データベース処理のための専用サーチ・プロセッサ†

佐藤和洋†† 中村史朗†† 吉田郁三††
熊谷多加史††† 石塚拓雄†††

本論文では，実現性に重点を置いて筆者らが提案する，低価格で，かつ単純な制御構造からなるデータベース処理専用プロセッサについて論じる。データベース処理でとくに重要な処理は検索であり，ここでは，リレーショナル・モデルの選択 (selection)，制限 (Restriction)，および射影 (projection) 操作に相当する処理からなるものを対象とする．提案プロセッサは，上記操作を，2次記憶装置からのデータ転送に同期して同時処理することにより，高速な検索処理を行う．専用プロセッサの効果を確認するために，上記演算を含む問合せ処理に要する処理時間を，実験システム IFC (integrated file controller) を用いる場合と比較し，本提案サーチ・プロセッサを用いることにより，大幅な処理時間短縮が期待できることを示す．

1. はじめに

新しい情報処理システムの形態として，専用マシンを有するシステムが考えられており，LSI 技術の進歩と相まって，各種の専用マシンの研究開発が行われている．このなかで最も期待されているものの一つがデータベース (以下，DB と略) マシンである．DB マシンは，DB 処理機能を専門に分担することにより，高性能 DB システムの実現を図り，情報システム全体としての性能向上を目指すものである．半導体技術の進歩[11]を背景に，大規模 DB の高速処理，高水準 DB システムの性能向上，などの要請に伴い，DB マシンの研究開発は活発である[1),2),5),7),16)~19)]．

DB マシンには，1) ホスト計算機の負荷軽減を狙ったソフトウェア後置型[6),10),20)]，2) I/O コントローラに DB 機能をもたせた知能コントローラ型[3),13),19)]，3) プロセッサとメモリからなるセルの線型配列による並列処理を指向した論理セル型[12),14),15)]，4) DB 処理を機能的に専用のプロセッサの複合体で実現するハードウェア後置型[4),9)]，など種々のタイプがある．製品レベルでは，1) および 2) がいくつかあるだけで，3) および 4) は，研究レベルでは多いが経済性および技術的な面で実現性に問題があり，製品化はまだ困難である．

本稿では，実現性および高性能化を考慮し，低価格かつ単純な制御構造の 2) のタイプのマシンを提案する．本マシンは，2次記憶からのデータ転送時間を有効利用し，高速検索を行う DB 専用のサーチ・プロセッサである．現在，機能確認のための試作が完了している．

DB 処理でとくに重要なのは検索処理である．ここでは，関係モデル[8)]の選択 (selectin)，制限 (Restriction)，および射影 (projection) の各演算に相当する処理を対象とする．提案プロセッサは，上記演算 (ただし，射影における重複排除は除く) を，2次記憶装置からのデータ転送に同期して同時処理し，高速検索処理を実現する．上記演算は，関係 DB でとくに処理時間を要する結合 (join) 演算の前処理として機能するものでもあり，上記演算の高速化は重要である．以下，2章では処理対象の演算を，3章ではサーチ・プロセッサのシステム概要を述べ，最後に，本方式の効果を確認するため，その机上評価について論じる．

2. データベース演算の表現

データベースの検索にあたっては，リレーショナルデータモデルにおける演算“選択 (selection)”および“制限 (restriction)”に相当する処理が支配的である．すなわち，検索条件 Q は，選択

$$R_D.\gamma_a\ \theta\ C_a$$

および制限

$$R_D.\gamma_b\ \theta\ R_D.\gamma_c$$

のいずれかを p_j^i とするとき，次のような積和標準形で表されるものである：

$$Q\equiv(p_1^1\wedge p_2^1\wedge\cdots\wedge p_{n_1}^1)\vee\cdots\vee(p_1^l\wedge p_2^l\wedge\cdots\wedge p_{n_l}^l)$$

ここに，各記号の意味は以下のとおりである．

$R_D.\gamma_a$：データベース D のレコード R_D の a 番目の

† Dedicated Search Processor for Database Processing by KAZUHIRO SATO, FUMIO NAKAMURA, IKUZO YOSHIDA (Systems Development Laboratory, Hitachi Ltd.), TAKASHI KUMAGAI and TAKUO ISHIZUKA (Kanagawa Works, Hitachi Ltd).

†† (株)日立製作所システム開発研究所
††† (株)日立製作所神奈川工場

フィールド ($R_D.\gamma_b$, $R_D.\gamma_c$ も同様の意味)

θ: 比較演算子 $\{<, \leq, =, \neq, \geq, >\}$ のいずれか一つ

$\wedge$: 論理演算子 AND

$\vee$: 論理演算子 OR

C_a: $R_D.\gamma_a$ と比較する具体値

次章以降，上記形式の検索条件の効率的な処理方式について述べる．

3. システム概要

本論文で提案するサーチ・プロセッサの目指すところは，主として次の点にある．

● 記憶媒体（磁気ディスク，磁気ドラム等）から転送されてくるデータストリームに同期して，その転送時間中に，検索条件を満足するデータの取得を行うこと（転送時間の有効利用）．

● サーチ・プロセッサ内での検索条件処理，ホスト側へのデータ転送量の削減等によるホスト計算機の負荷軽減を図ること（機能分散によるホスト計算機の有効利用）．

● サーチ・プロセッサの一構成要素であるサーチ・モジュールの並列化による複数の問合せ処理をサポートすること（検索処理の多重化によるシステム処理効率の向上）

全体のシステム構成を**図1**に示す．図において，各要素は次の機能を分担する（サーチ・プロセッサの詳細については，次章で述べる）．

1) ホスト計算機：ユーザからの問合せを受理し，その構文解析，意味解析を行い，第2章で述べた検索条件を切り出し，必要な情報を付加し，チャネルに転送する．さらに，チャネルから転送されてくる処理結果をユーザに返す．

2) マスタ・コントローラ：ホスト計算機から転送された検索条件式のサーチ・プロセッサへの転送と，サーチ・プロセッサからの処理結果のホスト側への転送を制御する．また，更新処理の制御も行う．

3) サーチ・プロセッサ：転送されてきた検索条件をサーチ・コントローラの制御の下で，一つのサーチ・モジュールに転送する．サーチ・モジュールはこの条件に基づいて，2次記憶上にあるデータをアクセスし，実際の検索処理を行い，結果を自身のローカルメモリ内に蓄える．

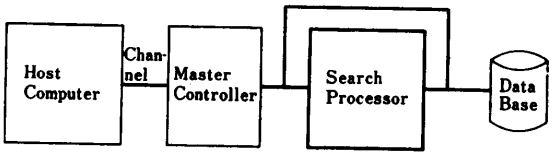


図 1 システム構成

Fig. 1 System organization.

4. サーチ・プロセッサの構成

4.1 基本構成

図2にサーチ・プロセッサの構成を示す．

(1) サーチ・モジュール：検索条件に基づいて，内容検索を実行する．

(2) サーチ・コントローラ：各サーチ・モジュールの実行制御を行う．

(3) データ転送ユニット：転送データのエラー検出，分配等を行う．

次節で，サーチ・モジュールの詳細を述べる．

4.2 サーチ・モジュール

4.2.1 サーチ・モジュールの機能

サーチ・モジュールは以下の機能を有する．

● マスタ・コントローラから転送されてくる検索条件を受理し，この情報を用いて，サーチ・モジュールを構成する各ユニットを初期化する機能．

● 2次記憶装置（データベース）から転送されてくるデータ・ストリームに同期して，選択，制限の各処理を行い，かつ，データの必要フィールドのみをメモリに格納する（これを擬射影 (pseudo-projection) ということにする）処理を，同時実行する機能．

● 検索処理が終了あるいは，サーチ・モジュールのメモリが満杯で終了の場合，自身の状態フラグ（空フ

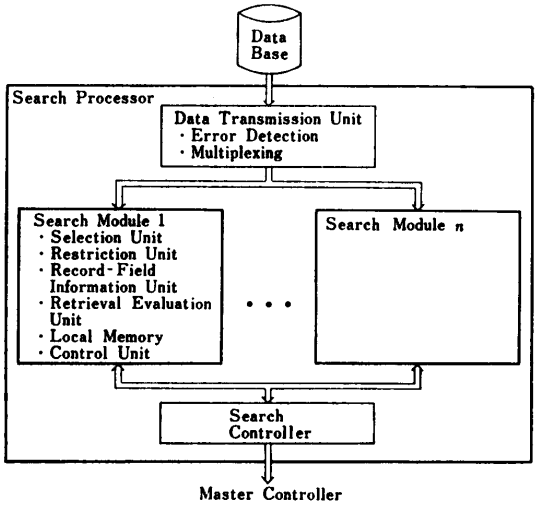


図 2 サーチ・プロセッサの構成

Fig. 2 Search processor structure.

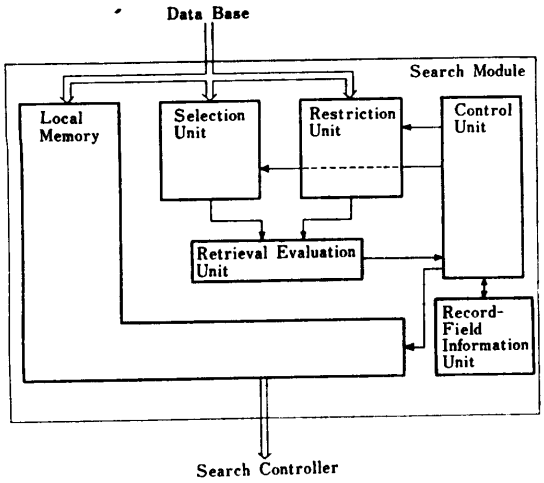
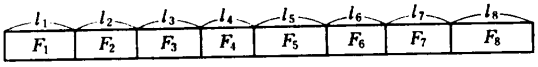


図 3 サーチ・モジュールの構成

Fig. 3 Search module structure.



F_i: フィールド名, l_i: フィールド i のフィールド長

図 4 レコード形式

Fig. 4 Record fomat.

検索条件: $(F_1 \geq C_1) \wedge (F_3 > F_5) \vee (F_6 \leq C_2) \wedge (F_4 > F_5)$
求めるデータ: フィールド F_1, F_3, F_4, F_8.

図 5 問合せの例

Fig. 5 Query example.

ラグ，実行中フラグ，満杯終了フラグ，検索終了フラグ）をセットする機能．

● サーチ・コントローラの制御の下で，メモリからマスタ・コントローラにデータを転送する機能．

サーチ・モジュールは，データ・ストリームに同期して，選択，制限および擬射影を直接ハードウェアで実行しており，データの転送時間を有効に利用した高速処理を実現するとともに，サーチ・モジュール内のメモリの有効利用を図っている．

4.2.2 サーチ・モジュールの構成および動作

● サーチ・モジュールの構成を**図 3** に示す．サーチ・モジュールは，次の六つのユニットからなる．

- Record-Field Information Unit
- Selection Unit
- Restriction Unit
- Retrieval Evaluation Unit
- Local Memory
- Control Unit

以下，おのおののユニットの機能および動作について述べる．まず，データベース・レコードは**図 4** に示すフィールド構成とする．ここで，レコードおよびフィールドは固定長とする．これらのレコードに対して，**図 5** に示す問合せの処理について考える．

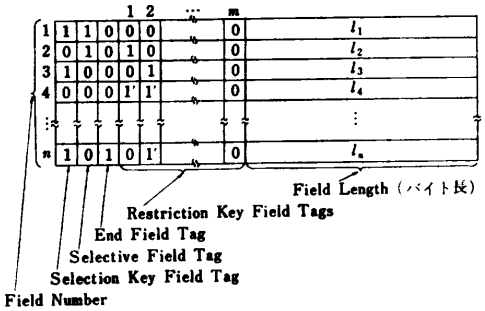


図 6 フィールド情報の構成

Fig. 6 Field information structure.

	1	2					
1	1	1	0	0	0	0	l_1
2	0	1	0	1	0	0	l_2
3	0	0	0	1′	0	0	l_3
4	0	1	0	0	1	0	l_4
5	0	0	0	0	1′	0	l_5
6	0	0	0	0	0	0	l_6+l_7
7	1	1	1	0	0	0	l_8

図 7 図 5 の問合せ例のフィールド情報

Fig. 7 Field information for the query example in Fig. 5.

サーチ・モジュールは，問合せに関する情報を自身のローカル・メモリに取り込む．次に，Control Unit はこの問合せ情報を用いて，以下の各ユニットに条件データをセットする．

1) Record-Field Information Unit には，アクセスすべきレコードの各フィールドに関する情報をセットする．この情報を**図 6** に示す．図中，縦の 1, 2, …, は，レコードの修正フィールド番号（元のレコードのフィールドにおいて，問合せに関与しないフィールドが連続する場合には，それらを一つにまとめて 1 フィールドとする）を表す．この番号は，レコードの先頭からフィールドの発生順に対応して付けられる．横の 1, 2, …, は，制限において使用するレジスタを表す．また $l_1, l_2, \cdots, l_n$ は，各修正フィールドの長さ（バイト数）を表す．**図 7** に，図 5 に対する Record-Field Information Unit の内容を示す．図 6 に示す各タグは，それぞれ次の役割をもつ．

① Selection key field tag: これは，レコードの各フィールドが選択に関与しているか否かを示すものである．このタグが "1" の場合には，対応するフィールドが選択に関与しており，"0" の場合は，関与していないことを表す．図 7 の例では，選択に関与するフィールド F_1, F_8 に対応して，修正フィールド 1, 7 が

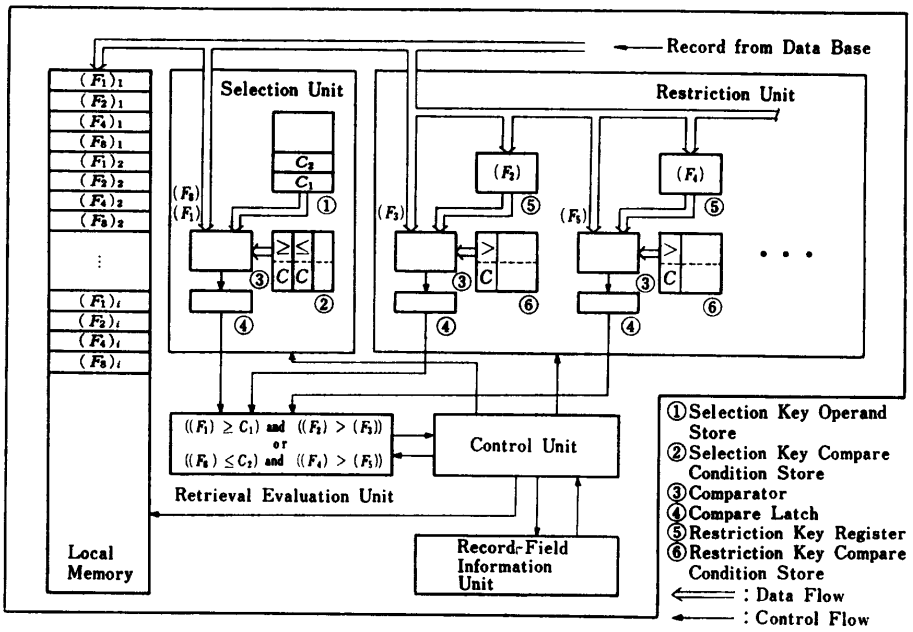


図 8 図5の問合せ例の処理概要

Fig. 8 Processing outline for the query example in Fig. 5

"1" にセットされている.

② Selective field tag: これは, 各フィールドをサーチ・モジュールのメモリ (local memory) に格納すべきか否かを示すものである(擬射影). このタグが "1" の場合は, メモリに格納し, "0" の場合は, 格納しない. 図7の例では, 修正フィールド 1, 2, 4, 7 に対応するフィールド F_1, F_2, F_4, F_8 をメモリに格納することを示している.

③ End field tag: これは, レコードの最後の修正フィールドを示すものである. 図7の例では, 修正フィールド7, すなわち, フィールド F_8 が最終修正フィールドであることを示している.

④ Restriction key field tag: これは, 制限に関与するフィールドを明示する. 後述する Restriction key Registers の数 (m 個) に対応して, m 個のカラムが本タグ用に割り当てられている. 各制限条件は, フィールド番号の若い順に並べられ, 初めのフィールドに対し "1" を, 後のフィールドに対し "1′" を, タグカラムの対応する修正フィールド位置にセットする. そのとき, 初めのフィールドが同一名称である制限条件が複数個存在する場合は, 同一のカラムが使用される. 図4の $m, n(m \leq n)$ に対しては, 最大 $m(2n-m-1)/2$ 個の制限が扱える. 図7の例では, 制限 $F_2 > F_3, F_4 > F_5$ に対応して, 修正フィールド2に "1", 3に "1′", 4に "1", 5に "1′" がセットされている. また,

2) Selection Unit は,

- Selection Key Operand Store
- Selection Key Compare Condition Store
- Comparator
- Compare Latch

で構成されている. このうち, 次の要素に条件データがセットされる.

① Selection Key Operand Store: 選択に関与するフィールドに対応する比較値.

② Selection Key Compare Condition Store には選択に関与するフィールドに対応した比較演算子, 比較データタイプ等.

3) Restriction Unit は,

- 複数の Restriction Key Register
- 複数の Restriction Key Compare Condition Store
- 複数の Comparator
- 複数の Compare Latch

で構成されている. このうち, Restriction Key Compare Condition Store に, 制限に関与するフィールド間の比較演算子, 比較データタイプ等の情報がセットされる.

さらに,

4) Retrieval Evaluation Unit には, 選択および

制限からなる検索条件に対応する評価式をセットする.

以上, 1), 2), 3), 4)の初期設定後に, サーチ・モジュール内の各ユニットは, 2次記憶装置からの転送データに対して処理を開始する. まずSelection Unitでは, 記憶装置からのデータ転送に同期して,

● Record-Field Information Unitのフィールド情報

● 上記各 Store 内の比較情報

に基づいて, Comparator により該当フィールドの比較を行う. 比較結果の真偽は, 比較フィールドごとにCompare Latch にセットする. 1レコードの全選択処理の終了後, ラッチデータが論理演算ユニットである Retrieval Evaluation Unit に転送される. このSelection Unit を多重にすることにより, 一つのフィールドに対して複数の選択処理が可能となる.

同様に, Restriction Unit では, Restriction key Compare Condition Storesの比較情報および, Record-Field Information Unit のフィールド情報に基づいて, 2次記憶装置からのデータ転送に同期しながら制限処理を行う. すなわち, 上記1)で述べた④の Restriction key field tags で "1" がセットされているフィールドに対応するデータが転送されてきたら, 一つの Restriction Key Register にそのデータをセットする. また, 当該フィールドに "1" が存在する場合には, 比較すべきデータがセットされている Restriction Key Registers (複数存在可能) を決定し, これらのレジスタ内のデータと転送されているデータを対応する Restriction Key Compare Condition Store の情報に基づいて, 各対応する Comparator で比較する. 比較結果の真偽は, 制限処理ごとに Compare Latch にセットする. 1レコードの全制限処理終了後, ラッチデータが Restriction Unit から Retrieval Evaluation Unit に転送される.

以上, Selection Unit および Restriction Unit における比較処理結果に基づいて, Retrieval Evaluation Unit では論理演算を行い, 検索条件を満足するレコードかの判定を行う. この情報に従い, Control Unit は, Local Memory のアドレス制御を行う. すなわち, Local Memory には, Record-Field Information Unit で述べた Selective Field tag 情報に基づいて, 上述比較処理に並行し, 必要フィールドデータが格納されており, Retrieval Evaluation Unit の出力が "真" の場合には当該データを保持し, "偽" の場合にはキャンセルする. 図8に, 図5に示した例に対する, サーチ・モジュールの処理概要を各ユニットの構成要素を用いて示した.

上述した処理を転送レコードに繰り返し適用し, 当該 Local Memory が満杯になった場合には, 検索処理を停止し, 状態フラグを "メモリ満杯終了フラグ" にセットする. このとき, 再開始情報も保持しておく. この状態で, サーチ・コントローラからデータ転送指示があると, サーチ・モジュールはデータを転送し, その終了後, その旨サーチ・コントローラに報告するとともに, 自身の状態フラグを "実行中フラグ" にセットし, 再開始処理情報を用いて処理を始める. また, サーチ・モジュールの状態フラグが "検索終了フラグ" の場合には, メモリからのデータ転送終了後, 状態フラグを "空フラグ" にセットし, サーチ・コントローラからの検索条件の転送を待つ. 検索条件が転送されてきたら, サーチ・モジュールは前述同様に, 各ユニットの初期設定を行い, 処理を開始する.

以上がサーチ・モジュールの機能および動作概要である.

4.3 サーチ・コントローラの動作概要

サーチ・コントローラは, サーチ・プロセッサ全体の制御を行い, とくに次の動作が重要である.

サーチ・モジュールには, 自身の状態を表す状態フラグ (空, 実行中, メモリ満杯終了, 検索終了) がある. サーチ・コントローラは, 周期的にサーチ・モジュールの状態フラグを監視する. 以下, 各状態フラグにおけるサーチ・コントローラの動作を示す.

● 空フラグの場合: サーチ・コントローラは, マスタ・コントローラに検索条件の転送を要求し, もし, 検索条件が存在すれば, 当該サーチ・モジュールにこの条件を転送する. 存在しない場合には, 次のサーチ・モジュールの状態フラグを調べる.

● 実行中フラグの場合: 次のサーチ・モジュールを調べる.

● メモリ満杯終了フラグ: 当該サーチ・モジュールにデータ転送指示を出し, マスタ・コントローラへのデータ転送の制御を行う. データ転送終了後, 次のサーチ・モジュールを調べる.

● 検索終了フラグ: メモリ満杯終了フラグと類似の動作をするが, データ転送後の処理が異なる. この場合は, データ転送終了後は, 空フラグ状態と同じ動作を行う.

以上がサーチ・コントローラの動作概要である.

5. サーチ・プロセッサの性能評価

本章では，解析式をベースにサーチ・プロセッサの処理時間の推定・評価を行う．評価に当たり，下記3ケースを設定し，比較する．

ケース 1: サーチ・プロセッサを用いる場合．単純化のために，サーチ・モジュール数は1とする．

ケース 2: IFC (Integrated File Controller)[19] を用いる場合．IFCは磁気ディスク制御装置を改造し，選択およびこれらの論理演算を転送データストリームに同期した形で実行できるようにした実験システムである．本ケースは，このIFCを用い，制限，射影処理はホスト計算機で実行するものとする．

ケース 3: すべての処理をホスト計算機で処理する場合である．

5.1 処理時間推定式

以下のように検索処理時間を推定する．

(1) 前提: 処理時間推定は，いわゆる問合せの構文解析等の前後処理に要する時間は含まず，選択，制限および射影の各処理に要する時間と，各プロセッサ間のデータ転送時間等を考え，これらの総和という形でとらえる．

(2) 処理時間を決定する処理コンポーネントおよび処理時間: **表1,2**に，使用するパラメータと，各処理コンポーネント対応の処理時間を各ケースについて示した．

5.2 処理時間評価結果

5.2.1 パラメータの設定

表1に示したパラメータに対し，**表3**に示すように具体値を設定した．設定に当たって，2次記憶として日立のH-8595磁気ディスク装置，ホスト計算機として中型のcpuを想定し，また，サーチ・モジュール内での処理に関しては，1マイクロ命令=200 nsとし，マイクロ命令ステップ数からおのおの関連するパラメータを設定した．

5.2.2 処理時間の評価

各ケースの処理時間を下記項目について検討し，その結果を**図9**に示した．

(イ) n: データ（レコード）数（図9の(a)）

(ロ) α_s: Selection Selectivity（図9の(b)）

(ハ) α_r: Restriction Selectivity（図9の(c)）

(ニ) α_p: Projection Selectivity（図9の(d)）

各図ともサーチ・プロセッサの効果を表している．ただ，各Selectivityが小さい場合には，ケース1とケース2はそれほど違いがなく，サーチ・プロセッサの効果は少ない．すなわち，選択処理機能だけでも大きな効果をもつことを示している．

6. むすび

本論文では，データベース処理においてとくに重要な検索処理を高速に行う専用サーチ・プロセッサを提案し，解析的方法により，既存システムとの比較評価を行った．その結果，本提案サーチ・プロセッサにより，大幅な処理時間短縮が期待できることを示した．

本稿では，より複雑な問合せ処理およびサーチ・モジュールによる並列処理等については触れなかった．これらは今後の課題である．また，本稿では，検索専用プロセッサの提案を主としたため，更新処理等については論じなかった．更新処理(追加，置換，削除)においては，各種制御情報に基づいたストレージ管理をサポートする必要があり，更新処理の高速化はむずかしい．ただ，固定長データの置換に関しては，本サーチ・プロセッサに更新条件を満たすデータの物理アドレス保持機能およびアドレスソート機能などを装備することにより置換処理の高速化が図れると考える．詳細検討は今後の課題である．

表1 記号一覧

Table 1 Symbols.

n	レコード数	t_{sort^h}	ホスト計算機におけるデータソート処理時間 (sec/バイト)
r	レコード長 (バイト)	t_{comp^h}	ホスト計算機におけるバイト比較処理時間 (sec/バイト)
X	Local Memory サイズ (バイト)	t_{move^h}	ホスト計算機におけるバイト移動格納時間 (sec/バイト)
$b_t(b_c)$	トラック（シリンダ）当りのバイト数	t_p	ホスト計算機におけるトータルミックス値 (sec/ステップ)
α_s	Selection Selectivity (セレクションを満たす率)	T_l	DASD のシリンダ位置決め時間 (平均) (sec)
α_r	Restriction Selectivity (リストリクションを満たす率)	T_r	DASD の回転待ち時間 (平均) (sec)
α_p	Projection Selectivity (プロジェクションされるフィールド率)	B_s	サーチ・モジュール Local Monory 満杯時に要するステップ数
α_{del}	レコード集合の非重複率	h_{or}	条件チェック時の平均 OR チェック数
α_{rl}	i 番目 Local Memory 満杯レコードの Restriction Selectivity	h_{and}	条件チェック時の平均 AND チェック数
t_s	DASD→サーチ・モジュールへのバイト転送時間 (sec/バイト)	ρ	セレクションおよびリストリクションに関与する平均フィールド長
t_t	サーチ・モジュール→ホストへのバイト転送時間 (sec/バイト)	t_{f^h}	ホスト計算機におけるバイトフェッチ転送時間 (sec/バイト)
t_{f^L}	Local Memory 内データのバイト転送時間 (sec/バイト)	t_{p^h}	ホスト計算機におけるサブレコード作成時間 (sec/バイト)

表 2 処理コンポーネントとその処理時間

Table 2 Processing components and their processing times.

ケース 1		ケース 2		ケース 3	
処理コンポーネント	処理時間	処理コンポーネント	処理時間	処理コンポーネント	処理時間
磁気ディスクのシーク・サーチ処理	$T_1\cdot\left\lceil\frac{n}{\lfloor b_c/r\rfloor}\right\rceil+T_r\cdot\left(\left\lceil\frac{n}{\lfloor b_t/r\rfloor}\right\rceil+\left\lfloor\frac{\alpha_p\alpha_r\alpha_s nr}{x}\right\rfloor\right)$	磁気ディスクのシーク・サーチ処理	ケース 1 の $\left\lfloor\frac{\alpha_p\alpha_r\alpha_s nr}{x}\right\rfloor$ を $\left\lfloor\frac{\alpha_s nr}{x}\right\rfloor$ に置換した式	磁気ディスクのシーク・サーチ処理	ケース 1 の $\left\lfloor\frac{\alpha_p\alpha_s\alpha_r nr}{x}\right\rfloor$ を除いた式
磁気ディスクからサーチ・モジュールへのデータ転送処理	$n\cdot r\cdot t_s$	磁気ディスクから IFC へのデータ転送処理	ケース 1 に同じ	磁気ディスクからホスト計算機へのデータ転送処理	ケース 1 に同じ
サーチ・プロセッサからホスト計算機へのデータ転送処理	$\alpha_p\cdot\alpha_r\cdot\alpha_s\cdot n\cdot r(t_f^L+t_t)$	IFC からホスト計算機へのデータ転送処理	$\alpha_s\cdot n\cdot\gamma\cdot(t_f^L+t_t)$		
データのソート処理（バブルソート使用：プロセッサメモリ満杯ごとに行う）	$\frac{1}{2}\{X\cdot Y\cdot(Y-1)+(\alpha_r\cdot\alpha_s\cdot n-X\cdot Y)(\alpha_r\cdot\alpha_s\cdot n-X\cdot Y-1)\}\alpha_p\cdot r\cdot t_{sort}^h$ ここで $X=X=\left\lfloor\frac{\alpha_p\alpha_r\alpha_s nr}{x}\right\rfloor$, $Y=\left\lfloor\frac{x}{\alpha_p r}\right\rfloor$ （以下同様に用いる）	ホスト内での検索条件式の評価	$\alpha_s\cdot n\cdot\rho\cdot h_{or}\cdot l_{and}\cdot t_f^h$	ホスト内での検索条件式の評価	$n\cdot\rho\cdot h_{or}\cdot l_{and}\cdot t_f^h$
		ホスト内での擬射影処理	$\alpha_p\cdot\alpha_r\cdot\alpha_s\cdot n\cdot r\cdot t_p^h$	ホスト内での擬射影処理	$\alpha_p\cdot\alpha_r\cdot\alpha_s\cdot n\cdot r\cdot t_p^h$
サーチ・モジュールのメモリ満杯終了により転送されたデータとすでにホスト計算機メモリにあるデータとのマージソート処理（このプロセスで重複排除処理も行う．また主メモリは十分大きくすべてのデータを収容できるものとする）	$\frac{1}{2}\Bigg[\Bigg\{Y\cdot\sum_{k=1}^{X-1}\min\Bigg(1,\sum_{j=1}^{k}\Bigg(\prod_{l=j}^{k}\alpha_{del}\Bigg)\Bigg)+\min\Big((\alpha_r\alpha_s n-X\cdot Y),\ Y\cdot\sum_{j=1}^{X}\Bigg(\prod_{l=j}^{X}\alpha_{del}\Bigg)\Big)\Bigg\}\alpha_p\cdot r\cdot t_{comp}^h+\Bigg\{Y\cdot\Bigg(X+\sum_{k=1}^{X-1}\sum_{j=1}^{k}\Bigg(\prod_{l=j}^{k}\alpha_{del}\Bigg)\Bigg)+\Bigg(\alpha_r\alpha_s\cdot n-X\cdot Y+Y\cdot\sum_{j=1}^{X}\Bigg(\prod_{l=j}^{X}\alpha_{del}\Bigg)\Bigg)\Bigg\}\alpha_p\cdot\gamma\cdot(t_{comp}^h+2t_{move}^h)-(X+1)\cdot\alpha_p\cdot\gamma\cdot t_{comp}^h\Bigg]$	データのソート処理（条件はケース 1 と同じ）	ケース 1 の式において X, Y を次のように置換した式 $X=\sum_{i=1}^{\lfloor\frac{\alpha_s nr}{x}\rfloor}\alpha_{ri}$ $Y=\left\lfloor\frac{x}{r}\right\rfloor$	データのソート処理（条件はケース 1 と同じで，主メモリは十分大きくすべてのデータを収容できソートできるものとする．重複排除も含む）	$\frac{1}{2}\{\alpha_r\alpha_s n(\alpha_r\alpha_s n-1)\}\cdot\alpha_p\gamma\left(t_{comp}^h+\frac{3}{2}t_{move}^h\right)$
		IFC のメモリ満杯終了により転送されたデータとすでにホスト計算機メモリにあるデータとのマージソート処理（重複排除処理含）	ケース 1 で $X=\left\lfloor\frac{\alpha_s nr}{x}\right\rfloor$, $Y=\left\lfloor\frac{x}{r}\right\rfloor$, $X=\sum_{j=1}^{X}\alpha_{rj}$, $\left(\prod_{l=j}^{k}\alpha_{del}\right)$ を $\left(\prod_{l=j}^{k}\alpha_{rj}\right)\alpha_{rj}$ に，さらに $\min(1,\cdots)$ の 1 を α_{rk} に置換した式		
サーチ・モジュールのメモリ満杯に関する処理	$X\cdot B_s\cdot t_p$	IFC メモリ満杯に関する処理	$\left\lfloor\frac{\alpha_s nr}{x}\right\rfloor\cdot B_s\cdot t_p$		

表 3 パラメータ値

Table 3 Values of parameters.

n	$10^4\le n\le 5\times10^4$	α_{ri}	$0.1\le\alpha_{ri}\le1.0$	t_p	3.0×10^{-6} (sec/バイト)
r	$100\le r\le500$ (バイト)	t_s	0.83×10^{-6} (sec/バイト)	T_1	2.0×10^{-2} (sec)
x	$2K\le x\le8K$ (バイト)	t_t	0.7×10^{-6} (sec/バイト)	T_r	8.4×10^{-3} (sec)
b_t	19069 (バイト)	t_f^L	2.5×10^{-6} (sec/バイト)	B_s	10^4 (ステップ)
b_c	572072 (バイト)	t_f^h	0.75×10^{-6} (sec/バイト)	h_{or}	2
α_s	$0.01\le\alpha_s\le0.05$	t_p^h	0.75×10^{-6} (sec/バイト)	h_{and}	4
α_r	$0.1\le\alpha_r\le1.0$	t_{sort}^h	2.0×10^{-6} (sec/バイト)	ρ	$100\le\rho\le500$ (バイト)
α_p	$0.1\le\alpha_p\le0.5$	t_{comp}^h	0.75×10^{-6} (sec/バイト)		
α_{del}	1.0	t_{move}^h	0.75×10^{-6} (sec/バイト)		

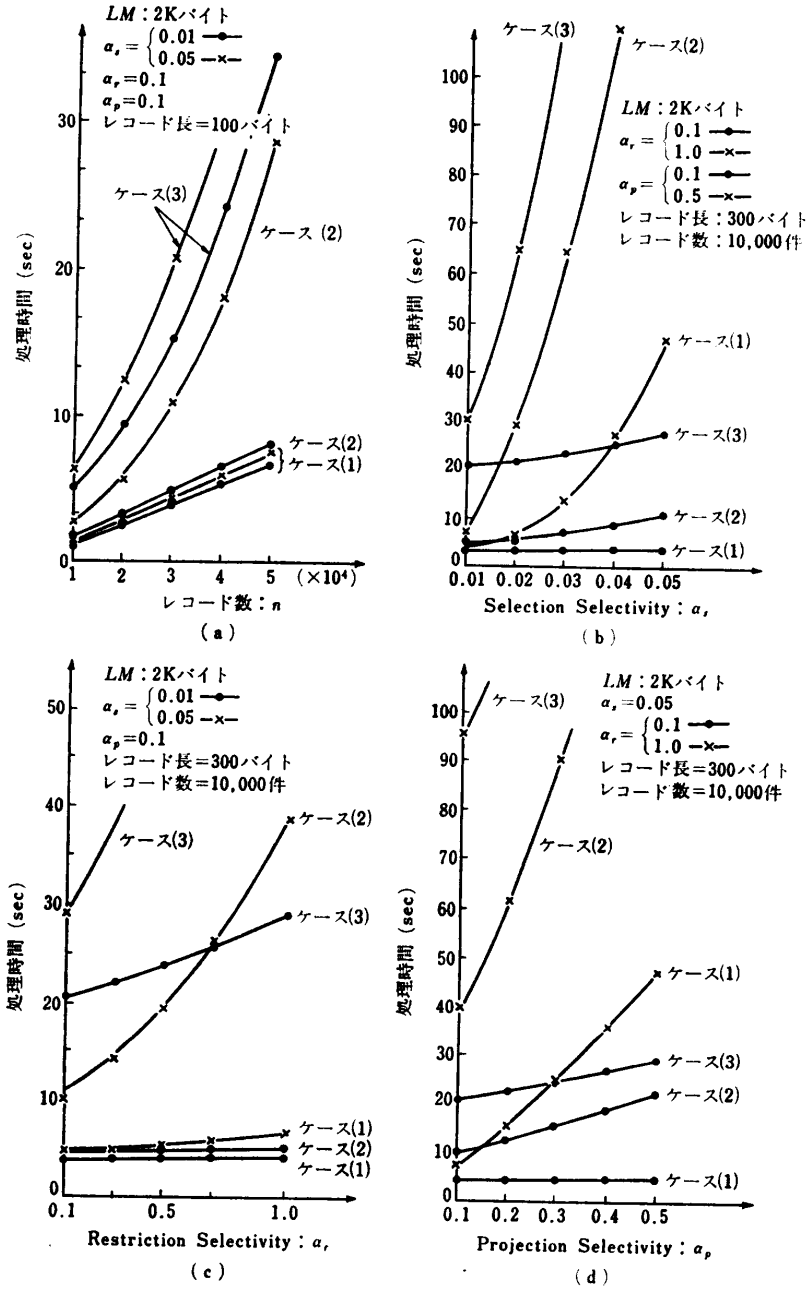


図 9 処理時間の比較

Fig. 9 Comparison of processing time.

謝辞 最後に，本研究の推進に当たり，ご指導いただいた当研究所の石原部長，また討論していただいた日立神奈川工場酒井主任技師，日立ソフトウェア工場石塚，原田両主任技師に深謝いたします．

参 考 文 献

1) 植村俊亮：データベースマシン・アーキテクチャ，電子通信学会誌，Vol. 62, No. 11, pp. 1289-1291 (1979. 11).

2) 関野：データベース・マシン，情報処理，Vol. 17, No. 10, pp. 940-946 (1976. 10).

3) Babb, E.: Implementing a Relational Database by means of Specialized Hardware, *ACM Trans. Database Syst.*, Vol. 4, No. 1, pp. 1-29 (1979. 1).

4) Banerjee, J.: Concepts and Capabilities of a

Database Computer, *ACM Trans. Database Syst.*, Vol. 3, No. 4, pp. 347-384 (1978. 12).
5) Baum, R. I,: Detabase Computers—A Step Towards Data Utilities, *IEEE Trans. Comput.*, Vol. C-25, No. 12, pp. 1254-1259 (1976. 12).
6) Canady, R. H.: A Backend Computer for Data Base Management, *Commun ACM*, Vol. 17, No. 10, pp. 575-582 (1974. 10).
7) Champine, G. A.: Current Trends in Data Base Systems, *IEEE Computer*, Vol. 12, No. 5, pp. 27-41 (1979. 5).
8) Date, C. J.: *An Introduction to Data Base Systems*, 2nd Ed., p. 536, Addison-Wesley, (1977).
9) Dewitt, D. J.: DIRECT —A Multi-Processor Organization for Supporting Relational Database Management Systems, *IEEE Trans Comput.*, Vol. C-28, No. 6, pp. 393-405 (1979).
10) Epstein, R.: Design Decisions for the Intelligent Database Machine, *Proc. AFIP NCC.*, Vol. 49, pp. 237-241 (1980).
11) Juliussen, J. E.: Bubbles and CCD Memories-Solid State Mass Storage, AFIPS NCC, pp. 1067-1075 (1978).
12) Lin, C. S.: The Design of a Rotating Associative Memory for Relational Database Applications, *ACM Trans. Database. Syst.*, Vol. 1, No. 1, pp. 53-65 (1976).
13) Maller, V. A. J.: Retrieving Information, *Datamation*, pp. 164-172 (1980. 9).
14) Ozkarahan, E. A.: RAP —An Associative Processor for Database Management, *Proc. AFIPS. NCC.*, Vol. 44, pp. 379-387 (1975).
15) Su, S. Y. W.: CASSM: A Cellular System for Very Large Databases, *Proc. Ist Int. Conf. on VLDB*, pp. 456-472 (1975).
16) Su, S. Y. W.: Database Machines and Some Issues on DBMS Standards, *Proc. AFIPS NCC.*, Vol. 49, pp. 191-208 (1980).
17) Data base Machines, *IEEE Computer*, Vol. 12, No. 3 (1979).
18) Special Issue on Database Machines, *IEEE Trans. Comput.*, Vol. C-28, No. 6 (1979).
19) 石塚：インテリジェントファイル制御機構の実験システムについて：情報処理学会計算機アーキテクチャ研究会 (1980. 9).
20) ADABAS データベースマシン：情報処理学会第22回全国大会 (1981. 3).

(昭和56年4月27日受付)
(昭和56年12月17日採録)