บทที่ 2

ทฤษฎีและผลงานวิจัยที่เกี่ยวข้อง

ในบทนี้กล่าวถึงรายละเอียดของทฤษฎีที่เกี่ยวข้อง ประเด็นสำคัญในงานวิจัยเรื่อง ควิกซอร์ตที่ใช้ผลต่างสืบเนื่อง และเทคนิควิธีการต่างๆ ที่นำมาใช้ในงานวิจัย รวมถึงผลงานวิจัยอื่นๆ ที่เกี่ยวกับการปรับปรุงประสิทธิภาพของควิกซอร์ต ซึ่งมีรายละเอียดดังต่อไปนี้

2.1 ทฤษฎีที่เกี่ยวข้อง

ในงานวิจัยเรื่อง ควิกซอร์ตที่ใช้ผลต่างสืบเนื่อง มีทฤษฎีเกี่ยวข้องที่สำคัญดังนี้ การจัดเรียงข้อมูล (Sorting) ควิกซอร์ต (Quicksort) และผลต่างสืบเนื่อง (Successive difference) โดยรายละเอียดมีดังต่อไปนี้

2.1.1 การจัดเรียงข้อมูล (Sorting)

การจัดเรียงข้อมูล เป็นวิธีสลับตำแหน่งที่ทำให้ข้อมูลมีการเรียงลำดับที่ถูกต้อง เพื่อการนำไปใช้งาน เช่น การหาข้อมูลที่เป็นค่าที่อยู่ลำดับกลางของชุดข้อมูล เป็นต้น และใช้ช่วยทำให้สามารถค้นข้อมูลได้สะดวกและรวดเร็วยิ่งขึ้น นอกจากนั้นการจัดเรียงข้อมูลเป็นหัวข้อที่สำคัญในการเรียนวิชาการออกแบบอัลกอริทึม

- การจัดเรียงข้อมูลเป็นงานพื้นฐานที่ใช้ในโปรแกรมประยุกต์ต่างๆ โดยข้อมูลที่มีการเรียงลำดับแล้วจะมีประโยชน์ต่อการใช้งานมากกว่าข้อมูลที่ไม่มีการเรียงลำดับ
- อัลกอริทึมที่ใช้เพื่อการจัดเรียงข้อมูลมีจำนวนมากที่เป็นตัวอย่างที่น่าสนใจ ที่ทำให้เกิดเทคนิคใหม่ๆ ที่สำคัญต่างๆ ได้แก่ Partial order, Recursion, Merge lists, การจัดเก็บ Binary tree ในอาร์เรย์ เป็นต้น
- อัลกอริทึมที่ใช้เพื่อการจัดเรียงข้อมูลแต่ละแบบมีข้อดีและข้อเสียแตกต่างกัน ขึ้นอยู่กับจำนวนชนิดของข้อมูล การกำหนดค่าเริ่มต้น ขนาดและค่าของข้อมูลที่จะทำการเรียงลำดับ สิ่งสำคัญที่ควรทำความเข้าใจ คือ การเลือกอัลกอริทึมที่เหมาะสมกับความต้องการในการทำงาน

ดังที่กล่าวมาจะเห็นได้ว่าการจัดเรียงข้อมูลมีความจำเป็นหลายประการ นอกจากจะทำให้สะดวกต่อการค้นข้อมูลแล้ว ยังมีผลต่อกระบวนการดึงข้อมูลมาใช้งาน นอกจากนี้อัลกอริทึม

หรือการคำนวณสำหรับแก้ปัญหาสลับซับซ้อนต่างๆ เช่น Job scheduling, Knapsack problem ฯลฯ ก็ต้องอาศัยการจัดเรียงข้อมูลทั้งสิ้น

การจัดเรียงข้อมูลแบ่งได้เป็นสองลักษณะ คือ การจัดเรียงภายใน (Internal sorting) และการจัดเรียงภายนอก (External sorting)

การจัดเรียงภายใน (Internal sorting) คือ การจัดเรียงข้อมูลซึ่งเกิดขึ้นในหน่วยความจำหลัก (Main memory) หมายถึงว่าจำนวนข้อมูลมีขนาดไม่ใหญ่กว่าขนาดของพื้นที่หน่วยความจำเป็นหลัก ดังนั้นข้อมูลทั้งหมดจะเก็บอยู่ในหน่วยความจำหลักและการคำนวณสำหรับการจัดเรียงสามารถอ่านข้อมูลจากหน่วยความจำหลักหรือเขียนข้อมูลสู่หน่วยความจำหลักได้โดยตรง โดยไม่จำเป็นต้องใช้หน่วยความจำสำรอง เช่น ดิสก์ (disk) หรือเทป (tape) สำหรับเก็บผลลัพธ์ และเหมาะกับข้อมูลจำนวนน้อย การเรียงลำดับประเภทนี้ได้แก่ การจัดเรียงแบบบับเบิล (Bubble sort), การจัดเรียงแบบแทรก (Insertion sort), การจัดเรียงแบบเลือก (Selection sort), การจัดเรียงแบบเชลล์ (Shell sort), การจัดเรียงแบบเรดิกซ์ (Radix sort), ควิกซอร์ต (Quick sort)

การจัดเรียงภายนอก (External sorting) คือ การจัดเรียงข้อมูลที่ใช้หน่วยความจำสำรองร่วมกับหน่วยความจำหลัก เหมาะกับข้อมูลจำนวนมากและส่วนใหญ่กระทำที่หน่วยความจำสำรอง หมายถึงถ้าข้อมูลมีจำนวนมากเกินกว่าที่จะบรรจุลงในพื้นที่ความจำหลักได้หมดในคราวเดียว จึงจำเป็นต้องแบ่งข้อมูลออกเป็นส่วนย่อยๆ ซึ่งแต่ละส่วนมีขนาดใหญ่พอที่จะอยู่ในหน่วยความจำหลักได้ และจะได้รับการจัดเรียงโดยใช้แบบการคำนวณสำหรับการเรียงภายในตามที่ผู้ออกแบบเห็นเหมาะสม หลังจากนั้นก็ต้องมีการนำข้อมูลย่อยของส่วนที่เรียงแล้วไปเก็บไว้ในดิสก์หรือหน่วยวามจำสำรองเป็นการชั่วคราว เพื่อรอที่จะรวมกับข้อมูลส่วนอื่นๆ ที่เรียงเสร็จแล้ว การจัดเรียงลำดับประเภทนี้ได้แก่ การจัดเรียงแบบผสาน (Merge sort)

ดังที่ได้กล่าวไว้ในบทแรก งานวิจัยนี้สนใจเฉพาะในส่วนของการจัดเรียงภายใน เนื่องจากปัจจุบันหน่วยความจำหลักมีขนาดใหญ่ขึ้นเพียงพอสำหรับทำงานกับข้อมูล ส่วนของวิธีการจัดเรียงที่งานวิจัยนี้เลือกใช้ในการทดลอง คือ ควิกซอร์ต เพราะเป็นการจัดเรียงที่มีประสิทธิภาพดีที่สุดกับข้อมูลขนาดกลางและขนาดใหญ่ [3] มีพฤติกรรมการสลับข้อมูลในขั้นตอนการแบ่งข้อมูลที่เหมาะสมและเป็นวิธีการจัดเรียงที่ถูกแนะนำในงานวิจัยเรื่องผลต่างสืบเนื่อง [7, 8] ในการนำมาประยุกต์ใช้กับการจัดเรียงข้อมูล

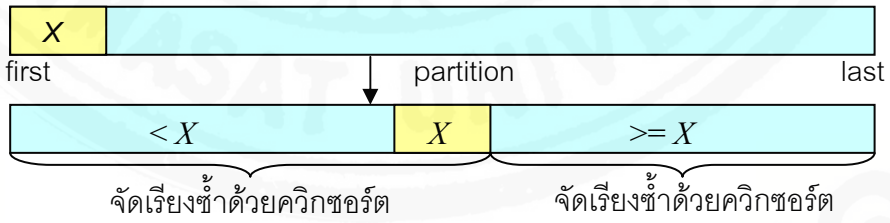
2.1.2 **ควิกซอร์ต** (Quick sort)

ควิกซอร์ตเป็นวิธีการจัดลำดับข้อมูลที่เสนอโดย C.A.R. Hoare ในปีค.ศ. 1961 [5], 1962 [6] เป็นการจัดเรียงที่ใช้เทคนิคการทำงานแบบแบ่งแยกและเอาชนะ (Divide and Conquer) ควิกซอร์ตจัดอยู่ในกลุ่มของการจัดเรียงภายใน (Internal sorting) ใช้เทคนิคการจัดเรียงแบบแลกเปลี่ยน (Sort by exchanging) ควิกซอร์ตเป็นที่ยอมรับกันว่าเป็นวิธีการจัดเรียงที่มีประสิทธิภาพที่สุดของการจัดเรียงภายใน

ขั้นตอนการทำงานของควิกซอร์ตที่ใช้เทคนิคแบบแบ่งแยกและเอาชนะ แบ่งการทำงานเป็น 3 ส่วนหลักๆ คือ

Divide: เป็นขั้นตอนการแบ่งข้อมูลให้เป็นสามกลุ่ม คือ กลุ่มที่ค่ามากกว่าตัวหลัก กลุ่มที่ค่าเท่ากับตัวหลักและกลุ่มที่ค่าน้อยกว่าตัวหลัก แต่ในปัจจุบันเหลือแค่สองกลุ่ม คือ กลุ่มที่ค่ามากกว่าหรือเท่ากับตัวหลักและกลุ่มที่ค่าน้อยกว่าตัวหลัก หรือกลุ่มที่ค่ามากกว่าตัวหลักและกลุ่มที่ค่าเท่ากับหรือน้อยกว่าตัวหลัก โดยควิกซอร์ตเรียกขั้นตอนนี้ว่าการแบ่งข้อมูล (Partition) โดยทำการเลือกข้อมูลจากชุดข้อมูลขึ้นมาหนึ่งตัว เรียกข้อมูลตัวที่เลือกขึ้นมาว่า ตัวหลัก (Pivot) และจะใช้ตัวหลักเป็นตัวแบ่งข้อมูลออกเป็น 2 ส่วน ดังรูปที่ 2.1

รูปที่ 2.1
การแบ่งข้อมูลของควิกซอร์ต

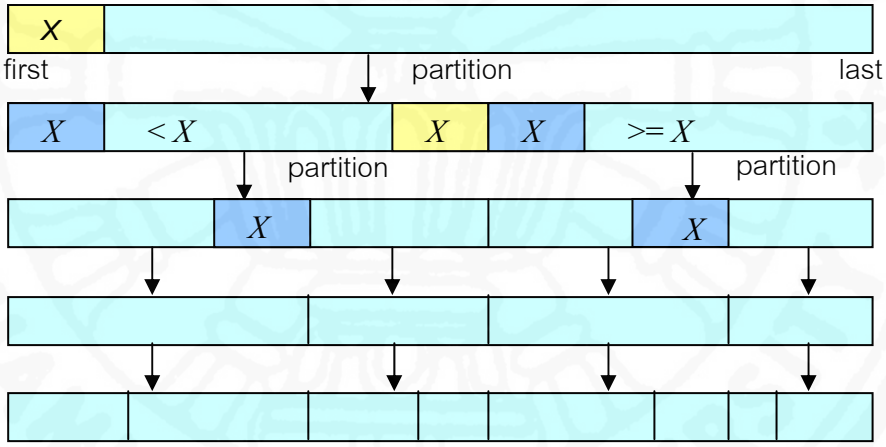


วิธีการเลือกตัวหลักของควิกซอร์ตที่เสนอโดย Hoare จะเลือกตัวหลักจากข้อมูลตัวแรก แต่ในปัจจุบันได้มีการเสนอวิธีการเลือกตัวหลักอีกหลายแบบเพื่อแก้ปัญหากรณีแย่ที่สุดจากการเลือกข้อมูลตัวแรก เช่น เลือกจากตำแหน่งกลาง, Median of three เป็นต้น เมื่อเลือกตัวหลักได้จากนั้นจะทำการตรวจข้อมูลตามเงื่อนไขของการจัดเรียง คือ ทำการตรวจข้อมูลจากตำแหน่งแรกโดยใช้ตัวชี้ไปทางขวาจนพบข้อมูลที่ค่ามากกว่าค่าของตัวหลักให้หยุดการตรวจข้อมูล และทำการตรวจข้อมูลจากตำแหน่งท้ายไปทางซ้ายจนพบข้อมูลที่ค่าน้อยกว่าตัวหลัก ทำการสลับค่าข้อมูล

ระหว่างคู่ข้อมูลที่ตรวจพบ จากนั้นให้ทำการตรวจต่อจากตำแหน่งเดิมทั้งสองด้าน จนกระทั่งค่าตัวชี้ด้านท้ายน้อยกว่าค่าตัวชี้ทางด้านหน้า นำค่าหลักสลับกับตัวชี้ทางด้านท้าย ซึ่งผลของการแบ่งข้อมูลจะทำให้ได้ข้อมูลย่อยสองส่วน คือ ส่วนแรกอยู่ในตำแหน่งตอนหน้าของชุดข้อมูล ซึ่งข้อมูลในส่วนนี้มีค่าน้อยกว่าหรือเท่ากับค่าตัวหลัก และส่วนที่สองอยู่ในตอนหลัง ซึ่งข้อมูลในส่วนนี้มีค่ามากกว่าค่าตัวหลัก

Conquer: นำข้อมูลย่อยแต่ละส่วนที่ได้จากการแบ่งข้อมูลมาทำการแบ่งข้อมูลด้วยวิธีการเดิมไปจนกระทั่งไม่สามารถแบ่งข้อมูลได้อีก ดังรูปที่ 2.2

รูปที่ 2.2
การวนซ้ำแบ่งข้อมูลของควิกซอร์ต

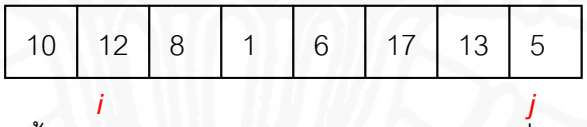


Combine: เป็นขั้นตอนการนำข้อมูลในแต่ละส่วนมารวมกันกลับเป็นข้อมูลชุดเดียว แต่ในปัจจุบันไม่มีขั้นตอนนี้เพราะควิกซอร์ตปัจจุบันจะใช้งานบนพื้นที่หน่วยความจำเดิมใช้การสลับเปลี่ยนตำแหน่งข้อมูลในชุดข้อมูล (In-place) ทำให้เมื่อเสร็จขั้นตอน Conquer ชุดข้อมูลที่ได้จะเป็นชุดข้อมูลที่เรียงลำดับ

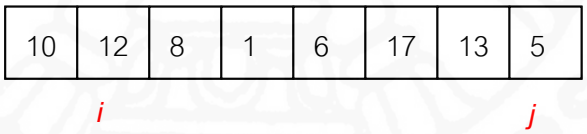
ในส่วนนี้เป็นการอธิบายเพิ่มเติมในส่วนขั้นตอนการแบ่งข้อมูล (Partition) ของควิกซอร์ต โดยเริ่มจากเลือกค่าตัวหลักจากข้อมูลตัวแรก กำหนดให้เป็นค่า X จากนั้นให้ทำการตรวจสอบค่าข้อมูลจากด้านหลังย้อนขึ้นมาด้านหน้าโดยใช้ตัวชี้ j เริ่มจากตำแหน่งท้ายสุด หยุดการตรวจสอบเมื่อพบค่าข้อมูลที่มีค่าน้อยกว่าหรือเท่ากับค่า X ต่อไปให้ตรวจสอบค่าข้อมูลจากด้านหน้าไปยังด้านหลัง โดยใช้ตัวชี้ i เริ่มถัดจากตำแหน่งแรก หยุดตรวจสอบเมื่อพบค่าข้อมูลที่มีค่ามากกว่าค่า X ถ้าค่า i ยังน้อยกว่าค่า j ให้สลับค่ากันระหว่างค่าในตำแหน่ง i และ ค่าใน

ตำแหน่ง *j* จากนั้นกลับมาตรวจสอบค่าของข้อมูลต่อโดยเริ่มจากส่วนท้ายก่อน โดยเริ่มทำต่อจากตำแหน่ง *j* ขึ้นมาด้านหน้า หยุดเมื่อพบค่าที่น้อยกว่าหรือเท่ากับค่า *X* กลับมาตรวจสอบด้านหน้าโดยทำต่อที่ตำแหน่ง *i* ตรวจสอบไปจนกระทั่งพบค่าที่มากกว่าค่า *X* ถ้าค่าของ *i* ยังน้อยกว่า *j* อยู่ก็จะสลับค่าข้อมูลระหว่างตำแหน่งที่ *i* กับ *j* ทำการพิจารณาไปจนกระทั่งค่า *i* มากกว่าค่า *j* จึงหยุดตรวจสอบ นำค่า X แทนที่ค่าข้อมูลตำแหน่ง *i* เมื่อถึงขั้นตอนนี้ชุดข้อมูลถูกแบ่งออกเป็น 2 ส่วน ประกอบด้วยข้อมูลตอนหน้า ค่าข้อมูลที่อยู่ในส่วนนี้มีค่าน้อยกว่าหรือเท่ากับค่า *X* และข้อมูลตอนหลังข้อมูลมีค่ามากกว่าค่า X จากนั้นให้นำข้อมูลย่อยแต่ละส่วนไปแบ่งข้อมูลด้วยวิธีการแบบเดียวกัน ดำเนินการไปจนกระทั่งทุกส่วนไม่สามารถแบ่งเป็นส่วนย่อยได้อีกต่อไป นั่นคือทุกๆ ชุดข้อมูลย่อยมีขนาดน้อยกว่า 2

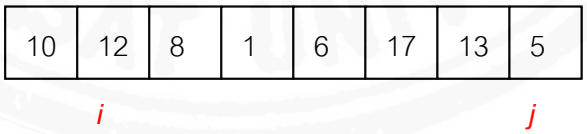
ตัวอย่าง การทำงานของควิกซอร์ต โดยให้ชุดข้อมูล E มีขนาด 8 ข้อมูลดังนี้ : 10,12,8,1,6,17,13,5 ทำการเลือกตัวหลักแบ่งข้อมูลโดยจะใช้ข้อมูลตัวแรกจะได้ว่า Pivot = E[1] = 10 ให้ *i* เป็นตัวชี้ที่เริ่มจากตำแหน่งที่สองของข้อมูลเพราะตัวแรกถูกใช้เป็นตัวหลักแบ่งข้อมูล และ *j* เป็นตัวชี้ที่เริ่มจากตำแหน่งท้ายของชุดข้อมูล



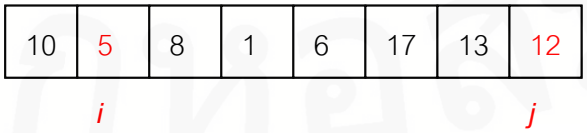
ทำการเดินตัวชี้ j มาย้อนด้านหน้า(ไปทางซ้าย) และหยุดเมื่อค่า E[*j*] <= 10



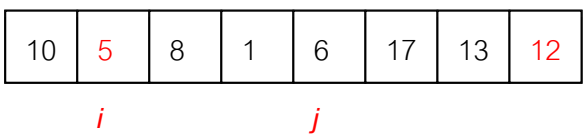
ทำการเดินตัวชี้ i ไปด้านท้ายชุดข้อมูล(ไปทางขวา) หยุดเมื่อค่า E[*i*] > 10



ทำการสลับค่าข้อมูลระหว่างข้อมูลตำแหน่งตัวชี้ i และตัวชี้ j



ทำการเดินตัวชี้ j มาย้อนด้านหน้าต่อ(ไปทางซ้าย) หยุดเมื่อค่า E[j] <= 10



ทำการเดินตัวชี้ i ไปด้านท้ายชุดข้อมูลต่อ(ไปทางขวา) หยุดเมื่อค่า E[i] > 10

10	5	8	1	6	17	13	12
				j	*i*		

ถ้าค่าตำแหน่งของตัวชี้ *i* < *j* ให้ทำการสลับค่าข้อมูลระหว่างข้อมูลตำแหน่งตัวชี้ *i* และตัวชี้ *j* ตามปกติและทำการเดินตัวชี้ต่อไป จนค่าตำแหน่งของตัวชี้ *i* > *j* แสดงว่าเดินตัวชี้จนครบทุกข้อมูลแล้วให้ทำการนำค่าข้อมูลลำดับที่ตัวชี้ *j* สลับกับค่าที่ตำแหน่งที่ 1 หรือตำแหน่งตัวหลักแบ่งข้อมูล

6	5	8	1	10	17	13	12
				j	*i*		

ถึงขั้นตอนนี้ชุดข้อมูลจะถูกแบ่งออกเป็นสองส่วน คือ ส่วนซ้ายของตัวหลักแบ่งข้อมูลเป็นข้อมูลที่มีค่าน้อยกว่าหรือเท่ากับค่าของตัวหลักซึ่งในตัวอย่างนี้ก็คือ น้อยกว่าหรือเท่ากับ 10 และส่วนขวาของตัวหลักเป็นข้อมูลที่มีค่ามากกว่าค่าตัวหลัก จากนั้นนำข้อมูลแต่ละส่วนไปทำการแบ่งกลุ่มด้วยกระบวนการเดิมซ้ำไปจนกระทั่งไม่สามารถแบ่งได้อีกก็จะเสร็จสิ้นขั้นตอนการเรียงข้อมูลของควิกซอร์ต

ควิกซอร์ตที่แสดงอยู่ในรูปแบบรหัสเทียม (Pseudo code) เป็นดังนี้

```
QUICKSORT(E, first, last)
    If(first < last) then
            Pivot = PARTITION(E, first, last);
            QUICKSORT(E, first, pivot-1);
            QUICKSORT(E, pivot+1, last);
end;

PARTITION(E, first, last)
    p = E[first];
    i = first+1;
    j = last;
    while(I < j ){
```

```
            while(E[j] > p) j = j-1;
            while(E[i] <= p) i = i+1;
            If(i<j) swap(E[i],E[j]);
        }
        swap(E[first],E[j]);
        return j;
    end;
```

การวิเคราะห์เวลาทำงานของควิกซอร์ต

เริ่มจากส่วนที่เป็นการแบ่งข้อมูลก่อน โดยพิจารณาการเปลี่ยนแปลงค่าของตัวชี้ (*j-i*) ภายในการวนซ้ำแบบ while พบว่าค่าเริ่มต้นเท่ากับ *n*+1 (กำหนดให้ข้อมูลในแถวลำดับที่แบ่งมี *n* ตัว) (*j-i*) มีค่าลดลงทีละหนึ่งจนหลุดจากการวนซ้ำ เมื่อ (*j-i*) <= 0 ดังนั้นการแบ่งข้อมูลใช้เวลา $\Theta(n)$

ส่วนต่อไปเป็นเวลาทำงานของควิกซอร์ต โดยกำหนดให้ $t(n)$ คือ เวลาการทำงานของควิกซอร์ต เพื่อเรียงลำดับข้อมูล *n* ตัว โดยผลการแบ่งส่วนข้อมูล *n* ตัว ได้ข้อมูลชุดซ้ายจำนวน *k* ตัว ($1 < k \leq n$) จะได้ว่า $t(n) = t(k) + t(n-k) + \Theta(n)$ โดยจะแยกพิจารณาเป็น 3 กรณี

กรณีแย่ที่สุด เกิดขึ้นเมื่อ $k = 1$ และจะต้องเกิดการแบ่งในลักษณะนี้ในทุกระดับของการเรียกวนซ้ำ จะได้ $t_{worst}(n) = t(1) + t_{worst}(n-1) + \Theta(n)$ ใช้วิธีคลี่ความสัมพันธ์วนซ้ำจะได้ $t_{worst}(n) = \Theta(n^2)$

กรณีดีที่สุด เกิดขึ้นเมื่อ k = n/2 และจะต้องเกิดการแบ่งในลักษณะนี้ในทุกระดับของการเรียกวนซ้ำ จะได้ $t_{best}(n) = 2\, t_{best}(n/2) + \Theta(n) = \Theta(n \log n)$

กรณีเฉลี่ย จากสองกรณีข้างต้นเห็นได้ว่า เวลาการทำงานนั้นขึ้นกับผลการแบ่งข้อมูลว่าจะได้ค่า *k* มีค่าเท่าไร สำหรับกรณีเฉลี่ยนั้นจะถือว่าผลของการแบ่งส่วนที่มีหลายๆ แบบนั้นแต่ละแบบมีโอกาสเกิดขึ้นได้เท่าๆ กัน เมื่อข้อมูลไม่มีตัวซ้ำกัน การแบ่งจะแบ่งข้อมูลได้ชุดซ้ายมีจำนวนข้อมูล *k* ตัว ก็เมื่อตัวหลักแบ่งข้อมูลเป็นข้อมูลตัวที่ *k*+1 สำหรับ $1 \leq k \leq n$-1 สำหรับกรณีที่ตัวหลักเป็นข้อมูลที่น้อยที่สุด จะได้ *k*=1 สรุปว่าเมื่อข้อมูลมี *n* ตัว การแบ่งข้อมูลจะแบ่งได้ *n* แบบ คือ *k* = 1,2,3,4,…,*n*-1 ดังนั้นเวลาการทำงานของควิกซอร์ตโดยเฉลี่ยจะเท่ากับ

$$t(n) = \frac{1}{n}\left(t(1) + t(n\text{-}1) + \sum_{k=1}^{n\text{-}1} t(k) + t(n\text{-}k)\right) + \Theta(n)$$

$$= \frac{1}{n}\sum_{k=1}^{n\text{-}1} t(k) + \frac{1}{n}\sum_{k=1}^{n\text{-}1} t(n\text{-}k) + \frac{1}{n}(t(1) + t(n\text{+}1)) + \Theta(n)$$

$$t(n) = \frac{2}{n}\sum_{k=1}^{n\text{-}1} t(k) + \Theta(n)$$

เราสามารถหาผลเฉลยของความสัมพันธ์วนซ้ำ ได้ดังนี้

แทนค่า $\Theta(n)$ ด้วย cn จะได้

$$t(n) = \frac{2}{n}\sum_{k=1}^{n\text{-}1} t(k) + cn$$

คูณด้วย n ตลอดจะได้

$$nt(n) = 2\sum_{k=1}^{n\text{-}1} t(k) + cn^2$$

แทน n ด้วย n-1 จะได้

$$(n\text{-}1)t(n\text{-}1) = 2\sum_{k=1}^{n\text{-}2} t(k) + c(n\text{-}1)^2$$

จากนั้นนำทั้งสองสมการด้านบนมาลบกัน หารตลอดด้วย $n(n\text{+}1)$ จากนั้นคลี่ความสัมพันธ์วนซ้ำจะได้

$$\frac{t(n)}{n\text{+}1} = \frac{t(n\text{-}1)}{n} + \frac{c'}{n\text{+}1}$$

$$t(n) < c'(n\text{+}1)H_{n+1}$$

$$t(n) = \Theta(n \log n)$$

กรณีเฉลี่ยของควิกซอร์ตที่ได้ ถ้านำข้อมูลเข้าที่เป็นไปได้ทั้งหมดมาเฉลี่ยกันแล้วโดยที่โอกาสแต่ละแบบจะเกิดขึ้นเท่ากัน ควิกซอร์ตทำงานในเวลาที่เป็น $\Theta(n \log n)$ แต่ก็มีโอกาสเกิดกรณีที่แย่ที่สุด ถ้าข้อมูลที่เข้ามาเป็นข้อมูลที่ทำให้เกิดกรณีที่แย่ที่สุด

ตารางที่ 2.1

แสดงการเปรียบเทียบเวลาการทำงานของแต่ละวิธีการจัดเรียง

Algorithm	Time(average)	Time(worst)	Note
Selection sort	$\Theta(n^2)$	$\Theta(n^2)$	• In-place
			• Slow (good for small inputs)
Insertion sort	$\Theta(n^2)$	$\Theta(n^2)$	• In-place
			• Slow (good for small inputs)
Quick sort	$\Theta(n \log n)$ expected	$\Theta(n^2)$	• In-place, randomized
			• Fastest (good for large inputs)
Heap sort	$\Theta(n \log n)$	$\Theta(n \log n)$	• In-place
			• Fast (good for large inputs)
Merge sort	$\Theta(n \log n)$	$\Theta(n \log n)$	• Sequential data access
			• Fast (good for huge inputs)

ข้อดี ข้อด้อย ของควิกซอร์ต

ข้อดี

- ทำงานได้เร็วมากกับชุดข้อมูลขนาดใหญ่
- เป็นการเรียงข้อมูลแบบ In-place ทำให้ใช้พื้นที่หน่วยความจำน้อย
- เวลาทำงานเฉลี่ยเป็น $\Theta(n \log n)$
- Implement ง่าย สามารถใช้งานได้จริง

ข้อด้อย

- การทำงานเป็นการวนซ้ำแบบ Recursive ถ้าต้องเปลี่ยนการทำงานเป็นวนซ้ำแบบ Iteration จะยากต่อการสร้างเพื่อการใช้งานมาก
- ในกรณีที่แย่ที่สุดจะใช้เวลาทำงาน $\Theta(n^2)$
- การทำงานช้ามากถ้าข้อมูลที่นำมาจัดเรียงมีค่าซ้ำๆ กัน

2.1.3 **ผลต่างสืบเนื่อง** (Successive Difference)

ผลต่างสืบเนื่อง คือ ค่าผลบวกสะสมของผลลบระหว่างข้อมูลที่มีลำดับติดกันแบบไม่คิดเครื่องหมาย สำหรับชุดข้อมูลที่มีขนาด *n* ตัว ค่าผลต่างสืบเนื่องของชุดข้อมูล สามารถคำนวณหาได้จากสมการ ดังนี้

$$d(k_1,k_2,...,k_n)=\sum_{i=1}^{n-1}\left|x_{k_{i+1}}-x_{k_i}\right| \quad (1)$$

เมื่อ k_1, k_2, ..., k_n คือ ดรรชนีของข้อมูล และ x_i คือ ค่าข้อมูลที่ดรรชนี i และ n คือ จำนวนชุดข้อมูล ตัวอย่างการคำนวนหาค่าผลต่างสืบเนื่อง โดยกำหนดให้ชุดข้อมูลขนาด 6 ข้อมูล มีข้อมูลดังนี้ k = 9, 5, 1, 3, 8, 2 จากชุดข้อมูลตัวอย่างค่าผลต่างสืบเนื่องของชุดข้อมูลเท่ากับ

$d(k_1,k_2,k_3,k_4,k_5,k_6)$ = |9-5| + |5-1| + |1-3| + |3-8| + |8-2| = 4 + 4 + 2 + 5 + 6 = 21

ค่าผลต่างสืบเนื่องของข้อมูลที่ไม่เรียงลำดับอีกแบบ

$d(k_4,k_1,k_6,k_2,k_5,k_3)$ = |3-9| + |9-2| + |2-5| + |5-8| + |8-1| = 6 + 7 + 3 + 3 + 7 = 26

ค่าผลต่างสืบเนื่องของข้อมูลที่เรียงลำดับจากค่าน้อยไปหาค่ามาก

$d(k_3,k_6,k_4,k_2,k_5,k_1)$ = |1-2| + |2-3| + |3-5| + |5-8| + |8-9| = 1 + 1 + 2 + 3 + 1 = 8

ค่าผลต่างสืบเนื่องของข้อมูลที่เรียงลำดับจากค่ามากไปหาค่าน้อย

$d(k_1,k_5,k_2,k_4,k_6,k_3)$ = |9-8| + |8-5| + |5-3| + |3-2| + |2-1| = 1 + 3 + 2 + 1 + 1 = 8

จากตัวอย่างการคำนวณค่าผลต่างสืบเนื่องของชุดข้อมูลจะเห็นได้ว่าค่าผลต่างสืบเนื่องของชุดข้อมูลที่เรียงลำดับ ไม่ว่าเป็นการเรียงลำดับจากค่าน้อยไปหาค่ามากหรือจากค่ามากไปหาค่าน้อย ค่าผลต่างสืบเนื่องจะเท่ากันเสมอและสำหรับข้อมูลที่ไม่เรียงลำดับจะมีค่าผลต่างสืบเนื่องมากกว่าเสมอ

จากงานวิจัย [7,8] อธิบายถึงสมบัติอีกอย่างของวิธีการผลต่างสืบเนื่อง คือ ค่าผลต่างสืบเนื่องของชุดข้อมูลที่เรียงลำดับจะมีค่าเท่ากับผลต่างระหว่างค่าที่มากที่สุดกับค่าที่น้อยที่สุด จากชุดข้อมูลตัวอย่างจะเห็นว่าค่าที่มากที่สุดของชุดข้อมูลตัวอย่าง คือ 9 ส่วนค่าที่น้อยที่สุดของชุดข้อมูลตัวอย่าง คือ 1 ผลลบระหว่างค่าที่มากที่สุดกับค่าที่น้อยที่สุดเท่ากับ 9 - 1 = 8 ซึ่งจะเห็นว่าค่าผลลบระหว่างค่าที่มากที่สุดกับค่าที่น้อยที่สุดของชุดข้อมูลมีค่าเท่ากับค่าผลต่างสืบเนื่องของชุดข้อมูลทั้งแบบเรียงจากค่าน้อยไปหาค่ามากหรือเรียงจากค่ามากไปหาค่าน้อย

ผลต่างสืบเนื่องสามารถทำให้เราสามารถตรวจสอบการเรียงลำดับของชุดข้อมูลได้ เพราะจากตัวอย่างจะเห็นว่าถ้าชุดข้อมูลที่เรียงลำดับ ค่าผลต่างสืบเนื่องของชุดข้อมูลจะมีค่าเท่ากับผลลบระหว่างค่าที่มากที่สุดกับค่าที่น้อยที่สุดในชุดข้อมูล แต่หากเป็นชุดข้อมูลที่ไม่

เรียงลำดับแล้วค่าผลต่างสืบเนื่องจะมากกว่าค่าผลต่างระหว่างค่าที่มากที่สุดกับค่าที่น้อยที่สุดเสมอ โดยจาก [7,8] ได้แสดงการพิสูจน์ข้อสรุปได้ดังนี้

ข้อพิสูจน์ที่ 1. ในชุดข้อมูลที่เรียงลำดับแบบจากค่าน้อยไปหาค่ามากขนาด *n* ข้อมูล พบว่าค่าผลต่างสืบเนื่องของชุดข้อมูลจะมีค่าเท่ากับผลต่างระหว่างข้อมูลตัวที่ค่ามากที่สุดกับข้อมูลตัวที่ค่าน้อยที่สุด

พิสูจน์. ให้ $x_{a_1} \le x_{a_2} \le ... \le x_{a_n}$ เป็นชุดข้อมูลแบบเรียงลำดับจากค่าน้อยสุดไปหาค่ามากสุด $d(a_1, a_2, ..., a_n)$

$$\begin{aligned}
&= \sum_{i=1}^{n-1} \left| x_{a_{i+1}} - x_{a_i} \right| = \left| x_{a_2} - x_{a_1} \right| + ... + \left| x_{a_n} - x_{a_{n-1}} \right| \\
&= (x_{a_2} - x_{a_1}) + (x_{a_3} - x_{a_2}) + ... + (x_{a_n} - x_{a_{n-1}}) \\
&= x_{a_n} - x_{a_1}.
\end{aligned}$$

เพราะฉะนั้น ค่าผลต่างสืบเนื่องเท่ากับค่าผลต่างระหว่างข้อมูลที่ค่ามากที่สุดกับข้อมูลที่ค่าน้อยที่สุด เช่นเดียวกันสำหรับชุดข้อมูลที่เรียงลำดับจากค่ามากไปหาค่าน้อย ก็จะมีค่าผลต่างสืบเนื่องที่เท่ากันเพียงแค่ข้อมูลสลับตำแหน่งกันเท่านั้น ดังนั้น

ผลที่ตามมาที่ 1. ในชุดข้อมูลแบบเรียงลำดับจากค่ามากไปหาค่าน้อยขนาด *n* ข้อมูล ค่าผลต่างสืบเนื่องของชุดข้อมูลจะมีค่าเท่ากับผลต่างระหว่างข้อมูลตัวที่มีค่ามากที่สุดกับข้อมูลตัวที่มีค่าน้อยที่สุด

เช่นกัน $x_{d_1} \ge x_{d_2} \ge ... \ge x_{d_n}$ สามารถเขียนได้อีกแบบ $-x_{d_1} \le -x_{d_2} \le ... \le -x_{d_n}$ จากข้อพิสูจน์ที่ 1 จะได้ว่า $-x_{d_n} - (-x_{d_1}) = x_{d_1} - x_{d_n}$ ซึ่งก็คือค่าข้อมูลตัวที่มีค่ามากสุดลบกับข้อมูลตัวที่มีค่าน้อยสุดเช่นเดิม

อีกส่วนที่สำคัญของแนวคิดผลต่างสืบเนื่อง คือ ถ้าชุดข้อมูลที่นำมาเรียงเป็นข้อมูลที่ไม่เรียงลำดับ ค่าผลต่างสืบเนื่องของชุดข้อมูลจะมีค่ามากกว่าค่าของผลต่างของข้อมูลตัวที่ค่ามากที่สุดกับข้อมูลตัวที่ค่าน้อยที่สุด ซึ่งสามารถแสดงการพิสูจน์ได้ดังนี้

ข้อพิสูจน์ที่ 2. ให้ $S = (X_1, X_2, ..., X_n)$ เป็นชุดข้อมูลที่มีข้อมูลภายในเป็นค่าที่ไม่เหมือนกัน *l* เป็นข้อมูลตัวที่ค่าต่ำสุดและ *u* เป็นข้อมูลตัวที่ค่าสูงสุด ถ้า $(X_1, X_2, ..., X_n)$ ไม่ใช้ชุดข้อมูลที่เรียงลำดับแล้วค่าผลต่างสืบเนื่องจะมากกว่าค่ามากสุดลบกับค่าต่ำสุด $d(1, 2, 3, ..., n) > u-l$

พิสูจน์. ใช้วิธีแบบอุปนัย (induction) ในการแสดงให้เห็นถึงการพิสูจน์ โดยกำหนดให้ *P(n)* แสดงแทนชุดข้อมูลขนาด *n* ตัว ที่มีข้อมูลแบบไม่ซ้ำกัน $(X_1, X_2, \ldots, X_n)$ *l* เป็นค่าที่น้อยที่สุด และ *u* เป็นค่าที่มากที่สุด

เนื่องจากในกรณีที่ $n = 1$ และ $n = 2$ จะเป็นจริงเสมอจึงไม่แสดงกรณีที่ $n = 1$ และ $n = 2$ จะเริ่มต้นที่ $n = 3$

ขั้นตอนพื้นฐาน, แสดงให้เห็นว่าเมื่อ $P(3)$ จะเป็นจริง โดยให้ (X_1, X_2, X_3) เป็นชุดข้อมูลแบบข้อมูลไม่ซ้ำกัน และมี 6 รูปแบบของการเรียงกันของข้อมูลดังนี้ แบบที่ 1 : $X_1 < X_2 < X_3$ แบบที่ 2 : $X_1 < X_3 < X_2$ แบบที่ 3 : $X_2 < X_1 < X_3$ แบบที่ 4 : $X_2 < X_3 < X_1$ แบบที่ 5 : $X_3 < X_1 < X_2$ และแบบที่ 6 : $X_3 < X_2 < X_1$

ในกรณีแบบที่ 1 และแบบที่ 6 จะเป็นกรณีที่ข้อมูลเรียงกันแบบจากมากไปน้อยและเรียงกันจากน้อยไปมากอยู่แล้วซึ่งเป็นจริงตามข้อพิสูจน์ที่ 1 ดังนั้นจึงแสดงการพิสูจน์ในแบบอื่น

แบบที่ 2 : $X_1 < X_3 < X_2$ จะได้ว่า $d(1, 2, 3) = |x_2 - x_1| + |x_3 - x_2| = x_2 - x_1 + x_2 - x_3 > x_2 - x_1 = u - l$.

แบบที่ 3 : $X_2 < X_1 < X_3$ จะได้ว่า $d(1, 2, 3) = |x_2 - x_1| + |x_3 - x_2| = x_1 - x_2 + x_3 - x_2 > x_3 - x_2 = u - l$.

แบบที่ 4 : $X_2 < X_3 < X_1$ จะได้ว่า $d(1, 2, 3) = |x_2 - x_1| + |x_3 - x_2| = x_1 - x_2 + x_3 - x_2 > x_1 - x_2 = u - l$.

แบบที่ 5 : $X_3 < X_1 < X_2$ จะได้ว่า $d(1, 2, 3) = |x_2 - x_1| + |x_3 - x_2| = x_2 - x_1 + x_2 - x_3 > x_2 - x_3 = u - l$.

จากทุกแบบที่แสดงจะเห็นได้ว่า $d(1, 2, 3) > u - l$.

ขั้นตอนการทำอุปนัย, กำหนดให้ *P(k)* เป็นจริง ในกรณีที่ $k > 2$ สำหรับทุกชุดข้อมูล *k* ตัวที่เป็นชุดข้อมูลแบบไม่ซ้ำกัน ที่มี *l* เป็นข้อมูลตัวที่มีค่าต่ำสุดและ *u* เป็นข้อมูลตัวที่มีค่าสุงสุด ถ้า $(X_1, X_2, \ldots, X_k)$ เป็นชุดข้อมูลที่ไม่เรียงลำดับ จะได้ $d(1,2,\ldots,k) > u\text{-}l$

จะแสดงว่า *P(k+1)* เป็นจริง กำหนดให้ $(X_1, X_2, \ldots, X_n, X_{k+1})$ เป็นชุดข้อมูลแบบไม่ซ้ำกัน มี *l* เป็นข้อมูลที่มีค่าต่ำสุด *u* เป็นข้อมูลตัวที่มีค่ามากสุด และเป็นข้อมูลแบบไม่เรียงลำดับ ในชุดข้อมูลให้ X_i คือ ข้อมูลที่ไม่ใช่ทั้งค่าที่มากที่สุดและค่าที่น้อยที่สุดแล้วเมื่อพิจารณาชุดข้อมูลที่ไม่มี X_i จะได้ว่า $(X_1, X_2, \ldots, X_{i-1}, X_{i+1}, X_{i+2}, \ldots, X_k, X_{k+1})$ ซึ่งมีข้อมูล *k* ตัว แล้วจะเกิดกรณีการเรียง 2 แบบที่เป็นไปได้ คือ แบบที่ 1 $(X_1, X_2, \ldots, X_{i-1}, X_{i+1}, X_{i+2}, \ldots, X_k)$ เป็นข้อมูลแบบเรียงลำดับ หรือแบบที่ 2 $(X_1, X_2, \ldots, X_{i-1}, X_{i+1}, X_{i+2}, \ldots, X_k)$ เป็นข้อมูลแบบไม่เรียงลำดับ

แบบที่ 1 กำหนดให้ชุดข้อมูลที่เป็นแบบเรียงลำดับมีค่า $d(1, 2, 3, \ldots, k) = u\text{-}l$ โดยที่ข้อมูลเดิมเป็นแบบยังไม่เรียงลำดับ เมื่อทำการเติม X_i กลับเข้าไปในชุดข้อมูลจะเกิดการเรียงของ X_i หลังการเติมเข้าไป 2 กรณี คือ กรณีที่ 1 $X_i < X_{i-1}$ และกรณีที่ 2 $X_i > X_{i+1}$ พิจารณา

แบบที่ 1 กรณีที่ 1 $X_i < X_{i-1}$ จะได้ว่าถ้า $X_i < X_{i-1}$, $X_i < X_{i-1} < X_{i+1}$,

$d(1, 2, 3, \ldots, k, k+1) = |x_2 - x_1|+|x_3 - x_2| + \ldots + |x_i - x_{i-1}|+|x_{i+1} - x_i|+|x_{i+2} - x_{i+1}|+ \ldots +|x_{k+1} - x_k|$

$d(1, 2, 3, \ldots, k, k+1) = -x_1 + x_{i-1} +|x_i - x_{i-1}|+|x_{i+1} - x_i|- x_{i+1} + x_{k+1}$

$d(1, 2, 3, \ldots, k, k+1) = -x_1 + x_{i-1} + x_{i-1} - x_i + x_{i+1} - x_i - x_{i+1} + x_{k+1}$

$d(1, 2, 3, \ldots, k, k+1) = -x_1 + 2(x_{i-1} - x_i) + x_{k+1}$

จากข้อสมมุติ ทราบว่า $X_1 = l$, $X_{k+1} = u$ และ $X_{i-1} - X_i > 0$ ดังนั้นจึงได้ว่า $d(1, 2, 3, \ldots, k, k+1) > u\text{-}l$

แบบที่ 1 กรณีที่ 2 $X_i > X_{i+1}$ จะได้ว่าถ้า $x_i > x_{i+1}$, $x_i > x_{i+1} > x_{i-1}$,

$d(1, 2, 3, \ldots, k, k+1) = |x_2 - x_1|+|x_3 - x_2| + \ldots + |x_i - x_{i-1}|+|x_{i+1} - x_i|+|x_{i+2} - x_{i+1}|+ \ldots +|x_{k+1} - x_k|$

$d(1, 2, 3, \ldots, k, k+1) = -x_1 + x_{i-1} +|x_i - x_{i-1}|+|x_{i+1} - x_i|- x_{i+1} + x_{k+1}$

$d(1, 2, 3, \ldots, k, k+1) = -x_1 +x_{i-1} +x_i - x_{i-1} +x_i - x_{i+1} - x_{i+1} +x_{k+1}$

$d(1, 2, 3, \ldots, k, k+1) = -x_1 + 2(x_i - x_{i+1}) + x_{k+1}$

จากข้อสมมุติ ทราบว่า $X_1 = l$, $X_{k+1} = u$ และ $X_i - X_{i+1} > 0$ ดังนั้นจึงได้ว่า $d(1, 2, 3, \ldots, k, k+1) > u\text{-}l$

แบบที่ 2 เป็นกรณีชุดข้อมูลที่ยังไม่เรียงลำดับ ได้ว่า

$d(1, 2, 3, \ldots, k) =|x_2 - x_1|+|x_3 - x_2|+ \ldots +|x_{i-1} - x_{i-2}|+|x_{i+1} - x_{i-1}|+|x_{i+2} - x_{i+1}| + \ldots +|x_{k+1} - x_k| > u - l$ and $d(1, 2, 3, \ldots, k, k+1) =|x_2 - x_1|+|x_3 - x_2| + \ldots + |x_i - x_{i-1}|+|x_{i+1} - x_i| +|x_{i+2} - x_{i+1}| + \ldots +|x_{k+1} - x_k|$.

และจาก $|x_{i+1} -x_{i-1}| \leq |x_{i+1} -x_i|+|x_i -x_{i-1}|$, จะได้ว่า $d(1, 2, \ldots, k, k+1) \geq |x_2 - x_1|+|x_3 - x_2|+ \ldots +|x_{i-1} - x_{i-2}|+|x_{i+1} - x_{i-1}|+|x_{i+2} - x_{i+1}| + \ldots +|x_{k+1} - x_k| = d(1, 2, 3, \ldots, k) > u - l$.

ดังนั้น $d(1, 2, \ldots, k, k+1) > u - l$

2.2 งานวิจัยและงานเขียนอื่น ๆ ที่เกี่ยวข้อง

ในหัวข้อนี้เป็นการนำเสนอเทคนิคต่างๆ ของการปรับปรุงประสิทธิภาพการทำงานของควิกซอร์ตและงานทางด้านผลต่างสืบเนื่อง จากงานวิจัยและงานเขียนที่น่าสนใจ ซึ่งใช้ประกอบการศึกษาในงานวิจัยนี้ โดยแบ่งเป็น 2 ส่วนหลัก คือ ส่วนของเทคนิคในการปรับปรุงประสิทธิภาพของควิกซอร์ต และส่วนของงานผลต่างสืบเนื่อง

2.2.1 งานวิจัยและงานเขียนที่เกี่ยวข้องกับการปรับปรุงประสิทธิภาพการทำงานของควิกซอร์ต

จากการศึกษาพบว่าจนถึงปัจจุบันแนวทางการปรับปรุงประสิทธิภาพของควิกซอร์ตถูกเสนอขึ้นมาหลายๆ วิธี ซึ่งแนวทางการปรับปรุงประสิทธิภาพของควิกซอร์ต สามารถจัดเป็นกลุ่มได้ 3 กลุ่ม ดังนี้

2.2.1.1 กลุ่มที่ศึกษาเพื่อหาวิธีการที่ดีในการเลือกตัวหลัก

2.2.1.2 กลุ่มที่ศึกษาขนาดข้อมูลย่อยที่เหมาะสมเพื่อเปลี่ยนไปใช้การจัดเรียงข้อมูลแบบอื่นๆ ร่วมกับควิกซอร์ต

2.2.1.3 กลุ่มที่ศึกษาการใช้เทคนิคอื่นๆ ปรับปรุงขั้นตอนวิธีของควิกซอร์ต

2.2.1.1 กลุ่มที่ศึกษาเพื่อหาวิธีการที่ดีในการเลือกตัวหลัก

กลุ่มนี้เสนอเทคนิคหรือวิธีการที่สามารถเลือกตัวหลัก เป็นการมุ่งเน้นการหาวิธีการที่เหมาะสมสำหรับการเลือกค่าตัวหลักที่ดี ทำให้การทำงานของควิกซอร์ตทำงานได้มีประสิทธิภาพดียิ่งขึ้นและแก้ปัญหาการเกิดกรณีแย่ที่สุดของควิกซอร์ตในกรณีที่นำข้อมูลที่เรียงลำดับพร้อมกับใช้การเลือกตัวหลักด้วยข้อมูลลำดับแรก งานในกลุ่มมีดังนี้

Hoare, C. A. R. [5, 6] ได้นำเสนอวิธีการเลือกตัวหลักเพิ่มเติมที่แตกต่างจากวิธีการเดิมของเขาที่เลือกตัวหลักจากข้อมูลตัวแรก โดยเสนอแนวทางในการเลือกตัวหลักไว้อีก 2 วิธี คือ

- สุ่มเลือก (Random) วิธีนี้ใช้การสุ่มเลือกข้อมูลตัวหลักจากชุดข้อมูลที่นำมาจัดเรียง โดยต้องสุ่มเลือกตัวหลักใหม่ทุกๆ รอบ
- ค่ากลางของข้อมูล (Median) วิธีนี้ต้องหาค่ากลางของข้อมูลทั้งหมด ใช้ข้อมูลค่ากลางเป็นตัวหลัก ซึ่งจะทำให้แบ่งข้อมูลเป็นสองส่วนเท่าๆ กัน ซึ่งจะทำให้ควิกซอร์ตทำงานได้อย่างมีประสิทธิภาพที่สุด

Scowen, R. S. [9] ได้เสนอแนวคิดการเลือกตัวหลักโดยใช้ข้อมูลตัวที่อยู่ตำแหน่งกลางของชุดข้อมูล (Middle element) โดยเรียกวิธีการในชื่อ Quickersort สาเหตุที่เลือกค่าหลักจากตำแหน่งกลางเพราะการหาค่ากลางของชุดข้อมูลใช้เวลามาก ค่าข้อมูลที่อยู่ตำแหน่งกลางของชุดข้อมูลน่าจะเป็นตัวแทนของค่ากลางที่ดีและในกรณีของข้อมูลเรียงลำดับ ข้อมูลตำแหน่งกลางก็เป็นค่ากลางพอดี ดังนั้นค่าข้อมูลตำแหน่งกลางน่าจะเป็นตัวหลักที่ดีและช่วยลดการเกิดปัญหากรณีแย่ที่สุดอีกด้วย

Singleton, R. C. [11] ได้เสนอแนวคิดในการเลือกตัวหลักใหม่ เพื่อให้ค่าที่ได้ใกล้เคียงค่ากลางมากขึ้น โดยใช้การเลือกค่ากลางจากข้อมูล 3 ตำแหน่ง คือ ตำแหน่งแรก ตำแหน่งกลาง และตำแหน่งท้าย แทนการเลือกจากตำแหน่งกลางเพียงตัวเดียว จากนั้นเลือก ค่ากลางจากข้อมูลทั้งสามตัวใช้เป็นค่าตัวหลัก เทคนิคนี้เป็นวิธีการที่ปรับปรุงการเลือกข้อมูลเพื่อให้ได้ตัวแทนของค่ากลางที่ดีขึ้น เนื่องจากชุดข้อมูลทั่วไปเป็นชุดข้อมูลแบบสุ่มมากกว่าแบบเรียงลำดับข้อมูลตำแหน่งกลางตัวเดียวอาจไม่ใช้ค่ากลางที่ดี ดังนั้นการเลือกข้อมูลเพิ่มอีกสองตำแหน่ง แล้วจึงใช้ค่ากลางจากข้อมูลทั้งสามค่า น่าจะได้ตัวแทนของค่ากลางที่ดีกว่าการเลือกจากตำแหน่งกลาง

Bhutoria, S. และ Konjevod, G. [2] ได้นำเสนอแนวคิดการเลือกตัวหลักจากข้อมูล 3 ตำแหน่งอีกแบบโดยเรียกว่าวิธี Greater-than/less-than วิธีการเลือกข้อมูล คือ เลือกข้อมูลตัวแรกจากนั้นทำการตรวจสอบข้อมูลตำแหน่งถัดไปจนเจอข้อมูลที่มีค่าใหญ่กว่าข้อมูล ตัวแรกเก็บเป็นค่าที่สอง จากนั้นทำการตรวจสอบข้อมูลตำแหน่งถัดไปจนเจอข้อมูลที่ค่าที่น้อยกว่าค่าข้อมูลตัวที่สอง จากนั้นเลือกค่ากลางของข้อมูลทั้งสามตัว ใช้เป็นค่าหลัก วิธีการนี้เป็นอีกรูปแบบของการหาตัวแทนของค่ากลางของชุดข้อมูลซึ่งทำให้ได้ค่ากลางที่ดีกว่าการหาตัวแทนค่ากลางจากวิธีการเลือกจากตำแหน่งแรก ตำแหน่งกลาง และตำแหน่งท้าย

2.2.1.2 **กลุ่มที่ศึกษาขนาดข้อมูลย่อยที่เหมาะสมเพื่อเปลี่ยนไปใช้การจัดเรียงข้อมูลแบบอื่น ๆ ร่วมกับควิกซอร์ต**

งานในกลุ่มนี้เป็นการศึกษาขนาดของข้อมูลย่อยที่เหมาะสม เพื่อเปลี่ยนไปใช้การจัดเรียงแบบอื่นแทนควิกซอร์ตเมื่อขนาดข้อมูลย่อยที่เกิดจากการแบ่งข้อมูลเล็กเกินไป เพราะมีการศึกษาพบว่าประสิทธิภาพของควิกซอร์ตแย่กว่าการจัดเรียงแบบแทรกหรือการจัดเรียงแบบเลือกเมื่อข้อมูลที่นำมาจัดเรียงมีขนาดเล็ก และศึกษาถึงวิธีการจัดเรียงแบบอื่นที่จะนำมาใช้แทนควิกซอร์ตเมื่อข้อมูลย่อยมีขนาดเล็กเพื่อกำจัดการเรียกซ้ำของควิกซอร์ต โดยงานวิจัยในกลุ่มมีดังนี้

Robert Sedgewick, R. [10] ได้เสนอการปรับปรุงโดยใช้การจัดเรียงข้อมูลแบบแทรก (Insertion sort) แทนเมื่อขนาดของชุดข้อมูลย่อย (Subfiles) มีขนาดเล็กกว่า 9 หรือ 10 ข้อมูล

Hossain, M. [4] ได้เสนอการปรับปรุงโดยใช้เทคนิค Merging ร่วมกับการจัดเรียงแบบควิกซอร์ต วิธีการนี้จะทำการแบ่งข้อมูลทั้งหมด กำหนดให้ขนาดข้อมูล *n* ตัว แบ่งเป็นข้อมูลย่อยจำนวน $\frac{n}{\sqrt{n}}$ ชุด ซึ่งแต่ละชุดจะมีข้อมูลขนาด $\sqrt{n}$ ข้อมูล เมื่อแบ่งข้อมูลแล้ว นำข้อมูลย่อยแต่ละชุดจัดเรียงข้อมูลด้วยการจัดเรียงแบบควิกซอร์ต เมื่อข้อมูลย่อยทุกชุดถูกจัดเรียงแล้วนำข้อมูลย่อยมาจัดเรียงแบบผสาน โดยผสานข้อมูลย่อยที่อยู่ติดกันทีละคู่ และทำการผสานกันจนรวมกันเป็นข้อมูลชุดเดียว

2.2.1.3 กลุ่มที่ศึกษาการใช้เทคนิคอื่น ๆ ปรับปรุงขั้นตอนวิธีการของควิกซอร์ต

งานในกลุ่มเป็นการปรับปรุงประสิทธิภาพการทำงานของควิกซอร์ตโดยใช้เทคนิคอื่นๆ นอกเหนือจากสองวิธีแรก เพื่อการปรับปรุงการทำงานกับข้อมูลบางลักษณะเช่น ข้อมูลที่เกือบเรียงลำดับหรือเรียงลำดับแล้ว เป็นต้น งานในกลุ่มนี้มีดังนี้

Cook, C. R. และ Kim, D. J. [3] ศึกษาหาอัลกอริทึมที่ดีที่สุดสำหรับชุดข้อมูลที่เกือบจะเรียง โดยเลือกวิธีการจัดเรียงที่นำมาทำการศึกษาดังนี้ Straight insertion sort, Shell sort, Straight merge sort, Quickersort และ Heap sort ซึ่งพบว่าการจัดเรียงที่ดีกับข้อมูลลักษณะเกือบเรียงลำดับมีด้วยกัน 2 วิธี คือ Straight insertion sort และ Quickersort โดยที่ Straight insertion sort ทำงานได้ดีกับข้อมูลขนาดเล็ก ส่วน Quickersort ทำงานได้ดีกับข้อมูลขนาดใหญ่ Cook และ Kim เสนอแนวคิดอัลกอริทึมใหม่ที่รวมการทำงานของ Straight insertion sort, Quickersort และ Merge sort เข้าด้วยกัน โดยวิธีการนี้จะทำการแยกข้อมูลโดยพิจารณา ทีละคู่ลำดับเพื่อแบ่งเป็น 2 ส่วน ส่วนแรกเป็นชุดข้อมูลคู่ลำดับติดกันที่มีลำดับถูกต้อง ส่วนที่สองเป็นชุดข้อมูลคู่ลำดับติดกันแต่มีลำดับไม่ถูกต้อง ตัวอย่าง ข้อมูลขนาด 10 ข้อมูล มีค่าดังนี้ 5, 0, 2, 4, 3, 7, 9, 1, 6, 8 เมื่อทำการแยกข้อมูลจะได้ชุดข้อมูล 2 ชุดดังนี้ ชุดแรกข้อมูลที่ลำดับถูกต้อง คือ 2, 4, 3, 7, 6, 8 ชุดที่สองข้อมูลที่ลำดับไม่ถูกต้อง คือ 5, 0, 9, 1 เมื่อได้ชุดข้อมูลทั้ง 2 ชุดแล้วถ้าขนาดของชุดข้อมูลใดเล็กกว่า 30 ข้อมูลให้จัดเรียงด้วย Straight insertion sort ส่วนชุดข้อมูลที่ขนาดใหญ่กว่า 30 ข้อมูลให้จัดเรียงด้วย Quickersort เมื่อชุดข้อมูลทั้ง 2 ชุดจัดเรียงเสร็จ นำมาผสานรวมกัน

Wainwright, R. L. [12] เสนอการปรับปรุงควิกซอร์ตโดยใช้เทคนิคการตรวจสอบแบบเปรียบเทียบ (if) กับตัวบ่งชี้ (flag) ร่วมกับการสลับข้อมูลแบบการจัดเรียงแบบบับเบิ้ล โดยเรียกวิธีการนี้ว่า Bsort เพื่อปรับปรุงให้การทำงานของควิกซอร์ตทำงานได้ดีขึ้นกับข้อมูลที่เกือบเรียงลำดับและหยุดการทำงานได้เมื่อชุดข้อมูลเรียงลำดับ โดยทำการตั้งค่าตัวบ่งชี้เริ่มต้นเป็นเท็จ ในระหว่างขั้นตอนการแบ่งข้อมูลให้ทำการตรวจสอบลำดับข้อมูลโดยการเปรียบเทียบกับข้อมูลข้างเคียง ถ้าพบข้อมูลที่ลำดับไม่ถูกต้องให้ทำการสลับค่าคู่ลำดับให้ถูกต้องและเปลี่ยนค่าตัวบ่งชี้ให้เป็นจริง ทำการตรวจสอบต่อไปจนกระทั่งเสร็จสิ้นขั้นตอนการแบ่งข้อมูล ให้ทำการตรวจสอบตัวบ่งชี้ ถ้าตัวบ่งชี้เป็นเท็จแสดงว่าข้อมูลเรียงลำดับถูกต้องให้หยุดการจัดเรียงของข้อมูล แต่ถ้าตัวบ่งชี้เป็นจริงแสดงว่าข้อมูลยังไม่เรียงลำดับจึงทำการจัดเรียงข้อมูลด้วยควิกซอร์ตกับข้อมูลย่อยต่อไป

Wainwright, R. L. [13] เสนอการปรับปรุง Bsort ให้มีประสิทธิภาพดีขึ้นโดยตัดขั้นตอนการสลับข้อมูลและปรับเปลี่ยนวิธีการตรวจสอบตัวบ่งชี้ โดยให้ตั้งค่าตัวบ่งชี้เริ่มต้นเป็นจริง ในขั้นตอนการแบ่งข้อมูลทำการตรวจสอบลำดับข้อมูลและตรวจสอบตัวบ่งชี้ ถ้าพบว่ามีข้อมูลไม่เรียงลำดับให้ทำการเปลี่ยนค่าตัวบ่งชี้เป็นเท็จ และเมื่อตัวบ่งชี้เป็นเท็จในระหว่างการแบ่งข้อมูลจะทำให้การตรวจสอบลำดับข้อมูลถูกยกเลิกออกไปในรอบการแบ่งข้อมูลนั้น และเริ่มตรวจสอบใหม่ในรอบการทำงานใหม่ เมื่อเสร็จสิ้นขั้นตอนการแบ่งข้อมูลตรวจสอบค่าตัวบ่งชี้ ถ้าค่าตัวบ่งชี้เป็นจริงแสดงว่าข้อมูลเรียงลำดับถูกต้องให้หยุดทำการจัดเรียงกับข้อมูลชุดนั้น แต่ถ้าค่าตัวบ่งชี้เป็นเท็จแสดงว่าข้อมูลยังไม่เรียงลำดับให้ทำการจัดเรียงต่อไป

2.2.2 งานวิจัยและงานเขียนที่เกี่ยวข้องกับวิธีการผลต่างสืบเนื่อง

ในหัวข้อนี้กล่าวถึงงานที่เกี่ยวข้องกับวิธีการผลต่างสืบเนื่อง งานในกลุ่มนี้มีดังต่อไปนี้

Sinapiromsaran, K. [7, 8] ได้เสนอวิธีการพิสูจน์สมบัติที่อยู่ในชุดข้อมูลจำนวนที่เรียงลำดับ โดยเรียกว่า ผลต่างสืบเนื่อง ซึ่งหมายถึงสมบัติในชุดข้อมูลจำนวนที่ได้จากผลบวกสะสมของผลลบระหว่างคู่ลำดับติดกันแบบไม่สนใจเครื่องหมาย ซึ่งค่าผลต่างสืบเนื่องในชุดข้อมูลตัวเลขที่เรียงลำดับต้องมีค่าเท่ากับผลลบระหว่างค่าสูงสุดกับค่าต่ำสุดในชุดข้อมูลนั้น จากสมบัติของผลต่างสืบเนื่องทำให้สามารถตรวจสอบการเรียงลำดับในชุดข้อมูลได้ โดยใช้ค่าผลต่างสืบเนื่องเปรียบเทียบกับผลต่างระหว่างค่าสูงสุดกับค่าต่ำสุด นอกจากนั้นในงานวิจัยได้ทำการประยุกต์วิธีการผลต่างสืบเนื่องเข้ากับการจัดเรียงแบบเลือก (Selection sort) โดยขั้นแรกจะทำการคำนวณค่าผลต่างสืบเนื่องของชุดข้อมูล จากนั้นในระหว่างการทำงานของการจัดเรียงแบบเลือก

ถ้ามีการสลับที่ข้อมูลให้ทำการปรับเปลี่ยนค่าผลต่างสืบเนื่อง และทำการตรวจสอบค่าของผลต่างสืบเนื่องทุกรอบการจัดเรียง ถ้าผลต่างสืบเนื่องมีค่าเท่ากับผลต่างระหว่างข้อมูลที่มีค่าสูงสุดกับข้อมูลที่มีค่าต่ำสุดแล้ว ให้หยุดการทำงานของการเรียงข้อมูลแบบแทรก