

PIEにおけるゴールフレームの
内部表現方式の検討と評価

2C-9

丸山 勉, 田中 英彦, 元岡 達
(東京大学 工学部)

1. はじめに

論理型言語を高並列に実行する高並列推論マシンにおいて、その並列基本処理単位の設定および基本処理単位を構成する構造データの共有制御方式の決定は、推論マシンの性能を決定する上で極めて重要な要素である[1]。これまでに幾つかの高並列推論マシンが提案されているがそれらに於てこれらの問題が十分に検討されているとは言えない。

高並列推論エンジンPIE[2]ではこれまで、処理単位のレベルとしては初期ゴールからの環境(不要となったものは縮退操作によって取り除かれる)、処理単位の表現形式としてはリテラル、ストラクチャ共にコピー、構造データの共有制御方式としては集中型構造メモリによるグラウンド・インスタンスのみの共有という方法を採用してきた。

本稿では、PIEに於けるこれまでの研究成果を基に、基本処理単位の設定について検討・評価を行なう。

2. 並列基本処理単位

基本処理単位を決定する上では、基本処理単位のレベル、スケルトンの共有/コピーの2つが重要なポイントとなる。

2.1 基本処理単位のレベル

高並列推論マシンに於ける基本処理単位のレベルとしては次の4つを考えることができる。

- ①引数
- ②リテラル
- ③定義節(のボディ部)
- ④初期ゴールからの環境

①はデータフォーマシンに適した処理方式である。以下②~④について簡単に説明する。

一般に②~④の順に従って処理単位長は大きくなり、コピー・転送等のオーバーヘッドが増加する。これに対して、④ではゴール間の共有は生じないが、②、③[図1]では親ゴールに対する子ゴール間での共有が生じる。従って、④では処理単位が完全に独立であるので自由な負荷分散を行なうことができるのに対し、②、③では、処理単位間の関係を考え

た処理単位の分散等の問題が生じる。

以上、述べたように処理単位のレベルの設定は基本的には、コピーと共有とのオーバーヘッドのトレードオフの問題である。

2.2 スケルトンの共有/コピー

逐次型推論マシンに於いては、ストラクチャのスケルトンの共有/コピーは微妙な問題であるが、高並列推論マシン(逐次型マシンの多数台結合は含まない)に於いては、ストラクチャのスケルトンの共有は、モレキュールを管理するメモリへのアクセスの集中等により、コピーに対して明らかに不利であると考えられる。従って、高並列推論マシンに於いては、リテラルのスケルトンの共有/コピーが主な問題となる。

3. PIEに於ける処理単位の構成

既に述べたようにPIEでは、基本処理単位のレベルとしては初期ゴールからの環境、表現形式としては、リテラル、ストラクチャのスケルトン共にコピーという方法を採用してきた。しかし、これまでの評価により、問題によってはゴール長が大きくなりすぎ、それによるオーバーヘッドが最悪の場合、コンパイル時間の十倍程度になることが解

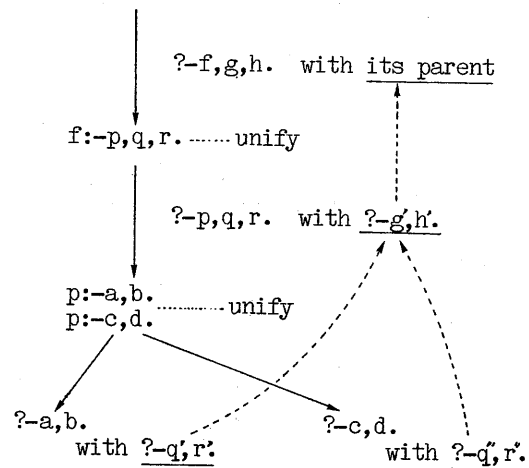


図1 定義節レベルに於ける実行の様子

った。従って、それらの場合にも十分に高速な処理を実現するためには基本処理単位の構成について再検討を行ない新たな処理方式を導入する必要があると考えられる。以下、PIEに於いて2.1で述べた②, ③(④はこれまでのPIEである)の処理レベルを導入した場合の処理手順とその利点・欠点について簡単に述べる。

(1) リテラル・レベル

この場合の処理手順は、ゴールからのサブゴール(リテラル)の切り分け、ユニフィケーション、新しいゴールの生成の順となる。サブゴールが単位節とユニフィケーションされた場合には、その結果がメモリ・モジュール中の親ゴールに送られる。親ゴールはメモリ・モジュール中でコピーされ、サブゴールの実行結果が、コピーされた親ゴールに結合される。親ゴール中にユニフィケーションすべきリテラルがある場合にはそのリテラルの切り分けが行なわれる。無い場合にはその親ゴールの実行結果は更にその親ゴールに対して送られる。

この方式では、他の方式に比べてゴール長を最も短くすることができるが、サブゴールの切り分けおよびサブゴールの実行結果の親ゴールへの結合の手間が問題となる。切り分けられたサブゴールに必要な最小限度のデータのみを親ゴールからコピーしようとしてもその手間は大きく親ゴール全てを単にコピーした場合と同等もしくはそれ以上の手間がかかりかねない。

(2) 定義節(のボディ部)レベル

この場合の処理手順は、ユニフィケーション、新しいゴールの生成の順となる。ゴール中のリテラルのユニフィケーションが全て終了した場合には、その結果がメモリ・モジュール中の親ゴールに送られる。親ゴールはコピーされ子ゴールの実行結果が結合される。親ゴール中にユニフィケーションすべきリテラルがある場合には、そのリテラルのユニフィケーションが行なわれる。

この方式では、(1)の場合よりゴール長が大きくなる可能性もあるが、そのコピーの手間は、(1)に於けるリテラルの切り分けの手間を考えるならば、大差はない。また、リテラルのスケルトンを共有した場合にはゴール長自体それ程変わらない(この方式ではリテラルのスケルトンを共有した方がゴール長の観点からは有利)と考えられる。

以上、各レベルについて簡単に述べた。(1)ではゴール長を短くすることはできるが、メモリ・モジュールに対する負荷の集中(リテラルの切り分け)

等の問題が考えられるため、現在(2)で述べた定義節レベルの処理方式について評価(シミュレーション)を行なっている。

これまでの試作単一化プロセッサ等の評価結果によれば[3]、ユニフィケーションの手間は約100ステップ(成功の場合)である。これに対して新しいゴールの生成の手間は、1セル当たり約2ステップ(試作単一化プロセッサでは5ステップかかるがパイプライン処理等によりこの程度にすることができる)程度である。従って、ユニフィケーションと他の処理(新しいゴールの生成、子ゴールの実行結果の取り込み)をパイプライン的に実行することを考えるならば、ゴール長は約40セル以下でなくてはならない。

構造メモリを用いた構造データの共有((2)でリテラルのスケルトンを共有した場合、ゴール中の変数番号の付け換えが不要なのでグラウンド・インスタンス以外の共有も可能)を行なえば、多くの場合ゴール長を40セル以下に抑えることができると考えられる。従って、基本処理単位間の共有に基づく負荷の集中を避けるために、ゴール長がその範囲内にある場合には、親ゴールを共有せずにコピーし、子ゴールと一緒に転送する方式がよいと考えられる。2.1④の方式に於いて、ゴール長がある程度以上になったらゴールの切り分けを行なう方法も考えられるが、その手間は比較的大きく、またパイプライン処理にも適さない。

4. おわりに

以上、高並列推論エンジンPIEに於ける基本処理単位に関する再検討を行なった。現在、3章で述べた処理方式についてシミュレーション評価を行っており、その結果に応じてPIEの処理方式の決定を行なう予定である。

<参考文献>

- [1] 平田, 丸山 他, “並列推論マシンにおける基本処理単位及び構造記憶に関する考察”, Logic Programming Conference '85, ICOT
- [2] Moto-oka, T., Tanaka, H. et al, “The Architecture of a Highly Parallel Inference Engine - PIE -”, FGCS '84, ICOT
- [3] Yuhara, T. et al, “A Unify Processor Pilot Machine for PIE”, Logic Programming Conference '84, ICOT