資源共有型並列計算機“砂丘”の応用について

5G－5

河野光明[1]　井上倫夫[1]　小林康浩[1]　加納尚之[2]
[1]（鳥取大学工学部）　[2]（米子工業高等専門学校）

１．はじめに

我々の研究室では，実験室レベルで特定ユーザーが利用する数値シミュレーションマシンとして，資源共有型並列計算機“砂丘”を開発している．“砂丘”は，複数のマイクロプロセッサ（μＰ）を密に結合したマルチマイクロプロセッサシステムである．

本報告では，地震波，音声の解析などに用いられているケプストラム分析を，本システムを用いて実際に行い，この問題をどのように並列処理させるかについて述べる．

２．システムの制御について

２．１　実際の並列処理について

まず，マスターとなるプロセッシング・ユニット（ＦＥＰ）が使用するメイルボックスを確保し，必要なデータをメイルボックスに書き込む．次にプロセス番号の初期化を行い，システムコントローラにどのメイルボックスを使うかの番号（これをモードという）を書き込む．すると，各ＰＵに割り込みがかかり，これと同時に同期フラグがセットされる．プロセスの起動の割り込みを受けたプロセッシング・ユニット（ＰＵ）はモードを読み込む．モードにはメイルボックス番号が書かれているので，その番号のボックスから必要な情報を読み込む．各ＰＵはその情報に従ってプロセスの実行を行う．プロセスの終了後，各ＰＵは同期フラグをリセットし，ＦＥＰは割り込みをかけた全てのＰＵの同期フラグがクリアされるのを確認し，全てクリアされると通信に使用したメイルボックスを解放する．以上のような手続きを踏んで並列処理を実行する．

２．２　プロセス制御

複数のプロセスの生成を行う場合，各プロセスが生成時に違ったプロセス番号を得る事により，プロセスがそれ自身を認識し，同じ処理の違った部分（または，完全に異なる処理）を行う事が可能となる．従って，多数のプロセスを限られたＰＵで効率よく並列処理するためには，各ＰＵの処理するプロセス番号管理が必要である．この管理を通常の共有変数を用いた場合，この変数に対する排他制御が必要である．この際，専有権争奪に於いて競合が起こり，プロセス起動時のオーバーヘッドが増大する．

本システムでは，プロセス番号の管理をハードウェア化した Fetch and Add操作で行うようにした．この番号をカウンタで管理し，ＰＵからのリードアクセスが終了すると，自動的に指示値を増すようにした．従って，プロセス番号操作の為のアクセスは１回となり，プロセス起動時のオーバーヘッドを抑えることができる．

３．応用例

並列処理実行の具体例として，ケプストラム分析を行った．図１に音声信号のケプストラム分析の処理フローを示す．このケプストラム分析を１度に扱うデータ量の大きさによって，２通りの並列処理の方法を考えた．

１）データ量の少ない場合（方法１）

音声を分析する場合には，音声が時間にともなって変化するということから，時間を追って分析する必要がある．従って，少ないデータ点数（２５６点）を１台のＰＵでケプストラムの処理を行った．時間の変化に応じて，処理するデータの位置が違うので，プロセス番号を用い

Parallel Processing on The Closely-Cuppled MIMD Type Parallel Processing System “SAKYU”
Mitsuaki Kouno[1], Michio Inoue[1], Yasuhiro Kobayashi[1], Naoyuki Kanou[2]
[1]Tottori Uviversity　[2]Yonago National College of Technology

て，どこのデータを処理するのかを決定し，違った場所のデータを各ＰＵが並列に実行した．この場合には，処理の終了したＰＵが次のプロセス番号を読みに行き，その番号のプロセスを次々と処理して行けば良いので，プロセスの起動時の通信も最初に行うだけでよく，プロセスの処理中には同期を取る必要はない．

２）データ量の多い場合（方法２）

地震波等を分析する場合には，音声の場合と違って，１回で大量の纏まったデータを処理する必要がある．そこで，フーリエ変換，ＬＯＧ等の処理を複数のＰＵで並列に実行した．この場合には各処理の始めに起動をかけるために通信を行い，終わりには同期を取る必要がある．Ｎ点のＦＦＴを行う場合，各ステージ（$\log_2 N$回）で通信及び同期を取る必要がある．

４．検討

１つのジョブを処理するのに要する時間をｔ（０）＝Ｔとし，そのジョブをｎ台のＰＵで処理するとする．この時のプロセスの起動時の通信及び同期を取る時間をｃ，また１つのジョブで通信・同期を行った回数をｍとする．この時に要する時間をｔ（ｎ，ｍ）とすると，速度向上比Ｓは次式のようになる．

$$S=\frac{t(0)}{t(n,m)}=\frac{T}{m\cdot c+\frac{T}{n}} \qquad (1)$$

本システムでは，プロセス番号を得るためにFetch and Add 回路を用いている．これを使わない場合は排他制御を行いながら，共有変数を操作しなければならず，データ量の少ない場合の処理を方法２で行った場合，Fetch and Add 回路を用いた場合の約２倍の通信時間がかかる．これを（１）式を用いてグラフにしてみると図２のようになる．

５．おわりに

以上，本システムを用いて実際に並列処理を行い，扱う問題に応じて，どのように並列処理させたらよいかを示した．これからの課題として，本システムを有効に利用できる専用言語の開発が望まれる．

参考文献

［１］　武田　他　：“資源共有型並列計算機“砂丘”のアクセス競合緩和法（そのアーキテクチャについて）”情報処理学会第４５回全国大会講演論文集 vol.6,pp.89-90,1992

［２］　金崎　他　：“資源共有型並列計算機“砂丘”のアクセス競合緩和法（その効果に関する検証）”情報処理学会第４５回全国大会講演論文集 vol.6,PP.91-92,1992

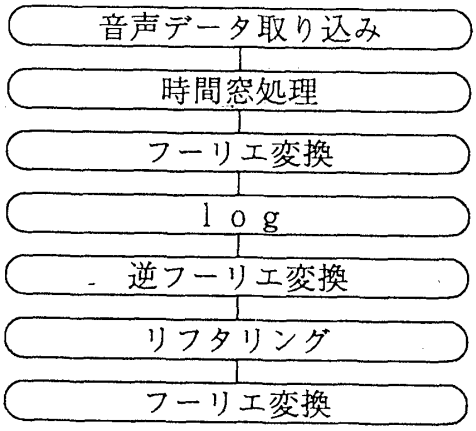

図１　ケプストラム分析の流れ図

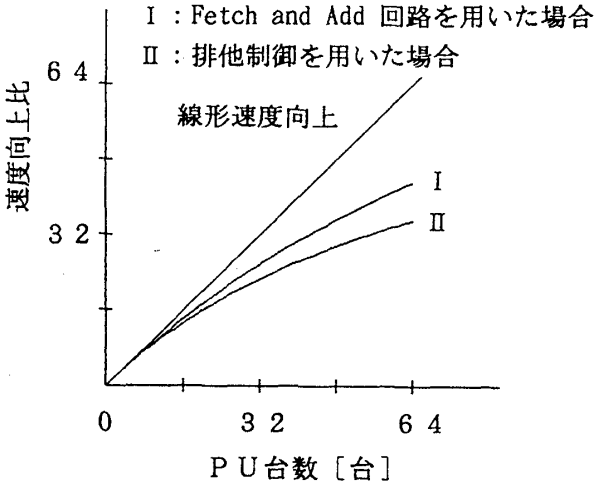

図２　通信時間の違う場合の速度向上比