Geração automatizada de gabarito e correção de exercícios em plataforma para o ensino do pensamento computaciona
A support tool for the teaching of computational thinking with support to the generation of jigs and automatic correction of exercises

Francisca Edyr Xavier
Laboratório de Desenvolvimento e Transferência de Tecnologias
Departamento de Sistemas e Computação
Universidade Regional de Blumenau (FURB) – Blumenau, SC – Brazil
exavier@furb.br

Luciana P. de Araújo Kohler
Laboratório de Desenvolvimento e Transferência de Tecnologias
Departamento de Sistemas e Computaçãos
Universidade Regional de Blumenau (FURB) – Blumenau, SC – Brazil
lpa@furb.br

Mauro Marcelo Mattos
Laboratório de Desenvolvimento e Transferência de Tecnologias
Departamento de Sistemas e Computação
Universidade Regional de Blumenau (FURB) – Blumenau, SC – Brazil
mattos@furb.br

Heitor Ugarte
Laboratório de Desenvolvimento e Transferência de Tecnologias
Departamento de Sistemas e Computação
Universidade Regional de Blumenau (FURB) – Blumenau, SC – Brazil
hucsilveira@furb.br

Bruno F. F. Santos
Laboratório de Desenvolvimento e Transferência de Tecnologias
Departamento de Sistemas e Computação
Universidade Regional de Blumenau (FURB) – Blumenau, SC – Brazil
bffsantos@furb.br

Leonardo Fronza
Laboratório de Desenvolvimento e Transferência de Tecnologias
Departamento de Sistemas e Computação
Universidade Regional de Blumenau (FURB) – Blumenau, SC – Brazil
leofronza@furb.br

ABSTRACT

Este artigo apresenta a automatização da geração e correção de exercícios em ambiente já existente para o ensino do pensamento computacional. Para o desenvolvimento, foi utilizada a última versão deste ambiente desenvolvido na linguagem Java. O algoritmo de Dijkstra foi utilizado para encontrar os caminhos mínimos entre os objetivos do mapa. A combinação de caminhos de menor tamanho é transformada em uma sequência de instruções para executar esse percurso, gerando o gabarito do exercício. A correção compara os resultados obtidos pelo aluno com o gabarito. Após testes realizados com crianças, teve-se como resultado que essas elas se sentiram motivadas a buscar melhores soluções nos exercícios por possuir um gabarito com a comparação.

ABSTRACT

This paper presents a automatization of generation and evaluation of exercises in exists environment to teach computational thinking. The Java language was used for development and the last version of the environment. The Dijkstra algorithm was used to find the minimum way among the objectives of the map. The combination of minimum ways is transform in a sequence of instructions to execute the route generating the template of exercise. The evaluation compares the results obtained by the student with the template. After the tests realized with children, it has a result that they feel motivated to found the better solution in the exercises because the platform has a template for comparision.

CCS Concepts

•**Theory of computation** → *Graph algorithms analysis;*
•**Applied computing** → *Education;*

Keywords

Dijkstra; automatização de correção de exercícios; menor caminho
Dijkstra; automation of exercise correction; shortest path

1. INTRODUÇÃO

A computação é um dos fatores responsáveis por acelerar a ocorrência de mudanças em todas as áreas do conhecimento, exigindo novas competências e habilidades cognitivas [8]. Dessa forma, é reconhecido pela Sociedade Brasileira de Computação (SBC) que é fundamental a inserção da computação na Educação Básica, pois os conhecimentos

básicos de computação são de grande relevância para a inserção de indivíduos na sociedade contemporânea, além de que as habilidades e competências adquiridas pelo pensamento computacional potencializam a capacidade de soluções de problemas [8].

Com intuito de incentivar o uso de tecnologias em escolas, bem como aplicar o ensino de pensamento computacional, foi desenvolvida uma plataforma com base em um *framework* utilizado desde 2008 para o ensino da disciplina de introdução de computação nos cursos de Ciência da Computação e Sistemas da Informação [1]. A plataforma foi desenvolvida para disponibilizar um ambiente atrativo e facilitador para o uso na Educação Básica como ferramenta de ensino de programação, que introduz conceitos tais como, programação sequencial, uso de condicionas e laços de repetição. Na plataforma, os alunos comandam um robô através de um mapa bidimensional, no qual devem desviar os obstáculos e coletar itens para concluir os objetivos propostos pelo enunciado do exercício. Os exercícios, assim como a configuração de todo o cenário é realizada em um documento eXtensible Markup Language (XML), os quais podem ser criados por um usuário através de um editor disponível na plataforma.

Ao concluir o objetivo, a plataforma informa para o aluno se completou a missão baseada na quantidade de tesouros coletados. Ainda, ela produz uma pontuação com base no código-fonte produzido. Caso este tenha a mesma quantidade de linhas que o código fonte definido como um gabarito do exercício dentro do XML e tenha coletado todos os objetivos, o aluno recebe uma bonificação maior. Contudo, há várias formas de resolver um mesmo exercício na plataforma. Na plataforma atual, a produção desse gabarito é feita de forma manual no arquivo XML. Dessa forma, a elaboração de um novo exercício, além do processo de correção, se torna limitada, pois os gabaritos e as verificação das atividades deve ser feitas de forma manual. Esse tempo poderia ser direcionado para alunos que possuem dificuldades para alcançar uma resolução ou para aprimoramento nas dinâmicas de ensino [6].

A versão ampliada do *framework* e da plataforma apresentada neste artigo, tem como objetivo automatizar o processo de geração dos gabaritos criados pela ferramenta de geração de mundo, através do uso do algoritmo de menor caminho Dijkstra, gerando a solução ótima para os objetivos do mapa e automatizar o processo de correção dos exercícios. Além disso, tem-se como objetivos específicos auxiliar no desenvolvimento do pensamento computacional, exercitando análise crítica dos alunos, os quais por conta própria podem tentar buscar um melhor resultado e formar alternativas de soluções. Outro objetivo também está em facilitar o uso da plataforma em sala de aula, simplificando o processo de criação e de correção dos exercícios.

Assim, esse artigo apresenta como foi feito o processo para efetuar a geração de gabaritos automatizados dos exercícios criados na ferramenta de geração de mundo e como o algoritmo de Dijkstra foi utilizado para auxiliar na busca do menor caminho, buscando assim a solução ótima dos mapas. Além disso, efetua a comparação dos gabaritos gerados com as resoluções dos alunos, automatizando o processo de correção. São demonstrados também os testes de performe da ferramenta de geração de gabarito, bem como respostas obtidas por alunos de uma turma de programação que já utilizou a plataforma.

2. COMPUTAÇÃO NA EDUCAÇÃO BÁSICA

Segundo [8], "A Sociedade Brasileira de Computação (SBC) entende que é fundamental e estratégico para o Brasil que conteúdos de Computação sejam ministrados na Educação Básica". Algumas abordagens veem sendo ressaltadas em vários trabalhos como apontado por [4], que busca reforçar a ideia da aplicação do ensino de computação nos anos iniciais, introduzindo a programação através do Scratch. Outra abordagem apresentada no trabalho de [6] é a aplicação da metodologia de ensino que utiliza o aluno como protagonista, incitando-o a pensar no problema a ser resolvido.

Para que um indivíduo esteja pronto para a sociedade contemporânea, ele deverá estar preparado para entender e tirar proveito dos avanços providos pela relação da Computação com as outras áreas de conhecimento, dessa forma a computação se torna um conhecimento tão importante quanto Matemática, Filosofia, Física e outras Ciências [8]. Os conhecimentos relacionados a Computação podem ser divididos em três eixos: Pensamento computacional; Mundo Digital; e Cultura Digital [8].

Nesse contexto, [9] ressalta a importância do pensamento computacional, pois é uma habilidade fundamental para potencializar a capacidade de resolução de problemas de um indivíduo. Para resolver problemas complexos é necessário entender que é possível criar uma representação abstrata para facilitar a construção de soluções, podendo as descrever em passos, possibilitando uma análise crítica do objetivo e buscando soluções melhores [9]. Dentro do ramo de Mundo Digital a codificação é descrita como a compreensão de como ela pode ser descrita e armazenada [8].

Por sua vez, a Cultura Digital tem como um de seus pilares a fluência tecnológica [8]. Portanto, vem se utilizado formas lúdicas para introdução de computação para crianças, assim como as abordagens feitas pela [2]. Essas abordagens trazem exercícios autoexplicativos que aumentam o grau de dificuldade progressivamente, incluindo novos comandos ou aumentando a complexidade do exercício, além de que traduzem problemas computacionais de forma visual para cativar o aluno utilizando personagens conhecidos. Ainda, alguns autores consideram a diversão parte da Cultura Digital, pois aumenta o engajamento e a motivação, o que facilita a aprendizagem.

3. PLATAFORMA ATUAL

A plataforma base para este trabalho teve a sua origem em 2008 a partir do desenvolvimento de um *framework* Java. Em 2017, o *framework* foi reformulado a partir de um projeto de extensão iniciado com intuito de ser utilizado em escolas de ensino básico, tendo seu ambiente sendo transformado em uma plataforma de ensino de programação [1].

Esssa plataforma é dividida em três níveis de dificuldade. No nível fácil se introduz os conceitos de programação se-

quencial a partir de setas do teclado. No nível médio as setas de teclado são substituídas pelo uso de código-fonte. Já no nível de dificuldade difícil o uso de laços de repetição utilizando características de um jogo é introduzido. A partir da programação de um robô, o objetivo é fazer com que o mesmo chegue até um destino coletando tesouros, evitando alienígenas e desviando das paredes. O aluno pode escolher um nível de dificuldade e em seguida deve escolher um exercício correspondente a este nível. Então é exibido o enunciado do exercício com a descrição de qual é o objetivo do cenário. O ambiente do jogo é demonstrado na Figura 1.

Figure 1: Exemplo do ambiente do jogo

O ambiente é formado pelo cenário do mapa, o qual é uma tabela com o tamanho definido pelo exercício, dividido em linhas e colunas. Na Figura 1 é possível observar os tesouros e anjos, sendo os itens coletáveis espalhados no mapa, além dos aliens e paredes que são os obstáculos a serem desviados. Ainda no mapa tem-se o robô, personagem programável para percorrer o caminho e alcançar o objetivo do exercício. No lado direito são visíveis a quantidade de itens atualmente coletados e a pontuação. Na parte inferior da tela se encontram os botões para voltar para a seleção de exercícios, abrir o console para programar, pausar o jogo e visualizar o enunciado do problema, respectivamente.

Para resolver o exercício, o aluno deve utilizar comandos para movimentar o robô. No nível fácil é possível movimentar o robô através das setas do teclado. Já no nível médio o aluno deve abrir o console de programação e utilizar comandos escritos para movimentar o robô, sendo `andarDireita()`, `andarEsquerda()`, `andarAcima()` e `andarAbaixo()`. No nível difícil é adicionado os comandos de enquanto para efetuar a programação com laços de repetição.

Após fazer a resolução, o aluno deve apertar o botão executar para validar os comandos e executar as instruções. Ao se movimentar, o robô deixa um rastro de óleo, o qual permite visualizar todo o caminho que ele percorreu no mapa. Ao coletar todos os tesouros ele irá considerar o exercício como resolvido e irá exibir em uma tela os comandos executados e a contagens dos pontos. Caso a quantidade de comandos executados seja a mesma que a identificada nas respostas do exercício, o aluno recebe um bônus na pontuação. Os itens coletados também geram pontos adicionais, já esbarrar em aliens ou na parede faz com que o aluno perca pontos.

A plataforma também possui uma interface visual para editar os mundos, visto na Figura 2, denominada Gerador de Mundo. O gerador permite desenhar os mapas com os obstáculos, itens coletáveis e inimigos, além de escrever o enunciado do exercício. Contudo, as fases criadas não estão integradas de forma dinâmica com a plataforma, isto é, para que o aluno possa joga-la é necessário recompilar o projeto pelo ambiente de desenvolvimento gerando um novo executável com a fase. Ainda, não é possível adicionar respostas para os exercícios através do Gerador de Mundo. Logo, para usar o recurso que efetua bonificação aos alunos pela resposta informada é necessário incluir de forma manual todos os nós respostas com seus respectivos valores no arquivo XML gerado ao salvar o mundo.

Figure 2: Gerador de mundo

4. FERRAMENTA DE GERAÇÃO DE GABARITOS

A Ferramenta de Geração de Gabaritos é responsável por ler o mapa desenhado através da interface visual do Gerador de Mundo da plataforma e gerar um objeto de classe Gabarito. Inicialmente o usuário deve desenhar o mundo do exercício, distribuindo como desejar os objetos no mapa. Duas opções iniciam o processo de geração de gabarito sendo: a opção de consultar o gabarito; e a opção de salvar o mundo. Em seguida, os componentes do mapa atual do gerador de mundo são classificados. Em paralelo é gerada a lista de adjacências. Se o processo for bem-sucedido, o fluxo passa para a etapa para encontrar o menor caminho. Caso existam objetivos opcionais, é gerada uma sequência de combinações de objetivos opcionais para gerar gabaritos opcionais.

Ao adicionar componentes no gerador de mundo, eles são carregados em uma lista de objetos, guardando informações de posição na tabela e do tipo do componente. Quando o usuário confirma a gravação do mapa ou quando efetua uma consulta no gabarito, o fluxo de atividade da Ferramenta de Geração de Gabarito é inicializado. Antes de efetuar a busca do menor caminho os objetos do mapa são classificados. Primeiro são classificados os objetos do tipo obstáculos que não fazem parte do caminho ou que não devem ser coletados no mundo. São considerados como obstáculos as paredes, os aliens e os números não marcados como objetivo. Em seguida são classificados os objetivos. Objetivos são componente que devem ser coletados para poder concluir o exercício, sendo que são obrigatórios. São considera-

dos objetivos os componentes tesouros e números marcados como objetivos. Ainda existem os objetivos opcionais que são componentes que podem ser coletados para dar pontos extras. É considerado como objetivo opcional o componente anjo. Além disso, deve-se saber o ponto de partida do robô.

A partir desses elementos é criada uma matriz análoga a tabela no qual os componentes da lista são redistribuídos com base em sua posição e uma matriz auxiliar de vértices. Essas matrizes são percorridas, os objetos classificados como obstáculos são ignorados e para cada outro campo da matriz, incluso os campos vazios, são criadas instâncias de vértices e atribuídos a posição equivalente na matriz auxiliar. Para cada vértice criado na matriz auxiliar será checado se existem vértices adjacentes. Todos os vértices são adicionados na lista de vértices. Os vértices que contém objetivos são adicionados na lista de objetivos e os que contém objetivos opcionais na lista de objetivos opcionais.

Após isso, verifica-se o menor caminho para a resolução do exercício. Para encontrar o menor caminho é realizada a busca da combinação de caminhos que possua o menor tamanho para alcançar todos os objetivos listados, sendo esse o caminho que passa por uma menor quantidade de vértices para alcançar todos os objetivos do mapa. Para isso, passa-se por todos os objetivos da lista, os quais são usados de destino e então é calculado o menor caminho a partir da origem. Para isso é aplicado o algoritmo de Dijkstra.

O Algoritmo de Dijkstra é um dos algoritmos utilizados para se encontrar o menor caminho entre dois pontos de um grafo. O Dijkstra é utilizado em grafos com arestas ponderadas com valores não negativos. Seu custo computacional é equivalente a uma busca em largura quando todos os vértices possuem o mesmo valor de atributo [5]. Já os grafos são estruturas de dados utilizadas na computação para representar um conjunto de elementos e o relacionamento entre eles, de forma que seja possível aplicar soluções de diversos problemas computacionais, dentre deles o problema de caminho mínimo [3].

Caso existam objetivos opcionais é efetuada a geração de combinações de objetivos opcionais. Para cada uma das combinações é efetuado o processo de buscar o menor caminho no qual a combinação de objetivos opcionais é adicionada à lista de objetivos a serem explorados pelo algoritmo. Quando existem objetivos opcionais no mapa é necessário efetuar validações.

Durante a geração de gabaritos opcionais deve ser garantido que todos os objetivos opcionais serão coletados antes de coletar o último objetivo obrigatório. Isso é necessário porque o exercício será finalizado quando o robô atingir o último objetivo obrigatório do mapa. Ocorre que essa validação é feita apenas nos destinos do caminho para garantir que em nenhum momento o vértice cruze com um objetivo obrigatório no meio do caminho.

Após identificar o menor caminho, o mesmo é adicionado no XML do mapa gerado como sendo a resposta para aquele mapa. Para aumentar o dinamismo da versão atualizada da plataforma, foi efetuada uma adaptação no código para que permita que os jogos criados pelo gerador de mundo sejam carregados em tempo de execução. Anteriormente seria necessário incluir o exercício na Plataforma e criar um executável. Com essa adaptação, os exercícios criados possuem acesso disponível através de um botão na tela na qual é feita a seleção do nível de dificuldade do exercício.

5. FERRAMENTA DE AUTO AVALIAÇÃO

A Ferramenta de Autoavaliação é responsável por retornar à comparação da resolução no momento no qual o jogo finalizar, ou seja, após alcançar todos os objetivos. Para incluir a autoavaliação nos exercícios, primeiramente foi necessário adaptar o processo responsável por carregar os exercícios no jogo.

Para saber as possíveis soluções, são carregados os gabaritos existentes do arquivo XML do exercício. Durante a execução do exercício na plataforma são guardadas as informações referentes ao desempenho do aluno, como colisões com obstáculos e itens coletados. O exercício será considerado concluído quando todos os componentes classificados como objetivos do mapa forem recolhidos. A versão apresentada nesse artigo permite definir números como objetivos do mapa. O processo anteriormente só considerava os componentes tesouros. Após o processo finalizado é aberta a Tela de Gabarito para a comparação com o código fonte gerado pelo aluno, mostrando a avaliação da resolução feita pelo aluno, conforme exibido na Figura 3.

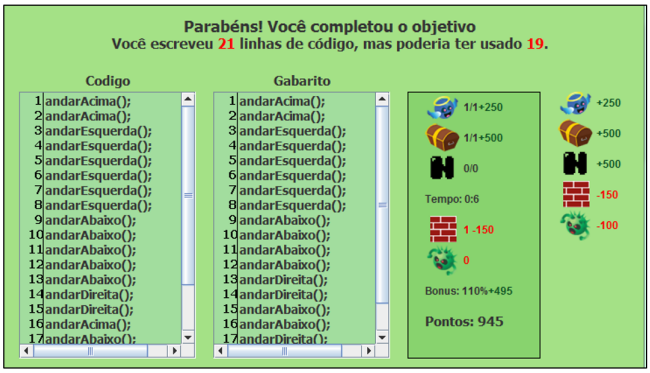

Figure 3: Gabaritos

A tela de comparação possui uma mensagem de parabéns por concluir o exercício e uma mensagem descrevendo a comparação com os comandos contidas no gabarito. Além disso, tem-se um painel à esquerda com os comandos gerados pelo aluno e ao lado dele um painel com os comandos do gabarito. Ainda, no lado direito há um painel contendo todos os elementos de pontuação discriminando a quantidade respectiva de pontos recebido pelo aluno e o valor individual de cada elemento.

A decisão de qual gabarito deve ser usado para mostrar na interface é feita de acordo com a quantidade de objetivos opcionais que o aluno coletou durante a resolução do exercício. Caso não existam objetivos opcionais será retornado o gabarito padrão. A partir do gabarito escolhido é feita a análise responsável por trazer uma mensagem de comparação. A condição para a exibição das mensagens pode ser vista na Tabela 1.

Conforme destacado na Tabela 1, há a comparação do código gerado pelo aluno com o código do gabarito de forma idêntica, identificado por “Resolução igual ao gabarito”, bem como resolução com a mesma quantidade de linhas identificado por “Resolução ótima diferente do gabartido”.

6. AVALIAÇÃO E DISCUSSÃO DOS RESULTADOS

Table 1: Mensagens

Condição	Mensagem
Não possui gabarito para o exercício	Você escreveu [quantidade de linhas escritas] de código.
Resolução igual ao gabarito	Você escreveu [quantidade de linhas escritas] de código. Igual ao Gabarito.
Resolução ótima diferente do gabarito	Você escreveu [quantidade de linhas escritas] de código. Equivalente ao Gabarito.
Resolução não ótima	Você escreveu [quantidade de linhas escritas] de código, mas poderia ter usado [quantidade de linhas do gabarito].

Esta seção apresenta as avaliações realizadas com a plataforma apresentada neste artigo. Inicialmente faz-se uma comparação desta com outras soluções semelhantes disponíveis no mercado. Em seguida demonstra-se um teste de performance realizado por meio de testes unitários. Por fim, apresenta-se uma avaliação prévia de usabilidade realizada com 4 crianças, sendo estas os usuários da plataforma.

Com o objetivo de comparar a solução desenvolvida com outras disponíveis no mercado, comparou-se esta com a aplicação Anna e Elsa da Code.org [2] e o Robozzo [7]. Anna e Elsa é uma ferramenta de ensino de programação gratuita que busca expandir o acesso da informática na escola e aumentar a participação de mulheres e minorias. Ela possui 20 exercícios introdutórios de programação fixos que utilizam personagens infantis populares, os quais devem ser resolvidos através do uso de blocos de programação. Após o aluno concluir o objetivo, retorna o desempenho do aluno e o código gerado em JavaScript. Possui um ambiente Web online e um ambiente offline. A personagem fornece dicas e instruções para a resolução da aplicação, incluindo se era possível se resolver com menos comandos.

Já o Robozzo é um jogo que tem como objetivo o entretenimento de quebra cabeça social online. Ele permite resolver quebras cabeças, no qual se utiliza comandos representados por símbolos para se mover um robô em forma de seta por um mapa em ladrilhos. O ambiente social disponibiliza um ambiente para elaboração e compartilhamento de quebra cabeças criados pelos usuários. Ao se concluir um quebra cabeça é possível comparar a quantidades de comandos usados na resposta com a de outros jogadores.

Comparando essas ferramentas, tem-se que a Anna e Elsa possui a capacidade de avaliar a resolução elaborada pelo usuário de forma automatizada, indicando a possibilidade de uma solução melhor e retornando para o usuário o código gerado por ele. A aplicação apresentada neste artigo também possui essa característica no processo de avaliação. Ela retorna também para o aluno o gabarito que foi efetuado para realizar esse exercício comparando com a quantidade de comandos que ele poderia ter utilizado e apresenta o percentual de bônus pelo acerto. A plataforma também define dentre os gabaritos gerados para o exercício, aquele que corresponde melhor a solução no caso do aluno atingir objetivos que não são obrigatórios.

No caso da ferramenta Robozzo, bem como na versão anterior da plataforma todas possuem um ambiente de criação de exercício. Robozzo obriga o usuário a fornecer uma solução válida para o exercício criado. A plataforma anterior permitia que fosse fornecida uma solução, mas a validade e a qualidade da solução não eram verificadas. Com a automatização do processo de validação de solução demostrada nesse artigo, qualquer mapa criado pelo ambiente visual vai primeiramente validar a existência de uma solução possível, além de gerar sempre uma solução ótima para a coleta dos objetivos obrigatórios do mapa. Para casos com objetivos opcionais, há uma solução ótima para cada quantidade diferente de objetivos a serem coletados.

Além desta comparação, foram efetuados testes unitários utilizando a ferramenta JUnit 4 na plataforma de desenvolvimento Eclipse. Os testes buscavam saber o tempo de demora que teria a execução responsável por retornar as instâncias de gabarito em relação a certas características possíveis para o mapa. Foi identificado o tempo e o aumento no tempo na geração de gabaritos com base na quantidade de objetivos existentes no mapa. O aumento do tempo de execução se tornou notável ao uso, quando se entra na faixa de dez objetivos no mapa, já que até então a geração dos gabaritos demora tempo inferior a 1 micro segundo. Ao adicionar restrição para só executar o Dijkstra, caso o caminho que se deseja percorrer for conhecido, o mesmo é resgatável de uma lista salva em memória. Esse primeiro teste foi baseado na média de execução de 100 gabaritos de um mapa com 10 colunas e com 10 linhas, sem nenhum obstáculo presente, com todos os objetivos a uma distância a partir de 10 vértices, sendo o ponto de início do mapa na coluna 1 linha 1.

Em outros testes feitos pela mesma ferramenta JUnit 4 foi efetuado teste com mapas de dimensões diferentes. A partir disso foi identificado o aumento no tempo médio na geração de gabaritos com com relação aos tamanhos dos mapas dos exercícios criados, como é possível observar na Figura 4.

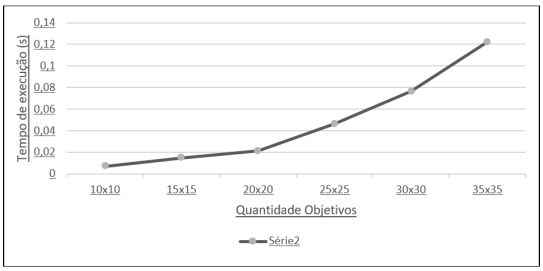

Figure 4: Gráfico de tempo médio da execução do método `gerarGabarito()` em mapas de tamanhos diferentes

O gráfico apresentado na Figura 4 tem seu valor baseado na média de tempo de execução de exercícios de 1 à 14 objetivos e do tempo médio da execução de 100 gabarito de mapas, sem nenhum obstáculo presente, com todos os objetivos a uma distância a partir de 10 vértices, sendo o ponto de início do mapa na coluna 1 linha 1.

Para realizar uma avaliação preliminar na prática a versão atualizada da plataforma, ela foi levada para ser usada em um curso de programação. Na turma estavam presentes 4

crianças, sendo duas de 9 anos de idade e duas de 11 anos. Três delas eram do sexo masculino e uma do feminino. Todos os alunos já tiveram aulas com a versão antiga da plataforma e todos já haviam programado nas 3 dificuldades disponíveis por ela. Como as crianças não foram identificadas na pesquisa, apenas solicitou-se autorização dos pais para que as mesmas participassem da pesquisa. Dessas crianças, uma já havia usado a ferramenta de geração de mundo da versão anterior da plataforma. Foi então disponibilizado para elas a versão descrita nesse artigo e instruído para que elas criassem seus próprios mapas e depois os resolvessem. Após as práticas foram feitas uma série de perguntas para as crianças conforme descrito na Tabela 2.

Table 2: Perguntas realizadas

Perguntas
Você compreendeu a tela do gabarito?
Você percebeu algum erro durante o jogo?
Você se sentiu incentivado a realizar o menor caminho nas fases?
O que você achou da apresentação das respostas no gabarito?

Quando foi questionado para as crianças se haviam compreendidos a tela, apenas uma delas apontou que não havia ficado claro os pontos, bem como porque chegou nessa pontuação. Quando questionado se foi identificado algum erro, um deles apontou que havia batido na parede e essa batida não foi contabilizada. Ao ser questionado se eles se sentiram incentivados a fazer o menor caminho tendo o gabarito para comparar no final, todos concordaram que sim e um apontou que dessa forma "tem um pouco mais de pressão".

Os comentários das crianças quando questionadas sobre o que acharam da apresentação das repostas foram: "Achei legal, tem formas de ver outras oportunidades"; "Tem a correção, é bom para saber o que errou", "Achei legal, porque mostra as possibilidades que dá para fazer", "Achei legal, porque mostra o que você faz, o tanto de pontos e se fez errado".

Com base nos resultados obtidos pode-se observar que foi possível automatizar o processo de geração de gabarito. Com base nos testes de performance de geração de gabaritos, é possível observar eventualmente se terá uma quantidade limitada de objetivos que se deve usar em um exercício para que os gabaritos sejam gerados em tempo hábil. Referente as respostas obtidas com a turma de crianças, ter acesso ao resultado ótimo motivou as crianças a analisarem seus próprios resultados. Pode-se concluir também que ter a autocorreção do exercício e resultados de forma mais detalhadas foi recebido de forma positiva pelos alunos.

7. CONCLUSÃO

Este artigo apresentou o desenvolvimento de uma ferramenta para automatizar o processo de geração de gabaritos do gerador de mundo de uma plataforma já existente, através da busca de menor caminho com o auxílio do algoritmo de Dijkstra. Além disso, apresentou uma ferramenta para automatizar a avaliação das resoluções desenvolvidas pelos alunos, auxiliando o desenvolvimento do pensamento computacional. As ferramentas foram desenvolvidas a partir do ambiente Eclipse na linguagem Java, implementado sobre a última versão do *framework* existente.

A versão ampliada da plataforma teve como objetivo automatizar o processo de geração dos gabaritos criados pela ferramenta de geração de mundo. Para a implementação foi utilizado o algoritmo de Dijkstra para encontrar o menor caminho entre a origem e os objetivos, os quais foram agrupados através de um processo de busca recursiva. Através disso foi possível encontrar o menor caminho ótimo dentre as combinações de caminhos possíveis. Os caminhos encontrados têm-se então as direções de movimentação identificada. Essas direções são traduzidas em comandos para a plataforma. Outro objetivo era facilitar o uso da plataforma em sala de aula para o auxílio do desenvolvimento do pensamento computacional. A versão atualizada da foi então levada para ser usada em um curso de programação com ela, no qual as crianças puderam criar seus próprios exercícios. Com o processo de geração de gabarito e análise da resolução, elas puderam jogar esses exercícios e obter a correção de forma automatizada.

O processo de geração de gabarito permite gerar não só os objetivos principais, mas também considerar itens opcionais que possam ser coletados pelo aluno durante a execução. Para permitir que o aluno receba a bonificação pela quantidade de itens opcionais coletados e a bonificação por se conseguir o menor caminho para coletar essas quantidades de itens é gerado um gabarito ótimo para cada quantidade de objetivo opcional que o aluno pode coletar.

A partir dos resultados obtidos é possível observar que o tempo de demora de geração de gabaritos começa a aumentar de forma abrupta ao possuir certa quantidade de componentes em tela, conforme o tamanho de mapa utilizado, levando em conta que o tempo varia dependo da distribuição dos componentes e da existência de obstáculos. Esse tempo impossibilitaria a criação de certos mapas serem feitos em sala de aula em um tempo hábil.

Em comparação com seus correlatos, a versão atualizada da plataforma consegue juntar duas características que eram oferecidas em ferramentas diferentes, oferecendo o processo de correção automatizada, além de permitir um ambiente de elaboração de exercícios, garantindo que o gabarito oferecido pela ferramenta será uma solução ótima.

Ressaltam-se as limitações identificadas no trabalho, o acréscimo abrupto que ocorre no tempo de demora para gerar os gabaritos por conta da relação de tamanho do mapa e quantidade de objetivos, o que se torna necessário estipular limites para conseguir uma solução em tempo hábil. Além do processo que se agrava ao gerar gabaritos com objetivos opcionais, por conta da quantidade de combinações de objetivos que precisam ser gerados e testados.

8. REFERENCES

[1] L. Araújo, H. U. C. da Silveira, and M. Mattos. Ensino do pensamento computacional em escola pública por meio de uma plataforma lúdica. In *Anais dos Workshops do VII Congresso Brasileiro de Informática na Educação (CBIE 2018)*. Brazilian Computer Society (Sociedade Brasileira de Computação - SBC), Oct. 2018.

[2] Code.org. Programação anna e elsa. Disponível em: https://studio.code.org/s/frozen/stage/1/puzzle/1, 2018.

[3] T. H. Cormen. *Introduction To Algorithms*. Mcgraw-Hill College, mar 1990.

[4] M. L. S. de Oliveira, A. A. de Souza, A. F. Barbosa, and E. F. S. Barreiros. Ensino de lógica de programação no ensino fundamental utilizando o scratch: um relato de experiência. In *Anais do XXXIV Congresso da Sociedade Brasileira de Computação - CSBC*, pages 1493–1502, 2014.

[5] W. Kocay. *Graphs, Algorithms, and Optimization (Discrete Mathematics and Its Applications)*. Chapman and Hall/CRC, nov 2004.

[6] M. Mattos, L. Araújo, H. U. C. da Silveira, L. Schlögl, G. C. Giovanella, B. Santos, L. Fronza, F. Zucco, N. Hein, G. C. de Oliveira, K. Z. da Cunha, and A. Sartori. Uma pesquisa-ação sobre o desenvolvimento do pensamento computacional com crianças. In *Anais do XXIV Workshop de Informática na Escola (WIE 2018)*. Brazilian Computer Society (Sociedade Brasileira de Computação - SBC), Oct. 2018.

[7] I. Ostrovsky. My hobby project: a social puzzle game developed in silverlight. Disponível em: http://igoro.com/archive/my-hobby-project-a-social-puzzle-game-developed-in-silverlight/, 2009.

[8] A. L. A. Raabe et al. Referenciais de formação em computação: Educação básica. porto alegre: Sbc. 9p., jul 2017.

[9] J. M. Wing. Computational thinking. *Commun. ACM*, 49(3):33–35, Mar. 2006.