疎結合並列計算機における多重結合演算の評価

1F－6

赤星直輝　原田リリアン　野口泰生
（株）富士通研究所

1. はじめに

並列データベース研究に関する研究が多くおこなわれており、その多くは結合演算の並列処理による高速化に関するものである。さらに、より重い演算である多重結合演算[1、2]についての研究がおこなわれている。我々も、多重結合演算の効率的並列処理手法に関する研究をおこなっており[3、4]、今回は効率的な演算パイプライン決定の詳細およびその評価について述べる。

2. 多重結合演算

本稿では、演算を実行する並列計算機環境として、疎結合並列計算機を対象とする。さらに、演算の対象となるリレーションは、全ての処理装置に均等に格納されていると仮定する。

多重結合演算の演算パイプラインの構成方法としては、図1に示すようにレフト木、ライト木[1]、Segmented-ライト木が提案されている[3]。木は、左側の葉にあたるリレーションでハッシュテーブルを作り、右側の葉にあたるリレーションをつきあわせることを意味する。しかし、これらの手法では、並列計算機の資源によっては、常に効率のよい方法を提供できないという問題点がある。すなわち、演算のパイプラインにおいて、I/Oの転送速度、ネットワークの転送速度、CPUの処理能力がバランスしていなければ、遅い部分が律速段階となってしまう。例えば、メモリに沢山のハッシュテーブルを載せて長いパイプラインを生成すると、読み出しと書き込みのI/Oの量は変わらなくても、長いパイプライン中の通信量が大きくなる。この通信を処理できるだけのネットワークの転送速度がない場合には、処理効率がかえって低下することになる。

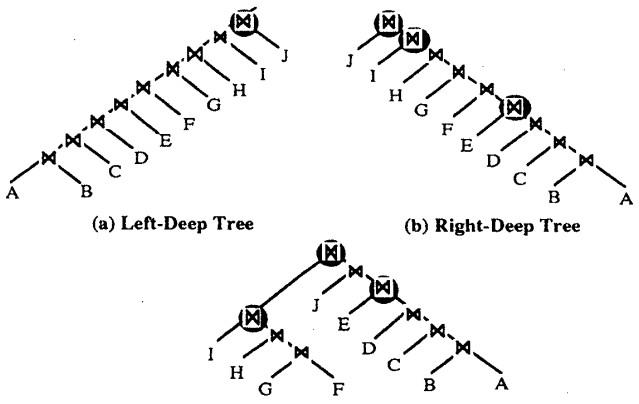


図1　レフト木、ライト木、Segmentedライト木

3. 計算機資源を考慮に入れたパイプライン

我々は既に資源を考慮したパイプライン処理方法の提案を行っており、今回は演算パイプラインの決定方法の詳細および評価について報告する。

多重結合演算を実行する場合には、演算の中間結果をファイルに書き出す場合と、中間結果を主記憶に展開して次の段の演算に利用する場合との２通りを取りうる。我々は、前者をWR-SYNC (Write-Read synchronization)と呼び、後者をBP-SYNC (Build-Probe synchronization)と呼ぶことにする（図2）。すべての演算実行プランは、この２種類のパイプラインで表すことができる。従来、演算パイプラインの長さを決めるのに、メモリ上にハッシュテーブルがどれだけ載るか（メモリの制約）が利用されてきたが、先にも述べたように、I/Oやネットワークの転送速度といった処理のバランス（資源の制約）によって最適な演算パイプラインの長さは変化する。このため、我々はパイプラインの長さの決定に関して、資源の制約を考慮に入れる。

以上を踏まえた上での、我々の演算パイプラインの決定方法は、メモリとハッシュテーブルのサイズや、I/Oや、ネットワークの転送速度、CPUの処理速度といった資源の制約を考慮して以下の3通りを検討するものである。

1)WR-syncで、メモリの制約、資源の制約を満たす
2)BP-syncで、メモリの制約、資源の制約を満たす
3)BP-syncで、メモリの制約を満たした上で、資源の制約を出来るだけ満たすようにする。

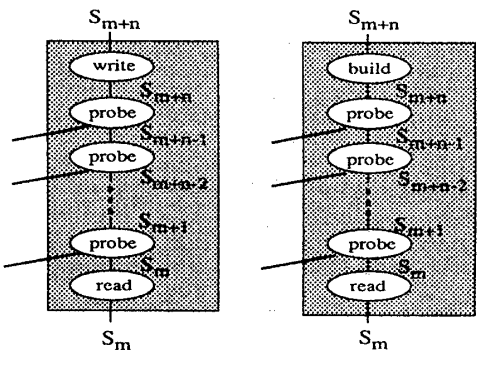


図2　2種類のパイプライン

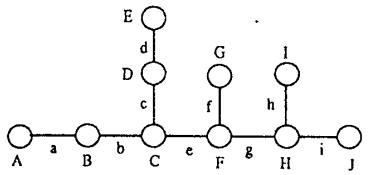
Evaluation Results of Multi-Way Joins in Shared-Nothing Environments
Naoki Akaboshi, Lilian Harada and Yasuo Noguchi
FUJITSU Laboratories Ltd.
1015, Kamikodanaka nakahara-ku, Kawasaki 211, Japan

我々の提案する方法では、はじめに中間結果をディスクに書き出しても処理をバランスできる(1)の木を生成するように試みる。しかし、I/O処理がボトルネックとなって処理がバランスできず、(1)の木が生成しにくい場合には、中間結果をメモリに残してI/Oを減らす(2)の木を生成しようとする。さらに、計算機の資源によっては(1)の木も(2)の木も生成できない場合があるので、その場合には、I/Oやネットワーク、CPUといった処理のバランスよりもメモリを有効に利用するような(3)の木を生成する。

4. 評価

多重結合演算に対して、我々のアルゴリズムを適用することによって、どの程度の効果が得られるかシミュレーションをおこなった。プロセッサは16台で、タプルサイズを208バイトとする。ディスク1台当たりの転送速度は5MB/sで、ネットワークのバンド幅は10MB/sである。与えられた多重結合演算の問い合わせを図3に示す。図3(a)において、線分で結ばれたリレーションは、その属性と結合率で結合演算がおこなわれることを示している。

演算結果をレフト、ライト、Segment-ライト木と比較して図4に示す。図から明らかなように、システムの資源を考慮した演算パイプラインが、効率的な処理をおこなっている。さらに、処理装置に2台のディスクを接続した場合と、1台の場合に、生成されたパイプラインを図5に示す。並列計算機の資源が変化した場合に、異なった長さのパイプラインを生成していることがわかる。



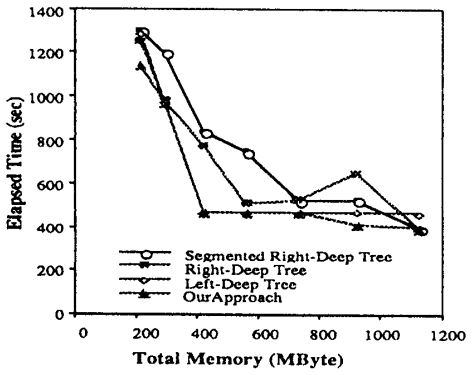
(a) Join Graph

relation	A	B	C	D	E	F	G	H	I	J
K Tuple	10	30	50	40	20	60	70	90	80	100

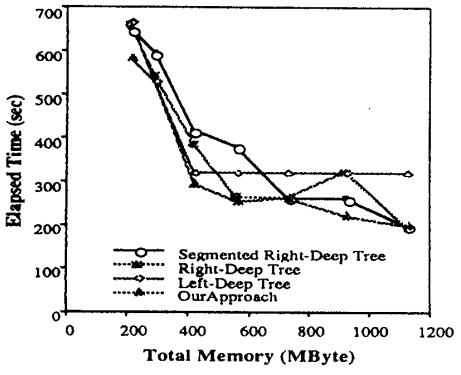
joinability	a	b	c	d	e	f	g	h	i
	0.05	0.05	0.02	0.05	0.025	0.0125	0.025	0.017	0.01

(b) Relations Sizes and Joinabilities

図3 問い合わせ

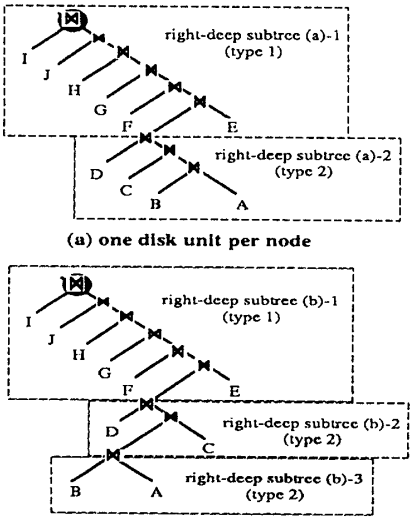


(a) one disk per node



(b) two disk units per node

図4 演算結果



(a) one disk unit per node

(b) two disk units per node

図5 資源の影響

5. おわりに

疎結合並列計算機上で多重結合演算を実行する際、資源を考慮した演算パイプラインの生成方法およびその評価について述べた。今後は、マルチユーザ環境への適用や、データにスキューがある場合に我々の手法を適用していきたい。

参考文献

[1]D.Shneider and D.J.DeWitt, "Tradeoffs in Processing Complex Join Qeuries via Hashing in MultiProcessor Database Machines", Proc. of the 1990 VLDB Conf.,pp.469-480, 1990

[2]M.S.Chen, M.Lo, P.S.Yu and H.C.Young, "Using Segmented Right-Deep Trees for the Execution of Piplined Hash Joins", Proc. of the 1992 VLDB Conf.,pp.150-26, 1992

[3]L.Harada and N.Akaboshi, "An Efficient Query Execution Plan for Multi-Way Joins in Shared-Nothing Database Environment", 情報処理学会第46回全国大会,1993

[4]L.Harada and N.Akaboshi, "Evaluation of Linear join Processing Trees in Shared-Nothing Database Environment", Proc. of the 1993 ICCI Conf.,1993