C 3　教育用FORTRANシステムHITFOR

電気通信大学　金山　裕・飯島純一
法政大学　犬飼和元*

0．はじめに

通常のOSの下でFORTRAN プログラムを実行するためには，コンパイラとリンケージ・エディタとローダを働かせなければならない。この型のコンパイラを「汎用」と呼ぶことにする。汎用コンパイラは(a)プログラム単位を個別に翻訳できること，(b)プログラム単位毎の目的プログラムをファイルできること，(c)比較的大きなプログラムを処理できること，(d)他の言語で書かれたプログラムと結合できること，などの利点がある。

ところが，これを教育用に使ってみると上のような特徴は格別生きてこないばかりでなく，多くの不具合が目立ってくる。たとえば(a)短いプログラムについても結果が得られるまでに10秒～1分の時間がかかること[1]，(b)多数のコントロール・カードを必要とすること，(c)不必要に多くのLP用紙を費やすこと，(d)エラーチェックが不十分であること，(e)エラーメッセージがわかりにくいこと，などが欠点として挙げられる。そのために，初心者を教育する目的に合致したコンパイラが考えられるようになってきた。上記のような欠点が改善されたこの種のコンパイラのことを「教育用」と呼ぶことにする。初期のものとしてはPUFFTとWATFORが挙げられる。[2)3)] 我国でもミニコンのためのもの[4)5)]，あるいはFACOM 230／45S，55 のためのFAST[6)]が報告されている。

ここで，有名なWATFORについて簡単に紹介しよう。これは32K語のIBM7040 のために1965年にWaterloo 大学において組み込まれた。言語レベルはFOTRAN Ⅳ である。翻訳速度を上げるために，IOCS，翻訳時プログラム，実行時プログラムを主記憶に常駐させ，翻訳が終了した時点で目的プログラムの絶対番地が割当て済みであり，ただちに実行に移される。リンケージ・エディタの機能は翻訳プログラムの中に含まれている。システムへの入力は磁気テープを通して与えられる。典型的な学生ジョブに要するCPU時間は2，3秒であるという。

本稿で報告するシステムはHITAC－8150のために日立製作所と筆者らが共同で開発した教育用FORTRAN HITFOR(HITACHI TEACHING FORTRANである。24KBをもつ8150にJIS 3000[7)]レベル以上のFORTRAN を組み込むことが目標とされたが，すべてのシステムを常駐とするWATFORの方式でこれを実現させることは不可能であった。この点を解決するためにとられた技術はおそらく従来の教育用コンパイラにみられなかったものであろう。[8)9)]

1．システム設計の境界条件

目標とした文法仕様はつぎの通りである。

(1) FORTRAN JIS 3000 を完全に含み，さらにつぎの機能を附加する。

(2) A変換

(3) 3次元配列

(4) 5桁の文番号，6文字の英字名

＊　現在はビジネスコンサルタント

以上の機能を組み込むべきHITAC－8150システムの標準機器構成はつぎの通りである。

(1) H－8150　処理装置，24KB　(40KBまで)

(2) ディスク 4.9MB

(3) コンリールディスプレイ

(4) カードリーダ，310枚／分

(5) 紙テープリーダ，500字／秒

(6) ラインプリンタ，430行／分

8150の命令実行時間は数十μ秒程度である。

ベースレジスタによるアドレス修飾は行われない（このために翻訳の際の番地処理は楽になっている）。頻繁に用いる命令は4Bまたは6B，また数値語は6Bを占めるから，24KBの主記憶に約5Kステップのプログラムを収容することができる。

設計開始時点において，つぎの目標をシステムが満たすことをねらった。

(1) カードリーダがソースカードを読む時間内に翻訳を完了すること。

(2) 100 ステップの長さと500 個の要素をもつ配列を扱うプログラムを処理できること。

(3) エラーチエックを（特に実行時のものを）厳しく行うこと。

(4) 簡単な演習問題が数十秒で実行できること。この目標はあいまいであるが，要するに，実行速度が過度に遅くならないこと。

(5) 発生したエラーの種類，ユーザ名，日時などを含むエラーファイルをもち，エラーの統計情報をユーザに提供できること。

以後の各節において，HITFORの操作，構造あるいは性能について述べるが，特に汎用のもとの相異点に焦点を合せてみたい。教育用と汎用のコンパイラには共通の機能をもつ部分が多く，それらについて述べた成書は多いからである。

2．設計方針

これらの設計目標を実現するために，次の様な方針を採った。

(1) 8150の汎用オペレーティング・システム（8150 Programming System，8150PS と呼ばれる）とは独立なHITFOR専用のOSを作成する。これによって，他のシステムとの互換性は制限されるが，処理時間の短縮と，主記憶のスペースの節約を図ることができ，さらに，OSとの種々のメッセージのやりとりとコントロール・カードを簡潔なものにできる（表1参照）。

(2) システムは次の部分から構成されると考えられる。

(a) 翻訳時プログラム。構文解析や目的コードの生成など，プログラムの翻訳だけに必要な部分。

(b) 実行時プログラム。基本外部関数，実行時入出力の処理などの目的プログラムの実行にのみ必要な部分。

(c) 管理プログラム。入出力装置の制御等，両者に共通な部分

これらをすべて主記憶上に配置することが望ましいが，これらの大きさの和は24KBを超える。そこで，これらを分割してディスク上に配置し，(c)は常に主記憶上におくが，(a)，(b)は必要な時点で主記憶に呼び込む（オーバーレイする）ことにした。即ち，1つのユーザーのプログラムを翻訳・実行する間にディスクから主記憶へ(a)，(b)の順に2回プログラムをロードする。この方式は，完全な常駐型ではないが，翻訳時プログラムと実行時プログラ

表１　HITFORコントロール・ステートメントとその機能一覧

項番	コントロール・ステートメント	機能概要
1	//HTFR	HITFORプログラムの始まりを示す。このステートメント以降，次のコントロール・ステートメントが現われるまでをFORTRANステートメントとみなして，コンパイルを開始する。
2	//EXEC	オブジェクト・プログラムを実行する。このステートメントが現われる直前に文法の誤りがなくコンパイルの終ったプログラムが実行される。又は，直前に実行の終ったプログラムが再実行される。
3	//END	一連のHITFORプログラムの終了をシステムに知らせる。HITFORシステムは初期状態になる。
4	//CHNG	FORTRANの入出力文で参照される入出力装置の論理機番と実装置との対応関係を変更する。
5	//ERLG	エラーファイル関係の作業を行う。

表２　システムディスクの構成の１例

	ディスクアドレス						用途
	LHEアドレス			RHEアドレス			
1	0C	0T	0S	0C	3T	23S	マイクロプログラム
2	1	0	0	1	0	2	ブートストラップ
3	1	1	23				ボリューム・ラベル
4	2	0	0	2	3	23	VTOC
5	3	0	0	3	2	23	管理プログラム
6	3	3	0	3	3	23	コード変換テーブル
7	4	0	0	4	1	23	翻訳時プログラム
8	4	2	0	4	3	23	実行時プログラム
9	5	0	0	5	0	23	エラーファイル処理プログラム
10	6	0	0	7	3	23	エラーファイル処理用ワーク・エリア
11	8	0	0	17	3	23	エラーファイル（10c）
12	18	0	0	47	3	23	SYSIPTエリア
13	48	0	0	72	3	23	FWK01（FORTRANファイル）
14	73	0	0	97	3	23	FWK02（　〃　）
15	98	0	0	200	3	23	空きエリア

注）LHEはディスクの若い方のアドレス，RHEは老いた方のアドレスをさす。

ムをディスク上の同一シリンダに配置して（表２参照），読取りヘッドの動きを押えれば，すべてのプログラムが常駐している場合とほとんど同じ効果を上げることができる。又，これによって初期値設定に特別なプログラムを必要としないなど二次的な効果もあって，主記憶領域の大巾な節約が可能となり，その分ユーザー領域を拡大できる。

(3) 実行時にインタプリタを多用する。

通常の方式をとると，翻訳時プログラムの方が，実行時のものよりも大きくなりやすい。そこでインタプリタを幾分か多用すれば，翻訳時プログラムが小さくなって，(2)の分割の結果として，ユーザー領域を大きくとれる。実際，両者はほぼ同じ体積をもっている。（図１参照）

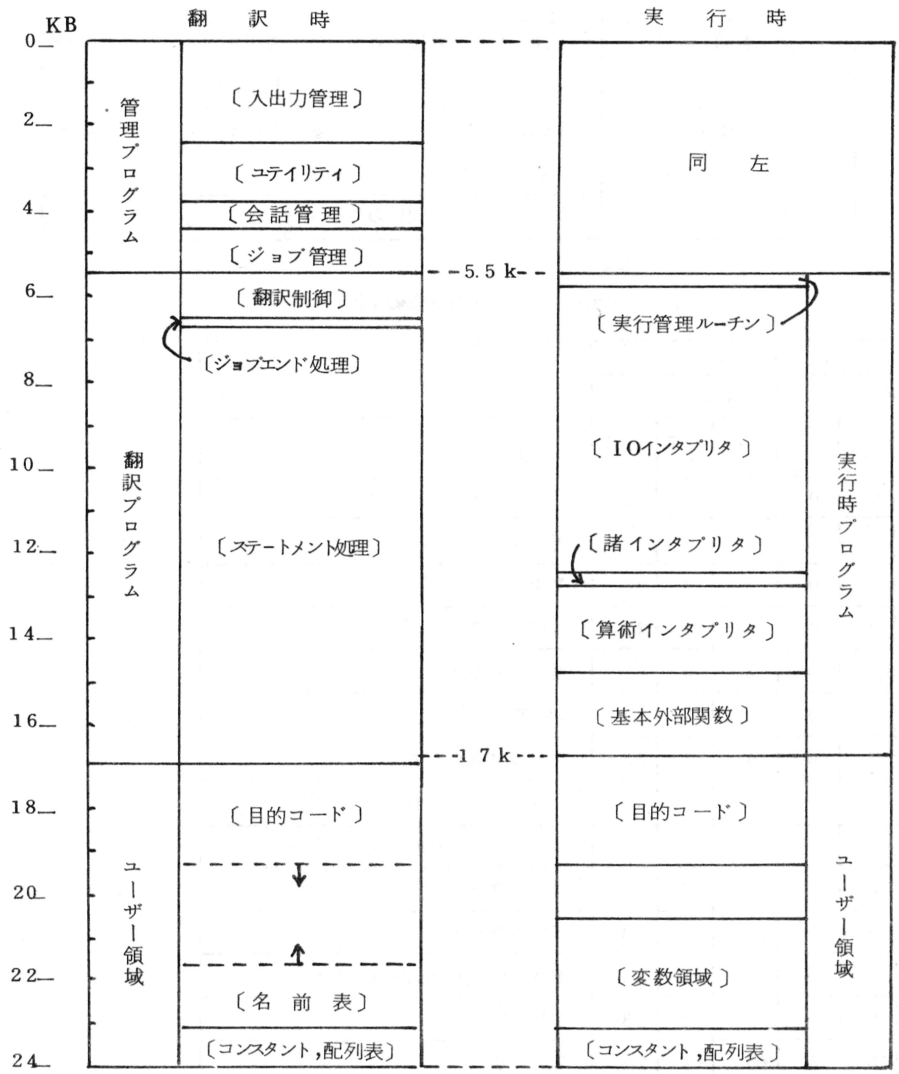

図１　メモリ・マップ

また翻訳速度は一般に速くなり，教育用コンパイラの目的にもよく合致する。

３．ユーザー領域の番地割り当て

２４ＫＢのシステムにおいてはユーザー領域は全体で７２７２Ｂあり，それが図２にみるとおりに，６７６０ＢのＡ領域と５１２ＢのＢ領域に分割されている。翻訳時にはＡ領域に絶対番地がほぼ割り付けられた目的プログラムと，名前表がそれぞれ上，下端からとられる。名前表には変数名，配列名などだけでなく，文番号なども登録されている。

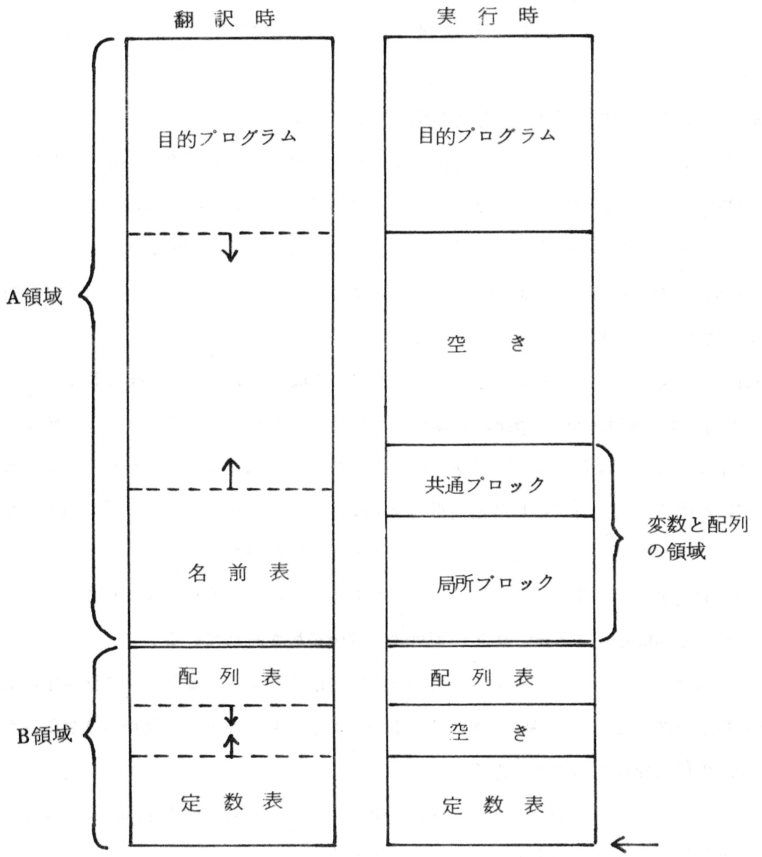

図２　ユーザー領域の番地割り当て

一方，実行時にはＡ領域の下方から変数（配列を含む）のための領域がとられる。その中でも，一般の局所的な変数が老いた番地にコモンの変数がそれと隣合った若い番地にとられる。この割り当てアルゴリズムを採用しているために，コモンの変数などの番地は実行可能プログラム全体が読み込まれてはじめて決定される。Ｂ領域には配列表と定数表が上下端からそれぞれ割り付けられる。翻訳が終了した時点の表の内容が，そのまま，実行時に持ち越される。

4．名前表の構成

名前表のエントリは１２Ｂから成り，図３に示すように５つの部分に分割されている。各部の意味はつぎの通りである。

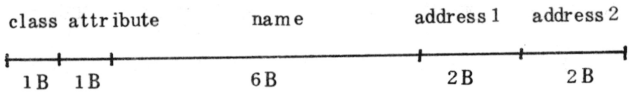

図３　名前表エントリの各部

(1)　class part は次の英字名，又は文番号名の類を区別する，

変数名，配列名，サブルーチン名，外部関数名，基本外部関数名，文関数名，組込み関数名，実行文番号，FORMAT文番号，未定義仮引数（これは翻訳過程において一時的に現われるものである）。

class part 以外の各部のもつ意味は，厳密には，class part が何であるかに依存してきまる。

(2)　attribute part の８ビットはそれぞれがほぼ独立の意味をもつ。例えば，実数型か否か，値が定義されたか否か，仮引数であるか否か，等である。

(3)　name part には英字名（最長６Ｂ），または文番号名（最長５Ｂ）がセットされる。

(4)　address 1 part，address 2 part には何らかの番地がセットされる。８１５０には最大４０ＫＢが実装可能であり，４０Ｋ＜６４Ｋ＝2^{16} であるから絶対番地がつねに２Ｂで表わせる。この部分にセットされる番地としては数値語に割り当てられた絶対番地または相対番地，目的プログラムの場所，鎖が作られているときの根元，サブルーチンの入口，配列表のエントリの番地などがある。

名前表の中の配列名エントリには，ベースアドレスや各寸法などの情報は含まれていない。これらは別に配列表の中に書かれ，address 1 part にその配列表の番地がしまわれている。

変数名，文番号などの，プログラム単位内で局所的な名前のエントリは原プログラム中にＥＮＤ行が現われてプログラム単位が閉じると（値が定義されているか否かなどをチエックしながら）名前表から取り除かれる。これは記憶場所の節約をはかるためである。

配列表の各エントリは配列の(a)ベースアドレス，(b)第１寸法，(c)第２寸法，(d)第３寸法，(e)全体の占めるバイト数，をそれぞれ表わす部分から成り，１３Ｂを占める。

5．定　数　表

実数型と整数型の定数は一般の数値語と等しく６Ｂを占める。実行可能プログラム中のすべての定数は（ＤＯのパラメータも含めて），一様に定数表に収められる。２個以上のプログラム単位で等しい定数Ｃを用いると，Ｃは表の中の１個にしかセットされず，いずれのプログラム単位からも同一番地が参照される。

6．翻訳時プログラムの処理

ステートメントを分類するルーチンは(a)文字欄記述子の中を除いて空白はどこにあってもよい，(b)英字名ＦＯＲＭＡＴを配列名または文関数名として使ってはならない点を除いては予約語を設けていない，という条件を満たすよ

うに作られている。

つぎにＤＯループの処理について簡単に述べる。

ＤＯ文を処理するためには翻訳時に適当な深さのスタックをもつ方式が一般的である。しかし，HITFOR においては島内氏によるつぎの方式を採用した。[5] それはＤＯループの翻訳に必要な領域を目的プログラムの中にとる，というものである。

HITFOR のDOループの目的プログラムは，最終的には，図４のようになる。ＤＡと名付けた部分は制御変数へ

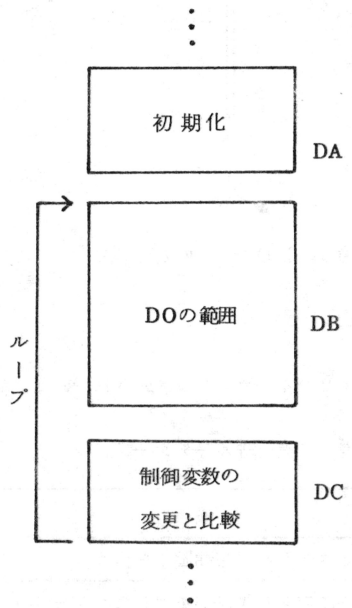

図４　ＤＯループの目的コード

の初期値設定，３つのパラメータの値のチエックを行うものである。ＤＢはＤＯの範囲を翻訳した部分である。ＤＣは制御変数に増分を加え，終値と比較する部分である。ＤＢの翻訳をしている段階では(a)端末文番号，(b)制御変数，(c)増分，(d)終値，(e)ＤＢの開始番地，に関する情報が必要である。このうちの(a)～(d)の情報（実はそれらの名前表中の番地）をＤＡ内の特定の場所にセットしておくことがこの方式の骨子である。その特定場所を知っていさえすれば，ＤＯがいつ閉じるかを判定すること，およびＤＣ部を生成することが可能となる。

例えば，ＤＯの入れ子が二重になっていて，内側のＤＯの範囲の翻訳をしている最中の目的プログラムの様子は図５のようになる。DA_1，DA_2，はそれぞれ外側と内側のＤＯの初期化を行う部分である。その中に，上記の情報をセットしておく他に，２Ｂのリンクを更に置いて，全体を鎖にしておく。その根元は特定番地（DPNTR，do pointer）にセットする。内側のＤＯの範囲が閉じるか否かはDPNTRを手掛りとしてDA_2中の特定の場所を調べるとわかり，閉じたならば，すべての必要な情報はやはりDPNTRだけからひくことができる。その後にDPNTRがDA_1を指すように変更すること，及びDA_2の部分を完成させることも容易である。

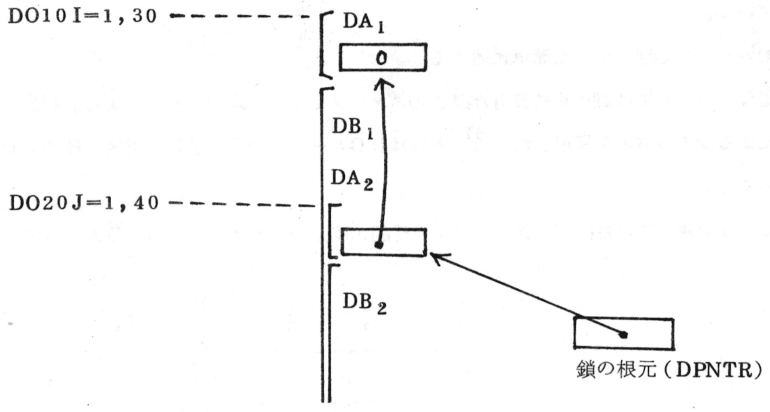

図5　DOループの処理方法

7．実行時プログラム

実行時プログラムは，機能別にみて，12のルーチンから構成されている。その機能の概略を表3に示す。

表3　実行時ルーチン一覧

名　　称	機　　能
実行制御ルーチン	目的プログラムの初期設定を行う。
実行時エラー処理ルーチン	実行時に起きたエラーに対してエラーメッセージを出力し，制御を管理プログラムに戻す。
STOP処理ルーチン	STOP文の目的コードにより制御を移され，目的プログラムの実行を終る。
PAUSEインタプリタ	PAUSE文の実行されたことをオペレータに知らせ，目的プログラムの実行を中断する。オペレータの指示によって，再開，打ち切りができる。
DOインタプリタ	DOループの初期値設定を行う。
IFインタプリタ	算術式の計算結果を判定し，正，負，ゼロに対応する飛び先に制御を移す。
計算型 goto インタプリタ	整変数の値に対応する飛び先に制御を移す。
配列入出力インタプリタ	入出力並びの中の配列に対する入出力処理を行う。
配列引数インタプリタ	引数として用いられた配列の引き渡しを行う。
IOインタプリタ	データの入出力を行う。
算術インタプリタ	算術代入文の処理を行う
基本外部関数	EXP，SQRTなどの基本外部関数群

実行時プログラムが主記憶にロードされると，最初に制御を渡されるのが実行制御ルーチンである。このルーチンによって，初期値の設定を行ったのち目的プログラムが呼び出され，実行される。目的プログラムは必要に応じて各種インタプリタを呼び出し処理を行う。目的プログラムが終了するのは次の場合である。

(1) STOP 文の目的コードを実行した。これによりSTOP 処理ルーチンが働き，制御は管理プログラムに戻る。

(2) PANSE文の目的コードを実行し，かつオペレータが続行しない指示をした場合。この場合は，PAUSEインタプリタが働き，これを経由して制御は管理プログラムに戻る。

(3) 実行時エラーの発生。あふれ，未定義変数を値で読み出すなどの誤りを生じた場合にはエラー処理ルーチンでエラーメッセージが出力される，制御は管理プログラムに戻る。

ここで図6に示されるFORTRANプログラムを例として，目的コード，各インタプリタの働きを述べる。ここでは次のインタプリタが用いられる。

```
HITAC-8150  HITFOR (02-00)        SOURCE PROGRAM LISTING        PROG, ID 741-111        DATE 10/29/73        PAGE 001

      C     SUMMATION                                                              SUM00000
001         DIMENSION A(20)                                                        SUM00010
002         READ(5,100)A                                                           SUM00020
003         TOTAL=0.                                                               SUM00030
004         DO 10 I=1,20                                                           SUM00040
005      10 TOTAL=TOTAL+A(I)                                                       SUM00050
006         WRITE(6,200)A,TOTAL                                                    SUM00060
007         STOP 11                                                                SUM00070
008     100 FORMAT(20F4.1)                                                         SUM00080
009     200 FORMAT(1H ,19(F4.1,2H +),F4.1,2H =,F5.1)                               SUM00090
010         END                                                                    SUM00100

 0.1 + 1.2 + 2.3 + 3.4 + 4.5 + 5.6 + 6.7 + 7.8 + 8.9 + 0.2 + 1.3 + 2.4 + 3.5 + 4.6 + 5.7 + 6.8 + 7.9 + 0.3 + 1.4 + 2.5 = 77.1

*** HITFOR STOP 11 ***
```

図6　プログラム例

(1) IOインタプリタ（IOPPEN, IOGET, IOCLSE）

(2) 配列入出力インタプリタ（ARYI）

(3) DOインタプリタ（DOI）

(4) 算術インタプリタ（FAI）

これらのインタプリタは次のような機能を持っている。

(1) IOインタプリタ

IOインタプリタは，入出力関係の処理を行うもので，3つのサブルーチンから構成されている。1つはIOOPENで，FORTRANの1つの入出力文に対して1回呼ばれ，その入出力文の種類（READ，WRITE，REWI－

ND，BACKSPACE，ENDFILE），入出力装置の論理機番の格納されている語の番地，FORMAT 文の格納されている領域の先頭番地の3つの情報を目的プログラムから引き渡しを受ける。第2は，IOGET で，FORMAT の走査，データの外部表現と内部表現の間の変換，実際の入出力動作を行う。 1 回呼ばれるごとに1語の処理を行う。第 3は IOCLSE で FORMAT の走査の後始末，最後のレコードの処理等を行う。

(2) 配列入出力インタプリタ

入出力並びの中に配列がある場合；配列表を引き，ベースアドレスと大きさを求めて，IOGETを必要な回数だけ繰返して呼び全要素の入出力を行う。

(3) DO インタプリタ

DO ループの入口で，初期値，終値，増分をそれぞれ制御変数，目的コードに埋め込まれた2つのバッファにセットし，それらが負でないことをテストする。DO のループの部分は，これらの値を用いて直接機械語の命令で構成されている。

(4) 算術インタプリタ

主に算術代入文の計算と代入の処理を行う。翻訳時プログラムは，算術式を逆ポーランド記法により表現された擬似命令の列に変換し目的コードとする。実行時にこのインタプリタが呼ばれると，擬似命令を解読して，1つのスタックの上で種々の演算を行う。このインタプリタの擬似命令とその機能を表 4 に示す。

表 4　算術インタプリタ擬似命令一覧

命　令	機　能　概　要
EXIT	算術インタプリタを出る。
LDT	整数型データをスタックにのせる。
LDR	実数型データをスタックにのせる。
LDIA	整数配列要素データをスタックにのせる。
LDRA	実数配列要素データをスタックにのせる。
STI	スタック上のデータを整数型で変数に代入
STR	スタック上のデータを実数型で変数に代入
STIA	スタック上のデータを整数型で配列要素に代入
STIR	スタック上のデータを実数型で配列要素に代入
ADDI	整数加算
SUBI	〃　減算
MLTI	〃　乗算
DIVI	〃　除算
ADDR	実数加算
SUBR	〃　減算
MLTR	〃　乗算
DIVR	〃　除算
ABS	組込関数ABS，IABS
INVT	符号かえ
SIGN	組込関数SIGN，ISIGN
IFIX	〃　IFIX
FLOAT	〃　FLOAT
PII	(整数)＊＊(整数)
PRI	(実数)＊＊(整数)
PRR	(実数)＊＊(実数)
NOOP	何もしない

図6のプログラムに対する目的プログラムを図7に示す。この目的プログラムの実行は，実行制御ルーチンが，3行目の，READ 文に対応する目的コードに制御を移すことから始まる。

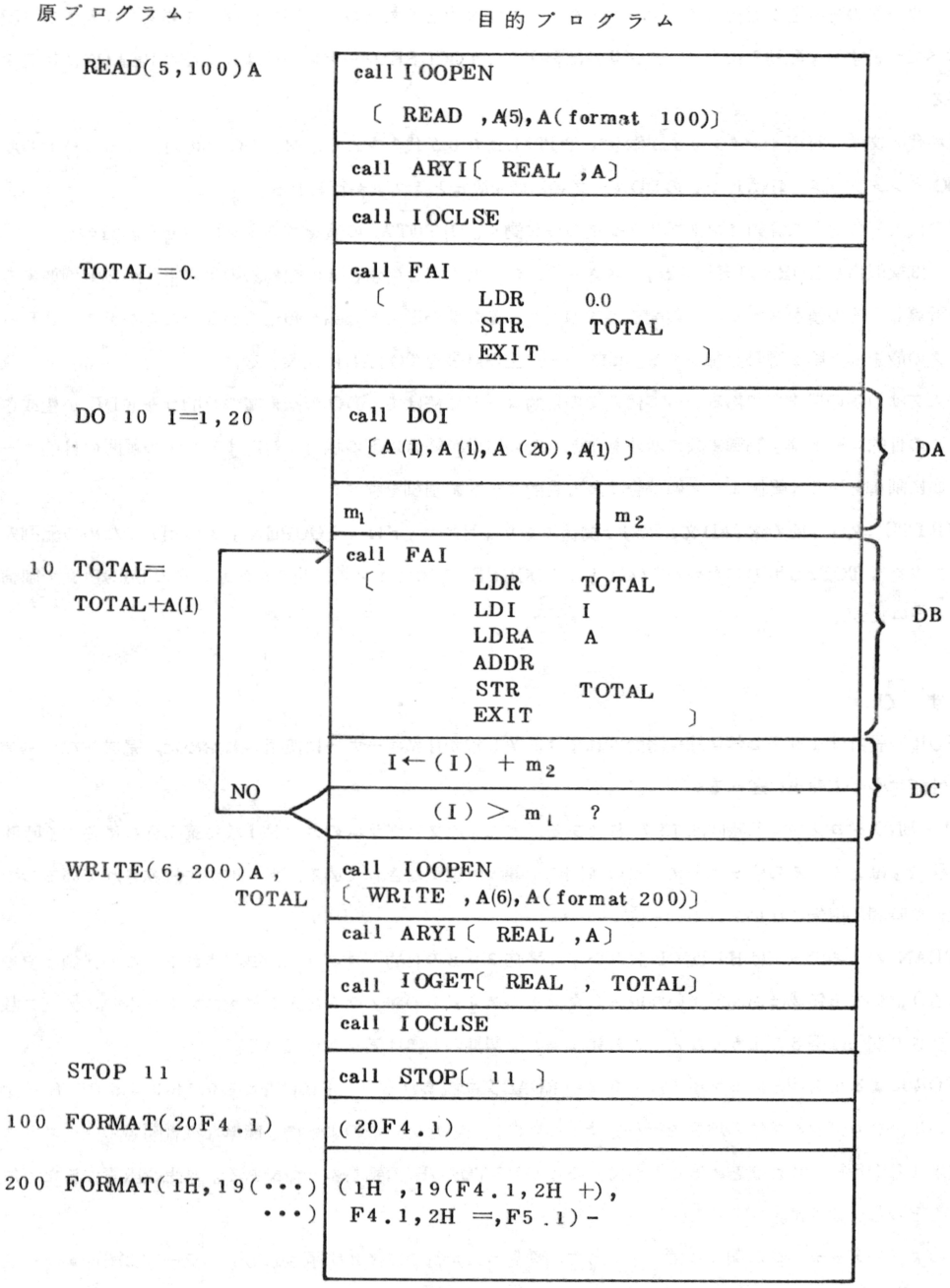

図7　目的プログラムの例

READ文の目的コードはまずIOGETルーチンを呼び，ＩＯインタプリタに，(1)READ文，(2)論理機番の格納されている語の番地（この場合，整定数５），(3)文番号１００をもつ，FORMAT 文の格納されている領域の先頭番地を引き渡す。次に配列入出力インタプリタ（ARYI）を呼ぶ。このインタプリタは，配列表からAのベースアドレスと大きさを求め，配列要素の数だけIOGETルーチンを呼ぶ。IOGETルーチンは，文番号１００をもつFORMAT文の走査とカードの読み込みを行って，カード上のデータを次々と（ただし，1回の呼びで1要素）代入する。配列要素をすべてつくすと，配列入出力インタプリタは終了し，IOCLSE ルーチンが呼ばれ，このREAD文に対応する処理が終る。

4行目の代入文は，算術インタプリタが働き，TOTALに0.0が代入される。次にDO 文に対応するコードDAによってDO インタプリタが呼ばれ，このDOループの初期値設定とテストが行われる。

6行目の代入文に対して算術インタプリタは次の様に働く。(1)TOTALの値をスタックにのせる，(2)Iの値をスタックにのせる。(3)擬似命令LDRAが実行され，スタックにのった添字Iの値とAの配列表のエントリから配列要素のアドレスを計算し，その値をスタックにのせる。このとき，Iの値が正でSize1を超えていないことがテストされる。(4)スタックの最上部と第2番目を加算する。(5)スタック上の結果をTOTALに代入する。

この代入文はDO の端末文であるから，代入文の目的コードに続けて，DO の端末文の目的コードDCが生成されている。この目的コードは，制御変数に増分を加え，その結果が終値以下の場合，対応するDO の範囲の目的コードDB の先頭に制御を戻し，終値より大の場合は次の目的コードに制御を移す。

次のWRITE 文は，前のREAD文と双対な関係にあり，目的コードは，IOOPEN，Aの出力のための配列入出力インタプリタ，TOTAL出力のためのIOGET，IOCLSE の順にインタプリタを呼ぶ。次のSTOP 文で制御は管理プログラムに戻る。

８．む す び

HITFORの設計は１９７２年7月に開始され，１９７３年4月にユーザーに使用され始めた。完成したシステムの特長と性能について簡単に述べる。

STOP－END ジョブの実行時間は1.97 秒である。また，原プログラムのカード１枚を読み込むに要する時間は0.185 秒で，ほとんどあらゆる場合に，この時間内で翻訳処理がなされている。カードが先読みされているので，翻訳のための時間的損失はない。

FORTRAN の１ステップはHITFORにおいて，平均３０Ｂの目的プログラムに翻訳される（この評価はやや大きめである）。いまB領域はあふれていないとしたとき，２４ＫＢのシステムに入る最大プログラムのステップ数と配列の大きさの関係は図８のようになる。これによると，初期の目標は満足されている。

HITFORによる出力例を図９に示す。００２のREAD文で読まれるカードには左から０１２３４５６７８９△が書かれている。プログラム単位の境目で行をあけることはしていない。このように，簡単な演習問題の　ープリストと出力結果をＬＰ用紙１枚にまとめることができるのがHITOFOR の特長の一つである。学生を教育するために禿山を作ることのないように。

翻訳時のエラーメッセージは図１０に示すように，発見された行の直後に印刷される。エラーは3桁のエラー名称と，ある場合には英字名などの附随情報が印刷される。[9] シンタクスエラーが発見される迄の時間が短い点についていては，汎用コンパイラとは比較にならない。エラーファイルを開いておくと，翻訳時，実行時を問わず，発生したすべてのエラーがディスクに記録され，必要に応じて図１１のような統計データを出力することができる。この資

料は実習担当教員にとって参考になるであろう。

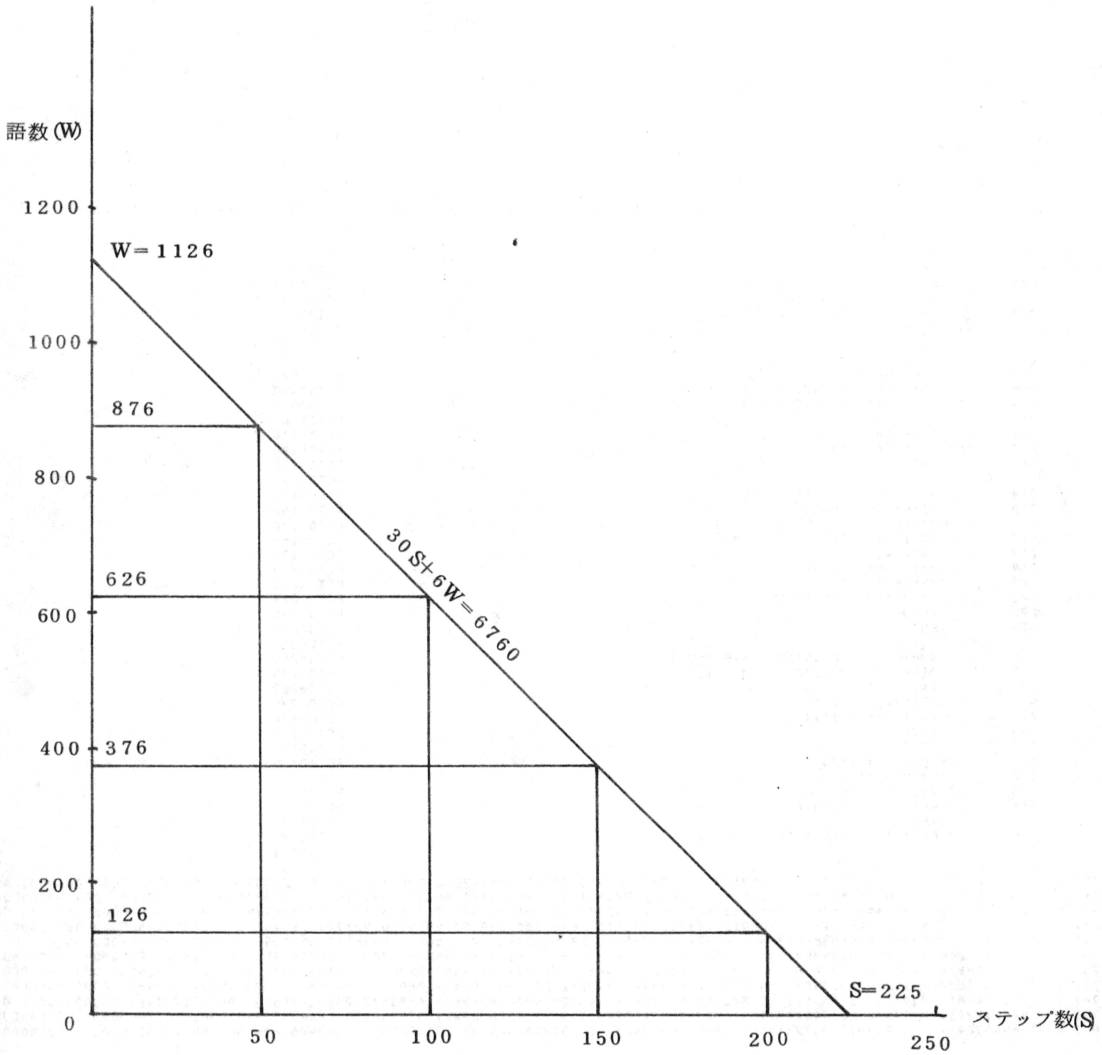

図8　プログラムのステップ数と使用語数の関係

```
HITAC-8150  HITFOR (02-00)        SOURCE PROGRAM LISTING        PROG. ID 741-110        DATE 10/29/73        PAGE 001


      C     FACTORIAL                                                          FCT00000
001           DIMENSION M(120),CHRCTR(11)                                      FCT00010
002           READ(5,100)CHRCTR                                                FCT00020
003           DO 10 I=1,119                                                    FCT00030
004        10 M(I)=0                                                           FCT00040
005           M(120)=1                                                         FCT00050
006           DO 20 N=1,80                                                     FCT00060
007           CALL NEXT(N,M)                                                   FCT00070
008           IF(70-N)11,11,20                                                 FCT00080
009        11 CALL PRINT(N,M,CHRCTR)                                           FCT00090
010        20 CONTINUE                                                         FCT00100
011           STOP 00                                                          FCT00110
012       100 FORMAT(11A1)                                                     FCT00120
013           END                                                              FCT00130
      C                                                                        FCT00140
014           SUBROUTINE NEXT(NN,MM)                                           FCT00150
015           DIMENSION MM(120)                                                FCT00160
016           MMM=0                                                            FCT00170
017           DO 10 I=1,120                                                    FCT00180
018           J=121-I                                                          FCT00190
019           MM(J)=MM(J)*NN+MMM                                               FCT00200
020           MMM=MM(J)/10                                                     FCT00210
021        10 MM(J)=MM(J)-MMM*10                                               FCT00220
022           RETURN                                                           FCT00230
023           END                                                              FCT00240
      C                                                                        FCT00250
024           SUBROUTINE PRINT(NN,MM,SYMBOL)                                   FCT00260
025           DIMENSION MM(120),SYMBOL(11),A(120)                              FCT00270
026           I=1                                                              FCT00280
027         1 IF(MM(I))2,2,3                                                   FCT00290
028         2 A(I)=SYMBOL(11)                                                  FCT00300
029           I=I+1                                                            FCT00310
030           GO TO 1                                                          FCT00320
031         3 DO 10 J=I,120                                                    FCT00330
032           K=MM(J)+1                                                        FCT00340
033        10 A(J)=SYMBOL(K)                                                   FCT00350
034           WRITE(6,100)NN,A                                                 FCT00360
035           RETURN                                                           FCT00370
036       100 FORMAT(6H FACT(,I3,3H) =,120A1)                                  FCT00380
037           END                                                              FCT00390

FACT( 70) =                    11978571669969891796072783721689098736458938142546425857555362864628009582789845319680000000000000000
FACT( 71) =                   850479588567862617521167644239926010288584608120796235886430763388588680378079017697280000000000000000
FACT( 72) =                  61234458376886086861524070385274672740778091784697328983823014963978384987221689274204160000000000000000
FACT( 73) =                 4470115461512684340891257138125051110076800700282905015819080092370422104067183317016903680000000000000000
FACT( 74) =                33078854415193864122595302822125378214568325182093497117061192683541123570097156545925087232000000000000000
FACT( 75) =               248091408113953980919464771165940336609262438865701228377958945126558426775728674094438154240000000000000000
FACT( 76) =              1885494701666050254987932260861146558230394535379329335672487982961844043495537923117729972224000000000000000
FACT( 77) =             14518309202828586963407078408630828498374037922420835884678157468806199134915642008006520786124800000000000000
FACT( 78) =            113242811782062978314575211587320462287317495794882519900489628256688353252342007662450862131773440000000000000000
FACT( 79) =           894618213078297528685144171559831652069808216779571907213868063227837990693501860533361810841010176000000000000000
FACT( 80) = 7156945704623490229481153372413653216558465734236575257711094450582270392554801488426689448672808140800000000000000000


*** HITFOR STOP 00 ***
```

図 9　出力例

```
                C     ERROR MESSAGES                                          ERR00010
          001         DIMENSIO A(20)                                          ERR00020
??ERROR 270
          002         READ(5,100)A                                            ERR00030
          003         TOTAL=0.                                                ERR00040
          004         DO 10 I = 1,20                                          ERR00050
          005         TOTAL=TOTAL+A(I)                                        ERR00060
??ERROR 146   (I)
          006         STOP                                                    ERR00070
          007         WRITE(6,200)A,TOTAL                                     ERR00080
??ERROR 106
          008     100 FORMAT(20F4.1)                                          ERR00090
                  200FORMAT(1H ,19(F4.1,2H +),F4.1,2H =,F5.1)                 ERR00100
          009         END                                                     ERR00110
??ERROR 300
??ERROR 301
??ERROR 307    10
??ERROR 307    200
          010         END                                                     ERR00120
??ERROR 300
??ERROR 302
COMPILED WITH 009 ERRORS
```

図10 エラーメッセージ例

HITAC-H150 HITFOR (02-00) ERROR FILE DUMP DATE 09/07/73

ERROR NAME = 120,180 PROGRAM NAME 1 = 007 PROGRAM NAME 2 = 110,169

	111	112	113	119	121	122	126	131	132	133	134	135	141	148	151	152	153	154	159	161	162	164	165	166	167	TOTAL
120	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	3
121	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1
122	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1
123	0	0	0	0	0	5	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	2	0	0	9
124	0	7	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	2	0	0	11
130	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
131	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
132	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
133	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
134	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
135	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
140	0	0	0	0	0	3	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4
141	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
142	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
143	0	6	0	0	0	6	2	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	3	0	0	19
144	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
145	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
146	0	0	0	0	0	2	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5
147	0	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	13
148	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
149	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
150	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
151	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
152	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
153	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
154	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
155	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
156	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
160	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1
161	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
162	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1
163	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
164	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
170	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
171	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
172	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
173	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
174	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
175	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
176	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
177	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
178	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
179	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
180	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TOTAL	0	26	0	0	0	16	5	1	0	0	4	1	2	4	0	0	0	1	1	1	0	0	7	0	1	70

図11 エラー統計データ

```
      C     MATRIX MULTIPLICATION                                          MTRX0000
001           DIMENSIONA(40,40),B(40,40)                                   MTRX0010
002           READ(5,100) N                                                MTRX0020
003           READ(5,200) ((A(I,J),J=1,N),I=1,N)                           MTRX0030
004           WRITE(6,300)((A(I,J),J=1,N),I=1,N)                           MTRX0040
005           DO20I=1,N                                                    MTRX0050
006           DO20J=1,N                                                    MTRX0060
007           S=0.                                                         MTRX0070
008           DO10K=1,N                                                    MTRX0080
009        10 S=S+A(I,K)*A(K,J)                                            MTRX0090
010        20 B(I,J)=S                                                     MTRX0100
011           WRITE(6,300)((B(I,J),J=1,N),I=1,N)                           MTRX0110
012           STOP 33                                                      MTRX0120
013       100 FORMAT(I2)                                                   MTRX0130
014       200 FORMAT(20F4.1)                                               MTRX0140
015       300 FORMAT(/(7X,20F5.1))                                         MTRX0150
016           END                                                          MTRX0160
```

```
 1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0
 1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0  1.0

 6.0  6.0  6.0  6.0  6.0  6.0  6.0  6.0  6.0  6.0  6.0  6.0  6.0  6.0  6.0  6.0  6.0  6.0  6.0  6.0
 6.0  6.0  6.0  6.0  6.0  6.0  6.0  6.0  6.0  6.0  6.0  6.0  6.0  6.0  6.0  6.0
```

```
*** HITFOR STOP 33 ***
```

図 12 行列積

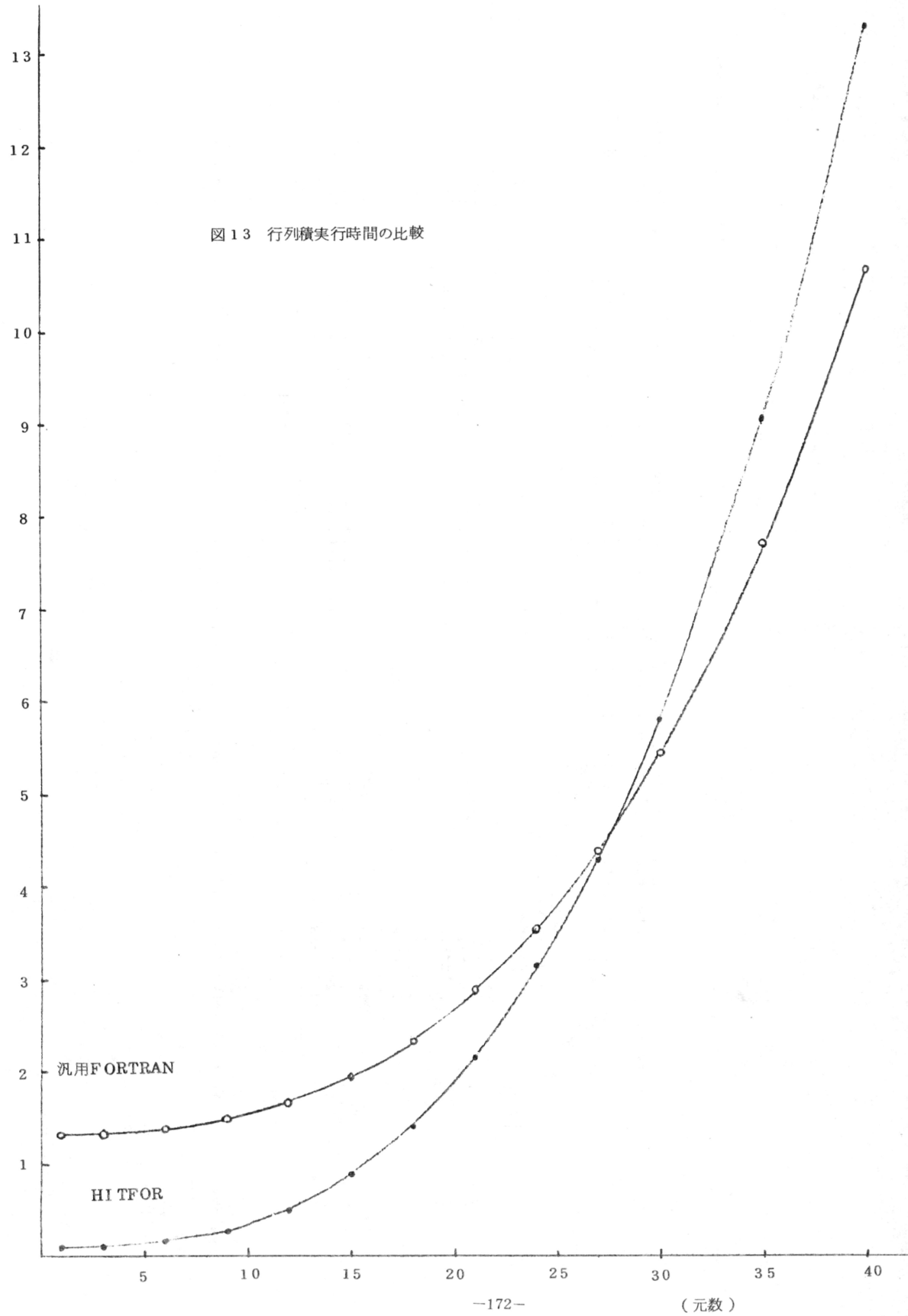

図13　行列積実行時間の比較

ｎｘｎの行列Ａを入力し，Ｂ＝Ａ^2 を求めて出力するプログラムと結果を図１２に示した。ここでＡの要素はすべて１，０，ｎ＝６としてある。ここでｎを変えると，当然この計算に要する時間は変化する。その所要時間をHI－TFORと，同じ8150で動く汎用のFORTRANコンパイラの両者で実行させ，測定した結果を図１３に示す。後者の所要時間の中にはコンパイルとリンケージエディットに要する時間が定数項として含まれている。実行時間が長くなると，両者の大小関係が逆転するのは，(a)実行時のエラーチエックにＨＩＴＦＯＲが時間をかけているため，と(b)翻訳時プログラムを小さくしようと努力したために，目的プログラムの能率が落ちているためと思われる。

教育用コンパイラの翻訳速度が速いことだけを取りあげて他と比較するのは，実のところ，公平を欠いている。というのは前者は後者に比して一般に多くの機能を欠いているからである。特に，HITFORは主記憶の制限が強かったこともあって，つぎの機能をもたない，身軽なシステムとなっている。

(1) 原プログラムの保持

(2) 目的プログラムの保持

(3) 各プログラム単位の単独翻訳

(4) 各種の運用，診断情報の出力

HITFORの設計，製作にあたって，日立製作所のコンピュータ第二事業部第一システム部，コンピュータ第一事業部教育センター部，ソフトウエアー工場小型プログラム部，システム開発研究所，および旭工場の皆様には種々の御援助をいただいた，ここに深く感謝する。

参考文献

1) Moriguchi, S., H. Kameda, D. Miura, H. Ishida, K. Yajima, and J. Tsunekawa, "A Comparativ Evaluation of Fortran Processing Systems," Proc. UJCC, PP. 153-160, 1972.

2) Rosen, S., R. A. Spurgeon, and J. K. Donnelly, "PUFFT——The Purdue University fast FORTRAN translator," CACM, vol. 8, no. 11, PP. 661-666, 1965.

3) Shantz, P. W., R. A. German, J. G. Mitchell, R. S. K. Shirley, and C. R. Zarnke, "WATFOR—The University of Waterloo FORTRAN IV Compiler," CACM, vol. 10, No. 1, pp41-44, 1967.

4) 森口繁一，"教育実習用ミニコンピュータのコンパイラ開発コンクールについて，"第１３回プログラミングシンポジウム報告集，ＰＰ．２４１－２６６，１９７２．

5) 島内剛一，筧捷彦，辻尚史，"FOTRANの実際"，サイエンス社，１９７３．

6) 日刊工業新聞，１９７３年９月１８日．

7) 日本工業標準調査会，"電子計算機プログラム用言語FORTRAN(水準3000)ＪＩＳＣ６２０１，"日本規格協会，１９７２．

8) "HITAC8150教育用FORTRAN HITFORプログラミング文法，"日立製作所．

9) "HITAC8150教育用FORTRAN HITFORオペレータ・ガイド，"日立製作所

本 PDF ファイルは 1965 年発行の「第 6 回プログラミング―シンポジウム報告集」をスキャンし、項目ごとに整理して、情報処理学会電子図書館「情報学広場」に掲載するものです。

この出版物は情報処理学会への著作権譲渡がなされていませんが、情報処理学会公式 Web サイトの https://www.ipsj.or.jp/topics/Past_reports.html に下記「過去のプログラミング・シンポジウム報告集の利用許諾について」を掲載して、権利者の捜索をおこないました。そのうえで同意をいただいたもの、お申し出のなかったものを掲載しています。

過去のプログラミング・シンポジウム報告集の利用許諾について

情報処理学会発行の出版物著作権は平成 12 年から情報処理学会著作権規程に従い、学会に帰属することになっています。

プログラミング・シンポジウムの報告集は、情報処理学会と設立の事情が異なるため、この改訂がシンポジウム内部で徹底しておらず、情報処理学会の他の出版物が情報学広場 (=情報処理学会電子図書館) で公開されているにも拘らず、古い報告集には公開されていないものが少からずありました。

プログラミング・シンポジウムは昭和 59 年に情報処理学会の一部門になりましたが、それ以前の報告集も含め、この度学会の他の出版物と同様の扱いにしたいと考えます。過去のすべての報告集の論文について、著作権者（論文を執筆された故人の相続人）を探し出して利用許諾に関する同意を頂くことは困難ですので、一定期間の権利者捜索の努力をしたうえで、著作権者が見つからない場合も論文を情報学広場に掲載させていただきたいと思います。その後、著作権者が発見され、情報学広場への掲載の継続に同意が得られなかった場合には、当該論文については、掲載を停止致します。

この措置にご意見のある方は、プログラミング・シンポジウムの辻尚史運営委員長 (tsuji@math.s.chiba-u.ac.jp) までお申し出ください。

加えて、著作権者について情報をお持ちの方は事務局まで情報をお寄せくださいますようお願い申し上げます。

期間：2020 年 12 月 18 日 〜 2021 年 3 月 19 日
掲載日：2020 年 12 月 18 日

プログラミング・シンポジウム委員会

情報処理学会著作権規程
https://www.ipsj.or.jp/copyright/ronbun/copyright.html