บทที่ 5

สรุปผลงานวิจัยและข้อเสนอแนะ

วิทยานิพนธ์ฉบับนี้มีวัตถุประสงค์เพื่อออกแบบระบบที่ชื่อว่า *เวอร์ชวลคูด้า* เพื่อจัดการ การใช้ทรัพยากรจีพียูร่วมกันสำหรับเวอร์ชวลแมชชีน จากเครื่องที่ให้บริการทรัพยากรจีพียูได้อย่างมีประสิทธิภาพ และสามารถจัดสรรทรัพยากรจีพียูให้เพียงพอต่อการใช้งานสำหรับคูด้าแอพพลิเคชันจากเวอร์ชวลแมชชีนที่ขอเข้าใช้งานจีพียูในเวลาเดียวกัน ดังนั้นในบทนี้เสนอ การสรุปผลการวิจัย และข้อเสนอแนะเพิ่มเติม รายละเอียดดังต่อไปนี้

5.1 สรุปผลการวิจัย

5.1.1 สรุปผลการทดลองประมวลผลผ่านซีพียูของเวอร์ชวลแมชชีนเปรียบเทียบกับการประมวลผลด้วยจีพียูผ่านเวอร์ชวลคูด้าของเวอร์ชวลแมชชีน

จากผลการทดลองตามรูปที่ 4.3 ซึ่งแสดงค่า Speedup เปรียบเทียบประสิทธิภาพการทำงานระหว่างจีพียูผ่านเวอร์ชวลคูด้ากับซีพียูบนเวอร์ชวลแมชชีน จะเห็นได้ว่าการใช้จีพียูเข้ามาช่วยประมวลผลสำหรับแอพพลิเคชันบางประเภทบนเวอร์ชวลแมชชีน เช่น การคูณเมตริกซ์ จะทำให้แอพพลิเคชันดังกล่าวใช้เวลาในการประมวลผลน้อยกว่านำไปประมวลบนซีพียู และจากภาพที่ 4.1 แสดงให้เห็นว่าแนวกราฟจะเพิ่มขึ้นตามขนาดของข้อมูลที่ใช้ในการประมวลผลที่มากขึ้น แสดงว่าหากข้อมูลที่ต้องนำไปประมวลผลมีมากซีพียูจะต้องใช้เวลาในการประมวลผลมากกว่าจีพียูตามข้อมูลที่เพิ่มขึ้นตามลำดับ

ดังนั้นสามารถสรุปได้ว่าประสิทธิภาพในการใช้งานจีพียูผ่านเวอร์ชวลคูด้านั้นดีกว่าประสิทธิภาพในการใช้งานซีพียูบนเวอร์ชวลแมชชีนมาก และเป็นการพิสูจน์ว่าถึงแม้ว่าเวอร์ชวลคูด้าจะต้องทำงานผ่านไลบรารี อีกทั้งต้องมีการถ่ายโอนข้อมูลไปมาระหว่างเวอร์ชวลแมชชีน กับโฮสโดยผ่านเน็ตเวิร์ค เวอร์ชวลคูด้าก็ยังทำงานได้ดีกว่าการไม่มีมันและผู้ใช้งานเวอร์ชวลแมชชีนต้องทำงานบนซีพียูของเวอร์ชวลแมชชีนเพียงอย่างเดียว

5.1.2 สรุปผลการทดลองของระยะเวลาที่เรียกใช้งานจีพียูผ่านเวอร์ชวลคูด้า

5.1.2.1 สรุปผลการเปรียบเทียบการทำงานระหว่างแบ็คเอ็นด์ และฟรอนท์เอนด์ ตั้งแต่ Initialization (คำสั่ง cuInit) ถึง Function Call (คำสั่ง cuLaunchGrid) ด้วยการคูณเมตริกซ์ ด้วยวิธีการวัดผลตามภาพที่ 4.2 เปรียบเทียบระยะเวลาที่ $T1$ กับ $T2$ จากผลการทดลองตามภาพที่ 4.3 สรุปได้ว่า การประมวลผลของฟรอนท์เอนด์จะใช้เวลาในการประมวลผลมากกว่าการประมวลผลของแบ็คเอ็นด์ ซึ่งโอเวอร์เฮดดังกล่าวเกิดจากเวลาในการคัดลอกข้อมูลของฟรอนท์เอนด์ผ่านเน็ตเวิร์คไปยังแบ็คเอ็นด์บวกกับเวลาที่เสียไปสำหรับการเข้าใช้งานจีพียูของแบ็คเอ็นด์ ก่อนที่จะส่งผลกลับมายังฟรอนท์เอนด์ว่าประมวลผลสำเร็จ

5.1.2.2 สรุปผลการเปรียบเทียบฟังก์ชั่นการคัดลอกข้อมูลแบบ Host to Device (จากซีพียูไปจีพียู) ระหว่างแบ็คเอ็นด์ และฟรอนท์เอนด์ ด้วยวิธีการวัดผลตามภาพที่ 4.2 เปรียบเทียบระยะเวลาที่ $T3$ กับ $T4$ จากผลการทดลองตามภาพที่ 4.4 สรุปได้ว่า เวลาที่ใช้ในคัดลอกข้อมูลแบบ Host to Device ของฟรอนท์เอนด์จะใช้เวลามากกว่าคัดลอกข้อมูลแบบ Host to Device ของแบ็คเอ็นด์ ซึ่งโอเวอร์เฮดดังกล่าวเกิดจากเวลาในการคัดลอกข้อมูลของฟรอนท์เอนด์ผ่านเน็ตเวิร์คไปยังแบ็คเอ็นด์ แต่สำหรับการคัดลอกข้อมูลในส่วนของแบ็คเอ็นด์นั้นไม่จำเป็นต้องเสียเวลาในการคัดลอกข้อมูลผ่านเน็ตเวิร์ค อีกทั้งเวลาในการคัดลอกข้อมูลนี้จะเพิ่มตามขนาดของข้อมูลที่ใช้ในการถ่ายโอนที่เพิ่มมากขึ้นตามลำดับ

5.1.2.3 สรุปผลการเปรียบเทียบฟังก์ชั่นการคัดลอกข้อมูลแบบ Device to Host (จากจีพียูไปซีพียู) ระหว่างแบ็คเอ็นด์ และฟรอนท์เอนด์ ด้วยวิธีการวัดผลตามภาพที่ 4.2 เปรียบเทียบระยะเวลาที่ $T5$ กับ $T6$ จากผลการทดลองตามภาพที่ 4.5 สรุปได้ว่า เวลาที่ใช้ในคัดลอกข้อมูลแบบ Device to Host ของฟรอนท์เอนด์จะใช้เวลามากกว่าคัดลอกข้อมูลแบบ Device to Host ของแบ็คเอ็นด์ ซึ่งโอเวอร์เฮดดังกล่าวเกิดจากเวลาที่ใช้ในการคัดลอกข้อมูลจากยังแบ็คเอ็นด์ผ่านเน็ตเวิร์คกลับมาที่ฟรอนท์เอนด์บวกกับเวลาที่แบ็คเอ็นด์คัดลอกข้อมูลจากจีพียูมาว่างไว้ยังซีพียูเพื่อเตรียมที่จะส่งกลับไปยังฟรอนท์เอนด์ แต่สำหรับการคัดลอกข้อมูลในส่วนของแบ็คเอ็นด์นั้นไม่จำเป็นต้องเสียเวลาในการคัดลอกข้อมูลผ่านเน็ตเวิร์ค

จากสรุปผลการทดลองทั้งหมดใน 5.1.2 นี้จะแสดงให้เห็นว่าโอเวอร์เฮดส่วนใหญ่ของการใช้งานผ่านเวอร์ชวลคูด้าจะเสียไปกับการถ่ายโอนข้อมูลไปมา แต่หากพิจารณาจากสรุปผลการทดลองที่ 5.1.1 ซึ่งแสดงให้เห็นว่าการใช้งานจีพียูผ่านเวอร์ชวลคูด้านั้น ช่วยทำให้การประมวลผลเสร็จเร็วกว่าใช้เพียงซีพียูประมวลผลเพียงอย่างเดียว ถึงแม้จะเสียโอเวอร์เฮดในการถ่ายโอนข้อมูล

ไปมาก็ตาม การใช้งานผ่านเวอร์ชวลคูด้าก็ยังคงให้ประสิทธิภาพสำหรับทำงานที่ดีกว่าการใช้เพียงซีพียูประมวลผลเพียงอย่างเดียว หรือการที่ไม่มีเวอร์ชวลคูด้า

5.1.3 สรุปผลการใช้งานจีพียูร่วมกันของเวอร์ชวลแมชชีน

จากผลการทดลองตามภาพที่ 4.8 และ 4.9 แสดงให้เห็นว่าเวอร์ชวลคูด้าสามารถให้บริการกับ 2 เวอร์ชวลแมชชีนที่ต้องการเข้าให้งานจีพียูพร้อมกันได้จริง เพราะขณะที่เมตริกซ์มีขนาด 800 x 1600, 1200 x 2400, 1600 x 3200, 2000 x 4000, 2400 x 4800, 2800 x 5600, 3200 x 6400 และ 3600 x 7200 จะใช้เวลาในการประมวลผลน้อยกว่า 2 เวอร์ชวลแมชชีนที่เข้าใช้งานจีพียูได้ครั้งละ 1 เวอร์ชวลแมชชีนตามรูปที่ 4.6 และใช้เวลาไม่เท่ากับ 2 เท่าเมื่อเปรียบเทียบ 1 เวอร์ชวลแมชชีนที่เข้าใช้งานจีพียูตามรูปที่ 4.7 แต่ขณะที่เมตริกซ์มีขนาด 4000 x 8000 และ 4400 x 8800 นั้น เวลาที่ใช้ประมวลผลจะใกล้เคียงกันกับการใช้งานจีพียูได้ครั้งละ 1 เวอร์ชวลแมชชีนกรณีเข้าใช้งานจีพียูพร้อมกัน 2 เวอร์ชวลแมชชีน เนื่องจากการเข้าใช้งานจีพียูพร้อมกันของ 2 เวอร์ชวลแมชชีนต้องใช้พื้นที่สำหรับข้อมูลที่ต้องนำไปประมวลผลเกินกว่าพื้นที่ ที่จีพียูมีให้บริการ จึงทำให้เกิดการรอที่จะประมวลผลของเวอร์ชวลแมชชีนตามที่แบ็คเอ็นด์ของเวอร์ชวลคูด้าจัดสรร

การทดลองนี้ได้กำหนดให้พื้นที่ ที่จีพียูสามารถให้บริการได้อยู่ที่ 640000000 Bytes หรือ 610.3515625 MB โดยประมาณ เนื่องจากพื้นที่ Memory ของ NVIDIA GeForce 8800GTS นี้มีพื้นที่ใช้งานอยู่ที่ 640 MB แต่ขณะที่ได้ทำการทดลองนั้น ผู้วิจัยได้กำหนดให้พื้นที่สำหรับให้บริการสำหรับหรับจีพียูอยู่ที่ 640 MB พอดี ปรากฏว่าระบบเวอร์ชวลคูด้าไม่สามารถให้บริการพร้อมกัน 2 เวอร์ชวลแมชชีนได้และมีค่าผิดพลาด (Error) คืนกลับมาว่า Memory ไม่เพียงพอ ซึ่งแสดงว่าจีพียูไม่สามารถให้บริการพื้นที่ Memory ที่ 640 MB ได้ ผู้วิจัยจึงทำการลดขนาดของพื้นที่ที่ให้บริการลงเหลือ 610.3515625 MB ทั้งนี้เพื่อให้เข้ากับขนาดของเมตริกซ์ที่ได้นำมาทดลองดังตารางที่ 4.1 ซึ่งเป็นค่าสูงสุดที่สามารถให้บริการได้ และเหมาะสมกับการทดลองดังกล่าว

5.1.4 แอพพลิเคชันที่เหมาะสมสำหรับระบบเวอร์ชวลคูด้า

จากการงานวิจัยนี้ ได้ทำการทดลองคูด้าแอพพลิเคชันการคูณเมตริกซ์สำหรับระบบเวอร์ชวลคูด้าที่พัฒนาขึ้น จึงสรุปได้ว่าแอพพลิเคชันที่เหมาะสมสำหรับระบบเวอร์ชวลคูด้า คือ แอพพลิเคชันที่ต้องใช้เวลามากในการประมวลผลบนซีพียู จึงนำแอพพลิเคชันดังกล่าวมาให้จีพียูช่วยในการประมวลผล โดยเรียกใช้งานผ่านระบบเวอร์ชวลคูด้า ซึ่งเวลาที่แอพพลิเคชันประมวลผล

บนจีพียูจะต้องมากกว่าเวลาที่ใช้ในการถ่ายโอนข้อมูลไปมาระหว่างเวอร์ชวลแมชชีนและเครื่องโฮส หรือแอพพลิเคชันที่ประมวลผลบนจีพียูโดยผ่านระบบเวอร์ชวลคูด้าแล้วใช้เวลาน้อยกว่าเมื่อนำไปประมวลผลบนซีพียู

5.2 ข้อเสนอแนะเพิ่มเติม

ในส่วนนี้จะกล่าวถึงข้อเสนอแนะเพิ่มเติมของงานวิจัย รายละเอียดดังต่อไปนี้

1. จากผลการทดลองทั้งหมดแสดงให้เห็นได้ว่าการทำงานของระบบเวอร์ชวลคูด้าจะเสียเวลาไปกับการถ่ายโอนข้อมูลไปมาระหว่างแบ็คเอ็นด์ และฟรอนท์เอนด์ เพื่อคัดลอกข้อมูลจากซีพียูของเวอร์ชวลแมชชีนไปยังจีพียูและจากจีพียูกลับมายังซีพียูของเวอร์ชวลแมชชีน หากลดเวลาในการถ่ายโอนข้อมูลไปมาระหว่างแบ็คเอ็นด์ และฟรอนท์เอนด์ของระบบเวอร์ชวลคูด้าได้ จะทำให้เวลาในการประมวลผลโดยใช้จีพียูผ่านเวอร์ชวลคูด้าใช้เวลาในการประมวลผลลดน้อยลง

2. ระบบเวอร์ชวลคูด้ารองรับเพียงฟังก์ชั่นพื้นฐานที่สำคัญของ CUDA Driver API ได้แก่

- cuInit
- cuDeviceGet
- cuCtxCreate
- cuModuleLoad
- cuModuleGetFunction
- cuMemAlloc
- cuMemcpyHtoD/ cuMemcpyDtoH
- cuParamSetv
- cuParamSetSize
- cuFuncSetBlockShape
- cuLaunchGrid
- cuCtxDetach
- cuMemFree

แต่ยังไม่รองรับฟังก์ชั่นอื่น ๆ ของ CUDA Driver API หากระบบเวอร์ชวลคูด้าสามารถรองรับฟังก์ชั่นอื่น ๆ ได้จะทำให้สามารถรองรับคูด้าแอพพลิเคชันได้หลากหลายมากขึ้น