異種アクセラレータを持つヘテロ型スーパーコンピュータ上のLinpackの性能向上手法

遠藤敏夫[†1,†2] 額田彰[†1,†2]
松岡聡[†1,†3,†2] 丸山直也[†1,†2]

グラフィックプロセッサ (GPU) と SIMD 型 ClearSpeed アクセラレータをを備えたヘテロ型スパコンである東工大 TSUBAME における Linpack ベンチマークの実行について報告する．TSUBAME の約 10000 の Opteron コア，640 の Xeon コア，648 基の ClearSpeed アクセラレータ，624 基の NVIDIA Tesla GPU を全て用いた Linpack 実行において，87TFlops を達成した．本論文ではまずこの結果を得るためのチューニングや負荷分散について述べる．一方でピーク性能は 163TFlops であり，ピークに対する実行効率は 53%と，他のシステムより低くなっている．このピークと Linpack 性能の乖離の理由についても，システムアーキテクチャの特質から議論する．

Linpack Tuning Method on a Heterogeneous Supercomputer with Hybrid Accelerators

Toshio Endo,[†1,†2] Akira Nukada,[†1,†2]
Satoshi Matsuoka[†1,†3,†2] and Naoya Maruyama [†1,†2]

We report Linpack benchmark results on the TSUBAME supercomputer, a large scale heterogenous system with graphics processing units (GPUs) and ClearSpeed SIMD accelerators. With all of about 10,000 Opteron cores, 640 Xeon cores, 648 ClearSpeed accelerators and 624 NVIDIA Tesla GPUs, we have achieved 87TFlops. This paper describes careful tuning and load balancing method required to achieve this performance. On the other hand, since the peak speed is 163 TFlops, the efficiency is 53%, which is slower than other systems. This paper also discusses the reason of this gap from the aspect of system architecture.

1. はじめに

高い演算性能を限られた電力やスペースで実現可能なアーキテクチャとしてアクセラレータが注目されている．多くは SIMD アーキテクチャ，もしくはそれらの複数構成をとっており，近年のアクセラレータとしては，Sony/IBM/東芝の Cell Broadband Engine，NVIDIA や AMD(ATI) のグラフィックプロセッサ (GPU)，ClearSpeed SIMD アクセラレータなどが挙げられる．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おり，複数の GPU を用いた並列流体演算の性能[9),10)] などについてもすでに報告されている．また複数アクセラレータのプログラミングを容易にすることを目的とした試みも行われている[7),8)]．一方，数百台以上のオーダのアクセラレータを用いた大規模ヘテロ型システムのスケーラビリティを実証した例は，我々の以前の報告[4),12)] や LANL RoadRunner を用いた研究[5)] を除き依然少ない．

これまでに我々は東京工業大学学術国際情報センター (GSIC) のスーパーコンピュータ TSUBAME 上において，Opteron CPU コア，Xeon CPU コア，ClearSpeed アクセラレータ，NVIDIA Tesla GPU という 4 種のプロセッサを混在させ，Linpack ベンチマークにおいて 77.48TFlops という性能を得たことを報告した[11)]．その後，さらなるアルゴリズムの改良や実行時の工夫により 87.01TFlops を達成しており (図 1)，それについて報告する．この結果は 2009 年 6 月の Top500 スパコンランキング[3)] で 41 位にランクされ，異種のプロセッサを用いたヘテロ型システムとしては Cell プロセッサ RoadRunner に次ぐ世界二位である．

本稿ではさらに Linpack の性能 (R_{peak}) と理論性能値 (R_{max}) の差について議論する．今回の TSUBAME 上の測定においては理論性能値は 163.2TFlops であり，実行効率 R_{peak}/R_{max} は $87.01/163.2 = 53.3\%$ である．Top500 上位の多くのシステムでは 75–85%であり，アクセラレータを用いた RoadRunner の 76%に比べても低くなっている．この理由については，TSUBAME アーキテクチャにおける PCI 通信オーバヘッド，ノード間のヘテロ性など複数が考えられるため，それぞれの影響について段階的な実験を通して詳細な議論を行う．

†1 東京工業大学
Tokyo Institute of Technology
†2 JST, CREST
†3 国立情報学研究所
National Institute of Informatics

	Jun06	Nov06	Jun07	Nov07	Jun08	Nov08	Jun09
Speed(TF)	38.18	47.38	48.88	56.43	67.70	77.48	87.01
Rank	7	9	14	16	24	29	41

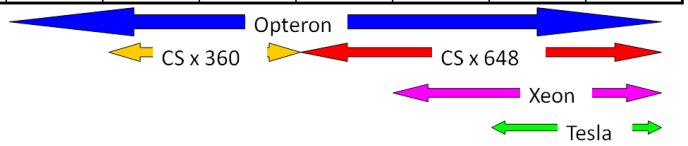

図 1 Top500 における TSUBAME の Linpack 性能と順位の経緯.

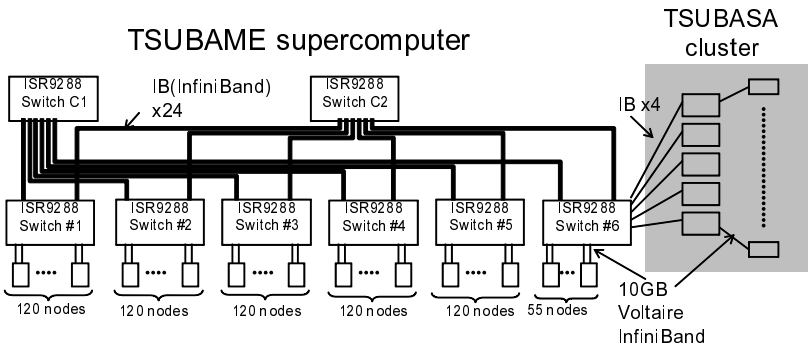

図 2 TSUBAME のネットワーク構成. 右方の網掛け部は tsubasa クラスタ

2. TSUBAME システム

TSUBAME では，655 ノードの計算サーバ SunFire X4600 と，合計 1.6PBytes のストレージサーバが InifiniBand により接続されている．以下，本論文に関連の深い部分について概要を示す.

TSUBAME 計算ノード: TSUBAME の各計算ノードは，dual core 2.4GHz Opteron 880 を 8 個を持ち，16 CPU core が 32GB のメモリを共有する．またノードは InifiniBand host channel adapter (HCA) を 2 つ持つ．オペレーティングシステムは 64bit 対応 SuSE Linux Enterprise Server 10 である．主要な I/O スロットは PCI-X および PCI-Express 1.0 x8 であり，ここに HCA および後述のアクセラレータが接続されている.

InfiniBand インターコネクト: 各ノードは 2 本の 10Gbps SDR InfiniBand により 288 ポートの Voltaire ISR9288 スイッチ群に接続される (図 2)．スイッチ間は，InifiniBand 24 本により接続される．並列プログラムを動作させるために，Message passing interface(MPI) の一実装である Voltaire MPI が利用可能となっており，本稿でもそれを用いる.

ClearSpeed アクセラレータ: 各計算ノードは，PCI-X バスによって接続される ClearSpeed アクセラレータボード X620[1] を一枚ずつ持つ．各アクセラレータは 2 つの CSX600 SIMD プロセッサと 1GB の DRAM(メモリバンド幅 6.4Gbytes/s) を持つ．各プロセッサには，420MFlops(倍精度) の演算能力を持つ 96 個の PE が含まれ，アクセラレータの理論性能は 80.64GFlops となる．なお，アクセラレータ上の演算の入出力データは，1.06GBytes/s の PCI-X バスを介してホストと通信する必要がある．アクセラレータあたりの消費電力は約 25W である.

アクセラレータを利用する手段として，SIMD 並列プログラミング言語 C^n，基本線形演算を行う CSXL ライブラリ，高速フーリエ変換を行う CSFFT ライブラリなどが提供されている．本稿ではこれらのうち CSXL ライブラリを利用する.

Tesla アクセラレータ: NVIDIA Tesla S1070 は，1U の筐体に 4 つのアクセラレータデバイス (グラフィックボードと同等) を含んでいる．システム全体では 170 筐体，680 デバイスが導入されている．TSUBAME 計算ノードのうち 316 ノードが Tesla と接続されており，基本的にノードあたり 2 デバイスが接続されている$^{\star 1}$．ノードと Tesla 筐体は PCI-Express gen1 x8 のインタフェースカードおよびケーブルで接続される.

各 Tesla デバイスは Tesla T10GPU を持ち，その中にはストリーミングマルチプロセッサ (SM) が 30 基存在する．また SM から共有され，102Gbytes/s のメモリバンド幅を持つ 4GB のデバイスメモリが搭載されている．各デバイスのピーク速度は，倍精度浮動小数演算では 86.4GFlops，単精度では 1.04TFlops である．消費電力については筐体あたり約 700W であり, デバイスあたり約 175W となる．アーキテクチャの詳細については NVIDIA の公開情報[2] を参照されたい.

Tesla の利用のためには CUDA プログラミング環境が提供されており，拡張された C 言語によるプログラミングを行うことができる．また BLAS ライブラリである CUBLAS，フーリエ変換ライブラリである CUFFT が提供されているが，本研究では CUBLAS を使わず

$\star 1$ 一部のノードは 4 デバイスと接続されているが，今回はノードあたり最大 2 デバイス利用している

に独自に後述するカーネルを作成した．

tsubasa クラスタ: TSUBAME とは別のシステムとして，tsubasa と呼ばれる Xeon クラスタが設置されており，TSUBAME とは 20 本の InfiniBand(計 200Gbps) で接続されている．図 2 の網掛け部が TSUBASA クラスタである．本研究ではこのクラスタも TSUBAME と協調させて Linpack を実行する．

このクラスタは 90 ノードの Sun Blade X6250 からなり，各ノードは Quad core Xeon E5440 (2.83GHz) を 2 つ，計 8CPU コアを持つ．メモリ容量はノードあたり 8GB または 16GB である．各ノードは SDR InfiniBand(10Gbps) でネットワークと接続される．

2.1 ヘテロ性について

本研究では以上のような計算資源をほぼ全て併用して並列プログラムを動作させる．Opteron，ClearSpeed，NVIDIA，Xeon という 4 種類のプロセッサが混在することになる (**ノード内ヘテロ性**)．また，ノード毎のコンフィグレーションに注目すると，以下の 3 種類が混在している (**ノード間ヘテロ性**)．

Tesla あり TSUBAME ノード: 16 コアの Opteron，1 つの ClearSpeed，2 つの Tesla デバイスを持つ．

Tesla なし TSUBAME ノード: 16 コアの Opteron と 1 つの ClearSpeed を持つ．

TSUBASA ノード: 8 コアの Xeon を持つ．

3. Linpack の概要

本稿では Linpack の良く知られた並列実装である High performance Linpack (HPL)[6] を，ソースコードの一部改変して実行に用いる．HPL は正方密行列を係数とする連立一次方程式をブロック化ガウス消去法で解く，MPI 並列ソフトウェアである．指定された行列サイズ N に対して乱数行列を生成し，方程式を解き，その速度を Flops 値で評価する．

計算に参加するプロセス群は概念的にサイズ $P \times Q$ のプロセス格子を形成し，行列はプロセス格子に従って二次元ブロックサイクリック方式で分散される．以下，行列サイズを N，ブロックサイズを B とする．計算のほとんどの部分をガウス消去法が占め，その各ステップ (ステップ番号 k とする) は，以下のような処理からなる．

パネル分解: 第 k ブロック列はパネル列と呼ばれ，その箇所の LU 分解を部分ピボット選択を用いて行う．

パネルブロードキャスト: パネル列の各ブロックの内容を他プロセスへブロードキャストする．ここではプロセス格子の各行内での通信が発生する．

行交換通信: 部分ピボット選択の結果に基づき，行交換を行う．ここではプロセス格子の各列内での通信が発生する．

更新計算: パネル列と，行交換後の第 k ブロック行の内容を用い，行列の未分解部分の更新計算を行う．

以上のうち，パネル分解の計算量総計は $O(N^2B)$，パネルブロードキャストと行交換通信の通信量総計は $O(N^2(P+Q))$，更新計算の計算量総計は $O(N^3)$ である．このことから，最も時間がかかるのは更新計算であり，その傾向は N が大きい程強いと分かる．そのため，並列 Linpack ベンチマークにおいて良い性能を得るためには，N をメモリ量の限界に近づけるように大きくとり，高速な行列積 (DGEMM) を行う BLAS 数値演算ライブラリを用いることが一般的に行われている．また行列は各 MPI プロセスにほぼ均等に分割されるので，ヘテロ型システムにおいても各プロセスの演算性能は均等であることが望ましい．

4. ヘテロ型システムにおける設計と実装

4.1 システムアーキテクチャからの議論

TSUBAME における Linpack の設計・実装については既論文で報告済[11),12)] であるが，ここでは概要を改めて示し，またヘテロ型システムである RoadRunner 上での設計[5] との比較について議論する．RoadRunner においては，各ノードが Opteron を 4 コアと Cell プロセッサの一種である PowerXCell 8i を 4 つ搭載する．システムは 3240 ノードからなり，ピーク性能 1.457PFlops，Linpack 性能 1.105PFlops を達成している．RoadRunner と TSUBAME はいずれも汎用 CPU とアクセラレータを持つヘテロ型システムである．しかしシステムアーキテクチャの差異のため，両システムでは大きく異なる Linpack 実装方針をとっている．

カーネル演算の主体: カーネル演算である行列積 (DGEMM) のために，汎用 CPU，ClearSpeed, Tesla の全てを使うこととした．これは Cell プロセッサのみを用いる RoadRunner とは異なる．RoadRunner においては演算性能の 96%(ピークの割合) を Cell が占めるのに対して，TSUBAME においては汎用 CPU(Opteron と Xeon) が 35%，ClearSpeed が 32%，Tesla が 33%を占めるため，いずれかのプロセッサ種類の利用をやめると大きな性能ロスが生じるためである．

行列データの配置場所: Linpack においては $N \times N$ の行列データを MPI プロセスに分散し保持させる．一方前述の通り，メモリサイズに収まる範囲で N が大きいほうが高性能のために望ましい．RoadRunner においては，CPU のみがアクセスできるホスト

メモリと Cell のみがアクセスするデバイスメモリの大きさは，どちらもノードあたり 16GB で同じとなっている．そのため行列データをデバイスメモリに置くという方針をとっている．一方 TSUBAME においてはホストメモリは 32GB，デバイスメモリは 2 つの Tesla と 1 つの ClearSpeed で 4GB+4GB+1GB となっており，Tesla つきノードでさえデバイスメモリの方がはるかに小さい．そのため TSUBAME では行列データをホストメモリに配置することとした[⋆1]．このときアクセラレータの演算の際に PCI-X/PCI-express 通信 (以下は単に PCI 通信と記述する) が必要である．

ノード間ヘテロ性への対応: TSUBAME においては上記の通りノード間の性能が異なる．その状況で各 MPI プロセスの演算性能をバランスさせる必要がある．RoadRunner ではそのような困難はない．

4.2 TSUBAME 上の実装

上述の通りカーネル演算には CPU, ClearSpeed, Tesla の全てを利用する．一方 MPI 通信やピボット選択などについては，オリジナルの実装通り CPU のみで行う．よって HPL の実装のうち DGEMM 呼び出しが主要な変更箇所となる．具体的には，DGEMM が呼ばれたとき，その内部で CPU と各アクセラレータに適切な割合で処理を分割し，CPU/アクセラレータそれぞれの BLAS に実質的な仕事をさせる．この負荷分散の割合については予備実験を通して手作業で決定している．

ノード間ヘテロ性については以下のように対応する．HPL では各プロセスに均一の部分行列を持たせ，各プロセスがほぼ均一の速度でそれへの処理を行うことを仮定している．この構造を保つために以下のようにする．性能の高いノードではより多くのプロセスを起動し，結果としてどのプロセスもほぼ均等の性能のプロセッサ群 (CPU, アクセラレータ) を利用するようにする．プロセス数はノードごとの合計 DGEMM 性能になるべく比例するようにし，今回の実験においては，Tesla あり TSUBAME ノードでは 4，Tesla なし TSUBAME ノードでは 2，tsubasa ノードでは 1 とした．

図 3 は Tesla あり TSUBAME ノードのプロセッサを，4 つのプロセスがカーネル実行時にどう使い分けるかの概念図である．色がプロセスに対応する．OS による予期しないスケジューリングを最小限にするため，sched_setaffinity システムコールでプロセスと CPU コアの対応付けを行っている．

⋆1 厳密には TSUBAME においてもデバイスメモリに収まるように行列サイズ N を設定した場合との比較を行うべきであるが，未実装である．ただし 1GB の ClearSpeed デバイスメモリに問題サイズを合わせると大きく性能低下すると予想される

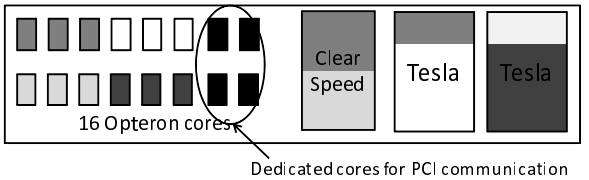

図 3 Tesla あり TSUBAME ノードにおけるプロセスとプロセッサ (Opteron, ClearSpeed, Tesla) の対応

4.3 チューニング

TSUBAME において高性能を実現するためにはアクセラレータの特性を考慮したチューニングが必要である．ここではそのうちのいくつかを述べるが，後の議論のために既発表のものも含める．

- アクセラレータ上の DGEMM 性能はブロックサイズ B に大きく影響される．更新計算においては $(M' \times B) \times (B \times N')$ のサイズの行列積演算が最も時間を消費する．ここで M', N' は各プロセスが持つ行列サイズに比例し，通常 B より大きい．この演算の計算量は $O(M'N'B)$ であるのに対して，PCI 通信量は入出力データサイズに比例するので $O(M'N' + M'B + N'B)$ である．通信コストを計算で十分に隠すには B が十分大きい必要がある．その他の条件としては (1) B が大きすぎるとパネル分解のコストや負荷の不均衡が大きくなること，(2) ClearSpeed においては B は 288 の倍数であるときに良好な性能であることがあり，予備実験の結果 $B = 1152$ を採用した．
  このブロックサイズは RoadRunner における $B = 128$ や，TSUBAME で Opteron のみで測定した際の $B = 240$ に比べると数倍大きい．行列データをデバイスメモリに配置する RoadRunner の場合と異なり，TSUBAME では DGEMM のたびに PCI 通信を行うため，それを抑える必要がある．
- アクセラレータと CPU の両方を DGEMM のために用いるが，全ての CPU コアを用いるのではなく，PCI 通信自身も CPU へ負荷をかけるため，その処理のために CPU コアを確保し，DGEMM には用いない．そのコアは行列データのホストメモリ上へのコピーのためにも用いる．ClearSpeed のために 2 コア，Tesla のために 1 デバイスあたり 1 コアを確保した．そのため Tesla あり TSUBAME ノードにおいては計 4 コアを確保し (図 3 の黒色のコア)，残り 12 コアを DGEMM に用いる．

他にも，以下のようなチューニングを行うことにより，2008 年 11 月発表時の 77TFlops から 87TFlops への性能向上が達成された．HPL 側のチューニング以外にも，Tesla 上の

DGEMM カーネルの性能向上を行っており，またノードの安定度が増し，利用ノード数をやや増加させた (ピーク性能で約 1%) ことなども影響している．

- 3 節で述べた HPL の処理のうち，行交換通信と更新計算のオーバラップを行った．各プロセスの更新対象行列を列方向に二分して処理を行った．(1) 左半分の行交換通信，(2) 左半分の更新計算，(3) 右半分の行交換通信，(4) 右半分の更新計算という処理が必要になるが，(2) と (3) は依存関係がないため同時に行うことができる．
- 我々の手法では，1 アクセラレータが複数 MPI プロセスにより共有されるケースがある．図 3 においては，3 つのアクセラレータはそれぞれ 2 プロセスによって利用される．この調停のために当初は，単純に各プロセスが投げる DGEMM の要求を FIFO で処理させていたが，以下のような問題が起こった．各プロセスが要求する DGEMM の問題サイズは，大きいもの (更新計算中の最も時間がかかる DGEMM) と小さいもの (パネル分解の途中で起こる場合など) が混在する．単純な FIFO では小さい DGEMM が，他プロセスの大きい DGEMM のために長時間待たされるケースが発生した．これを解消するために，大きい DGEMM の実行中に小さい DGEMM が割り込み，優先的に処理させるようにした．

5. Linpack 測定結果と考察

5.1 Linpack 測定結果

TSUBAME 全体を用いた Linpack 測定を，2009 年 3 月のシステムメンテナンス時に行った．利用したソフトウェア環境は，Voltaire MPI，GCC 4.1.2 である．BLAS ライブラリとしては，Opteron と Xeon においては GotoBLAS 1.26，ClearSpeed においては CSXL 3.11，Tesla においては新規に実装した DGEMM/DTRSM 関数を用いた．用いたノードは以下の通りである: (1)Tesla あり TSUBAME ノード 312 台，(2)Tesla なし TSUBAME ノード 336 台，(3)tsubasa ノード 80 台．MPI プロセス数は $312 \times 4 + 336 \times 2 + 80 \times 1 = 2000$ であり，サイズ 40×50 のプロセスグリッドを構成した．

この実験により $R_{max} = 87.01$TFlops を達成した．これは Opteron のみを用いた場合の 38.18TFlops に比べ 2.28 倍の性能である．前述の通り，この結果は 2009 年 6 月の Top500 において 41 位にランクされ，ヘテロ型システムとしては RoadRunner に次ぐ 2 位である．

図 4 に，理論性能と Linpack 性能における各プロセッサの割合を示す．理論性能においては前述の通り CPU と Xeon が 35%，ClearSpeed が 32%, Tesla が 33%を占める．Linpack 性能におけるプロセッサごとの内訳は，各 MPI プロセスにおける DGEMM の仕事の負荷分散から求めることができる．グラフから，理論性能と似た割合となっていることが分かる．

グラフはさらに消費電力の割合も示す．この電力の割合は以下のように求めた．まず Linpack の実行中に，TSUBAME ノード (Tesla あり・なし一台ずつ) と，Tesla S1070 筐体一台の電力をワットチェッカーで測定した．そして，TSUBAME ノードの電力から ClearSpeed の電力 (後述) を除いた値をノード数倍したものを，Opteron による消費電力とした．Tesla についても測定値を筐体倍した．ClearSpeed，Xeon(Sun X6250 ノード) については実測が困難だったためカタログの”typical power”の値を用いた．なお本グラフにはネットワーク・冷房の電力は含まれない．性能と電力の割合を比較すると，アクセラレータの優れた性能電力比が見てとれる．Linpack 性能においてはアクセラレータは 66%の性能貢献をしているが，消費電力は全体の 15%に抑えられている．ClearSpeed と Tesla を比較すると，前者の性能あたりの電力は約 1/7 となっている[⋆1]．ただし Linpack 以外のよりメモリバンド幅を必要とするアプリケーションにおいては，10 倍以上のメモリバンド幅を持つ Tesla の方が有利になると考えられる．

システム全体の理論性能値の合計 R_{peak} は 163.2TFlops であり，実行効率 R_{max}/R_{peak} は 53.5%である．この効率は同じヘテロ型システムである RoadRunner の 76%よりも小さい．次節ではこの乖離の原因について考察を行う．

5.2 性能の考察

TSUBAME 上の Linpack 性能がピーク性能と乖離する原因については，PCI 通信オーバヘッドやノード間のヘテロ性など複数考えられる．またそれらにはヘテロ性システムの設計上本質的でない点も含まれる．ここではそれぞれの影響を明らかにすることを試みる．主に注目するのは DGEMM の性能であり，図 5 のグラフを通して議論する．このグラフの左端は理論性能，右端は Linpack 性能を表し，それぞれプロセッサごとの内訳を示す．また参考として，TSUBAME の Opteron のみを用いた場合を図 6 に示す．理論性能と Linpack 性能の差を，段階的な小規模実験により考察する．

まずグラフの”core DGEMM”は，各プロセッサごとに DGEMM を実行したときの性能を求め，それぞれをコア数/デバイス数倍したものである．Opteron, Xeon においては 1 コアで GotoBLAS を実行したときの性能である，4.48GFlops(Opteron) と 10.74GFlops(Xeon) を基に算出した．なお，4.3 節で述べた PCI 通信用コアはここでは考えず，全 CPU コア数

⋆1 ただし ClearSpeed はカタログ値であるし，Tesla は独立筐体のために AC-DC 変換ロスなどを含む．そのため，厳密な比較ではない

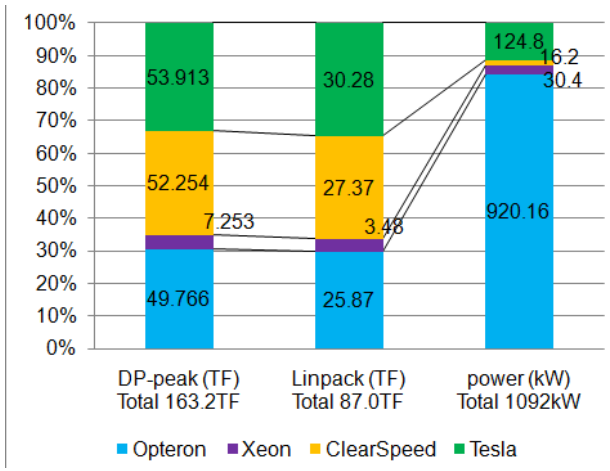

図 4 理論性能，Linpack 性能，消費電力における各プロセッサ種類の割合.

倍された値がグラフに示されている．Tesla においては我々の DGEMM 関数を 1GPU 上で on-board で実行したときの性能，つまり PCI 通信の影響を含まない性能 (80.33GFlops) を基にした．ClearSpeed では CSXL 3.11 の on-board DGEMM 関数の，CSX600 プロセッサひとつ (ボード 1 枚ではない) 性能である 32.15GFlops を基にした．いずれの場合も，DGEMM に与える行列サイズは Linpack 中の呼び出しの中で最大のものとしている．つまり，短辺の長さはブロックサイズ $B = 1152$ となる．各プロセッサの内訳をみると，DGEMM 単体の性能は Opteron, Xeon, Tesla においては理論性能の 93–95%であるのに対して，ClearSpeed では 80%と低くなっているのが分かる．システム全体では，理論値の 89%となっている．

次に”node DGEMM”は，1 ノード内のプロセッサを同時に用いて DGEMM を実行した際の性能を基に算出した性能である．ここにはノード内の PCI バスやメモリアクセスの衝突の影響が含まれる．また Linpack 実行時同様，PCI 通信用に CPU コアを確保する．ノードごとの測定結果は，Tesla あり TSUBAME ノードで 262.4GFlops，Tesla なし TSUBAME ノードで 121.2GFlops，tsubasa ノードで 85.86GFlops であった．プロセッサ種類毎の内訳を見ると，Opteron 部分の性能が大きく下がり，”core DGEMM”より 22%低下している．一方で図 6 の”core DGEMM”と”node DGEMM”の差は 1%であり，これと比べると PCI 通信用にコアを確保した影響は大きいといえる．なお行列データを全てデバイスメモリに置くことができる RoadRunner で同様の実験を行ったとすると，PCI 通信が起こらないため”core DGEMM” と”node DGEMM”の差は非常に小さいと考えられる．

”node hetero”は，ノード間ヘテロ性を考慮した性能である．我々の Linpack 実行では，DGEMM 性能が 262.4GFlops である Tesla ありノードに 4 プロセス，121.2GFlops である Tesla なしノードに 2 プロセスを起動する．Linpack 実行では最も遅いプロセスに足並みがそろうため，Tesla ありノードにおいても，有効な性能は $121.2/2 \times 4 = 242.4$GFlops となり，20GFlops 分の性能は無駄になってしまう．システム全体ではこの影響で 6.4%の性能を切り捨てていることになる．Opteron のみの場合や RoadRunner ではこの低下は起こらない．

これまでの性能値は，Linpack においてステップが進むごとに行列更新領域サイズが小さくなるという性質が反映されていない．またブロックサイクリック分割ではプロセス間の仕事の不均衡が起こる点も反映されていなかった．グラフ中の”BLAS size”はそれらの点を考慮し，Linpack 実行中に起こる DGEMM 呼び出しの問題サイズを再現したものである[⋆1]．実験は少数ノード (4 ノード) で行い，その結果を基にシステム全体の性能を算出した．グラフから，DGEMM サイズの縮小とプロセス間不均衡の影響により 12%の性能低下が起こっていることが分かる．一方で Opteron のみの場合では 5%であり，両者の違いは以下のように考えられる：(1) DGEMM サイズが小さくなる場合，アクセラレータでは PCI 通信コストが相対的に増大し，性能が下がる，(2) Opteron のみの場合のほうがブロックサイズが小さい (B=240) ので，プロセス間不均衡の影響がより小さい[⋆2]．

”BLAS size”と”Linpack”の差は，Linpack 実行中のカーネル演算以外，つまり MPI 通信やパネル分解などのコストとなる．システム全体ではこの差は 19%となっている．また Opteron のみの実行の場合は 13%となっている．この差のより詳細な内訳に興味があるが，そのためには大規模な検証実験が必要と考えられ，データは取れていない．

以上のことから，TSUBAME の Linpack 実行においては，DGEMM カーネル自体のオーバヘッド，PCI 通信のオーバヘッド，ノード間ヘテロ性，ブロックサイズが大きいことによる不均衡の影響がそれぞれ無視できず，全体として理論値の半分近くとなっていることが分かる．これらのオーバヘッドの全てがヘテロ型システムに本質的というわけではない．たと

⋆1 この実験では DGEMM に加え DTRSM も呼び出している
⋆2 なお Opteron のみで B=1152 として実験した場合，”node hetero” と”BLAS size”の差は 8%であった

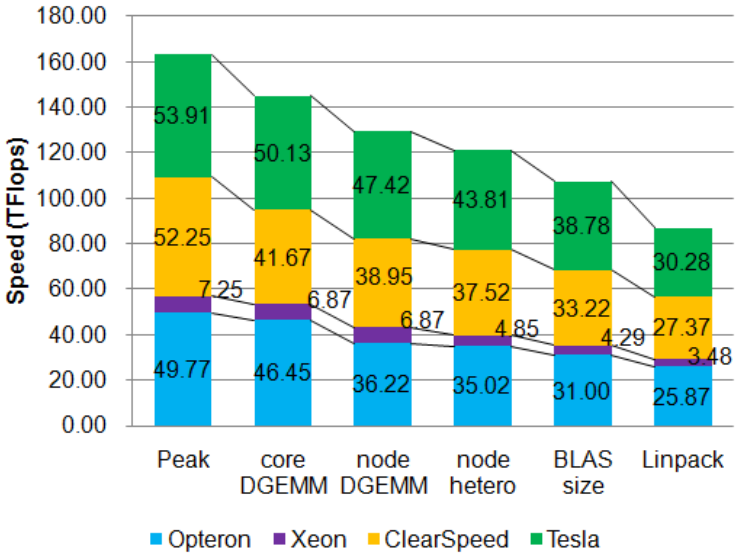

図 5 TSUBAME における性能における各プロセッサ種類の割合の詳細 (ブロックサイズ B=1152).

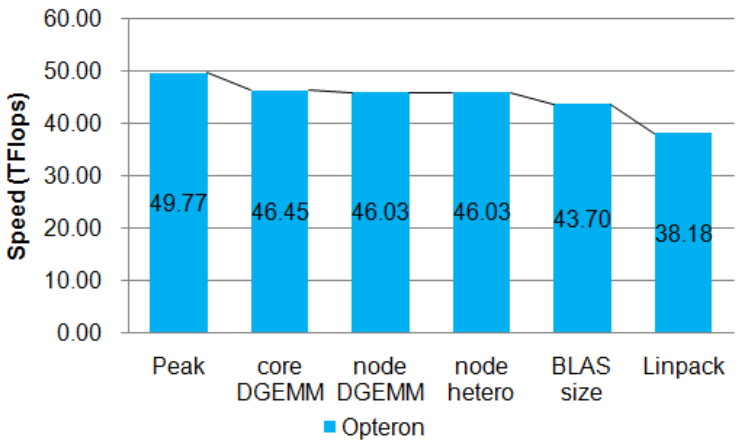

図 6 TSUBAME の Opteron のみ利用時の性能の詳細 (ブロックサイズ B=240).

えばアクセラレータが全ノードに均等に配置されていればノード間ヘテロ性の影響はなくなるし，RoadRunner のようにデータをデバイス側に置くことができれば PCI 通信のオーバヘッドははるかに小さくなる．

6. おわりに

TSUBAME スーパーコンピュータにおいて，10,000 コア以上の汎用プロセッサと，Tesla と ClearSpeed という異種アクセラレータを合計 1200 デバイス以上用いて Linpack を実行し，87.01TFlops の性能を達成した．新たに導入したチューニング手法を述べ，ヘテロ型システムである RoadRunner 上の実装との差異を議論した．また，理論性能と Linpack 性能の乖離の理由について，段階的な検証実験を行うことにより議論した．この議論の結果は，他の GPU クラスタや次期 TSUBAME システムにおいてアプリケーションを設計・実装する際の指標となると期待される．

謝辞 実験にあたって日本電気 (株)，サン・マイクロシステムズ (株)，NVIDIA Corp.，ClearSpeed Inc.，東京工業大学学術国際情報センターの皆様に多大なご協力を頂きました．tsubasa クラスタは東工大グローバル COE「計算世界観の深化と展開」により導入されました．また GOTO BLAS ライブラリの作者であるテキサス大学 後藤和茂氏に感謝致します．本研究の一部は JST CREST「ULP-HPC: 次世代テクノロジのモデル化・最適化による超低消費電力ハイパフォーマンスコンピューティング」および科学研究費補助金 (特定領域研究 課題番号 18049028) の援助による．

参 考 文 献

1) ClearSpeed Technology Inc. `http://www.clearspeed.com/`.
2) NVIDIA CUDA Documentation. `http://www.nvidia.com/object/cuda_develop.html`.
3) TOP500 supercomputer sites. `http://www.top500.org/`.
4) Toshio Endo and Satoshi Matsuoka. Massive supercomputing coping with heterogeneity of modern accelerators. In *Proceedings of IEEE International Parallel and Distributed Processing Symposium (IPDPS08)*, page 10 pages, 2008.
5) Michael Kistler, John Gunnels, Daniel Brokenshire, and Brad Benton. Petascale computing with accelerators. In *Proceedings of ACM SIGPLAN Principles and Practice of Paralle Computing (PPoPP09)*, 2009.

6) A.Petitet, R.C. Whaley, J.Dongarra, and A.Cleary. HPL - a portable implementation of the high-performance Linpack benchmark for distributed-memory computers. `http://www.netlib.org/benchmark/hpl/`.
7) JamesC. Phillips and John E.Stone andand KlausSchulten. Adapting a message-driven parallel application to GPU-accelerated clusters. In *Proceedings of IEEE SC08*, 2008.
8) Jeff Stuart and John Owens. Message passing on data-parallel architectures. In *Proceedings of IEEE International Parallel and Distributed Processing Symposium (IPDPS09)*, 2009.
9) 加藤 季広, 青木 尊之, 額田 彰, 遠藤 敏夫, 松岡 聡, and 長谷川 篤史. 姫野ベンチマークの GPU マルチノード実行における通信と演算のオーバーラップによる高速化〜32GPU で 700GFLOPS 超を達成 〜. In **情報処理学会研究報告** *2009-HPC-120*, page 6 pages, 2009.
10) 小川 慧 and 青木 尊之. CUDA による定常反復 poisson ベンチマークの高速化. In **情報処理学会研究報告** *2008-HPC-115*, pages 19–23, 2008.
11) 遠藤 敏夫, 額田 彰, 松岡 聡, 丸山 直也, and 實本 英之. 四種プロセッサからなるヘテロ型スーパーコンピュータにおける linpack チューニング. In **情報処理学会研究報告** *2009-ARC-182/HPC-119 (HOKKE2009)*, pages 85–90, 2009.
12) 遠藤 敏夫, 松岡 聡, 橋爪 信明, and 長坂 真路. ヘテロ型スーパーコンピュータ TSUBAME の Linpack による性能評価. **情報処理学会論文誌コンピューティングシステム**, 48(SIG 8 (ACS 18)):62–70, 2007.