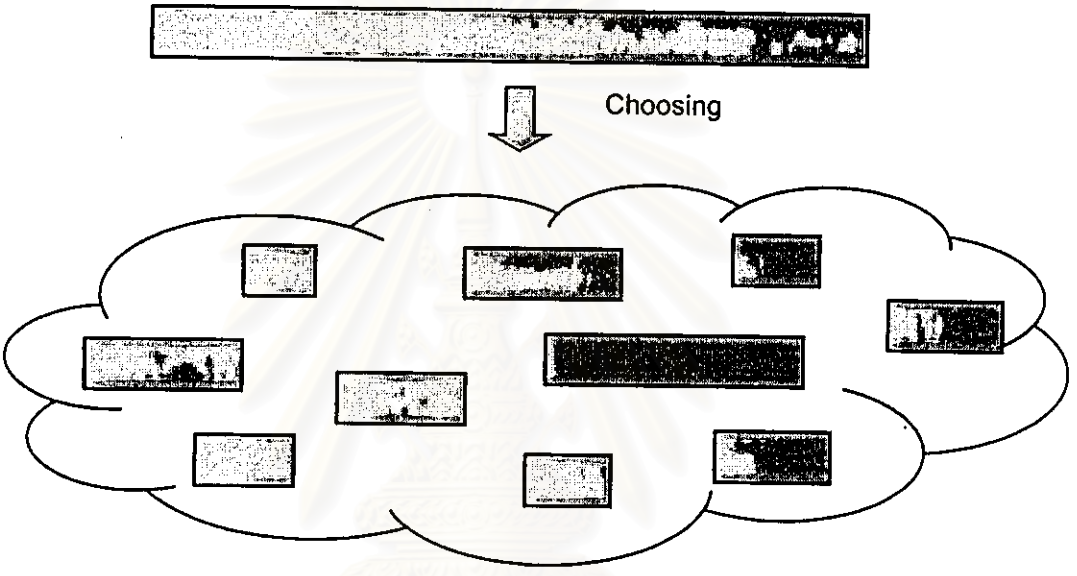
บทที่ 3

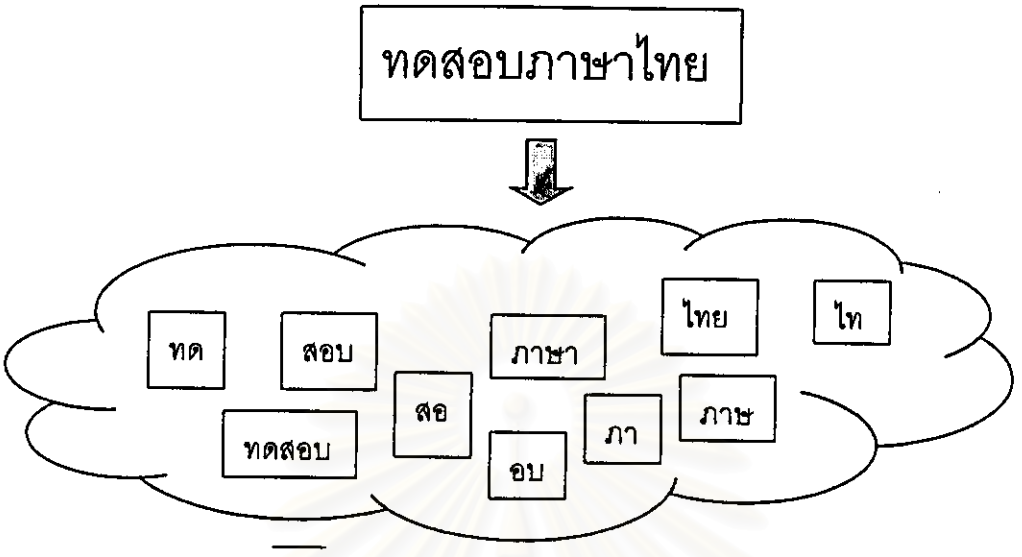
แนวคิดและการออกแบบอัลกอริทึมการจัดทำดัชนีภาษาไทย

แนวความคิดที่ 1



รูปที่ 3.1 แนวความคิดที่ 1

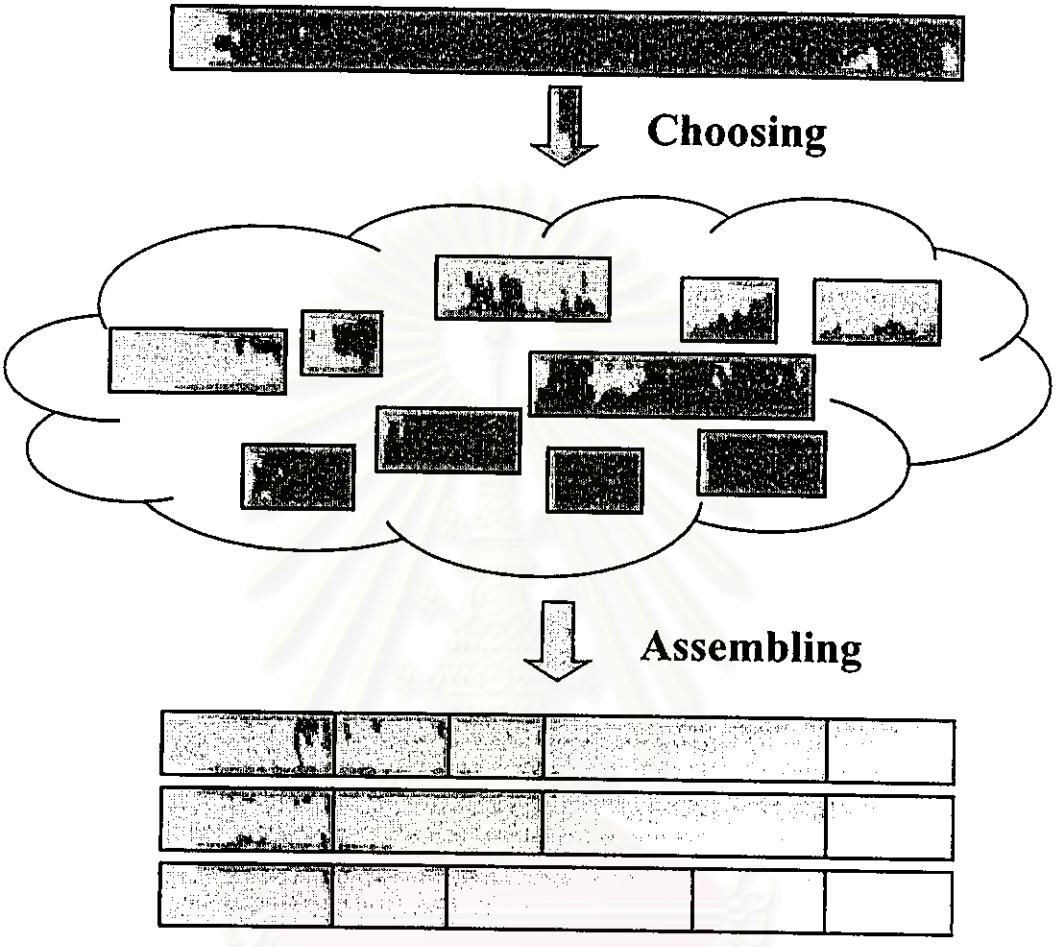
การแบ่งคำภาษาไทยโดยใช้พจนานุกรมนั้น วิธีการที่ง่ายที่สุดในการแบ่งคำ คือการดึงคำทุกคำที่เป็นไปได้ออกมาให้หมด ตัวอย่างเช่น คำว่า "ทดสอบภาษาไทย" สามารถดึงคำออกมาได้ดังรูปที่ 3.2



รูปที่ 3.2 คำที่สามารถดึงออกได้จากข้อความ "ทดสอบภาษาไทย"

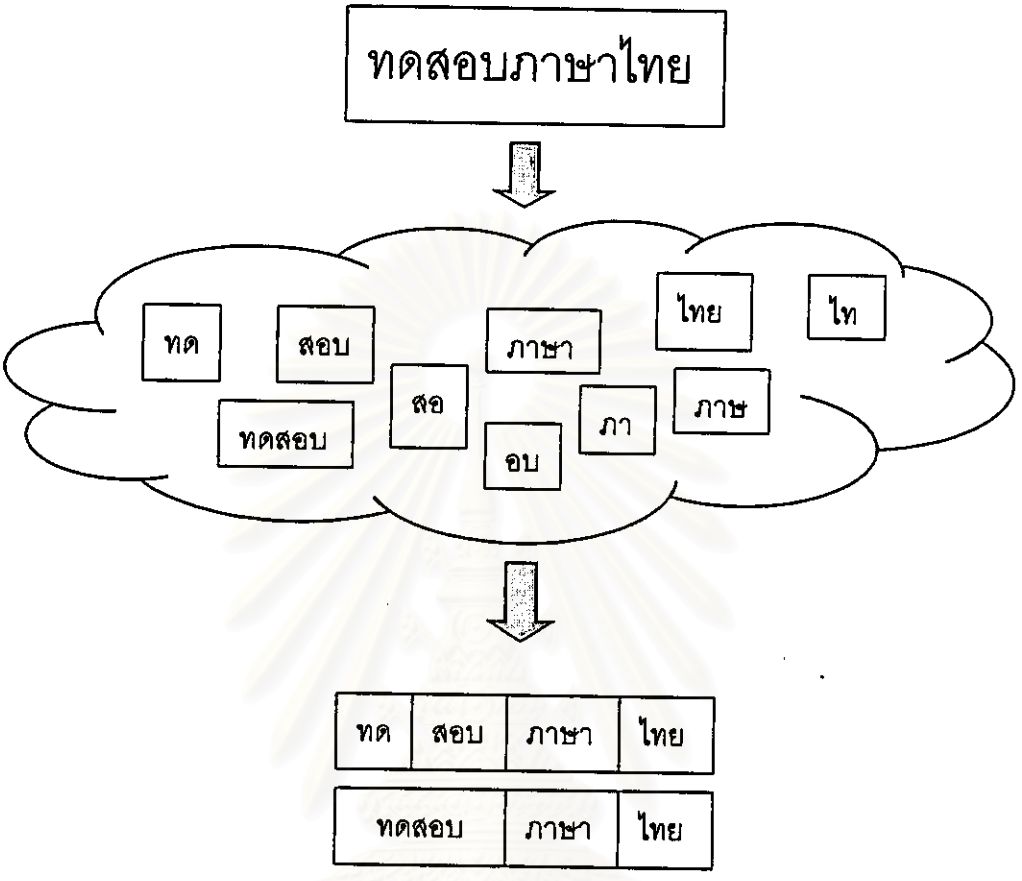
จากรูปที่ 3.2 คำที่นำมาทำจัดทำดัชนีมีดังนี้ "ทด", "สอบ", "ทดสอบ", "สอ", "อบ", " ภาษา", "ภาษ", "ภา", "ไทย" และ "ไท" คำที่ได้มีมากเกินความต้องการ ทำให้ประสิทธิภาพของระบบสืบค้นข้อมูลไม่ดี เนื่องจากมีคำหลักมาก เวลาในการสร้างดัชนีก็จะมาก รวมถึงในการสืบค้นข้อมูล จะได้ผลลัพธ์ที่ไม่มีประสิทธิภาพ

แนวความคิดที่ 2



รูปที่ 3.3 แนวความคิดที่ 2

จากแนวความคิดที่ 1 จะสังเกตได้ว่า คำที่ควรดึงออกมานั้นน่าจะเป็นคำที่สามารถต่อกับคำอื่นๆ ได้พอดี จากตัวอย่างข้างต้น คำว่า "สอ" ไม่สามารถหาคำที่ขึ้นต้นด้วยตัวอักษร "บ" ได้ ดังนั้นคำว่า "สอ" จึงไม่ควรแบ่งคำออกมาได้ เช่นเดียวกับคำว่า "อบ" ไม่สามารถหาคำที่อยู่หน้าและลงท้ายด้วยตัวอักษร "ส" ได้ จึงไม่ควรแบ่งคำออกมาได้ รูปที่ 3.3 แสดงผลที่ได้จากแนวความคิดนี้



รูปที่ 3.4 คำที่ได้จากการข้อความ "ทดสอบภาษาไทย"

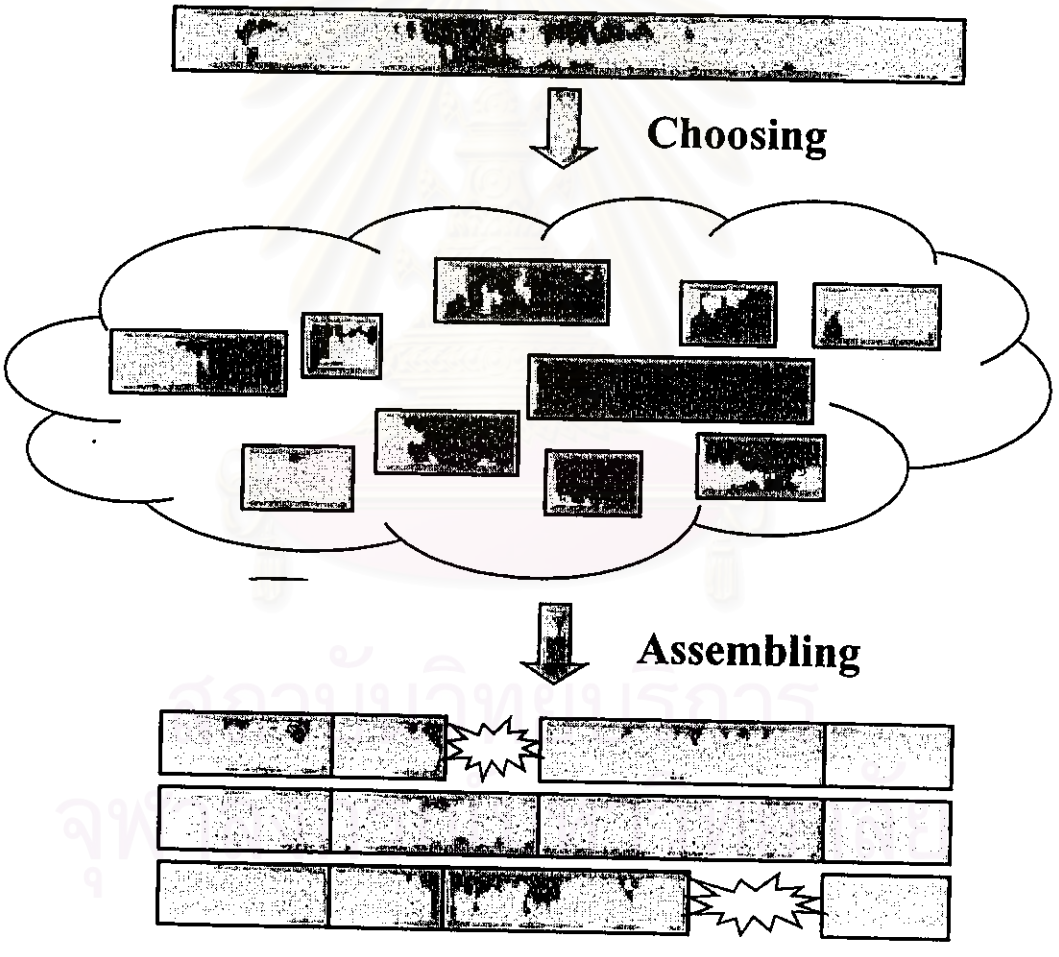
การนำวิธีนี้ไปใช้สามารถใช้สามารถสร้างฟังก์ชั่นการแบ่งคำ ดังนี้

```
int Sep( char *str , int doc , int pos );
{
  if( strlen(str) == 0 ) {
    Return TRUE and exit funciton
  } else {
    result = FALSE
    for( i = strlen(str)-1 ; i > 0 ; i--) {
      if( str0-i in dict )
  if( Sep(stri+1 ,doc,pos+i+1) == TRUE ) {
          Setposting( str0-i , doc , pos )
          result = TRUE
        }
    }
  }
  return result
```

}

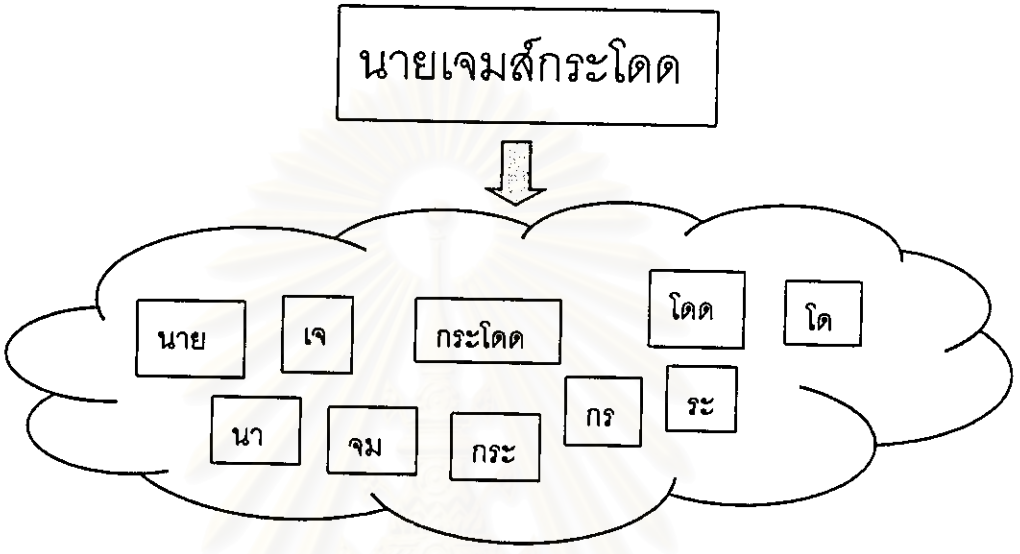
แนวความคิดที่ 3

จากวิธีที่กล่าวไว้ข้างต้นทั้ง 2 วิธี สามารถทำงานได้กับข้อความที่มีคำศัพท์อยู่ในพจนานุกรมทั้งหมด แต่ในกรณีที่ภายในข้อความมีคำศัพท์ที่ไม่มีอยู่ในพจนานุกรม จะไม่สามารถใช้วิธีการที่กล่าวไว้ข้างต้นได้ แต่ถ้านำวิธีการแรกคือการดึงคำทุกคำที่เป็นไปได้ออกมาแล้วนำกลุ่มตัวอักษรที่ไม่รู้จัก (ไม่รู้จักในที่นี้ คือไม่มีในพจนานุกรม) นำออกมาเก็บเป็นคำหลักด้วย ดังรูปที่ 3.5



รูปที่ 3.5 แนวความคิดที่ 3

ตัวอย่างเช่น "นายเจมส์กระโดด" สามารถดึงคำออกได้ดังรูปที่ 3.6



รูปที่ 3.6 คำที่ได้จากข้อความ "นายเจมส์กระโดด"

จากรูปที่ 3.6 สังเกตได้ว่าคำว่า "เจมส์" ไม่สามารถแบ่งคำออกได้ เนื่องจากในพจนานุกรมไม่มีคำดังกล่าวอยู่ แต่เมื่อนำเอาคำที่ได้มาจัดเรียงกันเพื่อให้ได้เป็นข้อความเดิม ซึ่งสามารถจัดเรียงได้หลายแบบดังนี้

นาย + เจ + มส์ + กระโดด
นาย + เจ + มส์ + กระ + โดด
นาย + เจ + มส์ + กระ + โด + ด
นาย + เจ + มส์ก + ระ + โดด
นาย + เจ + มส์ก + ระ + โด + ด
นาย + เ + จม + ส์ + กระโดด
นาย + เ + จม + ส์ + กระ + โดด
นาย + เ + จม + ส์ + กระ + โด + ด

นาย + เ + จม + ส์ก + ระ + โดด

นาย + เ + จม + ส์ก + ระ + โด + ด

ตัวอักษร หรือกลุ่มตัวอักษรที่ขีดเส้นใต้ แสดงถึงสิ่งที่เพิ่มเข้าไปเพื่อให้ข้อความสมบูรณ์ ดังนั้นคำหลักที่จะนำไปทำดัชนีได้แก่คำว่า "นาย", "เจ", "มส์", "กระโดด", "กระ", "โดด", "โด", "ด", "มส์ก", "ระ", "เ", "ส์" และ "ส์ก"

แนวความคิดที่ 4

วิธีที่ 3 สามารถปรับปรุงให้คำหลักที่จะนำไปทำดัชนี เป็นคำที่มีลักษณะเป็นคำ ไม่ใช่ตัวอักษรตัวเดียว หรือเป็นกลุ่มตัวอักษรที่ไม่น่าจะเป็นคำได้ ซึ่งจะทำให้ประสิทธิภาพของระบบสืบค้นข้อมูลดีขึ้น คือการนำเอากลุ่มตัวอักษร หรือตัวอักษรที่ต้องเก็บเพิ่มเติมจากวิธีที่ 1 นำมารวมกับคำที่อยู่หน้าและหลัง ทำให้เกิดเป็นคำขึ้นมาใหม่ จากตัวอย่างคำว่า "นายเจมส์กระโดด" วิธีที่ 3 สามารถแสดงได้ดังนี้

นาย + เจ + มส์ + กระโดด

"มส์" รวมคำหน้าเป็น "เจมส์" รวมคำหลังเป็น "มส์กระโดด"

นาย + เจ + มส์ + กระ + โดด

"มส์" รวมคำหน้าเป็น "เจมส์" รวมคำหลังเป็น "มส์กระ"

นาย + เจ + มส์ + กระ + โด + ด

"มส์" รวมคำหน้าเป็น "เจมส์" รวมคำหลังเป็น "มส์กระ"

"ด" รวมคำหน้าเป็น "โดด"

นาย + เจ + มส์ก + ระ + โดด

"มส์ก" รวมคำหน้าเป็น "เจมส์ก" รวมคำหลังเป็น "มส์กระ"

นาย + เจ + มส์ก + ระ + โด + ด

"มส์ก" รวมคำหน้าเป็น "เจมส์ก" รวมคำหลังเป็น "มส์กระ"

"ด" รวมคำหน้าเป็น "โดด"

นาย + เ + จม + ส์ + กระโดด

"เ" รวมคำหน้าเป็น "นายเ" รวมคำหลังเป็น "เจม"

"ส์" รวมคำหน้าเป็น "จมส์" รวมคำหลังเป็น "ส์กระโดด"

นาย + เ + จม + ส์ + กระ + โดด

"เ" รวมคำหน้าเป็น "นายเ" รวมคำหลังเป็น "เจม"

"ส์" รวมคำหน้าเป็น "จมส์" รวมคำหลังเป็น "ส์กระ"

นาย + เ + จม + ส์ + กระ + โด + ด

"เ" รวมคำหน้าเป็น "นายเ" รวมคำหลังเป็น "เจม"

"ส์" รวมคำหน้าเป็น "จมส์" รวมคำหลังเป็น "ส์กระ"

"ด" รวมคำหน้าเป็น "โดด"

นาย + เ + จม + ส์ก + ระ + โดด

"เ" รวมคำหน้าเป็น "นายเ" รวมคำหลังเป็น "เจม"

"ส์ก" รวมคำหน้าเป็น "จมส์ก" รวมคำหลังเป็น "ส์กระ"

นาย + เ + จม + ส์ก + ระ + โด + ด

"เ" รวมคำหน้าเป็น "นายเ" รวมคำหลังเป็น "เจม"

"ส์ก" รวมคำหน้าเป็น "จมส์ก" รวมคำหลังเป็น "ส์กระ"

"ด" รวมคำหน้าเป็น "โดด"

คำหลักที่ได้ของข้อความ "นายเจมส์กระโดด" มีคำว่า "นาย", "นายเ", "เจ", "เจม", "เจมส์", "เจมส์ก", "จม", "จมส์", "จมส์ก", "มส์กระโดด", "มส์กระ", "ส์กระโดด", "ส์กระ", "กระโดด", "กระ", "ระ", "โดด" และ "โด" ถ้าเพิ่มความรู้เรื่องของคำภาษาไทยเข้าไปสามารถตัดคำที่ไม่ใช่คำในภาษาไทยออกได้คือคำว่า "นายเ", "มส์กระโดด", "มส์กระ", "ส์กระโดด" และ "ส์กระ" เพราะคำว่า "นายเ" มีอักษร "เ" อยู่ท้ายคำซึ่งคำในภาษาไทยอักษร "เ" เป็นสระที่ต้องมีอักษรต่อท้ายเสมอ คำว่า "มส์กระโดด" และคำว่า "มส์กระ" มีอักษร "ม" นำหน้า "ส์" เพียงตัวเดียวไม่ได้ต้องมีอย่างน้อย 2 ตัว คำว่า "ส์กระ" เช่นกันที่ไม่ใช่คำเพราะไม่มีตัวอักษรนำหน้า "ส์" ดังนั้นคำหลักที่จะต้องนำไปทำดัชนีมีคำว่า "นาย", "เจ", "เจม", "เจมส์", "เจมส์ก", "จม", "จมส์", "จมส์ก", "กระโดด", "กระ", "ระ", "โดด" และ "โด"

แนวความคิดนี้เมื่อนำไปสร้างอัลกอริทึม ส่วนที่ยากในการสร้างอัลกอริทึมคือส่วนของการวิเคราะห์ว่าคำ ว่าคำไหนไม่ใช่คำ และยังมีปัญหาเกิดขึ้น พิจารณาจากหัวข้อการดึงคำภาษาไทยโดยใช้พจนานุกรมในกรณีที่ 2 แบบที่ 3 ในบทที่ 2 จะสังเกตได้ว่าถ้าเกิดมีคำที่มีในพจนานุกรมเกิดขึ้นในคำที่ไม่มีในพจนานุกรม จะทำให้เกิดกลุ่มคำที่ไม่รู้จักขึ้นเป็น 2 ส่วน ถึงแม้จะใช้การวิธีตามแนวความคิดที่ 4 นี้ก็ไม่สามารถรวมเป็นคำขึ้นมาได้อย่างสมบูรณ์

แนวความคิดที่ 5

จากแนวความคิดที่เสนอไปข้างต้น ไม่มีแนวความคิดไหนที่จะให้ผลได้ดีตามที่ต้องการ แนวความคิดที่ 5 นี้คือการพยายามแยกส่วนของคำที่มีอยู่ในพจนานุกรม และคำที่ไม่มีอยู่ในพจนานุกรม ออกจากกัน วิธีในการค้นหาคำที่มีอยู่ในพจนานุกรมจะใช้การค้นหาคำที่มีความยาวมากที่สุด เนื่องจากจะทำให้มีจำนวนคำที่จะนำไปทำดัชนีน้อยลง ทำให้ประสิทธิภาพในการทำดัชนีดีขึ้น ส่วนคำที่ไม่มีอยู่ในพจนานุกรม สามารถสร้างกลุ่มคำที่ไม่รู้จักขึ้นมา โดยอาศัยกฎในการตัดพยางค์เข้าช่วย โดยการทำงานของอัลกอริทึมจะอธิบายต่อไปนี้

อัลกอริทึมการจัดทำดัชนีภาษาไทย

กำหนดให้

T คือ ข้อความที่ต้องการแบ่งคำ

T_i คือ sistring (ภาคผนวก ค.) ของ T ที่มีตำแหน่งเริ่มต้นที่ตำแหน่ง i

$T_{i,j}$ คือ substring ของ T ที่มีตำแหน่ง i ถึงตำแหน่ง j

D คือพจนานุกรม

อัลกอริทึมแบ่งออกได้ 4 ขั้นตอนดังนี้

ขั้นตอนที่ 1

แต่ละ T_i โดย $i = 1, \ldots, n$ ค้นหาคำหลัก w_i ใน D ที่มีเงื่อนไขดังต่อไปนี้

1. w_i มีขนาดใหญ่ที่สุดของ T_i กำหนด $i' = i - 1 +$ ความยาวของ w_i ดังนั้น $w_i = T_{i,i'}$
2. w_i ไม่ใช่ substring ของทุกๆ w_j โดย $j < i$

ตัวอย่างเช่น กำหนด T = "นายเจมส์มาร์ตินต้องการผลิตรายการโทรทัศน์" ได้ w_i แสดงในตารางที่ 3.1 (sistring ที่ไม่สามารถหาค่าของ w_i ได้จะไม่นำมาแสดงในตาราง)

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

ตารางที่ 3.1 ตัวอย่างของ w_i ที่ได้จาก sistring

ขั้นตอนที่ 2

สร้างกราฟการต่อและทับกันของคำ G (V,E) ซึ่งเป็นกราฟแบบมีทิศทางและน้ำหนัก โดยที่

$V = \{ w_i \mid w_i$ คือผลจากขั้นตอนที่ 1, $w_i \neq \emptyset$, $1 \leq i \leq n \}$

$E = \{ (w_i, w_j) \mid w_i$ ต่อกันพอดี หรือทับกันกับ w_j , $i < j \}$

น้ำหนักของ (w_i , w_j) กำหนดกรณีในตารางที่ 3.2

ในกรณีแรกเป็นกรณีที่พบมากที่สุด คือคำสองคำต่อกันพอดี กรณีที่ 2 คำสองคำทับกันบางส่วน แต่ส่วนที่ไม่ได้ทับกันของทั้งสองคำนั้นเป็นคำที่มีความหมาย หรือคำหลักด้วย ตัวอย่างเช่น "ต้องการผลิต" (w_i = "ต้องการ" และ w_j = "การผลิต") สามารถแบ่งคำได้เป็น "ต้อง" + "การผลิต" หรือ "ต้องการ" + "ผลิต" กรณีที่ 3 คำสองคำทับกัน แต่ต้องมีอย่างน้อยหนึ่งรูปแบบในการแบ่งที่เป็นคำที่มีความหมาย ตัวอย่างเช่น "เพื่อนำ" (w_i = "เพื่อน" และ w_j = "อนำ") สามารถแบ่งได้เป็น "เพื่อ" + "นำ", "เพื่อน" + "ำ" หรือ "เพื่" + "อนำ" กรณีสุด

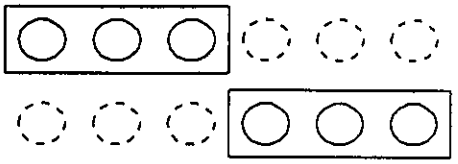
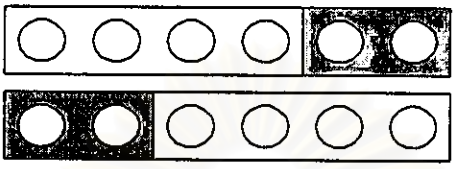
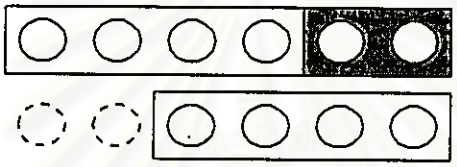
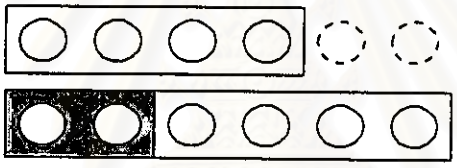
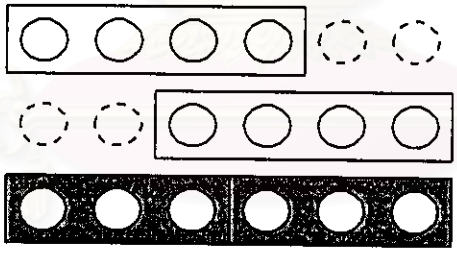
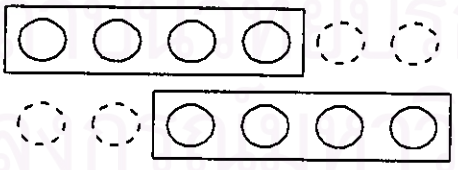
ท้าย (มีค่าน้ำหนักสูงสุด) คือกรณีที่มีการทับกันแต่ไม่อยู่ในกรณีที่ 2 และ 3 ข้อสังเกต ในกรณีที่ 3 และ 4 นั้นจะเป็นการทำให้สามารถควบคุมการผิดพลาดของข้อความได้ เช่น "เพื่อนำ" อาจเป็นคำว่า "เพื่อนทำ" ซึ่งมีการพิมพ์ตัวอักษร "ท" ขาดไป ในลักษณะเช่นนี้ในกรณีที่ 3 จะได้คำว่า "เพื่อน" และได้คำว่า "เพื่อ" และ "นำ" เพิ่มขึ้นอีก

หลักเกณฑ์ในการให้ค่าน้ำหนักในแต่ละกรณีนั้น จากที่ได้อธิบายไว้ข้างต้น ในกรณีที่ 1 เป็นกรณีที่ดีที่สุด ค่าของน้ำหนักจึงมีค่าน้อยสุด กรณีที่ 2 ดีเป็นอันดับที่ 2 จึงมีค่ามากกว่าในกรณีที่ 1 ส่วนในกรณีที่ 3 ดีเป็นอันดับ 3 จึงในค่าของน้ำหนักมากเป็นอันดับ 3 และในกรณีที่ 4 ซึ่งเป็นกรณีที่ไม่ดีที่สุด จะมีค่าของน้ำหนักมากที่สุด การกำหนดค่าของน้ำหนักในกรณีที่ 1 ให้มีค่าเท่ากับ 1 ในกรณีที่ 2 ให้มีค่าเท่ากับ 10 ไม่ได้หมายความว่า กรณีที่ 1 ดีกว่ากรณีที่ 2 10 เท่า หรือถ้าเกิดกรณีที่ 1 10 ครั้ง เลือกกรณีที่ 2 แทนได้ แต่การกำหนดตัวเลขเช่นนี้ เพื่อต้องการให้ดูง่าย ก่ารนำไปเขียนโปรแกรมสามารถทำได้ง่าย ในความเป็นจริงถ้าเกิดมีกรณีที่ 1 และกรณีที่ 2 ให้เลือกจะต้องเลือกกรณีที่ 1 เสมอ เช่นเดียวกับกรณีอื่นๆ คือกรณีที่ 2 ไม่ได้ดีกว่ากรณีที่ 3 10 เท่า และถ้าเกิดกรณีที่ 2 10 ครั้งก็ไม่สามารถเลือกกรณีที่ 3 แทนได้ แต่จะต้องเลือกกรณีที่ 2 เสมอ

กรณี	น้ำหนักของ (w_i, w_j)	เงื่อนไข
1	1	ถ้า $i' + 1 = j$ (w_i ต่อกันพอดีกับ w_j)
2	10	ถ้า $j \leq i'$ และ $T_{i,j-1}$, $T_{j,i'}$, $T_{i'+1,j}$ อยู่ในพจนานุกรม
3	100	ถ้า $j \leq i'$ และมี k ที่ $j-1 \leq k \leq i'$ โดย $T_{i,k}$ และ $T_{k+1,j'}$ อยู่ในพจนานุกรมทั้งสองคำ
4	1000	อื่นๆ

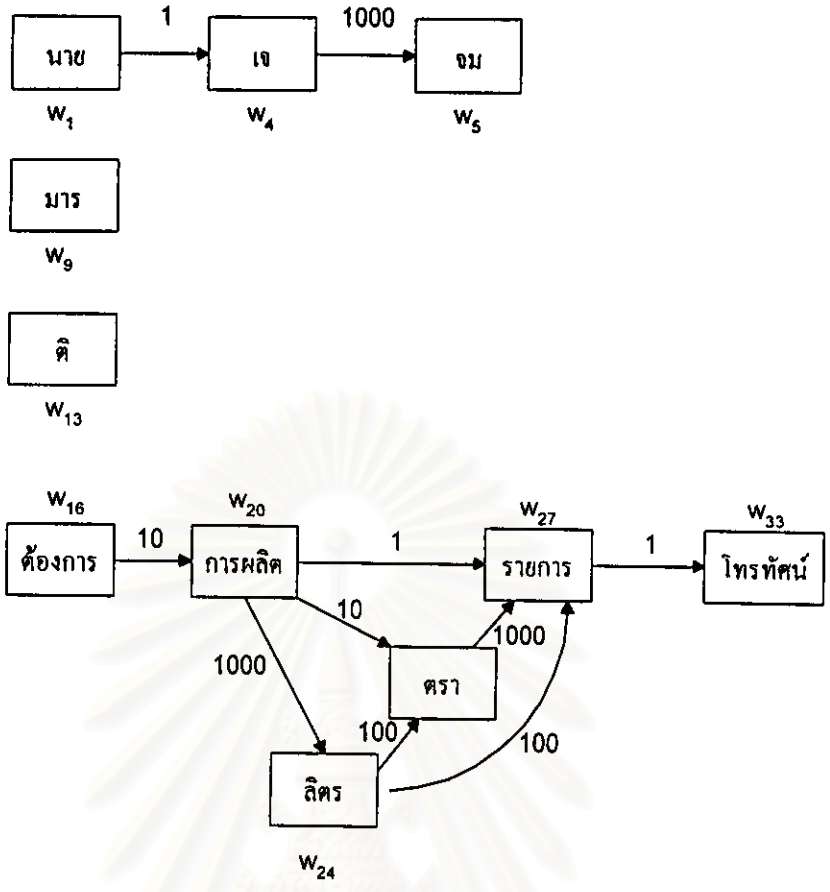
ตารางที่ 3.2 ค่าน้ำหนักของการทับกัน และการต่อกันพอดีของคำ

จากตารางที่ 3.2 สามารถนำมาเขียนได้เป็นรูปในตารางที่ 3.3 โดยวงกลม หมายถึงตัวอักษร หรือสระ หรือวรรณยุกต์ ในภาษาไทย สี่เหลี่ยมที่ล้อมรอบวงกลม หมายถึง คำในภาษาไทยที่มีอยู่ในพจนานุกรม ส่วนสี่เหลี่ยมที่มีการแรงเงา หมายถึงคำในพจนานุกรมที่ต้องเก็บเพิ่มเติมในขั้นตอนที่ 4

#	Conditions	Weights
1		1
2		10
3		100
		
		
4		1000

ตารางที่ 3.3 ค่าน้ำหนัก

จากตัวอย่างในขั้นตอนที่ 1 สามารถสร้างกราฟได้ดังนี้



รูปที่ 3.7 กราฟการต่อและทับกันของคำจากข้อความ
"นายเจมส์มาร์ตินต้องการผลิตรายการโทรทัศน์"

ขั้นตอนที่ 3

แต่ละส่วนของกราฟ หาเส้นทางที่มีค่าน้ำหนักน้อยที่สุดจากทางซ้ายสุด ไปทางขวาสุด ของแต่ละส่วนของกราฟ กำหนด $W = \{ w_i \mid w_i$ ที่มีเส้นทางที่สั้นที่สุด $\}$ จากตัวอย่างข้างต้นจะได้ $W = \{ w_1, w_4, w_5, w_9, w_{13}, w_{16}, w_{20}, w_{27}, w_{33} \}$

ขั้นตอนที่ 4

ในขั้นตอนนี้ กำหนด W′ คือเซตของคำที่แบ่งได้จากพจนานุกรม และ U′ คือเซตของคำที่แบ่งได้จากกฎ โดยสร้างจาก

4.1 กำหนด U เป็นเซตของกลุ่มตัวอักษรที่ไม่รู้จัก สามารถหาได้ดังนี้

4.1.1 สำหรับ (w_i, w_j) ที่มีค่าน้ำหนักเท่ากับ 1000 ในเส้นทางที่สั้นที่สุด (ผลจากขั้นตอนที่ 3) ซึ่งจะทำให้เกิดกลุ่มตัวอักษรที่ไม่รู้จักขึ้น 2 คำ เพิ่มคำ $T_{i,j-1}$ และ $T_{i'+1,j'}$ ไว้ใน U จากตัวอย่างข้างต้น มีคำว่า "เ" และ "ม" (ได้จาก (w4 , w5))

4.1.2 ทุกคู่ของโหนด w_i และ w_j ที่อยู่ต่างส่วนกันของ G, $i < j$ และไม่มี $w_k \in W$ โดยที่ $i < k < j$ เพิ่ม $T_{i'+1,j+1}$ ใน U จากตัวอย่างข้างต้นได้ " ส์", "ั", และ "น"

4.1.3 ถ้า $U \neq \varnothing$ นำ T ไปแบ่งพยางค์โดยใช้กฎ กำหนด X คือเซตของคำที่ได้จากการแบ่งพยางค์โดยใช้กฎ และ x_i คือพยางค์ที่มีตำแหน่งเริ่มต้นที่ตำแหน่ง i จากตัวอย่างสามารถแสดงได้ดังตารางที่ 3.4

i	T_i	x_i
1	นายเจมส์มาร์ตินต้องการผลิตรายการโทรทัศน์	นาย
4	เจมส์มาร์ตินต้องการผลิตรายการโทรทัศน์	เจมส์
9	มาร์ตินต้องการผลิตรายการโทรทัศน์	มาร์
13	ตินต้องการผลิตรายการโทรทัศน์	ติน
16	ต้องการผลิตรายการโทรทัศน์	ต้อง
20	การผลิตรายการโทรทัศน์	การ
23	ผลิตรายการโทรทัศน์	ผลิต
27	รายการโทรทัศน์	ราย
30	การโทรทัศน์	การ
33	โทรทัศน์	โทร
36	ทัศน์	ทัศน์

ตารางที่ 3.4 ตัวอย่างของ x_i ที่ได้จาก sistring

แต่ละ u_i โดย $u_i \in U$ ถ้า u_i เป็นส่วนหนึ่งส่วนใดของ x_j แล้ว $x_j \in U'$

จากตัวอย่างข้างต้น

$$U = \{ T_{4,4}, T_{6,6}, T_{7,8}, T_{12,12}, T_{15,15} \}$$
$$X = \{ x_1, x_4, x_9, x_{13}, x_{16}, x_{20}, x_{23}, x_{27}, x_{30}, x_{33}, x_{36} \}$$
$$U' = \{ x_4, x_9, x_{13} \}$$

4.2 W' สามารถหาได้ดังนี้ แต่ละ w_i โดย $w_i \in W$ ถ้า w_i ไม่เป็นส่วนหนึ่งส่วนใดของ x_j โดย $x_j \in U'$ แล้ว $w_i \in W'$ จากตัวอย่าง w_4 และ w_5 เป็นส่วนหนึ่งของ x_4 , w_9 เป็นส่วนหนึ่งของ x_9 และ w_{13} เป็นส่วนหนึ่งของ x_{13} ดังนั้น W' มีสมาชิกดังต่อไปนี้

$$W' = \{ w_1, w_{16}, w_{20}, w_{27}, w_{33} \}$$

และเพิ่มคำจากกรณีดังต่อไปนี้

4.2.1 พิจารณากรณีที่ 2 ในตารางที่ 3.2 เพิ่ม $T_{i,j-1}$ และ $T_{i'+1,j'}$ เข้าใน W′ จากตัวอย่างข้างต้น ได้คำว่า "ต้อง" และ "ผลิต" จาก w_{16} และ w_{20}

4.2.2 พิจารณากรณีที่ 3 ในตารางที่ 3.2 เพิ่ม $T_{i,k}$ และ $T_{k+1,j'}$ เข้าใน W′ โดยที่ k กำหนดในตารางที่ 3.3

จากอัลกอริทึมการแบ่งคำจะได้ W′ คือเซตของคำที่แบ่งได้จากพจนานุกรม และ U′ คือเซตของคำที่แบ่งได้จากกฎ ทั้ง W′ และ U′ จะใช้เป็นคำหลักในการนำไปทำดัชนีของระบบสืบค้นข้อมูลโดยใช้แฟ้มข้อมูลผกผัน จากตัวอย่างข้างต้นจะได้ W′ = { นาย, ต้องการ, การผลิต, รายการ, โทรทัศน์, ต้อง, ผลิต } และ U′ = { เจมส์, มาร์, ติน }

พิจารณาตัวอย่างต่อไปนี้

1. ตากลมอบอกไก่

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

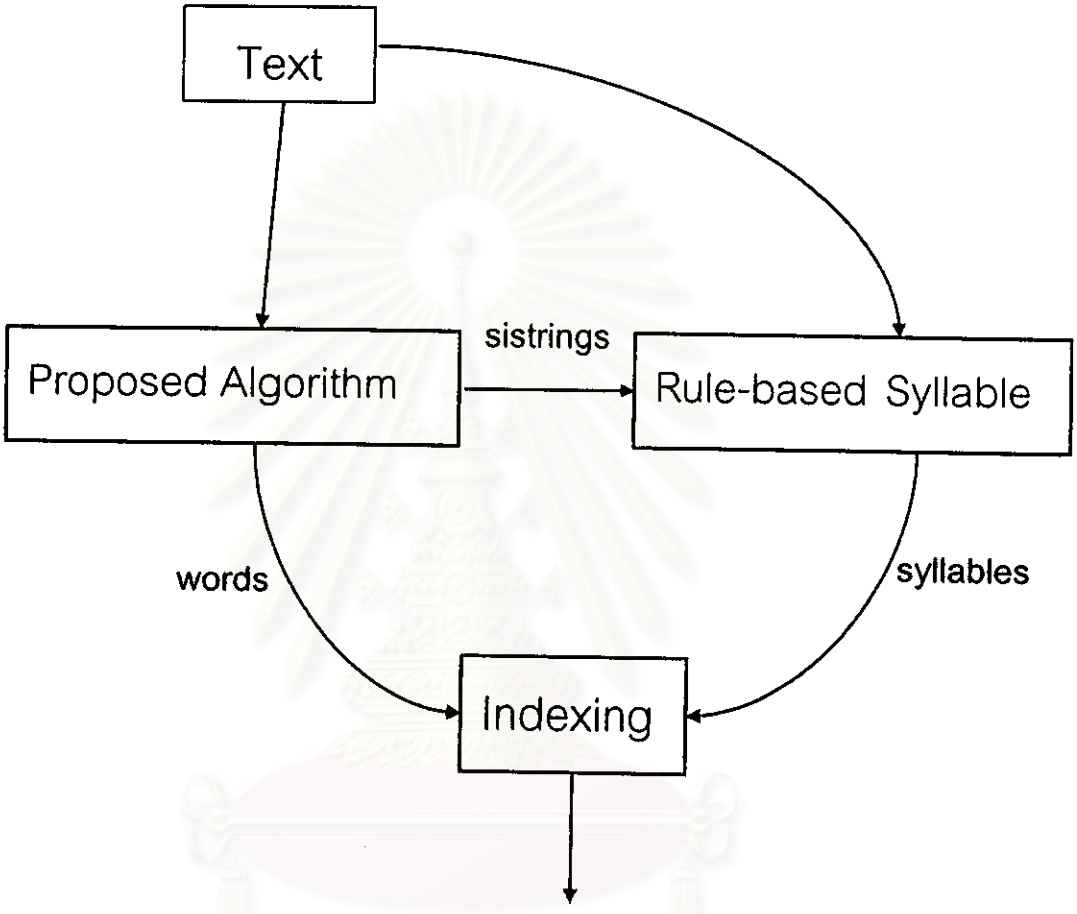
2. เขาได้ตำแหน่งท็อปชั้น

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 }

การทำงานของอัลกอริทึมนี้ สามารถแสดงได้ดังรูปที่ 3.8



รูปที่ 3.8 แสดงการนำเอาอัลกอริทึมที่ได้ไปประยุกต์ใช้งาน

จากรูปที่ 3.8 ข้อความภาษาไทยจะถูกส่งไปยังอัลกอริทึมที่คิดค้นขึ้นมา ผลที่ได้จากอัลกอริทึมคือ คำที่อยู่ในพจนานุกรม และกลุ่มของตัวอักษรที่ไม่มีในพจนานุกรม กลุ่มตัวอักษรที่ไม่มีในพจนานุกรมจะถูกส่งไปตัดคำโดยใช้กฎ ผลที่ได้คือพยางค์ นำคำที่มีอยู่ในพจนานุกรมที่ได้ครั้งแรก รวมกับพยางค์ที่ได้จากการตัดคำโดยใช้กฎ นำมาจัดทำดัชนีเพื่อใช้ในการค้นหาต่อไป

ผลจากการคิดค้นอัลกอริทึมการแบ่งคำขึ้นมา ได้นำไปเขียนโปรแกรมเพื่อแบ่งคำ โดยนำเอกสารภาษาไทยเข้าไปยังโปรแกรม จะได้เอกสารใหม่คืนออกมา โดยที่เอกสารที่คืนออกมานั้นจะมีข้อมูลของเอกสารที่ป้อนเข้า แต่จะมีการแบ่งช่องว่างระหว่างคำ เมื่อทำกับเอกสารภาษาไทยเช่นนี้ทุกๆ เอกสารที่ต้องการ นำโปรแกรม Search Engine ที่ถูกคิดค้นขึ้นก่อนหน้านี้ แต่ใช้ได้เฉพาะภาษาอังกฤษ นำมาทดสอบกับเอกสารที่ออกมาจากโปรแกรมแบ่งคำ ปรากฏว่าสามารถหาคำภาษาไทยได้อย่างถูกต้อง

แต่ผลการทดสอบข้างต้น ก็ไม่สามารถใช้ได้กับ Search Engine บางตัว ที่มีการตัดตัวอักษรที่ไม่ใช้ตัวอักษรภาษาอังกฤษออกไป ดังนั้น ถ้า Search Engine ภาษาอังกฤษ ซึ่งจะมีผู้ทำการวิจัย และพัฒนามากที่สุดนั้น ได้มีการพัฒนาขึ้นไปเรื่อยๆ ก็สามารถที่จะนำมาใช้กับภาษาไทยได้ โดยนำเอาเอกสารภาษาไทยนั้น ผ่านเข้าไปในโปรแกรมแบ่งคำ จากนั้นจึงนำเอกสารที่ได้ออกมา นำเข้าไปยัง Search Engine ภาษาอังกฤษ การสืบค้นข้อมูลภาษาไทยก็สามารถทำได้ โดยการนำข้อความที่ต้องการค้นหา ผ่านเข้าไปยังโปรแกรมแบ่งคำก่อน แล้วนำคำที่แบ่งได้ไปค้นหาใน Search Engine ภาษาอังกฤษต่อไป

อัลกอริทึมการจัดทำดัชนีภาษาไทยโดยใช้ sistring กับคำที่ไม่มีในพจนานุกรม

จากอัลกอริทึมข้างต้น จะใช้อัลกอริทึมการแบ่งพยางค์ด้วยกฎ โดยใช้อัลกอริทึมการแบ่งพยางค์ที่ใช้ในโปรแกรมซียูไรต์เตอร์ นำมาจัดการกลุ่มตัวอักษรที่ไม่รู้จัก ซึ่งจะมีปัญหาการแบ่งพยางค์ด้วยกฎนั้นอาจไม่ถูกต้องเสมอไป ปัญหาเหล่านี้แก้ได้ถ้าใช้ sistring เข้ามาจัดการกับกลุ่มตัวอักษรที่ไม่รู้จัก [16] โดยมีขั้นตอนของอัลกอริทึมดังต่อไปนี้

ขั้นตอนที่ 1

เหมือนอัลกอริทึมข้างต้น

ขั้นตอนที่ 2

เหมือนอัลกอริทึมข้างต้น

ขั้นตอนที่ 3

เหมือนอัลกอริทึมข้างต้น

ขั้นตอนที่ 4

ในขั้นตอนนี้ กำหนด W' คือเซตของคำที่แบ่งได้จากพจนานุกรม และ Y' เซตของ sistring ของกลุ่มตัวอักษรที่ไม่รู้จัก โดยสร้างจาก

4.1 กำหนด Y เป็นเซตของกลุ่มตัวอักษรที่ไม่รู้จัก สามารถหาได้ดังนี้

4.1.1 สำหรับ (w_i, w_j) ที่มีค่าน้ำหนักเท่ากับ 1000 ในเส้นทางที่สั้นที่สุด (ผลจากขั้นตอนที่ 3) ซึ่งจะทำให้เกิดกลุ่มตัวอักษรที่ไม่รู้จักขึ้น 2 คำ เพิ่มคำ $T_{i,j-1}$ และ $T_{i'+1,j'}$ ไว้ใน Y จากตัวอย่างข้างต้น มีคำว่า "เ" และ "ม" (ได้จาก $(w4, w5)$)

4.1.2 ทุกคู่ของโหนด w_i และ w_j ที่อยู่ต่างส่วนกันของ G, $i < j$ และไม่มี $w_k \in W$ โดยที่ $i < k < j$ เพิ่ม $T_{i'+1,j+1}$ ใน Y จากตัวอย่างข้างต้นได้ "ส์", " ", และ "น"

4.1.3 นำผลที่ได้จาก 4.1.1 และ 4.1.2 จากตัวอย่างข้างต้นคือ "เ", "ม", "ส์", " ", และ "น" พิจารณาจากข้อความเดิม และคำที่ได้จากขั้นตอนที่ 3 ค้นหาคำที่อยู่ติดกับผลที่ได้จาก 4.1.1 และ 4.1.2 ที่อยู่ติดทางด้านซ้ายจากข้อความเดิม และคำคำนั้นอยู่ใน W นำคำที่ได้รวมกับคำในผลที่ได้จาก 4.1.1 และ 4.1.2 จากตัวอย่างข้างต้นผลที่ได้คือ "นายเ", "เจม", "จมส์", "มาร์" และ "ติน"

4.1.4 กลุ่มตัวอักษรที่ได้จาก 4.1.3 ตัวไหนที่อยู่ติดกัน หรือมีส่วนที่ทับกัน สามารถรวมกันได้ นำมารวมกัน จากตัวอย่างข้างต้นผลที่ได้คือ $Y = \{$"นายเจมส์มาร์ติน"$\}$

4.2 W' สามารถหาได้จาก W ลบด้วยสมาชิกที่เป็นซับเซตของ Y'

$W' = \{ w_1, w_{16}, w_{20}, w_{27}, w_{33} \}$

และเพิ่มคำจากกรณีดังต่อไปนี้

4.2.1 พิจารณากรณีที่ 2 ในตารางที่ 3.1 เพิ่ม $T_{i,j-1}$ และ $T_{i'+1,j'}$ เข้าใน W' จากตัวอย่างข้างต้น ได้คำว่า "ต้อง" และ "ผลิต" จาก w_{16} และ w_{20}

4.2.2 พิจารณากรณีที่ 3 ในตารางที่ 3.2 เพิ่ม $T_{i,k}$ และ $T_{k+1,j'}$ เข้าใน W' โดยที่ k กำหนดในตารางที่ 3.3

จากอัลกอริทึมการแบ่งคำจะได้ W' คือเซตของคำที่มีอยู่พจนานุกรม และ Y' คือ เซตของ sistring ของกลุ่มคำที่ไม่อยู่ในพจนานุกรม W' จะใช้เป็นคำหลักในการนำไปทำดัชนีของระบบสืบค้นข้อมูลโดยใช้แฟ้มข้อมูลผกผัน ส่วน Y' ใช้โครงสร้างข้อมูลแบบ trie (ภาคผนวก ค.) ในการจัดเก็บ จากตัวอย่างข้างต้นจะได้ $W' = \{$ นาย, ต้องการ, การผลิต, รายการ, โทรทัศน์, ต้อง, ผลิต $\}$ และ Y' = เซตของ sistring ต่างๆ ของคำว่า "นายเจมส์มาร์ติน"

พิจารณาตัวอย่างต่อไปนี้

1. ตากลมอบอกไก่

$W = \{$ ตาก, กลม, มอบ, บอก, ไก่ $\}$

$Y = \{ \}$

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

$Y' = \{ \}$

2. เขาได้ตำแหน่งท็อปชั้น

$W = \{$ เขา, ได้, ตำแหน่ง, ชั้น $\}$

$Y = \{$ ตำแหน่งท็อป $\}$

$W' = \{$ เขา, ได้, ชั้น $\}$

Y' = เซตของ sistring ต่างๆ ของข้อความ "ตำแหน่งท็อป"

3. เขาได้อันดับท็อปชั้น

W = { เขา, ได้, อันดับ, บท, ชั้น }

Y = { บท็อป }

W′ = { เขา, ได้, อันดับ, ชั้น }

Y′ = เซตของ sistring ต่างๆ ของข้อความ "บท็อป"