TM-0249

GHC による関係演算処理プロセスの並列制御方式

三友雄司(日本システム)
宮崎収兄(沖電気工業)
武脇敏晃，伊藤英則

January, 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

GHCによる関係演算処理プロセスの並列制御方式

武脇敏晃、伊藤英則 (ICOT)、
三友雄司 (日本システム)、宮崎収兄 (沖電気工業)

1. はじめに

本稿では、並列論理型言語GHC[1]によって複数プロセスを並列に制御する方法について述べる。ここで考えているプロセスの処理対象は、ファクト・ベース(またはルール・ベース)である。また、並列制御は与えられたゴールに対する検索処理の対象オブジェクトの量、および、コマンドの種別による。

ここで述べる方式の評価の重点は、処理粒度と並列度の相関である。この方式を知識ベースシステム構築に適用することを提案する。

2. 知識ベース・システム

知識ベース・システムの最初の段階として、ホーン節の集合を知識とし、これを蓄え管理する演繹データベース・モデル(図1)を考える。これは、知識ベース・システム内で、外部データベース(ファクトの集合:以下、EDB)および内部データベース(EDBの使い方に関するルールの集合[view定義]:以下、IDB)を管理し、ユーザまたは応用プログラムは、知識ベース・システムに対してホーン節でIDBおよびEDBに対して問い合わせを行う三階層結合モデル[2]である。また、データベース管理層は、検索プロセッサ並列制御部、複数の検索プロセッサ、EDBの3つの部分からなる(図2)。

以下では、複数ある関係演算プロセスをどのように並列制御すればよいかについて述べる。

3. プロセスの並列制御

プロセスを並列に制御するために関係するパラメータは、プロセッサの数、コマンドの種別、対象データ量などである[3]。本節では、プロセッサの数を考慮した並列制御について述べ、次節では、コマンドの種別と処理粒度から並列性について述べる。

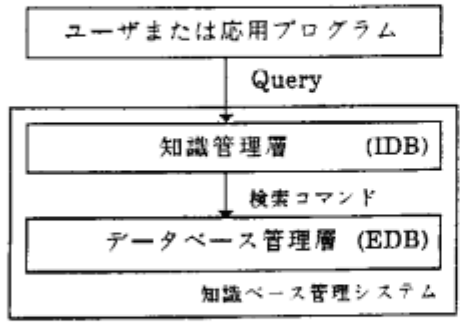


図1 三階層結合の知識ベース・モデル

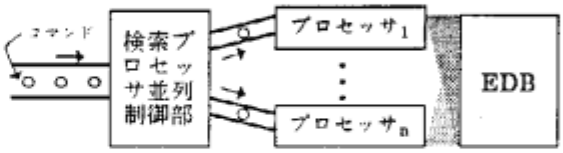


図2 データベース管理層

データベース管理層において、プロセスを並列に制御するために3つの戦略を用いた。以下に出現する記号n,iはそれぞれ、検索プロセッサの総数、ある時点における空きプロセッサ数を示している。

(1)データ非分割法(以下、戦略1)
これは、知識管理層からコマンドを1つ受け取ると、空きプロセッサを1台見つけ、そのプロセッサにコマンドを割り付ける。

(2)データ可変分割法(以下、戦略2)
これは、知識管理層からコマンドを1つ受け取ると、その時点での空きプロセッサをすべて(i台)見つけ、対象データをi分割することによって、i個のサブ・コマンドを生成し、各空きプロセッサにサブ・コマンドを割り付ける。

(3)データ固定分割法(以下、戦略3)
これは、知識管理層からコマンドを1つ受け取ると、対象データを一定数(例えばn)で分割することによって、n個のサブ・コマンドを生成し、戦略1のように空きプロセッサを1台見つけては、サブ・コマンドを1つ割り付けていく。

戦略1と3は空きであることだけがわかればよく、仕事をしているプロセッサを知る必要がない。このため並列制御部は、プロセッサからのデマンドで駆動させることができる。

戦略2は、両方の状態を知る必要があるため、制御部内で状態を管理し、空きプロセッサに仕事を割り当てていくようにする。図3は、プロセッサの状態を保持する述語'RP'である。引数はそれぞれ、プロセッサ番号、コマンド列である。述語sendToRPはプロセッサにコマンドを送るもので、処理が終了すると第三引数にendがインスタ

Parallel Control Techniques for Retrieve Processes in GHC

Toshiaki TAKEWAKI, Hidenori ITOH (ICOT),
Yuji MITOMO (Japan Systems Co. Ltd.), Nobuyoshi MIYAZAKI (Oki Electric Industry Co. Ltd.)

```
'RP'(N,[term  |Cs]) :- true | Command=[].
'RP'(N,[ins(C)|Cs]) :- true |
    C=free, 'RP'(N,Command).
'RP'(N,[cmd(C)|Cs]) :- true |
    sendToRP(N,C,Res),
    response(Res,Cs,NCs), 'RP'(N,NCs).

response(end,Cs         ,NCs) :- true |
    Cs=NCs.
response(R  ,[ins(C)|Cs],NCs) :- true |
    C=busy, response(R,Cs,NCs).
```

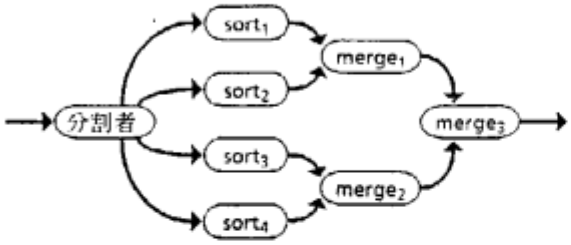
図3 プロセッサの状態を保持する述語

ンシェートされる。プロセッサが仕事をしている時に状態問合せコマンド(ins(C))が来ると述語responseによってbusyを返す。プログラムの詳細は、文献(4)に示した。

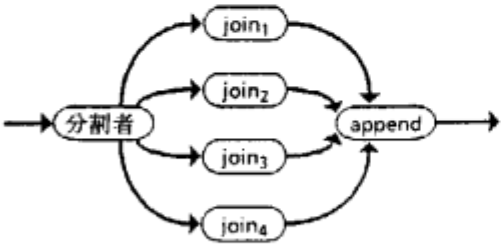
4. 処理粒度と並列性

戦略2、3では、データを分割することによって処理粒度を小さし、これによってプロセッサの空きを少なくし、並列性を高めようとしている。関係演算処理において、大きくウエイトを占めるものはソートとジョインである。以下ではソートとジョインを例にして述べる。

ソート・コマンドをn分割して処理する場合には、n個のソートと$\log_2 n$段n(またはn-1)個のマージが必要となる。つまり、ソート部分は処理粒度を小さくすればするほど処理は速くなるが、マージのコストが増加してくる。図4はソートとジョインを4分割処理した例であり、各プロセス間をGHCのストリームでつなぐことによってデータを流している。(a)は、$sort_1$と$sort_2$の結果を$merge_1$に、$sort_3$と$sort_4$の結果を$merge_2$に入力として与え、$merge_1$と$merge_2$の出力を$merge_3$の入力としてデータを流して行く。このように行え



(a)ソート・コマンド



(b)ジョイン・コマンド

図4 コマンドのデータ分割処理

```
make_join(Div,Op,Data,Result,C1-Cn) :-
  true |data_division(Div,Data,DD),
        div_join(DD,Op,Result-[],C1-Cn).

div_join([F|R],Op,R1-Rn,C1-Cn) :- true |
    C1=[cmd(join(Op,F,R1-R2))|C2],
    div_join(R,Op,R2-Rn,C2-Cn).
div_join([]   ,Op,R1-Rn,C1-Cn) :- true |
    R1=Rn, C1=Cn.
```

図5 差分リストによるjoinの分割

ば、3つのマージを並列に動かすことができ、マージの処理時間は分割数によって変化しなくなる。しかしながら、分割数を大きくするとマージ間の通信コストを無視できなくなってくる。なお、ソートに出現するマージは、整列された2つのストリームをデータの順序を考慮して併合するものである。

ジョイン・コマンドは、処理粒度を小さくする程並列に動かすことができ、処理時間が短縮される。そして、図5のように、分割するときに結果を差分リストで連結しておけばアペンドも必要なくなる。

5. おわりに

ここでは、GHCによる関係演算処理プロセスの並列制御の方式として3つ示した。この中で、データを分割することによって処理粒度を小さくする場合に、コマンドの種別によって並列性が異なることを示した。

ここでは述べなかったが、ホーン節変換[5]という知識管理層における知識コンパイル処理もGHCで記述することができた[4]。今後は、検索プロセッサもGHCで記述する予定である。

[謝辞] 本研究の機会を与えて下さったICOT研究所渕一博所長および有益な討論を頂いたICOT第1研究室、第3研究室の各氏に深謝いたします。

参考文献

(1)Ueda, K., "Guarded Horn Clauses", Logic Programming'85, Wada, E. (ed.), Lecture Notes in Computer Science 221, Springer-Verlag, pp. 168-179, 1986.

(2)宮崎、阿比留 他、KBMS PHI(5 情報処理学会第32回全国大会論文集、1985.

(3)Itoh, H., Abe, M. et al., "Parallel Control Techniques for Dedicated Relational Database Engines", Proc. of Data Engineering' 87, 1987.

(4)Itoh, H., Takewaki, T. et al., "A Deductive Database System Written in Guarded Horn Clauses", ICOT TR-214, 1986.

(5)Miyazaki, N., Yokota, H. et al., "Compiling Horn Clause Queries in Deductive Databases : A Horn Clause Transformation Approach", ICOT TR-183, 1986.