GPUコンピューティング（CUDA)講習会

GPUとGPUを用いた計算の概要

丸山　直也

2009/10/28

スケジュール

- **13:20-13:50 「GPUを用いた計算の概要」**
  - **担当　丸山**
- **13:50-14:30 「GPUコンピューティングによるHPCアプリケーションの高速化の事例紹介」**
  - 担当　青木
- **14:30-14:40　休憩**
- **14:40-17:00 「CUDAプログラミングの基礎」**
  - 担当　丸山

TSUBAMEのTesla利用方法:ログイン

1. 端末(iMac)へのログイン
   - 配布した紙に記載されているID, passwordを利用
2. Titech2006もしくは「移動」ユーティリティを選択し、X11.appを起動（xtermの起動）
3. Tsubameへログイン

   > ssh –Y –t login名@login.cc.titech.ac.jp tesladebug

TSUBAMEのTesla利用方法：準備

- GSIC「TESLA利用の手引き」を参照
  - http://www.gsic.titech.ac.jp/~ccwww/tebiki/tesla/tesla.html
- CUDAインストールディレクトリへのパスを各種環境変数に追加
  - お使いのシェルにあわせて、source cuda_setup.{csh,sh} としてください
- 詳しくは利用の手引き4節を参照

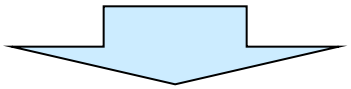
```
(csh系: cuda-setup.csh)
setenv PATH ${PATH}:/opt/cuda/bin
setenv LD_LIBRARY_PATH ${LD_LIBRARY_PATH}:/opt/cuda/lib
setenv MANPATH ${MANPATH}:/opt/cuda/man

(bash系: cuda-setup.sh)
export PATH=${PATH}:/opt/cuda/bin
export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:/opt/cuda/lib
export MANPATH=${MANPATH}:/opt/cuda/man
```

GPUコンピューティング

- GPUを一般アプリケーションの高速化に適用
  - GPUを計算アクセラレータとして利用
- GPGPU (General-Purpose Computing on GPU)とも言われる
- 2000年前半から研究としては存在。2007年にCUDAがリリースされてから大きな注目

計算加速器 (アクセラレータ)

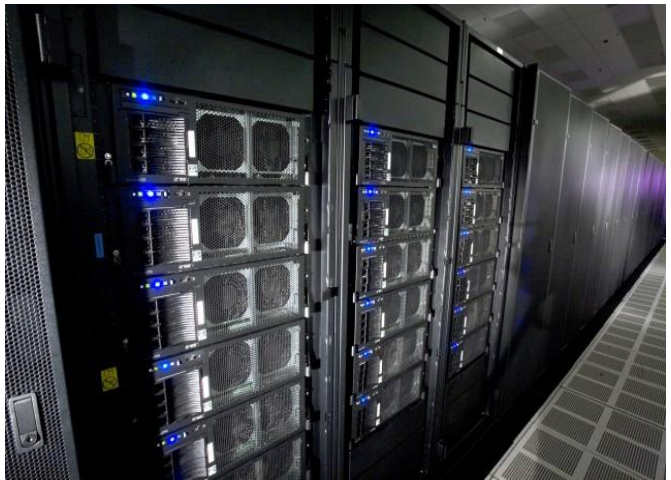
- Cell, GPU, GRAPE, ClearSpeed, FPGA, …
- 汎用CPUとは別に特定の計算のオフロードが可能なプロセッサ
- 汎用CPUと比較して高性能and/or低消費電力
- HPCではベクトル演算に特化したアクセラレータが注目



- ハイブリッドコンピューティング
  - 汎用CPUとアクセラレータの組み合わせ
  - HPCにおける最近の最もホットなトピックの1つ

例：Roadrunner at LANL

- Opteron + *PowerXCell 8i*
- 史上初ペタフロップ超えマシン
  - 1.105 PFLOPS (LINPACK)
- 現在世界最速スパコン
  - 2008年6月よりTOP500スーパーコンピュータランキングにて1位



例：TSUBAME @ 東工大GSIC

- Opteron (> 10K cores) + ClearSpeed (> 600) + NVIDIA Tesla (> 600)
- Peak: 170 TFLOPS (DP), Linpack: 87.01 TFLOPS (41st at Jun ’09 TOP500)

Tesla S1070

• 4 Tesla cards in a 1U node

• Connected to host machines via PCIe extension cables

TSUBAME 1.2. The most Heterogeneous Supercomputer in the world

- Three node configurations with four different processors → >30,000 cores, ~170TFlops system

SunFire X4600+ 2 TESLAs + ClearSpeed
- Opteron 2.4GHz 16 cores
- TESLA S1070 (30cores) 2boards
- ClearSpeed X620 (2cores) 1board

→ 78 cores, 330 Gflops peak

x 318nodes

SunFire X4600+ClearSpeed
- Opteron 2.4GHz 16 cores
- ClearSpeed X620 (2cores) 1board

→ 18 cores, 157 Gflops peak

x 330nodes

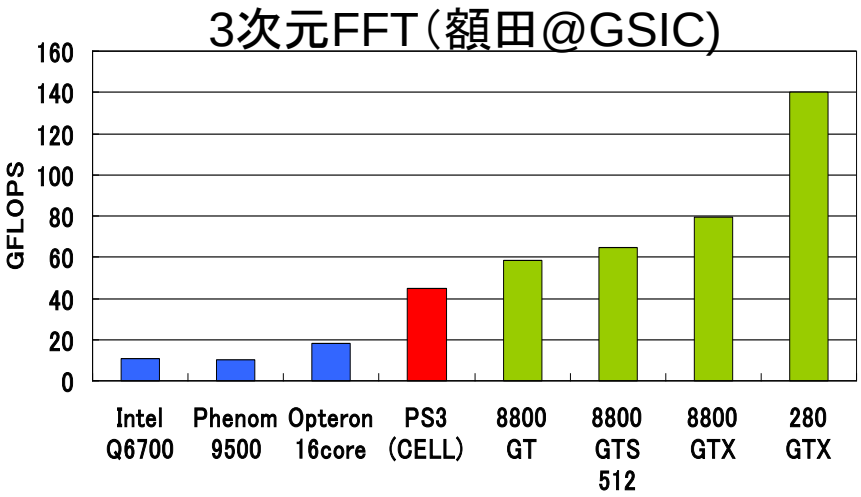
SunBlade X6250 (TSUBASA cluster)
- Xeon 2.83GHz 8 cores

→ 8 cores, 90.7 Gflops peak

x 90nodes

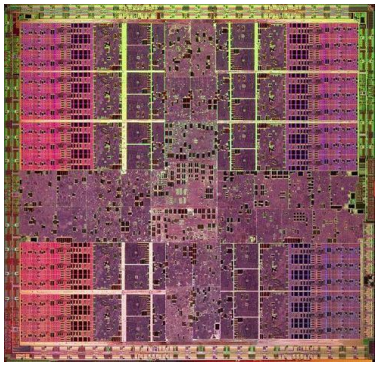
なぜGPU？

- CPUを大幅に上回る計算性能＋メモリバンド幅
  - Tesla
    - 1 TFOPS (SP) / 90 GFOPS (DP)
    - 100 GB/s
  - Core 2 Quad @ 3 GHz
    - 96 GFLOPS (SP) / 48 GFOPS (DP)
    - < 10 GB/s
- 多くのデータ並列なアプリ



Tesla 10 (T10)

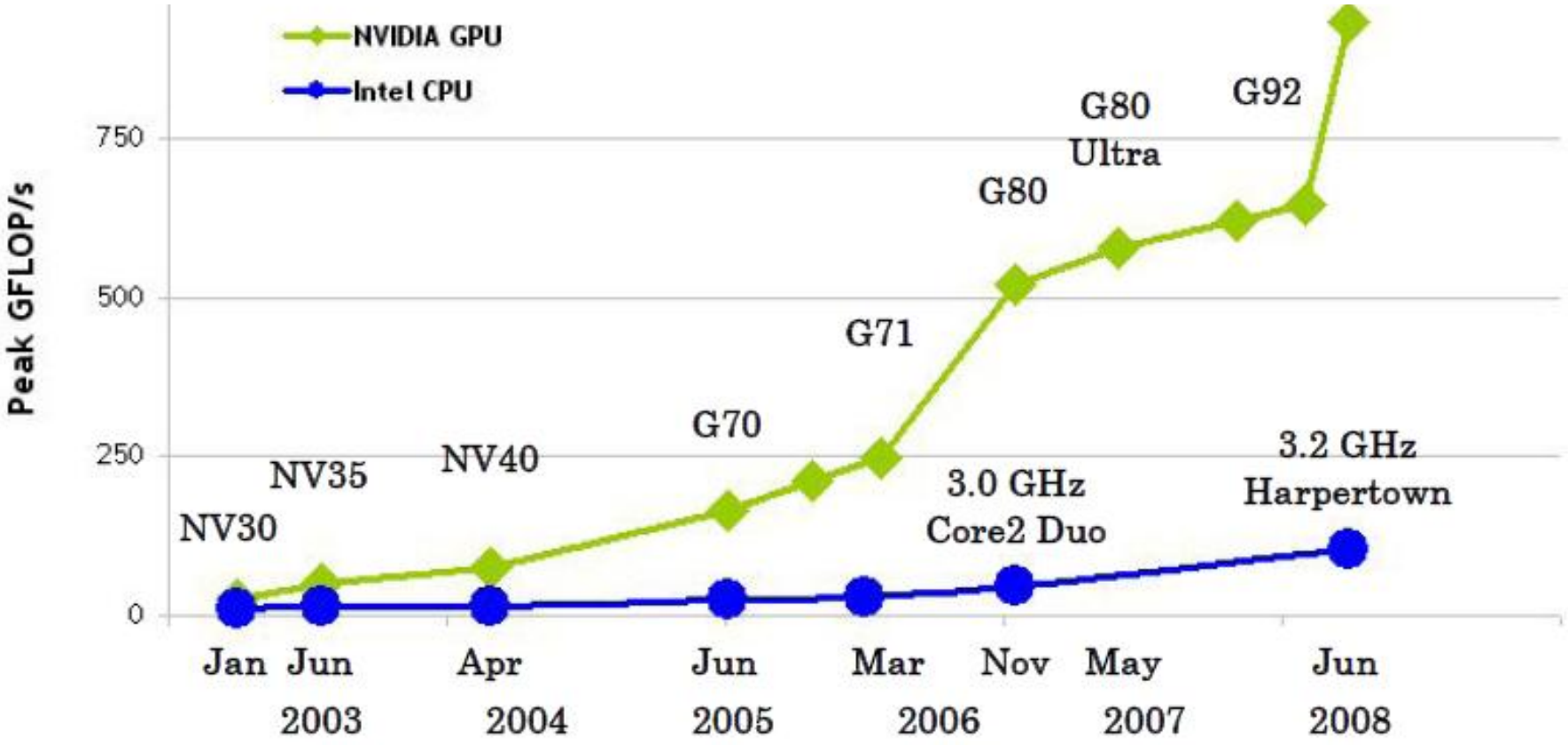
- NVIDIA G200系アーキテクチャによるHPC向けプロセッサ
  - コンシューマ向け → GeForce 280 GTX
- 240コア @ 1.29-1.44 GHz
- 4GB memory, 102 GB/s
- Peak: 1 TFLOPS (SP), 90 GFLOPS (DP)
- 製品
  - Tesla C1060: PCIe card
  - Tesla S1060: 1U system with 4 C1060 cards
- GeForceとの違い
  - ビデオ出力無し
  - 品質（NVIDIAによる全品検査 vs ボードメーカによるサンプル検査）
  - 価格 (C1060 @ $1,700, GTX 280 @ $400)



NVIDIA T10

性能トレンド

CUDA Programming Guideより



GT200 = GeForce GTX 280	G71 = GeForce 7900 GTX	NV35 = GeForce FX 5950 Ultra
G92 = GeForce 9800 GTX	G70 = GeForce 7800 GTX	NV30 = GeForce FX 5800
G80 = GeForce 8800 GTX	NV40 = GeForce 6800 Ultra	

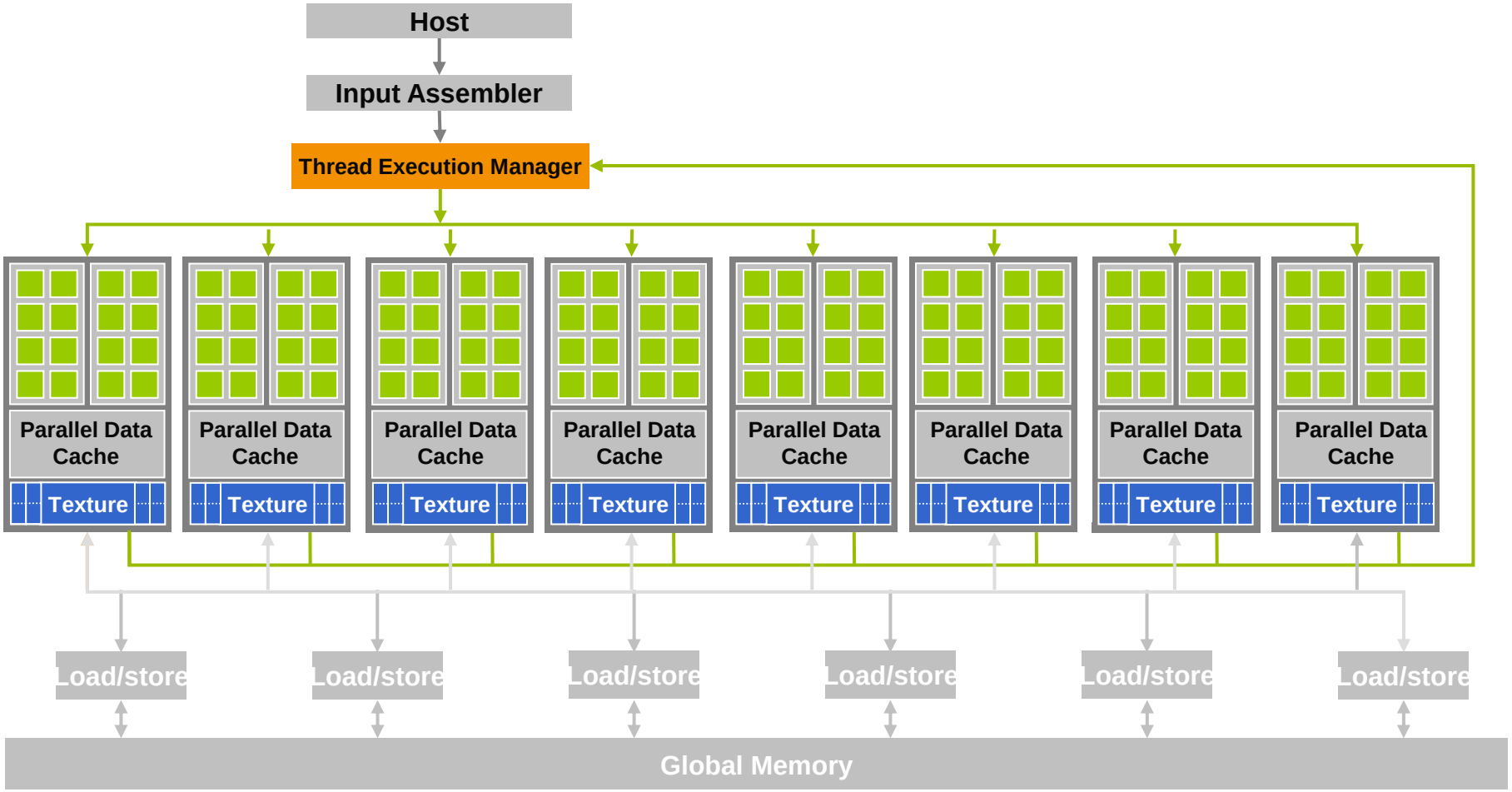
GPUコンピューティング

- GPUを一般アプリケーションの高速化に適用
  - GPU→アクセラレータと呼ばれるものの一種
- GPGPU (General-Purpose Computing on GPU)とも言われる
- 2000年前半から研究としては存在。2007年にCUDAがリリースされてから大きな注目

GPUコンピューティング : ハードウェア

- NVIDIA GPU
  - GeForceシリーズ :　一般のPCに搭載されているタイプで、比較的安価。GeForce 8800 GTXよりCUDAを実行可能
  - Teslaシリーズ:　GPUコンピューティング専用ハードウェア（ディスプレイ出力無し）。高価だがより高信頼（といわれている）。TSUBAMEに搭載
- AMD/ATI GPU
  - Radeonシリーズ
  - FireStreamシリーズ

GeForce 8800 GTX



2009/10/28

GPUコンピューティング：ソフトウェア

- NVIDIA CUDA
  - 2007年2月にNVIDIAが自社のGPU向けにリリース
  - C/C++の言語拡張
  - NVIDIAのGPU専用
  - 最も普及
- OpenCL
  - Appleによる提案に始まり、標準化団体により制定
  - 言語自体はベンダー非依存
  - Snow Leopardに標準搭載
  - NVIDIA/ATI GPU、x86 CPU向けSDKが利用可能
  - 普及はまだこれから
- その他
  - Brook/Brook+, RapidMind, DirectX Compute, etc.

OpenCL

- Khronos 標準化団体によって策定されたGPUコンピューティングのための共通仕様
  - KhronosはOpenGLを策定した団体
- 同一のソースコードでNVIDIA GPU、AMD GPU、Intel/AMD CPUなどで動作
- ただし、同一のプログラムがすべての環境に適しているわけではない
  - オンチップメモリのサイズ、ベクター長など
  - 特にベンダーの異なるGPU間では結局異なるプログラムを書くことに
- 本講習会で取り上げる内容は基本的にOpenCLプログラミングでも有効
  - 細かな技術的な違いはあるものの、概念的なレベルでは同じ

OpenCL vs. CUDA

- OpenCLの利点
  - ベンダー非依存（Intel Larrabeeでも動作？）
  - 業界標準
- CUDA の利点
  - NVIDIA GPUの最先端の機能を利用可能
    - CUDA3.0ではデバッガー等の機能も大幅に拡充
    - cf. OpenGL vs. DirectX
  - プログラミングの簡便さ
    - OpenCLはCUDAのドライバーAPIと同様にカーネル呼び出し等が煩雑
  - これまでの知識・経験の蓄積（OpenCLも普及すれば時間の問題）

DirectCompute

- Microsoft独自のDirectX 11に追加されたGPUコンピューティングのための仕様
  - OpenGL vs. DirectX
  - OpenCL vs. DirectCompute
- DirectX 11をサポートするGPU＋Windowsで利用可能
- CUDA、OpenCLとの違い
  - DirectXとよりシンプルかつ密な連携が可能（例えば3Dゲーム中のAIの計算などをGPUで高速化など）
  - Windowsのみだが、WindowsのDirectXにおける開発に慣れた人であればハードルはCUDA等より低い