K-036

ソースコードを段階的に発展させる課題における学習状況の顕在化
Exhibiting Learning Situation of Students During Stepwise Refinement of Source Codes

西本 航 § 小山 昂紘 ‡ 原田 史子 † 島川 博光 †
Wataru Nishimoto Takahiro Koyama Fumiko Harada Hiromitsu Shimakawa

1. はじめに

多くの大学では，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，学習者は課題の実装からプログラミングの知識を習得している．一般的に演習授業は少数の指導者が多数の学習者を指導する形態をとっている．そのため，効率よく学習者の状況を特定できなければ，すべての学習者に対して十分な指導を提供できない．しかし，指導者は学習者の状況をソースコード，実行結果，学習者による説明をもとに特定しているため，効率よく状況を特定できていない．

本論文では，学習者の状況を効率よく特定するために，プログラミング初学者が，学習目標との対応が明確，かつ，模範解答と類似した，綺麗なソースコードを記述できるようにする手法を提案する．

2. プログラミング教育の現状と問題点

2.1 学習者の状況特定の必要性

プログラミングの演習授業では学習者にとって新しい知識を使った課題に取り組むため，学習者は解答に行き詰まることがある．少人数である指導者は学習者の状況を迅速に特定し，適切な指導を与える必要がある．学習者はプログラミング学習途中であるため，難読なソースコードを書き，また，自身の状況を正しく理解していない場合が多くある．したがって，指導者は不正確かつ理解が困難な情報をもとに，学習者の状況を特定している．

迅速に学習者の状況を特定するには，他にも情報源が必要である．課題からより多くの情報を得られるならば，より効率よく学習者の状況を特定できると考えられる．既存の課題は各課題で必要となる知識が多くあるため，どの知識を学習者が理解できていないのかという情報を取得できない．学習者の状況を迅速に特定するには，各課題で必要となる知識を減らし，学習目標の到達状況のような的確な情報を得られるようにする必要がある．

2.2 最近接発達領域とスキナーの指導法

心理学者ヴィゴツキーが最近接発達領域にある課題を解かせることが効率良く学習者を成長させるという理論を提唱した [1]．最近接発達領域の範囲にある課題とは，学習者が一人では解けないが，少しの補助があれば解ける課題のことである．最近接発達領域は学習者ごとに異なるため，課題は学習者ごとに用意する必要がある．しかし，指導者が少数であるため，学習者ごとに課題を用意することは困難である．

心理学者スキナーが小問に分割された課題を学習者各自のペースで，多くのフィードバック受けとりながら，課題を進めさせるという指導法を提唱した [1]．学習者が行き詰まることのないように課題を分割するスキナーの指導法を使うことで，最近接発達領域にある課題を出題できると考えられる．

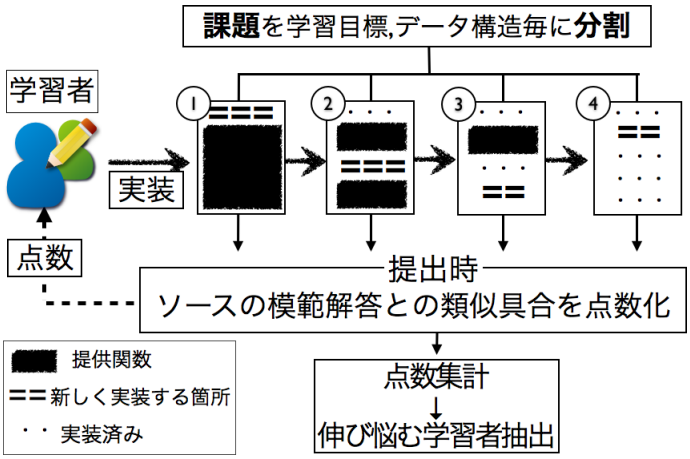

図 1: 手法の全体図

2.3 既存研究

指導者の負荷を軽減させる方法として自動採点やエラーメッセージの解析によって間違い箇所を特定する方法がある [2][3]．また，学習者によりよい学習環境を提供する研究として，コンテスト形式での演習など専用ツールの利用，小テストの実施が提案されている [2][4]．これらの手法による採点は主に学習者が実装したプログラムの出力結果と模範解答の出力結果の比較によるものである．出力結果を用いた自動採点では，学習目標に到達しているかは評価できない．また，出力結果を用いた自動採点やエラーメッセージからでは，学習者が学習目標に則って課題を進めているか判断できない．したがって，学習者が書いたソースコードが学習目標に到達しているかを評価できる手法が必要である．

そのためには，課題の学習目標と課題との対応が明確になっていることが重要であると考えられる．また，それらの課題を解くことで学習者はプログラミングの基礎を学習でき，綺麗なソースコードを書けるようになる．

3. 段階的に発展する課題

3.1 スキナーの指導法をもとにした演習形式

本論文では，プログラミング初学者が，学習目標との対応が明確で，かつ，模範解答と類似した，綺麗なソースコードを記述できるようにする手法を提案する．図 1 に手法の全体像を示す．スキナーの指導法の一部を適用して演習課題を出題する．演習課題を学習者には少し難しい，いくつかの小問に分割する．小問への分割は，学習目標やデータ構造をもとに実施する．各小問には，なるべく少数の学習目標しか含まれないように分割する．初めの小問では課題を実装する上で基礎となる内容を扱うようにし，小問が進むにつれて重要な部分や難しい部分を扱うように分割する．

§立命館大学情報理工学研究科
†立命館大学情報理工学部
‡三菱電機コントロールソフトウェア

各小問で学習者が実装するプログラムはもとの課題で実装することになっていたプログラムと同等の動作をするプログラムになる．同じ動作をするプログラムを実装させることで，実装した箇所のプログラム上での働きを学習者に理解させる．各小問で同じ動作のプログラムを学習者に実装させるために，後の小問で学習者が実装する部分の機能を関数として学習者に提供する．それらの関数を提供関数と呼ぶ．学習者は提供関数を使い課題を進める．提供関数の機能を適切に設定することで，学習目標と小問で実装すべき内容との対応を明確にできる．各学習目標に関連する機能のみを学習者が実装すれば，プログラムが正しく動作するように提供関数を準備する．

本手法は以下の特長を持つ．

- 学習者は提供関数の使用を義務付けられるので，学習目標と学習者が実装するソースコードとの対応関係が明確になる．
- 学習者が実装するソースコードは模範解答に類似した綺麗なソースコードになる．
- 綺麗なソースコードの場合，書かれるソースコードの形式を限定できるため，学習者が書いたソースコードの制御構造が正しく書けているかの検証が可能になる．

学習者が実装したソースコードが綺麗な場合，そのソースコードは模範解答と類似するため，学習者の行き詰まっている箇所の特定が容易になる．学習者がコンパイルしたときに検証することで，学習者が一区切りついたと考えるタイミングで検証できる．また，コンパイル頻度からコンパイルしていない学習者，検証結果から点数の伸びが悪い学習者が判明する．

3.2 提供関数を用いたプログラミング

課題の学習目標を明確にすることや提供関数を用意することは学習者にヒントを与えることに相当する．多くのヒントを学習者に与えることで学習者が実装するソースコードは模範解答と類似する．

本手法では小問が進むにつれて学習者が実装するソースコード量が増えていく．図 1 の中心部に示すように初めの小問では多くの機能が提供関数として与えられる．学習者には提供関数を使うことを義務付けるため，提供関数を使うためのソースコードを学習者は実装する．2 問目以降の小問では前の小問で実装したソースコードを再利用し，前の小問で使った提供関数と同等の機能の実装を学習者に要求する．前の小問までに実装した箇所と整合性の取れるようなソースコードを新たに学習者は実装する．そのため，学習者が書くソースコードは前の小問にあった提供関数の仕様に基づいたものになる．提供関数に基づいたソースコードを小問ごとに実装することで，学習者が実装するソースコードが模範解答と類似する綺麗なソースコードとなる．

3.3 模範解答との制御構造比較

学習者が実装するソースコードが模範解答と類似することを前提にできるため，ソースコードの制御構造をもとにした検証ができる．学習者が書いたソースコードの制御構造と模範解答の制御構造を比較すると，学習者が正しく制御構造を実装できているかがわかる．

制御構造を比較するために，提出されたソースコードに含まれる制御構造のうち代入，反復，条件分岐，関数定義，提供関数をノードとする木構造を生成する．変数の名前や条件式，関数定義の順番といった詳細な部分を無視し，前述の制御構造が存在することを示す単純な制御構造の木構造とする．同様の木構造を模範解答のソースコードからも生成する．模範解答をもとにした木構造と学習者が書いたソースコードから作られた木構造を編集距離をもとにして検証する．学習目標と対応するより重要な構造を実装できているかを検証できるようにするため，各ノードに重みを設定できる．例えば図 2(a) を含むソースコードを学生が提出した場合，これをもとにした木構造は図 2(b) になる．図 3(b) における LOOP に添えられた数字が重みである．この木構造では関数を:FUNCT，代入文を:ASSIGN， for 文を:LOOP， 提供関数を:PROV で表わしている．また，scanf 文は代入文の一種であると考え:ASSIGN として表わしている．図 3 に模範解答のコードとそれをもとにした木構造を示す．

```
main(){
  int i,j,x[100],y[100];
  scanf(“ %d ”,&i);
  for(j=0;j<i;j++){
    scanf(“ %d ”, &x[j]);
    scanf(“ %d ”, &y[j]);
  }
  r_printCosine(x,y,i);
}
```

(a) 元になるソースコード

```
:FUNCT
  :ASSIGN
  :LOOP
    :ASSIGN
    :ASSIGN
  :PROV
```

(b) 生成される木構造

図 2: 学習者のソースコードと木構造

```
main(){
  int i,j,x[100],y[100];
  scanf(“ %d ”,&i);
  for(j=0;j<i;j++) /*[5]*/
    scanf(“ %d ”, &x[j]);
  for(j=0;j<i;j++) /*[5]*/
    scanf(“ %d ”, &y[j]);
  r_printCosine(x,y,i);
}
```

(a) 元になるソースコード

```
:FUNCT
  :ASSIGN
  :LOOP [5]
    :ASSIGN
  :LOOP [5]
    :ASSIGN
  :PROV
```

(b) 生成される木構造

図 3: 模範解答のソースコードと木構造

表 1: 計算過程

	F	_A	_L[**5**]	__A	_L[**5**]	_A	_P
FUNCT (F)	0 [0]	1 [0]	2 [5]	3 [5]	4[10]	5[10]	6[10]
_ASSIGN (A)	1 [0]	0 [0]	1 [5]	2 [5]	3[10]	4[10]	5[10]
_LOOP (L)	2 [0]	1 [0]	0 [0]	0 [0]	2 [0]	3 [0]	4 [0]
__ASSIGN	3 [0]	2 [0]	1 [5]	0 [0]	1 [5]	2 [0]	3 [0]
__ASSIGN	4 [0]	3 [0]	2 [5]	1 [0]	0 [5]	1 [5]	2 [5]
_PROV(P)	5 [0]	4 [0]	3 [5]	2 [0]	1 [5]	1 [5]	**1 [5]**

木構造の深さを考慮しながら，表 1 のように編集距離を計算する．この表は，行に学習者のソースコードからできた木構造のノード，列に模範解答の木構造のノードを示し，これらの間の違いを距離として，各セルに示している．左上のセルから右下のセルに至る最短経路のルートを動的計画法で求め，その最短距離を編集距離とする．また，各セルの [] 内の数字は，各セルの重みである．重み付けを計算する式を式 1 に示す．

模範解答の木構造のノードを $N[]$ ノード数を $xMax$,

学習者のソースコードをもとにした木構造のノード数を $yMax$ とする．$xMax \times yMax$ 行列を $d[][]$ として作成し $d[0][0]$ から順に $d[xMax][yMax]$ まで式 1 を用いて計算する．$w[x]$ は模範解答の木構造のノード $N[x]$ につけられた重みを $W[x]$ として, 式 2 に従って計算される．

$$d[x][y] = Max(0, Min(d[x-1][y], d[x][y-1], d[x-1][y-1])) + w[x] \quad (1)$$

$$w[x] = \begin{cases} 0 & (w[x] = \emptyset)\ or\ (N[x] = N[y]) \\ W[x] & (otherwise) \end{cases} \quad (2)$$

重み付け計算の結果が，$d[xMax][yMax]$ に得られる．

本計算法での編集距離は，模範解答のソースコードより学習者のソースコードがいくつのノードで異なっているかを示している．重み付け計算の結果は，学習目標に掲げられている項目に対応するノードが実現されているかを重視した，模範解答のソースコードからのずれである．これら 2 つのずれをもとに，学習者が書いたソースコードを評価するために式 3 を用いる．

$$\left\{1 - \frac{(\text{編集距離}) + (\text{重み付け計算結果})}{(\text{模範解答のノード数}) + (\text{重みの総和})}\right\} * 100 \quad (3)$$

模範解答と異なるノードが学習者のソースコードの木構造に表われ，そのノードが重みづけられているとき，より大きく減点されることを式 3 は意味している．

分子は編集距離と重み付け計算の結果を合計したもの，分母を模範解答のノード数と重みの合計として，100 点満点換算する．この例では，編集距離が 1，重み付け計算の結果が 5 であるため分子は 6，模範解答のノード数から 7，重みの合計が 10 であるため分母は 17 となる．よって，この例の制御構造に基づいた検証結果は $1 - (5+1)/(7+10) * 100 = 65$ 点となる．

3.4 行き詰まる箇所の特定

学習者が実装するソースコードが模範解答と類似するため，学習者の理解状況を推測するための情報を課題から得られる．学習目標は学習者にとって新しい内容であることが多いため，小問の中で学習目標と対応する部分が学習者の間違える可能性の高い箇所となる．模範解答は指導者が小問の学習目標にあわせて実装したものであるため，ソースコードと学習目標の対応する箇所が指導者にはわかる．また，学習者の実装するソースコードも模範解答と類似するため，指導者は学習者が書いたソースコードに対してもどの構文がどの学習目標と一致するものなのかを推測できる．さらに，図 1 に示したように学習者は前の小問で実装したソースコードを再利用して小問を実装していく．提供関数とソースコードの再利用のため，学習者が新しく実装する箇所は非常に限られる．再利用したソースコードは既に正しく記述されているため間違うことはなく，学習者が間違えるのは新しく実装する箇所である．このようにして，学習者が間違える可能性の高い箇所が想定できるため，学習者が実装したソースコードから間違い箇所を見付けやすくなる．

また，小問ごとに学習目標との対応が明確に設定されているため，学習者の取り組んでいる小問によって学習者の大まかなレベルを想定できる．初めの小問では，プログラミングの基本となる動作を実装する小問になる．これらの小問で間違える学習者は基礎的な内容の理解ができていない学習者となり基礎からの指導が必要となる．後の小問では，新しく取り組む学習目標を設定することが多く，後の方で間違える学習者には，基礎はできているため新しく学習する内容についての指導を与えればよい．検証の結果からも学習者の状況を把握できる．低評価を取る学習者は制御構造を正しく実装できていないため，制御構造についての指導が必要になる．高評価を取る学生は，制御構造よりも，より詳細な部分ができていないため，制御構造についての指導ではなく制御の中身についての指導が必要である．

4. 実験

本手法を用いた課題と用いなかった課題を出題する演習をした．学習者は模範解答と類似したソースコードを実装するか．学習目標と課題との対応を明確にできているか．学習者は学習目標に設定した箇所を間違えるかの検証を目的とした．学習目標は，学習者にとって新しい内容である，もしくは，理解しがたい内容であることが多い．よって，学習者は学習目標で設定された項目に対応する箇所を実装するさいに間違えることが多い．

学習者はサーバにログインし，サーバ内でプログラムを開発した．学習者がプログラムをコンパイルしたさい，検証システムにソースコードが自動的に送信される．検証システムに送信されたソースコードと検証結果をデータベースに保存し，検証結果は端末に表示した．

被験者は立命館大学情報理工学部の 1 回生，4 回生以上，同大学院生の 11 名である．90 分間の演習を週に 1 度，2 週実施した．ただし，希望者は 90 分間の演習終了後も課題に取り組めるようにした．プログラミング言語として C，演習環境として OS は Linux，エディタは emacs を採用した．課題は手法を適用した課題としてバブルソートを実装する課題および木構造をたどる課題，手法を適用していない課題として選択ソートを実装する課題およびバブルソートを実装する課題の大問 4 問を出題した．手法を適用したバブルソートの課題では小問を 4 問，木構造をたどる課題では小問を 3 問出題した．

本手法では，課題内容で指定された項目が理解できているかを問うための小問が出題される．バブルソート課題 1 問目では，与えられた提供関数を使えるかが問われ，バブルソート課題 2 問目では，これら提供関数のうちのひとつを独力で実装することが学習者に求められる．このように，小問を進むにつれ，提供関数を学習者が独自に実装する形で全体のプログラムを実装することになる．実装すべき関数が複数あるなかで，各小問で実装するべき関数が限定されるので，学習目標に設定した項目を学習者が理解しているかを判定できる．

課題例として手法を適用したバブルソート課題を説明する．バブルソート課題で実現すべきプログラムの基本的な流れは以下のようになる．(1) 初めにキーボードで値の個数を入力し，(2) 次にキーボードからソート対象となる値を入力していく．(3) 入力した値を昇順に並べる．(4) そして，結果を表示する．

バブルソート課題 1 問目の問題文を図 4 に示す．1 問目では，上記 (1)，(2) を学習者は独力で実装し，提供関数 r_bubbleSort() を用いて上記 (3) を実装し，提供関数 r_printArray() を用いて上記 (4) を実装する課題である．提供関数 r_bubbleSort() は引数に与えられた配列を昇順

キーボードから任意の数 (n 個) の整数値を入力し，
入力された値を昇順に並び換えて表示する
プログラムを下記の提供関数を用いて実装せよ．
要素数 n はキーボードから入力するものとする．
ソートの方法はバブルソートを利用すること．
提供関数
```
    void r_bubbleSort(int x[], int max);
    void r_printArray(inx x[], int max);
```
r_bubbleSort():配列の要素を昇順に並べ替える関数
r_printArray():配列を印字する関数
[学習目標] 提供関数を使える
配列に値を代入できる

図 4: バブルソート課題 1 問目 問題文

バブルソート課題 1 で利用した，
提供関数 r_printArray() と
同等の関数 printArray() を実装せよ．
提供関数 r_bubbleSort() は引き続き利用すること
[学習目標] 可変長の配列の値表示ができる

図 5: バブルソート課題 2 問目 問題文

```
void printArray(int x[],int max);
int main(){
  // 要素数 max, 配列 x に値を代入する処理
  r_bubbleSort(x,element); //提供関数
  printArray(x,element);
}
void printArray(int x[],int max){
  int tmp;
  for(tmp=0;tmp<max;tmp++)
    printf("%d ",x[tmp]);
  printf("\n");
}
```

図 6: バブルソート課題 2 問目 模範解答抜粋

に並べ替える関数であり，提供関数 r_printArray() は引数に与えられた配列を印字する関数である．バブルソート課題 2 問目の問題文を図 5 に示す．2 問目では，1 問目で提供されていた提供関数 r_printArary() と同等の関数を学習者が独力で実装する課題となる．そのため，バブルソート課題 2 問目では上記 (4) の機能を学習者は実装する．模範解答を図 6 に示す．

バブルソート課題 2 問目の学習目標は図 5 に示したように,「可変長の配列の値表示ができる」である．そのため，配列の値を表示する関数以外の機能は前の小問で実装済みであるか，もしくは，提供関数で提供されている．バブルソート課題 2 問目で学習者が新しく実装するソースコードは図 6 の printArray() と上部のプロトタイプ宣言である．バブルソート課題 3 問目以降では，提供関数 r_bubbleSort() を実装していく課題となる．

5. 評価

表 2 に学習者がコンパイルしたソースコードの実装状況をまとめたものを示す．表 2 における Q1 は手法未適用の選択ソートを実装する課題，Q2-1 から Q2-4 は手法を適用したバブルソートを実装する課題，Q3 は手法未適用のバブルソートを実装する課題である．各数字は，示された実装状況にあてはまるソースコードの個数である．

表 2: ソート課題 実装状況

実装状況	Q1	Q2-1	Q2-2	Q2-3	Q2-4	Q3
正しく入力できない	**4**	**6**	**0**	**0**	**0**	2
正しく入力できない ソート部分を実装	**20**	**0**	**0**	**0**	**0**	7
正しい順番ではない	49	0	0	19	6	21
結果を表示していない	13	5	0	4	3	6

本手法を適用していない Q1 では，表 2 に示したように値の入力機能が正しく実装できていないにもかかわらず，ソート機能を学習者が実装しようとする場合が発生している．このような状況で学習者が指導者にソートの条件式について指導を求めた場合，指導者はソースコードから学習者がソートの条件式の前段階ができていないことを見付け出し，指導する必要がある．しかし，指導者は条件式についての質問であるという前提を持って学習者のソースコードを精読するため学習者の間違いに気付くことは難しい．

一方，表 2 の Q2-1 の列が示すように本手法を適用した課題では，提供関数によって学習者が実装すべき内容を切り分けて出題しているため，入力機能が正しく実装できていないソースコードでは，ソート機能を学習者は実装しない．各小問での学習目標が達成できなければ，次の小問には進めないので，小問 Q2-2 以降の手法を適用した 3 つの小問では「正しく入力できない」という行に該当するソースコードは提出されていない．

本手法を用いることによって，学習目標に則ったソースコードを学習者が実装しているかを容易に判断できた．

6. おわりに

本論文では，プログラミング初学者が，学習目標との対応が明確，かつ，模範解答と類似した，綺麗なソースコードを記述できるようにする手法を提案した．今後の課題として，本手法の長期的な有効性の検証，プログラム実装能力に関する影響の検証が挙げられる．

参考文献

[1] Robert E. Salvin, Educational Psychology, PEARSON, 2012.

[2] 倉田 英和， 富永 浩之， 林 敏浩， 山崎 敏範， 実行テストを用いたコンテスト形式の入門的 C プログラミング演習の大会運営サーバの開発，情報処理学会研究報告． コンピュータと教育研究会報告，2006(108)， 9-16， 2006.

[3] 西 輝之，劉 渤江，横田 一正，デバッガの連携による 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.

[4] 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.