ICOT Technical Report: TR-211

TR-211

共有メモリ構成クラスタ向きKL1処理方式

佐藤正俊、清水　肇、松本　明
六沢一昭、後藤厚宏

November, 1986

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

共有メモリ構成クラスタ向きKL1処理方式

KL1 execution Model for PIM Cluster with Shared Memory

佐藤 正俊, 清水 肇, 松本 明, 六沢 一昭, 後藤 厚宏
(財)新世代コンピュータ技術開発機構

ICOTで開発中の並列推論マシンPIMのクラスタ向きKL1処理方式について述べる。本処理方式はクラスタを共有メモリ構成で実現する場合の課題をKL1実行の局所性を用いて解決するものである。

１．はじめに

ＩＣＯＴでは，ＧＨＣ［上田］をベースとした並列論理型言語（ＫＬ１）を実行する並列推論マシン：ＰＩＭを開発中である。ＰＩＭの処理方式を決定する際，通信コストは重要な要因である。つまり，プロセッサ台数に対するプロセッサ間の通信コストがソフトウェアからみて滑らかに変化することは，ソフトウェアにとって負荷分散等の制御が容易であり，全体の処理効率を高められる。

このため，ＰＩＭの構成はクラスタにより階層化し，クラスタを構成する密結合マルチ・プロセッサとクラスタ間の疎結合マルチ・プロセッサより成る構成をとり，ＰＩＭの処理方式は，共有メモリを利用したアクセス同期通信を主体とした密結合マルチ・プロセッサ向き処理方式とメッセージ通信を主体とした疎結合マルチ・プロセッサ向き処理方式を融合したものとする［後藤］。後者は，Multi-ＰＳＩシステムとの共通課題として検討を進めている［宮崎］。

ここでは前者の共有メモリ構成クラスタ向き処理方式について述べる。

共有メモリ構成での処理方式は，ＫＬ１の並列実行におけるプロセス（ゴール）間の通信を共有メモリ上の共有資源を用いて低い通信コストで実現することである。

しかし，この実現には以下の課題がある。

① 共有資源の操作における正当性の保証（ロック）
② 共有資源の操作におけるレスポンスの保証
③ 物理的に共有される資源をアクセスするときのアクセスパス競合の軽減

我々は，これら課題を解決するために共有メモリ構成向き処理方式として，共有メモリ構成を構成する個々のプロセッサが局所的にＫＬ１を実行できるような処理方式を検討している。

以下では，ＫＬ１の実行のイメージを２章で述べ，次にこれを並列に実行する場合の課題及びその解決案について３章で触れ，４章ではこれら課題を解決するための局所的な処理方式について述べる。最後にこの処理方式の局所化効果について簡単な評価結果を示す。

２．ＫＬ１の実行イメージ

（１）ＫＬ１

ＫＬ１はFlatＧＨＣにメタ機能や負荷分散の制御用の指示（プラグマ）等を追加した並列推論言語である。FlatＧＨＣとは，処理効率の良いインプリメントを可能とするために，ＧＨＣにいくつかの制限等を加えたＧＨＣのサブセット言語である［ＩＣＯＴ］。その制限には，ガードに書けるゴールの制限やガードの逐次実行等があり，ゴールを処理単位とする実現が容易になっている。

ＫＬ１は，その処理単位をゴールとし，ゴールの実行は定義節（同一述語を持つ節の集合）を一つのコンパイル単位として表現した機械語（ＫＬ１－Ｂ）により行われる。以下，この機械語をコードと呼ぶ。

一方プラグマとは，プログラムの並列実行時にプログラムの特性を処理系に反映させるための処理系へのヒントであり，後述する負荷分散で述べる。

（２）データ

実行に必要なデータは，定義節を表わすコードとゴールの実行環境を表わすレディキュー，ゴールレコード，共有変数，サスペンドレコード，メタコールレコードである。これらのレコードは以下の構成要素を持つ。

ゴールレコード：状態，リンク，引数表，コードへのポインタ，メタコールレコードへのポインタ

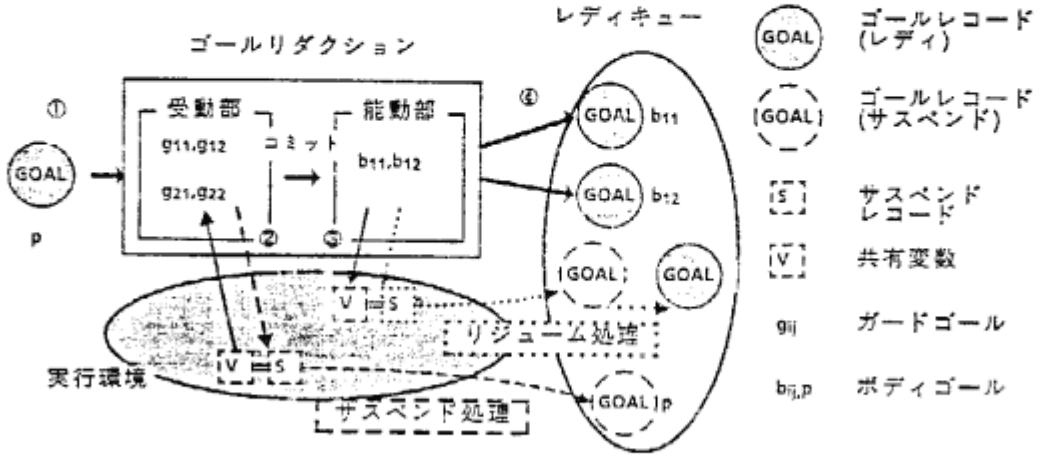


図１ ゴールリダクション

サスペンドレコード：リンク，ゴールレコードへのポインタ，ＯＲ待ちフラグへのポインタ
メタゴールレコード：状態，兄弟リンク，親リンク，子ゴール数，実行結果変数

（３）ゴールリダクション

ＫＬ１の実行はゴールリダクションであり，その実行は図１のようになる。ここで，ＫＬ１のプログラムは次の形をしたガード付Horn節の集合である。

```
H :- G1, ...Gm | B1, ...Bn. (m)=0,n)=0)
頭部      ガード      ボディ
  受動部              能動部
```

① レディキューより実行可能なゴールを選ぶ（スケジューリング）。
② 能動部を実行する節を１つに決定するために候補節の受動部をテストする。このとき読み出した値が具体化されていない場合は，中断原因の共有変数をスタックし次の候補節を試みる。
全ての候補節の受動部が成功できずかつ中断する候補節が存在する場合，ゴールの実行を中断し，中断原因の共有変数が具体化されるまで待ち合せる。この処理をサスペンド処理と呼び，待ち合せのためにその共有変数にサスペンドレコードをリンクする。
③ ②により選択（コミット）された節の能動部を実行する。このとき待ち合せゴールの存在する共有変数に対するユニフィケーションが起こると，サスペンドレコードによりその待ち合せゴールをレディキューに入れる。この処理をリジューム処理と呼ぶ。
④ 能動部の実行でさらにリダクション可能（ユーザ定義述語）である場合は，それらを新たなゴールとしてレディキューに入れる。ゴール木の管理はメタゴールレコードの子ゴール数を操作することによって実現する。この管理はフェイルの伝播を部分木にまとめるために必要である。ゴール木の構成を図２に示す。

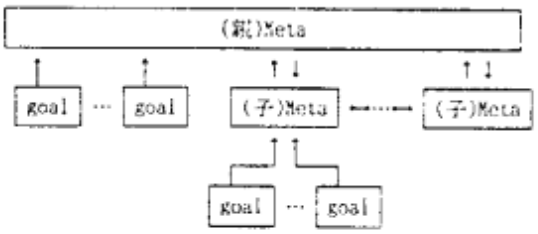


図２ ゴール木の構成

３．ＫＬ１の並列実行

（１）共有メモリを用いた並列実行における課題

共有メモリ構成の特徴は，プロセッサ間通信のコストを共有メモリを用いることで小さく抑えられる点である。ここでＫＬ１の並列実行は，２章で示したリダクションを多数のプロセッサ上で行なうことで実現する。このリダクションにおいて，単にリダクションに必要なデータ全てを共有メモリ上に置いて実現すると，データアクセスに関する排他制御，レスポンス，アクセスパス・ネック等の課題が生じ，共有メモリ構成の利点を活かした処理効率を得ることが難しくなる。

これらの課題を解決するためには，共有メモリ構成においても各プロセッサの局所性を活かした処理方式が必要である。ここで局所性とは，①処理方式上扱うデータ自身の持つ静的な局所性，②プロセッサ上のリダクション時に現れる動的な局所性，③プログラムの持つ特性を実行時に利用する静的な局所性を動的に利用する局所性がある。以下では①を局所的なメモリ配置，②を局所的スケジューリング，③をプログラム特性に応じた負荷分散の問題として整理する。

（２）局所的なメモリ配置

局所的なメモリ配置とは，ＫＬ１の実行におけるデータのアクセス特性に従い，データの論理的な領域を分離し，メモリ配置することである。領域は，スケジューリングにより生成と読み書きがプロセッサ個別に実現できるもの（他のプロセッサからの直接のアクセスはない）と生成はプロセッサ個別にするが読み書きは共有するものに分離できる。前者をローカル・メモリ，後者をグローバル・メモリと呼び，それぞれに置くデータは以下のようになる。

ローカル・メモリ：レディキュー，ゴールレコード

レディキューは，リダクション時のゴールの取り出しやゴール生成時にアクセスされ，スケジューリングによりプロセッサ個別に実現できる。

ゴールレコードは，そのアクセスが全メモリアクセスの１／３程度でありアクセス頻度が非常に高い。一方，そのアクセスはリダクションを行なうプロセッサが決った時点で一意に決るためローカル・メモリ上に置くことができる。

グローバル・メモリ：共有変数，構造データ，サスペンドレコード，メタゴールレコード

ＫＬ１の実行では，共有変数やサスペンドレコードによりゴール間の通信を実現し，ゴールにまたがるアクセス（ゴールがプロセッサにまたがるときはプロセッサ間通信となる）が主体となる。このためこれらをグローバル・メモリ上に置く。

メタゴールレコードは，部分木で共有されるデータでありグローバル・メモリ上に置く。しかし，これへのアクセスは部分木を構成するゴールが変るたびに起り，アクセスが集中する。このためメタゴールレコードの子ゴール情報に対してはデータを分離し，プロセッサ単位に分けて実現する。構造データは，共有する利点が大きいためグローバル・メモリ上に置く。

（３）局所的スケジューリング

局所的スケジューリングとは，リダクションゴール間の親子関係を利用しゴールを連続的に実行することで，プロセッサ内の親ゴールの実行環境をレジスタにより引き継ぐことである。これによりゴール間の通信コストは大幅に低減できる。これは実行時の時間的な局所性を処理系の記憶階層を用いて有効利用することである。

（４）プログラム特性に応じた負荷分散

プログラム特性に応じた負荷分散とは，実行時に生成される比較的大きな部分ゴール木を，プロセッサ毎の負荷のバランスが保てて，しかもプロセッサ間の移動が少なくなるように分散させることである。ここで分散の対象となる比較的大きな部分ゴール木は，プラグマを用いてプログラマが指定したものである。ゴール木の分散は要求駆動によ

り実現する．つまり，要求駆動により無駄なゴールの分配を抑え，かつそのためゴールをプラグマ付（プログラマ指定）のゴールに制限することで，実際に分配されるゴールを大きな部分木に限定できる．またこのため要求の回数自体も低く抑えられる．

このような負荷分散はメモリの空間的な局所性を有効利用している．つまり，プロセッサに対してゴールを局所的に割当てることは，リダクション実行時に使用するメモリ空間とプロセッサ個別のメモリ空間をほぼ等しくしている．

４．　共有メモリ構成クラスタ向きＫＬ１処理方式

上記の局所性を活かした処理方式に基づき並列ゴールリダクションを実現した場合，以下のようになる．

図３に，この実行の概念イメージを示す．

（１）ゴールのスケジューリング

スケジューリングはプロセッサ毎に独立に行なうことを基本とし，要求駆動により負荷を分散する．その実現は以下のようにする．

レディキューは優先順位の異なる２つのレディキューを用意し，プラグマ付ゴールは優先順位の低いキューにスケジューリングする．プラグマ無しのゴールは高い優先順位キューにスケジューリングする．またレディキューはスタック的に使用し，フォークゴールの内１つのゴールを連続的に実行する（つまり，レディキューを介さず実行する）．

プロセッサのレディキューが空になった場合は，他プロセッサにゴールの要求をする．この要求の方法は２段階あり始めは（この様子を図４に示す）他のプロセッサでゴールフォーク（P-Enqueue 命令の実行）時のプラグマ付ゴールを要求する．この要求は共有メモリを利用した状態の書き換えで行ない（図中のRequest Flag），ゴールの分配はレディキューとゴールレコードをローカル・メモリ上に置くためプロセッサ間のメッセージ通信により実現する（図中のMessage Handler ）．またこのメッセージ通信は共有メモリ上の通信バッファにより実現する．

一定時間でゴールが得られなかった時は次の要求法として優先順位の低いキューに既にあるゴールを要求する．この要求とゴールの分配はプロセッサ間のメッセージ通信で実現され，相手プロセッサのリダクション処理を中断し，ゴールレコードをコピーするため通信コストは前者に比べて高い．

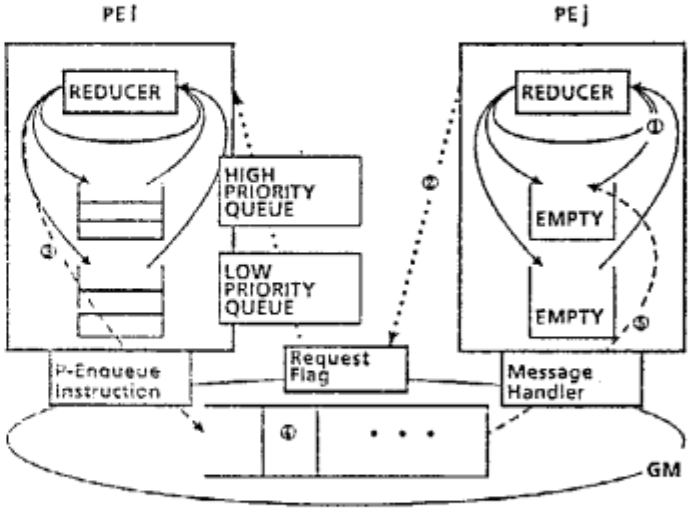


図4 負荷分散(第一段階)

（２）待ち合せチェックと受動部のテスト

受動部のテストは値の読み出し操作であり，共有メモリへのアクセスであってもロック操作は必要ない．ただし，サスペンド処理の中断原因変数にサスペンドレコードをチェインする操作はロックが必要となる．

（３）能動部の共有変数への具体化

能動部のユニフィケーションは，リジューム処理を伴う場合と伴わない場合で異なる．伴わない場合は，Prolog のユニフィケーションと同じ双方向ユニフィケーションでお

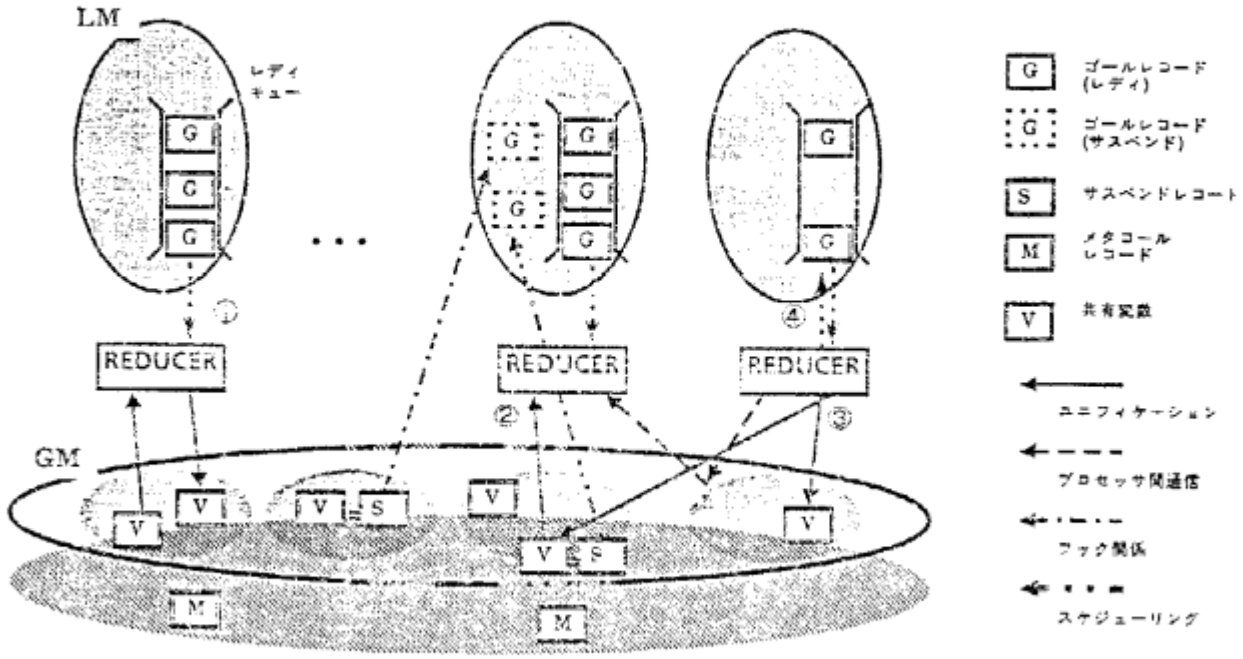


図３実行の概念イメージ

り，共有変数が共有メモリ上に置かれるためユニフィケーションは，その共有変数が具体化されていない場合，その変数に対してロックし，値を書き込む操作が必要となる。また，新たな共有変数や構造データセルの生成はプロセッサ個別のメモリ領域に行なう。

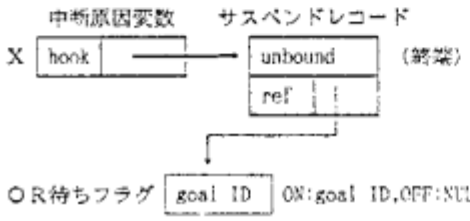
リジューム処理は，待ち合せゴールが他の変数とＯＲ待ちしている場合，その待ち合せの同期操作と待ち合せゴールをそのゴールがサスペンドされたプロセッサにスケジューリングする操作に分かれる。

同期操作はＯＲ待ちサスペンドレコードで共有される共有メモリ上のＯＲ待ちフラグをチェックし，ＯＮであればＯＲ待ちフラグをＯＦＦにし，スケジューリング操作を実行し，ＯＦＦであれば何もしない。

スケジューリング操作はサスペンドしたプロセッサと再開させるプロセッサが異なる場合，ゴールレコードをローカル・メモリ上に置いているためプロセッサ間のメッセージ通信で実現される。このためサスペンドレコードはプロセッサの番号とゴールレコードのＩＤを持つ。

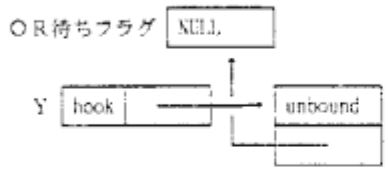
サスペンド処理とリジューム処理におけるロック操作をＯＲ待ち（Ｘ，Ｙで待つ）の例で示す。

1) まず，Ｘをロックしサスペンドレコードを登録，ロックを解除。（この時，ＯＲ待ちフラグにゴールＩＤをセットする）



2) 次に，Ｙのためのサスペンド処理を行なう前に，Ｘが具体化されたとすると，その具体化は成功しＯＲ待ちフラグをNULLに変え，ゴールをレディキューに繋ぐ（リジューム処理）。

3) 一方，サスペンド処理ではＹをロックし，中断処理を行なう（ロックの解除）。この後，Ｙが具体化されれば，そのリジューム処理でＯＲ待ちフラグがNULLであるのでサスペンドレコードは回収される。



（４）ゴールの生成及びゴール木の管理

ゴールの生成は，ローカル・メモリ上のゴールレコードへの書込みで実現する。またゴールの生成に伴う共有変数の生成は共有メモリへのアクセスであってもロック操作は必要ない。これはプロセッサ個別のメモリ領域に行っているためである。

ゴール木の管理のためのメタゴールレコードは共有メモリ上に置くが，子ゴールのカウンタに書き換えが頻繁に起こるためプロセッサ別に分離する。このためゴールの分配時はそのカウンタのプロセッサ単位の更新が必要であり，分配側がロック操作で行なう。

５．　シミュレーションによる評価

現在，４章で示した処理方式に基づいてソフトウェア・シミュレータを作成中である。ここではその途中結果としてグローバル・メモリでの共有変数及び構造データに関して，その負荷のバランスの状況及び局所化効果を評価する。

実行したプログラムは８queensとボトムアップパーサ（ＢＵＰ）であり，８queensはその実行において探索木が均質に広がり処理系にとっては扱いやすい理想的な問題である。またＢＵＰの探索木は比較的アンバランスに広がり処理系にとって扱いにくい問題である。

シミュレーション結果（表１）はidle率，割当て比，アクセス比（受動部／能動部）について，要素プロセッサ４台でそれぞれのプログラムを分配戦略（ランダム分配と局所分配）を変えて実行した場合の平均を示している。ここで，idle率とは，実行時間に対してのidle時間の割合であり，シミュレーションは１リダクションを単位時間と仮定している。また，比は自プロセッサと他３プロセッサの合計の比を表わし，割当て比とは自／他のゴール分配比（自を１としたときの値），アクセス比とは自／他のＧＭ上の共有変数及び構造データへのアクセス比を表わしている。

表１より局所的分配ではidle率を抑え，かつ割当て比とアクセス比を低く実現していることがわかる。これにより局所分配は局所化効果が充分にあるといえる。

表１　戦略による局所化効果

	８ｑｕｅｅｎｓ		ＢＵＰ	
	ランダム	局所分配	ランダム	局所分配
ｉｄｌｅ率	３．２％	０．４％	１３．４％	６．６％
割当て比	０．５６６	０．００１	０．０５３	０．０３１
アクセス比 受動部	１．５５１	０．０２３	０．８２７	０．３２２
アクセス比 能動部	０．６１２	０．０１０	０．２３８	０．１３８

６．　おわりに

中間ＰＩＭのクラスタにあたる密結合マルチ・プロセッサ構成でのＫＬ１実行方式について述べた。今後は，現在検討が進んでいる並列キャッシュ［松本］において，この処理方式の効果を並列キャッシュ・ソフトウェア・シミュレーションにより確認して行く予定である。

最後に，日頃御指導をいただくＩＣＯＴ４研内田室長をはじめ，御討論いただくＰＩＭグループの方々に深謝する。

<参考文献>

［上田］上田，"Guarded Horn Clauses"，ICOT TR-103.
［後藤］後藤，"並列推論マシンPIM-中間構想-"，情処全大第33回 3B-5.
［松本］松本，"並列推論マシンPIM-中間PIMのハードウェア構成について-"，情処全大第33回 3B-6.
［宮崎］宮崎，"Multi-PSIにおけるGHCの実行方式"，LPC'86.
［ＩＣＯＴ］電子計算機基礎技術開発成果報告書，基礎ソフトウェアシステム編 40-119 ICOT.