ICOT Technical Report: TR-269

TR-269

並列推論マシンPIM

後藤厚宏

June, 1987

ICOT

Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan

(03) 456-3191～5
Telex ICOT J32964

Institute for New Generation Computer Technology

並列推論マシン PIM

後藤厚宏 (新世代コンピュータ技術開発機構)

１．はじめに

人工知能向き計算機の開発は，より汎用な，または，より強力な計算機を作ることと考えてよい．なぜなら，人工知能向き計算機とは，単にゲーム木の探索を行うだけの計算機ではなく，これまでの計算機が持っている多くの機能も兼ね備えていなければならないからである．人工知能向き計算機においても，資源管理やユーザインタフェスを備えるオペレーティングシステム（ＯＳ）がさらに重要な意味を持つことは，ＩＣＯＴが逐次型推論マシンＰＳＩとＯＳ（ＳＩＭＰＯＳ）の開発を通して示した通りである．

ＩＣＯＴでは，論理型の枠組みの基に，

①ユーザ側からマシン側までを一貫性をもってサポートする論理型言語システム（ＫＬ１），

②ＫＬ１で記述した並列推論システムとしての機能を持つオペレーティングシステム（ＰＩＭＯＳ），

③並列推論マシン（ＰＩＭ）アーキテクチャ，

の３つを統一的に開発することを目指している．[1]

このような一貫性は言語とアーキテクチャの整合性を高め，性能の向上，システムとしての見通しの良さを実現する重要な条件である．例えば，論理に基づくメタ推論，学習，知識の獲得／管理といった高度な人工知能ソフトウェアまでもが，この一貫したシステムに取り込むことが可能になる．この結果，プログラミング技術やコンパイル技術，ＯＳにおける資源管理手法，機械語からアーキテクチャに至るシステム階層の整合性が確保され，最後はシステム性能の向上に貢献する．

並列マシンの構成方式を考えた場合，通信のコストの大きい部分と小さい部分が必ずある．このようなアーキテクチャ上の局所性が存在する以上，ソフトウェアのレベルからアーキテクチャの局所性を利用すること，および，ソフトウェアから利用し易いハードウェア構成を導入することが必要である．このためには，まず，

・アルゴリズムの設計の段階で並列性と局所性を考える．

・プログラミングにおいてそれを明確に表現できる，

ことが大切である．ただし，人工知能処理においては，処理の大きさ（負荷）が常に変化するため，

・動的に変化する負荷を問題の局所性によって制御できるメカニズム

が必須となる．

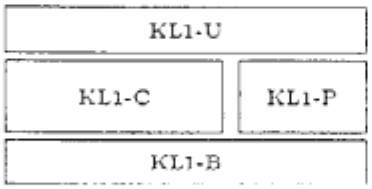


図１　KL1の構成

本稿では，ＩＣＯＴにおける並列推論マシンアーキテクチャの概要について述べ，並列処理における局所性の観点から，並列推論マシンにおける課題と特性について検討する．

２．並列推論マシンＰＩＭの全体像

2.1 核言語ＫＬ１

ＩＣＯＴにおいて，並列プログラミングの能力と，核言語としての簡明さ，汎用性ある記述能力の双方を持つ言語として生み出された言語がＧＨＣである[2]．通常，ＧＨＣはＡＮＤ並列型の論理型言語と呼ばれ，並列プロセスがゴールによって，プロセス間の同期やストリーム通信がＡＮＤ関係にあるゴールの共有変数によって表現される．核言語ＫＬ１は，ＧＨＣをベースにユーザ側，マシン側の双方に拡張した階層を持つ言語系として構成する．（図１）

ＫＬ１-cは，ＧＨＣの機能を一部制限した Flat ＧＨＣにメタ機能を追加した言語であり，まさにＫＬ１の核としての役割を持つ．

ＫＬ１-pは，局所性や負荷分散をプログラミングにおいて表現のために導入する言語機能である．プログラマは，ＫＬ１-pの機能を用いて，自分のプログラムがＰＩＭ上でどのように負荷分散した方が良いかについてのヒントをＰＩＭ上の処理系とそのＯＳに与える．

ＫＬ１-uは，ＫＬ１のユーザ側のインタフェスであり，オペレーティングシステム（ＰＩＭＯＳ）の記述言語としての役割と持つ．ＫＬ１-uでの言語上の意味はすべてＫＬ１-cで規定される．

ＫＬ１-bは，ＫＬ１のマシン側のインタフェスであり，後述するＰＩＭの機械語として位置付けられる．

2.2 ＰＩＭのハードウェア構成

ＰＩＭの全体構造を図２に示す．要素プロセッサはＫＬ１向き命令セットを持つプロセッサであり，単体でも十分に高速なもの(200~500KLIPS)を目指している．

１０台程度のプロセッサは，共有バス／共有メモリによって密に結合したクラスタを構成する．クラスタ内のプロセッサは一つのアドレス空間を共有し，メモリへのRead/Write操作によって同期をとるレスポンスが高速な通信が可能となる．さらに，各プロセッサにキャッシュを設け，プロセッサ毎の局所的な処理効率の向上とクラスタ内でのプロセッサ間通信の高速化を図る．一方，このような共有メモリ上の変数を介した通信では，メモリアクセスの排他制御機構が必要である．ＰＩＭにおいては，各プロセッサのキャッシュのブロック状態を利用したロック機構を設け，低コストの排他制御を可能にする．

これらのクラスタは，ネットワークによって疎に結合される．各クラスタは独立のアドレス空間を持ち，メッセージを用いてアドレス変換を伴う非同期プロセッサ間通信を行う．

これによって，クラスタ毎のメモリ管理（ＣＣ）が可能になると共に，クラスタ台数に関する拡張性がよくなる．ここで，ネットワークのトポロジ以上に重要なのが，メッセージ通信におけるパケットの組み立て／読み取りを支援するハードウェアとクラスタ間に渡るポインタの管理である．特に，クラスタ間に渡るポインタの管理は，アドレス変換に相当する操作であると同時に，システム全体のガーベッジコレクションの実現方法と強く関連する．このようなクラスタ間通信のために，クラスタ毎にクラスタコントローラ（ＣＣ）があり，通信に伴うアドレス変換，メッセージ送受をサポートする．

３．ＰＩＭの命令アーキテクチャと並列処理方式

3.1 ＫＬ１の抽象マシン命令

ＫＬ１におけるユニフィケーションの大部分は，コンパイラによって，引数の読み出し，データ型の判定，メモリへの値の書込みといった基本操作に分解できる．そこで，Ｐｒｏｌｏｇにおける WAM(Warren Abstract Machine) と同様に，ＫＬ１プログラムのＫＬ１プログラムにおいても，抽象マシン／抽象命令セットを以下のように定義した(3)．ここでは，適当数のレジスタを持つプロセッサとゴールやその変数を置くメモリを想定する．

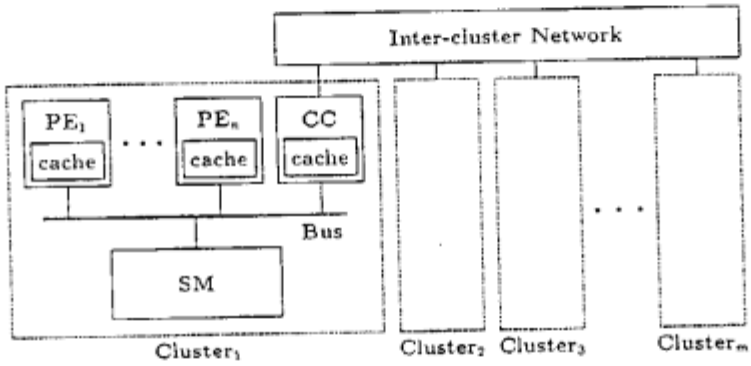
(1) ゴールの実行制御

ＫＬ１の抽象マシンでは，図３のようにＫＬ１ゴールをゴールレコード(GR)で表現し，メモリのゴールレコード領域に置く．またゴール間で共有される変数セルは，スタックではなくヒープ領域上に置く．ゴールレコードの要素は，ゴールの状態，引数リスト（変数領域へのポインタ），命令コードへのポインタである．また，ゴールの成功／失敗をメタレベルで扱えるように，すべてのゴールレコードはメタゴールレコードをノードとしたＡＮＤ木によって管理される．

ゴールレコード領域の中の実行可能なゴールは，互いに繋がれたゴールキューを構成する．プロセッサは，ゴールキューからゴールレコードを一つ取り出し，その引数リストをレジスタ上にロードするとともに命令コードの実行に移り，そのゴールのリダクション操作を進める．

(2) 受動部（ヘッド／ガード）の実行と中断処理

システム内に存在する多数のゴールは，概念上並列に処理可能である．ただし，各ゴールをいつでも実行できるわけではない．それをリダクションするために必要な入力引数が受動部の単一化として指定されているためである．



PE : Processor Element
SM : Shared Memory
CC : Cluster Controller

図2　PIMの全体構成

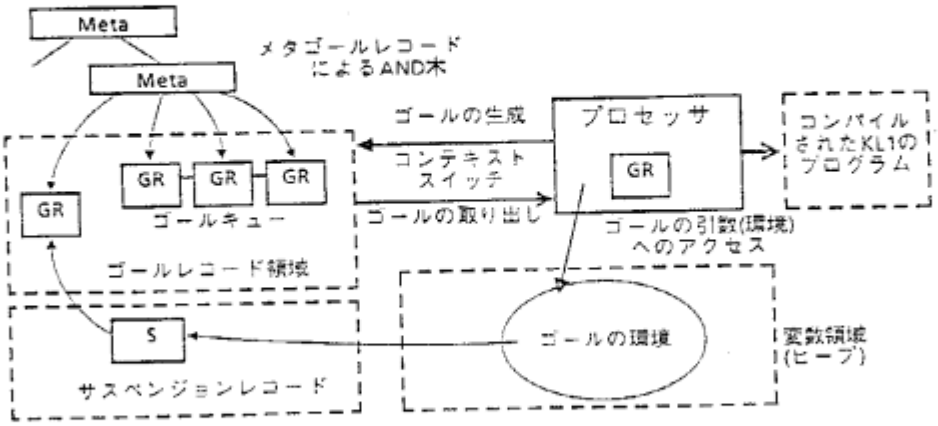


図3　KL1の抽象マシン

受動部の単一化は，レジスタ上の引数からヒープ領域を手繰り，具体化が必要な引数をテストする操作である．このような受動部の単一化は，ex.1）のようにＯＲ関係にあるクローズが複数ある場合，各クローズについて順次行い，最初にコミットしたものについて，その能動部の実行に移る．

```
ex.1)   ?- p(X,Y).

p(X,Y) :- X = a | ·····.
p(X,Y) :- X = b | ·····.
```

受動部の実行において変数の具体化を待ち合わせする必要がある場合はリダクションを中断し，ゴールレコードをメモリに戻すとともに，サスペンジョンレコード(S) によってその原因となった変数にゴールレコードを繋ぐ（サスペンド）．

(3) 能動部（ボディ）の実行

能動部の実行では，変数を具体化する単一化と新ゴールの生成がある．能動部の単一化では，変数が具体化されるのを待っている中断中のゴールがある場合，そのゴールをゴールキューに戻し，実行可能にする．

ex.2) のように新しくゴールを生成する場合，r(Y,Z)に相当するゴールレコードをゴールキューの先頭に作り，q(X,Y) については，ゴールキューに戻さずレジスタ上に作ってしまうことによって，新たにゴールレコードを取り出す手間を省く．（これにより，ゴールの取り出し戦略は，基本的に深さ優先となり，ゴールキューはゴールスタックと呼ぶ方が良いかもしれない）

```
ex.2)   p(X,Z) :- X = a | q(X,Y), r(Y,Z).
```

3.2 クラスタ内における並列処理

共有メモリ構成でクラスタを構成する目的は，クラスタ内のプロセッサが共有しあうメモリ領域にゴール間の共有変数を置き，プロセッサ間通信をメモリへのRead/Write操作とすることにより，プロセッサ間通信のコストを小さく抑えることにある．このため，上記の抽象命令セットを共有メモリアクセスでの排他制御（ロック操作）を伴うものに拡張する．

(1) ゴールキューと変数領域の分散

ＫＬ１プログラムにおけるゴールリダクションにおいて，単に必要なデータを全て共有しあうような方法では，共有データへのアクセスがネックになってしまい，共有メモリ構成の利点を活かした処理効率は得られない．そこで，ＫＬ１プログラムの実行における並列ゴールを各プロセッサ毎のゴールキューによって分散管理するとともに，各プロセッサのリダクションにおいて使用する変数領域の割り当てをプロセッサ毎に行う[4]．これにより，ゴールの起動制御，新ゴールの生成時における排他操作を不要にする．（図４）さらに，再開ゴールの優先的な実行等，プロセッサのデータアクセスの局所性が高まるような並列ゴールのスケジューリングを行うことにより，後述するキャッシュの効果を高める．

(2) キャッシュとロック機構

並列キャッシュ機構[5] は，プロセッサ毎に設けたキャッシュ間の一貫性をバスコマンドによって自動的に保持するものであり，すでに幾つかのキャッシュ制御方式が提案されている．

ＫＬ１の並列処理では，従来のマシンに比べて，書込み率が高く，ヒープ上の共有変数を介したプロセッサ間通信を行うため，並列キャッシュ機構として次のような特徴を持つ制御方式が適すると考えられる．

①複数キャッシュに共有されているブロックへの書込み時は，他のキャッシュブロックの内容を無効化することよって一貫性を保持する．

②Write-back型によって共有メモリへの書込みを抑える．

③単調にメモリを消費するメモリ書込みでは，キャッシュブロックを共有メモリからフェッチしない書込みを行う (Direct Write)．

ヒープ領域はゴール間で論理的に共有されるため，単一化における書込みには排他制御（メモリのロック操作）が必要である．一方，キャッシュでは各ブロックについて他のキャッシュにコピーがあるかどうかが検出できる．ＰＩＭのクラスタでは，このようなキャッシュブロックの状態を利用して，ロック制御を各プロセッサで分散制御すると供に，ロック操作のコストを十分に小さくする．

3.3 クラスタ間並列処理方式

ＰＩＭのクラスタは，それぞれ独立したアドレス空間を持ち，ネットワークで結合されている．このため，クラスタ間に渡る並列処理では，ＰＩＭに先行して開発されたマルチＰＳＩシステムと同様にメッセージによる並列ゴールリダクションを行う[6]．

クラスタ間に渡る負荷の分散は，クラスタ内と同様にゴール単位で行う．ただし，クラスタ毎にアドレス空間が異なるため，ゴールを受け取ったクラスタから元のクラスタにある共有変数を参照できる機構が必要である．このために，クラスタ部に輸出表と呼ぶアドレス変換表を設ける．他のクラスタへゴールを送出するときは，送出されるゴールの引数から指されている共有変数を登録し，ゴールを受け取ったクラスタには輸出表へのエントリ名を持つ変数セルを置く．（このような変数を外部参照変数と呼ぶ．）また，ゴールをクラスタ間で送受することによりＡＮＤ木によるゴール管理がクラスタ間に渡ることになる．このようなゴール管理に関するメッセージ通信のコストを抑えるために，ＡＮＤ木をクラスタ毎に分散管理する方式を導入する[7]．

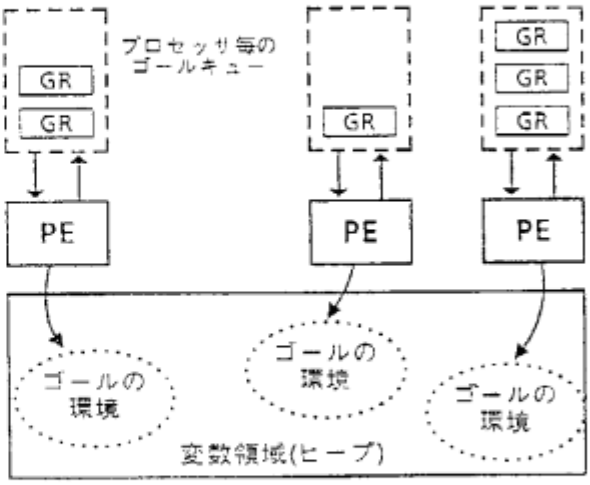


図4 クラスタにおける並列処理

一方，外部参照変数を含む単一化においては，クラスタ間のメッセージ通信を起動する．受動部の単一化の場合，プロセッサは外部参照変数をアクセスしようとしたゴールの実行を未定義変数へのアクセスと時と同様に中断し，実際の変数があるクラスタへ変数の値を問い合わせるメッセージを送る．問い合わせのメッセージを受け取ったクラスタでは，輸出表を通して変数がアクセスされる．すでに具体化している場合は，その値を問い合わせを行ったクラスタに返す．能動部の単一化の場合は，その単一化を一つのゴールと見做し，変数を持つクラスタにメッセージによって送出する．

4．検討

4.1 並列処理の粒度

ＫＬ１の抽象命令セットではＫＬ１のゴールリダクションを一つの処理単位としている．ＫＬ１のリダクションをさらに小さく分解して，並列性を高めることも可能である．しかし，実際のプログラムでは，非常に細かい粒の処理単位をそのまま並列処理する必要がない場合が多い．また，細かい粒に問題を分割すると，並列処理の効果に対する通信コストの比率が高くなりがちである．つまり，並列マシン上で考えるべき並列度は，上記の意味で"通信コストが小さく抑えられるような粒の数"として考えるべきである．

ＫＬ１のプログラムにおいては，メッセージを送受することによって内部状態を刻々と変化させるようなプロセスをゴールリダクションによって表現することが多いと考えられる．ＫＬ１の抽象命令で示したようなレジスタベースのゴールリダクションでは，プロセッサがメモリからゴールレコードを取り出したり戻したりするコンテキストスイッチのコストが大きい．しかし，一度レジスタに取り出された後のリダクション操作は高速に実行できる．つまり，ＫＬ１の抽象命令では，上述のようなプロセスの実行のように一度メモリから取り出されたゴールがレジスタ上に置かれたままリダクションを連続して行うことを想定していると言ってよい．言い換えれば，レジスタ上で連続して実行できるリダクション数が多いほど処理の効率が良い．

4.2 ゴールのスケジューリングと分散

一般に，深さ優先の実行によって連続したリダクションは，新ゴールを生成しない能動部を実行した場合と変数の待ち合わせが必要な場合に途切れる．後者は，ゴールの実行順序とゴール間の依存関係が合わないことによって生じるものである．このため，ゴールの実行順序（スケジューリング）の戦略は処理効率に大きく影響する．さらに，並列処理においてはプロセッサ間にゴールを分散して実行するため，ゴール間の依存性を十分考慮したゴールの分散が重要である．

4.3 並列処理における通信のコスト

並列処理における通信のコストは，プロセッサ間の通信を同期的に行なうか，非同期的に行なうにかよって性質が異なる．クラスタ間の通信網のように通信の応答時間が大きい場合，通信の応答を待ち合わせない非同期的な通信が行われる．この場合，並列マシンの内部に，並列に動作できるプロセスが十分多数存在することが前提となる．プロセッサは，通信の応答を必要とするプロセスを実行する度にプロセスを次々に切り換える．このため，通信コストはプロセッサにおけるプロセスの切り換え（コンテキストスイッチ）のコストが主になる．

一方，クラスタの内部のように通信網の通信速度が小さく抑えられるような並列マシンでは通信の応答を待ち合わせるような同期的通信が行われる．この場合，通信の待ち時間が主たる通信のコストである．

4.4 メモリ管理機構

ゴールの変数を置くヒープ領域はワード毎に２～３個のアクセスで"使い捨て"的に使用されるため，アクセス総数の割にメモリ使用量が大きい．これにより，頻繁なＧＣが必要となるだけでなく，メモリ参照の局所性も悪くなる．このため，ヒープ領域を動的に再利用する機構が必要である．

ヒープ領域を再利用するためには，参照カウント等による動的メモリ管理が必要である．しかし，参照カウントの管理は効率がよくない．そこで，ゴールからヒープにある論理変数セルへの参照数を１ビットのポインタタグ（ＭＲＢ）で管理する方法[8]による動的メモリ管理の導入を検討中である．ＭＲＢによるメモリ管理では，ヒープ領域をフリーリストで管理できるため，ヒープ領域の再利用回数が増し，キャッシュ効率も良くなると期待できる．ただし，単一化においてタグを管理するオーバヘッドがあるため，その効果を詳細に評価する必要がある．

5．おわりに

ＩＣＯＴで開発を進めている並列推論マシンＰＩＭのアーキテクチャと並列推論システムの研究開発において明らかになってきた人工知能における並列処理の課題と，について述べた．

現在，設計を詳細化するとともに，ソフトウェアシミュレーションとマルチマイクロプロセッサ上の並列処理系によつ評価を進めている．

<参考文献>

[1] Goto,A. et al., "Toward a High Performance Parallel Inference Machine — The Intermediate Stage Plan of PIM —", TR-201, ICOT, 1986.

[2] Ueda,K., "Guarded Horn Clauses", TR-103, ICOT, 1985.

[3] 木村 他，"並列推論マシンＰＩＭ — ＫＬ１の抽象命令とコンパイラ —"，第34回 情処全大，２Ｐ－１．

[4] Sato, M. et al. "KL1 Execution Model for PIM Cluster with Shared Memory", ICLP'87, 1987.

[5] 松本 他，"並列推論マシンＰＩＭ — 並列キャッシュとロック機構 —"，第34回 情処全大，２Ｐ－６．

[6] Ichiyoshi,N. et al. "A Flat GHC Implementation on the Multi-PSI", ICLP'87, 1987.

[7] 六沢 他，"並列推論マシンＰＩＭ — クラスタ間に渡るゴール管理方式 —"，第34回 情処全大，２Ｐ－５．

[8] T.Chikayama et al. "Multiple Reference Management in Flat GHC", ICLP'87, 1987.