計算機アーキテクチャ 119－33
（1996．8．29）

並列／分散ソフトウェア開発支援環境の構築

野口善昭† 金丸弘樹†
東由美† 岩根雅彦†

細粒度並列処理マルチプロセッサMSBMと汎用マルチプロセッサMDBM/FMMを制作して，細粒度静的コードスケジューリングアルゴリズムや動的スケジューラの開発などの細粒度並列処理の研究にスタンドアローンで使用してきた．研究環境を向上させるためだけでなく，並列処理と分散処理の効率のよい融合方法を研究するために，MSBMとMDBM/FMMをLANに接続して，Linux上でのクアイアント／サーバモデルを採用した．マルチプロセッサで並列実行されるアプリケーションプログラムに対してコントラクタという概念を導入して抽象化することによってハードウェア構成と分離した．そして複数のコントラクタを一つのマルチプロセッサで並列実行可能とした．コントラクタは動作環境ヘッダ，コーディネータ，マネージャおよびワーカで構成される．コーディネータはサーバとアプリケーションプログラムとのインタフェース，マネージャはアプリケーションプログラムのなかの親プロセスであり，ワーカは子プロセスとしての計算主体である．マネージャおよびワーカは要素プロセッサをまたがって並列実行される．動作環境ヘッダはコントラクタの効率的な要素プロセッサへの割当てに使用する．特に実行ノードリストによりコントラクタの他かのノードコンピュータへの遠隔手続き呼び出しの自動転送を可能にした．簡単な例で基礎実験を行った．実験結果からこの方式の有効性が確認された．

The Parallel and Distributed Applications Development Environment

YOSHIAKI NOGUCHI,† HIROKI KANAMARU,† YUMI HIGASHI† and MASAHIKO IWANE †

The parallel computer MSBM for fine grain tasks and the general purpose parallel computer MDBM/FMM has been developed. We study fine grain size parallel processing such as development of fine grain size code scheduling algorithms, dynamic scheduling systems and so on. MSBM and MDBM/FMM has been used as a stand-alone parallel computer. We connect MSBM and MDBM/FMM to LAN, and we adopt client/server model on Linux operating system not only to improve an environment for research but also to study how to fuse parallel and distributed processing. For this purpose, we has been introduced the contractor that separates hardware configurations from an application program. It consists of an operation environment header, a coordinator, a manager, and workers. A coordinator is an interface between server and an application program. A manager is parent process and a worker is child process in an application program. The manager and the worker compute using some processors. An operation environment header is used for efficient assigning contractor to processor. Specially, server sends a remote procedure call to other node-computer automatically. We made a simple test. It shows that this method is effective.

1. はじめに

複数のプロセッサを密結合接続した並列計算機が開発されてきた．一方ではローカルエリアネットワークLANによる分散処理が一般化している[1]．密結合並列計算機をスタンドアローンで使用するかまたはLANのノードコンピュータとして使用してほかのノードコンピュータから遠隔手続き呼び出しによってアプリケーションプログラムを実行している．また並列計算機をクラスタとして取り扱い高速ネットワークによってそれらを接続して超並列計算機を実現している[2]．我々は細粒度マルチプロセッサMSBM[3][4]および汎用マルチプロセッサMDBM/FMM[5]を開発してきたが，これらはMSDOSのもとでスタンドアローンで細粒度静的コードスケジューリングアルゴリズムの開発[6]，並列コンパイラの開発，細粒度動的スケジューラの開発[7]に使用してきた．MSBMとMDBM/FMMを前者の立場でLANに接続して，ソフトウェア開発支援環境を向上させると

† 九州工業大学工学部電気工学科
Department of Electrical Engineering, Faculty of Engineering,Kyushu Institute of Technology

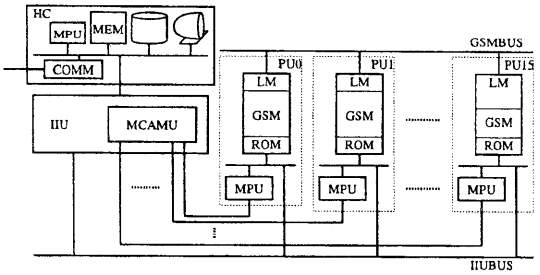

図1 MSBM ハードウェア構成

ともに，複数の並列計算機による細粒度並列処理と分散処理の効率的な融合処理の実現可能性を探る．本報告ではMSBM と MDBM/FMM のハードウェア構成を簡単に述べ，両マルチプロセッサに共通化した Linux 上で構築されたスタンドアローン用ソフトウェア構成を述べる．そしてネットワーク構成と融合処理の実現方法を述べ，簡単な実験を行い，その結果について検討した．

2. 並列計算機のハードウェア構成

2.1 MSBM

MSBM は細粒度並列処理の多重プログラミング環境における静的バリア同期管理 SBM とグループ共有メモリ GSM の評価のために開発したマルチプロセッサである．一つのアプリケーションプログラム AP の実行に割り当てられる要素プロセッサ PU をプロセッサグループ(単にグループ)，その AP の中でバリア同期をとる PU の組をバリアグループとよぶ．MSBM ではプログラムの実行開始前にバリア同期機構 MCAMU[8] に設定しプログラム終了後にその設定を解除する SBM をおこなった．

MSBM は図**1** に示すようにホストコンピュータ HC，統合インタフェース IIU および 16 台の PU から構成される．HC は 80386MPU，8MB メモリ，各種入出力装置からなるパーソナルコンピュータである．IIU は，DMA による HC メモリと PU メモリ間のマルチキャストデータ転送機構，PU から HC への割り込みの中継機構，SBM に適した MCAMU で構成される．PU は 8088+8087MPU，256kB メモリ，グループ内 PU メモリ間で共有メモリを実現する GSM インタフェース，PU から HC へ種々の処理を割り込みによって依頼するサービスコール SVC や HC と PU 間のデータ転送に使用する IIU インタフェース，PU と MCAMU 間のバリア信号で構成される．PU レジスタのうちグループ番号を保持する GPNO，局所メモリと共有メモリの境界を示す GSBASE，HC から PU への小規模通信のための RCTLPU は HC と PU の両方から設定できる．

2.2 MDBM/FMM

MDBM/FMM は細粒度並列処理の多重プログラミング環境における動的バリア同期管理 DBM と細粒度，中粒度，粗粒度並列処理における PU 間通信の検証のために開発したマルチプロセッサである．DBM はバリアグループの MCAMU への設定を AP の実行中にも可能としたものである．このために PU から MCAMU への設定時間は極力小さくしなければならない．汎用マルチプロセッサでは細粒度，中粒度，粗粒度並列処理を効率よくおこなう必要があるので，共有メモリ構成から局所メモリ構成まで PU メモリを段階的に選択できる再構成可能メモリ RSM が望ましい．そこで PU メモリを局所メモリ LM，グループ共有メモリ GSM，システムで共有するシステム共有メモリ SSM を任意の大きさに設定できるようにした．LM のみ必要で大量のプロセッサ間通信をおこなう AP に対しては PU メモリ間で DMA 転送をおこなう ILLIAC 網を用意した．図**2** に MDBM/FMM の構成を示す．MDBM/FMM は 80486MPU をもった HC，32-64 台の PU(8088+8087) と新統合インタフェース NIIU で構成される．MSBM と異なる箇所は高速バリア同期管理機構，RSM，NEWS 転送である．NEWS 転送はグループ内またはシステム内のすべての PU のメモリ間でおこなわれる．隣接 PU へ大量のデータ転送をおこなう MIMD 型の粗粒度並列化 AP では線形結合で十分であるので ILLIAC 網は PU 割り当ての容易さを確保するためである．

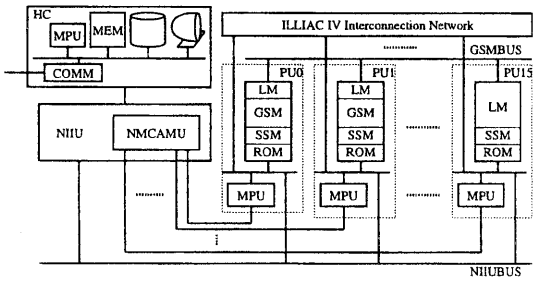

図2 MDBM/FMM ハードウェア構成

3. 並列計算機のソフトウェア構成

3.1 概　　要

MSBM，MDBM/FMM のソフトウェアはオペレーティングシステム Linux，PU などの資源管理をおこなうエクゼクティブとアプリケーションプログラムであるコントラクタから構成され，コントラクタはコーディネータとマネージャおよびワーカから構成される．エクゼクティブは Linux 上でつねに動作しており，キーボード入力からのコントラクタ実行の指示またはコントラクタの実行終了を待っている．PU 待ちキューの中から，図**3** に示したコントラクタファイルにある動作環境ヘッダの必要 PU 台数と PU 割り当て形態 (離散的 PU 割り当てまたは列 / 行方向に連続した PU 割り当て) をもとに，最小一致アルゴリズムによって，割り当て可能なコ

ントラクタにPUを割り当てる．同時にコーディネータをHCにローディングしてLinuxの一つのプロセスとして実行する．

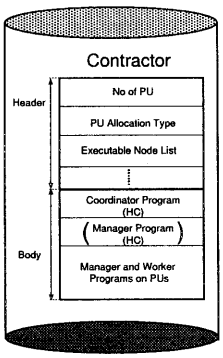

図3　コントラクタファイル

コーディネータはワーカをPUまたはHCにローディングして実行を指示するがこの手続きはコントラクタによって異なる．コーディネータがEXITを実行すればコントラクタは終了する．図4に示すように一つの並列計算機上で複数のコントラクタの実行が可能である．手間はかかるけれど各ユーザが都合のよいようにコントラクタを構成できる．これは細粒度並列処理用オペレーティングシステムなどの開発のために自由度を持たせたことによる．

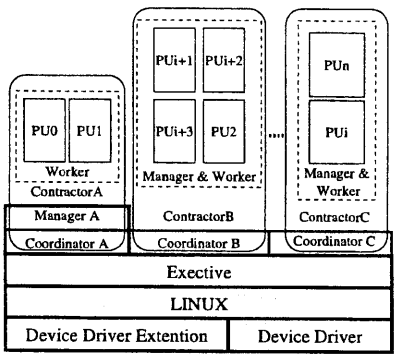

図4　ソフトウェア概要

3.2　デバイスドライバ

デバイスドライバはHCのLinux上に組み込まれ，IIUまたはNIIU(以後IIU)を介してHCとPU間のプログラムやデータの転送，PU間同志の大量のデータ転送，HCからのバリア同期管理，HCからPUの制御，PUからHCへのSVC受付をおこなう．デバイスファイルによりPU指定，グループ指定，ブロードキャスト，MCAMU指定，SVCのPU指定をおこなう．これによりHCとPU間の情報転送には通常のREAD，WRITE，LSEEKにより実行できる．またIIU内のDMACレジスタなどの設定やPU内のRCTLPUレジスタ，GSMBASEレジスタなどの設定にはIOCTLを用いる．IOCTLのIIU用特殊関数を**表1**に示す．またNEWS転送のためにIIU_NEWS(from_address,to_address,size,direction)，SVC受付のためにIIU_SVC(PU_no,count,data,current_time)のシステムコールを用意した．

表1　IIU用IOCTLコマンド

Command	Operation
RESET	Send reset signal to IIU and All PUs
SET_GSBASE	Set GSM Base register
SET_GPNO	Set Group No register
SET_RCTLPU	Se Control PU register
NEWS_Transfer	Set DMAC and Start NEWS Transfer
SET_SVCGROUP	Set SVC Group No

3.3　モニタ

PUモニタはPUメモリの8KBのROM上におかれている．アイドル時にはRCTLPUを検査してその値により異なる適切な処理をおこなう．HCのIOCTLによってあるいはPU自身によってRCTLPUが書き換えられたとき，実行中の一連の処理の終了後にPUメイルボックスをみてその処理をおこなう．**表2**に処理内容，**図5**にメイルボックスのメモリマップを示す．

表2　処理内容

RCTLPU	Operation
FF	No Operation
FE	Start PU Program
FC	RESERVED
⋮	⋮
00	RESERVED

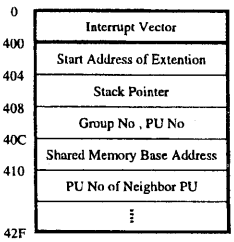

図5　PUメイルボックス

PU上のマネージャおよびワーカの実行開始はHCからのRCTLPUの設定により，メイルボックスに示された実行開始から実行させる．HCへのSVC要求や

MCAMの更新などはマネージャまたはワーカが直接出力命令を発行することによってモニタ処理は最小限にしている．動的スケジューリングを含んだオペレーティングシステムをPUに搭載したときは，マネージャおよびワーカの実行開始はこの方法とは当然異ったものとなる．

4. ネットワーク構成

4.1 分割利用と協調動作

図6に示すようにMSBMとMDBM/FMMは一つのノードコンピュータとしてWSやPCと同等に研究室内LANで接続されている．またルータであるWSより学外にも接続されている． WSまたはPC上のクライアントプロセスからMSBMまたはMDBM/FMMにあるコントラクタを実行するために，MSBMあるいはMDBM/FMMサーバプロセスにエクゼクティブを組み込んだことにより，サーバプロセスとクライアントプロセスが同一ノードコンピュータにあっても遠隔手続き呼び出しと局所手続き呼び出しは同じ取り扱いができる．サーバプロセスはクライアントプロセスから送付されたメッセージすなわちコントラクタファイル名を受け取り，ヘッダにある動作環境を調べて十分な空きPUが存在してすぐに実行可能であればこのノードコンピュータで実行する．実行が待たされるときは，コントラクタが実行できるノードコンピュータを示すコントラクタ実行ノードリストを調べて次のノードコンピュータにクライアントプロセスからのメッセージを転送する．このメッセージが実行ノードリストを一巡して最初のサーバプロセスに到着したときこのコントラクタをPU待ちキューに登録する．このようにユーザがノードコンピュータを直接指定しなくても空きノードコンピュータでコントラクタを実行できるようにした．またMSBM，MDBM/FMMでの複数のコントラクタの協調動作は通常のLinux上の協調プロセスとして実現した．図7に並列/分散ソフトウェア構成を示す．

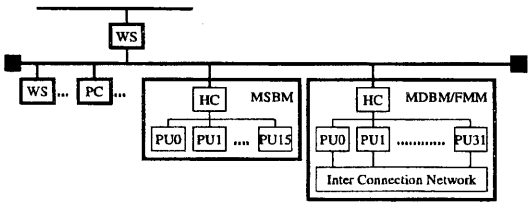

図6 ネットワーク構成

4.2 クライアントとサーバ

クライアントプロセスからMSBM(MDBM/FMM)サーバプロセスにコントラクタの実行を依頼するとき，well_knownなメッセージプールWKNMPとtemporaryなメッセージプールTMPMPを使用してプロセス通信をおこなう．コントラクタ名とコントラクタがクライアントとデータの送受をおこなうTMPMP名をWKNMPに書き込んでサーバプロセスに送信する．サーバプロセスはコントラクタのコーディネイタプロセスを生成してTMPMP名を渡す．コーディネイタプロセスはTMPMPからデータを受け取るとともに，動作環境ヘッダをみてマネージャやワーカをHCやPU上に生成してコントラクタを実行する．コーディネイタプロセスは実行結果をTMPMPに書き込みクライアントプロセスに送信する．クライアントプロセスでは実行結果を受け取って処理を続行する．図8にこの過程を示す．

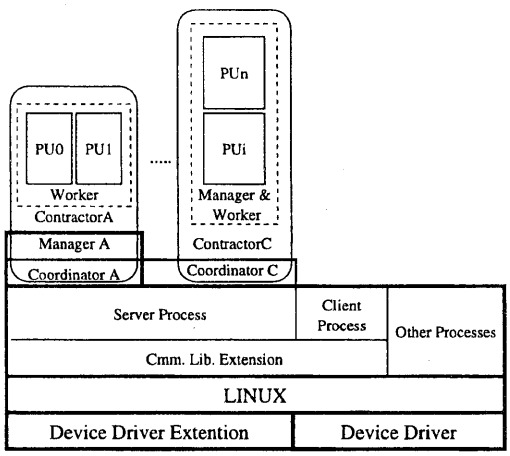

図7 並列/分散ソフトウェア構成

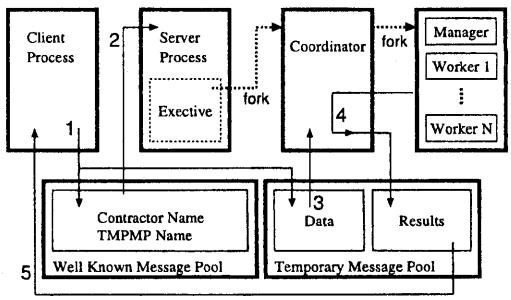

図8 遠隔手続き呼出し

これらのメッセージの送受信のために，送信用として非ブロック型手続きであるsend(pool_name,queue_name,message,size)，受信用としてブロック型手続きであるrecv(pool_name,queue_name,message,size)をライブラリとして用意する．なおメッセージプールには複数のキュー持つことができる．このようにTMPMPを導入したのはMSBMとMDBM/FMM上で同一のコントラクタの実行可能を確保することとオーバヘッドの削減を図るためである．

5. 実　　験

5.1 実験方法

コントラクタの自動転送による複数ノードコンピュータの分割利用と協調動作の実験を行う．クライアントプロセスからノードコンピュータへ送付された複数の並列実行可能なコントラクタを同時に処理し，PU 待ち状態のコントラクタをほかのノードコンピュータへ送付することにより分割利用と協調動作の確認とその実行時間を測定する．

ノードコンピュータとして MSBM と MDBM/FMM を用い，MSBM では 8 台，MDBM/FMM では 16 台の PU を使用する．クライアントプロセスを実行するマシンは Sun SPARCstation 10(SS10) を使用する．

各コントラクタは並列実行可能な 8 個のワーカを持ち 8PU を使用する．PU 割り当て形態は離散的 PU 割り当てを用いる．コーディネータは PU へのワーカプログラム及びデータのローディング，MCAM へのバリアグループの登録，SVC グループの設定，グループの設定，実行開始，終了確認を行う．ワーカは 6553600 回ループし 1 回のループごとにバリア同期をとるプログラムを実行する．ワーカは SCV を用いて処理の終了をコーディネータへ連絡する．ワーカプロセスの動作はマネージャを必要としないためマネージャは省略した．実行ノードリストについては，MDBM/FMM では MSBM，MSBM では MDBM/FMM を持つ．

クライアントプロセスは 10 個のコントラクタの実行要求メッセージを MDBM/FMM サーバに送り，すべてのコントラクタの実行終了を待つ．コントラクタの終了はクライアントプロセスが用意したメッセージプール TMPMP のキューにコーディネータからの終了メッセージが到着したことにより確認する．

PU 上で動作するワーカプログラムはアセンブリ言語で記述し，HC 上で動作する通信ライブラリ，サーバ，コーディネータ及び SS10 上で動作するクライアントプログラムは C++ 言語で記述した．ネットワークプロトコルは TCP/IP を用い，BSD socket インタフェースを用いて実装した．

動作の確認はサーバの動作記録を基に行った．実行時間の測定はサーバの動作記録のタイムスタンプとワーカの実行時間の 2 つを測定した．ワーカの実行時間はコーディネータがワーカに実行開始要求を送ってから，実行終了を通知する SVC を受けとるまでの時間で測定した．

5.2 実験結果

サーバの動作記録をもとに得られた動作結果を図 9 に示す．$C_0 \sim C_9$ はクライアントプロセスから駆動されたコントラクタを表す．また C_n の添字はクライアントプロセスから要求を発行した順番を示す．G_0，G_1，G_2 はグループを表わしそれぞれ 8PU で構成される．矢印はサーバによって自動的に送信されたコントラクタを表す．空欄の部分はアイドル状態であることを示し，斜線部分はコーディネータがワーカプログラムの実行環境の準備を行っているオーバヘッドを示す．

10 個のコントラクタのうち 8 個のコントラクタが MDBM/FMM を 2 つのグループに自動分割し同時に実行され，2 個のコントラクタが MSBM に自動送付され処理が行われた．10 個のコントラクタを自動分割，自動送付することなしに逐次実行する場合に比べ約 2.5 倍の速度向上が得られた．また，コーディネータによって得られた C_0，C_3，C_6，C_9 の実行時間と，サーバの実行記録のタイムスタンプによって得られた T_{G_0} から，図 9 の斜線で示されるオーバヘッドを計算したところ，T_{G_0} の 4.4% がオーバヘッドであることが得られた．

C_5 は 5 番目に駆動されたコントラクタであるがこの結果では最後に実行が行われている．この原因について検討する．C_0，C_1 はそれぞれ G_0，G_1 で実行が開始される．残りの $C_2 \sim C_9$ は実行が待たされるので実行ノードリストより MSBM のサーバに送られる．C_2 は MSBM で実行されるが $C_3 \sim C_9$ は再び MDBM/FMM のサーバに送付される．この時点で $C_3 \sim C_9$ は MDBM/FMM サーバの PU 待ちキューに登録される．C_0，C_1 はほぼ同時刻に実行を終了し MDBM/FMM サーバは新たに C_3，C_4 の実行を開始する．C_5 は資源不足のため実行が待たされるので，MSBM サーバに送られる．MSBM は C_2 を実行中であるため，C_5 は再び MDBM/FMM サーバへ戻る．このとき PU 待ちキューの最後に登録されるため C_5 の実行は最後に行われる．

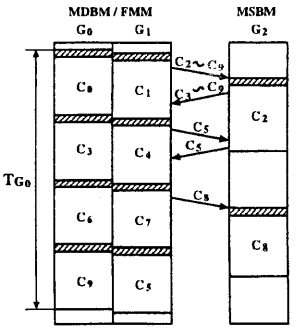

図 9　コントラクタの動作履歴

6. むすび

複数の細粒度並列処理可能なマルチプロセッサを LAN のノードコンピュータとして接続して細粒度並列処理と分散処理の効率的な融合を実現するソフトウェアシステムを構築した．マルチプロセッサ上で並列処理される AP に対してコントラクタなる概念を導入して，コン

トラクタの構成要素を動作環境を示すヘッダとコーディネータ，マネージャ，ワーカと抽象化することによってハードウェア構成と分離することが可能となった．コントラクタと実行ノードリストによってマルチプロセッサの分割利用と有効利用が実現できた．

細粒度並列処理ノードコンピュータや細粒度並列処理マルチプロセッサオンチップのソフトウェア構築の基盤を与えたが，今後はこのシステムの各種プログラムによる評価およびこのシステム上で細粒度並列処理可能なマルチプロセッサのオペレーティングシステムの開発，細粒度並列コンパイラを開発する予定である．

参 考 文 献

1) 藤田，篠田，今泉 (訳): 平行分散システム，トッパン (1996)
2) 富田眞治: 超並列処理計算機 JUMP-1 のアーキテクチャ，情報処理，Vol.36，No.6，pp528-537(1995)
3) 岩根，本石，野口，米沢: 細粒度マルチプロセッサ MSBM，情報処理論文誌，Vol.37，No.6，pp1196-1205(1996)
4) 岩根，野口，茶屋道，本石，重松: 細粒度並列計算機 MSBM の開発，九州工大研究報告 (工学)，Vol67，pp.61-68(1995)
5) 岩根，茶屋道，本石，立川，浦崎: マルチマイクロプロセッサ MDBM/FMM の開発，九州工大研究報告 (工学)，Vol.68，pp.49-56(1996)
6) 岩根，濱田，小島，松田: 式の分割による並列化アルゴリズム ESH，SWOPP'96(秋田)(1996)
7) 松田，武石，野口，岩根: 細粒度並列処理のためのハイブリッドスケジューリングシステム: スケジューラによるバリア同期管理手法，情報処理研資 OS-70，pp.113-120，SWOPP'95(別府)，(1995)
8) 岩根，重松，定家，茶屋道，金沢:FPGA によるバリア同期用機能メモリの開発，九州工大研究報告 (工学)，Vol.66，pp.45-52(1994)