การออกแบบแฟ้มผกผันเพื่อการค้นคืนข้อความไทย

โดย

สมชาย ประสิทธิ์จูตระกูล

โครงการวิจัยเลขที่ 49G-COM-2540

ทุนงบประมาณแผ่นดิน ปี 2540

สถาบันวิจัยและพัฒนาคณะวิศวกรรมศาสตร์

คณะวิศวกรรมศาสตร์

จุฬาลงกรณ์มหาวิทยาลัย

กรุงเทพฯ

พฤศจิกายน 2541

การออกแบบแฟ้มผกผันเพื่อการค้นคืนข้อความไทย

โดย

ผศ. ดร. สมชาย ประสิทธิ์จูตระกูล

โครงการวิจัยเลขที่ 49G-COM-2540

ทุนงบประมาณแผ่นดิน ปี 2540

สถาบันวิจัยและพัฒนาคณะวิศวกรรมศาสตร์

คณะวิศวกรรมศาสตร์

จุฬาลงกรณ์มหาวิทยาลัย

กรุงเทพฯ

พฤศจิกายน 2541

ชื่อโครงการ การออกแบบแฟ้มผกผันเพื่อการค้นคืนข้อความไทย

ชื่อผู้ดำเนินงาน สมชาย ประสิทธิ์จูตระกูล

เดือนและปีที่ทำวิจัยเสร็จ กันยายน 2541

บทคัดย่อ

งานวิจัยนี้นำเสนอขั้นตอนวิธีการหาคำเพื่อจัดทำดัชนีสำหรับระบบการค้นคืนข้อความไทยที่ใช้โครงสร้างแฟ้มผกผัน โดยอาศัยพจนานุกรมช่วยในการแยกคำ และยังสามารถจัดการกับกรณีที่ข้อความที่ได้รับมีคำที่ไม่ปรากฏในพจนานุกรม อาทิเช่นคำทับศัพท์ หรือคำที่สะกดผิดเป็นต้น โดยอาศัยกฎการแบ่งพยางค์ข้อความไทย ขั้นตอนวิธีนี้จำลองปัญหาด้วยกราฟการต่อและซ้อนกันของคำ ซึ่งมีโหนดแทนคำและเส้นเชื่อมแทนการต่อหรือซ้อนกันของคำ โดยมีเส้นทางสั้นสุดจากซ้ายไปขวาในกราฟนี้ แทนรายการคำพื้นฐานที่ควรถูกจัดทำดัชนีสำหรับแฟ้มผกผัน เวลาการทำงานของการหาคำนี้เป็น $O(n^2)$ โดยที่ n คือความยาวข้อความ ขั้นตอนวิธีนี้จะถูกใช้ทั้งในขั้นตอนการเตรียมเอกสารก่อนการทำดัชนี และการประมวลข้อคำถามก่อนการสืบค้น ผลการทดลองพบว่าจำนวนคำที่หาได้เพื่อทำดัชนีนั้นมีจำนวนประมาณ 30-50% ของจำนวนคำที่เป็นไปได้ทั้งหมดที่ปรากฏในข้อความทดสอบ

นอกจากนี้งานวิจัยนี้ยังได้นำเสนอขั้นตอนวิธีในการเข้ารหัสคำทับศัพท์ เพื่อรองรับการค้นคืนคำทับศัพท์ข้ามภาษาจากอังกฤษมาไทย นั่นคือระบบสามารถค้นคืนเอกสารที่มีคำสำคัญภาษาอังกฤษ หรือคำทับศัพท์เป็นภาษาไทยของคำอังกกฤษนั้น การเข้ารหัสนี้ปรับปรุงวิธีการเข้ารหัสเสียงและตารางการเข้ารหัสในระบบซาวน์เดกซ์ วิธีนี้ใช้เวลาการเข้ารหัสแปรเชิงเส้นตามความยาว จากผลที่ได้จากการทดลองพบว่าได้ค่าเรียกคืนและความแม่นยำมากกว่า 80% เมื่อจำกัดการพิจารณาเฉพาะคำที่รหัสเสียงมีความยาวเกิน 4

Project Title	Design of Inverted File for Thai-Text Retrieval
Name of the Investigator	Somchai Prasitjutrakul
Year	1998

Abstract

This work presents an algorithm for finding words used for indexing in a Thai-text retrieval system using inverted file structures. A dictionary is used during word separation. The algorithm can deal with text containing unknown words to the system dictionary such as transliterated words and words with typographical errors using a set of Thai syllable separation rules. The algorithm models the problem by constructing a word-adjacency-overlapping graph where vertices represent words and edges represent the word adjacency-overlapping relationships. A shortest path from the left-most vertex to the right-most vertex of the graph is a list of words reserved to be used as indices in the inverted file. The running time is $O(n^2)$ where n is the text length. The algorithm is used both in text preparation preprocessing before indexing and also in query processing before the actual search. Experimental results showed that the number of words obtained is approximately 30-50% of the total number of possible words appearing in the given text.

In addition, this work also presents an algorithm for encoding transliterated words suitable for cross-language retrieval system. Incorporating this feature enables the system to retrieve not only documents containing the English keywords, but also documents containing the corresponding transliterated words in Thai. The encoding algorithm modifies the Soundex encoding table and algorithm whose running time is linearly proportional to the word length. Experimental results showed that a high recall and precision of more than 80% can be achieved especially when the phonetic codes are longer than four.

กิตติกรรมประกาศ

ผู้วิจัยขอขอบคุณสถาบันวิจัยและพัฒนาคณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ที่ได้อนุมัติเงินทุนงบประมาณแผ่นดิน ประจำปีงบประมาณ ๒๕๔๐ สำหรับโครงการวิจัยนี้ และขอขอบคุณฝ่ายวิจัย คณะวิศวฯ ที่ช่วยประสานงานให้การดำเนินการวิจัยเป็นไปอย่างสะดวกยิ่ง

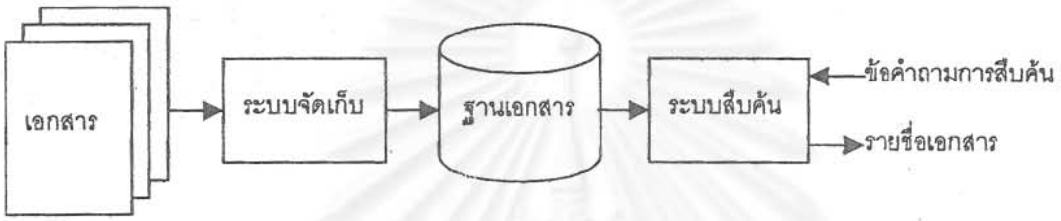
งานวิจัยนี้คงดำเนินไปด้วยความยากยิ่ง หากไม่ได้รับการร่วมนำเสนอแนวคิดและร่วมแรงจากผู้ช่วยวิจัยอันได้แก่เปรมิน จินดาวิมลเลิศ วิฑูรย์ กัลยาณวัฒน์ และประยุทธ สุวรรณวิสารธ ผู้วิจัยขอขอบคุณมา ณ โอกาสนี้ด้วย

สมชาย ประสิทธิ์จูตระกูล

สารบัญ

1. บทนำ

เป็นที่ทราบกันโดยทั่วไปว่าสารสนเทศเป็นหนึ่งในปัจจัยสำคัญในการดำเนินกิจการใดในปัจจุบัน การใช้สารสนเทศได้อย่างมีประสิทธิผลและประสิทธิภาพนั้น จำต้องอาศัยระบบการค้นคืนสารสนเทศ (Information Retrieval : IR) ซึ่งทำหน้าที่จัดเก็บและสืบค้นเอกสาร (รูปที่ 1) ระบบการจัดเก็บรับเอกสารในรูปแบบต่างๆ ประมวลผลคำภายในเอกสาร และสร้างระบบฐานเอกสาร ในขณะที่ระบบสืบค้นนั้นรับคำถามการสืบค้นจากผู้ใช้ ประมวลผลคำถาม สืบค้นฐานเอกสาร ประมวลผลคำตอบที่ได้ เพื่อนำเสนอกลับคืนสู่ผู้ใช้



รูปที่ 1 ระบบการค้นคืนสารสนเทศ

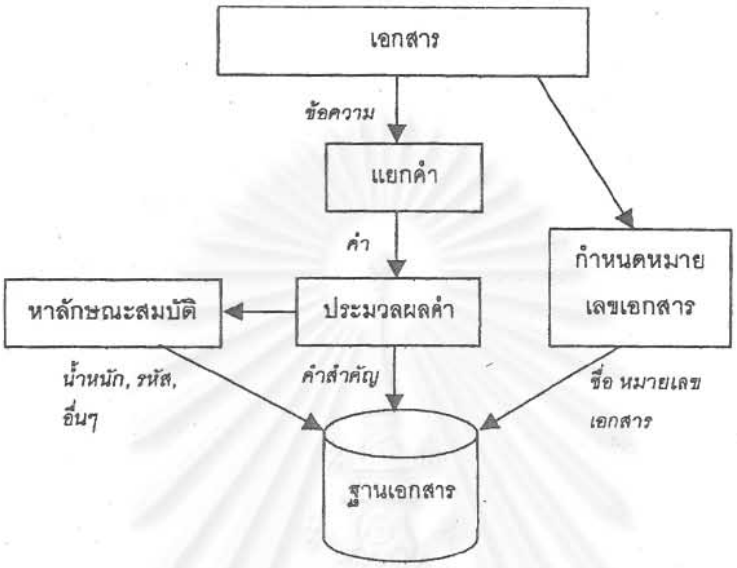
ขั้นตอนการทำงานโดยทั่วไปในการจัดเก็บ (ดูรูปที่ 2 ประกอบ) เริ่มจากการกำหนดหมายเอกสารให้กับแต่ละเอกสาร โดยข้อความในแต่ละเอกสารถูกแยกเป็นคำต่างๆ จากนั้นประมวลผลคำเพื่อให้ได้ชุดคำที่แทนเนื้อหาของเอกสารนั้น การประมวลผลคำอาจมีได้หลายลักษณะตัวอย่างเช่น

- การกรองคำไม่สำคัญ (stop words) ออก คำไม่สำคัญคือคำที่ระบบจะไม่จัดเก็บ เนื่องจากเป็นคำที่ปรากฏในเอกสารเป็นจำนวนมาก ตัวอย่างเช่น *ซึ่ง และ หรือ คือ เรา เขา* และอื่นๆ อย่างไรก็ตามในบางกรณีคำเหล่านี้อาจมีความสำคัญก็เป็นเป็นได้ ดังนั้นระบบการค้นคืนโดยทั่วไปจะมีรายการคำไม่สำคัญ (stoplist) ที่ผู้ใช้เลือกกำหนดได้

- การหารากคำ (stemming) ของคำสำคัญเหล่านั้น เพื่อจัดเก็บแทนคำนั้นๆ ทั้งนี้ เพื่อการประหยัดเนื้อที่ และเพื่อเพิ่มประสิทธิผลการสืบค้น ตัวอย่างเช่นคำว่า compute computing และ computer ต่างมีรากเดียวกัน หากจัดเก็บแยกกันอาจทำให้ผู้สืบค้นพลาดบางเอกสารที่มีคำว่า computer เมื่อเขาสืบค้นคำว่า computing เป็นต้น

- การแก้ไขคำผิด เอกสารที่จัดเก็บโดยทั่วไปนั้นจะมีความผิดพลาดเนื่องจากการสะกดคำผิด ซึ่งหากจัดเก็บทั้งๆที่ผิดนั้น ประสิทธิผลการค้นคืนจะลดลง ดังนั้นในกรณีที่ตรวจสอบพบว่าคำที่ได้มีความผิดพลาดก็ควรแก้ไขให้ถูกต้อง

หลังจากการประมวลผลคำ ระบบจัดเก็บจะหาลักษณะสมบัติของคำนั้น เพื่อใช้ประกอบการจัดเก็บและการสืบค้นโดยมีจุดประสงค์เพื่อเสริมประสิทธิผลการสืบค้น ตัวอย่างลักษณะสมบัติของคำมีอาทิเช่น จำนวนครั้งที่ปรากฏในเอกสาร รหัสการอ่านออกเสียงของคำ (เพื่อการค้นคืนที่ใช้การพ้องเสียงเป็นหลัก เช่นการค้นชื่อคน ชื่อเฉพาะ เป็นต้น) ภาษาที่ใช้เขียนคำนั้น เป็นต้น คำสำคัญ ลักษณะสมบัติต่างๆ และชื่อกับหมายเลขเอกสารจะถูกจัดเก็บเพิ่มเข้าไปในฐานเอกสาร เพื่อการสืบค้น

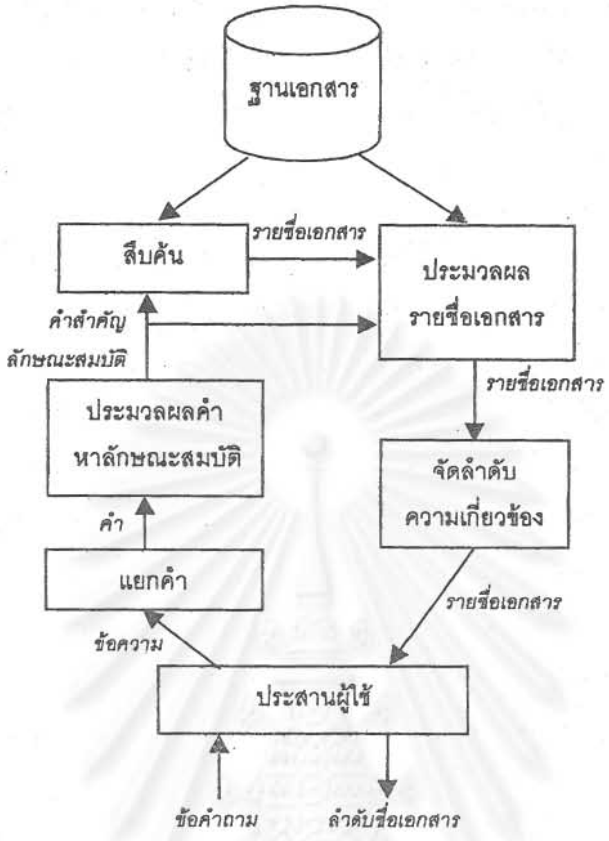


รูปที่ 2 ขั้นตอนการจัดเก็บเอกสาร

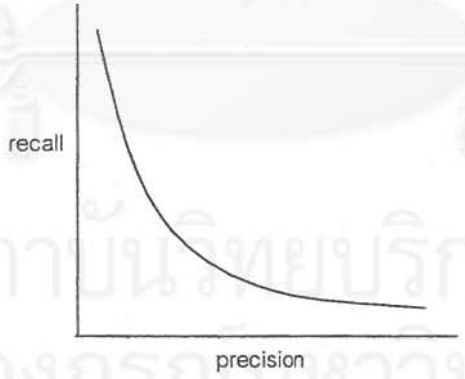
สำหรับการสืบค้นนั้นมีขั้นตอน (ดูรูปที่ 3 ประกอบ) เริ่มจากการรับข้อคำถาม (query) จากผู้ใช้ แยกข้อคำถามนี้ออกเป็นชุดของคำ เพื่อนำเป็นประมวลผลและหาลักษณะสมบัติในทำนองเดียวกันกับขั้นตอนการจัดเก็บ จากนั้นใช้คำสำคัญและลักษณะสมบัติที่หาได้เข้าสืบค้นในฐานเอกสาร จะได้รายชื่อของเอกสารที่เก็บคำที่สืบค้น นำมาประมวลผลรายชื่อเอกสาร หาความสัมพันธ์ระหว่างเอกสารกับคำสำคัญและลักษณะสมบัติของคำที่สืบค้น เพื่อได้เป็นคะแนนความเกี่ยวข้อง (relevancy) ของเอกสารต่างๆกับข้อความคำถามของผู้ใช้ หรือประมวลผลรายการเอกสารเชิงตรรก แล้วจึงนำรายชื่อเอกสารที่สืบค้นได้ไปจัดลำดับตามคะแนนความเกี่ยวข้อง เพื่อนำเสนอกลับเป็นผลลัพธ์การสืบค้นสู่ผู้ใช้

จากองค์ประกอบและขั้นตอนการจัดเก็บและสืบค้นเอกสาร ในระบบค้นคืนสารสนเทศที่กล่าวข้างต้นนี้ จะสังเกตได้ว่าประสิทธิภาพของการค้นคืนประเมินได้จากเวลาในการสืบค้น ซึ่งขึ้นกับการออกแบบโครงสร้างแฟ้มข้อมูลเพื่อจัดเก็บเอกสาร ส่วนคุณภาพของการค้นคืนประเมินได้จากความเกี่ยวข้องของรายชื่อเอกสารที่สืบค้นได้กับข้อคำถามที่ผู้ใช้ป้อนเข้าสู่ระบบ ซึ่งขึ้นกับขั้นตอนการประมวลผลคำสำคัญ และการประมวลผลรายชื่อเอกสาร คุณภาพของการค้นคืนมักวัดกันด้วย *ค่าความแม่นยำ* (precision) ของการค้นคืน ซึ่งคืออัตราส่วนของจำนวนเอกสารที่เกี่ยวข้องที่ค้นได้กับจำนวนเอกสารที่ค้นคืนกลับมา และ*ค่าเรียกคืน* (recall) ซึ่งคืออัตราส่วนเอกสารที่เกี่ยวข้องที่ค้นได้กับจำนวนเอกสารที่เกี่ยว

ข้องทั้งหมดในฐานเอกสาร ระบบการค้นคืนเอกสารโดยทั่วไปจะให้ค่าทั้งสองแปรผกผันซึ่งกันและกัน ตามรูปที่ 4



รูปที่ 3 ขั้นตอนการสืบค้น



รูปที่ 4 ความสัมพันธ์ระหว่างค่าความแม่นยำและค่าเรียกคืนของระบบค้นคืน

ระบบการค้นคืนสารสนเทศส่วนใหญ่ถูกออกแบบไว้สำหรับใช้งานกับภาษาอังกฤษ (หรือภาษาทางตะวันตกที่มีลักษณะใกล้เคียงกัน) จะสังเกตได้ว่าขั้นตอนการแยกข้อความในเอกสารหรือในข้อคำถามออกเป็นคำๆ นั้น จะกระทำได้ไม่ยาก แต่สำหรับกรณีข้อความภาษาไทยนั้นจะมีปัญหาเกิดขึ้น ซึ่งส่งผลต่อ

คุณภาพในการทำงานของขั้นตอนอื่นๆ ทั้งนี้เนื่องจากไม่สามารถระบุขอบเขตของคำในข้อความไทยได้อย่างเด่นชัด ตัวอย่างเช่น ข้อความ "โคลงเรือจนเรือโคลง" นั้นสามารถแบ่งแยกเป็นคำๆได้หลายแบบ อาทิ "โคลง เรือ จน เรือ โคลง" หรือ "โค ลง เรือ จน เรือ โคลง" เป็นต้น หรือในกรณีที่ข้อความนั้นมีคำเฉพาะที่ไม่ปรากฏในพจนานุกรมของระบบ อาจทำให้การแยกคำเกิดความผิดพลาดได้ อาทิเช่น "นายเจมส์มาร์ตินลาศึกษา" เป็นต้น ปัญหาการแยกข้อความออกเป็นชุดของคำ จึงเป็นปัญหาพื้นฐานที่สำคัญมากในระบบค้นคืนสารสนเทศภาษาไทย

งานวิจัยนี้มุ่งเน้นปัญหาการหาคำสำคัญ เพื่อจัดทำดัชนีให้กับระบบค้นคืนข้อความที่ใช้แฟ้มผกผันเป็นโครงสร้างหลักในการทำงาน ประสิทธิภาพการทำงานของระบบจะแปรผันตามคุณภาพของการหาคำสำคัญเพื่อจัดทำดัชนี หากใช้คำทุกๆคำที่ปรากฏในเอกสารเป็นคำสำคัญ จะได้ระบบการค้นคืนที่มีค่าเรียกคืนสูงแต่ค่าความแม่นยำต่ำ หากใช้คำสำคัญที่ยาวเกินไปก็ได้ค่าความแม่นยำสูงในขณะที่ค่าเรียกคืนต่ำ นอกจากนี้ยังมีผลต่อปริมาณเนื้อที่ของแฟ้มผกผันอีกด้วย รายงานวิจัยนี้มีลำดับการนำเสนอดังนี้ หลังจากบทนำที่ได้เกริ่นถึงระบบค้นคืนเอกสารและปัญหาการค้นคืนเอกสารภาษาไทยแล้ว บทที่ 2 นำเสนอเนื้อหาคร่าวๆของลักษณะโครงสร้างแฟ้มผกผัน บทที่ 3 นำเสนอในรายละเอียดของขั้นตอนวิธีการหาคำเพื่อทำดัชนีที่ได้ออกแบบไว้พร้อมผลการทดลอง ตามด้วยขั้นตอนวิธีการเข้ารหัสคำทับศัพท์เพื่อสนับสนุนการค้นคืนข้ามภาษาไทยระหว่างไทยและอังกฤษ และผลการทดลองในบทที่ 4 ปิดท้ายด้วยบทที่ 5 ซึ่งสรุปเนื้อหาของงานวิจัยพร้อมทั้งรายการข้อเสนอแนะประเด็นปัญหาที่ต้องทำวิจัยต่อไป

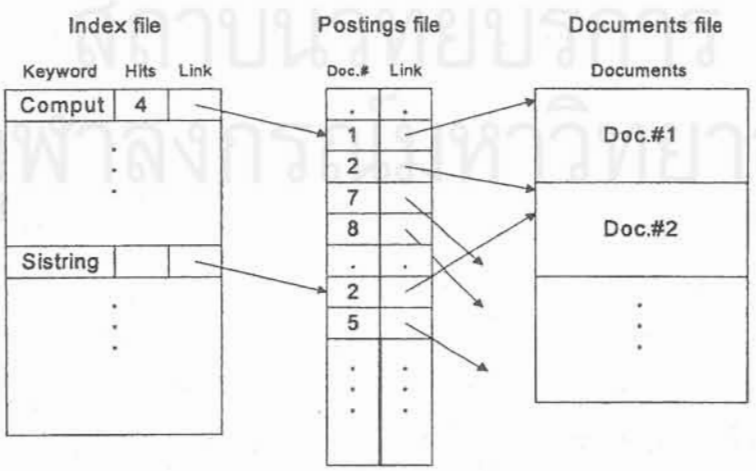
2. โครงสร้างแฟ้มข้อมูลผกผัน

โครงสร้างแฟ้มข้อมูลที่ใช้ในระบบค้นคืนข้อความพอแบ่งอย่างคร่าวๆ ได้เป็นสามรูปแบบคือ การใช้ดัชนีคำที่เรียงลำดับ การจัดกลุ่มแฟ้ม และการใช้ดัชนีคำโดยใช้แนวคิดการแฮชข้อมูล [5] สำหรับวิธีการจัดเรียงดัชนีคำนั้นใช้รูปแบบแฟ้มข้อมูลแบบผกผันในการจัดเก็บ แนวคิดของแฟ้มผกผันคือการจัดเก็บดัชนีซึ่งเป็นรายการของคำสำคัญและลักษณะสมบัติต่างๆ โดยที่แต่ละคำสำคัญจะมีตัวชี้ตำแหน่งของเอกสารที่มีคำสำคัญนั้นๆ การค้นหาคำสำคัญใด จึงเป็นการค้นหาในดัชนี จะได้รายการของตัวชี้ตำแหน่งเอกสาร แล้วจึงวิ่งตามตัวชี้ดังกล่าวเพื่อไปยังเอกสารต้นฉบับ โดยทั่วไปแฟ้มผกผันให้ประสิทธิภาพในการค้นหาที่ดี แต่มีข้อเสียตรงที่ต้องการเนื้อที่จัดเก็บดัชนี ดังนั้นระบบค้นคืนข้อความที่คำนึงถึงเนื้อที่เสริมในการจัดเก็บดัชนีนี้ จะต้องมีคุณสมบัติในการลดจำนวนคำสำคัญที่ไม่จำเป็นออกจากระบบ

โครงสร้างแฟ้มผกผันมีได้หลายรูปแบบ ในที่นี้จะนำเสนอรูปแบบที่สำคัญสามรูปแบบคือ แบบแถวเรียงลำดับ แบบต้นไม้บี และแบบทรัย

2.1 โครงสร้างแบบแถวเรียงลำดับ

โครงสร้างแบบแถวเรียงลำดับจัดเก็บดัชนีด้วยแถวลำดับ แถวลำดับนี้เก็บคำสำคัญเรียงจากน้อยไปมาก เพื่อใช้ค้นหาคำได้รวดเร็วด้วยการค้นหาแบบทวิภาคซึ่งใช้เวลาการค้นหาเป็น O(log n) แต่ละช่องในแถวลำดับเก็บตัวชี้ไปยังระเบียนของแฟ้มตำแหน่ง (posting file) ซึ่งเก็บหมายเลขเอกสารและตำแหน่งของเอกสาร ดังแสดงในรูปที่ 5 โครงสร้างแบบแถวเรียงลำดับนี้จะมีความยุ่งยากเมื่อมีการปรับเปลี่ยนข้อมูลในแถวลำดับ แต่ง่ายในการสร้าง

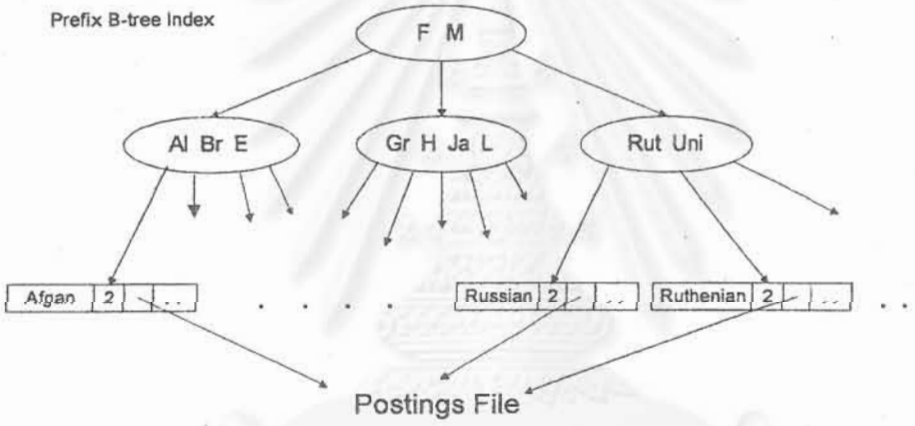


รูปที่ 5 แฟ้มผกผัน [5]

2.2 โครงสร้างแบบต้นไม้บี

โครงสร้างแบบต้นไม้บี จัดเก็บข้อมูลในต้นไม้ค้นหาหลายทาง (multiway search tree) ที่เรียกว่า ต้นไม้บี (B-tree) การค้นหาข้อมูลจะเริ่มกระทำที่รากของต้นไม้ โดยข้อมูลที่เก็บในแต่ละโหนดนั้นจะเป็นตัวเปรียบเทียบเพื่อเลือกกิ่งที่จะลงหาต่อในต้นไม้ เพื่อนำไปสู่จุดหมาย เวลาการค้นหาแปรตามความสูงของต้นไม้ (โดยปกติต้นไม้บีจะมีทางแยกต่อโหนดเป็นจำนวนมาก จึงเป็นต้นไม้ที่เตี้ย ทำให้การค้นหาข้อมูลกระทำได้รวดเร็ว) การปรับเปลี่ยนข้อมูลในต้นไม้บีกระทำได้ในเวลาที่แปรตามความสูงของต้นไม้เช่นกัน [29]

โครงสร้างแบบต้นไม้บีประเภทหนึ่งที่เหมาะสำหรับการจัดเก็บคำ คือโครงสร้างต้นไม้บีแบบนำหน้า (prefix B-tree) ข้อมูลที่เก็บตามโหนดต่างๆจะไม่ใช่ตัวอักษรทุกตัวของคำ แต่จะเป็นเพียงตัวอักษรนำหน้าบางตัวของคำที่เพียงพอต่อการเปรียบเทียบเพื่อนำไปสู่โหนดจุดหมาย จึงเป็นการประหยัดเนื้อที่ อีกทั้งทำให้จำนวนกิ่งต่อโหนดมีมากขึ้นด้วย ตัวอย่างโครงสร้างต้นไม้บีแบบนำหน้าแสดงในรูปที่ 6



รูปที่ 6 ตัวอย่างโครงสร้างต้นไม้บีแบบนำหน้า [5]

เมื่อเปรียบเทียบกับการจัดเก็บด้วยโครงสร้างแถวเรียงลำดับแล้ว โครงสร้างต้นไม้บีใช้เนื้อที่มากกว่า และมีวิธีการที่ซับซ้อนกว่า แต่การเปลี่ยนแปลงดัชนีทำได้ง่ายกว่า และสามารถค้นหาคำได้เร็วกว่า

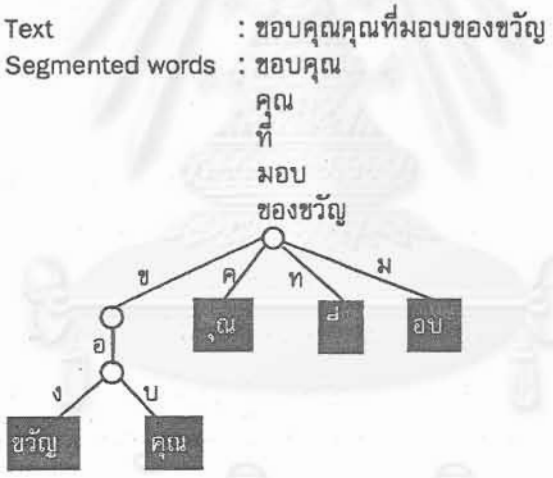
2.3 โครงสร้างทรัย

โครงสร้างทรัย (Trie structure) ถูกใช้เพื่อจัดเก็บดัชนีของข้อความจำนวนมาก [6],[12],[14] ทรัยก็คือต้นไม้เชิงเลข (digital tree) อย่างหนึ่งซึ่งเส้นเชื่อมต่างๆ แทนตัวอักษรต่างๆที่ใช้ในเข้ารหัสข้อมูล ดังนั้นคำๆ หนึ่งที่เก็บในทรัยถูกแทนด้วยเส้นเชื่อมต่างๆตามทางเดินทางหนึ่งจากรากถึงใบ ทรัยรองรับการค้นหาแบบเทียบชุดตัวอักษรนำหน้า (prefix searching) ได้ดี เนื่องจากคำต่างๆ ที่มีชุดตัวอักษรนำหน้าเหมือนกัน จะใช้ทางเดินจากรากถึงโหนดภายในโหนดหนึ่งร่วมกัน นอกจากนี้การค้นหาแบบประมาณ

(approximate search) ก็สามารถทำได้ดีในโครงสร้างทรัย ซึ่งเหมาะมากกับการค้นหาคำที่อาจสะกดผิดได้ [26] รายละเอียดวิธีการสร้างทรัยเพื่อจัดเก็บในหน่วยความจำสำรองนั้นหาได้ใน [14],[25]

รูปที่ 7 แสดงตัวอย่างของทรัยต้นหนึ่งที่เก็บคำต่างๆที่หาได้จากข้อความหนึ่ง ตำแหน่งของคำต่างๆ ในข้อความถูกจัดเก็บที่ใบต่างๆ ของทรัย เมื่อต้องการค้นคำใด ก็วิ่งไล่หาคำนั้นโดยใช้ตัวอักษรต่างๆ ของคำนั้นเป็นตัวกำหนดทางเดินจากรากลงไปตามกิ่งต่างๆ ตัวอย่างเช่นถ้าต้องการค้นหาคำว่า *ขอบคุณ* เราเริ่มจากรากลงไปที่กิ่ง *ข* ตามด้วยกิ่ง *อ* และกิ่ง *บ* ก็พบใบซึ่งมีข้อมูลตรงกับตัวอักษรที่เหลือ จึงสรุปได้ว่าประสิทธิภาพการค้นหาคำใด จะแปรตามจำนวนตัวอักษรของคำนั้น

กอนเน็ต [7] ได้นิยามสายอักษรกิ่งอนันต์ (semi-infinite string or sistring) หรือเรียกว่า ซิสตริง ให้เป็นส่วนหลังทั้งหมดของข้อความตั้งแต่ตำแหน่งที่กำหนด รูปที่ 8 แสดงซิสตริงสี่ตัวแรกของข้อความข้อความที่ยาว n มีซิสตริงเป็นจำนวนอย่างมาก n ถ้าซิสตริงทุกๆ ตัวถูกจัดเก็บในทรัย วิธีการค้นหาซิสตริงก็ เหมือนกับที่ได้กล่าวไว้ข้างต้น ถึงแม้ว่าการจัดเก็บซิสตริงทุกๆ ตัวของข้อความ จะให้ค่าเรียกคืนของการค้นคืนที่สูง แต่จะได้ความแม่นยำที่ต่ำ ตัวอย่างเช่นการค้นหาข้อความที่มีคำว่า *อบ* ก็จะได้ข้อความในรูปที่ 8 กลับคืนมาด้วย (ถึงแม้ว่าข้อความ *ขอบคุณคุณที่มอบของขวัญ* จะมีตัวอักษร *อบ* แต่ความจริงไม่มีคำว่า *อบ* ปรากฏอยู่ตามความหมายของรูปประโยค)



รูปที่ 7 ตัวอย่างโครงสร้างทรัย

Text : ขอบคุณคุณที่มอบของขวัญ
Sistrings : ขอบคุณคุณที่มอบของขวัญ
: อบคุณคุณที่มอบของขวัญ
: บคุณคุณที่มอบของขวัญ
: คุณคุณที่มอบของขวัญ

รูปที่ 8 ตัวอย่างซิสตริงสี่ตัวแรกของข้อความ *ขอบคุณคุณที่มอบของขวัญ*

3. การหาคำเพื่อจัดทำดัชนี

ในขณะที่เอกสารต่างๆในปัจจุบันเป็นเอกสารที่ถูกผลิตโดยใช้เครื่องคอมพิวเตอร์ และเครื่องคอมพิวเตอร์ทั้งหลายเชื่อมต่อสื่อสารกันได้ทั้งระยะใกล้และระยะไกล ระบบการค้นคืนข้อความจึงเป็นหนึ่งในเครื่องมือที่สำคัญมากในการจัดการสารสนเทศ โดยเฉพาะอย่างยิ่งกับสารสนเทศในเวิลด์ไวด์เวบ (World Wide Web) และในรูปแบบของซีดีรอมที่มีปริมาณสูง วิธีการค้นคืนข้อความแบ่งออกคร่าวๆ เป็นสี่วิธีคือ การกวาดดูทั้งแฟ้ม (full text scanning) การผกผัน (inversion) การใช้แฟ้มลายเซ็นเอกสาร (signature file) และการจัดกลุ่มเอกสาร (clustering) (รายละเอียดของวิธีต่างๆเหล่านี้หาอ่านได้ใน [4], [5], [23], [30]) การกวาดดูทั้งแฟ้ม ค้นหาเอกสารโดยการเปรียบเทียบข้อมูลในทุกๆเอกสารเพื่อค้นหาคำที่ต้องการ วิธีนี้ไม่ต้องใช้เนื้อที่เสริมในการจัดทำดัชนี แต่เวลาการค้นจะแปรผันโดยตรงตามขนาดของข้อความ วิธีการผกผันจัดเก็บคำสำคัญ (เรียงตามลำดับตัวอักษร) ที่พบในทุกๆเอกสารในแฟ้มดัชนี (ซึ่งสามารถใช้โครงสร้างแฟ้มข้อมูลแบบแถวเรียงลำดับ ต้นไม้บี โครงสร้างทรัย หรืออื่นๆได้หลายแบบ) คำสำคัญแต่ละคำจะมีรายการของตำแหน่งของเอกสารที่มีคำสำคัญนั้นๆปรากฏอยู่ วิธีนี้เป็นวิธีที่สร้างง่ายและค้นคืนได้รวดเร็ว แต่ต้องการเนื้อที่เสริมมากเพื่อจัดเก็บแฟ้มดัชนี และแฟ้มตำแหน่งเอกสาร วิธีการใช้แฟ้มลายเซ็นเอกสารนั้นใช้เทคนิคการเข้าและรวมรหัสคำต่างๆในเอกสาร (ผ่านฟังก์ชันแฮช) เพื่อแทนเอกสารต่างๆด้วยเลขฐานสองที่มีความยาว k โดย k มีค่าน้อยกว่าความยาวของเอกสารมาก เลขฐานสอง k บิตนี้เปรียบเสมือนลายเซ็นที่ใช้แทนเอกสารนั้น จากนั้นเก็บลายเซ็นของทุกๆเอกสารไว้ในแฟ้มลายเซ็น การค้นคืนจะกระทำที่แฟ้มลายเซ็นก่อน ซึ่งกระทำได้รวดเร็วกว่าที่แฟ้มเอกสารต้นฉบับมาก เนื่องจากขนาดเล็กกว่ามาก แต่อาจเกิดข้อผิดพลาดได้ (เรียกว่าเกิด false hit) สำหรับวิธีการจัดกลุ่มเอกสารนั้น ใช้หลักการการรวมกลุ่มเอกสารที่มีลักษณะคล้ายกันเข้ากันเป็นกลุ่มๆ การค้นคืนกระทำกับกลุ่มเอกสารแทนจึงรวดเร็ว อีกทั้งได้ผลดีได้เรื่องคุณภาพของการค้นคืน แต่จะมีปัญหาก็ตรงขั้นตอนในการจัดกลุ่มที่กระทำแบบอัตโนมัติได้ยาก

สำหรับเทคนิควิธีการผกผันนั้น ขั้นตอนที่สำคัญขั้นตอนหนึ่งระหว่างการจัดทำฐานข้อมูลสำหรับการค้นคืนข้อความ คือการหาคำสำคัญของเอกสารเพื่อจัดทำแฟ้มดัชนี วิธีหนึ่งในการหาคำสำคัญคือการให้คำทุกคำในเอกสารเป็นคำสำคัญ ที่ผู้ใช้สามารถค้นหาได้ แต่วิธีนี้อาจให้ค่าความแม่นยำของการค้นคืนที่ต่ำ การแยกคำออกจากข้อความให้ถูกต้องนั้นเป็นขั้นตอนที่ซับซ้อนขั้นตอนหนึ่งสำหรับข้อความภาษาไทย (และภาษาในทวีปเอเชียอื่นๆด้วย) เนื่องจากคำต่างๆที่ประกอบขึ้นเป็นข้อความในภาษาไทยนั้น เขียนติดกันหมด (ไม่เหมือนกับกรณีภาษาอังกฤษที่มีช่องว่างคั่นระหว่างคำ) ในปัจจุบันมีอยู่สองวิธีในการแยกคำ คือการใช้กฎ และการใช้พจนานุกรม วิธีการใช้กฎนั้นประกอบด้วยกฎต่างๆ ที่สามารถใช้เพื่อหาขอบเขตของพยางค์ในข้อความ [2],[27],[24] ตัวอย่างเช่น *มหาวิทยาลัย* จะถูกแบ่งเป็น *มหา-วิท-ยา-ลัย*

เป็นต้น ถึงแม้ว่าวิธีนี้จะได้ผลถูกต้องสูงมาก แต่การหาขอบเขตพยางค์นั้นไม่ตรงจุดประสงค์หลักของการหาคำเพื่อจัดทำดัชนี วิธีการใช้พจนานุกรมนั้นใช้การค้นคำต่างๆ ในพจนานุกรมมาประกอบกัน ให้ได้เป็นข้อความที่กำหนดให้ การเลือกคำต่างๆ นั้นมีด้วยกันหลายรูปแบบ เช่นการเลือกคำยาวสุดก่อน [18] การเลือกเพื่อให้ได้จำนวนคำน้อยสุด [28] หรือ การใช้สถิติของคำประกอบการพิจารณา [9] เป็นต้น ปัญหาการแยกคำออกจากข้อความจะซับซ้อนมากขึ้น เมื่อข้อความที่กำหนดให้มีคำที่ไม่ปรากฏในพจนานุกรมที่ใช้ ซึ่งอาจมีสาเหตุเนื่องมาจากว่าเป็นคำสะกดผิด เป็นคำเฉพาะ หรือเป็นคำทับศัพท์ เป็นต้น

บทนี้นำเสนอขั้นตอนวิธีการหาคำเพื่อจัดทำดัชนีสำหรับระบบค้นคืนข้อความไทยแบบอัตโนมัติ วิธีที่ออกแบบนี้ใช้พจนานุกรมช่วยในการแยกคำ แต่สามารถจัดการกับกรณีที่ข้อความที่ได้รับมีคำที่ไม่ปรากฏในพจนานุกรมโดยใช้กฎในการแยกพยางค์ ขั้นตอนวิธีการหาคำนี้ประกอบด้วยการสร้างกราฟการต่อและซ้อนกันของคำ ซึ่งโหนดต่างๆบนทางเดินสั้นสุดจากซ้ายไปขวาในกราฟนี้ คือรายการคำที่ควรถูกจัดทำดัชนีสำหรับแฟ้มผกผัน หัวข้อย่อยที่หนึ่งจะบรรยายขั้นตอนวิธีที่ออกแบบอย่างละเอียดสำหรับการหาคำเพื่อทำดัชนีสำหรับแฟ้มผกผันแบบทรัย หัวข้อที่สองนำเสนอการปรับปรุงขั้นตอนวิธีที่นำเสนอเพื่อให้ใช้ได้กับโครงสร้างแฟ้มผกผันแบบอื่นๆพร้อมผลการทดลอง โดยจะสรุปเนื้อหาของบทนี้ในหัวข้อย่อยที่สาม

3.1 ขั้นตอนวิธีการหาคำเพื่อทำดัชนีสำหรับแฟ้มผกผันแบบทรัย

ขั้นตอนวิธีการหาคำเพื่อทำดัชนีที่ออกแบบไว้นี้ ใช้พจนานุกรมช่วยในการหารายการคำในข้อความที่มีจำนวนคำที่น้อยสุดทั้งที่ปรากฏและไม่ปรากฏในพจนานุกรม คำที่ปรากฏในพจนานุกรมที่หาได้จะถูกจัดเก็บในทรัย ในขณะที่สตริงที่ไม่ปรากฏเป็นคำในพจนานุกรมจะถูกจัดเก็บเป็นซิสตริงทั้งหมดของสตริงนั้น โดยเราสามารถใช้กฎการแบ่งพยางค์ภาษาไทย (ในภาคผนวก ก) เป็นตัวกรองซีสตริงที่ไม่น่าเป็นจุดเริ่มต้นของคำออก ตัวอย่างเช่นซิสตริงของคำว่า "เจมส์มาร์ติน" มีทั้งหมด 12 ตัว (เท่ากับจำนวนตัวอักษร) ซีสตริง "ส์มาร์ติน", "์มาร์ติน", "ร์ติน", "ร์ติน", "์ติน" และ "ิน" เหล่านี้จะถูกตัดออกไม่นำไปเก็บในทรัย

ก่อนอื่นขอให้คำนิยามของสัญลักษณ์ต่างๆดังต่อไปนี้

- T คือข้อความที่กำหนดให้หาคำเพื่อทำดัชนี
- T_i คือซิสตริงของ T ที่เริ่มที่ตัวอักษรตัวที่ i
- T_{ij} คือสตริงย่อยของ T ที่ประกอบด้วยตัวอักษรตั้งแต่ตัวที่ i ถึงตัวที่ j
- i' คือตำแหน่งตัวอักษรขวาสุดของคำ w_i ในสตริง T ($i' = i-1+$ ความยาวของ w_i)
- D คือพจนานุกรมที่ใช้

ขั้นตอนวิธีการหาคำเพื่อทำดัชนีแบ่งออกเป็นสี่ขั้นตอนดังนี้

1. สำหรับแต่ละ T_i, $i = 1,...,n$, หาคำ w_i ใน D ที่ตรงตามเงื่อนไขต่อไปนี้

- w_i เป็นคำที่ยาวสุดใน D ที่ $w_i = T_{i,i'}$
- w_i ไม่ปรากฏเป็นสตริงย่อยใดๆใน w_j โดยที่ $j < i$

ซิสตริง T_i ใดที่หาคำใน D มาเทียบไม่ได้ ก็จะไม่มี w_i ตัวอย่างเช่น T = "นายเจมส์มาร์ตินต้องการผลิตรายการโทรทัศน์" จะได้ w_i ต่างๆดังแสดงในรูปที่ 9 w_i ต่างๆเหล่านี้ก็คือคำที่จะได้รับการพิจารณาเป็นคำสำคัญเพื่อจัดทำดัชนี โดยบางคำในนี้จะถูกตัดทิ้งในขั้นตอนต่อไป

i	T_i	w_i
1	นายเจมส์มาร์ตินต้องการผลิตรายการโทรทัศน์	นาย
4	เจมส์มาร์ตินต้องการผลิตรายการโทรทัศน์	เจ
5	จมส์มาร์ตินต้องการผลิตรายการโทรทัศน์	จม
9	มาร์ตินต้องการผลิตรายการโทรทัศน์	มาร
13	ตินต้องการผลิตรายการโทรทัศน์	ติ
16	ต้องการผลิตรายการโทรทัศน์	ต้องการ
20	การผลิตรายการโทรทัศน์	การผลิต
24	ลิตรายการโทรทัศน์	ลิตร
26	ตรายการโทรทัศน์	ตรา
27	รายการโทรทัศน์	รายการ
33	โทรทัศน์	โทรทัศน์

รูปที่ 9 ตัวอย่างคำต่างๆที่ได้ในขั้นตอนที่หนึ่ง

2. สร้างกราฟการต่อและซ้อนกันของคำ (ซึ่งเป็นกราฟที่มีน้ำหนักแบบมีทิศทาง) G = (V,E) โดยที่

- $V = \{w_i \mid w_i$ หาได้ขั้นตอนที่ 1 $\}$
- $E = \{ (w_i, w_j) \mid w_i$ เป็นคำที่ต่อกันหรือซ้อนกันบางส่วนกับ w_j ในข้อความ, โดยที่ $i < j \}$ น้ำหนักต่างๆของเส้นเชื่อม (w_i, w_j) กำหนดจากกรณีต่างๆในตารางที่ 1 ดังนี้

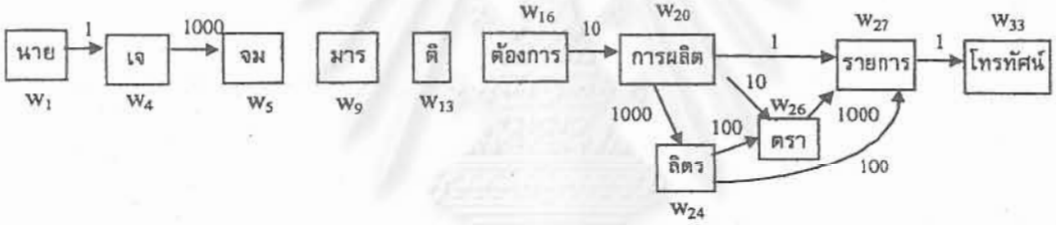
ตารางที่ 1 การกำหนดน้ำหนักของเส้นเชื่อม (w_i, w_j)

#	น้ำหนัก of (w_i, w_j)	เงื่อนไข
1	1	ถ้า $i'+1 = j$, นั่นคือเมื่อ w_i ต่อติดกับ w_j
2	10	ถ้า $j \le i'$ และ ทั้ง $T_{i,j-1}$ และ $T_{i'+1,j}$ ปรากฏในพจานุกรม.
3	100	ถ้า $j \le i'$ และ มีบางตำแหน่ง k, $j-1 \le k \le i'$ ที่ทำให้ทั้ง $T_{i,k}$ และ $T_{k+1,j'}$ ปรากฏในพจนานุกรม
4	1000	กรณีอื่นๆที่ไม่ปรากฏในทั้ง 3 กรณีข้างบนนี้

- ค่า i' คือตำแหน่งตัวอักษรขวาสุดของคำ w_i ในสตริง T
- ตารางนี้ใช้กับคู่คำที่ต่อกันหรือซ้อนกันบางส่วนกับ w_j ในข้อความเท่านั้น

กรณีที่ 1 เป็นกรณีที่ต้องการให้เกิดขึ้นมากที่สุด นั่นคือเมื่อคำสองคำต่อติดกันในข้อความ กรณีที่ 2 เกิดขึ้นเมื่อคำสองคำมีบางส่วนซ้อนกันในข้อความ ซึ่งส่วนที่ไม่ซ้อนกันนั้นต่างก็เป็นคำที่ปรากฏในพจนานุกรมทั้งสิ้น ตัวอย่างเช่นส่วนของข้อความ *ต้องการผลิต* มี w_1 = *ต้องการ* และ w_4 = *การผลิต* ซึ่งสามารถแยกเป็น *ต้อง+การผลิต* หรือ *ต้องการ+ผลิต* กรณีที่ 3 เป็นกรณีที่มีการซ้อนทับกันของคำทั้งสองในข้อความ โดยส่วนที่ไม่ซ้อนกันไม่ปรากฏในพจนานุกรม แต่เราสามารถหาคำในพจนานุกรมที่มีความยาวสั้นกว่า มาประกอบกันได้เหมือนข้อความย่อยนั้น ตัวอย่างเช่นข้อความย่อย *เพื่อนำ* มี w_1 = *เพื่อน* และ w_5 = *อนำ* สามารถแยกคำได้เป็น *เพื่อน+ำ* หรือ *เพื่ + อนำ* ซึ่ง *ำ* และ *เพื่* ไม่ใช่คำในพจนานุกรม แต่ข้อความย่อยนี้แยกคำได้เป็น *เพื่อ+นำ* กรณีสุดท้ายซึ่งมีน้ำหนักมากสุด (หมายความว่าเป็นกรณีที่สร้างปัญหามากสุด) แทนสถานะการณ์เมื่อการซ้อนกันของคำสองคำไม่สามารถแยกคำได้ตามคำในพจนานุกรม กรณีที่ 3 และ 4 นั้นเป็นกรณีที่อาจเกิดขึ้นเมื่อมีการสะกดคำผิดได้ ตัวอย่างเช่นข้อความย่อย *เพื่อนำ* นั้นโดยความเป็นจริงแล้วอาจเป็น *เพื่อนทำ* ก็ได้ เนื่องจากพิมพ์ *ท* ตกไป ดังนั้นคำว่า *เพื่อน* จะต้องถูกจัดเก็บเป็นดัชนีด้วยนอกจากคำว่า *เพื่อ* และ *นำ*.

จากตัวอย่างที่แสดงในขั้นตอนที่ 1 เราสามารถสร้างกราฟการต่อและซ้อนกันของคำได้ดังแสดงในรูปที่ 10



รูปที่ 10 กราฟการต่อและซ้อนกันสำหรับข้อความ *นายเจมส์มาร์ตินต้องการผลิตรายการโทรทัศน์*

ให้สังเกตว่ากราฟนี้อาจมีหลายองค์ประกอบ เนื่องจากคำที่หาได้บางคู่อาจไม่ต่อหรือซ้อนกัน ดังตัวอย่างที่แสดงข้างบนนี้ เป็นกราฟที่มีสามองค์ประกอบ

3. สำหรับแต่ละองค์ประกอบของกราฟที่สร้างขึ้น ให้หาเส้นทางสั้นสุดจากโหนดซ้ายสุดไปยังโหนดขวาสุด (กำหนดให้วาดกราฟเพื่อให้โหนดของ w_i อยู่ทางซ้ายของโหนดของ w_j เมื่อ $i<j$) กำหนดให้ $W = \{w_i \mid w_i$ มีโหนดของมันอยู่บนเส้นทางสั้นสุดของทุกๆองค์ประกอบที่หาได้ $\}$ จากตัวอย่างเดิมข้างต้นจะได้ $W = \{w_1, w_4, w_5, w_9, w_{13}, w_{16}, w_{20}, w_{27}, w_{33}\}$

4. ในขั้นตอนนี้ เราจะหา W' ซึ่งคือเซ็ตของคำที่แยกได้ และ U' ซึ่งคือเซ็ตของซิสตริงสำหรับสตริงย่อยที่ไม่ปรากฏในพจนานุกรม ดังนี้

4.1 กำหนดให้ U คือเซ็ตของสตริงย่อยของ T ที่มีคำที่ไม่ปรากฏในพจนานุกรม ซึ่งหาได้ดังนี้

4.1.1 สำหรับแต่ละเส้นเชื่อม (w_i, w_j) ที่มีน้ำหนัก 1000 ที่เป็นส่วนหนึ่งของเส้นทางสั้นสุด เราเพิ่ม $T_{i,j-1}$ และ $T_{i'+1,j'}$ เข้าใน U จากตัวอย่างข้างต้นจะได้ U = {$T_{4,4}$, $T_{6,6}$ } ซึ่งคือสตริง เ และ ม จากเส้นเชื่อม (w_4, w_5)

4.1.2 สำหรับแต่ละคู่โหนด w_i และ w_j ที่อยู่ต่างองค์ประกอบกันของกราฟ G โดยที่ i < j, และไม่มีคำ $w_k \in W$, โดยที่ i < k < j, เราเพิ่ม $T_{i'+1,j-1}$ เข้าใน U จากตัวอย่างข้างต้นจะได้สตริง ส์ ์ และ น ทำให้ U = { $T_{4,4}$, $T_{6,6}$, $T_{7,8}$, $T_{10,10}$, $T_{15,15}$ }

4.1.3 เพิ่ม w_i เข้าใน U เมื่อ $w_i \in W$ ที่ต่อกับทางซ้ายของบางสตริงของ U ในข้อความ จากตัวอย่างข้างต้นจะได้ U = { $T_{4,4}$, $T_{6,6}$, $T_{7,8}$, $T_{10,10}$, $T_{15,15}$, w_1 , w_4 , w_5 , w_9 , w_{13} } ซึ่งคือสตริง เ, ม, ส์, ์, น, นาย, เจ, จม, มาร, ติ

4.1.4 รวมสตริงต่างๆใน U ที่ต่อกันหรือซ้อนกันเข้าด้วยกัน จากตัวอย่างจะได้ U = { $T_{1,15}$ } ซึ่งคือสตริง นายเจมส์มาร์ติน

หลังจากได้ U แล้ว เซ็ต U′ ก็จะประกอบด้วยซิสตริง T_i , T_{i+1},..., T_k ของสตริง $T_{i,k} \in U$ เซ็ต U′ นี้เก็บซิสตริงที่มีคำที่ไม่ปรากฏในพจนานุกรมของ T เป็นส่วนประกอบ เพื่อให้ระบบสามารถค้นหาคำดังกล่าวได้ เนื่องจากเราไม่ทราบตำแหน่งเริ่มต้นของคำเหล่าดังนั้นซิสตริงทั้งหมดของสตริงใน U จึงต้องถูกจัดเก็บในทรัย (ความจริงแล้วเราสามารถใช้กฎการแบ่งพยางค์เพื่อกำจัดบางซิสตริงออกได้ ตัวอย่างเช่น คงไม่ต้องเก็บซิสตริงใดที่ขึ้นต้นด้วยการันต์ เป็นต้น)

4.2 กำหนดให้ W′ = W − {w_i | w_i เป็นคำที่ถูกใช้ในการพิจารณาในขั้นตอนที่ 4.1.3 พิจารณาคำสองคำ w_i และ w_j ใดๆ ใน W′ ที่

- มีคุณสมบัติตามกรณีที่ 2 ในตารางที่ 1 เราเพิ่ม $T_{i,j-1}$ and $T_{i'+1,j'}$ เข้าใน W′ จากตัวอย่างข้างต้นจะได้คำว่า *ต้อง* และ *ผลิต* จาก w_{16} และ w_{20}.
- มีคุณสมบัติตามกรณีที่ 3 ในตารางที่ 1 เราเพิ่ม $T_{i,k}$, and $T_{k+1,j'}$ เข้าใน W′ โดยที่ k เป็นค่าที่นิยามในตาราง

ผลลัพธ์ที่ได้จากขั้นตอนวิธีที่ได้นำเสนอมานี้ คือเซ็ต W′ ซึ่งคือเซ็ตของคำที่แยกได้ และ U′ ซึ่งคือเซ็ตของซิสตริงสำหรับสตริงย่อยที่ไม่ปรากฏในพจนานุกรม จากตัวอย่างข้อความ T = *นายเจมส์มาร์ตินต้องการผลิตรายการโทรทัศน์*, เราจะได้ W′ = {ต้องการ, การผลิต, รายการ, โทรทัศน์, ต้อง, ผลิต} [1] และ U′ = { T_i | T_i คือซิสตริงของสตริง "*นายเจมส์มาร์ติน*" ใน T } เพื่อจัดเก็บเป็นดัชนีในโครงสร้างทรัย.

---

[1] ในบางครั้งจะขอเขียนเซ็ตในรูปแบบที่แจงตัวสตริง แทนที่ใช้สัญลักษณ์อย่างเป็นทางการ เนื่องจากจะเข้าใจได้ง่ายกว่า

ประสิทธิภาพเชิงเวลาของขั้นตอนวิธีที่กล่าวมาเป็นดังนี้ ขั้นตอนที่ 1 เปรียบเทียบชิสตริงแต่ละตัวกับคำในพจนานุกรมเพื่อหาคำยาวสุดที่เทียบได้ กำหนดให้ k คือคำที่ยาวที่สุดในพจนานุกรม เนื่องจากชิสตริง n ตัวใน T ที่ยาว n ดังนั้นจึงใช้เวลาเป็น $O(nk)$ (เพราะพจนานุกรมที่ใช้เก็บในโครงสร้างแบบทรัยการค้นหาคำจะแปรตาม k) ขั้นตอนที่ 2 สร้างกราฟซึ่งใช้เวลาเป็น $O(n^2)$ คิดจากกรณีเลวสุดที่คำทุกๆ คำซ้อนกันกับคำทุกๆคำในข้อความ ขั้นตอนที่ 3 เป็นขั้นตอนหาเส้นทางสั้นสุดของทุกองค์ประกอบซึ่งใช้เวลา $O(n^2)$ และในขั้นตอนสุดท้ายเป็นการวิ่งไล่ในข้อความอีกครั้งเพื่อหาเซ็ตผลลัพธ์ ใช้เวลา $O(n)$ ดังนั้นใช้เวลาทั้งหมดเป็น $O(nk + n^2)$ หรือ $O(n^2)$ เมื่อ $k < n$ สำหรับประโยคยาว

3.2 ขั้นตอนวิธีการหาคำเพื่อทำดัชนีสำหรับแฟ้มผกผันแบบอื่น

สำหรับแฟ้มผกผันแบบอื่นๆ ที่ไม่ใช่ทรัยนั้น สิ่งที่จัดเก็บต้องเป็นคำหรือสตริง เนื่องจากแฟ้มผกผันแบบอื่นไม่รองรับการจัดเก็บชิสตริง ซึ่งสามารถปรับปรุงขั้นตอนวิธีที่นำเสนอในหัวข้อที่แล้ว เพื่อให้ได้ผลลัพธ์เป็นคำและสตริงย่อยของข้อความเท่านั้น โดยขั้นตอนวิธีนี้จะประกอบด้วย 4 ขั้นตอน ซึ่งมี 3 ขั้นตอนแรกเหมือนกันขั้นตอนวิธีที่นำเสนอในหัวข้อที่แล้ว ส่วนขั้นตอนที่ 4 มีกระบวนการทำงานดังนี้ (กำหนดให้ผลลัพธ์ของขั้นตอนนี้คือ W' ซึ่งคือเซตของคำที่แบ่งได้จากพจนานุกรม และ U' ซึ่งเซตของคำที่ไม่ปรากฏในพจนานุกรมที่หาได้จากกฎการแบ่งพยางค์)

4.1 กำหนดให้ U คือเซ็ตของสตริงย่อยของ T ที่มีคำที่ไม่ปรากฏในพจนานุกรม ซึ่งหาได้ดังนี้ (ขั้นตอนที่ 4.1.1 และ 4.1.2 เหมือนขั้นตอนข้อ 4.1.1 และ 4.12 ในหัวข้อที่แล้ว)

4.1.1 สำหรับแต่ละเส้นเชื่อม (w_i, w_j) ที่มีน้ำหนัก 1000 ที่เป็นส่วนหนึ่งของเส้นทางสั้นสุด เราเพิ่ม $T_{i,j-1}$ และ $T_{i'+1,j'}$ เข้าใน U จากตัวอย่างข้างต้นจะได้ $U = \{T_{4,4}, T_{6,6}\}$ ซึ่งคือสตริง เ และ ม จากเส้นเชื่อม (w_4, w_5)

4.1.2 สำหรับแต่ละคู่โหนด w_i และ w_j ที่อยู่ต่างองค์ประกอบกันของกราฟ G โดยที่ $i < j$, และไม่มีคำ $w_k \in W$, โดยที่ $i < k < j$, เราเพิ่ม $T_{i'+1,j-1}$ เข้าใน U จากตัวอย่างข้างต้นจะได้สตริง ส์ ์ และ น ทำให้ $U = \{T_{4,4}, T_{6,6}, T_{7,8}, T_{10,10}, T_{15,15}\}$

4.1.2 ถ้า $U \neq \varnothing$ นำ T ไปแบ่งพยางค์โดยใช้กฎ (กฎการแบ่งพยางค์ที่ใช้ [27] แสดงในภาคผนวก ก) กำหนดให้ X คือเซตของพยางค์ที่ได้จากการแบ่งพยางค์โดยใช้กฎ และ x_i คือพยางค์ที่มีตำแหน่งเริ่มต้นที่ตำแหน่ง i ตัวอย่างข้างต้นจะได้ X = {นาย, เจมส์, มาร์, ติน, ต้อง, การ, ผลิต, ราย, การ, โทร, ทัศน์} พิจารณาแต่ละ $u_i \in U$ ถ้า u_i เป็นส่วนหนึ่งส่วนใดของ x_j แล้ว ให้เพิ่ม x_j เข้าใน U' จากตัวอย่างข้างต้นจะได้ U' = {เจมส์, มาร์, ติน}

4.2 กำหนดให้ $W' = W - \{w_i \mid w_i \in W$ และ w_i ไม่เป็นส่วนหนึ่งส่วนใดของ $x_j \in U'\}$ จากตัวอย่างข้างต้น W = {นาย, เจ, จม, มาร, ติ, ต้องการ, การผลิต, รายการ, โทรทัศน์} จะได้ W' =

{นาย, ต้องการ, การผลิต, รายการ, โทรทัศน์} เนื่องจาก เจ และ จม เป็นส่วนหนึ่งของ *เจมส์*, *มาร* เป็นส่วนหนึ่งของ *มาร์* และ *ติ* เป็นส่วนหนึ่งของ *ติน* พิจารณาคำสองคำ w_i และ w_j ใดๆ ใน W' ที่

- มีคุณสมบัติตามกรณีที่ 2 ในตารางที่ 1 เราเพิ่ม $T_{i,j-1}$ and $T_{i'+1,j'}$ เข้าใน W' จากตัวอย่างข้างต้นจะได้คำว่า *ต้อง* และ *ผลิต* จาก w_{16} และ w_{20}.
- มีคุณสมบัติตามกรณีที่ 3 ในตารางที่ 1 เราเพิ่ม $T_{i,k}$, and $T_{k+1,j'}$ เข้าใน W' โดยที่ k เป็นค่าที่นิยามในตาราง

จากตัวอย่างข้างต้นจะได้ W' = { นาย, ต้องการ, การผลิต, รายการ, โทรทัศน์, ต้อง, ผลิต } และ U' = { เจมส์, มาร์, ติน }

ผลลัพธ์ที่ได้คือ W' คือเซตของคำที่หาได้จากพจนานุกรม และ U' คือเซตของคำที่ไม่ปรากฏในพจนานุกรมที่หาได้จากกฎการแบ่งพยางค์ สามารถนำทั้ง W' และ U' ไปจัดเก็บเพื่อทำดัชนีด้วยแฟ้มผกผันทั่วไป จะเห็นได้ว่าคำว่า *มาร์ติน* ซึ่งเป็นชื่อเฉพาะทับศัพท์นั้น ถูกแยกเป็นสองคำ ดังนั้นในกระบวนการค้นคืนนั้น จะต้องมีการค้นหาคำว่า *มาร์* และคำว่า *ติน* แล้วนำผลลัพธ์ที่ได้จากการค้นทั้งสองครั้งมากตรวจสอบผลลัพธ์ที่ได้อีกทีหนึ่ง ซึ่งจะใช้เวลามากขึ้น

พิจารณาตัวอย่างต่อไปนี้

1. ตากลมอบบอกไก่

W = { ตาก, กลม, มอบ, บอก, ไก่ }
U = { }
W' = { ตาก, กลม, มอบ, บอก, ไก่, ตา, ลม, กล, อบ, มอ, อก }
U' = { }

2. เขาได้ตำแหน่งท็อปชั้น

W = { เขา, ได้, ตำแหน่ง, ชั้น }
U = { ท็อป }
W' = { เขา, ได้, ตำแหน่ง, ชั้น }
U' = { ท็อป }

3. เขาได้อันดับท็อปชั้น

W = { เขา, ได้, อันดับ, บท, ชั้น }

U = { ็อป }

W′ = { เขา, ได้, อันดับ, ชั้น }

U′ = { ท็อป }

3.3 ผลการทดลอง

การวัดผลการทำงานของขั้นตอนวิธีการหาคำเพื่อทำดัชนีที่ออกแบบขึ้นนั้น ใช้ข้อมูลในการทดลองอยู่ 4 ประเภท ดังนี้

1. โคลงกลอน ลักษณะข้อมูลประเภทนี้จะมีคำที่เกิดจากการแผลงคำ การตัดคำ (ลดรูป) การเติมคำ ฯลฯ
2. ข่าว โดยเฉพาะข่าวต่างประเทศ จะมีคำที่ทับศัพท์ คำที่เป็นชื่อคน ชื่อสถานที่ต่างๆ ที่ไม่มีอยู่ในพจนานุกรม
3. เนื้อเพลง คำต่างๆ ในเนื้อเพลงมักจะเป็นคำที่มีอยู่ในพจนานุกรม
4. ข้อสอบเข้ามหาวิทยาลัยสายวิทยาศาสตร์ มีคำศัพท์เฉพาะ ศัพท์ทางวิทยาศาสตร์ รวมถึงวิชาภาษาไทย จะมีคำศัพท์ที่ไม่มีในพจนานุกรม

ขนาดของข้อมูลในแต่ละประเภทที่ใช้ในการทดสอบ มีขนาดดังแสดงในตารางที่ 2

ตารางที่ 2 ขนาดของข้อมูลประเภทต่างๆที่ใช้ในการทดสอบ

ประเภทของข้อมูล	*ขนาดของข้อมูล (Bytes)*
Poem	1,950,970
News	1,572,730
Lyric	1,626,092
Entrance	2,360,350

แต่ละประเภทของข้อมูลที่นำมาทดสองนั้น จะมีรูปแบบของคำแตกต่างกัน โดยได้ผลดังนี้

1. สัดส่วนจำนวนคำที่ได้ในแต่ละขั้นตอน เมื่อเทียบกับจำนวนคำทั้งหมดที่ปรากฏในข้อความ แสดงในตารางที่ 3

ตารางที่ 3 เปอร์เซนต์จำนวนคำที่ได้หลังทำงานในแต่ละขั้นตอนเทียบกับจำนวนคำทั้งหมด

ประเภทข้อมูล	*ขั้นตอนที่*			
	1	2	3	4
Poem	41.01	35.76	36.40	35.99
News	38.10	33.75	34.62	33.70
Lyric	50.96	46.62	47.55	46.98
Entrance	41.41	38.62	39.37	37.61

2. สัดส่วนค่าน้ำหนักของเส้นเชื่อมที่ได้ของแต่ละประเภท ในขั้นตอนที่ 2 แสดงในตารางที่ 4 (กรณีที่ 5 นั้นคือกรณีที่กราฟแยกออกเป็นหลายองค์ประกอบ)

ตารางที่ 4 เปอร์เซนต์การกระจายของน้ำหนักเส้นเชื่อมในกรณีต่างๆ

ประเภทข้อมูล	*กรณีที่*				
	1	2	3	4	5
Poem	69.59	0.66	9.39	19.85	0.51
News	67.07	0.72	8.54	22.76	0.91
Lyric	72.42	0.35	6.48	20.08	0.66
Entrance	65.46	0.41	8.22	24.27	1.64

ตารางที่ 3 แสดงให้เห็นจำนวนที่ได้จากขั้นตอนที่ 1 มีประมาณ 40-50% เมื่อเทียบกับจำนวนคำทั้งหมดในข้อความ จากนั้นจะได้จำนวนคำที่ลดลงจากที่หาได้เฉลี่ยประมาณ 30-46% จะสังเกตได้ว่าข้อมูลประเภทเนื้อเพลงนั้นมีสัดส่วนของคำที่หาได้มากสุด ทั้งนี้เนื่องจากคำต่างๆที่ปรากฏในเนื้อเพลงส่วนใหญ่จะเป็นคำที่มีในพจนานุกรม ส่วนตารางที่ 4 นั้นแสดงให้เห็นว่า จากคำต่างๆที่ได้ในขั้นตอนที่ 1 นั้นประมาณ 70% เป็นคำที่ต่อติดกันในข้อความ (กรณีที่ 1) หรือกล่าวได้ว่าสามารถหารอยแยกของข้อความเพื่อแบ่งคำได้กว่า 70% จะมีประมาณ 10% เป็นกรณีที่ไม่สามารถหารอยแยกถ้าใช้คำยาวที่หาได้ แต่ถ้าใช้คำอื่นที่สั้นกว่าในพจนานุกรมจะแยกคำได้ (กรณีที่ 3) และมีอีกประมาณ 20% ที่คำที่หาได้ที่ตัวอักษรซ้อนกันและไม่สามารถหารอยแยกได้ (กรณีที่ 4) สำหรับกรณีที่ 2 และ 5 นั้นเกิดขึ้นน้อยมาก

ผลที่สังเกตได้อีกประการหนึ่งคือเอกสารที่ใช้ทดสอบนั้น ในทางปฏิบัติแล้วมีคำที่สะกดผิด เป็นคำทับศัพท์ หรือเป็นชื่อเฉพาะ ซึ่งล้วนแล้วแต่เป็นคำที่ไม่ปรากฏในพจนานุกรมทั้งสิ้น ยังผลให้เกิดกรณีที่ 3 และ 4 รวมเป็นสัดส่วนที่สูงถึง 30% จากผลการทดลอง

ขั้นตอนวิธีการหาคำเพื่อทำดัชนีที่ได้ออกแบบมานี้ได้ถูกทดลองรวมเข้ากับระบบการค้นคืนเอกสาร เพื่อใช้งานให้บริการข้อมูลบนเวิลด์ไวด์เวบอีกด้วย รายละเอียดการรวมและผลที่ได้รับถูกนำเสนอในภาคผนวก ข

3.4 สรุป

บทนี้นำเสนอขั้นตอนวิธีการหาคำสำคัญเพื่อใช้จัดทำดัชนีสำหรับโครงสร้างแฟ้มผกผัน ทั้งสำหรับโครงสร้างทรัย ซึ่งรองรับการจัดเก็บและค้นซิสตริง และแบบอื่นๆ ที่รองรับการเก็บข้อมูลเป็นคำๆ เท่านั้น ขั้นตอนวิธีที่นำเสนอนี้ใช้พจนานุกรมเป็นฐานความรู้หลักในการหาคำ และใช้กฎการแบ่งพยางค์เป็นแนวทางในการแยกคำ เมื่อพบคำที่ไม่ปรากฏในพจนานุกรม ขั้นตอนวิธีนี้ใช้เวลาทำงานเป็น $O(nk)$

4. การเข้ารหัสเสียงของคำข้ามภาษาอังกฤษ-ไทย

ปัญหาหนึ่งที่เกิดขึ้นในระบบค้นคืนข้อความคือ เมื่อผู้ใช้ป้อนคำเพื่อการค้น โดยใช้คนละภาษากับคำในเอกสารที่จัดเก็บ ตัวอย่างเช่นการค้นคืนเอกสารที่มีคำว่า "*ALEXANDER*" จะไม่ได้เอกสารที่มีคำว่า "*อเล็กซานเดอร์*" คำต่างๆทางด้านเทคนิคที่ปรากฏในเอกสารภาษาไทยนั้น บางครั้งก็ใช้คำทับศัพท์ของคำภาษาอังกฤษ หรือบางครั้งก็ใช้คำภาษาอังกฤษเลย การค้นคืนข้ามภาษาคือการค้นคืนเอกสารที่ภาษาที่ใช้ในเอกสารไม่เหมือนกับภาษาที่ผู้ใช้ตั้งข้อคำถามสืบค้น [15] การใช้พจนานุกรมสองภาษาในระบบค้นคืนคงไม่สามารถแก้ปัญหานี้ได้หมด เนื่องจากคำทับศัพท์ส่วนใหญ่ไม่ปรากฏในพจนานุกรม [10] ปัญหาในลักษณะนี้จะลดค่าเรียกคืนของระบบการค้นคืน

มีงานวิจัยที่แก้ปัญหาในลักษณะนี้อยู่หลายงาน ขั้นตอนวิธีที่นำเสนอใน [11] และ [13] แปลงคำภาษาอังกฤษและภาษาญี่ปุ่นไปเป็นรหัสกลางเพื่อจัดเก็บและใช้การเปรียบเทียบรหัส ระหว่างการค้นคืนลักษณะการแปลงคำเป็นรหัสที่ว่านี้ หากแปลงไปกลับระหว่างสองภาษาจะเกิดการสูญเสียรูปคำเดิม ทำให้ในบางครั้งรหัสของคำมาจากการทับศัพท์ จะไม่เหมือนกัน ในขณะที่ขั้นตอนวิธีของ [3] เข้ารหัสคำคาตะคานะ (Katakana) โดยใช้รหัสการออกเสียง และใช้การเปรียบเทียบบางส่วนกับคำภาษาอังกฤษ (ที่ไม่ต้องเข้ารหัส) แต่ขั้นตอนวิธีที่นำเสนอนั้นใช้เทคนิคการค้นตามแนวลึกซึ่งใช้เวลานานกว่าแบบเปรียบเทียบตรงๆ จึงต้องมีการเพิ่มแนวคิดฮิวรีสติกบางอย่างเพื่อลดเวลาการค้น งานวิจัยใน [16] นำเสนอขั้นตอนวิธีในการเข้ารหัสคำภาษาอังกฤษไปยังเซ็ตของเสียงภาษาไทยโดยใช้ตารางการเข้ารหัสและกฎต่างๆ ผู้วิจัยไม่ได้นำเสนออย่างละเอียด อีกทั้งไม่ได้ประเมินผลของวิธีดังกล่าว จึงไม่สามารถประเมินข้อดีข้อเสียของวิธีดังกล่าวได้

ในบทนี้เรานำเสนอขั้นตอนวิธีในการเข้ารหัสคำทับศัพท์ เพื่อรองรับการค้นคืนคำทับศัพท์ข้ามภาษาจากอังกฤษมาไทย นั่นคือระบบสามารถค้นคืนเอกสารที่มีคำสำคัญภาษาอังกฤษ หรือคำทับศัพท์เป็นภาษาไทยของคำอังกกฤษนั้น การเข้ารหัสนี้ปรับปรุงวิธีการเข้ารหัสเสียงและตารางการเข้ารหัสของโอเดลและรัสเซล [1] ทำให้ได้ค่าเรียกคืนสูงในการค้นคืนข้ามภาษา ภายใต้ค่าความแม่นยำที่ดี เราจะนำเสนอขั้นตอนวิธีการเข้ารหัสของโอเดลและรัสเซลในหัวข้อย่อยที่ 1 จากนั้นขั้นตอนวิธีที่ออกแบบขึ้นจะถูกนำเสนออย่างละเอียดในหัวข้อย่อยที่ 2 ผลการทดลองแสดงในหัวข้อย่อยที่สาม และสรุปเนื้อหาของบทนี้ในหัวข้อย่อยที่ 4

4.1 ขั้นตอนวิธีการเข้ารหัสเสียงของโอเดลและรัสเซล

โอเดลและรัสเซส (M. K. Odell and R. C. Russell) ได้ออกแบบระบบในการเข้ารหัสชื่อ โดยใช้การออกเสียงเป็นเกณฑ์ เพื่อให้ชื่อที่ออกเสียงคล้ายกันมีรหัสเดียวกัน ระบบนี้เรียกว่าซาวน์เดกซ์ (Soundex) [1] ระบบนี้ใช้หลักการที่ว่า ชื่อเฉพาะภาษาอังกฤษนั้นสามารถแยกความแตกต่างได้โดยพิจารณาเฉพาะตัวพยัญชนะในชื่อเท่านั้น วิธีการเข้ารหัสการออกเสียงกระทำโดยการแปลงตัวอักษรแต่ละตัว (ไม่รวมตัวซ้ายสุด) ไปเป็นรหัสตัวเลขโดยใช้ตารางเข้ารหัสแสดงในตารางที่ 5 จากนั้นลบศูนย์ทุกตัวในรหัสที่ได้ออกหมด พร้อมทั้งรหัสที่ติดกันที่เหมือนกัน ให้ลดเหลือเพียงตัวเดียว รหัสซาวน์เด็กซ์คือตัวอักษรตัวซ้ายสุดของคำตามด้วยรหัสที่แปลงได้สามตัวแรก ตัวอย่างเช่น ALEXANDER มีรหัสซาวน์เดกซ์คือ A425

ตารางที่ 5 ตารางการเข้ารหัสซาวน์เดกซ์

ตัวอักษร	*รหัสตัวเลข*
A E I O U H W Y	0
B F P V	1
C G J K Q S X Z	2
D T	3
L	4
M N	5
R	6

การเข้ารหัสแบบนี้ทำงานเร็วและโดยทั่วไปให้ค่าเรียกคืนที่สูงมาก แต่จะได้ค่าความแม่นยำที่ต่ำ

4.2 การเข้ารหัสคำทับศัพท์อังกฤษ-ไทย

พยัญชนะไทยมี 44 ตัว สามารถจำแนกตามเสียงได้ 21 กลุ่มดังแสดงในตารางที่ 6 [17] การปรับปรุงตารางซาวน์เดกซ์เพื่อรองรับภาษาไทยทำได้โดย จับกลุ่มเสียง 21 กลุ่มที่ออกเสียงคล้ายกลุ่มเสียง 7 กลุ่มของตารางซาวน์เดกซ์ ได้ดังแสดงในตารางที่ 7 จากนั้นปรับปรุงตารางการเข้ารหัสเสียงได้ดังตารางที่ 8 ดังนี้

- ใช้ตัวเลขเป็นรหัสสำหรับตัวอักษรตัวแรก ทั้งนี้เนื่องจากมีหลายกรณีที่ตัวอักษรภาษาอังกฤษหลายตัวถูกแปลงเป็นตัวอักษรภาษาไทยตัวเดียวกันในการทับศัพท์ [19] เช่น V และ W แปลงเป็น ว เป็นต้น

- เพิ่มรหัสเสียงอีกสามเสียง 7, 8, 9 สำหรับเสียงในกลุ่ม 0 เดิม กรณีที่ปรากฏในตำแหน่งที่ไม่ใช่ตำแหน่งซ้ายสุด ทั้งนี้เนื่องจากการแปลงคำอังกฤษมาไทยนั้นกลุ่มตัวอักษรเหล่านี้ มีผลต่อการทับศัพท์ในคำไทย

- เพิ่มรหัสเสียงสำหรับตัว ง ให้เป็นเลขสองหลักคือ 52 เนื่องจากตัว ง ใช้แทนการออกเสียง NG หรือ NK

- ขยายความยาวของรหัสแบบไม่จำกัด แทนที่จะใช้เพียงสี่หลัก เนื่องจากต้องการเพิ่มค่าความแม่นยำในการค้นคืน ซึ่งจะลดค่าเรียกคืนลงบ้าง และจะจำกัดให้ความยาวน้อยสุดของรหัสให้เป็น k นั่นคือจะพิจารณาเฉพาะคำที่รหัสมีความยาวไม่น้อยกว่า k ค่าของ k จะเป็นพารามิเตอร์ที่หาได้จากการทดลองที่จะนำเสนอในภายหลัง

ให้สังเกตว่าสระและวรรณยุกต์จะไม่ถูกนำมาพิจารณาในการเข้ารหัสแต่อย่างใด เนื่องจากในงานนี้เราสนใจเฉพาะคำทับศัพท์ภาษาอังกฤษมายังภาษาไทยเท่านั้น

ตารางที่ 6 กลุ่มเสียง 21 กลุ่มของพยัญชนะไทย

ก	ฎ ด	ฝ ฟ
ข ฃ ค ฅ ฆ	ฏ ต	ม
ง	ฐ ฑ ฒ ถ ท ธ	ร
จ	ณ น	ล ฬ
ฉ ช ฌ	บ	ว
ซ ศ ษ ส	ป	ห ฮ
ญ ย	ผ พ ภ	อ

ตารางที่ 7 กลุ่มเสียงพยัญชนะอังกฤษและไทยที่คล้ายกัน 7 กลุ่ม

ภาษาอังกฤษ	*ภาษาไทย*
A E I O U H W Y	อ ห ฮ ว ญ ย
B F P V	บ ฝ ฟ ป ผ พ ภ ว
C G J K Q S X Z	ข ฃ ค ฅ ฆ ฉ ช ฌ ก จ ช ศ ษ ส
D T	ฎ ด ฏ ต ฐ ฑ ฒ ถ ท ธ
L	ล ฬ
M N	ม ณ น
R	ร

ตารางที่ 8 ตารางเข้ารหัสเสียงไทย-อังกฤษ

ภาษาอังกฤษ	*ภาษาไทย*	รหัส	Note
AEIOUHWY	อ	0	#1
BFPV	บ ฝ ฟ ป ผ พ ภ ว	1	
CGJKQSXZ	ข ฃ ค ฅ ฆ ฉ ช ฌ ก จ ซ ศ ษ ส	2	
DT	ฎ ด ฏ ต ฐ ท ฒ ถ ท ธ	3	
L	ล ฬ	4	
MN	ม ณ น	5	
R	ร	6	
AEIOU	อ	7	#2
H	ห ฮ	8	#2
W	ว	1	#2
Y	ย ญ	9	#2
	ง	52	

#1 : รหัสสำหรับตัวอักษรตัวซ้ายสุดของคำ

#2 : รหัสสำหรับตัวอักษรทุกตัวของคำยกเว้นตัวซ้ายสุด

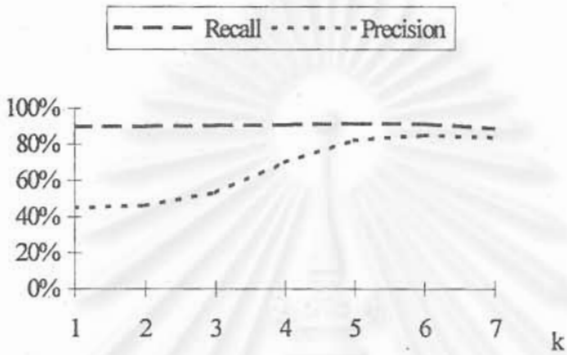
วิธีการเข้ารหัสที่ได้นำเสนอมานี้ใช้เวลาการเข้ารหัสเสียงต่อคำเป็น O(n) โดยที่ n คือจำนวนตัวอักษรของคำที่ถูกเข้ารหัส

4.3 ผลการทดลอง

เราได้เขียนโปรแกรมที่ทำงานตามขั้นตอนวิธีการเข้ารหัสเสียงที่ได้นำเสนอในหัวข้อที่แล้ว โดยเปลี่ยนแปลงตาราง และฟังก์ชันการเข้ารหัสใน [1] จากนั้นทำการทดสอบโปรแกรมดังกล่าวด้วยข้อมูลคำภาษาอังกฤษ และคำทับศัพท์ภาษาไทยของคำนั้นเป็นจำนวน 1,902 คู่ คำส่วนใหญ่เป็นชื่อเฉพาะ (อาทิเช่น ชื่อสินค้า ชื่อประเทศ ชื่อนักวิทยาศาสตร์ นักคณิตศาสตร์ เป็นต้น) และคำศัพท์เทคนิคทางวิทยาศาสตร์ คณิตศาสตร์ และเคมี ได้มาจาก [19], [20], [21], และ [22]

การทดลองกระทำโดยการเข้ารหัสเสียงทุกๆคู่คำ แล้วเก็บรหัสทั้งหมดในฐานข้อมูล จากนั้นค้นหาทุกๆคำที่เก็บ ทีละคำ โดยใช้รหัสเสียงของคำในการค้น เพื่อวัดค่าความแม่นยำ และค่าเรียกคืน เราทำการทดลองนี้ซ้ำกันหลายๆหน โดยแปรค่า k ซึ่งเป็นค่าความยาวน้อยสุดของรหัสเสียงที่ถูกพิจารณา (ที่นำเสนอในหัวข้อที่แล้ว) เพื่อศึกษาผลกระทบของค่า k กับผลของการค้นคืน รูปที่ 11 แสดงผลการทดลองที่ได้ อนึ่งในชุดของข้อมูลที่ถูกทดสอบนั้น มีเพียง 1% ของข้อมูลเท่านั้นที่รหัสเสียงมีความยาวเกิน 7 ดังนั้นจึงไม่ได้แสดงผลการทดลองสำหรับค่า k ที่เกิน 7

จากผลที่ได้ในรูปที่ 11 ค่าเรียกคืนของการค้นคืนมีค่าสูงราว 90% และค่อยๆลดลงเมื่อ k เพิ่มขึ้น ซึ่งมีพฤติกรรมตรงข้ามกับค่าความแม่นยำที่เริ่มที่ราว 45% และเพิ่มขึ้นไปที่ 80% เมื่อ k เพิ่มขึ้น ถึงแม้จะเป็นที่คาดได้ล่วงหน้าอยู่แล้วว่าค่าเรียกคืนจะลดเมื่อค่าความแม่นยำเพิ่มในระบบการค้นคืนทั่วไป แต่สำหรับในกรณีนี้ค่าเรียกคืนลดในอัตราที่ช้ากว่าอัตราการเพิ่มของค่าความแม่นยำ โดยเฉพาะอย่างยิ่งเมื่อ $k > 4$ การลดและเพิ่มของค่าทั้งสองจะเริ่มอยู่ตัว จึงสรุปได้ว่าขั้นตอนวิธีที่นำเสนอนั้นจะใช้ได้ผลดีในการค้นคืนเมื่อพิจารณาเฉพาะคำที่มีความรหัสเสียงเกิน 4 (หรือจากการสังเกตคำและรหัสเสียงจะได้ว่า เหมาะสำหรับคำที่มีความยาวเกิน 7 โดยประมาณ)



	1	2	3	4	5	6	7
Recall	0.90	0.90	0.90	0.91	0.91	0.91	0.89
Precision	0.45	0.46	0.53	0.7	0.82	0.85	0.84

รูปที่ 11 กราฟแสดงความสัมพันธ์ค่าเรียกคืนและค่าความแม่นยำ กับความยาวน้อยสุดของรหัสเสียง

4.4 สรุป

บทนี้ได้กล่าวถึงขั้นตอนวิธีการเข้ารหัสเสียงเพื่อการค้นคืนคำทับศัพท์ข้ามภาษา เมื่อมีการค้นคำเฉพาะภาษาอังกฤษที่อาจมีการทับศัพท์เป็นภาษาไทยในเอกสารด้วย จะค้นคำนั้นด้วยรหัสเสียงของคำแทน การเข้ารหัสเสียงนี้ปรับปรุงวิธีมาจากระบบซาวน์เดกซ์ โดยเพิ่มเติมการเข้ารหัสกลุ่มตัวอักษรทั้งอังกฤษและไทยที่มีเสียงคล้ายกันในตารางเข้ารหัสเดียวกัน และเปลี่ยนกฎเกณฑ์การเข้ารหัสจากวิธีเดิมเพียงเล็กน้อยใช้เวลาการเข้ารหัสแปรตามความยาวคำ โดยผลที่ได้จากการทดลองพบว่าได้ค่าเรียกคืนและค่าความแม่นยำมากกว่า 80% เมื่อจำกัดการพิจารณาเฉพาะคำที่รหัสเสียงมีความยาวเกิน 4

5. บทสรุปและข้อเสนอแนะ

งานวิจัยนี้นำเสนอขั้นตอนวิธีในการประมวลผลข้อความในระบบค้นคืนเอกสารไทย ที่ใช้โครงสร้างแฟ้มผกผัน การประมวลผลดังกล่าวประกอบด้วยการหาคำเพื่อทำดัชนี และการเข้ารหัสคำเพื่อรองรับการค้นคืนข้ามภาษา ขั้นตอนวิธีการหาคำเพื่อจัดทำดัชนีอาศัยพจนานุกรมช่วยในการแยกคำ และสามารถจัดการกับข้อความที่อาจมีคำที่ไม่ปรากฏในพจนานุกรม อาทิเช่นคำทับศัพท์ หรือคำที่สะกดผิดเป็นต้น โดยอาศัยกฎการแบ่งพยางค์ข้อความไทย ขั้นตอนวิธีนี้สร้างกราฟการต่อและซ้อนกันของคำที่หาได้จากพจนานุกรม ซึ่งมีโหนดแทนคำ และเส้นเชื่อมแทนการต่อหรือซ้อนกันของคำ เส้นทางสั้นสุดจากโหนดซ้ายสุดไปยังโหนดขวาสุดของกราฟนี้ แทนรายการคำสำคัญที่ควรถูกจัดทำดัชนีสำหรับแฟ้มผกผัน เวลาการทำงานของการหาคำนี้เป็น $O(n^2)$ โดยที่ n คือความยาวข้อความ ระบบค้นคืนข้อความสามารถรวมขั้นตอนวิธีนี้ในกระบวนการเตรียมเอกสารก่อนการทำดัชนี และในกระบวนการประมวลข้อคำถามก่อนการสืบค้น ผลการทดลองพบว่าจำนวนคำที่หาได้เพื่อทำดัชนีนั้นมีจำนวนประมาณ 30-50% ของจำนวนคำที่เป็นไปได้ทั้งหมดในข้อความทดสอบ

สำหรับขั้นตอนวิธีในการเข้ารหัสคำทับศัพท์ เพื่อรองรับการค้นคืนคำทับศัพท์ข้ามภาษาจากอังกฤษมาไทยนั้น ทำให้ระบบสามารถค้นคืนเอกสารที่มีคำสำคัญภาษาอังกฤษ หรือคำทับศัพท์เป็นภาษาไทยของคำอังกฤษนั้นได้ (ตัวอย่างเช่นเมื่อต้องการค้นหาเอกสารที่มีคำว่า "CLINTON" จะได้เอกสารที่มีคำว่า "คลินตัน" กลับคืนมาด้วย) รหัสที่หาได้นี้เป็นรหัสของการออกเสียงเพื่อให้คำภาษาอังกฤษ และคำทับศัพท์เป็นภาษาไทยนั้นมีรหัสเสียงเดียวกัน วิธีเข้ารหัสนี้ปรับปรุงมาจากวิธีเข้ารหัสเสียงและตารางเข้ารหัสในระบบซาวน์เดกซ์ วิธีนี้ใช้เวลาการเข้ารหัสแปรเชิงเส้นตามความยาว ผลที่ได้จากการทดลองพบว่าได้ค่าเรียกคืนและความแม่นยำมากกว่า 80% เมื่อจำกัดการพิจารณาเฉพาะคำที่รหัสเสียงมีความยาวเกิน 4 หรือสำหรับคำที่มีความยาวเกิน 7

การค้นคืนข้อความไทยนั้นยังมีประเด็นปัญหาที่ต้องพิจารณาอีกมากมาย หากจะขีดวงเฉพาะกับปัญหาการหาคำเพื่อทำดัชนี และปัญหาการค้นคืนข้ามภาษาที่งานวิจัยนี้ได้นำเสนอขั้นตอนวิธีในการแก้ปัญหาในระดับหนึ่งแล้ว จะพบว่ายังมีประเด็นที่รอทำการวิจัยในอนาคตดังนี้

- *ลักษณะของพจนานุกรม* ในงานวิจัยนี้พจนานุกรมคือฐานข้อมูลคำภาษาไทย (พจนานุกรมที่ใช้ในงานวิจัยนี้เก็บประมาณห้าหมื่นกว่าคำ) ซึ่งคำบางคำในพจนานุกรมสร้างปัญหาในการแยกคำ เนื่องจากเป็นคำที่เป็นส่วนนำหน้าของคำอื่นที่ไม่ปรากฏในพจนานุกรม อาทิเช่นคำว่า "การก" มีส่วนนำหน้า "การ" ที่เป็นส่วนนำหน้าของคำอื่นๆ เป็นต้น ปัญหาที่ควรทำวิจัยคือการวิเคราะห์หาความสัมพันธ์ของคำต่างๆในพจนานุกรม การจัดเก็บความถี่ของการใช้คำเพื่อ

การประกอบการตัดสินใจเลือกใช้คำในขั้นตอนการแยกคำ ผลกระทบของการจัดเก็บคำผสมกับประสิทธิผลในการแยกคำ เป็นต้น

- *การจัดการข้อความที่มีความผิดพลาดจากการพิมพ์* ระบบการค้นคืนโดยทั่วไปสามารถรองรับความผิดพลาดของข้อความได้ โดยใช้ลักษณะการเทียบแบบประมาณ (approximate matching) แต่เนื่องจากความผิดพลาดอันมีสาเหตุจากการพิมพ์นั้นเป็นปัญหาที่เกิดขึ้นบ่อยมากในการจัดเตรียมเอกสาร และมีรูปแบบความผิดพลาดที่เด่นชัด เช่นการพิมพ์ตก พิมพ์ซ้ำ พิมพ์เกิน พิมพ์สลับ และพิมพ์ผิดตำแหน่งแป้นที่ใกล้เคียง เป็นต้น หากนำต้นแบบความผิดพลาดเหล่านี้ที่เกิดขึ้นการพิมพ์ภาษาไทยจำลองในระบบการค้นคืน ย่อมได้ผลลัพท์ในการค้นคืนที่ดีกว่าการเทียบแบบประมาณโดยทั่วไป

- *การจัดการคำย่อในเอกสาร* คำย่อที่ปรากฏในเอกสารนั้นมักเป็นคำที่ไม่ปรากฏในพจนานุกรม เป็นผลให้การแยกคำผิดพลาดจากที่ควรเป็น ในภาษาไทยนั้นคำย่อบางครั้งเขียนมีจุดคั่นระหว่างตัวอักษรย่อ บางครั้งมีจุดปิดท้าย อีกทั้งบางครั้งตัวย่อถูกพิมพ์โดยมีช่องว่างคั่นอีก เป็นการสร้างความสับสนให้กับระบบค้นคืน

- การค้นคืนคำทับศัพท์ภาษาไทยไปเป็นภาษาอังกฤษ ตัวอย่างเช่นเมื่อต้องการค้นเอกสารที่มีคำว่า "สมชาย" จะได้เอกสารที่มีคำว่า "SOMCHAI" คืนมาด้วย ปัญหานี้จะซับซ้อนกว่าปัญหาการทับศัพท์ในทิศทางกลับกันที่ได้นำเสนอในงานวิจัยนี้ เนื่องจากระบบเสียงของภาษาไทยมีมากกว่าภาษาอังกฤษ อีกทั้งมีกระบวนวิธีการถ่ายเสียงที่ไม่เป็นมาตรฐานนัก

ภาคผนวก ก. กฎการแบ่งพยางค์ภาษาไทย

การแบ่งพยางค์ภาษาไทยในโปรแกรมซียูไรท์เตอร์รุ่น 1.52 [27] ใช้อัลกอริทึมการแบ่งพยางค์โดยใช้กฎ จากการวิเคราะห์สถิติการใช้ตัวอักษรภาษาไทย และหลักทางภาษาศาสตร์แล้วสามารถแบ่งตัวอักษรได้เป็น 5 กลุ่มใหญ่ ๆ คือ

1) พยัญชนะ ปัจจุบันมีใช้อยู่ 42 ตัว (ไม่นับ ข ค) สามารถแบ่งพยัญชนะออกเป็น 5 กลุ่มย่อยๆ ดังนี้

- พยัญชนะที่จะเป็นพยัญชนะต้นเสมอ
  ฉ ผ ฝ ฮ
- พยัญชนะที่ปกติจะเป็นพยัญชนะตัน ได้แก่
  ห ภ ฆ ฟ ซ ฑ
- พยัญชนะที่เป็นได้ทั้งพยัญชนะและสระ ได้แก่
  อ ว ร (ร ใช้ในรูป รร)
- พยัญชนะที่ปกติจะเป็นตัวสะกด ได้แก่
  ศ ณ ญ ษ ฐ ฎ ฏ ฒ ฬ ฌ
- พยัญชนะที่เป็นได้ทั้งตัวสะกดและพยัญชนะต้น ได้แก่
  ก ข ค ง จ ช ด ต ถ ท ธ
  น บ ป พ ม ย ล ส

2) สระ ที่ใช้อยู่ในปัจจุบันมี 17 ตัว ซึ่งสามารถแบ่งออกเป็นกลุ่มย่อย ได้ 5 กลุ่ม ดังนี้

- สระ ที่ปกติจะเป็นตัวอักษรแรกของคำ ได้แก่
  เ แ ไ ใ โ
- สระที่ปกติจะเป็นตัวอักษรตัวสุดท้ายของคำ ได้แก่
  ะ ำ
- สระที่ปกติจะต้องการสะกด ได้แก่
  ◌ั ◌็ ◌ิ

- สระที่มี หรือ ไม่มีตัวสะกดก็ได้ ได้แก่

  า ิ ี ุ ู

- สระพิเศษ ที่ใช้เฉพาะในคำบางคำ ได้แก่

  ฤ

3) วรรณยุกต์ มี 4 ตัว ได้แก่ ่ ้ ๊ ๋

4) สัญลักษณ์พิเศษ มี 21 ตัว ได้แก่

  ์ ฯ ๆ ( ) [ ] - { } “ !

  : ; | $ _ . ? / blank

5) ตัวเลข มี 10 ตัว ได้แก่ ตัวเลข 0-9

ตามหลักไวยากรณ์ของภาษาไทย สามารถแบ่งรูปแบบของคำได้ 7 รูปแบบดังนี้

[<ว>]

<พ>[<พ>][<ต>[<ต>]][<ก>] : จะ ถลา ถ้า กล้า บาน มฤต กานต์

[<ว>]

<ส>

<พ>[<พ>][<ต>[<ต>]][<ก>] : กิน หมั้น ที่ ฉันท์ คลี่ มือ

[<ว>]

<พ>[<พ>][<ต>[<ต>]][<ก>] : จุ อุ๊ คุ้ม บรณ์

<ส>

[<ว>]

<ส>[<พ>][<พ>][<ต>[<ต>]][<ก>] : ไพร แม่ โพยม โพธิ์

[<ว>]

<ส>

<ส><พ>[<พ>]<ต>[<ต>][<ก>] : เถิน เพลิน เปิ่น เป็น เซ็นต์

[<ว>]

<ส>

<ส><พ>[<พ>]<ว>[<ต>[<ต>][<ก>] : เกือบ เกลียด เพลี้ย

[<ว>]

<ส><พ>[<พ>]<ว>[<ว>]][<ก>] : เสาร์ เกล้า เช้า เกาะ เธอ

โดยที่ <พ> = พยัญชนะต้น <ส> = สระ

<ว> = วรรณยุกต์ <ต> = ตัวสะกด

<ก> = การันต์ [] = ให้เลือก อาจมีหรือไม่มีก็ได้

จากการวิเคราะห์รูปแบบของคำ จึงได้สร้างกฎเกณฑ์ของการแบ่งคำเป็นข้อๆ โดยยึดหลักการทางภาษาศาสตร์และข้อมูลทางสถิติ แต่เนื่องจากภาษาไทยประกอบด้วยคำที่มีรูปแบบแตกต่างกันมากมาย ดังนั้นกฎที่ใช้ทุกกฎจึงต้องมีคำยกเว้นของกฎนั้นๆ

กฎเกณฑ์การแบ่งคำไทยที่สร้างขึ้น ได้รวบรวมเป็นหมวดหมู่ไว้แล้ว โดยจะยึดหลักการให้เป็นกฎที่มีความแน่นอน เพื่อจะใช้กับคอมพิวเตอร์ได้ โดยเฉพาะจะมีการกำหนดข้อยกเว้นต่างๆไว้ด้วย เพื่อให้มีความสมบูรณ์ของกฎเกณฑ์ ซึ่งจะมีหลักการดังนี้คือ

กฎข้อที่ 1 เครื่องหมายพิเศษ (Special Character) สามารถใช้แบ่งคำได้ โดยจะแบ่งเครื่องหมายพิเศษนี้ ออกเป็น 2 กลุ่ม ดังนี้ คือ

- ประเภทที่เป็นวงเล็บเปิดและเครื่องหมายพิเศษบางตัวจะแบ่งคำหน้าตัวอักษรเหล่านั้นได้แก่ ([{ | /
- ประเภทที่เป็นวงเล็บปิดและเครื่องหมายพิเศษอื่นๆ จะแบ่งคำตรงตำแหน่งตัวอักษรนั้นๆ ได้แก่)] } ! $ % - : ; ?

กฎข้อที่ 2 ตัวการันต์ (์) มักจะใช้เป็นตัวสุดท้ายของคำ เช่น ศิลป์ สันต์ เป็นต้น แต่ก็มีคำยกเว้นอยู่หลายคำ โดยมากมักจะเป็นคำที่มาจากภาษาอังกฤษ เช่น บอร์ด ฟิล์ม เป็นต้น ซึ่งคำเหล่านี้มักจะนำหน้าการันต์ด้วยตัว ร และ ล เสมอ

กฎข้อที่ 3 ตัวอักษรสระอะ (ะ) และสระอำ (ำ) มักจะใช้เป็นตัวสุดท้ายของคำซึ่งก็มีข้อยกเว้น กรณีสระอะมีตัวอักษร ห์ บงท้าย เช่น เคราะห์ เป็นต้น หรือกรณีสระอำ มีวรรณยุกต์ตามมาวรรณยุกต์ก็จะเป็นตัวสุดท้ายแทน เช่น ช้ำ เป็นต้น

กฎข้อที่ 4 ตัวสระไม้ม้วน (ใ) จะใช้นำหน้าพยัญชนะเสมอ จึงจะเป็นตัวอักษรแรกสุดของคำ (คำในภาษาไทยที่ใช้ ใ มี 20 คำเท่านั้น)

กฎข้อที่ 5 ตัวสระไม้หันอากาศ (ั) สระอิ (ิ) สระอี (ี) และ สระอึ (ึ) โดยปกติมักจะต้องการตัวสะกด 1 ตัว เช่น กิน ตัด ยืด แต่มีคำยกเว้นหลายคำ เช่น นัยน์ เกิน เรือง เป็นต้น

กฎข้อที่ 6 ตัวสระเอ (เ) สระแอ (แ) สระโอ (โ) และสระไม้มลาย (ไ) ปกติจะใช้นำหน้าพยัญชนะ แต่ก็จะมีคำยกเว้นอยู่หลายคำ เช่น มเหสี สแลง อโหสิ สไบ เป็นต้น

กฎข้อที่ 7 ตัวพยัญชนะ ฉ ผ ฝ ฮ จะใช้เป็นพยัญชนะต้นนำหน้าเสมอ เช่น ฉกรรจ์ ผไท ฝรั่ง ฮ้อ เป็นต้น แต่อาจจะมีตัวสระในกฎข้อที่ 4, 6 นำหน้าได้ เช่น เฉลียง เผย โฮ้ง เป็นต้น

กฎข้อที่ 8 ตัวสระอุ (ุ) และสระอู (ู) มักจะใช้ไว้ในใต้พยัญชนะตัวแรก หรือพยัญชนะตัวที่สองของคำ เช่น คุณ กุล ปลุก สนุก ขูด มูก กรูด อณู เป็นต้น แต่อาจจะมีคำที่สระไปอยู่ใต้พยัญชนะตัวอื่นๆ (ไม่ใช่พยัญชนะตัวแรกหรือตัวที่สอง) เช่น เหตุ ธาตุ เรณู เมนู ไอศูรย์ เป็นต้น

กฎข้อที่ 9 ตัวสระอา (า) โดยปกติจะต้องมีพยัญชนะนำหน้าอย่างน้อย 1 ตัว เสมอ โดยจะพิจารณาเป็น 2 กรณี คือกรณีที่ในคำไม่มีการใช้วรรณยุกต์เลย เช่น ปากกา นาน สบาย สตางค์ และกรณีที่ในคำมีการใช้วรรณยุกต์ร่วมอยู่ด้วย เช่น กร๊าฟ ฝ่าย หม้าย เป็นต้น โดยในแต่ละกรณีจะแบ่งคำหน้าหรือหลังสระก็ได้ แล้วแต่ความเหมาะสมของรูปแบบของคำนั้นๆ

กฎข้อที่ 10 ตัวอักษร อ ที่ใช้ร่วมกับวรรณยุกต์ จะพิจารณาจากตัวอักษรที่อยู่หน้าและหลังตัวอักษร อ โดยจะดูว่าพยัญชนะต้นแต่ละตัวนั้น เมื่อใช้กับ อ แล้ว จะมีตัวสะกดเป็นตัวใดได้บ้าง เช่น พยัญชนะต้นเป็นตัว ก สำหรับวรรณยุกต์ (่) แล้ว จะมีตัวสะกดเป็นตัวใดได้บ้าง เช่น พยัญชนะต้นเป็นตัว ก สำหรับวรรณยุกต์แล้ว จะมีตัวสะกดเพียงตัวเดียวคือ น (ก่อน) ส่วนวรรณยุกต์ (้) จะมีตัวสะกด คือ น ง ย (ก้อน ก้อง ก้อย) เป็นต้น

กฎข้อที่ 11 ตัวอักษรไม้ไต่คู้ (็) จะเป็นตัวที่เปลี่ยนรูปมาจากสระ เ-าะ เ-ะ แ-ะ ที่มีตัวสะกด ดังนั้นรูปแบบที่ใช้จะเป็น -อ- เ-- แ-- ทุกรูปแบบจะมีตัวสะกด 1 ตัว ยกเว้นถ้ามีตัวการันต์

กฎข้อที่ 12 สระผสม เ-ย และ เ-อ ที่จะใช้ร่วมกับวรรณยุกต์ต่างๆ โดยจะพิจารณาว่าแต่ละรูปแบบจะมีตัวสะกดหรือไม่และถ้ามีตัวสะกด ก็จะใช้ตัวใดได้บ้าง เพื่อจะพิจารณาตำแหน่งแบ่งคำก่อนหน้าหรือหลังสระนั้น

กฎข้อที่ 13 ตัวอักษร ฤ โดยปกติจะใช้อยู่ถัดไป จากพยัญชนะต้นนำหน้า เช่น กฤษณะ หฤทัย คฤหัสถ์ พฤกษ์ เป็นต้น แต่ก็จะมีข้อยกเว้นที่ใช้ตัวอักษร ฤ เป็นพยัญชนะต้นนำหน้าได้ เช่น ฤดู ฤดี ฤชา ฤกษ์ ฤทธิ์ ฤทัย ฤาษี เป็น

กฎข้อที่ 14 ตัวอักษร ห มักจะใช้เป็นพยัญชนะนำหน้าเสมอ แต่ก็มีคำยกเว้น เช่น สห มหา คหบดี มหกรรม มหรสพ มหัศ มหิ พรหม เคราะห์ เป็นต้น นอกจากนั้นจะเป็นคำที่มาจากต่างประเทศ เช่น จอห์ โอห์ม เป็นต้น หรือมีคำนำหน้าเป็นสระ เช่น เหา แห่ง เป็นต้น

กฎข้อที่ 15 ตัวอักษร ว จะต้องมีตัวสะกดอย่างน้อย 1 ตัว เมื่อใช้อยู่ต่อจากวรรณยุกต์ เช่น ม้วน ล้วน ม่วง เป็นต้น และอาจจะใช้ในรูปของสระอัว (-ว) เช่น ตัว มัว ขั้ว เป็นต้น

กฎข้อที่ 16 ตัวอักษร ร โดยปกติจะใช้ในรูปแบบของสระ -รร เช่น วรรณ จรรยา บรรจง เป็นต้น

กฎข้อที่ 17 เป็นกฎของสระลดรูป ซึ่งจะมีตัวสะกดเป็นตัวพยัญชนะต่อไปนี้ คือ ก ง ค น ม บ เช่น คน ชก กด เป็นต้น

กฎข้อที่ 18 ตัวอักษร ศ ฉ ญ ษ ฐ ฎ ฏ ฒ พ ฌ มักจะใช้เป็นตัวสะกดเสมอ เช่น กีฬา คณิกา อุษา อัศจรรย์ หญิง ชฎา ปฏัก แต่ก็มีข้อยกเว้นที่จะใช้เป็นพยัญชนะต้นนำหน้าคำได้ เช่น ศุลี ฐาน ฎีกา ญวร เป็นต้น

ภาคผนวก ข. ตัวอย่างระบบค้นคืนข้อความไทย

ภาคผนวกนี้จะนำเสนอตัวอย่างการรวมขั้นตอนวิธีการหาคำเพื่อทำดัชนี ที่ได้นำเสนอในงานวิจัยนี้เข้ากับระบบการค้นคืนข้อความที่มีอยู่ โดยจะใช้ระบบค้นคืนข้อความ EWS ของบริษัท Excite [2] เป็นกรณีศึกษา พร้อมทั้งทดสอบการทำงานกับฐานเอกสารจำนวนขนาดปานกลางราว 5,000 เอกสาร

ระบบค้นคืนข้อความ EWS ประกอบด้วยระบบย่อยในการจัดทำดัชนีแฟ้มผกผัน และระบบย่อยในการค้นเอกสารจากข้อคำถาม ระบบ EWS นี้ทำงานร่วมกับระบบบริการเวบ (Web server) โดยถูกเรียกใช้ผ่านหน้าเอกสารบนเวิลด์ไวด์เวบที่จัดทำขึ้น นั่นหมายความว่าการค้นคืนเอกสารนั้นกระทำได้ผ่านเครือข่ายอินเตอร์เน็ต ซึ่งเป็นแนวคิดของการพัฒนาระบบค้นคืนข้อความที่ได้รับความนิยมเป็นอย่างสูง ผู้วิจัยได้รวมขั้นตอนวิธีการหาคำเพื่อจัดทำดัชนีในงานวิจัยนี้ เข้ากับทั้งระบบย่อยจัดทำดัชนี และระบบย่อยค้นคืนเอกสารดังนี้ (ดูรูปที่ 12 ประกอบ)

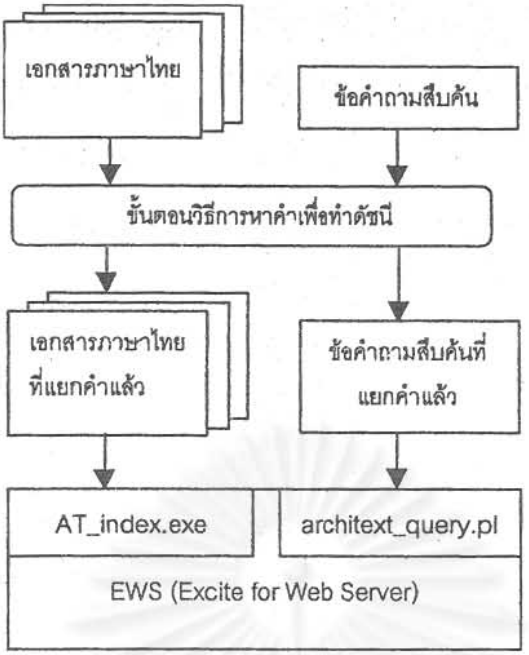
1. *ระบบย่อยจัดทำดัชนี* ในส่วนนี้ได้พัฒนาโปรแกรมจัดเตรียมข้อมูลที่แปลงแฟ้มเอกสารที่มีอยู่ ไปเป็นแฟ้มเอกสารใหม่ที่ประกอบด้วยคำที่ต้องจัดทำดัชนี [3] ซึ่งใช้ขั้นตอนวิธีการหาคำเพื่อจัดทำดัชนีที่รองรับภาษาไทยที่ได้นำเสนอในงานวิจัยนี้ โดยมีการแทรกรหัสแบ่งคำ เพื่อให้ระบบย่อยจัดทำดัชนีของ EWS รับรู้ขอบเขตของคำ เพื่อจัดทำดัชนีด้วยโปรแกรม AT_index.exe ของ EWS ตามปกติต่อไป

2. *ระบบย่อยค้นคืนเอกสาร* ในส่วนนี้ได้พัฒนาส่วนประมวลข้อคำถามการสืบค้นจากผู้ใช้ เพื่อแปลงข้อคำถามที่ผู้ใช้บรรยาย ไปสู่รายการของคำที่จะต้องนำไปค้น ตัวอย่างเช่นผู้ใช้สามารถตั้งข้อคำถามการสืบค้นว่าต้องการค้นคืนเอกสารที่มีคำสำคัญ "*ดื่มสุรามึนเมา*" ส่วนประมวลข้อคำถามนี้จะจัดการแปลงข้อคำถามนี้เป็น "*ดื่ม สุรา มึนเมา*" นั่นคือให้ระบบย่อยค้นคืนเอกสาร (โปรแกรม architext_query.pl) ของ EWS โดยใช้คำสำคัญสามคำที่หาได้ การประมวลข้อคำถามนี้ก็ใช้ขั้นตอนวิธีเดียวกับวิธีการหาคำเพื่อทำดัชนีที่ได้นำเสนอในงานวิจัยนี้

---

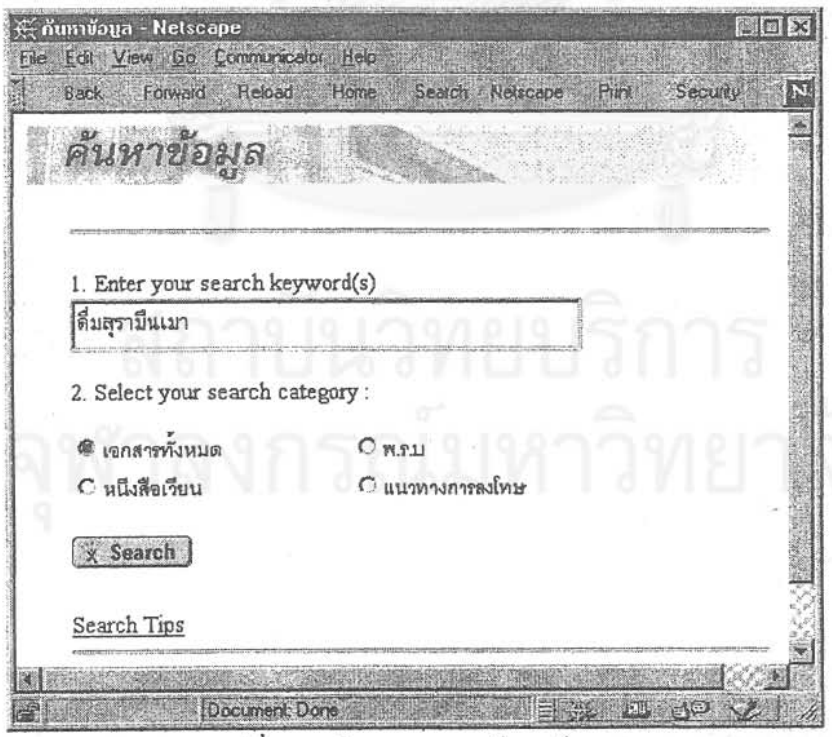
2 http://www.excite.com

3 โดยแฟ้มใหม่นี้จะมีแต่เฉพาะข้อความเท่านั้น ข้อมูลลักษณะอื่นๆ เช่น รูปภาพ เสียง เป็นต้น ในเอกสารต้นฉบับ จะไม่ถูกนำมาเก็บในแฟ้มใหม่ที่สร้างขึ้น



รูปที่ 12 การรวมขั้นตอนวิธีการหาคำเพื่อทำดัชนีเข้ากับระบบค้นคืน EWS

รูปที่ 13 เป็นตัวอย่างหน้าจอการค้นหาเอกสาร ผ่านหน้าเอกสารสืบค้น โดยใช้คำสำคัญคือ *ดื่มสุรามึนเมา* ซึ่งได้ผลลัพธ์ปรากฏเป็นหน้าเอกสารในรูปที่ 14 การค้นหาในลักษณะนี้โดยทั่วไปจะได้เอกสารกลับคืนมามากกว่าหนึ่งเอกสาร เพื่อให้ผู้ใช้เลือกกด (ผ่านตัวเชื่อมเอกสารของระบบเวิลด์ไวด์เวบ) อีกทีหนึ่ง โดยจะแสดงเพียงหัวเรื่องของเอกสารให้ผู้ใช้อ่านเท่านั้น



รูปที่ 13 หน้าจอระบบการค้นหาข้อมูล

รูปที่ 14 หน้าจอแสดงผลลัพธ์การค้นหาเอกสาร

ในกรณีที่ระบบสามารถหาเอกสารที่ต้องการตามคำสำคัญ ได้มากกว่าหนึ่งเอกสาร ระบบจะเรียงลำดับเอกสารที่ค้นหาได้ตามความมั่นใจว่าเอกสารต่างๆ นั้นเป็นเอกสารที่ผู้ใช้ต้องการ โดยมีการให้น้ำหนักความเชื่อมั่น ซึ่งแสดงเป็นเปอร์เซนต์ความมั่นใจ กำกับไว้ข้างหน้าชื่อของเอกสาร ตัวอย่างเช่นในรูปที่ 14 ระบบมีความมั่นใจ 83% ว่าเอกสาร "แนวทางการลงโทษ : ตัวอย่างที่ 789" ตรงตามที่ผู้ใช้ต้องการค้น และยังค้นพบเอกสารอื่นๆ อีกด้วยตามความมั่นใจที่ลดหลั่นลงไปตามลำดับ [3]

---

3 ในที่นี้เอกสารแนวทางการลงโทษ : ตัวอย่างที่ 789 มีใจความว่า "*เกษตรอำเภอควนกาหลง (เจ้าหน้าที่บริหารงานการเกษตร ๖) จังหวัดสตูล ได้ดื่มสุรามีนเมาเป็นประจำ ส่งเสียงดังพูดจาก้าวร้าว และด่าว่านายอำเภอซึ่งเป็นผู้บังคับบัญชา โทษ ลดขั้นเงินเดือน ๑ ขั้น*"

ภาคผนวก ค. รายชื่อบทความจากงานวิจัยนี้

1. P. Suwanvisat and S. Prasitjutrakul, "Thai-English Cross-Language Transliterated Word Retrieval using Soundex Technique" *the National Computer Science and Engineering Conference 1998*, Kasetsart University, Bangkok, Thailand

2. W. Kanlayanawat and S. Prasitjutrakul, "Automatic Indexing for Thai Text with Unknown Words using Trie Structure",*Proceeding of the Natural Language Processing Pacific Rim Symposium 1997 (NLPRS'97)*, pp. 115-120, Phuket, Thailand, December 2-4, 1997.

บรรณานุกรม

[1] A. Binstock and J. Rex, *Practical Algorithms for Programmers*, Addison Wesley, 1995.

[2] S. Charnyapornpong, "A Thai Syllable Separation Algorithm," M.Eng. Thesis, Asian Institute of Technology, Aug. 1983.

[3] N. Collier, A. Kumano, and H. Hirakawa, "Acquisition of English-Japanese proper nouns from noisy-parallel newswire article using Katakana matching", *Proc. of the Natural Language Processing Pacific Rim Symposium 1997*, Phuket, Thailand, Dec. 2-4, pp. 309-320.

[4] C. Faloutsos and D.W. Oard, "A Survey of Information Retrieval and Filtering Methods," Technical Report CS-TR-3514, University of Maryland, College Park, August 1995.

[5] W. B. Frakes and R. Baeza-Yates eds., *Information Retrieval : Data Structures and Algorithms*, Englewood Cliffs, N.J. : Prentice-Hall.

[6] G. Gonnet, "Unstructured Data Bases or Very Efficient Text Searching," *ACM PODS*, vol. 2, pp. 117-124, 1983.

[7] G. Gonnet, R. Baeza-Yates, and T. Snider "New Indices for Text: PAT Trees and PAT Arrays," in *Information Retrieval : Data Structures and Alforithms*, ed., W. B. Frakes and R. Baeza-Yates, Englewood Cliffs, N.J. : Prentice-Hall

[8] P. Jindavimonlert, "A Thai Text Retrieval System using the PAT tree," M.Sc. Thesis, Department of Computer Engineering Chulalongkorn University, 1996.

[9] A. Kawtrakul, C. Thumkanon, and S. Seriburi, "A Statistical Approach to Thai Word Filtering," Proc. of the second Symposium on Natural Language Processing, pp. 398-406, 1995

[10] K. Knight and J. Graehl, "Machine Transliteration", *Annual Meeting of the Association for Computational Linguistics (ACL-97/EACL-97)*.

[11] A. Kumano, "Building a technical term dictionary with Katakana-English Matching", *Gengoshorigakai - Annual Conf. of the Japanese Association for Natural Language Processing*, (in Japanese) Japan, March, pp.221-223.

[12] U. Manber and G. Myers, "Suffix Arrays: A New Method for On-line String Searches," *First ACM-SIAM Symp. on Discrete Algorithms*, pp. 319-327, San Francisco, 1990.

[13] Y. Matsuo and S. Shirai, "Using pronunciation to automatically extract bilingual word pairs", *Shizengengoshori*, (in Japanese), November, pp.101-106.

[14] T.H. Merrett and H. Shang, "Trie Methods for Representing Text," *Proc Fourth Int'l Conf., FODO'93*, LNCS 730, pp. 130-145, Chicago: Springer-Verlag, Oct. 1993.

[15] D. Oard and B. Dorr, "A Survey of Multilingual Text Retrieval", *Technical Report UMIACS-TR-96-19 CD-TR-3615*, University of Maryland, College Park, April 1996.

[16] S. Ongroongruang, R. Prongsirivattana, and V. Jantarasukree, "English to Thai Word Retrieval Using Sound Index", *Proc. 2nd SNLP'95*, Bangkok Thailand, Aug. 2-4, 1995, pp. 407-413.

[17] P. Prapapitayakorn and et. al., *รู้จักภาษาไทย* Odien Bookstore, 1976

[18] Y. Poovorawan and V. Imarom, "Dictionary-based Thai Syllable Segmentation (in Thai)," 9th Electrical Engineering Conference, 1986.

[19] Royal Academy, *หลักเกณฑ์การทับศัพท์* (Transliteration Guideline), 1992.

[20] Royal Academy, *ศัพท์วิทยาศาสตร์* (Science Dictionary), 1993.

[21] Royal Academy, *ศัพท์คณิตศาสตร์* (Mathematics Dictionary), 1997.

[22] Royal Academy, *หนังสือเรียนวิชาเคมี เล่ม 1 หลักสูตรมัธยมปลาย 2524* (Chemistry Book 1: High School Level 1981), 1987.

[23] G. Salton and M.J. McGill, *Introduction to Modern Information Retrieval*. McGraw-Hill, 1983

[24] D. Sawamibhadhi, "Implementation of Thai Grammar Analysis Software under UNIX system (in Thai)", Thammasart Univ., 1990.

[25] H. Shang, "Trie Methods for Text and Spatial Data on Secondary Storage," Ph.D. Dissertation, School of Computer Science, McGill University, Nov. 1994.

[26] H. Shang and T.H. Merrett, "Tries for Approximate String Matching," *IEEE Trans. on Knowledge and Data Eng.*, Vol. 8, No. 4, pp. 540-547, Aug. 1996.

[27] D. Sintupunpratum and C. Bandhitanont, "Thai Word Processing (in Thai)", Proc. of the second Symposium on Natural Language Processing in Thailand, pp. 322-376, March 1993.

[28] V. Sornlertlamvanich, "Thai Word Segmentation in Language Translation System," *Computerized Language Translation* (in Thai), p. 50-55, 1993.

[29] M. Weiss, *Data Structures and Algorithms in C, 2nd Ed.* Addison Wesley, 1997.

[30] I.H. Witten, A. Moffat, and T.C. Bell, *Managing Gigabytes : Compressing and Indexing Documents and Images*, N.Y., Van Nostrand Reinhold.

Automatic Indexing for Thai Text with Unknown Words using Trie Structure *

Witoon Kanlayanawat and Somchai Prasitjutrakul
Department of Computer Engineering
Chulalongkorn University
Bangkok 10330, Thailand
Phone : (66-2)-218-3743, Fax : (66-2)-215-3554, E-Mail : somchaip@computer.org

Abstract

In this paper, we present an automatic indexing for Thai text retrieval system where given documents can have words that are unknown to the system's dictionary. Trie structure is used as a main file structure for indexing which supports word, pattern, and approximate searching. Since there is no explicit inter-word delimiter in Thai text, we also propose a word segmentation algorithm that segments a given text to a set of words and *sistrings* (semi-infinite strings) of unknown words for adding to the trie. The algorithm first finds a set of words maximally matching all the sistrings of a given text. Then it constructs an overlapping graph whose shortest paths represent a smallest list of words minimizing unknown strings of the text. By using our proposed dictionary-based word segmentation algorithm which can deal with unknown words, along with the use of trie structure to store the index, precision and recall of the retrieval can be enhanced.

1. Introduction

Text retrieval has become one of the most essential tools for managing information as computer-generated documents get published and computers get connected locally and globally, especially with the advent of the World Wide Web and CD-ROM. Traditionally, there are four major techniques, full text scanning, inversion, signature file, and clustering for text retrieval. A detail survey of these technique can be found in [2], [3], [10], [14]. Full text scanning locates the documents by searching through all documents for the specified string pattern. The technique does not need additional space for index but the search time is linearly proportional to the size of text. In inversion technique, keywords for all the documents are stored in alphabetical order in an index file (which can be implemented using sorted array, B-tree, or trie structures). For each keyword, there is a list of pointers (kept in a posting file) to associated documents. The technique is easy to implement and gives fast response time. However, it does suffers from the high storage overhead for the index. In signature file technique, each document has an associated *signature* which is a bit string created by using hashing on its words and superimposed coding. Then signatures of all the documents are stored in a separate file called signature file which is much smaller than the original document files and thus a search for signature pattern is much faster than a search for string pattern. Since it uses a signature as a representation of a document, it introduces a notion of *false hit* where a matched signature does not always mean that the corresponding string pattern matches. In clustering technique, similar documents are grouped together to form clusters to improve the efficiency and effectiveness of retrieval.

* This research was supported by the Thai Government Research Fund.

When building the database for text retrieval, text in each document is segmented into words which are optionally compared against a stoplist (a list of words having no index value). Non-stoplist words are then stemmed so that variations of the same words are represented with only one pattern. Then, each of the stemmed word is assigned a weight used during the search to rank retrieved documents. Finally, stemmed words along with their weights and locations are kept into the database.

Segmenting a given text into words is a nontrivial task in Thai (and other Asian languages) text processing since there is no explicit inter-word delimiter. There are currently two approaches in Thai word segmentation, rule-based and dictionary-based approaches. The rule-based approach uses a set of extensively studied rules for Thai syllable [1],[13],[15]. Although the technique achieves high precision in syllable segmentation, it is not suitable for indexing application since it is word not syllable that is required to create the index. The dictionary-based approach matches words in dictionary or lexicon with a given text. The matching can use longest-word-matching greedy strategy [16], least-number-of-matched-words strategy [17], or statistical data of word tags [7]. The word segmentation gets more complicated if the text to be segmented contains some unknown words (we define the unknown words to be words not kept in the dictionary being used) which can be proper names, transliterated words, words with spelling error, etc.

In this paper, we present an automatic indexing for Thai text retrieval system. By using our proposed dictionary-based word segmentation algorithm which can deal with unknown words, along with the use of trie structure to store the index, precision and recall of retrieval can be enhanced. The word segmentation algorithm constructs an overlapping graph whose shortest path represents a smallest list of words minimizing unknown strings of the given text. The words and unknown strings obtained then determines corresponding locations of sistrings (semi-infinite strings) to be kept in a trie structure. Trie structure is reviewed in Section 2. Detail of our word segmentation algorithm is described in Section 3. Section 4 summarizes the entire automatic indexing and retrieval. Conclusion of the paper is given in Section 5.

2. Trie Structures

Tries have been used for indexing large texts [4],[8],[9]. Tries are trees whose edges represent letters of the alphabet encoding the data. Therefore a word is represented as a path from root to leaf. Tries support efficient prefix searching (prefix searching is a searching for any word matching a given prefix) since all words sharing a prefix share the same path from root to an internal nodes. In addition, approximate matching can also be efficiently performed in trie structure which is applicable for searching text with error [12]. Implementations of trie structures on secondary storage can be found in [9],[11]. In this work, we use trie structures to store both the dictionary and index of all of the documents.

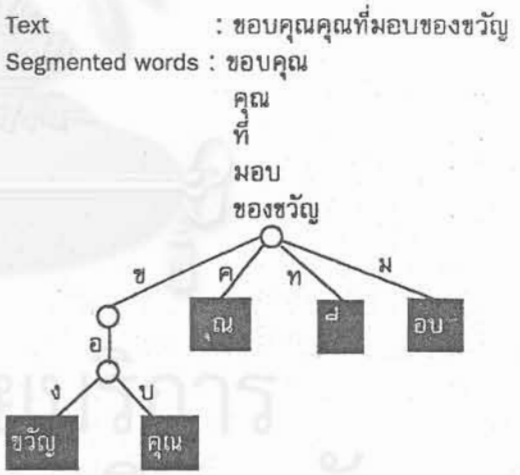


Figure 1. An example of a trie

Figure 1 shows an example of a trie consisting of segmented word from a text. Locations of the words are stored in its corresponding external node of the trie. When searching for a word, we follow branches determined from characters of the word. For example, to search for ขอบคุณ, we first go to the left branch (ข), go down (อ), and then go to the

right branch (บ) where we hit an external node whose content matches characters of the rest of the word (คุณ). Therefore the search time is proportional to the word length.

Gonnet [5] defined a semi-infinite string, or *sistring*, to be a suffix of the text starting at some position. Figure 2 shows the first four sistrings of a given text. A text of length *n* can have at most *n* sistrings. All of the sistrings can be stored in a trie where the search procedure remains the same as described before. However, it is obvious that we can eliminate many useless sistrings whose starting locations are not the beginning of the word. In [6], sistrings of Thai text whose starting location can potentially be words (using word formation rule-based approach) are stored in a PAT tree [5] (which is a special binary trie). Although, the concepts of storing sistring gives high recall (i.e., it is unlikely to miss any occurrences of the given search word), but the precision of retrieval will be low if we keep too much useless sistrings (i.e., many retrieved documents are not what we actually want) since the search acts like a pattern matching without knowing the notion of word. For example, the phrase in Figure 2 will match the search word อบ in spite that it is not a word in the phrase.

Text : ขอบคุณคุณที่มอบของขวัญ
Sistrings : ขอบคุณคุณที่มอบของขวัญ
: อบคุณคุณที่มอบของขวัญ
: บคุณคุณที่มอบของขวัญ
: คุณคุณที่มอบของขวัญ

Figure 2 Examples of sistrings

However, sistring can be used where we can not determine word boundaries as will be presented in the next section.

3. Word Segmentation Algorithm

Given a text, our word segmentation algorithm uses dictionary-based approach to find a smallest list consisting of known words and minimal unknown words. The known words are directly stored in the trie whereas the unknown words are stored in the trie as a set of sistrings whose starting locations are determined to potentially be syllable boundary using rule-based approach.

Let T be a given text to be segmented, T_i be a sistring of T starting at the i-th character, $T_{i,j}$ be a substring of T consisting of the i-th thru j-th characters, and D be a dictionary.

The algorithm consists of four steps as follows :

1. For each T_i , i = 1,...,*n*, find a word, w_i , in D satisfying the following conditions :

   - w_i maximally matches T_i . Let $i' = i-1+$ length of w_i. Therefore $w_i = T_{i,i'}$.
   - w_i is not a substring of any w_j where $j < i$

   For example, T = นายเจมส์มาร์ตินต้องการผลิตรายการโทรทัศน์ we have w_i's as shown in Figure 3. (Other sistrings of T not shown in the figure have their w_i's equal to null strings.)

i	T_i	w_i
1	นายเจมส์มาร์ตินต้องการผลิตรายการโทรทัศน์	นาย
4	เจมส์มาร์ตินต้องการผลิตรายการโทรทัศน์	เจ
5	จมส์มาร์ตินต้องการผลิตรายการโทรทัศน์	จม
9	มาร์ตินต้องการผลิตรายการโทรทัศน์	มาร
13	ตินต้องการผลิตรายการโทรทัศน์	ติ
16	ต้องการผลิตรายการโทรทัศน์	ต้องการ
20	การผลิตรายการโทรทัศน์	การผลิต
24	ลิตรายการโทรทัศน์	ลิตร
26	ตรายการโทรทัศน์	ตรา
27	รายการโทรทัศน์	รายการ
33	โทรทัศน์	โทรทัศน์

Figure 3 Example shows w_i's for sistrings

2. Construct an overlapping graph (which is a weighted directed graph) G =(V,E) where

- $V = \{w_i \mid w_i$ obtained from step 1, $w_i \neq \emptyset, 1 \leq i \leq n\}$

- $E = \{ (w_i, w_j) \mid w_i$ is adjacent to or overlap with w_j , $i < j \}$ weight of an edge (w_i, w_j) is determined from cases shown in Table 1 :

The first case is the most favorable one where two words are adjacent. In case 2, we have two overlapping words which can be further segmented yielding segmentations with no unknown strings. For example, ต้องการผลิต (w_1 = ต้องการ and w_4 = การผลิต) can be segmented to ต้อง + การผลิต or ต้องการ + ผลิต. Case 3 represents cases where two overlapping words can be segmented yielding some segmentations having unknown strings . For example, เพื่อนำ (w_1 = เพื่อน and w_5 = อนำ) can be segmented to เพื่อ + นำ, เพื่อน + ำ, or เพื่ + อนำ. The last case (with the highest weight) represents cases where two overlapping words can be segmented yielding segmentations with unknown strings. Notice that cases 3 and 4 are used to handle text with errors. For example, เพื่อนำ may actually be เพื่อนทำ where the ท is missing in the text. In this situation, case 3 keeps the เพื่อน in addition to เพื่อ and นำ.

Table 1. Edge weighting for overlapping and adjacent words

#	Weights of (w_i, w_j)	Conditions
1	1	if $i'+1 = j$, i.e., w_i is adjacent to w_j
2	10	if $j \leq i'$ and $T_{i, j-1}$ and $T_{i'+1, j}$ are all in the dictionary.
3	100	if $j \leq i'$ and there exists a position k, $j-1 \leq k \leq i'$, such that both $T_{i, k}$, and $T_{k+1, j'}$ are in the dictionary.
4	1000	otherwise

From the example in step 1, we can construct the corresponding overlapping graph as shown in Figure 4.

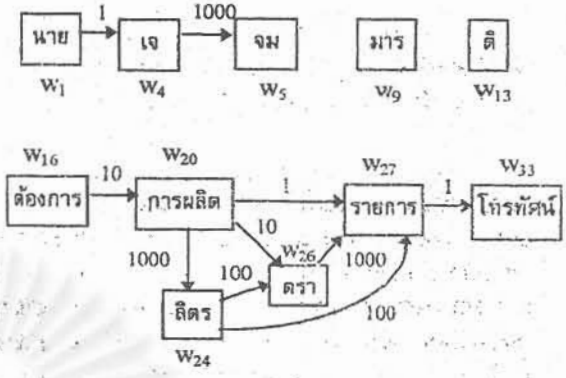


Figure 4 The overlapping graph for a text นายเจมส์มาร์ตินต้องการผลิตรายการโทรทัศน์

3. For each component of the graph, find a shortest path from the leftmost to the rightmost nodes of the component. Let $W = \{w_i \mid w_i$ has its corresponding node on the shortest paths obtained $\}$. From the above example, we get $W = \{w_1, w_4, w_5, w_9, w_{13}, w_{16}, w_{20}, w_{27}, w_{33}\}$

4. In this step, we determine W′, a set of segmented words, and U′, a set of sistrings for unknown words as follows.

   4.1 Let U be a set of unknown strings. U can be determined as follows :

      4.1.1 For each edge (w_i, w_j) with weight of 1000 on the shortest path (these are the two unknown strings discussed in case 4 of Table 1), we add $T_{i,j-1}$ and $T_{i'+1,j'}$ to U. From the above example, they are เ and ม (determined from edge (w_4, w_5)).

      4.1.2 For each pair of nodes w_i and w_j belonging to different components of G, $i < j$, and there is no $w_k \in W$, where $i < k < j$, we add $T_{i'+1, j-1}$ to U. From the above example, they are ส์, ์, and น.

4.1.3 We concatenate each string $T_{j,k}$ obtained from the two steps above to a word $w_i \in W$ which is adjacent to the left of $T_{j,k}$ in T. From the above example, U = { นายเ, เจม, จมส์, มาร์, ติน }.

4.1.4 This step combines any strings in U that overlap or adjacent to each others. From the above example, U= {นายเจมส์มาร์ติน }.

Then, U′ consists of sistrings T_i , T_{i+1},..., T_k of string $T_{i,k} \in U$. This step deals with unknown words which can be proper names, transliterated words, words with spelling error, etc. These words usually can be segmented to known words with some unknown strings in between. Since we do not know the exact word boundary of these unknowns, they are kept as sistrings in the trie. (Actually, we also apply syllable segmentation rules to eliminate some useless sistrings from the set U′.)

4.2 Let W* = W – {w_i | w_i being used to form unknown string in step 4.1.3}. Starting with W′ = W* , for any w_i and w_j of W*

- satisfying case 2 in Table 1, we add $T_{i,j-1}$ and $T_{i'+1,j'}$ to W′. From the above example, they are ต้อง and ผลิต determined from w_{16} and w_{20}.

- satisfying case 3 in Table 1, we add $T_{i,k}$, and $T_{k+1,j'}$ to W′ where k is defined in Table 1.

From the word segmentation algorithm just described, we obtained W′, a set of known segmented words, and U′, a set of sistrings of unknown words, to be kept in the trie. From the above example given a text T = นายเจมส์มาร์ตินต้องการผลิตรายการโทรทัศน์, we obtain W′ = {ต้องการ, การผลิต, รายการ, โทรทัศน์, ต้อง, ผลิต} and U′ = { T_i | T_i is sistring of string นายเจมส์มาร์ติน in T } to be stored in the index trie.

Performances of the four steps of the algorithm are as follows. Step 1 maximally matches each sistring of T to words in dictionary. There are n sistrings of T (n is the number of characters in T). Since our dictionary is also kept in a trie structure. Then it takes O(nk) where k is the maximum length of word in the dictionary. Step 2 constructs graph G=(V,E) which takes O(|V|+|E|). The worst case (happens when there are n w_i's where all of them overlap each other which is very unlikely) is O(n^2). Step 3 finds a shortest path for each component which is O(n^2). And the last step, re-scanning the string for unknown strings, takes O(n). Therefore the word segmentation algorithm takes worst case time of O($nk + n^2$) or O(n^2) where $k < n$ for a long sentence.

4. Thai Text Retrieval

From the trie structure along with the word segmentation algorithm presented, documents can be indexed by first parsing the document for explicit sentence or phrase delimiters e.g., space, then these sentences or phrases are segmented to obtain a set of words and a set of sistrings of unknown words. The two sets can be fed to other text processing modules such as word filtering, stoplist, stemming, and weight assignment before adding to the index trie.

For the retrieval phrase, user's query words must be segmented using the same algorithm so that an obtained set of words and sistrings are used for querying the trie structure.

5. Conclusion

An automatic indexing for Thai text retrieval system was presented in this paper where given documents can have words that are unknown to the system's dictionary. Unknown words can be misspelled words, proper words, transliterated words, etc. The fundamental file structure used for keeping index is trie which supports both

word, pattern, and approximate matching. A word segmentation algorithm was also proposed for segmenting a given text to a set of words and sistrings of unknown words for adding to the trie. The algorithm finds a smallest set of words and sistrings with the objective of minimizing number of sistrings to enhance retrieval precision. While the use of sistring for unknown words enhances the retrieval recalls.

References

[1] S. Charnyapornpong, "A Thai Syllable Separation Algorithm," M.Eng. Thesis, Asian Institute of Technology, Aug. 1983.

[2] C. Faloutsos and D.W. Oard, "A Survey of Information Retrieval and Filtering Methods," Technical Report CS-TR-3514, University of Maryland, College Park, August 1995.

[3] W. B. Frakes and R. Baeza-Yates eds., *Information Retrieval : Data Structures and Algorithms*, Englewood Cliffs, N.J. : Prentice-Hall.

[4] G. Gonnet, "Unstructured Data Bases or Very Efficient Text Searching," *ACM PODS*, vol. 2, pp. 117-124, 1983.

[5] G. Gonnet, R. Baeza-Yates, and T. Snider "New Indices for Text: PAT Trees and PAT Arrays," in *Information Retrieval : Data Structures and Alforithms*, ed., W. B. Frakes and R. Baeza-Yates, Englewood Cliffs, N.J. : Prentice-Hall

[6] P. Jindavimonlert, "A Thai Text Retrieval System using the PAT tree," M.Sc. Thesis, Department of Computer Engineering Chulalongkorn University, 1996.

[7] A. Kawtrakul, C. Thumkanon, and S. Seriburi, "A Statistical Approach to Thai Word Filtering," Proc. of the second Symposium on Natural Language Processing, pp. 398-406, 1995

[8] U. Manber and G. Myers, "Suffix Arrays: A New Method for On-line String Searches," *First ACM-SIAM Symp. on Discrete Algorithms*, pp. 319-327, San Francisco, 1990.

[9] T.H. Merrett and H. Shang, "Trie Methods for Representing Text," *Proc Fourth Int'l Conf., FODO'93*, LNCS 730, pp. 130-145, Chicago: Springer-Verlag, Oct. 1993.

[10] G. Salton and M.J. McGill, *Introduction to Modern Information Retrieval*. McGraw-Hill, 1983

[11] H. Shang, "Trie Methods for Text and Spatial Data on Secondary Storage," Ph.D. Dissertation, School of Computer Science, McGill University, Nov. 1994.

[12] H. Shang and T.H. Merrett, "Tries for Approximate String Matching," *IEEE Trans. on Knowledge and Data Eng.*, Vol. 8, No. 4, pp. 540-547, Aug. 1996.

[13] D. Sintupunpratum and C. Bandhitanont, "Thai Word Processing (in Thai)", Proc. of the second Symposium on Natural Language Processing in Thailand, pp. 322-376, March 1993.

[14] I.H. Witten, A. Moffat, and T.C. Bell, *Managing Gigabytes : Compressing and Indexing Documents and Images*, N.Y., Van Nostrand Reinhold.

[15] D. Sawamibhadhi, "Implementation of Thai Grammar Analysis Software under UNIX system (in Thai)", Thammasart Univ., 1990.

[16] Y. Poovorawan and V. Imarom, "Dictionary-based Thai Syllable Segmentation (in Thai)," 9th Electrical Engineering Conference, 1986.

[17] V. Sornlertlamvanich, "Thai Word Segmentation in Language Translation System," *Computerized Language Translation* (in Thai), p. 50-55, 1993.

Thai-English Cross-Language Transliterated Word Retrieval using Soundex Technique *

Prayut Suwanvisat
Graduate Student
g40psw@cp.eng.chula.ac.th

Somchai Prasitjutrakul
Assistant Professor
somchaip@chula.ac.th

Department of Computer Engineering
Chulalongkorn University
Bangkok 10330, Thailand
Tel : (66-2)-218-6981
Fax : (66-2)-218-6955

Abstract

This paper presents an algorithm for Thai-English cross-language transliterated word retrieval. The algorithm enables retrieval of documents containing either the English keywords or the corresponding English-to-Thai transliterated words. This is done by retrieving documents based on phonetic codes of keywords rather than the keywords themselves. The phonetic coding is based on the Soundex coding of Odell and Russell where the encoding table is slightly modified to incorporate Thai characters and the code is extended to unlimited length. Experimental results showed that a high recall and precision of more than 80% can be achieved especially when the phonetic codes are longer than four.

1. Introduction

Text retrieval has become one of the most essential tools for managing information as computer-generated documents get published and computers get connected locally and globally, especially with the advent of the World Wide Web and CD-ROM. The effectiveness of any text retrieval system is commonly measured in terms of precision and recall [3] where precision is the ratio of the number of relevant documents retrieved over the total number of documents retrieved, and recall is the ratio of relevant documents retrieved for a given query over the number of relevant documents for the query in the textbase. One problem arises when a user enter query keywords that are in one language where the documents to be managed are in another. For example, searching for documents containing "*ALEXANDER*" does not return documents containing "*อเล็กซานเดอร์*" (the corresponding Thai transliterated word). This problem causes recall of the retrieval to be lower than it should be.

Here we are interested in Thai-English cross-language transliterated word retrieval. Cross-language information retrieval is defined as the retrieval of documents when the language in which the documents are expressed is not the same as the language in which the queries are expressed [7]. It is very common in Thai documents that most of English proper nouns and technical terms appearing in the documents are either in English or transliterated into Thai. Using bilingual dictionaries as thesauri of the retrieval system does not solve the problem since most of the transliterated words are not found in the dictionaries [4]. Therefore, querying using one language will miss documents containing corresponding keywords in the other language if the system does not support the cross-language feature.

There are previous researchs working on the problem. The algorithms presented in [5] and [6] transcribe English and Japanese words into intermediate codes and use exact code matching during retrieval. Since transliterating the two languages back and forth loses some information, two corresponding words may not be exactly matched . Whereas the algorithm in [2] encodes each Katakana word into a phonetic string representation and uses partial matching with English words. Two words are considered to be in transliteration relation when the number of matched characters is more than a certain threshold. The algorithm uses a depth-first search which trends to take longer time than a straightforward matching so that some heuristics are incorporated to reduce search time. [8] presents an algorithm for encoding English word to a set of possible Thai sounds using a set of encoding tables and rules. The encoding tables are not fully elaborated and no details on effectiveness of the methods are reported.

In this paper, we present an algorithm supporting Thai-English cross-language transliterated word retrieval. In other words, the system enables retrieval of documents containing either the English keywords or the corresponding English-to-Thai transliterated words. By slightly modifying coding table and algorithm of the Odell and Russell's Soundex code [1], a higher recall on document for cross-language retrieval queries on transliterated words is achieved with good precision. The rest of the paper is organized as follows. Section 2 explains the Odell and Russell's Soundex coding and algorithm. Section 3 presents our new modified encoding

* This research was supported by the Thai Government Research Fund.

algorithm with experimental results presented in Section 4. Then the paper is concluded in Section 5.

2. Odell and Russell's Soundex Algorithm

M. K. Odell and R. C. Russell designed a system to encode names based on their pronunciation so that names that sound alike would have the same phonetic code. Their system is called Soundex [1]. The system is based on the idea that English names can be distinguished based only on the consonants. It constructs the phonetic code by converting each letter (ignoring the leftmost letter) into a numeric code using the coding table shown in Table 1. Then all the zeros are removed and any runs of the same digits are reduced to one digit. The final Soundex code is the first (left-most) letter of the word followed by the first three digits of the converted code. For example, ALEXANDER is converted to A425.

Table 1. Soundex Coding Table

Letter	*Numeric code*
AEIOUHWY	0
BFPV	1
CGJKQSXZ	2
DT	3
L	4
MN	5
R	6

The Soundex system is fast and usually matches names that it should find, but often causes false hit i.e., incorrectly matches names that are not actually sound alike.

3. Our Proposed Encoding Method

There are 44 consonants in Thai which can be categorized into 21 phonetic groups as shown in Table 2 [9]. To incorporate Thai letters into the Soundex coding table, we can assign an English letter to each group having the similar phonetic, then the 21 groups are further grouped into seven groups according to phonetic similarity of the letters in the Odell and Russell's Soundex coding table as shown in Table 3. We propose to modify the original Soundex coding (shown in Table 4) as follows :

- use numeric representation for the first letter rather than the letter itself in the code. This is due to the fact that there are many cases where more than one English letter can be mapped to the same Thai letter [10], e.g., V and W are mapped to ว. as shown as the note #2 in Table 4. Therefore we need to introduce three more numbers (7, 8, 9) in the table.
- add another numeric code for the Thai letter ง since this is the only one left unencoded from the Thai consonant. The transliteration to ง is normally from the letters NG or NK e.g. KING is transliterated to คิง. Therefore its corresponding code is 52 (5 is for N, and 2 is for both G and K).
- extend the code length to be unlimited rather than of length four in the original Soundex code. (This is for enchancing precision but potentially will reduce recall of retrieval.) And also set the minimum limit of code length to be *k*, i.e, that is to only considered words having code length of length not less than *k*, where *k* is a parameter to be determined later.

Note that all Thai vowels and tones are all ignored as being done in English for the Soundex coding. (Remember that we mainly concern with the English to Thai transliteration.)

Table 2. The 21 phonetic groups of Thai consonants

ก	ฎ ด	ฝ ฟ
ข ฃ ค ฅ ฆ	ฏ ต	ม
ง	ฐ ท ฒ ถ ท ธ	ร
จ	ณ น	ล ฬ
ฉ ช ฌ	บ	ว
ซ ศ ษ ส	ป	ห ฮ
ญ ย	ผ พ ภ	อ

Table 3. The seven similarly phonetic groups according to Soundex code table

English	*Thai*
AEIOUHWY	อ ห ฮ ว ญ ย
BFPV	บ ฝ ฟ ป ผ พ ภ ว
CGJKQSXZ	ข ฃ ค ฅ ฆ ฌ ฉ ช ฌ ณ ก จ ซ ศ ษ ส
DT	ฎ ด ฏ ต ฐ ท ฒ ถ ท ธ
L	ล ฬ
MN	ม ณ น
R	ร

Table 4. The new modified Thai/English coding table

English	*Thai*	Code	Note
AEIOUHWY	อ	0	#1
BFPV	บ ฝ ฟ ป ผ พ ภ ว	1	
CGJKQSXZ	ข ฃ ค ฅ ฆ ฌ ฉ ช ฌ ณ ก จ ซ ศ ษ ส	2	
DT	ฎ ด ฏ ต ฐ ท ฒ ถ ท ธ	3	
L	ล ฬ	4	
MN	ม ณ น	5	
R	ร	6	
AEIOU	อ	7	#2
H	ห ฮ	8	#2
W	ว	1	#2
Y	ย ญ	9	#2
	ง	52	

#1 : for the first (leftmost) letter of the word
#2 : for the second letter and the rest of the word

4. Experimental Results

We implemented the encoding algorithm presented in the previous section by slightly modifying the encoding table and algorithm in the Soundex coding function in [1]. Then the algorithm is tested using a set of 1,902 pairs of English and English-to-Thai transliterated words which

are mostly proper names (brand names, country names, scientist and mathematician names, etc.) and technical terms in science, mathematics, and chemistry obtained from [10], [11], [12], and [13].

We ran experiments by having all the words (and their corresponding phonetic codes presented in the previous section) stored in the database and then querying all of the words, one by one, to measure recall and precision of the retrievals. The experiments were tested repeatedly by varying the minimum limit of code length, k presented in the previous section, in order to show how the minimum limit of code length affects effectiveness of the retrieval. The experimental results are shown in Figure 1. In the set of tested words, the number of words whose code length is more than seven accounts for only 1% so we do not include the results in the plots which may mislead the interpretation.

From the plot in Figure 1, we can notice that recall of the retrieval is very high around 90% and then starts to drop slowly as the minimum limit of code length increases. This is in the opposite behavior for the precision which starts at around 45% and then climbs sharply to around 80% as we increase the minimum limit of code length. This observed behavior is the nature of recall and precision which grows in the opposite direction. However, in this particular problem and proposed coding algorithm, the recall slowly declines as precision gets sharply improved by increasing the minimum limit of code length especially after $k > 4$. So the proposed algorithm is generally effective for words having code length greater than 4, i.e., for words of length longer than approximately seven characters (this is concluded by an observation from the length of tested words and their corresponding code lengths).

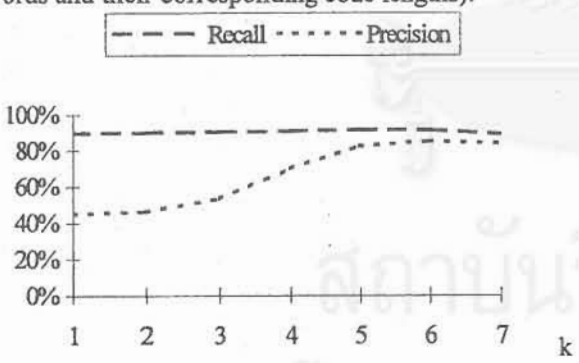


Figure 1 Plots of recall and precision against the minimum code length

5. Conclusion

In this paper, we presented an algorithm for Thai-English cross-language transliterated word retrieval. The retrieval is done by using phonetic codes retrieval based on a modified Soundex coding rather than searching for the words themselves. The system enables retrieval of documents containing either the English keywords or the corresponding English-to-Thai transliterated words. Words that are generally get retrieved by this approach are proper nouns and technical terms. In the proposed algorithm, we modify the Soundex coding algorithm of Odell and Russell by slightly modifying the encoding table and extending the code to unlimited length. Experimental results showed that a high recall and precision of more than 80% can be achieved especially when the phonetic codes are longer than four.

References

[1] A. Binstock and J. Rex, *Practical Algorithms for Programmers*, Addison Wesley, 1995.

[2] N. Collier, A. Kumano, and H. Hirakawa, "Acquisition of English-Japanese proper nouns from noisy-parallel newswire article using Katakana matching", *Proc. of the Natural Language Processing Pacific Rim Symposium 1997*, Phuket, Thailand, Dec. 2-4, pp. 309-320.

[3] W. Frakes, "Introduction to Information Storage and Retrieval System", *Information Retrieval : Data Structures & Algorithms*, W.B. Frakes and R. Baeza-Yates ed., Prentice Hall, 1992.

[4] K. Knight and J. Graehl, "Machine Transliteration", *Annual Meeting of the Association for Computational Linguistics (ACL-97/EACL-97)*.

[5] A. Kumano, "Building a technical term dictionary with Katakana-English Matching", *Gengoshorigakai - Annual Conf. of the Japanese Association for Natural Language Processing*, (in Japanese) Japan, March, pp.221-223.

[6] Y. Matsuo and S. Shirai, "Using pronunciation to automatically extract bilingual word pairs", *Shizengengoshori*, (in Japanese), November, pp.101-106.

[7] D. Oard and B. Dorr, "A Survey of Multilingual Text Retrieval", *Technical Report UMIACS-TR-96-19 CD-TR-3615*, University of Maryland, College Park, April 1996.

[8] S. Ongroongruang, R. Prongsirivattana, and V. Jantarasukree, "English to Thai Word Retrieval Using Sound Index", *Proc. 2nd SNLP'95*, Bangkok Thailand, Aug. 2-4, 1995, pp. 407-413.

[9] P. Prapapitayakorn and et. al., *รู้จักภาษาไทย* Odien Bookstore, 1976

[10] Royal Academy, *หลักเกณฑ์การทับศัพท์* (Transliteration Guideline), 1992.

[11] Royal Academy, *ศัพท์วิทยาศาสตร์* (Science Dictionary), 1993.

[12] Royal Academy, *ศัพท์คณิตศาสตร์* (Mathematics Dictionary), 1997.

[13] Royal Academy, *หนังสือเรียนวิชาเคมี เล่ม 1 หลักสูตรมัธยมปลาย 2524* (Chemistry Book 1: High School Level 1981), 1987.