穴埋め問題における誤答の分類を用いた プログラミング教育の改善提案

中山 陽平[†1]　掛下 哲郎[†1]

概要：我々は効果的なプログラミング教育のための支援ツールとして pgtracer を開発している。 pgtracer はプログラムとトレース表から構成される穴埋め問題を出題し、学生の答案を自動的に採点する。また、学生の解答データとその採点結果をログデータとして収集する。本論文では統計解析用言語 R を用いて、pgtracer で収集したログデータを分析する。問題と穴埋めの難易度を評価し、学生の理解度を把握することでラーニングアナリティクスを実現する。そのため、熊本高専の授業で使用した 18 個の穴埋め問題（穴抜き総数 127）について、学生の答案毎に誤答を分類した。これにより、良くある誤答を抽出した。また、誤答の分類結果についてアソシエーション分析を行い、同時に出現しやすい誤答の組み合わせを特定した。穴埋め問題をクラスタ分析することで、学生の能力を評価するテスト問題や能力向上を目的とした練習問題を構築するためのクラスタを特定した。

キーワード：統計解析言語 R、穴埋め問題、プログラミング教育、ラーニングアナリティクス、階層的クラスタ分析、アソシエーション分析

A proposal to Improve Programming Education by Utilizing Classification of Incorrect Answer of Fill-in-the-Blank Question

YOHEI NAKAYAMA[†1]　TETSURO KAKESHITA[†1]

Abstract: We are developing pgtracer as a support tool for effective programming education. Pgtracer provides a fill-in-the-blank problem consisting of a program and a trace table, and automatically evaluate student answers. At the same time, student's answer and the evaluation results are collected as log data. We shall analyze the log data using statistical analysis language R in this paper. Then we can implement learning analytics by evaluating the problems and difficulty level to analyze students' level of understanding. To this end, we classify the type of incorrect answer in the student answer for 18 fill-in-the-blank questions with 127 blanks collected at Kumamoto National College of Technology. Then frequently observed incorrect answers are identified. In addition, cluster analysis was performed on the classification results of incorrect answers to identify common combinations of incorrect answers appearing simultaneously. Through the cluster analysis of questions, we identified the clusters for constructing test questions to evaluate students' ability and exercises for competency improvement.

Keywords: Statistical language R, fill-in-the-blank questions, programming education, learning analytics, association analysis

1. はじめに

プログラミング教育は大学や高専において重要性が高い。2020 年度からは小学校から順次プログラミング教育が始まろうとしており。プログラミング教育の重要度がますます高まっている。我々は効果的なプログラミング教育を行うためにプログラミング教育支援ツール pgtracer を開発している[1,3]。pgtracer はプログラミング穴埋め問題を学習者に提供するほか、自動採点、学習データの収集などの機能を持ち、学習者および授業担当教員を支援する。

pgtracer は佐賀大学と熊本高等専門学校で導入され、数年間にわたり学習データを蓄積してきた[2,4,5]。佐賀大学では情報専門学科に所属する 2 年生を対象とし、熊本高専ではプログラミング初学者を対象として学習データを収集している。我々は pgtracer を用いて収集した学習データを用いて個別の穴埋めや問題の難易度を評価し、学生の理解度を把握することで、より良い穴埋め問題の作成を支援する。また、プログラミング教育の改善につながる知見の獲得や、学生の理解度に対する適切な評価を目指す。佐賀大学と熊本高専で収集した学習データを分析することで、プログラミングに関する前提知識の有無にかかわらず、共通の分析手法を見出すことを目指している。

本論文では熊本高等専門学校で収集された学習データ（18 個の穴埋め問題・穴抜き総数 127）を用いて分析を行う。情報工学を専門としていない学生を対象とする専門科目「プログラミング基礎 I 」2017 年度、受講生 130 名の学習データを用いている。2018 年度には、同じデータを用いて pgtracer を用いたプログラミング科目の実践報告として学生の学習行動や導入状況を報告したが[5]、本論文ではより詳細な分析を行い、プログラミング教育を改善するための知見を得ることを目的とする。

本論文では、学習データの分析に統計解析用言語 R[6]を用いている。R では主成分分析やクラスタ分析、回帰分析など 20 種類ほどの高度な分析方法が適用でき、Learning

†1 佐賀大学
Saga University

Analytics を実現できる。また、無料で利用できるデータ解析環境として大学だけでなく、企業にも普及しつつある。

2018 年度の研究では穴埋め問題についての正解率と解答所要時間の関係を調べた。その結果、穴抜きの解答所要時間と正解率は､必ずしも相関関係にないことが分かった。また、解答所要時間は文字列の最小単位であるトークン数に依存する傾向がある。学習データの分類では穴抜きと学生の 2 種類のデータに対してクラスタ分析を行った｡穴抜きの分類では全てのクラスタの解答所要時間と特定のクラスタの正解率について有意差のある穴抜きを分類できた。また、解答所要時間に有意差のある学生のクラスタを検出し、分類した学生の解答過程を観察した。

本論文では学生の入力する答案に注目し、誤答の種類の把握と分類を行った。これにより、良くある誤答を抽出できる。また、アソシエーション分析を用いて誤答の種類の間に成立する関連を抽出した。これらの結果を活用することで、教員は講義を行う際に注意して説明すべき箇所を把握できる。また、学生の能力を測るテストや、特定の教育項目に関する能力向上を目的とする練習問題といった目的に適する問題を作成するための基礎データとなる。

本論文は以下のように構成されている。2 節でプログラミング教育支援ツール pgtracer および pgtracer を用いた教育プロセスを説明する。3 節では、分析対象の穴埋め問題について説明し、4 節で学生の答案を分類するための基準と分類結果を示す。5 節では穴埋め問題および個別の穴抜きにおける正解率と未解答率に基づいて、問題と穴抜きの難易度を考察する。6 節では穴抜き毎の誤答分類データを用いてクラスタ分析を行い、問題の分類と誤答分類の 2 つの観点からプログラミング教育に対する改善提案を行う。7 節では、誤答案のアソシエーション分析を行う。最後にまとめと今後の課題について述べる。

2. プログラミング教育支援ツール pgtracer

2.1 pgtracer によるプログラミング教育

プログラミング教育支援ツール pgtracer を用いた教育プロセスを図 1 に示す。pgtracer はプログラミング初学者を主な対象とした演習や補習を行うためのツールとして開発している。そのため、プログラムを 1 から記述させるといった出題形式ではなく、穴埋め形式の問題を採用している。プログラムとトレース表に対する穴埋め問題を学生に出題し、学生が解答した際の点数や開始時刻、終了時刻、穴埋めする際の答案や所要時間、最終的な答案や正誤などをログデータとして収集する。教員は収集したログデータを分析し、個々の学生や全体の理解度・不得意箇所を把握し、穴埋め問題の難易度を評価する。また、分析結果をもとに、理解度が低い箇所や不得意箇所を重点的に教育できるように問題を再作成する。このサイクルを繰り返すことにより、プログラミング教育を継続的に改善できる。

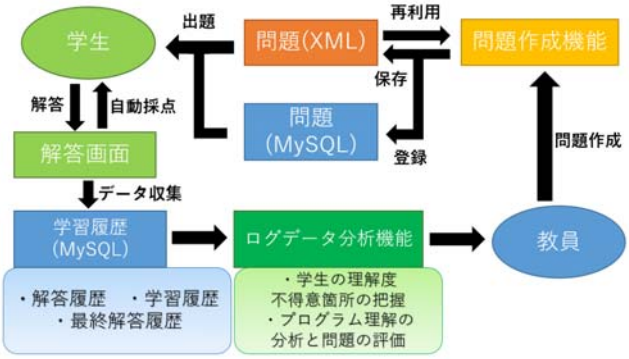


図 1：pgtracer を用いたプログラミング教育プロセス

2.2 問題出題機能

問題の登録・編集機能で登録された問題は、MySQL テーブルの情報をもとにテーマ、タイトル、難易度毎に一覧表示される。学生はテーマ、タイトル、難易度から自分の学力に合った、もしくは教員に指示された問題を選択することで解答画面を表示できる（図 3, 4)。 学生が解答画面で穴抜き個所に答案を入力し､「解答終了」ボタンを押すことで pgtracer は答案を自動採点する。それと同時に pgtracer はログデータを自動収集する。

2.3 学習ログデータ

Pgtracer で登録されている問題や学生の学習ログデータは 6 つのテーブルで保持されている。登録されている問題を定義するテーブルとして pgtracer、pgtracer_theme、pgtracer_question の 3 つがあり、実施年度、コース（カリキュラム別)、登録ユーザー、問題などのデータを保持している。学習ログデータを構成するテーブルとして図 2 に示す 3 つがあるが、このうち pgtracer_study_log では問題の開始時刻や終了時刻を、pgtracer_student_answer では最終的な答案、正解，正誤判定結果などのデータを保持している。

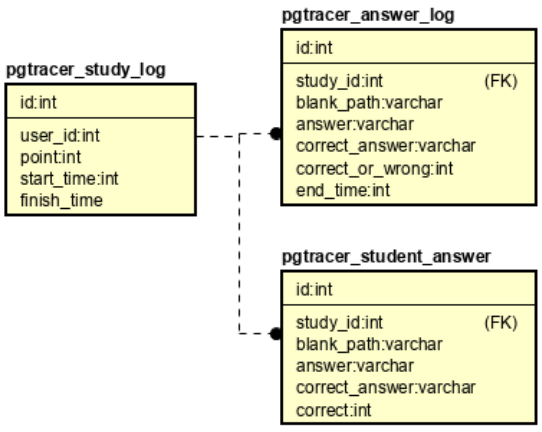


図 2：学習ログデータのテーブル

表 1：登録されている問題と穴抜き

問題番号	プログラム中の穴抜き数	トレース表中の穴抜き数	問題の説明
3-3	5	0	2 次元配列において指定された行を入れ替えるプログラム。具体的なコメント文はなく、指定する数値はトレース表から読み取らなくてはならない。
4-1	0	22	4×3の 2 次元配列に格納された学生の得点データから、得点が高いものから順位を求め表示する。
4-2	5	0	2 次元配列に格納された 3 つの点の座標から、原点と最も離れた点の座標を表示する。
5-1	0	7	整数を格納する変数とそれを指すポインタを定義し、変数に直接代入し表示する。また、ポインタを用いて数値を代入し値を表示する。
5-2	6	0	2 つの実数型の変数の和を計算して表示する。変数を参照するポインタを定義し、ポインタを用いて変数に代入する。また、ポインタを用いて変数の和を計算する。
5-3	5	0	ポインタを用いて 3 つの変数の最大値を求める。
6-1	0	11	整数 n と m を入力し、文字配列 str の n 文字目から m 文字を表示する。
6-2	5	0	文字列中の 1 文字を別の文字に置換する。
6-3	5	0	1 次元配列においてポインタを用いて要素を参照する。
7-1	5	0	整数値を入力し、n 回「Hello World!」と表示する関数を定義する。
7-2	5	0	整数の累乗の計算を行う。実数、整数計算の 2 つの関数があり、適切な引数を設定する。
7-3	0	10	関数を用いて二次関数を計算する。（引数は 0~10 の整数）
8-1	0	6	2 つの座標を入力して、原点から距離が大きいものを出力する。
8-2	5	9	2 変数を入れ替える。そのための関数を作成し、ポインタを用いて処理を行う。
8-3	5	0	文字列の中から、指定した文字を数える。一致する文字を数える関数を作成する。
11-1	3	5	入力した文字列の長さを表示する。ライブラリより適切な関数を呼び出す。
11-2	3	3	3 つの座標を入力して、原点からの距離を出力する。距離を計算する関数を定義する。
11-3	5	0	配列中の最大値を求める。配列でポインタを利用し、最大値を求める関数を作成する。
13-2	0	5	配列の要素数だけ*を表示して縦棒グラフを作図する。
14-1	0	8	2 つの整数の最大値を求める関数を用いて 3 つの整数の最大値を求める。

3. 問題の構成

Pgtracer が出題する穴埋め問題の区分は、プログラム問題とトレース表問題、それらを組み合わせた問題からなる(表 1)。トレース表はプログラムの実行過程をステップごとに変数と出力の値を示している。また、分析に用いた問題は表のようになっており、問題の数が 18 個、穴抜きの総数が 127 個であった。

学習ログデータでは一つの穴抜きで穴番号、正答、所要時間、正解率、学生の解答のデータを保持している。この学習データは最高得点時と初回受験時の 2 回保存されており、本研究では初回受験時のデータを用いて分析を行った。pgtracer で出題された問題の解答には回数制限がなく、何度も受験することができる。初回受験時のデータでは学生が理解できていない箇所や誤答が明確に現れるため、誤答分析に適する。

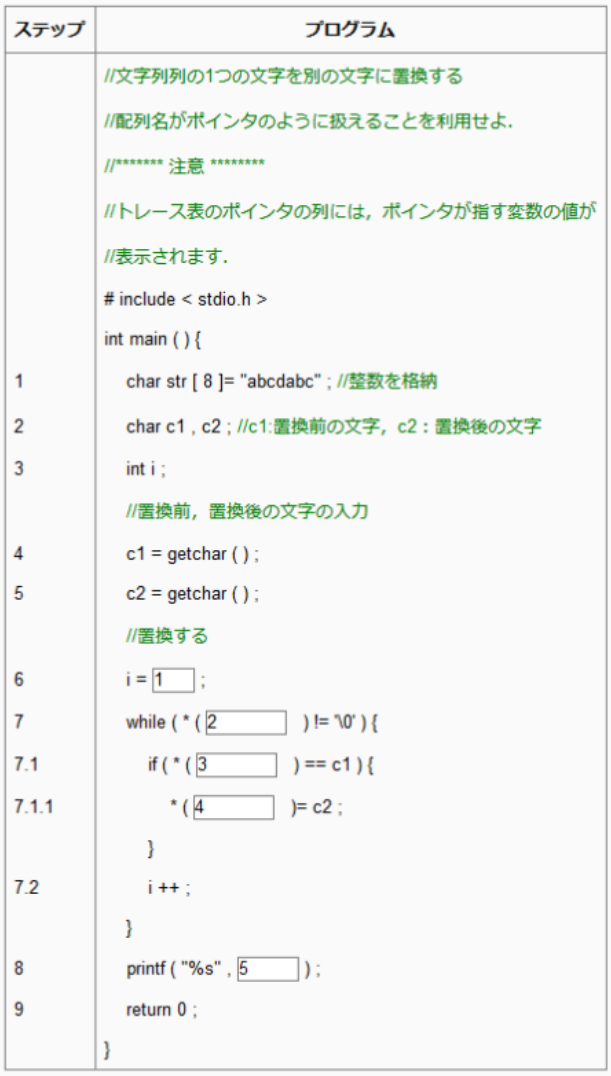
4. 穴抜きの分類

学生の理解が不十分な原因を特定するため、本研究では以下の基準に基づいて解答を分類した。分類の方法としてはそれぞれの穴抜きについて誤答に対する理由付けを行い、分類に最も近いものを当てはめた。また、誤答の詳細について異なる原因で生じたと思われる誤答は分類の種類を増やすことで対応し、分類表を作成した。分析に用いたデータの総数は 647 個であった。まず，以下の 5 つの大分類 A～E を定義した。

A) 空欄（解答内容をイメージできなかった、または解答する意思がなかった）
B) プログラム中にはコメントの形で指示を記述している。しかし、その指示が理解できていないと思われる誤答
C) コメントを用いた記述した指示は理解できているが、プログラミング言語（C++）での表現方法が分からないと思われる誤答
D) 正答
E) トレース表を正しく追跡できていない誤答

これらの大分類のうち，誤答理由が複数あると考えられるものは、より詳細な分類（例：C1～C9）を定義する。また、誤答理由が複数の大分類にまたがる場合は，最も妥当と考えられる大分類として集計する。

大分類 A では処理内容を理解できずに解答をしなかったという理由の他、自動採点後に正しい解答を見るためにあ



ステップ	プログラム
	//文字列列の1つの文字を別の文字に置換する
	//配列名がポインタのように扱えることを利用せよ.
	//******* 注意 ********
	//トレース表のポインタの列には，ポインタが指す変数の値が
	//表示されます.
	# include < stdio.h >
	int main ( ) {
1	char str [8]= "abcdabc" ; //整数を格納
2	char c1 , c2 ; //c1:置換前の文字，c2：置換後の文字
3	int i ;
	//置換前，置換後の文字の入力
4	c1 = getchar ( ) ;
5	c2 = getchar ( ) ;
	//置換する
6	i = [1] ;
7	while (* ([2]) != '\0') {
7.1	if (* ([3]) == c1) {
7.1.1	* ([4])= c2 ;
	}
7.2	i ++ ;
	}
8	printf ("%s" , [5]) ;
9	return 0 ;
	}

図 3：問題 6-2

えて何も入力せずに解答を終えたと思われるものが含まれる。また、大分類 B では書こうという努力は見られるが問題の主旨とずれた解答をしているものも該当する。大分類 C と E では問題の内容を理解している思われる解答ではあるが表現方法が分かっていないものが挙げられる。

4.1 誤答の種類に基づいた答案の分類

図 3 では実際に出題された問題（問題 6-2）を示している。問題はプログラムのみ穴抜きがあり、文字列中の 1 文字を別の文字に置換するプログラムとなっている。

問題 6-2 の穴番号 2 における学生の解答を表 2 に示す。穴抜きは配列の先頭要素を参照させる処理に該当しており、直前のポインタに気づき配列名をポインタのように使えることを理解しなければならない。

表 2 を見ると、特徴的な解答として空欄のほかに、”str[i]”といったポインタを意識しておらず、直接配列要素を解答しているものが多く見られた。このような解答は配列名を用いた配列の先頭要素の参照方法を理解できていない他、

表 2：問題 6-2 穴番号 2 の答案

答案	誤答理由	人数	分類
	空欄	40	A
char str	配列の定義は該当する穴抜き以前にあり、再定義にあたる。変数、ポインタの定義を理解していない。	1	B2
getcHar	問題の理解以前にプログラム自体の理解が不足している。指示が理解できていない。	1	C1
Str	ポインタを意識した解答だが、表現方法の理解不足である。誤字	7	C1,C8
str++	ポインタを意識した解答だが、表現方法の理解不足である。	1	C1
str+1	ポインタを意識した解答だが、表現方法の理解不足である。	1	C2
str+i	正答	29	D1
str[8]	穴抜き直前の「*」を見落としたと考えられる。不正なアクセスであるため配列の理解も不足している.	1	C2
str[i-1]	穴抜き直前の「*」を見落としたと考えられる.	1	B2
str[i]	穴抜き直前の「*」を見落としたと考えられる.	6	B2

穴抜きの直前の「*」の見落としから来る誤答であると考えられる。そのため、この解答は指示が理解できていない解答であると考えられ B に分類される。また、問題中で具体的なコメントによる指示がないため B1 と分類される。

ポインタの理解と配列の理解、その後のアルゴリズムの理解といった複数要素が組合わされているため、解答できなかった学生が多く見られたと考えられる。

問題 8-2（図 4）は関数を用いて 2 つの変数を入れ替える問題で、穴抜き 4 は関数呼び出しの引数に該当する。穴抜き 4 における学生の答案を表 3 に示す。呼び出す関数ではポインタが使われており、アドレス渡しをしなければならない。アドレスとポインタの違いが理解できるか、関数の定義と呼び出しの違い、仮引数と実引数の違い、引数の受け渡しを理解しているかが問われる。この穴抜きでは表 3 の 6 行目に該当するポインタの表現方法が理解できていない解答を C1 として分類し、その他、8 行目のポインタの理解が不足している解答としてアドレスとポインタの違いが理解できていない解答を C1.1 に分類した。この穴抜きでは関数呼び出しとポインタ処理に伴うアドレス渡しという複数要素が重なったため誤答が多く出現したと考えられる。

4.2 誤答分類表

ログデータに対して分類ルールを適用することで表 4 のような誤答分類を特定することができた。この分類表を見ることで、学生が苦手としている問題と穴抜きの性質が分かり、教育時に重点的に実施すべき箇所が分かった。

ステップ	プログラム
	# include < stdio.h >
	//2つの変数の値を入れ替える関数
	void swap (int [1] , int [2]) {
1	int tmp ; //一時保存用
2	tmp = *x ;
3	*x = *y ;
4	*y = tmp ;
5	return ;
	}
	int main ( ) {
1	int a = 5 , b = 10 ; //変数の定義と初期化
	//aとbの表示
2	printf ("a=%d__b=%d\n" , a , b) ;
	//aとbを入れ替える
3	[3] ([4] , [5]) ;
	//aとbの表示
4	printf ("a=%d__b=%d\n" , a , b) ;
5	return 0 ;
	}

図 4：問題 8-2

分類 B で最も多く見られた問題は 7-2 で、19 個であった。この問題は整数と実数の累乗を計算する関数を呼び出す問題である。プログラムのみに穴抜きが存在し、抽象的なコメントとトレース表から適切な関数を呼び出せるかが問われている。また、最も多かった穴抜きは問題 11-3 の 2 番に該当し、関数呼び出しの引数に当たる穴抜きであった。B では関数の呼び出しに関する問題とその穴抜きが多いことが分かり、特にその引数についてどのように解答するのかわからないが何か解答は残すといった傾向がみられた。

C が最も多い問題は 8-2 で 2 つの変数を入れ替える関数とその呼び出しについて問われている。プログラムとトレース表の両方に穴抜きがある問題に該当し、最も多い穴抜きは問題 8-2 の 2 番にあたる部分であった。穴抜きは関数定義に当たる部分で、答案では直前に「int」があるにも関わらず 2 重に定義しているもとが多く見られた。

E では問題 6-1 が最も多く、82 個見られた。穴抜きも同様の問題の 1 と 2 番で 14 個ずつ見られた。また、分類 E の特に E2 に当たる誤答が最も多く見られた。このようなことから、プログラムとトレース表の問題ともにポインタに関する間違いが多いことが分かった。

表 3：問題 8-2 穴番号 4 の答案

答案	誤答理由	人数	分類
	空欄	8	A
&a	正答	28	D1
*a	アドレスとポインタの違いが分かっていない。	1	C1.1
*x	アドレスとポインタの違いが分かっていない。仮引数と実引数の違いが分かっていない。	1	C1.1, C4
A	ポインタの表現方法が理解できていない。	12	C1
a*	ポインタの表現方法が理解できていない。	1	C1
int a	ポインタの表現方法が理解できていない。	4	C1
int x	仮引数と実引数の違いが理解できていない。	1	C4
X	仮引数と実引数の違いが分かっていない。ポインタの表現方法が理解できていない。	3	C1,C4

5. 未解答率と正解率の関係

図 4 に問題毎の未解答率と正解率の分布を示す。横軸は正解率、縦軸は未解答率を示している。未解答率と正解率の相関係数は-0.91 と高くなっており、正解になりやすい問題であるほど未解答の答案が少なる傾向にあることが分かった。これは簡単な問題ほど何らかの解答を入力しやすく、一方、難しいと判断した問題は解答を諦める傾向が強くなるためだと考えられる。

また、問題の分類毎に解答を集計した結果、トレース表の問題、プログラムの問題、プログラムとトレース表が混在する問題の順に空欄が増える傾向にあることが分かった。これは過去の我々の論文[6]でも言及した問題の分類と正解率の関係と一致する結果であり、より複雑な問題構成になるにつれ未解答率が増える傾向にある。

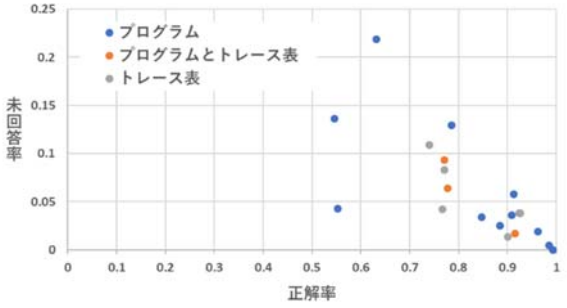


図 4：問題毎の正解率と未解答率

穴抜き毎の正解率と未解答率の相関関係についても調べたところ、こちらも正解率が増えると未解答率も減る傾向にあった。また、正解率が 80～100%の未解答率が最も高くなっており、狭い区間に集中していることが分かった。これはトレース表問題の性質が原因ではないかと考えられる。

表 4：誤答分類表

区分	分類	解答数	説明
A		389	空欄（指示が理解できない，または，解答する意思がない）
B			コメントで示された指示が理解できていない
	B1	25	具体的な指示がコメントに書かれており，その指示が理解できれば正解できる.
	B2	89	コメントに書かれた指示が抽象的であり，自力で具体化しなければ正解できない.
C			指示は理解できているが，具体的な表現方法が誤っている（ないし分からない）
	C1	72.5	ポインタの概念が理解できていない解答（表現方法が理解できていない）
	C1.1	30.5	アドレスとポインタの区別がついていない解答
	C1.2	13.5	ポインタの参照方法が理解できていない。
	C2	24.5	配列に関する間違い（例：ポインタを使うべき箇所で配列を使った）
	C3	56	関数の定義と呼び出しに関する間違い
	C4	9	仮引数と実引数に関する間違い
	C5	1	単純な for 文または二重 for 文に関する間違い
	C6	27	変数の定義域に関する間違い
	C7	14	変数の定義に関する間違い
	C8	43	誤字、計算間違い、何らかの認識違いと思われる解答
	C9	4	値の更新をする式としない式の区別がついていない解答
D			正答
	D1	8605	正答
	D2	16	題意にそぐわない正答
E			トレース表を正しく追跡できていない解答
	E1	13	インクリメントによるデータ更新を理解できていない解答
	E2	73.5	ポインタの概念が理解できていない解答
	E2.1	13	ポインタの参照先の理解不足
	E3	83	関数の性質理解ができていない解答
	E4	20	ルーチンの役割の理解とその範囲が理解できていない解答
	E5	25	トレース表の記入方法を理解していない解答
	E6	12	引数の穴抜きにおける誤答

また､プログラムとトレース表が混在する問題でも正解率が高く、未解答率が低くなっているものがあった。これは問題の作成時に学生に pgtracer の仕組みを理解させることを目的としたテスト問題を含むためだと考えられる。

最も正解率の低い穴抜きは問題 11-1 の 3 番目の穴抜きであった。この問題では適切なヘッダーファイルを include 指定して読み込む必要があり、さらに処理に該当する関数を呼び出す必要がある。そのどちらもファイル名、関数名の指定部分に穴抜きが設定されており、指示されたヘッダーファイルの理解と関数呼び出しの複数要素の理解が必要となる穴抜きであった。

6. 問題毎のクラスタ分析

6.1 分析データ

分類したデータでは、1 つのレコードに問題の区分毎に、穴番号、誤答の種類毎の人数のデータを保持している。総数は 647 個であり、このデータを穴抜き、分類毎に集計することで分析データを作成した｡分析データの行は問題に、列は誤答の種類に対応しており、各要素は問題と誤答の種類に対応する答案数を保持する。

6.2 分析方法

階層的クラスタ分析では初めにクラスタ数を決めずにクラスタリングを行う手法である。そのため、分析後のデンドログラムからいくつのクラスタに分けるか決めなければならない。クラスタ数の決定には明確な決まりがなく、分析者にとって良い解釈が得られる段階を採用する。

解答データのクラスタ分析にはウォード法を採用した[6]。最短距離法や最長距離法などいくつかの分析方法を試したところウォード法が最も均等にクラスタが形成されていた。他の分析方法よりも計算量は多いがデータ間の違いが分かりやすいという利点があり、分析データの数がそれほど多くないことからも問題なく実行できた。

6.3 分析結果と考察

分析データの行は問題、列は誤答の種類に対応しており、各要素は問題と誤答の種類に対応する答案数を保持する（表 5)。なお、複数の原因がある場合，それらの間で按分したため、小数点以下の値が発生している。問題番号の右には穴抜きの個所（T：トレース表のみ、P：プログラムの

表 5：問題毎の誤答数（分類毎）

	問題		A	B1	B2	C1	C1.1	C1.2	C9	C2	C3		C4	C5	C6	C7	C8	D2	E1	E2	E2.1	E3	E4	E5	E6	正解率
G1	4-1	T	9														7		13							98 %
	5-1	T															3			2						99 %
	6-1	T	23																	69	13					89 %
	7-3	T	11		2												10			2		57				90 %
	8-2	B	12			21.5	12.5				3		9							0.5				2.5		92 %
G2	3-3	P	8		5					2				1												96 %
	4-2	P	26		8					5								4								91 %
	5-2	P	20			12.5	4								0.5	2										93 %
	5-3	P	15	4		8.5	4	5									1.5									90 %
	8-1	P	16														0					3			12	92 %
G3	6-2	P	76		16	21				11.5							3.5									63 %
	6-3	P	50	12		4	2	1	4		1					9										79 %
	7-2	P	32		19						15					1	12	7								77 %
	8-3	P	18		14	4	5	5.5			2				13.5		1									78 %
	11-3	P	25		16		3			6	2				13	2	2									77 %
	11-1	B	36		3			2			22											23				75 %
G4	7-1	P	4	9		1					2						2	3								85 %
	11-2	B	8		6						9						1	2					20			77 %
	13-2	P	2		4									9.5	5.5											55 %
	14-1	P	3	1							3.5					2.5										55 %

み、B：トレース表とプログラムの両方）を示す。R を用いたクラスタ分析の結果、4 つのクラスタ G1~G4 が形成された。

クラスタ G1 では問題 8-2 を除き全ての問題がトレース表のみに穴抜きがある問題であった。また、穴抜きの数が 7~11 と比較的多くかつ正解率が 89～99%の問題の内、ほぼトレース表からなるものから構成されていた。G2 は穴抜きが 5～6 個で、正解率が 90～96%の問題から構成されていた。G3 は穴抜きの数が 5～8 個であり、正解率は 63～79%の問題から構成されていた。G4 は穴抜きの数が 5～8 個で正解率が 55～85%の問題から構成されていた。G2 と G3 では１つを除きプログラムの問題から構成されていた。G4 ではプログラムと混合問題の内正解率の低い問題から構成されていた。4 つのクラスタでは正解率と問題の種類がクラスタの形成に大きく影響を与えていると考えられる。

これらの結果から教員は問題を作成するに有益なデータであると考え得られる。学生の能力を評価するテストとして出題する場合、異なるクラスタから問題を構成することで多様な解答を促すことができ、学生の能力を向上させる練習問題として出題する場合、一つのクラスタから問題を選択することで似た傾向の問題を重点的に提供できると考えられる。

7. 誤答種類毎のアソシエーション分析

7.1 分析方法

収集した学習データについて、誤答分類表を作成した。具体的には A～E からなる大分類 5 個と詳細な誤答理由を挙げ 24 個の分類をした。その分類をもとに、誤答分類間をアソシエーション分析することで関連して表れやすい誤答やその要因を相関ルールとして抽出し考察する。

アソシエーション分析[6]は POS（Point Of Sales:販売時点情報管理）等で用いられている分析方法であり、蓄積されたデータの中から有益な情報を見つけ出すことができる。商品の販売時の情報を蓄積し、商品間の関連性と規則性があるものについて、apriori アルゴリズムを用いて相関ルールを抽出する。蓄積しているデータをトランザクションデータと呼び、分析対象となるデータセットを構成する。データセットからは「商品 A を買うと商品 B も買う」事実相関ルールとして抽出し、「{X}⇒{Y}」と表記する。

ルールの評価基準は支持度（support）、確信度（confidence）、リフト（lift）の 3 つである。支持度はデータセットの中で X と Y がデータセット全体に占める比率であり、支持度が高いとは、データセットの中で Y に属する分類は X という組み合わせが多い傾向にある。確信度は X と Y を含むデータセットの数を、X を含むデータセットの数で割った値を表しており、確信度が高いとは X という要素の組み合わせを持つ分類は Y に属することが多い傾向にある。リフトは確信度を支持度で割った値であり、リフト値が高いということは条件 X の時事象 Y が起こりやすい。

7.2 相関ルールの抽出

分析に用いるデータセットはこの 3 つのテーブルより、2018 年度導入時点での学習履歴 ID、答案、正誤判定結果、blank_path を抽出した。また、抽出した答案について誤答分類表との対応付けを行い、学習履歴 ID ごとに出現する誤答の種類を特定した。

以上のデータより分析では学習履歴 ID（学生 ID と問題 ID の組）と答案の誤答分類結果を１つのトランザクションとしたデータセットを作成し、相関ルールを抽出した。

分析に用いたデータセットは 2284 レコードであった。最も出現していた誤答分類として A が 1560 回確認された。他の誤答分類では 30 個前後が確認されているが、A とはかなりの差があることが分かった。そこで全体の出現頻度を考慮し、支持度を 0.03、最小確信度は 0.55 として相関ルールの抽出を行った。

7.3 分析結果と考察

表 6 では上記の支持度と確信度に基づいた相関ルールの一覧を示している。分析の結果では支持度順に 8 つの相関ルールを抽出することができた。

相関ルール{}⇒{A}は空欄の出現数が他の分類よりも突出して多いことが原因であると考えられ、{D1}⇒{C8}は正解と誤字による間違いの分類であるため、関連性が高くみられるのは自明であり、教育上のメリットは低いと思われる。また、分類 A ではデータセットの 7 割で出現することが分かっており、それらの相関ルールを除き最も支持度が高く表れた誤答分類の組み合わせは{E6}⇒{E3}であった。165 回の組み合わせの内、E3 が 94%の確率で表れており、リフト値も高くなっている。また、{E6}と{E3}の条件部と結論部を入れ替えた合わせも同様の結果が言えることから前後関係によらず分類の関連性が強いといえる。この組み合わせでは関数の性質理解に関する誤答と引数の理解に関する誤答であることがわかっている。{C5}⇒{C6}では 30 個の組み合わせで見られ確信度は 100%であった。この組み合わせでは for 文に関する誤答と変数の定義に関する誤答であることが分かっている。

これらの相関ルールは学生が問題解答時に同時に間違い易い組み合わせを特定しており、教育上有益であると考えられる。

表 6：誤答分類結果の相関ルールの確信度上位 8 件

条件部⇒結論部	支持度	確信度	リフト値
{}⇒{A}	0.68 (1560)	68%	1.00
{E6}⇒{E3}	0.07 (165)	94%	9.70
{E3}⇒{E6}	0.07 (165)	74%	9.70
{D1}⇒{C8}	0.02 (42)	87%	10.74
{C5}⇒{C6}	0.01 (30)	100%	30.05
{C6,D1}⇒{C8}	0.004 (10)	100%	12.28
{B2,C6}⇒{C8}	0.004 (8)	57%	7.02
{B2,C8}⇒{C6}	0.004 (8)	57%	17.17

8. おわりに

本研究ではプログラミング教育支援ツール pgtracer を用いて収集した学習データについて解答の分類と分析を行った。解答データについて分類ルールを設け、原因と思われる理由を列挙することで分類表を作成した。その結果よくある間違いを特定した。また、問題毎のクラスタ分析では問題作成時にテスト問題や練習問題の目的に応じた種類の問題を作成できるクラスタを特定できた。誤答分類毎にクラスタ分析では特定の穴抜きでしか出現しない誤答や同時に表れやすい誤答の組み合わせが分かり、教育時に重点的に実施する箇所を特定した。

今後の課題として、今回の分析では正解率の低い穴抜きのデータを意図的に扱っていない。そのため今後分析データとして加えることで分析の精度が増すことが期待される。また、本稿で提案した改善事項について実際の講義で評価実験を行うことを検討している。

謝辞　本研究は JSPS 科研費（課題番号 17K01036）の支援を受けています。また、データ収集にご協力頂いた熊本高等専門学校の村田 美友紀 准教授、並びに学生の皆様に感謝いたします。

参考文献

[1] 掛下哲郎，柳田峻，太田康介，“穴埋め問題を用いたプログラミング教育支援ツール pgtracer の開発と評価”，情報処理学会論文誌：教育とコンピュータ, Vol. 2, No. 2, pp. 20-36, 2016 年 10 月.

[2] Tetsuro Kakeshita, Miyuki Murata, “Application of Programming Education Support Tool pgtracer for Homework Assignment”, *International Journal of Learning Technologies and Learning Environments*, Vol. 1, No. 1, pp. 41-60, 2018 年 3 月.

[3] Tetsuro Kakeshita, Kosuke Ohta, “Student log analysis functions for web-based programming education support tool pgtracer”, *IPSJ Trans. on Computers and Education*, Vol. 5, No. 2, pp. 1-13, 2019 年 6 月.

[4] 中山陽平，掛下哲郎，“プログラミング穴埋め問題における穴抜きの難易度と学生の解答過程のクラスタ分析”，情報処理学会 情報教育シンポジウム SSS2018, pp. 166 – 173，2018 年 8 月.

[5] 村田 美友紀，嘉藤 直子，掛下 哲郎，“プログラミング学習支援ツール pgtracer を自学習に活用したプログラミング科目の実践報告”，情報処理学会 情報教育シンポジウム SSS2018, pp. 150-157，2018 年 8 月.

[6] 金明哲，“R によるデータサイエンス（データ解析の基礎から 最新の手法まで）”，森北出版，2017.