INT8 行列乗算器を活用した 倍精度行列積エミュレーション手法の最適化

2431112 中嶋 陸 成見研究室

1 はじめに

近年の GPU は機械学習モデル向けに低精度の行列積計算に適した高性能な行列エンジンが搭載されている．一方，数値計算においては倍精度が必須であるが，倍精度の行列積エンジンの性能の向上は見られない．このような背景から，低精度演算ユニットを活用しつつ高精度な数値計算を実現するための手法が研究されている．尾崎らは高精度な行列乗算手法 (尾崎スキーム) を提案した [1]．この手法は，行列積を複数の低精度行列積の和に変換して計算する手法である．倍精度行列乗算 (DGEMM) については，FP16 や INT8 の行列乗算器を用いた尾崎スキームによるエミュレーションが提案されている．特に INT8 行列乗算器を使用した手法は ozIMMU と呼ばれている [2]， [3]．本研究では，ozIMMU_EF の実装の最適化手法及び倍精度複素行列積への拡張を提案する．

2 先行研究

2.1 FP32 を用いた DGEMM エミュレーション

FP32 演算を用いて FP64 精度をエミュレートする手法が提案されている [4]．代表的な方法として，倍々精度（Double-Double, DD）演算があり，FP64 の数を 2 つの FP32 で表現し，誤差補償を行うことで高精度演算を実現する．この手法は FP64 を備えないコンシューマ向け GPU などで有効である一方，演算回数の増加による性能低下が課題であり，FP64 性能が高いデータセンター向け GPU では有効性が低い．

2.2 ozIMMU

浮動小数点数集合を $\mathbb{F}$，b ビット浮動小数点数集合を $\mathbb{F}_b$，b ビット符号付整数集合を $\mathbb{I}_b$ とする．ozIMMU は尾崎スキームに基づき $A \in \mathbb{F}_{64}^{m\times k}$，$B \in \mathbb{F}_{64}^{k\times n}$ に対し，INT8 行列乗算器を用いて $C = AB$ をエミュレーションする．与えられた $\ell \in \mathbb{N}$ に対し，仮数部を β ビット以下となるよう抽出し，対角スケーリング $\mu \in \mathbb{F}_{64}^{m}$，$\nu \in \mathbb{F}_{64}^{n}$ を用いて

$$A = \mathrm{diag}(\mu)\sum_{i=1}^{\ell} 2^{1-i\beta}A^{(i)} + \Delta A, \tag{1}$$

$$B = \sum_{i=1}^{\ell} 2^{1-i\beta}B^{(i)} \cdot \mathrm{diag}(\nu) + \Delta B, \tag{2}$$

と無誤差分解する．ここで $A^{(i)} \in \mathbb{I}_8^{m\times k}$，$B^{(i)} \in \mathbb{I}_8^{k\times n}$ である．β は通常 7 を用いる．ozIMMU における行列積の近似値 $C \in \mathbb{F}_{64}^{m\times n}$ は

$$C \leftarrow \mathrm{diag}(\mu)\sum_{g=2}^{\ell+1}\sum_{s=1}^{g-1} 2^{2-g\beta}A^{(s)}B^{(g-s)} \cdot \mathrm{diag}(\nu) \tag{3}$$

として計算される．ここで各 $A^{(s)}B^{(g-s)}$ は INT8-GEMM により無誤差に計算され，それ以外の演算は FP64 で行う．

さらに，ozIMMU_EF では r 個の行列積の和が INT32 で無誤差に計算可能であることを利用する [3]．r を

$$r := \max\left(1, 2^{31-2\beta-\lceil \log_2 k\rceil}\right) \tag{4}$$

と定めると，式 (3) は

$$C \leftarrow \mathrm{diag}(\mu)\sum_{g=2}^{\ell+1} 2^{2-g\beta}\sum_{h=1}^{\lceil (g-1)/r\rceil} C^{(h)} \cdot \mathrm{diag}(\nu), \tag{5}$$

$$C^{(h)} \leftarrow \sum_{s=(h-1)r+1}^{\min(hr,g-1)} A^{(s)}B^{(g-s)} \tag{6}$$

と再構成できる．式 (6) の $C^{(h)}$ は INT8-GEMM のアキュムレータを用いて効率的かつ無誤差に計算される．

3 提案

ozIMMU_EF は INT8-GEMM が計算の大半を占めるが，INT8-GEMM は小規模行列では理論性能に達せず，データセンター向け GPU でも大規模行列で性能劣化が見られる．本研究ではこれらの性能劣化を改善するため行列サイズに応じた 2 種類の最適化手法を提案する．

3.1 小規模行列での最適化 (ozIMMU_EF_S)

ozIMMU_EF において，式 (4) で定義される INT32 で無誤差に計算することができる r 個の行列をひとつの大きな行列として計算することで性能を上げる．行列 $A^{(i)}, A^{(i+1)}, \ldots, A^{(j)}$，$B^{(i)}, B^{(i+1)}, \ldots, B^{(j)}$ に対し以下のような表記を導入する:

$$A^{(i,j)} := [A^{(i)}, A^{(i+1)}, \ldots, A^{(j)}] \in \mathbb{I}_8^{m\times k(j-i+1)} \text{ for } j \geq i,$$
$$B^{(i,j)} := [B^{(j)}; B^{(j-1)}; \ldots; B^{(i)}] \in \mathbb{I}_8^{k(j-i+1)\times n} \text{ for } j \geq i.$$

$A^{(i,j)}$，$B^{(i,j)}$ はそれぞれ $j-i+1$ 個の行列を行方向，列方向に連結した行列である．このとき式 (6) の右辺は以下のように変換される:

$$\sum_{i=(h-1)r+1}^{\min(hr,g-1)} A^{(i)}B^{(g-i)} = A^{((h-1)r+1,\min(hr,g-1))}B^{(g-\min(hr,g-1),g-((h-1)r+1))}.$$

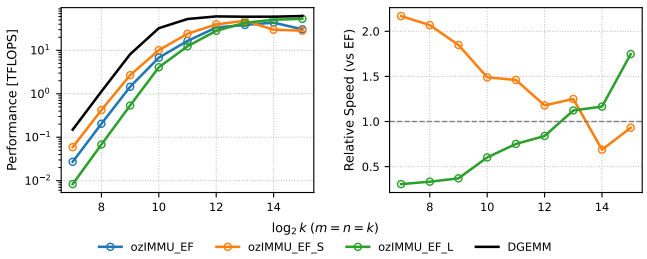

図 1　DGEMM エミュレーション性能比較 ($\ell = 7$)

3.2　大規模行列での最適化 (ozIMMU_EF_L)

行列をブロック分割することで INT8-GEMM の入力を適当な行列サイズへ縮小し，性能向上を図った．具体的には，入力行列 A, B のうち大きい行列に対して，A 行列の場合行方向に，B 行列の場合列方向に分割する．また，より効率的な計算のために，ブロック行列の積はストリームレベルの並列処理により行う．

3.3　複素行列積計算への拡張

3.3.1　Karatsuba 法を用いた拡張 (ozIMMU_EF_C_K_S)

Karatsuba 法では，複素行列の実部と虚部を用いて，複素行列積を三つの実行列積の組合せとして表現できる．具体的には，実部同士の積，虚部同士の積，および実部と虚部をそれぞれ加算した行列同士の積を計算することで，複素行列積の実部および虚部を構成できる．本研究では，この Karatsuba 法に基づく複素行列積の各実行列積を ozIMMU_EF_S により計算することで ZGEMM のエミュレーションを行う．

3.3.2　拡大行列法を用いた拡張 (ozIMMU_EF_C_E_S)

拡大行列法では，複素行列を実部と虚部に分解し，それらを用いて実数成分のみからなる拡大行列を構成する．具体的には，元の複素行列の実部および虚部をブロックとして配置した，行列サイズが 2 倍となる実行列を定義する．この拡大行列同士の積を計算すると，その結果は元の複素行列積の実部および虚部を含む構造を持つ行列となる．得られた行列の各ブロック成分を対応付けることで，複素行列積の結果を復元できる．本研究では，この拡大行列積を ozIMMU_EF_S により計算することで ZGEMM エミュレーションを行う．

4　性能評価

性能評価は NVIDIA Grace Hopper Superchip 上で行った．ozIMMU_EF_S および ozIMMU_EF_L の性能を図 1 に示す．分割数 $\ell = 7$ において，小規模行列では最大 2.2 倍，大規模行列では最大 1.7 倍の性能向上を達成した．

また, ozIMMU_EF_C_K_S, ozIMMU_EF_C_E_S および ZGEMM の性能を図 2 に示す．その結果，小規模行列に対しては拡大行列法が，大規模行列に対しては Karatsuba 法が有効であることを確認した．また，行列サイズが 2^{13} の時に，エミュレーションの方が ZGEMM 性能を超えることが分かる．

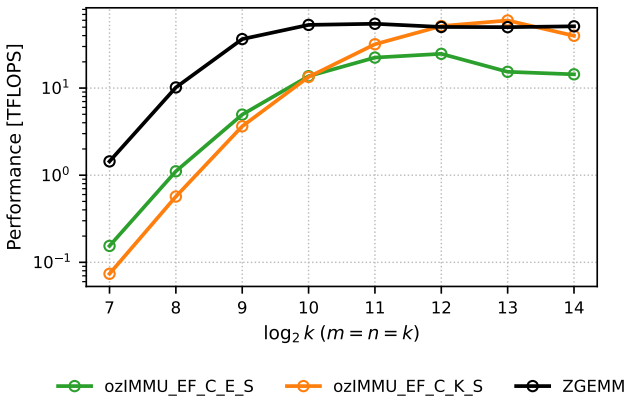

図 2　ZGEMM エミュレーション性能比較 ($\ell = 7$)

5　性能予測

本研究では, ozIMMU_EF_S の性能モデルを提案した．提案した性能モデルは，ozIMMU_EF_S を計算処理とデータハンドリングに分解し，ルーフラインモデルを尾崎スキームの構造を考慮し拡張したものである．性能予測では，最新の GPU である Grace Blackwell Superchip 上での性能予測を行い，ozIMMU_EF_S が DGEMM 以上の性能を達成する可能性を示した．

6　結論

本論文では，INT8 行列乗算器を用いた高精度行列積エミュレーション手法 ozIMMU_EF に対し，行列サイズに応じた最適化および倍精度複素行列積への拡張を行った．小規模行列には連結行列による統合手法を，大規模行列にはブロック分割とストリーム並列化を適用し，いずれの場合も性能向上を確認した．また，Karatsuba 法および拡大行列法に基づく複素行列積手法を提案した．性能予測では将来世代 GPU に対しても有効であることを示した．今後の課題として，より低精度演算器への拡張や分散環境での評価が挙げられる．

参考文献

[1] Katsuhisa Ozaki, Takeshi Ogita, Shin'ichi Oishi, and Siegfried M Rump. Generalization of error-free transformation for matrix multiplication and its application. *Nonlinear Theory and Its Applications, IEICE*, Vol. 4, No. 1, pp. 2–11, 2013.

[2] Hiroyuki Ootomo, Katsuhisa Ozaki, and Rio Yokota. DGEMM on integer matrix multiplication unit. *The International Journal of High Performance Computing Applications*, Vol. 38, No. 4, pp. 297–313, 2024.

[3] Yuki Uchino, Katsuhisa Ozaki, and Toshiyuki Imamura. Performance enhancement of the Ozaki Scheme on integer matrix multiplication unit. *The International Journal of High Performance Computing Applications*, Vol. 39, No. 3, pp. 462–476, 2025.

[4] T. J. Dekker. A floating-point technique for extending the available precision. *Numerische Mathematik*, Vol. 18, No. 3, pp. 224–242, June 1971.