ANÁLISE AUTOMÁTICA DE ATIVIDADES DE INTRODUÇÃO À PROGRAMAÇÃO COM O SCRATCH

Franciely Alves de Souza[1]- 0000-0002-2203-9948

Taciana Pontual Falcão[1] - 0000-0003-2775-4913

Rafael Ferreira Mello[1,2] - 0000-0003-3548-9670

[1]Universidade Federal Rural de Pernambuco (UFRPE) - Recife - PE - Brasil
[2]CESAR School - Recife - PE - Brasil

francielyalves835@gmail.com, {taciana.pontual, rafael.mello}@ufrpe.br

Resumo. O objetivo do presente artigo é propor visualizações automáticas de avaliações de atividades feitas no Scratch, a fim de dar suporte para os professores no acompanhamento do aprendizado dos alunos. Foram geradas visualizações gráficas a partir de atividades de programação básica realizadas no Scratch com alunos do 2º ao 5º ano do ensino fundamental. As atividades foram avaliadas automaticamente através de uma versão modificada da ferramenta Dr. Scratch, e foram geradas visualizações gráficas na ferramenta online Data Studio. Os resultados apresentam as análises das atividades, onde o professor pode visualizar o desenvolvimento do aprendizado do aluno, e analisar quais são as possíveis dificuldades apresentadas por cada aluno ou turma, e consequentemente adaptar as aulas a fim de supri-las.

Palavras-Chave: Scratch, Programação em Blocos, Dr. Scratch, Pensamento Computacional.

AUTOMATIC ANALYSIS OF INTRODUCTION ACTIVITIES TO PROGRAMMING WITH SCRATCH

Abstract: The purpose of this article is to propose automatic visualizations of assessments of activities made in Scratch, in order to support teachers in monitoring student learning. Graphic visualizations were generated from basic programming activities carried out in Scratch with students from the 2nd to the 5th year of elementary school. The activities were automatically evaluated using a modified version of Dr. Scratch, and graphical views were generated in the Data Studio online tool. The results present the analysis of the activities, where the teacher can visualize the development of the student's learning, analyze what are the possible difficulties presented by each student or class, and consequently adapt the classes in order to address them.

Keywords: Scratch, Block Programming, Dr. Scratch, Computational Thinking.

1. Introdução

A tecnologia está levando a grandes mudanças na nossa forma de nos comunicarmos e nos relacionarmos com os outros, e cada vez mais no modo como aprendemos (Bates, 2017). Dentre as inovações tecnológicas que regem o século XXI, está a popularização da habilidade de desenvolver programas computacionais por meio de algoritmos escritos em linguagens de programação. A área de programação é um dos pilares da

profissionalização em informática e de grande importância nas matrizes curriculares (Bini e Koscianski, 2009).

Desde a década de 90, uma das motivações para a introdução da computação ou programação na Educação Básica consiste na demanda crescente por profissionais com a capacidade de entender e produzir novas tecnologias digitais (Manyika, 2017). Entretanto, aprender programação é um processo complexo, que exige competências como raciocínio lógico e abstração (Bittencourt, 2013). A aprendizagem de programação é um processo que requer muita dedicação e prática para obter bons resultados, e na educação básica é fundamental aplicar metodologias que possam engajar os alunos, facilitando a compreensão dos conceitos abordados e tornando as aulas mais atrativas.

De acordo com Marcon Júnior e Boniatti (2015), é necessária a criação ou a utilização de metodologias diferenciadas que apresentem benefícios ao ensino-aprendizagem de programação, em especial, no desenvolvimento do raciocínio lógico. Neste sentido, é importante considerar a habilidade de Pensamento Computacional (PC), vislumbrada por estudiosos como uma ferramenta cognitiva que se tornará indispensável no futuro. Wing (2006) caracteriza o PC com um conjunto de competências e habilidades atreladas à Ciência da Computação, habilidades estas que os estudantes deveriam incorporar desde os primeiros anos escolares. Dentro deste contexto, Araújo, Andrade e Guerreiro (2016) mapearam a literatura brasileira a respeito do PC, a partir disso constataram que a abordagem mais empregada a fim de estimular o PC é a programação.

Neste contexto, as linguagens de programação em blocos foram desenvolvidas para estudantes iniciantes, com intuito de facilitar o aprendizado inicial de lógica de programação (Marji, 2014). Em particular, a ferramenta de programação em blocos Scratch[1], desenvolvida no Media Lab do Instituto de Tecnologia de Massachusetts, se tornou extremamente popular e tem sido usada com crianças mundialmente, inclusive fomentando a criatividade. Moretti (2019) afirma que a linguagem Scratch se constitui numa maneira de se desenvolver as habilidades características do PC.

Por outro lado, ainda não existe um consenso sobre como avaliar o desenvolvimento do PC dos estudantes a partir de soluções expressas através de linguagens de programação em blocos. Para códigos desenvolvidos no Scratch, existe o Dr. Scratch[2], uma aplicação web de código livre, que consiste em analisar projetos Scratch a partir de sua URL (acrônimo para *Uniform Resource Locator*), de forma automática, ajudando os professores a identificarem pontos fracos como códigos mortos e variáveis não utilizadas; e também pontos fortes como a utilização de sete conceitos relacionados ao PC[3] (Moreno-Leon et al., 2016). Ao avaliar um projeto, serão informados os conceitos presentes na programação e a pontuação alcançada, numa escala de 0 a 3.

Porém, o Dr. Scratch não disponibiliza nenhum tipo de visualização que possa orientar o professor sobre a evolução dos alunos de acordo com a pontuação de cada conceito; e nenhum modo de extração dos resultados das atividades avaliadas (essas pontuações só podem ser extraídas de forma manual). Após avaliar uma atividade, o professor precisa observar os resultados e adicioná-los manualmente em algum documento externo, a fim de gerar visualizações para acompanhar o desempenho de cada aluno ou turma. A motivação deste trabalho é suprir as dificuldades de avaliação e

[1] scratch.mit.edu/
[2] http://www.drscratch.org/
[3] Acesso a tabela de conceitos avaliados pela ferramenta Dr. Scratch:
Conceitos avaliados pela ferramenta Dr. Scratch.pdf

análise de códigos desenvolvidos no Scratch. A fim de proporcionar para os professores uma maneira de acompanhar o desenvolvimento dos alunos nessas programações, propomos visualizações a partir de análise automática de avaliações de atividades.

Este trabalho está organizado da seguinte maneira: na Seção 2 são apresentados os trabalhos relacionados e como este trabalho se diferencia dos demais. O processo metodológico é descrito na Seção 3. O processo de avaliação com o Dr. Scratch é descrito na Seção 4. Os resultados obtidos são discutidos na Seção 5. Por fim, as considerações finais e trabalhos futuros são apresentados na Seção 6.

2. Trabalhos Relacionados

Eloy (2019) utiliza *learning analytics* na avaliação de projetos no Scratch e aplicação de questionários para coleta de dados, na intenção de verificar se os projetos estão sendo relevantes para o aprendizado de programação dos estudantes. Para a avaliação ocorrer, uma URL do projeto Scratch precisa ser utilizada, ou um arquivo Scratch exportado. As informações dos projetos são coletadas por uma API (acrônimo para *Application Programming Interface*) de código Python, que armazena todos os dados na base de dados e em seguida gera as visualizações necessárias para a análise. De acordo com o autor, análises automatizadas das atividades podem contribuir para uma melhor compreensão das necessidades de aprendizagem, além de prover uma representação gráfica do processo de aprendizagem percorrido pelos alunos. Desta forma, o trabalho deixa claro que as avaliações automáticas podem contribuir fortemente para uma análise mais robusta, onde pode-se obter e armazenar registros de aprendizados de cada aluno.

O trabalho de Araújo et al. (2020) também se assemelha a este trabalho por avaliar atividades Scratch, tendo a intenção de verificar se habilidades do PC são desenvolvidas e acompanhar o progresso dos alunos nas programações. Para o processo de avaliação, foi construída uma ontologia de nome “OntoScratch”, que é dividida em classes que representam o projeto Scratch e classes que representam a avaliação do PC a ser realizada. A cada novo bloco adicionado na programação, a ontologia Scratch cresce, pois possibilita a criação de novas classes. Para verificar a eficiência da ontologia na avaliação de habilidades do PC, as mesmas atividades que foram avaliadas na OntoScratch foram submetidas a uma avaliação no Dr. Scratch, o que resultou em dados equivalentes. Consequentemente, a representação obtida pela OntoScratch se mostrou eficaz, e também permite armazenar de maneira estruturada o percurso dos alunos na realização dos projetos no Scratch.

O presente trabalho difere dos demais por apresentar uma versão do Dr. Scratch modificada, instalada localmente, em que os dados extraídos são automaticamente adicionados a um banco de dados, onde a análise das atividades de cada aluno fica armazenada, podendo ser acessada a qualquer momento, não sendo necessário que a mesma atividade seja reavaliada caso o professor necessite visualizar estes dados em outro momento. Desta forma, o professor também obtém a liberdade de extrair os dados em formato CSV (*Comma-Separated-Values*), na intenção de gerar uma visualização gráfica, o que até então nenhum trabalho do nosso conhecimento utilizou. Sendo assim, pode-se constatar pelos trabalhos relacionados que, embora utilizem a mesma ferramenta de análise, os meios para obter e visualizar os dados não abordaram as mesmas técnicas.

3. Metodologia

Os dados utilizados neste trabalho foram coletados no período de maio a dezembro de 2020, em aulas de introdução à programação que ocorreram remotamente devido ao

período de isolamento decorrente da pandemia do covid-19. As aulas foram ministradas pela primeira autora em turmas do ensino fundamental do 2º ao 5º ano, em uma escola privada do estado de Pernambuco. As aulas ocorriam uma vez por semana, de forma remota síncrona, por meio da plataforma Zoom. Nas aulas, os alunos utilizavam a versão online da ferramenta Scratch, onde podiam realizar as atividades e compartilhá-las para a correção. A cada atividade semanal finalizada, um formulário era respondido pelos alunos, onde era adicionado o link de compartilhamento do projeto, para as atividades serem acessadas e avaliadas.

As atividades eram propostas aos alunos de forma contextualizada por meio de *storytelling* (contação de histórias). A figura 1 apresenta um exemplo de atividade, onde o desafio foi construir um cenário e desenvolver uma programação onde o personagem estivesse praticando esporte. A bola precisaria ter seus movimentos programados para chegar até a cesta, e os pontos deveriam ser armazenados em uma variável.

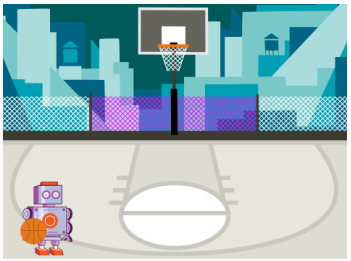

Figura 1 - Exemplo de storytelling utilizado nas aulas. Fonte: Scratch.

O storytelling foi utilizado na intenção de possibilitar a imersão em um mundo lúdico, e proporcionar uma melhor compreensão da atividade. A partir da história narrada, os alunos precisavam observar a problemática apresentada e resolvê-la.

Inicialmente, as atividades foram avaliadas por meio do Dr. Scratch. Entretanto, observou-se que a ferramenta apresenta as pontuações obtidas em cada atividade, mas não provê de nenhuma representação gráfica dos resultados que possam ajudar os professores a analisar o desenvolvimento do aprendizado dos alunos. Assim, os dados das atividades precisam ser extraídos de forma manual, copiando os resultados para uma planilha externa, que não tem nenhuma conexão com a ferramenta, para então conseguir gerar visualizações dos resultados equivalentes às atividades entregues, com a finalidade de obter uma representação gráfica que possa orientar os professores sobre o desempenho no aprendizado e possíveis dificuldades apresentadas nas atividades.

Percebeu-se que tal análise manual se torna impraticável, pois analisar repetidamente atividades de cada aluno ou turma demanda um grande período de tempo utilizado para extração e organização dos dados, a fim de gerar visualizações dos resultados. A partir disso, a fim de propor uma análise sistemática dos dados fornecidos pelo Dr. Scratch, foi feita uma adaptação para possibilitar a extração automática dos dados das atividades, e gerar visualizações dos resultados também de forma automática.

4. Avaliação com o Dr. Scratch

A adaptação no código do Dr. Scratch foi possível já que a ferramenta possui código livre em linguagem Python, fazendo uso do framework Django e containers Docker. Para conseguir extrair os dados automaticamente, um sistema de gerenciamento de banco de dados MySQL foi acoplado aos containers Docker do Dr. Scratch, desta forma sendo possível visualizar por meio do banco de dados, o local onde ficam armazenados os dados das pontuações referentes aos conceitos avaliados, já que ao adicionar uma URL do projeto Scratch, a ferramenta Dr. Scratch avalia, e as pontuações são salvas e

podem ser acessadas diretamente no banco. Após ter acesso aos dados necessários para efetuar as análises, as pontuações passaram a ser extraídas automaticamente em formato CSV para gerar as visualizações gráficas. Após a extração, o arquivo CSV foi adicionado na ferramenta online Google Data Studio, que converte os dados em gráficos, desta forma gerando as visualizações que facilitam a interpretação dos dados obtidos.

O objetivo foi prover visualizações automáticas, geradas por meio do Data Studio, onde o professor pode visualizar os dados das atividades de cada aluno, permitindo-lhe analisar o desenvolvimento do aprendizado. Para testar a solução desenvolvida, foram utilizadas atividades de 16 alunos de turmas distintas, que entregaram a mesma quantidade de atividades, no mesmo período. Passaram pela avaliação 6 atividades de cada aluno. Foram selecionadas 2 atividades do início do período de aulas, 2 atividades no período intermediário e 2 atividades do fim das aulas, somando um total de 102 atividades avaliadas.

Após as visualizações serem geradas, foi produzido um formulário[4] contendo 34 perguntas objetivas, onde foram apresentadas todas as visualizações obtidas na análise automática, a fim de analisar se essas visualizações são de fácil compreensão. Neste formulário, foram feitas perguntas sobre as informações dadas pelas visualizações gráficas, para verificar se os participantes responderiam corretamente, indicando o nível de compreensão.

5. Resultados

Quatro tipos de visualização de dados foram propostas: por trimestre, por turma, por aluno e por quantidade de faltas às aulas, utilizando os dados que são extraídos automaticamente do Dr Scratch. As visualizações geradas foram avaliadas por 28 voluntários, sendo profissionais de 10 diferentes áreas, conforme apresentado na tabela 1.

Tabela 1 - Perfil dos voluntários que responderam o formulário.

Curso	Quantidade
Licenciatura em Computação	10
Ciência da Computação	8
Sistema de Informação	1
Licenciatura em Pedagogia	3
Teologia	1
Economia	1
Licenciatura em Geografia	1
Medicina veterinária	1
Licenciatura em Filosofia	1
Psicologia	1

[4] Formulário.pdf

Para fins de apresentação dos resultados, nesta seção a visualização avaliada será apresentada juntamente com o gráfico com as respostas obtidas no formulário. Em cada visualização, no eixo Y estão representadas as pontuações que cada atividade pode obter, enquanto no eixo X, está representada a quantidade de atividades analisadas.

Na análise realizada por trimestre, foram avaliadas 30 atividades em cada trimestre, e o conceito que menos pontuou no primeiro trimestre foi lógica. A visualização mostrada no gráfico 1 foi apresentada para que os participantes pudessem interpretar, na intenção de verificar a clareza da apresentação dos dados. 57,1% dos participantes responderam corretamente qual foi o conceito menos pontuado, como podemos verificar no gráfico 2.

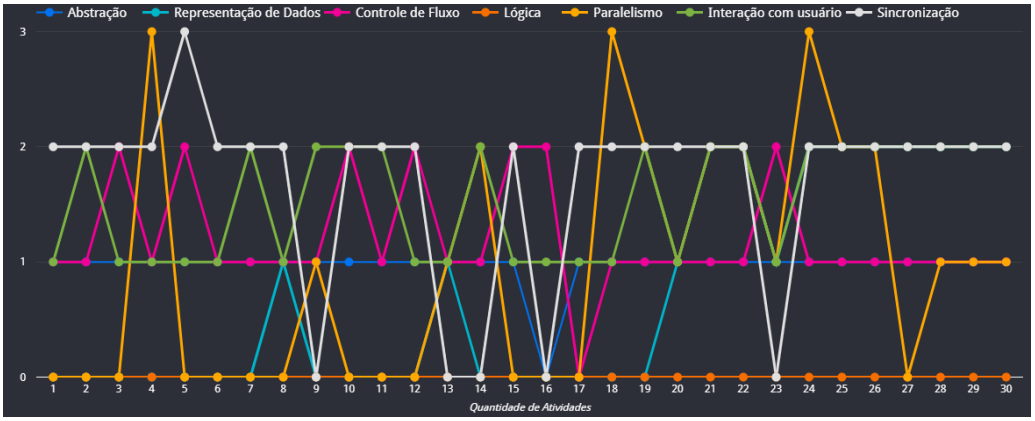

Gráfico 1 - Visualização do 1º Trimestre.

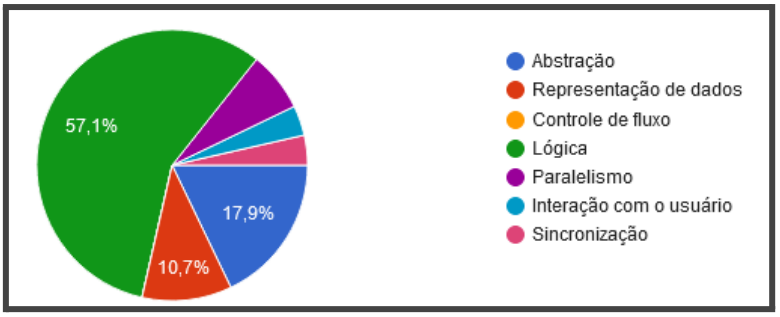

Gráfico 2 - Resultados da pergunta: Qual conceito foi menos pontuado no primeiro trimestre?

Foram analisados também os dados por turma, onde cada representação visual foi interpretada individualmente. Nos dados do quinto ano, apresentados no gráfico 3, foi necessário observar qual conceito não pontuou em 6 atividades, sendo o paralelismo. Conforme os resultados do gráfico 4, a maior porcentagem de respostas foi correta: 46,4% dos participantes conseguiram identificar corretamente o conceito.

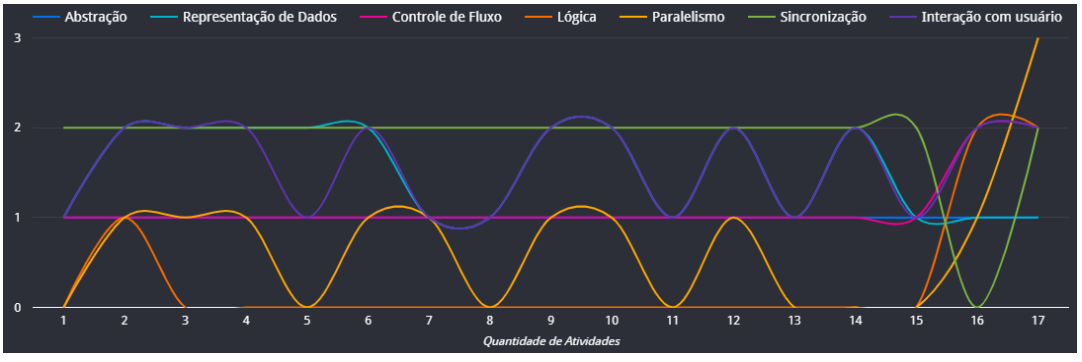

Gráfico 3 - Visualização do 5º Ano.

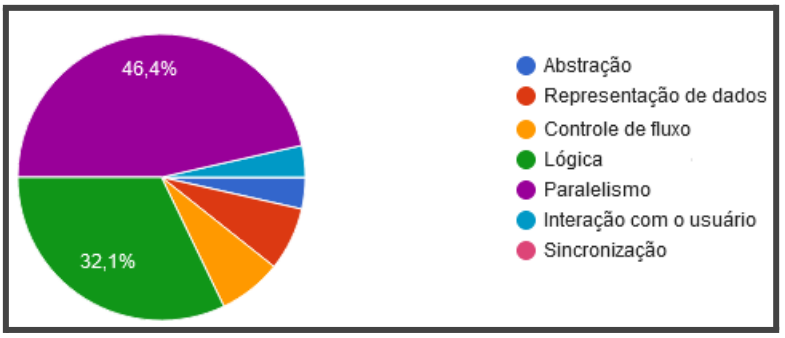

Gráfico 4 - Resultados da pergunta: Nas atividades do 5º ano, qual conceito não teve pontuação em 6 atividades?

Os dados das atividades de cada aluno também fizeram parte das perguntas no questionário. Analisando o gráfico de atividades de um aluno, vários conceitos estavam presentes em todas as atividades, sendo: abstração, controle de fluxo, interação com usuário e sincronização, conforme o gráfico 5. Conforme os resultados presentes no gráfico 6, pode-se observar que as respostas foram em sua maioria positivas, onde os conceitos corretos possuem as maiores porcentagens.

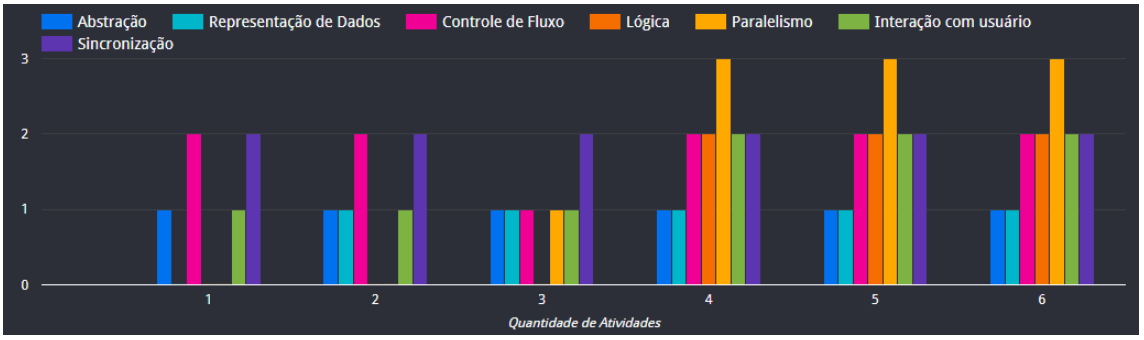

Gráfico 5 - Visualização referente às atividades de um aluno.

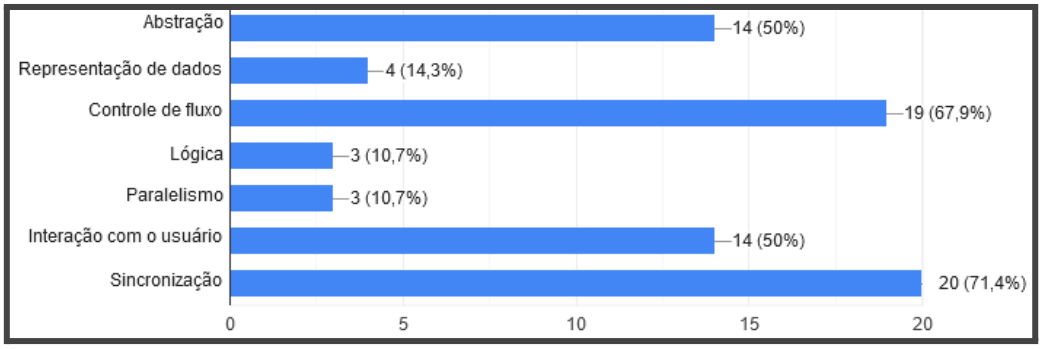

Gráfico 6 - Resultados da pergunta: Nas atividades do aluno, quais conceitos estão presentes em todas as atividades?

Para realizar a análise por faltas, foram utilizadas as atividades dos alunos que tiveram o maior número de faltas nas aulas, a fim de analisar se este fator influenciou no resultado das atividades. O intuito é verificar se houve queda nas pontuações, e quais são os conceitos que mais precisam ser trabalhados. Os participantes precisaram identificar qual conceito apresentou uma pontuação constante na maioria das atividades, sendo o conceito de lógica, que pouco pontuou, tendo uma pontuação constante que se manteve zerada na maioria das atividades, conforme o gráfico 7. O resultado do questionamento pode ser observado no gráfico 8, onde a maioria das respostas foi correta, indicando o conceito de lógica.

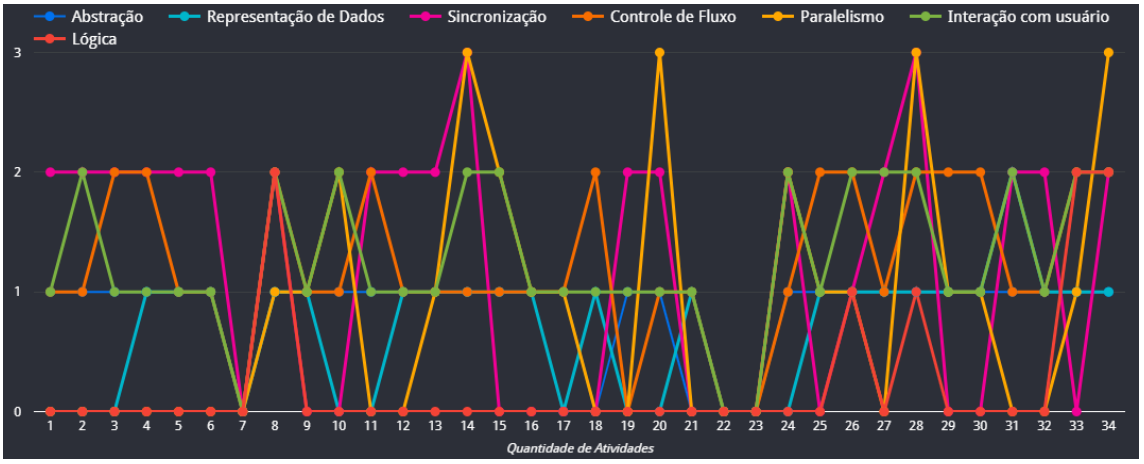

Gráfico 7 - Visualização da análise por faltas nas aulas.

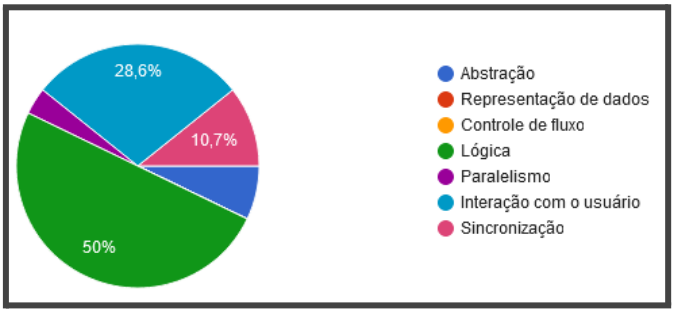

Gráfico 8 - Resultados da pergunta: Qual conceito se manteve com pontuação constante na maioria das atividades?

Os dados das visualizações foram apresentados em estilos diferentes, com a intenção de obter uma resposta sobre qual estilo de gráfico possibilita melhor interpretação de tais visualizações. Com essa análise das visualizações, percebeu-se que para apresentar os dados, é importante escolher o melhor estilo de gráfico que facilite a visualização das informações, evitando possíveis interferência nos resultados. A partir da análise das respostas dos voluntários, concluímos que o estilo de gráfico em linhas com marcação dos pontos, conforme o gráfico 9, foi o que gerou uma melhor compreensão.

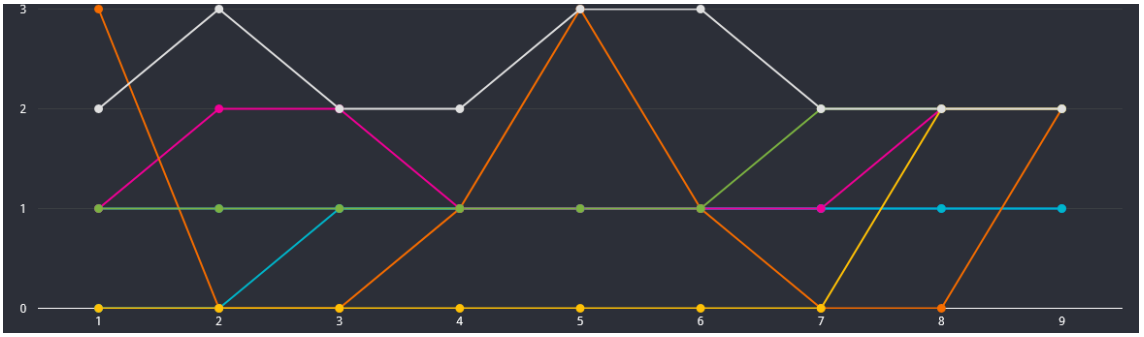

Gráfico 9 - Estilo de gráfico mais compreensível, a partir dos resultados da pesquisa.

Outras visualizações foram analisadas através do formulário. Dada a limitação de espaço, os demais resultados podem ser consultados em um documento[5] complementar.

[5] Acesso a mais resultados obtidos: RESULTADOS.pdf

6. Considerações Finais e Trabalhos Futuros

Este trabalho apresenta uma proposta de procedimento de extração automática de dados utilizando uma versão modificada da ferramenta Dr. Scratch, para gerar visualizações gráficas dos dados das avaliações de atividades do Scratch, em busca de ajudar os professores na análise do desempenho de alunos em atividades de introdução a programação com o Scratch. Tal processo seria impraticável se os dados fossem extraídos de forma manual, pois diversas atividades de cada aluno precisariam ser analisadas, e o processo de coletar as pontuações para cada atividade seria dificultoso e demorado. A partir das visualizações geradas, se necessário o professor pode planejar novas metodologias que impactem na melhoria do desempenho dos alunos, consequentemente buscando contribuir na perspectiva de atingir os objetivos relacionados ao aprendizado de introdução à programação com o Scratch.

Como contribuições voltadas para a análise automática dos dados, temos:

- Diferentes estilos de gráficos para a visualização dos dados podem ser gerados e alterados facilmente por meio do Data studio, para apresentar os resultados da melhor forma;
- A proposta de visualização permite inferências acerca do desempenho dos alunos em geral no período de trimestres diferentes comparando com a análise das turmas, onde pode-se verificar quais conceitos foram mais ou menos pontuados, ao longo das atividades;
- A análise de atividades por aluno proporciona que o professor possa identificar dificuldades dos alunos individualmente e assim moldar estratégias de ensino que ajudem a suprir essas dificuldades.

O presente trabalho teve algumas limitações relacionadas ao contexto de aulas remotas, que impactaram na quantidade de atividades disponíveis para análise, tais como:

- Falta de entrega das atividades, atividades entregues incompletas;
- Quem não conseguiu, por algum eventual motivo, entregar as atividades por meio da URL, passou a entregar a execução do projeto por vídeo e o código por foto, sendo inviável avaliar no Dr. Scratch;
- Atividades que foram entregues, mas no momento da análise, apresentaram compartilhamento não disponível.

Como possíveis trabalhos futuros, temos o desenvolvimento de uma ferramenta com interface de usuário para que cada professor possa realizar suas análises de atividades facilmente e sem necessidade de conhecimento técnico. Além disso, a realização de uma pesquisa com professores de introdução a programação na educação básica, a fim de verificar a contribuição da avaliação automática de atividades e seus resultados de visualização no acompanhamento da aprendizagem dos alunos, em um contexto real.

Referências

ARAÚJO, Ana Liz Souto Oliveira; ANDRADE, Wilkerson L.; GUERRERO, Dalton D. Serey. Um Mapeamento Sistemático sobre a Avaliação do Pensamento Computacional

no Brasil. V Congresso Brasileiro de Informática na Educação (CBIE 2016). Rio Tinto, PB.

ARAÚJO, Nícolas de; PRIMO, Tiago Thompsen; PERNAS, Ana Marilza. OntoScratch: ontologias para a avaliação do ensino de Pensamento Computacional através do Scratch. *In*: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 31. , 2020, Online. **Anais** [...]. Porto Alegre: Sociedade Brasileira de Computação, 2020.

BATES, Tony. Educar na era digital. São Paulo: Artesanato Educacional, 2017. (Coleção tecnologia educacional; 8). Livro Digital.

BINI, Elena Mariele; KOSCIANSKI, André. O Ensino de Programação de Computadores em um Ambiente Criativo e Motivador. Florianópolis, 2009.

ELOY, Adelmo Antonio da Silva. Contribuições para aplicação de learning analytics no apoio à avaliação em atividades de introdução à programação com Scratch. 2019. Dissertação (Mestrado em Sistemas Eletrônicos) - Escola Politécnica, Universidade de São Paulo, São Paulo, 2019.

MANYIKA, J., Chui, M., Madgavkar, A., & Lund, S. (2017). Technology, jobs, and the future of work. McKinsey Global Institute.

MARCON JÚNIOR, Rogério Paulo; BONIATI, Bruno Batista. LogicBlocks: Uma Ferramenta para o Ensino de Lógica de Programação. Rio Grande do Sul. Anais do EATI - Encontro Anual de Tecnologia da Informação e Semana Acadêmica de Tecnologia da Informação, 2015.

MARJI, Majed. Aprenda a programar com Scratch: Uma introdução visual à programação com jogos, arte, ciência e matemática. São Paulo: Novatec, 2014. Tradução: Lúcia Kinoshita.

MORENO-LEON, Jesus; ROBLES, Gregorio; ROMAN-GONZALEZ, Marcos. Comparing computational thinking development assessment scores with software complexity metrics. IEEE Global Engineering Education Conference, EDUCON, v. 10-13-April-2016, n. April, p. 1040–1045, 2016.

MORETTI, Vinícius Fernandes. O pensamento computacional no ensino básico: potencialidades de desenvolvimento com o uso do Scratch. 2019. Trabalho de conclusão de curso (Graduação) - Universidade Federal do Rio Grande do Sul.

WING, J. Computational thinking. Communications of the ACM, v. 49, n. 3, p. 33-35, 2006.