FFT向け並列計算機*

6K－3

桐山佳隆 柳瀬正俊 朝生良教 林達也
名古屋工業大学 電気情報工学科†

1. はじめに

本稿では，高速フーリエ変換 (FFT) を並列計算機 AP1000 上で高速に処理するための方式について考察する．

AP1000 はメッセージパッシング型並列計算機であるが，メッセージパッシング方式は到着したデータを一度バッファに格納するため，すぐにデータを使用できないためにオーバヘッドが生じる．そこで本方式は FFT のように SIMD 的に動作するアプリケーションに対して，受信プロセッサのメモリに直接データを書き込むことによってオーバヘッドを削減するものである．その結果通信時間が短縮され，また受信と計算を同時に実行することも可能になり，処理の高速化が図れることを示す．

2. 並列FFTアルゴリズムとAP1000の通信方式

本節では今回用いた並列 FFT アルゴリズムについて述べ，次に AP1000 の通信方式を説明する．

2.1 並列FFTアルゴリズム

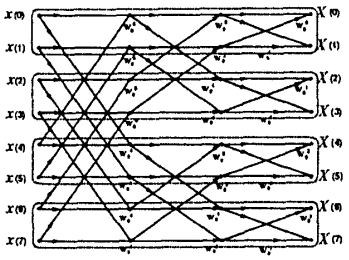

図1: 並列 FFT (N = 8, P = 4 の時)

N 点 FFT は図 1 のようなバタフライ演算で表される．但し N 個の複素入力データ列を $x(n)$, $x(n)$ のフーリエ係数 $X(n)$ $(0 \leq n \leq N-1)$, 回転子 $W_N = \exp(-2\pi j/N)$ と表す．これを P 台のプロセッサで並列処理させるために，図 1 の点線で囲んだように連続する N/P 個のデータをプロセッサ 0 から順に割り付け，データ授受はプロセッサ間通信を用いて行なう．

2.2 AP1000の通信方式

AP1000 の相互結合網はトーラス結合網であり，各プロセッサはローカルメモリの他に 送信用，受信用のメッセージバッファとメッセージバッファと結合網の間の操作を行なう DMA を備えている．これらを用いてプロセッサ間通信は以下の手順で行なわれる.[3]

- 送信
  (1) 指定されたデータを送信用バッファにコピーする．
  (2) DMA が送信用バッファのデータを結合網へ流し，指定プロセッサへ送られる
- 受信
  (1) データが到着すると割り込みが発生し DMA が起動され，データは受信用バッファに格納される．
  (2) バッファにあるデータをメモリへコピーする．

FFT は前述したアルゴリズムにより行なわれるため，各プロセッサ内での処理は 送信，受信，演算の繰り返し操作であることがわかる．よって上記の手順で 送信 (1)，受信 (2) の操作を Copy，送信 (2) の操作を Send，受信 (1) の操作を Recv で表すと，プロセッサ内での処理は図 2 で示す通信モデルのようになる．

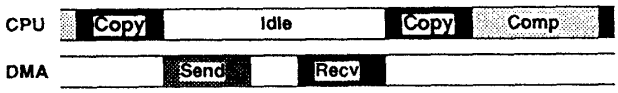

図2: AP1000 の通信方式

この図から Copy 操作が通信オーバヘッドになっていることがわかる．よって次節では，このオーバヘッドを削減する方式について述べる．

*Parallel Computer Architecture for FFT
†Department of Electrical and Computer Engineering, Nagoya Institute of Technology, Gokiso Shouwa-ku Nagoya 466, Japan

3. FFT に適した通信方式

Copy 操作はメッセージバッファを用いることに起因する. オーバヘッド削減のためメッセージバッファを用いないとすると, 送信プロセッサのメモリから受信プロセッサのメモリへ直接転送する方式が考えられるが, この場合次の問題が生じる可能性がある.

(1) 送信中に CPU が次の処理をできなくなる.

(2) データを書き換えてしまい, 間違った値を使用してしまう.

FFT の場合について考察すると, 図 2 から明らかなように送信中 CPU はアイドルになっているので (1) に関しては問題ない. 次に (2) のようなことが生じるのは, 演算中にデータの到着がある場合であるから, データを書き換えて違う値を使用することはない. よって FFT の場合はメモリからメモリへ転送しても問題はないといえる.

次にその実現方式について述べる. バッファを用いない送信はラインセンド [3] という機能で既に実装されており, これを用いればよいと考える. この機能はキャッシュ上にデータが存在する場合はキャッシュからデータを結合網に流すものである. 一方受信に関しては, データに受信プロセッサ内のアドレスを付加して転送するようにすればよい. こうすることによって, データが到着すると DMA が起動されそのアドレスを読むことによって直接メモリに書き込むことが可能になる. この時送信側で受信するアドレスが前もってわかっていることが必要であるが, FFT ではデータの動きがわかっているので実現可能である.

以上により Copy 時間がなくなり DMA は到着したデータをメモリへ格納する操作を行なうことになるが, この間依然として CPU アイドルのままである. このことはすべてのデータの到着を待って演算を開始していることが原因である. そこで利用可能なデータが到着次第演算を開始できるとすると, DMA がメモリに書き込み中に演算を実行することが可能になる. これは同期ビットを付加するなどの方法で, 違う値を使用することを防ぐことで実現できる.

4. 性能の見積り

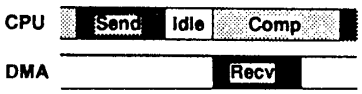

図 3: オーバヘッドを削減した通信方式

前節の方式によって通信モデルは図 3 のようになる. この図と図 2 を比較すると, 削減されるのは Copy 時間と Recv と Comp の小さい方の時間であることがわかる.

よって本方式の全処理時間は, AP1000 で実行させた時の全処理時間からこれらの時間を引けばよい. 図 4 に AP1000 の全処理時間で正規化した結果を示す. 但し AP1000 ではイベント別に処理時間を測定できるが, 送信に関しては送信時の Copy 時間と Send 時間の合計しか測定できないため, 図は送信時の Copy 時間を含む.

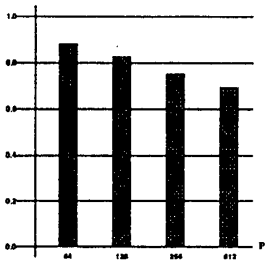

図 4: 処理時間の比 (N = 1024)

これから P(プロセッサ台数) が大きいほど, 効果は大きいことがわかる. これは P が大きいと全処理時間に占める通信時間の割合も大きくなるためである.

5. おわりに

AP1000 で FFT を行なうのに適した方式として, メモリへ直接転送し受信と演算を同時に実行する方式を示し, 性能を見積もった. その結果, 小さいサイズのデータを何度もやりとりする場合は特に効果があることがわかった.

今回は FFT の場合について考察したが, この方式は一斉に通信, 演算を行なうようなアプリケーションならば, かなりオーバヘッドは削減できると考えられる.

参考文献

[1] 富田眞治, "並列計算機構成論", 昭晃堂, 1985

[2] 辻井重男 , 鎌田一雄, "ディジタル信号処理", 昭晃堂, 1990

[3] T. Shimizu, T. Horie, and H. Ishihata, "Low-latency message communication support for the AP1000", Proc. of 19th ISCA, pp. 46-53, May 1992