6ZC-3

プログラミング学習における学習者の実力と問題の難易度の動的な推定方式

石川 斉 † 寺田 実 † 丸山 一貴 ‡
Hitoshi Ishikawa Minoru Terada Kazutaka Maruyama

1 はじめに

オンラインジャッジの拡張として, 問題も投稿で集める, プログラミング学習サイトを考える. この場合, 学習者の実力に対しての適切なフィードバックと, 次に取り組むべき問題についての指針を与えられることが望ましい.

本研究では, 先行研究の計算式を改良することで, プログラミング学習サイト向けの学習者の実力と問題の難易度を推定する方法を考案した. これをシミュレーション実験によって有効性を検証する.

2 先行研究

菅沼らの研究 [1] はオンライン授業で使われる Web データを用いて, 学習者（文献では学生）に問題が含まれている文章を提示し, 学習者は文章中の問題を解くことで, 学習者の実力（文献では理解度）および解いた問題の難易度を再評価し, 次に出題する問題を自動生成する研究である.

2.1 学習者の実力評価

時刻 t における, 学習者 i の実力 $s_{i,t}$ は (1) 式で計算される.

$$s_{i,t} = \begin{cases} s_{i,t-1} + \dfrac{\sum_{j \in Q}(q_{j,t} - s_{i,t-1})\delta_{i,j}}{\sum_{j \in Q}\delta_{i,j}} \\ \qquad (if \sum_{j \in Q}\delta_{i,j} \neq 0) \\ s_{i,t-1} \qquad (otherwise) \end{cases} \tag{1}$$

ここで, Q は学習者 i が解答した問題の集合（最近の 30 回の解答）を表し, $q_{j,t}$ は時刻 t における問題 j の難易度を表す. $\delta_{i,j}$ は, 実力よりも高い難易度の問題に正解した場合か, 実力よりも低い難易度の問題に間違えた場合に 1 となり, それ以外では 0 となる.

2.2 問題の難易度評価

問題 j の難易度 $q_{j,t}$ は (2) 式で計算される.

$$q_{j,t} = \begin{cases} q_{j,t-1} + \dfrac{\sum_{i \in S}(s_{i,\tau} - q_{j,t-1})\xi_{i,j}}{\sum_{i \in S}\xi_{i,j}} \\ \qquad (if \sum_{i \in S}\xi_{i,j} \neq 0) \\ q_{j,t-1} \qquad (otherwise) \end{cases} \tag{2}$$

ここで, S は時刻 *t-1* から t の間に問題 j を解答した学習者の集合を表し, $s_{i,\tau}$ は時刻 $\tau (t-1 \leq \tau \leq t)$ における学習者の実力を表す. $\xi_{i,j}$ は, $\delta_{i,j}$ と同じ条件である.

3 提案

先行研究における, 実力と難易度の動的評価はお互いに影響を与える関係にある. そこで, 実力の計算式のみの改良でも推定全体に変化が起きると考えた. まず, 実行結果の略式名称および, その内容を以下に示す.

- CE : コンパイルエラー
- RE : 実行時エラー
- NC : $ct = 0$
- CA : $1 \leq ct < at$
- AC : $ct =$ at

ct は学習者が正解したテストデータの数, at はその問題の全テストデータの数を表す.

また, 複数の方式を実験したが, 本稿ではそのうちの DC 法について述べる.

3.1 DC 法

DC(Difficulty Controll) 法は, 実行結果に応じて, (1) 式の計算に適応する難易度 ($q_{j,t}$) を変化させる. 難しい問題（実力 < 難易度）だった場合, 実行結果の AC と CA を正解とし, それ以外を不正解とした. これにより実行結果が CA の場合, 正解したテストデータの数に応じて部分点を与えることができる. また, 簡単な問題（実力 ≥ 難易度）だった場合, 実行結果の AC を正解とし, それ以外を不正解とした.

4 シミュレーション実験

4.1 概要

DC 法の有効性を検証するため, 先行研究の計算式, DC 法の 2 つを同一の問題と学習者の集合に対して実行し, 実力と難易度の推移を比較することで, 有効性を検証する. 実験の内容は以下のものを行った.

- 実験 1：異なる学習者モデルによる影響
- 実験 2：真の実力が途中で変化した場合の影響

4.2 学習者のモデル

学習者のモデルを以下のように仮定した. 学習者のモデルとして真の実力（s^{TRUE}）と真の難易度（q^{TRUE}）から実行結果を決める. まず, 以下の (3) 式で正解率を計算する. 正解率の上限と下限はそれぞれ 100 と 0 である.

$$\text{正解率 (\%)} = 50 - 20 * (s^{TRUE} - q^{TRUE}) \tag{3}$$

その後, 0〜100 の範囲でランダムな整数（rand_x）を生成する. 最後に正解率と rand_x を使って図 1 のグラフ上で決

Dynamic estimation of the ability of students and the difficulty of problems in programming excercises

† 電気通信大学 The University of Electro-Communications
‡ 明星大学 Meisei University

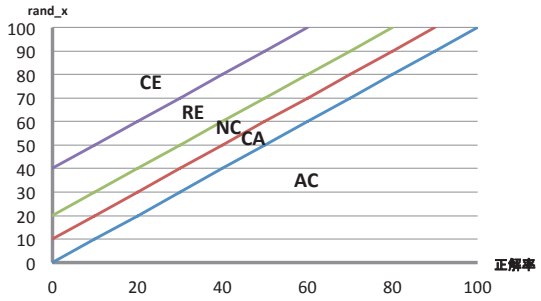

図 1　正解率と rand_x を用いた実行結果の決定グラフ

まる座標が実行結果となる．

4.3　実験方法

学習者の数を 100 人, 問題の数を 100 問とし, それぞれには真の実力と実力, および真の難易度と難易度を与えた．真の実力および真の難易度は両方とも変動しないものである．学習者は一人ずつ順番にランダムに選ばれた問題に挑戦し, その結果に応じて実力が再計算される．全学習者が一回ずつ終わった段階を 1 セットとする．これを 1,000 セット繰り返し, 実力の推移を記録した．難易度に関しては, 100 セット毎に難易度の再計算を行い, それを記録した．

4.4　実験結果

4.4.1　実験 1

実験 1 では全く同じ実験条件のもと, 図 1 の結果領域を変化させたとき, 実力と難易度の推移にどのような変化が起こるのかを検証した．まず, 図 1 の状態をモデル 1 とした．モデル 1 とは結果領域が異なるモデル 2, モデル 3 を図 2 に示す．

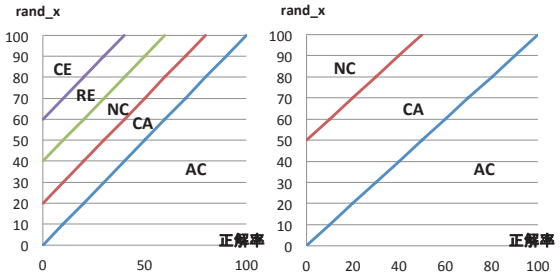

図 2　左がモデル 2, 右がモデル 3

実力の推移は図 3 になった．モデルの番号が上がるにつれて, 高い実力になっている．つまり, DC 法では学習者の特性を捉えた実力の推定を行えていることが分かった．また, モデルを変更したことによる難易度の推移は図 4 になった．モデルを変更しても, 難易度の推定には大きな影響を与えていないことが分かった．

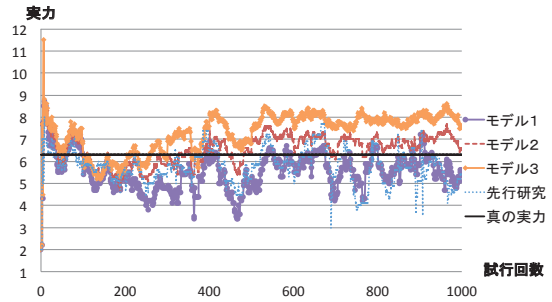

図 3　モデル毎における実力の推移

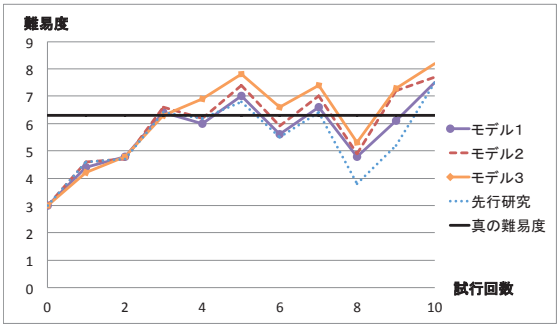

図 4　モデル毎における難易度の推移

4.4.2　実験 2

実験 2 では, 500 セット終わった段階で全学習者の真の実力を 3 上げた場合, 実力および難易度の推移にどのような変化がおこるのかを検証した．学習者のモデルは最初に仮定したモデル 1 である．学習者の理解度推移は図 5 のようになった．グラフから分かるように, 実力の変化に対応し

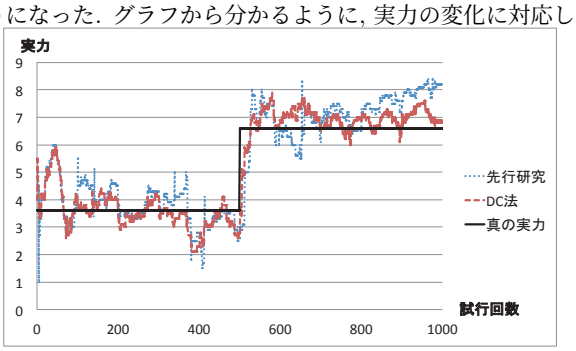

図 5　真の実力が変化した場合の実力の推移

て推移していることが分かった．また, ほとんどの問題の難易度は, 真の値に近付く推移となっていた．

しかし, 真の難易度が 3 の問題では, うまく推定されていなかった．原因としては, 学習者の実力分布が変化したためであると考えられる．本実験では, 真の難易度 3 を境にして前後 2.5 の範囲の真の実力を持つ学習者が均等な数で存在している．そこで, 学習者の真の実力を 3 上げた場合, 3 よりも実力が低い人がいなくなる．そのため, 均衡が崩れうまく推定できなかった．

5　まとめ

本研究では, シミュレーション実験を用いて, 改良した計算式の有効性を検証した．その結果, 学習者の実行結果を段階分けにすることで, 学習者の特性を捉えた推定を行う事ができるようになった．また, 実験によりこの推定方式では, 学習者の実力が上がっていく事で対応できない状態が存在することがわかった．そのため, 今後はこういった状況でも正しく推定できるように改良を加えることが必要である．

参考文献

[1] 菅沼明, 峯恒憲, 正代隆義．“学生の理解度と問題の難易度を動的に評価する練習問題自動生成システム “．情報処理学会論文誌, Vol.46, No.7, 2005-7.