5K－5　言語Cプログラミングにおける学習者の行動分析

玉木裕二，並木美太郎，高橋延匡
(東京農工大学　工学部　電子情報工学科)

1．はじめに

我々は，研究室内における言語Cプログラム教育用のシステムの研究，開発を行っている[1]．このシステムでは，学習者が計算機上で演習形式で学習を行い，計算機が教師となり，適切なときに適切な指導を行えるようにすることを目標にしている．このためには，学習者はどのようなときに，どのように考え，どのような行動をとるかが明らかになっていなければならない．

今回，学習者の行動分析の第一歩として，学習者が出したコンパイルエラーと，学習者のプログラムの編集の過程などを，学習者の行動データとして記録し，その解析を行った．その結果，プログラミングについて特徴的な点が判明したので，報告する．

2．学習者の行動分析の目的

計算機を用いた教育システムでの本質的な問題の一つに，コースウェアの設計をどうするかということが挙げられる．そのためには，学習者が，どのようなときにどのような誤りを犯すか，あるいは，どのような誤りを犯すときどのような行動を取るかなど，誤りをモデル化し，学習者モデルの作成を先行させる必要がある．すなわち，これらの学習者モデルをもとにコースウェアの作成を考えなければ，実用に耐えるものを実現することは難しい．

我々は，このようなモデル化を行うことを目的として，学習者がプログラミングを行っているとき，どのようなときに，どのような考え，どのような行動を取るか，その行動データを収集し，解析を行う．

3．学習者の行動分析の方法

我々は，上記の目的のもとで学習者の行動分析を行うが，まず，その第一歩として，学習者が出したコンパイルエラーと，学習者のプログラムの編集の過程などを，学習者の行動データとして記録し，その解析を行った．ここでは，データとして記録した内容，記録した内容の解析方法について述べる．

3．1 データ収集の対象

データの収集は，我々の研究室に新しく配属された学部生を対象に行った．我々の研究室では，上記の学生を対象に，言語C勉強会を開いているが，この学生は，計算機科学の基礎知識を持っているが，1万行を越えるプログラミング経験がなく，また，言語Cによるプログラミング経験は少ない．

本年度の勉強会では，住所録プログラムや正規表現を含む文字列マッチングを行うプログラムなどを演習問題として出題したが，学習者がこれらのプログラムを作成しているときの行動をデータとして収集した．

なお，データを記録するときは，プログラムの編集，コンパイルなどを普通に行うだけで，自動的に記録する環境を構築し，学習者には，データを取っていることを意識させないようにした．

3．2 記録したデータ内容

データの記録には，追記型光ディスクを用いた．追記型光ディスクは，記憶容量が数百メガバイトと大きいため，データの収集を長期的に行うことができる．

記録したデータは，次に示すとおりで，ファイルにログを取ることで行った．

(1) ログイン，ログアウトした時刻とユーザ名
(2) コンパイルを開始した時刻
(3) コンパイラがオープンしたファイル名
(4) コンパイラが出力したエラーコード

3．3 データの解析方法

OS／omicron第2版[2]上に実現されている，追記型光ディスクの仮想的な書換えによる世代管理機能[3]により，ソースプログラムをファイルに書き込んだ各世代の，内容のすべてを保存することができる．各世代間の差分を取ることにより，学習者がソースプログラムをどのように編集したか分かる．このデータと，3．2で述べたデータとの関連を調べるツールを開発し，解析を行った．また，必要に応じて，各世代のソースプログラムを読み，または実行して動作の確認を行うという作業を行った．

4．学習者の行動分析の結果

ログデータの記録を，今年度の4月から行っているが，これらのうち，学習者A，B，Cについて解析を行った．解析の対象としたプログラムは，表1に示すとおりである．

これらのデータを解析した結果，プログラミングについて，以下の特徴的な点が判明したので，それについて具体的に述べる．

(1) 学習者Aの場合

この学習者は，住所録プログラムを作成していたが，図1に示すような流れで開発を行っていた．まず最初に，キーボードから名前，住所を入力し，それを配列の先頭から格納するというテスト的なものを作成した．その段階で，プログラムの正常動作を確認した後，新たに機能を追加するというプログラミングを，表2に示すように4回に分けて行っている．これから，学習者Aは，設計をせずにコーディングを開始し，思い付きでプログラムを作成していたことがうかがえる．

表1　解析の対象としたプログラム

開発者	期間	開発プログラム	行数	コンパイル回数
学習者A	4/22 ～ 4/30	住所録	258	70
学習者B	4/20 ～ 4/30	住所録	440	105
学習者C	6/25 ～ 7/ 2	文字列マッチング	824	90

表2　学習者Aの作成したプログラムの機能拡張

回数	機能拡張の内容	コンパイル回数	総行数
1	一覧表示機能の追加	4	68
2	検索機能の追加	6	84
3	削除機能の追加	21	116
4	構造体の使用	35	258

An Analysis of the Learner's Behavior in Language C Programming
Yuji Tamaki, Mitaro Namiki, Nabumasa Takahashi
Tokyo University of Agriculture and Technology

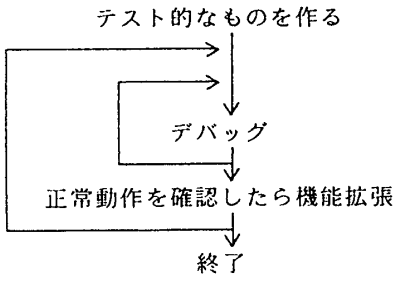

図１　学習者Ａの開発の流れ

(2) 学習者Ｂの場合

学習者Ｂも住所録プログラムを作成していたが，そのプログラム中に，キーボードからの入力が“Y”かどうかチェックするルーチンがある．このルーチンのデバッグを図２に示すようにプログラムを書き換えていた．これから，このルーチンのデバッグをしていた初期の段階では，文字列の使い方を充分理解していなくて，試行錯誤的にデバッグを行っていたことがうかがえる．

```
char ans ;
scanf("%s", &ans) ;
if(ans == 'Y')
        ↓
char ans ;
scanf("%s", &ans) ;
if(ans == "Y")
        ↓
char ans ;
ans = getchar() ;
if(ans == "Y")
        ↓
char ans[4] ;
scanf("%s", &ans) ;
if(strcmp(ans, "Y"))
        ↓
char ans[4] ;
scanf("%s", ans) ;
if(strcmp(ans, "Y") == NULL)
```

図２　学習者Ｂのデバッグの流れ

(3) 学習者Ｃの場合

学習者Ｃは，正規表現を含む文字列マッチングを行うプログラムを作成していたが，デバッグの初期の段階で，そのプログラムの中にデバッグ用のテストデータを埋め込んでいた．このため，テストデータを変更するだけの目的の再コンパイルを５回行っていた．

最後に，11枚の追記型光ディスクに記録されたログデータについて，本年度の４月から７月までの４箇月の間に記録されたコンパイルエラーメッセージの集計を行った．総コンパイル回数5954回のうち，頻度の多かったエラーメッセージを表３に示す．なお，このエラーメッセージは，本研究室で開発，実用している，言語ＣコンパイラCAT [4] が出力したものである．

５．考察

学習者に共通して，ポインタに関する間違いが多かった．これには，文字列の扱い方，ポインタ変数にアドレスを代入する前に使用するなど様々であるが，学習者のレベルによって，どのように間違えるかを定式化できれば，それをもとにポインタを，段階的に理解させるコースウェアを作成できるだろう．

表３　出現頻度の多かったエラーメッセージ

エラーメッセージ	回数
未定義データである	1875
式文(式;)のセミコロンがない	873
式の終わりがおかしい(最後が演算子になっている)	711
外部定義/宣言の終わりにセミコロンがない	702
プロトタイプ宣言と呼び出しが一致しない	450
コンパイルがEND OF FILEで終わらない	444
メンバ演算子'->'の被演算子が構造体型でない	358
複文が'}'で終らない	343
#defineの2重定義(最新の定義で置き換える)	317
指定されたファイルをオープンできない	301
外部定義/宣言中での変数の二重定義	236
メンバ演算子'.'の被演算子が構造体型でない	205
演算子が位置的におかしい	188
if()の')'がない	131
sizeof(型名)の型がおかしい	129

また，今回の解析で，学習者に特徴的なプログラミングとして，設計を行わずにコーディングを開始する傾向があるということが判明した．また，デバッグの仕方として，チェックプリント文の挿入によるデバッグが大部分を占めていたが，このチェックプリント文を網羅的に挿入していた．これは，デバッグ時のことを考えたコーディングを行っておらず，デバッグでは，明確なチェックポイントを持っていないからであると思われる．

問題の本質は，ソフトウェアのライフサイクルという概念が欠如していることであると考えている．つまり，ソフトウェアのライフサイクルを念頭に置いた開発を行えるように教育することが重要であり，そのようなコースウェアを開発しなければならない．

今回の解析手法では，学習者がプログラムを編集するとき，ソースプログラム中のどの部分を，どの位注目していたかなどの，どの位考えていたか計測することができない．このため，エラーメッセージとプログラムの編集作業の関係を，深く調べることができなかった．このようなデータを取るためには，キーボードレベルで行動を記録する必要があるだろう．

また，表３に示したコンパイルエラーメッセージは，必ずしも同じエラーを同定しているとは限らない．エラーメッセージとプログラムの編集の関係を調べ，その結果を用いて，エラーメッセージの設計を行うことも興味深い．

５．おわりに

今回の解析では，２節で述べた目的を達成するには至らなかったが，現在，学習者の行動をさらに詳しく記録できるような，プログラム実行環境の開発を行っている．今後は，この実行環境を実現し，さらに詳しく学習者の行動分析を行って行っていく予定である．

参考文献

[1] 森田雅夫他：“言語ＣのCAIシステムの設計とそのプログラム実行環境の開発”，情処学会プログラミングシンポジウム報告集，情報処理学会，1991．1

[2] 高橋延匡：“研究プロジェクト総説OS／omicronの開発”，情処学会オペレーティングシステム研究会報告39－５，1988．5

[3] 横関隆他：“追記型光ディスクの仮想的な書換えと世代管理機能の実現”，電通学会論文誌Ｄ－Ｉ　Vol．J72－D－I No．6，pp．414－422，1989．6

[4] 並木美太郎他：“OS／omicron用システム記述言語Ｃ処理系Catのソフトウェア工学的見地からの方式設計”，電通学会論文誌Ｄ　Vol．J71－D　No．4，pp．652－660，1988