研究課題別評価

1 研究課題名： 主記憶上のデータの高速かつ高信頼な処理の実現

2 研究者氏名： 宮崎 純

3 研究の狙い：

高度情報化社会において最重要な事柄は、より多くの情報を安全に蓄積し、蓄積された大量のデータから必要な情報を高速に検索し加工する技術である。これはデータベースとして過去四十年以上に渡り研究開発され、ビジネスの世界は言うまでに及ばず、多岐多彩に渡る分野で広く利用されている。しかし、昨今の技術革新による半導体デバイス技術や計算機システムの変化、多様化にほとんど対応できていない。

従来のデータベースは高価なサーバ型計算機で大量のデータを処理していたが、近年二極化しつつある。一つは需要が減少しつつあるが、より高性能、より大容量を目指したサーバ型計算機への展開、もう一つは需要が増加しつつある、PC やPDA、組込システムといった小型計算機への展開である（図 1）。特に、後者は従来のデータベースだけでなく、センサーデータ、ストリームデータ等の新しいデータ処理を行うためにデータベース技術が利用されるようになってきた。

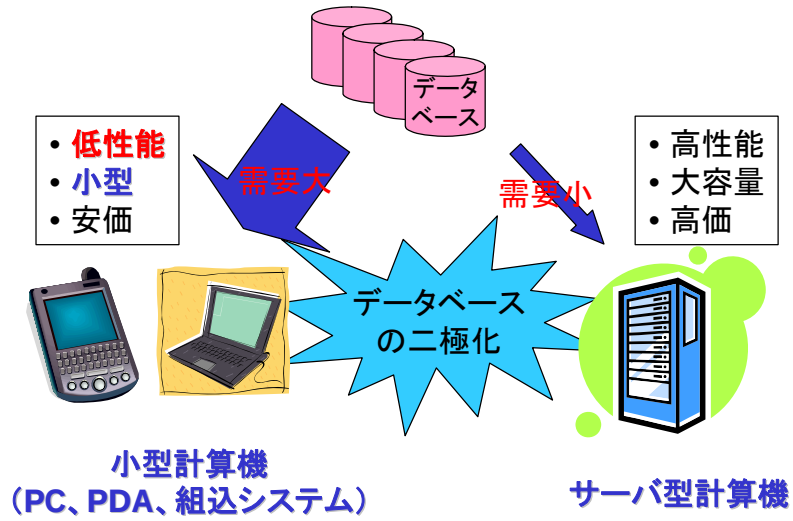

図1: データベースの利用状況の変化

一方でデバイス技術の発展により、半導体メモリ、特に計算機の主記憶に使われる DRAM は、急速にそのビット単価が下がってきている。このため、今までディスク上でしか格納できなかったデータベースを、主記憶上に全て格納する主記憶データベースが現実的なものとなりつつある。主記憶データベースはディスク格納型データベースよりも高速なデータアクセスが可能であり、データベースへの問合せ処理の高速化が可能である。このような背景のもと、近年安価となった半導体メモリを利用し、小型計算機上で効率の良いデータ処理を実現することにより、高速データベ

ースだけでなく、高速ストリーム処理やセンサーデータ処理等、将来不可欠となる高度なデータ処理のための基礎技術を築くことが本研究の狙いである。

この目的を実現するために、ディスクを主ストレージとした従来のデータ処理ではなく、大容量記憶領域を安価に利用できるようになった半導体メモリに着目し、メモリ上で高速、高度かつ新しいデータ処理方法の研究を行った。主記憶に使用される半導体メモリである DRAM は、その記憶容量を上げるためにアクセス速度が犠牲となっている。そこで、サーバ型計算機のような高価かつ大掛かりな技術による解決ではなく、PC や PDA、組込システムといった小型計算機の利用を前提として、DRAM の特徴を最大限に利用しつつ、主記憶上のデータの効率の良い読み出し方法とその効果的な利用方法に関して、ハードウェアとソフトウェアを連携させることにより、小型計算機上でより高速なデータベース処理の実現を目指す。

4　研究の成果:

主記憶に利用されるDRAMのアクセス速度は、最近の著しいプロセッサの速度向上と比較すれば、この二十年間ほとんど改善が無かったに等しく、プロセッサとメモリのアクセスギャップは開く一方である。つまり、主記憶にデータをアクセスすること自体が非常に時間のかかる処理であり、いわゆるメモリの壁(図 2)が問題となっている。このメモリの壁の問題が生じる以前に、主記憶上のデータベースの研究が盛んに行われたが、当然ながらメモリの壁の解決を目指したものはない。メモリの壁を解決する一手段としてキャッシュの利用が挙げられるが、データベースは扱うデータ量が膨大なため、キャッシュが有効に働かない場合が多い。そこで、DRAM の持つ特徴を最大限に利用して、メモリから効率よくデータを読み出す方法を開発することにより、アクセスするデータ量の多いデータベースでも高速処理を可能にし、メモリの壁の問題を解決する。

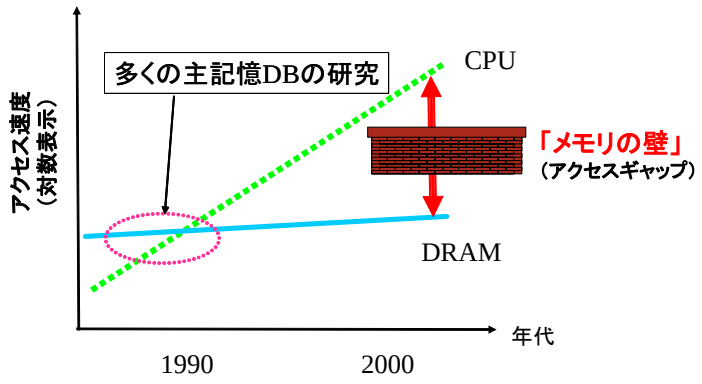

図 1: メモリの壁

本研究では、現在主流である関係データベースのデータベース演算を前提として、以前の研究成果である固定ストライドデータのデータ読み出し方式(stride data transfer; SDT)以外に、新しいメモリからのデータ読み出し方式として、

- ビットマップ形式のアドレス指定に基づくデータ読み出し方式 (Bitmap-based data transfer; BDT)

- 比較器を利用した読み出しデータの圧縮化方式（Comparator-based data compression; CMP）

を提案し、これらを用いた演算アルゴリズムだけでなく、

- SDT、BDT、CMP ならびに従来のキャッシュ指向のデータ読み出し方式を組み合わせたデータベース問合せの最適化

に関しても、研究を行った。

関係データベースは、図 3 のように複数の属性からなるタプルを一つのデータ単位として、それらを表構造にしたテーブルに対して演算を行うことを基本としている。そのため、通常はタプルごとに連続したメモリアドレス上に順に配置される（図 3）。ここで、例えば文字列データ等はその長さを一定にできない可能性があるため、表領域以外の空間に配置することで、表構造へのアクセスと、データそのものへのアクセスを分離して、データベース処理を効率化することが可能である。

関係データベースの演算で最も基本となるのは、選択演算、結合演算、射影演算である。選択演算とは、一つの表に関して、タプルの属性値がある条件を満たすものを取り出す演算であり、結合演算とは、二つの表から取り出した任意のタプルの組み合わせのうち、それぞれのタプルの属性値の関係が、ある条件を満たす場合のみそれらのタプル同士をつなぎ合わせて、一つの表に統合する操作のことである。射影演算は、ある属性のみを無条件に取り出す操作である。

いずれのデータベース演算についても、データアクセスは特定の属性に対して行われるため、その参照アドレスは不連続（図 3 の赤矢印）となる。つまり、データベース演算は、連続アドレス上のデータへのアクセスを大前提としている現在のプロセッサのキャッシュ指向データアクセスに適合しない。属性へのアクセスを連続化するために、タプル単位でなく、同一属性を単位としてメモリアドレスに配置する PAX(Partition Attributes Across)方式が Ailamaki らにより提案されているが、表からタプルが削除されればメモリ空間上に空きが生じるため、一般的には、連続して属性を読み出せる場合の方が稀である。つまり、データの参照アドレスが不連続となることは避けられない。

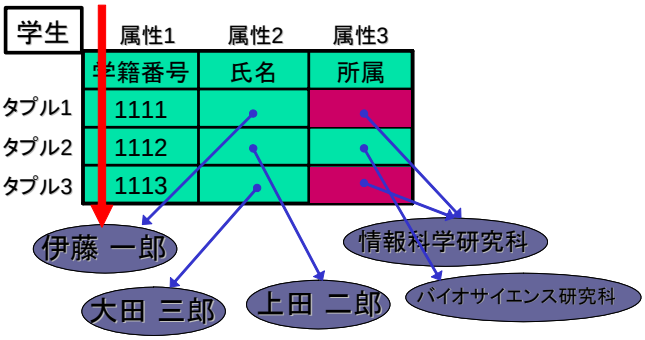

図 3: 関係データベースの表構造とそのアクセス

現在のプロセッサのキャッシュ指向アクセスにより、不連続アドレスのデータを読み出す場合には、計算に不要なデータの読み出しを伴う（図 4 上）。さらに、プロセッサ外部はプロセッサ内部よりも数分の一から数十分の一の遅い速度で動作しているため、計算に不要なデータの読み出しに、

非常に長い無駄な時間を費やすことになる。この無駄な時間をなくすためには、理想的には図 4 下に示すように、真に必要なデータだけをメモリから読み出してプロセッサに転送すればよい。

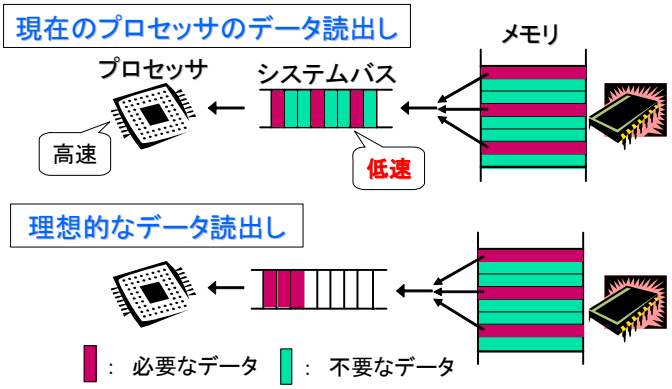

図 4:メモリからプロセッサへのデータ読出し

理想的なデータ読み出しを実現するには、DRAM の持つページモードと呼ばれる機能を利用することで解決できる。ページモードとは、DRAM 中の数 KB の領域を、たとえ参照アドレスが不連続であっても、データを連続して出力できるモードのことである。近年の DRAM ではページモードをサポートしていないが、擬似的にページモードに相当する機能を実現することは可能である。

4.1 ビットマップ形式のアドレス指定に基づくデータ読み出し方式

このページモードを利用して、読み出したい属性の先頭アドレス、データ間隔、データ数を指定して、ある一つの属性を効率よく読み出す Stride Data Transfer (SDT)方式がある。しかし、SDT 方式ではタプル中の複数の属性を同時に読み出したり、削除されたタプルの属性を読み飛ばしたりといった、要求するデータのアドレスに規則性がない場合には対処できないという問題がある。

この問題を解決するために、参照アドレスを実番地で指定するのではなく、アクセスするデータの先頭アドレスとそれに対する相対位置を 0 もしくは 1 からなるビットマップで指定する、Bitmap-based Data Transfer (BDT)方式を開発した(図 5)。BDT 方式ではビットマップを任意に設定することにより、任意のアドレスのデータを無駄なく読み出せる。

例えば、図 5 右は緑の属性値のみ読み出したい場合であり、読み出す必要のある属性位置を 1、読み出しが不要な属性位置を 0 で表し、それらのビットをアドレスの順に並べると参照アドレスパターンを示すビットマップができる。DRAM には最終的に物理アドレスを入力しなくてはならないため、ビットマップから物理アドレスへの変換のためのハードウェアが必要となる。そのハードウェアに関する詳細については後述する。

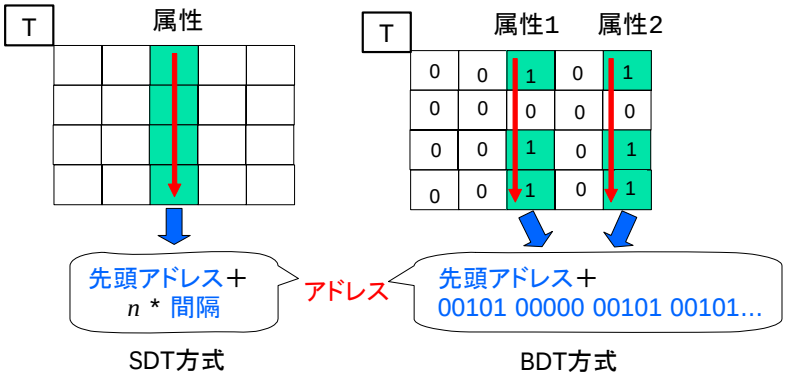

図 5: SDT 方式と BDT 方式の参照アドレス指定

BDT 方式の効果を確認するために、プロセッサ内部と外部の速度比を 10:1 としてシミュレーションにより評価を行った。図 6 はタプル中の三つの属性を同時にアクセスする選択演算の問合せ、図 7 は選択演算と結合演算の組み合わせを含んだ現実的な問合せをそれぞれ実行した際の、問合せ処理が終了するまでのプロセッサのクロック数、すなわち実行時間を計測したものである。”Normal”はタプル単位のデータ格納方式における従来のキャッシュ指向データ読み出し方式、”PAX”は属性単位のデータ格納方式におけるキャッシュ指向データ読み出し方式、SDT、BDT は、タプル単位のデータ格納方式における本研究で開発したデータ読み出し方式である。

図中の”i-cache hit”（紫色）は、プロセッサが休みなく働いており、効率の良い処理が行われている部分、”d-cache miss”（水色）はプロセッサが計算に必要なデータをメモリから到着するのを待っている状態であり、無駄な待ち時間を意味する。両図から明らかなように、現在のプロセッサのキャッシュ指向データアクセスでは、d-cache miss を頻発し、非常に効率が悪いことが分かる。PAX 方式によりキャッシュ指向アクセスに適合するように工夫しても、d-cache miss の発生や、PAX 方式特有の処理のために効率が悪い。一方、BDT 方式は複数の属性を同時にアクセスする場合に有効である。現実的な問合せでは SDT 方式より劣るように見えるが、SDT 方式は、タプルが削除されている場合等には対応できないため、結局、いかなる参照アドレスパターンにも柔軟に対応できる BDT 方式が優れていると結論付けることができる。

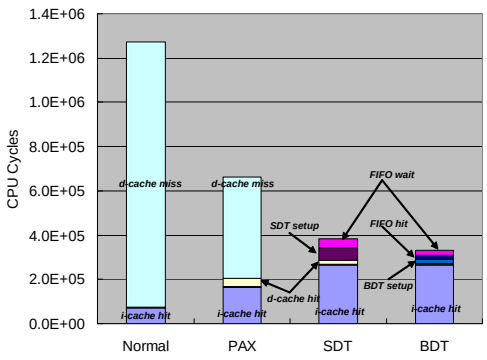

図 6: 三属性同時にアクセスする問合せ

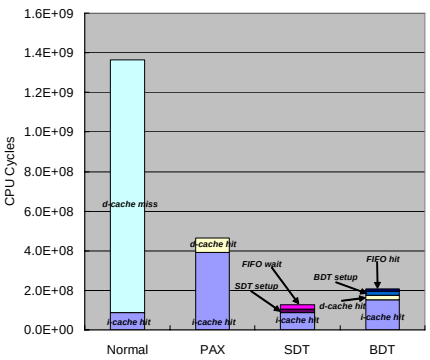

図 7: 現実的な問合せ

DDR2 を利用した通常のメモリモジュールでは、8bit や 16bit のデータ幅の DDR2 チップを複数並べ、ワードデータを垂直分割してそれぞれのチップにデータフラグメントを格納する。例えば、8bit 幅のデータ幅を持つ DDR2 チップを四つ使用し、32bit ワード単位でデータ転送を行うメモリモジュールを考える。DDR2 には posted コマンド機能があり、ACTV コマンド入力後、規定されている遅延時間を待たずに additive レイテンシと呼ばれるバスクロック数だけ先行して READ もしくは WRITE コマンドを入力できる。posted コマンドはコマンド入力のタイミングを調整し、コマンド間の衝突を回避するためのものである。ここで規定遅延時間を 4、additive レイテンシ 3、CAS レイテンシ 4、バースト長 4 と仮定すれば、posted READ コマンドを入力後、additive ならびに CAS レイテンシ分遅れてワードデータが 0.5 バスクロックごとに出力される。図 8 左では、{D0.0, D0.1, D0.2, D0.3}、{D1.0, D1.1, D1.2, D1.3}、…がそれぞれワードを構成する。ACTV コマンドを入力してからバーストデータ転送が完了するまでの時間は、この例では 10 バスクロックとなる。

このデータ読出し方式では、連続アドレスのデータがバースト長分だけ読出されるため、ページモードのような不連続アドレスのデータを読出すことができない。しかし、データを水平分割して各チップへ格納し、各チップの動作を 0.5 バスクロックずつずらせることにより、ページモードに相当するデータ読出しが可能である(図 8 右参照)。水平分割方式でも、{D0.0, D0.1, D0.2, D0.3}、{D1.0, D1.1, D1.2, D1.3}、…がそれぞれワードを構成する。しかし、ワードデータが揃うまでの間データフラグメントを保存するバッファが必要となる。垂直分割方式の例と同じチップとパラメタを用いた場合、この構成では、4 ワードのバースト転送の完了には、Chip0 の最初の ACTV コマンドを発行してから 11.5 バスクロック必要である。外部からみれば、通常の垂直分割方式より 1.5 バスクロック長いレイテンシ、すなわち高々15%遅延時間が長いモジュールのように見える。しかし、各チップにごとに posted READ コマンドと同時にそれぞれ不連続の列アドレスを入力すれば、不連続アドレスのデータを読出せるため、ページモードと同等の機能が得られる。

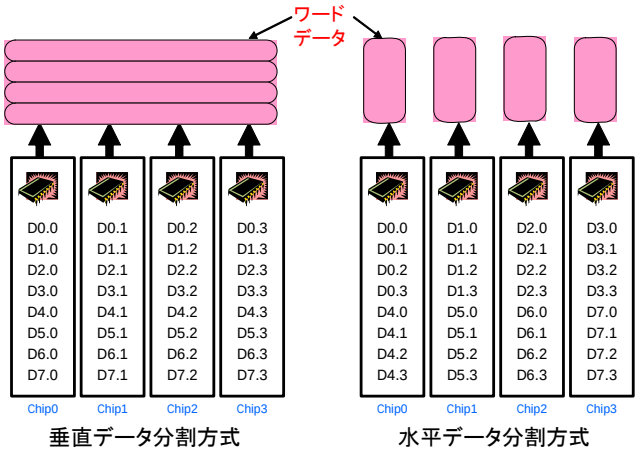

図 8: DDR2-DRAM を利用した BDT 用のデータ格納方式

ここで注意が必要なのは、ストライド間隔が DRAM チップ数の倍数となるとき、前述の例では 4 の倍数になるとき、同じチップからデータを読出す必要があり、アクセスの衝突が生じる。これはベクトル計算機のバンク衝突問題と同じである。図 3 で示したタプル長を固定サイズとするデータ構造の場合、この問題を軽減するためには、(i) タプル長がチップ数の倍数となる場合はダミーの属性を一つ追加する、(ii) 同時に読出される可能性の高い属性同士はその配置間隔をチップ数の倍数とならないように格納する等の対策が必要である。

BDT を実現するためには、通常の小規模計算機の構成と異なる点として、メモリコントローラへの BDT を実現するためのビットマップから DRAM の列アドレスに変換するためのハードウェアの追加、ならびに BDT により得られたデータストリームを CPU で受信するための FIFO の追加が挙げられる。ビットマップからのアドレス変換部分に関して、Altera 社の QuartusII 論理合成ツールを用いてアドレスバス幅およびデータバス幅がそれぞれ 32bit のアドレス変換器を、StratixII シリーズの FPGA を対象に作成したところ、論理回路のゲート数に対応する ALUT 数が 1511、レジスタ数 1139 となった。なお、比較として、DRAM コントローラ部分の回路規模は、Altera 社の提供する評価版の DDR/DDR2-SDRAM コントローラコアの場合、ALUT 数 2237、レジスタ数 1639、メモリ数 2304 であった。

4.2 比較器の利用した読み出しデータの圧縮化方式

基本データベース演算、特に選択演算と結合演算を突き詰めて考えると、単純な属性値の比較、例えば図 3 において所属が"情報科学研究科"かどうか、等の比較に過ぎない。すなわち、データベース演算の核心は、属性値の比較条件が成立するか成立しないかの二者択一を判定することである。この単純な比較をプロセッサで行うために、一つのデータの単位が 32bit や 64bit である情報を、プロセッサ外部の遅い経路を経由してメモリからプロセッサへ読み出すのは非効率である。

そこで、メモリモジュール内に比較器を導入し、メモリモジュール内部でデータベース演算の比較条件の検査を行い、その比較条件が成立するかしないかを、1 と 0 の 1bit に圧縮してプロセッサに比較結果を返す Comparator-based data compression (CMP)方式を開発した。先ほどの結果から、BDT 方式のアドレス指定方式が有効であると分かったため、CMP 方式のアドレス指定も BDT 方式と同じくビットマップにより指定することにする。CMP 方式により、複数の比較演算結果をメモリからプロセッサへビットマップの形で一括して転送でき、プロセッサ外部の低速なデータ転送経路を有効に利用できるようになる。

この CMP 方式を、選択演算に関してシミュレーションで評価した結果を図 9 に示す。この結果から、CMP 方式は、タプルのサイズが小さい場合に有効であることが分かる。結合演算に関しても調べたが、同じくタプルのサイズが小さい場合に有効であった。

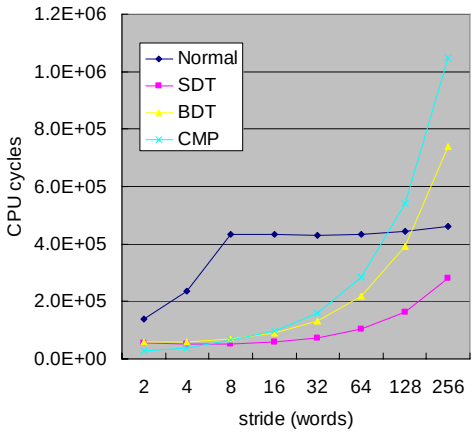

図 9: CMP 方式での選択演算の効率

4.3 データ読み出し方式を組み合わせたデータベース問合せの最適化

ここまでで、SDT 方式、BDT 方式、CMP 方式に加えて現在のプロセッサのキャッシュ指向メモリ読み出し方式(Normal)の、合計四種類のデータ読み出し方式が登場した。それぞれのデータ読み出し方式にはそれぞれ利点、欠点があり、状況に応じて、四種類の中から最も適切なデータ読み出し方法を選択することができれば、データベースの問合せ処理全体の効率化が図れる。このようなデータ読み出し方法の組み合わせ選択は、問合せ最適化と呼ばれる問題の一種である。

データベースの問合せは、選択演算のみといった単一の基本演算だけで構成されるものではなく、通常、複数の基本演算の組み合わせから構成される。例えば、図 9 の問合せでは、二つの表から、一方の表をある条件の下で選択演算を行い、引続いて、もう一方の表と先ほどの選択演算の結果である中間結果の表との間で、ある条件で結合演算を行う。このとき、選択演算にどのデータ読み出し方式を利用し、結合演算にどのデータ読み出し方式を利用するかは、40 通りの組み合わせの中から選ぶこととなる。

図 9 の問合せ例について、図 10 のように、ある表からデータを読み出し(a)、選択演算を行い、その中間結果の表を出力し、引き続いて行われる結合演算でその中間結果の表からデータを読み出す(b)ときのデータの流れに着目すると、(a)の読み出しに CMP 方式、(b)の読み出しに BDT 方式を利用すれば、(a)の出力結果のビットマップが、(b)の BDT 方式のデータ読み出し時の参照アドレスパターンとして直接利用できることが分かる。さらに、中間結果の表への書き出しデータはタプルそのものではなく、ビットマップとして圧縮されており、メモリへ書き込む際にデータがキャッシュから溢れ出す割合を低く抑えることができる、という二つの興味深い優れた特徴を見出すことができる。

しかしながら、この組み合わせがいかなる状況においても最善であるとは限らない。そこで、選択演算に成功するタプル数の割合を変化させた場合の数種類のデータ読み出し方式の組み合わせに関して、シミュレーションにより問合せ処理性能の比較を行った。

図 10 の(c)-(a)の組み合わせに関してその結果を図 11 に示す。 (b)は、(a)が CMP 方式の場合は BDT 方式を、それ以外の場合は SDT 方式としている。

選択演算に成功するタプルの割合がかなり低い場合を除き、(a)と(b)に CMP-BDT 方式を組み合わせて、選択演算から結合演算にデータを流すことにより、著しい問合せ処理の効率化を実現できることが分かる(特に図 11 中の黄色の組み合わせ)。

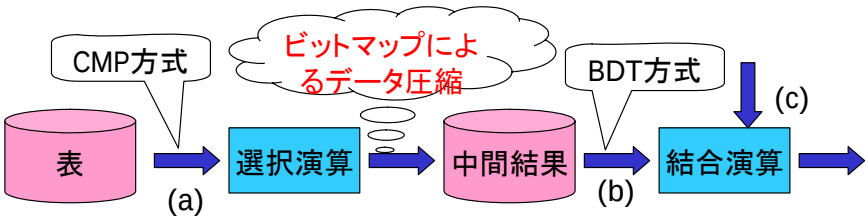

図 10: CMP･BDT 方式の適用による問合せ処理の流れ

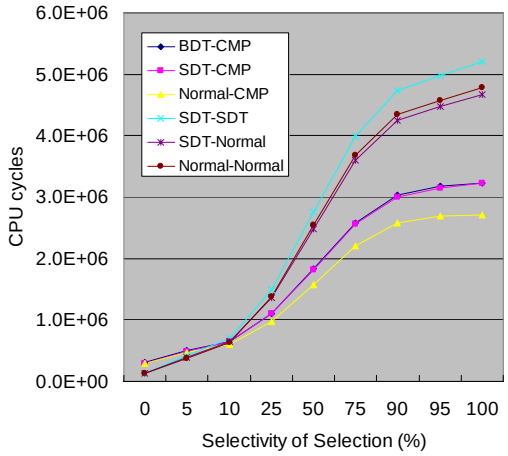

図 11: データ読み出し方式の組み合わせと問合せ処理時間

4.4 研究成果のまとめ

小型計算機において、効率の良いデータベース処理を行うためには、メモリからのデータ読み出しが全体の性能を大きく左右する。そこで、データベース演算に適したメモリ読み出し方法を提案し、それぞれの特性について明らかにした。また、与えられた問合せに対して、複数のメモリ読み出し方式から適切に選択するための組み合わせに関して、今回開発したメモリ読み出し方法がうまく利用できることを明らかにした。これらの結果は、現在の計算機の構成がデータベース演算に適していないことを如実に物語っているが、現在利用できる技術を工夫すれば、データベース処理の著しい高速化が可能であることを実証した。

5 自己評価:

本研究は、非常に多くの分野での利用実績を誇る関係データベースに関して、その演算処理が現在のプロセッサの仮定するキャッシュメモリを前提としたメモリアクセス方式ではデータ構造のチューニング等のソフトウェア技術を用いても不十分であることを明らかにし、現在利用可能な

技術でデータベース処理をどこまで高速化できるか、という問題に取り組んだ。

具体的には、主記憶に利用される DRAM の特性に着目し、固定ストライドデータ読み出し(SDT)、ビットマップ形式のアドレス指定に基づくデータ読み出し方式 (BDT)、比較器を利用した読み出し方式(CMP)を提案するとともに、これらの機能をソフトウェアと連携させ、データベースの問合せ最適化にまで踏み込んでデータベース処理の高速化を追求した。その結果、現在のデータベース処理上の問題であった主記憶からのデータアクセス待ちを著しく減少させ、プロセッサの実質的な利用効率を高めることができることを明らかにした。このような主記憶上のデータベース処理に関して、ハードウェアおよびソフトウェアの両面から検討した例は、世界的にみて極めて独創的かつ挑戦的な研究であり、水準の高い成果を上げることができたと考えている。

近年、センサーデータ処理やストリームデータ処理等で、データベース技術の利用が盛んに行われている。これらは、今後さまざまな場所に組込みシステムとして埋め込まれ、個々のシステムにおいて、主記憶上のデータベースの問合せ処理の高速化が要求される。このような組込みシステムには、小型化のためにプロセッサと DRAM メモリが一つの LSI に搭載される DRAM 混載 LSI が利用されると考えられる。DRAM 混載 LSI は、現在の DRAM チップとは異なり、メモリ周りの設計が自由となる。その際に、本研究で得られた成果がそのまま利用可能であると考えている。

一方で、既存の関係データベースとは概念の異なる XML データベースが脚光を浴びている。XML データベースは木構造が基本であり、表構造の関係データベースよりも遥かにメモリ参照パターンが複雑という問題点がある。しかし、XML の木構造を、本研究と同様の原始データ型による表構造に変換することにより、本研究成果が利用できる可能性は極めて高く、関係データベースだけでなく XML データベース処理の高速化にも貢献できると考えている。

本研究では、主要なデータベースの問合せ処理技術のうち、索引のみが課題として残されている。しかしながら、CMP 方式を利用することにより複数のキーの比較を同時に行うことにより、既存の B 木をより効率的に扱うことが可能であると考えられる。今後は、索引等のデータベースシステム技術だけにとらわれず、一段階抽象度の高いデータ工学システムの観点から、計算機システムの性能を最大限に引き出すための方法論の一般化を行い、XML データベース、ストリームデータ処理、センサーデータ処理、データマイニング等の様々なデータ工学領域の問題に対して、効率的にデータ処理を行うための共通の枠組について検討していく予定である。

6 研究総括の見解:

主記憶に利用される DRAM の特性に着目し、固定ストライドデータ読み出し(SDT)、ビットマップ形式のアドレス指定に基づくデータ読み出し方式 (BDT)、比較器を利用した読み出し方式(CMP)を提案し、それらがプロセッサの実質的な利用効率を著しく高めることを示した。主記憶上のデータベース処理に関して、ハードウェアおよびソフトウェアの両面から検討した研究は独創的であり、水準の高い成果と考えられる。今後の更なる展開や体系化とともに、本研究で得られた成果が DRAM 混載 LSI などで実用化されることを期待する。

7 主な論文等:

論文

(1) 宮崎純: “ビットマップに基づくデータアクセスを利用した小規模計算機向け主記憶データベース処理”, 電子情報通信学会論文誌, Vol.J90-D, No.2, (2007)(採録決定)

(2) Jun Miyazaki: “Query Optimization for Main Memory Databases Using a Memory System with a Comparator Array”, 情報処理学会 DBWeb2006 シンポジウム, (2006)

(3) Jun Miyazaki: “A Memory Subsystem with Comparator Arrays for Main Memory Database Operations”, Proc. of the 21st Annual ACM Symposium on Applied Computing (SAC 2006), Special Track on Database Theory, Technology and Applications (DTTA), pp.511-512, (2006)

(4) Jun Miyazaki: ”Hardware Supported Memory Access for High Performance Main Memory Databases”, ACM First International Workshop on Data Management on New Hardware (DaMoN 2005), pp.41-46, (2005)

(5) 宮崎純, 府川智治, 田中清文: “ストライドデータアクセスによる主記憶データベースの問合せ処理の評価”, 日本データベース学会 Letters, Vol.3, No.2, pp.41-44, (2004)

特許

(1) 宮崎純: “メモリコントローラ、情報処理システム及びリードアクセス方法”, 国際出願 PCT/JP2006/311628, 奈良先端科学技術大学院大学, 2006 年 6 月 9 日

(2) 宮崎純: “メモリコントローラ、情報処理システム及びリードアクセス方法”, 特願 2005-171073, 奈良先端科学技術大学院大学, 2005 年 6 月 10 日