

Viral_goritmos: jogo educacional para ensino e aprendizagem de algoritmos

Miguel Ferreira da Silva Neto
Universidade Federal da Grande Dourados
Dourados, Mato Grosso do Sul, Brasil
miguelfsneto04@gmail.com

Valguima Victoria Viana Aguiar Odakura
Universidade Federal da Grande Dourados
Dourados, Mato Grosso do Sul, Brasil
valguimaodakura@ufgd.edu.br

Resumo

This article presents the development of an educational game aimed at teaching and assisting higher education students with learning algorithms and programming. Due to the significant problem-solving difficulties faced by students, the low pass rates in algorithms and programming courses, and a gap in games that cover this content, a game was developed to provide an interactive environment to experience different ways of teaching abstract concepts of algorithms, maintaining its focus on Portugol Studio's syntax, as well as to serve as a tool to support teaching and learning of algorithms and programming courses. The assessment that determined the relevance of this game was conducted following game evaluation models proposed in scientific studies, the model focused on evaluating criteria such as motivation, UX and learning value. The educational game was positively evaluated by university students who make part of the target audience and was considered relevant with its proposal.

Keywords

Jogo educacional, Algoritmos, Ensino-aprendizagem, Serious game

1 Introdução

A utilização de jogos digitais no contexto educacional tem se mostrado uma abordagem muito eficaz, oferecendo uma maneira interativa e envolvente para transmitir conteúdos. A capacidade de jogos de capturar a atenção, estimular a motivação e promover a aprendizagem ativa se alinha com as necessidades contemporâneas de ensino. A aprendizagem baseada em jogos (*Game-Based-Learning*) traz uma combinação de elementos lúdicos e educacionais para criar experiências que divertem e também ensinam. A eficácia dessa abordagem é muito alta devido a capacidade de proporcionar *feedback* imediato, resolução de problemas e um ambiente supervisionado e experimental [2][8]. Esses aspectos são muito importantes em áreas de conhecimento que exigem noções e habilidades complexas, como algoritmos e programação, onde os conceitos são muito abstratos e a prática é essencial para a aprendizagem.

Além disso, há uma alta taxa de reprovação em disciplinas de algoritmos nos cursos de computação no ensino superior [4], o que reforça ainda mais a existência da lacuna no processo de aprendizado desses conteúdos, e também a necessidade de intervenção para baixar esses índices.

Diante disso, este artigo tem como objetivo combater as dificuldades enfrentadas por estudantes ao lidar com conceitos iniciais de algoritmos e programação. Para isso, propõe-se o desenvolvimento de um jogo educacional com o foco no ensino de algoritmos e programação, uma vez que este é uma ferramenta com alto potencial de valor pedagógico. O jogo desenvolvido e avaliado é o

"Viral_goritmos", que apresenta conceitos iniciais de algoritmos e programação para estudantes de graduação em Computação.

O restante deste artigo está organizado como se segue. Na Seção 2 é apresentado um estudo sobre as principais dificuldades no ensino de algoritmos e programação. Na Seção 3 o tema sobre jogos digitais na educação é apresentado, abordando tanto conceitos de aprendizagem baseada em jogos quanto formas de documentar e avaliar jogos. Na Seção 4 são apresentados alguns trabalhos relacionados que agregaram no desenvolvimento do jogo deste artigo. Na Seção 5 é detalhado como foi realizado o desenvolvimento do jogo, apresentando sua documentação e telas. Na Seção 6 são apresentados os resultados da avaliação do jogo desenvolvido. Por fim, na Seção 7 são descritas as considerações finais deste trabalho.

2 Dificuldades na aprendizagem de algoritmos e programação

As disciplinas de algoritmos e programação são fundamentais para dominar quase qualquer assunto da área de computação, porém, existem altas taxas de reprovação e desistência nos cursos e disciplinas de programação introdutória [4]. Entre os problemas que podem contribuir para este cenário estão:

- **Os estudantes não sabem resolver problemas ou não possuem habilidades sofisticadas de resolução de problemas:** falta habilidades de resolução de problemas, como entender o problema, relacionar conhecimentos, refletir e persistir;
- **Linguagens de programação possuem muitos detalhes e sintaxes complexas:** a maioria das linguagens de programação são desenvolvidas para uso profissional e não têm suporte para ensino didático, além de serem extensas em detalhes e complexidade;
- **Programação requer um alto nível de abstração:** a programação requer várias habilidades como abstração, generalização, pensamento crítico, etc., sobrecarregando os estudantes com altas demandas de diversas habilidades;
- **Muitos estudantes não possuem conhecimento lógico e matemático suficiente:** matemática e raciocínio lógico é uma base sólida para os conceitos de programação e algoritmos, então certamente a falta de domínio neles leva a uma grande dificuldade no aprendizado.

O principal problema encontrado no aprendizado de programação e algoritmos é a falta de domínio em resolver problemas, porém é importante considerar que o ato de programar ou desenvolver algoritmos requer um leque de habilidades específicas e complexas. Portanto é primordial abordar tanto a base da programação (resolução de problemas e matemática), mas também as habilidades específicas. [4]

3 Jogos digitais educacionais

O conhecimento sobre a existência de jogos digitais e seus aspectos vem crescendo nos últimos anos devido a sua incorporação natural nos assuntos e rotina da atualidade. Eles possuem um grande alcance, imersão e interatividade, visto que os mesmos, se desenvolvidos de forma em que seus elementos (narrativa, estética, mecânica e tecnologias) possuem uma grande harmonia entre si, podem gerar ótimos objetos de entretenimento [3].

A aprendizagem baseada em jogos digitais procura descrever aspectos relacionados à concepção, ao desenvolvimento e ao uso de jogos no processo de aprendizagem. Essa abordagem tem um grande potencial motivador, visto que a mesma é versátil, divertida e adaptativa, sendo possível até a combinação dela com outras estratégias pedagógicas [8]. Além disso, teorias recentes sobre aprendizagem efetiva sugerem que o aprendizado é mais efetivo quando ele é ativo, experimental, situado, baseado em problemas e com *feedbacks* imediatos [2], e jogos digitais parecem atender todas essas características.

3.1 Game Design Document (GDD)

O documento de projeto de jogo, *Game Design Document* (GDD), consiste em um documento de texto bem ilustrado e desenvolvido por um *game designer*, que descreve todos os principais elementos e características presentes em um jogo, como estética, premissa, cenários, personagens, mecânicas, narrativa, projeto de níveis, sons, etc. [7]. A função principal desse documento é guiar todas as pessoas envolvidas no desenvolvimento do jogo [7], além de também servir como uma base contratual para detalhar o jogo para investidores [1].

Não existe uma estrutura padrão para o GDD porém, é muito comum este documento ser muito extenso, visto que ele deve abordar cada elemento significativo do jogo. De acordo com o escopo do projeto e o detalhamento, o documento pode ter de dezenas a centenas de páginas [9].

3.2 Short Game Design Document (SGDD)

Em casos de jogos de pequeno porte, é muito comum que o tempo disponível não seja suficiente para o desenvolvimento de um documento tão detalhado e extenso como o GDD, porém, ainda é necessário a criação de um documento para fins de organização do projeto. Nesse sentido, foi elaborado um modelo de GDD para jogos de pequeno porte que visa a otimização de tempo gasto e detalhes documentados, chamado *Short Game Design Document* (SGDD).

A base do *design* do SGDD consiste em três itens [6]:

- **Encadeamento de informações:** os eventos do jogo seguem uma linearidade que acompanha a mecânica geral do jogo;
- **Listas:** criar várias listas de elementos do jogo (listas de personagens, itens, mecânicas, telas, etc.);
- **Telas:** descrições das telas do jogo, como telas iniciais, de comandos, história, vitória, derrota, etc.

O SGDD também define que, para uma melhor organização e compreensão do documento, ele deve ser elaborado de forma que o *designer* deve marcar o texto, separar e classificar cada elemento em sua respectiva área de atuação, normalmente sendo: arte/interface, música/sons e programação [6].

O objetivo principal do documento reduzido é continuar sendo útil para a etapa de desenvolvimento, organizando o projeto e guiando os desenvolvedores e membros envolvidos nele. Além disso, o SGDD deve descrever todo o jogo e seu enredo num texto corrido, preferencialmente em apenas uma página, marcar e classificar as características principais contidas no jogo, e ser capaz de ser utilizado em condições limitadas de desenvolvimento como *game jams* [6].

3.3 Modelo de Avaliação de Jogos Educacionais

A avaliação é uma etapa muito importante para o desenvolvimento de jogos educacionais, uma vez que é preciso saber se o jogo é divertido e atraente para os estudantes, além de atender os objetivos pedagógicos. Um modelo de avaliação de jogos educacionais foi proposto por Savi et al., baseado no modelo de avaliação de treinamento de Kirkpatrick, estratégias motivacionais do modelo ARCS de Keller, a área de estudo de UX e uma taxonomia de objetivos educacionais de Bloom [11].

Modelo de avaliação de treinamentos de Kirkpatrick: Consiste em um modelo para avaliar treinamentos baseado em quatro níveis. Segundo Kirkpatrick, cada nível tem sua importância e sua complexidade que aumenta gradativamente conforme se avança entre os níveis [11]. Os níveis são:

- (1) **Reação:** é medida a satisfação e valor percebido do treinamento pelos participantes;
- (2) **Aprendizagem:** levanta o quanto os participantes podem mudar de atitude, ampliar seus conhecimentos e/ou habilidades;
- (3) **Comportamento:** identifica o quanto os participantes mudaram seu comportamento em decorrência do que foi aprendido;
- (4) **Resultados:** identifica os ganhos obtidos com o treinamento.

O modelo de avaliação de jogos educacionais proposto utiliza apenas o primeiro nível deste modelo de Kirkpatrick (Reação) [11].

Modelo ARCS: O modelo foca na interação dos estudantes com os ambientes de aprendizagem, derivado da teoria expectativa-valor de Keller [11]. O modelo consiste em quatro categorias que formam o acrônimo ARCS, sendo elas:

- **Atenção:** respostas cognitivas dos estudantes aos estímulos instrucionais;
- **Relevância:** o nível de associação que os estudantes conseguem perceber entre seus conhecimentos prévios e as novas informações;
- **Confiança:** criar expectativas positivas para os estudantes, por experiências de sucesso alcançadas pelas próprias habilidades e esforço deles, tem bastante influência na persistência dos estudantes;
- **Satisfação:** gerar sentimentos positivos sobre a experiência de aprendizagem, com recompensas e reconhecimento, também é importante dar oportunidades para a aplicação do que foi aprendido o mais cedo possível.

User Experience (UX): A área de UX abrange a interação do indivíduo com o produto por inteiro, levando em consideração sentimentos, prazer, pensamentos, e diversas percepções e sensações humanas que vêm da interação. Os conceitos de UX utilizados no

modelo de Savi são: **imersão, interação social, desafio, diversão, controle e competência**[11].

Taxonomia de Bloom: Foi criada num contexto acadêmico por Benjamin Bloom, consiste em uma classificação de níveis de raciocínio em seis níveis, do mais simples até o mais complexo [11]. As categorias são:

- (1) **Conhecimento:** lembrar informações específicas como fatos, datas, palavras, métodos etc;
- (2) **Compreensão:** entender e captar a informação e aplicá-la em vários contextos;
- (3) **Aplicação:** aplicar o conhecimento;
- (4) **Análise:** identificar as partes e analisar o resultado;
- (5) **Síntese:** combinar e organizar as partes analisadas;
- (6) **Avaliação:** julgar o valor do conhecimento obtido.

O modelo de avaliação proposto abrange apenas os três primeiros níveis da Taxonomia de Bloom (Conhecimento, compreensão e aplicação) [11].

Baseado em seu repertório teórico, o objetivo do modelo de avaliação proposto por Savi et al. é definir uma estrutura para avaliar jogos educacionais. Desse modo, a avaliação pode ser realizada pela aplicação de um questionário com 27 questões para os usuários indicarem o quanto concordam com elas de acordo com a escala de Likert de 5 pontos variando de “discordo fortemente” até “concordo fortemente” [11].

4 Trabalhos relacionados

O jogo "ProGame" é um jogo educacional que busca ensinar conteúdos de algoritmos e programação. A ideia principal gira em torno da criação e edição de algoritmos, todos pré-definidos, visando testar habilidades essenciais para o aprendizado do conteúdo. O jogador deve passar pelos níveis e superar seus desafios, para então ser confrontado com a necessidade de resolver um algoritmo relacionado com algum elemento do nível [10]. Este jogo traz uma perspectiva interessante de progressão de níveis e resolver quebra-cabeças de forma sequencial e com dificuldade gradual ao fim de cada nível, conceito que foi estudado e implementado de forma adaptada no jogo digital elaborado neste artigo, abordando conteúdos relacionados de algoritmos e programação, porém de uma forma mais complexa.

Outro jogo é o "Program your Robot", um *serious game* criado para permitir estudantes a praticarem os conceitos introdutórios de programação, também auxiliando na prática de pensamento computacional. O objetivo do jogo é auxiliar um robô a escapar de uma série de plataformas construindo um plano de escape chamado algoritmo de solução, montando-o com vários comandos de programação separados em forma de ações e blocos de código, para então o robô executar o algoritmo. Com 6 fases no jogo, cada uma aborda um problema diferente, e mantendo uma dificuldade gradual com um aumento no tamanho das plataformas e sua complexidade geral [5]. Uma característica muito interessante deste jogo é a ideia de utilizar blocos para "programar", utilizando o paradigma de programação orientada a blocos, muito utilizado no ensino introdutório de lógica de programação, algo que ajuda na abstração dos conteúdos de algoritmos e programação. Essa característica foi utilizada como inspiração de alguns *puzzles* de "Viral_goritos", porém adaptados

para mesclar o conceito de programação em blocos com códigos mais próximos de linguagens de programação comuns.

O jogo *Super Mario* (de Árvores AVL) é um jogo educacional focado em ensinar sobre árvores AVL, um conteúdo mais avançado da disciplina de Algoritmos e Estruturas de Dados. O jogo toma como inspiração em uma boa parte do seu *design* geral o aclamado jogo digital "*Super Mario Bros*", porém, o objetivo principal neste é manipular árvores AVL, abordando conceitos de busca, rotação, inserção e remoção de elementos em árvores AVL, além de árvores binárias de busca. O jogo mantém uma história simples porém efetiva para conectar e relacionar os objetivos do personagem com o jogo em si [12]. Este jogo foi o pontapé inicial na idealização do jogo digital "Viral_goritos", por causa da sua abordagem em conceitos mais complexos de algoritmos e programação. Resultando na ideia de desenvolver um jogo de dificuldade intermediária, que trabalha com conteúdos iniciais de algoritmos e programação, porém com uma abordagem mais avançada. Diante disso, viu-se a possibilidade de desenvolver um jogo inspirado nessa ideia, dando origem a idealização e elaboração do jogo educacional "Viral_goritos".

5 Desenvolvimento do jogo Viral_goritos

O jogo "Viral_goritos" foi desenvolvido com o objetivo de ensinar conceitos de algoritmos e programação para estudantes do ensino superior. O jogo é uma ferramenta educacional que auxilia a fixação de conceitos abstratos de programação, como estruturas condicionais, de repetição e estruturas de dados homogêneas, de uma forma interativa e visual. O jogo pode ser jogado em qualquer computador que possua os dispositivos de entrada teclado e *mouse*.

5.1 Ferramentas utilizadas

A ferramenta utilizada para o desenvolvimento e geração de *software* executável do jogo foi o motor gráfico "Unity"¹, um *software* de desenvolvimento de jogos 2D e 3D que utiliza a linguagem de programação C# e possui bibliotecas e ferramentas que auxiliam na criação de jogos digitais.

As artes visuais do jogo foram todas criadas utilizando o "Piskel"², um editor de imagens focado em *Pixel Art* gratuito.

Os áudios do jogo foram criados utilizando uma versão gratuita do *software* de criação de músicas focado em áudios *chiptunes* chamado "Deflemask"³.

A linguagem de programação utilizada como base para elaboração dos algoritmos e códigos presentes como parte pedagógica do jogo, foi a linguagem "Portugol", utilizada no ambiente de desenvolvimento integrado (IDE) "Portugol Studio"⁴, essa linguagem e IDE foram criadas com o objetivo de aliviar os problemas que existiam no aprendizado de estudantes iniciantes em conceitos de algoritmos e programação, mitigando a necessidade de compreender o idioma inglês e facilitando a visualização dos códigos, definindo-os com uma linguagem mais próxima do português, bem como manter a ferramenta fácil de usar e simples de entender e se localizar, visando a acessibilidade para iniciantes na área.

¹Unity, disponível em: <https://unity.com/pt>, acesso em outubro de 2024

²Piskel, disponível em: <https://www.piskelapp.com/>, acesso em outubro de 2024

³Deflemask, disponível em: <https://www.deflemask.com/>, acesso em outubro de 2024

⁴Portugol Studio, disponível em: <https://univali-lite.github.io/Portugol-Studio/>, acesso em outubro de 2024

Todos os *assets* (artes, *sprites*, e sons) do jogo foram criados pelo próprio desenvolvedor.

5.2 Metodologia

O desenvolvimento do jogo "Viralgoritmos" começou com a idealização e prototipação do jogo, pela elaboração do *Short Game Design Document* (SGDD) para orientação e organização dos seus requisitos e funcionalidades. A criação do SGDD fornece um resumo dos elementos principais do jogo, como a história, jogabilidade, som, arte, e fluxo do jogo, e também auxilia bastante na orientação e foco no desenvolvimento.

Na Figura 1 são apresentados o enredo base do jogo e o fluxo do jogo descrito em um texto corrido tudo conforme os critérios de elaboração do SGDD.

História

O computador pessoal de uma estudante do ensino superior chamada Gabriela foi infectado por um vírus de computador e ela precisa recuperar o controle e restaurar o funcionamento da sua máquina, para isso, é executado um processo do programa de antivírus instalado no computador. Ao executá-lo, um agente é enviado para dentro do dispositivo para combater as ameaças e restaurar as partes da máquina que foram infectadas.

Jogo

O jogo começa com a **tela** de **menu** e **botões** de **Jogar**, **Controles**, **Créditos** e **Fechar**. Quando se clica no botão de jogar, o jogo se inicia com o **personagem** que pode ser **controlado** em uma **sala** cheia de **circuitos** [**música tema de fundo**], como uma representação do interior de um computador, neste espaço também há o **"Staply"** (personagem não-jogável), que o personagem pode **interagir** uma **mesa** com **armas** que podem ser **equipadas** [**som de interação**], uma **porta** fechada, e um **núcleo** da sala (semelhante a um processador) que pode ser **interagido** pelo personagem. As armas são um sabre de luz que possui um ataque à curta distância [**som de sabre**], e um "Blaster" que atira lasers à longa distância [**som de laser**]. Quando o Agente **interage** com o núcleo, uma **janela** é aberta, contendo **códigos** escritos em Português com algumas linhas e **espaços** vazios, **blocos** **arrastáveis** com pedaços de código, **botão** de **fechar** a janela e **botão** de **executar** código. Ao completar o código com os blocos, o Agente pode **executar** o código, caso o código esteja incorreto, o Staply **avisará** o Agente que a tentativa foi falha, e **apresentará dicas** em formato de **diálogo** para auxiliar a solução do problema, caso o código esteja correto, o "quebra-cabeça" é **resolvido**, um **feedback** é **apresentado** em formato de **diálogo** do Staply e a **porta** para a próxima sala é **aberta**. Nas salas seguintes, existem **inimigos** que podem **atacar** [**som de ataque**] o personagem e **diminuir** sua vida, caso o personagem chegue a zerar sua vida, uma nova **tela** é apresentada contendo um **texto** escrito "Game Over" e **botões** para **voltar ao menu**, ou **tentar novamente**. O jogo consiste em várias **salas** com diversos "quebra-cabeças" e **inimigos**, cada sala possui um "quebra-cabeça" diferente, como alternativas, arrastáveis, ou até preenchimento de matrizes, que é resolvido ao **interagir** com o **núcleo** da sala e as **janelas** que são **abertas**. A dificuldade dos quebra-cabeças e inimigos aumenta gradualmente até chegar na última sala, que há uma batalha contra o Dot, um **chefão** [**música tema do chefe**] que possui um **ataque** corpo a corpo [**som de ataque do chefe**] e um "quebra-cabeça", onde o Agente deve **digitar** as partes de código que estão faltando. Quando a vida do Dot **diminui** pela metade, ele **entra** em um novo estágio, status **aumentados**, além de um novo **ataque** à distância [**som de segundo ataque do chefe**]. Ao **derrotar** o Dot, o Staply **aparece** na sala, apresentando **diálogos** de congratulações e logo depois **surge** um **portal** na sala, que quando é **interagido**, a **tela** de "Jogo concluído" é apresentada, com um **texto** de agradecimento e um **botão** para **retornar** ao menu principal.

Figura 1: História e fluxo do jogo descritos no SGDD.

Na Figura 2 são apresentados os componentes identificados no texto corrido do fluxo do jogo. O texto corrido é estruturado seguindo as regras do SGDD, com os componentes marcados com diferentes cores para posterior identificação, os componentes sendo: **arte** (amarelo), **programação** (azul) e **áudio** (rosa).

Programação

- Menu (interface) e botões [jogar, controles, créditos, fechar, voltar ao menu];
- Controle do personagem [movimentação, ataque];
- Interação com objetos do mundo [coletar armas, núcleo];
- Interação com elementos da interface dos puzzles [blocos, botões, espaços];
- Diálogos;
- *Puzzles*;
- Inimigos [ataques e comportamento].

Arte

- Telas [menu principal, janelas dos *puzzles*, "game over", jogo concluído];
- Botões;
- Personagens [Agente, Staply, Inimigos, Dot];
- Salas e objetos [cenário, núcleo, portas, portal];
- Armas/ferramentas de combate;
- Textos e arte dos diálogos;

Áudio

- Música tema de fundo;
- Efeitos Sonoros [interação, ataques];
- Efeitos sonoros das armas [sabre e laser];
- Música tema do chefe [normal e alterada].

Figura 2: Arte, programação e sons descritos no SGDD.

O jogo consiste em um mapa com diferentes salas que possuem tanto inimigos para derrotar quanto seções de quebra-cabeça para resolver. Cada sala possui um quebra-cabeça único que aborda algum conceito de algoritmos e programação, como condicionais, laços de repetição, variáveis compostas homogêneas, etc. O objetivo do jogo é fornecer um ambiente onde o jogador pode aplicar e visualizar os conceitos ao tentar resolver os *puzzles*, de forma que auxilie o jogador a fixar esses conceitos.

Na Figura 3 é apresentado um diagrama que representa o fluxo do jogo, apresentando os estados principais durante uma jogatina média. O jogo é separado por salas que possuem núcleos representando os *puzzles* do jogo e seções de combate contra inimigos, cada núcleo aborda um conteúdo e ao ser resolvido libera a próxima sala. Após algumas salas o jogador se encontra com um "chefão" que possui dois estágios de combate e uma seção de *puzzle*, ao derrotar o "chefão" o jogo termina.

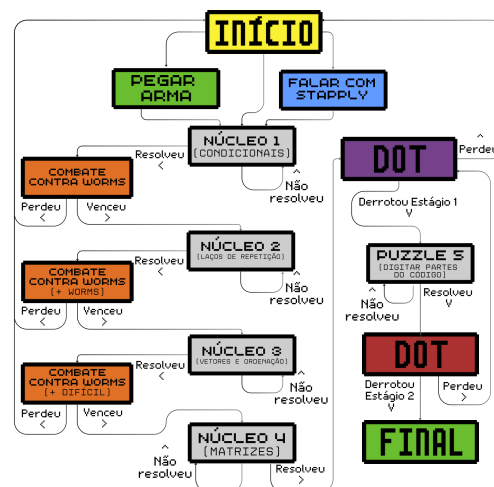


Figura 3: Diagrama do fluxo do jogo.

Na Figura 4 é apresentada a tela inicial do jogo "Viral_goritmos", com um título estilizado, botões para navegação e algumas artes.



Figura 4: Tela inicial do jogo "Viral_goritmos".

Na Figura 5 é apresentado um momento de diálogo inicial do personagem principal com um personagem não jogável (NPC) chamado "Stapply", que acompanha e auxilia o jogador durante toda a jornada.



Figura 5: Tela de momento de diálogo.

A Figura 6, apresenta a interface de um dos *puzzles* do jogo, onde o jogador deve arrastar os códigos corretos para os espaços.



Figura 6: Tela de quebra-cabeça ou (*puzzle*).

Na Figura 7 é apresentada uma captura de tela do jogo no momento em que o jogador entrou em uma sala nova, com os inimigos percebendo sua presença e se locomovendo para a sua direção para atacá-lo.



Figura 7: Tela de combate contra inimigos.

O jogo pode ser encontrado para *download* no Google Drive⁵ para ser jogado de maneira *offline* instalado na máquina, ou de maneira *online* no site do grupo de pesquisa Ponte⁶ (requer *software* de navegador executado em um computador com teclado e *mouse*).

6 Resultados

A avaliação do jogo digital Viral_goritmos foi realizada utilizando o modelo de avaliação de jogos educacionais proposto por Savi et al. [11], através de questionários de auto avaliação, utilizando um formulário hospedado no Google Forms como forma de coleta dos *feedbacks* dos estudantes.

Neste modelo de avaliação são utilizadas 24 afirmações classificadas em categorias como motivação, experiência do usuário e aprendizagem (afirmações da categoria de interação social foram desconsideradas), todas utilizando como estrutura a escala Likert de 5 pontos: desde "Discordo fortemente" até "Concordo fortemente". [11] Também foram utilizadas duas perguntas abertas adicionais para maior liberdade dos estudantes expressarem suas opiniões sobre o jogo. Estudantes do ensino superior de cursos de computação, que não possuem vínculos com o desenvolvimento do jogo digital, compõem o grupo em que foi conduzida a coleta dos *feedbacks*. No total foram obtidas 14 respostas.

Para assegurar a integridade ética da pesquisa, foram adotados procedimentos de conduta ética e respeito aos direitos dos participantes em todas as etapas do estudo. Utilizou-se um formulário eletrônico anônimo para a coleta de dados, garantindo a confidencialidade e a privacidade dos participantes. Antes de dar início à pesquisa, todos os estudantes envolvidos foram adequadamente informados sobre os objetivos, procedimentos e possíveis impactos do estudo, recebendo o consentimento livre e esclarecido. A participação foi voluntária.

⁵Google Drive, disponível em: <https://bit.ly/viral-goritmos-v1a>

⁶Viral_goritmos, disponível em: https://ponteducacional.com.br/projetos/viral_goritmos/

6.1 Motivação

As Figuras 8, 9, 10 e 11, compõem as respostas do questionário avaliativo referentes à dimensão de motivação. Os gráficos de barra 100% preenchidos estão distribuídos e divididos de acordo com cada sub-categoria dentro das dimensões, com cores diferentes para cada nível da resposta submetida por cada estudante.

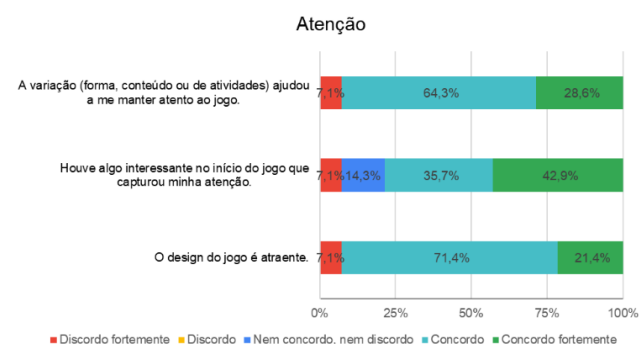


Figura 8: Motivação - Questões de atenção.

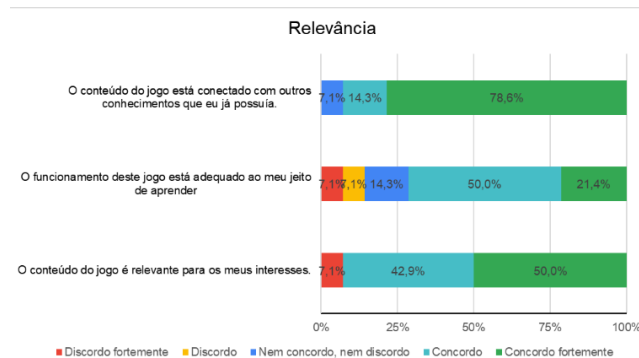


Figura 9: Motivação - Questões de relevância.

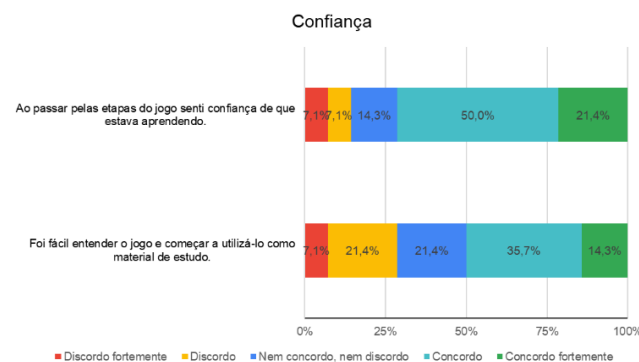


Figura 10: Motivação - Questões de confiança.

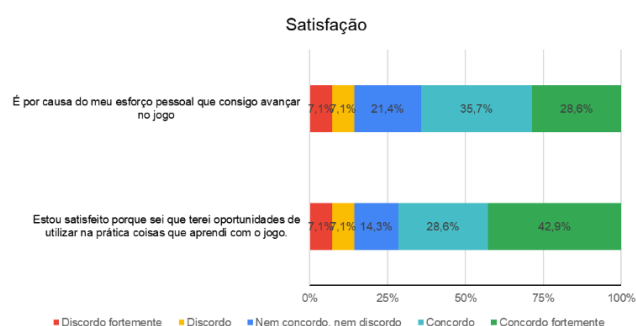


Figura 11: Motivação - Questões de satisfação.

A maioria das respostas desta dimensão mostram que o jogo foi capaz capturar bem a atenção dos estudantes, e que o conteúdo do jogo é relevante para seus interesses, como é mostrado nas Figuras 8 e 9, com mais de 50% das respostas positivas em todas as afirmativas. Portanto, é evidente que o jogo atende esses requisitos e é relevante para o ensino de algoritmos e programação.

6.2 User Experience (UX)

Nas Figuras 12, 13, 14 e 15, é abordada a dimensão da Experiência do Usuário.

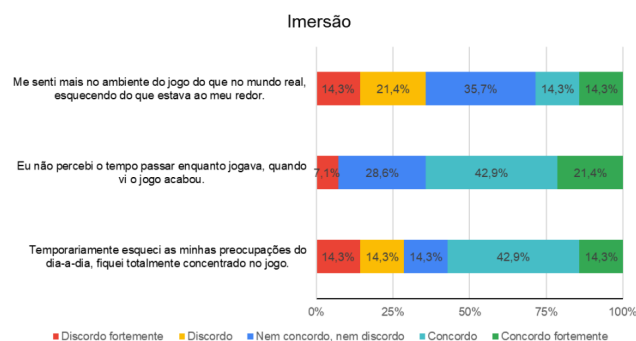


Figura 12: UX - Questões de imersão.

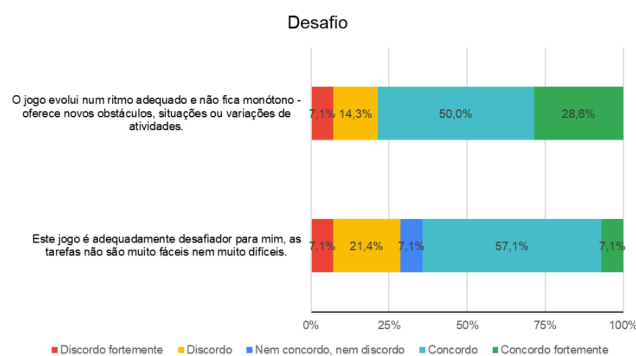


Figura 13: UX - Questões de desafio.

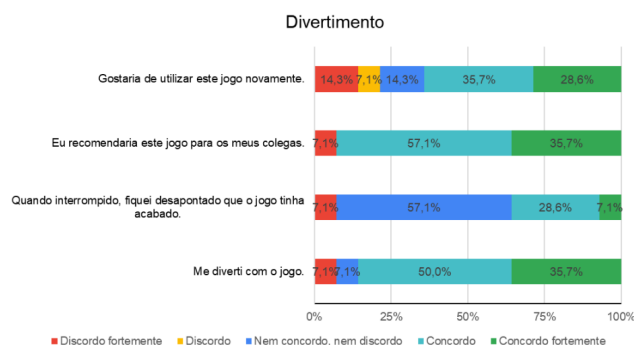


Figura 14: UX - Questões de divertimento.

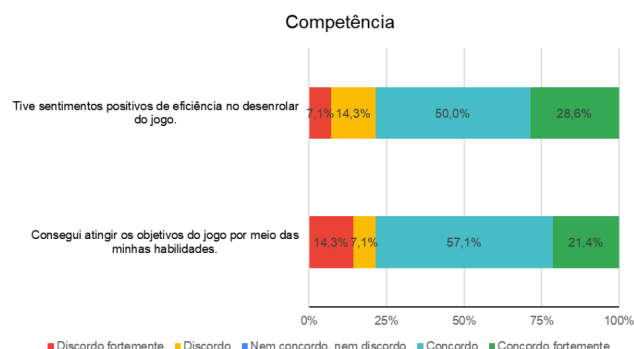


Figura 15: UX - Questões de competência.

Os resultados da dimensão de Experiência do Usuário apresentam que a experiência geral que os estudantes tiveram com o jogo foi agradável, com grande parte de afirmações como "Eu recomendaria este jogo para os meus colegas" ou "Tive sentimentos positivos de eficiência no desenrolar do jogo" havendo mais de 70% das respostas submetidas em níveis como "Concordo" ou acima. Inclusive, mais de 85% das respostas foram positivas na afirmativa "Me diverti com o jogo", o que reforça mais ainda a ideia de que jogar este jogo educacional forneceu uma experiência divertida e satisfatória.

6.3 Aprendizagem

Na Figura 16 é detalhada a dimensão relacionada com o fator de aprendizagem do jogo.

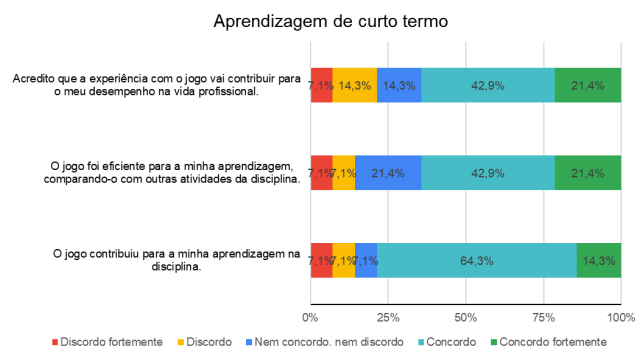


Figura 16: Aprendizagem - Questões de aprendizagem de curto termo.

Seus dados apresentam que mais de 60% das respostas sobre o potencial de ensino do jogo foram positivas, e também mais de 70% das respostas da afirmação "O jogo contribuiu para a minha aprendizagem na disciplina" foram "Concordo" ou acima, ou seja, positivas. Diante disso, é evidente que, para a grande maioria dos estudantes, o jogo contribuiu e foi eficiente na tarefa de potencializar a aprendizagem do conteúdo, confirmando sua capacidade de ensinar os conceitos abordados pelo mesmo.

As questões dissertativas que perguntavam a opinião do jogador sobre o jogo no geral, considerando a jogabilidade, história, gráficos e desafios, bem como sugestões, críticas e comentários no geral, obtiveram respostas críticas e ao mesmo tempo muito positivas. Um dos estudantes que avaliou o jogo cita: "A proposta é muito boa, gostei muito da ideia, as fases são bem desafiadoras e força a pessoa a entrar no ambiente do jogo.", ou seja, este jogo educacional possui uma proposta concisa e é capaz de fazer o jogador ficar imerso no ambiente do jogo. Outro estudante aponta que: "Achei o jogo bem cativante principalmente em relação aos seus desafios e sua jogabilidade.", apresentando assim uma coerência encontrada pelos estudantes diante os quebra-cabeças elaborados no jogo digital.

Outra resposta de um estudante que realizou a avaliação foi: "[...] Encontrei apenas um bug durante o jogo. Algumas coisas não ficaram muito claras e outras poderiam ser melhores para melhorar a experiência do jogador, como por exemplo na parte de laços e matriz, poderia limpar o desenho que escolhemos para que na próxima fase o jogador não precise fazer isso.". Ou seja, houveram tanto sugestões de pequenos ajustes de "qualidade de vida", quanto alertas de bugs encontrados no jogo, após a seção de avaliação, essas sugestões foram atendidas para enriquecimento e aprimoramento do objeto de aprendizagem como um todo.

7 Considerações finais

Este trabalho descreveu e resumiu todo o processo de desenvolvimento e avaliação do jogo educacional "Viral goritos", criado com o objetivo de educar e auxiliar estudantes sobre conceitos abstratos de algoritmos e programação. A relevância deste jogo educacional foi confirmada a partir das avaliações conduzidas, pois a forma ativa de interação, *feedbacks* imediatos e os elementos visuais de jogos digitais são eficientes e respondem bem aos conteúdos conceituais

de algoritmos e programação, apresentando um grande potencial educacional.

Como trabalhos futuros pretende-se aprimorar a compatibilidade do jogo “Viralgoritmos” com mais dispositivos, como *mobiles*, bem como atualizá-lo para possuir mais salas abordando outros aspectos dos conteúdos de algoritmos e programação, assim expandindo seu potencial educacional. Além disso, também pretende-se utilizar o jogo digital desenvolvido neste trabalho como objeto de aprendizagem em cursos de algoritmos e programação, visando auxiliar estudantes que estejam nesta etapa de aprendizado dos conceitos abordados.

Agradecimentos

Agradecimentos especiais para a Fundação de Apoio ao Desenvolvimento do Ensino, Ciência e Tecnologia do Estado de Mato Grosso do Sul (FUNDECT), pelo apoio e financiamento do projeto por meio do Programa Institucional de Bolsas de Iniciação Científica (PIBIC).

Referências

- [1] Ernest Adams. 2007. The Designer’s Notebook: Why Design Documents Matter. *Gamasutra [online]* (2007).
- [2] Thomas M Connolly, Elizabeth A Boyle, Ewan MacArthur, Thomas Hainey, and James M Boyle. 2012. A systematic literature review of empirical evidence on computer games and serious games. *Computers & education* 59, 2 (2012), 661–686.
- [3] Dulce Márcia Cruz and Daniele Karine Ramos. 2019. Games e formação docente. *Informática na Educação: Pensamento computacional, robótica e coisas inteligentes* (2019), 1–26.
- [4] Anabela Gomes and António José Mendes. 2007. Learning to program-difficulties and solutions. In *International Conference on Engineering Education–ICEE*, Vol. 7. 1–5.
- [5] Cagin Kazimoglu, Mary Kiernan, Liz Bacon, and Lachlan Mackinnon. 2012. A serious game for developing computational thinking and learning introductory computer programming. *Procedia-Social and Behavioral Sciences* 47 (2012), 1991–1999.
- [6] Rodrigo L Motta and José Trigueiro Junior. 2013. Short game design document (SGDD). *Proceedings of SBGames 2013* (2013), 115–121.
- [7] Roger E Pedersen. 2003. *Game design foundations*. Wordware Publishing, Inc.
- [8] Marc Prensky. 2021. *Aprendizagem baseada em jogos digitais*. Editora Senac São Paulo.
- [9] Tim Ryan. 2009. Learning the ways of the game development wiki. *Game Developer*. <https://www.gamedeveloper.com/design/learning-the-ways-of-the-game-development-wiki> (2009).
- [10] Chrystian Gesteira Sales and Vanessa Farias Dantas. 2010. ProGame: um jogo para o ensino de algoritmos e programação. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*, Vol. 1.
- [11] Rafael Savi, Christiane Gresse von Wangenheim, and Adriano Ferreti Borgatto. 2011. A model for the evaluation of educational games for teaching software engineering. In *2011 25Th brazilian symposium on software engineering*. IEEE, 194–203.
- [12] Debabi Wassila and Bensebaa Tahar. 2012. Using serious game to simplify algorithm learning. In *International Conference on Education and e-Learning Innovations*. IEEE, 1–5.