

Giới thiệu môn học Lập trình song song

Trần Trung Kiên
ttkien@fit.hcmus.edu.vn

Cập nhật lần cuối: 16/09/2021



KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

fit@hcmus

Q: Tại sao ta lại muốn lập trình song song?

A: Để tăng tốc chương trình

Q: Lập trình song song so với lập trình tuần tự thì khó hơn hay dễ hơn?

A:

☐ Khó hơn 😞

☒ Lập trình tuần tự: chỉ phải điều khiển một đũa

☒ Lập trình song song: phải điều khiển cùng lúc nhiều đũa

☐ Nhưng cũng thú vị hơn 😊

Trong môn học, sẽ tập trung vào lập trình song song trên GPU (Graphics Processing Unit)

Yên tâm, nếu bạn không có GPU thì vẫn có thể học được ;-)

CPU vs GPU



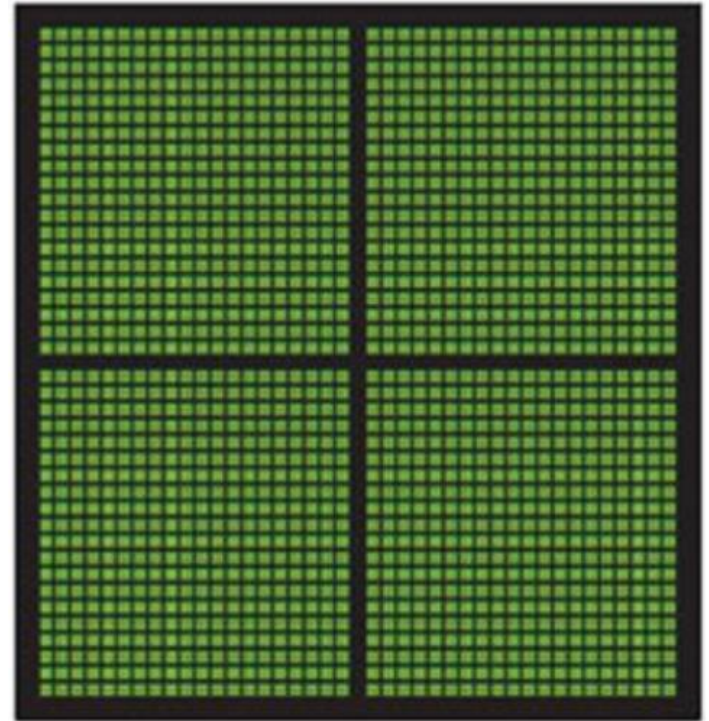
CPU

Có **một vài** core, mỗi core **mạnh** và **phức tạp**



GPU

Có **rất rất nhiều** core, mỗi core **yếu** và **đơn giản**



CPU

Có **một vài** core, mỗi core **mạnh và phức tạp**

Tập trung tối ưu hóa **độ trễ (latency)**; độ trễ = thời gian hoàn thành một công việc

VD: công việc là đưa một người từ địa điểm A đến địa điểm B cách nhau 4500 km

Xe hơi: 2 người, 200 km/h

Độ trễ = ? h

Băng thông = ? người/h

GPU

Có **rất rất nhiều** core, mỗi core **yếu và đơn giản**

Tập trung tối ưu hóa **băng thông (throughput)**; băng thông = số lượng công việc hoàn thành trong một đơn vị thời gian

Xe bus: 40 người, 50 km/h

Độ trễ = ? h

Băng thông = ? người/h

CPU

Có **một vài** core, mỗi core **mạnh và phức tạp**

Tập trung tối ưu hóa **độ trễ (latency)**; độ trễ = thời gian hoàn thành một công việc

VD: công việc là đưa một người từ địa điểm A đến địa điểm B cách nhau 4500 km

Xe hơi: 2 người, 200 km/h

Độ trễ = $4500/200 = 22.5$ h

Băng thông = $2/22.5 = 0.09$ người/h

GPU

Có **rất rất nhiều** core, mỗi core **yếu và đơn giản**

Tập trung tối ưu hóa **băng thông (throughput)**; băng thông = số lượng công việc hoàn thành trong một đơn vị thời gian

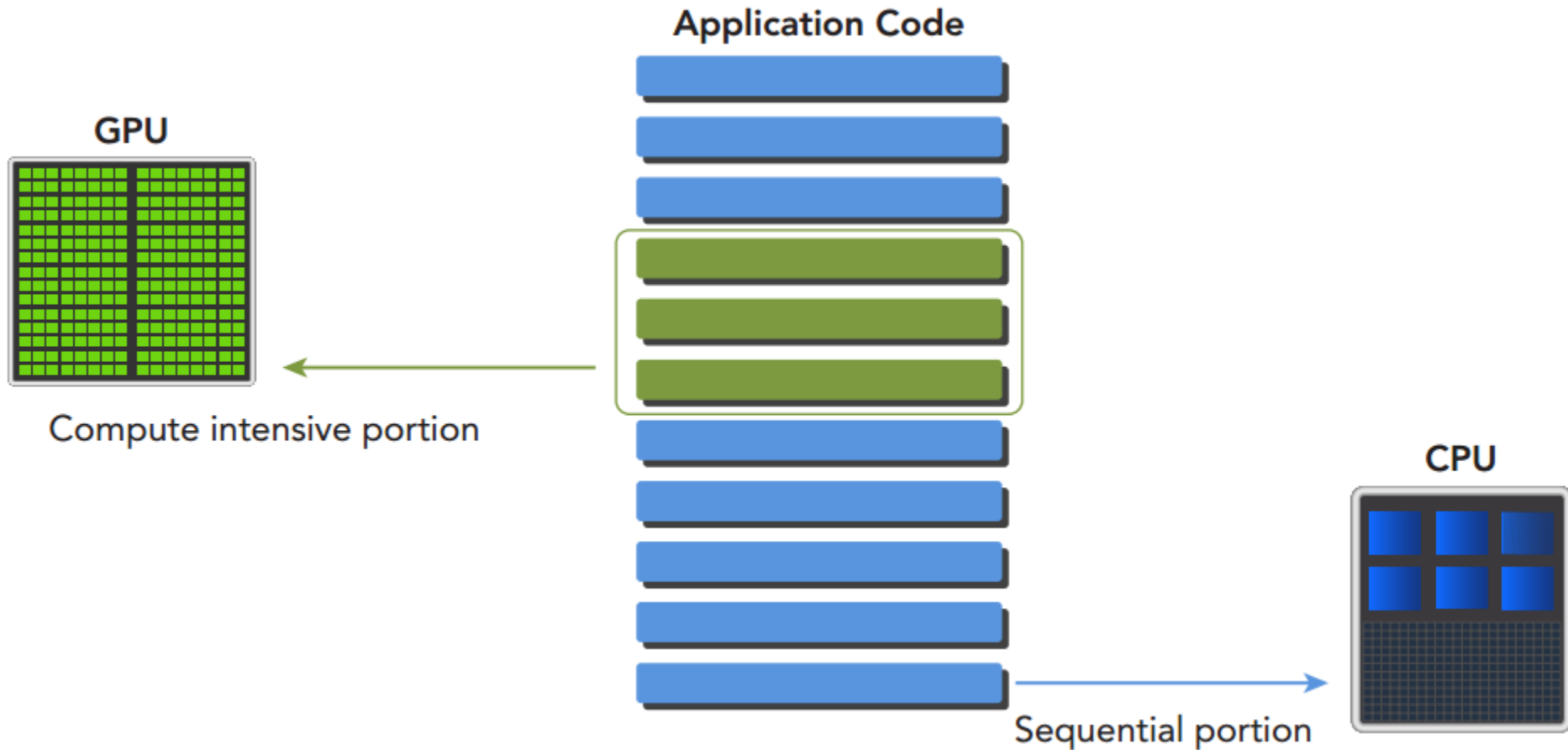
Xe bus: 40 người, 50 km/h

Độ trễ = $4500/50 = 90$ h

Băng thông = $40/90 = 0.44$ người/h

Vậy xe nào tốt hơn?

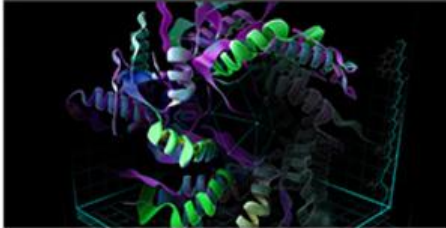
CPU + GPU



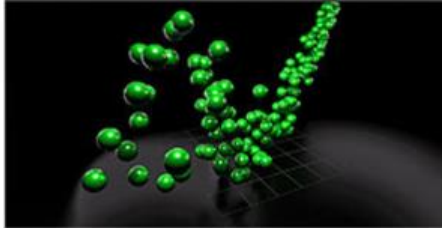
CUDA (Compute Unified Device Architecture) **C/C++** là ngôn ngữ C/C++ được mở rộng, cho phép viết chương trình chạy trên cả CPU và GPU (NVIDIA): những phần code tuần tự chạy trên CPU, *những phần code song song mức độ lớn chạy trên GPU*

Các lĩnh vực ứng dụng lập trình song song trên GPU

BIOINFORMATICS



COMPUATIONAL CHEMISTRY



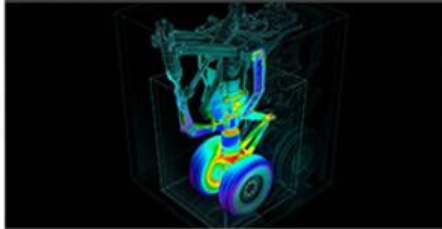
COMPUTATIONAL FINANCE



COMPUTATIONAL FLUID DYNAMICS



COMPUTATIONAL STRUCTURAL MECHANICS



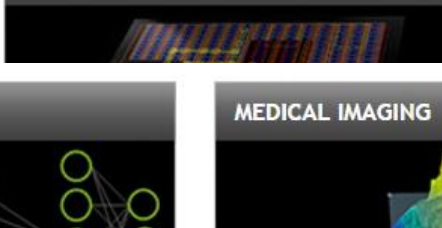
DATA SCIENCE



DEFENSE



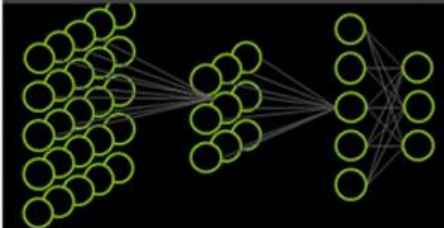
ELECTRIC DESIGN AUTOMATION



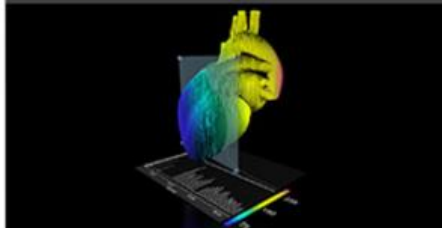
IMAGING & COMPUTER VISION



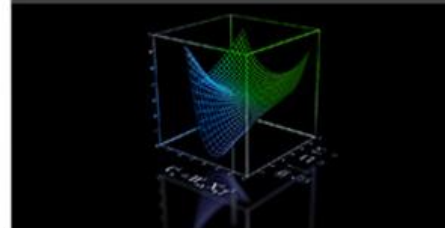
MACHINE LEARNING



MEDICAL IMAGING



NUMERICAL ANALYTICS



WEATHER AND CLIMATE








Nguồn ảnh: <http://www.nvidia.com/object/gpu-applications-domain.html>

Nội dung môn học:

- ☐ Giới thiệu CUDA; ví dụ: cộng véc-tơ, convolution, ...
- ☐ Cách thực thi song song trong CUDA; ví dụ: reduction, ...
- ☐ Các loại bộ nhớ trong CUDA; ví dụ: convolution, reduction, ...
- ☐ Ví dụ: scan, histogram, sort, ...
- ☐ Qui trình tối ưu hóa một chương trình CUDA; các chủ đề mở rộng (nếu có thời gian)

Sau khi học xong môn học này, SV có thể:

-  Song song hóa được các tác vụ thường gặp và cài đặt được bằng CUDA
-  Vận dụng được hiểu biết về cách thực thi song song trong CUDA để tăng tốc chương trình
-  Vận dụng được hiểu biết về các loại bộ nhớ trong CUDA để tăng tốc chương trình
-  Vận dụng được qui trình tối ưu hóa chương trình CUDA
-  Vận dụng được kỹ năng làm việc nhóm để hoàn thành đồ án nhóm trong môn học

Đánh giá môn học

- ☐ **Các bài tập cá nhân** (gồm cả lý thuyết & thực hành) trong suốt quá trình học: 50%
- ☐ **Đồ án nhóm** vấn đáp vào cuối kỳ: 50%

Đánh giá môn học

Nên nhớ mục tiêu chính ở đây là học, học một cách chân thật

Bạn có thể thảo luận ý tưởng với bạn khác cũng như là tham khảo các tài liệu, nhưng code và bài làm phải là của bạn, dựa trên sự hiểu của bạn

Nếu vi phạm thì sẽ bị 0 điểm cho toàn bộ môn học

Lời khuyên

- ☐ Cố gắng giữ một tinh thần tốt trong mọi hoàn cảnh, “find light in darkness”
- ☐ Cố gắng đơn giản hóa, tập trung vào những việc cần thiết, hạn chế những việc không cần thiết
- ☐ Cố gắng tham gia đầy đủ các buổi học, cố gắng tập trung khi học
- ☐ Cố gắng tham gia thảo luận khi học (khi học mình thường hay hỏi các câu hỏi)

Các tài liệu tham khảo của môn học

- ☐ David B. Kirk, Wen-mei W. Hwu. *Programming Massively Parallel Processors*. Morgan Kaufmann, 2016
- ☐ Cheng John, Max Grossman, and Ty McKercher. *Professional Cuda C Programming*. John Wiley & Sons, 2014
- ☐ Lê Hoài Bắc, Vũ Thanh Hưng, Trần Trung Kiên. *Lập trình song song trên GPU*. NXB KH & KT, 2015
- ☐ NVIDIA. [*Intro to Parallel Programming*](#). Udacity
- ☐ NVIDIA. [*CUDA Toolkit Documentation*](#)

Setup môi trường lập trình

- ☐ Đào đâu ra máy có GPU?
 - ☐ Google Colab: free và đã được cài sẵn CUDA ☺
 - ☐ Cho dù máy bạn có GPU thì bạn vẫn nên dùng Google Colab vì khi chấm bài Thầy sẽ dùng Google Colab
- ☐ Quy trình viết code, biên dịch và chạy:
 - ☐ Viết và lưu code (file .cu) ở máy cá nhân bằng một editor nào đó (với các editor không tự động nhận biết đuôi .cu, cách đơn giản nhất để có màu là chỉnh language/syntax là C/C++)
 - ☐ Mở notebook ở Colab, chọn “Runtime, Change runtime type, Hardware accelerator” là GPU, upload file .cu lên
 - ☐ Ở cell Colab, biên dịch: `!nvcc tên-file.cu -o tên-file-chạy`
Nếu không chỉ định tên-file-chạy thì mặc định sẽ là a.out
 - ☐ Ở cell Colab, chạy: `!./tên-file-chạy`
- ☐ Demo ...

Setup môi trường lập trình

Nếu máy bạn có GPU của NVIDIA (compute capability nên > 3) và bạn muốn dùng thì bạn xem hướng dẫn cài đặt CUDA Toolkit [ở đây](#) (mục “Installation Guides”)

Khi làm các bài tập, bạn có thể chạy trên GPU cá nhân, nhưng trước khi nộp bài cần test trên Google Colab vì khi chấm Thầy sẽ dùng Google Colab