ＰＩＥにおける構造データ処理方式の効率化
6F-9　～　構造データ共有方式　～

平田　圭二，相田　仁，後藤　厚宏，田中　英彦，元岡　達

（東京大学　工学部）

１．はじめに

高並列推論エンジンＰＩＥ第１次モデルにおいて大きな構造データを処理する場合、処理時間構造データの大きさに比例して増加する。その要因としてネットワークの転送時間の増大、ゴールフレーム（ＧＦ）縮退の際のコピー時間の増加が挙げられる。我々は効率良く構造データを処理する方式として構造データ共有方式を提案し、その有効性を確認した［１，２］。ここでは構造データ共有方式を導入した場合のメモリモジュール（ＭＭ）に要求される機能、構成、処理アルゴリズムについて考察した。ＰＩＥの処理環境においてＭＭで行われる操作は次の４つに大別できる。

① ＭＭに格納されるＧＦ中の未定義変数を含まない構造データ部分（Ground Instance，ＧＩ）に対する印付け、
② ＧＩの切り分け、
③ 追加読み出し（ＬＦ）セルの送出、
④ ガーベジコレクション（ＧＣ）、

以下各操作について説明し、あわせて現在検討している方式について述べる。

２．ＧＩの印付けと切り分け

構造データ共有方式では単一化プロセッサ（ＵＰ）で生成されＭＭに送られてくる子ＧＦ中には、未定義変数を含まない構造データ（ＧＩ）が含まれている。このＧＩは、それを含むＧＦが再びＭＭより送出される時ＭＭ中に残され、孫ＧＦによって共有される。この処理機能を実現する為には、子ＧＦの持つ構造データを調べ、その中のＧＩ部分に対して“切り分け”の操作を施さなければならない。切り分けに先立ち“印付け”の操作を行っても良いし、印付けと切り分けを同時に行っても良い。これら２つの操作をＧＦがどの処理段階で施せば良いかを表１に示す。

印付けや切り分けの為だけに構造データを辿ることは行わないとすれば、これらの操作は構造データをコピーしたり辿ったりする時、副次的に実行できる。そのような時点として、

・ＵＰ内でＧＦに縮退操作を行う時、
・ＭＭに子ＧＦを書き込む時、
・ＭＭからＧＦを読み出す時、

がある。切り分けの操作は、印付けの操作と別々に行うことも、同時に行うことも可能である。例えばＵＰ内で印付けを行ったあとＭＭへ書き込む時に切り分けを行っても良いし、ＭＭ内で印付けとメモリへの書き込みを同時に行っても良い。

考えられる組合わせは５通りある。印付けと切り分けを異なるタイミングで行うと、印付けの結果を切り分け操作時まで保持する為、各セル毎に１ビットずつのフラグを必要とする。ＧＩの印付けと切り分けをＧＦのＭＭ書き込み時に行うと、ＧＩとそうでない構造データを別々のメモリに分けて格納することになる。この場合、フラグは不要であるがメモリは論理的に２つ必要である。次にＧＩの印付けと切り分けをＧＦがＭＭからＵＰへ送出される時に行う場合を考える。ＭＭ中ＧＦの構造データ部の内、切り分け操作を施されていない部分、即ちＵＰから送られてきたままのＧＦを区別する為にこのフラグが必要となる。現在実装中のシミュレータでは以上のことを考慮し、＊印のタイミングを選んだ。

表１．ＧＩの印付けと切り分けのタイミング

	ＵＰ内縮退時	ＭＭへ書き込む時	ＭＭから読み出す時
ＧＩの印付け	○	○	○ ＊
ＧＩの切り分け		○	○ ＊

○ … 実行可能

３．追加読み出し（Lazy Fetch）

これまでシミュレータ上で行っていたＬＦは、ポインタを一段たぐった先のデータのみをＭＭからＵＰに持って来るという操作であった。これはハードウェア寄りのかなり基本的な１つの操作と見做せる。しかしこの操作については今までのシミュレーション評価により、

・１回の転送データ量は高々数セルと少なく、
・その回数は非常に多い、
ということが確認されている。ここで言うセルとはＧＦを構成する最小単位を意味する、即ち、ポインタやシンボルは１セルで表現でき、リストのノードは２セルで表現できる。従って転送に要する時間の内、大部分がネットワークの経路設定の時間、メモリアクセス競合による待時間で占められてしまう。この問題を解決する為、単一化中に必要となるデータはできるだけまとめてＵＰへ転送する、即ちＬＦをあるレベルまでまとめて行うことを提案する。ここで言う１レベルとはポインタを１段たぐる深さのことを言う。どのレベルまでＬＦを行うかは表２の規則で決定される。

表２．　追加読み出し（ＬＦ）規則

ゴール側ポインタ ＼ 定義節側ポインタ	未定義変数を含む部分	未定義変数を含まない部分
ＵＰ内	単一化続行	単一化続行
ＭＭ内	①	②

表２において、ＵＰ内とは単一化中のポインタがＵＰ内の構造データを指す場合を、ＭＭ内とは単一化中のポインタがＭＭ内の構造データを指す場合を意味する。また、定義節中の構造データにも未定義変数を含む部分／含まない部分という印を付け、単一化中のポインタがどの部分を指しているかの区別を行う。①②は以下の操作を意味する。

①　定義節中の構造データにおいて変数の存在するレベルまでゴール側構造データの追加読み出しを行う。
②　ゴール側ポインタの指す構造データ全体に対して追加読み出しを行なう。

この規則は単一化中のポインタに対して常に適用されＬＦの実行をチェックする。

ＬＦは単一化操作の中断を引き起こすので、その回数は必要最小限に抑えなければならない。単一化の時に必要となって追加読み出しされるデータはＧＩの浅いレベルに位置していることが多い。従って単一化の対象となるリテラルに含まれるＧＩに対してのみ、あらかじめポインタをたぐり、根ノードから数レベル分のＧＩを最初から暫定読み出しＧＦに含ませてしまうことも考えている。これにより暫定読み出しＧＦのサイズは多少大きくなるが、全体として見ると単一化、縮退に要する処理時間は短縮するだろう。

４．ガーベジコレクション（ＧＣ）

我々は現在、構造データ共有方式を取り入れたＰＩＥの並列ＧＣアルゴリズムについて検討している。以下概略を述べる。

◇　ローカルＧＣは各メモリバンクに１台ずつ割り当てられ、バンク内参照の管理とセル領域の圧縮を行う。及び全体で１台の移動マーカを用意し、バンク間参照を管理する。つまり、ローカルＧＣと移動マーカで辿りと印付けを行い、圧縮はローカルＧＣが行う。このようにＧＣは移動マーカとローカルＧＣの２レベルからなる。ＵＰ－ＭＭペアは移動マーカからは論理的に環状に配置された構造に見え、常に一方向に回転しながらバンク間にまたがるポインタの管理を行う。
◇　移動マーカはポインタ管理の為、全論理アドレス空間分のビットテーブルを持ち、そのセルへのポインタが存在する時に対応するビットをオンにする。すべてのポインタは一方向で管理され、移動マーカもポインタと同じ方向に周回する。
◇　ＧＣの実行によるセルの移動に伴う他バンク中に存在する該当ポインタの書き換えは困難である。従って各バンクごとにアドレス変換テーブルを用意する。セルの参照は必ずこのテーブルを介して行う。

我々は現在このＧＣ方式をシミュレータに実装し、性能評価を行っている。

５．おわりに

ＰＩＥ第一次モデルに構造データ共有方式を取り入れた場合、ＭＭに要求される処理機能としてＧＦ中ＧＩの印付け、切り分け、ＬＦ方式、及びＧＣ方式について検討した。さらに構造データが複数バンクにまたがって存在する時のＬＦ方式も検討せねばならない。また、提案したＧＣ方式をより効率の良いものへと発展させたい。

＜＜　参考文献　＞＞

［１］平田　他：高並列推論エンジンＰＩＥにおけるゴールメモリ構成法に関する一考察，第２７回情処全大，４Ｐ－１３，1983.
［２］平田　他：高並列推論エンジンＰＩＥにおける構造データの効率的な処理方式について，信学技報ＥＣ８３－３８，1983.