2H-02

高精度行列-行列積における疎行列演算実装選択の自動チューニングの検討

青木 将太† 片桐 孝洋‡ 大島 聡史‡ 永井 亨‡

名古屋大学 情報学部 コンピュータ科学科†

名古屋大学 情報基盤センター‡

1. はじめに

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である BLAS (Basic Linear Algebra Subprograms)は，多くの数値計算プログラムで利用されている．しかし，従来の数値計算ライブラリは，演算速度の向上は考慮しているが演算精度の向上に関する考慮が不十分であることが多い．

一方，尾崎が提案した高精度行列-行列積演算アルゴリズム（以降，尾崎の方法）は，利用している浮動小数点演算型の精度限界まで高精度演算を行うことができる．

市村らの研究[1]では，尾崎の方法を用いた場合に計算途中で発生する行列の特性に注目し，行列-行列積演算を疎行列演算で行う実装の評価が行われている．本稿では，尾崎の方法の途中で発生する疎行列演算による複数の実装について，実行前に最適実装を選択する自動チューニング（Auto-tuning，以降 AT）方式開発に関する検討を行う．

2. 尾崎の方法と予測方法

2.1 尾崎の方法の概要

尾崎の方法ではまず、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)

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

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）

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

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．

本研究では,処理 II)における，複数回の行列-行列積演算に対して，文献[1][2]で用いた以下の実装を用いて，AT 方式の開発に資する方法を検討する：

C1. degmm を用いる実装
C2. CRS 形式における SpMV(内部並列)
C3. CRS 形式における SpMV(外部並列)
C4. CRS 形式における SpMV(複数右辺による内部並列化)
C5. CRS 形式における SpMV(複数右辺による内部並列化(ブロック幅を 100 に固定))
C6. ELL 形式における SpMV(内部並列)
C7. ELL 形式における SpMV(外部並列)
C8. ELL 形式における SpMV(複数右辺による内部並列化)
C9. ELL 形式における SpMV(複数右辺による内部並列化(ブロック幅を 100 に固定))
G1. dgemm を用いる実装(GPU)
G2. CRS 形式における SpMV(GPU)
G3. ELL 形式における SpMV(GPU)
G4. CRS 形式における SpMM(GPU)

疎行列演算は疎度 90 以上の場合のみ適応し，それ以外は G1 を除いて CPU の dgemm で計算を行う．

2.2 実行時間の予測方法の提案

実際に測定して得られた実行時間を使用して予測を行う．ここで予測に利用するのは，行列-行列積演算を 50 回繰り返した中の最速の実行時間とする．

予測する際に，処理 I)によって分解された各行列の疎度は分かっているものとする．また，予測の評価に使用する行列（試験行列）と，予測のために使用する行列の，行列サイズと乱数の種は同一とする．

II）の疎行列演算は，エラーフリー変換後に定まるが，これは分解行列 $A^{(i)}$の疎行列への変換と，$A^{(i)}$に対する q 回の行列-行列積演算を p 回繰り返す．

本稿では，疎度 90 と 98 の行列を尾崎の方法を用いて行列-行列積演算を 50 回行う．エラーフリー分解すると $A^{(1)}$は，本稿の行列では，疎度 90 と 98 に近い疎度に分解される．この特性を利用し，$A^{(1)}$を疎行列形式に変換する時間の最速なものと，疎度に対する変換時間に対して，一次関数を用いて実行時間を近似する．

ここで，各疎度に対する変換時間を予測する．$A^{(i)}$の変換時間を $t^{(i)}$とする．

疎度 90 と 98 の行列を分解した $A^{(1)}B^{(1)},\cdots,A^{(1)}B^{(q)}$ を 50 回ずつ繰り返して $A^{(1)}B^{(j)}(1\leqq j\leqq q)$の 1 回当たりの演算時間の最速の時間を求める．また，疎度に対する演算時間の一次関数を求め，各疎度に

A Discussion of Auto-tuning for Implementation Selection of Sparse Matrix Computations in High Accurate Matrix-Matrix Multiplication

† Shota Aoki, Computer Science, School of Informatics, Nagoya University

‡ Takahiro Katagiri, Satoshi Ohshima, Toru Nagai, Information Technology Center, Nagoya University

	実装												
疎度	C1	C2	C3	C4	C5	C6	C7	C8	C9	G1	G2	G3	G4
90	0.799	2.641	1.532	1.878	3.422	3.151	1.565	1.929	3.751	1.034	1.904	4.012	0.794
92	0.801	2.516	1.231	1.408	2.700	2.388	1.227	1.460	2.944	1.034	1.820	3.652	0.760
94	0.797	2.024	0.911	0.976	1.970	2.158	0.923	1.033	2.168	1.032	1.546	3.052	0.737
96	0.798	1.578	0.598	0.606	1.249	1.646	0.630	0.639	1.368	1.036	1.460	2.526	0.704
98	0.799	1.086	0.277	0.266	0.375	1.155	0.304	0.286	0.449	1.036	1.362	2.069	0.681

図 1 尾崎の方法における行列-行列積演算部の実行時間

	実装												
疎度	C1	C2	C3	C4	C5	C6	C7	C8	C9	G1	G2	G3	G4
90	0.807	2.884	1.489	1.888	3.388	3.112	1.504	1.921	3.691	1.044	1.828	4.043	0.783
92	0.806	2.437	1.183	1.477	2.653	2.628	1.200	1.508	2.887	1.044	1.699	3.545	0.751
94	0.805	1.991	0.878	1.066	1.919	2.145	0.896	1.096	2.085	1.043	1.571	3.047	0.718
96	0.805	1.551	0.578	0.661	1.197	1.670	0.597	0.691	1.295	1.043	1.445	2.557	0.686
98	0.804	1.105	0.272	0.250	0.463	1.187	0.292	0.278	0.492	1.042	1.317	2.059	0.654

図 2 尾崎の方法における行列-行列積演算部の予測 2 による予測時間

対する演算時間を予測する．いま，$A^{(i)}$の演算時間を$t^{(i)}$とすると，全体の時間は式(1)で予測できる．

予測 1 $(t^{(1)}+\cdots+t^{(p)}) + (t^{(1)}+\cdots+t^{(p)}) \times q.$ …(1)

また，疎度 90 と 98 の行列を分解した $A^{(1)}B^{(q)}$を 50 回繰り返して $A^{(1)}B^{(q)}$の演算時間の平均から，疎度に対する平均の演算時間の一次関数を求め，演算時間を予測する方法も考えられる．いま，$A^{(i)}$の演算時間を $a^{(i)}$とすると，式(2)で予測できる．

予測 2 $(t^{(1)}+\cdots+t^{(p)}) + (a^{(1)}+\cdots+a^{(p)}) \times q.$ …(2)

なお，$A^{(i)}$が dgemm で計算する場合は $t^{(i)}=0$ とする．

予測 1 と予測 2 を用いて，尾崎の方法を用いた行列-行列積演算を 50 回繰り返した中の最速の結果を予測する．

3. 性能評価

3.1 評価環境

名古屋大学情報基盤センターに設置されたスーパーコンピュータ「不老」Type II サブシステムを利用して性能評価を行った．構成を表 1 に示す．

表 1 評価環境：「不老」Type II サブシステム

製品名：FUJITSU PRIMERGY CX2570M5		
CPU	Intel Xeon Gold 6230	
	プロセッサ数	2CPU
	周波数	2.1 - 3.9 GHz
GPU	NVIDIA Tesla V100 SXM2 版	
	プロセッサ数	4GPU
	理論演算性能	7.8 TFLOPS / GPU
理論演算性能		33.888 TFlops / node
主記憶容量		384 TiB / node
主記憶帯域幅		281.5 GB/sec / node
コンパイラ		Intel C++ コンパイラ (ver.19.5.281)
コンパイルオプション		-xHost -O3 -qopenmp

3.2 実験条件

行列サイズは N=3000 とする．利用ノード数は 1 ノード，スレッド数は，20 スレッドで実行した．CPU 環境では 1CPU，GPU 環境では 1GPU で実行した．

3.3 試験行列

$C = A\,B$ 計算時の行列 A，B は，ともに疎度 90，92，94，96，98 の疎行列を使用する．このとき，いずれも $p=3$，$q=5$ となる．

3.4 結果と考察

図 1 に実測した結果を示す．試験行列は，疎度が高くなるほど，分解後の $A^{(i)}$の疎度が高くなるため，疎度が高くなると，疎行列演算を用いる実装の方が，演算時間が短くなる．

また，図 2 に予測 2 による予測時間を示す．評価の結果，相対誤差の絶対値の最大は予測 1 で 29.5%，予測 2 で 23.3% であった．

予測 1 による場合も，予測 2 による場合も白字で示す最速の実装は，図 1 の実測値での最速の実装と一致したが，相対誤差の絶対値は，予測 1 よりも，予測 2 の方が小さい場合が多かった．この原因の 1 つは，$A^{(1)}$ $B^{(1)}$, …, $A^{(p)}$ $B^{(q)}$ の pq 回の行列-行列積演算の実行時間の変動が大きいからと考えられる．

予測 1 の結果，各行列-行列積演算の実行時間の詳細と誤差の分析は，当日に発表する．

4. おわりに

本稿では，尾崎の方法による高精度行列-行列積演算において，AT 方式開発に資する予測方式を提案した．名古屋大学情報基盤センター設置の「不老」Type II サブシステムの CPU と GPU 環境を利用して性能評価を行った．性能評価の結果，予測 2 が正確であることが多いことがわかった．

今後の課題としては，試験行列生成時に乱数の種を変えて評価した場合の予測精度の再評価があげられる．

謝辞

本研究は JSPS 科研費 JP19H05662，および，学際大規模情報基盤共同利用・共同研究拠点の支援による(課題番号：jh200008-NAH)．

参考文献

[1] 市村駿太郎, 片桐孝洋, 尾崎克久, 荻田武史, 永井亨, 荻野正雄：マルチコア計算機による高精度行列 - 行列積アルゴリズムの性能評価，研究報告ハイパフォーマンスコンピューティング(HPC)，Vol.2017-HPC-160(16), pp.1-8 (2017).

[2] F. Ishiguro, T. Katagiri, S. Ohshima, T. Nagai: Performance Evaluation of Accurate Matrix-Matrix Multiplication on GPU Using Sparse Matrix Multiplications, Proc. of CANDAR2020 (2020)