プログラミング演習におけるアドバイス提示支援システムの提案
― 制御構造に着目したアドバイス対象ソースコードの検索 ―

2019SE004 藤谷 梨々香　2019SE005 福吉 真由　2019SE006 舟橋 杏
指導教員：蜂巣吉成

1　はじめに

大学で実施されるプログラミング演習では，教員が問題を出題し，学習者が各自演習に取り組む．教員の指導形式は，各学習者に解説をする個別指導と学習者全体に解説をする全体指導がある．個別指導において，複数の学習者の同じ質問に対して教員や TA が何度も同一なアドバイスすることがある．この場合，教員は全体指導をすることがあるが，該当の問題に到達しておらず，指導内容を聞いていない学習者も存在する．また，オンライン演習や時間外学習では全体指導をすることが困難である．

同一な質問に対して教員が何度も同じアドバイスする問題を解決するために，演習時間内に教員がある学習者にアドバイスすると，同様なソースコードの学習者を検索して，アドバイスを提示するシステムを提案する．本研究では，同一なアドバイスに該当するソースコードの検索手法を提案する．技術的課題は，教員のアドバイスの意図を形式的に表すことができず，同様な学習者のソースコードの基準が不明確なことである．アプローチ方法として，学習者のソースコードに対するアドバイス内容を調査し，教員がアドバイスに至った根拠となるアドバイス理由の分類を提案する．本研究では，コンパイル前の編集途中のソースコードも扱うこととし，処理の流れに大きく関わる制御構造に着目する．各アドバイスに該当する学習者のソースコードと模範解答を比較し，基準を明確にする．また，アドバイスとソースコードの対応箇所は一意に決まらないので，アドバイス理由の分類とシステムを結びつけるための検索条件を提案し，教員に選択させる．本研究は C 言語の初学者向け演習を対象とし，模範解答ソースコードを教員が用意し，学習者は WebIDE などでソースコードを作成し，編集中のソースコードが随時取得できるものとする．

以下，教員がはじめにアドバイスしたい学習者のソースコードを選択ソースコード，検索の対象になるソースコードを検索対象ソースコード，検索条件によって抽出された学習者のソースコードを抽出ソースコード，教員がアドバイスしたいソースコードを対象ソースコードとする．

提案するシステムのイメージ図を図 1 に示す．

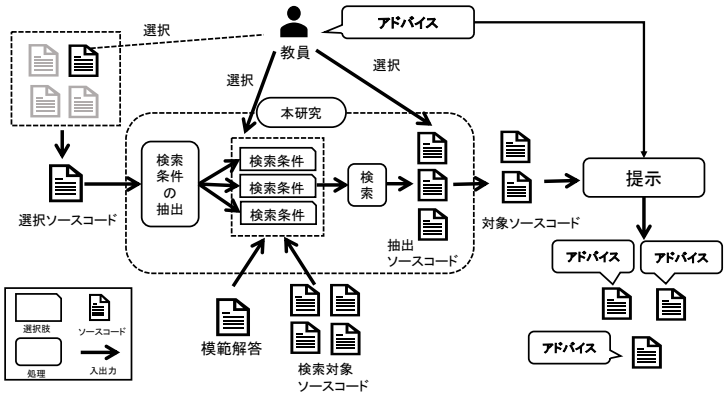

図 1　全体のイメージ図

2　関連研究

井垣らは，ソースコードの行数，課題ごとのコーディング時間，単位時間あたりのエディタ操作数，課題ごとのエラー継続時間を計測し，可視化処理を行う C3PV を用いて，進捗を把握している [1]．池富らは同値類分割や N-gram を特徴量としたクラスタリングを用いることで，教員が確認する必要があるソースコードの総数の削減を実現している [2]．蟹江らは，制御構造と条件式における同値関係をまとめることで進捗を把握している [3]．久保田らは模範解答への近付き具合を表す進捗度と単位時間あたりの文字の変化量を表す活動度を定義し，サポートが必要な学習者を特定している [4]．学習者の進捗状況を把握するシステムは提案されているが，状況を把握した上での指導に関する研究は見当たらなかった．

3　アドバイス理由と検索条件の分類

3.1　概要

技術的課題の解決のためにアドバイス理由とソースコードの関係を整理し，アドバイス理由を 3 段階で分類する．はじめに文・式レベルに関する分類し，更に問題の全体像に大きく関わる制御構造に着目し分類した．次に制御構造における分類を自動で判別するためにシステム基準の分類をした．最後に，システム基準の分類から自動化に向けた検索条件の分類をした．

3.2　文・式レベルに関するアドバイス理由の分類

本研究における演習問題は指定された関数を作る問題とする．関数の本体は文の並びで構成され，文は式で構成されている．文・式レベルに着目しアドバイス内容を次の 6 つに分類した．

A1. 文が不足している

A2. 余分な文がある

A3. 書くべき文の種類が異なる

A4. 文を書く場所が異なる

A5. 文の種類は正しいが，式が異なる

A6. 冗長な記述をしている

3.3 制御構造におけるアドバイス理由の分類

3.2 節の文・式レベルに関するアドバイス分類 A1〜A4 を問題の全体像の理解に関わる制御構造に着目して，次の 6 つに分類した．

B1. 制御構造が不足している

B2. 余分な制御構造がある

B3. 模範解答が含まれている上で，余分な制御構造がある

B4. 誤った制御構造を用いている

B5. 出現順が異なる

B6. 記述する階層が異なる

3.4 システム基準のアドバイス理由の分類

本研究では，蟹江ら [3] の研究を基に，ソースコードから選択構造の if,else,else if，反復構造の for,while を抽出し，順次構造と入れ子関係を表したものを抽象制御構造と定義する．3.3 節の分類 B を自動で判別するために抽象制御構造を基に次の 7 つに分類した．

C1. 抽象制御構造が不足している

C2. 余分な抽象制御構造がある

　　C2-1. 模範解答を含む

　　C2-2. 模範解答を含まない

C3. 入れ子構造は正しいが，予約語が異なる

C4. 予約語とその出現順は正しいが，入れ子構造が異なる

C5. 予約語と入れ子構造は正しいが，出現順が異なる

C6. 予約語は正しいが，出現順が異なる

3.5 検索条件の分類

同一のアドバイスとなるソースコードの検索条件を一意に決めるのは困難なので，複数の候補を教員に提示し，そのうちの 1 つ選択させる．検索条件を次の 5 つに分類し，分類 C と対応付けた．選択ソースコードと模範解答との差分で抽出された制御構造を特定の制御構造と呼ぶ．

1. 制御構造が同じソースコードを探す
2. 特定の制御構造が足りないソースコードを探す
3. 特定の制御構造が余分にあるソースコードを探す
4. 模範解答＋αがあるソースコードを探す
5. 模範解答を含むかつ特定の制御構造が余分なソースコードを探す

3.3 節の分類 B と 3.4 節の分類 C と検索条件の対応付けを図 2 に示す．ここで，3.4 節の分類 C から検索条件の優先度が低いものを点線で示す．

4 ソースコード検索システム

1 章で示したアドバイスシステムのうち，本研究ではソースコード検索システムの部分を扱う．

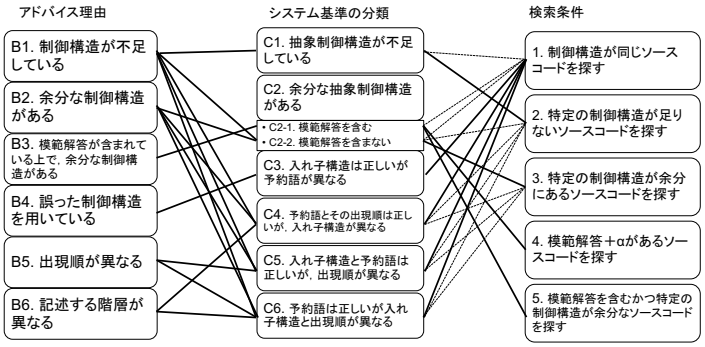

図 2 アドバイス理由の分類 B と分類 C と検索条件の関係

4.1 前提条件

提案するシステムは C 言語の初学者向け演習を対象とする．C 言語で記述されたソースコードのうち，制御構造の入れ子の階層は 3 階層目までを対象とし，アドバイスは制御構造に関わるものとする．

4.2 ソースコードの分類・検索条件の提示

システムは，教員が選択した選択ソースコードと模範解答の差分を基に分類を行い，図 2 の対応に従って検索条件を提示する．模範解答がソースコード 1，選択ソースコードがソースコード 2 の場合を例に挙げる．この問題は 5 章問 4 の「文字列 s から文字 c を削除する関数」を記述する問題である．ソースコード 2 へのアドバイスは「文字を詰める処理は不要」であり，分類 B3 に該当する．

ソースコード 1 模範解答

```
1  void strdel(char *s, char c)
2  {
3    char *t = s;
4    while (*s != '\0') {
5      if (*s != c) {
6        *t = *s;
7        t++;
8      }
9      s++;
10   }
11   *t = '\0';
12 }
```

ソースコード 2 選択ソースコード

```
1  void strdel(char *s, char c)
2  {
3   while (*s != EOF) {
4     if (*s == c) {
5       while (*s!=EOF) {
6         *s = (*s+1);
7       }
8     }
9   }
10 }
```

このとき，2 つのソースコードの差分から分類 C2-1 に分類される．その結果，検索条件として次の 3 つが抽出さ

れ，教員に提示される．なお，制御構造が 1 つ内側に入ることを@で示し，検索条件の予約語の階層数は@で表現するものとする．

1. 模範解答 + αがあるソースコードを探す
2. 模範解答を含むかつ@@while/for が余分なソースコードを探す
3. 制御構造が同じソースコードを探す

4.3 検索条件の選択・抽出ソースコードの提示

提示された検索条件に対し，教員はアドバイスに対応する検索条件を選択する．4.2 節のソースコード例に対し，文字を詰める処理に関わるのは 2 階層目の for 文であるため，2 つ目の検索条件を選択する．

教員の選択を受け，システムは検索対象ソースコードから選択された検索条件に合致したものを探す．検索後，抽出ソースコードを一覧にし出力する．

5 評価

本システムで得られる抽出ソースコードが期待されるもの，つまり選択ソースコードと同一なアドバイスをするソースコードか評価する．

5.1 ソースコードの収集

評価を行うために，ソフトウェア工学を専攻している大学 3 年生 12 人の協力のもとソースコードの収集を行った．1 時間 10 分で C 言語の演習問題 5 問を WebIDE を用いて 12 人に解答してもらい，1 分毎とコンパイル時のソースコードを保存する．演習問題には，二重ループなど複数の制御文を含む問題を 5 問用意した．問 1 は＊を階段式に表示する関数，問 2 は＊をダイヤモンドの形に表示する関数，問 3 は配列の最大値と 2 番目に大きい値を求める関数，問 4 は文字列から文字を削除する関数，問 5 はデジタルルートを求める関数の問題である．問 1 は 266 個，問 2 は 221 個，問 3 は 183 個，問 4 は 148 個，問 5 は 161 個のソースコードを収集することができた．また各問に要した解答時間の平均は，問 1 は 20.58 分，問 2 は 16.83 分，問 3 は 13.25 分，問 4 は 10.41 分，問 5 は 11.75 分であった．

5.2 選択ソースコードと対象ソースコード

評価で用いる選択ソースコードと対象ソースコードを用意する．収集したソースコードを保存時刻の 3 分毎に教員が確認し，制御構造に関わる指摘の必要性を感じたソースコードや全体より遅れている学習者のソースコードを選択ソースコードとして選んだ．各選択ソースコードへの 17 個のアドバイスとその分類 B，問題番号をまとめた表を表 1 に示す．

収集した全ソースコードに対して選択ソースコードを選んだときと同じ基準でアドバイスを考えた．選択ソースコードと同一のアドバイスをするソースコードを対象ソースコードとして用意した．

表 1 17 個のアドバイス

	問題番号	分類 B	アドバイス
1	問 1	B1	空白の繰り返しが足りない
2	問 1	B1	行の繰り返しが足りない
3	問 1	B1	空白の繰り返しの出力がない
4	問 2	B1	後半の出力を書く
5	問 2	B1	図形の上と下部分を分けて記述する
6	問 2	B1	図形の上と下部分を分けて記述する
7	問 2	B2	最後の空白の繰り返し出力は不要です
8	問 3	B2	繰り返しは 1 回でいい
9	問 3	B2	繰り返しは 1 回でいい
10	問 2	B2	if 文は不要です
11	問 2	B4	繰り返しは while より for が適切
12	問 4	B3	文字を詰める処理は必要ない
13	問 4	B1	NULL 文字まで繰り返す処理が必要です
14	問 4	B2	二重ループは不要です
15	問 3	B2	二重ループは不要です
16	問 5	B1	繰り返しの処理が必要です
17	問 5	B6	並列の繰り返しを入れ子にする必要がある

5.3 アドバイス意図と差分の対応の相違

選択ソースコードとその検索条件を確認したところ，表 1 の分類 B と各選択ソースコードの分類 C との対応に相違が生じた．分類 B では挿入だが分類 C では置換と判定される選択ソースコードがあった．これは，分類 C3「入れ子構造は正しいが予約語が異なる」を置換に該当する分類 B4「誤った制御構造を用いている」のみに対応づけていたためである．ソースコードが分類 C3 に該当してもアドバイス意図は置換ではない場合があったので，図 2 で示した対応関係の分類 B1，2 と分類 C3，分類 C3 と検索条件 2，3 の間の対応を追加し，検索条件「特定の制御構造が不足・余分なソースコードを探す」を提示することとした．

分類 B では単に削除だが，分類 C では模範解答を含むと判定されるものがあった．そこで分類 B2 と分類 C2-1 を対応づけ，模範解答を含むと判定された場合も「特定の制御構造が余分なソースコードを探す」を提示することで対応した．以下，修正後の対応関係を用いて評価を行った．

5.4 抽出ソースコードの評価

5.4.1 方法

再現率と適合率で抽出ソースコードを評価する．再現率では，システムが対象ソースコードを漏れなく抽出することができているかを評価できる．適合率では，教員の意図に合った対象ソースコードが抽出されているかを評価できる．対象ソースコードの数を p，抽出ソースコードの数を q，対象ソースコードかつ抽出ソースコードの数を r とする．再現率は r/p，適合率は r/q で求める．

手が止まっていた学習者は同じ状態のソースコードを複数持つため，対象・抽出ソースコードの数を重複して数えることになる．そこで収集したソースコードを抽象制御構造に変換後，各問各学習者のソースコードの中で同一の抽象制御構造のソースコードをまとめて 1 つと数える方法をとる．ただし，同一の抽象制御構造であっても条件式や本体を見ていないので同じ処理をする制御文とは限らない．

この脅威を知った上で，対象ソースコードと抽出ソースコードを同一の抽象制御構造を統合した後の個数で表す．

解き始めは考えている学習者が多いため，解き始めを検索対象ソースコードにすると適合率が低くなる傾向がある．そこで問 1〜3 は解き始め 5 分間のソースコードを，問 4〜5 は解き始め 3 分間のソースコードを除いて対象ソースコードと抽出ソースコードを集計する．中には，問題に取りかからず，1 分ほどアクセスしデータが収集されている学習者がいる．その場合，連続して取り掛かった時間の一番初めを解き始めとした．

5.4.2 分類 B に着目した再現率と適合率

分類 B の 6 つに着目しアドバイスによる再現率と適合率の傾向を調べる．ここでの再現率と適合率は，表 1 の分類 B ごとに該当する p，q，r それぞれの総和を用いて計算する．この計算方法では，総和の中には重複して数えている場合があるが，分母と分子どちらにも関わるので再現率や適合率に大きく影響しない．分類 B と再現率，適合率をまとめた表を表 2 に示す．

表 2 分類 B と再現率・適合率

分類 B	再現率	適合率
B1	0.895 (68/76)	0.324 (68/210)
B2	0.492 (31/63)	0.886 (31/35)
B3	0.667 (2/3)	1 (2/2)
B6	1 (2/2)	0.667 (2/3)

分類 B4 は，対象ソースコードと抽出ソースコードが共に存在しなかったため，再現率と適合率を求められなかった．表 1 より分類 B5 に該当する選択ソースコードへのアドバイスがなかったため，再現率と適合率の調査は行えなかった．

6 考察

6.1 得意不得意とするアドバイスや問題の特徴

5.4.2 節より，提案したシステムには得意・不得意とするアドバイスや問題があることが分かった．得意とするのは，特定の制御構造に直結したアドバイスや，問題自体の考え方が限られている場合である．一方で不得意とするのは，条件式に左右されるアドバイスや，同じ制御構造が使われる，記述する制御構造が多い，階層が多く必要である，別解が存在するといった特徴のある問題の場合である．これらは模範解答と学習者ソースコードの制御構造の対応が正しく取れていないことが原因である．

模範解答の制御構造が 1 つの場合，その制御文を書けば，どのような条件式でも「模範解答を含む」に該当する．条件式についての判定の追加と模範解答を含む判定の改善については今後の課題とする．

6.2 複数の誤りがある場合のアドバイス優先順位

実際の演習において，学習者のソースコードは複数の誤りがあることがある．本システムは 1 つのソースコードに対して 1 つのアドバイスをすることを前提としているので，3.4 節で挙げた分類 C のいずれか 1 つに分類される．例えば，1 つのソースコードに対して置換に関わるアドバイスと挿入に関するアドバイスが必要な場合，置換に該当する分類 C3 は入れ子構造が正しいことを前提としているので，挿入に該当する分類 C1 に分類され，検索条件も不足に関わるものが優先して提示される．

ソースコードに複数の誤りがある場合においては，アドバイスの優先順位として差分における挿入と削除が優先される．移動や置換は複数の制御構造が関わるため，挿入・削除の修正より後に指摘されることは妥当だと考える．

6.3 実用化に向けて

実際の演習で提案システムを利用する場合，学習者が必要とするタイミングでアドバイスを提示することが望まれる．編集が滞っている学習者はアドバイスを必要とし，記述中の学習者はすぐには必要としていない可能性が高い．井垣ら [1] や久保田ら [4] の進捗把握の研究を用いて，ソースコードの変化から学習者の状況を把握し，アドバイスを必要とするタイミングを判定することでより効果の高いアドバイスができると考える．

再現率と適合率の低下の原因である別解については，ソースコードがどの模範解答に近いのかを池富ら [2] や久保田ら [4] の研究を用いて判定し，判定された模範解答と選択ソースコードから検索条件を抽出することで対応する．

7 おわりに

本研究では，プログラミング演習時において，制御構造に着目したアドバイスに対して同一なアドバイスに該当する学習者の検索手法を提案した．条件式を見た判定や，実用化に向けた提示方法は今後の課題である．

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.330-339 (2013).

[2] 池富蓮，上田遼太：プログラミング進捗把握のための学習者ソースコードのクラスタリング手法とクラスタ提示方法の提案，2021 年度南山大学卒業論文 (2022).

[3] 蟹江茉衣香，松原知奈美，佐竹玲己衣：プログラミング演習における制御構造と条件式を用いた進捗状況把握方法の提案，2015 年度南山大学卒業論文 (2016).

[4] 久保田詩門，蜂巣吉成，吉田敦，桑原寛明：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，研究報告コンピュータと教育 (CE) (2019-CE-151)，pp.1-8 (2019).