Mensurando o desenvolvimento do Pensamento Computacional por meio de Mapas Auto-Organizáveis: Comparação de métricas de complexidade de Software com Dr. Scratch e CT-Test.

Rafael Pereira Ribeiro[1], Alexandra Souza[1], Thiago Barcelos[1], Leandro A. Silva[2]

[1] Instituto Federal de Educação, Ciência e Tecnologia de São Paulo (IFSP)
Av. Salgado Filho, 3501, Guarulhos, SP, Brasil – CEP 07115-000.

[2] Universidade Presbiteriana Mackenzie
Rua da Consolação, 930, São Paulo, SP, Brasil – CEP 01302-907.

r.ribeiro@aluno.ifsp.edu.br; {tsbarcelos,alexandra.souza}@ifsp.edu.br;
leandroaugusto.silva@mackenzie.br

Abstract. *In this paper we perform a visual and exploratory analysis on the data generated by the Dr. Scratch tool from codes produced by students in digital game workshops using Scratch environment. Our findings point to a weak linear relationship between Dr. Scratch's rubric and cyclomatic complexity, since codes with low cyclomatic complexity were developed by students who had better scores in the CT-Test. This may indicate that the use of more advanced Scratch features could be related to the advance of the student's acquisition of Computational Thinking skills.*

Resumo. *Neste artigo realizamos uma análise visual e exploratória dos dados gerados pela ferramenta Dr. Scratch a partir de códigos produzidos por alunos em oficinas de produção de jogos digitais utilizando o ambiente Scratch. Nossas descobertas apontam uma fraca relação linear entre a rubrica do Dr. Scratch e a complexidade ciclomática, visto que códigos com complexidade ciclomática baixa foram desenvolvidos por alunos que obtiveram melhor pontuação no CT-Test, o que pode indicar que o uso dos recursos mais avançados do ambiente Scratch pode estar relacionado com o progresso do aluno na aquisição de competências do Pensamento Computacional.*

1. Introdução

Dentre as estratégias de ensino adotadas para o desenvolvimento do Pensamento Computacional está a construção de jogos digitais, sendo o Scratch um ambiente bastante utilizado (Barcelos et al., 2014). Por outro lado, existe ainda uma carência em relação ao acompanhamento do progresso do aluno na aquisição de competências do Pensamento Computacional. Em geral, utiliza-se de técnicas de pesquisa de paradigma qualitativo, tais como a observação etnográfica e as entrevistas, bem como testes e questionários específicos (Araujo; Andrade; Guerrero, 2016). Ainda neste contexto, iniciativas como da CSTA (2011) buscam assegurar o desenvolvimento de um conjunto mínimo de competências e habilidades por meio da definição de currículos de referência. A exemplo disso, Román-González (2015) define um questionário baseado

DOI: 10.5753/cbie.wcbie.2018.609

em uma rubrica educacional para a avaliação pessoal das competências do Pensamento Computacional em um contexto específico. Paralelamente, Boe et al., (2013) e Moreno-León et al., (2015, 2016) apresentam mecanismos automatizados, denominados respectivamente Hairball e Dr. Scratch, para a análise do código de artefatos produzidos no Scratch baseado em rubricas previamente estabelecidas.

O Dr. Scratch tem gerado muitos trabalhos de pesquisa que utilizam os dados produzidos pela ferramenta como os de Barcelos et al. (2017), para mensurar o desenvolvimento do Pensamento Computacional através de dados provenientes de uma oficina de jogos digitais por meio de Mapas Auto-Organizáveis. Pesquisa semelhante é feita por Moreno-León et al. (2017) em que são analisados 250 projetos extraídos da plataforma on-line do Scratch com o objetivo de identificar diferenças e realizar agrupamento hierárquico de dados. Chang et al. (2018), por outro lado, utilizam as métricas Halstead para análise de complexidade de programas Scratch. Moreno-León et al. (2016) também analisam 100 jogos escolhidos aleatoriamente em uma base de dados extraída da plataforma on-line do Scratch, a partir do cálculo de duas métricas de complexidade de código provenientes da Engenharia de Software: complexidade ciclomática de McCabe e métricas de Halstead. Neste último estudo, os autores apresentam uma correlação linear entre a pontuação geral do Dr. Scratch do artefato de código gerado com as métricas já citadas. No entanto, ainda se identifica uma lacuna na literatura em relação à validade de tais análises comparativas quando aplicadas a atividades didáticas mais estruturadas sequencialmente e com maior duração.

A partir do contexto exposto, este trabalho tem o objetivo de realizar uma análise exploratória dos dados gerados pela ferramenta Dr. Scratch a partir de códigos produzidos por alunos em uma oficina de jogos digitais com duração média de quatro meses, utilizando uma rede neural no formato dos Mapas Auto-Organizáveis de Kohonen ou *Self-Organizing Maps* (Kohonen, 2013). O SOM, como o algoritmo é também conhecido, funciona como uma ferramenta de apoio para agrupamento de dados, permitindo a partir de uma análise exploratória e visual identificar a correlação existente entre as métricas geradas no Dr. Scratch e a complexidade ciclomática dos artefatos produzidos. Complementarmente, o resultado analisado por meio do SOM será relacionado com o resultado da pontuação obtida na aplicação do questionário proveniente do trabalho de Román-González (2015), denominada CT-Test, implementado na plataforma Kahoot.

Além da introdução, que visa contextualizar a proposta e apresentar o objetivo desta pesquisa, na Seção 2 uma introdução das principais referências é apresentada. Na Seção 3 é definida a metodologia adotada neste experimento. Na Seção 4 são apresentados e discutidos os principais resultados e, por fim, na Seção 5 são apresentadas as conclusões.

2. Referencial Teórico

2.1. Análise automatizada de código Scratch

Conforme proposto por Brennan e Resnick (2012) a aquisição de competências do Pensamento Computacional pode ser evidenciada pela complexidade dos artefatos computacionais produzidos por alunos, dentre eles está o código de programas. Diante deste cenário várias iniciativas para efetuar a análise de códigos produzidos no ambiente

Scratch vêm sendo desenvolvidas e avaliadas. Wolz et al. (2011) apresentaram o Scrape, um sistema para análise e apresentação visual do tipo e frequência de utilização das estruturas de programação em um programa Scratch, Boe et al. (2013) desenvolveram o Hairball, um sistema baseado em *plug-ins* com o objetivo de extrair diversas características por meio da análise do código Scratch, tais como o emprego de práticas não recomendadas (*bad smells*) e Moreno-León et al. (2015) propuseram o Dr. Scratch, uma ferramenta web que analisa os códigos Scratch, utilizando parte dos *plug-ins* do Hairball e definindo uma rubrica que associa o nível de complexidade e frequência das estruturas de programação utilizadas a sete categorias conceituais relacionadas ao Pensamento Computacional.

2.2. Complexidade Ciclomática

Segundo Pressman (1995) Complexidade Ciclomática (CC) é uma métrica de software que determina de forma quantitativa a complexidade de um programa. Baseada na teoria dos grafos, avalia o número de caminhos independentes existentes no código, ou seja, qualquer fluxo do programa que introduza pelo menos um novo conjunto de instruções de processamento ou ainda novas condições (McCabe, 1976).

Uma das aplicações da complexidade ciclomática está na determinação da dificuldade de realização de testes, mas também pode ser utilizada para determinar o quão difícil será realizar a manutenção de um código, pois conforme constatado em Kafura e Reddy (1987) quanto maior a complexidade do sistema, maior será o tempo gasto com sua manutenção.

2.3. Kahoot! e a aplicação de jogos em sala de aula

Kahoot é uma plataforma de aprendizagem gratuita baseada em jogos para utilização de professores e alunos em sala de aula, com o objetivo de promover um ambiente de disputa engajando o aprendizado dos estudantes (Kahoot, 2017).

Alguns trabalhos defendem que a utilização dessas ferramentas desperta a motivação e o interesse dos alunos pelos conteúdos passados em sala de aula. Trabalhos como Romio e Paiva (2017) propõem que o Kahoot gerou mais entusiasmo na turma e apresentou mais benefícios na aprendizagem devido às características que o jogo apresenta. Segundo Wang e Lieberoth (2016) a ferramenta relatou no estudo apresentado pelos autores estados como concentração, engajamento, divertimento, aprendizado e motivação. No contexto deste experimento o Kahoot foi utilizado como plataforma de oferecimento do Teste de Pensamento Computacional (CT-Test) de Román-González (2015) buscando promover o engajamento dos estudantes na participação do teste e ao mesmo tempo tornar a atividade divertida ao aluno.

2.4. Mapas auto-organizáveis

Na década de 1980, Teuvo Kohonen desenvolveu um estudo que deu origem a uma das mais famosas arquiteturas de Redes Neurais Artificiais (RNA), chamada de Mapas Auto-organizáveis (do inglês, Self Organizing-maps - SOM), se trata de um tipo de RNA interconectada e não supervisionada que resolve tarefas de agrupamento de dados, visualização e abstração (Kohonen, 2013). Estes estudos de Kohonen são baseados no princípio da formação de mapas de unidades cerebrais (neurônios) que, de forma auto

organizada, estabelecem uma ordenação espacial que permite representar a informação (Silva et al. 2016).

A utilização do SOM foi proposta para a resolução da tarefa de agrupamento de dados e se justifica pela necessidade de exploração visual da correlação dos dados obtidos pela ferramenta Dr. Scratch, a partir da análise de códigos Scratch, com as variáveis referentes a complexidade ciclomática e a pontuação resultante da aplicação do CT-Test dos estudantes em lógica de programação.

3. Metodologia

Diferentemente do argumento apresentando em Moreno-León et al. (2016) onde pode-se observar uma relação linear entre a rubrica do Dr. Scratch e métricas de complexidade de código, as hipóteses que este estudo busca endereçar são: (1) Entre duas soluções para um mesmo problema proposto na Oficina de Produção de Jogos Digitais, a solução que melhor desenvolve as características do Pensamento Computacional segundo métrica do Dr. Scratch apresenta a menor complexidade ciclomática; (2) Os alunos com melhor desenvolvimento de competências e habilidades relacionadas ao Pensamento Computacional, conforme mensurado por meio do CT-Test, produziram um código de menor complexidade ciclomática na oficina de programação de jogos digitais.

Esta seção descreve o projeto, planejamento e execução dos passos necessários para a validação das hipóteses de pesquisa. O estudo foi estruturado em duas partes: a primeira envolve a classificação, por meio de mapas auto-organizáveis, dos dados das sete características do Dr. Scratch gerados por meio de códigos produzidos em oferecimentos da Oficina de Produção de Jogos Digitais nos anos de 2013 e 2017 (Barcelos et al., 2017), bem como na edição que ocorreu em 2018, e verificar a relação existente entre os grupos formados e a complexidade ciclomática dos códigos produzidos. Na segunda parte do estudo a classificação gerada será também relacionada com os dados coletados na aplicação da avaliação CT-Test (Román-González, 2015) implementada por meio da plataforma Kahoot, sendo que esta parte do estudo foi realizada somente com as turmas de 2017 e 2018.

3.1. Participantes do estudo e intervenção didática

Para a primeira parte do estudo foram coletados os dados de três oferecimentos da Oficina a alunos com perfis semelhantes. Em 2013, a Oficina foi oferecida em 12 semanas para alunos do curso técnico concomitante ao ensino médio do Instituto Federal de São Paulo, matriculados na disciplina de Lógica de Programação. Naquela ocasião, 40 alunos de 15 a 17 anos participaram das atividades. No primeiro semestre dos anos de 2017 e 2018 a Oficina foi novamente oferecida na disciplina de Lógica de Programação, mas agora para os alunos do curso técnico integrado ao ensino médio. Novamente o oferecimento aconteceu para 40 participantes em 2017 e 41 em 2018 matriculados no primeiro ano do ensino médio, com idades entre 14 e 15 anos. O tempo disponível para aplicação das atividades foi ligeiramente maior em ambas ocasiões, chegando a 18 semanas.

A Oficina de Produção de Jogos Digitais é uma sequência de atividades de construção de jogos digitais no ambiente Scratch que visa desenvolver em estudantes do

ensino médio competências e habilidades do Pensamento Computacional relacionadas à programação de computadores. Orientada por princípios do construcionismo e da Aprendizagem Baseada em Problemas, cada atividade da Oficina visa desenvolver as competências de forma progressiva, solicitando a mobilização de competências já desenvolvidas enquanto novos tópicos são explorados pelos alunos. Os resultados da Oficina em termos da motivação dos participantes e articulação com competências de outras áreas do conhecimento já foram discutidas anteriormente em (Barcelos et al., 2014). Todas as atividades têm como fator motivador a construção de uma ou mais mecânicas de um jogo digital e ao final são desenvolvidos entre sete e nove jogos completos, dependendo de fatores operacionais relacionados ao ano de oferecimento da Oficina.

Para a segunda parte do estudo o experimento consistiu em recrutar os participantes da Oficina em 2017 e 2018, no momento em que desenvolviam o projeto final, para participar da dinâmica que envolveu a aplicação do CT-Test usando a plataforma Kahoot. Responderam o questionário 68 alunos já participantes da oficina.

3.2. Variáveis de estudo e coleta de dados

As variáveis dependentes para a validação do estudo foram:

- *Métrica do Dr. Scratch*, utilizado para mensurar a aquisição de competências do Pensamento Computacional por meio da análise do código fonte produzido na oficina. É atribuída uma pontuação de 1 a 3 pontos para cada uma das sete características definidas pelas métrica (abstração, paralelismo, lógica, sincronização, controle de fluxo, interação com o usuário e representação de dados), bem como uma pontuação geral (somatória dos pontos adquiridos em cada característica, que por sua vez varia de 0 a 21 pontos).
- *Complexidade Ciclomática,* calculada a partir da análise do código fonte dos jogos produzidos e utilizada como um indicador da qualidade do código desenvolvido pelos estudantes.
- *Qualificação do aluno no CT-Test,* onde foi coletada a quantidade de respostas corretas no teste (0 a 28) para validar se o desempenho condiz com os resultados obtidos pelo mesmo aluno ao longo da oficina de programação de jogos conforme mensurado pelas demais variáveis.

3.3. Procedimentos do Estudo

Para a primeira parte do estudo foram coletados os dados dos jogos Scratch nos três oferecimentos já mencionados da Oficina de Produção de Jogos Digitais, criando uma base com 875 projetos desenvolvidos que foram então submetidos ao processamento do *plugin mastery*[1] do projeto Dr. Scratch para extrair as métricas relacionadas às sete características do Pensamento Computacional propostas pela ferramenta, bem como a pontuação geral obtida para cada artefato gerado. Em seguida foi calculada a CC de cada programa utilizando um plugin do Hairball de Boe et al. (2013) chamado *metrics*[2], um analisador estático de código para Scratch escrito em Python.

[1] https://github.com/jemole/hairball/blob/master/hairball/plugins/mastery.py
[2] https://github.com/jemole/hairball/blob/master/hairball/plugins/metrics.py

Como um primeiro procedimento, foi calculada a regressão linear entre a pontuação geral do Dr. Scratch obtida por um jogo e valor da complexidade ciclomática do código. A plotagem dos dados é apresentada na Figura 1, onde é possível verificar uma fraca relação linear dos dados (r^2=0,0907), contrariando inicialmente o que foi apresentado em Moreno-León et al. (2016).

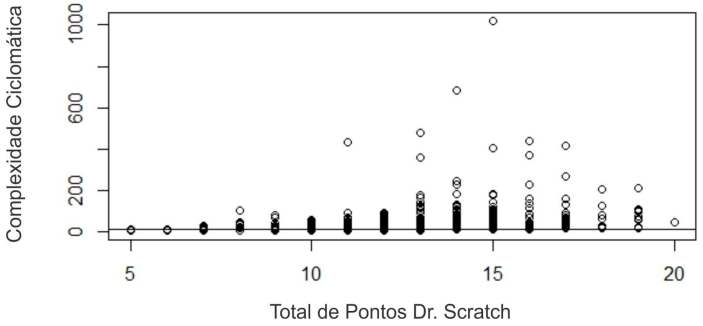

Figura 1. Plot com melhor linha em r^2 =0,0907 (875 artefatos das oficinas de 2013, 2017 e 2018)

Para a segunda etapa do estudo foi realizada a tradução de todas as perguntas e imagens contidas no CT-Test 2.0 para a língua portuguesa e criação do questionário na plataforma Kahoot!. Finalmente, com as métricas do Dr. Scratch, CC e CT-Test foi então efetuada a análise de agrupamento e da correlação existente entre os dados por intermédio da linguagem R e o seu pacote Kohonen (Supervised and Unsupervised Self-Organising Maps) com a função `som` parametrizada conforme apresentado na Tabela 3.

Tabela 3. Parametrização da função SOM utilizada para análise

Parâmetro	Configuração
Dados	Base de 875 registros
Grid	Dimensão 6x5 com com *lattice* Regular
Vizinhança	Retangular
Épocas	1000
Taxa de Aprendizado	Inicial 0,05 com redução linear até 0,01

O treinamento da rede SOM foi realizado com a pontuação de cada programa nos níveis de competência das sete categorias definidas na rubrica de Moreno-León et al. (2015). O propósito da análise foi verificar a influência do nível de competência mensurado em cada categoria na composição de cada nodo e a consequente distribuição topológica dos elementos da base de dados sobre o mapa. Os demais dados de cada registro (pontuação geral Dr. Scratch, complexidade ciclomática, pontuação CT-Test, jogo, aluno e ano de oferecimento da oficina) não fizeram parte do agrupamento e foram analisados para a identificação de tendências.

4. Resultados

No resultado do treinamento da rede SOM, apresentado na Figura 3, é possível observar o agrupamento das características do Pensamento Computacional e a clusterização dos níveis observados, identificados através da hierarquia de cluster,

contendo três clusters, o que vai ao encontro do observado em Moreno-León et al. (2017) no que se refere aos níveis de evolução do Pensamento Computacional (básico, intermediário e avançado). Na parte inferior direita do mapa é possível observar o agrupamento de características dos alunos que desenvolveram todas as habilidades do Pensamento Computacional seguindo a métrica do Dr. Scratch. Os nodos que agrupam essas características são indicados pelo cluster representado na cor laranja.

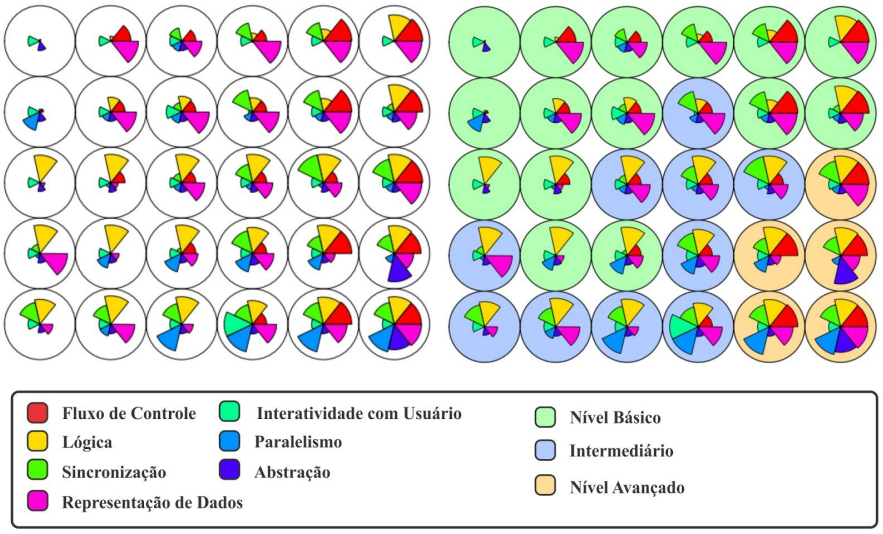

Figura 3. Características do Pensamento Computacional conforme métrica do Dr. Scratch e clusterização dos níveis de desenvolvimento

Para todas as análises realizadas a partir desse mapa SOM foi adotada a seguinte estratégia: identificar o nodo em que predominam os jogos iniciais e finais de cada uma das oficinas, extrair desse os artefatos com a menor e a maior complexidade ciclomática, e rotulá-los com o identificador de artefato do Aluno A (alta) e Aluno B (baixa) para os jogos iniciais e de Aluno C (alta) e Aluno D (baixa) para os jogos finais. A partir deste agrupamento foi possível observar a distribuição da complexidade ciclomática dos jogos desenvolvidos no ano de 2013, 2017 e 2018, conforme apresentado na figura 4.

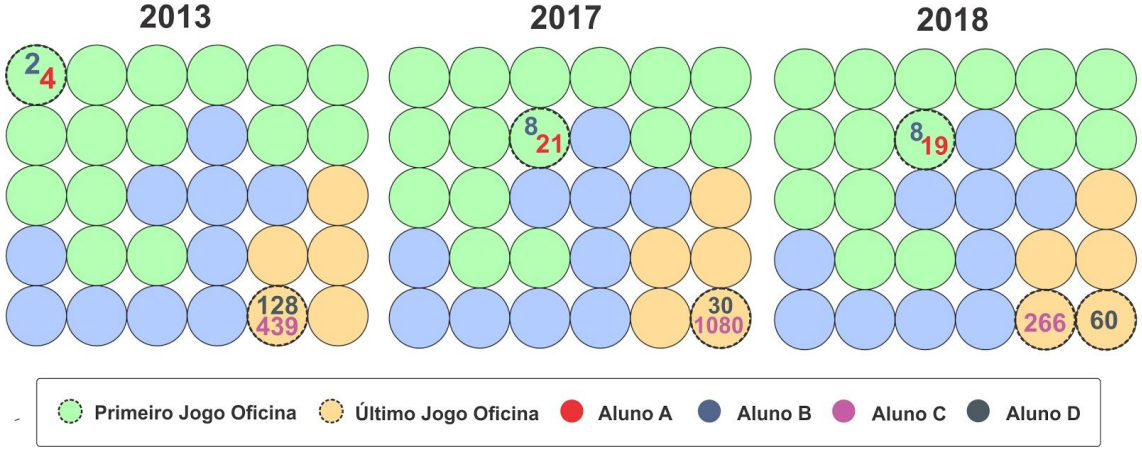

Figura 4. CC das Oficinas de Jogos Digitais de 2013, 2017 e 2018

No mapa que representa a oficina de 2013, no nodo em que predomina o jogo

inicial verificamos uma complexidade ciclomática baixa desenvolvida no jogo do aluno A e B. Isso se deve ao fato que o primeiro jogo da oficina não exige muitas habilidades do Pensamento Computacional e de fato é um jogo simples de ser resolvido o que justifica a CC ser baixa. Para o projeto final foram analisados os jogos do aluno C e D e a diferença da CC nesta observação é que o aluno C implementou o jogo (Pacman) com menos recursos que o aluno D. Uma observação relevante sobre o contexto é que a versão do Scratch utilizada em 2013 ainda não tinha suporte ao recurso de "clonagem" de *sprites*, replicando totalmente sua funcionalidade, o que levou os alunos a duplicarem os *sprites* da aplicação para obter recursos como os fantasmas do jogo, duplicando por consequência o código do programa.

No mapa que representa a oficina de 2017 é possível identificar a mesma característica para o primeiro jogo desenvolvido pelos alunos A e B, onde a CC baixa se justifica pela complexidade exigida pelo jogo. No caso do jogo final é possível identificar que o aluno C, apesar de ter desenvolvido um jogo que denota o desenvolvimento de todas as competências do Pensamento Computacional mensuradas pela rubrica do Dr. Scratch, construiu um código mais complexo segundo a CC, ao contrário do aluno D que também desenvolveu as referidas habilidades porém construiu um programa com CC mais baixa. Uma análise dos artefatos produzidos por ambos os alunos indica um elevado número de replicação do código de *sprites* no programa do aluno C enquanto que o aluno D usou o recurso de clone o que possibilitou a escrita de um jogo com menos linhas de código, e consequentemente, com menos caminhos de decisão, característica que influencia diretamente no cálculo da complexidade ciclomática. Para a oficina de jogos de 2018, representado pelo terceiro mapa da esquerda para a direita da figura 4, foi possível notar uma consistência em relação ao ano de 2017 dado que as mesmas características já descritas foram observadas ao analisar os artefatos dos jogos inicial e final.

Quando incluímos o resultado do CT-Test para as turmas de 2017 e 2018, conforme demonstrado na figura 5, é possível perceber uma correlação entre a CC do jogo desenvolvido pelo aluno e seu desempenho no CT-Test. Na análise realizada, apresentada na tabela 4, os alunos que desenvolveram os jogos finais com menor complexidade ciclomática, representados como alunos D, obtiveram melhor pontuação no CT-Test em comparação aos que produziram códigos de complexidade ciclomática alta, representados como alunos C.

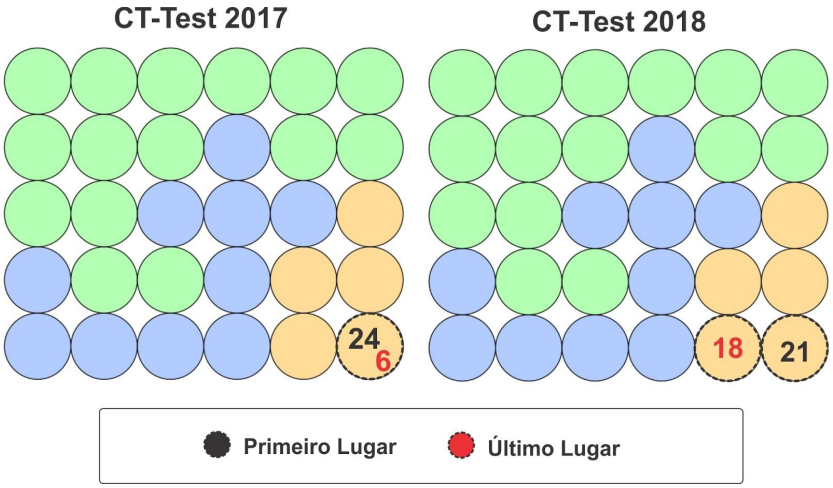

Figura 5. Ranking do CT-Test nos anos de 2017 e 2018.

5. Conclusão

Este artigo apresentou uma validação preliminar do uso de mapas auto-organizáveis como técnica para correlacionar a pontuação das características do Pensamento Computacional, a partir da rubrica proposta por Moreno-León et al. (2015) e implementada no software Dr. Scratch, dos artefatos produzidos nas Oficinas de Produção de Jogos Digitais com seus respectivos valores de complexidade ciclomática, além dos resultados extraídos do CT-Test, realizado por meio do Kahoot, aplicado aos alunos participantes da referida oficina.

Foi identificado que existe uma correlação entre as pontuações das sete dimensões da rubrica do Dr. Scratch, dos artefatos produzidos no ambiente Scratch com a complexidade ciclomática, mas diferentemente do que foi apresentado em Moreno-León et al. (2016), se trata de uma uma fraca relação linear dos dados devido ao fato da complexidade ciclomática se reduzir quando o aluno desenvolveu o código utilizando recursos mais avançados de abstração como, por exemplo, clones. Neste cenário o aspecto apresentado no trabalho de Robles et al. (2017) que indica uma limitação relacionada às dimensões avaliadas pela rubrica do Dr. Scratch no que se refere a mitigação de ações da cópia de código é válida, porém o emprego da complexidade ciclomática deve ser observada como uma solução a ser estudada.

Também foi verificado que os alunos que desenvolveram códigos para os projetos finais com complexidade ciclomática baixa também tiveram melhor pontuação no CT-Test, o que pode indicar que o uso dos recursos mais avançados do ambiente Scratch pode estar relacionado com o progresso do aluno na aquisição de competências do Pensamento Computacional.

Algumas limitações à validade do estudo devem ser destacadas, como o tamanho da amostra, possível competição dos alunos durante a resposta do CT-Test, assim como a ausência de uma aplicação do teste para os mesmos alunos antes da participação da oficina de jogos. No entanto, entende-se que a contribuição do trabalho é relevante por se tratar da avaliação dos resultados de uma oficina de média duração, em contraposição a trabalhos anteriores, e que uma relação não-usual entre a complexidade ciclomática e rubricas previamente utilizadas para a avaliação do Pensamento Computacional foi identificada.

Referências

Araujo, A. L. S. O.; Andrade, W. L.; Guerrero. Um Mapeamento Sistemático sobre a Avaliação do Pensamento Computacional no Brasil. In: 2º *WAlgProg*, 2016, Uberlândia: SBC, 2016.

Barcelos, T. et al. Informal HCI: Fixing Playability Issues As A Strategy To Improve The Skills Of Novice Programmers. IEEE Latin America Transactions, v. 12, n. 1, p. 29–35, jan. 2014

Barcelos, T., et al. Mensurando o desenvolvimento do pensamento computacional por meio de mapas auto-organizáveis: um estudo preliminar em uma oficina de jogos digitais. VI Congresso Brasileiro de Informática na Educação (CBIE 2017), pages

932–941.

Boe, B. et al. Hairball: lint-inspired static analysis of scratch projects. 2013, Denver, Colorado, USA. Proceedings of the 44th SIGCSE. Denver, Colorado, USA: ACM, 2013. p. 215–220

Brennan, K.; Resnick, M. New frameworks for studying and assessing the development of computational thinking. In: Proceedings of AERA 2012. Vancouver: American Educational Research Association, 2012.

Chang, Z., Song, R e Sun, Y., (2018) Validating Halstead Metrics for Scratch Program using Process Data, in *IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*

Kafura D. e Reddy, G.R., The Use of Software Complexity Metrics in Software Maintenance, in *IEEE Transactions on Software Engineering*, vol. SE-13, no. 3, pp. 335-343, March 1987.

Kahoot! (2017). What is Kahoot!? https://kahoot.com/what-is-kahoot/. [Online; Acesso em: 02 nov. 2017].

Kohonen, T. (2013). Essentials of the self-organizing map. Neural Netw., 37:52–65.

McCabe, T.J. (1976). A complexity measure. IEEE Trans. on Software Engineering. SE-2(4), p. 308-320

Moreno-León, J., Robles, G. e Román-González, M. (2015). Dr. Scratch: Automatic Analysis of Scratch Projects to Assess and Foster Computational Thinking. RED-Revista de Educación a Distancia.

Moreno-León, J., Robles, G. e Román-González, M. (2016). Comparing Computational Thinking Development Assessment Scores with Software Complexity Metrics.

Moreno-León, J., Robles G., e Román-González, M. (2017). Towards Data-Driven Learning Paths to Develop Computational Thinking with Scratch, in *IEEE Transactions on Emerging Topics in Computing*.

Robles, G., Moreno-León, J., Aivaloglou, E. e Hermans F., Software clones in scratch projects: on the presence of copy-and-paste in computational thinking learning, *2017 IEEE 11th International Workshop on Software Clones (IWSC)*, Klagenfurt, 2017.

Romio, T. and Paiva, S. C. M. (2017). Kahoot e goconqr: uso de jogos educacionais para o ensino da matemática. Scientia cum Industria, 5(2):90–94.

Román-González, M. (2015). Computational thinking test: Design guidelines and content validation. EDULEARN15, pages 2436–2444.

The CSTA Standards Task Force. CSTA K-12 Computer Science Standards. New York: ACM Computer Science Teachers Association, 2011.

Wang, A. I. and Lieberoth, A. (2016). The effect of points and audio on concentration, engagement, enjoyment, learning, motivation, and classroom dynamics using kahoot! In European Conference on Games Based Learning, page 738. Academic Conferences International Limited. ACM technical symposium on Computer science education