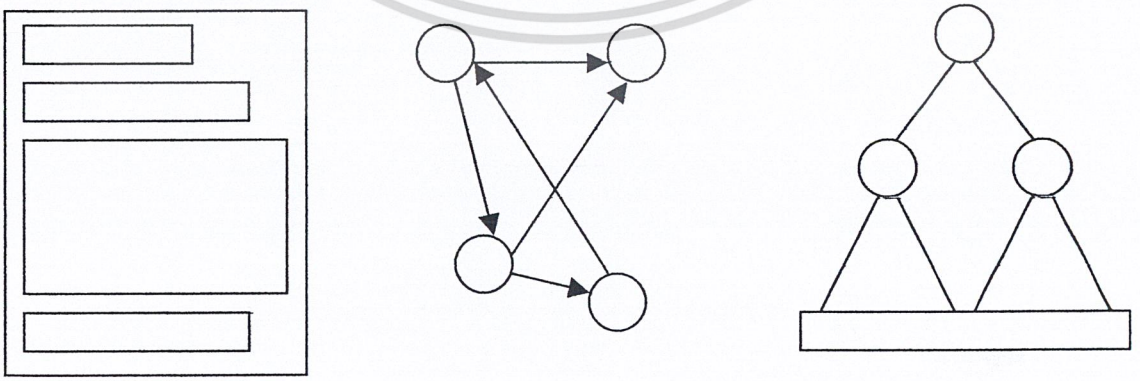
เหมาะที่จะใช้กับข้อมูลที่แบ่งออกเป็นส่วนชัดเจน ลักษณะคล้ายการกรอกแบบฟอร์ม โดยแบ่งส่วนข้อมูลที่จะใช้สืบค้นออกจากกันเป็นส่วนๆอย่างชัดเจน เช่น สืบค้น mail ได้ตาม ชื่อผู้ส่ง, ชื่อผู้รับ และชื่อ mail เป็นต้น ซึ่งจะเห็นได้ว่ารูปแบบนี้เหมาะกับข้อมูลที่เป็น Relational Database ที่มีจำนวน Column คงที่และตายตัว โดยแต่ละส่วนอาจเป็น Column ในตารางของฐานข้อมูลที่ใช้สืบค้น

Hypertext

รูปแบบนี้ค่อนข้างมีโครงสร้างที่ไม่แน่นอน โดยอาศัยการเชื่อมต่อ (Link) ระหว่างข้อมูลที่เกี่ยวข้องกัน ทำให้ผู้ใช้สามารถสืบค้นข้อมูลที่เกี่ยวข้องกันได้จากการเชื่อมต่อเหล่านี้ จะเห็นได้ว่า การสืบค้นวิธีนี้เหมาะที่จะใช้กับเอกสารบน Web

Hierarchical Structure

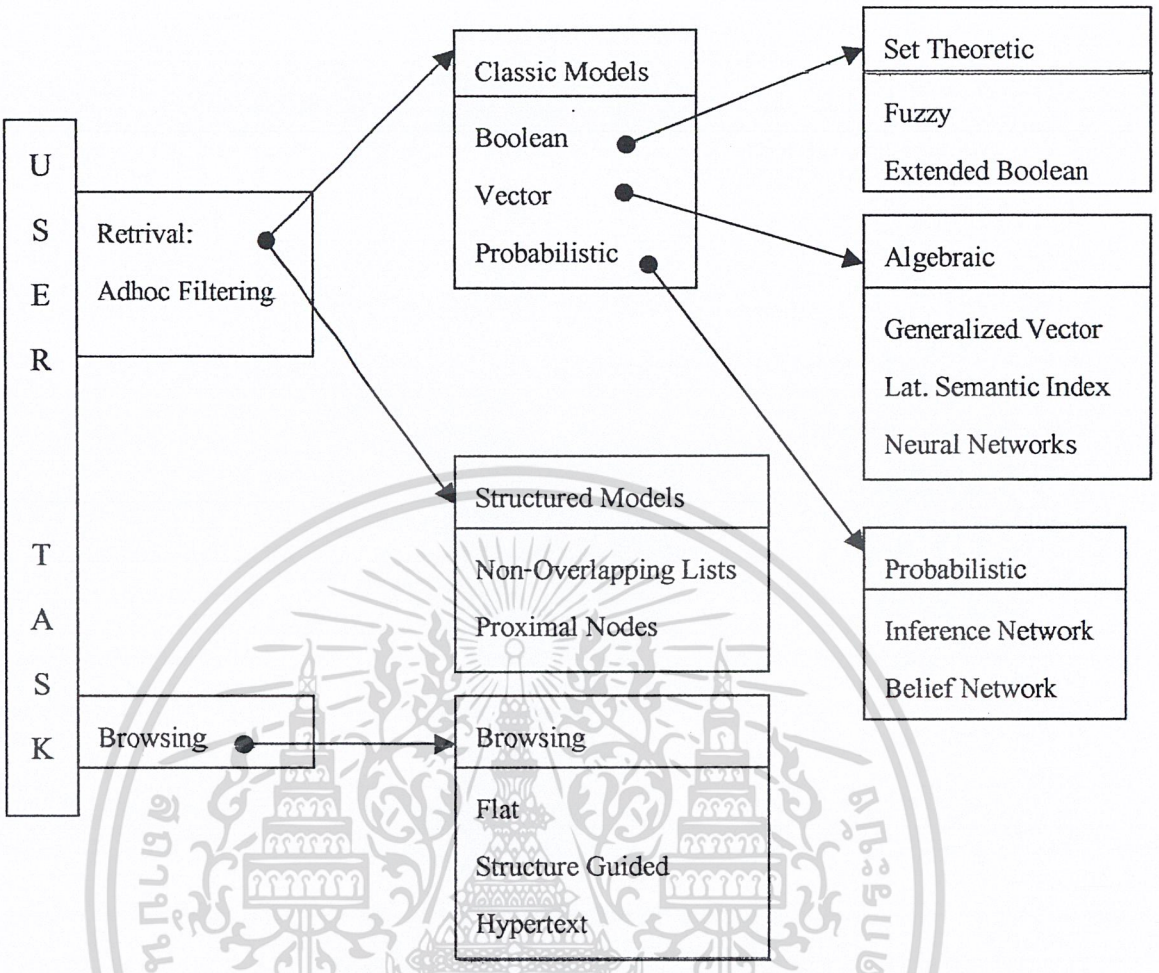
รูปแบบนี้เป็นการสืบค้นโดยผู้ใช้สามารถเข้าถึงส่วนต่างๆของข้อมูลได้เต็มที่ สามารถกำหนดการค้นหาลงไปได้ถึงส่วนประกอบย่อยของข้อมูล เช่น อาจสืบค้นเอกสารจากคำบรรยายรูปภาพได้ เป็นต้น



(ก) (ข) (ค)

รูปที่ 6-2 แสดงการสืบค้นตามโครงสร้างข้อมูล

(ก) Fixed Structure (ข) Hypertext Structure (ค) Hierarchical Structure



รูปที่ 6-3 ความสัมพันธ์ของรูปแบบต่าง ๆ ที่นำมาใช้ในระบบค้นคืน

สรุป

เนื่องจากเอกสารในโครงงานนี้มีรูปแบบส่วนใหญ่คล้ายกัน และข้อมูลที่เกี่ยวข้องกับเอกสารจัดเก็บในลักษณะของ Relational Database จึงเหมาะที่จะใช้รูปแบบของ Fixed Structure ในการสืบค้น และเนื่องจากใช้ Fuzzy Model เราจึงใช้ Natural Language Queries เป็นพื้นฐานในการสืบค้น เพื่อที่ผู้ใช้จะสามารถเข้าใจและสืบค้นได้ง่าย

ผลลัพธ์

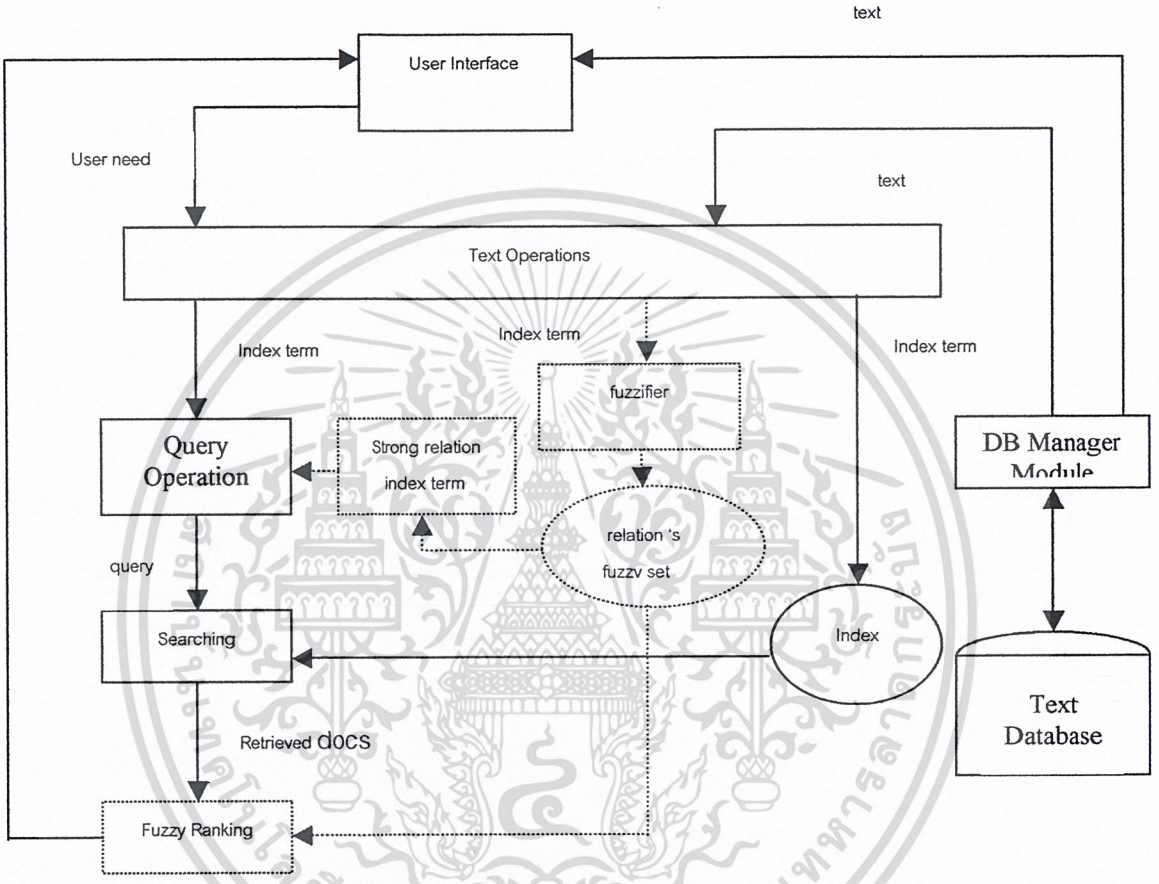
ในการแสดงผลลัพธ์ของ Fuzzy Model จำเป็นต้องมีการเรียงลำดับ เพื่อแสดงถึงระดับความเป็นสมาชิกในเซตของผลลัพธ์ของเอกสารนั้นๆ ซึ่งอาศัยการคำนวณตามหลักการของ Fuzzy Modelนอกจากนี้ยังต้องมีการแสดงรายละเอียดของเอกสาร เพื่อให้ผู้ใช้สามารถเห็นภาพรวมของเอกสารและสามารถเลือกเอกสารที่ต้องการได้ โดยไม่ต้องเข้าไปอ่านเนื้อหาของเอกสารทั้งหมด

นอกจากนี้ในการออกแบบต้องคำนึงถึงความสะดวกสบาย และทำให้ผู้ใช้เข้าใจได้ง่าย สามารถเข้าถึงเอกสารที่ต้องการได้อย่างรวดเร็ว

บทที่ 7

การออกแบบระบบ

7.1 องค์ประกอบของระบบ



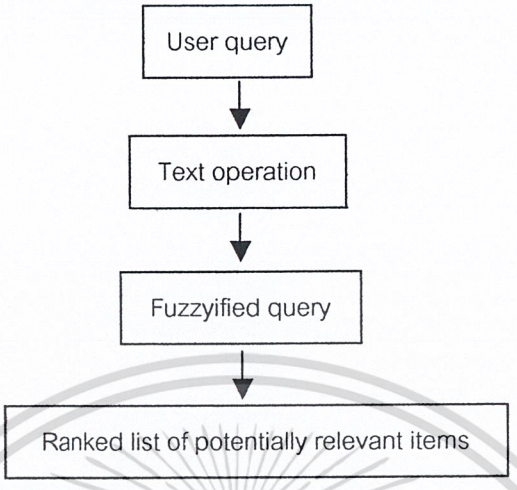
รูปที่ 7-1 ระบบการค้นคืนโดยภาพรวม

จากรูปที่ 7-1 ได้นำระบบการค้นคืนสารสนเทศที่ได้กล่าวไว้ในบทที่ 2 มาขยายประสิทธิภาพ โดยจากรูปจะนำเอาโครงสร้าง ของ ฟัซซี่ มาประกอบเข้าไปกับระบบเดิม

เราสามารถแบ่ง ระบบค้นคืนออกเป็นระบบ ย่อย ได้ 3 ส่วนด้วยกันคือ

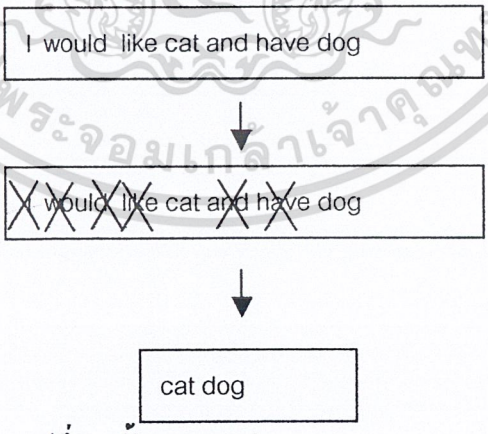
1. ส่วนการค้นหาเอกสาร โดยใช้ ทฤษฎี ฟัซซี่ลอจิก
2. ส่วนการทำ index อัตโนมัติ
3. ส่วนการแก้ไขข้อมูลที่เกิดจากบรรณารักษ์

7.2 การค้นหาเอกสารโดยใช้ ทฤษฎี ฟัซซี่ลอจิก



รูปที่ 7-2 ขั้นตอนการค้นหาเอกสารโดยใช้ ทฤษฎี ฟัซซี่ลอจิก

จากรูป 7-2 ซึ่งแสดงขั้นตอนการค้นหา เอกสาร โดยใช้หลักการของ ฟัซซี่ ลอจิก โดยในขั้นแรกนั้น โปรแกรมจะรับความต้องการของผู้ใช้โดยผู้ใช้จะส่ง query เข้ามา โดย คำที่ผู้ใช้ต้องการค้นหา อาจมาในรูปแบบภาษาธรรมชาติ ได้ โดย โปรแกรมจะทำหน้าที่ช่วยการจัดโครงสร้างของภาษาเสียใหม่ ในส่วนของ Text Operations ยกตัวอย่างเช่น เมื่อผู้ใช้ใส่ query เข้ามาเป็น "I would like papers with have program and JSP" ในขั้นตอนการทำ Text operations จะทำการกำจัด คำที่ไม่ต้องการทิ้งออกไปให้เหลือแต่ คำที่ใช้ในการ query ได้



รูปที่ 7.3 ขั้นตอนการทำ Text Operation

เมื่อได้ index term ที่สามารถใช้ในการค้นหาได้แล้วในที่นี้คือ “cat” และ “dog” และทำการแยกคิด index term แต่ละตัว ในขั้นแรกจะนำ index term “cat” มาคิดก่อนโดยจะนำ index term “cat” ไปหาดูว่าในเอกสารไหนบ้างที่มีคำว่า “cat” ในตาราง ซึ่งการจัดเก็บเป็นดังนี้

Keyword	PNumber
cat	C0001
cat	C0078
dog	C0080
...	...
...	...

จากตัวอย่าง จะได้ เอกสารที่มี “cat” เป็น index term คือ C0001 และ C0078 หมายถึงเป็นเอกสารที่มี index term “cat” เป็นสมาชิกอย่างเต็มตัว ดังนั้นเอกสารเหล่านี้จะมีค่าความเป็นสมาชิกกับ index term “cat” เท่ากับ 1 แล้วนำ index term “cat” ไปหาคำเหมือน (Thesaurus) ซึ่งindex termที่เป็นคำเหมือนกับ index term “cat” จะถูกนำไปค้นหาด้วย แต่ index term เหล่านี้ จะมีค่าความเป็นสมาชิกไม่เท่ากับ index term “cat”

จากนั้นนำ index term “cat” ไปหาว่ามีความสัมพันธ์กับindex term อื่น ๆ อีกอะไรบ้างจากสูตรในการหาความสัมพันธ์ระหว่างindex termดังนี้

$$c_{i,j} = \frac{n_{i,j}}{n_i + n_j - n_{i,j}}$$

โดยจากสูตรสามารถอธิบายได้ดังนี้

$c_{i,j}$ คือ ความสัมพันธ์คู่index term i กับ j

$n_{i,j}$ คือ จำนวนเอกสารที่มีทั้ง index term i และj เป็นindex term

n_i คือ จำนวนเอกสารที่มี index term i เป็นindex term

n_j คือ จำนวนเอกสารที่มี index term j เป็นindex term

แต่ในทางปฏิบัติ ถ้าต้องทำการคำนวณความสัมพันธ์ทุก ๆ ครั้งที่มีการค้นหาจะทำเสียเวลาเป็นอย่างมากในการค้นคืนข้อมูลในผู้ใช้ ดังนั้น จะทำการจัดเก็บข้อมูลความสัมพันธ์ของ index termไว้ล่วงหน้าเพื่อความรวดเร็วในการค้นคืนข้อมูล ในการคิดคำนวณหาความสัมพันธ์ของindex termนั้นจะกระทำในขั้นตอนการใส่ข้อมูลของ บรรณารักษ์ หรือ การแล้วไขข้อมูลที่เกิดจาก บรรณารักษ์ หรือ ผู้ดูแลระบบเท่านั้น โดยการคำนวณทุก ๆ ครั้งจะคำนวณเฉพาะ index term ที่มีการเปลี่ยนแปลงเท่านั้น เพื่อใช้เวลาน้อยที่สุดในการคำนวณ

ตัวอย่างตารางความสัมพันธ์ของindex term

Index 1	Index 2	Relate
Cat	kitty	0.5
Cat	dog	0.25
Dog	puppy	0.5

เมื่อต้องการค้นหา index term คำว่า "cat" เราจะพบว่า index term "kitty" และ index term "dog" มีความสัมพันธกับindex term "cat" ดังนั้นต้องใช้index term "kitty" และ index term "dog" มาใช้ในการค้นหาเอกสารด้วย แต่ความเป็นสมาชิกของเอกสารที่มีต่อindex term "cat" จะไม่เท่ากับ 1 โดยจะขึ้นอยู่กับ คู่ความสัมพันธ์นั้น ๆ เช่น index term "cat" กับindex term "kitty" คือ 0.5 ดังนั้นเอกสารที่มี index term "kitty" ประกอบอยู่จะมีความเป็นสมาชิกต่อindex term "cat" เท่ากับ 0.5 และ เอกสารที่มี index term "dog" ประกอบอยู่ จะมีความเป็นสมาชิกต่อindex term "cat" เท่ากับ 0.25 แล้วนำไปเก็บไว้เพื่อรอการคำนวณ

ต่อมาก็ใช้ index term "dog" ในการค้นหาในทำนองเดียวกับการค้นหาโดยใช้ index term "cat" ในการค้นหาเมื่อได้ทำการเลือกเอกสารและกำหนดดความสัมพันธ์ของเอกสารแล้วก็นำไปคำนวณโดยใช้สูตร

$$\mu_{i,j} = 1 - \prod_{k_l \in d_j} (1 - c_{i,l})$$

$\mu_{i,j}$ ความเป็นสมาชิกของเอกสารกับindex term

$\prod_{k_l \in d_j} (1 - c_{i,l})$ อินเวิด ของค่าความเป็นสมาชิกรวมในอินเด็กซ์ทุก ๆ เทอม

และเมื่อทำคำนวณเสร็จแล้วผลที่ได้ คือ ค่าความเป็นสมาชิกรวมของทุก ๆ อินเด็กซ์ที่มีผลต่อเอกสารนั้น ๆ โดยindex termหลัก คือ index term i เมื่อมีindex termหลาย ๆ ตัวเป็นตัว query จะต้องคำนวณตาม ลอจิก อีกครั้งเพื่อเป็นผลรวมของความสัมพันธ์ของ index termทุก ๆ ตัวที่เป็นตัว query ในการค้นคืนต่อเอกสารแต่ละฉบับโดยมีสูตรในการคำนวณดังต่อไปนี้

การกระทำ ลอจิก แบบ อินเตอร์เซ็ก

$$\mu_{first \cap president \cap u.s.a}(x) = \mu_{first}(x).\mu_{president}(x).\mu_{u.s.a}(x) \qquad \forall x \in U$$

การกระทำ ลอจิก แบบ ยูเนี่ยน

$$\mu_{first \cup president \cup u.s.a}(x) = (1-(1-\mu_{first}(x)).(1-\mu_{president}(x)).(1-\mu_{u.s.a}(x))) \qquad \forall x \in U$$

การกระทำ ลอจิก แบบ คอมพรีเมนต์

$$\bar{\mu}_{first}(x) = (1 - \mu_{first}(x)) \qquad \forall x \in U$$

เมื่อคำนวณตามรูปแบบเบื้องต้นแล้ว จะได้ผลรับที่ถูกเก็บไว้ในตัวแปรซึ่งมีรูปแบบดังนี้

Paper Number	Relation
C00001	1
C25789	0.667
R54873	0.873
R6579	0.248

เมื่อได้ข้อมูลออกมาแล้วนั้นยังไม่สามารถนำมาแสดงผลได้เพราะจำเป็นต้องทำการเรียงลำดับเอกสาร ที่มีความเป็นสมาชิกสูง ๆ ออกมาแสดงก่อน โดยต้องอาศัย การเรียงข้อมูลจากมากไปหาน้อย ในระบบค้นคืนข้อมูลโดยอาศัยทฤษฎีฟัซซี่ลอจิก นี้ ได้ใช้การเรียงข้อมูลแบบ Quick Sort

การเรียงลำดับข้อมูลโดยใช้ Quick Sort Algorithm

เนื่องจากการค้นคืนข้อมูลความสำคัญนอกเหนือจากความถูกต้องแล้ว ส่วนที่สำคัญมากอีกประการหนึ่งคือความเร็วในการค้นคืน การเรียงลำดับข้อมูลเป็นปัจจัยที่สำคัญยิ่งต่อการเพิ่มประสิทธิภาพของเรื่องความเร็ว โดยระบบค้นคืนโดยอาศัยทฤษฎีฟัซซี่ลอจิกนี้ ได้ใช้การเรียงข้อมูลแบบ Quick Sort Algorithm โดยคุณสมบัติของ Quick Sort Algorithm คือด้านความเร็วในการเรียงข้อมูล ซึ่งความซับซ้อนจะเท่ากับ O(n log n) ซึ่งจะเร็วกว่าการเรียงข้อมูลแบบอื่น ๆ เช่นการเรียงข้อมูลแบบ Bubble Sort Algorithm ซึ่งความซับซ้อนจะสูงกว่าคือ O(n^2) ซึ่งจะทำให้ประสิทธิภาพเรื่องความเร็วลดต่ำลง

เมื่อทำการเรียงลำดับเอกสารเสร็จเรียบร้อยแล้ว ข้อมูลก็อยู่ในรูปที่สามารถนำเสนอต่อผู้ค้นข้อมูลได้

7.3 การทำ index อัตโนมัติ

การทำ index อัตโนมัตินั้น แบ่งออกได้เป็น 2 ขั้นตอนการทำงาน ได้แก่

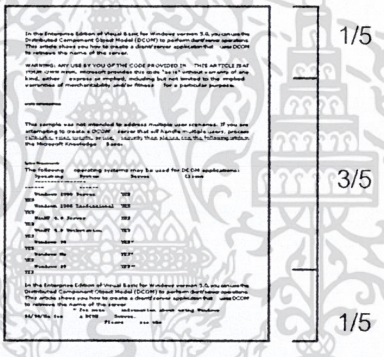
1) การเก็บสถิติความถี่ของคำต่างๆจากเอกสาร
2) การวิเคราะห์ข้อมูลด้วย Neural Network

การเก็บสถิติของคำจากแฟ้มข้อมูล

สำหรับข้อมูลที่จะนำมาวิเคราะห์นั้น นอกจากความถี่ของการพบคำในเอกสารแล้ว ตำแหน่งของคำที่ปรากฏในเอกสารก็เป็นสิ่งที่เราให้ความสนใจเช่นเดียวกัน

จากสมมติฐานที่ว่า คำที่พบในส่วนต้นของเอกสาร เช่น ส่วน Abstract, Title และส่วนท้ายของเอกสาร เช่น ส่วน Conclusion, Reference น่าจะเป็นคำที่เหมาะที่จะเป็น Index term ของเอกสารมากกว่าคำที่พบในส่วนกลางของเอกสาร ดังนั้นเราจึงมีการแบ่งข้อมูลที่จะเข้าสู่ Neural Network ออกเป็น 4 ส่วนตามรูปที่ 7-4 ได้แก่

- ความถี่ของการพบคำในส่วนต้นของเอกสาร (n < 1/5 ของจำนวนคำทั้งหมดในเอกสาร)
- ความถี่ของการพบคำในส่วนกลางของเอกสาร
  (n >= 1/5 ของจำนวนคำทั้งหมดในเอกสาร AND n <= 4/5 ของจำนวนคำทั้งหมดในเอกสาร)
- ความถี่ของการพบคำในส่วนปลายของเอกสาร (n > 4/5 ของจำนวนคำทั้งหมดในเอกสาร)
- จำนวนของคำทั้งหมดในเอกสาร



รูปที่7-4 แสดงการแบ่งเอกสารเป็นส่วน เพื่อใช้ในการเลือก Index terms

ในส่วนนี้ต้องมีการทำ Text Operations ก่อน โดยทำส่วนแบ่งคำ, ส่วนกำจัด Stopwords, ส่วนกำจัด Suffix (Porter's Algorithm) ก่อน และคำที่กำจัด Suffix แล้วเหมือนกันจะรวมเข้าด้วยกัน ซึ่งในขั้นตอนนี้เราใช้การสร้าง Hash function โดยใช้อักษรตัวแรก ร่วมกับ การสร้าง Binary Tree รูปที่ 7-5 เพื่อใส่ข้อมูล ซึ่งจะช่วยลดระยะเวลาในการเปรียบเทียบระหว่างคำลงได้

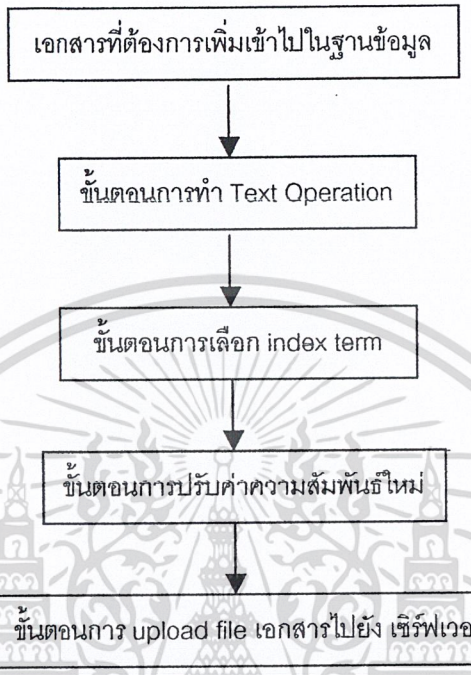
7.4 ส่วนการแก้ไขข้อมูลที่เกิดจากบรรณารักษ์

ส่วนนี้เป็นงานที่เกิดจาก บรรณารักษ์ หรือ ผู้ดูแลระบบซึ่งแบ่งเป็นงานย่อย ๆ ได้ออกเป็น 2 ส่วนด้วยกันคือ

1. งานในการเพิ่มเอกสารเข้าไปในระบบค้นคืนข้อมูล
2. งานในการแก้ไขข้อมูลของเอกสารซึ่งเก็บในฐานข้อมูล

7.4.1 งานในการเพิ่มเอกสารเข้าไปในระบบค้นคืนข้อมูล

โดยรูปแบบการ ใส่ข้อมูลเพิ่มไปยัง ฐานข้อมูลที่จัดเก็บไว้มีรูปแบบดังนี้



รูปที่ 7-6 แสดงขั้นตอนการเพิ่มเอกสารเข้าไปในระบบค้นคืนข้อมูล

จากรูปที่ 7-6 สามารถอธิบายรายละเอียดของขั้นตอนการเพิ่มเอกสารเข้าไปในระบบค้นคืนข้อมูลได้ดังนี้

ขั้นตอนที่ 1 เอกสารที่ต้องการเพิ่มเข้าไปในฐานข้อมูล ขั้นตอนนี้เป็นขั้นตอนที่ บรรณารักษ์ต้องกรองข้อมูลบางส่วนเอง เช่น ชื่อเรื่อง คำอธิบายเกี่ยวกับ เอกสารเรื่องนั้น ผู้แต่งเอกสารนั้น โดยข้อมูลเหล่านี้จะถูกจัดเก็บลง ฐานข้อมูลเพื่อ ใช้ในการค้นคืน โดยการค้นคืนประเภทนี้จะ ไม่ได้ใช้ การค้นคืนแบบ ฟัซซี่ลอจิก แต่จะเป็นการค้นคืนในรูปแบบ extract เช่นผู้ใช้ อาจมี ความต้องการ ชื่อเรื่องที่มี คำว่า JSP programming เป็นส่วนประกอบของ ชื่อเรื่อง ในการหาเอกสารที่ผู้ใช้ป้อนเฉพาะ ชื่อเรื่องนั้น จะใช้ความสามารถของ ภาษา เอสคิวแอล(SQL) ในการค้นหา โดยในตัวอย่างนี้ จะใช้คำสั่ง เอสคิวแอล(SQL) คือ

```
Select * from Paper where description like  '%"jsp programming"%'
```

ประโยคภาษา เอสคิวแอล(SQL)ข้างบนนี้จะทำการค้นหาเอกสารที่ มีคำว่า "jsp programming" เป็นส่วนประกอบภายใน column description เช่นเดียวกับการค้นหาโดยใช้ชื่อผู้แต่งในการค้นหาจะทำในทำนองนี้เช่นกัน

ขั้นตอนที่ 2 ขั้นตอนการทำ Text Operation เป็นขั้นตอนการทำให้ คำให้อยู่ในรูปแบบที่จะใช้ในการจัดเก็บ โดยรายละเอียดในการทำ Text Operation สามารถดูได้ที่ บทที่ 4 จะเป็นขั้นตอนการทำ Text Operation และเมื่อทำให้คำ อยู่ในรูปแบบที่สามารถใช้ในการค้นหาได้แล้วนั้น จะส่ง คำ เหล่านี้ไปยังขั้นการทำ index อัตโนมัติ

ขั้นตอนที่ 3 ขั้นตอนการทำ index อัตโนมัติ โดยในขั้นตอนนี้สามารถดูได้ที่ บทที่ 5 เรื่องการทำ index อัตโนมัติ และ หัวข้อ 7.3 ประกอบ และเมื่อเลือกค่าindex termได้แล้วนั้นindex termพวกนี้จะถูกไปจัดเก็บในฐานข้อมูล และ นำไปปรับค่าความสัมพันธ์ในขั้นตอนที่ 4

ขั้นตอนที่ 4 ขั้นตอนการปรับค่าความสัมพันธ์ของindex term เมื่อได้รับค่า index termมาแล้วจะต้องนำไปปรับค่าในตารางความสัมพันธ์ของ index termที่อยู่ในตารางความสัมพันธ์

Index 1	Index 2	Relate
Cat	kitty	0.5
Cat	dog	0.25
Dog	puppy	0.5

โดยในการคำนวณนั้นถ้าเราคำนวณค่าความสัมพันธ์ใหม่ทั้งหมดจะทำให้เสียเวลานานมาก เพราะ การคำนวณหาความสัมพันธ์ในแต่ละครั้งนั้น ถ้ามีจำนวนindex term ทั้งหมด n จำนวน จะมีความซับซ้อนถึง O(n^2) ดังนั้นในการคำนวณแต่ละครั้งนั้นเราจะคำนวณเฉพาะ index termที่มีการเปลี่ยนแปลงเท่านั้น โดยหลักการทำก็ง่าย ๆ แต่ให้ความถูกต้องเท่ากันคือ จะเลือกindex termที่เกิดจากการเพิ่มเอกสารเข้าไปใหม่ ยกตัวอย่างเช่นมีindex termที่เกิดจากเอกสารใหม่ k เทอม ค่าความซับซ่อน จะคิดได้เป็น O(k^2) ซึ่งเวลาในการคำนวณจะแตกต่างกันมาก

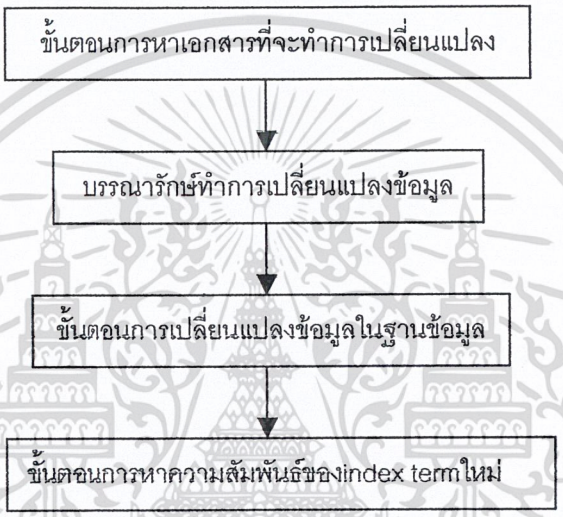
ขั้นตอนที่ 5 ขั้นตอนการส่ง File เอกสารไปยัง เซิร์ฟเวอร์ ในขั้นตอนนี้จะทำการส่ง File ไปยัง เซิร์ฟเวอร์ โดยมีรูปแบบอินพุด FORM ดังนี้

```
<FORM METHOD="POST" ACTION="/jspsmartupload/jsp/sample1.jsp" ENCTYPE="multipart/form-data">
  <INPUT TYPE="FILE" NAME="FILE1" SIZE="50"><BR>
</FORM>
```

เมื่อทำการส่ง File เอกสารเสร็จสิ้นก็เป็นการเสร็จขั้นตอนการเพิ่มเอกสารเข้าไปในระบบค้นคืนข้อมูล

7.4.2 งานในการแก้ไขข้อมูลของเอกสารซึ่งเก็บในฐานข้อมูล

ข้อมูลของเอกสารในบางโอกาสจำเป็นต้องมีการแก้ไข เช่น แก้ไขชื่อผู้แต่ง แก้ไขชื่อเรื่อง หรือแก้ไขindex term ซึ่งเป็นไปได้ทั้งสิ้น โดยเฉพาะการแก้ไขindex termนั้นเป็นสิ่งจำเป็นมาก เพราะ index termที่ได้จากการทำ อินเด็กซ์อัตโนมัติอาจไม่ถูกต้อง 100% ดังนั้นต้องมีกระบวนการที่เอื้ออำนวยความสะดวกให้แก่ บรรณารักษ์ หรือ ผู้ดูแลระบบ ในการเปลี่ยนแปลงข้อมูลที่จัดเก็บได้ง่าย ๆ โดยขั้นตอนในการแก้ไขข้อมูลของเอกสาร สามารถใช้รูปข้างล่างนี้ในการอธิบายได้



รูปที่ 7-7 ขั้นตอนงานในการแก้ไขข้อมูลของเอกสารซึ่งเก็บในฐานข้อมูล

จากรูปที่ 7-7 สามารถอธิบายรายละเอียดของขั้นตอนงานในการแก้ไขข้อมูลของเอกสาร ได้ดังนี้

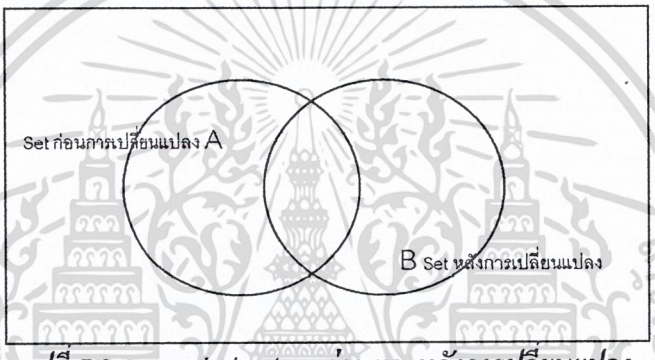
ขั้นตอนที่ 1 ขั้นตอนการหาเอกสารเพื่อทำการเปลี่ยนแปลง ขั้นตอนนี้จะรับ อินพุต ของบรรณารักษ์ หรือผู้ดูแลระบบ โดยจะนำอินพุตที่ได้นั้นไปหารูปแบบของภาษา เอสคิวแอล(SQL) ที่ใช้ในการค้นการเอกสารได้โดยภาษา เอสคิวแอล(SQL)จะมีรูปแบบดังนี้

Select * from paper where "----------"

โดยส่วนที่ขีดเส้นนั้นหมายถึงจำนำ อินพุต ไปเขียนในส่วนของ where เพื่อใช้ในการเลือกข้อมูลในฐานข้อมูลออกมาแสดงในแก่ บรรณารักษ์ หรือ ผู้ดูแลระบบ แล้ว บรรณารักษ์ จะทำการเลือกเอกสารที่ต้องการแก้ไขอีกที เพราะ ค่า Recordset ที่ถูกเลือกออกมาอาจไม่ใช่เป็นค่าเดียวแต่อาจเป็น set ของ record ก็เป็นได้ เมื่อทำการเลือกเอกสารแล้วจะนำข้อมูลภายในการจัดเก็บของเอกสารฉบับ นั้น ๆ มาแสดง

ขั้นตอนที่ 2 ขั้นตอนการเปลี่ยนแปลงข้อมูล จะนำตัวแปรมาจดจำ เซต ของ index ก่อนการเปลี่ยนแปลงไว้ก่อน หลังจากนั้นบรรณารักษ์จะทำการเปลี่ยนแปลงข้อมูลได้ทั้ง ชื่อผู้แต่ง ชื่อเรื่อง หรือ index term

ขั้นตอนที่ 3 ขั้นตอนการเปลี่ยนแปลงข้อมูลในฐานข้อมูล หากข้อมูลผู้แต่ง หรือ ข้อมูล ชื่อเรื่อง เปลี่ยนแปลงจะทำได้โดยง่าย โดยการส่ง ภาษา เอสคิวแอล(SQL) ไป update ข้อมูลในฐานข้อมูลเท่านั้น แต่ การเปลี่ยน แปลงอินเด็กเทอมนั้น จะต้องมีขั้นตอนการคำนวณหาความสัมพันธ์ของindex termมาเกี่ยวข้องด้วยดังนั้นต้องมีโครงสร้างของโปรแกรมที่ช่วยในการคำนวณหาค่าความสัมพันธ์ใหม่เนื่องจากการเปลี่ยนแปลง index term โดย จะมีการเก็บค่า set ของindex term ก่อนและหลังการเปลี่ยนแปลง เพื่อช่วยให้การคำนวณหาความสัมพันธ์ของindex termได้มีความเร็วมากขึ้น โดยมีแนวคิดดังนี้ดังแสดงในรูปที่ 7-8



รูปที่ 7-8 เซตของindex termก่อนและหลังการเปลี่ยนแปลง

โดยในการคำนวณนั้นจะคำนวณเฉพาะส่วนที่มีการเปลี่ยนแปลงเท่านั้น โดยจะคิดส่วนที่ลบออกก่อน ในที่นี้คือ A-B แล้วนำไปคำนวณหาค่าความสัมพันธ์ใหม่ และ คำนวณหาค่าความสัมพันธ์ของindex termใหม่โดยใช้ส่วน B-A ในการคำนวณหาค่าความสัมพันธ์ใหม่

7.5 การออกแบบฐานข้อมูล

PAPER

←----p,k.----→

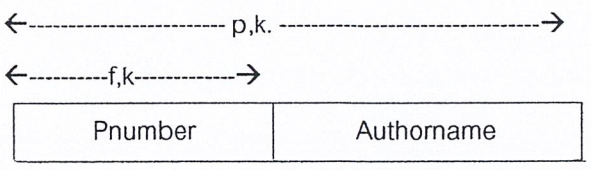
Pnumber	Pname	Abstract	IndexRef	Location	Filesize

ตาราง paper ทำหน้าที่เก็บข้อมูลของเอกสารแต่ละเอกสาร

- Pnumber เป็นคีย์หลักทำหน้าที่ในการอ้างอิงเอกสาร
- Pname ชื่อหัวเรื่องของเอกสาร
- Abstract บทคัดย่อของเอกสาร

- IndexRef index term อ้างอิงเพื่อใช้ในการแก้ไขเพราะ index ที่ใช้ในการหาค่าความสัมพันธ์จะอยู่ในรูปแบบที่เป็นรากศัพท์ ดังนั้นผู้แก้ไขจะไม่ทราบที่มาของ index term นั้น

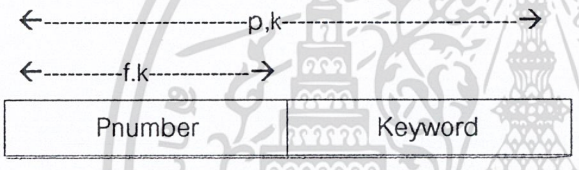
AUTHOR



ตาราง Author ทำหน้าที่เก็บข้อมูลผู้แต่งเนื่องจากเอกสารแต่ละฉบับ อาจมีผู้แต่งได้มากกว่า 1 ท่าน

- Pnuber หมายเลขกำกับเอกสารเพื่อใช้ในการค้นหา
- Authorname ชื่อผู้แต่ง

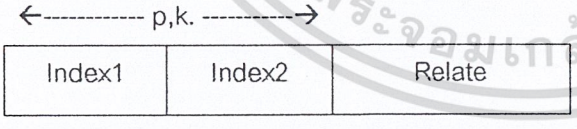
PAPERKEY



ตาราง PAPERKEY ใช้สำหรับเก็บ index term ของแต่ละเอกสารนั้น

- Pnuber หมายเลขกำกับเอกสารเพื่อใช้ในการค้นหา
- Keyword index term

RELATION



ตาราง RELATION ใช้สำหรับเก็บคู่ความสัมพันธ์ของ index term

- Index1,Index2 คู่ Index term
- Relate ค่าความสัมพันธ์มีค่าอยู่ระหว่าง 0-1

บทที่ 8
ผลการทดลองและการวัดประสิทธิภาพ

8.1 ผลการทดลอง

ผลการทดลองของระบบการค้นคืนสารสนเทศแบบฟัซซี่ลอจิก นี้สามารถอธิบายได้โดยการยกตัวอย่างในการประกอบ เพื่อระบุสิ่งที่ผู้ใช้ระบบต้องการค้นหา โดยตัวอย่าง ซึ่งมีสิ่งแวดล้อมต่าง ๆ ดังนี้

1. คำที่ผู้ใช้ต้องการค้นหา คือ คำว่า {dog,cat}
2. เอกสารที่มี {dog,cat} เป็น index term และ มีความเป็นสมาชิกเท่ากับ 1 ได้แก่

   d1 = {dog,cat,bird,zebra,zoo}

   d2 = {cat,kitty,fish}

   d3 = {dog,puppy,house,robber}

   d4 = {ant,sugar}
3. ความสัมพันธ์ของ index term แต่ละตัวที่มีต่อ indexเทอม “cat” , “dog”

   cat = {dog/0.33,bird/0.5,zebra/0.5,zoo/0.5,kitty/0.5,fish/0.5}

   dog = {cat/0.33,bird/0.5,zebra/0.5,zoo/0.5,puppy/0.5,house/0.5,robber/0.5}

เมื่อนำสิ่งแวดล้อมดังข้างบนมาคำนวณ โดยระบบค้นคืนสารสนเทศแบบฟัซซี่เป็นผู้คำนวณจะได้ค่าดังนี้

cat = {d1/1,d2/1,d3/0.33,d4/0}

dog = {d1/1,d2/0.33,d3/1,d4/0}

และเมื่อนำ index term 2 ตัวมาทำการอินเตอร์เซ็กกันจะได้ค่าดังนี้

$\mu_{cat \cap dog}$ = {d1/1,d2/0.33,d3/0.33,d4/0}

และเมื่อนำ index term 2 ตัวมายูเนียนกันจะได้ค่าดังนี้

$\mu_{cat \cap dog}$ = {d1/1,d2/1,d3/1,d4/0}

และเมื่อนำค่ามาทำคอมพรีเมนต์แล้วทำอินเตอร์เซ็กกันจะได้ค่าดังนี้

$\mu_{\bar{cat} \cap \bar{dog}}$ = {d1/0,d2/0,d3/0,d4/1}

ขั้นต่อมาก็นำข้อมูลมาเรียงจากมากไปหาน้อย เช่น ในการทำแบบ อินเตอร์เซ็ก จะได้ผลดังนี้

d1,d2,d3,d4

แล้วนำหมายเลขเอกสารไปค้นหาเอกสารในฐานข้อมูลมาแสดงให้แก่ผู้ใช้จะได้ดังนี้

d1 เรื่อง “life in zoo”

d2 เรื่อง “cat ‘s food”

d3 เรื่อง “The advantage of dog”

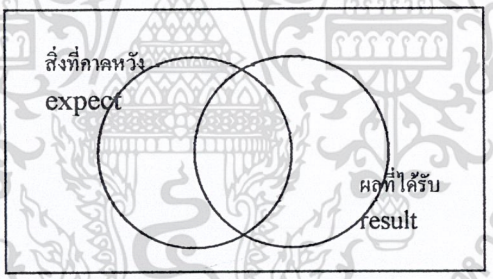
ส่วนเอกสาร d4 จะไม่ถูกแสดงออกมาเพราะไม่มีความเป็นสมาชิกเลย หรือ มีค่าเป็น 0

8.2 การวัดประสิทธิภาพ

การวัดประสิทธิภาพสามารถวัดได้หลายรูปแบบด้วยกันแต่ ระบบค้นคืนที่ดีนั้นต้องมี ประสิทธิภาพ 2 สิ่งนี้ที่สำคัญคือ

1. ประสิทธิภาพทางด้วยความถูกต้องและตรงตามความต้องการของผู้ค้นหา
2. ประสิทธิภาพทางด้านความเร็วในการค้นหา

8.2.1 ประสิทธิภาพทางด้วนความถูกต้องและตรงตามความต้องการของผู้ค้นหา



รูป 8.1 แสดงคำจำกัดความ ของ precision และ recall

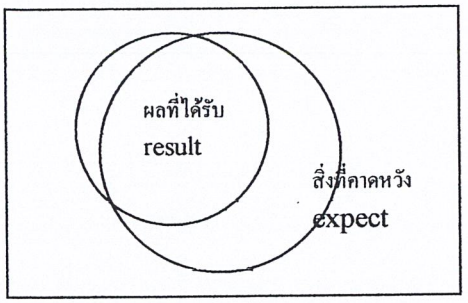
จะมีนิยามที่พูดถึงการวัดประสิทธิภาพด้านนี้ด้วยกัน 2 ค่าคือ

- ค่า Precision = $$\frac{|\,expect \cap result\,|}{|\,result\,|}$$

- ค่า Recall = $$\frac{|\,expect \cap result\,|}{|\,expect\,|}$$

และค่าในอุดมคติคือ Precision = Recall ซึ่งหมายถึง เอกสารทุก ๆ เอกสารที่เกิดจากการค้นคืนของระบบ เป็นสิ่งที่ผู้ใช้ต้องการ แต่ไม่มี เอกสารใดเลยที่ผู้ใช้ไม่ต้องการ ซึ่งในความเป็นจริงแล้วทำได้ยากมาก ๆ เพราะ ระบบนั้นต้องสามารถเรียนรู้ความต้องการของผู้ใช้ที่มีความแตกต่างกันออกไปได้และระบบจะต้องทำการเก็บข้อมูลการค้นของผู้ใช้แล้วนำค่าที่ได้ไปทำการเรียนรู้

ดังนั้นระบบที่ดีทั้ว ๆ ไปจะทำให้อยู่ในรูปแบบดังรูปนี้



รูป8.2 รูปประสิทธิภาพของการวัดความถูกต้องและตรงกับความต้องการ

จากรูปจะหมายถึง ระบบจะต้องพยายามทำให้ผลที่ได้รับ(result) ที่ออกมาให้ตรงกับสิ่งที่คาดหวัง(expect)ได้ทั้งหมดหรือ เกือบทั้งหมด เพื่อเพิ่มค่าความแม่นยำของระบบ

8.2.2 ประสิทธิภาพทางด้านความเร็วในการค้นหา

ประสิทธิภาพอีกเรื่องของระบบค้นคืนสารสนเทศคือเรื่องความเร็ว โดยเราต้องคำนึงถึงว่าในระบบค้นคืนสารสนเทศจริง ๆ แล้วนั้นเราไม่ได้มี เอกสารเพียง 3-4 เอกสารแต่จำนวนเอกสารจะมีจำนวนมากอย่างน้อย ต้อง 10000 เอกสารขึ้นไปเรื่องความเร็วจะมีผลอย่างยิ่งต่อระบบสารสนเทศนั้นว่าดีหรือไม่ ซึ่งในการแก้ปัญหาด้านความเร็วมีวิธีแก้ได้ 3 แบบด้วยกัน

1. แก้ที่ Hard ware ต้องที่เพียงพอต่อการเข้าถึงข้อมูลของผู้ใช้และช่วยให้การประมวลผลทำได้อย่างรวดเร็ว เช่นในระบบการค้นคืนที่ใหญ่ ๆ ยกตัวอย่างเช่น Google และ Altavista ซึ่งเป็นระบบค้นคืนสารสนเทศที่มีประสิทธิภาพมากและเป็นที่รู้จักกันดี ในการทำงานของ Google นั้น จะใช้โครงสร้างการค้นหาเป็นแบบ เว็คเตอร์ และ ระบบในการแบ่งหน้าที่ช่วยในการประมวลผล จะใช้แบบ clustering ส่วน Altavista นั้นจะมีระบบการประมวลผลคือการมี หลาย ๆ หน่วยประมวลผล(CPU) แบ่งแยกงานในการช่วยการประมวลผล แต่ในระบบสารสนเทศแบบฟัซซี่นี้ไม่อาจเพิ่มความเร็วโดยการ ใช้ Hard wareช่วยได้มากนัก เพราะปัจจัยหลายๆ เรื่องดังนั้นเรื่องการปรับประสิทธิภาพต้องใช้การปรับค่าอื่น
2. การปรับประสิทธิภาพโดยการปรับโครงสร้างของโปรแกรม เป็นการปรับค่าประสิทธิภาพที่เราสามารถกระทำได้โดยการใช้ อัลกอลิทึม ที่เหมาะสมและรวดเร็ว และจากการได้ปฏิบัติการเขียนโปรแกรมจริงแล้วนั้นปัญหาที่ทำให้ระบบช้ามีได้หลายกรณี แต่กรณีที่เห็นเด่นชัดคือการเขียน โปรแกรมติดต่อฐานข้อมูล ดังนั้นควรเขียนโปรแกรมให้มีการติดต่อกับฐานข้อมูลให้น้อยที่สุดเท่าที่ทำได้ หรือ การทำค่าในฐานข้อมูลมาเก็บในตัวแปรแทนการต้องติดต่อฐานข้อมูลทุกครั้ง ก็จะช่วยให้ระบบทำงานเร็วขึ้นได้
3. การปรับประสิทธิภาพโดยใช้ DBMS ที่มีประสิทธิภาพ ในการใช้งานจริง ๆ แล้วเราอาจแบ่งฐานข้อมูลออกเป็น 2 ชุดคือ ชุดที่ไว้สำหรับการอ่านอย่างเดียว และ ชุดที่ต้องมีการแก้ไขบ่อย

ๆ ในการกระทำเช่นนี้ ชุดที่ไว้สำหรับการอ่านอย่างเดียวจะเป็นฐานข้อมูลที่ถูก queryบ่อยครั้งแต่ รูปแบบการ query ส่วนใหญ่ จะเป็นรูปแบบการอ่าน ดังนั้นเราอาจ ทำการ denormalize เพื่อลดการ join กันของฐานข้อมูล เพื่อประสิทธิภาพด้านความเร็วในการค้นคืนมากขึ้น ส่วน ฐานข้อมูลที่มีการเปลี่ยนแปลงบ่อย ๆ บรรณารักษ์หรือผู้ดูแลระบบจะเป็นผู้ใช้งาน ดังนั้นเมื่อมีการเปลี่ยนแปลงข้อมูลจะกระทำต่อฐานข้อมูลชุดนี้เท่านั้นไม่ไปยุ่งกับ ฐานข้อมูลที่ใช้โดย ผู้ใช้ เพราะการเปลี่ยนแปลงค่า จะต้องทำการ lock record row ที่ทำการเปลี่ยนแปลง และ ผลิตภัณฑ์ ชั้นดี อย่างเช่น DB2 หรือ ORACLE สามารถตั้งเวลาของคำสั่งได้ดังนั้นการนำเอาข้อมูลจาก ฐานข้อมูลที่ บรรณารักษ์ใช้ไป update ฐานข้อมูลของผู้ใช้สามารถทำได้โดยง่าย ซึ่งในระบบค้นคืนสารสนเทศแบบฟัซซี่นี้ ใช้ DB2 สามารถตั้งเวลาของคำสั่งได้ โดยจะให้ทำงานตามคำสั่งในเวลาที่คิดว่ามีผู้ใช้ฐานข้อมูลน้อย เช่นในเวลาหลังเที่ยงคืนก็เป็นได้

บทที่ 9
บทสรุป

9.1 บทสรุป

ระบบค้นคืนสารสนเทศโดยใช้ฟัซซี่ลอจิก เป็นระบบที่เข้ามาช่วยเพิ่มประสิทธิภาพในการค้นคืนข้อมูลซึ่งข้อดีที่เห็นเด่นชัดมากที่สุดคือ การค้นหาเอกสารที่มีความคล้ายกัน โดยหลักการของ ฟัซซี่ เซตจะช่วยในการอธิบายความแตกต่าง และ ความเหมือนได้ โดยจากการทดลองใช้งานจริงแล้วนั้น ระบบที่จะใช้การค้นคืนสารสนเทศโดยใช้ฟัซซี่ลอจิกได้ดีนั้น จะมีปัจจัยที่สำคัญอยู่ 2 ปัจจัยคือ

1. จำนวนเอกสารที่ถูกจัดเก็บภายในระบบ
2. ความใกล้เคียงกันของเอกสารในแต่ละฉบับ

ซึ่งสามารถอธิบายได้ คือ จำนวนเอกสารที่ถูกจัดเก็บภายในระบบ นั้นต้องมีจำนวนที่มากถึงระดับนึงเพื่อใช้เป็นเครื่องมือในการสร้างความสัมพันธ์ของ index term ได้ โดยถ้าเอกสารมีจำนวนมากแล้ว ความหลากหลายของ index term ก็จะมากขึ้นด้วยดังนั้นค่าความสัมพันธ์ของอินเด็กซ์ จะมีความถูกต้องต่อการใช้งานจริงมากขึ้น และอีกปัจจัยหนึ่ง คือ ความใกล้เคียงกันของเอกสาร หมายถึง ถ้าเอกสารมีความใกล้เคียงกันมากแล้ว ค่าความสัมพันธ์ของตัว index term จะถูกพัฒนาได้รวดเร็ว กล่าวโดยสรุปคือ ระบบค้นคืนสารสนเทศโดยใช้ฟัซซี่ลอจิกจะมีประสิทธิภาพดีหรือไม่นั้นขึ้นอยู่กับปัจจัย 2 ปัจจัยข้างต้นเป็นหลัก

ดังนั้นระบบค้นคืนสารสนเทศโดยใช้ฟัซซี่ลอจิกควรถูกพัฒนาใน การจัดการกับเอกสารที่มีจำนวนมาก ๆ หรือ ถูกนำไปใช้งานในส่วนการค้นหาเฉพาะด้าน

9.2 แนวทางการพัฒนาต่อ

1. สามารถค้นหาแบบ phrase ได้ยกตัวอย่างเช่น ค้นหาคำว่า "information retrieval" ซึ่งต้องเป็น index term ตัวเดียวกัน

2. กำหนดน้ำหนักของแต่ละ index term ในเอกสารได้ หมายความว่า ในแต่ละเอกสาร อาจจะมี index term ที่เหมือนกันแต่ น้ำหนักของ index term ในเอกสารไม่เท่ากัน เพื่อบ่งบอกว่า เอกสารฉบับไหน ให้ความสำคัญกับ index term นั้นมากกว่ากัน ประโยชน์ที่ได้รับจากแนวทางนี้คือการค้นคืนเอกสารจะทำให้มีประสิทธิภาพมากยิ่งขึ้น

3. การนำเอาหลักการฟัซซี่ไปใช้เก็บข้อมูลเช่นรูปภาพเพื่อใช้ในการค้นคืน ยกตัวอย่างเช่น ผู้ใช้อยากมีความต้องการได้รูปที่ "รูปที่มีสีแดง และ เป็นรูปเหลี่ยม ๆ" หลักการฟัซซี่จะสามารถอธิบายขั้นตอนการค้นคืนลักษณะเช่นนี้ได้

ภาคผนวก ก

จาวาเซิร์ฟเล็ต(Java Servlet)

เซิร์ผเล็ตมีการทำงานแบบซีจีไอ โดยทำงานบนจาวาเวอร์ชวลแมชชีน(JVM) บนเซิร์ฟเวอร์ ซึ่งต่างจากวาจาแอปเพล็ต คือ บราวเซอร์ไม่จำเป็นต้องสนับสนุนการใช้งาน และต่างจากซีจีไอตรงที่ซีจีไอจะใช้ Multiple-process เพื่อจัดการกับโปรแกรมหลาย ๆ โปรแกรมและรีเควสหลาย ๆ ครั้ง ส่วนเซิร์ฟเล็ตจะจัดการโดยแยกเป็นหลาย ๆ เธรด ในการประมวลผลของเว็บเซิร์ฟเวอร์

7.1 ความสามารถของเซิร์ฟเล็ต

- สามารถทำงานข้ามแพลตฟอร์มได้ เซิร์ฟเล็ตเป็นภาษาจาวา ซึ่งมีความสามารถนี้ ดังนั้นเราอาจจะพัฒนาเซิร์ฟเล็ตบนวินโดว์ แล้วนำโปรแกรมไปใช้บนยูนิกซ์ก็ได้
- ความสามารถของจาวาเซิร์ฟเล็ตจะสามารถใส่ความสามารถของจาวาได้อย่างเต็มที่ เช่นการเข้าถึงระบบเน็ตเวิร์ก และ URL มัลติเธรด การเชื่อมต่อกับฐานข้อมูล การใช้ Remote Method Invocation(RMI)
- ความมีประสิทธิภาพ เซิร์ฟเล็ตจะทำงานได้อย่างมีประสิทธิภาพ เมื่อโหลดเซิร์ฟเล็ตขึ้นมาในหน่วยความจำในลักษณะของ single object instance เซิร์ฟเล็ตจะจัดการร้องขอพร้อมๆ กันได้โดยใช้เธรด และสามารถคงสถานะเดิมของมันได้ เช่น การเชื่อมต่อกับฐานข้อมูล เพราะตัวของเซิร์ฟเล็ตเองจะอยู่ในหน่วยความจำของเซิร์ฟเวอร์
- ความปลอดภัย ซึ่งจะมีการรักษาความปลอดภัยที่เป็นคุณสมบัติที่สืบทอดมาจากคุณลักษณะของจาวา และยังมีคุณสมบัติในตัวมันเองของ API ด้วย คือมีพอยเตอร์ในการดูแลหน่วยความจำ และมีการรองรับความผิดพลาดต่าง ๆ ที่อาจเกิดขึ้นโดยใช้ Exception-handing mechanism ซึ่งจะ throw exception โดยไม่ทำให้ระบบเสียหาย
- ความสะดวกในการพัฒนา สามารถเรียก Servlet API มาใช้ได้ โดยมันจะประกอบไปด้วยคลาสต่าง ๆ ที่ช่วยในการทำงานของเซิร์ฟเล็ต ทำให้พัฒนาโปรแกรมได้ง่ายขึ้น
- ความยืดหยุ่น Servlet API ถูกออกแบบมาเพื่อรองรับการขยายงานในรูปแบบต่าง ๆ ได้ง่าย เช่นในปัจจุบันนี้ มีการพัฒนาเพื่อรองรับการทำงานสำหรับ HTTP Servlet

7.2 HTTP Servlet

พื้นฐานของ Http Requests, Response และ Headers นั้น การทำงานของโพรโตคอล Http นั้นฝั่งไคลเอนท์ (ในที่นี้คือเว็บบราวเซอร์) จะทำการรีเควส และ เว็บเซิร์ฟเวอร์ก็จะตอบสนองกลับมา ในการส่งรีเควสของไคลเอนท์ สิ่งแรกที่ต้องระบุถึงคือ Http command ซึ่งเป็นเมธอดที่จะบอกเซิร์ฟเวอร์ถึง

แอคชันที่มันจะทำ บรรทัดแรกของการรีเควสจะกำหนดแอดเดรส และเวอร์ชันของโปรโตคอล Http ดังตัวอย่าง

Get/intro.html HTTP/1.0

ในการรีเควสนี้จะใช้เมธอด GET เพื่อร้องขอ intro.html โดยใช้ Http หลังจากรีเควสแล้วไคลเอ็นท์อาจส่งข้อมูลอื่น ๆ เช่น ซอฟท์แวร์ที่ไคลเอนท์ทำงานอยู่ ดังตัวอย่าง

User-Agent:Mazilla/4.0 (compatible;MSIE4.0;Windows95)

หลังจากได้รับรีเควสของไคลเอนท์แล้ว เซิร์ฟเวอร์จำทำการประมวลผลและส่งผลตอบสนองกลับไป บรรทัดแรกของการตอบสนองจะแสดงสถานะ (status line) ที่ระบุเวอร์ชันของโปรโตคอล HTTP เช่น HTTP/1.0 200 OK

หลังจากได้รับ status line แล้ว เซิร์ฟเวอร์จะส่งผลตอบสนองเพื่อบอกรายละเอียดเกี่ยวกับตัวมัน ดังตัวอย่าง

Date	: Saturday,23-May-98 03:25:12 GMT
Server	: JavaWebServer/1.1.1
MIME-version	: 1.0
Content-type	: text/html
Content-length	: 1029
Last-modified	: Thursday, 7-May-98 12:15:35 GMT

7.3 GET & POST METHOD

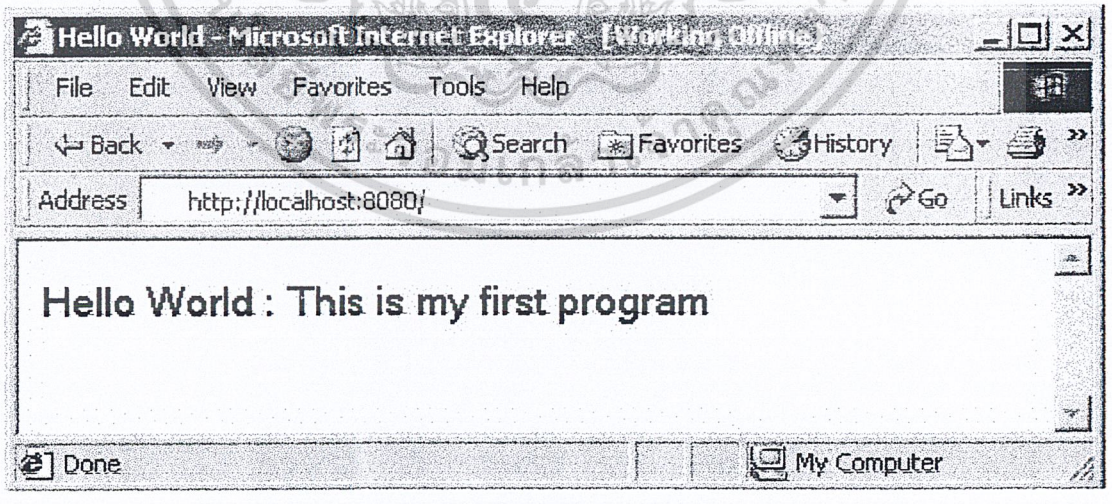
เมื่อไคลเอนท์ติดต่อกับเซิร์ฟเวอร์และทำ HTTP request รูปแบบของการร้องขอมีหลายรูปแบบที่มักจะใช้อยู่เป็นประจำคือเมธอด GET และ เมธอด POST

เซิร์ฟเล็ตจะใช้คลาสและอินเตอร์เฟสจาก 2 แพ็คเก็จ คือ javax.servler.http การเขียนเซิร์ฟเล็ตต้องอิมพลีเมนท์อินเตอร์เฟส javax.servlet.Service หรือ javax.servlet.http.HttpServlet ในแต่ละครั้งที่เซิร์ฟเวอร์ส่งผ่านการร้องขอ (request) ไปยังเซิร์ฟเล็ตมันจะไปเรียกเมธอด service() ของเซิร์ฟเวอร์

พื้นฐานอย่างหนึ่งของ HTTP เซิร์ฟเล็ต คือการแสดงหน้าเว็บเพจ HTML ซึ่งมันจะสามารถทำงานได้เหมือนกับซีจีไอทั้งด้านทำฟอร์ม HTML หรือการติดต่อกับฐานข้อมูล คือตัวอย่างจะเป็นการใช้ เซิร์ฟเล็ตเขียนโปรแกรมให้แสดงออกมาในรูปแบบของ HTML

ตัวอย่างโปรแกรม Hello World

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class Hello World extends HttpServlet
{
        public void doGet(HttpServlet req,HttpServletResponse res)
        throws ServletException,IOException
        {
                res.setContentype("text/html");
                PrintWriter out = res.getWriter();
                out.println("<HTML>");
                out.println("<HEAD><TITLE>Hello Word</TITLE></HEAD>");
                out.println("<BODY>");
                out.println("<BIG>Hello World : This is my first program</BIG>");
                out.println("</BODY></HTML>");
        }
}
```



รูปที่ ก-1 รูปแสดงการใช้เซิร์ฟเล็ตในการสร้างหน้าจอออกมา

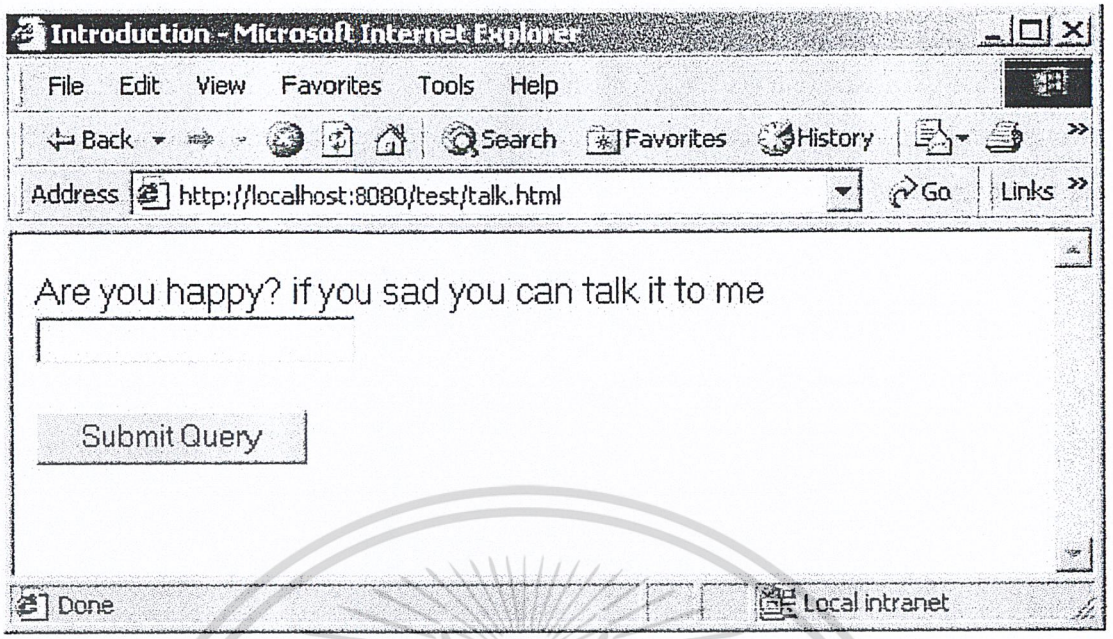
เซิร์ฟเล็ตจะเอ็กซ์เทนด์คลาส HttpServlet และโอเวอร์โหลดเมธอด doget() แต่ละครั้งที่เซิร์ฟเวอร์ได้รับรีเควสแบบ GET แล้วเซิร์ฟเวอร์จะเรียกเมธอด doget() เพื่อผ่านค่าออบเจ็ค HttpServletRequest และ HttpServletResponse

HttpServletRequest จะรอบรับการร้องขอต่าง ๆ ของไคลเอ็นท์โดยออบเจ็คนี้จะเข้าถึงข้อมูลเกี่ยวกับไคลเอนท์และยังสามารถใช้ออบเจ็คเพื่อการเซ็ต Httpresponse header ด้วย

เมธอดที่ตัวอย่างโปรแกรมใช้ตอนแรก คือเมธอด setContent Type() ของออบเจ็ค response เพื่อดึงค่าของ content type ของการตอบสนองที่เป็น "text/html" และใช้เมธอด getWriter() เพื่อเก็บค่าของ PrintWriter ในการที่จะส่ง "Hello World" ในรูปแบบของ HTML ไปยังไคลเอนท์

เซิร์ฟเล็ตจะสามารถทำการสร้าง (generate) รูปแบบตามภาษา HTML ได้ โดยจะมีค่าดังต่อไปนี้

```
<HTML>
        <HEAD>
                <TITLE>Introduction</TITLE>
        </HEAD>
        <BODY>
                <FORM METHOD = "GET" ACTION = "/servlet/pray_to_god">
                        Are you happy? if you sad you can talk it to me
                        <INPUT TYPE = "text" name ="sad_word"><p>
                        <INPUT TYPE = "SUBMIT">
                </FORM>
        </BODY>
</HTML>
```



รูปที่ ก-2 แสดงการใช้เซิร์ฟเล็ตในการสร้าง form

จากแบบฟอร์มนี้เมื่อผู้ใช้ทำการกรอก สิ่งที่เป็นทุกข์อยู่ ลงในเท็กซ์ฟิลด์แล้ว ทำการกดปุ่ม Submit Query แล้วจะเกิด action ขึ้น คือ ข้อความนั้น จะถูกส่งไปที่ pray_to_god Servlet และจะไปทำการเรียก method GET ซึ่งข้อมูลที่ถูกส่งไปจะส่งไปในรูปแบบต่อไปนี้

http://server:880/servlet/pray_to_god?sad_word=key

ตัวอย่างโปรแกรม pray_to_god

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class Pray_to_god extends HttpServlet
{
        public void doGet(HttpServletRequest req,HttpServletResponse res)
        throws ServletException,IOException
        {
                res.setContenType("text/html");
                printlnWriter out.res.getWriter();
                String pray = req.getParameter("sad_word");
```

```
                    out.println("<HTML>");

                    out.println("<HEAD><TITLE>++PRAYTOGOD++</TITLE></
                    HEAD>");
                    out.println("<BODY>");
                    out.println(pray+"will get out of you .I sure");
                    out.println("</BODY><HTML>");
            }
}
```

เมธอด req.getParameter("sad_word")จะใช้เป็นตัวแปรรับค่า sad_word ของผู้ใช้ตัวอย่างเมธอด POST จะมีรูปแบบดังนี้

```
public void doPost(HttpServletRequest req,HttpservletResponse res)
throws ServletException,IOException
{
}
```

โดยในฟอร์มของการส่ง HTML นั้นจะอยู่ในรูปแบบ

```
<FORM METHOD = "GET" ACTION = "/servlet/pray_to_god">
```

ภาคผนวก ข.

Porter's Algorithm

Porter's Algorithm เป็น Algorithm ที่ใช้ในการกำจัด Suffix ออกจากคำ เพื่อให้ได้รากศัพท์เดิม ซึ่งถูกเสนอโดย M.F. Porter ในปี ค.ศ. 1980

ข้อกำหนดเบื้องต้น (Definition) :

- Consonant แทนตัวอักษรที่ไม่ใช่สระในภาษาอังกฤษ คือตัวอักษรทุกตัวยกเว้น A, E, I, O, U
- Vowel แทนสระในภาษาอังกฤษ คือ A, E, I, O, U
- กรณี Y อาจเป็นได้ทั้ง Consonant และ Vowel ขึ้นอยู่กับตำแหน่งที่ปรากฏในคำ เช่น

  TOY Y ทำหน้าที่เป็น Consonant , WYVERN Y ทำหน้าที่เป็น Vowel

  ซึ่งการที่จะระบุว่า Y ทำหน้าที่ใดในคำหนึ่งๆ อาจใช้วิธีพิจารณาจากอักษรก่อนหน้าว่าเป็น Consonant หรือ Vowel โดย Y จะทำหน้าที่ตรงกันข้าม
- c แทน Consonant 1 ตัว ที่ปรากฏในคำ
- v แทน Vowel 1 ตัว ที่ปรากฏในคำ
- C แทน กลุ่มของ c ที่มีความยาวมากกว่า 0 ขึ้นไป
- V แทน กลุ่มของ v ที่มีความยาวมากกว่า 0 ขึ้นไป
- () ใช้แบ่งกลุ่มตัวอักษร
- [ ] ใช้แสดงกลุ่มตัวอักษรที่อาจจะมีหรือไม่ก็ได้
- {m} เมื่อ m แทนตัวเลขใดๆ หมายถึง การปรากฏรูปแบบนั้นๆ ซ้ำเป็นจำนวน m ครั้ง
- กฎต่างๆ จะเขียนอยู่ในรูป (condition) S1 -> S2 โดยหาก condition เป็นจริง จะแทนที่ส่วนหลังของคำที่ตรงกับ S1 ด้วย S2 (S2 อาจเป็น NULL)
- M ใช้แทนรูปแบบ [C](VC){m}[V] เมื่อ M มีค่าเท่ากับ m
- *S แทนคำที่ลงท้ายด้วย 'S'
- *v* แทนคำที่มี Vowel อยู่ภายในคำ
- *d แทนคำที่ลงท้ายด้วย Consonant ตัวเดียวกัน 1 คู่ (เช่น –EE, -SS)
- *o แทนคำที่ลงท้ายด้วย cvc ซึ่ง c ตัวที่ 2 ไม่ใช่ตัวอักษร 'W', 'X', 'Y'
- condition อาจประกอบด้วย and, or หรือ not และอาจเป็น NULL ได้
- จะไม่นำ Case ของตัวอักษรมาพิจารณา ('A' = 'a')

Porter's Algorithm แบ่งออกเป็น 5 ขั้นตอน ซึ่งแต่ละขั้นตอนอาจประกอบด้วยขั้นตอนย่อยได้ โดยในแต่ละ condition จะแสดงตัวอย่างของคำที่ตรงกับเงื่อนไขนั้นๆ และรูปของคำที่เปลี่ยนไปด้วย เพื่อจะได้เข้าใจขั้นตอนในการทำงานแต่ละขั้นได้อย่างชัดเจน

Porter's Algorithm

Step 1a

SSES -> SS	caresses -> caress
IES -> I	ponies -> poni
	ties -> ti
SS -> SS	caress -> caress
S ->	cats -> cat

Step 1b

	(M>0) EED -> EE	feed -> feed
		agreed -> agree
+	(*v*) ED ->	plastered -> plaster
		bled -> bled
+	(*v*) ING ->	motoring -> motor
		sing -> sing
	ถ้ากรณีที่ + เป็นจริง {	
	AT -> ATE	conflat(ed) -> conflate
	BL -> BLE	troubl(ed) -> trouble
	IZ -> IZE	siz(ed) -> size
	(*d and not (*L or *S or *Z)) double letters -> single letter	hopp(ing) -> hop
		tann(ed) -> tan
		fall(ing) -> fall
		hiss(ing) -> hiss
		fizz(ed) -> fizz
	(m=1 and *o) -> E	fail(ing) -> fail
		fil(ing) -> file
	}	

Step 1c

(*v*) Y -> I	happy -> happi
	sky -> sky

Step 2

(m>0) ATIONAL -> ATE	relational -> relate
(m>0) TIONAL -> TION	conditional -> condition
	rational -> rational
(m>0) ENCI -> ENCE	valenci -> valence
(m>0) ANCI -> ANCE	hesitanci -> hesitance
(m>0) IZER -> IZE	digitizer -> digitize
(m>0) ABLI -> ABLE	conformabli -> conformable
(m>0) ALLI -> AL	radicalli -> radical
(m>0) ENTLI -> ENT	differentli -> different
(m>0) ELI -> E	vileli -> vile
(m>0) OUSLI -> OUS	analogousli -> analogous
(m>0) IZATION -> IZE	vietnamization -> vietnamize
(m>0) ATION -> ATE	predication -> predicate
(m>0) ATOR -> ATE	operator -> operate
(m>0) ALISM -> AL	feudalism -> feudal
(m>0) IVENESS -> IVE	decisiveness -> decisive
(m>0) FULNESS -> FUL	hopefulness -> hopeful
(m>0) OUSNESS -> OUS	callousness -> callous
(m>0) ALITI -> AL	formaliti -> formal
(m>0) IVITI -> IVE	sensitiviti -> sensitive
(m>0) BILITI -> BLE	sensibiliti -> sensible

Step 3

(m>0) ICATE -> IC	triplicate -> triplic
(m>0) ATIVE ->	formative -> form
(m>0) ALIZE -> AL	formalize -> formal
(m>0) ICITI -> IC	electriciti -> electric
(m>0) ICAL -> IC	electrical -> electric
(m>0) FUL ->	hopeful -> hope
(m>0) NESS ->	goodness -> good

Step 4

(m>1) AL ->	revival -> reviv
(m>1) ANCE ->	allowance -> allow
(m>1) ENCE ->	inference -> infer
(m>1) ER ->	airliner -> airlin
(m>1) IC ->	gyroscopic -> gyroscop
(m>1) ABLE ->	adjustable -> adjust
(m>1) IBLE ->	defensible -> defens
(m>1) ANT ->	irritant -> irrit
(m>1) EMENT ->	replacement -> replac
(m>1) MENT ->	adjustment -> adjust
(m>1) ENT ->	dependent -> depend
(m>1 and (*S or *T)) ION ->	adoption -> adopt
(m>1) OU ->	homologou -> homolog
(m>1) ISM ->	communism -> commun
(m>1) ATE ->	activate -> activ
(m>1) ITI ->	angulariti -> angular
(m>1) OUS ->	homologous -> homolog
(m>1) IVE ->	effective -> effect
(m>1) IZE ->	bowdlerize -> bowdler

Step 5a

(m>1) E ->	probate -> probat
	rate -> rate
(m=1 and not *o) E ->	cease -> ceas

Step 5b

(m > 1 and *d and *L) double letters -> single letter	controll -> control
	roll -> roll

ภาคผนวก ค.

Stopwords list

Stopwords ที่ใช้ในโครงงานนี้ มี 319 คำ ได้แก่

a	about	above	across	after
afterwards	again	against	all	almost
alone	along	already	also	although
always	am	among	amongst	amoungst
amount	an	and	another	any
anyhow	anyone	anything	anyway	anywhere
are	around	as	at	back
be	became	because	become	becomes
becoming	been	before	beforehand	behind
being	below	beside	besides	between
beyond	bill	both	bottom	but
by	call	can	cannot	cant
co	computer	con	could	couldnt
cry	de	describe	detail	do
done	down	due	during	each
eg	eight	either	eleven	else
elsewhere	empty	enough	etc	even
ever	every	everyone	everything	everywhere
except	few	fifteen	fify	fill
find	fire	first	five	for
former	formerly	forty	found	four
from	front	full	further	get
give	go	had	has	hasnt
have	he	hence	her	here
hereafter	hereby	herein	hereupon	hers
herself	him	himself	his	how
however	hundred	i	ie	if
in	inc	indeed	interest	into

is	it	its	itself	keep
last	latter	latterly	least	less
ltd	made	many	may	me
meanwhile	might	mill	mine	more
moreover	most	mostly	move	much
must	my	myself	name	namely
neither	never	nevertheless	next	nine
no	nobody	none	noone	nor
not	nothing	now	nowhere	of
off	often	on	once	one
only	onto	or	other	others
otherwise	our	ours	ourselves	out
over	own	part	per	perhaps
please	put	rather	re	same
see	seem	seemed	seeming	seems
serious	several	she	should	show
side	since	sincere	six	sixty
so	some	somehow	someone	something
sometime	sometimes	somewhere	still	such
system	take	ten	than	that
the	their	them	themselves	then
thence	there	thereafter	thereby	therefore
therein	thereupon	these	they	thick
thin	third	this	those	though
three	through	throughout	thru	thus
to	together	too	top	toward
towards	twelve	twenty	two	un
under	until	up	upon	us
very	via	was	we	well
were	what	whatever	when	whence
whenever	where	whereafter	whereas	whereby
wherein	whereupon	wherever	whether	which
while	whither	who	whoever	whole

whom	whose	why	will	with
within	without	would	yet	you
your	yours	yourself	yourselve	

ภาคผนวก ง

ภาษาเอสคิวแอล(SQL) (SQL : Structured Query Language)

ภาษาเอสคิวแอล(SQL)เป็นภาษายุคที่ 4 ภาษาหนึ่งซึ่งพัฒนาโดยบรัษัทไอบีเอ็ม (IBM) ในปัจจุบันเป็นที่แพร่หลายกันมาก เป็นภาษาที่คล้ายกับภาษาอังกฤษ ใช้ในการปฏิบัติงานและควบคุมฐานข้อมูลเป็นภาษาที่คล้ายกับภาษาอังกฤษ ในการประมวลผลข้อมูลหรือมีเพียงเล็กน้อย ก็สามารถที่จะเรียนรู้โครงสร้างพื้นฐานของภาษาเอสคิวแอล(SQL)ได้อย่างรวดเร็ว และสำหรับผู้ที่อยู่ในระดับที่มีความรู้ด้านนี้สูง ก็จะพบว่าพาษาเอสคิวแอล(SQL)นั้น จะให้คำสั่งซึ่งมีความสามารถ และมีความสมบูรณ์ในตัวในการดำเนินงานได้อย่างดี

ดังนั้นภาษาเอสคิวแอล(SQL)จึงกลายเป็นภาษาร่วมกันระหว่างผู้ใช้งานธรรมดา กับผู้ที่มีความรู้ประสบการณ์ในการประมวลผล แต่สำหรับผู้ใช้ทั่วไปแล้ว คงเป็นการลำบากที่จะใช้ภาษาเอสคิวแอล(SQL) ในการสร้างคำถามที่ซับซ้อนได้ ดังนั้น ตามความเป็นจริงแล้ว ในปัจจุบันภาษาเอสคิวแอล(SQL)ใช้โดยบุคคลในวงการระดับที่มีความรู้และประสบการณ์ในการประมวลผล , นักพัฒนาระบบงาน, นักบริหารข้อมูล (DBA : Database Administrator), ระดับผู้บริหาร และทีมงานนักสารนิเทศ

ภาษาเอสคิวแอล(SQL)เป็นภาษาฐานข้อมูลแบบเชิงสัมพันธ์ (Relational Database) คือประกอบด้วยตาราง (Table) หลายตาราง และในตารางหนึ่ง ๆ จะมี 2 มิติได้แก่ หลัก (columns) ในแนวตั้งและแถว (rows) ในแนวนอน

การใช้ภาษาเอสคิวแอล(SQL)กระทำได้ 3 วิธี ได้แก่

1. ออกคำสั่งแบบออนไลน์ กล่าวคือ ผู้ใช้สามารถพิมพ์ประโยคคำสั่งผ่านทางเทอร์มินัล (Terminal) โดยที่คำ สั่งเหล่านี้จะถูกปฏิบัติงานโดยทันที
2. ส่งคำสั่งในลักษณะงานออฟไลน์ (Off line) หรืองาน batch ลักษณะการใช้งานประเภทนี้เหมาะกับการสร้างรายงาน หรือประเภทของงานที่ไม่จำเป็นต้องการทราบผลโดยทันที
3. สอดแทรกประโยคคำสั่งไว้ในโปรแกรมประยุกต์ที่เขียนขึ้นมาสำหรับการใช้ระบบฐานข้อมูล ซึ่งโปรแกรมประยุกต์เหล่านี้อาจจะเขียนด้วยภาษา โคบอล, ฟอร์แทรน, ภาษาซี, ภาษาจาวา ฯลฯ ก็ได้

ภาษาเอสคิวแอล(SQL)มีประเภทคำสั่งโดยสรุปดังนี้

- การคิวรี (Query) เป็นการสอบถามข้อมูลจากฐานข้อมูล
- การดำเนินงานกับข้อมูล (Data Manipulation) ได้แก่ การเพิ่มเติม (insert), การลบ (delete) และการแก้ไข (update) ข้อมูลในฐานข้อมูล
- การกำหนดลักษณะของข้อมูล (Data Definition) ได้แก่ การกำหนดตาราง (tables), วิว (views) และดัชนีในการค้นหา (indexs) ในฐานข้อมูล
- การควบคุมข้อมูล (Data Control) ได้แก่ การป้องกัน ควบคุมข้อมูลให้ปลอดภัย จากผู้ใช้แต่ละคน

รูปแบบของคำสั่งเอสคิวแอล(SQL)พื้นฐานโดยทั่วไป

Select ....

From (ชื่อตาราง)

Where(เงื่อนไข)

Group by(จัดกลุ่ม)

Having

Order by.....

ซึ่งผู้ใช้สามารถระบุสิ่งต่าง ๆ ในคำสั่งได้ดังนี้

- ส่วนหลัง Select ใช้ระบุคอลัมน์หรือกลุ่มขอคอลัมน์ที่เราต้องการดูข้อมูล
- ส่วนหลัง From ใช้ระบุชื่อของตารางที่เราต้องการดูข้อมูล
- ส่วนหลัง Where ใช้ระบุเงื่อนไขของข้อมูลที่เราสนใจซึ่งอาจจะมีหรือไม่มีก็ได้
- ส่วนหลัง Group by ใช้กำหนดกลุ่มของข้อมูลที่เราสนใจ จะมีหรือไม่มีก็ได้
- ส่วนหลัง Having ใช้กำหนด ฟังชัน ที่ใช้ในการค้นหา จะมีหรือไม่มีก็ได้
- ส่วนหลัง Order by ใช้ระบุวิธีการเรียงลำดับการแสดงผลข้อมูลซึ่งอาจจะมีหรือไม่มีก็ได้

ตัวอย่างง่าย ๆ ของภาษาเอสคิวแอล(SQL) เช่น

Select * from Paper

จากคำสั่งนี้จะทำการแสดงข้อมูลของทุก rows และ ทุก columns ที่มีอยู่ในตารางชื่อ Paper

บรรณานุกรม

[1] Ricardo Baeza-Yates and Berthier Ribeiro-Neto(1999) : *Modern Information Retrieval* , Addison-Wesley

[2] Stuart Russell and Peter Norvig : *Artificial Intelligence a Modern Approach* , Prentice Hall

[3] Timothy J. Ross : *Fuzzy Logic with Engineering Applications* , McGraw-Hill, Inc

[4] Salton, G., and M.J. McGill. *Introduction to Medern Information Retrieval* .New York: International Book Company, McGraw-Hill, 1984

[5] Radecki,T. "Fuzzy Set Theoretical Approach to Document Retrieval." *Information Processing & Management*, Vol. 15 (1979)