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エアで構築する．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エアシステムを設計，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 TUSUBAME ノードを用いた稼動実験をおこなった．

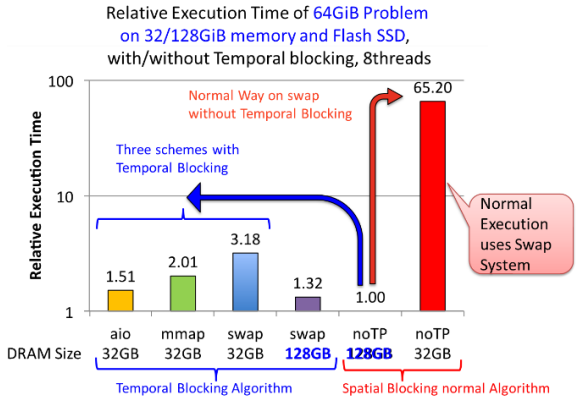

図１ ７点ステンシル計算 64GB 問題における２手法の相対性能
（主メモリサイズ 32GiB，128GiB）[3]

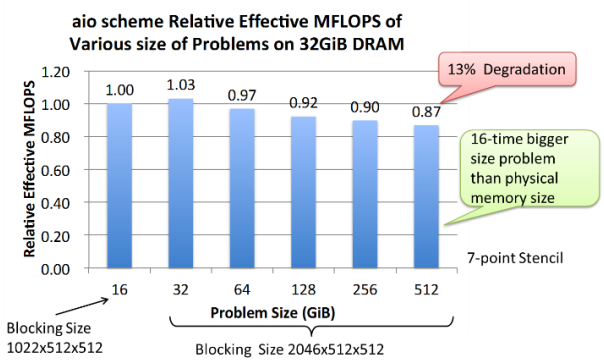

図２ aio 手法における７点ステンシル計算の相対性能
（問題サイズ：16GiB − 512GiB，主メモリサイズ 32GiB）[3]

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)

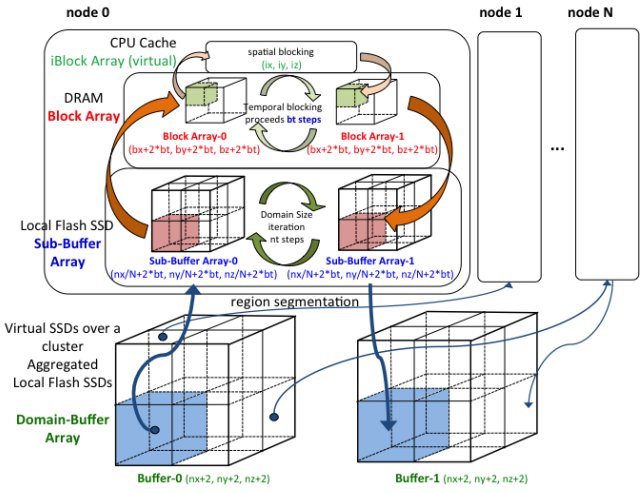

図３ 複数ノードにおけるローカル SSD を用いた Out-of-core ステンシル計算のデータ構造

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．

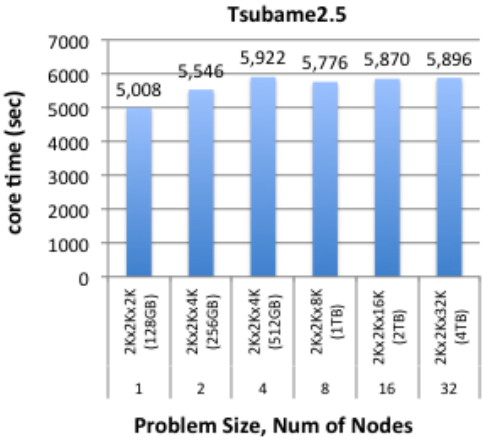

図 4 TSUBAME 複数ノードにおける 7 点ステンシル計算時間

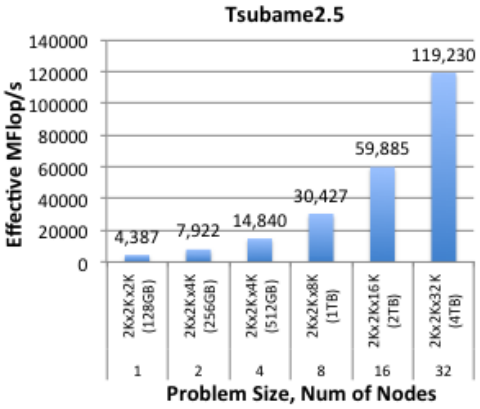

図５ TSUBAME 複数ノードにおける 7 点ステンシル実効性能

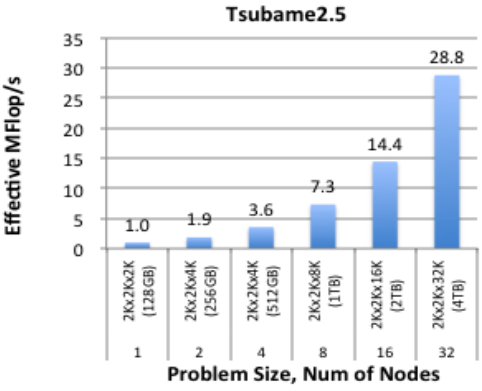

図６　TSUBAME 複数ノードにおける相対実効性能

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 SSD 128GiB 利用時の結果である．

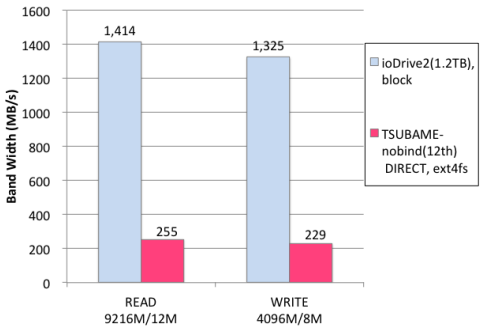

図７　複数ノードにおける相対実効性能

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-Buffur 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

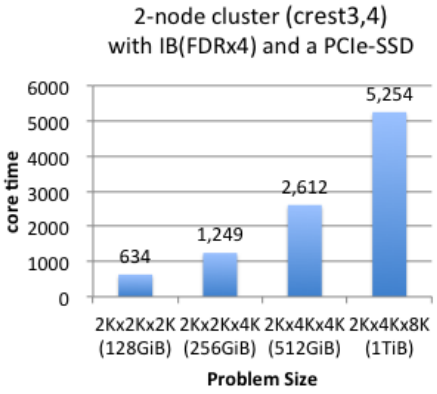

図８　crest システムにおける２ノードによる計算時間

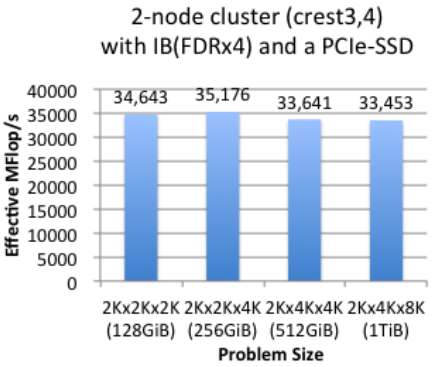

図９　crest システムにおける２ノードによる実効性能
（問題サイズ 128GiB - 1TiB）

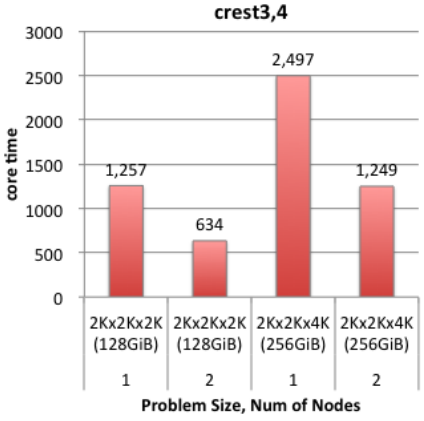

図 10　crest システムにおける 1,2 ノードによる計算時間
（問題サイズ 128GiB，256GiB）

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 x 15.16 = 507147Mflops 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が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，単一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い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-Volitile 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