データベースシステムに最適なディスクキャッシュ方式

3B－4

鷹取 功人、野口 昌弘、金森 卓郎

三菱電機（株） 情報システム研究所

１．はじめに

データベースシステムの性能ボトルネックの１つはディスク装置等の外部記憶装置へのデータ更新時間である。この研究はデータ更新時間がデータベースシステムの性能にどのように影響を与えているかを調査したものである。また、ディスクキャッシュの構成をデータベースに最適化し評価した。この結果、ディスクキャッシュによりｼｽﾃﾑの総合処理性能がディスクキャッシュ無しの場合に比較して2倍程度まで向上することを確認した。

２．評価システム

データベースを使用した代表的なアプリケーションにリレーショナルデータベースを使用したオンライントランザクション処理がある。今回は、その典型として銀行の預金取引業務をモデルとしたＴＰＣ－Ｂ[1]を参考にしたベンチマークテストを作成し実施した。データ件数（account）は200万件、データベースのサーバ上のデータバッファに2.6Mバイトを割り当てた。図１にトランザクションの内容を示す。

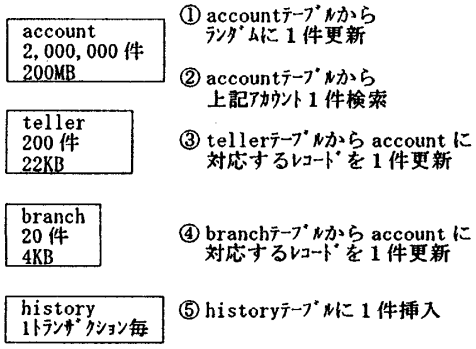

図１

また、ディスク制御装置のトレース機能を使用して、ベンチマークを動作させた時のコマンド等を逐次トレースした。このトレースをソースとしてシミュレーションを実行し、各種の情報を得る。

３．測定結果

トレースした結果、ライトコマンドが全体の62%，リードコマンドが全体の38%であった。また、転送カウントは全て2KBであった。アクセスしたディスク装置上のアドレス分布については、トランザクションログ（以下log領域と呼ぶ）の領域に全体の37%、account,historyテーブルのディスク装置上の領域（以下account、history領域と呼ぶ）が60%を占めていた。また、teller,及び、branchテーブルのディスク装置上の領域（以下teller、branch領域と呼ぶ）にアクセスしたアドレス範囲は狭く、実際のディスク装置へのアクセスはほとんど無い。逆に、account、history領域には、アクセスが広範囲にわたっていた。log領域のアドレス範囲は集中しているにもかかわらず、ディスク装置へのアクセスが４割もある。

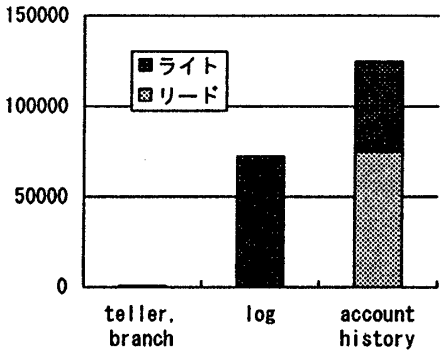

図２

図２に領域別のコマンドをグラフ化した。log領域に対しては、ほぼ全てがライトコマンドである。account,history領域に対しては、40%がライトコマンド、残り60%がリードコマンドであり、全リードコマンドのほとんど全てをこの領域がしめる。teller,branch領域はﾗｲﾄが大半をしめるが、絶対数が少ないため全体に及ぼす影響はほとんどない。

４．シミュレーションと結果

次に、今回試作したディスクキャッシュのヒット率やアクセス時間を上記トレースデータを基にシ

---

*A Method of Disk Cache for RDBMS
Norihito Taktori, Masahiro Noguchi,
Takuro Kanamori
Mitsubishi Electric Corp.

ミュレーションを行った。このディスクキャッシュはストアイン型の16WAY セットアソシエイティブ方式でキャッシュメモリ容量は64Mバイトである。キャッシュメモリは4Kバイトのブロック毎に分けて管理する。キャッシュメモリ上に空きが無くなるとLRU (Least Recently Used) アルゴリズムに従って最も使われていないデータブロックをリプレースする。

ディスクキャッシュの動作は、リードヒット時とライトヒット時にはディスク装置へアクセスせず、直接ディスクキャッシュとホスト間でデータ転送を行う。リードミスヒット時とライトミスヒット時には、ブロック単位でディスク装置へアクセスを行い、ホストへは要求分だけ転送する。また、リプレースするブロックがライトブロックの場合にはこのブロックデータをディスク装置へ書き戻す処理が加わる。

シミュレーションの結果、ライトのヒット率は、log領域 83%、account、history領域は非常に高く98%、その他の領域も90%近くある。リードのヒット率は、account,history領域が46%ある。log領域2%、teller,branch領域 53%となっている。但し、これらの領域は絶対数が少ないので性能に与える影響は少ない。以上を総合すると、全ライトコマンドの89%、全リードコマンドの46%はディスクキャッシュにヒットし、全体としてはヒット率は73%になった。

次に、ディスクキャッシュの構成を変更し、キャッシュ容量は同一のままで1ブロック当たりのセクタ数を変更してシミュレーションした。これは、1ブロック当たりのセクタ数を１にして、ライトミスヒット時のデータ補完のためのディスクリードを要らなくした場合の性能向上を調べ、また、ブロックサイズを大きくして先読み量を多くした時に、どれだけキャッシュのヒット率に影響するかを調べるためである。

その結果、1ブロックを1セクタのサイズにすることにより、キャッシュのヒット率は下がるものの、ライトミスヒット時のディスクリードが無くなって全体としてディスクアクセスが減少した。また、1ブロックのサイズを逆に8セクタにすることにより、キャッシュのヒット率が向上し、上記のディスクリードは必要になるものの、全体としてディスクアクセスが減少した。さらに、1ブロックのサイズを16セクタにすると、今度はキャッシュのヒット率は若干向上するものの、ブロックの大きさによるデータ転送時間が増加し、逆に平均応答時間が増加してしまう事が判った。

この平均応答時間は、１つのコマンドを処理するために要する時間であり、次のように計算した。

平均応答時間 = ROT + SEEK + OVR + TRNSFR

ここで、

ROT　：ディスク装置の平均回転待ち時間 = 8.3ms

SEEK　：ディスク装置の平均シーク時間　= 13.5ms

OVR　：ｺﾏﾝﾄﾞ処理に要するｵｰﾊﾞｰﾍｯﾄﾞ　= 1.5ms

TRNSFR：ﾃﾞｰﾀ転送時間　= 転送容量 * 0.5ms/KB

なお、シミュレーションによる平均応答時間をディスクキャッシュが無い場合を１として図３にグラフ化した。

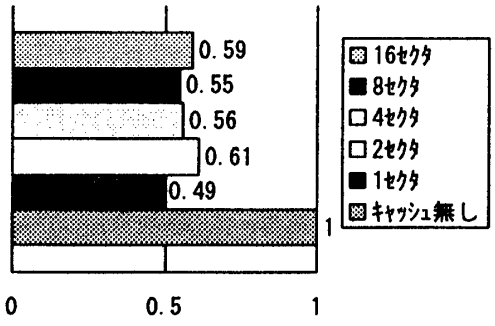

図３

５．結論

今回実施したようなアプリケーションにおいては、キャッシュの１ブロックを１セクタにした方式が総合的に最良であり、ライト動作において特に効果がある。また、ディスクアクセスが４割に減少し、その時の平均応答時間が半減することが導き出された。

６．おわりに

実際に上記のストアイン型のディスクキャッシュを試作し、実機にてベンチマークテストを実施し、ディスクキャッシュが無い場合に比べてシステムの性能が約２倍になることを確認した。

今後は、よりキャッシュ制御を改良し、さらに性能向上を計る考えである。

参考文献

[1] Transaction Processing Performance Council TPC B Benchmark B Standard Specification 23 August 1990