解　説

マルチプロセッサスーパコンピュータ PHI の研究開発

マルチプロセッサスーパコンピュータ HPP システム†

神谷幸男††　橋本　伸††

1. はじめに

ここでは，通産省・工業技術院・NEDO によって 1981 年から 1990 年にかけて行われた，大型工業技術研究開発制度の「科学技術用高速計算システム」の研究開発の一環として行われた高速演算用並列処理装置(*HPP : High-speed Parallel Processor) についての解説を行う．HPP は，4 台の局所記憶装置をもつベクトルプロセッサを共用記憶装置で結合したマルチプロセッサスーパコンピュータであり，1990 年 1 月に試験運用を行った．

1.1 PHI の中の HPP

「科学技術用高速計算システム」の研究開発の中で作成された総合システムを *PHI と呼ぶ．

HPP は，PHI システムの一部であり，高速演算用並列処理装置 (High-speed Parallel Processor) の略称である．一般的に，コンピュータは，制御装置，演算装置，記憶装置及び入出力装置などから構成されていると考えることができる．PHI を 1 台のコンピュータとみなした場合に，HPP は演算装置，特に浮動小数点演算装置に相当する．

このプロジェクトが開始された 1981 年当時には，最大の演算能力をもつコンピュータの性能は 100 *MFLOPS 程度であった．その 100 倍の性能 (10 *GFLOPS) をもつコンピュータを構築しようというのが，このプロジェクトの目的であった．この目的を達成するために，非シリコン系素子の開発と，並列処理が必要であると考えられた．HPP では，*HEMT 素子の実装と，並列処理の効率的利用によってこれを実現した．

1.2 HPP 開発の方針

巨大な科学技術計算を行うためには，以下の要件を満たすようなシステムが必要である．

（1） 高速演算能力
（2） 大容量記憶
（3） アプリケーション作成の容易性

この(1)，(2)，(3)を解決するための方針を示しているのが PHI の P と H と I，すなわち並列処理 (Parallel) と階層構造記憶 (Hierarchical Memory) とヒューマンインタフェース (Human Interface) である．

システムの最大性能は，ハードウェアによって決まるが，実効性能は，その他の条件に依存する．その他の条件とは，

（1） 演算能力とデータ供給能力の比率
（2） ソフトウェア効率
（3） 問題プログラムの性質

などである．(1)，(2)，(3)はそれぞれハードウェア，制御ソフトウェアとコンパイラ，アプリケーションに依存して変化する．HPP では，広い範囲のアプリケーションを対象に高い実行効率が得られることを目標とした．

大規模科学技術計算においては，大容量記憶が必要であり，記憶領域の不足が計算時間の増大の原因となる場合も多い．HPP では，階層記憶を導入することによって，大容量記憶の必要性に対応した．

アプリケーション開発を容易に行うためには，開発環境の整備が必要である．HPP のソフトウェアは，コンパイラや制御プログラムを開発環境の 1 ツールと位置付け，その他のツールとの連携を考慮して，この課題を解決しようとした．

1.3 HPP の概要

この解説の中では，以下の 4 通りのアーキテクチャを議論の対象とする．

（1） *UP (Uni-Processor : 単一処理装置)

1 個のプロセッサと 1 個の主記憶装置から構成される．

† Multi-Processor Supercomputer HPP by Sachio KAMIYA and Shin HASHIMOTO (Fujitsu Limited, Program Product Division).
†† 富士通(株) PP 事業部
*：用語解説にあることを示す記号

VP200, S820, SX—2, CRAY—1 などがある.

(2) *TCMP (Tightly Coupled Multi-Processor: 密結合型並列処理装置)

複数個のプロセッサと1個の主記憶装置から構成され主記憶装置には各プロセッサからアクセスできる.

VP2000, SX—3, CRAY Y—MP などがある.

(3) *HCMP (Hierarchical-Memory Coupled Multi-Processor: 階層記憶型並列処理装置)

複数個のプロセッサとプロセッサごとの局所記憶装置と1個の共有記憶装置をもち, 共有記憶装置には各プロセッサからアクセスできる.

HPP はこのアーキテクチャをもつ.

(4) *NCMP (Network Coupled Multi-Processor: ネットワーク結合型並列処理装置)

複数個のプロセッサとプロセッサごとの局所記憶装置をもち, プロセッサ間の通信を行う.

PHI の画像処理部分である. CAP, VPP, DP などを含み, 商用では, CM—5, Paragon, nCube 2 などがある.

HPP のアーキテクチャは, HCMP である. HPP は, 複数の局所記憶装置をもつベクトルマシンを一つの共用記憶装置で結合して構成されている. HPP は, 上記の高速演算能力と大容量記憶を満たすようにこのアーキテクチャを選択した.

HPP のソフトウェアは, 実際に使われている広い範囲のプログラムを開発できるように設計された. それは, 単純なプログラムが動作するというレベルの実験システムや, 特定の問題だけを対象とした専用システムではなく, 実用的な総合システムを目標としたためである.

1.4 HPP と他のシステムとの比較

HCMP を他のアーキテクチャの比較を表-1 に示す.

並列処理において, 結合するプロセッサ台数が多いほうが最大性能は出しやすいことは明らかである. しかし, プロセッサ台数が多くなるほど相対的なデータ供給能力が劣化するため, 対象とする分野が限定される.

われわれは, 広い分野の計算に対して実効性能を出すことを目標としていたため, 階層記憶型並列処理を選択した.

表-1 アーキテクチャの比較

	UP	TCMP	HCMP	NCMP
プロセッサ数	1	2~32	16~128	64 以上
最大性能	小	中	大	大
実効性能/最大性能	大	中	小	小
記憶容量	小	中	大	不定
制御	容易	少し難	少し難	難
粒度	任意	細粒度	中粒度	大粒度
手続き分割	不要	必要	必要	必要
データ分割	不要	不要	必要	必要

2. HPP システムのアーキテクチャ

2.1 HPP の構成

HPP は, メモリの階層構造をもつマルチプロセッサシステムである.

高速演算用並列処理装置 (HPP) は, 共用記憶装置 (*CSU : Common Storage Unit) とプロセッシングエレメント (*PE) とで構成されている (図-1 参照).

PE は, 従来の意味でのベクトルプロセッサであり, 主記憶装置とベクトルユニット, スカラユニットで構成されている. CSU は, 主記憶装置よりはアクセス速度の遅い半導体記憶装置であり, PE 間の通信や, 巨大配列の格納に用いられる. 各 PE は共用記憶制御装置 (*CMU ; Common Mapping Unit) をとおして CSU に結合されている. また, CSU は大容量高速記憶 (*LHS ; Large High-speed Storage) と, LHS 結合装置 (*LHSA ; LHS Adapter) をとおして結合されている.

試用機の実装では PE を4台結合した. PE 0, PE 1 及び PE 3 は最大 2 GFLOPS, PE 2 は最大 5 GFLOPS の演算性能であった. したがって試用機のシステム最大性能は, 11 GFLOPS であった.

2.2 ハードウェアの開発目標と技術的課題

HPP ハードウェアの目標は, 以下の2点であった.

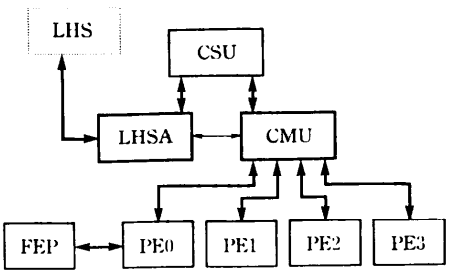


図-1 HPP システム構成図

— 最大処理速度　10 GFLOPS 以上

— 最大転送速度　1.5 GByte／秒以上 (LHS との転送)

そのための技術的課題として,

— システムの並列処理方式

— LHS 接続制御方式

を確立する必要があった. また, 実効性能向上のための技術的課題として,

— PE 間のデータ転送方式

— PE 間の同期処理方式

が重要であった.

2.3 並列処理方式

HPP は, CSU によってベクトル演算機能をもつ PE を複数台結合している. これは, 共用記憶装置と各 PE の主記憶装置 (局所記憶装置) という階層をもつ記憶方式である. CSU はすべての PE から読出し／書き込みが可能であるので, PE 間のデータ転送は, CSU をとおして行うことができる.

並列処理において, PE 間の同時処理 (PE 間通信) は重要である. HPP では, PE 間の同期処理も共用記憶装置をとおして行っている.

2.4 共用記憶制御装置

共用記憶制御装置 (CMU) は CSU と PE の間にあって, CSU と PE の間のデータ転送・PE 間の CSU アクセス競合や同期処理を制御する.

CMU は, CSU・LHSA・複数台の PE に接続されている. CMU のブロック図を図-2 に示す.

PE と CMU のクロックは非同期であり, CMU は, データ以外に CLOCK と DATA VALID 信号を送っている.

PE からのデータは RCV QUEUE→CSU MOVER→CSU INTERFACE をとおって, CSU に転送される. CSU からのデータは, CSU INTERFACE→SEND BUFFER→BUS DRIVER をとおって PE に転送される. RCV QUEUE, CSU MOVER, SEND BUFFER 及び BUS DRIVER はそれぞれ複数あり, 並列動作が可能である.

同期制御はデータ転送から独立している. したがって同期制御とデータ転送とは並列動作が可能である. CMU と PE との間の通信は非同期であり, CMU は PE にクロック信号, データ, データバリッド信号を送る (図-3).

常温 HEMT 素子をバスドライバとして採用することにより, 6 ns でのデータ転送を可能にした.

2.5 LHS アダプタ

LHS アダプタ (LHSA) は, CSU と LSH との間にあって, これらの間のデータ転送を行い, さらに LHS 転送と PE 転送とのアクセス競合や同期処理を制御する.

LHSA は, CSU, CMU 及び LHS に接続されている. LHSA のブロック図を図-4 に示す.

LHSA は 8 KByte のバッファを2組もっている. LHSA は, 一方のバッファで CSU と転送を

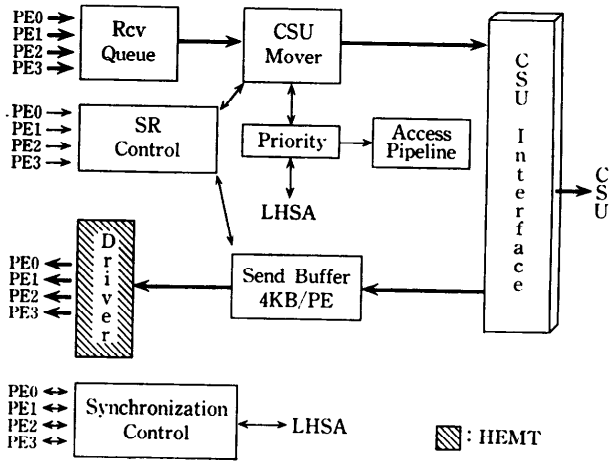


図-2 CMU ブロック図

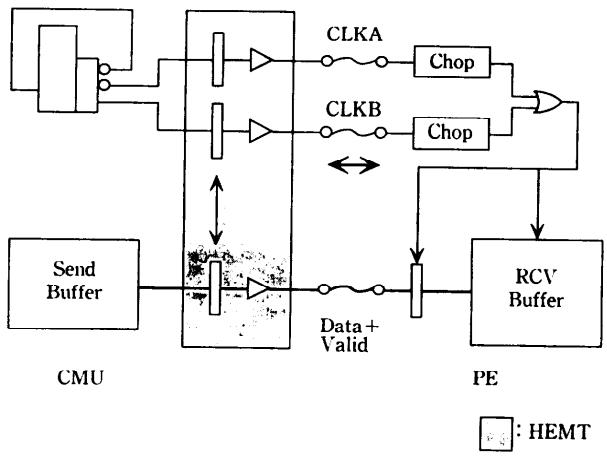


図-3 CMU—PE 間データ転送

行いながら，他方のバッファで LHS との転送を行うことができる．これによって，LHSA では，LHS から CSU へのデータ転送と CSU から LHS へのデータ転送を並行して行うことができる．また，LHS からは 40 ns 周期で 64 バイトのデータが送られてくるが，LHSA 内部では 10 ns 周期で 16 バイトごとにバッファへ書き込んでいる．

このようにして，LHSA では LHS との間の高速転送を実現した．

3. ソフトウェア

3.1 ソフトウェアの課題

密結合型並列処理は商用機でも実現されている．しかし，並列処理のアイデアが古くからあったことを考えると，並列処理が順調に普及してきたとは言い難い．

この原因は主にソフトウェアにある．ひとつには，効率良く並列処理を制御するソフトウェアの開発が難しいとこがある．そして，それよりもはるかに重要な問題は，並列実行するためのアプリケーションプログラムの開発が難しいことである．

さらに，階層記憶型並列処理装置上の並列処理においては，データの割当や転送などの階層記憶管理や，共用記憶アクセスが主記憶よりも低速であることから，密結合型並列処理に比べて粒度の大きい並列処理を行わなければならないという問題がある．

このように，HPP では，粒度が大きいために自動並列化が困難であるような問題を，どのように並列化するかを解決しなければならなかった．この問題を解決するために，HPP では，プログラム記述言語に並列記述を導入し，ユーザが並列処理を記述することをサポートするツール群を構築することを試みた．

また，PHI システムでは，並列処理言語 *PARAGRAM を使って問題プログラムを記述するので，HPP のソフトウェアは，PARAGRAM トランスレータの出力を実行モジュールに変換する機能が必要である．

3.2 ソフトウェアシステムの構成

前項にあげた課題を解決するために，ソフトウェアシステムを，言語系・実行系に支援系を加えた三つのサブシステムから構成することにした（図-5 参照）．

言語系は *Phil と呼ばれる並列処理言語とその翻訳システムからなり，ユーザとのインタフェースの中心となる部分である．

実行系は，階層記憶管理ライブラリや並列処理管理ライブラリなどの制御用ソフトウェアを指す．

支援系は，ソース解析情報や実行時トレース情報をもとに，並列用アプリケーション開発を促進するためのユーザ支援ツールの総称である．支援系が重要であるとの観点から，言語系と実行系には，支援系とのインタフェースを設計時から考慮し，支援系のための機能を組み込んだ．

3.3 HPP 上でのプログラム開発

HPP 上でのプログラム開発は以下のように行う（図-6 参照）．

1) Phil ソースコードを作成する．または，

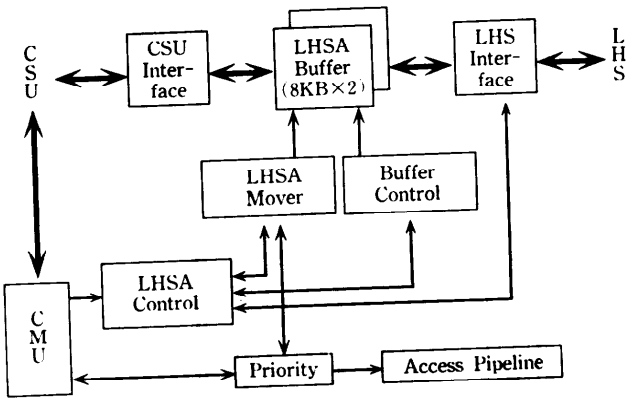


図-4 LHSA ブロック図

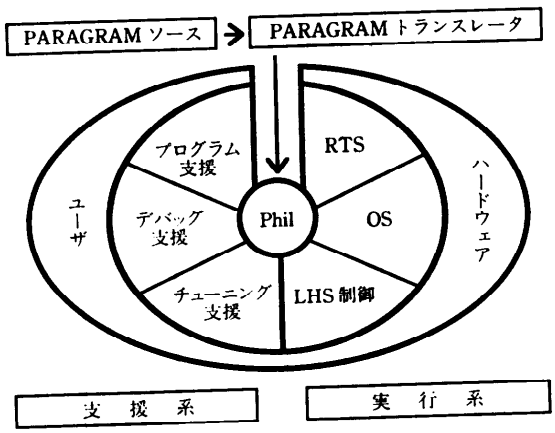


図-5 ソフトウェアシステム構成

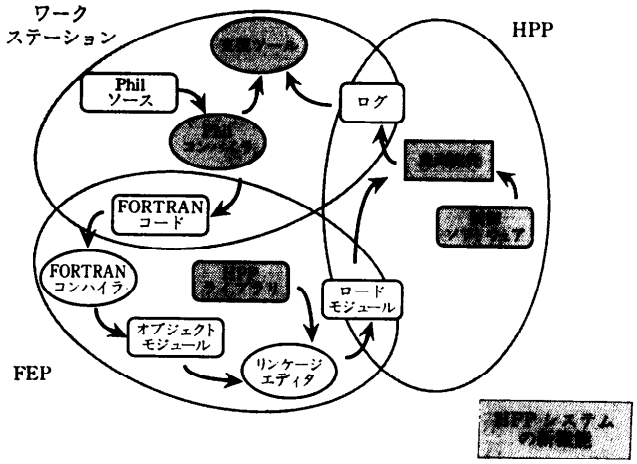


図-6 HPP システム向けプログラム開発

PARAGRAM ソースコードを作成し，PARAGRAM トランスレータで Phil ソースコードに変換する．

2) ワークステーション上で Phil ソースコードを Phil コンパイラにかける．この結果，HPP 制御ライブラリの呼び出しを含む FORTRAN ソースプログラムを得る．また，このとき得られる構文解析情報は，支援ツールによって使用される．

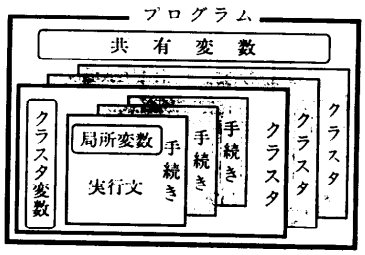


図-7 Phil の構造

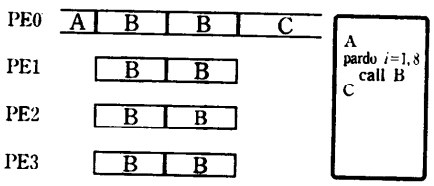


図-8 同種並列処理

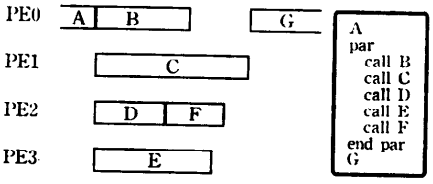


図-9 異種並列処理

3) この FORTRAN ソースプログラムをフロントエンドプロセッサ (*FEP) 上で FORTRANコンパイラで翻訳する．そして得られたオブジェクトモジュールを HPP 制御ライブラリと結合して並列実行可能ロードモジュールを得る．

4) この並列実行可能ロードモジュールを HPP 上で実行する．このとき得られる実行情報は，支援ツールに渡される．

3.4 並列処理言語 Phil の特長

言語 Phil は，階層記憶記述と並列処理記述をもつ科学技術計算向けの言語である．Phil 言語のプログラムは，並列処理のソース上での単位であるクラスタと共有変数で構成されている．クラスタは，クラスタ内共有変数と1個のクラスタ主手続き，複数の副手続きから構成される (図-7 参照)．各クラスタは，FORTRAN のプログラムに相当するものであると考えることができる．

Phil 言語の変数・配列は，スコープ (可視範囲属性) やイクステント (動的／静的割当て属性) のほかに，記憶域属性をもつ．記憶域属性には NEAR と FAR があり，HPP の局所記憶と CSU

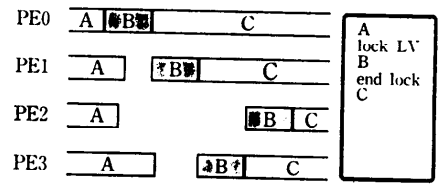


図-10 LOCK 機構

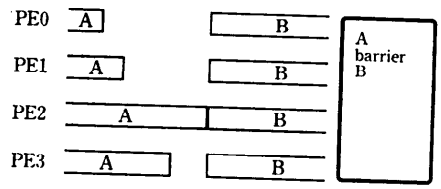


図-11 BARRIER 機構

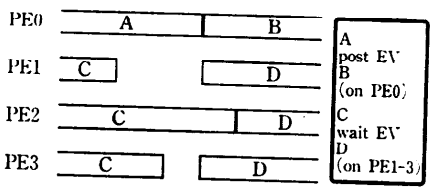


図-12 POST/WAIT 機構

に対応している.

Phil 言語は，並列処理を起動するために二つの並列記述，pardo と par をもつ．pardo は複数個の同じ処理 (homogeneous processes) を並列に実行するための記述である (図-8 参照)．一方，par は複数の異なった処理(heterogeneous processes)を並列に実行するための記述である(図-9 参照)．いずれの並列処理も子手続きが全て終了するまで，起動した親手続きは待ち状態になる．

Phil 言語には，三つの同期機構が準備されて

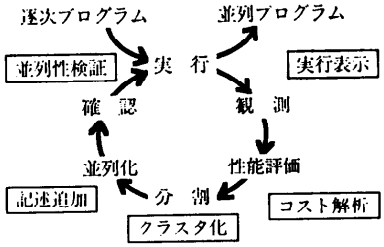


図-13 チューニングサイクル

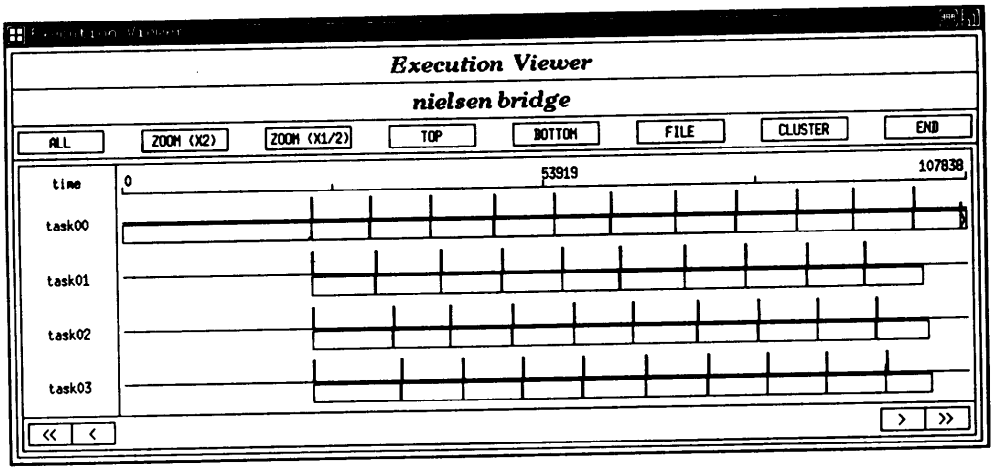


図-14 実行表示ツール

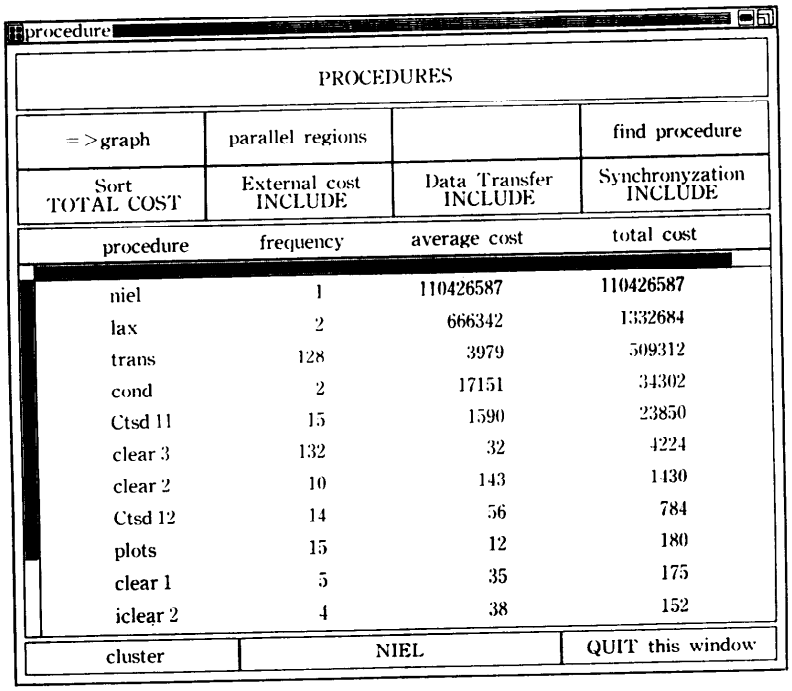


図-15 実行時間解析ツール

いる．LOCK, BARRIER 及び POST/WAIT である（**図-10, 11, 12 参照**）．LOCK は lock 変数と呼ばれる変数を使った排他処理であり図-10 のBの部分は同時には一つしか実行されない．BARRIER は，複数の並列処理単位が特定の場所で待ち合わせる，一斉待ち合わせ処理である．図-11 では，全ての並列処理単位でAが終了するまで，Bは実行されない．POST/WAIT は，event 変数と呼ばれる変数を用いた個別の待ち合わせ処理である．図-12 では，PE0 でAが終了するまで，PE 1～3 のDは実行できない．

3.5 並列処理の実行イメージ

Phil 言語で記述された並列処理は，以下のように実行される．

並列処理の実行単位（クラスタインスタンスまたはプロセスと呼ばれるもの）が pardo あるいは par に到達すると，そのクラスタインスタンスは実行を中断する．そして，pardo あるいは par で呼び出されているクラスタのインスタンスが並列に実行される．それらの実行が全て終了すると実行を中断していたクラスタインスタンスが実行を再開する．

また，pardo あるいは par で呼び出されているクラスタのインスタンスの並列実行の順番は，原則として動的に決定される．このようにダイナミックスケジューリング方式を採用したため，システムのプロセッサ構成が変化した場合に，その構成での CPU 能力を有効に生かすことができる．

3.6 支援ツールの特長

並列科学技術計算プログラムの開発は難しいと言われている．われわれは，並列プログラムの開発モデルとしては，以下の三つを検討した．

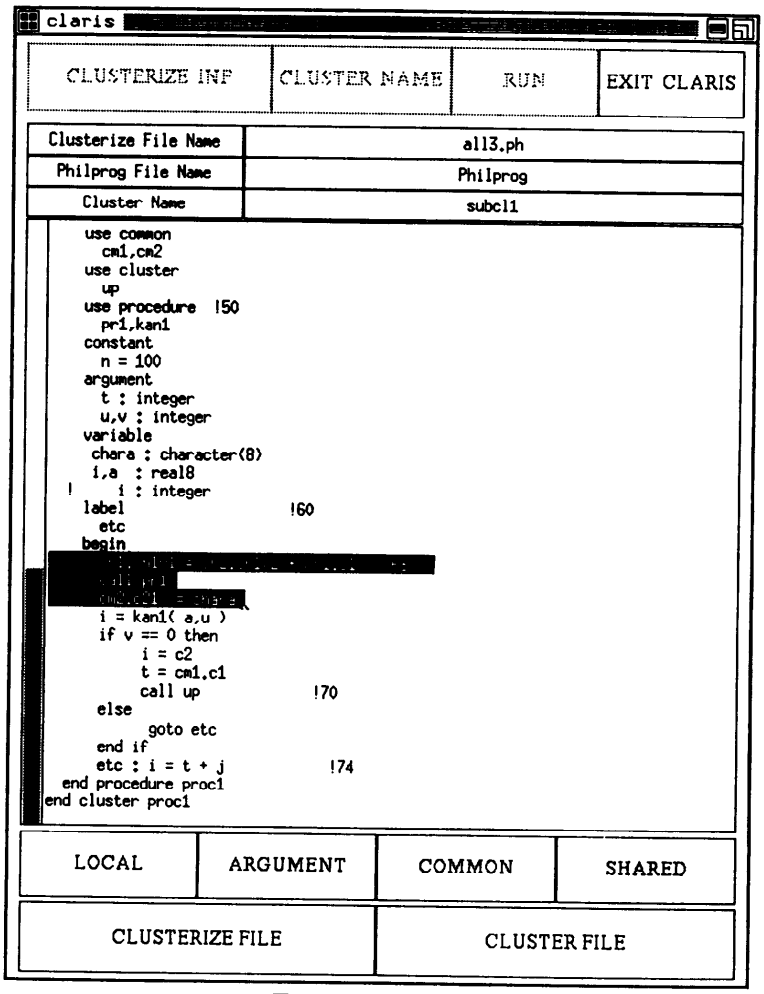


図-16 クラスタ化ツール

(1) 逐次プログラムをコンパイラが並列化して実行する.

これは,演算レベルの並列性や,TCMP の場合に適用されるモデルである.

(2) 逐次プログラムを作成してそれを並列化してコンパイルし実行する.

これは,自動並列化が困難なシステムに対して有効である.

(3) 並列プログラムは直接作成しコンパイルして実行する.

このモデルは,並列性が明確であるような問題について有効である.

われわれは,(2)のプログラム開発モデルを想定し(**図-13** 参照),このモデルの開発効率を高めるためのツールについて検討を行った.

われわれのモデルでは,開発者はまず逐次プログラムを作成し,正しい答えが得られるまでデバッグを行う.次に,そのプログラムを並列化していくことによってパフォーマンスの良い並列プログラムを得る.並列化では,プログラムを実行して実行の様子を観測し,効率を測定して並列化する部分を決め,その部分を分割して並列に実行するようにし,さらにその実行結果がそれまでと変わらないことを検証する.

この並列化の各作業を支援するために,実行表示,実行時間解析,クラスタ化,並列記述追加及び並列性検証の各ツールを作成した.

1) 実行表示ツール

実行表示ツールはグラフを使って実行の状態を表示するツールである.時間を横軸に,プロセッサを縦軸にとって,各プロセッサがある時点でどのクラスタを実行していたかを表示する.また,プロセッサと CSU とのデータ転送がプロセッサグラフの上に折れ線グラフで表示される.

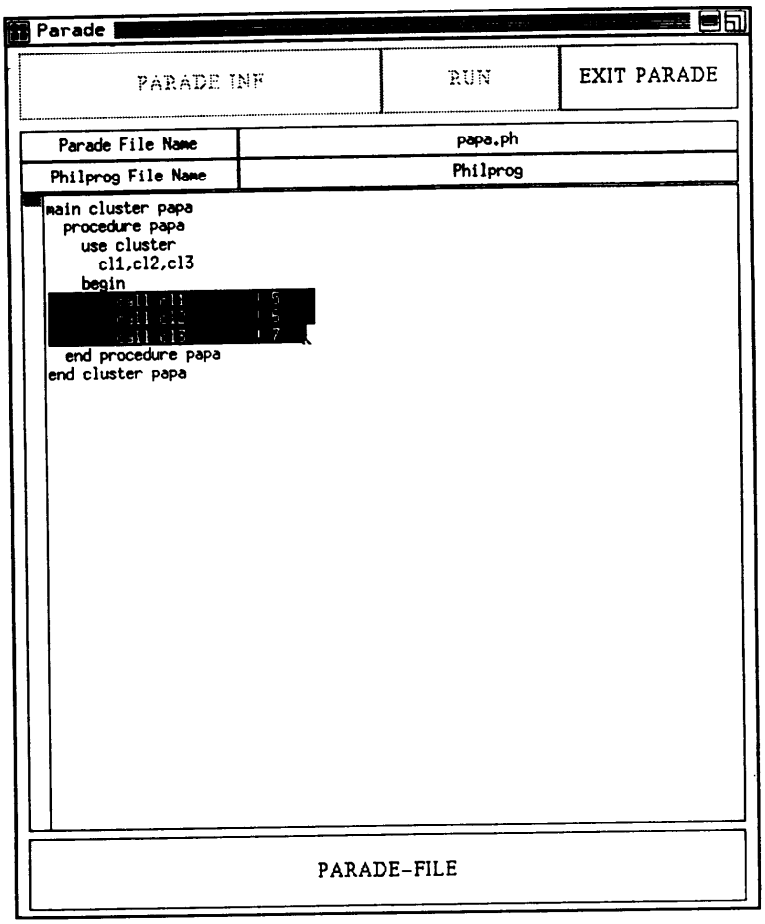


図-17 記述追加ツール

このツールを使うことによって，ユーザは負荷バランス，データ転送量，逐次実行部分の実行時間などを容易に知ることができる（**図-14** 参照）．

2) 実行時間解析ツール

実行表示ツールが動的な並列処理の状態を表示するのに対して，実行時間解析ツールはソースに対応した実行時間の情報を表示する．このツールは，Phil プログラムの階層構造に従って，プログラム全体，各クラスタ，各手続き，各 DO ループについて，それを実行するためにかかった時間とデータ転送量を表形式またはグラフ形式で表示する．これは，チューニングを行うべき場所を決定するために，最も時間のかかるクラスタを調べ，その中の最も時間のかかる手続きを調べるという作業を容易に行うためである．プログラム構造の各階層に対して段階的に詳細な情報を表示する機能を備え，各レベルにウィンドウを一つ割り当てるようになっている．**図-15** は，1 個のクラスタに含まれる手続きの実行時間を表形式で表示するウィンドウである．並列処理のチューニングにおいて，その内部で呼び出される手続きを含む実行時間の情報と含まない実行時間の情報が共に必要であるため，ボタンによって表示する情報を切り換える機能がある．

3) クラスタ化ツール

クラスタ化ツールは，ユーザが指定したプログラムの一部を別のクラスタ（並列処理のソース上の単位）として分割するためのツールである．クラスタ化ツールは，手続き間解析を行うことによって，クラスタ化される部分の各変数をどのような変数にするべきかを判断して割り当てる．しかし，人間の判断により，この割り当てを変更することも可能であるように設計されている．**図-16** の LOCAL, ARGUMENT, COMMON, SHARED

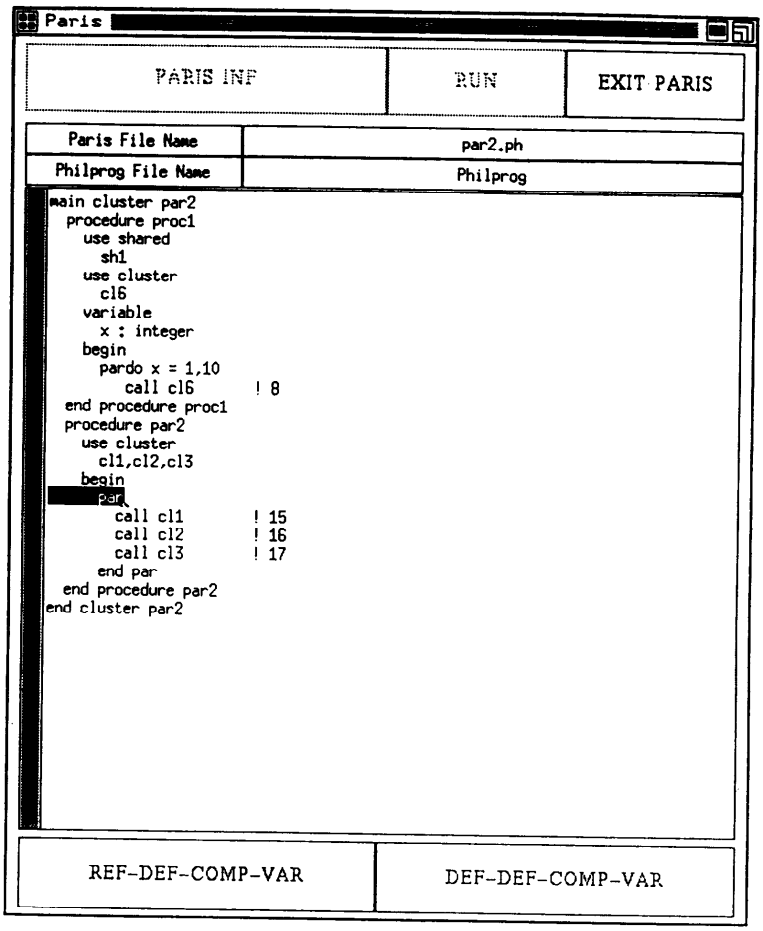


図-18 並列性検証ツール

というボタンを選択すると，それぞれに割り当てられた変数のリストが表示されるので，変数を選択して別の属性を指示することができる．

4) 並列記述追加ツール

このツールは，ユーザが指定した部分を並列呼び出しに変更する機能をもつ（**図-17** 参照）．

5) 並列性検証ツール

並列処理では，何回か実行を繰り返したときに，実行結果が変わる場合がある．これは，並列に実行される部分の，実行順序に依存するような変数の値を使用している場合に起こるものであり，デバッグすることが難しい．並列性検証ツールは，このような実行順序に依存する結果の不安定性を検出するツールである．**図-18** のウィンドウでチェックの対象となる並列処理部分（par または pardo）を選択すると，このツールは，並列処理中に出現する全ての変数の中で，同時に実行される可能性のある複数の並列処理単位で共に定義されているか，または一方で定義され他方で参照されているものを検出して表示する（**図-19** 参照）．

4. 評　価

平成2年1月に，HPP システムを実際に動作させた．

階層記憶型並列処理を行うことによって，数十台の演算装置を結合する技術が開発された．

ハードウェアの最大演算性能 10.9 GFLOPS を実測した．これによって目標の 10 GFLOPS 以上を達成した（**図-20** 参照）．

CSU—LHS 間の転送では，目標転送性能 1.5 GByte/秒に対して，最大転送速度 1.6 GByte/秒を達成した．

CMU によって，並列処理用同期通信機能と共用記憶アクセス制御機能を実現した．また，常温 HEMT 素子を CMU—PE 間のバスドライバとして使用し，高速転送を達成した．これは，世界で始めて計算機に対する HEMT の実装である．

原子力研究所コード SPIN，土木工学コード NIEL，円周率計算，行列計算などのプログラムを Phil 言語と PARAGRAM 言語で記述し，並列実行を行った．

4台の PE（それぞれ最大性能 2 GFLOPS）を結合して，2.1 倍から 3.8 倍の台数効果を得た（**図-21** 参照）．

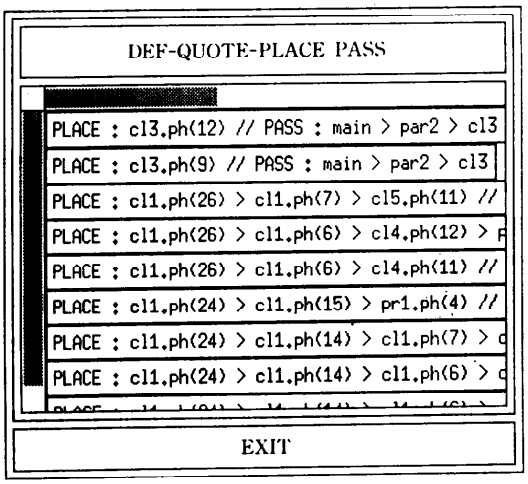


図-19 並列性検証ツールの検出結果

```
Performance Test Started

First start time (A)              :  15 : 57 : 14.110.213
First end time (B)                :  15 : 57 : 15.954.212
First calculation count (X)       :  16358400 (K)
Second start time (C)             :  15 : 57 : 15.954.229
Second end time (D)               :  15 : 57 : 19.296.427
Second calculation count (Y)      :  32716800 (K)
Calculation time
    T=(D-C)-(B-A)                 :  1498199(usec)
Calculation performance
    (Y-X)/T                       :  10918.8 (MFLOPS)
```

図-20 最大性能の実測結果

プログラム	特　徴	目　的	1PE	4PE
SNBS	三重ブロック対角行列	PARAGRAM の記述性	281	101 2.78
SPIN 2**20要素 36温度点	イジングモデルによるスピングラス法		1811	479 3.78
SPIN 2**26要素 1温度点		LHS の検証	5936	1855 3.20
LU 16K次元	密行列の LU 分解		20181	6300 3.20
PI 1000000桁	円周率計算	高速性	3521	1375 2.56
NIEL	土木工学計算	支援ツールの効果	228	108 2.11

上段：エラプス時間（秒），下段：1台に対する比率

図-21 実プログラムの性能

2 GFLOPS の PE を3台と 5 GFLOPS の PE を1台結合して，さらに高速に実行できることを確認し，非対称なシステム構成におけるダイナミックスケジューリングの有効性を確認した．

さらに，支援ツールを使用して，1週間程度か

かる並列化の作業が1日に短縮できることを確認した.

5. まとめ

HPP では，階層記憶をもつ並列処理システムのハードウェアとソフトウェアを開発し，その上で実際に用いられるレベルの科学技術用アプリケーションプログラムを並列に実行した.

並列処理は逐次処理と比較して，並列処理制御のオーバヘッドとデータ供給のオーバヘッドをもつために，適用できる分野が限定される．オーバヘッドを減らすためには，プロセッサ間の同期処理の高速化，データ転送の高速化を行うハードウェアが必要である．HPP では，共用記憶装置と共用記憶制御装置を開発することによって，これに対応した．今後，同期処理は専用ハードウェアが主流になり，さらに高速化するものと考えられる．データ転送の高速化は必要とされる素子の数が多いために，ハードウェア的には限界があり，ソフトウェアによる解決が主流となるであろう.

ソフトウェアでは，制御ソフトウェアのオーバヘッドを少なくすることと，並列処理の粒度を大きくすることが重要である．HPP では，効率の良い制御ロジックを検討して実装した．また言語に並列処理記述を追加して，並列化支援ツールと併用することによって大粒度化を行った．将来においても，ソフトウェア的に効率を向上することは難しいと考える．むしろ，効率の良い制御ロジックに加えて理解しやすいモデルをもつものが主流になると考える.

並列性認識技術による自動並列化と，並列処理記述をもつ並列処理言語は今後の並列処理において，共に必要とされる．並列処理言語は，われわれが考えた機能とほぼ同じ機能をもつ言語も現れている．並列処理が普及するためには並列処理言語の標準化を急ぐ必要がある.

並列プログラムの開発環境は徐々に整備が進んできている．各研究者・メーカの努力が無駄にならないためにも，並列処理言語の標準化だけでなく，ヒューマンインタフェースを含めた標準化に期待する.

性能評価のためのアプリケーションプログラムの開発においては，大規模科学技術計算における，大容量記憶の重要性を痛感した．外部記憶装置の高速化だけでなく，主記憶装置の大容量化が必要であり，アドレス空間の拡張は必須であると考える.

「科学技術用高速計算システム」の研究開発が終了して2年が過ぎた．この2年間の技術の飛躍には目を見張るものがある．並列処理には解決すべき問題も数多く残っている．研究者・メーカ・ユーザの努力と協力により，並列処理の時代が到達することを確信している.

謝辞 ご協力いただいた，日本原子力研究所物理部 別役主任研究員，日本原子力研究所東海研究計算センター 横川氏，東京大学大型計算機センター 金田助教授，京都大学工学部土木工学教室 渡邊教授に感謝する.

参考文献

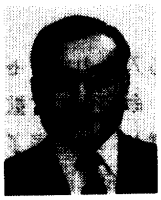
橋本他：階層記憶型並列処理，情報処理学会第41回全国大会 (1990).
加藤他：階層記憶型並列処理の制御ソフトウェア，情報処理学会第41回全国大会 (1990).
渡辺他：並列処理におけるプログラム開発支援システム，情報処理学会第41回全国大会 (1990).
橋本他：Hierarchical Memory Connected Multi-Processor，電子情報通信学会 CPSY 90—5 (1990).

(平成3年11月28日受付)

用語解説

HPP：高速演算用並列処理装置
High-speed Parallel Processor
PHI：総合システムの名称
Parallel Hierarchical Interface
MFLOPS：100万浮動小数演算/秒
GFLOPS：10億浮動小数演算/秒
HEMT：高電子移動度トランジスタ
UP：単一処理装置
Uni-Processor
TCMP：密結合型並列処理装置
Tightly Coupled Multi-Processor
HCMP：階層記憶型並列処理装置
Hierarchical Memory Coupled Multi-Processor
NCMP：ネットワーク結合型並列処理装置
Network Coupled Multi-Processor
CSU：共用記憶装置
Common Storage Unit
PE：プロセッシングエレメント

Processing Element
CMU：共用記憶制御装置
Common Mapping Unit
LHS：大容量高速記憶
Large High-speed Storage
LHSA：LHS 結合装置
LHS Adapter
PARAGRAM：問題向け並列処理言語
Phil：PHI 用言語
PHI Language
FEP：フロントエンドプロセッサ
Front End Processor



神谷 幸男（正会員）
1949 年生．1972 年名古屋大学理学部卒業．同年富士通(株)入社．FORTRAN コンパイラの開発に従事後，スーパコンピュータ向けソフトウェアの設計・開発に従事．

橋本 伸（正会員）
昭和 30 年生．昭和 53 年京都大学理学部卒業．昭和 60 年大阪市立大学大学院理学研究科卒業．理学博士．同年富士通(株)入社．記号処理言語の開発，並列処理システムの開発，スーパコンピュータ向けソフトウェアの設計・開発に従事．