Vol.2021-CE-162 No.28
Vol.2021-CLE-35 No.28
2021/12/5

解答の状態遷移の閉路の分析による ジグソー・コードにおける試行錯誤の検出

山口 琢[1]　松澤 芳昭[2]　新美 礼彦[3]　大場 みち子[3]

概要：新しい教育では試行錯誤を肯定的に重要視し，その内容にも注目する．本稿では，取捨選択・並べ替え型のプログラミング・パズル「ジグソー・コード」における学習者の試行錯誤を機械的に検出するために，約 230 人が解いた 12 問について，解答枠内の並び順の状態遷移における閉路 (cycle) の有用性を検証する．まず，閉路の終点 (すなわち始点) は解答者にとって「ここまでは正しい」とみなした状態であると考えられる．そこで，閉路の終点が正解の部分列になっているか評価したところ，出現頻度の多い閉路 26 個のうち 25 個 (96%) の終点の並び順が正解の部分列になっていた．また，終点が正解の部分列になっている率が大きい問題は正答率が高くなる傾向があった．閉路の終点は，解答者が「そこまでは正しい」とみなしていたと考えられる．次に，閉路の始点直後の操作対象と閉路脱出直後の操作対象とのペアは互いに代替になっていると考えられる．そこで，これらのペアを抽出して，作問者が意図的に含ませた選択肢との適合性を評価したところ，79%で適合した．これらのペアは実際に解答者が取捨選択したものと考えられる．これら 2 つの結果より，解答枠内の並び順の状態遷移における閉路を分析することは，試行錯誤を評価するために有用と考えられる．

キーワード：試行錯誤，状態遷移，閉路，取捨選択，並べ替えプログラミング・パズル，学習分析

Detecting Trial-and-error in Solving Jigsaw Code by Analyzing Cycles in Solution Transitions

Taku Yamaguchi[1]　Yoshiaki Matsuzawa[2]　Ayahiko Niimi[3]　Michiko Oba[3]

1. はじめに

近年，教育では考え方／思考力といったプロセスが重要視され，試行錯誤を肯定的に評価し，かつ評価にあたって試行錯誤に種類を想定している．中央教育審議会は「人間は，…多様な文脈が複雑に入り交じった環境の中でも，…自ら目的を設定し、…答えのない課題に対して、…目的に応じた納得解を見いだしたりすることができる．」と指摘して，「…解き方があらかじめ定まった問題を効率的に解いたり，定められた手続を効率的にこなしたりすることにとどまらず，…自分なりに試行錯誤したり…」することが重要としている [1](p.10)．国立教育政策研究所の教育課程研究センターは学習評価を解説する事例 [2](pp.54-59) で，数式のパラメータを決めるときに「適当な数を入力していき調整する」ようなものを「見通しを持たない試行錯誤」とし，「目安を求めてから調整する」ようなものを「見通しを持った試行錯誤」としている．そのうえで，見通しのない試行錯誤を行った場合に「おおむね満足できる」，見通しのある試行錯誤を行った場合に「十分満足できる」と解説している．試行錯誤に種類があることと，いずれも肯定的に評価していることが特徴である．

そこで，これからの学習分析 (Learning Analytics) 研究は，学習者が問題をどのように解決しようとしているのか，

[1] 公立はこだて未来大学システム情報科学研究科
Graduate School of Future University Hakodate
[2] 青山学院大学 社会情報学部 社会情報学科
School of Social Informatics, Aoyama Gakuin University
[3] 公立はこだて未来大学システム情報科学部
Faculty of Systems Information Science, Future University Hakodate

Vol.2021-CE-162 No.28
Vol.2021-CLE-35 No.28
2021/12/5

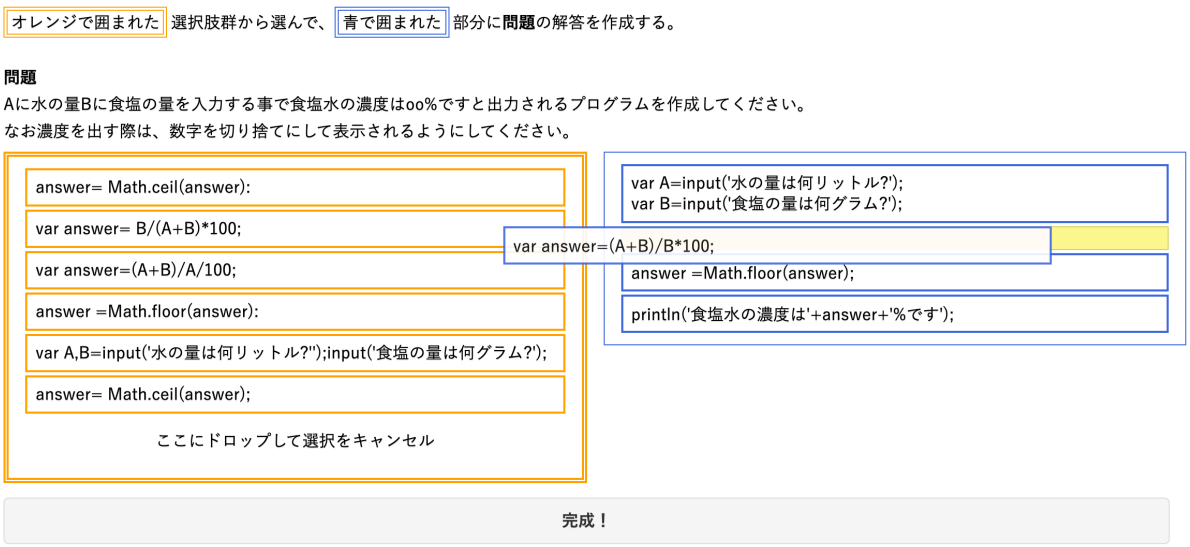

図 1 取捨選択・並べ替えプログラミング・パズルのジグソー・コードの問題「1-2 濃度判定プログラム．ドラッグ&ドロップで取捨選択・並べ替える．操作対象となるコードのかたまりをピースと呼ぶ．

Fig. 1 A select-and-discard and rearranging puzzle Jigsaw Code 2 and the puzzle “1-2 Calculate salinity concentration.”

試行錯誤を肯定的に捉えて具体的な内容を検出することが求められるだろう．

本研究では試行錯誤の一種として取捨選択に注目する．プログラミング・パズルの「ジグソー・コード」を題材にして，学生の取捨選択操作を具体的に検出して，学生の取り組みや作問を評価するのに役立つようなデータ分析手法の開発に取り組む．

本稿ではまず，2 章でプログラミング・パズルを解くプロセスの表現方法や分析手法の先行研究を検討する．その中で，本研究の題材であるジグソー・コードも紹介する．3 章で本研究の目的を定義する．4 章では提案する解答プロセス表現と分析手法，そして提案手法の検証方法を述べる．5 章で結果と検討，6 章で考察と結論，7 章でまとめる．

2. 先行研究

プログラミング教育では，0 からコードを書かせるのではなく，シャッフルされたプログラムの断片群から取捨選択・並べ替えて完成させるタイプの教材が使われることがある．本章で取り上げる Parsons Problem[3][6][12] や，ジグソー・コード [8] や，短冊型プログラミング問題 [7] が，そのようなタイプの教材である．本稿では，このような教材をプログラミング・パズルと呼ぶことにする．

プログラミング・パズルを解くプロセスの研究では，プロセスの表現を工夫する．これは，アプリケーションがどのようなデータを記録しているかにも依存するが，研究者が試行錯誤をどのように捉えているかにも影響される．

2.1 非生産的で誤解の表れとしての試行錯誤

Kumar[3] は，学生が Parsons puzzles を解く戦略の表現として，プログラムの行 (line) を，それらが正しい位置に置かれた順番に並べたもの (solution sequence) を提案している．Parsons puzzle (または Parsons problem) は，後で述べるジグソー・コードや短冊形プログラミング問題と同様のプログラミング・パズルである．「プログラムの行」とは，ジグソー・コードで言えば「ピース」であり，短冊形プログラミング問題で言えば「短冊」である．Solution sequence をジグソー・コード流に例えると，学生が s3，s5，s4，s2，s1，s5，s2，s4 の順にピースを動かして，その結果最終的にそれらが正しい位置に置かれたとき，solution sequence は [s3, s1, s5, s2, s4] となる．つまり，s4 が 3 番めに動かされたことなどは表現されない．Kumar は，後戻り (backtracking) や堂々巡り (loop) といった試行錯誤 (trial-and-erro activities) を非生産的 (unproductive) で誤解 (misconception) に基づくものだと考えている．試行錯誤を問題解決の戦略 (strategy) の一部とは考えていない．

Kumar が試行錯誤をそのように捉えるのは，Parsons puzzle をプログラミング言語の構文を丸暗記 (rote learning) するためのアプリケーションとみなしているからかもしれない．後で述べる短冊形プログラミング問題はそうではない．久野 [4] は，短冊形プログラミング問題を使って思考力・判断力・表現力を評価する手法を提案している．

2.2 プログラミング・パズルとジグソー・コード

ジグソー・コードは，プログラムをいくつかのピース (断片) に分割し，さらに不要なピースも混ぜてシャッフルし

Vol.2021-CE-162 No.28
Vol.2021-CLE-35 No.28
2021/12/5

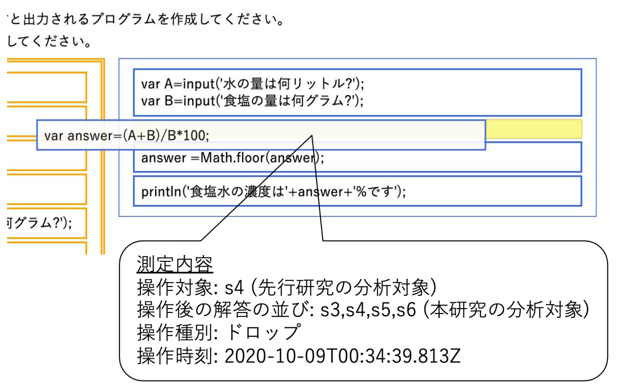

図 2 ジグソー・コードの測定内容: s4 などはピースの ID．操作対象や操作後の解答の並びを記録している．

たものから，取捨選択・並べ替えて正しいプログラムを完成させるジグソー・パズルである．われわれが開発・運用して授業や実験で使っている [8].

図 1 は，取捨選択型のジグソー・コード 2 の画面である．左側のオレンジで囲まれた選択肢枠内の最初の並び順は，解答を始めるたびにシャッフルされている．右側の枠の上下には，あらかじめ固定ピースが配置されていることもある．解答者は，左側の枠から，適切なピースを選んで，右側の青で囲まれた解答枠にドラッグ&ドロップで移動する．後で不要と分かったピースは，右から左にドラッグ&ドロップで戻す．左右どちらの枠内でも，ピースをドラッグ&ドロップで並べ替えることができる．これらの操作を繰り返して，ピースを右側の解答枠内に適切な順序で並べてプログラムを完成させる [9][10][11]．解答を完成させると document id と呼んでいる ID が表示される．Document id は問題提示から解答完成までのプロセスごとにユニークに発行される．

ジグソー・コードは，ユーザの並べ替え操作を測定している．左の枠から右の枠へピースを移動して選ぶ，右の枠から左の枠へピースを移動して捨てる，それぞれの枠内でピースを移動して並べ替える，これらすべての操作について，操作対象のピース ID や操作後の解答枠内のピースの並びや操作時刻などを document id と共に記録している．

2.3 ジグソー・コード操作の取捨選択操作の共起分析

取捨選択を検出するために，われわれは，左の枠から右の枠へピースを取る操作と，右の枠から左の枠へピースを捨てる操作，すなわち取捨選択操作を集計する共起行列を提案した [5]．2 つの取捨選択操作の間に，右の枠内での並べ替え操作が入る場合があるが，そのような操作があったとしても 2 つの操作を共起としてカウントする．

図 3 で，まず，s7 を取り，続いて s6 を枠内で上に移動し，s4 を捨てたとする．2.3 節の共起分析では s7 と s6，s6 と s4 が共起するが，s7 と s4 は共起とみなさない．提案手法では，s7 と s4 が共起したとみなし，s7 と s6 および s6 と s4 の共起はカウントしない．

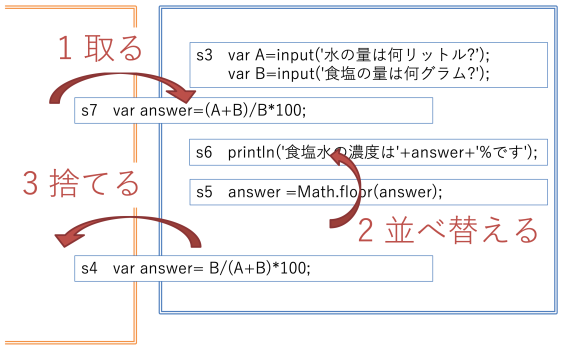

図 3 取捨選択操作の集計
Fig. 3 Count the number of select and discard operations

n \ n+1	s3	s4	s5	s6	s7	s8	s9	s10	s11	s12
s3	0	2	1	2	2	4	0	35	0	2
s4	-	3	1	0	16	2	0	1	8	3
s5	-	-	2	2	2	7	3	3	0	20
s6	-	-	-	0	1	0	1	2	1	1
s7	-	-	-	-	3	0	0	2	5	2
s8	-	-	-	-	-	3	0	0	1	2
s9	-	-	-	-	-	-	0	0	0	0
s10	-	-	-	-	-	-	-	3	0	1
s11	-	-	-	-	-	-	-	-	2	0
s12	-	-	-	-	-	-	-	-	-	1

図 4 取捨選択の手順的な共起行列
Fig. 4 Procedual co-occurrence matrix of select-and-discard operations

図 4 は図 1 のパズル (問題) を 223 人の学生が解いたときの，操作ログを集計した取捨選択の共起行列である．どちらを先に選んだか順序を問わないので，共起行列の右上に集計されている．

対角線上のセルは，ある同じ 1 つのピースを取った後に捨てた，あるいは捨てた後に取り直した回数である．これは，「ある 1 つのピースを選ぶか選ばないか考えた」などと解釈されることになるだろう．

セルの値が平均 + 標準偏差より大きいセルを黄色 (薄い背景色)，平均 + 標準偏差 ×2 より大きいセルを赤 (濃い背景色に白い文字) で強調している．図 4 では，s3 の次に s10 を動かした回数が「35」回で，他に比べて多いことが分かる．

2.4 状態遷移としての解答の時間変化の分析

上野ら [7] は，短冊型プログラミング問題に解答するプロセスを解答の状態遷移で可視化することで，解答者がつまづいたポイントを発見しやすくするシステムを提案し評価した．ここで解答とは，最終的に提出された解答だけで

Vol.2021-CE-162 No.28
Vol.2021-CLE-35 No.28
2021/12/5

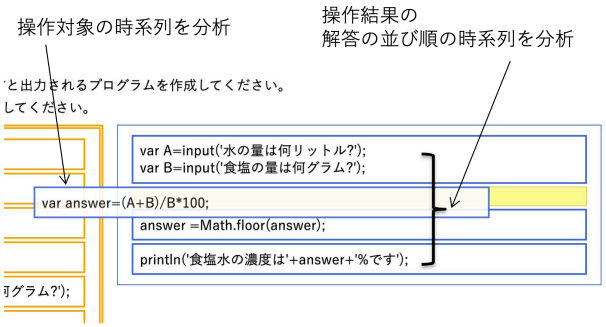

図 5 分析対象の違い: 先行研究には操作対象の時間変化を分析する手法と，操作によって変化する解答の並び順の時間変化を分析する手法がある．

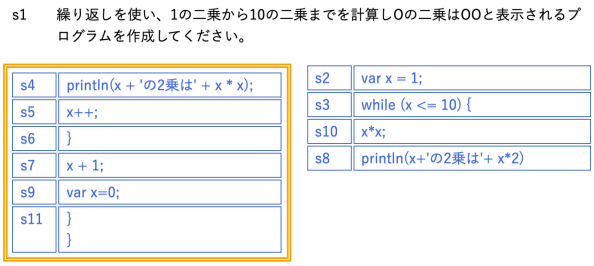

図 6 問題「3-2 二乗を計算」の解答プロセスの途中．右側の解答欄におけるピースの並び”s2,s3,s10,s8”が，この時点の状態

Fig. 6 In the middle of the solution process. The sequence of pieces ”s2,s3,s10,s8” is the state at the time.

なく，そこにいたる途中の状態も含む．解答の状態は短冊の並び順で表現される．状態遷移とは短冊の並び順の変化である．前に述べた 2.3 節の手法は操作対象のピースを分析するのに対して，上野らの手法は操作によって変化する解答の並びを分析する (**図 5**).

解答の並び順を分析する上野らの分析手法は，前節 2.3 節で述べた操作対象を分析する手法と同程度，またはより詳細に解答プロセスを分析できると期待できる．まず，状態が記録されるタイミングが操作の都度となっている点が同じである．また，解答の状態は短冊の粒度で記述されいて，解答に含まれる短冊の数が同じでも短冊が異なれば異なる状態だし，同じ短冊を含んでいても並び順が異なれば異なる状態である．この粒度は，どのピース (短冊) を操作したかと同程度またはより詳細なものである．

そこで，解答の並び順の時間変化を分析する手法を使っても，2.3 節で述べた手法と同様に試行錯誤を検出できると期待できる．

3. 目的

本研究の目的は，取捨選択・並べ替え型のプログラミング・パズル「ジグソー・コード」を題材にして，学習者の試行錯誤を機械的に検出するために，解答枠内の並び順の状態遷移における閉路 (cycle) の有用性を検証することである．有用性とは，閉路を分析することで次の 2 つを推定できることである:

- 閉路の始点=終点は試行錯誤の起点であり，解答者が「ここまでは正しい」と考えた状態であると推定できる．
- 閉路の始点直後の操作対象ピースと閉路を脱出した直後の操作対象ピースとのペアは，解答者の試行錯誤における選択肢であると推定できる．

閉路などの用語は 4 章「研究方法」で定義する．

本稿における「取捨選択」とは，パズルを解くプロセスにおいて，特定の複数のピースを「どれかが正しい可能性のある」選択肢 (2 択，3 択，…) と捉えて，そこからピースを取り，あるいは取ったピースを捨てて，最終的にピースの選択を確定させる一連の操作を指す．測定・分析による検出の内容は，解答者がどのピースを選択肢と捉え，どれを・どのように選択したかである．

4. 研究方法

本稿では，解答の状態遷移における閉路を分析して，それに基づいて解答者の試行錯誤を検出する手法を提案する．本章では，提案方式を 2 つのステップで述べ，続いて提案方式の検証方法を述べる．

4.1 提案方式 ステップ 1: 解答の状態遷移の閉路の分析

閉路の分析では次の 2 つを調べる:

- 閉路の終点 (=始点) を抽出する．
- 閉路の始点直後の操作対象ピースと閉路を脱出した直後の操作対象ピースとのペアを抽出する．

以下に順を追って説明する．

解答の状態とは，問題を解くプロセスにおける，解答欄内のピースの並び順である．これをピース ID の並びで表現する．本稿では「解答」を最終的に提出された解答だけでなく，そこにいたる途中の状態も指す．**図 6** は問題「3-2 二乗を計算」の解答プロセスの途中である．”s2”などはピースの ID で通常は画面表示されない．画面右側の解答欄におけるピースの並び”s2,s3,s10,s8”が，この時点での解答の状態である．

解答の状態遷移とは，解答プロセスの進展に対応した，解答の状態の変化である．ソフトウェアの設計などで使われる状態遷移図は状態遷移の可能性を表現するものだが，本稿の状態遷移は実際に起きた状態の遷移を表す．**図 7** は状態遷移の 3 つの例である．エッジに添えた数字は，そのエッジをたどった回数である．右の状態遷移では，”s2,s3”から”s2,s3,s10”へ 2 回遷移している．グラフは d3-graphviz[*1](と D3.js) を使ってブラウザ上に描いたものである．

[*1] GitHub - magjac/d3-graphviz: Graphviz DOT rendering and animated transitions using D3
https://github.com/magjac/d3-graphviz

Vol.2021-CE-162 No.28
Vol.2021-CLE-35 No.28
2021/12/5

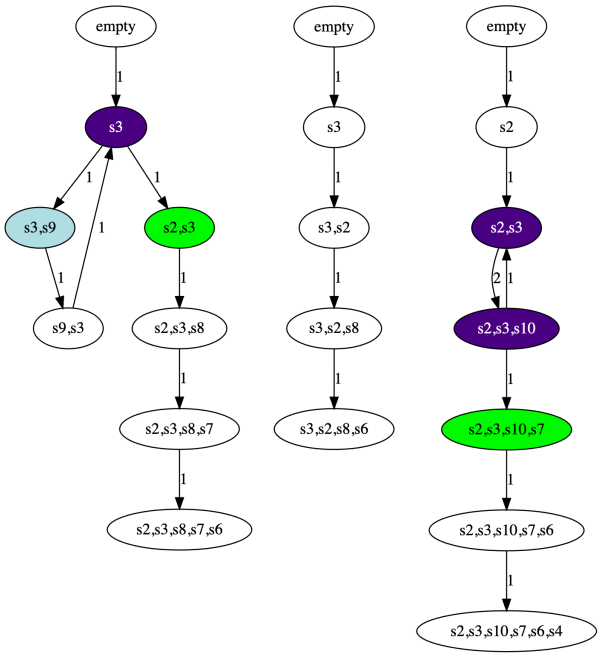

図 7 状態遷移の例．左は閉路を 1 つ含む．真ん中は閉路を含まない．右は閉路を 2 つ含む．

Fig. 7 Examples of state transition. The left includes one cycle. The middle includes no cycles. The right includes two cycles.

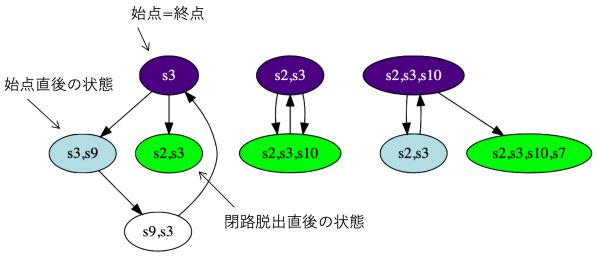

図 8 閉路の始点=終点，始点直後の状態，閉路脱出直後の状態

Fig. 8 Start and endpoint of cycle, the first state after the start, the first state after get out of the cycle.

閉路とは，解答の状態遷移において，同じ状態が複数回現れたとき，その状態を始点と終点に持ち，途中には含まないような状態遷移の部分である．図 7 の左側は閉路を 1 つ含む．真ん中は閉路を含まない．右は 2 つ含む．これ以降，状態遷移の図において，藍色の背景に白い文字のノードは閉路の始点=終点，水色の背景は始点直後の状態，緑の背景は閉路から脱した直後の状態を表す．始点=終点や始点直後や閉路から脱した直後の状態は互いに重なることがあり，必ずしも 3 つに色分けられない．**図 8** は図 7 から閉路を抜き出したものである．左の閉路が図 7 の左の状態遷移図から，真ん中と右は図 7 の右の状態遷移図から抜き出した 2 つの閉路である．

閉路の始点直後の操作対象ピースとは，始点と始点直後の状態との差分のピースである．図 8 左の状態遷移図では，”s9”が始点直後の操作対象である．図 8 の右の閉路の

s1　繰り返しを使い、1の二乗から10の二乗までを計算し0の二乗は00と表示されるプログラムを作成してください。

s7	x + 1;
s8	println(x+'の2乗は'+ x*2)
s9	var x=0;
s10	x*x;
s11	} }

s2	var x = 1;
s3	while (x <= 10) {
s4	println(x + 'の2乗は' + x * x);
s5	x++;
s6	}

図 9 問題「3-2 二乗を計算」の正解．”s2,s3,s4,s5,s6”が正解の並び

Fig. 9 The correct answer for ”3-2 Calcurate the square”

ように，始点直後の操作がピースを削除する操作の場合もある．右の閉路では”s10”が始点直後の操作対象である．

閉路を脱出した直後の操作対象ピースとは，終点 (すなわち始点) と閉路を脱出した直後の状態との差分のピースである．図 7 左の状態遷移図では，”s2”が脱出直後の操作対象である．閉路の始点=終点が最終的な解答の場合，脱出直後の状態はない．

4.2 提案方式 ステップ 2: 試行錯誤の推定

前節のように閉路を分析したうえで，次のように推定することを提案する:

- 閉路の終点は，解答者が「ここまでは正しい」と考えた状態である．
- 閉路の始点直後の操作対象ピースと閉路を脱出した直後の操作対象ピースとのペアは，解答者の試行錯誤における選択肢である．

このような推定が妥当であることは，次のように検証する．

4.3 検証方法: 解答者にとって「ここまでは正しい」

閉路の終点となった状態が解答者にとって「ここまでは正しい」ものであったことを，それら状態が正解の部分列かどうかによって検証する．

正解の部分列とは，残りのピースの順序を変えないように，正解からいくつかのピースを削除することによって得られるピース列である．**図 9** は「3-2 二乗を計算」の正解で，正解の並びは”s2,s3,s4,s5,s6”となる．”s2,s4,s6”は正解の部分列だが，図 6 の”s2,s3,s10,s8”は正解の部分列ではない．

もちろん，解答者が間違っている場合，解答者が「ここまでは正しい」と考えた状態が正解の部分列とは限らない．そこで検証内容は次の 2 つである:

- 多く出現する閉路終点は正解の部分列になっている．
  「多く出現する」は出現頻度が平均 + 標準偏差 ×2 よりも大きいものとする．
  みんなが「ここまでは正しい」とみなすならば，実際に正しい可能性が高いと考えるのである．
- 終点全体において正解の部分列になっているものの割

Vol.2021-CE-162 No.28
Vol.2021-CLE-35 No.28
2021/12/5

合は，問題の正答率と正に相関する．

『当人は「ここまでは正しい」とみなしたが実際には間違った状態』が多く出現するならば，正答率が低くなると考えるのである．

これら 2 つが満たされれば，閉路の終点は解答者にとって「ここまでは正しい」ものであったと推定してよいと考えられる．

4.4 検証方法: 試行錯誤の選択肢

閉路の始点直後と閉路を脱出した直後との操作対象ピースのペアが，解答者の試行錯誤における選択肢であったことを，それらが問題を作った先生の作問意図と適合するかどうかで検証する．提案方式が「実際に起きた」として抽出したペアのうち出現頻度が多いものが，作問者にとって納得できるものか (正確性) を評価するのである．

提案方式が抽出するペアは，まず閉路の始点直後の操作対象ピースと閉路を脱出した直後の操作対象ピースとのペアの出現頻度を集計し，出現頻度が平均 + 標準偏差 ×2 よりも大きいものを選ぶ．

先生の出題意図は，意図的に含ませた「よく似た間違えやすい複数の選択肢」を，ピースのペアのリストとしてデータ化する．

そして，作問意図のペアのリストを正解として，提案方式が検出したペアの適合率 (precision) を評価する．適合率は，提案手法が検出した操作の中に，作問者の意図に適合した操作がどれだけ含まれているかという正確性を示す．再現率 (recall) は，作問者の出題意図に適合するペアのうちで，どれだけのペアを提案手法が検出したかという網羅性を示すが，そもそも解答者が出題意図通りに取捨選択しない (「ひっかからない」) 場合があるので，この検証では重要視しない．

また，実際に起きた取捨選択のペア，すなわち True Positive と False Negative の合計が，可能なペア (全組み合わせ) の数に対して小さいことを確認する．これが大きいと，当てずっぽうで検出しても適合する可能性が大きくなる．

4.5 検証データの収集

提案方式の検証に使うため，提案方式を適用する「問題を解くプロセスの測定データ」を用意する．測定データは文献 [5] で分析したのと同じデータを使う．

測定の対象となったのは，大学の文理融合系学部の 1 年生を対象とした必修授業「コンピューティング実習」で，授業内の小テストでジグソー・コードを使い，学生の操作を測定した．受講生は約 230 人．授業は，JavaScript, HTML やタートルのライブラリを使って変数，条件分岐，繰り返し，関数など，プログラミングの初歩を学ぶものである．授業はオンラインで行われた．

小テストの作問にあたっては，学習のポイントを考えさせるように意図して，不要な選択肢ピースを混ぜる工夫をして学生に取捨選択させた．作問意図は，取捨選択が起きるであろうピースのペアのリストとしてデータ表現する．小テストの内容は，図形を描くためにタートルを動かす順序，塩分濃度の計算，Go To トラベル割引料金の計算の仕方などを考えさせるものである．1 回の授業で 2〜3 問，合計 13 問を，新たに開発して出題した．

小テストの実施については，まず初回の小テストの前に，実験の概要，ジグソー・コードの操作方法および document id の提出方法を説明した．小テストは授業の最初に，前回までの内容を理解しているかの演習として実施した．解答後に，問題の解説を提示した．

表 1 結果: 頻度の多い閉路の終点のうち，正解の部分列になってるものの比率．多頻度終点は頻度の多い閉路終点の数．部分列の数は正解の部分列になっている閉路終点の数，比率は正解の部分列になっている比率．

問題	多頻度終点	部分列の数	比率
1-1 砂時計	3	3	1.00
1-2 濃度判定	2	2	1.00
1-3 整数入れ替え	4	3	0.75
2-1 PCR	4	4	1.00
2-2 ひまわり	2	2	1.00
3-2 二乗を計算	1	1	1.00
3-3 九九出力	2	2	1.00
4-1 ドーナツを描く	2	2	1.00
4-2 閏年判定	1	1	1.00
5-1 GO TO Travel 1	1	1	1.00
5-2 GOTO Travel 2	4	4	1.00
合計と平均比率	26	25	0.96

5. 結果

5.1 閉路の終点が正解の部分列か

出現頻度の多い閉路 26 個のうち 25 個 (96%) の終点の並び順が正解の部分列になっていた (**表 1**)．

また，全ての閉路について集計すると，終点が正解の部分列になっている率が大きい問題は，正答率が高くなる傾向があった (**表 2**，**図 10**)．図 10 のピアソンの積率相関係数は 0.86 である．

図 11 は正答率が最も低い問題「3-2 二乗を計算」の，閉路の出現回数の頻度分布である．出現回数が最も多い 9 回出現した閉路は正解の部分列になっているが，全体の中では少数派なことが分かる．**図 12** は正答率が最も高い問題「4-2 閏年判定」の，閉路の出現回数の頻度分布である．出現回数が最も多い 11 回出現した閉路は正解の部分列になっている．他の閉路に比べて出現回数は多いものの，図 11 ほど極端に多いわけではない．どちらも約 230 人が解いたものだが，縦軸の目盛りが 1 桁違うことに注目して欲しい．問題「3-2 二乗を計算」は，閉路の種類が多いことが図 11 から分かり，その 63%が正解の部分列ではなく間違ってい

Vol.2021-CE-162 No.28
Vol.2021-CLE-35 No.28
2021/12/5

表 2 結果: 全ての閉路終点において，正解の部分列になっているものの比率と，問題の正答率．閉路終点は閉路終点の総数．正解の部分列は，正解の部分列になってる閉路終点の数．比率は閉路終点が正解の部分列になっている比率．

問題	閉路終点	正解部分列	比率	正答率%
1-1 砂時計	160	110	0.69	57.6
1-2 濃度判定	61	39	0.64	51.5
1-3 整数入れ替え	142	53	0.37	9.7
2-1 PCR	107	68	0.64	36.2
2-2 ひまわり	28	12	0.43	37.5
3-2 二乗を計算	68	25	0.37	8.3
3-3 九九出力	86	52	0.60	63.9
4-1 ドーナツを描く	115	46	0.40	30
4-2 閏年判定	48	36	0.75	69.3
5-1 GO TO Travel 1	61	39	0.64	50.2
5-2 GOTO Travel 2	108	48	0.44	40.6
合計と平均	984	528	0.54	41.1

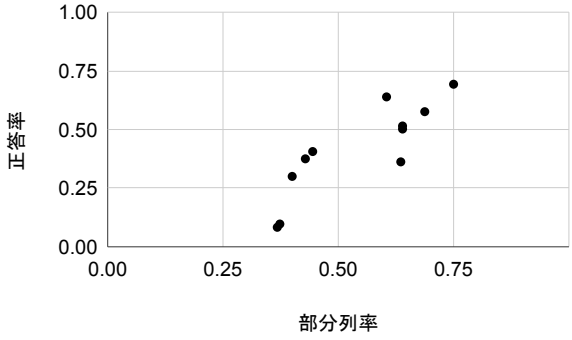

図 10 全ての閉路終点において正解の部分列になっているものの比率 (横軸) と，その問題の正答率 (縦軸)．ピアソンの積率相関係数は 0.86.

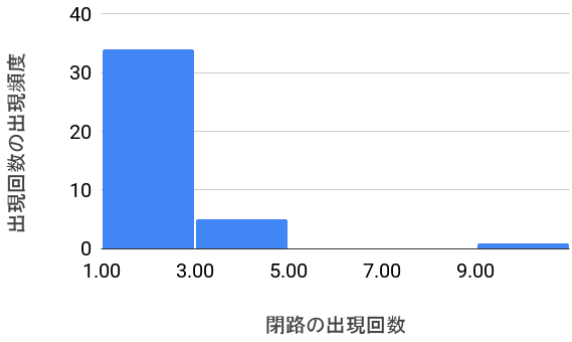

図 11 正答率が低い問題「3-2 二乗を計算」における閉路の出現回数の頻度分布

ることが表 2 の「比率」列の 0.37 から分かる．

5.2 試行錯誤の選択肢か

抽出したペアと作問者が意図的に含ませた選択肢との適合性を評価したところ，79%で適合した．これはまずまずの結果と言えるだろう．また，全組合せ数に対する「実際に取捨選択されたペア (TP+FN)」の比率は 6%であった．6%であれば当てずっぽうで当たるとは言えない．(**表 3**, **表 4**)．ここで全組合せ数は:

全組合せ数 $= {}_nC_2 + n, n$: 動かせるピース数 (表中のピ数列)

TN は全組合せ数から TP，FP，FN の合計を引いたものである:

TN = 全組合せ数 - TP - FP - FN

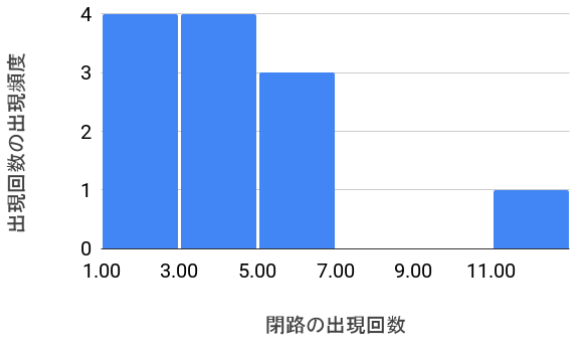

図 12 正答率が高い問題「4-2 閏年判定」における閉路の出現回数の頻度分布

表 3 結果 3: 閉路から抽出した選択肢と出題意図．ピ数は，その問題において動かせるピースの数．TP は True Positive，FP は False Positive，FN は False Negative，TN は True Negative．比率は (TP + FN)/組合せ数．ここで組合数は，ピ数のピースから 2 つ取り出す組合せの数 + ピ数．

問題	ピ数	TP	FP	FN	TN	比率
1-1 砂時計	11	1	3	1	61	0.03
1-2 濃度判定	10	1	0	5	49	0.11
1-3 整数入れ替え	12	3	0	3	72	0.08
2-1 PCR	13	2	2	0	87	0.02
2-2 ひまわり	9	3	0	0	42	0.07
3-2 二乗を計算	10	1	0	2	52	0.05
3-3 九九出力	7	2	0	0	26	0.07
4-1 ドーナツを描く	9	1	0	1	43	0.04
4-2 閏年判定	7	1	0	1	26	0.07
5-1 GO TO Travel 1	7	2	0	0	26	0.07
5-2 GO TO Travel 2	12	2	0	2	74	0.05

表 4 結果 4: 適合率と，全組合せ数に対する TP+FN の比率

適合率	79%
再現率	56%
全組合せ数に対する TP+FN の比率	6%

6. 考察

6.1 閉路の意義，それを検出できる意義

「ここまでは正しい」と認識する考え方は役に立つ．「ここまでは正しい」と判断して，さらに進めた後で間違ったと思ったら「ここ」に戻ること，そして戻れるようにしておくことは，モノの考え方，作業の進め方として有用と言

Vol.2021-CE-162 No.28
Vol.2021-CLE-35 No.28
2021/12/5

えるだろう．そのような解き方を実践していると検出できる点で，提案手法は有用である．実際，プログラムを開発・修正するときは，段階的に機能を実装しながら，そのたびにコミットして戻れるようにする．山歩きで迷ったら正しいルートに戻るべきで，そのためには自分が正しい位置にいるかを要所要所で確認する．閉路の存在はこのような考え方ができていることの表れであることを本稿は検証した．

もちろん，閉路がないからといって，そのような考え方をしなかった，あるいはそのような考え方ができないとは言えない．

6.2 閉路判定の厳密さと閉路の数

閉路を含む解答プロセスはせいぜい半分であった．本稿の閉路判定では，ピースの並び順序も同じであることを要求した．この条件を緩めて，並び順が異なってもよいことにすれば，閉路の数が増えるかもしれない．すなわち，ピースの並びではなく，ピースの集合として状態を定義する．あるいは，ピースの個数だけを見て，個々のピースが何であるかは問わないことにすれば，さらに閉路が多くなるだろう．その一方で，そのように閉路の条件を緩めると，それら閉路の終点を「ここまでは正しい」と解答者がみなした可能性は小さくなるのではないか．これらの検討は今後の課題とする．

6.3 閉路の終点そのものの分析と評価

本稿では閉路の数を集計したが，閉路そのものは評価しなかった．閉路の終点は「ここまでは正しい」と解答者がみなしたものだと主張したが，では，それは正解のどのような部分列であろうか．解答者はどのような部分列から試行錯誤を始めるのだろうか？また，閉路の終点には，単に「正解の部分列である」以外に評価ポイントはあるだろうか？それによって解答者を評価できるであろうか？この検討は今後の課題とする．

6.4 閉路の長さと思考能力

本稿では閉路の長さを考慮しなかった．閉路が長ければ長いほど，「ここまでは正しい」と見なした状態から遠くへ探索してると言える．言い換えると，「これは間違っている」と気づいた状態から「ここまでは正しい」と見なした状態まで長い道のり (エッジ数) を正確に戻ったことになる．より遠くへ探索して正確に戻ってこれるということは，より明確に「ここまでは正しい」と意識して問題を解いていると言えるかもしれない (**図 13**)．閉路の長さは，解答者の何らかの考え方や思考力の高さとして評価できるかもしれない．この検討は今後の課題とする．

6.5 作問の工夫

「ここまでは正しい」という考え方ができるかどうかを意図的に問う問題とはどのような問題だろうか？また，こういう考え方ができるかどうかを，従来の思考力を試す問題のように，解答の正誤から判定できるだろうか？これも今後の課題となるかもしれない．

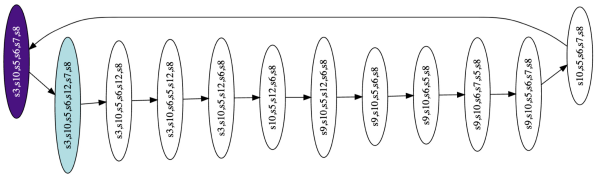

図 13 長い閉路の例．”s3,s10,s5,s6,s7,s8”が最終的な解答であった．
Fig. 13 Long cycle example. ”S3,s10,s5,s6,s7,s8” is the final solution.

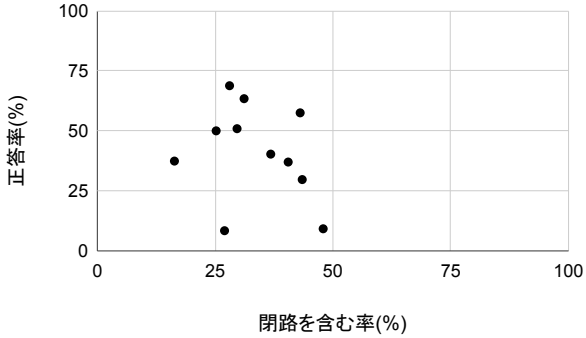

図 14 状態遷移が閉路を含む比率と正答率の関係．ピアソンの積率相関係数は-0.26.

6.6 アンドゥ機能

アプリ開発者であれば，本稿を読んで「アンドゥ (undo) 機能が必要だ」と考えるかもしれない．本稿が問題にしたのは，アンドゥ機能の有無ではなく，アンドゥ機能の使いこなしであったと言えるだろう．

6.7 閉路の有無と正解/不正解

閉路の数だけで問題が難しかった／簡単だったと推定するのは難しい．**図 14** は解答プロセス (状態遷移) が閉路を含む比率と問題の正答率との関係を示す散布図で，ピアソンの積率相関係数は-0.26 である．**図 15** は全閉路数を解答プロセスの数 (すなわち解答者の人数) で割ったものと正答率との関係を示す散布図で，ピアソンの積率相関係数は-0.19 であった．どちらも，正答率との関係は図 10 ほど強くない．また，閉路を含む解答プロセスはせいぜい半分であることが分かる．

6.8 結論

閉路の終点は，解答者が「そこまでは正しい」とみなしていたと考えられる．また，閉路の始点直後と閉路を脱出した直後との操作対象ピースのペアは，実際に解答者が取捨選択したものと考えられる．これら 2 つの結果より，解

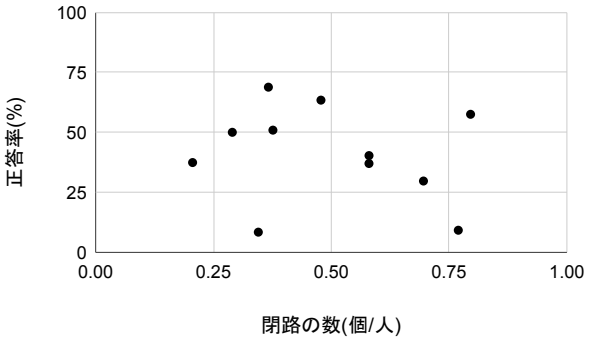

図 15 解答プロセスあたりの閉路の数と正答率の関係．ピアソンの積率相関係数は-0.19.

答枠内の並び順の状態遷移における閉路を分析することは，試行錯誤を評価するために有用と考えられる．

7. まとめ

閉路の終点 (=始点) は解答者にとって「ここまでは正しい」とみなした状態であることを検証できた．出現頻度の多い閉路 26 個のうち 25 個 (96%) の終点の並び順が正解の部分列になっていた．また，終点が正解の部分列になっている率が大きい問題は正答率が高くなる傾向があった．また，閉路の始点直後の操作対象と閉路脱出直後の操作対象とのペアは互いに代替になっていることを検証できた．これらのペアと作問者が意図的に含ませた選択肢との適合性を評価したところ，79%で適合した．これら 2 つの結果より，解答枠内の並び順の状態遷移における閉路を分析することは，試行錯誤を評価するために有用と考えられる．

従来は，取捨操作を分析して解答者が何を選択肢として捉え何を選択したかを具体的に検出できた．本研究は，解答プロセスの別の表現として，解答の並び順の状態遷移における閉路を分析することで同様に取捨選択を検出できたことに新規性がある．加えて，同時に解答者が「ここまでは正しい」という考え方をしてることを検出できた点でも新規性がある．また，学習者の考え方を重視し試行錯誤の内容を肯定的かつ具体的に評価する新しい教育において，本研究は学習者が思考の過程で設定する「ここまでは正しい」ポイントやその後の取捨選択を検出できるものであり，この点で有用性を確認できた．

単なる閉路の有無は正答率にあまり関係しなかった．今後，「正解の部分列である」以外の閉路終点の評価ポイントや閉路の長さなど閉路の分析・評価が進めば，新しい教育が試行錯誤を肯定的に重要視し，試行錯誤の仕方を問うことの妥当性を裏付けるものとなるだろう．

謝辞 本研究は JSPS 科研費 20H01728 の助成を受けたものである．

参考文献

[1] 中央教育審議会：幼稚園、小学校、中学校、高等学校及び特別支援学校の学習指導要領等の改善及び必要な方策等について (答申)(中教審第 197 号)，文部科学省（オンライン），入手先 <https://www.mext.go.jp/b_menu/shingi/chukyo/chukyo0/toushin/1380731.htm¿〉（参照 2021-09-13）

[2] 国立教育政策研究所 教育課程研究センター：「指導と評価の一体化」のための学習評価に関する参考資料 (高等学校編) 情報，（オンライン），入手先〈https://www.nier.go.jp/kaihatsu/shidousiryou.html〉(参照 2021-09-13)．

[3] Kumar, A.: Representing and Evaluating Strategies for Solving Parsons Puzzles, Proc. *15th International Conference*(*ITS 2019*) , pp.193-203 (online), DOI: 10.1007/978-3-030-22244-4_24, (2019).

[4] 久野 靖：思考力・判断力・表現力を評価する試験問題の作成手順，情報処理学会 情報教育シンポジウム論文集，Vol.2018, No.1, pp.1-8, 入手先〈http://id.nii.ac.jp/1001/00190680/〉(2018)．

[5] 山口 琢，松澤 芳昭，新美 礼彦，大場 みち子：取捨選択操作の時間的な共起分析によるプログラミング・プロセスでの迷いの検出，情報処理学会研究報告コンピュータと教育 (CE)，Vol.2021-CE-160，No.8，pp.1-6（オンライン），入手先〈http://id.nii.ac.jp/1001/00211364/〉(2021)．

[6] Du, Y., Luxton-Reilly, A., Denny, P.: A Review of Research on Parsons Problems, Proc. {ACE'20: Twenty-Second Australasian Computing Education Conference}, pp.195–202 (online), DOI: 10.1145/3373165.3373187, (2020).

[7] 上野 真，照井 佑季，今村 瑠一郎，久野 靖，江木 啓訓：短冊型プログラミング問題における解答プロセスに基づく指導支援，情報処理学会 マルチメディア，分散協調とモバイルシンポジウム 2021 論文集，Vol.2021，No.1，pp.1451-1458（オンライン），入手先〈http://id.nii.ac.jp/1001/00212996/〉(2021)

[8] 山口 琢，大場 みち子：できごと、手順、プログラムや地理の並べ替え操作の測定と分析，情報処理学会 情報教育シンポジウム論文集，Vol.2018，No.25，pp.179-184（オンライン），入手先〈http://id.nii.ac.jp/1001/00190704/〉(2018)．

[9] Yamaguchi, T., Oba, M.: Measurable Interactive Application to Find Out User Recognition and Strategy when Problem Solving, (JSW), Vol. 15, No. 1, pp.12-22 (online), DOI: 10.17706/jsw.15.1.12-22 (2020).

[10] 中村 陽太，大場 みち子，山口 琢，伊藤 恵：学習進度に対応するパズルを利用したプログラミング思考過程の分析，情報処理学会研究報告コンピュータと教育 (CE)，Vol.2019-CE-151，No.1，pp.1-9（オンライン），入手先〈http://id.nii.ac.jp/1001/00199559/〉(2019)．

[11] 中村 陽太，大場 みち子，山口 琢，伊藤 恵：授業進度に対応するパズルを利用したプログラミング思考過程の分析と教育支援システムの開発，情報処理学会 第 82 回全国大会講演論文集，Vol.2020，No.1，pp.701-702（オンライン），入手先〈http://id.nii.ac.jp/1001/00205909/〉(2020)．

[12] Maharjan, S. and Kumar, A.: Using Edit Distance Trails to Analyze Path Solutions of Parsons Puzzles, Proc. *Thirteenth International Conference on Educational Data Mining* (*EDM 2020*) , (online), available from 〈https://educationaldatamining.org/files/conferences/EDM2020/papers/paper_163.pdf 〉 (2020).