並列ＤＢサーバＨｉＲＤＢにおけるＳＱＬ最適化処理方式

３Ｑ－５　金丸　剛　広畠　清美　高木　久美樹　正井　一夫　宮崎　光夫
日立製作所ソフトウエア開発本部

１．はじめに

計算機を用いた業務の拡大とともに、データベース化の望まれるデータ量は増大する一方であり、データベース処理において高いスケーラビリティが必要とされる。スケーラビリティを得るための１つの方式として、データベース処理の並列化が考えられる。日立製作所では、検索処理だけでなく更新処理も並列に実行することができるスケーラブルデータベースサーバ（HiRDB）を開発した[1,2]。本稿では、HiRDB において、分割並列処理、パイプライン並列処理を考慮し、CPU 負荷のバランスをはかりながら、高いスケーラビリティを実現するための SQL 最適化方式について報告する。

２．HiRDB の特徴

２．１　概要

HiRDB は UNIX ベースの並列リレーショナルデータベースシステムであり、複数のサーバマシンを高速ネットワークで連結したクラスタシステム上で動作する。高いスケーラビリティを実現するために、アーキテクチャとして、各プロセサが DB（データベース）を格納したディスクを共有しないシェアドナシングのデザインを採用している。シェアドナシングのデザインでは、DB を格納した各ディスクに対するアクセス（データのヒット件数）の偏りが、そのまま CPU 負荷の偏りになる。CPU 負荷の偏りはスケーラビリティの妨げになるため、HiRDB では、フロータブルサーバと呼ぶ実表アクセスをしないプロセサを設定し、フロータブルサーバに処理を振り分けることで CPU 負荷のバランスをとる。

２．２　HiRDB の構成

HiRDB は、主に次の４つのサーバ（並列データベースの機能の単位）から構成される（図１）。

（ａ）SQL サーバ：SQL の解析と各サーバへの並列実行要求、結果の収集を行う。
（ｂ）DB アクセスサーバ：DB へのアクセス、ソート等のデータ演算処理を行う。
（ｃ）フロータブルサーバ：ソート等のデータ演算処理を行う。
（ｄ）ディクショナリサーバ：DB 定義情報を管理する。

クライアントからの問合せは SQL を用いて SQL サーバに入力される。SQL サーバでは、ディクショナリサーバから各種の定義情報を受け取り、それを基に SQL の解析をおこない SQL を並列実行可能な単位に分解する。SQL の解析が終わると、SQL サーバは、DB アクセスサーバ、フロータブルサーバに SQL 実行要求を出す。要求を受けたサーバは問合せを実行し、処理結果を返す。

２．３　並列処理の例

HiRDB では、分割並列、パイプライン並列の２つの並列化技術を用いている。ここでは、ソートマージジョイン処理を用いて、分割並列、パイプライン並列処理を説明する。クライアントから入力される SQL は、どのようなデータを検索したいかを記述しているだけである。そのため、SQL を並列実行可能な単位に分割することを考える。ソートマージジョインは、各表からのデータの取出し処理、取出したデータのソート処理、ソートされたデータの結合処理、および、クライアントへの結果の返送処理の４つの処理に分解することができる。データの取出し処理時には、複数のサーバに分割格納されたデータを同時に取出すことで、分割並列性を得る。次に取出したデータを順次、他のサーバに転送し、転送先のサーバにおいてソート処理を行うことで、パイプライン並列性を得る。ソート処理が完了すれば、次に結合処理を実行する。そして、結合されたデータから順次結果をクライアントへ返送する。ここでも、パイプライン並列性を得ることができる。

２．４　スケーラビリティを得るための課題

２．１で述べたように、シェアドナシングのデザインを採用している HiRDB では、各ディスクにアクセスできるプロセサは特定されるため、DB の特定データにアクセスが集中した場合、特定のプロセサに負荷が偏ってしまう。そして、その部分がネックになるため、トランザクション全体のスループットが上がらないという問題点がある。特に、ソート処理は、I/O よりも CPU の負荷が大きくなる傾向にありこの問題が顕在化しやすい。そこで、負荷バランスを取り直すためには結合処理、グループ化処理等にて発生する、ソート処理を適切に分割し、どのサーバを何台用いて処理すれば最適であるかを決める必要がある。

３．HiRDB の SQL 最適化方式

HiRDB の SQL 最適化処理では、入力された SQL に対する DB アクセス手順の生成と、生成したアクセス手順を実行するサーバの決定を行う。アクセス手順の生成に関しては報告されているので[3,4]、本稿では生成したアクセス手順を実行するサーバの決定方式について述べる。

３．１　ソート・ジョイン処理用のサーバ数の決定

最適化処理は、SQL コンパイル時と SQL 実行時に行う。SQL コンパイル時には、入力された SQL を並列実行可能な単位に分解し、DB アクセス手順を生成する。並列実行可能な単位とは、DB を格納したディスクからのデータの取出し処理、取出したデータの演算処理（ソート、グループ化、ジョイン等）、結果の収集・返送処理である。デー

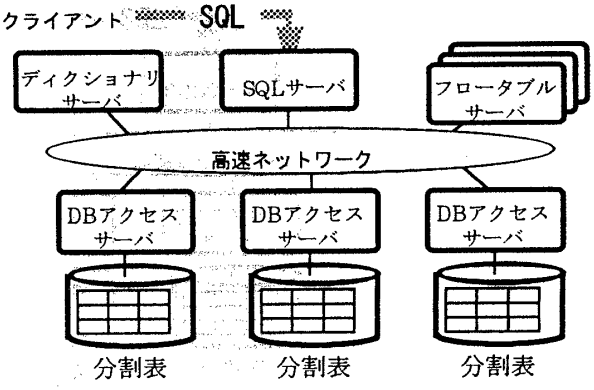

図１　HiRDB のサーバ構成

The query optimization for paralell DB server HiRDB
Tsuyoshi Kanamaru, Kiyomi Hirohata, Kumiki Takagi, Kazuo Masai, Mitsuo Miyazaki
Hitachi Software Development Center

タの取出し処理には必要なデータが格納されたDBアクセスサーバを用いるのでサーバ数は表の分割状態から一意に決まる。結果の収集・返送処理はSQLサーバ1台で行うので決定する必要はない。しかし、ソート・ジョイン処理に用いるサーバは、プロセサの負荷分散と分割・パイプライン並列効果が最適になるように、必要なサーバ台数を求める。ソート用のサーバ数が増加すれば、ソートにかかる時間は減少する。それに対して、ソート用のサーバ数が増加してもデータ取出しにかかる時間は減少しない。そこで、ソート用のサーバ数はデータ取出し処理に対して、ソート処理が問題ない時間で終わる台数であればよいことになる。ソート用のサーバ台数をnとすれば、nは以下の計算式であらわされる。

(データ取出し時間) × (4.0／pow(
4.0,log(max((読出しページ数)／(リストページ数),1))))
＝ (ソート処理時間) ／n ……(1)

3．2 ソート・ジョイン処理用サーバの割当て

SQL実行時に行う最適化では、SQLコンパイル時に作成したDBアクセス手順に対して、各々の手順を実行するサーバを割当てる。DBからデータ取出しに用いるサーバは、シェアドナシングデザインの場合、一意に決定できるが、ソート・ジョイン処理用にサーバを割当てる時には並列効果が上がるように任意のサーバを割当てなければならない。ソート・ジョイン処理用に必要なサーバ台数は、SQLコンパイル時に(1)式のnを解いて求めるが、実際にサーバを割当てるときの戦略を以下に示す。

strategy1：フロータブルサーバとデータ取出しに用いないDBアクセスサーバを割当候補として、その候補の中からn台ランダムに選択する。

strategy2：フロータブルサーバとデータ取出しに用いないDBアクセスサーバをすべて割当てる。

strategy3：全DBアクセスサーバを候補として、その中からn台ランダムに選択する。

各戦略の選択条件を表1に示す。

表1 サーバ割当戦略の選択条件

条件	戦略
A−B＞n	strategy1
0＜A−B＜＝n	strategy2
A−B＝0	strategy3

n：(1)式で示すソート・ジョイン処理に最適なサーバ数
A：DBアクセスサーバ数＋フロータブルサーバ数
B：データ取出し処理に用いるDBアクセスサーバ数

サーバ数が十分に確保できる場合、パイプライン並列の効果をあげるため、DBアクセスサーバの負荷集中を避けるため（シェアドナシングの弱点克服）、ソート・ジョイン処理にはデータ取出しに用いないサーバを積極的に割当てる。このようにしてHiRDBは、シェアドナシングデザインでは問題となる負荷の偏りを取り除くことをおこない、各サーバにかかる負荷を積極的に均一化する。このとき、ソート・ジョイン処理用のサーバをランダムに選択することで、高トランザクションな環境でのフロータブルサーバの負荷の分散を図る。また、DBアクセスサーバから、フロータブルサーバへのデータ転送時には、ハッシングによってデータの均等分配をおこない、負荷の均一化を図る。

ジョイン処理が階層的に複数存在したときは、同位のジョイン処理に対して異なるサーバを割当てる。また、小規模なシステムでフロータブルサーバが存在しない時には、全DBアクセスサーバの中からランダムに選択することで、適切な台数で負荷分散しソート・ジョイン処理を実行できる。

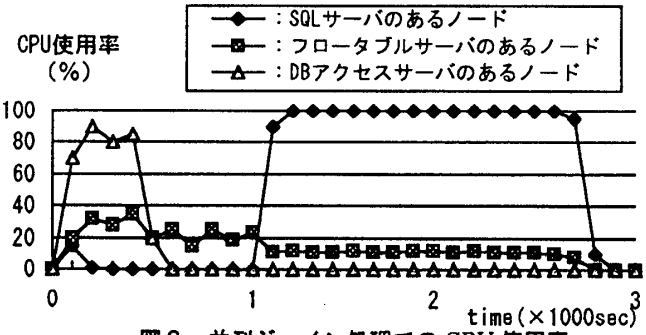

図2 並列ジョイン処理でのCPU使用率

4．結果

DBアクセスサーバ11台、フロータブルサーバ3台、SQLサーバ1台、ディクショナリサーバ1台を用いて、2表の結合処理を行ったときに、各サーバの存在するノードにかかる負荷の平均を図2に示す。DBアクセスサーバとフロータブルサーバは、それぞれ別ノードに配置し、SQLサーバとディクショナリサーバは同一ノードに配置した。DBアクセスサーバでは、DBからのデータの取出し処理、フロータブルサーバでは、ソート・結合処理を行っている。

SQLサーバを含むノードの負荷は、結果の返送処理が始まってから100%になっているが、これは、同一ノードにクライアントが存在したので、そのオーバヘッドである。データ取出しに用いるサーバの負荷は一時的に90%程度まで上昇するが、その後は減少し、他のトランザクションに対応可能であることがわかる。また、DBアクセスサーバとフロータブルサーバでパイプライン並列に動作していることがわかる。一時的に全サーバの負荷が減少する部分があるが、これは、ソート処理用にI/Oが発生し、I/Oネックになっているためである。ソート用のバッファを拡張できれば性能が上がることを示している。

5．おわりに

パラレルデータベースでシェアドナシングデザインを採用したときに各ノードにかかる負荷を分散しながらSQLを実行するサーバの決定方式について述べた。本方式では、分割されたDBへのアクセスが偏った場合でも、データ取出しに用いるプロセサは、全検索時間の1／6程度で開放され、他のトランザクションに対応可能である。ゆえに、高トランザクションな環境でデータ量の増大に対して一定の応答性能を保証するようなスケーラビリティを得るために有用な方式である。

参考文献

[1]正井他：「更新処理を並列実行するUNIX向けDBMSを開発」，日経エレクトロニクス，1995.2.27(No.630).
[2]根岸他：「並列リレーショナルデータベースシステムHiRDBの概要と基本技術」，電子情報通信学会 信学技法DE95-79，1995.
[3]土田他：「RDBにおける埋込み型問合せの最適化方式の提案」，情報処理学会第36回全国大会，3F-3，1988.
[4]岩田他：「並列DBサーバシステムにおける問合せ処理方式の提案」，情報処理学会第48回全国大会，1F-2，1994.