JAERI-Data/Code
96-006

並列計算機における等方乱流数値シミュレーション・コードの並列化

1996年3月

松山雄次

日本原子力研究所
Japan Atomic Energy Research Institute

本レポートは，日本原子力研究所が不定期に公刊している研究報告書です。
入手の問合わせは，日本原子力研究所技術情報部情報資料課（〒319-11 茨城県那珂郡東海村）あて，お申し越しください。なお，このほかに財団法人原子力弘済会資料センター（〒319-11 茨城県那珂郡東海村日本原子力研究所内）で複写による実費頒布をおこなっております。

This report is issued irregularly.
Inquiries about availability of the reports should be addressed to Information Division, Department of Technical Information, Japan Atomic Energy Research Institute, Tokai-mura, Naka-gun, Ibaraki-ken 319-11, Japan.

編集兼発行　日本原子力研究所
印　　刷　いばらき印刷㈱

JAERI-Data/Code 96-006

並列計算機における等方乱流数値シミュレーション・コードの並列化

日本原子力研究所計算科学技術推進センター
松山　雄次

(1996年2月7日受理)

ベクトル計算機用に最適化された等方乱流数値シミュレーション・コード（Trans5）を題材にして、並列計算機Paragon XP/S、ベクトル並列計算機VPP500、および日本原子力研究所で開発されたベクトル並列計算機Monte-4での並列化解析ツール、並列化手法、並列最適化環境について検討した。アーキテクチャの異なるこれらの並列計算機における、高速フーリエ変換の並列最適化の効果についても報告する。

日本原子力研究所：〒153　東京都目黒区中目黒2－2－54

JAERI-Data/Code 96-006

Parallelization of the Numerical Simulation Code for Homogeneous Turbulence on Parallel Computers

Yuji MATSUYAMA

Center for Promotion of Computational Science and Engineering
Japan Atomic Energy Research Institute
Nakameguro, Meguro-ku, Tokyo

(Received February 7, 1996)

With use of the numerical simulation code for homogeneous turbulence (Trans5), which was optimized for a vector computer, parallelizing tools, techniques, and environments for parallel systems such as Paragon XP/S, vector system VPP500, and the other vector system Monte-4, which was developed by JAERI, are investigated. Parallel optimization effect of FFT on these different architectures is also reported.

Keywords: Paragon XP/S, VPP500, Monte-4, Parallelization, FFT, Analyzer

目　　次

Contents

1．はじめに

1台のCPU（Central Processing Unit）の高速化が限界に近づきつつある今日、より高速な演算性能を得るには、複数のCPUで並列処理を行うのが、現在一般に最も有効とされている方法である。また、高速CPUの開発には技術的ブレイク・スルーのため莫大な経費が必要となり、必然的に高価な計算機とならざるを得ない。並列計算機においてSPP（Scalable Parallel Processing）は低価格計算機である程度の性能を得る方法であり、MPP（Massively Parallel Processing）は今まで計算解析不可能だった超大規模計算を実現する方法である。しかしながら、SPPで有効であった並列化コードも、並列度を高くしたMPPでは一般的に極端に並列化効率が落ち、PE数に比例した十分な加速率を得ることができない。

共有主記憶型のMPPシステムでは複数PE（Processing Element）からの主記憶同時アクセスが発生するため、バンクコンフリクトの影響を極端に受ける。現時点では、このバンクコンフリクトを回避できるだけのバンク数を、発熱量、経費、高速演算に対応した計算機の物理的スケール等から装備することができない。このため、MPPシステムでは分散主記憶型のシステム構造を採用するのが一般的である。スーパーコンピュータに代表されるベクトル型マルチプロセッサ・システムではこの問題のため、MPPレベルのPE数（PE数が3桁以上）を実現することは不可能である。

MPPシステムにはPEにスカラ・プロセッサを採用した計算機とベクトル・プロセッサを採用した計算機がある。前述したとおり、一般的並列化コードではPE数に比例したリニアな加速率を得ることができないため、SPPでは1台のPEの性能は高ければ高いほどよい。しかしながら、MPPでは本来、並列度が極端に上がるアルゴリズムの超大規模計算でしかMPPシステムの特性を生かすことができず、その意味ではどのタイプのプロセッサを採用したら良いかは一概に言えない。

日本原子力研究所には原子力分野における大規模な計算需要に対応するため、PEに256台のスカラ・プロセッサを採用した分散主記憶型MPPシステムParagon XP/S、PEに42台のベクトル・プロセッサを採用した分散主記憶型MPPシステムVPP500（ただし42台の構成ではMPPシステムの範疇に属さない）、CPUに4台のベクトル・プロセッサを採用した共有主記憶型マルチプロセッサ・システムMonte-4[1)]がある。これらのアーキテクチャは全く異なり、PE間ネットワークを見てもParagon XP/Sが2次元メッシュ、VPP500がクロスバ、Monte-4はベクトル型マルチプロセッサ・システムでCPUは密結合されている。また、ソフト的な並列化の手段も、各々の計算機で全く異なっている。

本報告書では、Paragon XP/S、VPP500、Monte-4について、各機種における並列化あるいは最適化のためのツール、Fortranパラダイムとネットワーク通信の比較等を踏まえたシステムの使用環境、FFT（高速フーリエ変換）ルーチンの並列化手法と並列最適化プログラム開発工数等、および自動並列化の機能とその結果について記述する。

2．Trans5

ナビエ・ストークス方程式を渦度を用いて表現し、速度と渦度を３次元フーリエ展開すると次式が得られる。

$$\frac{d}{dt}\tilde{\omega}_j(\boldsymbol{k}) = \varepsilon_{jkl} k_k k_m \tilde{u_l u_m}(\boldsymbol{k}) - \nu k^2 \tilde{\omega}_j(\boldsymbol{k})$$

$$\sum_j k_j \tilde{u}_j(\boldsymbol{k}) = 0$$

$$\tilde{\omega}_j(\boldsymbol{k}) = -\varepsilon_{jkl} k_k \tilde{u}_l(\boldsymbol{k})$$

ただし、$\boldsymbol{\omega} = (\omega_1, \omega_2, \omega_3)$, $\boldsymbol{k} = (k_1, k_2, k_3)$, $j = 1,2,3$

ε_{jkl}は交代テンソル、$u_j(\boldsymbol{x})$は速度、$\omega_j(\boldsymbol{x})$は渦度、

ここで、$\tilde{u}_j(\boldsymbol{k})$、$\tilde{\omega}_j(\boldsymbol{k})$、$\tilde{u_l u_m}(\boldsymbol{k})$は、$u_j(\boldsymbol{x})$、$\omega_j(\boldsymbol{x})$、$u_l(\boldsymbol{x})u_m(\boldsymbol{x})$のフーリエ係数であり、以下のように定義される。

$$u(x) = i \sum_{k_1=-\frac{N}{2}}^{\frac{N}{2}-1} \sum_{k_2=-\frac{N}{2}}^{\frac{N}{2}-1} \sum_{k_3=-\frac{N}{2}}^{\frac{N}{2}-1} \tilde{u}(\boldsymbol{k}) \exp[i\boldsymbol{k} \cdot \boldsymbol{x}]$$

$$\omega(x) = \sum_{k_1=-\frac{N}{2}}^{\frac{N}{2}-1} \sum_{k_2=-\frac{N}{2}}^{\frac{N}{2}-1} \sum_{k_3=-\frac{N}{2}}^{\frac{N}{2}-1} \tilde{\omega}(\boldsymbol{k}) \exp[i\boldsymbol{k} \cdot \boldsymbol{x}]$$

$$u_l(x)u_m(x) = \sum_{k_1=-\frac{N}{2}}^{\frac{N}{2}-1} \sum_{k_2=-\frac{N}{2}}^{\frac{N}{2}-1} \sum_{k_3=-\frac{N}{2}}^{\frac{N}{2}-1} \tilde{u_l u_m}(\boldsymbol{k}) \exp[i\boldsymbol{k} \cdot \boldsymbol{x}]$$

Trans5[2)]は、擬スペクトル法を用いた等方乱流数値シミュレーション・コードである。時間積分にはルンゲ - クッタ法を用い、各ステージで非線形項の計算を行っている。また、３次元FFTはX、Y、Z方向に１次元FFTを行って３次元としている。

Trans5にはフーリエ空間の速度を物理空間の速度に逆フーリエ変換するルーチン（vdft3b）と、畳み込み積分の計算後、またフーリエ空間に戻すルーチン（vdft3f）があり、これらのルーチンで、計算機により負荷は異なるが、全体の90%近くの計算時間が消費され、コード自体は酷似している。後述する並列化手法の参考、あるいは並列化コードとの比較のために、Fig. 2.1 にベクトル版のオリジナル・ルーチンvdft3bのみを示す。

Trans5を実行するのに必要となる主記憶容量は、Fig. 2.2 に示すTrans5のメイン・ルーチンでの変数宣言より、32ビット・マシンすなわち一語が４バイトのシステムでは、（$7\times np\times n^2+38\times nk^3+nk+2\times n\times nl1+162$）×4byteとなる。これは、N=32で約2.1MB、N=1024では約73GBに相当する。本コードではフーリエ・モード数Nには32を与えた。

プログラムは前処理、本体（Trans5）、後処理の３本に別れており、各々のサブルーチンの本数と総ステップ数は、前処理が14本で973ステップ、本体が16本で1174ステップ、後処理が12本で890ステップである。ここでは、計算負荷の比較的高い本体の並列化を行なった。

```
      subroutine  vdft3b( n, np, n2, nl1, fr, fi, cr, ci, cn, sn, ibr )

      dimension  fr(np, n, n),   fi(np, n, n),   cr(np, n, n),   ci(np, n, n)
      dimension  cn(n2*nl1),   sn(n2*nl1),   ibr(n*nl1)
      common  /cmvdft/ leven, oncube

c -----( Fourier transforms in the x-direction )----------------------

*vdir novector
      do  100  istage = 1, nl1, 2

        ibase  =  n2*(istage-1)
*vocl loop, novrec
*vdir nodep
        do  110  k = 1, n
          do  110  i = 1, n2
            i2  =  2*i
            i1  =  i2 - 1
            do  110  j = 1, n
              cr(i1, j, k)  =  fr(i, j, k)  +  fr(i+n2, j, k)
              ci(i1, j, k)  =  fi(i, j, k)  +  fi(i+n2, j, k)
              temp1         =  fr(i, j, k)  -  fr(i+n2, j, k)
              temp2         =  fi(i, j, k)  -  fi(i+n2, j, k)
              cr(i2, j, k)  =  temp1*cn(i+ibase)  -  temp2*sn(i+ibase)
              ci(i2, j, k)  =  temp2*cn(i+ibase)  +  temp1*sn(i+ibase)
 110    continue

        if( (leven. eq. 0) . and.  (istage. eq. nl1) )  go to 100

        ibase  =  n2*istage
*vocl loop, novrec
*vdir nodep
        do  120  k = 1, n
          do  120  i = 1, n2
            i2  =  2*i
            i1  =  i2 - 1
            do  120  j = 1, n
              fr(i1, j, k)  =  cr(i, j, k)  +  cr(i+n2, j, k)
              fi(i1, j, k)  =  ci(i, j, k)  +  ci(i+n2, j, k)
              temp1         =  cr(i, j, k)  -  cr(i+n2, j, k)
              temp2         =  ci(i, j, k)  -  ci(i+n2, j, k)
              fr(i2, j, k)  =  temp1*cn(i+ibase)  -  temp2*sn(i+ibase)
              fi(i2, j, k)  =  temp2*cn(i+ibase)  +  temp1*sn(i+ibase)
 120    continue

 100  continue

      if( leven . ne. 0 )  then
        do  200  k = 1, n
          do  200  j = 1, n
            do  200  i = 1, np
              cr(i, j, k)  =  fr(i, j, k)
              ci(i, j, k)  =  fi(i, j, k)
 200    continue
      endif

*vocl loop, novrec
*vdir nodep
      do  300  k = 1, n
        do  300  i = 1, n2
          i2  =  2*i
          i1  =  i2 - 1
          do  300  j = 1, n
            fr(ibr(i1), j, k)  =  cr(i, j, k)  +  cr(i+n2, j, k)
            fi(ibr(i1), j, k)  =  ci(i, j, k)  +  ci(i+n2, j, k)
            fr(ibr(i2), j, k)  =  cr(i, j, k)  -  cr(i+n2, j, k)
            fi(ibr(i2), j, k)  =  ci(i, j, k)  -  ci(i+n2, j, k)
 300  continue
```

Fig. 2.1 vdft3b original code (to be continued)

```
c -----( Fourier transforms in the y-direction )--------------------

*vdir novector
      do  400  istage = 1, nl1, 2

        jbase = n2*(istage-1)
*vocl loop,novrec
*vdir nodep
        do  410  k = 1, n
          do  410  j = 1, n2
            j2 = 2*j
            j1 = j2 - 1
            do  410  i = 1, n
              cr(i,j1,k) = fr(i,j,k) + fr(i,j+n2,k)
              ci(i,j1,k) = fi(i,j,k) + fi(i,j+n2,k)
              temp1      = fr(i,j,k) - fr(i,j+n2,k)
              temp2      = fi(i,j,k) - fi(i,j+n2,k)
              cr(i,j2,k) = temp1*cn(j+jbase) - temp2*sn(j+jbase)
              ci(i,j2,k) = temp2*cn(j+jbase) + temp1*sn(j+jbase)
 410    continue

        if( (leven.eq.0) .and. (istage.eq.nl1) )  go to 400

        jbase = n2*istage
*vocl loop,novrec
*vdir nodep
        do  420  k = 1, n
          do  420  j = 1, n2
            j2 = 2*j
            j1 = j2 - 1
            do  420  i = 1, n
              fr(i,j1,k) = cr(i,j,k) + cr(i,j+n2,k)
              fi(i,j1,k) = ci(i,j,k) + ci(i,j+n2,k)
              temp1      = cr(i,j,k) - cr(i,j+n2,k)
              temp2      = ci(i,j,k) - ci(i,j+n2,k)
              fr(i,j2,k) = temp1*cn(j+jbase) - temp2*sn(j+jbase)
              fi(i,j2,k) = temp2*cn(j+jbase) + temp1*sn(j+jbase)
 420    continue

 400  continue

      if( leven .ne. 0 )  then
        do  500  k = 1, n
          do  500  j = 1, n
            do  500  i = 1, np
              cr(i,j,k) = fr(i,j,k)
              ci(i,j,k) = fi(i,j,k)
 500    continue
      endif

*vocl loop,novrec
*vdir nodep
      do  600  k = 1, n
        do  600  j = 1, n2
          j2 = 2*j
          j1 = j2 - 1
          do  600  i = 1, n
            fr(i,ibr(j1),k) = cr(i,j,k) + cr(i,j+n2,k)
            fi(i,ibr(j1),k) = ci(i,j,k) + ci(i,j+n2,k)
            fr(i,ibr(j2),k) = cr(i,j,k) - cr(i,j+n2,k)
            fi(i,ibr(j2),k) = ci(i,j,k) - ci(i,j+n2,k)
 600  continue
```

Fig. 2.1 vdft3b original code (to be continued)

```
c -----( Fourier transforms in the z-direction )---------------------

*vdir novector
      do  700  istage = 1, nl1, 2

        kbase =  n2*(istage-1)
*vocl loop,novrec
*vdir nodep
        do  710  k = 1, n2
          k2     = 2*k
          k1     = k2 - 1
          do  710  j = 1, n
            do  710  i = 1, n
              cr(i,j,k1)  =  fr(i,j,k)  +  fr(i,j,k+n2)
              ci(i,j,k1)  =  fi(i,j,k)  +  fi(i,j,k+n2)
              temp1       =  fr(i,j,k)  -  fr(i,j,k+n2)
              temp2       =  fi(i,j,k)  -  fi(i,j,k+n2)
              cr(i,j,k2)  =  temp1*cn(k+kbase)  -  temp2*sn(k+kbase)
              ci(i,j,k2)  =  temp2*cn(k+kbase)  +  temp1*sn(k+kbase)
 710    continue

        if( (leven.eq.0) .and. (istage.eq.nl1) )  go to 700

        kbase =  n2*istage
*vocl loop,novrec
*vdir nodep
        do  720  k = 1, n2
          k2     = 2*k
          k1     = k2 - 1
          do  720  j = 1, n
            do  720  i = 1, n
              fr(i,j,k1)  =  cr(i,j,k)  +  cr(i,j,k+n2)
              fi(i,j,k1)  =  ci(i,j,k)  +  ci(i,j,k+n2)
              temp1       =  cr(i,j,k)  -  cr(i,j,k+n2)
              temp2       =  ci(i,j,k)  -  ci(i,j,k+n2)
              fr(i,j,k2)  =  temp1*cn(k+kbase)  -  temp2*sn(k+kbase)
              fi(i,j,k2)  =  temp2*cn(k+kbase)  +  temp1*sn(k+kbase)
 720    continue

 700  continue

      if( leven .ne. 0 )  then
        do  800  k = 1, n
          do  800  j = 1, n
            do  800  i = 1, np
              cr(i,j,k)  =  fr(i,j,k)
              ci(i,j,k)  =  fi(i,j,k)
 800    continue
      endif

*vocl loop,novrec
*vdir nodep
      do  900  k = 1, n2
        k2  =  2*k
        k1  =  k2 - 1
        do  900  j = 1, n
          do  900  i = 1, n
            fr(i,j,ibr(k1))  =  cr(i,j,k)  +  cr(i,j,k+n2)
            fi(i,j,ibr(k1))  =  ci(i,j,k)  +  ci(i,j,k+n2)
            fr(i,j,ibr(k2))  =  cr(i,j,k)  -  cr(i,j,k+n2)
            fi(i,j,ibr(k2))  =  ci(i,j,k)  -  ci(i,j,k+n2)
 900  continue

c ---------------------------------------------------------------------

      return
      end
```

Fig. 2.1 vdft3b original code

```
dimension upr(np,n,n,3)
dimension wkr(np,n,n),    wki(np,n,n),    w2r(np,n,n),
&         w2i(np,n,n)

dimension ur(nk,nk,nk,3), ui(nk,nk,nk,3)
dimension wr(nk,nk,nk,3), wi(nk,nk,nk,3)
dimension tr(nk,nk,nk,3), ti(nk,nk,nk,3)
dimension wor(nk,nk,nk,3), woi(nk,nk,nk,3)

dimension w11r(nk,nk,nk), w12r(nk,nk,nk), w13r(nk,nk,nk)
dimension w22r(nk,nk,nk), w23r(nk,nk,nk), w33r(nk,nk,nk)
dimension w11i(nk,nk,nk), w12i(nk,nk,nk), w13i(nk,nk,nk)
dimension w22i(nk,nk,nk), w23i(nk,nk,nk), w33i(nk,nk,nk)

dimension rkk(nk,nk,nk), okk(nk,nk,nk), co(nk)
dimension cn(n2*nl1),    sn(n2*nl1),    ibr(n*nl1)

dimension forcer(3,3,3,3), forcei(3,3,3,3)
```

Fig. 2.2 Data declaration in the main routine

3．アナライザ

並列化において、各ルーチンまたは各ループのプログラム全体に対する実行負荷を把握することは極めて重要である。理想的状態における並列化の最大加速率はベクトル化のそれと同じくアムダールの式で表現される。

$$S = \frac{1}{\beta + \frac{1-\beta}{N}}$$

ただし、Sは加速率、Nは並列化したPE数、βは単体で実行する割合（$0 \leqq \beta \leqq 1$）

例えば、プログラムの95%を完璧に並列化した場合、オーバーヘッドを無視しても加速率は高々20倍であり、128PEにて並列化を試みるのはアムダールの式より明らかに適当ではない。よって、並列計算機には少なくとも実行負荷を容易に把握できる機能が必要である。さらに並列化できる部分すなわち並列化できる割合、最終的な並列化の効果まで推定できるシステムが完備されていれば理想である。また、コンパイラに自動並列化機能があれば、並列化を行う上で作業量が大幅に短縮される。現実には、並列化できる部分の割合は最適化手法に少なからず依存する場合が多いが、例えば90%以上の並列化を完了した後のさらなる並列化率の向上は、全体の加速率に影響を及ぼす率が大で、並列化できる割合を推定できるシステムは非常に有用である。また、並列化によるオーバーヘッド等を考慮し、並列化を試みるPE数から得られる加速率を予測できるシステムが有ればなお良い。

ここでは、今回使用した各種実験機器で利用できる並列化ツール、並列化支援プログラム解析ツールについて述べる。

Table 3.1 に各機種のアナライザ機能をまとめた。

3.1 Paragon XP/S

Paragon XP/Sではデバッガの機能にサンプラがあり、大体の所のサブルーチンの実行負荷が把握できる。正確に各ループまたは各ステップのプログラム全体に対する実行負荷を把握するためには、ユーザが時刻採取のためのファンクションdclock()をプログラムの随所に挿入して、各々のループ負荷を特定するかしなければならない。

超大規模計算プログラムを高並列で実行するつもりであれば、実行負荷如何に関わらず全てのループに対して並列化を試みるべきところであるが、効率的に並列最適化を行うという意味で、多角的な解析機能を備えたプログラム解析ツールが必要である。

今回、Paragon XP/Sをターゲットとした並列化を行うにあたっては、Paragon XP/Sのサンプラでは十分な解析が行えないため、Monte-4のアナライザを使用しプログラムの解析を行った。

Paragon XP/Sにはシステム全体でのPE稼働状況やPE間通信の状況を、X window systemsのX11に対応したワークステーション上で動的に監視できるシステムSPV（System Performance Visualization tool）[3]がある。SPVが持つ大まかな機能はParagon XP/S本体のフロント・パネル

でも確認可能であるが、ワークステーション上ではより詳細な情報を遠隔地で入手可能である。この機能を使用して、ある特定PEグループの演算状況や通信負荷に片寄りが発生していないか確認することができ、プログラム最適化の参考とすることができる。また、デッド・ロック状態等も容易に観測可能である。この機能は他の並列計算機には存在しないが、利用法に依ってはかなり有効である。

3.2 VPP500

VPP500にはafrtプログラム解析ツール（VP用）やparasamp最適化支援ツール、PSC（Parallel Serial Comparator）[4]デバッグ支援ツールが用意されている。チューナ等も用意されており、また、時間測定には実行時のtimexやソース・レベルでのgettod()等が利用できる。

1）parasamp

これらの中でも、サンプラparasampはqsub実行シェルに追加するだけで使用でき、使用法も簡単である。サンプラであるため正確なデータは採取できないが、サブルーチン単位の実行負荷、あるいはプロセッサ毎の実行負荷を測定でき、並列最適化を行うにあたり非常に有効である。parasampのリストには、サブルーチン単位の並列化率（Parallelization ratio）、加速率（Parallel to serial supped ratio）、負荷バランス（Load balance）、非同期転送待ち発生率（Asynchronous transfer ratio）、逐次総合情報（Synthesis information）が記載されている。

尚、parasampとgettod()を併用すると領域外参照という、原因とエラーメッセージの関連付けが困難なメッセージを出力してジョブがアボートする。元来、parasampで実行負荷を測定する場合、サンプリングのため負荷が発生する。したがって、gettod()で正確な時間測定を行うことはできないので、このような使用法は避けなければならない。

parasampでTrans5を実行した解析例をFig. 3.1 に示す。

2）PSC

PSCは逐次実行と並列実行の途中結果を比較報告するツールである。PSCを使用するには次の３段階の手続きを要する。まず最初にFortranプログラムよりcfrtpxコマンドで、デバッグ用の逐次実行ファイルと並列実行ファイル、および比較する変数名と比較タイミングを記述した比較指示ファイルの雛型を出力する。このとき出力された比較指示ファイルの雛型を必要に応じ編集するか、新に作成して比較指示ファイルを準備する。第２段階では逐次実行ファイルを実行して、比較指示ファイルに記述された内容に従いデータを採取して比較用情報ファイルとして保存する。最後に第３段階として並列実行ファイルを実行して、保存した逐次実行の比較用情報ファイルと比較しリストを作成する。

PSCを利用して、いくつかの問題点に遭遇したのでここに列挙する。

a) UXP/M用のVPPアナライザ使用手引書は162ページあるが、十分の一のページ数に抑えることが可能である。該当マニュアルの説明は極めて冗長かつ理解しづらい。

b) ３段階に分けたPSCの設計自体に問題があり、結果の確認までに、cfrtpxコマンド１回の実行と、比較指示ファイルの編集、第２、第３段階の実行シェルの作成、およびジョブのサブミット２回が必要となる。ここでは、逐次実行ファイル、並列実行ファイル、比較用情報フ

ファイルにユーザが介入することはないので、PSCが管理していれば十分である。Fortranプログラムと比較指示データを入力として結果のリストを表示するような設計がよい。ワークステーション上にGUI（Graphic User Interface）を持っていれば理想的である。Fig. 3.2 に処理のデータの流れを示す。

c) 比較結果は値が同じときにOK、そうでないときにはNGで示される。データを採取しているので、値も持っているはずであり、この値もデバッグの参考となり得るが出力はされない。Fig. 3.3 にPSCの出力例を示す。

3）afrt（VP用）

afrtはフロント・エンド用のベクトル最適化用支援ツールであるが、種々のベクトル化情報と共に、サブルーチン単位で内包されるdoループまたはステップ毎の実行コストを出力する。この情報は並列最適化に際して非常に有用である。並列化に有効な解析例の一部をFig. 3.4 に示す。

3.3 Monte-4

Monte-4では、ANALYZER/m4、ANALYZER-P/m4[5]およびPARALLELIZER/m4[6]が使用可能である。しかしながら、ANALYZER/m4とANALYZER-P/m4の機能は重複しており、並列化においてANALYZER/m4は必要としない。また、PARALLELIZER/m4は、一旦ANALYZER-P/m4で採取した情報ファイルを元に、X11R4に対応したワークステーション上でデータが見られるシステムである。PARALLELIZER/m4では、使用中にANALYZER-P/m4で採取した情報ファイルが簡単に破壊され、その都度、再度ANALYZER-P/m4で情報ファイルを生成して、再びPARALLELIZER/m4で破壊するというような事態が発生する。ANALYZER-P/m4の出力はテーブル出力としては十分に洗練されており、直接情報ファイルを解析した方が効率的である。PARALLELIZER/m4はANALYZER-P/m4の機能を包含して、X window上で情報の生成と解析が行えるようにした方が利用しやすい。また、ファイル保護機構を十分に検討しなければならない。

ANALYZER-P/m4の機能には大別して静的解析機能と動的解析機能があり、動的解析には実行時間解析と実行回数解析がある。ANALYZER-P/m4の機能はfanpコマンドを実行することによって得られる。静的解析機能ではクロス・リファレンス（Cross reference list）、プログラム構造（Program structure list）、プログラム相互参照（Program reference list）、引き数（Argument list）、コモン・リファレンス（Common block reference list）、要約（Summary list）の各リストが得られる。解析例をFig. 3.5 に示す。動的実行時間解析機能では要約（Summary list）、サブルーチンのCPU時間（Program unit Summary list）、コール・パス（Call pass list）、ループ実行回数（Program unit detail list）の各リストが得られる。解析例をFig. 3.6 に示す。動的実行回数解析機能では要約（Summary list）、サブルーチンの実行コストとベクトル化率（Program unit summary list）、ステップ実行回数リスト（Format list）の各リストが得られる。解析例を Fig. 3.7 に示す。

Table 3.1 Analyzer Functions

	Paragon XP/S	VPP500/42	Monte-4
各PEの挙動解析	○	○	
サブルーチン単位の並列化率		○	
サブルーチン単位の並列化実績加速率		○	
プログラム構造			○
プログラム相互参照			○
引き数一覧			○
コール・パス			○
クロス・リファレンス	○	○	○
コモン・リファレンス	○	○	○
サブルーチンの実行時間			○
サブルーチンの実行コスト		○	○
サブルーチンのベクトル化率		○	○
各ステップ毎の実行回数		○	○

```
Status                    : Parallel
Number of Processors      : 4

Type                      : cpu
Interval (msec)           : 10
```

Performance Information : Parallel speedup	Parallel Information : Parallelization ratio	Parallel to serial speed ratio	Load balance	Asynchronous transfer ratio	Name
3.72398190	1.00000000	3.51792336	0.12621359	0.00194175	vdftzf_
2.91101695	0.97900600	3.46737481	0.12067039	0.00000000	vdft3f_
2.89655172	0.95869565	3.50000000	0.12045889	0.00000000	vdftzb_
3.44827586	1.00000000	3.48936170	0.16346154	0.00000000	vdft3b_
1.00000000	0.88477366	1.00000000	0.09459459	0.00168919	getksp_
1.00000000	0.62500000	1.13636364	0.25242718	0.00000000	putksp_
1.00000000	1.00000000	1.00000000	0.00000000	0.00000000	calcfn_
1.00000000	0.79166667	1.00000000	0.00000000	0.00000000	omg2pv_
1.66666667	0.80000000	2.00000000	0.00000000	0.00000000	_start
1.00000000	1.00000000	1.00000000	0.00000000	0.00000000	addvar_
1.00000000	1.00000000	1.00000000	0.97356828	0.00440529	getfld_
1.00000000	0.00000000	1.00000000	0.00000000	0.00000000	mkcnst_
1.00000000	0.33333333	1.00000000	0.00000000	0.00000000	MAIN__
0.00000000	*****	1.00000000	0.95000000	0.00000000	datain_
0.00000000	*****	1.00000000	0.99958403	0.00041597	putfld_
0.00000000	*****	1.00000000	1.00000000	0.00000000	printl_
0.00000000	*****	1.00000000	1.00000000	0.00000000	nextjb_
2.60042061	1.00000000	1.99036539	0.48047117	0.00077495	TOTAL

Synthesis Information (Count)

PM	PMW	PMMW	RM	RMW	RMMW	ALL	AW	AMW	VL	Name
221	27	1	900	76	1	1030	130	2	34	vdftzf_
236	33	0	778	91	0	895	108	0	56	vdft3f_
145	23	0	460	40	0	523	63	0	32	vdftzb_
87	15	0	360	60	0	416	68	0	102	vdft3b_
162	12	0	162	12	0	592	56	1	171	getksp_
37	11	0	40	13	0	103	26	0	607	putksp_
36	0	0	36	0	0	150	0	0	99	calcfn_
16	0	0	16	0	0	54	0	0	147	omg2pv_
6	0	0	10	0	0	18	0	0	-	_start
2	0	0	2	0	0	11	0	0	1996	addvar_
1	1	0	1	1	0	227	221	1	-	getfld_
1	0	0	1	0	0	1	0	0	20	mkcnst_
1	0	0	1	0	0	2	0	0	2048	MAIN__
0	0	0	0	0	0	20	19	0	-	datain_
0	0	0	0	0	0	2404	2403	1	-	putfld_
0	0	0	0	0	0	3	3	0	-	printl_
0	0	0	0	0	0	3	3	0	-	nextjb_
951	122	1	2767	293	1	6452	3100	5	103	TOTAL

Fig. 3.1 Parasamp (to be continue)

```
Synthesis Information (Percent)
      PM        PMW       PMMW|        RM        RMW       RMMW|       ALL        AW       AMW| Name
    23.2        2.8        0.1|      32.5        2.7        0.0|      16.0        2.0        0.0| vdftzf_
    24.8        3.5        0.0|      28.1        3.3        0.0|      13.9        1.7        0.0| vdft3f_
    15.2        2.4        0.0|      16.6        1.4        0.0|       8.1        1.0        0.0| vdftzb_
     9.1        1.6        0.0|      13.0        2.2        0.0|       6.4        1.1        0.0| vdft3b_
    17.0        1.3        0.0|       5.9        0.4        0.0|       9.2        0.9        0.0| getksp_
     3.9        1.2        0.0|       1.4        0.5        0.0|       1.6        0.4        0.0| putksp_
     3.8        0.0        0.0|       1.3        0.0        0.0|       2.3        0.0        0.0| calcfn_
     1.7        0.0        0.0|       0.6        0.0        0.0|       0.8        0.0        0.0| omg2pv_
     0.6        0.0        0.0|       0.4        0.0        0.0|       0.3        0.0        0.0| _start
     0.2        0.0        0.0|       0.1        0.0        0.0|       0.2        0.0        0.0| addvar_
     0.1        0.1        0.0|       0.0        0.0        0.0|       3.5        3.4        0.0| getfld_
     0.1        0.0        0.0|       0.0        0.0        0.0|       0.0        0.0        0.0| mkcnst_
     0.1        0.0        0.0|       0.0        0.0        0.0|       0.0        0.0        0.0| MAIN__
     0.0        0.0        0.0|       0.0        0.0        0.0|       0.3        0.3        0.0| datain_
     0.0        0.0        0.0|       0.0        0.0        0.0|      37.3       37.2        0.0| putfld_
     0.0        0.0        0.0|       0.0        0.0        0.0|       0.0        0.0        0.0| printl_
     0.0        0.0        0.0|       0.0        0.0        0.0|       0.0        0.0        0.0| nextjb_

Processor Information (Count)
  VPID = 1

                                                                                    省略

Processor Information (Percent)
  VPID = 1

                                                                                    省略

Processor Information (Count)
  VPID = 2

                                                                                    省略

Processor Information (Percent)
  VPID = 2

                                                                                    省略

Processor Information (Count)
  VPID = 3

                                                                                    省略

Processor Information (Percent)
  VPID = 3

                                                                                    省略

Processor Information (Count)
  VPID = 4

                                                                                    省略

Processor Information (Percent)
  VPID = 4

                                                                                    省略
```

Fig. 3.1 Parasamp

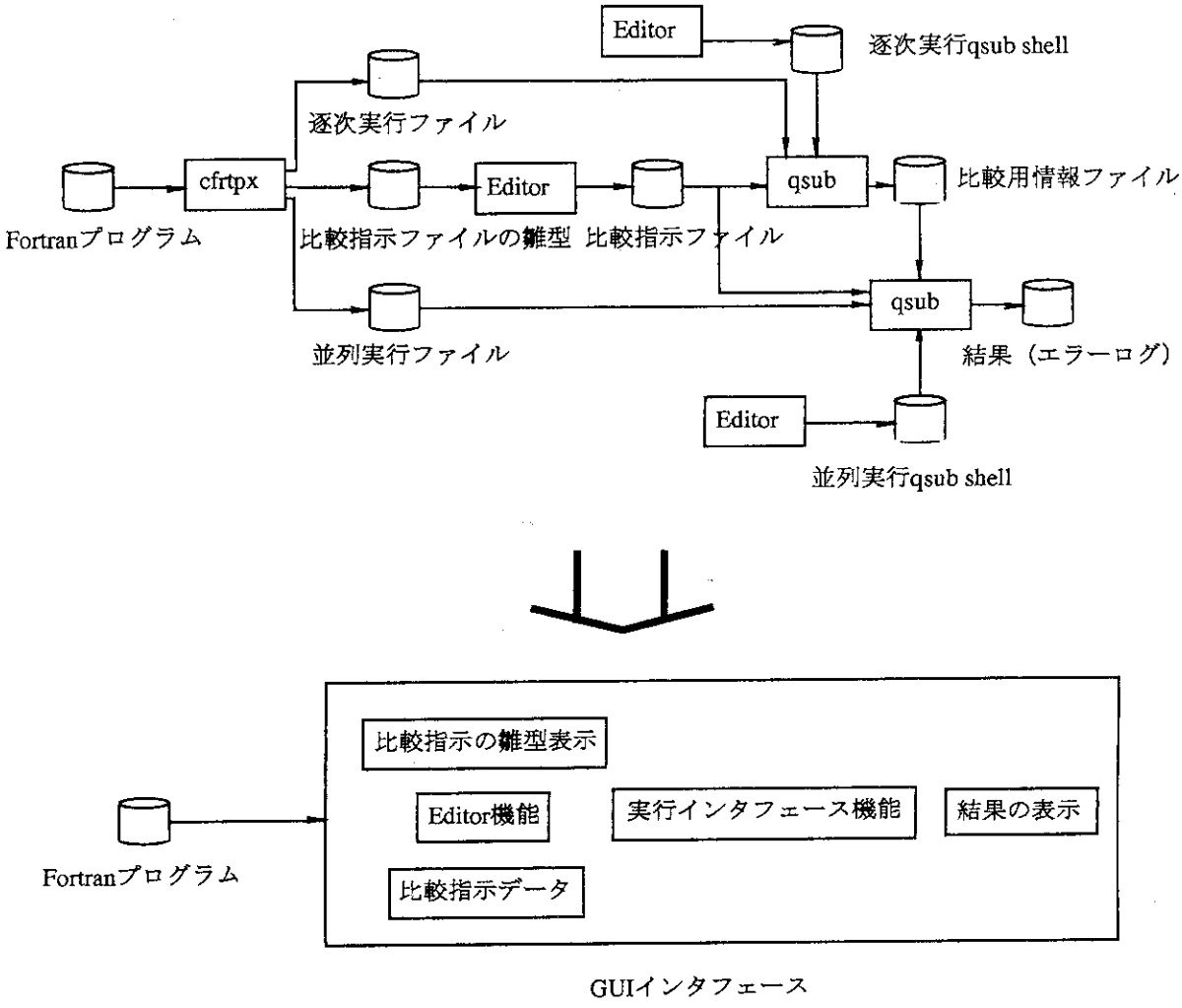


Fig. 3.2 PSC - flow

```
*****test.f, LINE 20, PASS 1, sum, WKR : NG
*****test.f, LINE 20, PASS 1, sum, V1 : OK
*****test.f, LINE 20, PASS 1, sum, V2 : OK
*****
```

Fig. 3.3 PSC - output

```
  vectorize - routine list ------------------------------------------------------

routine    ex-count     v-cost     %     s-cost     %     v-leng  v-rate
VDFT3F         1200   .4316E10  99.8   .4413E11  99.9        33    99.8

vectorize - statement list -- VDFT3F ------------------------------------------

isn       ex-count   true    v-cost    s-cost v o c--------1---------2---------3
00000001      1200            21600     21600           subroutine  vdft3f( n, n
00000002
00000003                                                  dimension  fr(np,n,n),
00000004                                                  dimension  cn(n2*nl1),
00000005                                                  common  /cmvdft/ leven,
00000006
00000007                                                c -----( Fourier transforms in
00000008
00000009                                                *vdir novector
00000010      1200            18000     18000 S           do  700  istage = 1, nl1
00000011
00000012      2400             4800      4800 S             kbase  =  n2*(istage-1
00000013                                                *vocl loop,novrec
00000014                                                *vdir nodep
00000015      2400            12000     12000 S             do  710  k = 1, n2
00000016     38400            38400     38400 S               k2      =  2*k
00000017     38400            38400     38400 S               k1      =  k2 - 1
00000018     38400           192000    192000 S 2             do  710  j = 1, n
00000019   1228800          6144000   6144000 V 2               do  710  i = 1, np
00000020  41779200          .1044E09  .1086E10 V 2                 cr(i,j,k1)  =  f
```

.
.
.

以下省略

Fig. 3.4 afrt (part of the result)

```
*-----------------------------------------*  Only for MKCNST

   CROSS REFERENCE LIST
      PROGRAM UNIT NAME = MKCNST
      FILE NAME         = mkcnst.f
*-----------------------------------------*

NAME          CLASS     SCOPE  TYPE          SIZE    REF. LINE

CO            ARRAY     UK     R*4              ?    000001(DA )  000003       *000009      000015
I             VARIABLE  PV     I*4              4    *000008      000009(AA )  *000014      000015      000016
                                                     000017       000019       *000025      000026
J             VARIABLE  PV     I*4              4    *000013      000015       000016       000017      000019
                                                     *000024      000026
K             VARIABLE  PV     I*4              4    *000012      000015       000016       000017      000019
                                                     *000023      000026
MKCNST        EXT. SUB                               000001
NK            VARIABLE  UK     I*4              4    000001(DA )  000003       000008       000012      000013
                                                     000014       000023       000024       000025
NK2           VARIABLE  UK     I*4              4    000001(DA )  000009(AA )
OKK           ARRAY     UK     R*4              ?    000001(DA )  000003       *000017      *000019
REAL          GEN. FUNC        R*4              4    000009
RKK           ARRAY     UK     R*4              ?    000001(DA )  000003       *000015      000016      000017
                                                     *000026
RNU           VARIABLE  UK     R*4              4    000001(DA )  000026


*-------------------------*
   PROGRAM STRUCTURE LIST
*-------------------------*
    (  ****:PROG. ENTRY  ++++:SUB. ENTRY  ####:TASK. ENTRY /MAIN. ENTRY/  (EXT. PROC)  "MACRO. SUBR"  )

       ****:TRANS5

STRUCTURE NO:1
LINE     ****:TRANS5
           I
 000030    I--DATAIN
           I
 000033    I--MKCNST
           I  I
 000009    I  I--:REAL:
           I
 000036    I--VDFT3I
           I  I
 000007    I  I--:REAL:
           I  I
 000008    I  I--:MOD:
           I  I
 000009    I  I--:REAL:
           I  I
 000012    I  I--:ACOS:
           I  I
 000014    I  I--:REAL:
           I  I
 000015    I  I--:COS:
           I  I
 000016    I  I--:SIN:
           I  I
 000029    I  I--:MOD:
           I  I
 000044    I  I--:MOD:
           I
 000039    I--GETFLD
           I  I
 000007    I  I--:INDEX:
           I
 000061    I--PRINTL
           I
 000092    I--CALCFN:<--D002
           I  I
 000035    I  I--OMG2PV
           I  I  I
 000046    I  I  I--PUTKSP
           I  I  I
 000047    I  I  I--PUTKSP
           I  I  I
```

Fig. 3.5 fanp - static analysis (to be continued)

```
000048   I  I  I--VDFT3B(143)
         I  I  I
000051   I  I  I--PUTKSP
         I  I  I
000052   I  I  I--PUTKSP
         I  I  I
000053   I  I  I--VDFT3B(143)
         I  I  I
000056   I  I  I--PUTKSP
         I  I  I
000057   I  I  I--PUTKSP
         I  I  I
000058   I  I  I--VDFT3B(143)
         I  I
000042   I  I--CONVOL:<--D001
         I  I  I
000025   I  I  I--VDFT3F
         I  I  I
000028   I  I  I--GETKSP
         I  I  I
000029   I  I  I--GETKSP
         I  I
000045   I  I--CONVOL:-->D001
         I  I
000048   I  I--CONVOL:-->D001
         I  I
000051   I  I--CONVOL:-->D001
         I  I
000054   I  I--CONVOL:-->D001
         I  I
000057   I  I--CONVOL:-->D001
         I
000097   I--ADDVAR
         I
000098   I--ADDVAR
         I
000102   I--CALCFN:-->D002
         I
000107   I--ADDVAR
         I
000108   I--ADDVAR
         I
000112   I--CALCFN:-->D002
         I
000117   I--ADDVAR
         I
000118   I--ADDVAR
         I
000122   I--CALCFN:-->D002
         I
000127   I--ADDVAR
         I
000146   I--PUTFLD
         I  I
000007   I  I--:INDEX:
         I
000152   I--NEXTJB
```

Fig. 3.5 fanp - static analysis (to be continued)

```
*-------------------------*
  PROGRAM REFERENCE LIST
*-------------------------*
```

PROGRAM	CLASS	TYPE	DEF. FILE	LINE	REF. PROGRAM	LINE				
ACOS	GEN. FUNC	R*4			VDFT3I	000012				
ADDVAR	EXT. SUB		addvar. f	000001	TRANS5	000097	000098	000107	000108	000117
						000118	000127			
CALCFN	EXT. SUB		calcfn. f	000001	TRANS5	000092	000102	000112	000122	
CONVOL	EXT. SUB		convol. f	000001	CALCFN	000042	000045	000048	000051	000054
						000057				
COS	GEN. FUNC	R*4			VDFT3I	000015				
DATAIN	EXT. SUB		datain. f	000001	TRANS5	000030				
GETFLD	EXT. SUB		getfld. f	000001	TRANS5	000039				
GETKSP	EXT. SUB		getksp. f	000001	CONVOL	000028	000029			
INDEX	GEN. FUNC	I*4			GETFLD	000007				
					PUTFLD	000007				
MKCNST	EXT. SUB		mkcnst. f	000001	TRANS5	000033				
MOD	GEN. FUNC	I*4			VDFT3I	000008	000029	000044		
NEXTJB	EXT. SUB		nextjb. f	000001	TRANS5	000152				
OMG2PV	EXT. SUB		omg2pv. f	000001	CALCFN	000035				
PRINTL	EXT. SUB		printl. f	000001	TRANS5	000061				
PUTFLD	EXT. SUB		putfld. f	000001	TRANS5	000146				
PUTKSP	EXT. SUB		putksp. f	000001	OMG2PV	000046	000047	000051	000052	000056
						000057				
REAL	GEN. FUNC	R*4			MKCNST	000009				
					VDFT3I	000007	000009	000014		
SIN	GEN. FUNC	R*4			VDFT3I	000016				
TRANS5	MAIN		trans5. f	000001						
VDFT3B	EXT. SUB		vdft3b. f	000001	OMG2PV	000048	000053	000058		
VDFT3F	EXT. SUB		vdft3f. f	000001	CONVOL	000025				
VDFT3I	EXT. SUB		vdft3i. f	000001	TRANS5	000036				

```
*-----------------*  Only for VDFT3B & VDFT3F
  ARGUMENT LIST
*-----------------*
```

PROGRAM	REF. LINE	ARGUMENT								
*** VDFT3B ***		N	NP	N2	NL1	FR	FI	CR	CI	CN	SN
		IBR								
OMG2PV	000048	N	NP	N2	NL1	#1	WKI	W2R	W2I	CN	SN
		IBR								
		1:UPR(1, 1, 1, 1)								
	000053	N	NP	N2	NL1	#1	WKI	W2R	W2I	CN	SN
		IBR								
		1:UPR(1, 1, 1, 2)								
	000058	N	NP	N2	NL1	#1	WKI	W2R	W2I	CN	SN
		IBR								
		1:UPR(1, 1, 1, 3)								
*** VDFT3F ***		N	NP	N2	NL1	FR	FI	CR	CI	CN	SN
		IBR								
CONVOL	000025	N	NP	N2	NL1	WKR	WKI	W2R	W2I	CN	SN
		IBR								

Fig. 3.5 fanp - static analysis (to be continued)

```
*-------------------------*  Only for CMFILE
  COMMON BLOCK REFERENCE LIST
*-------------------------*

OFFSET              ELEMENT      TYPE      REF. PROGRAM    LINE

*-------------------------* COMMON *-------------------------*

*** CMFILE  ***
0        :31        HEADER       CH*32     DATAIN          .000004  .000004  *000009   000023
                                            GETFLD          .000004  .000004   000007   000010
                                            NEXTJB          .000004  .000004   000014
                                            PUTFLD          .000004  .000004   000007   000010
                                            TRANS5          .000025  .000025
32       :63        FLNAME       CH*32     DATAIN          .000004  .000004
                                            GETFLD          .000004  .000004  *000010   000012   000032
                                            NEXTJB          .000004  .000004
                                            PUTFLD          .000004  .000004  *000010   000012   000013
                                            TRANS5          .000025  .000025
64       :68        FILENO       CH*5      DATAIN          .000004  .000004
                                            GETFLD          .000004  .000004  *000009   000010
                                            NEXTJB          .000004  .000004
                                            PUTFLD          .000004  .000004  *000009   000010
                                            TRANS5          .000025  .000025


*--------------*
  SUMMARY LIST
*--------------*

ANALYZER-P/M4 REVISION :  REV.157
ANALYZED DATE          :  Thu Jul 13 14:26:02 1995
FANP OPTIONS           :  -st -AD o=st.dbs  -AL f=st.l
                                : xref all programs
                                : struct all programs
                          iall line prog arg com

ANALYZED DATABASE
    ACCESS MODE                 : OUTPUT
    DIRECTORY                   : st.dbs

PROGRAM UNIT ANALYSIS INFOMATION
    PROGRAM UNIT (INPUT)        : 16
                 (ANALYZED)     : 16

PROGRAM UNIT INFORMATION

    PROGRAM       ATR.    SYNTAX ERROR                  AD FILE       FILE

    TRANS5        MAIN    NO ERROR                      OUTPUT        trans5.f
    DATAIN        SUB     NO ERROR                      OUTPUT        datain.f
    MKCNST        SUB     NO ERROR                      OUTPUT        mkcnst.f
    CALCFN        SUB     NO ERROR                      OUTPUT        calcfn.f
    ADDVAR        SUB     NO ERROR                      OUTPUT        addvar.f
    OMG2PV        SUB     NO ERROR                      OUTPUT        omg2pv.f
    PUTKSP        SUB     NO ERROR                      OUTPUT        putksp.f
    GETKSP        SUB     NO ERROR                      OUTPUT        getksp.f
    CONVOL        SUB     NO ERROR                      OUTPUT        convol.f
    GETFLD        SUB     NO ERROR                      OUTPUT        getfld.f
    PUTFLD        SUB     NO ERROR                      OUTPUT        putfld.f
    PRINTL        SUB     NO ERROR                      OUTPUT        printl.f
    NEXTJB        SUB     NO ERROR                      OUTPUT        nextjb.f
    VDFT3I        SUB     NO ERROR                      OUTPUT        vdft3i.f
    VDFT3F        SUB     NO ERROR                      OUTPUT        vdft3f.f
    VDFT3B        SUB     NO ERROR                      OUTPUT        vdft3b.f

PROGRAM ANALYSIS INFORMATION
    MACROTASK                   : NO
    MICROTASK                   : NO
    ERROR (ARGUMENT)            : NO ERROR
```

Fig. 3.5 fanp - static analysis

```
*-------------------------*
  SUMMARY LIST
*-------------------------*

ANALYZER-P/M4 REVISION :  REV.157
ANALYZED DATE          :  Fri Jul 14 15:54:39 1995
FANP OPTIONS           :  -tm -AL f=tm.l lpath
                             : lpunit all programs
                          -af ana.tm -AO
                             : do all programs
                          -AR trans5.tim

EXECUTION TIME         :  CPU TIME      =   0 : 00 ' 26 " 782 (    26.782 sec)
                          ELAPSED TIME   =   0 : 25 ' 19 " 197 (  1519.197 sec)

INSTRUCTION INFORMATION:  INSTRUCTION COUNT           =     1178884035
                          FLOATING POINT ELEMENT COUNT =     4235369544
                          VECTOR INSTRUCTION COUNT     =      272701586
                          VECTOR ELEMENT COUNT         =     9206433186

PERFORMANCE            :    377.597 MOPS    158.145 MFLOPS  (BY CPU TIME)

MEMORY INFORMATION     :  BANK CONFLICT : NONE

VECTOR INFORMATION     :  VECTOR OPERATION RATIO =  91.04 %
                          AVERAGE VECTOR LENGTH  =  33.8
```

```
*-----------------------------------------------------------------------*
  PROGRAM UNIT SUMMARY LIST
*-----------------------------------------------------------------------*
```

PROG. UNIT	ATR.	CODE	FREQUENCY	INCLUSIVE CPU TIME(%)	EXCLUSIVE CPU TIME(%)	MOPS	MFLOPS	V.OP. RATIO	AVER. V.LEN	BANK CONF
VDFT3F	SUB		1200	17.858(66.7)	17.858(66.7)	345.4	152.8	91.42	34.1	
VDFT3B	SUB		600	5.445(20.3)	5.445(20.3)	552.5	238.3	90.68	33.5	
GETKSP	SUB		2400	1.546(5.8)	1.546(5.8)	72.7	0.0	68.33	17.3	
PUTKSP	SUB		1200	0.867(3.2)	0.867(3.2)	147.5	0.0	80.36	31.9	
CONVOL	SUB		1200	19.881(74.2)	0.478(1.8)	548.6	87.5	95.64	64.0	
OMG2PV	SUB		200	6.572(24.5)	0.260(1.0)	426.8	147.5	93.59	28.0	
CALCFN	SUB		200	26.637(99.5)	0.184(0.7)	1265.2	522.0	93.52	22.8	
ADDVAR	SUB		350	0.118(0.4)	0.118(0.4)	727.1	285.4	98.13	64.0	
TRANS5	MAIN		1	26.782(100.0)	0.017(0.1)	300.0	0.0	94.79	64.0	
PUTFLD	SUB		5	0.004(0.0)	0.004(0.0)	22.8	0.0	0.67	46.5	
DATAIN	SUB		1	0.003(0.0)	0.003(0.0)	37.1	0.0	0.28	33.7	
GETFLD	SUB		1	0.001(0.0)	0.001(0.0)	22.1	0.0	4.28	46.5	
MKCNST	SUB		1	0.001(0.0)	0.001(0.0)	118.6	34.9	86.60	23.2	
PRINTL	SUB		5	0.001(0.0)	0.001(0.0)	16.7	0.1	0.00	0.0	
NEXTJB	SUB		1	0.000(0.0)	0.000(0.0)	25.0	0.1	0.00	0.0	
VDFT3I	SUB		1	0.000(0.0)	0.000(0.0)	447.7	27.2	89.93	27.1	

Fig. 3.6 fanp - dynamic analysis - execution time (to be continued)

```
*-------------------------------------------------------------*
  CALL PATH LIST
*-------------------------------------------------------------*

LINE          NESTING                  FREQUENCY  INCLUSIVE          EXCLUSIVE          P/V  AVERAGE   BANK
                                                  CPU TIME(   %)     CPU TIME(   %)          LOOP.LEN. CONF

000001        TRANS5                           1   26.782(100.0)      0.000(  0.0)
000030        -DATAIN                          1    0.003(  0.0)      0.003(  0.0)
000033        -MKCNST                          1    0.001(  0.0)      0.000(  0.0)
000008--000010 --MKCNST/DO 100                 1    0.000(  0.0)      0.000(  0.0)  V          20
000012--000021 --MKCNST/DO 200                 1    0.001(  0.0)      0.000(  0.0)             20
000013--000021 ---MKCNST/DO 200               20    0.001(  0.0)      0.000(  0.0)             20
000014--000021 -----MKCNST/DO 200             400    0.001(  0.0)      0.001(  0.0)  V          20
000023--000027 --MKCNST/DO 300                 1    0.000(  0.0)      0.000(  0.0)  W          20
000024--000027 ---MKCNST/DO 300
000025--000027 -----MKCNST/DO 300
000036        -VDFT3I                          1    0.000(  0.0)      0.000(  0.0)
000013--000017 --VDFT3I/DO 100                 1    0.000(  0.0)      0.000(  0.0)  V          16

                                                                        .
                                                                        .
                                                                        .

000096--000112 ------CALCFN/DO 400         20000    0.018(  0.1)      0.018(  0.1)  V          20
000127        ---ADDVAR                       50    0.017(  0.1)      0.000(  0.0)
000008--000014 -----ADDVAR/DO 100             50    0.017(  0.1)      0.017(  0.1)  W           3
000009--000014 ------ADDVAR/DO 100
000010--000014 -------ADDVAR/DO 100
000011--000014 --------ADDVAR/DO 100
000132--000144 --TRANS5/DO 400                 5    0.000(  0.0)      0.000(  0.0)             3
000134--000144 ---TRANS5/DO 400               15    0.000(  0.0)      0.000(  0.0)             3
000136--000144 ----TRANS5/DO 400              45    0.000(  0.0)      0.000(  0.0)             3
000146        --PUTFLD                         5    0.004(  0.0)      0.004(  0.0)
000152        -NEXTJB                          1    0.000(  0.0)      0.000(  0.0)
```

Only for CONVOL

```
*-------------------------------------------------------------*
  PROGRAM UNIT DETAIL LIST
     PROGRAM UNIT NAME = CONVOL
     FILE NAME         = convol.f
*-------------------------------------------------------------*

LINE          NESTING                  FREQUENCY  INCLUSIVE          EXCLUSIVE          P/V  AVERAGE   BANK
                                                  CPU TIME(   %)     CPU TIME(   %)          LOOP.LEN. CONF

000001        CONVOL                        1200   19.881( 74.2)      0.024(  0.1)
000012--000016 -DO 100                      1200    0.275(  1.0)      0.275(  1.0)  W          32
000013--000016 --DO 100
000014--000016 ---DO 100
000018--000022 -DO 200                      1200    0.179(  0.7)      0.179(  0.7)  W          32
000019--000022 --DO 200
000020--000022 ---DO 200
000025        -VDFT3F                       1200   17.858( 66.7)     17.858( 66.7)
000028        -GETKSP                       1200    0.773(  2.9)      0.773(  2.9)
000029        -GETKSP                       1200    0.773(  2.9)      0.773(  2.9)
```

Fig. 3.6 fanp - dynamic analysis - execution time

```
*-------------------------*
  SUMMARY LIST
*-------------------------*


ANALYZER-P/M4 REVISION :  REV. 157
ANALYZED DATE          :  Wed Jul 12 15:21:02 1995
FANP OPTIONS           :  -ct -AL f=ct.l cost 90
                               : fmt all programs
                          -wkk wkk

EXECUTION INFORMATION  :  CPU TIME                    =   0 : 00 ' 31 " 553
                          TOTAL EXECUTION COUNT       = 3981212188

VECTOR INFORMATION     :  VECTORIZATION RATIO         =  99.97 %



*-----------------------------------*
  PROGRAM UNIT SUMMARY LIST
*-----------------------------------*

PROG. UNIT  ATR.     FREQUENCY  EXEC    VECTOR
                                COST%   RATIO%

VDFT3F      SUB           1200   57.2    99.99
VDFT3B      SUB            600   28.5    99.99
CONVOL      SUB           1200    5.9    99.97
PUTKSP      SUB           1200    3.3    99.85
GETKSP      SUB           2400    1.7    99.43
CALCFN      SUB            200    1.6    99.96
ADDVAR      SUB            350    0.9    99.96
OMG2PV      SUB            200    0.7    99.95
TRANS5      MAIN             1    0.1    99.79
MKCNST      SUB              1    0.0    99.92
VDFT3I      SUB              1    0.0    99.20
PUTFLD      SUB              5    0.0     0.00
GETFLD      SUB              1    0.0     0.00
DATAIN      SUB              1    0.0     0.00
PRINTL      SUB              5    0.0     0.00
NEXTJB      SUB              1    0.0     0.00
```

Fig. 3.7 fanp - dynamic analysis - execution count (to be continued)

```
*-------------------------------------------------------------------*  Only for CONVOL
  FORMAT LIST
     PROGRAM UNIT NAME     = CONVOL
     FILE NAME             = convol.f

     EXECUTION CPU TIME    =   0 : 00 ' 01 " 333 (      1333 MSEC) (  4.2% )
     EXECUTION FREQUENCY   =    169658400                          (  4.3% )
     EXECUTION COST RATIO  =      5.9%
*-------------------------------------------------------------------*

 LINE    EXECUTION COST(%) TRUE%   LOOP          FORTRAN STATEMENT

000001       1200                                  subroutine  convol( n,     np,    n2,    nk,    nk2,   nl1,
000002                                            &                   wrr,   wri,   v1,    v2,
000003                                            &                   wkr,   wki,   w2r,   w2i,   cn,    sn,    ibr )
000004                                             *
000005                                             dimension   wrr(nk,nk,nk),   wri(nk,nk,nk)
000006                                             dimension   v1(np,n,n),      v2(np,n,n),     wkr(np,n,n)
000007                                             dimension   wki(np,n,n),     w2r(np,n,n),     w2i(np,n,n)
000008                                             dimension   cn(n2*nl1),      sn(n2*nl1),      ibr(n*nl1)
000009                                             *
000010                                             *
000011
000012       1200(  0.00)                          do  100  k = 1,  n
000013      38400(  0.00)       *---------->       !  do  100  j = 1,  n
000014    1228800(  0.03)       ! *---------->     !  !  do  100  i = 1,  np
000015   41779200(  0.81)       ! ! W------->      !  !  !  wkr(i,j,k)  =  v1(i,j,k)*v2(i,j,k)
000016   41779200(  3.04)       *-*-W--------  100  continue
000017
000018       1200(  0.00)                          do  200  k = 1,  n
000019      38400(  0.00)       *---------->       !  do  200  j = 1,  n
000020    1228800(  0.03)       ! *---------->     !  !  do  200  i = 1,  np
000021   41779200(  0.20)       ! ! W------->      !  !  !  wki(i,j,k)  =  0.0d0
000022   41779200(  1.82)       *-*-W--------  200  continue
000023                                             *
000024                                             *
000025       1200(  0.00)                          call  vdft3f( n, np, n2, nl1, wkr, wki, w2r, w2i, cn, sn, ibr )
000026                                             *
000027                                             *
000028       1200(  0.00)                          call  getksp( n, np, nk, nk2, wkr, wrr )
000029       1200(  0.00)                          call  getksp( n, np, nk, nk2, wki, wri )
000030                                             *
000031                                             *
000032       1200(  0.00)                          return
000033                                             end
```

Fig. 3.7 fanp - dynamic analysis - execution count

4．並列化

並列化には大まかに分けて次の３つの方法がある。

1）メッセージ・パッシング・ライブラリを使用した並列化
2）Fortranパラダイムによる並列化
3）自動並列化

メッセージ・パッシング・ライブラリを使用した並列化は、並列化で発生するPE間の制御や分散主記憶に分割された情報を、利用者が通信ライブラリを使用してプログラムを作成し並列化を行うことである。また、Fortranパラダイムによる並列化は、並列化機能を備えたFortranを使用して並列化することであり、自動並列化とは、コンパイラまたはプリプロセッサがプログラムを解析し、マルチタスキングを行うコードあるいはプログラムを自動的に生成する機能を利用しての並列化を意味する。

メッセージ・パッシング・ライブラリを使用した並列化においては、並列化の性能は基本的な部分でライブラリ自身の性能とそれを利用したプログラムの通信アルゴリズムに依存するため、十分な機能と性能を有した洗練されたライブラリ群が提供されていなければならない。一方Fortranパラダイムによる並列化では、利用しようとする並列Fortranに関する規則、用語、常識、習慣等を十二分に理解していることが前提となる。それゆえ、親切で簡潔かつ漏れのないマニュアル等が整備されていることが必須である。特記すべくもなく、自動並列化機能を装備した計算機が最も利用する面で容易である。しかしながら、現時点では自動並列化機能を分散主記憶型の並列計算機に展開することが難しく成果は上がっていない。

Paragon XP/Sではメッセージ・パッシング・ライブラリを使用した並列化を行い、VPP500ではFortranパラダイムによる並列化を、また、Monte-4では自動並列化を行った。

4.1 Paragon XP/S

並列計算機では計算順序の制御や各PEが計算した結果の交換のために通信が発生する。Paragon XP/S には、グローバル・アドレッシングの機能がないため、すべての場合においてPE間のデータ通信ソフトを組まなければならない。一般的に、一定量の通信すべきデータ量があるとき、細かく分割して送受信を行うより、一回あたりの通信量を増やし通信回数を抑制したほうが通信のオーバーヘッド総量が減少し効率がよい。

データはFig. 4.1 に示すとうりCコンパイラとFortranコンパイラで領域の確保の仕方が異なる。Fortranで記述されたプログラムでは配列データは最後の次元でまとめられており、ここでデータの分割を行うのが一回あたりの通信量は増えるが通信回数を抑制でき自然である。

Paragon XP/Sでプログラムを並列化する時には、入出力の部分に注意を払う必要がある。単純に並列化すると各PEで重複して同ファイルをアクセスするため、最初にclose命令を発行したPEがファイルを閉じてしまい、他のPEでエラーが発生する。したがって、入力時には代表の

PEがデータを読み込んだ後各PEに転送し、出力時には代表PEがデータを集めファイルに書き込む。プログラムの入出力の部分は全てこのような書換が必要である。

Trans5の３次元FFTルーチンは各方向に１次元FFTを実行する。ここで、X軸方向の並列化を単純にZ軸成分でPEに分散しても、Y軸方向への影響がないため、Y軸方向の並列化も簡単に行うことができる。しかしながら、Z軸方向の並列化はZ軸成分にfr(i,j,k+n2)等の添え字の演算が発生し、単純に各PEに分散することができない。したがって、各PEが計算してきたZ軸方向のXY軸成分を交換、すなわち「all to all」の通信が必要となってくる。また、Z軸方向の計算後、順次XY軸方向の計算が継続されるのでZ軸方向の計算後も「all to all」の通信が必要である。この際、前述した通信回数を抑制するため、転置行列を作成してデータをまとめておく作業も必要となってくる。ここで、Z軸方向のFFT前の転置行列の作成は、次のZ軸方向のFFTを行うときに分割する「軸」で並列化可能である。「all to all」の通信や再転置では並列化は不可能であるが、vdft3bはプログラム全体のに占めるコストが大きく、この部分の加速率へ及ぼす影響は大である。Fig. 4.2 に並列化概念図を示す。

「all to all」の通信を行うために利用できるライブラリには、csend - crecvの同期送受信システム・コールとisend - irecvの非同期送受信システム・コールがある。また、isend - crecvのような組み合わせも可能である。csendは受信側の受信完了を待ち合わせるためのシステム・コールではない。csendでは送信データをシステム・バッファに書き込んで制御を呼び出し側に返すが、isendは即、制御を呼び出し側に返すため、必要に応じmsgwaitで待ち合わせを行う。受信も同様で、crecvでは受信データをシステム・バッファから読み込んで制御を呼び出し側に返すが、isendは即、制御を呼び出し側に返すため、必要に応じmsgdoneで待ち合わせを行う。読み込みが完了したかはflickでシステムに制御を返し、一定時間をおいてmsgdoneで読み込みが完了するまでチェックを行う。vdft3bでは受信が完了しないと次に進めないため、isend - irecvでは完了確認が必要となる。通常、同期通信の方が非同期通信より高速であるが、読み込み完了チェックを行ったためか、csend - crecvはisend - irecvと比較して20%から30%高速であった。また、isend - irecvの送受信をループで括り実行すると、１回目は正常終了するが、２回目にはmsgdoneで何度チェックしても「ready」が返って来ないという現象も発生し、本並列化では高速なcsend - crecvシステム・コールで通信ルーチンを作成した。

並列化したvdft3bをFig. 4.3 に示す。

4.2 VPP500

Fortranパラダイムfrtpxを使用した並列化では、宣言による並列化作業の比重が大きい。実際、並列化に必要な作業は８割方以上がデータの新規および再宣言、コモンの定義し直し、サブルーチンの引き数の変更等である。VPP500ではグローバル・アドレッシングが提供されているが、この機能を使用すると並列化性能は著しく低下する。最適化では、グローバル・アドレッシング機能があっても、この機能を如何に使用しないかがポイントとなってくる。また、VPP500では入出力を並列化しても、自動的にマスタ・プロセッサ１PEのみがアクセスするように制御されているが、目的とするPE数毎のプログラム・コンパイルが必要である。今回、試みた並列化

手順を以下に示す。

1）doループ内の分割処理する変数を各PEからアクセス可能なグローバル変数として宣言する。
2）1）のとき、各PEへのデータの分割割り付けも同時に設計する。
3）1）に対応する変数を、PEに固有なローカル・データとして宣言し、equivalence文で結合する。
4）3）の変数名でサブルーチン内の旧変数名を書き換える。
5）1）と6）の変更を踏まえて1）の変数をサブルーチン間で共通なコモンとして宣言しておく。
6）サブーチンは一般的にアドレスで引き数渡しを行っているので、すべて値渡しに変更する。さらに、コモン宣言した部分もあるので、コーリング・シーケンスの変更とそれに伴うプログラム変更を行う。

この過程において、サブルーチンvdft3bのZ軸方向をvdftzbとして分割した。6）の作業によって変更した呼び出し側部分をFig. 4.4 に例示する。各サブルーチンで1）の変数宣言が一致しないと、コンパイラがノー・エラーで終了しても、リンクを取ったプログラムは誤動作し正常終了しない。このためこの段階では、計算結果の確認ができないので、通常の場合5）と6）に関する変更をしたVPP500並列化用単一版を別に作成し、変更途中で一旦プログラムの動作チェックをするのが適当である。

7）1）の変数を使用しているすべてのサブルーチンのデータ宣言を書き直す。

このとき、サブルーチン毎に同様の書き変えを行わなくてはいけないので、宣言部分をInclude文として、その内容を作成するのが適当である。

8）データの分割と処理の分割が一致するように処理の分割をおこなう。（今回はサイクリックに分割した。）
9）FFTルーチンのZ軸方向の処理分割は転置行列を作成して、初めて並列化可能なため、転置を掛けて、最後にまた戻しておく。

このようにして作成した並列版Trans5は、オリジナル単一版と比較して15倍低速なプログラムとなった。原因は前述したグローバル・アドレッシングの乱用にあるため、以下の作業を行う。

10）効率的転置作業を行うためには、spread moveとspread doを利用する。また、グローバル・アドレッシングをできる限り廃止し、必要な場合はデータを連続領域に集め、転送される量を増やしてspread moveで一括転送を行うようにプログラムを最適化する。

以上でプログラムの並列最適化が完了するわけであるが、10）の最適化を徹底しないと加速率

1倍以上を得るのは困難である。

また、9）終了後の計算結果の確認作業において、変更量が膨大であったため、途中結果をwrite文とstop文で確認したところ、単一版と異なる途中結果が並列版で得られた。VPPで並列化を行うとグローバル変数がある場合、そのループはベクトル化されず、スカラ版とベクトル版の結果一致は保証されていない。このことはデバッグ上大変な問題であり、解決策としては、変数の精度を上げて計算を実行する。それでも、結果の正否の判断は困難である。本コードに関しては、write文とstop文を外して最終結果を比較したところ、完全な一致を見た。

vdft3bの一部で別サブルーチンにしたvdftzbのみをFig. 4.5 に示す。

4.3 Monte-4

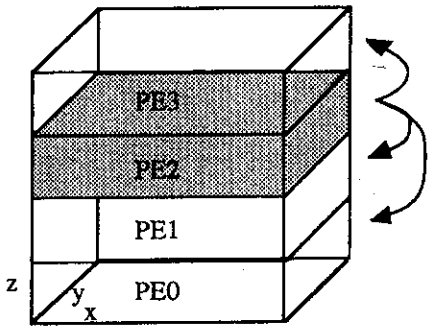
Monte-4では自動並列化機能がサポートされており、この機能を使用して並列化を行った。Monte-4の自動並列化ではfopp（最適化プリプロセッサ）が並列コードを吐き出すように設計されており、主だったdoループを別サブルーチンとして定義し並列化を行っている。自動並列化されたvdft3bのサブルーチンには新規にVDFT3B$1からVDFT3B$12までの12本のサブルーチンが組み込まれた。一本のサブルーチンの分岐と復帰に要する時間を測定したところ7.30E-05秒から7.50E-05秒程度の時間が消費されていることが判明した。並列化されたvdft3bの実行負荷は4.35E-03秒であったので、約1/5の時間が分岐オーバーヘッドとして消費されていることになる。Monte-4のFortranのオプションpi（サブルーチンのインライン展開機能）は、デフォルトでline=50となっており、これらの12本の子サブルーチンはいずれも50行以下で、すべてインライン展開の対象となるが、デフォルトがNpi（サブルーチンのインライン展開をしない）に設定されているため、結局、これらのサブルーチンはデフォルトではインライン展開されない。

自動並列化機能を使用したvdft3bのサブルーチン・リストをFig. 4.6 に示す。

(2,2,2)のデータ領域の取られ方

C	(1,1,1)	(1,1,2)	(1,2,1)	(1,2,2)	(2,1,1)	(2,1,2)	(2,2,1)	(2,2,2)
Fortran	(1,1,1)	(2,1,1)	(1,2,1)	(2,2,1)	(1,1,2)	(2,1,2)	(1,2,2)	(2,2,2)

z方向で並列化し、計算結果を「all to all」で各PEに還元する。




```
iam=mynode()
nodes=numnodes()

do 300 k=iam+1,n,nodes
   do 300 j=1,n
      do 300 i=1,n
         f(i,j,k) = g(i,j,k)
300  continue
```

Fig. 4.1 Data assignment

```
subroutine vdft3b
    X軸方向のFFT
    Y軸方向のFFT
    「all to all」・・・・・・・・・・・Z軸方向のFFTを計算する準備（並列化不可）
    転置行列の作成・・・・・・・・・Z軸方向の計算後「all to all」を実行する準備
    Z軸方向のFFT
    「all to all」・・・・・・・・・・・XY軸方向のFFTを計算する準備（並列化不可）
    再転置・・・・・・・・・・・・・・XY軸方向のFFTを計算する準備（並列化不可）
return
end
```

Fig. 4.2 Parallelizing for FFT（vdft3b）

```
      subroutine  vdft3b( n, np, n2, nl1, fr, fi, cr, ci, cn, sn, ibr )

      dimension  fr(np,n,n),  fi(np,n,n),  cr(np,n,n),  ci(np,n,n)
      dimension  cn(n2*nl1),  sn(n2*nl1),  ibr(n*nl1)
      common  /cmvdft/ leven, oncube

      include 'fnx.h'
      iam =mynode()
      nodes = numnodes()

c ------( Fourier transforms in the x-direction )----------------------

      do  100  istage = 1, nl1, 2
        ibase  =  n2*(istage-1)
        do  110  k = iam+1, n, nodes
          do  110  i = 1, n2
            i2  =  2*i
            i1  =  i2 - 1
            do  110  j = 1, n
              cr(i1,j,k)  =  fr(i,j,k)  +  fr(i+n2,j,k)
              ci(i1,j,k)  =  fi(i,j,k)  +  fi(i+n2,j,k)
              temp1       =  fr(i,j,k)  -  fr(i+n2,j,k)
              temp2       =  fi(i,j,k)  -  fi(i+n2,j,k)
              cr(i2,j,k)  =  temp1*cn(i+ibase)  -  temp2*sn(i+ibase)
              ci(i2,j,k)  =  temp2*cn(i+ibase)  +  temp1*sn(i+ibase)
110     continue
        if( (leven.eq.0) .and. (istage.eq.nl1) )  go to 100
        ibase  =  n2*istage
        do  120  k = iam+1, n, nodes
          do  120  i = 1, n2
            i2  =  2*i
            i1  =  i2 - 1
            do  120  j = 1, n
              fr(i1,j,k)  =  cr(i,j,k)  +  cr(i+n2,j,k)
              fi(i1,j,k)  =  ci(i,j,k)  +  ci(i+n2,j,k)
              temp1       =  cr(i,j,k)  -  cr(i+n2,j,k)
              temp2       =  ci(i,j,k)  -  ci(i+n2,j,k)
              fr(i2,j,k)  =  temp1*cn(i+ibase)  -  temp2*sn(i+ibase)
              fi(i2,j,k)  =  temp2*cn(i+ibase)  +  temp1*sn(i+ibase)
120     continue
100   continue
      if( leven .ne. 0 )  then
        do  200  k = iam+1, n, nodes
          do  200  j = 1, n
            do  200  i = 1, np
              cr(i,j,k)  =  fr(i,j,k)
              ci(i,j,k)  =  fi(i,j,k)
200     continue
      endif
      do  300  k = iam+1, n, nodes
        do  300  i = 1, n2
          i2  =  2*i
          i1  =  i2 - 1
          do  300  j = 1, n
            fr(ibr(i1),j,k)  =  cr(i,j,k)  +  cr(i+n2,j,k)
            fi(ibr(i1),j,k)  =  ci(i,j,k)  +  ci(i+n2,j,k)
            fr(ibr(i2),j,k)  =  cr(i,j,k)  -  cr(i+n2,j,k)
            fi(ibr(i2),j,k)  =  ci(i,j,k)  -  ci(i+n2,j,k)
300   continue
```

Fig. 4.3 Paralleled code for FFT (vdft3b) on the Paragon XP/S (to be continued)

```
c ------( Fourier transforms in the y-direction )----------------------

      do  400  istage = 1, nl1, 2
        jbase  =  n2*(istage-1)
        do  410  k = iam+1, n, nodes
          do  410  j = 1, n2
            j2  =  2*j
            j1  =  j2 - 1
            do  410  i = 1, n
              cr(i,j1,k)  =  fr(i,j,k)  +  fr(i,j+n2,k)
              ci(i,j1,k)  =  fi(i,j,k)  +  fi(i,j+n2,k)
              temp1       =  fr(i,j,k)  -  fr(i,j+n2,k)
              temp2       =  fi(i,j,k)  -  fi(i,j+n2,k)
              cr(i,j2,k)  =  temp1*cn(j+jbase)  -  temp2*sn(j+jbase)
              ci(i,j2,k)  =  temp2*cn(j+jbase)  +  temp1*sn(j+jbase)
  410   continue
        if( (leven.eq.0) .and. (istage.eq.nl1) )  go to 400
        jbase  =  n2*istage
        do  420  k = iam+1, n, nodes
          do  420  j = 1, n2
            j2  =  2*j
            j1  =  j2 - 1
            do  420  i = 1, n
              fr(i,j1,k)  =  cr(i,j,k)  +  cr(i,j+n2,k)
              fi(i,j1,k)  =  ci(i,j,k)  +  ci(i,j+n2,k)
              temp1       =  cr(i,j,k)  -  cr(i,j+n2,k)
              temp2       =  ci(i,j,k)  -  ci(i,j+n2,k)
              fr(i,j2,k)  =  temp1*cn(j+jbase)  -  temp2*sn(j+jbase)
              fi(i,j2,k)  =  temp2*cn(j+jbase)  +  temp1*sn(j+jbase)
  420   continue
  400 continue
      if( leven .ne. 0 )  then
         do  500  k = iam+1, n, nodes
           do  500  j = 1, n
             do  500  i = 1, np
               cr(i,j,k)  =  fr(i,j,k)
               ci(i,j,k)  =  fi(i,j,k)
  500    continue
      endif
      do  600  k = iam+1, n, nodes
        do  600  j = 1, n2
          j2  =  2*j
          j1  =  j2 - 1
          do  600  i = 1, n
            fr(i,ibr(j1),k)  =  cr(i,j,k)  +  cr(i,j+n2,k)
            fi(i,ibr(j1),k)  =  ci(i,j,k)  +  ci(i,j+n2,k)
            fr(i,ibr(j2),k)  =  cr(i,j,k)  -  cr(i,j+n2,k)
            fi(i,ibr(j2),k)  =  ci(i,j,k)  -  ci(i,j+n2,k)
  600 continue
```

Fig. 4.3 Paralleled code for FFT (vdft3b) on the Paragon XP/S (to be continued)

```
c -----( Fourier transforms in the z-direction )----------------------

c transfer data (all to all) (z) ccccccccccccccccccccccccccccccccccc
      is = np * n * 4
      indiam = iam + 1
      do 10 k=1,n
           if (k .eq. indiam) then
                indiam = indiam + nodes
                call csend(k,fr(1,1,k),is,-1,mypid())
                call csend(k,fi(1,1,k),is,-1,mypid())
           else
                call crecv(k,fr(1,1,k),is)
                call crecv(k,fi(1,1,k),is)
           endif
 10   continue
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      do 101 j=iam+1,n,nodes
           do 101 k=1,n
                do 101 i=1,np
                     cr(i,k,j) = fr(i,j,k)
                     ci(i,k,j) = fi(i,j,k)
 101  continue
      call gsync()
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

*vdir novector
      do  700  istage = 1, nl1, 2
        kbase  =  n2*(istage-1)
*vocl loop,novrec
*vdir nodep
        do  710  j = iam+1,  n,  nodes
          do  710  k = 1,  n2
            k2       =  2*k
            k1       =  k2 - 1
            do  710  i = 1,  n
              fr(i,k1,j)  =  cr(i,k,j)  +  cr(i,k+n2,j)
              fi(i,k1,j)  =  ci(i,k,j)  +  ci(i,k+n2,j)
              temp1       =  cr(i,k,j)  -  cr(i,k+n2,j)
              temp2       =  ci(i,k,j)  -  ci(i,k+n2,j)
              fr(i,k2,j)  =  temp1*cn(k+kbase)  -  temp2*sn(k+kbase)
              fi(i,k2,j)  =  temp2*cn(k+kbase)  +  temp1*sn(k+kbase)
 710    continue
        if( (leven.eq.0) .and. (istage.eq.nl1) )  go to 700
        kbase  =  n2*istage
        do  720  j = iam+1,  n,  nodes
          do  720  k = 1,  n2
            k2       =  2*k
            k1       =  k2 - 1
            do  720  i = 1,  n
              cr(i,k1,j)  =  fr(i,k,j)  +  fr(i,k+n2,j)
              ci(i,k1,j)  =  fi(i,k,j)  +  fi(i,k+n2,j)
              temp1       =  fr(i,k,j)  -  fr(i,k+n2,j)
              temp2       =  fi(i,k,j)  -  fi(i,k+n2,j)
              cr(i,k2,j)  =  temp1*cn(k+kbase)  -  temp2*sn(k+kbase)
              ci(i,k2,j)  =  temp2*cn(k+kbase)  +  temp1*sn(k+kbase)
 720    continue
 700  continue
```

Fig. 4.3 Paralleled code for FFT (vdft3b) on the Paragon XP/S (to be continued)

```
      if( leven .ne. 0 )  then
         do  800  j = iam+1, n, nodes
           do  800  k = 1, n
             do  800  i = 1, np
               fr(i,k,j)  =  cr(i,k,j)
c              fi(i,k,j)  =  ci(i,k,j)
 800     continue
      endif
*vocl loop,novrec
*vdir nodep
      do  900  j = iam+1, n, nodes
        do  900  k = 1, n2
          k2  =  2*k
          k1  =  k2 - 1
          do  900  i = 1, n
            cr(i,ibr(k1),j)  =  fr(i,k,j)  +  fr(i,k+n2,j)
c           ci(i,ibr(k1),j)  =  fi(i,k,j)  +  fi(i,k+n2,j)
            cr(i,ibr(k2),j)  =  fr(i,k,j)  -  fr(i,k+n2,j)
c           ci(i,ibr(k2),j)  =  fi(i,k,j)  -  fi(i,k+n2,j)
 900  continue

c transfer data (all to all) (z) ccccccccccccccccccccccccccccccccccc
      is = np * n * 4
      indiam = iam + 1
      do 20 k=1,n
           if (k .eq. indiam) then
                indiam = indiam + nodes
                call csend(k,cr(1,1,k),is,-1,mypid())
           else
                call crecv(k,cr(1,1,k),is)
           endif
 20   continue
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
      do 102 k=1,n
           do 102 j=1,n
                do 102 i=1,np
                     fr(i,j,k) = cr(i,k,j)
 102  continue
      call gsync()
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

c ---------------------------------------------------------------

      return
      end
```

Fig. 4.3 Paralleled code for FFT (vdft3b) on the Paragon XP/S

```
      call  putksp( n, np, nk, nk2, ur(1,1,1,1), upr(1,1,1,1) )
      call  putksp( n, np, nk, nk2, ui(1,1,1,1), wki )
      call  vdft3b( n, np, n2, nl1, upr(1,1,1,1), wki, w2r, w2i,
     &              cn, sn, ibr )
      call  putksp( n, np, nk, nk2, ur(1,1,1,2), upr(1,1,1,2) )
      call  putksp( n, np, nk, nk2, ui(1,1,1,2), wki )
      call  vdft3b( n, np, n2, nl1, upr(1,1,1,2), wki, w2r, w2i,
     &              cn, sn, ibr )
      call  putksp( n, np, nk, nk2, ur(1,1,1,3), upr(1,1,1,3) )
      call  putksp( n, np, nk, nk2, ui(1,1,1,3), wki )
      call  vdft3b( n, np, n2, nl1, upr(1,1,1,3), wki, w2r, w2i,
     &              cn, sn, ibr )
```

並列化のための変更(フーリエ空間から実空間への変換の場合)

```
      call  putksp( ur(1,1,1,1), ui(1,1,1,1) )
      call  vdft3b(1)
      call  putksp( ur(1,1,1,2), ui(1,1,1,2) )
      call  vdft3b(2)
      call  putksp( ur(1,1,1,3), ui(1,1,1,3) )
      call  vdft3b(3)
```

```
      call  convol( n,    np,   n2,   nk,   nk2,  nl1,
     &              w11r, w11i, upr(1,1,1,1), upr(1,1,1,1),
     &              wkr,  wki,  w2r,  w2i,  cn,   sn,   ibr )
      call  convol( n,    np,   n2,   nk,   nk2,  nl1,
     &              w12r, w12i, upr(1,1,1,1), upr(1,1,1,2),
     &              wkr,  wki,  w2r,  w2i,  cn,   sn,   ibr )
      call  convol( n,    np,   n2,   nk,   nk2,  nl1,
     &              w13r, w13i, upr(1,1,1,1), upr(1,1,1,3),
     &              wkr,  wki,  w2r,  w2i,  cn,   sn,   ibr )
      call  convol( n,    np,   n2,   nk,   nk2,  nl1,
     &              w22r, w22i, upr(1,1,1,2), upr(1,1,1,2),
     &              wkr,  wki,  w2r,  w2i,  cn,   sn,   ibr )
      call  convol( n,    np,   n2,   nk,   nk2,  nl1,
     &              w23r, w23i, upr(1,1,1,2), upr(1,1,1,3),
     &              wkr,  wki,  w2r,  w2i,  cn,   sn,   ibr )
      call  convol( n,    np,   n2,   nk,   nk2,  nl1,
     &              w33r, w33i, upr(1,1,1,3), upr(1,1,1,3),
     &              wkr,  wki,  w2r,  w2i,  cn,   sn,   ibr )
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      subroutine  convol( n,    np,   n2,   nk,   nk2,  nl1,
     &                    wrr,  wri,  v1,   v2,
     &                    wkr,  wki,  w2r,  w2i,  cn,   sn,   ibr )
      dimension   wrr(nk,nk,nk),   wri(nk,nk,nk)
      dimension   v1(np,n,n),      v2(np,n,n),      wkr(np,n,n)
      dimension   wki(np,n,n),     w2r(np,n,n),     w2i(np,n,n)
      dimension   cn(n2*nl1),      sn(n2*nl1),      ibr(n*nl1)
      do  100  k = 1,  n
        do  100  j = 1,  n
          do  100  i = 1,  np
            wkr(i,j,k)  =  v1(i,j,k)*v2(i,j,k)
100   continue
      do  200  k = 1,  n
        do  200  j = 1,  n
          do  200  i = 1,  np
            wki(i,j,k)  =  0.0d0
200   continue
      call  vdft3f( n, np, n2, nl1, wkr, wki, w2r, w2i, cn, sn, ibr )
      call  getksp( n, np, nk, nk2, wkr, wrr )
      call  getksp( n, np, nk, nk2, wki, wri )
      return
      end
```

並列化のための変更(実空間からフーリエ空間への変換の場合)

```
      call  vdft3f( 1, 1 )
      call  getksp( w11r, w11i )
      call  vdft3f( 1, 2 )
      call  getksp( w12r, w12i )
      call  vdft3f( 1, 3 )
      call  getksp( w13r, w13i )
      call  vdft3f( 2, 2 )
      call  getksp( w22r, w22i )
      call  vdft3f( 2, 3 )
      call  getksp( w23r, w23i )
      call  vdft3f( 3, 3 )
      call  getksp( w33r, w33i )
```

Fig. 4.4 Paralleled code for subroutines

```
      subroutine  vdftzb(iv)

      common  /cmvdft/ leven,  oncube
      include  './Include/cmparm'
      common /cupr/ upr(np,n,n,3)
      common /cw/ wkr(np,n,n),wki(np,n,n),w2r(np,n,n),w2i(np,n,n)
      common /ccsi/ cn(n2*nl1),sn(n2*nl1),ibr(n*nl1)
      include  './Include/nofpe'
!xocl processor pe(npe)
!xocl subprocessor spe(npe)=pe(1:npe)
!xocl index partition ipl=(spe,part=cyclic)
!xocl global wkr(:,:,/ipl),wki(:,:,/ipl),w2r(:,:,/ipl),w2i(:,:,/ipl)
!xocl local fr(:,:,/ipl),fi(:,:,/ipl),cr(:,:,/ipl),ci(:,:,/ipl)
      dimension  fr(np,n,n),  fi(np,n,n),  cr(np,n,n),  ci(np,n,n)
      equivalence (fr,wkr),(fi,wki),(cr,w2r),(ci,w2i)
!xocl local ffr(:,/ipl,:),ffi(:,/ipl,:),ccr(:,/ipl,:),cci(:,/ipl,:)
      dimension  ffr(np,n,n),ffi(np,n,n),ccr(np,n,n),cci(np,n,n)

c -----( Fourier transforms in the z-direction )--------------------

ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
!xocl spread move :,/ipl
      do 101 k=1,n
           do 101 j=1,n
                do 101 i=1,np
                     ffr(i,j,k)  = wkr(i,j,k)
                     ffi(i,j,k)  = wki(i,j,k)
 101  continue
!xocl end spread (b1)
!xocl movewait (b1)
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

      do  700  istage = 1,  nl1,  2
        kbase  =  n2*(istage-1)
*vocl loop,novrec
!xocl spread do /ipl
        do  710  j = 1,  n
          do  710  k = 1,  n2
            k2      =  2*k
            k1      =  k2 - 1
            do  710  i = 1,  n
              ccr(i,j,k1)  =  ffr(i,j,k)  +  ffr(i,j,k+n2)
              cci(i,j,k1)  =  ffi(i,j,k)  +  ffi(i,j,k+n2)
              temp1       =  ffr(i,j,k)  -  ffr(i,j,k+n2)
              temp2       =  ffi(i,j,k)  -  ffi(i,j,k+n2)
              ccr(i,j,k2)  =  temp1*cn(k+kbase)  -  temp2*sn(k+kbase)
              cci(i,j,k2)  =  temp2*cn(k+kbase)  +  temp1*sn(k+kbase)
 710    continue
!xocl end spread
```

Fig. 4.5 Paralleled code for FFT on the VPP500 (vdftzb ⊂ vdft3b) (to be continued)

```
       if( (leven.eq.0) .and. (istage.eq.nl1) )  go to 700
       kbase = n2*istage
*vocl loop,novrec
!xocl spread do /ipl
       do 720 j = 1, n
         do 720 k = 1, n2
           k2      = 2*k
           k1      = k2 - 1
           do 720 i = 1, n
             ffr(i,j,k1) = ccr(i,j,k) + ccr(i,j,k+n2)
             ffi(i,j,k1) = cci(i,j,k) + cci(i,j,k+n2)
             temp1       = ccr(i,j,k) - ccr(i,j,k+n2)
             temp2       = cci(i,j,k) - cci(i,j,k+n2)
             ffr(i,j,k2) = temp1*cn(k+kbase) - temp2*sn(k+kbase)
             ffi(i,j,k2) = temp2*cn(k+kbase) + temp1*sn(k+kbase)
 720   continue
!xocl end spread
 700  continue
      if( leven .ne. 0 )  then
!xocl spread do /ipl
        do 800 j = 1, n
          do 800 k = 1, n
            do 800 i = 1, np
              ccr(i,j,k) = ffr(i,j,k)
c             cci(i,j,k) = ffi(i,j,k)
 800    continue
!xocl end spread
      endif
*vocl loop,novrec
!xocl spread do /ipl
      do 900 j = 1, n
        do 900 k = 1, n2
          k2 = 2*k
          k1 = k2 - 1
          do 900 i = 1, n
            ffr(i,j,ibr(k1)) = ccr(i,j,k) + ccr(i,j,k+n2)
c           ffi(i,j,ibr(k1)) = cci(i,j,k) + cci(i,j,k+n2)
            ffr(i,j,ibr(k2)) = ccr(i,j,k) - ccr(i,j,k+n2)
c           ffi(i,j,ibr(k2)) = cci(i,j,k) - cci(i,j,k+n2)
 900  continue
!xocl end spread

ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
!xocl spread do /ipl
      do 102 j=1,n
           do 102 k=1,n
                do 102 i=1,np
                     upr(i,j,k,iv) = ffr(i,j,k)
 102  continue
!xocl end spread
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

c ---------------------------------------------------------------------

      return
      end
```

Fig. 4.5 Paralleled code for FFT on the VPP500 (vdftzb ⊂ vdft3b)

```
   Monte-UX  R5.10        FORTRAN77/M4   Rev.094              DATE:  Wed Nov 29 17:44:15 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b
FORMAT LIST

   ELN   ILN    LABEL REF.       LOOP       FORTRAN STATEMENT

000001     1                                    subroutine  vdft3b( n,  np,  n2,  nl1,  fr,  fi,  cr,  ci,  cn,  sn,  ibr )
000002     2                                    c                                                                              2
000003     3                                    c...Translated by fopp       4.07K16 17:44:12  11/29/95
000004     4                                    c...Switches: -eadejlpuvx18 -dchimr07 -ech0 -gi
000005     5                                    dimension  fr(np,n,n),  fi(np,n,n),  cr(np,n,n),  ci(np,n,n)
000006     6                                    dimension  cn(n2*nl1),  sn(n2*nl1),  ibr(n*nl1)
000007     7                                    common  /cmvdft/ leven, oncube
000008     8                                    c                                                                              6
000009     9                                    c -----( Fourier transforms in the x-direction )----------------------------    7
000010    10                                    c                                                                              8
000011    11                                    *vdir novector
000012    12                                    integer i3,  i4,  i5,  i6
000013    13                                    *PDIR RESERVE
000014    14    22                              do  100  istage = 1,  nl1,  2
000015    15                                    c                                                                             11
000016    16                1---------->        !  ibase  =  n2*(istage-1)
000017    17                !                   !  call VDFT3B$1 (n,  n2,  ibase,  np,  fr,  cr,  fi,  ci,  nl1,  cn,  sn)
000018    18    22          !                   !  if( (leven.eq.0) .and.  (istage.eq.nl1) )  go to 100
000019    19                !                   !  c                                                                          28
000020    20                !                   !  ibase  =  n2*istage
000021    21                !                   !  call VDFT3B$2 (n,  n2,  ibase,  np,  cr,  fr,  ci,  fi,  nl1,  cn,  sn)
000022    22                1----------  100    continue
000023    23                                    c                                                                             45
000024    24                                    c                                                                             46
000025    25                                    if( leven .ne.  0 )  then
000026    26                                    !  call VDFT3B$3 (n,  np,  fr,  cr,  fi,  ci)
000027    27                                    endif
000028    28                                    c                                                                             55
000029    29                                    c                                                                             56
000030    30                                    call VDFT3B$4 (n,  n2,  np,  cr,  nl1,  ibr,  fr,  ci,  fi)
000031    31    39                              do  400  istage = 1,  nl1,  2
000032    32                                    c                                                                             74
000033    33                2---------->        !  jbase  =  n2*(istage-1)
000034    34                !                   !  call VDFT3B$5 (n,  n2,  jbase,  np,  fr,  cr,  fi,  ci,  nl1,  cn,  sn)
000035    35    39          !                   !  if( (leven.eq.0) .and.  (istage.eq.nl1) )  go to 400
000036    36                !                   !  c                                                                          91
000037    37                !                   !  jbase  =  n2*istage
000038    38                !                   !  call VDFT3B$6 (n,  n2,  jbase,  np,  cr,  fr,  ci,  fi,  nl1,  cn,  sn)
000039    39                2----------  400    continue
000040    40                                    c                                                                            108
000041    41                                    c                                                                            109
000042    42                                    if( leven .ne.  0 )  then
000043    43                                    !  call VDFT3B$7 (n,  np,  fr,  cr,  fi,  ci)
000044    44                                    endif
000045    45                                    c                                                                            118
000046    46                                    c                                                                            119
000047    47                                    call VDFT3B$8 (n,  n2,  np,  cr,  nl1,  ibr,  fr,  ci,  fi)
000048    48    56                              do  700  istage = 1,  nl1,  2
000049    49                                    c                                                                            138
000050    50                3---------->        !  kbase  =  n2*(istage-1)
000051    51                !                   !  call VDFT3B$9 (n2,  n,  kbase,  np,  fr,  cr,  fi,  ci,  nl1,  cn,  sn)
000052    52    56          !                   !  if( (leven.eq.0) .and.  (istage.eq.nl1) )  go to 700
000053    53                !                   !  c                                                                          155
000054    54                !                   !  kbase  =  n2*istage
000055    55                !                   !  call VDFT3B$10 (n2,  n,  kbase,  np,  cr,  fr,  ci,  fi,  nl1,  cn,  sn)
000056    56                3----------  700    continue
000057    57                                    c                                                                            172
000058    58                                    c                                                                            173
000059    59                                    if( leven .ne.  0 )  then
000060    60                                    !  call VDFT3B$11 (n,  np,  fr,  cr,  fi,  ci)
000061    61                                    endif
000062    62                                    c                                                                            182
000063    63                                    c                                                                            183
000064    64                                    call VDFT3B$12 (n2,  n,  np,  cr,  nl1,  ibr,  fr,  ci,  fi)
000065    65                                    *PDIR RELEASE
000066    66                                    stop
000067    67                                    *PDIR RELEASE
000068    68                                    return
000069    69                                    end
```

Fig. 4.6 Paralleled code for FFT on the Monte-4 by autotasking (vdft3b) (to be continued)

```
   Monte-UX  R5.10          FORTRAN77/M4   Rev.094                  DATE:  Wed Nov 29 17:44:15 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b
DIAGNOSTIC LIST

      ELN      LEVEL    NO.   DIAGNOSTIC MESSAGE

    000014       VEC     2 : Unvectorized DO loop
    000031       VEC     2 : Unvectorized DO loop
    000048       VEC     2 : Unvectorized DO loop
    000068        **   111 : This statement can never be reached  ( PT=7 )
                  **   107 : Variable or array not used, therefore not allocated : i6
                  **   107 : Variable or array not used, therefore not allocated : i4
                  **   107 : Variable or array not used, therefore not allocated : i5
                  **   107 : Variable or array not used, therefore not allocated : i3

   Monte-UX  R5.10          FORTRAN77/M4   Rev.094                  DATE:  Wed Nov 29 17:44:15 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$1
FORMAT LIST

  ELN   ILN    LABEL REF.        LOOP       FORTRAN STATEMENT

000070     1                                      SUBROUTINE VDFT3B$1(n,n2,ibase,np,fr,cr,fi,ci,n11,cn,sn)
000071     2                                      INTEGER n, n2, ibase, np, n11
000072     3                                      REAL fr(np,n,n), cr(np,n,n), fi(np,n,n), ci(np,n,n), cn(n2*n11),
000073     4                                     1   sn(n2*n11)
000074     5                                      INTEGER k, i, i2, i1, j, i5, i6
000075     6                                      REAL temp1, temp2
000076     7                                      *PDIR PARDO FOR = 4
000077     8                                      *VDIR NODEP
000078     9                                      do k = 1, n
000079    10                                      !  *VDIR    NOASSUME
000080    11                  P---------->        !  do i5 = 0, n2 - 1, maxv1()
000081    12                  ! V-------->        !  !  i6 = min0(n2 - i5,maxv1())
000082    13                  ! !                 !  !  do j = 1, n
000083    14                  ! !                 !  !  !  *VDIR          NODEP
000084    15                  ! !                 !  !  !  *VDIR          SHORTLOOP
000085    16                  ! ! 5------->       !  !  !  do i = 1, i6
000086    17                  ! ! ! V----->       !  !  !  !  cr(2*(i5+i)-1,j,k) = fr(i5+i,j,k) + fr(n2+i5+i,j,k)
000087    18                  ! ! ! !             !  !  !  !  ci(2*(i5+i)-1,j,k) = fi(i5+i,j,k) + fi(n2+i5+i,j,k)
000088    19                  ! ! ! !             !  !  !  !  temp1 = fr(i5+i,j,k) - fr(n2+i5+i,j,k)
000089    20                  ! ! ! !             !  !  !  !  temp2 = fi(i5+i,j,k) - fi(n2+i5+i,j,k)
000090    21                  ! ! ! !             !  !  !  !  cr(2*(i5+i),j,k) = temp1*cn(ibase+i5+i) - temp2*sn(
000091    22                  ! ! ! !            1!  !  !  !     ibase+i5+i)
000092    23                  ! ! ! !             !  !  !  !  ci(2*(i5+i),j,k) = temp2*cn(ibase+i5+i) + temp1*sn(
000093    24                  ! ! ! !            1!  !  !  !     ibase+i5+i)
000094    25                  ! ! ! V------       !  !  !  end do
000095    26                  ! ! 5--------       !  !  end do
000096    27                  ! V----------       !  end do
000097    28                  P------------       end do
000098    29                                      return
000099    30                                      end

   Monte-UX  R5.10          FORTRAN77/M4   Rev.094                  DATE:  Wed Nov 29 17:44:15 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$1
DIAGNOSTIC LIST

      ELN      LEVEL    NO.   DIAGNOSTIC MESSAGE

    000078       PAR     1 : Parallelized by DO index k
    000080       VEC     1 : Vectorized by DO index i5
    000085       VEC     1 : Vectorized by DO index i
                  **   107 : Variable or array not used, therefore not allocated : i2
                  **   107 : Variable or array not used, therefore not allocated : i1
```

Fig. 4.6 Paralleled code for FFT on the Monte-4 by autotasking (vdft3b) (to be continued)

```
   Monte-UX  R5.10          FORTRAN77/M4   Rev.094                DATE:  Wed Nov 29 17:44:16 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$2
FORMAT LIST

  ELN   ILN     LABEL REF.        LOOP       FORTRAN STATEMENT

000100     1                                      SUBROUTINE VDFT3B$2(n,n2,ibase,np,cr,fr,ci,fi,nl1,cn,sn)
000101     2                                      INTEGER n, n2, ibase, np, nl1
000102     3                                      REAL cr(np,n,n), fr(np,n,n), ci(np,n,n), fi(np,n,n), cn(n2*nl1),
000103     4                                     1   sn(n2*nl1)
000104     5                                      INTEGER k, i, i2, i1, j, i3, i4
000105     6                                      REAL temp1, temp2
000106     7                                      *PDIR PARDO FOR = 4
000107     8                                      *VDIR NODEP
000108     9                                      do k = 1, n
000109    10                                      !  *VDIR    NOASSUME
000110    11                  P---------->        !  do i3 = 0, n2 - 1, maxvl()
000111    12                  ! V-------->        !  !  i4 = min0(n2 - i3,maxvl())
000112    13                  ! !                 !  !  do j = 1, n
000113    14                  ! !                 !  !  !  *VDIR          NODEP
000114    15                  ! !                 !  !  !  *VDIR          SHORTLOOP
000115    16                  ! ! 5------->       !  !  !  do i = 1, i4
000116    17                  ! ! ! V----->       !  !  !  !  fr(2*(i3+i)-1,j,k) = cr(i3+i,j,k) + cr(n2+i3+i,j,k)
000117    18                  ! ! ! !             !  !  !  !  fi(2*(i3+i)-1,j,k) = ci(i3+i,j,k) + ci(n2+i3+i,j,k)
000118    19                  ! ! ! !             !  !  !  !  temp1 = cr(i3+i,j,k) - cr(n2+i3+i,j,k)
000119    20                  ! ! ! !             !  !  !  !  temp2 = ci(i3+i,j,k) - ci(n2+i3+i,j,k)
000120    21                  ! ! ! !             !  !  !  !  fr(2*(i3+i),j,k) = temp1*cn(ibase+i3+i) - temp2*sn(
000121    22                  ! ! ! !            1!  !  !  !     ibase+i3+i)
000122    23                  ! ! ! !             !  !  !  !  fi(2*(i3+i),j,k) = temp2*cn(ibase+i3+i) + temp1*sn(
000123    24                  ! ! ! !            1!  !  !  !     ibase+i3+i)
000124    25                  ! ! ! V------       !  !  !  end do
000125    26                  ! ! 5--------       !  !  end do
000126    27                  ! V----------       !  end do
000127    28                  P----------         end do
000128    29                                      return
000129    30                                      end

   Monte-UX  R5.10          FORTRAN77/M4   Rev.094                DATE:  Wed Nov 29 17:44:16 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$2
DIAGNOSTIC LIST

       ELN      LEVEL    NO.   DIAGNOSTIC MESSAGE

    000108       PAR      1 : Parallelized by DO index k
    000110       VEC      1 : Vectorized by DO index i3
    000115       VEC      1 : Vectorized by DO index i
                  **    107 : Variable or array not used, therefore not allocated : i2
                  **    107 : Variable or array not used, therefore not allocated : i1

   Monte-UX  R5.10          FORTRAN77/M4   Rev.094                DATE:  Wed Nov 29 17:44:16 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$3
FORMAT LIST

  ELN   ILN     LABEL REF.        LOOP       FORTRAN STATEMENT

000130     1                                      SUBROUTINE VDFT3B$3 (n, np, fr, cr, fi, ci)
000131     2                                      INTEGER n, np
000132     3                                      REAL fr(np,n,n), cr(np,n,n), fi(np,n,n), ci(np,n,n)
000133     4                                      INTEGER k, j, i
000134     5                                      *PDIR PARDO FOR = 4
000135     6                                      *VDIR NODEP
000136     7                                      *VDIR NOASSUME
000137     8                                      do k = 1, n*n*np
000138     9                  PV---------->       !  cr(k,1,1) = fr(k,1,1)
000139    10                  !                   !  ci(k,1,1) = fi(k,1,1)
000140    11                  PV----------        end do
000141    12                                      return
000142    13                                      end
```

Fig. 4.6 Paralleled code for FFT on the Monte-4 by autotasking (vdft3b) (to be continued)

```
   Monte-UX  R5.10         FORTRAN77/M4   Rev.094                DATE:  Wed Nov 29 17:44:16 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$3
DIAGNOSTIC LIST

       ELN      LEVEL    NO.   DIAGNOSTIC MESSAGE

    000137        VEC      1 : Vectorized by DO index k
    000137        PAR      1 : Parallelized by DO index k
                  **     107 : Variable or array not used, therefore not allocated : j
                  **     107 : Variable or array not used, therefore not allocated : i

   Monte-UX  R5.10         FORTRAN77/M4   Rev.094                DATE:  Wed Nov 29 17:44:17 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$4
FORMAT LIST

  ELN   ILN     LABEL REF.        LOOP       FORTRAN STATEMENT

000143     1                                     SUBROUTINE VDFT3B$4 (n, n2, np, cr, nl1, ibr, fr, ci, fi)
000144     2                                     INTEGER n, n2, np, nl1, ibr(n*nl1)
000145     3                                     REAL cr(np,n,n), fr(np,n,n), ci(np,n,n), fi(np,n,n)
000146     4                                     INTEGER k, i, i2, i1, j
000147     5                                     *PDIR PARDO FOR = 4
000148     6                                     *VDIR NODEP
000149     7                                     *
000150     8                                     *
000151     9     19                              do 300 k = 1, n
000152    10     19            P------------>    !  do 300 i = 1, n2
000153    11                  ! V---------->    !  !  i2 = 2*i
000154    12                  ! !               !  !  i1 = i2 - 1
000155    13                  ! !               !  !  *VDIR       NODEP
000156    14     19           ! !               !  !  do 300 j = 1, n
000157    15                  ! ! V-------->    !  !  !  fr(ibr(i1),j,k) = cr(i,j,k) + cr(i+n2,j,k)
000158    16                  ! ! !             !  !  !  fi(ibr(i1),j,k) = ci(i,j,k) + ci(i+n2,j,k)
000159    17                  ! ! !             !  !  !  fr(ibr(i2),j,k) = cr(i,j,k) - cr(i+n2,j,k)
000160    18                  ! ! !             !  !  !  fi(ibr(i2),j,k) = ci(i,j,k) - ci(i+n2,j,k)
000161    19                  P-V-V--------  300 continue
000162    20                                     return
000163    21                                     end

   Monte-UX  R5.10         FORTRAN77/M4   Rev.094                DATE:  Wed Nov 29 17:44:17 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$4
DIAGNOSTIC LIST

       ELN      LEVEL    NO.   DIAGNOSTIC MESSAGE

    000151        PAR      1 : Parallelized by DO index k
    000152        VEC      1 : Vectorized by DO index i
    000156        VEC      1 : Vectorized by DO index j

   Monte-UX  R5.10         FORTRAN77/M4   Rev.094                DATE:  Wed Nov 29 17:44:17 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$5
FORMAT LIST

  ELN   ILN     LABEL REF.        LOOP       FORTRAN STATEMENT

000164     1                                     SUBROUTINE VDFT3B$5(n,n2,jbase,np,fr,cr,fi,ci,nl1,cn,sn)
000165     2                                     INTEGER n, n2, jbase, np, nl1
000166     3                                     REAL fr(np,n,n), cr(np,n,n), fi(np,n,n), ci(np,n,n), cn(n2*nl1),
000167     4                                    1   sn(n2*nl1)
000168     5                                     INTEGER k, j, j2, j1, i
000169     6                                     REAL templ, temp2
000170     7                                     *PDIR PARDO FOR = 4
000171     8                                     *VDIR NODEP
000172     9     21                              do 410 k = 1, n
000173    10     21            P------------>    !  do 410 j = 1, n2
000174    11                  ! V---------->    !  !  j2 = 2*j
000175    12                  ! !               !  !  j1 = j2 - 1
000176    13                  ! !               !  !  *VDIR       NODEP
000177    14     21           ! !               !  !  do 410 i = 1, n
000178    15                  ! ! V-------->    !  !  !  cr(i,j1,k) = fr(i,j,k) + fr(i,j+n2,k)
000179    16                  ! ! !             !  !  !  ci(i,j1,k) = fi(i,j,k) + fi(i,j+n2,k)
000180    17                  ! ! !             !  !  !  templ = fr(i,j,k) - fr(i,j+n2,k)
000181    18                  ! ! !             !  !  !  temp2 = fi(i,j,k) - fi(i,j+n2,k)
000182    19                  ! ! !             !  !  !  cr(i,j2,k) = templ*cn(j+jbase) - temp2*sn(j+jbase)
000183    20                  ! ! !             !  !  !  ci(i,j2,k) = temp2*cn(j+jbase) + templ*sn(j+jbase)
000184    21                  P-V-V--------  410 continue
000185    22                                     return
000186    23                                     end
```

Fig. 4.6 Paralleled code for FFT on the Monte-4 by autotasking (vdft3b) (to be continued)

```
   Monte-UX  R5.10          FORTRAN77/M4   Rev.094                DATE:  Wed Nov 29 17:44:17 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$5
DIAGNOSTIC LIST

       ELN      LEVEL    NO.   DIAGNOSTIC MESSAGE

    000172        PAR      1 : Parallelized by DO index k
    000173        VEC      1 : Vectorized by DO index j
    000177        VEC      1 : Vectorized by DO index i

   Monte-UX  R5.10          FORTRAN77/M4   Rev.094                DATE:  Wed Nov 29 17:44:18 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$6
FORMAT LIST

  ELN   ILN     LABEL REF.        LOOP        FORTRAN STATEMENT

000187     1                                      SUBROUTINE VDFT3B$6(n,n2,jbase,np,cr,fr,ci,fi,n11,cn,sn)
000188     2                                      INTEGER n,  n2,  jbase,  np,  n11
000189     3                                      REAL cr(np,n,n),  fr(np,n,n),  ci(np,n,n),  fi(np,n,n),  cn(n2*n11),
000190     4                                     1   sn(n2*n11)
000191     5                                      INTEGER k,  j,  j2,  j1,  i
000192     6                                      REAL templ,  temp2
000193     7                                      *PDIR PARDO FOR = 4
000194     8                                      *VDIR NODEP
000195     9     21                               do 420 k = 1,  n
000196    10     21             P------------>    !  do 420 j = 1,  n2
000197    11                    ! V---------->    !  !  j2 = 2*j
000198    12                    ! !               !  !  j1 = j2 - 1
000199    13                    ! !               !  !  *VDIR       NODEP
000200    14     21             ! !               !  !  do 420 i = 1,  n
000201    15                    ! ! V-------->    !  !  !  fr(i,j1,k) = cr(i,j,k) + cr(i,j+n2,k)
000202    16                    ! ! !             !  !  !  fi(i,j1,k) = ci(i,j,k) + ci(i,j+n2,k)
000203    17                    ! ! !             !  !  !  templ = cr(i,j,k) - cr(i,j+n2,k)
000204    18                    ! ! !             !  !  !  temp2 = ci(i,j,k) - ci(i,j+n2,k)
000205    19                    ! ! !             !  !  !  fr(i,j2,k) = templ*cn(j+jbase) - temp2*sn(j+jbase)
000206    20                    ! ! !             !  !  !  fi(i,j2,k) = temp2*cn(j+jbase) + templ*sn(j+jbase)
000207    21                    P-V-V---------  420 continue
000208    22                                      return
000209    23                                      end

   Monte-UX  R5.10          FORTRAN77/M4   Rev.094                DATE:  Wed Nov 29 17:44:18 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$6
DIAGNOSTIC LIST

       ELN      LEVEL    NO.   DIAGNOSTIC MESSAGE

    000195        PAR      1 : Parallelized by DO index k
    000196        VEC      1 : Vectorized by DO index j
    000200        VEC      1 : Vectorized by DO index i

   Monte-UX  R5.10          FORTRAN77/M4   Rev.094                DATE:  Wed Nov 29 17:44:18 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$7
FORMAT LIST

  ELN   ILN     LABEL REF.        LOOP        FORTRAN STATEMENT

000210     1                                      SUBROUTINE VDFT3B$7 (n,  np,  fr,  cr,  fi,  ci)
000211     2                                      INTEGER n,  np
000212     3                                      REAL fr(np,n,n),  cr(np,n,n),  fi(np,n,n),  ci(np,n,n)
000213     4                                      INTEGER k,  j,  i
000214     5                                      *PDIR PARDO FOR = 4
000215     6                                      *VDIR NODEP
000216     7                                      *VDIR NOASSUME
000217     8                                      do k = 1,  n*n*np
000218     9                    PV---------->     !  cr(k,1,1) = fr(k,1,1)
000219    10                    !                 !  ci(k,1,1) = fi(k,1,1)
000220    11                    PV----------      end do
000221    12                                      return
000222    13                                      end
```

Fig. 4.6 Paralleled code for FFT on the Monte-4 by autotasking (vdft3b) (to be continued)

```
   Monte-UX  R5.10          FORTRAN77/M4   Rev.094                  DATE:  Wed Nov 29 17:44:18 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$7
DIAGNOSTIC LIST

       ELN      LEVEL    NO.   DIAGNOSTIC MESSAGE

    000217        VEC      1 : Vectorized by DO index k
    000217        PAR      1 : Parallelized by DO index k
                   **    107 : Variable or array not used, therefore not allocated : j
                   **    107 : Variable or array not used, therefore not allocated : i

   Monte-UX  R5.10          FORTRAN77/M4   Rev.094                  DATE:  Wed Nov 29 17:44:19 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$8
FORMAT LIST

  ELN   ILN    LABEL REF.        LOOP       FORTRAN STATEMENT

000223     1                                    SUBROUTINE VDFT3B$8 (n, n2, np, cr, nl1, ibr, fr, ci, fi)
000224     2                                    INTEGER n, n2, np, nl1, ibr(n*nl1)
000225     3                                    REAL cr(np,n,n), fr(np,n,n), ci(np,n,n), fi(np,n,n)
000226     4                                    INTEGER k, j, j2, j1, i
000227     5                                    *PDIR PARDO FOR = 4
000228     6                                    *VDIR NODEP
000229     7                                    c
000230     8                                    c
000231     9    19                              do 600 k = 1, n
000232    10    19            P---------->      !  do 600 j = 1, n2
000233    11                  ! V-------->      !  !  j2 = 2*j
000234    12                  ! !               !  !  j1 = j2 - 1
000235    13                  ! !               !  !  *VDIR       NODEP
000236    14    19            ! !               !  !  do 600 i = 1, n
000237    15                  ! ! V------>      !  !  !  fr(i,ibr(j1),k) = cr(i,j,k) + cr(i,j+n2,k)
000238    16                  ! ! !             !  !  !  fi(i,ibr(j1),k) = ci(i,j,k) + ci(i,j+n2,k)
000239    17                  ! ! !             !  !  !  fr(i,ibr(j2),k) = cr(i,j,k) - cr(i,j+n2,k)
000240    18                  ! ! !             !  !  !  fi(i,ibr(j2),k) = ci(i,j,k) - ci(i,j+n2,k)
000241    19                  P-V-V--------  600 continue
000242    20                                    return
000243    21                                    end

   Monte-UX  R5.10          FORTRAN77/M4   Rev.094                  DATE:  Wed Nov 29 17:44:19 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$8
DIAGNOSTIC LIST

       ELN      LEVEL    NO.   DIAGNOSTIC MESSAGE

    000231        PAR      1 : Parallelized by DO index k
    000232        VEC      1 : Vectorized by DO index j
    000236        VEC      1 : Vectorized by DO index i

   Monte-UX  R5.10          FORTRAN77/M4   Rev.094                  DATE:  Wed Nov 29 17:44:19 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$9
FORMAT LIST

  ELN   ILN    LABEL REF.        LOOP       FORTRAN STATEMENT

000244     1                                    SUBROUTINE VDFT3B$9(n2,n,kbase,np,fr,cr,fi,ci,nl1,cn,sn)
000245     2                                    INTEGER n2, n, kbase, np, nl1
000246     3                                    REAL fr(np,n,n), cr(np,n,n), fi(np,n,n), ci(np,n,n), cn(n2*nl1),
000247     4                                   1   sn(n2*nl1)
000248     5                                    INTEGER k, k2, k1, j, i
000249     6                                    REAL temp1, temp2
000250     7                                    *PDIR PARDO FOR = 4
000251     8                                    *VDIR NODEP
000252     9    21                              do 710 k = 1, n2
000253    10                  PV---------->     !  k2 = 2*k
000254    11                  !                 !  k1 = k2 - 1
000255    12    21            !                 !  do 710 j = 1, n
000256    13                  !                 !  !  *VDIR       NODEP
000257    14    21            ! 4-------->      !  !  do 710 i = 1, n
000258    15                  ! ! V------>      !  !  !  cr(i,j,k1) = fr(i,j,k) + fr(i,j,k+n2)
000259    16                  ! ! !             !  !  !  ci(i,j,k1) = fi(i,j,k) + fi(i,j,k+n2)
000260    17                  ! ! !             !  !  !  temp1 = fr(i,j,k) - fr(i,j,k+n2)
000261    18                  ! ! !             !  !  !  temp2 = fi(i,j,k) - fi(i,j,k+n2)
000262    19                  ! ! !             !  !  !  cr(i,j,k2) = temp1*cn(k+kbase) - temp2*sn(k+kbase)
000263    20                  ! ! !             !  !  !  ci(i,j,k2) = temp2*cn(k+kbase) + temp1*sn(k+kbase)
000264    21                  PV4-V--------  710 continue
000265    22                                    return
000266    23                                    end
```

Fig. 4.6 Paralleled code for FFT on the Monte-4 by autotasking (vdft3b) (to be continued)

```
    Monte-UX  R5.10          FORTRAN77/M4   Rev.094                DATE:  Wed Nov 29 17:44:19 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$9
DIAGNOSTIC LIST

       ELN      LEVEL    NO.   DIAGNOSTIC MESSAGE

     000252        VEC      1 : Vectorized by DO index k
     000252        PAR      1 : Parallelized by DO index k
     000257        VEC      1 : Vectorized by DO index i

    Monte-UX  R5.10          FORTRAN77/M4   Rev.094                DATE:  Wed Nov 29 17:44:19 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$10
FORMAT LIST

   ELN   ILN     LABEL REF.        LOOP        FORTRAN STATEMENT

000267     1                                      SUBROUTINE VDFT3B$10(n2,n,kbase,np,cr,fr,ci,fi,nl1,cn,sn)
000268     2                                      INTEGER n2, n, kbase, np, nl1
000269     3                                      REAL cr(np,n,n), fr(np,n,n), ci(np,n,n), fi(np,n,n), cn(n2*nl1),
000270     4                                     1   sn(n2*nl1)
000271     5                                      INTEGER k, k2, k1, j, i
000272     6                                      REAL temp1, temp2
000273     7                                      *PDIR PARDO FOR = 4
000274     8                                      *VDIR NODEP
000275     9     21                               do 720 k = 1, n2
000276    10                    PV---------->      !  k2 = 2*k
000277    11                    !                  !  k1 = k2 - 1
000278    12     21             !                  !  do 720 j = 1, n
000279    13                    !                  !  !  *VDIR       NODEP
000280    14     21             ! 4--------->      !  !  do 720 i = 1, n
000281    15                    ! ! V------->      !  !  !  fr(i,j,k1) = cr(i,j,k) + cr(i,j,k+n2)
000282    16                    ! ! !              !  !  !  fi(i,j,k1) = ci(i,j,k) + ci(i,j,k+n2)
000283    17                    ! ! !              !  !  !  temp1 = cr(i,j,k) - cr(i,j,k+n2)
000284    18                    ! ! !              !  !  !  temp2 = ci(i,j,k) - ci(i,j,k+n2)
000285    19                    ! ! !              !  !  !  fr(i,j,k2) = temp1*cn(k+kbase) - temp2*sn(k+kbase)
000286    20                    ! ! !              !  !  !  fi(i,j,k2) = temp2*cn(k+kbase) + temp1*sn(k+kbase)
000287    21                    PV4-V--------  720 continue
000288    22                                       return
000289    23                                       end

    Monte-UX  R5.10          FORTRAN77/M4   Rev.094                DATE:  Wed Nov 29 17:44:19 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$10
DIAGNOSTIC LIST

       ELN      LEVEL    NO.   DIAGNOSTIC MESSAGE

     000275        VEC      1 : Vectorized by DO index k
     000275        PAR      1 : Parallelized by DO index k
     000280        VEC      1 : Vectorized by DO index i

    Monte-UX  R5.10          FORTRAN77/M4   Rev.094                DATE:  Wed Nov 29 17:44:20 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$11
FORMAT LIST

   ELN   ILN     LABEL REF.        LOOP        FORTRAN STATEMENT

000290     1                                      SUBROUTINE VDFT3B$11 (n, np, fr, cr, fi, ci)
000291     2                                      INTEGER n, np
000292     3                                      REAL fr(np,n,n), cr(np,n,n), fi(np,n,n), ci(np,n,n)
000293     4                                      INTEGER k, j, i
000294     5                                      *PDIR PARDO FOR = 4
000295     6                                      *VDIR NODEP
000296     7                                      *VDIR NOASSUME
000297     8                                      do k = 1, n*n*np
000298     9                    PV---------->     !  cr(k,1,1) = fr(k,1,1)
000299    10                    !                 !  ci(k,1,1) = fi(k,1,1)
000300    11                    PV----------      end do
000301    12                                      return
000302    13                                      end
```

Fig. 4.6 Paralleled code for FFT on the Monte-4 by autotasking (vdft3b) (to be continued)

```
   Monte-UX  R5.10         FORTRAN77/M4   Rev.094                DATE:  Wed Nov 29 17:44:20 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$11
DIAGNOSTIC LIST

       ELN      LEVEL    NO.   DIAGNOSTIC MESSAGE

    000297        VEC      1 : Vectorized by DO index k
    000297        PAR      1 : Parallelized by DO index k
                  **     107 : Variable or array not used, therefore not allocated : j
                  **     107 : Variable or array not used, therefore not allocated : i

   Monte-UX  R5.10         FORTRAN77/M4   Rev.094                DATE:  Wed Nov 29 17:44:20 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$12
FORMAT LIST

  ELN   ILN    LABEL REF.        LOOP       FORTRAN STATEMENT

000303     1                                    SUBROUTINE VDFT3B$12 (n2, n, np, cr, nl1, ibr, fr, ci, fi)
000304     2                                    INTEGER n2, n, np, nl1, ibr(n*nl1)
000305     3                                    REAL cr(np,n,n), fr(np,n,n), ci(np,n,n), fi(np,n,n)
000306     4                                    INTEGER k, k2, k1, j, i
000307     5                                    *PDIR PARDO FOR = 4
000308     6                                    *VDIR NODEP
000309     7                                    *
000310     8                                    *
000311     9    19                              do 900 k = 1, n2
000312    10                 PV---------->      !  k2 = 2*k
000313    11                 !                  !  k1 = k2 - 1
000314    12    19           !                  !  do 900 j = 1, n
000315    13                 !                  !  !  *VDIR      NODEP
000316    14    19           ! 4--------->      !  !  do 900 i = 1, n
000317    15                 ! ! V------->      !  !  !  fr(i,j,ibr(k1)) = cr(i,j,k) + cr(i,j,k+n2)
000318    16                 ! ! !              !  !  !  fi(i,j,ibr(k1)) = ci(i,j,k) + ci(i,j,k+n2)
000319    17                 ! ! !              !  !  !  fr(i,j,ibr(k2)) = cr(i,j,k) - cr(i,j,k+n2)
000320    18                 ! ! !              !  !  !  fi(i,j,ibr(k2)) = ci(i,j,k) - ci(i,j,k+n2)
000321    19                 PV4-V--------  900 continue
000322    20                                    return
000323    21                                    end

   Monte-UX  R5.10         FORTRAN77/M4   Rev.094                DATE:  Wed Nov 29 17:44:20 1995
FILE NAME : v.vdft3b.f
PGNAME    : vdft3b$12
DIAGNOSTIC LIST

       ELN      LEVEL    NO.   DIAGNOSTIC MESSAGE

    000311        VEC      1 : Vectorized by DO index k
    000311        PAR      1 : Parallelized by DO index k
    000316        VEC      1 : Vectorized by DO index i
*************************************************************************
*                                                                       *
*  vdft3b        F : 0    S : 0    W : 5    0 : 0                      *
*                                                                       *
*  vdft3b$1      F : 0    S : 0    W : 2    0 : 0                      *
*                                                                       *
*  vdft3b$2      F : 0    S : 0    W : 2    0 : 0                      *
*                                                                       *
*  vdft3b$3      F : 0    S : 0    W : 2    0 : 0                      *
*                                                                       *
*  vdft3b$4      NO ERROR                                              *
*                                                                       *
*  vdft3b$5      NO ERROR                                              *
*                                                                       *
*  vdft3b$6      NO ERROR                                              *
*                                                                       *
*  vdft3b$7      F : 0    S : 0    W : 2    0 : 0                      *
*                                                                       *
*  vdft3b$8      NO ERROR                                              *
*                                                                       *
*  vdft3b$9      NO ERROR                                              *
*                                                                       *
*  vdft3b$10     NO ERROR                                              *
*                                                                       *
*  vdft3b$11     F : 0    S : 0    W : 2    0 : 0                      *
*                                                                       *
*  vdft3b$12     NO ERROR                                              *
*                                                                       *
*  ERROR TOTAL   F : 0    S : 0    W : 15   0 : 0                      *
*                                                                       *
*************************************************************************
```

Fig. 4.6 Paralleled code for FFT on the Monte-4 by autotasking (vdft3b)

5．比較考察

ここでは、並列化方式、主記憶容量、ジョブ・スケジューリングおよび、各実験機器で実行した並列化の結果について述べる。

5.1 並列化方式

Paragon XP/Sでの並列化には洗練された通信プログラムを作成する力とセンスが必要である。しかしながら、殆ど知識は必要なく、能力さえあれば並列化をスムーズに推進することが可能である。また、プログラミングの手間を省くため、高速で使用に耐える通信ライブラリの拡充が必要である。

一方VPP500では、パラダイムに関しての相当量の知識が必要であり、膨大かつ多彩なマニュアルや手引き書に適当で洗練されたものが少なく、声を掛けられる範囲に熟練者がいないと取りつくのに障壁が多い。また、作業量も多く、特に常套手段であるアドレス渡しでサブルーチンが作成されているとさらに作業量が増す。しかし、一旦習得すると、膨大な書換は必要であるが、殆ど頭を使うことなく並列化を書換の単純作業に帰着させることができる。また、後述するようにデータ分割に関しては自然にこれを遂行することができ、計算資源（分散型主記憶装置）を効率良く活用することが可能である。

Monte-4に限らず、自動並列化機能を装備していることは理想であるが、通信による並列化もパラダイムによる並列化も、PE疎結合かつ分散主記憶型であるがための機能提供がなされている。共有主記憶型の計算機ではこれらはオペレーティング・システムの担当で、複数計算機でクラスタを構成することがなければ、演算を行う過程において通信が表面化した問題となることはない。共有主記憶型の計算機は使用しやすく、ある程度の規模の解析では十分高速であり、結果をより早く得ることが可能である。ただ、超大規模計算まで対応できないので、高度な自動並列化機能を装備した分散主記憶型の計算機の出現が望まれる。

5.2 主記憶容量

Paragon XP/Sは32MB/ノードであるが、ユーザ（プログラム）使用可能領域は約20MB/ノードで、32PEでは約640MB/システム、256PEでは約5GB/システムである。VPP500は256MB/ノードであるが、ユーザ使用可能領域は約200MB/ノードで、42PEでは約8.2GB/システムである。Monte-4はオペレーティング・システムの領域を含め512MBの主記憶容量を有する。前述したようにTrans5をN=1024で実行するためには約73GBが必要で、これを各PEに分散して持っても主記憶装置に乗り切らない。ディスク入出力により強引に実行できるように改造しても、発生するオーバーヘッドにより効果が上がるか疑問である。超大規模演算を実行するためにはそれなりの主記憶容量が必要となり、並列化を容易にするためにはPEあたりの主記憶容量と軽いオペレーティング・システムが必須となる。

また、グローバル・アドレッシングが実現されていれば、他PEのデータをそのまま参照し演算レジスタに読み込むことが可能であるが、この機能がないと、他のPEのデータを受信する領

域を各受信側PEで確保しなければならず、主記憶装置の利用に無駄が発生する。このことは、特に配列で切った変数等の「all to all」の通信が発生する場合では、データ分割した意味がなくなるほど重大な問題となる。また、一括転送を行わなければ、プログラムの実行は可能であるが、通信オーバーヘッドが飛躍的に増大し現実的な並列化手法とは言えない。例えばVPP500では（x,y,z）をデータ分割して転置行列を作成しようとすると、zでデータ分割した（x,y,z）と、xかyで分割した転置行列用の領域のみが必要となる。これをParagon XP/Sで行うと、zでデータ分割をしても転置行列作成の前準備として「all to all」の通信が発生し、結局、受信側各PEで作業領域として（x,y,z）が必要となる。よって、主記憶装置に乗らない巨大プログラムをデータ分割により実行することは難しい。したがって、システム全体の総主記憶容量も重要であるが、PEあたりの主記憶容量も十分に実装されていることが必要である。

5.3 ジョブ・スケジューリング

Monte-4では効率的ジョブ・スケジューリングが行われ、マルチ・ジョブ環境が実現できている。すなわち、プログラムを並列化した場合に各CPUで実行されるタスクは、他のジョブのタスクとタイムスライスにより交互に実行される。これにより、計算資源の効率的運用が実現されているわけであるが、Paragon XP/S、VPP500に限らず現存するMPPシステムで並列クラスのマルチ・ジョブ環境を実現しているシステムは殆どない。例えば、ある並列クラスであるジョブが実行を開始すると、このクラスは占有され、他のジョブはこのジョブが終了するまで待たされる。運用形態の一つとして、長時間ジョブはユーザが実行を中断し再実行を繰り返すことで他のユーザに一時占有しているクラスを解放している場合がある。実際の運用は殆どの場合独占ユーザを認められない状況にあり、さらに、デバッグ途中のジョブは通常極めて短時間で終了することを考慮しても、マルチ・ジョブ環境を実現したオペレーティング・システムを開発すべきである。数秒のデバッグ・ジョブの終了を数時間あるいは数日以上待つのは非現実的であり、クラスの一時明け渡しを超大規模ジョブを実行するユーザの責任とするのも適当ではない。また、デバッグ用のクラスを独立して設定するのも著しい計算資源の無駄に通ずる。ジョブの切り替えは、レジスタをセーブして制御の切り替えを行うだけである。主記憶領域が足りない場合は、ロールアウトまたはロールインのオーバーヘッドが発生するが、ジョブ・サイズと関連付けられたタイムスライスの設定が問題で、運用状況に即した設定を行えば、オーバーヘッドを最小限に抑えることも可能である。

5.4 並列化の結果

Trans5は16本のサブルーチンから構成される。Paragon XP/Sではその内８本の変更を行い並列最適化に要した日数は３日である。VPP500ではその内７本の変更を行い並列最適化に要した日数は１週間である。Monte-4では自動並列化を使用し、その使用法と並列処理した結果を確認するまで数時間を要した。ただし、Paragon XP/Sではisend-irecvの調査に１週間、VPP500では単一版と並列版での途中結果が合わないため２週間を費やした。Paragon XP/Sでのisend-irecvの問題は、すぐにより高速なcsend-crecvに切り替えたため、これに要した１週間は並列最適化所要日数に影響しない純粋な調査期間である。しかし、VPP500で発生した問題は並列化が先に進まな

かったため、並列最適化所要日数を３週間とすべきかも知れない。

プログラムの実行時間を計測すると、機種により、また計測タイミングにより、例えシステムを占有した状態であっても数10%の誤差が生じることがある。よって、並列化の加速率を問うには一考を要する。数回計測して、最小値または最大値を取る方法、算術、調和、加重付算術平均を取る方法等が考え得るが、一般的規則はない。平均値や最大値を取ると当然最小値より低速な結果を得るが、最小値に比較して当然コンディションの悪かったケースで比較することになる。絶対的な比較で最も有効なのは、あらゆる不都合な環境が排除された状態、すなわち最小値であり、最小値による比較検討をここで提唱したい。

Monte-4の最大CPU数が４台のため、各計算機につきオリジナル単一版の単体での実行と並列版の４並列での計測のみを比較し一覧表にした。FFT（vdft3b）ルーチンおよびTrans5についての結果をTable 5.1 に示す。

FFTについては各実験機器の中でParagon XP/Sが最大で、加速率4.00倍を示しており、測定誤差を考慮してもほぼ無駄のない並列化が行われていることを示す。一方Monte-4の自動並列化では、前述した生成されたサブルーチンの分岐と復帰に要するオーバーヘッドと、並列化のオーバーヘッドおよび、共有主記憶型並列コード特有のバンクコンフリクト発生による影響のため、加速率は2.03倍に留まった。Trans5全体の加速率についてはParagon XP/Sが最大で加速率1.91倍を示した。FFTではVPP500はMonte-4と比較し1.7倍高速な加速率を示したが、Trans5全体ではMonte-4と同じ加速率に留まった。

プログラム全体に占めるファイル入出力の負荷割合が大きいと、演算部分をPE数に伴いリニアに高速化しても、プログラム全体の加速率は頭打ちになる。プログラム並列化に際して、入出力をできるだけ少なくするよう最適化を行うのも、現実の並列化において重要なポイントである。

また、実行時間については、FFTではVPP500がMonte-4に対し1.1倍、Paragon XP/Sに対し40倍高速であったが、Trans5全体ではMonte-4がVPP500に対し2.1倍、Paragon XP/Sに対し41倍高速であった。

Table 5.1 Parallelizing results

FFTルーチン

実験機器	1PEでの実行時間（秒）	4PEでの実行時間（秒）	加速率
Paragon XP/S	0.63167	0.15783	4.00
VPP500	0.01379	0.00392	3.52
Monte-4	0.00885	0.00435	2.03

Trans5全体

実験機器	1PEでの実行時間（秒）	4PEでの実行時間（秒）	加速率
Paragon XP/S	702.00	367.00	1.91
VPP500	35.85	19.16	1.87
Monte-4	16.78	8.97	1.87

6．おわりに

超並列計算機における科学技術分野の並列処理が脚光を浴び出して久しくなる。しかしながら、使用した計算機には、改善の余地が広範囲に渡り多く残されている。超並列計算機のピーク性能に恥じない性能を研究開発に携わる科学者、技術者が容易に引き出せる環境が整備実現されない限り、そのようなシステムに普及型計算機としての将来はないのではないだろうか。コンピュータ専門家の、たゆまぬ一層の努力が期待される。

謝辞

システム運用係の庄司氏には、Monte-4のシステム占有環境を設定するため、特別の配慮を頂いたことに感謝いたします。
並列処理支援技術開発グループの折居氏には、VPP500使用に関してアドバイスを頂き感謝いたします。また、横川研究員には、Trans5のコードを提供していただき感謝いたします。

参考文献

（1）Higuchi, K. et al.：Development of Monte Carlo Machine for Particle Transport Problem, J. Nucl. Sci. Technol., Vol. 32, No. 10 (Oct. 1995)
（2）横川　三津夫、蕪木　英雄：等方乱流シミュレーションコードの性能評価
IPSJ SIG Notes HPC 46-7, pp.37-44 (1993)
（3）PARAGON　System Performance Visualization Tool Users Guide
（4）UXP/M　VPPアナライザ使用手引書　V10用
（5）超高速モンテカルロ装置　ANALYZER-P/m4　利用の手引
（6）超高速モンテカルロ装置　PARALLELIZER/m4　利用の手引

6．おわりに

超並列計算機における科学技術分野の並列処理が脚光を浴び出して久しくなる。しかしながら、使用した計算機には、改善の余地が広範囲に渡り多く残されている。超並列計算機のピーク性能に恥じない性能を研究開発に携わる科学者、技術者が容易に引き出せる環境が整備実現されない限り、そのようなシステムに普及型計算機としての将来はないのではないだろうか。コンピュータ専門家の、たゆまぬ一層の努力が期待される。

謝辞

システム運用係の庄司氏には、Monte-4のシステム占有環境を設定するため、特別の配慮を頂いたことに感謝いたします。
並列処理支援技術開発グループの折居氏には、VPP500使用に関してアドバイスを頂き感謝いたします。また、横川研究員には、Trans5のコードを提供していただき感謝いたします。

参考文献

（1）Higuchi, K. et al.：Development of Monte Carlo Machine for Particle Transport Problem, J. Nucl. Sci. Technol., Vol. 32, No. 10 (Oct. 1995)
（2）横川　三津夫、蕪木　英雄：等方乱流シミュレーションコードの性能評価
IPSJ SIG Notes HPC 46-7, pp.37-44 (1993)
（3）PARAGON　System Performance Visualization Tool Users Guide
（4）UXP/M　VPPアナライザ使用手引書　V10用
（5）超高速モンテカルロ装置　ANALYZER-P/m4　利用の手引
（6）超高速モンテカルロ装置　PARALLELIZER/m4　利用の手引

6．おわりに

超並列計算機における科学技術分野の並列処理が脚光を浴び出して久しくなる。しかしながら、使用した計算機には、改善の余地が広範囲に渡り多く残されている。超並列計算機のピーク性能に恥じない性能を研究開発に携わる科学者、技術者が容易に引き出せる環境が整備実現されない限り、そのようなシステムに普及型計算機としての将来はないのではないだろうか。コンピュータ専門家の、たゆまぬ一層の努力が期待される。

謝辞

システム運用係の庄司氏には、Monte-4のシステム占有環境を設定するため、特別の配慮を頂いたことに感謝いたします。
並列処理支援技術開発グループの折居氏には、VPP500使用に関してアドバイスを頂き感謝いたします。また、横川研究員には、Trans5のコードを提供していただき感謝いたします。

参考文献

（1）Higuchi, K. et al.：Development of Monte Carlo Machine for Particle Transport Problem, J. Nucl. Sci. Technol., Vol. 32, No. 10 (Oct. 1995)
（2）横川　三津夫、蕪木　英雄：等方乱流シミュレーションコードの性能評価
IPSJ SIG Notes HPC 46-7, pp.37-44 (1993)
（3）PARAGON　System Performance Visualization Tool Users Guide
（4）UXP/M　VPPアナライザ使用手引書　V10用
（5）超高速モンテカルロ装置　ANALYZER-P/m4　利用の手引
（6）超高速モンテカルロ装置　PARALLELIZER/m4　利用の手引

付録

Paragon XP/S ----- コンパイル (Makefile)

```
FC  = if77 -nx -O3
ISRCS  = setfld.f predin.f mkcnst.f makvel.f makvor.f setvel.f ¥
      tospec.f getksp.f ranfld.f putfld.f
TSRCS  = trans5.f datain.f mkcnst.f calcfn.f addvar.f omg2pv.f ¥
      putksp.f getksp.f convol.f getfld.f putfld.f printl.f ¥
      nextjb.f
SSRCS  = statis.f mkcnst.f stat01.f stat02.f stat03.f omg2uv.f ¥
      getfld.f mklist.f putksp.f
DFTSRCS = vdft3i.f vdft3f.f vdft3b.f
UNISRCS = nrdist.f
all : setfld.o predin.o mkcnst.o makvel.o makvor.o setvel.o ¥
      tospec.o getksp.o ranfld.o putfld.o ¥
      trans5.o datain.o calcfn.o addvar.o omg2pv.o ¥
      putksp.o convol.o getfld.o printl.o nextjb.o ¥
      statis.o stat01.o stat02.o stat03.o omg2uv.o ¥
      mklist.o ¥
      vdft3i.o vdft3f.o vdft3b.o nrdist.o
setfld.o : setfld.f
    $(FC) -c $@ setfld.f
predin.o : predin.f
    $(FC) -c $@ predin.f
mkcnst.o : mkcnst.f
    $(FC) -c $@ mkcnst.f
----------省略----------
nrdist.o : nrdist.f
    $(FC) -c $@ nrdist.f
IOBJS  = ${ISRCS:.f=.o}
TOBJS  = ${TSRCS:.f=.o}
SOBJS  = ${SSRCS:.f=.o}
DFTOBJS = ${DFTSRCS:.f=.o}
UNIOBJS = ${UNISRCS:.f=.o}
all: setfld trans5 statis
setfld: $(IOBJS) $(DFTOBJS) $(UNIOBJS)
    $(FC) -o $@ $(IOBJS) $(DFTOBJS) $(UNIOBJS)
trans5: $(TOBJS) $(DFTOBJS)
    $(FC) -o $@ $(TOBJS) $(DFTOBJS)
statis: $(SOBJS) $(DFTOBJS)
    $(FC) -o $@ $(SOBJS) $(DFTOBJS)
clean:
    rm -f core *.o setfld trans5 statis vortex *~ #*#
```

Paragon XP/S ----- 実行

```
#!/bin/csh -f
#@$-lt 10:00
#@$
cd $HOME/trans5/para
/usr/bin/time trans5 -plk -sz 4 < input/input5.fort02
```

VPP500 ----- コンパイル（Makefile）

```
FC  = frtpx -Wx
```

以下省略「Paragon XP/S ----- コンパイル（Makefile）」を参照のこと

VPP500 ----- サンプラparasampの実行

```
#!/bin/csh -f
#@$-lt 10:00
#@$
setenv F7PARASAMP file:$HOME/wkvfl/trans5.time,interval:10,type:vtime
cd $HOME/trans5/para
timex -H trans5 < input/input5.fort02
parasamp trans5
```

VPP500 ----- アナライザafrtの実行

```
#!/bin/csh -f
#@$-C trans5
#@$
cd $HOME/trans5/ana
afrt -px240 -e -f alist -w80 -Wc,'-I./,-L./' addvar.o convol.o datain.o getfld.o getksp.o mkcnst.o nextjb.o omg2pv.o printl.o putfld.o
putksp.o vdft3b.o vdft3f.f vdft3i.o calcfn.o vdft3f.f trans5.f < input/input5.fort02
```

Monte-4 ----- コンパイル（Makefile）

```
FC  = f77sx -multi -fopp par -L fmtlist
```

以下省略「Paragon XP/S ----- コンパイル（Makefile）」を参照のこと

Monte-4 ----- アナライザANALYZER/m4の実行

```
set FANPOPT = '-tm  -af ana.tm -AR trans5.tim  -AL f=tm.l lpath lpunit -AO do'
#set FANPOPT = '-ct -AL fmt cost 90 f=ct.l'
#set FANPOPT = '-st -AD o=st.dbs  -AL f=st.l struct iall line xref prog com arg'
#set F77SXOPT = '-As -multi -float2'
set TSRCS  = 'trans5.f datain.f mkcnst.f calcfn.f addvar.f omg2pv.f putksp.f getksp.f convol.f getfld.f putfld.f printl.f nextjb.f'
set DFTSRCS = 'vdft3i.f vdft3f.f vdft3b.f'
cd $HOME/Trans5/para
fanp $FANPOPT $TSRCS $DFTSRCS < input/input5.fort02
```