ソート及びハッシュジョインの並列処理

7T-7

野口 泰生 , 武 理一郎

㈱富士通研究所

1. はじめに

我々は，関係データベース処理をマルチプロセッサ上で高速に実行することを考えている．しかし，ソートやジョイン等は，通信負荷が高いため台数効果を出し難い処理である．多くの場合，通信負荷を減らすためには何らかのクラスタリング技法が用いられ，大量の通信を行うプロセス同士は近いプロセッサに割り振られる．しかし，プロセスのクラスタリングは通信のクラスタリングを招き，特定のリンクが輻輳し他のリンクは遊んでいるということになりがちである．

我々は，通信オーバヘッドをデクラスタリングによって抑えようと考えている．通信の負荷を均等にネットワーク内に分散し遊んでいるリンクをなくすことで，通信資源待ちの時間を減らすことができる．又，予め，ネットワーク全体に渡って一様な通信が起きることが分かっていれば，プリスケジュールによりアービトレーションの時間も減らすことができる．既に，我々は，この考え方に沿ったネットワークとしてドラゴンネットと呼ぶ超立方体形ネットワークを提案した．[1)]

我々は，ソート及びハッシュジョインの並列処理にドラゴンネットを適用し良好な結果を得たので報告する．

2. ドラゴンネット

ドラゴンネット[1)]とは，超立方体形ネットワーク上での全点対全点の通信を，プリスケジュールにより輻輳なく処理するネットワークである．

例えば，２次元の超立方体でのスケジューリングは下図の様になる．フェーズｓでは各点ｖからｖ⊕ｓへ（ｖが偶の点の時）或いはｖ⊕ｓ^{c}へ（ｖが奇の点の時）のメッセージが送信される．この様なフェーズを２^{d}回繰り返すことで全点対全点通信が処理される．各メッセージが通るパスは，宛先と現在地の差ビットを下位から解消していくという普通のやり方で決定される．

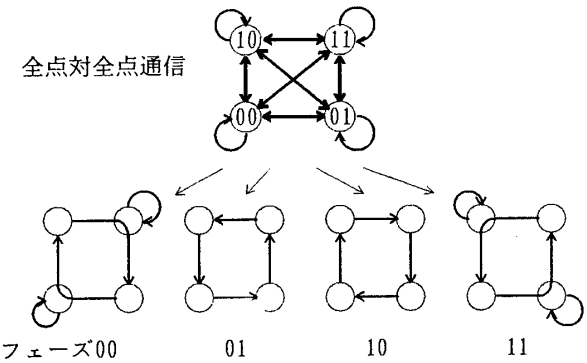

図-１ 全点対全点通信の分割と超立方体へのマッピング

Sorting and Hashjoin on Hypercube Networks

Yasuo Noguchi and Riichiro Take
Fujitsu Laboratories Ltd.

一般に，ｄ次元の超立方体での全点対全点通信が処理されるには，各リンクが少なくとも２^{d}回（即ち各点が持つメッセージの数）働かねばならない．ドラゴンネットに於いてはこの最適値が実現されている．

全点対全点通信が上手く処理できると，これと対照的な通信パターンである点対点ストリーム通信も上手く処理することができる．点対点のストリームを２つの全点対全点通信に分解することにより，Benes 網に於ける様なダイナミックなルーティングをせずに，輻輳のない置換網が実現できる．

3. ドラゴンネット上でのデータベース処理方式

3.1 ハッシュジョイン

リレーションは各プロセッサに均等に水平分散されていると仮定する．ハッシュジョインに於いては，先ず，２つのリレーションをプロセッサ間でハッシュし，その後，各プロセッサ内でジョインを行う．マルチプロセッサハッシュはまさに全点対全点通信であり，ドラゴンネットにより効率良く処理することができる．

ハッシュジョインについては，これまで主に，ハッシュバケットのあふれの問題が研究されて来た．マルチプロセッサ上でのハッシュジョインについても，通信処理の時間は，入出力処理やプロセッサ内処理に隠れてしまい重大な問題となるには至っていない．[2)] しかし，我々は，プロセッサの数が増え，データベースの主記憶化が進むと，通信コストが処理全体の中に占める割合は大きくなると考えている．

3.2 ソート

マルチプロセッサ上でのデータベース処理向きソートとして，我々は既に，並列分割ソートと呼ぶ方式を提案した．[3)] このアルゴリズムに於いては，先ず，各プロセッサがソート後に持つべきキー値の範囲を調べ（分割フェーズ），次に,各プロセッサがその受持のキー範囲に含むまれるキーを持つ様にデータの再配置を行い（再配置フェーズ），最後に,内部ソートを行う（内部ソートフェーズ）．バイトニックソートがマージソートなどの様な easy-split hard-join に対応するとすれば，この方式は，クイックソートなどの様な hard-split easy-join に対応するものである．

再配置フェーズは，キー値の範囲とプロセッサとを対応付ける一種のマルチプロセッサハッシュであり，全点対全点通信を引き起こす．従って，ドラゴンネットが有効である．

4. 実験と評価

4.1 環境

実際にマルチプロセッサ上で，ドラゴンネットを実現し，ハッシュジョインとソートの処理について性能を評価した．使用したマルチプロセッサシステムは，富士通研究所で開発された CAP-256である．[4)]

CAP-256 は，セルと呼ばれるプロセッサエレメントから成る．各セルは８MHzのi80186と２Mバイトのメモリを持ち，６本のリンクを通じて他のセルと通信を行う．

トーラス構成をとる時には 256台まで組み合わせることができるが，超立方体構成をとる時には最大64台（6次元）までとなる．

CAP-256 のセルはバイパスと呼ばれる機能を持ち．セル内に持つプログラマブルなスイッチボックスにより，リンクの組を電気的に直結することができる．この機能により，目的地につくまでに複数のセルを通過するメッセージの処理を高速に行うことができる．全点対全点通信の処理について，store-and-forward に処理すると，12Kバイト/s 程度の通信性能であるが，ドラゴンネットとバイパス機能を組み合わせると，約 130Kバイト/s の性能を出すことができる．

4.2 ハッシュジョイン

リレーションは各セル上に水平分散され，メモリ上の配列として表現されている．ここでは I/Oコストについては考えていない．配列の巾は64バイトであり，キーは2バイトの整数である．2つのリレーションはそれぞれ，1Kタプルと10Kタプルのサイズを持つ．

処理は以下の手順で行われる．
① 各セル内で，2つのリレーションのマルチプロセッサハッシュ用ハッシュバケットを作成する．
② セル間で，マルチプロセッサハッシュを行う．
③ 各セル内でハッシュジョインを行う．

マルチプロセッサハッシュに於ける通信をstore-and-forward 風に行った場合と，ドラゴンネットで行った場合とを比較したグラフを下に示す．ドラゴンネットにより，マルチプロセッサハッシュが約16倍に高速化され，処理全体でも，約9倍の高速化が達成されている．

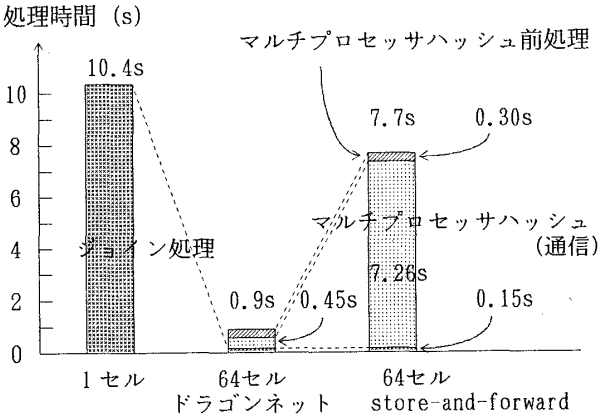

図-2 ハッシュジョインの処理時間の比較

4.3 ソート

前述の並列分割ソートと，従来から高速な並列ソートとして知られているブロックバイトニックソートの比較を行った．ここでもリレーションは各セルに水平分散され、メモリ上の配列として表現されている．タプルの巾は64バイト，キーは 2バイトの整数である．リレーションの大きさは 1Kタプル，4Kタプル，16Kタプルについて実験した．セルは 4台，16台，64台を使用した．

並列分割ソートの分割値の発見に関して我々が提案したアルゴリズム[3]には，セル台数にほぼ比例した時間を要するという欠陥があった．このため本実験システム上ではセルが多くなると，この方法で台数分の分割値を求めるよりも，タプル全体は無視してキーだけのブロックバイトニックソートを行い，分割値として必要なキーを用いる方が処理時間が短くなった．

従って，並列分割ソートは以下の手順で行った．
① 各セル内をクイックソートする．
② キーをブロックバイトニックソートし，分割値を発見する．
③ ドラゴンネットを用いてタプルを再配置する．
④ 再度各セル内をクイックソートする．

全ての場合に並列分割ソートがブロックバイトニックソートより高速であった．(図-3a, 3b) 以前の報告[3]では一部両者の処理時間に逆転があった．今回の結果はドラゴンネットによる全点対全点通信処理の改善を示すものである．並列分割ソートが高速なのはキーだけの移動時間がタプル全体の移動時間より遙かに小さいためである．

並列分割ソートでリレーションの大きさが 1Kタプルの場合，セル64台と16台で処理時間の逆転があった．(図-3b) ドラゴンネットでは，一定の通信量に対し最適なセル台数が定まっている．これは通信時間がセル台数にほぼ反比例するのに対し通信のためのパケット生成などに要する時間はセル台数に比例するためである．

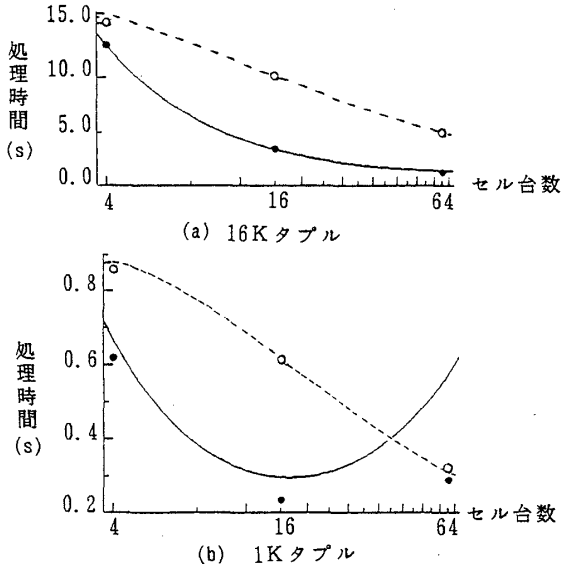

図-3 処理時間とセル台数の関係

5. まとめ

超立方体形ネットワーク上でのハッシュジョインとソートの処理について，その方式の説明と性能の評価を行った．超立方体上での全点対全点通信を輻輳なく処理するネットワーク：ドラゴンネットにより，これらの処理を高速化できることが示された．

参考文献

1) 武，超立方体形ネットワークに於ける全点対全点通信の最適ルーティング法，情処第35回全国大会，1987
2) D.J.DeWitt,et al., A Single User Evaluation of the Gamma Database Machine, proc. IWDM, 1987
3) Y.Yamane and R.Take, Parallel Partition Sort for Database Machines, proc. IWDM, 1987
4) 石畑 他，セルラアレイプロセッサ組織化チップのアーキテクチャ，信学会ＣＡＳ86-210，1986