2 東京工業大学 TSUBAME におけるアクセラレータ活用事例

遠藤敏夫（東京工業大学情報理工学研究科）

はじめに

高性能計算を実現するアーキテクチャとして，限られた電力やスペースで高い演算性能を実現可能なアクセラレータアーキテクチャが注目されている．近年のアクセラレータとしては，Sony/IBM/ 東芝の Cell Broadband Engine，ClearSpeed SIMD アクセラレータ，NVIDIA や AMD（ATI）のグラフィックプロセッサ（GPU）などが挙げられる．また Intel からリリースが予定されている Larrabee メニーコアプロセッサも，本稿の文脈ではアクセラレータと見なしてよいだろう．

アクセラレータ上の計算技術の研究は近年急速に増加しているが，その恩恵を，一般の計算機ユーザが受けられるためには依然多くのハードルが存在する．まず高性能を達成するための大規模アクセラレータ環境と，それを多数のユーザが共有するためのバッチキューシステムの対応が必要である．またアクセラレータを利用するためのアプリケーションレベルの対応や数値演算ライブラリ・プログラミング環境の整備が必要である．さらに，アクセラレータは汎用プロセッサと大きく異なるアーキテクチャを持つためにそれぞれ専用のプログラミングモデル・言語を持つ．そのような技術の浸透のためにはプログラミング講習会などの教育活動も必須となる．

アクセラレータを積極的に搭載した大規模計算機システムとしては，約 10,000 の Cell プロセッサを持ち，史上初のペタフロップスを達成した LANL の Roadrunner スーパーコンピュータや，本稿で述べる東京工業大学学術国際情報センター（GSIC）の TSUBAME スーパーコンピュータ[1]（図 -1）が挙げられる．TSUBAME は汎用プロセッサである AMD Opteron に加え，ClearSpeed Advance X620 アクセラレータを備えたシステムである．また 2008 年 10 月に，NVIDIA Tesla S1070 アクセラレータを搭載し，General Purpose GPU（GPGPU）技術の大規模な利用が可能となった．

本稿ではアクセラレータ技術の浸透に向けての GSIC を中心とした取り組みや，主に TSUBAME におけるアクセラレータの利用例について述べる．

図 -1　TSUBAME スーパーコンピュータ

東京工業大学 TSUBAME スパコン

TSUBAME は東京工業大学に 2006 年 4 月に導入されたシステムであり，現在東工大の内外の 1,000 人以上のユーザによって利用されている．システムは 655 台の SunFire X4600 計算ノードからなり，各ノードでは SUSE Enterprise Linux server OS が稼働している．計算ノードと 1.6 ペタバイト（raw capacity）のストレージが，InfiniBand により接続されている．

各ノードには 2.4GHz のデュアルコア Opteron が 8 個と 32GB のメモリが搭載される（一部ノードはプロセッサコアが 2.6GHz，メモリ容量が 64GB または 128GB である）．さらに ClearSpeed Advance X620 アクセラレータが 1 枚ずつ搭載される（図 -2）．

それに加えて約半数（320 ノード程度）には NVIDIA Tesla S1070 が接続される．S1070 は 1U タイプの筐体に 4 枚のアクセラレータ（GPU デバイス）が搭載された製品であり，TSUBAME においては 2 つの計算ノードと 1 つの S1070 が対応する．1 ノードあたり 2 枚の GPU が対応する．これらのアクセラレータの詳細につ

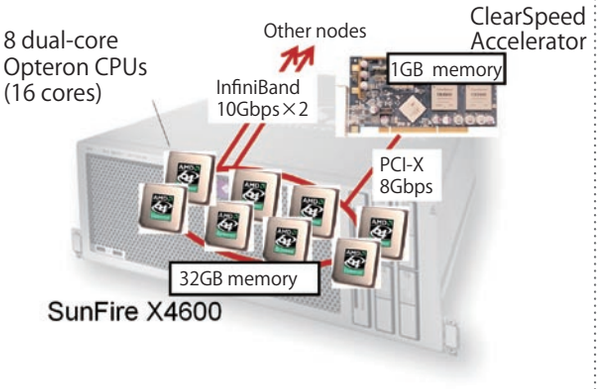

図 -2　OpteronとClearSpeedを搭載したTSUBAMEノードの構成

図 -4　ClearSpeed を装着したノードの内部

図 -3　ClearSpeed Advance X620 アクセラレータ

いては後述する．

なお，TSUBAME も Roadrunner も，汎用プロセッサとアクセラレータの双方を持つシステムであるが，そのバランスは異なる．Roadrunner では Cell プロセッサ（PowerXCell 8i）を 12,960 個に対して Opteron が 12,960 コアと，アクセラレータと汎用プロセッサコアが同数になっている．TSUBAME では，ClearSpeed が 648 枚，Tesla が 680 枚に対して Opteron が 10,480 コアと，汎用プロセッサコアの方がはるかに多い．TSUBAME では，既存ソフトウェア資産をより重視しているといえる．

ClearSpeed アクセラレータ

TSUBAME のほとんどすべてのノードに ClearSpeed Advance X620 アクセラレータ（図 -3）が 1 枚ずつ搭載されている．これは PCI-X スロットに装着されるカード型のアクセラレータである．図 -4 はノード内部に装着された様子を示す．

ClearSpeed アクセラレータの主な性能を以下に示す．

- プロセッサ：CSX600×2 プロセッサ
- ピーク速度性能：80.6GFlops（倍精度，210MHz 動作時）
- メモリ容量：1GB
- メモリバンド幅：6.4Gbytes/s
- 消費電力：約 25W

特徴は低い消費電力と高い速度性能である．アクセラレータを除いた TSUBAME ノードが，約 1kW の消費電力で，ピーク速度性能 76.8GFlops であるのに比べると，約 2% の電力で同等の速度性能を持つことになる．

アクセラレータのメモリは計算ノードのメモリと独立しており，アクセラレータで計算するデータは前もって PCI-X バスを介してアクセラレータ側に送っておく必要がある．PCI-X バスの速度は 1Gbytes/s である．

アクセラレータ上のプログラミングは，C 言語を SIMD 拡張した，Cⁿ プログラミング言語[2]を用いて行う．各 CSX600 プロセッサは 96 並列の SIMD プロセッサであり，アーキテクチャを意識したプログラミングを行うことになる．図 -5 は非常に単純な Cⁿ プログラムを示す．‘poly’ つきで宣言された変数は，長さ 96 の暗黙的な配列を意味し，その演算は並列に行われる．図中最後の行の ‘c=a+b;’ は，長さ 96 の配列同士の加算を意味する．

ClearSpeed アクセラレータの利用方法は，主に以下のようになる．

- Cⁿ プログラミング言語の利用：利用者は ClearSpeed アクセラレータ用のプログラムを作成し，インタラクティブに実行またはバッチキューで実行することができる．前述のようにアーキテクチャを意識したプログラミングを行う必要がある．また，ホスト側のメモリ

```
{
    poly int a, b, c;
    poly int i;
    i = get_penum(); /* i = 0..95 */
    a = i;
    b = i*i;
    c = a+b;
}
```

図 -5　簡単な C^n プログラムコードの例

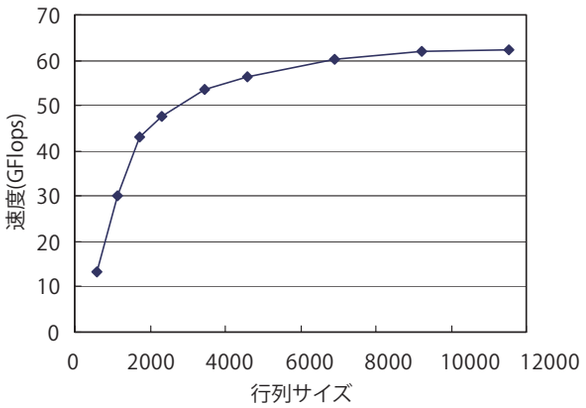

図 -6　CSXL ライブラリの行列積関数の性能

とアクセラレータ側のメモリの間の通信についてもプログラミングすることになる．

- 数値演算ライブラリの利用：アクセラレータ上で動作する BLAS（線形演算）ライブラリ，FFT（高速フーリエ変換）ライブラリが用意されている．BLAS ライブラリである CSXL は，ホスト用の通常の BLAS ライブラリと同様のインタフェースを持ち，透過的に利用することができる．ホスト―アクセラレータ間の通信についてもライブラリ内で自動的に行われる．そのため，MATLAB や Mathematica などの BLAS を利用するアプリケーションについて，CSXL ライブラリを利用するように設定するだけで，簡単にアクセラレータによる加速を利用することができる．

その性能については，問題サイズに大きく影響されるので注意を要する．図 -6 は，CSXL ライブラリ 3.11 の倍精度行列積（DGEMM）の性能を示す．正方行列どうしの積で，横軸は行列の一辺のサイズ，縦軸は速度性能を GFlops で表す．行列が十分に大きいときは約 62GFlops の速度であり，これは TSUBAME の Opteron コア 1 つの約 14 倍にあたる．一方行列が小さい場合は性能が大きく下がる．行列が大きいほど，PCI-X を介した通信コストが，演算コストに対して相対的に小さくなるためである．

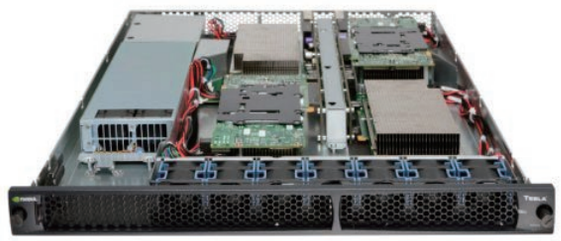

図 -7　Tesla S1070

- 移植されたアプリケーション：TSUBAME では分子動力学のアプリケーションである AMBER を利用することができるが，一部の演算手法について ClearSpeed に移植されている．移植された AMBER9 の GB1, GB2 手法の場合を測定したところ Opteron コア 1 つの約 2.6 倍の速度であった．

以上のように ClearSpeed アクセラレータを利用することができる．速度については，計算の性質や問題サイズに大きく影響を受けることが分かる．たとえば行列演算においては行列サイズと PCI-X 通信コストの関係に留意する必要がある．自分でプログラミングを行う場合には，アクセラレータのメモリの再利用を最大限に行い，通信を削減することが望ましいであろう．この議論は，次に述べる Tesla においても同様に成り立つ．

NVIDIA Tesla アクセラレータ

2008 年度に GSIC は，NVIDIA 社の Tesla S1070 アクセラレータ（図 -7）を 170 台導入した．S1070 は 1U のラックマウント筐体に 4 枚の GPU 技術に基づいたアクセラレータデバイスを含んでいる．アクセラレータとホストは PCI-Express のインタフェースカードおよびケーブルで接続される．TSUBAME においては，S1070 は計算ノードのラック中のノード間のすきまに設置されている（図 -8）．2 つのアクセラレータが 1 本の PCI-Express ケーブルに対応し 1 台の計算ノード（ホスト）に接続されている（図 -9）（一部のホストは 4 つのアクセラレータに接続される）．接続方式は 2Gbytes/s の PCI-Express 1.0 x8 であるが，これは TSUBAME ホスト側の制限によるものであり，Tesla S1070 自身は

図 -8　TSUBAME ラックに設置された Tesla（写真中央）の様子

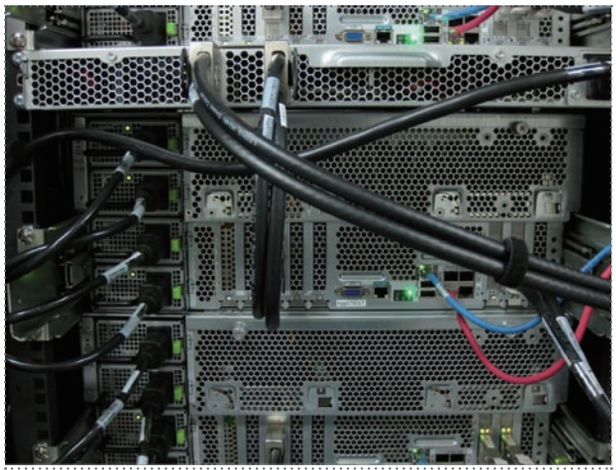

図 -9　ラックの背面で，Tesla（写真上部）とホストが接続されている様子

PCI-Express 2.0 x16 まで対応している．

Tesla S1070 に含まれるアクセラレータの主な性能を以下に示す．

- プロセッサ：Tesla T10 GPU（ストリーミングマルチプロセッサ 30 基）
- ピーク速度性能：86.4GFlops（倍精度），1.04TFlops（単精度）
- メモリ容量：4GB
- メモリバンド幅：102Gbytes/s
- 消費電力：約 175W（筐体あたり約 700W）

大きな特徴は，高い単精度浮動小数演算性能と，メモリバンド幅である．高いメモリバンド幅により，局所性の抽出の困難な計算，たとえば高速フーリエ変換においても高い性能を達成することができる [3]．また，過去の多くの GPU はハードウェアによる倍精度演算に対応しなかったが，S1070 を含め新しい世代のものは対応する．

ClearSpeed 同様，アクセラレータのメモリはホストメモリと別個なので，メモリ間のデータ移動についてもプログラミングする必要がある．Tesla 上のプログラミングは，CUDA プログラミング言語 [4] を用いて行う．これは C 言語が拡張されたものであり，特徴の 1 つはホスト上の関数とアクセラレータ上で動作する関数を同一のソースファイルに記述できることである．

図 -10 は単純な CUDA プログラムコードの例を示す（アクセラレータ上の関数のみ）．func 関数は並列に実行され，その並列度は呼び出し側で自由に設定できる．ポインタ a, b, c はアクセラレータ上のデバイスメモリと呼ばれる領域を指す．デバイスメモリはすべてのストリーミングマルチプロセッサからアクセスすることができる．ほかに，ストリーミングマルチプロセッサごとのメモリも存在し，明示的に扱うことになる．詳細はプログラミングガイド等を参照されたい．

```
__global__ void func(int *a, int *b,
                     int *c)
{
    int i;
    i = threadIdx.x;
    a[i] = i;
    b[i] = i*i;
    c[i] = a+b;
}
```

図 -10　単純な CUDA プログラムコードの例

TSUBAME 上の Tesla は本稿執筆時点では導入されたばかりであるが，Tesla の大きな利点は，世界中で利用が広まっている NVIDIA GPU におけるソフトウェア資源を用いることができる点である．東京工業大学においてもすでに，GPU 上の高性能計算の研究が数多くなされている．たとえば 3 次元タンパク質ドッキングの研究（計算工学専攻 秋山泰教授・GSIC 松岡聡教授ら）や，気候・海洋などの大規模流体シミュレーションの研究が進行している（GSIC 青木尊之教授ら）．前者に関しては畳み込み演算・フーリエ変換が重要な演算となるが，GPU の高いメモリバンド幅を活用して効率的に行えることが分かっている．後者に関連しては，理研ベンチマークという Poisson 方程式を解くベンチマークを，CUDA 言語を用いて移植し，4 枚の GPU（NVIDIA GeForce 8800Ultra）上で実行したところ，問題サイズ S で 51.9GFlops を達成したことが報告されている [5]．この結果は汎用プロセッサの約 50 倍の速度であり，2007 年度理研ベンチマークコンテストで優勝している．TSUBAME への Tesla 導入により，これらのような研

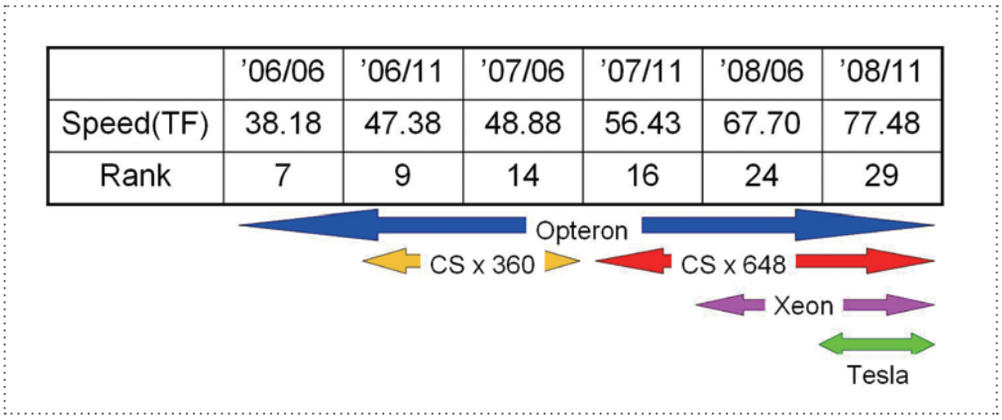

図-11　Top500 ランキングにおける TSUBAME の性能と順位．アクセラレータの活用により性能を伸ばしている．

究を強力に推進できると期待できる．GPU に基づくアクセラレータであるため，パソコンショップで購入した GPU 用に作られたソフトウェアが，そのまま（再チューニングすることがより望ましいとはいえ）メモリサイズが大きく，多数のアクセラレータを持つ大規模環境へと移行可能なためである．

汎用プロセッサとアクセラレータを併用した Linpack 実行

これまでに述べてきたアクセラレータの主な利用方法は，アクセラレータを単独で用いるものか，複数だが 1 種類のアクセラレータを用いるものであった．では，異なる種類のアクセラレータや，アクセラレータと汎用プロセッサを併用して演算を行うことにより，どれくらい性能が得られるだろうか？　ここでは，筆者らが TSUBAME 上の Opteron と ClearSpeed，Tesla をほぼすべて用いて Linpack ベンチマークを実行した結果について報告する（松岡聡教授，GSIC 額田彰研究員らとの共同研究）．

Linpack は密行列連立一次方程式を直接法で解くベンチマークである．スーパーコンピュータの著名な世界ランキングである Top500（http://www.top500.org）の順位は，このベンチマークの速度で決められている．Top500 においては，利用するプログラムは自由，解くべき行列サイズも自由とされる．我々は，よく知られた Linpack の MPI 並列実装である High performance Linpack（HPL）[6] をベースに，アクセラレータ向けの改造・チューニングを行った（詳細は文献 7）を参照されたい）．

HPL においては，実行のほとんどの時間は密行列どうしの積の演算に費やされる．その部分では BLAS（basic linear algebra subprogram）ライブラリの行列積関数 DGEMM が呼び出されており，以下に高速な DGEMM 関数を用いるかがポイントの 1 つとなる．我々はその部分において，汎用プロセッサとアクセラレータを使い分けることとした．しかしそれだけでは汎用プロセッサとアクセラレータの性能に差があるため，並列プログラムでは遅いほうに足を引っ張られてしまい，速度向上は望めない．そのために我々は，各プロセスに対してほぼ均等な性能の計算資源が割り当てられるよう，プロセスとプロセッサ（汎用プロセッサ，アクセラレータの双方）のマッピングを調節した．より具体的には，各プロセスは自分が担当する部分行列のうち，一部を（スレッド並列化された）Opteron 用 BLAS で計算し，一部をアクセラレータで計算する．その割合をプロセスごとに調整している．本研究で用いた BLAS ライブラリは以下の通りである．Opteron および後述の Xeon においては GotoBLAS[8]，ClearSpeed アクセラレータにおいては前述の CSXL を用いた．一方 Tesla アクセラレータのためには我々のグループで開発した行列積関数を用いている．

ほかにも，ホスト―アクセラレータ間の通信の影響を軽減するためのブロックサイズのチューニング，TSUBAME ネットワーク上流の混雑を避けるためのプロセスマッピングなどを行っている．

図-11 は TSUBAME 上の Linpack 性能と Top500 における順位を示している．Top500 は年 2 回発表されるが，TSUBAME 初登場時には約 38TFlops で世界 7 位であったが，このときはアクセラレータを利用せず，Opteron プロセッサのみであった．図中下部の矢印は，利用したプロセッサの種類を表す（CS は ClearSpeed）．2 回目以降はアクセラレータの併用，追加などにより毎回性能を伸ばしている．なお，ハードウェア構成が変わらずに性能が伸びている回もあるが，これはライブラリのバージョンアップや再チューニングなどによるものである．なお図中の Xeon というのは，Xeon プロセッサ 720 コアを持つクラスタシステムを併用したことを表している．このクラスタは，東京工業大学グローバル COE「計算世界観の深化と展開」により導入さ

れたもので，GSIC に設置され，TSUBAME と高速な InfiniBand ネットワークで接続されている．

このように，Top500 において 6 回連続で性能向上させたというのは，世界的にもきわめて異例となっている．一方，順位については，世界中でスーパーコンピュータが構築され競い合う現状で，下がっていることが分かる．それでも，汎用プロセッサとアクセラレータを併用するヘテロ型スパコンとしては，RoadRunner に次ぐ世界 2 位(初登場時は 1 位)となっている．

TSUBAME のバッチキューシステムのアクセラレータへの対応

TSUBAME のように汎用プロセッサとアクセラレータを搭載するシステムにおいては，マルチユーザでそれらの資源を共有するために，バッチキューシステムのレベルでの対応も必要となる．その目的は，各アクセラレータを利用するジョブ数を正しく 1 つに制御することと，ノード内のプロセッサ利用ジョブとアクセラレータ利用ジョブの間の干渉を最小限にすることである．

まず，基本的な（アクセラレータを含まない）TSUBAME のバッチキュー構成について簡単に述べる．Sun N1 Grid Engine のカスタマイズされた版が用いられている．詳細については GSIC Web ページ（http://www.gsic.titech.ac.jp）を参照されたい．大別して以下の 2 種類のキューが提供され，それぞれ異なるノード集合に対応している．

- 性能保証サービスキュー（SLA キュー）：ジョブへの計算資源の割り当てはノード単位で行われる．複数ジョブが同一ノードに混在することはない．仮に 1 プロセッサコアしか使わないジョブであっても，1 ノード(16 コア)を占有することになる．
- ベストエフォートサービスキュー（BES キュー）：ジョブへの計算資源の割り当てはプロセッサコア単位で行われる．複数ジョブが同一ノードで混在し得る．

以下では断らない限り ClearSpeed をバッチキューを介して利用する場合について述べる．SLA キューでアクセラレータを利用するのは比較的容易である．各ジョブはノードを占有するので，そのジョブがプロセッサを使おうがアクセラレータを使おうが，バッチキューシステムが関知する必要はない．

ただし Tesla の場合は，装着ノードと非装着ノードが混在する問題がある．Tesla 装着ノードからなる，Tesla_SLA キューが設置され，SLA キューとノードを一部共有している．

BES キューはより複雑である．まずアクセラレータを利用するジョブを，1 プロセッサコアを使うジョブと同様に扱おうとすると，以下のように問題が起きる．BES キューでは同一ノードに複数ジョブが割り当てられ得るため，複数のアクセラレータ利用ジョブが同一ノードに割り当てられる可能性があり，それらのジョブの実行は失敗するか，遅くなってしまう．

GSIC で採用された方法は，既存の BES キューと別に，アクセラレータ専用のキューが，前者とノードを共有して設置されている，というものである．アクセラレータ専用のキューを介して投入されたジョブは，ノードあたり 1 つしか実行されない．さらに，アクセラレータを利用するジョブに対しては，プロセッサコアを 1 つ以上割り当ててやる必要がある．これはアクセラレータを利用するためには計算ノードとアクセラレータの PCI 通信が発生し，そのためにプロセッサを利用するためである．その場合でも，アクセラレータ利用ジョブが走行していないノードでは，BES キューから 16 プロセッサコアのすべてが利用可能になっている．このように，すべての計算資源を有効に利用するようになっている．

アクセラレータ技術の教育への取り組み

これまで述べてきたように，アクセラレータを用いた研究は多く進行している．今よりもさらにアクセラレータ技術の裾野を広げ，特に応用分野の研究者・学生がアクセラレータ技術を利用してそれぞれの研究へフィードバックするためには，利用技術の教育が必要不可欠である．計算によってはライブラリ層の利用でも対応可能だが，それが難しい応用のためにはやはり，アクセラレータ上のプログラミングの教育が必要となる．これまでに C^n 言語や CUDA 言語について触れてきた．これらは MPI, OpenMP などの汎用プロセッサ向け並列プログラミング環境とも異なる概念を含むため，とっつきにくい面がある（むろん，既存の並列プログラミングを知っていればはるかに早く理解できる）．特に十分高速なプログラムを作成するためには各アクセラレータのアーキテクチャ，特にメモリ階層の知識・それに基づくチューニングが必要となる．

このような教育の一環として，2007 年度には ClearSpeed 社の技術者を講師とする C^n 言語の講習会を GSIC で開催した．講習会には，量子化学や流体などの応用分野から計算機科学まで広い分野の研究者が参加した．

また CUDA 言語については，グローバル COE「計算世界観の深化と展開」の 2008 年度の講義「計算数理実践 — HPC —」において取り上げている．この講義は計

算機科学だけでなく応用数理・数学などの分野の大学院生も対象とし，高性能計算の実践的な技術を身につけることを目的としている．MPI や OpenMP などについての講義に加え，教室から TSUBAME にログインし，並列プログラミング実習を行っている．その一環として CUDA 言語について取り上げ，TSUBAME に導入された Tesla を用いた実習も行っている．

おわりに

東京工業大学におけるアクセラレータの活用の現状について，TSUBAME スーパーコンピュータに搭載された ClearSpeed アクセラレータ，NVIDIA Tesla アクセラレータについて述べた．また多数のユーザがそれらを利用するためのシステムからのサポートについて簡単に述べた．アクセラレータ技術を広い分野の研究者が手軽に利用するためには，使いやすいプログラミング環境とその教育が必要不可欠である．その点で NVIDIA CUDA 言語はこれまでの GPU 上でのプログラミングよりはるかに簡明であり，その登場は大きなインパクトを与えた．しかしながら，この分野にはメーカをまたいだ統一的な環境がない問題がある．Cell, ClearSpeed, NVIDIA GPU, AMD/ATI GPU それぞれにおいて異なるプログラミング言語やライブラリが提供されている．最近，それらの異なるアクセラレータ，さらにマルチコア汎用 CPU を含めた統一的なプログラミング環境として OpenCL（http://www.khronos.org）の規格が制定されようとしている．そのような動向も注視しつつ，アクセラレータ技術の活用・教育を推進していきたい．

謝辞　TSUBAME およびアクセラレータの運用に尽力されている，日本電気（株），サン・マイクロシステムズ（株），ClearSpeed Technology Inc.，NVIDIA Corporation，東京工業大学学術国際情報センターの皆様に深く感謝いたします．

参考文献

1) 松岡　聡：TSUBAME の飛翔：ペタスケールへ向けた「みんなのスパコン」の構築，情報処理学会研究報告 2006-HPC-107 (SWoPP06), pp.37-42 (2006).
2) ClearSpeed Technology Inc. : Introductory Programming Manual, https://support.clearspeed.com
3) Nukada, A., Ogata, Y., Endo, T. and Matsuoka, S. : Bandwidth Intensive 3-D FFT Kernel for GPUs Using CUDA, In Proceedings of IEEE/ACM International Conference for Supercomputing (SC08) (Nov. 2008).
4) NVIDIA Corporation : NVIDIA CUDA Compute Unified Device Architecture Programming Guide Version 2.0, http://www.nvidia.com/object/cuda_home.html
5) 小川　慧，青木尊之：CUDA による定常反復 Poisson ベンチマークの高速化，情報処理学会研究報告 2008-HPC-115, pp.19-23 (2008).
6) Petitet, A. et al. : HPL - A Portable Implementation of the High-Performance Linpack Benchmark for Distributed - Memory Computers, http://www.netlib.org/benchmark/hpl/
7) Endo, T. and Matsuoka, S. : Massive Supercomputing Coping with Heterogeneity of Modern Accelerators, In Proc of IEEE IPDPS, (2008).
8) Goto, K. : GotoBLAS, http://www.tacc.utexas.edu/resources/software/

(平成 20 年 12 月 30 日受付)

遠藤敏夫(正会員)
endo@is.titech.ac.jp
東京工業大学情報理工学研究科グローバル COE「計算世界観の深化と展開」特任准教授．博士（理学）．主に分散処理やヘテロ型アーキテクチャ上での並列計算の研究に従事．日本ソフトウェア科学会，ACM，IEEE-CS 各会員．