6A-6

# プログラミングにおけるソース文の変化特性分析

土田康太† 水穂良平‡ 浜名隆広‡ 佐藤匡正†
島根大学総合理工学部† 島根大学総合理工学研究科‡

## 1. 序論

プログラミング教育の主目的のひとつとして，学習者が問題を正確に解釈し，プログラムを作成する能力を養うことが挙げられる．現状の教育方法は団体一律の指導が主流であり，学習者の個々の状況の把握は十分とは言えない．したがって，副題や宿題によるプログラムを分析することによって学習者の理解の程度を把握することができる．しかし，この手法によって学習者個別のプログラミングにおける設計能力の把握は難しく，個人指導につなげるには不十分である．

学習者が学生の場合，プログラミング過程は解法とコーディングを区別しない傾向が見られる．ほとんどの場合，コーディングを行いながら問題の解決法を模索する．このため，学習者から提出された最終版プログラムによって，問題の解釈から解法へつなぐ道筋を知ることは難しい．効果的な個別指導を行うには，この道筋を知ることが重要である．

この道筋を知るために，プログラムの作成過程から学習者の問題解決の考え方を個別に把握することを試みる．この試みには，プログラム作成者の操作過程および解法の認知過程から分析する考えがある[1][2]．この考えでは，プログラミング中の鍵盤の操作法と発話を記録して，その変遷を分析し，個々の作成者がどのように問題の解法を導いているかを知る．しかし，この記録にはソース文との関係などが含まれていないため，学習者の考え方が正確に把握できない可能性がある．

本稿では，問題解決の考え方を把握する新たな試みとして，複数のプログラムの作成開始から完成に至るまでに生じるソース文をすべて記録し，ここにおいて生ずる処理および論理についての変化に着目する．この変化を分類し，頻度を計測することにより，その特性を把握する．

## 2. 実験方法

問題の解釈から解法へつなぐ道筋を把握するために，次のプログラミング演習を実施する（表1）．

### 2.1. 概要

6名の情報系大学生の被験者を用意し,3題の演習課題についてプログラミングを実施する．最終版に至るまでソース・プログラムの全員分の全版を保存する．これらの全版について被験者ごとに，コンパイル回数および後述する構造要素と機構要素における変化を分析する．

表1 演習概要

| 演習項目 | 内容 |
|---|---|
| 演習題 | 3題 |
| 被験者 | 6名（情報系大学生） |
| 使用言語 | C言語 |
| 演習方法 | 質疑：言語仕様，コンパイラ機能に限定<br>完了：予め用意した試験データによって完了検査，不備な場合は修正後に再検査 |

以下に，課題のあらましを示す．(出題順)

**(1) 16進dump**

入力したファイルを16バイト単位で区切って画面に表示する．表示する項目は,「番地」,「16進表示列」,「文字列」である．

**(2) 英文編集**

入力ファイルの英文に含まれる項番号，行つなぎ(-)などについて自動編集を行い,結果をファイルに出力する．

**(3) 電子透かし**

指定したビットマップ形式ファイルの末尾から，入力した文章を1バイトずつ下位1ビットに埋め込み，新たなビットマップ形式ファイルに出力する．また,埋め込んだ文章を抜き出し,画面に表示する．

### 2.2. 変化計数法

ソース文は,便宜的に行(LOC)や文によって計数される．ここでは設計につながる変化を知るために，文の分類を試みる．プログラムを，解法の枠組みを与える構造およびこの構造に一意性を与える機構から成る[3]と考える.構造を構成する文を構造要素とし，機構を構成する文を機構要素とする．

任意の前版と現行版を比較して変化のある文を検出し，次の区別をして構造要素と機構要素に分けて計数する．

追加：前版に対応する文が存在しない文の数
削除：前版に存在する文で現行版に対応する文が存在しない文の数
更新：前版および現行版において対応のとれる文のうち，内容に変化のある文の数

現行版に対して，同時に追加および更新を生じ得

A Transition Analysis of Source Code in Programming
†Tsuchida Kouta, Satou Tadamasa・Interdisciplinary Faculty of Science and Engineering, Shimane University
‡ Mizuho Ryouhei, Hamana Takahiro・Interdisciplinary Graduate School of Science and Engineering, Shimane University

表 2　変化度合のパターン

| | 16 進 dump | | | | | | 英文編集 | | | | | | 電子透かし | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | A | | B | | C | | A | | B | | C | | A | | B | | C | |
| | 処理 | 論理 | 処理 | 論理 | 処理 | 論理 | 処理 | 論理 | 処理 | 論理 | 処理 | 論理 | 処理 | 論理 | 処理 | 論理 | 処理 | 論理 |
| コンパイル回数 | 0.88 | | 0.70 | | 1.23 | | 0.42 | | 1.19 | | 1.76 | | 0.52 | | 1.05 | | 1.05 | |
| 追加 | 0.07 | 0.00 | 0.37 | 0.27 | 1.88 | 2.03 | 0.20 | 0.00 | 0.57 | 0.36 | 2.52 | 2.97 | 0.24 | 0.25 | 0.42 | 0.25 | 1.60 | 2.40 |
| 削除 | 0.02 | 0.00 | 1.62 | 1.25 | 2.55 | 2.50 | 0.14 | 0.16 | 0.76 | 0.16 | 3.31 | 4.16 | 0.07 | 0.00 | 1.88 | 0.43 | 1.33 | 2.17 |
| 更新 | 0.38 | 0.12 | 0.71 | 1.62 | 0.28 | 2.12 | 0.35 | 0.44 | 1.40 | 1.16 | 1.76 | 0.08 | 0.35 | 0.25 | 2.10 | 1.15 | 0.75 | 0.51 |

表中の数値は平均値を 1 とした値

表 3　全変化の比較

| | A | B | C |
|---|---|---|---|
| コンパイル回数 | 0.59 | 1.00 | 1.36 |
| 追加 | 0.17 | 0.42 | 2.11 |
| 削除 | 0.13 | 1.06 | 2.75 |
| 更新 | 0.34 | 1.38 | 1.43 |

| 前版 | 現行版 | |
|---|---|---|
| | a = 0; | 追加（構造要素） |
| a = b + c; | a = b + c; | |
| ・・・ | | |
| if(a == 0){ | if(a != 0){ | 更新（機構要素） |
| b = 0; | b = 0; | |
| } | } | |

図 1　計数例

る．この場合の計数値は両者の和とする．

計数例を図 1 に示す．

## 3.　分析結果と考察

6 名のデータは 3 パターンに分類できた．これらの代表値を A，B，C として表 2 に示す．また，表 3 は全演習題に対して全変化の比較を示している．

### (1)　言語理解の格差

C 言語の記法等に関する質問はあがらなかった．よって，プログラミング言語の最低限の知識は差がないと考えられる．

### (2)　全体の傾向

本例では，異なる 3 つのプログラム作成を実施したが，表 3 より被験者に一貫した傾向が表れた．A は全項目に関して，平均値を下回っており，C が最もソース文の変化が著しいといえる．これは被験者の技能および問題解決の考え方を含めた総合的なプログラミング能力が反映されていると推察できる．

### (3)　設計の意識

学習者 A は最初のコンパイル時に解法の実装が完了し，細かな修正を行うことで完成に至っている．これは，コーディング前に紙面上で問題の整理を行い，解法と作業手順を明確にしてからコーディングを行っていたことに起因していた．これに対して，学習者 B,C はソース文中に大きな変化がみられることから，コーディング中で問題の解法を考えていることが分かる．設計とコーディングを工程として明確に意識することで，ソース文の変化度合に差が表れることが確認できた．

### (4)　学習者の設計能力

学習者がコーディング前に行う問題解釈の程度によって，各頻度に高低差が発生している．学習者 A の問題解決の考え方は，ソフトウェア開発の設計方法と類似すると考えられる．他の学習者は，設計方法自体の知識の不足から，問題の解法をコーディング中に模索している可能性がある．

## 4.　結論

学習者の問題解決の考え方を個別に把握することを目的とし，プログラム作成過程におけるソース文の変化に着目して分析を行った．変化をプログラムの機能単位として，処理および論理の追加，削除，更新に分類し，その頻度を計測した．その結果，異なる課題に学習者毎で共通の傾向が表れ，学習者の問題解釈の程度によりソース文の変化が発生した．また，これが学習者のプログラミング能力によって発生していることが確認できた．

**参考文献**

[1]　前田恵三，中野靖夫：プログラム作成過程の分析，日本教育工学雑誌，Vol.19，No.3，pp.171-180，1995

[2]　前田恵三，中野靖夫：プログラミング過程の認知面からの考察，電子情報通信学会技術研究報告.ET，教育工学，Vol.98，No.443，pp.29-36，1998

[3]　Satou Tadamasa，Kishimoto Yorinori：Program Code Analysis Focused on its Structure，WSEAS TRANSACTION on COMPUTERS，Issue 1，Volume 2，pp24-29(2003)