Vol.2017-CE-139 No.4
2017/3/11

Scratchプログラムの可視化による類似度推定

岩科 智彩[1,a)]　森下 汐美[1,b)]　来住 伸子[1,c)]

概要：近年プログラミング教育が推進され，日本でも小学校での導入が検討されている．子供のプログラミング教育の方法として近年よく利用されているのが米国マサチューセッツ工科大学のメディアラボが開発したScratchである．Scratchのサイトではあるプログラムが引用された場合，その関係をリミックスツリーという図で表している．これでは引用関係は分かるが実際に引用されたもの同士の類似度は表されていないため，一箇所の変更を加えたもの，多数の変更を加えたものはリミックスツリー上では引用元のプロジェクトから同距離で表されている．本研究ではリミックスツリー上では表されていない類似性の数値化を行い，より分かりやすく表示させることを目指した．

キーワード：プログラミング教育，Scratch，リミックスツリー，類似度，可視化

Similarity estimation by visualization of Scratch program

Iwashina Chisae[1,a)]　Morishita Shiomi[1,b)]　Kishi Nobuko[1,c)]

Abstract: In recent years programming education has been promoted and introduction in elementary schools is being considered in Japan. Scratch developed by Media Lab of the US Massachusetts Institute of Technology is used as a method of children's programming education in recent years. When a certain program is cited on the Scratch site, the relation is represented by a diagram called a remix tree. Although the citation relations are understood, since the similarity between those actually cited is not represented, one with a change in one place and one with a lot of modifications taken from the citation source project on the remix tree is represented by the same distance. In this research, we aimed to display similarities that are not represented on the remix tree by numerical conversion and display them more clearly.

Keywords: Programming Education, Scratch, Remix Tree, Similarity, Visualization

1. はじめに

2006年に開発されたScratchという教育用プログラミング環境がある．Scratchとは米国マサチューセッツ工科大学のメディアラボが開発したフリーツールであり，キーボードでコードを打つのではなくパズルのようにブロックを組み合わせてプログラムを完成させる．近年，世界ではIT化が進み，物とインターネットを繋ぐIoTが次々と増えていく中エンジニア不足が叫ばれている．しかし世界的に見るとプログラミング分野では日本は遅れている．総務省平成26年度「教育・学習分野の情報化に係る国内外の動向と先進事例」では世界各国でプログラミング教育が活発化していることが分かる([1])．日本の旧学習指導要領の必修科目では，パソコンでワードやエクセルの使い方の教育しか行っていないが，イスラエルではコンピュータの使い方よりも原理やプログラミングを教える教育が行われている．そのため文部科学省が2016年5月19日に発表をした成長戦略素案では第4次産業革命を支える人材を強化するため情報活用能力を伸ばそうと，2020年度から小学校，2021年度に中学校，2022年度には高校でプログラミング

[1] 津田塾大学学芸学部情報科学科
〒187-0025 東京都小平市津田町2丁目1－1
Tsuda College, Faculty of Liberal Arts, Department of Computer Science
2-1-1 Tsuda-machi, Kodaira-shi, Tokyo 187-8577, Japan
[a)] g13908ic@gm.tsuda.ac.jp
[b)] g13924ms@gm.tsuda.ac.jp
[c)] kishi@tsuda.ac.jp

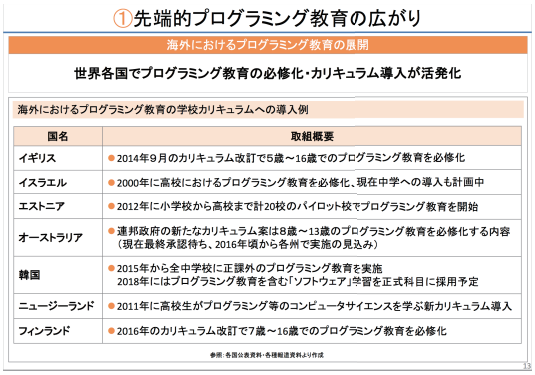

図 1　平成２６年度総務省 教育・学習分野の情報化に係る国内外の動向と先進事例 [1]

教育での必修化が予定されている．(図 1) それに先駆けて国内の学校では実際にプログラミングを行う授業も増えているという ([2])．品川区立京陽小学校では 2014 年度より Scratch が導入され各教科の中で活用されている。例えば理科の授業で実験結果をシミュレーションするプログラムの作成に Scratch が用いられているようだ．

Scratch では全世界のユーザーのうち日本でのユーザーは現状で 1%（図 2）である．Scratch 公式では利用者の数、年齢，使っているブロック等の統計データの公表があるが，Scratch 自体を題材にしている先行研究は日本では数が多くない．そこで義務教育化に伴い日本でも教育現場で Scratch を導入しやすくするため公開されているプログラムやデータを利用して解析を行う．

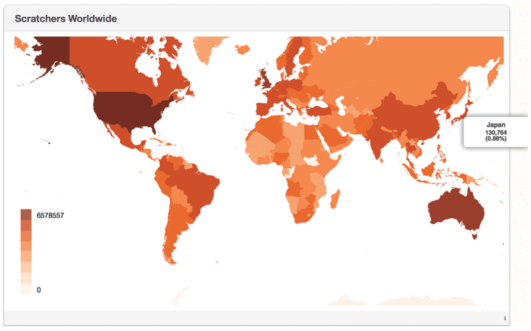

図 2　全世界の Scratch ユーザー　日本　130,764 人　0.88% [6]

2.　先行研究

2.1　Scratch ブロックの利用頻度別分布の視覚化

Scratch の公式サイトではさまざまなデータが公開されている．その一つが使用されたブロックのカテゴリ別分布である．カテゴリとはブロックの機能のことで，この分布では高い割合で使用されているブロックほど面積が大きい．(図 3) 項目をクリックするとそれぞれの機能の中の利用度の高いブロックが同じように面積が大きく表示される．(図 4) このデータでは Scratch 全体で使われているブロックの利用度の把握が可能である．しかしユーザー個人個人が作成したプログラムのブロックの利用度が分かるということではない．([3])

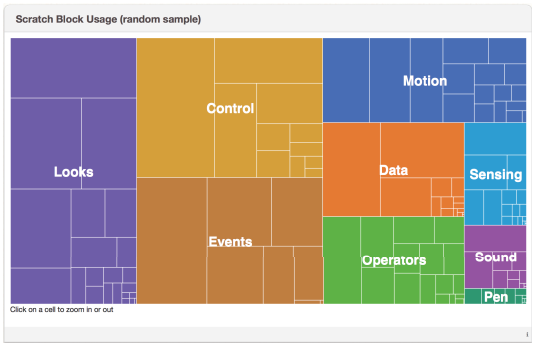

図 3　全ユーザーカテゴリ別利用頻度

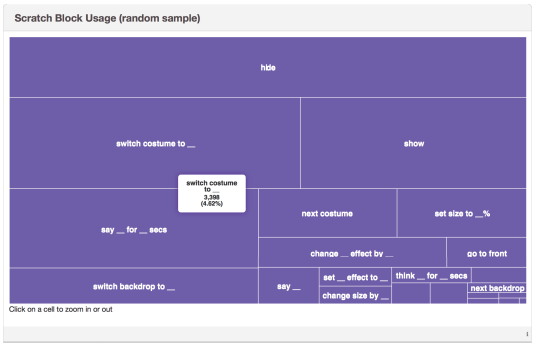

図 4　全ユーザー制御ブロック利用度　Looks（制御ブロック）をクリックした時

2.2　Scratch を用いた小学校プログラミング授業の実践

奈良女子大学附属小学校の 4 年生で Scratch を使った授業を行い，この授業を通して Scratch の導入について評価している．

研究方法として，「Scratch のコマンド探し」，「図形を描く」，「作品づくり 1」，「作品づくり 2」の 4 つのフェーズに分けて授業を進める．「作品づくり 1」は個人作業，「作品づくり 2」はグループ作業となっている．提出された作品は，ブロック数やスプライト数，使用されているブロックの機能別分布，さらに一つ一つのブロックの使用割合などを集計し，最後にブロックの理解度と利用率についてまとめている．

これらの評価を踏まえ，Scratch を授業に取り入れることで，条件分岐やキー入力の判別処理のような複雑なプロ

グラミングを使用して作品を作ることができた児童が 8 割を超え，プログラミング導入の目的を達成することができたとまとめている [4].

この研究は，実際に児童を対象に Scratch での授業を行い，このツールがプログラミング導入に適切かどうか評価している．評価の中で，ブロック数やスプライト数などいくつかのデータを収集し，児童がどれだけブロックを使用できているかを調査しているが，児童が作った作品同士の類似度を分かりやすく可視化はしていない．可視化を提案することで，児童ごとの進度をより分かりやすく示すことができ，授業内で役立てられるのではないかと期待している．

2.3 プログラム間の類似性の定量化手法

Scratch プログラム間の類似性を数字で表す方法を知るため，2011 年度の神戸市立工業高等専門学校電子工学科で Pascal 言語のプログラムを利用し類似性を定量化した研究を参考にする．この研究ではまず学校の課題で提出される学生のプログラムは、似た動きをするほとんど同じ内容のものが多いため，類似性を機械的に抽出ができるのではないかと仮定している．類似度の計算を，新しく提案したカーネル法を基にして算出し，算出結果をクラスタリングしどのような特徴を持つ手法であるかを調査している [5].

本研究と [5] の研究では題材であるプログラム形式は異なる．Scratch プログラムでは要素同士がリスト化されている JSON 形式であるため類似度の計算手法を別のものに変更する必要がある．また先行研究ではクラスタが可視化されたグラフがあるように，本研究でも可視化をすることで Scratch 公式サイトにて公表されているグラフィックを上回る結果を目標とする．

3. Scratch について

3.1 Scratch とは

Scratch とは，プログラミングを学ぶ子どもや初心者が，構文の書き方や使い方を覚えることなく結果を得ることができる，プログラミング言語学習環境である。2006 年に MIT メディアラボのミッチェル・レズニックが主導するライフロング・キンダーガーデン・グループによって開発された．プログラミングを可能な限り簡単に学べるよう，視覚表現でキャラクターを動かすことができ，ビジュアルプログラミングとも言われている?.

Scratch では自身でプログラムを作成するだけでなく，そのプログラムをコミュニティに公開したり，公開されているプログラムを引用（リミックス）することも可能である．

Scratch 上で使われている用語の中に，「プロジェクト」・「スクリプト」がある．「プロジェクト」は組み立てたブロックやキャラクター，実行画面全てのことを指し，その中のブロックの組み立て部分を「スクリプト」としている．本研究では分かりやすくするために，「プロジェクト」と「スクリプト」をまとめて「プログラム」として表現している．

3.2 リミックスツリー

Scratch では他のユーザーのプログラムを参照することができ，その際に引用することができる．引用関係はリミックスツリーに表示される。ページ下部の一番左に木のマークがあり，リンク先にはひとつの木が現れる．これはリミックスツリーというもので，このプログラムがどれだけのユーザーに引用されたのか，可視化されたものである．Scratch は簡単に他のユーザの作品をコピーすることができるが，それは誰かの作品を引用し自分にはないアイデアや方法を学びながら作っていくことを大事にしているからである．木構造になっているリミックスツリーは根元のプログラムを元として引用関係を表している (図 5)．さらに引用されたプログラムを引用した場合も反映される (図 6).

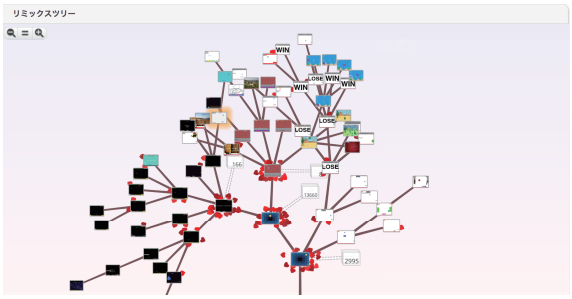

図 5　Pong Starter のリミックスツリー

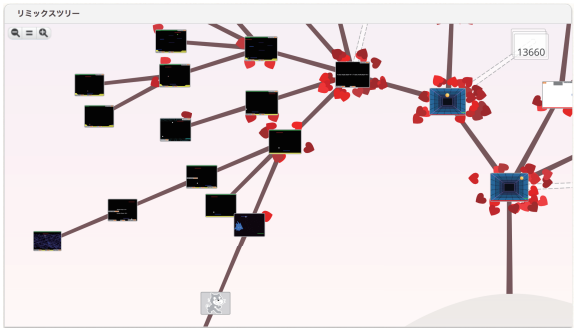

図 6　Pong Starter のリミックスツリー　図 5 の詳細

3.3 Scratch の現状

リミックスツリーでは元のプログラムと新たなプログラムは引用したという関係のみが表されており，全て同じ長さの線で結ばれているため，加えられた変更の大きさは表示されていない．たくさんのプログラムがリミックスされ

ている場合，どのプログラムがどのくらい似ているか，どのくらい変更されているかの数値を用いて類似度を表す可視化を実現することで，学ぶ側はレベルの把握，教える側には評価のしやすさが期待できる．

4. 研究するにあたって

本研究では収集データを元に類似度を数値化し，可視化を行ったのち類似性を評価する．この”類似度”と”類似性”の定義として，先行研究（[5]）を参考に，”類似度”は実際にデータを計算式に当てはめ，数値として表示させたものを指し，”類似性”は最終的に可視化した結果が正しいのか，主観的に判断したものを指すことにする．

5. 提案する手法

本研究では元プログラムと各プログラムをすべてのブロックが記録された 1 つの辞書型データを使用し，類似度推定を行う．類似度尺度は cos 類似度を用いる（[7]）．cos 類似度とは，ベクトル空間モデルにおいて文書同士を比較する際に用いられる類似度計算方法である．

収集できる Scratch プロジェクトのデータは json 形式（[8]）であるため，Python 環境のもと json モジュールをインポートすることでファイルを読み込む．Scratch では数多くのブロックが用意されており，どのブロックを使用するかによって，まったく異なるプログラムを作成することができるため，本研究ではブロックの種類ごとの個数を集計をした後，ソートしたものをベクターに直すことで類似度計算を行う．また，使用されているブロック数とスプライト (オブジェクト) 数も類似度を推定する際の要素として，使用する．

$$\cos(\vec{A},\vec{B}) = \frac{\vec{A}\cdot\vec{B}}{|\vec{A}||\vec{B}|} = \frac{\vec{A}}{|\vec{A}|}\cdot\frac{\vec{B}}{|\vec{B}|} = \frac{\sum_{i=1}^{|n|} A_i B_i}{\sqrt{\sum_{i=1}^{|n|} A_i^2}\cdot\sqrt{\sum_{i=1}^{|n|} B_i^2}} \quad (1)$$

6. 実験方法と仮説

本研究では 2 万弱の引用がされている”Pong Starter”（卓球ゲーム）(図 7) の大規模なリミックスツリー ([9]) と 200 弱の引用がされている”Maze Game”（迷路ゲーム）(図 8) の小規模なリミックスツリー（[10]）を利用し，元プログラムと各プログラムを比較をした．使用されているブロック・スプライトを数え上げ，元プログラムとの類似性を表す cos 類似度を数値化し，これらのデータを使用して可視化する．元のプログラムから引用されたものを 1 段目，以後 1 段目のプログラムから引用されたものを 2 段目，2 段目から引用されたものを 3 段目とし（図 9），各段ごとにグラフを作成する。リミックスツリーで表されているのは引用関係であるため，実際に 1 段目にあるものより 3 段目にあるものの方が元のプログラムと類似しているまた逆の結果が表れることが予測され．

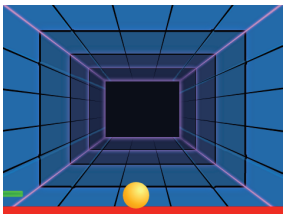

図 7 Pong Starter

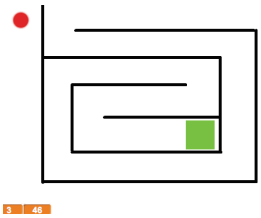

図 8 Maze Game

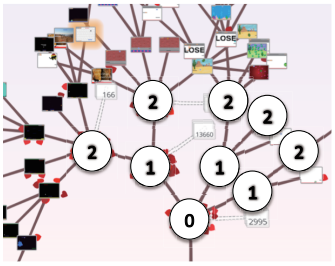

図 9 段数の数え方

7. 実験結果

出力されたデータを組み合わせてさまざまな可視化を行った．散布図では縦軸に cos 類似度の値，横軸に出力されたブロック数（スプライト数）を元のプログラムの数値で割った値をとり，”Pong Starter”の場合はさらに対数値をとる．グラフ上では（1,1）が最も類似しているプロジェクトであるため，この点から同距離のものを似ているものから順に青，赤，緑，紫，水色と区別してプロットをする．カラーマップでは各要素の個数を集計し，頻度の高いものから色別でプロットした．ただし，カラーバーはそれぞれのデータごとの最大値に合わせてある．

7.1 Pong Starter

7.1.1 ブロック数と cos 類似度の散布図

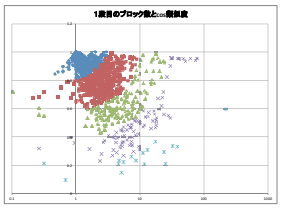

図 10 1 段目のグラフ

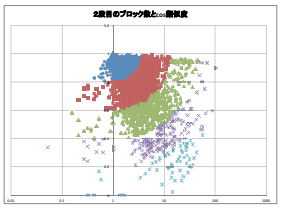

図 11 2 段目のグラフ

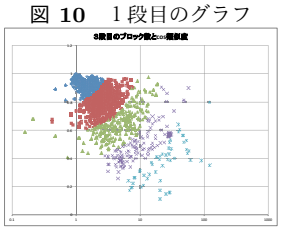

図 12 3 段目のグラフ

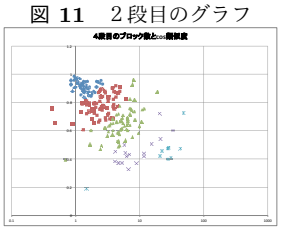

図 13 4 段目のグラフ

7.1.2 スプライト数と cos 類似度の散布図

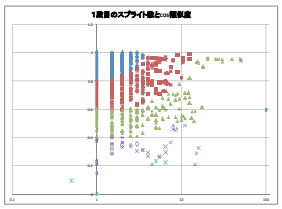

図 14 1段目のグラフ

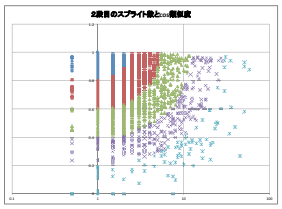

図 15 2段目のグラフ

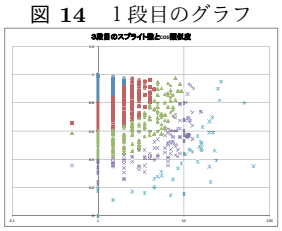

図 16 3段目のグラフ

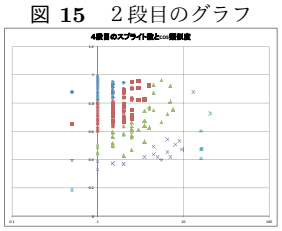

図 17 4段目のグラフ

7.1.3 ブロック数と cos 類似度のカラーマップ

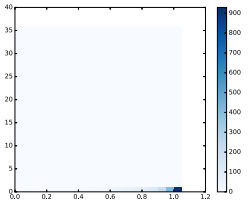

図 18 1段目のグラフ

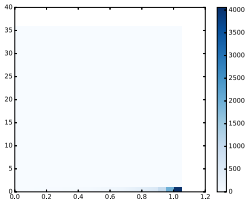

図 19 2段目のグラフ

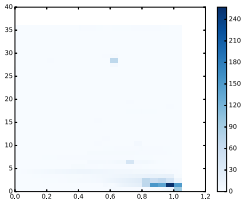

図 20 3段目のグラフ

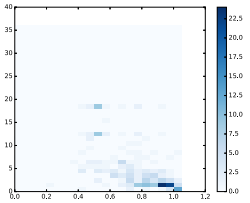

図 21 4段目のグラフ

7.1.4 スプライト数と cos 類似度のカラーマップ

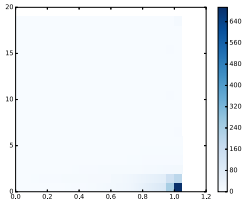

図 22 1段目のグラフ

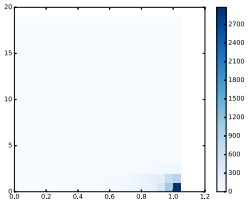

図 23 2段目のグラフ

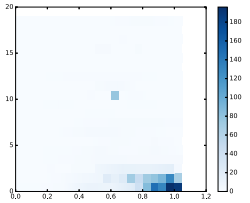

図 24 3段目のグラフ

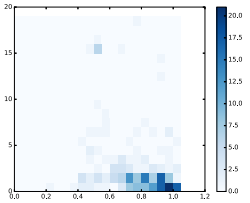

図 25 4段目のグラフ

7.2 Maze Game

7.2.1 ブロック数と cos 類似度の散布図

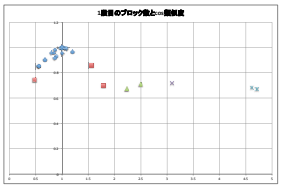

図 26 1段目のグラフ

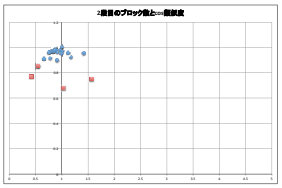

図 27 2段目のグラフ

7.2.2 スプライト数と cos 類似度の散布図

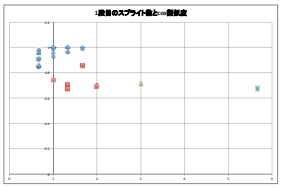

図 28 1段目のグラフ

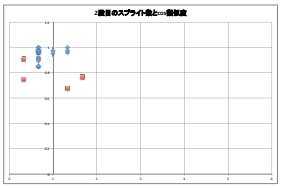

図 29 2段目のグラフ

7.2.3 ブロック数と cos 類似度のカラーマップ

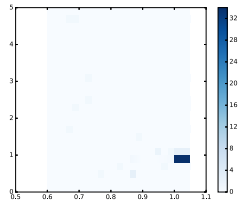

図 30 1段目のグラフ

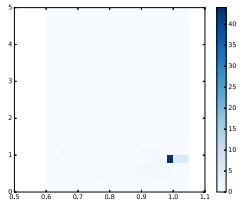

図 31 2段目のグラフ

7.2.4 スプライト数と cos 類似度のカラーマップ

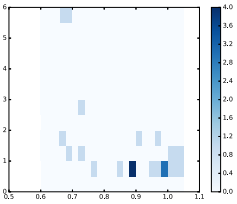

図 32 1 段目のグラフ

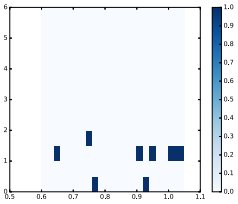

図 33 2 段目のグラフ

8. 評価

8.1 自己評価

8.1.1 Pong Starter の可視化

作成したグラフより，最も似ている数値のプログラム，最も離れている数値のプログラム，その中間のものを選び抜いた．最も似ているものは (1,1) 付近にあるものが元のプログラムに最も近いものと判断する．1 段目，2 段目，4 段目では (1,1) のプロジェクトが多く存在するため，その中から 2 つに絞る．これらのプログラムと元のプログラム（ID:10000036）を、操作・音・外装・プログラムの動き・ブロックの木構造・ゲーム終了時の 6 つの項目で比較し，可視化した結果が正しいかどうか検証した．

ブロック数と cos 類似度のグラフ，スプライト数と cos 類似度のグラフどちらにおいても同じような結果が出ており，それぞれの段で最も似ているもの・最も離れているもの・中間という条件で取り出したプログラムは，やはりよく似ている・違う・少し違う内容となっていた．よって，今回使用したブロック数・スプライト数・類似度はプログラム同士の類似度を図る上で重要な要素であったと言える．

また，1 段目から 4 段目まですべてを通して見てみると，2 段目以降にも元のプログラムと同じような内容のものが含まれていることが分かった．これは，リミックスツリーの根元（元のプログラム）から離れたプログラムも，元のプログラムと同じ場合が生じているということが言え，リミックスツリーではその事実までは可視化されていなかったということが伺える．本研究で行った可視化手法では，例え元のプログラムへいくつも遡るところに位置するプログラムであったとしても，ほぼ正確な類似関係が可視化できており，新たな可視化として提案できるのではないか。ここに本研究の新規性を見出すことができた．

8.1.2 Maze Game の可視化

作成したグラフより，最も似ている数値のプログラム，最も離れている数値のプログラムを選び抜いた．最も似ているものは (1,1) 付近にあるものが元のプログラムに最も近いものと判断する．これらのプログラムと元のプログラム（ID:2768574）を，操作・音・外装・プログラムの動き・ブロックの木構造・ゲーム終了時の 6 つの項目で比較し，可視化した結果が正しいかどうか検証した．

ブロック数・スプライト数どちらのグラフも段数に限らずばらつきが見られ，一回のリミックスだけで元プログラムに比べて改良されているプログラムも複数あることが分かった．全体的に大規模なリミックスツリーと同じような傾向が見られたが，プログラムの数が少ないため，類似度がかなり低いプログラムはあまり見られなかったが，逆にそのプログラムの特定はがしやすい可視化ができた．

また，1 段目より 2 段目の方が同じところに密集している理由としては，1 段目のあるプログラムをリミックスしたものが 2 段目の大半を占めていたためだと考えられる．

ブロック数とスプライト数を比較すると，ブロック数と cos 類似度の散布図の方がばらつきが見られ，プログラム同士の小さな変更点も可視化できている結果となった．これは，リミックスした際に変更が加えられやすいのはブロック数であるからだと考えられる．

8.2 教育者による評価の平均

8.2.1 評価方法

自己評価で抽出したプログラムを Scratch を教えたことのある教育関係者はどのように評価するのか，アンケートを実施した．方法としては，アンケートフォーラムを用意し回答を収集した．アンケート内容は， 自己評価の際に抽出したプログラムの中から Pong Starter を 8 個（質問 1.1 から 1.8），Maze Game を 8 個（質問 2.1 から 2.8）選択し，それぞれのプログラムと元プログラムを実際に動かし，似ているか似ていないかを 5 段階評価してもらう．全部で 16 個の評価対象があるが，Pong Starter4 個（質問 1.1 から 1.4）を必須項目とし，残り 12 個は任意での回答を求めた．アンケート回答者にはそれぞれのプログラムは何段目のものなのかは伏せ、並列した数字で表示した。また、どの観点で評価したかを操作・音・外装・プログラムの動き・ブロックの木構造・ゲーム終了時・その他の 7 つから該当するものすべてを選択してもらった．

8.2.2 評価結果

合計 14 名の回答を得ることができた．必須項目であった質問 1.1 から 1.4 と評価の観点は 14 名，質問 1.5 から 1.8 と 2.1 から 2.8 は 6 名から 8 名の回答を得た．

16 個の質問に対する回答から，評価結果の平均を点数（1〜5）*その評価を選んだ人数/全体の人数で求めたものと，cos 類似度をまとめた表を用意した．（表 1，表 2）

またこれらの表を散布図にしたものが，図 34 と図 35 である。

表 1　評価平均と cos 類似度（プログラム 1.1 から 1.8）

	平均	cos 類似度
プログラム 1.1	4.5	1
プログラム 1.2	2.28	0.59
プログラム 1.3	3.85	1
プログラム 1.4	2.5	0.87
プログラム 1.5	4.62	1
プログラム 1.6	1.62	0.35
プログラム 1.7	4.62	0.98
プログラム 1.8	3.5	0.72

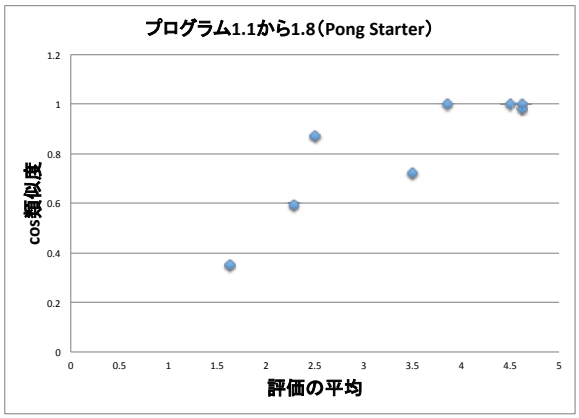

図 34　プログラム 1.1 から 1.8 の散布図

表 2　評価平均と cos 類似度（プログラム 2.1 から 2.8）

	平均	cos 類似度
プログラム 2.1	4.71	0.50
プログラム 2.2	4.14	0.42
プログラム 2.3	3.83	0.48
プログラム 2.4	4.57	0.48
プログラム 2.5	4.0	0.54
プログラム 2.6	3.0	0.44
プログラム 2.7	4.57	0.51
プログラム 2.8	4.57	0.48

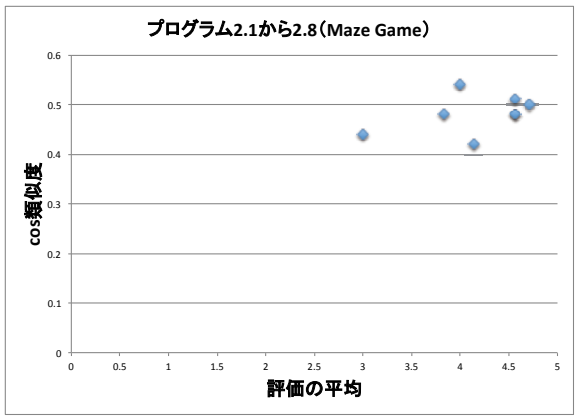

図 35　プログラム 2.1 から 2.8 の散布図

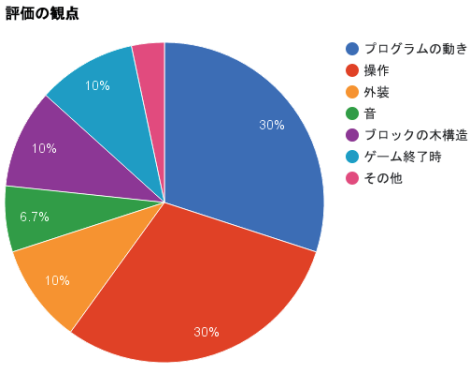

図 36　評価の観点

評価の平均は数値が高いほど似ていることを表し，cos 類似度は 1 に近いほど似ている．評価の結果，もともと可視化中の「似ているもの」,「似ていないもの」を条件にピックアップしたプログラムは，おおよそ数値的に「似ているもの」,「似ていないもの」として評価されていた．よって，可視化された結果（類似度）と類似性が関連していることが伺える．

プログラム 2.1 から 2.8 の散布図が右上の方に集中しているのは，リミックスツリーの規模が小さく 2 段目までしか広がりがなかったため，変更されている度合いも少なかったためだと考えられる．プログラムデータを収集している中で，1 段目のあるプログラムをリミックスしたものが 2 段目のプログラム全体の大半を占めていること分かった．同じプログラムをリミックスしたプログラムはやはり内容として似ている傾向にあることが分かった．

また，評価の観点は，「プログラムの動き」・「操作」が全体の半分以上を占めた。その他には違う役割のスプライトが増えているかどうか，という回答が得られた．（図 36）この結果から，類似性の可視化をするにあたり，ブロック数・スプライト数・cos 類似度を要素としたことは妥当だったのではないかと考えられる．

9.　まとめと今後の展望

9.1　まとめ

実験結果による可視化（類似度）と自己評価、アンケートによる評価（類似性）関連があることが分かった．これにより，本研究で提案した可視化は，教育者が考える類似性を表せていたのではないかと考える．そして，さまざまな可視化を試みた結果，ブロック数と cos 類似度の散布図が一番類似度を表せていることが分かった．また，学校で使用することを想定して行った小規模リミックスツリーの可視化も，新たな教育支援ツールとして期待出来る結果と

なったのではないか．具体的には，教師が作成した模範回答を生徒がリミックスし，自分の変更を加えた後，教師が成績をつける際，模範解答との相違点を一目で判定できる可視化が表示されるツールが存在すれば，教育現場におけるプログラミング教育の発展に貢献できると考えられる．

9.2 今後の展望

今回は Pong Starter と Maze Game の 2 つのゲームデータの収集しか行えなかったため，2 種類のデータに限られてしまった．Scratch サイトで公開されている他多数のプログラムについてさらに分析を行い，プログラムごとの傾向を探りたい．また分析して得た cos 類似度とブロック数，スプライト数の 3 種類を全て使用した 3 次元可視化をすることで，誤差が減りより精度の高い可視化を提案できるのではないかと考えている．そして，本研究の 3 要素以外にも，ブロックの組み方や並び順などプログラムの木構造も考慮した新たな類似度尺度を採用し，可視化を試みたいと考えている．

謝辞 本研究を進めるにあたり，指導教員の来住伸子先生には日々丁寧かつ熱心なご指導を賜りました．そして青山学院大学客員教授の阿部和広先生，助教授の吉田葵先生にも貴重な時間を割いていただきアドバイスをいただきました．また，評価の際は Scratch 教育関係者の方々よりアンケートのご協力をいただきました。ここに感謝の意を表します．

参考文献

[1] 教育・学習分野の情報化に係る国内外の動向と先進事例, 総務省 http://www.kknews.co.jp/maruti/news/2014/0804_4a.html
[2] 小学校でプログラミング　アイデアを形にする力をつける, 教育課程新聞 http://www.kknews.co.jp/maruti/news/2014/0804_4a.html (2014 年 8 月 4 日)
[3] Scratch Statistcs - Imagine, Program, Share, MIT Media Lab https://scratch.mit.edu/statistics/
[4] 「Scratch を用いた小学校プログラミング授業の実践～小学生を対象としたプログラミング教育の再考～」森秀樹・杉澤学・張海・前迫孝憲, https://www.jstage.jst.go.jp/article/jjet/34/4/34_KJ00007142887/_pdf（2011 年）
[5] 「プログラム間の類似性の定量化手法」, 小田悠介・若林茂, http://www.kobe-kosen.ac.jp/activity/publication/kiyou/Kiyou12/Data/Vol51Paper103-108.pdf (2013 年)
[6] Scratch - Imagine, Program, Share, MIT Media Lab https://scratch.mit.edu/
[7] コサイン類似度 http://www.cse.kyoto-su.ac.jp/ g0846020/keywords/cosinSimilarity.html
[8] 【Python 入門】JSON 形式データの扱い方 (2016 年 12 月 12 日), Morio http://qiita.com/Morio/items/7538a939cc441367070d
[9] Remix tree for Pong Starter https://scratch.mit.edu/projects/10000036/remixtree/
[10] Remix tree for Maze Game https://scratch.mit.edu/projects/2768574/remixtree/