並列 SQL サーバ SDC-II におけるバッチ問合せ処理方式

7D－2

田村孝之　喜連川 優　高木幹雄
東京大学 生産技術研究所

1　はじめに

SDC-II (Super Database Computer II) は，大規模関係データベースに対する非定型問合せ処理の高速化を目的として開発された，高並列関係データベースサーバである [4]. SDC-II はデータ処理モジュール (DPM) と呼ぶ処理ノード 8 台を間接多段網 (変形オメガネットワーク) で結合した shared-nothing アーキテクチャを採用しており，さらに各 DPM を 4 台の磁気ディスク装置と最大 7 台のプロセッサからなる共有バスクラスタで構成している．SDC-II では，I/O インテンシブな環境に適したデータ駆動型のプロセス実行モデルを採用しており，これまでに，複数の結合演算を含む単一問合せに対して実装を行ない，その有効性を示してきた．

一方，現実のアプリケーションでは DBMS に対して複数の問合せをまとめて発行することが多いため，これらの問合せに共通する演算を抽出し，一括して処理することで，問合せの集合全体に対する処理時間を短縮できると期待される．現在，バッチ問合せに対する高速化技法は，問合せ最適化コンパイラの分野でシミュレーションに基づいて研究されている [1, 2]. しかし，実行時に出現する様々な問題を把握し，最適化のアルゴリズムにフィードバックを与えるためには，具体的な問合せを実機上で処理する実験環境が不可欠である．そこで，共通リレーションのロードを一括して行なう場合を対象とし，SDC-II 上でバッチ処理を支援する機能について検討を行なった．

2　共通リレーションに対するバッチ処理

複数のハッシュ結合演算を含む問合せの集合が与えられた時に，それらをバッチ処理することで，

1. 共通するリレーションのロード
2. 共通するハッシュテーブルのビルド
3. 関係演算木の共通する部分木の実行

などを 1 回にまとめられる可能性がある．一つのアプリケーションから複数の問合せが発行される場合，同じリレーションに繰り返しアクセスする可能性が高いため，1 回のロードでどれだけの問合せを処理できるかという問題は主にメモリ容量などの実行環境による制約を受ける．これに対して，ハッシュテーブルや部分木あるいは中間結果を共有するには結合属性，選択述語，および射影演算などが全て一致している必要があり，より厳しい条件を満たさなければならない．

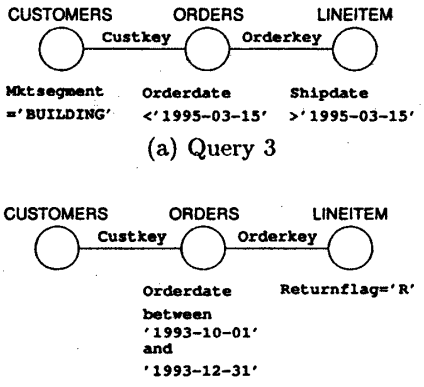

図 1: TPC-D ベンチマーク問合せの結合グラフ

ここで問合せの具体的な例として，意思決定支援システムを対象とする TPC-D ベンチマークを考えることにする．このベンチマークは，現実的なビジネス分野での応用を反映し，大量のデータに対する複雑な演算が行なわれるよう設計されている [3]. これには，6 つのリレーションと 17 の問合せが含まれるが，選択述語がそれぞれ異なるのでハッシュテーブルや中間結果の共有は行なえない．しかし，射影演算によってタプル長が短くなり，中間結果も小さいためメモリ容量には余裕があり，多くのハッシュテーブルを同時に構築することが可能である．従って図 1に示すように同一のリレーションにアクセスする問合せに対してロードをまとめて行なうことで，I/O 負荷を大きく低減することができる．このように，リレーションのロードはバッチ化できる可能性が高く，問合せが I/O バウンドで処理されている限りは十分な性能向上が得られるので重要性が高いといえる．

一般に，バッチ問合せを処理するにはコンパイラが出力したバッチ処理専用のコードを実行することになる．ハッシュテーブルや中間結果のレベルで共有が可能な場合には全体の実行プランが影響を受けるだけであるが，ロードの一括化においてはリレーションを共有する複数の問合せの処理を並行して行なうようにコードが変更される．多重結合演算の実行時には，ハッシュテーブルのオーバフローが発生したらハッシュテーブルの分割やパイプラインの短縮を行なう，パイプラインのステージ間

Batch query processing method in the parallel SQL server SDC-II
T.Tamura, M.Kitsuregawa, M.Takagi
Institute of Industrial Science, University of Tokyo
7-22-1, Roppongi, Minato-ku, Tokyo 106, Japan.

のフローコントロールのために各プロセス毎の優先順位を制御する，など多くの問題を解決しなければならない．ロードの一括化を行なうと多数のハッシュテーブルが同時に構築されるため，これらの問題には一層敏感になることに注意する必要がある．全ての問合せを単一のプロセスを用いて処理する方式では，ある問合せの処理が滞ったり，一つのハッシュテーブルがオーバフローしただけで全ての問合せの実行に影響が及んでしまう．このような状況では問題の起こった問合せだけ実行プランを変更することにすれば，ディスクに退避するデータ量や実行を遅延する問合せの数を減らすことができるので，全体の処理時間を短縮できるはずである．そのためには問合せ毎に独立のプロセスを用い，各プロセスに単一問合せ処理と同じものを用いることによって，動的にバッチ処理の多重度を変更できるようにしなければならない．

3 SDC-II プロセス実行方式のバッチ問合せへの対応

SDC-II では shared-everything アーキテクチャを持つ DPM 内においてはデータ駆動型の実行モデルを採用しており，大容量データベースに対するシーケンシャルアクセス性能の向上と多重結合演算におけるパイプライン処理の効率的実行を達成している [4]．これは，プロセスが主体となって入力操作を要求してデータを得る従来の方式とは対照的に，到着したデータに対してアイドル状態のプロセッサが 1 つ割り当てられ，そのデータを処理すべきプロセスを実行するというものである．従来のプロセスモデルにおいて問合せ毎に独立したプロセスを割り当てて共通リレーションのロードをバッチ化した場合，共有データに対するアクセスタイミングの制御が難しくなり，コンテクストスイッチのオーバヘッドも無視できなくなってしまう．しかし，SDC-II の実行モデルにおいては一つのデータから駆動されるプロセスを複数指定できるようにすれば，プロセス間でのデータの共有を容易に実現することができる．

そこで，この変更を可能にするためにシステムソフトウェアの拡張を行なった．これまでは，入力データを集中管理するディスパッチャの部分とデータを処理するプロセスとは明確に分離されていなかったが，多様な問合せに対する柔軟性を得るため各プロセスを入力データへのポインタを引数として受け取るコールバック関数として記述するようにした．これらのプロセスは，read や recv などの入力用のプリミティブは持たず，受け取ったデータを加工してファイルやネットワークへ出力したり，ハッシュテーブルへ追加したりする．大抵の場合，これらの処理に必要な記述量は短く，出力プリミティブの中でフリーメモリの不足や出力待ち行列長の制限などの例外事象が起こってもブロックしないようにすることで，プロセス毎にコンテクストを保持する必要をなくすことができる．問合せの処理を開始する際には，実行木のリンクに相当するディスクやネットワークのデータストリームを開設し，それらのデータストリームのそれぞれに実行木のノードに当たるプロセスをコールバック関数として対応付ける．

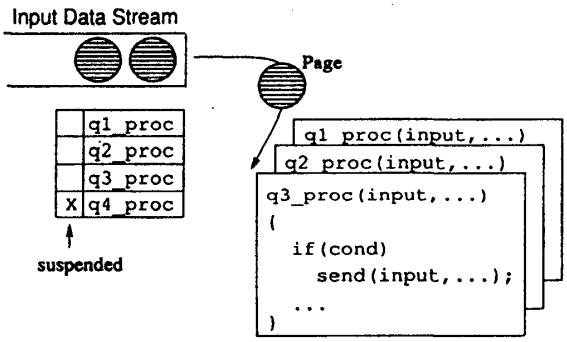

図 2: SDC-II プロセスモデルにおけるデータの共有

このような環境の下でロードの一括処理を行なうには，一つのデータストリームに対して複数のコールバックを登録し，入力データに対してそれらを順に評価すればよい (図 2)．メモリオーバフローなどの例外事象が発生しない限りは，単一プロセスによる実行に比べて関数コール程度のオーバヘッドで処理を進めることができる．また，処理の途中で例外事象が発生した時には，実行を中止する問合せを選択し，中間結果の退避や割り当てられていたメモリの解放などを行なって対応するコールバック関数の登録を抹消することにより，他の問合せの処理を継続することができる．

4 まとめ

複数問合せをバッチ処理する際に最も頻繁に見られる共通リレーションのロードについて検討し，実行時の例外処理を効率良く行なうために SDC-II のプロセス実行モデルを拡張してバッチ処理を支援するための方式について述べた．今後はバッチ問合せ処理を行なった時の性能向上を評価し，実行時に現れる問題とその解決法について調べていきたい．

参考文献

[1] M.Mehta, V.Soloviev, and D.J.DeWitt. “Batch scheduling in Parallel Database Systems”, Proc. of 9th DE Conf., 1993, pp.400–410.

[2] H.Lu and K.L.Tan. “Batch query processing in shared-nothing multiprocessors”, Proc. of 4th DASFAA Conf., 1995.

[3] TPC. “TPC Benchmark^{TM} D (Decision Support)”, Working Draft 6.0, 1993.

[4] 中村，平野，田村，喜連川，高木. “スーパーデータベースコンピュータ SDC-II におけるシステムソフトウエアの設計と実装”, 信学論 Vol.J78-D-I, No.2, 1995, pp.129–141.