บทที่ 2

การเรียงลำดับข้อมูล

วิธีการเรียงลำดับข้อมูลมีหลายวิธี ดังกล่าวแล้วในบทที่ 1 แต่หลังจากที่ได้ศึกษาแล้ว สรุปได้ว่า ควิกซอร์ตเป็นวิธีการเรียงลำดับ ที่มีประสิทธิภาพและรวดเร็วที่สุดวิธีหนึ่ง เนื่องจากเหมาะกับข้อมูลจำนวนมาก เพราะใช้หลักการแบ่งข้อมูลออกเป็นกลุ่มย่อย ๆ แล้วจึงเรียงกลุ่มข้อมูลย่อยนั้น และในการเรียงลำดับใช้หลักการเรียกซ้ำ (Recursive) ซึ่งภาษาซี สามารถทำได้อย่างมีประสิทธิภาพ ดังนั้น จึงได้เลือกควิกซอร์ตเป็นขั้นตอนวิธีที่ใช้ในวิทยานิพนธ์นี้

2.1 ขั้นตอนวิธีการเรียงลำดับข้อมูล

ควิกซอร์ต[1] เป็นการเรียงลำดับแบบเปรียบเทียบ ที่ทำการเปรียบเทียบในแต่ละครั้งด้วยคีย์คู่หนึ่ง ทำเช่นนี้ไปเรื่อย ๆ ขั้นตอนขั้นแรก ก็คือนำระเบียนข้อมูลมา 1 ระเบียน ในที่นี้สมมติว่าเป็น R_1 และเลื่อนตำแหน่งของระเบียนนี้ไปเรื่อย ๆ จนถึงตำแหน่งที่ควรจะอยู่จริงในแฟ้มข้อมูลที่เรียงลำดับแล้ว เรียกตำแหน่งนี้ว่า s ในขณะที่ทำการเรียงลำดับจนถึงตำแหน่งในขั้นสุดท้ายนั้น ระเบียนนี้จะต้องเปลี่ยนตำแหน่งกับระเบียนอื่นเสมอ โดยจัดให้ระเบียนทางซ้ายของ s มีค่าน้อยกว่าหรือเท่ากับ s และระเบียนทางขวามีค่ามากกว่าหรือเท่ากับ s ด้วยเหตุนี้ แฟ้มข้อมูลจึงถูกแบ่งออก เป็นแฟ้มข้อมูลย่อย 2 กลุ่ม คือ แฟ้มข้อมูลย่อยของระเบียนที่ $R_1 \ldots R_{s-1}$ และแฟ้มข้อมูลย่อยของระเบียนที่ $R_{s+1} \ldots R_N$ จากนั้น จึงใช้เทคนิคเดียวกันนี้กับแฟ้มข้อมูลย่อยแต่ละกลุ่ม จนกระทั่งข้อมูลทั้งหมดถูกเรียงลำดับเรียบร้อย

การแบ่งแฟ้มข้อมูลย่อยให้เป็นกลุ่มทางซ้ายและทางขวา ทำโดยเก็บตัวชี้ (Pointer) 2 ตัว คือ i และ j โดยที่ i = 2 และ j = N เป็นค่าเริ่มต้น ถ้าสมมติให้ R_i เป็นแฟ้มข้อมูลย่อยทางซ้าย (เปรียบเทียบ K_i กับ K_1) เพิ่มค่า i ทีละ 1 ไปเรื่อย ๆ จนกระทั่งพบระเบียน

---

[1] D.E. Knuth, The Art of Computer Vol. 3, Sorting and Searching (Addison-Wesley Publishing Company, 1973), pp. 114-117.

R_i ซึ่งเป็นของกลุ่มทางขวา ในขณะเดียวกันลดค่า j ทีละ 1 จนกระทั่งพบระเบียน R_j ซึ่งเป็นของกลุ่มทางซ้าย ถ้า $i < j$ ให้สลับที่ระหว่าง R_iกับ R_j แล้วทำระเบียนถัดไปด้วยวิธีเดียวกันนี้ไป จนกระทั่ง $i \geq j$ ให้สลับ R_j กับ R_1 ดังตัวอย่างในรูปที่ 2.1 ที่แฟ้มข้อมูลประกอบด้วยตัวเลข 16 ตัว คือ

```
          i                                                       j
          |   แฟ้มข้อมูลเริ่มต้น                                  |
          ↓                                                       ↓
[503 087 512 061 908 170 897 275 653 426 154 509 612 677 765 703]
          เปลี่ยนตำแหน่งครั้งที่ 1

 503 087 512 061 908 170 897 275 653 426 154 509 612 677 765 703
          เปลี่ยนตำแหน่งครั้งที่ 2

 503 087 154 061 908 170 897 275 653 426 512 509 612 677 765 703
          เปลี่ยนตำแหน่งครั้งที่ 3

 503 087 154 061 426 170 897 275 653 908 512 509 612 677 765 703
          ตัวชี้

 503 087 154 061 426 170 275 897 653 908 512 509 612 677 765 703
          แฟ้มข้อมูลย่อย

[275 087 154 061 426 170]503[897 653 908 512 509 612 677 765 703]
                          ↑   ↑
                          j   i
```

หมายเหตุ : แสดงตำแหน่งของ i, j, k_i และ k_j ด้วยการขีดเส้นใต้

รูปที่ 2.1 ตัวอย่างข้อมูลควิกซอร์ต ตัวอย่างที่ 1

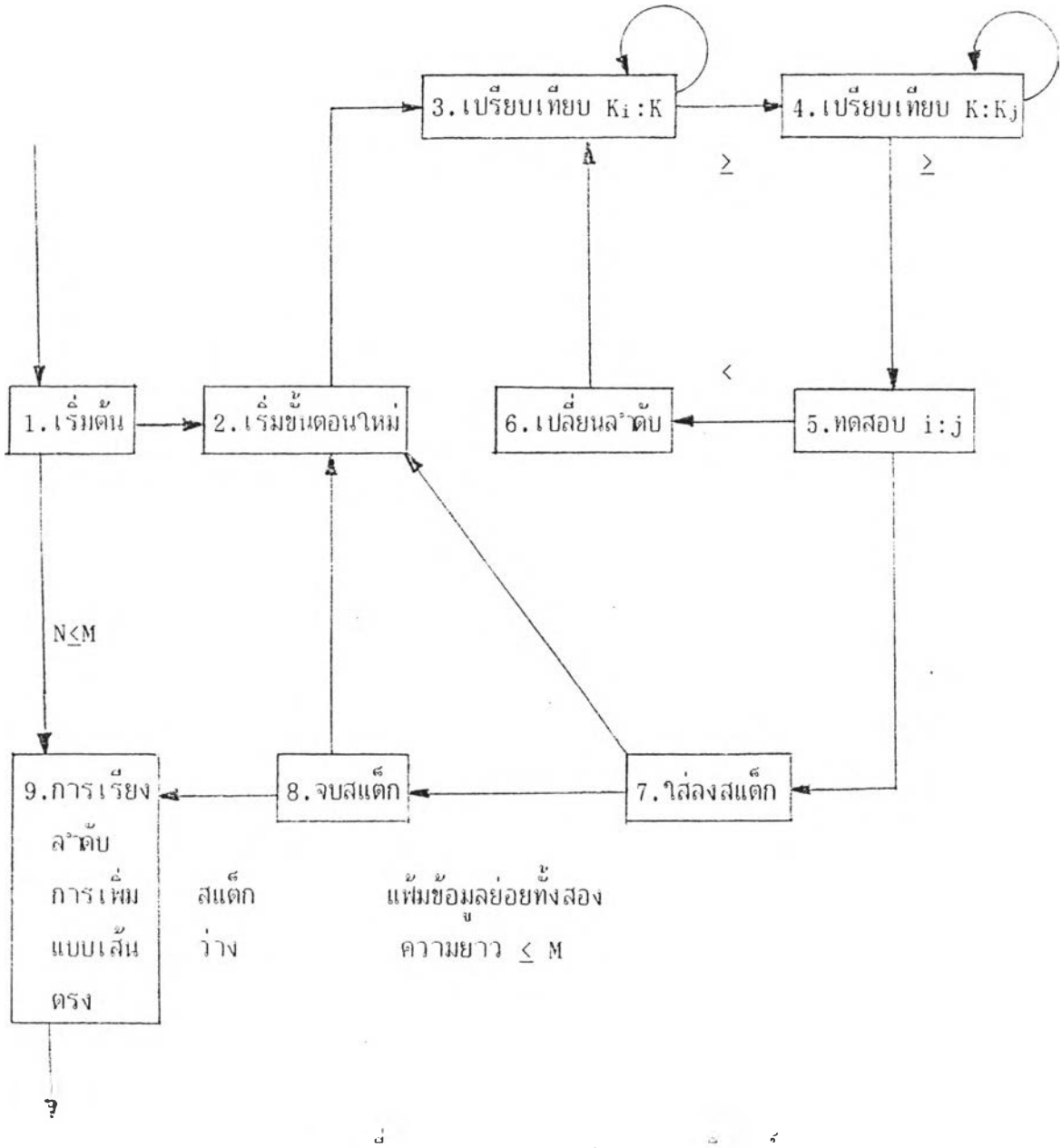
ในรูปที่ 2.2 แสดงการเรียงลำดับข้อมูลทั้งหมด เสร็จสิ้นลงใน 11 ขั้นตอน ค่าในวงเล็บ หมายถึงแฟ้มข้อมูลย่อยที่ยังต้องถูกเรียงลำดับอีก แฟ้มข้อมูลย่อยเหล่านี้จะแทนด้วยตัวแปร 2 ตัว คือ l และ r และใส่ตัวแปรทั้งคู่ลงเก็บในสแต็ก คือ (l_k, r_k) ในแต่ละครั้งที่แฟ้มข้อมูลถูกแบ่ง เสมอจะใส่ขอบเขตของแฟ้มข้อมูลย่อยที่มีขนาดใหญ่กว่า และทำเช่นนี้ จนกระทั่งถึงแฟ้มข้อมูลที่เล็กที่สุด

การเรียงลำดับวิธีนี้ บางทีเรียกว่า การเรียงลำดับแบบเปลี่ยนส่วน (Partition-Exchange Sorting) ซึ่ง ซี.เอ.อาร์.โฮร์ (C.A.R. Hoare) ได้ตั้งชื่อวิธีนี้ว่า "ควิกซอร์ต" เพราะการทำงานอาศัยหลักการวนซ้ำ (Loop) ซึ่งคอมพิวเตอร์สามารถทำได้รวดเร็วมาก การเปรียบเทียบทั้งหมด ที่เกิดขึ้นในแต่ละขั้นตอน จะกระทำคล้ายคลึงกัน ดังนั้น ค่าที่จะเก็บไว้ในเรจิสเตอร์และมีเพียงค่านี้เท่านั้นที่ถูกเปลี่ยนระหว่างการเปรียบเทียบ จากตัวอย่างตามรูปที่ 2.2 จำนวนการเคลื่อนย้ายข้อมูล ใช้เพียง 17 ครั้งเท่านั้น

```
                                                          (l,r)  สแต็ก
[503 087 512 061 908 170 897 275 653 426 154 509 612 677 765 703]
                                                          (1,16)  -
[275 087 154 061 426 170]503[897 653 908 512 509 612 677 765 703]
                                                          (1,6)_(8,16)
[170 087 154 061]275 426 503[897 653 908 512 509 612 677 765 703]
                                                          (1,1) (8,16)
[061 087 154]170 275 426 503[897 653 908 512 509 612 677 765 703]
                                                          (1,3) (8,16)
 061[087 154]170 275 426 503[897 653 908 512 509 612 677 765 703]
                                                          (2,3) (8,16)
 061 087 154 170 275 426 503[897 653 908 512 509 612 677 765 703]
                                                          (8,16)  -
 061 087 154 170 275 426 503[765 653 703 512 509 612 677]897 908
                                                          (8,14)  -
 061 087 154 170 275 426 503[677 653 703 512 509 612]765 897 908
                                                          (8,13)  -
 061 087 154 170 275 426 503[509 653 612 512]677 703 765 897 908
                                                          (8,11)  -
 061 087 154 170 275 426 503 509[653 612 512]677 703 765 897 908
                                                          (9,11)  -
 061 087 154 170 275 426 503 509[512 612]653 677 703 765 897 908
                                                          (9,10)  -
 061 087 154 170 275 426 503 509 512 612 653 677 703 765 897 908
                                                            -     -
```

รูปที่ 2.2 ตัวอย่างข้อมูลควิกซอร์ต ตัวอย่างที่ 2

วิธีการเก็บค่าพิเศษ (Extra Keeping) ที่ใช้ควบคุม i, j และสแต็กจะไม่แตกต่างกันนัก แต่จะทำให้ควิกซอร์ต เหมาะสำหรับข้อมูลจำนวนมาก ๆ ดังรูปที่ 2.3



รูปที่ 2.3 แสดงการทำงานของควิกซอร์ต

ขั้นตอนวิธีควิกซอร์ต

ระเบียน $R_1,\ldots,R_N$ จะถูกเรียงลำดับตำแหน่งใหม่ โดยจะเรียงตามลำดับ ของ $K_1 \le \ldots \le K_N$ มีสแต็กขนาดล็อก N (log N) ใช้เป็นที่เก็บสำรอง ต่อไปนี้จะกล่าวถึงขั้นตอนวิธีควิกซอร์ต โดยมีการเปลี่ยนแปลงเล็กน้อย เพื่อประสิทธิภาพที่ดีขึ้น

1. สมมุติให้ค่าคีย์ $K_0 = -\infty$ และ $K_{n+1} = +\infty$ ดังนั้น

$$K_0 \le K_i \le K_{n+1} \text{ โดยที่ } 1 \le i \le N$$

2. แฟ้มข้อมูลย่อยขนาด M ระเบียน หรือของกลุ่มที่ยังไม่ได้เรียงลำดับทางซ้ายจะถูกเรียงลำดับด้วยวิธีการเพิ่มแบบเส้นตรง (Straight Insertion) ในที่นี้เลือก $M \ge 1$ เป็นพารามิเตอร์ (ถ้าจำเป็นต้องลดเวลาการทำงานของเครื่อง อาจประยุกต์ใช้การเรียงลำดับแบบเส้นตรง กับแฟ้มข้อมูลย่อยแต่ละกลุ่ม)

3. ระเบียนซึ่งมีคีย์เท่ากันจะถูกเปลี่ยนตำแหน่ง แม้ว่าจะไม่จำเป็น (เพื่อให้การวนซ้ำภายในทำได้รวดเร็ว และช่วยในการแบ่งแฟ้มข้อมูลย่อยให้ใกล้เคียงกับครึ่งหนึ่ง)

สรุป ขั้นตอนการทำงาน ซึ่งกำหนดตำแหน่งคีย์ โดยค่าของ r และ l

1) เริ่มต้น

ถ้า $N \le M$ ไปทำข้อที่ 9 มิเช่นนั้น ทำสแต็กให้ว่าง และกำหนดให้

l <-- 1, r <-- N

2) เริ่มขั้นตอนใหม่

เมื่อต้องการเรียงลำดับแฟ้มข้อมูลย่อย $R_l \ldots R_r$ โดยมี $r \ge l+M$ และ $K_{l-1} \le K_i \le K_{r+1}$ เมื่อ $l \le i \le r$ กำหนดให้ i <-- l, j <-- r+1 และ K <-- K_l, R <-- R_l

3) เปรียบเทียบ $K_i : K$

เป็นขั้นตอนที่แฟ้มข้อมูลจะถูกเรียงลำดับใหม่ ดังนี้

$K_k \le K$ โดย $l-1 \le k \le i$, $K \le K_k$, $j \le k \le r+1$ และ $l \le i < j$) เพิ่มค่า i ทีละ 1 ดังนั้น

ถ้า $K_i < K$ ทำซ้ำขั้นตอนนี้อีก

(เมื่อ $K_j \ge K$ ต้องจบด้วย $i \le j$)

4) เปรียบเทียบ K : K_j

ลดค่า j ทีละ 1 ดังนั้น ถ้า $K < K_j$ ทำซ้ำขั้นตอนนี้อีก

(เมื่อ $K \geq K_{i-1}$ ต้องจบด้วย $j \geq i-1$

5) ทดสอบ i : j

(จากข้อที่ 3) ยกเว้นให้ $k = i$ และ $k = j$ และ $K_i \geq K \geq K_j$ และ $r \geq j \geq i-1 \geq 1$ ด้วย)

ถ้า $j \leq i$ สับเปลี่ยน $R_1 <--> R_j$ และไปทำข้อที่ 7)

6) เปลี่ยนลำดับ

สับเปลี่ยน $R_i <--> R_j$ และกลับไปทำข้อที่ 3) อีก

7) ใส่ลงสแต็ก

(ขณะนี้ แฟ้มข้อมูลย่อย $R_1 \ldots R_i \ldots R_r$ มีการแบ่งย่อย ดังนั้น $K_k \leq K_i$ โดย $1-1 \leq k \leq i$ และ $K_j \leq K_k$ โดย $j \leq k \leq r+1$)

ถ้า $r-j \geq j-1 > M$ ให้เพิ่ม (j+1,r) บนสุดของสแต็ก

กำหนดให้ r <-- j-1 และกลับไปข้อที่ 2)

ถ้า $j-1 > r-j > M$ ให้เพิ่ม (1,j-1) บนสุดของสแต็ก

กำหนดให้ 1 <-- j+1 และกลับไปข้อที่ 2)

(แต่ละจุดรับเข้า(a,b) บนสแต็ก จะถูกใช้ในการเรียงลำดับแฟ้มข้อมูลย่อย $R_a \ldots R_b$ ในบางขณะในตอนต่อไป)

มิเช่นนั้น ถ้า $r-j > M \geq j-1$, กำหนดให้ 1 <-- j+1 และกลับไปข้อที่ 2)

หรือ ถ้า $j-1 > M \geq r-j$, กำหนดให้ r <-- j-1 และกลับไปข้อที่ 2)

8) จบสแต็ก

ถ้าสแต็กไม่ว่างนำจุดรับเข้า (l',r') ออกจากสแต็ก กำหนดให้ l <-- l', r <-- r' และกลับไปข้อที่ 2)

9) การเรียงลำดับการแทรกแบบเส้นตรง

ให้ j = 2 , 3,...,N ถ้า $K_{j-1} > K_j$ กระทำต่อไปนี้

กำหนดให้ K <-- K_j, R <-- R_j, i <-- j-1 ดังนั้น

กำหนดให้ R_{i+1} <-- R_i และ i <-- i-1 ทำซ้ำหนึ่งครั้งหรือมากกว่าจนกระทั่ง $K_i \leq K$

จากนั้น ใส่ข้อมูลระเบียน R ลงในระเบียน R_{i+1} แทน

(ข้อที่ 9) อาจจะไม่มี ถ้า M = 1)

2.2 การเรียงลำดับข้อมูลภาษาไทย

การเก็บรหัสข้อมูลภาษาไทย ที่ใช้กับคอมพิวเตอร์มีหลายรหัส วิทยานิพนธ์นี้ข้อมูลสามารถเป็นได้ทั้งรหัสของสำนักงานมาตรฐานผลิตภัณฑ์อุตสาหกรรม (สมอ.) และรหัสของเกษตร

ส่วนการเรียงลำดับข้อมูลที่เป็นภาษาไทยนั้น ได้ยึดหลักให้ใกล้เคียงกับการเรียงลำดับคำของพจนานุกรม ฉบับราชบัณฑิตยสถาน พ.ศ.2525 (พิมพ์ครั้งที่4 พ.ศ.2531) มากที่สุด

จะขอกล่าวถึงคำชี้แจงหลักการเขียนและวิธีใช้พจนานุกรม ที่เขียนไว้ในพจนานุกรม ฉบับราชบัณฑิตยสถาน พ.ศ. 2525[2] พอเป็นสังเขป ดังนี้

คำชี้แจงหลักการเขียนและวิธีใช้พจนานุกรม (เฉพาะที่เกี่ยวข้องกับวิทยานิพนธ์)

ก. การเรียงลำดับคำ

1. ตัวพยัญชนะนั้นลำดับไว้ตามตัวอักขระ คือ ก ข ฃ ค ฅ ฆ ง จ ฉ ช ซ ฌ ญ ฎ ฏ ฐ ฑ ฒ ณ ด ต ถ ท ธ น บ ป ผ ฝ พ ฟ ภ ม ย ร ล ว ศ ษ ส ห ฬ อ ฮ ไม่ได้ลำดับตามเสียง เช่น จะค้นคำ ทราบ จะต้องไปหาในหมวดตัว ท; จะค้นคำ เหมา จะต้องไปหาในหมวดตัว ห. ฤ ฤๅ ลำดับไว้หลังตัว ร, ฦ ฦๅ ลำดับไว้หลังตัว ล

2. สระนั้นก็ไม่ได้ลำดับไว้ตามเสียง ลำดับไว้ตามรูปสระดังนี้ ะ ั า ิ ี ึ ื ุ ู ฺ เ แ โ ใ ไ รูปสระที่ประสมกันหลายรูปก็จัดเรียงตามลำดับ รูปสระที่อยู่ก่อนและหลังตามลำดับข้างบนนี้เหมือนกัน ดังเช่นลำดับไว้ให้ดูต่อไปนี้

ะ	เ็	(เสือ)
ั (กัน)	เ็ะ	(เกือะ)
ั ะ (ผ้าะ)	แ	
า	แ ะ (เกะ)	แ ะ (แพะ)
ำ	เ า (เขา)	โ
ิ	เ าะ (เจาะ)	โ ะ
ิ	เ ็ (เด็น)	ใ
ิ	เ ็ (เสีย)	ไ
ิ	เ ็ ะ (เบียะ)	

ตัว อ า ย นั้นลำดับอยู่ในพยัญชนะเสมอ

---

[2]" พจนานุกรม ฉบับราชบัณฑิตยสถาน พ.ศ. 2525, " ราชบัณฑิตยสถาน (พ.ศ. 2531); (1).

3. พยัญชนะกับสระประสมกัน ลำดับตามตัวพยัญชนะก่อนแล้วจึงลำดับรูปสระ ส่วนพยัญชนะที่ไม่มีสระปรากฏ เป็นรูปประสมอยู่ด้วย เอาไว้ข้างหน้า เช่น ขลา อยู่หน้า ขะข่า, กน อยู่หน้า กา

4. ◌่ ไม้เอก, ◌้ ไม้โท, ◌๊ ไม้ตรี, ◌๋ ไม้จัตวา, ◌็ ไม้ไต่คู้,และ◌์ ไม้ทัณฑฆาต เหล่านี้ไม่ได้จัดเข้าในลำดับ

พยัญชนะไทยที่มีทั้งหมด 44 ตัว ดังนี้

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

สระ มีทั้งหมด 21 รูป ดังนี้

ะ า ◌ิ ◌ี ◌ึ ◌ื เ แ โ ใ ไ ◌ั ◌ ◌ ◌ ◌ ◌ ◌ ฤ
ฤๅ ฤ ฦ ฦๅ อ ย ว

2.2.1 ขั้นตอนก่อนการเรียงลำดับ

2.2.1.1 การสลับตำแหน่ง

ลักษณะการวางตำแหน่งของสระหรือพยัญชนะภาษาไทยนั้น มีหลายลักษณะ แต่เมื่อนำมาเรียงลำดับต้องมีการสลับตำแหน่ง เพื่อให้อยู่ในกลุ่มพยัญชนะตัวเดียวกัน ในกรณีที่เป็นคำที่มีสระนำหน้าพยัญชนะ ให้สลับตำแหน่งพยัญชนะมาอยู่หน้าสระแทน ตัวอย่างเช่น

เก จะสลับเป็น กเ
แกง จะสลับเป็น กแง
ไก่ จะสลับเป็น กไ◌่

ส่วนกรณี ที่เป็นวรรณยุกต์และไม้ไต่คู้ จะสลับตำแหน่งให้ไปอยู่ท้ายของคำนั้น ตัวอย่างเช่น

ค้าง จะสลับเป็น คาง◌้
จ๊ะ จะสลับเป็น จะ◌๊
บล็อก จะสลับเป็น บลอก◌็

2.2.1.2 การเพิ่มหรือลดน้ำหนักของอักขระ

เนื่องจากรหัสแอสกีในส่วนอักขระภาษาไทย ยังตามมาตรฐาน

สถาบันวิทยบริการ

สมอ. และเกษตร จัดเรียงให้ตัวอักขระมาก่อน แล้วจึงตามด้วยสระและวรรณยุกต์ ดังนั้น ถ้านำเอาค่าของรหัสแอสกีภาษาไทย มาใช้เป็นน้ำหนักในการเรียงลำดับโดยตรง ผลที่ได้จะไม่ถูกต้อง ตัวอย่างเช่น เรียงคำ 4 คำ คือ กา, ก่า, กาง, ก้าง ด้วยโปรแกรม เปรียบเทียบกับการจัดเรียงของพจนานุกรม

เขียนแบบบรรทัดเดียว	กา	ก่า	กาง	ก้าง
จัดเรียงตามรหัสแอสกีโดยตรง	กา	กาง	ก้าง	ก่า
ผลการเรียงจากโปรแกรม	กา	กาง	ก้าง	ก่า
ผลการเรียงจากพจนานุกรม	กา	ก่า	กาง	ก้าง

การแก้ปัญหาสามารถทำได้ โดยกำหนดน้ำหนักของตัวอักขระแต่ละตัวขึ้นใหม่ ซึ่งอาจมีค่าเพิ่มขึ้นหรือลดลงจากค่ารหัสแอสกีของตัวอักขระนั้น ในวิทยานิพนธ์นี้ให้น้ำหนักของอักขระ โดยพยัญชนะยังคงใช้น้ำหนักตามรหัสแอสกีตามเดิม ส่วนวรรณยุกต์, ไม้ยมก และไม้ไต่คู้ให้น้ำหนักที่น้อยกว่าพยัญชนะ สำหรับสระให้น้ำหนักมากกว่าพยัญชนะ รวมทั้งสระที่เกิดจากการยุบรูปด้วย ซึ่งการให้น้ำหนักของตัวอักขระ ทั้งรหัสของสมอ. และเกษตร แสดงไว้ในภาคผนวก ข

2.2.1.3 การยุบรูป

จากคำชี้แจงการเรียงลำดับคำตามหลักพจนานุกรม ซึ่งได้กำหนดลำดับการเรียงของสระเอาไว้ การสลับตำแหน่ง และการเพิ่มหรือลดน้ำหนักของอักขระจะทำงานได้ถูกต้องกับสระเดี่ยวเท่านั้น กรณีที่เป็นสระประสม คือมีสระมากกว่า 1 ตัว รวมอยู่ในคำ ๆ นั้น การเรียงลำดับจะเรียงตามเพียงสระตัวใดตัวหนึ่งเท่านั้น จะต้องใช้การยุบรูปเข้าช่วย โดยยุบสระประสมทั้งหมดเป็นอักขระพิเศษ ที่มีน้ำหนักที่เหมาะสม จากนั้นจึงนำไปเรียงลำดับ ตัวอย่างเช่น

เก	จะสลับเป็น	กเ
เกะ	ยุบรูปเป็น	กX (X หมายถึง เ_ะ)
เกา	ยุบรูปเป็น	กY (Y หมายถึง เ_า)
เกาะ	ยุบรูปเป็น	กZ (Z หมายถึง เ_าะ)

2.2.1.4 การตัดคำ

เนื่องจากลักษณะภาษาไทย ไม่มีการใช้เครื่องหมาย หรือช่องว่างเป็นตัวกำหนดจุดสิ้นสุดของคำ ดังนั้น จึงเป็นหน้าที่ของโปรแกรมจัดเรียงลำดับ ที่จะต้องหาจุดสิ้นสุดของคำให้ได้ เพื่อผลเพื่อ

016329

1) ย้ายวรรณยุกต์และไม้ไต่คู้ของคำ ไปอยู่ท้ายคำนั้น

2) เพื่อตรวจสอบว่าคำนั้นมีสระประสมอยู่หรือไม่ และมีสระอะไรประสมอยู่ เพื่อทำการยุบรูปได้ถูกต้อง

ขั้นตอนดำเนินการตัดคำ มี 3 ขั้นตอน โดยในแต่ละขั้นตอนจะพิจารณาการตัดคำ จากองค์ประกอบของคำในแง่มุมต่าง ๆ กัน และทุกครั้งที่ตัดคำได้จะใส่เครื่องหมายพิเศษ "|" ลงคั่น ซึ่งเครื่องหมายพิเศษนี้ จะต้องไม่ตรงกับอักขระตัวใดทั้งสิ้น

ขั้นที่ 1

ตัดคำ เฉพาะสระและอักขระที่มีหลักเกณฑ์แน่นอน ได้แก่

1) สระที่เป็นตัวแรกเสมอ จะตัดคำหน้าสระเหล่านี้ เช่น

เ แ โ ใ ไ

เช่น เท แล โกย ใน ไอ เป็นต้น

2) สระที่เป็นตัวท้ายเสมอ จะตัดคำหลังสระเหล่านี้ เช่น

ะ ำ

เช่น กะ อำ เป็นต้น

3) เครื่องหมายที่เป็นตัวบอกสิ้นสุดคำ จะตัดคำหน้าและท้ายเครื่องหมายเหล่านี้ เช่น

ๆ ๆ

เช่น กรุงเทพฯ ดำๆ แดงๆ เป็นต้น

4) อักขระใด ๆ นอกเหนือจากตัวภาษาไทย คืออาจเป็นตัวภาษาอังกฤษหรือตัวเลข ให้ตัดคำทั้งหน้าและท้ายอักขระเหล่านี้ เช่น

เลขที่XY374 ตัดคำเป็น **เลขที่|X|Y|3|7|4**

ขั้นที่ 2

ทำการตรวจสอบ หาวรรณยุกต์ เมื่อพบก็จะตรวจต่อไปว่ามีการตัดคำจากขั้นที่ 1 แล้วหรือไม่ ถ้ายังไม่มีการตัดคำ จะทำการวิเคราะห์ เพื่อตัดคำที่มีส่วนคำที่ไม่มีวรรณยุกต์และไม่มีการตัดคำ จะผ่านเลยไป เพราะไม่มีผลต่อการเรียงลำดับ

ขั้นที่ 3

กรณีที่เป็นสระประสม ในขั้นตอนนี้จะนำข้อมูลที่ถูกตัดคำ

แล้ว มาทำการยุบรูป จากนั้น จึงทำการสลับตำแหน่งของสระ, ย้ายตำแหน่งของวรรณยุกต์และไม้ไต่คู้ไปไว้ท้ายคำ และทำการกำหนดค่าน้ำหนัก พร้อมกับส่งผลที่ได้ไปเตรียมจัดเรียง โดยจะตัดเครื่องหมายคั่นระหว่างคำ ("|") ทิ้งหมดทั้งไป

2.2.1.5 การสร้างตัวชี้

เมื่อทำเสร็จขั้นที่ 3 จะได้คีย์สำหรับใช้ในการเรียง ซึ่งถูกแปลงเสร็จเรียบร้อยแล้ว เก็บอยู่ในหน่วยความจำหลัก จากนั้น จะทำการสร้างตัวชี้ เพื่อใช้เชื่อมโยง ระหว่างแฟ้มข้อมูลเข้ากับแฟ้มข้อมูลออก โดยนำหมายเลขระเบียนขณะนั้นของแฟ้มข้อมูลเข้าเก็บเป็นเลขจำนวนเต็มบวกขนาด 2 ไบต์ ลงต่อส่วนท้ายของคีย์ ดังนั้น ขีดจำกัดของแฟ้มข้อมูลเข้า จึงอยู่ที่จำนวนระเบียน มีได้ไม่เกิน 65,535 ระเบียน ส่วนขนาดจะมีจำนวนกี่ไบต์ก็ได้ กรณีที่ต้องการเรียงลำดับแฟ้มข้อมูลเข้า ที่มีจำนวนระเบียนมากกว่านี้ สามารถทำได้โดย นำชุดคำสั่งภาษาต้นฉบับ (Source Program) ของโปรแกรมเรียงลำดับมาแก้ไขการระบุประเภทของตัวชี้ จากประเภทชอร์ต (short) ซึ่งมีขนาด 2 ไบต์ เปลี่ยนเป็นประเภทลอง (long) ซึ่งมีขนาด 4 ไบต์ อันจะทำให้ใช้งานกับแฟ้มข้อมูลที่มีจำนวนระเบียนถึง 4,294,967,296 ระเบียน อย่างไรก็ตาม การใช้ตัวชี้ขนาด 4 ไบต์ จะทำให้ประสิทธิภาพของการเรียงลำดับต่ำลง เพราะใช้เนื้อที่หน่วยความจำหลักมากขึ้น จึงต้องทำการเรียงแบบภายนอกมากครั้งขึ้น

2.2.2 ขั้นตอนหลังการเรียงลำดับ

ในขณะที่ทำการเรียงลำดับข้อมูล จะมีการสลับคีย์ให้ไปอยู่ตำแหน่งที่ถูกต้อง ซึ่งตัวชี้ที่ส่วนท้ายของคีย์นั้น จะถูกสลับตามไปด้วย จนเมื่อการเรียงลำดับเสร็จสิ้น ถ้าเป็นการเรียงจากน้อยไปมาก คีย์ที่น้อยสุดจะเรียงอยู่ลำดับแรก และคีย์ที่มากกว่าจะอยู่ลำดับถัดไป จากนั้นจึงทำขั้นตอนหลังการเรียงลำดับ

ขั้นตอนหลังการเรียงลำดับเป็นการเชื่อมโยง เอาข้อมูลเดิมของแฟ้มข้อมูลเข้ามาสร้างเป็นแฟ้มข้อมูลออกที่มีการเรียงเรียบร้อยแล้ว ขั้นตอนคือ ดึงเอาตัวชี้ของคีย์ที่น้อยที่สุดมาใช้เป็นหมายเลขระเบียนของแฟ้มข้อมูลเข้า อ่านข้อมูลระเบียนนั้นแล้วบันทึกลงเป็นระเบียนแรกของแฟ้มข้อมูลออก ทำเช่นนี้กับคีย์ถัดไป จนกระทั่งครบทุกคีย์ ตัวอย่างเช่น

ให้ระเบียนที่ 8 ของแฟ้มข้อมูลเข้า มีคีย์ภาษาไทย เป็นคำว่า **ด้ามจ๊ว** และเมื่อถูกเรียงลำดับแล้ว คีย์นี้จะอยู่เป็นลำดับที่ 3 ลักษณะของคีย์จะเป็นดังนี้

คีย์ลำดับที่ 1 กาน้_มไ้_ 4 (ก้านไม้)

คีย์ลำดับที่ 2 ขไ่_กไ่_ 6 (ไข่ไก่)

คีย์ลำดับที่ 3 ดาม้_จิ_ว_8 (ด้ามจิ๋ว)

คีย์ลำดับที่ 4 อาง่_ทอง 1 (อ่างทอง)

จะทำการอ่านระเบียนที่ 8 ของแฟ้มข้อมูลเข้า แล้วบันทึกลงเป็นระเบียนที่ 3 ของแฟ้มข้อมูลออก