การดำเนินการเชื่อมความสัมพันธ์สำหรับฐานข้อมูลเชิงพื้นที่โดยใช้เทคนิคอาร์-ทรี บนหน่วยประมวลผลกราฟิกส์

นางสาวต้องใจ แย้มผกา

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2554

SPATIAL JOIN OPERATION FOR SPATIAL DATABASE WITH R-TREE TECHNIQUE ON GRAPHICS PROCESSING UNIT

Miss Tongjai Yampaka

A Thesis Submitted in Partial Fulfillment of the Requirements

for the Degree of Master of Science Program in Computer Science

Department of Computer Engineering

Faculty of Engineering

Chulalongkorn University

Academic Year 2011

หัวข้อวิทยานิพนธ์	การดำเนินการเชื่อมความสัมพันธ์สำหรับฐานข้อมูลเชิงพื้นที่โดยใช้เทคนิคอาร์-ทรีบนหน่วยประมวลผลกราฟิกส์
โดย	นางสาวต้องใจ แย้มผกา
สาขาวิชา	วิทยาศาสตร์คอมพิวเตอร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	ศาสตราจารย์ ดร.ประภาส จงสถิตย์วัฒนา

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

…….. คณบดีคณะวิศวกรรมศาสตร์
(รองศาสตราจารย์ ดร.บุญสม เลิศหิรัญวงศ์)

คณะกรรมการสอบวิทยานิพนธ์

……. ประธานกรรมการ
(ผู้ช่วยศาสตราจารย์ ดร.สุกรี สินธุภิญโญ)

…….. อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก
(ศาสตราจารย์ ดร.ประภาส จงสถิตย์วัฒนา)

……. กรรมการภายนอกมหาวิทยาลัย
(ผู้ช่วยศาสตราจารย์ ดร.วรเศรษฐ สุวรรณิก)

ต้องใจ แย้มผกา : การดำเนินการเชื่อมความสัมพันธ์สำหรับฐานข้อมูลเชิงพื้นที่โดยใช้เทคนิคอาร์-ทรีบนหน่วยประมวลผลกราฟิกส์. (SPATIAL JOIN OPERATION FOR SPATIAL DATABASE WITH R-TREE TECHNIQUE ON GRAPHICS PROCESSING UNIT) อ.ที่ปรึกษาวิทยานิพนธ์หลัก: ศ.ดร. ประภาส จงสถิตย์วัฒนา, 45 หน้า.

การดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุ (Spatial Join) เป็นการดำเนินการสำหรับฐานข้อมูลเชิงพื้นที่ซึ่งจะดำเนินการเชื่อมความสัมพันธ์ของวัตถุสองวัตถุตามเงื่อนไขที่กำหนดใช้ระยะเวลาในการประมวลผลค่อนข้างมากเนื่องด้วยข้อมูลที่เก็บในฐานข้อมูลเชิงพื้นที่นั้นมีความซับซ้อนแตกต่างจากการดำเนินการ ของฐานข้อมูลเชิงสัมพันธ์ทั่วไป งานวิจัยในด้านการดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุที่ผ่านมาได้มุ่งที่จะพัฒนาความเร็วในการประมวลผลดังกล่าวซึ่งวิธีการหนึ่งที่พบว่าสามารถเพิ่มความเร็วในการประมวลผลได้คือวิธีการประมวลผลแบบขนาน กระบวนการประมวลผลแบบขนานนั้นจะถูกใช้ในส่วนของการเปรียบเทียบพิกัดจุดของวัตถุระหว่างสองวัตถุ และด้วยสาเหตุที่ข้อมูลของฐานข้อมูลเชิงพื้นที่นั้นมีจำนวนข้อมูลจำนวนมาก เช่น พิกัดจุดที่ประกอบขึ้นมาเป็นวัตถุที่มีลักษณะเป็นหลายมิติงานวิจัยนี้จึงได้นำเทคนิคของอาร์-ทรีเข้ามาช่วยในการเข้าถึงข้อมูลได้เร็วขึ้น

งานวิจัยนี้เสนอการดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุด้วยวิธีการแบบขนานบนหน่วยประมวลผลกราฟิกเพื่อเพิ่มความเร็วในการประมวลผลในส่วนของการเปรียบเทียบพิกัดจุดระหว่างวัตถุ และเพิ่มความเร็วในการเข้าถึงข้อมูลด้วยการทำดัชนีโดยใช้เทคนิคอาร์-ทรี จากผลการทดลองพบว่าสามารถเพิ่มความเร็วในการประมวลผลการดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุสำหรับฐานข้อมูลเชิงพื้นที่ได้เพิ่มขึ้นเป็นสองเท่าเมื่อเปรียบเทียบกับหน่วยประมวลผลกลางที่ใช้ข้อมูลและวิธีการเดียวกัน

ภาควิชาวิศวกรรมคอมพิวเตอร์ ลายมือชื่อนิสิต .....

สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ ลายมือชื่อ อ.ที่ปรึกษาวิทยานิพนธ์หลัก .....

ปีการศึกษา 2554 .....

5371463221 : MAJOR COMPUTER SCIENCE

KEYWORDS : SPATIAL JOIN OPERATION / SPATIAL JOIN WITH R-TREE / GRAPHICS PROCESSING UNIT

TONGJAI YAMPAKA : SPATIAL JOIN OPERATION FOR SPATIAL DATABASE WITH R-TREE TECHNIQUE ON GRAPHICS PROCESSING UNIT.

ADVISOR: PROF PRABHAS CHONGSTITVATANA, 45 pp.

Spatial operations such as spatial join combine two objects on spatial predicates. It is different from relational join because objects have multi dimensions and spatial join consumes large execution time. Recently, many works investigate methods to improve the execution time. Parallel spatial join is one of the methods. Comparison between objects can be done in parallel. Because spatial datasets are large, R-Tree data structure is used improve the performance of the access to data.

In this paper, we design a parallel spatial join on Graphics processing unit (GPU). We use GPU which has many processors to accelerate the computation. The experiment is carried out to compare the spatial join between a sequential implementation with C language on CPU and a parallel implementation with CUDA C language on GPU. The result shows that the spatial join on GPU is faster than on a conventional processor.

Department : Computer Engineering Student's Signature .....

Field of Study : Computer Science Advisor's Signature .....

Academic Year : 2011

กิตติกรรมประกาศ

วิทยานิพนธ์นี้ดำเนินการได้สำเร็จตรงตามวัตถุประสงค์เนื่องด้วยได้รับคำแนะนำและได้รับการสนับสนุนทั้งทางด้านการให้ความรู้เชิงวิชาการ และให้คำแนะนำด้านอื่น ๆ จากการให้คำปรึกษาของ ศ.ดร.ประภาส จงสถิตย์วัฒนาและ ผศ. ดร.สุกรี สินธุภิญโญ รวมถึงกรรมการผู้สอบจากภายนอกได้แก่ ผศ.ดร.วรเศรษฐ สุวรรณิก ที่ได้ให้คำแนะนำและข้อเสนอแนะในการทำวิทยานิพนธ์ฉบับนี้

นอกจากคณาจารย์ที่ให้คำแนะนำแล้วยังมีผู้ที่เกี่ยวข้องอันได้แก่เจ้าหน้าที่ทุกท่านในภาควิชาวิศวกรรมคอมพิวเตอร์ที่ได้ช่วยให้คำแนะนำในส่วนของขั้นตอนทั้งการจัดเตรียมการสอบและการยื่นเอกสารที่เกี่ยวข้อง และขอขอบคุณเพื่อนที่ทำงานร่วมกันในกลุ่มที่ให้การช่วยเหลือในส่วนของการติดตามงาน และเอกสารสำคัญที่ต้องดำเนินการ และมีส่วนกระตุ้นให้งานสำเร็จลุล่วงตามระยะเวลา

สารบัญ

หน้า

สารบัญตาราง

หน้า

สารบัญภาพ

หน้า

บทที่ 1
บทนำ

ความเป็นมาและความสำคัญของปัญหา

ฐานข้อมูลเชิงพื้นที่ (Spatial Database) คือ ฐานข้อมูลใช้สำหรับจัดเก็บและมีการจัดการข้อมูลในเชิงความเป็นพื้นที่ เช่น ระบบสารสนเทศ ภูมิศาสตร์ (Graphic Information System: GIS) หรือ ซอฟท์แวร์ที่ใช้คอมพิวเตอร์ช่วยในการออกแบบผลิตภัณฑ์ (Computer-aided design: CAD) เป็นต้น รูปแบบของฐานข้อมูลเชิงพื้นที่นั้นจะบรรยายความเป็นรูปร่างของวัตถุที่แสดงออกมาในรูปต่าง ๆ กัน เช่น ในการบรรยายแผนที่จะใช้รูปหลายเหลี่ยม (Polygon) แทนเมือง ใช้เส้น (Line) แทนแม่น้ำหรือถนน เป็นต้น ซึ่งโดยหลักในการประมวลผลภาพออกมาแสดงได้นั้น ภาพดังกล่าวจะต้องประกอบไปด้วยความสัมพันธ์ของพิกัดจุดบนแกน X แกน Y หรือ แกน Z ขึ้นอยู่กับมิติของภาพและพิกัดความสัมพันธ์ของวัตถุซึ่งวัตถุหนึ่งนั้นไม่ได้มีเพียงความสัมพันธ์ของพิกัดจุดเพียงจุดเดียวจะประกอบขึ้นมาจากความสัมพันธ์ของพิกัดจุดหลาย ๆ พิกัดเพื่อแสดงออกมาเป็นรูปร่าง ด้วยสาเหตุที่ลักษณะของฐานข้อมูลเชิงพื้นที่มีลักษณะที่ต่างจากฐานข้อมูลเชิงสัมพันธ์ทั่วไปการดำเนินการกับข้อมูลจึงมีความซับซ้อนมากกว่าฐานข้อมูลเชิงสัมพันธ์ทั่วไป

ด้วยเหตุผลที่ลักษณะข้อมูลของฐานข้อมูลเชิงพื้นที่ที่มีความซับซ้อนและต่างจากฐานข้อมูลเชิงสัมพันธ์ทั่วไป การดำเนินการของฐานข้อมูลเชิงพื้นที่จึงต้องใช้เทคนิคเฉพาะในการสืบค้นข้อมูล เช่น การดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุสำหรับฐานข้อมูลเชิงพื้นที่ (Spatial Join) ก็เช่นเดียวกัน การดำเนินการนั้นจะดำเนินการรวมวัตถุสองวัตถุเข้าด้วยกันตามเงื่อนไขที่กำหนด เช่น การค้นหาว่ามีเมืองใดที่มีพื้นที่ทับซ้อนกันกับแม่น้ำบ้าง ในฐานข้อมูลเชิงพื้นที่นั้นจะเก็บข้อมูลเฉพาะพิกัดจุดของเมืองแต่ละเมืองและแม่น้ำแต่ละสายว่ามีพิกัดจุดใดบ้าง ไม่ได้เก็บเขตข้อมูลที่แสดงการเชื่อมโยงกันระหว่างข้อมูลสองชุดนี้ ดังนั้นในการหาจึงต้องดำเนินการอ่านข้อมูลพิกัดจุดของข้อมูลทั้งสองชุดนี้มีดำเนินการเปรียบเทียบกันว่ามีพิกัดจุดใดบ้างที่ตรงตามเงื่อนไขที่ต้องการซึ่งแต่ละวัตถุก็มีกลุ่มของพิกัดจุดที่มากกว่าหนึ่งจุดดังนั้นในการนำพิกัดจุดดังกล่าวมาเปรียบเทียบหาความสัมพันธ์ที่ตรงเงื่อนไขจึงมีผลกระทบต่อเวลาในการประมวลผลที่เพิ่มมากขึ้นด้วย

นอกจากปัญหาความซับซ้อนของข้อมูลที่เก็บแล้วการเข้าถึงข้อมูลของฐานข้อมูลเชิงพื้นที่ก็เป็นอีกปัญหาหนึ่งเช่นกันเนื่องจากข้อมูลที่เก็บเป็นชุดของข้อมูลที่มีปริมาณมากและมีขนาดของฐานข้อมูลที่ใหญ่ เพื่อแก้ไขปัญหาดังกล่าวงานวิจัยนี้จึงได้มีการนำเทคนิคอาร์-ทรีมาช่วยในการเข้าถึงข้อมูล ซึ่งเทคนิคดังกล่าวเป็นการทำดัชนีข้อมูลเพื่อใช้ในการเข้าถึงข้อมูลได้อย่างรวดเร็ว

อาร์-ทรีมีลักษณะเหมือนบี-ทรีจะต่างกันที่อาร์-ทรีเป็นการทำดัชนีข้อมูลที่มีลักษณะข้อมูลที่เป็นหลายมิติเนื่องจากวิธีนี้สามารถตัดเส้นทางที่ไม่นำไปสู่คำตอบออกกับกรณีที่วัตถุใดที่มีพิกัดที่ไม่มีโอกาสที่จะตรงตามเงื่อนไขออก และจะประมวลผลเฉพาะเส้นทางที่จะนำไปสู่คำตอบเท่านั้นส่งผลให้ลดระยะเวลาในการประมวลผลได้ ดังเช่นงานวิจัย [1] ได้นำเทคนิคอาร์-ทรีไปใช้กับการดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุของฐานข้อมูลเชิงพื้นที่พบว่าสามารถเพิ่มความเร็วในการประมวลผลได้ อย่างไรก็ตามในงานวิจัยดังกล่าวได้ออกแบบวิธีการและทดสอบในการทำงานแบบลำดับเท่านั้น

ที่ผ่านมามีงานวิจัยเพื่อเพิ่มความเร็วในการประมวลผลไว้หลายงานวิจัยด้วยกันโดยมีงานเกี่ยวกับการสำรวจเทคนิคของการดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุสำหรับฐานข้อมูลเชิงพื้นที่ [2] งานวิจัยเหล่านี้ได้แสดงถึงเทคนิคและวิธีการที่จะช่วยเพิ่มความเร็วในการประมวลผล ซึ่งเทคนิคในการประมวลผลแบบขนานก็เป็นเทคนิคหนึ่งที่พบว่าสามารถเพิ่มความเร็วในการประมวลผลสำหรับการดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุของฐานข้อมูลเชิงพื้นที่ได้

สำหรับงานวิจัยนี้ได้นำการประมวลผลแบบขนานมาช่วยเพิ่มความเร็วในการประมวลผลโดยใช้หน่วยประมวลผลกราฟิก (Graphics Processing Unit: GPU) เนื่องด้วยหน่วยประมวลผลกราฟิกสามารถรองรับการทำงานแบบขนานและด้วยวิวัฒนาการของหน่วยประมวลผลกราฟิกที่พัฒนาให้สามารถในการประมวลผลเร็วขึ้นสาเหตุเนื่องจากความต้องการงานในด้านการประมวลผลภาพที่ต้องการความสมจริง ความเร็วในการแสดงภาพ เช่น งานในด้านเกมส์คอมพิวเตอร์ งานแสดงภาพสามมิติ และงานแอนนิเมชัน เป็นต้น ทำให้การพัฒนาของหน่วยประมวลผลกราฟิกมีความสามารถและมีคุณสมบัติให้รองรับการประมวลผลดังกล่าวให้มีประสิทธิภาพมากขึ้น [3] การทำงานของหน่วยประมวลผลกราฟิกจะเป็นแบบ Single Instruction Multiple Data (SIMD) กล่าวคือหนึ่งชุดคำสั่งสามารถประมวลผลได้กับข้อมูลที่ต่างกันในเวลาเดียวกันแบบขนาน ส่งผลให้สามารถประมวลผลได้รวดเร็วขึ้น นอกจากการประมวลผลแบบขนานแล้วหน่วยประมวลผลกราฟิกยังสามารถใช้ในการประมวลผลคำสั่งอื่นนอกเหนือจากงานในด้านกราฟิกเพียงอย่างเดียวเพื่อเปิดโอกาสให้นักพัฒนาได้ใช้คุณสมบัติของหน่วยประมวลผลกราฟิกในการแก้ปัญหาอื่น ๆ เช่น งานวิจัยทางวิทยาศาสตร์หรือคณิตศาสตร์ที่ต้องการความรวดเร็วในการประมวลผลกับข้อมูลจำนวนมาก

งานวิจัยนี้มีแนวคิดที่จะเพิ่มความรวดเร็วในการประมวลผลการเชื่อมความสัมพันธ์ระหว่างวัตถุสำหรับฐานข้อมูลเชิงพื้นที่ในเงื่อนไขที่หาพิกัดของวัตถุสองวัตถุที่มีการทับซ้อนกัน (Intersection Join) เนื่องด้วยเงื่อนไขนี้จะต้องทำการเปรียบเทียบกับทุกพิกัดจุดของทั้งสองวัตถุจึงมีความเกี่ยวข้องกับข้อมูลพิกัดจุดจำนวนมากจึงได้นำเทคนิคในการประมวลผลแบบขนานเพื่อ

เพิ่มความรวดเร็วในการประมวลผล และการทำดัชนีข้อมูลโดยใช้เทคนิคอาร์-ทรีเพื่อลดระยะเวลาในการเข้าถึงข้อมูล

ขั้นตอนในการดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุสำหรับฐานข้อมูลเชิงพื้นที่นั้นได้แบ่งออกเป็น 2 ขั้นตอนดังนี้ [4]

1) Filtering step เป็นการกรองขอบเขตของวัตถุที่สนใจโดยใช้เทคนิคกรอบสี่เหลี่ยมเล็กสุดที่สามารถครอบคลุมวัตถุ (Minimum Bounding Rectangles: MBRs) เพื่อเป็นการจำกัดขอบเขตของวัตถุที่สนใจ

2) Refinement step เป็นการดึงข้อมูลของวัตถุข้อมูลที่ได้เป็นข้อมูลจริงที่อ่านมาจากหน่วยความจำที่เก็บข้อมูลและนำมาประมวลผลตามเงื่อนไขที่ต้องการ

หลายงานวิจัยที่ผ่านมาสนใจในการเพิ่มความเร็วในการประมวลผลในขั้นตอนกรองขอบเขตของวัตถุที่สนใจเท่านั้นมีข้อสมมติฐานว่าได้มีการคำนวณและเตรียมข้อมูลในส่วนของกรอบสี่เหลี่ยมเล็กสุดที่สามารถครอบคลุมวัตถุไว้แล้วและนำข้อมูลดังกล่าวมาดำเนินการในส่วนของการดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุได้ทันที ในขั้นตอนการสร้างอาร์-ทรีเพื่อเป็นดัชนีใช้ในการเข้าถึงข้อมูลจะมีการเตรียมข้อมูลดังกล่าวบนหน่วยประมวลผลกลางก่อนที่จะส่งข้อมูลไปยังหน่วยประมวลผลกราฟิกเพื่อดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุแบบขนาน

เพื่อเปรียบเทียบความเร็วในการประมวลผลระหว่างแบบลำดับกับแบบขนานงานวิจัยนี้ได้เขียนคำสั่งทดสอบเวลาทั้งสองแบบเพื่อทำการเปรียบเทียบเวลาในการประมวลผลภายใต้วิธีการชุดข้อมูลและเงื่อนไขเดียวกัน โดยในการประมวลผลแบบลำดับใช้ภาษาซีในการเขียนคำสั่งและใช้ CUDA C ในการเขียนคำสั่งแบบขนาน โดยทั้งสองวิธีเสียเวลาในการเตรียมข้อมูลก่อนที่จะประมวลผลการดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุภายใต้เงื่อนไขเดียวกัน แต่ความแตกต่างอยู่ที่ขั้นตอนในการประมวลผลการดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุบนหน่วยประมวลผลกลางเป็นแบบลำดับ แต่บนหน่วยประมวลผลกราฟิกเป็นแบบขนานเปรียบเทียบเวลาในการประมวลผลระหว่างแบบลำดับและแบบขนาน

วัตถุประสงค์ของงานวิจัย

วัตถุประสงค์ของงานวิจัยเพื่อเพิ่มความเร็วในการประมวลผลการดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุสำหรับฐานข้อมูลเชิงพื้นที่ในขั้นตอนการกรองขอบเขตของวัตถุที่สนใจในการเพิ่มความเร็วในการประมวลผลนั้นแบ่งปัญหาออกเป็นสองส่วนคือการประมวลผลการเชื่อมความสัมพันธ์ระหว่างวัตถุกับการเข้าถึงข้อมูล เพื่อเพิ่มความเร็วในขั้นตอนการเชื่อมความสัมพันธ์ระหว่างวัตถุจะใช้เทคนิคการประมวลผลแบบขนานเนื่องจากในการเปรียบเทียบวัตถุที่ตรงตาม

เงื่อนไขเกี่ยวข้องมีการเปรียบเทียบข้อมูลพิกัดจุดของวัตถุจำนวนมากโดยเขียนคำสั่งให้ประมวลผลแบบขนานบนหน่วยประมวลผลกราฟิกซึ่งสามารถรองรับการทำงานแบบขนานได้ งานวิจัยที่ผ่านมา [5] ได้นำเทคนิคการประมวลผลแบบขนานในการค้นหาข้อมูลพิกัดจุดของฐานข้อมูลเชิงพื้นที่จากพิกัดของกรอบที่ต้องการว่ามีวัตถุใดที่ตรงตามพิกัดกรอบที่กำหนด (Windows query) ในการค้นหาจะดำเนินการค้นหาแบบขนานไปพร้อม ๆ กันกับหลาย ๆ วัตถุในเวลาเดียวกันผลลัพธ์คือเพิ่มความเร็วในการค้นหาได้ ในการเพิ่มความเร็วในการเข้าถึงข้อมูลจะดำเนินการเปรียบเทียบจากพิกัดจุดของกรอบสี่เหลี่ยมที่คลุมพื้นที่ว่ามีพิกัดที่มีการทับซ้อนกันหรือไม่ซึ่งวิธีการที่ไม่ซับซ้อนคืออ่านข้อมูลมาแล้วนำวัตถุทั้งหมดมาเปรียบเทียบกันทีละวัตถุวิธีการดังกล่าว เหมาะกับจำนวนวัตถุที่น้อยหากจำนวนวัตถุที่ต้องนำมาเปรียบเทียบนั้นมีจำนวนมากวิธีการนี้จะส่งผลต่อความเร็วในการประมวลผลเพื่อแก้ปัญหานี้จึงได้นำเทคนิคของอาร์-ทรีมาช่วยในการทำดัชนีในการเข้าถึงวัตถุก่อนที่จะนำวัตถุนั้นมาเปรียบเทียบกันเพื่อที่จะลดจำนวนครั้งในการประมวลผลได้

นอกจากการเพิ่มความเร็วในการประมวลผลการดำเนินการเชื่อมความสัมพันธ์สำหรับฐานข้อมูลเชิงพื้นที่แล้วนั้นวัตถุประสงค์อื่นของงานวิจัยนี้คือศึกษาถึงโครงสร้างกระบวนการในการประมวลผลและแนวคิดที่จะต้องเปลี่ยนการประมวลผลแบบลำดับไปเป็นแบบขนานเพื่อที่จะใช้คุณสมบัติของหน่วยประมวลผลกราฟิกได้อย่างเต็มประสิทธิภาพและเพิ่มความเร็วในการประมวลผล การเขียนคำสั่งให้หน่วยประมวลผลกราฟิกนั้นต่างจากการเขียนคำสั่งทั่วไปรูปแบบคำสั่งมีความซับซ้อนการติดต่อกับหน่วยประมวลผลนั้นต้องดำเนินการผ่านโปรแกรม Application Programming Interfaces (APIs) ในงานวิจัยนี้นำ CUDA C [6] มาเขียนคำสั่งในการประมวลผลเนื่องจาก CUDA C มีชุดคำสั่งสำเร็จภาพที่ผู้พัฒนาสามารถเรียกใช้เพื่อติดต่อกับหน่วยประมวลผลกราฟิกได้ง่ายโดยไม่จำเป็นต้องมีความรู้เฉพาะทางในส่วนของการเขียนคำสั่งและโครงสร้างการทำงานของหน่วยประมวลผลมากนัก

ขอบเขตของการวิจัย

1. งานวิจัยนี้ออกแบบแนวทางเพิ่มความเร็วในการประมวลผลในส่วนการดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุบนหน่วยประมวลผลกราฟิก
2. ข้อมูลกรอบสี่เหลี่ยมเล็กที่สุดของวัตถุจะถูกเตรียมไว้ก่อนที่จะนำไปดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุ
3. กระบวนการในการสร้างต้นไม้จะถูกเตรียมบนหน่วยประมวลผลกลางก่อนที่จะส่งไปประมวลผลที่หน่วยประมวลผลกราฟิก

4. การทดลองบนหน่วยประมวลผลกลางแบบลำดับจะใช้ภาษาซีในการเขียนโปรแกรมทดสอบ และการทดลองบนหน่วยประมวลผลกราฟิกจะใช้ CUDA C ในการเขียนโปรแกรมทดสอบ

5. เปรียบเทียบการประมวลผลบนหน่วยประมวลผลกลางและหน่วยประมวลผลกราฟิกด้วยแนวคิดและวิธีเดียวกัน

ประโยชน์ที่คาดว่าจะได้รับ

สามารถประยุกต์ใช้คุณสมบัติของหน่วยประมวลผลกราฟิกมาใช้การดำเนินการกับฐานข้อมูลเชิงพื้นที่ในส่วนของการประมวลผลคำสั่งเชื่อมความสัมพันธ์ระหว่างวัตถุ (Spatial Join) เพื่อหาวัตถุที่มีการทับซ้อนกัน (Intersection join) และเปรียบเทียบความเร็วในการประมวลผล ระหว่างหน่วยประมวลผลกลางและหน่วยประมวลผลกราฟิกโดยใช้เทคนิคอาร์-ทรี เพื่อเป็นแนวทางทำโปรแกรมประยุกต์ที่เกี่ยวข้องกับการประมวลผลฐานข้อมูลเชิงพื้นที่

ขั้นตอนการดำเนินการวิจัย

1. ศึกษาทฤษฎีที่เกี่ยวข้องกับฐานข้อมูลเชิงพื้นที่และหลักการประมวลผลหลักการดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุของฐานข้อมูลเชิงพื้นที่เพื่อหาวัตถุที่มีการทับซ้อนกันโดยใช้เทคนิคอาร์-ทรี

2. ศึกษาโครงสร้างการทำงานของหน่วยประมวลผลกราฟิก และการเขียนคำสั่งบนหน่วยประมวลผลกราฟิกด้วย CUDA C

3. ออกแบบโครงสร้างข้อมูล, ขั้นตอนวิธีที่จะใช้ในการประมวลผล และออกแบบวิธีการทดลอง

4. เขียนคำสั่งประมวลผลโดยใช้ภาษาซีในการทดลองบนหน่วยประมวลผลกลาง และ CUDA C ในการทดลองบนหน่วยประมวลผลกราฟิก

5. เปรียบเทียบความเร็วในการประมวลผลระหว่างหน่วยประมวลผลกลางและหน่วยประมวลผลกราฟิก

6. วิเคราะห์ผลการทดลอง

7. สรุปผลและเรียบเรียงวิทยานิพนธ์

โครงสร้างของวิทยานิพนธ์

โครงสร้างของวิทยานิพนธ์ฉบับนี้ประกอบด้วย 5 บทหลักดังนี้ บทนำกล่าวถึงความเป็นมาและความสำคัญของปัญหาที่นำไปสู่งานวิจัย ทฤษฎีและงานวิจัยที่เกี่ยวข้องกล่าวถึงทฤษฎีที่ใช้สนับสนุนงานวิจัยนี้และการทบทวนวรรณกรรมสำหรับงานวิจัยที่ผ่านมา ระเบียบขั้นตอนวิธีที่

นำเสนอกล่าวถึงกระบวนการวิธีคิดและขั้นตอนวิธีการทดลองของ การทดลองและผลการทดลอง แสดงการทดลองและผลลัพธ์ที่ได้ และบทสุดท้ายเป็นบทสรุปของงานวิจัยและแนวทางการพัฒนา

ผลงานที่ตีพิมพ์จากวิทยานิพนธ์

ส่วนหนึ่งของวิทยานิพนธ์ฉบับนี้ได้รับการตอบรับให้ตีพิมพ์เป็นบทความทางวิชาการในหัวข้อเรื่อง “Spatial Join with R-tree on Graphic Processing Unit” โดย ต้องใจ แย้มผกา และประภาส จงสถิตย์วัฒนา ในงานประชุมวิชาการของ “The 8th International Conference on Computing and Information Technology” ที่ประเทศไทยระหว่างวันที่ 9-10 พฤษภาคม 2555

บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

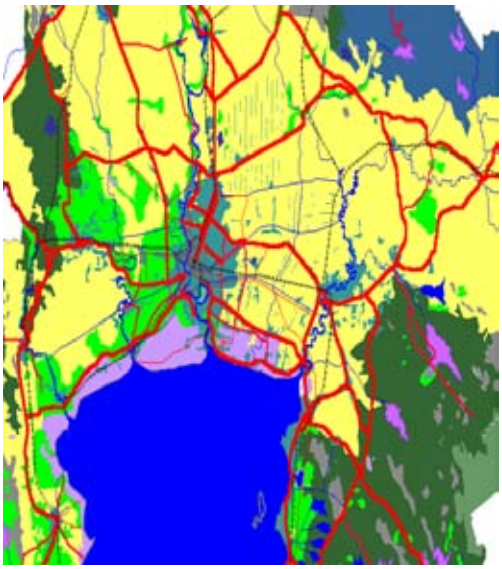
2.1 ทฤษฎีที่เกี่ยวข้อง

2.1.1 ลักษณะของฐานข้อมูลเชิงพื้นที่

ฐานข้อมูลเชิงพื้นที่ คือ ฐานข้อมูลที่เก็บรายละเอียดของวัตถุ และบอกถึงลักษณะของพื้นที่ ตำแหน่ง หรือ รูปร่างของวัตถุใด ๆ เช่น ฐานข้อมูลที่เก็บข้อมูลแล้วแสดงออกมาในรูปของแผนที่ การจัดเก็บข้อมูลแบ่งออกเป็น 2 ประเภท คือ ข้อมูลที่เป็นเชิงบรรยาย (Non-spatial data) กับข้อมูลเชิงพื้นที่ (Spatial data)

Road_id	1025
Road_type	1
Road_name	สุวินทวงศ์ [304]
From_to	กทม. - จ.ฉะเชิงเทรา
Length	75.6 กม.
Desc	คอนกรีต 2 ช่องทาง
Create_date	17 พ.ค. 2508

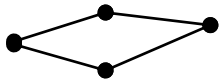
(ก)



(ข)

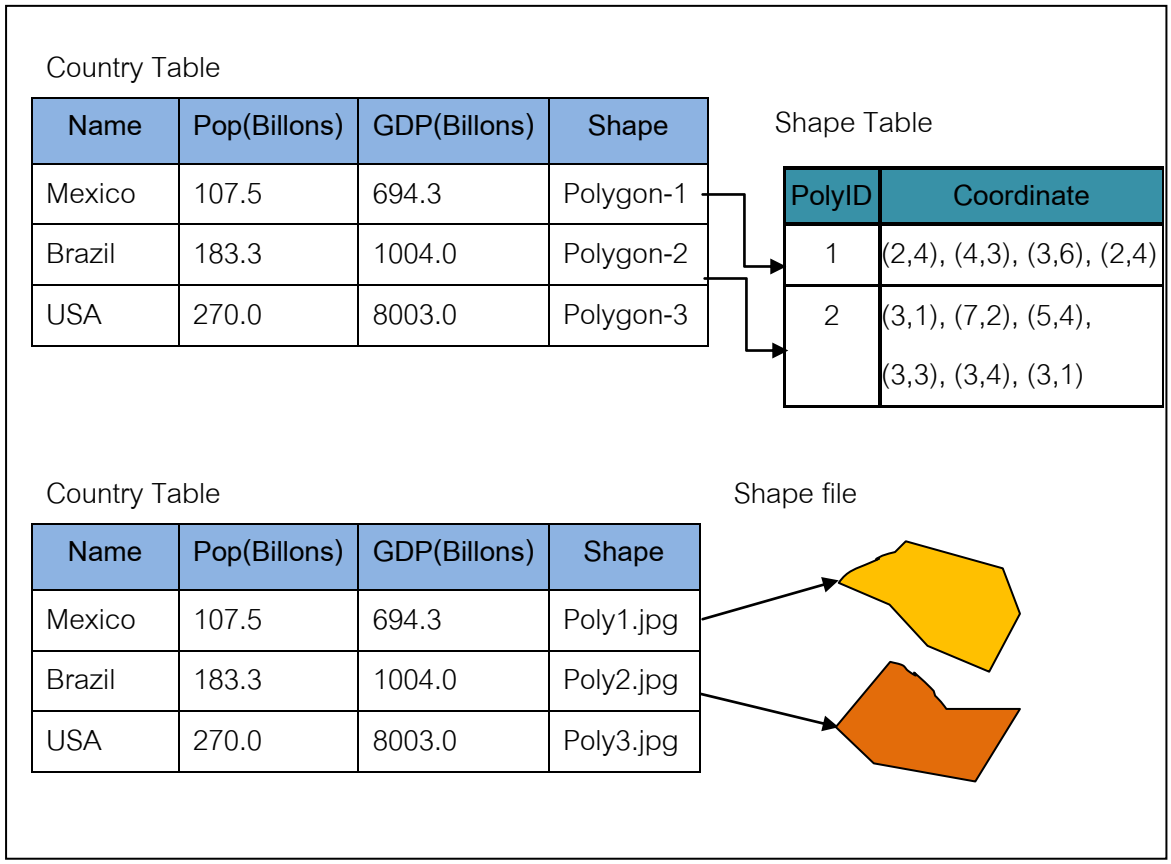
ภาพที่ 2.1 ตัวอย่างข้อมูลเชิงบรรยาย (ก) และข้อมูลเชิงพื้นที่ (ข)

ข้อมูลเชิงพื้นที่นั้นเก็บอยู่ในรูปแบบเดียวกันกับฐานข้อมูลเชิงสัมพันธ์แต่ต่างกันที่ชนิดของข้อมูลที่เก็บในส่วนของข้อมูลเชิงพื้นที่ที่ลักษณะของข้อมูลจะมีการเชื่อมโยงไปยังข้อมูลประเภทที่เป็นพื้นที่ซึ่งจะแสดงอยู่ในรูปของ จุด, เส้น, รูปหลายเหลี่ยม เป็นต้น การแสดงออกถึงลักษณะหรือรูปร่างของวัตถุนั้นเกี่ยวข้องกับการประมวลผลรูปภาพซึ่งแสดงออกมาเป็นรูปร่างประกอบไปด้วยพิกัดจุดบนระนาบแกน X แกน Y หรือ แกน Z ขึ้นอยู่กับมิติของภาพนั้น ๆ และพิกัดนั้นจะไม่ได้มีความสัมพันธ์กันเพียงแค่พิกัดจุดเดียวแต่ประกอบไปด้วยชุดหรือกลุ่มความสัมพันธ์ของพิกัดจุดที่ประกอบเป็นรูปร่างขึ้นมา

รูปแบบวัตถุ	ชุดความสัมพันธ์พิกัดจุด	รูปร่างที่แสดง
จุด	(10,10)	
เส้น	(10,10),(20,20),(30,30)	
รูปหลายเหลี่ยม	(10,10),(20,20),(15,15),(5,5),(10,10)	

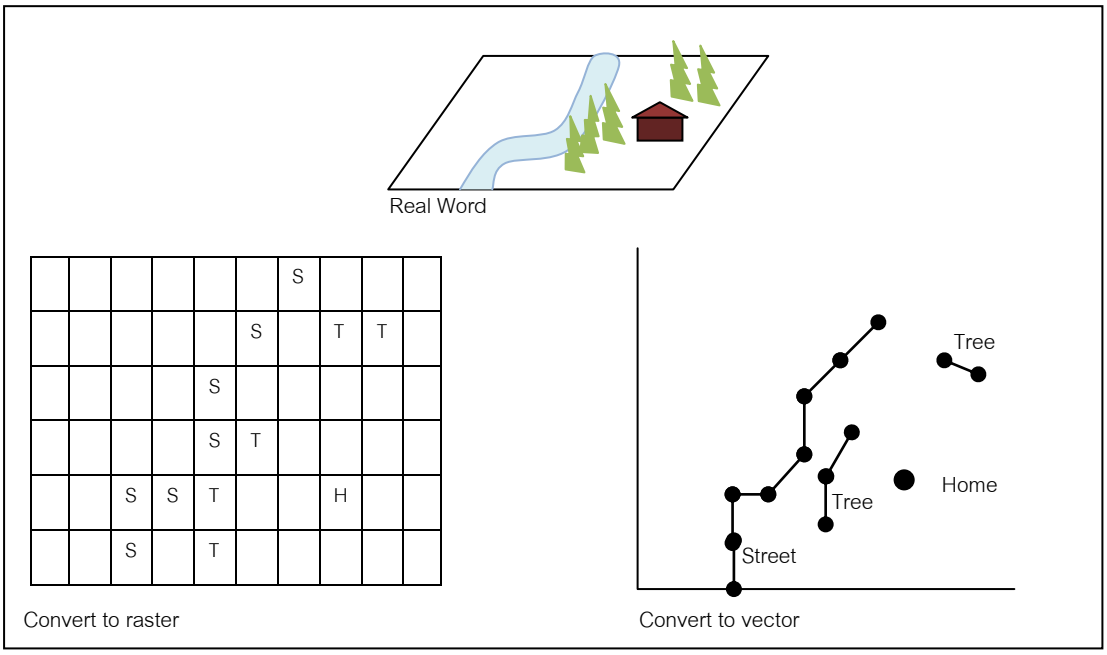
ตารางที่ 2.1 การแสดงรูปร่างจากความสัมพันธ์ของพิกัดจุด

ข้อมูลที่อยู่ในฐานข้อมูลนั้นจะแบ่งออกเป็นสองประเภทมีข้อมูลที่เป็นข้อมูลเชิงบรรยายส่วนหนึ่งและอีกส่วนหนึ่งเก็บข้อมูลที่เป็นพิกัดจุดของรูปร่างนั้น ๆ ในการประมวลผลต้องอ่านข้อมูลพิกัดจุดต่าง ๆ ที่เก็บในตารางแล้วมาดำเนินการแต่บางฐานข้อมูลเก็บข้อมูลที่เป็นรูปร่างของวัตถุเป็นไฟล์ภาพ



ภาพที่ 2.2 ตัวอย่างข้อมูลที่เก็บแบบเก็บพิกัดจุดในตารางและลักษณะข้อมูลที่เก็บแบบไฟล์ภาพ

นอกจากนั้นการแสดงรูปร่างยังแบ่งรูปแบบการแสดงออกเป็น 2 ประเภท คือ แบบราสเตอร์แสดงเป็นแบบจุดพิกเซลบนตารางกริด และแบบเวคเตอร์แสดงบนความสัมพันธ์ระหว่างแกน X และแกน Y



ภาพที่ 2.3 การเปลี่ยนจากพื้นที่จริงไปเป็นรูปแบบของ Raster และ Vector

2.1.2 การดำเนินการของฐานข้อมูลเชิงพื้นที่

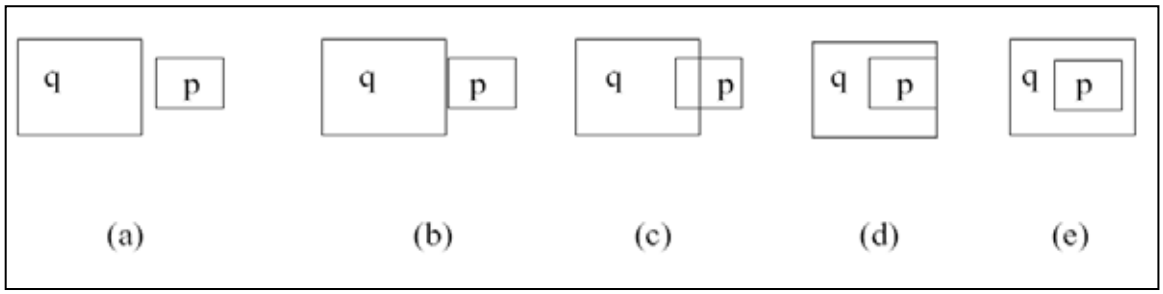
ลักษณะของข้อมูลเชิงพื้นที่ทำให้การดำเนินการของฐานข้อมูลเชิงพื้นที่ต้องใช้เทคนิคที่ต่างจากฐานข้อมูลเชิงสัมพันธ์ทั่วไปในการประมวลผล ในกรณีที่เก็บข้อมูลพิกัดจุดในตารางในการดำเนินการต้องอ่านพิกัดจุดเหล่านั้นขึ้นมาเพื่อดำเนินการตามเงื่อนไขที่ต้องการ

2.1.2.1 การดำเนินการทั่วไปของฐานข้อมูลเชิงพื้นที่

การดำเนินการพื้นฐานที่ใช้ร่วมกับการดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุโดยใช้เทคนิคอาร์-ทรีนั้นมีการดำเนินการที่ใช้ในการหาความสัมพันธ์ระหว่างวัตถุด้วยวิธี Range and topological queries [7] เพื่อเป็นการประมาณการพื้นที่ที่เกี่ยวข้องกับเงื่อนไขที่สนใจ เช่น การดำเนินการค้นหาวัตถุที่อยู่ในกรอบสี่เหลี่ยมที่เป็นพื้นที่ที่ต้องการกำหนดพิกัดสี่เหลี่ยมที่ต้องการมาเพื่อหาวัตถุที่ตรงตามเงื่อนไข ตัวอย่างเช่น การหาวัตถุที่อยู่ในกรอบ (Contains) หรือ การหาวัตถุที่มีการทับซ้อนกัน (Overlap) เป็นต้น โดยที่งานวิจัยนี้ได้ใช้เงื่อนไขการหาวัตถุที่มีการทับซ้อนกันเพื่อช่วยกรองเฉพาะวัตถุที่ตรงตามเงื่อนไขที่ต้องการ Topological Relations ที่ใช้แบ่งออกเป็น 2 ประเภทหลักดังนี้

- Disjoint เป็นความสัมพันธ์ระหว่างวัตถุที่แต่ละวัตถุไม่มีพิกัดใดที่มีความสัมพันธ์กันเลยวิธีการนี้จะเสียเวลาในส่วนของการอ่านข้อมูลขึ้นมาเปรียบเทียบสูงเนื่องจากไม่มีการกรองวัตถุที่มีความเป็นไปได้ก่อน

- Non-disjoint เป็นวิธีที่ใช้ในการกรองหาวัตถุที่มีความเป็นไปได้กับส่วนที่เป็นโหนดรากของอาร์-ทรีก่อนที่จะดำเนินการต่อในส่วนของวัตถุจริงวิธีการนี้จะประกอบไปด้วย Meet, Equal, Overlap, Contains และ Covers



ภาพที่ 2.4 Topological relations (a) Disjoint(q,p), (b) Meet(q,p), (c) Overlap(q,p), (d) Cover(q,p), (e) Contaians(q,p) or Inside(q,p)

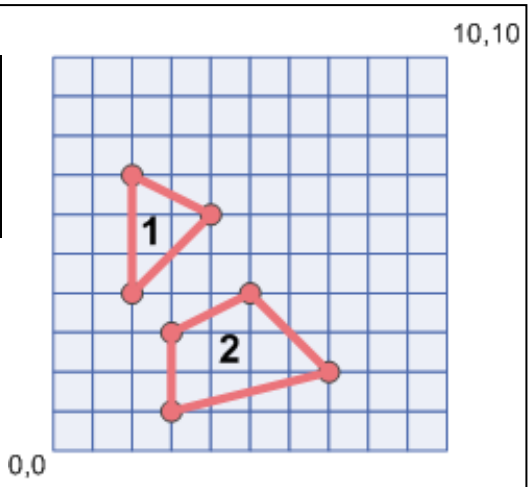
ในงานวิจัยนี้จะใช้ Topological relations ในการกรองวัตถุที่เป็นไปได้ร่วมกับการใช้เทคนิคอาร์-ทรี คือ หาพิกัดของ Minimum bounding rectangles ส่วนที่เป็นโหนดรากที่มีพิกัดที่ทับซ้อนกัน (Overlap) ด้วยข้อสมมติฐานที่ว่าหากวัตถุที่มีความทับซ้อนกันในระดับโหนดรากแล้วในระดับที่เป็นโหนดลูกก็จะทับซ้อนกันด้วย ในทางกลับกันกรณีที่โหนดรากไม่ทับซ้อนกันดังนั้นในระดับที่เป็นโหนดลูกก็จะไม่มีโอกาสที่จะทับซ้อนกันด้วย

2.1.2.2 การดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุ

การดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุเป็นการเชื่อมโยงวัตถุสองวัตถุเข้าด้วยกันตามเงื่อนไขที่กำหนดเพื่อหาว่าระหว่างวัตถุสองวัตถุนั้นมีความสัมพันธ์กันหรือไม่ เช่น การหาว่ามีพื้นที่เขตใดที่เกิดน้ำท่วมบ้างต้องนำพิกัดของน้ำมาเชื่อมความสัมพันธ์กันโดยใช้เงื่อนไขหาพื้นที่ที่มีพิกัดที่ทับซ้อนกันระหว่างน้ำกับพื้นที่เขต กระบวนการในการเชื่อมความสัมพันธ์แบ่งออกเป็น 2 ขั้นตอนดังนี้ [2]

เนื่องด้วยแต่ละวัตถุจะประกอบไปด้วยกลุ่มของพิกัดจุดหลาย ๆ จุดจึงมีการจำกัดขอบเขตของวัตถุให้ชัดเจนก่อนที่จะนำวัตถุดังกล่าวมาหาความสัมพันธ์กันวิธีการจำกัดขอบเขตของวัตถุด้วยเทคนิค Minimum bounding rectangles (MBRs) เพื่อตีกรอบประมาณการวัตถุที่เราสนใจ

ID	Coordinate
1	(2,4), (4,3), (3,6), (2,4)
2	(3,1), (7,2), (5,4), (3,3), (3,4), (3,1)

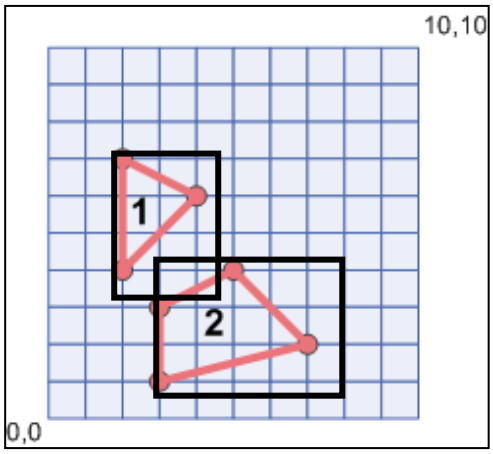


ภาพที่ 2.5 ตัวอย่างการนำพิกัดจุดมาสร้างเป็นรูป

จากข้อมูลพิกัดจุดต่าง ๆ ของวัตถุจะนำมาดำเนินการตีกรอบให้ครอบคลุมพื้นที่ตำแหน่งพิกัดของวัตถุที่มีความสัมพันธ์กันระหว่าง x,y ได้โดยกำหนดกรอบจากการหาค่า ดังนี้

x น้อยที่สุด min_x และพิกัด x มากที่สุด max_x

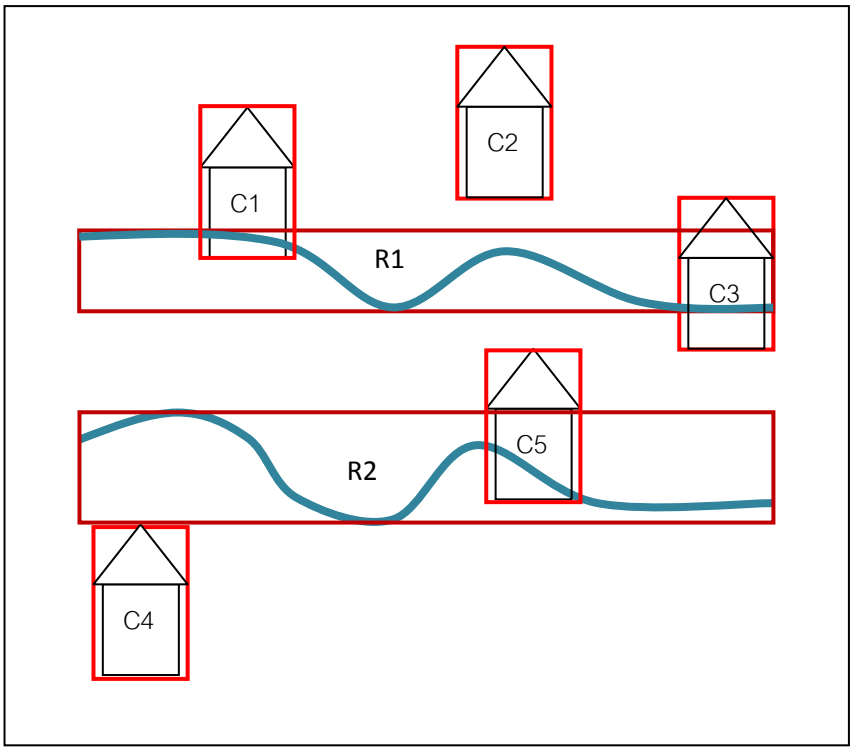
y น้อยที่สุด min_y และพิกัด y มากที่สุด max_y



ภาพที่ 2.6 ตัวอย่างการนำวัตถุมาจำกัดขอบเขตด้วยเทคนิค Mininmum bounding rectangles

หลังจากจำกัดขอบเขตของวัตถุได้แล้วในส่วนของการเชื่อมความสัมพันธ์ระหว่างวัตถุตามเงื่อนไขที่ต้องการในงานวิจัยนี้สนใจเงื่อนไขในการหาพื้นที่ที่ทับซ้อนกันตัวอย่างโจทย์ เช่น “ต้องการหาว่ามีพื้นที่บ้านใดที่มีเส้นทางน้ำไหลผ่านบ้าง” จากในฐานข้อมูลตารางของทางน้ำจะเก็บข้อมูลเป็นพิกัดหรือรูปร่างของทางน้ำ และพิกัดหรือรูปร่างของพื้นที่บ้านต่าง ๆ ไม่ได้มีเขตใดที่แสดงว่าแม่น้ำไหลผ่านพื้นที่บ้านใดบ้าง ในการหาคำตอบนั้นใช้ Spatial join

โดยการหาว่าพิกัดหรือรูปร่างของวัตถุหนึ่งมีการทับซ้อนกับวัตถุใดบ้าง จากตัวอย่างกำหนดให้ C แทนพื้นที่บ้านประกอบด้วย {C1, C2, C3, C4, C5} และทางน้ำแทนด้วย R ประกอบด้วย {R1, R2}



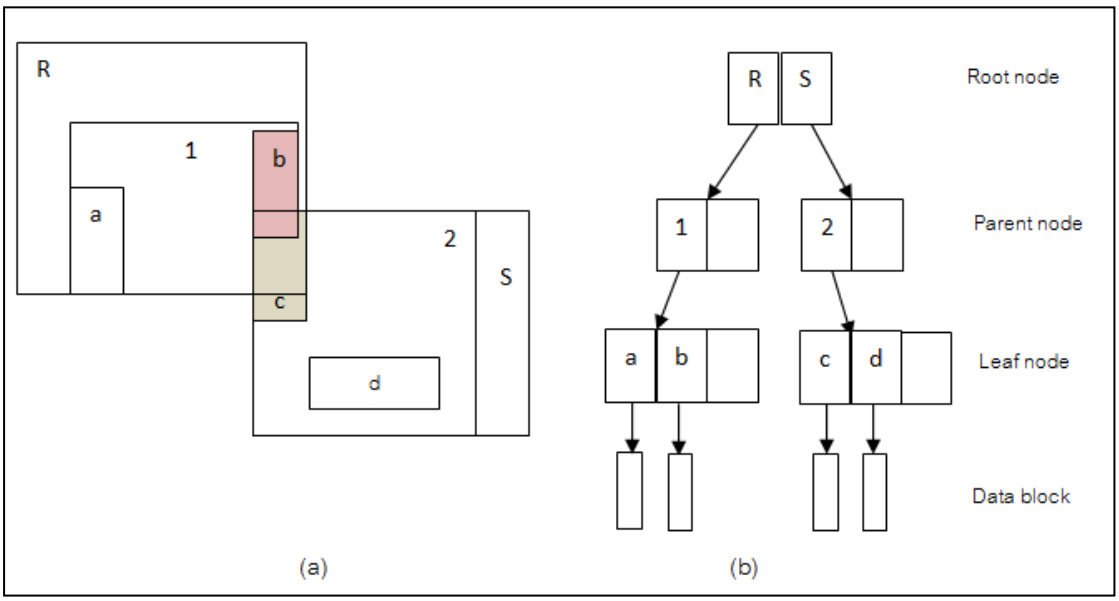
ภาพที่ 2.7 การเชื่อมความสัมพันธ์ระหว่างวัตถุระหว่างพิกัดของบ้านกับพิกัดของน้ำ

จากรูป 2.7 เส้นโค้งแสดงทางน้ำที่ไหลผ่าน R1, R2 และพื้นที่บ้าน C1-C6 ก่อนที่จะดำเนินการหาว่ามีพิกัดของทางน้ำใดที่ทับซ้อนกับพิกัดพื้นที่บ้านบ้างนั้นต้องตีกรอบจำกัดขอบเขตของแต่ละวัตถุก่อนดังกรอบสี่เหลี่ยมแล้วจึงหาว่ากรอบสี่เหลี่ยมของทางน้ำกับพิกัดพื้นที่บ้านใดที่มีการทับซ้อนกันบ้างจากตัวอย่างได้คำตอบดังนี้ R1 Overlap C1, C3 และ R2 Overlap C5 วิธีการที่ง่ายที่สุดในการดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุคือการนำวัตถุทุก ๆ วัตถุมาเปรียบเทียบกัน วิธีการนี้ไม่ซับซ้อนแต่ใช้ได้ดีเฉพาะในกรณีที่มีจำนวนวัตถุที่น้อย จากตัวอย่างภาพที่ 2.7 จำนวนครั้งในการประมวลผล คือ จำนวนวัตถุ R x จำนวนวัตถุ C = 10 ครั้ง กรณีที่จำนวนวัตถุมากทำให้จำนวนครั้งในการประมวลผลมากขึ้นในงานวิจัย [2] สำรวจเทคนิคในการเชื่อมความสัมพันธ์ระหว่างวัตถุไว้หลายวิธีด้วยกัน ในงานวิจัยนี้ใช้เทคนิคอาร์-ทรีมาช่วยลดจำนวนครั้งในการประมวลผล

2.1.2.3 การดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุโดยใช้เทคนิคอาร์-ทรี

อาร์-ทรีเป็นวิธีที่หลายงานวิจัยใช้เพื่อเพิ่มความเร็วในการทำ Spatial Join งานวิจัย [1] และ [8] พบว่าการใช้อาร์-ทรีสามารถเพิ่มความเร็วในการทำ Spatial Join ได้

การดำเนินการพื้นฐานของอาร์-ทรีอ้างอิงมาจากบี-ทรีเป็นการค้นหาคำตอบที่มีโครงสร้างของข้อมูลเป็นต้นไม้โดยมีโหนดที่เป็นรากเป็นทางเข้าไปสู่โหนดที่เป็นพ่อแม่เพื่อนำไปสู่โหนดลูกที่จะชี้ไปยังคำตอบที่ต้องการ ความต่างของบี-ทรีกับอาร์-ทรีอยู่ที่ในส่วนของการดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุต้นไม้สร้างจากขั้นตอนในการทำ Minimum Bounding Rectangles แล้วนำวัตถุที่เกี่ยวข้องใน MBRs นั้นมาแปลงให้อยู่ในรูปของต้นไม้แล้วหาวัตถุที่เกิดการทับซ้อนกัน (Intersection) ผลลัพธ์ที่ได้จากขั้นตอนของการค้นหาข้อมูลในต้นไม้นั้นดัชนีที่ชี้ไปยังข้อมูลที่เก็บจริงใน Data Block จะอยู่ในส่วนของโหนดลูกที่จะอ่านข้อมูลจริงจาก Data Block ขึ้นมาเพื่อดำเนินการในขั้นตอนต่อไป



ภาพที่ 2.8 ตัวอย่างวัตถุที่อยู่ในรูปของ MBRs (a) และแปลงให้อยู่ในรูปของต้นไม้ (b)

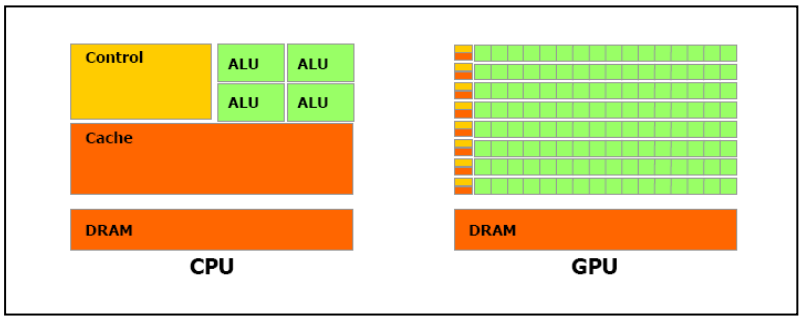
จากภาพที่ 2.8 ส่วนที่วัตถุเกิดการทับซ้อนกันคือ b และ c โดยในการค้นหาในต้นไม้นั้นจะทำการตรวจสอบว่าในโหนดลูกของ R กับ S ได้แก่ R{a,b} และ S{c,d} มีวัตถุใดบ้างที่มีการทับซ้อนกันเพื่อที่จะอ่านข้อมูลขึ้นมาทำการเชื่อมความสัมพันธ์กันระหว่างวัตถุโดยมีข้อสันนิษฐานว่าวัตถุที่มีพิกัดที่ทับซ้อนกันจากโหนดรากจะมีโอกาสที่จะทับซ้อนกันในโหนดลูก ในทางกลับกันถ้าวัตถุใดที่ไม่มีพื้นที่ใดที่มีพิกัดที่ทับซ้อนกันในโหนดรากจะไม่เกิดการทับซ้อนกันในโหนดลูกด้วยจากนั้นจะอ่านข้อมูลจากบล็อกเก็บข้อมูลในหน่วยความจำที่ชี้จากดัชนีในโหนดลูกที่เกิดการทับซ้อนกันขึ้นมาเพื่อดำเนินการเชื่อมความสัมพันธ์ จากตัวอย่างเมื่อค้นหาเข้าไปใน

ต้นไม้จะพบว่ามีโหนด b และ c ที่ตรงตามเงื่อนไข ทั้งสองโหนดมีพิกัดที่ทับซ้อนกันหลังจากนี้จะนำบล็อกเก็บข้อมูลที่อยู่ใน b และ c มาเปรียบเทียบกันว่ามีแถวข้อมูลใดบ้างที่ตรงตามเงื่อนไขที่ต้องการ

2.1.3 หน่วยประมวลผลกราฟิก

2.1.3.1 วิวัฒนาการและคุณสมบัติของหน่วยประมวลผลกราฟิก

Graphics Processing Unit [3] คือ ส่วนที่ใช้ในการประมวลผลกราฟิกอยู่ในการ์ดแสดงผลที่มีอยู่ในคอมพิวเตอร์ด้วยความต้องการในงานด้านกราฟิกไม่ว่าจะเป็นเกมส์ ภาพยนตร์ หรืองานในด้าน แอนนิเมชันอื่น ๆ มีความต้องการที่จะแสดงความสมจริงของงานจึงมีความต้องการความรวดเร็วในการประมวลผลส่งผลให้มีการพัฒนาหน่วยประมวลผลกราฟิกเพื่อให้มีความรวดเร็ว และประสิทธิภาพในการประมวลผลมากขึ้น โดยลักษณะการทำงานนั้นเป็นแบบขนานมีจำนวนหน่วยประมวลผลเป็นจำนวนมากเพื่อเพิ่มความรวดเร็วในการประมวลผล โครงสร้างเป็นแบบ Multithread และ High memory bandwidth ลักษณะการประมวลผลเป็นแบบ Single Instruction Multiple Data (SIMD) คือในหนึ่งคำสั่งสามารถประมวลผลได้กับข้อมูลที่ต่างกันได้พร้อมกันทำให้ลดระยะเวลาในการประมวลผลกับข้อมูลมาก ๆ ในการประมวลผลหนึ่งรอบคำสั่ง



ภาพที่ 2.9 การเปรียบเทียบโครงสร้างของ CPU และ GPU

นอกจากหน่วยประมวลผลกราฟิกจะประมวลผลในงานด้านกราฟิกแล้ว ปัจจุบันได้มีการพัฒนานำคุณสมบัติของหน่วยประมวลผลกราฟิกมาใช้แก้ปัญหาในงานอื่นได้ โดยสามารถเขียนคำสั่งให้หน่วยประมวลผลกราฟิกประมวลผลในงานอื่นได้ Owens et al. [9] ได้ทำการสำรวจว่ามีการนำหน่วยประมวลผลกราฟิกไปใช้ในการแก้ปัญหาอื่นการดำเนินการฐานข้อมูลเชิงพื้นที่ก็มีงานวิจัยที่นำคุณสมบัติของหน่วยประมวลผลกราฟิกไปประยุกต์ใช้ เช่น งานวิจัย [10] ใช้หน่วยประมวลผลกราฟิกแก้ปัญหาในการเชื่อมความสัมพันธ์ระหว่างวัตถุของฐานข้อมูลเชิงพื้นที่ ผลการทดลองของงานวิจัยที่ผ่านมาพบว่าสามารถเพิ่มความเร็วในการประมวลผลการดำเนินการฐานข้อมูลเชิงพื้นที่ได้

2.1.3.2 General purpose computing on Graphic processing

ด้วยวิวัฒนาการ ความเร็วและประสิทธิภาพของหน่วยประมวลผลกราฟิกในปัจจุบันจึงมีการพัฒนาให้สามารถเขียนโปรแกรมคำสั่งให้ประมวลผลกับปัญหาอื่น ๆ เช่น งานในด้านวิทยาศาสตร์ และการคำนวณคณิตศาสตร์ที่มีความซับซ้อนที่ต้องการความรวดเร็ว การดำเนินการฐานข้อมูลเชิงพื้นที่ก็เช่นเดียวกัน ในการเขียนคำสั่งให้หน่วยประมวลผลกราฟิกนั้นต้องมีการติดต่อกับส่วนของอุปกรณ์ที่อยู่ในการ์ดแสดงผลผ่าน Application Programming Interfaces (APIs) หรือ Driver ของอุปกรณ์ซึ่งจะต้องมีภาษาสำหรับเขียนโปรแกรมเฉพาะ เช่น OpentGL, DirectX หรือ CUDA C ในงานวิจัยนี้จะนำ CUDA C มาใช้ซึ่งลักษณะของภาษามีโครงสร้างที่เหมือนกับภาษาซี และมีชุดคำสั่งให้ผู้เขียนโปรแกรมสามารถเขียนคำสั่งได้ง่ายโดยที่ผู้เขียนโปรแกรมไม่จำเป็นต้องมีความรู้เกี่ยวกับในเชิงของ Graphic rendering code มากนัก

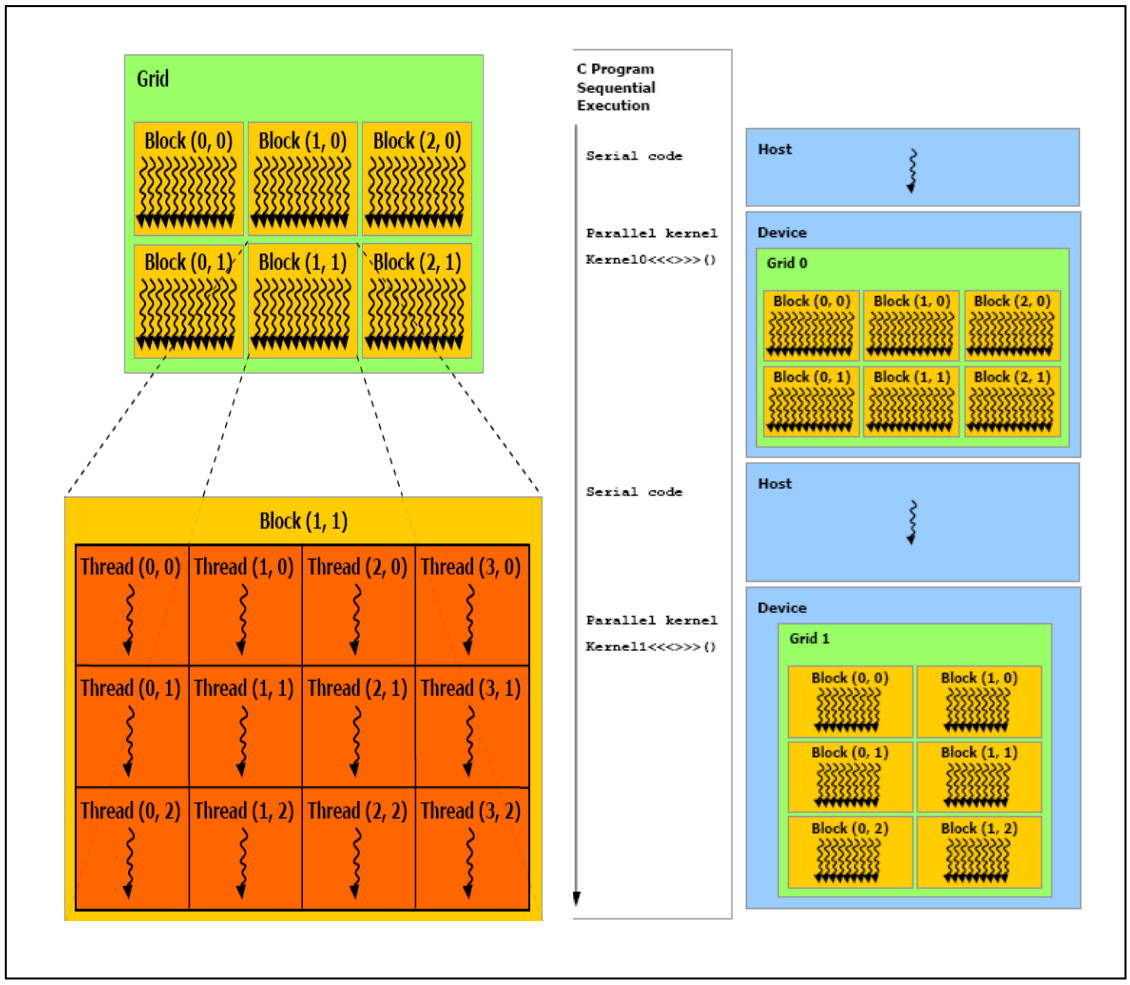
2.1.3.3 **การเขียนคำสั่งบนหน่วยประมวลผลกราฟิกโดยใช้** Compute Unified Device Architecture

- Thread Hierarchy หน่วยประมวลผลกราฟิกจะแบ่งโครงสร้างภายในออกเป็นบล็อกภายในบล็อกจะแบ่งย่อยออกเป็นเทรด เรียกว่า กลุ่มของเทรด [6] ในแต่ละเทรดสามารถใช้ทรัพยากรร่วมกันได้ เช่น หน่วยความจำภายใน เป็นต้น ในหนึ่งคำสั่งแต่ละเทรดจะประมวลผลกับข้อมูลของตัวเองและจะประมวลผลหลาย ๆ เทรดไปพร้อม ๆ กันแต่เนื่องจากข้อจำกัดของหน่วยประมวลผลกราฟิกยังไม่สามารถทำงานได้เองทั้งหมดยังมีงานบางส่วนที่ต้องดำเนินการบนหน่วยประมวลผลกลางดังนั้นในการเขียนโปรแกรมให้หน่วยประมวลผลกราฟิกแบ่งออกเป็น 2 ส่วนคือ ส่วนที่ทำงานบนหน่วยประมวลผลกลางเรียกว่า Host Code ส่วนที่เป็นการประมวลผลบนหน่วยประมวลผลกราฟิกเรียกว่า Device Code ขั้นตอนในการประมวลผลนั้น Host Code กำหนดค่าเริ่มต้นต่าง ๆ ก่อนการเริ่มประมวลผล เช่น การกำหนดตัวแปร การเตรียมข้อมูลที่จะใช้ประมวลผล และควบคุมการส่งข้อมูลระหว่างหน่วยประมวลผลกลางเพื่อไปใช้ประมวลผลบนหน่วยประมวลผลกราฟิก ในการประมวลผลมีขั้นตอน 3 ขั้นตอนดังนี้

1) Host Code จองพื้นที่บนหน่วยประมวลผลกราฟิกสำหรับอินพุตที่ใช้ในการประมวลผล และสำหรับเอาท์พุตที่ได้จากการประมวลผล หลังจากจองพื้นที่บนหน่วยประมวลผลกราฟิกแล้วจะทำการส่งข้อมูลขึ้นไปให้หน่วยประมวลผลกราฟิกเพื่อประมวลผล

2) Device Code หน่วยประมวลผลกราฟิกเรียกส่วนเคอร์เนและแบ่งการทำงานออกเป็นเทรดแต่ละเทรดทำคำสั่งเดียวกันพร้อมกันแต่จะประมวลผลกับข้อมูลที่ต่างกัน

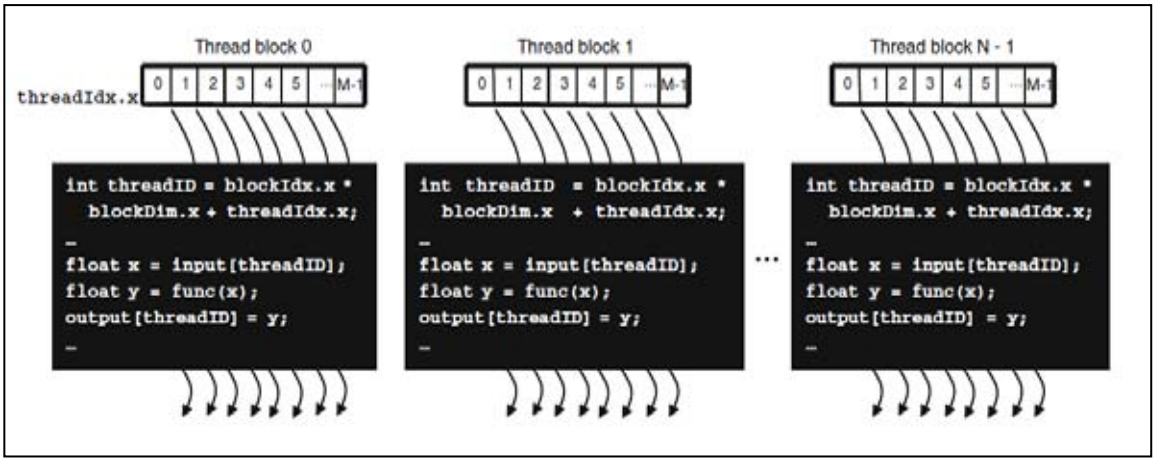
3) เมื่อแต่ละเทรดประมวลผลเสร็จแล้วจะทำการส่งผลลัพธ์ที่ได้ส่งกลับมายัง Host



ภาพที่ 2.10 การประสานการทำงานระหว่างบนหน่วยประมวลผลกลาง
กับหน่วยประมวลผลกราฟิก

- CUDA Thread organization ในการเขียนคำสั่งให้หน่วยประมวลผลกราฟิกด้วย CUDA C เมื่อพบการเรียกใช้เคอร์เนลจะส่งไปประมวลผลบนหน่วยประมวลผลกราฟิก CUDA C มีคำสั่งในการกระจายการประมวลผลออกไปเป็นเทรดที่ซึ่งแต่ละเทรดมีหมายเลขของตนเองที่ไม่ซ้ำกันเรียกว่า TheradID ในแต่ละเทรดประมวลผลคำสั่งเดียวกันแต่นำข้อมูลคนละข้อมูลไปประมวลผลพร้อม ๆ กันแบบขนานซึ่งจะต้องกำหนดจำนวนว่าจะใช้เทรดจำนวนเท่าใดและแต่ละเทรดจะประมวลผลกับข้อมูลชุดใดโดยการอ้างอิงจากหมายเลข ThreadID เมื่อเสร็จแล้วก็จะส่งผลลัพธ์กลับไปยังหน่วยประมวลผลกลาง

การแบ่งเทรดบนหน่วยประมวลผลกราฟิกแบ่งออกเป็นลำดับชั้น [11] คือ หน่วยใหญ่สุดจะเป็นกริดภายในกริดแบ่งเป็นบล็อกและย่อยลงไปเป็นเทรดโดยการอ้างถึงแต่ละบล็อกหรือเทรดจะใช้หมายเลขของบล็อก (blockIdx) และเทรด (threadIdx)



ภาพที่ 2.11 การกระจายเทรดภายในหน่วยประมวลผลกราฟิก

ภาพที่ 2.11 เป็นตัวอย่างกรณีที่กริดแบ่งเป็น 1 มิติ CUDA สร้างเทรดและกระจายงานไปยังเทรดต่าง ๆ โดยภายในกริดจะแบ่งออกเป็นบล็อกจะใช้ blockIdx.x เป็นหมายเลขอ้างอิง โดยมีค่าตั้งแต่ 0 ถึง N-1 บล็อกและภายในบล็อกจะแบ่งย่อยออกไปเป็นเทรดจะใช้ threadIdx.x เป็นหมายเลขอ้างอิง โดยจะมีค่าตั้งแต่ 0 ถึง M-1 เทรดจากการแบ่งดังกล่าวจะได้จำนวนเทรดที่ใช้ประมวลผลทั้งหมด N * M เทรดตัวอย่าง เช่น มี 10 บล็อก แต่ละบล็อกมี 10 เทรดดังนั้นจำนวนเทรดที่ใช้จะเท่ากับ 10 x 10 = 100 เทรด

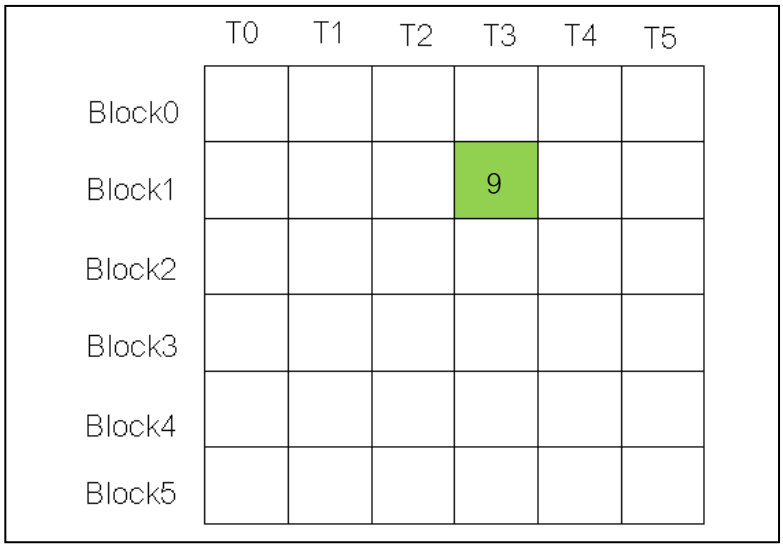
ถึงแม้ว่าลักษณะการแบ่งเทรดของ CUDA สามารถมองเป็นแบบ 2 มิติหรือ 3 มิติได้ แต่โดยปกติลักษณะของพื้นที่บนหน่วยความจำทางกายภาพมองเพียงแค่ 1 มิติ ดังนั้นการอ้างถึงหมายเลขเทรดของ CUDA จะหาได้ดังนี้

threadID = blockIdx.x * blockDim + ThreadIdx

blockIdx.x คือหมายเลขประจำบล็อก

blockDim คือจำนวนเทรดแต่ละบล็อก

ตัวอย่าง หา threadID ของเทรดที่ 3 ของบล็อกที่ 1 กำหนดให้ภายในบล็อกมี 6 เทรดจะหาได้ดังนี้ threadID = 1*6+3 = 9 ดังนั้น threadID ของเทรดที่ 3 ของบล็อกที่ 1 คือ 9



ภาพที่ 2.12 การคำนวณหาหมายเลขเทรด

ในการกำหนดหมายเลข ThreadID ก็เพื่อใช้ในการอ้างถึงสำหรับนำข้อมูลที่เป็นอินพุตและเอาท์พุตบนหน่วยประมวลผลกราฟิก โดยทั่วไปแล้วโครงสร้างแบ่งกริดออกเป็น 2 มิติ แต่ละบล็อกจะแตกเทรดออกได้เป็น 3 มิติทั้งนี้จำนวนบล็อกและเทรดที่จะแตกออกไปขึ้นอยู่กับข้อจำกัดของการ์ดแสดงผลในแต่ละรุ่นด้วยและขึ้นอยู่กับการเขียนคำสั่งกำหนดให้กระจายเทรดในการประมวลผลอย่างไรให้หน่วยประมวลผลกราฟิกสามารถใช้งานได้อย่างเต็มประสิทธิภาพ

การพิจารณาการใช้หน่วยประมวลผลกราฟิกในการแก้ปัญหาโดยมองการประมวลผลเป็นแบบขนานเหมาะสำหรับปัญหาที่มีจำนวนข้อมูลที่จะประมวลผลจำนวนมาก โดยข้อมูลแต่ละจุดทำงานด้วยคำสั่งเดียวกันถูกประมวลผลในเวลาเดียวกันทำให้การประมวลผลทำได้เร็วขึ้น เช่น ปัญหาในการบวกเวกเตอร์ เป็นคำสั่งในการบวกชุดตัวเลข ในแถวลำดับสองชุดแล้วเก็บคำตอบไว้ในแถวลำดับชุดที่สาม การอ้างถึงข้อมูลแถวลำดับด้วยหมายเลขตำแหน่งของข้อมูลในแถวลำดับ

Array A = [0 1 2 3 4]

+

Array B = [5 6 7 8 9]

=

Array C = [5 7 9 11 13]

จากตัวอย่างโจทย์การประมวลผลจะนำ A[i] + B[i] และเก็บผลลัพธ์ไว้ที่ C[i] และทำซ้ำคำสั่งเดิมจนหมดข้อมูลในแถวลำดับ กรณีที่เป็นการประมวลผลแบบลำดับจำนวนรอบของการทำงานจะเท่ากับจำนวนข้อมูลในแถวลำดับแต่ถ้านำปัญหานี้มาดำเนินการแบบขนานจะสามารถทำงานเพียงแค่รอบเดียวกล่าวคือสามารถบวกข้อมูลในแถวลำดับได้พร้อมกันในเวลาเดียวกัน

```
void add( int *a, int *b, int *c ) {
        for(int i=0; i<N; i++){
        c[i] = a[i] + b[i];
        }
}
```

ภาพที่ 2.13 การประมวลผลบวกเวกเตอร์แบบลำดับ

```
__global__ void add( int *a, int *b, int *c ) {
    int tid = blockIdx.x;    // this thread handles the data at its thread id
    if (tid < N)
        c[tid] = a[tid] + b[tid];
}
```

ภาพที่ 2.14 การประมวลผลบวกเวกเตอร์แบบขนาน

- **CUDA C Programming** CUDA C พัฒนาโดย NVIDIA [3] เพื่อลดข้อจำกัดในการเขียนคำสั่งให้หน่วยประมวลผลกราฟิกจากเดิมผู้เขียนคำสั่งจะเขียนคำสั่งให้หน่วยประมวลผลกราฟิกได้จะต้องมีความรู้ส่วนของการประมวลผลภาพเพื่อที่จะเขียนคำสั่ง เพื่อลดข้อจำกัดดังกล่าวจึงได้มีการพัฒนา CUDA C ขึ้นเพื่อให้ผู้พัฒนาสามารถเขียนคำสั่งโดยผ่าน APIs ของ CUDA C ได้ง่ายขึ้นโดยมีรูปแบบโครงสร้างของภาษาคล้ายคลึงกับภาษาซี คำสั่งที่จะถูกประมวลผลบนหน่วยประมวลผลกราฟิกนั้นจะเรียกว่า kernels เมื่อมีคำสั่งเรียกใช้งานส่วนของ kernels จะประมวลผลไปพร้อม ๆ กันในแต่ละเทรดการเขียนฟังก์ชันของ CUDA C จะมีลักษณะคล้าย ๆ กับภาษาซีแต่จะต่างกันที่ชุดคำสั่งที่เรียกใช้ เช่น รูปแบบการประกาศฟังก์ชันบนหน่วยประมวลผลกราฟิกจะมีดังนี้

```
// Kernel definition
__global__ void VecAdd(float* A, float* B, float* C)
{
    int i = threadIdx.x;
    C[i] = A[i] + B[i];
}

int main()
{
    ...
    // Kernel invocation with N threads
    VecAdd<<<1, N>>>(A, B, C);
}
```

ภาพที่ 2.15 การประกาศฟังก์ชันของ CUDA C

จากรูป 2.15 แสดงให้เห็นการทำงานที่ต้องประสานกันระหว่างหน่วยประมวลผลกลางกับหน่วยประมวลผลกราฟิกส่วนที่เป็นฟังก์ชันที่จะทำงานบนหน่วยประมวลกราฟิกนั้นจะอยู่ภายใต้ฟังก์ชัน __global__ void function_name(parameter) ข้อมูลที่จะประมวลผลจะถูกส่งไปประมวลผลบนเทรดต่าง ๆ การอ้างถึงโดยใช้หมายเลข threadIdx.x ส่วนคำสั่งที่เรียกใช้ฟังก์ชันจะถูกเรียกใช้บนหน่วยประมวลผลกลางภายใต้ฟังก์ชันหลัก main() เมื่อมีการเรียกใช้ฟังก์ชันให้ประมวลผลต้องกำหนดจำนวนเทรดที่จะใช้ในการประมวลผลการกำหนดจำนวนเทรดภายใต้สัญลักษณ์ <<<blocks, threads>>>

การคำนวณจำนวนเทรดได้โดยระบุจำนวนบล็อกที่ใช้และกำหนดมิติของบล็อกว่าจะมีกี่เทรดในหนึ่งบล็อก

ตัวอย่าง กำหนดจำนวนเทรดดังนี้ VecAdd<<<1, 100>>>(a, b, c) หมายถึงใช้จำนวนบล็อก 1 บล็อกในบล็อกมีจำนวนเทรด 100 เทรด ดังนั้นจำนวนเทรดที่ใช้ประมวลผลมีทั้งหมด 1x 100 = 100 เทรด

ก่อนที่จะประมวลผลนั้นต้องกำหนดตัวแปรและข้อมูลที่ใช้ซึ่งกระบวนการนี้จะทำบนหน่วยประมวลผลกลางจากนั้นข้อมูลทั้งหมดถูกส่งไปประมวลผลบนหน่วยประมวลผลกราฟิกดังนั้นจึงต้องมีการจองพื้นที่ที่นำข้อมูลไปเก็บ และพื้นที่ส่วนที่จะใช้เก็บผลลัพธ์ ข้อมูลถูกส่งจากหน่วยประมวลผลกลางไปยังหน่วยประมวลผลกราฟิกด้วยการจับคู่กันระหว่างตำแหน่งของแถวลำดับบนหน่วยประมวลผลกลางกับหมายเลขเทรดบนหน่วยประมวลผลกราฟิก

```
    cudaMalloc(&d_C, size);

    // Copy vectors from host memory to device memory
    cudaMemcpy(d_A, h_A, size, cudaMemcpyHostToDevice);
    cudaMemcpy(d_B, h_B, size, cudaMemcpyHostToDevice);

    // Invoke kernel
    int threadsPerBlock = 256;
    int blocksPerGrid =
            (N + threadsPerBlock - 1) / threadsPerBlock;
    VecAdd<<<blocksPerGrid, threadsPerBlock>>>(d_A, d_B, d_C, N);

    // Copy result from device memory to host memory
    // h_C contains the result in host memory
    cudaMemcpy(h_C, d_C, size, cudaMemcpyDeviceToHost);

    // Free device memory
    cudaFree(d_A);
    cudaFree(d_B);
    cudaFree(d_C);

    // Free host memory
    ...
```

ภาพที่ 2.16 การจองพื้นที่และส่งข้อมูลไปยังหน่วยประมวลผลกราฟิก

จากรูป 2.16 คำสั่งในการจองพื้นที่จะใช้ cudaMalloc(variable, size) คำสั่งในการส่งข้อมูลจากหน่วยประมวลผลกลางไปยังหน่วยประมวลผลกราฟิกใช้ cudaMemcpy(device_variable, host_variable, size, cudaMemcypHostToDevice) เมื่อหน่วยประมวลผลกราฟิกประมวลผลเสร็จส่งผลลัพธ์กลับมายังหน่วยประมวลผลกลางโดยใช้คำสั่ง cudaMemcpy(host_variable, device_variable, size, cudaMemcypDeviceToHost) หลังจากนั้นคืนพื้นที่หน่วยความจำโดยใช้คำสั่ง cudaFree(variable)

จากคุณสมบัติของหน่วยประมวลผลกราฟิกที่รองรับการทำงานแบบขนานสามารถประมวลผลได้เร็วกว่าแบบลำดับและสามารถประมวลผลได้กับปัญหาที่มีความซับซ้อนและมีจำนวนข้อมูลที่ต้องการประมวลผลปริมาณมากงานวิจัย [9] ได้สำรวจงานวิจัยที่นำหน่วยประมวลผลกราฟิกไปช่วยเพิ่มความเร็วในการประมวลผลพบว่าสามารถเพิ่มความเร็วในการประมวลผลได้

2.2 งานวิจัยที่เกี่ยวข้อง

2.2.1 งานวิจัยเกี่ยวกับการเชื่อมความสัมพันธ์ระหว่างวัตถุสำหรับฐานข้อมูลเชิงพื้นที่ใช้เทคนิคอาร์-ทรี

- Sequential R-tree งานวิจัย [1] นำเทคนิคอาร์-ทรีมาดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุเพื่อเพิ่มความเร็วในการประมวลผลเนื่องจากวิธีของอาร์-ทรี สามารถลดจำนวนครั้งในการประมวลผลลงได้โดยอาร์-ทรีจะทำการตัดเส้นทางที่ไม่นำไปสู่คำตอบออกไปและจะประมวลผลเฉพาะเส้นทางที่จะนำไปสู่คำตอบเท่านั้น และจากผลการทดลองของงานวิจัยดังกล่าวพบว่าช่วยลดระยะเวลาอ่านข้อมูลจากหน่วยความจำได้เพราะจะเข้าถึงข้อมูลเฉพาะส่วนที่จะนำไปสู่คำตอบเท่านั้นซึ่งจะต่างจากการทำแบบ Nested-loop เพราะวิธีการแบบวนลูปจะทำการเปรียบเทียบกับข้อมูลทั้งหมด โดยวิธีการของอาร์-ทรีขั้นตอนวิธี ไว้ดังนี้

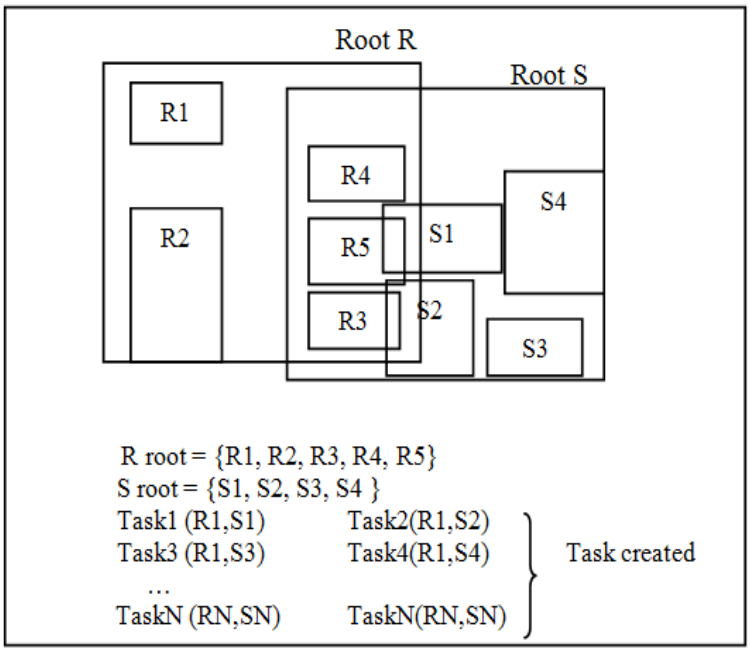
```
SpatialJoin(R,S: R_node);
        For (all E_S ∈ S) DO
                For (all E_R ∈ R with E_R.rect ∩ E_S.rect ≠ ∅ ) DO
                If (R is a leaf page) Then
                        Output (E_R , E_S )
                Else
                        ReadPage(E_R.ref); ReadPage(E_S.ref)
                        SpatialJoin(E_R.ref, E_S.ref)

                End
        End
End SpatialJoin;
```

ภาพที่ 2.17 รหัสเทียมการประมวลผลอาร์-ทรี

มีงานวิจัยที่นำมาประยุกต์ใช้ต่อในการเพิ่มประสิทธิภาพและความเร็วในการประมวลผลการสร้างความสัมพันธ์ระหว่างวัตถุสำหรับฐานข้อมูลเชิงพื้นที่อีกหลากหลายแต่เนื่องจากการออกแบบวิธีการและการทดลองยังใช้หน่วยประมวลผลกลางเท่านั้น

- Parallel R-tree งานวิจัย [8] ได้นำเทคนิคอาร์-ทรีมาประยุกต์ให้ทำงานแบบขนานเพื่อเพิ่มความเร็วในการประมวลผลโดยจะกระจายงานออกเป็นงานย่อยในส่วนของการเปรียบเทียบว่าส่วนของ MBRs ใดบ้างที่มีพิกัดที่เกิดการทับซ้อนกันโดยแต่ละงานย่อยจะประมวลผลไปพร้อม ๆ กัน

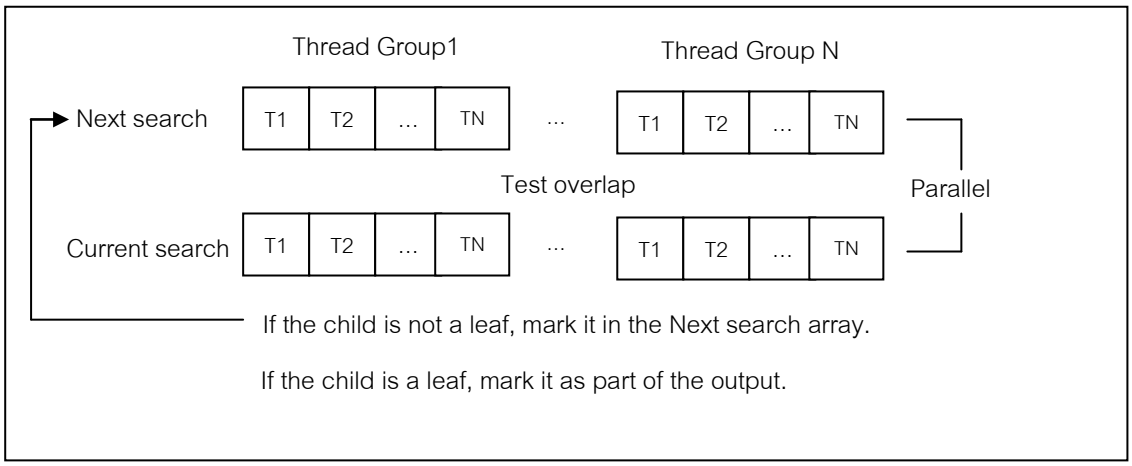
แต่งานวิจัยดังกล่าวได้ออกแบบและทำการทดลองกระบวนการสำหรับประมวลผลบนหน่วยประมวลผลกลางเท่านั้น ในงานวิจัยนี้จึงได้นำแนวคิดในการกระจายงานเพื่อประมวลผลแบบขนานนี้ไปประยุกต์ใช้ในการประมวลผลบนหน่วยประมวลผลกราฟิก



ภาพที่ 2.18 การกระจายงานแบบขนานในเทคนิคอาร์-ทรี

2.2.2 งานวิจัยที่ใช้หน่วยประมวลผลกราฟิกมาใช้สำหรับฐานข้อมูลเชิงพื้นที่

งานวิจัย [12] ได้นำเทคนิคการค้นหาข้อมูลของฐานข้อมูลเชิงพื้นที่มาประมวลผลบนหน่วยประมวลผลกราฟิกโดยใช้เทคนิคของอาร์-ทรี โดยกระบวนการในการประมวลผลจะมีขั้นตอนเหมือนกับการใช้เทคนิคของ Parallel R-tree ทั่วไปแต่ใช้หน่วยประมวลผลกราฟิกกระจายการทำงานออกไปบนเทรดต่าง ๆ บนหน่วยประมวลผลกราฟิกและทำงานเป็นแบบขนานร่วมกันกับเทคนิคอาร์-ทรี



ภาพที่ 2.19 การดำเนินการค้นหาวัตถุแบบขนานบนหน่วยประมวลผลกราฟิก

ถึงแม้ว่างานวิจัยดังกล่าวจะออกแบบการประมวลผลบนหน่วยประมวลผลกราฟิก แต่วิธีการดังกล่าวออกแบบเพื่อการค้นหาข้อมูลเท่านั้นยังไม่ได้ออกแบบสำหรับการดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุสำหรับฐานข้อมูลเชิงพื้นที่จึงได้มีแนวคิดจะประยุกต์กระบวนการดังกล่าวดำเนินการต่อในส่วนของการดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุสำหรับฐานข้อมูลเชิงพื้นที่บนหน่วยประมวลผลกราฟิก

บทที่ 3
ระเบียบขั้นตอนวิธีที่เสนอ

ระเบียบขั้นตอนวิธี

จากปัญหาความซับซ้อนของฐานข้อมูลเชิงพื้นที่ และการดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุสำหรับฐานข้อมูลเชิงพื้นที่เปรียบเทียบพิกัดจุดระหว่างวัตถุสองวัตถุว่าตรงตามเงื่อนไขที่กำหนดหรือไม่ในงานวิจัยนี้สนใจเพิ่มความเร็วในการประมวลผลในส่วนกระบวนการในการเปรียบเทียบระหว่างสองวัตถุ จากเดิมที่เป็นการประมวลผลแบบลำดับเปรียบเทียบวัตถุทีละคู่ให้เป็นการประมวลผลแบบขนานสามารถเปรียบเทียบวัตถุหลาย ๆ คู่ในเวลาเดียวกันการดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุสำหรับฐานข้อมูลเชิงพื้นที่นั้นได้ให้ความสำคัญกับความเร็วในการประมวลผลในงานวิจัยนี้เพิ่มส่วนการทำดัชนีข้อมูลให้กับวัตถุที่จะนำมาเปรียบเทียบกันโดยใช้เทคนิคอาร์-ทรีโดยสร้างดัชนีให้กับวัตถุโดยใช้วิธีการตีกรอบสี่เหลี่ยมที่เล็กที่สุดที่ครอบคลุมวัตถุที่สนใจโดยที่ก่อนที่จะนำวัตถุสองวัตถุมาเปรียบเทียบกันจะเปรียบเทียบจากกรอบสี่เหลี่ยมที่อยู่ในระดับที่เป็นโหนดรากก่อนเมื่อพบว่าโหนดรากนั้นเกิดการทับซ้อนกันแสดงว่ามีแนวโน้มที่โหนดลูกจะเกิดการทับซ้อนกันด้วย ในทางกลับกันถ้าไม่เกิดการทับซ้อนกันจะแสดงให้เห็นว่าโหนดลูกจะไม่มีโอกาสที่จะทับซ้อนกันด้วย

ในการประมวลผลแบบขนานจะใช้การเขียนคำสั่งบนหน่วยประมวลผลกราฟิกบน CUDA C ข้อจำกัดของการเขียนโปรแกรมดังกล่าวคือยังต้องทำงานร่วมกันระหว่างหน่วยประมวลผลกลางกับหน่วยประมวลผลกราฟิก การเตรียมข้อมูลและการสร้างดัชนีข้อมูลทำบนหน่วยประมวลผลกลางเมื่อกระบวนการในการสร้างความสัมพันธ์ระหว่างวัตถุถูกเรียกใช้งานข้อมูลที่เกี่ยวข้องจะถูกส่งไปยังหน่วยประมวลผลกราฟิกและกระจายการทำงานอยู่บนแต่ละเทรดของหน่วยประมวลผลกราฟิกเพื่องเปรียบเทียบพิกัดจุดระหว่างวัตถุหลาย ๆ คู่ไปในเวลาเดียวกันซึ่งแต่ละเทรดจะทำงานคำสั่งเดียวกันในเวลาเดียวกันแต่จะใช้ข้อมูลพิกัดจุดของวัตถุคนละชุดกัน เมื่อประมวลผลคำสั่งเปรียบเทียบวัตถุเสร็จสิ้นแล้วจะส่งผลลัพธ์กลับมายังหน่วยประมวลผลกลางการประมวลผลคำสั่งเปรียบเทียบนี้จะดำเนินการโดยใช้เทคนิคอาร์-ทรีร่วมด้วยโดยเปรียบเทียบวัตถุตั้งแต่ระดับที่เป็นโหนดรากไปจนถึงโหนดลูกกรณีโหนดรากใด ๆ ที่ไม่ตรงตามเงื่อนไขก็จะไม่ถูกประมวลผลต่อ ขั้นตอนวิธีที่นำเสนอแบ่งการทำงานออกเป็นดังนี้

3.1 ออกแบบโครงสร้างและเตรียมข้อมูลใช้สำหรับการประมวลผล

3.2 สร้างดัชนีข้อมูลให้กับวัตถุ

3.3 การดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุบนหน่วยประมวลผลกลาง

3.4 การดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุบนหน่วยประมวลผลกราฟิก

นิยาม

min_x คือ พิกัดของแกน X ที่น้อยที่สุด

min_y คือ พิกัดของแกน Y ที่น้อยที่สุด

max_x คือ พิกัดของแกน X ที่มากที่สุด

max_y คือ พิกัดของแกน Y ที่มากที่สุด

R คือ ชุดข้อมูลพิกัดจุดของวัตถุที่หนึ่ง

S คือ ชุดข้อมูลพิกัดจุดของวัตถุชุดที่สอง

RootR คือ โหนดรากของชุดข้อมูลวัตถุที่หนึ่ง

RootS คือ โหนดรากของชุดข้อมูลวัตถุที่สอง

M คือ จำนวนโหนดมากที่สุด

m คือ จำนวนโหนดน้อยที่สุด

3.1 ออกแบบโครงสร้างและเตรียมข้อมูลใช้สำหรับการประมวลผล

งานวิจัยนี้มีข้อสมติฐานว่ามีการสร้างกรอบสี่เหลี่ยมเล็กที่สุดที่ครอบคลุมวัตถุไว้ก่อนแล้ว ข้อมูลดังกล่าวจะประกอบด้วยพิกัด min_x, min_y, max_x, และ max_y ข้อมูลเก็บไว้ในรูปของแฟ้มข้อมูลและจะถูกอ่านมาใส่ในตัวแปรแถวลำดับที่มีรูปแบบเป็นโครงสร้างบนข้อกำหนดของภาษาซี โดยชุดแถวข้อมูลแบ่งออกเป็น 2 ชุด คือ ชุดแถวข้อมูลที่เป็นระดับโหนดรากเก็บพิกัดจุดของโหนดรากและเก็บดัชนีชี้ไปยังโหนดลูกที่โหนดรากนั้นครอบคลุมอยู่ และชุดแถวข้อมูลที่เป็นพิกัดจุดของวัตถุ การออกแบบโครงสร้างข้อมูลมีรูปแบบดังนี้

3.1.1 โครงสร้างข้อมูลโหนดราก

```
Struct MBR_root {
        int min_x,max_x,min_y,max_y;
        MBR_object child[numberOfchild];
};
/*x, y coordinate rectangle of root*/

MBR_root  rootR [numberOfrootR];
MBR_root  rootS [numberOfrootS];
/*Array of rootR and rootS relation*/
```

ภาพที่ 3.1 การประกาศตัวแปรโครงสร้างข้อมูลโหนดราก

3.1.2 โครงสร้างข้อมูลโหนดลูก

```
Struct MBR_object {
        int min_x,max_x,min_y,max_y;
};
/*x, y coordinate rectangle of object*/
MBR_object  objectR [numberOfobjectR];
MBR_object  objectS [numberOfobjectS];
/*Array of objectR and objectS relation*/
```

ภาพที่ 3.2 การประกาศตัวแปรโครงสร้างข้อมูลของวัตถุ

3.2 สร้างดัชนีข้อมูลให้กับวัตถุโดยใช้เทคนิคอาร์-ทรี

อาร์-ทรีเป็นเทคนิคในการสร้างดัชนีข้อมูลเช่นเดียวกันกับบี-ทรีแต่ต่างกันที่ลักษณะของข้อมูลอาร์-ทรีใช้สำหรับข้อมูลเชิงพื้นที่ที่มีลักษณะเป็นรูปร่างที่มีหลายมิติในงานวิจัยนี้ใช้หลักการสร้างดัชนีด้วยวิธี On Packing R-tree [13] เป็นการสร้างดัชนีข้อมูลจากระดับโหนดลูกขึ้นไปยังโหนดราก (Bottom up) วิธีการนี้จะแบ่งออกเป็น 5 ขั้นตอนดังนี้

1) หาจำนวนวัตถุทั้งหมดโดยใช้คำสั่งในการนับแถวข้อมูลทั้งหมด

2) คำนวณจำนวนวัตถุที่จะบรรจุเข้าไปในแพ็คโดยต้องระบุจำนวนของโหนดรากที่ต้องการก่อนแล้วดำเนินการหาจำนวนโหนดลูกได้ดังนี้

จำนวนวัตถุใน 1 กลุ่ม = จำนวนแถวข้อมูล/จำนวนโหนดรากที่ต้องการ

ตัวอย่าง มีวัตถุอยู่ 100 วัตถุกำหนดให้มีจำนวนโหนดรากหรือกลุ่มอยู่ 10 กลุ่ม ดังนั้นจะหาจำนวนวัตถุต่อกลุ่มได้ดังนี้

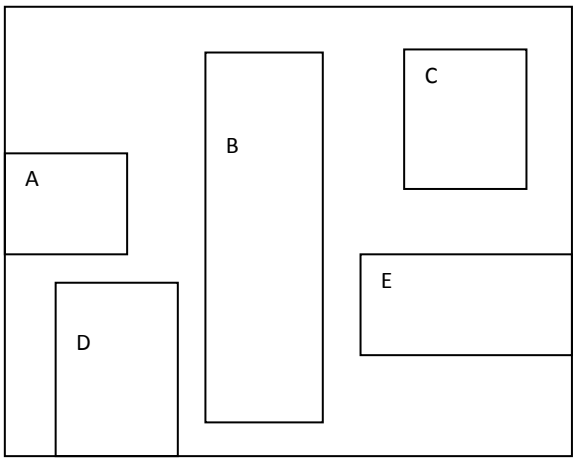
จำนวนวัตถุใน 1 กลุ่ม = 100/10 = 10 วัตถุต่อกลุ่ม

โดยข้อกำหนดของการสร้างอาร์-ทรีกำหนดให้จะต้องมีจำนวนโหนดลูกในกลุ่มอย่างน้อย (m) และไม่มากกว่า (M) เช่นจากตัวอย่าง กำหนดให้ m=1 และ M = 10 หมายความว่าในโหนดรากจะต้องมีโหนดลูกอย่างน้อย 1 โหนดและไม่มากกว่า 10 โหนด

3) เรียงข้อมูลพิกัดจุดของวัตถุในพิกัด min_x หรือ min_y อย่างใดอย่างหนึ่งเพื่อใช้ในเวลาที่บรรจุวัตถุลงในโหนดรากจะได้วัตถุที่มีพิกัดใกล้เคียงกันอยู่ในโหนดรากเดียวกันและเพื่อไม่ให้วัตถุหนึ่งวัตถุไปอยู่ในโหนดรากที่มากกว่าหนึ่งโหนด

4) ดำเนินการบรรจุวัตถุลงในโหนดรากตามลำดับที่ได้จัดเรียงไว้

5) เมื่อครบจำนวนในกลุ่มแล้วให้ทำการปรับปรุงพิกัดของโหนดรากและปรับปรุงข้อมูลดัชนีที่ชี้ไปยังโหนดลูกที่ครอบคลุมอยู่



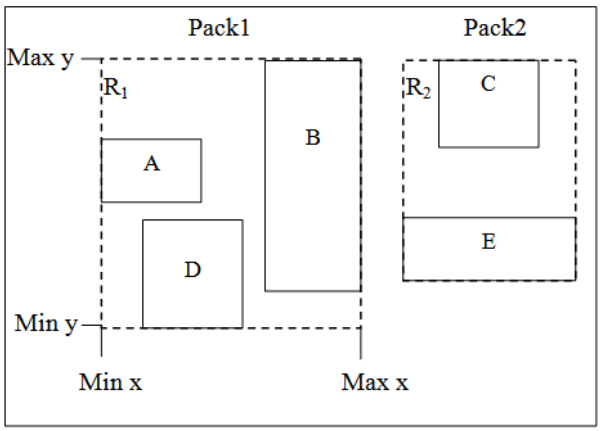
ภาพที่ 3.3 กรอบสี่เหลี่ยมของวัตถุก่อนทำการแบ่งโหนด

จากภาพที่ 3.1 ประกอบด้วยวัตถุ 5 วัตถุกำหนดให้โหนดรากมี 2 โหนดดังนั้นโหนดราก แต่ละโหนดจะมีจำนวนวัตถุ 3 วัตถุ จากนั้นนำวัตถุมาจัดเรียงตามค่า min_x จะได้สมาชิกของแถวข้อมูลดังนี้ { A, D, B, E, C } หลังจากนั้นบรรจุวัตถุเข้าสู่โหนดรากจนครบตามจำนวนจะได้

Pack 1 = {A, D, B}

Pack 2 = {C, E}

ขั้นตอนต่อไปเป็นการปรับปรุงค่าพิกัดของโหนดรากจะดำเนินการหาค่า min_x, min_y, max_x และ max_y ของวัตถุในแต่ละกลุ่มมาปรับปรุงเป็นค่าพิกัดของโหนดราก



ภาพที่ 3.4 กรอบสี่เหลี่ยมของวัตถุหลังทำการแบ่งโหนด

จากรูป 3.2 หลังการแบ่งโหนดจะประกอบไปด้วย Pack 1 มีโหนดราก R1 มีโหนดลูกที่ครอบคลุมอยู่คือ A, D, B และ Pack 2 มีโหนดราก R2 มีโหนดลูกที่ครอบคลุมอยู่คือ C, E

กระบวนการในการสร้างอาร์-ทรีนั้นจะทำบนหน่วยประมวลผลกลางในการเชื่อมความสัมพันธ์ระหว่างวัตถุจะต้องประกอบไปด้วยชุดข้อมูล 2 ชุดมาดำเนินการเชื่อมความสัมพันธ์ตามเงื่อนไขที่กำหนดดังนั้นทั้งสองชุดข้อมูลต้องดำเนินการสร้างอาร์-ทรีด้วยวิธีการเดียวกันและไม่

ว่าจะเป็นการประมวลผลแบบลำดับและแบบขนานก็ต้องดำเนินการขั้นตอนนี้ให้เสร็จสิ้นก่อนดำเนินการประมวลผลส่วนของการเชื่อมความสัมพันธ์ระหว่างวัตถุดังนั้นในการวัดเวลาในการประมวลผลจึงไม่รวมในส่วนของการสร้างอาร์-ทรีด้วยเนื่องจากทั้งสองวิธีจะต้องดำเนินการขั้นตอนนี้เช่นเดียวกัน

กระบวนการต่อไปเป็นการเชื่อมความสัมพันธ์ระหว่างวัตถุในงานวิจัยนี้สนใจการเชื่อมความสัมพันธ์ระหว่างวัตถุด้วยเงื่อนไขการหาพิกัดพื้นที่ระหว่างวัตถุที่มีการทับซ้อนกัน (Intersection join) เพื่อเป็นการเปรียบเทียบความเร็วในการประมวลผลดังนั้นงานวิจัยนี้จึงได้ออกแบบระเบียบวิธีและเขียนคำสั่งในการประมวลผลดังกล่าวทั้งแบบลำดับบนหน่วยประมวลผลกลางและแบบขนานบนหน่วยประมวลผลกราฟิกมีวิธีการดังนี้

3.3 การดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุบนหน่วยประมวลผลกลาง

เนื่องจากลักษณะในการประมวลผลการเชื่อมความสัมพันธ์ระหว่างวัตถุเป็นการเปรียบเทียบพิกัดวัตถุสองวัตถุว่าตรงตามเงื่อนไข ซึ่งข้อมูลตัวอย่างอยู่ในรูปของชุดข้อมูลประกอบไปด้วยหลาย ๆ วัตถุในชุดข้อมูลเดียวกันและในการเชื่อมความสัมพันธ์ต้องดำเนินการเปรียบเทียบระหว่างชุดข้อมูลสองชุด ดังนั้นในการทำงานแบบลำดับทั่วไปจะมองการเปรียบเทียบดังกล่าวอยู่ในรูปการทำงานแบบวนลูปซึ่งชุดข้อมูลที่หนึ่งจะอยู่ในลูปนอก และมีอีกชุดข้อมูลหนึ่งอยู่ในลูปในการตรวจสอบเงื่อนไขมีการวนรอบเพื่อเปรียบเทียบระหว่างข้อมูลสองชุด ให้ครบทุก ๆ วัตถุโดยใช้ดัชนีของข้อมูลในการอ้างถึง

1) Overlap ในขั้นตอนนี้จะดำเนินการหาวัตถุที่มีการทับซ้อนกันในระดับที่เป็นโหนดราก ในกระบวนการนี้เป็นขั้นตอนการกรองข้อมูลโดยใช้เงื่อนไขดังนี้

```
For i
        For j
                If overlap(Ri, Sj) then
                  r = child of (Ri)
                  s = child of (Sj)
                If overlap(r,s) then report(r,s)
```

ภาพที่ 3.5 รหัสเทียมระเบียบวิธีการหาวัตถุที่มีการทับซ้อนกันแบบลำดับ

2) Find child ข้อมูลของโหนดรากนอกจากจะเก็บพิกัดกรอบสี่เหลี่ยมของโหนดรากแล้วยังเก็บข้อมูลของดัชนีที่ชี้ไปหาโหนดลูกที่ครอบอยู่ ในขั้นตอนนี้จะอ่านข้อมูลดัชนีที่ชี้ไปยังโหนดลูก

ขึ้นมาและส่งผลลัพธ์ที่ได้เป็นข้อมูลนำเข้าสำหรับขั้นตอนในการหาวัตถุที่มีการทับซ้อนกันในระดับโหนดลูกต่อไปเป็นการเรียกใช้ฟังก์ชันแบบเรียกซ้ำ (Recursive function)

3.4 การดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุบนหน่วยประมวลผลกราฟิก

1) Overlap ในขั้นตอนนี้จะดำเนินการหาวัตถุที่มีการทับซ้อนกันในระดับที่เป็นโหนดรากในกระบวนการนี้เป็นขั้นตอนการกรองข้อมูลโดยใช้เงื่อนไขดังนี้

```
overlap((R, S) return TRUE, FALSE is
    Load MBR of R, S
        If(((Sj.x_min < Ri.x_max)
            and (Sj.x_max > Ri.x_min)
            and (Sj.y_min < Ri.y_max)
            and (Sj.y_max > Ri.y_min))
            then return TRUE
        Else return FALSE
```

ภาพที่ 3.6 รหัสเทียมระเบียบวิธีการหาวัตถุที่มีการทับซ้อนกัน

2) Find child ข้อมูลของโหนดรากนอกจากจะเก็บพิกัดกรอบสี่เหลี่ยมของโหนดรากแล้วยังเก็บข้อมูลของดัชนีที่ชี้ไปหาโหนดลูกที่ครอบอยู่ ในขั้นตอนนี้จะอ่านข้อมูลดัชนีที่ชี้ไปยังโหนดลูกขึ้นมาและส่งผลลัพธ์ที่ได้เป็นข้อมูลนำเข้าสำหรับขั้นตอนในการหาวัตถุที่มีการทับซ้อนกันในระดับโหนดลูกต่อไป

ในงานวิจัยนี้สนใจเชื่อมความสัมพันธ์ระหว่างวัตถุเงื่อนไขหาพิกัดของวัตถุที่มีการทับซ้อนกันดังนั้นในกระบวนการเชื่อมความสัมพันธ์ระหว่างวัตถุในระดับโหนดรากและโหนดลูก จะใช้เงื่อนไขเดียวกันคือหาพิกัดวัตถุที่ทับซ้อนกันแต่จะต่างกันที่ข้อมูลนำเข้าที่ใช้ในการประมวลผล

4) การแปลงการประมวลผลจากลำดับเป็นแบบขนาน ในการดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุนั้นเป็นการเปรียบเทียบหาพิกัดจุดที่ตรงตามเงื่อนไขของวัตถุดังนั้นงานที่สามารถทำแบบขนานให้สามารถประมวลผลไปพร้อม ๆ กันในเวลาเดียวกันได้นั้นเป็นส่วนของการดำเนินการเปรียบเทียบพิกัดจุดระหว่างสองวัตถุซึ่งวัตถุที่ต้องประมวลผลนั้นประกอบไปด้วยหลายวัตถุด้วยกัน ในงานวิจัยนี้จึงได้มีแนวคิดในการกระจายงานในส่วนของการเปรียบเทียบพิกัดจุดระหว่างวัตถุ โดยวัตถุแต่ละคู่จะถูกแตกงานออกไปในแต่ละเทรดบนหน่วยประมวลผลกราฟิกโดยใช้ดัชนีของชุดแถวข้อมูลในการอ้างถึงวัตถุแต่ละวัตถุ จากการประมวลผลแบบลำดับในการเปรียบเทียบใช้การวนลูปออกเป็น 2 ลูปโดยลูปนอกจะเป็นข้อมูลชุดที่หนึ่งและลูปในเป็นข้อมูลชุดที่สองในการเปรียบเทียบจะวนลูปข้อมูลชุดในให้หมดก่อนถึงจะดำเนินการเลื่อนค่าดัชนีลูปนอก

แนวคิดในการแปลงวิธีการจากการประมวลผลแบบลำดับเป็นแบบขนานในส่วนการประมวลผล การเปรียบเทียบวัตถุจากลักษณะของหน่วยประมวลผลกราฟิกมีดังนี้

	TID0	TID1	…	TIDN
Block0	(0,0)	(0,1)	…	(0,N)
Block1	(1,0)	(1,1)	…	(1,N)
...	…	…	…	…
BlockM	(M,0)	(M,1)	…	(M,N)

ภาพที่ 3.7 ลักษณะการกระจายเทรดของหน่วยประมวลผลกราฟิก

	ObjS0	ObjS1	…	ObjSN
ObjR0	(R0,S0)	(R0,S1)	…	(R0,SN)
ObjR1	(R1,S0)	(R1,S1)	…	(R1,SN)
...	…	…	…	…
ObjRN	(RM,S0)	(RM,S1)	…	(RM,SN)

ภาพที่ 3.8 การกระจายงานลงแต่ละเทรด

จากภาพที่ 3.6 และ 3.7แสดงลักษณะการกระจายเทรดของหน่วยประมวลผลกราฟิกโดยการอ้างถึงข้อมูลโดยใช้หมายเลขบล็อกแทนการอ้างถึงข้อมูลลูปนอก และใช้หมายเลขเทรดแทนการอ้างถึงข้อมูลลูปใน โดยในแต่ละเทรดสามารถประมวลผลได้พร้อม ๆ กันในเวลาเดียวกัน

5) การระบุจำนวนเทรดที่ใช้ในการประมวลผล

ในการดำเนินการกระจายการทำงานไปยังเทรดต่าง ๆ โดยการแปลงลูปแบบลำดับไปเป็นแบบขนานโดยการใช้ประโยชน์จากลักษณะการกระจายงานของหน่วยประมวลผลกราฟิกนั้นจะใช้หมายเลขบล็อกและหมายเลขเทรดในการอ้างถึงข้อมูลที่จะนำมาประมวลผลดังแสดงในภาพที่ 3.6 และ 3.7 เมื่อมีการเรียกใช้คำสั่งการประมวลผลบนหน่วยประมวลผลกราฟิกจะต้องมีการระบุจำนวนบล็อกและเทรดที่ใช้ในการประมวลผล ในงานวิจัยนี้

จะระบุจำนวนบล็อกและเทรดตามจำนวนของวัตถุโดยที่ชุดข้อมูลวัตถุที่หนึ่งแทนด้วยบล็อกและชุดข้อมูลของวัตถุที่สองแทนด้วยเทรดดังนี้

Function<<<number of outer loop, number of inner loop>>>

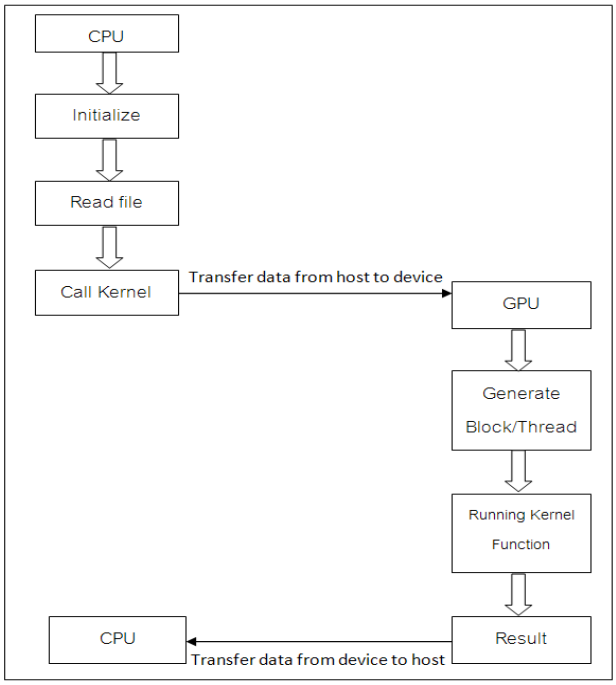
ตัวอย่าง กรณีมีชุดข้อมูลที่หนึ่ง 100 วัตถุและชุดข้อมูลที่สองมี 100 วัตถุในการระบุจำนวนเทรดสามารถทำได้ดังนี้

Function<<<100, 100>>>

การกระจายการประมวลผลออกเป็น 100 บล็อกและ 100 เทรดดังนั้นจำนวนเทรดที่ใช้ทำงานจะมีทั้งสิ้น 1,000 เทรดซึ่งจำนวนเทรดที่สามารถใช้ในการประมวลผลได้สูงสุดเท่าใดนั้นขึ้นอยู่กับประสิทธิภาพของหน่วยประมวลผลกราฟิกในแต่ละรุ่นด้วย

6) กระบวนการการประมวลผลบนหน่วยประมวลผลกราฟิก

เนื่องด้วยข้อจำกัดของหน่วยประมวลผลกราฟิกที่การทำงานต้องมีการประสานกันระหว่างหน่วยประมวลผลกลาง ในส่วนของการอ่านข้อมูล การกำหนดค่าพื้นฐาน จากนั้นจะส่งต่อข้อมูลไว้จากหน่วยประมวลผลกลางต่อไปยังหน่วยประมวลผลกราฟิก เมื่อมีการเรียกใช้คำสั่งหน่วยประมวลผลกราฟิกจะดำเนินการกระจายการทำงานไปยังเทรดบนหน่วยประมวลผลกราฟิกเมื่อประมวลผลเสร็จสิ้นแล้วหน่วยประมวลผลกราฟิกจะส่งต่อผลลัพธ์กลับมายังหน่วยประมวลผลกลาง



ภาพที่ 3.9 ขั้นตอนการประสานการทำงานระหว่างหน่วยประมวลผลกลางและหน่วยประมวลผลกราฟิก

บทที่ 4
สรุปผลการทดลอง

4.1 การทดลอง

4.1.1 คุณสมบัติของหน่วยประมวลผลที่ใช้ในการทดลอง

เพื่อทำการวัดเวลาในการประมวลผลเปรียบเทียบระหว่างแบบลำดับบนหน่วยประมวลผลกลางกับแบบขนานบนหน่วยประมวลผลกราฟิกจึงได้มีการเขียนคำสั่งทั้งแบบลำดับและแบบขนาน โดยแบบลำดับจะใช้หน่วยประมวลผลกลางรุ่น Intel Core i3 2.93 GHz หน่วยความจำ 2 GB แบบขนานจะใช้หน่วยประมวลผลกราฟิกรุ่น NVIDIA GT440 1092 MHz หน่วยความจำ 1 GB CUDA 96 Cores ซึ่งบนหน่วยประมวลผลกลางจะใช้ภาษาซี และบนหน่วยประมวลผลกราฟิกจะใช้ CUDA C

4.1.2 ชุดข้อมูลที่ใช้ในการทดลอง

ในงานวิจัยนี้มีข้อสมมุติฐานว่ามีการคำนวณในขั้นตอนการประมาณการวัตถุด้วยการหา Minimum bounding rectangles ไว้ก่อนแล้วชุดข้อมูลที่เก็บจะอยู่ในรูปของแฟ้มข้อความก่อนที่จะประมวลผลจะมีการอ่านข้อมูลจากแฟ้มขึ้นมาเก็บไว้ในตัวแปรแถวลำดับก่อนข้อมูลดังกล่าวเป็นข้อมูลที่ใช้สำหรับงานวิจัยทางด้านฐานข้อมูลเชิงพื้นที่โดยเฉพาะ [14] ชุดข้อมูลที่เลือกมาใช้ในการทดลองนั้นมีดังนี้

Pair of dataset	Amount MBRs	Data type	Data size
Greece			
Rivers join Roads	47,918	Float	0.7 MB
Germany			
Streets join Rail roads	67,008	Integer	0.6 MB

ตารางที่ 4.1 ชุดข้อมูลที่ใช้สำหรับการทดลอง

ชุดข้อมูลที่ 1 เป็นชุดข้อมูลพิกัดของแม่น้ำกับถนนในประเทศกรีซมีจำนวนวัตถุรวมกัน 47,918 วัตถุที่ต้องนำมาเปรียบเทียบกันและมีขนาดของข้อมูล 0.7 MB ข้อมูลชุดที่ 2 เป็นชุดข้อมูลพิกัดของถนนกับทางรถไฟในประเทศเยอรมันมีจำนวนวัตถุรวมกัน 67,008 วัตถุที่ต้องนำมาเปรียบเทียบกันและมีขนาดของข้อมูล 0.6 MB จากจำนวนของวัตถุและขนาดของข้อมูลจะเห็นได้ว่าสามารถนำไปประมวลผลได้ทั้งบนหน่วยประมวลผลกลางและหน่วยประมวลผลกราฟิก

โดยไม่มีปัญหาในแง่ของขนาดของหน่วยความจำหรือขนาดของตัวแปรแถวลำดับที่จะดำเนินการในการประมวลผล

4.2 สรุปผลการทดลอง

ในการวัดระยะเวลาในการประมวลผลนั้นจะไม่ได้รวมถึงขั้นตอนในการอ่านข้อมูลจากแฟ้ม และไม่รวมถึงขั้นตอนในการสร้างต้นไม้เนื่องด้วยทั้งการประมวลผลแบบลำดับและแบบขนานจะต้องเสียเวลาในส่วนนี้เช่นเดียวกัน ในการวัดเวลาในการประมวลผลนั้นจะเริ่มวัดเวลาในส่วนของการประมวลผลเชื่อมความสัมพันธ์ระหว่างวัตถุในส่วนการเปรียบเทียบหาพิกัดวัตถุที่มีการทับซ้อนกันทั้งในการเปรียบเทียบในระดับที่เป็นโหนดราก และส่วนที่เป็นโหนดลูก

4.2.1 ผลการทดลองบนหน่วยประมวลผลกลาง

ในการวัดเวลาในการประมวลผลในหน่วยประมวลผลกลางนั้นจะใช้คำสั่งการวัดเวลาในการประมวลผลของภาษาซีคำสั่งเริ่มจับเวลาจะถูกใส่ไว้ก่อนเรียกใช้ฟังก์ชันและคำสั่งหยุดจับเวลาเมื่อสิ้นสุดฟังก์ชันในการจับเวลานั้นหน่วยที่ได้จะเป็นมิลลิวินาที

```
clock_t start, stop, start2, stop2;
double t = 0.0;
double t2 = 0.0;
/* Start timer */
assert((start = clock())!=-1);
//start function

stop = clock();
//stop function
t = (double) (stop-start)/CLOCKS_PER_SEC;
//calculate execution time
```

ภาพที่ 4.1 คำสั่งในการวัดเวลาการประมวลผลภาษาซี

ในการจับเวลาจะแยกระหว่างฟังก์ชันที่เปรียบเทียบวัตถุของโหนดรากและโหนดลูกแยกจากกันด้วยเหตุผลที่จำนวนวัตถุของโหนดรากจะมีน้อยกว่าจำนวนวัตถุของโหนดลูกซึ่งจำนวนวัตถุที่นำมาเปรียบเทียบนั้นส่งผลกระทบต่อเวลาในการประมวลผลผลลัพธ์ที่ได้จากการประมวลผลบนหน่วยประมวลผลแบบลำดับมีดังนี้

Pair of dataset	Amount MBRs	At root (ms)	At child (ms)	Total (ms)
Greece				
Rivers join Roads	47,918	18	72.67	90.67
Germany				
Streets join Rail roads	67,008	5.33	74.00	79.33

ตารางที่ 4.2 ผลการทดลองบนหน่วยประมวลผลกลาง

ผลจากการทดลองในการประมวลผลแบบลำดับของโหนดรากในชุดข้อมูลที่หนึ่งใช้เวลาในการประมวลผล 18 มิลลิวินาที โหนดลูกใช้เวลาในการประมวลผล 72.67 มิลลิวินาที เวลารวม 90.67 มิลลิวินาที ส่วนในชุดข้อมูลที่สองใช้เวลาในการประมวลผลโหนดราก 5.33 มิลลิวินาทีโหนดลูกใช้เวลาในการประมวลผล 74.00 มิลลิวินาที เวลารวม 79.33 มิลลิวินาที

4.2.2 ผลการทดลองบนหน่วยประมวลผลกราฟิก

ในการวัดเวลาการประมวลผลบนหน่วยประมวลผลกราฟิกนั้นใช้คำสั่งการวัดเวลาในการประมวลผลของ CUDA C คำสั่งเริ่มจับเวลาตั้งแต่เวลาเริ่มส่งข้อมูลจากหน่วยประมวลผลกลางไปยังหน่วยประมวลผลกราฟิก เวลาในการเรียกใช้ฟังก์ชันในส่วนที่เป็นเคอร์เนลจนกระทั่งประมวลผลเสร็จและส่งผลลัพธ์กลับมายังหน่วยประมวลผลกลางจึงสิ้นสุดระยะเวลาในการประมวลผลในการจับเวลานั้นหน่วยที่ได้จะเป็นมิลลิวินาที

```
/* Start timer */
cudaEventCreate(&start);
cudaEventCreate(&stop);
cudaEventRecord(start,0);
//Transfer data host to device
//start function
kernel<<<blocks,threads>>>
//stop function
//Transfer data device to host
cudaEventRecord( stop, 0 );
cudaEventSynchronize( stop );
float   elapsedTime;
cudaEventElapsedTime( &elapsedTime,start, stop );
cudaEventDestroy( start );
cudaEventDestroy( stop );
//calculate execution time
```

ภาพที่ 4.2 คำสั่งในการวัดเวลาการประมวลผล CUDA C

Pair of dataset	Amount MBRs	At root (ms)	At child (ms)	Total (ms)
Greece				
Rivers join Roads	47,918	4	22.33	26.33
Germany				
Streets join Rail roads	67,008	4	39.67	43.67

ตารางที่ 4.3 ผลการทดลองบนหน่วยประมวลผลกราฟิก

ผลจากการทดลองในการประมวลผลแบบขนานของโหนดรากในชุดข้อมูลที่หนึ่งใช้เวลาในการประมวลผล 4 มิลลิวินาที โหนดลูกใช้เวลาในการประมวลผล 22.33 มิลลิวินาที เวลารวม 26.33 มิลลิวินาที ส่วนในชุดข้อมูลที่สองใช้เวลาในการประมวลผลโหนดลูก 4 มิลลิวินาที โหนดลูกใช้เวลาในการประมวลผล 39.67 มิลลิวินาที เวลารวม 43.67 มิลลิวินาที

บนหน่วยประมวลผลกราฟิกนั้นนอกจากมีการทำงานแบบขนานช่วยเพิ่มความเร็วในการประมวลผลยังสามารถเพิ่มส่วนของการเข้าถึงข้อมูลบนหน่วยประมวลผลกราฟิกเพื่อให้การเข้าถึงข้อมูลเร็วขึ้นโดยสามารถปรับการเข้าถึงข้อมูลบน Global memory เป็นบน Local memory เพื่อลดระยะเวลาการรอข้อมูลที่เข้าใช้งานหลายเทรดพร้อมกันโดยแทนที่ทุก ๆ เทรดจะเข้ามาอ่านข้อมูลที่เดียวกันก็ปรับเป็นการนำข้อมูลไปไว้บน Local memory ของแต่ละบล็อกและให้แต่ละเทรดอ่านข้อมูลภายในบล็อกของตนเองจากการปรับการเข้าถึงข้อูลดังกล่าวสามารถเพิ่มความเร็วในการประมวลผลได้ดังนี้

Pair of dataset	At root (ms)		At child (ms)		Total (ms)		Speed up Global &Share
	Global memory	*Share memory*	*Global memory*	*Share memory*	*Global memory*	*Share memory*	
Greece							
Rivers join Roads	4.27	4.19	27.32	22.97	31.59	27.16	x1
Germany							
Streets join Rail roads	4.3	4.43	40.63	39.78	44.93	44.21	x1

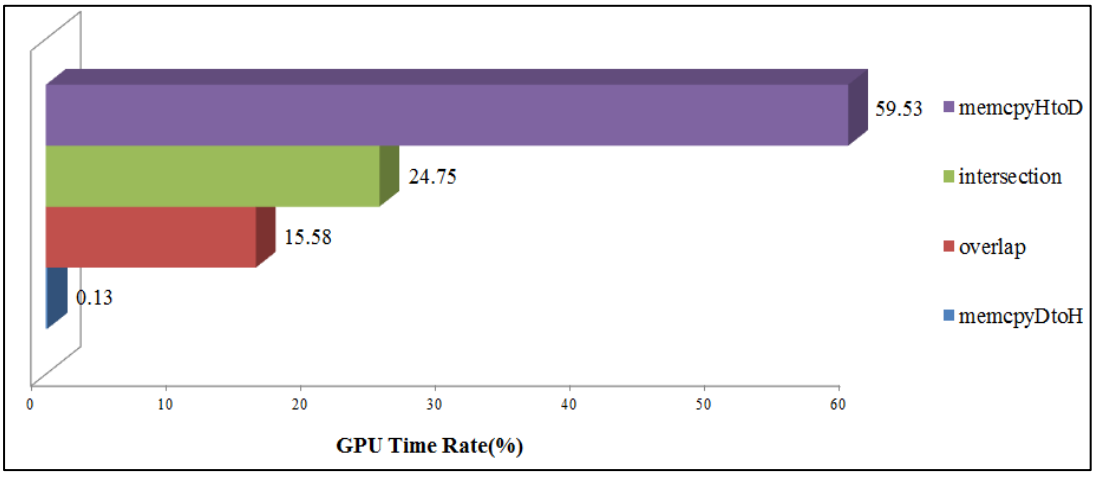
ตารางที่ 4.4 ผลการเปรียบเทียบระหว่าง Global memory และ Share memory

4.2.3 การเปรียบเทียบระยะเวลาในการประมวลผลระหว่างบนหน่วยประมวลผลกลางและหน่วยประมวลผลกราฟิก

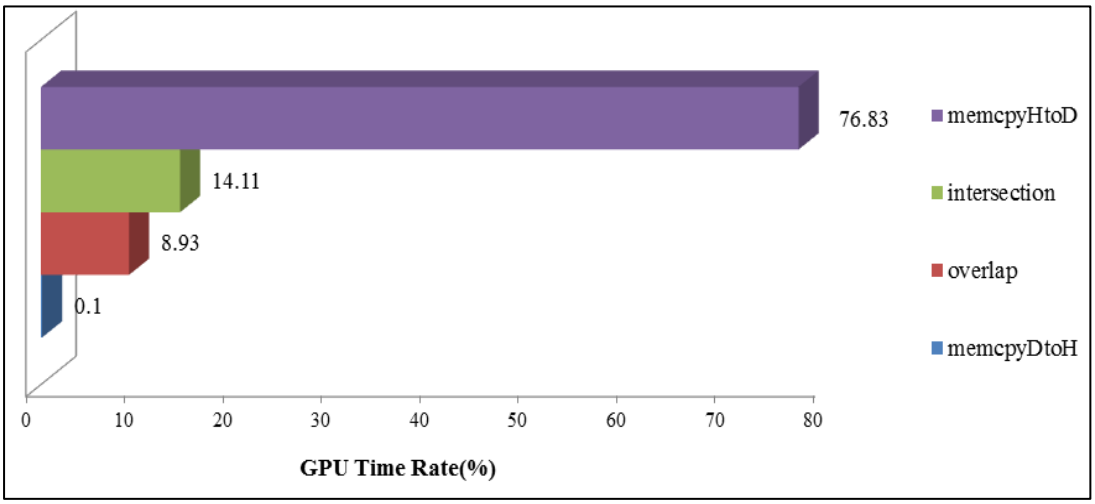
Pair of dataset	At root (ms)			At child (ms)			Total (ms)		
	CPU	*GPU Global*	*GPU Share*	*CPU*	*GPU Global*	*GPU Share*	*CPU*	*GPU Global*	*GPU Share*
Greece									
Rivers join Roads	18	4.27	4.19	72.67	27.32	22.97	90.67	31.59	27.16
Germany									
Streets join Rail roads	5.33	4.3	4.43	74.00	40.63	39.78	79.33	44.93	44.21

ตารางที่ 4.5 สรุปผลการเปรียบเทียบระยะเวลาการประมวลผล

เนื่องจากข้อจำกัดของหน่วยประมวลผลกราฟิกที่ยังต้องประสานการทำงานระหว่างหน่วยประมวลผลกลางในส่วนของการรับ-ส่งข้อมูล ดังนั้นระยะเวลาในการรับ-ส่งข้อมูลก็ส่งผลกับเวลาในการประมวลผลด้วย



ภาพที่ 4.3 อัตราของเวลาในการรับ-ส่งข้อมูลต่อเวลาในการประมวลผลทั้งหมดของชุดข้อมูลที่หนึ่ง



ภาพที่ 4.4 อัตราของเวลาในการรับ-ส่งข้อมูลต่อเวลาในการประมวลผลทั้งหมดของชุดข้อมูลที่สอง

จากชุดข้อมูลที่หนึ่งมีจำนวนวัตถุ 47,918 ขนาดของข้อมูล 0.7 MB เวลาที่ใช้ไปในส่วนของการส่งข้อมูลจากหน่วยประมวลผลกลางไปยังหน่วยประมวลผลกราฟิกคิดเป็นร้อยละ 59.53 ของเวลาในการประมวลผลทั้งหมด และใช้เวลาในการส่งผลลัพธ์กลับจากหน่วยประมวลผลกราฟิกกลับมายังหน่วยประมวลผลกลางคิดเป็นร้อยละ 0.13 ของเวลาในการประมวลผลทั้งหมด

จากชุดข้อมูลที่สองมีจำนวนวัตถุ 67,008 ขนาดของข้อมูล 0.6 MB เวลาที่ใช้ไปในส่วนของการส่งข้อมูลจากหน่วยประมวลผลกลางไปยังหน่วยประมวลผลกราฟิกคิดเป็นร้อยละ 76.83 ของเวลาในการประมวลผลทั้งหมด และใช้เวลาในการส่งผลลัพธ์กลับจากหน่วยประมวลผลกราฟิกกลับมายังหน่วยประมวลผลกลางคิดเป็นร้อยละ 0.1 ของเวลาในการประมวลผลทั้งหมด

เวลาในการรับ-ส่งข้อมูลระหว่างหน่วยประมวลผลกลางและหน่วยประมวลผลกราฟิกในการทดลองนั้น เวลาในการประมวลผลได้รวมส่วนของเวลาในการรับ-ส่งข้อมูลด้วย จากการทดลองจะเห็นว่าเวลาส่วนใหญ่จะเสียไปกับเวลาในการรับ-ส่งข้อมูลแต่อย่างไรก็ตาม ถึงแม้ว่าจะรวมเวลาในการรับ-ส่งข้อมูลแล้วหน่วยประมวลผลกราฟิกยังสามารถประมวลผลได้เร็วกว่าบนหน่วยประมวลผลกลาง

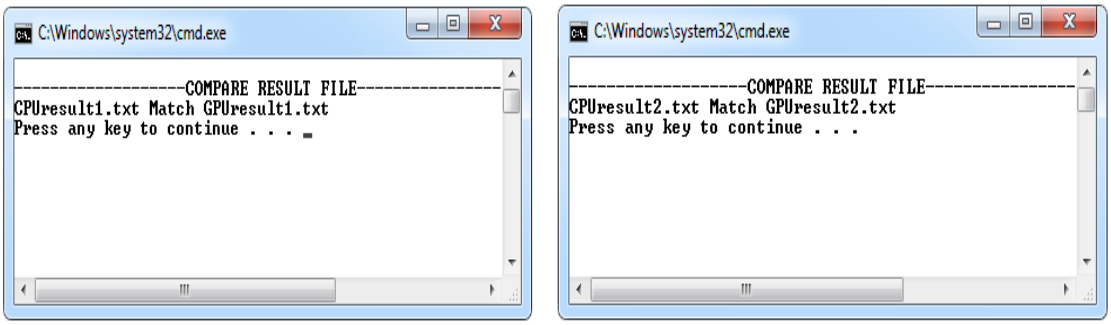
4.3 การตรวจสอบความถูกต้องของผลลัพธ์

การดำเนินการตรวจสอบความถูกต้องของผลลัพธ์นั้นได้เขียนคำสั่งให้ส่งผลลัพธ์เป็นแฟ้มที่มีลักษณะเป็นข้อความจะได้แฟ้มที่เป็นผลลัพธ์ 2 แฟ้มดังนี้ แฟ้มผลลัพธ์ที่ประมวลผลโดยหน่วย

ประมวลผลกลางและแฟ้มผลลัพธ์ที่ประมวลผลโดยหน่วยประมวลผลกราฟิก หลังจากนั้นทำการเขียนคำสั่งอ่านแฟ้มทั้งสองแฟ้มขึ้นมาแล้วทำการเปรียบเทียบกัน

```
int main ( void )
{
staticconstchar filename[] = "CPUresult.txt";
staticconstchar filename2[] = "GPUresult.txt";
char line[500];
   FILE *file = fopen ( filename, "r" );
if ( file != NULL ){
while ( fgets ( line, sizeof line, file ) != NULL ){
      }
            fclose ( file );
      }
else{
      perror ( filename );
   }
char line2[500];
   FILE *file2 = fopen ( filename2, "r" );
if ( file2 != NULL ){
while ( fgets ( line2, sizeof line2, file2 ) != NULL ){
      }
            fclose ( file2 );
      }
else{
      perror ( filename2 );
   }
int i=0;
while (i<1){
      if (strcmp(line,line2) == NULL){
            printf ("%s Not Match %s\n",filename,filename2);
      }
      else {
            printf ("%s Match %s\n",filename,filename2);
           }
      i++;
      }
      return 0;
}
```

ภาพที่ 4.5 คำสั่งในการอ่านข้อมูลจากแฟ้มเพื่อเปรียบเทียบผลลัพธ์



ภาพที่ 4.6 หน้าจอแสดงผลการตรวจสอบผลลัพธ์

จากผลการเปรียบเทียบแฟ้มผลลัพธ์ที่ได้จากการประมวลผลบนหน่วยประมวลผลกลางและหน่วยประมวลผลกราฟิกพบว่าผลลัพธ์ตรงกันทั้งสองชุดข้อมูล

สรุปผลการทดลองในการเปรียบเทียบเวลาในการประมวลผลในส่วนของการประมวลผลการเชื่อมความสัมพันธ์ระหว่างวัตถุสำหรับฐานข้อมูลเชิงพื้นที่บนหน่วยประมวลผลกราฟิกที่มีรูปแบบการประมวลผลแบบขนานซึ่งการประมวลผลแบบขนานนั้นจะดำเนินการเฉพาะในส่วนการเปรียบเทียบหาพิกัดของวัตถุที่เกิดการทับซ้อนกันพบว่าการเปรียบเทียบแบบขนานสามารถประมวลผลได้เร็วกว่าแบบลำดับ

จากการทดลองโดยใช้ Global memory เวลาในการประมวลผลจากชุดข้อมูลที่หนึ่งโหนดรากความเร็วในการประมวลผลเพิ่มขึ้น 4.2 เท่าโหนดลูกความเร็วในการประมวลผลเพิ่มขึ้น 2.7 เท่าเวลาในการประมวลผลรวมเพิ่มขึ้น 2.9 เท่า จากชุดข้อมูลที่สองโหนดรากความเร็วในการประมวลผลเพิ่มขึ้น 1.2 เท่าโหนดลูกความเร็วในการประมวลผลเพิ่มขึ้น 1.8 เท่าเวลาในการประมวลผลรวมเพิ่มขึ้น 1.8 เท่า

จากการทดลองโดยใช้ Share memory เวลาในการประมวลผลจากชุดข้อมูลที่หนึ่งโหนดรากความเร็วในการประมวลผลเพิ่มขึ้น 4.3 เท่าโหนดลูกความเร็วในการประมวลผลเพิ่มขึ้น 3.16 เท่าเวลาในการประมวลผลรวมเพิ่มขึ้น 3.34 เท่า จากชุดข้อมูลที่สองโหนดรากความเร็วในการประมวลผลเพิ่มขึ้น 1.2 เท่าโหนดลูกความเร็วในการประมวลผลเพิ่มขึ้น 1.9 เท่าเวลาในการประมวลผลรวมเพิ่มขึ้น 1.8 เท่า

ความเร็วในการประมวลผลจะขึ้นอยู่กับประเภทของข้อมูลด้วย จากผลการทดลองจะเห็นได้ว่าชุดข้อมูลที่หนึ่งมีจำนวนวัตถุที่น้อยกว่าจำนวนวัตถุของชุดข้อมูลที่สองแต่ใช้เวลาในการประมวลผลมากกว่าเนื่องจากชุดข้อมูลที่หนึ่งเป็นข้อมูลชนิดที่มีหน่วยทศนิยมมีจำนวนหลักของข้อมูลต้องนำมาเปรียบเทียบที่มากกว่าชุดข้อมูลที่สองจึงส่งผลให้ชุดข้อมูลที่หนึ่งใช้เวลาในการประมวลผลมากกว่า

บทที่ 5
ข้อเสนอแนะแนวทางการพัฒนาต่อ

5.1 ข้อเสนอแนะ

จากแนวคิดแก้ปัญหาการดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุสำหรับฐานข้อมูลเชิงพื้นที่ที่มีความต้องการที่จะเพิ่มความเร็วในการประมวลผล ซึ่งการดำเนินการ เชื่อมความสัมพันธ์ระหว่างวัตถุนั้นการทำงานหลักคือการหาพิกัดของวัตถุที่มีความสัมพันธ์ตรงตามเงื่อนไขที่กำหนดซึ่งหากมีจำนวนวัตถุจำนวนมากจะส่งผลกระทบต่อเวลาในการประมวลผล ในงานวิจัยนี้เสนอแนวทางในการเพิ่มความเร็วในการประมวลผลโดยใช้แนวคิดในการประมวลผลแบบขนานบนหน่วยประมวลผลกราฟิกในส่วนของการเปรียบเทียบพิกัดของวัตถุ นอกจากใช้การประมวลผลแบบขนานในการเปรียบเทียบพิกัดของวัตถุแล้วยังใช้เทคนิคการเข้าถึงข้อมูลแบบอาร์-ทรีมาช่วยลดจำนวนครั้งในการประมวลผลเพราะหากวัตถุมีจำนวนมากทำให้จำนวนครั้งในการเปรียบเทียบก็มากขึ้นด้วย

แต่ด้วยข้อจำกัดของหน่วยประมวลผลกราฟิกยังคงต้องมีการประสานการทำงานกับหน่วยประมวลผลกลางในส่วนงานที่หน่วยประมวลผลกลางมีความสามารถทำได้ดีกว่าส่วนที่ต้องประมวลผลบนหน่วยประมวลผลกลาง เช่น การอ่านข้อมูลจากแฟ้มเนื่องด้วยหน่วยประมวลผลกราฟิกยังไม่สามารถทำการอ่านเขียนข้อมูลจากหน่วยความจำสำรองได้ อีกงานหนึ่งที่ต้องใช้หน่วยประมวลผลกลางคือการสร้างดัชนีในการเข้าถึงข้อมูล การเปรียบเทียบการทำงานแบบลำดับกับแบบขนานจะต้องใช้โครงสร้างข้อมูลและการเข้าถึงแบบเดียวกันแต่จะต่างกันในส่วนของการดำเนินการเชื่อมความสัมพันธ์ระหว่างวัตถุและงานวิจัยนี้สนใจเพิ่มความเร็วในการประมวลผลในส่วนนี้เท่านั้น ดังนั้นขั้นตอนในการสร้างดัชนีจึงเสร็จสิ้นบนหน่วยประมวลผลกลางหลังจากนั้นกระบวนการในการเปรียบเทียบหาพิกัดที่มีความสัมพันธ์กันตรงตามเงื่อนไขที่กำหนด (โดยงานวิจัยนี้สนใจเงื่อนไขหาพิกัดของวัตถุที่ทับซ้อนกัน) โดยข้อมูลที่เกี่ยวข้องจะถูกส่งไปยังหน่วยประมวลผลกราฟิกเพื่อใช้ในการประมวลผลเมื่อประมวลผลเสร็จสิ้นผลลัพธ์จะถูกส่งกลับมายังหน่วยประมวลผลกลาง

จากผลการทดลองที่วัดเวลาในการประมวลผลส่วนการเปรียบเทียบวัตถุทั้งแบบลำดับบนหน่วยประมวลผลกลางและแบบขนาน พบว่าสามารถประมวลผลแบบขนานบนหน่วยประมวลผลกราฟิกได้และสามารถเพิ่มความเร็วในการประมวลผลโดยสรุปได้ดังนี้

จากการทดลองโดยใช้ Global memory เวลาในการประมวลผลจากชุดข้อมูลที่หนึ่งโหนดรากความเร็วในการประมวลผลเพิ่มขึ้น 4.2 เท่าโหนดลูกความเร็วในการประมวลผลเพิ่มขึ้น

2.7 เท่าเวลาในการประมวลผลรวมเพิ่มขึ้น 2.9 เท่า จากชุดข้อมูลที่สองโหนดรากความเร็วในการประมวลผลเพิ่มขึ้น 1.2 เท่าโหนดลูกความเร็วในการประมวลผลเพิ่มขึ้น 1.8 เท่าเวลาในการประมวลผลรวมเพิ่มขึ้น 1.8 เท่า

จากการทดลองโดยใช้ Share memory เวลาในการประมวลผลจากชุดข้อมูลที่หนึ่งโหนดรากความเร็วในการประมวลผลเพิ่มขึ้น 4.3 เท่าโหนดลูกความเร็วในการประมวลผลเพิ่มขึ้น 3.16 เท่าเวลาในการประมวลผลรวมเพิ่มขึ้น 3.34 เท่า จากชุดข้อมูลที่สองโหนดรากความเร็วในการประมวลผลเพิ่มขึ้น 1.2 เท่าโหนดลูกความเร็วในการประมวลผลเพิ่มขึ้น 1.9 เท่าเวลาในการประมวลผลรวมเพิ่มขึ้น 1.8 เท่า

5.2 แนวทางในการพัฒนาต่อ

ในงานวิจัยนี้ได้อธิบายถึงแนวทางในการเพิ่มความเร็วในการประมวลผลการเชื่อมความสัมพันธ์ระหว่างวัตถุสำหรับฐานข้อมูลเชิงพื้นที่โดยการประมวลผลแบบขนานบนหน่วยประมวลผลกราฟิก อย่างไรก็ตามการเขียนคำสั่งให้หน่วยประมวลผลกราฟิกในการแก้ปัญหานั้นยังมีข้อจำกัดในส่วนของการทำงาน ยังคงต้องมีการประสานกันระหว่างหน่วยประมวลผลกลางและหน่วยประมวลผลกราฟิก นอกจากนั้นเวลาที่ใช้การส่งข้อมูลระหว่างหน่วยประมวลผลกลางและหน่วยประมวลผลกราฟิกยังส่งผลกระทบต่อเวลาในการประมวลผลด้วย ในการเขียนคำสั่งบนหน่วยประมวลผลกราฟิกยังไม่สามารถประยุกต์ใช้ให้ประมวลผลแบบอัตโนมัติได้ยังคงเป็นการเขียนคำสั่งในการทำงานเฉพาะเจาะจงอยู่ ดังนั้นแนวทางในการพัฒนาต่อคือการออกแบบวิธีการหรือการจัดตารางประสานการทำงานระหว่างหน่วยประมวลผลกลางและหน่วยประมวลผลกราฟิกให้สามารถทำงานได้อัตโนมัติ และให้สามารถเพิ่มเข้าไปเป็นส่วนหนึ่งของโปรแกรมบริหารจัดการฐานข้อมูลอื่น ๆ ได้

รายการอ้างอิง

[1] B. Thomas, Hans P. Kriegel and S. Bernhard. Efficient Processing of Spatial Joins Using R-tree, *ACM SIGMOD international conference on Management of data*. 1993:237-246.

[2] E.H. Jacox and H. Samet. Spatial Join Techniques. *ACM Transactions on Database Systems (TODS)*. Volume 32 Issue 1. March 2007:1–45.

[3] NVIDIA CUDA™. *NVIDIA CUDA Programming Guide*. Version 3.2. 2010.

[4] G. Antonin. R-tree : A Dinamic Index Structure for Spatial Searching. *ACM SIGMOD international conference on Management of data*. June 18-211984.

[5] X. Xiang and S. Tuo. R-Tree: A Hardware Implemention. *In Proceeding of International Conference on Computer Design*. July 14-17, 2008:3-9.

[6] J. Sanders and E. Kandrot. *CUDA BY EXAMPLE An Introduction to General-Purpose GPU Programming*. 2010.

[7] A. Nanopoulos, A. N. Papadopoulos and Y. Theodoridis Y. Manolopoulos. *R-trees: Theory and Applications*. Springer. 2006.

[8] L. Mutenda and M. Kitsuregawa, Parallel R-tree Spatial Join for a Shared-Nothing Architecture. *Database Applications in Non-Traditional Environments*. 1999: 423-430.

[9] N. Govindaraju, M. Harris, J. Kruger, A. E. Lefohn, and T. Purcell J. D. Owen D. Luebke. A survey of general-purpose computation on graphics hardware. *Computer Graphics Forum*. Volumn 26. 2007.

[10] S. Zhang, J. Han, Z. Liu, K. Wang, and Z. Xu. SJMR : Parallelizing spatial join with MapReduce on clusters. *CLUSTER '09. IEEE International Conference.* 2009:1-8.

[12] M. Kunjir and A. Manthramurthy. Using Graphics Processing in Spatial Indexing Algorithm. *Research report*, *Indian Institute of Scienc*e. 2009.

[13] K. Ibrahim and F. Cristos. On Packing R-tree. *CIKM '93 Proceedings of the second international conference on Information and knowledge management*. 1993:490-499.

[14] T. Yannis. *The R-tree Portal*. [Online]. 2005 Available from : http://www.rtreeportal.org [2011, February]

ประวัติผู้เขียนวิทยานิพนธ์

ชื่อผู้เขียนวิทยานิพนธ์ : นางสาวต้องใจ แย้มผกา
ประวัติส่วนตัว : เกิดเมื่อวันที่ 24 มิถุนายน 2523 จังหวัดกรุงเทพมหานคร
ที่อยู่ปัจจุบัน : บ้านเลขที่ 388/91 หมู่บ้านร่มเกล้าวิลล่า ซอยร่มเกล้า 22 เขตมีนบุรี แขวงมีนบุรี กทม. 10510
E-mail : Tongjai.Y@Student.chula.ac.th
ประวัติการศึกษา : ประกาศนียบัตรวิชาชีพ และประกาศนียบัตรวิชาชีพชั้นสูง สาขาคอมพิวเตอร์ธุรกิจ โรงเรียนสยามบริหารธุรกิจ
วิทยาศาสตรบัณฑิต สาขาเทคโนโลยีคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรี ปีการศึกษา 2551