データベースプロセッサＲＩＮＤＡの関係演算方式

5Q-6

武田英昭、　佐藤哲司、　中村敏夫、　福岡秀樹
ＮＴＴ情報通信処理研究所

1　はじめに

関係演算処理において、選択演算、射影演算はO(n)（O：オーダ、n：データ数）の時間で処理が可能である一方、結合演算はO(nlogn)と負荷の重い処理である。データベースプロセッサＲＩＮＤＡ[1]では、この結合演算を高速化するため、専用のハードウェア（関係演算プロセッサ：ＲＯＰ）を備えている。本稿では、ＲＩＮＤＡにおける結合演算を中心とした関係演算処理方式についてについて述べる。

2　ＲＯＰの位置付けと構成

ＲＩＮＤＡでは、結合演算処理をネステッドループ方式、あるいは、ソートマージ方式で実現している。前者はデータ量が少ない場合であり、本体装置のみで処理する一方、後者は結合データ量が多い場合であり、本体装置とＲＯＰにより実現している[2]。ＲＯＰの構成を図１に示す。ＲＯＰと本体装置間は、ページ単位でデータ転送を行っている。

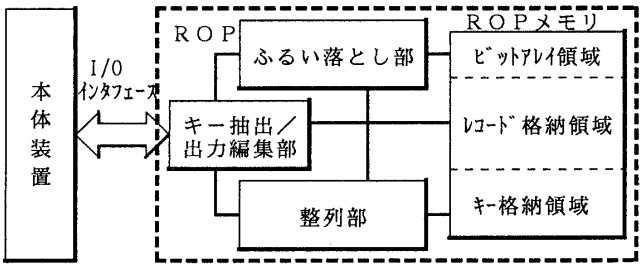

図１　ＲＯＰの構成

ソートマージ結合演算の基本的な処理法は次の通りである。

① 一方の表を本体装置からＲＯＰに転送し、ＲＯＰにおいて結合キー（以下、キー）の抽出、ソートを行う。
② ソート後、ソート結果の表を本体装置に転送する。
③ 他方の表を本体装置からＲＯＰに転送し、ＲＯＰにおいてキーの抽出、ふるい落とし、残ったキーのソートを行う。
④ ②と同様。
⑤ 本体装置で、ソート済みの二つの表をマージ結合する。

3　ＲＯＰの機能

3.1　機能概要

ＲＯＰは以下のような機能を持つ。

① キー抽出機能：本体装置からの転送に追従し、行からキー（複数列も可）の抽出、列ごとに各種属性から絶対値整数形式への変換およびキーの固定長化を行う。
② ふるい落とし機能：結合する前に、一方の表あるいは両表から予め結合可能性のない行を除去する。
③ ソート機能：キーを昇順（降順）にソートする。
④ 重複除去機能：キーの値が重複する行を除去する。
⑤ 重複計数機能：キーの値が重複する行数を重複ごとに計数する。
⑥ 出力編集機能：処理結果得られたキー、計数値を元の行形式に編集する。

以下に、上記の機能のうち、特に、ふるい落とし機能、ソート機能について述べる。

3.2　ふるい落とし機能とその実現方法

(1)　ふるい落とし手法

ＲＯＰにおけるふるい落しには、一方の表のみをふるい落とす片ハッシュ法と両方の表をふるい落とす両ハッシュ法とがある。いずれの方法においても、その実現にハッシュ化ビットアレイ[3]を用いている。

片ハッシュ法は、１つのビットアレイを用い、以下の手順でふるい落とし処理を行う。

① 表Ｔ１の各キーをハッシングし、ハッシュ値に対応するビットアレイ位置に'１'を設定。［設定処理］
② 表Ｔ２の各キーを①と同じハッシュ関数でハッシングし、ハッシュ値に対応するビットアレイ位置を参照し、そこが'１'のキーのみ出力の対象とする。［参照処理］

両ハッシュ法では、２つのビットアレイを用い、上記②処理時に同時に他方のビットアレイに設定処理を行い、その後、③として表Ｔ１の参照処理を行う。

ふるい落としにより、ソートキー数すなわち結合行数を削減でき、結合時間の短縮を図ることができる。

(2)　ハッシュ関数

筆者らは従来、キー長の変化に強いハッシュ関数として回転重ね合わせ法[4]を提案していたが、この方式ではキー値の偏りによる衝突の多発が問題であった。ＲＯＰではこれを

RINDA －Relational Database Processor : Hardware Architecture for Join Operations
Hideaki TAKEDA, Tetsuji SATOH, Toshio NAKAMURA, Hideki FUKUOKA
NTT Conmmunications and Information Processing Laboratories

解決する乗算重ね合わせ法を採用した。この方式では、キーを一定長の切片に区切り、キーの入力に同期し、先頭の切片から順に、① 乗算表により切片の値の変換を行い、② それまでのハッシュ結果を一定ビットシフトし、それに①の結果を重ね合わせる(ＸＯＲをとる)、処理を行う。

本方式は、従来方式に①を付加したため、切片内で一度プレハッシュされ、キー値の偏りにも強い方式となっている。

3.2 ソート機能とその実現方式

ハードウェアによるソート方法としては、従来から種々の方式が提案されている。しかし、いずれの方式もキー比較器の数がソートするキー数に依存するものであった。これに対し、ＲＯＰでは10万件以上という多量のキーをソートできるようにする一方、装置を小型化するため、多段マージによるソート法[5]を採用した。

(1) 多段マージによるソート法

本ソート手法は、⌈ｋ/２⌉個（ｋは自然数）の比較器を一次元アレイ上に配置したソートアレイとＲＯＰメモリを用い、ｋウェイマージ処理を段階的に繰り返すことによりソートを行うものである。図２に本ソート回路の構成を示す。

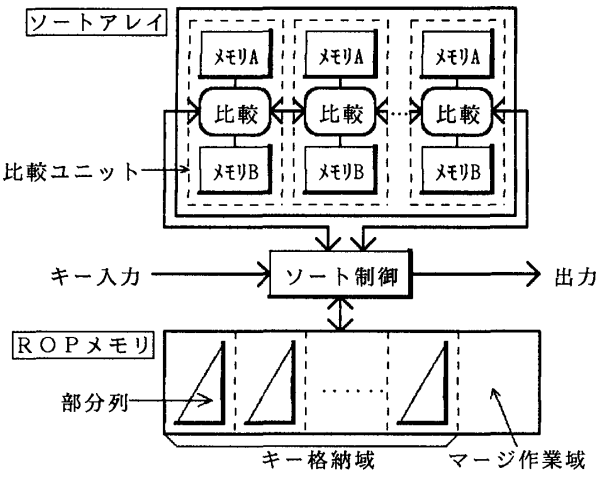

図２ 整列回路の構成

ソート手順は、以下の２種類の段階からなり、①を処理後、マージ結果が１本になるまで②を繰り返す。以降、①を１段目のマージ、以下、②を順に２段目、３段目、…と呼ぶ。

① 初期ソート段階：キーを入力しながら、ソートアレイを用いて、ｋ個ずつソートし(これを部分列と呼ぶ)、ＲＯＰメモリに格納する。

② マージ段階：部分列をｋ本ごとにマージする段階であり、ⓐを実行後、読み出し部分列がなくなるまでⓑを繰り返す。

ⓐ ＲＯＰメモリに格納されている各部分列の先頭キー(部分列の最小キー)を読み出し、ソートアレイに入力。

ⓑ ソートアレイから最小キーを出力し、ＲＯＰメモリに書き込む。次に、そのキーが属していた部分列の次のキーをＲＯＰメモリから読み出し、ソートアレイに入力。

多段マージによるソートを行うことにより、一定のソートアレイで多量のキーのソートが可能となる。一方、多段マージに伴う処理時間の増加に対しては、マージウェイ数（ｋ）を多くし、マージ段数を削減することで解決を図っている。

(2) マージ作業域の削減

本ソート法において解決すべき問題の一つにマージ作業域の削減がある。マージ作業域は、キーの個数によっては最悪キー格納域と同じ大きさだけ必要であり、キー数が多い場合には特に問題となる。これを解決するにはマージ段のいずれかにおいてマージする部分列の調整を行う。

図３に１段目のソート後に調整［初段最適化］、(最終－１)段目のマージ前に調整［終段最適化］した場合のキー数とメモリ使用量との関係（ｋ＝４の場合）を示す。

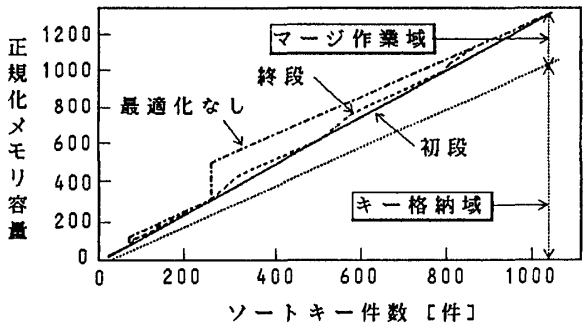

図３ 多段マージソートのメモリ使用量

初段最適化と終段最適化とでは前者の方がメモリ削減効果は大きいものの、マージ本数の切り替えがｋ２回（前者は１回）と制御が複雑となる。ＲＯＰでは終段最適化を採用した。

4 おわりに

ＲＩＮＤＡでは、専用のハードウェアによって結合可能性のないデータのふるい落とし、多段マージによる大容量データのソートを行っている。これにより、関係演算処理のうち、特に処理負荷の重い結合演算処理の高速化を実現した。

［参考文献］

〔１〕 井上他、「データベースプロセッサＲＩＮＤＡのアーキテクチャ」、情処第37回全国大会、1988

〔２〕 中村他、「データベースプロセッサＲＩＮＤＡの最適化処理方式」、情処第37回全国大会、1988

〔３〕 C.D.McGregor, et al. "High Performance Hardware for Database Systems",Syst.for Large Data Bases,1976

〔４〕 武田他、「関係演算処理のための専用ハードウェアの構成とその評価」、情処ﾃﾞｰﾀﾍﾞｰｽｼｽﾃﾑ 57-5、1987

〔５〕 佐藤、武田、「大容量データベース処理に適したソート手法」、信学技法ＤＥ、1988