ハイパフォーマンス
コンピューティング 82−15
（2000．8．4）

分散共有メモリ向け手続き間自動データ分散方法の実装と評価

廣岡孝志　太田寛　飯塚孝好　菊池純男
日立製作所システム開発研究所

分散共有メモリ向けコンパイラにおける手続き間自動データ分散方法の実装を行った．データ分散方法として，OS が行うファーストタッチ方式データ分散をコンパイラで制御する「ファーストタッチ制御方法」とデータ分散指示文を併用し，従来のデータ分散方法が不得手とするプログラムパターンに対し，最適なデータ分散を実現する点を特徴とする．これに手続き間解析機能を搭載し，プログラム全体の解析結果に基づくデータローカリティ最適化を実現した．SGI/Origin2000 を用いた評価の結果，ベンチマークプログラム NPB2.3serial/FT，SP，CG，SPECfp95/tomcatv の 4 題について，コンパイラによる自動データ分散を行わない場合に比べて 16 プロセッサ時で平均 35.3%性能が向上することを確認した．

Implementation and Evaluation of Interprocedural Automatic Data Distribution Method for Distributed Shared Memory

Takashi HIROOKA, Hiroshi OHTA, Takayoshi IITSUKA
and Sumio KIKUCHI
Systems Development Laboratory, Hitachi Ltd.

We implemented an interprocedural automatic data distribution method for distributed shared memory compilers. This method can achieve appropriate data distribution for program patterns where usual data distribution method is unsuitable, by using at the same time the data distribution directive and the *first touch control method* by which the compiler controls the first touch data distribution of the operating system. In addition, we implemented the interprocedural analysis method, which improves the data locality of the whole program. By evaluation in SGI/Origin2000 with the benchmark FT, SP and CG of NPB2.3serial and tomcatv of SPECfp95, we show performance improvement of 35.3% in the case of 16 processors.

1．はじめに

分散共有メモリ(DSM)型並列計算機は，共有メモリの容易な並列プログラミング環境を提供しながら，分散メモリのスケーラビリティを確保できるアーキテクチャとして注目を集めている．並列計算機を効率良く利用するためには，並列化率，ロードバランス，データローカリティが重要となる．本研究では，データローカリティの最適化に着目した．物理的に分散されたメモリを有する DSM では，最適なデータ分散が必要不可欠であり，このためコンパイラの果たすべき役割は大きいと考えられる．そこで，本研究では，最適なデータ分散形状をコンパイラで自動的に決定する方法を検討し，実装した．

従来，物理分散メモリを有する並列計算機に対するデータ分散方法として，データ分散指示文によるユーザ指示[1]，OS が行うファーストタッチ方式データ分散[1]などが研究されてきた．また，コンパイラによる自動データ分散方法としては，主にデータ分散指示文を自動生成する方法が利用されてきた[2,3,4,5,6]．最適なデータ分散形状を決定する問題は NP 完全であるため，Kennedy ら[3]，Gupta ら[4]，辰巳ら[5]，松浦ら[6]などから，近似解を求める様々なヒューリスティクが提案されてきた．ところが，従来の自動データ分散方法では，実プログラム中に比較的良く表れるプログラムパターンに対して容易に最適なデータ分散形状を実現できない場合があった．この問題を解決するため，前報告[8]では，OS が行うファーストタッチ方式データ分散をコンパイラで制御する**ファーストタッチ制御方法**を提案した．本報告では，ファーストタッチ制御方法

を組み込んだ自動データ分散方法の実装，評価について述べる．本研究で実装した自動データ分散方法は，ファーストタッチ制御方法とデータ分散指示文を併用し，従来，主に利用されてきたデータ分散指示文の不得手とする場面を克服するところが利点である．加えて，手続き間解析機能によりプログラム全体の解析結果に基づくデータローカリティ最適化を実現した．

本報告は次の構成をとる．2 章では，研究の背景となる分散共有メモリ機構について説明した後，実装した DSM 自動並列化コンパイラの構成を述べる．3 章では，DSM 向け手続き間自動データ分散方法の概要を述べた後，データ分散形状，およびデータ分散実施手段の決定方法等の実装について説明する．4 章で評価結果を示し，5 章で結論を述べる．

2．システム構成

DSM を実現する手段の主流として，仮想メモリ空間をページ単位で切り分け，各ノードの物理メモリに割り付ける方法がある．今回，プラットフォームとして用いた代表的 DSM 型並列計算機 SGI/Origin2000 もこれを採用している．ページをどのノードの物理メモリに割り付けるかを決める方法をデータ分散方法という．Origin2000 には，自ノードに割り付けられたデータの参照の割合，すなわちデータローカリティを向上させる目的で用意されたデータ分散方法が 2 つある [1]．

(1)データ分散指示文によるデータ分散

プログラム中にユーザが挿入したデータ分散指示文に従って，コンパイラが各ノードにデータを割り付ける．

(2)ファーストタッチ方式データ分散

OS が，各ページを最初にアクセスしたノードの物理メモリに，そのページを割り付ける．

上記従来方法では，カーネルループにおいて配列の一部のみが参照される場合や，手続き毎に引数配列の宣言形状が異なる場合に，容易に最適なデータ分散が実現できないという問題点があった．前報告 [8]では，この問題を解決するファーストタッチ制御方法を提案した．これは，OS が行うファーストタッチ方式データ分散をコンパイラが制御することによるデータ分散方法である．図 1 を用いて説明する．図 1(a)に示した例題プログラム 1 の場合，従来方法では図 1(b)に示すようなデータ分散指示文をプログラムの先頭に挿入していた．提案方法では，コンパイラが図 1(c)に示すようなカーネルループ中の参照パターンを再現するダミーループを生成してプログラムの先頭に挿入し，OS が行うファーストタッチ方式データ分散に従ってデータ分散させることにより，カーネルループ向けのデータ分散を実現させる（以後，この一連のコードを**ファーストタッチ制御コード**と呼ぶ）．なお，本報告では，上記ファーストタッチ制御コードとデータ分散指示文の総称をデータ分散実施コードと呼ぶ．

次に，提案する DSM向け手続き間自動データ分散方法のシステム構成を図 2 に示す．本自動データ分散技術は，既存の SMP 向け自動並列化コンパイラ WPP[7]にデータ分散部を挿入することで実現した．以後，本コンパイラを DSM 自動並列化コンパイラと呼ぶ．DSM 自動並列化コンパイラは，Fortran77 で記述された逐次ソースプログラムを入力し，これに処理分散指示文（OpenMP 指示文），およびデータ分散実施コードを挿入したソースプログラムを出力する．さらに，DSM 向けにデータ分散指示文を追加した拡張 OpenMP コンパイラが，このソースプログラムを入力し，DSM 並列オブジェクトプログラムを出力して DSM 型並列計算機に入力し，DSM 並列実行を実現する．DSM 型並列計算機としては Origin2000 を用い，拡張 OpenMP コンパイラとしては Origin2000 に搭載された MIPSPro Fortran90 コンパイラを用いた．これに伴い，DSM 自動並列化コンパイラが挿入する処理分散指示文としては OpenMP 指示文を用い，データ分散指示文としては Origin2000 向けのデータ分散指示文(SGI 指示文)を用いている．

データ分散部では，まず処理分散部で決定した並列

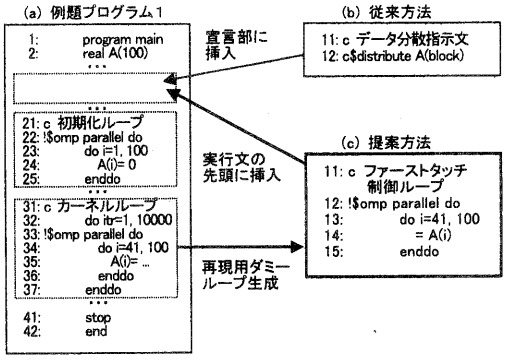

図 1 ファーストタッチ制御方法

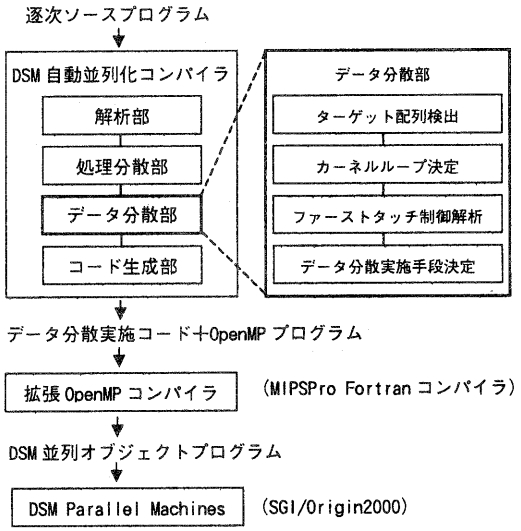

図 2 DSM 自動並列化コンパイラの構成

化ループを検出し，このループ本体に参照を有する配列（以後，**ターゲット配列**と呼ぶ）を検出する．次にターゲット配列毎に各並列化ループ向けデータ分散形状を決定し，プログラム中で最も実行頻度が高いループ（以後，**カーネルループ**と呼ぶ），およびターゲット配列のデータ分散形状を決定する．また，このときデータ再分散解析を行い，ターゲット配列のデータ再分散形状を決定する．次に提案方法の特徴技術の一つであるファーストタッチ制御データ分散を適用するために必要な解析を行い，最後にデータ分散実施手段の決定を行なう．

3．DSM向け手続き間自動データ分散方法

3.1 概要

分散メモリ機構を有する並列計算機の場合，各データの配置については，そのデータが初めて参照されるまでには何らかの形状で配置されている必要がある．したがって，手続き間に跨って参照される配列のデータ分散について考えると，その配列が初めて参照される手続きで決定したデータ分散形状をプログラム全体で利用し続けるか，必要な場面でデータ再分散を行うしかない．その結果，マシンの特性によっては，最初に行ったデータ分散の形状，およびデータ再分散コストが，性能に大きく影響を及ぼすことになる．

そこで，本自動データ分散方法では，手続き間解析によりプログラム全体でデータ分散形状を固定させた場合に最適となるデータ分散形状を決定して初期分散させ，データ再分散解析の結果を基にコストに見合う場合にデータ再分散を実施してプログラム全体のデータローカリティを最適化させる手法をとる．

本章では，実装した自動データ分散方法の概要を示し，具体的にデータ分散形状を決定していく過程を説明する．本方法では，以下のステップを経てデータ分散形状，およびデータ分散実施手段を自動決定する．

(1)ターゲット配列の検出
(2)各並列化ループ向けデータ分散形状の決定
(3)手続き内カーネルループの決定
(4)手続き間カーネルループの決定
(5)ファーストタッチ制御向け情報の検出
(6)データ分散実施手段の決定

次節で，これらを詳細に説明する．

3.2 データ分散形状，データ分散実施手段の決定

図3の例題プログラム2が，本DSM自動並列化コンパイラの入力となった場合の処理について説明する．ここでは，図2の処理分散部[7]において，5つのループのうち,L1とL4が並列化ループとして決定済であると仮定する．

本方法では，入力プログラムに対し，図4に示すようなデータ分散グラフを生成し解析に用いる．グラフは，配列ノード，ループノード，参照パターンノードから構成され，エッジはノード間の参照の有無を表す．図3と図4を用いて解析の過程を説明する．

(1)ターゲット配列の検出

並列化ループ(L1,L4)を検出し，並列化ループのループ本体で参照される配列(A,B)をターゲット配列とする．グラフでは，配列ノードAとBを生成する．

```
 1: do i= 1, 100                    ; L1
 2:   do j= 1, 100                  ; L2
 3:     A(i,j)= ・・・
 4:   enddo
 5: enddo
    ・・・
11: do iter= 1, 10                  ; L3
12:   do i= 2, 50                   ; L4
13:     do j= 2, 99                 ; L5
14:       B(j,i)= A(j+1,i+1)
15:             + A(j+1,i-1)
16:             + A(j-1,i-1)
17:     enddo
18:   enddo
19: enddo
```

図3 例題プログラム2

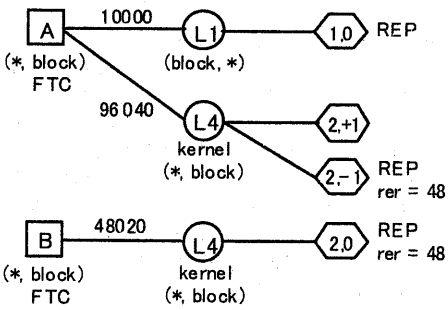

図4 データ分散グラフ

(2)各並列化ループ向けデータ分散形状の決定

ターゲット配列毎に参照をループ本体に含む並列化ループを検出し，各ループ向けデータ分散形状を決定する．ループ向けデータ分散形状は，ループ本体の参照において並列化ループのループ制御変数が最も多く現れる次元をblock分散した形状とする．

グラフでは，まず始めに配列ノードAに対し，ループノードL1とL4を生成して接続し，配列ノードBに対し，ループノードL4を生成して接続する．そして，各々のループノードの属性としてデータ分散形状を設定する．

次に，各ループにおけるターゲット配列の参照パターンを検出し，参照パターンノードを接続する．参照パターンノード中に示した情報の1要素目は並列化ループのループ制御変数を含む次元，2要素目は並列化ループのループ制御変数単独式からの距離を示す（例えば，i+1の場合，距離は1）．次に，全参照パターンの中から以下の優先順位で代表参照パターンを決定する．

・参照回数が最大の参照パターン
・左辺の参照パターン
・ループ制御変数単独式との距離が最小の参照パターン

代表参照パターンを示すフラグは，参照パターンノー

ドの属性として登録する(図4のREP)．次に，代表参照パターンの添字式と並列化ループから各ループ向けのデータ分散形状を決定し，ループノードの属性として登録する．

(3)手続き内カーネルループの決定

データ分散形状毎に並列化ループをグループ化し，各並列化ループの演算量を見積って合計が最大となるグループを決定する．このグループの中で演算量最大の並列化ループをターゲット配列の手続き内カーネルループとする．このカーネルループ向けのデータ分散形状（(2)で決定したもの）を，ターゲット配列の手続き内データ分散形状とする．グラフでは，配列ノードとループノード間のエッジの属性として代表参照パターンの参照回数を登録し，参照回数が最大のループを手続き内カーネルループと決定する(図4のkernel)．

(4)手続き間カーネルループの決定

ターゲット配列が引数の場合，引き渡される各手続きにおける手続き内カーネルループを検出する．次に，データ分散形状毎に手続き内カーネルループをグループ分けし，演算量の合計が最大となるグループを検出する．このグループの中で演算量最大の手続き内カーネルループを手続き間カーネルループとする．

(5)ファーストタッチ制御向け情報の検出

ターゲット配列毎の代表参照パターンを検出し，添字式に含まれるループ制御変数の上限，下限，増分，およびループネスト順序を保持する．

(6)データ分散実施手段の決定

ターゲット配列が以下の条件を満たす場合，ファーストタッチ制御方法を選択する．

・カーネルループにおいて部分配列参照が存在する．
・引数配列であり，手続き毎に宣言形状が異なる．

その他の場合，指示文による方法を選択する．

具体的には，まず宣言された全要素数に対するカーネルループにおいて参照される要素数の割合を参照要素率とと定義し，代表参照パターンの参照要素率を解析する．参照要素率（図4のrer）が50%以下の場合，データ分散実施手段として，ファーストタッチ制御による方法（図4のFTC）を選択し，その他の場合，データ分散指示文による方法（DIR）を選択する．決定したデータ分散実施手段は，配列ノードの属性として登録する．グラフでは，カーネルループの代表参照パターンの参照要素率から，ターゲット配列A，BともにFTCが選択された．以上の解析結果に基づき図5に示すようなデータ分散実施コードが生成されプログラムの先頭に挿入される．

```
1:  !$omp parallel do
2:  do i= 2, 50
3:     do j= 2, 99
4:        B(j,i)= 0
5:        A(j+1,i-1)= 0
6:     enddo
7:  enddo
```

図5 データ分散実施コード

4．評価

Origin2000を用いて，実装した自動データ分散方法の評価を行った．結果を報告する．評価には，SPEC Benchmarkの SPECfp95/tomcatv，および NASA 提供の標準的ベンチマーク NPB2.3serial/FT，SP，CG の4題を用いた[10]．これらのベンチマークを入力とした際の本方法による解析結果を表2にまとめる．表には，各ベンチマークの主要配列のうち自動データ分散機能がターゲット配列として検出したものについて，そのデータサイズ，および決定したデータ分散形状とデータ分散実施手段を示す．以下で各ベンチマークの性能に対する影響を考察する．

表1 測定条件

マシン	SGI/Origin2000
ノード	16ノード(2cpu/ノード)
CPU	MIPS RISC R10000(195MHz)
キャッシュ	L1:32KB, L2:4MB
メモリ	11GB(11264MB)
OS	IRIX6.5.4
コンパイラ	MIPSPro Fortran90(Version7.3)
コンパイルオプション	-mp -64 -Ofast=IP27 -OPT:IEEE_arith=3

まず，測定条件を表1に示し，各ベンチマークの性能測定結果を図6から図9に示す．グラフは，横軸にプロセッサ数，縦軸に性能向上比を示し，本方法による自動データ分散を行った場合(dist)と行わない場合(no dist)の性能を比較したものである．なお，no distでは，OSが行うファーストタッチ方式データ分散に従う．測定に用いたOrigin2000は，1ノード2プロセッサ構成であるため，4プロセッサ以降でデータローカリティの影響が発生する．

4.1 FT

FTは，カーネル部分と主要配列が比較的限定されている．本方法により，主要配列の多くが自動データ分散のターゲットとして検出され，各配列に対し，表1に示すようなデータ分散形状が求められた．データ分散実施手段としては，主要配列の宣言形状が手続き毎に異なることから，4題のベンチマークで唯一ファーストタッチ制御方法(FTC)が選択された．指示文による方法では，カーネルループにおいて最適となるデータ分散形状を宣言形状の異なる最親手続きで指示することは困難である．

性能向上比を図6に示す．各クラス共，コンパイラによる自動データ分散を行わない場合，プロセッサ数の増大と共にデータローカリティが低下するため，十分なスケーラビリティを得ることができない．一方，自動データ分散機能を適用した場合，最適なデータ分散が実施され，データローカリティが改善されて大幅な性能向上を実現した．classB, 16プロセッサ時で89%性能が向上した．また，classAとclassBの比較から，データサイズが増大すると性能差が拡大する傾向にあることを確認した．

FTの場合，プログラム中にデータ再分散により大き

表2　自動データ分散機能が決定した主要配列のデータ分散形状

Program	Key Arrays	Data Size		Data Distribution	Method
		classA	classB		
FT	U1, INDEXMAP	256x256x128	512x256x256	(*,*,block)	FTC
	U0, U2	256x256x128	512x256x256	(*,block,*)	
SP	LHS	64x64x63x15	102x102x101x15	(*,*,block,*)	DIR
	FORCING, RHS, U	64x64x63x5	102x102x101x5	(*,*,block,*)	
	SQUARE, SPEED, RHO_I AINV, QS, WS, VS, US	64x64x63	102x102x101	(*,*,block)	
CG	COLIDX	2198000	15825000	(block)	DIR
	ROWSTR	14001	75001	(block)	
tomcatv	AA, DD, X, Y, RX, RY	513x513		(*,block)	DIR
	D	513x513		(block,*)	

くデータローカリティが改善される部分がある．しかし，本方法ではコスト計算の結果，データ再分散は実施されなかった．これは，Origin2000 のデータ再分散コストが大きいためである．データ再分散のコストが小さくなり，データ再分散を実施してデータローカリティの改善が実現できれば，さらにスケーラビリティが向上すると考えられる．

4.2 SP

SP では表 1 に示すような主要配列を検出し，表記のようなデータ分散形状を求めることができた．データ分散実施手段は，指示文による方法(DIR)が選択された．

性能向上比を図 7 に示す．SP は，コンパイラによる自動データ分散を実施しない場合でも比較的良好な並列性能を示していた．しかし，本自動データ分散方法により，さらに性能を向上させることができ，classB, 16 プロセッサ時で 12%の性能向上を実現した．また，データサイズが増大するとプロセッサ数の増加と共に自動データ分散機能による性能向上の割合が拡大する傾向を確認した．

4.3 CG

CG は，カーネルループと主要配列が限定されている．また，処理分散の並列化効率が高く，データローカリティに問題がなければ良好なスケーラビリティを得ることができると予想される．本方法により，主要配列 3 つのうち 2 つがターゲットとして検出された．

性能向上比を図 8 に示す．CG は，classA では主要配列のデータが各プロセッサのキャッシュに載りきるため，コンパイラによる自動データ分散機能の有無に関係なく良好な並列性能を示した．しかし，classB 以降の大きなデータサイズでは，データローカリティの悪影響により，大きくスケーラビリティを悪化させている．コンパイラによる自動データ分散を適用した場合，少しスケーラビリティが改善し，classB, 16 プロセッサ時で 39%性能が向上するが，十分な並列性能を得ることはできなかった．

分析の結果，CG では残る 1 つの主要配列が間接参照を有するため，ターゲット配列の対象から除外されていた．また，この配列のデータローカリティが性能に大きく影響を及ぼすことが分かったため，この配列をターゲット配列に含めた場合の性能を測定した．結果を図 8 の(dist + indirect)に示す．このように良好な並列性能を得ることが確認できた．よって，間接参照を有する配列をターゲット配列に含める改善を現在実施中である．本機能が実装できれば，コンパイラによる自動データ分散を実施しない場合に比べて，classB, 16 プロセッサ時で 295%性能を向上させることが可能となる．

4.4 tomcatv

性能向上比を図 9 に示す．tomcatv でも，自動データ分散機能により，データローカリティが改善され，16 プロセッサ時で 20%性能が向上した．

4.5 全体について

本自動データ分散方法では，決定済の処理分散を起点にデータ分散を行う．したがって，処理分散の良し悪しは考慮しない．現状の処理分散でデータローカリティの影響により十分な並列性能が出せていない場合に効果を発揮する．その意味で測定した 4 題のベンチマークは，各々この条件を満たし性能向上を得ることができた．

また，本評価は OS が行うファーストタッチ方式データ分散とコンパイラによる自動データ分散の性能差を比較したものである．したがって，前者の場合，初期化ループの並列化が重要である．

- 初期化ループが，データフロー依存上や解析不能情報のために並列化不能である．
- ターゲット配列の宣言形状が，カーネルループを含む手続きと異なる．
- 初期化ループの構造や並列化ネスト位置が，カーネルループと異なる．

といった場合，低いデータローカリティの影響でスケーラビリティを著しく悪化させるケースがある．FT, CG がこれに該当し，本自動データ分散方法の効果が大きいタイプと考えられる．

5．まとめ

本稿では，DSM 向け手続き間自動データ分散方法の実装と評価について述べた．データ分散方法として，文献 [8] で報告したファーストタッチ制御方法とデータ分散指示文を併用し，手続き間解析機能を搭載した自動データ分散方法を実装したことにより，従来のデータ分散方法では困難であった最適データ分散，および

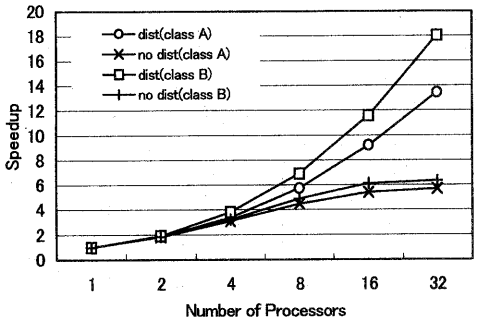

図6 FT/ NPB2.3serial

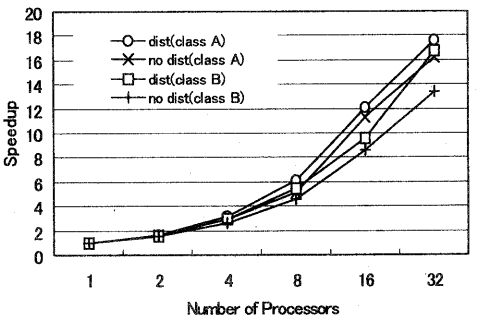

図7 SP/ NPB2.3serial

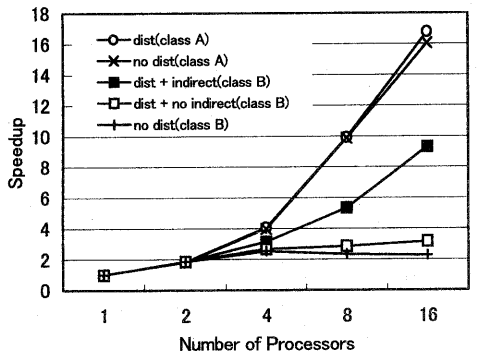

図8 CG/ NPB2.3serial

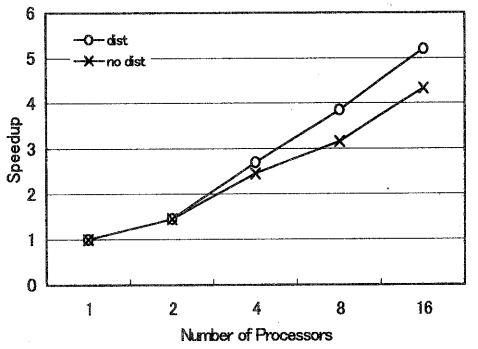

図9 tomcatv/SPECfp95

プログラム全体で最適となるデータ分散が実現可能となった．Origin2000を用いた評価の結果，ベンチマークプログラムNPB2.3serial/FT，SP，CG，SPECfp95/tomcatvの4題で，コンパイラによる自動データ分散を行わない場合に比べて16プロセッサ時，平均35.3%性能が向上することを確認した．本研究の成果は，上記のような自動データ分散方法，および手続き間解析方法を示し実装した点，および実機での性能評価により，代表的なベンチマークで良い結果を得て本方法の有効性を示した点にある．

文献[8]と今回の報告で，本方法の提案から実装，評価までをまとめた．今後は，本自動データ分散方法の妥当性やファーストタッチ制御方法の適用条件に関する検討を進め，さらに多くのベンチマークによる評価を実施して機能拡張の検討に用い，次バージョンの設計，実装へ反映させていく予定である．

参考文献

1) R.Chandra, D.Chen, R.Cox, D.E.Maydan, N.Nedeljkovic, J.Anderson, "Data Distribution Support on Distributesd Shared Memory Multiprocessors", Silicon Graphics Computer Systems, Digital Western Research Lab,PLDI'97 (1997).

2) T.N.Nguyan, Z.Li, "Interprocedural Analysis for Loop Scheduling and Data Allocation", Parallel Computing, Vol.24, No3-4, pp.477-504, (1998).

3) K.Kennedy, U.Kremer, "Automatic Data Layout for High Performance Fortran", Proc. of Supercomputing'95 (1995).

4) Gupta, M., Banerjee, P.,: PARADIGM: A Compiler for Automatic Data Distribution on Multicomputers, Proc. ICS'93, pp.87-96 (1993).

5) 辰巳尚吾，窪田昌史，五島正裕，森眞一郎，中島浩，富田眞治，"並列化コンパイラTINPARにおける自動データ分割部の実現"，情報処理学会研究報告，96-PRO-8, pp.25-30 (1996).

6) 松浦健一郎，村井均，末廣謙二，妹尾義樹，"データ並列プログラムに対する高速な自動データ分割手法"，JSPP'99論文集，pp.87-94 (1999).

7) 青木雄一郎，佐藤真琴，飯塚孝好，佐藤茂久，菊池純男，"手続き間自動並列化コンパイラ WPP の試作 － 実機性能評価 －"，情報処理学会研究報告，98-ARC-130, pp.43-48 (1998).

8) 廣岡孝志，太田寛，菊池純男，"ファーストタッチ制御による分散共有メモリ向け自動データ分散方法"，情報処理学会論文誌，Vol.41, No.5, pp.1430-1438 (2000).

9) SGI MIPSpro Fortran77 Programmer's Guide, Silicon Graphics Inc

10) The NAS Parallel Benchmarks, http://www.nas.nasa.gov/Software/NPB/