特　集　「AIマシン」

並列論理型核言語に基づく知識ベースマシン

Knowledge Base Machine Based on Parallel Kernel Language

伊藤　英則*
Hidenori Itoh

*（財）新世代コンピュータ技術開発機構研究所
Research Center, Institute for New Generation Computer Technology (ICOT).

1987年8月17日　受理

Keywords: parallel logic programming, knowledge base machine, term, relational term, unification.

あらまし

知識は変数を含む論理構造体である項(term)で表現されたものとする．知識ベースマシンの1つのモデルとして，項ベースマシンを研究・開発している．この項ベースマシンは，並列論理型核言語に基づいた，並列推論機構および項ベースの蓄積・検索機構から構成される．ここでは，並列論理型核言語処理系がもつデータストリーム制御・並列制御機能を活かし，全解収集機能を補強した項ベースの単一化検索機構に重点を置いて述べる．

1．はじめに

新世代コンピュータ技術開発機構では，知識情報処理と並列処理コンピュータをロジックプログラミングで融合するパラダイム(1)から研究開発を行っている．

このため，まず始めに，従来のコンピュータの機械語に相当する核言語をロジックプログラミングにより定義して，これをシステムの中核に据える．しかもこの核言語は並列型である．次に，この並列型核言語を基にして下にはハードウェアシステムを，上にはソフトウェアシステムを構築する．ハードウェアシステムは，並列推論サブシステムと知識ベースサブシステムからなる．ソフトウェアシステムは，並列オペレーティングシステムと知識情報処理のための基本応用と実証ソフトウェアからなる．

知識ベースマシンの定まった明確な定義はいまだ存在していない．ここでは以下を前提にした知識ベースマシンについて述べる．

知識ベースマシンは，知識ベース機構と並列推論機構を兼ね備えるものとする．また，処理効率の観点から，応用プログラムが走行する並列推論機構と知識ベース機構とは密結合していて，これら2つの機構は単一のオペレーティングシステムの下で制御されるものとする，また，一階述語論理で用いられる変数を持つ論理的構造体である項(term)を知識とする．応用プログラム間で共有される知識の集合を知識ベースとする．なお，その意味については知識ベース管理プログラムで扱い，知識ベース機構では扱わないものとする．

この理由は以下である．これまでにも応用分野指向の知識表現言語が種々開発されているが，項はこれらのプリミティブ要素といえる．応用分野指向の知識表現言語から項への変換，またその逆変換処理も容易といえる．共有知識を格納・検索する知識ベース機構内の知識表現（の中間）言語を項とすれば，種々の応用プログラムへの汎用性が保証できる(2)．

以上を前提にするので，これ以降の議論は知識を項と読み替えることができる．以下に，知識ベースモデルとそのモデル上の操作・演算の定義，および並列論理型核言語に基づく知識ベースマシンへの要求機能を中心にして述べる．さらに，最後に知識ベースマシンモデルについても若干述べる．

2．知識ベースモデルと単一化検索

文献(7)で知識ベースモデルとして関係型項ベースモデルと，その上で単一化による検索基本演算を定義した．ここではこれらの概要と，単一化検索基本演算を効率化するために必要な項の整理技法について述べ

る．なお，関係型項ベースモデルは大量知識をもつマシン用の下位基盤のモデルであり，応用プログラムまでの上位モデルを包含するものではない．

2・1 関係型項ベースモデル

新しい知識ベースモデルとして，横田はルール集合とファクト集合を融合して持つ，関係型項ベースモデル[7]①を提案した*．すなわち，項 $t_1, \ldots, t_n$ が関係Rにあることを以下のように定義した．

$$R(t_1, \ldots, t_n) \qquad ①$$

これは関係型データモデルのデータを，項にまで拡張したものである（蛇足ながら前述のファクトは t_i で表現できる）．この拡張を最も単純に解釈すれば，項の格納の仕方，検索の仕方，項間の関連づけなど（従来のビュー（view）に相当）もまた項で表現できる．さらに極端な解釈もできる．たとえば，Rを，t_1 がPrologプログラムの節（clause）のヘッド，t_2，…，t_n がそのボディの関係とすれば，Prologプログラムの節を知識として扱えることになる．この解釈では，知識とプログラムの区別をしなくてよいことになる．どの解釈でシステム化するかは適用する応用による．以降，この知識ベースモデルを並列論理型核言語による並列知識ベースマシンPKBM[6][8]で扱うことについて述べる．

2・2 単一化検索

2・1節で述べた関係型項モデル上で検索基本演算について述べる．関係型データモデル上で定義される演算の基本はデータの等価性チェックであった．関係型項モデル上でも項の等価性チェックをその演算の基本とすることもできるが，これでは，項のパターン整合によるフィルタリング処理ができる程度にとどまってしまう．そこで，文献(7)では項の単一性チェックにまで拡張した．これを基本にして検索することを単一化検索（RBU：Retrieval By Unification）と名付けた．

$$\sigma f(a,x) \diamond 1 \left\{ \begin{matrix} (f(a,y),g(y,z)) \\ (f(b,z),g(a,x)) \\ (f(w,w),g(c,w)) \end{matrix} \right\}_{[1,\ 2]} = \left\{ \begin{matrix} (f(a,x),g(x,z)) \\ (f(a,a),g(c,a)) \end{matrix} \right\}_{[1',\ 2']}$$

◇：単一化制約

図1 単一化制約演算の例

制約関係演算と単一化演算を融合して単一化制約演算を，また，結合関係演算と単一化演算を融合して単一化結合演算を定義した[7][9]．

ここで，図1に単一化制約演算と，図2に単一化結合演算の簡単な例を示す．

図1の左辺の単一化制約演算子◇の左側をクェリ，右側を項関係集合，右辺を単一化制約演算によって検索された項関係集合と解釈する．図2も同様の解釈が可能である．

この基本演算を用いて，知識ベースを前述のPrologプログラムであると解釈した場合の検索アルゴリズムを図3に示す．

次に，この単一化検索アルゴリズムを用いて図4に示す先祖親子関係知識から“a”の子孫“X”を求めることができる．その単一化検索過程の一部を図5に示す．

単一化検索アルゴリズムで単一化関係演算を繰り返して最終解まで求められる（図8で後述）．実際の応用では解の中間結果を推論機構に渡して，システム全体として処理効率を上げることもできる．この最適化は個々の応用ごとに定めることになる．知識ベース機構では，たとえば，解の候補を10分の1から1000分の1に絞るだけにとどめることもできる．このときは知識ベース機構内では大量の知識に浅い推論処理を繰り返し適用すればよいことになる．

$$\left\{ \begin{matrix} (f(a,y),g(x,y)) \\ (f(b,z),g(a,a)) \\ (f(v,v),g(b,y)) \end{matrix} \right\}_{[1,\ 2]} \underset{2\diamond 1'}{\bowtie} \left\{ \begin{matrix} (g(a,z),h(z,b)) \\ (g(b,w),h(y,c)) \end{matrix} \right\}_{[1',\ 2']} = \left\{ \begin{matrix} (f(a,z),g(a,z),h(z,b)) \\ (f(b,a),g(a,a),h(a,b)) \\ (f(a,w),g(b,w),h(y,c)) \\ (f(v,v),g(b,w),h(y,c)) \end{matrix} \right\}_{[1'',\ 2'',\ 3'']}$$

図2 単一化結合演算の例

2・3 項の順序づけ，索引づけ

大量の項の単一化検索演算を効率的に処理するためには，項の順序づけ，索引づけを行うことが重要である．大量データについてはこれらはすでに確立されて

---

* これまでに，ファクト集合を関係型データモデルととらえ関係データベースマシンDELTA[3][4]を，汎用コンピュータを基礎として開発した．また，ルール集合とファクト集合に分離して双方を持つ演繹データベースマシンPHI[5][6]を，逐次型推論マシン（SIM：Sequential Inference Machine）を基礎にして開発している．

```
(R : result) ← φ
K0 ← σgoal ◇ head (KB : term relation)
i ← 0
    while Ki = φ do
  begin
    R ← (Ki の body が [] の Ki の head) UR⋈
    Ki+1 ← ∏Ki の head, KB の body (Ki body ◇ head KB)
    i ← i + 1
  end (ただし, ∏は通常の射影)
```

図 3 単一化検索アルゴリズムの例

KB	head	body
	[an (A,B) | Tail]	[pa (A, B) | Tail]
	[an (A,B) | Tail]	[pa (A, C), an (C, B) | Tail]
	[pa (a, b) | Tail]	Tail
	[pa (b, c) | Tail]	Tail

図 4 先祖・親子関係知識

K_0	head	body
	[an (a,x)]	[pa (a,x) | Tail]
	[an (a,x)]	[pa (a,C), an (C,x) | Tail]

(a) 単一化制約演算例 ($K_0 = \sigma_{an(a,x) \diamond head}$ KB)

k_1	1	2	3
	[an (a,b)]	[pa (a,b)]	Tail
	[an (a,x)]	[pa (a,b), an (b,x)]	[an (b,x) | Tail]

(b) 単一化結合演算例 $k_1 = k_{0\ body} \diamond\!\!\bowtie_{head}$ KB

k_2	1	3
	[an (a,b)]	Tail
	[an (a,x)]	[an (b,x) | Tail]

(c) 射影例 ($K_1 = \prod_{1,2,3 \to 1,3}[k_1]$)

図 5 単一化検索過程

いる．ここでは，これらを項に拡張したこれまでに提案した事項の概要を述べる．

〔1〕 **項の順序づけ**

単一化結合演算では2つの項集合間の全ての組合せを試みることが要求される．ここでデータの場合と同様に，項がある条件で整列していれば組合わせ処理量を減らすことができる．この条件に，置換からみたジェネラリティの概念を導入した[7][9]．このジェネラリティは②を満たす半順序関係である．

$$t_2 \geq t_1 \text{ iff } {}^{\exists}\rho,\ t_2\rho = t_1 \quad (\rho\ ;\text{置換}) \qquad ②$$

3章で後述するように，項をストリームとして処理するには，このジェネラリティを保存しながらさらに全順序づけを行う必要がある．文献(18)では，項が木で表現できることに注目して，与えられた木を線形表現し，その文字列を辞書式に全順序をつけた．

木の線形化表現には，家族順とレベル順がある．これらの単一化結合演算における処理効率の比較評価の詳細については文献(18)を参照されたい(文献(18)では木を線形化したものをさらにトライ(trie)化して組合わせ処理量を減らすことと，項の格納スペースを縮めることの効果の評価も行っている)．

〔2〕 **項の索引づけ**

従来のデータの重ね合わせ方式の概要は以下である．ファイルFのレコードをR_iとする．R_iのいくつかのキーワードにある関数(ハッシュ関数)を施してその2進符号語(BCW)を作成する．あるレコードR_iに対して，その全てのキーワードのBCWをビットごとにOR演算したものを重ね合わせ符号語(SCW : Superimposed Code Word) S_iと呼ぶ．次に，これと同一関数で検索キーワードのBCWを作成する．また，検索キーワードが複数個あれば同様にそのSCWを作成する．これをクェリマスクQとする．このとき，③によってS_iからふるい落とされたレコードの中から，本来検索したいレコードを選び出す．

$$Q \wedge S_i = Q \quad (\wedge : \text{and 演算}) \qquad ③$$

ここでは項の重ね合わせとその単一化検索に拡張する．重ね合わせ符号を用いた項の検索については，主にPrologの処理系の高速化の技法として研究されてきている．森田は，検索される側のレコード項に含まれる変数に対しては1…，(または0…)と符号化し，検索する側のクェリ項に含まれる変数に対しては0…，(または1…)とする新しい方法を提案した．項を木表現したときの節・葉にあたる記号と変数は，その位置が深くなればなるほどその符号長を短くして重ね合わせる符号化方式である．これをSSCW (Structural SCW)[10][11]と名付けた．この方式により，適当なハッシュ関数を定めれば単一化からみた項の連想サーチも可能とすることができる．

BCW長，ハッシュ関数，単一化率，および項の絞り込み率についての評価および他の方式との比較は文献(12)を参照されたい．

3. 並列論理型核言語の処理系と単一化検索の結合

まず，並列論理型核言語のシンタックス，セマンティックスとその処理系について述べる．次に2章で

述べた，単一化検索処理と並列論理型核言語の処理系を結合する方法について述べる．

3・1 GHC (Guarded Horn Clauses)

上田は並列論理型核言語としてGHCを提案した[13]．以下にそのシンタックス，セマンティックスとその処理系の概要を，以降の議論に必要な部分に絞って述べる．

〔1〕 GHCのシンタックス

GHCのシンタックスは以下である．

$$H :- G_1, \cdots, G_n \mid B_1, \cdots, B_m. \qquad ④$$

∣は許可記号であり，∣の左側がガード部，右側がボディ部である．他の記号は通常のPrologで使用されているものである．

〔2〕 GHCのセマンティックスとその処理系

GHCの場合もPrologの場合と同様に，与えられたゴール節から空節を導き出す．このときのルールも極めて単純で，以下である(正確には文献(13)を参照されたい)．

(1) ガード部が許可されるまではボディ部とガード部の変数は単一化できない．

(2) 同じヘッドがある場合は最初にガード部が許可された節のボディ部のみ処理を続行する．このとき，その他の候補であった同じヘッドを持つ節は消去する．

(3) ボディ部はAND並列で実行する．

ここに，(2)はドントケア非決定性(don't care non-deterministic)処理である．

次に，GHCによる簡単なプログラムとその処理系を図6に例示する．ここにp(X), q(X)は共有変数Xを持ったプロセスである．このプログラムは共有変数Xを介して，データストリーム[a∣Y]が流れる．p(X)はデータストリームジェネレータ，q(X)はデータストリームコンシューマであり，双方はデータストリームの送受に同期をとる．

図6に示したように，GHC処理系は，与えられたプログラムを並列ゴールプール管理，スケジューラ，ガードテスト，ボディ実行のサイクル順に実行する．

3・2 GHCの処理系と単一化検索の結合

GHCの処理系と単一化検索の結合方法について述べる．

GHCの特徴の1つであるデータストリームによるプロセスの入出力処理と大量の項関係演算処理を論理的に結合できれば，後述する並列推論機構と知識ベース検索機構とからなる知識ベースマシンの構築が容易であり処理の効率化が図れる．ただし，GHCは処理系の単純さから並列推論処理の性能を追求するためにドントケア非決定性処理を採用していて，単一化検索で要求される全部の解を収集する処理とは相矛盾する．何らかの手立てが必要である．

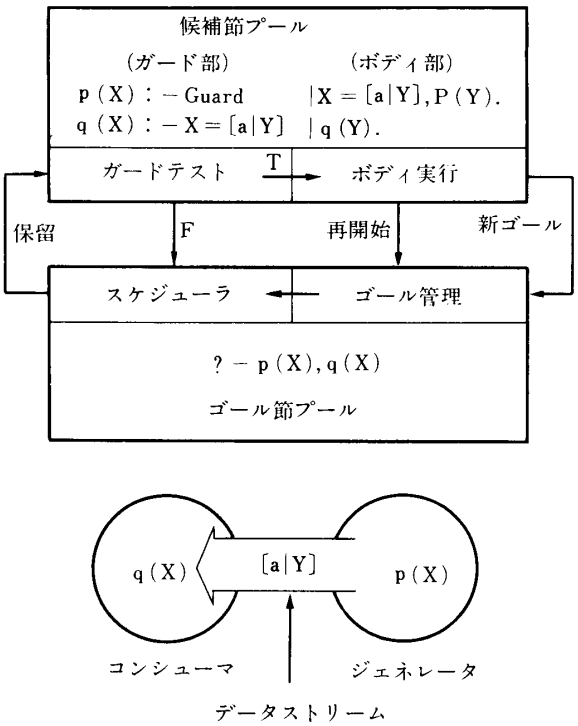


図6 GHC処理系と簡単なプログラム例

このために，GHCの処理系に組込み述語を取り入れてこの矛盾を解決した[8]．図7に組込み述語"rbu"を持つプログラム例を示す．ここにプロセス"loop"はコマンドジェネレータであり，"rbu"はコマンドコンシューマである．プロセス"solve"は"rbu"と"loop"で全ての解を収集するまで消滅しない．第1番目の"loop"節は終了条件であり，第2番目の"loop"節は"rbu"からの答をスタックする．また，最後の"loop"節は与えられたKBとGOALから単一化検索を行い，その時点の中間結果Sとこれまでの結果Lをマージして新中間結果Nとする．また，Sはプロセス"loop"とプロセス"rbu"間のResultがデータストリームとして流れるチャネルでもある．図4で示した先祖親子関係知識KBを例にして，"a"の子孫を検索する"rbu"の内部処理を図8に示す．"unification-restriction"は図3のアルゴリズムを実行する．特に，[1, 2], [1, 3]は図5のプロジェクションを意味する．図7のプログラムはコマンドストリームC_1を受けながら，S_1から順に解を求めていく．この場合，R = [ ]であるS_2とS_5が解集合であり，図7のResultにスタックされる．また，S_8とS_9が[ ]であり，第1番目の"loop"節の終了条件を満たし計算

```
solve(KB,Goal,Result) :- true |
       loop(KB,cmd(C1,C2),[[[Goal],[Goal]]],Result),
       rbu(cmd(C1,C2)).

loop(KB,cmd(C1,C2),[]  ,X) :- true  | X = [], C1=C2.
loop(KB,CMD       ,[[G,R]|L],X) :- R = [] |  X = [G|Y],
       loop(KB,CMD,L,Y).
loop(KB,cmd(C1,C3),[[G,R]|L],X) :- R \= [] |
  C1 = [unification_restriction(KB,[1=G,2=R],[1,3],S)|C2],
       merge(L,S,N),
       loop(KB,cmd(C2,C3),N,X).
```

図 7　組込み述語 rbu と GHC の結合

```
unification_restriction(kb, [1=[an(a,X)],2=[an(a,X)]],[1,3],S1)
  S1 = [[[an(a,X)],[pa(a,X)]],[[an(a,X)],[pa(a,C),an(C,X)]]]
unification_restriction(kb, [1=[an(a,X)],2=[pa(a,X)]],[1,3],S2)
  S2 = [[[an(a,b)],[]]]
unification_restriction(kb, [1=[an(a,X)],2=[pa(a,C),an(C,X)]],[1,3],S3)
  S3 = [[[an(a,X)],[an(b,X)]]]
unification_restriction(kb,  [1=[an(a,X)],2=[an(b,X)]],[1,3],S4)
  S4 = [[[an(a,X)],[pa(b,X)]],[[an(a,X)],[pa(b,C),an(C,X)]]]
unification_restriction(kb, [1=[an(a,X)],2=[pa(b,X)]],[1,3],S5)
  S5 = [[[an(a,c)],[]]]
unification_restriction(kb, [1=[an(a,X)],2=[pa(b,C),an(C,X)]],[1,3],S6)
  S6 = [[[an(a,X)],[an(c,X)]]]
unification_restriction(kb, [1=[an(a,X)],2=[an(c,X)]],[1,3],S7)
  S7 = [[[an(a,X)],[pa(c,X)]],[[an(a,X)],[pa(c,C),an(C,X)]]]
unification_restriction(kb, [1=[an(a,X)],2=[pa(c,X)]],[1,3],S8)
  S8 = []
unification_restriction(kb, [1=[an(a,X)],2=[pa(c,C),an(C,X)]],[1,3],S9)
  S9 = []
```

図 8　rbu コマンド単一化検索例

が終了する．なお，ここでは説明の単純化のために単一化制約演算だけを用いた．

またここでは Goal から単純にトップダウンに検索したが，トップダウンとボトムアップ検索を組み合わせた効率的アルゴリズムの開発も盛んである[14]-[17][28]．

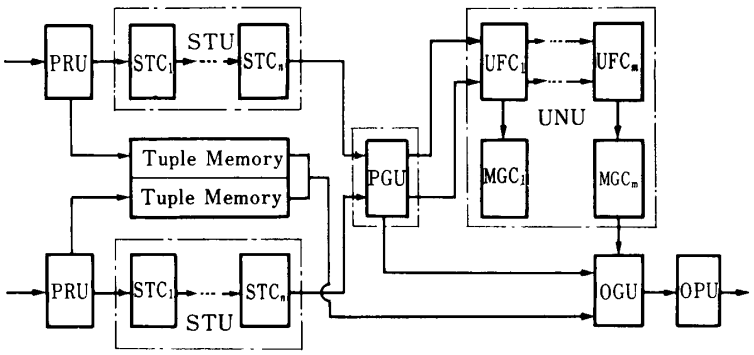
4．RBU 専用装置の構成

これまでの議論は，知識が主記憶上にあるか外部記憶上にあるかを区別する必要はなかった．これから以降の議論は知識が外部記憶上にあることを前提にする．特に，大量の項が外部記憶に蓄積されているか，または多量のデータに混じって項が蓄積されている場合を想定する．このように大量に蓄積された知識の単一化関係演算による検索は，大部分が単純演算の繰返しである．この繰返し演算部分を専用装置でオンザフライ処理すれば，システム全体の効率化が図れる．このために，3 章で述べた組込み述語“rbu”の専用ハードウェアを実現する．この専用装置を単一化検索専用装置（RBU専用装置）[9] と呼び，並列推論機構と外部記憶機構の間に複数台設定する．並列推論機構と RBU 専用装置の入出力インタフェースは，3 章で述べた推論プロセスと検索プロセス間のそれと同様にデータストリーム形式にできる．

RBU 専用装置の構成概念を図 9 に示す．項ソートユニット（2 個），ペア生成ユニット，単一化ユニットから構成する．項ソートユニットは 2・3 節〔1〕で述べた全順序に従い項を整列させる．ペア生成ユニットは項ソートユニットからの出力を入力として，項のペアを作成し，単一化ユニットに出力する．ただし，ここでの項のペアは可能な限り単一化可能性のあるものにだけ絞られる．単一化ユニットは項のペアに対して単一化操作を実行する．これらのユニット全体でデータストリームをパイプライン処理実行[18] する．

各ユニットは，DELTA で開発した関係演算専用装置[3][19] の構成との相似としてみることができる．項ソートユニットはデータのソートユニット，ペア生成ユニットはマージャ，単一化ユニットは関係演算器にそれぞれ相当する．

項ソートユニットでも 2 way-sort/merge アルゴ



PRU：前処理ユニット　PGU：ペア生成ユニット　OGU：出力項生成ユニット
STC：ソートセル　MGC：最汎単一化処理セル　OPU：後処理ユニット
STU：ソートユニット　UNU：単一化ユニット

図 9　RBU 専用装置の構成概念図

リズム[20]が使える．また，このアルゴリズムにより項の整列も入力量nに対してオーダnの処理時間を保証できる（ただし，n≦ソートユニット内レジスタ容量）．データの場合は，2 way-sort/merge アルゴリズムによりマージ処理はマージャで同様にオーダnの処理時間を保証できたが，ペア生成ユニットでは項のペア生成は，置換によるジェネラリティが半順序であるので最悪はオーダn^2かかる．単一化ユニットでは，最汎単一子（most general unifier）を求めてから単一化を実行することにより，無駄な項の流れを最少にして処理の効率化が図れる[18]．

5．RBU 専用装置の GHC による並列制御

十分大きな粒度の項関係には，それを分割して並列に処理すれば効率的である．そのために RBU 専用装置を複数台備える．すなわち，3 章で述べた“rbu”コマンドの処理を 4 章で述べた複数台の RBU 専用装置で並列実行する．なお，GHC で定義されるデータストリームによって 1 つ 1 つの RBU 専用装置へ起動がかかり，並列処理が実行される．

1 つの“rbu”コマンドが指定する演算対象を，複数台の RBU 専用装置に割り付けるには下の 3 つの方法がある．

(1)　演算対象を全て 1 つの専用装置に割り付ける．
(2)　その時点で空いている専用装置に分割して割り付ける．
(3)　常に固定数（たとえば専用装置台数）に分割して割り付ける．

(1)は分割処理が不必要であり，演算対象が比較的小さくコマンド件数が多いときに有利である．(3)はその逆であり，分割処理が必要であり，演算対象が比較的大きくコマンド件数が少ないときに有利である．(2)は分割処理と，特に専用装置の現在のステータス（空/稼動中）の監視が必要である．コマンド件数が多いときは(1)に近く，コマンド件数が少ないときは(3)に近くなる[21][22]．ステータス監視処理負荷が小さければ，(2)が効率的な部分は(1)，(3)の中間に存在する[21]．図 10 にステータス監視機能をもつ方法(2)の GHC によるプログラムを例示する．

これらの 3 つの方法のうちどれがその時点で最適かは，そのときの“rbu”コマンド種別，処理すべき対象の粒度，専用装置の稼動状況と，その制御方式自身の処理量によって決められる．特に，“rbu”コマンド種別により専用装置の並列使用形態が異なる．たとえば，大量の項のソート処理はデータの場合と同様に多段フェーズが必要である．第iフェーズで専用装置をn_i台使用したとすれば，第$i+1$フェーズではn_iの半分を使用する．また，単一化制約演算系では処理フェーズは 1 段ですむので可能な限り多くの専用装置を用いたほうが有利である．単一化結合演算は項のソート処理後に実行したほうが得策である（これらの詳細評価については文献 (22) を参照されたい）．

このような最適化を考慮したメタコントロールも GHC により記述できる．メタコントロールプログラム例を図 11 に示す．図 10 の“REscheduler”を図 11 のプログラムで置き換えれば，RBU 専用装置のメタレベルまでの動的並列制御が可能である．

6．知識ベースマシンモデル

知識ベースマシン（KBM）の概念モデルを図 12 に

```
% Top level
'REscheduler'([C|T],Stream) :- true |
   availableREs(Stream, NS,Free), divide(Free,[C|T],NS).
'REscheduler'([]    ,Stream) :- true | closeStream(Stream).

availableREs(St, NS,Free) :- true |
   inspect(St,NS,Ins), checkingFree(Ins,Free).

% Inspection of REs status
inspect([]                  ,New,Ins) :- true | New=[], Ins=[].
inspect([stream(N,St)|Rest],New,Ins) :- true |
   New=[stream(N,SR)|NR], Ins=[(N,State)|IR],
   St=[ins(State)|SR], inspect(Rest,NR,IR).

divide([] ,C     ,St) :- true     |
   'REscheduler'(C,St).                          % All REs are busy.
divide(REs,[C|T],St) :- REs\=[] |                % send out C to free REs
   division(C, REs, SubC), sendOut(REs,SubC,St,NS),
   'REscheduler'(T,NS).

% 'RE' manages status of REs, SendToRE sends C to Nth RE.
'RE'(N,[term  |Cmd]) :- true | Cmd=[].          % termination
'RE'(N,[ins(C)|Cmd]) :- true |
   C=free, 'RE'(N,Cmd).                          % inspection of status
'RE'(N,[cmd(C)|Cmd]) :- true |                   % retrieval command
   sendToRE(N,C,Res), response(Res,Cmd,Next), 'RE'(N,Next).

response(end,Cmd          ,N) :- true | Cmd=N. % Process ends.
response(R  ,[ins(C)|Cmd],N) :- true | C=busy,response(R,Cmd,N).
```

図 10 GHC による RBU 専用装置並列制御プログラム（例）

```
metaControl([] ,                        ST, REs) :- true |
      closeStream(ST).
metaControl([cmd(C,Type,Size)|Rest], ST, REs) :- true |
      availableREs(ST, IS, Free),
      strategy(C,Type,Size,Free, Method),
      solve(Method, REs, Free, cmd(C), IS, NS),
      metaControl(REs, Rest,NewST).

solve(method1, REs, Free, C, ST, NS) :- true |
      selectRE(Free,RE), sendOut([C],[RE],ST,NS).
solve(method2, REs, Free, C, ST, NS) :- true |
      division(C,Free,SubC), sendOut(SubC,Free,ST,NS).
solve(method3, REs, Free, C, ST, NS) :- true |
      division(C,REs, SubC), sendOut(SubC,Free,ST,NS).
```

図 11 GHC によるメタコントロールプログラム（例）

示す．

知識ベースマシンは並列推論機構と知識ベース機構から構築される．上段のネットワーク（NW）とプロセスエレメント（PE）のクラスタ（CL）は，並列推論マシン（PIM）として研究開発されている[23]．知識ベース機構は，複数の RBU 演算専用装置（RE）とそれを接続する下段の NW およびグローバル共有記憶機構（GSM）から構成する．GSM は各 CL からアクセスされる．また，各 CL 内にローカル共有記憶機構（LSM）がある．LSM と GSM により階層化共有記憶機構を構成する[24]．上段と下段の NW は物理的構成は同一となろう．

RE は CL から 4 章で述べた RBU 検索コマンドを NW を介して受ける．なお，RE 台数は CL 台数の約 1/10 を相定する．NW は RBU 検索コマンドを動的に RE に割り付ける．6 章で述べたように，これには GHC の並列制御機能を活かして実行する．NW に接続している CL と RE は，単一の並列オペレーティングシステム（POS）によってプラグマティックに制御される．特に，RE は POS の KBM クラスとして

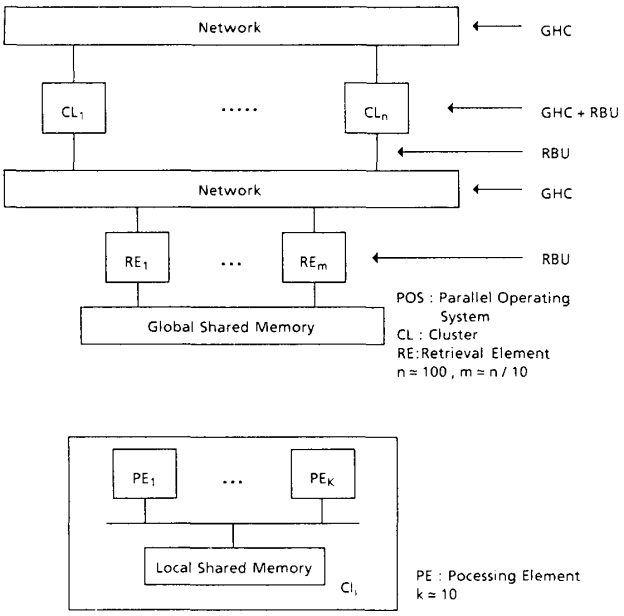


図 12 知識ベースマシンの構成概念図

並列制御される.

RE の内部構成・機能については 5 章で詳述した. 再述するが CL との入出力はデータストリーム形式である.

GSM はいくつかのメモリバンクをもっている. 各バンクはリード・ライト専用ポートを持つ. GSM では論理的ページを各バンクにまたがらせて蓄積・管理するので, 同一ページが複数のポートから同時にアクセスされ得る[25]. 共有記憶機構への多重アクセスメカニズムは並列プロセッサの環境では特に重要である. ページサイズ, バンク/ポート数, リード・ライト速度, 検索対象の知識粒度の相関の評価については文献 (26), (27) を参照されたい.

7. おわりに

項を知識として, 前半ではまず, マシン用の下位基盤モデルとして関係型項ベースモデルを提案し, このモデル上で単一化演算と関係演算を結合した RBU 演算について述べた. また, 並列論理型核言語 GHC の下で RBU 演算を可能とするための方策について 4 章で詳述した. 以下に要約する.

GHC, RBU 双方ともに一階述語論理を基礎としていることと, 以下の 2 つの点で GHC の下で RBU 演算処理を行うことは親和性がよい.

(1) RBU 演算専用装置の入出力処理に, GHC で定義されているプロセスのデータストリーム型入出力機能がそのまま使える.

(2) 大量の項関係演算処理の並列実行に GHC の並列処理系が使える.

ただし, 条件を満足する全部の知識を検索収集することが要求される知識ベース検索処理と, GHC のドントケア非決定性処理 (ある時点で 1 つの解が求まると同時にその他の候補の解の検索は全て放棄する) とは親和性がない. このために, 同一条件で検索処理が続行できるようにする組込み述語を, GHC 処理系に付加して RBU 演算で全知識の収集を可能とした.

後半では, GHC に基づいた, 並列推論機構および複数の RBU 演算専用装置をもつ知識ベース機構から構成される並列知識ベースマシン (PKBM) モデルについて述べた. 現在は, この並列推論機構および知識ベースの蓄積・検索機構の実験機の開発とその構成要素の部分的評価を行っている. またさらに, システム全体の評価のために, 大量知識の検索を必要とする組合せ・協調問題の応用プログラムの作成に着手している.

謝 辞

知識ベースマシンの研究開発にあたって, 有益なご示唆をいただいた渕所長に感謝します. また, ここで引用したプログラム例[8]は, ICOT に在籍された横田 (富士通 (株)), 武脇 ((株) 東芝) 氏によるものを引用させていただいた. ここに感謝します. また, 日ごろ熱心なご討論に参加されている第一・第三研究室の諸氏, 特に, 田中, 森田両氏, および KBM ワーキンググループと関連株式会社の皆様方に感謝します.

◇ 参 考 文 献 ◇

(1) Fuchi, K. : Revisiting Original Philosophy of Fifth Generation Computer Systems Project, Proc. of the Int'l. Conf. on Fifth Generation Computer Systems, ICOT (1984).

(2) 横田, 伊藤 : 知識ベースシステム, 計測自動制御学会誌, Vol. 25, No. 4 (1986).

(3) 岩田, 柴山, 酒井, 伊藤, 村上 : 関係代数専用装置の評価, 情報処理学会誌, Vol. 28, No. 7 (1987).

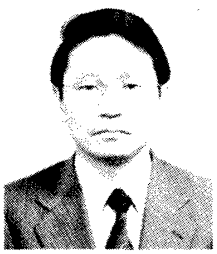
(4) Murakami and Shibayama, *et al.* : A Relational Database Machine : First Step to Knowledge Base Machine, Proc. of 10th Ann. Int'l. Symp. on Computer Architecture (1983).

(5) 伊藤, 森田, 大場, 山崎 : KBMS PHI (1)分散知識ベースシステムのシステム構成方式, 情報処理学会第 32 回大会, 3 (1986).

(6) Itoh : Research and Development of Knowledge Base

Systems at ICOT, The 12th Int'l. Conf. on VLDB (Aug. 1986).
(7) Yokota and Itoh : A Model and an Architecture for a Relational Knowledge Base, The 13th Int'l. Symp. on Computer Architecture, pp. 2-9, IEEE (1986).
(8) Itoh, Takewaki and Yokota : Knowledge Base Machine Based on Parallel Kernel Language, The 5th Int'l. Workshop on Database Machines (Oct. 1987).
(9) Morita, Yokota and Itoh, *et al.* : Retrieval-by-Unification Operation on a Relational Knowledge Base Model, The 12th Int'l. Conf. on VLDB (Aug. 1986).
(10) 森田, 和田, 伊藤：スーパインポーズドコードを用いた構造体の検索方式, 情報処理学会第33回大会, 10 (1986).
(11) Wada, Morita, Miyazaki and Itoh, *et al.* : Performance Evaluation of Superimposed Code Scheme for Relational Operations, The 5th Int'l. Workshop on Database Machines (Oct. 1987).
(12) 森田, 物井, 中瀬, 柴山：構造体のインデックス方式に関する一考察, 情報処理学会第35回大会, 10 (1987).
(13) Ueda : Guarded Horn Clauses, Logic Programming '85, E. Wada (ed), Lecture Notes in Computer Science 221, Springer-Verlag (1986).
(14) Yokota, Sakai and Itoh : Deductive Database System Based on Unit Resolution, The Second Data Eng. Computer Int'l. Conf., pp. 228-235, IEEE (1986).
(15) Bancilon, *et al.* : An Amateur's Introdution to Recursive Query Processing Strategies, ACM SIGMOD '86, pp. 16-52 (1986).
(16) Ceri, *et al.* : Translation and Optimization of Logic Queries : The Algebraic Approach, VLDB '86, pp. 395-402 (1986).
(17) Miyazaki and Itoh : Restricted Least Fixed Points and Recursive Query Processing Strategies, ICOT TR-183, (1987).
(18) Morita, Oguro and Itoh : Performance Evaluation of a Unification Engine for a Knowledge Base Machine, ICOT TR-225 (1987), または, 情報処理学会研資データベースシステム, 57-4, 1 (1987).
(19) Itoh and Oba, *et al.* : Design and Evaluation of Retrieval Database Engine for Variable Length Records, The 5th Int'l. Workshop on Database Machines (Oct. 1987).
(20) Todd : Algorithm and Hardware for Merge Sort Using Multiple Processors, IBM Journal of Research and Development, 22 (1978).
(21) Itoh, Abe, Sakama and Mitomo : Parallel Control Techniques for Dedicated Relational Database Engines, The Third Int'l. Conf. on Data Eng. (Feb. 1987).
(22) 武脇, 伊藤：並列論理型言語による検索処理プロセスの並列制御とその評価, 人工知能学会第1回全国大会 (1987. 7).
(23) Goto and Uchida : Toward a High Performance Parallel Inference Machine, ICOT TR-201 (1986).
(24) Itoh and Uchida : Parallel Inference Machine and Knowledge Base Machine, Computers and Communications Technology Toward 2000, IEEE Region 10 Conf. (Aug. 1987), TENCON '87, ICOT TR-234.
(25) Tanaka : A Multiport Page-Memory Architecture and a Multi-Port Disk-Cache System, New Generation Computing, Vol. 2, pp. 241-260, OHMSHA, (Feb. 1984).
(26) Monoi, Morita and Itoh, *et al.* : Parallel Control Technique and Performance of a MPPM Knowledge Base Machine, The Fouth Int'l. Conf. on Data Eng. to appear (Feb. 1988). ICOT TR-284, または, 情報処理学会研資データベースシステム, 60-4, 7 (1987).
(27) Sakai, Shibayama, Monoi and Itoh : Knowledge Base Machine using Multi-Port Page Memory, The 5th Int'l. Workshop on Database Machines (Oct. 1987).
(28) 世木：演繹データベースにおける再帰問合せ処理の一方法について, 情報処理学会第35回大会, 9 (1987).

著者紹介



伊藤　英則（正会員）

昭和44年3月福井大学工学部卒業. 昭和49年3月名古屋大学大学院工学系研究科博士課程電気・電子専攻修了. 工学博士. 昭和49年4月, NTT電気通信研究所入社. 現在, (財)新世代コンピュータ技術開発機構に勤務, 研究室長. これまでに, オートマトンと数理言語理論の研究とDIPS大型計算機オペレーティングシステムの研究開発に従事. 現在, 人工知能, 知識情報処理システムの研究開発に従事. 電子情報通信学会, 情報処理学会, 人工知能学会各会員.