K-022

提供関数の段階的機能縮小化における関数呼出し履歴の分析
Analyzing Function Call Logs During Stepwise Increment Of Assignment Difficulty

西本 航 † **高橋 渉** ‡ **原田 史子** † **島川 博光** †
Wataru Nishimoto Wataru Takahashi Fumiko Harada Hiromitsu Shimakawa

1. はじめに

情報系の大学ではプログラミングの課題を演習形式で出題する授業があり，学習者は演習課題を実装することにより，プログラミングの知識を習得している．一般的なプログラミングの演習授業では，少数の指導者が多数の学習者に対して指導をしている．そのため，授業時間内に全学習者に対して十分な指導をするには，効率よく指導する必要がある．しかし，現状では学習者の行き詰まっている箇所を指導者は課題のソースコードや実行結果から特定しているため効率よく指導できていない．

本論文では段階的に課題を出題したうえで，学習者が関数を呼び出した順序と引数を，正しい順序や引数と比較することによって行き詰まっている箇所を特定する手法を提案する．本手法により学習者が行き詰まっている箇所を効率よく特定できる．

2. プログラミング演習授業の現状と問題点

2.1 行き詰まり状況特定の必要性

プログラミングでは数学の公式，回文やゲームのルールといったプログラミング能力とは関係ない知識を使うことがあり，そのような課題がプログラミング演習の授業においても出題されている．しかし，これらを利用した課題を実装できない学習者が多い．

本論文ではプログラミング能力とは関係のない知識を一般知識と呼び，プログラミング言語の文法やプログラムの構成法に関する知識であるプログラミング知識とは区別する．一般知識を用いた課題を実装できない理由として，学習者が一般知識を正しく理解できていない，もしくは学習者のプログラミング知識が不足していることが考えられる．指導者はこれらを考慮した上で適切に指導する必要がある．

現在，指導者は学習者の行き詰まっている箇所をプログラムの実行結果およびソースコードから特定している．学習者が多く指導者が少ない現状では効率よく行き詰まっている箇所を特定する手法が必要となる．

2.2 既存研究

行き詰まり箇所を特定するための研究として，コンパイラのエラーメッセージや正解プログラムとの比較によって間違い箇所を特定する研究 [1] [2] や，小テストによって学習者の理解状況を特定する研究 [3] がある．しかし，これらの手法はエラーメッセージを使う場合や文法的な間違いを見つける場合は有効であるが，これらで論理的な間違い箇所を特定することは難しい．また，小テストを受けた時点の学習者の理解状況と，現在の学習者の理解状況は一致しない可能性がある．全学習者に適切な指導をするためには，現在の理解状況および行き詰まり箇所を特定する必要がある．

†立命館大学情報理工学部
‡立命館大学大学院情報理工学研究科

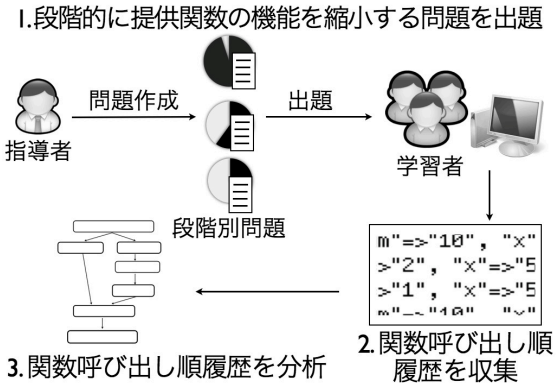

図 1: 手法の概要図

3. 段階的な課題の出題とその分析

3.1 手法の概要

本論文では現在の理解状況，および行き詰まり箇所を特定する手法を提案する．本手法では学習者の行き詰まり箇所を特定するために課題をいくつかの段階に分割し，出題する．課題を段階に分割することによって，学習者にとって既知であると仮定される知識を設定することができ，学習者の理解状況を絞り込むことができる．段階ごとに問題のプログラムを実装するために必要な機能を含む関数を学習者に提供する．これら関数を提供関数と呼ぶ．提供関数が呼び出された順序と引数を記録するため，学習者には提供関数の使用を義務付ける．提供関数が呼び出された順序と引数を並べたものを関数呼び出し順履歴と呼ぶ．サーバは関数呼び出し順履歴を記録する．

提供関数の機能を調整することにより課題の分割における初めの段階では，プログラミング知識を多く必要とせず，一般知識を多く用いる問題を出題し，段階が進むにつれプログラミング知識が多く必要となり，一般知識をあまり用いないような問題を出題していく．段階別に一般知識を多く必要とする問題やプログラミング知識を多く必要とする問題を設定することで，プログラミング知識と一般知識の理解度を切り分けて評価できるため，行き詰まり箇所がいずれの原因によるものかを特定しやすくなる．さらに課題ごとの関数呼び出し順履歴における関数呼び出し順とその引数の評価によって行き詰まり箇所を詳細に特定できる．

3.2 課題の分割

課題は課題導入段階，課題把握段階，基本実装段階の 3 段階に分割し，段階ごとに提供関数を用意する．課題導入段階は課題で実装すべきプログラムの仕様を学習者が理解するための段階である．また，指導者が既知であると設定している一般知識が学習者にとっても既知であるか判定するため，プログラミングとしては簡単な問題とする．

課題把握段階では実装すべきプログラムにおける処理を複数の関数に分割したものを提供関数として提供し，

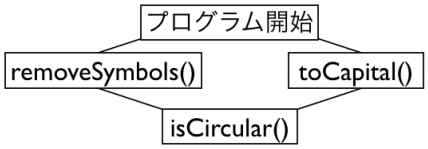

図 2: 関数呼出し順ルール

表 1: 関数呼び出し順履歴 1

関数呼び出し順	引数
toCapital()	”ab,Cb!!A”
isCircular()	”AB,CB!!A”

それらを用いて実装する．また，数値計算などの処理は提供関数として提供する．この段階では提供関数を正しい順序と正しい引数で呼び出すことが求められる．

基本実装段階は課題把握段階の提供関数の中で，もっとも重要な関数の仕様を理解し実装する段階である．それ以外の関数は引続き提供関数として提供する．この段階では関数を実装する必要があるため，プログラミング知識が重要になる．

3.3 行き詰まり箇所の特定

入力された文字列を回文判定するプログラムを作成する課題を考える．この問題は回文判定課題における課題把握段階の問題であり，学習者が回文判定をする際の処理の流れを理解することを目的としている．また，提供関数として文字列を大文字に統一する関数 toCapial()，文字列から記号を排除する関数 removeSymbols()，文字列の正順と逆順を比較する関数 isCircular() を用意する．

提供関数が呼ばれるべき正しい順序は一意ではないため，段階ごとに提供関数の呼ばれるべき正しい順を規定したルールを制定する必要がある．このルールを関数呼び出し順ルールと呼ぶ．本問題における関数呼び出し順ルールを図 2 に示す．関数呼び出し順ルールにおける各分岐点の分岐先ノードは順不同ですべて呼び出す必要があるものとする．したがって，図 2 により導出される正解関数呼び出し順は，removeSymbols() → toCapital() → isCircular() と toCapital() → removeSymbols() → isCircular() のふたつがある．

ひとつ目の例として表 1 のような関数呼び出し順履歴が収集された場合，正解関数呼び出し順と表 1 の関数の呼び出し順を比較すると removeSymbols() を呼び出していないことがわかる．この問題は課題把握段階であるため，学習者は回文判定の流れを理解する必要がある．この間違い箇所から文字列から記号を排除するという回文判定の流れを学習者が理解していないと特定できる．

ふたつ目は引数評価をする必要のある例である．表 2 の左 2 カラムが関数呼び出し順履歴であり，右 1 カラムは各関数に与えるべき正解引数である．図 2 により導出された正解関数呼び出し順と表 2 の関数呼び出し順を

表 2: 関数呼び出し順履歴 2 と正解引数例

関数呼び出し順	引数	正解引数例
toCapital()	”ab,Cb!!A”	”ab,Cb!!A”
removeSymbols()	”ab,Cb!!A”	”AB,CB!!A”
isCircular()	”ab,Cb!!A”	”ABCBA”

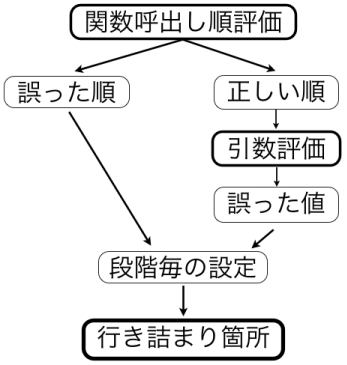

図 3: 行き詰まり箇所特定の流れ

比較すると，順序は問題ないことがわかる．また，表 2 の引数と正解引数例を比較すると，removeSymbols() と isCircular() の引数が間違っていることがわかる．学習者は回文判定の流れは理解しているが，関数に引数を与え，戻り値を次の処理に利用するというプログラミングにおける処理を理解していないと特定できる．以上のように本手法を使うことで学習者の行き詰まり箇所を特定できる．

3.4 関数呼び出し順履歴の分析

図 3 の流れに沿い，関数呼出し順評価，引数評価の 2 つの評価を用いて間違い箇所を特定し，関数呼び出し順ルールや学習者が理解すべき項目といった課題ごとの設定を組合わせることで行き詰まり箇所を特定する．関数呼び出し順評価とは関数呼出し順ルールに規定された提供関数の呼ばれるべき順序と関数呼び出し順履歴の関数呼び出し順を比較する評価法である．

引数評価とは関数呼び出し順履歴における各関数の引数が正しい引数であるかを判断する評価法である．関数呼び出し順評価では関数呼出し順が正しい場合，引数評価を行なう．引数評価において間違い箇所が発見された場合，その関数と引数および段階ごとの設定と組合わせることで行き詰まり箇所を特定する．

4. おわりに

本論文では学習者の課題に対する理解状況，および行き詰まり箇所を関数呼び出し順履歴を用いて特定する手法を提案した．今後は本手法の有効性を検証，および本手法を適用できる課題を増やしていく予定である．

参考文献

[1] 西 輝之，劉 渤江，横田 一正，デバッガとの連携による C 言語学習支援システムの提案，電子情報通信学会技術研究報告. ET, 教育工学，Vol.106， No.583，pp.173-178， 2007

[2] 野口 貴弘，竹内 章，学習者 Prolog プログラムのデータフローを利用した診断と採点，日本教育工学会論文誌，Vol.34， Nu.3， pp.249-257， 2010

[3] 新開 純子，早勢 欣和，宮地 功，Moodle を基盤としたプログラミング教育のための穴埋め問題生成に関する検討，電子情報通信学会技術研究報告. ET, 教育工学，Vol.108， No.247， pp.5-10， 2008