บทที่ 2

วิธีการจัดรวมข้อมูลอย่างมีลำดับทั่วไป

วิธีการที่ใช้ในการจัดรวมข้อมูลอย่างมีลำดับนั้น มีด้วยกันหลายวิธี แต่ละวิธีจะมีข้อดีและข้อเสียต่างกัน บางวิธีใช้ได้กับข้อมูลเฉพาะอย่างซึ่งจะทำให้การทำงานเกิดความรวดเร็วและมีประสิทธิภาพมากยิ่งขึ้น

การจัดรวมข้อมูลอย่างมีลำดับเป็นวิธีการจัดรวมแฟ้มข้อมูลที่เรียงลำดับแล้วตั้งแต่ 2 แฟ้มข้อมูลขึ้นไป ให้รวมอยู่ในแฟ้มข้อมูลเดียวกัน การจัดรวมแฟ้มข้อมูลสามารถที่จะจัดเรียงตามคีย์ที่ได้ทำการเรียงลำดับมาก่อนแล้ว จากน้อยไปหามากหรือจากมากไปหาน้อย ซึ่งงานแต่ละชนิดจะมีลักษณะแตกต่างกัน ดังนั้นจึงได้มีผู้คิดค้นวิธีการจัดรวมข้อมูลอย่างมีลำดับขึ้นมาหลายวิธี เพื่อให้เหมาะสมกับงานและประหยัดเวลาในการทำงาน นอกจากนี้ยังได้นำไปประยุกต์ใช้ในการจัดเรียงลำดับข้อมูลอีกด้วย วิธีการจัดรวมข้อมูลอย่างมีลำดับที่สำคัญมีดังนี้

2.1 แบบ ทู-เวย์ (Two-Way Merge)

2.2 แบบ ไบนารี (Binary Merge)

2.3 แบบ ลิสท์ (List Merge)

2.4 แบบ สลับตำแหน่ง (Merge Exchange)

2.5 แบบ สมดุล (Balance Merge)

2.6 แบบ โพลี่เพลส (Polyphase Merge)

2.7 แบบ แค็สเคต (Cascade Merge)

2.1 แบบ ทู-เวย์ (Two-Way Merge) (1)

แบบ ทู-เวย์ เป็นวิธีการที่ใช้ในการจัดรวมข้อมูลอย่างง่าย ๆ วิธีหนึ่ง ซึ่งจะใช้ได้ผลดีกับงานที่มีจำนวนข้อมูลไม่มาก การทำงานจะใช้วิธีการเปรียบเทียบระหว่างแฟ้มข้อมูล 2 แฟ้ม โดยเปรียบเทียบทีละตัวของคีย์ไปเรื่อย ๆ และวิธีนี้ยังนำไปใช้ช่วยในการจัดเรียงลำดับข้อมูล

วิธีดำเนินการ แบบ ทู-เวย์

ส่วนประกอบ

- แฟ้มข้อมูลนำเข้า X ซึ่งมีข้อมูลดังนี้ $x_1 \leq x_2 \leq \ldots \leq x_m$
- แฟ้มข้อมูลนำเข้า Y ซึ่งมีข้อมูลดังนี้ $y_1 \leq y_2 \leq \ldots \leq y_n$
- แฟ้มข้อมูลนำออก Z ซึ่งจะต้องลักษณะข้อมูลดังนี้ ,

$z_1 \leq z_2 \leq \ldots \leq z_{m+n}$

1 : ให้ค่า $i \leftarrow 1$, $j \leftarrow 1$, $k \leftarrow 1$

2 : เปรียบเทียบ ถ้า $x_i \leq y_j$

ถ้าเป็นจริงให้ไปทำข้อ 3

ถ้าไม่เป็นจริงให้ไปทำข้อ 5

3 : ให้ค่า $z_k \leftarrow x_i$, $k \leftarrow k+1$, $i \leftarrow i+1$

เปรียบเทียบ ถ้า $i \leq m$

ถ้าเป็นจริงให้ไปทำข้อ 2

ถ้าไม่เป็นจริงให้ทำข้อ 4

4 : ให้ค่า $z_k \leftarrow y_j$
$\vdots \quad \vdots$
$z_{m+n} \leftarrow y_n$, เสร็จแล้วให้หยุดการทำงาน

5 : ให้ค่า $z_k \leftarrow y_j$ $k \leftarrow k+1$, $j \leftarrow j+1$

เปรียบเทียบ ถ้า $j \leq n$

ถ้าเป็นจริงให้ไปทำข้อ 2

ถ้าไม่เป็นจริงให้ไปทำข้อ 6

6 : ให้ค่า $z_k \leftarrow x_i$
$\vdots \quad \vdots$
$z_{m+n} \leftarrow x_m$, เสร็จแล้วให้หยุดการทำงาน

2.2 แบบ ไบนารี (Binary Merge)[1]

แบบไบนารี อาศัยวิธีการของการค้นหาแบบไบนารี (Binary Search) ช่วยในด้านการค้นหาตำแหน่งที่จะใส่ข้อมูลตัวใหม่ลงไปและยังช่วยลดจำนวนครั้งของการเปรียบเทียบลงอีกด้วย โดยใช้สูตร

$$t = \left\lfloor \log_2 \left\{ \begin{matrix} n/m \\ \text{or} \\ m/n \end{matrix} \right\} \right\rfloor$$

ช่วยในการหาตำแหน่งของคีย์ที่จะถูกนำมาเปรียบเทียบ ทำให้เวลาที่ใช้ไปในการทำงานลดน้อยลง เพราะเหตุว่าไม่จำเป็นต้องทำการเปรียบเทียบหมดทุกคีย์

* หมายเหตุ สัญญลักษณ์ $\lfloor x \rfloor$ หมายถึงค่า k ที่เป็นเลขจำนวนเต็มที่มากที่สุด ซึ่งค่า k จะมีค่าน้อยกว่าหรือเท่ากับค่า x

<u>วิธีดำเนินการ แบบ ไบนารี</u>

ส่วนประกอบ

- แฟ้มข้อมูล A มีข้อมูลดังนี้ $A_1 \leq A_2 \leq \ldots \leq A_m$
- แฟ้มข้อมูล B มีข้อมูลดังนี้ $B_1 \leq B_2 \leq \ldots \leq B_n$

1 : เปรียบเทียบ ถ้า m หรือ n เท่ากับ 0

ถ้าเป็นจริงให้หยุดการทำงาน

ถ้าไม่เป็นจริงให้ ;

เปรียบเทียบ ถ้า m > n

ถ้าเป็นจริงให้ $t \leftarrow \lfloor \log_2(m/n) \rfloor$

แล้วไปทำข้อ 4

ถ้าไม่เป็นจริงให้ $t \leftarrow \lfloor \log_2(n/m) \rfloor$

แล้วไปทำข้อ 2

2 : เปรียบเทียบ ถ้า $A_m : B_{n+1-2^t}$

ถ้า A_m น้อยกว่าให้ $n \leftarrow n-2^t$ แล้วกลับไปทำข้อ 1

ถ้า A_m มากกว่าให้ทำข้อต่อไป

3 : ใช้การค้นหาแบบไบนารี หาจุดที่จะใส่ค่า A_m ลงไปในช่วงระหว่าง

$B_{n+1-2^t}, \ldots, B_n$

เปรียบเทียบ หาค่า k

ถ้าค่า k เป็นค่าสูงสุด ดังนั้น $B_k < A_m$, ให้ $m \leftarrow m-1$, $n \leftarrow k$ แล้วกลับไปทำข้อ 1

ถ้าค่า k ไม่เป็นค่าสูงสุดกลับไปทำข้อ 1

4 : เปรียบเทียบ ถ้า $B_n < A_{m+1-2^t}$

ถ้าเป็นจริงให้ $m \leftarrow m-2^t$ แล้วกลับไปทำข้อ 1

ถ้าไม่เป็นจริงให้ทำข้อต่อไป

5 : ใช้การค้นหาแบบไบนารี หาจุดที่จะใส่ค่า B_n ลงไปในช่วงระหว่าง

$A_{m+1-2^t}, \ldots, A_m$

เปรียบเทียบ หาค่า k

ถ้า k เป็นค่าสูงสุด ดังนั้น $A_k < B_n$, $m \leftarrow k$

$n \leftarrow n-1$ แล้วกลับไปทำข้อ 1

ถ้า k ไม่เป็นค่าสูงสุดให้กลับไปทำข้อ 1

2.3 แบบ ลิสท์ (List Merge)[1]

แบบลิสท์ เป็นวิธีการจัดรวมข้อมูลอีกแบบ โดยอาศัยการเปรียบเทียบของคีย์เช่นกัน ต่างกันตรงที่ว่าการเคลื่อนข้อมูลจะไม่กระทำโดยการเคลื่อนข้อมูลจากที่หนึ่งไปยังอีกที่หนึ่งดังเช่นวิธีอื่น แต่จะอาศัยรายชื่อตัวเชื่อม (Link list) ช่วยโดยการเปลี่ยนแปลงเพียงค่าในรายชื่อตัวเชื่อมเท่านั้น เมื่อทำการจัดรวมข้อมูลอย่างมีลำดับเสร็จเรียบร้อยแล้วจะทำการเรียกตำแหน่งเริ่มต้นของตัวเชื่อมเท่านั้นก็จะได้ผลลัพธ์ออกมา

วิธีดำเนินการ แบบ ลิสท์

ให้ระเบียนข้อมูล $R_1, \ldots, R_n$ มีคีย์ $K_1, \ldots, K_n$ และมีเขตข้อมูลเก็บตัวเชื่อม (Link field) $L_1, \ldots, L_n$ นอกจากนี้ยังมีเขตข้อมูลเก็บตัวเชื่อมสำรอง L_0 และ L_{n+1} ของระเบียนข้อมูล R_0 และ R_{n+1} ซึ่งเป็นระเบียนตัวเริ่มต้นและตัวสุดท้าย

การจัดเรียงลำดับข้อมูลจะเรียงจากน้อยไปหามาก หลังจากจัดเรียงลำดับข้อมูลเสร็จ L_0 จะเป็นดัชนีของระเบียนที่มีคีย์เล็กที่สุด และสำหรับ $1 \leq k \leq N$, L_k จะเป็นดัชนีของระเบียนตัวที่ R_k หรือ $L_k = 0$ ถ้า R_k เป็นระเบียนที่มีคีย์ใหญ่ที่สุด

ในการทำงาน R_0 และ R_{n+1} จะใช้เป็นตำแหน่งเริ่มต้นของตัวเชื่อมลิสท์ 2 ลิสท์ ซึ่งลิสท์ย่อยทั้ง 2 จะถูกจัดรวมเข้าหากัน กรณีตัวเชื่อมติดลบจะหมายถึงจุดจบของลิสท์ย่อยที่ถูกจัดเรียงลำดับแล้ว ถ้าตัวเชื่อมเป็น 0 หมายถึงจุดจบของลิสท์

$|L_s| \leftarrow p$ หมายถึง ให้ L_s มีค่าเท่ากับ p หรือ -p ขึ้นอยู่กับเครื่องหมายของ L_s ตัวก่อนหน้านี้

1 : (ตระเตรียม 2 list) ให้ $L_0 \leftarrow 1$, $L_{n+1} \leftarrow 2$, $L_i \leftarrow -(i+2)$
สำหรับ $1 \leq i \leq n-2$ และ $L_{n-1} \leftarrow L_n \leftarrow 0$

2 : ให้ $s = 0$, $t \leftarrow N+1$, $p \leftarrow L_s$, $q \leftarrow L_t$
เปรียบเทียบ ถ้า $q = 0$
ถ้าเป็นจริงให้หยุดการทำงาน
ถ้าไม่เป็นจริงให้ทำข้อต่อไป

3 : เปรียบเทียบ ถ้า $K_p > K_q$
ถ้าเป็นจริงให้ไปทำข้อ 6
ถ้าไม่เป็นจริงให้ทำข้อต่อไป

4 : ให้ $|L_s| \leftarrow p$, $s \leftarrow p$, $p \leftarrow L_p$
เปรียบเทียบ ถ้า $p > 0$
ถ้าเป็นจริงให้ไปทำข้อ 3
ถ้าไม่เป็นจริงให้ทำข้อต่อไป

5 : ให้ $L_s \leftarrow q$, $s \leftarrow t$
ดังนั้น ให้ $t \leftarrow q$ และ $q \leftarrow L_q$ จนกระทั่ง $q \leq 0$
เสร็จแล้วให้ไปทำข้อ 8

6 : ให้ $|L_s| \leftarrow q$, $s \leftarrow q$, $q \leftarrow L_q$

เปรียบเทียบ ถ้า $q > 0$

ถ้าเป็นจริงให้ไปทำข้อ 3

ถ้าไม่เป็นจริงให้ทำข้อต่อไป

7 : ให้ $L_s \leftarrow p$, $s \leftarrow t$

ดังนั้น ให้ $t \leftarrow p$, $p \leftarrow L_p$ จนกระทั่ง $p \leq 0$

8 : ให้ $p \leftarrow -p$, $q \leftarrow -q$

เปรียบเทียบ ถ้า $q = 0$

ถ้าเป็นจริงให้ $|L_s| \leftarrow p$

$|L_t| \leftarrow o$ เสร็จแล้วไปทำข้อ 2

ถ้าไม่เป็นจริงให้ไปทำข้อ 3

2.4 แบบ ส่ลับตำแหน่ง (Merge Exchange)[1]

แบบส่ลับตำแหน่ง ได้ถูกนำไปใช้ช่วยในการจัดเรียงลำดับข้อมูล โดยการจับคู่กันเปรียบเทียบเป็นคู่ไม่มีการคาบเกี่ยวกัน แล้วทำการจัดรวมคู่ของข้อมูลที่ทำการจัดเรียงลำดับแล้วภายหลัง

<u>วิธีดำเนินการ แบบ ส่ลับตำแหน่ง</u>

ระเบียน $R_1, \ldots, R_n$ หลังจากจัดเรียงลำดับแล้วตามคีย์

$K_1 \leq , \ldots, \leq K_n$ สมมุติให้ $N \geq 2$

1 : ให้ $p \leftarrow 2^{t-1}$, โดยที่ $t \leftarrow \lceil \log_2 N \rceil$ เป็นเลขจำนวนเต็มที่น้อยที่สุด เพื่อว่า $2^t \geq N$

2 : ให้ $q \leftarrow 2^{t-1}$, $r \leftarrow 0$, $d \leftarrow p$

3 : เปรียบเทียบ สำหรับทุก ๆ i ที่ $0 \leq i \leq N-d$ และ $i \wedge p = r$

ถ้าเป็นจริงให้ไปทำข้อ 4

ถ้าไม่เป็นจริงให้ไปทำข้อ 5

4 : เปรียบเทียบ ถ้า $K_{i+1} > K_{i+d+1}$

ถ้าเป็นจริงให้เปลี่ยนกลับค่า $R_{i+1} \leftrightarrow R_{i+d+1}$ ทำต่อข้อ 3

ถ้าไม่เป็นจริงให้ทำข้อ 6

5 : เปรียบเทียบ ถ้า $q \neq p$

ถ้าเป็นจริงให้ $d \leftarrow q-p$, $q \leftarrow q/2$, $r \leftarrow p$

แล้วกลับไปทำข้อ 3

ถ้าไม่เป็นจริงให้ทำข้อต่อไป

6 : ให้ $p \leftarrow \lfloor p/2 \rfloor$

เปรียบเทียบ ถ้า $p > 0$

ถ้าเป็นจริงให้กลับไปทำข้อ 2

ถ้าไม่เป็นจริงให้หยุดการทำงาน

* หมายเหตุ สัญญลักษณ์ $\lceil x \rceil$ หมายถึงค่า k ที่เป็นเลขจำนวนเต็มที่น้อยที่สุด ซึ่งค่า k จะมีค่ามากกว่าหรือเท่ากับค่า x

2.5 แบบ สมดุล (Balance Merge)[1]

วิธีการนี้ช่วยในการจัดเรียงลำดับข้อมูล ในกรณีที่มีข้อมูลมากเกินไปที่จะเก็บไว้ในหน่วยความจำหลักทั้งหมดได้ จึงได้ทำการจัดแบ่งข้อมูลออกเป็นส่วน ๆ เท่า ๆ กัน แล้วจัดเรียงข้อมูลแต่ละส่วนได้แฟ้มข้อมูลย่อยหลาย ๆ แฟ้ม นำแฟ้มข้อมูลย่อยมารวมกันซึ่งอาจจะรวมครั้งละ 2 แฟ้มข้อมูลย่อย หรือครั้งละ 3 แฟ้มข้อมูลย่อย หรือมากกว่านี้ก็ได้

2.6 แบบ โพลี่เพส (Polyphase Merge)[1]

แบบโพลี่เพส เป็นการจัดรวมข้อมูลด้วยเทปแม่เหล็ก โดยข้อมูลทั้งหมดจะต้องผ่านวิธีดำเนินการของการแทนที่ (Replacement Selection) เสียก่อน เพื่อทำการจัดเรียงลำดับข้อมูลเป็นชุดย่อย ๆ ซึ่งแต่ละชุดย่อยจะเรียกว่าแฟ้มข้อมูลย่อย โดยการกระจายแฟ้มข้อมูลย่อยตามวิธี Fibonacci Distribution แล้วจึงจัดรวมแฟ้มข้อมูลย่อยเข้าด้วยกัน โดยการอ่านเทปแม่เหล็กเดินหน้าทางเดียวจนจบแล้วทำการกรอเทปแม่เหล็กกลับแล้วค่อยทำงานต่อไป

การจัดรวมข้อมูลแบบโพลี่เพส จะใช้ (T-1)-way merge ตลอดเวลา เช่น ถ้ามีเทปแม่เหล็ก 3 ตัว จะใช้แบบทู-เวย์ จะทำงานดังนี้

1 : จะกระจายกลุ่มข้อมูลเรียงลำดับเริ่มต้นไปยัง T1, T2

2 : การจัดรวมกลุ่มข้อมูลเรียงลำดับจาก T1 และ T2 เก็บบน T3 ถ้า T3 มีเพียงกลุ่มข้อมูลเรียงลำดับเดียวจะหยุดการทำงาน

3 : ทำการลอกกลุ่มข้อมูลเรียงลำดับบน T3 ลงบน T1 เพียงครึ่งหนึ่งของจำนวนกลุ่มข้อมูลเรียงลำดับทั้งหมด

4 : การจัดรวมกลุ่มข้อมูลเรียงลำดับจาก T1 และ T3 เก็บบน T2, ถ้า T2 มีเพียงกลุ่มข้อมูลเรียงลำดับเดียวจะหยุดการทำงาน

5 : ทำการลอกกลุ่มข้อมูลเรียงลำดับบน T2 ลงบน T1 เพียงครึ่งหนึ่งของจำนวนกลุ่มข้อมูลเรียงลำดับทั้งหมด, แล้วกลับไปทำงานยังข้อ 2

2.7 แบบ แค็ล่เคต (Cascade Merge)[1]

วิธีการทำงานแบบแค็ล่เคตนี้ ถูกค้นพบก่อนวิธีการแบบโพลีเพส ซึ่งทั้ง 2 วิธีมีลักษณะการทำงานที่คล้ายคลึงกันมาก เริ่มแรกจะทำการกระจายชุดข้อมูลต่าง ๆ ขึ้นบนเทปแม่เหล็กก่อนเหมือนกัน แต่วิธีที่ใช้ในการกระจายชุดข้อมูลต่างกันโดยวิธีของแค็ล่เคต ใช้วิธีที่เรียกว่า Perfect Distribution

ตาราง2.1แสดง Perfect Distribution สำหรับการจัดรวมข้อมูลแบบแค็สเคต โดยใช้เทปแม่เหล็ก 6 ม้วน

level	T1	T2	T3	T4	T5
0	1	0	0	0	0
1	1	1	1	1	1
2	5	4	3	2	1
3	12	14	12	9	5
4	55	50	41	29	15
5	190	175	146	105	55

.....

Perfect Distribution หาได้จากสมการต่อไปนี้

n	a_n	b_n	c_n	d_n	e_n
$n+1$	$a_n+b_n+c_n+d_n+e_n$	$a_n+b_n+c_n+d_n$	$a_n+b_n+c_n$	a_n+b_n	a_n

เมื่อทำการกระจายกลุ่มข้อมูลเรียงลำดับแล้ว ขั้นต่อไปจะทำการจัดรวมกลุ่มข้อมูลเรียงลำดับต่าง ๆ เข้าด้วยกัน โดยวิธีของแค็สเคต จะทำการจัดรวมโดยใช้ (T-1)-way, (T-2)-way, (T-3)-way merge ไปเรื่อย ๆ ซึ่งต่างกับวิธีโพลี่เพส ใช้ (T-1)-way merge ตลอดเวลาไม่มีการเปลี่ยนแปลง ข้อแตกต่างกันนี้ทำให้วิธีแค็สเคต ทำงานดีกว่าและมีประสิทธิภาพมากกว่าวิธีโพลี่เพลส์

ในกรณีที่ใช้เทปแม่เหล็กตั้งแต่ 6 ตัวขึ้นไป

ตาราง 2.2 แสดงลักษณะการทำงานของแค็สเคต

	T1	T2	T3	T4	T5	T6	Initial run process
pass 1	1^{55}	1^{50}	1^{41}	1^{29}	1^{15}	-	190
pass 2	-	1^{5}_{*}	2^{9}	3^{12}	4^{14}	5^{15}	190
pass 3	15^{5}	14^{4}	12^{3}	9^{2}	5^{1}_{*}	-	190
pass 4	-	15^{1}_{*}	29^{1}	41^{1}	50^{1}	55^{1}	190
pass 5	190^{1}	-	-	-	-	-	190

การทำงานเริ่ม pass 2 จะทำการจัดรวมข้อมูลโดยใช้ 5-way merge จาก T1, T2, T3, T4, T5 กับ T6 จนกลุ่มข้อมูลบน T5 หมด ผลจะเก็บไว้บน T6 มีความยาวกลุ่มข้อมูลเท่ากับ 5 กลุ่มรวมกันและประกอบด้วย 15 กลุ่ม, ต่อไปใช้ 4-way merge จาก T1, T2, T3, T4 กับ T5 จนกลุ่มข้อมูลบน T4 หมด ผลจะเก็บไว้บน T5 มีความยาวกลุ่มข้อมูลเท่ากับ 4 กลุ่ม รวมกันและประกอบด้วย 14 กลุ่ม, ต่อไปใช้ 3-way merge กับ T4, 2-way merge กับ T3, ขั้นสุดท้ายใน pass 2 ใช้ 1-way merge กับ T2 เสร็จแล้วไปทำ pass 3 ในลักษณะเดียวกันต่อไป

การทำงานของวิธีแค็สเคต ไม่จำเป็นต้องมีการลอกข้อมูลไปยังเทปแม่เหล็กอีกม้วน แต่ในตารางข้างบนในกรณีใช้เทปแม่เหล็ก 6 ตัว ซึ่งจะมีเพียง 25 run จาก 950 run ซึ่งนับว่ามีจำนวนน้อยมาก