プログラミング演習における教え合い促進のための学習者グループの動的形成手法の提案

A Proposal of Methods for Dynamically Forming Learner Groups to Promote Mutual Teaching in Programming Exercises

田部井　晃汰[*1]，中山　祐貴[*1]，大沼　亮[*1]，神長　裕明[*1]，宮寺　庸造[*2]，中村　勝一[*1]
Kota TABEI[*1], Hiroki NAKAYAMA[*1], Ryo ONUMA[*1], Hiroaki KAMINAGA[*1]
Youzou MIYADERA[*2], Shoichi NAKAMURA[*1]
[*1] 福島大学　共生システム理工学類／共生システム理工学研究科
[*1]Faculty of Symbiotic Systems Science, Fukushima University
[*2] 東京学芸大学　教育学部
[*2]Faculty of Education, Tokyo Gakugei University
Email: tabei@cs.sss.fukushima-u.ac.jp, {hnakayama, onuma, kami}@sss.fukushima-u.ac.jp, miyadera@u-gakugei.ac.jp, nakamura@sss.fukushima-u.ac.jp

あらまし：プログラミング教育の実施機会が広がり，スキル育成ニーズが高まっている．プログラミング演習では，学習者が躓く場面が多く，自力で解決することが難しいことが少なくない．そのため，学習者の状況に応じてこまめな指導を行うことが望ましいが，多数の学習者に対して少人数の教授者で対応する必要があるため限界がある．学習者ペアまたはグループによる教え合いが有望な対処方法の一つであるが，既存手法は，殆どが演習前の情報に基づく静的なグループ編成にとどまっている．本研究では，教え合い促進のための学習者グループの動的形成手法の開発を目指す．本稿では，躓きの有無やソースコードの類似性などを考慮した学習者グループの動的形成の概要を示す．
キーワード：グループプログラミング，ペアプログラミング，ソースコード分析，プログラミング教育

1.　はじめに

近年，プログラミング教育の実施機会が広がり支援ニーズが高まっている．プログラミング教育では，演習に取り組むことで学んでいく．しかし，躓く場面が多く，自力で解決することが難しい場面が少なくない．そのため，学生の状況に応じてこまめな指導を行うことが望ましいが，多くの学習者に対し，少数の教授者で対応するため限界がある．

これに対して，プログラミング演習支援に関する研究が多数報告されている．井垣らは，プログラミング演習における進捗状況把握のためのシステム[1]を提案している．コーディング過程の可視化を試みている点は興味深いが，躓きの把握後は従来通り教員による指導を前提としている．

人的資源の制約に対する有望な対処方法として，学生がソースコードを見せ合い，相互に指摘・教え合いを行う「ペアプログラミング」や「モブプログラミング」がある．ペアプログラミングにおけるペア編成の最適化を考慮した実践[2]が報告されているが，これらの既存手法は過去のペア編成や座席配置に基づく静的編成にとどまっている．つまり，演習中の最新状況を考慮した学習者ペアの編成については，十分に有効な方法が実現されていない．

本研究では，最新の状況（取り組んでいる課題，躓きの有無，ソースコードの類似性など）を考慮した形で学習者グループを動的に形成する手法を開発する．これにより，プログラミング演習における教え合い促進のための新たな支援の可能性を探る．

2.　問題点と支援方針

2.1 問題点

本研究では以下の問題点に焦点を当てる．

（問題点 1）各学習者の躓きを把握することが難しい

（問題点 2）各学習者のソースコードを把握・対比するには手間を要する

（問題点 3）最新の状況に応じて学習者グループを構成すべきであるが，十分に行うことが難しい

2.3 方針

本研究ではまず，正解コードに対する類似度の分析に基づく躓き推定手法を開発する（問題点 1 への対応）．次に，類似度と躓き，学習者間のソースコード比較に基づいたグループ生成手法の開発を行う（問題点 2，3 への対応）．

その上で，これらの手法に基づいて動的なグルーピングシステムを開発することで，問題点全体の解決を目指す．支援の流れを図 1 に示す．

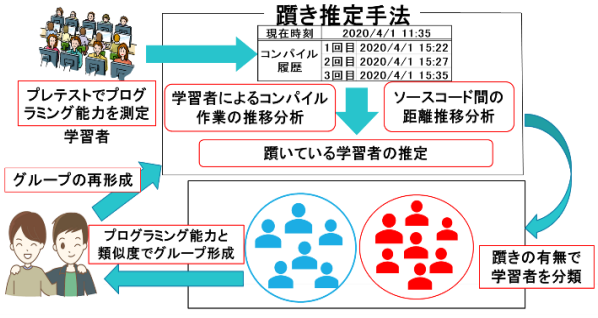

図 1　支援の概要

3.　ソースコード間の類似度分析に基づいた躓き推定手法

まず，学習者によるコンパイル作業の推移分析を行う．コンパイル履歴をもとに該当する学習者，コンパイル 日時, ソースコードを一時候補として抽出する． その後，ソースコード間の距離推移分析を行う．最初に抽出したものについて，該当ソースコードと正解 （解答例）との距離（類似度）が近づいていないケース を抽出する．

本研究では, 類似度を算出する際に, OLD, TED の 2 つの算出手法の中から，その時の状況に応じて，一番適したものを自動的に採用する．

4.　学習者グループの動的形成手法

4.1　演習設定

本研究では，プログラミングの基礎を終えた学習者を対象とする．学習者のプログラミング能力は，演習を行う前にプレテストを行い，そのテストの点数で能力を測定する．プログラミング能力は 5 段階で評価する．1 回の演習授業の中で 1 つの演習に取り組む．プログラミング演習は，最初はグループを作らず各個人で作業を進めていく．

4.2　学習者の分類

演習が始まって半分経過時点での各学習者の正解コードに対する類似度の算出と行き詰まりの有無を判定する．行き詰まりは半分経過までの間にロジック構成面の行き詰まりが発生したか，発生した場合は半分経過までに解決しているか否かで判定する．学習者の分類は行き詰まりの有無で 2 つのグループに分類する．

4.3　正解コードに対する類似度の学習者間比較

まず，躓きのあるグループの学生を 1 人抽出して躓きのないグループの学生の正解ソースコードに対する類似度を比較する．自分よりも正解ソースコードに対する類似度が低い場合や，同程度の場合は，現在躓いている問題の解決につながらない可能性が高い. そこで, 正解ソースコードに対する類似度が，躓きのある学習者よりも高い人を抽出する．例を図 2 に示す．

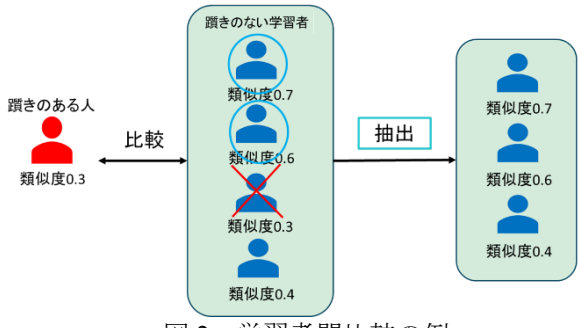

図 2　学習者間比較の例

4.4　各学習者のソースコード間の類似度の分析

次に，躓きのある学習者と 4.3 で抽出した学習者のソースコードを比較して学習者間の類似度を算出する．授業という限られた時間で行う場合，類似度が小さすぎると問題解決に対するアプローチが異なり，お互いの考えを理解できず，問題解決に対する進捗度が変わらない可能性が高い．そのため，ソースコードの類似度が高い学習者を抽出することで，授業内で問題が解決できるようにする．

4.5　プログラミング能力を考慮したグループ形成

鈴木ら[3]は自身とプログラミングのスキルが近い学習者との協調的な学習活動が学習内容の理解を深めるうえで重要だと述べている．そこで，本研究では， 4.4 で抽出した学習者の中から，プログラミング能力の差が小さい学習者とグループを形成する．4.3～4.5 の作業を躓きのあるグループの学生全員分行うことで動的なグループを形成する．グループ形成の流れを図 3 に示す．

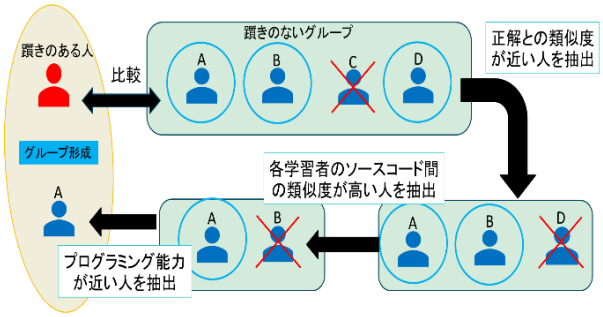

図 3　グループ形成の流れ

4.6　グループの再形成

グループ形成後，一定時間経過したら，躓いていないグループはそのまま残しておき，躓きが解消できていないグループはすべて解体する．その後，再び学習者の分類を行い，正解ソースコードに対する類似度の学習者間を比較し，プログラミング能力を考慮したグループ形成を行う．

5.　おわりに

本稿では，演習中の最新状況に応じた学習者グループの動的形成について述べた．具体的には，躓きの有無やプログラミング能力，正解コードに対するソースコードの類似性，および，各学習者のソースコード間の類似度などを考慮した学習者グループ形成について概説した．

今後は，実際のプログラミング演習のデータに提案手法を適用した実験を重ね，有効性検証と改善を進めたい．

参考文献

(1) 井垣宏，齊藤俊，井上亮文，中村亮太，楠本真二，“プログラミング演習における進捗状況把握のため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, No.1, pp.1-10 (2013).
(2) 鈴木聡,“ペア編成・座席配置の最適化を利用した複数回のペアプログラミング実習の実践”,情報処理学会研究報告, Vol.2014-CE-127, No7, pp1-8 (2014).
(3) 鈴木聡，廣川佐千男，“ペアプログラミングと反転授業を導入したコンピュータシミュレーション実習における履修者の学習活動の分析”，日本教育工学会論文誌，Vol.41,No.3，pp.255-269，2017.