BỘ GIÁO DỤC VÀ ĐÀO TẠO
ĐẠI HỌC ĐÀ NẴNG

TRƯƠNG VĂN HIỆU

NGHIÊN CỨU CÁC GIẢI THUẬT SONG SONG TRÊN HỆ THỐNG XỬ LÝ ĐỒ HỌA GPU ĐA LÕI

Chuyên ngành: KHOA HỌC MÁY TÍNH
Mã số : 60.48.01

TÓM TẮT LUẬN VĂN THẠC SĨ KỸ THUẬT

Đà Nẵng - Năm 2011

Công trình được hoàn thành tại
ĐẠI HỌC ĐÀ NẴNG

Người hướng dẫn khoa học: **TS. Nguyễn Thanh Bình**

Phản biện 1: **PGS.TS. Phan Huy Khánh**

Phản biện 2: **TS. Trương Công Tuấn**

Luận văn được bảo vệ tại Hội đồng chấm Luận văn tốt nghiệp thạc sĩ kỹ thuật họp tại Đại học Đà Nẵng vào ngày 11 tháng 9 năm 2011.

Có thể tìm hiểu luận văn tại:

- Trung tâm Thông tin - Học liệu, Đại học Đà Nẵng
- Trung tâm Học liệu, Đại học Đà Nẵng

MỞ ĐẦU

1. Lý do chọn đề tài

Nhu cầu tính toán trong lĩnh vực khoa học, công nghệ ngày càng cao và trở thành một thách thức lớn. Từ đó các giải pháp nhằm tăng tốc độ tính toán đã được ra đời, từ năm 2001 đến năm 2003 tốc độ của Pentium 4 đã tăng gấp đôi từ 1.5GHz lên đến 3GHz. Tuy nhiên hiệu năng của CPU (Central Processing Unit) không tăng tương xứng như mức gia tăng xung của CPU và việc gia tăng tốc độ xung của CPU nhanh chóng chạm phải ngưỡng tối đa mà cụ thể trong khoảng thời gian 2 năm từ năm 2003 đến năm 2005 tốc độ của CPU chỉ tăng từ 3GHz lên 3.8GHz. Trong quá trình tăng tốc độ xung của CPU các nhà sản xuất đã gặp phải vấn đề về nhiệt độ của CPU sẽ quá cao và các giải pháp tản nhiệt khí đã đến mức tới hạn không thể đáp ứng được khả năng làm mát khi CPU hoạt động ở xung quá cao như vậy. Vì vậy việc gia tăng xung hoạt động của CPU không sớm thì muộn cũng sẽ đi vào bế tắc.

Trước tình hình này, các nhà nghiên cứu vi xử lý đã chuyển hướng sang phát triển công nghệ đa lõi, nhiều lõi, với cơ chế xử lý song song trong các máy tính nhằm tăng hiệu năng và tiết kiệm năng lượng.

Một trong các công nghệ xử lý song song ra đời đó là GPU (Graphics Processing Unit - bộ xử lý đồ họa). Ban đầu, việc chế tạo GPU chỉ với những mục đích công việc phù hợp với khả năng là tăng tốc độ xử lý đồ họa, cũng như trong ngành trò chơi là chủ yếu. Nhưng đến thời điểm GPU NV30 của NVIDIA ra đời, GPU bắt đầu tham gia vào những công việc khác ngoài đồ họa như: Hỗ trợ tính toán dấu chấm động đơn, hỗ trợ tính toán lên cả ngàn lệnh. Vì thế đã

nảy sinh ra ý tưởng dùng GPU để xử lý, tính toán song song những chương trình không thuộc đồ họa.

Câu hỏi được đặt ra là làm thế nào để ứng dụng GPU vào việc xử lý tính toán song song? Câu hỏi này nhanh chóng được giải quyết bằng công nghệ CUDA (Compute Unified Device Architecture – kiến trúc thiết bị hợp nhất cho tính toán) của NVIDIA ra đời năm 2007. Với CUDA, các lập trình viên nhanh chóng phát triển các ứng dụng song song trong rất nhiều lĩnh vực khác nhau như: Điện toán hóa học, sắp xếp, tìm kiếm, mô phỏng các mô hình vật lý, chuẩn đoán y khoa, thăm dò dầu khí, … CUDA là bộ công cụ phát triển phần mềm trên GPU được xây dựng bằng ngôn ngữ lập trình C. Với CUDA các lập trình viên dùng để điều khiển GPU để xử lý, tính toán song song các dữ liệu lớn.

Việc tăng tốc trong quá trình tính toán không những đòi hỏi những thiết bị GPU có khả năng xử lý tốc độ cao với dữ liệu khổng lồ mà cần phải có những giải thuật song song hữu hiệu.

Xuất phát từ nhu cầu trên tôi chọn đề tài: “**Nghiên cứu các giải thuật song song trên hệ thống xử lý đồ họa GPU đa lõi”**.

2. Mục tiêu và nhiệm vụ nghiên cứu

Để hoàn thành mục đích ý tưởng đề ra cần nghiên cứu các nội dung như sau:

Tìm hiểu các giải thuật tính toán song song, các cách thiết kế mẫu trong tính toán song song.

Tìm hiểu cấu trúc của GPU

Tìm hiểu và triển khai lập trình song song với CUDA

Phát biểu, phân tích, cài đặt giải thuật cho bài toán đặt ra.

Xây dựng giải thuật và ứng dụng áp dụng giải thuật tính toán song song trên thiết bị đồ họa GPU.

Đánh giá kết quả theo yêu cầu của đề tài.

3. Đối tượng và phạm vi nghiên cứu

Đối tượng nghiên cứu

Trong khuôn khổ luận văn thuộc loại nghiên cứu và ứng dụng, tôi chỉ giới hạn nghiên cứu các vấn đề sau:

- Lý thuyết tính toán song song.

- Chuyển đổi một số giải thuật xử lý trình tự sang tính toán song song sao cho tốc độ tính toán nhanh hơn giải thuật cũ, phát biểu bài toán thực tế có áp dụng giải thuật trên, cài đặt và giải quyết trên thiết bị xử lý đồ họa GPU bằng ngôn ngữ lập trình CUDA.

Phạm vi nghiên cứu

Nghiên cứu chuyển một số giải thuật cơ bản tuần tự sang song song chạy trên thiết bị đồ họa GPU của NVIDIA bằng ngôn ngữ CUDA.

4. Phương pháp nghiên cứu

Phương pháp nghiên cứu lý thuyết

- Nghiên cứu lý thuyết về tính toán song song, các giải thuật tính toán song song.

- Nghiên cứu lý thuyết về cơ chế hoạt động tính toán trong GPU.

Phương pháp nghiên cứu thực nghiệm

Sử dụng phương pháp nghiên cứu lý thuyết kết hợp với nghiên cứu thực nghiệm:

- Thiết kế giải thuật song song và cài đặt bằng CUDA.

- Triển khai xây dựng ứng dụng.

- Chạy thử nghiệm và lưu trữ các kết quả đạt được, sau đó đánh giá lại kết quả.

5. Ý nghĩa khoa học và thực tiễn của đề tài

Ý nghĩa khoa học

- Nắm được các giải thuật, các mẫu thiết kế tính toán song song.

- Khai thác các bộ thư viện CUDA SDK ứng dụng trong ngôn ngữ lập trình song song bằng CUDA.

Ý nghĩa thực tiễn

Việc nghiên cứu và đề xuất giải pháp để "Nghiên cứu các giải thuật song song trên hệ thống xử lý đồ họa GPU", làm cơ sở để giải quyết một số bài toán cần lượng tính toán lớn với dữ liệu khổng lồ.

6. Bố cục luận văn

Luận văn bao gồm 3 chương sau đây:

Chương 1: Cơ sở lý thuyết tính toán song song. Trong chương này giới thiệu tổng quan về tính toán song song như: Lịch sử phát triển song song, phân loại kiến trúc song song, các mô hình lập trình song song và đánh giá hiệu quả tính toán song song. Trình bày nguyên lý thiết kế giải thuật song song, giới thiệu một số mẫu thiết kế giải thuật song song bằng phương pháp chia để trị, phương pháp cây nhị phân. Giới thiệu bài toán sắp xếp, bài toán tính tổng dùng giải thuật song song. Qua đây đưa ra cái nhìn tổng quan hơn về tính toán song song.

Chương 2. Cấu trúc hệ thống xử lý đồ họa GPU và công nghệ tính toán hỗ trợ song song dữ liệu CUDA. Chương này giới thiệu về thiết bị đồ họa GPU đa lõi của hãng NVIDIA gồm các vấn đề: lịch sử phát triển, mô tả kiến trúc, nguyên lý hoạt động và những ứng dụng trên thiết bị đồ họa này. Đưa ra những so sánh khác biệt của CPU và GPU. Trình bày ngôn ngữ lập trình song song CUDA trên thiết bị đồ họa GPU của hãng NVIDIA. Áp dụng giải quyết một số bài toán cộng ma trận, nhân ma trận, …bằng phương pháp song song dùng ngôn ngữ CUDA thực thi trên thiết bị đồ họa.

Chương 3. Xây dựng ứng dụng áp dụng giải thuật song song trên hệ thống đa lõi xử lý đồ họa GPU cho bài toán bắt cặp trình tự. Trong chương này trình bày một số khái niệm cơ bản về tin sinh học: AND, protein, … Từ đó tập trung mô tả bài toán so sánh trình tự sử dụng giải thuật Smith-Waterman và giải quyết bằng giải thuật song song bằng CUDA đưa ra những đánh giá về lợi ích của việc sử dụng giải thuật song song.

CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

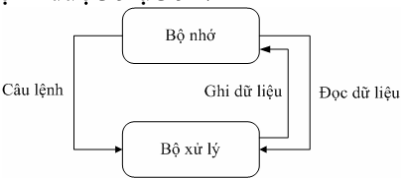
Trong chương này giới thiệu tổng quan về tính toán song song như: Lịch sử phát triển song song, phân loại kiến trúc song song, các mô hình lập trình song song và đánh giá hiệu quả giải thuật tính toán song song. Trình bày nguyên lý thiết kế giải thuật song song, giới thiệu một số mẫu thiết kế giải thuật song song bằng phương pháp chia để trị, phương pháp cây nhị phân. Giới thiệu bài toán sắp xếp, bài toán tính tổng dùng giải thuật song song. Qua đây đưa ra cái nhìn tổng quan hơn về tính toán song song.

1.1. TỔNG QUAN VỀ TÍNH TOÁN SONG SONG

1.1.1. Tổng quan về tính toán song song

1.1.1.1. Lịch sử ra đời tính toán song song

Trong những thập niên 60, nền tảng để thiết kế máy tính đều dựa trên mô hình của John Von Neumann (Xem 0Hình 1.1.), với một đơn vị xử lý được nối với một vùng lưu trữ làm bộ nhớ và tại một thời điểm chỉ có một lệnh được thực thi.



Hình 1.1. Mô tả kiến trúc Von Neumann

Với những bài toán yêu cầu về khả năng tính toán và lưu trữ lớn thì mô hình kiến trúc này còn hạn chế. Để tăng cường sức mạnh tính toán giải quyết các bài toán lớn có độ tính toán cao, người ta đưa ra kiến trúc mới, với ý tưởng kết hợp nhiều bộ xử lý vào trong một máy tính, mà hay gọi là xử lý song song (Multiprocessor) hoặc kết hợp sức mạnh tính toán của nhiều máy tính dựa trên kết nối mạng.

Kể từ lúc này, để khai thác được sức mạnh tiềm tàng trong mô hình máy tính nhiều bộ xử lý song song, cũng như trong mô hình mạng máy tính xử lý song song thì các giải thuật tuần tự không còn phù hợp nữa cho nên việc xây dựng thiết kế giải thuật song song là điều quan trọng. Giải thuật song song có thể phân rã công việc trên các phần tử xử lý khác nhau.

1.1.1.2. Tại sao phải tính toán song song

1.1.1.3. Một số khái niệm xử lý song song

Định nghĩa về xử lý song song

Xử lý song song là quá trình xử lý gồm nhiều tiến trình được kích hoạt đồng thời và cùng tham giải quyết một bài toán. Nói chung, xử lý song song được thực hiện trên những hệ thống đa bộ xử lý.

Phân biệt xử lý song song và xử lý tuần tự

Trong tính toán tuần tự với một bộ xử lý thì tại mỗi thời điểm chỉ được thực hiện một phép toán. Trong tính toán song song thì nhiều bộ xử lý cùng kết hợp với nhau để giải quyết cùng một bài toán cho nên giảm được thời gian xử lý vì mỗi thời điểm có thể thực hiện đồng thời nhiều phép toán.

Mục đích của xử lý song song

Thực hiện tính toán nhanh trên cơ sở sử dụng nhiều bộ xử lý đồng thời. Cùng với tốc độ xử lý nhanh, việc xử lý song song cũng sẽ giải được những bài toán phức tạp yêu cầu khối lượng tính toán lớn.

1.1.2. Phân loại các kiến trúc song song

1.1.2.1. Kiến trúc đơn dòng lệnh đơn luồng dữ liệu (SISD)

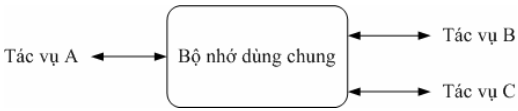
1.1.2.2. Kiến trúc đơn dòng lệnh đa luồng dữ liệu (SIMD

1.1.2.3. Kiến trúc đa dòng lệnh đơn dữ liệu (MISD)

1.1.2.4. Kiến trúc đa dòng lệnh đa luồng dữ liệu (MIMD)

1.1.3. Các mô hình lập trình song song

1.1.3.1. Lập trình bộ nhớ dùng chung



Hình 1.6. Mô tả lập trình giữa các tác vụ dùng chung bộ nhớ

1.1.3.2. Lập trình truyền thông điệp

1.1.3.3. Mô hình song song dữ liệu

1.1.3.4. Mô hình hướng đối tượng

1.1.3.5. Mô hình logic

1.1.4. Đánh giá hiệu quả tính toán song song

1.1.4.1. Thời gian thực hiện

Thời gian tính toán

Thời gian truyền thông

Thời gian rỗi

1.1.4.2. Tăng tốc và hiệu quả

1.1.4.3. Tính qui mô

1.2. THIẾT KẾ GIẢI THUẬT SONG SONG

Trong phần này đề cập đến phương pháp thiết kế giải thuật song song cho bài toán, quá trình thiết kế giải thuật thật sự không dễ dàng để có thể rút gọn thành một công thức đơn giản như công thức giải hệ phương trình bậc hai, giải hệ phương trình tuyến tính, … mà yêu cầu có sự sắp xếp tư duy sáng tạo. Mục đích của phần này đưa ra một

khung thiết kế, một sự đánh giá mang tính toán học nhằm giảm bớt những chi phí do phải quay lui lại sau khi lựa chọn phương án không hợp lý.

1.2.1. Nguyên lý thiết kế giải thuật song song

Khi thiết kế giải thuật song song, cần phải thực hiện:

- Phân chia dữ liệu cho các tác vụ.
- Chỉ ra cách truy cập và chia sẻ dữ liệu.
- Phân các tác vụ cho các tiến trình (cho bộ xử lý).
- Các tiến trình được đồng bộ ra sao.

Khi thiết kế giải thuật song song, cần tuân thủ 5 nguyên lý sau:

Các nguyên lý lập lịch: Mục đích là giảm tối thiểu các bộ xử lý sử dụng trong giải thuật sao cho thời gian tính toán là không tăng (xét theo khía cạnh độ phức tạp).

Nguyên lý hình ống: Nguyên lý này được áp dụng khi bài toán xuất hiện một dãy các thao tác { T_1, T_2, . . ., T_n}, trong đó T_{i+1} thực hiện sau khi T_1 kết thúc.

Nguyên lý chia để trị: Chia bài toán thành những phần nhỏ hơn tương đối độc lập với nhau và giải quyết chúng một cách song song.

Nguyên lý đồ thị phụ thuộc dữ liệu: Phân tích mối quan hệ dữ liệu trong tính toán để xây dựng đồ thị phụ thuộc dữ liệu và dựa vào đó để xây dựng giải thuật song song.

Nguyên lý điều kiện tranh đua: Nếu hai tiến trình cùng muốn truy cập vào cùng một mục dữ liệu chia sẻ thì chúng phải tương tranh với nhau, nghĩa là chúng có thể cản trở lẫn nhau.

1.2.2. Nhận thức vấn đề và chương trình có thể song song hóa

Trước khi dùng thời gian và sức lực nhằm phát triển giải pháp song song cho một vấn đề, hãy xác định có phải đó là một trong những vấn đề mà trên thực tế có thể song song hóa được hay không.

1.2.3. Thiết kế giải thuật song song bằng phân rã phục thuộc dữ liệu

1.2.4. Thiết kế giải thuật song song bằng phương pháp chia để trị

1.2.5. Ví dụ thiết kế giải thuật song song cho bài toán tính tổng

1.2.5.1. Xây dựng giải thuật tuần tự

1.2.5.2. Xây dựng giải thuật song song

1.2.6. Ví dụ thiết kế giải thuật song song cho bài toán sắp xếp

1.2.6.1. Giải thuật sắp xếp tuần tự

1.2.6.2. Xây dựng giải thuật song song

1.3. TỔNG KẾT CHƯƠNG 1

Trong chương này đã trình bày được một số lý thuyết cơ bản về lập trình song song như: Phân loại các kiến trúc song song, các mô hình lập trình song song và cách thức đánh giá hiệu quả giải thuật song song.

Ngoài ra, trong chương một còn trình bày một số nguyên lý thiết kế giải thuật song song cho bài toán chia để trị, phân rã phụ thuộc dữ liệu, … và áp dụng xây dựng giải thuật song song cho một số bài toán cơ bản như sắp xếp từ đó áp dụng để song song hóa một bài toán trình tự.

Môi trường để triển khai các giải thuật song song trong luận văn dùng ngôn ngữ CUDA thực thi trên thiết bị đồ họa GPU của hãng NVIDA. Nội dung chi tiết về phần này được trình bài trong chương hai.

CHƯƠNG 2: CẤU TRÚC HỆ THỐNG XỬ LÝ ĐỒ HỌA GPU VÀ CÔNG NGHỆ TÍNH TOÁN HỖ TRỢ SONG SONG DỮ LIỆU CUDA

Chương này giới thiệu về thiết bị đồ họa GPU đa lõi của hãng NVIDIA gồm các vấn đề: Lịch sử phát triển, mô tả kiến trúc, nguyên lý hoạt động và những ứng dụng trên thiết bị đồ họa này và đưa ra những so sánh khác biệt của CPU và GPU. Trình bày ngôn ngữ lập trình song song CUDA trên thiết bị đồ họa GPU của hãng NVIDIA. Áp dụng giải quyết một số bài toán cộng ma trận, nhân ma trận, …bằng phương pháp song song dùng ngôn ngữ CUDA thực thi trên thiết bị đồ họa.

2.1. CẤU TRÚC HỆ THỐNG XỬ LÝ ĐỒ HỌA GPU

2.1.1. Giới thiệu công nghệ GPU

Bộ xử lý đồ họa (Graphic Proccessing Unit) gọi tắc là GPU đã trở thành một phần không thể tách rời của hệ thống máy tính ngày nay. Trong sáu năm vừa qua đã đánh dấu sự gia tăng ấn tượng trong hiệu suất và khả năng của GPU. GPU hiện đại không chỉ là một công cụ xử lý đồ họa mạnh mà còn là một bộ xử lý hỗ trợ lập trình song song ở mức cao, giúp giải các bài toán số học cần khả năng xử lý số học phức tạp và băng thông bộ nhớ tăng hơn đáng kể so với CPU cùng loại. Sự tăng tốc nhanh chóng của GPU trong cả khả năng hỗ trợ lập trình và năng lực tính toán của nó đã tạo ra một xu hướng nghiên cứu mới. Một cộng đồng đã nghiên cứu và đã ánh xạ thành công một lượng lớn các vấn đề phức tạp đòi hỏi tính toán lớn vào GPU. Điều này trong nỗ lực chung nhằm mục đích ứng dụng GPU vào giải quyết các bài toán hiệu năng cao của tính toán hiện đại. Tính toán mục đích thông dụng trên GPU là một thay thế hấp dẫn cho CPU tại trong hệ thống máy tính hiện đại. Trong một tương lai không xa,

GPU sẽ đảm nhận thay cho CPU những công việc như xử lý hình ảnh, đồ họa, các tính toán phức tạp thay vì chỉ dừng lại ở những ứng dụng trò chơi 3D.

2.1.2. Kiến trúc GPU

GPU là một bộ xử lý với dư thừa tài nguyên tính toán. Tuy nhiên, xu hướng quan trọng nhất gần đây đó là trưng bày khả năng tính toán đó cho các lập trình viên. Những năm gần đây, GPU đã phát triển từ một hàm cố định, bộ xử lý chuyên dụng tới bộ xử lý lập trình song song, đầy đủ tính năng độc lập với việc bổ sung thêm các chức năng cố định và các chức năng chuyên biệt. Hơn bao giờ hết các khía cạnh về khả năng lập trình của bộ xử lý chiếm vị trí trung tâm.

2.1.2.1. Đường ống dẫn đồ họa

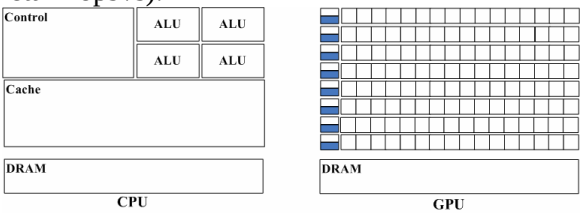
2.1.2.2. Tiến hóa của kiến trúc GPU

2.1.2.3. Kiến trúc GPU hiện đại

2.1.3. So sánh GPU và CPU

CPU là bộ vi xử lý trung tâm dùng để tính toán và xử lý các chương trình vi tính, dữ kiện... và đóng vai trò điều phối hoạt động của các thiết bị khác.

Còn GPU là bộ vi xử lý chuyên xử lý các dữ liệu về hình ảnh, đồ họa. Ngày nay cả CPU và GPU đều có những bước phát triển thần tốc, một GPU cao cấp có khả năng xử lý đạt tốc độ hàng tỷ phép tính trên giây (TetaFLops /s).



Hình 2.1. So sánh kiến trúc CPU và GPU

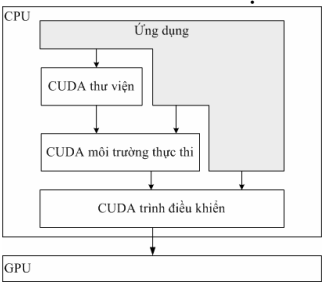
Hình 2.1. cho thấy số phần tử toán học GPU nhiều hơn hẳn CPU, điều này mang đến cho GPU một khả năng xử lý song song cực kỳ hiệu quả.

2.2. CÔNG NGHỆ TÍNH TOÁN HỖ TRỢ SONG SONG DỮ LIỆU CUDA

2.2.1. Giới thiệu công nghệ CUDA

CUDA là từ viết tắt của thuật ngữ Compute Unified Device Architecture, tạm dịch là kiến trúc thiết bị hợp nhất cho tính toán. CUDA bắt đầu xuất hiện từ tháng bảy năm 2007 với vai trò ban đầu là một bộ công cụ phát triển phần mềm dựa trên ngôn ngữ lập trình C. Bây giờ CUDA đang tiến hóa thành kiến trúc điện toán GPU, hay còn gọi là GPGPU của NVIDIA. CUDA có mặt trên hầu hết các GPU đời mới của NVIDIA, từ dòng GeForce giành cho giải trí đến Quadro giành cho điện toán hình ảnh chuyên nghiệp và dòng Tesla cho tính toán hiệu năng cao.

Bộ phần mềm CUDA có các lớp mô tả trong Hình 2.3. gồm: Bộ điều khiển cho phần cứng, API lập trình, môi trường thực thi và hai thư viện toán học mức cao hơn của các hàm thường dùng: CUFFT và CUBLAS. Phần cứng được thiết kế để hỗ trợ bộ điều khiển hạng nhẹ và lớp môi trường thực thi. Từ đó cho tốc độ cao.



Hình 2.3. Kiến trúc bộ phần mềm CUDA

2.2.2. Ứng dụng của CUDA trong các lĩnh vực công nghệ

2.2.2.1. CUDA cho ngành công nghiệp trò chơi

2.2.2.2. CUDA cho các ứng dụng video số

2.2.3. *Môi trường lập trình với CUDA*

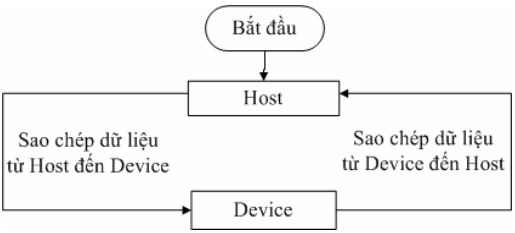
Để chương trình CUDA hoạt động được trong môi trường windows hoặc linux, cần phải có các thư viện hỗ trợ. Các thư viện này do NVIDIA cung cấp gồm có các phần sau: Trình điều khiển thiết bị đồ họa cho GPU của NIVIDA, bộ công cụ phát triển CUDA (gọi là CUDA Toolkit) và bộ CUDA SDK.

2.2.4. Cơ chế hoạt động một chương trình CUDA

Để hiểu cách hoạt động một chương trình CUDA (Xem Hình 2.6.), cần thống nhất một số các khái niệm sau:

Host: Là những tác vụ và cấu trúc phần cứng, phần mềm được xử lý từ CPU.

Divice: Là những tác vụ và cấu trúc phân cứng, phần mềm được xử lý tại GPU.



Hình 2.6. Sơ đồ hoạt động truyền dữ liệu giữa Host và Device

Cách hoạt động được mô tả như sau:

Bước 1: Dữ liệu cần được tính toán luôn ở trên bộ nhớ của Host vì vậy bước đầu tiên là truyền dữ liệu cần tính toán từ bộ nhớ Host qua bộ nhớ Device.

Bước 2: Sau đó Device sẽ gọi các hàm riêng của mình để tính toán dữ liệu đó. Sau khi tính toán xong, dữ liệu cần được trả về lại cho bộ nhớ của Host.

2.2.5. Mô hình lập trình

2.2.5.1. Bộ đồng xử lý đa luồng mức cao

2.2.5.2. Gom lô các luồng

2.2.6. Mô hình bộ nhớ

2.2.7. Tìm hiểu ngôn ngữ lập trình CUDA

2.2.7.1. Ngôn ngữ lập trình CUDA là mở rộng của ngôn ngữ lập trình C

2.2.7.2. Những mở rộng của ngôn ngữ lập trình CUDA so với ngôn ngữ C

2.2.7.3. Từ khóa phạm vi kiểu hàm

2.2.7.4. Từ khóa phạm vi kiểu biến

2.2.7.5. Thực hiện cấu hình

2.2.7.6. Các biến Built-in

2.2.7.7. Các lệnh điều khiển

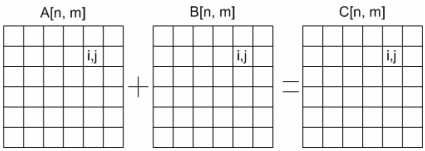
2.2.7.8. Các lệnh bộ nhớ

2.2.7.9. Biên dịch với NVCC

2.2.8. Ví dụ tính toán song song bằng CUDA

2.2.8.1. Cộng hai ma trận

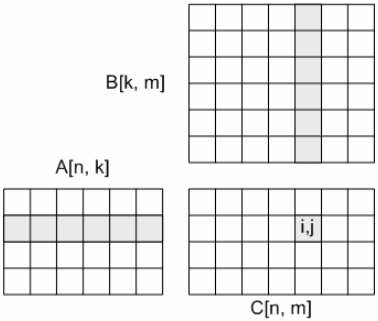
Mô tả bài toán: Cộng hai ma trận A[n][m] và B[n][m], kết quả trả về ma trận C[n][m].



Hình 2.9. Cộng hai ma trận

2.2.8.2. Nhân hai ma trận

Mô tả bài toán: Nhân hai ma trận A[n][k] và B[k][m], kết quả trả về ma trận C[n][m].



Hình 2.10. Nhân hai ma trận

2.3. TỔNG KẾT CHƯƠNG 2

Trong chương này đã trình bày kiến trúc của thiết bị đồ họa GPU, so sánh sự khác nhau giữa GPU và CPU, các ứng dụng trên thiết bị GPU đa lõi. Trình bày sơ lược về ngôn ngữ lập trình CUDA, cách biên dịch một chương trình CUDA chạy trên thiết bị đồ họa GPU của NVIDA. Viết một số ví dụ về lập trình song song, từ đó thấy được sức mạnh tính toán trên thiết bị đồ họa GPU đa lõi.

Trong chương tiếp theo, trình bày cách xây dựng giải thuật song song để giải quyết một bài toán chạy bằng CUDA.

CHƯƠNG 3: XÂY DỰNG ỨNG DỤNG GIẢI THUẬT SONG SONG TRÊN HỆ THỐNG ĐA LÕI XỬ LÝ ĐỒ HỌA GPU CHO BÀI TOÁN SO SÁNH TRÌNH TỰ

Chương này sẽ vận dụng cơ sở lý thuyết về lập trình song song đã trình bày trong chương một và sử dụng dụng công nghệ lập trình song song CUDA trên thiết bị đồ họa (GPU) đã trình bày trong chương hai để giải bài toán về tin sinh học so sánh trình tự bằng giải thuật quy hoạch động, giải thuật Smith-Waterman. Để tìm hiểu giải quyết bài toán so sánh trình tự, chương này còn trình bày một số khái niệm cơ bản về tin sinh học như: AND, protein, … Từ đó mô tả bài toán và giải quyết bằng giải thuật song song với CUDA. Đưa ra những so sánh, đánh giá về lợi ích của việc sử dụng giải thuật song song đối với bài toán này.

3.1. GIỚI THIỆU

Phần này giới thiệu những kiến thức cơ bản về tin sinh học và một số khái niệm về tin sinh học nhằm phục vụ cho việc tìm hiểu bài toán so sánh trình tự theo hệ gen.

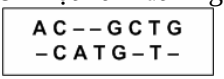
3.1.1. So sánh trình tự

Trong quá trình phân tích trình tự thì khái niệm so sánh trình tự đóng vai trò quan trọng. Đây là nền tảng cơ bản cho việc phân tích. So sánh trình tự giúp cho quá trình dự báo sự giống nhau về chức năng của các trình tự, dự báo cấu trúc bậc ba của DNA, protein. Trong việc tìm hiểu một gene mới, mọi người thường quan tâm đến việc xác định những đặc điểm để phân biệt gene đồng thời đưa ra những giả thuyết về chức năng của gene. Việc đưa ra những giả thuyết về chức năng của gene thường dựa vào những giải thuật đánh giá sự giống nhau, tương đồng giữa các trình tự.

3.1.2. Định nghĩa so sánh trình tự

So sánh trình tự (còn gọi là phép gióng hàng, gióng cột) là quá trình nghiên cứu sự giống nhau giữa các chuỗi trình tự (sequence), đo lường sự giống nhau giữa các trình tự. Cách thức so sánh giữa hai hay nhiều trình tự dựa trên việc so sánh một chuỗi các thành phần (ký tự) của trình tự để tìm ra những điểm tương đồng, giống nhau giữa các trình tự. Các trình tự được đề cập trong phần nghiên cứu này là các chuỗi trình tự DNA, RNA hoặc các trình tự amino acid (protein).

Xét 2 chuỗi: A C G C T G, C A T G T và đo lường sự giống nhau giữa hai chuỗi này. Hình 3.1. là một ví dụ về kết quả đo lường sự giống nhau của hai chuỗi trên. Tiêu chí để đánh giá sự giống nhau của hai chuỗi trên sẽ dựa trên một hàm đánh giá (scoring function).



Hình 3.1. Ví dụ về so sánh hai trình tự

Ký tự "–" được gọi là một gap, gap thể hiện cho ý nghĩa trong sinh học đó là một phần của trình tự đã bị mất đi do, các hành vi của quá trình tiến hóa sinh vật như: đột biến, sự mất đi một thành phần trong chuỗi trình tự. Giá trị hàm đánh giá càng cao thì kết quả so sánh trình tự càng tốt. Xét một hàm đánh giá đơn giản như sau: Nếu hai thành phần trong chuỗi là giống nhau thì hàm đánh giá sẽ có kết quả +2, nếu hai thành phần trong chuỗi khác nhau thì hàm đánh giá tại vị trí này sẽ có kết quả -1. Như vậy kết quả so sánh ở trên sẽ có giá trị hàm đánh giá là: 3*(2)+(-1)*5=1.

3.1.3. Hệ thống ký tự

Tập hợp các phần tử có thể xuất hiện trong trình tự gọi là một hệ thống các ký tự (Σ). Trong ADN, sử dụng một thệ thống kí tự Σ =

{A, C, G, T, λ} trong đó A, C, G, T đại diện cho bốn nucleotides: adenine (A), cytosine (C), guanine (G) và thymine (T). λ là ký tự đặc biệt đại diện cho một khoảng trống là một vị trí mà không có nucleotide thực tế. Trong hầu hết các trường hợp, ký tự gap ('-') có thể được sử dụng để thay thế cho λ. Bất kỳ một trình tự nào cũng là một sự thể hiện bởi các phần tử có thể xuất hiện trong trình tự và được định nghĩa trong Σ .

3.1.4. Các phép biến đổi

Phép thay thế: $x_0, x_1, x_2, x_3, x_4, x_5 \rightarrow x_0, x_1, x_2', x_3', x_4, x_5$

Phép chèn – xóa: $x_0, x_1, x_2, x_3, x_4, x_5 \rightarrow x_0, x_2, x_3, x_4, x_6, x_5$

Phép đảo ngược: $x_0, x_1, x_2, x_3, x_4, x_5 \rightarrow x_0, x_1, x_4, x_3, x_2, x_5$

Phép dịch chuyển: $x_0, x_1, x_2, x_3, x_4, x_5 \rightarrow x_0, x_3, x_4, x_5, x_1, x_2$

3.1.5. Khoảng cách

3.1.6. Sắp hàng trình tự hệ gen

3.1.7. Các thuật toán sắp hàng trình tự hệ gen

3.2. PHÁT BIỂU BÀI TOÁN SO SÁNH TRÌNH TỰ

3.2.1. Mô tả bài toán

Gọi S1 và S2 là hai chuỗi, một sự so sánh trình tự giữa S1 và S2 tạo ra hai chuỗi S1' và S2' bằng cách thêm các ký tự gap "-" vào S1 và S2, trong đó:

* |S1'|=|S2'|
* Nếu loại bỏ các ký tự "-" khỏi S1' và S2' ta sẽ có S1 và S2.

Với | S1|, | S2| lần lượt là chiều dài của S1 và S2.

Hình 3.2. So sánh hai trình tự

3.2.2. Hướng giải quyết bằng giải thuật quy hoạch động

Phần này giới thiệu hướng giải quyết bài toán so sánh trình tự bằng phương pháp quy hoạch động. Kỹ thuật này cho phép xây dựng

một phép so khớp tối ưu. Xét hai chuỗi trình tự S1 và S2, |S1| = n, |S2| = m, mục đích là tìm kiếm một phép so khớp tối ưu cho S1 và S2.

3.2.2.1. Giới thiệu về giải thuật quy hoạch động

3.2.2.2. Giải thuật quy hoạch động cho bài toán so sánh hai trình tự

Định nghĩa 3.1: Xét định nghĩa về một cơ chế đánh giá như sau:

- Nếu a và b là hai ký tự đại diện cho các amino acid (nucleotide) trong hai trình tự. Gọi σ (a,b) là hàm đánh giá của cặp a, b trong trong ma trận đánh giá.

- Hàm giá trị của gap phụ thuộc vào r > 0 và chiều dài k (γ (k) = − r * k), trong đó r là hằng số.

- T là một chuỗi, định nghĩa |T| là chiều dài của chuỗi T. Giá trị phép so khớp A của hai chuỗi S1, S2 với kết quả S1', S2':

$$\sum_{i \in K_1} \sigma(S_1'[i], S_2'[i]) - l_2 * r \qquad (1.14)$$

Trong đó, với K1 là tập các phần tử không là gap, |K1|=l1, l2 là tổng chiều dài của gap, l1 + l2 = | S1'| = | S2'|.

Phép so sánh trình tự tối ưu là phép so khớp có giá trị lớn nhất có thể có của hai chuỗi [6].

Định nghĩa 3.2: Gọi S(i, j) là giá trị của một phép so sánh trình tự tối ưu của hai chuỗi S1i (S1[1], …, S1[i]) và S2j (S2[1], …, S2[j]) (1≤ i ≤ n ,1≤ j ≤ m). Như vậy, S(n, m) sẽ là giá trị của một phép so sánh trình tự tối ưu của S1 và S2. S(i, j) được định nghĩa như sau:

S(0, 0) = 0

S(i, 0) = S(i-1, 0) - r, i > 0

S(0, j) = S(0, j-1) - r, j > 0

Từ định nghĩa của S(i, j) ta có công thức tính S(i, j) như sau:

S(i,j) = max{S(i-1,j-1) + σ(S1[i], S2[j]) , S(i-1,j) - r , S(i,j-1) - r}

với i > 0, j > 0 (1)

Trong đó S(i-1, j-1) là giá trị của một phép so sánh trình tự tối ưu của hai chuỗi S1i(S1[1],.., S1[i-1]) và S2j(S2[1],…,S2[j-1]) [6].

Công thức S(i, j) cho phép dễ dàng vận dụng kỹ thuật quy hoạch động. Xem ví dụ sau: Xét hai chuỗi "ACGCTG" và "CATGT".

Hàm đánh giá: $\sigma(a,b)=\begin{cases}2 & a=b\\ -1 & a\neq b\end{cases}$

Cho r = 1, thu được ma trận giá trị S(i, j) của hai chuỗi trình tự như Hình 3.4.

i \ j		0	1	2	3	4	5
			C	A	T	G	T
0		0	-1	-2	-3	-4	-5
1	A	-1	-1	1	0	-1	-2
2	C	-2	1	0	0	-1	-2
3	G	-3	0	0	-1	2	1
4	C	-4	-1	-1	-1	1	1
5	T	-5	-2	-2	1	0	3
6	G	-6	-3	-3	0	3	2

Hình 3.4. Ma trận đánh giá S(i, j)

Từ ma trận kết quả này có S(6, 5) = 2. Như vậy, giá trị phép so sánh trình tự tối ưu của hai chuỗi trên là hai. Từ kết quả này, ta đi tìm các phép so sánh trình tự tối ưu cho S(6, 5) = 2. Sử dụng kỹ thuật lưu vết có được ba trong số các kết quả như Hình 3.5.

i \ j		0	1	2	3	4	5
			C	A	T	G	T
0		0	0	0	0	0	0
1	A	0	-1	2	1	0	-1
2	C	0	2	1	1	0	-1
3	G	0	1	1	0	3	2
4	C	0	2	1	0	2	2
5	T	0	1	1	3	2	4
6	G	0	0	0	2	5	4

Hình 3.5. Kết quả giải thuật quy hoạch động cho bài toán PSA

Dựa vào các con đường được sinh ra do kỹ thuật lưu vết trong quá trình điền giá trị cho ma trận từ S(m, n) đến S(0, 0), các so sánh trình tự được sinh ra dựa trên nguyên tắc:

Nếu con đường đi theo hướng đường chéo từ S(i, j) đến S(i-1, j-1) thì hai ký tự đại diện cho S1[i] và S2[j] được ghi vào kết quả.

Nếu con đường đi theo hướng từ S(i, j) đến S(i-1, j) thì ký tự đại diện cho S1[i] và '-' được ghi vào kết quả (phép chèn một gap được thực hiện).

Nếu con đường đi theo hướng từ S(i, j) đến S(i, j-1) thì ký tự đại diện cho S2[j] và '-' được ghi vào kết quả (phép xóa một gap được thực hiện).

3.3. THIẾT KẾ GIẢI THUẬT SONG SONG

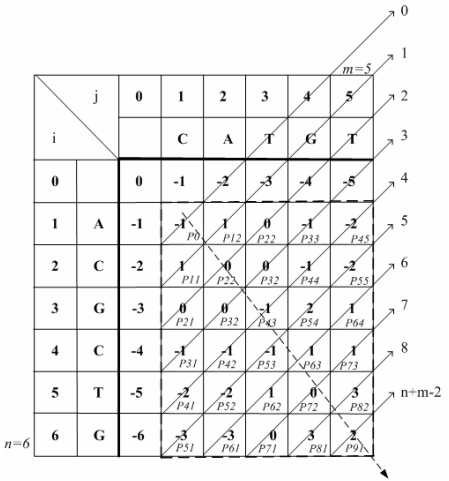
3.3.1. Xây dựng giải thuật

Như đã trình bày ở mục 3.2.2. giải thuật quy hoạch động đã giải quyết bài toán so sánh hai trình tự gen khá tốt. Tuy nhiên để nâng cao tốc độ tính toán cho bài toán này, có thể giải quyết bằng phương pháp tính toán song song trên thiết bị đồ họa GPU của hãng NVIDIA bằng công nghệ CUDA.

Nhận xét: Tại bước xây dựng ma trận đánh giá ta nhận thấy khi tính giá trị cho các phần tử nằm trên đường chéo phụ không phụ thuộc lẫn nhau, do đó có thể tính toán riêng rẽ từng phần tử trên từng luồng khác nhau.

Từ nhận xét trên ta đi xây dựng giải thuật song song cho bài toán so sánh trình tự hai hệ gen như sau:

- Với từng phần tử của đường chéo của ma trận đánh giá được tính bởi một luồng riêng. Như vậy đường chéo có n phần tử thì cần n luồng để tính giá trị cho đường chéo(Xem Hình 3.6.).



Hình 3.6. Phương pháp song song tính giá trị các phần tử ma trận đánh giá

3.3.2. Cài đặt giải thuật trên CUDA

3.3.3. Kết quả

3.4. ĐÁNH GIÁ KẾT QUẢ CHẠY CHƯƠNG TRÌNH

3.5. TỔNG KẾT CHƯƠNG 3

Trong chương này đã mô tả bài toán cơ bản trong sinh học, bài toán so sánh trình tự các hệ gen. Xây dựng giải thuật song song bằng CUDA chạy trên thiết bị đồ họa GPU Geforce 310M của NVIDA. So sánh hiệu năng giải bằng phương pháp quy hoạch động và phương pháp song song. Từ đó thấy được sự cần thiết của giải thuật song song trong các bài toán có số lượng tính toán khổng lồ.

KẾT LUẬN

Xử lý song song ra đời với mục đích làm tăng khả năng tính toán của máy tính bằng cách kết hợp nhiều bộ xử lý tham gia đồng thời vào quá trình xử lý.

Với sự phát triển của kiến trúc máy tính và mạng máy tính cho thấy rằng trong tương lai xử lý song song không những được thực hiện trên những siêu máy tính mà có thể được thực hiện trên các trạm làm việc, máy tính cá nhân, mạng máy tính. Nhưng hầu hết các thuật toán ngày nay đều là những thuật toán tuần tự. Cho nên cần xây dựng những thuật toán, cấu trúc dữ liệu cho phép xử lý song song nhằm tăng hiệu suất tính toán.

Kết quả đạt được của luận văn là trình bày tổng quan lý thuyết tính toán song song, phân loại các kiến trúc song song, các mô hình lập trình song song và đánh giá hiệu quả việc sử dụng thuật toán song song. Nắm được một số nguyên lý thiết kế giải thuật song song, trình bày một số phương pháp thiết kế giải thuật song song như: Thiết kế giải thuật song song bằng phân rã phụ thuộc dữ liệu, thiết kế giải thuật song song bằng phương pháp chia để trị,…Trình bày phương pháp nhận dạng một chương trình có thể chuyển sang lập trình song song. Nghiên cứu một số bài toán cơ bản đưa ra giải thuật trình tự và giải thuật song song. Từ đó đưa ra lợi ích áp dụng giải thuật song song.

Luận văn đã giới thiệu công nghệ hỗ trợ tính toán song song trên thiết bị đồ họa GPU của hãng NVIDIA và trình bày ngôn ngữ lập trình song song CUDA. Áp dụng một số bài toán cơ bản như: Cộng hai ma trận, nhân hai ma trận, … giải quyết bằng phương pháp lập trình song song trên thiết bị đồ họa GPU với ngôn ngữ CUDA.

Xây dựng ứng dụng giải quyết bài toán so sánh trình tự trong tin học dùng giải thuật Smith-Waterman bằng phương pháp quy hoạch động và phương pháp song song. Đưa ra một số kết quả tính toán trên các chuỗi có độ dài tăng dần bằng hai giải thuật trên. Nhận xét lợi ích dùng giải thuật tính toán song song trên thiết bị hỗ trợ tính toán song song GPU giúp giảm rất nhiều thời gian hơn so với dùng giải pháp tuần tự. Chương trình đã góp phần vào việc giảm thời gian tính toán bài toán so sánh các trình tự trong tin học. Đây là một trong số các bài toán cơ bản trong tin học luôn đòi hỏi thời gian tính toán lớn.

Bên cạnh những kết quả đạt được, luận văn này còn có những hạn chế sau: Chưa trình bày đầy đủ một số giải thuật cơ bản tuần tự sang giải thuật song song, số ví dụ giải thuật song song chưa nhiều. Chương trình tương tác với người sử dụng ở dạng dòng lệnh, chưa trực quan, sinh động.

Từ đó hướng nghiên cứu và triển khai tiếp theo của luận văn là:

Chương trình cần thử nghiệm trên các hệ thống thiết bị đồ họa có năng lực tính toán mạnh hơn, áp dụng với dữ liệu lớn hơn để đánh giá độ so khớp tin cậy hơn.

Nghiên cứu cải tiến thêm thuật toán để chương trình có thể chạy nhanh hơn, nhằm nâng cao hiệu suất và có thể tính toán với khối lượng đầu vào lớn.

Xây dựng giao diện đồ họa trực quan để dễ dàng tương tác với người dùng. Có thể chuyển ứng dụng này lên trang web nhằm giúp mọi người trong lĩnh vực tin sinh học có công cụ hỗ trợ để nghiên cứu.