บทที่ 5
กลยุทธ์การค้นหาข้อมูล

วัตถุประสงค์

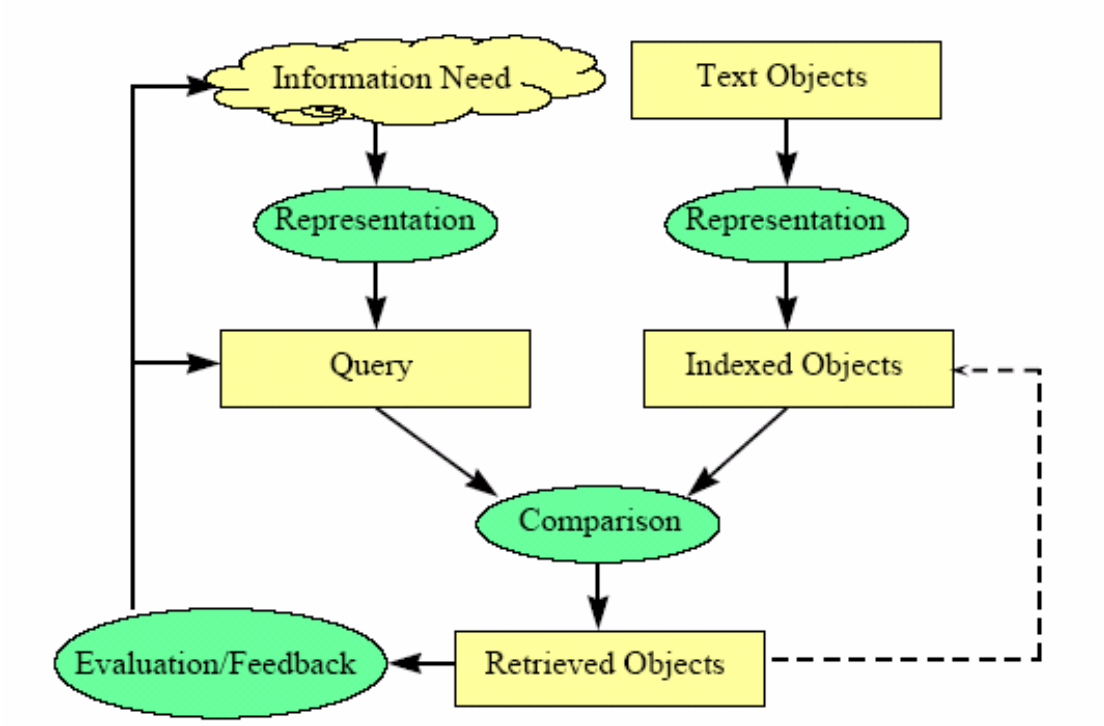
1. เพื่อให้นักศึกษาทราบกลวิธีการค้นหาข้อมูล
2. เพื่อให้นักศึกษาทราบBoolean Search ,Serial Search ,Cluster-Based Retrieval
3. เพื่อให้นักศึกษาทราบถึงความแตกต่างระหว่าง Matching Function และ Association Measure อย่างไร
4. เพื่อให้นักศึกษาทราบถึงวิธีหาตัวแทนคลัสสเตอร์ ในหลายๆวิธี
5. เพื่อให้นักศึกษาทราบถึงความหมายของFeedback ประโยชน์ และผลที่ได้

สารบัญ

สำหรับในบทนี้จะกล่าวถึงกลยุทธ์ในการค้นหาข้อมูล วิธีการค้นหาข้อมูลในระบบค้นคืนสารสนเทศ(IR) นั้น มักจะสนใจเพียงแต่ว่า เอกสารที่ถูกดึงออกมานั้นเกี่ยวข้อง (Relevant) กับข้อคำถาม (Request) มากน้อยเพียงไร ทุกๆ กลวิธีในการค้นหาได้อยู่บนรากฐานของการเปรียบเทียบระหว่างข้อคำถามกับเอกสารที่ถูกเก็บ บางครั้งการเปรียบเทียบนี้สามารถกระทำได้โดยอ้อม เมื่อข้อคำถามถูกเปรียบเทียบกับคลัสเตอร์ หรือถ้าจะกล่าวให้ถูกต้อง ก็คือ นำข้อคำถามมาเปรียบเทียบกับตัวแทนคลัสเตอร์นั่นเอง

ข้อแตกต่างระหว่างกลวิธีในการค้นหาข้อมูลชนิดต่างๆ บางครั้ง อาจจะดูได้จากภาษาสอบถาม (Query language) ซึ่งเป็นข้อความที่ผู้ใช้ต้องการเป็นภาษาธรรมชาติ ธรรมชาติของภาษาคำถามมักจะเป็นข้อมูลในการกำหนดกลยุทธ์ในการค้นหาข้อมูล

อย่างเช่น ภาษาสอบถามหนึ่ง เป็นคำสั่งในการค้นหาข้อมูลที่ต้องการ (Search Statements) แสดงอยู่ในรูปแบบทางด้านตรรกศาสตร์(Logical Combinations) ของคีย์เวิร์ดมีการใช้ตัวกระทำทางด้านตรรกเช่น AND หรือ OR หรือ NOT โดยปกติจะใช้ในการค้นหาที่เป็น Boolean Search ผลลัพธ์จะเกิดจากการเปรียบเทียบ(Comparison)ทางตรรกะระหว่างข้อคำถามกับเอกสาร ซึ่งผลของการเปรียบเทียบได้ผลลัพธ์เป็น TRUE หรือ FALSE อย่างใดอย่างหนึ่ง ดังรูปที่ 5.1



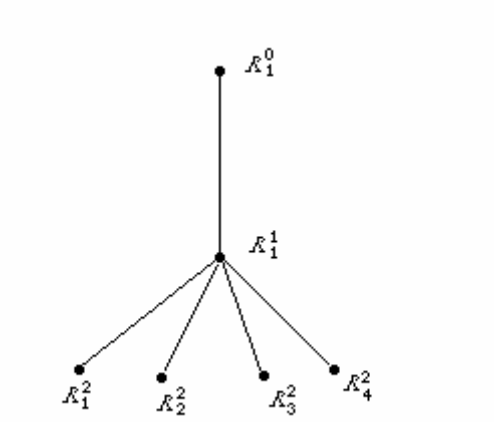
รูปที่ 5.1 : Simple model of IR

5.1 Boolean Search

กลยุทธ์ Boolean Search เป็นการค้นหาที่นิยมใช้มากโดยการสืบค้นจะดึงเอกสารที่ผลจากการเปรียบเทียบทางตรรกะระหว่างข้อคำถามกับเอกสารมีค่าเป็นจริง (True) เท่านั้น ซึ่งข้อคำถามที่ดีจะถูกแสดงในรูปแบบของคำดัชนี และถูกเชื่อมโยงด้วยตัวกระทำทางตรรกะ AND,OR และ NOT

สมมุติว่าข้อคำถาม Q = (K1 AND K2) OR (K3 AND K4) กลยุทธ์ Boolean Search จะดึงทุกๆ เอกสารที่ให้ผลลัพธ์เป็น True

บางระบบที่ใช้วิธีการค้นแบบ Boolean Search มีการกำหนดขอบเขตในการค้นหาของผู้ใช้ให้อยู่ในวงที่กำหนด โดยกำหนดให้ผู้ใช้เข้าถึงข้อมูลเฉพาะที่มีการกำหนดให้เท่านั้น ซึ่งคีย์เวิร์ดที่เกี่ยวข้องซึ่งอาจจะต้องการให้สืบค้นข้อมูลที่มีความหมายกว้างกว่า หรือสืบค้นข้อมูลที่เจาะจงมากกว่าก็ได้ ตามตัวอย่างในโครงสร้างของต้นไม้(tree) ดังรูปที่ 5.2



รูปที่ 5.2 : แสดงลำดับชั้นของคีย์เวิร์ด

จากรูปที่ 5.2 นั้น คีย์เวิร์ด K นั้นบรรจุอยู่ในคีย์เวิร์ดที่มีความหมายทั้งในวงแคบและวงกว้างโดยแตกออกเป็น 4 คีย์เวิร์ด ซึ่งได้แก่ K1, K2, K3 ,K4 ดังนั้น ถ้าหากเป็นระบบ Interactive ที่มีการโต้ตอบกันทางเครื่องแล้ว การค้นหาข้อมูลนั้นสามารถถูกเปลี่ยนรูปแบบหรือก่อรูปได้อย่างง่ายดาย โดยการใช้คำที่มีความสัมพันธ์เกี่ยวข้องกันของคีย์เวิร์ดเหล่านี้

วิธีการหนึ่งที่ชัดเจนในการกระทำ Boolean Search ก็โดยผ่าน Inverted File เราได้บันทึกลิสท์ (List) หนึ่งสำหรับแต่ละคีย์เวิร์ดและในแต่ละลิสท์จะมีแอดเดรส (หรือหมาย

K1 – List : D1, D2, D3, D4

K2 – List : D1, D2

K3 – List : D1, D2, D3

K4 – List : D1

ถ้าข้อคำถาม Q = (K1 AND K2) OR (K3 AND K4)
ซึ่งผลลัพธ์ที่ได้ก็คือ Set { D1, D2, D3}

วิธีการหนึ่ง ซึ่งเป็นวิธีการที่มีการเปลี่ยนแปลงแก้ไขเล็กน้อยเรียกว่า Fully Boolean Search ก็คือ วิธีที่อนุญาตให้มี AND Logic แต่นำเอาจำนวนที่แท้จริงของค่าดัชนีที่ข้อคำถามมีตรงกันกับของเอกสารหนึ่งมาคิดด้วย ตัวเลขจำนวนนี้ เราเรียกว่าเป็น Co-ordination Level ส่วนวิธีการค้นหานั้น เราเรียกว่า Simple Matching เนื่องจากว่าในแต่ระดับ เราสามารถที่จะมีเอกสารได้มากกว่าหนึ่งเอกสาร ดังนั้น เอกสารเหล่านั้นจึงถูกเรียกว่าเป็น partially Ranked โดย Co-ordination Level

ยกตัวอย่างเดียวกันกับตัวอย่างที่กล่าวมาแล้วข้างต้น และให้

ข้อคำถาม Q = K1 AND K2 AND K3

เราจะได้ Ranking ต่อไปนี้

Co-ordination Level	
3	D1, D2
2	D3
1	D4

ผลลัพธ์นั้นเกิดจากการใช้วิธีการวัดความคล้ายคลึงโดยวิธี Simple Matching ซึ่งคำนวณจาก $|D \cap Q|$ กล่าวคือ ขนาดของโอเวอร์แล็พระหว่าง D และ Q ซึ่งจะถูกแสดงขนาดเป็นชุดของคีย์เวิร์ด ก็คือ Simple Matching Coefficient นั่นเอง วิธีนี้นั้นใช้ทฤษฎีของเซตและพีชคณิต เป็นวิธีการในการค้นคืนข้อมูล และได้รับการประยุกต์ในระยะเริ่มต้น ข้อเสียของวิธีนี้คือ ผลลัพธ์ที่ได้อาจเป็นเอกสารที่จำแนกเป็นตรงหรือไม่ตรงตามความต้องการของผู้ใช้ก็ได้เป็นเพียงแค่การค้นคืนข้อมูลไม่ใช่ข่าวสาร ผู้ใช้มีความยากลำบากในการแสดงคำขอ ไม่มีการให้นำหนักคำดัชนี การเชื่อมโยงชุดคำขอด้วย และ (And), หรือ (or) และ

ระบบที่ประยุกต์มีลำดับในการประมวลผลที่แตกต่าง ถ้าไม่ใส่ลำดับในการประมวลผล เช่น A or B and C อาจตีความได้ A or (B and C) หรือ (A or B) and C เป็นต้น ตัวดำเนินการไม่ (NOT) ก็เป็นอีกตัวดำเนินการหนึ่งที่ยากต่อการตีความ เช่น (NOT A) and B and C ในการประมวลผลควรคืนเฉพาะเอกสารที่ไม่มีคำ A แล้วจึงประมวลผลหาเอกสารที่มีคำ A และ C ประกอบ

ตัวอย่างที่ 5.1 Boolean Search in SQL

```
(SELECT docID FROM InvertedFile
  WHERE word = "window"
 INTERSECT
 SELECT docID FROM InvertedFile
  WHERE word = "glass" OR word = "door")
EXCEPT
SELECT docID FROM InvertedFile
 WHERE word="Microsoft"
ORDER BY magic_rank()
```

5.2 Matching Functions

ในบางกรณีที่ต้องการค้นหาข้อมูลที่ข้อมูลที่ต้องการมีความซับซ้อน การค้นหาข้อมูลนั้นอาจปรับเปลี่ยนกลยุทธ์โดยค้นหาโดยใช้วิธีการที่เรียกว่า Matching Function ซึ่งเป็นวิธีการที่คล้ายคลึงกับ Association Measure แตกต่างกันตรงที่ว่า Matching Function นั้นจะวัดความใกล้ชิดระหว่างข้อคำถามกับตัวแทนเอกสารหรือตัวแทนคัสเตอร์ ในขณะที่ Association Measure นั้นจะใช้กับการเปรียบเทียบในทางคณิตศาสตร์ ซึ่งทั้ง 2 ฟังก์ชั่นมีคุณสมบัติเหมือนกันแตกต่างกันเพียงแต่ในความหมายเท่านั้น

Matching Function มีหลายๆตัวอย่าง แต่ฟังก์ชั่นที่ง่ายที่สุดได้แก่ฟังก์ชั่นที่เกี่ยวข้องกับ Simple Matching Search Strategies ดังนี้

กำหนด M เป็น Matching Function

D เป็น เซ็ทของคีย์เวิร์ดที่ใช้แทนเอกสาร

Q เป็น เซ็ทของคีย์เวิร์ดที่ใช้แทน Query

ดังนั้น

$$M = \frac{2|D \cap Q|}{|D| + |Q|}$$

อีกตัวอย่างหนึ่งของ Matching Function ซึ่งคล้ายๆ กับ Dice's Coefficient วิธีหนึ่งที่ถูกนิยมใช้โดย SMART Project ซึ่งเราเรียกว่า Cosine Correlation

สมมุติว่า เอกสารและข้อคำถาม ถูกแสดงแทนโดยเวคเตอร์ที่เป็นตัวเลขใน t-Space นั่นคือ Q = (q1,q2, ... ,qt) และ D = (q1,q2, ... ,qt) ซึ่ง qi และ di เป็นน้ำหนักที่เกี่ยวข้องกับคีย์เวิร์ด i

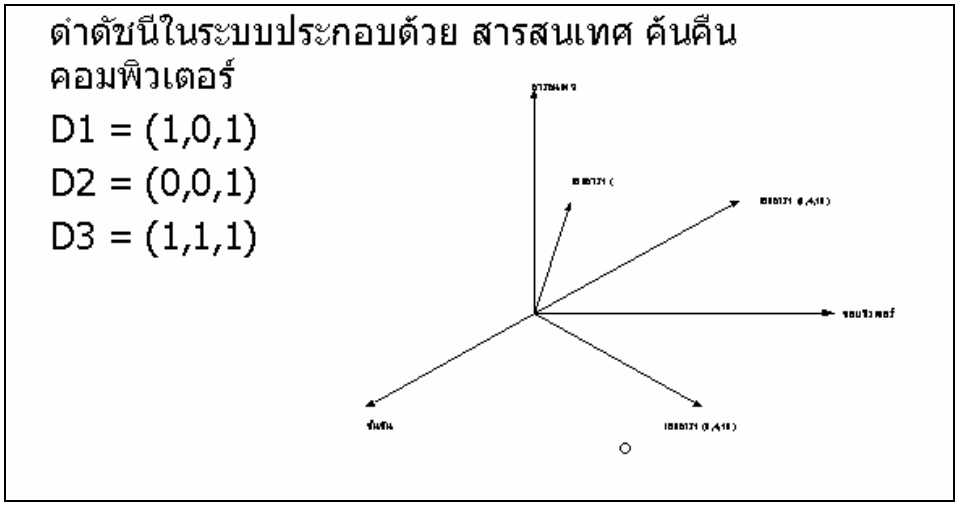
$$\text{Cosine Correlation} = \frac{\sum_{i=1}^{t} qidi}{\left(\sum_{i=1}^{t} (qi)^2 \sum_{i=1}^{t} (di)^2\right)^{1/2}}$$

หรือถ้าจะเขียนนิยามสำหรับ Vector Space หนึ่งที่มี Euclidean Norm

$$r = \frac{(Q,D)}{\|Q\|\|D\|} = \text{cosine } \alpha$$

ซึ่ง α จะเป็นมุมระหว่างเวคเตอร์ Q และ D

ตัวอย่างที่ 5.2 การแทนเอกสารด้วยเวกเตอร์โมเดล



การค้นหาแบบเวกเตอร์ ไม่ใช้น้ำหนักแบบไบนารีเหมือนกับระบบของบูลีน ค่าความเหมือนระหว่างเอกสารและคำขอ คำนวณด้วยค่าของมุม cosine ซึ่งผลลัพธ์ที่ได้สามารถจัดลำดับให้ตรงกับความต้องการของผู้ใช้ได้ คำในคำขอก็สามารถนำมาคำนวณเพื่อถ่วงน้ำหนักได้ มีหลายกรรมวิธีในการถ่วงน้ำหนักคำดัชนี หลักการหลักในการถ่วงน้ำหนักคือการจัดหมู่ข้อมูล (Clustering) การจัดหมู่ข้อมูลเป็นการแยกแยะว่าเอกสารใดตรงกับคำขอคุณลักษณะในการพิจารณาการจำแนก เพื่อจำแนกว่าวัตถุหรือเอกสารใดบ้างควรอยู่ในเซต

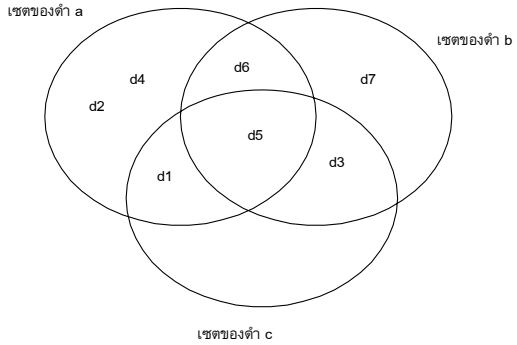
นิยามของเวกเตอร์โมเดล

- กำหนดให้ wij เป็นนำหนักของคำที่ ki ของเอกสาร dj เอกสารแต่ละฉบับจะแทนด้วย dj = (w1,j, w2,j,…, wtj)
- คำค้นคืน (Query) ก็มีการให้นำหนักของคำด้วยแบบเดียวกัน คือ wi,q เป็นนำหนักของคำที่ ki กับคำขอ q ดังนั้นเวคเตอร์คำขอจะอยู่ในรูปของ q = (w1,q, w2,q, …, wt,q)
- การค้นคืนข้อมูลจะวัดด้วยค่าความคล้ายกันของเอกสารคือ

$$sim(d_j, q) = \frac{\overline{d_j} * \overline{q}}{|\overline{d_j}| * |\overline{q}|} = \frac{\sum_{i=1}^{t} w_{i,j} * w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^{2}} * \sqrt{\sum_{i=1}^{t} w_{i,q}^{2}}}$$

การค้นคืนของเวคเตอร์โมเดลจะคืนตามค่าความเหมือนของความเหมือนที่คำนวณได้ เอกสารจะถูกค้นคืนให้แม้จะมีค่าความเหมือนเพียงบางส่วนการจัดทำดัชนีคำค้นจะทำได้หลายวิธี และส่วนใหญ่จะใช้หลักการของระบบน้ำหนักคำเข้ากำหนดน้ำหนักคำ การจัดกลุ่มของเอกสารก็ใช้หลักการเดียวกันกับการคำนวณค่าความเหมือน (sim)

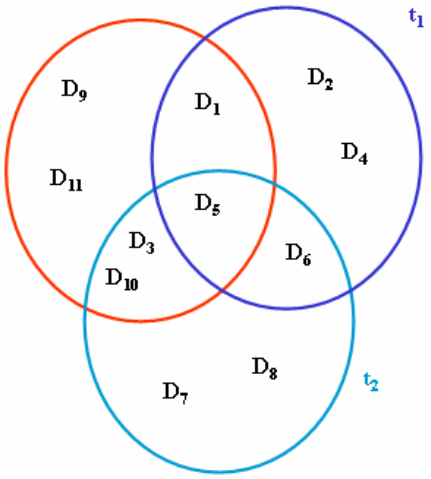
ตัวอย่างที่ 5.3 กลยุทธ์ในการค้นหาโดยใช้ Matching Function



	a	b	c	q*dj
d1	1	0	1	2
d2	1	0	0	1
d3	0	1	1	2
d4	1	0	0	1
d5	1	1	1	3
d6	1	1	0	2
d7	0	1	0	1
q	1	1	1	

docs	*t1*	*t2*	*t3*	RSV=Q.Di
D1	1	0	1	4
D2	1	0	0	1
D3	0	1	1	5
D4	1	0	0	1
D5	1	1	1	6
D6	1	1	0	3
D7	0	1	0	2
D8	0	1	0	2
D9	0	0	1	3
D10	0	1	1	5
D11	0	0	1	4
Q	1	2	3	
	q1	*q2*	*q3*	

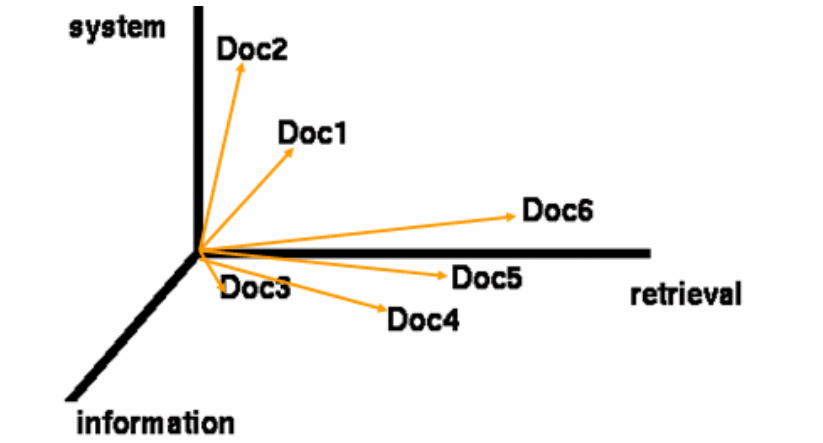
Q is a query – also represented



Boolean term combinations

เวกเตอร์เอกสาร (Document Vector)

เอกสารนั้นถูกแทนเป็น “bags of words” ในรูปของเวกเตอร์ โดยวิธีการคำนวณ ซึ่งเวกเตอร์เหมือนกับข้อมูลชนิดแถว(array) ของ floating point มีทั้งทิศทาง(direction) และน้ำหนัก(magnitude) แต่ละเวกเตอร์สามารถแทนชุดของคำที่มีการรวบรวมไว้ เอกสารถูกแทนเป็นเวกเตอร์ในเทอมของ space อาจเป็นรากศัพท์(stems) หรือเป็น เลขไบนารีหรือน้ำหนัก (binary or weighted vectors of terms) สมมุติฐานหลักของ Vector Space Model: คือ เอกสารจะถูกนำมารวมด้วยกันในspace เป็นความคล้ายคลึงกันในความหมายการแทนข้อสอบถาม(Queries) ที่ผู้ใช้ร้องขอจะถูกแทนในลักษณะเดียวกันกับเอกสาร ซึ่งน้ำหนักข้อสอบถามและเอกสาร เป็นพื้นฐานของความยาวและทิศทางของเวกเตอร์ การวัดระยะทางของเวกเตอร์ระหว่างข้อสอบถามและเอกสารถูกใช้สำหรับเป็น rank ในการดึงเอกสาร ดังรูปที่ 5.3



รูปที่ 5.3 : Documents in 3D Space

ตัวอย่างที่ 5.4 สมมุติว่าเอกสารและข้อสอบถามถูกแทนในรูปของเวกเตอร์

ตำแหน่งที่ 1 สอดคล้องกับ term 1
ตำแหน่งที่ 2 สอดคล้องกับ term 2
ตำแหน่งที่ t สอดคล้องกับ term t
น้ำหนักของแต่ละเทอมถูกเก็บในแต่ละตำแหน่ง

ดังนั้น

$$D_i = w_{d_{i1}}, w_{d_{i2}}, ..., w_{d_{it}}$$

$$Q = w_{q1}, w_{q2}, ..., w_{qt}$$

$$w = 0 \text{ if a term is absent}$$

สมมุติว่า เวกเตอร์ของเอกสารเป็นดังนี้

ID	nova	galaxy	heat	h'wood	film	role	diet	fur
A	10	5	3					
B	5	10						
C				10	8	7		
D				9	10	5		
E							10	10
F							9	10
G	5		7			9		
H		6	10	2	8			
I				7	5		1	3

จากข้อมูลข้างต้นเห็นได้ว่า

“Nova” occurs 10 times in text A

“Galaxy” occurs 5 times in text A

“Heat” occurs 3 times in text A

(Blank means 0 occurrences.)

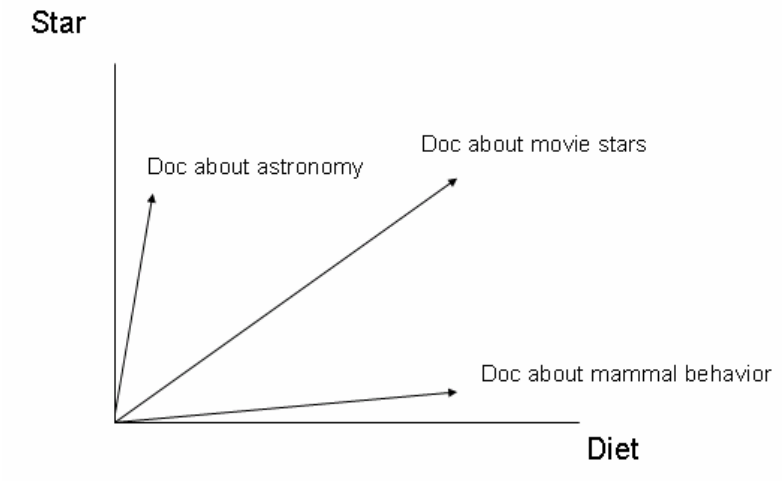
“Hollywood” occurs 7 times in text I

“Film” occurs 5 times in text I

“Diet” occurs 1 time in text I

“Fur” occurs 3 times in text I

เราสามารถ plot เวกเตอร์ได้ดังนี้



ความสำเร็จในการใช้ Vector space Model อยู่ที่ความเข้ากันได้ดีในเชิงมิติ (Dimensional Compatibility) คือการเปรียบเทียบของเอกสารกับคำร้องขอ จะอยู่บนพื้นฐานของการเปรียบเทียบคำที่เหมือนกันในแต่ละเอกสาร การกำหนดน้ำหนักให้กับคำ ในเวกเตอร์เป็นกระบวนการที่ซับซ้อน น้ำหนักสามารถถูกกำหนดโดยอัตโนมัติในเอกสารขึ้นอยู่กับการนับความถี่ของคำ คำที่ปรากฏยิ่งบ่อยในเอกสารจะมีความสำคัญต่อเอกสารนั้นมาก ยกเว้นคำ stop word สรุปได้ว่ารูปแบบการทำงานของ Vector Model นั้น จะให้ความสำคัญความถี่ของคำที่ปรากฏอยู่ในเอกสารและความถี่มีผลต่อการให้น้ำหนักของคำ

-Term Frequency คือการใช้ความถี่ของคำ เช่น พบ 1 ครั้งเรียกว่า Term ทั้งนี้ขึ้นอยู่กับจำนวนคำของเอกสาร โดยเทอมจะแทนคำศัพท์ของแต่ละคำ

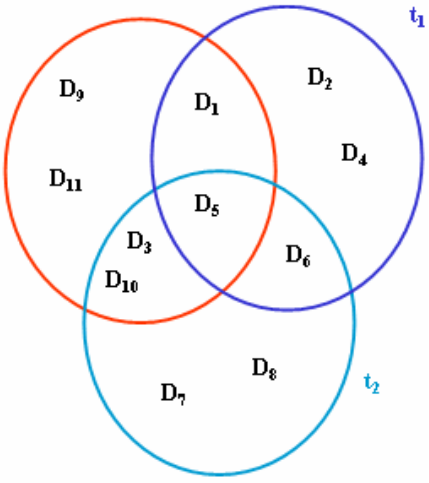
-Term Weight คือน้ำหนักของคำ ซึ่งเป็นความถี่ของคำๆหนึ่งที่พบในทุกๆเอกสาร

เราสามารถนำมาจัดอันดับของเอกสารโดยใช้เกณฑ์ความสำคัญของคำและการ Match กันของคำ ข้อดีของวิธีการนี้คือ ใช้คณิตศาสตร์เรียบง่ายในการคิด มีการพิจารณาจากความถี่ของคำและสามารถจัด Ranking ของเอกสารได้ สามารถใช้กับเอกสารที่มีข้อมูลมากๆได้ดี สำหรับข้อเสียนั้นคือ ไม่สนใจความหมายของคำ วลี โครงสร้างของคำ หรือคำที่มีความหมายเหมือนกัน(synonymy) และวิธีการนี้ไม่สามารถค้นหาหรือสืบค้นใส่เงื่อนไขแบบ Boolean Model ได้

ตัวอย่างที่ 5.5 กำหนด เวกเตอร์ของเอกสารและคำร้องขอ ดังนี้

docs	*t1*	*t2*	*t3*	RSV=Q.Di
D1	1	0	1	4
D2	1	0	0	1
D3	0	1	1	5
D4	1	0	0	1
D5	1	1	1	6
D6	1	1	0	3
D7	0	1	0	2
D8	0	1	0	2
D9	0	0	1	3
D10	0	1	1	5
D11	0	0	1	4
Q	1	2	3	
	q1	*q2*	*q3*	

Q is a query – also represented as a vector



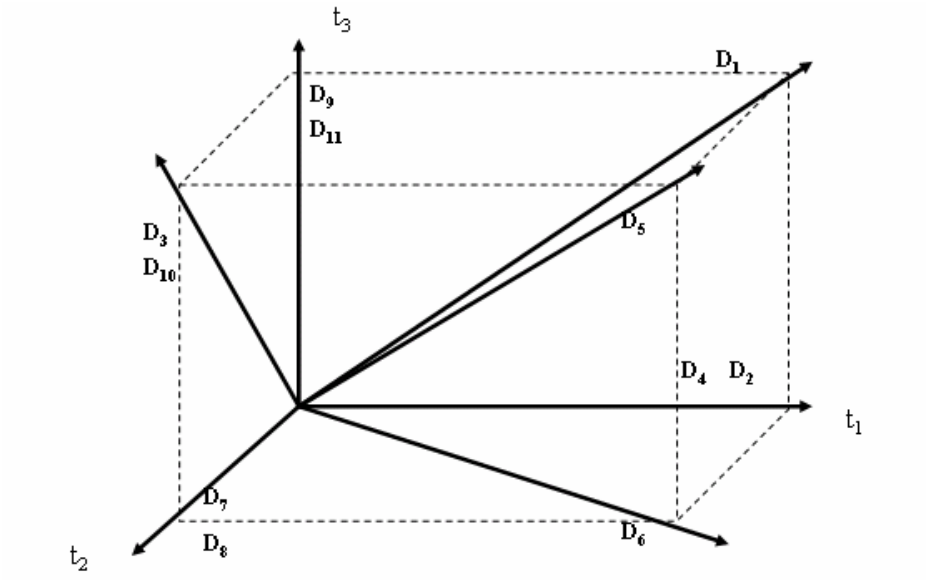
Boolean term combinations

โดย q1,q2,q3 คือ ข้อสอบถาม

t1,t2,t3 เป็น เทอมของคำสำคัญหรือดรรชนี

D1,D2,…D11 คือเอกสาร

นำความสำคัญของคำและการ Match กันของคำระหว่าข้อสอบถามและเอกสารมาสร้างเป็นเวกเตอร์ได้ดังนี้



5.3 Serial Search

การค้นหาอนุกรม(Serial Search) นั้นค่อนข้างจะใช้เวลา แต่มักจะถูกใช้เป็นส่วนหนึ่งของการค้นหาข้อมูลในระบบที่ใหญ่ วิธีการนี้แสดงถึงการใช้ Matching Function

สมมุติว่า ในระบบหนึ่งมีเอกสาร Di อยู่ N จำนวน Serial Search จะทำได้โดยการคำนวน N ค่าของ M(Q,Di) สำหรับ I = 1 ถึง N หรือกล่าวได้ว่า Matching Function นั้นจะถูกประเมินค่าที่แต่ละเอกสารสำหรับข้อคำถาม Q เดียวกัน บนพื้นฐานของค่า M(Q,Di) นี้ จะมีการกำหนดเอกสารชุดหนึ่งที่จะถูกดึงออกมา (Retrieved) ซึ่งมีอยู่ 2 วิธี คือ

(1) จะดึงเอกสารที่มีค่ามากกว่า Threshold ที่ถูกกำหนดมาให้อย่างเหมาะสมของ Matching Function นั้น แล้วทิ้งส่วนที่มีค่าน้อยกว่า ถ้า T เป็น Threshold ดังนั้น เซ็ทของเอกสารที่จะถูกดึงออกมาก็จะได้แก่

Set{Di | M(Q,Di) >= T}

(2) เอกสารถูกจัดลำดับตามค่าที่เพิ่มขึ้นของ Matching Function เลือก Rank Position R ให้เป็น Cut-off และทุกๆ เอกสารภายใต้ Rank นี้จะถูกดึงออกมา เพื่อว่า

B = { Di | r(i) <= R }

ซึ่ง r(i) เป็น Rank Position ที่ถูกกำหนดให้ Di เป็นที่คาดหวังว่าเอกสารที่เกี่ยวข้องกับข้อเรียกร้องจะถูกบรรจุอยู่ในเซ็ทของเอกสารที่ถูกดึงออกมา (Retrieved Set)

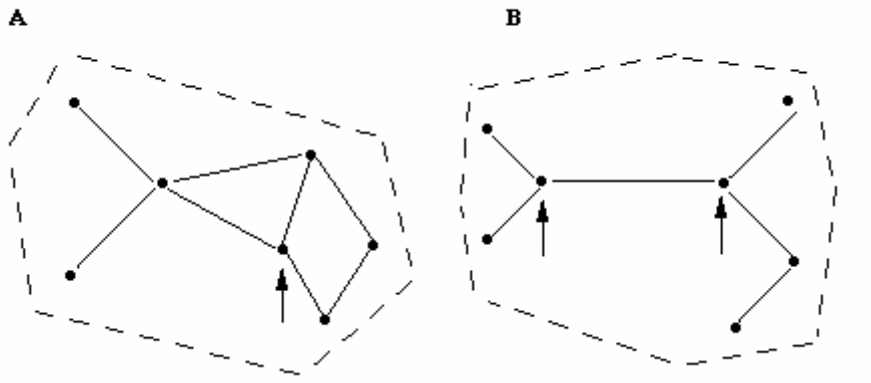
ปัญหาส่วนใหญ่ของวิธีการค้นหาทั้ง 2 แบบที่กล่าวมานี้ ก็คือ การกำหนดค่าของ Threshold หรือ Cut-off อาจจะมีค่าแตกต่างกันไป เนื่องจากว่า ไม่มีวิธีการใดที่จะบอกได้ว่า ค่า Threshold ค่าไหนจึงจะทำให้เกิดการดึงข้อมูลที่มีประสิทธิภาพมากที่สุด สำหรับแต่ละข้อคำถาม การกำหนดค่า Threshold นั้นจะได้จากการทดลองและสังเกตเพื่อหาค่าที่เหมาะสม

5.4 ตัวแทนคลัสเตอร์ (Cluster Representatives)

ก่อนที่จะพูดถึงวิธีการประยุกต์ใช้กลยุทธ์ในการค้นหากับกลุ่มของเอกสารที่ถูกจัดหมวดหมู่แล้ว จำเป็นที่จะต้องกล่าวถึงวิธีที่ใช้ในการแสดงแทนหมวดหมู่หรือคลัสเตอร์สักเล็กน้อย ในขณะที่ทำการค้นหาแบบตามลำดับ (Serial Search) จำเป็นที่จะต้องมีกลไกบางอย่างที่จะสามารถเปรียบเทียบข้อคำถาม กับแต่ละเอกสารในแฟ้มข้อมูลนั้น เมื่อเราทำการค้นหาจากแฟ้มข้อมูลคลัสเตอร์ จำเป็นที่จะต้องเปรียบเทียบข้อคำถามกับคลัสเตอร์ ในที่นี้เราเรียกว่า ตัวแทนคลัสเตอร์ (Cluster Representatives) ซึ่งจะพยายามที่จะสรุปเนื้อหาและเน้นคุณลักษณะเด่นของคลัสเตอร์ของเอกสารต่างๆ

ตัวแทนคลัสเตอร์หนึ่งควรจะเป็นสิ่งที่ว่า เมื่อข้อคำถามถูกใส่เข้ามา ตัวแทนคลัสเตอร์สามารถที่จะแยกแยะ เอกสารกลุ่มที่เกี่ยวข้อง (Relevant) กับข้อคำถามออกจากเอกสารกลุ่มที่ไม่เกี่ยวข้อง (Irrelevant) สิ่งนี้เป็นจุดที่ต้องการ แต่ในทางปฏิบัติแล้ว ไม่มีทฤษฎีไหนที่จะทำให้เราสามารถที่จะเลือกตัวแทนคลัสเตอร์ชนิดที่ถูกต้องได้พอดี วิธีหนึ่ง

ในที่นี้ จะให้ตัวอย่างหนึ่งของตัวแทนคลัสเตอร์แบบดั้งเดิม ถ้าหากเราสมมุติว่าคลัสเตอร์นี้ได้จากวิธีการคลัสเตอร์ที่ใช้พื้นฐานของความไม่คล้ายคลึงกัน แล้วเราสามารถที่จะแสดงแทนแต่ละคลัสเตอร์ที่บางระดับของความไม่คล้ายคลึงกันโดยกราฟหนึ่ง ดังในรูป 5.4 จากรูปนี้ A และ B เป็นสองคลัสเตอร์ โหนดจะแสดงแทนเอกสารและ เส้นตรงระหว่างโหนดสองโหนดใดๆก็ตามแสดงถึง เอกสารที่ถูกแสดงแทนโดยโหนดทั้งสองนั้น มีค่าของความไม่คล้ายคลึงกันน้อยกว่าระดับที่กำหนด วิธีการหนึ่งที่จะแสดงแทนคลัสเตอร์หนึ่งก็คือการเลือก Typical Member จากคลัสเตอร์นั้น วิธีง่ายวิธีหนึ่งที่จะทำตามจุดมุ่งหมายข้อนี้โดยการหาเอกสารที่ซึ่งถูกเชื่อมโยงกับเอกสารอื่นๆ ในคลัสเตอร์นั้นมากที่สุด



รูปที่ 5.4 : แสดงตัวอย่างของ Maximally Linked Documents

จากรูปที่ 5.4 แสดงถึงตัวแทนคลัสเตอร์ชนิดนี้ ที่เรียกว่า Maximally Linked Documents โดย คลัสเตอร์ A และ คลัสเตอร์ B มีพอยน์เตอร์ แสดงโดยลูกศรชี้ให้เห็นว่ามีเอกสารไหนบ้างที่ควรถูกเลือกให้เป็นตัวแทนคลัสเตอร์ ในบางครั้ง ตัวแทนของคลัสเตอร์หนึ่งอาจจะมีได้มากกว่าหนึ่งก็ได้ ตัวอย่างเช่น B มีตัวเลือกอยู่ 2 ตัว อาจจะทำการเลือกเอาตัวใดตัวหนึ่งเป็นตัวแทนคลัสเตอร์ก็ได้ หรือคลัสเตอร์อาจจะมีตัวแทนคลัสเตอร์เป็นลิสท์ (List) หนึ่งก็ได้

นอกจากนี้ เราจะแสดงถึงวิธีการอื่นที่จะแสดงแทนคลัสเตอร์ โดยการหาตัวแทนที่เป็นการเฉลี่ย Descriptions ของสมาชิกภายในคลัสเตอร์เหล่านั้น ซึ่งจะทำได้โดยการคำนวณหาจุดศูนย์ถ่วง (Centroid) ของคลัสเตอร์หนึ่ง

สมมุติว่า{ D1, D2, ... , Dn}เป็นเอกสารในคลัสเตอร์หนึ่ง และแต่ละ Di นั้นถูกแสดงโดย Numerical Vector (d1, d2, ... , dt) ซึ่งจุดศูนย์ถ่วงหรือ Centroid C ของคลัสเตอร์นั้น จะถูกกำหนดได้ดังนี้

$$C = \frac{1}{n} \sum_{i=1}^{n} \frac{D_i}{\| D_i \|}$$ (n เป็นจำนวนของเอกสารในคลัสเตอร์)

||Di|| นั้นโดยปกติก็เป็น Euclidean Norm, กล่าวคือ

||Di|| = $\sqrt{d12 + d22 + d32 + ... + dt2}$

บ่อยครั้งที่เอกสารไม่ได้ถูกแสดงแทนโดย Numerical Vector แต่จะแสดงแทนโดยไบนารีเว็คเตอร์ (Binary Vector) หรือคีย์เวิร์ดกลุ่มหนึ่ง ในกรณีเช่นนี้ เราก็ยังสามารถที่จะใช้ตัวแทนคลัสเตอร์ชนิดจุดศูนย์ถ่วงได้ แต่การ Normalization นั้นจะถูกแทนที่ด้วยวิธีการที่จะ Threshold ส่วนประกอบต่างๆ ของผลรวม $\sum$ Di ถ้ากล่าวให้ละเอียดอาจจะสมมุติว่า

Di เป็น ไบนารี่เว็คเตอร์ที่มีค่าดังต่อไปนี้

Di = 1 ถ้ามีคีย์เวิร์ดอันที่ j ในเอกสารนี้

0 ถ้าไม่มีคีย์เวิร์ดอันที่ j ในเอกสารนี้

ดังนั้น $S = \sum_{i=1}^{n} D_i$ (n เป็นจำนวนของเอกสารในคลัสเตอร์นั้น)

หรือให้ C = (C1, C2, ... , Ct) เป็นตัวแทนคลัสเตอร์

และ [Di], เป็นส่วนประกอบอันที่ j ของไบนารี่เว็คเตอร์ Di

เราอาจจะกำหนดค่า Cj ได้ 2 แบบดังนี้

(1) $c_j = \begin{cases} 1 \text{ if } \sum_{i=1}^{n} (D_i)_j > 1 \\ 0 \text{ otherwise} \end{cases}$

or

(2) $c_j = \begin{cases} 1 \text{ if } \sum_{i=1}^{n} (D_i)_j > \log_2 n \\ 0 \text{ otherwise} \end{cases}$

จากข้างต้นนี้ เราจะได้ตัวแทนคลัสเตอร์อันหนึ่งซึ่งเป็นไบนารี่เว็คเตอร์ C จากกรณี (1) สังเกตได้ว่า คีย์เวิร์ดที่เกิดขึ้นเพียงครั้งเดียวภายในคลัสเตอร์จะถูกละทิ้งไป ส่วนในกรณีที่ (2) นั้นมีการ Normalize ขนาด n ของคลัสเตอร์

$$\sum_{i=1}^{n} \sum_{j=1}^{d} ((D_i)_j - c_j)^2$$

$$\sum_{i=1}^{n} \sum_{j=1}^{d} ((D_i)_j - \bar{D}_j)^2 + n \sum_{j=1}^{d} (\bar{D}_j - c_j) \qquad (*)$$

where

$$\bar{D}_j = \frac{1}{n} \sum_{i=1}^{n} (D_i)_j$$

$$\sum_{j=1}^{d} ((D_i)_j - c_j)^2$$

(3) $c_j = \begin{cases} 1 & \text{if } \bar{D}_j > t_j \\ 0 & \text{otherwise} \end{cases}$

$\{Di \mid M(Q, Di) > T\}$

การที่จะได้ตัวแทนคลัสเตอร์นั้นอาจจะทำได้หลายวิธีและหลายแบบ แต่ดูเหมือนว่าประสิทธิภาพของการดึงข้อมูลนั้นไม่ได้เปลี่ยนแปลงมากนักเมื่อมีตัวแทนคลัสเตอร์ที่แตกต่างกันไป

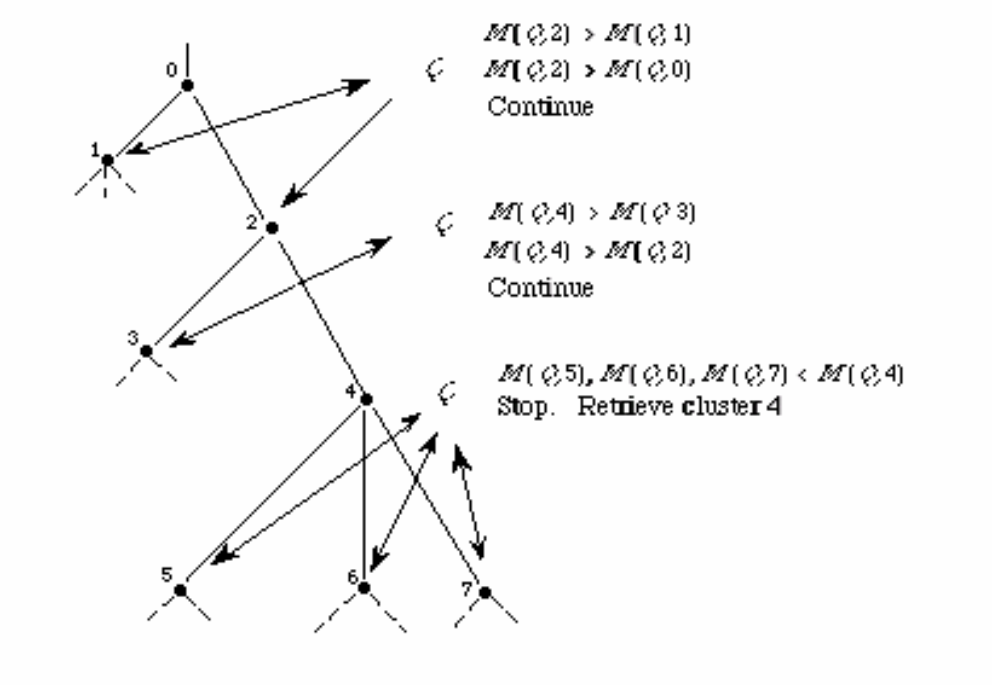
5.5 การดึงข้อมูลโดยอาศัยคลัสเตอร์เป็นพื้นฐาน(Cluster-Based Retrieval)

การดึงข้อมูลโดยอาศัยคลัสเตอร์เป็นพื้นฐานนั้น ต้องอาศัยข้อสมมุติฐานของคลัสเตอร์ (Cluster Hypothesis) ที่กล่าวว่า

"<u>เอกสารที่มีความใกล้ชิดคล้ายคลึงกัน ย่อมมีแนวโน้มที่จะเกี่ยวข้องกับข้อคำถามอันเดียวกัน</u> "

วิธีการคลัสเตอร์นั้นได้เลือกเอาเอกสารที่มีความสัมพันธ์ใกล้ชิด และจัดให้เอกสารเหล่านั้นอยู่ในกลุ่มหรือคลัสเตอร์เดียวกัน ในที่นี้ จะกล่าวถึงวิธีการที่คลัสเตอร์เหล่านี้จะถูกค้นหาเพื่อดึงเอาเอกสารที่ต้องการออกมาได้อย่างไร

สมมุติว่า มีเอกสารที่ถูกแบ่งตาม Hierarchic Classification แล้ววิธีการค้นหา ค้นหาแบบง่ายๆ อาจจะทำได้ดังต่อไปนี้



รูปที่ 5.5 :แสดงเอกสารที่ถูกแบ่งตาม Hierarchic Classification

จากรูปที่ 5.5 การค้นหาจะเริ่มต้นจาก (Root) ของทรี เช่น โหนด 0 เป็นต้น ซึ่งจะดำเนินไปเรื่อยๆ โดยการประเมินค่าของ Matching Function ที่โหนดต่างๆ ที่อยู่ถัดไปจากโหนด 0 ซึ่งในตัวอย่างก็ได้แก่ โหนด 1 และ 2 จะทำอย่างนี้ลงไปตามลำดับ การ

1) สมมุติว่าการดึงข้อมูลที่มีประสิทธิภาพสามารถกระทำได้โดยการหาเพียงคลัสเตอร์เดียว
2) สมมุติว่า แต่ละคลัสเตอร์สามารถูกแสดงแทนโดยตัวแทนคลัสเตอร์หนึ่ง เพื่อหาตำแหน่งที่ตั้งของคลัสเตอร์ที่บรรจุเอกสารที่เกี่ยวข้อง
3) ถ้าค่าสูงสุดของ Matching Function นั้นไม่ได้มีเพียงค่าเดียว จำเป็นที่จะต้องกระทำการอะไรพิเศษสักอย่าง
4) การค้นหานั้นจะสิ้นสุดลงและดึงอย่างน้อยหนึ่งเอกสารออกมา

การ Generalization ของการ Search แบบนี้ ก็คือ การที่อนุญาตให้มีการค้นหาลงไปได้มากกว่าหนึ่งกิ่งของทรี ซึ่งจะทำให้สามารถดึงข้อมูลที่ต้องการได้มากกว่าหนึ่งคลัสเตอร์ ด้วยความจำเป็น Decision Rule และ Stopping Rule นั้นจะต้องสลับซับซ้อนมากขึ้นเล็กน้อยข้อแตกต่างส่วนใหญ่ก็คือ จะต้องทำอะไรบางอย่างเพื่อการย้อนร้อยเก่า (Back – Tracking) สิ่งนี้จะเกิดขึ้นเมื่อกลวิธีในการค้นหาได้ประมาณ (โดยอาศัยค่าปัจจุบันของ Matching Function) ว่า การที่จะเคลื่อนลงไปกิ่งใดกิ่งหนึ่งข้างล่างนี้ จะเป็นการเสียเวลาโดยใช่เหตุ ซึ่งที่จุดนั้นอาจจะดึง หรือไม่ดึงคลัสเตอร์ปัจจุบัน การค้นหาก็จะย้อนรอยกลับไปยังจุดแยกและจะเลือกเอากิ่งอื่นลงไปตามทรี

กลวิธีข้างต้นนี้ เรียกว่า Top-Down Search ส่วนวิธี Bottom-up Search นั้นเป็นวิธีที่จะเริ่มเข้าสู่ทรีที่ Terminal Nodes โหนดใดโหนดหนึ่ง และจะเคลื่อนไปในทิศทางขึ้นบนไปสู่รูทของทรี ในวิธีนี้ จะเคลื่อนผ่านชุดหนึ่งของ Nested Clusters ที่มีขนาดใหญ่ขึ้นเรื่อยๆ และไม่ต้องใช้ Decision Rule เราเพียงแต่ต้องการ Stopping Rule ซึ่งอาจจะเป็น Cut-off ก็ได้ ตัวอย่างหนึ่งของการค้นหา อาจจะทำได้โดยการค้นหาคลัสเตอร์ที่ใหญ่ที่สุดที่บรรจุเอกสารที่ถูกแสดงแทนโดยโหนดที่เริ่มต้น และจะไม่เลยผ่านขนาดของ Cut-off เมื่อคลัสเตอร์นี้ถูกค้นพบครั้งหนึ่ง เซ็ทของเอกสารที่อยู่ภายในคลัสเตอร์นี้ก็จะถูกดึงออกมา การที่จะเริ่มต้นค้นหาเพื่อสนองตอบต่อคำถามในแต่ละครั้ง จำเป็นที่จะต้องรู้ล่วงหน้าว่า

5.6 การค้นหาข้อมูลแบบถามตอบ(Interactive Search Formulation)

เป็นการยากที่ผู้ใช้จะสามารถที่จะแสดงข้อมูลที่เขาต้องการเพียงครั้งเดียว เมื่อเขาดึงข้อมูลโดยระบบดึงข้อมูลแบบอัตโนมัติ ผู้ใช้มักจะต้องการที่จะทำการทดลองและพบข้อผิดพลาดซ้ำแล้วซ้ำอีก ในขณะที่เขาจะสร้างข้อคำถามของเขาเพื่อว่า ระบบอาจจะบอกอะไรแก่เขาได้บ้างเกี่ยวกับข้อคำถามของเขา ข้อมูลที่เขาต้องการเพื่อใช้สำหรับในการจัดสร้างข้อคำถามของเขาก็คือ

1. ความถี่ของการเกิดขึ้นของค่าที่เขาใช้ในการค้นหาภายในฐานข้อมูลนั้น
2. จำนวนของเอกสารที่เป็นไปได้ที่จะถูกดึงออกมาโดยข้อคำถามของเขา
3. ค่าต่างๆ ที่เกี่ยวข้องกับค่าที่เขาใช้ในการค้นหาข้อมูล
4. ตัวอย่างเล็กๆ ของ Citations ทีอาจจะถูกดึงออกมา
5. ค่าที่ถูกใช้ในการทำดัชนีของ Citations ใน 4. เป็นต้น

ทั้งหมดนี้ สามารถที่จะถูกเตรียมไว้ให้แก่ผู้ใช้ในระหว่างการค้นหาของเขาโดยระบบค้นคืนสารสนเทศแบบโต้ตอบ (Interactive Retrieval System) ถ้าผู้ใช้พบว่าค่าที่เขาใช้ในการค้นหาค่าหนึ่งเกิดขึ้นบ่อยครั้งมาก เขาอาจต้องการที่จะทำให้ค่านั้นมีความเฉพาะเจาะจงมากขึ้น โดยการพิจารณาดูจาก Hierarchic Dictionary ซึ่งจะบอกให้เขาทราบเกี่ยวกับสิ่งที่เขาต้องการจะเลือกนั้นว่ามีอะไรบ้าง ในทำนองเดียวกัน ถ้าหากข้อคำถามของเขาทำให้มีการดึงเอกสารมากเกินไป เขาสามารถที่จะทำให้มันอยู่ในวงแคบหรือเจาะจงมากขึ้น

ตัวอย่างของ Citations และการให้ดัชนี จะให้ข้อคิดเห็นแก่ผู้ใช้เกี่ยวกับชนิดของเอกสารที่จะดึงออกมาและให้ข้อคิดเห็นบางอย่างเกี่ยวกับวิธีการที่จะทำให้ค่าที่เขาใช้ในการค้นหานั้นได้แสดงถึงข้อมูลที่ต้องการอย่างมีประสิทธิภาพ เขาอาจจะเปลี่ยนแปลงแก้ไขข้อคำถามของเขาโดยอาศัยพื้นฐานบนผลลัพธ์ที่ได้จากการค้นหาจริงๆ ซึ่งเป็นรูปแบบหนึ่งของ Feedback

ตัวอย่างทั้งทางปฏิบัติและการทดลองของระบบที่มีกลไกสำหรับวิธีนี้ ได้แก่ ระบบ MEDLINE และ MEDUSA ซึ่งทั้งสองวิธีนี้มีรากฐานมาจากระบบ MEDLARS

5.7 Feedback

คำว่า Feedback โดยปกติจะถูกใช้เพื่อการอธิบายถึงกลไกที่ระบบหนึ่งสามารถที่จะที่จะพัฒนาการทำงาน (Performance) ของระบบบนงานหนึ่ง โดยการนำเอาการทำงานที่ผ่านไปแล้วมาพิจารณา หรืออาจจะกล่าวได้ว่า ระบบอินพุท- เอาท์พุท (Input-Output) แบบง่ายๆ นี้จะให้ผลลัพธ์กลับมาเกี่ยวกับข้อมูลต่างๆ จากผลลัพธ์ที่ได้ เพื่อว่าจะสามารถที่จะถูกนำไปใช้ในการพัฒนาการทำงานบนอินพุทอันถัดไป นิยามของ Feedback นั้นได้ถูกกำหนดเป็นอย่างดีใน Biological and Automatic Control System ซึ่งได้ถูกแสดงในหนังสือ Cybernatics ของ Norbert Wiener เกี่ยวกับการเอา Feedback มาใช้ในระบบดึงข้อมูล และพบว่ามีประสิทธิภาพไม่น้อย

ในที่นี้ ลองพิจารณาถึงกลวิธีการดึงข้อมูล (Retrieval Strategy) ที่ถูก Implemented โดย Matching Function M และสมมุติว่าทั้งข้อเรียกร้อง Q และตัวแทนเอกสาร D เป็นเวคเตอร์ t มิติ ซึ่ง t เป็นจำนวนของค่าดัชนีทั้งหมดในระบบ

โดยปกติแล้วกลวิธีการดึงข้อมูลทุกๆวิธีมีจุดมุ่งหมายเพื่อที่จะดึงแต่เอกสารทีเกี่ยวข้อง A และไม่ดึงเอกสารที่ไม่เกี่ยวข้อง A แต่เป็นที่น่าเสียดายว่าในทางปฏิบัตินั้นเป็นไปได้ยากในกรณีของการค้นหาแบบลำดับ (Serial Search) < T ซึ่ง T เป็น Threshold ที่กำหนดมาให้ ปรากฏว่า ในกรณีที่ M เป็น Cosine Correlation Function, ie

$$M(Q,D) = \frac{(Q,D)}{\|Q\| * \|D\|}$$

$$= \frac{1}{\|Q\| * \|D\|} * (q_1 d_1 + q_2 d_2 + .. + q_t d_t)$$

ซึ่งจะดึงเอกสารที่ M(Q,D) – T > 0 เท่านั้น

สมมุติว่า เราไม่ทราบเกี่ยวกับ A และ Ã ล่วงหน้า ดังนั้น รูปแบบของข้อคำถาม (Query Formulation) ที่ถูกต้อง Q_o อาจจะเป็นข้อคำถามอันใดอันหนึ่งที่

$M(Q_o,D) > T$ เมื่อ $D \varepsilon A$

และ $M(Q_o,D) <= T$ เมื่อ $D \varepsilon \tilde{A}$

การเริ่มต้นด้วยข้อเรียกร้อง Q ใด ๆ ก็ตาม สามารถที่จะปรับปรุงหรือเปลี่ยนแปลงแก้ไขข้อเรียกร้องนั้นซ้ำแล้วซ้ำอีกโดยการใช้ ข่าวสารที่ได้จากการโต้ตอบกลับมา (Feedback Information) เพื่อว่าข้อเรียกร้องนั้นได้เบนเข้าหา Qo มีทฤษฎีหนึ่งที่กล่าวว่า หากว่ามี Qo จริง ก็จะมี Iterative Procedure ที่จะประกันได้ว่า Q จะเบนเข้าหา Qo ในจำนวนสะเต็ปที่นับได้ Iterative Procedure นี้ถูกเรียกว่า Fixed-Increment Error Correction Procedure ซึ่ง

$Qi = Qi-1 + cD$ ถ้า $M(Qi-1, D) - T < 0$ และ $D \varepsilon A$

$Qi = Qi-1 - C$ ถ้า $M(Qi-1, D) - T > 0$ และ $D \varepsilon \tilde{A}$

และจะไม่มีการเปลี่ยนแปลง Qi - 1 ถ้าสามารถที่จะแบ่งแยกเอกสารได้ถูกต้อง

C เป็น Correlation Increment ซึ่งมีค่าเป็นค่าใดๆ ก็ได้ และจะต้องถูกทดลองกำหนดให้กับระบบหลายๆ ครั้ง จนกว่าจะพบว่าน้ำหนัก C นั้นจะสามารถแบ่งแยกเอกสารกลุ่ม A และ Ã ได้

ในทางปฏิบัติ, การดึงข้อมูลให้ถูกต้องนั้นไม่ใช่สิ่งง่ายเลย เพราะเราไม่สามารถหาค่า Qo ที่ถูกต้องได้ Salton ได้ใช้วิธี Feedback ในการปรับปรุงข้อคำถามเพื่อให้ดัชนีในเอกสารที่เกี่ยวข้องและดึงออกมานั้นมีน้ำหนักมากขึ้น และดัชนีในเอกสารที่ไม่เกี่ยวข้องนั้นมีน้ำหนักน้อยลง Feedback จะช่วยเพิ่มประสิทธิผลของการค้นคืนสารสนเทศ

5.8 ระบบสืบค้นเนมเพจ(Named Pages Finding System)

หลายปีที่ผ่านมางานวิจัยด้านระบบสืบค้นข้อมูลจำนวนมากได้มุ่งเน้นไปที่การค้นหากลยุทธ์ในการสืบค้นที่เหมาะสมและตอบสนองความต้องการของผู้ใช้ได้ดีที่สุด ระบบสืบค้นข้อมูลที่ผ่านมาในอดีตได้ใช้กลยุทธ์การค้นหาแบบตรงตามหัวข้อ(Subject Search Stratergy) ซึ่งมีหลักในการค้นหาให้พบเอกสารที่ไม่เกี่ยวข้องให้น้อยที่สุด ซึ่งกลยุทธ์ในการสืบค้นรูปแบบดังกล่าวจะให้ผลการค้นหาที่มีจำนวนมาก และยากต่อการเลือกคำตอบของผู้ใช้

กลยุทธ์การค้นหาอีกรูปแบบหนึ่งคือ กลยุทธ์การค้นหาแบบสิ่งที่เคยพบมาก่อนแล้ว (known-Item Search Strategy) กลยุทธ์นี้มีหลักการในการค้นหาโดยใช้คำสำคัญจากผู้ใช้ที่สามารถระบุได้ถึงความหมาย ชื่อ หรือเนื้อหา โดยสรุปของเอกสารที่ต้องการ และค้นหา

ระบบสืบค้นข้อมูลที่ประยุกต์ใช้กลยุทธ์การค้นหาแบบสิ่งที่เคยพบมาก่อนแล้วเป็นกลยุทธ์หลักในการสืบค้นคือระบบสืบค้นเนมเพจ(Named Page Finding System) ซึ่งได้รับการพัฒนาโดยกลุ่มนักวิจัยภายใต้การประชุมวิชาการ TREC(Text Retrieval Conference) ระบบดังกล่าวทำหน้าที่ค้นหาเว็บเพจที่ผู้ใช้เคยพบมาก่อนและค้นหาเว็ปเพจที่มีชื่อตรงกับชื่อที่ผู้ใช้กำหนด ระบบสืบค้นเนมเพจได้ถูกพัฒนาโดยกลุ่มนักวิจัยหลายสถาบันและได้ประสิทธิภาพในการสืบค้นที่แตกต่างกันไป แต่ระบบสืบค้นส่วนใหญ่ยังมีผลการสืบค้นที่มีความแม่นยำไม่เพียงพอต่อการใช้งาน การค้นหาสิ่งที่ปรากฏเด่นชัดจากฐานข้อมูลนั้น

กำหนดให้ $i = \{i_1, i_2, i_3, \ldots, i_m\}$ เป็นเซตของสิ่งที่ต้องการวิเคราะห์(item)
T เป็นไอเท็มเซตหรือเซตที่บรรจุสิ่งที่ต้องการวิเคราะห์(Itemset)
D เป็นฐานข้อมูลเป็นเซตของ T และ T เป็นซับเซตของ I

Tแต่ละเซตจะมีค่าสถิติเมื่อเทียบกับฐานข้อมูลทั้งหมดเรียกว่าค่าสนับสนุน(Support Value) ค่าสนับสนุนของไอเท็มเซต X หาได้จาก

$$\text{ค่าสนับสนุน (X,D)} = \frac{|\{T \in D \mid X \subseteq T\}|}{|D|}$$

กำหนดให้ X คือไอเท็มเซตใดๆ ซึ่ง T บรรจุ X ก็ต่อเมื่อ X เป็นซับเซตของ T ค่าสนับสนุนของไอเท็มเซต X ก็คือจำนวนร้อยละของ T ในฐานข้อมูลD ที่บรรจุ X ไอเท็มเซ็ต X จะถูกวิเคราะห์ว่าปรากฏเด่นชัดก็ต่อเมื่อค่าสนับสนุนของไอเท็มเซ็ต X ไม่น้อยกว่าค่าที่กำหนดโดยผู้ใช้ค่าหนึ่งที่เรียกว่าค่าสนับสนุนขั้นต่ำ (minimum Support)

จากข้อสมมติฐานที่ระบุว่าระบบสืบค้นเนมเพจที่มีการวิเคราะห์ชื่อที่แท้จริงของเอกสารสามารถค้นหาตัวแทนเอกสารได้อย่างถูกต้องและทำให้ระบบสืบค้นมีความแม่นยำมากขึ้น ส่วนการวิเคราะห์ตัวแทนเอกสารจึงถูกออกแบบเพิ่มเข้าไปในระบบสืบค้นเนมเพจเพื่อทำหน้าที่วิเคราะห์ชื่อเอกสารตามสมมุติฐานข้างต้น

ระบบเริ่มต้นทำงานในขั้นตอนการเตรียมข้อมูล(Data Preparation Phase) จากส่วนสกัดข้อมูล (Data Extraction Module) ส่วนสกัดข้อมูลทำหน้าที่คัดแยกส่วนของข้อมูลที่ต้องการออกจากเอกสาร เช่นคัดแยกเฉพาะส่วนหัวข้อหรือส่วนคำบรรยายลิงค์เป็นต้น ข้อมูลที่

ในขั้นตอนการสืบค้นเนมเพจ(Named Page Retrievel Phase) เริ่มต้นจากระบบรับคำถาม(Query) จากผู้ใช้งานและจัดรูปแบบให้อยู่ในรูปแบบเดียวกันกับตัวแทนชื่อเอกสารโดยส่วนการเตรียมคำถาม(Query Formulation Module) คำถามที่ได้จะถูกนำไปเปรียบเทียบความใกล้เคียงกับเอกสารที่ปรากฏเด่นชัดโดยส่วนสืบค้นเนมเพจ(Named Page Retrieval Module)ส่วนสืบค้นเนมเพจจะให้ผลลัพธ์การสืบค้นเป็นเอกสารที่มีตัวแทนเอกสารใกล้เคียงกับคำถามของผู้ใช้มากที่สุด

สำหรับงานวิจัยของ จิรัสย์ มาลาวงษ์ และอานนท์ รุ่งสว่าง ได้ทำการศึกษาเรื่องการเพิ่มประสิทธิภาพระบบสืบค้นเนมเพจด้วยการวิคราะห์ชื่อที่ปรากฏเด่นชัด ในส่วนวิเคราะห์ชื่อเอกสารที่ปรากฏเด่นชัดใช้กรรมวิธีอพิออรี(Aprioro Algorithm) เป็นหลักในการวิเคราะห์โดยกำหนดให้เซ็ตของไอเท็ม I = { W_1, W_2 , W_3 ,..., W_n } เป็นเซ็ตของคำในเอกสารคำบรรยายลิงค์และกำหนดให้ไอเท็มเซ็ต T ซึ่ง T เป็นซับเซตของ I เป็นคำบรรยายลิงค์แต่ละอันในเอกสารคำบรรยายลิงค์ จากนั้นจึงใช้กรรมวิธีอพิออรี(Apriori Algorithm)ที่ประยุกต์เพื่อใช้กับข้อมูลตัวแทนชื่อเอกสารค้นหาชื่อที่ปรากฏเด่นชัด ดังมีขั้นตอนการทำงานดังนี้

D = Anchor Document

L_k = Frequent Anchor Description Length k

Min sup = 25%

Scan D to find L1

Let k = 2

while L_{k-1} <> Ø do

C_k = apripri-gen(L_{k-1}).

for all transaction t € D do

C_t = subset(C_k , t).

for all candidate c € Ct do

c.count = c.count + 1.

end for

end for

L_k = {c V C_k |c.count >= min sup}.

end while

return $Y_k L_k$.

สำหรับส่วนสืบค้นเนมเพจ เป็นการเปรียบเทียบความใกล้เคียงระหว่างคำถามของผู้ใช้และตัวแทนชื่อเอกสารแต่ละอันในฐานข้อมูลทั้งหมด เอกสารที่มีค่าความใกล้เคียงกับคำถามของผู้ใช้มากที่สุดจะเป็นคำตอบของการสืบค้น ระบบสืบค้นข้อมูลทั่วไปนิยมใช้หลักการของความถี่ของคำที่ปรากฏในเอกสารรวมกับน้ำหนักของคำในเอกสารเป็นตัวแปรที่ระบุความใกล้เคียงของเอกสารและคำถามของผู้ใช้ แต่ระบบสืบค้นเนมเพจในงานวิจัยนี้ไม่สามารถใช้ตัวแปรที่ระบุข้างต้นได้เนื่องจากความถี่ของคำในเอกสารได้ถูกวิเคราะห์และแสดงเป็นชื่อเอกสารที่ปรากฏเด่นชัดแทนทำให้ตัวแปรความถี่ของคำไม่สามารถนำมาระบุความใกล้เคียงได้ ดังนั้นส่วนสืบค้นเนมเพจในงานวิจัยนี้จึงใช้ฟังก์ชันในการเปรียบเทียบความใกล้เคียงแจ็คการ์ด (Jaccard Similarity Function) ฟังก์ชันดังกล่าวนิยมใช้ในการเปรียบเทียบความใกล้เคียงของกลุ่มข้อมูล ฟังก์ชันนี้สามารถระบุค่าความใกล้เคียงของข้อมูล 2 ชุดได้อย่างมีประสิทธิภาพโดยไม่ต้องใช้ค่าตัวแปรความถี่ของข้อมูล

กำหนดให้ Q คือเซตของคำของคำถามี่ผู้ใช้กำหนดให้ระบบ

F_{ik} คือเซตของคำของตัวแทนชื่อเอกสารที่ปรากฏเด่นชัดลำดับที่ i ในเอกสารคำบรรยายลิงค์หมายเลข k

ค่าความใกล้เคียงของตัวแทนชื่อเอกสารแต่ละอันจะถูกรวบรวมเป็นค่าความใกล้เคียงรวมทั้งเอกสารหรือค่าสัมประสิทธิ์แจ็คการ์ดขอเอกสาร(Document's Jaccard Coefficient) ดังสมการดังต่อไปนี้

$$\text{Jaccard Coefficient}(Q,A_k) = \sum_{i=1}^{n} \frac{Q \cap F_{ik}}{Q \cup F_{ik}}$$

$$= \sum_{i=1}^{n} Jaccard(Q, F_{ik})$$

ผลของการวิจัยนี้สามารถเพิ่มความแม่นยำในการแทนเอกสารส่งผลให้ระบบสืบค้นมีค่าความแม่นยำเฉลี่ยสูงขึ้นและยังใช้ขนาดของฐานข้อมูลเล็กกว่าฐานข้อมูลทั้งหมด

5.9 บทสรุป

การค้นหาข้อมูลในระบบค้นคืนสารสนเทศอาจจะทำได้หลายวิธี ขึ้นอยู่กับโครงสร้างแฟ้มข้อมูลที่เก็บและธรรมชาติของภาษาข้อคำถามที่ผู้ใช้ใส่เข้ามา ในที่นี้อาจจะแบ่งออกเป็นหลายประเภท ได้แก่

1. Boolean Search เป็นวิธีการค้นหาข้อมูลโดยการเปรียบเทียบทางตรรกะระหว่างข้อคำถามกับเอกสาร

2. Serial Search เป็นการค้นหาข้อมูลที่ค้นหาเอกสารที่ต้องการไปตามลำดับโดยการเปรียบเทียบ

หาเอกสารที่มีค่าความใกล้ชิดกับข้อคำถามมากที่สุด

3. Cluster-Based Retrieval เป็นการค้นหาข้อมูลที่อาศัยข้อสมมุติฐานของคลัสเตอร์โดยอาศัยกฎการตัดสินใจ (Decision Rule) ในการค้นหาเส้นทางที่จะไปสู่เอกสารที่ต้องการ กับกฎของการสิ้นสุดการค้นหา (Stopping Rule) ในการค้นหาแบบนี้ คลัสเตอร์จำเป็นต้องมีตัวแทนคลัสเตอร์ที่จะใช้ในการเปรียบเทียบความใกล้ชิดระหว่าง คลัสเตอร์ที่มีเอกสารที่ต้องการกับข้อคำถาม

การที่ผู้ใช้จะใส่ข้อคำถามเข้าไปในระบบค้นคืนสารสนเทศเพื่อค้นหาเอกสารที่เขาต้องการนั้น ระบบจะช่วยให้ผู้ใช้สามารถปรับปรุงข้อคำถามของเขา โดยวิธีการ Feedback ข้อมูลต่าง ๆที่จะช่วยให้ผู้ใช้ได้พิจารณาและปรับปรุงข้อคำถามเพื่อให้ได้ข้อคำถามที่มีความสมบูรณ์มากที่สุด และสามารถดึงเอกสารที่ต้องการออกมาได้ใกล้เคียงกับสิ่งที่ผู้ใช้ต้องการมากที่สุด ซึ่งจะเป็นการช่วยเพิ่มประสิทธิผลของระบบ

แบบฝึกหัด

1. กลวิธีการค้นหาข้อมูลอาจจะทำได้หลายอย่าง อะไรจะเป็นตัวบังคับว่าควรใช้กลวิธีใดในการค้นหาข้อมูล
2. จงอธิบายวิธีการค้นหาข้อมูลต่อไปนี้

   2.1 Boolean Search

   2.2 Serial Search

   2.3 Cluster-Based Retrieval
3. Matching Function ต่างจาก Association Measure อย่างไร
4. ตัวแทนคลัสเตอร์ หมายถึงอะไร มีประโยชน์อย่างไร จงแสดงวิธีหาตัวแทนคลัสสเตอร์มาสัก 3 วิธี
5. Feedback หมายถึงอะไร มีจุดประสงค์อย่างไร และจงบอกข้อมูลที่อาจจะได้จากการ Feedback

บรรณานุกรม

สุมาลี เมืองไพศาล ,"การจัดการข้อมูล และการเรียกใช้ข้อมูล" ,สำนักพิมพ์มหาวิทยาลัยรามคำแหง ,CS337 ,2535

จิรัสย์ มาลาวงษ์ , อานนท์ รุ่งสว่าง ,"การเพิ่มประสิทธิภาพระบบสืบค้นเนมเพจด้วยการวิเคราะห์ชื่อที่ปรากฏเด่นชัด" , ภาควิชาวิศวกรรมคอมพิวเตอร์ มหาวิทยาลัยเกษตรศาสตร์ , https://pindex.ku.ac.th/file_research/NCSEC03-Final.pdf