

TSUBAME 共同利用 平成 26 年度 学術利用 成果報告書

## 利用課題名 高性能計算向け分散メモリ・ストレージ統合システムの研究

英文: The Unified System of Distributed Memory and Storages for High Performance Computing

## 利用課題責任者

緑川 博子

## 所属

成蹊大学 理工学部

<http://www.ci.seikei.ac.jp/midori/>

## 邦文抄録(300 字程度)

高性能の NAND Flash SSD を主メモリ拡張として用い、主メモリサイズを超える大規模問題（ステンシル計算）を高速に処理するため、空間、時間の双方の局所性ブロッキングアルゴリズムと、マルチコアによる高並列・非同期入出力を併用したアルゴリズムを開発した。1 ノード向けのアルゴリズムを複数ノード向けに拡張し、PCIe バス接続 NAND Flash を備えた 2 ノードサーバにおける性能評価結果をもとに、TSUBAME2.5 クラスタにおいて 32 ノードと低速な SATA 型ローカル SSD を利用して、128GiB から 4TiB の問題を処理してノード並列性を評価した。

## 英文抄録(100 words 程度)

This paper evaluates a novel algorithm for out-of-core stencil computation using distributed memories and SSDs over a cluster. Not only the multi-tiling algorithm in temporal and spatial spaces but also a highly parallel asynchronous input/output in Linux kernel library are introduced in the algorithm to increase data access locality and to bridge DRAM and Flash SSD latency divide. The result indicates the effectiveness of our algorithm even when using traditional SATA-based SSDs distributed over multiple nodes in TSUBAME2.5.

*Keywords:* Out-of-core ; stencil computing; temporal blocking; asynchronous I/O; SSD

## 背景と目的

多くの科学技術計算や工学的シミュレーションにおいては、より現実に近い大規模なサイズの問題を解くこと、より正確な結果を得るために、高解像度、細粒度でのシミュレーション計算を行うことが求められており、常に大容量のメモリを必要としている。

一方、スーパーコンピュータの各ノードに搭載できる DRAM の容量は、システム全体の消費電力や実装可能なメモリスロット数などの面で制限されている。また、一つのスーパーコンピュータを共用する様々なユーザの用途に応じた計算・メモリ利用の要求レベルは異なっていることが多く、要求の最大サイズに合わせて、システムの各ノードの搭載メモリ容量をすべて増やすには限界がある。

本利用課題では、各ノードの物理メモリサイズ (DRAM) の合計をこえるような大きなサイズの問題に対して、従来ファイルの置き場として用いられてきた SSD や HDD などのストレージも活用し、統合的なメモリシステムをソフトウェアで構築する。これにより、多様なユーザの要求レベルに対する柔軟性を確保しつつ、高性能かつ大規模なメモリ利用を可能にする。すなわち、現在、高性能計算において性能上大きなボトルネックになっているメモリアクセ

スについて、1 ノードの物理メモリサイズを超えるような大容量なデータに対しても高性能なアクセスを可能にするような記憶システムの構築を目的とする。

各ノードにある DRAM メモリ、SSD などを用い、複数ノードに分散した記憶資源を統合的に扱うためのソフトウェアシステムを設計、構築するための実装実験、性能評価を行う。特に記憶階層について着目し、各ノードの計算処理において、メモリアクセス局所性を生かすようなアルゴリズムを導入し、大規模サイズの問題に対して、多数ノードを用いて高速処理できるようにすることを目指す。応用分野は、当面、ステンシル計算、行列計算などの典型的な科学技術計算などとする。

## 概要

本年度は、1 月に利用申請したため約二ヶ月間、TSUBAME を利用した。これまで申請者組織において 2 ノードを利用してアルゴリズムの開発と検証をおこなった高速 SSD を用いたステンシル計算プログラムを、複数ノードで実行させてデバッグ、動作確認、性能評価を行うことが目的であった。

これまでに、主メモリサイズを超える問題サイズ

であるステンシル計算 (Out-of-Core Stencil Computations) を、テンポラルブロッキング最適化によりデータアクセス局所性を高め、PCIe バス接続型の高速 Flash SSD を主メモリの拡張として用いることにより、3つの手法で評価を行ってきた[1]-[6]. この結果、非同期入出力を用いた手法 (aio 法) は、mmap による手法 (SSD をファイルシステムとし、ファイルをメモリマップして用いる方法) や、スワップによる手法 (SSD をスワップデバイスとして用いる方法) に比べ、高い性能が得られることがわかった(図 1). aio 法では、DRAM だけを用いた場合の性能 (MFlops) の 80%-87%程度 (図 2) の性能が得られることがわかっている. さらに、aio 法を用い、複数ノードによる処理へ拡張したアルゴリズムを構築し、申請者組織におけるサーバ(表 1 上)において PCIe 接続 FlashSSD(ioDrive2)を装着した 2 ノードを用いた実験を行い、正常に稼動し、高い性能が得られることを確認した.

本年度の TSUBAME 利用の主目的は、この複数ノード版アルゴリズムに関し、さらにノード数を増やした場合のプログラムの動作確認、ノード数に対する

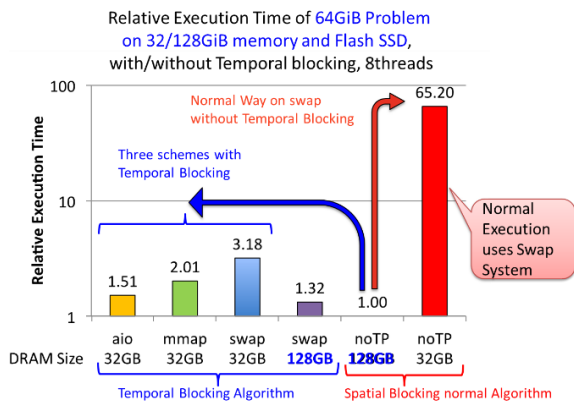


図 1 7点ステンシル計算 64GB 問題における 2 手法の相対性能 (主メモリサイズ 32GiB, 128GiB) [3]

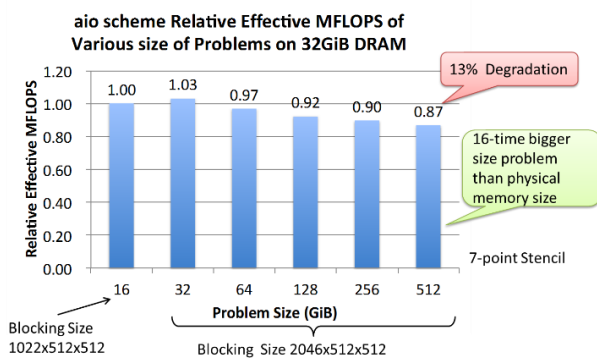


図 2 aio 手法における 7点ステンシル計算の相対性能 (問題サイズ: 16GiB - 512GiB, 主メモリサイズ 32GiB) [3]

スケーラビリティを調査することである. TSUBAME の各ノードに現在装着されている SSD は SATA 接続でアクセス性能が低く、容量の点でも、主メモリサイズに比べせいぜい 2-3 倍程度でしかない(表 1 下). このため、SSD の容量の制限から、1 ノードの主メモリサイズの 10~20 倍を超える問題サイズを処理する場合には、通常以上にノード数が必要である. また各ノードの実行時間は、最新型の SSD を備えたノード(表 1 上)に比べ遅いため、絶対実行時間の評価としては、適していない. しかし、このような条件においても、ノード数を増やすことによる、相対性能の向上についての調査は可能であるため、1 ノードから 32 ノードの TSUBAME ノードを用いた稼動実験をおこなった.

表 1 PCIe-SSD 付 crest サーバと TSUBAME2.5 S96 ノード

Crest3-4 (1-2 Node)	
CPU/node	Intel Xeon CPU E5-2687W (3.1GHz) 2CPU×8core/node = 16core/node
Memory	64GiB/node(2 nodes)
Flash SSD	ioDrive2(PCIe2.0x4), 785GiB, 1.2TiB
Network	Infiniband single FDRx4(56Gbps)
OS	CentOS6.4 (x86_64) 3.13.0
Compiler	gcc 4.4.7 20120313 -O3
MPI Lib	MVAPICH2-2.1a (Thread Support Level=MPI_THREAD_MULTIPLE)

TSUBAME2.5 (1-8 Node)	
CPU/node	Intel Westmere-EP X5670 (2.93GHz) 2CPU×6core/node = 12core/node
Memory	96GiB/node (use 64GiB)
Flash SSD	SATA-SSD ( HP MK0120EAVDT 120GB x2), 160GiB
Network	Infiniband dual QDRx4(80Gbps)
OS	SUSE Linux Enterprise Server 11 SP3 3.0.76-0.11-default
Compiler	gcc 4.3.4 -O3
MPI Lib	MVAPICH2-2.1a (Thread Support Level=MPI_THREAD_MULTIPLE)

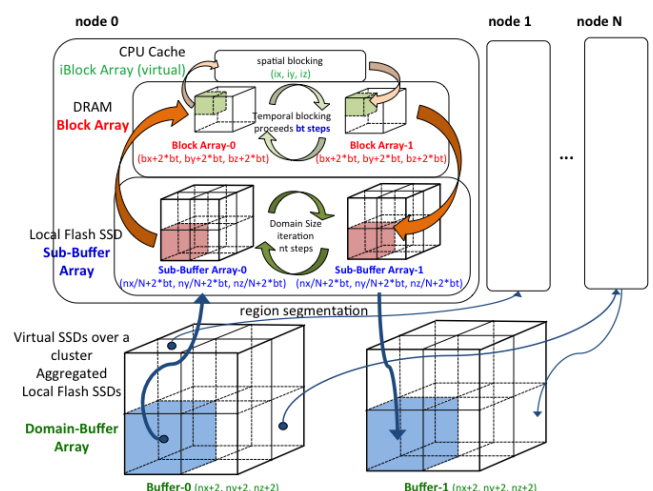


図 3 複数ノードにおけるローカル SSD を用いた Out-of-core ステンシル計算のデータ構造

本実験では、7点ステンシル処理の3次元データ格子を分割し、TSUBAMEの各ノードにあるSSDに配置し(図3)、各ノードにおいて、主メモリを超える計算を行う。テンポラルブロッキングサイズごとにノード間での袖領域の交換(MPI通信)も行う。扱う問題では、計算時間ステップは256、時間ブロックサイズは128としている。

### 結果および考察

TSUBAMEのノードの中で、ノード数が十分にあり、主メモリサイズ(94GiB)に比べ大きい容量を持つSSD(160GiB)を備えているノードとして、S96を用いた。従来の1ノードの実験では、1.2TiBのSSDと64GiBの主メモリを用い、主メモリの16倍のサイズの1TiBの問題を扱うように設定していた。今回のTSUBAMEでは、SSD容量が主メモリの1.5倍程度しか利用できないため、64GiBの主メモリしかもたないノード(実際には94GiBあるが一部しか利用しない)において、SSDに入るサイズである128GiBの部分問題を受け持って処理する構造とした。したがって、1ノードにおける主メモリの拡張としては、主メモリサイズの2倍程度の容量拡張しか行えない。実際には、DRAM上のBlock Array(図3)としては、50GiB程度の3次元配列を用いている。

しかし、TSUBAMEの1ノードから32ノードを利用することにより、128GiBから4TiBまでのサイズの問題を、提案手法により計算した。この時のステンシル計算部分の時間と、実効性能(Mflops)を図4と図5に示す。図4によると、ノード数を増加させて、扱う問題サイズを大きくしても、ほぼ一定時間で処理ができることがわかる。この時間は、DRAM上にあるBlock Array一つ分のステンシル計算時間、これをSSDに非同期入出力する時間、1ノードが担当するデータ(ローカルSSDにあるSub-Buffer Array)がBlock Arrayの何個分か、の3つによって定まる。図4では、扱う問題サイズに比例してノード数を増加させ、1ノードあたりの担当データサイズをほぼ一定にしている。同じサイズの問題を多くのノードで処理する場合には、各ノード間で重複する交換領域の冗長計算が増加する。図6は各問題サイズに対する相対性能を示したものである。図6から計算すると、各ノードにおける並列効率は90%程度にとどまっている。TSUBAMEでは、1ノードあたりの担当計算部分がSSD容量の制限で十分大きくする

ことができないため、比較的頻繁にバリア同期などが発生することになるのも一因と考えられる。

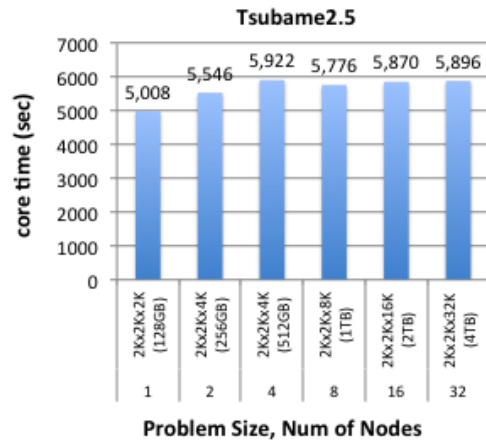


図4 TSUBAME複数ノードにおける7点ステンシル計算時間

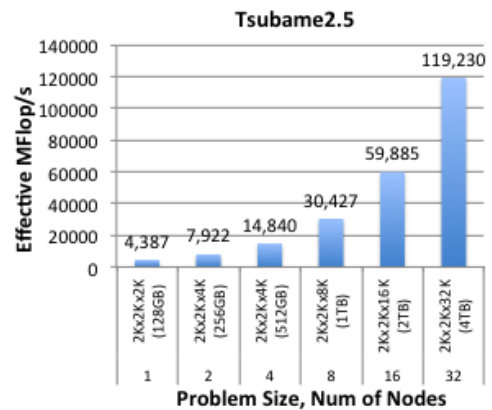


図5 TSUBAME複数ノードにおける7点ステンシル実効性能

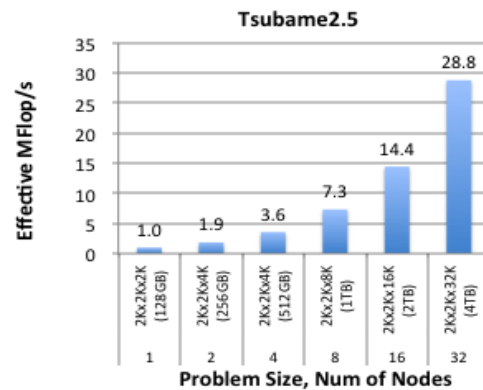


図6 TSUBAME複数ノードにおける相対実効性能

TSUBAMEにあるSATA-SSDのaio性能は、PCIe接続型SSDに比べ、非常に低い。図7は、64GiBの問題サイズにおいて実際にSSDに対してリードライトするデータサイズを用いて、各SSDの性能を調査した図である。これによると、TSUBAMEに用いられているSSD(HP MK0120EAVDT 120GB x2)のバンド幅は表1のcrestサーバ(表1上)で用いたPCIe接続

SSD(ioDrive2)に比べ18%程度の性能しかない。しかし、全体時間に占める非同期入出力時間が3-5%であるため、TSUBAMEの低速なSATA-SSDであっても、全体の性能に及ぼす影響が今回は少なかった。SSDの容量がもっと大きく、主メモリサイズとの比が高い場合、1ノードの担当処理領域が増やすことができるが、その分、SSDへのアクセス回数も増加し、SSD性能がもうすこし全体に影響すると考えられる。

図4-6は、いずれも、問題サイズ128GiB-4TiB、主メモリ64GiB、SATA SSD128GiB利用時の結果である。

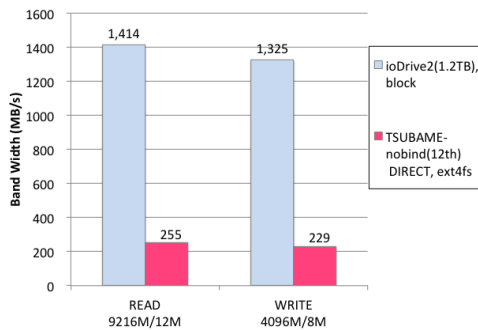


図7 複数ノードにおける相対実効性能

次に、ioDrive2を備えた2ノードcrestサーバにおいて、128GiBから1TiBのサイズの問題を処理した場合の計算時間と実効性能を図8、9に示す。ノード数を増加させずに問題サイズを大きくした場合であっても、1TiB問題の実効性能の劣化は128GiB問題の約3%程度にとどまっている。すなわち、各ノードに割り当てられた部分問題、すなわちローカルSSDにあるSub-Buffer Arrayの計算のために、ローカルSSDからDRAMにあるBlock arrayに読み書き(非同期aio)する時間(回数)が増えても、全体へ及ぼす影響は問題サイズが8倍で3%程度にとどまることになる。

図10に示す1ノードと2ノードの実行時間から換算すると、128GiB、256GiB問題のいずれにおいても、2ノードの並列効率率は約100%となっており、DRAM(64GiB)サイズを超える問題を扱う際に、複数ノードにすることによるオーバーヘッドはほとんどないことがわかる。これは、1つのBlock arrayの計算時間に対し、ノード間通信の時間の割合が非常に低いことに起因する。

2ノードを用いた場合の実効性能(図9)を、TSUBAME(図5)と比較すると、256GiBサイズ問題でそれぞれ7922(Mflops)、35176(Mflops)で、PCIe接続型SSDを利用したcrestサーバにおいては4.4

倍の性能が得られている。

得られた結果から、乱暴ではあるが、各ノードがcrestノードの仕様と同レベルであった場合の試算をしてみる。主メモリサイズ(64GiB)の2倍の部分問題を各ノードが担当する場合、前述のように、2ノードで256GiB問題を処理する性能は、TSUBAMEの4.4倍の性能がcrestサーバでは得られている。したがって、32ノードで4TiB問題を扱う場合のTSUBAMEの性能119230Mflops(各ノードにおける処

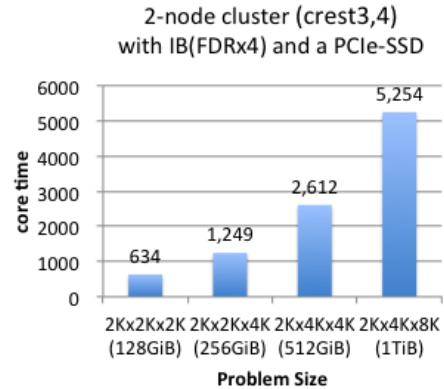


図8 crestシステムにおける2ノードによる計算時間

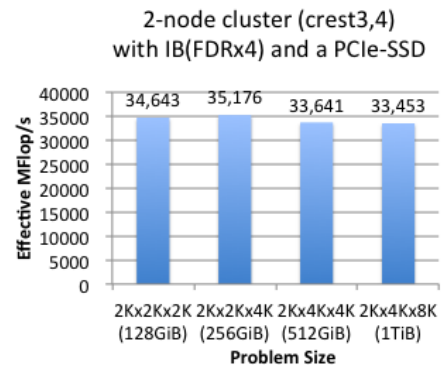


図9 crestシステムにおける2ノードによる実効性能 (問題サイズ128GiB - 1TiB)

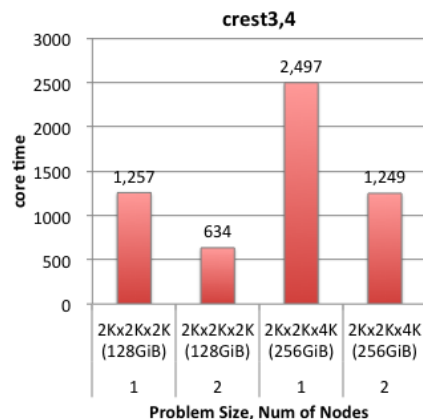


図10 crestシステムにおける1,2ノードによる計算時間 (問題サイズ128GiB, 256GiB)

理は2ノードのときと同じ)を単純に4.4倍すると、524612Mflopsの性能が得られる可能性がある。

次に、ローカルSSD容量が、主メモリの8倍(512GiB)である場合を試算する。図8から、2ノード利用時において1TiB問題は256GiB問題に比べ4.7%の性能劣化がある(各ノードあたりの担当サイズが増え、aioの回数が増加したことなどに依存)が、crestサーバでは33453Mflopsが得られている。一方、図6のTSUBAMEのノード数増加による性能向上比は、2ノードに対して32ノード利用の場合、15.16倍にとどまる。したがって、各ノードにおいて、主メモリの8倍の問題を処理する場合、32ノードを利用すると32TiBの問題を解くことが可能で、そのときの性能は、 $33453 \times 15.16 = 507147\text{Mflops}$ 程度が得られる可能性がある。

今回扱った実験環境では、TSUBAMEの場合、各ノードでの処理時間の内訳の平均は、MPIデータ送受信時間1%以下、SSDのI/O 3-5%程度、ノード間バリア同期4%程度で、90%以上がCPU計算に消費されている。このため、複数ノードにすることによる性能劣化は非常に少なく、ノード数に比例して大きな問題を処理することが可能である。

#### まとめ、今後の課題

当初1ノード向けに開発されたアルゴリズムを拡張し、PCIe接続の高速NAND FLASH SSDを利用して行った2ノードによる事前実験の評価をもとに、今回の実験では、さらに多ノード化した場合の効果について調べた。64GiBの主メモリと128GiBのSSDを用いたノード1~32台により、128GiB~4TiBまでのステンシル問題を評価した。この結果、限られたノード数と、限られた容量、性能のSSDを用いた場合でも、開発したアルゴリズムが、複数ノード並列の効果十分あることが明らかになった。

現在のTSUBAMEノードでは、Sノードの場合、ローカルSSDは主メモリ以下の容量しか持っていないため、本研究の主メモリ拡張のための実験には適さない。S96ノードは140GiB程度までSSD利用が可能であるが全体で41ノードしかないので、2累乗のノード数とすると32ノードが最大で、4TiB程度の問題が最大となる。ローカルSSDと主メモリの比ではL128Fノードが最大で、400GiBのローカルSSDを持つが、L128Fは10ノードしかないため、最大限利用したとしてもやはり4TiB以下の問題しか扱うこと

ができない。このため、out-of-coreの実験としては、TSUBAMEにおいては4TiB程度の問題サイズが限界ということになる。したがって、実験データをもとに非常に大まかな性能試算もおこなった。

今後の研究としては、以下のものを考えている。今回のTSUBAMEにおける実験では、問題の分割方式として、単次元方向に全体のデータドメインを分割して各ノードに割り当てる方式を採用したが、さらに柔軟性を高めるために、メッシュ状の分割方式をとり入れるアルゴリズムを開発する。実際のデータ配列のメモリレイアウトは、aio方式の並列入出力による制限として、SSDのブロックサイズの倍数に、入出力先頭番地、サイズなどを合わせる必要がある。これらを考慮したレイアウトと、境界領域のノード間通信の機構を導入し、複数ノードを用いて、さらに大きな問題サイズを扱えるアルゴリズムとする。

また、今回は、3次元ドメインデータをダブルバッファとして1組もつ単純なステンシル計算を対象とし、処理や解析を単純化したが、今後はさらに実用的な応用に対し、簡単に非同期aio方式のアルゴリズムを用いることができるようなユーザインターフェース、APIなどを開発していきたいと考えている。

また、TSUBAMEの特徴であるGPUを利用して計算部分を短縮化することも考えている。

#### 参考文献

- [1] Hiroko Midorikawa, Hideyuki Tan and Toshio Endo: "An Evaluation of the Potential of Flash SSD as Large and Slow Memory for Stencil Computations", Proc of the 2014 International Conference on High Performance Computing and Simulation (IEEE HPCS2014), pp.268-277, July 2014
- [2] 緑川博子, 丹英之: "大規模ステンシル計算のためのFlash SSD向けテンポラルブロッキングの性能評価", 情報処理学会, ハイパフォーマンス研究会 Vol.2014-HPC-145, No.22, pp.1-9, (2014, 7/29)
- [3] Hiroko Midorikawa, "Using a Flash as Large and Slow Memory for Stencil Computations". Flash Memory Summit 2014, August 2014
- [4] 丹英之, 緑川博子: "ブロックデバイス非同期I/Oによるフラッシュストレージを用いたステンシル計算の性能評価", 電子情報通信学会, コンピュータシステム研究会 CPSY2014-52, pp.31-36, (2014, 10/10)
- [5] Hiroko Midorikawa, "Using a Flash SSDs as Main Memory Extension with a Locality-aware Algorithm". 2015 Non-Volatile Memories Workshop, in UCSD, March, 2015
- [6] Hiroko Midorikawa, Hideyuki Tan, "Locality-Aware Stencil Computations using Flash SSDs as Main Memory Extension", Proc. of CCGric2015, May, 2015