情報処理学会論文誌　教育とコンピュータ　Vol.8　No.2　37–50 (June 2022)

ショートペーパー

オンラインScratchプログラミング演習支援にむけたコードメトリクス可視化ツールの提案および評価

槇原 絵里奈[1,a)]　米田 浩崇[2,b)]　小野 景子[1,c)]

受付日 2021年6月30日，再受付日 2021年12月4日，
採録日 2022年2月19日

概要：小学校のプログラミング教育において，自己の知識・理解の不足に課題を感じる教員が多いことが指摘されている．プログラミング演習における教員支援として，コードの行数や複雑さ（サイクロマティック複雑度）など，コードメトリクスを可視化する支援手法が存在する．しかし，教員のプログラミングに対する知識が，可視化されたコードメトリクスの解釈に与える影響は明らかになっていない．そこで我々は，コードメトリクスをリアルタイムで可視化可能なシステムを構築し，教育やプログラミングの知識の違いが，可視化されたコードメトリクスの判断にどのような影響を及ぼすか分析を行った．可視化されたコードメトリクスを教育およびプログラミング経験の異なる教員らに提示した結果，同様の学習者を指摘する場合でも参照するメトリクスやメトリクス数が異なることを確認した．

キーワード：Scratch，プログラミング教育，コードメトリクス，可視化

Visualizing Code Metrics in Scratch to Support Distance Programming Exercise

Erina Makihara[1,a)]　Hirotaka Yoneda[2,b)]　Keiko Ono[1,c)]

Received: June 30, 2021, Revised: December 4, 2021,
Accepted: February 19, 2022

Abstract: In programming education for elementary school, many teachers concern themselves for lack of knowledge/comprehension of programming. It is reported that visualized code metrics (e.g., lines of code, cyclomatic complexity) is able to support educators in programming education. However, it is not clear how the difference in programming knowledge affects interpretations of code metrics. Therefore, in this paper, we investigate that the difference in education and programming knowledge affects comprehension of visualized code metrics in Scratch. Our investigation reveals the differences in metrics and the number of metrics when educators, who have different programming knowledge, point out a student.

Keywords: scratch, programming education, code metrics, visualization

1. 序論

情報通信技術が社会に与える影響は多分野に拡大し，世界的に初等教育段階からのプログラミング教育が本格化している．日本においても2020年度より小学校においてプログラミング教育が必修化された．一方で，2016年度に522名の小学校教員を対象に行われた調査 [1] では，92.0%がプログラミング教育に関する知識・理解が不足していると述べており，プログラミングに対する不安がうかがえる．加えて，2020年度よりCOVID-19の影響によりオンライン授業を余儀なくされた教育機関も多く，多くの小学校教員が慣れない環境下でプログラミングを教えることになり，学習者だけでなく教員の支援も急務であるといえる．

[1] 同志社大学理工学部
Faculty of Science and Engineering, Doshisha University, Kyotanabe, Kyoto 610–0394, Japan
[2] 同志社大学大学院理工学研究科
Graduate School of Science and Engineering, Doshisha University, Kyotanabe, Kyoto 610–0394, Japan
[a)] emakihar@mail.doshisha.ac.jp
[b)] hyoneda@mikilab.doshisha.ac.jp
[c)] kono@mail.doshisha.ac.jp

そこで，本研究では小学校におけるオンラインプログラミング教育支援として，コードメトリクスの可視化に着目する．コードメトリクスとはソースコードの状態を数値的に表したものであり，ソースコードの行数や変数の数，循環的複雑度などが含まれる．先行研究 [2] において Scratch コードメトリクス可視化・分析ツール CRAVER を提案した．CRAVER は Scratch プログラミング環境である学習者用 UI と教員用 UI を持ち，学習者 UI から取得されたソースコードの編集履歴を取得する．そして，教員用 UI において，全学習者の編集履歴を可視化したコードメトリクスの一覧機能を提供する．CRAVER はほかに変数名のワードクラウドやブロック数などの下限上限の指定によるフィルタリング・検索機能，指定したユーザの特定の時間のソースコードを表示する詳細表示機能を持つ．教員や TA（ティーチングアシスタント）はこれらの機能を用いることで，主体的に学習者全体の進捗を確認しつつ，相対的に遅れている学習者や不適切な変数名などを使用している学習者のソースコードを確認することができる．しかしながら，既存研究において，可視化されたデータの解釈は各個人に委ねられることがほとんどである．教員のプログラミングに対する知識が少ない場合，メトリクスの解釈が誤っていたり，指導が必要な学習者を発見できない可能性がある．一方，適切な可視化が行われている場合，プログラミングやプログラミング教育に対する知識量に関係なく，問題のある学生やその原因を特定できる可能性も考えられる．

したがって本研究では，コードメトリクスの可視化による教員支援のための第 1 歩として，プログラミングや教育経験の違いが，可視化されたコードメトリクスの判断にどのような影響を及ぼすか，また，各メトリクスがユーザへ与える効果について調査を行った．調査ではまず，CRAVER を利用して学習者のコーディング履歴のログを収集した．次に，収集したログを CRAVER を通して可視化し，Scratch プログラミングや，プログラミング演習における TA 経験の異なる被験者に提示した．被験者がどのメトリクスに基づき学習者にどのように指導するかを調査することで，可視化したメトリクスの効果を分析する．本研究の結果は，オンラインにおける教員支援のツールを開発するときのデータの選定や，経験の浅い教員に対するアドバイスなどに使用することができると考える．

2. 準備

2.1 Scratch

Scratch とは，MIT メディアラボが開発したブロックベースのビジュアルプログラミング言語およびその学習環境である[*1]．ゲームやインタラクティブなアニメーションの作成に特化しており，実行結果がアニメーションで表示されるため初学者への導入が容易である．Scratch はプログラミング入門教育のための環境でありながらもチューリング完全であり，非常に大規模で複雑な作品も作成可能である点で，教材として有用である．

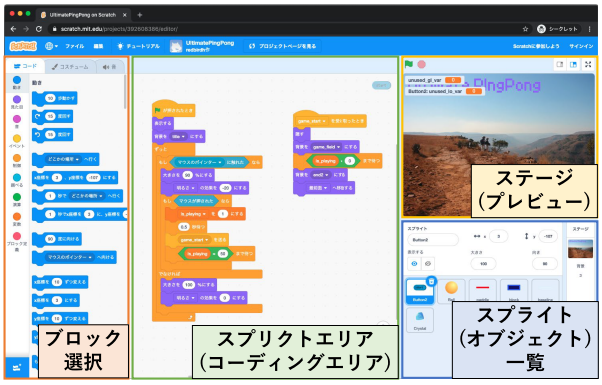


図 1　Scratch プログラミングの例
Fig. 1　An example of Scratch program.

図 1 に Scratch のコーディング画面を示す．Scratch では，実行画面上に表示される各画像をスプライトと呼び，各スプライトに対し期待する動作を実装しプログラミングを行う．このスプライトに実装される一連のブロックをスクリプトと呼ぶ．通常スクリプトは，実行開始やキー入力などのイベントを受け取るブロックを先頭に持ち，イベントをトリガーとして実行される．このようなブロックをハットブロックと呼ぶ．1 つのスプライトは，複数のハットブロックを持つことができ，その場合，イベントを満たしたスクリプトはすべて非同期に実行される．

2.2 関連研究

2.2.1 Scratch 支援に関する研究

一般的なテキストプログラミング言語と同様に，Scratch でも同じ動作に複数の実装方法がある．そのため，Scratch で作成された作品にもプログラムの品質の良し悪しが存在し，品質改善に関わる研究が多く行われている [3], [4]．

Hermans ら [3] は，ソースコードが問題を抱えており改善が必要であることを示す指標 Code Smell に注目し，Scratch プログラムにおける Code Smell を調査した．調査の結果，特定の Code Smell の存在は学習者らのプログラムに対する理解度の低下や，プログラムの修正を困難にすることが分かった．また，Techapalokul ら [4] は，Code Smell に基づいた Scratch のリファクタリングを 4 種類定義し，自動リファクタリング機能を備えた Scratch 環境を提案している．

これら Code Smell の多くはコードメトリクスの測定より検出が可能である．すなわち，コードメトリクスを基に Scratch プログラムの品質改善は可能であり，コードメトリクスは教員にとっても有益なデータである．一方，プロ

*1　https://scratch.mit.edu/

グラミングに対する知識がコードメトリクスの解釈に及ぼす影響については明らかになっていない．

2.2.2　コードメトリクスを利用したプログラミング演習支援に関する研究

田中ら [5] はビジュアルプログラミング言語が，コンパイルエラーを回避可能である点に着目した．Java とビジュアルプログラミング言語を選択可能なプログラミング環境を構築し，コーディングメトリクスを基に，各メトリクスの収集および可視化，そしてコンパイルエラーの回避が学習者へ寄与する効果について調査した．田中らの提案するメトリクスは，Java と比較を行うため，エディタ利用時間やエラー継続時間など，プログラミング環境の操作量を示すものが多く，コードの状態を示すメトリクスはコード行数のみであった．また，田中ら [6] は Java 演習においてもコードメトリクスを提案，可視化し，可視化結果の閲覧による授業改善に関する議論の質的評価を行った．

オンラインの演習において，学習者の演習に対するモチベーションや集中力を測るためにはエディタの操作量なども重要であると考える．一方，本研究ではオンラインのプログラミング演習において，プログラムの品質改善につながる指導を教員が主体的に行うことを目指すため，プログラムのメトリクスに着目する．

2.2.3　プログラミング経験に着目した研究

プログラミングの経験に着目した研究として，Busjahn ら [7] はプログラミング初心者と熟練者のコードリーディング時における視線運動の違いを明らかにした．また，Lahtinen ら [8] は教員と学習者へプログラミングに関する同様の質問を行い，教員と学習者が難しいと感じる要素の違いを調査した．

これらの研究では，プログラミング経験の違いによって，プログラムの読解や理解，また難しいと感じる箇所が異なることを指摘している．加えて，プログラミング初学者と熟練者の違いを明らかにすることで，初学者のプログラム理解の促進や，効率的なプログラミング学習につながることが示唆されている．すなわち，同じ教員に対しても，プログラミング経験やプログラミング教育経験に対する違いが，学習者指導へどのような影響を与えているか調査することで，質の高い教育や効果的な教育につながることが考えられる．

以上より，本研究では複数のコードメトリクスを可視化可能な教員支援システムを提案し，プログラミングに関する知識や教育経験が異なる教員が注目するコードメトリクスの違いや傾向について調査する．

3.　コードメトリクス可視化システム CRAVER の提案

3.1　概要

本研究では授業内で学習者全員の Scratch プロジェクトをリアルタイムに分析および可視化することで教員支援を行うシステム CRABER（Collaborative Real-time Analyzer Based on Extracted Rules in Scratch）を提案する [2]．CRABER の概要図を**図 2** に示す．CRABER は Web アプリケーションであり，教授者と学習者によって構成される Scratch プログラミング教育で使用されることを想定する．教員・学習者ともにブラウザを通した利用が可能である．CRABER は全学習者のプロジェクトの状況を 1 分間隔で取得および分析し，全学習者のプロジェクトの分析結果を教員向け画面に表示する．

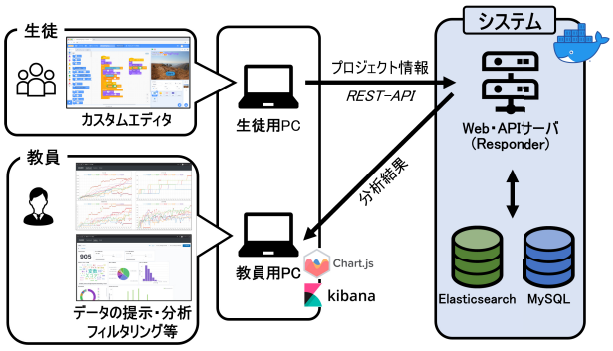


図 2　CRABER のシステム概要図
Fig. 2　An overview of CRAVER.

CRABER の学習者ページには図 1 と同様の Scratch のカスタムエディタを表示する．学習者ページのエディタは，オリジナルの Scratch エディタ[*2]に，バックグラウンドでプロジェクトの内容を 1 分間隔でサーバに送信する機能を追加したものである．したがって，学習者は図 1 に示される通常の Scratch プログラミングと同様の環境で，各自自由にコーディングを行うことができる．

教員向けページには学習者ページより収集した Scratch プログラム編集履歴のログを分析した結果を表示する．教員向け画面で提示される機能を以下に示す．

Dashboard：メトリクス一覧表示機能　全学習者の編集履歴の概要をコードメトリクスに基づき一覧表示する

Explore：フィルタリング・検索機能　変数名のワードクラウドやブロック数などの下限上限を指定することで，特定の特徴を持つユーザのみを表示する

Detail：詳細表示機能　指定したユーザの特定の時間のソースコードを表示する

図 3 は Dashboard の画面であり，3.2 節で述べる各種メトリクスを可視化する．教員は全学習者の可視化された各メトリクスをリアルタイムに閲覧可能である．特定の学習者のグラフをクリックすることで，具体的な値も閲覧できる．なお，教員向けページにおいて表示されるデータは 1 分ごとに最新の状態に更新される．各種データの探索と可

[*2] https://github.com/LLK/scratch-gui

図 3　CRABER の教員向けメトリクス一覧ページの UI
（左上：BLN，左下：CCN 平均値，右上：SPN，右下：LOB）

Fig. 3　User interface of program metrics view for educator in CRABER (upper left: BLN, lower left: CCN, upper right: SPN, lower right: LOB).

視化には Kibana[*3] と Chart.js[*4] を用いた．

3.2　コードメトリクスの収集・分析

プロジェクトの分析はコードメトリクスを基に行う．コードメトリクスは既存研究より頻繁に用いられる，コード行数，インスタンス数，トークン数，循環的複雑度を対象とした．しかしながら，コードメトリクスに関する既存研究はテキストベースのプログラミング言語を対象にしたものが多いため，本研究ではビジュアルプログラミング言語の特徴を考慮し，以下のように修正し使用した．

BLN（BLock Number）：プロジェクト全体で使用されているブロックの総数

SPN（SPrite Number）：プロジェクト全体で使用されているスプライトの総数

LOB（Lines Of Block）：各スクリプト（実行されるプログラムの塊）あたりのプログラムの長さの平均値

CCN（Cyclomatic Complexity Number）：各スクリプトの循環的複雑度の平均

BLN はエディタ上に存在するすべてのブロックを含み，実行されていないブロックも含まれる．LOB はプログラム開始点であるハットブロックから始まる実際に実行されるブロックの長さである．図 1 の場合，17 個と 7 個のブロックの塊があるため，BLN は 24，LOB は 12 となる．既存研究 [12], [13] において，BLN や LOB のようなソースコード行数から，学習者の進捗やつまずき，相対的に遅れている学習者の特定が可能であることが示されている．

また，SPN はプログラム中に使われるイラストや背景を指すため，課題によって必要な数が明確である特徴を持つ．したがって，SPN の低い学習者は進捗が悪く，高い学生は Scratch のクローンという機能を使いこなせていない，課題を理解していないなどの推測につながる．CCN はスクリプトで使用されるブロックのうち，処理が分岐する可能性のある条件分岐，繰り返し処理のブロック数に 1 を足したものである．分岐の数に 1 を足す計算は，一般的なテキストプログラミングにおける循環的複雑度の計算と同様であり，ソースコードの質の評価に広く使用される [9]．また，Scratch は複数のスクリプト，すなわち実行ボタンが押された際に実行されるソースコードの塊を複数作成することができる．そのためスクリプトの行数と循環的複雑度は 1 つのプログラムに複数存在することとなる．CRAVER では，メトリクス一覧表示機能において LOB と CCN を表示するにあたり，平均値を用いることで，いずれかのスクリプトで異常値が発生していることをユーザへ提示する．異常値の発生原因に関しては，たとえば BLN が急激に減った場合，LOB が大きく変わらないのであればデッドコードを削除したため，SPN も減少しているのであればスプライ

*3　https://www.elastic.co/jp/kibana/
*4　https://www.chartjs.org/

トを削除したためなど，複数メトリクスを組み合わせることで推測可能である．また，本調査では用いないものの，異常値が発生した具体的な箇所やそのときのソースコードに関しては，詳細表示機能で閲覧可能である．

Aivaloglou ら [10] による既存の Scratch プロジェクト約 2,800 万を対象にした調査の結果，初学者のソースコードにも深刻な問題が存在する兆候である Code Smell（コードの匂い）が存在することが分かった．Code Smell は長すぎるメソッドや複雑な設計などが含まれ，BLN や CCN などから検出が可能である．一方，Code Smell の検知にはプログラミングに対する深い知識と経験が必要であるといわれる [11]．したがって，CRAVER のコードメトリクス可視化画面では，小学校における教員を支援するために，学習者内において相対的にソースコードが長く，あるいは複雑であるユーザを明示する機能が必要であると考えた．また既存研究 [12] より，コードメトリクスから相対的に遅れている学生の検出も可能であるため，複雑なプログラムだけでなく簡素なプログラムも検出することが，進捗の把握には必要であると考えた．

以上より，Scratch プログラミングにも詳しく，プログラミング教育に関する授業を複数持つ教員とともに，各メトリクスがどれほど中央値から外れた場合に，ソースコードに問題を抱えている，あるいは課題につまずいており指導が必要であると判断するかを議論した．その結果，BLN は 20，LOB は 3，CCN は 3，SPN は 2 以上の差が 3 分以上生じ続けた場合，CRABER はその学習者のメトリクスは全体から大きく離れていると見なし，グラフ上にクロスマークを付与する．すなわち，特定の時点においてクロスマークが付与されたユーザは，他のユーザと比べ，相対的に複雑，あるいは簡素なプログラムを記述していることとなり，指導の基準の 1 つとして考慮することが可能となる．本調査では解答例の規模を基に上記の基準を設けたが，クロスマークを付与する基準値はシステムで自由に変更できる．したがって，課題と異なる回答をどれほど許可するかといった教員のポリシーや，課題の内容に合わせた閾値を用いることが可能である．

4. 調査

4.1 対象データ

本調査を行うにあたり，先行研究 [2] において実施した，情報系学部に属する 20 代大学生 10 名に，実際に CRAVER を用いて Scratch プログラミングを行ってもらった際の編集履歴のログを使用する．使用した課題は小問 5 つからなり，小問の誘導に従い 1 つのゲームを完成させるものである．学習者へ与えた課題の完成図を図 4 に，小問を以下に示す[*5]．

[*5] https://sites.google.com/mikilab.doshisha.ac.jp/mscratch

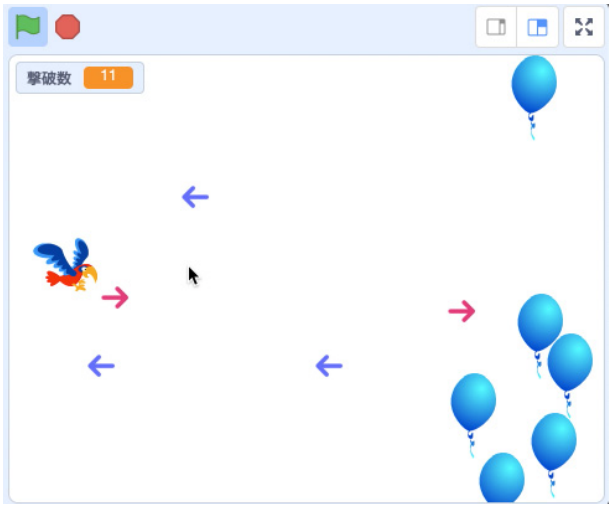


図 4　課題の完成イメージ図
Fig. 4　An example of completed program.

小問 1　鳥のスプライト（スプライト一覧画面「Parrot」で検索）を追加し，マウスカーソルの上下に追従するようにする．横には移動しない．

小問 2　矢印のスプライトを追加し，キーボードのスペースキーを押下することで鳥が矢印を発射できるようにする．

小問 3　風船のスプライトを追加し，ランダムなタイミングで右下から右上に一定速度で上がるようにする．出現頻度はランダムとする．

小問 4　風船と矢印の当たり判定を実装する．矢印に当たると風船は消えるが矢印は貫通する．矢印が破壊した風船の数をスコアとして表示する．

小問 5　風船も鳥と同様矢印を発射し，これに鳥が当たると即ゲームが終了するようにする．ゲームが終了とは，「すべてを止める」ブロックが実行されることを指す．発射間隔はランダムとする．

本調査では各小問をサブゴールと見なし，課題に完答しなかった場合もサブゴールの達成個数を用いて評価を行う．制限時間は 60 分であり，各小問へ割り当てる時間や休憩をとるタイミングは各学習者自由とした．被験者の Scratch プログラミング経験は統一しなかったが，基本操作については事前に説明を行った．以下，本研究では先行研究において Scratch 編集履歴のログを取得した際の被験者を学習者，あるいは学習者 A から J と表記する．

データの収集は実プログラミング演習に即し，教員から学習者へ課題を与え，制限時間内に解くように指示した．ただし，実施形態は新型コロナウイルスの影響によりオンラインとし，実験終了後すぐにヒアリングなどを行うことから 5 名ずつの実施とした．被験者には課題は独力で解き進めるように指示し，参考書や Web の使用は許可したが，他者との相談や課題に関する質問は不可とした．

ログ取得実験では，制限時間内に課題が完成した学習者

表 1　各フェーズにおいて学習者が達成したサブゴール（F：フェーズ）
Table 1　The number of accomplished sub-goals by each student.

学習者 ID	F1	F2	F3	F4	F5	F6
学習者 A		1		2		
学習者 B			1			
学習者 C	1，2	3	4			
学習者 D	1，2					3
学習者 E	1，2				3	
学習者 F		1	2		3	4
学習者 G		1				
学習者 H		1		2，3		4
学習者 I		1				
学習者 J		1		2	3	

はいなかった．しかしながら，本調査の目的は学習者の課題の完成ではなく，可視化された学習者のコードメトリクスがユーザへ与える影響調査であるため，本データはすべて調査対象と見なし分析を行う．そこで，分析を行うにあたり，各学習者が小問をクリアすることをサブゴールの達成と見なし，着目する．各学習者のサブゴール達成個数および，実験時間を 10 分おきに区切った際の各サブゴール達成時のフェーズについて**表 1** に示す．実験では各学習者に対し，小問 1 から解き進めること，さらに小問の題意を満たすプログラムを作成したと判断したら，その旨を第 2 著者である実験監督者へ伝えてから次の小問へ移ることを伝えていた．そのため，全学習者が小問 1 より順に着手しており，未完成にもかかわらず次の小問へ移った学習者はいなかった．したがって，本研究ではサブゴールの達成をプログラムの進捗率の判断として扱うこととする．

4.2　調査概要

本調査の目的は，プログラミングに関する知識や教育経験が異なるユーザが可視化されたメトリクスを基に指導を行う際，注目するメトリクスおよびメトリクスの解釈にどのような違いが生じるかを明らかにすることである．そこで，4.1 節で得られた学習者のログを可視化したものを，Scratch に関する知識やプログラミング演習系科目の TA 経験が異なる被験者に提示し，どのようなメトリクスに注目し，どのような指導が必要だと考えるかについて調査を行う．

一般的に，対象のデータについて詳細を知らないユーザでも，データを適切に可視化することで，そのデータが示す情報を直感的に理解することが可能になるといわれる．一方，プログラミング教育において，学習者のプログラム課題の進捗は 1 つのコードメトリクスから図ることは困難であり，複数のデータを組み合わせる必要がある．

具体例について，4.1 節における小問 1 を解き進めたときのソースコードおよび各メトリクスの遷移を**図 5** に示す．

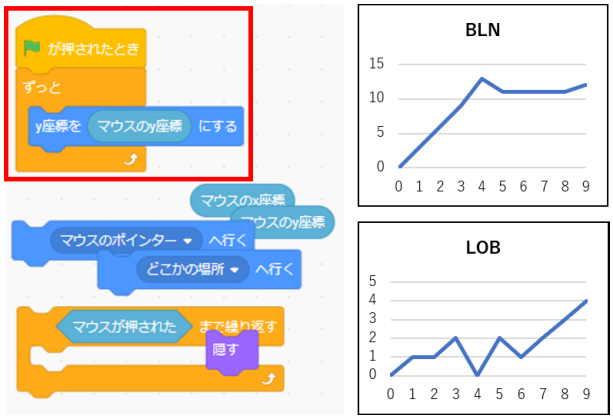


図 5　BLN と BOL の遷移の例
Fig. 5　An example of the transition of BLN and BOL.

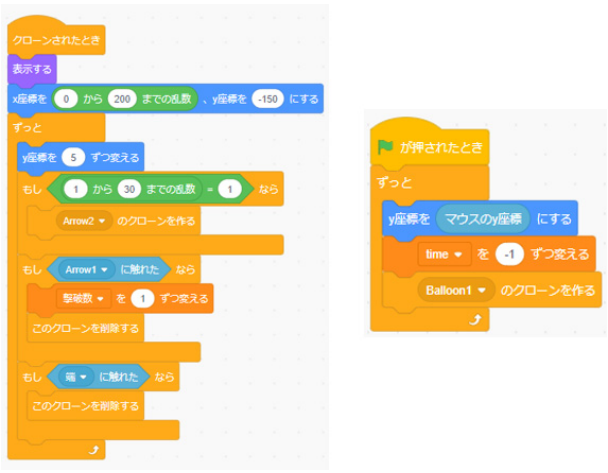


図 6　異なる CCN の例
Fig. 6　An example of the different number of CNN.

図 5 の BLN の遷移のみに着目したとき，途中まで順調に増えていたブロック数が後半に停滞していることから，後半に課題に行き詰まり手が止まったことが予想される．しかし LOB，すなわち実行プログラムの平均ブロック数は変化していることが分かる．Scratch は図 5 の左に示すように，プログラミング環境内に実行しないソースコードを置くことができる．画面上には 11 のブロックが存在するが，実行されるのは左上に赤枠で囲ったブロックのみである．すなわち，Scratch においては単純にブロック数の変化だけで進捗を図ることは容易ではない．実際に図 5 のソースコードは，序盤から不必要なコード片をプログラミング環境上に置いて闇雲な試行錯誤を行っていたものの，後半には完成形のイメージを持ってコードを組み合わせていた．Scratch の知識がある教員の場合，このような Scratch の特徴に鑑みた指導ができる可能性がある．

また，**図 6** は異なる被験者が小問 4 へ取り掛かっているときの，鳥（Parrot）に対するソースコードである．実際は他のスプライトも存在するものの，図 6 のみの CCN は，左図が 4，右図が 1 となる．同様の進捗でもこのよう

に特定のスプライトに対するプログラムは異なっており，CCN に注目することで，学習者がどれほど複雑なプログラムを作成しているかが推測可能となる．

教育経験が豊富な場合，学習者の様々な状況について推察できると考える．たとえば，Scratch や Java などプログラミング言語に関係なく，プログラミング初学者は正答に近づかないような試行錯誤を行い，手は動いているもののコーディングが停滞する場合がある．プログラミング演習などの授業においてそのような学習者の試行錯誤を確認している場合，順調にメトリクスが変化している学習者に対しても，指導が必要な可能性を示唆できると考える．図 5 の前半の場合，BLN が増えていることを順調と見なすのではなく，CCN と組み合わせることでソースコードが複雑になっていないか注意を行うことや，SPN と組み合わせることで不要なスプライトを作成せずに必要なスプライトに対する処理を優先して考えることなどのアドバイスが可能である．また，スプライトは Scratch 独自の機能であるが，特定のものに対する処理を記述するという点では，他言語におけるオブジェクトやファイルと見なすこともできる．すなわち，上述したアドバイスは Scratch のプログラミング経験に関係なく，プログラミング初学者がコーディングに行き詰まる事例に対する知識の有無でアドバイスできる可能性がある．

一方，参照するメトリクスが過剰である場合，かえって教員が混乱する恐れがある．実行ブロックを修正しているかは BLN と LOB の組合せから，実行ブロックの中でも条件分岐や繰り返し処理に試行錯誤しているかは BLN と CCN から，今回の課題のように各小問で必要なスプライト数が分かっている場合，どの小問に着手しているかは BLN 関係なく SPN のみから推測可能である．すなわち，学習者の状況を想定するために，適切に各メトリクスを選定できるかについても，教員のプログラミング能力や教育経験が関与する可能性がある．

以上より，本研究ではプログラミングに関する知識と教育経験が，可視化されたコードメトリクスの解釈に与える影響について調査するために，以下のリサーチクエスチョンを設定する．

RQ1　プログラミング経験が異なる場合，参照するメトリクスの数や種類に違いが生じるか？

RQ2　教育経験が異なる場合，指導内容に違いが生じるか？

RQ1 に対応するために，各教員役被験者が注目したメトリクスを取得する．その際，プログラミング経験のある教員役被験者が注目したメトリクスの特徴を調べるために，注目されたメトリクス数やその割合，各メトリクスの平均値との差に注目し，定量的に調査を行う．また，RQ2 に対応するために，各教員役被験者が RQ1 で着目したメトリクスに対しどのような指導内容を想定し注目したのか，アンケートによる定性的調査を行う．

4.3　教員役被験者

プログラミングや教育経験の違いが，可視化されたのコードメトリクスの解釈に及ぼす影響について調査を行うため，プログラミング経験およびプログラミングに関する演習の TA 経験が異なる，情報系学部に所属する 3 名を被験者とし，可視化したコードメトリクスを閲覧してもらった．以降，3 名の被験者を T1 から T3 と示す．T1 から T3 の属性について以下に示す．

T1：プログラミングに関する複数の講義の TA を経験しており，Scratch に関する知識も深い．

T2：プログラミングに関する複数の講義の TA を経験しているが，Scratch は数回の使用経験である．

T3：TA 経験はなく，Scratch も数回の使用経験である．

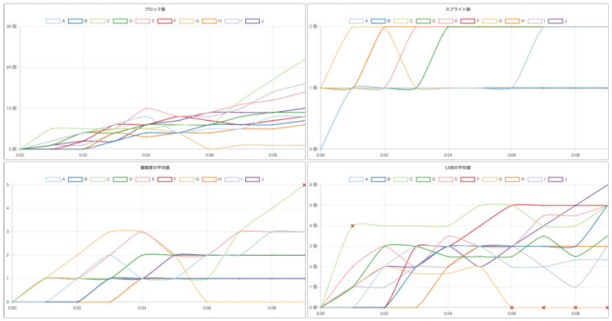
被験者は 3 名とも 20 代大学生であり，C 言語や Java など他のプログラミング言語については授業などで学習済みである．T1 は自身の研究でも Scratch を用いており，また複数のプログラミング演習に関する科目の TA を経験している．一方，T2 と T3 は Scratch を含むビジュアルプログラミング言語に関する経験が浅く，実験後のヒアリングにおいても，Scratch 独自である SPN や LOB をどのように解釈して良いかが困難であったことが指摘された．すなわち，本調査において C 言語や Java に関する知識を有していても，Scratch に関する知識が浅いことによる影響が存在することが示唆される．以降，本調査におけるプログラミング経験やプログラミングに関する知識は，Scratch に関するものを指す．

4.4　調査の流れ

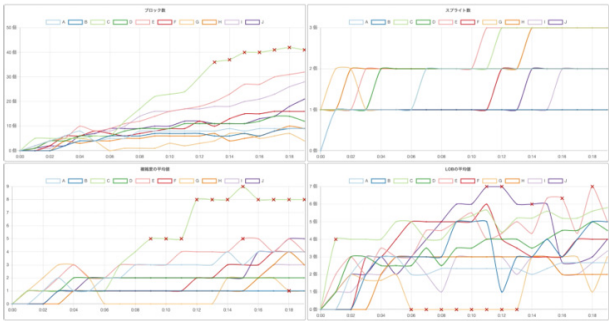
本調査ではリアルタイムに演習が行われていることを想定するため，異なる日時で取得した全学習者のコーディング履歴をすべて 0 分から時間進行に合わせ，T1 から T3 が閲覧する画面へ徐々に表示した．最初の 10 分が経過したときの教員用画面を**図 7** (a) に，20 分が経過したときの画面を図 7 (b) に示す．図 3 は 60 分経過した時点でのグラフである．そして，本調査は実際の授業で学習者指導を行っていることを想定してもらうこと，すなわち各学習者が課題を時間内に終了するように，指導内容を考えることを伝えた．そして課題の内容とその模範解答例を見せ，指導が必要な学習者とその理由を調べるように伝えた．

先述したとおり CRAVER はリアルタイムに学習者のソースコードを収集し，分析，可視化することが可能である．したがって，実際のプログラミング演習で使用する際は，教員は学習者らの課題の進捗を実時間で確認し，アドバイスやフィードバックを与えることができる．しかし本調査では，1 分ごとに各メトリクスのグラフが変わっていくことで，教員役被験者が特定のメトリクスの組合せや，

学習者に対する気づきを見逃す恐れがあると考えた．そのため，教員役被験者には図 7 のように 10 分の経過を 1 つのフェーズとし，フェーズが終わるごとに今までのログを見て指導が必要だと思う学習者はいるか，またその根拠となったメトリクスおよび学習者の状況をどのように考えて



(a) フェーズ 1 終了時のメトリクス一覧



(b) フェーズ 2 終了時のメトリクス一覧

図 7　経過時間によるメトリクスの遷移
Fig. 7　The transition of all metrics.

いるかなどについて質問した．フェーズを分けることで，相対的に遅れている学習者が，どのメトリクスを基に検出されるか，また教員役被験者のプログラミングに対する知識や教育経験によってメトリクスに変化が生じるのかを調査することができる．注目する学生や注目するメトリクスに違いが生じた場合，優先して注目すべきメトリクスの選定や，教員支援ツールに必要な機能の選定につながると考える．なお，実際の CRAVER の画面にはフェーズごとの区切り線は自動挿入されないが，本稿では次章の分析結果を明示するために，フェーズごとに灰色の太線を挿入した．CRAVER の UI や機能の改善は，評価実験の結果を基に，今後改善していきたいと考える．

5.　分析結果

5.1　RQ1：プログラミング経験が異なる場合，参照するメトリクスの数や種類に違いが生じるか？

T1 から T3 が各学習者に対し指導を行うべきと判断したフェーズと，そのときに参照したメトリクスを**表 2** に示す．すべてのメトリクスが参照されていた場合，ALL と示す．さらに，各フェーズで参照されたメトリクスの合計を**表 3** に，各教員が参照したメトリクスの個数を**表 4** に示す．

分析の結果，各メトリクスの中で最も参照されたものは BLN であった．また，BLN は全フェーズを通して参照される一方で，次に多く参照された SPN は主に後半に参照された．さらに，表 2 より，全体として単一のメトリクス

表 2　各教員が各学習者の指導の際に参照したメトリクス
Table 2　The metrics which each teacher referred when she/he pointed out a student.

	フェーズ 1			フェーズ 2			フェーズ 3			フェーズ 4			フェーズ 5			フェーズ 6		
	T1	T2	T3	T1	T2	T3	T1	T2	T3	T1	T2	T3	T1	T2	T3	T1	T2	T3
学習者 A				BLN						BLN SPN	BLN SPN	SPN	SPN					
学習者 B							BLN CCN		BLN SPN	BLN SPN	BLN SPN		CCN SPN	ALL	BLN LOB CCN	BLN	ALL	ALL
学習者 C		BLN			BLN CCN	LOB CCN		CCN SPN	SPN									
学習者 D										LOB CCN								
学習者 E						BLN LOB												
学習者 F																		
学習者 G	BLN LOB	BLN LOB	LOB CCN	BLN LOB	BLN LOB CCN								BLN SPN	SPN		BLN		
学習者 H																	CCN SPN	SPN
学習者 I														SPN				
学習者 J						BLN LOB CCN												

表 3　各フェーズで参照されたメトリクス（F：フェーズ）
Table 3　The metrics which teachers referred in each phase (F: phase).

	F1	F2	F3	F4	F5	F6	
BLN	3	6	2	4	3	4	22
LOB	3	5	0	1	2	2	13
CCN	1	4	2	1	3	3	14
SPN	0	0	3	5	6	4	18
Total	7	15	7	11	14	13	67

表 4　各教員が参照したメトリクス（割合（%））
Table 4　The metrics which each teacher referred (percentage (%)).

	T1	T2	T3	
BLN	9(45%)	8(31%)	5(24%)	22
LOB	3(15%)	4(16%)	6(29%)	13
CCN	3(15%)	6(23%)	5(24%)	14
SPN	5(25%)	8(31%)	5(24%)	18
Total	20	26	21	67

より複数メトリクスを参照し指導を判断することが多かった．また，表 4 より，参照したメトリクス数は T2 が最も多く，次に T3，そして T1 となった．T1 から T3 が各メトリクスを参照した割合は，T1 は BLN が最も多く，続いて SPN，そして LOB・CCN だった．T2 は BLN と SPN が同数で最も多く，続いて CCN，そして LOB となった．T3 は LOB の参照がほかに比べ 1 回多いものの，すべてのメトリクスをほぼ同じ割合で参照していた．

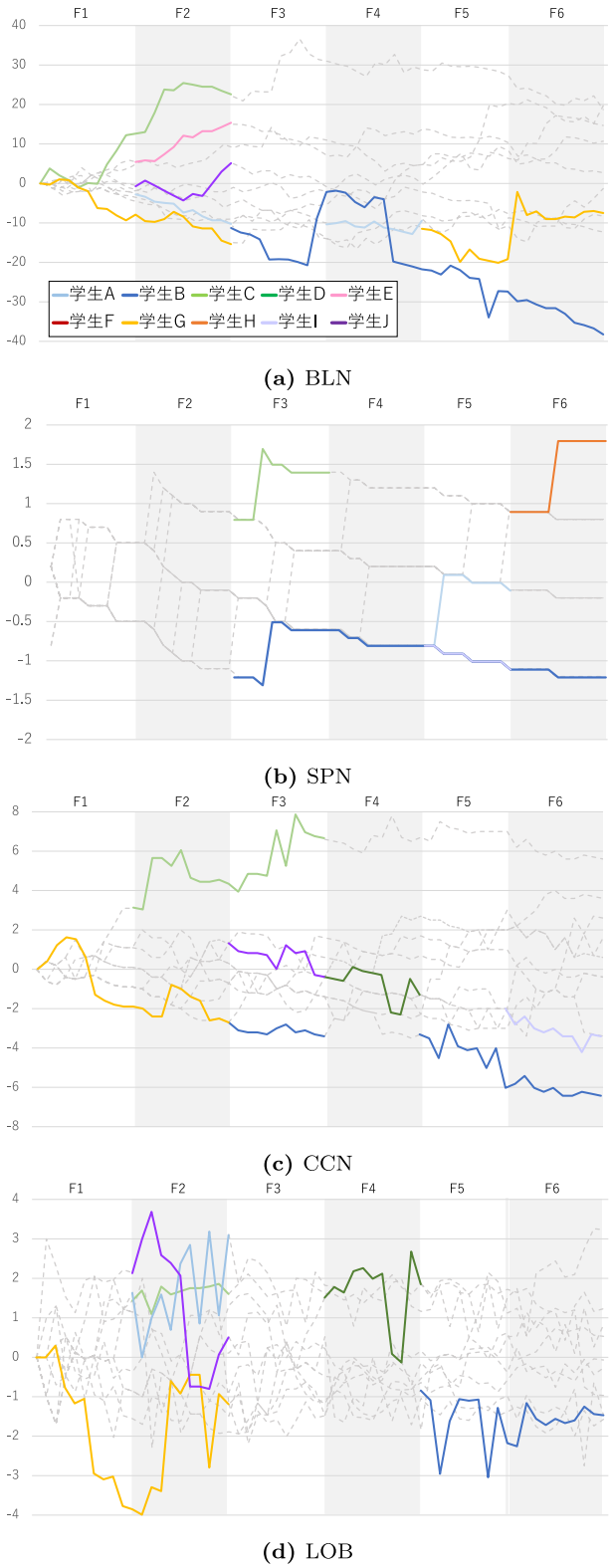
さらに，各メトリクスの特徴を調べるため，**図 8** に各メトリクスの平均値との差を示す．横軸は時間およびフェーズであり，縦軸がその時点での平均値との差を示す．また，特定のフェーズにおいて指導のために参照されたメトリクスは指導対象の学習者に応じた色の実線で，それ以外は灰色の破線で示す．フェーズ 6 において，T2 と T3 はすべてのメトリクスを参照し学習者 B の指導を判断していたが，図 8 より学習者 B はすべてのメトリクスが平均から大きく離れていたことが分かる．

5.2　RQ2：教育経験が異なる場合，指導内容に違いが生じるか？

T1 から T3 が各学習者に対し，どのメトリクスを参照しどのような指導が必要であると判断したか，アンケートの結果を**表 5** に示す．また，T1 から T3 に対し，実験終了後にメトリクス全体を見て気づいた点があるかヒアリングした結果を以下に示す．

T1
- BLN を一番見ていた．BLN を見て，気になったら原因を探るために他のメトリクスを確認した
- どの小問に着手しているか，進捗の把握には SPN を見た
- 実行したタイミングとかも表示できるとよさそう

T2
- BLN を意識していたけど，どの小問に着手しているかは SPN のほうが分かりやすい．ただし問題の設計次第だと思う．



(a) BLN

(b) SPN

(c) CCN

(d) LOB

図 8　各メトリクスの平均値との差
Fig. 8　The difference from average of each metrics in each phase.

表 5　各教員が各学習者の指導の際に参照したメトリクス
（*：学生のコーディング状況の実態と指摘内容が異なるもの）

Table 5　The detailed metrics which each teacher referred when she/he pointed out a student in each phase (*: There are some differences between the content which a teacher pointed out and the content a student's actual state).

フェーズ 1		フェーズ 4	
学習者 C	T2：LOB より過度に複雑な考えを行っているかもしれない*	学習者 A	T1：BLN と SPN が少ないから遅れてそう T2：同じ SPN の人と比べて BLN 少ない T3：SPN が 2 つで長く変化がない，（小問 2 で使う）クローンが理解できていないのかも
学習者 G	T1：最初なので Scratch の挙動を確かめているか課題を理解していない気がする T2：BLN，LOB 両方少ないので考え込んで進捗がなさそう T3：何かを試みて失敗したのではないかと思う	学習者 B	T1：BLN と SPN が少ないから遅れてそう T2：LOB が急に減った，SPN も少ないしうまくいっていなくってまとめて消したかも
		学習者 D	T1：LOB と CCN の増減が激しい．繰り返し処理か条件分岐で悩んでそう
フェーズ 2		フェーズ 5	
学習者 A	T1：進捗がなさそう	学習者 A	T1：SPN が変わっていない，明らかに進捗が悪く問題がある
学習者 C	T2：CCN と LOB から複雑な考えを行って行き詰っている気がする* T3：LOB と CCN がかなり高いので，どういう状況かを確認したい	学習者 B	T1：SPN と CCN の変化が少なすぎる，解決策が何も思いつかない状態かもしれない T2：全てのメトリクスが相対的に低いため進捗が悪いと思う，優先的に指導すべき T3：全てのメトリクスが停滞しているから手が止まっている気がする
学習者 E	T3：BLN が増えてるけど LOB が上下してるので，いくつかの実装パターンを試しては失敗してそう		
学習者 G	T1：BLN・LOB 両方停滞している，迷っている気がする T2：BLN・LOB・CCN 全体的に低い，進捗が滞ってそう	学習者 G	T1：SPN と LOB も少ない，進捗が悪いと思う T2：SPN が少ないので進捗が悪いと思う
学習者 J	T3：BLN も CCN もさほど高くないのに LOB が高いので，理由は分からないが長い関数を作っていると思う	学習者 I	T2：SPN が少ないので進捗が悪いと思う
フェーズ 3		フェーズ 6	
学習者 B	T1：BLN が低く，CCN が上がっている．条件分岐で悩んでいるのかな T3：LOB も SPN も上がったのが遅い．ずっと同じスプライトを作り込んで遅れているのかも．	学習者 B	T1：BLN が低いので進捗は悪いと思う T2：全体のメトリクスが停滞していて解決策が思い浮かんでいないと思う，アルゴリズムの説明が必要だ T3：全体のメトリクスの停滞が顕著なので諦めていると思う
学習者 C	T2：SPN と CCN が多い，できてないのに次のスプライトを置いてるかも．そうなら課題を一つずつやったほうがいいと指摘したい： T3：SPN は順調に増えているが，本当に順調なのか気になるので確認したい	学習者 G	T1：BLN の急上昇が気になるので確かめたい
		学習者 H	T2：SPN が多い，CCN も増えているしクローンを分かっていないためソースコードが複雑になっているかもしれない T3：SPN が多い，理由が分からないので確かめたい

- 模範解答の各メトリクスの遷移も見たい
- 現状の表示方法では 10 人以上になると分かりにくくなる

T3
- BLN と LOB の増減は一致すると思ったので一番確認していた．増減が一致しなかったときに他のメトリクスにも注目した
- 単純に手が止まっているのか，ブロックを探しているのかの区別がつかないので，それらが分かると嬉しい

表 5 より，フェーズ 1 の学習者 G のように，T1 から T3 全員が指導あるいは確認が必要だと指摘している場合でも，その原因の推測内容が異なる場合がある．また，指

導の内容には具体的に問題点を示すものだけではなく，学習者の進捗や，現状でのコーディングの方針を確認するべきだといった内容もあった．

本調査では学習者のログ収集と教員役被験者の実験は別の日に行ったため，実際に教員役被験者から学習者に対する指導やアドバイスは行われなかった．そのため，表 1 や学習者のログ取得実験後に行ったアンケート結果に加え，著者らが学習者の Scratch プログラム編集履歴を確認し，教員役被験者の指摘内容が学習者の実態に対し適していると思われるかを調査した．調査の結果，表 5 にアスタリスク “*” を付与した 2 点以外，そのときの学習者に対して有効であったと判断した．有効ではないと判断されたものとして，T2 が前半のフェーズで学習者 C に対し指摘した内容があげられる．学習者 C は Scratch プログラム経験が豊富であり，他の学習者より進捗が早く，作成したプログラムも複雑なものであった．そのため，T2 と T3 は学習者 C の各メトリクスが高いことに対し，複雑なプログラムを作ろうとしてコーディングに行き詰っていると判断した．そして，どのようにプログラムしようと考えているかや，小問を 1 つずつクリアしていくべきだなどを指導すべきだと述べた．

6. 考察

6.1 RQ1：プログラミング経験が異なる場合，参照するメトリクスの数や種類に違いが生じるか？

T1 から T3 が注目したメトリクスについて考察を行う．T2 と T3 はフェーズ 5 以降にすべてのメトリクスを参照し，指導が必要な学習者を見つけている．一方で，T1 も複数メトリクスを参照することが多かったものの，3 つ以上のメトリクスを参照することはなかった．例としてフェーズ 6 の学習者 B があげられる．ここでは，3 名の教員が全員，学習者 B は他学習者に比べ進捗が悪く，指導が必要であると述べたが，T1 は BLN から，T2 と T3 はすべてのメトリクスから判断した．

フェーズ 6 の学習者 B の編集履歴を確認したところ，実際に学習者 B は全体を通して進捗が悪く，小問も表 1 に示すとおり 5 つ中 1 つ目までしか解くことができなかった．すなわち，T1 と同様に，T2，T3 ともに，4 つのメトリクスから適切な指導，状況の把握を行うことができていた．フェーズ 5 の学習者に対しても，T1 は T2，T3 に比べ少ないメトリクスから状況を判断している．一方で，T2，T3 は 5.2 節において，これ以上学習者が増えるとデータの把握が困難になることを述べており，判断に用いるメトリクスを絞り込む必要があることを示す．

また，表 4 より T3 はすべてのメトリクスを均等に参照しているが，T1 と T2 は BLN と SPN の参照回数が多い．すなわち，T1 と T2 は教育経験を基に注目すべきメトリクスを取捨選択でき，さらに T1 は有するプログラミングに関する知識より，学習者の状況を把握するために最小限のメトリクスに着目することができた可能性がある．したがって，優先すべきメトリクスやメトリクス間の相関関係を調査することで，知識が少ない教員でも複数あるメトリクスに混乱することなく学習者指導が可能になると考えられる．

また，今回使用した課題は，SPN は最低 4 つで解けるものであり，SPN は 4 つのメトリクスの中で最も値の変化が少ないメトリクスである．しかしながら，値の変化が少ないにもかかわらず，SPN は BLN に続き頻繁に参照された．表 5 や 5.2 節の実験後のヒアリングより，今回の問題は小問ごとに必要な SPN が明確であったため，T1 と T2 は SPN を各学習者の進捗の把握として頻繁に利用していた．実際に，フェーズ全体を通して SPN の数が 1 個か 2 個と少なかった学習者 B は，図 1 からも進捗が悪く，フェーズ全体を通して指導が必要な状態であったため，T1 と T2 の SPN に対する判断は正しかったといえる．

一方，BLN や LOB の増減が激しい，あるいは停滞している学習者は，ブロックを追加したり削除したりするなど，Scratch の操作は行えているものの，アルゴリズムに関する試行錯誤がうまくいっていないため指導が必要な場合が多かった．実際に，フェーズ 2 の区間では，学習者 A の BLN の変化が小さく，停滞していることから T1 は指導タイミングであると判断していた．学習者 A のソースコードの編集履歴を確認したところ，実際にフェーズ 2 においてアルゴリズムの実装に必要なブロックを探したり付け替えたりする試行錯誤を行っていた．

以上より，Scratch に対する知識が異なる場合，注目するメトリクスや組合せは異なることが分かった．さらに，メトリクスの可視化によって各学習者のソースコードを精査せずとも，つまずきや試行錯誤の推測が可能となったと考える．本調査のようなプログラミングの経験が豊富な教員が注目するメトリクスやその理由の調査は，知識が少ない教員でも主体的に学習者指導を行い，演習を円滑に進められることにつながると考える．

6.2 RQ2：教育経験が異なる場合，指導内容に違いが生じるか？

表 5 より，T1 から T3 の指導内容や学習者の状況の把握に関する考察を述べる．

たとえば，フェーズ 4 の学習者 A と学習者 B に対し，T1 と T2 は同様のメトリクスを参照し指導が必要だと判断した．TA 経験がある T1 と T2 は，TA 経験のない T3 に比べ，学習者がどのような状況で質問を行うか，すなわち学習者が指導を必要とする状況に関する知見を有するといえる．フェーズ 4 において，学習者 A は図 3 に示すとおりすべてのグラフが相対的に低く，進捗が悪いうえにほとんど手が動いておらず，学習者 B は手は動いているものの

不必要な修正を行っており，両者ともメトリクスの遷移やつまずきの種類は異なるが，指導が必要な状態だった．実際，表 5 より T1，T2 ともに学習者 A，学習者 B に対する適切な状況の把握を行っている．さらに，T2 に関してはフェーズ 4 の学習者 B「試行錯誤したがうまくいっていなかったため，まとめてブロックを消したと思う」と詳しい状況も，学習者のソースコードを見ずとも主体的に把握することができていた．

一方で，T3 もフェーズ 4 において SPN の数から，とりかかっている小問を推測し，解答例を基に学習者の状況を推測していた．また，フェーズ 1 の学習者 G をはじめ，T1，T2 と同様の学習者を指導すべきと判断したタイミングも多い．これはプログラミング演習に対する教育経験や Scratch プログラミングの経験がほぼない場合でも，CRAVER を通してコーディング過程のメトリクスを可視化することで，主体的な指導につながったことを示唆する．しかしながら，T3 の指摘内容には，状況を確認したいといった内容や，進捗の停滞を指摘する内容が多かった．T1 や T2 は進捗の停滞に加え，T1 はフェーズ 3 の学習者 B へ「条件分岐か繰り返し処理で悩んでいる」，T2 はフェーズ 3 の学生 C へ「小問を 1 つずつ取り組むべき」といったように，より詳しい学習者の状況や，指導内容も述べることがあった．これは，実際に小学校などで Scratch プログラミングを教育する際，教員に対し，学習者の状況に対応した各メトリクスの変化や指導内容について，実例を基にした事前の説明が必要であることを意味する．

また，フェーズ 2 の学習者 E，学習者 J に関しては T3 のみ反応している．学習者 E，学習者 J の編集履歴について目視で確認したところ，T3 が想定した学習者の状況は正しかったものの，表 1 からも相対的に他の学習者より進捗やソースコードの質が悪いわけではなかった．実際に，フェーズ 2 以降，学習者 E，学習者 J は指導すべきだと判断されていない．表 5 より，他のフェーズにおいて，T2 は優先して指導すべき学習者を指摘していた（フェーズ 5 の学習者 B）．T1，T2 のヒアリングからも，両者は優先度を考えた指導を心がけており，プログラミング演習の TA を通した知見が反映されていたと考えられる．

以上より，プログラミング演習の TA 経験がなくても，コードメトリクスを可視化することで，各学習者の進捗の停滞を把握し指導することは可能であることが分かった．一方で，プログラミング演習 TA 経験の有無により，各学習者のコーディング状況の推測や，優先度をつけた学習者の指導に違いが生じた．また，表 1 より全体として相対的に遅れている学習者は，学習者 B，G，I であり，学習者 B と学習者 G は停滞やメトリクスの低さが顕著であったため指摘されることが多かったものの，学習者 I は特徴のあるメトリクスの遷移をしていないため，指摘はフェーズ 5 における 1 回のみだった．したがって，相対的に遅れている学習者をすべて検出できていたわけではないといえる．今後は実際に学習者が指導を必要としたタイミングについても，追加実験を通して調査を進めたい．そして，学習者が実際に指導を求めるタイミングおよびその理由を基に，各教員が参照したメトリクスの妥当性について，指導された学習者のコードメトリクスの変化も含めいっそう検討したいと考える．

6.3　各メトリクスの有効性

各メトリクスの有効性に関し，5 章，6.1 節，6.2 節の結果もふまえ考察を行う．

BLN に関し，T1，T2 が指摘するように，本実験で扱ったほどの課題の規模であれば，BLN で順調に進んでいるかを判断できた．よってコードメトリクスを可視化する際，BLN と他メトリクスを組み合わせるような機能や UI が求められる．同様に，どの小問に着手しているかを把握するためには，SPN が重要であることが分かった．一方，本実験で扱った問題は小問を順番に解くことを指定しており，また T2 は小問ごとに必要な SPN は明確であったため，SPN で着手している小問を把握できたと指摘している．すなわち，SPN を有効に扱うためには，課題に必要な SPN を明記するといった工夫が必要である．

また，LOB と CCN は単体で確認されることはなく，必ず他のメトリクスと同時に確認された．BLN と SPN が進捗の把握に使われる一方，LOB と CCN はソースコードの質を把握するために使われた．また，T3 は 5.2 節のヒアリングにおいて，BLN と LOB の関連性に注目していたが，図 3 より CCN の方が BLN にともない増減している．これは，スプリクトを複数作成し通信するといった試行錯誤より，条件分岐や繰り返し処理に関する試行錯誤が多かったことを示す．すなわち，LOB と CCN よりソースコードの質だけはなく，学習者全体の試行錯誤の傾向を知ることができると考えられる．

LOB は最も増減が激しかったため，急な増減だけでなく，停滞した際にも注目されることが多かった．また，LOB と CCN では平均値を採用したが，BLN でソースコードの編集を，SPN で必要なスプライト数を扱っているかを併せることで，学習者に対する指導の内容や進捗の把握に用いることが可能であった．たとえば，フェーズ 3 の学習者 B に対する T1 の「BLN が低く，CCN が上がっている．条件分岐で悩んでいるのかな」という指摘や，フェーズ 3 の学習者 B に対する T3 の「LOB も SPN も上がったのが遅い．ずっと同じスプライトを作り込んで遅れているのかも」という指摘があげられる．

以上より本研究で取り上げた 4 つのメトリクスは，学習者の状況や進捗の判断，また指導内容を導くにあたり有効であったといえる．今後は特定の学習者のグラフを選択した際に対応する他のメトリクスもハイライトするなど，メ

トリクスどうしの組合せを分かりやすく表示したり，学習者が課題を放置した際に課題未達成となる確率などを予測することで，優先して指導すべきメトリクスや学習者の特定なども組み合わせて支援したいと考える．

6.4　妥当性に関する議論

指導の必要がない場合のメトリクスの参照

本調査では指導が必要な学習者に焦点を絞り調査を行ったが，指導が必要でないと判断した理由にも各メトリクスの遷移が関係している可能性がある．例として，フェーズ3の学習者Cがあげられる．学習者Cは他の学習者より課題の進捗が早いことから，システム上では他の学習者よりBLNなどの値が極端に大きくなっているが，実際には順調に進んでおり指導が必要なタイミングではない．T2は，SPNとCCNの大きさから学習者Cに関して「1つの課題ができていないのに次の課題にも手を出している可能性がある」と述べ，T3も同様に，学習者CのSPNの大きさに注目したが「実際に課題が進んでいるかはわからない」と述べた．一方，T1は学習者Cに関してLOBにも注目し「学習者Cは全体としてメトリクスは大きいが，LOBが大きくなりすぎてはいないため順調に進んでいると思う」と述べた．

今回の調査では教育経験とプログラミング経験が異なる被験者に，メトリクスの可視化の結果から指導が必要な学習者を推測してもらった．しかし先述したとおり，指導が必要でないと考えた理由にも教育およびプログラミング経験が関係している可能性がある．今後は注目したメトリクスとその理由に関し，指導の有無とともに調査を行う必要がある．

被験者に関する内部妥当性

本実験では学習者・教員役被験者ともに情報科学系の大学生を採用したが，本研究で支援する対象は小学校におけるScratchプログラミング演習である．したがって，教員役被験者の指摘内容には他のプログラミングに関する知識が反映されている可能性がある．一方で，LINEみらい財団が2020年に小学校教員618名に行った調査 [14] によると，年代別にプログラミング教育へ不安を抱える教員は20代に最も多く，さらに66.5%の教員がプログラミング経験ありと答えたことが分かった．すなわち，プログラミングに関する経験は，必ずしもプログラミング教育に対する不安を払拭するものではなく，プログラミング経験のある場合でも，教員支援の必要性が示唆される．また，学習者となった被験者は情報科学系の大学生であったため，他のプログラミング言語の影響があったことが考えられる．たとえば，表1より学習者Bと学習者G，学習者Iは進捗が悪かったが，適切な変数・メッセージ名，矩形とひし形など異なる形は組み合わせられないことなど，プログラミングに関する基礎操作に関しては問題はなかった．実際に小学生が対象となった場合，変数の概念や，組み合わせらないブロックなど，基礎的なプログラミングの要素や，プログラミング環境の扱いなどのつまずきが生じる可能性がある．一方，上記はプログラミング経験が積まれることによって緩和されることも考えられ，回数を重ねたコードメトリクスの遷移も調査する必要がある．

7.　結論

本研究では小学校における教員支援を目的にScratchコードメトリクスの可視化に注目し，コードメトリクス可視化ツールCRABERを提案した．さらにプログラミングやプログラミング教育の知識・経験の差によって生じる，注目するメトリクスやメトリクスの解釈の違いを調査した．調査の結果，注目するメトリクスに違いはあったものの，プログラミングに関する知識・TAなどの教育経験が異なる被験者でも，可視化されたメトリクスを基に同様の学習者を指導対象として指摘することができた．すなわち，コードメトリクスの可視化は，プログラミング経験が浅い教員に対しても有用な情報であると考えられる．また，本提案ツールはデータの収集・可視化を自動で行うため，教員の人材不足やオンライン授業の支援にも貢献すると考える．

一方で，各教員役被験者が注目するメトリクスは異なっており，教育経験とプログラミング経験が及ぼした影響について調査を続ける必要がある．今後の課題として被験者や取得するログを増やし，時系列データ解析やクラスタリングなど定量的な分析から，各メトリクスの傾向や相関関係について調査したいと考える．

謝辞　本研究はJSPS科研費20K14101の助成を受けたものです．

参考文献

[1] 黒田昌克，森山　潤：小学校段階におけるプログラミング教育の実践に向けた教員の課題意識と研修ニーズとの関連性，日本教育工学会論文誌，Vol.34, No.3, pp.387–394 (2011).

[2] 米田浩崇，槇原絵里奈，小野景子：Scratchを用いたプログラミング授業における教員支援のためのリアルタイム分析可視化システムCRABERの提案，第27回 ソフトウェア工学の基礎ワークショップ (2020).

[3] Hermans, F. and Aivaloglou, E.: Do code smells hamper novice programming? A controlled experiment on Scratch programs, *Proc. 24th ICPC*, pp.1–10 (2016).

[4] Techapalokul, P. and Tilevich, E.: Code quality improvement for all: Automated refactoring for Scratch, *Proc. VL/HCC*, pp.117–125 (2019).

[5] 田中良樹，松澤芳昭，酒井三四郎：コーディングメトリクスを用いたブロック言語使用による構文エラー回避効果の計測，情報処理学会情報教育シンポジウム SSS2016 (2016).

[6] 田中良樹，松澤芳昭，木谷友哉，酒井三四郎：Hanabi：プログラミング教育改善のための横断的フィルタ機能を有するダッシュボード，情報処理学会研究報告 (2017).

[7] Busjahn, T., Schulte, C., Sharif, B., Simon, Begel,

A., Hansen, M., Bednarik, R., Orlov, P., Ihantola, P., Shchekotova, G. and Antropova, M.: Eye tracking in computing education, *Proc. ICER*, pp.3–10 (2014).
[8] Lahtinen, E., Ala-Mutka, K. and Järvinen, H.-M.: A study of the difficulties of novice programmers, *ACM SIGCSE Bulletin*, Vol.37, pp.14–18 (2005).
[9] Sarwar, M.M.S., Shahzad, S. and Ahmad, I: Cyclomatic complexity: The nesting problem, *Proc. ICDIM2013*, pp.274–279 (2013).
[10] Aivaloglou, E. and Hermans, F.: How kids code and how we know: An exploratory study on the Scratch repository, *Proc. ICER*, pp.53–61 (2016).
[11] Fontana, F.A., Zanoni, M., Marino, A. and Mäntylä, M.V.: Code Smell Detection: Towards a Machine Learning-Based Approach, *International Conference on Software Maintenance* (2013).
[12] 井垣　宏，齊藤　俊，井上亮文，中村亮太，楠本真二：プログラミング演習における進捗状況把握のためのコーディング過程可視化システム C3PV の提案，情報処理学会論文誌，Vol.54, No.1, pp.330–339 (2013).
[13] 藤原賢二，上村恭平，井垣　宏，吉田則裕，伏田享平，玉田春昭，楠本真二，飯田　元：スナップショットを用いたプログラミング演習における行き詰まり箇所の特定，コンピュータソフトウェア，Vol.35, No.1 (2018).
[14] 一般財団法人 LINE みらい財団：プログラミング教育必修化に関する調査報告書，入手先 ⟨https://d.line-scdn.net/stf/linecorp/ja/csr/LINEMiraiFoundation Report_20200730.pdf⟩（参照 2021-12-02）

推薦文

本論文は，小学校のプログラミング教育において広く使われている Scratch によるプログラミング演習において，学習者のコードメトリクスの変化を可視化することによって教員支援を行えるようにしたツールを提案している．Scratch はブロックベースのビジュアルプログラミング言語であるので，メトリクスはテキストベースの場合のコード行数などの代わりに，ブロック数やスプライト数を用いたものを提案している．教員役としてバックグラウンドが異なる大学生を被験者として行った実験においては，メトリクスの可視化により，指導に関する有用な情報を与えることができたと評価している．教育経験の浅さをサポートするにはどのようなメトリクスを活用すべきかなど，教育現場で役立てるためにはさらなる分析と評価が必要であるが，今後の研究の発展により，プログラミング教育を支援する重要な研究となることが期待できる．

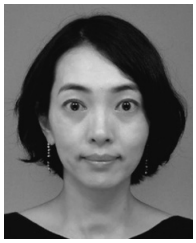
（情報処理学会理事・論文誌「教育とコンピュータ」アドバイザー　西田 知博）

槇原 絵里奈　（正会員）

2013 年大阪工業大学情報科学部情報システム学科卒業．2018 年奈良先端科学技術大学院大学博士後期課程修了．博士（工学）．同年より同志社大学理工学部インテリジェント情報工学科助教．現在に至る．ソフトウェア工学教育，プログラミング教育支援に関する研究に従事．電子情報通信学会，日本ソフトウェア科学会各会員．

米田 浩崇

2019 年同志社大学理工学部インテリジェント情報工学科卒業．2021 年同志社大学大学院理工学研究科修士課程修了．在学時，Scratch プログラミング支援の研究に従事．



小野 景子　（正会員）

2007 年同志社大学大学院工学研究科博士課程修了．博士（工学）．2009 年同志社大学研究開発推進機構省エネルギー照明システム研究センター特定任用研究員（助教）．2010 年龍谷大学理工学部電子情報学科助教．2014 年龍谷大学理工学部電子情報学科講師．2020 年同志社大学理工学部インテリジェント情報工学科准教授．現在に至る．並列処理，最適設計，進化計算等の研究に従事．IEEE，進化計算学会各会員．