プログラミング演習における個別指導のための コーディング状況把握方法の提案
―進捗度の観点から―

2015SE018 五十里貴斗
指導教員：蜂巣吉成

1　はじめに

大学におけるプログラミング演習では，学習者数十名，指導者 1 名，TA 少数名という形式で授業が行われる事が多い．演習時間は限られているため，指導者 (以下，教員と TA を指導者と記述) は学習者に対し効率的に指導することが求められる．しかし，学習者の進捗には個人差がある．質問のために挙手した学習者に指導員が指導したり，指導者が教室を巡回して，ソースプログラムを読み，行き詰まった学習者を発見して指導したりすることが行われるが，演習時間内に少数の指導者が多くの学習者を効率的に指導することは難しい．

石元ら [1] はプログラミングにおいて重要な，変数の値を変化させるという観点に着目し，ソースコード中の演算と変数の型について同値類分割を行う方法を提案している．同値類分割を行うことで読むソースコードの総数を減らすことができるので，少数のソースコードから全体の進捗状況の把握を行う．演算が単純な問題については同値類分割はまとまるが，演算が長くなる問題については記述が複数あり，分割結果がまとまらないという課題がある．また，全体の進捗状況を把握することはできるが，学習者個人の進捗状況を把握できない．

本研究では，学習者の進捗度の観点から，学習者のコーディング状況を把握する方法を提案する．進捗度とは，学習者がどれだけ演習課題を解き進められているかを表したものである．学習者の記述したソースコードを [1] の方法を用いて正規化，抽象化したもの（以下，評価コードと記述）と，あらかじめ用意された模範解答の編集距離を比較し，その差を比較することで進捗を把握する．しかし，別解の場合には，解にたどり着いていても進捗が低いと捉えられてしまうことが考えられる．そこで，実行結果通りのプログラムができているかによっても進捗を把握する．実行例が正しく動作すれば，進捗していると言える．これらから，学習者ごとに進捗度を求め，指導が必要となる学習者を特定する．

2　関連研究

文献 [2] では，ソースコードの行などの 4 種類のメトリクスから生徒の学習状況を把握し，その状況に即した指導を可能としている．文献 [1] では，ソースコード中の演算と変数の型について同値類分割を行うことで，少数のソースコードから全体の進捗状況の把握している．

文献 [2] では，相対的に遅れている学生を特定する点，文献 [1] では，学習者個人の進捗状況を把握することができない点で，本研究とは異なる．

3　プログラミング演習における進捗度の提案

学習者がどれだけ演習課題を解き進められているかを表す，進捗度を定義する．進捗度は 0 から 100 とし，100 が完成した状態である．

文献 [1] の WebIDE を用いてソースコード，コンパイル情報，実行結果を収集し，ソースコードは 1 分ごとに，コンパイル情報はコンパイル時，実行結果は実行時にそれぞれ保存するものとする．

3.1　編集距離の分析

学習者の評価コードと模範解答の編集距離を計測する．学習者 s の時刻 t における編集距離による進捗度 $progress_{edis}(s,t)$ の定義を次に示す．

$$progress_{edis}(s,t) = 100\frac{m-x_{s,t}}{m}$$

m：何も書いてない時と模範解答の編集距離
$x_{s,t}$：t 分の学習者 s の評価コードと模範解答の編集距離

3.2　実行結果の分析

別解の場合には編集距離による進捗度が 100 にはならない．しかし，解にはたどり着いているので，実行結果による分析により適切な進捗度を算出できる．実行結果による進捗度 $progress_{exec}(s,t)$ の定義を次に示す．

$$progress_{exec}(s,t) = \begin{cases} 30 & \text{(最初にコンパイルしたとき)} \\ 50 & \text{(最初にコンパイルが成功したとき)} \\ 50 + a\frac{50}{n} & \text{(実行結果通りに動いたとき)} \end{cases}$$

a は意図通り動いた実行例の数であり，n は実行例の数である．重みの付け方は課題が終了するまでの時間のかけ方の割合を基準としている．過去のデータから，解き始めてから半分ぐらいの時間で多くの学習者がコンパイル成功していた．

3.3　総合的な進捗度の分析

編集距離の分析による進捗度 $progress_{edis}(s,t)$ と実行結果の分析による進捗度 $progress_{exec}(s,t)$ の大きい方をその学習者の総合的な進捗度 $progress(s,t)$ とする．

4　実験

文献 [1] のデータを用い，次の実験を行う．

実験 1　編集距離による進捗度が 100 に近づくかを確認する．

実験 2　実行結果による進捗度が 100 に近づくかを確認する．

実験 3　総合的な進捗度により個別指導が必要な学習者が見つかるかを確認する．

問題は 5 問ある．実行結果が 1 つもあってないと実行結果による進捗度は最大 50 なので，実験 3 では進捗度が 50 以下の場合を個別指導が必要とする．

今回の実験では編集距離の重み付け (括弧の中の数値) を以下とする．

- for ⇔ while の置換 (0.5)
- if ⇔ for，while の置換 (2)
- for,while,if,else の挿入，削除 (2)
- 波括弧の挿入，削除 (0.5)

4.1　結果

実験 1:図 1 に学習者 B の編集距離による進捗度を示す．図の y 軸は進捗度であり，x 軸は演習経過時間 (分) を表す．青が問 1，赤が問 2，黄が問 3，緑が問 4，紫が問 5 である．

実験 2:図 2 に学習者 B の実行結果による進捗度を示す．

実験 3:図 3 に学習者 B の総合的な進捗度を示す．

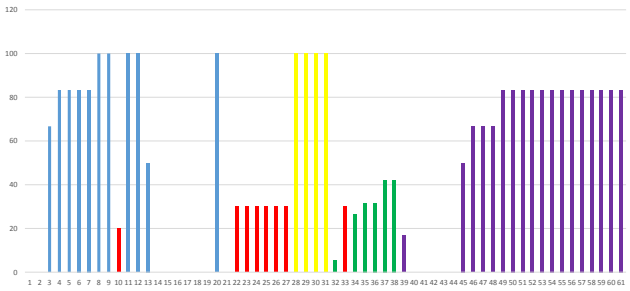

図 1　学習者 B の編集距離による進捗度

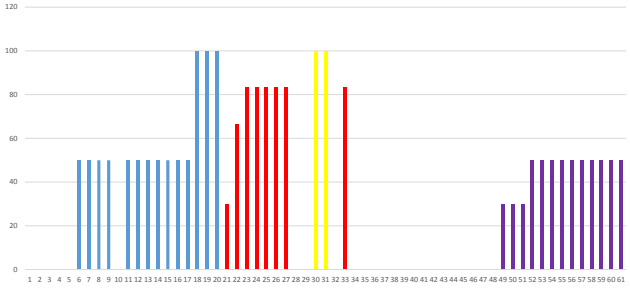

図 2　学習者 B の実行結果による進捗度

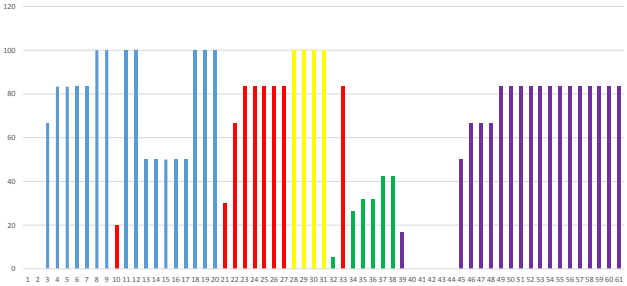

図 3　学習者 B の総合的な進捗度

5　考察

実験 1:解にたどり着いていても，別解のある問題だと編集距離が 100 にならない場合があるため，編集距離による進捗度は別解の場合には 100 に近づくとは言えない．編集距離による進捗度は初期段階から進捗が把握できる．

実験 2:別解の場合でも解にたどり着いていれば，すべての学習者の進捗度は 100 になっていたため，実行結果による進捗度は最終的に 100 に近づくと言える．しかし，最初にコンパイルをするまでは進捗度が 0 のままであるため，初期段階では進捗を把握することができない．

実験 3:編集距離による進捗度と実行結果による進捗度により，初期段階で進捗を把握することができ，別解の場合にも進捗度が 100 になる．また，進捗度が 50 以下の場合に，確かに解に全くたどり着けていない．これらから，総合的な進捗度により個別指導が必要な学習者が見つかると言える．しかし，制御構造がない場合に初期段階で進捗度が 100 になってしまう．編集距離による進捗度の最大値を低くすることにより適切に進捗度が求められると考える．

6　おわりに

本研究では，プログラミング演習における進捗度の観点から，学習者のコーディング状況を把握する方法を提案した．学習者の評価コードと模範解答の編集距離及び実行結果に着目し，手法の提案及び実験を行った．学習者の進捗度を確認することができた．今後の課題としては，指導者の意向に合わせた他の設問や学習者を対象とした実験，設問ごとに重みづけをするなどの検証が必要である．

参考文献

[1] 石元慎太郎，蜂巣吉成，吉田敦，桑原寛明，阿草清滋:プログラミング演習における構文要素の種類毎のビューによるコーディング状況把握方法の提案，情報処理学会，情報教育シンポジウム，8pages (2018).

[2] 井垣宏，井上亮文，齊藤俊，山田誠:プログラミング演習における学生のコーディング過程可視化システム C3PV の提案，情報処理学会論文誌，Vol.54，No1，pp.330-339 (2014).