プログラミング学習における意図を考慮した空欄補充問題の自動生成

2009SE040 長谷川 靖成　2009SE234 大竹 諒　2009SE271 田島 侑典
指導教員：蜂巣 吉成

1　はじめに

プログラミング言語の学習では，演習問題を繰り返し解き，多くのプログラムに触れることが重要である．プログラムをすべて記述するような演習問題では，採点者はプログラムを目視し，コンパイル，実行して動作確認するなどの作業が必要となり，負担が大きい．

採点の手間を軽減する方法として，moodle などの教育支援システムを用いて，プログラムの一部が空欄になった問題を出題して，自動採点を行う方法が知られている．この方法は，問題作成者が学習者に学習して欲しい内容に合わせて，プログラム中の適切な箇所に空欄を設定して多くの問題を作成しなければならない．自動採点を行う際，解答も用意しなくてはならないが，プログラミング言語には，正解となる記述が複数ある場合があり，全ての解答を列挙することは手間である．

本研究では，これらの問題を解決するために，学習意図を考慮して，プログラムミング言語の空欄補充問題の自動生成を行う方法を提案する．学習意図とは，問題作成者が学習してほしいと思う内容であり，文法や条件式の記述方法，変数の初期化の必要性などである．一般的に一つのプログラムは，変数宣言，代入文，入出力文，条件分岐，繰り返しなどの複数の要素から構成されている．適用する学習意図を変更することで一つのプログラムから複数の空欄補充問題を作成できる．学習意図は複数組み合わせて使えるが，生成された問題の品質について考える必要がある．例えば，空欄箇所の数が多すぎると問題の難易度があがってしまう．また，解答によって他の空欄箇所の解答が決まるような，相関性がある問題もある．

本研究の技術的課題は次の通りである．

1. プログラミング学習における学習意図の分類とパターン化
2. 複数解答がある場合の解答自動生成
3. 相関性のある空欄箇所の解答自動生成
4. 複数の学習意図を組み合わせた際の問題の品質

2　空欄補充問題の自動生成における関連研究

「初級プログラミング学習のための自動作問システム」[1] は，対象言語は Java であり，問題作成者は，問題となるプログラムとその説明をシステムに登録する．システムは登録されたプログラムからキーワードや識別子を抽出し，その部分に空欄をあけることで空欄補充問題を生成する．問題中の空欄は乱数によって設定されるので，毎回異なった箇所に空欄が生成されるので，同じプログラムでも異なった問題が生成される．反復練習が可能となるが，識別子や予約語などを指定できないので，出題者の意図に沿った学習意図を細かく指定できない．

「学習者に合わせた C 言語演習穴埋め問題の自動生成」[2] では，本研究と同じように，プログラムを解析し，そこに問題作成者の問題作成意図を適用することで，演習問題の自動生成を行っている．問題点として，構文を利用して空欄箇所を指定しているが，問題作成意図が提案されている方法で表現できるか明確でない．複数解答についても考察されていない．

「いかにプログラム空欄補充問題を作るか?」[3] では，プログラムにおける処理の流れの要所となる部分を PDG(プログラム依存グラフ) を用いて見つけ，その箇所を空欄として設定する．PDG とは，プログラムの各文と文の間の依存関係を示す有向グラフであり，データの流れを定量的に判断できるので，データが集中している文が流れの中心であると判断することが出来る．しかし，問題作成者の意図に合わせた空欄箇所を設定することが出来ない．文法を理解している上級者向であり，本研究で対象としている学習者には合わない．

3　空欄補充問題の自動生成

本研究での空欄補充問題は初学者向けの問題とし，プログラミング言語の文法を，条件分岐や繰り返しなどの単元毎に学習していくとする．3.1 節では問題作成に用いる学習意図について述べる．3.2 節では解答の自動生成における問題について説明する．

3.1　学習意図

C 言語などのプログラミング言語の教科書を参考にして，単元毎に入出力，演算と型，分岐，繰返し，配列，文字列，関数，ポインタ，構造体の学習意図を考え，問題集 [4] などを参考にしてどこが学習意図に応じた空欄箇所なのかを調査した．学習意図の例を次に示す．

1. 分岐
   (a) if の理解
   if($\boxed{a > 1}$)~
   (b) if~else の理解
   if(a > 1)~ $\boxed{\text{else}}$
   (c) if~else if の理解
   if(a > 1)~else if($\boxed{b > 1}$)~
2. 繰返し
   (a) while 文の初期値の理解
   $\boxed{a=1}$;
   while(a > 0){
   (b) for 文の初期値の理解
   for($\boxed{i = 1}$;a > 0;i++)

(c) for 文の条件式の理解
for(i = 1 ; $\boxed{a > 0}$;i++)

(d) 繰返しの中で代入される変数の初期値の理解
ans = $\boxed{1}$;
for(i = 1 ; i<=n ; i++)
ans = ans*i

3.1.1 学習意図の表現方法

学習意図を指定する方法として，構文要素で指定できる意図，構文要素で指定できない意図があることがわかった．

1. 構文要素で指定できる意図
   (a) 制御構文の式
   if の条件式，for の初期値，for の条件式，while の条件式
   (b) 予約語
   if，for，while，do，switch，case，default
   (c) 演算子
   条件式の論理演算子，アドレス演算子
   (d) 関数
   return 文における関数の返り値，関数呼び出し式
2. 構文要素で指定できない意図
   (a) 同一変数に対する計算代入
   (b) 繰返しの中で代入される変数の初期化文

2 の学習意図は左辺と右辺で同じ変数が出現する代入文や，while 文の条件式の中で使われている変数の while 文の前の代入式などを指定しなければならない．

3.1.2 パターン記述

プログラム断片や構文単位でプログラムを解析することができ，決まった文の並びなどをパターンで書くことができる解析器である TEBA[5] を利用してパターンを表現する．TEBA により構文だけでなく 3.1.1 節の 2 の意図も記述できる．

パターン例

```
sum.pt「同一変数に対する計算・代入の理解」
% before
${var1:ID_VAR}=
${var1}${ope:OPE}${var2:ID_VAR};
%after
${var1}=<Q>${var1}${ope}${var2}</Q>;
%end
```

上記のパターン例は，TEBA を用いたパターン変換記述の例である．学習意図ごとにパターンがあり，印<Q></Q>をつけて空欄箇所として設定している．

3.2 解答の自動生成

空欄補充問題を作成する際，問題作成者が空欄箇所の答えとなりうる解答を設定し，別解が存在する場合も考慮する必要がある．しかし，設定された空欄箇所に入る解答を別解も含め作成するのは負担である．そこで，空欄補充問題を自動で生成すると同時に，解答も自動生成する方法を提案する．以下では解答を自動生成する際に考慮すべき点について述べる．

3.2.1 複数解答

空欄箇所には解答が複数存在するものがある．ここではその解答群を複数解答と呼ぶ．本研究では，空欄となる以前の元の記述から，複数解答の自動生成を行う．

3.2.2 同じ意味となる式

同じ意味となる式の記述の例は以下のものが挙げられる

- a==b, b==a
- a>=b, b<=a
- a+b, b+a
- i++, i=i+1, i+=1

これらの同じ意味となる複数解答の自動生成は問題生成と同様に TEBA のパターンを利用し自動生成を行う．

3.2.3 相関性のある解答

ソースコード内に空欄箇所が二つ以上ある場合，その空欄間に相関性が生じる可能性がある．二つ以上の解答が複数ある空欄箇所において，一つの解答が決まると他の解答が決まる場合，その空欄箇所の間には相関性があるとする．例えば，空欄１と空欄２があり，１でＡと答えた場合，２の解答はＢ，１でＣと答えた場合，２の解答はＤとなる場合がある．空欄間に相関性がある場合，学習者によって解答が変わってくるので解答判定が難しい．相関性が生じる可能性がある空欄箇所となる場合は次の 3 通りである

1. for 文の初期値と終端条件の部分に空欄が設定された場合
2. for 文の初期値と終端条件増分の，部分に空欄が設定された場合
3. scanf 文が二つ以上あり入力変数の部分に空欄が設定された場合

相関性の判定方法

4 節であげた 3 通りの相関性について判定を行う方法を述べる．for に関する 1，2 は同じ方法で判定可能であるので２つをまとめて，for に関する相関性，3 を scanf に関する相関性とする．

どちらの相関性の判定も，ソースコードを字句解析した結果を基に行う．相関性が生じる可能性がある箇所に，空欄が設定されているソースコードのみを対象に相関性の判定を行う．

for に関する相関性

for に関する相関性は，for 文の初期値や終端条件の部分で使われた変数が，同じ for 文のループ内の処理で使われていない場合に生じる．よって，for 文の該当する場所で使われている変数を記憶しておき，その変数を for 文のループの処理内から探し，見つかった場合は相関性は生じない．見つからなかった場合は相関性が生じると判定する．また回数を数える繰返しにおいて初期値と条件の値の組み合わせは無数に考えられるが，一般的に 0 か 1 を基準に数える場合が多いので，本研究でもその形に限定し相関性の解答を生成する．

for 文の初期値が 0 か 1 で条件式の値が 2 以上の数字となる場合

for 文の形が for(i = 0; i < #; i++) の場合は，
i=1, i< @を相関性の解答として生成する．#は定数であり，@は#に 1 加えた数値である．

for 文の形が for(i = 1; i < #; i++) の場合は，
i=0, i< *を相関性の解答として生成する．#は定数であり，*は#から 1 引いた数値である．

for 文の初期値が 2 以上の数字で条件式の値が 0 か 1 となる場合

for 文の形が for(i = #; i > 0; i--) の場合は，
i=@, i> 1 を相関性の解答として生成する．

for 文の形が for(i = #; i > 1; i--) の場合は，
i=*, i> 0 を相関性の解答として生成する．#，@，*は上記と同じ意味である．

for 文の初期値が 0 か 1 で条件式の値が変数となる場合

for 文の形が for(i = 0 ; i < n; i++) の場合は，
i=1，i<= n を相関性の解答として生成する．

for 文の形が for(i = 1; i <= n; i++) の場合は，
i=0，i< n を相関性の解答として生成する．

scanf に関する相関性

例

身長と体重を入力して，BMI を計算して表示するプログラム

```
double height, weight, bmi;

printf("身長を入力してください ");
scanf("%lf", &height );
printf("体重を入力してください ");
scanf("%lf", &weight );

bmi = weight/(height*height) ;

printf("あなたの BMI = %f\n",bmi);
return (0);
```

scanf 文の入力変数の箇所が空欄であり，そこで使われた変数が scanf 文以降の記述で空欄になっている場合，相関性が生じる場合がある．しかし，プログラミング言語を学ぶ場合，変数名は重要なものであり，適切な変数名を使用することが望ましい．入力を促す printf 文で適切なメッセーシを表示することで，scanf 文における適切な入力変数を決めることができる．よって scanf 文に関しては相関性がある問題としない．例では height は身長を意味する単語であるので，身長の変数名として扱い，weight は体重を意味する単語であるので，体重の変数名として扱い，相関性は生じない．

4 空欄補充問題の自動生成システム

空欄補充問題作成の際の問題作成者の負担を軽減するために，ひとつのソースコードから複数の空欄補充問題を作成するツールの作成を行った．選択された学習意図から，元となるソースコードに含まれる学習意図の組み合わせの数の空欄補充問題を生成する．

空欄補充問題の自動生成ツールの実現にあたり，シェルスクリプトを用いて，元のソースコードが空欄補充問題に変換されるまでに必要な処理をまとめた．

4.1 学習意図の選択方法

問題作成者が学習意図を選択する際，数多くある学習意図をひとつひとつ選択するのは手間がかかる．そこで学習意図を if に関するもの，while に関するものなどのカテゴリごとに分類し，そのカテゴリを選択することで問題に意図を反映させる．カテゴリは複数選択でき，カテゴリに含まれる学習意図のすべての組み合わせを適用する．

4.2 システムの概略

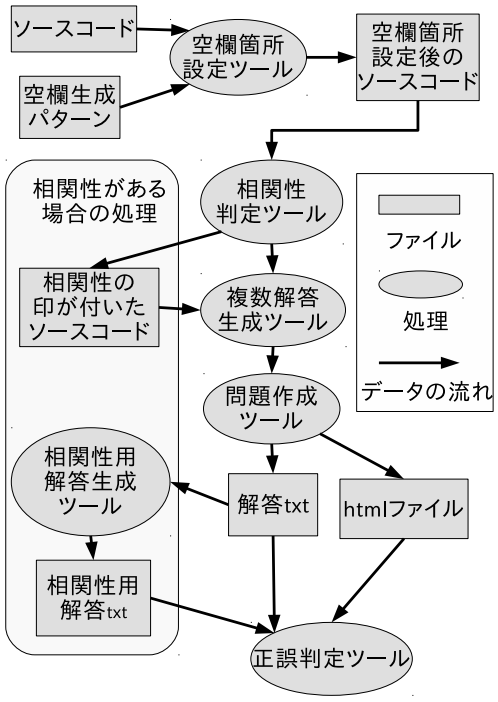

図 1 システムの流れ

提案する空欄補充問題の自動生成システムの処理の流

れを図 1 に示す.

問題作成者は空欄補充問題の元となるソースコードを用意し，適用させる学習意図を選択する．このソースコードはどのような処理を行うものなのかコードの初めに記述されているものとする．学習意図ごとにコードの空欄箇所となる場所に印<Q></Q>を挿入する空欄生成パターンが設定されており，TEBA を利用してパターンを適用することでソースコードを書き換え，空欄箇所を設定する．

空欄箇所が設定されたソースコードを相関性を判定するツールに渡して，空欄箇所間に相関性が生じるかどうか判定する．その後，複数解答生成パターンを適用することで，解答が複数存在する場合，複数解答を生成する．

空欄箇所を設定し相関性，複数解答についての処理を行ったソースコードから，空欄箇所の印を元に空欄問題 html ファイルと正誤判定を行うための解答用 txt ファイルを生成する．

5 評価・考察

点数分布グラフ表示プログラム，素数判定のプログラム，最小値最大値を表示するプログラム，逐次探索の 4 つのソースコードを元に，一つのソースコードに対して，どれだけの学習意図が適用できるか，またその結果，いくつの空欄が設定されるのかを観点として評価を行った．表 1 は評価結果である．

表 1 評価結果

	行数	空欄箇所	学習意図
点数分布	40	26	if,for,do-while
素数判定	19	12	if,for
最大最小	26	16	if,for, 入力, 出力
逐次探索	36	11	if,while, 入力, 配列

評価結果の学習意図はソースコードに含まれる学習意図であり，空欄箇所は含まれる学習意図をすべて適用した場合に設定される最大の空欄数である．適用させる学習意図を減らすことで，様々な学習意図をもった問題が生成でき，一つのソースコードから複数の空欄補充問題を生成できることが確認できた．適用した学習意図を次に示す.

- if: if の条件式の理解，if else の理解，if if else の理解，if 文の予約語の理解
- for: for 文の初期値の理解，for 文の条件式の理解，for 文の増分の理解，for 文の予約語の理解
- while: while の初期値の理解，while の条件式の理解，while の予約語の理解
- do-while: do-while の条件式の理解，do-while の予約語の理解
- 配列: 配列要素数の理解，二次元配列の要素数の理解

実現したツールで，学習意図の選択の方法により，点数分布は 37 問，素数判定は 10 問，最大最小は 20 問，逐次探索は 19 問の問題を作成できた．提案手法により 1 つのソースプログラムから複数の問題を作成できることを確認した.

3.1 節であげた学習意図により問題集 [4] の空欄箇所の 78 %を空欄にできた．設定できなかった空欄箇所はライブラリ関数名を空欄にするものや printf 関数における書式文字列を空欄にする問題であった．関数名についてはパターンを記述すれば，関数名を空欄にすることができる．書式文字列を空欄にする場合は，文字列リテラルにパターンを記述する方法が必要である.

3.1 節では単元や構文の観点から分類したが，意味の観点から分類することも考えられる．例えば，条件式，変数の初期化，型，同一変数を使った代入文などが挙げられる．これらに該当するパターンを学習意図のカテゴリとして記述することもできる.

学習意図を複数選択する場合，空欄箇所が多く設定されすぎてしまい，難易度の高い問題や解答者が解きづらい問題が生成される可能性がある．空欄補充問題の品質の評価を行う必要がある．また，複数のカテゴリを選択する際，選択する順番を変更することで，同じカテゴリを選択しても異なる問題が生成される場合や，空欄箇所が二重に設定されてしまい，うまく問題が作成されない場合があった．学習意図をソースコードに適用させる順番や組み合わせについても調査をしていく必要がある.

6 おわりに

本研究では，複数の学習意図を用いて少ないソースコードから多くの空欄補充問題を自動生成し，出題者の負担を軽減した．今後の課題として，問題の品質の評価基準を明確にすることがあげられる．ソースコードの行数に対する空欄箇所の数や，宣言された変数がどれだけ空欄箇所として設定されているか，空欄内のトークン数などを考えているが，考察をする必要がある．学習意図を適用した際にパターンマッチした箇所すべてに空欄が設定されてしまうので，ランダムに設定することも今後の課題である.

参考文献

[1] 内田 保雄, “初級プログラミング学習のための自動作問システム,” 情報処理学会研究報告 Vol2007, No.123, pp.109–113, Dec.2007.

[2] 有安 浩平, 池田 絵里, 岡本 辰夫, 國島 丈生, 横田 一正 “学習者に合わせた C 言語演習穴埋め問題の自動生成,” 第 1 回データ工学と情報マネジメントに関するフォーラム (DEIM Forum 2009),5 pages,2009.

[3] 柏原 昭博, 寺井 淳裕, 豊田 順一, “いかにプログラム空欄補充問題を作るか？,” 電子情報通信学会技術研究報告, Vol.99, No.81, pp.9–16, May. 1999.

[4] 柴田 望洋, 赤尾 浩, 肘井 信一, 高木 宏典, 解きながら学ぶ C 言語, ソフトバンククリェイティブ株式会社,2004 年.

[5] 吉田 敦, 蜂巣 吉成, 沢田 篤史, 張 漢明, 野呂 昌満, “属性付き字句系列に基づくソースコード書き換え支援環境,” 情報処理学会論文誌 Vol.53, No.7, pp.1832-1849, July 2012