知識ベース指向並列処理システム*

1D-6

横田 治夫、 北上 始、 服部 彰
富士通株式会社

1．はじめに

並列推論モジュールと並列知識検索モジュールからなる知識ベース指向の並列処理システム（KOPPS: Knowledge-base Oriented Parallel Processing System）の構成と，マルチマイクロ計算機（Sequent 社の Symmetry）上に実現した試作システムによる実験の結果について述べる．

KOPPS は，大きな知識ベースを利用するような知識処理を，並列に効率よく実行できる環境を提供することを目的としている．我々は，知識検索のモデルとして RBU[1] を採用し，並列論理型言語 GHC[2] から RBU に並列にアクセスして推論を行う方法を取った．RBU は知識を項（変数を含んだ構造体）で表現し，その集合を項関係（項を要素とするテーブル）として管理し，単一化（Unification）機能を使って検索するモデルである．試作システムでは，RBU の機構として大量の知識に対し効率よく検索するための専用インデックス[3]と，検索／更新コマンド単位で並列に処理するための同時実行メカニズムを採用している．また，GHC とRBU との間は，GHC の論理変数を使ったストリーム通信によって，検索コマンドや検索結果を転送し合う．

2．試作システムの概要

図1に，試作システムの構成を示す．Symmetry の複数のプロセッサ上に，GHC のプロセスと RBU のプロセスを置いて，プロセス間で通信をしながら処理を進める．利用するプロセッサの台数は，実験のため変化させる．この時，あるプロセッサは，並列推論モジュール(以下，単に推論モジュールと略)用か並列知識検索モジュール（同様に，検索モジュールと略）用に振り分けられ，1つのプロセッサ上で2種類のプロセス両方を実行することがないように指定する．GHC は述語単位で，RBU は検索／更新コマンド単位でプロセスとして実行される．実際に起動をかけられるプロセスは，負荷分散に従って，それぞれのモジュール内で動的に決定される．

知識ベース（項関係の集まり）は，共有メモリ上に置かれ，複数の RBU のプロセスから同時にアクセスされる．この並列アクセスに対する排他制御は項関係単位で行われ，検索処理か更新処理かによって共有モードと排他モードを使い分ける．この排他制御のための情報は，項関係の1つであるディクショナリに格納され，排他機構はセマフォによって実現している．

推論モジュールと検索モジュールの間の接続も，共有メモリ上の共有領域を用いて実現している．共有メモリ上に，知識ベースの領域とは別に，コマンドページと呼ぶ領域を用意して，

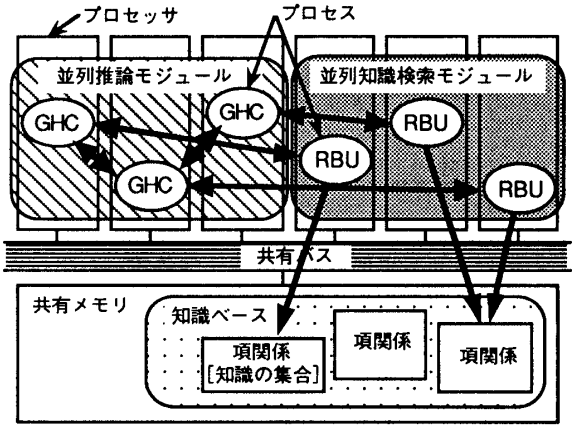

図1 試作システムの構成

双方のモジュールからアクセスする．つまり，推論モジュール側からの検索や更新の指示は，このコマンドページを介して検索モジュールに渡され，その結果もまたコマンドページを介して返される．

コマンドや結果が上書きされたり，複数プロセスに重複して読み込まれたりしないように，アクセスする時にコマンドページにロックをかける．このためコマンドページが1つだと，コマンドページへのアクセスが並列処理のボトルネックになる．そこで，コマンドページを複数設けて，アクセスを分散させている．試作システムでは，検索モジュールに割り当てられたプロセッサの台数と同じ個数のコマンドページを用意した．ただし，負荷が片寄らないように，両モジュールの各プロセスはどのコマンドページへもアクセスできる．

この複数コマンドページを用いた動的コマンド割り振りにより，一般にはコマンドの実行順序が保証されないことになる．推論モジュール側でコマンド間の順序関係を明示的に制御したい場合には，検索／更新コマンド終了時に検索モジュール側から返されるステータス情報をプロセス起動に利用する．

2つのモジュール間のデータ転送のインタフェースは，リスト状につながれた GHC の論理変数を用い，要求駆動によるストリーム通信となっている．つまり，推論モジュールは，バインドされていない変数をリストセルに入れて，転送要求として検索モジュールに渡す．検索モジュールは，この要求に従って検索結果を1つ1つ変数にバインドさせる．この転送は，検索処理と並列に，全ての検索結果がそろうのを待つことなく，結果が得られた時点で行われる．

3．実験とその結果

上述した試作システムの性能を評価するため，実際に使われるものに近い，計算機に関する知識ベース（556タプル）を用意して，推移閉包を含めた並列検索の実験を行った．

実験のために用いた GHC のプログラムを図2に示す．この

Knowledge-base Oriented Parallel Processing System *
Haruo Yokota, Hajime Kitakami, Akira Hattori FUJITSU LIMITED
* 本研究は第五世代コンピュータプロジェクトの一環として行われたものである．

プログラム中の search_loop という述語の第3引数によって，処理ループ中で生成される RBU コマンドをインタフェース用の組込述語 rbu_stream にストリームとして渡している．ループ中で呼ばれる GHC の述語と RBU コマンドは，すべて並列に動作可能である．なお，プログラムの中で，S=[_,_|_] あるいは L=[_,_,|_] と記述してある部分は，ダブルバッファリングを使った要求駆動処理のために空のリスト2つ分を用意しているところである．

処理速度の目安として，現在最も広く使われている Prolog の処理系の1つである Quintus Prolog のインタープリタと比較を行った．比較対象をインタープリタとしたのは，試作システムがインタープリタ的に動作するためである．同じ内容の知識ベースを SUN3/60 上の Quintus Prolog 1.6 上に入れ，setof 述語で同様の問い合わせについて検索した場合の実行時間から Dhrystone 1.1の値を使って Symmetry 上での処理速度を換算した．

この換算値を1とした時のプロセッサ台数による処理速度比の推移グラフを，インデックスを張らない場合と張った場合について，それぞれ図3と図4に示す．処理速度を抑えているモジュールを明確にするため，図3では GHC 用のプロセッサ台数を固定して RBU 用プロセッサ台数による変化を折れ線とし．図4では RBU 用プロセッサ台数を固定して GHC 用プロセッサ台数による変化を折れ線とした．

インデックスを張らない場合は検索モジュール側の，張った場合は推論モジュール側のプロセッサ台数の割合を増やすことにより，直線的に処理速度が向上する様子がわかる．このように，検索と推論の処理負荷の比重に合わせてそれぞれの割り当てプロセッサを増やすことにより，プロセッサ台数に対応した処理性能の向上が得られる．ただし，この検索処理と推論処理

```
traverse(KB,Arg1,Arg2) :- true |
    rbu_stream(RBU),
    file_stream(OUT),
    search_loop([[Arg1,Arg2,Arg1,Arg2],-1],KB,RBU,OUT).

search_loop([[Q1,Q2,Q3,Q4]|L],KB,R1,O1) :- Q1 \= nil |
    R1 = [urs(KB,[1=Q1,2=Q2,5=Q3,6=Q4],[3,4,5,6],S)|R2],
    S = [_,_|_],
    search_loop(S,KB,R3,O2),
    L = [_,_|_],
    search_loop(L,KB,R4,O3),
    merge(R3,R4,R2),
    merge(O2,O3,O1).
search_loop([[nil,empty,Q1,Q2]|L],KB,R,O1) :- true |
    O1 = [writeterm([Q1,Q2])|O2],
    L = [_,_|_],
    search_loop(L,KB,R,O2).
search_loop([-1,_|L],KB,R,O) :- true |
    L = [],
    R = [],
    O = [].
```

図2　RBU インタフェースを用いた GHC プログラム

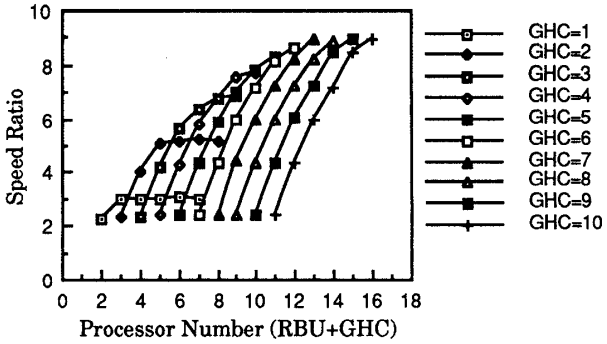

図3　プロセッサ台数による処理速度比の推移
（インデックスを張らない場合）

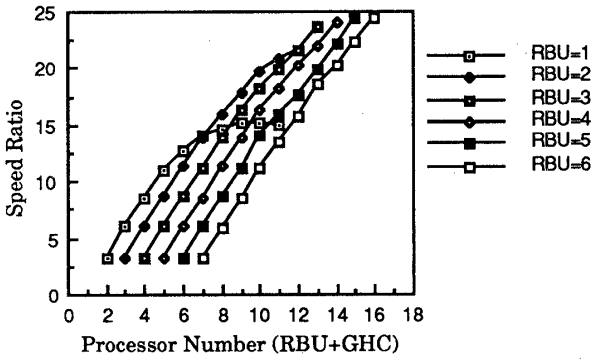

図4　プロセッサ台数による処理速度比の推移
（インデックスを張った場合）

のプロセッサ台数の比率は，知識ベースの大きさや問題の性質に依存するため，問題によってカスタマイズする必要がある．

4．おわりに

並列推論モジュールと並列知識検索モジュールからなる知識ベース指向並列処理システム KOPPS の構成と，試作システムを用いた実験について述べた．本構成で，推論と検索の負荷バランスを制御することによって，十分実用になる性能で処理可能なことが示せた．これより，KOPPS を並列マシン上での知識処理の実行環境として利用できる見通しを得た．

［謝辞］

日頃ご指導をいただく ICOT 内田 第4研究室長，ICOT KL1-TG メンバ，富士通研究所 林 人工知能研究部長，試作に御協力頂いた人工知能研の小沢，細井両氏，ならびに富士通 SSL の山崎，北島，竹中の3氏に感謝します．

［参考文献］

1. H. Yokota and H. Itoh, " A Model and Architecture for a Relational Knowledge Base" *Proc. of the 13th Int'l Sympo. on Computer Architecture*, pp. 2-9, 1986.
2. K. Ueda, "Guarded Horn Clauses," *Logic Prgramming '85*, E. Wada (ed). Lecture Notes in Computer Science 221, Springer-Verlag, 1986.
3. H. Yokota, H. Kitakami, and A. Hattori, "Term Indexing for Retrieval by Unification" *Proc. of 5th Int'l Conf. on Data Engineering*, pp.313-320, 1989.