計算機アーキテクチャ　111−10
（１９９５．３．１０）

主記憶共有型ベクトル並列スーパコンピュータ向け ＬＵ分解アルゴリズムの高速化手法

田中輝雄，玉置由子，伊藤昌尚，榊原忠幸，山本有作

(株)日立製作所　中央研究所

{ ttanaka, tamaki, m-itoh, sakakiba, yamamoto }@crl.hitachi.co.jp

要旨

主記憶共有型ベクトル並列スーパコンピュータにおけるＬＵ分解アルゴリズムによる密行列で表現される連立一次方程式の解法プログラムの高速化手法を検討した．ベクトル処理の高速化として多重ループ構造のループ展開手法を，また，並列処理の高速化として，Section 型並列処理機構，Post / Wait ライブラリを用いた並列制御処理時間の短縮および並列化適用範囲の拡大を行なった．その結果，主記憶共有型ベクトル並列スーパコンピュータＳ－３８００を用いて，比較的小規模な１０００元の行列サイズにおいても，シングルプロセサで６.５４GFLOPS，４台のマルチプロセサで２０.５GFLOPSの性能を実現した．

Techniques to raise the performance of the LU decomposition algorithm on a shared memory vector parallel supercomputer

Teruo TANAKA, Yoshiko TAMAKI, Masanao ITOH, Tadayuki SAKAKIBARA and Yuusaku YAMAMOTO

Central Research Laboratory, Hitachi Ltd.

Abstract

We present techniques to raise the performance of a shared memory vector parallel supercomputer when solving dense systems of linear equations by LU decomposition. For vector processing, we have studied loop unrolling for multiple loops. For parallel processing, we have studied the reduction of the overhead of parallel processing control and the expansion of parallelized code when using two parallel control facilities. As a result, when solving a relatively small system of equations (order 1000), we have obtained sustained performance of 6.54 GFLOPS with a single processor and 20.5 GFLOPS with four processors on Hitachi's S-3800 shared memory vector parallel supercomputer.

1．はじめに

ベクトル処理方式を基本とするスーパコンピュータが実用化されてから２０年以上が経ち，ハードウエア技術およびソフトウエア技術ともに格段の進歩を遂げた．ハードウエアはマシンサイクルの短縮および論理方式の改良によるシングルプロセサの性能向上とともに，並列処理方式の導入により台数効果による性能向上を実現している．特に，主記憶共有型のベクトル並列スーパコンピュータは，ＣＲＡＹのＸ－ＭＰ，Ｙ－ＭＰ，Ｃ９０，富士通のＶＰ２４００／４０，ＮＥＣのＳＸ－３，ＳＸ－４そして日立のＳ－３８００と各メーカの商用スーパコンピュータの主流となっている．

一方，ソフトウエアは，コンパイラの技術として従来の自動ベクトル化技術の強化とともに新たに自動並列化技術を開発した．特に，主記憶共有型の自動並列化技術は，分散記憶型のようにデータの分割配置を必要とせず，手続きの分割のみで並列化を可能とするために，自動ベクトル化の延長上に技術開発を進めることができた．しかし，まだソフトウエアがハードウエアの持つ能力を限界まで使い切っているとはいいがたい．このため，スーパコンピュータのシステム性能を十分引き出すためには，プログラムをスーパコンピュータ向きに書きなおす，いわゆるリコーディングがまだまだ必要な状況にあり，そのための高度利用技術を高めていく必要がある．

スーパコンピュータの性能評価の観点から見ると，ユーザ側からは，自分の興味のあるベンチマークプログラムを実行することにより，そのスーパコンピュータシステムの優劣を判断する．ベンチマークプログラムとしては，A(I) = B(I) + C(I) のような基本演算性能，種々の主記憶アクセスパタンによる主記憶アクセス性能，リバモアカーネルのような基本ループから実際の応用プログラムまでさまざまなレベルから評価を行ない，システムの優劣をトータルに判断する．

一方，システム開発者側からの性能評価では，そのハードウエアあるいはソフトウエアの長所／短所を知りぬいた上で，そのシステムを最大性能を引き出すことを行ない，システムの限界性能を追及し，その中から，コンパイラやライブラリなどのソフトウエアへの改良の指針や次期システムの研究開発への方針を設定する．

以下では，後者の立場から評価を行ない、ベンチマークプログラムとしてＬＵ分解アルゴリズムによる連立一次方程式の求解演算を用いる．このＬＵ分解アルゴリズムのベクトル化，並列化(特に分散記憶型)については，いままでも研究開発が進められている[8][9][10]．ここでは，さらに，主記憶共有型スーパコンピュータの特性（特にハードウエア）を十分に引き出すための手法について報告する．主記憶共有型スーパコンピュータの例として，日立のスーパコンピュータＳ－３８００[1]を用いる．

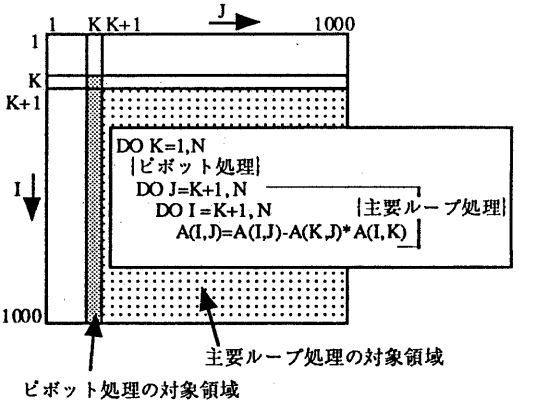

図１　対象行列 A(I,J) の構造（展開前）

2．ＬＵ分解アルゴリズムの概要とスーパコンピュータS-３８００の特徴

2．1　ＬＵ分解アルゴリズムの概要

ＬＵ分解アルゴリズムは，密行列の連立一次方程式の求解演算手法である．以下では，対象とする密行列のサイズを１０００元とする．ＬＵ分解アルゴリズムを選択した理由を示す．

（１）行列計算は数値シミュレーションの基本演算であり，その高度利用技術は他のシミュレーションプログラムへ広く適用可能であること．

（２）スーパコンピュータのベンチマークプログラムとしては問題規模が比較的小規模（必要な主記憶容量が８ＭＢ）であり，ハードウエアやソフトウエアの詳細特性を把握できること．

（３）特に１０００元のＬＵ分解アルゴリズムは，一般にLINPACK[注1] ベンチマーク[6]として，広く使用されていること．

プログラム主要部とそれに対応する行列Ａ(Ｉ，Ｊ)の構造を図１に示す．この基本的な構造はピボット

[注1] LINPACK(LINear equations PACKage)は，米国Tennessee大学のJ.J.Dongarra教授を中心に開発された線形計算用ライブラリで，そのうち，特にGaussの消去法に基づく連立一次方程式の求解ルーチンがLINPACKベンチマークとして，スーパコンピュータの実効性能を計る"ものさし"として広く利用されている．

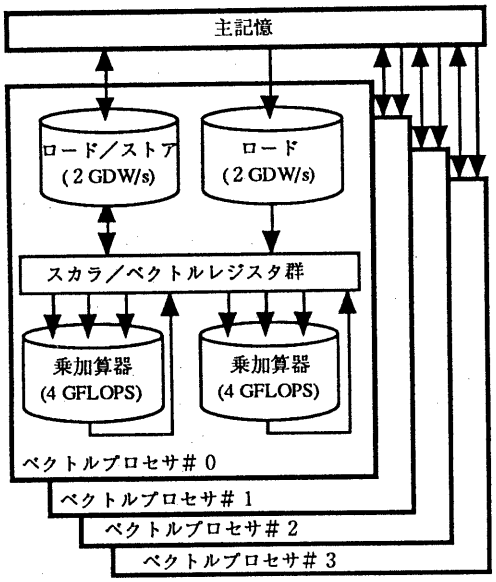

図2　S-3800の演算機構およびロードストア機構

処理[注2]と密な2重ループを含む3重ループであり，ループ内の演算処理は積和計算となる[注3]．演算量の大部分はこの密な2重ループ処理であり，以下ではこの処理を主要ループと呼ぶ．また，ベクトル化の対象であるIに関する最内側ループのループ長は反復回数ごとに行列サイズNから1まで順次短く変化する．

2．2　スーパコンピュータS－3800の特徴

評価の対象とする主記憶共有型ベクトル並列スーパコンピュータとして，スーパコンピュータS－3800を例題とする．S－3800は最大4台のベクトルプロセサが主記憶を共有する主記憶共有型のベクトル並列スーパコンピュータである．まずはS－3800のハードウエアの特徴をその前機種であるスーパコンピュータS－820と対比しながら概略説明を行なう．詳細なS－3800の特徴については，文献[2][3][4][5]を参照されたい．

2．2．1　演算能力について

図2にS－3800の概略構成を示す．主記憶を

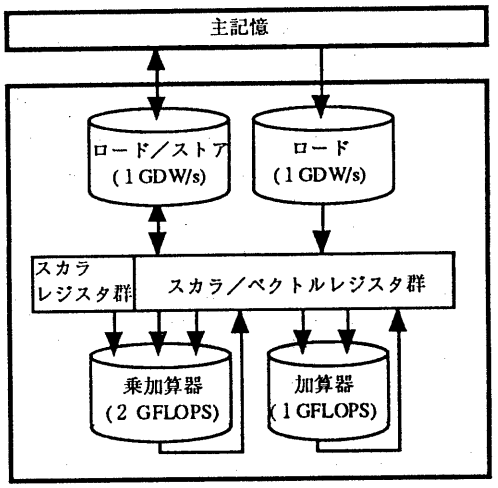

図3　S-820の演算機構およびロードストア機構

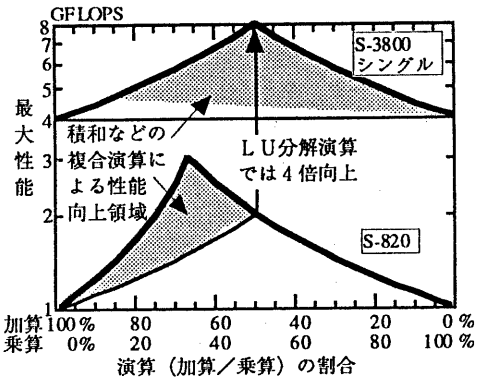

図4　演算の割合による演算最大性能の変化

共有する最大4台のベクトルプロセサから構成され，各ベクトルプロセサは2組の乗加算器を持つ．それぞれの乗加算器は最大2 GFLOPS[注4]の能力を持つ加算器と乗算器からなり，複合命令である乗加算命令を実行することにより，最大4 GFLOPSの演算性能を有する．したがって，ベクトルプロセサあたりの最大演算性能は8 GFLOPSとなる．さらに4台のベクトルプロセサによる並列処理によりシステムとしての最大演算性能は32 GFLOPSとなる．

一方，図3にS－820の概略構成を示す．S－820は単一のベクトルプロセサから構成され，乗加算器と加算器を1組づつ持つ．S－820の基本マシンサイクルはS－3800の1／2である．そのために乗加算器の最大演算性能は2 GFLOPSとな

注2　ピボット処理は解の安定性を保持するために行なう処理であり，配列Aの第K列内の最大値を検索し，その最大値を含む行とA(K,K)を含む行との交換を行なう[7].

注3　LINPACKベンチマーク(1000元)の定義では，アルゴリズム自体の改造も許されているが，本報告では，アルゴリズム自体は固定して検討する．

注4　GFLOPS：Giga Floating Operations Per Second．1秒間に10億回の浮動小数点演算を実行可能な能力．

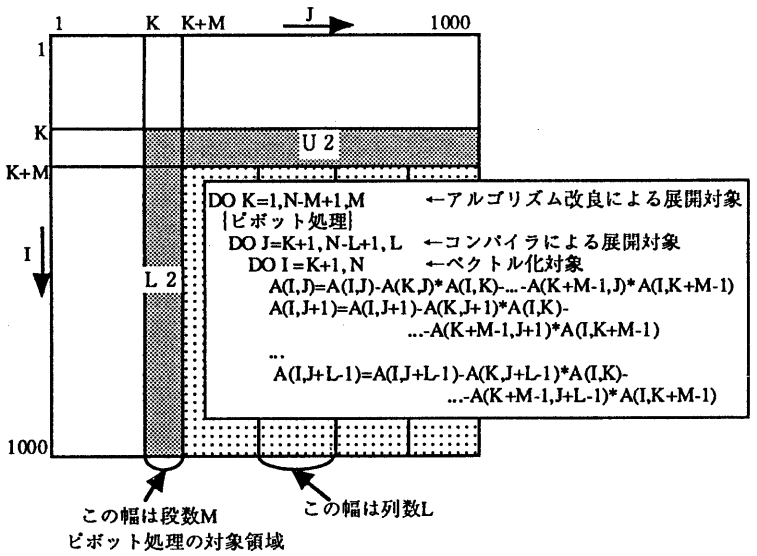

図5　対象行列 A(I,J) の構造（M段同時L列展開）

り，加算器は1 GFLOPSである．したがって，S－820の最大演算性能は3 GFLOPSとなる．

演算中の加算と乗算の割合と最大演算性能の関係を図4に示す．S－820では，乗算と加算の比が1対1で最大演算性能が2 GFLOPS，加算と乗算を1つずつ含む複合命令[注5]と加算命令が1対1の割合ではじめて最大演算性能が3 GFLOPSとなる．これに対して，S－3800では，加算命令と乗算命令の割合には関係なく最大演算性能は4 GFLOPSが上限と一定である．さらに，複合命令を用いることにより，2つの乗加算器が有効に動作するときに最大演算性能は8 GFLOPSとなる．

図1に示すように，LU分解アルゴリズムの主要ループの演算は，

A(I,J)＝A(I,J)－A(K,J)＊A(I,K) [式1]

とV＝V＋(-S)＊Vの積和形式となり，乗加算器を有効に動作させることができる．ここで，Vはベクトルデータ，Sはスカラデータを示す．

上記の積和演算により乗算と加算の比が1対1になる．したがって，この演算に対してS－820の最大性能が2 GFLOPSであるのに対して，S－3800では1台のベクトルプロセサの理想的な最大性能は8 GFLOPSとなり，S－820と比較して演算器構成の改良とマシンサイクルの向上により理想的に4倍の高速化がはかられることになる．

さらに，LU分解アルゴリズムの主要ループの演算は，図1よりわかるようにアルゴリズム的に完全に並列化可能であり，4台マルチプロセサでの並列実行時には主要ループの理想的最大性能の上限は32 GFLOPSとなる．

2．2．2　主記憶のデータアクセス能力について

スーパコンピュータの実効性能を決める要因として，主記憶とのデータアクセス能力がある．図2に示すように，S－3800のベクトルプロセサあたりの主記憶とのロード／ストアパイプラインは2 GDW[注6]×2である．これは，図3に示すS－820と同一構成であり，絶対性能の比はマシンサイクル比と同じ2倍にとどまっている．そのため，S－3800においては，演算器を効率良く動作させるためには，ロード／ストア命令に対する複合(積和)演算命令の比を1：1以上とする必要がある．

3．ベクトル処理機構の高速化

3．1　主要ループのループ展開，多段同時消去

2．2節で説明したように，ベクトル処理機構の高速化では，主記憶へのアクセス処理を抑えることが性能向上のための最重要課題である．

LU分解演算の主要ループは，図1に示すように多重ループを形成しておりループ展開により主記憶へのロード／ストア回数を削減することができる．多重ループのループ展開の例を図5に示す．図5の3重ループ構造において，Iに関するDOループはベクトル化の対象であり展開の対象とはしない．外側のJおよびKに関する2つのDOループが展開の対象である．

（1）Jに関するDOループをL倍にループ展開（いわゆるアンローリング）することにより，右辺のベクトルデータA(I,K)，A(I,K+1)，...，A(I,K＋M－1)が共通化可能となり，主

[注5] S－820で適用できる複合命令はV＝S＊V±V，（ここで，Vはベクトルあるいはスカラデータ，Sはスカラデータ）のみであるのに対して，S－3800では，V＝V＊V±V，V＊（V±V）をサポートしており，複合命令の適用範囲を拡大している．

[注6] GDW／s：Giga Double Words / sec. 1秒間に10億回の倍精度データを転送する能力．

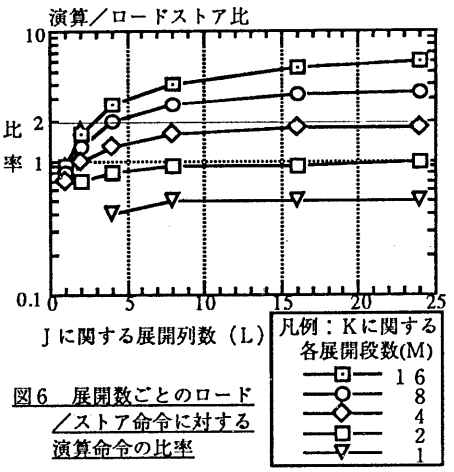

図6　展開数ごとのロード／ストア命令に対する演算命令の比率

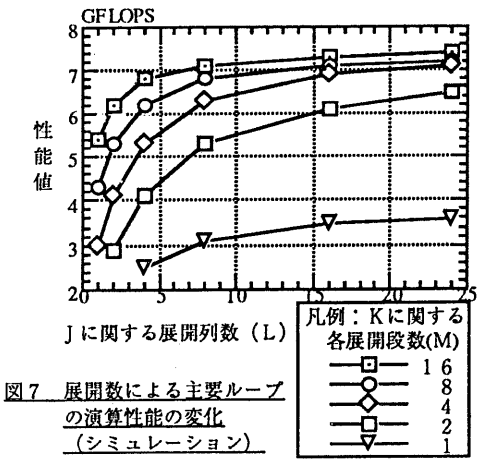

図7　展開数による主要ループの演算性能の変化（シミュレーション）

記憶からのロード処理を削減できる．

（2）外側のKに関するDOループをM段同時消去しM倍に展開することにより，左辺A(I,J)へ一度に複数のベクトルデータA(I,K), ..., A(I,K+M−1)を足しこむことになり，主記憶へのストア処理の回数を減らすことができる．

図6に，M段L列展開したときの主要ループ内のロード／ストア命令に対する演算命令（複合命令は2とカウント）の比を示す．2つの演算器を有効に動作させるためには，この比の値は最低2以上確保する必要がある．また，展開段数，展開列数を変化したときの主要ループの性能をシミュレーションにより求めた値を図7に示す．図7の評価値は，1000元の問題を解く時に必要なループ長(1000～1)をすべてシミュレーションし，全体の演算時間を求め，対応する全体の浮動小数点演算量から割った結果である．

展開数を増やすことにより，しばらくは性能が向上するが，この展開による性能向上にも限界がある．その理由は，次の2点である．

（1）Kに関するM段展開により，ピボット処理をM列まとめて行なう必要がある．そのために，本来主要ループで処理するはずの消去計算の一部(図5中のL2の領域)をピボット処理と交互に逐次的に行なう必要が生じ，主要ループの演算割合が低くなる．

（2）(中間)のベクトルデータを保持するベクトルレジスタが足りなくなるために，中間のベクトルデータの退避回復が必要となる．

1000元規模のLU分解演算に対しては，S−3800においては，8段同時消去16列展開あるいは10段同時消去10列展開程度が最適となる．

3．2　その他のベクトル処理機構の高速化手法

（1）キャッシュ転送の削減

S−3800では，スカラデータはキャッシュを介して取り込まれる．また，主記憶とキャッシュの間は，数から数10の連続したデータがまとめて転送される．したがって，スカラデータを複数必要とする場合は主記憶上の連続した領域から確保した方が効率がよい．LU分解法では，主要ループの演算（前述の[式1]）において，スカラデータA(K,J)を用いる．主要ループでは，Jに関するDOループはJ方向に変化するのでスカラデータは連続とはならない．そこで，A(K,J)[J＝K+M, N]のデータをワーク領域W(J)[J＝K+M, N]に事前に移しておき[式1]を

$$A(I,J)=A(I,J)-W(J)*A(I,K)$$

とすることにより，効率良くスカラデータを取り込むことができる．

（2）M段展開により派生した消去計算の高速化

このピボット処理により派生した消去計算処理(図4のL2領域の処理)も小規模ではあるがM段同時消去L列展開処理を適用することができる．それぞれ可能なレベルでの展開処理を行なう．

4．S−3800並列処理機構の高速化手法

4．1　並列化チューニングのための方針

前章でのベクトルプロセサ向けチューニングを施したプログラムを実行し，S−3800の1プロセサモデルで6.54GFLOPSの実効性能を実現した．

次に，主記憶共有型マルチプロセサに適用するために，Jに関するDOループに対しDO ALLタイプの並列化を施した．その結果，S−3800の4プロ

セサモデルを用いて１６.７GFLOPSの性能を得た．しかしこれは台数効果として，４台で２.６倍と効率が悪い．そこでプログラムを分析し，演算時間の内訳を調べた．その結果を表１に示す．

表１から，並列化の対象とした主要ループはじゅうぶんベクトル並列化の効果が得られていることがわかる．一方，浮動小数点演算が高々１.３％しかないその他のループの実行時間が全体の３４％を占めている．その他のループには，ピボット処理およびＫに関するDOループの展開により主要ループから分離したＬ２領域（図５）の消去計算処理がある．この処理はＬ２領域の列ごとのピボット処理と同時に行なう必要があり，まとめて並列化はできない．このＬ２領域のピボット処理および消去計算処理を処理Ｐとする．また，同様に図５のＵ２領域の消去計算処理を処理Ｕ，Ｋ＋Ｍ行からＮ行，Ｋ＋Ｍ列からＮ列の消去計算処理を処理Ａと呼ぶことにする．これらの処理の位置関係を図５を簡略化した補助図８ａに示す．また，図８にはＪに関するDOループに対して，DO ALLタイプの並列化を施し，処理Ａのみを並列化の対象としたときの処理手順のフローチャートを示す．図８では，簡単のためにプロセサ数は２としている．

図８の枠の中は，プログラムの概略であり，

```
*POPTION PARALLEL
```

は，Ｊに関するDOループに対して，DO ALL型の並列化の指示を与えている．

以下では，並列化の対象外であった処理Ｐと処理Ｕを並列化対象処理とする方法を示す．

４．２　高速化手法

ベクトルプロセサ２台の並列処理環境での例を用いながら検討を進める．

４．２．１　SECTION型並列処理の利用

並列化適用範囲の拡大を容易にするために，並列制御をDO ALL型ではなく，SECTION型に置き換える．また，並列処理のコストを削減するために[注7]，FORTRANライブラリにあるPOST/WAIT機能[注8]を利用する．

表１　S-3800４プロセサモデルでの演算時間内訳

	浮動小数点演算比率	実効性能	実行時間
	MFLOP's	GFLOPS	ms
主要ループ	659 (98.5%)	27	24.5 (61%)
その他のループ(含ピボット処理)	9 (1.3%)	0.7	13.4 (34%)
求解部	1 (0.2%)	0.5	2.0 (5%)
全体	669 (100%)	16.8	39.9 (100%)

SECTION型並列の例を図９に示す．ここでは２つのサブルーチンLPK1およびLPK2を用意し，それぞれに並列化後のＩ，Ｊ，Ｋの３重ループの処理を割り付ける．処理の分割は図８と同じであり，LPK1には処理Ｐ，Ｕ，Ａ１が割り当てられ，LPK2には処理Ａ２が割り付けられる．SECTION文によるLPK1とLPK2の並列化は最初の１回のみ行なわれ，DOループ処理は各サブルーチンごとに独自に行なう．各ループごとでのLPK1とLPK2の間の順序制御はPOST/WAIT機能を利用する．このように，SECTION型の並列処理により異なる処理の並列化が可能となる．

なお，頻繁に使用されるPOST/WAIT機能のコストをさらに削減するために，POST/WAITをプログラム上の一般変数で実現することもできる．共有変数を用いてカウンタセマフォとして実現する．

４．２．２　領域Ｕの並列化

POST/WAITによる低コストの並列制御の実現により，並列化対象の拡大を行なう．まずは，補助図１０ａに示すＵの領域の並列化をはかる．領域Ｕは領域Ａと同様に列ごとに並列性を有しており，並列化が可能である．領域Ａと領域Ｕの並列化のための分割の位置を合わせることにより，効率良く領域Ｕを並列化することができる．このときのフローチャートを図１０に示す．

４．２．３　ピボット処理と主要ループの並列実行

ピボット処理を含む領域Ｐの並列化を検討する．領域Ｐ自体は，処理量が少なく並列化粒度が小さくなるために並列化はあまり得策ではない．そこで領域Ｐを領域ＡやＵと並列に実行することを試みる．Ｋ回目の領域ＡおよびＵの処理Ａ(K)，Ｕ(K)および次の反復Ｋ＋Ｍ回目の領域Ｐの処理Ｐ(K+M)の関係を調べる．処理Ｐ(K+M)は処理Ａ(K)のうちＫ＋Ｍ列からＫ＋Ｍ×２－１列までの処理が終了すれば実行することができる（補助図１１ａの点線の領域）．

[注7] 1000元のＬＵ分解アルゴリズムを４台マルチプロセサで並列実行する場合，実行時間は30～40msecである．この中で150～200回程度の並列処理起動を行なうため，１回あたりの粒度は100μｓ～200μｓ程度となる．DO ALL型の１回あたりの並列制御処理時間は数～10数μｓであり，数％と無視できる範囲では無くなる．POST/WAITを用いることにより並列制御処理時間を数分の１に短縮できる．

[注8] このPOST/WAIT機能は，ＯＳのPOST/WAITマクロではなく，FORTRANのライブラリの一部である．したがって，コストは比較的軽い．

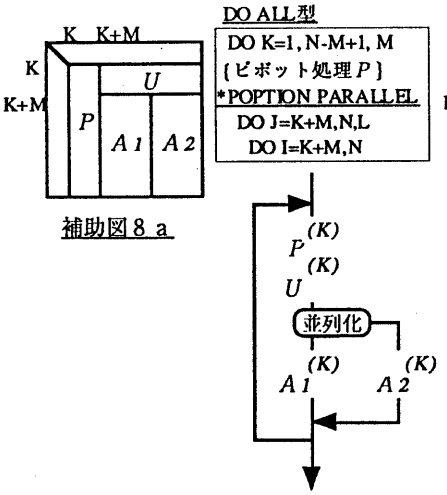

図８　DO ALL型並列処理による領域Ａの並列化

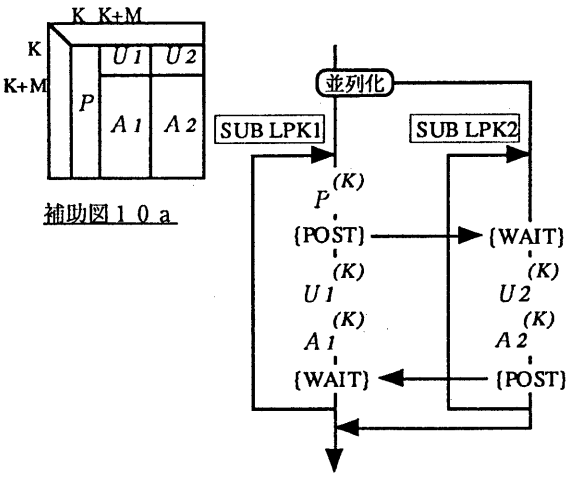

図１０　領域Ｕの並列化

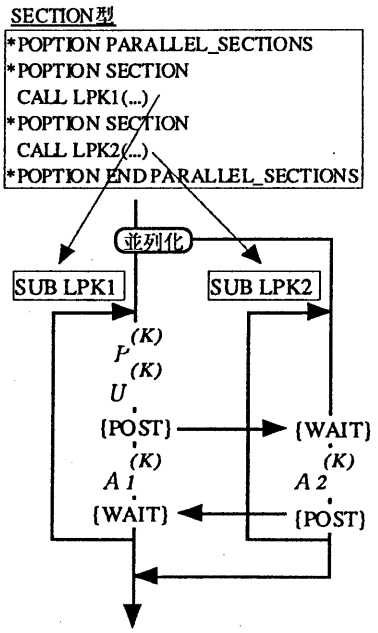

図９　SECTION型並列処理

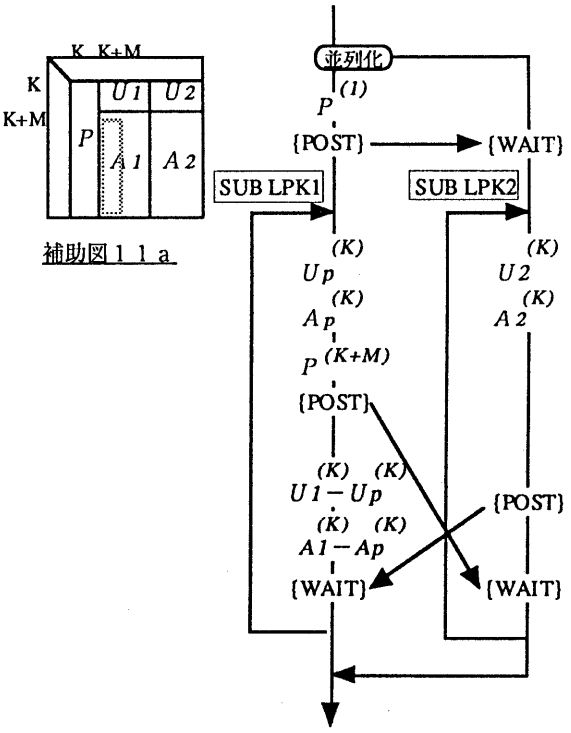

図１１　ピボット処理Ｐと更新処理Ｕ，Ａとの並列化

そこで，U1(K)，A1(K)およびP(K+1)を同一の反復計算中に組込む，もう一方のプロセサでは，U2(K)およびA2(K)の処理を実行する．ここで，バランスをとるために，処理量はU1(K)＋A1(K)＋P(K+M)とU2(K)＋A2(K)が均等になるように負荷分散を行なう．ただし，このとき処理A1(K)には最低限処理P(K+M)に必要なK＋M列からK＋M×2－1列までの処理を含める必要がある．さらに，U1(K)，A1(K)をP(K+M)で対象とする領域の処理Up(K)とU1(K)－Up(K)，Ap(K)とA1(K)－Ap(K)に分ける．

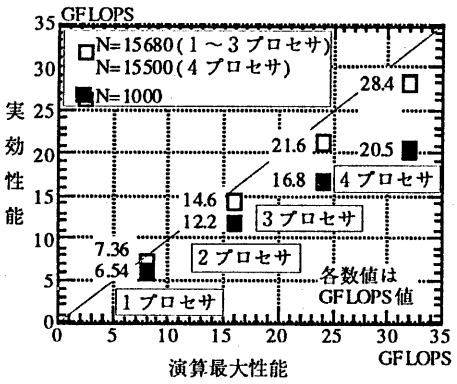

図１２　Ｓ－３８００の各モデルでの実測結果

P(K+M)は，Up(K)およびAp(K)終了後，すぐに実行することが可能となる．それに伴い，POST処理をできるだけ早く，WAIT処理をできるだけ遅く，実行することにより，WAIT処理での待ち時間を少なくすることができる（図１１参照）．

以上の並列化効率の改善により，Ｓ－３８００の４プロセサモデルを用いた時の性能値として２０.５GFLOPSを得た．これは，台数効果として，３.１倍にあたり，４台のベクトルプロセサで３台以上の実効性能を達成することができたことになる．

なお，ここでの並列化適応範囲の拡大の方法は，ＲＩＳＣ，スーパスカラなどの高性能マイクロプロセサを要素プロセサとする高／超並列計算機のような分散記憶型計算機への適用も可能である．

５．結論

主記憶共有型ベクトル並列計算機の特長を意識したＬＵ分解アルゴリズムのチューニングを行ない，Ｓ－３８００の各モデルに適用した結果を図１２に示す．図１２には，比較のために問題規模を非常に大きくしたときの結果も合わせて示す．1000元という比較的小さい問題規模にも関わらず，４プロセサモデルにおいて，演算最大性能の６４％，問題規模が非常に大きいときの性能の７２％を実現した．

今回のチューニングでは，ハードウエアパラメタはＳ－３８００を用いたが，基本的な考え方は他のベクトル並列計算機にも適用することができる．

さらに，ハードウエアの特長を引き出すチューニング手法を検討するとともに，この技術をコンパイラあるいはライブラリに適用できるようにし，一般のユーザでもより高い性能をより簡単に引き出せるようなシステム研究開発を行なっていきたい．

謝　辞

本研究を進めるにあたり有益な助言および協力をいただきました日立製作所汎用コンピュータ事業部石井幸一主任技師，深川正一技師，同ソフトウエア開発本部　後保範主任技師，浜口信行技師に感謝いたします．

参考文献

[1] Ishii, K., H. Abe, S. Kawabe and M. Hirai：An Overview of the HITACHI S-3800 Series Supercomputer, Proc. of Supercomputing'92 (1992).

[2] Kitai, K., T. Isobe, Y. Tanaka, Y. Tamaki, M. Fukagawa, T. Tanaka and Y. Inagami：Parallel Processing Architecture for the Hitachi S-3800 Shared-Memory Vector Multiprocessor, ICS93 (1993.7).

[3] Sakakibara, T., Kitai, K., T. Isobe, S. Yazawa, T. Tanaka, Y. Inagami and Y. Tamaki：Scalable Parallel Memory Architecture with s Skew Scheme, ICS93, pp.157-166 (1993.7).

[4] Sakakibara, T., Kitai, K., T. Isobe, S. Yazawa, T. Tanaka, Y. Tamaki and Y. Inagami：An Interprocessor Memory Access Arbitrating Scheme for the S-3800 Vector Supercomputer, ISPAN (1994.12).

[5] 浜口信行：主記憶共有型スーパコンピュータの性能評価，情報処理学会研究報告94-HPC-51, pp.41-48 (1994).

[6] Dongarra JJ.：Performance of Various Computers Using Standard Linear Equations Software, Comp. Science Dept., Univ. of Tennessee (1995.1).

[7] Forsythe G.E., Moler C.B.：Computer Solution of Linear Algebraic Systems, Prentice-Hall, Inc. (1967). 渋谷，田辺訳：線形計算の基礎－連立一次方程式のプログラミング－，培風館 (1969).

[8] 村田，小国，唐木：スーパコンピュータ，培風館 (1986).

[9] 高橋：分散メモリ型計算機ＰＡＸ上におけるガウスの消去法，JSPP91, pp.333-340 (1991.5).

[10] 中西，三上：ブロッキングＬＵ分解法のＶＰ２６００シリーズ向けチューニングについて，情報処理学会第４２回全国大会予稿集，vol.1, pp.71-72 (1991.3).