

汎用 DBMS の性能推定システム†

米田 茂^{††} 山口 康隆^{††}

ネットワーク構造の汎用 DBMS を対象に作成した、性能推定システム PEAL について論じる。DBMS の性能パラメータを記述した後、PEAL の主要要素である (1)ハッシング・ファイルの解析モデル、(2)磁気ディスク・サービス時間推定方法、(3)CPU 時間設定方法について定量的に議論する。

PEAL は、汎用に設計されており、他システムと比較して、データベースの初期作成から、初期作成後のデータベースの検索・更新処理に至る一連の性能推定を一貫したシステム体系の中で実現している点が特徴となっている。

性能推定に当たって、PEAL は、(1)データベースの記述言語、(2)応用プログラムの記述言語、の2種類の高級言語を用いて建設されたアプリケーション・モデルの内容に基づいて処理を行う。本稿では、アプリケーション・モデル記述言語を、具体例によって示す。また、本 PEAL の推定精度、および能力を、実測値との比較によって考察する。

1. はじめに

データベースを中心としたアプリケーション・システムの普及に伴い、データベース・マネジメント・システム (DBMS) に対し、機能的な要求ばかりでなく、性能に関する評価が重要視されてきた。多種多様な応用プログラムで共用するデータベースの利用に対応するため、計算機システム建設段階で、DBMS のデータ処理能力を把握・評価しておくことが必須と見なされている。

T.J. Teorey et al. は、ネットワーク構造のデータベースを管理する IDS^①を対象に、解析モデルによる性能推定システム DBDE を報告した^②。DBDE は、シングル環境における IDS データベース・システムの性能推定が、簡便に行えることを主目的に、開発されたものである。この結果、ワーク・ロードの記述の簡略化、およびディスク・アクセス時間を平均ディスク・アクセス時間で近似した点など、性能評価対象の厳密なモデリングを犠牲にしたシステムとなっている。

C. Hulten et al. は、上記と同様、ネットワーク構造のデータベースを管理する DBMS を対象に、シミュレーション・モデルを使用した性能推定ツール ARTS/DB を報告している^③。ARTS/DB は、そのモデリング手法からも判明するように、詳細な DBMS のモデル化を意図しており、ワーク・ロードの記述、

ソフトウェア・ハードウェア要素の記述に関し厳密なモデルとなっている。ARTS/DB の問題点は、シミュレーション手法採用による、計算機所用時間の大きである点である。

我々は、上記した、解析モデルとシミュレーション・モデルの両者の長を併用した、性能推定システムの開発を計画した。この計画の目標は解析モデルの利用上の簡便性と、シミュレーション・モデルの厳密性を性能推定システム上で実現することである。また、前記に加えて、従来の性能評価システムにおいては実現されなかった、データベースの初期作成から、データベースの検索・更新までの性能推定を1つのシステム上で実現し、性能推定の首尾一貫性を保つことも意図した。

日立製作所においては、PDM, ADM という2種の汎用 DBMS を開発・提供している。本論文では、PDM を対象として開発した、PEAL (PDM Performance Estimation and Administration Library) について、その主要事項、すなわち、(1)解析モデル、(2)処理時間推定方法、および、(3)PEAL 実現形態について報告する。

2. 解析モデル

2.1 PDM

PDM は、マスタ・データ・セット (MDS) とバリエーション・データ・セット (VDS) の2つのデータセット編成方法を持っている。MDS は複数の VDS と接続可能であり、VDS は複数の MDS と接続可能である。この機能により、PDM は、ファイル・ネットワーク構造のデータベースを実現する。

† A Performance Estimation System for General Purpose DBMS by SHIGERU YONEDA and YASUTAKA YAMAGUCHI (Systems Development Laboratory, Hitachi, Ltd.).

†† (株)日立製作所システム開発研究所

データベース処理性能に顕著に影響を及ぼす、(1)ランダムマイジング手法、(2)リンケージ・パスについて説明する。PDMの基本的な用語を付録に示した。

(1) ランダムマイジング手法

MDSは、ユーザが指定したキーをランダムマイズ(ハッシュ)することによってアクセスされる。指定されたキーは、PDM内部のアルゴリズムによってデータ・セットの先頭から番号付けられた相対レコード番号(RRN: Relative Record No)に変換される。このランダムマイジングの結果、異なるキーが必ずしも、異なるRRNに変換されるとは限らない。この現象をシノニム(コリジョン)が発生したといい、そのレコードをシノニム・レコードと呼ぶ。シノニム・レコードは、それが本来登録されるアドレス、すなわち、ホーム・アドレス中のレコードと互いにポインタを結ぶことによって管理される。シノニムが発生した場合、ホーム・アドレスを含むブロック中に空エリアがある場合、当該エリアに登録し、ブロック中に空エリアがない場合、同一シリンダ中の空きエリアをシリアルに検索し、この結果発見された空エリアに、シノニム・レコードを登録する。

(2) リンケージ・パス

MDSとVDSは、リンケージ・パスによって接続される。(PDMでいうリンケージ・パスは、その概念、および機能に関して、CODASYLのセット・タイプとほとんど同様である。このとき、オーナは常にMDSで編成されるデータ・セットで、メンバは常にVDSで編成されるデータ・セットである)。1つのVDSに複数のリンケージ・パスが定義されたとき、1つのリンケージ・パスをベース・リンケージ・パスといい、他のリンケージ・パスをセカンダリ・リンケージ・パスという。ベース・リンケージ・パス処理において、同一マスタ・レコードに複数のバリエブル・レコードが接続される場合、該当マスタ・レコードのリスト内(CODASYLでいうセット・オカレンス)で、データのローカリティを保つため、基本的には、同一のブロック、あるいは同一のシリンダ内にバリエブル・レコードを登録する。一方、セカンダリ・リンケージ・パス処理においては、既にベース・リンケージ・パス処理において、レコードの物理的な位置付けは定まっており、この結果、該当マスタ・レコードのリスト内で、バリエブル・レコード群のデータのローカリティはほとんどなく、データセット内に分散されて登録される。

2.2 解析モデルの開発

我々の性能推定システムPEALの核となっている解析モデルについて、MDSに対する解析モデルの要点を述べる。

説明を容易にするため、ブロッキング・ファクタが1の場合について、MDSの解析モデルを述べる²⁾。

N 個のアドレス・スペースを持つデータ・セットに n 個のレコードをランダムマイズするとする。このとき、1つのアドレスに着目し、そこに k 個のレコードがランダムマイズされる確率 P_k は、(1)の2項分布、およびその極限として、 $\rho (=n/N)$ をパラメータとする、(2)のポアソン分布で近似されることが良く知られている。

$$P_k = {}_n C_k (1/N)^k ((N-1)/N)^{n-k} \quad (1)$$

$$P_k(\rho) = \rho^k \times \exp(-\rho) / k! \quad (2)$$

上記、式(2)を用いると、平均シノニム・リスト長 l 、およびランダム検索の平均アクセス回数 a は、次式(3)、および(4)で各々表現される⁶⁾。

$$l = \sum_{i=1}^{\infty} i P_i(\rho) / (1 - P_0(\rho)) \\ = \rho / (1 - \exp(-\rho)) \quad (3)$$

$$a/\rho = \sum_{i=1}^{\infty} P_i(\rho) + 2 \times \sum_{i=2}^{\infty} P_i(\rho) + \dots \\ + k \times \sum_{i=k}^{\infty} P_i(\rho) + \dots \\ = \sum_{i=1}^{\infty} (i^2 + i) P_i(\rho) / 2 = \rho + \rho^2 / 2 \quad (4)$$

次に、MDSに、レコードを1件追加するときの解析モデルを考える。

(1) Q空間の導入

今ランダムマイズされたレコードに対し、次の操作を行う。

i) ホーム・レコードには1、スペース・エリアには0のラベルを貼る。

ii) シノニム・レコードには、シノニム・ポインタの逆方向から、2, 3, ... とラベルを貼る。

iii) シノニム・レコードをラベル0のエリアに移す。

上記操作を行った後(図1)、 Y を任意のアドレス・スペースに貼られたラベルの値をとる確率変数とし、 Y が値 k をとる確率を Q_k とする。 Q_k の $k=0, 1, 2, \dots$ に対応し、その物理的な意味を次に示す。

Q_0 : スペースの存在する確率

Q_1 : ホーム・レコードの存在する確率

Q_{i+1} : シノニム・ポインタの後方から i 番目に存在

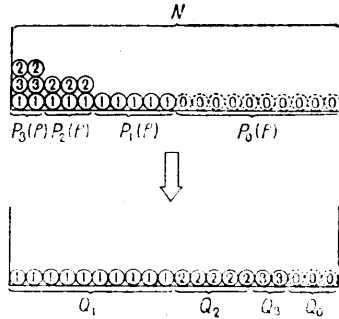


図1 Q空間
Fig. 1 Q-SPACE

するシノニム・レコードの存在する確率

各 Q_k は、ポアソン確率 $P_i(\rho)$ を用いると、次式で表現される。

$$\left. \begin{aligned} Q_0 &= 1 - \rho \\ Q_k &= \sum_{i=k}^{\infty} P_i(\rho) \quad (k=1, 2, \dots) \end{aligned} \right\} \quad (5)$$

(2) 1件のレコード追加の解析モデル

Q空間と、ポアソン分布の初等的な操作によって、MDSに1件レコードを追加するとき発生する入出力回数が求まる。MDSへのレコード追加手続に従って、入出力回数の推定手順を図2に示した。

3. 処理時間の推定方法

解析モデルによって、PDM 処理の制御方法を確率的に定めることができ、この結果、入出力回数が推定される。本章では、データベース処理時間を推定するため、次に要求される(1) CPU 時間、(2) ディスク・サービス時間の設定方法について述べる。

3.1 CPU 時間

特定のデータ操作言語処理で必要となるCPU時間の算出式は、例えば、 $(aX + bY + cZ + d)$ として表現される。ここで、 a, b, c, d は、指定されたデータ操作を行うに際して、途中の繰り返し処理を行うとき1回の繰り返しで必要となるCPUの単位時間である。一方、 X, Y, Z は、例えば、シノニム・レコード検索による繰り返し回数、空きエリアの探索による繰り返し回数など、データベース定義、およびデータ操作実行段階に解析モデル、によって決定される変数である。

上記 a, b, c, d など CPU 時間は、命令語トレーサを利用して、概略以下に示す方法で設定

した。

(1) PDM を命令語トレーサ下で実行し、データベース処理のため次々と実行される PDM, アクセス法, OS の各プログラム中の命令語列を取得する。

(2) 解析プログラムによって、命令語トレーサ取得のデータを、分岐処理、繰り返し処理を基準として分類した処理区分に応じて、各処理区分別に CPU 時間 (a, b, c, d などの係数値) を求める。

(3) 上記で求めた特定の条件下での CPU 時間に、レコード長、キー長など一般的な条件下では可変な、主に、長さ属性の変化が CPU 時間に及ぼす補正を施し、最終的な、CPU 時間算出式と、その係数値を定める。

3.2 ディスク・サービス時間

ディスク・サービス時間は、(1) 磁気ヘッドの移動量を定め、(2) 磁気ヘッドの移動量・転送データ長から、a) シーク時間、b) サーチ時間、c) データ転送時間 (リード・ライト時間) を求めること、によって推定できる。

(1) 磁気ヘッドの移動量の推定

本項では、磁気ヘッド移動量の推定で基本的な、

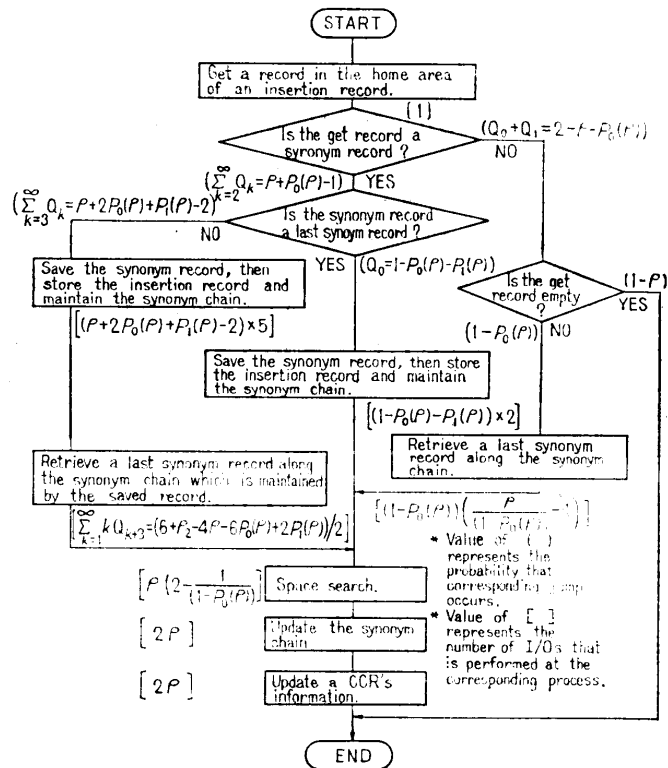


図2 追加処理と I/O 回数の推定
Fig. 2 Insertion process and the number of I/Os.

a) 全くランダムに磁気ヘッドが移動する場合と、
 b) セカンダリ・リンゲージパスのリストとして接続されたレコードをリストに従って検索する場合について述べる。

a) 全くランダムに磁気ヘッドが移動する場合
 c_i を、データ・セットの先頭シリンダを1としたとき、相対的な第 i 番目のシリンダに登録されているレコード数、 m をデータ・セットの総シリンダ数とする。

1 から $\sum_{i=1}^m c_i$ までの間で、一様整数乱数を発生し、その結果 q が得られたとする。このとき、 $0 \leq \sum_{i=1}^k c_i$ を満足する最小の k が、今回の処理でアクセスが行われるデータ・セットの相対シリンダ番号を表わす。

b) セカンダリ・リンゲージ・パスに関して、同一リスト上の i 番目に登録されたレコードから、 $(i+1)$ 番目に登録されたレコードを処理する場合。

$i+1$ 番目に登録されたレコードのデータ・セット上の相対シリンダ番号を n 、同一リスト上の i 番目に登録されたレコードが、相対シリンダ番号 j に存在する確率を $s_{i+1}(j)$ とする。 $s_i(j)$ の確率分布は、リスト長 l とデータ・セットのレコード件数とシリンダ容量によって定まる。 $(0, 1)$ の一様確率乱数を発生し、その結果 p が得られたとする。このとき、 $\sum_{j=n}^k s_{i+1}(j) \geq p \left(1 - \sum_{i=1}^{n-1} s_{i+1}(j) \right)$ を満足する最小の k が、今回の処理でアクセスが行われるデータ・セットの相対シリンダ番号を近似的に表わす。

c) 磁気ヘッド移動量の設定

代表例として a), b) で、今回アクセスする磁気ヘッドの移動をデータ・セット上の相対シリンダ番号として求めた。この値の実際の磁気ヘッド移動量への変換は、次に示す方法を用いている。

データベース定義中、エクステント(ディスク上へのデータ・セット配置)情報が、 (v_i, s_i, d_i) として与えられている。ここで、添字 i は、データ・セットが複数のエクステントから構成されていたとき、第 i 番目のエクステント情報を示しており、 v_i, s_i, d_i は、該当エクステントが存在するディスク名称、先頭シリンダの番号、およびシリンダ数を各々表わしている。

d_i 情報を用い、前に算出した相対シリンダ番号が何番目のエクステントに属するか求め、この後、 s_i 情報によって、今回アクセスすべきシリンダ番号を求める。

最後に、 v_i 情報から、当該ディスク上の現在位置と、前のステップで得たシリンダ番号との差の絶対値

を算出し、磁気ヘッド移動量を求める。

(2) ディスク・サービス時間の設定

シーク時間、サーチ時間、データ転送時間は、磁気ヘッドの移動量、転送データ長、およびディスクのハードウェア特性に基づいて算出している。

特に、シーク時間は、磁気ヘッドの移動量とシーク時間の関係を 5~6 本の折れ線グラフで近似し値を算定している。この結果、前項の処理とともに、ディスクに対するアクセスの偏りを表現した、ディスク・サービス時間の設定が可能となっている。

4. PEAL

PDM を用いて建設されるデータベース・システムの処理能力を、システム建設前に推定するツール PEAL は、今まで説明した、解析モデル、処理時間の推定法から判明するように、(1)解析手法、(2)シミュレーション手法の両手法を併用している。

PEAL の全体の構成を図 3 に示した。

4.1 PEAL 処理の分類

PEAL 処理を、次の 3 つに大きく分類する。

(1) MDS データ・ローディング・モデル

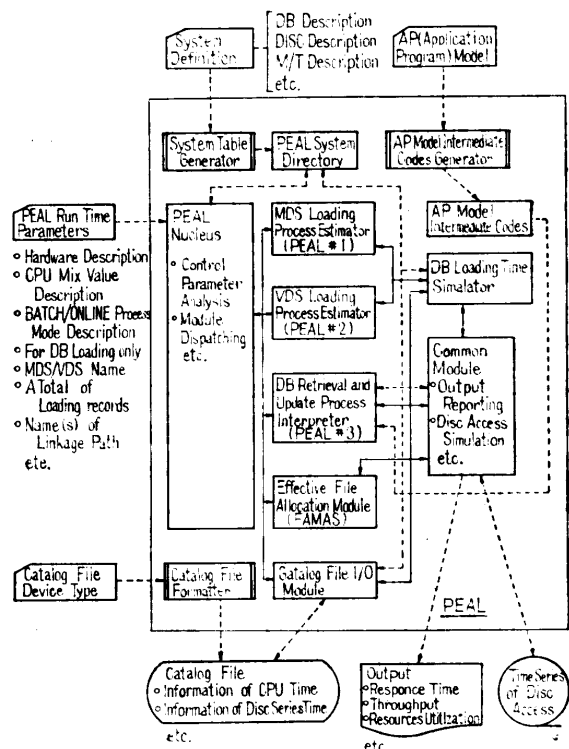


図 3 PEAL の構成

Fig. 3 PEAL Configuration.

(PEAL #1 と略す)

(2) VDS データ・ローディング・モデル

(PEAL #2 と略す)

(3) DB 検索・更新モデル

(PEAL #3 と略す)

(1) PEAL #1, PEAL #2 モデル

PEAL #1, PEAL #2 は、実行時パラメータとして与えられた、a) トランザクション件数、b) ハードウェア定義、c) 初期作成用データ・セット名称などの指示に従い、PDM データベースの初期作成時間を推定する。

この初期作成処理において、MDS, VDS いずれも、データ・セット内に収容されるデータの統計情報、すなわち、a) MDS においては、シノニム発生率、シノニム長など、b) VDS においては、リンケージ・パス・リスト長、シリンダ別データ収容個数など、が得られる。上記の結果、データベースの初期作成処理で、それ以降のデータベースの検索・更新処理の PDM 処理時間を推定する全データがそろふことになる。

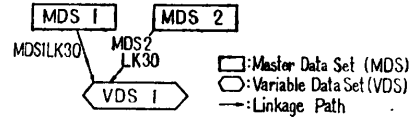
PEAL #1, PEAL #2 においては、PEAL #3 処理を行う準備として、該当処理で推定した統計情報に基づいて、PDM の各データ操作言語に対応した処理能力の情報を算定し、PEAL カタログ・ファイルに登録される。

(2) PEAL #3 モデル

PEAL #3 は、PEAL #1, PEAL #2 で作成された PEAL カタログ・ファイルの性能情報、および PEAL 言語で記述した応用プログラム・モデルを参照し、PDM 処理時間を推定する。PEAL #3 においては、PEAL #1, PEAL #2 の簡便な実行時パラメータの入力によって処理能力を推定するのは異なり、応用プログラム・モデルに基づいて処理能力を推定している。この理由は、データベースの検索・更新処理は、各応用プログラムのデータ処理特性に依存することが多く、柔軟な応用プログラムのモデリング機能が要求されるからである。

また、PEAL #3 では、応用プログラム・モデルの指示に従って、PDM データベースの状態を変更することによって、後続する処理能力推定の処理で参照できるようにしている。今までの PEAL #1, #2, #3 の説明から把握されるように、PEAL カタログによって、処理時間推定の首尾一貫性、無矛盾性が保証される。

(SAMPLE DATA STRUCTURE)



(SYSTEM TABLE DEFINITION)

```

SYSTBL PLGEN  DEVFIL=H-8578, TIMER=3
MDS 1  DSGEN  IOBUF=PRIV, DTYPE=MDS,
        KEYLEN=8, BLKSIZE=2295,
        RECSIZE=765, ELEMENT=8
        ALLOC  VSN 1=PDMO 10, DATA 1=84,
        STARTC 1=35
MDS 2  DSGEN  IOBUF=PRIV, DTYPE=MDS,
        KEYLEN=8, BLKSIZE=2292,
        RECSIZE=382, ELEMENT=6
        ALLOC  VSN 1=PDMO 10, DATA 1=34,
        STARC 1=1
VDS 1  DSGEN  IOBUF=PRIV, DTYPE=VDS,
        BLKSIZE=1674, RFCSIZE=93,
        ELEMENT=5, CLL=80
        ALLOC  VSN 1=PDMO 10, DATA 1=41,
        STARTC 1=93
        LPGEN  BLINK=MDS 1 LK 30, BREC=6,
        SLINK 1=MDS 2 LK 30, SCODE 1=31,
        SREC 1=10
        VLGEN  VSN 1=PDMO 10, DEV 1=H-8578,
        VSN 2=PDMO 20, DEV 2=H-8578
MT 00  MTGEN  TYPE=INPUT, BLKSIZE=2000,
        RECSIZE=100, DEVICE=H-8453
MT 01  MTGEN  TYPE=OUTPUT, BLKSIZE=5000,
        RECSIZE=100, DEVICE=H-8453
        PRGEN  DEVICE=H4611
        PLEND
        END

```

図 4 システム・テーブル定義の例

Fig. 4 Example of system table definition.

4.2 モデル・パラメータ

PEAL の入出力情報は、図 3 で概括的に把握でき、本節では主要な入力情報について説明する。

(1) システム・テーブル定義

データベースの構造、ディスク、磁気テープ、ラインプリンタに関するハードウェア特性を、PLGEN から PLEND の間で、テーブル定義言語を用いて記述する。システム・テーブル定義の例を図 4 に示す。

データベースは、DSGEN, ALLOC, LPGEN の組み合わせによって表現され、各定義言語の役割を次に示す。

DSGEN は、データ・セットの名称、タイプ、入出力バッファの共用化機能の有無、レコード長、ブロック長、データ・エレメント数など、データ・セットに固有な情報を記述する。本 DSGEN 定義によって、データ・セットの定義情報、データ操作言語処理時間情報を登録したデータ・セット・テーブル、および入出力バッファ制御に用いる入出力テーブルを生成す

る。ALLOC は、DSGEN で定義したデータ・セットのディスク上への割り当てを、a) ディスクの名称、b) ディスク上の該当エリアの先頭シリンダ番号、および、c) 該当エリアのシリンダ数として、必要なエクステント数分、繰り返し指定する。本 ALLOC 定義によって、データ・セットのディスクへの配置情報、各シリンダ内レコードの統計情報を登録するエクステント・テーブルを生成する。LPGEN は、DSGEN で定義した VDS のベース・リンケージ・パス、セカンダリ・リンケージ・パスを、a) リンケージ・パス名称、b) 平均リスト長、および、c) 処理コードとして記述する。本 LPGEN 定義によって、リンケージ・パス名称(先頭4文字が対応する MDS を表わす)から VDS と MDS との関連付、リンケージ・パス処理時間推定の統計情報を登録するリンケージ・パス・テーブルを生成する。

一方、周辺装置の定義を行う、VLGEN, MTGEN, PRGEN では、各々ディスク、磁気テープ、ライン・プリンタの特性を該当装置の型でもって記述する。この結果、該当装置の型に固有なハードウェアの、シーク時間、サーチ時間、データ転送時間などの性能特性情報を示すテーブルを生成する。

(2) 応用プログラム・モデルの記述

データベース初期作成処理の応用プログラムは、比較的単純なため、実行時パラメータの入力情報として簡単な論理を表現できる手段を与えることによって、モデル化することを考えた。PEAL #3 においては、PEAL 言語に基づくモデリング手段を与えている。ここでは、PEAL 言語を用いた応用プログラム・モデルについて、簡単に述べる。

PEAL 言語は、応用プログラムのモデルを記述するに際して必要となる、以下の各機能を持つコマンドから構成されている。

- a) データ処理の流れ、および繰り返し回数を表現する、CASE, IF, GOTO, LOOP, CONTINUE 文、
- b) 任意の段階で処理時間推定の開始、終了、および出力を表現する、STIME, RESET, BREAK 文、
- c) 応用プログラムの処理を記述する PROCESS, PDM 文

PEAL 言語を利用した応用プログラム・モデルの記述例を図5に示した。

図から、言語仕様として、PEAL 言語は、FORTRAN と同仕様の高級言語であることが把握される。この理由は、FORTRAN がよく使用された言語であ

```

PLSTATE START      INPUT=MDS 1, MDS 2, VDS 1, ×
                   OUTPUT=TEMP, TEMP, DBO 1
                   STIME      T1, T2
                   PROCESS    C=300, GET=MT00
                   CASE       (1000 P=20), (2000 P=20), ×
                               (3000 P=10), (3000 P=50)
1000  PROCESS      C=100, GET=MT00
                   BREAK     T=T 1, OPT=A, PCHANG=N
                   GO TO     5000
2000  PROCESS      C=200, GET=MT00
                   PDM       PROC=ADD-M, DS=MDS 1, I=1,
                               M=R
                   GO TO     5000
3000  PROCESS      C=300, GET=MT00
                   PDM       PROC=READM, DS=MDS 2, I=1,
                               M=R
                   PDM       PROC=READV, DS=VDS 1, ×
                               LP=MDS1LK00, I=1, M=S, V=L
5000  CONTINUE
                   BREAK     T=T 2, OPT=A, PCHANG=N
                   END
    
```

図5 応用プログラム・モデルの記述例

Fig. 5 A user application program model written by PEAL language.

り、理解し易い点を考慮したことによっている。また、実際の処理時間推定に当たっては、PEAL 言語を、一旦中間コードに変換し、それを解釈・実行し、処理を進めている。中間コードは、下のフォーマット

/ 2 bytes \	/ 2 bytes \	/ n bytes \
Length Attribute	Operation Code	Operand

を有しており、本フォーマットにより各 PEAL 言語を表現する。中間言語の解釈・実行に際しては、次にどのインストラクション (PEAL 言語) を実行するかを、中間コードの長さ属性を用いてあらかじめ決定しておく。上記に対し GOTO, CASE などの分岐処理が該当中間コードで表現されていた場合は、PEAL 文実行のため用意されている実行時ルーチンの制御によって、指定された分岐先に分岐するように、インストラクション実行アドレスを修正する。

4.3 推定精度の検証、計算機所要時間

PDM 用ソフトウェア・モニタ、ハードウェア・モニタを用いて実測した PDM の性能データに基づいて、PEAL の推定情報の検証を行った。

ソフトウェア・モニタは、主に、MDS のランダムイズ・アルゴリズムのランダム性の検証、および入出力回数推定の精度検証の問題に適用した。この結果、解析モデルの入出力回数推定の精度として、90% 以上が期待され、実システム適用上のモデリング面での妥当性が保証された。

以下では、PDM 処理時間の推定精度、および PEAL

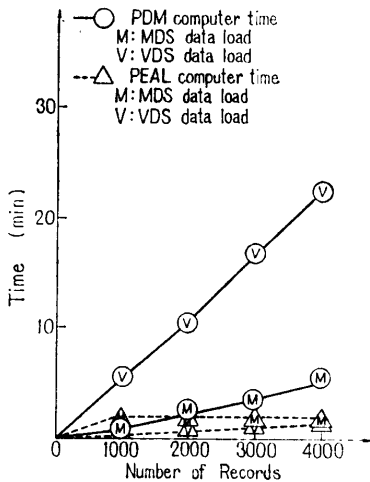


図 6 実測との比較による PEAL の推定精度
Fig. 6 PEAL estimation accuracy in experiments.

の推定処理に必要な計算機所要時間に関し、PDM の実測内容と、PEAL 実行結果によって、比較検討する。

図 6、図 7 にデータベースの初期作成時における実測結果と、PEAL との比較を、精度、計算機所要時間、各々について示した。

図 6 のグラフは、PEAL の推定精度が、実 PDM 処理時間の 95%~110% であることを示している。我々は、他の実アプリケーションへの適用時にも、本図と同等の推定精度を得ており、本 PEAL の推定精度として、95%~110% 程度と考えている。

図 7 の計算機所要時間の比較結果から、

a) MDS の初期作成では、データ処理件数が多い程、PEAL の効果が大きい点 (高々 1000 の代表点によって処理時間推定を実施している理由による)。

b) VDS の初期作成では、PEAL は実 PDM のデータ処理時間の 1/10 程度の計算機所要時間である点、などが把握できる。

また、データベースの検索・更新処理では、PEAL の計算機所要時間は、実 PDM の 1/20~1/30 であることを確かめている。

5. おわりに

HITAC 8000 シリーズ用 DBMS PDM を対象に作成した。性能推定システム PEAL について報告した。本 PEAL の特長は、(1)データベースの初期作成から検索・更新までの一貫したデータ処理能力推定を可能にし、(2)解析手法、および性能推定用統計情報のカタログ化機能、によって、性能推定用計算機時間の

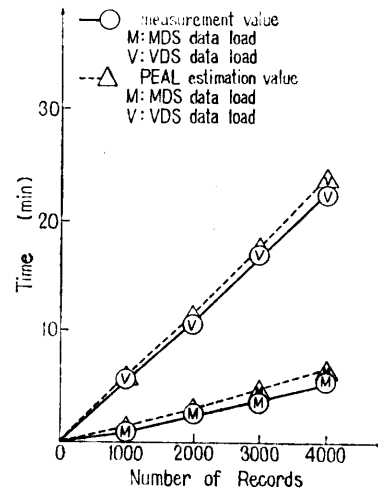


図 7 性能推定に必要な計算機所要時間の比較
Fig. 7 PEAL computer time vs.
PDM computer time.

短縮 (実処理の 1/10~1/20) とシミュレーション手法の導入により、ディスク・アクセスの偏りの表現など、推定精度の高精度化を実現し、(3)FORTRAN レベルの応用プログラム・モデル記述言語開発により、モデリング作業の負担を軽減した、点にある。

我々は、今回報告した性能推定ツール PEAL の開発に見られるように、データベース・システムの建設、運用段階でのシステムの性能、および操作性の向上に重点を置いて技術の蓄積を図ってきた。その他の性能評価の面では、シミュレータ、ソフトウェア・モニタ、データベース・トレース解析ツールなどを開発している。今後これらを通じ、汎用的なシステム設計技法の確立を推進していきたい。

参 考 文 献

- 仁平ほか 4 名：中小規模データ・マネジメント・システム PDM の性能特性と評価ツール，日立評論，Vol. 58, No. 9, pp. 73-77 (1976).
- 米田，山口ほか：汎用 DBMS の性能評価システムについて，情報処理データベース管理システム研究会資料，DB 31-3 (1976).
- 米田，山口：汎用 DBMS のソフトウェア・モニタについて，情報処理学会第 16 回大会予稿集，pp. 535-536 (1975).
- 日立製作所：EDOS/EDOS-MSO データマネジメント・システム PDM リファレンス・マニュアル。
- Teorey, T. J. and Oberlander, L. B.: Network data base evaluation using analytical modeling, Proc. NCC, pp. 833-842 (1978).
- Pool, J. A. van der: Optimum Storage Al-

location for Loading of a File, IBM J. Res. Dev. Vol., 16, No. 4 (1972).

7) Hulthen, C. and Soderlund, L.: A Simulation Model for Performance Analysis of Large Shared Data Bases, Proc. 3rd VLDB, pp. 524-532 (1977).

8) Honeywell Information Systems, Inc.: Integrated Data Store, BR 69 (1971).

付 録 PDM に関連した基本用語の説明

(1) (データ)・マイテム: 氏名・年齢など, 情報として意味を持つ最小単位である.

(2) (データ)・エレメント: 1つ以上のデータ・マイテムの集合である. PDM と応用プログラムとのデータ受け渡しの単位となっている.

(3) (データ)・レコード: ロジカル・レコードと

もいう. 1つ以上のデータ・エレメントから構成され, PDM がデータベースを操作する単位である.

(4) (データ)・ブロック: フィジカル・レコードともいう. 1つ以上のデータ・レコードから構成され, 主メモリとディスクとの間のデータ転送の単位である. 1ブロック中に登録できるレコードの最大数をブロッキング・ファクタと呼ぶ.

(5) データ・セット: (データ)・ファイルともいう. 同じ性質を持つデータ・レコードの集合で, ディスク上に確保される領域 (エクステンツ) と対応する.

(6) データベース: 種々のアプリケーションによって処理されるデータセットの集合である.

(昭和53年10月17日受付)

(昭和54年2月13日採録)