11-NA01

高精度行列 - 行列積アルゴリズムにおける並列化手法の開発

片桐孝洋（東京大学）

概要　行列-行列積に代表される基本線形計算を集約したライブラリ BLAS (Basic Linear Algebra Subprograms) は、多くの線形計算で必須の処理である。しかし、従来の数値計算ライブラリは、演算速度は考慮しているが演算精度の考慮が不十分である。解の精度保証が重要な課題となっている。本研究では、大石グループで開発された高精度行列-行列演算に 2 種のスレッド並列化を行った。予備評価の結果、並列処理の規模に応じ並列化方式を切り替える必要があることが判明した。またその切り替えを実現できる自動チューニング（AT）を、AT 言語の ABCLibScript を用いて実現した。T2K オープンスパコン（1 ノード、16 スレッド）を用いた性能評価の結果、AT による並列化方式の切り替えで最大で 5 倍程度の速度向上を確認できた。

1. 研究の目的と意義

行列-行列積に代表される基本線形計算を集約したライブラリ BLAS (Basic Linear Algebra Subprograms)は、多くの線形計算で必須の処理である。一般に BLAS を含む従来の数値計算ライブラリでは演算速度は考慮しているが、計算結果の正確性の考慮が不十分なことが多い。解の精度保証が重要な課題となっている。一方で、BLAS を用いた汎用的な数値計算ライブラリ（例：LAPACK）において、精度保証をする研究や実装は多くない。

BLAS を精度保証する研究が、早稲田大学の大石教授のグループにより進められている。本研究は大石グループで開発された高精度行列-行列演算（尾崎の方法[1]）を基本とし、その並列化を中心する以下の研究開発を行うことを目的とする。

i. 「疎行列 - 密行列積」、もしくは、「疎行列 - 疎行列」の実装方式の開発
ii. スレッド並列化手法の開発
iii. 分散メモリ型計算機による並列化手法の開発

2. 当拠点公募型共同研究として実施した意義

(1) 共同研究を実施した大学名と研究体制

- 東京大学(代表者：片桐孝洋)
- 芝浦工業大学（副代表者：尾崎克久）
- 東京女子大学（荻田武史）
- 早稲田大学（大石進一）

(2) 共同研究分野

- 精度保証計算、高精度計算
- 高性能計算（HPC）
- 並列処理
- マルチコア・メニーコア最適化
- ソフトウェア自動チューニング

(3) 当公募型共同研究ならではという事項など

本研究は、精度保証計算を専門とする数値解析分野の数理科学者と、並列化および高性能計算を専門とする高性能計算分野の計算機科学者との協業で成り立つ。従来は、このような境界的な研究テーマに関しては、数値解析分野の研究者が並列化を行うことが多かった。しかし近年の計算機アーキテクチャの複雑化により、数値解析分野の研究者による高効率な並列化がますます困難になっている。

一方で、超並列実行により、解くべき問題サイズ（行列の次元）は大規模化している。連立一次方程式では、数百万次元の密行列がベンチマークレベルでは解かれているが、このような大規模問題における実問題の解の品質について検証が十分になされていない。近い将来、現状の倍精度演算では要求精度を満たさなくなる可能性は否定できない。これらの要請に答えられるのは、計算機科学分野の研究者では不可能である。数値解析分野の研究者の知識を必要とする。

以上から、高速実行・超並列化技術の確立と、

解の精度保証（高精度化）の２点が、大規模数値計算には必要であると結論付けられる。本研究は、適用対象が行列 - 行列積に限定はされてはいるが、一部の要請に答えることが可能である。この点が、当公募型共同研究ならではの特徴である。

3. 研究成果の詳細と当初計画の達成状況

(1) 研究成果の詳細について

概要

本研究で採用する高精度な行列-行列積アルゴリズムは、尾崎らが開発したアルゴリズム[1](以降、**尾崎の方法**）を用いる。尾崎の方法は、入力行列に対して以下の無誤差変換を行う：

I)行列 A と行列 B を下記のように分解する（インデックスが若いほうが高いビットを持つようにする）

$$A = A^{(1)} + A^{(2)} + A^{(3)} + \ldots + A^{(p)}$$
$$B = B^{(1)} + B^{(2)} + B^{(3)} + \ldots + B^{(q)}$$

II)行列積 AB を以下のように計算する（ $p\,q$ 個の行列積となる）

$$AB = (A^{(1)} + A^{(2)} + \ldots + A^{(p)})(B^{(1)} + B^{(2)} + \ldots + B^{(q)})$$
$$= (A^{(1)} + A^{(2)} + \ldots + A^{(p)})(B^{(1)} + B^{(2)} + \ldots + B^{(q)})$$
$$= A^{(1)}B^{(1)} + A^{(1)}B^{(2)} + A^{(2)}B^{(1)} + \ldots + A^{(p)}B^{(q)} \quad \ldots(1)$$

処理 I）の分解の仕方を工夫することで、処理 II)における行列積を無誤差の演算にすることができる。行列サイズが大きくなるに従い、ほとんどの演算時間は行列積処理部分となる。

また、分解数である p、q を限定すれば、部分的な多倍長化演算と同様の効果を得ることができる。したがって、高精度化を図ることが可能である。

処理 I)の分解の過程で、入力行列の要素値のレンジに依存し疎行列が生成される。このため密行列を入力としても、分解の過程で疎度が大きくなる場合は、密行列から疎行列化したほうがよい。すなわち、「疎行列 - 密行列積」、および、「疎行列 - 疎行列積」の演算に切り替えるほうが、全体の計算量（実行時間）の縮減が期待できる。ただし一般の数値計算ライブラリは、「疎行列 - 密行列積」、および、「疎行列 - 疎行列積」の専用実装は提供されていないので、新規開発が必要となる。さらにこの並列化は単純でないことが予想され、コードの独自開発が必要となる。

以下の前提を置く。

- LU 分解や QR 分解の精度保証への適用については、荻田が開発したアルゴリズムを利用する。
- 既存の数値計算ライブラリへの適用は LAPACK などのフリーウェアを利用する。
- 「疎行列 - 密行列積ルーチン」ではないが、類似の機能である「疎行列 - ベクトル積」の並列化に関し、代表者の片桐が行った高スレッド並列化の研究がある。この研究では、自動チューニング機能付き数値計算ライブラリ Xabclib へ開発ルーチンを実装した。開発コードの実用ライブラリへの展開技術も有している。

以上の経験を基に、高精度行列-行列積アルゴリズムの開発を行う。数値計算ライブラリ全体でみると、単に実行速度のチューニングだけではなく、演算精度を考慮したチューニングも可能となる点にも新規性確保が期待される。

尾崎の高精度行列 - 行列積アルゴリズム

尾崎の方法では、図１が主演算となす。

```
Function F = EFT_Mul(A, B)

  [A, n_A] := Split_A; [B, n_B] := Split_B; k := 1;

  for i =1: n_A

    for j =1: n_B

      F{ k } := A{ i } * B{ j };    k := k + 1;
  end;  end;  end
```

図１ $AB = \sum_{k=1}^{n_A \cdot n_B} F^{(k)}$ の変形部分

ここで図１の F の和には、faithful と呼ばれる

結果を得るアルゴリズムを用いている。そのため演算結果は、ほぼ最良になることが知られている。

図２に行列Aの分解部分の詳細を載せる。

```
Function [D, n ] = Split_A(A)
          A

   n  := 0; n := size(A, 2)
    A

   while (norm (A, 1 )~=0)

      n  := n  + 1;
       A     A

      μ  := max(abs(A), [ ], 2);

                                  -1
      τ  := 2.^ceil ((log2(u  ) + log2(n +1))/2);

       t  := 2.^ceil((log2(μ)* τ;

       δ^(nA)
        A      := repmat(t, 1, q);

      A[n ] := fl((A +  δ^(nA) ) -  δ^(nA) );
       A                A          A

      A := fl(A  -  A[n ]);
                    A

   end;  end
```

図２　行列Aの分解部分の詳細

図１、図２で用いられている表記法について、以下にまとめる。

- 表記の説明

  fl（・）：　最近点への丸めで計算

  F： 浮動小数点数の集合

  A： サイズが m 行 n 列の行列

  B： サイズが n 行 p 列の行列

  u ： the unit round off

  2^{-24}: IEEE 754 binary32 (single precision)

  2^{-53} : IEEE 754 binary64 (double precision)

性能評価環境

東京大学情報基盤センターに設置された T2K オープンスパコン（東大版）(1 ノード、16 コア）を利用してスレッド並列化による並列化を行った。

日立製作所による C コンパイラ（日立最適化 C）を使用し、OpenMP 並列化を行った。最適化オプションは“-Os -omp”である。また GOTO BLAS ver.1.26（スレッド並列版,および遂次版の双方）を BLAS 演算部分には利用している。

試験行列

試験行列は以下である.

- A, Bの要素を pow(10,rand()%Φ)　で生成

Φの値が大きいと、行列要素の値の分散が大きくなる。この場合、尾崎の方法による行列分割の回数が増え、行列 - 行列積の演算回数も増える。総合的な演算量（演算時間）が増加するので、並列処理の効果に影響する。

自動チューニング言語機能への展開

尾崎の方法の並列実装に AT を導入するには、既存の AT 用計算機言語で、尾崎の方法を実現したプログラム（ライブラリ化した場合のカーネルコードなど）に AT 方式を記述することが最も簡単な実現方法である。代表者の片桐は、AT 言語の 1 つである ABCLibScript を開発している。ABCLibScript による尾崎の方法の AT 化がもっとも容易な方法である。

ABCLibScript では、チューニング変数の定義、チューニング変数の値の自動調整、任意のループの展開などコード自動生成、アルゴリズム選択に使えるコード選択など、AT で必須となる基本機能について提供する計算機言語である。

ABCLibScript による AT の記述形式は＜ディレクティブ＞形式である。ユーザプログラムに AT 機能のコメント行を挿入するだけで、任意のプログラムに AT 機能を実現できる。元の逐次・並列プログラムの実行をまったく阻害せず、必要に応じ AT 機能が付加できることが最大の特徴である。

この特徴は、レガシーコードにも AT 機能が容易に実装できることを意味している。したがって、数値計算ライブラリ分野のように、膨大で、かつ、高性能なレガシーコードを有する分野でのソフトウェア開発者のプログラムにおいて、AT 機能を付加する場合の開発コスト削減に寄与できる。

ABCLibScript は、計算機言語ごとにプリプロセ

ッサを提供している。C 言語版の ABCLibScript プリプロセッサでは、

#pragma ABCLib … region start

～

#pragma ABCLib … region end

で囲まれた任意のプログラムの領域に AT を施すことができる。この指定後、専用プリプロセッサにより、AT 用のコード（C 言語でユーザが読めるもの）が自動生成される

AT 言語適用例

図 3 に、**インストール時 AT**（ソフトウェアのインストール時に行う AT）機能を、尾崎の方法のメインカーネルにおいて、複数の行列-行列積部分の＜実コード＞に適用した例を示す。

```
#pragma ABCLib install select region start
#pragma ABCLib name SelectOzaki
#pragma ABCLib select sub region start
  for (i=0;i<Ak;i++) {
    for (j=0;j<Bk;j++) {
      dgemm_(chn, chn, &m, &p, &n, &one,
        A+i*m*n, &lda, B+j*n*p,
        &ldb, &zero,
        C+m*p*(j+Bk*i), &ldc);
    } }
#pragma ABCLib select sub region end
#pragma ABCLib select sub region start
#pragma omp parallel for private(i,j)
  for (k=0;k<Ak*Bk;k++) {
    i = k / Bk; j = k % Bk;
    dgemm_seq(chn, chn, &m, &p, &n, &one,
       A+i*m*n, &lda, B+j*n*p, &ldb, &zero,
       C+m*p*(k), &ldc); }
#pragma ABCLib select sub region end
#pragma ABCLib install select region end
```

図 3　尾崎の方法のメインカーネルに対する ABCLibScript 記述の追加

本稿のスレッド並列化の例は、アルゴリズム選択機能が必要である。これは、ABCLibScript の＜select 指定子＞で指定可能である。Select 指定子で選択されるプログラム対象は、

#pragma ABCLib select sub region start

～

#pragma ABCLib select sub region start

の間に記載する。図３に、そのコードを示す。

図３では、Fortran で記述されている BLAS の静的ライブラリ（GOTO BLAS）の関数コールに関し、（１）スレッド並列化された関数 dgemm_(…) をコールする実装と、（２）逐次 BLAS 関数 dgemm_seq(…)を問題サイズの並列性を利用してコールしている実装、の２種を選択できる。

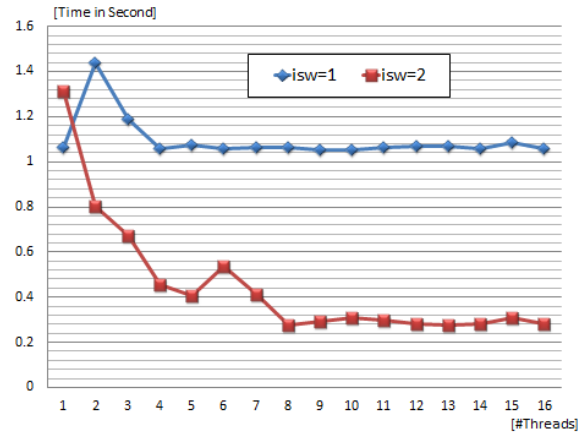
問題レベル並列性を利用するスレッド並列化実装では、OpenMP を利用しているので、OpenMP のディレクティブが混載する。ABCLibScript では、OpenMP など、他言語のディレクティブと共存が可能である。この点も、ABCLibScript のメリットである。

ここでは説明を割愛するが、AT 時のスレッド数や問題サイズの指定が、図３では記載されていない。これらは、ABCLibScript のシステム変数を用いて、その対象と範囲を指定できる。たとえば、問題サイズを 100 から 1000 まで 100 刻みで調べる、という指定が、容易に設定可能である。

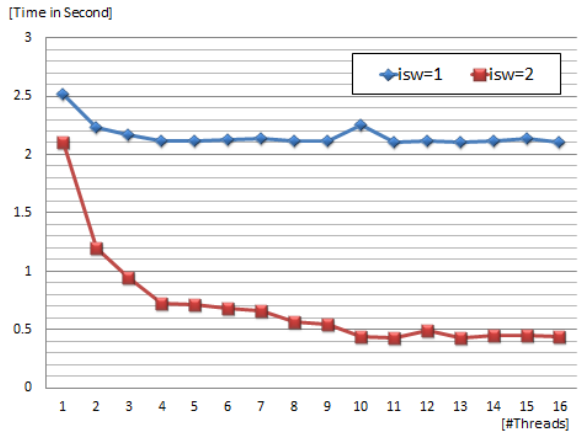
そこで本実装では、問題サイズを固定した上で、スレッド数を 1 から 16 まで 1 刻みで変化させるチューニングの事例を紹介する。

AT 言語による AT の効果

図４に、ABCLibScript による自動チューニングの結果（実行時間[秒]）を載せる。



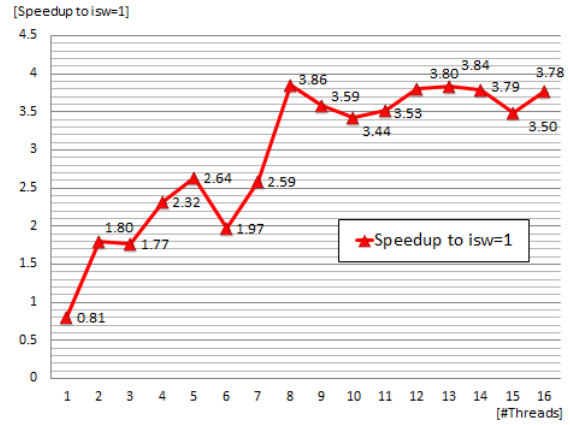
(a) n = 1000、 Φ = 1 の結果



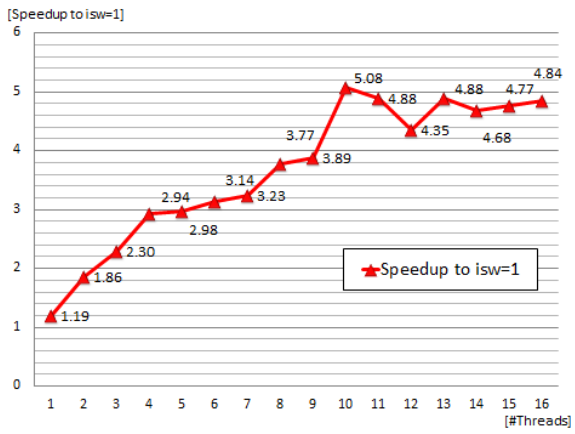
(b) n = 1000、 Φ = 10 の結果

図４　自動チューニングの効果（実行時間[秒]）。isw=1 は dgemm スレッド並列化。isw=2 は逐次 dgemm で問題レベル並列化。

図５に、ABCLibScript による自動チューニングの効果（dgemm でスレッド並列化(isw=1)に対する速度向上）を載せる。



(a) n = 1000、 Φ = 1 の結果



(b) n = 1000、 Φ = 10 の結果

図５　自動チューニングの効果（dgemm でスレッド並列化(isw=1)に対する速度向上）。isw=1 は dgemm スレッド並列化。isw=2 は逐次 dgemm で問題レベル並列化。

考察

図４、５から、n = 1000 では、

- 逐次 dgemm スレッド並列化（isw=1）が有効なのは、Φ = 1 でスレッド数 1 の時のみ。
- それ以外は、逐次 dgemm で問題レベル並列化(isw=2)が高速。
- 通常実現されると思われる逐次 dgemm スレッド並列化（isw=1）に対する AT の効果（速度向上）は、最大で 5 倍程度である。スレッド数が増えるほど一般に isw=2 が有効なので、AT の効果も大きい。

なお、問題の性質は実行時にならないと判明しないので、この例題の AT をインストール時に行うことはできない。したがって、問題を固定した後で AT を行う、実行起動前時 AT に本機能を入れ込む方がよい。

実行起動前時 AT も、ABCLibScript で提供している。実行起動前時 AT を行うためには、ディレクティブ中の“install”を“static”に変更するだけでよい。

精度評価

ここでは、尾崎の方法の演算精度評価を行う。

評価に使用した CPU は、Xeon X5550、2.67 GHz(2

CPU、合計 8 core)である。MATLAB 2011a を用いた。MATLAB 2011a の通常の倍精度演算の結果(A*B)と、尾崎の方法による演算結果の精度を比べる。

尾崎の方法は C コードを MATLAB External Interfaces を利用して記述し、MATLAB 2011a に付属している libmwblas.lib を利用し dgemm を呼び出す。演算結果の行列の各要素について、真値との相対誤差を調べ、その最大を表にまとめる。

精度評価の結果

表 1(a)に、前節と同様の方法で行列要素を生成し、かつ行列要素の符号もランダムにつけた行列に対する、相対誤差の最大値を載せる。一方、表 1(b)は、表 1(a)の行列に対し逆行列を生成し、その生成した逆行列と、元の乱数行列の行列 - 行列積を行った結果をのせる。

表1と表2から、尾崎の方法を利用することで、通常の倍精度演算の dgemm では実現できない高精度演算ができる。

表１　尾崎の方法と通常の行列 - 行列積の演算精度（相対精度の最大値）

(a)　乱数行列の積

乱数 / phi	1		5	
次元	近似	提案	近似	提案
10	8.83E-15	1.00E-16	6.79E-15	1.01E-16
100	5.21E-12	1.10E-16	1.08E-12	1.10E-16
1000	3.05E-09	1.11E-16	4.77E-09	1.11E-16
8000	2.03E-08	1.11E-16	1.96E-08	1.11E-16

乱数 / phi	10	
次元	近似	提案
10	1.97E-14	1.02E-16
100	1.30E-11	1.11E-16
1000	4.66E-09	1.11E-16
8000	2.37E-07	1.11E-16

(b) 乱数行列と乱数行列から生成される逆行列との積

inv / phi	1	
次元	近似	提案
10	3.73E+02	1.10E-16
100	1.54E+03	1.11E-16
1000	6.83E+04	1.11E-16
8000	9.34E+05	1.11E-16

(2) 当初計画の達成状況について

本提案では研究進捗を以下の３フェーズに分け、研究の意義を達成することを目的にしていた。

- フェーズ１（2011 年 4 月～7 月）: 既存の高精度行列-行列アルゴリズムの処理を分析し、計算機で性能の予備評価を行う。特に、疎行列化する場合の比ゼロ分布の特徴と有効な実装法を調査することで、本問題特有となる効率的な実装方式を開発する。
- フェーズ２（2011 年 7 月～12 月）: フェーズ１の結果を基に、OpenMP を用いてスレッド並列化する。実際の計算機を用いて並列性能を評価する。その結果に基づき、スレッド並列実装法を研究する。一方で、分散メモリ並列化のためのプロトタイピングを行う。分散並列化性能の予備評価を行う。
- フェーズ３（2012 年 1 月～3 月）: フェーズ２での結果を基に、性能パラメタを抽出する。得られた性能パラメタをもちいてチューニング効果の検証をする。自動性能チューニング方式を研究開発し、その方式の予備評価を行う。具体的には分解される行列において、どこまで疎度が上がったら密行列積から疎行列積に切り替えるのかのタイミングに関する自動チューニング方式があげられる。

以上の３フェーズの進捗状況により、以下の課題についても柔軟に研究を進めることにしていた。

- 行列分解、たとえば、LU 分解や QR 分解に本アルゴリズムを適用した場合の性能について評価する。
- 既存の数値計算ライブラリ、たとえば LAPACK

での性能と演算精度について評価する。

上記のうち、演算の疎行列化、および、分散メモリ並列化の研究進捗が本年度に実現できなかった。これらは来年度の課題としたい。

参考文献

[1] K. Ozaki, T. Ogita, S. Oishi, S. M. Rump: Error-Free Transformation of Matrix Multiplication by Using Fast Routines of Matrix Multiplication and its Applications, Numerical Algorithms, Springer, 2011.
(DOI: 10.1007/s11075-011-9478-1)

4. 今後の展望

本研究では、高精度行列-行列積アルゴリズムにおいて、（1）OpenMP によるスレッド並列化、（2）AT 言語を用いた AT 方式の実装、および（3）並列化に関する基礎性能データ、の取得には成功した。また、AT 言語については、本研究成果の機能を含む、オープンソースコードのリリースを行った。以上から、基礎的な並列化と AT 化は達成できたと判断する。

今後の展望として、より広範なアプリケーションに本方式を適用するため、および、より高速化するために、疎行列演算の対応を行うことがあげられる。また、MPI を利用した分散並列化を行うことで、既存の分散並列版の多倍長演算ライブラリとの性能比較を行うことが可能となる。このことで、ポストペタ環境での超並列実行における尾崎の方法の並列性の検証を行っていきたい。

5. 研究成果リスト

(1) 学術論文：

- K. Ozaki, T. Ogita, S. Oishi, S. M. Rump: Error-Free Transformation of Matrix Multiplication by Using Fast Routines of Matrix Multiplication and its Applications, Numerical Algorithms, Vol.59:1, pp. 95-118, 2012.

(2) 国際会議プロシーディングス：なし

(3) 国際会議発表：なし

(4) 国内会議発表：

- 尾崎克久、荻田武史：浮動小数点数として最高の結果を返す行列積の計算法、京都大学 数理解析研究所 研究集会「科学技術計算における理論と応用の新展開」、2011 年 10 月
- 片桐孝洋、尾崎克久、荻田武史、大石進一：高精度行列 - 行列積アルゴリズムのスレッド並列化と ABCLibScript への機能実装、第 133 回 HPC 研究会、情報処理学会研究報告 HPC-133、2012 年 3 月

(5) その他（特許、プレス発表、著書等）

- ABCLibScript の C 言語版プリプロセッサに尾崎の方法の AT 機能を実現、2011 年 10 月 23 日リリース版
  http://www.abc-lib.org/format.htm