FORTRAN プログラムのフローチャート作成（I）

青江俊夫・尾高好政・一村　稔

岡山理科大学応用数学科

（昭和55年9月29日受理）

1. はじめに

情報処理部門における大きな問題の1つに，プログラムの生産性向上とプログラムの品質改善の問題がある。この問題の解決には，設計の段階から，十分な管理のもとに進められるべきことは言うまでもない。現在では，効果的プログラム開発技法（ＩＰＴ），ストラクチャード・コーディング，トップ・ダウン・プログラミング，プログラム・モジュール化，複合設計等種々の手法の開発が行われてきた。これらの作業過程には，「文書化」と呼ばれる作業が含まれる。この作業は，情報処理システムを開発するメンバー間のコミュニケーションに，あるいは，保守用にも必要なものである。特に，プログラムに関しては，文書化の1つにフローチャート（流れ図）の作成は，ぜひ必要なものである。フロチャートは問題のアルゴリズムを用いて，データを中心にした処理を流れ図として簡明に表現したものである。プログラマは，フローチャトを基にしてプログラムを作成すると共に，将来の保守の際の重要な資料として使用する。

一般にフローチャートは，コーディングの前段階に描くものであるが，ごく少数の場合を除いてそれが最後まで適用されるのは希で，通常は，問題の一部訂正が行われたり，プログラマの考え違いなどによりフローチャートはしばしば修正を余儀なくされる。しかし，その都度全面的にフローチャートを訂正するのは時間を要し，非常に面倒な作業である。したがって，フローチャートを手軽に作成・入手する方法があれば，プログラマの仕事量が大幅に軽減される。

そこで我々は，FORTRAN 言語でコーディングされたプログラムに対するフローチャートをラインプリンタ上に作図する **FORTFLOW** システムを作成した。

2. FORTFLOW の概要

FORTFLOW の構成は大別して，

(1) コントロール・ルーチン

(2) 構文解析ルーチン

(3) 作画ルーチン

の各ルーチンから成り立っている。各々のルーチンは，複数個のルーチンで構成されている。

2．1　コントロール・ルーチン

コントロール・ルーチンは，**FORTFLOW** システム全般の処理の流れを管理すると共に，実行時に必要な各種のオプション指定を解析・処理するルーチンである。

2．2　構文解析ルーチン

フローチャートを描く時に必要なことは，それぞれの FORTRAN 文(ステートメント)にどんな流れ図記号を当てはめるか決定することである。そのため，各ステートメントがどんな種類のステートメント（例ば，算術代入文，**IF** 文，**DO** 文，**GOTO** 文，入出力文，

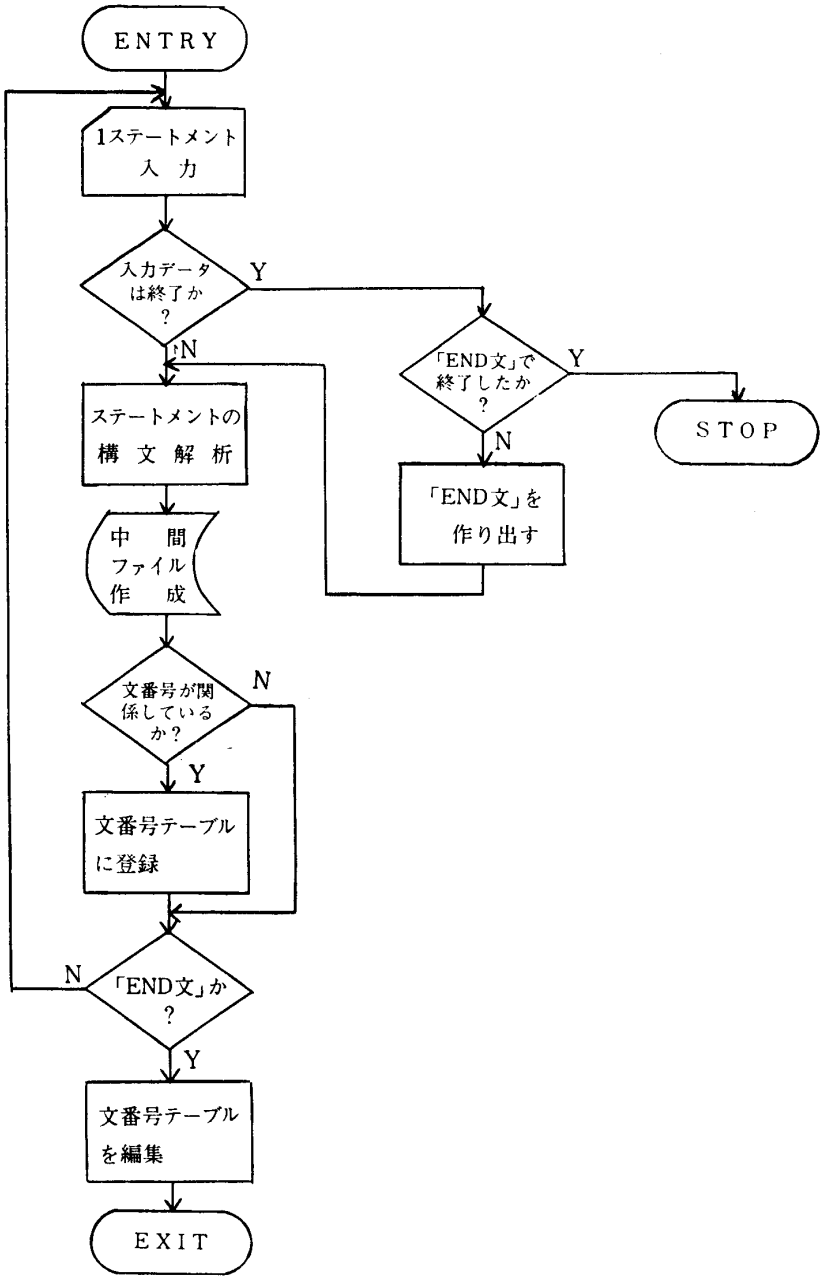


図 1.　構文解析ルーチンのブロック・チャート

など）であるか判別する。構文解析は，FORTRAN コンパイル時に，文法上の誤りの無いプログラムに対してのみ行う。構文解析ルーチンの概略は次のようである。また，処理のブロックチァートを図1に示す。

（a） 1ステートメントを入力

入力ルーチンにより，1ステートメントが入力される。本システムでは，プログラム単位ごとに処理を行うので，**END** 文かどうかを判定し，**END** 文であれば(e)の処理に移る。そうでなければ，そのステートメント内において文法上の意味を損なわない限り不要な空白を削除し，有効な文字のみに圧縮する。

（b） キーワードとの比較

FORTRAN 言語の文法上には予約語（特定の目的以外への使用を認められない語。）が存在しない。しかし，構文チェックの都合上，キーワード・テーブルを準備しておく。このテーブルには，キーワード自身とその文字列の長さ及び文の種類を示す識別番号が設

表 1. キーワード・テーブル

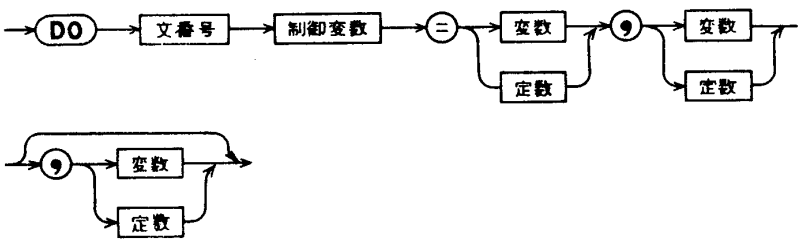
キーワード	キーワードの長さ	識別番号
ABNORMAL	8	21
ACCEPT	6	171
ASSIGN	6	201
⋮	⋮	⋮
COMMON	6	22
COMPLEX	7	41
CONTINUE	8	91
DATA	4	23
⋮	⋮	⋮

定されている。識別番号は，後の作画ルーチンにおいて，流れ図記号を決める際に利用される。

(a)で処理されたステートメントは，キーワード・テーブルを利用して最初のn文字（nは比較するキーワードにより可変）が比較され，等しくなければテーブルが終るまで順次要素との比較が続けられる。ここで，どのキーワードとも一致しない場合は，代入文とみなして(d)の処理に移る。

（c） 構文チェック

(b)でキーワードと等しいものが見つかれば，それに関する文法規則に一致するかどうかを調べる。例えば，次に示す構文規則は，**DO** 文のものである。



本システムでは，この通りにチェックするのではない。何故なら，前にも述べたように，データとして入力する FORTRAN プログラムには文法上のエラーは無いからである。そこで，本システムでは，上記の構文規則のうち○印の付いた「=」と「,」の2つについてのみ調べれば十分なのである。

一般に，FORTRAN 言語において，「=」が用いられるのは，

1. 代入文
2. **DO** 文
3. 入出力文の入出力並び
4. **FORMAT** 文
5. リテラル
6. 文関数定義
7. 注釈文（Comment 行）

の7通りである。このうち，3.と4.の場合には必ず括弧で囲まれた中に存在するし，5.の場合には「=」が独立した文字として出て来ることは無い。また，7.の場合は，最初から処理の対象から除外されている。次に，1.の代入文では，「,」は1対の括弧の中で用いられるのみである。文関数定義においても，代入文と同様に考えられる。したがって，「=」と「,」が独立した文字として現われれば **DO** 文と解釈できる。

このように考えて，本システムにおいて構文解析ルーチンでは詳しい文法を調べなくて，

表 2. 中間ファイルのレコード形式

シーケンス番号
ステートメントの識別番号
定義された文番号
テキスト部の継続指示
テキストの最初の文字の位置
テキストの最後の文字の位置
参照する文番号のテーブル内でのポインタ
テキスト部（最大66文字）
第1カラムの文字（空白，C，X，S，etc.）
論理 IF 文の指示

特殊文字のみに注目してステートメントの構文チェックを行っている。

（d） 中間ファイルの作成

(c)段階で文の種類が決ったら，後の作画ルーチンで必要な情報を中間ファイルとして一時的に磁気ディスク上に出力する。このファイルは，後の作業の都合上キー付のファイルとする。

（e） 文番号の処理

GOTO 文や **IF** 文等で通常の制御の流れを変る必要がある時，流れ図中でもそれが表現できるように線で結ばなければならない。そのために，文番号が定義されたり参照された時点で必要な情報を文番号テーブルに登録する。このテーブルには，現在のシーケンス番号，文番号，それの使用方法，すなわち，**GOTO** 文か，**DO** 文か，**IF** 文かと言ったことが解る情報と共に作られる。（表３－１）

表3. 文番号テーブル

（１） テーブル作成時点

シーケンス番号	文番号	テーブル内のポインタ	文番号の使用方法
1	-100	1	11
2	100	2	4
2	100030	2	5
3	100030	3	0
5	300	4	1
5	100	4	2
5	200	4	3
6	200	5	0
6	100	5	8
7	300	6	0
8	100	7	0
8	-100	7	10
⋮	⋮	⋮	⋮

（２） 編集終了時点

シーケンス番号	(作業用に使用)	流れ線の位置	文番号の使用方法
1	6	-2	11
2	5	1	4
2	1	0	5
3	0	0	0
5	4	-1	1
5	5	1	2
5	3	0	3
6	0	0	0
6	5	1	8
7	0	-1	0
8	0	1	0
8	0	-2	10
⋮	⋮	⋮	⋮

このテーブルは，１プログラム単位のデータが処理された後に流れ線で結合し易い形に編集される。（表３－２）

以上の５段階が，構文解析ルーチンで行われる主要な処理である。

2．3 作画ルーチン

FORTFLOW システムは，フローチャート（流れ図）をラインプリンタに出力する。また，流れ図は，見易くするため２頁続きで描かれ，各ステートメントの種類ごとの記号

と共に，制御ステートメント等により現在位置より前方あるいは後方で定義された文番号を参照する場合を示す流れ線が描かれる。この行先を変更する情報は2．2で作った文番号テーブルに含まれている。

図形作画ルーチンの処理は，図2に示す。

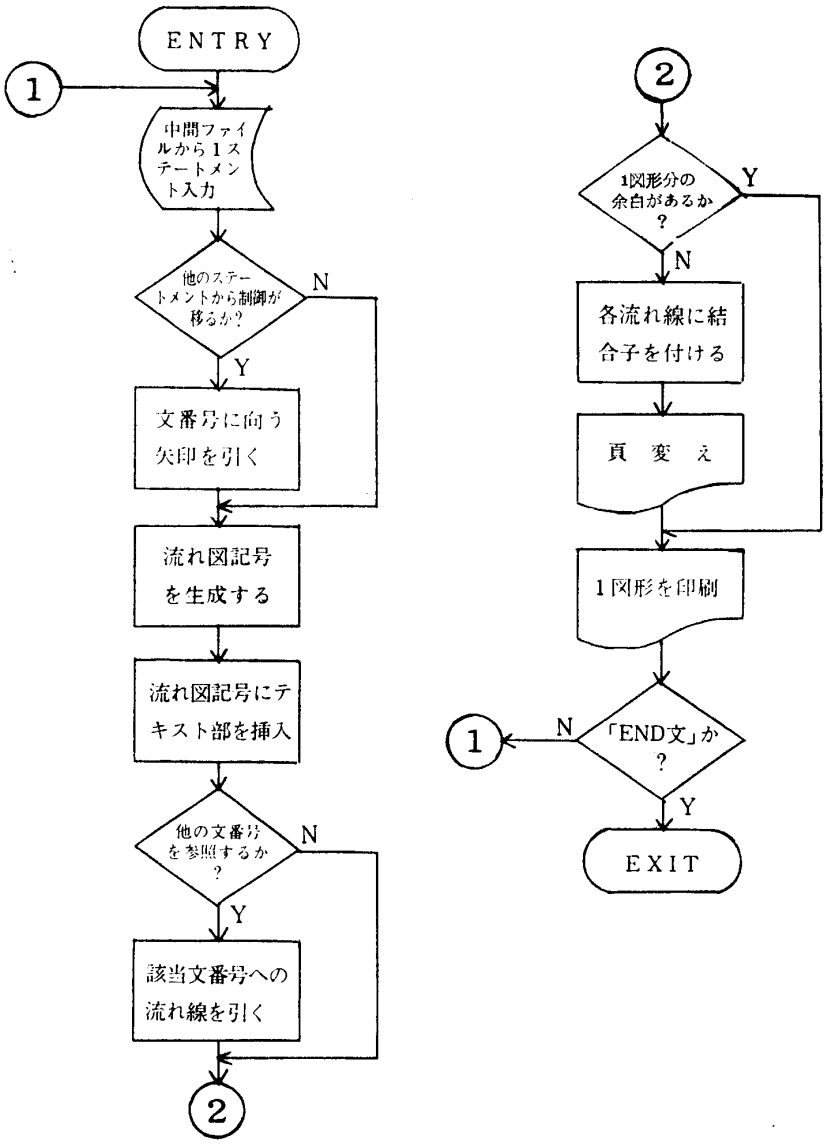


図 2．作画ルーチンのブロック・チャート

（a） ファイル入力

2．2で出力した中間ファイルから1レコードを入力する。このステートメントが持つ文番号に，他のステートメントから制御が移る場合には，流れ線から文番号に向う矢印を描く。

（b） 図形（流れ図記号）を作る

各ステートメントに対応する図形のデータは，予め表の形で用意しておく。（表4，5）図形を生成する時には，まず入力したレコード中のステートメントの識別番号と，表5中

の同番号の部分を探し，そのデータを使用して表4の分解された図形データから必要な情報を集めて図形を組み立てる。

表 4. 分解した図形データ表

番号	分解した図形データ
1	・・・・・・・・・・※・・・・・・・・・・
2	: :
3	: ABNORMAL :
4	: COMMON :
⋮	⋮
49	・※ ※・
50	・※ ※・
51	・※ ※・
52	※・ ・※
53	※・ ・※
54	※・ ・※
55	※・ ・※
⋮	⋮

表 5. 流れ図記号データ

識別番号	分解した図形データのポインタ									
11	20	25	26	28	29	30	21	0	0	0
21	1	2	3	2	19	21	0	0	0	0
22	1	2	4	2	19	21	0	0	0	0
23	1	2	5	2	19	21	0	0	0	0
61	30	49	50	53	54	55	22	21	0	0
71	30	49	50	51	52	53	54	55	22	21
81	21	21	21	0	0	0	0	0	0	0
131	56	57	58	58	59	21	0	0	0	0
134	64	65	66	67	67	68	54	69	21	0
212	70	71	71	71	72	73	21	0	0	0
⋮	⋮									

（c） テキスト部のセット

(b)で作った流れ図記号の中に，テキスト部から必要な文字を選択して挿入する。また，場合により図形外にもテキスト部分の一部をセットする。流れ図記号の内外にどんな文字を何文字転送するかはステートメントによりそれぞれ異なるので，この部分の処理は，ステートメントの識別番号により該当部分に分岐して処理が行われる。テキスト部が複数レコードに分かれている場合は，それらが終了するまで入力され，テキスト部の継続として処理される。

（d） 文番号の処理

現在処理中のステートメントが制御ステートメント等で，他の文番号を参照する場合には，流れ線でその文番号と結び付ける。その際には，文番号テーブルの該当するシーケンス番号を探して流れ線を描く位置を決める。次にその位置に，すでに流れ線が有るかどうかを調べて，もし有ればその線と結合し，無ければ新しく流れ線を作る。また，現在のステートメントが文番号を持っている場合には，その番号を流れ図中に書き出す。

（e） 図形の印刷

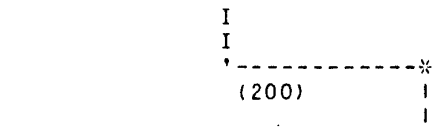
本システムでは，流れ図記号の印刷は全部まとめて出力するのではなくて，1ステートメント分の流れ図記号を単位として出力する。

複数頁にわたる流れ線には適当に，結合子（connector）を入れる。

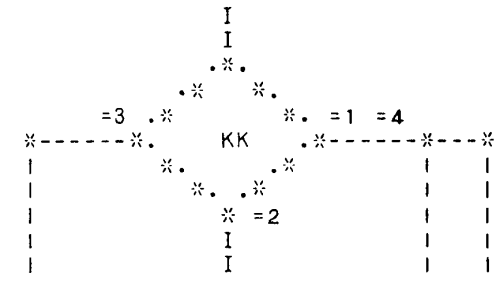
3. 代表的シンボル

3.1 GOTO 文

GOTO 文には，単純 **GOTO** 文，計算形 **GOTO** 文，割当て形 **GOTO** 文の3種類がある。単純 **GOTO** 文は図3(1)の流れ図記号で表わす。計算形と割当て形 **GOTO** 文については，



(1) 単純GOTO文



(2) 計算形及び割当て形GO TO文

図 3. **GOTO** 文の流れ図記号

次に制御が移るべき文番号の位置を表わす値が計算された変数名を菱形の中に書き入れ，

それぞれの文番号への流れ線を引く（図 3(2)）。流れ図記号中の＝1，＝2，……というのは，分岐先を表わす文番号を左から順に示す。

3．2 DO 文

DO 文は図 4 のように描く。最初の行に，制御変数の変化が理解しやすいように **DO** 形並びを，次の行に「DO」と端末文の文番号を示す。次の図形から **DO** の範囲が始まる。最後に，端末文の文番号とステートメントの内容を印刷する。

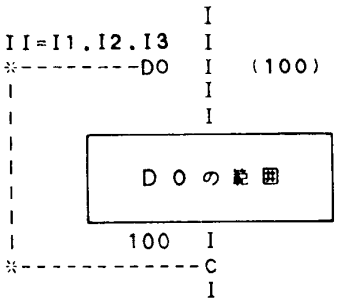


図 4．**DO** 文の流れ図記号

3．3 IF 文

IF 文には，算術 **IF** 文と論理 **IF** 文の 2 種類が有る。算術 **IF** 文では，算術式を評価した値によりゼロより小さいか，ゼロか，ゼロより大きいかによって 3 方向に分岐するのであるから菱形の 3 つの頂点を利用して流れ線を引く。その際，各頂点が正，0，負のどれ

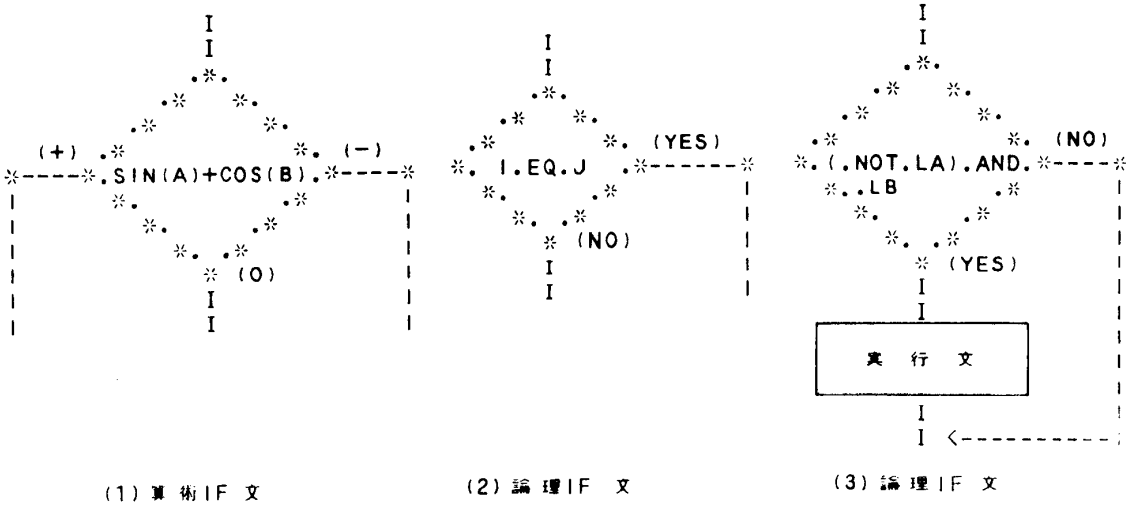


図 5．IF 文の流れ図記号

に対応するかを，（＋），（0），（－）で示すことにする。菱形のどの頂点が正，0，負のいずれに対応するかは，周囲の状況により変化する。

論理 **IF** 文は，与えられた論理式が評価されて，真か偽かの 2 方向に分岐をする。論理値が真の場合実行するステートメントが **GOTO** 文ならば，図 5(2)の形になる。それ以外の場合には，実行ステートメントを菱形の下に配置し，偽の時の流れ線を菱形の横の頂点から発して，再配置した実行ステートメントの下に持って来るように変形する。（図 5(3)）

3．4 入出力装置

本システムでは入出力装置の流れ図記号として，カード読取装置，ラインプリンタ装置，磁気ディスク装置，磁気テープ装置，システム・コンソール，またはTSS端末装置の５種類が用意されている。FORTRAN 言語において入出力装置に対するファイルの構成には，シーケンシャル・ファイルとランダム・ファイルの２種類がある。上記入出力装置のうち，シーケンシャル・ファイルにはすべての装置が対応するが，ランダム・ファイルには磁気ディスク装置のみが対応する。プログラム上の命令では，ファイル構成別の命令があるのでランダム・ファイルについては装置の割当てがすぐにできるが，シーケンシャル・ファ

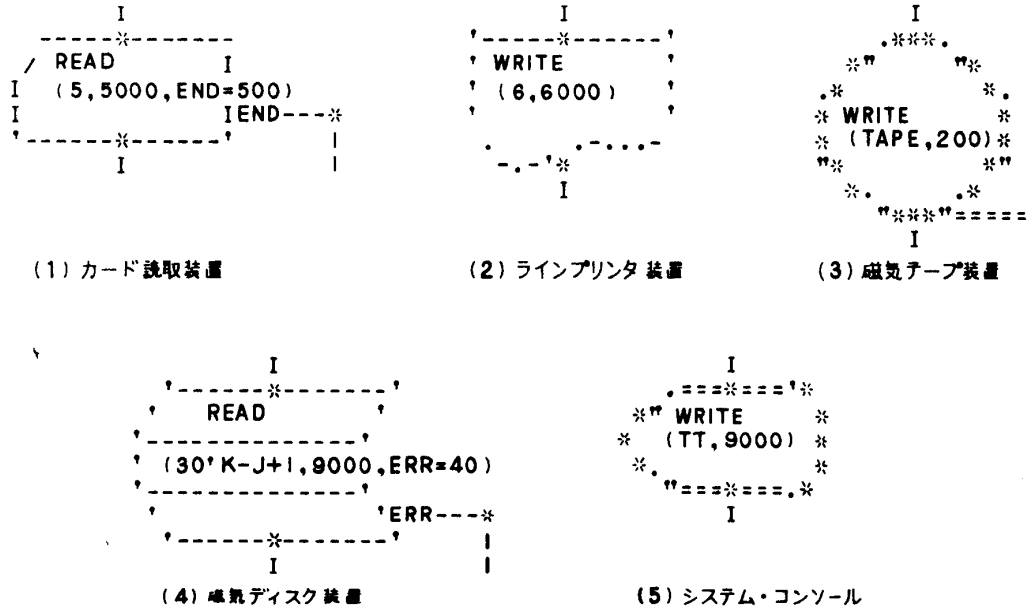


図 6. 入出力装置の流れ図記号

イルにはそれができない。そこで入出力装置の種類を表わす装置番号，または，変数名をコントロール・ルーチンで予め登録しておいてその装置番号，または変数名が出て来るごとにどの装置であったかを探して流れ図記号を決定する。何も指定が無い時は，入力装置としてはカード読取装置が，出力装置としてはラインプリンタ装置が標準的に仮定される。

4． 結果の検討

我々は，本システムの性能評価のためにいくつかのテスト用プログラムを用意して実験を行った。この実験には，プログラム内での各ステートメントの種類ごとの使用頻度に著しい偏りが生じてはならないので，テスト用に特別に作成したプログラムではなく，実際の運用に供しているプログラムを使用した。

まず，プログラム単位ごとでの実験を行った。その結果，処理するプログラムの実行ステートメントのカード枚数と，出力する頁数及び処理時間の間には，ほぼ正比例することが確認された。

次に3種類のプログラムについて，テストを行った。データAは，各プログラム単位のカード枚数がいずれも多い場合，データBは逆にカード枚数が少ない場合，データCはそ

表 6. テスト・データの比較

	全実行ステートメント数	プログラム単位数	累積出力頁数	累積処理時間
データ A	1320枚	4個	106頁	9分16秒
データ B	1279枚	26個	110頁	7分30秒
データ C	1444枚	8個	118頁	9分29秒

れらが混っている場合であるが，全カード枚数はいずれもほぼ等しい。カード枚数に対する出力頁数と処理時間の関係を図7，8に示す。この2つのグラフから，出力頁数及び処理時間は，プログラム単位数には無関係で，カード枚数に依存していることがわかる。ただ，処理時間に関して言及すれば，カード枚数が同じでもプログラム単位数が多いほど累積処理時間が少なくなる点に注目したい。このことは，構造化プログラミングやプログラムのモジュール化が進められることが，流れ図作成，保守の面でも有利であることを示すものである。

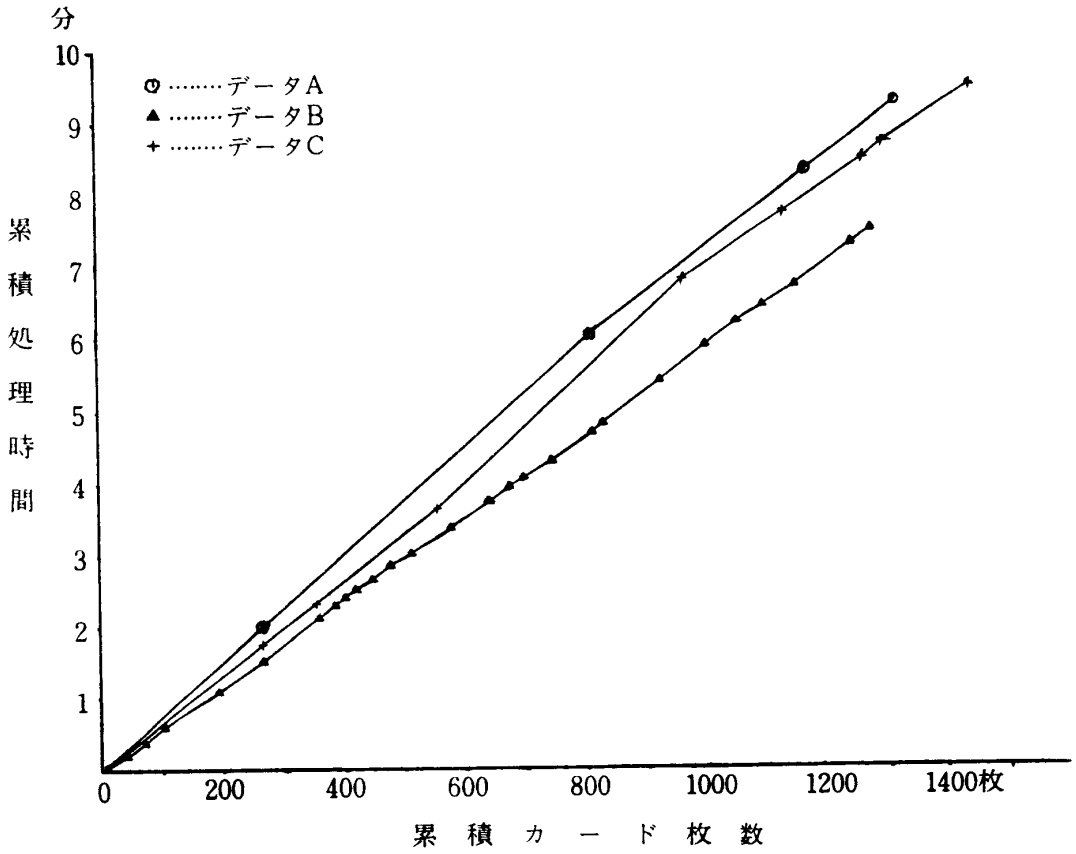


図 7. 処理時間の比較

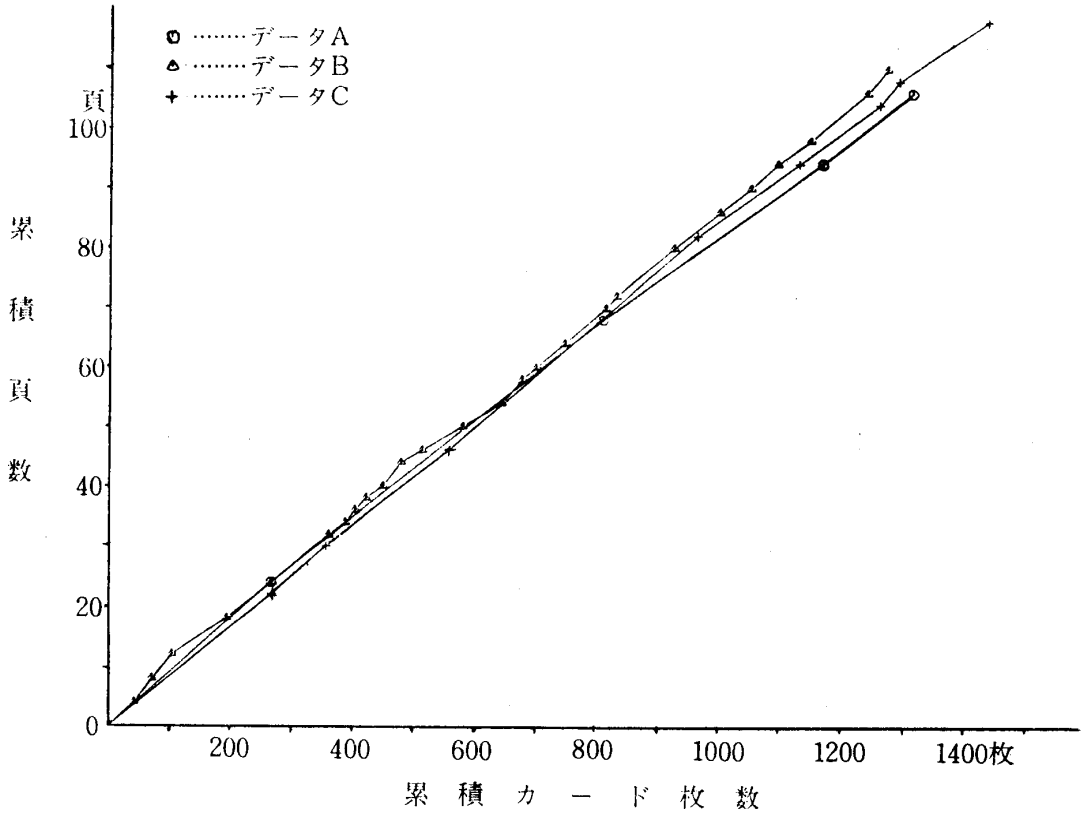


図 8. 出力頁数の比較

5. おわりに

本システムの作成目標は，一応達成することができた。しかし，構文解析ルーチンには，まだ手直しを必要とする所が何か所かある。例えば，ユーザーに対して合理的なプログラムを作るために，データとして入力したプログラムのステートメントの配置変えを指示するようなシステムに改良したい。

現在本システムと並行して，PDP 11/45 に附属しているX—Yプロッタ上にフローチャートを作図するシステムも作成中なので，次回はそのシステムについて報告するつもりでいる。また，構造化プログラミングを行ったものと，従来のプログラム手法とによる流れ図の比較検討も行いたい。

参考文献

1) 前川 守：自動フロー・チャーティング，情報処理，Vol. 9, No. 3, pp. 131-137 (1968).
2) 山本，山口：自動フローチャーティング，情報処理，Vol. 11, No. 12, pp. 711-720 (1970).
3) Krider, Lee.: A Flow Analysis Algorithm, Jour. ACM, Vol. 11, No. 4, pp. 429-436 (1964).
4) 菊地，高橋，吉岡，菅野：FORTRAN 自動フローチャート出力システム，情報処理学会第20回大会講演論文集，pp. 287-288 (1979).
5) MELCOM FLOW 7，三菱電機株式会社 (1974).

System for Flow Charting of FORTRAN Programs (I)

Toshio AOE, Yoshimasa ODAKA and Minoru ICHIMURA

Department of Applied Mathematics Okayama University of Science, Ridaicho 1-1, Okayama 700, JAPAN

(Received September 29, 1980)

Flow charts are one of the most important documentations in system developments and maintenance of data processing system. The system for flow charting of FORTRAN programs, FORTFLOW, is prepared, which generate flow charts on a line printer. These charts are useful for program modifications, communications between members of system development group and decrease of development costs. This system helps especially an inexperienced user of a data processing system to understand a problem.

FORTELOW are coded in FORTRAN language and are available for FORTRAN users by general purpose electronic computers.