724. 並列計算機 PACS-32 による BWR 炉心計算

星 野 力*

(1981年 2月3日 受理)

(1981年 6月25日 再受理)

Attempt of BWR Core Simulation by Parallel Computer PACS-32

By Tsutomu HOSHINO

A three-dimensional BWR core simulation has been attempted with the use of PACS-32, a newly developed parallel microcomputer system. The PACS system is an MIMD-type microprocessor complex, consisting of 32 processors, that was specially designed for the numerical analysis of the partial differential equations.

The neutron diffusion model in the simulation is based on the modified two-group finite-difference coarse-mesh method, and the thermo-hydraulic model assumes the energy balance in the stationary state.

The analysis of the parallel processing program revealed that the system overheads, *i.e.* the loss of computation power pertinent to the parallel processing, is independent of the number of processors, and that the efficiency of processors, *i.e.* the ratio of the effective calculation over the total calculation, amounts to 70%, and even upto 88% if the computation is limited to the reactor core region.

By the measurement of the execution time, it has been demonstrated that the computation speed of nearly twice as fast as a middle size general purpose machine is realized, with 20 times better cost-performance ratio. The PACS system could be used as the practical BWR core simulator for more complicated core models.

KEYWORDS: *three-dimensional calculations, parallel microcomputer system, computer architecture, MIMD-type, numerical analysis, partial differential equations, modified two-group, coarse mesh, system overheads, reactor simulators, BWR type reactors, reactor core, performance, computerized simulation, computer codes*

I. はじめに

動力炉運転時の炉心内部の核的熱的特性を計算することは、すでにルーチンの業務として、炉の設計や運用に携わるオフィスでひんばんに行われている。これは数値計算という観点から見れば、偏微分方程式の数値解析に属するものであり、もっぱら大型汎用計算機が用いられている。計算機の技術動向としては、汎用機のいままでの速度向上を実現してきた論理素子の速度に、光速と冷却方法などの原理的技術的限界が見え出していることや、汎用機の大形化、特にソフトウェアシステムの巨大化に伴い信頼性や生産性に難しい問題が生じてきている。

一方、1970年代後半から本格的実用期に入った大規模集積回路(LSI)は計算機ハードウェアの価格を大幅に低下させた。このため、各問題あるいは分野ごとに最適に設計された専用計算機によって、性能価格比の大幅な向上が期待できるようになり、いままでの常識であった「汎用の計算機を多数の利用者が共同利用する」という形態を変えることが予想される。

例えば、炉心の3次元核熱的特性計算は、計算機中のデータの流れ方に著しい特徴を有する。物理的には近接作用により形成されている、流体、熱の移動や中性子拡散現象であり、数学的には偏微分方程式の有限差

* 筑波大学構造工学系(Inst. Structural Eng., Univ. of Tsukuba)

分近似をすれば容易にわかるように、データは隣接または近接格子点間でやりとりされるだけである。したがって、多数の高集積化されたプロセッサあるいはコンピュータを配置して(プロセッサ・アレイと呼ぼう)空間格子点と対応させ、隣接または近接プロセッサ間のデータ通信を行えるようにすれば、計算はすべてのプロセッサにおいて並列に進行し、ほぼプロセッサの台数に比例した計算速度向上が得られるはずである。

もっとも、すべての計算が並列処理可能なのではない。例えば、入出力には本質的に逐次処理にならざるをえない部分がある。しかし、計算時間のほとんどを占めているのは、偏微分方程式を過緩和法などの繰り返し法で解く部分であり、この部分の並列処理は計算時間の著しい短縮をもたらす。もちろん通常の計算機による逐次の処理では現れない並列特有の仕事も生ずる。例えば、隣接プロセッサとのデータ交換、全部あるいは部分的にプロセッサ間の同期をとる仕事などである。しかし、ハードウェアをうまく設計することによってこれらの雑用(オーバーヘッド)を少なくすることが可能である。

本報では、32台のマイクロプロセッサを2次元的に配置したプロセッサ・アレイ PACS-32により、3次元BWR炉心の定常核熱水力計算が可能であることを報告する。筆者らはPACSを一般に近接作用が支配する連続体・場などの偏微分方程式系の数値シミュレーション専用として試作し、1979年に9台のマイクロプロセッサより成るシステムPACS-9で、中性子拡散方程式が解けることを報告した⁽¹⁾。1980年に、これに多少の改良と拡張を行なったのがPACS-32である⁽²⁾。

マイクロコンピュータの著しい普及に伴い、すでに類似の考えに基づく提案と性能予測がなされている。例えば、Honeck⁽³⁾による評価では、計算機のアーキテクチャがPACSと異なって、プロセッサ間の通信をミニコンピュータに逐次的に行わせるなど、オーバーヘッドの大きいシステムを想定している。PACSは偏微分方程式系のシミュレーション用途において逐次的処理が少なくなるよう最適に設計されており、Honeckの予測を上回る性能が得られる。PACSは単なる仮空のマシンではなく実機であり、BWRのシミュレーションもほぼ実用規模のモデルを用いて行い、この種のアイディアが実現可能であることを実証した。

本報では、PACSの概要を紹介し、次にBWR炉心シミュレーションのモデル、並列計算技法、シミュレーション例と評価について述べる。最後にPACSの可能性について将来へ外挿した議論を行う。

II. PACS の概要

PACS(Processor Array for Continuum Simulation)は、流体などの連続体や電磁界のような場を記述する偏微分方程式の並列処理技術の研究を目的として試作したマイクロプロセッサ複合体である⁽²⁾⁽⁴⁾⁽⁵⁾。システムは32台のPU(Processing Unit)より構成されている。

(1) 場は物理的には「近接作用」により形成されているから、自然現象のメカニズムに即したシミュレーションを行うことにし、隣接PU間のデータ交換を効率的に行えるようにしてある。また同じ理由ですべての点で同時に同じ物理のプロセスが進行することはないから、すべてのPUでの命令とデータの流れが異なりうるMIMD(Multiple Instruction-Multiple Data Stream)方式⁽⁶⁾を採用している。

(2) すべてのPUとCU(Control Unit)を結ぶ1本のコモンバス(処理装置やメモリー等を結合する共通の通信制御線)があるが、これはプログラムと初期データのロード、計算結果の回収にのみ用いる。PU間のデータ通信は、隣接PU間におかれ、クロックに同期して時分割で両側のPUに結合される通信用メモリーCM(Communication Memory)を介して行われる。PU相互間やPUからCUへの割込みは不可能とし、アクセス競合をできるだけ避けるようになっている。基本的にはPUをアナログコンピュータの演算増幅器に対応させたようなアーキテクチャと考えることができる。

(3) 問題を並列処理可能な部分に分解し、これらをPUに割りあて、またPU間の同期をとることは、対象としている問題の構造に応じて行うのが最適であるから、問題を熟知しているユーザにまかせることにし、それを容易に行えるようなソフトウェアを開発した。

(4) 汎用計算機(HOST)のバックエンド・プロセッサとしてPACSシステムは数値計算に専念し、HOSTのオペレーティングシステムの支配下に入るようになっている。

ハードウェアの概略をFig. 1, 2に示す。CUとPUはモトローラ社8ビットマイクロプロセッサMC 6800を用い、各PUは18キロバイトのメモリーと浮動小数点ならびに関数演算用マイクロプロセッサAm 9511をもつ。CUとPUはアドレス空間を共有しCUからPUへ一斉に同内容の情報を放送することも可能である。

PUは両端がエンドレスになっている8×4の2次元アレイになっている。このアレイを縦横に通る行列制

御線により、CU から PU ごとに制御することも可能である。メモリー CM (Communication Memory) は、MC 6800 の 2 相クロックの一つ ϕ_2 に同期して時分割で両側の PU に結合される。PU は 8×4 のアレイ上で、チェッカーボード状に 2 種類の PU (偶 PU と奇 PU) に分けられ、両組の PU は互いにクロック ϕ_1, ϕ_2 が反転している。MC 6800 では ϕ_2 が高であるときのみプロセッサはバスを使用すると考えてよいので、このような構成によって、CM に対し両側のプロセッサからのアクセスが競合することはない。

CU から PU への指令は 4 ビットの CR (Control Register) を経由して PU に伝えられる。逆に PU から CU への状態報告は、各 PU ごとに 4 ビットの SR (Status Register) に書き込まれ、全 PU の SR の AND (論理積) と OR (論理和) をとったものを CU から読み取ることができる。並列処理中の同期や並列処理の終了判定はこの CR と SR を介して高速に行われる。

ジョブの並列処理の進み方は Fig. 3 に示す通りである。まず、ソースプログラムはアセンブラ言語 MASP、コンパイラ言語 SPLM によって記述される。このコンパイラ言語は構造化プログラム言語 SIMPLE のを母体として、入出力を省き PACS のメモリーの種類に応じた変数宣言文をつけ加えたものである。HOST によって翻訳された目的プログラムは、原則として全 PU について同一であるが、別別のプログラムをロードすることも可能である。実行は HOST 中ではじまり制御が CU に移る。CU は PU に演算開始指令を与え、PU から SR 経由で終了ステータスが返るのを待つ。PU は種々のモードの並列処理を行い、終了を自己判定して終了ステータスを SR に置く。並列処理のモード、とくに同期モードと並列処理特有の技巧については後述する。

III. BWR 炉心モデル

モデルの構造を PAD (Problem Analysis Diagram, [付録] 参照) で表現したものを Fig. 4 に示す。まず、炉心の燃料集合体の配置などの幾何学的パラメータ、中性子束ループ繰返し計算回数やボイド分布繰返し回数などのパラメータ、ボイド依存の 2 群炉定数などの核的パラメータおよび出力分布の初期推定を入力し、ボイド分布収束ループに入る。初期推

CRT: Cathode ray tube, CDR: Card reader, PTR: Paper tape reader, LP: Line printer, DP: Disc pack, MT: Magnetic tape, KB: Key board, CMT: Cassette magnetic tape, PU: Processing unit

Fig. 1 System configuration of PACS-32

PU: Processing unit, MPU: Micro processing unit, APU: Arithmetic processing unit, CM: Communication memory, LM: Local memory, RM: Result memory, RCC: Row-column controller

Fig. 2 Configuration of processing unit

Fig. 3 Time chart of job processing

Fig. 4 PAD representation of BWR core model

定出力分布を出力分布 P とおいてボイドの計算を行う。ボイド計算は定常状態におけるエネルギー釣合い式

$$\alpha \rho_g V_g h_{fg} A_c = \int_0^z P(z) dz, \quad (1)$$

入口における流量の釣合い式

$$\alpha V_g \rho_g + (1 - \alpha) \rho_f V_f = V_i \rho_f \quad (2)$$

と、気液二相の速度関係式

$$V_g = s V_f \quad (3)$$

ただし、 α : ボイド率、 ρ_g : 蒸気密度

V_g : 蒸気流速、 ρ_f : 水の密度

V_f : 水の流速

h_{fg} : 蒸発エンタルピー

A_c : チャネルの断面積

S : スリップ比(一定とする)

V_i : 入口冷却材流速

$P(z)$: チャネル上方向の出力密度分布

とより近似的に求めている。仮定 $\alpha \rho_g \ll (1 - \alpha) \rho_f$ の下では、 α は簡単に

$$\alpha(z) = \frac{Q(z)}{C + Q(z)} \quad (4)$$

と解ける。ここに

$$C = V_i s h_{fg} A_c \rho_g$$

$$Q(z) = \int_0^z P(z) dz \quad (6)$$

入口にサブクールがある場合は、サブクールパワー Q_{sub} を

$$P_{sub}(z) = \frac{2Q_{sub}}{H} \left(1 - \frac{z}{H}\right) \quad (7)$$

ここに、 H : 炉心高さ
のように z 方向に分布させる。すると、(6)式は

$$Q(z) = \int_0^z [P(z) - P_{sub}(z)] dz \quad (6)'$$

におきかえられる。領域 $Q(z) < 0$ では、 $\alpha(z) = 0$ とおいておく。

このサブクールパワー Q_{sub} は炉心と炉心外のループにおけるエンタルピーの釣合いより

$$Q_{sub} = \frac{Q_i(h_{sat} - h_{fw})}{(h_{sat} + h_{fg} - h_{fw})} \quad (8)$$

ただし、 Q_i : 炉心熱出力

h_{sat} : 飽和エンタルピー

h_{fw} : 給水エンタルピー

として得られる。

このようにして求められた3次元ボイド分布 α の2次関数として、各点における拡散係数 $D(\alpha)$ と核定数 $\Sigma(\alpha)$ を求める。この2次関数は燃料のタイプ、濃縮度、燃焼度の違いに応じて入力データとして与えられるものとする。また、初期熱出力分布と同形の初期核分裂中性子源分布 S と、その体積積分 λ を求めておく。

次に、中性子拡散方程式

$$\nabla D \nabla \phi - \Sigma_r \phi = -S/\lambda \quad (9)$$

を解くループに入る。この中性子拡散モデルは、3次元粗メッシュ差分法⁽⁸⁾(PRESTO コードに用いられているもの)に基づいている。

この PRESTO モデルは単純化された2群拡散モデルで、速中性子束については有限差分近似を用い、熱中性子束については漸近的な取扱いをしている。またノード境界における中性子束の勾配は、理論的な考察よりきまる重み因子を用いて近似的に与えられている。

また、差分スキームに現われる拡散係数の算術平均を幾何平均で近似し、このため中性子束 ϕ に拡散係数の平方根 $\sqrt{D}$ を乗じたものをコード中では取り扱っている。これらの技巧により十分な精度を保ちながらデータのメモリー量を少なくでき、またプログラム、したがって実行時間も短くできる。また7点階差スキームであることは、隣接ノードのみとのデータ交換で済むことを意味し、PACS 向けの解法として適している。

並列処理の技法を説明する必要があるのと、原モデルからの変更箇所も多少あるので、ここで採用した中性子拡散方程式(9)の解法を少し詳しく述べておく。まず、規格化 $\phi = \phi \sqrt{D}$ を行なって速中性子群に対する

(9)式と同形の式の差分近似を第 i ノードについて書く、

$$Q_i \phi_i = T_i \bar{S}_i + (\sum_{4j} \phi_j + R \sum_{2j} \phi_j) \quad (10)$$

ただし、

$$T_i = \sqrt{D_i} / \left(\frac{D_i}{h^2} - \sum_{r,i} c \right) \quad (11)$$

$$Q_i = T_i \left(q_i \frac{\sqrt{D_i}}{h^2} + \frac{\sum_{r,i} (b + c r_i c)}{\sqrt{D_i}} \right) \quad (12)$$

$$q_i = \frac{1}{\sqrt{D_i}} (\sum_{4j} \sqrt{D_j} + R \sum_{2j} \sqrt{D_j}) \quad (13)$$

$$R = h^2 / k^2 \quad (14)$$

$\bar{S}_i$: ノード i で平均化された規格化されて
いない核分裂中性子源

h : 水平方向ノード間隔

k : 垂直方向ノード間隔

$\sum_{4j}$: 水平方向隣接4ノードにおける和

$\sum_{2j}$: 垂直方向隣接2ノードにおける和

b, c : 前述の理論的考察により決まるパラメータ

これらの式の導出で隣接ノードにおける和をとる際、隣接ノードが炉心外に出た場合は $\phi_j = 0$, $D_j = D_i$ (すなわち境界ノード i における値) とする。差分式(10)の右边を $\phi, \bar{S}_i, \lambda$ の初期推定値を用いて計算し、左辺から ϕ_i を求める。これを用いて新たに(10)式の右边を計算し、この中性子束繰返しループを I_n (入力パラメータ) 回繰り返す。

次に、この ϕ_i を用いて熱中性子束を求める。

まず、ノード内平均速中性子束

$$\bar{\phi}_i = (b + c r_i) \phi_i + c (\sum_{4j} \phi_j + R \sum_{2j} \phi_j) \quad (15)$$

漸近熱中性子束

$$\phi_{ii} = \left(\frac{\sum_{r,i} \sqrt{D_i}}{\sum_{a,i} \sqrt{D_i}} \right) \bar{\phi}_i \quad (16)$$

ノード内平均熱中性子束

$$\bar{\phi}_{ii} = (b_i + c r_{ii}) \phi_{ii} + c_i (\sum_{4j} \phi_{ij} + R \sum_{2j} \phi_{ij}) \quad (17)$$

核分裂中性子源

$$\bar{S}_i = \nu_i \sum_{f,i} \frac{\bar{\phi}_i}{\sqrt{D_i}} + \nu_{ii} \sum_{f,ii} \frac{\bar{\phi}_{ii}}{\sqrt{D_{ii}}} \quad (18)$$

核分裂密度

$$F_i = \sum_{f,i} \frac{\bar{\phi}_i}{\sqrt{D_i}} + \sum_{f,ii} \frac{\bar{\phi}_{ii}}{\sqrt{D_{ii}}} \quad (19)$$

固有値(臨界係数)

$$\lambda = \int_{\text{core}} \bar{S}_i dv \quad (20)$$

を次々に求め、前回の核分裂源 $\bar{S}_{i,old}$ と今回の $\bar{S}_i$ から核分裂源の規格化と加速を行う。

$$\bar{S}_{i, \text{new}} = \frac{\bar{S}_i}{\lambda} + \theta \left(\frac{\bar{S}_i}{\lambda} - \bar{S}_{i,old} \right) \quad (21)$$

これを用いて再び(10)式にもどり、この核分裂源繰返しループを I_0 (入力パラメータ) 回繰り返す。ただし、初回は(21)式で $\theta = 0$ として加速は行わない。コード中では簡単のため、核分裂源繰返しの最終回以外では熱中性子は拡散しないとして、(17)式における $\sum_{4j} \phi_{ij}$ と $\sum_{2j} \phi_{ij}$ をそれぞれ $4\phi_{ii}$, $2\phi_{ii}$ とおいた。

このようにしてループが終了し、1つの核分裂密度 F_i が求められると、新たな熱出力分布 P_i を

$$P_i = \bar{P} F_i V / \int_{\text{core}} F_i dv \quad (22)$$

ただし、 P : 炉心平均熱出力密度、 V : 炉心体積により求め、これを用いて再びボイド分布の計算(4)式にもどる。このボイドループを I_n (入力パラメータ) 回繰り返す。このボイドループが終了すると、炉心の熱出力分布 P とボイド分布 α が求まり、これをプリンタに出力して計算は終了する。

IV. 並列処理の計算技法

並列処理のモードは次の3つのモードが可能である。

1. 同期モード

プログラム中必要な個所でサブルーチン SYNC(CODE) を CALL することにより、各 PU から同期符号 CODE が CU に送られ、全 PU からの CODE がそろった時点で、PU に統行指令が与えられる。これは、全 PU のプログラム進行を命令ステップごとに同期させるといったものではなく、同期をとりたい時点で歩調をそろえるにすぎない。同期をとった直後は、もしプログラムがすべて同一であれば、命令レベルでも実行の同期が保たれているが、処理速度がデータに依存する演算や条件分岐があれば、当然それ以後のプログラムの実行はバラバラになる。この同期モードはラプラシアンなどの偏微分作用素の差分近似における隣接ノードデータのとり込みや、後述するルーティング手法に用いられるモードである。

2. 非干渉モード

PU が他の PU の動作と無関係に処理の開始、終了、CU とのデータ転送を行うもので、モンテカルロ法などの処理に用いられるほか、特定の PU から CU と HOST を経由してデータを出力するなど、デバッグ時の出力にも用いられる。

3. 非同期モード

隣接 PU でのデータが揃えば計算がスタートできる場合、データに応じて処理が進んでゆくデータ駆動的動作が可能である。データの受け渡しは、CM メモリー上で行い 1 個のデータ・レディ・フラグ(0 でデータが書き込み可であることを示し、1 で読み出し可であることを示す情報)を用いて読み書きの整理をする。この非同期モードは偏微分方程式における SOR(Successive Overrelaxation)法や ADI(Alternating Direction Implicit)法において採用すれば、多少のオーバーヘッドの減少が期待できるモードである。

この BWR の炉心計算では、ほとんど前節 1. の同期モードを用いている。例えば(10)式や(13)式に現われる $\sum_{4j}$ の計算では、まず SYNC サブルーチンを用いて全 PU の同期をとり、各 PU での ϕ の値を前後左右の PU との共通の CM メモリーに書き込む。ここまでは全 PU について全く同じプログラムであり全 PU で同期して進む。次に、このメモリーから ϕ を読みとりながら $\sum_{4j}$ の計算を行う。したがって、CM 領域に書き込んである間に隣りの PU のプログラムでその領域を読みとるということは起らない。

一般に、プログラムの作成方法には、大別して次の 2 通りがある。

- (1) プログラムを特定の炉心専用として実行時の能率を最適にする方法：これは発電所における炉心シミュレータとして考えられるもので、炉心境界や対称軸の通り方(ノード中心を通るかノード境界を通るか)、燃料の装荷方法などを固定してプログラムを各 PU ごとに作成し、無駄な炉心パラメータに関する判定を省くものである。
- (2) プログラムを汎用コードとする方法：プログラムは全 PU あるいはほとんどの PU で同一とし、炉心の幾何学的パラメータにより条件分岐して、あらゆる場合の計算を可能とするやり方で、設計計算用と考えることができる。

ここでは、両者の折衷型とした。すなわちコードの扱える最大限の炉心を 8×8 の燃料集合体より成る $\frac{1}{4}$ 炉心とし、1 つの PU が 2 ノードずつを分担することとして、Fig. 5 のように縦 8、横 4 の合計 32 台の PU アレイ上に分配した。この分割方法は固定的であるが、対称軸の通り方や炉心境界の形状、 z 軸方向のノード分点の数などは、可変になっている。

PU 上のプログラムはすべて同一とした。各 PU はデータとして全炉心の形状情報をもっており、これと各 PU 固有の座標との比較から自己の炉心内位置を知

Fig. 5 Correspondence between core fuel array with processing unit array

る。したがって、(10)、(13)、(14)式などでは境界点、境界外に応じて条件分岐することになる。条件分岐があっても並列計算が可能であるのは、一つ一つの PU がそれ自身独立に動作可能なマイクロコンピュータであるというMIMD方式のアーキテクチャのためである。ILLIAC-IV のような SIMD(Single Instruction-Multi Data Stream)方式⁶⁾のアーキテクチャを持つマシンでは、こういう場合逐次的処理になり並列度が上がらないであろう。

この同期型並列処理モードで可能な例としてノルム計算がある。例えば(20)式や(22)式に現われる体積積分、あるいはここに記していないが、各ループの収束を前回との相対誤差の最大値によって判定するときに現われる最大値などの計算である。これらの計算ではすべての PU におけるデータの加算または比較の演算が必要なのであるが、ILLIACのテキスト⁶⁾にあるようなルーティング手法を用いることにより、並列に実行することができる。ここでは(20)式の $\bar{s}$ の全炉心上の体積積分を例にあげる。最大値の計算では加算が比較の演算になるだけで、同じ手法が使える。 z 方向分点を 12 とすると 1 つの PU は $2 \times 12 = 24$ ノードを分担している。

- (1) まず、24 ノードに関する部分和を全 PU で並列に求める。
- (2) 次に、この部分 and を全 PU で一斉に右隣りの PU に送る。これをルーティングという(右端の PU は左端の PU へ送る)。
- (3) 全 PU で左より送られてきた部分 and と、自分自身の部分 and を加える。これで自分と左隣りの

PU での S の部分和がすべての PU に現われた。

- (4) 次に、この部分合を2つ右隣りの PU へ送る。
- (5) 全 PU で送られて来た部分合と自己の部分合を加える。これで全 PU にその行における S の部分合が現われた。
- (6) 以上と同様の手法を上下方向にも行う。ただし、部分合の送り方は、まず上隣り、2つ上隣り、4つ上隣りへ送る。
- (7) これにより全 PU で S の総和が現われる。ただし、計算機内部では有限の有効数字桁数であるため、加算の順序により答が異なる可能性がある。なので、どれかの PU の値を全 PU へ放送する必要がある。これは、一旦データを CU へ送り、次にこれを全 PU へ放送モードで書き込みばよい。

なお、ルーティングを行うときは直前で SYNC サブルーチンにより同期をとる。このルーティング手法により、一般に $N \times M$ の PU アレイ上のデータの加算は、 $\log_2 N + \log_2 M$ 回の加算で終了する。ここに、PACS のアーキテクチャではデータの転送には加算に比してほとんど時間を要しないので、加算回数だけの評価でよい。縦 $8 \times$ 横 4 の PU アレイの場合は、これは5回となる。CU により逐次的に加算するときには31回となるので、並列処理はノルム計算にも有効であることがわかる。

また、炉心内部に限定された体積積分をこのルーティングで求めるのは難しいので、炉心外の値をゼロとおいて炉心内外の区別なくルーティングに参加させなければならない。

V. 数値例と評価

例にとった炉心は Fig. 5 のように $\frac{1}{4}$ 炉心、 xy 平面 52 ノード、 z 方向 12 ノード、合計 624 ノードの BWR 炉心である。収束計算は性能評価の容易さなどの理由により誤差判定打ち切りとせず、ある一定回数の繰返し計算としている。採用した繰返し回数は、中性子束繰返し $I_n=4$ 、核分裂源繰返し $I_0=5$ 、ボイド繰返し $I_v=3$ となっている。

PACS による並列計算の場合と、HOST の汎用機での同様の計算を行なった場合を Table 1 に比較する。

並列処理部分のみの比較では、32 台の PU の並列処理は HOST の約 1.9 倍の計算速度に達していることがわかる。ただし、これにはコンパイラ言語による差やプログラム技巧による差が含まれている。したがって単純なハードウェア速度の比率とこの実行時間の比率とから並列計算のオーバーヘッドを算出することは困難であろう。

オーバーヘッドの見積りは、原始プログラムにおいて演算カウント(演算回数に演算の種類による重みを乗じて加え合わせたもの)を計算して推定できる。ハードウェア速度とコンパイラの作成する目的プログラムから推定して、浮動小数点演算と浮動固定小数間変換の重みを 1 とし、固定小数点演算、配列要素の添字計算、分岐判定、サブルーチンの呼出し、引数の引渡し、式中の変数の出現と左辺への代入の重みをいずれも 0.1 とし、平方根計算のそれを 8 とした。

オーバーヘッドの見積りをこのように行なっていくとき、同期モードの並列処理特有のオーバーヘッドあ

Table 1 Performance measurement and evaluation

Processed by	Language used	CPU time used (s)	Average execution time for total system (μ s)			
			Gibson-mixed average		Floating point average	
			Nominal	Effective	Nominal	Effective
PACS system:						
Parallel processing part in PUs	SPL/M	9.72†	0.422	0.600††	2.03	2.89††
Serial processing part in HOST and HOST-CU communication	FORTRAN	9.97	—	—	—	—
HOST system only:						
Serial processing of the parallelly processed part in PUs	FORTRAN	18.3	2.0†††		5.5†††	
Serial processing of the serially processed part in HOST	FORTRAN	6.62				

† CPU times for one-through calculation: 0.019 s for initial setting, 0.043 s for void distribution, 0.653 s for cross section and 0.352 s for neutron flux.

†† Effective speed with PU efficiency 0.703.

††† Estimated values from the other computers.

るいは無効な計算として考えられるのは、

- (1) PU の処理速度、処理手続の量の多少による PU のアイドルによるもの： 炉心内点、境界点、炉心外点によりプログラムが分岐し、早く処理を終えた PU は次に現われる同期サブルーチン SYNC 直前で、全体の PU の処理が終るまでアイドル状態に入る。処理の長短はプログラム中の分岐によって命令ステップが変ることにより生じるが、そのほか数値によって算術演算時間に差があることによっても生じる。

また、炉心外に対応している PU の計算は初期値ゼロより出発する全く無意味な計算を行なっている。これらは結果の算出に貢献していないので、PU の計算能力の無駄と考えられる。

- (2) 隣接 PU との通信のためのもの： これは隣接 PU ヘデータを渡す場合、まず PU のローカルメモリー (LM) から CM メモリーへ書き込み、次に隣接 PU は、この CM メモリーから自己の LM へ書き出す。この手続は並列処理に特有のものであり、オーバーヘッドと見なせる。

まず、(1) のアイドルによるオーバーヘッドを推定する。各 PU に対応する手続の演算カウントを計算すると、Fig. 6 のような分布となる。ここで炉心外のノードの演算は無意味であるので、ゼロとしている。Fig. 6 の数値は全プログラムの統計であるが、実際は各同期点において各 PU のアイドルを算定し、これを全プログラムで加え合わせねばならない。すなわち、全同期点 l についてのアイドルの和からオーバーヘッド

$$\beta_1 = \frac{\sum_k \sum_l (c_{\max, l} - c_{k, l})}{\sum_k c_{\max, l}} \quad (23)$$

0			
78006	73412	73412	72577
73412	68818	68818	69654
73412	68818	68818	69654
73412	68818	68818	34827
73412	68818	68818	34827
73412	68818	69654	0
73412	69654	34827	0
74665	34827	0	0

— Core Boundary

Fig. 6 Distribution of operation count

ここに、 $c_{k, l}$: k 番目の PU のプログラムにおける l と $l-1$ 同期点間の演算カウント

$c_{\max, l}$: PU に関する $c_{k, l}$ の最大値

を求めることになる。

同期点 l はプログラム上で 2 箇所あるが、そのうちの 1 箇所のルーティング・サブルーチン中では、直前の同期点からの演算カウントが同一であり、ほとんどアイドルが生じないと考えてよい。もう 1 箇所は拡散方程式や係数計算中の $\sum_k$ の和をとるサブルーチンの出口にあるが、ここで生じるアイドルは、このルーチン自身の種々の分岐の演算カウントの差から推定できる。その結果

$$\beta_1 = \frac{\sum_k (c_{\max} - c_k)}{\sum_k c_{\max}} = 0.259 \quad (24)$$

となる。アイドル・カウントのうち約 72% が、炉心外のノードを分担した 6 個の PU の計算が全部あるいは半分無効であることによるもので、残りの 28% は、炉心内の計算が中心のノードで最大となり、他の PU がその処理を待つことから生ずる。したがって、炉心内に注目すれば $\beta_1 = 0.070$ となる。

次に、(2) の通信によるオーバーヘッドは、全有効計算 (炉心内部のノードについて) の演算カウントに対する CM への書き込み・読出しのその比として推定できる。これは炉心を分担する PU では同一であり

$$\beta_2 = c_{\text{CM}} / c_{\text{total}} = 0.052 \quad (25)$$

となる。

全部の 32 個の PU の動作時間のうち、計算結果を導くのに有効なものの比率を PU 効率 $\alpha \equiv (1 - \beta_1)(1 - \beta_2)$ として定義すると、 $\alpha = 0.703$ となる。もし PU を炉心を分担しているものに限定すれば $\alpha = 0.882$ となる。

なお、これらのオーバーヘッドは、1 PU の分担するノード数や境界条件の扱い方などプログラム技法に依存して決っており、PU の総台数には無関係な量である。このため PU 台数を 32 台より増加したときも、この PU 効率がほぼ保たれると考えてよい。

1 台の PU のハードウェアの演算性能はギブソンミックス平均演算速度が $13.5 \mu\text{s}/\text{PU}$ 、浮動小数点平均速度が $65 \mu\text{s}/\text{PU}$ であるから、32 台の PU の並列処理速度は Table 1 のように、ギブソンミックス平均 $0.422 \mu\text{s}$ 、浮動小数点平均 $2.03 \mu\text{s}$ となるが、実際には、これにさらに α の効率を考慮した実効速度で評価せねばならない。

その結果、32 台の PU による並列処理の実効速度はギブソンミックス平均 $0.600 \mu\text{s}$ 、浮動小数点平均 $2.89 \mu\text{s}$ となる。ただし、この性能評価では HOST の逐次処理

には全く無駄なものがなく、並列処理には無駄があると仮定している。一方、HOST の計算機の演算速度(公表されていないが、推定値としてはTable 1 のようにギブソンミックス平均 $2 \mu s$ 、浮動小数点平均 $5.5 \mu s$)から考えると、PU と HOST との並列処理部分の所要時間比 $18.3/9.72=1.88$ は、ギブソンミックス平均速度の比 $2.0/0.6=3.3$ よりも浮動小数点演算速度の比 $5.5/2.89=1.90$ に近くなっていることがわかる。

このことにより、並列処理部分の実効速度(浮動小数点演算平均実行時間) S に関しては、1 台の PU の速度 S_{PU} 、PU の台数 N_{PU} 、PU 効率 α から $S=S_{PU}/(\alpha N_{PU})$ と評価できることがわかる。

Ⅶ. おわりに

偏微分方程式専用の並列マイクロコンピュータシステム PACS-32 を用いて沸騰水型原子炉炉心の 3 次元核熱水力計算を行なった。測定された性能は、中型汎用計算機の約 2 倍である。一方、汎用機 CPU の価格に比して PACS のハードウェア価格は約 1/10 であるので、性能価格比では約 20 倍の向上が得られていると思われる。

並列処理部分のみの比較では PU は効率 70% で動作しており、オーバーヘッドは比較的少ない。特に、この PU 効率が PU 台数に依存しないことが重要で、台数を増すことにより、台数に比例した速度向上が得られることが期待できる。

この試みは、市販の、それも少し古い 8 ビット系マイクロプロセッサを用いて構成したものであるが、最近の 16 ビット系のプロセッサ(例えば、インテル 8086, 8087)により同じようなアーキテクチャを試みれば、性能価格比はさらに約 2~5 倍程度向上するものと推定される。

炉心モデルも、本報では比較的簡単なものであるが、これより複雑なモデルを用いても同様の並列技法によりシミュレーションは可能である。PU 上の並列部分と HOST 上の逐次処理部分の時間比は、現在 1:1 である。炉心モデルを電力系統の負荷変動に追従可能なものとして Xe 動特性を含むものを現在試みている。Xe 動特性や流量制御は並列処理可能であり、全処理時間中の並列処理部分の比率はより多くなるであろう。

得られた良好な性能価格比などの性能は、ミニコンピュータよりも優っており、より実績を積むことで、設計計算用やオンサイトシミュレータとして、実用に供することも可能と思われる。しかし、用いたハード

ウェアは、並列計算技法のテストを目的として試作したものであり、BWR 炉心計算に対して理想的なものではない。また、この PACS は拡散方程式のみならず MHD モデル、プラズマ粒子シミュレーション、モンテカルロシミュレーションなどをも対象とし、将来の超高速計算機へのプロトタイプマシンとして設計しているものである。将来のあるべきマシンは 10,000 台の PU より成る 10 GFLOPS (10^9 Floating point Operations per Second) のものであるが、このスーパーコンピュータ向けの利用技術の検討と蓄積、将来マシンの設計データを得るうえで、この PACS-32 による BWR 炉心シミュレーションの経験は貴重である。

終りに、この研究を始めるに当り、動機づけと貴重な助言を頂いた川合敏雄氏(慶応大学)、また終始変わぬご鞭撻を頂いた若林二郎教授(京都大学)に深く感謝します。また BWR のモデルは、筆者がノルウェー滞在中 S. Børresen 氏、K.O. Solberg 氏より教示をうけたものであり、ここに感謝します。また、PACS システムは京都大学工学部学生、大学院生の東野純一、山岡彰、澤田和男、伊藤八大、佐藤 隆の各氏の協力により作成されたものであり、ここに深く感謝します。

—参考文献—

- (1) 星野 力, 東野純一: 原子力学会昭54年会要旨集, D26, D27, (1979).
- (2) 山岡 彰, 他: 昭 55 電子通信学会総合全国大会, 1400, 1401, (1980).
- (3) HONECK, H.C.: On the feasibility of using parallel microprocessors for the calculation of 3-dimensional rating distributions in operating reactors, NEACRP Specialist's Meeting on Calculation of 3-Dimensional Rating Distributions in Operating Reactors, OECD, Paris, (1979).
- (4) 星野 力: 原子力誌, 22[4], 204~212 (1980).
- (5) プラズマ数値シミュレータに関するワークショップ報告, 核融合研究, 44, 別冊その 3, (1980), 名大プラ研.
- (6) FLYNN, M.J.: IEEE TC, C-21(9), 948(1972).
- (7) 中西正和, 大野義夫: “やさしいコンパイラの作り方”, (昭55), 共立出版.
- (8) BØRRESEN, S.: Nucl. Sci. Eng., 44, 37 (1971).
- (9) 加藤満佐夫, 苗村憲司: “並列計算機”, (昭51), オーム社.
- (10) 二村良彦, 川合敏雄: bit, 12, 580 (1980).

〔付 録〕

PAD はフローチャートよりも簡潔で生産性の高い問題構造表現図である¹⁰⁾。左上より計算が始まり、順次下方へ進み、左下で終る。右辺が 2 重線になっているブロックは繰返し条件を表わし、それが満たされているうちは、その右方にある枝を同じく上方から計算する。2 重の横線は、その左右が等価であることを示す。無条件ジャンプは許されない。