「マルチメディア，分散，協調とモバイル
(DICOMO2021) シンポジウム」 令和3年6月-7月

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

上野 真[1]　照井 佑季[1]　今村 瑠一郎[1]　久野 靖[1]　江木 啓訓[1]

概要：本研究は，初学者を対象とした短冊型プログラミング問題の解答プロセスを，状態遷移図を用いて可視化する．学生の解答プロセスを適切な形で可視化することで，教員や TA の指導の支援につなげることを目的とする．実際の試験のデータに基づいて，Ruby 言語の問題について分析を行った．正答者のみ，誤答者のみ，誤答者のつまずき箇所発見の 3 つの状態遷移図を描画して分析した．分析の結果，プログラムの外側から組み立てて考えることが指導のポイントにつながる可能性が明らかになった．授業の TA を担当した経験のある被験者を対象に，状態遷移図が指導のために役立てられるかを確認する実験を行った．評価実験の結果，現状の状態遷移図は情報は十分であり，解答プロセスを分析する手段として状態遷移図がふさわしいことがわかった．一方で情報量が多すぎることから，傾向やつまずきポイントが判断しにくい結果となった．これらのことから，学生の解答プロセスを適切な形で可視化することで，教員や TA の指導の支援につながる可能性が示唆された．

Teaching Support Based on the Answering Process for Split-Paper Testing

SHIN UENO[1]　YUKI TERUI[1]　RYUICHIRO IMAMURA[1]　YASUSHI KUNO[1]　HIRONORI EGI[1]

1. はじめに

プログラミング学習において，多くの初学者が困難を抱えているという問題がある．McCracken らは，多くの学生がプログラミング入門コースの終了時に，プログラミングの書き方を習得していない可能性があることを指摘している [1]．このことについて，Lister らは問題解決能力の欠如が原因であると考えた [2]．そこで，学生に対してテストを行い，問題解決のために必要なスキルが脆弱であることを明らかにしている．また，Gomes らはプログラミングに対して苦手な学生が少なくないことを課題として挙げ，その原因はアルゴリズムの理解が不十分であると指摘している [3]．

一般的にプログラミング学習においては，抽象的な概念を正しく理解する必要がある [4]．また，多くの学生がプログラミング学習上の問題を抱えているが，全ての学生に対し有益であるように指導を設計することは困難である．しかし，丁寧に設計された教材とアプローチにより，教員は学生の知識とスキルの構築を指導することが可能になる．

短冊型プログラミング問題は，採点の容易なプログラミング能力試験を作る意図で開発された完全自動採点の問題である [5]．この問題では，正解のプログラムが行単位で分割され選択肢となる．さらに，誤答の選択肢と混ぜた上で，選択記号が割り振られ，問題の選択肢となる．学生は出題された問題に対して表示されている選択肢を解答フォームにドラッグ＆ドロップし，プログラムを組み立てて完成させる．学生が問題を解答する際の選択記号の履歴は，Web のシステムから学生個人のデータとして取得することが可能である．

短冊型プログラミング問題を用いたプログラミング学習アドバイスツールが開発されている [6]．久野は，プログラミング問題における基本的な誤りを採点前に指摘する目的で，ツールを作成した．実際の授業で使用した結果，ツールを使用した学生が使用してない学生に比べ，誤りが有意

[1] 電気通信大学大学院 情報理工学研究科 情報学専攻
Graduate School of Informatics and Engineering, The University of Electro-Communications

に少なくなっていることを明らかにしている．

2. 関連研究

2.1 パーソンズ問題

短冊型プログラミング問題と同様の形式をしたパーソンズ問題の研究が進められている．Parsons 問題には，さまざまなバリエーションがある [7].

一般的なプログラミング試験の練習は，コードトレース (読み取り) とコードライティングの 2 種類の問題を中心に行われる．これらとパーソンズ問題を比較する研究が行われている．Denny らは，インタビューと定量的な試験問題の比較を通して，パーソンズ問題の得点とコードライティング問題の得点の間に相関があることを明らかにしている [8]．また，Lister らはコードトレース問題のスコアと Parson の問題の間に相関関係があることを発見している [9]．これにより，パーソンズ問題は従来のコードライティング問題に代わる優れた問題であると述べている．

Helminen らはフィードバックシステムを開発し，状態遷移図を用いてパーソンズ問題の解法を分析している [10]．この研究で分析対象とした問題は，従来のパーソンズ問題にインデントの操作を加えたものである．フィードバックシステムは行数が少ないこと，順序が誤っていること，インデントが誤っていることを知らせるものである．学生の解法を分析した結果，解法パスにばらつきがあること，短冊コードを先に並べた後にインデントをまとめて揃えることを明らかにした．その分析結果は，自動フィードバックシステムの改善につなげられると述べている．

Maharjan らは，パーソンズ問題の解法パスの距離に注目した分析手法を提案した．この研究ではレーベンシュタイン距離を修正した編集距離に基づいて分析を行い，学生は解答の最後の数段階で困難に陥る傾向があることを明らかにしている [11].

2.2 ジグソー・コード

大場らは，短冊型プログラミング問題，パーソンズ問題と同様の問題形式である，ジグソー・コードの研究を行っている．操作の時間的な経過を分析して操作のパターンを検出するために，操作の時間的な共起行列を用いて，プログラミング思考課程を分析している [12]．問題作成時の狙いに対して，授業の演習問題で出題されなかった構文は理解が薄いことや，性質を理解しないと解けない問題では，正解率が低くなることを明らかにしている．

パーソンズ問題において，maharjan らは並べ替え操作における最適な解答プロセスでは，正解との距離が単調に減少することを前提としている．大場らはジグソー・コードを題材に，並べ替え操作をした前後の正解との編集距離の変化を調べ，単調減少の前提の妥当性を分析している [13]．分析の結果，「各局面で適切な問題解決プロセスで，正解との距離が単調減少するという前提を，無条件でおくのは妥当でない」と述べている．

3. アプローチ

第 1 章で述べたプログラミング学習における基礎的な課題を解決するためには，教員や TA が初学者に適した指導をする必要がある．また，第 2 章で述べたように，短冊型プログラミング問題の解答プロセスの分析に関する研究は，様々なアプローチで行われている．しかし，これらは自動フィードバックシステム等の作成が主たる目的であり，教員や TA らの指導の直接的な支援を目的として分析している研究ではない．

そこで本研究では，教員や TA らの指導の支援を最終的な目的として状態遷移図を用いて分析を行う．学生の解答プロセスを，教員や TA らが理解可能な形で可視化することができれば，指導の支援に繋がり苦手な学生の数が減少することが期待できると考えた．状態遷移図から取得できる情報を，教員や TA にどのように可視化して理解してもらうかを判断するために，経験のある TA に対して評価実験を行う．

4. 分析

4.1 分析対象

分析対象は，理工系大学 1 年生の必修科目であるプログラミング授業で出題された問題とする．1 クラス約 60 人が受講しており，11 クラス全体で 682 人が分析対象である．5 つの問題セットがあり，問題セット 1 は 4 クラス分の 241 人，問題セット 2，3，4 は 2 クラス分のそれぞれ 125 人，126 人，128 人，問題セット 5 は 1 クラス分の 62 人が解答した．問題は問題セットごとに異なるが，問われる分野は同じであり，14 題で構成される．実際の試験で使用された問題は Ruby 言語が 10 題，C 言語が 4 題であるが，本研究では Ruby 言語のみ分析する．本研究で分析する Ruby 言語の問題内容を表 1 に示す．

分析する問題は，全て第 1 章で説明している短冊型プログラミング問題である．図 1 に学生の解答画面を示す．問題は右画面に選択する領域，左画面にコード領域がそれぞれ存在する．右画面の選択する領域では，正解のプログラムを 1 行ずつ分割し，誤答の選択肢と合わせて選択肢が存在する．学生はその選択領域から選択肢をドラッグ＆ドロップで左画面のコード領域へ移動させる．また，コード領域での解答は自動でインデントされる．

4.2 分析手法

学生が選択肢を追加したり，削除したり，順番を変更させた履歴が解答の列として取得できる．このログは学生ごとに，コードエリアで行った操作ごとの時間や解答列が表示され，CSV ファイルとして出力される．この CSV ファ

図 1　学生の解答画面

表 1　Ruby 言語の問題内容

設問番号	内容
問題 1	基本的な四則演算
問題 2	分岐
問題 3	配列生成
問題 4	手続きと再帰
問題 5	画像の点の設定と書き出し
問題 6	二次元配列の生成
問題 7	配列生成と書き換え
問題 8	モンテカルロ法を用いたメソッドの定義
問題 9	クラス定義
問題 10	単連結リストの生成

イルは，学生一人の解答につき一件のデータとして格納する．問題内容と正答データは XML ファイルで出力される．この 2 つのファイルを Python に取り込み，Graphviz を用いて状態遷移図を描画する．本研究では，以下の 3 タイプの状態遷移図を作成する．その際，以下の 3 つの分析する目的によって出力する情報を変更する．

(1) 正答者のみの状態遷移図
(2) 誤答者のみの状態遷移図
(3) 誤答者のつまずき箇所発見のための状態遷移図

これらの状態遷移図は，該当する全ての学生の解答を統合して作成する．

状態遷移図はノード，パスによって構成され，PDF，PNG ファイルのいずれかで出力される．ノードは，初期状態（空欄）から最終状態（提出する解答）までの 1 つ 1 つの選択履歴を表す．例えば，選択肢 A を選択し，次に選択肢 B を選択した場合，A というノードから AB というノードにパスが引かれる．start と書かれた開始ノードと正解ノードは，濃い灰色で表示している．パスの線の太さは，ノードから次のノードに遷移した人数に比例して太くする．線の重みは線の太さによって表現する．パスのラベルは，遷移した人数を表す．

誤答者のつまずきポイント発見のための状態遷移図では，ノードの大きさを変更する．ノードのサイズは，ノードの半径の大きさを変更して表現する．このノードをつまずきノードと定義する．対象ノード（つまずきノード）から次のノードに遷移する人のうち，正答者と誤答者の両方が存在する対象ノード（つまずきノード）の大きさを変更する．その大きさは，問題の正答率とノードに存在する誤答者の積に比例している．もし問題の誤答者数のみに比例させて描画すると，可視化した状態遷移図のつまずきノードのみが過大となり，グラフに偏りが生じてしまうことが考えられる．また，正答率が低い問題は誤答者数が多く，つまずきノードが大きくなり，同様にしてグラフに偏りが生じてしまうことが考えられる．これらの 2 つの理由から，問題の正答率を掛けて比例させた．つまずきノードは他のノードと区別をするために，薄い灰色で表示している．

また，パスの色を正答者の人数によって 10 段階で表す．ノードからノードへ遷移する全体の人数のうち，正答者の人数の割合を百分率によって表す．図 2 に正答者の割合を表すカラーバーを示す．正答者の割合が高い方を青い色，低い方を赤い色によって表現する．パスのラベルは遷移した合計の人数（そのうちの正答者の人数）で表す．

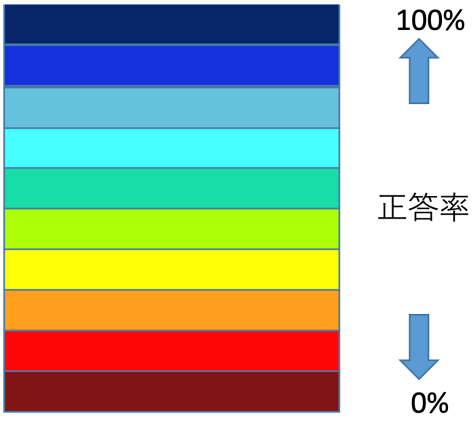

図 2　正答者の割合を表すカラーバー

4.3　分析結果・考察

試験セットのうち試験セット 5 の問題 1〜10 について，正答者のみの状態遷移図，誤答者のみの状態遷移図，誤答者のつまずきポイント発見のための状態遷移図を作成した．本論文には，その中の問題 5 について 3 つの図を示し，具体的な分析結果を示す．また，正答者のみの状態遷移図，誤答者のみの状態遷移図のうち，特徴的な選択をしたと考えられる学生数の内訳を表 1 に示す．さらに，5 つの試験セットにあまり相違がないことを確かめるために，問題セット 1〜5 の各問題における正答率を比較した．最

後に，問題 1〜10 を分析した全体的な結果と考察を示す．

4.3.1 問題 5 の 3 つの図の結果と考察

図 3 に問題 5 の正答者のみの状態遷移図を示す．この問題では，選択肢 A が def，選択肢 C が Array.new() であるから，正答者のうち 18 人（86 %）がはじめに def, Array.new() を選択していたことがわかる．この問題は二次元配列を作成する問題であるため，メソッドを定義すること，配列を作成することが容易に想像できたためだと考えられる．また，ノード ACEFH を選択するのではなく，ノード ACEFG，ACEFJ に遷移した人が 4 人（8 %）いたことがわかる．選択肢 HGJ は if 文の条件が異なる選択肢であり，迷っていたことが考えられる．

図 4 に問題 5 の誤答者のみの状態遷移図を示す．26 人 26 人（63 %）がノード ACEF に遷移していることがわかる．そして，ノード ACEFG，ACEFI，ACEFJ に遷移している人が合計 7 人（17 %）いることがわかる．正答は ACEFHPBBBDB であり，選択肢 G，H，I，J はいずれも if 文である．したがって，if 分の条件の中身を間違えたことが原因で誤答していると考えられる．また，ACEFHPBBDB を最終解答にしている人が 4 人（10 %）いた．これは選択肢 B の end が 1 つ足りないことが原因で誤答している．ACEFHPBBBB を最終解答にしている人が 3 人（7 %）いた．これは，選択肢 D の return が足りていないことが原因で誤答している．

図 5 に問題の誤答者のつまずき箇所発見のための状態遷移図を示す．この状態遷移図では，つまずきノードから出ている赤いパスが他のつまずきノードに繋がっていることがわかる．ACEF が一番大きいつまずきノードであり，このノードから出ている赤いパスを辿ると，ACEFJ，ACEFP に遷移していることがわかる．選択肢 J，P はいずれも if 文であり，正答で使用する選択肢 H と異なる条件を選択していることがわかる．AC は次に大きいつまずきノードであるが，このノードから出ている赤いパスを辿ると，ノード ACG，ACN に遷移していることがわかる．選択肢 G，N とも if 文であり，不要な if 文を使用していることがわかる．つまずきノードである ACE から出ている赤いパスを辿ると，ノード ACEG，ACGE に遷移していることがわかる．選択肢 G は不要な if 文であり，これを選択していることが誤答の原因となっている．

4.3.2 特徴的な選択をした学生

問題 1〜10 の中の正答者のみの状態遷移図と誤答者のみの状態遷移図から，特徴的な選択をしたと考えられる学生数の内訳を表 2 および表 3 にそれぞれ示す．d は，解答のはじめに，def を解答した学生を表現している．順番は，解答を正答の選択肢の順番通りに解答した学生を表現している．例えば，ABC が正答であった場合，A → AB → ABC

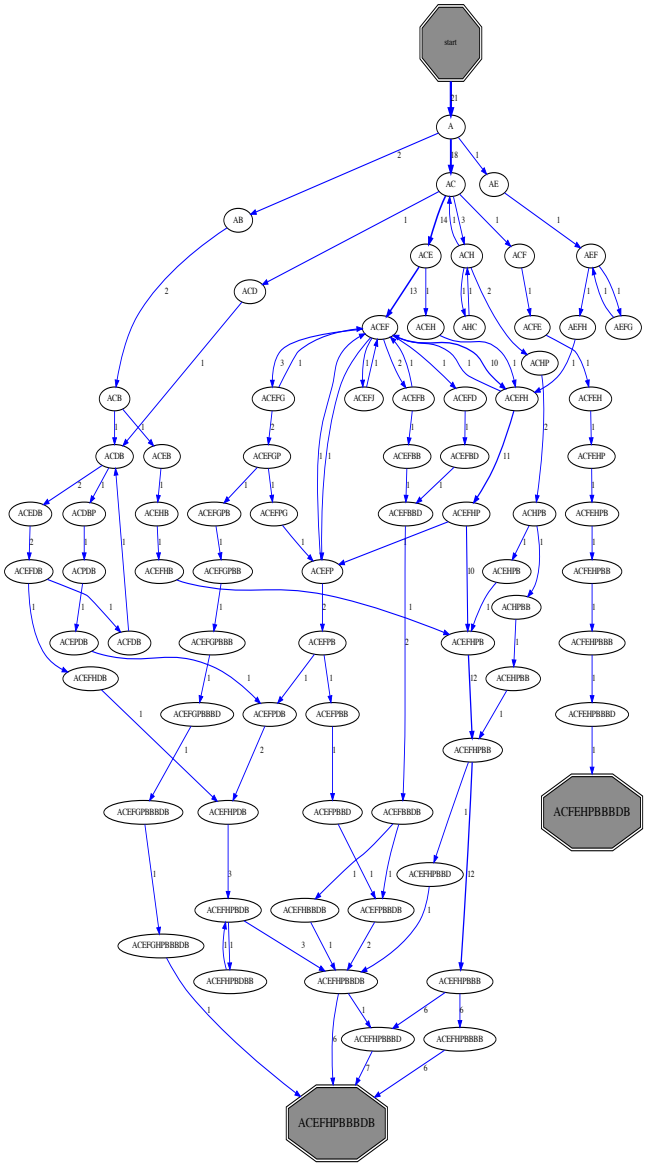

図 3 問題 5 の正答者のみの状態遷移図

と解答した学生を順番とカウントする．d-e は，解答のはじめに，def と end を解答した学生を表現している.def と end の間に他の選択肢を選択した場合は，カウントしない．d-r-e は，解答のはじめに def，return，end をこの順番で解答した学生，または def， writeimage，end を表現している．writeimage は，プログラムで書き込んだ配列の中身を出力するメソッドとして定義されたものである．def，return，end の間，def，writeimage， end の間に他の選択肢を選択した場合は，カウントしない．表 1 を見ると，問題 1,2,6 は正答率が高い問題であり，正答の順番通りに解答している傾向にあることがわかる．これは，プログラムの全構成をイメージし易かったため，順番通りに解答したことが考えられる．問題 1,2,3,5,6,8 のように，def と end を解答のはじめに選択する人が一定数いることがわかる．これは，プログラムの外側から組み立ててから中身を考えて

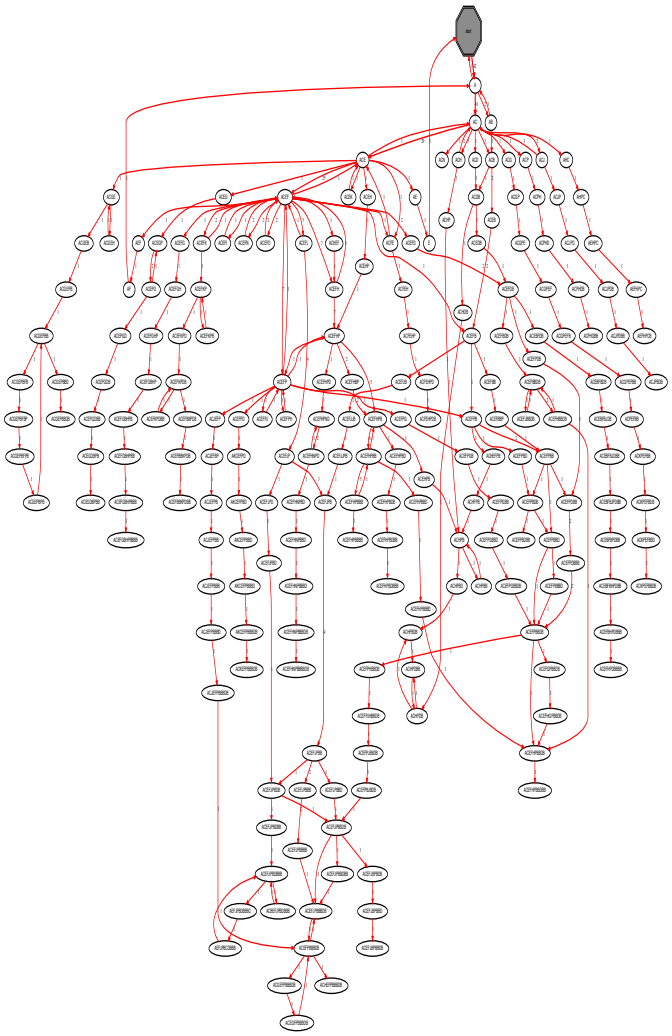

図 4　問題 5 の誤答者のみの状態遷移図

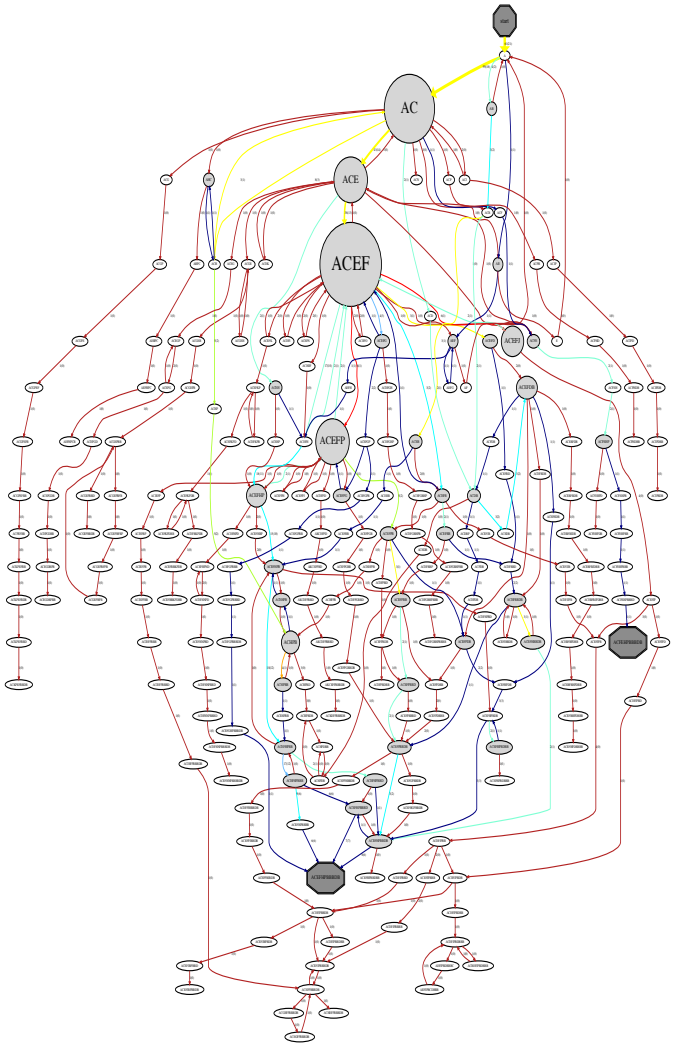

図 5　問題 5 の誤答者のつまずき箇所発見のための状態遷移図

表 2　問題 1〜10 の正答者のみの特徴的な選択内容

問題名	正答率	正答者数	順番	d-e	d-r-e
問題 1	73 %	45	26（58 %）	5（11 %）	4（ 9 %）
問題 2	79 %	46	35（76 %）	5（11 %）	5（11 %）
問題 3	39 %	24	6（25 %）	1（ 4 %）	0（ 0 %）
問題 4	3 %	2	0（ 0 %）	0（ 0 %）	0（ 0 %）
問題 5	34 %	21	7（33 %）	2（10 %）	0（ 0 %）
問題 6	79 %	50	33（66 %）	4（ 8 %）	10（20 %）
問題 7	24 %	15	7（47 %）	0（ 0 %）	0（ 0 %）
問題 8	31 %	19	1（ 5 %）	2（10 %）	0（ 0 %）
問題 9	15 %	9	0（ 0 %）	0（ 0 %）	0（ 0 %）
問題 10	2 %	1	0（ 0 %）	0（ 0 %）	0（ 0 %）

表 3　問題 1〜10 の誤答者のみの特徴的な選択内容

問題名	正答率	誤答者数	d-e	d-r-e
問題 1	73 %	17	1（ 6 %）	0（ 0 %）
問題 2	79 %	20	2（10 %）	0（ 0 %）
問題 3	39 %	38	2（ 5 %）	2（ 5 %）
問題 4	3 %	60	0（ 0 %）	0（ 0 %）
問題 5	34 %	41	2（ 5 %）	0（ 0 %）
問題 6	79 %	13	0（ 0 %）	2（15 %）
問題 7	24 %	49	4（ 8 %）	0（ 0 %）
問題 8	31 %	43	2（ 5 %）	0（ 0 %）
問題 9	15 %	51	0（ 0 %）	0（ 0 %）
問題 10	2 %	54	10（19 %）	0（ 0 %）

いると推測できる．表 2 を見ると，正答者同様に，def と end を解答のはじめに選択する人が一定数いた．また，解答のはじめに def,return,end の順に選択した人はいたが，正答者に比べて少ないという結果になった．

4.3.3　5 つの試験セットの正答率

5 つの試験セットの正答率を表 4 に示す．問題 4 を除いて，分野が同じ問題に対する各試験セットの正答率は大きく変わらない．従って，本研究で分析を行った試験セット 5 の結果は，他の試験セットと変わらないと考えられる．

4.3.4　全体的な分析結果と考察

正答者のみの状態遷移図からは，全体を通して，def や Class をはじめに選択することがわかる．確実に使用すると考えられるメソッド，クラスを一度選択してから中身を考えている可能性がある．また，正答率の高い問題は，正答の順番通りに解答している傾向にある．簡単な問題であれば，プログラム全体が把握できるためだと考えられる．また，一定数の正答者が def，end をはじめに選択してい

表 4　問題 1〜10 の問題セットごとの正答率

問題名	セット 1	セット 2	セット 3	セット 4	セット 5
問題 1	80 %	69 %	82 %	86 %	73 %
問題 2	71 %	72 %	80 %	68 %	77 %
問題 3	38 %	54 %	34 %	49 %	39 %
問題 4	49 %	35 %	14 %	16 %	3 %
問題 5	28 %	49 %	48 %	33 %	34 %
問題 6	77 %	85 %	98 %	91 %	79 %
問題 7	19 %	22 %	2 %	38 %	24 %
問題 8	52 %	52 %	51 %	59 %	31 %
問題 9	28 %	12 %	5 %	10 %	15 %
問題 10	14 %	4 %	2 %	20 %	2 %

た．これはプログラムの外側を先に組み立ててから，中身を考えている．これらの結果は，指導のポイントとして役立てられる可能性がある．

誤答者のみの状態遷移図からは，正答者のみの状態遷移図と比較して，傾向があるとはいえない．しかし，誤答の原因が考えられる選択肢も存在する．end や return などの制御構造にとって大事な記述が抜けていたり，そもそも構造がわかっていない可能性がある誤答者がみられた．また，プログラムの動きとしては問題ないが，冗長性があると判断され，誤答となった者もいた．end を挿入することを忘れるなどの単純だが重要なミスをなくすために，プログラムの外枠から組み立てることが指導のポイントになる可能性がある．

誤答者のつまずきポイント発見のための状態遷移図からは，正答率がかなり低い問題は，この状態遷移図では得られる情報が少ないことがわかった．また，つまずきノードから出ている赤いパスが他のつまずきノードと繋がっていることがあった．この場合，いくつかの選択肢が重要になる．正答者とは異なり，不要な選択肢を追加したり，必要な選択肢を削除してしまうため誤答となる場合が多い．誤答者のつまずき箇所発見のための状態遷移図は正答者のみ，誤答者のみの状態遷移図に比べて，傾向が掴みにくい．しかし，どの部分で正答のルートから外れてしまっているのか，どのような選択肢で間違えてしまっているかがわかった．

5.　実験

分析で使用した誤答者のつまずき箇所発見のための状態遷移図を用いて，TA 経験者を対象とする評価実験を行った．解答プロセスを可視化した状態遷移図を，指導に役立てるための課題を確認することを目的として実施した．

5.1　実験設計

評価実験の様子を図 6 に示す．被験者にはモニター，解答インタフェース（PC）を閲覧・操作しながら，記入用紙に記入してもらった．分析する問題は Ruby 言語のみであり，4 題分行う．正答率がそれぞれ異なる問題 3，6，7，9 を分析対象とした．モニターには，誤答者のつまずきポイント発見のための状態遷移図を表示した．PDF 形式で状態遷移図が出力されたため，選択肢の検索が可能であることを被験者に説明した．解答インタフェースは，学生が解答したものと同様に，選択肢をドラッグ＆ドロップで操作することによって，解答を組み立てることが可能である．学生が使用する画面とは異なり，あらかじめ正答を表示している．記入用紙には，各問題について，誤答者のつまずきだと思うポイント，誤答者のつまずきの内容，状態遷移図から気づいたことを自由に記述してもらった．被験者による分析が終わった後，アンケート，インタビューを行った．

アンケートの内容を表 6 に示す．いずれもリッカート尺度 (1:全くそう思わない，2:そう思わない，3:どちらとも いえない，4:そう思う，5:とてもそう思う) で尋ねた．質問 Q1-1 および Q1-4 については，4 つの問題についての平均値を指している．

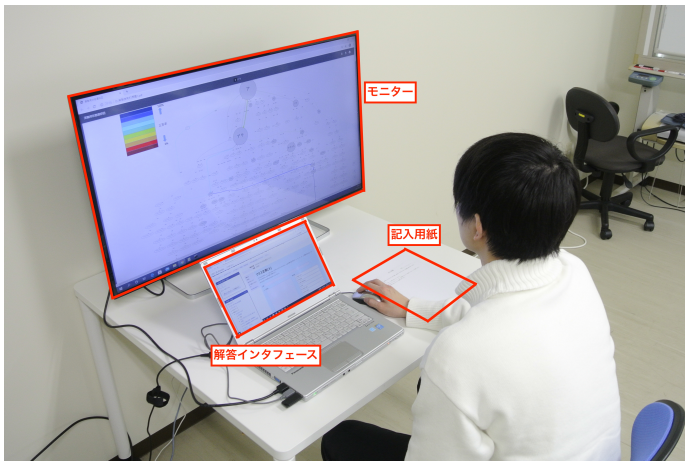

図 6　実験の様子

表 5　アンケートの内容

設問番号	内容
Q1-1	図から解答の状況を読み取れた
Q1-2	図に表示されている情報は十分であった
Q1-3	図から誤答者がつまずいているポイントがわかった
Q1-4	誤答者のつまずきとなるプログラムの内容がわかった
Q2-1	学生の解答プロセスに興味を持った
Q2-2	解答全体に傾向があると感じた
Q2-3	図が学生の指導において役立つと感じた
Q2-4	図を学生の指導において実際に使用したいと感じた

5.2　実験結果と考察

記入用紙，アンケート，インタビューの結果について分析と考察を行う．

5.2.1 記入用紙の結果と考察

記入用紙の結果について考察する．問題 3 について，多くの TA が初期化のタイミングでつまずいていることを指摘した．問題 6 について，色の間違い，writeimage メソッドの使い方がわかっていないと指摘しているものが多かった．問題 7 について，多くの TA が n の定義・宣言について指摘した．問題 8 について，get メソッドの動作がわかっていないと指摘していた．

全体として，つまずきと思う箇所について，比較的同じ箇所を指摘していた．つまずきの内容に関しても同様な指摘をしていたため，個人差はあるものの，ばらつきのない可視化ができたことが言える．

5.2.2 アンケートの結果と考察

アンケートの結果を表 6 に示す．Q1-2 について，図に表示されている情報は十分であるという結果になった．作成した状態遷移図は，クラスタリングをせずに遷移している人数が 1 人のパスも表示した．そのため，詳細な情報まで検索することができたため，解答の状況が読み取れたと推測する．また，Q1-4 について，平均値は若干有意な結果となった．TA1，2，3 からは良い評価であったが，TA4，5 は 2.50 と少し低い値であった．図に表示されている情報は十分であるという一方で，TA によっては情報量が過多だったと考えられる．Q1-3 の結果からも情報量が多すぎて，誤答者のつまずきポイントやその内容が掴みにくかったことが，考えられる．Q2-1 の結果より，実験参加者の TA 全員が解答のプロセスに興味を持つきっかけとなったことがわかる．このことから，TA が学生を指導する上で状態遷移図の活用が有用である可能性がある．一方で，Q2-3，Q2-4，Q2-6 の結果から，若干の有効性が確認された．TA1〜3 からは好意的な評価を受けたが，TA5 からはかなり非好意的な評価を受けた．しかし，状態遷移図に表示されている情報量を調整することで，この結果を改善することができると考えられる．

表 6 アンケート結果

設問番号	TA1	TA2	TA3	TA4	TA5	Ave.	S.D.
1-1	4.00	4.00	4.25	3.00	3.50	3.75	0.20
1-2	3.50	4.00	4.00	3.50	5.00	4.00	0.30
1-3	4.00	3.50	3.75	2.75	3.25	3.45	0.19
1-4	4.25	4.75	4.00	2.50	2.50	3.60	0.87
2-1	4.00	5.00	5.00	5.00	4.00	4.60	0.49
2-2	4.00	2.00	4.00	4.00	3.00	3.40	0.80
2-3	4.00	4.00	4.00	3.00	2.00	3.40	0.80
2-4	4.00	4.00	5.00	3.00	1.00	3.40	1.36

5.2.3 インタビューの結果と考察

誤答者のつまずき箇所をどのように判断したかインタビューしたところ，サイズが大きい灰色のノードから見たとう意見が多かった．ノードの大きいものから出ている赤いパスをたどったり，該当の選択肢を見比べて判断したと述べていた．一方で，全体的な傾向を掴むのが難しく，状態遷移図の確認により発見できる情報が何かがわからなかったという意見もあった．これは情報量が多すぎて，判断が難しかったことが考えられる．

次に，傾向について尋ねたところ，正答者の傾向はわかりやすいが誤答者の傾向はわかりづらいという回答が多かった．正答者の傾向は，図の青いパスをたどり，ラベルの人数やノードを見て判断したと述べていた．誤答者の傾向は，特定しづらかったという意見が多かった．これは，赤いパスに大きな特徴がなかったことが考えられる．

また，状態遷移図が解答プロセスを分析する手段としてふさわしいか質問したところ，現状の状態遷移図がふさわしいと感じた被験者が多かった．学生が解答した過程を知るためには状態遷移図がふさわしいが，改良が必要であるという意見もあった．誤答者のつまずき箇所を判断できないことを指摘した TA は，情報量を減らした方が見やすいと述べていた．状態遷移図の改善点として，情報量が多いため削って欲しいという点が指摘された．現状の状態遷移図では，ノード間を遷移している人数が 1 人であっても表示しているため，クラスタリングや非表示機能などの工夫を検討する必要がある．また，つまずきノードに合計の人数を表示する機能や，選択肢同士の関係を考慮された形の表現などの改善案が出た．また，インタビュー後に正答者と誤答者の状態遷移図を分けた方がいいのではないかという意見も得られた．

以上のように，現状の状態遷移図にはさらなる工夫の余地があり，教員や TA らに支援しやすい形に改善する必要があることがわかった．

6. おわりに

本研究では，初学者を対象とした短冊型プログラミング問題における学生の解答プロセスを状態遷移図を用いて可視化し，分析と評価実験を行った．

分析の結果，正答者の傾向として，def，end などのプログラムの外側から組み立てること，正答率の高い問題であれば，正答の順番通りに解答することがわかった．誤答者の傾向としては，end や return などの単純だが重要な選択肢が抜けていることがわかった．このことから制御構造のプログラムから組み立てて考えることが，指導のポイントにつながる可能性があると言える．

評価実験の結果，現状の状態遷移図は情報は十分であり，解答プロセスを分析する手段として状態遷移図がふさわしいことがわかった．一方で情報量が多すぎることから，傾向やつまずきポイントが判断しにくい結果となった．インタフェース上の工夫や，表示する情報の工夫などの改善の余地があることがわかった．以上のことから，学生の解答

プロセスを適切な形で可視化することで，教員や TA らの指導の支援につながる可能性が示唆された．

インタビューで指摘された内容のように，現状の状態遷移図には改善の余地がある．情報量を減らすためのクラスタリングや，表示される情報を自由に操作できるインタフェースの改善などが考えられる．

また，状態遷移図を用いて個別の学生の傾向をどのように判定し，TA に伝えるかが課題である．実際に TA の指導の支援になる可視化の方法を模索する必要がある．例えば，リアルタイムで学生の解答状況を収集し，座席表に情報を表示し，それをもとに TA に指導をしてもらうことなどが考えられる．状態遷移図から得られる情報を定量化して表示したり，正答者と誤答者と個別の状態遷移図を比較したりといったことが考えられる．

これらのことを踏まえ，学生への指導のポイントとなるような気づきが得られる TA への可視化方法を探索し，実際に教員や TA らに支援可能な手法を考案してゆく．

謝辞　本研究の一部は，JSPS 科研費 JP19H01710 の助成を受けたものである．

参考文献

[1] M. McCracken, V. Almstrum, D. Diaz, M. Guzdial, D. Hagan, Y. B.-D. Kolikant, C. Laxer, L. Thomas, I. Utting, and T. Wilusz. A Multi-National, Multi-Institutional Study of Assessment of Programming Skills of First-Year CS Students. ACM SIGCSE Bulletin, Vol. 33, No. 4, pp. 125-180, 2001.

[2] R. Lister, E. S. Adams, S. Fitzgerald, W. Fone, J. Hamer, M. Lindholm, R. McCartney, J. E. Moström, K. Sanders, O. Seppälä, B. Simon, and L. Thomas. A Multi-National Study of Reading and Rracing Skills in Novice Programmers. ACM SIGCSE Bulletin, Vol. 36, No. 4, pp. 119-150, 2004.

[3] Gomes Anabela and Mendes.A.J. Learning to program-difficulties and solutions. International Conference on Engineering Education 2007(ICEE2007), pp. 283-287, 2007.

[4] Essi Lahtinen, Kirsti Ala-Mutka, and Hannu-Matti Jarvinen. A study of the difficulties of novice programmers. ACM SIGCSE Bulletin, Vol. 37, pp. 14-18, 2005.

[5] Yasuichi Nakayama, Yasushi Kuno, and Hiroyasu Kakuda. Split-paper testing: A novel approach to evaluate programming performance. Information Processing, Vol. 28, pp. 733-743, 2020.

[6] 久野靖. 短冊型問題を用いたプログラミング学習アドバイスツール. 第 61 回プログラミングシンポジウム報告集, p. 129-139, 2020.

[7] P. Ihantola and V. Karavirta. Two-Dimensional Parson’s Puzzles: The Concept, Tools, and First Observations. Journal of Information Technology Education: Innovations in Practice, Vol. 10, pp. 1-14, 2011.

[8] P. Denny, A. Luxton-Reilly, and B. Simon. Evaluating a New Exam Question: Parsons Problems. In Proceedings of the Fourth international Workshop on Computing Education Research, pp. 113-124, 2008.

[9] R. Lister, T. Clear, D. Bouvier, P. Carter, A. Eckerdal, J. Jacková, M. Lopez, R. McCartney, P. Robbins, O. Seppälä, et al. Naturally Occurring Data as Research Instrument: Analyzing Examination Responses to Study the Novice Programmer. ACM SIGCSE Bulletin, Vol. 41, No. 4, pp. 156-173, 2010.

[10] Juha Helminen, Petri Ihantola, Ville Karavirta, and Lauri Malmi. How do stu- dents solve parsons programming problems? - an analysis of interaction traces. Nineth Annual International Conference on International on Computing Education Research(ICER’12), pp. 119-126, 2012.

[11] Salil Maharjan and Amruth Kumar, Using Edit Distance Trails to Analyze Path Solutions of Parsons Puzzles, Educational Data Mining 2020 (EDM 2020), 2020

[12] Taku Yamaguchi and Michiko Oba. Measurable Interactive Application to Find Out User Recognition and Strategy when Problem Solving. Journal of Software, Vol. 15, No. 1, pp. 12-22, 2020.

[13] 米持幸寿, 大場みち子. プログラム・コードの並べ替えパズルにおける正解との距離の変化. 情報教育シンポジウム論文集, pp. 47-53, 2020