Pthread を用いた MRI 画像再構成アプリケーションの高速化

南波 孝輔 † 伊藤 聡志 † 山田 芳文 † 大津 金光 † 横田 隆史 † 馬場 敬信 †
† 宇都宮大学工学部情報工学科

1 はじめに

コンピュータの普及とプログラミング言語の発達によりさまざまな分野で処理を行う実アプリケーションが作られると同時に、その実時間応答性の向上を求められるようになった。その要求に対し、マルチプロセッサ技術を利用したマルチスレッドプログラミングで実アプケーションの高速化を図る。本稿で対象とする実アプリケーションは医療画像処理で使用されている MRI 画像再構成アプリケーションである。マルチスレッドプログラミング適用範囲は全実行時間のうち一番多く実行時間を占める関数に Pthread を用いてマルチスレッドプログラミングを適用し、その場合のスレッド数と速度向上率の関係性を示す。

2 MRI 画像再構成アプリケーション

2.1 概要

MRI 装置で収集した投影再構成映像法の NMR 信号データを読み込み，投影再構成により画像を再生するアプリケーションである。投影再構成は、フィルタ補正逆投影法と２次元逆フィルタ法の２種類の画像再構成法から選択可能である。

2.2 適用範囲

本稿では、実行時間を多く占める関数に対して、マルチスレッドプログラミングを適用し、スレッド数と速度向上率の関係性を示す。MRI 画像再構成アプリケーションにおいて、画像再構成を行う関数 reconstruction に対してマルチスレッドプログラミングを適用する。関数内においてフィルタ補正逆投影法と２次元逆フィルタ法の２種類の画像再構成法があり、それぞれのアルゴリズムに対し適用する。関数 reconstruction の処理内容を以下に示す。

1. NMR 信号データを２次元作業配列に格納する。
2. NMR 信号データは投影データの逆フーリエ変換になっているので２次元作業配列を行方向にフーリエ変換する。
3. 投影データは実数であるので絶対値をとる。
4. コマンドラインで指定されたフィルタ補正逆投影法または２次元逆フィルタ法を適用し、画像再構成を行う。

3 並列アルゴリズム

3.1 概要

画像再構成を行う関数 reconstruction は NMR 信号データを２次元作業配列に格納し、その配列に対しさまざまな処理を行う。その処理は各１画素ごとに行うため２重ループや３重ループで構成されている。そこで処理対象である２次元作業配列をブロック分割することでマルチスレッドプログラミングを適用し、スレッドごとに画像再構成を並列に行うアルゴリズムである。画像再構成におけるフィルタ補正逆投影法と２次元逆フィルタ法はいずれもブロック分割によりマルチスレッドプログラミングを適用している。並列アルゴリズムに関しても２種類とも同一である。以下に並列アルゴリズムの概要を示す。

1. 関数 reconstruction へ突入する直前でメインスレッドから子スレッドを生成する。
2. スレッドごとに２次元作業配列のブロック分割を行う。
3. スレッドごとに割り当てられた行列領域を画像再構成を行う。

3.2 スレッド生成

本稿でマルチスレッドプログラミング適用関数である関数 reconstruction に突入する直前にメインスレッドより指定された数の子スレッドを生成する。このとき同時にスレッドごとに総スレッド数とスレッド識別番号を付加する。

3.3 ブロック分割

画像再構成において、データファイルを作業用２次元配列に格納し、２次元作業配列対する２重ループや３重ループでの処理が行われる。そこで本稿ではブロック分割でマルチスレッドプログラミングを適用する。ブロック分割とは図１のようにスレッドごとに行列の領域を均等に分割し、スレッドごとに処理を行う適用方法である。例えば、スレッド数が２ならば１プロセッサあたりの処理量が１/２になり、実行時間の短縮を図ることができる。ブロック分割を適用するにあたり２次元作業配列に対するループをスレッドごとに分割することで実現する。そこで適用のため以下の段階を踏む。

1. スレッドごとのループイテレーションの初期値と終了値の決定する。
2. 初期値と終了値をループへ適用する。

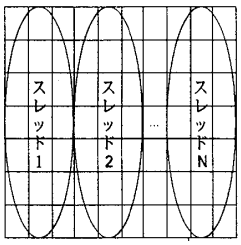

図 1: ブロック分割イメージ

3.3.1 スレッドごとのループイテレーションの初期値と終了値の決定

ブロック分割によるマルチスレッドプログラミングを適用するにあたり２次元作業配列をスレッドごとに分割する必要がある。そこでスレッドごとの総スレッド数とスレッド識別番号を用いてループイテレーションの初期値と終了値を決定する。その具体的決定方法を以下に示す。

The speedup of the MRI image reconstruction application by Pthread

† Kousuke Nanba , Satoshi Ito , Yoshifumi Yamada , Kanemitsu Ootsu , Takashi Yokota and Takanobu Baba Department of Information Science、Faculty of Engineering,Utsunomiya University, (†)

$$初期値 = \frac{行列サイズ \times スレッド識別番号}{総スレッド数} \quad (1)$$

$$終了値 = \frac{行列サイズ \times (スレッド識別番号 + 1)}{総スレッド数} \quad (2)$$

3.3.2 2重、3重ループへの適用

上記の初期値と終了値をループに適用することによってブロック分割を実現している。適用方法は、最外ループのループイテレーションに上記の初期値と終了値を設定することによってスレッドごとにブロック分割を実現している。

3.4 バリア同期

処理はスレッドごとに行われるため、たとえスレッドごとに同じ処理量を割り当てたとしてもスレッドごとに実行時間にずれが生じる可能性がある。それによりデータ依存関係や処理手順が保たれない場合が生じる。そこで本稿では、データ依存関係や処理手順を保つための同期方法としてバリア同期を使用する。

バリア同期とは、全スレッドがある段階まで処理を進めるのを待つ同期法である。スレッドごとの実行時間差分だけプログラムの流れを止めるものであるためオーバーヘッドとなる。

具体的に図2を例に説明する。処理1から処理2へデータ依存関係があるとする。処理1のスレッドごとの処理量がたとえ同じであってもメインスレッドと子スレッドが同時に処理1を終了するとは限らない。そこですべてのスレッドが処理1を終了するまで待機させるためバリア同期を使用する。

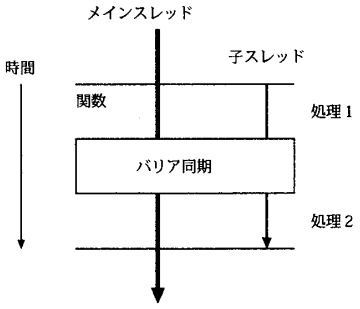

図 2: バリア同期

4 評価

4.1 実行環境

実行環境を以下の表1に示す。

表 1: 実行環境

OS	CentOS release 5
カーネル	2.6.18-8.1.8.el5
CPU	Intel Xeon 3.6GHz × 2
メモリ	3.7GB
コンパイラ	gcc Compiler
最適化オプション	-O3

4.2 スレッド数と速度向上率

マルチスレッドプログラミング適用関数 reconstruction の画像再構成におけるフィルタ補正逆投影法と2次元逆フィルタ法に上記のブロック分割を適用した場合のスレッド数と速度向上率を示す。対象データファイルの行列サイズは 256 とする。

4.2.1 フィルタ補正逆投影法

画像再構成においてフィルタ補正逆投影法を適用した場合を対象とした平均実行時間 (sec) と速度向上率 (倍) を以下の表2に示す。平均実行回数は5回で MAX と MIN は5回の実行のうちの最長実行時間 (sec) と最短実行時間 (sec) を示している。速度向上率は以下の式に基づく。(表3も同様)

$$速度向上率 = \frac{シングルスレッド実行時間}{マルチスレッド実行時間} \quad (3)$$

スレッド数を増やすごとに高い速度向上率を得ることができたが、線形速度向上とはならなかった。原因としてスレッド生成によるオーバーヘッドとバリア同期の多さが挙げられる。

表 2: スレッド数と速度向上率

スレッド数	1	2	3	4
平均実行時間	0.1926	0.1145	0.1115	0.1060
MAX	0.1952	0.1160	0.1129	0.1084
MIN	0.1911	0.1123	0.1089	0.1041
速度向上率	1.0000	1.6830	1.7279	1.8172

4.2.2 2次元逆投影法

画像再構成において2次元逆投影法を適用した場合を対象とした平均実行時間と速度向上率を以下の表3に示す。スレッド数を増やすごとに高い速度向上率を得ることができたが、線形速度向上とはならなかった。原因としてスレッド生成によるオーバーヘッドとバリア同期の多さが挙げられる。

表 3: スレッド数と速度向上率

スレッド数	1	2	3	4
平均実行時間	0.2095	0.1114	0.1082	0.0954
MAX	0.2121	0.1134	0.1098	0.0972
MIN	0.2064	0.1097	0.1071	0.0947
速度向上率	1.0000	1.8809	1.9357	2.1965

5 おわりに

本稿では，実アプリケーションである MRI 画像再構成アプリケーションに Pthread を用い、一番多くの実行時間を占める関数 reconstruction に対し、マルチスレッドプログラミングを適用した場合のスレッド数と速度向上率を示した。

今後の課題は、より実行時間の長い実アプリケーションなどさまざまな実アプリケーションに対し評価するとともに、他の手法や異なるアルゴリズムでのマルチスレッドプログラミングを適用した場合の評価も重要となる。

謝辞 本研究は、一部日本学術振興会科学研究費補助金（基盤研究 (B)18300014、同 (C)19500037、若手研究 (B)17700047）および宇都宮大学重点推進研究プロジェクトの援助による。

参考文献

[1] 河田聡、南茂夫編著:"科学計測のための画像データ処理",pp.221-236,1994,CQ 出版

[2] Bradford Nicbols, Dick Buttlar, Jacqueline Proulx Farrell 共著 榊 正憲 訳:"Pthreads プログラミング" オイラリー・ジャパン出版