6ZC-2

# 初学者のプログラムの自動正誤判定における部分点付与の提案

伊藤 亜樹† 丸山 一貴‡ 寺田 実†
電気通信大学† 明星大学‡

## 1 背景

近年のIT社会において，プログラミング学習者は増加しており，その学習方法も，一人で学ぶ場合と，講義などで他の人と一緒に学ぶ場合の大きく二つに分けられる．そしてプログラミング学習においては，プログラムの概念の理解だけでなく，実際にその作成の過程を繰り返すことも大事であり，どちらの学習方法であっても，学習意欲を保つことは重要である．

学習方法の一つとして，オンラインジャッジ [1] や，コンテスト [2] が挙げられる．そしてそのどちらも正誤の判定に重点が置かれている．しかし，完答が難しいことも多いプログラミングが不得意な学習者にとっては，それではハードルが高い．理由として，もしも完答できなかった場合，学習者は自分が提出したプログラムがどれくらい合っていたのかを一見して判断することは困難だからである．よって,「完全に合っているか否か」のみで判断するよりも,「どれくらい合っていたか」という判断基準も導入することで,「もう少しで完答できる」という思いから，学習者の学習意欲が高まるのではないかと考えた．

## 2 目的

前述より，本研究では，プログラミングが不得意な学習者，またはプログラミング学習を始めたばかりの人が書いたプログラムの正誤判定を行った場合に,「どれくらい合っているか」を求めることを検討することとした．

プログラムの解析方法には，プログラムを実行させて行う動的解析と，逆に実行することなく行う静的解析があるが，本研究では静的解析を取り扱う．静的解析を行うことによって，コンパイルはできるが実行時エラーは起きてしまうような，だが人間が見た場合に「惜しい」と思われるものに部分点が与えられるようになると考えられる．

Estimation of partial scoring for novice students' incorrect programs
†Aki Ito, Minoru Terada
The University of Electro-Communications
‡Kazutaka Maruyama
Meisei University

## 3 関連研究

プログラミング初学者のための研究には，さまざまなものがある．内田 [3] は，反復して演習することができるように穴埋め形式のドリルを自動生成するシステムを開発した．中村ら [4] は，プログラミング教育の入門サポートのために，PENというツールを考案した．

川崎ら [5] は，大学におけるプログラミング演習において，小コンテスト形式を提案している．そして一つの問題に対し，複数の予備テストと最終テストを用意し，段階的な実装を誘導する．最終テストでは完全な解答が要求されるが，予備テストでは，部分点が与えられる．たとえば，平均と最大値を求める問題で，平均だけの出力が合えばよいとしたり,“2.00”が正解の場合に“2”が出力された場合でも正解とする．また，実行結果のみに基づいて判定が行われ，ソースコード内部については問われない．ソースコードの類似度を判定するモジュールを組み込んでいるが，不正コピー防止のためであり，正誤判定には使われていない．

## 4 提案システム

### 4.1 概要

本システムでは，正解のプログラムと学習者のプログラムがどれくらい似ているかを，ある尺度を設け，それに従って静的解析によって判定する．プログラミングを学び始める場の一つである講義などで使用されることを想定しており，その教師が正解プログラムを用意することとする．また対象のプログラムはJavaとしており，Eclipseを使ってJavaで生成したAST（抽象構文木，Abstract Syntax Tree）にて比較を行っている．用いる尺度としては，ソースコード内部における制御構造に着目した．そしてその中でも，プログラムを学ぶ上で重要かつ初期の頃から頻繁に出てくる `while`, `for`, `if`，及び，その条件式のトップレベルに出現する比較演算子を比較対象とした．但し，右オペランドが定数の場合には右オペランドも比較対象とする．

### 4.2 アルゴリズム

表1は，ペナルティを再帰的に求めるアルゴリズムである．ここでの type とは，while，if，for のことを指す．但し，type が等しいかどうかを判定する際，while と for は同一と見なすとする．

表 1: ペナルティを求めるアルゴリズム

| |
|---|
| ・type が等しくない　→　10.0 |
| ・type が等しい場合<br>　・条件式の処理<br>　　・operator が等しくない　→　5.0<br>　　・右 operand が等しくない　→　5.0<br>　・本体の処理<br>　　・先頭から再帰的に比較　→　1/2n<br>　　・type の数が等しくない　→　5.0 |
| ・Importance が存在<br>　→　各々の Importance をかけていく |

### 4.3 重要度の設定

ソースコード1のように，多重ループが使われているプログラムの場合に，内側の率を設定することによって，ループの内側が間違っていてもペナルティは軽くなる．しかし，ソースコード1の場合で，11行目の，出力を行う for がない場合，ループの内側ではないため内側の率は適用されず，ペナルティは重いものとなる．ただ，正しい出力がされないものの，それ以外は合っており，人間が見た場合には「とても惜しい」といえる．そのため、それぞれの type に対して任意で重要度を設定できるようにした．たとえば，正解プログラムにおいて重要度を設定したい type のある行で“ // Importance 0.5 ”のように記述すればよい．

ソースコード1：整数配列 a[] が与えられたとき，その中の値を昇順に並べ替えて，その結果を表示するプログラム

```
1  int[] a={10,3,2,1};
2  for(int i=0; i < a.length-1; i++) {
3          for(int j=0; j < a.length-i-1; j++)
               {
4              if(a[j] > a[j+1]) {
5                      int s = a[j];
6                      a[j] = a[j+1];
7                      a[j+1] = s;
8              }
9          }
10 }
11 for(int i = 0; i < a.length; i++) {
12         System.out.print(a[i]+"␣");
13 }
```

## 5 実験と考察

実際に，二種類のプログラムに対し，誤ったプログラムをそれぞれ複数用意し，数名の被験者に10点満点で自由に採点してもらった．

その結果，ロジックが間違っていれば点数を与えないか大幅に減点していた．なお，一部の採点者から誤答プログラムの実行結果を見て回答したという記述があった．

先にも述べたが，本システムでは動的解析ではなく静的解析を行っている．そのため，ソースコード内部について判定は行うが実行結果については問わない．だが，予めロジックに対して重要度を設定することで，ある程度システムと人間の点数を近づけることは可能ではないかと考えられる．

## 6 今後の課題

今後の課題は，次の二点が挙げられる．

まず，初学者はプログラミングを行う上で，複雑な記述方法をしないという考えから，制御構造では一部の尺度しか用いておらず，すべてのプログラムの正誤判定には対応していない．今回比較対象としたもの以外でも，ソースコードのどのような部分に着目すれば有効なのかを，比較検討する必要がある．

次に，同様の考えから，構造が異なるプログラムの場合にも未対応で，正解プログラムを予め用意する必要がある．

## 参考文献

[1] AIZU ONLINE JUDGE Programming Challenge, http://judge.u-aizu.ac.jp/onlinejudge/.

[2] ACM-ICPC，http://icpc.baylor.edu/.

[3] 内田保雄，“初級プログラミング学習のための自動作問システム”,『コンピュータと教育』，情報処理学会研究報告 2007(123)，pp.109-113，2007.

[4] 中村亮太，西田知博，松浦敏雄，“高等学校での「プログラミング」教育の導入-PEN を用いて”,『コンピュータと教育』，情報処理学会研究報告 2008(42)，pp.41-47，2008.

[5] 川崎慎一郎，富永浩之，“競争型学習を取り入れた入門的 C プログラミング演習 - 実行テスト系列による部分採点のための柔軟な照合機能 -”,『コンピュータと教育』，情報処理学会研究報告 2010-CE-103(10)，pp.1-8，2010.