5L-01

Android スマートフォンにおける NEON 命令を用いた階層的並列処理

Hierarchical Parallel Processing Using NEON Instructions on Android Smartphone

西倉　佑騎 †　　吉田 明正 †
Yuki Nishikura　　Akimasa Yoshida

1　はじめに

ARM マルチコアを搭載した Android スマートフォンは，マルチコアによるスレッドレベル並列処理と NEON 命令 (ARM advanced SIMD)[1] による SIMD 並列処理を併用して高速化を実現することが期待されている．Android 向け並列ソフトウェアは，NDK(Native Development Kit)[2] を用いて，OpenMP[3] 及び NEON のライブラリ関数を伴う C プログラムとして実装することが可能である．本稿では，OpenMP によるループ並列化と NEON 命令による SIMD 並列化を組み合わせた階層的並列処理手法を提案する．性能評価では，2 次元ウェーブレット変換プログラムに対して提案手法を適用しており，スマートフォン ASUS ZenFone 5Z で行った性能評価の結果，提案手法の有効性が確認された．

2　Android 環境における OpenMP/NEON を用いた並列処理

Android スマートフォンのアプリは，Android Studio の統合開発環境を用いて，Java 言語による開発が広く行われているが，NDK(Native Development Kit) の利用により，Android アプリの一部 (もしくはすべて) を C/C++で作成することが可能である．C/C++で作成したコードは，コンパイラにより共有モジュール (.so ファイル) として出力され，Java の実行ファイルから呼び出して利用する．NDK の利用により，OpenMP による並列化や SIMD 命令による並列化が可能になり，Android アプリの高速化が可能になる．

2.1　OpenMP による並列処理

スマートフォンのヘテロジニアスマルチコアを用いて，OpenMP によるループ並列処理を実現するために，指示文#pragma omp parallel for 及びその指示節として，スケジューリング方式を指定することができる．本手法では，4.1 節で述べる 4 種類のスケジューリング方式を伴うループ並列処理を行う．

2.2　SIMD 命令 (NEON) による並列処理

ARM プロセッサでは，NEON[2] と呼ばれる SIMD 命令が用意されている．NEON のデータ型は次のように宣言する．(type)(size)(number of lanes)_t と型が決まっており，例えば，int16x4_t は符号付 16bit 整数を 4 個持つ型を意味する．NEON の主な組み込み関数は表 1 の通りである．NEON は 128bit 幅の Q レジスタと 64bit 幅の D レジスタを使用して演算を行う．

3　離散ウェーブレット変換の階層的並列処理

本章では，性能評価に用いる 2 次元ウェーブレット変換とその並列化手法について述べる．

†明治大学総合数理学部ネットワークデザイン学科
Department of Network Design, School of Interdisciplinary Mathematical Sciences, Meiji University

表 1　NEON の主な組み込み関数.

ロード	vld1q_f32(a)
ストア	vst1q_f32(a)
加算	vaddq_f32(float32x4_t a, float32x4_t b)
乗算	vmulq_f32(float32x4_t a, float32x4_t b)
減算	vsubq_f32(float32x4_t a, float32x4_t b)
除算	vdivq_f32(float32x4_t a, float32x4_t b)

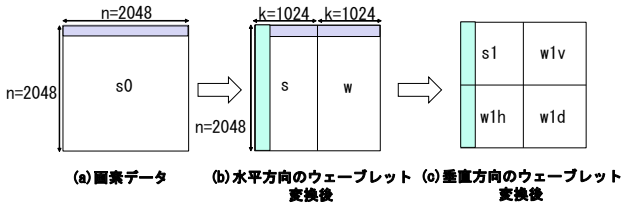

図 1　2 次元ウェーブレット変換の概念.

3.1　Ingrid-Daubechies ウェーブレット変換

Ingrid-Daubechies ウェーブレット変換 [4] は，多重度解像度解析を伴って画像圧縮に用いられるが，本稿では図 1 のようなレベル $j(j=1)$ の計算のみ扱う．スケーリング係数 $s_k^{(j)}$ およびウェーブレット展開係数 $w_k^{(j)}$ は，文献 [4] の数列 p_k と q_k（N=10，数列長=20）を使用し，式 (1) と式 (2) により求める．今回のプログラムでは図 1 のような $n \times n$ の画像 (n=2048) を入力とし，ウェーブレット変換によって，n × k(k=1024) のスケーリング係数 s，n × k のウェーブレット展開係数 w を求める．同様に s と w に対して垂直方向にウェーブレット変換を適用し，s1,w1h,w1v,w1d を求める．s1 には s0 を平均化した低域成分，w1h には水平方向の高域成分，w1v には垂直方向の高域成分，w1d には対角方向の高域成分が現れる．

$$s_k^{(j)} = \sum_n \overline{p_{n-2k}} s_n^{(j-1)} \tag{1}$$

$$w_k^{(j)} = \sum_n \overline{q_{n-2k}} s_n^{(j-1)} \tag{2}$$

3.2　ウェーブレット変換の並列処理

本節ではレベル j=1 の水平方向のウェーブレット変換の過程，即ち図 1(a) の配列 s0 を入力として，図 1(b) 左側のスケーリング係数 s を求める過程を説明する．まず，図 1(a) の行単位でマルチコアを用いたループ並列処理を適用する．この並列コードは OpenMP 指示文により実現される．

次に，各行のデータに対して文献 [4] の数列 p_k（数列長=20）との内積を求める際に，NEON の SIMD 命令を用いる．具体的には図 2(a) のような NEON コードを

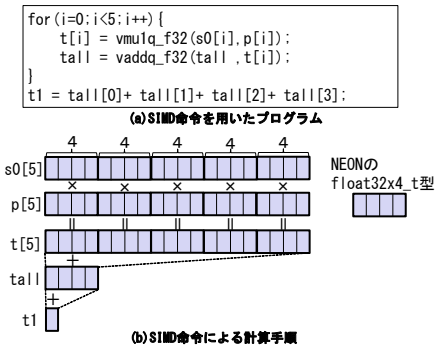

図 2　2 次元ウェーブレット変換プログラムにおける SIMD 命令の適応.

表 2　性能評価に用いる Android 搭載マシン.

マシン	ASUS ZenFone 5Z
プロセッサ	Qualcomm Snapdragon 845
CPU コア	Cortex-A75(2.8GHz)+A55(1.8GHz) (4 コア+4 コア)
メモリ	6GB
OS	Android 8.0(API level 26)

実行することにより，スケーリング係数の 1 つ（この例では変数 t1）を求めることができる．NEON 命令では 128 ビットのレジスタを用いるため，float 型変数の場合，4 要素までしか同時に演算することができない．そこで，20 要素のベクトル内積（図 2(b) の s0[5] と p[5] の内積）の場合，4 要素単位の部分ベクトルの内積として求め，それらの総和を tall に求め，4 要素の和を t1 に求める．これらの方法により，ループ並列処理と NEON による SIMD 並列化を階層的に行うことが可能になる．

4　Snapdragon845 搭載スマートフォン上での性能評価

本稿では，2 次元ウェーブレット変換プログラム [5] を用いて性能評価を行う．評価マシンは表 2 のスマートフォンとした．本性能評価では，画像圧縮の対象として画像データ (2K:2048*2048) を用意した．画素値としては Y，Cb，Cr に変換された Y 成分のウェーブレット変換の処理時間を測定した．

4.1　ループ並列処理 (NEON なし) による性能評価

本節では，SIMD 命令（NEON）を利用せず，ループ並列処理（4 種類のスケジューリング）のみを 2 次元ウェーブレット変換プログラムに適用し評価を行う．実行結果は図 3 に示す通りであり，ループイタレーションのスケジューリング方法は，(static,1)：サイクリック分割，(static)：ブロック分割，(dynamic,8)：チャンク=8 の動的割り当て，(dynamic,1)：チャンク=1 の動的割り当ての 4 種類としている．評価の結果から，(dynamic,1) の 8 スレッドの場合が最も高速であり，逐次実行比で 4.97 倍の速度向上率が得られた．

4.2　階層的並列処理 (NEON あり) による性能評価

次に，ループ並列処理（(dynamic,1) のスケジューリング）と NEON による SIMD 並列化を組み合わせた階層的並列処理による実行結果を図 4 に示す．NEON なし 8 スレッド実行では，実行時間は 1113[ms] となり逐次実行比で 3.86 倍であった．それに対して，NEON あり 8 スレッド実行では，実行時間は 215[ms] となり逐次実行比で 19.9 倍となった．これらの結果から OpenMP による並列化と NEON 命令による並列化を組み合わせた階層的並列処理手法の有効性が確認できた．

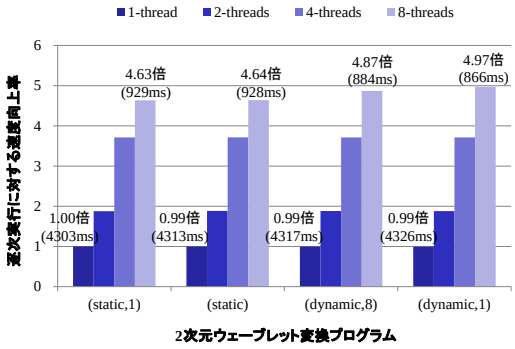

図 3　2 次元ウェーブレット変換プログラム上のループ並列処理 (NEON なし) による性能評価.

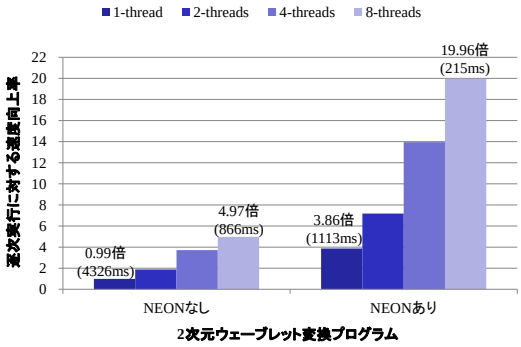

図 4　2 次元ウェーブレット変換プログラム上の階層的並列処理 (NEON なし，あり) による性能評価.

5　おわりに

本稿では，OpenMP を用いたループ並列処理と NEON 命令による SIMD 並列化を組み合わせた階層的並列処理手法を提案した．性能評価では 2 次元ウェーブレット変換プログラムに対して提案手法を適用し，Snapdragon845 搭載スマートフォンで性能評価を行った．ループスケジューリング方法は，(dynamic,1) が最も高速であり，さらに SIMD 並列化を組み合わせた階層的並列処理では，8 スレッド実行で最大 19.96 倍の速度向上が得られた．これらの結果から，NEON 命令を用いた階層的並列処理の有効性が確認された．今後の課題としては，Android 上でのタスク駆動型粗粒度並列処理 [6] に本手法を組み込むことが挙げられる．

参考文献

[1] ARM NEON Intrinsics Reference. http://infocenter.arm.com/help/topic/com.arm.doc.ihi0073b/IHI0073B_arm_neon_intrinsics_ref.pdf, 2016.

[2] 出村成和．Android NDK ネイティブプログラミング第 2 版，秀和システム，2013.

[3] OpenMP．https://www.openmp.org/，2018.

[4] 中野宏毅, 山本鎮男, 吉田靖夫．ウェーブレットによる信号処理と画像処理．共立出版,1999.

[5] 吉川一輝，高平真由，吉田明正．マルチ GPU 上での画像圧縮における離散ウェーブレット変換の階層的並列処理，情報処理学会第 16 回情報科学技術フォーラム I-014, 2018.

[6] A.Yoshida, A.Kamiyama, H.Oka. A Task-Driven Parallel Code Generation Scheme for Coarse Grain Parallelization on Android Platform. Journal of Information Processing, Vol.25, pp.426-437, 2017.