ＰＩＥの構造データ共有方式におけるガーベジコレクション

6F-8

内山　明 ，　平田圭二 ，　田中英彦 ，　元岡　達
（　東京大学　　工学部　）

１．はじめに

構造データ共有方式［１，２］によるＰＩＥの構造データ処理環境はＬＩＳＰの場合とは大いに異なるため、従来から提案されているＬＩＳＰ処理系向きＧＣアルゴリズムは、そのままでは適用できない。我々はＰＩＥにおけるＧＣアルゴリズムについて考察し、今後の研究の為の基礎データをシミュレーションにより収集することにした。単一化プロセッサ（ＵＰ）、メモリーモジュール（ＭＭ）がそれぞれ一台という最も簡単なモデルに、変更を加えたＢａｋｅｒのアルゴリズムを実装した。本報告では、実装に際して加えた変更点を述べ、ＧＣを含めたマシンの動的な挙動に関する考察を行う。

２．ＧＣアルゴリズム

一般に並列マシンにおけるＧＣアルゴリズムに要求される条件として

・実時間性が保証されていること
・プロセッサのオーバーヘッドが少ないこと
・逐次、一括型ＧＣと比べてメモリ容量の増加が少ないこと

などがある。シミュレータで採用したアルゴリズムはＢａｋｅｒがＬＩＳＰマシン用に提案したものを基本としている。

２．１　参照カウンタ方式との比較

ＧＣアルゴリズムの候補の一つとして参照カウンタを用いる方式が考えられる。この方式ではリスト処理を行なう度に参照カウンタの上げ下げの操作をしなければならないので、メモリをバンク分けした場合バンク間での操作の送受が必要となる。このネットワークトラヒックの問題はデータフローマシンでも指摘されている［３］。さらに推論マシンの環境では、ＬＩＳＰの場合とは異なり構造データの参照数が非常に大きくなることが予想される。以上の考察より参照カウンタ方式は適当でないと判断し、Ｂａｋｅｒのアルゴリズムを採用した。

２．２　Ｂａｋｅｒのアルゴリズム［４］

このアルゴリズムではメモリ空間を２つの部分空間（新空間、旧空間）に分ける。セルの生成操作を新空間で行なう度に、ある定数ｋ個ずつセルを旧空間から新空間へ圧縮しながら移動させる。これよりＧＣの処理は分散して行なわれることになり実時間性が保証される。また参照操作を行う時、もしポインタが旧空間を指しているならそのセルを旧空間から新空間へ移動させる。セルの移動が完了した時点であらゆるポインタが新空間を指すようになると、次のサイクルが始まりもう一方の部分空間へ生きているセルの移動が行なわれる。

一般に、必要な記憶容量は（１＋１／ｋ）に比例し、セル生成操作に要する時間は　ｋ＋ｃ（ｃは適当な定数）に比例する。従って、記憶容量の制限が厳しくない限りｋの値は小さい方が良い。特に大きな記憶容量の使える場合、ｋを平均的に１以下にすることも考えられる。

この方式の利点としては、屑の目印づけと詰めかえの操作が同時に完了してしまうことが挙げられる。欠点としては、必要な記憶領域が逐次、一括型のＧＣ方式より大きいこと、各リスト処理に要する時間が多少長くなることが挙げられる。

２．３　シミュレータに実装する際の変更点

以下、変更点を列挙する。

①　ＬＩＳＰなどの記号処理システムの場合、構造データの根ノードの数は高々十数個であるが、ＰＩＥの場合は根ノードが並列処理される多数のゴールフレーム（ＧＦ）中に存在するのでその数は非常に大きなものとなりうる。従って、新空間と旧空間との入れ換え時に一度にＧＦに含まれるすべての根ノードを移動させると処理の長時間の中断が生じる。そこで次の２つの方法で根ノードを旧空間から新空間に移動させる。

・ＧＦがＵＰに送られる時、そのＧＦに含まれる根ノードは必ず旧空間から新空間へ移動する。
・ＵＰで処理が進んでいる間は、セルの生成をする度に定数個ずつＭＭ中にある根ノードを捜して移動させる。

②　ＰＩＥではＬＩＳＰのＣＯＮＳに相当する操作は、子ＧＦをＵＰからＭＭへ転送するときだけにしか生じない。したがってセルの移動が時間的に局在

する。セルの移動をなるべく分散させるために追加読み出しセルの参照（CAR，CDRに相当）の時にもセルの移動を行なえるようになった。
③　可変長セルを取り扱えるようにした。
④　kの値をユーザが任意に変化させることができ、様々な環境でシミュレーションデータが得られるようにした。

3．シミュレーション方針

GCのパラメータとして
・セル生成の時に移動させるセル数k
・セル参照の時に移動させるセル数k´
の2つをとった。以下の項目について測定を行なう。
・GCにかかる時間の処理時間全体に対する割合
・GC処理のかたより具合の検証
・必要な記憶容量の変化
これらの測定値からk，k´の最適値についての考察を行なう。

4．シミュレータの構成

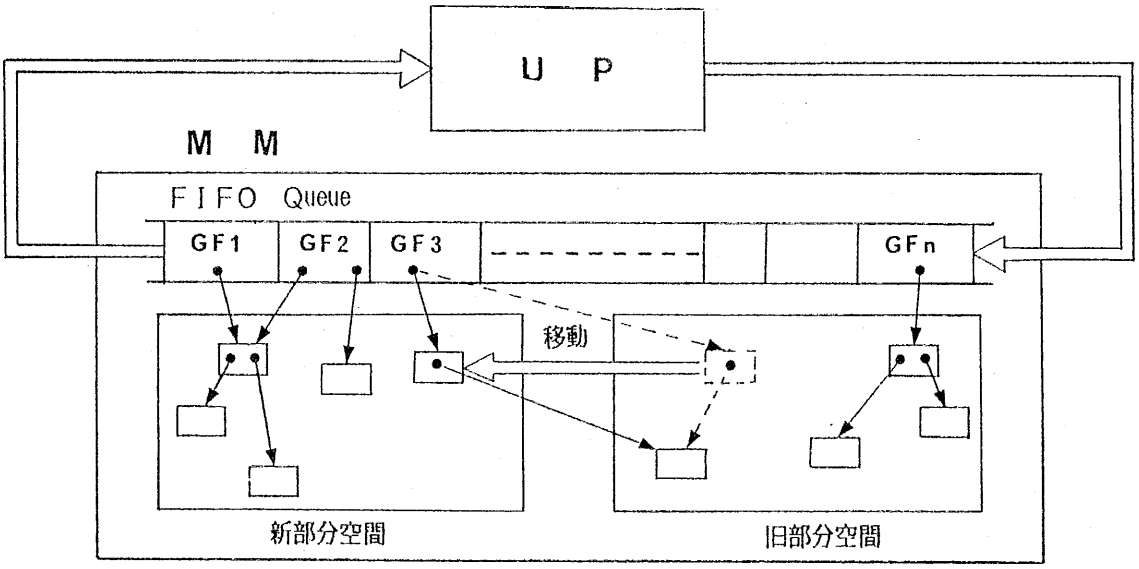
シミュレータの構成図を示す（図1）。このシミュレータではUPとMMは、それぞれ一台である。UPで親GFの単一化を行ない、生成された子GFはMM中のFIFOキューに格納される。このシミュレータで採用した構造データ共有方式では、未定義変数を含まない構造データ部分をGF間で共有することにより処理の効率化をはかっている。この共有される構造データを格納するメモリへのポインタ即ち根ノードはGF中にのみ存在する。従って、根ノードの探索とはFIFOキュー中の構造データ格納メモリへのポインタを探索することに等しい。また単一化に際し、メモリ中の構造データが必要となった場合は、GFからポインタをたぐりデータを読み出さなくてはならない。

5．おわりに

今後、次のような点を検討しなければならない。
①　処理の状態に応じてkの値を動的に変化させればメモリ効率はさらに向上することが予想される。よってkの値の制御方法についての検討が必要である。
②　PIEでは根ノードの移動がGCの処理全体に占める割合は大きい。従って根ノードの移動法をさらに工夫せねばならない。

《　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方に関する一考察”，第27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”，信学技法　EC83－38，1983．
［3］中村、長谷川、雨宮；”リスト処理向きデータフローマシン用構造体メモリの設計と評価”、芯学技法　EC81－32．
［4］Henry G. Baker, Jr.,”List Processing in Real Time on a Serial Computer”, CACM 21. 4（April 1978）



（図1．）シミュレータの構成