C言語プログラム構造解析によるフローチャートの生成*

5AE－1

‡貞石裕司、‡葛崎偉、§斗納宏敏
‡山口大学教育学部
§富士通テン株式会社

1．はじめに

現在、ソフトウェアの開発における生産性や品質が問題となっており（ソフトウェア危機）、それらを向上させるための方法論が注目され、様々な提案が行われている[1]。

ソフトウェアの開発には、新規に開発する場合と、既存のソフトウェアを修正したり改良したりする場合がある。既存のソフトウェアの場合には、さらに、既存のプログラム要素1つ1つからソフトウェア全体の設計や仕様を再構築する必要があり、その上で追加修正部分を含めた全体が正しく仕様に合致することを確かめる作業が加わる。現実には、この作業に人間の多大な思考と労力と時間を要している。なぜなら、既存のプログラムとは、他人が書いたものや、自分が書いたものでも書いてから時間が経過しているものであるため、例え、そのプログラムに細かく注釈が付いていても理解には困難を要するからである[1]。

従って、ソフトウェアの開発において、プログラム要素1つ1つの仕様や設計を容易に理解できる技術が不可欠となる。その観点から、本研究において、仕様や設計を容易に理解するためのフローチャート作成に力点を置き、システム設計、ブロックの役割などに具体例を含めて概説し、最後に結論を述べる。

2．システム概要と設計

2.1 システム概要

本システムは、プログラムを解析し、フローチャートを生成することを目的とし、対象プログラムは、C言語プログラムとする。

C言語は関数型言語であり、その基本単位は関数である。各関数ごとの働きをフローチャートで表示すること（図1）により、プログラム全体の流れの理解を容易にすることができる。従って、本研究では関数解析を行うこととした。

なお、C言語プログラムにおいてgoto文、longjmp関数、setjmp関数は制御の流れを不規則に変えるため嫌われており、今回の解析の対象外とした。

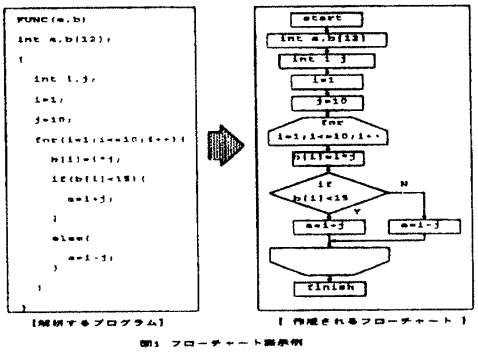
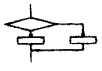


図1 フローチャート図例

2.2 フローチャート図式

フローチャート図式[2]とは、以下のものから構成されて、処理、分岐、ループはすべて端子間のパス上にあるものとする。

[処理]: 演算、もしくは演算群の実行などの任意の種類の処理機能を表す。



[分岐]: 1つの入口と幾つかの出口を持ち、条件に従って出口を選ぶ機能を表す。

[ループ]: 2つの部分からなり、ループの始まりと終わり表す。

本研究では、主に以上の図式を用いてフローチャート作成を行った。

2.3 システム設計

フローチャート作成のために、以下の手順で行う。

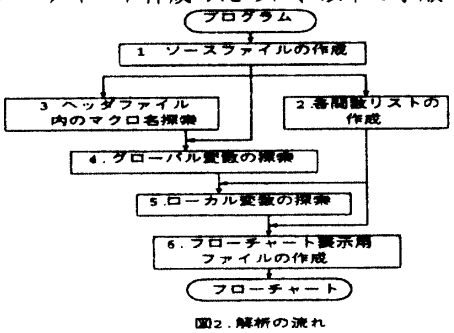


図2．解析の流れ

[ソースファイルの作成]:メイクファイルを解析し、ソースファイル名を取り出す。

[各関数リストの作成]:各関数ごとの内容をリストとして取り出す。

[ヘッダファイル内のマクロ名探索]:ソースファイルに関係のあるヘッダファイル内で宣言されているマクロ名を取り出す。

*Generating Flow-Chart for C-Program via Structural Analysis

‡Yuji Sadaishi ,‡Qi-Wei Ge and §Hirotoshi Tonou
‡Faculty of Education,Yamaguchi University,Japan
§Fujitsu Ten Co.,Ltd

[グローバル変数の探索]:ソースファイル、ヘッダファイル内で宣言されているものを、グローバル変数、構造体、マクロなどに分類する。

[ローカル変数の探索]:各関数内で宣言されているものをローカル変数、構造体などに分類する。

[フローチャート表示用ファイルの作成]:次章で説明。

3．フローチャート作成

フローチャートを作成するために、文字または文字列である”トークン” ごとに解析を行う（図３）。例えば、プログラム上に”for” というトークンが出現した場合、そのトークンが制御を表し、そこからループが始まるという事を認識しなければならない。そのため、関数ごとに分けられたプログラム上の１つ１つのトークンまで調べる必要がある。

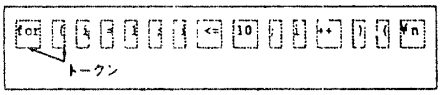


図3 トークン例

そのトークンの種類により、ブロックの役割やつながり（次章で説明）が決定され、それらからTcl/Tk[3]を用いてフローチャート表示用ファイルが作成される。

4．ブロックの役割とつながり

4.1 ブロックの役割

ブロックの役割の決定は、以下のようにして行う。

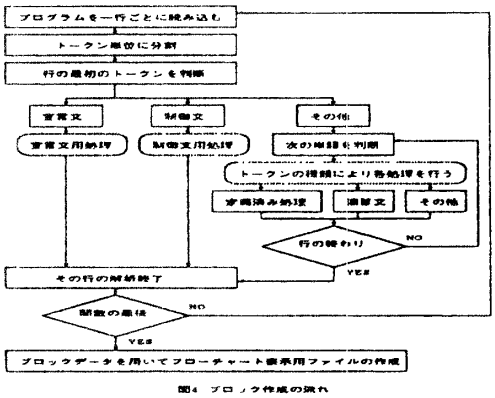


図4 ブロック作成の流れ

以上のようにして、読み込んだ１行の分類を決定し、分類が制御文、定義済み処理以外で直前のブロックの分類と同じであるなら読み込んだ１行は直前のブロックの一部となる。そうでないなら、その行は単独で１つのブロックを形成し、ブロックに番号（ブロックナンバー）を付けて次の行の解析に移る。ここで、ブロックナンバー、ブロック分類、ブロック階層（次項で説明）の３個のデータをブロックデータとして持たせておく。

4.2 ブロックのつながり

ブロックのつながりは、基本的にはプログラムの記述の上下関係であるが、制御文が出現すると、その上下関係が崩される。そこで制御文が現れた場合を考える。

Ｃ言語の制御文は、goto 文を除くと、ループ(for,while,do)、スキップ(break,continue,return)、分岐(if,switch)処理の３パターンに分けられる。ループ処理における制御の流れは、ループの終端、つまり、有効範囲を示すことで上下になる。スキップ処理の場合は制御の行き先のみに注意すればよく、有効範囲は考慮しなくても良い。分岐処理においては、制御の行き先と有効範囲の両方を考慮しなければならない。つまり、ブロックのつながりの解析に必要なデータは、制御文の行き先と有効範囲である。

ここで、ブロックデータを調べる。ブロック階層とは、ブロックが含まれている制御文の数のことであり、ブロック階層が１増えるごとに各階層の制御の種類を記録させ、ブロック階層が１減る時がその制御の終端となる。故に、制御の有効範囲の解析が可能である。

制御の行き先については、ブロック階層により導くことが出来る。ブロック分類がスキップ、分岐処理を表す場合以外はブロックのつながりはブロックナンバーの順番となる。スキップ、分岐処理の場合はそのブロック分類ごとに制御の行き先を探す。

従って、ブロックデータを調べることで、ブロックのつながりを認識することができるのである。

5．おわりに

以上に述べた手法を用いて実際にシステム設計とコーディングを行い、Ｃ言語プログラムを関数ごとにフローチャートを作成することができ、プログラム理解とデバッギングを容易にさせることができた。今後の課題としては、よりプログラムの流れが理解し易くなるための、日本語が表示されるフローチャートの作成などが挙げられる。

参考文献

[１] 秋山 義博, 他:ソフトウェア開発支援システム; システムと制御, 第31巻, 第９号,pp.653～659(1987)

[２] 室 章治郎, 他:プログラム理論; システムと制御, 第22巻, 第６号,pp.331～339(1978)

[３] 浅野理森:はじめてのTcl／Tk, 技術評論社(1995)

[４] 葛崎偉, 他:自動車用電子機器プログラムの自動生成システム－ APES; 富士通テン技報, Vol.11,No.3,pp.43～52(1993)