B-019

PC クラスタによる FFT 処理の実行と性能評価
Performance Evaluation of FFT on PC Cluster

齋藤　毅　　當山　孝義　　谷澤　茂
日本工業大学　電気電子工学科

１．はじめに

近年、DVD レコーダーやデジタルビデオカメラの普及により、デジタル動画が身近なものとなっている。これにともない、パソコンで動画の視聴や作成、編集を行なうことが容易に可能となり、普及しつつある。このとき画像の高圧縮化及び、高画質化を目的として各種フィルタ処理を施す場合が多い。しかし、高精細な画像に対してはこれらの処理の負荷が高く、短時間で十分な処理を行なうために高性能計算の必要性が高まっている。一方、パソコンの価格低下に伴い、一般家庭においても複数台のパソコンが置かれるようになっている。

そこで、本研究では、廉価なパソコンとネットワークで構成された PC クラスタによる、高性能な動画処理の可能性を探ることを目的とする。本稿では、PC クラスタ上でフィルタ処理の一つである FFT を実行し、その性能を評価する。

２．評価システム

今回使用した PC クラスタシステム構成を示す。使用したパソコン９台のうち、１台をホスト、残り８台をノードとする。ホスト上の画像データに対し処理を行なう。

表 2.1　評価システム

PC９台	CPU		Intel Pentium4 1.6GHz
	マザーボード		ASUS P4T533-C
	メモリ		RD-RAM PC-800 256MB
	NIC	ホスト	BUFFALO LGY-PCI32-GT
		ノード	Intel PRO/1000MT
	OS		Red Hat Linux 9.0
	並列ライブラリ		MPICH 1.2.5
			LAM 6.5.8
Hub			MicroSolution GSH0801a×2（トランキング接続）

３．画像の FFT 処理

画像処理においては画像の縦軸、横軸それぞれに一次元 FFT を行なうことが多い[1]。そのため、並列化の方法として、一次元 FFT 自体を並列化する方法[2]（並列 FFT）と、画像を各ノードに分割配分して並列に行なう方法（画像分割法）について実験を行なう。

３．１　並列 FFT

FFT のアルゴリズムは、バタフライ演算と呼ばれる演算ユニットの組み合わせで構成されている。図 3.1 にバタフライ演算を使用したプロセッサ数 p=2、データ数 n=16 の場合の並列アルゴリズムを示す。図中の X は入力、Y は出力である。

このバタフライ演算をある段階から分けることができ、計算の段階数をｒ、計算を分けることができるポイントをｄとすると、以下の関係が成り立つ。

$$r = \log_2 n$$
$$d = \log_2 p$$

図の場合は、m=1 の段階から２つの計算に分けることができ、それを２つのプロセッサで並列計算を行なわせることが可能となる。

並列 FFT では、各行ごとにノードからデータを集める必要があるため、通信回数が増えてしまい、全体の処理時間を増加させる可能性がある。

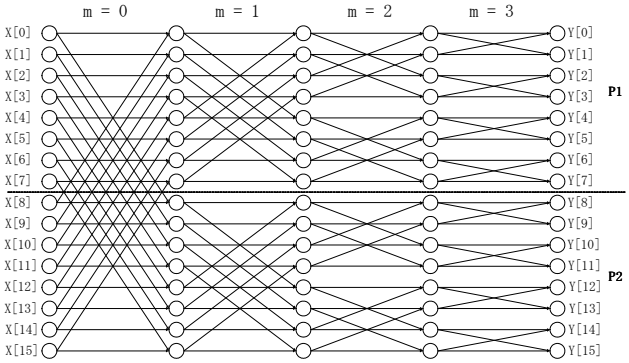

図 3.1　並列 FFT

３．２　画像分割法

図 3.2 に、１枚の画像データをノード数で分割し、各ノードで逐次アルゴリズムにより FFT を行なう画像分割法を示す。

この方法では、並列 FFT のように各行ごとにノードからデータを集める必要がないため通信回数を少なくできる。

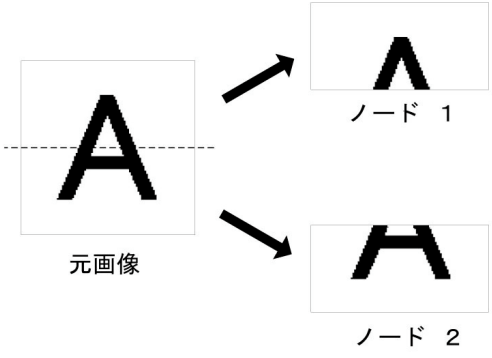

図 3.2　画像分割法

Tsuyoshi Saitoh, Takayoshi Touyama and Shigeru Tanisawa
Dept. of Electrical and Electronics Engineering , Nippon Institute of Technology

４． 実験

上記の２つの方法を、MPICH と LAM の２種類の MPI ライブラリを用いて実装し、その実行時間を測定する。以下に解像度 1024×1024、2048×2048、4096×4096 のカラー画像に対する実験結果を示す。

図 4.1，4.2 は解像度 1024×1024 に対する実行時間である。逐次プログラムでは約２秒であった。MPICH、LAM のどちらの場合も、画像分割法ではノード数の増加につれて実行時間は短くなり、ノード数８では逐次プログラムよりも高速化が得られている。しかし、並列 FFT では全体的に高速化が得られず、ノード数に比例した高速化が得られていない。これは、ノード数が増えると並列計算が可能な部分が少なくなってしまうことと、通信回数がノード数に比例して増えてしまうためだと考えられる。

また、LAM の方が全体的に MPICH よりも実行時間が早い結果が得られた。画像分割法では、MPICH と LAM の差はそれほど大きくはなかったが、並列 FFT では２つの差が画像分割法に比べて大きく、ノード数が８の場合には MPICH の実行時間が LAM の倍近い値になっている。

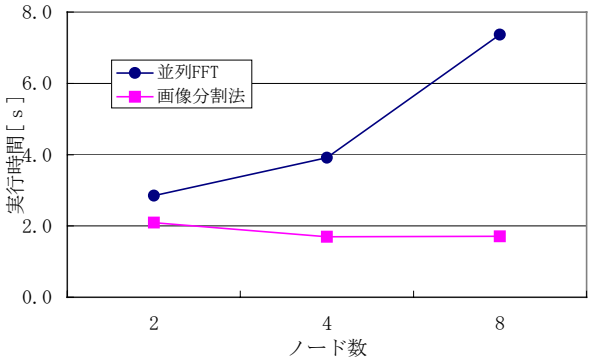

図 4.1　FFT の実行時間
（MPICH，解像度 1024×1024）

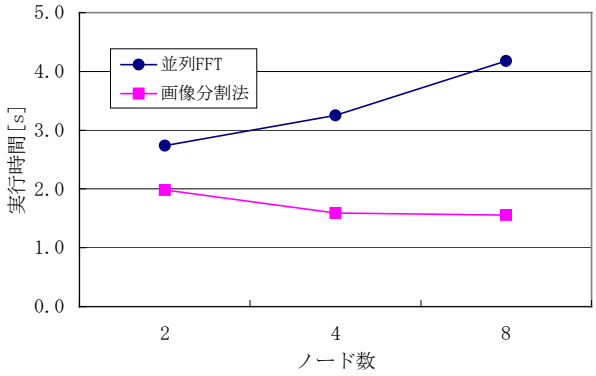

図 4.2　FFT の実行時間
（LAM，解像度 1024×1024）

図 4.3 は MPICH、図 4.4 は LAM の逐次プログラムに対する高速化率のグラフである。画像分割法では全体的に高速化が得られている。

並列 FFT では全体的に高速化が得られていないが、LAM では解像度に比例して高速化率が上がっており、より大きな画像であれば高速化が得られると考えられる。

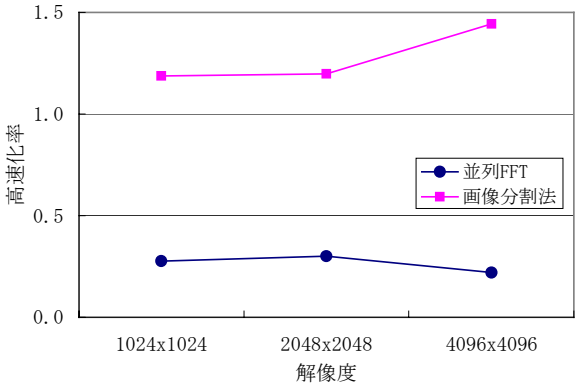

図 4.3　逐次プログラムに対する高速化率
（ノード数８，MPICH）

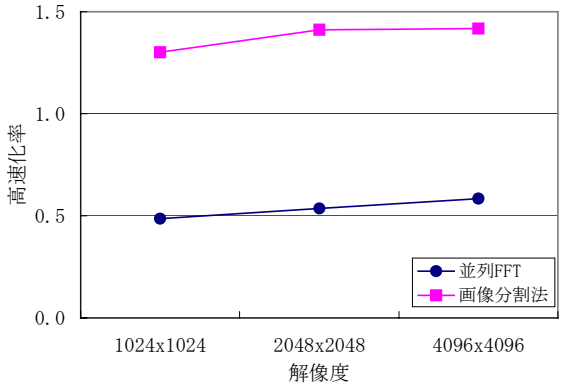

図 4.4　逐次プログラムに対する高速化率
（ノード数８，LAM）

５． おわりに

本稿では、パソコン９台からなる一般的なクラスタシステム上で FFT 処理を行い、その性能を評価した。

その結果、現在の画像に対する FFT のような処理においては通信部分の影響が大きく、その効率化が重要な課題であることが明らかになった。

参考文献

[1]井上、他：C 言語で学ぶ実践画像処理，第 1 版，P151-172，オーム社(1999)

[2]Kumar et,al.：Introduction to Parallel Computing，P377-388，Benjamin/Cummings(1994)