データベース・システム 47－5
（1985・5・20）

情報提供サービスに適用可能な超大規模リレーショナル・データベースマシン

井上 潮、　北村 正、　速水治夫、　中村敏夫
（NTT 横須賀電気通信研究所）

1．はじめに

現在の情報提供型サービスでは、原データ（一次情報）をもとに計算機による処理向きに作成し直したデータ（二次情報）をデータベース（DB）化して利用者に提供する形態が一般的である。例えば文献検索サービスでは、文献から書誌事項を抽出し、抄録とともに検索用のキーワードを付加したものがDB化されている。このように一次情報そのものをDB化するのではなく、二次情報をDB化したものを提供するサービスは、以下のような問題を含んでいる。

①二次情報の作成（特に抄録作成やキーワード抽出など）に専門的なスキルを持った多大なマンパワーを必要とする。

②同じ情報に対する利用目的が時代とともに変ってくることがあり、インデックス等でアクセス法を固定してしまうとこの変化に柔軟に対処することが難しい。

③DBの検索結果がキーワードに依存するため利用者が検索する上でのノウハウが必要でありかつ検索できる情報に限りがある。

これに対して一次情報をそのままDB化することは、多様な利用が可能であるという点で大きなメリットがあり、最近のデータ格納媒体、入出力技術の進歩により、実現するための環境条件が整いつつある。しかし、データ量が膨大になる一次情報DBによるオンラインサービスを現在の汎用マシンで実現することは、処理能力の点から不可能である。

一方、LSI技術の進歩に伴うハードウェアコストの低下を背景にリレーショナル・データベース（RDB）処理専用のアーキテクチャを有する計算機（RDBマシン）の研究開発も活発に進められている[1][2][3]。一次情報をDB化する場合、多様な利用を可能とする意味からRDBが適しているが、従来研究開発されてきたRDBマシンは、連想機能を有する特殊な記憶装置を使用していたり、メインの処理をマイクロプロセッサが分担していたりするため、取り扱えるDBが小・中規模に限定されており、一次情報による超大規模（100GB程度）のDBに対応できるものは性能、コスト上の問題から実現されていない。

本稿では、一次情報を主体とした超大規模データベース向きのリレーショナル・データベースマシン（ADAM）を提案する。ADAMは、高速ディスクアクセス方式、高速サーチ処理方式、高速ジョイン演算方式を実現しており、情報提供サービスに適用した場合は同一素子技術を用いた汎用マシンと比較して、性能で2桁、コスト性能比で1桁以上の向上を達成できる見通しである。

以下、第2章では超大規模RDBマシン実現上の問題点を明らかにする。第3章ではADAMの設計思想と全体構成を述べる。引き続き第4章では高速ディスクアクセスの実現方法、第5章では高速サーチ処理の実現方法、第6章では高速ジョイン演算の実現方法について説明する。第7章ではADAMの性能およびコスト性能比の評価結果を述べる。最後に第8章で結論と今後の研究課題を述べる。

2．超大規模RDBマシン実現上の問題点

一次情報を主体とした超大規模DBを取り扱えるRDBマシンを実現するためには、以下の技術課題を解決しなければならない。

（1）ディスクアクセスの高速化

超大規模DBを用いてオンライン情報提供サービスを実現することを考えると、データの格納媒体はここ数年の間はコスト的な観点から現在汎用マシンで使用されている可動ヘッド磁気ディスクにならざるを得ない。一次情報を主体とするRDBではインデックスによる高速化手法を用いることが困難であるため、データは基本的にフルサーチとなる。このため、数百MB程度と予想されるリレーション中の全タプルを数秒で読み込むことを可能とするディスクアクセスの高速化機構が必要である。

（２）サーチ処理の高速化

サーチ処理の対象となるデータ量が膨大なために、全データをいったん主記憶に読み込んだ後にサーチ処理を行う従来の汎用マシンの処理方法では、ＣＰＵの処理時間および主記憶へのデータ転送量の２点が問題となる。このため、比較的時間のかかるディスクアクセス時間を有効利用し、かつデータ転送量の削減を可能とするサーチ処理方式が必要である。

（３）ジョイン演算の高速化

ＲＤＢではジョイン演算は必要不可欠な機能であり、特に各種統計情報をＤＢ化した場合にはその有用性は極めて高くなる。しかし、ジョイン演算は複数のリレーションを対象とし、かつ各タプルの値どうしを逐一つきあわせて条件を比較する必要があるため、単純な処理方式では処理時間がタプル数の２乗のオーダ（以下、Ｏと記す）で増加する。このためＤＢが大規模化したときの処理時間は膨大な長さとなり、汎用マシンでの処理は実際上不可能となる。

３．ＡＤＡＭの概要

（１）基本的な考え方

ＲＤＢマシンを実現する場合、関係代数の各演算をどのようなアルゴリズムで実行するかによりマシンアーキテクチャが決定される。ＡＤＡＭを実現する上での演算アルゴリズムに関する基本的な考え方は、以下の通りである。

（ａ）すべての演算のＯ(ｎ)化

複数リレーションを対象とする演算、および単一リレーションを対象とする演算でも該リレーション中のタプル相互間の関係を調べる演算は、単純なアルゴリズムで実現した場合、$O(n^2)$ の処理時間が必要となり、演算対象タプル数（ｎ）が大きくなるとこの時間は極めて大きくなる。これらの演算も、リレーション中のタプルがソートされていればその後の処理はＯ(ｎ)の処理時間で行えるため、ＡＤＡＭではＯ(ｎ)の処理時間でソートを行うハードウェア・ソータを利用することにより、すべての演算のＯ(ｎ)化を実現する。

（ｂ）処理の並列化

ＲＤＢではリレーション中のタプルは互いに独立であるため、単一のリレーションの演算は適当な数に分割して並列処理する。また複数リレーションを対象とした演算は、個々のリレーションに対する操作とリレーション間の突き合せを行う操作に分割し、前者はリレーション単位で並列処理する。後者は並列処理が難しいため、当面は直列処理とする。

（ｃ）処理のパイプライン化

各演算はいくつかの処理に分割し、それらをパイプライン的に動作させることにより、処理の高速化とハードウェアの利用率の向上を図る。パイプライン動作の単位としてはデータ転送に伴う制御オーバヘッドを考慮して、タプル単位のきめ細かいパイプライン動作と、適当な大きさのバッファ単位のパイプライン動作を適宜組み合せることとする。

（２）具体的なアプローチ法

上記の基本的な考え方に基づき、前章で述べた大規模ＲＤＢマシン実現上の問題点に対して、以下のアプローチにより解決を図る。

（ａ）ディスクアクセスの高速化

リレーションをクラスタリングして格納しておくことにより、ディスクへの必要アクセス量を削減する。さらに、１つのリレーションを複数のディスクに分散して格納しておき、これらを並列にアクセスすることにより、データの読み込み時間を短縮する。

（ｂ）サーチ処理の高速化

ディスクコントローラ（ＤＫＣ）にサーチ処理専用のハードウェアを付加し、ディスクからのデータ転送に重畳してサーチ処理を行い、サーチ条件に合致したタプルのみを主記憶に転送する。これにより、サーチ処理の高速化とデータ転送量の削減を図る。

（ｃ）ジョイン演算の高速化

ジョイン演算を、ふるい落し、ソート、キー比較（結合）の３段階の処理に分割し、これらをパイプライン動作させる３フェイズジョイン方式により実行する。このうちふるい落し処理は、ジョイン演算の対象となる２つのリレーション中のタプルのキー値をハッシュの技法を使用して照合し、結合可能性のないタプルを除く処理である。このふるい落し処理とソート処理は、いずれもO(n)の時間内で処理することができる専用のハードウェアで実現する。

（３）ＡＤＡＭのアーキテクチャ

ＡＤＡＭのハードウェア構成を図３．１ に示す。ＡＤＡＭはサーチ処理のための専用プロッセッサであるサーチプロセッサ（ＳＨＰ）、関係演算の基本機能を実現するための専用プロセッサである関係演算高速化機構（ＤＡＰ）、およびこれら専用プロセッサの制御と関係演算を実行する制御ＣＰＵの３つを基本コンポーネントとする。この構成は、サービスの要求条件に合せて制御ＣＰＵの能力、ＳＨＰ，ＤＡＰの台数を自由に増減できるため、柔軟性が高くかつ実現が容易である。

（ａ）サーチプロセッサ（ＳＨＰ）

ＳＨＰは磁気ディスク制御装置（ＤＫＣ）中に組み込まれる。この装置をインテリジェント化ＤＫＣ（ＩＤＫＣ）と呼ぶ。ＩＤＫＣはディスクからのデータの読み出しと重畳してサーチ処理を行い（オンザフライ処理）、検索条件に合致したタプルのみを制御ＣＰＵの主記憶上へ転送する。

（ｂ）関係演算高速化機構（ＤＡＰ）

ＤＡＰは関係演算（特にジョイン演算）の高速化を目的として、ハッシュの技法を使用したふるい落し処理を実現するフィルタと、ふるい落し後のタプルのソート処理を行うソータから構成される。ＤＡＰはＩＤＫＣによって主記憶上へ転送されたタプルに対してふるい落しとソートを行い、その結果を主記憶上へ転送する。

（ｃ）制御ＣＰＵ

制御ＣＰＵはＩＤＫＣとＤＡＰの制御および主記憶上のタプルに対する結合処理等を実行するとともに、言語処理、排他制御、障害処理等の従来のＤＢＭＳと同様の機能を実現する。

４．高速ディスクアクセス

ディスクのアクセスを高速化する方法としては、ディスクへのＩ／Ｏ動作そのものを高速化する方法と、ディスクへのアクセス範囲を局所化する方法に大別できる。ＡＤＡＭでは、複数ディスクへの並列アクセスによるディスクＩ／Ｏの高速化と、改良拡張K-d treeを用いたクラスタリングによるディスクアクセス範囲の局所化を併用する。

（１）複数ディスクへの並列アクセス[4]

Ｉ／Ｏ動作を高速化する方法としては、半導体ディスクやトラックパラレル・アクセスディスクなど本質的に高速な二次記憶媒体を使用する方法と、１つのリレーションを複数の汎用ディスクに分散して格納しておき、並列にアクセスする方式がある。これらの方式を比較すると表４．１ のようになり、超大規模ＤＢを実現するためには、コスト面から汎用ディスクを並列アクセスする方式が現実的である。

（２）改良拡張K-d treeによるクラスタリング[5]

リレーションをクラスタリングする方法としては、従来から種々の方式が考案されている。これらの方式の中では、複数アトリビュートへの適用、範囲条件による検索機能、利用特性に応じた最適化等を考慮すると、拡張K-d treeを用いる方法がすぐれている。ＡＤＡＭでは、アトリビュートの値に重複が存在することを前提としてリレーションの分割方法を改良した改良拡張K-d treeによるクラスタリングを実現する。

５．高速サーチ処理

通常の検索では、サーチ処理によって原リレーションの一部のみを抽出した小さなリレーションを作り出すことが可能である。サーチ処理の実現方式としては、汎用マシンのＣＰＵで行う方式、多数のマイクロプロセッサで並列処理する方式と、サーチ処理専用回路を用いる方式が考えられるが、ディスクからのデータ転送速度に追随したオンザフライ処理を安価なコストで実現するためには、専用回路を用

いる方式がすぐれている[6]。ＡＤＡＭでは、サーチ処理をＤＫＣ中に組み込まれたＳＨＰにより実現する。

ＳＨＰと類似の装置は従来でも研究開発されている[7][8]。ＡＤＡＭでは、一次情報を主体としたＤＢに対する多様な検索機能を安価なコストで提供できるようにするため、テーブル制御によるテキストサーチ方式と複合条件判定方式を実現する。

（１）テキストサーチ方式[9][10]

テキストサーチとは、文章等の文字列データ中に検索キーとして指定された文字列が含まれているかどうかを判定する処理のことである。二次記憶に格納された文字列データを対象としたテキストサーチを実現する方法としては、複数の文字を一度に比較する並列比較器法、一文字単位の比較器をカスケード接続する直列比較器法と、有限オートマトンの原理を利用したテーブルドリブン法が代表的である[11]。これらの方式を比較すると表５．１ のようになり、ハードウェアの単純性、実現機能の柔軟性の面でテーブルドリブン法が最もすぐれている。

テーブルドリブン法における技術上の課題はテーブルの作成方法にある。筆者らは、小規模かつ単一のハードウェアで多様な条件による検索を矛盾なく行えるようにするため、以下の４つの機能を同時に実現するテーブル作成アルゴリズムを開発した。

①複数の検索キーについてのＡＮＤ，ＯＲ，ＮＯＴの任意の組み合わせよりなる条件で検索が可能である。
②日本語文章データについては、１６ビットの文字コードを二分割して８ビットずつ処理する。
③可変長don't careおよび固定長don't careを含む検索キーの使用が可能である。
④１つのテーブル上に複数のアトリビュートに対するそれぞれ異なった検索キーを設定できる。

（２）複合条件判定方式[12]

ＳＨＰに与えられる検索条件は、選択演算、制約演算、テキストサーチ等の個々の条件式（単位条件式）をＡＮＤ，ＯＲで結合した論理式で表現される。論理式の判定方法としては、加法標準形で表現された論理式をそのまま連想メモリにマッピングする方法、ラッチとゲートで構成される比較回路を用いる方法と、論理式を主加法標準形に変換し単位条件式の成否のすべての組み合わせをＲＡＭに設定する方法がある。これらの方法を比較すると表５．２ のようになる。

現状の技術レベルでは連想メモリの使用は現実的でないため、残りの２つの方式について、ＳＨＰとして使用できる単位条件式の最大数を１０または２０とした時の論理式判定回路に必要なＩＣチップ数を評価すると図５．１ のようになる。ＳＨＰでは、実際に使用される単位条件式の最大数は高々２０程度であると想定し、主加法標準形に変換する方法を実現する。

６．高速ジョイン演算

ジョイン演算の実現アルゴリズムは、ネステッドループを基本とするものとソートマージを基本とするものに大別できる。ネステッドループは制御が単純であるが、原理上の処理時間は$O(n^2)$であり、$O(n)$の時間内に処理を完了させるためにはｎ台のプロセッサによる並列処理が必要となる。一方ソートマージの場合は、原理上の処理時間が$O(n \cdot \log n)$であるため$(\log n)$台のプロセッサによる並列処理を行えばよく、ネステッドループに比べて実現性が高くなる。ＡＤＡＭでは、ソートマージを基本とした３フェイズジョイン方式によりジョイン演算の高速化を実現する。

（１）３フェイズジョイン方式[4]

３フェイズジョイン方式では、ふるい落し、ソート、キー比較（結合）の３段階の処理でジョイン演算が行われる。ふるい落しはＤＡＰ内のフィルタによりハッシュ化ビットアレイの技法によって実現され、ソートはＤＡＰ内のソータにより２ウエイマージの技法によって実現される。最後にタプルを実際に結合するためのキー比較処理は制御ＣＰＵによって行われる。ＩＤＫＣによるサーチ処理とフィルタにおけるハッシュ化ビットアレイの設定処理、フィルタによるふるい落し処理とソータへのタプル入力処理、ソータからのタプル出力処理と制御ＣＰＵによるキー比較処理はそれぞれパイプライン的に動作する。この概念を図６．１ に示す。

（２）クロスハッシュ方式[13]

ハッシュ化ビットアレイを用いたふるい落しの実現方法としては、ハッシュ回路とビットアレイを各々１つずつ使用して一方のリレーションのみをふるい落す片ハッシュ方式、ハッシュ回路を１つとビットアレイを２組使用して２つのリレーションのふるい落しを逐次に行うシリアルハッシュ方式が従来提案されている。しかし、片ハッシュ方式は一方のリレーションのみのふるい落しとなるためふるい落しの効果は低く、どちらのリレーションをふるい落すかの判定が難しい。また、シリアルハッシュ方式はリレーションごとに逐次にビットアレイの設定処理と参照処理を行うため、所要時間が長くなるという欠点があった。

上記問題点を解決するため、筆者らはハッシュ回路とビットアレイを各々２つずつ使用して２つのリレーションを同時にふるい落すクロスハッシュ方式を考案した。本方式と従来方式の概念を表６．１ に示す。本方式と機能的に同一であるシリアルハッシュ方式について、ふるい落しのための所要時間を評価すると図６．２ のようになる。この図より、クロスハッシュ方式は２つのリレーションのふるい落しができかつ両リレーションの大きさの比にほとんど依存しない高速な方式であることがわかる。

７．ＡＤＡＭの性能とコスト

ＡＤＡＭの効果を定量的に明らかにするため、２種類の検索処理モデルについて処理時間とコストを試算し、現在の超大型汎用マシンで実現した場合との比較を行った。

（１）評価モデル

検索処理モデルＡは、文章ＤＢへの適用を想定したものであり、長大な文字列データを含むリレーションをテキストサーチにより検索するモデルである。検索処理モデルＢは、統計ＤＢへの適用を想定したものであり、２つのリレーションについて選択演算を行った後ジョイン演算を行うモデルである。各モデルの条件を表７．１ に示す。

（２）マシン性能とコスト

マシンの性能は、上記検索処理モデルの実行開始から完了までの処理時間で評価することとし、ＣＰＵ時間、ディスクアクセス時間、専用ハードウェアでの処理時間をもとにして算出した。コストは、ＣＰＵと専用ハードウェアのコストのみを評価することとし、ＡＤＡＭと汎用マシンとで共通の設備となるチャネル、ディスク等のコストは除外した。性能とコストに関する評価式を以下に示す。

①汎用マシン（検索処理モデルＡ）
　Ｔ＝Ｔｄｋ＋Ｔｅｄ
　Ｔｄｋ＝ＭＡＸ（Ｔｉｏ，Ｔｃｓｅ）
　Ｐ＝Ｃｃｐｕ
②ＡＤＡＭ（検索処理モデルＡ）
　Ｔ＝Ｔｄｋ＋Ｔｅｄ
　Ｔｄｋ＝Ｔｉｏ
　Ｐ＝Ｃｃｐｕ＋Ｎｄｋ＊Ｃｓｈｐ
③汎用マシン（検索処理モデルＢ）
　Ｔ＝Ｔｄｋ＋Ｔｊｏ＋Ｔｅｄ
　Ｔｄｋ＝ＭＡＸ（Ｔｉｏ，Ｔｃｓｅ）
　Ｔｊｏ＝Ｔｃｓｏ＋Ｔｃｍｐ
　Ｐ＝Ｃｃｐｕ
④ＡＤＡＭ（検索処理モデルＢ）
　Ｔ＝Ｔｄｋ＋Ｔｊｏ＋Ｔｅｄ
　Ｔｄｋ＝ＭＡＸ（Ｔｉｏ，Ｔｄｉｎ）
　Ｔｊｏ＝Ｔｄｓｏ＋Ｔｃｍｐ
　Ｐ＝Ｃｃｐｕ＋Ｃｄａｐ＋Ｎｄｋ＊Ｃｓｈｐ

ただし、

Ｔ　：全処理時間	Ｐ　：全コスト
Ｔｄｋ：ＤＫｱｸｾｽ時間	Ｎｄｋ：ＤＫＣ台数
Ｔｅｄ：編集出力時間	Ｃｃｐｕ：ＣＰＵ価格
Ｔｊｏ：ｼﾞｮｲﾝ処理時間	Ｃｓｈｐ：ＳＨＰ価格
Ｔｃｓｅ：ＣＰＵ選択時間	Ｃｄａｐ：ＤＡＰ価格
Ｔｉｏ：ＤＫ転送時間	Ｔｄｉｎ：ＤＡＰ転送時間
Ｔｃｓｏ：ＣＰＵｿｰﾄ時間	Ｔｄｓｏ：ＤＡＰｿｰﾄ時間
Ｔｃｍｐ：ｷｰ比較時間	

（３）評価結果

現状のハードウェア性能、コスト等をもとに、上記評価式に基づいた評価結果を図７．１ に示す。検索処理モデルＡについては主にテキストサーチのオンザフライ処理化の効果、検索処理モデルＢについては主に３フェイズジョイン方式の効果によって、性能で２桁、コスト性能比で１桁以上の向上が達成されている。

８．まとめ

ＡＤＡＭは、一次情報を主体とした超大規模データベース向きのリレーショナル・データベースマシンであり、以下の方式を採用することにより従来の汎用マシンと比較して性能で２桁、コスト性能比で１桁以上の向上を可能としている。

①ディスクアクセスの高速化
- ・複数ディスクへの並列アクセス
- ・改良拡張K-d treeを用いたクラスタリング

②サーチ処理の高速化
- ・テーブルドリブン法によるテキストサーチ
- ・主加法標準形による複合条件判定

③ジョイン演算の高速化
- ・３フェイズジョインによるパイプライン処理
- ・クロスハッシュ方式による並列ふるい落し処理

今後の課題としては、ＳＨＰとＤＡＰを試作して制御ＣＰＵと接続して動作させることによるＡＤＡＭの実現性の検証と、ＳＨＰ，ＤＡＰを利用した並列／パイプライン処理によるＤＢの更新方式の確立がある。

〈参考文献〉

[1] G.G.LANGDON,Jr et al."Special Issue on Database Machines",IEEE Trans.Comput.,Vol.C-28 No.6,1979

[2] 植村、弓場他「磁気バブルデータベースマシン」、電総研大型工業技術研究開発PIPS-R-NO.28,29,1981

[3] 疋田、川上他「リレーショナル・データベースマシンＦＲＥＮＤ」、情処データベースシステム39-2,1984

[4] 佐藤、北村他「大規模ＲＤＢ向きデータベースマシンアーキテクチャに関する一考察」，情処データベースシステム35-3,1983

[5] 板倉、井上「リレーションのデータベースマシン向きのクラスタリング方法に関する一考察」、情処データベースシステム47-6,1985

[6] 速水、北村、井上「サーチ処理専用プロセッサの検討」、情処２９回全国大会,1984

[7] E.BABB "Implementing a Relational Database by Means of Specialized Hardware",ACM TODS,Vol.4 No.1,1979

[8] 佐藤、中村他「データベース処理のための専用プロセッサ」、情処論文誌,Vol.23 No.4,1982

[9] 井上、速水、中村「日本語情報データベースに対するテキストサーチ法の一提案」、信学５９年度総全大会,1984

[10] 速水、井上、佐藤「複数索引語の同時検索可能なテキストサーチ法の一提案」、信学５９年度総全大会,1984

[11] L.A.HOLLAAR "Text Retrieval Computers",COMPUTER,Vol.12 No.3,1979

[12] 速水、井上、板倉「検索専用プロセッサにおける論理式判定法の一提案」、情処３０回全国大会,1985

[13] 武田、井上、中村「並列ふるい落しアルゴリズムに関する一考察」、情処２６回全国大会,1983

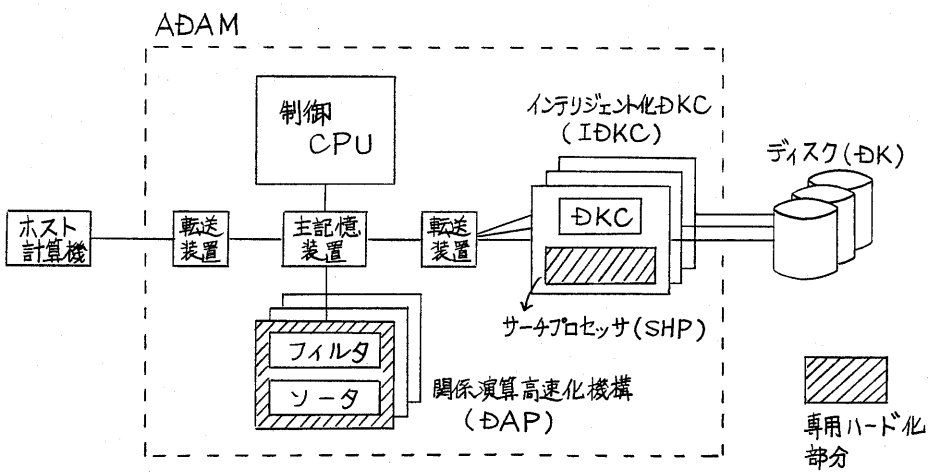


図3.1　ADAMのハードウェア構成

表4.1 I/O動作の高速化方法

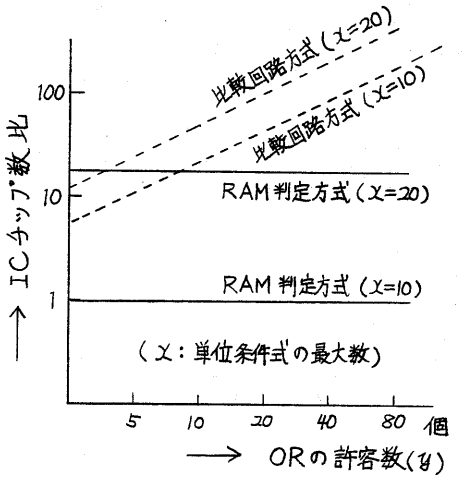
方式 / 項目	汎用ディスク並列アクセス方式	トラックパラレルアクセスディスク方式	半導体ディスク方式
概要	1つのリレーションを複数のディスクに分散格納し、並列アクセスする.	複数トラックから同時読み出し可能なディスクを使用する.	RAM、磁気バブル等の半導体素子を用いたディスクを使用する.
転送速度比	n (並列ボリューム数)	m (ディスクヘッド数)	100程度
ディスクコスト比	1	10 程度	100程度
特徴	・nは最大100程度まで可能. ・既存ハードの有効利用.	・mは最大30程度.	・高速性能を生かすためのメモリ・インタフェースが技術課題.
評価	○	×	×

表5.1 テキストサーチの実現方法

方式 / 項目	並列比較器法	直列比較器法	テーブルドリブン法
概要	入力テキストを1文字ずつシフトしながら、検索キーと比較する.	カスケード接続された1文字比較器全部に対し、テキストを1文字ずつ入力する.	有限オートマトン理論に基づいた状態遷移テーブルを使用する.
ハードウェアの単純性、回路規模	シフトレジスタ、検索キーレジスタ、比較器等多種多数のハードが必要.	単純な構造の比較セルを多数接続することが必要.	基本的な構成要素はRAMであり、論理回路はほとんど不要.
複数の検索キー、don't careを含むキーの検索	可変長don't careを含むキーによる検索は原理的に不可能.	複数の検索キーを使用する場合、セル間の接続が困難.	テーブルをうまく作ることにより、複数検索キー、可変長don't careへの対応可能.
検索キー数、検索キー長の拡張性	回路規模は、キー数、キー長の最大値の積に比例して増加.	回路規模は、キー長の最大値に比例して増加.	テーブルサイズは、検索キーのトータル長で決定される.
評価	×	×	○

表5.2 複合条件判定方法

方式 / 項目	連想メモリ方式	比較回路方式	RAM判定方式
概要	複合条件成立のパターンをそのまま連想メモリに設定する.	ラッチとゲートを組み合せた比較回路を用いる.	主加法標準形に展開した結果を設定したテーブルを使用する.
ハードウェア量	$AM(x \cdot y)$ ただし、$AM(n)$はnビットの連想メモリ.	$L(x \cdot y) + G((x+1) \cdot y)$ ただし、$L(n)$, $G(n)$はそれぞれnビットのラッチ、ゲート.	$M(2^x)$ ただし、$M(n)$はnビットのRAM.
	x: 単位条件式の最大数, y: ORの許容数		
検索実行時間	有意差なし		
検索準備時間	短い	短い	長い ただし、ディスクのシーク、サーチ時間内で完了可能.
評価	×	×	○



<算出条件>
ラッチ(L) 10 bit/chip
ゲート(G) 10 gates/chip
RAM(M) 64 kbits/chip

図5.1 主要構成ICチップ数

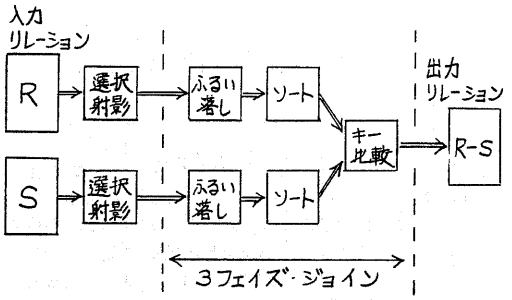
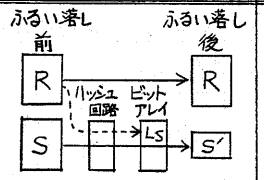
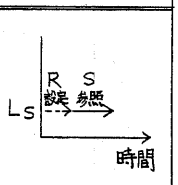
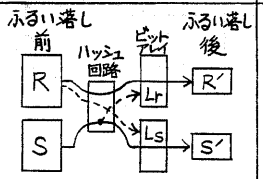
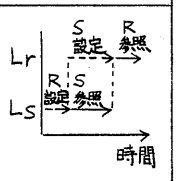
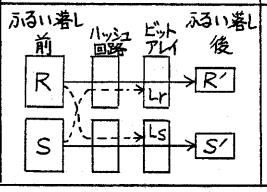
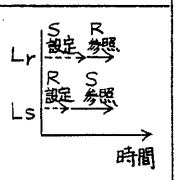
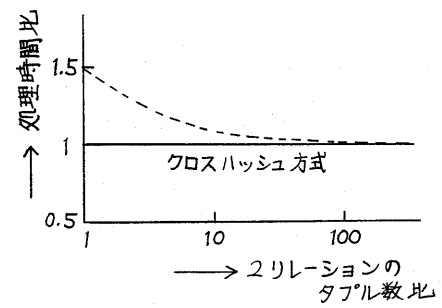


図6.1　3フェイズ・ジョイン方式の概念

表6.1　ふるい落し処理の実現方法

方式	概念図	タイムチャート
片ハッシュ		
シリアルハッシュ		
クロスハッシュ		



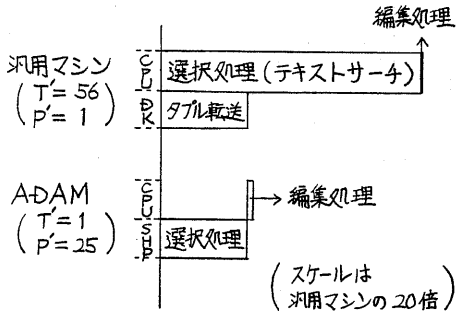
<算出条件>
・ビットアレイ設定時間＝ビットアレイ参照時間
＝ソート入力処理時間

図6.2　クロスハッシュ方式の効果

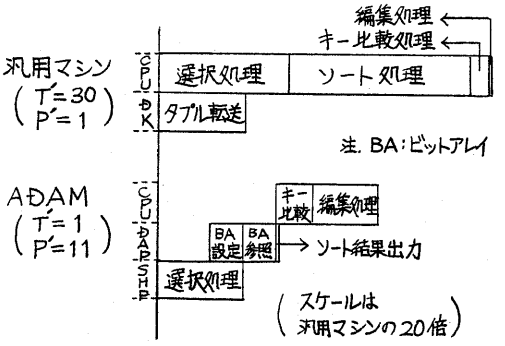
表7.1　評価モデル

項目＼モデル	モデルA	モデルB
検索対象リレーション数	1個	2個
リレーションのタプル数	20万件	各100万件
タプルの平均長	200B	200B
選択後のタプル数	2000件	10万件
ふるい落し後のタプル数	—	1万件
出力タプル数	2000件	1万件

検索処理モデルA



検索処理モデルB



T': 処理時間比
P': コスト性能比

図7.1　評価結果