3C-8

GRACEプロトタイプシステムにおける記憶管理プロセスの構成

中山 雅哉† 伏見 信也† 喜連川 優†† 田中 英彦† 元岡 達†

(†東京大学 工学部 ††東京大学 生産技術研究所)

1. はじめに

我々が研究，開発を進めているGRACEにおいて，メモリモジュール（MM）は，Join，Aggregation 等の負荷の重い関係代数演算をプロセシングモジュール（PM）で高速に効率良く処理する為に，Hashを基にした記憶管理と，プロセシングクラスタ生成の為のスケジューリングを行っている。特にMMでの記憶管理は，通常のデータベースシステムにおけるバッファ管理とは異なり，Hash操作に基づいたバケットを単位として行っており，従来とは異なる方式により，記憶の仮想化を計っている。

今回は，MMにおける記憶管理の方式と，試作システム上での実装方式について報告する。

2. GRACEにおけるメモリモジュール

GRACEは，コントロールモジュール（CM），ディスクモジュール（DM），MM，PMと，それらを結合する2つの網から構成されている〔1〕。PMにおいてJoin, Aggregation 等の関係代数演算は，HashとSortに基づいて実行され，O（n）時間での演算処理を実現している。MMでは，DMで Selection,Projection 演算処理を施したデータ流に対して，PMでの処理が効率良く実行されるように，タプル毎にハッシュを施し，複数のバケットからなる空間に分割して管理を行っている。

3. 記憶管理の二方式

MM空間を複数のバケットに分割し管理を行う為には，

（a）静的にバケット空間を割り当て，各バケット空間内で，記憶の仮想化を行う方式.

（b）各バケット空間には，動的に必要な領域を割り当て，MM空間全体として，記憶の仮想化を行う方式.

が考えられる。

前者は，Hash関数が適当で，各バケットのタプル数が均等となる場合はよいが，各バケットに対するタプル数に変動がある場合（Hash関数に偏りがある場合）には，メモリ資源の利用効率が低くなるという欠点を有する。また，GRACEでは，主記憶の仮想化において，DMからのデータ流のステージングが終了した時点で，PMの処理が即時に開始できるように，使用するタプル群（プロセシングクラスタ）を主記憶中に置く様にスケジューリングを行う事が望ましいが，静的に割り当てられたバケット空間を各々で管理するこの方式では，要求されるスケジューリングをうまく実現できない。

それに対し後者は，バケット空間に必要な領域を動的に割り当てる為，Hash関数の偏りに対しても，柔軟に対応することができる。また，MM空間全体で仮想管理を行っており，要求に応じたスケジューリングを柔軟に行うことが可能となる。

この様な理由から，試作システムにおいては，後者の方式を用いて実装を行うことにした。

4. 記憶空間の仮想化方式

試作システムにおけるMM空間は，複数のブロックの集りとして管理され，先に述べたバケット空間に対する動的な領域の割り当ては，このブロックを単位として行われる。また，我々のシステムでは，前述のように，特殊なスケジューリングを行いながら，データ流のステージングを行っている。通常のOS等では，LRU法，MRU法等を用いることにより，主記憶の仮想化を実現しているものが多いが，これら従来からのリプレースメントアルゴリズムは，本システムのスケジューリングには適当でない。

そこで我々は，MM空間中に空ブロックが無くなった時に，以下に示すような方法を用いて空き領域を生成し，主記憶の仮想化を実現している。

（i）既に追い出されているブロックを有するバケットで，データが1ブロック以上に渡って主記憶上に存在していれば，そのバケットの各ブロックを追い出す。

（ii）そうでなければ，現時点で，最も多くのブロックを主記憶上に有するバケットの各ブロックを追い出す。

この方法を用いると，データ流のステージングが終了した時点の主記憶中には，小さなバケット空間が複数個存在し，大きなバケット空間は，そのほとんどが，作業用のディスク内に格納されている状況となる。

作業用ディスク内に格納されているブロックは，PMが小さなバケット空間を処理している間に再ステージングされる。これにより，PMで処理するデータは，いつも主記憶中に存在する様にスケジュールすることができる。また，PMに収容できない程大きなバケットに関しては，再度ハッシュを施して処理可能な大きさにすることができる。

5. 記憶管理プロセスの実装

我々は，MELCOM80／500 システム上に，ソフトウェアによるGRACE試作システムの実装を進めている〔2，3〕（図1）。

本システムでは，M－プロセスとWD－プロセスにより，GRACEのMMにおける機能を実現している。各プロセスの記憶管理に関する資源は，図2に示す通りである。

実際のMM空間は，ブロックに分割されて，両プロセスに共通な領域上に採られる。これは，各プロセス間で通信を行う際の，オーバーヘッドを軽減する為である。また，空ブロックに関しては，2つのプロセスからのアクセスが必要となる（M－プロセスからの取り出し，WD－プロセスからの追加）為，両プロセスに共通な領域上のQueue として管理している。

M－プロセスは，主記憶内におけるバケットの管理と，追い出しブロックの決定を行う。BMT－M（Bucket Management Table for Memory）は，各バケットの先頭ブロック，最終ブロック，トータルのブロック数，最終ブロック中のタプル数，スワップアウトフラグを保持するものであり，これを用いて，追い出しブロックが決定される。また，M－LT（M Link Table）は，M空間中の各ブロックのアドレスと，次ブロックへのポインタを保持しており，これら2つのテーブルを用いて，主記憶中のバケットの管理を行っている。

WD－プロセスは，主記憶から追い出されたブロックの作業用ディスク（WD）内におけるバケットの管理を行っている。BMT－WD（Bucket Management Table for Working Disk）は，各バケットのWD内の先頭ブロックと最終ブロックを保持している。WD－LT（WD Link Table）は，WD内での各ブロックのアドレスと，次ブロックへのポインタを保持しており，これら2つのテーブルを用いて，WD内のバケットの管理を行っている。

プロセス間の通信には，M空間中のブロックアドレスと，バケットIDだけが用いられ，データ転送量は，十数バイトで済む為，そのオーバーヘッドは小さくてすむ。また，WD－プロセス内で使用しているI／Oドライバは，OSを経由する通常のI／Oとは異なり，任意の主記憶空間とディスク間でのデータ転送を行うことができる。これを用いることにより，OS内で管理する入出力バッファ領域とユーザ空間とのデータ転送を避けることができ，処理のオーバーヘッドは軽減される。しかし，ここで用いる特殊なI／Oの制約から，本システムでは2kB／ブロックとして，主記憶の分割管理を行っている。

6．おわりに

GRACEにおける記憶管理の方式を示し，記憶の仮想化方式として，従来からOSのバッファ管理等で用いられているリプレースメントアルゴリズムとは異なる方式について説明した。また，この方式を試作システム上の記憶管理プロセスとして実装した際の構成について示した。現在，試作システムの実装はほぼ終えている。そのシステム全体についての構成，性能評価については，機会をかえて報告するつもりである。

参考文献

〔1〕伏見他，『データベースマシンGRACEのアーキテクチャとその実行制御系』，アドバンスト・データベースシンポジウム（1984）

〔2〕伏見他，『データベースマシンGRACEのプロトタイプシステム』，情報処理学会第31回全国大会，1B-6（1985）

〔3〕中山他，『GRACEプロトタイプシステムにおけるソフトウェア構成』，情報処理学会第31回全国大会，1B-7（1985）

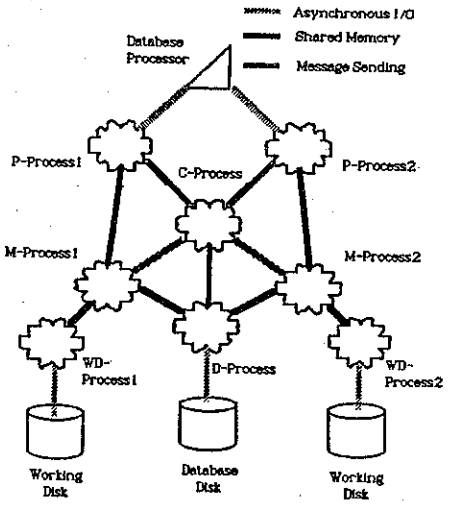

図1．試作システムの全体構成

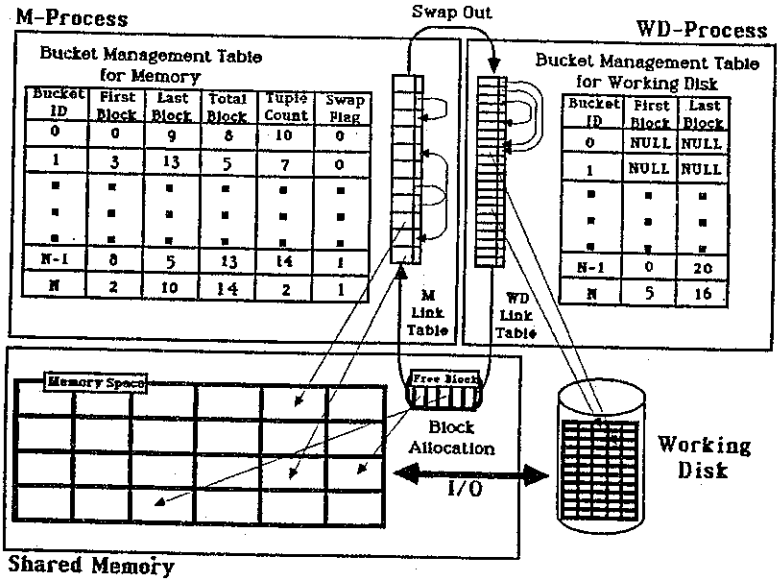

図2．記憶管理プロセスの実装方式