６４ビットＯＳにおける大規模メモリ効果

爲重　貴志† 　高本　良史† 　大辻　彰‡ 　中島　隆夫‡
（株）日立製作所中央研究所† 　（株）日立製作所ソフトウェア事業部‡

１．はじめに

近年、CPU チップ・レベル、OS レベル、アプリケーション・レベルで 64 ビットへの対応がなされ、64 ビットシステムが実用化され始めている。データベース分野では、リレーショナル・データベース（RDB）における VLM（Very Large Memory）あるいは、LIMD（Large-scale In-Memory Database）と呼ばれる高速化手法によって、64 ビット処理の有効性が明らかになりつつある。今後、主に情報系データベースを中心としたビジネスアプリケーションを基盤として 64 ビット化が進むと考えられる。

本稿では、データベース分野における 64 ビット OS における大規模メモリ効果について定量的な測定を行った。具体的には、構築した大規模データベースに対し、シーケンシャルとランダムというアクセス特性をもつ SQL を発行し、クエリ実行に要する全処理時間や CPU 処理時間などの評価パラメータを用いて、大規模メモリ効果を性能向上率*という形で定量的に評価した。

* 性能向上率
=[最小ﾒﾓﾘ構成時の処理時間]/[最大ﾒﾓﾘ構成時の処理時間](倍)

２．６４ビット OS における実機測定

２．１．データベース構成

使用データベースのベンチマークは、情報系データベースの評価では標準となっている TPC-H を用いた。今回、データベースの大きさを表す指標である SF（Scale Factor）を SF=7 とした。これにより、データベース全体は 9600MB となり、64 ビット OS の性能効果の一つである大規模メモリ効果を測定するためには十分なデータベース量を確保できる。

本データベースは、表１に示すように４つのテーブルから構成されている。各テーブルは、４～９つのカラムから成り、各カラムには integer 型・varchar 型・decimal 型・date 型といった型が存在する。また、検索条件カラムに対しては、インデックスが作成されている。

表１．各テーブルとレコード数

テーブル名	レコード数
PART	1,400,000
CUSTOMER	1,050,000
ORDERS	10,500,000
LINE_ITEM	42,000,000
SUPPLIER	70,000

２．２．実行 SQL

実行した SQL は、（１）全件テーブル検索、（２）３ウェイ・ジョイン、（３）４ウェイ・ジョイン、の３種類である。ディスクアクセスに関して、表２に示すような特性を持っている。シーケンシャル特性を持つクエリでは、順番にデータが検索される。ランダムアクセス特性を持つクエリでは、無秩序にデータが検索される。全ての測定は、warm start で実行した。

表２．各 SQL のディスクアクセス特性

#	SQL	ﾃﾞｨｽｸｱｸｾｽ特性
(1)	全件ﾃｰﾌﾞﾙ検索**	シーケンシャル
(2)	３ｳｪｲ･ｼﾞｮｲﾝ	小規模ランダム
(3)	４ｳｪｲ･ｼﾞｮｲﾝ	大規模ランダム

** インデックス使用せず。
他の２クエリではインデックス使用。

２．３．測定環境

図１に本実機測定に用いたデータベースエンジン、データベース用メモリ、データベースの概略図を示す。本実機測定では、データベース用メモリを変化させた時の全処理時間と CPU 処理時間を測定した。

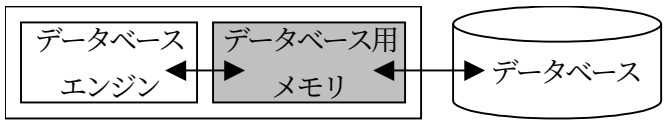

図１．測定環境の概略図

Performance Analysis of Very Large Memory on 64bit Operating System
†Hitachi Ltd., Central Research Laboratory
‡Hitachi Ltd., Software Division

２．４．実機測定結果

（１）全件テーブル検索

クエリ実行に必要なデータがメモリ上に常駐化される 4GB 以上で大規模メモリ効果による高速化が実現されている。メモリが 128MB のときを基準として、メモリが 4GB 以上のときの性能向上率を計算すると、全処理時間で 1.63 倍、CPU 処理時間で 1.01 倍を確認した。ただし、クエリ実行に必要なデータがメモリに常駐化されるまで高速化は図れず、またその間、メモリ上に使用しないデータを常駐することになる欠点がある。

それに対し、プリフェッチを使用した場合、メモリヒット率が大きく向上、全処理時間が大幅に短縮できた（図２参照）。これは、本クエリがデータアクセスの面から見てシーケンシャル性が強いことに起因する。このことから、データアクセス特性がシーケンシャルの場合には、プリフェッチによるメモリ制御がクエリ実行の高速化に有効であることが分かった。

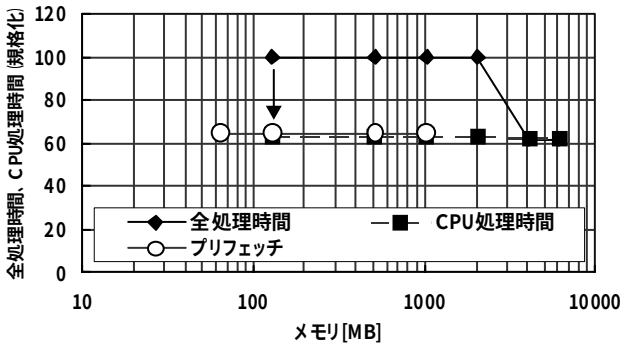

図２．全件テーブル検索における測定結果

（２）３ウェイ・ジョイン

本クエリ実行では、クエリ実行に必要なデータ（インデックス）が 128MB でメモリ上に常駐化され、高速化が実現されている。

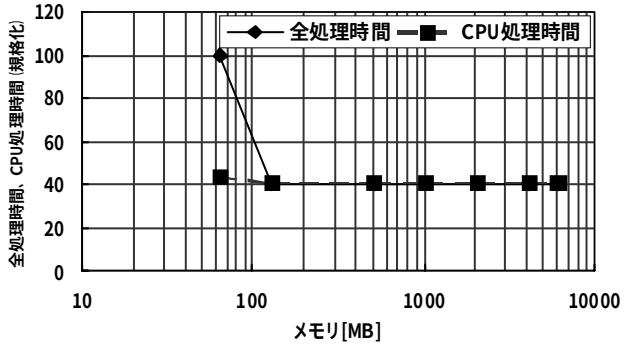

図３．３ウェイ・ジョインにおける測定結果

このように、32 ビット OS の範疇で高速化が可能なクエリも存在した。また、本クエリのディスクアクセス特性はランダムのため、プリフェッチ機能によるメモリヒット率向上は見られなかった。本クエリでは、インデックスのメモリ常駐化が有効であった。

（３）４ウェイ・ジョイン

メモリが 64MB のときを基準として、メモリが 2GB 以上のときの性能向上率を計算すると、全処理時間で 10.20 倍、CPU 処理時間で 1.28 倍を確認した。クエリ実行に必要となる全てのデータ（インデックス）がバッファリングされたことによって、ディスクアクセス回数が削減され、CPU にほとんど待ち時間を与えることなくデータを処理することが出来た結果である。このとき、メモリのヒット率は 100%に達した。また、メモリのミスヒット処理が削減された事で、CPU 処理時間の性能向上につながっている。本クエリのディスクアクセス特性は（２）3 ウェイ・ジョイン同様、ランダムアクセスのため、プリフェッチ機能による高速化は見られなかったが、インデックスのメモリ常駐化が効果的であることが分かった。

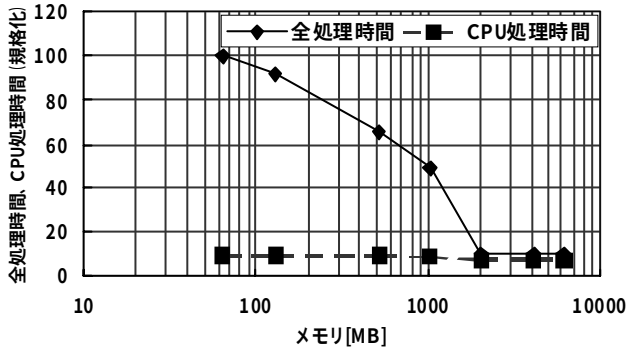

図４．４ウェイ・ジョインにおける測定結果

３．おわりに

64 ビット OS による大規模メモリ効果について、TPC-H を用いて定量的に評価を行い、その有効性を示した。インデックスをメモリに常駐化することで、10.20 倍の性能向上率を得られる場合を示した。また、シーケンシャルなデータアクセスの場合には、プリフェッチによるメモリ制御が有効であることを述べた。ランダムなデータアクセスの場合や、データベースが超大規模な場合には、どのデータをメモリ内に常駐させるかを制御する必要がある。ディスクやテーブルのアクセス密度を用いたメモリ制御方法などが考えられるが、効果的なメモリ制御方式の検討が今後の課題である。