プログラミング技術診断システム　～文法知識の診断に向けて～

A Programming Technique Diagnostic System: toward Diagnosing Grammatical Knowledge

今泉俊幸† 橋浦弘明‡ 井上悠一‡ 古宮誠一‡
Toshiyuki IMAIZUMI† Hiroaki HASHIURA‡ Yuichi INOUE‡ Seiichi KOMIYA‡

† 芝浦工業大学　工学部
‡ 芝浦工業大学大学院　工学研究科
† Faculty of Engineering, Shibaura Institute of Technology.
‡ Graduate School of Engineering, Shibaura Institute of Technology.

要旨

近年，ソフトウェア開発は大規模化，複雑化している．このような状況から，企業はソフトウェア開発に必要な知識やスキルを持つ人材の育成を大学に求めている．ソフトウェア開発に必要な技術の中でもプログラミング技術は必要不可欠なものである．プログラミング技術を効率よく習得させるためには，教授者が学習者の理解状況(欠陥)を正しく把握する必要がある．しかし，プログラミングは目的に対し無数の解が生じるため，学習者にどのような技術が不足しているかを判断することは難しい．また，一般に学習者は教授者よりも人数が多いため，教授者の手間の面からも困難である．本稿では，自動的にプログラミング技術の欠陥を診断する方法を示す．

1. はじめに

プログラミング技術はソフトウェア開発に必要不可欠なものである．しかし，プログラミング技術を習得させる際には問題点がある．それは，プログラミングは高度な知的作業であり，解が一つに定まらないため，人によって理解できない場所が異なるという点が挙げられる．そこで教授者は，学習者ごとに理解の度合いを正しく把握し，個人個人に適切な指導を行う必要がある．しかし，学習者の理解の度合いを把握することは，人によってコードの書き方が異なることや間違えた理由の推測が難しいということなどから困難である．そのため，どのような指導をすれば良いか判断することも難しい．また，一般的に学習者は教授者よりも人数が多いため，教授者一人の手間の問題からも非常に困難である．

2. プログラミング技術診断システムの提案

以上で述べた問題に対処するため，自動的に学習者のプログラミング技術の欠陥を診断するシステムを提案する．このシステムを用いることで，教授者は学習者にどのような指導をすれば良いか分かる．

2.1. プログラミング技術を習得する過程

学習者のプログラミング技術が，ソフトウェア開発を行うレベルに達するまでには，

① アルゴリズムの実現に必要なプログラミング言語の文法知識を習得
② アルゴリズムをプログラミング言語で表現する技術を習得
③ 目的に適したアルゴリズムを取捨選択し，それらを組み合わせて目的のプログラムを作成する技術を取得

という３つの段階が必要と考えた．

プログラムを作成するにはアルゴリズムの理解とプログラミング言語の知識の両方が必要である．しかし，学習者がプログラムを正しく作成できていない際に，ソースコードを見るだけではどちらの欠如が原因か分からない．よって，まずプログラミング言語の知識を判断するために，プログラミング言語の文法を使いこなすことができるか判断する．本研究では①の文法知識の習得に着目し，診断システムの最初のステップとして，文法知識を診断するシステムを開発する．

文法知識を習得している状態として，「使用する変数や処理を，具体的に指示されることで，目的のプログラムを実装することができる状態」と定義する．具体的な指示とは 図 1 のようなものである．

2.2. プログラミング技術診断システムの診断内容

2.2.1. 診断する対象

前節の③の技術を診断するための例として，ソートアルゴリズムの取捨選択問題がある．ソートのアルゴリズムは多様なものがあり，データの規模や特性により最適なアルゴリズムが異なるからである．本研究では，文法事項の学習が一通り終わり，アルゴリズムの習得を始めようとする学習者が最終的にはソフトウェア開発をできるレベルまでプログラミング技術が向上することを目的とする．その指針として，ソートのアルゴリズムを適切に選択できるようになるということを考えた．その為には，まず多くのソートのアルゴリズムをプログラムで実現できるようになる必要がある．ソートのアルゴリズムの中でも高速に動作し，よく使われるクイックソートを行うプログラムの実現に必要なプログラミング技術を診断することを目標とする．また，プログラミング言語は，プログラミングの学習の際に多くの教育機関で用いられている C 言語を対象として，その文法知識を診断する．

2.2.2. 診断の方法

学習させたいアルゴリズムに必要な文法を定義する．その文法事項の確認をするコードを書かせる問題を出題することで，学習者の文法知識を判断する．診断の流れを以下に示す．

1. 習得目的のアルゴリズムを決定する．
2. 必要な文法事項を定義し，問題，正答群を作成する．問題は調べる文法事項以外の要素が極力入らなくなるまで小さくする．たとえば if 文の知識を調べるならば，「変数 left が変数 right より小さいならば, left の値を 1 足せ」などといった問題を作成する．複数の正答を準備する理由として，「left<right」と「right>left」のようにソースコードは複数の書き方が考えられるからである．
3. 出題順序を設定する．変数の利用や演算などの最も基礎的な文法事項については，最初に出題し，以後注目しないこととする．
4. 作成した問題を学習者に対し出題する．
5. 学習者の解答と，作成してある正答に対し，本システムを用いて比較を行う．間違った解答をした問題について，その問題が測る文法知識については学習者の知識が足りないと判断する

3. プログラミング技術診断をする仕組み

ソースコードは作成者によって，同じ目的のプログラムを記述しても，内容が異なる．この問題を回避するために，学習者が解答した C 言語のソースコードの構文木を，XML 形式に変換して解答と比較する(図 2)．変換の際，変換前のソースコードに含まれる情報は失われないようにする．

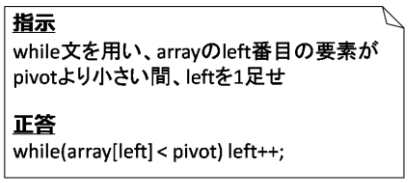

図 1 具体的な指示

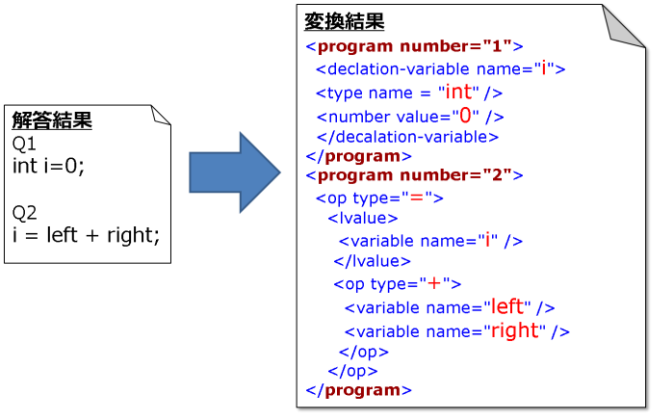

図 2 ソースコードを XML 形式に変換

プログラミング技術を診断する仕組みを図 3 に示す．構文木を作成するために，字句解析器生成ツールである flex[1]と，構文解析器生成ツールである bison[2]を用いる．まず flex/bison に対し，C 言語を XML 形式に変換するルールを入力として与え，C 言語のソースコードを XML 形式に変換するプログラムを生成する．プログラミング技術診断システムは，学習者が解答した C 言語のソースコードを入力として

受け取り，それを flex/bison によって生成されたプログラムを通じて XML 形式に変換する．そして，その XML と解答群として既に用意してある XML に対し XPath を用いて比較を行う．一致しなかった部分については指導が必要と判断し，教授者に対し学習者の能力の診断結果を出力する．

文法の誤りの多くはコンパイラが指摘してしまうが，この方法では，図 4 のようなコンパイラが検出しない誤り(意図しない動きをするコード)も発見することができる．

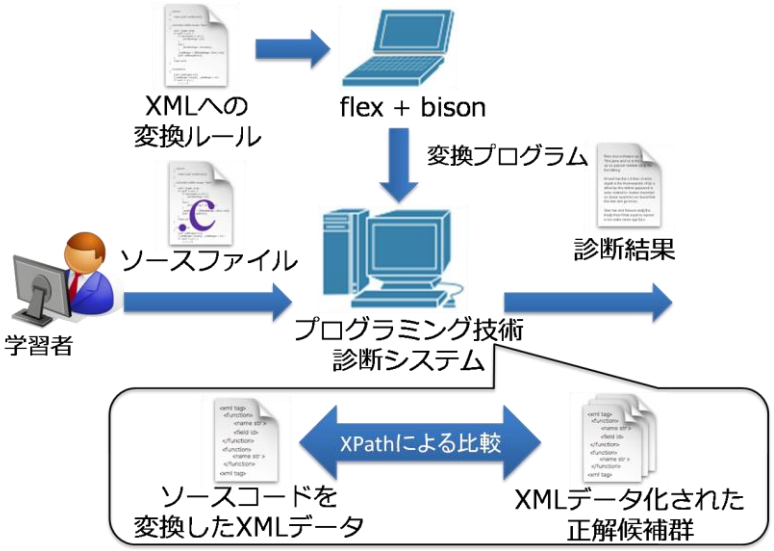

図 3 システムの動作イメージ

問題	誤答	正答
要素数5のint型配列arrayの 中身をfor文を用いて全て出力せよ	for(i=1;i<5;i++) printf("%d",array[i]);	for(i=0;i<5;i++) printf("%d",array[i]);

図 4 コンパイラが検出しない誤り

4. 問題の妥当性を調べる実験

上記で述べた方法で，文法知識を正しく診断することができるかを調べる実験を行った．

- 被験者
  - 芝浦工業大学工学部情報工学科 1 年生 1 名，2 年生 2 名
  - 芝浦工業大学システム工学部電子情報システム工学科 1 年生 5 名
- 内容
  - クイックソートのプログラムを具体的な指示に従うことで実装させる(以下指示問題)
  - 上記のプログラムを作成に必要と考えられる文法知識の問題を解答させる(以下文法問題)

出題した文法知識の問題は，表 1 のとおりである．解答の際 IDE を使用すると，学習者にヒントを与えてしまい正しい知識が計測されない．よって，解答はメモ帳(notepad)を用いて PC 上で記述させた．

被験者の C 言語の知識に関しては，1 年生は今回出題した問題のうち，関数，マクロ以外の学習は済んでいる．2 年生は出題した全ての文法知識を習得済みである．

表 1　出題内容

分類	項目	分類	項目
変数	宣言，代入，初期化，参照， 複数宣言	算術演算	和，差，割り算，計算順序， インクリメント
論理演算	<，<=	if	単文，複合文
while	単文，複合文	for	単文
printf	文字表示，変数表示	配列	宣言，代入，初期化，参照
マクロ	include	関数	宣言，定義，呼び出し

実験の結果，指示問題と，それを解くのに必要な知識を問う文法問題をともに間違った件数は 33 件あった．その内訳は以下の通りである(表 2).

表 2　知識不足を検出できた文法知識

項目	件数	項目	件数	項目	件数
変数の初期化	2	変数の複数宣言	2	インクリメント	3
printf の変数出力	1	配列の宣言	2	配列の初期化	4
配列の代入	3	配列の参照	2	for	1
関数の宣言	5	関数の定義	3	関数の利用	5

また，文法問題で正解しながら，指示問題で書けないケースは存在しなかった．このことから，文法問題で文法知識が正しく測れることを確認した．また，文法知識を計測する際に問題となる要素についても考察を行った．

- ケアレスミスの存在

プログラミング初心者は数行程度の短い処理であってもセミコロンや閉じ括弧の抜けが多いことがわかった．このようなコードを構文解析すると，構文エラーが生じ，正常に解析することができなくなってしまう．システムを用いて自動的に解析する際に，知識不足による構文エラーとケアレスミスによる構文エラーを区別することが必要である．発見されたミスと，その対策は以下のようにする(表 3).

表 3　ケアレスミスの種類と対策

種類	対策
文末のセミコロンの抜け	構文エラーが生じた際, セミコロンが予期された部分に自動的に補間する
複合文における閉じ大括弧}の抜け	構文エラーが生じた際, 大括弧が予期された部分に自動的に補間する
関数，変数名の書き間違い	行わない

- 意図が伝わりづらい問題の存在

プログラムは通常ある目的のために作成するものである．しかし，今回の文法問題では目的がないため，指示された内容を理解することができず，意図しないコードが書かれた．「i に 1 を加算しなさい」という出題に対し，「i++」，「i=i+1」といった解答を想定したが，「i+1」と解答する学生が複数人存在した．問題文の日本語はどのような表現が適当か調査の必要がある．

5. 今後の課題

本システムを利用することにより，指導が必要な学習者を発見することができ，その学習者が何の文法知識が不足しているかを正しく検出できるかどうかを調べる．そのために，本学の情報工学科 1，2 年生に対し文法問題を出題，解答に対し本システムを適用する．システムの解析結果と人手で判断した結果を比較し，システムがどれだけ正しく検出でき，どの程度教授者の負担が減らせるかを計測する．

6. おわりに

プログラミング技術を効率よく習得させるための問題として，学習者の能力の把握が困難であることを挙げ，対策として，学習者のプログラミング技術を自動的に診断するシステムを提案した．プログラミング技術を習得する過程として，三つの段階が必要と考え，最初の段階であるプログラミング言語の文法知識の計測を行う方法を述べた．システムを用いて文法知識の計測を行う際に生じる問題点について，実験を行い確かめ，システムの実装案を示した．

参考文献

[1] Flex: the Fast Lexical Analyzer,　2006.　http://flex.sourceforge.net/. (2008/11 現在).
[2] Bison: GNU parser generator,　2006.　http://www.gnu.org/software/bison/.(2008/11 現在).